INSTITUTO TECNOLÓGICO SUPERIOR DE TEZIUTLÁN

Tesis

"DESARROLLO DE APLICACIÓN WEB PARA LA OPTIMIZACIÓN DE INVENTARIOS Y GESTIÓN DE STOCK DE PRODUCTOS ORGÁNICOS DEL COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT)"

PRESENTA:

ERICK DE JESÚS HERNÁNDEZ TOLEDANO

CON NÚMERO DE CONTROL

19TE0561

PARA OBTENER EL TÍTULO DE:

INGENIERO EN SISTEMAS COMPUTACIONALES

CLAVE DEL PROGRAMA ACADÉMICO

ISIC-2010-224

DIRECTOR (A) DE TESIS:

MARÍA EUGENIA CARREÓN ROMERO

"La Juventud de hoy, Tecnología del Mañana"

TEZIUTLÁN, PUEBLA, ENERO 2024

Sub_Aca 05b

PRELIMINARES

Agradecimientos

A MI FAMILIA.

A quienes les debo mi agradecimiento eterno.
Gracias por su inquebrantable apoyo,
amor y comprensión a lo largo de este proyecto.
Ustedes son la razón detrás de mi motivación
y determinación.

A MIS AMIGOS Y COLEGAS.

Por su inquebrantable apoyo
y aliento durante los momentos
más desafiantes de este viaje.
Su amistad y confianza fueron un
bálsamo que alivió las dificultades.

AL GRUPO CPMT.

Por su colaboración fundamental
en la realización de esta investigación.
Su disposición y participación activa
hicieron posible el éxito de este proyecto.

A MI ASESORA, LA MTRA. MARÍA EUGENIA.

Expreso mi sincero agradecimiento
por su invaluable guía, paciencia
y conocimientos. Su orientación
y apoyo fueron esenciales para
dar forma a este estudio.

Resumen

La gestión adecuada del inventario es esencial para cualquier empresa, ya que tiene un impacto directo en su eficiencia operativa. Un manejo eficaz del stock garantiza que se tenga información del inventario y de algunas otras cuestiones relacionadas al mismo, ayudando en la toma de decisiones de la empresa y en la mejora de sus operaciones.

El presente trabajo de tesis se enfoca en el desarrollo de una aplicación web destinada a optimizar la gestión de inventarios y stock de manzana orgánica en el Colectivo de Productores de Manzana Teziutlán (CPMT). La problemática central radica en la ineficiencia en la administración de inventarios, lo cual conlleva a dificultades para llevar un control de su stock de productos y un desconocimiento de la cantidad que sale debido a ventas.

Para abordar esta problemática, se ha diseñado y desarrollado una aplicación web personalizada que emplea tecnologías como una base de datos robusta con MySQL y PHP con Slim para manejar dicha base de datos, con los cuales se puede tener un funcionamiento que permita tener una idea del stock de productos de diversas maneras, e información del stock que sale en cada venta, además de funcionalidades adicionales en la parte de ventas centradas en el inventario.

Los resultados obtenidos arrojaron mejoras sustanciales en la precisión de las proyecciones de demanda al verificar las cantidades que se mueven de cada manzana debido a las ventas, también, un manejo más sencillo de las operaciones del colectivo en la parte de inventarios.

Introducción

Según la investigación de Corella Parra, L. M., & Olea Miranda, J. (2023), la saturación en el mercado, donde se ofrecen servicios o bienes idénticos, deja a las empresas en una posición vulnerable si no logran adaptarse a las fluctuaciones en la demanda o destacar con un valor diferenciador que las distinga de la competencia. La presencia de desafíos organizativos en el almacén se manifiesta en la prolongación de los tiempos de atención al cliente, agotamiento, exceso de inventario, pérdida de productos y discrepancias en los registros.

Por su parte para González, A. (2020), la administración de inventarios dentro de una empresa se presenta como una actividad íntimamente ligada a la cadena de valor de la organización, debiendo ajustarse a la estrategia y tácticas de la empresa para asegurar la satisfacción de los clientes.

La ineficiencia en la administración de inventarios, sumada al desconocimiento de las existencias y registros de las cantidades que se mueven dentro de las ventas, plantea una problemática que va más allá de los muros del "Colectivo de Productores de Manzana Teziutlán (CPMT)". El objetivo de este estudio es desarrollar una aplicación web buscando soluciones que no solo optimicen la gestión de inventarios y stock del colectivo de productores de manzana Teziutlán, sino que ayude en su operatividad de diversas maneras, la aplicación web cuenta con el manejo de stock de diversos tipos de manzanas orgánicas, donde el administrador puede estar al tanto de su inventario de productos y visualización de ventas de los mismos con información que le puede ayudar en la toma de decisiones estratégicas, así como ayudarle en la gestión de diversas cuestiones del colectivo y de su logística, proporcionando un mejor manejo del mismo. En un entorno donde las pérdidas económicas y la incertidumbre sobre la disponibilidad de producto se han convertido en obstáculos significativos, esta investigación se propone ofrecer respuestas sólidas que puedan ayudar a generar estrategias.

Para dar inicio con el Capítulo I, donde se establecen las bases del estudio, describiendo el contexto de la empresa y el estado del arte, los problemas de investigación, las preguntas planteadas, y los objetivos generales y específicos. Se

justifica la investigación en función de su relevancia para el CPMT. En el Capítulo II, se proporciona una sólida base teórica que consigue respaldar los fundamentos que sustentan la revisión documental dentro del marco del proyecto de investigación. El Capítulo III detalla el procedimiento y las actividades realizadas, incluyendo el alcance y enfoque de la investigación, hipótesis, diseño y metodología, selección de muestra, recolección y análisis de datos. El Capítulo IV presenta los resultados obtenidos que incluyen capturas de pantalla del funcionamiento de la aplicación en el aspecto del manejo de inventario del colectivo. Las Conclusiones en el Capítulo V resaltan los hallazgos generales y específicos, así como las recomendaciones y aportaciones originales. Se reconocen las limitaciones y se ofrecen recomendaciones para futuras investigaciones. El Capítulo VI identifica y describe las competencias desarrolladas durante el proyecto, como una reflexión de las fortalezas y áreas de mejora del estudiante en su formación continua. El Capítulo VII enumera las fuentes de información consultadas durante la investigación. Finalmente, el Capítulo VIII presenta los anexos que complementan el proyecto donde se incluye un índice de imágenes que facilita la navegación y comprensión del trabajo.

Índice

CAPÍTULO I

GENERALIDADES DEL PROYECTO

1.1 Datos generales de la empresa

La tesis aborda un proyecto desarrollado dentro del "Instituto Tecnológico Superior de Teziutlán (ITST) ", es una destacada institución de educación superior ubicada en el hermoso municipio de Teziutlán, en el Estado de Puebla, ha consolidado su reputación gracias a su compromiso con la excelencia académica. En la actualidad, el ITST ofrece un abanico de opciones educativas, incluyendo 6 licenciaturas, cuyo reconocimiento a nivel internacional refleja el arduo trabajo y dedicación que la institución ha invertido en la búsqueda de estándares de calidad educativa sobresalientes.

Como parte de su continua búsqueda de la excelencia, el Instituto Tecnológico Superior de Teziutlán ha establecido, implementado, mantenido y mejorado de manera constante su Sistema de Gestión de Calidad, siguiendo rigurosamente los requisitos de la Norma ISO 9001:2015. Esto garantiza que tanto los estudiantes como el personal académico y administrativo de la institución se beneficien de un ambiente de aprendizaje óptimo y de procesos efectivos que promueven la calidad en cada aspecto de su labor educativa.

Dentro de este contexto de búsqueda de calidad, el ITST ha decidido enfocar sus esfuerzos en la mejora y el desarrollo de la propuesta educativa en el campo de Ingeniería en Sistemas Computacionales. Este enfoque demuestra su compromiso con la evolución constante y su voluntad de satisfacer las necesidades cambiantes de la industria y los estudiantes en esta disciplina en particular. Con esta decisión, el Instituto Tecnológico Superior de Teziutlán continúa su trayectoria de contribuir a la formación de profesionales altamente capacitados y preparados para afrontar los retos de la sociedad y la industria en el siglo XXI.

Por su parte el Colectivo de Productores de Manzana Teziutlán (CPMT) es quien se verá beneficiado con el desarrollo del proyecto. El CPMT se sitúa en la pintoresca región de Teziutlán ubicada en la parte sur, conocida como Loma Bonita. Los

productores a pesar de enfrentar restricciones de recursos y seguir métodos de producción convencionales han logrado estimular la producción de manzanas en la región, destacándose por su enfoque orgánico durante el proceso de producción. Estas prácticas empíricas han recibido reconocimiento por parte de la Secretaría de Turismo del Estado de Puebla.

El colectivo surge como respuesta a la necesidad de consolidar y potenciar las actividades de los productores locales, la iniciativa se erige en el corazón de una comunidad comprometida con la producción sostenible y la preservación de las prácticas agrícolas tradicionales.

La empresa nace de la colaboración y esfuerzo conjunto de productores comprometidos con la calidad, la sostenibilidad y el desarrollo económico de la región. Desde su fundación, el CPMT ha buscado destacar la excelencia en la producción de manzanas orgánicas, estableciendo estándares que van más allá de lo comercial para abrazar valores medioambientales y sociales.

El enfoque principal del CPMT abarca el ámbito de la producción agrícola, con especial énfasis en la calidad de procesos, control de calidad, comercialización y promoción de productos orgánicos. Su importancia radica en la contribución al desarrollo económico local, la preservación del medio ambiente y la oferta de productos de alta calidad en el mercado.

1.2 Problema de investigación

El reto principal que afronta el Colectivo de Productores de Manzana Teziutlán (CPMT) se centra en la gestión de inventarios y la comercialización de su manzana orgánica. En la actualidad, la falta de herramientas adecuadas para llevar un control preciso del stock disponible representa un obstáculo significativo. Esta limitación impacta directamente en la toma de decisiones y, en última instancia, en los ingresos de los productores.

A lo largo del tiempo, la práctica cultural de contar con intermediarios en la comercialización ha persistido, lo que ha dado lugar a la falta de visibilidad y control por parte de los productores sobre sus precios de mercado. Este escenario dificulta aún más la gestión eficiente de sus recursos; paralelamente, las mejoras en las técnicas de producción han aumentado la cantidad de manzanas orgánicas disponibles año tras año.

Por consiguiente, es esencial contar con herramientas y procesos de comercialización que permitan al CPMT ser más eficiente en la gestión de inventarios, esta información será manejada por el usuario con el rol de administrador, quien actualizará dichos datos con base en la producción notificada por los productores. La creación de una plataforma digital interna que ofrezca una visión clara del stock disponible se presenta como una solución crucial. Al mismo tiempo, es fundamental mejorar la presencia en línea del CPMT para atraer a más posibles compradores y mejorar las ventas.

La problemática radica en la falta de acceso a herramientas tecnológicas que faciliten una gestión interna de inventarios más efectiva, lo que a su vez limita la comercialización de la manzana orgánica producida por el colectivo.

Factibilidad:

- Recursos.

  Humanos: Conocimientos en la creación y manejo base datos, así como en experiencia de usuarios (UX), también se debe contar con el conocimiento en diversos lenguajes de programación como PHP, y librerías como SLIM, además del conocimiento en el manejo de una API.

  Materiales: Servidor de alojamiento de la aplicación y la base de datos en función a los requerimientos del cliente.

- Hardware

    - Computadoras.
- Software
    - Visual Studio Code.
    - MySQL.
    - PHP
    - Navegadores de internet.
- Servicios
    - Servicio de servidores para alojar la aplicación.
    - Leaflet.

1.2 Preguntas de investigación

- ¿La implementación de una aplicación web para la gestión de inventario contribuirá en la eficiencia de la gestión de stock de manzana orgánica producida por el Colectivo de Productores de Manzana Teziutlán (CPMT)?

1.3 Objetivos

Objetivo General

Optimizar la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán (CPMT) mediante el desarrollo de una aplicación web, con el fin de agilizar el control de inventarios y garantizar la disponibilidad adecuada de productos para satisfacer la demanda de los clientes.

Objetivos Específicos

• Desarrollar una aplicación web intuitiva y amigable que permita a los productores del CPMT gestionar de una manera eficiente los inventarios.

• Integrar un sistema de gestión de inventario para asegurar que la información de los productos disponibles en la aplicación web esté actualizada y disponible para los clientes.

• Capacitar al personal involucrado en el manejo de inventarios para garantizar el correcto uso de la aplicación web y solucionar las posibles dudas de su uso.

1.4 Justificación

La optimización de la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán (CPMT) mediante el desarrollo de una aplicación web presenta una serie de razones fundamentales que justifican su realización. En primer lugar, se pretende demostrar que la implementación de esta aplicación web no solo simplificará la gestión de inventarios, sino que también mejorará la eficiencia y la productividad de los productores. La magnitud del problema radica en la complejidad actual de gestionar el inventario para conocer en todo momento la disponibilidad de las manzanas para venta y distribución a clientes locales y nacionales; lo que puede generar pérdidas significativas y la insatisfacción de los clientes. La importancia de esta investigación se refleja en su repercusión directa en la economía del Colectivo de Productores de Manzana Teziutlán (CPMT), ya que este colectivo desempeña un papel fundamental en la producción de manzana orgánica en la región. Una gestión de inventarios más efectiva no solo beneficia a los productores al reducir pérdidas, sino que también puede contribuir al crecimiento económico del propio colectivo, generando mayores ingresos y fortaleciendo su posición en el mercado. Además, la implementación de esta aplicación web tiene un impacto social importante, ya que brinda una solución tecnológica a los productores y les permite competir de manera más efectiva en el mercado. Al mismo tiempo, la capacitación del personal en el manejo de la aplicación garantiza la correcta adopción de la tecnología, lo que se traduce en un mayor desarrollo de habilidades y competencias. En resumen, esta investigación busca abordar una problemática real y significativa en la gestión de inventarios de manzana orgánica. La magnitud del problema, su impacto económico y social, y la necesidad de soluciones tecnológicas eficaces justifican plenamente la realización de esta investigación, con la aspiración de mejorar la eficiencia de los productores del CPMT y contribuir al crecimiento del propio colectivo.

CAPÍTULO II
MARCO TEÓRICO

2.1 Fundamentos teóricos

Control de inventarios

Conforme a las reflexiones de Arroba Salto, J. E., Angulo Rosales, Y. A., y Naula Valla, S. M. (2018), la gestión eficiente de inventarios se revela como una herramienta indispensable en el funcionamiento organizacional. Este enfoque no solo posibilita el rastreo meticuloso de ingresos, consumo y comercialización de insumos o productos, sino que también establece un orden minucioso, categorizando dichos elementos según su valor e importancia. Tal control meticuloso, al ser implementado, genera un impacto positivo palpable en la rentabilidad de las empresas. El control de inventarios, según las autoras, constituye el corazón mismo de cualquier entidad que se dedique a la transacción de bienes o servicios. La afirmación subraya la idea de que la eficacia en la administración de inventarios se traduce directamente en un desempeño financiero superior. En este sentido, la gestión adecuada de inventarios se revela como una práctica empresarial esencial para optimizar resultados financieros y garantizar la sostenibilidad y el éxito a largo plazo de la organización.

Productos orgánicos

De acuerdo con Andrade, C. M., & Ayaviri, D. (2018), los alimentos orgánicos aportan de manera más efectiva a la nutrición humana. Para ser clasificados como orgánicos, estos deben haber sido cultivados y procesados mediante prácticas de agricultura orgánica, que se caracterizan por el reciclaje de productos y la promoción de la biodiversidad. Es esencial que estos productos crezcan sin el uso de abonos sintéticos, variedades transgénicas, fertilizantes derivados del petróleo y otros agroquímicos.

Marketing

Godin (2019) propone una visión del marketing como un acto generoso destinado a ayudar a otros a lograr sus metas deseadas. En este contexto, destaca la importancia

de crear narrativas auténticas y conmovedoras que puedan compartirse de manera significativa. Desde esta perspectiva, los profesionales del marketing deben ofrecer soluciones sin caer en el consumismo excesivo, evitar el envío de mensajes no solicitados y prescindir de estrategias que induzcan vergüenza. En lugar de ello, abogan por proporcionar oportunidades que permitan a las personas abordar problemas y avanzar hacia sus objetivos. Según Godin, cuando estas ideas encuentran difusión, contribuyen a dar forma a la cultura, construyendo algo que sería extrañado si no estuviera presente; algo que proporciona significado, forja conexiones y abre nuevas posibilidades (p. 21).

Marketing digital

La perspectiva de Selman (2017) sobre el marketing digital destaca su naturaleza orientada a estrategias de mercado ejecutadas en línea, con el objetivo de estimular a los usuarios de un sitio web a realizar acciones previamente planificadas. A diferencia de las formas convencionales de ventas y mercadeo, el marketing digital se distingue por integrar diversas estrategias y técnicas diseñadas exclusivamente para el entorno digital. En este contexto, se busca aprovechar las características únicas de la plataforma en línea para lograr objetivos específicos, marcando así una clara diferencia con las tácticas tradicionales de marketing (p. 6).

Canales de marketing

Para Vásquez (2009), el propósito fundamental de los canales de distribución o marketing consiste en atender de manera óptima los deseos asociados al consumo, procurando ofrecer condiciones favorables en términos de lugar, tiempo, calidad, precio y presentación. Esto se persigue con la menor inversión posible y de la manera más eficaz.

Aplicación web

Según Molina Ríos, J. R., Zea Ordóñez, M. P., Contento Segarra, M. J., & García Zerda, F. G. (2017), en la actualidad, Internet se ha consolidado como un medio de

comunicación fundamental. Es por esta razón que las aplicaciones web han emergido como agentes intermedios para la difusión de información y la prestación de servicios a los usuarios. En respuesta a esta evolución, se han diseñado diversas metodologías para modelar aplicaciones web con el objetivo de abordar diversos desafíos inherentes al desarrollo de este tipo de software.

Manejo de base de datos

Beynon-Davies (2018) conceptualiza una base de datos como una recopilación organizada de datos que refleja un Dominio de Discurso (UdD). En esta perspectiva, los datos son considerados como hechos, representaciones simbólicas o agrupaciones de símbolos utilizados para denotar conceptos. Cabe destacar que estos datos, en su forma individual, carecen de significado intrínseco y requieren interpretación para adquirir utilidad. En contraste, la información se define como un conjunto de datos interpretados y contextualizados, imbuidos de significado. Específicamente, la información representa un conjunto de datos al que se le ha asignado un significado o una semántica (p. 6).

MySQL

Conforme a los datos proporcionados por Oracle. (s. f.), el software MySQL se distingue por su capacidad para ofrecer un servidor de bases de datos SQL (Structured Query Language) excepcionalmente rápido, que opera de manera multiproceso y multiusuario, garantizando una robustez significativa. Este servidor, conocido como MySQL Server, ha sido diseñado específicamente para hacer frente a sistemas de producción que experimentan cargas intensivas y demandas críticas. Además de su rendimiento destacado, se destaca por su capacidad para integrarse eficazmente en aplicaciones de software de implementación masiva, ampliando su versatilidad y utilidad en diversos entornos y contextos.

PHP

En la contemporaneidad, PHP se posiciona como uno de los lenguajes de programación más prominentes, siendo ampliamente preferido y utilizado en la

comunidad de código abierto. Su popularidad se sustenta en su destacado desempeño en la construcción de aplicaciones web de gran envergadura. En el ámbito de la industria, PHP se erige como un recurso invaluable debido a su naturaleza compatible, escalable, segura y multidisciplinaria. En particular, la versatilidad de PHP se manifiesta en su capacidad para adaptarse a diversas necesidades y contextos de desarrollo. Su compatibilidad le confiere la capacidad de integrarse de manera eficiente con otras tecnologías, facilitando así la creación de soluciones tecnológicas integrales. Además, la escalabilidad inherente al lenguaje permite a los desarrolladores abordar proyectos de diferentes dimensiones, desde aplicaciones más simples hasta aquellas de complejidad considerable.

La seguridad que proporciona PHP es un componente crucial en el desarrollo de aplicaciones web, especialmente en un entorno donde la protección de datos y la integridad de la información son de suma importancia. La robustez de sus mecanismos de seguridad lo convierte en una elección confiable para la construcción de sistemas que manejan información sensible. La multidisciplinariedad de PHP es otro aspecto destacado, ya que este lenguaje no solo es empleado en el ámbito web, sino que también se integra de manera efectiva en diversos campos y aplicaciones. Esta capacidad de adaptación lo convierte en una herramienta valiosa para el desarrollo de soluciones tecnológicas en diferentes sectores. En este contexto, la relevancia de PHP se manifiesta en su capacidad para permitir el desarrollo de aplicaciones ágiles, óptimas e inmediatas, respondiendo así a los requisitos dinámicos de la sociedad actual. Esta valoración se refleja en las observaciones de Peña Cáceres, O. J. M., More More, M. A., Cornejo Sojo, R. E., & Garay Silupu, E. R. (2022), quienes destacan la significativa contribución de PHP en el panorama actual de la programación y el desarrollo de software.

Librerías

Para Schaf, J. (2019), se define una librería o framework en programación como una compilación de rutinas predefinidas que pueden ser utilizadas por un programa. Estas rutinas, generalmente almacenadas en formato objeto, se incorporan

fácilmente en el código del programa, ofreciendo a los desarrolladores una herramienta eficaz para evitar la creación manual de funciones comunes. Esta práctica, no solo agiliza el proceso de desarrollo de software, sino que también contribuye a optimizar el tiempo y los recursos de los programadores al eliminar la necesidad de escribir código desde cero en situaciones recurrentes.

SLIM

Sunardi, A., & Suharjito. (2019) señalan la existencia de varios frameworks populares y ampliamente empleados en el desarrollo, los cuales están programados en diversos lenguajes. Estos frameworks comparten una estructura común que facilita el proceso de aprendizaje y comprensión. Entre ellos, se destacan los Marcos PHP con Slim, que son particularmente favorecidos por los desarrolladores. Slim, un micro-framework PHP, se utiliza específicamente para la creación de servicios web (API) y desempeña un papel crucial en la conexión de múltiples aplicaciones que no pueden compartir el mismo recurso.

JWT

Darmawan, I., Karim, A. P. A., Rahmatulloh, A., Gunawan, R., & Pramesti, D. (2021), el token de Notación de Objetos JavaScript (JSON) para la web se presenta como una técnica de autenticación que brinda una manera abierta y segura de representar declaraciones entre dos partes. Este token, respaldado por una firma criptográfica, ha sido diseñado con la intención de ser inalterable y resistente a la falsificación.

MD5

En la investigación llevada a cabo por Thompson, E. (2005), se hace hincapié en la persistente dificultad que han enfrentado los criptógrafos en su búsqueda de establecer un sistema criptográfico infalible. En este contexto, el criptógrafo Ron Rivest introdujo la función hash MD5 en 1994 como una opción más robusta en comparación con el algoritmo MD4, desarrollado en 1992. La estrategia del algoritmo implica la división de un archivo en bloques de entrada de 512 bits, cada uno

sometido a una serie de funciones para generar un valor hash único de 128 bits asociado al archivo. Es crucial subrayar que cualquier modificación, incluso mínima, en un solo bit de cualquiera de los bloques de entrada debería desencadenar un efecto cascada, alterando de manera completa los resultados del hash.

Angular

Según la investigación realizada por Llerena Ocaña, L. A., Fernández Villacres, G. E., Viscaino Naranjo, F. A., & Baño Naranjo, F. P. (2021), la mayoría de los marcos en el contexto del Modelo-Vista-Controlador (MVC) se construyen sobre componentes. Existe una abundancia de información acerca de los marcos de desarrollo desarrollados en TypeScript, lo que facilita el proceso de aprendizaje para los estudiantes y la búsqueda de soluciones a diversos problemas. El código dentro de estos frameworks demuestra ser fácilmente adaptable y moldeable en los aspectos fundamentales del aprendizaje, permitiendo mejoras en las interfaces con la ayuda de CSS, SASS y LESS. Aunque la curva de aprendizaje con Angular es más extensa en comparación con otros marcos de desarrollo, una vez que se ha adquirido el conocimiento, su aplicación se vuelve muy sencilla en cualquier framework.

Por su parte, Soles, N., y Salazar, L. (2021) opinan que Angular simplifica y mejora el desarrollo de sistemas web, ya que este framework trabaja de manera eficiente al separar el front end y el back end, evitando la repetición innecesaria de código y manteniendo la organización gracias al Modelo-Vista-Controlador (MVC). Este enfoque garantiza la rapidez en el desarrollo web y, al mismo tiempo, facilita cambios y actualizaciones de manera eficaz.

Diagrama UML

Según la investigación de Hernández-González, Anaisa, Enríquez-Hernández, Dairis Milagros, & Ruano-Chichatskaia, Alexandra (2021), el Lenguaje de Modelado Unificado (UML) constituye un sistema de modelado que cuenta con una notación sólida para la creación y desarrollo de sistemas. Este lenguaje brinda la tecnología esencial necesaria para respaldar las prácticas de la ingeniería de software. Al

erigirse como un estándar, el UML se consolida como la notación predilecta empleada por los desarrolladores de software para documentar el proceso de desarrollo de software.

Diagrama de secuencia

Conforme a la investigación realizada por Vidal Silva, C. L., Villarroel, R. H., López Cortés, X. A., & Rubio, J. M. (2019), un diagrama de secuencia en UML tiene como objetivo principal ilustrar las interacciones entre diferentes objetos en un sistema, representando el intercambio de mensajes en escenarios específicos. De esta manera, estos diagramas son eficaces para modelar el comportamiento de los objetos que participan en casos de uso definidos, teniendo en cuenta la descripción de las clases y sus elementos constitutivos, como atributos y métodos. Dada la existencia de diversos escenarios potenciales, se enfatiza la importancia de clasificarlos en función de su relevancia para la representación de los comportamientos más destacados. Además, los diagramas de secuencia UML presentan la característica adicional de poder representar interacciones algorítmicas mediante fragmentos combinados, que se corresponden con diagramas de secuencia secundarios dentro de un diagrama principal, estableciendo así una composición de comportamiento de nivel superior a nivel inferior.

Diagrama de caso de uso

Los diagramas de caso de uso son una técnica empleada para capturar información acerca de cómo opera un sistema o negocio, involucrando elementos como casos de uso y actores. Estos actores, que pueden representar entidades con comportamiento, como personas identificadas por un rol, sistemas informáticos u organizaciones, se vinculan entre sí en el diagrama para ilustrar las relaciones existentes. (Larman, 2003).

Visual Studio Code

Microsoft. (s. f.), proporciona una definición detallada de Visual Studio Code, describiéndolo como un editor de código fuente que combina ligereza y potencia, diseñado para funcionar en entornos de escritorio y compatible con los sistemas operativos Windows, macOS y Linux. Este editor presenta características integradas de soporte para JavaScript, TypeScript y Node.js, ofreciendo así una plataforma versátil para el desarrollo de aplicaciones. Además, Visual Studio Code se distingue por su extenso ecosistema de extensiones, que abarcan diversos lenguajes de programación y entornos de ejecución, incluyendo, pero no limitado a, C++, C#, Java, Python, PHP, Go y .NET. Este enfoque expansivo y flexible permite a los desarrolladores personalizar su experiencia de programación según sus necesidades específicas, contribuyendo así a la amplitud y diversidad de posibilidades en el ámbito del desarrollo de software.

API REST

Conforme a las investigaciones de Arsaute, A., Zorzán, F. A., Daniele, M., González, A., & Frutos, M. (2018), en el contexto contemporáneo, se percibe un incremento considerable de proyectos y aplicaciones que emplean una API REST, una alternativa innovadora en el terreno de Servicios Web (WS) destinada a la construcción de servicios profesionales. La arquitectura REST se configura como una interfaz entre sistemas que se sirve del protocolo HTTP para la obtención de datos o la ejecución de operaciones sobre estos datos, destacando por su capacidad de adaptarse a diversos formatos como XML, JSON, HTTP, entre otros. Este enfoque se erige como una elección sumamente versátil y de amplia aceptación en el desarrollo de servicios, presentando una solución dinámica y ágil frente a las demandas actuales del panorama de desarrollo de software.

Postmant

Según la descripción de Hyams, R., & Pilko, T. (2022), Postman se presenta como una plataforma de API con una interfaz de usuario intuitiva que simplifica el proceso

de realizar llamadas API. Además, facilita la gestión de solicitudes en grandes volúmenes y secuencias concatenadas.

CAPÍTULO III
DESARROLLO Y METODOLOGÍA

3.1 Procedimiento y descripción de actividades realizadas

1. **Definición de objetivos**

Establecer claramente el propósito de la aplicación web, centrándose en optimizar la gestión de inventarios de la manzana orgánica del CPMT.

2. **Análisis de requisitos**

Identificar y comprender los requisitos específicos de los productores de manzana Teziutlán para integrarlos en la aplicación web.

3. **Diseño de la interfaz de usuario**

Desarrollar una interfaz intuitiva y amigable que facilite la gestión eficiente de inventarios por parte del administrador y que atraiga a los clientes.

4. **Desarrollo del sistema de gestión de inventario**

Implementar un sistema que garantice la actualización constante de la información de productos para el administrador dentro de la aplicación.

5. **Implementación de funcionalidades específicas**

Integrar características clave como la seguridad por medio de tokens de autorización dentro de la aplicación, registro de productos y gestión de pedidos para satisfacer las necesidades del CPMT.

6. **Pruebas de funcionalidad**

Ejecutar pruebas para verificar que el programa funcione correctamente y que no genere errores, garantizando así la conformidad con los requisitos establecidos.

7. **Capacitación del personal**

Preparar y ejecutar sesiones de capacitación para el personal encargado del manejo de inventarios, garantizando una comprensión completa de la aplicación.

8. **Pruebas de usabilidad con usuario final**

Se esperan al menos 3 meses desde su implementación para comprobar la usabilidad con usuarios finales y obtener resultados.

Tabla 1.- Actividades realizadas.

Actividades	JUL 24-29	J-A 31-05	AGO 07-12	AGO 14-19	A-S 28-02	SEP 04-09	SEP 11-16	SEP 18-23	SEP 25-30	OCT 02-07	OCT 09-14	OCT 16-21	OCT 23-28	O-N 30-04	NOV 06-11	NOV 13-18	NOV 20-25
Definición de objetivos	■	■	■														
Análisis de requisitos			■	■	■												
Diseño de la interfaz de usuario						■	■	■	■								
Desarrollo del sistema de gestión de inventario										■	■	■	■				
Implementación de funcionalidades específicas												■	■	■	■		
Pruebas de funcionalidad														■	■	■	
Capacitación del personal																	■

Fuente: Propia, 2023.

3.2 Alcance y enfoque de la investigación

La investigación tiene como alcance la mejora de la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán (CPMT) a través del desarrollo de una aplicación web. El objetivo principal es agilizar el control de inventarios, asegurando la disponibilidad adecuada de productos para satisfacer la demanda de los clientes. La aplicación se ha desarrollado como respuesta a las necesidades identificadas, y su implementación será evaluada cuantitativamente mediante los datos de producción reflejados en la base de datos, asi como, los pedidos, las ventas, y el manejo de los diferentes puntos de venta gestionados por el colectivo. El enfoque de la investigación se define como descriptivo de tipo transversal, generado con los resultados de las encuestas, con preguntas dirigidas a validar la eficiencia en el manejo de inventarios mediante la aplicación web. Esta elección metodológica se debe a la naturaleza específica de la investigación, centrada en la evaluación de la eficacia y la aceptación de la aplicación por parte del personal involucrado en la gestión de inventarios.

3.3 Hipótesis

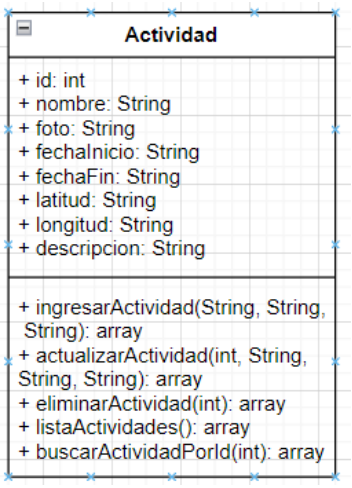
Se plantea que la implementación de una aplicación web para la gestión de inventario contribuirá en la eficiencia de la gestión de stock de manzana orgánica producida por el Colectivo de Productores de Manzana Teziutlán (CPMT)

3.4 Diseño y metodología de la investigación

La metodología de investigación seleccionada adoptará un enfoque cuantitativo, dentro de un marco descriptivo que abarcará un estudio de tipo transversal. Se implementará un instrumento específico de evaluación de satisfacción del sistema diseñado para la gestión de inventarios de la manzana orgánica dentro del Colectivo de Productores de Manzana Teziutlán (CPMT). Además, se emplearán fórmulas para calcular la muestra de la población objetivo. Como instrumento para la recolección de datos se utilizó el cuestionario con respuestas medidas en escala de liker; posteriormente se concentraron los datos para su posterior análisis e interpretación de datos, y poder conocer la percepción y aceptación de la aplicación, y en qué medida la aplicación contribuyó al mejor manejo y control de inventarios a los miembros del CPMT. Con respecto al desarrollo de la aplicación web, se inicia con el diseño de la base de datos para lo cual se realizaron los diagramas de clase, posteriormente, para el diseño de la aplicación se crearon los diagramas de caso de uso, y diagramas de secuencia, además del desarrollo de la misma utilizando angular para la creación del frontend y el framework de Slim para el backend.

Diagramas de clase

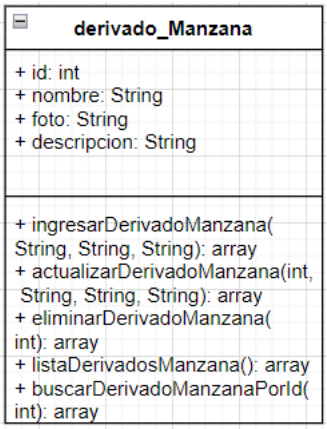
Ilustración 1.- Clase de actividad.



Fuente: Propia, 2023.

En la Ilustración 1 se puede apreciar la clase correspondiente a actividad, con la cual se plantea almacenar actividades importantes para el colectivo con el fin de darlas a conocer, cuenta con el nombre de la actividad, la fecha de inicio y fin de dicha actividad, la latitud y longitud para obtener las coordenadas donde se desarrolla el evento, asi como la descripción del evento para más información.

Ilustración 2.- Clase de derivado de manzana.

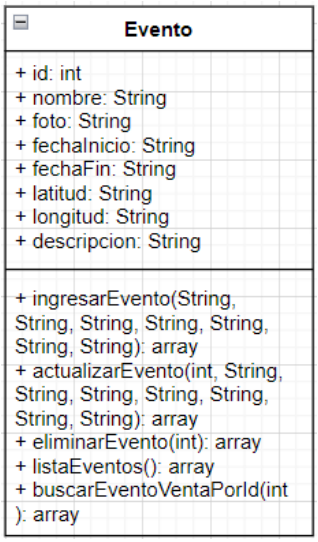


Fuente: Propia, 2023.

En la ilustración 2 se puede apreciar la clase UML para los derivados de la manzana, con el fin de promocionar otros productos que se hacen con las manzanas en el

colectivo, dicha clase cuenta con el nombre, la foto, y la descripción del derivado de la manzana.

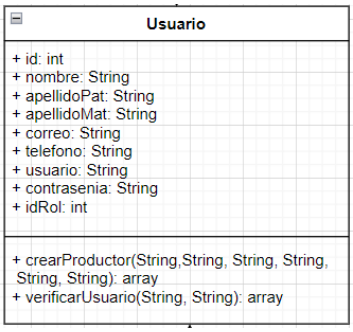
Ilustración 3.- Clase de evento.



Fuente: Propia, 2023.

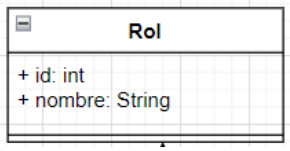
En la Ilustración 3 se puede apreciar la clase UML de evento, dicha clase tiene el fin de almacenar los eventos de promoción y de otros tipos del colectivo, dicha clase cuenta con el nombre, la foto, la fecha de inicio, la fecha de fin, el lugar como las coordenadas, la descripción, todo del evento que se desea registrar.

Ilustración 4.- Clase de usuario.



Fuente: Propia, 2023.

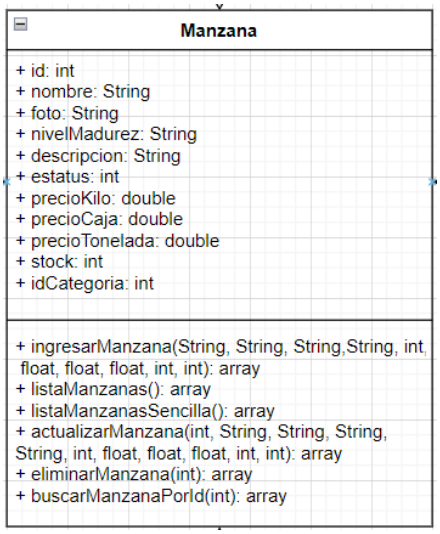
Ilustración 5.- Clase de rol.



Fuente: Propia, 2023.

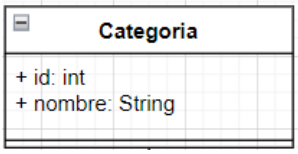
En la Ilustración 4 se puede observar la clase UML para el usuario, dicha clase está pensada para almacenar los datos del administrador del sitio y de los productores, ya que solo ellos tienen la capacidad de iniciar sesión en base a la funcionalidad del sitio, para diferenciar a los productores del administrador se creó la clase UML de rol que se puede apreciar en la Ilustración 5.

Ilustración 6.- Clase de manzana.



Fuente: Propia, 2023.

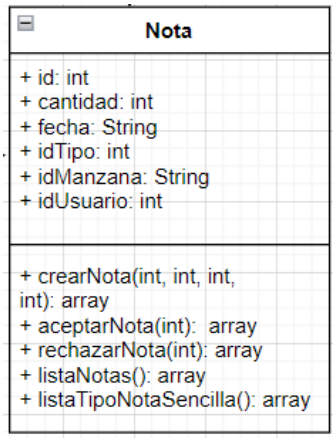
Ilustración 7.- Clase de categoría.



Fuente: Propia, 2023.

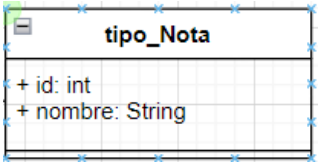
En la Ilustración 6 se observa una de las clases UML principales de todo el proyecto, la clase manzana, dicha clase cuenta con los datos de las manzanas que se venden dentro del colectivo, cuenta con el nombre, la foto, el nivel de madurez, la descripción para más información, el estatus, precio por kilo, precio por caja, precio por tonelada, el stock, y el id de la categoría al que pertenece dicha manzana para ser clasificada, esta relación se lleva a cambio mediante ese atributo con respecto a la clase UML de categoría que se aprecia de forma detallada en la Ilustración 7.

Ilustración 8.- Clase de nota.



Fuente: Propia, 2023.

Ilustración 9.- Clase de tipo de nota.

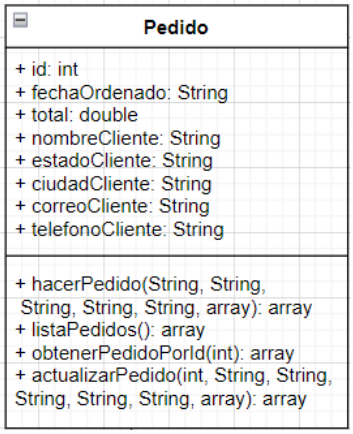


Fuente: Propia, 2023.

En la Ilustración 8 se encuentra la clase UML referente a nota, dicha clase tiene la funcionalidad de almacenar solicitudes de modificación del stock como si fueran notas, la Ilustración 9 por su parte es la clase UML para el tipo de nota que representan las notas antes mencionadas, para poder conocer si es una solicitud de aumento o disminución del stock, la clase de la Ilustración 8 cuenta con la cantidad,

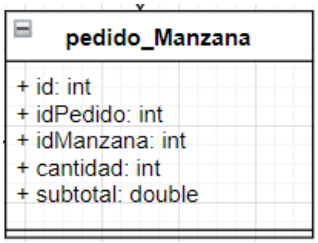
el tipo con el cual se relaciona con la clase de la Ilustración 9, la manzana con la que está relacionada esa nota, y el usuario que hizo dicha nota.

Ilustración 10.- Clase de pedido



Fuente: Propia, 2023.

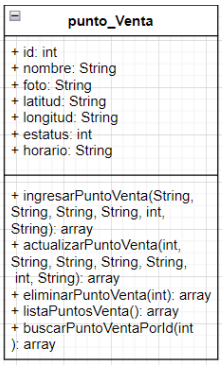
Ilustración 11.- Clase de clase intermedia entre pedido y manzana.



Fuente: Propia, 2023.

En la Ilustración 10 se aprecia la clase UML de pedido, dicha clase almacena los pedidos de manzana solicitados por los clientes, debido a la relación muchos a muchos que existe entre las clases de manzana y de pedido se creó la clase UML intermedia que se aprecia en la Ilustración 11, la clase pedido toma la fecha en que se ordenó el pedido, el total de todo el pedido, el nombre, estado, ciudad, correo, y teléfono del cliente, mientras que la clase intermedia de la Ilustración 11 cuenta con el id del pedido, el id de la manzana a la cual pertenece, la cantidad solicitada de dicha manzana, y el subtotal por esa manzana en el pedido.

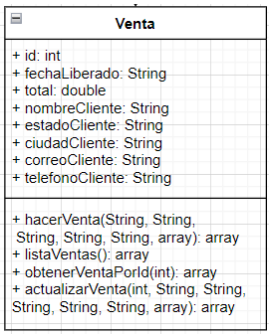
Ilustración 12.- Clase de clase para el punto de venta



Fuente: Propia, 2023.

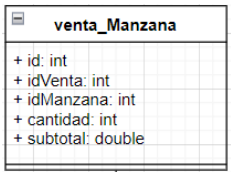
En la Ilustración 12 se muestra la clase UML para almacenar un punto de venta, dicha clase representa los lugares físicos donde el colectivo vende sus productos, la clase almacena el nombre, la foto, las coordenadas, el estatus, y el horario de atención del punto de venta en cuestión.

Ilustración 13.- Clase de clase venta.



Fuente: Propia 2023.

Ilustración 14.- Clase de clase intermedia entre venta y manzana

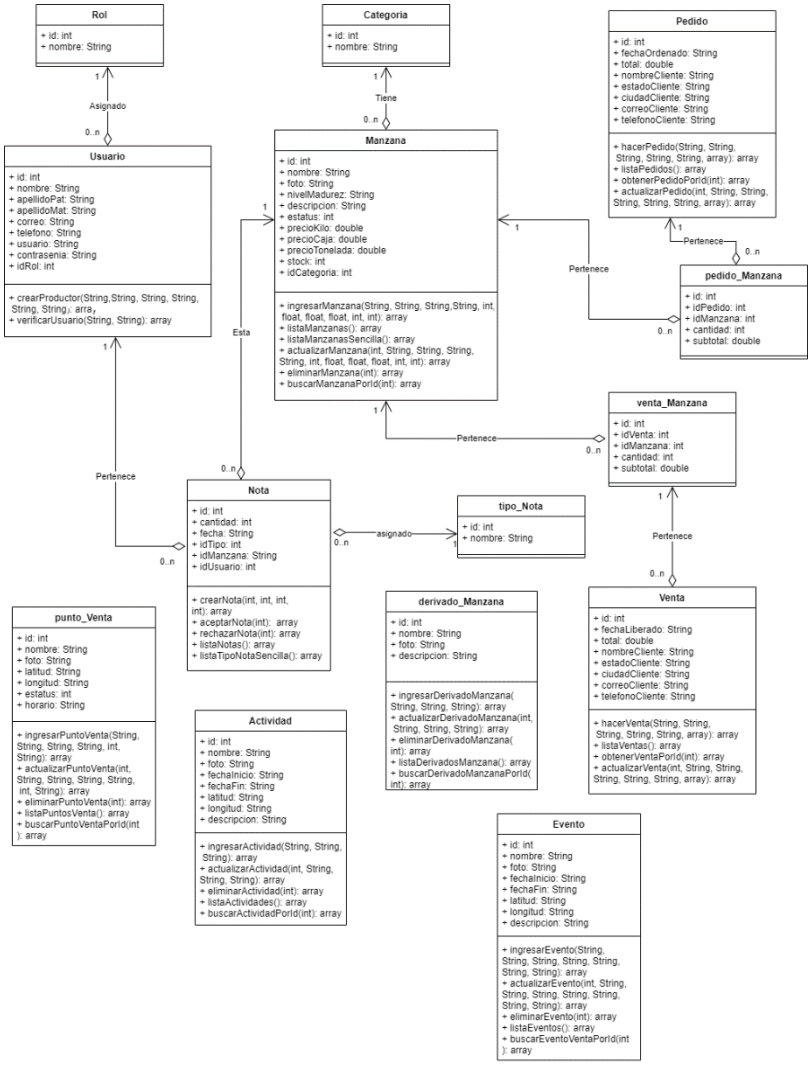


Fuente: Propia 2023.

En la Ilustración 13 se muestra la clase UML referente a venta, dicha clase almacena todas las ventas de manzanas al mayoreo realizadas por el colectivo una vez que el

administrador libere los pedidos, debido a la relación muchos a muchos que existe entre venta y manzana se tiene la clase UML intermedia de la Ilustración 14, en la clase venta se almacena la fecha en la que se liberó la venta, el total por esa venta, el nombre del cliente, el estado del cliente, la ciudad del cliente, el correo de cliente, y el teléfono del cliente, mientras que la clase intermedia de la Ilustración 14 cuenta con el id de la venta, el id de la manzana a la cual pertenece, la cantidad vendida de dicha manzana, y el subtotal por esa manzana en la venta.

Ilustración 15.- Diagrama de clases completo.

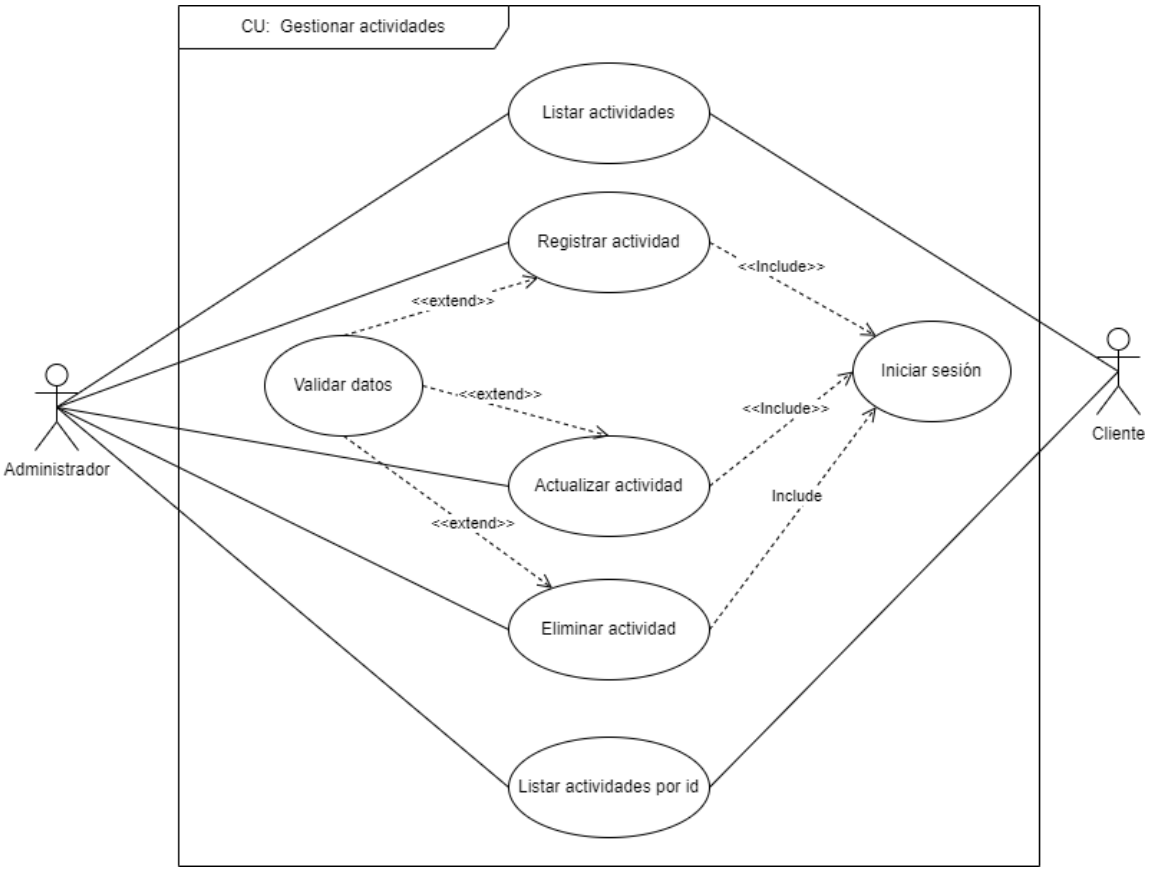


Fuente: Propia, 2023.

En la Ilustración 15 se muestra el diagrama UML completo, dicho diagrama muestra todas las clases dentro de la aplicación, con sus respectivas relaciones y conexiones entre ellas.

Diagramas de casos de uso

Ilustración 16.- Diagrama de caso de uso de gestión de actividades.



Fuente: Propia, 2023.

En la Ilustración 16 se aprecia el diagrama de caso de uso referente a la gestión de actividades, en dicho diagrama se muestra que tanto el administrador como los clientes tienen acceso a los métodos que son solo de consulta, donde se listan las

actividades generales, o una en específico mediante su id, pero solo el administrador tiene acceso a los métodos que modifican la base de datos, dichos métodos son registrar actividad, actualizar actividad, y eliminar actividad, estos mismos tuvieron que haber pasado por el inicio de sesión y se tiene que pasar por la validación de datos para poder llevarlos a cabo.

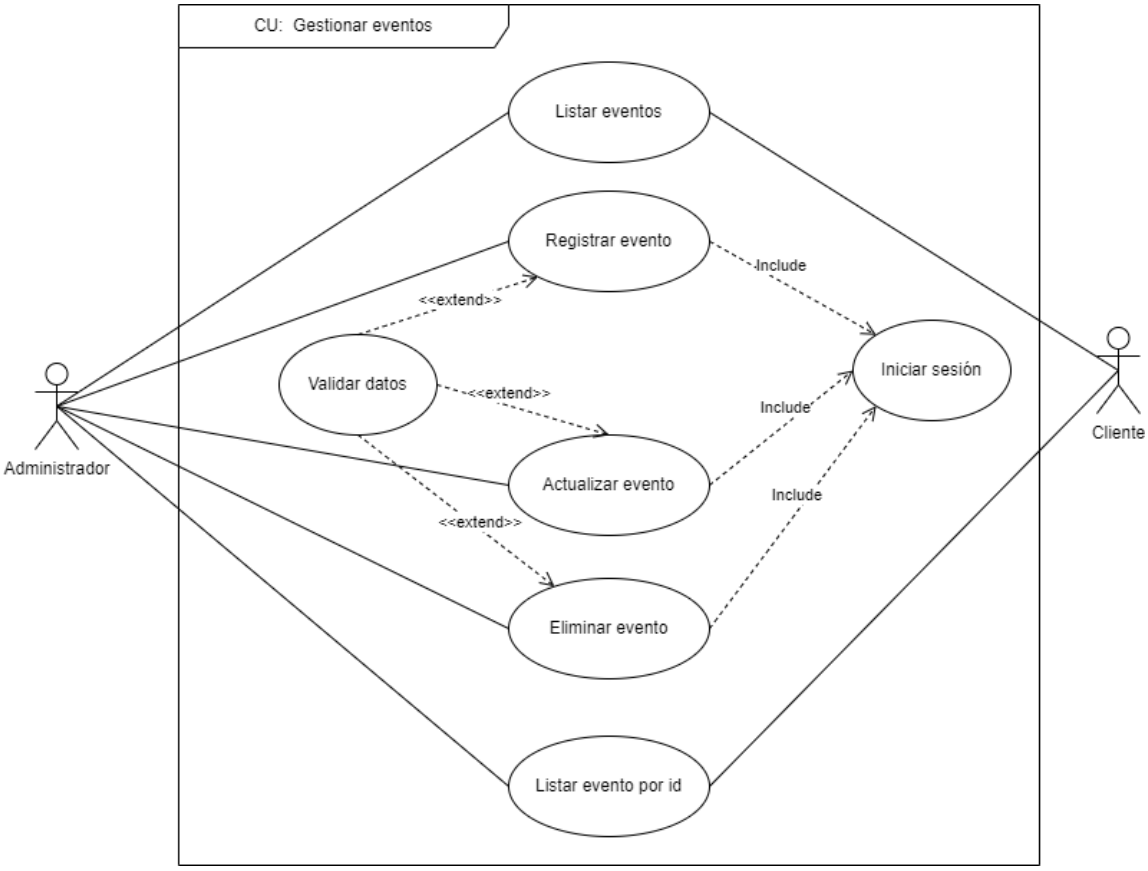
Ilustración 17.- Diagrama de caso de uso de gestión de derivados de la manzana.

Fuente: Propia, 2023.

El diagrama de caso de uso relacionado a la gestión de derivados de la manzana se puede observar con claridad en la Ilustración 17, en la misma se puede observar que el cliente solo tiene acceso a listar los derivados de la manzana, ya sea de manera general o solo un derivado con base a su id, mientras que el administrador

también puede listar los derivados de la manzana de ambas maneras, además de poder registrar un nuevo derivado, actualizar uno ya existente, o eliminar alguno, para hacer estos últimos procesos se debe de pasar por el inicio de sesión y posteriormente en los métodos se debe pasar por la validación de datos.

Ilustración 18.- Diagrama de caso de uso de gestión de eventos.

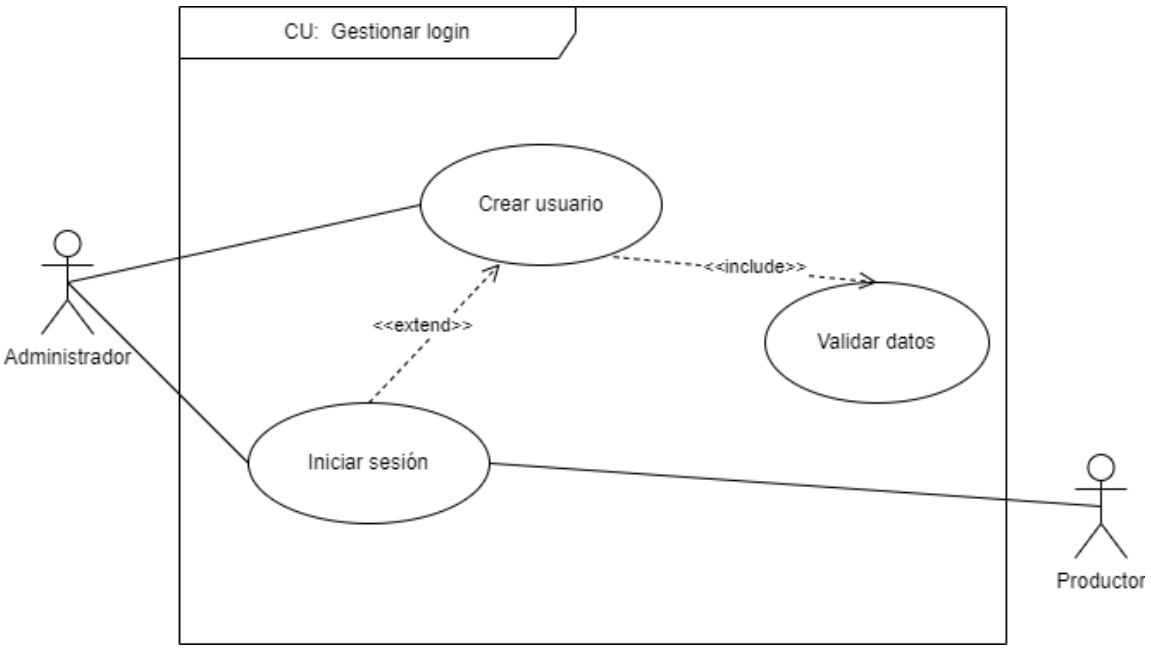


Fuente: Propia, 2023

En la Ilustración 18 se encuentra el diagrama de caso de uso para la gestión de los eventos del colectivo, basándose en el funcionamiento de la aplicación, el cliente solo puede visualizar los eventos, ya sea todos a la vez, o uno en específico, por ello el hecho de que el cliente solo tenga acceso a listar eventos y listar evento por id, mientras que el administrador además de poder hacer estas últimas dos funciones,

también puede registrar, actualizar y eliminar un evento, siempre y cuando estos validen los datos en cada método, y se haya iniciado sesión anteriormente para que pueda tener acceso a los métodos.

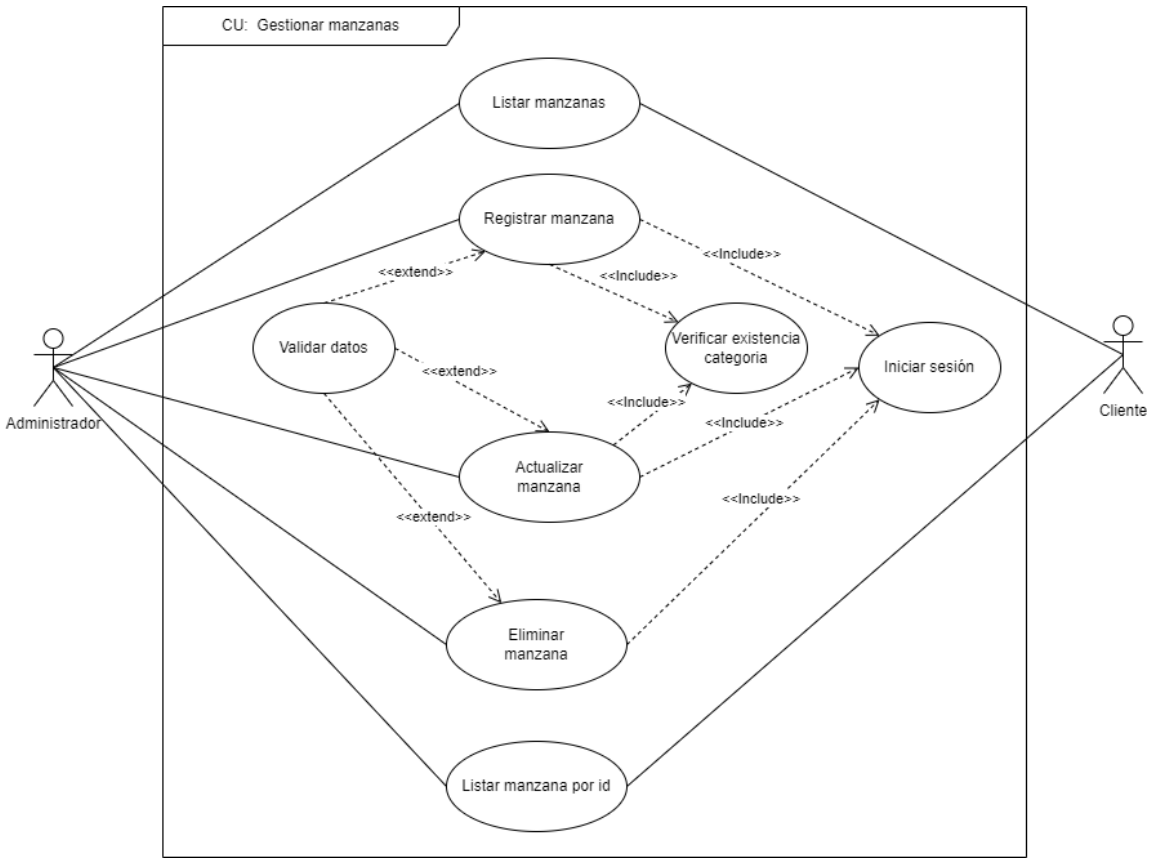
Ilustración 19.- Diagrama de caso de uso de inicio de sesión.



Fuente: Propia, 2023

Para la parte del inicio de sesión, se puede observar el diagrama de caso de uso que lo representa en la Ilustración 19, en dicho diagrama se muestra que el administrador y el productor son quienes pueden iniciar sesión, sin embargo, solo el administrador puede crear usuarios adicionales, el crear un usuario implica que antes se hayan validado los datos para realizarlo, y posterior a la creación del usuario se puede iniciar sesión, o ir directamente desde un inicio a esta opción.

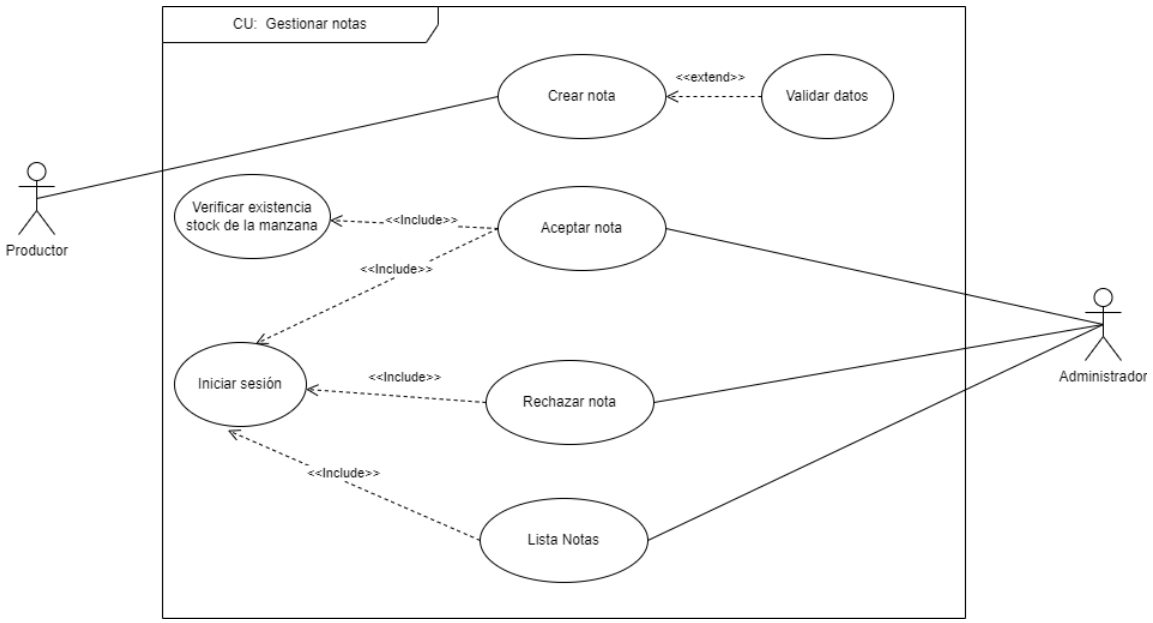
Ilustración 20.- Diagrama de caso de uso de la gestión de las manzanas.



Fuente: Propia, 2023.

La gestión y manejo del producto principal del colectivo es muy importante, en la Ilustración 20 se maneja esta gestión por medio de su diagrama de caso de uso, donde el cliente solo puede listar las manzanas, y listar una manzana en específico mediante su id, mientras que el administrador además de estas dos opciones también puede registrar una manzana, actualizarla, o eliminarla siempre y cuando se haya iniciado sesión para poder acceder a estos métodos, y además de que haya escogido una categoría válida para la manzana, además se debe hacer una validación de datos posterior a los métodos.

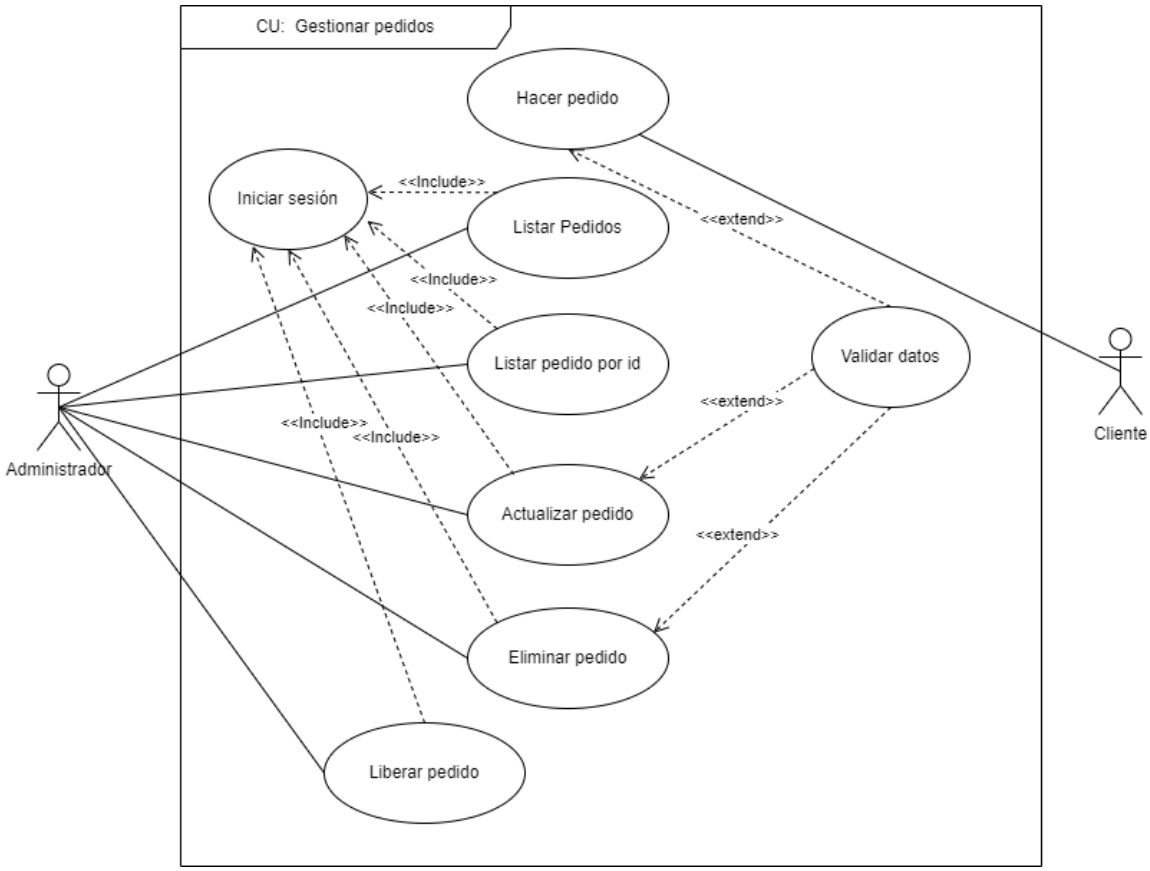
Ilustración 21.- Diagrama de caso de uso de la gestión de notas.



Fuente: Propia, 2023.

En la Ilustración 21 se muestra el diagrama de caso de uso de la gestión de notas, en dicho diagrama se muestra como el productor es quien realiza las solicitudes de modificación al stock de las manzanas por medio de notas, de la creación de una nota se desprende la validación de los datos de dicha nota, por su parte el administrador es quien puede aceptar, rechazar o ver las notas, siempre y cuando se haya pasado por el inicio de sesión, y en el caso de aceptar la nota, que primero se verifique el stock de la manzana.

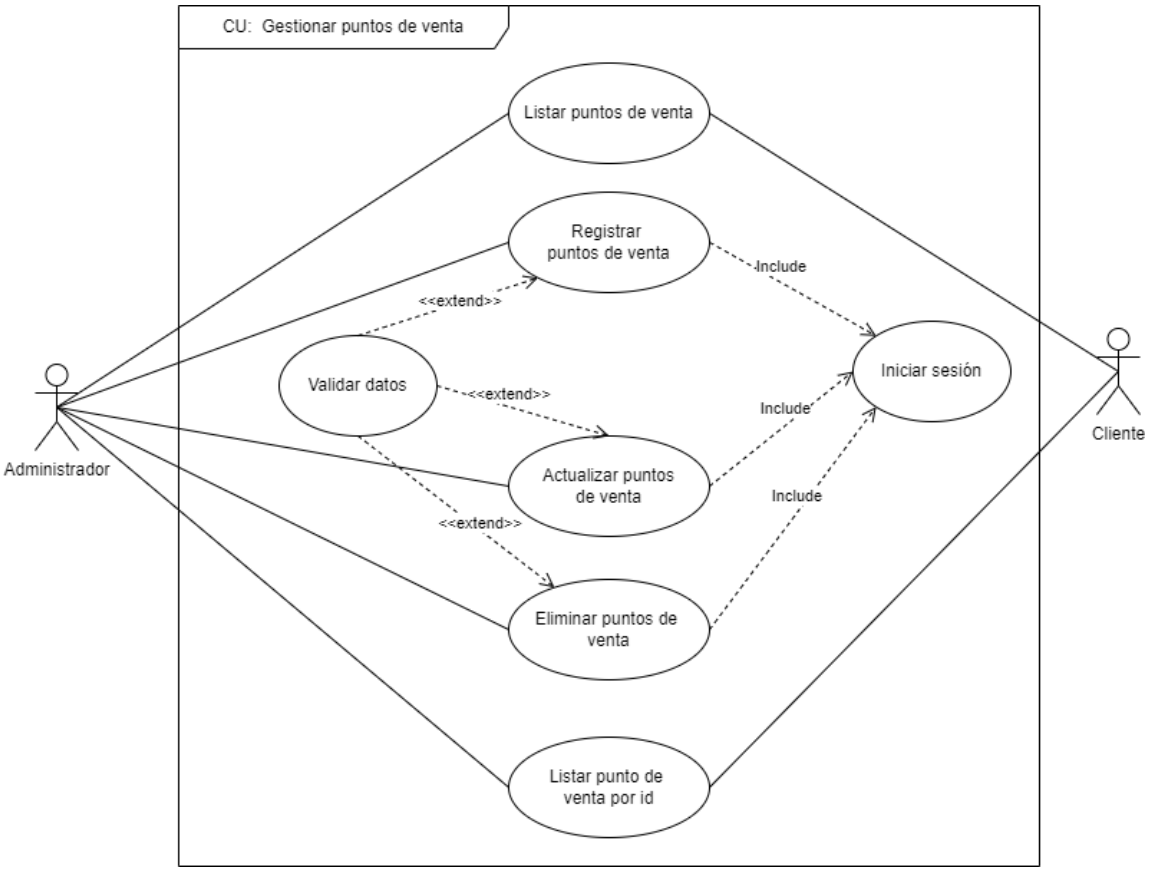
Ilustración 22.- Diagrama de caso de uso de la gestión de pedidos.



Fuente: Propia, 2023.

En la Ilustración 22 se muestra el diagrama de caso de uso referente a la gestión de pedidos, donde el cliente es quien hace los pedidos, mientras que el administrador es quien puede listar los pedidos, listar un pedido en específico por su id, actualizar un pedido, eliminar el pedido, o liberar el pedido, en todas las acciones que realiza el administrador se debe pasar primero por el inicio de sesión, mientras que para hacer, actualizar, y eliminar un pedido, se deben validar los datos para su correcto funcionamiento.

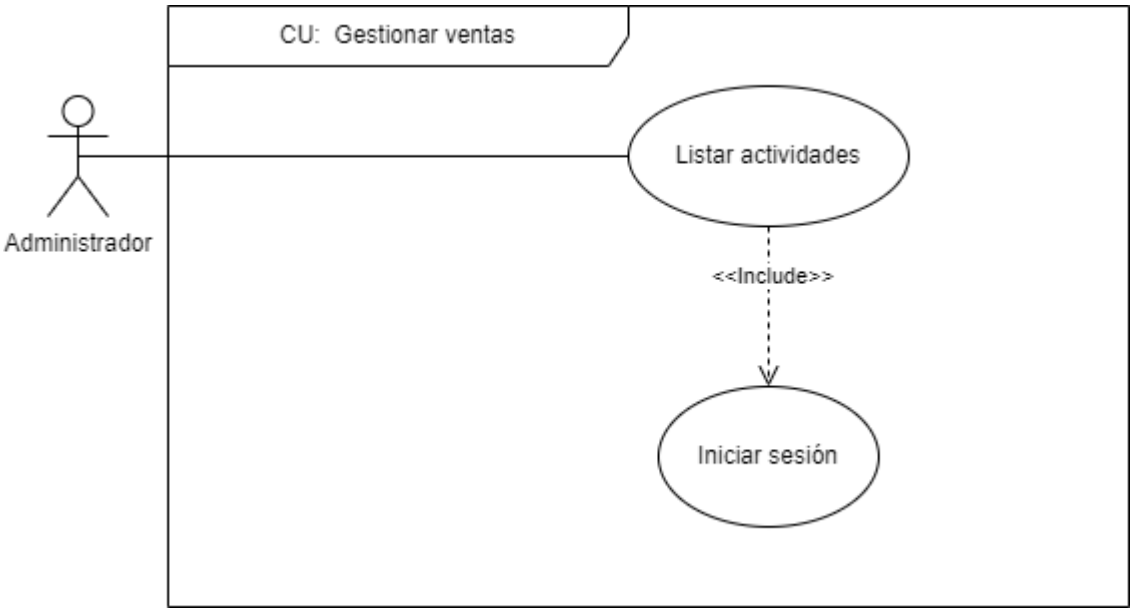
Ilustración 23.- Diagrama de caso de uso de la gestión de puntos de venta.



Fuente: Propia, 2023.

En la Ilustración 23 se presenta el diagrama de caso de uso relacionado con la gestión de puntos de venta. En este diagrama, se evidencia que tanto el administrador como los clientes cuentan con la posibilidad de acceder a métodos exclusivamente de consulta. Estos métodos permiten listar actividades generales o visualizar una actividad específica mediante su identificador. Sin embargo, únicamente el administrador posee acceso a los métodos que implican modificaciones en la base de datos. Estos métodos incluyen el registro de una actividad, la actualización de la información de una actividad existente, así como la eliminación de una actividad. Es importante señalar que, para ejecutar estos procesos, es necesario realizar un inicio de sesión y someterse a la validación de datos correspondiente.

Ilustración 24.- Diagrama de caso de uso de la gestión de ventas.

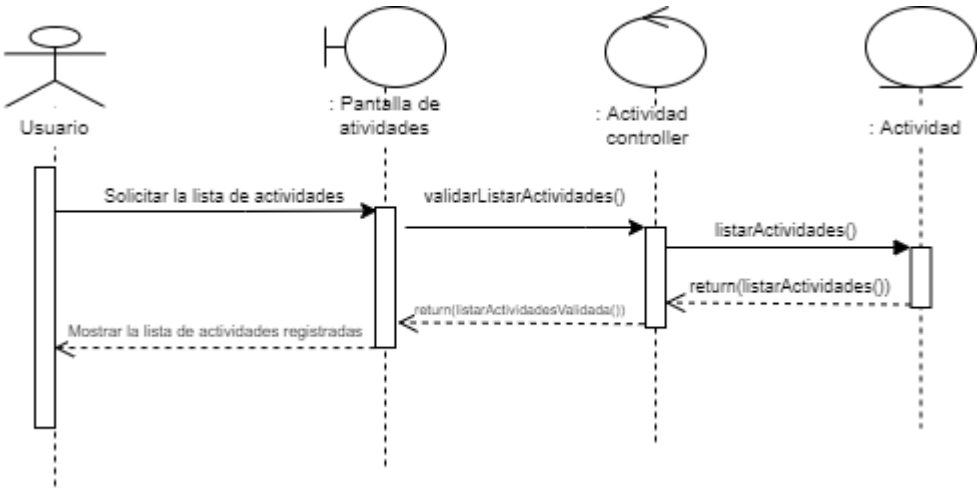


Fuente: Propia, 2023.

En la Ilustración 24 se muestra el diagrama de caso de uso para la gestión de ventas, solo se cuenta con un caso de uso, el listar actividades, dicho caso de uso solo puede ser accedido por el administrador de la aplicación, siempre y cuando se haya iniciado sesión.

Diagramas de secuencia

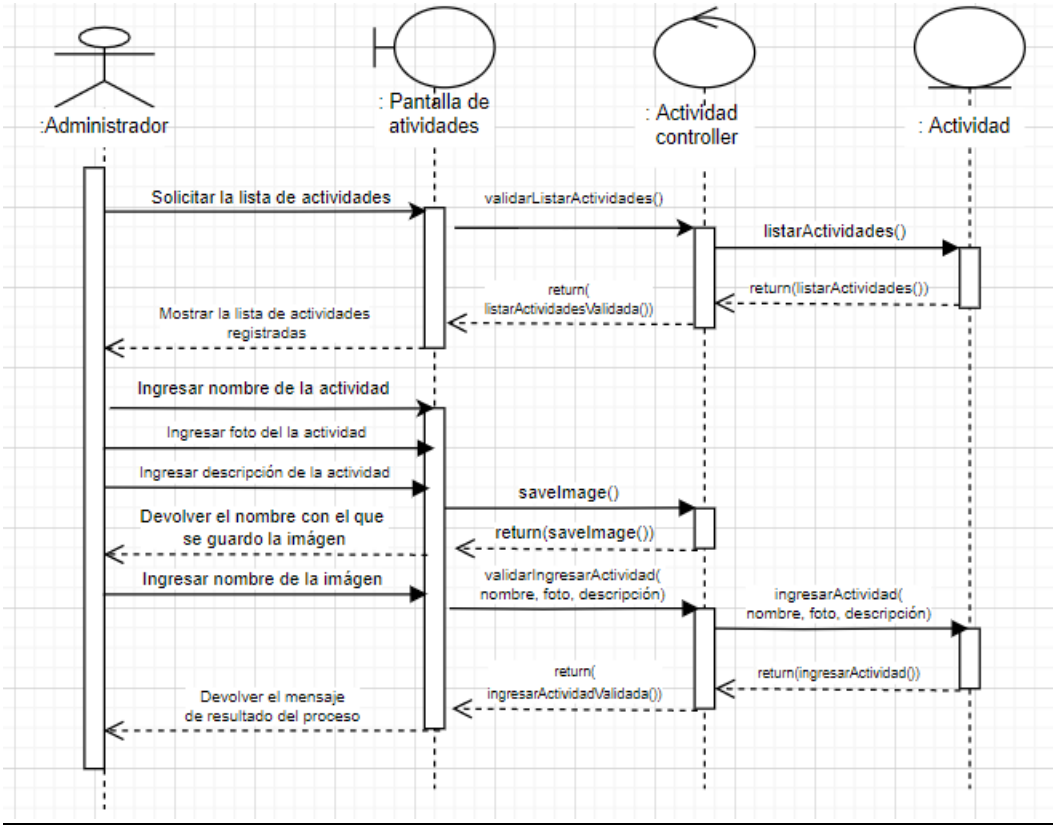
Ilustración 25.- Diagrama de secuencia para listar las actividades.



Fuente: Propia, 2023.

En la Ilustración 25 se muestra el diagrama de secuencia para listar las actividades, donde el usuario solicita la lista a la pantalla de actividades, la pantalla solicita los datos y validaciones al controlador de las actividades, posteriormente se manda a traer la lista de la tabla en la base de datos, y se le muestra al usuario.

Ilustración 26.- Diagrama de secuencia para registrar una actividad.

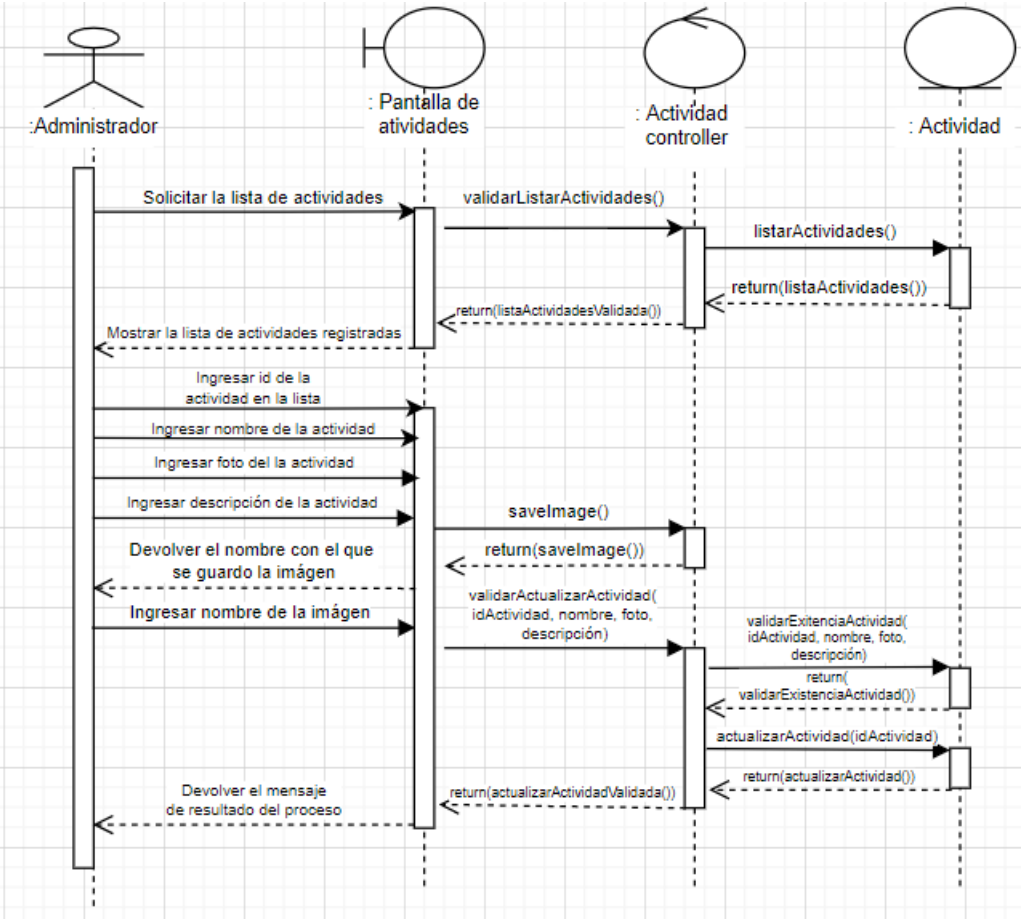


Fuente: Propia, 2023.

En la Ilustración 26 se muestra el diagrama de secuencia para registrar una actividad donde se solicita la lista de actividades en la pantalla, posteriormente se valida la lista en el controlador de actividades, después de la validación de la solicitud se obtiene la lista de actividades de la base de datos, y se le muestra al administrador la lista, posteriormente el administrador ingresa el nombre, la foto, y la descripción de la actividad, la foto es guardada por el controlador, y se devuelve el nombre con el que se almacenó dicha foto, la cual se envía junto con los datos anteriores

ingresados, y se válida la inserción de la actividad en el controlador de actividades, el cual ingresa la actividad, y finalmente devuelve el mensaje de resultado del proceso de inserción al administrador.

Ilustración 27.- Diagrama de secuencia para actualizar una actividad.

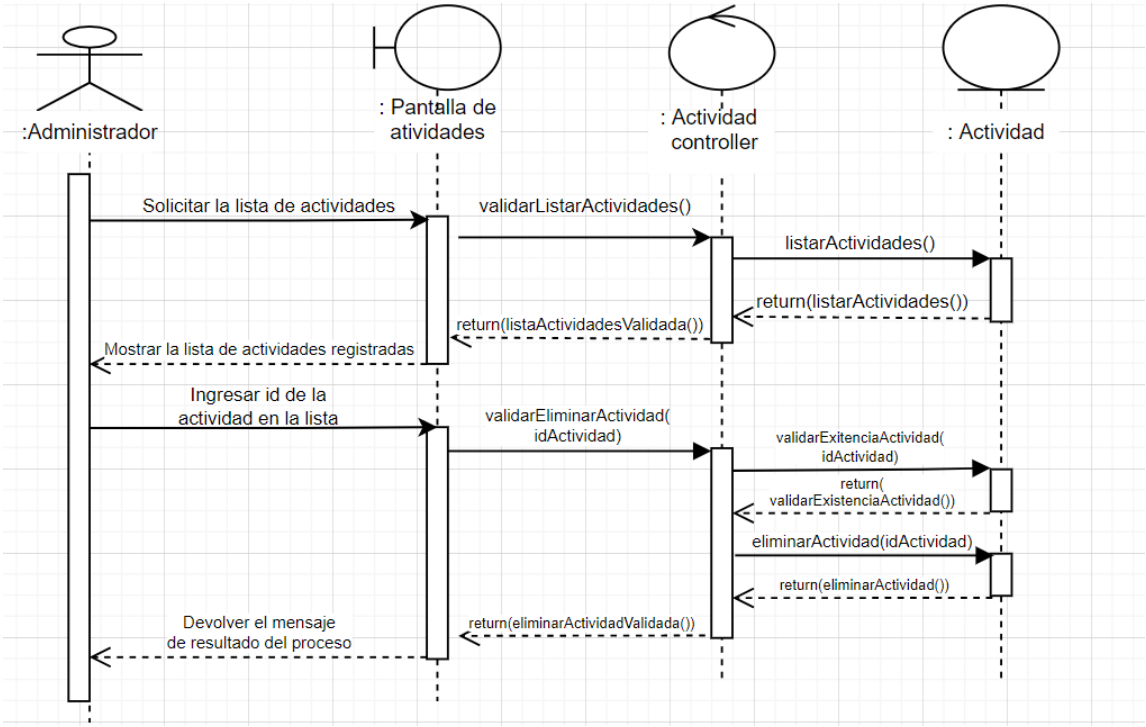


Fuente: Propia, 2023.

En la Ilustración 27, se presenta el diagrama de secuencia correspondiente al proceso de actualización de una actividad. En este proceso, se inicia solicitando la lista de actividades en la pantalla. Luego, la lista se valida en el controlador de actividades. Después de validar la solicitud, se accede a la base de datos para obtener la lista actualizada de actividades, la cual se presenta al administrador. A continuación, el administrador introduce el identificador de la actividad, el nombre, la foto y la descripción de la actividad. El controlador se encarga de almacenar la foto, devolviendo el nombre con el que se guarda dicha imagen. Este nombre, junto

con los datos previamente ingresados, se envía como un conjunto al controlador para su validación, en donde se verifica que exista la actividad que se desea modificar, una vez que se comprueba esto, se procede a actualizar la actividad con los nuevos datos, y finalmente se le muestra al administrador el mensaje de resultado del proceso.

Ilustración 28.- Diagrama de secuencia para eliminar una actividad.

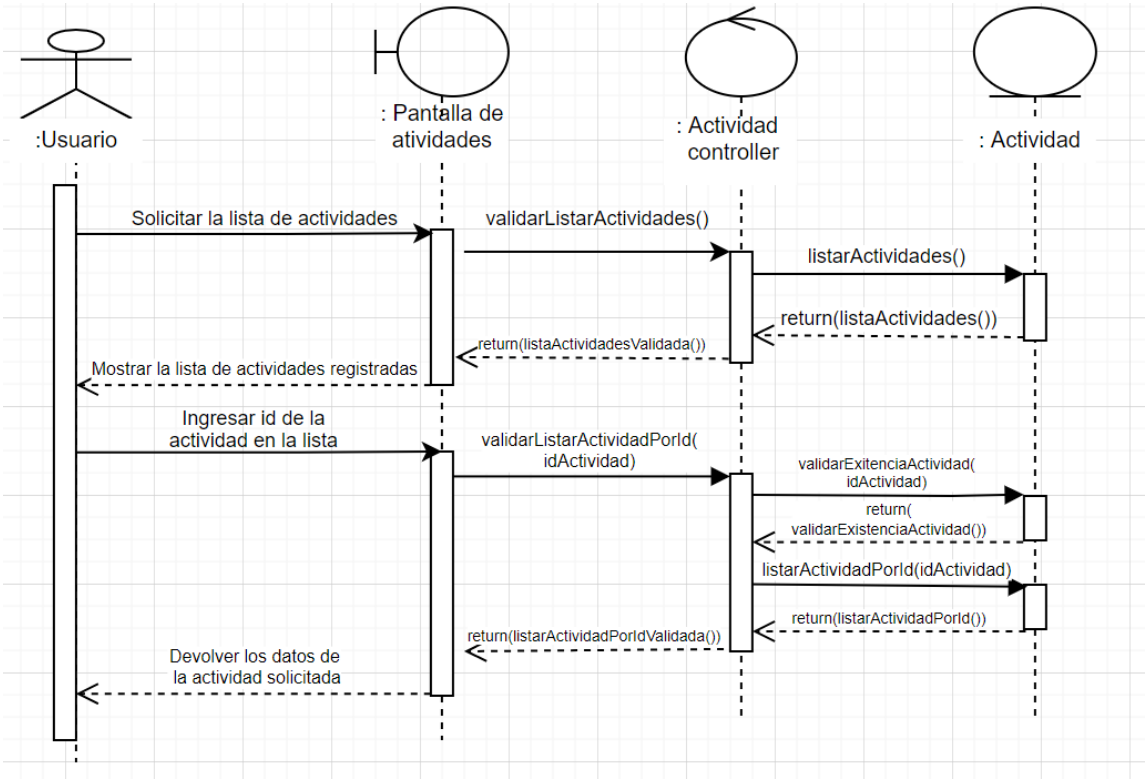


Fuente: Propia, 2023.

En la Ilustración 28, se exhibe el diagrama de secuencia que ilustra el proceso de eliminación de una actividad. En este proceso, se inicia solicitando la lista de actividades en la pantalla, la cual posteriormente es validada por el controlador de actividades. Después de la validación de la solicitud, se accede a la base de datos para obtener la lista de actividades, la cual se presenta al administrador. Seguidamente, el administrador introduce el identificador de la actividad, la pantalla se encarga de solicitar la validación para eliminar la actividad en el controlador, dicho controlador verifica si existe la actividad que se desea eliminar en la base de datos,

una vez pasada la validación, se ejecuta la eliminación de la actividad en la base de datos y se le muestra al usuario el mensaje de resultado del proceso.

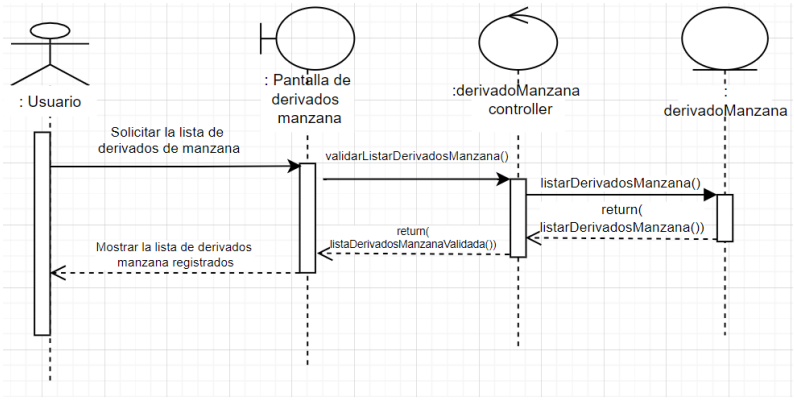
Ilustración 29.- Diagrama de secuencia para listar por id una actividad.



Fuente: Propia, 2023.

En la Ilustración 29 se describe el diagrama de secuencia para listar una actividad en específico. El proceso comienza solicitando la lista de actividades en la pantalla, la cual es luego validada por el controlador de actividades. Después de la validación de la solicitud, se recupera la lista de actividades desde la base de datos mediante su tabla correspondiente y se presenta al usuario. A continuación, el usuario introduce el id de la actividad, la pantalla solicita la validación de la actividad al controlador, el controlador de las actividades se encarga de verificar en la base de datos la existencia de la actividad solicitada, una vez validada, se solicitan los datos de dicha actividad a la base de datos, y se le muestra esa información al usuario.

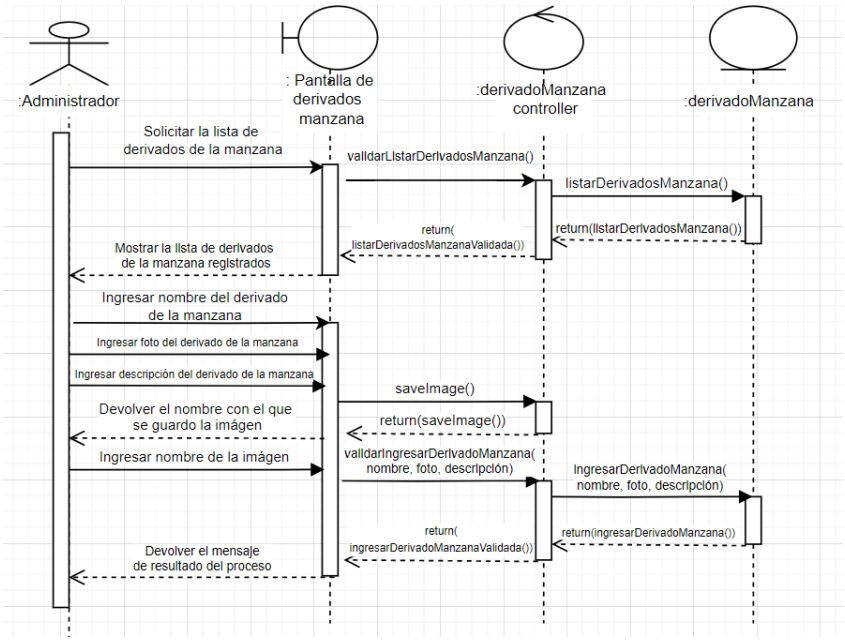
Ilustración 30.- Diagrama de secuencia para listar derivados de la manzana.



Fuente: Propia, 2023.

En la Ilustración 30 se presenta el diagrama de secuencia para visualizar los derivados de la manzana. En este proceso, el usuario solicita la lista de productos derivados de la manzana a la pantalla correspondiente. La pantalla, a su vez, pide los datos y validaciones al controlador de productos derivados de la manzana. Luego, se realiza la consulta de la lista desde la tabla correspondiente en la base de datos, y finalmente, se presenta al usuario la lista resultante.

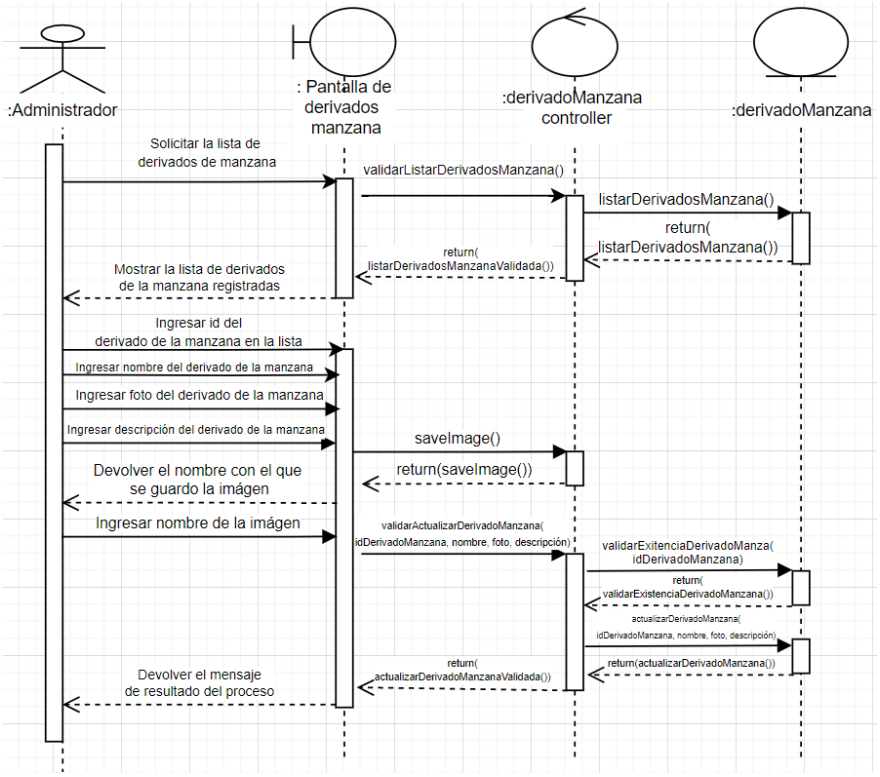
Ilustración 31.- Diagrama de secuencia para registrar un derivado de manzana.



Fuente: Propia, 2023.

En la Ilustración 31 se presenta el diagrama de secuencia para el registro de un derivado de la manzana. El proceso inicia solicitando la lista de derivados en la pantalla, la cual es posteriormente validada por el controlador de derivados. Tras la validación de la solicitud, se recupera la lista de derivados desde la base de datos y se presenta al administrador. Luego, el administrador introduce el nombre, la foto y la descripción del derivado. La imagen se guarda mediante el controlador, y se devuelve el nombre bajo el cual se ha almacenado la foto, junto con los datos previamente ingresados. Posteriormente, se valida la inserción del derivado en el controlador correspondiente, el cual registra el derivado de la manzana y finalmente devuelve un mensaje de resultado del proceso de inserción al administrador.

Ilustración 32.- Diagrama de secuencia para actualizar un derivado de manzana.

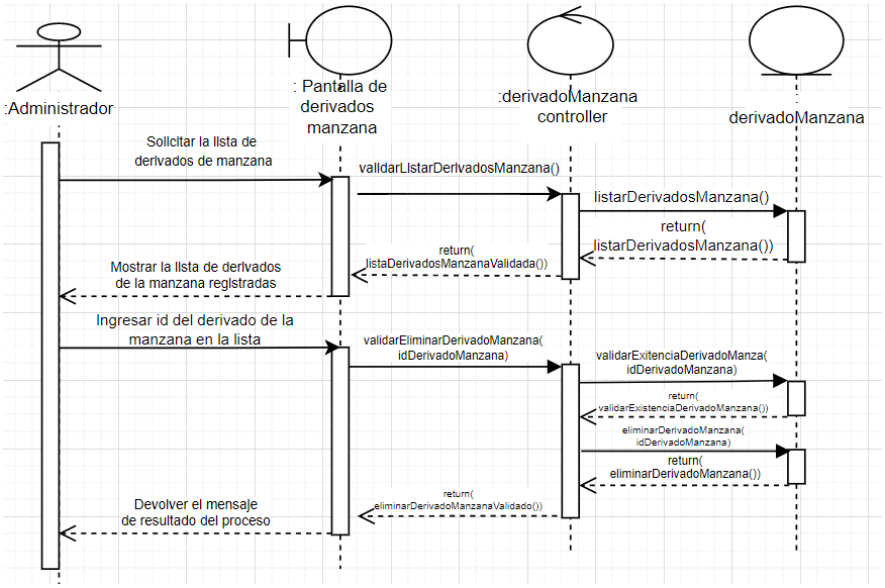


Fuente: Propia, 2023.

En la Ilustración 32, se exhibe el diagrama de secuencia correspondiente al procedimiento de actualización de un derivado de la manzana. Este proceso inicia

con la solicitud de la lista de derivados de manzana en la pantalla. Posteriormente, la lista se somete a validación por parte del controlador de productos. Tras la validación de la solicitud, se accede a la base de datos para obtener la lista actualizada de productos derivados de la manzana, la cual se presenta al administrador. A continuación, el administrador ingresa el identificador del derivado de la manzana, junto con el nombre, la foto y la descripción de dicho derivado. El controlador se encarga de gestionar y almacenar la foto, devolviendo el nombre bajo el cual se guarda dicha imagen. Este nombre, junto con los datos previamente ingresados, se envía como un conjunto al controlador para su validación. En este punto, se verifica la existencia del derivado que se desea modificar. Una vez confirmado esto, se procede a actualizar el derivado de manzana con los nuevos datos. Finalmente, se presenta al administrador el mensaje de resultado del proceso.

Ilustración 33.- Diagrama de secuencia para eliminar un derivado de manzana.

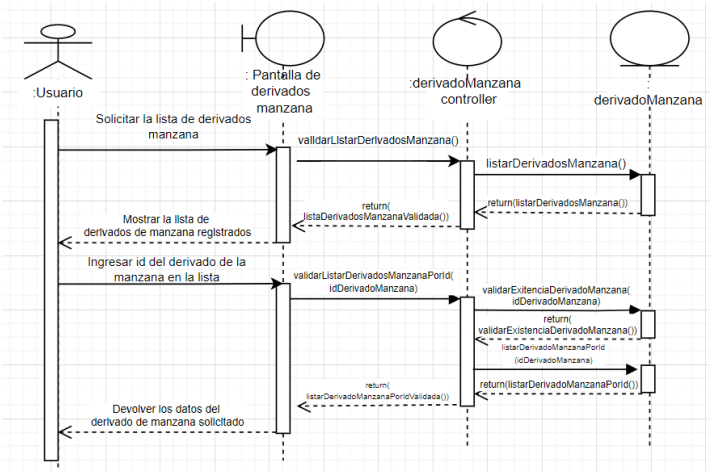


Fuente: Propia, 2023.

En la Ilustración 33 se presenta el diagrama de secuencia que detalla el proceso de eliminación de un derivado de la manzana. El procedimiento comienza solicitando la lista de estos derivados en la pantalla, la cual es luego validada por el controlador correspondiente. Después de validar la solicitud, se accede a la base de datos para obtener la lista de los derivados de la manzana, la cual se muestra al administrador.

Acto seguido, el administrador ingresa el identificador del derivado, y la pantalla gestiona la solicitud de validación para eliminar dicho derivado mediante el controlador. Este controlador verifica la existencia del derivado en la base de datos y, una vez superada la validación, ejecuta la eliminación del derivado en la base de datos. Finalmente, se presenta al usuario un mensaje que indica el resultado del proceso.

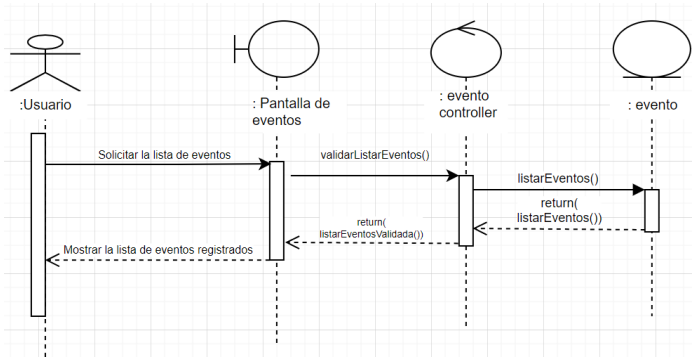
Ilustración 34.- Diagrama de secuencia para listar por id un derivado de manzana.



Fuente: Propia, 2023.

En la Ilustración 34 se detalla el diagrama de secuencia para la visualización de un derivado de la manzana en particular. El procedimiento se inicia al requerir la lista de derivados de la manzana en la pantalla, la cual se valida a través del controlador correspondiente. Tras validar la solicitud, se obtiene la lista de derivados desde la base de datos utilizando la tabla pertinente, y esta información se presenta al usuario. Luego, el usuario ingresa el identificador (id) del derivado de la manzana. La pantalla solicita la validación de dicho derivado al controlador, que se encarga de verificar su existencia en la base de datos. Una vez validado, se solicitan los datos asociados a ese derivado específico a la base de datos, y esta información se muestra al usuario.

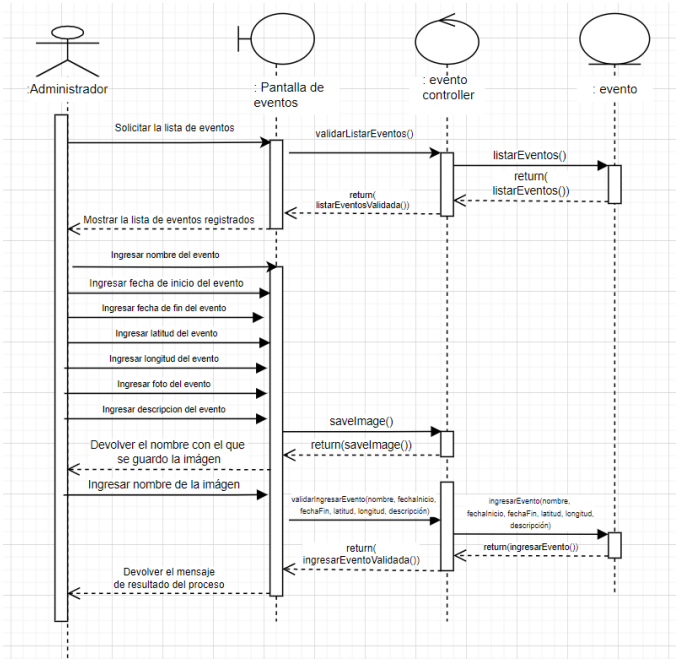
Ilustración 35.- Diagrama de secuencia para listar eventos.



Fuente: Propia, 2023.

En la Ilustración 35 se expone el diagrama de secuencia destinado a la visualización de eventos. En este procedimiento, el usuario requiere la lista de eventos a la pantalla correspondiente. La pantalla, a su vez, solicita los datos y las validaciones al controlador de eventos. Posteriormente, se lleva a cabo la consulta de la lista desde la tabla correspondiente en la base de datos, culminando con la presentación al usuario de la lista resultante.

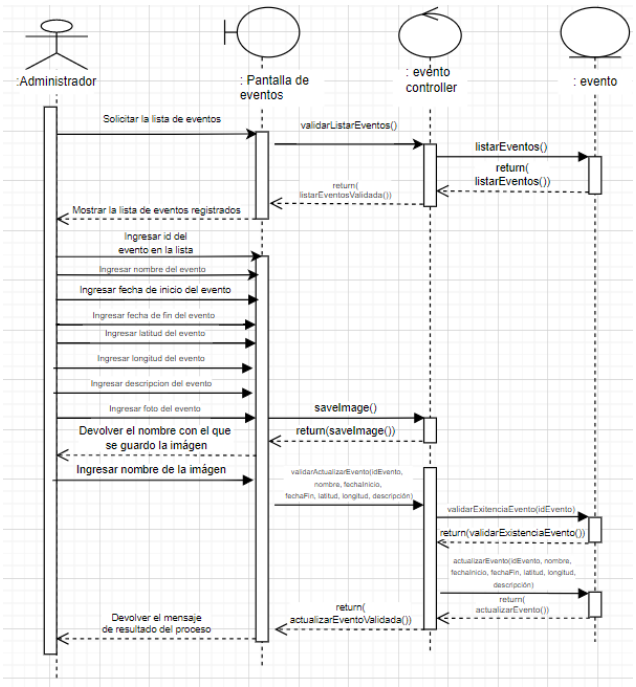
Ilustración 36.- Diagrama de secuencia para registrar un evento.



Fuente: Propia, 2023.

En la Ilustración 36 se exhibe el diagrama de secuencia para el registro de un evento. El procedimiento comienza solicitando la lista de eventos en la pantalla, la cual es luego validada por el controlador de eventos. Después de la validación de la solicitud, se recupera la lista de eventos desde la base de datos y se presenta al administrador. Seguidamente, el administrador introduce el nombre, la fecha de inicio, fecha de fin, las coordenadas, la foto y la descripción del evento. La imagen se guarda a través del controlador, y se devuelve el nombre bajo el cual se ha almacenado la foto, junto con los datos previamente ingresados. A continuación, se verifica la inserción del evento en el controlador correspondiente, el cual registra el evento en la base de datos y finalmente envía un mensaje que indica el resultado del proceso de inserción al administrador.

Ilustración 37.- Diagrama de secuencia para actualizar un evento.

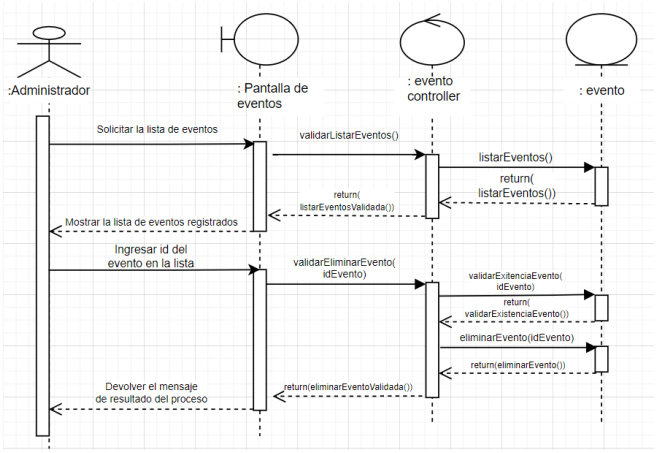


Fuente: Propia, 2023

En la Ilustración 37, se presenta el diagrama de secuencia que detalla el proceso de actualización de un evento. El procedimiento comienza con la solicitud de la lista de eventos en la pantalla, la cual es luego validada por el controlador de eventos. Después de la validación de la solicitud, se accede a la base de datos para obtener

la lista actualizada de eventos, la cual se presenta al administrador. Seguidamente, el administrador introduce el identificador del evento, así como el nombre, la fecha de inicio, fecha de fin, las coordenadas, la foto y la descripción del evento. La gestión y almacenamiento de la foto son manejados por el controlador, que devuelve el nombre asociado a la imagen almacenada. Este nombre, junto con los datos ingresados previamente, se envía como un conjunto al controlador para su validación. En este punto, se verifica la existencia del evento que se desea modificar. Una vez confirmado esto, se procede a actualizar el evento con los nuevos datos. Finalmente, se muestra al administrador el mensaje de resultado del proceso.

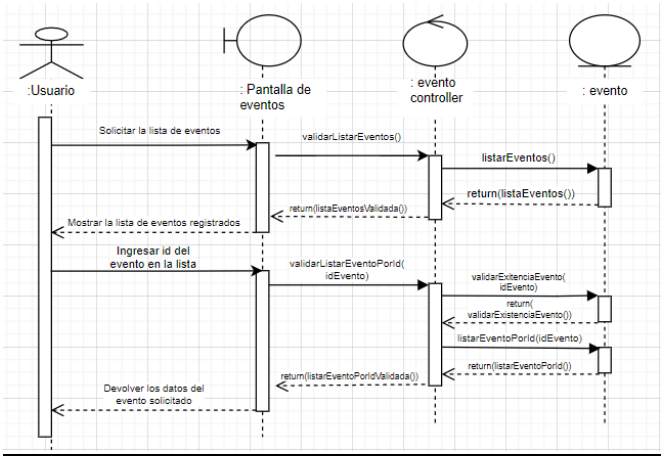
Ilustración 38.- Diagrama de secuencia para eliminar un evento.



Fuente: Propia, 2023.

En la Ilustración 38 se expone el diagrama de secuencia que describe el proceso de eliminar un evento. La secuencia se inicia al solicitar la lista de eventos en la pantalla, la cual es posteriormente validada por el controlador correspondiente. Después de validar la solicitud, se accede a la base de datos para recuperar la lista de eventos, presentándola al administrador. A continuación, el administrador proporciona el identificador del evento, y la pantalla gestiona la solicitud de validación para eliminar dicho evento mediante el controlador. Este controlador verifica la existencia del evento en la base de datos y, una vez superada la validación, procede a eliminar el evento en la base de datos. Por último, se muestra al usuario un mensaje que refleja el resultado del proceso.

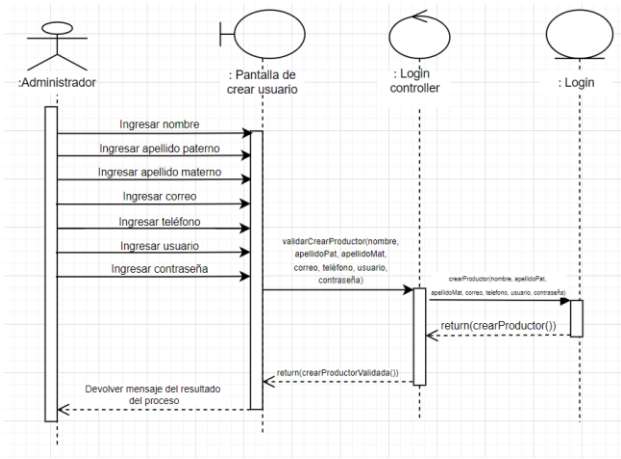
Ilustración 39.- Diagrama de secuencia para listar por id un evento.



Fuente: Propia, 2023.

En la Ilustración 39 se describe el diagrama de secuencia para la visualización de un evento en particular. El proceso comienza al requerir la lista de eventos en la pantalla, la cual es validada a través del controlador correspondiente. Después de validar la solicitud, se recupera la lista de eventos desde la base de datos utilizando la tabla pertinente, y esta información se presenta al usuario. A continuación, el usuario introduce el identificador (id) del evento. La pantalla solicita la validación de dicho evento al controlador, encargado de verificar su existencia en la base de datos. Una vez validado, se solicitan los datos asociados a ese evento específico a la base de datos, y esta información se muestra al usuario.

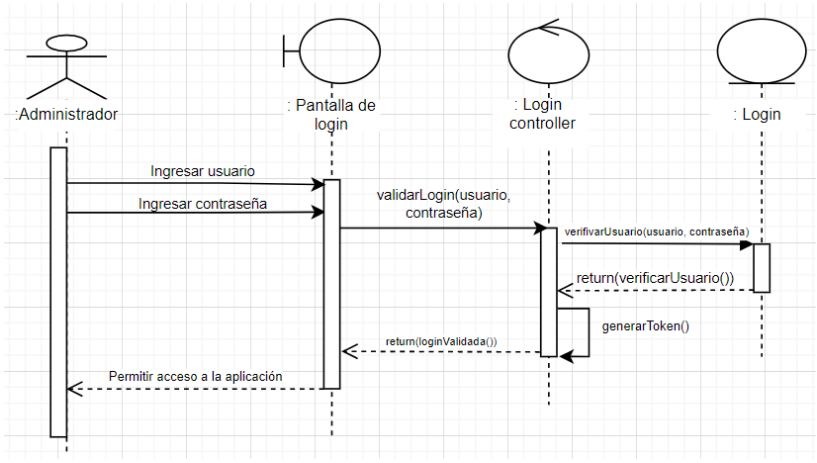
Ilustración 40.- Diagrama de secuencia para crear un usuario.



Fuente: Propia, 2023.

En la ilustración 40 se aprecia el diagrama de secuencia referente a la creación de un usuario. En el diagrama el administrador ingresa en la pantalla el nombre, apellido paterno, apellido materno, correo, teléfono, usuario, y contraseña del usuario que se plantea crear. Posteriormente, el usuario manda estos datos para su validación por medio del controlador de inicio se sesión, una vez validados los datos se crear el usuario en la base de datos, y finalmente, se devuelve el mensaje de resultado del proceso al administrador.

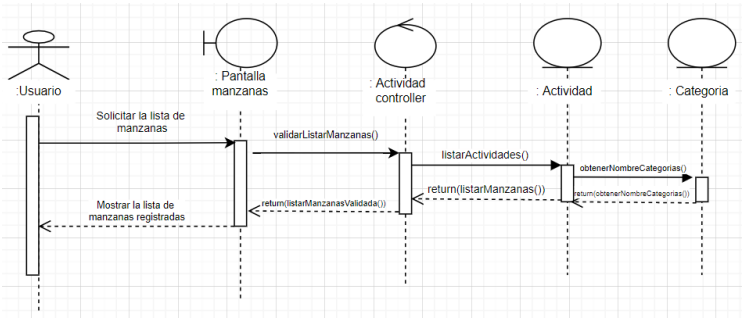
Ilustración 41.- Diagrama de secuencia para validar un usuario.



Fuente: Propia, 2023.

En la Ilustración 41 se muestra el diagrama de secuencia con el cual se valida el inicio de sesión. El administrador inicia ingresando el usuario y la contraseña en la pantalla de inicio de sesión, a continuación, la pantalla solicita la validación de los datos en el controlador, el controlador valida los datos en la base de datos, una vez validados los datos el controlador genera el token con el cual se puede acceder a las partes de administración a la aplicación, y finalmente se le da acceso al administrador.

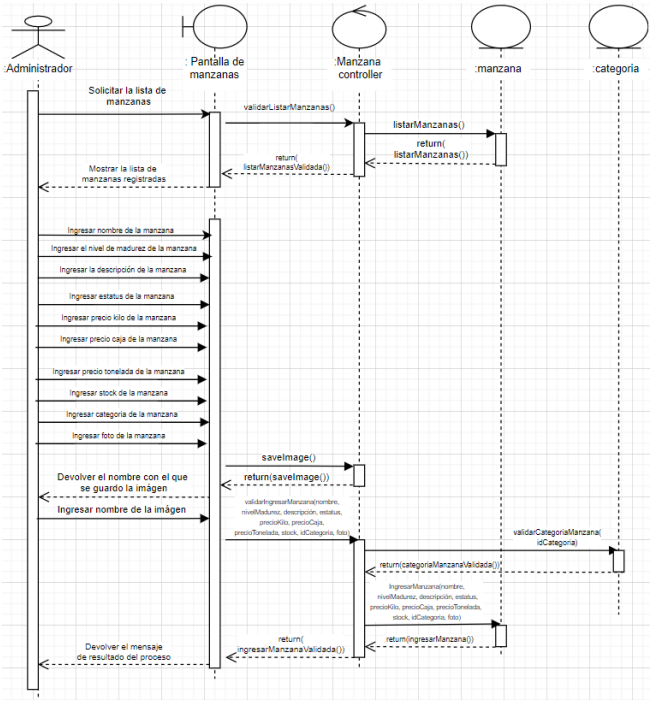
Ilustración 42.- Diagrama de secuencia para listar manzanas.



Fuente: Propia, 2023.

En la Ilustración 42 se presenta el diagrama de secuencia orientado a la exhibición de información sobre manzanas. En este proceso, el usuario busca obtener la lista de manzanas en la pantalla correspondiente. La pantalla, a su vez, solicita los datos y las validaciones al controlador de manzanas. Luego, se realiza la consulta de la lista desde la tabla correspondiente en la base de datos, además de traer la categoría de dichas manzanas desde la tabla de categorías, concluyendo con la presentación al usuario de la lista resultante.

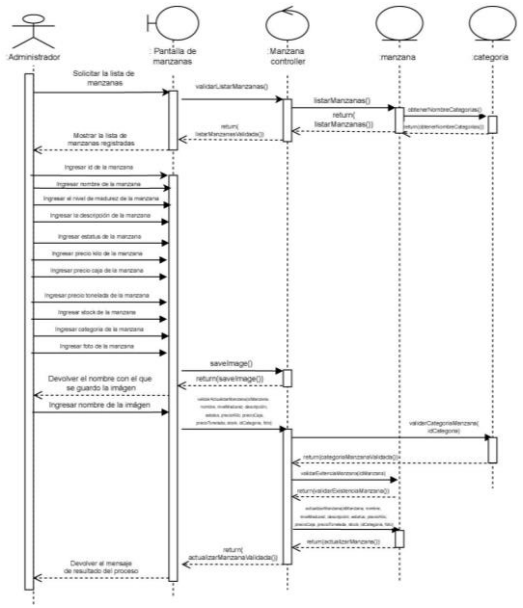
Ilustración 43.- Diagrama de secuencia para registrar una manzana.



Fuente: Propia, 2023.

En la Ilustración 43 se revela el diagrama de secuencia enfocado en la presentación de información acerca de manzanas. En este procedimiento, el administrador busca obtener el listado de manzanas en la pantalla designada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para manzanas. Después, se lleva a cabo la consulta de la lista desde la tabla correspondiente en la base de datos, conjuntamente con la recuperación de la categoría de estas manzanas desde la tabla de categorías. Posteriormente, se presenta al administrador la lista resultante, el administrador ingresa los datos que desea que tenga la manzana, dentro de estos datos el administrador manda una foto, dicha foto es enviada por la pantalla y almacenada en el controlador para ser almacenada en el mismo devolviendo un nombre que será mandado de nuevo junto con los datos de la manzana para su validación en el controlador, el mismo se encarga de validar los datos, además de verificar si la categoría de la manzana existe, una vez verificado todo esto se registra la actividad, finalmente, se devuelve el mensaje de resultado del proceso al administrador.

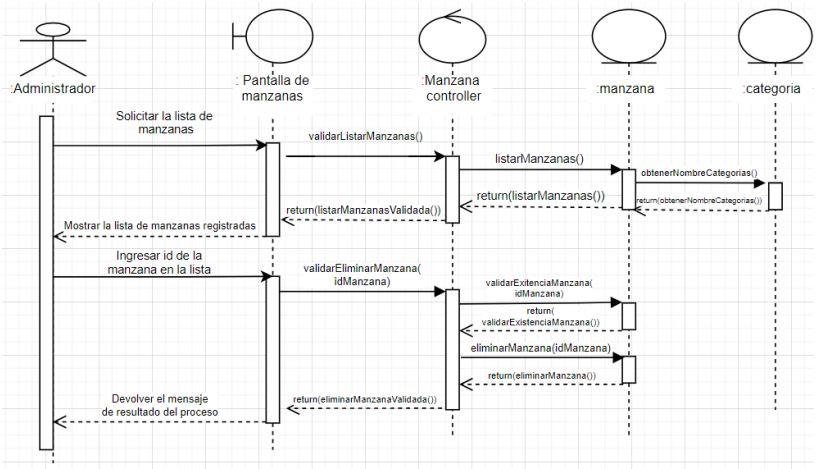
Ilustración 44.- Diagrama de secuencia para actualizar una manzana.



Fuente: Propia, 2023.

En la Ilustración 44 se expone el diagrama de secuencia centrado en la actualización de información sobre manzanas. En este proceso, el administrador busca obtener el listado de manzanas en la pantalla designada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para manzanas. Después, se lleva a cabo la consulta de la lista desde la tabla correspondiente en la base de datos, junto con la recuperación de la categoría de estas manzanas desde la tabla de categorías. A continuación, se presenta al administrador la lista resultante. El administrador, entonces, introduce los datos deseados para la actualización de la manzana, incluyendo el id de la manzana en cuestión y el envío de una foto. Esta imagen es transmitida por la pantalla y almacenada en el controlador para su registro, devolviendo un nombre que se envía de vuelta junto con los datos de la manzana para su validación en el controlador. Este último se encarga de validar los datos y verificar la existencia de la categoría de la manzana, así como de la manzana que se desea actualizar. Una vez verificado todo esto, se lleva a cabo la actualización, y finalmente, se devuelve al administrador el mensaje de resultado del proceso.

Ilustración 45.- Diagrama de secuencia para eliminar una manzana.

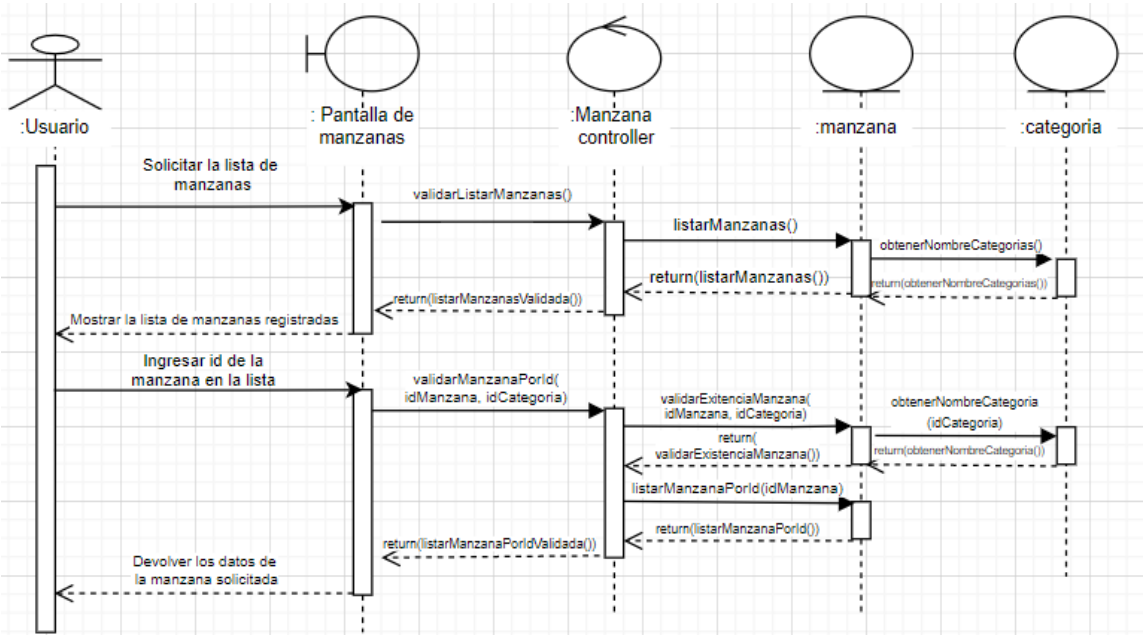


Fuente: Propia, 2023.

En la Ilustración 45 se expone el diagrama de secuencia centrado en la eliminación de manzanas. Durante este proceso, el administrador busca obtener el listado

actualizado de manzanas en la pantalla asignada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para manzanas. Luego, se realiza la consulta de la lista desde la tabla correspondiente en la base de datos, junto con la recuperación de la categoría de estas manzanas desde la tabla de categorías. Seguidamente, se presenta al administrador la lista resultante. El administrador ingresa en la pantalla el id de la manzana para eliminar, el identificador de la manzana es mandado al controlador correspondiente para su validación. El controlador se encarga de validar los datos y verificar la existencia de la manzana. Una vez verificado todo esto, se lleva a cabo la eliminación de la manzana, concluyendo con el envío del mensaje de resultado del proceso al administrador.

Ilustración 46.- Diagrama de secuencia para listar por id una manzana.

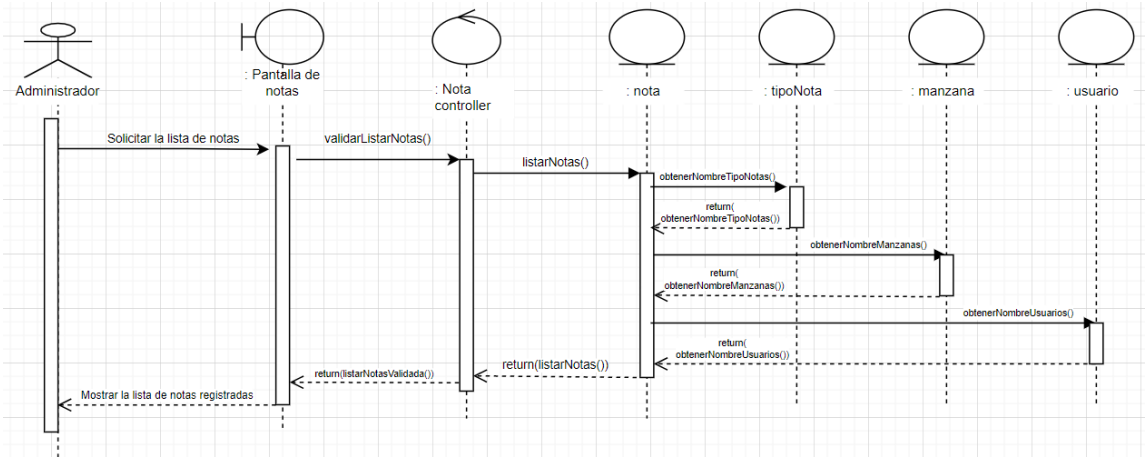


Fuente: Propia, 2023.

En la Ilustración 46 se muestra el diagrama de secuencia enfocado en listar la información de una manzana mediante su identificador. En este proceso, el administrador busca obtener la lista actualizada de manzanas en la pantalla designada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para manzanas. Posteriormente, se realiza la consulta de la lista desde la tabla correspondiente en la base de datos, incluyendo la recuperación de la categoría

de estas manzanas desde la tabla de categorías. Después de ello, se presenta al administrador la lista actualizada. El administrador introduce en la pantalla el identificador de la manzana a buscar, y este identificador se envía al controlador correspondiente para su validación. El controlador se encarga de validar los datos y verificar la existencia de la manzana junto con su categoría. Una vez completada esta validación, se procede a obtener los datos de la manzana, concluyendo con el envío de un mensaje que informa al administrador sobre el resultado del proceso.

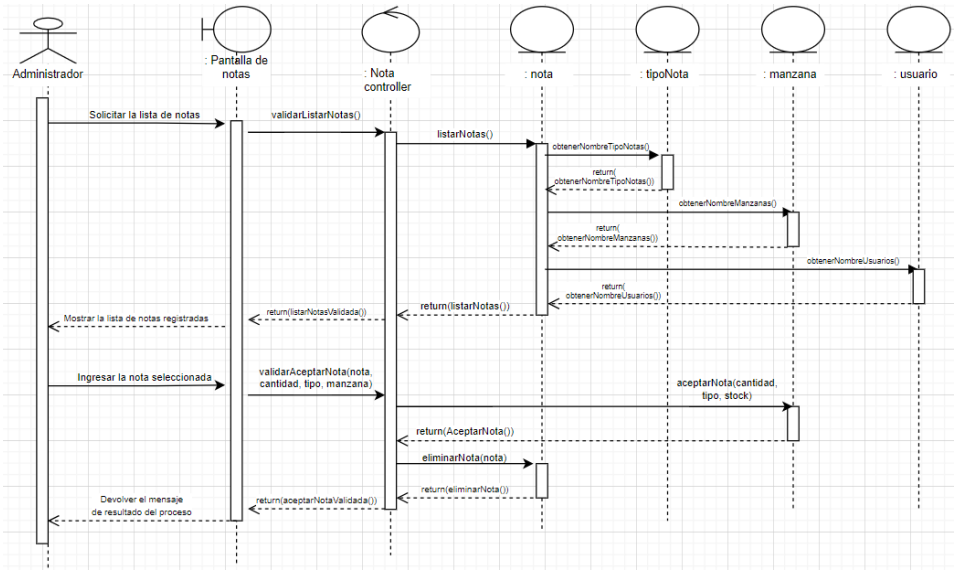
Ilustración 47.-Diagrama de secuencia para listar notas.



Fuente: Propia, 2023.

En la Ilustración 47 se muestra el diagrama de secuencia enfocado en la presentación de información sobre notas. En este procedimiento, el administrador busca obtener el listado de notas en la pantalla designada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para notas. Posteriormente, se lleva a cabo la consulta de la lista de las notas desde la tabla correspondiente de nota en la base de datos, además de contener el tipo de la nota, la manzana, y el usuario que solicita la nota, todas y cada una de sus respectivas tablas, culminando con la presentación al administrador de la lista resultante.

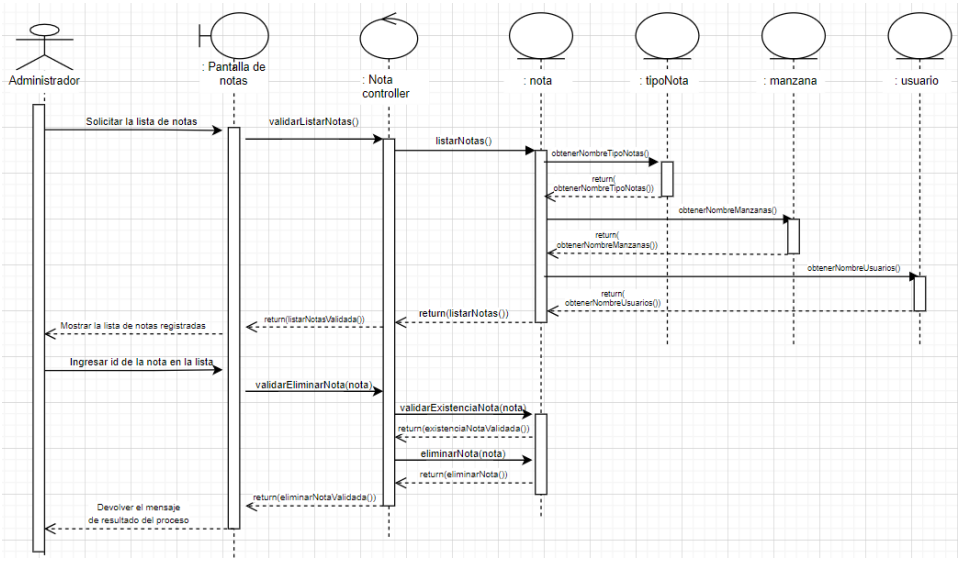
Ilustración 48.- Diagrama de secuencia para aceptar una nota.



Fuente: Propia, 2023.

En la Ilustración 48 se presenta el diagrama de secuencia centrado en la aceptación de notas. Durante este proceso, el administrador busca obtener el conjunto de notas en la pantalla asignada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para notas. A continuación, se realiza la consulta de la lista de notas desde la tabla correspondiente en la base de datos, incluyendo el tipo de nota, la referencia a la manzana y el usuario que solicita la nota, todos provenientes de sus respectivas tablas. Posteriormente se le presenta al administrador la lista resultante y el mismo manda la nota desde la pantalla al controlador con los datos de la nota, el controlador valida los datos y modifica el stock de la manzana de la nota, además, elimina la nota en cuestión una vez realizada, finalmente se devuelve el mensaje del proceso terminado.

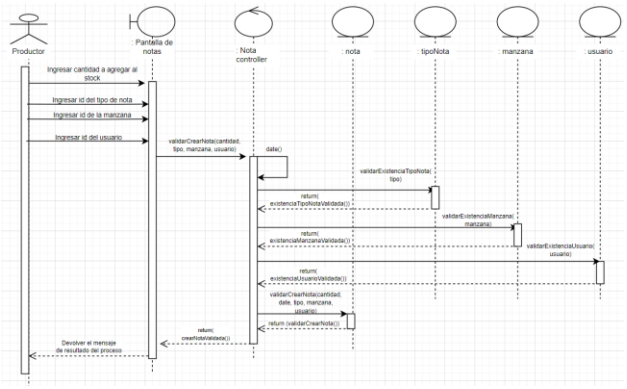
Ilustración 49.- Diagrama de secuencia para rechazar una nota.



Fuente: Propia, 2023

En la Ilustración 49 se muestra el diagrama de secuencia enfocado en la eliminación o rechazo de notas. En este proceso, el administrador intenta obtener el conjunto de notas en la pantalla designada. La pantalla, a su vez, solicita los datos y validaciones al controlador especializado para notas. Posteriormente, se lleva a cabo la consulta de la lista de notas desde la tabla correspondiente en la base de datos, incluyendo el tipo de nota, la referencia a la manzana y el usuario que solicitó la nota, todos provenientes de sus respectivas tablas. Posteriormente se le presenta al administrador la lista resultante y el mismo manda el identificador de la nota en la pantalla, la cual a su vez solicita la validación a su controlador correspondiente, el controlador verifica la existencia de la nota en la base de datos, y finalmente ejecuta la eliminación devolviendo al administrador el mensaje de resultado del proceso.

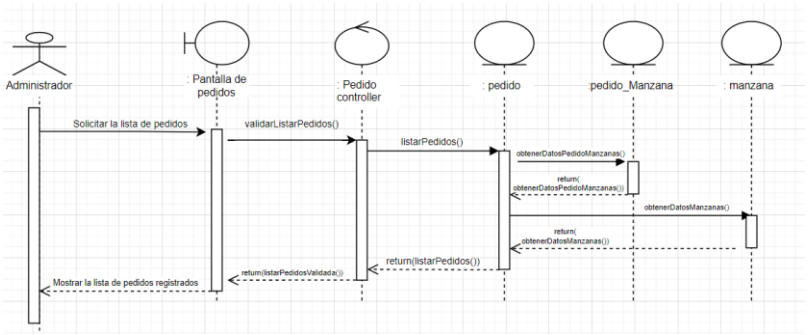
Ilustración 50.- Diagrama de secuencia para crear una nota.



Fuente: Propia, 2023.

En la Ilustración 50 se presenta el diagrama de secuencia para crear una nota, dicho proceso inicia cuando el productor ingresa en la pantalla la cantidad de stock, si es para agregar o eliminar del mismo como el tipo de nota, el identificador de la manzana a la cual hace referencia la nota, además del identificador del usuario que hace la nota, posteriormente, la pantalla manda estos datos al controlador para su validación, el controlador valida los datos, además de usar la función date() para obtener la fecha en la que se hace la nota, válida la existencia del tipo de nota, la manzana, y el usuario en sus respectivas tablas, finalmente, crea el usuario y devuelve el resultado del proceso.

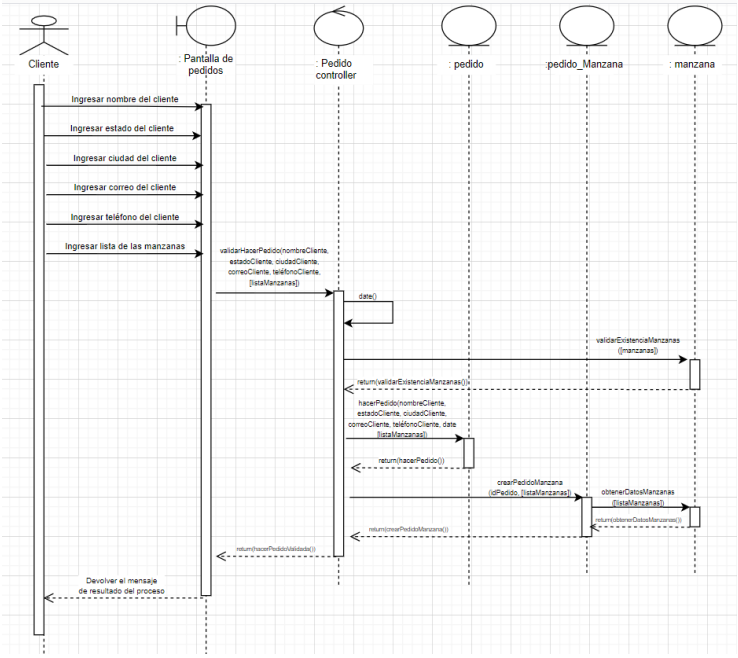
Ilustración 51.- Diagrama de secuencia para listar pedidos.



Fuente: Propia, 2023

En la Ilustración 51 se presenta el diagrama de secuencia orientado a la exhibición de información sobre pedidos. En este procedimiento, el administrador intenta obtener el listado de pedidos en la pantalla asignada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para pedidos. Posteriormente, se realiza la consulta de la lista de pedidos desde la tabla correspondiente en la base de datos, incluyendo los datos de la tabla intermedia entre el pedido y la manzana, y los datos de la propia manzana, todos provenientes de sus respectivas tablas. El proceso culmina con la presentación al administrador de la lista resultante.

Ilustración 52.- Diagrama de secuencia para registrar un pedido.

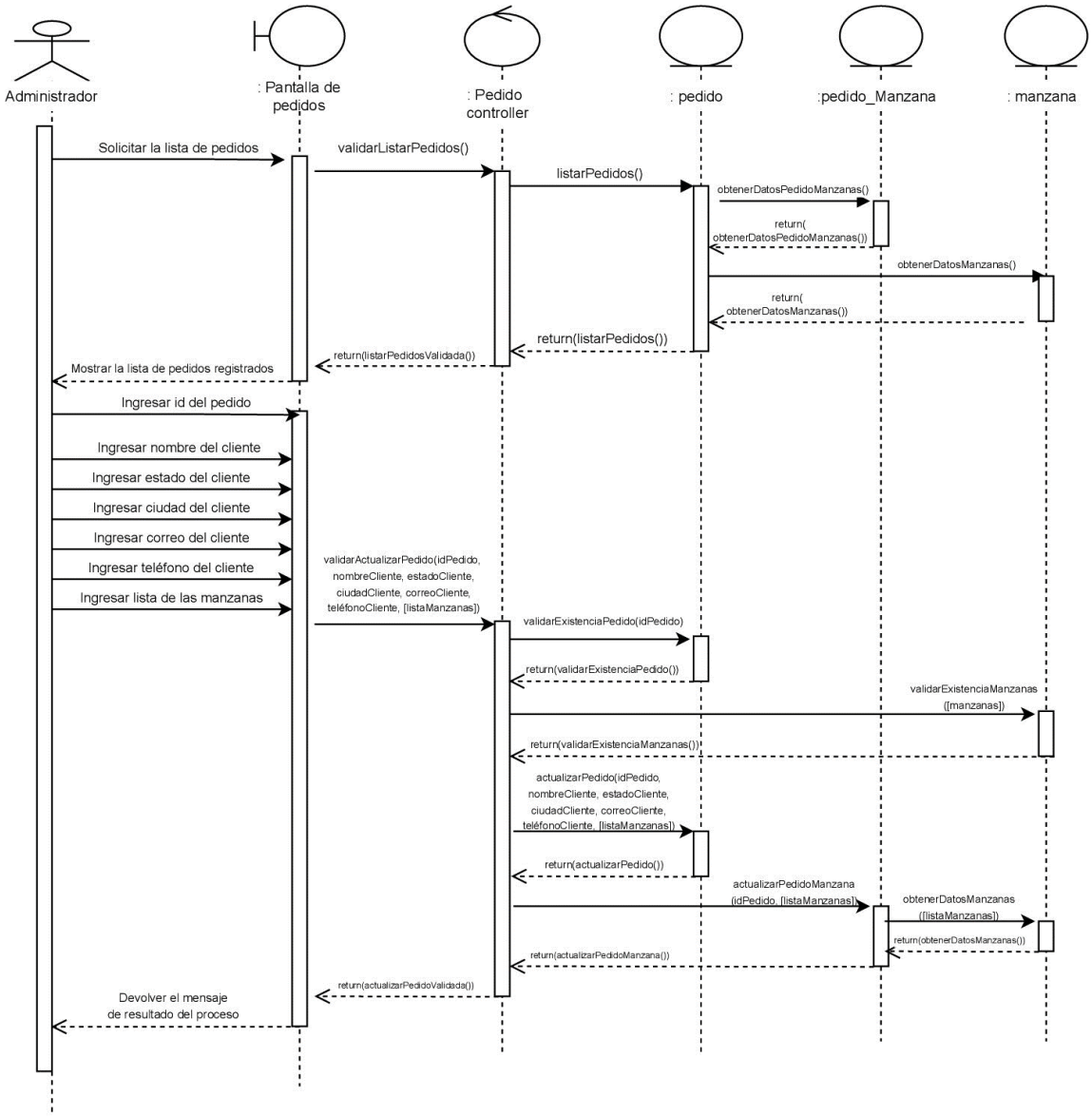


Fuente: Propia, 2023.

En la Ilustración 52 se exhibe el diagrama de secuencia centrado en el registro de un pedido. Durante este proceso, el cliente empieza ingresando los datos de su pedido en la pantalla, dichos datos son mandados por la pantalla al controlador para su validación, una vez validados el controlador procede a calcular la fecha con la función date(), posteriormente válida la existencia de las manzanas en el pedido, una vez que se valida este dato el controlador procede a registrar el pedido en la base de datos, así como insertar en la tabla auxiliar entre los pedidos y la manzana

los datos correspondientes de cada manzana en el pedido, finalmente, se le regresa el mensaje de resultado del proceso al cliente.

Ilustración 53.- Diagrama de secuencia para actualizar un pedido.

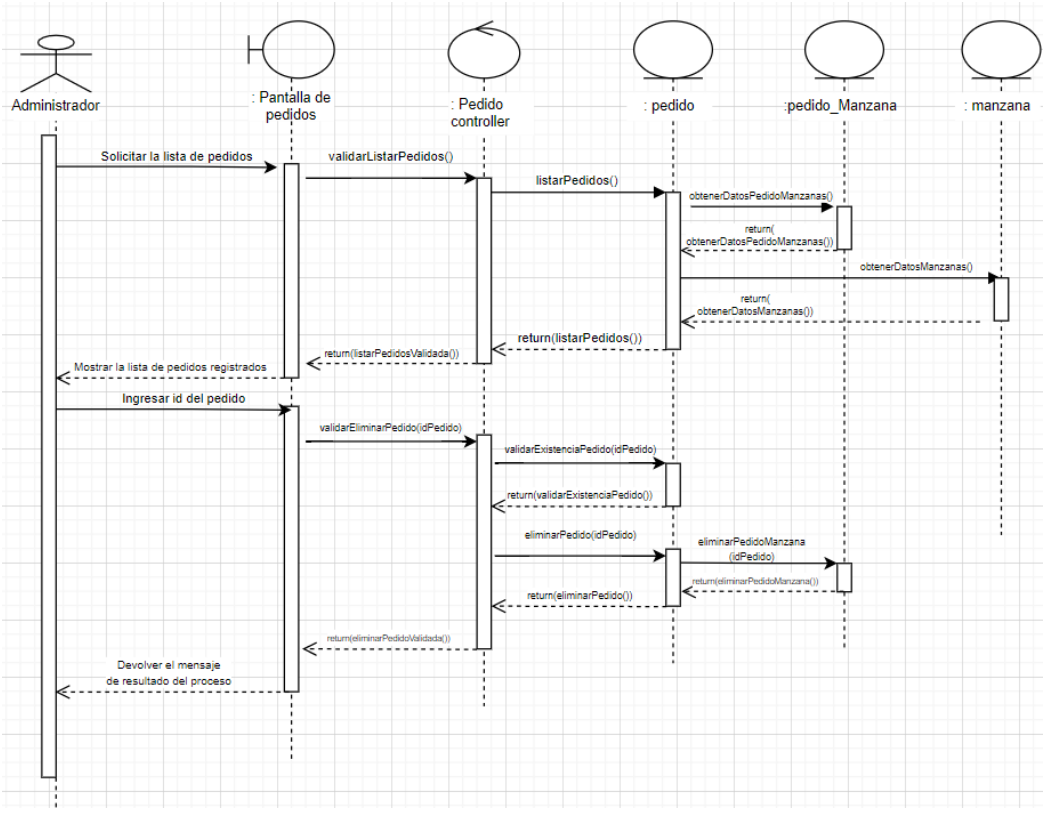


Fuente: Propia, 2023.

En la Ilustración 53 se exhibe el diagrama de secuencia centrado en actualización de información de los pedidos. Durante este proceso, el administrador busca obtener el conjunto de pedidos en la pantalla designada. La pantalla, a su vez, solicita los datos y validaciones al controlador específico para pedidos. A continuación, se lleva a cabo la consulta de la lista de pedidos desde la tabla correspondiente en la base

de datos, incluyendo la información de la tabla intermedia entre el pedido y el producto, así como los datos del propio producto, todos obtenidos de sus respectivas tablas. Continúa la secuencia con la presentación al administrador de la lista resultante, el administrador ingresa en la pantalla los datos nuevos del pedido junto con el identificador del pedido, estos datos son validados por el controlador, el controlador válida los datos junto con la existencia del pedido y la existencia de la manzana, una vez que se compruebe la existencia el controlador actualiza el pedido en su tabla correspondiente y también en la tabla intermedia entre los pedidos y la manzana, donde se obtienen los datos de las nuevas manzanas en el pedido, finalmente, se le devuelve al administrador el resultado del proceso.

Ilustración 54.- Diagrama de secuencia para eliminar un pedido.

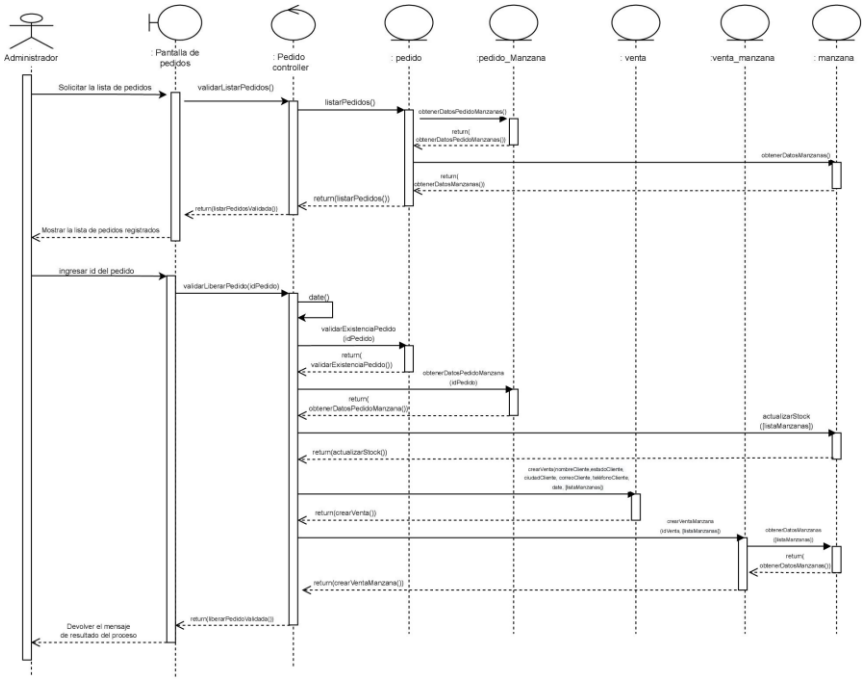


Fuente: Propia, 2023.

En la Ilustración 54 se presenta el diagrama de secuencia enfocado en la eliminación de un pedido. En este procedimiento, el administrador intenta obtener el conjunto de pedidos en la pantalla asignada. La pantalla, a su vez, solicita los datos y

validaciones al controlador específico para pedidos. Posteriormente, se realiza la consulta de la lista de pedidos desde la tabla correspondiente en la base de datos, incluyendo la información de la tabla intermedia entre el pedido y el producto, así como los datos propios del producto, todos extraídos de sus respectivas tablas. La secuencia continúa con la presentación al administrador de la lista resultante. El administrador introduce en la pantalla el identificador del pedido que se desea eliminar. Estos datos son sometidos a validación por el controlador, quien verifica la existencia del pedido. Una vez confirmada la existencia, el controlador elimina el pedido y a su vez se eliminan los registros en la tabla intermedia referente a ese pedido. Finalmente, se comunica al administrador el resultado del proceso.

Ilustración 55.- Diagrama de secuencia para liberar un pedido.

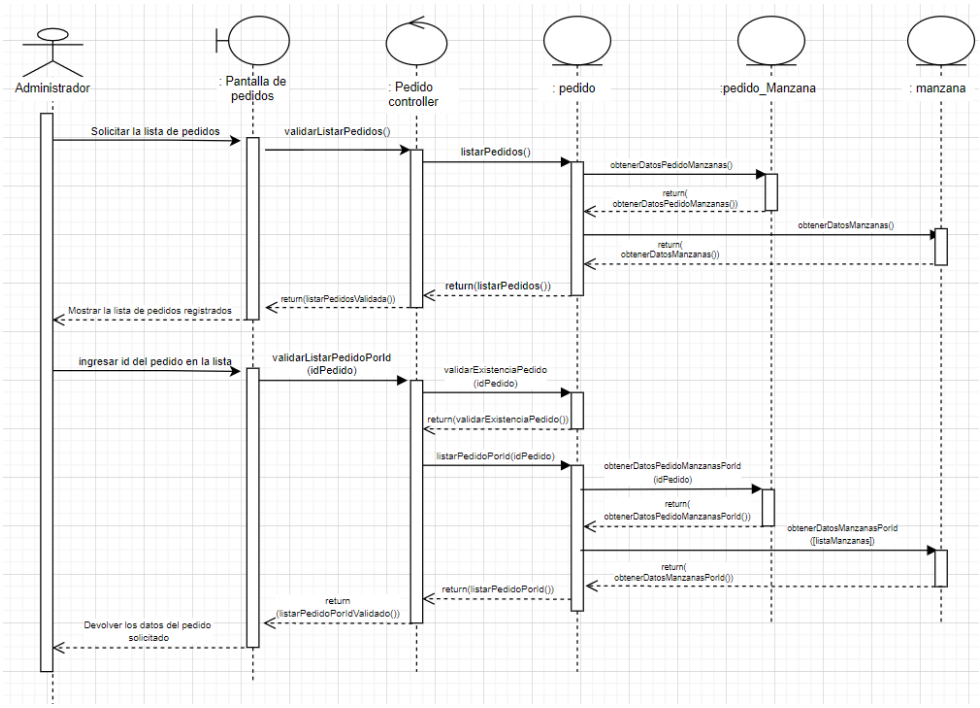


Fuente: Propia, 2023.

En la Ilustración 55 se presenta el diagrama de secuencia enfocado en la liberación de un pedido. En este procedimiento, el administrador intenta obtener el conjunto de pedidos en la pantalla asignada. La pantalla, a su vez, solicita los datos y validaciones al controlador específico para pedidos. Posteriormente, se realiza la consulta de la lista de pedidos desde la tabla correspondiente en la base de datos,

incluyendo la información de la tabla intermedia entre el pedido y el producto, así como los datos propios del producto, todos extraídos de sus respectivas tablas. La secuencia continúa con la presentación al administrador de la lista resultante. El administrador introduce en la pantalla el identificador del pedido que se desea liberar. Estos datos son sometidos a validación por el controlador, quien obtiene la fecha de liberación del pedido, además verifica la existencia del pedido. Una vez confirmada la existencia el controlador trae los datos de las manzanas de ese pedido en la clase intermedia entre pedido y manzana, el controlador se encarga de modificar el stock de las manzanas, también, crea una venta con los datos que tiene el pedido y la fecha en la que se está liberando el mismo, así como registrar los datos pertinentes a las manzanas en la venta dentro de su propia clase intermedia entre manzanas y ventas eliminando el pedido. Finalmente, se comunica al administrador el resultado del proceso.

Ilustración 56.- Diagrama de secuencia para listar por id un pedido.

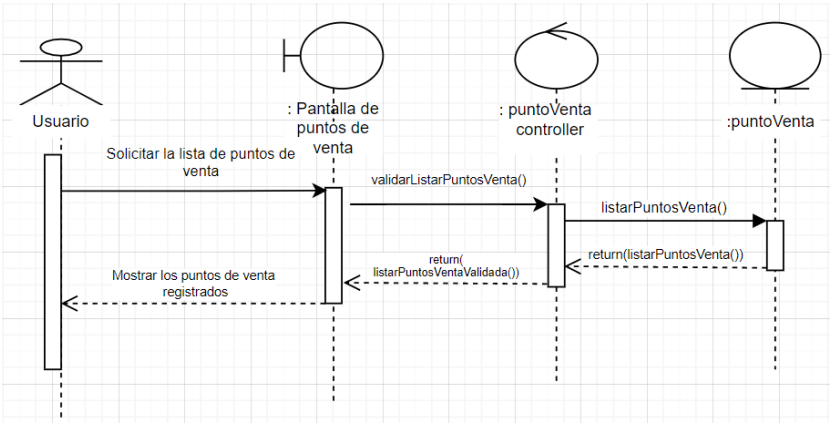


Fuente: Propia, 2023.

En la Ilustración 56 se exhibe el diagrama de secuencia focalizado en mostrar la información de un pedido por medio de su identificador. Durante este proceso, el

administrador busca obtener el conjunto de pedidos en la pantalla asignada. La pantalla, a su vez, solicita los datos y validaciones al controlador específico para pedidos. Posteriormente, se efectúa la consulta de la lista de pedidos desde la tabla correspondiente en la base de datos, incluyendo la información de la tabla intermedia entre el pedido y el producto, así como los datos propios del producto, todos extraídos de sus respectivas tablas. La secuencia continúa con la presentación al administrador de la lista resultante. El administrador ingresa en la pantalla el identificador del pedido que desea visualizar sus datos. Estos datos son validados por el controlador, quien verifica la existencia del pedido. Una vez confirmada la existencia, el controlador procede a eliminar el pedido y, simultáneamente, se eliminan los registros en la tabla intermedia relacionada con ese pedido. Para concluir, se comunica al administrador el resultado del proceso.

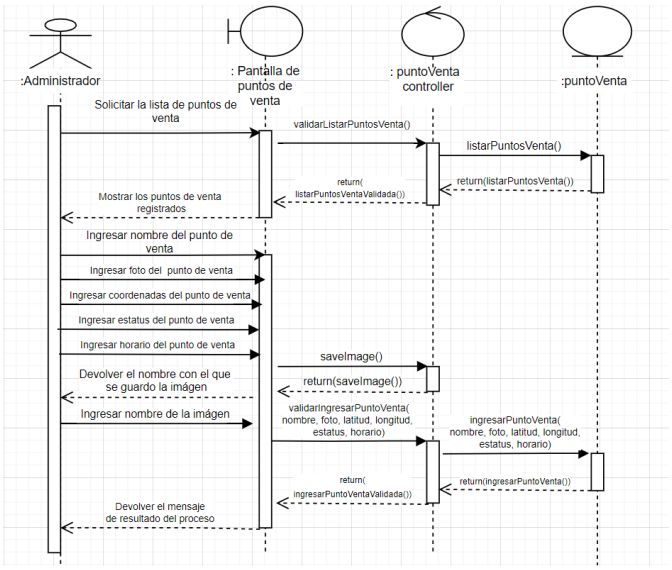
Ilustración 57.- Diagrama de secuencia para listar los puntos de venta.



Fuente: Propia, 2023.

En la Ilustración 57 se presenta el diagrama de secuencia enfocado en la visualización de puntos de venta. En este proceso, el usuario busca obtener el listado de puntos de venta en la pantalla asignada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para puntos de venta. Posteriormente, se realiza la consulta de la lista desde la tabla correspondiente en la base de datos, concluyendo con la presentación al usuario de la lista resultante.

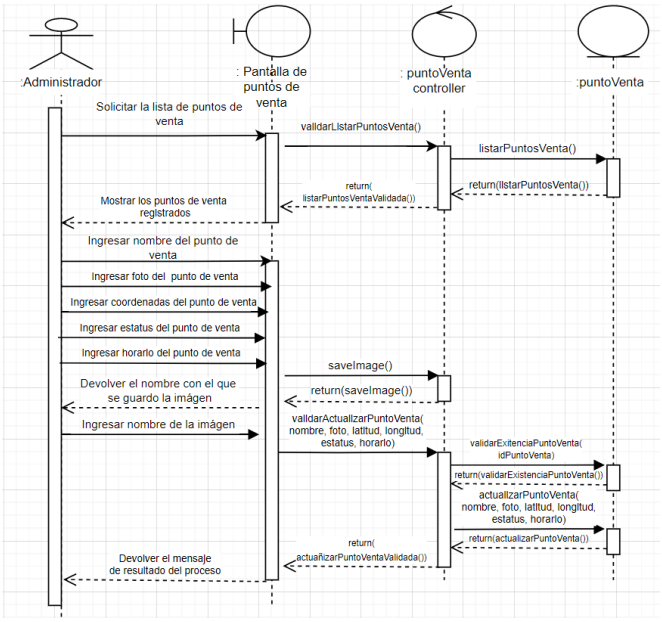
Ilustración 58.- Diagrama de secuencia para registrar un punto de venta.



Fuente: Propia, 2023.

En la Ilustración 58 se presenta el diagrama de secuencia para el registro de un punto de venta. El procedimiento inicia con el administrador solicitando la lista de puntos de venta en la pantalla, la cual es validada por el controlador de puntos de venta. Después de validar la solicitud, se obtiene la lista de puntos de venta desde la base de datos y se muestra al administrador. Luego, el administrador ingresa el nombre, foto, coordenadas, estatus, y horario del evento. La imagen se almacena a través del controlador, y se devuelve el nombre bajo el cual se ha guardado la foto, junto con los datos ingresados previamente se vuelven a enviar al controlador para su validación. Posteriormente, se verifica la inserción del punto de venta en el controlador correspondiente, el cual registra la información en la base de datos y finalmente envía un mensaje que indica el resultado del proceso de inserción al administrador.

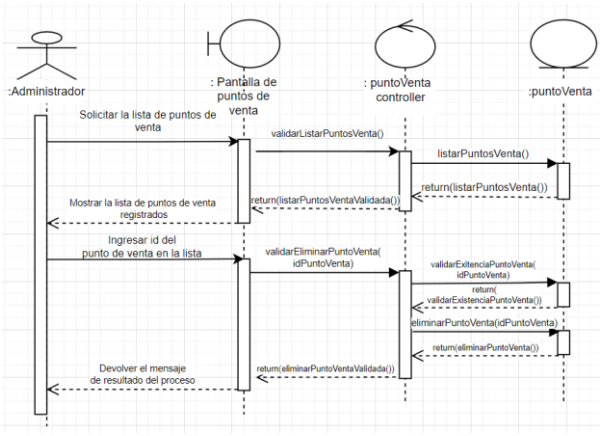
Ilustración 59.- Diagrama de secuencia para la actualización de un punto de venta.



Fuente: Propia, 2023.

En la Ilustración 59 se exhibe el diagrama de secuencia que describe el proceso de actualización de un punto de venta. El proceso se inicia con la solicitud de la lista de puntos de venta en la pantalla, la cual es posteriormente validada por el controlador específico para puntos de venta. Después de validar la solicitud, se accede a la base de datos para obtener la lista actualizada de puntos de venta, que se presenta al administrador. A continuación, el administrador ingresa el identificador del punto de venta, así como los nuevos datos que se desea para el punto de venta seleccionado. El controlador gestiona y almacena la foto, devolviendo el nombre asociado a la imagen almacenada. Este nombre, junto con los datos ingresados previamente, se envía como un conjunto al controlador para su validación. En este punto, se verifica la existencia del punto de venta que se desea modificar. Una vez confirmado esto, se procede a actualizar el punto de venta con los nuevos datos. Finalmente, se muestra al administrador el mensaje de resultado del proceso.

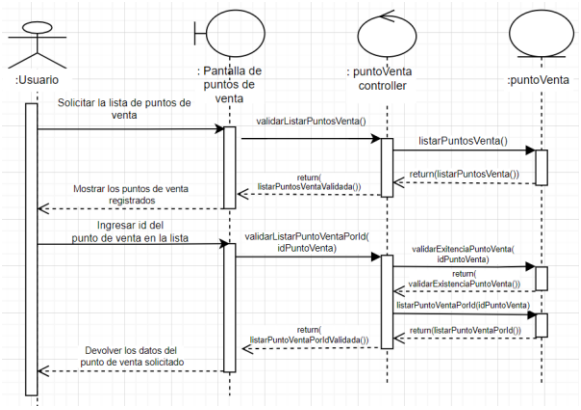
Ilustración 60.- Diagrama de secuencia para eliminar un punto de venta.



Fuente: Propia, 2023.

En la Ilustración 60 se muestra el diagrama de secuencia que detalla el proceso de eliminar un punto de venta. La secuencia se inicia con la solicitud de la lista de puntos de venta en la pantalla, la cual es validada posteriormente por el controlador correspondiente. Después de validar la solicitud, se accede a la base de datos para recuperar la lista de puntos de venta, presentándola al administrador. Luego, el administrador ingresa el identificador del punto de venta, y la pantalla gestiona la solicitud de validación para eliminar dicho punto de venta mediante el controlador. Este controlador verifica la existencia del punto de venta en la base de datos y, una vez superada la validación, procede a eliminar el punto de venta en la base de datos. Finalmente, se muestra al usuario un mensaje que refleja el resultado del proceso.

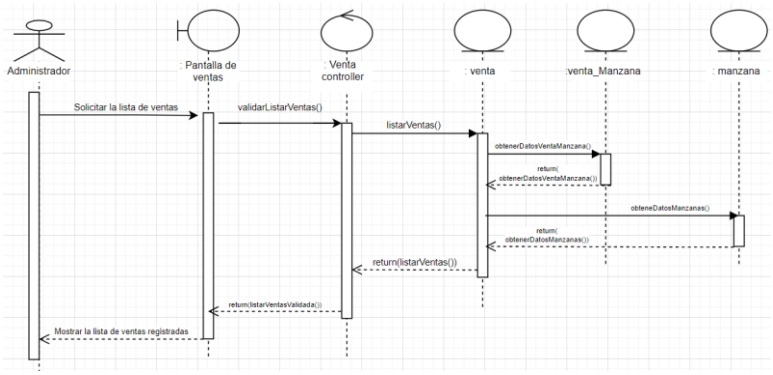
Ilustración 61.- Diagrama de secuencia para listar por id un punto de venta.



Fuente: Propia, 2023.

En la Ilustración 61 se expone el diagrama de secuencia dedicado a la visualización de un punto de venta específico. La secuencia se inicia al solicitar la lista de puntos de venta en la pantalla, la cual se somete a validación mediante el controlador correspondiente. Posterior a la validación de la solicitud, se obtiene la lista de puntos de venta desde la base de datos utilizando la tabla pertinente, y esta información se presenta al usuario. Seguidamente, el usuario proporciona el identificador (ID) del punto de venta. La pantalla solicita la validación de dicho punto de venta al controlador, encargado de verificar su existencia en la base de datos. Una vez validado, se requieren los datos asociados a ese punto de venta específico desde la base de datos, y esta información se muestra al usuario.

Ilustración 62.- Diagrama de secuencia para listar las ventas.

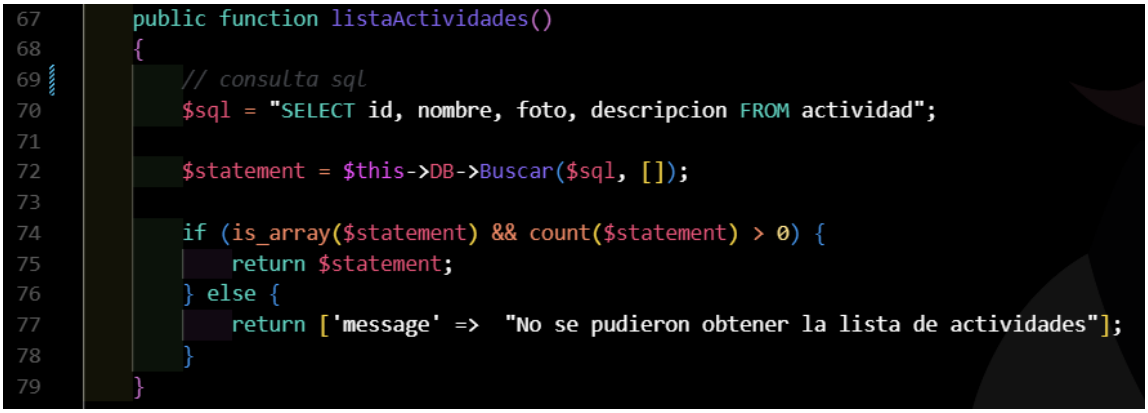


Fuente: Propia, 2023.

En la Ilustración 62 se muestra el diagrama de secuencia centrado en la presentación de información sobre ventas. En este proceso, el administrador busca obtener el listado de ventas en la pantalla asignada. La pantalla, a su vez, solicita los datos y las validaciones al controlador específico para ventas. Posteriormente, se realiza la consulta de la lista de ventas desde la tabla correspondiente en la base de datos, incluyendo los datos de la tabla intermedia entre la venta y el producto, así como los datos propios de la manzana, todos obtenidos de sus respectivas tablas. El procedimiento concluye con la presentación al administrador de la lista resultante.

Pantallas de código

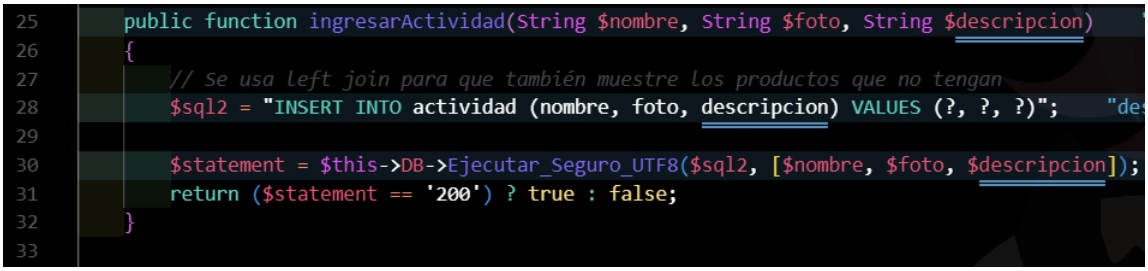
Ilustración 63.- Pantalla de código de la función para listar actividades.



Fuente: Propia, 2023.

En la Ilustración 63 se puede apreciar en código la función "listaActividades", la cual realiza una consulta SQL para seleccionar los campos id, nombre, foto y descripción de la tabla de actividades. Posteriormente, utiliza el método Buscar de la clase asociada a la base de datos para ejecutar la consulta. Si el resultado es un array con más de cero elementos, la función devuelve la lista de actividades. En caso contrario, retorna un array con un mensaje indicando que no se pudieron obtener las actividades.

Ilustración 64.- Pantalla de código de la función para ingresar una actividad.

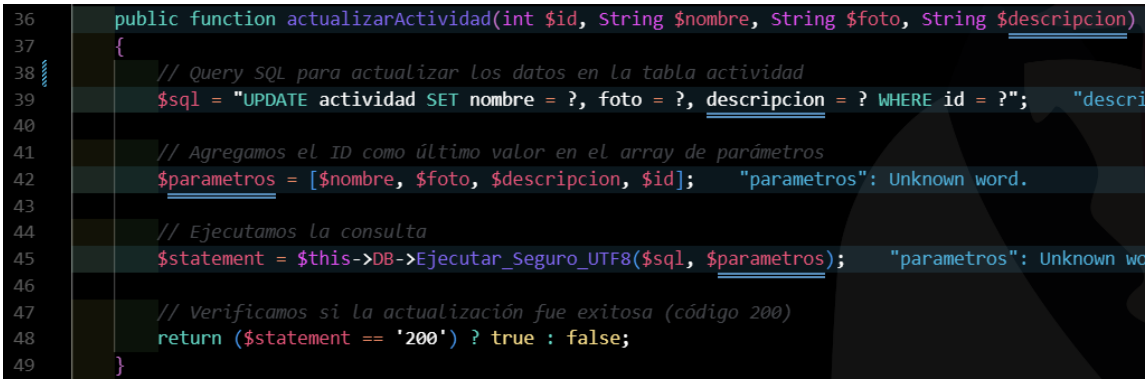


Fuente: Propia, 2023.

En la Ilustración 64 se muestra la función "ingresarActividad", la cual realiza la inserción de una nueva actividad en la base de datos. Utiliza una consulta SQL con un left join para asegurar que también se muestren los productos que no tienen coincidencias. Los parámetros de nombre, foto y descripción se pasan de manera

segura a la consulta mediante la conexión a la base de datos. La función devuelve un valor booleano indicando el éxito de la operación, siendo verdadero si la ejecución de la consulta retorna '200', y falso en caso contrario.

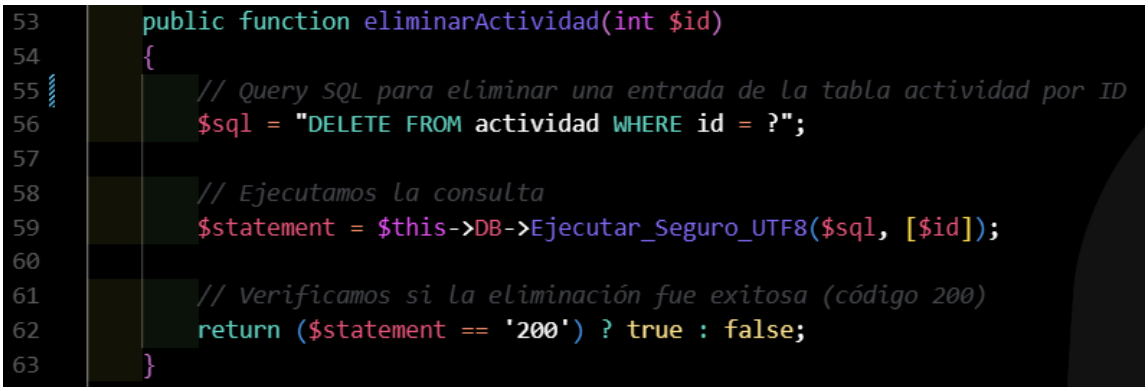
Ilustración 65.- Pantalla de código de la función para actualizar una actividad.



Fuente: Propia, 2023.

En la Ilustración 65 se encuentra el código de la función "actualizarActividad", dicha función tiene como propósito actualizar los datos de una actividad en la base de datos. Utiliza un comando SQL para ejecutar la actualización en la tabla de actividades, tomando como referencia el ID de la actividad a modificar. Se reciben como parámetros el ID de la actividad, el nuevo nombre, la nueva foto y la nueva descripción. Estos valores se incorporan a un array de parámetros, y la consulta SQL se ejecuta mediante el método "Ejecutar_Seguro_UTF8" proporcionado por la conexión a la base de datos. La función devuelve true si la actualización se realizó con éxito (código 200), de lo contrario, devuelve false.

Ilustración 66.- Pantalla de código de la función para eliminar una actividad.

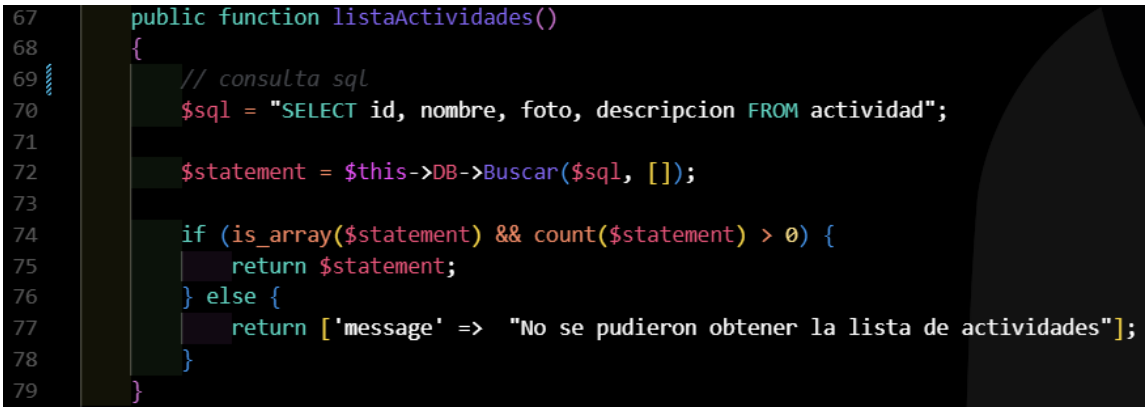


```php
public function eliminarActividad(int $id)
{
    // Query SQL para eliminar una entrada de la tabla actividad por ID
    $sql = "DELETE FROM actividad WHERE id = ?";

    // Ejecutamos la consulta
    $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, [$id]);

    // Verificamos si la eliminación fue exitosa (código 200)
    return ($statement == '200') ? true : false;
}
```

Fuente: Propia, 2023.

La función "eliminarActividad" se encuentra en la Ilustración 66, dicha función elimina una entrada en la tabla de actividades mediante un identificador proporcionado como parámetro. Utiliza una consulta SQL para eliminar la entrada correspondiente al ID especificado. La ejecución de la consulta se realiza a través de un objeto de conexión a la base de datos. La función devuelve true si la eliminación fue exitosa (código 200), y false en caso contrario.

Ilustración 67.- Pantalla de código de la función para listar actividades.



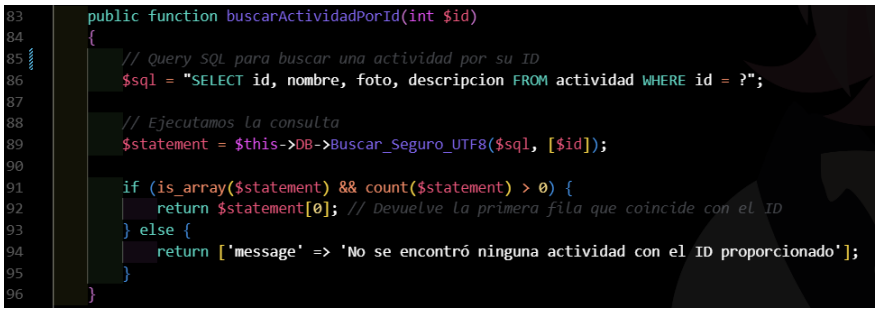
```php
public function listaActividades()
{
    // consulta sql
    $sql = "SELECT id, nombre, foto, descripcion FROM actividad";

    $statement = $this->DB->Buscar($sql, []);

    if (is_array($statement) && count($statement) > 0) {
        return $statement;
    } else {
        return ['message' =>  "No se pudieron obtener la lista de actividades"];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 67 denominada `listaActividades()`, realiza una consulta SQL para seleccionar la información de actividades, incluyendo campos como id, nombre, foto y descripción, desde la tabla de actividades en la base de datos. El método `Buscar` del objeto de base de datos (`$this->DB`) ejecuta la consulta y devuelve los resultados. Si se obtiene un array con resultados y su cantidad es mayor

a cero, la función devuelve dicho array con la información de las actividades. En caso contrario, devuelve un array con un mensaje indicando que no se pudo obtener la lista de actividades.

Ilustración 68.- Pantalla de código de la función para listar por identificador una actividad.

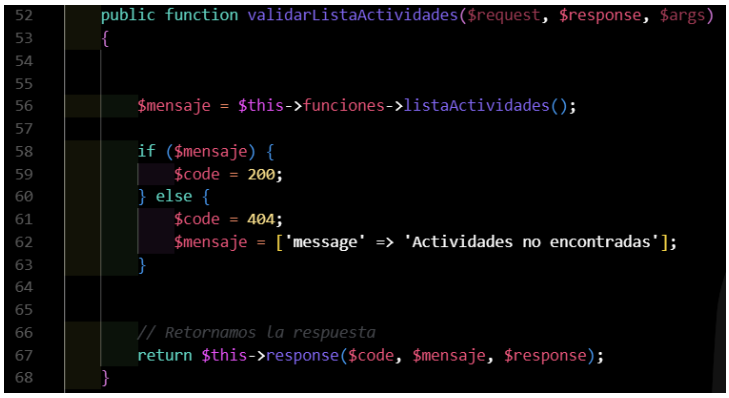


```php
public function buscarActividadPorId(int $id)
{
    // Query SQL para buscar una actividad por su ID
    $sql = "SELECT id, nombre, foto, descripcion FROM actividad WHERE id = ?";

    // Ejecutamos la consulta
    $statement = $this->DB->Buscar_Seguro_UTF8($sql, [$id]);

    if (is_array($statement) && count($statement) > 0) {
        return $statement[0]; // Devuelve la primera fila que coincide con el ID
    } else {
        return ['message' => 'No se encontró ninguna actividad con el ID proporcionado'];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 68 se muestra la función "buscarActividadPorId", dicha función toma como parámetro un ID y realiza una consulta SQL para buscar una actividad en la base de datos con ese ID. Se utiliza un método de búsqueda seguro UTF-8 proporcionado por la clase que maneja la conexión a la base de datos. Si la consulta devuelve al menos una fila, se devuelve la primera fila que coincide con el ID; de lo contrario, se devuelve un mensaje indicando que no se encontró ninguna actividad con el ID proporcionado.

Ilustración 69.- Pantalla de código del controlador para listar actividades.



```php
public function validarListaActividades($request, $response, $args)
{


    $mensaje = $this->funciones->listaActividades();

    if ($mensaje) {
        $code = 200;
    } else {
        $code = 404;
        $mensaje = ['message' => 'Actividades no encontradas'];
    }


    // Retornamos la respuesta
    return $this->response($code, $mensaje, $response);
}
```

Fuente: Propia, 2023.

La función "validarListaActividades" de la Ilustración 69 es parte de un código que válida la existencia de actividades. Utiliza el método "listaActividades" de la clase

"funciones" para obtener un mensaje. Si el mensaje se obtiene correctamente, el código de respuesta es 200, indicando éxito. En caso contrario, el código es 404, y se establece un mensaje de error indicando que no se encontraron actividades. La función luego retorna la respuesta, que incluye el código y el mensaje, utilizando el método "response" con los parámetros adecuados.

Ilustración 70.- Pantalla de código del controlador para ingresar una actividad.

```
21    public function validarIngresarActividad($request, $response, $args)
22    {
23        $params = (array)$request->getParsedBody();
24
25        $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
26        $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
27        $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';
28
29
30
31        $mensaje = ['message' => ''];
32
33        if ($nombre != '' && $descripcion != '' && $foto != '') {    "descripcion": Unknown word.
34            $mensaje = $this->funciones->ingresarActividad($nombre, $foto, $descripcion);    "desc
35
36            if ($mensaje) {
37                $code = 200;
38            } else {
39                $code = 404;
40            }
41        } else {
42            $code = 400;
43            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
44        }
45
```

Fuente: Propia, 2023.

La función de la ilustración 70 denominada "validarIngresarActividad" recibe una solicitud, extrae los parámetros del cuerpo de la solicitud y valida la presencia de tres campos: nombre, foto y descripción. Si estos campos están presentes y no son vacíos, se invoca la función "ingresarActividad" del objeto "$this->funciones" con los datos proporcionados. Dependiendo del resultado de esta operación, se establece un código de respuesta HTTP (200 si tiene éxito, 404 si falla). Si los campos no son correctos o están vacíos, se establece un código de respuesta 400 y se devuelve un mensaje indicando la situación. Finalmente, se utiliza el método `response` para retornar la respuesta al cliente.

Ilustración 71.- Pantalla de código del controlador para actualizar una actividad.

```php
public function validarActualizarActividad($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
    $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
    $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
    $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';



    $mensaje = ['message' => ''];

    if ($id > 0 && $nombre != '' && $foto != '' && $descripcion != '') {

        $mensaje = $this->funciones->actualizarActividad($id, $nombre, $foto, $descripcion);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 71 `validarActualizarActividad` recibe una solicitud, extrae parámetros relevantes como el ID, nombre, foto y descripción de la actividad a actualizar. Luego, verifica la validez de estos datos, asegurándose de que el ID sea mayor que cero y que los campos nombre, foto y descripción no estén vacíos. Si la validación es exitosa, invoca la función `actualizarActividad` a través del objeto `$this->funciones` y determina el código de respuesta en base al éxito de la actualización. Si la operación es exitosa, el código es 200; de lo contrario, es 404. En caso de que los datos sean incorrectos o estén vacíos, el código es 400 y se proporciona un mensaje indicando el problema. Finalmente, la función retorna una respuesta con el código y el mensaje correspondiente.

Ilustración 72.- Pantalla de código del controlador para eliminar una actividad.

```
106     public function validarEliminarActividad($request, $response, $args)
107     {
108         $params = (array)$request->getParsedBody();
109 
110         $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
111 
112 
113         $mensaje = ['message' => ''];
114 
115         if ($id > 0) {
116             $mensaje = $this->funciones->eliminarActividad($id);
117 
118             if ($mensaje) {
119                 $code = 200;
120             } else {
121                 $code = 404;
122             }
123         } else {
124             $code = 400;
125             $mensaje = ['message' => 'Datos incorrectos o vacíos'];
126         }
127 
128         // Retornamos la respuesta
129         return $this->response($code, [$mensaje], $response);
130     }
```

Fuente. Propia, 2023.

La función de la Ilustración 71 "validarEliminarActividad" recibe parámetros de solicitud, respuesta y argumentos. Extrae el identificador de actividad de la solicitud y lo valida. Si el identificador es mayor que cero, intenta eliminar la actividad utilizando una función llamada "eliminarActividad" y evalúa el resultado. Si la eliminación es exitosa, devuelve un código de respuesta 200; de lo contrario, devuelve un código 404. Si el identificador es incorrecto o nulo, establece un código de respuesta 400 y un mensaje indicando datos incorrectos o vacíos. Finalmente, la función retorna la respuesta al cliente.

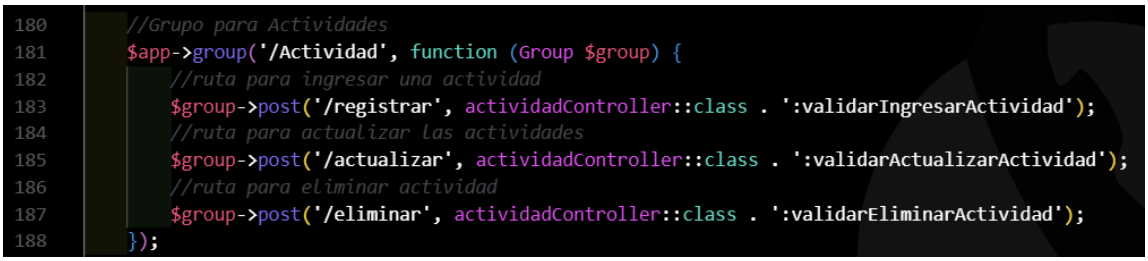
Ilustración 73.- Pantalla de código del controlador para buscar una actividad por id.

```
134     public function validarBuscarActividadId($request, $response, $args)
135     {
136         $params = (array)$request->getParsedBody();
137 
138         $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
139 
140 
141         $mensaje = ['message' => ''];
142 
143         if ($id > 0) {
144 
145             $mensaje = $this->funciones->buscarActividadPorId($id);
146 
147             if ($mensaje) {
148                 $code = 200;
149             } else {
150                 $code = 404;
151             }
152         } else {
153             $code = 400;
154             $mensaje = ['message' => 'Datos incorrectos o vacíos'];
155         }
156 
157         // Retornamos la respuesta
158         return $this->response($code, [$mensaje], $response);
159     }
```

Fuente: Propia, 2023.

La función de la Ilustración 73 "validarBuscarActividadId" recibe una solicitud, extrae los parámetros relevantes y busca una actividad por su identificador en base a esos parámetros. Si el identificador es mayor que cero, realiza la búsqueda utilizando un método proporcionado por la clase "funciones". Si la búsqueda es exitosa, devuelve un código de estado 200; de lo contrario, devuelve un código 404. Si el identificador es menor o igual a cero, establece un código de estado 400 y un mensaje indicando datos incorrectos o vacíos. Finalmente, la función retorna la respuesta adecuada.

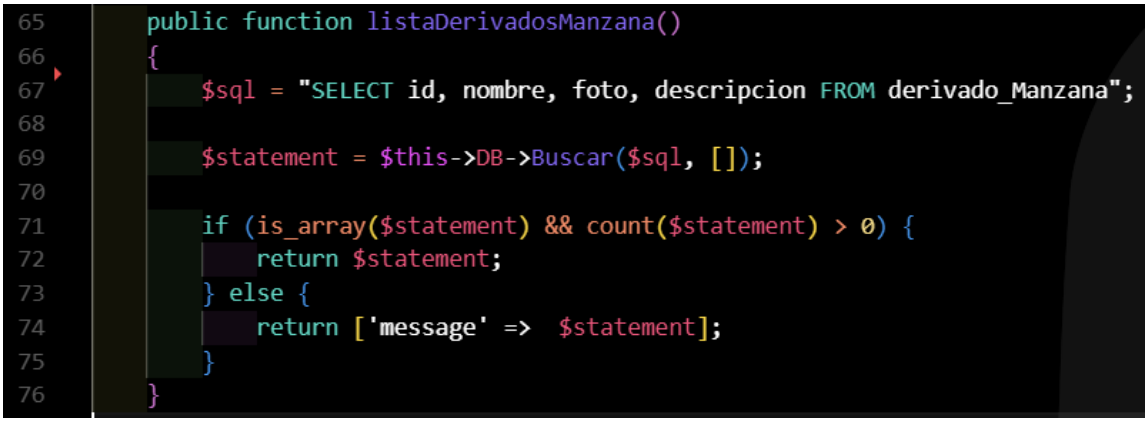
Ilustración 74.- Rutas de actividades.



Fuente: Propia, 2023.

En la Ilustración 74 se muestra el código del grupo de actividades para acceder a los métodos.

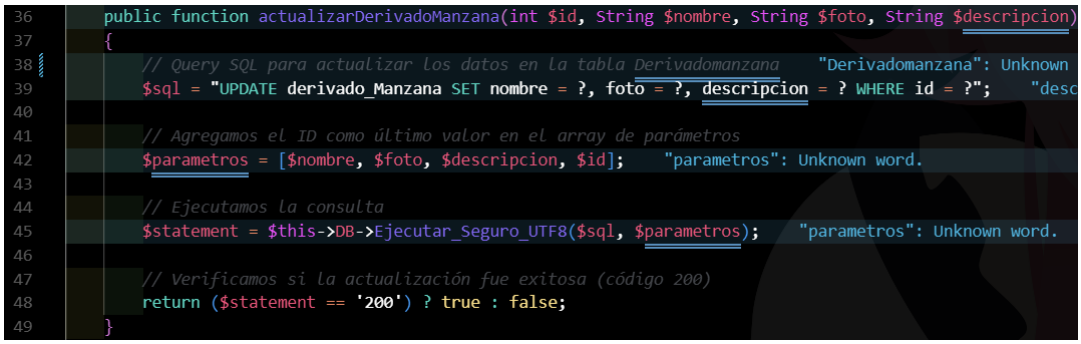
Ilustración 75.- Pantalla de código de la función para listar derivados de la manzana.



Fuente: Propia, 2023.

La función de la Ilustración 75 llamada "listaDerivadosManzana()", está diseñada para recuperar información específica de la base de datos relacionada con derivados de manzana. Utiliza una consulta SQL para seleccionar los campos de id, nombre, foto y descripción de la tabla "derivado_Manzana". Después, ejecuta esta consulta utilizando el método "Buscar()" proporcionado por la conexión a la base de datos. Si el resultado es un array con más de cero elementos, se devuelve ese array con los datos obtenidos. En caso contrario, se devuelve un array con un mensaje de error.

Ilustración 76.- Pantalla de código de la función para actualizar un derivado de manzana.

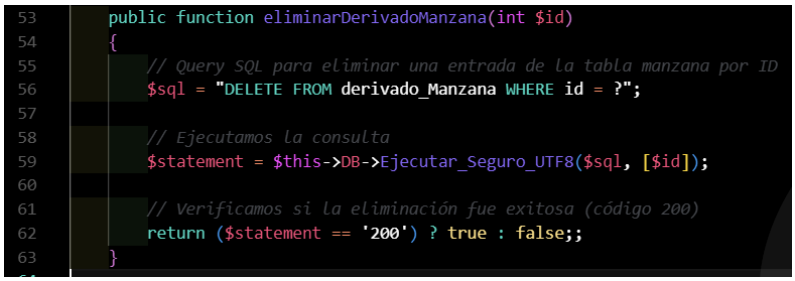


```
36    public function actualizarDerivadoManzana(int $id, String $nombre, String $foto, String $descripcion)
37    {
38        // Query SQL para actualizar los datos en la tabla Derivadomanzana    "Derivadomanzana": Unknown
39        $sql = "UPDATE derivado_Manzana SET nombre = ?, foto = ?, descripcion = ? WHERE id = ?";    "desc
40
41        // Agregamos el ID como último valor en el array de parámetros
42        $parametros = [$nombre, $foto, $descripcion, $id];    "parametros": Unknown word.
43
44        // Ejecutamos la consulta
45        $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, $parametros);    "parametros": Unknown word.
46
47        // Verificamos si la actualización fue exitosa (código 200)
48        return ($statement == '200') ? true : false;
49    }
```

Fuente: Propia, 2023.

La función de la Ilustración 76 `actualizarDerivadoManzana` se encarga de actualizar los datos de un derivado de manzana en la base de datos. Utiliza una consulta SQL para actualizar las columnas de nombre, foto y descripción en la tabla `derivado_Manzana` según el ID proporcionado. El ID se agrega como último valor en el array de parámetros, y la consulta se ejecuta mediante un método proporcionado por la clase que representa la conexión a la base de datos. La función devuelve verdadero si la actualización fue exitosa (código 200), de lo contrario, devuelve falso.

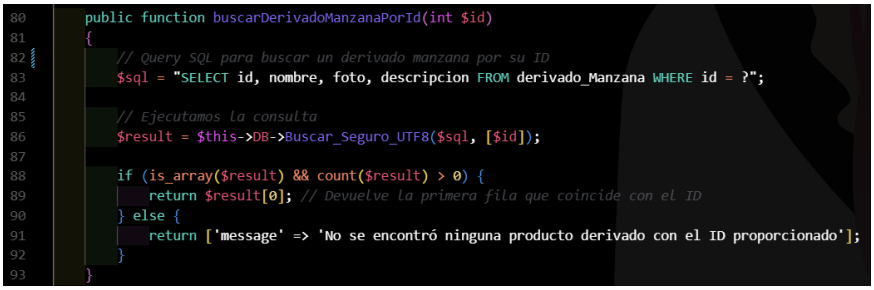
Ilustración 77.- Pantalla de código de la función para eliminar un derivado de manzana.



Fuente: Propia, 2023.

La función de la Ilustración 77 "eliminarDerivadoManzana" en el código proporcionado realiza la eliminación de una entrada específica en la tabla "derivado_Manzana" mediante una consulta SQL DELETE, utilizando el identificador proporcionado ($id). La consulta se ejecuta utilizando un objeto de base de datos ($this->DB) y se verifica si la operación de eliminación fue exitosa mediante la comprobación del código de estado devuelto por la ejecución de la consulta. Si el código es igual a '200', la función devuelve verdadero; de lo contrario, devuelve falso.

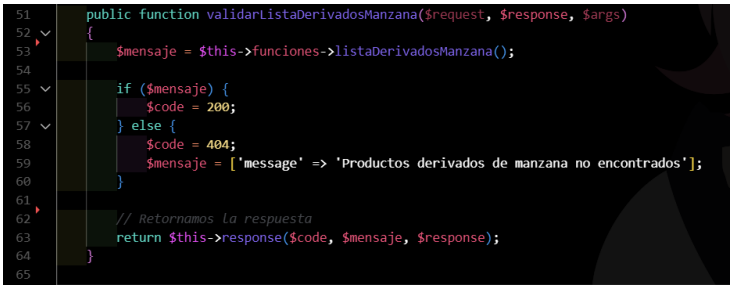
Ilustración 78.- Pantalla de código de la función para listar por id un derivado manzana.



Fuente: Propia, 2023.

La función de la Ilustración 78 `buscarDerivadoManzanaPorId` realiza una consulta SQL para buscar un producto derivado de manzana según su ID en una tabla específica. La consulta se ejecuta utilizando un método proporcionado por la clase asociada a la base de datos. Si se encuentra al menos una fila que coincide con el ID, se devuelve esa fila como un arreglo asociativo con los campos id, nombre, foto y descripción. En caso contrario, se devuelve un arreglo con un mensaje indicando que no se encontró ningún producto derivado con el ID proporcionado.

Ilustración 79.- Pantalla de código del controlador para listar derivados de manzana.



Fuente: Propia, 2023.

La función de la Ilustración 79 `validarListaDerivadosManzana` recibe una solicitud, una respuesta y argumentos, y utiliza el método `listaDerivadosManzana` de un objeto `funciones` para obtener un mensaje. Dependiendo de la existencia de este mensaje, se asigna un código de respuesta HTTP 200 si hay datos disponibles, o 404 si no se encuentran productos derivados de manzana, acompañado de un mensaje explicativo. Finalmente, la función retorna una respuesta HTTP junto con el código y el mensaje correspondiente.

Ilustración 80.- Pantalla de código del controlador para ingresar un derivado de manzana.

Fuente: Propia, 2023.

La función de la Ilustración 80 "validarIngresarDerivadoManzana" recibe una solicitud HTTP, extrae los parámetros del cuerpo de la solicitud y valida la existencia de 'nombre', 'foto' y 'descripcion'. Si estos datos están presentes, se llama a la función "ingresarDerivadoManzana" con esos parámetros y se verifica si se ejecutó

correctamente. Dependiendo del resultado, se establece un código de respuesta HTTP 200 en caso de éxito, 404 si hay un problema, y 400 si los datos son incorrectos o están vacíos. El mensaje de respuesta se estructura de acuerdo con el resultado y se devuelve como parte de la respuesta HTTP.

Ilustración 81.- Pantalla de código del controlador para actualizar un derivado de manzana.

```
69    public function validarActualizarDeribadoManzana($request, $response, $args)    "Deribado": Unknown
70    {
71        $params = (array)$request->getParsedBody();
72
73        $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
74        $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
75        $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
76        $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';
77
78        if ($id > 0 && $nombre != '' && $foto != '' && $descripcion != '') {
79
80            $mensaje = $this->funciones->actualizarDerivadoManzana($id, $nombre, $foto, $descripcion);
81
82            if ($mensaje) {
83                $code = 200;
84            } else {
85                $code = 404;
86            }
87        } else {
88            $code = 400;
89            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
90        }
91
92        // Retornamos la respuesta
93        return $this->response($code, [$mensaje], $response);
94    }
```

Fuente: Propia, 2023.

La función de la Ilustración 81 `validarActualizarDeribadoManzana` recibe una solicitud HTTP con parámetros, los cuales se obtienen y validan. Se extraen el identificador, nombre, foto y descripción, asegurándose de que el identificador sea mayor a cero y que los campos nombre, foto y descripción no estén vacíos. Si la validación es exitosa, se invoca la función `actualizarDerivadoManzana` para actualizar la información de una manzana en la base de datos. Dependiendo del resultado de la actualización, se establece un código de respuesta HTTP 200 si es exitosa, o 404 si falla. Si la validación inicial falla, se retorna un código 400 junto con un mensaje indicando que los datos son incorrectos o vacíos. La función finalmente devuelve la respuesta HTTP al cliente.

Ilustración 82.- Pantalla de código del controlador para eliminar un derivado de manzana.

```php
public function validarEliminarDerivadoManzana($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;


    $mensaje = ['message' => ''];

    if ($id > 0) {


        $mensaje = $this->funciones->eliminarDerivadoManzana($id);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 81 `validarEliminarDerivadoManzana` recibe una solicitud, extrae los parámetros relevantes, y valida la presencia y corrección del identificador proporcionado. Si el identificador es válido, se procede a eliminar el derivado de la manzana a través de la función `eliminarDerivadoManzana`. El resultado de la operación determina el código de respuesta: 200 si la operación fue exitosa, 404 si no se encontró el elemento a eliminar y 400 si los datos son incorrectos o están vacíos. La función devuelve la respuesta correspondiente utilizando el método `response`.

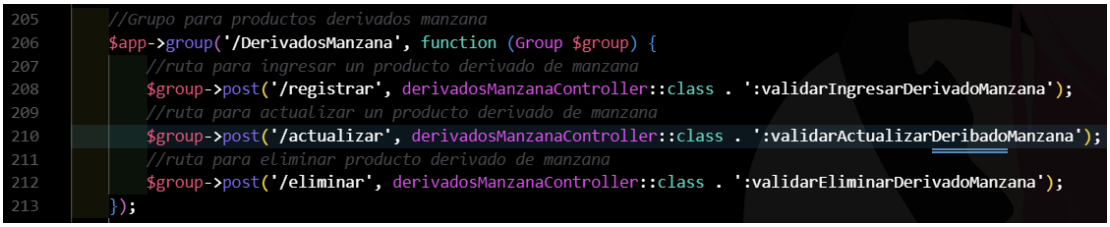
Ilustración 83.- Pantalla de código del controlador para listar un derivado manzana por id.

```php
public function validarBuscarDerivadoManzanaPorId($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;


    $mensaje = ['message' => ''];

    if ($id > 0) {

        $mensaje = $this->funciones->buscarDerivadoManzanaPorId($id);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 83 `validarBuscarDerivadoManzanaPorId` procesa una solicitud HTTP para buscar un derivado de una manzana por su identificador. Valida los parámetros de la solicitud y devuelve una respuesta HTTP con un código y un mensaje según el resultado de la búsqueda. Si el identificador es válido, se realiza la búsqueda y se establece el código 200 si se encuentra el derivado, o 404 si no. En caso de datos incorrectos o vacíos, se establece el código 400.

Ilustración 84.- Rutas de derivados de manzana.



Fuente: Propia, 2023.

En el código de la Ilustración 84 se crea un grupo para manejar operaciones relacionadas con productos derivados de manzana. Se definen tres rutas para registrar, actualizar y eliminar estos productos, cada una vinculada a un método específico del controlador llamado `derivadosManzanaController`.

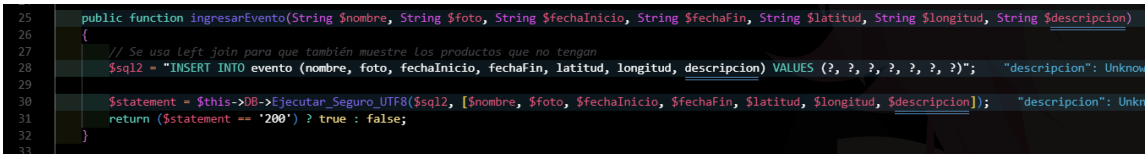
Ilustración 85.- Pantalla de código de la función para listar eventos.

Fuente: Propia, 2023.

La función de la Ilustración 85 llamada `listaEventos`, realiza una consulta a la base de datos para seleccionar las columnas 'id', 'nombre', 'foto', 'fechaInicio', 'fechaFin', 'latitud', 'longitud', y 'descripcion' de la tabla 'evento'. Utiliza un método llamado

`Buscar` de la clase que contiene esta función, pasando la consulta SQL y un array vacío de parámetros. Luego, verifica si el resultado de la consulta es un array y si su longitud es mayor a cero. Si es así, devuelve el array con los datos de los eventos; de lo contrario, devuelve un array con un mensaje de error.

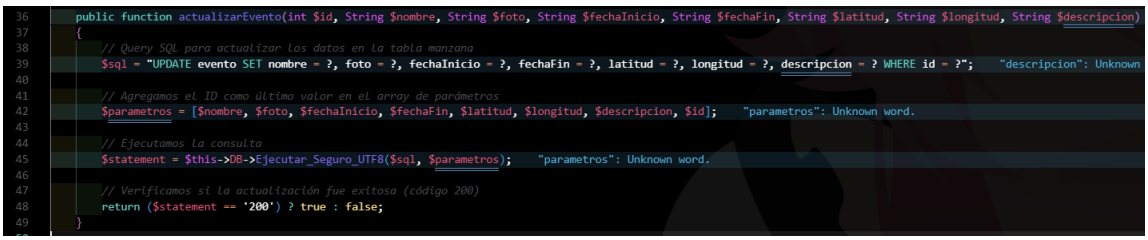
Ilustración 86.- Pantalla de código de la función para ingresar un evento.



Fuente: Propia, 2023.

La función de la Ilustración 86 "ingresarEvento" realiza la inserción de un nuevo evento en la base de datos. Los parámetros de entrada incluyen el nombre, la foto, las fechas de inicio y fin, la latitud, la longitud y la descripción del evento. La ejecución segura de la consulta se lleva a cabo a través del objeto de conexión a la base de datos, y el resultado indica el éxito o fracaso de la operación, retornando `true` en caso de éxito y `false` en caso contrario.

Ilustración 87.- Pantalla de código de la función para actualizar un evento.

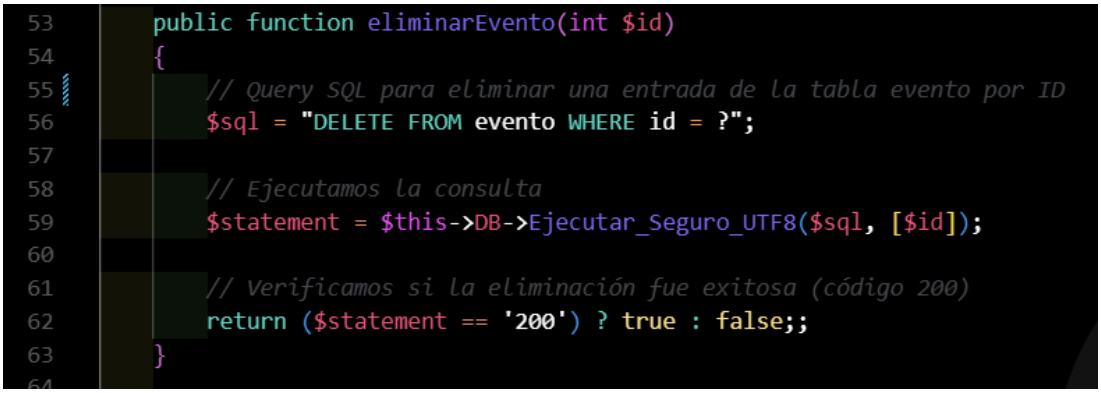


Fuente: Propia, 2023.

La función de la Ilustración 87 `actualizarEvento` se encarga de modificar los datos de un evento en la base de datos. Utiliza una consulta SQL de actualización, especificando los nuevos valores de nombre, foto, fechas de inicio y fin, latitud, longitud y descripción, asociados al ID del evento proporcionado. Los parámetros se agregan a un array que incluye el ID como último elemento. La consulta se ejecuta mediante un método proporcionado por la clase que maneja la conexión a la base

de datos. La función devuelve `true` si la actualización fue exitosa (código 200), y `false` en caso contrario.

Ilustración 88.- Pantalla de código de la función para eliminar un evento.

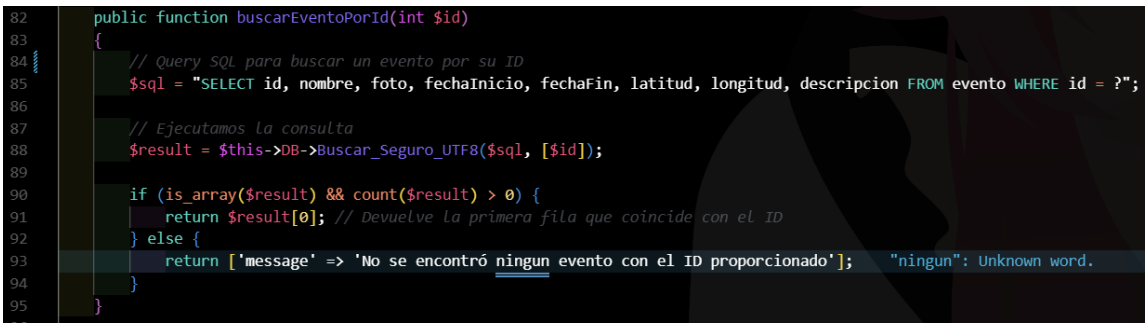


```
53      public function eliminarEvento(int $id)
54      {
55          // Query SQL para eliminar una entrada de la tabla evento por ID
56          $sql = "DELETE FROM evento WHERE id = ?";
57
58          // Ejecutamos la consulta
59          $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, [$id]);
60
61          // Verificamos si la eliminación fue exitosa (código 200)
62          return ($statement == '200') ? true : false;;
63      }
```

Fuente: Propia, 2023.

La función de la Ilustración 88 "eliminarEvento" tiene como objetivo eliminar una entrada en la tabla de eventos mediante un ID proporcionado. Utiliza una consulta SQL de eliminación y ejecuta dicha consulta a través de la conexión a la base de datos. La función devuelve true si la eliminación fue exitosa, identificada por el código 200, y false en caso contrario.

Ilustración 89.- Pantalla de código de la función para buscar un evento por su id.



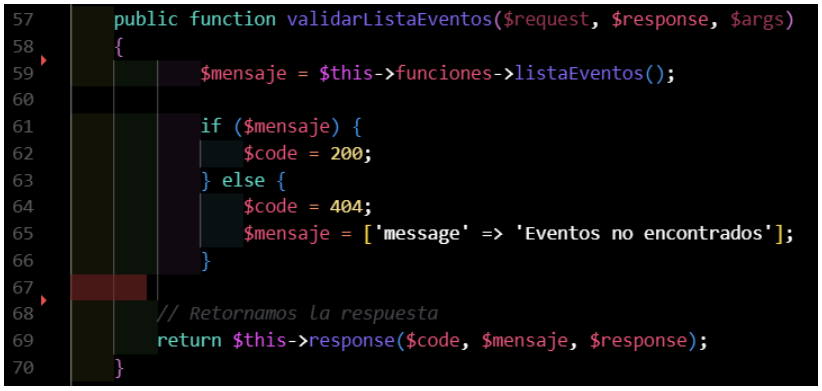
```
82      public function buscarEventoPorId(int $id)
83      {
84          // Query SQL para buscar un evento por su ID
85          $sql = "SELECT id, nombre, foto, fechaInicio, fechaFin, latitud, longitud, descripcion FROM evento WHERE id = ?";
86
87          // Ejecutamos la consulta
88          $result = $this->DB->Buscar_Seguro_UTF8($sql, [$id]);
89
90          if (is_array($result) && count($result) > 0) {
91              return $result[0]; // Devuelve la primera fila que coincide con el ID
92          } else {
93              return ['message' => 'No se encontró ningun evento con el ID proporcionado'];    "ningun": Unknown word.
94          }
95      }
```

Fuente: Propia, 2023.

La función de la Ilustración 89 busca un evento en una base de datos mediante su ID. Utiliza una consulta SQL para seleccionar información específica del evento, como su nombre, foto, fechas de inicio y fin, ubicación geográfica, y descripción. La consulta se ejecuta a través de un método seguro proporcionado por una conexión a la base de datos. Si la consulta devuelve al menos una fila, la función devuelve los

detalles del primer evento encontrado. En caso contrario, devuelve un mensaje indicando que no se encontró ningún evento con el ID proporcionado.

Ilustración 90.- Pantalla de código del controlador para listar eventos.

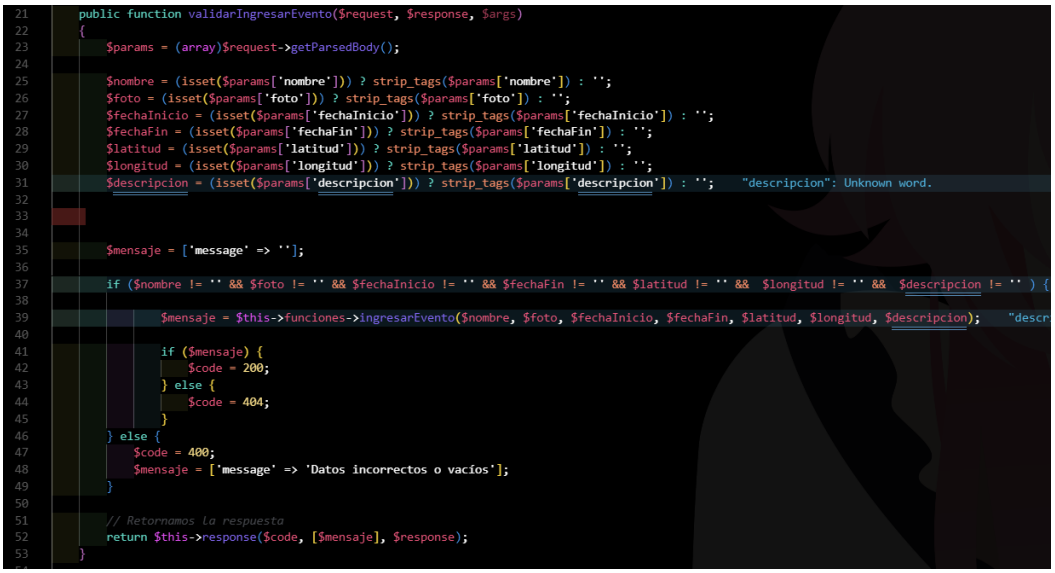


```
57      public function validarListaEventos($request, $response, $args)
58      {
59              $mensaje = $this->funciones->listaEventos();
60
61              if ($mensaje) {
62                  $code = 200;
63              } else {
64                  $code = 404;
65                  $mensaje = ['message' => 'Eventos no encontrados'];
66              }
67
68          // Retornamos la respuesta
69          return $this->response($code, $mensaje, $response);
70      }
```

Fuente: Propia, 2023.

La función de la Ilustración 90 `validarListaEventos` recibe una solicitud, una respuesta y argumentos. En su ejecución, obtiene un mensaje mediante la función `listaEventos`. Si el mensaje existe, establece el código de respuesta como 200; de lo contrario, lo establece como 404 y define un mensaje indicando que no se encontraron eventos. Finalmente, retorna la respuesta con el código y el mensaje obtenidos.

Ilustración 91.- Pantalla de código del controlador para ingresar un evento.

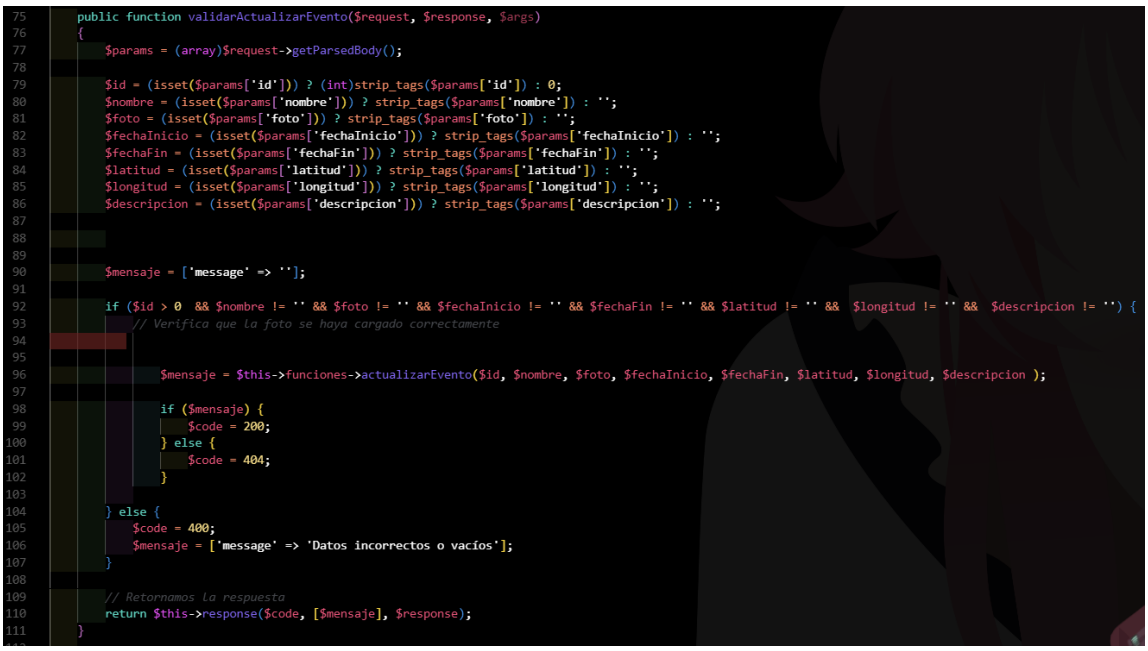


```
21      public function validarIngresarEvento($request, $response, $args)
22      {
23          $params = (array)$request->getParsedBody();
24
25          $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
26          $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
27          $fechaInicio = (isset($params['fechaInicio'])) ? strip_tags($params['fechaInicio']) : '';
28          $fechaFin = (isset($params['fechaFin'])) ? strip_tags($params['fechaFin']) : '';
29          $latitud = (isset($params['latitud'])) ? strip_tags($params['latitud']) : '';
30          $longitud = (isset($params['longitud'])) ? strip_tags($params['longitud']) : '';
31          $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';    "descripcion": Unknown word.
32
33
34
35          $mensaje = ['message' => ''];
36
37          if ($nombre != '' && $foto != '' && $fechaInicio != '' && $fechaFin != '' && $latitud != '' &&  $longitud != '' &&  $descripcion != '' ) {
38
39              $mensaje = $this->funciones->ingresarEvento($nombre, $foto, $fechaInicio, $fechaFin, $latitud, $longitud, $descripcion);    "descr
40
41              if ($mensaje) {
42                  $code = 200;
43              } else {
44                  $code = 404;
45              }
46          } else {
47              $code = 400;
48              $mensaje = ['message' => 'Datos incorrectos o vacíos'];
49          }
50
51          // Retornamos la respuesta
52          return $this->response($code, [$mensaje], $response);
53      }
```

Fuente: Propia, 2023.

La función de la Ilustración 91 `validarIngresarEvento` recibe una solicitud, extrae parámetros del cuerpo de la solicitud y válida la presencia de datos esenciales como nombre, foto, fechas, coordenadas y descripción. Si todos los datos requeridos están presentes, se utiliza el método `ingresarEvento` de la clase asociada para intentar ingresar un nuevo evento en la base de datos. Dependiendo del resultado, se establece un código de respuesta HTTP (200 si se ingresó correctamente, 404 si hubo un error, 400 si faltan datos o son incorrectos). La respuesta incluye un mensaje que indica el estado del proceso.

Ilustración 92.- Pantalla de código del controlador para actualizar un evento.



```php
public function validarActualizarEvento($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
    $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
    $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
    $fechaInicio = (isset($params['fechaInicio'])) ? strip_tags($params['fechaInicio']) : '';
    $fechaFin = (isset($params['fechaFin'])) ? strip_tags($params['fechaFin']) : '';
    $latitud = (isset($params['latitud'])) ? strip_tags($params['latitud']) : '';
    $longitud = (isset($params['longitud'])) ? strip_tags($params['longitud']) : '';
    $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';



    $mensaje = ['message' => ''];

    if ($id > 0  && $nombre != '' && $foto != '' && $fechaInicio != '' && $fechaFin != '' && $latitud != '' &&  $longitud != '' &&  $descripcion != '') {
        // Verifica que la foto se haya cargado correctamente


        $mensaje = $this->funciones->actualizarEvento($id, $nombre, $foto, $fechaInicio, $fechaFin, $latitud, $longitud, $descripcion );

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 92 `validarActualizarEvento` recibe parámetros de solicitud y argumentos, los procesa y válida para asegurarse de que contengan la información necesaria para actualizar un evento. Extrae y verifica datos como el ID, nombre, foto, fechas, coordenadas y descripción del evento. Si todos los datos son válidos, invoca un método para realizar la actualización y retorna un código de respuesta 200 si la operación fue exitosa, o 404 si hubo un problema. En caso de datos incorrectos o faltantes, devuelve un código 400 junto con un mensaje

indicando el problema. La respuesta final se construye mediante un método `response` que incluye el código y el mensaje correspondiente.

Ilustración 93.- Pantalla de código del controlador para eliminar un evento.

```php
public function validarEliminarEvento($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;


    $mensaje = ['message' => ''];

    if ($id > 0) {


        $mensaje = $this->funciones->eliminarEvento($id);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 93 `validarEliminarEvento` recibe una solicitud, una respuesta y argumentos. Primero, obtiene los parámetros del cuerpo de la solicitud y extrae el valor del parámetro 'id'. Luego, inicializa un mensaje vacío. Si el 'id' es mayor que cero, intenta eliminar el evento utilizando un método llamado `eliminarEvento` y asigna el resultado al mensaje. Dependiendo del resultado, establece un código de respuesta 200 si se elimina correctamente o 404 si no se encuentra el evento. Si el 'id' es igual o menor que cero, establece un código de respuesta 400 y actualiza el mensaje con la indicación de datos incorrectos o vacíos. Finalmente, la función retorna la respuesta con el código y el mensaje correspondientes.

Ilustración 94.- Pantalla de código del controlador para buscar por id un evento.

```php
public function validarBuscarEventoId($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;


    $mensaje = ['message' => ''];

    if ($id > 0) {


        $mensaje = $this->funciones->buscarEventoPorId($id);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 94 `validarBuscarEventoId` toma una solicitud, extrae los parámetros del cuerpo, específicamente el identificador (`id`) del evento, y realiza validaciones. Si el identificador es mayor que cero, se invoca la función `buscarEventoPorId` a través de la instancia de la clase `$funciones`. Dependiendo del resultado de esta búsqueda, se establece un código de respuesta HTTP 200 si se encuentra el evento, o 404 si no se encuentra. En caso de que el identificador sea incorrecto o esté ausente, se establece un código de respuesta 400 y se asigna un mensaje indicando datos incorrectos o vacíos. Finalmente, se devuelve la respuesta utilizando el método `response`.

Ilustración 95.- Rutas de eventos.

```php
//Grupo para Eventos
$app->group('/Eventos', function (Group $group) {
    //ruta para ingresar un evento
    $group->post('/registrar', eventoController::class . ':validarIngresarEvento');
    //ruta para actualizar un evento
    $group->post('/actualizar', eventoController::class . ':validarActualizarEvento');
    //ruta para eliminar un evento
    $group->post('/eliminar', eventoController::class . ':validarEliminarEvento');
});
```

Fuente: Propia, 2023.

El código de la Ilustración 95 utiliza el framework Slim para definir un grupo de rutas relacionadas con eventos. Dentro de este grupo, se han establecido tres rutas distintas: una para registrar un evento, otra para actualizar un evento y la tercera para eliminar un evento. Cada ruta está asociada a un controlador de eventos específico que se encargará de validar y procesar las solicitudes correspondientes.

Ilustración 96.- Pantalla de código de la función para verificar usuarios.

```php
public function verificarUsuario(string $usuario, string $contrasenia)
    {
        $sql = "SELECT id, idRol FROM usuario WHERE usuario =? AND contrasenia = ?";
        $statement = $this->DB->Buscar_Seguro($sql, [$usuario, $contrasenia]);
        if (is_array($statement) && count($statement) > 0) {
            return $statement[0];
        }
    }
```

Fuente: Propia, 2023

La función de la Ilustración 96 "verificarUsuario" realiza una consulta segura a la base de datos para verificar la existencia de un usuario con las credenciales proporcionadas. Utiliza una consulta SQL para seleccionar el ID y el rol del usuario cuyo nombre de usuario y contraseña coinciden con los parámetros dados. Se utiliza un método de búsqueda seguro proporcionado por la clase que maneja la conexión a la base de datos. Si la consulta arroja resultados y devuelve un array con al menos un elemento, la función retorna la información del primer resultado, que incluye el ID y el rol del usuario verificado.

Ilustración 97.- Pantalla de código de la función para crear un productor.

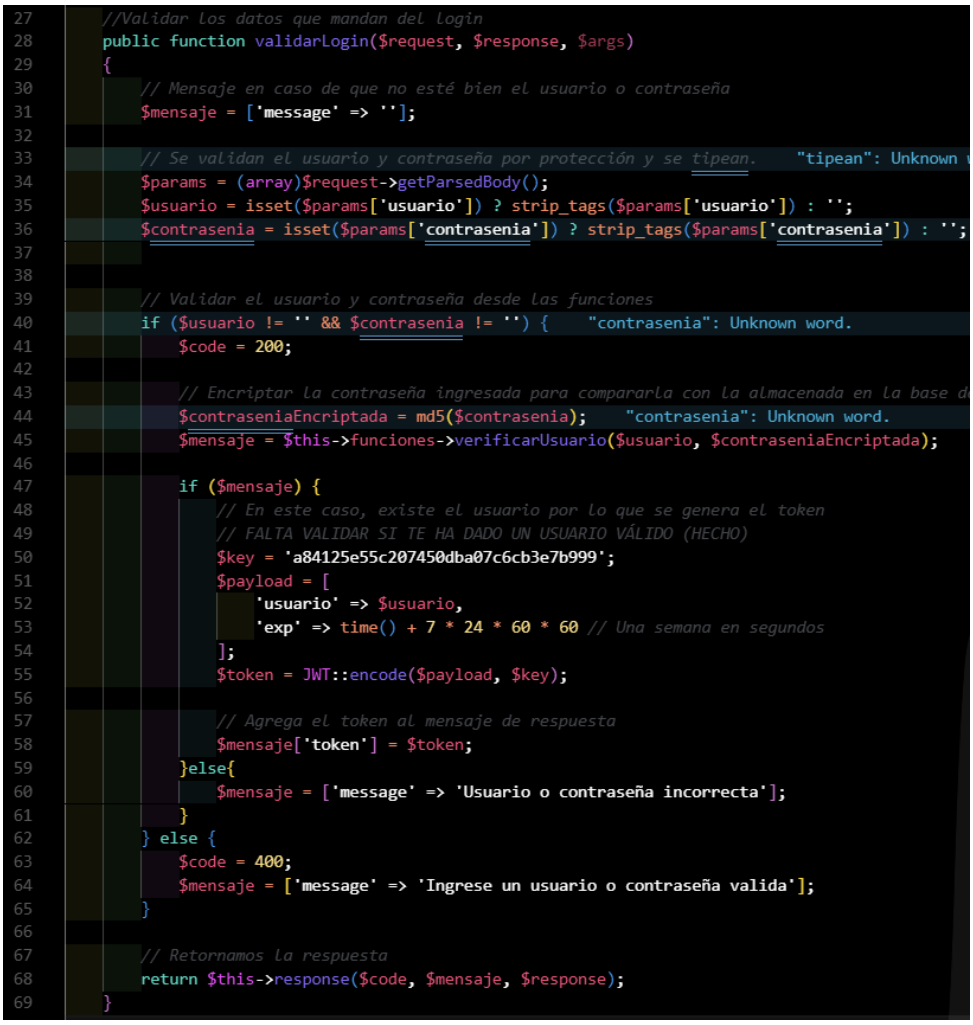
```php
public function crearProductor(string $nombre, string $apellidoPat, string $apellidoMat, string $correo, string $telefono, string $usuario, string $contraseniaEncriptada)
{

    // Verificar si el correo o el usuario ya existen en la base de datos
    $sqlVerificarExistencia = "SELECT COUNT(*) AS count FROM usuario WHERE correo = ? OR usuario = ?";
    $existencia = $this->DB->Buscar_Seguro_UTF8($sqlVerificarExistencia, [$correo, $usuario]);

    if (!empty($existencia) && $existencia[0]['count'] > 0) {
        return ['error' => "El correo o el usuario ya están en uso."];
    }

    $sql = "INSERT INTO usuario (nombre, apellidoPat, apellidoMat, correo, telefono, usuario, contrasenia, idRol) VALUES (?, ?, ?, ?, ?, ?, ?, ?)";    "telefono": Unkn
    $save = $this->DB->Ejecutar_Seguro_UTF8($sql, [$nombre, $apellidoPat, $apellidoMat, $correo, $telefono, $usuario, $contraseniaEncriptada, 2]);

    if ($save == '200') {
        return ['message' => "Usuario creado con éxito"];
    } else {
        return ['error' => "No se pudo crear el usuario."];
    }

}
```

Fuente: Propia, 2023.

La función de la ilustración 97 "crearProductor" se encarga de agregar un nuevo productor a la base de datos. Primero, verifica si el correo o el nombre de usuario

ya existen en la base de datos. Si encuentra coincidencias, devuelve un mensaje de error indicando que el correo o el usuario ya están en uso. En caso contrario, procede a insertar los datos del nuevo productor en la tabla de usuarios, asignándole un rol específico. Si la inserción se realiza con éxito, retorna un mensaje indicando que el usuario se ha creado correctamente; de lo contrario, devuelve un mensaje de error informando que no se pudo crear el usuario.

Ilustración 98.- Pantalla de código del controlador para validar el inicio de sesión.



```php
//Validar los datos que mandan del login
public function validarLogin($request, $response, $args)
{
    // Mensaje en caso de que no esté bien el usuario o contraseña
    $mensaje = ['message' => ''];

    // Se validan el usuario y contraseña por protección y se tipean.
    $params = (array)$request->getParsedBody();
    $usuario = isset($params['usuario']) ? strip_tags($params['usuario']) : '';
    $contrasenia = isset($params['contrasenia']) ? strip_tags($params['contrasenia']) : '';


    // Validar el usuario y contraseña desde las funciones
    if ($usuario != '' && $contrasenia != '') {
        $code = 200;

        // Encriptar la contraseña ingresada para compararla con la almacenada en la base d
        $contraseniaEncriptada = md5($contrasenia);
        $mensaje = $this->funciones->verificarUsuario($usuario, $contraseniaEncriptada);

        if ($mensaje) {
            // En este caso, existe el usuario por lo que se genera el token
            // FALTA VALIDAR SI TE HA DADO UN USUARIO VÁLIDO (HECHO)
            $key = 'a84125e55c207450dba07c6cb3e7b999';
            $payload = [
                'usuario' => $usuario,
                'exp' => time() + 7 * 24 * 60 * 60 // Una semana en segundos
            ];
            $token = JWT::encode($payload, $key);

            // Agrega el token al mensaje de respuesta
            $mensaje['token'] = $token;
        }else{
            $mensaje = ['message' => 'Usuario o contraseña incorrecta'];
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Ingrese un usuario o contraseña valida'];
    }

    // Retornamos la respuesta
    return $this->response($code, $mensaje, $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 98 `validarLogin` tiene como objetivo verificar la autenticación de un usuario. En primer lugar, se obtienen y validan los datos del usuario y contraseña provenientes de la solicitud. Luego, se encripta la contraseña

ingresada para compararla con la almacenada en la base de datos. Si la verificación es exitosa, se genera un token JWT para el usuario autenticado, el cual se agrega a la respuesta. En caso de credenciales incorrectas, se devuelve un mensaje correspondiente. Además, se manejan situaciones en las que no se proporcionan datos o se ingresan de manera incorrecta. Finalmente, la función retorna una respuesta HTTP junto con el código correspondiente y el mensaje generado.

Ilustración 99.- Pantalla de código del controlador para validar la creación de un productor.

```php
public function validarCrearProductor($request, $response, $args)
{

    //se validan los datos para crear un usuario.
    $params    = (array)$request->getParsedBody();
    $nombre    = (isset($params['nombre']))    ? strip_tags($params['nombre'])    : '';
    $apellidoPat    = (isset($params['apellidoPat']))    ? strip_tags($params['apellidoPat'])    : '';
    $apellidoMat  = (isset($params['apellidoMat']))    ? strip_tags($params['apellidoMat'])    : '';
    $correo    = (isset($params['correo']))    ? strip_tags($params['correo'])    : '';
    $telefono    = (isset($params['telefono']))    ? strip_tags($params['telefono'])    : '';
    $usuario    = (isset($params['usuario']))    ? strip_tags($params['usuario'])    : '';
    $contrasenia  = (isset($params['contrasenia']))    ? strip_tags($params['contrasenia'])    : '';

    $contraseniaEncriptada = md5($contrasenia);

    if ($telefono != '' && $correo != '' && $apellidoMat != '' && $apellidoPat != '' && $nombre != '' && $usuario != '' && $correo != ''
    && $contraseniaEncriptada != '') {

        $respuesta = $this->funciones->crearProductor($nombre, $apellidoPat, $apellidoMat, $correo, $telefono, $usuario, $contraseniaEncriptada);

        if ($respuesta) {
            $code = 200;
        } else {
            $code = 400;
        }
    } else {
        $respuesta =  ['message' => 'Valores vacíos'];
        $code = 400;
    }


    return $this->response($code, [$respuesta], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 99 `validarCrearProductor` se encarga de validar los datos recibidos a través de una solicitud y crear un nuevo productor en el sistema. Los parámetros son obtenidos del cuerpo de la solicitud y luego se filtran y asignan a variables correspondientes, como nombre, apellidos, correo, teléfono, usuario y contraseña. Se realiza la encriptación de la contraseña mediante el algoritmo MD5. Posteriormente, se verifica la existencia de valores no vacíos para los campos esenciales como nombre, apellidos, correo, teléfono, usuario y contraseña encriptada. Si todos los campos esenciales tienen valores no vacíos, se procede a llamar a la función `crearProductor` para realizar la creación del productor con los datos proporcionados. Si la creación es exitosa, se establece el código de respuesta HTTP como 200; de lo contrario, se establece como 400. En caso de que existan

valores vacíos en los campos esenciales, se genera un mensaje de error y se establece el código de respuesta como 400. La respuesta final se envía al cliente a través de la función `response`.

Ilustración 100.- Rutas del login.

```php
//rutas para loguearse    "loguearse": Unknown word.
$app->post('/login', loginController::class . ':validarLogin');

$app->group('/admin', function (Group $group) {
    $group->post('/registrarProductor', loginController::class . ':validarCrearProductor');
});
```

Fuente: Propia, 2023.

El código de la Ilustración 100 proporciona rutas para la autenticación. En particular, se define una ruta POST "/login" que invoca el método "validarLogin" del controlador "loginController". Además, se crea un grupo de rutas bajo "/Admin", y dentro de este, se establece una ruta POST "/registrarProductor" que utiliza el método "validarCrearProductor" del mismo controlador. Estas rutas están diseñadas para gestionar la autenticación y el registro de productores en la aplicación.

Ilustración 101.- Creación del token.

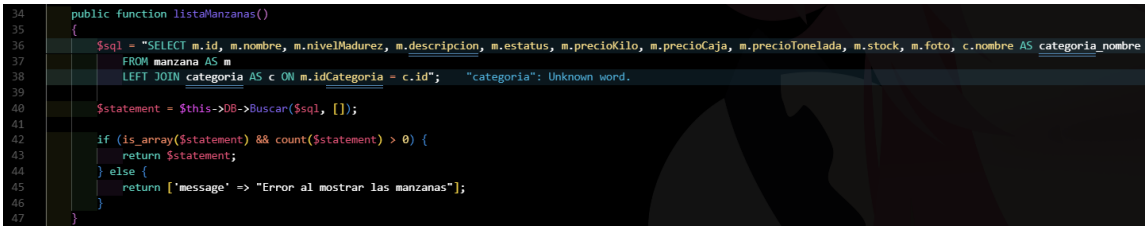
```php
return function (App $app) {
    $app->add(new Tuupola\Middleware\JwtAuthentication([    "Tuupola": Unknown word.
        // Rutas que requieren el token
        "path" => [
            "/Admin", "/Actividad", '/Carrito', '/DerivadosManzana', '/Eventos', '/Manzana', '/Nota', 'Pedidos', 'Ventas', '/PuntoVenta'
        ],
        // "Rutas que no requieren el token"
        "secret" => 'a84125e55c207450dba07c6cb3e7b999',
        "before" => function ($request, $arguments) use ($app) {
            $token = $arguments["decoded"];
            return $request;
        },
        "error" => function ($response, $args) {
            $data["statusCode"] = 401;
            $data["message"] = "Token no valido";
            $response->getBody()->write(
                json_encode($data, JSON_UNESCAPED_SLASHES | JSON_PRETTY_PRINT)
            );
            return $response->withHeader("Content-Type", "application/json");
        }
    ]));
```

Fuente: Propia, 2023.

El fragmento de código de la Ilustración 101 define un middleware de autenticación basado en tokens JWT para una aplicación construida con el framework Slim. El middleware se añade a la aplicación con la finalidad de proteger ciertas rutas que requieren autenticación mediante token. Las rutas protegidas incluyen "/Admin", "/Actividad", '/Carrito', '/DerivadosManzana', '/Eventos', '/Manzana', '/Nota', 'Pedidos', 'Ventas', y '/PuntoVenta'. Se utiliza una clave secreta

('a84125e55c207450dba07c6cb3e7b999') para validar los tokens. Además, se proporciona una función antes de la autenticación donde se extrae y guarda el token decodificado para su uso posterior. En caso de error de autenticación, se devuelve una respuesta con código 401 y un mensaje indicando que el token no es válido en formato JSON.

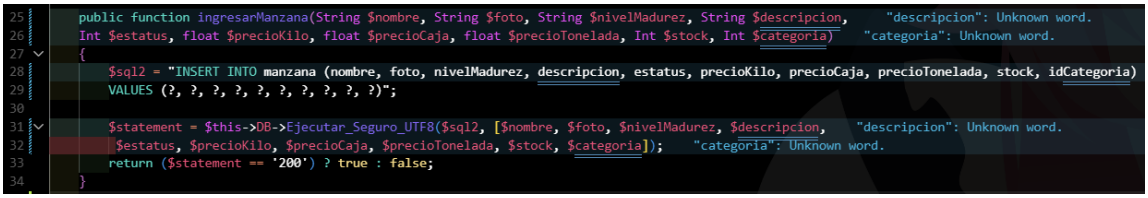
Ilustración 102.- Pantalla de código de la función para listar manzanas.



Fuente: Propia, 2023.

La función de la Ilustración 102 `listaManzanas` realiza una consulta a la base de datos para obtener información detallada sobre las manzanas, incluyendo su identificador, nombre, nivel de madurez, descripción, estado, precios, stock, foto y la categoría a la que pertenecen. La consulta utiliza unión izquierda entre las tablas de manzanas y categorías. Si la consulta tiene resultados, devuelve el array con la información; de lo contrario, devuelve un mensaje de error.

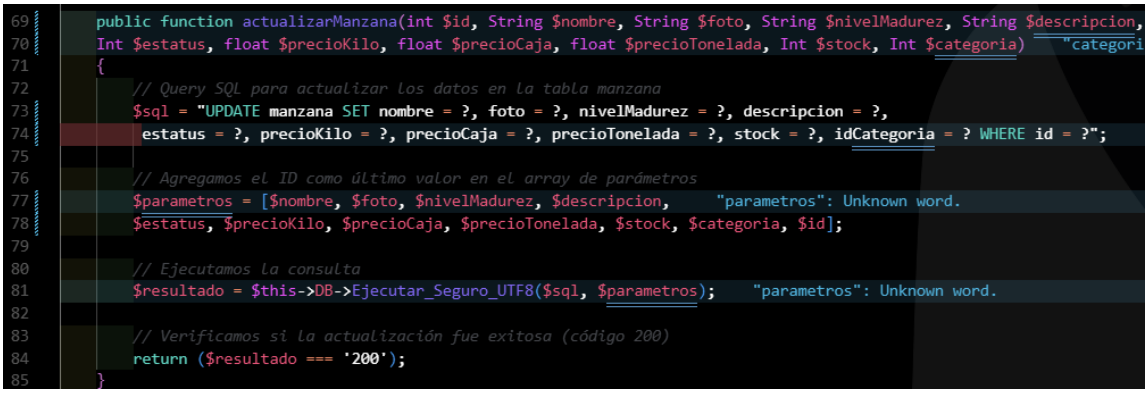
Ilustración 103.- Pantalla de código de la función para ingresar una manzana.



Fuente: Propia, 2023.

La función de la Ilustración 103 "ingresarManzana" añade información de una manzana a la base de datos, incluyendo nombre, foto, nivel de madurez, descripción, estatus, precios (kilo, caja, tonelada), stock y categoría. Retorna verdadero si la operación es exitosa, falso de lo contrario.

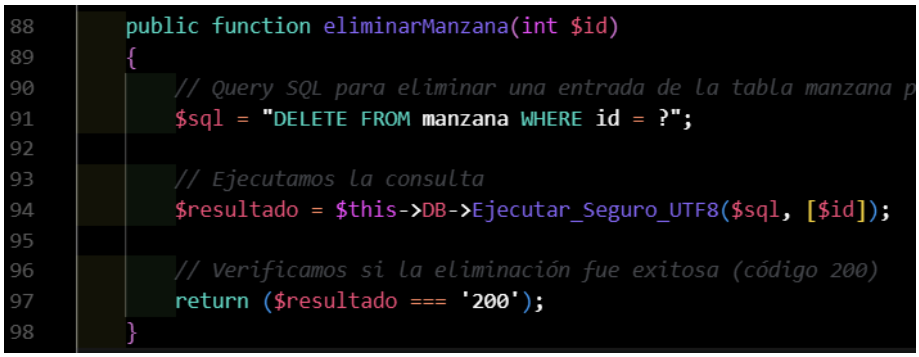
Ilustración 104.- Pantalla de código de la función para actualizar una manzana.



Fuente: Propia, 2023.

La función de la Ilustración 104 `actualizarManzana` modifica los datos de una manzana en la base de datos. Recibe parámetros como ID, nombre, foto, nivel de madurez, descripción, estado, precios y stock. Utiliza una consulta SQL para actualizar la entrada correspondiente en la tabla de manzanas. Retorna `true` si la actualización es exitosa, y `false` de lo contrario.

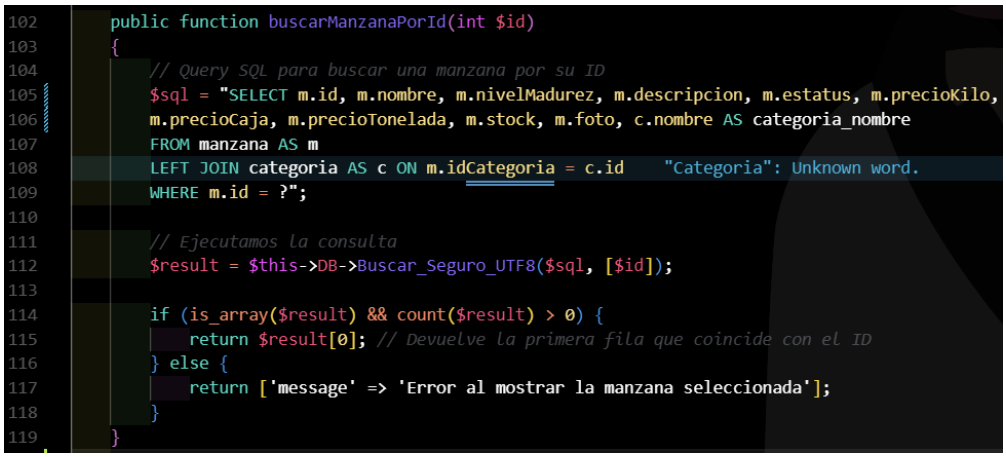
Ilustración 105.- Pantalla de código de la función para eliminar una manzana.



Fuente: Propia, 2023.

La función de la Ilustración 105 "eliminarManzana" toma un parámetro entero que representa el ID de una manzana. Internamente, utiliza una consulta SQL para eliminar la entrada correspondiente en la tabla "manzana" utilizando el ID proporcionado. La ejecución de la consulta se realiza a través de la conexión a la base de datos, y el resultado se verifica para determinar si la eliminación fue exitosa, devolviendo true si el código de resultado es 200 y false en caso contrario.

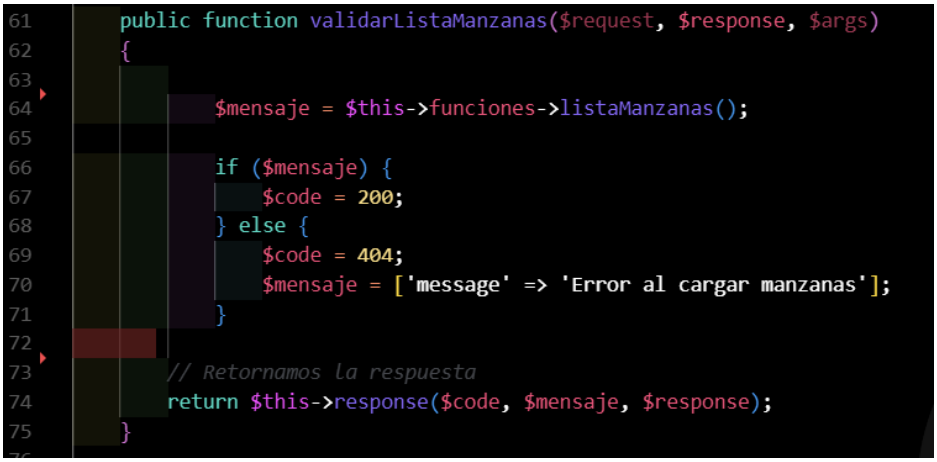
Ilustración 106.- Pantalla de código de la función para buscar por id una manzana.



Fuente: Propia, 2023.

La función de la Ilustración 106 `buscarManzanaPorId` realiza una consulta SQL para obtener información específica de una manzana según su ID. Incluye datos como nombre, nivel de madurez, descripción, estatus, precios, stock, foto y categoría. Se utiliza un JOIN con la tabla de categorías. Si se encuentra la manzana, se devuelve la información; de lo contrario, se emite un mensaje de error.

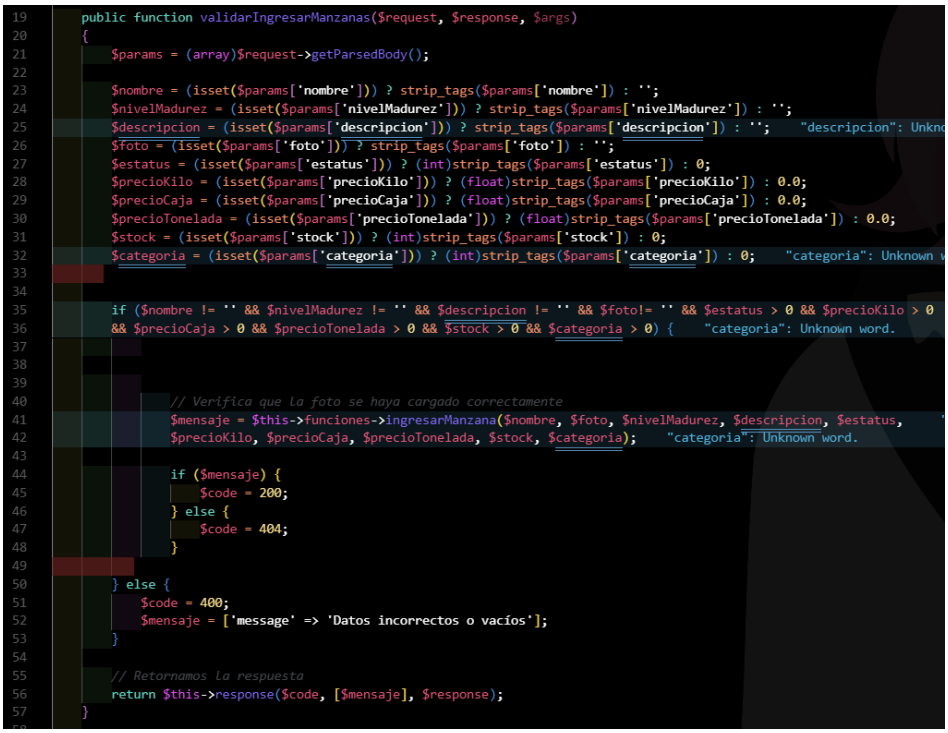
Ilustración 107.- Pantalla de código del controlador para validar listar manzanas.



Fuente: Propia, 2023.

La función de la Ilustración 106 "validarListaManzanas" se encarga de obtener la lista de manzanas mediante la invocación de la función interna "listaManzanas". Si la operación es exitosa, se establece un código de respuesta 200; de lo contrario, se asigna el código 404 junto con un mensaje de error indicando la falla en la carga de las manzanas. Finalmente, la función devuelve la respuesta con el código y el mensaje correspondiente utilizando el método "response".

Ilustración 108.- Pantalla de código del controlador para validar ingresar una manzana.

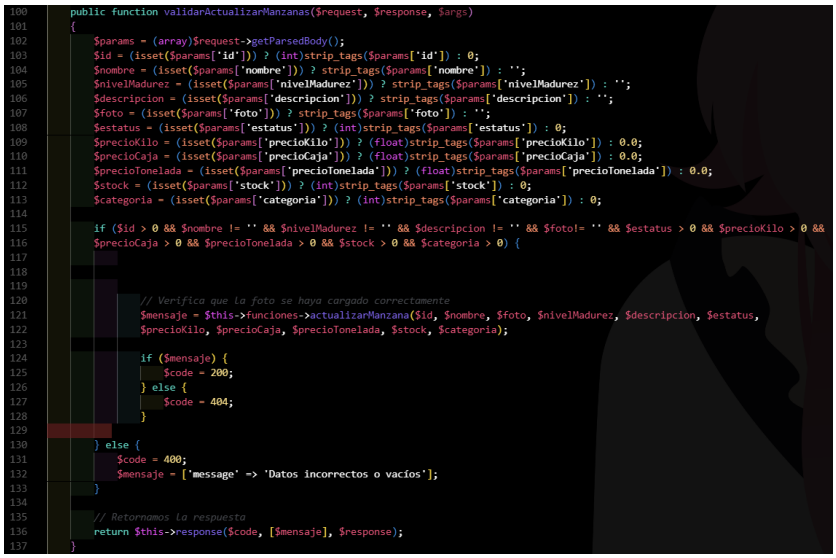


```php
public function validarIngresarManzanas($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
    $nivelMadurez = (isset($params['nivelMadurez'])) ? strip_tags($params['nivelMadurez']) : '';
    $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';    "descripcion": Unkno
    $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
    $estatus = (isset($params['estatus'])) ? (int)strip_tags($params['estatus']) : 0;
    $precioKilo = (isset($params['precioKilo'])) ? (float)strip_tags($params['precioKilo']) : 0.0;
    $precioCaja = (isset($params['precioCaja'])) ? (float)strip_tags($params['precioCaja']) : 0.0;
    $precioTonelada = (isset($params['precioTonelada'])) ? (float)strip_tags($params['precioTonelada']) : 0.0;
    $stock = (isset($params['stock'])) ? (int)strip_tags($params['stock']) : 0;
    $categoria = (isset($params['categoria'])) ? (int)strip_tags($params['categoria']) : 0;    "categoria": Unknown w


    if ($nombre != '' && $nivelMadurez != '' && $descripcion != '' && $foto!= '' && $estatus > 0 && $precioKilo > 0
    && $precioCaja > 0 && $precioTonelada > 0 && $stock > 0 && $categoria > 0) {    "categoria": Unknown word.



        // Verifica que la foto se haya cargado correctamente
        $mensaje = $this->funciones->ingresarManzana($nombre, $foto, $nivelMadurez, $descripcion, $estatus,
        $precioKilo, $precioCaja, $precioTonelada, $stock, $categoria);    "categoria": Unknown word.

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 108 `validarIngresarManzanas` procesa y valida los datos de una solicitud para ingresar información de manzanas. Se verifica que los campos necesarios estén presentes y tengan el formato correcto. Si la validación es exitosa, se intenta ingresar la información utilizando un método llamado `ingresarManzana`. Dependiendo del resultado, se retorna un código de respuesta (200 o 404) indicando el éxito o fallo de la operación. En caso de falla en la validación inicial, se retorna un código 400 con un mensaje indicando datos incorrectos o vacíos.

Ilustración 109.- Pantalla de código del controlador para validar actualizar una manzana.

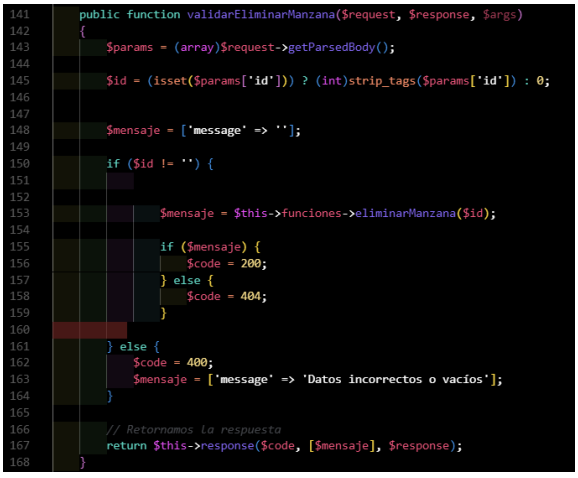


```php
public function validarActualizarManzanas($request, $response, $args)
{
    $params = (array)$request->getParsedBody();
    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;
    $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
    $nivelMadurez = (isset($params['nivelMadurez'])) ? strip_tags($params['nivelMadurez']) : '';
    $descripcion = (isset($params['descripcion'])) ? strip_tags($params['descripcion']) : '';
    $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
    $estatus = (isset($params['estatus'])) ? (int)strip_tags($params['estatus']) : 0;
    $precioKilo = (isset($params['precioKilo'])) ? (float)strip_tags($params['precioKilo']) : 0.0;
    $precioCaja = (isset($params['precioCaja'])) ? (float)strip_tags($params['precioCaja']) : 0.0;
    $precioTonelada = (isset($params['precioTonelada'])) ? (float)strip_tags($params['precioTonelada']) : 0.0;
    $stock = (isset($params['stock'])) ? (int)strip_tags($params['stock']) : 0;
    $categoria = (isset($params['categoria'])) ? (int)strip_tags($params['categoria']) : 0;

    if ($id > 0 && $nombre != '' && $nivelMadurez != '' && $descripcion != '' && $foto!= '' && $estatus > 0 && $precioKilo > 0 &&
    $precioCaja > 0 && $precioTonelada > 0 && $stock > 0 && $categoria > 0) {



        // Verifica que la foto se haya cargado correctamente
        $mensaje = $this->funciones->actualizarManzana($id, $nombre, $foto, $nivelMadurez, $descripcion, $estatus,
        $precioKilo, $precioCaja, $precioTonelada, $stock, $categoria);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 109 `validarActualizarManzanas` recibe datos de una solicitud HTTP, valida y actualiza la información de una manzana en la base de datos. Se verifica la presencia y validez de ciertos campos esenciales. Si la validación es exitosa, se realiza la actualización y se devuelve un código de respuesta junto con un mensaje. En caso de datos incorrectos o vacíos, se devuelve un código de error 400 con un mensaje indicando el problema.

Ilustración 110.- Pantalla de código del controlador para validar eliminar una actividad.

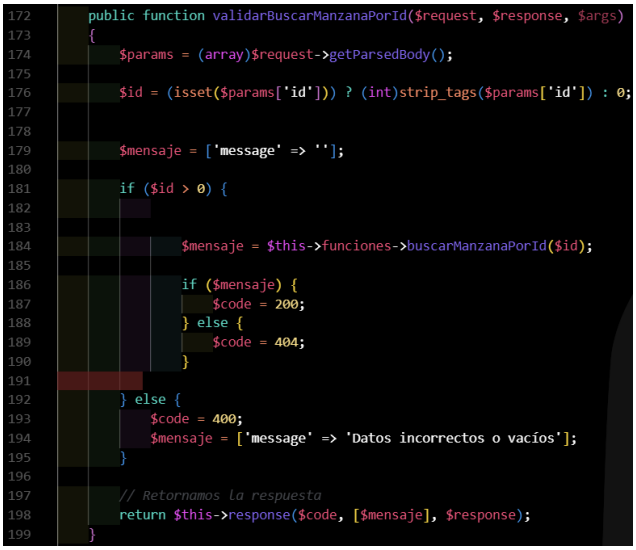


```php
public function validarEliminarManzana($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $id = (isset($params['id'])) ? (int)strip_tags($params['id']) : 0;


    $mensaje = ['message' => ''];

    if ($id != '') {


        $mensaje = $this->funciones->eliminarManzana($id);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 110 `validarEliminarManzana` recibe una solicitud y extrae un identificador de manzana. Luego, intenta eliminar la manzana y devuelve un código de respuesta (200 en caso de éxito, 404 si no se encuentra, 400 si hay datos incorrectos). La respuesta incluye el código y un mensaje asociado.

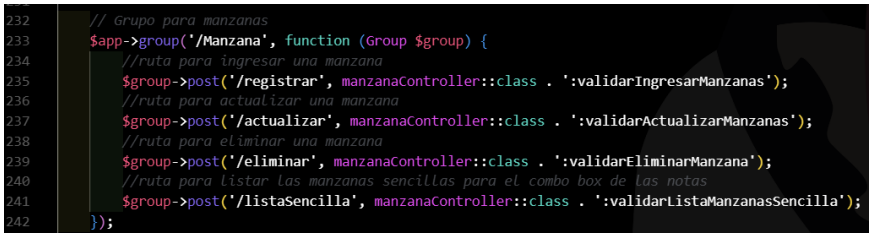
Ilustración 111.- Pantalla de código del controlador para validar buscar por id una manzana.



Fuente: Propia, 2023.

La función 111 `validarBuscarManzanaPorId` toma una solicitud, extrae los parámetros del cuerpo de la solicitud y valida un identificador de manzana. Si el identificador es válido, se utiliza una función auxiliar para buscar la información de la manzana correspondiente. Si se encuentra la información, se devuelve un código de estado 200; de lo contrario, se devuelve un código de estado 404. En caso de un identificador no válido o ausente, se retorna un código de estado 400 con un mensaje indicando datos incorrectos o vacíos.

Ilustración 112.- Rutas de manzanas.



Fuente: Propia, 2023.

El código de la Ilustración 112 utiliza el framework Slim para definir un grupo de rutas relacionadas con manzanas. Dentro de este grupo, se han establecido cuatro rutas distintas: una para registrar una manzana, otra para actualizar una manzana, la tercera para eliminar una manzana, y la quinta para listar manzanas de forma sencilla. Cada ruta está asociada a un controlador de eventos específico que se encargará de validar y procesar las solicitudes correspondientes.

Ilustración 113.- Pantalla de código de la función para listar notas.

```php
public function listaNotas()
{
    $sql = "SELECT n.id, n.cantidad, n.fecha, t.nombre AS tipoNota, m.nombre AS manzanaNota, u.usuario AS usuarioNota
        FROM nota n
        INNER JOIN tipo_Nota t ON n.idTipo = t.id
        INNER JOIN manzana m ON n.idManzana = m.id
        INNER JOIN usuario u ON n.idUsuario = u.id";


    $statement = $this->DB->Buscar($sql);

    if (is_array($statement) && count($statement) > 0) {
        return $statement;
    } else {
        return ['message' =>  "Problemas para obtener las notas"];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 113 "listaNotas" consulta y devuelve información sobre notas, incluyendo identificador, cantidad, fecha, tipo, manzana y usuario. Si hay datos, los retorna; de lo contrario, indica problemas en la obtención.

Ilustración 114.- Pantalla de código de la función para crear una nota.

```php
public function crearNota(int $cantidad, int $tipo, int $manzana, int $usuario)
{
    // Obtener la fecha actual en formato 'YYYY-MM-DD'
    $fecha = date('Y-m-d');

    // Consulta SQL para insertar la nota con la fecha actual
    $sql = "INSERT INTO nota (cantidad, fecha, idTipo, idManzana, idUsuario) VALUES (?, ?, ?, ?, ?)";

    // Ejecutar la consulta con los parámetros
    $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, [$cantidad, $fecha, $tipo, $manzana, $usuario]);

    // Verificar el resultado de la inserción
    return ($statement == '200') ? true : false;
}
```

Fuente: Propia, 2023.

La función de la Ilustración 114 "crearNota" inserta una nueva nota en la base de datos con la cantidad, tipo, manzana y usuario dados. Utiliza la fecha actual y devuelve true si la inserción es exitosa, false en caso contrario.

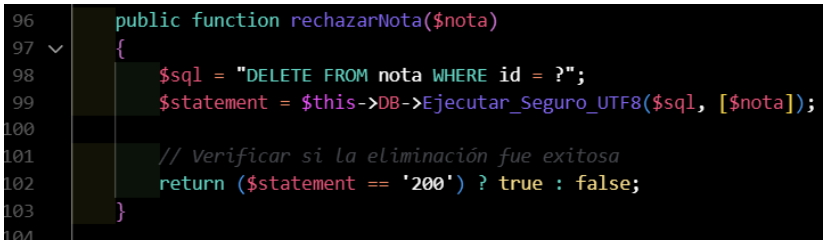
Ilustración 115.- Pantalla de código de la función para aceptar una nota.

```php
public function aceptarNota(int $nota)
{
    // Obtener información de la nota, incluyendo el idTipo
    $sql = "SELECT n.cantidad, t.nombre AS tipoNota, n.idManzana as manzana FROM nota n
        INNER JOIN tipo_Nota t ON n.idTipo = t.id
        WHERE n.id = ?";
    $notaInfo = $this->DB->Buscar_Seguro_UTF8($sql, [$nota]);

    $notaInfo = $notaInfo[0];
    // Comprobar si se encontró la nota
    if ($notaInfo) {
        $cantidad = $notaInfo['cantidad'];
        $tipoNota = $notaInfo['tipoNota'];
        $idManzana = $notaInfo['manzana'];

        // Comprobar el tipo de nota
        if ($tipoNota == "aumentar") {
            // Incrementar el stock de la manzana
            $sqlUpdate = "UPDATE manzana SET stock = stock + ? WHERE id = ?";
            $this->DB->Ejecutar_Seguro_UTF8($sqlUpdate, [$cantidad, $idManzana]);
            // Eliminar la nota después de procesarla
            $sqlDelete = "DELETE FROM nota WHERE id = ?";
            $statement = $this->DB->Ejecutar_Seguro_UTF8($sqlDelete, [$nota]);
            return ($statement == '200') ? true : false;
        } elseif ($tipoNota == "disminuir") {
            // Comprobar si la cantidad a disminuir no es mayor que el stock actual
            $sqlStock = "SELECT stock FROM manzana WHERE id = ?";
            $manzanaInfo = $this->DB->Buscar_Seguro_UTF8($sqlStock, [$idManzana]);

            if ($manzanaInfo && $manzanaInfo['stock'] >= $cantidad) {
                // Disminuir el stock de la manzana
                $sqlUpdate = "UPDATE manzana SET stock = stock - ? WHERE id = ?";
                $this->DB->Ejecutar_Seguro_UTF8($sqlUpdate, [$cantidad, $idManzana]);
                // Eliminar la nota después de procesarla
                $sqlDelete = "DELETE FROM nota WHERE id = ?";
                $statement = $this->DB->Ejecutar_Seguro_UTF8($sqlDelete, [$nota]);
                return ($statement == '200') ? true : false;
            } else {
                return "La cantidad a disminuir es mayor que el stock actual de la manzana.";
            }
        }

        return "Error inesperado, lo sentimos.";
    } else {
        return "Nota no encontrada.";
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 115 "aceptarNota" procesa transacciones de notas, ajustando el stock de manzanas. Obtiene la información de la nota, verifica su existencia, y realiza operaciones de aumento o disminución en el stock de manzanas según el tipo de nota. La función devuelve mensajes indicando el resultado de la operación, como éxito, error inesperado, falta de la nota, o si la cantidad de disminución excede el stock disponible.

Ilustración 116.- Pantalla de código de la función para rechazar una nota.

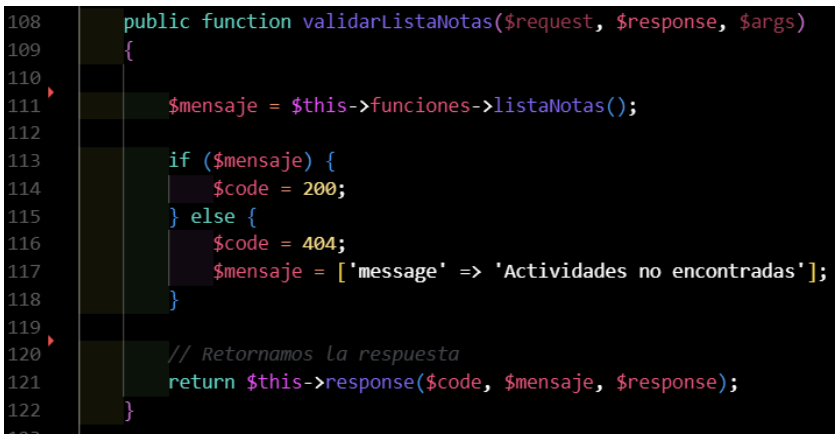


```php
public function rechazarNota($nota)
{
    $sql = "DELETE FROM nota WHERE id = ?";
    $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, [$nota]);

    // Verificar si la eliminación fue exitosa
    return ($statement == '200') ? true : false;
}
```

Fuente: Propia, 2023.

La función de la Ilustración 116 "rechazarNota" elimina una nota específica de la base de datos utilizando una sentencia SQL DELETE. El identificador de la nota se pasa como parámetro y se ejecuta de manera segura. La función verifica el éxito de la eliminación a través de la respuesta del servidor, devolviendo true si fue exitosa y false en caso contrario.

Ilustración 117.- Pantalla de código del controlador para validar listar notas.



```php
public function validarListaNotas($request, $response, $args)
{

    $mensaje = $this->funciones->listaNotas();

    if ($mensaje) {
        $code = 200;
    } else {
        $code = 404;
        $mensaje = ['message' => 'Actividades no encontradas'];
    }

    // Retornamos la respuesta
    return $this->response($code, $mensaje, $response);
}
```

Fuente: Propia, 2023

Esta función de la Ilustración 117 llamada `validarListaNotas`, utiliza el método `listaNotas` para obtener las notas. Si existen, devuelve un código de respuesta 200; de lo contrario, devuelve 404 junto con un mensaje indicando la falta de actividades.

Ilustración 118.- Pantalla de código del controlador para validar crear una nota.

```php
public function validarCrearNota($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $cantidad = (isset($params['cantidad'])) ? (int)strip_tags($params['cantidad']) : 0;
    $tipo = (isset($params['tipo'])) ? (int)strip_tags($params['tipo']) : 0;
    $manzana = (isset($params['manzana'])) ? (int)strip_tags($params['manzana']) : 0;
    $usuario = (isset($params['usuario'])) ? (int)strip_tags($params['usuario']) : 0;



    $mensaje = ['message' => ''];

    if ($cantidad > 0 && $tipo > 0 && $manzana > 0 && $usuario > 0) {
        $mensaje = $this->funciones->crearNota($cantidad, $tipo, $manzana, $usuario);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 118 valida y procesa la creación de una nota con parámetros específicos. Verifica que los datos necesarios sean mayores que cero, llama a la función `crearNota` y establece el código de respuesta. Si tiene éxito, el código es 200; de lo contrario, es 404. En caso de datos incorrectos o vacíos, el código es 400 con un mensaje indicativo. Retorna la respuesta con el código y el mensaje.

Ilustración 119.- Pantalla de código del controlador para validar aceptar una nota.

```php
public function validarAceptarNota($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $nota = (isset($params['nota'])) ? (int)strip_tags($params['nota']) : 0;



    $mensaje = ['message' => ''];

    if ($nota > 0) {
        $mensaje = $this->funciones->aceptarNota($nota);

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 119 `validarAceptarNota` procesa una solicitud, verifica y extrae el valor de la nota. Luego, valida si es un número entero positivo. Si es así, llama a `aceptarNota` y establece el código de respuesta en base al resultado. Si la nota es inválida o la operación falla, el código es 404; si los datos son incorrectos o vacíos, el código es 400. La función devuelve la respuesta HTTP con el código y mensaje correspondientes.

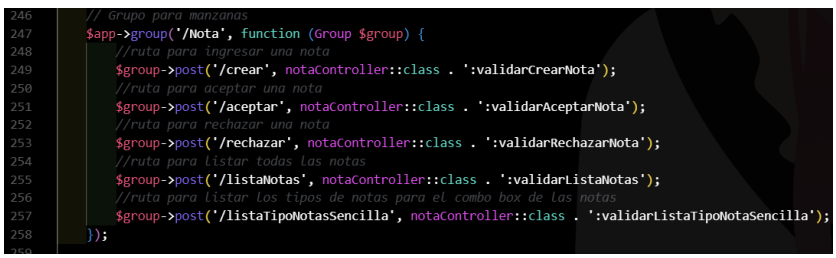
Ilustración 120.- Pantalla de código del controlador para validar rechazar una nota.

```php
    public function validarRechazarNota($request, $response, $args)
    {
        $params = (array)$request->getParsedBody();

        $nota = (isset($params['nota'])) ? (int)strip_tags($params['nota']) : 0;


        $mensaje = ['message' => ''];

        if ($nota > 0) {
            $mensaje = $this->funciones->rechazarNota($nota);

            if ($mensaje) {
                $code = 200;
            } else {
                $code = 404;
            }
        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
        }

        // Retornamos la respuesta
        return $this->response($code, [$mensaje], $response);
    }
```

Fuente: Propia, 2023.

La función de la Ilustración 120 `validarRechazarNota` verifica la presencia y validez de la nota en la solicitud. Si es válida, intenta rechazarla mediante la función `rechazarNota`. Se establece un código de respuesta 200 si tiene éxito y 404 si falla. Si la nota es inválida o no está presente, devuelve un código 400 con un mensaje indicando datos incorrectos o vacíos.

Ilustración 121.- Ruta de notas.

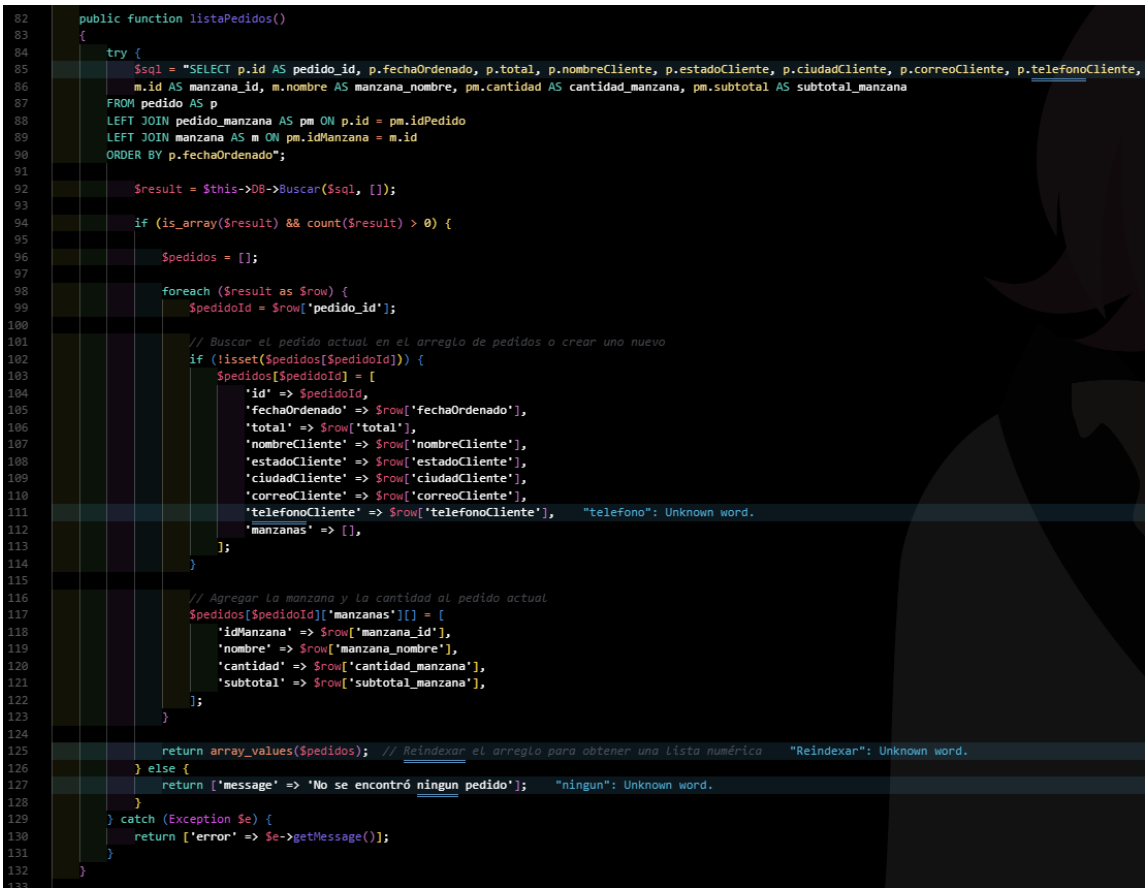


```php
// Grupo para manzanas
$app->group('/Nota', function (Group $group) {
    //ruta para ingresar una nota
    $group->post('/crear', notaController::class . ':validarCrearNota');
    //ruta para aceptar una nota
    $group->post('/aceptar', notaController::class . ':validarAceptarNota');
    //ruta para rechazar una nota
    $group->post('/rechazar', notaController::class . ':validarRechazarNota');
    //ruta para listar todas las notas
    $group->post('/listaNotas', notaController::class . ':validarListaNotas');
    //ruta para listar los tipos de notas para el combo box de las notas
    $group->post('/listaTipoNotasSencilla', notaController::class . ':validarListaTipoNotaSencilla');
});
```

Fuente: Propia, 2023.

En la Ilustración 121 se define un grupo para operaciones relacionadas con notas. Incluye rutas para crear, aceptar, rechazar y listar notas, así como obtener tipos de notas para un cuadro combinado. Cada ruta se vincula a un controlador específico llamado `notaController`.

Ilustración 122.- Pantalla de código de la función para listar pedidos.

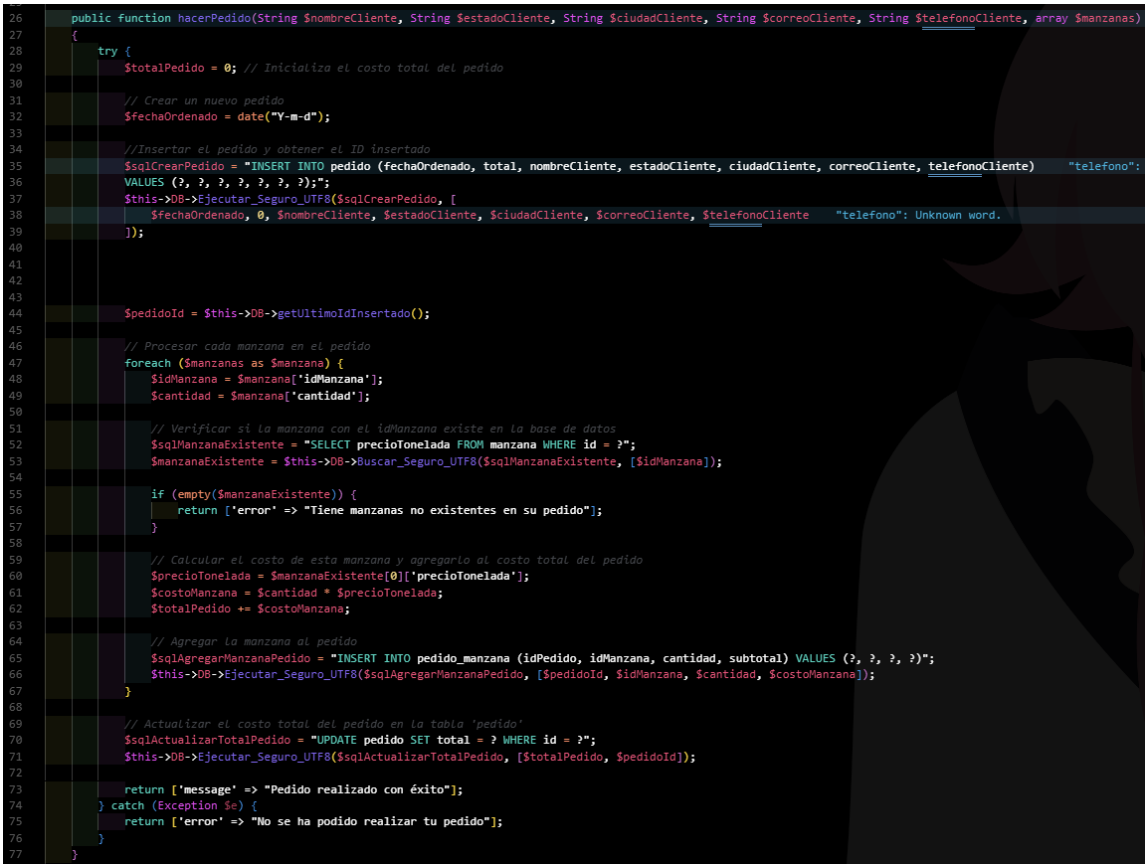


```
public function listaPedidos()
{
    try {
        $sql = "SELECT p.id AS pedido_id, p.fechaOrdenado, p.total, p.nombreCliente, p.estadoCliente, p.ciudadCliente, p.correoCliente, p.telefonoCliente,
        m.id AS manzana_id, m.nombre AS manzana_nombre, pm.cantidad AS cantidad_manzana, pm.subtotal AS subtotal_manzana
    FROM pedido AS p
    LEFT JOIN pedido_manzana AS pm ON p.id = pm.idPedido
    LEFT JOIN manzana AS m ON pm.idManzana = m.id
    ORDER BY p.fechaOrdenado";

        $result = $this->DB->Buscar($sql, []);

        if (is_array($result) && count($result) > 0) {

            $pedidos = [];

            foreach ($result as $row) {
                $pedidoId = $row['pedido_id'];

                // Buscar el pedido actual en el arreglo de pedidos o crear uno nuevo
                if (!isset($pedidos[$pedidoId])) {
                    $pedidos[$pedidoId] = [
                        'id' => $pedidoId,
                        'fechaOrdenado' => $row['fechaOrdenado'],
                        'total' => $row['total'],
                        'nombreCliente' => $row['nombreCliente'],
                        'estadoCliente' => $row['estadoCliente'],
                        'ciudadCliente' => $row['ciudadCliente'],
                        'correoCliente' => $row['correoCliente'],
                        'telefonoCliente' => $row['telefonoCliente'],    "telefono": Unknown word.
                        'manzanas' => [],
                    ];
                }

                // Agregar la manzana y la cantidad al pedido actual
                $pedidos[$pedidoId]['manzanas'][] = [
                    'idManzana' => $row['manzana_id'],
                    'nombre' => $row['manzana_nombre'],
                    'cantidad' => $row['cantidad_manzana'],
                    'subtotal' => $row['subtotal_manzana'],
                ];
            }

            return array_values($pedidos);  // Reindexar el arreglo para obtener una lista numérica    "Reindexar": Unknown word.
        } else {
            return ['message' => 'No se encontró ningun pedido'];    "ningun": Unknown word.
        }
    } catch (Exception $e) {
        return ['error' => $e->getMessage()];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 122 `listaPedidos` consulta y organiza datos detallados de pedidos, clientes y manzanas en un arreglo estructurado. Si hay pedidos, devuelve la información organizada; si no hay pedidos, indica la ausencia de datos. En caso de errores, informa sobre el problema ocurrido.

Ilustración 123.- Pantalla de código de la función para hacer un pedido.

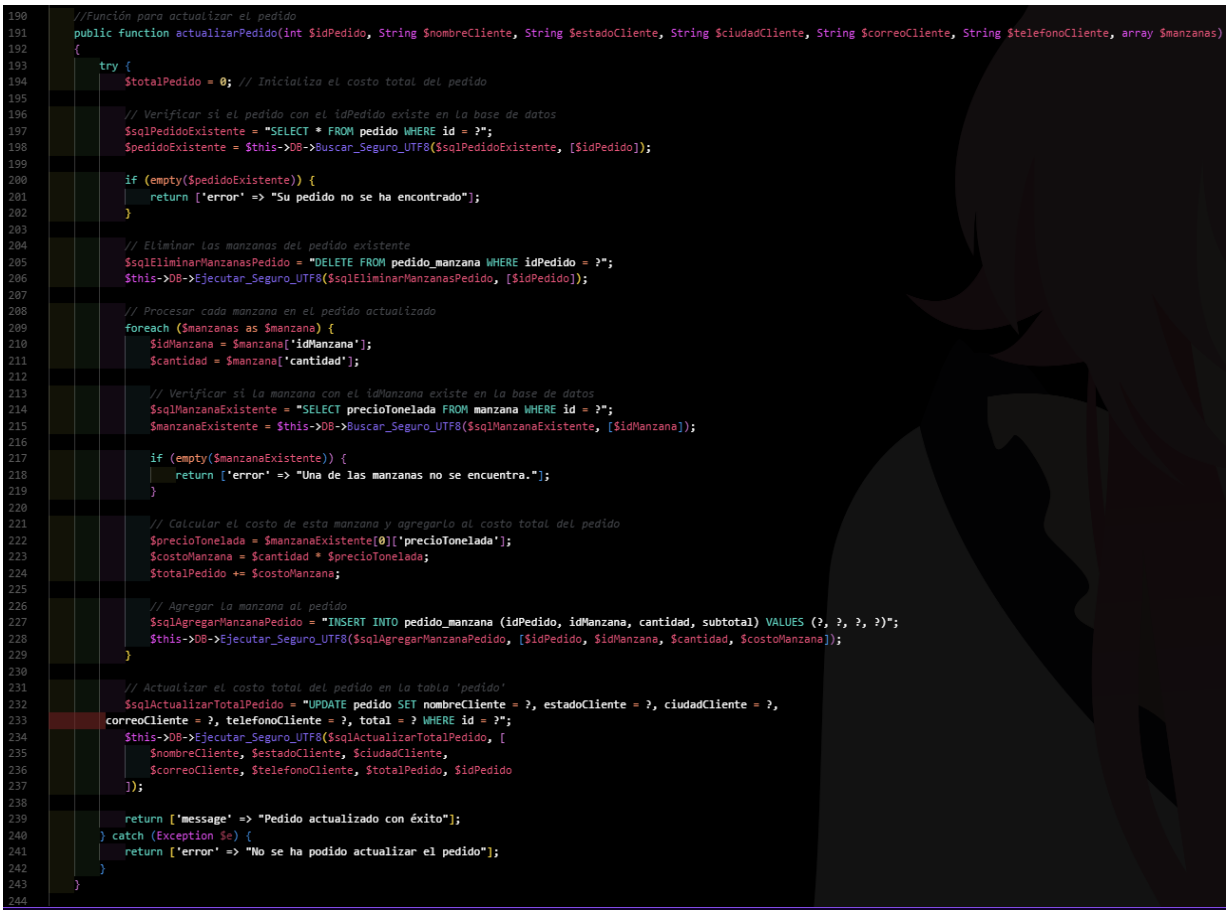


```php
public function hacerPedido(String $nombreCliente, String $estadoCliente, String $ciudadCliente, String $correoCliente, String $telefonoCliente, array $manzanas)
{
    try {
        $totalPedido = 0; // Inicializa el costo total del pedido

        // Crear un nuevo pedido
        $fechaOrdenado = date("Y-m-d");

        //Insertar el pedido y obtener el ID insertado
        $sqlCrearPedido = "INSERT INTO pedido (fechaOrdenado, total, nombreCliente, estadoCliente, ciudadCliente, correoCliente, telefonoCliente)     "telefono":
        VALUES (?, ?, ?, ?, ?, ?, ?);";
        $this->DB->Ejecutar_Seguro_UTF8($sqlCrearPedido, [
            $fechaOrdenado, 0, $nombreCliente, $estadoCliente, $ciudadCliente, $correoCliente, $telefonoCliente    "telefono": Unknown word.
        ]);




        $pedidoId = $this->DB->getUltimoIdInsertado();

        // Procesar cada manzana en el pedido
        foreach ($manzanas as $manzana) {
            $idManzana = $manzana['idManzana'];
            $cantidad = $manzana['cantidad'];

            // Verificar si la manzana con el idManzana existe en la base de datos
            $sqlManzanaExistente = "SELECT precioTonelada FROM manzana WHERE id = ?";
            $manzanaExistente = $this->DB->Buscar_Seguro_UTF8($sqlManzanaExistente, [$idManzana]);

            if (empty($manzanaExistente)) {
                return ['error' => "Tiene manzanas no existentes en su pedido"];
            }

            // Calcular el costo de esta manzana y agregarlo al costo total del pedido
            $precioTonelada = $manzanaExistente[0]['precioTonelada'];
            $costoManzana = $cantidad * $precioTonelada;
            $totalPedido += $costoManzana;

            // Agregar la manzana al pedido
            $sqlAgregarManzanaPedido = "INSERT INTO pedido_manzana (idPedido, idManzana, cantidad, subtotal) VALUES (?, ?, ?, ?)";
            $this->DB->Ejecutar_Seguro_UTF8($sqlAgregarManzanaPedido, [$pedidoId, $idManzana, $cantidad, $costoManzana]);
        }

        // Actualizar el costo total del pedido en la tabla 'pedido'
        $sqlActualizarTotalPedido = "UPDATE pedido SET total = ? WHERE id = ?";
        $this->DB->Ejecutar_Seguro_UTF8($sqlActualizarTotalPedido, [$totalPedido, $pedidoId]);

        return ['message' => "Pedido realizado con éxito"];
    } catch (Exception $e) {
        return ['error' => "No se ha podido realizar tu pedido"];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 123 `hacerPedido` realiza la creación de un nuevo pedido, calcula el costo total considerando las manzanas seleccionadas y las agrega al pedido junto con su información correspondiente. Utiliza consultas SQL seguras para interactuar con la base de datos, verificando la existencia de las manzanas seleccionadas y actualizando el costo total del pedido. En caso de éxito, devuelve un mensaje indicando que el pedido se ha realizado exitosamente; en caso de error, informa que no se pudo completar el pedido. La función maneja excepciones para asegurar un proceso controlado y seguro.

Ilustración 124.- Pantalla de código de la función para actualizar un pedido.

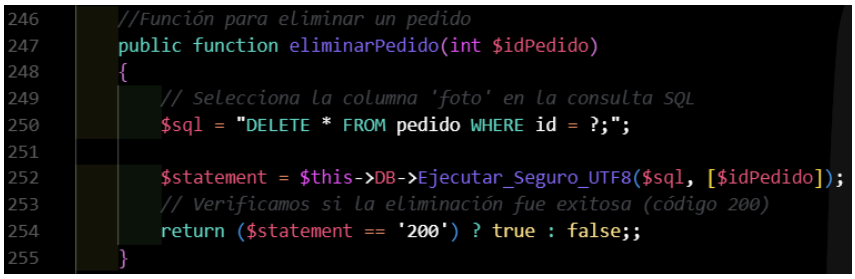


```php
//Función para actualizar el pedido
public function actualizarPedido(int $idPedido, String $nombreCliente, String $estadoCliente, String $ciudadCliente, String $correoCliente, String $telefonoCliente, array $manzanas)
{
    try {
        $totalPedido = 0; // Inicializa el costo total del pedido

        // Verificar si el pedido con el idPedido existe en la base de datos
        $sqlPedidoExistente = "SELECT * FROM pedido WHERE id = ?";
        $pedidoExistente = $this->DB->Buscar_Seguro_UTF8($sqlPedidoExistente, [$idPedido]);

        if (empty($pedidoExistente)) {
            return ['error' => "Su pedido no se ha encontrado"];
        }

        // Eliminar las manzanas del pedido existente
        $sqlEliminarManzanasPedido = "DELETE FROM pedido_manzana WHERE idPedido = ?";
        $this->DB->Ejecutar_Seguro_UTF8($sqlEliminarManzanasPedido, [$idPedido]);

        // Procesar cada manzana en el pedido actualizado
        foreach ($manzanas as $manzana) {
            $idManzana = $manzana['idManzana'];
            $cantidad = $manzana['cantidad'];

            // Verificar si la manzana con el idManzana existe en la base de datos
            $sqlManzanaExistente = "SELECT precioTonelada FROM manzana WHERE id = ?";
            $manzanaExistente = $this->DB->Buscar_Seguro_UTF8($sqlManzanaExistente, [$idManzana]);

            if (empty($manzanaExistente)) {
                return ['error' => "Una de las manzanas no se encuentra."];
            }

            // Calcular el costo de esta manzana y agregarlo al costo total del pedido
            $precioTonelada = $manzanaExistente[0]['precioTonelada'];
            $costoManzana = $cantidad * $precioTonelada;
            $totalPedido += $costoManzana;

            // Agregar la manzana al pedido
            $sqlAgregarManzanaPedido = "INSERT INTO pedido_manzana (idPedido, idManzana, cantidad, subtotal) VALUES (?, ?, ?, ?)";
            $this->DB->Ejecutar_Seguro_UTF8($sqlAgregarManzanaPedido, [$idPedido, $idManzana, $cantidad, $costoManzana]);
        }

        // Actualizar el costo total del pedido en la tabla 'pedido'
        $sqlActualizarTotalPedido = "UPDATE pedido SET nombreCliente = ?, estadoCliente = ?, ciudadCliente = ?,
    correoCliente = ?, telefonoCliente = ?, total = ? WHERE id = ?";
        $this->DB->Ejecutar_Seguro_UTF8($sqlActualizarTotalPedido, [
            $nombreCliente, $estadoCliente, $ciudadCliente,
            $correoCliente, $telefonoCliente, $totalPedido, $idPedido
        ]);

        return ['message' => "Pedido actualizado con éxito"];
    } catch (Exception $e) {
        return ['error' => "No se ha podido actualizar el pedido"];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 124 "actualizarPedido" recibe como parámetros la información actualizada de un pedido, incluyendo el ID del pedido, datos del cliente, y una lista de manzanas con sus respectivas cantidades. La función primero verifica la existencia del pedido en la base de datos y, en caso de no encontrarlo, devuelve un mensaje de error. Luego, elimina las manzanas asociadas al pedido existente. Posteriormente, procesa cada manzana del nuevo pedido, verificando su existencia en la base de datos. Calcula el costo de cada manzana y actualiza el costo total del pedido. Se procede a insertar las manzanas actualizadas en la tabla de relación "pedido_manzana". Finalmente, se actualiza el costo total del pedido en la tabla principal "pedido" con los datos del cliente actualizados. En caso de algún error durante el proceso, se devuelve un mensaje de error; de lo contrario, se confirma el éxito de la actualización del pedido.

Ilustración 125.- Pantalla de código de la función para eliminar un pedido.



```
246     //Función para eliminar un pedido
247     public function eliminarPedido(int $idPedido)
248     {
249         // Selecciona la columna 'foto' en la consulta SQL
250         $sql = "DELETE * FROM pedido WHERE id = ?;";
251 
252         $statement = $this->DB->Ejecutar_Seguro_UTF8($sql, [$idPedido]);
253         // Verificamos si la eliminación fue exitosa (código 200)
254         return ($statement == '200') ? true : false;;
255     }
```

Fuente: Propia, 2023.

La función de la Ilustración 125 "eliminarPedido" se encarga de eliminar un pedido de la base de datos. Recibe como parámetro el ID del pedido a eliminar. La consulta SQL asociada busca eliminar la entrada correspondiente en la tabla de pedidos utilizando el ID proporcionado. La ejecución de la consulta se realiza a través de un objeto de base de datos. La función devuelve `true` si la eliminación se lleva a cabo con éxito (código 200), de lo contrario, devuelve `false`.

Ilustración 126.- Pantalla de código de la función para liberar un pedido parte 1.

```
304     public function liberarPedido(int $idPedido)
305 {
306     try {
307         // Obtener los datos del pedido a cerrar
308         $sqlObtenerPedido = "SELECT * FROM pedido WHERE id = ?";
309         $pedido = $this->DB->Buscar_Seguro_UTF8($sqlObtenerPedido, [$idPedido]);
310 
311         if (empty($pedido)) {
312             return ['error' => "Lo lamentamos, su pedido no se encuentra"];
313         }
314 
315         // Verificar el stock de cada manzana en el pedido
316         $sqlVerificarStock = "SELECT pm.idManzana, m.stock, SUM(pm.cantidad) as total_cantidad
317             FROM pedido_manzana pm
318             INNER JOIN manzana m ON pm.idManzana = m.id
319             WHERE pm.idPedido = ?
320             GROUP BY pm.idManzana";
321         $resultVerificarStock = $this->DB->Buscar_Seguro_UTF8($sqlVerificarStock, [$idPedido]);
322 
323         foreach ($resultVerificarStock as $row) {
324             $idManzana = $row['idManzana'];
325             $stockDisponible = $row['stock'];
326             $cantidadPedido = $row['total_cantidad'];
327 
328             if ($stockDisponible < $cantidadPedido) {
329                 return ['error' => "No hay suficiente stock de la manzana con ID $idManzana para satisfacer el pedido"];
330             }
331         }
332 
333         // Actualizar el stock de cada manzana
334         foreach ($resultVerificarStock as $row) {
335             $idManzana = $row['idManzana'];
336             $cantidadPedido = $row['total_cantidad'];
337 
338             $sqlActualizarStock = "UPDATE manzana SET stock = stock - ? WHERE id = ?";
339             $this->DB->Ejecutar_Seguro_UTF8($sqlActualizarStock, [$cantidadPedido, $idManzana]);
340         }
```

Fuente: Propia, 2023.

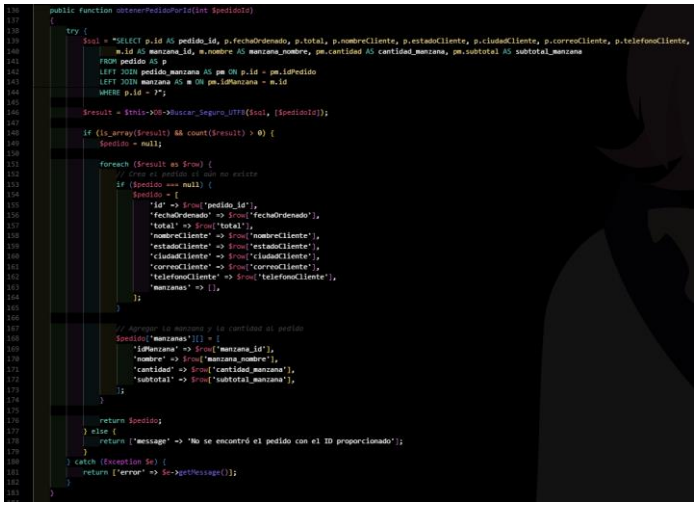
Ilustración 127.- Pantalla de código de la función para liberar un pedido parte 2.

```
        // Crear una nueva venta con los datos del pedido y obtener el ID insertado
        $fechaLiberado = date("Y-m-d");
        $total = $pedido[0]["total"];
        $nombreCliente = $pedido[0]['nombreCliente'];
        $estadoCliente = $pedido[0]['estadoCliente'];
        $ciudadCliente = $pedido[0]['ciudadCliente'];
        $correoCliente = $pedido[0]['correoCliente'];
        $telefonoCliente = $pedido[0]['telefonoCliente'];

        $sqlCrearVenta = "INSERT INTO venta (fechaLiberado, total, nombreCliente, estadoCliente, ciudadCliente, correoCliente, telefonoCliente)
        VALUES (?, ?, ?, ?, ?, ?, ?);";
        $ventaIdQuery = $this->DB->Ejecutar_Seguro_UTF8($sqlCrearVenta, [
            $fechaLiberado, $total, $nombreCliente, $estadoCliente, $ciudadCliente, $correoCliente, $telefonoCliente
        ]);
        $ventaId =  $this->DB->getUltimoIdInsertado();

        // Transferir los registros de pedido_manzana a venta_manzana
        $sqlTransferirManzanas = "INSERT INTO venta_manzana (idVenta, idManzana, cantidad, subtotal)
        SELECT ?, idManzana, cantidad, subtotal FROM pedido_manzana WHERE idPedido = ?";
        $this->DB->Ejecutar_Seguro_UTF8($sqlTransferirManzanas, [$ventaId, $idPedido]);

        // Eliminar el pedido
        $sqlEliminarPedido = "DELETE FROM pedido WHERE id = ?";
        $this->DB->Ejecutar_Seguro_UTF8($sqlEliminarPedido, [$idPedido]);

        return ['message' => "Pedido liberado"];
    } catch (Exception $e) {
        return ['error' => "Error al liberar pedido"];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 126 y 127 `liberarPedido` realiza varias operaciones al cerrar un pedido. Primero, obtiene los datos del pedido a cerrar y verifica si el pedido existe. Luego, verifica el stock disponible para cada manzana en el pedido y asegura que haya suficiente stock para satisfacer el pedido. Después, actualiza el stock de cada manzana en la base de datos. Posteriormente, crea una nueva venta con los datos del pedido, obteniendo un nuevo ID de venta. Transfiere los registros de manzanas del pedido a la tabla de venta_manzana. Finalmente, elimina el pedido original. En caso de algún error durante el proceso, la función devuelve un mensaje de error correspondiente. Si todo se ejecuta correctamente, se devuelve un mensaje indicando que el pedido ha sido liberado.

Ilustración 128.- Pantalla de código de la función para obtener por id un pedido.



```
public function obtenerPedidoPorId($pedidoId)
{
    try {
        $sql = "SELECT p.id AS pedido_id, p.fechaOrdenado, p.total, p.nombreCliente, p.estadoCliente, p.ciudadCliente, p.correoCliente, p.telefonoCliente,
                m.id AS manzana_id, m.nombre AS manzana_nombre, pm.cantidad AS cantidad_manzana, pm.subtotal AS subtotal_manzana
            FROM pedido AS p
            LEFT JOIN pedido_manzana AS pm ON p.id = pm.idPedido
            LEFT JOIN manzana AS m ON pm.idManzana = m.id
            WHERE p.id = ?";

        $result = $this->DB->Buscar_Seguro_UTF8($sql, [$pedidoId]);

        if (is_array($result) && count($result) > 0) {
            $pedido = null;

            foreach ($result as $row) {
                // Crea el pedido si aún no existe
                if ($pedido === null) {
                    $pedido = [
                        'id' => $row['pedido_id'],
                        'fechaOrdenado' => $row['fechaOrdenado'],
                        'total' => $row['total'],
                        'nombreCliente' => $row['nombreCliente'],
                        'estadoCliente' => $row['estadoCliente'],
                        'ciudadCliente' => $row['ciudadCliente'],
                        'correoCliente' => $row['correoCliente'],
                        'telefonoCliente' => $row['telefonoCliente'],
                        'manzanas' => [],
                    ];
                }

                // Agregar la manzana y la cantidad al pedido
                $pedido['manzanas'][] = [
                    'idManzana' => $row['manzana_id'],
                    'nombre' => $row['manzana_nombre'],
                    'cantidad' => $row['cantidad_manzana'],
                    'subtotal' => $row['subtotal_manzana'],
                ];
            }

            return $pedido;
        } else {
            return ['message' => 'No se encontró el pedido con el ID proporcionado'];
        }
    } catch (Exception $e) {
        return ['error' => $e->getMessage()];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 128 `obtenerPedidoPorId` obtiene información detallada de un pedido mediante su ID, utilizando una consulta SQL con JOINs. El resultado se estructura en un arreglo que incluye datos del pedido y una lista de manzanas con cantidades y subtotales. Se manejan casos de éxito y error, retornando el pedido o un mensaje de no encontrarlo. Se capturan excepciones y se devuelve un mensaje de error en caso de surgir problemas.

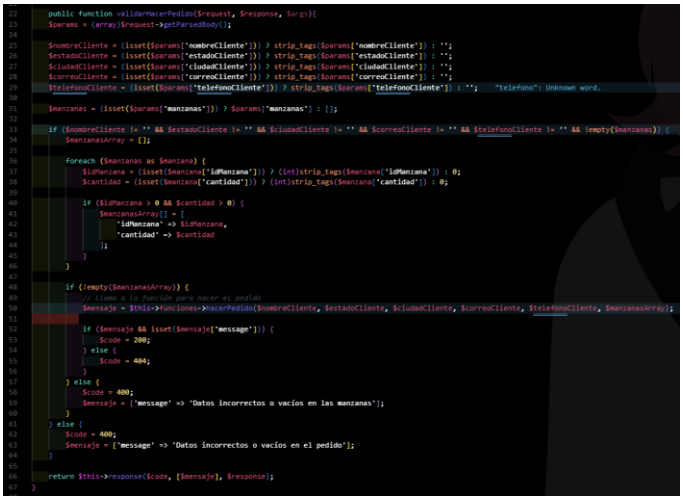
Ilustración 129.- Pantalla de código del controlador para validar listar pedidos.

```
70  public function validarListaPedidos($request, $response, $args)
71  {
72
73
74      $mensaje = $this->funciones->listaPedidos();
75
76      if ($mensaje) {
77          $code = 200;
78      } else {
79          $code = 404;
80          $mensaje = ['message' => 'Error al cargar los pedidos'];
81      }
82
83
84      // Retornamos la respuesta
85      return $this->response($code, $mensaje, $response);
86  }
```

Fuente: Propia, 2023.

La función de la Ilustración 129 "validarListaPedidos" verifica la existencia de una lista de pedidos a través del método "listaPedidos" de la clase funciones. Si la lista existe, se devuelve con un código de estado 200; de lo contrario, se establece un código 404 y se genera un mensaje de error. Finalmente, la respuesta con el código y mensaje correspondientes se retorna mediante el método "response".

Ilustración 130.- Pantalla de código del controlador para validar hacer un pedido.

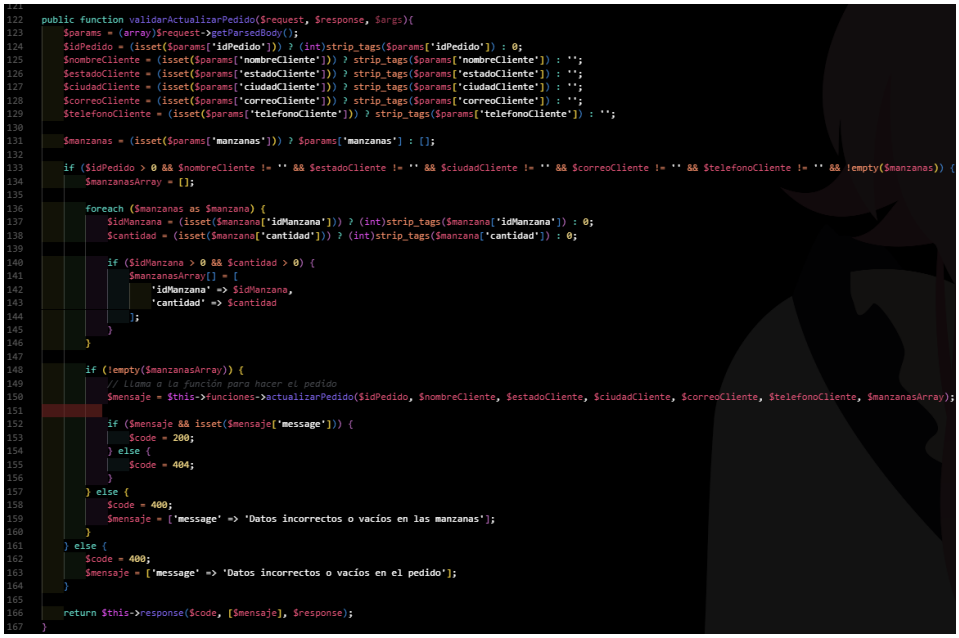


```
public function validarHacerPedido($request, $response, $args){
    $params = (array)$request->getParsedBody();

    $nombreCliente = (isset($params['nombreCliente'])) ? strip_tags($params['nombreCliente']) : '';
    $estadoCliente = (isset($params['estadoCliente'])) ? strip_tags($params['estadoCliente']) : '';
    $ciudadCliente = (isset($params['ciudadCliente'])) ? strip_tags($params['ciudadCliente']) : '';
    $correoCliente = (isset($params['correoCliente'])) ? strip_tags($params['correoCliente']) : '';
    $telefonoCliente = (isset($params['telefonoCliente'])) ? strip_tags($params['telefonoCliente']) : '';    "telefono": Unknown word.

    $manzanas = (isset($params['manzanas'])) ? $params['manzanas'] : [];

    if ($nombreCliente != '' && $estadoCliente != '' && $ciudadCliente != '' && $correoCliente != '' && $telefonoCliente != '' && !empty($manzanas)) {
        $manzanasArray = [];

        foreach ($manzanas as $manzana) {
            $idManzana = (isset($manzana['idManzana'])) ? (int)strip_tags($manzana['idManzana']) : 0;
            $cantidad = (isset($manzana['cantidad'])) ? (int)strip_tags($manzana['cantidad']) : 0;

            if ($idManzana > 0 && $cantidad > 0) {
                $manzanasArray[] = [
                    'idManzana' => $idManzana,
                    'cantidad' => $cantidad
                ];
            }
        }

        if (!empty($manzanasArray)) {
            // Llama a la función para hacer el pedido
            $mensaje = $this->funciones->hacerPedido($nombreCliente, $estadoCliente, $ciudadCliente, $correoCliente, $telefonoCliente, $manzanasArray);

            if ($mensaje && isset($mensaje['message'])) {
                $code = 200;
            } else {
                $code = 404;
            }
        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos en las manzanas'];
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos en el pedido'];
    }

    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 130 `validarHacerPedido` procesa datos de solicitud para realizar un pedido de manzanas. Verifica la validez de la información del cliente y las manzanas. Si todo es correcto, llama a la función `hacerPedido`. Devuelve un mensaje con el código de respuesta correspondiente. En caso de datos incorrectos, emite un código de error y un mensaje descriptivo.

Ilustración 131.- Pantalla de código del controlador para validar actualizar pedido.



```
public function validarActualizarPedido($request, $response, $args){
    $params = (array)$request->getParsedBody();
    $idPedido = (isset($params['idPedido'])) ? (int)strip_tags($params['idPedido']) : 0;
    $nombreCliente = (isset($params['nombreCliente'])) ? strip_tags($params['nombreCliente']) : '';
    $estadoCliente = (isset($params['estadoCliente'])) ? strip_tags($params['estadoCliente']) : '';
    $ciudadCliente = (isset($params['ciudadCliente'])) ? strip_tags($params['ciudadCliente']) : '';
    $correoCliente = (isset($params['correoCliente'])) ? strip_tags($params['correoCliente']) : '';
    $telefonoCliente = (isset($params['telefonoCliente'])) ? strip_tags($params['telefonoCliente']) : '';

    $manzanas = (isset($params['manzanas'])) ? $params['manzanas'] : [];

    if ($idPedido > 0 && $nombreCliente != '' && $estadoCliente != '' && $ciudadCliente != '' && $correoCliente != '' && $telefonoCliente != '' && !empty($manzanas)) {
        $manzanasArray = [];

        foreach ($manzanas as $manzana) {
            $idManzana = (isset($manzana['idManzana'])) ? (int)strip_tags($manzana['idManzana']) : 0;
            $cantidad = (isset($manzana['cantidad'])) ? (int)strip_tags($manzana['cantidad']) : 0;

            if ($idManzana > 0 && $cantidad > 0) {
                $manzanasArray[] = [
                    'idManzana' => $idManzana,
                    'cantidad' => $cantidad
                ];
            }
        }

        if (!empty($manzanasArray)) {
            // Llama a la función para hacer el pedido
            $mensaje = $this->funciones->actualizarPedido($idPedido, $nombreCliente, $estadoCliente, $ciudadCliente, $correoCliente, $telefonoCliente, $manzanasArray);

            if ($mensaje && isset($mensaje['message'])) {
                $code = 200;
            } else {
                $code = 404;
            }
        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos en las manzanas'];
        }
    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos en el pedido'];
    }

    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 131 `validarActualizarPedido` procesa la actualización de un pedido, validando los datos del cliente y las manzanas asociadas. Si la validación es exitosa, se llama a la función `actualizarPedido` y se devuelve un mensaje y código de respuesta correspondientes. En caso de datos incorrectos, se establece un código de error y se retorna el mensaje adecuado.

Ilustración 132.- Pantalla de código del controlador para validar eliminar un pedido.

```
public function validarEliminarPedido($request, $response, $args)
    {
        $params = (array)$request->getParsedBody();

        $idPedido = (isset($params['idPedido'])) ? (int)strip_tags($params['idPedido']) : 0;

        $mensaje = ['message' => ''];

        if ($idPedido > 0 ) {


                $mensaje = $this->funciones->eliminarPedido($idPedido);

                if ($mensaje) {
                    $code = 200;
                } else {
                    $code = 404;
                }

        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
        }

        // Retornamos la respuesta
        return $this->response($code, [$mensaje], $response);
    }
```

Fuente: Propia, 2023.

La función de la Ilustración 132 `validarEliminarPedido` valida y elimina un pedido según el identificador proporcionado en la solicitud. El resultado determina el código de respuesta HTTP y el mensaje de salida. Si tiene éxito, devuelve 200; de lo contrario, 404. En caso de datos incorrectos o vacíos, devuelve 400 con un mensaje correspondiente.

Ilustración 133.- Pantalla de código del controlador para validar liberar un pedido.

```php
public function validarLiberarPedido($request, $response, $args)
    {
        $params = (array)$request->getParsedBody();

        $idPedido = (isset($params['idPedido'])) ? (int)strip_tags($params['idPedido']) : 0;

        $mensaje = ['message' => ''];

        if ($idPedido > 0 ) {


                $mensaje = $this->funciones->liberarPedido($idPedido);

                if ($mensaje) {
                    $code = 200;
                } else {
                    $code = 404;
                }

        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
        }

        // Retornamos la respuesta
        return $this->response($code, [$mensaje], $response);
    }
```

Fuente: Propia, 2023.

La función de la Ilustración 133 valida y libera un pedido según el identificador proporcionado. Retorna un código 200 si tiene éxito, 404 si falla, y 400 en caso de datos incorrectos o vacíos. La respuesta se envía mediante el método `response`.

Ilustración 134.- Pantalla de código del controlador para validar listar pedido por id.

```php
public function validarListarPedidoId($request, $response, $args)
    {
        $params = (array)$request->getParsedBody();

        $idPedido = (isset($params['idPedido'])) ? (int)strip_tags($params['idPedido']) : 0;

        $mensaje = ['message' => ''];

        if ($idPedido > 0 ) {


                $mensaje = $this->funciones->obtenerPedidoPorId($idPedido);

                if ($mensaje) {
                    $code = 200;
                } else {
                    $code = 404;
                }

        } else {
            $code = 400;
            $mensaje = ['message' => 'Datos incorrectos o vacíos'];
        }

        // Retornamos la respuesta
        return $this->response($code, [$mensaje], $response);
    }
```

Fuente: Propia, 2023.

La función de la Ilustración 134 `validarListarPedidoId` toma una solicitud, extrae parámetros, valida y obtiene un pedido por su ID. Si el ID es válido y existe, devuelve el pedido con un código 200; si no, devuelve un código 404. En caso de datos incorrectos o vacíos, retorna un código 400 con un mensaje correspondiente. La respuesta se maneja a través del método `response`.

Ilustración 135.- Rutas de pedidos.

```php
//Grupo para Pedidos
$app->group('/Pedidos', function (Group $group) {
    //ruta para listar pedidos
    $group->post('/lista', pedidoController::class . ':validarListaPedidos');
    //ruta para listar un pedido por id
    $group->post('/litarId', pedidoController::class . ':validarListarPedidoId');
    //ruta para actualizar un pedido
    $group->post('/actualizar', pedidoController::class . ':validarActualizarPedido');
    //ruta para eliminar un pedido
    $group->post('/eliminar', pedidoController::class . ':validarEliminarPedido');
    //ruta para liberar un pedido y subirlo como venta
    $group->post('/liberar', pedidoController::class . ':validarLiberarPedido');

});
```

Fuente: Propia, 2023.

En la Ilustración 135 se define un grupo de rutas para operaciones relacionadas con pedidos. Incluye rutas para listar pedidos, listar un pedido por su ID, actualizar un pedido, eliminar un pedido y liberar un pedido para convertirlo en venta. Cada ruta está vinculada a un método específico en el controlador de pedidos para validar y procesar las solicitudes correspondientes.

Ilustración 136.- Pantalla de código de la función para ingresar un punto de venta.

```php
public function ingresarPuntoVenta(String $nombre, String $foto, String $latitud, String $longitud, int $estatus, String $horario)
{
    $sql2 = "INSERT INTO punto_venta (nombre, foto, latitud, longitud, estatus, horario) VALUES (?, ?, ?, ?, ?, ?)";

    $statement = $this->DB->Ejecutar_Seguro_UTF8($sql2, [$nombre, $foto, $latitud, $longitud, $estatus, $horario]);
    return ($statement == '200') ? true : false;
}
```

Fuente: Propia, 2023.

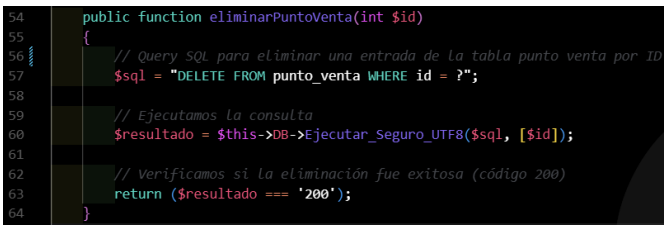
La función de la Ilustración 136 "ingresarPuntoVenta" recibe parámetros como nombre, foto, latitud, longitud, estatus y horario para insertar un nuevo punto de venta en la base de datos. Utiliza una consulta SQL segura para la inserción, ejecuta la consulta con los valores proporcionados y devuelve true si la operación fue exitosa, o false en caso contrario.

Ilustración 137.- Pantalla de código de la función para actualizar un punto de venta.

Fuente: Propia, 2023.

La función de la Ilustración 137 "actualizarPuntoVenta" actualiza datos en la base de datos para un punto de venta específico. Retorna verdadero si la actualización es exitosa, falso en caso contrario.

Ilustración 138.- Pantalla de código de la función para eliminar un punto de venta.



Fuente: Propia, 2023.

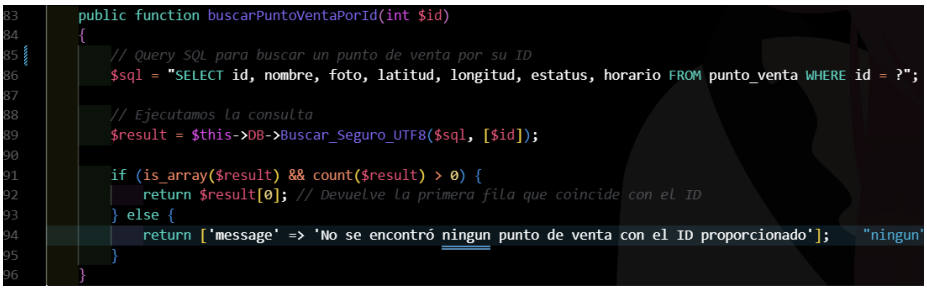
La función de la Ilustración 138 "eliminarPuntoVenta" elimina una entrada en la tabla "punto_venta" mediante una consulta SQL que utiliza el ID proporcionado como parámetro. Después de ejecutar la consulta, verifica si la eliminación fue exitosa basándose en el código de resultado, devolviendo true si el código es 200.

Ilustración 139.- Pantalla de código de la función para listar puntos de venta.

Fuente: Propia, 2023.

La función de la Ilustración 139 `listaPuntosVenta` realiza una consulta SQL para seleccionar los campos id, nombre, foto, latitud, longitud, estatus y horario de la tabla punto_venta. Luego, utiliza un método de búsqueda en la base de datos y devuelve el resultado de la consulta. Si la consulta devuelve un array con datos, se retorna ese array; de lo contrario, se retorna un array con el mensaje de error.

Ilustración 140.- Pantalla de código de la función para listar por id un punto de venta.



```
public function buscarPuntoVentaPorId(int $id)
{
    // Query SQL para buscar un punto de venta por su ID
    $sql = "SELECT id, nombre, foto, latitud, longitud, estatus, horario FROM punto_venta WHERE id = ?";

    // Ejecutamos la consulta
    $result = $this->DB->Buscar_Seguro_UTF8($sql, [$id]);

    if (is_array($result) && count($result) > 0) {
        return $result[0]; // Devuelve la primera fila que coincide con el ID
    } else {
        return ['message' => 'No se encontró ningun punto de venta con el ID proporcionado'];    "ningun"
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 140 busca un punto de venta en la base de datos según su ID. La función ejecuta la consulta y verifica si se encontraron resultados. Si hay coincidencias, devuelve la primera fila correspondiente al ID; de lo contrario, devuelve un mensaje indicando que no se encontró ningún punto de venta con el ID proporcionado.

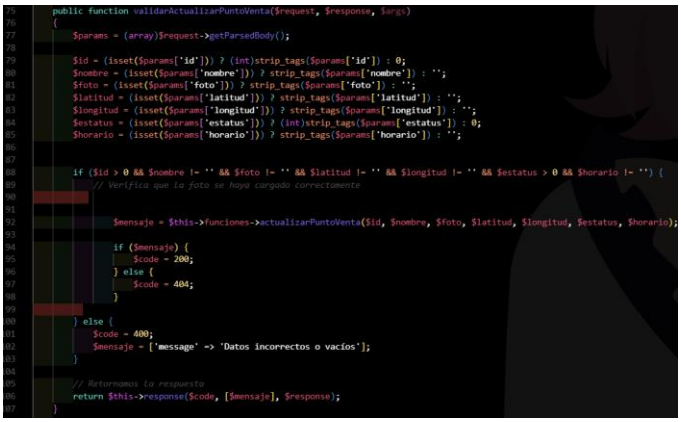
Ilustración 141.- Pantalla de código del controlador para validar ingresar un punto de venta.

```
public function validarIngresarPuntoVenta($request, $response, $args)
{
    $params = (array)$request->getParsedBody();

    $nombre = (isset($params['nombre'])) ? strip_tags($params['nombre']) : '';
    $foto = (isset($params['foto'])) ? strip_tags($params['foto']) : '';
    $latitud = (isset($params['latitud'])) ? strip_tags($params['latitud']) : '';
    $longitud = (isset($params['longitud'])) ? strip_tags($params['longitud']) : '';
    $estatus = (isset($params['estatus'])) ? (int)strip_tags($params['estatus']) : 0;
    $horario = (isset($params['horario'])) ? strip_tags($params['horario']) : '';


    if ($nombre != '' &&  $foto != '' && $latitud != '' && $longitud != '' && $estatus > 0 && $horario != '' ) {
            $mensaje = $this->funciones->ingresarPuntoVenta($nombre, $foto, $latitud, $longitud, $estatus, $horario);

            if ($mensaje) {
                $code = 200;
            } else {
                $code = 404;
            }

    } else {
        $code = 400;
        $mensaje = ['message' => 'Datos incorrectos o vacíos'];
    }

    // Retornamos la respuesta
    return $this->response($code, [$mensaje], $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 141 valida y registra un punto de venta con los datos proporcionados en una solicitud HTTP. Devuelve un código 200 si tiene éxito, 404 si falla al registrar, y 400 si los datos son incorrectos o incompletos.

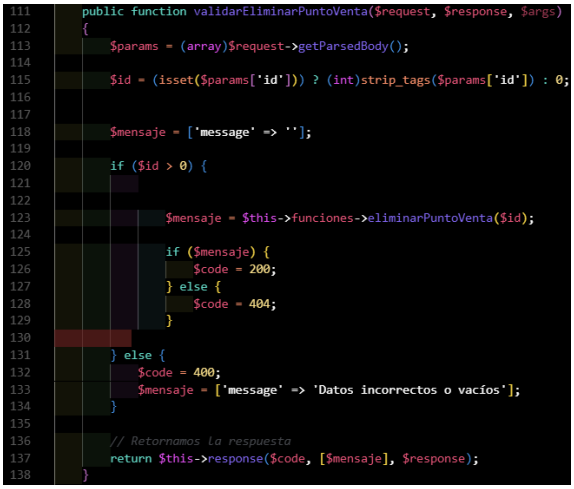
Ilustración 142.- Pantalla de código del controlador para validar actualizar un punto de venta.



Fuente: Propia, 2023.

La función de la Ilustración 142 `validarActualizarPuntoVenta` procesa parámetros de solicitud, verifica su validez y actualiza un punto de venta. Retorna códigos de respuesta según el resultado (200 o 404). Si hay datos incorrectos o vacíos, devuelve un código 400 con un mensaje adecuado.

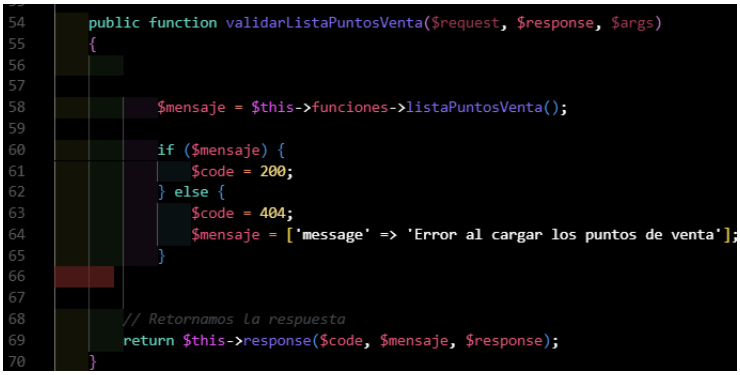
Ilustración 143.- Pantalla de código del controlador para validar eliminar punto de venta.



Fuente: Propia, 2023.

La función de la Ilustración 143 "validarEliminarPuntoVenta" recibe un ID, intenta eliminar el punto de venta y devuelve un código 200 en caso de éxito, 404 si no encuentra el punto de venta, y 400 por datos incorrectos o vacíos. La respuesta se envía al cliente mediante "response".

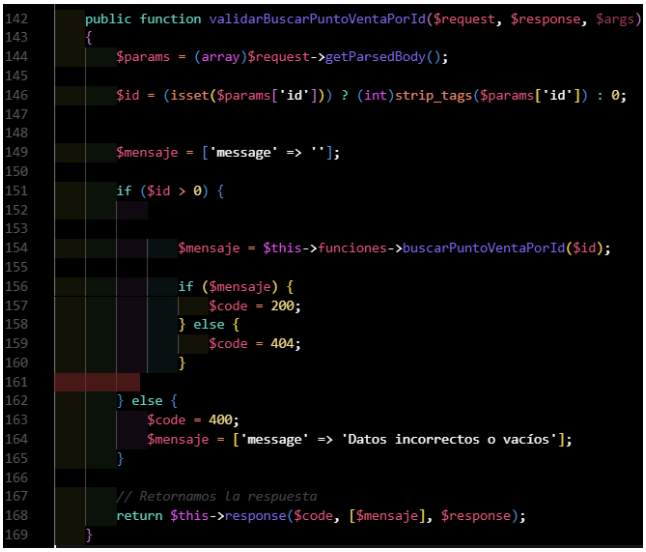
Ilustración 144.- Pantalla de código del controlador para validar listar puntos de venta.



Fuente: Propia, 2023.

La función de la Ilustración 144 "validarListaPuntosVenta" verifica la existencia de puntos de venta. Si se encuentra la lista, devuelve un código de estado 200; de lo contrario, devuelve un código de estado 404 con un mensaje de error. La respuesta final se gestiona a través de la función "response".

Ilustración 145.- Pantalla de código del controlador para validar buscar por id una actividad.



Fuente: Propia, 2023.

La función de la Ilustración 145 valida y busca un punto de venta por su ID. Retorna un código de respuesta (200 si encuentra, 404 si no) y un mensaje. En caso de datos incorrectos, retorna un código 400 con un mensaje correspondiente.

Ilustración 146.- Rutas de los puntos de venta.

```php
//Grupo para puntos de venta
$app->group('/PuntoVenta', function (Group $group) {
    //ruta para ingresar un punto de venta
    $group->post('/registrar', puntoVentaController::class . ':validarIngresarPuntoVenta');
    //ruta para actualizar un punto de venta
    $group->post('/actualizar', puntoVentaController::class . ':validarActualizarPuntoVenta');
    //ruta para eliminar un punto de venta
    $group->post('/eliminar', puntoVentaController::class . ':validarEliminarPuntoVenta');
});
```

Fuente: Propia, 2023.

En la Ilustración 146 se define un grupo para gestionar puntos de venta con rutas para registrar, actualizar y eliminar puntos de venta. Cada ruta utiliza el controlador "puntoVentaController" con funciones de validación asociadas.

Ilustración 147.- Pantalla de código de la función para listar ventas.

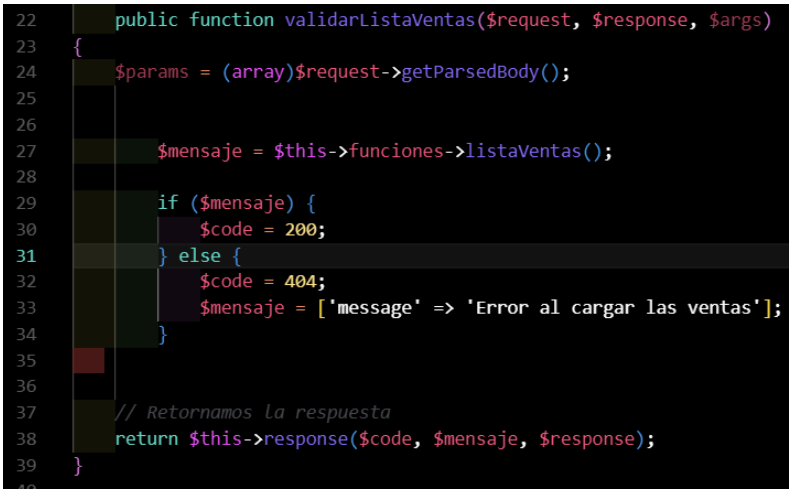
```php
public function listaVentas(){
    try {
        $sql = "SELECT v.id AS venta_id, v.fechaLiberado, v.total, v.nombreCliente, v.estadoCliente, v.ciudadCliente, v.correoCliente, v.telefonoCliente,
                m.id AS manzana_id, m.nombre AS manzana_nombre, vm.cantidad AS cantidad_manzana, vm.subtotal AS subtotal_manzana
            FROM venta AS v
            LEFT JOIN venta_manzana AS vm ON v.id = vm.idVenta
            LEFT JOIN manzana AS m ON vm.idManzana = m.id
            ORDER BY v.fechaLiberado";

        $result = $this->DB->buscar($sql, []);

        $ventas = [];
        $currentVenta = null;

        foreach ($result as $row) {
            $ventaId = $row['venta_id'];

            if ($currentVenta === null || $ventaId !== $currentVenta['id']) {
                // Nueva venta
                $currentVenta = [
                    'id' => $ventaId,
                    'fechaLiberado' => $row['fechaLiberado'],
                    'total' => $row['total'],
                    'nombreCliente' => $row['nombreCliente'],
                    'estadoCliente' => $row['estadoCliente'],
                    'ciudadCliente' => $row['ciudadCliente'],
                    'correoCliente' => $row['correoCliente'],
                    'telefonoCliente' => $row['telefonoCliente'],    "telefono": Unknown word.
                    'manzanas' => [],
                ];
                $ventas[] = $currentVenta;
            }

            // Agregar la manzana y la cantidad a la venta actual
            $currentVenta['manzanas'][] = [
                'id' => $row['manzana_id'],
                'nombre' => $row['manzana_nombre'],
                'cantidad' => $row['cantidad_manzana'],
                'subtotal' => $row['subtotal_manzana'],
            ];
        }

        return $ventas;
    } catch (Exception $e) {
        return ['error' => $e->getMessage()];
    }
}
```

Fuente: Propia, 2023.

La función de la Ilustración 147 `listaVentas` realiza una consulta SQL para obtener información detallada sobre ventas, incluyendo datos de clientes y manzanas asociadas. Utiliza JOINs para unir las tablas relevantes y ordena los resultados por la fecha de liberación de las ventas. Posteriormente, organiza la información en un

formato estructurado, donde cada venta tiene sus datos generales y una lista de manzanas con detalles específicos. En caso de algún error durante el proceso, se devuelve un mensaje de error.

Ilustración 148.- Pantalla de código del controlador para validar listar ventas.

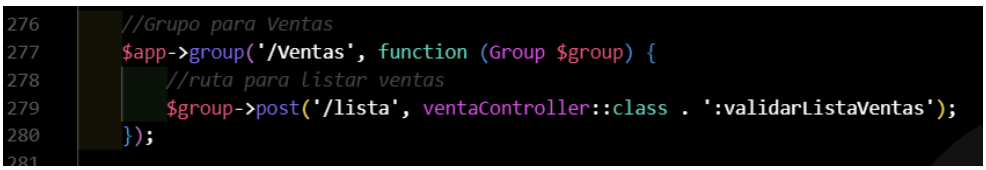


```php
    public function validarListaVentas($request, $response, $args)
{
    $params = (array)$request->getParsedBody();


        $mensaje = $this->funciones->listaVentas();

        if ($mensaje) {
            $code = 200;
        } else {
            $code = 404;
            $mensaje = ['message' => 'Error al cargar las ventas'];
        }


    // Retornamos la respuesta
    return $this->response($code, $mensaje, $response);
}
```

Fuente: Propia, 2023.

La función de la Ilustración 148 "validarListaVentas" recibe parámetros de solicitud, obtiene una lista de ventas utilizando el método "listaVentas" de la clase "funciones", y determina el código de respuesta y mensaje en función de si la lista se obtiene correctamente o no. El resultado se devuelve como respuesta.

Ilustración 149.- Rutas de ventas.



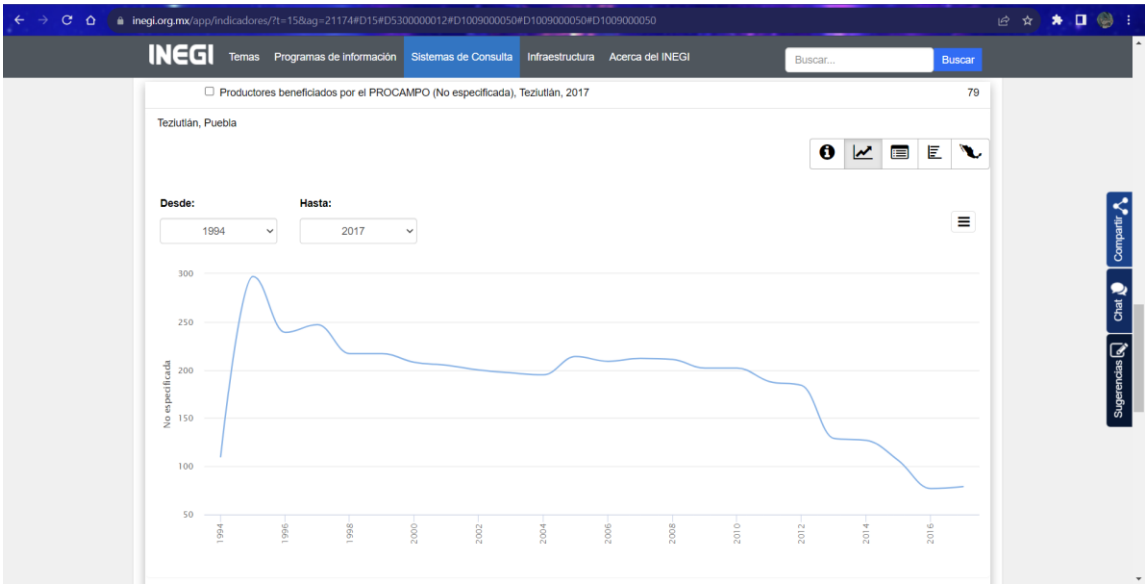
```php
//Grupo para Ventas
$app->group('/Ventas', function (Group $group) {
    //ruta para listar ventas
    $group->post('/lista', ventaController::class . ':validarListaVentas');
});
```

Fuente: Propia, 2023.

En la Ilustración 149 se define una ruta '/Ventas/lista' que utiliza el método POST para validar la lista de ventas a través del controlador.

3.5 Población y muestra:

Ilustración 150.- Instituto Nacional de Estadística y Geografía (INEGI).



Fuente: Instituto Nacional de Estadística y Geografía (INEGI), 2017.

En la Ilustración 150 se puede apreciar la población de productores beneficiados por el PROCAMPO en el año 2017 en la región de Teziutlán Puebla según el INEGI: 79.

Ilustración 151.- Fórmula general para calcular el tamaño de la muestra en una población finita con un muestreo aleatorio simple.

$$n = \frac{N * Z^2 * p * q}{d^2 * (N-1) + Z^2 * p * q}$$

Fuente: Propia, 2023.

Tabla 2.- Variables para calcular el tamaño de la muestra.

N=	79	
Z=	1.96	
p=	0.9	
q=	0.1	
d=	0.05	
Muestra n=	27.313776	50.51147308
	0.540744	

Fuente: Propia, 2023.

En la Tabla 2 se puede apreciar los valores con los que aplica la fórmula de la Ilustración 151, dicho resultado es de 50.51147308, redondeando este valor se obtiene 50 el cual es el tamaño de la muestra para este caso, dicha cantidad es el número de productores entrevistados.

3.6 Recolección de datos

3.6.1 Selección del instrumento

Esta investigación se centra en la optimización de la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán (CPMT) a través del desarrollo de una aplicación web. El propósito es agilizar el control de inventarios para garantizar una oferta adecuada que satisfaga la creciente demanda. Como parte de este proyecto, se ha elegido utilizar una encuesta de satisfacción con ponderaciones diferenciadas. Esta selección tiene como objetivo evaluar no solo la satisfacción de los usuarios, sino también aspectos cruciales como el acoplamiento, la capacitación y la usabilidad de la aplicación entregada. Al asignar diferentes valores a estas variables, se busca obtener una comprensión detallada de la experiencia del usuario, proporcionando información valiosa para la mejora continua de la aplicación web desarrollada.

Tabla 3.- Tabla de ponderaciones.

Ponderaciones	
Inciso	Valor
a) Totalmente de acuerdo.	3 puntos.
b) En gran medida de acuerdo.	2 puntos.
c) En cierta medida en desacuerdo.	1 punto.
d) En desacuerdo.	0 puntos.

Fuente: Propia, 2023.

Cuestionario

1.- ¿La aplicación web mejora la agilidad del control de inventarios?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

2.- ¿La aplicación web mejora la comunicación y coordinación del inventario entre los productores de manzanas dentro del Colectivo?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

3.- ¿La aplicación web simplifica los procesos de gestión de stock en comparación con métodos anteriores?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

4.- ¿La aplicación web logra un equilibrio efectivo entre la disponibilidad de productos y la gestión de inventarios?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

5.- ¿La aplicación web genera una reducción de errores en el registro de datos de los pedidos?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

6.- ¿La aplicación web incrementa la transparencia en la gestión de inventarios dentro del Colectivo de Productores de Manzana Teziutlán?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

7.- ¿La aplicación web, con su sistema de manejo de inventario y la información incorporada, puede lograr una mejora sustancial en el cumplimiento de pedidos?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

8.- ¿La aplicación web cumple con sus expectativas en términos de mejorar la visibilidad y accesibilidad de la información de gestión del inventario?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

9.- ¿La aplicación web contribuye significativamente al manejo y control de inventarios en el Colectivo de Productores de Manzana Teziutlán?

a) Totalmente de acuerdo.

b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

10.- ¿La capacitación de la aplicación web fue efectiva para el manejo y uso adecuado de la misma?

a) Totalmente de acuerdo.

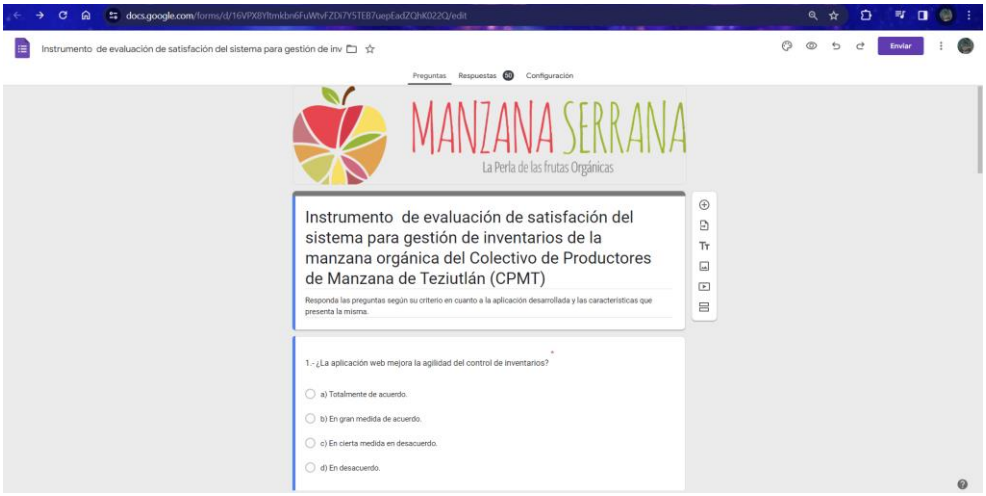
b) En gran medida de acuerdo.

c) En cierta medida en desacuerdo.

d) En desacuerdo.

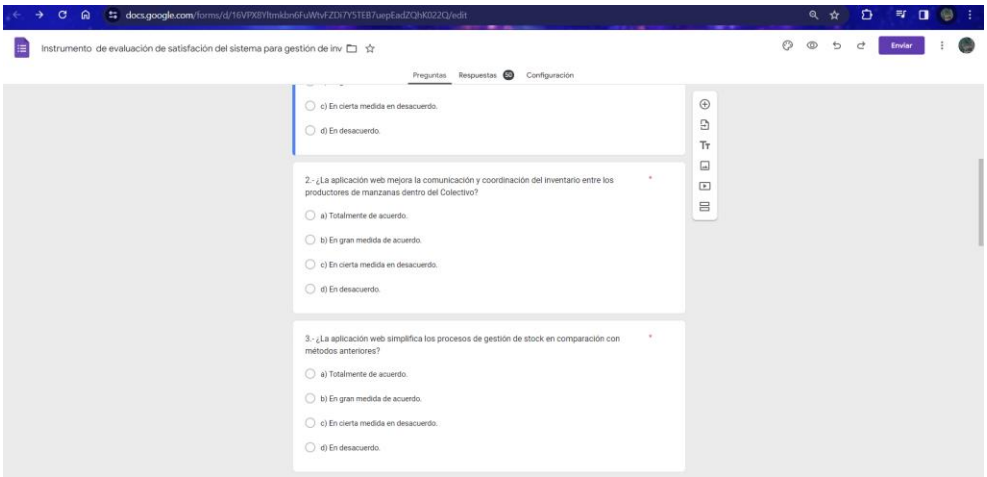
3.6.2 Aplicación del instrumento

Ilustración 152.- Aplicación del instrumento de evaluación parte 1.



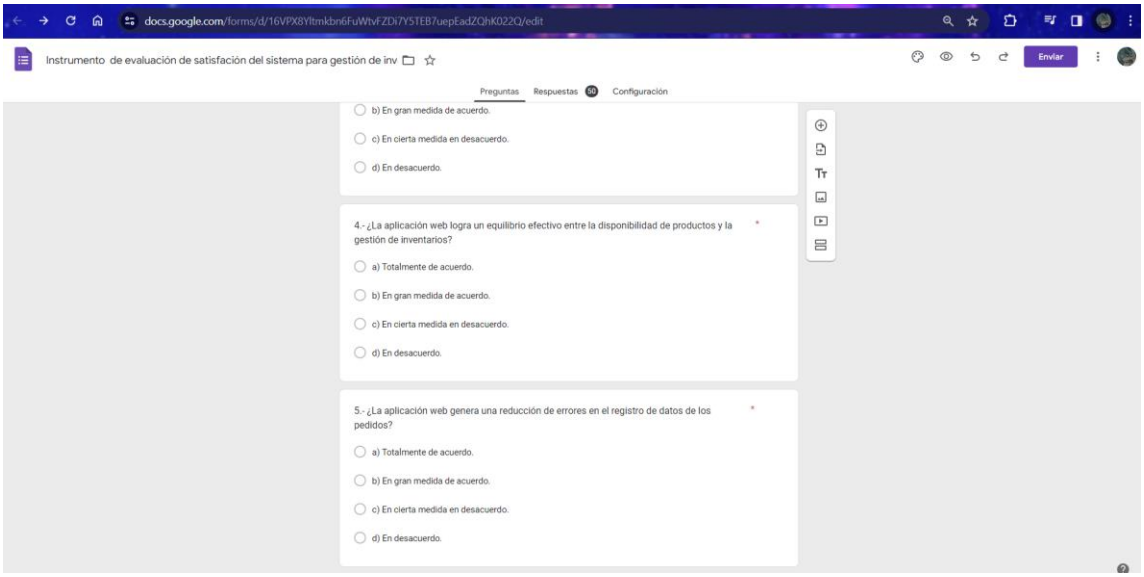
Fuente: Google Forms, 2023.

Ilustración 153.- Aplicación del instrumento de evaluación parte 2.



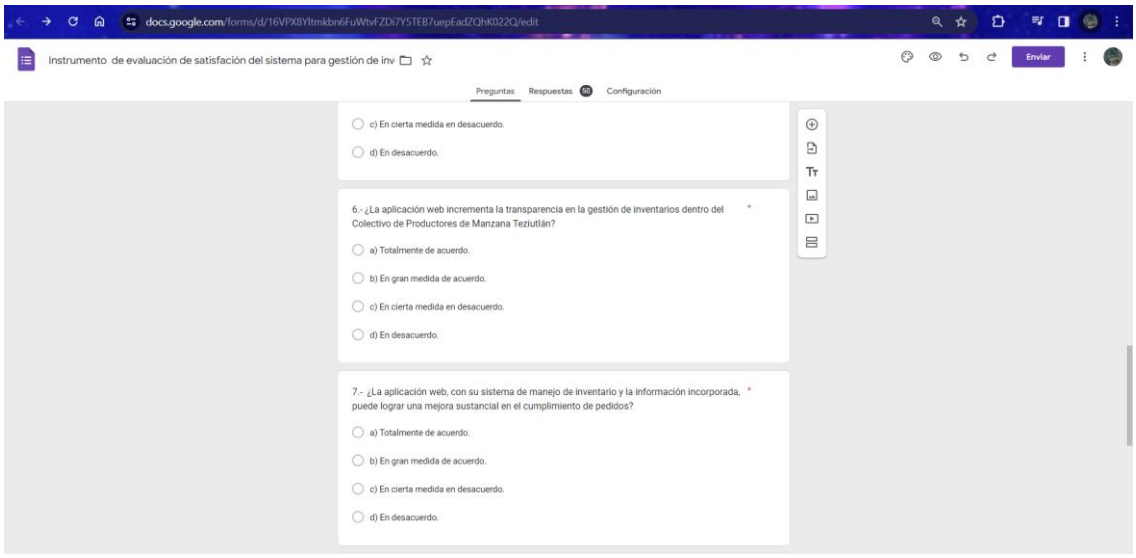
Fuente: Google Forms, 2023.

Ilustración 154.- Aplicación del instrumento de evaluación parte 3.



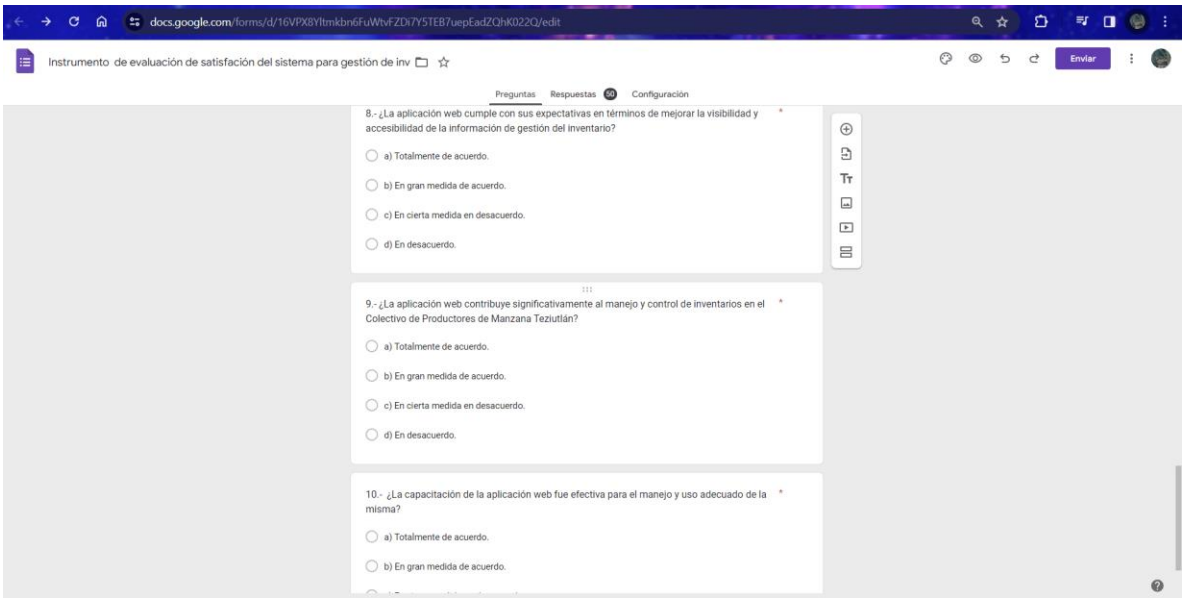
Fuente: Google Forms, 2023.

Ilustración 155.- Aplicación del instrumento de evaluación parte 4.



Fuente: Google Forms, 2023.

Ilustración 156.- Aplicación del instrumento de evaluación parte 5.

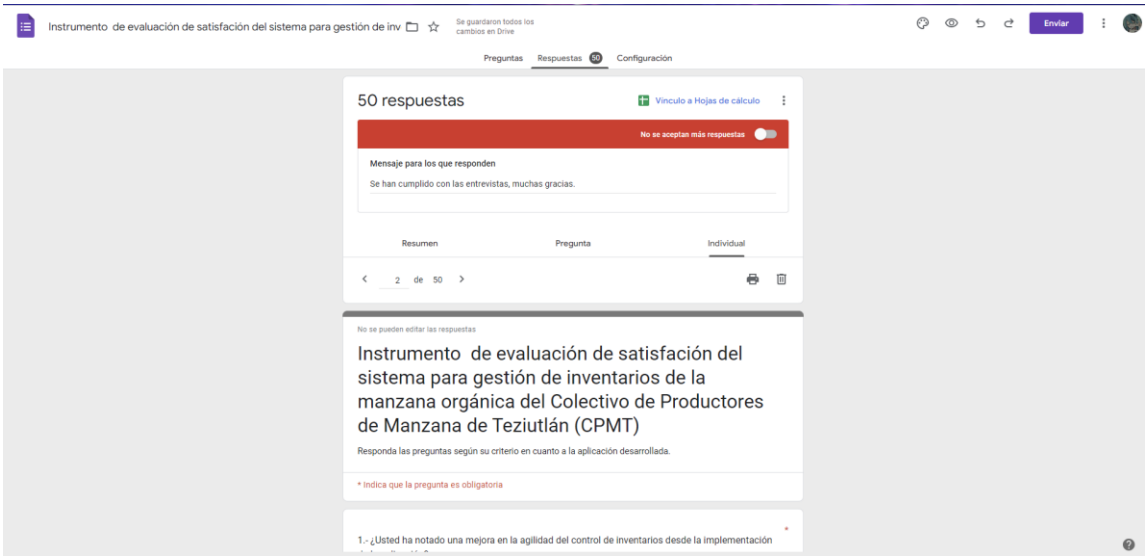


Fuente: Google Forms, 2023.

En la Ilustración 152, Ilustración 153, Ilustración 154, Ilustración 155, e Ilustración 156 se muestra la aplicación del instrumento por medio de Google Forms, el instrumento se compartió con la muestra seleccionada para verificar la hipótesis planteada. La elección de Google Forms para la aplicación del instrumento de evaluación se fundamenta en su accesibilidad, facilidad de uso y capacidad para recopilar datos de manera eficiente. Google Forms proporciona una plataforma en línea intuitiva que permite a los participantes responder fácilmente a las preguntas del instrumento desde cualquier dispositivo con conexión a Internet. Además, la herramienta ofrece una variedad de opciones de respuesta y la capacidad de organizar y analizar los datos de manera automática. La utilización de Google Forms simplifica el proceso de recopilación de información, garantizando una experiencia fluida para los participantes y facilitando la posterior interpretación de los resultados para una evaluación exhaustiva del impacto de la aplicación web en la gestión de inventarios de manzana orgánica.

3.6.3 Preparación de datos

Ilustración 157.- Aplicación del instrumento.



Fuente: Google Forms, 2023.

En la Ilustración 157 se aprecia como se han recolectado satisfactoriamente los datos de la muestra para obtener los resultados de satisfacción en cuanto a si el programa optimizó la gestión de inventarios por parte de los productores del colectivo.

Tabla 4.- Respuestas del instrumento.

Productor	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4	Pregunta 5	Pregunta 6	Pregunta 7	Pregunta 8	Pregunta 9	Pregunta 10
Productor 1	3	3	3	2	3	2	3	3	3	3
Productor 2	3	2	2	3	3	3	3	1	3	3
Productor 3	3	3	3	3	3	3	3	3	2	3
Productor 4	3	3	3	3	3	3	2	2	2	3
Productor 5	2	3	3	2	3	3	3	3	3	3
Productor 6	3	2	2	3	3	3	3	2	3	3
Productor 7	3	3	3	3	2	3	3	3	2	3
Productor 8	3	3	3	2	2	3	2	3	2	3
Productor 9	3	3	3	3	3	3	3	3	3	3
Productor 10	3	3	3	2	3	2	3	3	3	3

Productor 11	3	3	3	2	3	2	3	3	3	2
Productor 12	2	2	3	3	3	3	2	3	3	3
Productor 13	3	3	3	3	3	2	2	3	3	1
Productor 14	2	3	3	2	2	3	3	3	2	0
Productor 15	2	3	2	3	3	2	3	3	2	3
Productor 16	3	3	2	2	2	3	3	3	3	3
Productor 17	3	3	2	3	3	3	3	1	2	3
Productor 18	2	2	3	3	3	3	2	2	2	3
Productor 19	3	3	3	3	3	2	2	3	3	3
Productor 20	2	2	3	3	3	3	3	3	2	3
Productor 21	3	0	1	2	2	2	3	3	3	3
Productor 22	2	2	2	3	3	3	3	3	3	3
Productor 23	3	3	2	2	2	3	3	3	3	3
Productor 24	3	3	3	3	3	3	3	2	3	2
Productor 25	3	3	3	2	3	3	2	3	3	3
Productor 26	3	3	3	2	2	3	3	3	3	3
Productor 27	3	2	3	2	3	2	2	3	3	3
Productor 28	2	2	3	3	3	3	3	2	3	2
Productor 29	3	3	2	2	2	2	2	2	2	2
Productor 30	3	3	2	3	3	2	2	3	3	2
Productor 31	1	3	3	2	1	3	3	2	2	2
Productor 32	1	3	2	2	3	3	3	3	3	3
Productor 33	2	2	2	2	3	3	3	3	3	2
Productor 34	2	3	2	3	3	2	2	3	3	3
Productor 35	3	3	3	3	3	3	2	2	2	3
Productor 36	3	3	2	3	2	3	2	3	3	3
Productor 37	2	2	3	3	2	1	3	2	2	2
Productor 38	3	3	2	3	2	2	3	2	2	2
Productor 39	3	2	3	3	2	3	2	3	3	2
Productor 40	3	3	2	3	2	3	2	3	2	2
Productor 41	1	2	1	2	2	2	2	1	2	1
Productor 42	3	3	3	3	3	3	3	3	3	3
Productor 43	3	3	3	3	3	3	3	3	3	3

Productor 44	3	3	3	3	3	3	2	3	3	3
Productor 45	3	3	3	3	3	3	3	3	3	3
Productor 46	3	3	3	3	3	3	3	3	3	3
Productor 47	2	2	3	2	2	2	3	3	3	2
Productor 48	3	3	3	2	2	3	3	2	3	2
Productor 49	3	3	3	3	2	2	3	2	3	3
Productor 50	3	3	3	3	3	3	3	3	3	3

Fuente: Propia, 2023.

En la Tabla 4 se muestran los resultados registrados de las entrevistas que se hicieron en Google Forms, dicha herramienta exporto los datos a una tabla cuyos valores de respuesta fueron sustituidos por su valor correspondiente en la Tabla 3, esto con el fin de obtener un mejor panorama por pregunta y como es que respondió cada productor.

3.7 Análisis de datos

Tabla 5.- Valores de los datos.

Pregunta	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4	Pregunta 5	Pregunta 6	Pregunta 7	Pregunta 8	Pregunta 9	Pregunta 10
Total	132	134	131	131	131	133	133	132	134	130
Porcentaje	88.00%	89.33%	87.33%	87.33%	87.33%	88.67%	88.67%	88.00%	89.33%	86.67%

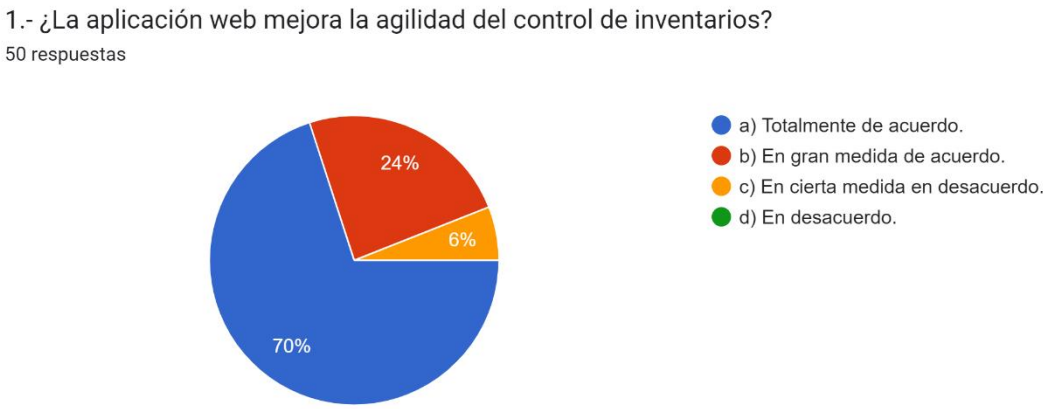
Fuente: Propia, 2023.

En la Tabla 5 se muestra el total correspondiente a la sumatoria de cada pregunta según lo contestado por los productores y su ponderación asignada por respuesta, se muestra el total por pregunta así como el porcentaje de ese valor, considerando que el 100% es 150, ya que según las ponderaciones este sería el valor máximo por pregunta, se pueden observar resultados de porcentajes muy buenos considerando que la muestra para el instrumento fue de 50 productores, la pregunta 1 obtuvo 132 puntos que representan un 88% de aprobación en cuanto a si se nota una mejora en la agilidad del control de inventarios con la aplicación, este porcentaje es sumamente bueno, ya que se necesita un programa ágil para aspectos críticos como

el manejo de inventario. La pregunta 2 obtuvo 134 puntos que representan el 89.33% referente a la aprobación que le dan los productores en sí la aplicación mejora la comunicación y coordinación entre los productores del colectivo, es un porcentaje prometedor debido a la necesidad de comunicación entre los integrantes del colectivo para tener una mejor operatividad. En cuanto a la pregunta 3 se tiene un total de 131 puntos que es el 87.33%, donde ese porcentaje considera que la aplicación web con su funcionamiento simplifica los procesos de gestión de stock en comparación con métodos anteriores, este porcentaje es de igual manera prometedora, ya que se está implementando una solución tecnológica al manejo de inventario del colectivo, con lo cual se puede percatar que el impacto de esta nueva manera de gestionar el stock es sumamente positivo. En la pregunta 4 obtuvo 131 puntos que también representan el 87.33% denotando que a los productores han considerado que la aplicación logra un equilibrio efectivo entre la disponibilidad de productos y la gestión de inventarios que puede ayudar a garantizar una oferta continua y satisfacer la demanda de manera eficaz, las validaciones mayormente positivas a esta pregunta indican que la manera en cómo está pensada la aplicación en donde el administrador elige si surte o no el pedido basándose en el stock que se tiene, puede generar una oferta continua y con ello que la demanda aumente. En la pregunta 5 se obtienen valores similares con un total de 131 puntos que de igual manera a los casos anteriores representa el 87.33%, con ese porcentaje se observa que los productores consideran que la aplicación web genera una reducción de errores en el registro de datos de los pedidos, denotan que la aplicación si tiene esta característica tan crítica para sus operaciones. En la pregunta 6 se obtuvieron 133 puntos equivalentes al 88.67%, en donde se deja ver que, para la mayor parte de los productores del colectivo, la implementación de la aplicación web mejora la transparencia en la gestión de inventarios, esto es esencial para que se tenga un mejor manejo y registro del stock y se eviten perdidas, además, es de utilidad para dar cuentas a los productores por si surgen dudas. En la pregunta 7 se obtuvieron 133 puntos equivalentes de igual manera al 88.67% de aprobación por parte de los productores en cuanto a si la aplicación web con su sistema de manejo de inventario

y la información incorporada, puede lograr una mejora sustancial en el cumplimiento de pedidos, los números indican que los productores consideran que la aplicación es de gran utilidad para manejar los pedidos y los movimientos de inventario que estos conllevan, esto es importante debido a las implicaciones directas que tienen los pedidos con el inventario. En la pregunta 8 se obtuvo un total de 132 puntos que equivale a un 88% de aprobación de los productores en cuanto a uno de los puntos más importantes de la aplicación, dicha pregunta es acerca de si la aplicación cumple las expectativas en términos de tener la capacidad de mejorar la visibilidad y accesibilidad de la información de gestión del inventario, es importante que la aplicación sea una mejora y de resultados más claro en cuanto a la información del stock y otros aspectos del colectivo, por ello es de suma importancia este punto. En la pregunta 9 se obtuvieron 134 puntos equivalentes al 89.33%, dicho puntaje es la aprobación que le dieron los productores en cuanto a uno de los puntos que tiene mucho que ver con la hipótesis y la pregunta de investigación, es lo referente a si ellos consideran que la aplicación web contribuye significativamente al manejo y control de inventarios en el colectivo, es el punto más importante debido a que ese es el objetivo general de la aplicación a grandes rasgos, y la principal función de la misma, se puede observar que es muy positiva la aceptación de la aplicación y que para los productores la aplicación es una gran ayuda que podrán usar para muchas cuestiones partiendo del enfoque de inventarios que tiene, y este mismo es una mejora dentro del colectivo, no solo para darse a conocer, sino en tener un control de inventario más robusto generando mejoras en las actividades productivas del colectivo. En la pregunta 10 se obtuvieron 130 puntos equivalentes al 86.67%, este porcentaje es referente a la aprobación por parte de los productores en la capacitación recibida para el manejo adecuado de la aplicación, el resultado es buena debido a que se tiene una aprobación de la mayoría de productores, dando a entender que se les instruyó en el uso correcto de la aplicación de forma adecuada.

Gráfico 1.- Porcentaje de respuestas de la pregunta 1.



Fuente: Google Forms, 2023.

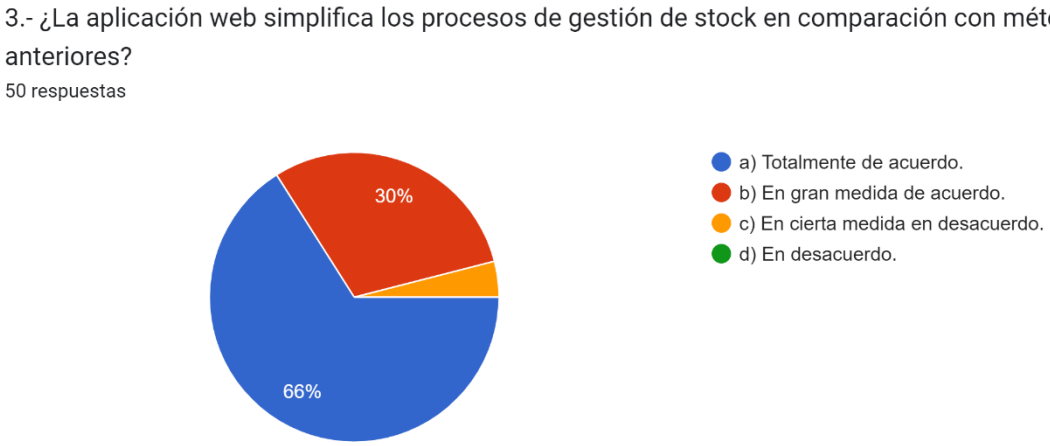
En el Gráfico 1 se muestra los resultados de la pregunta 1, los datos arrojan que el 70% de los productores consideran en su totalidad que la aplicación agiliza el control de inventarios con la aplicación, mientras que el 24% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 6% ven que en cierta medida la aplicación no cumple en su totalidad con la agilización de inventarios, son resultados buenos, su importancia radica en que proporciona una evaluación directa de la eficacia de la solución tecnológica que se desarrolló. Las respuestas a esta pregunta permitieron mostrar no solo la eficiencia operativa con la que cuenta la aplicación, sino también validar la capacidad que tiene la misma en términos de optimización y agilidad en el control de inventarios, lo cual es fundamental para la toma de decisiones estratégicas y la mejora continua del sistema.

Gráfico 2.- Porcentaje de respuestas de la pregunta 2.

Fuente: Google Forms, 2023.

En el Gráfico 2 se muestra los resultados de la pregunta 2, los datos arrojan que el 72% de los productores consideran en su totalidad que la aplicación web mejora la comunicación y coordinación entre los productores, mientras que el 26% ven que en gran medida no se tiene la capacidad de lograr ese objetivo, y solo una pequeña parte correspondiente al 2% están en desacuerdo en cuanto a la mejora en la comunicación, con estos resultados positivos se puede decir que queda destacada la eficacia de la aplicación en fortalecer los lazos y la colaboración entre los miembros del colectivo. Una mejora significativa en la comunicación y coordinación puede traducirse en beneficios como una mayor eficiencia en las operaciones, la optimización de recursos y la capacidad de respuesta más rápida a los desafíos. Estos resultados positivos no solo validarían el valor de la aplicación en términos prácticos, sino que también respaldarían la idea de que la tecnología puede desempeñar un papel clave en el fomento de la sinergia y el éxito colectivo dentro de la comunidad de productores de manzanas.

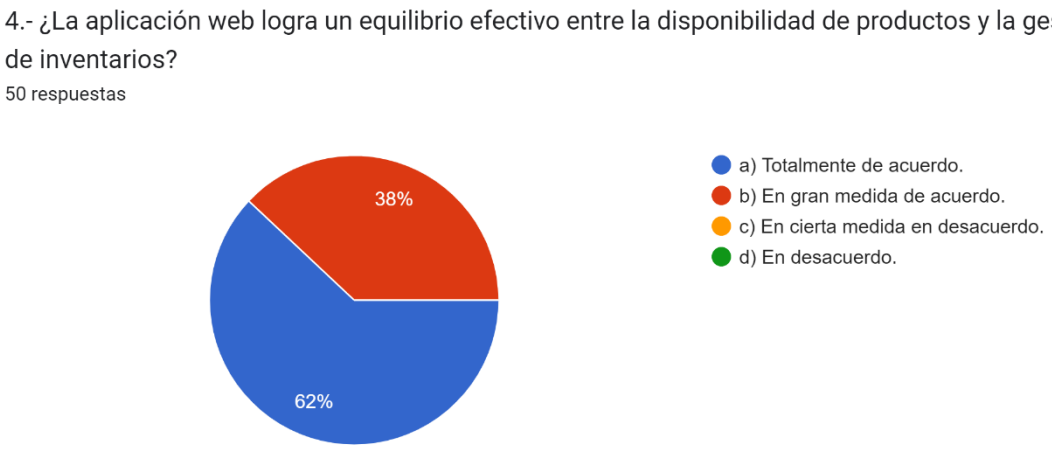
Gráfico 3.- Porcentaje de respuestas de la pregunta 3.



Fuente: Google Forms, 2023.

En el Gráfico 3 se muestra los resultados de la pregunta 3, los datos arrojan que el 66% de los productores consideran en su totalidad que la aplicación tiene la característica de simplificar los procesos de gestión del inventario en comparación con métodos anteriores, mientras que el 30% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 4% están en desacuerdo en cuanto a la simplificación de los procesos de manejo de stock. Un consenso afirmativo resalta la capacidad de la aplicación para mejorar la eficiencia operativa y reducir la complejidad asociada con los métodos de gestión de stock previos. Esta percepción positiva no solo sugiere una posible transición exitosa hacia la plataforma web, sino que también apunta a la optimización de los procesos internos de la gestión de inventario, lo que puede traducirse en ahorros de tiempo, recursos y una toma de decisiones más informada.

Gráfico 4.- Porcentaje de respuestas de la pregunta 4.



Fuente: Google Forms, 2023.

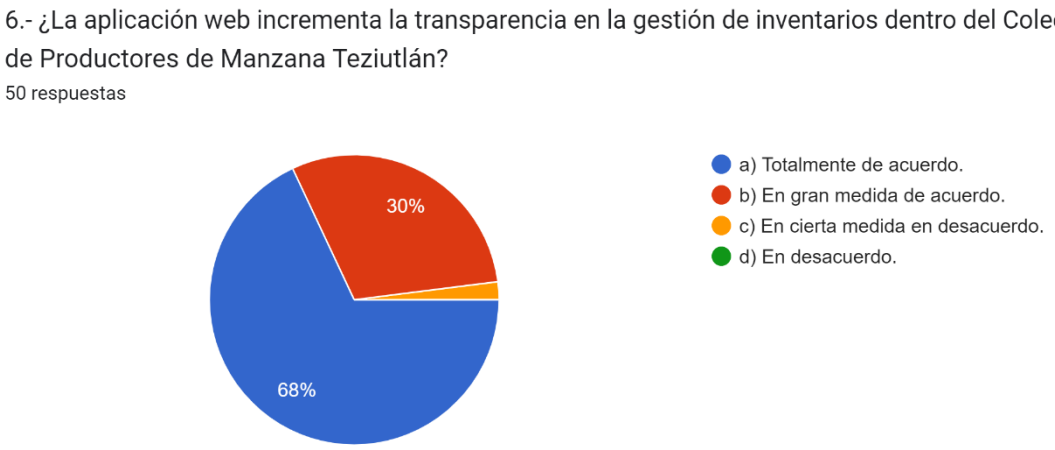
En el Gráfico 4 se muestra los resultados de la pregunta 4, los datos arrojan que el 66% de los productores consideran que la aplicación muestra un equilibrio entre la disponibilidad de productos y la gestión de inventarios para garantizar una oferta continua al consumidor, mientras que el 30% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 4% están en cierta medida en desacuerdo en el logro de este equilibrio por parte de la aplicación, con estos resultados positivos se indica que la aplicación ayuda a optimizar los procesos del colectivo, garantizando una disponibilidad constante de productos mediante la gestión del inventario, lo cual es fundamental para la fidelización de clientes y el éxito sostenible del proyecto. Además, una gestión eficaz de inventarios puede traducirse en eficiencia financiera al evitar pérdidas por excedentes u oportunidades perdidas debido a la falta de productos en stock.

Gráfico 5.- Porcentaje de respuestas de la pregunta 5.

Fuente: Google Forms, 2023.

En el Gráfico 5 se muestra los resultados de la pregunta 5, los datos arrojan que el 64% de los productores consideran en su totalidad que la aplicación web genera una reducción de errores en el registro de datos de los pedidos, mientras que el 34% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 2% están cierta medida en desacuerdo con la reducción de errores en el registro de datos de los pedidos que se brinda con la aplicación web, con estos resultados se refleja un posible impacto positivo en la reducción de errores relacionados con la generación de pedidos debido a la manera en cómo funciona la aplicación, lo cual no solo mejoraría la precisión de los registros, sino que también puede traducirse en ahorros significativos de tiempo y recursos.

Gráfico 6.- Porcentaje de respuestas de la pregunta 6.



Fuente: Google Forms, 2023.

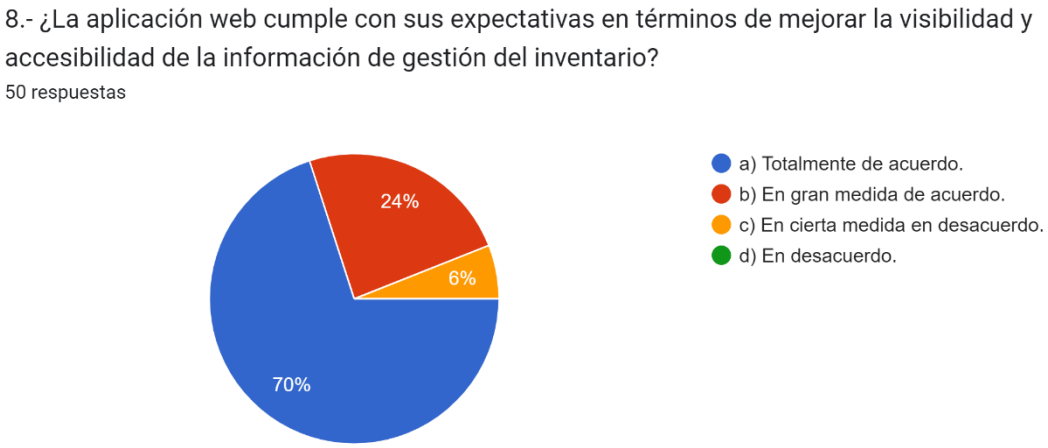
En el Gráfico 6 se muestra los resultados de la pregunta 6, los datos arrojan que el 68% de los productores consideran en su totalidad que la aplicación incrementa la transparencia de la gestión de inventarios dentro del colectivo, mientras que el 30% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 2% están cierta medida en desacuerdo en cuanto a la mejora en la transparencia de inventarios, con estos resultados positivos se refleja una indicación crucial de la efectividad de la herramienta en mejorar la transparencia en la gestión de inventarios debido a su naturaleza. Este aspecto es fundamental, ya que la transparencia no solo facilita una toma de decisiones más informada, sino que también fomenta la confianza entre los miembros del colectivo al brindar una visión clara y accesible de las operaciones relacionadas con los inventarios. Una mayor transparencia puede traducirse en una colaboración más efectiva, y en última instancia, en un mejor rendimiento general del Colectivo de Productores de Manzana Teziutlán. Así, la positividad en las respuestas a esta pregunta no solo respaldaría el valor de la aplicación web en términos de transparencia, sino que también sugiere un impacto positivo en la eficacia y la cohesión del colectivo.

Gráfico 7.- Porcentaje de respuestas de la pregunta 7.

Fuente: Google Forms, 2023.

En el Gráfico 7 se muestra los resultados de la pregunta 7, los datos arrojan que el 66% de los productores consideran que la aplicación web, con su sistema de manejo de inventario y la información incorporada, puede lograr una mejora sustancial en el cumplimiento de pedidos, mientras que el 34% ven que en gran medida se ha logrado este objetivo, con estos resultados positivos se refleja una indicación clave del impacto positivo de la herramienta en la eficiencia operativa y la gestión de pedidos. Un posible aumento en el cumplimiento de pedidos con la aplicación web puede mejorar la capacidad de la empresa para procesar y ejecutar pedidos de manera más efectiva en comparación a como actualmente se maneja este aspecto dentro del colectivo, lo que puede traducirse directamente en una mayor satisfacción del cliente. Además, un cumplimiento más eficiente puede tener implicaciones positivas en términos de optimización de recursos, reducción de costos y fortalecimiento de la posición competitiva de la empresa en el mercado.

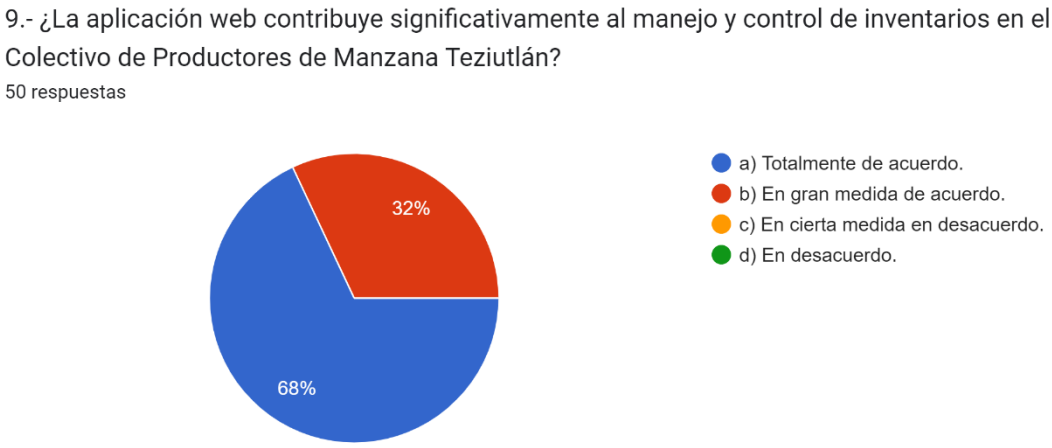
Gráfico 8.- Porcentaje de respuestas de la pregunta 8.



Fuente: Google Forms, 2023.

En el Gráfico 8 se muestra los resultados de la pregunta 8, los datos arrojan que el 70% de los productores consideran que la aplicación cumple con sus expectativas en la mejora de la visibilidad y accesibilidad de la información del inventario, mientras que el 24% ven que en gran medida se ha logrado este objetivo, y solo una pequeña parte correspondiente al 6% están con cierta medida en desacuerdo en cuanto a no lograr su expectativa en la visibilidad y accesibilidad del inventario, con estos resultados positivos se subraya la eficacia y utilidad de la aplicación en la optimización de la visibilidad y accesibilidad de la información relacionada con el inventario. Una evaluación afirmativa indica que la herramienta ha logrado su objetivo de manera efectiva, lo que resalta su valor estratégico para mejorar la eficiencia operativa y la toma de decisiones informada. En este contexto, la confirmación positiva de esta pregunta no solo respalda que la aplicación cuenta con características que dan mejor información acerca del inventario y una forma de acceso a ella más fácil, sino que, también válida su posible impacto directo en la gestión eficiente del inventario, destacando así su importancia dentro del marco general del proyecto.

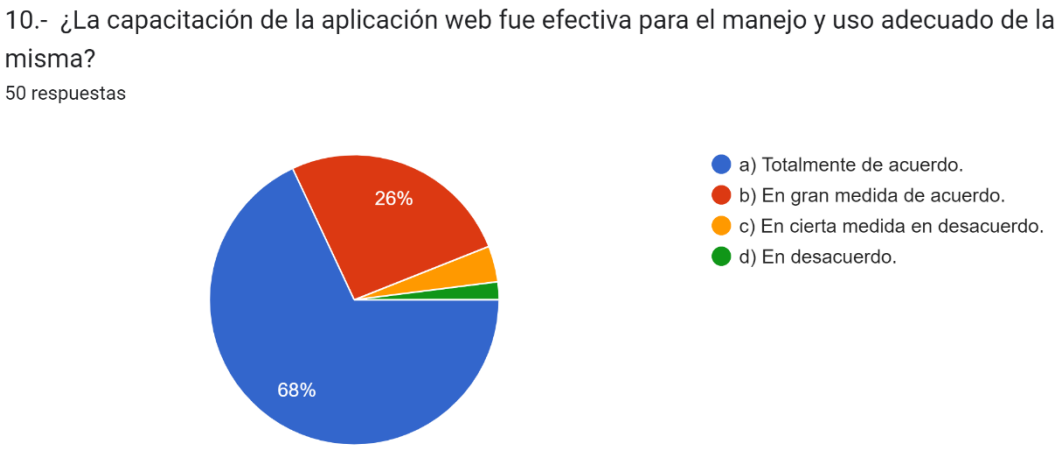
Gráfico 9.- Porcentaje de respuestas de la pregunta 9.



Fuente: Google Forms, 2023.

En el Gráfico 9 se muestra los resultados de la pregunta 9, los datos arrojan que el 68% de los productores consideran que la aplicación web contribuye significativamente en el control de inventarios del colectivo debido a su diseño y funcionamiento, cumpliendo la aplicación su propósito original, mientras que el 32% ven que en gran medida se ha logrado este objetivo, con estos resultados positivos se refleja que la aplicación ha sido efectiva en esta área, se sustenta la idea de que la herramienta tecnológica implementada tiene funciones que mejoran sustancialmente la gestión de inventario. Este resultado positivo no solo valida la eficacia del programa considerada por los productores, sino que también señala oportunidades para optimizar procesos, reducir pérdidas y mejorar la rentabilidad en el contexto específico del colectivo de productores de manzana Teziutlán. Asimismo, confirma la pertinencia de seguir utilizando y potenciando la aplicación como una herramienta clave para el desarrollo y éxito continuo de las operaciones relacionadas con la producción y comercialización de manzanas en el CPMT.

Gráfico 10.- Porcentaje de respuestas de la pregunta 10.



Fuente: Google Forms, 2023.

En el Gráfico 10 se muestra los resultados de la pregunta 10, los datos arrojan que el 68% de los productores están totalmente de acuerdo en que la capacitación recibida para el manejo de la aplicación fue efectiva, mientras que el 26% ven que en gran medida se ha logrado este objetivo, por su parte, un 4% de los productores tienen cierto desacuerdo de la capacitación recibida, y solo una pequeña parte correspondiente al 2% están completamente en desacuerdo con la eficacia de la capacitación, sin embargo se pude apreciar que son datos bastante positivos que reflejan que los participantes perciben la capacitación como beneficiosa y exitosa. Esto no solo implica que la aplicación web y todas sus funcionalidades son entendibles, sino también sugiere una mejora en las habilidades y competencias de los usuarios para utilizar la aplicación de manera eficaz.

CAPÍTULO IV

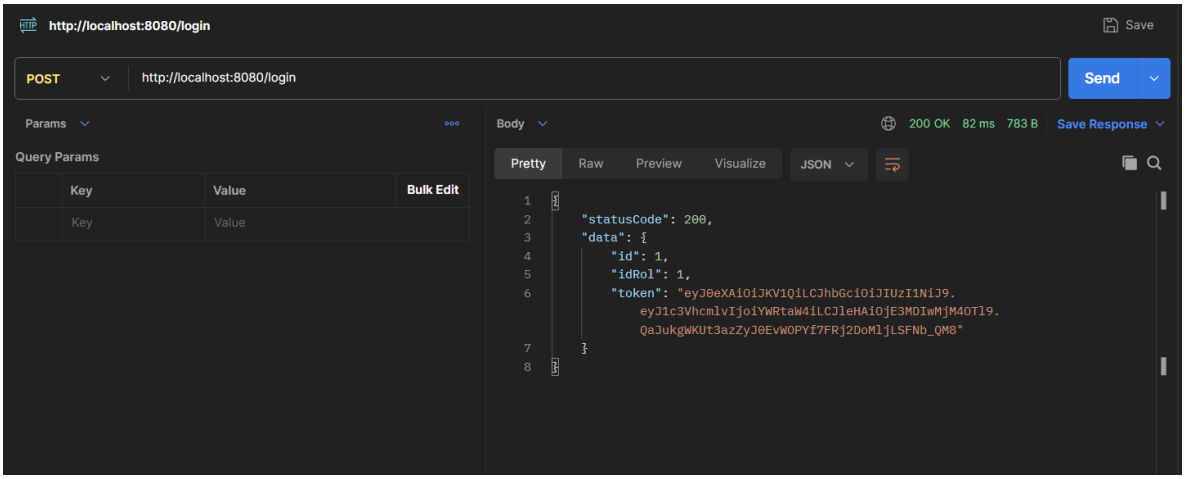
RESULTADOS

4.1 resultados del desarrollo de la aplicación

Ilustración 158.- Inicio de sesión.

Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

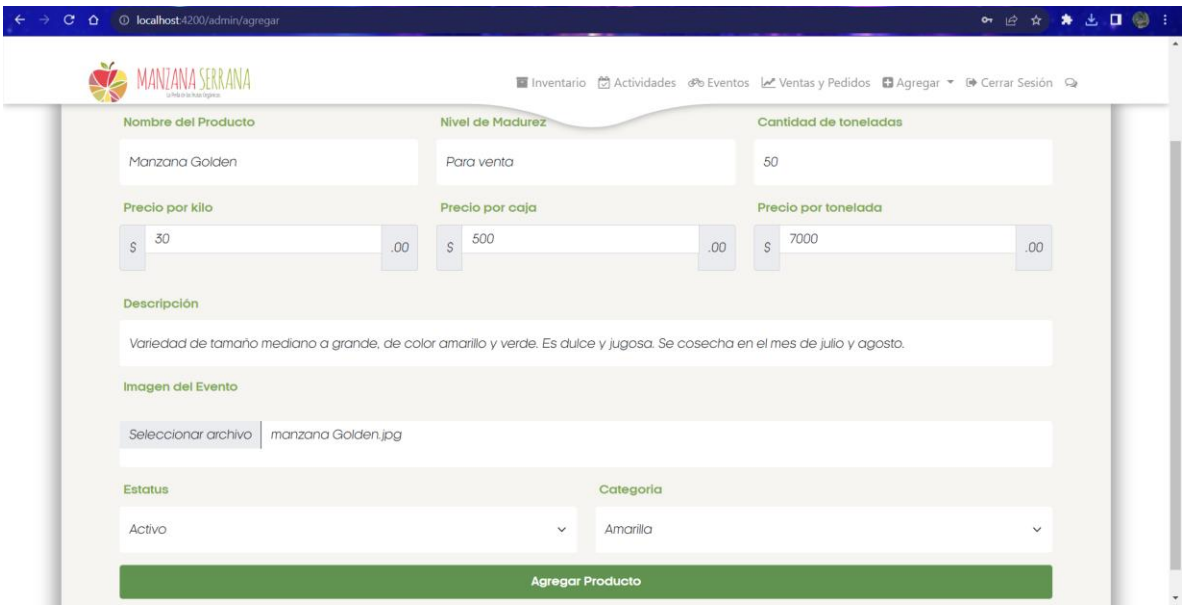
Ilustración 159.- Validación de inicio de sesión en Postman.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

En la Ilustración 158 se puede apreciar el inicio de sesión que deberá realizar el administrador para poder ir a la parte de gestión de inventarios, por su parte en la Ilustración 159 se aprecia por medio de Postman la validación que se hace verificando si ese usuario existe en la base de datos y el tipo de rol que ocupa para poder dar acceso dentro de la aplicación.

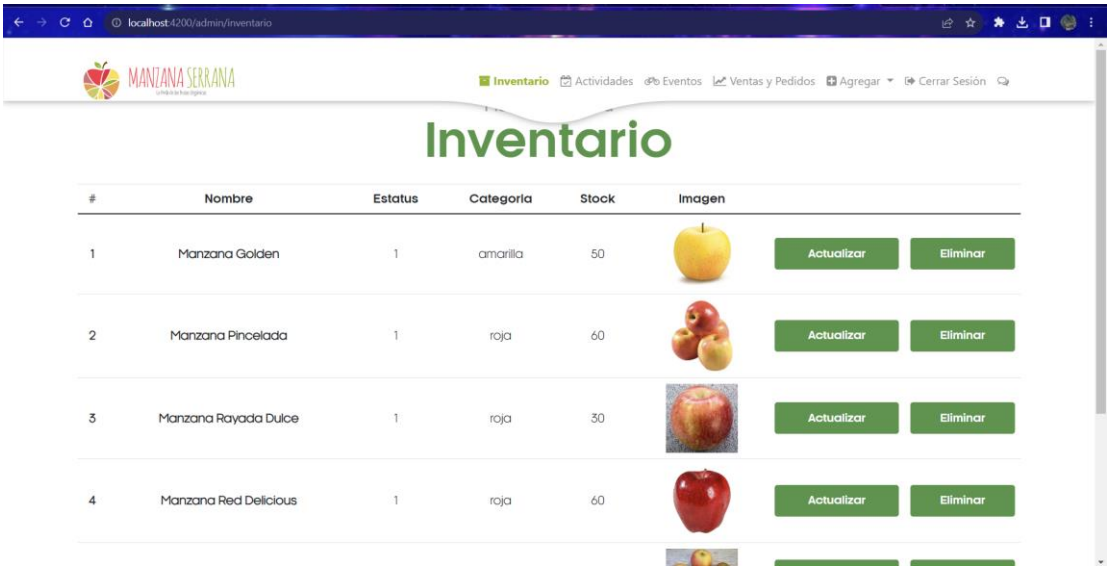
Ilustración 160.- Creación de una manzana.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

En la Ilustración 160 se puede apreciar cómo se crea una manzana, para ello el administrador debe dirigirse en la barra de navegación a agregar y posteriormente a producto, ahí el administrador tiene la capacidad de poner los datos de dicha manzana, además, tiene la oportunidad de registrar desde un inicio la cantidad de toneladas que se tienen en el inventario de dicho producto.

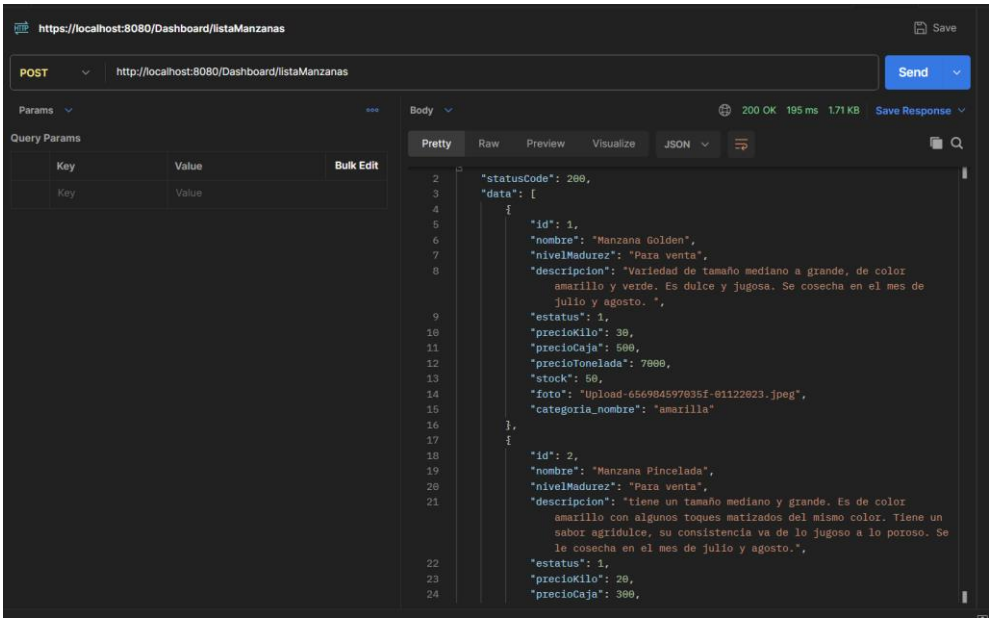
Ilustración 161.- Manejo del inventario.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

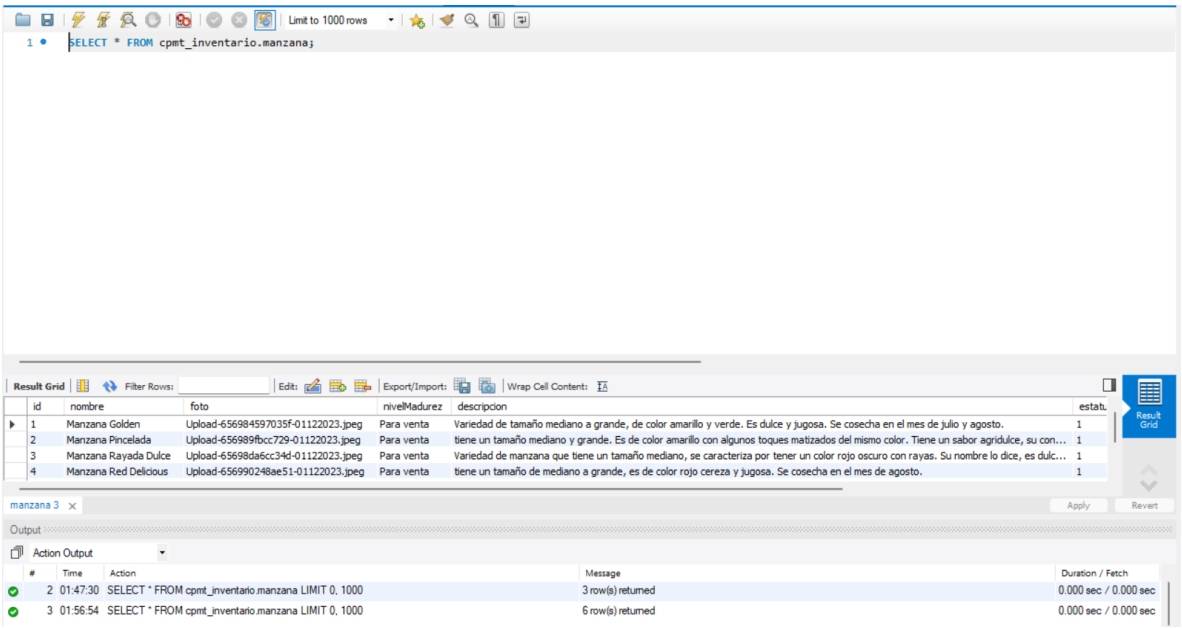
En la Ilustración 161 se muestra el manejo principal del inventario, en donde se puede gestionar de una manera fácil los productos del colectivo, en este caso son las manzanas, puede actualizar dicha manzana, eliminarla, o sencillamente verificar el estatus, la cantidad de stock y algunas otras cuestiones de las manzanas, es muy intuitiva la gestión del inventario, si se desea actualizar simplemente se elige esa opción, se presenta un formulario como el que se mostró cuando se registró el producto y el administrador modificara los datos que considere pertinentes de la manzana seleccionada, si desea por alguna cuestión eliminar esa manzana, existe el botón eliminar que solo pedirá verificar dicha acción y eliminará el producto, como se puede observar se tiene un manejo de inventarios demasiado claro para su fácil manejo y donde se puede tener un mejor manejo del stock de productos.

Ilustración 162.- Lista de manzanas en Postman.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

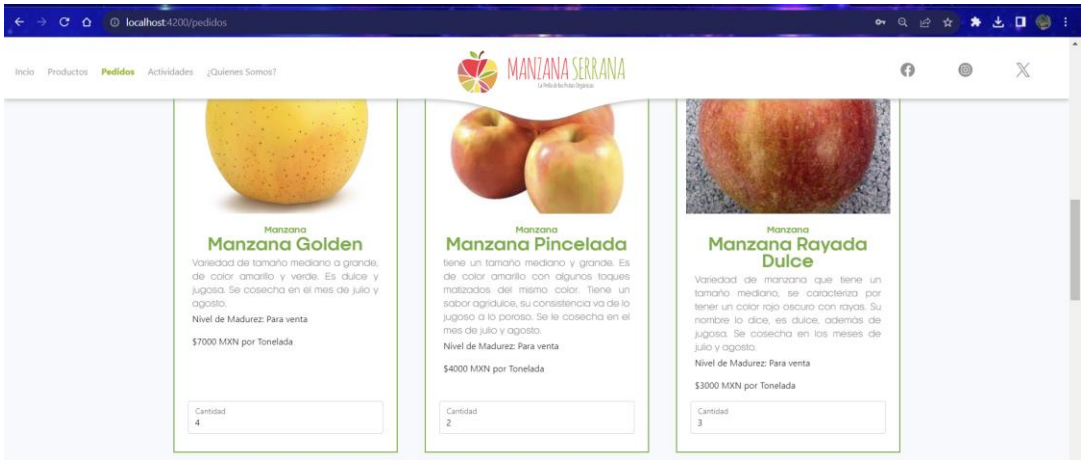
Ilustración 163.- Lista de manzanas en la base de datos.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

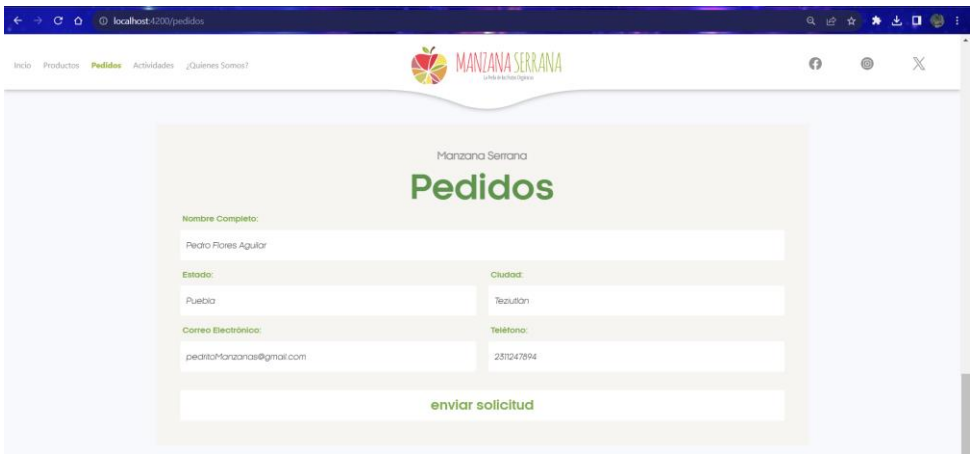
En la Ilustración 162 se muestra la lista de manzanas por medio de Postman, demostrando que la API está interviniendo entre la interfaz y la base de datos, mientras que por su parte en la Ilustración 163 se deja en claro que los datos si se están almacenando en la base de datos de una manera satisfactoria

Ilustración 164.- Selección de manzanas del pedido.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

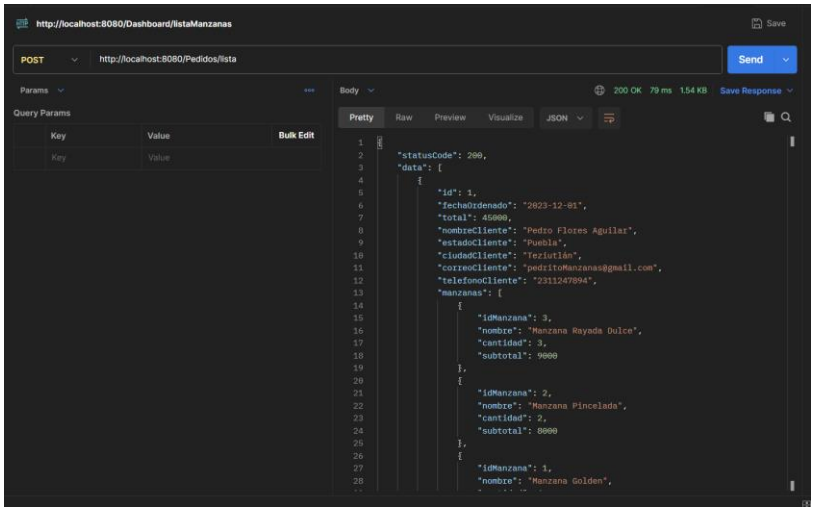
Ilustración 165.- Datos del pedido.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

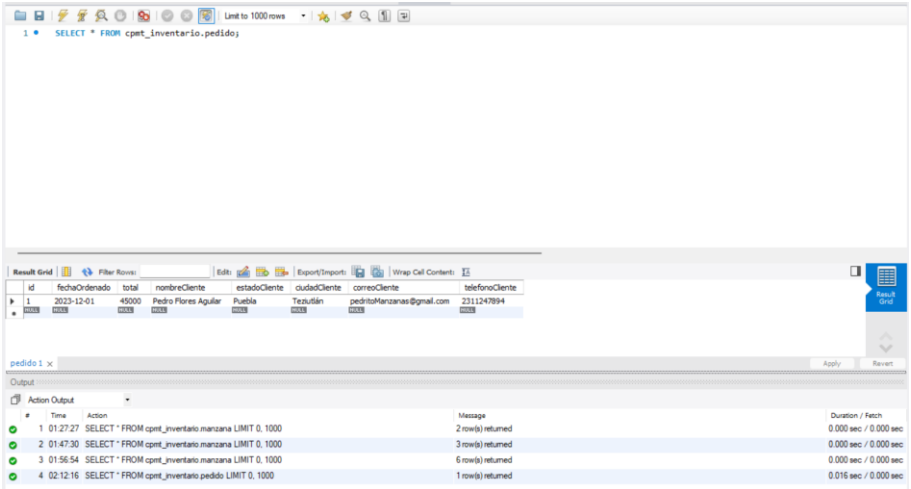
En la Ilustración 164 e Ilustración 165 se hace un pedido en la aplicación por parte de un usuario en el apartado de pedidos al mayoreo en el home de la aplicación, aquí se crea un pedido con los datos necesarios como la lista de manzanas que desea comprar y los datos del comprador.

Ilustración 166.- Lista de pedidos en Postman.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

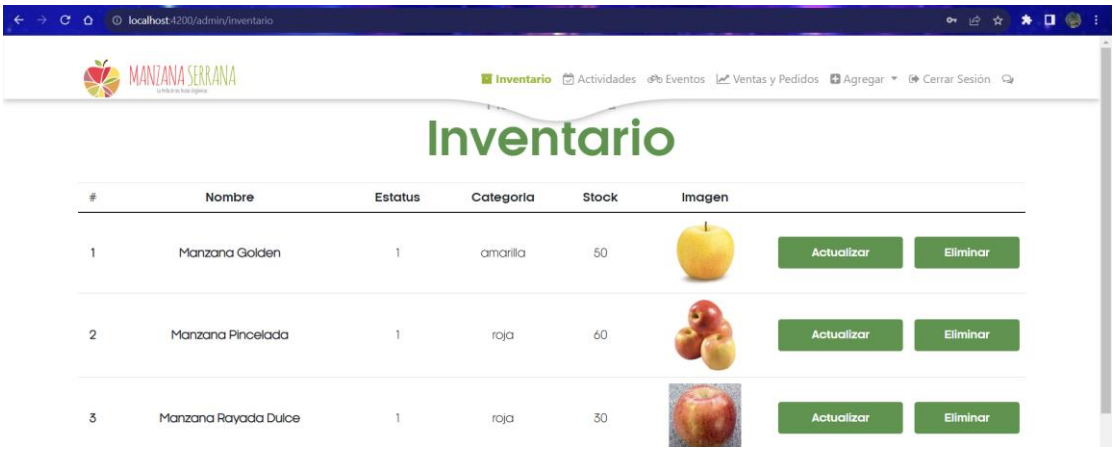
Ilustración 167.- Lista de pedidos en la base de datos.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

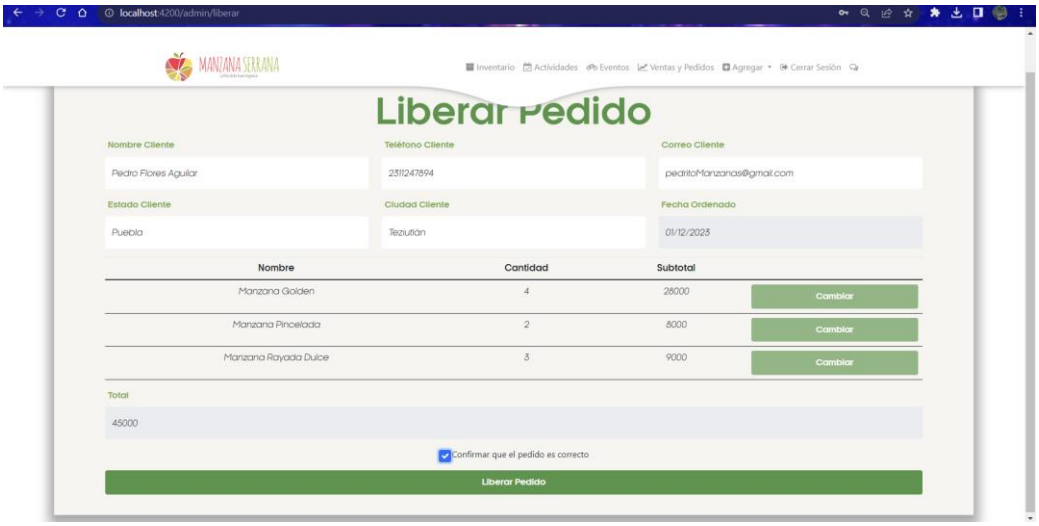
En la Ilustración 166 e Ilustración 167 se muestra que el pedido realizado si ha sido procesado por la aplicación y posteriormente se han registrado esos valores dentro de la base de datos para la posterior liberación por parte del administrado una vez se acepte y se realice el pedido.

Ilustración 168.- Stock anterior.



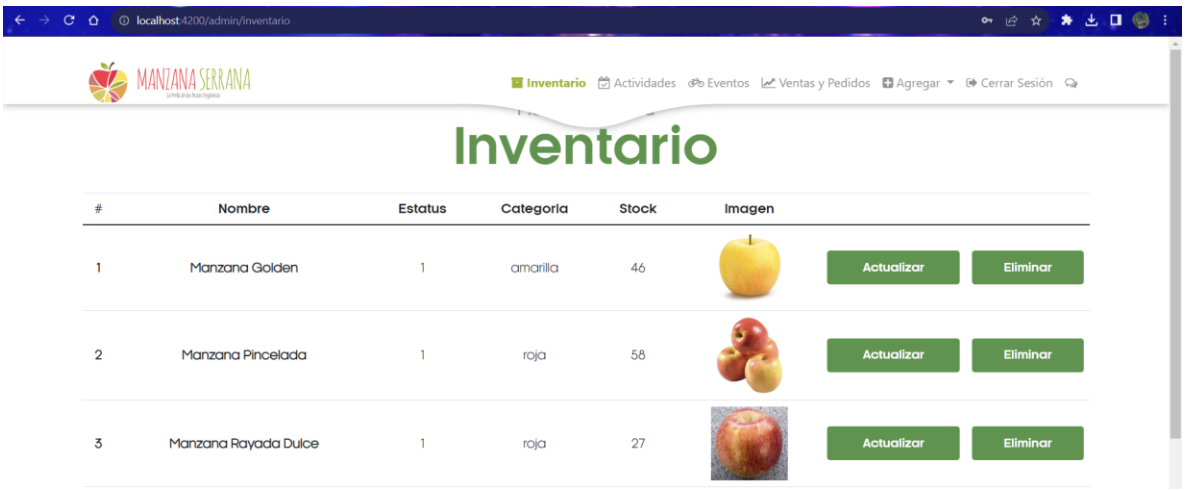
Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 169.- Liberación del pedido.



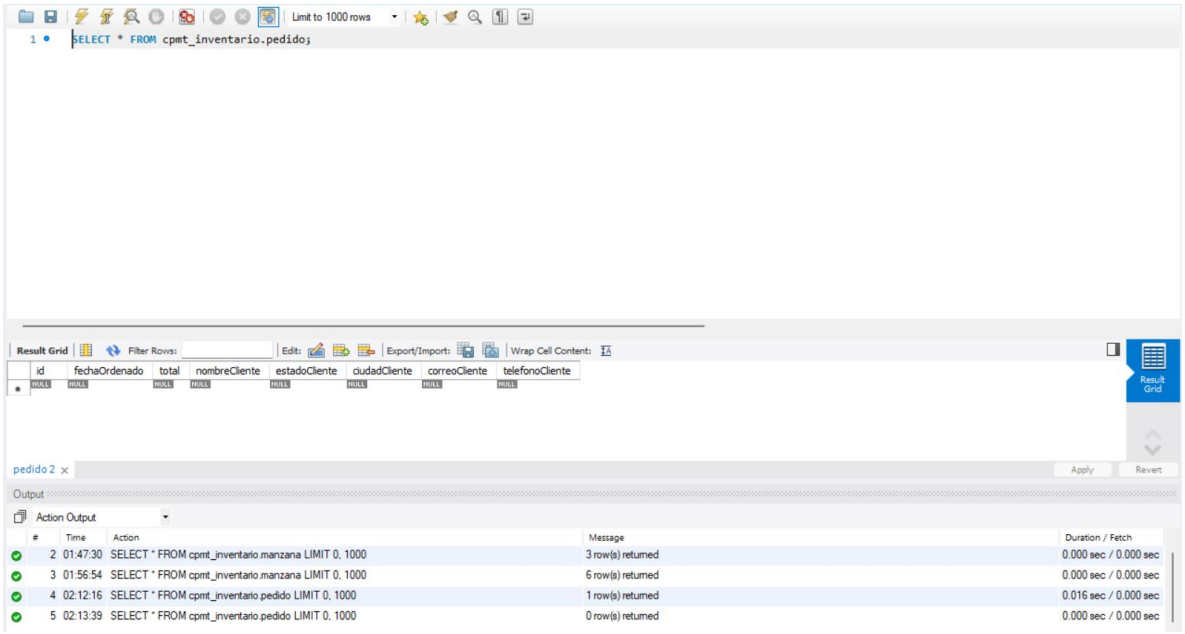
Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 170.- Verificación de nuevo stock.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

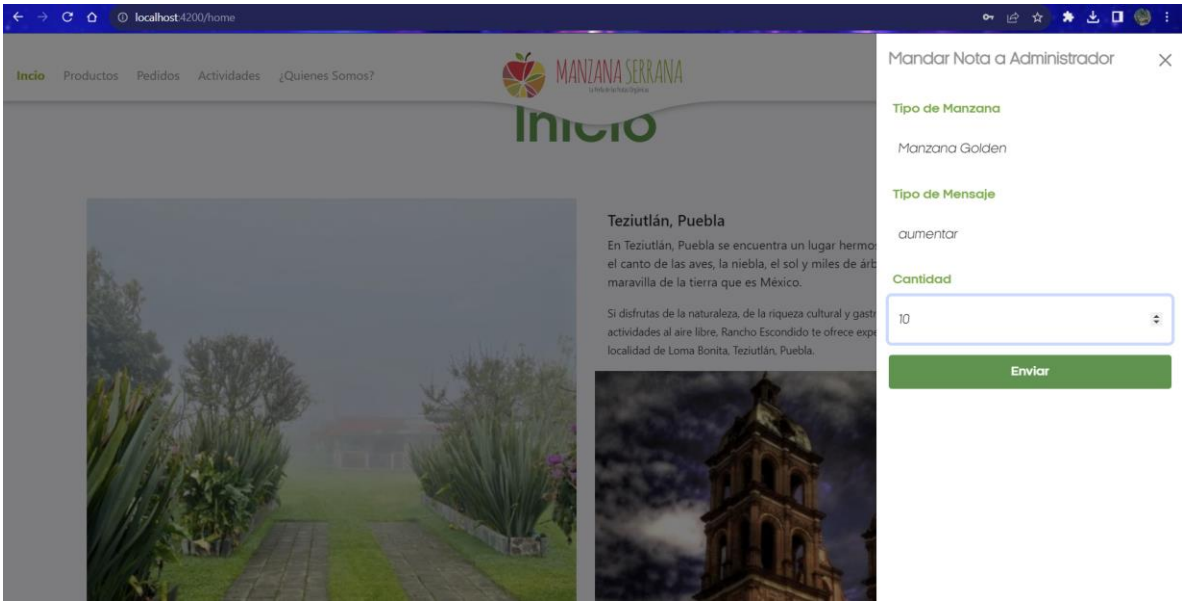
Ilustración 171.- Lista de pedidos luego de la modificación del stock.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

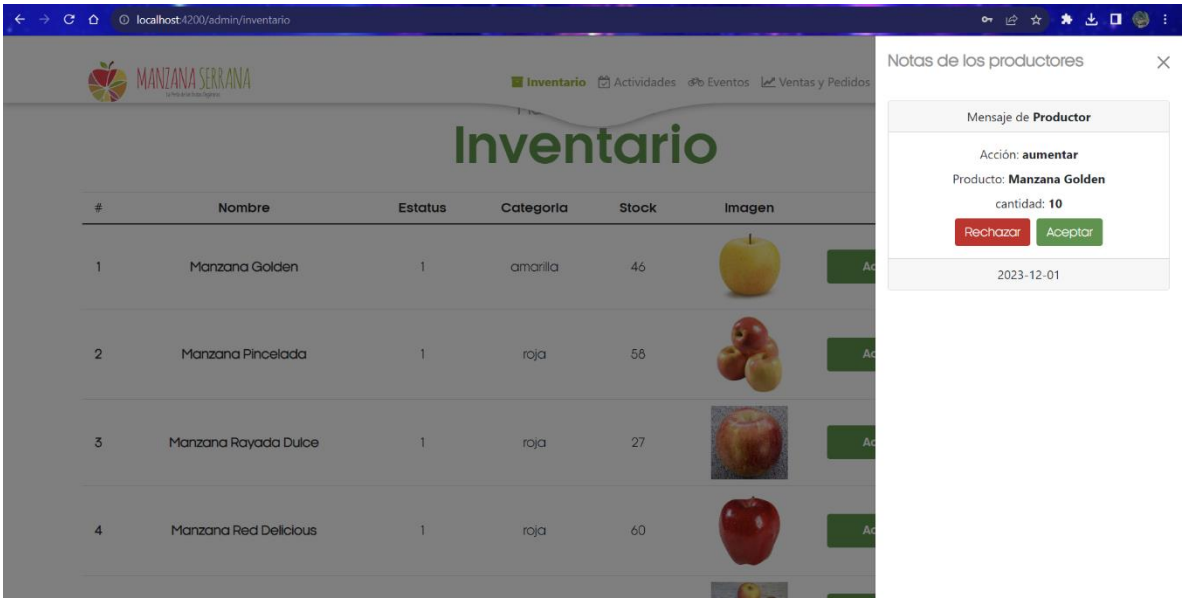
En la Ilustración 168 se muestra el stock de las manzanas que se solicitaron en el pedido, previo a la liberación del mismo. En la Ilustración 169 se muestra la liberación de un pedido, donde el administrador antes de liberar dicho pedido verifica los datos del mismo, dándole la opción de modificar alguno de los datos en caso de ser necesario, o incluso de modificar las manzanas en el pedido solicitado, en caso de que la cantidad solicitada exceda el stock de alguna de las manzanas no permitirá realizar dicha operación, una vez liberadas se aprecia en la Ilustración 170 que el stock se ha modificado automáticamente después de liberar el pedido, y en la Ilustración 171 se observa que la API ha borrado el registro del pedido en la base de datos una vez liberado.

Ilustración 172.- Solicitud de modificación de stock de un productor.



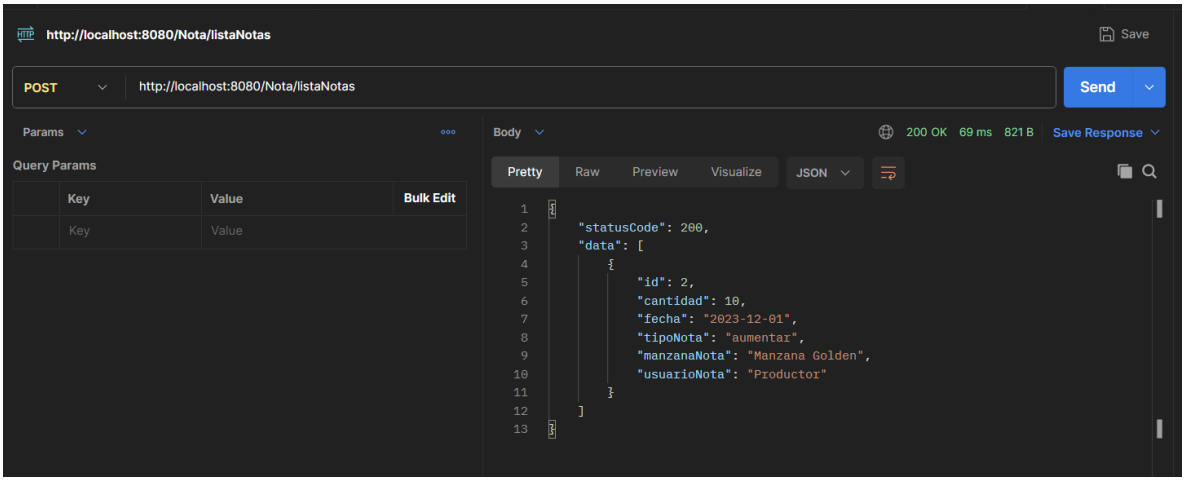
Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 173.- Notas de modificación de stock para el administrador.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 174.- Lista de notas en Postman.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 175.- Inventario modificado por solicitudes.

Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

En la Ilustración 172 se muestra lo que pasa si se inicia sesión como un productor, el mismo tendrá acceso a solicitudes en el stock de las manzanas por medio de mensajes que aumenten o disminuyan el inventario de la misma, en dicha imagen se aprecia el formulario que el productor debe de llenar, en el formulario el escoge la manzana de la que quiere modificar el stock, también el tipo de nota que está solicitando, si es para aumentar o disminuir, y finalmente, ingresa la cantidad que desea modificar del stock, en la Ilustración 173 se puede observar que estos datos son mandados al administrador quien ve las solicitudes en la barra de navegación en el icono de mensaje, ahí se despliega las notas de los productores, en donde el administrador decide si aceptar o no la solicitud, en caso de que sea de tipo reducción y supere el stock que se tiene de la manzana, no se podrá realizar la acción, si el administrador decide aceptarla se eliminará la nota y se harán las modificaciones en el stock, si se decide eliminar, se eliminara la nota de igual manera, pero sin modificar el inventario, en la Ilustración 174 se puede observar la lista de notas por medio de Postman, las cuales son procesadas por la API, finalmente, en la Ilustración 175 se muestra el stock modificado una vez se acepté la solicitud.

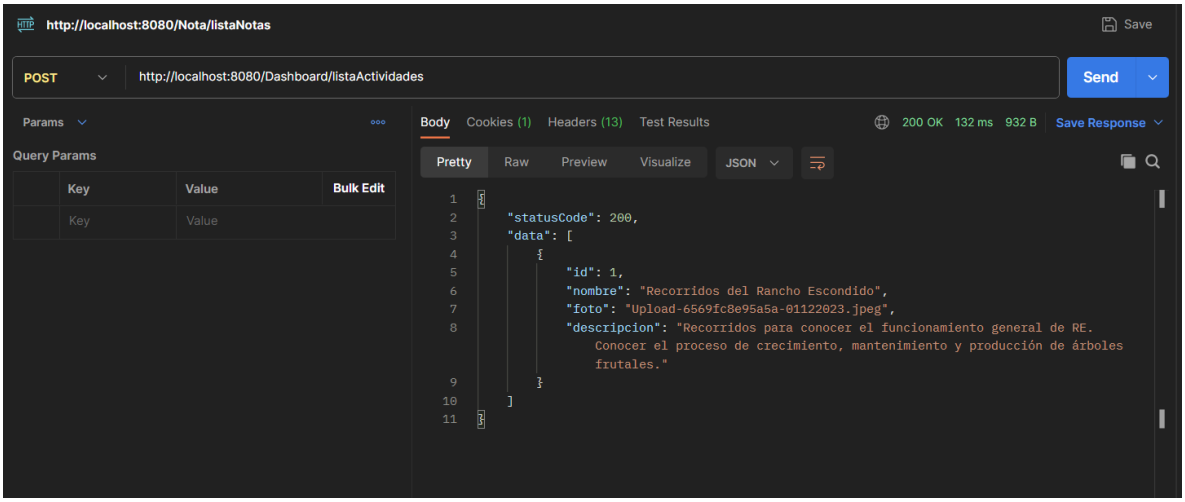
Ilustración 176.- Agregar funciones del colectivo.

Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

Ilustración 177.- Gestionar funciones del colectivo.

Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

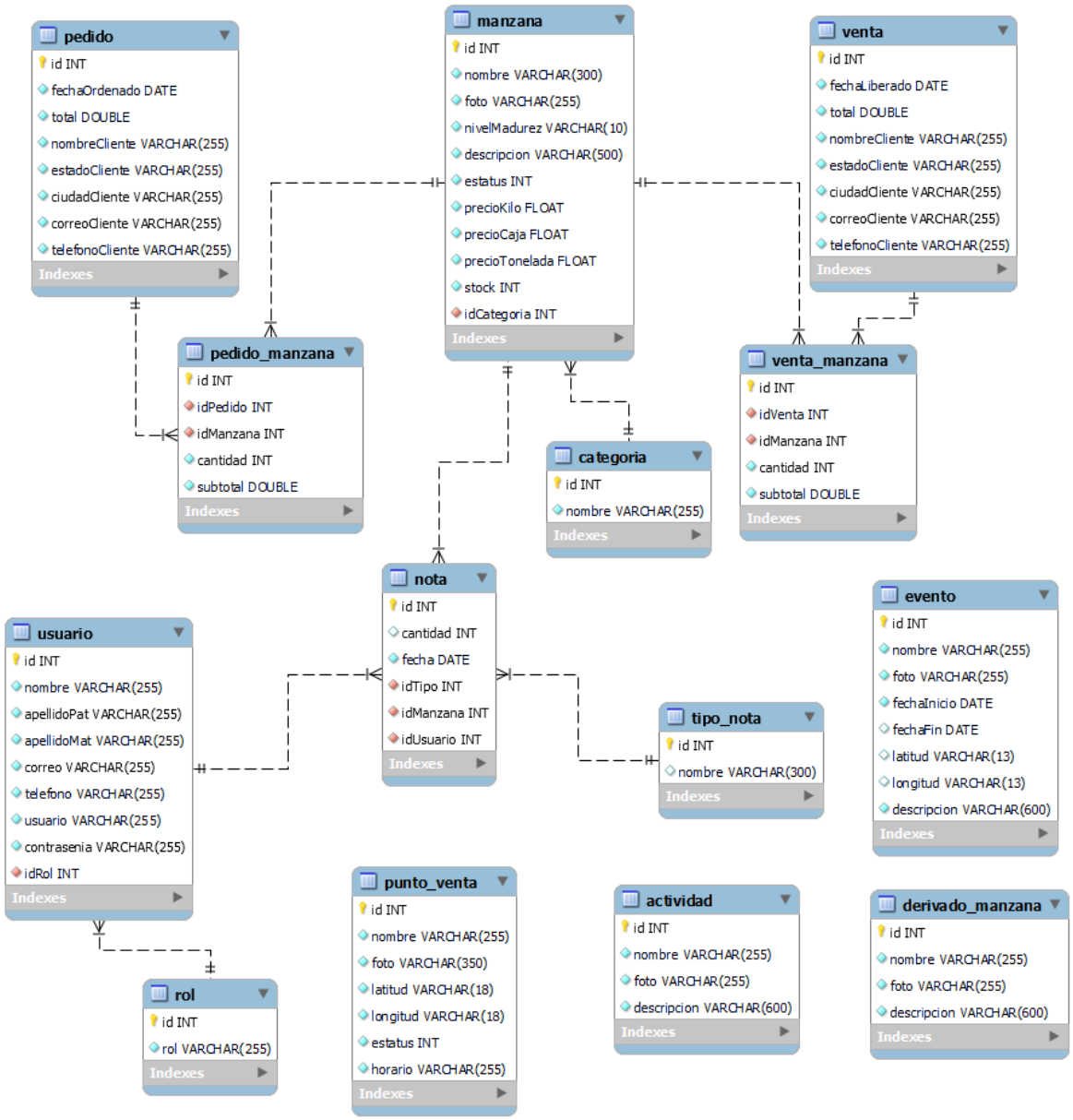
Ilustración 178.- Funciones del colectivo en Postman.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

En la Ilustración 176 se muestra cómo se pueden agregar otras funciones del colectivo para su gestión, por ejemplo, actividades, eventos, derivados de la manzana o puntos de venta, esto mediante la barra de navegación donde se debe de seleccionar agregar, y escoger entre las opciones mencionadas, donde se abrirá un formulario que se deberá llenar como el de la Ilustración, con los datos que se necesiten. En la Ilustración 177 se muestra cómo se pueden gestionar las diversas funciones del colectivo, mediante la barra de navegación, se puede acceder a la gestión de la función que se necesite, en donde podrá actualizar, eliminar o simplemente visualizar las actividades, eventos, derivados de la manzana o puntos de venta, cada una en su respectiva pestaña de la barra de navegación, en la Ilustración 178 se puede visualizar con Postman que estas funciones son gestionadas por la API, y tienen conexión con la base de datos.

Ilustración 179.- Esquema de base de datos.



Fuente: COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT), 2023.

En la Ilustración 179 se muestra el diagrama UML que brinda el gestor de base de datos, con lo cual se evidencia la creación y existencia de la BD, quedando de esta manera en MySQL.

CAPÍTULO V

CONCLUSIONES

5.1 Conclusiones del proyecto, recomendaciones y experiencia profesional y personal adquirida

Con base en las respuestas altamente positivas obtenidas durante las entrevistas, y al correcto funcionamiento de la aplicación para gestionar el stock de las manzanas producidas por el colectivo, se puede concluir que el desarrollo de la aplicación web para la optimización de inventarios y gestión de stock de manzana orgánica en el Colectivo de Productores de Manzana Teziutlán ha cumplido con éxito el objetivo del trabajo.

La hipótesis plantea que la implementación de una aplicación web para la gestión de inventario contribuirá en la eficiencia de la gestión de stock de manzana orgánica producida por el Colectivo de Productores de Manzana Teziutlán (CPMT); lo cual ha sido respaldado de manera sustancial por las respuestas basadas en las experiencias positivas de los productores con el uso de la aplicación web, información analizada de manera cuantitativa. Los resultados obtenidos sugieren una mejora significativa en el manejo de inventarios dentro del colectivo con el funcionamiento de la aplicación al mostrar datos precisos de la disponibilidad de manzanas para venta.
En este contexto, los productores han expresado su apreciación por la eficiencia y agilidad que la aplicación aporta a la gestión de su inventario, esta información es de gran ayuda en la toma de decisiones, y en un manejo de su inventario más controlado. La disponibilidad adecuada de productos se denota como una característica que implementa la solución tecnológica y que para los productores será de gran ayuda para la parte de marketing.

La eficacia de la herramienta no solo se refleja en la optimización de los procesos de inventario, sino que también impacta positivamente en la coordinación interna y en la satisfacción de los clientes. Asimismo, la aplicación tiene la capacidad de generar una mayor colaboración y coordinación entre los miembros del colectivo, fortaleciendo las relaciones internas y mejorando la eficiencia operativa.

Con respecto a la pregunta de investigación sobre si ¿La implementación de una aplicación web para la gestión de inventario contribuirá en la eficiencia de la gestión de stock de manzana orgánica producida por el Colectivo de Productores de Manzana Teziutlán (CPMT)?, las conclusiones revelan un panorama alentador. La aplicación demostró ser una herramienta efectiva para optimizar el control de inventarios, permitiendo una gestión más precisa y transparente de los recursos disponibles dentro de sus características, la evaluación cuantitativa de la implementación se llevará a cabo utilizando los datos de producción registrados en la base de datos, además de analizar los pedidos, las ventas y la gestión de los diversos puntos de venta, donde el colectivo vera la eficiencia en números de la solución tecnológica. Además, la aplicación facilita el dar a conocer la información de los inventarios a todo el colectivo, contribuyendo así a información más clara y concisa del mismo. Estas conclusiones respaldan la utilidad y el impacto positivo que tiene la aplicación en la mejora de la eficiencia y comunicación.

En la conclusión del estudio, se confirma que los resultados obtenidos respaldan la hipótesis planteada, indicando que la aplicación web es una contribución positiva que contribuye en la optimización de inventarios y gestión de stock de manzana orgánica, aumentando la eficiencia del mismo dentro del CPMT. Este logro representa un avance significativo hacia la consecución del objetivo general de mejorar la gestión de inventarios dentro del contexto del colectivo.

5.2 Conclusiones relativas a los objetivos específicos

Las conclusiones derivadas de los objetivos específicos reflejan un éxito significativo en el proyecto de desarrollo de la aplicación web para la gestión de inventarios de la manzana orgánica en el Colectivo de Productores de Manzana Teziutlán (CPMT).

En primer lugar, el objetivo de crear una aplicación web intuitiva y amigable se ha alcanzado con éxito, según las respuestas positivas en las entrevistas. Los productores destacan la facilidad de uso y la eficiencia en la gestión de inventarios proporcionada por la aplicación.

En segundo lugar, la integración del sistema de gestión de inventario se revela como un acierto, ya que las respuestas indican que la aplicación demuestra ser una posible herramienta efectiva para asegurar la disponibilidad de productos para satisfacer la demanda de los clientes.

Asimismo, la capacitación del personal involucrado en el manejo de inventarios se ha revelado como un factor crucial. Las respuestas positivas indican que el personal se siente confiado en el uso adecuado de la aplicación, reduciendo dudas y mejorando la eficiencia en la gestión.

5.3 Conclusiones relativas al objetivo general

Las conclusiones referentes al objetivo general de optimizar la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán (CPMT) mediante el desarrollo de una aplicación web, con el fin de agilizar el control de inventarios y garantizar la disponibilidad adecuada de productos para satisfacer la demanda de los clientes se ha alcanzado con éxito, la aplicación optimiza la gestión de inventarios que tiene el colectivo, agilizando el control que se tiene de los mismos, además de ser una forma de marketing digital que ayudará al colectivo en la disponibilidad de sus productos para sus clientes.

5.4 Aportaciones originales

Desarrollo de una Aplicación Específica para la Gestión de Inventario de Manzana Orgánica:

El núcleo innovador de este proyecto reside en la concepción y construcción de una aplicación web exclusivamente diseñada para la gestión de inventarios de manzana orgánica. Esta iniciativa no solo representa una respuesta a la ausencia de soluciones específicas en el ámbito del Colectivo de Productores de Manzana Teziutlán (CPMT) sino que también constituye una contribución única al campo de la gestión de inventarios en el contexto agrícola. La adaptabilidad y funcionalidad a medida de esta aplicación reflejan la atención meticulosa a las necesidades particulares del Colectivo, transformando el proceso de control de inventarios en una experiencia

eficiente y adaptada a las características específicas de la producción de manzanas orgánicas.

Optimización de Procesos de Pedidos y Entregas:

La simplificación de los procesos vinculados a la gestión de pedidos y entregas, junto con la reducción de los tiempos de respuesta, representa un avance significativo en la eficiencia operativa del Colectivo de Productores de Manzana Teziutlán. Esta mejora tangible no solo impacta directamente en la productividad, sino que también incide en la experiencia general de los productores y clientes. La optimización de estos procesos contribuye a una mayor agilidad en la satisfacción de la demanda, fomentando la eficacia y la competitividad del Colectivo en el mercado.

Integración Estratégica de la Gestión:

La aplicación trasciende su función de gestión interna al transformarse en un elemento clave para la promoción integral de los productos del Colectivo, la destacada visibilidad de eventos significativos, la gestión de actividades próximas, y la identificación de puntos de venta disponibles. Esta característica se erige como una contribución original al robustecer la presencia del Colectivo, al mismo tiempo que optimiza las operaciones internas. Al proporcionar una interfaz digital multifuncional, la aplicación se convierte en una herramienta estratégica de gestión que va más allá de la administración de inventarios, actuando como un medio dinámico y completo para conectar con los consumidores. Este enfoque holístico no solo fortalece la visibilidad del Colectivo en el mercado, sino que también fomenta la participación activa de los consumidores en las diversas actividades impulsadas por el Colectivo.

5.5 Limitaciones del modelo planteado

Restricción en la Visibilidad del Stock: La limitación principal reside en la restricción de visualización del stock exclusivamente al administrador. Esta restricción, aunque puede ser necesaria para ciertos aspectos de seguridad, limita un poco la

colaboración entre los miembros del Colectivo de Productores de Teziutlán (CPMT), dificultando la toma de decisiones compartida.

Ausencia de Informes Detallados: La falta de generación de informes detallados constituye una limitación en la capacidad analítica de la aplicación. La ausencia de informes detallados puede restringir la capacidad de los productores y administradores para obtener una comprensión profunda de los patrones y tendencias en la gestión de inventarios, limitando así la capacidad de realizar análisis estratégicos.

Excesiva Dependencia del Control Administrativo: La marcada dependencia de la aplicación al control administrativo representa una limitación en términos de flexibilidad y autonomía. Esto puede generar cuellos de botella en la toma de decisiones, especialmente si el administrador se convierte en un punto único de falla o si la carga de trabajo se vuelve abrumadora.

Limitación en la Gestión de Stock para Comercio Electrónico Directo: La aplicación no ha sido completamente adaptada para una gestión de stock orientada al comercio electrónico directo. Esta limitación puede dificultar la expansión de las operaciones del Colectivo hacia plataformas de venta en línea, privándolo de oportunidades potenciales de llegar a un público más amplio y diversificado.

Complejidad en la Implementación de Gestión del Stock para Comercio Electrónico: La implementación de una gestión de stock específicamente diseñada para el comercio electrónico puede ser un desafío adicional, considerando las dinámicas y requerimientos particulares de este entorno. La complejidad técnica y operativa asociada con esta expansión podría requerir recursos adicionales y capacitación especializada.

Limitación en la Colaboración y Compartición de Datos: La falta de herramientas específicas para la colaboración y compartición de datos entre los miembros del Colectivo puede limitar la eficiencia en la comunicación y la toma de decisiones

colaborativa. La aplicación podría beneficiarse de características que fomenten la colaboración directa entre los usuarios autorizados.

Estas limitaciones representan oportunidades de mejora y áreas clave para el desarrollo continuo del modelo propuesto, permitiendo así la evolución de la aplicación hacia un sistema más completo, inclusivo y adaptable a las necesidades cambiantes del Colectivo de Productores de Manzana Teziutlán.

5.6 Recomendaciones

Recomendaciones para Futuras Actividades y Programas de Acción:

Continuar la adaptación y mejora de la aplicación. Se sugiere un enfoque continuo en la adaptación y mejora de la aplicación, considerando retroalimentación constante de los usuarios. Esto garantizará que la herramienta evolucione de manera dinámica, respondiendo a las cambiantes necesidades del Colectivo y del mercado enfocándose en la mejora de la gestión de inventarios.

Gestión de stock para comercio electrónico directo. Para abordar la limitación en la gestión de stock orientada al comercio electrónico, se recomienda realizar una evaluación exhaustiva de las plataformas de comercio electrónico disponibles en el mercado. Se sugiere seleccionar una plataforma compatible con las operaciones del Colectivo, que ofrezca herramientas avanzadas de gestión de inventarios, integración con sistemas existentes y una experiencia de usuario óptima

Ampliar la capacitación del personal. Proporcionar sesiones de capacitación periódicas para el personal involucrado en la gestión de inventarios. Este enfoque garantizará que todos los usuarios estén plenamente capacitados en el uso eficiente de la aplicación, maximizando así su potencial y minimizando posibles errores operativos.

Establecer un sistema de retroalimentación continua. Crear un sistema estructurado de retroalimentación continua con los productores y clientes para evaluar la eficacia de la aplicación. Este mecanismo permitirá identificar áreas de mejora y ajuste

constante, asegurando que la aplicación siga siendo una solución óptima en el tiempo.

Crear un programa de seguimiento post-implementación. Establecer un programa estructurado de seguimiento post-implementación para evaluar continuamente el impacto de la aplicación a lo largo del tiempo. Esto permitirá ajustes y mejoras proactivas basadas en la evolución de las necesidades y desafíos del Colectivo.

Implementación de notificaciones automáticas. Se aconseja la incorporación de un sistema de notificaciones automáticas que alerte a los productores sobre niveles críticos de inventario o eventos importantes, como la próxima cosecha. Esto permitirá una respuesta proactiva y una gestión más eficiente.

Integración de herramientas de análisis predictivo. Explorar la integración de herramientas de análisis predictivo permitirá anticipar tendencias futuras en la demanda y el comportamiento del inventario. Esto respaldará decisiones estratégicas y mejorará la planificación a largo plazo.

Recomendaciones de experiencia profesional:

Aplicación práctica de habilidades desarrolladas. Se recomienda aplicar activamente las habilidades técnicas y de gestión adquiridas. La experiencia en el desarrollo de la aplicación y la gestión de inventarios ofrece una base sólida para abordar desafíos similares en entornos laborales más amplios.

Incorporación de mejores prácticas en la gestión agrícola. Se aconseja integrar las mejores prácticas en la gestión de inventarios en futuros roles profesionales relacionados con la agricultura. La experiencia práctica en el Colectivo de Productores de Manzana Teziutlán puede ser un activo valioso al contribuir a la eficiencia operativa y la sostenibilidad en otras organizaciones agrícolas.

Participación continua en proyectos de innovación. Se sugiere buscar oportunidades continuas de participación en proyectos de innovación y tecnología aplicada a la agricultura. Esta participación activa permitirá mantenerse actualizado en las últimas

tendencias y contribuirá al crecimiento profesional en el ámbito de la gestión agrícola y tecnológica.

Recomendaciones de experiencia personal:

Reflexión para el crecimiento personal. Invitar a reflexionar sobre la experiencia personal adquirida en el proyecto, identificando valores personales fortalecidos y áreas de crecimiento. Esta reflexión puede servir como guía para el desarrollo personal y la toma de decisiones alineadas con las aspiraciones individuales. Exploración de oportunidades emprendedoras. Se alienta a considerar la posibilidad de explorar oportunidades emprendedoras en el ámbito agrícola, utilizando la experiencia del proyecto para iniciar iniciativas propias. Esta iniciativa no solo impulsa la autorrealización personal, sino que también contribuye al desarrollo del sector agrícola local.

CAPÍTULO VI

COMPETENCIAS DESARROLLADAS

6.1. Competencias desarrolladas y/o aplicadas

Competencias genéricas:

A lo largo del proyecto para optimizar la gestión de inventarios de la manzana orgánica del Colectivo de Productores de Manzana Teziutlán, se ha cultivado diversas competencias genéricas que han enriquecido el desarrollo personal y profesional del residente.

- Comunicación efectiva. Durante la creación de la interfaz de usuario y la capacitación del personal en el uso de la aplicación web, se perfecciono la habilidad para comunicarme de manera clara y persuasiva. Esta competencia no solo implica transmitir información de manera efectiva, sino también adaptarse a las necesidades y comprensión de mi audiencia.
- Habilidades tecnológicas. El desarrollo de la aplicación web ha proporcionado habilidades tecnológicas sólidas. Desde la programación hasta la implementación de nuevas tecnologías de información y comunicación, se ha logrado no solo entender, sino también aplicar con destreza las herramientas digitales necesarias para la gestión eficiente de inventarios.
- Resolución de problemas. Enfrentar los desafíos inherentes a la gestión de inventarios y las características del programa ayudaron a formar un solucionador de problemas ágil. Identificar las raíces de los problemas y proponer soluciones prácticas ha fortalecido la capacidad para abordar desafíos con un enfoque proactivo.
- Capacidad analítica. La capacidad de abstraer, analizar y sintetizar información se ha vuelto una segunda naturaleza en la optimización de inventarios. Esta competencia ha sido vital para garantizar que la información fluya de manera coherente y efectiva en la aplicación.
- Toma de decisiones La gestión de inventarios demanda la toma de decisiones rápidas y efectivas. El proyecto ha permitido desarrollar habilidades para evaluar situaciones y tomar decisiones informadas, considerando siempre el impacto a corto y largo plazo.

- Compromiso ético. La integridad y transparencia en el manejo de la información han sido pilares en el proyecto.
- Trabajo en equipo. La capacitación del personal involucrado en la gestión de inventarios ha resaltado la importancia del trabajo en equipo. Fomentar la colaboración y crear un ambiente propicio para el aprendizaje mutuo ha sido esencial en el éxito del proyecto.
- Autogestión del aprendizaje. La naturaleza cambiante de la tecnología requiere una constante autogestión del aprendizaje. La realización del proyecto ha permitido desarrollar la capacidad de aprender de forma autónoma, adaptándose continuamente a las innovaciones tecnológicas.
- Espíritu emprendedor. El diseño y ejecución del proyecto reflejan el espíritu emprendedor. Buscar constantemente oportunidades de mejora y encontrar soluciones innovadoras son rasgos que se han fortalecido a lo largo de esta experiencia.
- Gestión de calidad. La implementación de modelos y sistemas de calidad en la aplicación web ha afianzado el compromiso con la excelencia. Asegurar la eficacia y eficiencia en la gestión de inventarios se ha vuelto una parte fundamental del enfoque profesional.
- Habilidades de investigación. La constante búsqueda de mejoras y la aplicación del método científico en la toma de decisiones han fomentado las habilidades de investigación. Estar al tanto de las últimas tendencias y adaptar mi enfoque según los resultados de la investigación son aspectos claves del desarrollo.

Competencias específicas:

Las competencias específicas adquiridas a lo largo del proyecto, han proporcionaron un conjunto sólido de habilidades tanto técnicas como de gestión, fundamentales para el desarrollo y éxito de proyectos tecnológicos complejos.

- Desarrollo de Interfaz de Usuario. Se diseñó e implementó una interfaz de usuario que simplifica la gestión de inventarios para los productores de

manzana en Teziutlán. Se adquirieron habilidades en diseño web centrado en el usuario, asegurando una experiencia intuitiva y eficiente.

- Integración de Sistemas de Gestión de Inventario. Se logró integrar un sistema de gestión de inventario, garantizando que la información de productos esté siempre actualizada y accesible. Esta competencia llevó a entender la importancia de la sincronización y la coherencia de datos en aplicaciones web.
- Capacitación del Personal. Se desarrollaron habilidades para capacitar al personal en el manejo de la aplicación, desde la navegación básica hasta la resolución de problemas. Esto implicó comunicar conceptos tecnológicos de manera clara y facilitar la adopción efectiva de la herramienta.
- Gestión de Proyectos Tecnológicos. Se realizó el proyecto desde la concepción hasta la implementación, aprendiendo a planificar, asignar recursos y gestionar el cronograma. Esto abarcó desde la toma de decisiones estratégicas hasta la supervisión del progreso del proyecto.
- Solución de Problemas Tecnológicos. Se desarrollaron habilidades para abordar desafíos tecnológicos específicos durante la creación de la aplicación. Esto incluyó resolver problemas de codificación, integración de sistemas y asegurar la funcionalidad constante de la aplicación.
- Diseño de aplicaciones web. Se adquirió el conocimiento para crear diagramas de secuencia, casos de uso y otros diagramas UML para visualizar y comunicar de manera efectiva la arquitectura y funcionalidad de la aplicación. Esta competencia fortaleció la capacidad para planificar y diseñar sistemas de manera estructurada y comprensible para los demás miembros del equipo.
- Gestión de Base de Datos. Se desarrollaron competencias avanzadas en el manejo de bases de datos, específicamente MySQL, para asegurar la integridad y eficiencia en la gestión de inventarios. Esto incluyó diseño de esquemas, consultas complejas y optimización del rendimiento de consultas, ampliando mi conocimiento en el manejo efectivo de datos.

- Dominio de Slim con PHP. Se adquirió un profundo conocimiento en el manejo de Slim, un microframework de PHP, al desarrollar la aplicación web. Esto incluyó la configuración de rutas, el manejo de middleware y la optimización del rendimiento, mejorando así las habilidades en el desarrollo de aplicaciones PHP eficientes y escalables.

CAPÍTULO VII
FUENTES DE INFORMACIÓN

- Corella Parra, L. M., & Olea Miranda, J. (2023). *Desarrollo de un sistema de control de inventario para una empresa comercializadora de sistemas de riego*. Ingeniería Investigación y Tecnología, 24(01), pp. 1-10. https://doi.org/10.22201/fi.25940732e.2023.24.1.006
- González, A. (2020). *Un modelo de gestión de inventarios basado en estrategia competitiva*. Ingeniare. Revista chilena de ingeniería, 28(1), pp. 133-142. https://dx.doi.org/10.4067/S0718-33052020000100133
- Arroba Salto, J. E., Angulo Rosales, Y. A., & Naula Valla, S. M. (2018). *Control de inventarios y su incidencia en los estados financieros*. Revista Observatorio de la Economía Latinoamericana, (noviembre 2018), pp. 2-3. Recuperado de: https://www.eumed.net/rev/oel/2018/11/inventarios-estados-financieros.html
- Andrade, C. M., & Ayaviri, D. (2018). *Demand and Consumption of Organic Products in the Riobamba Cantón, Ecuador*. Información tecnológica, 29(4), pp. 217-226. https://dx.doi.org/10.4067/S0718-07642018000400217
- Godin, S. (2019). "*Esto es marketing*". España: Editorial Planeta, S.A.
- Selman, H. (2017). "*Marketing digital*". Editorial Ibukku.
- Vásquez, G. (2009). *Los canales de distribución y el valor para el consumidor*. Temas de management, 2, pp. 10-16. Recuperado de: https://ucema.edu.ar/cimei-base/download/research/71_Vasquez.pdf
- Molina Ríos, J. R., Zea Ordóñez, M. P., Contento Segarra, M. J., & García Zerda, F. G. (2017). *Estado del arte: Metodologías de desarrollo en aplicaciones web*. 3C Tecnología: glosas de innovación aplicadas a la pyme, 6(3), pp. 54-71. http://dx.doi.org/10.17993/3ctecno.2017.v6n3e23.54-71
- Beynon-Davies, P. (2018). "*Sistemas de bases de datos*". Editorial Reverté S. A.
- Oracle. (s. f.). "*Capítulo 1: Información general".* Noviembre de 2023, de MySQL: https://dev.mysql.com/doc/refman/8.2/en/introduction.html

- Peña Cáceres, O. J. M., More More, M. A., Cornejo Sojo, R. E., & Garay Silupu, E. R. (2022). *Distanciamiento social ante la COVID-19: Simulación del aforo máximo de personas mediante PHP*. Ingenius. Revista de Ciencia y Tecnología, (27), pp. 9-16. https://doi.org/10.17163/ings.n27.2022.01
- Schaf, J. (2019). *The meaning of motions and their effects in Einstein's empty space and in the scenario of the Higgs quantum fluid space*. Journal of Modern Physics, 10(03), pp. 256–280. https://doi.org/10.4236/jmp.2019.103018
- Sunardi, A., & Suharjito. (2019). *MVC Architecture: A Comparative Study Between Laravel Framework and Slim Framework in Freelancer Project Monitoring System Web Based*. Procedia Computer Science, 157, pp. 134-141. https://doi.org/10.1016/j.procs.2019.08.150
- Darmawan, I., Karim, A. P. A., Rahmatulloh, A., Gunawan, R., & Pramesti, D. (2021). *JSON Web Token Penetration Testing on Cookie Storage with CSRF Techniques*. IEEE Xplore, pp. 1-5. https://doi.org/10.1109/ICADEIS52521.2021.9701965
- Thompson, E. (2005). *MD5 collisions and the impact on computer forensics*. Digital investigation, 2(1), pp. 36-40. https://doi.org/10.1016/j.diin.2005.01.004
- Llerena Ocaña, L. A., Fernández Villacres, G. E., Viscaino Naranjo, F. A., & Baño Naranjo, F. P. (2021). *Frameworks basados en typescript para el desarrollo de aplicaciones web interactivas*. Dilemas contemporáneos: educación, política y valores, 8(3), 23. https://doi.org/10.46377/dilemas.v8i3.2644
- Salazar, L. K., & Soles, N. A. M. (2021). *Sistema web utilizando el framework angular para el control de inventario en empresas gastronómicas* [Tesis para obtener título profesional, Universidad César Vallejo]. Repositorio de la Universidad César Vallejo. https://hdl.handle.net/20.500.12692/80013

- Hernández González, A., Enríquez Hernández, D. M., & Ruano Chichatskaia, A. (2021). *De modelos de negocio a requisitos del software*. Ingeniería Industrial, 42(2), pp. 32-59. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59362021000200032&lng=es&tlng=es
- Vidal Silva, C. L., Villarroel, R. H., López Cortés, X. A., & Rubio, J. M. (2019). *Una Propuesta de Algoritmo Spin / Promela para el Análisis y Diagnóstico de Errores en Diagramas de Secuencia UML*. Información tecnológica, 30(1), pp. 263-272. https://dx.doi.org/10.4067/S0718-07642019000100263
- Larman," *C. UML y Patrones*". Madrid, Pearson Educación. 624p. 2003.
- Microsoft. (s. f.). "Getting Started". Noviembre 2023, de Código de estudio visual: https://code.visualstudio.com/docs
- Arsaute, A., Zorzán, F. A., Daniele, M., González, A., & Frutos, M. (2018). "*Generación automática de API REST a partir de API Java, basada en transformación de Modelos (MDD)*". In XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste). http://sedici.unlp.edu.ar/handle/10915/67777
- Hyams, R., & Pilko, T. (2022). "*You could use the API!": A Crash Course in Working with the Alma APIs using Postman*. Code4Lib Journal, (54). https://journal.code4lib.org/articles/16597?utm_source=rss&utm_medium=rss&utm_campaign=you-could-use-the-api-a-crash-course-in-working-with-the-alma-apis-using-postman
- Instituto Nacional de Estadística y Geografía (INEGI). (2017). "*Banco de indicadores*". Noviembre 2023, de INEGI: https://www.inegi.org.mx/app/indicadores/?t=15&ag=21174#D15#D5300000012#D1009000050#D1009000050#D1009000050#D1009000050

CAPÍTULO VIII

ANEXOS

Anexo A. Dictamen

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico Superior de Teziutlán

Asunto: **Asignación de Asesor(a), Comisión Revisora, Entrega de Trabajo Profesional y Dictamen**

Teziutlán, Puebla, 03/enero/2024

Asesor(a): **MARIA EUGENIA CARREON ROMERO**
Integrante de Comisión Revisora: **ROMAN GUTIERREZ MARCOS**
Integrante de Comisión Revisora: **GABRIEL BAUTISTA ORONZOR**
Presentes

Por este medio me permito informar que ha sido asignado como asesor(a) y comisión revisora del trabajo profesional que se convertirá en Tesis de:

Alumno(a): **HERNANDEZ TOLEDANO ERICK DE JESUS**
Apellido paterno/materno/nombre (s)

Número de Control: **19TE0561** Licenciatura o Posgrado: **INGENIERIA EN SISTEMAS COMPUTACIONALES**

Plan: **2010** Correo Electrónico: **L19TE0561@TEZIUTLAN.TECNM.MX**

Cuyo tema es: **DESARROLLO DE APLICACION WEB PARA LA OPTIMIZACION DE INVENTARIOS Y GESTION DE STOCK DE PRODUCTOS ORGANICOS DEL COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLAN (CPMT)**

25 palabras (máximo)

Se ha enviado a su correo institucional el trabajo profesional o de grado, por lo cual la comisión revisora tendrá 5 día há biles para realizar las observaciones al alumnado, el(la) interesado(a) tendrá igualmente **5 días** para corregir y las enviará al correo electrónico institucional de la comisión revisora, agradezco de antemano su valioso apoyo en esta importante actividad para la formación profesional de licenciatura o de posgrado de nuestro alumnado egresado.

Dictamen de Comisión Revisora, Aprobación de Impresión o Grabación y Autorización para subirla al Repositorio del TecNM

Siendo el día: 24 de enero de 2024 se reunieron los miembros de la comisión para revisar el trabajo asignado y una vez analizado se decidió liberarlo y aprobarlo para su grabación y programación de examen profesional.

MARIA EUGENIA CARREON ROMERO
Nombre y Firma del(la) Asesor(a)

ROMAN GUTIERREZ MARCOS
Nombre y Firma de Integrante de la Comisión Revisora

GABRIEL BAUTISTA ORONZOR
Nombre y Firma de Integrante de la Comisión Revisora

MYRIAM SÁNCHEZ PEREZ
Subdirección Académica

S.E.P. GOBIERNO DEL ESTADO

INSTITUTO TECNOLÓGICO SUPERIOR DE TEZIUTLÁN SUBDIRECCIÓN ACADÉMICA

R08/06/2023
Folio:17

F-SAC-18

TEC de Teziutlán | 100% PLASTIC | PUEBLA

Fracción I y II s/n Aire Libre, Teziutlán, Puebla, C.P. 73960 Tels. 231 311 4000 / 4001 / 4002 / 4003
e-mail: itsteziutlan@hotmail.com | www.teziutlan.tecnm.mx

2024 Felipe Carrillo PUERTO

Anexo B. Carta de Autorización, Consulta y Publicación

Tecnológico Nacional de México
Instituto Tecnológico Superior de Teziutlán

CARTA DE AUTORIZACIÓN DEL(LA) AUTOR(A) PARA LA CONSULTA Y PUBLICACIÓN ELECTRÓNICA DEL TRABAJO DE INVESTIGACIÓN

El que suscribe:

ERICK DE JESÚS **HERNÁNDEZ** **TOLEDANO**

Con Número de Control: **19TE0561**

Perteneciente al Programa Educativo: **INGENIERÍA EN SISTEMAS COMPUTACIONALES**

Por este conducto me permito informar que he dado mi autorización para la consulta y publicación electrónica del trabajo de investigación en los repositorios académicos.

Registrado con el producto: **TESIS**

Cuyo Tema es:

DESARROLLO DE APLICACIÓN WEB PARA LA OPTIMIZACIÓN DE INVENTARIOS Y GESTIÓN DE STOCK DE PRODUCTOS ORGÁNICOS DEL COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT)

Correspondiente al periodo:

AGOSTO-DICIEMBRE 2023

Y cuyo(a) director(a) de tesis es:

MARÍA EUGENIA CARREÓN ROMERO

ATENTAMENTE

ERICK DE JESÚS HERNÁNDEZ TOLEDANO

Nombre y firma

Fecha de emisión: **29/01/2024**
c.c.p. Subdirección Académica

Anexo C. Licencia de uso

LICENCIA DE USO OTORGADA POR **Erick de Jesús Hernández Toledano**, de nacionalidad **Mexicana,** mayor de edad, con domicilio ubicado en la calle **Morelos s/n, localidad de Otra Banda en la Ciudad de Tlapacoyan, Veracruz**, en mi calidad de titular de los derechos patrimoniales y morales y autor(a) de la tesis denominada **"DESARROLLO DE APLICACIÓN WEB PARA LA OPTIMIZACIÓN DE INVENTARIOS Y GESTIÓN DE STOCK DE PRODUCTOS ORGÁNICOS DEL COLECTIVO DE PRODUCTORES DE MANZANA TEZIUTLÁN (CPMT)"** en adelante "LA OBRA" quien para todos los fines del presente documento se denominará "EL(LA) AUTOR(A) Y/O EL(LA) TITULAR", a favor del Instituto Tecnológico de **Superior de Teziutlán** del Tecnológico Nacional de México, la cual se regirá por las cláusulas siguientes:

PRIMERA –OBJETO: "EL(LA) AUTOR(A) Y/O EL(LA) TITULAR", mediante el presente documento otorga al Instituto Tecnológico **de Superior de Teziutlán** del Tecnológico Nacional de México, licencia de uso gratuita e indefinida respecto de "LA OBRA", para almacenar, preservar, publicar, reproducir y/o divulgar la misma, con fines académicos, por cualquier medio en forma física y a través del repositorio institucional y del repositorio nacional, éste último consultable en la página: (https://www.repositorionacionalcti.mx/).

SEGUNDA - TERRITORIO: La presente licencia se otorga, de manera no exclusiva, sin limitación geográfica o territorial alguna, de manera gratuita e indefinida.

TERCERA -ALCANCE: La presente licencia contempla la autorización para formato uso de "LA OBRA" en cualquier formato o soporte material y se extiende a la utilización, de manera enunciativa más no limitativa a los siguientes medios: óptico, magnético, electrónico, virtual (en red), mensaje de datos o similar, conocido o por conocerse.

CUARTA – EXCLUSIVIDAD: La presente licencia de uso aquí establecida no implica exclusividad en favor del Instituto Tecnológico de **Superior de Teziutlán**; por lo tanto, "EL(LA) AUTOR(A) Y/O EL(LA) TITULAR" conserva los derechos patrimoniales y morales de "LA OBRA", objeto del presente documento.

QUINTA – CRÉDITOS: El Instituto Tecnológico de **Superior de Teziutlán** y/o el Tecnológico Nacional de México reconoce que "EL(LA) AUTOR(A) Y/O EL(LA) TITULAR" es el(la) único(a), primigenio(a) y perpetuo(a) titular de los derechos morales sobre "LA OBRA"; por lo tanto, siempre deberá otorgarle los créditos correspondientes por la autoría de la misma.

SEXTA – AUTORÍA: "EL(LA) AUTOR(A) Y/O EL(LA) TITULAR" manifiesta ser el(la) único(a) titular de los derechos de autor que derivan de "LA OBRA" y declara que el material objeto del presente fue realizado por él(ella), sin violentar o usurpar derechos de propiedad intelectual de terceros; por lo tanto, en caso de controversia sobre los mismos, se obliga a ser el(la) único(a) responsable.
Dado en la Ciudad de **Teziutlán, Puebla**, a los **quince** días del mes **enero** de 2024.

"EL(LA) AUTOR(A) Y/O EL(LA) TITULAR",	"EL INSTITUTO TECNOLÓGICO **SUPERIOR DE TEZIUTLÁN"**
Erick de Jesús Hernández Toledano Nombre y Firma	**Arminda Juárez Arroyo** Directora General Nombre, Firma y Sello

Índice de figuras

Índice de tablas

Índice de gráficos