SEP TecNM

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE ACAPULCO

TEMA:

SISTEMA DE GESTIÓN DE ORDENES (APP) PARA RESTAURANTES

OPCIÓN I:
TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:
MAESTRÍA EN SISTEMAS COMPUTACIONALES

PRESENTAN:
JORGE JIMÉNEZ CASTAÑÓN

DIRECTOR DE TESIS:
MTI. RAFAEL HERNÁNDEZ REYNA

CO-DIRECTOR DE TESIS
MTI. JUAN MIGUEL HERÁNDEZ BRAVO

Acapulco, Gro, MAYO 2018

Índice

Introducción ---------- 1
Delimitación ---------- 3
Capítulo 1 Antecedentes. ---------- 4
1.1.- Antecedentes del Problema a Resolver. ---------- 4
1.2.- Planteamiento del Problema. ---------- 7
1.3.- Objetivo General. ---------- 11
1.4.- Objetivos Específicos. ---------- 11
1.5.- Justificación ---------- 12
1.6.- Hipótesis ---------- 13
Capítulo 2.- Estado del Arte. ---------- 14
2.1.- Importancia de las Tecnologías Móviles. ---------- 14
2.2.- La Tecnología y la Industria de la Comida. ---------- 16
2.3.- Software para Restaurante. ---------- 23
2.3.1.- Soft Restaurant. ---------- 24
2.3.1.1.-Módulos. ---------- 25
2.3.1.2.- Punto de Venta. ---------- 25
2.3.1.3.- Inventarios. ---------- 26
2.3.1.4.- Administración y Seguridad. ---------- 26
Capítulo 3.- Conceptos Teóricos. ---------- 28
3.1.- Java. ---------- 28
3.1.1.- La Seguridad del Lenguaje Java. ---------- 29
3.1.2.- Portabilidad de Java. ---------- 30
3.1.3.- Java en la Actualidad. ---------- 31
3.2.- Ambientes de Desarrollo. ---------- 31
3.2.1.- NetBeans. ---------- 32
3.2.2.- NetBeans IDE. ---------- 32
3.3.- Computación Móvil. ---------- 33
3.4.- Evolución de la Computación Móvil. ---------- 35
3.5.- Tipos de Dispositivos Móviles. ---------- 36
3.5.1.- Tipos de Dispositivos Móviles. ---------- 36
3.5.2.- PDA's (Personal Digital Assintant). ---------- 37
3.5.3.- Smartphone o Teléfono Inteligente. ---------- 37
3.6.- Sistemas Operativos. ---------- 38

3.6.1.- Sistemas Operativos para Dispositivos Móviles. ---- 38
3.6.2.- Sistema Operativo Android. ---- 40
3.6.3.- Aplicación Móvil. ---- 43
3.6.4.- Aplicación Web. ---- 45
3.6.5.- Aplicación Nativa. ---- 45
3.7.- Android Studio. ---- 46
3.7.1.- Depuración Integrada. ---- 47
3.7.2.- Estructura de un Proyecto. ---- 47
3.7.3.- Sistema de Compilación Gradle. ---- 49
3.7.4.- Módulos. ---- 50
3.7.5.- Modulo de App para Android. ---- 50
3.7.6.- Modulo de Biblioteca. ---- 51
3.8.- WAMPSERVER. ---- 52
Capítulo 4.- Metodología. ---- 53
4.1.- Planeación. ---- 53
4.2.- Metodología a Utilizar. ---- 54
4.2.1.- Etapas de Scrum. ---- 56
4.2.1.1.- Análisis. ---- 57
4.2.1.2.- Diseño. ---- 58
4.2.1.3.- Desarrollo. ---- 59
4.2.1.4.- Pruebas de Funcionamiento. ---- 59
4.2.1.5.- Entrega. ---- 60
4.3.- Viabilidad. ---- 61
4.3.1.- Requerimientos de Sistema. ---- 61
4.3.1.1.- Requerimientos Funcionales. ---- 62
4.3.1.2.- Requerimientos de Datos. ---- 63
4.3.1.3.- Requerimientos de rendimiento. ---- 64
4.3.1.4.- Restricciones. ---- 65
4.3.2.- Modelado. ---- 65
4.3.2.1.- Análisis de caso de uso. ---- 65
4.3.2.2.- Diagramas de Casos de uso. ---- 66
4.3.2.3.- Diagrama de Clases. ---- 74
4.3.2.4.- Diagrama de Secuencia. ---- 77
4.3.3.- Modelo Entidad-Relación. ---- 86
4.3.3.1.- Diccionario de Datos. ---- 86
Capítulo 5.- Implementación. ---- 92

5.1.- Conexión a la base de datos.---92
5.2.- Acceso.---94
5.3.- Ventana Principal.---97
5.4.- Gestión de locales. ---99
5.4.1.- Ingresar datos. --- 100
5.4.2.- Modificar Datos.--- 103
5.4.3.- Eliminar datos.--- 106
5.5.- Gestión de Departamento. --- 108
5.5.1.- Ingresar Datos.--- 110
5.5.2.- Modificar Datos.--- 112
5.5.3.- Eliminar Datos. --- 114
5.6.- Gestión de Empleados. --- 116
5.6.1.- Ingresar datos. --- 118
5.6.2.- Modificar datos. --- 122
5.6.3.- Eliminar datos.--- 125
5.7.- Menús de Platillos y Bebidas. --- 127
5.7.1.- Ingresar Datos.--- 130
5.7.2.- Modificar Datos.--- 134
5.7.3.- Eliminar Datos. --- 136
5.8.-Modo Monitor. --- 138
5.8.1.- Ingreso de datos.--- 141
5.8.2.- Modificar Datos.--- 147
5.8.3.- Eliminar Datos. --- 148
5.9.- Módulo móvil.--- 150
5.9.1.- Inicio de sesión. --- 153
5.9.2.- Menús--- 154
Conclusiones.--- 158
Anexos. --- 160
Aseguramiento Técnico Material--- 160
Costo de Desarrollo--- 160
Costo de Implementación--- 162
Bibliografía.--- 163

Índice de Figuras.

Figura 2.1.- Diagrama de Caso de uso [Gupta, 2012a]. ---- 20
Figura 2.2.- Framework de usabilidad (ISO 9241-11) [Isabel, 2013]. ---- 22
Figura 3.1.- Capas del Sistema Android [Álvaro Zapata, 2012a]. ---- 42
Figura 3.2.- Lenguajes utilizados para desarrollo de aplicaciones. ---- 44
Figura 4.1.- Ciclo de desarrollo ágil [Gallego, 2012a]. ---- 55
Figura 4.2.- Etapas de la metodología [Maira Cecilia Gasca Mantilla, 2013a]. ---- 56
Figura 4.3.- Diagrama de caso de uso para el inicio de sesión. ---- 66
Figura 4.4.- Caso de uso para la gestión de seguridad. ---- 67
Figura 4.5.- Caso de uso para la gestión de menús. ---- 69
Figura 4.6.- Caso de uso de gestión de órdenes operador-cocinero. ---- 70
Figura 4.7.- Caso de uso de gestión de órdenes Administrador-Supervisor. ---- 72
Figura 4.8.- Diagrama de clases. ---- 74
Figura 4.9.- Despliegue del acceso del Administrador al sistema. ---- 77
Figura 4.10.- Despliegue del ingreso de usuario Cocina al sistema. ---- 78
Figura 4.11.- Despliegue de la Gestión de locales. ---- 78
Figura 4.12.- Despliegue de la gestión de departamentos. ---- 79
Figura 4.13.- Despliegue de la gestión de empleados. ---- 80
Figura 4.14.- Despliegue de la gestión de platillos. ---- 81
Figura 4.15.- Despliegue de la gestión de bebidas. ---- 83
Figura 4.16.- Despliegue de la generación de órdenes para Administrador. ---- 83
Figura 4.17.- Despliegue de la modificación de órdenes en usuario Cocina. ---- 85
Figura 4.18.- Diagrama entidad relación de la base de datos. ---- 86
Figura 5.1.- Creación de la clase conexión. ---- 92
Figura 5.2.- Adición a las librerías del conector a la base de datos. ---- 93
Figura 5.3.- ventana para el acceso al sistema. ---- 94
Figura 5.4.- Escritorio del sistema de gestión. ---- 98
Figura 5.5.- Menú archivo. ---- 98
Figura 5.6.- Menú Administración. ---- 98
Figura 5.7.- Menú de Operación. ---- 98
Figura 5.8.- Interfaz Gráfica de la ventana local. ---- 100
Figura 5.9.- Tabla Local de la base de datos. ---- 100
Figura 5.10.- Guardar una nueva fila en la tabla local desde el sistema. ---- 103
Figura 5.11.- Tabla local con nueva fila insertada. ---- 103
Figura 5.12.- Selección de fila en el formulario. ---- 104
Figura 5.13.- Modificación de la columna Sucursal en la base de datos. ---- 105
Figura 5.14.- Tabla local modificada. ---- 106
Figura 5.15.- Fila eliminada desde el formulario. ---- 107
Figura 5.16.- Mensaje de sistema. ---- 107
Figura 5.17.- Tabla local después de eliminar una fila. ---- 108
Figura 5.18.- Formulario de gestión de Departamentos. ---- 109
Figura 5.19.- Tabla Departamento. ---- 110
Figura 5.20.- Fila agregada en formulario Departamento. ---- 111
Figura 5.21.- Fila nueva en tabla departamento. ---- 111
Figura 5.22.- Obtención de datos de la JTable del formulario Departamento. ---- 112

Figura 5.23.-Fila 4 modificada en formulario departamento.---------- 113
Figura 5.24.- Contenido de la tabla departamento en la base de datos.---------- 113
Figura 5.25.- Fila 4 eliminada del formulario Departamento. ---------- 115
Figura 5.26.- Mensaje de error del formulario.---------- 115
Figura 5.27.- Fila eliminada de la tabla departamento.---------- 115
Figura 5.28.- Tabla empleado. ---------- 117
Figura 5.29.- Vista v_empleados. ---------- 118
Figura 5.30.- Formulario Usuarios. ---------- 118
Figura 5.31.- Items del JComboBox CB_LOCAL. ---------- 119
Figura 5.32.- Fila 6 agregada al formulario Usuarios.---------- 121
Figura 5.33.- Fila insertada en la tabla empleado de la base de datos.---------- 122
Figura 5.34.- Selección de filas en formulario Usuarios. ---------- 123
Figura 5.35.- Fila 3 del formulario Usuario modificada.---------- 125
Figura 5.36.- Tabla empleado de la base de datos modificada.---------- 125
Figura 5.37.- Filas 4 y 5 del Formulario Usuarios.---------- 126
Figura 5.38.- Tabla empleado después de la eliminación de filas.---------- 127
Figura 5.39.- Interfaz gráfica del formulario Platillos. ---------- 129
Figura 5.40.- Tabla platillos.---------- 130
Figura 5.41.- Cuadro de dialogo de formulario Platillos. ---------- 131
Figura 5.42.- Imagen visualizada en formulario. ---------- 132
Figura 5.43.- Ingreso de fila en el formulario Platillos. ---------- 133
Figura 5.44.- Tabla platillos con fila ingresada.---------- 134
Figura 5.45.- Formulario Platillos con fila 08 modificada.---------- 136
Figura 5.46.- Tabla platillos modificada. ---------- 136
Figura 5.47.- Formulario Platillos después de eliminar la fila 08.---------- 137
Figura 5.48.- Tabla platillos después de la eliminación de la fila 08.---------- 138
Figura 5.49.- Formulario Monitor. ---------- 140
Figura 5.50.- Comparación de la tabla orden (arriba y v_orden (abajo) en la base de datos. --------- 141
Figura 5.51.- Formulario Monitor con la fila 65 modificada.---------- 148
Figura 5.52.- Estado de la orden 65 en la base de datos modificada. ---------- 148
Figura 5.53.- Fila 65 eliminada del formulario Monitor. ---------- 149
Figura 5.54.- Orden 65 eliminada de la tabla orden.---------- 149
Figura 5.55.- Archivos PHP en carpeta TESIS.---------- 150
Figura5.56.- Layout inicio.---------- 152
Figura 5.57.- Layout Menu.---------- 156

Índice de Tablas.

Tabla 3.1.- Sistemas Operativos Móviles [Ing. Pedro Julio Colorado Ángel, 2015a].----------41
Tabla 4.1.- Descripción de caso de uso inicio de sesión. ----------67
Tabla 4.2.- Descripción de caso de uso gestión de seguridad.----------68
Tabla 4.3.- Descripción de caso de uso gestión de menús.----------69
Tabla 4.4 .- Descripción de caso de uso gestión de órdenes operador-cocinero.----------71
Tabla 4.5.- Descripción de Caso de uso de gestión de órdenes Administrador-Supervisor.----------73
Tabla 4.6.- Tabla Local detallada.----------87
Tabla 4.7.- Tabla Departamento detallada.----------87
Tabla 4.8.- Tabla Empleado detallada.----------88
Tabla 4.9.- Relaciones de la tabla empleado.----------89
Tabla 4.10.- Tabla Bebida detallada. ----------89
Tabla 4.11.- Tabla Platillo detallada. ----------90
Tabla 4.12.- Tabla Orden detallada. ----------91
Tabla 4.13.- Relaciones de la tabla orden.----------91
Tabla 5.1.- Elementos que componen la interfaz gráfica del acceso. ----------94
Tabla 5.2.- Elementos de la ventana principal.----------97
Tabla 5.3.- Elementos del formulario Local. ----------99
Tabla 5.4.- Elementos del formulario Departamento. ---------- 109
Tabla 5.5.- Elementos del Formulario Empleados.---------- 116
Tabla 5.6.- Elementos del Formulario Platillos. ---------- 128
Tabla 5.7.- Elementos del formulario Monitor. ---------- 139
Tabla 5.8.- Elementos de Layout Inicio.---------- 152
Tabla 5.9.- Elementos de Layout Menu.---------- 155
Tabla A.0.1.- Costos de Desarrollo.---------- 161
Tabla A.0.2.- Costos de Implementación. ---------- 162

Introducción

La situación que habitualmente se puede ver en un restaurante durante su operación diaria, es el manejo de los pedidos, tiempos de espera, facturación correcta entre otros, en la mayoría de los casos no es la ideal, esto resulta en algunas ocasiones en un mal servicio prestado a los clientes, sobre todo durante los momentos de mayor afluencia durante la jornada diaria.

Dicha operación requiere el desplazamiento del personal de piso de venta de un lugar a otro, un gran número de veces para realizar una determinada operación, ya sea en punto de venta o en el piso de venta, lo que conlleva a algunos errores en la operación o un considerable tiempo de espera el cual puede resultar algo molesto para algunos clientes.

Para el establecimiento esto es un aspecto negativo en la operación de piso de venta, que da como resultado, la generación de merma que representa pérdidas para el establecimiento y lo más importante, una percepción negativa por parte del cliente. De tal forma que algunos establecimientos determinan que es necesario reducir los errores cometidos durante la operación, lo cual representa el principal problema en piso de ventas durante las horas pico y los periodos de mayor carga en los periodos vacacionales.

Por lo que dichos establecimientos buscan soluciones; ya sea modificando el proceso de operación para ofrecer mayor comodidad al cliente, la contratación de personal para reducir la carga de trabajo en el área de piso de venta y cocina, o la búsqueda de soluciones tecnológicas que permitan llevar un registro de las actividades de operación.

Por otro lado hay establecimientos que llegan a la misma conclusión pero ya sea por cuestiones financieras o por simple falta de conocimiento o experiencia no buscan una solución al problema y presentan en algunas ocasiones una resistencia al cambio, lo que termina en la concurrencia de los errores cometidos y en algunas ocasiones en un aumento en las pérdidas o la disminución de clientes.

En el presente documento se elaborará un proyecto de trabajo en el que se planea desarrollar un sistema móvil (app) el cual se pueda ejecutar en prácticamente cualquier dispositivo móvil basado en el sistema operativo Android, que permite la administración de un restaurante, utilizando herramientas de programación y modelado de base de datos, así como para que sea accesible para cualquier tipo de establecimiento.

El sistema proyectará una buena imagen del establecimiento, así como la reducción de los errores durante la operación de piso de venta y evitar la mala publicidad o reseñas debido a la mala percepción del servicio por parte de los clientes, consecuencia de la relativa demora que el cliente percibe, así como la carga de trabajo que demora al personal que habitualmente comete errores en las tomas de las comandas.

Delimitación

Con el presente proyecto se pretende el desarrollar un sistema de gestión de ordenes para restaurantes, que a diferencia de los sistemas estáticos, pueda manejar los procesos que intervienen en la operación de piso de venta de manera directa sin la necesidad de desplazarse para la entrega del pedido al área de preparación o estación de pedido/punto de venta, dejando al área de cocina que maneje sus propios tiempos de preparación.

Capítulo 1 Antecedentes.

1.1.- Antecedentes del Problema a Resolver.

La operación de un establecimiento dedicado a la preparación de alimentos y bebidas que cuenta con servicio de meseros en su piso de venta siempre debe tomar en cuenta la satisfacción del cliente ya que de esto depende el que dicho cliente regrese una vez más o que recomiende a otra persona el acudir a dicho establecimiento.

En un principio el servicio de atención al cliente en un establecimiento es el servicio de recepción, si no se cuenta con este, entonces la responsabilidad recae en el servicio de meseros, estos son la primera impresión para el cliente. Las tareas de un recepcionista no se limitan a recibir y acomodar a los clientes, sino resolver cualquier otra necesidad del cliente, como r ejemplo; preguntas sobre el establecimiento, la zona, incluso el menú.

No todos los establecimientos cuentan con el servicio de recepción, pero dentro del ámbito hotelero el servicio de recepción es más que indispensable, por el hecho de que este es parte fundamental dentro de la operación, por lo que son los responsables de realizar las reservaciones para ingresar en los restaurantes que se encuentran en el interior de las instalaciones.

En algunos casos no es necesaria la reservación durante los horarios de desayunos y almuerzos, pero durante las horas de comida y cena si son necesarios. En el caso de los establecimientos que se encuentran fuera de las instalaciones de los hoteles, solo unos pocos cuentan con el sistema de reservaciones, por lo general son los restaurantes con servicio a la carta, ya que para comodidad de los clientes realizan las reservaciones de manera escalonada esto es para evitar

los tiempos de espera y en caso de haberlas evitar las filas para brindar una mayor comodidad a los clientes.

Para los establecimientos que no cuentan con el servicio de reservaciones o recepción se atiende a los clientes conforme arriban y si es que se cuenta con el espacio dentro de las instalaciones.

Todo lo anterior mencionado es para brindar mayor comodidad y brindar una mejor percepción del servicio por parte del cliente y es parte del servicio que se brinda, una vez el cliente haya ingresado a las instalaciones es decir el piso de venta, es cuando el personal de esta área inicia sus operaciones, el cual en esencia debería ser para dar la mayor comodidad al cliente durante su estancia y promover a que regrese o recomiende el establecimiento, lo que ayudaría a incrementar los ingresos.

Pero debido a varios detalles dentro de la operación, se cometen errores y termina siendo todo lo contrario, que sería el brindar un mal servicio y posiblemente el que los clientes no deseen volver al establecimiento, por lo que la mayoría de los establecimientos buscan el reducir estos errores lo mejor posible.

Para dicho propósito algunos establecimientos diseñan protocolos específicos que seguir durante ciertas condiciones en la operación de piso de venta con el propósito de agilizar el servicio y brindar una mayor comodidad al cliente durante su estancia, lamentablemente esto no es suficiente, ya que los errores continúan ocurriendo durante la operación en horas pico, especialmente durante los periodos vacacionales que es cuando la carga de trabajo es mayor.

En esos casos los establecimientos se preparan contratando y capacitando nuevo personal antes del inicio de dichos periodos, por lo que la carga de trabajo se reduce para el personal de piso de venta y el área de cocina, esto permitía reducir la concurrencia de los errores, pero lamentablemente no los evita completamente.

En algunos casos las cadenas de restaurantes o los establecimientos independientes que pueden costearlo buscan soluciones en las nuevas tecnologías que hoy en día permite el brindar mayor comodidad para los clientes dentro de las instalaciones, al contar con servicios que hoy en día se podrían considerar como básicos, un ejemplo de estos seria el servicio de internet para los clientes.

Ahora, del lado del personal también se implementan tecnologías de manera que se pueda facilitar el trabajo que realizan durante la operación, esto se logra mediante el uso de estas herramientas que años atrás no estaban disponibles, y hoy en día son indispensables para la operación de los establecimientos, algunos optan solo por un sistema de punto de venta que permite el registro de las transacciones y la facturación.

Otra opción por la que se inclinan los establecimientos son los sistemas de gestión de pedido que permitan reducir los errores en piso de venta, así como un registro de las comandas realizadas, estos sistemas pueden ser estáticos o móviles.

La gran mayoría elige el uso de los sistemas estáticos, los cuales suelen ser más accesibles que los sistemas móviles, por lo que el número de las estaciones varía, dependiendo del tamaño del piso de venta y el número de mesas con las que se cuenta, las estaciones son ubicadas en áreas clave del piso de venta.

Para maximizar su eficacia se establece a cierto número del personal para uso de una determinada estación en una pequeña área asignada, lo que permite una mejor operación en el área donde está asignada la operación.

En otros casos se opta por el mismo tipo de sistema el cual se implementa en el área del punto de venta donde todo el personal de piso de venta tiene que trasladarse para el realizar el pedido de la comanda. Lo que termina por retrasar el cobro de las transacciones realizadas.

Básicamente la gran mayoría de los establecimientos opta por la versión estática por ser mucho más accesibles que los nuevos sistemas que se están utilizando actualmente.

1.2.- Planteamiento del Problema.

Durante la operación de un establecimiento dedicado al servicio de la preparación de alimentos y bebidas es inevitable el que haya pérdidas ocasionadas por la operación del piso de venta, las razones pueden ser; errores cometidos por el personal de piso de venta, errores del personal de cocina o la relativa tardanza que percibe el cliente al esperar la comanda que ordenó.

Estos errores son ocasionados por la saturación de las comandas que se realizan durante la operación del establecimiento, ya que algunos establecimientos se resisten al cambio y aun utilizan el sistema de lápiz y papel, lo que resulta en errores del personal de piso de venta al tomar la orden, esto debido a la prisa en atender al resto de los clientes dentro de su área de trabajo que le fue asignada.

La problemática continúa al área de cocina o bar, donde por la prisa del personal de piso de venta escriben las comandas con errores y el personal de la cocina al no ser consciente del error de los primeros toma la comanda como correcta y procede a la elaboración, en caso de que la comanda no tenga errores pero por las prisas el personal de piso de venta escribe la comanda con una letra ilegible, lo que provoca una mala interpretación de la comanda y produce errores en la elaboración de las comandas, terminando con la insatisfacción de los clientes.

En cualquiera de los dos casos anteriores el cliente termina insatisfecho por el servicio provocando tres posibles resultados:

- El cliente se molesta y se retira de las instalaciones del establecimiento sin pagar la comanda.
- El cliente tiene paciencia y tolera el error.
- El cliente tiene paciencia y tolera el error, pero rechaza pagar esa comanda y solicita una nueva comanda.

En cualquiera de los casos el establecimiento genera pérdidas, en el primer y tercer caso, dependiendo del tipo de alimentos se podría entregar inmediatamente a otro cliente que espera la misma comanda, caso contrario la comanda se merma ya que no puede ser recalentada o refrigerada.

En el tercer caso, al generar nuevamente la comanda esta pasa a esperar su turno para ser elaborada ya que las comandas siempre deben de ser preparadas según su orden de llegada al área de preparación que le corresponda.

Con el segundo caso el cliente tolera el error, consume y paga su comanda, pero se retira insatisfecho y posiblemente no vuelva a consumir dentro de las instalaciones y es muy seguro que no recomiende el establecimiento a sus conocidos.

En otros establecimientos como los que se ubican dentro de las instalaciones de un hotel o centro turístico se opta por el uso de estaciones fijas para el uso del personal en el piso de venta de los establecimientos de alimentos y bebidas, colocándolas en áreas clave dentro del área que se asignan a los empleados del establecimiento, sin dejar el uso particular de una estación a algún miembro del personal, haciendo que cualquiera del personal pueda usar cualquiera de las estaciones de pedido que se encuentra más cercana.

Esto durante la operación del establecimiento produce que no haya un orden en el área de trabajo, por lo que los capitanes de meseros son los responsables de mantener el orden y resolver los problemas que se presenten con el personal o los clientes, pero, aún con un orden establecido durante el transcurso de una jornada tranquila el sistema de pedidos estático es muy eficiente, pero durante las jornadas más cargadas de trabajo o los periodos vacacionales se puede observar el cuello de botella que se produce por la saturación de las comandas.

Estos cuellos de botella se producen por el hecho de que el personal del área de venta tiene que tomar nota a mano de la comanda que el cliente realiza y se traslada a una de las estaciones para enviar su pedido al área que corresponda para su preparación, esto en un día tranquilo.

Durante las jornadas más laboriosas como las horas pico, fines de semana o periodos vacacionales el personal al trasladarse a la estación más cercana para solicitar la preparación de las comandas, puede ser abordado por alguien del personal que labora en las instalaciones

del hotel o centro turístico para notificarle que hay algún huésped /cliente que está solicitando el servicio.

Por lo que sí está cerca tiene que acudir a prestar el servicio en caso contrario le avisa a otro miembro del personal del establecimiento que este libre para que preste el servicio, cuando llega a la estación de pedidos, si está libre puede realizar su pedido, pero si no lo está, tiene que esperar a que el personal que está utilizando la estación termine para poder realizar el pedido de la comanda.

Esto sin contar que el pedido se capturó a mano al ser solicitada por el huésped/cliente, por lo que podrá presentar alguno de los errores que se mencionaron anteriormente con el sistema de lápiz y papel.

La saturación de las comandas también se deben a las normas con las que se rigen los establecimientos, como ejemplo hay algunos que operan de la siguiente forma:

Primero el cliente solicita su comanda, el personal de piso de venta anota con papel y lápiz la comanda y procede a trasladarse a captura de la misma en la estación de pedido, está pasa al área donde corresponda su preparación y una vez realizada, el personal de piso de venta traslada la comanda hacia el cliente, cuando el cliente está listo para retirarse y solicita la cuenta, el personal de piso de venta se dirige a la estación de pedido e imprime un ticket para el pago de la comanda el cual se entrega al cliente, este último entrega el efectivo que cubre el costo de la comanda y el personal nuevamente se traslada, pero ahora se dirige a un punto de venta para realizar el pago y entregar la factura o ticket al cliente, finalizando así el servicio.

Analizando esto podemos ver el recorrido del personal dentro de la elaboración de una comanda, notando el número de traslados que realiza el personal de piso de venta podemos ver que el tiempo de espera en la operación es algo considerable aun tomando en cuenta que el servicio se hizo de manera oportuna.

1.3.- Objetivo General.

Desarrollar un sistema informático para la gestión de órdenes en restaurantes de alimentos y bebidas.

1.4.- Objetivos Específicos.

Con la elaboración de este proyecto se busca:

- Analizar las necesidades del establecimiento.
- Seleccionar los elementos para una base de datos que permita acceder a la información necesaria para la operación del software a desarrollar.
- Proponer y desarrollar un software de gestión de pedidos que pueda ejecutarse en dispositivos móviles
- Experimentar con el sistema para un análisis de los tiempos en la operación del establecimiento.

1.5.- Justificación

Debido al ambiente turístico en el que laboran los restaurantes, el brindar un buen servicio es vital para el buen desarrollo de la zona, ya que al mantener siempre satisfecho al cliente se fomenta a que este regrese y comente cuando vuelva a su hogar la agradable experiencia que experimento, haciendo que más posibles clientes deseen el trasladarse y experimentarlo por ellos mismos.

En el caso de los establecimientos de preparación de alimentos y bebidas, como son los restaurantes, el brindar un buen servicio a los clientes, fomenta el que regresen a usar las instalaciones y dicho sea de paso el comentar de manera positiva su experiencia, haciendo que no solo sea posible que el cliente regrese, sino que otras personas puedan llegar a ir a experimentarlo por ellos mismos.

Durante la operación de un establecimiento dedicado a la elaboración de alimentos y bebidas la generación de cancelaciones es algo que se debe de evitar en medida de lo posible para prevenir perdidas al establecimiento.

Un establecimiento tiene un mínimo de cancelaciones del 1% estimado durante la operación diaria, aumentando este porcentaje durante fines de semana y periodos vacacionales a 2%.

Durante la operación promedio de un establecimiento con 48 mesas para 4 personas con 5 miembros del personal para atender a los clientes, genera alrededor de las 90 órdenes diarias en promedio, tomando en cuenta los periodos regulares, el monto asciende a 18 cancelaciones diarias en promedio y 126 semanales.

Hoy en día el uso de la tecnología en el sector turístico tiene un mayor valor que en años anteriores, con el uso del internet y los dispositivos móviles es básico el contar con Access point en el área de piso de venta para el uso de la conexión en los dispositivos de los clientes, lo que brindan una mayor comodidad y buena percepción del servicio por parte del cliente, por el lado de la operación interna el uso de las tecnologías también ha adquirido un gran valor ya que facilita la operación, desde el uso de dispositivos como pantallas táctiles, hasta el uso de software para uso específico de los restaurantes, que permiten llevar un mejor control de los recursos, el personal y los ingresos que se generan.

1.6.- Hipótesis

El uso de un sistema informático para la gestión de órdenes en restaurantes de alimentos y bebidas provocará la disminución en las pérdidas económicas.

Capítulo 2.- Estado del Arte.

2.1.- Importancia de las Tecnologías Móviles.

Hoy en día el rápido aumento en el uso de los dispositivos móviles en la población ha propiciado el rápido desarrollo de los mismos, lo que impacta en la vida diaria de los usuarios al estar en todo momento disponibles para su uso, pero no solo eso, sino que también las empresas han notado que el mercado de los dispositivos móviles es un mercado que se encuentra en rápido crecimiento, así como el desarrollo de las aplicaciones móviles que se ejecutan en los dispositivos.

Una aplicación móvil es aquella que se puede ejecutar en un dispositivo móvil que puede usarse con una sola mano que es fácil de usar y accesible desde cualquier lugar. Hoy en día hay muchas personas que cuentan con algún dispositivo móvil, por medio del cual usando alguna aplicación móvil pueden mantenerse conectados con amigos, navegar en internet, gestionar archivos, crear o manipular documentos, así como simplemente acceder a una fuente de entretenimiento [Mazumder, 2010] .

En un principio los dispositivos móviles no permitían el libre uso de las aplicaciones móviles como lo es ahora, como se mencionó anteriormente las aplicaciones móviles son ejecutadas en un dispositivo móvil, aunque su correcto funcionamiento depende de varios factores, como pueden ser la resolución de la pantalla, problemas de conectividad, las marcadas limitaciones del hardware y de interacción que se presentaban en los primeros años de desarrollo en los dispositivos móviles.

Pero con las posibles aplicaciones para estos dispositivos, las compañías dedicadas al desarrollo de la tecnología empleada en tales dispositivos, los desarrollan con una mejor resolución de pantalla, resolviendo los problemas de conectividad y mayor capacidad de almacenamiento y procesamiento.

Por lo que hoy en día las aplicaciones móviles pueden realizar alguna tarea específica para cada necesidad de los usuarios, al ser estas descargables, fáciles de usar y debido a que la gran mayoría de la población cuenta como mínimo con uno de estos dispositivos, las compañías comenzaron a notar la importancia de ellas, ya que cuentan con un significativo número de clientes potenciales, ya que el 81% de la población productiva entre los 25 y los 34 años cuenta con estos dispositivos [nice Agency, 2014a], incluso al día de hoy hay muchas compañías, grandes o pequeñas que obtienen ingresos debido al uso las aplicaciones móviles que están al servicio de los usuarios.

Incluso para las operaciones internas de las empresas actualmente se está optando por el uso de aplicaciones móviles en lugar de las aplicaciones web, esto debido a que representan un ahorro significativo en la operación.

Un ejemplo de esto es un estudio que se realizó en USA a 700 empresas el 4% de ellas utiliza aplicaciones móviles en las operaciones internas como por ejemplo en logística, suministros, mantenimiento, informes de ventas, servicios, gestión. Solo el 4% de las empresas consultadas en el estudio es un número muy pequeño pero el 100% de ellas cree que este número puede aumentar hasta el 50% en los próximos 2 años. Esto último debido a que el 17% de las empresas en Estados Unidos ahorran entre $25 000 a $100 000 anuales por pasar sus operaciones de aplicaciones móviles a usar aplicaciones móviles para uso interno [nice Agency, 2014a].

Las primeras aplicaciones en el mercado fueron muy sencillas y podrían ser, por decir algunos ejemplos: relojes de alarma, calculadoras, entre otras, para los desarrolladores actuales esto podría ser muy simple, pero en sus inicios este mercado fue muy subestimado por algunos, un ejemplo podría ser el caso de América donde debido a esta razón tuvo un tardío inicio en el mercado de las aplicaciones móviles [Mazumder, 2010].

Mientras que en otras partes del mundo el desarrollo de la tecnología móvil, así como el desarrollo de las aplicaciones móviles y su uso en diversos sectores.

En los últimos años el desarrollo de los dispositivos móviles ha propiciado que estos ahora obtengan mucho más poder de computo lo que incremento el desarrollo de las aplicaciones móviles, ya que hoy en día diversos desarrolladores de varios países aseguran que encuentran difícil la idea de salir de sus hogares sin su móvil.

Tan solo en México el 40% de la población que corresponde a los usuarios entre los 15 a los 34 años cuentan con acceso a internet por medio de algún dispositivo móvil [INEGI, 2016].

2.2.- La Tecnología y la Industria de la Comida.

Los dispositivos móviles se han vuelto indispensables y nuestra obvia dependencia a ellos para la toma de decisiones y el acceso a la información está creciendo cada vez más. Esto es más evidente al consultar las opciones de compra, específicamente las opciones de comedor. Donde con el uso de un dispositivo móvil se puede encontrar el restaurante más cercano, localizar en un mapa la ruta más corta para llegar, las opciones del menú, así como las reseñas de otros

usuarios, los clientes de los restaurantes dependen cada vez más de esta para llevarlos a la mesa [Riehle, 2013a] .

A pesar de la demanda de los consumidores el uso de las tecnologías en los restaurantes sigue siendo mínimo, sin embargo estos mismos también ven el valor que la tecnología puede aportar a sus negocios, el 73% de los encuestados acordaron que invertir en tecnologías para restaurantes haría sus operaciones más rentables [Riehle, 2013b].

Uno de los primeros avances en las tecnologías para restaurantes fue el uso de tabletas en las mesas junto a los condimentos en restaurantes como Chili's, Applebee's and Bufalo Wild Wings. Creando una experiencia más personalizada donde las tabletas toman el rol de un tercer servidor por medio del cual los clientes pueden ordenar sus bebidas, comidas, postres e incluso el pagar la cuenta [Group, 2016].

Con el ejemplo anterior los operadores de restaurantes pueden percibir la importancia del uso de las tecnologías en los restaurantes, con las cuales los clientes tienen una mayor participación en su experiencia dentro del restaurante. También permite que los clientes recomienden la experiencia con otros posibles clientes, lo que generaría más ganancias al restaurante.

En 2014 el porcentaje de restaurantes que optaron por dedicar más recursos a las tecnologías fueron algo dispares en la orientación de sus objetivos pero se representan de la sig. Manera:

En el caso de la tecnología orientada al cliente (como wifi, ipads/tabletas, Smartphone) para la realización de pedidos el 43% de los restaurantes de comida rápida optaron por esta clase de

tecnología, mientras que los operadores de restaurantes casuales se inclinaron más por esta alternativa llegando hasta un 57 %.

En el caso de las tecnologías de punto de venta o para uso dentro de las instalaciones para la toma de capturas, los servicios de comida rápida opinaron que sí podrían usar estas tecnologías contando con un 36% de aceptación, mientras que los servicios de comida casual obtuvieron un 43% de aceptación en el uso de estas tecnologías.

Con el rápido incremento en el uso de las tecnologías móviles o inalámbricas que tiene un gran impacto en nuestras vidas, al grado de ya no poder salir de casa sin algún dispositivo móvil, sería natural y comprensible que las personas busquen aplicaciones que satisfagan sus necesidades.

Por otro lado la mayoría de los restaurantes buscan cualquier aplicación móvil que realce la experiencia gastronómica así como que aumente los beneficios. En cualquier restaurante que cuenta con un sistema estático o punto de venta, el proceso que se realiza en la elaboración de una orden es el siguiente:

Primero el mesero se acerca a la mesa y anota la orden en lápiz y papel, una vez realizado se dirige a alguna de las estaciones de pedido e ingresan la orden, posteriormente el pedido pasa al área de elaboración, sea esta el bar o la cocina, esto genera mucho tiempo desperdiciado para el mesero en el ir y venir del mesero entre el cliente y el área de preparación, además de generar un desperdicio de papel, al este ser utilizado para anotar la orden y una vez ingresada la orden el papel utilizado para la anotación es desechado.

Para poder superar esta problemática en cualquier restaurante mediano o grande, se podría diseñar e implementar una aplicación inalámbrica, esta se puede instalar en una tableta, en la cual el cliente selecciona la comida del menú que se le proporciona y le da la orden al camarero, las ordenes de la tableta del cliente son enviadas a la cocina de forma inalámbrica, así como se actualiza la base de datos [Gupta, 2012b].

Las siguientes son unas ventajas del uso de las aplicaciones móviles:

- Debido a las tabletas, los camareros no necesitan salir de la mesa para procesar el pedido.
- Los camareros pueden pasar más tiempo en satisfacer a los clientes y no habrá mezcla de órdenes.
- El camarero sabrá la disponibilidad de comida, ya que los mensajes pueden ser enviados desde la cocina a la mesa del camarero.
- El estado del alimento ordenado puede ser comprobado en cualquier momento por el camarero así como el cliente.

El problema descrito en el artículo es la problemática que se presenta en la mayoría de los restaurantes que no cuentan con un sistema móvil de gestión para los pedidos, por lo que se mencionan algunas de las ventajas que representa el desarrollo e implementación de un sistema móvil.

El sistema descrito es pensando en un dispositivo móvil que se encuentre en la mesa para que la orden que se genere una vez preparada sea transportada a la mesa correspondiente, además con el sistema se estaría eliminando un trayecto que sería el que corresponde al de ingresar la orden a la estación o punto de venta designado, en caso de no contar con un sistema de gestión, se eliminaría el traslado de la orden de la mesa hasta el área de preparación.

En el caso del sistema que se desea elaborar en este proyecto no sería un dispositivo móvil que se encuentre en la mesa de los clientes, sino un dispositivo que sería utilizado por el mesero ya que en los establecimientos en los que se realizó la investigación de las necesidades, el sistema eliminaría un traslado, lo cual le permitirá al mesero usar el tiempo de ese recorrido en llevar los complementos a la mesa que se está prestando el servicio.

En el artículo [Gupta, 2012a] se especifican los recursos que se utilizaran para la operación de dicho sistema que se implementara, en el cual se menciona por que se seleccionan las tecnologías que se necesitaran, como el bluetooth que fue considerado junto al wi-fi, pero debido a que el wi-fi permite el trabajar en áreas más grandes que el bluetooth, además de la diferencia en la potencia de la señal que manejan ambas tecnologías.

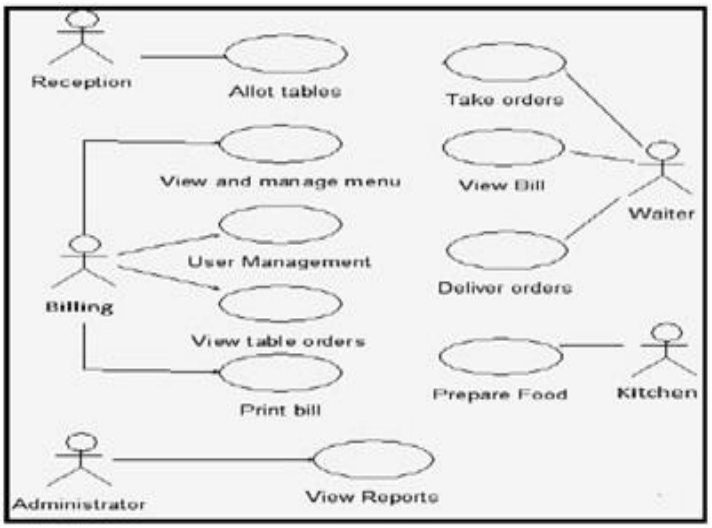

Figura 2.1.- Diagrama de Caso de uso [Gupta, 2012a].

Así mismo el uso de las tablets por sobre las PDA, donde la elección ideal son las tablets ya que estas por su tamaño resultan ser mucho más cómodas para los usuarios, quienes están acostumbrados a pantallas del tamaño de una laptop o de una pantalla de pc, además de que cuenta con sistema operativo Android, el cual es de uso libre, por lo que no requiere la compra

de software adicional, es personalizable, basado en Linux en cuestiones básicas como la seguridad y permite a los desarrolladores el realizar programación en Java, además de que las aplicaciones pueden ser desarrolladas en Windows, Mac, Linux [de la Vega, 2012a].

La metodología que se emplea tiene el principal objetivo de aumentar la eficiencia del sistema de pedidos de alimentos y reducir los errores humanos y proporcionar, servicios de calidad a los clientes de los restaurantes. La aplicación en las tabletas debe ser capaz de comunicarse inalámbricamente con otros dispositivos.

La usabilidad del sistema en general tiene que ver con la forma en que se usa algún elemento, la facilidad con que se usa y si permite hacer lo que se necesita. Formalmente, la definición más utilizada o reconocida de usabilidad es la que se expone en la norma ISO 9241-11, en la que usabilidad se describe como el grado con el que un producto puede ser usado por usuarios específicos para alcanzar objetivos específicos con efectividad, eficiencia y satisfacción, en un contexto de uso específico [Isabel, 2013].

La norma define como especificar y medir la usabilidad de productos y aquellos factores que tienen un efecto en la misma; también destaca que la usabilidad en terminales con pantalla de visualización es dependiente del contexto de uso y que el nivel de usabilidad alcanzado dependerá de las circunstancias específicas en las que se utiliza el producto. Las relaciones que existen entre el usuario, el producto, los atributos, el contexto de uso y los objetivos que se quieren lograr, se pueden observar en el framework de usabilidad propuesto en la norma citada (Fig.2.2).

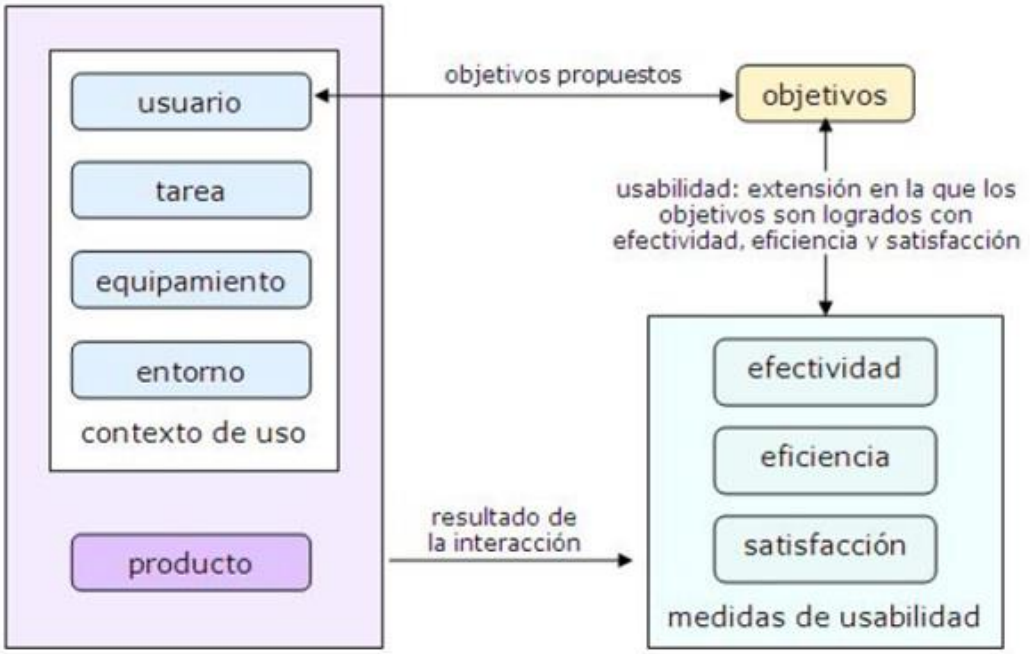

Figura 2.2.- Framework de usabilidad (ISO 9241-11) [Isabel, 2013].

En primer lugar la persona en la recepción está autorizada para asignar mesa a los clientes, a través de una tableta. Entonces, tan pronto como se asigna la mesa, el cliente es dirigido a su mesa con un mesero esperando tomar su orden. El cliente ve la tarjeta de menú categorizada en forma digital en la tableta. El mesero o mesera introduce los pedidos en las tabletas de mano. Las órdenes son enviadas a la cocina vía Wi-Fi. El personal de la cocina envía una notificación si la comida está disponible o no.

El lado de la cocina también puede enviar el informe de progreso de la comida a la mesa del camarero. Cuando la cocina envía una notificación de que el alimento ha sido preparado, el camarero en la cocina sirve la comida en la respectiva mesa.

Si hay una necesidad de modificación en el menú de alimentos, el administrador modifica el menú en la base de datos. El menú cambiado se actualiza en el camarero.

El desarrollo de aplicaciones para plataformas móviles se ha visto influenciado por la necesidad en la administración de los negocios, en el caso de los restaurantes el avance tecnológico es casi nulo, por el hecho de operar con un mismo sistema en común desde los primeros (lápiz y papel) [de la Vega, 2012b].

La funcionalidad del sistema consiste básicamente en una aplicación móvil que permita realizar los pedidos en un restaurante de manera digital por medio de un dispositivo móvil con lo el sistema previamente instalado, el sistema contara con los precios y la disponibilidad de los platillos para las ordenes, de esta manera los meseros no tiene por qué tener contacto con el cliente, con la excepción del momento en el que entregue los platillos preparados desde el área de cocina o el pago de la cuenta [de la Vega, 2012c]. Esto permite un ahorro de tiempo y personal dentro de las instalaciones de los restaurantes.

Otra variante de estos sistemas de gestión de pedidos son los sistemas basados en dispositivos móviles, pero tomando en cuenta los dispositivos móviles que utilizan los mismos clientes [D.Valencia, 2012], donde dicho sistema permite el realizar los pedidos de las ordenes en el interior del establecimiento mediante el uso de códigos QR que permitirán el identificar la mesa que realizó el pedido, así como el poder realizarlo desde el hogar de los clientes.

2.3.- Software para Restaurante.

En cuestión de tecnologías para el uso de la industria restaurantera no se puede evitar el hablar del software requerido para el uso de dichas tecnologías, aunque el costo de adquisición y posterior renta del mismo suele ser un costo muy alto, tomando en cuenta que cada establecimiento requiere una solución diferente a la de cualquier otro, el software utilizado no

puede ser el mismo para todos por lo que es lo que necesaria una personalización de la solución, lo que conlleva a la adquisición de módulos adicionales a la versión estándar del software, lo que por consiguiente se traduce como un gasto adicional en la adquisición de la licencia y una aumento en la renta del mismo.

Actualmente hay pocas opciones en cuanto a empresas dedicadas a la implementación de este tipo de software, por lo que se describirá el software de mayor uso en los establecimientos en los que se realizó la investigación de los análisis de requerimientos.

2.3.1.- Soft Restaurant.

Actualmente el software más utilizado es el Soft Restaurant, el cual en la mayoría de los establecimientos en los que se realizó la investigación de las necesidades mencionaron operar con dicho sistema, el cual es adquirido en su formato estático (punto de venta).

Soft Restaurant es el software especializado en el sector restaurantero para contar con un control operativo y administrativo total de su restaurante. Soft Restaurant es una herramienta completa que permite administrar un negocio evitando errores u omisiones, proporcionando una mayor seguridad en el manejo de una empresa. Su diseño lo convierte en una herramienta fácil de utilizar y dominar. Soft Restaurant es ideal para todo tipo de restaurantes, bares, cafeterías, pizzerías, fast food y taquerías.

La versión Standard es ideal para pequeños establecimientos, la versión Professional ofrece mayor seguridad y rendimiento en el resguardo de su información.

Cuenta con diversos reportes, facturación, control de recetas, inventarios, almacenes, mesas, reservaciones, servicio rápido, pedidos a domicilio, módulos de fidelización de clientes, monitor de producción, módulos adicionales, mapeo de mesas, control antifraudes y muchas características más. Esta versión del software es dedicado a las estaciones fijas o punto de ventas, por lo que su adquisición es de $11,600 m.n. y con una renta mensual por la licencia de $ 870 m.n. [Soft, 2016b].

2.3.1.1.-Módulos.

Los módulos con los que cuenta el software dependen de las necesidades que el restaurante desee satisfacer o el presupuesto con el que se permita invertir en el software, en cualquiera de los casos el costo por la licencia de dicho software cubre cierto número de módulos que se pueden personalizar para el establecimiento, pero esto requiere el pago de una renta mensual, en caso de requerir un módulo que el software no incluya en la licencia adquirida el establecimiento debe comprar la licencia del módulo extra, además de pagar mensualmente la renta, esto es prácticamente obligatorio ya que este módulo no es independiente ya que requiere que el establecimiento cuente con el software principal para poder operar.

2.3.1.2.- Punto de Venta.

Maneja una excelente interfaz para llevar su operación en el punto de venta con los diferentes tipos de servicios, rápido, comedor, a domicilio y drive thru, contando con opciones y funciones como: juntar mesas, juntar cuentas, cambio de mesero, cambio de mesa, reapertura de cuentas, multi-impresión, cancelaciones, descuentos, promociones, mapa de mesas, reloj

checador, renta de mesa de billar, últimos pedidos (servicio a domicilio) y muchas más que harán que la operación sea más eficiente.

2.3.1.3.- Inventarios.

La parte más importante de un restaurante, manejado de la forma correcta incrementara sus ventas y reduce sus costos, permitiendo dar de alta a sus proveedores, las presentaciones de sus insumos, manejar stock mínimo y máximo, registrar las compras, control de sus recetas, producción de insumos, explosión de productos y con la facilidad de poder generar los reportes como existencias, traspasos de almacén y muchos más, lo cual generara una correcta rotación y la reducción de mermas.

2.3.1.4.- Administración y Seguridad.

Con las herramientas que ofrece Soft Restaurant se enfoca en hacer crecer un negocio, ya que cuenta con los reportes necesarios para crear estrategias, incrementar sus ventas y dirigir de manera adecuada, como también con la seguridad necesaria para llevar el control de los perfiles y los usuarios, permisos de acceso, contraseñas, programar envíos de alertas antifraudes y diferentes opciones que ofrece el sistema para el mejor desempeño de su personal y de su empresa.

Los módulos adicionales de este software oscilan entre los $460 m.n. hasta los $6960 m.n. y estos pueden ser: Monitores de ventas web, monitor de cocina básico, monitor de cocina profesional, huella digital, Soft Restaurant móvil.

Soft Restaurant móvil, es un módulo más del software, que no puede funcionar de manera independiente, sino que debe de funcionar con el sistema principal [Soft, 2016a].

Capítulo 3.- Conceptos Teóricos.

El proyecto a desarrollar en la propuesta de tesis busca el desarrollar una versión móvil de los sistemas de gestión de pedidos existentes, lo cuales son muy costosos tanto en su adquisición como en su renta y mantenimiento de los equipo que se necesitan para su óptimo funcionamiento (Estaciones fijas).

Por lo que se desarrollara un sistema móvil que permita el ser utilizado en los establecimientos, que reduzca los tiempos de espera, así como agilizar la operación del piso de venta en el establecimiento.

Por lo que para llevar a cabo dicho desarrollo se optó por el utilizar el lenguaje de programación Java, ya que es un lenguaje de programación que puede ser ejecutado en prácticamente cualquier dispositivo, lo que resulta muy conveniente dado el hecho de que la aplicación a desarrollar será ejecutada en distintos tipos de dispositivos, tanto móviles como fijos basados en el sistema operativo Android.

3.1.- Java.

La primera versión de java empezó en 1991 y fue escrita en 18 meses en Sun Microsystems. De hecho, en ese momento, ni siquiera se llamó Java; se llamó Oak y se utilizó en Sun para uso interno, lo que había de novedoso en Oak era que era que en un principio se quería crear un lenguaje que se pudiera utilizar en electrónica y terminó siendo algo que podía ejecutarse en todas las pc, razón por la cual rápidamente se popularizó en las corporaciones que lo adaptaron para su uso interno en lugar de C++ [Holzner, 2001].

A partir de 1995 Oak pasaría a llamarse Java cuando se lanzó para su uso público y supuso un éxito inmediato, ya que un programa en Visual C++ de Microsoft es grande ya que un programa MFC (Microsoft Foundation Class) puro no baja de los 5MB sin incluir las librerías de enlace dinámico (DDLs) que la plataforma Windows necesita para ejecutar los programas Visual C++ [H, 2009].

Los programas Java, por el contrario se constituyen de forma diferente el propio lenguaje Java esta implementado como la máquina virtual de Java (JVM) que la aplicación en la que actualmente se ejecuta Java.

Los programas Java son compilados creando bytecodes compactos y son estos lo que JVM lee e interpreta para ejecutar el programa, de esta forma los programas Java son pequeños y por lo tanto también por eso es que puede ser ejecutado en cualquier computadora, ya que todo el código máquina necesario para ejecutarlo ya está en la computadora [Corporation, 2015].

3.1.1.- La Seguridad del Lenguaje Java.

La seguridad en java ha llegado a ser un asunto extremadamente importante, cuando se ejecuta un programa JVM puede monitorizar estrictamente lo que va ocurriendo y si hay algo dudoso, como puede ser tratar de escribir en un archivo, puede prevenir esta operación.

Los programas Java son robustos, lo que significa que son fiables y pueden gestionar bien los errores, con frecuencia son sencillos de escribir comparados con C++, son multihilo, por lo que se pueden ejecutar un número de tareas al mismo tiempo y ofrecen un alto rendimiento [Holzner, 2001].

El resultado final es que se puede escribir un programa Java una vez y puede descargarse fácilmente y ejecutarse en todo tipo de máquinas.

3.1.2.- Portabilidad de Java.

Los creadores de Java intentaron acabar con el problema inherente a la falta de portabilidad en el código objeto introduciendo el nivel de código byte entre los niveles de código fuente y código objeto. Los compiladores de Java no compilan en forma de código objeto [Holzner, 2001]. En lugar de eso lo hacen a nivel de código byte, que posee las mejores características, tanto de código objeto como de código fuente:

- Como el código objeto, el código byte utiliza un formato que funciona de forma muy cercana con el hardware de la computadora, por lo que se ejecuta de manera rápida.

- Como el código fuente, el código byte es genérico, por consiguiente puede ejecutarse en cualquier tipo de computadora.

Cuando el código byte Java de un programa se ejecuta, este se traduce en código objeto mediante el programa interprete de código byte de la computadora el intérprete de código byte se conoce también como Java Virtual Machine (máquina virtual de Java) o JVM.

Para ejecutar el código byte Java, una computadora debe tener instalada la JVM. Afortunadamente, la instalación de una JVM es muy sencilla. Es un programa, y no requiere mucho espacio en memoria; es fácil de obtener; cualquiera puede bajarlo de forma gratuita de internet [Corporation, 2015].

3.1.3.- Java en la Actualidad.

Hoy en día, los programadores utilizan Java en muchos ambientes. Aun insertan programas Java en las páginas web, que se denominan applets, la popularidad inicial de los applets ayudo a posicionar a Java como uno de los lenguajes de programación líderes en el mundo.

A pesar de que los applets aún tienen un papel significativo en el éxito actual de Java, otros tipos de programas Java han venido a superarlos en términos de popularidad.

Repasando estos programas Java que han influido en el éxito actual de Java tenemos:

- Un applet es un programa que está contenido en una página web.
- Una Java Server Page (JSP) es una página web que tiene fragmentos de un programa Java, no un programa Java completo como en las applet
- Una aplicación Java Micro Edition (ME) es un programa de Java que se ejecuta en un dispositivo con recursos limitados, como por ejemplo los celulares o televisores con decodificador de señales digitales.
- Una aplicación Java Edición Estándar (SE) es un programa que se ejecuta en una computadora estándar.

3.2.- Ambientes de Desarrollo.

Existen diferentes formas de introducir un programa Java en una computadora. Se puede utilizar un ambiente de desarrollo integrado o un editor de texto plano.

Un ambiente de desarrollo integrado (IDE, por sus siglas en inglés) es más que una larga pieza de software, que nos permite introducir, compilar y ejecutar programas. La introducción, compilación y ejecución son parte del desarrollo de un programa y están integradas juntas en un ambiente, de ahí su nombre. Algunos IDE son gratuitos y algunos bastante caros [Holzner, 2001].

Un editor de texto plano es una pieza de software que permite al usuario el introducir y salvar texto a manera de archivos. Los editores de texto plano no saben nada acerca de compilación y ejecución de un programa. Si se utiliza un editor de texto plano para introducir un programa, sería necesario utilizar herramientas de software separadas para compilar y ejecutar el programa [H, 2009].

3.2.1.- NetBeans.

NetBeans es un proyecto exitoso de código abierto. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos [NetBeans, 2015a]. Al día de hoy hay disponibles dos productos: el NetBeans IDE y NetBeans Platform.

3.2.2.- NetBeans IDE.

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE.

NetBeans IDE le ofrece aplicaciones esqueleto en forma de plantillas de proyectos para todas las tecnologías que admite. Además, proporciona un conjunto de aplicaciones de ejemplo, algunas de las cuales se pueden recrear paso a paso.

El IDE proporciona plantillas de proyectos y proyectos de ejemplo que ayudan a crear aplicaciones Java SE, aplicaciones Java EE, aplicaciones Java ME, aplicaciones HTML5, aplicaciones de la plataforma NetBeans, aplicaciones PHP y aplicaciones C / C ++ [NetBeans, 2015b].

También está disponible NetBeans Platform; una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones [NetBeans, 2015c].

3.3.- Computación Móvil.

Es la comunicación de diferentes equipos portátiles o móviles de hardware y software, que hacen uso de la computación para realizar diferentes tareas computacionales permitiendo la movilidad y la conexión a otros dispositivos por medio de diferentes tecnologías de comunicación inalámbrica y en la administración de forma óptima del procesamiento, almacenamiento y el consumo de la energía.

Entre los dispositivos móviles se encuentran actualmente las computadoras portátiles, mini computadoras, teléfonos celulares, Smartphone, Tablets, e-Readers, en general cualquier

dispositivo que tenga y permita la conexión a otros dispositivos por medio de diferentes tecnologías de comunicación inalámbrica (Wi-Fi (Wireless Fidelity), GSM (Global System for Mobile), Bluetooth, RFID (Radio Frequency Identification), GPRS (General Packet Radio Service) y Satelital)[Gabriel, 2014].

Entre las principales ventajas de la computación móvil se encuentra la movilidad y flexibilidad que permite enviar y recibir información sin los obstáculos o restricciones de un espacio físico (oficina, campus, edificio), alta escalabilidad para configurar una amplia variedad de topologías de red, facilidad de instalación, bajos costos de implementación, amplia la capacidad de toma de decisiones porque permite obtener y analizar datos críticos, incrementa la productividad y aumenta la cercanía con los clientes de negocio de una organización.

Igualmente, la computación móvil tiene asociadas unas limitaciones, entre las cuales podemos mencionar los dispositivos con recursos limitados para almacenamiento y procesamiento, anchos de banda menores comparado con las redes fijas, problemas de comunicación provocados por cambios en condiciones del ambiente, variación de la intensidad de la señal, riesgos de seguridad asociados a los dispositivos y/o señales emitidas.

La computación móvil tiene una variedad de aplicaciones y actualmente se usa bastante en los medios de transporte para establecer la ubicación vía GPS, mantener actualizada la información de condiciones ambientales, estado de vías, noticias, mensajes con vehículos cercanos mediante redes sociales especializadas y comunicación personal; también se utiliza bastante en el campo de salud, donde los funcionarios médicos utilizando dispositivos inalámbricos obtienen y comparten información del estado de los pacientes; en el campo de los negocios se

usa bastante esta tecnología pues permite el acceso a información de oficinas, ventas, inventarios o compras de productos [Ing. Pedro Julio Colorado Ángel, 2015b].

3.4.- Evolución de la Computación Móvil.

La computación móvil se desarrolla con el nacimiento de la necesidad de transportar la información, inicialmente solo se trabajaba con computadoras centralizadas y la información estaba almacenada en un solo lugar, cuando se necesitaba se debía acudir físicamente al lugar donde se encontraba.

Esto funcionaba correctamente pero con el aumento de la información y los usuarios, los centros se hicieron insuficientes para atender la cantidad de peticiones de información solicitadas por los usuarios [Ing. Pedro Julio Colorado Ángel, 2015b].

Un hito importante para la computación móvil fue la aparición de las redes, específicamente las redes inalámbricas, las cuales con su evolución permitían el uso de los equipos de cómputo conectarse a una red sin necesidad de estar cableada o modificar la estructura de la red, ofreciendo conveniencia y flexibilidad [Roopa, 2005].

La construcción de equipos de comunicación y teléfonos celulares posibilito mejoras en las comunicaciones, permitiendo el uso de estas características para convertirse en servicios para los usuarios, hasta llegar a lo que actualmente se denomina como tecnologías de última generación en cuanto a redes de comunicación, el uso de equipos de comunicación inteligente y de servicios de internet sobre las redes de comunicación celular, es decir, la unión de la red de redes {internet- con las redes celulares[Ing. Pedro Julio Colorado Ángel, 2015c].

3.5.- Tipos de Dispositivos Móviles.

Un dispositivo móvil se puede definir como un aparato de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, que ha sido diseñado específicamente para una función, pero que puede llevar a cabo otras funciones más generales. De acuerdo con esta definición existen multitud de dispositivos móviles, desde los reproductores de audio portátiles hasta los navegadores GPS, pasando por los teléfonos móviles, los PDAs o los Tablet PCs.

3.5.1.- Tipos de Dispositivos Móviles.

Dado el variado número de niveles de funcionalidad asociado con dispositivos móviles, era necesario hacer una clasificación de los mismos:

Dispositivo Móvil de Datos Limitados (Limited Data Mobile Device): teléfonos móviles clásicos. Se caracterizan por tener una pantalla pequeña de tipo texto. Ofrecen servicios de datos generalmente limitados a SMS y acceso WAP [Arturo Baz Alonzo, 2012a].

Dispositivo Móvil de Datos Básicos (Basic Data Mobile Device): se caracterizan por tener una pantalla de mediano tamaño, menú o navegación basada en iconos, y ofrecer acceso a emails, lista de direcciones, SMS, y, en algunos casos, un navegador web básico. Un típico ejemplo de este tipo de dispositivos son los teléfonos inteligentes (“smartphones") [Arturo Baz Alonzo, 2012a].

Dispositivo Móvil de Datos Mejorados (Enhanced Data Mobile Device): se caracterizan por tener pantallas de medianas a grandes, por encima de los 240 x 120 pixeles, navegación de tipo stylus, y que ofrecen las mismas características que el "Dispositivo Móvil de Datos Básicos" más aplicaciones nativas como aplicaciones de Microsoft Office Mobile y aplicaciones corporativas usuales, en versión móvil, como SAP, portales intranet, etc. Este tipo de dispositivos incluyen los S.O. como Windows Mobile [Arturo Baz Alonzo, 2012a].

3.5.2.- PDA's (Personal Digital Assintant).

Un PDA, es una computadora de mano originalmente diseñada como agenda electrónica con un sistema de reconocimiento de escritura. Inicialmente los PDA's incluían aplicaciones estrictamente relacionadas con su función como agenda electrónica, es decir, se reducían a calendario, lista de contactos, bloc de notas y recordatorios. Con el paso del tiempo han ido evolucionando hasta los dispositivos actuales que ofrecen un rango mucho más extendido de aplicaciones, como juegos, acceso al correo electrónico o la posibilidad de ver películas, crear documentos, navegar por Internet o reproducir archivos de audio.

Las características del PDA moderno con pantalla sensible al tacto, conexión a una computadora para sincronización, ranura para tarjeta de memoria, y al menos Infrarrojo, Bluetooth o Wi-Fi [Arturo Baz Alonzo, 2012a].

3.5.3.- Smartphone o Teléfono Inteligente.

Un "Smartphone" o teléfono inteligente en español es un dispositivo electrónico que funciona como un teléfono móvil con características similares a las de una Pc. Es un elemento entre un

teléfono móvil clásico y una PDA ya que permite hacer llamadas y enviar mensajes de texto como un móvil convencional pero edemas incluye características cercanas a las de una Pc. Una característica importante de casi todos los teléfonos inteligentes es que permiten la instalación de programas para incrementar el procesamiento de datos y la conectividad [Arturo Baz Alonzo, 2012b].

Estas aplicaciones pueden ser desarrolladas por el fabricante del dispositivo, por el operador o por un tercero. Los teléfonos inteligentes se distinguen por muchas características, entre las que destacan las pantallas táctiles, un sistema operativo así como la conectividad a Internet y el acceso al correo electrónico [Arturo Baz Alonzo, 2012b].

3.6.- Sistemas Operativos.

Un Sistema Operativo gestiona los recursos del sistema, optimiza su uso y resuelve conflictos. El sistema operativo va a coordinar todo el funcionamiento del hardware, iniciando todos los elementos para que estén preparados para recibir trabajo, va a ordenar cuando y como debe trabajar el hardware. Es el sistema operativo el que va a asignar los recursos hardware a los distintos programas, va a coordinar y llevar el seguimiento de la ejecución de todos los programas en el sistema, va a tomar las decisiones para evitar que se produzcan conflictos entre ellos y va a tratar que el sistema sea lo más eficiente [Santiago, 2007].

3.6.1.- Sistemas Operativos para Dispositivos Móviles.

Un sistema operativo para dispositivos móviles es considerado el programa principal y es capaz de administrar todos sus recursos para ser utilizados de manera eficiente, cómoda y sin

interrupciones, de tal manera que el usuario pueda mantener una comunicación sin problema haciendo uso de los recursos que el hardware le suministra [Ing. Pedro Julio Colorado Ángel, 2015d].

Las características más relevantes de un sistema operativo móvil son:

- Kernel Unificado
- Construido por Capas
- Multiproceso y Multitarea
- Soporte a diferentes Pantallas
- Soporte Multilenguaje
- Multihilo
- Conectividad Inalámbrica
- Administración del Hardware
- Administración de Aplicaciones
- Navegación Web
- Capacidad de Adaptación
- Reinvención y Mejoramiento
- Personalizable
- Multiusuario
- Inteligente

El uso de uno u otro S.O determinara las capacidades multimedia de los dispositivos, y la forma de éstas de interactuar con el usuario. Existen multitud de opciones, si bien las más extendidas son Symbian, BlackBerry OS, Windows Mobile, y recientemente iPhone OS y el sistema móvil

de Google, Android, además por supuesto de los dispositivos con sistema operativo Linux [Arturo Baz Alonzo, 2012c].

3.6.2.- Sistema Operativo Android.

En los últimos años los teléfonos móviles han experimentado una gran evolución, desde los primeros dispositivos móviles, grandes y pesados, pensados solo para hablar por teléfono en cualquier parte, a los últimos modelos, con los que el término “medio de comunicación" se queda bastante pequeño.

Es así como nace Android, Android es un sistema operativo y una plataforma software, basado en Linux para teléfonos móviles. Además, también usan este sistema operativo (aunque no es muy habitual), tablets, netbooks, reproductores de música e incluso PC's.

Es un sistema operativo para dispositivos móviles, que fue presentado en Noviembre de 2007, hoy en día es el principal producto de Open Handset Alliance, un grupo de fabricantes y desarrolladores de software y hardware [Álvaro Zapata, 2012b].

Dispone de una gran comunidad de desarrolladores que escriben Las aplicaciones para extender la funcionalidad de los dispositivos de las cuales, dos tercios son gratuitas y están disponibles para la tienda de aplicaciones oficial de Android que es el Android Market o su actualización, PlayStore, sin tener en cuenta aplicaciones de otras tiendas no oficiales para Android.

Sistema Operativo	Descripción
Android	Plataforma móvil de código abierto disponible libremente para quien desee utilizarlo. En 2008 con la unión a la *Open Handset Alliance (OHA)* Android fue lanzado bajo la licencia de código abierto, permitiendo a los fabricantes de dispositivos personalizar y permitir nuevas experiencias de usuario, impulsar la innovación y elección del consumidor. Tiene como base el kernel del sistema operativo Linux. Entre las versiones más conocidas se encuentran Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean y Kitkat Es el sistema operativo móvil con mayor crecimiento en el mercado. Se ha convertido en el favorito de los consumidores y desarrolladores, impulsando un fuerte incremento en el consumo de aplicaciones. Mensualmente los usuarios de Android descargan más de 1.5 millones de aplicaciones y juegos de Google Play36.
Apple iOS	Desarrollado por la compañía Apple para para aprovechar al máximo la avanzada tecnología del hardware (iPhone, iPad e iPod), para lo cual fue diseñado en base a una variante del kernel de MacOS X.
Windows Phone	Desarrollado por Microsoft Corporation para dispositivos móviles, tiene una plataforma de comunicación convergente para las aplicaciones existentes y como ventaja las características compartidas con Microsoft Windows.
BlackBerry	Desarrollado por la compañía Research in Motion Limited (actualmente BlackBerry Limited), para distribuirlo con sus equipos Smartphone. Este sistema operativo ofrece las soluciones de movilidad empresarial seguras e integradas.
Firefox OS	Desarrollado por Mozilla para teléfonos inteligentes, su código es abierto y fue creado completamente utilizando HTML5 y otros estándares web abiertos, que lo hace libre de las normas y restricciones de las plataformas privadas existentes.
Tizen	Patrocinado por Linux Foundation y la Asociación Tizen, siendo un sistema operativo abierto y flexible construido desde cero para hacer frente a las necesidades de todos los actores del ecosistema de dispositivos móviles, incluidos los fabricantes de dispositivos, operadores móviles, desarrolladores de aplicaciones y proveedores de software independientes

Tabla 3.1.- Sistemas Operativos Móviles [Ing. Pedro Julio Colorado Ángel, 2015a].

Actualmente Android posee aproximadamente el 32,9% de cuota de mercado a escala mundial de los teléfonos inteligentes, por delante de Symbian que posee aproximadamente el 30,6%. En tercer lugar se sitúa iOS con un 16% del mercado [Zulma Cataldi, 2012]. La estructura del sistema operativo Android se compone de aplicaciones que se ejecutan en un framework Java, son orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual, poco

después de su lanzamiento Google liberó la mayoría del código de Android bajo la licencia Apache, que es libre y de código abierto.

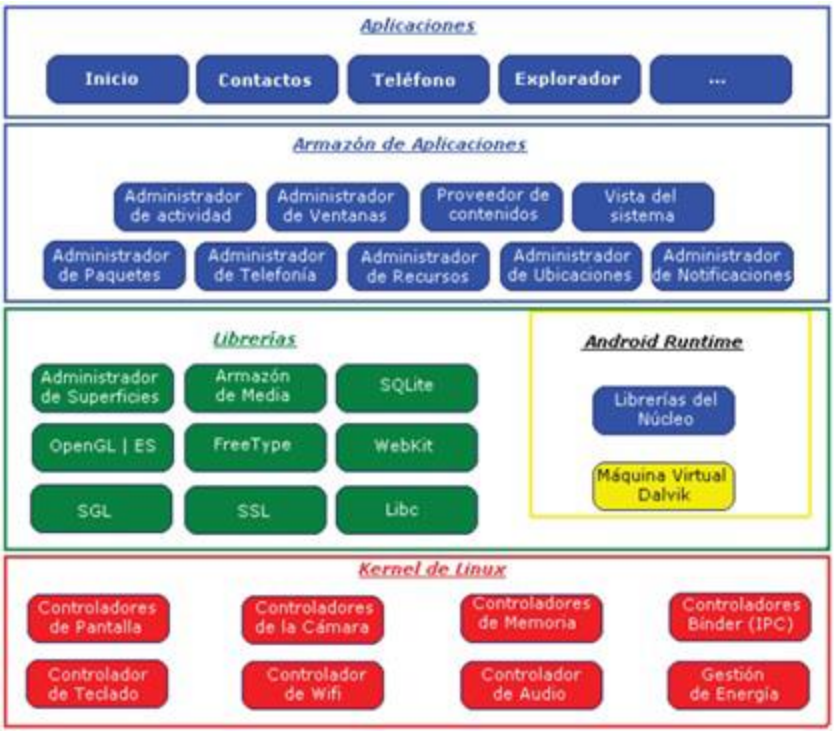

Figura 3.1.- Capas del Sistema Android [Álvaro Zapata, 2012a].

Android ha tenido numerosas actualizaciones desde su liberación inicial. Estas actualizaciones al sistema operativo base típicamente arreglan “bugs" y agregan nuevas funciones. Generalmente cada actualización del sistema operativo Android es desarrollada con un nombre en código de un elemento relacionado con postres (iniciando con Cupcake en las versiones 1.5 al actual Nougat con la versión 7.0) [Álvaro Zapata, 2012a].

Android se puede usar como una herramienta de trabajo y de aprendizaje. La opción es posible sobre todo porque es una solución basada en software libre. En algunos casos, Android provee de algunas soluciones que son más eficaces que las de los competidores.

Un ejemplo de esto es un estudio que se realizó en USA a 700 empresas el 4% de ellas utiliza aplicaciones móviles en las operaciones internas como por ejemplo en logística, suministros, mantenimiento, informes de ventas, servicios, gestión. Solo el 4% de las empresas consultadas en el estudio es un número muy pequeño pero el 100% de ellas cree que este número puede aumentar hasta el 50% en los próximos 2 años. Esto último debido a que el 17% de las empresas en Estados Unidos ahorran entre $25 000 a $100 000 anuales por pasar sus operaciones de aplicaciones móviles a usar aplicaciones móviles para uso interno [nice Agency, 2014b].

La capacidad o facilidad para poder adaptarlo a diferentes dispositivos móviles, en este sentido, Android es el que mayor adaptabilidad presenta, ya que cada vez se está empleando en más dispositivos, no solo teléfonos móviles, sino también en netbooks. En cambio el resto de sistemas operativos tienen una adaptabilidad algo menor y más complicada.

La arquitectura de Android está formada por 4 capas que permiten la generación de aplicaciones. El acceso a cada capa se realiza por intermedio de librerías, en las cuales cada capa puede utilizar los elementos de la capa inferior para realizar sus funciones. Las capas de la arquitectura Se muestran en la fig. 3.1.

3.6.3.- Aplicación Móvil.

Enfocando el concepto en el área de la computación móvil, las aplicaciones móviles son los conjuntos de instrucciones lógicas, procedimientos, reglas, documentación, datos e información asociada a estas que funcionan específicamente en dispositivos móviles, como por ejemplo teléfonos inteligentes, televisores inteligentes o tabletas.

Este tipo de aplicaciones se desarrollan teniendo en cuenta las limitaciones de los propios dispositivos, como por ejemplo el bajo poder de cómputo, la escasa capacidad de almacenamiento o su ancho de banda limitado.

Los principales lenguajes de programación utilizados para el desarrollo de aplicaciones móviles son: Java, Objetive C, Bada, WebOS, C#, C++, HTML5, HTML/CSS/JavaScript. En la figura 3.2 [Ing. Pedro Julio Colorado Ángel, 2015e] se muestran los lenguajes de programación más utilizados.

Figura 3.2.- Lenguajes utilizados para desarrollo de aplicaciones.

Según su desarrollo: web, nativas e hibridas. Estos tipos se diferencian en como son desarrolladas, lo que pueden hacer, como funcionan y como se distribuyen [CORPORATION, 2012].

3.6.4.- Aplicación Web.

Los dispositivos móviles incluyen navegadores web completamente funcionales, por lo que es posible acceder desde ellos a cualquier sitio web al que pueda accederse desde una computadora. Las aplicaciones web diseñadas para dispositivos móviles emplean los mismos componentes que las aplicaciones web tradicionales y acceden a los mismos datos a través de los mismos servidores. La única diferencia importante entre las aplicaciones web diseñadas para computadoras estándar y aquellas diseñadas para dispositivos móviles es como son representadas.

La principal ventaja con respecto a la aplicación nativa es la posibilidad de programar independiente del sistema operativo en el que se ejecutaría la aplicación. De esta forma se pueden ejecutar en diferentes dispositivos sin tener que crear varias aplicaciones. Las aplicaciones web se ejecutan dentro del propio navegador web del dispositivo a través de una URL.

3.6.5.- Aplicación Nativa.

Son aplicaciones descargadas y ejecutadas en dispositivos móviles. Desarrolladas de forma específica para un tipo de dispositivo y su sistema operativo, se basan en la instalación de código ejecutable en el dispositivo del usuario. Tienen la ventaja de acceder a las funciones del dispositivo, como por ejemplo: almacenamiento, GPS (sistema de posicionamiento global), SMS (servicio de mensajes cortos), mails. Ofrecen generalmente mejor rendimiento que las aplicaciones web ejecutadas en navegadores móviles y están mejor integradas con el hardware disponible.

Dentro de estas aplicaciones nativas se encuentran las aplicaciones hibridas, las cuales son aplicaciones que contienen componentes de navegador web que cargan y ejecutan aplicaciones web. Son un compromiso entre una aplicación web y una aplicación nativa. Con las aplicaciones hibridas, los desarrolladores pueden utilizar componentes de aplicación nativa para personalizar el aspecto y el manejo de la aplicación y componentes de aplicación web para ayudar a superar las limitaciones de actualización de las aplicaciones nativas.

3.7.- Android Studio.

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan tu productividad durante la compilación de apps para Android [developer android, 2016a], como las siguientes:

- Sistema de compilación fleexible basado en Gradle.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
- Instant Run, para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub, para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.

- Herramientas Lint para detectar problemas de rendimiento, uso, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte integrado para Google Cloud Platform, que facilita la integración de Google Cloud Messaging y App Engine.

3.7.1.- Depuración Integrada.

La depuración integrada para mejorar las revisiones de código en la vista del depurador con verificación integrada de referencias, expresiones y valores de variables [developer android, 2016c]. La información de depuración integrada incluye:

- Valores de variables integradas.
- Objetos que hacen referencia a un objeto seleccionado.
- Valores de retorno de métodos.
- Expresiones Lambda y de operador.
- Valores de información sobre herramientas.

3.7.2.- Estructura de un Proyecto.

Un proyecto de Android Studio contiene todo lo que define el lugar de trabajo para una app; desde código fuente y recursos hasta para código de prueba y configuraciones de compilación [developer android, 2016d].

Cada proyecto en Android Studio contiene uno o más módulos con archivos de código fuente y archivos de recursos. Entre los tipos de módulos se incluyen los siguientes:

- Módulos de apps para Android
- Módulos de bibliotecas
- Módulos de Google App Engine

De forma predeterminada, en Android Studio se muestran los archivos del proyecto en la vista de proyectos de Android, esta vista está organizada en módulos para acceder rápidamente a los archivos de origen claves del proyecto.

Todos los archivos de compilación son visibles en el nivel superior de Secuencias de comando de Gradle y cada módulo de la aplicación contiene las siguientes carpetas:

- Manifestó: contiene el archivo AndroidManifest.xml.
- Java: contiene los archivos de código fuente de Java, incluido el código de prueba JUnit.
- Res: Contiene todos los recursos, como diseños XML, cadenas de IU e imágenes de mapa de bits.

La estructura del proyecto para Android en el disco difiere de esta representación plana. También se puede personalizar la vista de los archivos del proyecto para concentrarse en aspectos específicos del desarrollo de la app.

3.7.3.- Sistema de Compilación Gradle.

Android Studio usa Gradle como base del sistema de compilación, y proporciona más características específicas de Android a través del Complemento de Android para Gradle. Este sistema de compilación se ejecuta en una herramienta integrada desde el menú de Android Studio, y lo hace independientemente de la línea de comandos [developer android, 2016e]. Usa las funciones del sistema de compilación para lo siguiente:

- Personalizar, configurar y extender el proceso de compilación.
- Crear varios APK para una app con diferentes funciones usando el mismo proyecto y los mismos módulos.
- Volver a usar códigos y recursos entre conjuntos de orígenes.

Recurriendo a la flexibilidad de Gradle, se puede lograr todo esto sin modificar los archivos de origen de una app. Los archivos de compilación de Android Studio se denominan build.gradle. Son archivos de texto sin formato que usan sintaxis Groovy para configurar la compilación con elementos proporcionados por el complemento de Android para Gradle.

Cada proyecto tiene un archivo de compilación de nivel superior para todo el proyecto y archivos de compilación de nivel de módulo independientes para cada módulo. Cuando importas un proyecto existente, Android Studio genera automáticamente los archivos de compilación necesarios.

3.7.4.- Módulos.

Un módulo es un conjunto de archivos de origen y configuraciones de compilación que permiten dividir un proyecto en unidades discretas de funcionalidad. Un proyecto puede tener uno o más módulos y un módulo puede usar otro como dependencia. Cada módulo se puede compilar, probar y depurar de forma independiente.

Los módulos adicionales a menudo son útiles cuando se crean bibliotecas de código dentro del propio proyecto o cuando se desean crear diferentes conjuntos de código y recursos para diferentes tipos de dispositivos, como teléfonos y wearables, y al mismo tiempo mantener todos los archivos almacenados dentro del mismo proyecto y compartir código [developer android, 2016b].

3.7.5.- Modulo de App para Android.

Proporciona un contenedor para el código fuente de una app, los archivos de recursos y las configuraciones de niveles de app como el archivo de compilación de nivel de módulo y el archivo de manifiesto de Android. Cuando creas un proyecto nuevo, el nombre del módulo predeterminado es “app".

Android Studio ofrece los siguientes módulos de app:

- Módulo de teléfono y Tablet.
- Módulo Android Wear.
- Módulo Android TV.

- Módulo Glass.

Cada uno proporciona archivos esenciales y algunas plantillas de códigos que son apropiadas para el tipo de dispositivo o app correspondiente.

3.7.6.- Modulo de Biblioteca.

Proporciona un contenedor para código reutilizable, que se puede usar como una dependencia en otros módulos de app o importar a otros proyectos. A nivel estructural, un módulo de biblioteca es lo mismo que un módulo de app, pero cuando se compila crea un archivo de código en lugar de un APK, de modo que no se puede instalar en un dispositivo. Android Studio ofrece los siguientes módulos de biblioteca:

Biblioteca de Android: Este tipo de biblioteca puede contener todos los tipos de archivos admitidos en un proyecto de Android, como código fuente, recursos y archivos de manifiesto. El resultado de compilación es un archivo Android Archive (AAR) que puedes agregar como dependencia para los módulos de app de Android.

Biblioteca Java: Este tipo de biblioteca puede contener solo archivos de origen Java. El resultado de compilación es un archivo de Java (JAR) que se puede agregar como dependencia para los módulos de app de Android u otros proyectos de Java.

3.8.- WAMPSERVER.

WAMP es un acrónimo formado a partir de las iniciales del sistema operativo Micros oft Windows y los principales componentes del paquete (Windows, Apache, MySQL,PHP):

Apache, MySQL y uno de PHP, Perl o Python. Apache es un servidor web. MySQL es una base de datos de código abierto. PHP es un lenguaje de scripting que puede manipular la información contenida en una base de datos y generar páginas web dinámicamente cada vez que un navegador solicita contenido [Dr. K.Baskara, 2012].

Capítulo 4.- Metodología.

Para el desarrollo de una aplicación móvil es necesario el utilizar una metodología de programación como si se tratara de un sistema a desarrollar para uso en PC, dicha metodología debe considerar las limitadas capacidades de procesamiento en los dispositivos móviles que se utilizaran para la ejecución de la aplicación. Por lo que se optó por el uso de una metodología iterativa ya que este tipo de metodología permite el desarrollo de la aplicación por periodos, lo que facilita las pruebas y el corregir errores en el código de la aplicación.

4.1.- Planeación.

Debido a la heterogeneidad de los dispositivos móviles se requiere que el llevar a cabo una cuidadosa planificación y diseño antes de la implementación de la aplicación. Ejecutar una aplicación en un dispositivo móvil introduce una serie de consideraciones que un desarrollador/programador debe conocer:

Distintas velocidades y características de la red, al ser dispositivos móviles atravesarán distintas redes. Errores de red. La falta de una red de datos disponible es muy probable, en estos casos la aplicación debe tener un modo fuera de línea. Variación del rendimiento de la plataforma de hardware. Lo ideal es que la aplicación esté disponible para tantos dispositivos como sea posible. Esto significa soportar diferentes dispositivos y diferentes plataformas. Distintos tamaños y resoluciones de pantallas. Los diferentes dispositivos cuentan con diferentes pantallas con distintas funcionalidades. Difícil de probar las aplicaciones por completo. Dada la variedad de dispositivos, se torna difícil probar todos los dispositivos actuales y los nuevos que ingresan al mercado.

4.2.- Metodología a Utilizar.

Para la realización del proyecto planteado sea optado por utilizar la metodología Scrum ya que Scrum es adecuado para aquellas empresas en las que el desarrollo de los productos se realiza en entornos que se caracterizan por tener:

- Incertidumbre: Sobre esta variable se plantea el objetivo que se quiere alcanzar sin proporcionar un plan detallado del producto. Esto genera un reto y da una autonomía que sirve para generar una “tensión" adecuada para la motivación de los equipos.
- Auto-organización: Los equipos son capaces de organizarse por sí solos, no necesitan roles para la gestión pero tienen que reunir las siguientes características: Autonomía, Auto-superación, Auto-enriquecimiento.
- Control moderado: Se establecerá un control suficiente para evitar descontroles. Se basa en crear un escenario de “autocontrol entre iguales" para no impedir la creatividad y espontaneidad de los miembros del equipo.
- Transmisión del conocimiento: Todo el mundo aprende de todo el mundo. Las personas pasan de unos proyectos a otros y así comparten sus conocimientos a lo largo de la organización.

Scrum al ser una metodología de desarrollo ágil tiene como base la idea de creación de ciclos breves para el desarrollo, que comúnmente se llaman iteraciones y que en Scrum se llamarán “Sprints" o “iteración". [Gallego, 2012b].

Figura 4.1.- Ciclo de desarrollo ágil [Gallego, 2012a].

Para entender el ciclo de desarrollo de Scrum es necesario conocer las 5 fases que definen el ciclo de desarrollo ágil fig.4.1:

Concepto: Se define de forma general las características del producto y se asigna el equipo que se encargará de su desarrollo.

Especulación: en esta fase se hacen disposiciones con la información obtenida y se establecen los límites que marcaran el desarrollo del producto, tales como costes y agendas. Se construirá el producto a partir de las ideas principales y se comprueban las partes realizadas y su impacto en el entorno. Esta fase se repite en cada iteración y consiste en rasgos generales, en: Desarrollar y revisar los requisitos generales, Mantener la lista de las funcionalidades que se esperan, Plan de entrega. Se establecen las fechas de las versiones, hitos e iteraciones. Medirá el esfuerzo realizado en el proyecto.

Exploración: Se incrementa el producto en el que se añaden las funcionalidades de la fase de especulación.

Revisión: El equipo revisa todo lo que se ha construido y se contrasta con el objetivo deseado.

Cierre: Se entregará en la fecha acordada una versión del producto deseado. Al tratarse de una versión, el cierre no indica que se ha finalizado el proyecto, sino que seguirá habiendo cambios, denominados “mantenimiento", que hará que el producto final se acerque al producto final deseado.

4.2.1.- Etapas de Scrum.

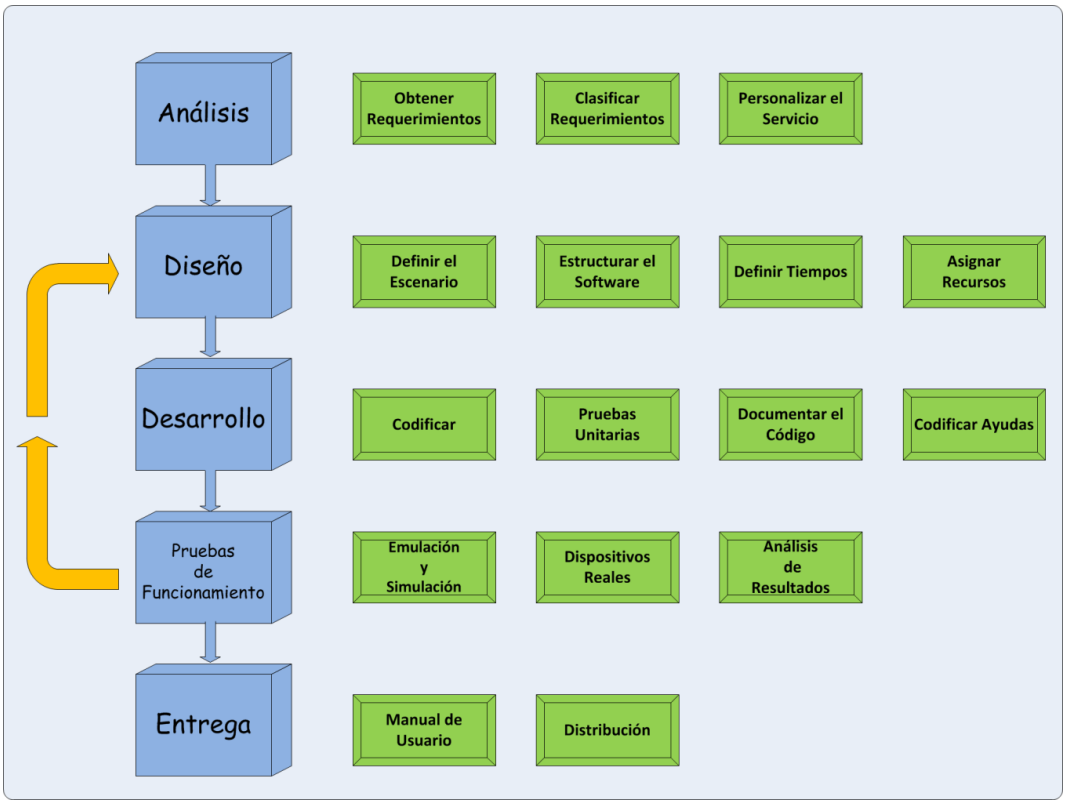

Figura 4.2.- Etapas de la metodología [Maira Cecilia Gasca Mantilla, 2013a].

4.2.1.1.- Análisis.

En esta fase se analizan las peticiones o requerimientos de las personas o entidad para la cual se desarrolla el servicio móvil “Cliente", el propósito es definir las características del mundo o entorno de la aplicación. Se realizan tres tareas: obtener requerimientos, clasificar los requerimientos y personalizar el servicio [Maira Cecilia Gasca Mantilla, 2013b].

Obtener requerimientos: se sugiere hacer una serie de entrevistas al cliente, para que manifieste los síntomas del problema o necesidades que se pretenden solucionar con las tecnologías móviles, o simplemente, para que señale las características que debe tener la aplicación.

Clasificar los requerimientos: una vez identificados los requerimientos que debe tener el software, se procede a clasificarlos. Dichos requerimientos se pueden dividir en entorno, mundo, funcionales y no funcionales:

- El entorno se refiere a todo lo que rodea al servicio.
- El mundo es la forma como interactúan el usuario y la aplicación.
- Los requerimientos funcionales son todos aquellos que demandan una función dentro del sistema. Se deben definir claramente cada una de las tareas que debe realizar la aplicación.
- Los requerimientos no funcionales son la estabilidad, la portabilidad, el rendimiento, el tiempo de salida al mercado y el costo.
- Personalizar el servicio: adicionalmente se deben analizar aspectos de la cotidianidad del cliente como preferencias, costumbres y particularidades del usuario, con el propósito de garantizar la aceptación del servicio.

4.2.1.2.- Diseño.

El objetivo de esta etapa es plasmar el pensamiento de la solución mediante diagramas o esquemas, considerando la mejor alternativa al integrar aspectos técnicos, funcionales, sociales y económicos. A esta fase se retorna si no se obtiene lo deseado en la etapa prueba de funcionamiento [Maira Cecilia Gasca Mantilla, 2013b].

Se realizan cuatro actividades en esta fase: definir el escenario, estructurar el software, definir tiempos y asignar recursos. Definir el escenario: las aplicaciones móviles se pueden diseñar para ejecutarse en diferentes escenarios, dependiendo del sistema de conexión y sincronización con el servidor o aplicación central.

Estructurar el software: se deben utilizar algunos diagramas de Modelado de Lenguaje Unificado (UML), es decir, traducir los requerimientos obtenidos de la etapa anterior en un diagrama que describa en forma objetiva el servicio por implementar.

Definir tiempos: se establecen los plazos para cada una de las actividades restantes, con el objetivo de terminar la aplicación a tiempo para su salida al mercado. Se debe tener en cuenta el diseño computacional del software realizado en la tarea anterior y, las características volátiles y dinámicas de los servicios móviles.

Asignar recursos: se asignan los recursos para realizar cada actividad y alcanzar los objetivos propuestos, se deben considerar recursos humanos, financieros y tecnológicos. Además, se deben seleccionar las herramientas para el desarrollo de la aplicación móvil.

4.2.1.3.- Desarrollo.

El objetivo de esta fase es implementar el diseño en un producto de software. En esta etapa se realizan las siguientes actividades [Maira Cecilia Gasca Mantilla, 2013c]:

Codificar: se escribe en el lenguaje de programación seleccionado, cada una de las partes definidas en los diagramas realizados en la etapa de diseño.

Pruebas unitarias: se verifica el funcionamiento de la aplicación.

Documentar el código: a medida que se codifica y se prueba cada elemento, se redacta la pequeña documentación sobre lo desarrollado.

Codificar ayudas: además del manual de instalación y de usuario, deben existir una serie de ayudas que informen de manera didáctica lo que puede hacer el usuario con la aplicación.

4.2.1.4.- Pruebas de Funcionamiento.

El objetivo de esta fase es verificar el funcionamiento de la aplicación en diferentes escenarios y condiciones; para esto se realizan las siguientes tareas:

Emulación y simulación: se realizan pruebas simulando el escenario y emulando el dispositivo móvil, explorando todas las utilidades y funciones de la aplicación, introduciendo diferentes datos, inclusive erróneos, para medir la funcionalidad y el nivel de robustez del software. Si se encuentran algunas fallas, se debe regresar a la etapa de codificación en la fase de desarrollo

para solucionar los problemas, si las pruebas son satisfactorias se procede a la etapa de pruebas con dispositivos reales.

Dispositivos reales: deben hacerse pruebas de campo en equipos reales para medir el desempeño y el rendimiento del aplicativo. Si se encuentran fallas en el tiempo de ejecución, si el software no cumple con los requerimientos especificados, o si el cliente solicita un cambio de última hora, hay que regresar a la fase de diseño para reestructurar y solucionar el inconveniente presentado.

Análisis de las 6 M's [Maira Cecilia Gasca Mantilla, 2013a]: para valorar el potencial de éxito del servicio, es decir, los seis atributos que se miden para evaluar el éxito del servicio propuesto: Movement (Movimiento), Moment (Momento), Me (Yo), Multi-user (Multiusuario), Money (Dinero) y Machines (Máquinas).

4.2.1.5.- Entrega.

Terminada la depuración de la aplicación y atendidos todos los requerimientos de última hora del cliente se da por finalizada la aplicación y se procede a la entrega del ejecutable, el código fuente, la documentación y el manual del sistema [Maira Cecilia Gasca Mantilla, 2013d].

Manuales: el objetivo es el entrenamiento; una aplicación móvil debe constar de un manual del sistema donde se indique el proceso de instalación, la atención a posibles fallas en el tiempo de ejecución y, las especiaciones técnicas mínimas de hardware y software que requiere el equipo, para el funcionamiento adecuado del aplicativo desarrollado.

Distribución: se define el canal de comercialización de la aplicación, con el propósito de adecuar la aplicación al medio de distribución.

4.3.- Viabilidad.

Para la correcta elaboración de este proyecto se deben de tomar en cuenta varios aspectos, como los recursos humanos, materiales, la disponibilidad de tiempo, los requerimientos del sistema y las funciones solicitadas por el cliente, por lo que se realiza un análisis de la viabilidad del proyecto.

4.3.1.- Requerimientos de Sistema.

Desde el punto de vista del cliente, se tomará como referencia la opinión del Lic. Jorge Sánchez Ayala, gerente del restaurante Sanborns, para que el sistema sea viable debe de brindar una solución a los problemas de la empresa.

El sistema a realizar debe permitir tener un control de los registros de las órdenes generadas, así como el estado de las mismas en tiempo real, por lo que al tomar en cuenta esas condiciones se han determinado los requerimientos de funcionalidad en el sistema, así como los no funcionales que corresponden a requerimientos de datos, de rendimiento y restricciones del sistema.

4.3.1.1.- Requerimientos Funcionales.

Inicio de sesión

- Al ejecutar el sistema tanto en la pc como en el dispositivo móvil debe desplegar una ventana en la que se ingresen los datos de inicio de sesión para el acceso al sistema.
- Si los datos ingresados son correctos, el sistema identifica al usuario habilitando los privilegios correspondientes a cada tipo de usuario.
- Si los datos ingresados son incorrectos el sistema debe desplegar una ventana en la que indique el error.

Seguridad

- Creación de nuevos usuarios con acceso al sistema.
- Asignación de privilegios a los usuarios basado en el tipo de usuario correspondiente.
- Asignación de contraseña a los usuarios del sistema.

Gestión de datos del personal

- El sistema debe permitir el guardar, modificar, consultar y eliminar información del personal.

Gestión de datos de los menús

- El sistema debe permitir el guardar, modificar, consultar y eliminar información de los menús de los productos.

Supervisión en tiempo real

- El sistema debe abrir una ventana en la que despliegue el contenido de las órdenes, así como el estado de las mismas.
- Cada orden generada debe de reflejar al usuario de que la genero en el sistema.

Gestión de Ordenes

- El usuario de piso de venta debe poder realizar registro de órdenes, así como su consulta
- El usuario correspondiente a la cocina debe poder consultar y manipular el estado de las órdenes.
- El usuario supervisor debe poder realizar todas las operaciones anteriores

4.3.1.2.- Requerimientos de Datos.

- El sistema debe almacenar información del local.
    - Nombre.
    - Dirección.
    - Información de contacto.
- Debe almacenar la información de los empleados.
    - Nombre.
    - Apellido Paterno.
    - Apellido Materno.
    - Contraseña.
    - Tipo de usuario.

- Debe almacenar los nombres de los departamentos.
- Debe almacenar información de los platillos.
    - Nombre del platillo.
    - Descripción del platillo.
    - Imagen del platillo.
    - Precio del platillo.
- Debe almacenar la información de las bebidas.
    - Nombre de la bebida.
    - Descripción de la bebida.
    - Imagen de la bebida.
    - Precio de la bebida.
- Debe contener información de las órdenes realizadas.
    - Estado de las órdenes.
    - Contenido de la orden.
    - Usuario que realizo la orden.
    - Precio total de la orden.

4.3.1.3.- Requerimientos de rendimiento.

- El sistema debe de registrar las órdenes generadas por los usuarios.
- Las consultas realizadas por el sistema no debe tener un retraso mayor a los 5 seg.
- El sistema debe realizar las funciones de registro en la operación de piso de ventas.
- El sistema debe realizar una notificación al usuario autor de la orden.

4.3.1.4.- Restricciones.

Para la operación del prototipo a realizar el sistema cuenta con las siguientes restricciones:

- El sistema debe ejecutarse en dispositivos móviles, así como en pc.
- El sistema debe de ser compatible con el sistema operativo Windows 7 o superior.
- El sistema debe de ser compatible con el sistema operativo Android 5.0 o superior.
- Debe de ser creado en el lenguaje de programación Java.
- El sistema debe limitarse a su uso en las operaciones de piso e ventas.

4.3.2.- Modelado.

Analizando los requerimientos establecidos anteriormente, se procede a realizar el modelado del sistema para comenzar a desarrollar el diseño del mismo, los escenarios en los que el sistema de desenvolverá y definir los usuarios y sus restricciones en el uso del sistema.

4.3.2.1.- Análisis de caso de uso.

Actores

- Administrador. Es el usuario que posee todos los privilegios del sistema, realizando operaciones como lectura, modificación y eliminación de los datos almacenados en la base de datos.

- Supervisor. Este usuario debe contar con algunos privilegios restringidos, permitiéndole únicamente ingresar datos y modificar los realizados por el mismo u otros usuarios.
- Operador. Este usuario debe contar con privilegios restringidos para las operaciones del sistema, pudiendo realizar únicamente operaciones como registros en la base de datos, mas no la eliminación o modificación de los registros realizados por el u otros usuarios.
- Cocinero. Es el usuario con los privilegios más restringidos ya que solo contara con la función de modificar el registro de los estados de las órdenes generadas por los operadores.

4.3.2.2.- Diagramas de Casos de uso.

Inicio de sesión

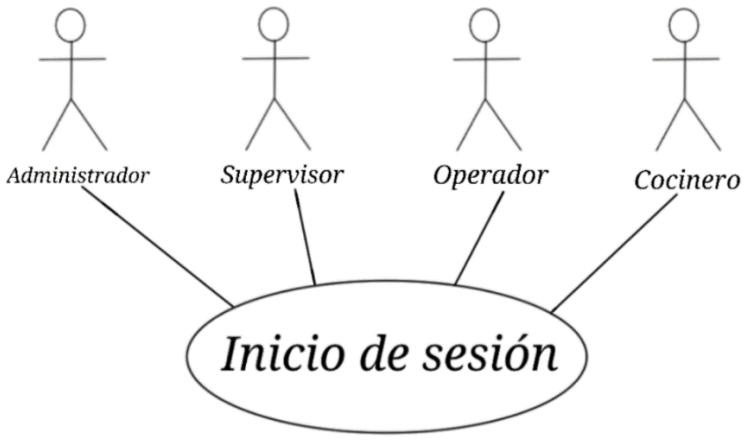

Figura 4.3.- Diagrama de caso de uso para el inicio de sesión.

Nombre	Actores	Descripción
Inicio de Sesión.	Administrador, supervisor, operador, cocinero.	Los usuarios con perfiles de administrador, supervisor, operador y cocinero.

Tabla 4.1.- Descripción de caso de uso inicio de sesión.

Seguridad

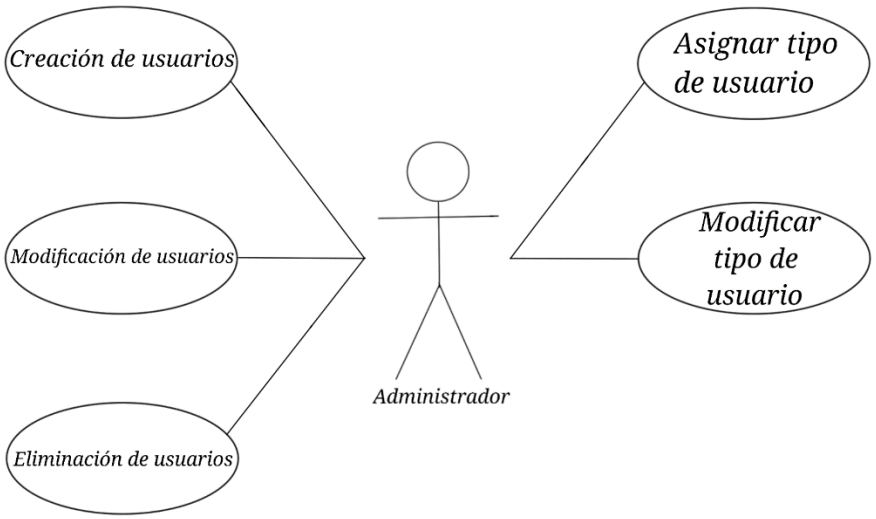

Figura 4.4.- Caso de uso para la gestión de seguridad.

Nombre	Actores	Descripción
Creación de usuarios	Administrador	El usuario con perfil de administrador pude crear nuevos usuarios del sistema.
Asignar tipo de usuario	Administrador	El usuario con perfil de administrador debe asignar el tipo de usuario, al generado, lo que permite definir su perfil.
Modificación de usuarios	Administrador	El usuario con perfil de administrador puede modificar la información de los usuarios existentes.
Modificar tipo de usuario	Administrador	El usuario con perfil de administrador puede modificar el tipo de usuario existentes asignados.
Eliminación de usuarios	Administrador	El usuario con perfil de administrador puede eliminar a los usuarios existentes.

Tabla 4.2.- Descripción de caso de uso gestión de seguridad.

Gestión de datos de los menús

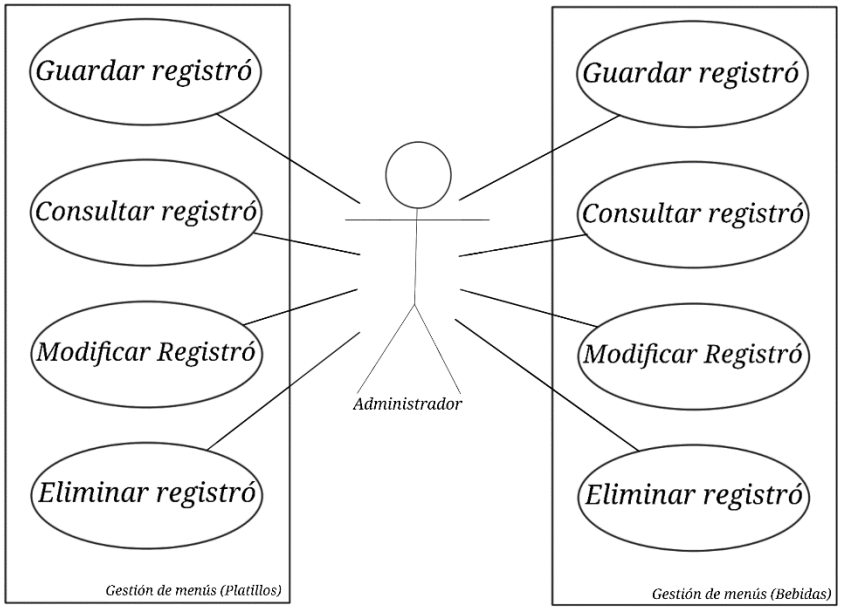

Figura 4.5.- Caso de uso para la gestión de menús.

Nombre	Actor	Descripción
Guardar registro	Administrador	El usuario con perfil de administrador puede agregar nuevos registros a los menús de bebidas y platillos.
Consultar registro	Administrador	El usuario con perfil de administrador puede consultar la información existente de los registros en los menús de platillos o bebidas.
Modificar registro	Administrador	El usuario con perfil de administrador puede modificar la información existente en los registros de los menús de platillos o bebidas.
Eliminar registro	Administrador	El usuario con perfil de administrador puede eliminar los registros existentes de los menús de platillos o bebidas.

Tabla 4.3.- Descripción de caso de uso gestión de menús.

Gestión de Ordenes

Operador-cocinero.

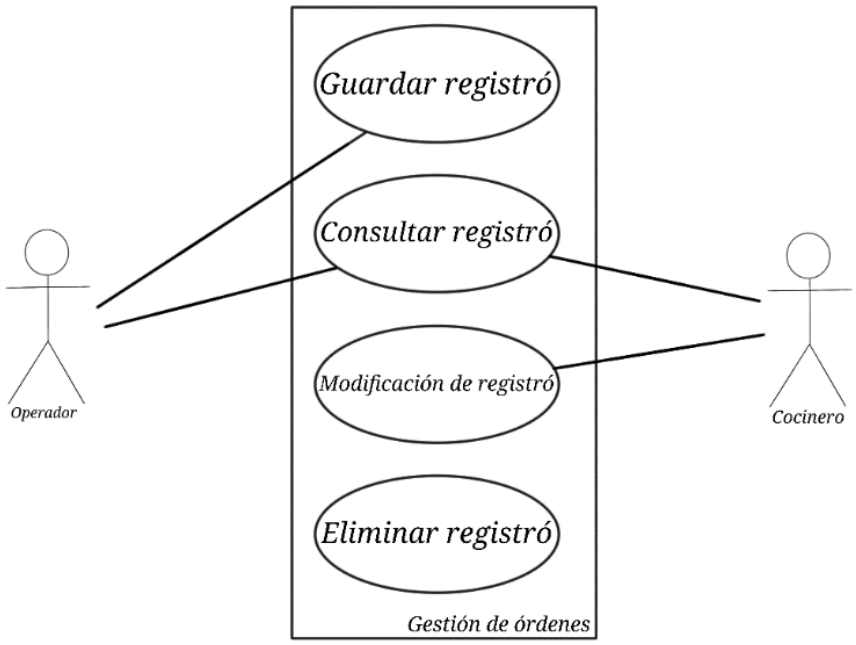

Figura 4.6.- Caso de uso de gestión de órdenes operador-cocinero.

Nombre	Actores	Descripción
Guardar Registro	Operador	El usuario con perfil de operador, cuenta con la opción de realizar un registro en la gestión de órdenes.
Consultar Registro	Operador	El usuario con perfil de operador, puede consultar los registros en la gestión de órdenes.
Consultar Registro	Cocinero	El usuario con perfil de cocinero, puede consultar los registros de la gestión de órdenes.
Modificación de registro	Cocinero	El usuario con perfil de cocinero, puede realizar la modificación en los registros de estado de las órdenes.

Tabla 4.4 .- Descripción de caso de uso gestión de órdenes operador-cocinero.

Administrador-Supervisor.

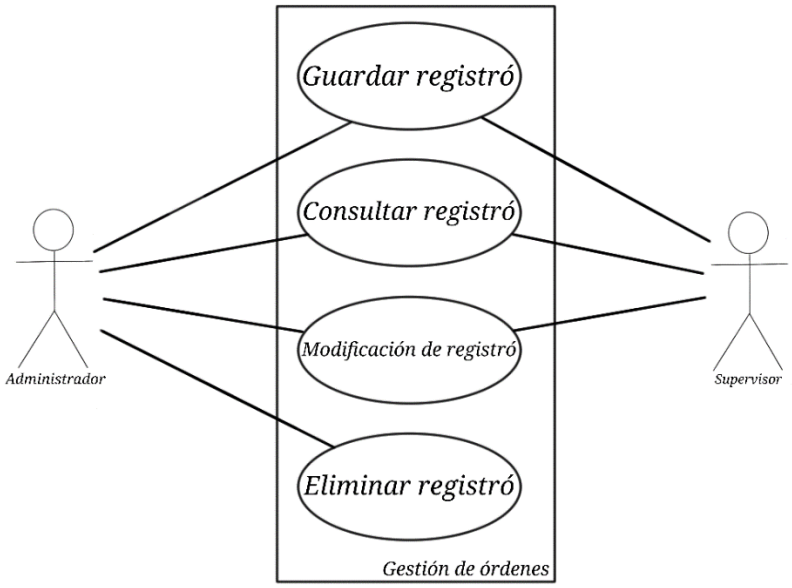

Figura 4.7.- Caso de uso de gestión de órdenes Administrador-Supervisor.

Nombre	Actores	Descripción
Guardar Registro	Administrador	El usuario con perfil de administrador puede guardar registros en la gestión de órdenes.
Guardar Registro	Supervisor	El usuario con perfil de supervisor puede guardar registros en la gestión de órdenes.
Consultar Registro	Administrador	El usuario con perfil de administrador puede consultar los registros existentes dentro de la gestión de órdenes.
Consultar Registro	Supervisor	El usuario con perfil de supervisor puede consultar los registros existentes dentro de la gestión de órdenes.
Modificación de registro	Administrador	El usuario con perfil de administrador pude modificar la información existente dentro de la gestión de órdenes.
Modificación de registro	Supervisor	El usuario con perfil de supervisor pude modificar la información existente dentro de la gestión de órdenes.
Eliminar Registro	Administrador	El usuario con perfil de administrador será el único con la capacidad de eliminar los registros de la gestión de órdenes.

Tabla 4.5.- Descripción de Caso de uso de gestión de órdenes Administrador-Supervisor.

4.3.2.3.- Diagrama de Clases.

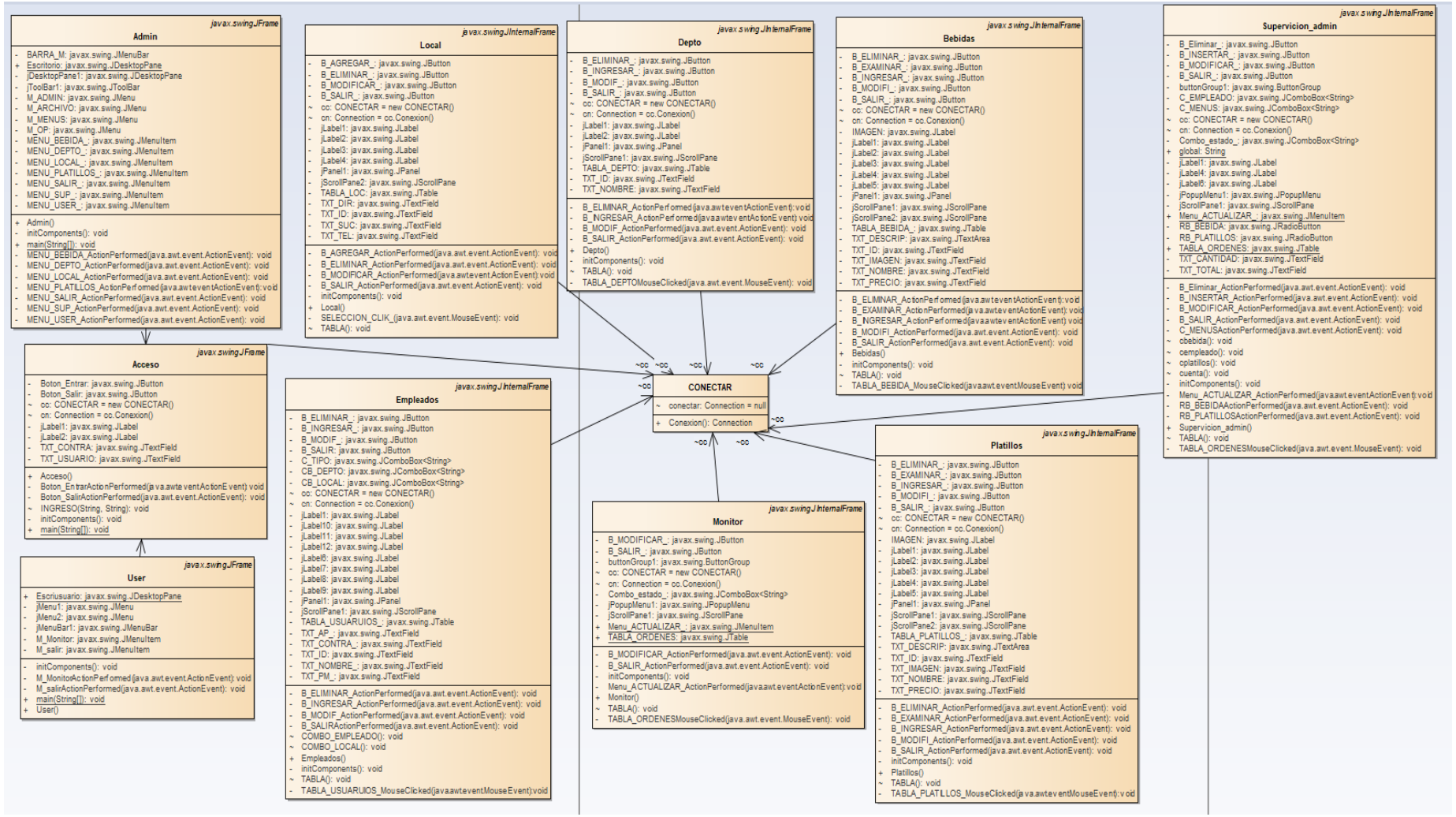

Figura 4.8.- Diagrama de clases.

Como se puede observar en la figura 4.8, se muestran las clases responsables de las funciones que el sistema realiza, en ellas se pueden observar elementos visibles y los procesos no visibles que realizan.

4.3.2.4.- Clases.

CONECTAR.

Esta clase es la responsable de la conexión a la base de datos, por lo que es utilizada por todas las clases para realizar todas las funciones necesarias para la operación del sistema.

Acceso.

Es la clase responsable del ingreso al sistema, reconociendo al usuario por su clave de ingreso, desplegando como resultado la siguiente clase correspondiente a dicho usuario.

User.

La clase User es la ventana principal correspondiente al usuario *Cocina*, el cual solo posee un único privilegio dentro del sistema y desde esta puede acceder a la única función disponible.

Admin.

La clase Admin es la ventana principal del usuario *Administrador*, el cual posee todos los privilegios dentro del sistema y desde esta puede acceder a todas las funciones.

Local.

Es la clase que contiene las funciones para la gestión del establecimiento, en caso de contar con más de uno la clase permite agregarlos.

Depto.

Es responsable de la gestión de los departamentos internos del establecimiento, que después deben ser asignados a los empleados cuando son registrados.

Empleados.

Clase responsable de la gestión del manejo de la información de los empleados, en esta se registran los empleados que tendrán acceso al sistema, los privilegios con los que deben contar.

Bebidas

Clase que contiene la información de las bebidas que son elaboradas u ofrecidas dentro del establecimiento, permitiendo realizar acciones como el agregar, eliminar o modificar los registros en la base de datos.

Platillos.

Clase responsable de gestionar la información de los platillos que son preparados dentro del establecimiento, permitiendo el ingreso de un nuevo registro, modificar los registros existentes o eliminarlos.

Supervicion_admin.

Clase que monitorea el flujo de las órdenes que son generadas desde piso de venta, el usuario *Administrador*, en esta clase cuenta con los privilegios necesarios para el ingreso de nuevas órdenes, esto último es necesario en caso de producirse una contingencia ajena a las funciones del sistema. La clase permite la modificación de las órdenes, siendo este usuario el único capaz de realizar una cancelación de la orden y de ser necesario el borrarla de los registros.

Monitor.

Esta clase es exclusiva del usuario Cocina, la cual permite monitorear las órdenes generadas, además cuenta con el único privilegio asignado al usuario, el cual es modificar el estado de las órdenes generadas.

4.3.2.4.- Diagrama de Secuencia.

Ingreso al Sistema.

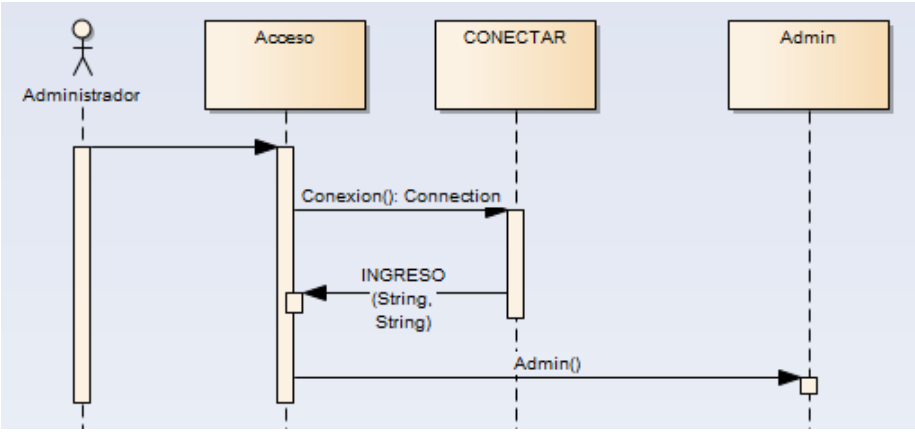

Figura 4.9.- Despliegue del acceso del Administrador al sistema.

En el caso del usuario *Administrador* se inicia ingresando los datos necesarios para el ingreso, con lo que mediante el método *Conexión ()*, compara los datos ingresados con los contenidos en la base de datos, con lo que el método *INGRESO ()*, después de verificar los datos despliega la ventana que en este caso es la correspondiente al *Administrador*, habilitando todas sus funciones.

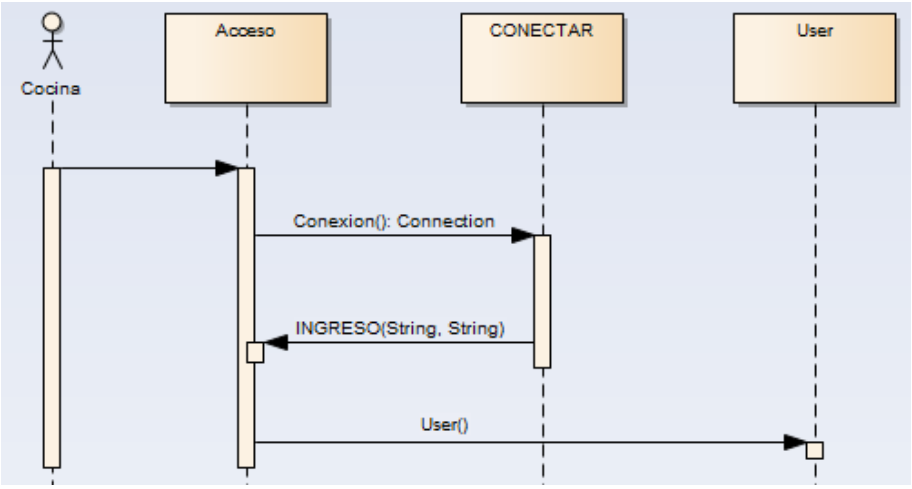

Figura 4.10.- Despliegue del ingreso de usuario Cocina al sistema.

En el caso del acceso al sistema por parte del usuario *Cocina*, el sistema se comporta igual que el caso del *Administrador*, con la diferencia de que el sistema despliega la ventana que el corresponde, junto con la única función con la que el usuario cuenta.

Gestión de Locales.

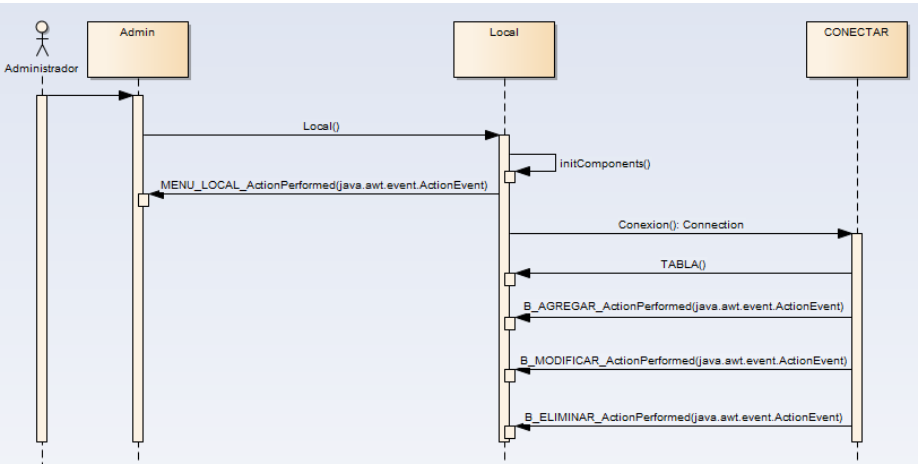

Figura 4.11.- Despliegue de la Gestión de locales.

Continuando con el usuario *Administrador*, el cual una vez identificado como tal ingresa a la pantalla principal (*Admin*), en la cual puede acceder al formulario correspondiente a la gestión de los locales (en caso de contar con más de uno) por medio del método generado en el menú ítem (*MENU_LOCAL*).

Una vez ingresado al formulario se comienza con la inicialización de los componentes, por lo que se conecta con la base de datos con lo que el método *TABLA* () puede acceder a la información que se despliega en el formulario, la cual por medio de los métodos generados en los botones B_AGREGAR, B_MODIFICAR y B_ELIMINAR, es manipulada para agregar, modificar y eliminar registros respectivamente.

Gestión de Departamentos.

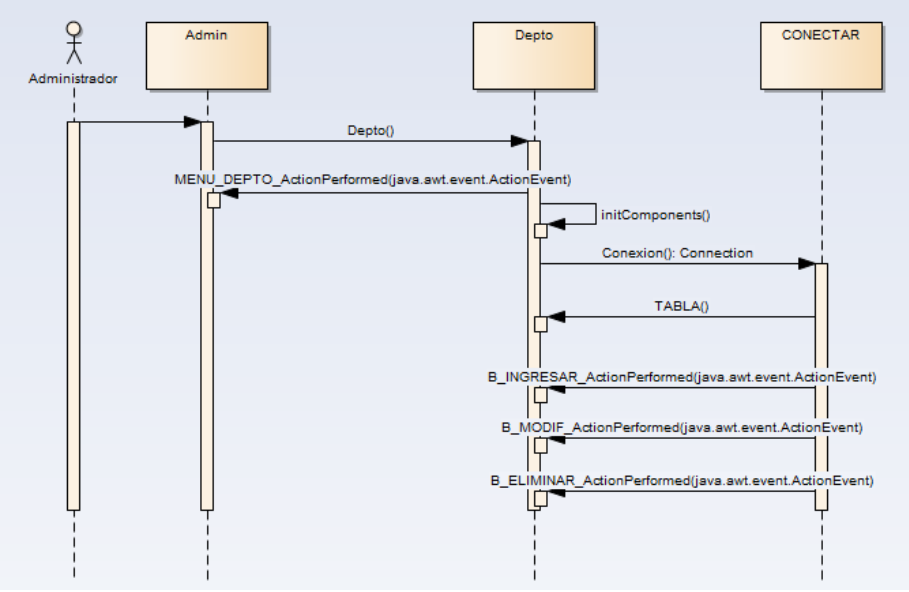

Figura 4.12.- Despliegue de la gestión de departamentos.

Desde el formulario *Admin* se accede por medio del menú ítem de *MENU_DEPTO_* al formulario correspondiente a la gestión de la información de los departamentos, inicializando los componentes y la conexión a la base de datos, así con el método de *TABLA* () se visualiza la información en el formulario, facilitando la manipulación del contenido de la tabla correspondiente a los *departamento* mediante el uso de los métodos generados en los botones de acción B_INGRESAR_, B_MODIF_ Y B_ELIMINAR_.

Gestión de Empleados.

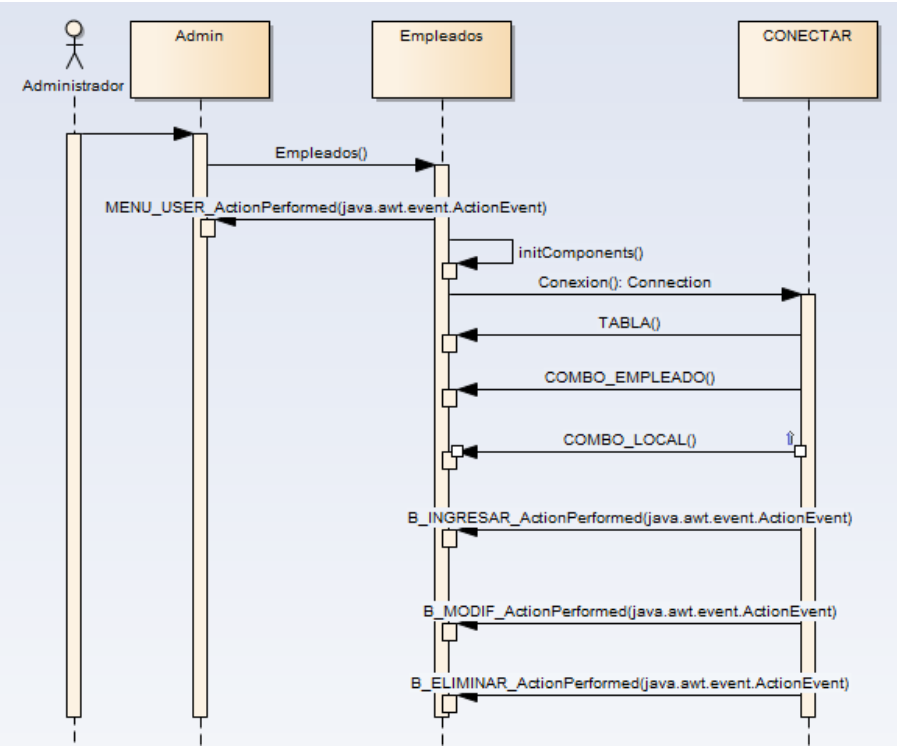

Figura 4.13.- Despliegue de la gestión de empleados.

Desde el formulario *Admin* el usuario *Administrador* accede al correspondiente a la gestión de los empleados que laboran dentro del establecimiento, por lo que el formulario al ingresar inicializa sus componentes accediendo a la base de datos con lo que, por medio del método

TABLA () el formulario muestra el contenido de la tabla *empleado*, después se cargan los contenidos de los JComboBox correspondientes al o los locales y a los departamentos.

Ya cargados los componentes iniciales mediante los métodos generados en los botones de acción (B_INGRESAR_, B_MODIF_ Y B_ELIMINAR_) con los que se manipulara la información contenida en la tabla *empleado*.

Gestión de Platillos y Bebidas

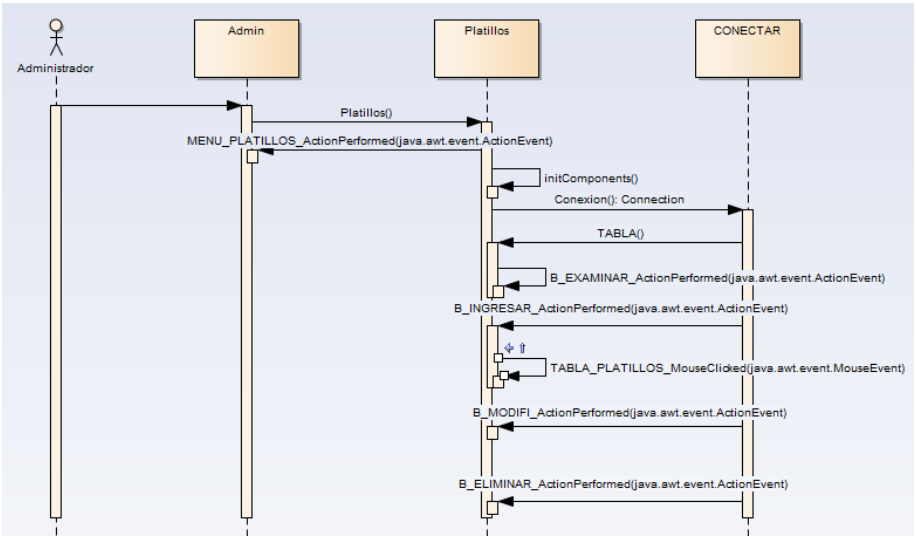

Figura 4.14.- Despliegue de la gestión de platillos.

Desde el formulario *Admin* el usuario *Administrador* puede gestionar el menú de los platillos que se preparan dentro del establecimiento, por lo que una vez seleccionado el menú ítem *MENU_PLATILLOS_* el evento creado en el permite el acceso al formulario, el cual inicializa sus componentes, conectando a la base de datos extrayendo la información que se despliega, para visualizar todos los cambios que se realicen a su contenido.

Para realizar el ingreso de un registro a la base de datos, se necesita forzosamente de una imagen que acompañe a la información, por lo que con el método *B_EXAMINAR_* se realiza la búsqueda de la imagen, seleccionando dicha imagen de cualquier ubicación de la pc en la que se ejecute esa acción, posteriormente ya que se cuente con la imagen necesaria se ejecuta el método creado en el botón de ingreso (*B_INGRESAR_*).

Para realizar alguna modificación a los registros de la base de datos es necesario acceder a la imagen que fue guardada en dicha ubicación, para eso se ejecuta el método *TABLA_PLATILLOS_*, con el que no solo se extrae la información del registro, sino que también se extrae la imagen que se guardó.

Ya contando con la información se procede a la modificación, esto se logra mediante el uso del método *B_MODIFI_*, el cual está contenido en el botón de acción al que corresponda dicho procedimiento, al igual que con el proceso de borrado el cual está contenido en su respectivo botón de acción (*B_ELIMINAR_*).Los mismos procedimientos aplican con la gestión de las bebidas, tal como puede verse con la fig. 4.15.

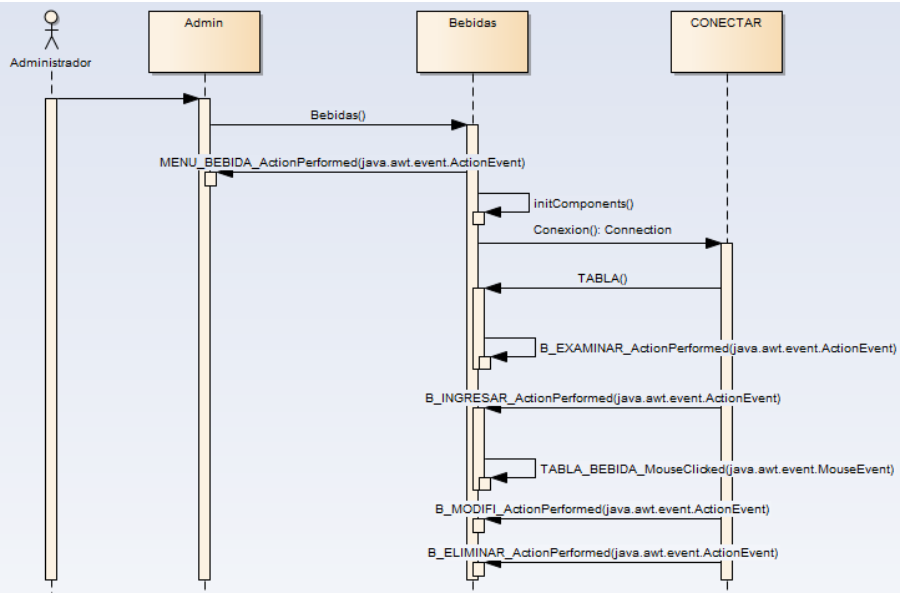

Figura 4.15.- Despliegue de la gestión de bebidas.

Creación de órdenes.

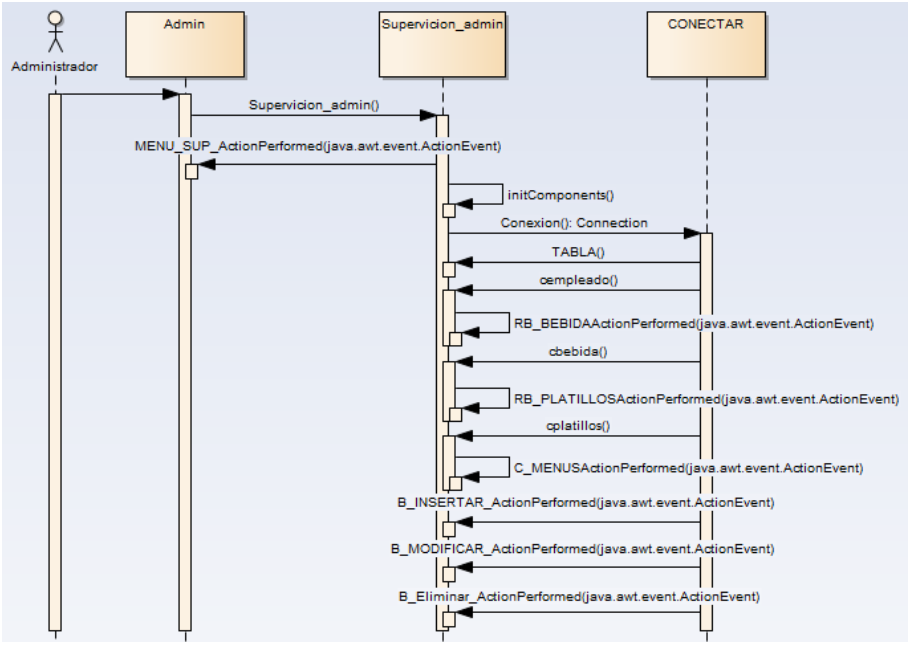

Figura 4.16.- Despliegue de la generación de órdenes para Administrador.

Después de haber ingresado como administrador al sistema, desde el formulario *Admin* se accede al formulario de *Supervicion_admin*, el cual inicializa sus componentes iniciales, conectándose a la base de datos y cargando el contenido del ComboBox correspondiente a los empleados del establecimiento, posteriormente, dependiendo de la selección de uno de los RadianButton será el ComboBox que se cargue, seguido de la utilización del método *C_MENUS*, por el cual se realizan las operaciones para el cálculo del total de la orden multiplicando la cantidad por el precio del platillo o la bebida.

Con los procedimientos anteriores se posee la información necesaria para el poder guardar el registro a la base de datos, mediante el uso del método *B_INSERTAR_*, el cual guardara el registro automáticamente con el estado de *Abierta*, en el caso de querer modificar el contenido de una fila (*B_MODIFICAR_*), el sistema no permite dicha acción, ya que el contenido de la orden una vez capturada no puede ser modificada, únicamente permite modificar el *estado* de la orden para indicar el estado en el que se encuentra, por lo que en caso de querer tener la necesidad de eliminar una orden, únicamente el *Administrador* cuenta con los privilegios para realizar dicha acción.

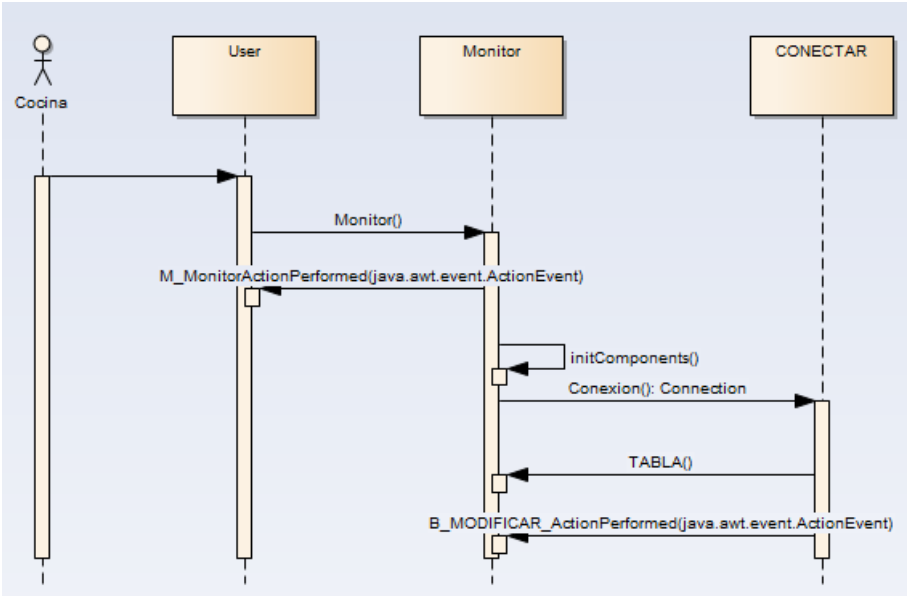

Figura 4.17.- Despliegue de la modificación de órdenes en usuario Cocina.

En el caso de acceder como el usuario Cocina, este únicamente cuenta con la opción de modificar el estado de la orden.

4.3.3.- Modelo Entidad-Relación.

Para el correcto funcionamiento del sistema se debe contar con una base de datos que almacene la información necesaria para le operación de piso de venta, por lo que el siguiente diagrama entidad relación muestra la base de datos anterior mencionada.

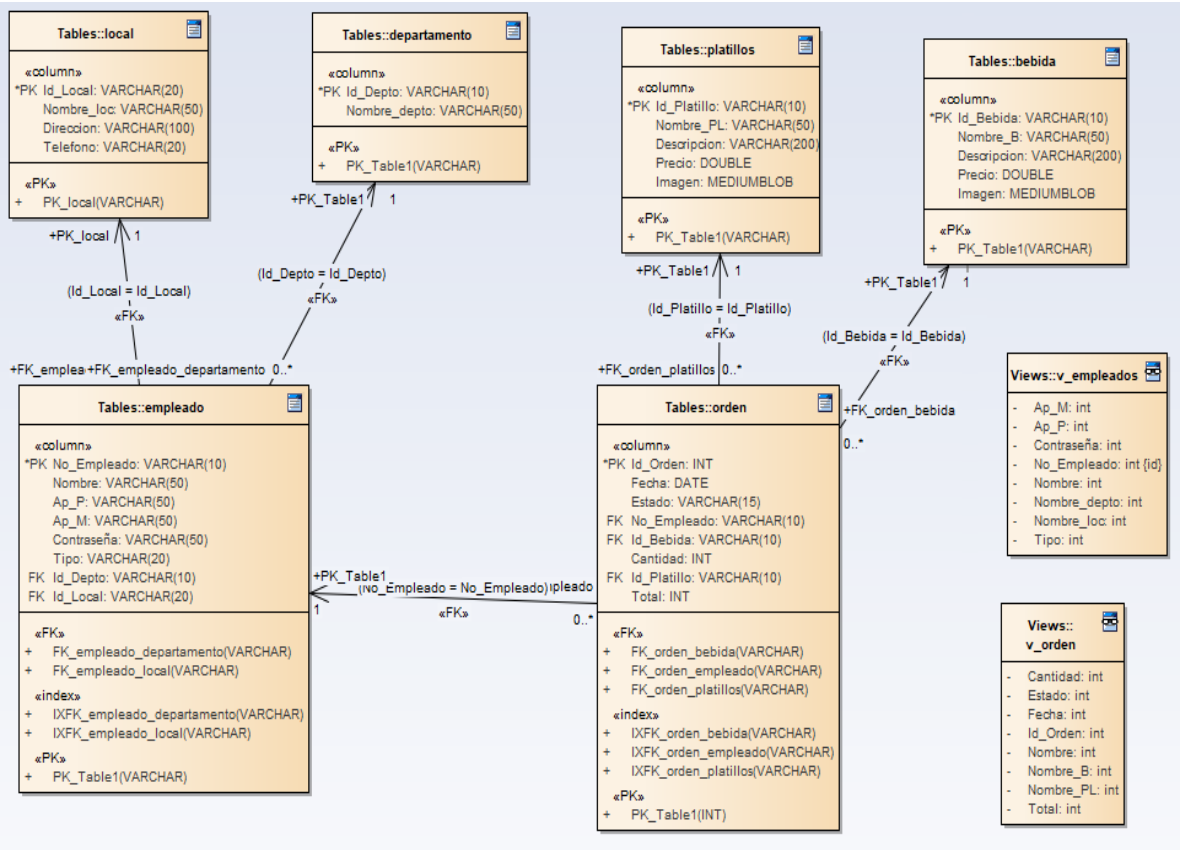

Figura 4.18.- Diagrama entidad relación de la base de datos.

4.3.3.1.- Diccionario de Datos.

Local

Esta tabla cuenta con la información relacionada con el local o locales (en caso de contar con más de uno) en el que se implementa el sistema.

Nombre	Tipo de datos	Not null	Descripción
Id_Local	VARCHAR (20) PK	x	Identificador del local.
Nombre_loc	VARCHAR (50)	x	Nombre del local o nombre interno del mismo (en caso de contarlo).
Dirección	VARCHAR (100)	x	Dirección física del local.
Teléfono	VARCHAR (20)	x	Número telefónico de contacto.

Tabla 4.6.- Tabla Local detallada.

Departamento

Esta tabla contiene la información correspondiente a los departamentos internos del local, el nombre de los departamentos que se almacenaran en la tabla deben ser únicos y el único usuario que puede manipular la información contenida en esta tabla es el administrador.

Nombre	Tipo de datos	Not null	Descripción
Id_Depto	VARCHAR (10) PK	x	Identificador del departamento.
Nombre_depto	VARCHAR (50)	x	Nombre del departamento dentro del local.

Tabla 4.7.- Tabla Departamento detallada.

Empleado

Esta tabla debe contener la información personal de los empleados que laboran dentro del local, es aquí donde se guarda el tipo de usuario y la contraseña con la que debe acceder al sistema por lo que solo el usuario con perfil de administrador es el único que puede realizar estas funciones.

Nombre	Tipo de datos	Not null	Descripción
No_Empleado	VARCHAR (10) PK	x	Numero de trabajador asignado por la empresa.
Nombre	VARCHAR (50)	x	Nombre del empleado.
Ap_P	VARCHAR (50)	x	Apellido paterno del empleado.
Ap_M	VARCHAR (50)	x	Apellido materno del empleado.
Contraseña	VARCHAR (50)	x	Contraseña generada para el acceso al sistema del sistema.
Tipo	VARCHAR (20)	x	Tipo de usuario asignado al empleado para su perfil dentro del sistema, pueden ser Administrador, supervisor, operador y cocinero.
Id_Depto	VARCHAR (10) FK	x	Departamento al que pertenece el empleado.
Local	VARCHAR (20) FK	x	Local al que pertenece el empleado.

Tabla 4.8.- Tabla Empleado detallada.

Relaciones de la tabla empleado				
Nombre	**Padre**	**Hijo**	**Columna**	**Tipo**
FK_Empleado_Local	Local	Empleado	Id_Local	1:*
FK_Empleado_Departamento	Departamento	Empleado	Id_Depto	1:*

Tabla 4.9.- Relaciones de la tabla empleado.

Bebida

En esta tabla se debe contener toda la información correspondiente a las bebidas elaboradas dentro del establecimiento, dicha información es muy importante por lo que el administrador es el único con los privilegios necesarios para la manipulación de la información contenida en la tabla.

Nombre	Tipo de datos	Not null	Descripción
Id_Bebida	VARCHAR (10) PK	X	Número de identificación para la bebida preparada por el establecimiento.
Nombre_B	VARCHAR (50)	X	Nombre de la bebida.
Descripción	VARCHAR (200)	X	Descripción del contenido o preparación de la bebida.
Precio	DOUBLE	X	Precio de la bebida.
Imagen	MEDIUMBLOB	X	Imagen o descripción grafica relacionada con la bebida.

Tabla 4.10.- Tabla Bebida detallada.

Platillos

En esta tabla se debe contener toda la información correspondiente a los platillos elaborados dentro del establecimiento, dicha información es muy importante por lo que el administrador es el único con los privilegios necesarios para la manipulación de la información contenida en la tabla.

Nombre	Tipo de datos	Not null	Descripción
Id_Platillos	VARCHAR (10) PK	X	Número de identificación para el platillo elaborado por el establecimiento.
Nombre_PL	VARCHAR (50)	X	Nombre del platillo.
Descripción	VARCHAR (200)	X	Descripción del contenido o preparación del platillo.
Precio	DOUBLE	X	Precio del platillo.
Imagen	MEDIUMBLOB	X	Imagen o descripción grafica relacionada con el platillo.

Tabla 4.11.- Tabla Platillo detallada.

Orden

Esta es la principal tabla de la base de datos, ya que debe contener la información de las órdenes generadas por los usuarios del sistema y con la cual el sistema realiza las principales funciones en la operación d punto de venta.

Nombre	Tipo de datos	Not null	Descripción
Id_Orden	INT(11) PK	x	Número de identificación de la orden.
Fecha	DATE	X	Contiene la fecha de registro de la orden con formato yyyy-mm-dd.
Mesero	VARCHAR (10) FK	x	Contiene el Id del mesero que genera la orden.
Estado	VARCHAR (15)	x	Estado en el que se encuentra la orden (Abierta, Cancelada, Lista).
Bebida	VARCHAR (10) FK	x	Contiene el Id correspondiente a la bebida seleccionada.
Platillo	VARCHAR (10) FK	x	Contiene el Id correspondiente al platillo seleccionado.
Cantidad	INT(3)	x	Cantidad de los platillos o bebidas seleccionados.
Total	INT(11)	x	Precio total de la orden generada

Tabla 4.12.- Tabla Orden detallada.

Relaciones de la tabla orden				
Nombre	**Padre**	**Hijo**	**Columna**	**Tipo**
FK_orden_empleado	empleado	orden	No_Empleado	1:*
FK_orden_bebida	bebida	orden	Id_Bebida	1:*
FK_orden_platillos	platillos	orden	Id_Platillo	1:*

Tabla 4.13.- Relaciones de la tabla orden.

Capítulo 5.- Implementación.

Tomando en cuenta el análisis realizado en los capítulos anteriores de este documento, se procederá a realizar la construcción del sistema, mostrando las principales funciones de los usuarios que deben contar con acceso al sistema atreves de una pc, como lo son el *Administrador*, así como el personal de cocina que se encuentra encargado durante el turno.

Por lo que el inicio del presente capitulo es la demostración de la construcción del acceso al sistema.

5.1.- Conexión a la base de datos.

Para el uso correcto del sistema se debe asegurar que el usuario que accede sea el indicado, ya que se hace uso de la base de datos que contiene información importante para la operación del establecimiento. Por lo que se inicia con la creación de una clase la que se nombrara *CONECTAR*, la cual será la clase conexión.

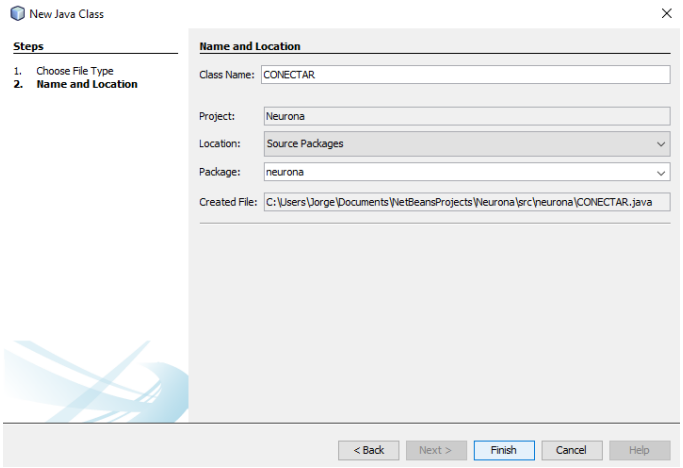

Figura 5.1.- Creación de la clase conexión.

Una vez que se crea la clase que servirá para la conexión a la base de datos, se agrega a las librerías el controlador para la conexión.

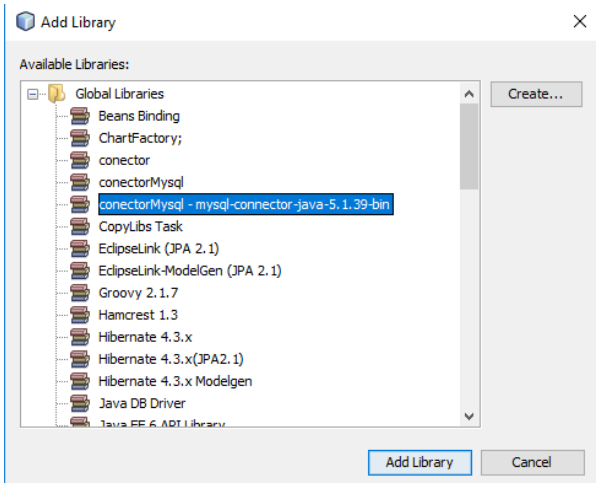

Figura 5.2.- Adición a las librerías del conector a la base de datos.

Una vez agregado a las librerías, por lo que el siguiente paso a seguir es la generación del código necesario realizar la conexión, el cual presenta la siguiente sintaxis:

```
Class.forName("com.mysql.jdbc.Driver");
conectar=DriverManager.getConnection("jdbc:mysql://localhost/tesis?user=jorge&password=1234");
```

Donde se establece la conexión con la base de datos de nombre “tesis” y se establecen las credenciales necesarias para el acceso como lo son el nombre de usuario y contraseña del administrador.

5.2.- Acceso.

Por seguridad el acceso al sistema debe ser controlado, de tal forma que solo los usuarios autorizados y que estén registrados en la base de datos ingresen, por lo que a continuación de enlistan los elementos que componen la ventana que se utiliza para el ingreso:

Elementos de la interfaz gráfica para el acceso		
Elemento	**Nombre de la variable**	**Texto**
jLabel	jLabel1	Usuario
jLabel	jLabel1	Contraseña
JTextField	TXT_USUARIO	
JTextField	TXT_CONTRA	
JButton	Boton_Entrar	Entrar
JButton	Boton_Salir	Salir

Tabla 5.1.- Elementos que componen la interfaz gráfica del acceso.

Con los elementos de listados en la tabla 5.1 se construye la interfaz gráfica representada en la figura 5.3.

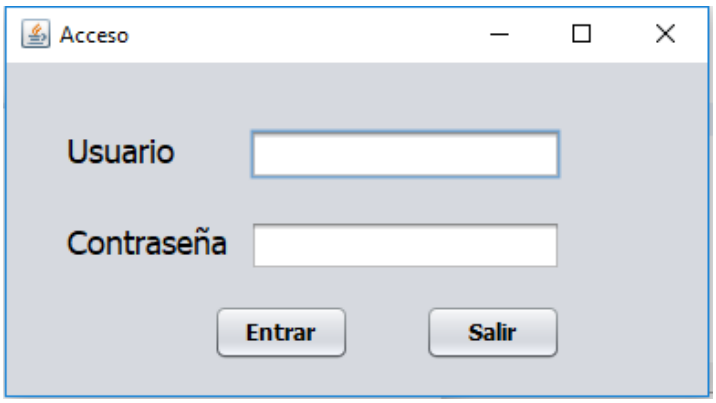

Figura 5.3.- ventana para el acceso al sistema.

Una vez creada la interfaz gráfica para el acceso se procede con la codificación de las funciones, comenzando con la creación de un método que permita identificar los usuarios con autorización de acceso a la base de datos.

```
String Captura="";
    String sql="SELECT * FROM empleado WHERE No_Empleado ='"+usuario+"'&& Contraseña= '"+contraseña+"'";

    try {

        Statement st= cn.createStatement();
        ResultSet rs= st.executeQuery(sql);

        while (rs.next())
        {
            Captura=rs.getString("Tipo");
        }
```

Con dicha sintaxis se crea una variable de tipo String nombrada *Captura*, seguida de la cadena que realiza la consulta almacenada en *sql*, en la que se selecciona todo de la tabla *empleado* donde *No_Empleado* y *Contraseña* sean iguales a los valores almacenados en las variables cuyo valor se obtendrá más delante de los campos de texto.

Seguido de la conexión a la base de datos con la ejecución de la sentencia contenida en *sql*, donde mientras se obtenga un resultado positivo en la consulta se guarda en la variable de tipo String *Captura* el campo *Tipo* contenido en la fila seleccionada por la consulta.

En el siguiente código si la consulta realizada resulto exitosa se procede a realizar la comparación con la base de datos, por lo que si el contenido de Captura es igual a “Administrador” que corresponde al campo de *Tipo* en la tabla *Empleado* se accede al formulario que contendrá las funciones principales del sistema.

```
if (Captura.equals("Administrador"))
{
    this.setVisible(false);
    JOptionPane.showMessageDialog(null,"Bienvenid@");
    Admin ABRIR =new Admin();
    ABRIR.setVisible(true);
}
```

El mismo procedimiento se aplica en los demás usuarios, en el caso de que el usuario no cuente con acceso al sistema se le debe negar el acceso, por lo que se realiza la siguiente sintaxis:

```
if ((!Captura.equals("Cocina")) && (!Captura.equals("Administrador")))
{
   JOptionPane.showMessageDialog(null,"Datos Incorrectos");
}
```

En la que si el contenido de Captura es diferente a los especificados, que en este caso es “Administración” y “Cocina” se despliega un mensaje en el que se informa que los datos ingresados son incorrectos.

Habiendo realizado la construcción del método se procede a crear un evento en el que al presionar el botón *ingresar* (Boton_Entrar) en el que se obtiene los contenidos de los JTextBox para después de usarlos como valores iniciales en el método anteriormente contenido.

```
String usu=TXT_USUARIO.getText();
String ctr=TXT_CONTRA.getText();
INGRESO(usu,ctr);
```

5.3.- Ventana Principal.

Una vez el usuario es identificado se abre la ventana principal, la cual cuenta con el acceso a las funciones principales asignadas al usuario, para esta sección se documentara la ventana principal del administrador, la cual está construida con los siguientes elementos enlistados en la tabla 5.2.

Elementos del formuario Principal		
Elemento	**Nombre**	**Texto**
JDesktopPane	Escritorio	
JMenuBar	BARRA_M	
JMenu	M_ARCHIVO	Archivo
JMenu	M_ADMIN	Administración
JMenu	M_OP	Operación
JMenu	M_MENUS	Menús
JMenuItem	MENU_SALIR_	Salir
JMenuItem	MENU_LOCAL_	Local
JMenuItem	MENU_DEPTO_	Departamento
JMenuItem	MENU_USER_	Usuarios
JMenuItem	MENU_PLATILLOS_	Platillos
JMenuItem	MENU_BEBIDA_	Bebidas
JMenuItem	MENU_SUP_	Supervisión

Tabla 5.2.- Elementos de la ventana principal.

Una vez agregados al JFrame se genera la ventana principal la cual actuara de escritorio para el sistema, ya que todas las aplicaciones se ejecutaran internamente, mediante los

JInternalFrame, cuyos accesos se encuentran en los elemento de menú (JMenuItem) dando como resultado una interfaz gráfica mostrada en las figura 5.4.

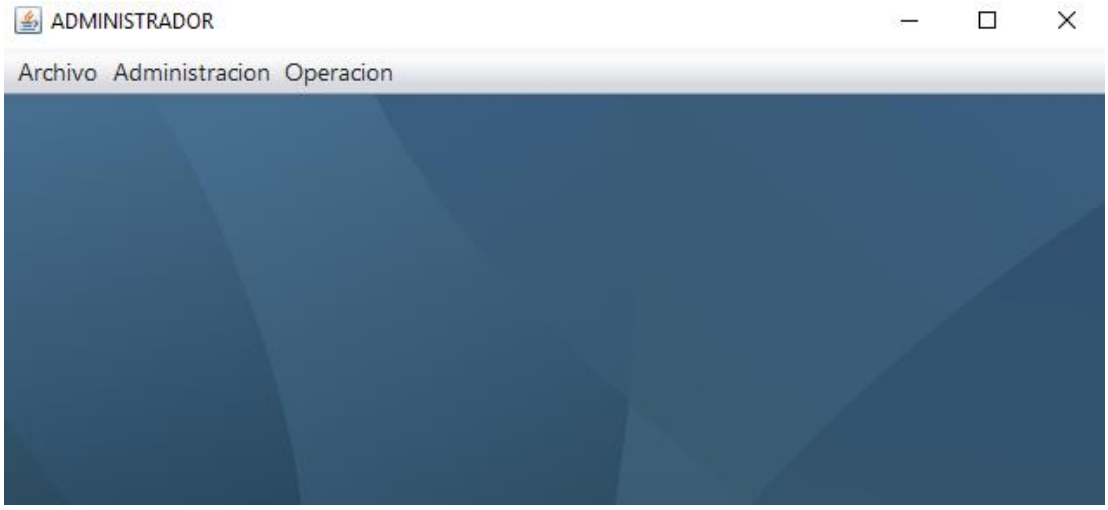

Figura 5.4.- Escritorio del sistema de gestión.

En dicha interfaz gráfica se agregan los menús para el acceso a las funciones del sistema, por lo que los menús una vez agregados se muestran en las figuras 5.5, 5.6 y 5.7.

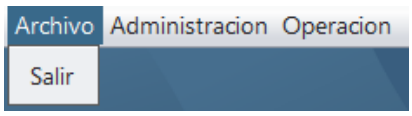

Figura 5.5.- Menú archivo.

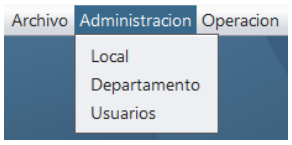

Figura 5.6.- Menú Administración.

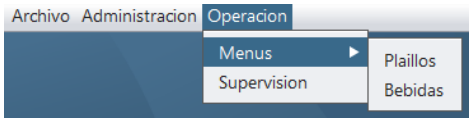

Figura 5.7.- Menú de Operación.

5.4.- Gestión de locales.

En caso de que el o los clientes que adquieran el sistema cuenten con más de un establecimiento se generó un formulario para su gestión, en el que se puede manipular con facilidad la información relacionada con ellos, siendo la interfaz gráfica conformada con los elementos listados en la tabla 5.3.

Elementos del Formulario Local		
Elemento	**Nombre**	**Texto**
JLabel	JLabel1	Id
JLabel	JLabel2	Sucursal
JLabel	JLabel3	Dirección
JLabel	JLabel4	Teléfono
JTextField	TXT_ID	
JTextField	TXT_SUC	
JTextField	TXT_DIR	
JTextField	TXT_TEL	
JPanel	JPanel1	
JButtton	B_AGREGAR_	Registrar
JButtton	B_MODIFICAR_	Modificar
JButtton	B_ELIMINAR_	Eliminar
JButtton	B_SALIR_	Salir
JTable	TABLA_LOC	

Tabla 5.3.- Elementos del formulario Local.

Con los elementos enlistados se crea la interfaz gráfica mostrada en la fig. 5.8 la cual ya cuenta con datos que al comparar con la fig. 5.9 se puede demostrar la conexión a la base de datos, con lo que se procede a realizar el análisis de la construcción del formulario, el cual está siendo ejecutado en un JInternalFrame.

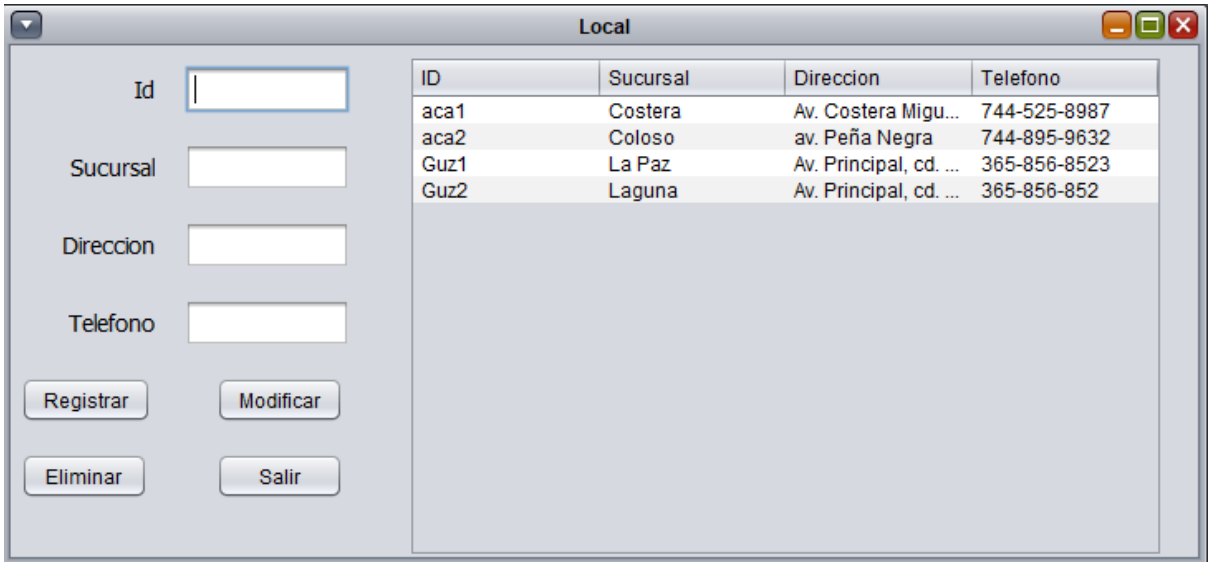

Figura 5.8.- Interfaz Gráfica de la ventana local.

Id_Local	Nombre_loc	Direccion	Telefono
aca1	Costera	Av. Costera Miguel Aleman, S/N	744-525-8987
aca2	Coloso	av. Peña Negra	744-895-9632
Guz1	La Paz	Av. Principal, cd. Guzman	365-856-8523
Guz2	Laguna	Av. Principal, cd. Guzman	365-856-852

Figura 5.9.- Tabla Local de la base de datos.

5.4.1.- Ingresar datos.

Se comienza con la construcción del JTable que se utiliza para visualizar los contenidos de la tabla con la que se realizaran las operaciones, por lo que como se puede observar en la sintaxis, se agregan las columnas y se asignan los nombres de las mismas.

```
DefaultTableModel modelo=new DefaultTableModel();
modelo.addColumn("ID");
modelo.addColumn("Sucursal");
modelo.addColumn("Direccion");
modelo.addColumn("Telefono");

TABLA_LOC.setModel(modelo);
String [] datos = new String [20];
```

Ya armado el cuerpo del JTable se continúa con la conexión a la base de datos la cual se realiza mediante el uso de la clase conexión CONECTAR que se creó con anterioridad.

```
CONECTAR cc=new CONECTAR();
Connection cn= cc.Conexion();
```

Una vez establecida la conexión a la base de datos, se agregan las líneas de código correspondientes a la consulta a la basa de datos, seleccionando todo de la tabla *local*, por lo que mientras la consulta sea positiva se agregan los contenidos de la tabla consultada en las columnas de la *JTable*.

```
try {
    Statement st =cn.createStatement();
    ResultSet rs= st.executeQuery("SELECT * FROM local");

    while(rs.next()){
        datos[0]=rs.getString(1);
        datos[1]=rs.getString(2);
        datos[2]=rs.getString(3);
        datos[3]=rs.getString(4);
        modelo.addRow(datos);
    }
    TABLA_LOC.setModel(modelo);
```

Ahora que se obtiene la información de la tabla el próximo paso es guardar información nueva en dicha tabla por lo que se crea un evento en el JButtton *Registrar*, el cual contendrá la siguiente sintaxis:

```
PreparedStatement pst= cn.prepareStatement

("INSERT INTO local(Id_Local,Nombre_loc,Direccion,Telefono) VALUES (?,?,?,?)");

pst.setString(1,TXT_ID.getText());
pst.setString(2,TXT_SUC.getText());
pst.setString(3,TXT_DIR.getText());
pst.setString(4,TXT_TEL.getText());
pst.executeUpdate();

TABLA();
```

Con dicha sintaxis se realiza la consulta a la base de datos en la que se inserta en la tabla *local* los datos que se encuentran dentro de los campos de texto, realizando una actualización al ejecutar la acción y posteriormente se carga el contenido en la *JTable*.

Como se demuestra en las figura 5.10 al ingresar los datos en el formulario y presionar el botón Registrar (B_AGREGAR_) estos se guardan en la base de datos y automáticamente se actualizan en la JTable.

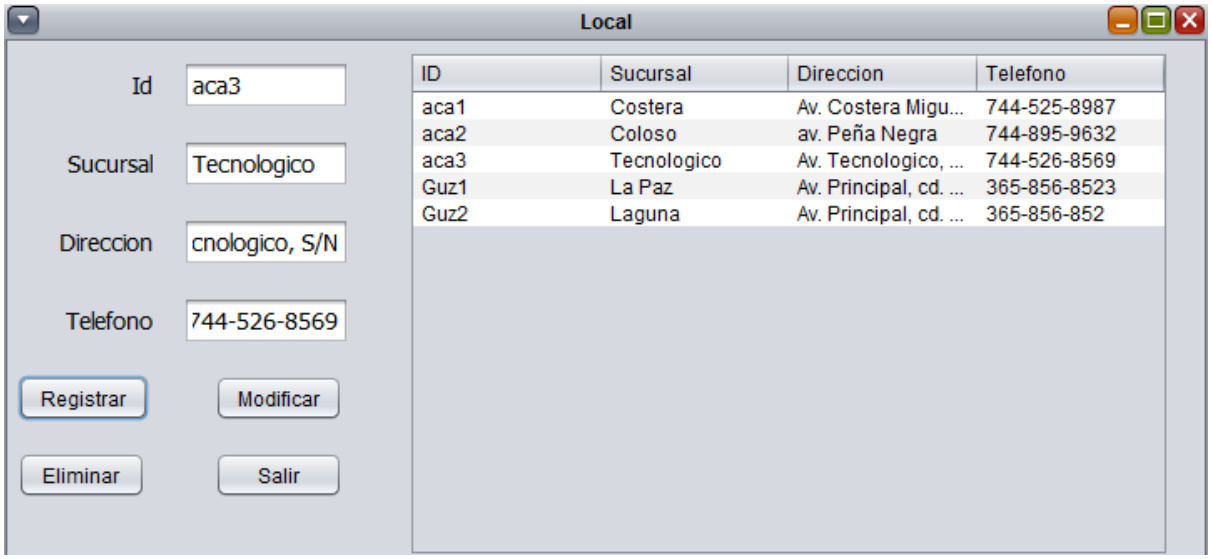

Figura 5.10.- Guardar una nueva fila en la tabla local desde el sistema.

Como se puede apreciar en la fig. 5.11, la nueva fila es ingresada exitosamente en la tabla local.

Id_Local	Nombre_loc	Direccion	Telefono
aca1	Costera	Av. Costera Miguel Aleman, S/N	744-525-8987
aca2	Coloso	av. Peña Negra	744-895-9632
aca3	Tecnologico	Av. Tecnologico, S/N	744-526-8569
Guz1	La Paz	Av. Principal, cd. Guzman	365-856-8523
Guz2	Laguna	Av. Principal, cd. Guzman	365-856-852

Figura 5.11.- Tabla local con nueva fila insertada.

5.4.2.- Modificar Datos.

Para realizar la modificación de los datos contenidos en la tabla, primero se deben enviar los datos de la fila seleccionada para su modificación, a los respectivos campos de texto, por lo que se crea un nuevo evento en el que cada vez que una fila es seleccionada su contenido es enviado al *JTextField* que lo genero.

```
 int fila= TABLA_LOC.getSelectedRow();
if(fila>=0){

TXT_ID.setText(TABLA_LOC.getValueAt(fila,0).toString());
TXT_SUC.setText(TABLA_LOC.getValueAt(fila,1).toString());
TXT_DIR.setText(TABLA_LOC.getValueAt(fila,2).toString());
TXT_TEL.setText(TABLA_LOC.getValueAt(fila,3).toString());
}
else{
    JOptionPane.showMessageDialog(null, "no seleccionaste fila");

}
```

Como se muestra en la sintaxis se crea una variable tipo Int nombrada *fila*, en la que se guardara la fila seleccionada, si el contenido de *fila* es mayor o igual al cero toma los valores de las columnas y los envía nuevamente a sus respectivo campo de texto, en caso contrario despliega un mensaje en el que se notifica que no se ha seleccionado una fila correcta, lo que facilita la próxima tarea, la cual es la modificación de los datos seleccionados, como lo demuestra la fig. 5.12.

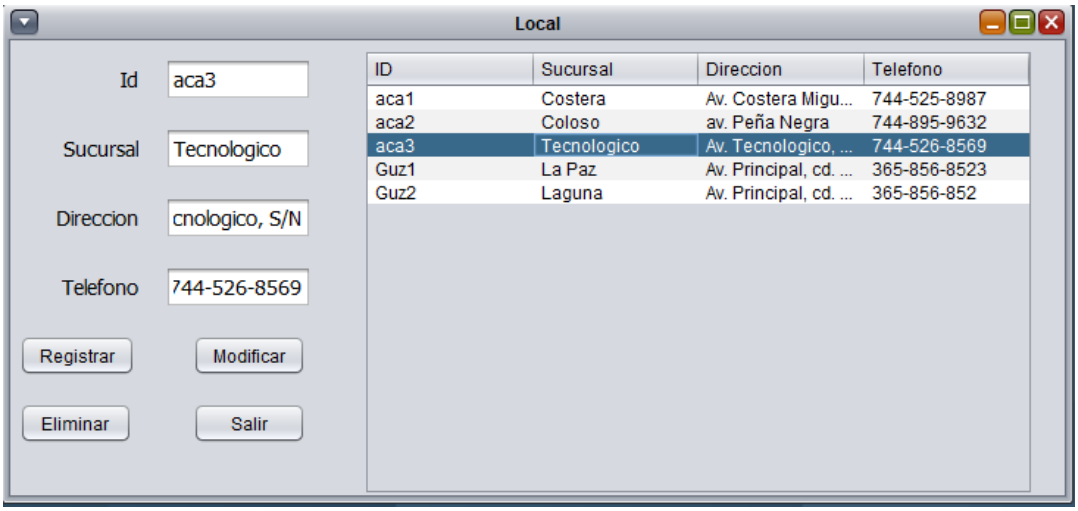

Figura 5.12.- Selección de fila en el formulario.

Una vez seleccionada la fila y obtenidos los datos a modificar se genera un evento más para el botón Modificar (B_MODIFICAR_) en el que se actualiza la tabla *local* en las columnas que

se especifican en la consulta, las cuales obtienen su nuevo contenido del texto contenido dentro de los campos de texto que se les están asignando.

```
PreparedStatement pst = cn.prepareStatement
("UPDATE local SET Nombre_loc ='"+TXT_SUC.getText()+"',Direccion='"
+TXT_DIR.getText()+"',Telefono='"+TXT_TEL.getText()+"' WHERE Id_Local='"
+TXT_ID.getText()+"'" );
pst.executeUpdate();
TABLA();
```

Una vez realizada la modificación de los datos, se procede a realizar la visualización de los cambios en la JTable, tal como se muestra en la fig. 5.13

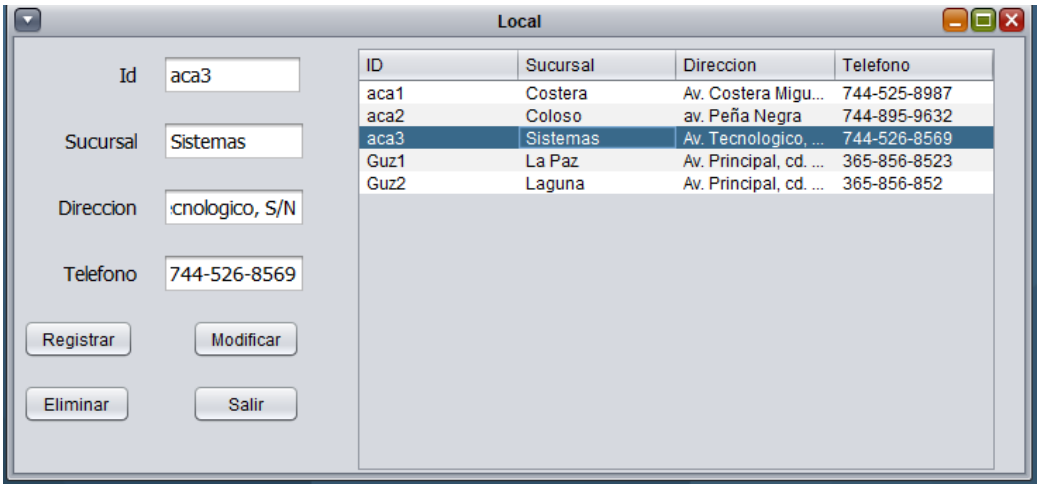

Figura 5.13.- Modificación de la columna Sucursal en la base de datos.

Lo anterior queda demostrado al revisar el contenido de la tabla local en la base de datos, tal y como se muestra en la fig. 5.14

Id_Local	Nombre_loc	Direccion	Telefono
aca1	Costera	Av. Costera Miguel Aleman, S/N	744-525-8987
aca2	Coloso	av. Peña Negra	744-895-9632
aca3	Sistemas	Av. Tecnologico, S/N	744-526-8569
Guz1	La Paz	Av. Principal, cd. Guzman	365-856-8523
Guz2	Laguna	Av. Principal, cd. Guzman	365-856-852

Figura 5.14.- Tabla local modificada.

5.4.3.- Eliminar datos.

Para la eliminación de una fila en la base de datos se crea un evento para el botón Eliminar (B_ELIMINAR_) en el que se escribe el siguiente código:

```
 int fila= TABLA_LOC.getSelectedRow();
 String Cod="";
Cod=TABLA_LOC.getValueAt (fila, 0).toString();
```

En el que la fila seleccionada se guardara en la variable de tipo Int nombrada *fila*, que a su vez es en otra variable del tipo String nombrada *Cod*, en la que sirve para especificar la fila y la columna en la que se realizara la consulta a la base de datos.

```
PreparedStatement pst = cn.prepareStatement
("DELETE FROM local WHERE Id_Local='"+Cod+"'");
pst.executeUpdate();
TABLA();
```

En el código la consulta elimina de la tabla local donde el Id_Local sea igual a la contenida en la variable *Cod*, que donde se guarda la Id de la fila que se desea eliminar.

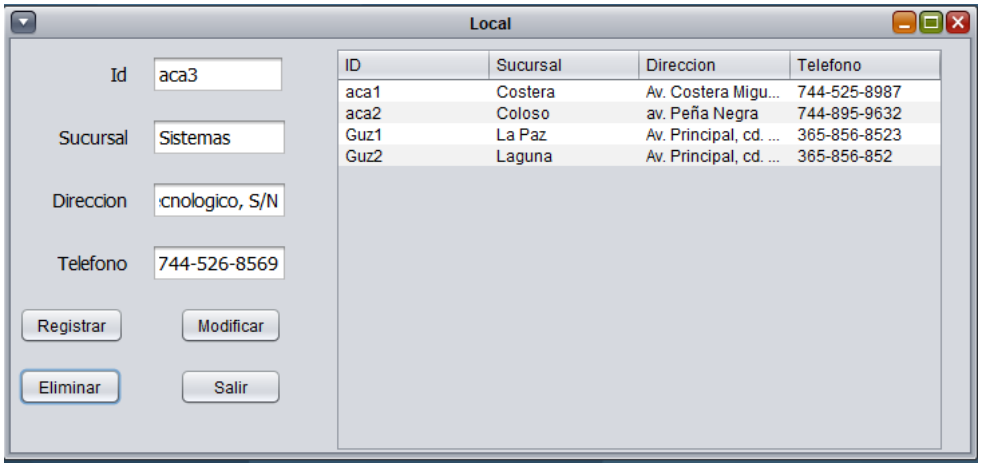

Figura 5.15.- Fila eliminada desde el formulario.

Como se muestra en la fig. 5.15, la fila seleccionada ha sido eliminada, aunque como medida de seguridad ante errores por parte del usuario, en el que haya eliminado una fila que no deba, los datos de la fila eliminada se mantienen en los JTextField correspondientes para que puedan nuevamente ser ingresados.

Y en caso de que el establecimiento a borrar cuente con algún empleado asignado a él, se desplegara un mensaje con el que se notificara que no puede ser borrado porque aun cuenta con empleados (fig. 5.16).

Figura 5.16.- Mensaje de sistema.

Como se puede ver en la base de datos, la primer fila en ser seleccionada es eliminada de la tabla local, más sin embargo, la segunda fila en ser seleccionada no fue eliminada debido a que aún cuenta con empleados asignados a él.

Id_Local	Nombre_loc	Direccion	Telefono
aca1	Costera	Av. Costera Miguel Aleman, S/N	744-525-8987
aca2	Coloso	av. Peña Negra	744-895-9632
Guz1	La Paz	Av. Principal, cd. Guzman	365-856-8523
Guz2	Laguna	Av. Principal, cd. Guzman	365-856-852

Figura 5.17.- Tabla local después de eliminar una fila.

5.5.- Gestión de Departamento.

El sistema también requiere el manejo de los departamentos a los que pertenecen los empleados ya que al separarlos por departamentos permite repartir las funciones que realizara cada empleado en su puesto de trabajo dentro del establecimiento, por lo que para la construcción del formulario que se utilizara se requieren los siguientes elementos listados en la tabla 5.4.

Elementos del formulario Departamento		
Elemento	**Nombre**	**Texto**
JPanel	Jpanel1	
JTextField	TXT_ID	
JTextField	TXT_NOMBRE	
JTable	TABLA_DEPTO	
JLabel	JLabel1	ID
JLabel	JLabel2	Nombre
JButton	B_INGRESAR_	Ingresar
JButton	B_MODIF_	Modificar
JButton	B_ELIMINAR_	Eliminar
JButton	B_SALIR_	Salir

Tabla 5.4.- Elementos del formulario Departamento.

Una vez implementados los elementos se construye el formulario mostrado en la fig. 5.18

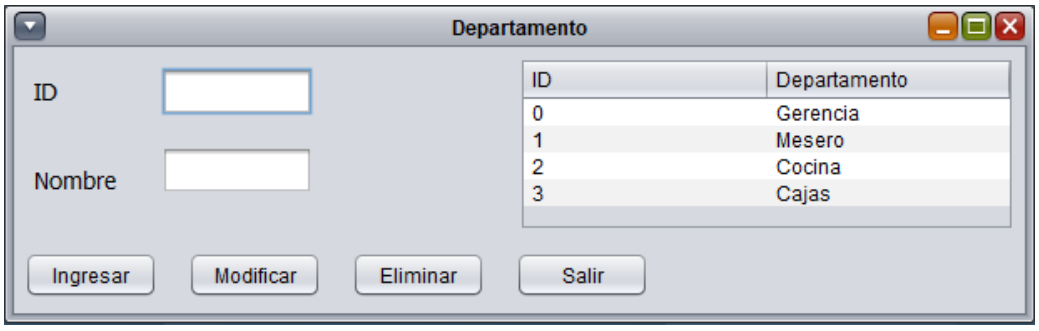

Figura 5.18.- Formulario de gestión de Departamentos.

Como se muestra en la fig. 5.18 se implementa el siguiente código el cual permite visualizar todo el contenido de la tabla *departamento*, en el código se definen las columnas de que se visualizaran en el JTable.

```
DefaultTableModel modelo=new DefaultTableModel();
modelo.addColumn("ID");
modelo.addColumn("Departamento");
```

El método que se utiliza para la conexión a la base de datos utiliza la siguiente sintaxis, como la que se realiza la consulta, en la que se selecciona todo el contenido de la tabla *departamento*.

```
CONECTAR cc=new CONECTAR();
Connection cn= cc.Conexion();

Statement st =cn.createStatement();
ResultSet rs= st.executeQuery("SELECT * FROM departamento");
```

Id_Depto	Nombre_depto
0	Gerencia
1	Mesero
2	Cocina
3	Cajas

Figura 5.19.- Tabla Departamento.

5.5.1.- Ingresar Datos.

Para el ingreso de un nuevo departamento a la base datos, en el caso de así requerirlo, el formulario permite realizarlo, por lo que se crea un evento en el botón Ingresar (B_INGRESAR_), por lo que dentro de dicho evento se ingresa el condigo.

```
PreparedStatement pst= cn.prepareStatement
("INSERT INTO departamento (Id_Depto,Nombre_depto) VALUES (?,?)");
```

```
pst.setString(1,TXT_ID.getText());
pst.setString(2,TXT_NOMBRE.getText());
pst.executeUpdate();
TABLA();
```

Con el que se inserta en la tabla departamento el contenido de los J*TextBox* en las columnas *Id_Depto* e *Nombre_depto*, una vez ejecutado el evento como se puede observar en la fig. 5.20, la fila nueva es ingresada a la base de datos (fig. 5.21) y reflejada en la JTable (TABLA_DEPTO).

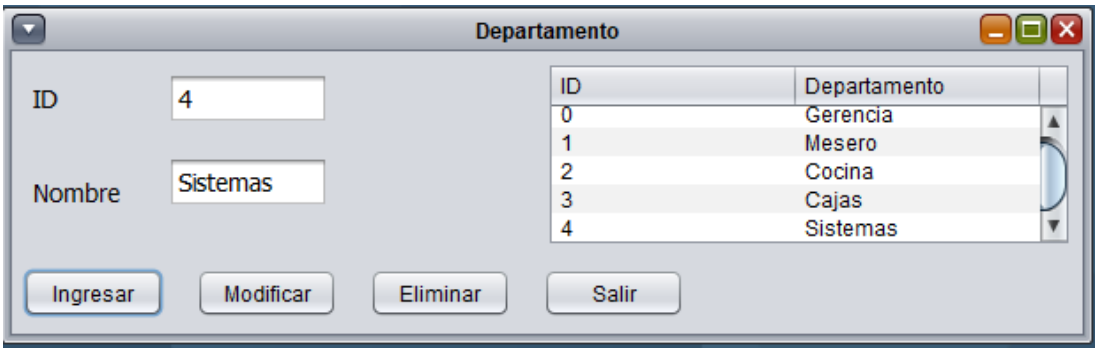

Figura 5.20.- Fila agregada en formulario Departamento.

Id_Depto	Nombre_depto
0	Gerencia
1	Mesero
2	Cocina
3	Cajas
4	Sistemas

Figura 5.21.- Fila nueva en tabla departamento.

Una vez logrado el ingreso de datos a la base atreves del formulario Departamento el siguiente paso a seguir es el modificar la información contenida en la tabla Departamento.

5.5.2.- Modificar Datos.

El sistema en caso de requerirlo puede modificar la información de un departamento, por lo que para dicho proceso se deben obtener los datos de la fila que se requiera modificar, para lograr eso se requiere la creación de un evento en el que se inserte el siguiente código:

```
 int fila= TABLA_DEPTO.getSelectedRow();
if(fila>=0){

TXT_ID.setText(TABLA_DEPTO.getValueAt(fila,0).toString());
TXT_NOMBRE.setText(TABLA_DEPTO.getValueAt(fila,1).toString());
```

En dicho evento, los valores de la fila seleccionada de la JTable (TABLA_DEPTO) en el formulario de Departamento, son enviados a sus respectivos campos de texto (JTextField) como se puede ver en la fig. 5.22.

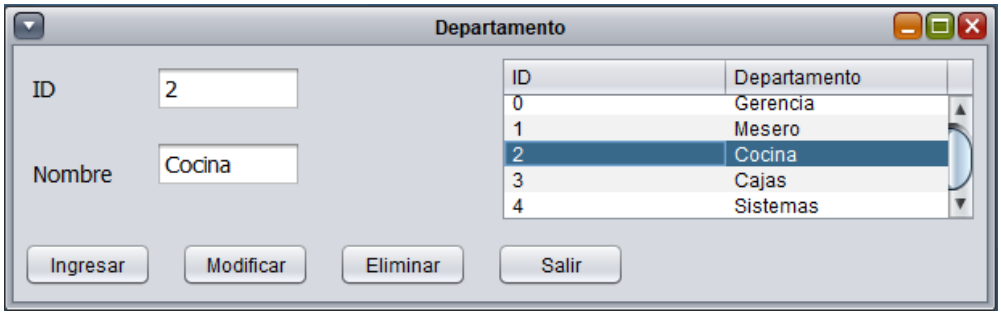

Figura 5.22.- Obtención de datos de la JTable del formulario Departamento.

Una vez se han obtenido los datos de la fila que se desea modificar se procede a crear un nuevo evento dentro del botón Modificar (B_MODIF_).

```
PreparedStatement pst = cn.prepareStatement
("UPDATE departamento SET Nombre_depto ='"+TXT_NOMBRE.getText()+"'
WHERE Id_Depto='"+TXT_ID.getText()+"'" );
```

En dicho evento se actualiza el contenido de la tabla departamento con el contenido de los JTextField que se asigna a cada columna de la tabla, donde el id sea el contenido en el JTextField de *ID* (TXT_ID). Como se observa en la fig. 5.23 la fila 4 es modificada pasando a ser el departamento con *ID* 4 de Sistemas a Tecnológico.

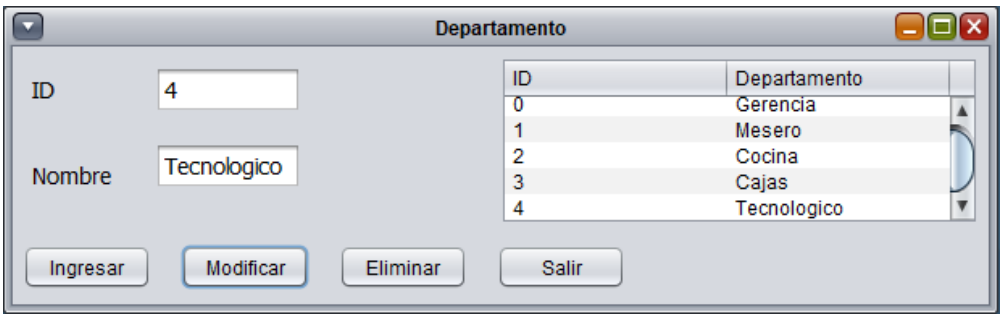

Figura 5.23.-Fila 4 modificada en formulario departamento.

El cambio de departamento es confirmado al visualizar el contenido de la tabla departamento en la base de datos (fig. 5.24), con lo que queda demostrado el funcionamiento del método.

Id_Depto	Nombre_depto
0	Gerencia
1	Mesero
2	Cocina
3	Cajas
4	Tecnologico

Figura 5.24.- Contenido de la tabla departamento en la base de datos.

5.5.3.- Eliminar Datos.

En el caso de que el cliente requiera el eliminar un departamento del establecimiento el sistema lo hace posible creando un método en el botón *Eliminar* (B_ELIMINAR_).

```
int fila= TABLA_DEPTO.getSelectedRow();
String Cod="";
Cod=TABLA_DEPTO.getValueAt (fila, 0).toString();
```

Se introduce el código con el que en una variable del tipo int nombrada *fila* en la que se guarda la tabla seleccionada de la JTable (TABLA_DEPTO) del formulario, para después en una variable String nombrada *Cod* guardar la primera columna de la fila seleccionada.

```
PreparedStatement pst = cn.prepareStatement
("DELETE FROM departamento WHERE Id_Depto='"+Cod+"'");
pst.executeUpdate();
TABLA();
```

En las siguientes líneas de código se realiza la consulta a la base de datos en la que se elimina de la tabla departamento donde Id_Depto sea igual al contenido de la variable Cod y después visualizar su contenido en la JTable, tal como se muestra en la fig. 5.25.

Como medida de seguridad ante errores por parte de los usuarios del sistema el sistema mantiene en los JTextBox la información de la fila recientemente eliminada para poder volver a ser ingresadas ya sea con los mismos datos o con algunas modificaciones, en caso de que el departamento aun cuente con empleados asignados, el sistema despliega un mensaje en el que se indica esa situación.

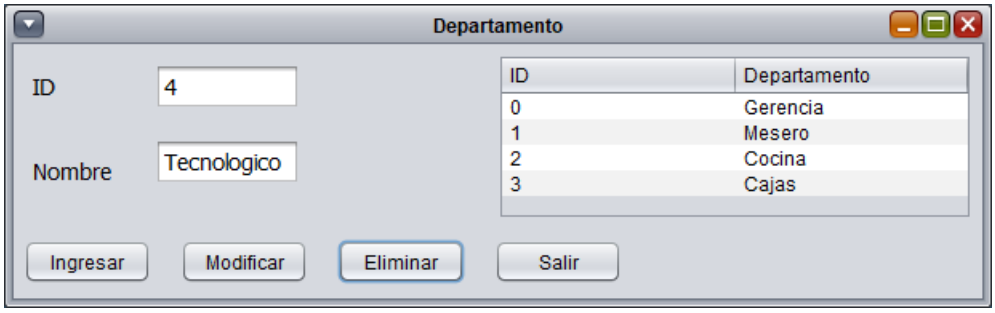

Figura 5.25.- Fila 4 eliminada del formulario Departamento.

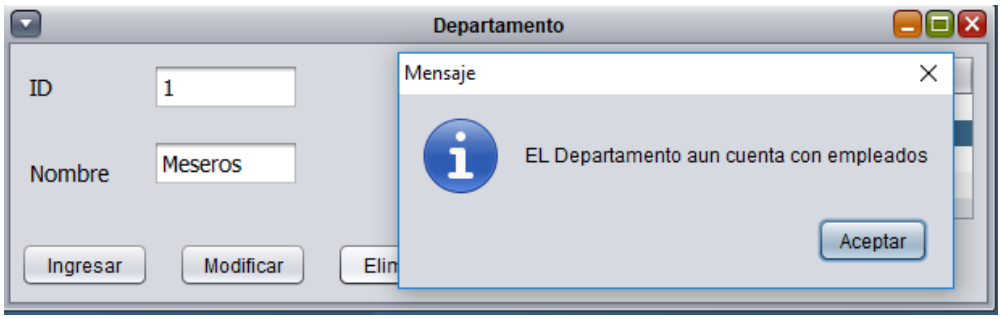

Figura 5.26.- Mensaje de error del formulario.

Como se puede observar en la fig. 5.27 que muestra el contenido de la tabla departamento la fila 4 fue eliminada de la base de datos.

Id_Depto	Nombre_depto
0	Gerencia
1	Mesero
2	Cocina
3	Cajas

Figura 5.27.- Fila eliminada de la tabla departamento.

5.6.- Gestión de Empleados.

Cualquier establecimiento tiene la necesidad de contar con empleados que lleven a cabo las funciones necesarias, ya que las operaciones antes mencionadas del sistema permiten la distribución de los empleados para el funcionamiento de establecimiento.

Elementos del Formulario Usuarios		
Elemento	**Nombre**	**Texto**
JTable	TABLA_USUARUIOS_	
JLabel	JLabel1	No. de Trabajador
JLabel	JLabel6	Nombre
JLabel	JLabel7	Apellido Paterno
JLabel	JLabel8	Apellido Materno
JLabel	JLabel9	Contraseña
JLabel	JLabel10	Tipo
JLabel	JLabel11	Departamento
JLabel	JLabel12	Local
JTextField	TXT_ID	
JTextField	TXT_NOMBRE_	
JTextField	TXT_AP_	
JTextField	TXT_PM_	
JTextField	TXT_CONTRA_	
JComboBox	C_TIPO	
JComboBox	CB_DEPTO	
JComboBox	CB_LOCAL	
JButton	B_INGRESAR_	Registrar
JButton	B_MODIF_	Modificar
JButton	B_ELIMINAR_	Eliminar
JButton	B_SALIR	Salir

Tabla 5.5.- Elementos del Formulario Empleados.

Por lo que, como en los casos anteriores se inicia con la creación de una JTable en el formulario que permita visualizar el contenido de la tabla *empleado*, iniciando con la creación de la conexión a la base de datos.

```
CONECTAR cc=new CONECTAR();
Connection cn= cc.Conexion();

DefaultTableModel modelo=new DefaultTableModel();
modelo.addColumn("No. de Empleado");
modelo.addColumn("Nombre");
modelo.addColumn("Apellido Paterno");
modelo.addColumn("Apellido Materno");
modelo.addColumn("Contraseña");
modelo.addColumn("Tipo de Usuario");
modelo.addColumn("Departamento");
modelo.addColumn("Local");
```

Seguido de la creación de las columnas que conforman la tabla, y posterior mente la consulta a la base de datos en la que se selecciona todo el contenido de la tabla empleado (fig. 5.28), para facilidad del usuario se crea una vista de la tabla empleados, en la que se visualizan las descripciones de los departamentos y los locales en lugar de únicamente los Id´s.

```
Statement st =cn.createStatement();
ResultSet rs= st.executeQuery("SELECT * FROM v_empleados");
```

No_Empleado	Nombre	Ap_P	Ap_M	Contraseña	Tipo	Id_Depto	Local
0001	jorge	jimenez	castañon	contraseña1	Administrador	0	aca1
0002	Claudia Josefina	Villalvazo	Martinez	contraseña2	Administrador	0	Guz1
0003	Duke	El Perro	Feliz	contraseña3	Operador	1	aca2
0004	Lucas	El Gato	Rebelde	contraseña4	Cocina	1	Guz1
0005	Osiris	La Gata	Coqueta	contraseña5	Supervisor	3	Guz2

Figura 5.28.- Tabla empleado.

No_Empleado	Nombre	Ap_P	Ap_M	Contraseña	Tipo	Nombre_depto	Nombre_loc
0001	jorge	jimenez	castañon	contraseña1	Administrador	Gerencia	Costera
0002	Claudia Josefina	Villalvazo	Martinez	contraseña2	Administrador	Gerencia	La Paz
0003	Duke	El perro	Feliz	contraseña3	Operador	Meseros	Coloso
0004	Lucas	El Gato	Rebelde	contraseña4	Cocina	Meseros	La Paz
0005	Osiris	La Gata	Coqueta	contraseña5	Supervisor	Cajas	Laguna

Figura 5.29.- Vista v_empleados.

Utilizando los elementos enlistados en la tabla 5.5 se construye el formulario mostrado en la figura 5.30 en el que se puede observar la JTable creada ya enlazada a la vista de la tabla empleados de la base de datos.

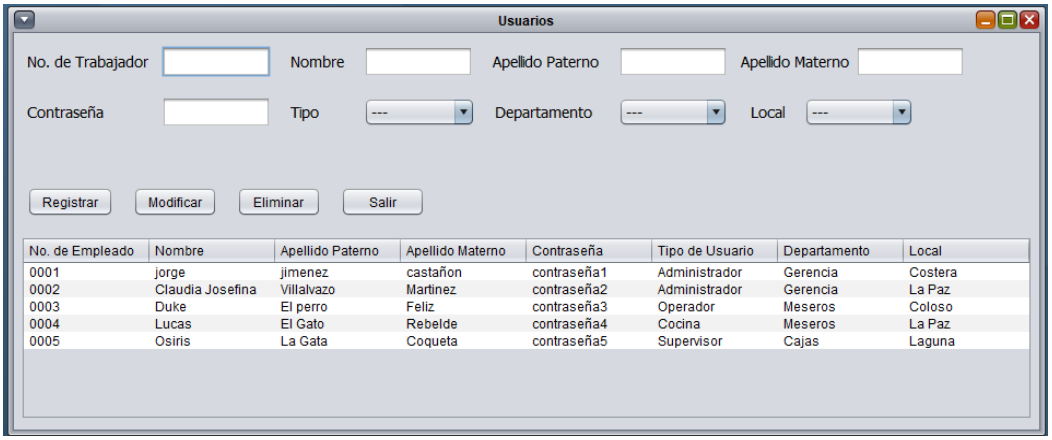

Figura 5.30.- Formulario Usuarios.

5.6.1.- Ingresar datos.

Para el ingreso de los datos a la tabla empleados se requiere el configurar los JComboBox para que desplieguen el contenido que se desea ingresar, por lo que se procede a la generación de los ítems que cada uno de ellos poseerá, para la demostración en el documento se utilizara el JComboBox que corresponde a la llave foránea *Local*.

Se inicia por crear un método, por el cual se cargaran, el cual requiere las siguientes líneas de código:

```
String sql = "Select * from local";


try {
    Statement st=cn.createStatement();
    ResultSet rs=st.executeQuery(sql);
    while (rs.next()){
        CB_LOCAL.addItem(rs.getString("Nombre_loc"));
    }
```

Se comienza realizando la consulta a la tabla que contiene la información a la que se desea accede, en esta ocasión se selecciona todo el contenido de la tabla local, para después agregar al JComboBox (CB_LOCAL) el contenido de la columna "*Nombre_loc*".

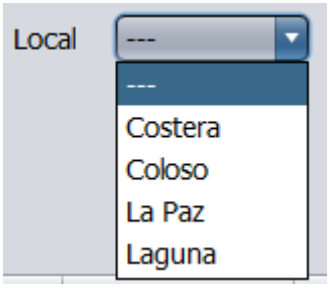

Figura 5.31.- Items del JComboBox CB_LOCAL.

Una vez listos los tres JComboBox con los que cuenta el formulario, se procede a crear un evento en el botón de ingreso (B_INGRESAR_) , con el que se realiza la función de guardar la información contenida en los JTextBox y la seleccionada de los JComboBox.

Antes de realizar el guardado de la información a la base de datos se bebe el id de los ítems seleccionados de los JComboBox, ya que los métodos creados con anterioridad para ellos solo los rellena con las descripciones de las tablas y no con las Id's.

Entonces se crea una variable de tipo String llamada local, consecutivamente de otra del mismo tipo llamada *loc* en la cual se guardara el ítem seleccionado del JComboBox, en este caso para la documentación nuevamente se tomara el JComboBox loca (CB_LOCAL).

```
String local="";
String loc=(String) CB_LOCAL.getSelectedItem();
```

En las siguiente líneas de condigo se estable que se debe seleccionar todo de la tabla local donde el *Nombre_loc* sea igual al contenido en la variable *loc* y se guarda el resultado de la consulta, el cual es el Id_Local en la variable *local*, la cual fue anteriormente creada.

```
Statement st = cn.createStatement();
ResultSet rs = st.executeQuery("Select * from local WHERE Nombre_loc = '"+loc+"' ");
while (rs.next()) {
    local = rs.getString("Id_Local");
```

Ahora que se cuenta con las variables conteniendo la información necesaria se procede a realizar la codificación de ingreso a la base de datos.

```
PreparedStatement pst= cn.prepareStatement
("INSERT INTO empleado(No_Empleado,Nombre,Ap_P,Ap_M,Contraseña,Tipo,Id_Depto,Local)
VALUES (?,?,?,?,?,?,?,?)");
```

```
pst.setString(1,TXT_ID.getText());
pst.setString(2,TXT_NOMBRE_.getText());
pst.setString(3,TXT_AP_.getText());
pst.setString(4,TXT_PM_.getText());
pst.setString(5,TXT_CONTRA_.getText());
pst.setString(6,C_TIPO.getSelectedItem().toString());
pst.setString(7, depto);
pst.setString(8, local);
pst.executeUpdate();

TABLA();
```

Con el código ingresado se guarda en la tabla *empleado* los valores ordenados en la línea de condigo, posteriormente se envían los contenidos de las JTextBox y las variables que contienen los Id's de los ítems seleccionados, luego de realizarlo, se despliega en la JTable (TABLA_USUARUIOS_).

Como se puede ver en la fig. 5.32 los eventos creados funcionan correctamente y permite el ingresar un nuevo registro en la tabla empleado, lo cual se confirma al verlo reflejado en la base de datos fig. 5.33

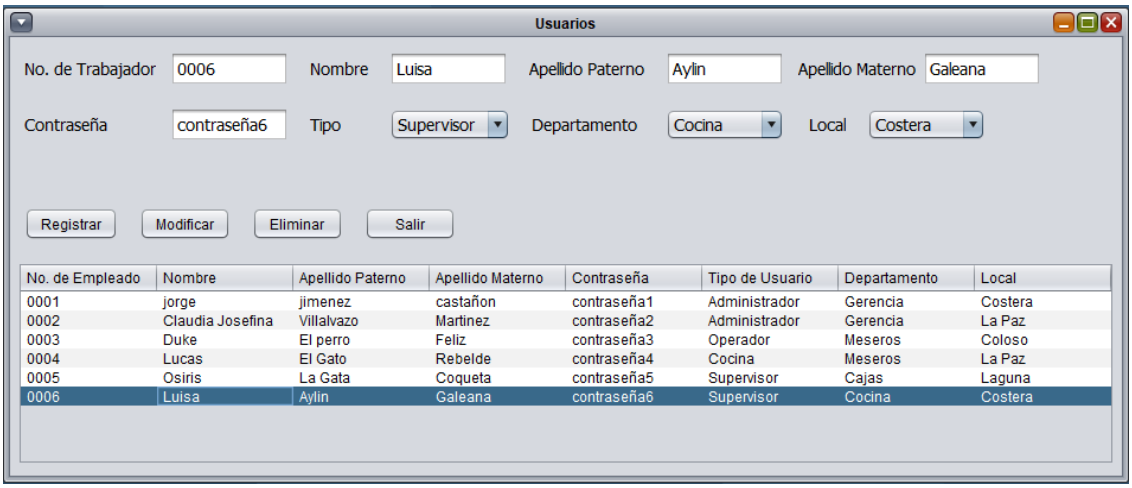

Figura 5.32.- Fila 6 agregada al formulario Usuarios.

No_Empleado	Nombre	Ap_P	Ap_M	Contraseña	Tipo	Id_Depto	Local
0001	jorge	jimenez	castañon	contraseña1	Administrador	0	aca1
0002	Claudia Josefina	Villalvazo	Martinez	contraseña2	Administrador	0	Guz1
0003	Duke	El perro	Feliz	contraseña3	Operador	1	aca2
0004	Lucas	El Gato	Rebelde	contraseña4	Cocina	1	Guz1
0005	Osiris	La Gata	Coqueta	contraseña5	Supervisor	3	Guz2
0006	Luisa	Aylin	Galeana	contraseña6	Supervisor	2	aca1

Figura 5.33.- Fila insertada en la tabla empleado de la base de datos.

5.6.2.- Modificar datos.

En caso de que algún empleado requiera una modificación en su registro, ya sea por cambio de departamento o cambio de perfil de usuario, para lo cual se requiere el obtener los datos de la fila seleccionada del formulario.

```
int fila= TABLA_USUARUIOS_.getSelectedRow();
if(fila>=0){

TXT_ID.setText(TABLA_USUARUIOS_.getValueAt(fila,0).toString());
TXT_NOMBRE_.setText(TABLA_USUARUIOS_.getValueAt(fila,1).toString());
TXT_AP_.setText(TABLA_USUARUIOS_.getValueAt(fila,2).toString());
TXT_PM_.setText(TABLA_USUARUIOS_.getValueAt(fila,3).toString());
TXT_CONTRA_.setText(TABLA_USUARUIOS_.getValueAt(fila,4).toString());
C_TIPO.setSelectedItem(TABLA_USUARUIOS_.getValueAt(fila,5).toString());
CB_DEPTO.setSelectedItem(TABLA_USUARUIOS_.getValueAt(fila,6).toString());
CB_LOCAL.setSelectedItem(TABLA_USUARUIOS_.getValueAt(fila,7).toString());
```

Como se observa en la fig. 5.34 cada vez que una fila es seleccionada, automáticamente se envían los datos de la misma a los JTextField que los corresponde en el caso de los JComboBox con el Item que le corresponda.

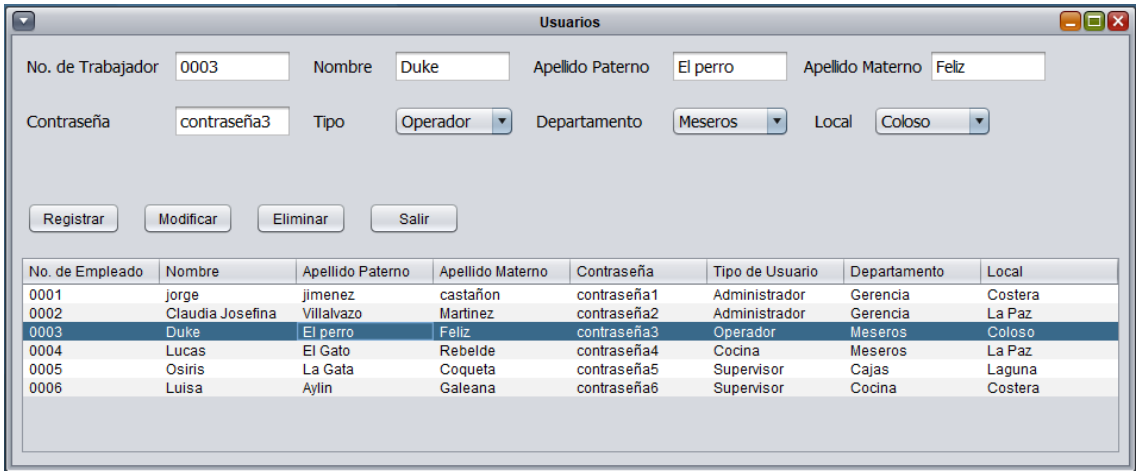

Figura 5.34.- Selección de filas en formulario Usuarios.

Ya obtenida la información necesaria, se crea un nuevo evento en el botón Modificar (B_MODIF_), con el cual se inicia por la selección del correcto Id de cada ítem que sea seleccionado de los JComboBox, que en este caso para demostración de su construcción se utilizara el JComboBox correspondiente a los departamentos (CB_DEPTO).

```
String depto="";
String dep=(String) CB_DEPTO.getSelectedItem();
```

Se crea una variable del tipo String llamada *Depto* después se crea una nueva con el nombre de *dep* la cual contendrá el ítem seleccionada del JComboBox (CB_DEPTO).

```
Statement st = cn.createStatement();

ResultSet rs = st.executeQuery

("Select * from departamento WHERE Nombre_depto = '"+dep+"' ");
```

Se establece la conexión a la base de datos y se realiza la consulta, en la cual se solicita que se seleccione todo el contenido de la tabla departamento donde el *Nombre_depto* sea igual al contenido en la variable *dep*.

```
while (rs.next()) {
    depto = rs.getString("Id_Depto");
}
```

Cuando la consulta es exitosa la variable *depto* guarda el contenido de la columna *Id_Depto* de la fila en la coincida la consulta realizada.

```
PreparedStatement pst = cn.prepareStatement
("UPDATE empleado SET Nombre ='"+TXT_NOMBRE_.getText()+"',
Ap_P='"+TXT_AP_.getText()+"',Ap_M='"+TXT_PM_.getText()+"',
Contraseña='"+TXT_CONTRA_.getText()+"',Tipo='"+C_TIPO.getSelectedItem()+"',
Id_Depto='"+depto+"', Local='"+local+"' WHERE No_Empleado='"+TXT_ID.getText()+"'" );
```

A continuación se realiza nuevamente una consulta en la que se solicita que se actualice el contenido de la tabla empleado, indicando el contenido que se desea modificar y el origen del nuevo contenido a guardar y que lo realice donde el *No_Empleado*, que es el id sea igual al contenido a TXT_ID.

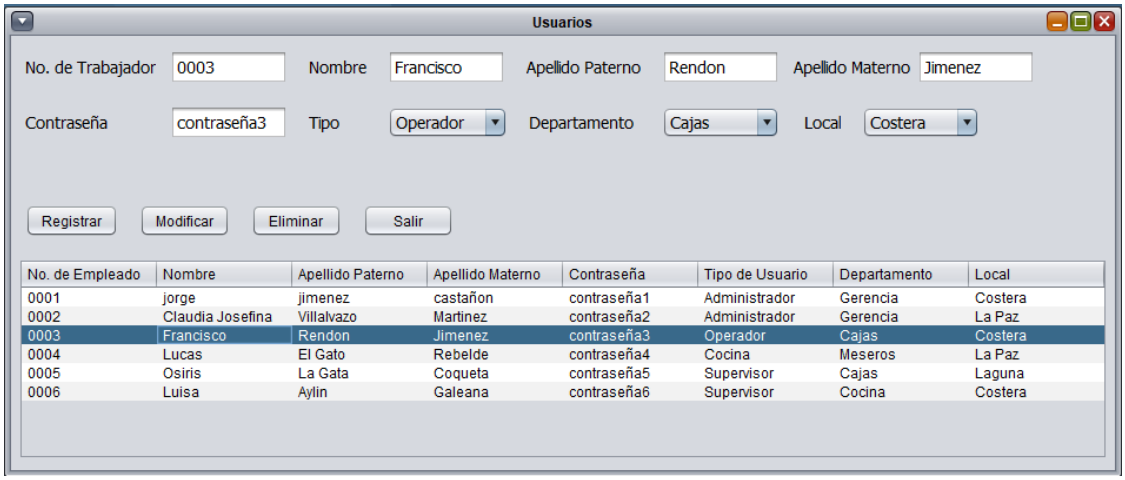

Figura 5.35.- Fila 3 del formulario Usuario modificada.

Como se puede ver en la fig. 5.35 la fila número 3 del formulario ha sido modificada teniendo un contenido completamente nuevo, lo cual se ve reflejado en la base de datos (fig. 5.36). Con lo que se confirma el correcto funcionamiento de la modificación de los datos.

No_Empleado	Nombre	Ap_P	Ap_M	Contraseña	Tipo	Id_Depto	Local
0001	jorge	jimenez	castañon	contraseña1	Administrador	0	aca1
0002	Claudia Josefina	Villalvazo	Martinez	contraseña2	Administrador	0	Guz1
0003	Francisco	Rendon	Jimenez	contraseña3	Operador	3	aca1
0004	Lucas	El Gato	Rebelde	contraseña4	Cocina	1	Guz1
0005	Osiris	La Gata	Coqueta	contraseña5	Supervisor	3	Guz2
0006	Luisa	Aylin	Galeana	contraseña6	Supervisor	2	aca1

Figura 5.36.- Tabla empleado de la base de datos modificada.

5.6.3.- Eliminar datos.

Para los casos en los que el administrador requiera el dejar de contar con los servicios de algún empleado, este puede concluir sus labores y ser eliminado de la base de datos, junto con su usuario y contraseña, para así evitar su acceso al sistema del establecimiento.

```
int fila= TABLA_USUARUIOS_.getSelectedRow();
String Cod="";
Cod=TABLA_USUARUIOS_.getValueAt (fila, 0).toString();
```

Se inicia con la selección de la fila, por lo que se crea un variable del tipo int llamada *fila*, en la que se guarda la fila, seguido de la creación de otra variable de nombre *Cod* en la cual se guarda la columna 0 de la fila seleccionada, la cual es la que contiene el Id del usuario que se desea eliminar.

```
PreparedStatement pst = cn.prepareStatement
("DELETE FROM empleado WHERE No_Empleado='"+Cod+"'");
pst.executeUpdate();
TABLA();
```

Después se realiza la consulta a la base de datos, en la que se elimina de la tabla *empleado* todo donde el *No_Empleado* sea igual, al contenido de la variable *Cod*, ya realizada la operación esta se ve reflejada en la JTable (TABLA_USUARUIOS_) que se encuentra dentro del formulario (fig. 5.37).

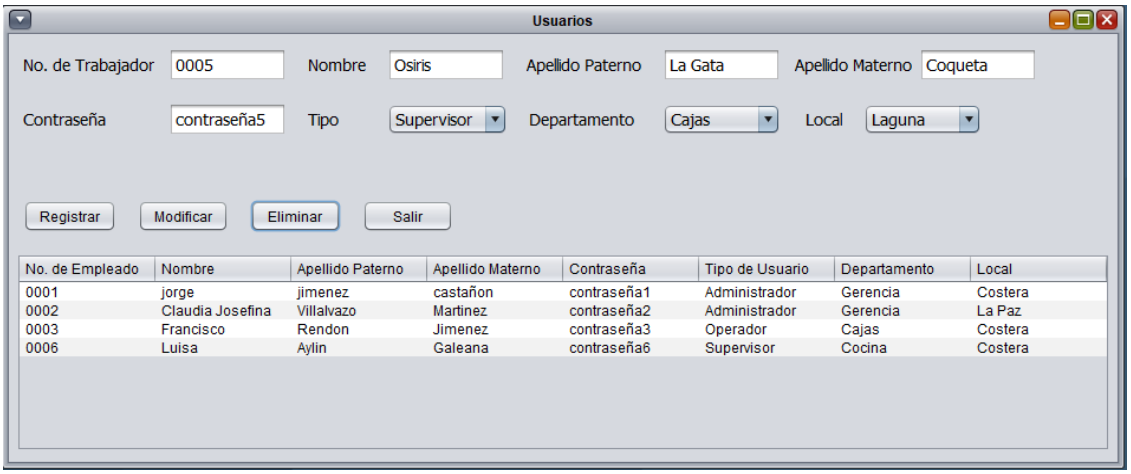

Figura 5.37.- Filas 4 y 5 del Formulario Usuarios.

Como en los formularios anteriores por seguridad el formulario retiene los datos de la fila eliminada para poder ser ingresada nuevamente.

No_Empleado	Nombre	Ap_P	Ap_M	Contraseña	Tipo	Id_Depto	Local
0001	jorge	jimenez	castañon	contraseña1	Administrador	0	aca1
0002	Claudia Josefina	Villalvazo	Martinez	contraseña2	Administrador	0	Guz1
0003	Francisco	Rendon	Jimenez	contraseña3	Operador	3	aca1
0006	Luisa	Aylin	Galeana	contraseña6	Supervisor	2	aca1

Figura 5.38.- Tabla empleado después de la eliminación de filas.

5.7.- Menús de Platillos y Bebidas.

Para la operación de establecimiento es necesario el uso de los menús para el uso de los de los próximos formularios, por lo que, el contar con un formulario en el que se puede gestionar la información de los platillos o bebidas elaborados dentro del establecimiento, por lo que para describir su elaboración en este documento se utilizara el formulario de *Platillos*.

.

Elementos del Formulario Platillos		
Elemento	**Nombre**	**Texto**
JTable	TABLA_PLATILLOS_	
JPanel	JPanel1	
JLabel	JLabel1	ID
JLabel	JLabel2	Nombre
JLabel	JLabel3	Descripción
JLabel	JLabel4	Precio
JLabel	JLabel5	Imagen
JLabel	IMAGEN	
JTextField	TXT_ID	
JTextField	TXT_NOMBRE	
JTextField	TXT_PRECIO	
JTextField	TXT_IMAGEN	
JTextArea	TXT_DESCRIP	
JButton	B_INGRESAR_	Ingresar
JButton	B_MODIFI_	Modificar
JButton	B_ELIMINAR_	Eliminar
JButton	B_SALIR_	Salir
JButton	B_EXAMINAR_	Examinar

Tabla 5.6.- Elementos del Formulario Platillos.

Como se muestra en la tabla 5.6, entre los elementos con los que se construye el formulario, una de las JLabel es nombrada *IMAGEN*, esto es debido a que dicha imagen se utilizara para el despliegue de la imagen del platillo en la base de datos, por lo que al finalizar el interfaz del formulario se muestra en la fig. 5.7.

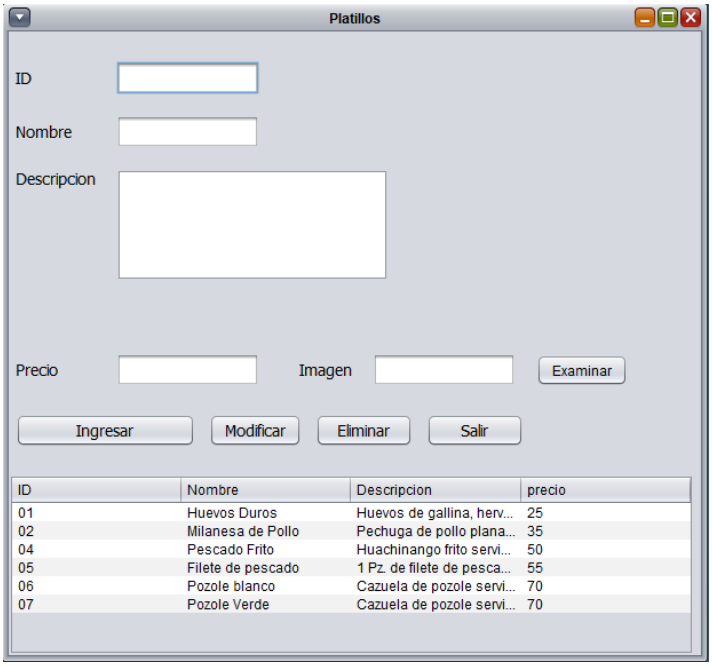

Figura 5.39.- Interfaz gráfica del formulario Platillos.

Para el inicio como en los formularios anteriores se necesita la conexión a la base de datos y la creación de las columnas, por lo que se crea un método con el cual se cargaran los datos desde la tabla platillos.

```
DefaultTableModel modelo=new DefaultTableModel();
modelo.addColumn("ID");
modelo.addColumn("Nombre");
modelo.addColumn("Descripcion");
modelo.addColumn("precio");
```

Con estas líneas de código se agregar a la JTable (TABLA_PLATILLOS_) las columnas que se usaran.

```
CONECTAR cc=new CONECTAR();
Connection cn= cc.Conexion();
```

```
Statement st =cn.createStatement();
ResultSet rs= st.executeQuery("SELECT * FROM platillos");
```

Se continúa con la conexión a la base de datos, junto a la consulta que se utilizara para seleccionar todo de la tabla platillos, lo cual puede confirmarse al comparar los contenidos de la tabla platillos (fig. 5.40) y el formulario (fig. 5.39).

Id_Platillo	Nombre	Descripcion	Precio	Imagen
07	Pozole Verde	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE100184578696600004949 2A0008000000000...
06	Pozole blanco	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE000104A4649460001010000010001 0000FFE...
04	Pescado Frito	Huachinango frito servido con ensalada	50	0xFFD8FFE000104A4649460001010000010001 0000FFD...
02	Milanesa de Pollo	Pechuga de pollo planada y empanizadafreida, acompaña...	35	0xFFD8FFE000104A4649460001010000480048 0000FFD...
01	Huevos Duros	Huevos de gallina, hervidos con hiervasde olor y partio...	25	0xFFD8FFE000104A4649460001010101 2C012C0000FFD...
05	Filete de pescado	1 Pz. de filete de pescado empanizado yfrito, servido c...	55	0xFFD8FFE000104A4649460001010000010001 0000FFF...

Figura 5.40.- Tabla platillos.

5.7.1.- Ingresar Datos.

Ahora que la interfaz esta lista, se procede con el ingresar la información de los platillos que se desplegarán en el menú, pero antes de iniciar se necesita crear un evento que permita el obtener una imagen, para después ser agregada a la base de datos, por lo que dicho evento se crea en el botón *Examinar* (B_EXAMINAR_).

```
FileNameExtensionFilter FILTRO = new FileNameExtensionFilter
("FORMATOS IMAGENES (*.PNG; *.JPG; *.JPEG; *.BMP; *.TIFF; *.GIF)","png","jpg","jpeg","bmp","tiff","gif");
JFileChooser ARCHIVO = new JFileChooser ();
ARCHIVO.addChoosableFileFilter(FILTRO);
```

Con el código se crea un filtro para así se selecciona únicamente los archivos con las extensiones que se necesitan.

```
int VENTANA = ARCHIVO.showOpenDialog(null);
if (VENTANA == JFileChooser.APPROVE_OPTION)
```

Posteriormente con las siguientes líneas de código se abre el cuadro de dialogo fig. 5.41 con el que se seleccionara la imagen en el PC.

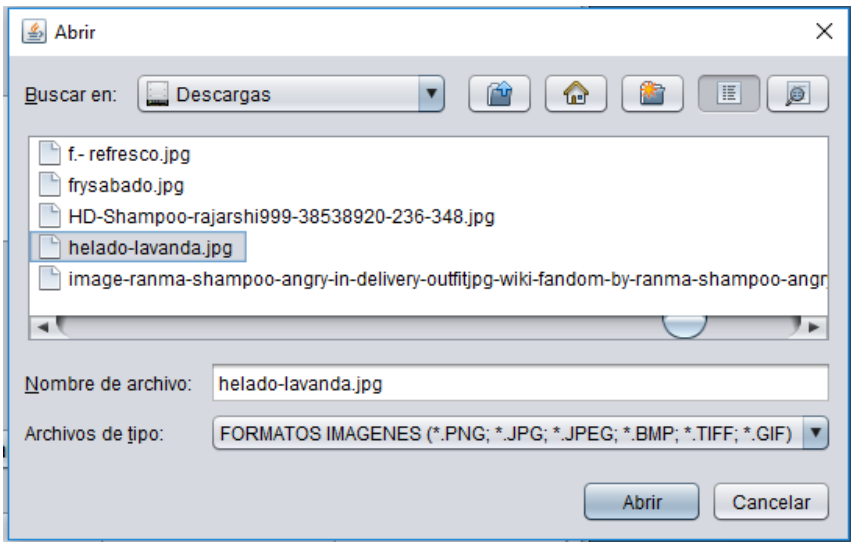

Figura 5.41.- Cuadro de dialogo de formulario Platillos.

Con las siguientes líneas de código, se crea una variable *FOTO* en la que se guarda la ruta de la imagen, y en esta misma después se escala para poder visualizarse en la JLabel *IMAGEN* para posteriormente esta misma ser guardada en la base de datos (fig. 5.42).

```
Image FOTO = getToolkit().getImage(TXT_IMAGEN.getText());
FOTO = FOTO.getScaledInstance(200, 250, Image.SCALE_DEFAULT);
IMAGEN.setIcon(new ImageIcon (FOTO));
```

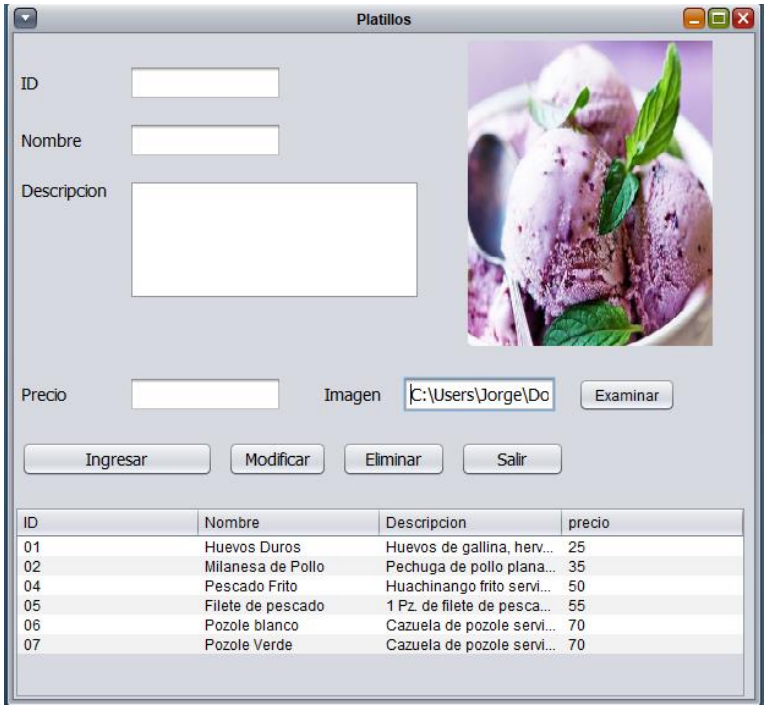

Figura 5.42.- Imagen visualizada en formulario.

Ahora que se cuenta con la imagen dentro de la JLabel *IMAGEN*, se procede a crear en segundo evento para el ingreso de datos a la tabla *Platillos*, este se crea en el botón *Ingresar* en el cual se ingresa el siguiente código:

```
PreparedStatement pst= cn.prepareStatement
("INSERT INTO platillos(Id_Platillo,Nombre,Descripcion,Precio,Imagen)
 VALUES (?,?,?,?,?)");
 FileInputStream ARCHIVOFOTO;
 ARCHIVOFOTO = new FileInputStream(TXT_IMAGEN.getText());
```

En el cual se realiza la consulta a la base de datos en la que se inserta en la tabla *platillos* los datos en las columnas indicadas, además se crea una variable ARCHIVOFOTO en el que se guardara byte por byte la imagen.

```
pst.setString(1,TXT_ID.getText());
pst.setString(2,TXT_NOMBRE.getText());
pst.setString(3,TXT_DESCRIP.getText());
pst.setString(4,TXT_PRECIO.getText());
pst.setBinaryStream(5, ARCHIVOFOTO);
pst.executeUpdate();
TABLA();
```

Posteriormente se guardan los contenidos de los JTextField y el JTextArea en el orden que se indica, además del contenido de la variable *ARCHIVOFOTO* en la tabla *platillo* de la base de datos.

Figura 5.43.- Ingreso de fila en el formulario Platillos.

Como se puede observar en la fig. 5.43, al momento de presionar el botón *Ingresar*, el contenido de los JTextField, así como los contenidos de la JLabel *IMAGEN* junto con el JTextArea son guardados correctamente en la tabla *platillos*, tal como se demuestra en la fig. 5.44.

Id_Platillo	Nombre	Descripcion	Precio	Imagen
07	Pozole Verde	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE1001845786966000049492A0008000000000...
06	Pozole blanco	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE000104A4649460001010000010001 0000FFE...
04	Pescado Frito	Huachinango frito servido con ensalada	50	0xFFD8FFE000104A4649460001010000010001 0000FFD...
02	Milanesa de Pollo	Pechuga de pollo planada y empanizadafreida, acompaña...	35	0xFFD8FFE000104A4649460001010000480048 0000FFD...
01	Huevos Duros	Huevos de gallina, hervidos con hiervasde olor y partio...	25	0xFFD8FFE000104A4649460001010101 2C012C0000FFD...
08	Helado	3 Bolas de helado de lavanda sevido con hojas de men...	30	0xFFD8FFE000104A4649460001010100600060000 0FFD...
05	Filete de pescado	1 Pz. de filete de pescado empanizado yfrito, servido c...	55	0xFFD8FFE000104A4649460001010000010001 0000FFF...

Figura 5.44.- Tabla platillos con fila ingresada.

5.7.2.- Modificar Datos.

En el caso de que algunos de los platillos contenidos en el menú requieran una modificación ya sea en su descripción, nombre o precio, el sistema permite el realizar dichos cambios, por lo que al igual que con los formularios anteriores, se inicia por obtener la información de la fila que se manipulara, para este fin se crea un evento en la JTable.

```
int fila= TABLA_PLATILLOS_.getSelectedRow();
if(fila>=0){

TXT_ID.setText(TABLA_PLATILLOS_.getValueAt(fila,0).toString());
TXT_NOMBRE.setText(TABLA_PLATILLOS_.getValueAt(fila,1).toString());
TXT_DESCRIP.setText(TABLA_PLATILLOS_.getValueAt(fila,2).toString());
TXT_PRECIO.setText(TABLA_PLATILLOS_.getValueAt(fila,3).toString());
```

Primeramente se inicia con obtener los datos de las filas que corresponden a los JTextField y el JTextArea, para posteriormente hacerlo con la imagen que se visualizara en el JLabel *IMAGEN*.

```
String SQL ="SELECT Imagen FROM platillos WHERE Id_Platillo= "+TXT_ID.getText();

ImageIcon FOTO;
InputStream IS;
```

Primeramente de realiza la consulta a la base de datos, en la que se selecciona *imagen* de la tabla *platillo* donde la *Id_Platillo* coincida con el contenido en el JTextField *TXT_ID*, posteriormente se declaran las variables que se utilizaran.

```
Statement st= cn.createStatement();
ResultSet rs= st.executeQuery(SQL);
```

Después se ejecuta la consulta a la base de datos, para posteriormente en la variable IS guarda la secuencia binaria del resultado de la consulta, para después convertirla a imagen nuevamente.

```
IS =rs.getBinaryStream(1);

BufferedImage BI = ImageIO.read(IS);

FOTO = new ImageIcon (BI);
Image IMG = FOTO.getImage();
Image IMG2 = IMG.getScaledInstance(200, 250, Image.SCALE_DEFAULT);
ImageIcon ICON2 = new ImageIcon(IMG2);

IMAGEN.setIcon(ICON2);
```

Usando la variable *FOTO*, en la que se guarda la imagen contenida en *BI*, se vuelve a crear y a escalar la imagen, pasándola nuevamente a la JLabel *IMAGEN* por lo que ya habiendo realizado esto, se podrá obtener la información completa en la fila seleccionada dentro del formulario.

Para la modificación de la información obtenida se procede a realizar la consulta a la base de datos, en la que se indica que se actualicen en la tabla *platillos* los campos indicados, tomando como origen de los nuevos datos el contenido de sus respectivos JTextField.

```
PreparedStatement pst = cn.prepareStatement
("UPDATE platillos SET Nombre ='"+TXT_NOMBRE.getText()+"',
```

```
Descripcion='"+TXT_DESCRIP.getText()+"',Precio='"+TXT_PRECIO.getText()+"'
WHERE Id_Platillo='"+TXT_ID.getText()+"'" );
```

Como se demuestra en la fig. 5.45 la modificación a la fila se realizó exitosamente, siendo confirmado al observar el contenido modificado dentro de la tabla en la base de datos.

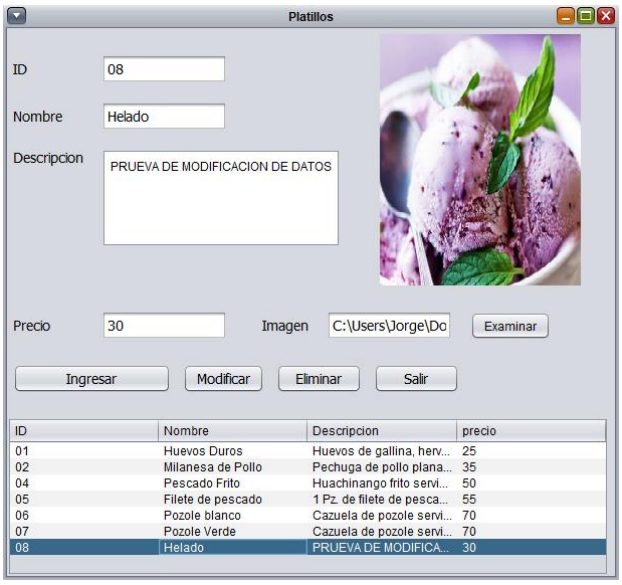

Figura 5.45.- Formulario Platillos con fila 08 modificada.

Id_Platillo	Nombre	Descripcion	Precio	Imagen
07	Pozole Verde	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE1001845786966000049492A0008000000000...
06	Pozole blanco	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE000104A4649460001010000010001000FFE...
04	Pescado Frito	Huachinango frito servido con ensalada	50	0xFFD8FFE000104A4649460001010000010001000FFD...
02	Milanesa de Pollo	Pechuga de pollo planada y empanizadafreida, acompaña...	35	0xFFD8FFE000104A4649460001010000480048000FFD...
01	Huevos Duros	Huevos de gallina, hervidos con hiervasde olor y partio...	25	0xFFD8FFE000104A4649460001010101012C012C0000FFD...
08	Helado	PRUEVA DE MODIFICACION DE DATOS	30	0xFFD8FFE000104A4649460001010100600060000FFD...
05	Filete de pescado	1 Pz. de filete de pescado empanizado yfrito, servido c...	55	0xFFD8FFE000104A4649460001010000010001000FFF...

Figura 5.46.- Tabla platillos modificada.

5.7.3.- Eliminar Datos.

Cuando el administrador determine que uno de los platillos contenidos en la base de datos ya no deba ser preparado dentro del establecimiento, este con total libertad puede retirarlo del menú, como en el caso de los formularios anteriores, al seleccionar una de las filas del

formulario se obtiene la información contenida en ella, por lo que para realizar el borrado, se crea un evento.

```
PreparedStatement pst = cn.prepareStatement
("DELETE FROM platillos WHERE Id_Platillo='"+Cod+"'");
pst.executeUpdate();
TABLA();
```

En el que se borre de la tabla platillo todo el contenido de la fila donde el *Id_Platillo* sea igual al contenido de la variable *Cod*, como se demuestra en las Figuras 5.47 y 5.48, en las que se puede observar como en ambas figuras la fila 08 es eliminada por completo de la base de datos, por seguridad ante errores cometidos por el administrador, la información de la fila borrada es contenida en los JTextField, JTextArea y JLabel, para reingresarse de ser necesario.

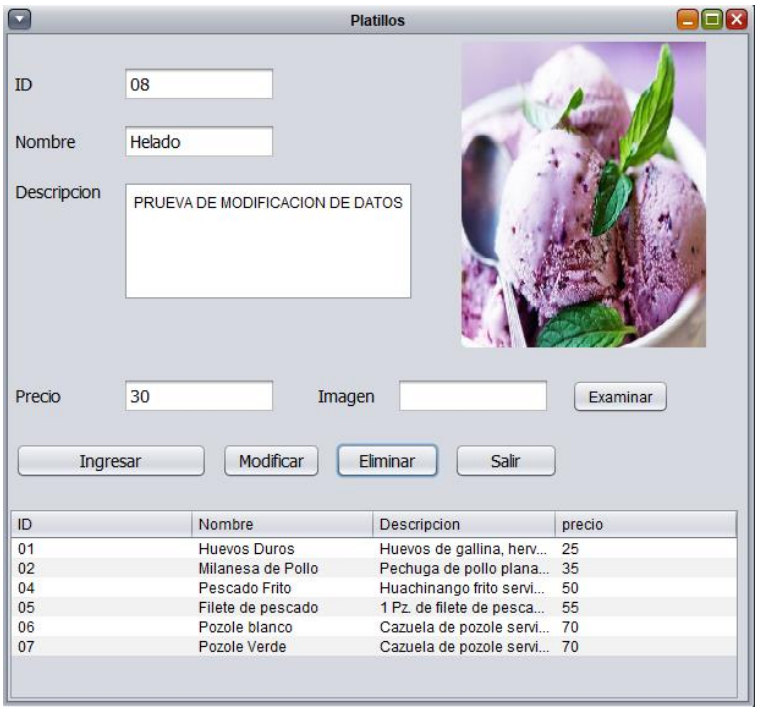

Figura 5.47.- Formulario Platillos después de eliminar la fila 08.

Id_Platillo	Nombre	Descripcion	Precio	Imagen
07	Pozole Verde	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE100184578696600004949 2A0008000000000...
06	Pozole blanco	Cazuela de pozole servida con carne de pollo o puerco ...	70	0xFFD8FFE000104A464946000101000001000100000FFE...
04	Pescado Frito	Huachinango frito servido con ensalada	50	0xFFD8FFE000104A464946000101000001000100000FFD...
02	Milanesa de Pollo	Pechuga de pollo planada y empanizadafreida, acompaña...	35	0xFFD8FFE000104A464946000101000048004800000FFD...
01	Huevos Duros	Huevos de gallina, hervidos con hiervasde olor y partio...	25	0xFFD8FFE000104A464946000101010120120000FFD...
05	Filete de pescado	1 Pz. de filete de pescado empanizado yfrito, servido c...	55	0xFFD8FFE000104A464946000101000001000100000FFF...

Figura 5.48.- Tabla platillos después de la eliminación de la fila 08.

5.8.-Modo Monitor.

El modo monitor permite al usuario de cocina recibir los pedidos que se generan desde piso de venta, visualizando así las ordenes generadas y su contenido, ya que su único privilegio es el modificar el estado de las ordenes, para esta documentación se utilizara el mono monitor correspondiente al usuario *Administrador* ya que cuenta con más privilegios y por lo tanto con todas las funciones.

Elementos de formulario Monitor		
Elemento	**nombre**	**Texto**
JTable	TABLA_ORDENES	
JComboBox	C_EMPLEADO	
JComboBox	C_MENUS	
JComboBox	Combo_estado_	
buttonGroup	buttonGroup1	
JLabel	JLabel1	Total
JLabel	JLabel4	Pzas.
JLabel	JLabel6	Mesero
JTextField	TXT_CANTIDAD	0
JTextField	TXT_TOTAL	0
JRadioButton	RB_BEBIDA	Bebidas
JRadioButton	RB_PLATILLOS	Platillos
JButton	B_INSERTAR_	Insertar
JButton	B_MODIFICAR_	Modificar
JButton	B_Eliminar_	Eliminar
JButton	B_SALIR_	Salir

Tabla 5.7.- Elementos del formulario Monitor.

Con los elementos listados en la tabla 5.7 se construye la interfaz del formulario con la que se podrá visualizar los procesos realizados en los registros de las órdenes existentes.

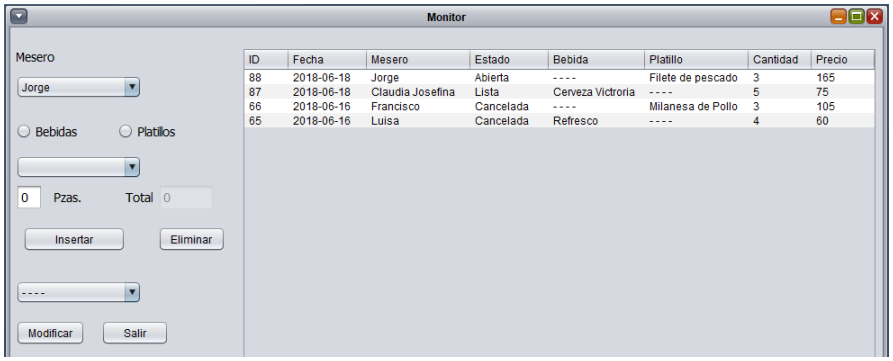

Figura 5.49.- Formulario Monitor.

Para iniciar la construcción del formulario, se crean las columnas que se visualizaran en el JTable *TABLA_ORDENES*, realizar la consulta mediante el método *TABLA ()*.

```
void TABLA(){
    DefaultTableModel modelo=new DefaultTableModel();
        modelo.addColumn("ID");
        modelo.addColumn("Fecha");
        modelo.addColumn("Mesero");
        modelo.addColumn("Estado");
        modelo.addColumn("Bebida");
        modelo.addColumn("Platillo");
        modelo.addColumn("Cantidad");
        modelo.addColumn("Precio");
```

Con dicho método se realiza la conexión a la base de datos, así como la consulta a la base de datos, que para comodidad del usuario, en lugar de visualizar el contenido de la tabla *orden* que cuenta con la información necesaria para los procedimientos, que se visualiza como los Id en el caso de tratarse de una llave foránea, por lo que para evitar confusiones con los identificadores de las llaves foráneas, se crea una vista de la tabla, para así identificar más cómodamente la información desplegada, aunque en la programación si se trabaja directamente con el contenido de la tabla *orden* y los identificadores de dichas llaves.

```
Statement st =cn.createStatement();
ResultSet rs= st.executeQuery("SELECT * FROM v_orden");
```

Como se puede visualizar en la fig. 5.50 el contenido de la tabla *orden* y el visualizado desde la vista creada en el formulario (fig.5.49) coincide.

Id_Orden	Fecha	Mesero	Estado	Bebida	Platillo	Cantidad	Total
88	2018-06-18	0001	Abierta	00	05	3	165
87	2018-06-18	0002	Lista	05	00	5	75
66	2018-06-16	0003	Cancelada	00	02	3	105
65	2018-06-16	0006	Cancelada	04	00	4	60

Id_Orden	Fecha	Nombre	Estado	Nombre_B	Nombre_PL	Cantidad	Total
88	2018-06-18	Jorge	Abierta	- - - -	Filete de pescado	3	165
87	2018-06-18	Claudia Josefina	Lista	Cerveza Victroria	- - - -	5	75
65	2018-06-16	Luisa	Cancelada	Refresco	- - - -	4	60
66	2018-06-16	Francisco	Cancelada	- - - -	Milanesa de Pollo	3	105

Figura 5.50.- Comparación de la tabla orden (arriba y v_orden (abajo) en la base de datos.

5.8.1.- Ingreso de datos.

Para el ingreso de información desde el formulario únicamente pueden realizarlo los usuarios Administrador y Supervisor, que son los únicos que pueden realizar dicha acción desde el sistema, el resto de los usuarios no cuentan con los privilegios necesarios para realizar esta acción, como puede ser el caso del usuario Operador, que es el único que puede realizar un ingreso mediante su dispositivo móvil.

Cabe destacar que el uso de este proceso se desarrolló en caso de sufrir alguna contingencia ajena a los procesos del sistema, por lo que, tomando en cuenta las medidas de seguridad físicas para el equipo en el que se ejecuta y permita el correcto apagado del mismo, para posteriormente realizar los registros de las cuentas manualmente.

Para iniciar el proceso de ingreso a la base de datos, se necesita que la información esté preparada para el proceso, ya que el sistema no permite el ingreso de campos vacíos.

El ingreso de la *Id_Orden* se realiza de manera automática, ya que esta última cuenta con la capacidad de autoincrementarse, la fecha en la que se realiza la operación no es necesario que se ingrese manualmente, el sistema permite el sistema guarda dicha información de manera automática.

```
Date now = new Date(System.currentTimeMillis());
SimpleDateFormat date = new SimpleDateFormat("yyyy-MM-dd");
```

Con las líneas de código anteriores se genera la fecha tomando la misma con la que cuenta el sistema operativo y posteriormente dando el formato con el que se guardara en la tabla *orden*.

El siguiente dato a preparar es el mesero que genera la orden, por lo que en el caso del sistema en PC, este se guarda de manera manual, es decir seleccionando el mesero que levanto la orden, para esta acción se crea un método en el JComboBox
C_EMPLEADO, para cargar su contenido.

```
String sql = "Select * from empleado";

try {
    Statement st=cn.createStatement();
    ResultSet rs=st.executeQuery(sql);
    while (rs.next()){
    C_EMPLEADO.addItem(rs.getString("Nombre"));
    }}
```

Con dicho método se selecciona el contenido de la tabla *empleado*, para posteriormente mostrar el contenido de la columna *Nombre*, dicho método se utilizar también para el resto de los JComboBox que se necesitan para la carga de los menús de platillos y bebidas en sus respectivos JComboBox.

El siguiente dato a preparar es el estado de la cuenta, este automáticamente, guarda la orden generada con el estado de *Abierta*, por lo que el siguiente paso es preparar el contenido de la orden, dicho sea el menú de platillos o bebidas que se elaboran dentro del establecimiento, para ello se crean dos eventos, uno en cada JRadioButton, para así cargar el contenido de dicho menú dentro del JComboBox *C_MENUS*, el cual es el mismo metodo que se utiliza en el caso del JComboBox correspondiente a los empleados, salvo la consulta y el nombre del JComboBox.

Después de haber seleccionado la bebida o platillo se ingresa manualmente la cantidad de dicha selección en el JTextField *TXT_CANTIDAD*, con lo que finalmente solo queda el realizar la operación de cálculo del total de la orden que se genera, para lo cual se realiza un método en el JComboBox *C_MENUS*.

```
public static String global;
```

Antes de la creación del método se declara una variable global del tipo String nombrada *global*, la cual debe realizar la función de almacenar el contenido de la columna *Precio* del menú que es seleccionado.

Con la selección del menú a desplegar en los JRadioButton se inicia el próximo evento que como se mencionó anteriormente está contenido en JRadioButton, por lo que se inicia definiendo el valor de la variable del tipo int *q*, la cual dependiendo de la selección realizada por el usuario obtendrá el valor de 1 o 2.

```
int q=0;

if (RB_BEBIDA.isSelected()){
  q=1;
}  else{
    q=2;
}
```

Tomando en cuenta el caso en que la selección haya sido el JRadioButton RB_BEBIDA y q obtenga el valor de 1, se realiza el case correspondiente, por lo que inicia obteniendo el ítem seleccionado del JComboBox *C_MENUS,* después se obtiene los valores convertidos a int de los contenidos en los JTextField *TXT_CANTIDAD* y *TXT_TOTAL*.

```
switch (q){
    case 1:

          String y=(String) C_MENUS.getSelectedItem();

          Integer a = Integer.valueOf(TXT_TOTAL.getText());
          Integer b = Integer.valueOf(TXT_CANTIDAD.getText());
          int c;
          try {
```

Posteriormente se realiza la conexión a la base datos para ejecutar la consulta en la que se selecciona todo el contenido de la tabla *bebida*, donde el *Nombre_B* sea igual al contenido en *y*, que es la variable que contiene el ítem seleccionado del JComboBox *C_MENUS.*

```
          Statement st = cn.createStatement();
          ResultSet rs = st.executeQuery

          ("Select * from bebida WHERE Nombre_B = '"+y+"' ");3
```

Una vez realizada la consulta, mientras haya sido exitosa la variable *global* anteriormente declarada obtiene el valor de la columna *Precio*.

```
while (rs.next()){
global = rs.getString("Precio");
```

Ya contando con el precio del platillo o la bebida seleccionada, se procede a realizar la operación matemática, para dicho propósito la variable *c* del tipo int, al finalizar debe contener el resultado de la operación, se convierte el contenido de la variable *global* para después ser multiplicado por la variable *b*, la cual contiene la cantidad ingresada por el usuario y el resultado es sumado por la cantidad almacenada dentro de la variable *a*, la cual contiene el valor almacenado en *TXT_TOTAL* (en caso de haber realizado una operación previa).

```
c= (Integer.valueOf(global)*b)+a;
TXT_TOTAL.setText(String.valueOf(c));
break;
```

Con este proceso la información ya se encuentra lista para el ingreso a la base de datos, por lo que, se crea el método principal para dicho propósito, el cual será contenido en el JButton *B_INSERTAR_*.

Reutilizando el código para la obtención de los precios correspondientes a los ítems seleccionados, en el método de ingreso se adapta para que sea la columna correspondiente al Id correspondiente a la selección de los JComboBox. Por lo tanto una vez que la información está lista para el ingreso y se obtienen los Id de cada selección realizada se procede a realizar el ingreso.

```
PreparedStatement pst= cn.prepareStatement
("INSERT INTO orden(Fecha,Mesero,Estado,Bebida,Platillo,Cantidad,Total)
VALUES (?,?,?,?,?,?,?)");
```

Realizando la conexión a la base de datos también se ejecuta la consulta en la que se indica que se inserte en la tabla *orden*, los registros que se listan, dado que en este caso, no puede haber campos vacíos se indica que se realice la acción en todos los campos.

El siguiente paso en el proceso es indicar el origen de la información a ingresar, indicando por orden, que debe ser el mismo que el descrito en la consulta, comenzando con la fecha del sistema, el id del empleado que genera la orden, seguido del estado de la orden, el cual debe ser *Abierta*, en seguida, usando el método para la extracción de precios, se aplica a la extracción del id de los platillos y bebidas, los cuales están almacenados en las variables *A* y *B*, en seguida se obtiene los últimos datos de los JTextField correspondientes al total de la cuenta y la cantidad de la misma.

```
pst.setString(1, date.format(now));
pst.setString(2, Empleado);
pst.setString(3, "Abierta");
pst.setString(4, A);
pst.setString(5, B);
pst.setString(6,TXT_CANTIDAD.getText());
pst.setString(7,TXT_TOTAL.getText());

pst.executeUpdate();
TABLA();
```

Finalizando con la actualización de la consulta y generando nuevamente el contenido de la JTable *TABLA_ORDENES*.

5.8.2.- Modificar Datos.

Una vez generada una orden, esta no puede ser modificada, al menos en su contenido, lo único que puede modificarse del registro es el estado de la misma, ya que permite el tener un mejor control de los tiempos de preparación.

Para el realizar la operación de modificar el estado de una orden se requiere que el sistema reconozca la fila que se está seleccionando, por lo que se ingresara el siguiente código en el evento que corresponde a la selección de dicha fila el cual está contenido en la JTable *TABLA_ORDENES*.

```
int fila= TABLA_ORDENES.getSelectedRow();
if(fila>=0){
Combo_estado_.setSelectedItem(TABLA_ORDENES.getValueAt(fila,1).toString());
}
```

Con esta acción, del contenido de la fila seleccionada, solo el estado de la orden es enviado a su correspondiente JComboBox. Una vez realizada se procede a crear otro evento en el botón Modificar, dicho código será para la actualización de la tabla *orden* en la columna correspondiente al estado de la orden en cuestión.

```
int fila= TABLA_ORDENES.getSelectedRow();
String id = (String) TABLA_ORDENES.getValueAt(TABLA_ORDENES.getSelectedRow(), 0);


PreparedStatement pst = cn.prepareStatement

("UPDATE orden SET Estado='"+Combo_estado_.getSelectedItem()+"' WHERE Id_Orden='"+id+"'" );
```

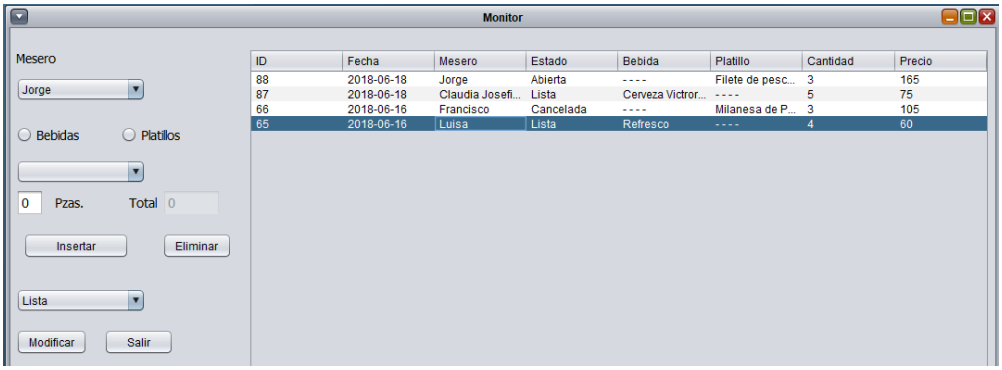

Figura 5.51.- Formulario Monitor con la fila 65 modificada.

Como se muestra en la fig. 5.51 la fila seleccionada fue modificada en el estado de la misma, pasando de estar lista a cancelada, esto último se puede confirmar al observar la tabla orden en la base de datos.

Id_Orden	Fecha	Nombre	Estado	Nombre_B	Nombre_PL	Cantidad	Total
88	2018-06-18	Jorge	Abierta	----	Filete de pescado	3	165
87	2018-06-18	Claudia Josefina	Lista	Cerveza Victroria	----	5	75
65	2018-06-16	Luisa	Lista	Refresco	----	4	60
66	2018-06-16	Francisco	Cancelada	----	Milanesa de Pollo	3	105

Figura 5.52.- Estado de la orden 65 en la base de datos modificada.

5.8.3.- Eliminar Datos.

En caso de requerir el retirar una orden del formulario y la base de datos, el formulario lo permite con el uso de un solo botón, con el que se elimina la fila seleccionada, donde la *Id_Orden* sea igual al contenido de la variable *Cod*.

```
PreparedStatement pst = cn.prepareStatement
("DELETE FROM orden WHERE Id_Orden='"+Cod+"'");
```

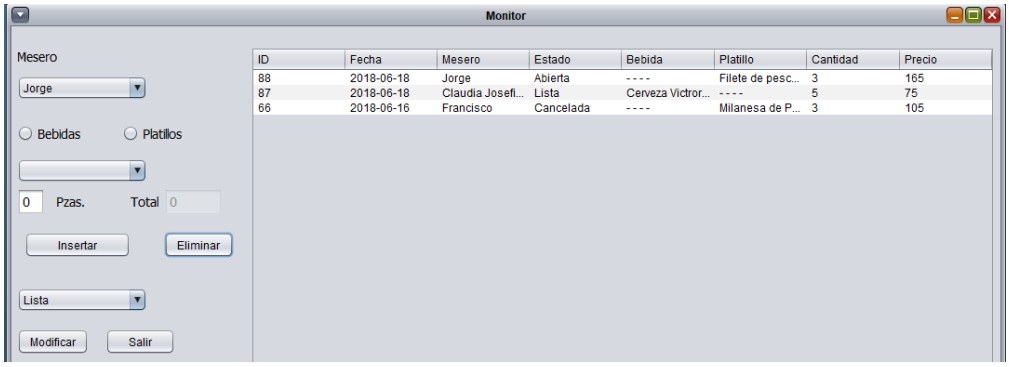

Figura 5.53.- Fila 65 eliminada del formulario Monitor.

Como se puede observar en el formulario (fig.5.53) la fila 40 ha sido eliminada y esto se confirma al compararlo con el contenido de la tabla orden en la base de datos (fig.5.54).

Id_Orden	Fecha	Nombre	Estado	Nombre_B	Nombre_PL	Cantidad	Total
88	2018-06-18	Jorge	Abierta	- - - -	Filete de pescado	3	165
87	2018-06-18	Claudia Josefina	Lista	Cerveza Victroria	- - - -	5	75
66	2018-06-16	Francisco	Cancelada	- - - -	Milanesa de Pollo	3	105

Figura 5.54.- Orden 65 eliminada de la tabla orden.

5.9.- Módulo móvil.

En el módulo móvil del sistema, el usuario Operador, el cual únicamente le es asignado a los empleados que laboran en piso de venta, específicamente los meseros por lo que dentro del módulo móvil, una vez identificados, los usuarios solo pueden realizar la consulta de los menús de platillos y bebidas, por lo que primero se asignan los procesos que se realizaran utilizando archivos php dentro de la carpeta TESIS, la cual está contenida en la carpeta www de WAMPSERVER

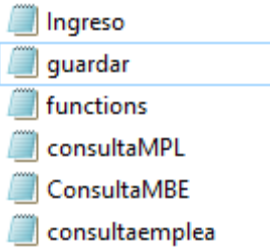

Figura 5.55.- Archivos PHP en carpeta TESIS.

Para iniciar, se necesita realizar la conexión a la red del dispositivo, por lo cual se necesitan los permisos para ingresar, esto se logra mediante el uso del siguiente código que deben agregarse a la manifest de la aplicacion:

```
package="com.example.jorge.prototipo">
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Sin las últimas dos líneas la aplicación no puede conectarse a la red y por lo tanto no realizara sus funciones, una vez que la aplicación cuenta con las permisos para ingresar a la red, se necesita el acceder a los archivos que contienen las funciones a realizar.

Ya una vez con los permisos adquiridos se ingresa un método con el cual se obtiene la url que se necesita para ingresar a los archivos contenidos en la carpeta www de WAMPSERVER.

```
private String downloadUrl(URL url) throws IOException {
    InputStream stream = null;
    HttpsURLConnection connection = null;
    String result = null;
```

Ya obtenida la url que, lo que las siguientes líneas de codgo hacen, es el abrir la conexión, seguido de definir el tiempo de lectura y conexión, el cual se dejara con los valores de 3000, ya que son valores que recomienda Android Studio en sus sugerencias para conectar a la red la aplicación, seguido del método GET, el cual es el que método que se utiliza para la funcionalidad de la aplicación.

```
connection = (HttpsURLConnection) url.openConnection();
connection.setReadTimeout(3000);
connection.setConnectTimeout(3000);
connection.setRequestMethod("GET");
```

Hora que se cuenta con los métodos necesarios para la obtener la url y establecer la conexión de la aplicación a la red se procede a la construcción del Layout, por lo que se necesitaran los elementos de la tabla 5.8

.

Elementos del Layout Inicio	
Elemento	**Nombre**
TexView	TexView
TexView	TexView2
ImageView	ImageView
editText	editText
editText	editText2
Button	Button
Button	Button2

Tabla 5.8.- Elementos de Layout Inicio.

Una vez implementados los elementos se crea la pantalla mostrada en la fig. x.x, por lo que ahora solo queda el crear el método que se utilizara para la consulta de la base de datos.

Figura5.56.- Layout inicio.

5.9.1.- Inicio de sesión.

Para el ingreso al sistema mediante la aplicación móvil se necesita que el usuario ingrese su id y contraseña, que previamente han sido asignados por el usuario Administrador. Por lo que primero se declara la url del archivo que se utiliza para el aceso al sistema.

New

Consulta().execute(**"http://localhost/TESIS/Ingreso.php?No_Empleado="**editText.getTex t().toString()+**"Contraseña="**editText2.getText().toString());

Como se puede observar se especifica el archivo php en el que se encuentra la función para el ingreso al sistema, seguido de los datos a buscar dentro de la tabla empleado.

En el archivo php se encuentra el método para la consulta de los datos de la tabla a utilizar, en este caso es la tabla empleado, en la que se especifica que seleccione todo el contenido de la tabla *empleado*, donde el *No_Empleado* sea igual al proporcionado por el sistema.

```
<?php
include('functions.php');
$No_Empleado=$_GET["No_Empleado"];

if($resultset=getSQLResultSet("SELECT * FROM `empleado` WHERE No_Empleado='$No_Empleado'")){
        while ($row = $resultset->fetch_array(MYSQLI_NUM)){
                echo json_encode($row);
        }
}

?>
```

Con lo que si los datos proporcionados son los correctos, el usuario ingresa al sistema sin problema alguno. Cambiando a la siguiente ventana utilizando el código:

```
Intent.menu= new Intent(MainActivity.this, Main2Activity.class)
```

En el que indicamos el Activity en el que se encuentra y posteriormente el Activity al que se desea ingresar.

5.9.2.- Menús

Una vez ingresado al sistema el usuario solo cuenta con la opción de realizar el registro de la orden a la tabla *orden* de la base de datos.

Elementos del Layout Orden	
Elemento	**Nombre**
LinearLayout	LinearLayout1
LinearLayout	LinearLayout2
LinearLayout	LinearLayout3
LinearLayout	LinearLayout4
LinearLayout	LinearLayout5
LinearLayout	LinearLayout6
Button	Button1
Button	Button2
Button	Button3
Button	Button4
Button	Button5
TextView	TextView1
TextView	TextView2
TextView	TextView3
TextView	TextView4
TextView	TextView5
editText	editText1
editText	editText2
editText	editText3
editText	editText4
editText	editText5

Tabla 5.9.- Elementos de Layout Menu.

Utilizando los elemento de la tabla 5.9, el layout mostrado en la fig. 5.57 en el cual se muestra el contenido de la tabla bebida o platillos, dependiendo del menú a elegir, por lo que se utilizan los LinearLayout, el LinearLayout1 corresponde a la horientacion vertical y LinearLayout2 en adelante corresponden a la horizontal

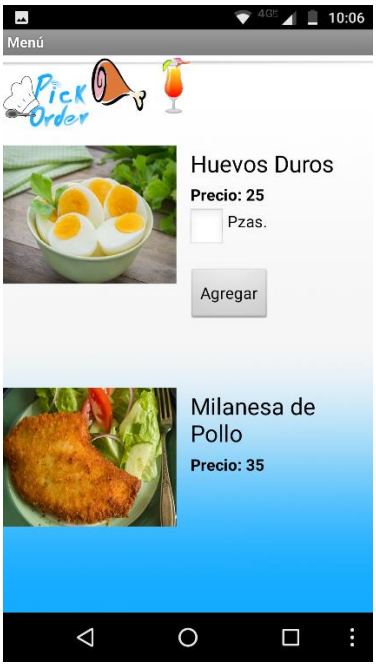

Figura 5.57.- Layout Menu.

En cada imageView se mostraran las imágenes extraídas de la tabla correspondiente al menú seleccionado en la pantalla.

```
Bitmap bitmap = BitmapFactory.decodeStream(is);
ImageView imageView = (ImageView) findViewById(R.id.image_view);
imageView.setImageBitmap(bitmap);
```

Con lo que para ingresar el Nuevo registro a la base de datos se realiza la misma línea de código para el ingreso, con la diferencia de que ahora se ingresa al archivo php *guardar*, el cual es responsable de guardar la selección del layout menú en la tabla orden:

```
<?php include ('functions.php');
$Fecha=$_GET['Fecha'];
$Estado=$_GET['Estado'];
$Mesero=$_GET['Mesero'];
$Bebida=$_GET['Bebida'];
$Platillo=$_GET['Platillo'];
$Cantidad=$_GET['Cantidad'];
$Precio=$_GET['Precio'];

ejecutarSQLCommand("INSERT INTO  `orden` (Fecha, Estado, Mesero, Bebida, Platillo, Cantidad, Precio )
VALUES (
'$Fecha',
'$Estado',
'$Mesero',
'$Bebida',
'$Platillo',
'$Cantidad',
'$Precio',
)
```

Conclusiones.

Con el uso del prototipo este sistema de gestión de pedidos (app) se pueden monitorear en tiempo real la carga de trabajo por usuario, las transacciones realizadas y los tiempos de espera entre una comanda y otra.

Esto permitirá el mejorar la administración del establecimiento y por lo tanto una mejor operación durante la jornada diaria al posteriormente agregarse módulos que se adapten a la operación del establecimiento.

Como se mencionó en la hipótesis, con el desarrollo de este proyecto se pretende reducir los tiempos de espera en la operación de piso de venta en un restaurante, lo que permitirá brindar una mejor percepción del servicio por parte del cliente, lo que se reflejará en la reducción de las cancelaciones, lo que permitiría el reducir la perdida producida durante la operación del piso de venta.

Ya que las pérdidas relacionadas con las cancelaciones durante un periodo vacacional pueden llegar a ser hasta el 10% de los ingresos percibidos, con el uso de esta aplicación se reducirían las cancelaciones llegando éstas a un porcentaje aproximado del 2%.

Por lo que tomando en cuenta que al día en promedio se procesan 150 órdenes, el 2% corresponde a 3 de esas órdenes con un costo promedio de $70 resulta un costo total de $210 diarios, lo que nos daría $ 1,470 semanales que se acumulan hasta ser $76,440 anual que representaría el nuevo costo de las mermas, si lo restamos al 10% de las mermas que se

producen sin el uso de la aplicación móvil que nos representa un costo aproximado de $382,200, obtenemos una diferencia de $305, 160 pesos de ahorro con el uso de la aplicación móvil.

Anexos.

Aseguramiento Técnico Material

Para el desarrollo e implementación del sistema gestor a desarrollar, se necesita contar con las herramientas necesarias, tanto de software como de hardware, por lo que para comprobar la viabilidad del sistema se deben de contemplar los costos de desarrollo que contempla la adquisición de dichas herramientas, así como el costo humano por las horas invertidas en la codificación del sistema.

El costo de implementación, en el cual se contemplan los equipos necesarios para el funcionamiento correcto del sistema dentro del ambiente para el que ha sido desarrollado.

Costo de Desarrollo

El desarrollo del sistema gestor de cuenta con la ventaja de utilizar herramientas de desarrollo de software gratuitas, sin embargo, el equipo necesario para utilizar dichas herramientas tienen un costo para su adquisición y el costo de las horas hombre utilizadas en la codificación del sistema.

En la siguiente tabla se especifican las características y los costos de las herramientas necesarias para el desarrollo del sistema, así como la codificación de los módulos necesarios para su funcionamiento, tomando en cuenta que actualmente el salario de un programador junior es de $ 34.00 por hora.

Descripción	Características	Consto MXN
Computadora	Laptop Dell Inspiron 15,Sistema Operativo Windows 10, Disco duro: 1 TB, Memoria: 8Gb, Procesador: Intel i5	$12,000.00
Tablet	Samsung Galaxy Tab S2, Procesador Octa-core, Velocidad de procesador: 1.9 GHz, Memoria Interna: 32 Gb, Memoria RAM: 3Gb, Sistema Operativo Android 7.0 (Nougat)	$5,900.00
Celular	ZTE Blade V6, Procesador: **Mediatek MT6752,** Velocidad de procesamiento: 1.7 GHz, Memoria Interna: 16 GB, Memoria Ram: 2GB, Sistema operativo: Android 5.1.1 (lolipop)	$2,500.00
Procesador de Texto	Open office con licencia gratuita.	$0.00
NetBeans	Entorno de programación Java de licencia gratuita	$0.00
Eclipse	Entorno de programación Java de licencia gratuita para migración de código	$0.00
María DB	Gestor de base de datos gratuito	$0.00
Android Studio	Entorno de programación y emulación de aplicaciones para sistema Android de licencia gratuita	$ 0.00
Codificación del Sistema	* Inicio de Sesión (12 hrs.) $ 408.00 * Menú de Platillos (10 hrs) $340.00 * Menú de Bebidas (8 hrs) $ 272.00 *Reportes de Ventas (22hrs) $ 748.00 *Seguridad (20 hrs) $ 680.00 * Comandas (38 hrs) $ 1,292.00 *Cocina (12 hrs) $ 480.00	$4,220.00
Total		$ 24,620.00

Tabla A.0.1.- Costos de Desarrollo.

Costo de Implementación

El costo de implementación que se presenta a continuación solo contempla la implementación del prototipo que se implementara en el establecimiento que permitió el acceso y la implementación de dicho sistema.

Descripción	Características	Consto MXN
Computadora	Disco duro 1TB, Memoria 16 GB, Procesador I7	$ 14,399.00
Computadora Tink client	HP T620 Flexible Thin Client, procesador AMD GX-217GA a 1.65 GHz Dual Core, Memoria RAM 4Gb, Disco duro 16GB,	$ 2,500.00
Swicht	Switch Cisco Fast Ethernet WS-C2960-24LT-L, 24 puertos 10/100 Mbps, 1 Gbit/s	$ 1,600.00
Antena (Access Point)	Access Point Cisco WAP121, Inalambrico, 300 Mbit/s, 2.4GHz	$ 1,639.00
Celular	ZTE Blade V6, Procesador: **Mediatek MT6752,** Velocidad de procesamiento: 1.7 GHz, Memoria Interna: 16 GB, Memoria Ram: 2GB, Sistema operativo: Android 5.1.1 (lolipop)	$ 2,500.00
Impresora térmica	Impresora Termica punto de venta usb de 58 mm	$ 600.00
Gastos diversos	*rollo de cable UTP categoría 5e 305 metros (2 rollos) $ 1400.00 *paquete de Conectores RJ45 (100 pzas) $ 200.00 *Pinzas de ponchado $ 370.00	$ 1,900.00
Total		$ 25,138.00

Tabla A.0.2.- Costos de Implementación.

Bibliografía.

[Arturo Baz Alonzo, 2012a] Arturo Baz Alonzo, Irene Ferreira Artime, R. G. B. (2012a). Dispositivos móviles. Ingeniería de Telecomunicación, Universidad Oviedo, Página 2.

[Arturo Baz Alonzo, 2012b] Arturo Baz Alonzo, Irene Ferreira Artime, R. G. B. (2012b). Dispositivos móviles. Ingeniería de Telecomunicación, Universidad Oviedo, Página 3.

[Arturo Baz Alonzo, 2012c] Arturo Baz Alonzo, Irene Ferreira Artime, R. G. B. (2012c). Dispositivos móviles. Ingeniería de Telecomunicación, Universidad Oviedo, Página 5.

[CORPORATION, 2012] CORPORATION, I. (2012). Como garantizar la seguridad de las aplicaciones para dispositivos móviles. Software Group, Página 3.

[Corporation, 2015] Corporation, O. (2015). Conozca más sobre la tecnología java. https://www.java.com/es/about/.

[De la Vega, 2012a] de la Vega, E. C. M. . M. S. (2012a). Aplicación móvil para el control de pedidos en un restaurante. Departamento de Ingeniería de Sistemas y Telecomunicaciones, Universidad de Córdoba, Página 2.

[De la Vega, 2012b] de la Vega, E. C. M. . M. S. (2012b). Aplicación móvil para el control de pedidos en un restaurante. Departamento de Ingeniería de Sistemas y Telecomunicaciones, Universidad de Córdoba, Página 1.

[De la Vega, 2012c] de la Vega, E. C. M. . M. S. (2012c). Aplicación móvil para el control de pedidos en un restaurante. Departamento de Ingeniería de Sistemas y Telecomunicaciones, Universidad de Córdoba, Página 4.

[Developer android, 2016a] developer android (2016a). Conoce android studio. https://developer.android.com/studio/intro/index.html?hl=es-419.

[Developer android, 2016b] developer android (2016b). Guía de usuario, , módulos. https://developer.android.com/studio/projects/index.html?hl=es-419.

[Developer android, 2016c] developer android (2016c). Guía de usuario, depuración integrada. https://developer.android.com/studio/intro/index.html?hl=es-419#debug-perf.

[developer android, 2016d] developer android (2016d). Guía de usuario, estructura del proyecto. https://developer.android.com/studio/intro/index.html?hl=es-419#project-structure.

[developer android, 2016e] developer android (2016e). Guía de usuario, sistema de compilación de gradle. https://developer.android.com/studio/intro/index.html?hl=es-419#sistemadecompilaciondegradle.

[Dr. K.Baskara, 2012] Semantic Information Retrieval sing WAMP Server, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 11.

[D.Valencia, 2012] D.Valencia, V.Andrade, L. . J. (2012). Diseño e implementación de una aplicación móvil de gestión de pedidos para restaurantes en barranquilla. Universidad Simón Bolívar, Página 1.

[Gabriel, 2014] Gabriel, R. (2014). La importancia de la computación móvil: pasado, presente y futuro. Revista Especializada en Telecomunicaciones, Electrónica y Sistemas, Volumen 2, Número 2, Página 3.

[Gallego, 2012a] Gallego, M. T. (2012a). Gestión de proyectos informáticos, metodólogía scrum. Página 34.

[Gallego, 2012b] Gallego, M. T. (2012b). Gestión de proyectos informáticos, metodología scrum. Página 32 & 33.

[Group, 2016] Group, C. F. (2016). A recipe for success: Technology & the restaurant industry. Citizens Commercial Banking, Page 1.

[Gupta, 2012a] Gupta, P. . E. (2012a). Application on order management system in restaurants. International Journal of Application or Innovation in Engineering & Management, Page 61.

[Gupta, 2012b] Gupta, P. . E. (2012b). Application on order management system in restaurants. International Journal of Application or Innovation in Engineering & Management, Page 59.

[H, 2009] H, J. S. D. . R. (2009). Introducción a la programación con java. Mc Graw Hill.

[Holzner, 2001] Holzner, S. (2001). La biblia de java 2. ANAYA Multimedia.

[INEGI, 2016] INEGI (2016). Anuario estadístico y geográfico de los Estados Unidos Mexicanos. INEGI edición 2016, ISBN 978-607-530-009-2, Página 723.

[Ing. Pedro Julio Colorado Ángel, 2015a] Ing. Pedro Julio Colorado Ángel, I. I. J. T. B. (2015a). ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE. UNAD, Escuela de Ciencias Básicas, Tecnología e IngenierÍa, Especialización en Seguridad Informática, Página 30.

[Ing. Pedro Julio Colorado Ángel, 2015b] Ing. Pedro Julio Colorado Ángel, I. I. J. T. B. (2015a). ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE. UNAD, Escuela de Ciencias Básicas, Tecnología e IngenierÍa, Especialización en Seguridad Informática, Página 24.

[Ing. Pedro Julio Colorado Ángel, 2015c] Ing. Pedro Julio Colorado Ángel, I. I. J. T. B. (2015a). ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE. UNAD, Escuela de Ciencias Básicas, Tecnología e IngenierÍa, Especialización en Seguridad Informática, Página 25.

[Ing. Pedro Julio Colorado Ángel, 2015d] Ing. Pedro Julio Colorado Ángel, I. I. J. T. B. (2015a). ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL

SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE. UNAD, Escuela de Ciencias Básicas, Tecnología e IngenierÍa, Especialización en Seguridad Informática, Página 29.

[Ing. Pedro Julio Colorado Ángel, 2015e] Ing. Pedro Julio Colorado Ángel, I. I. J. T. B. (2015a). ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE. UNAD, Escuela de Ciencias Básicas, Tecnología e IngenierÍa, Especialización en Seguridad Informática, Página 32.

[Isabel, 2013] Isabel, D. C. S. (2013). Usabilidad en aplicaciones móviles. ICT-UNPA, Página 28.

[Álvaro Zapata, 2012a] Álvaro Zapata, Manuel Báez, M. S. (2012a). Introducción a android. Universidad Compútense de Madrid, Página 2 & 3.

[Álvaro Zapata, 2012b] Álvaro Zapata, Manuel Báez, M. S. (2012b). Introducción a android. Universidad Compútense de Madrid.

[Maira Cecilia Gasca Mantilla, 2013a] Maira Cecilia Gasca Mantilla, Luis Leonardo Camargo Ariza, B. M. D. (2013a). Metodología para el desarrollo de aplicaciones móviles. Universidad del Magdalena, Página 24.

[Maira Cecilia Gasca Mantilla, 2013b] Maira Cecilia Gasca Mantilla, Luis Leonardo Camargo Ariza, B. M. D. (2013a). Metodología para el desarrollo de aplicaciones móviles. Universidad del Magdalena, Página 25.

[Maira Cecilia Gasca Mantilla, 2013c] Maira Cecilia Gasca Mantilla, Luis Leonardo Camargo Ariza, B. M. D. (2013a). Metodología para el desarrollo de aplicaciones móviles. Universidad del Magdalena, Página 26.

[Maira Cecilia Gasca Mantilla, 2013d] Maira Cecilia Gasca Mantilla, Luis Leonardo Camargo Ariza, B. M. D. (2013a). Metodología para el desarrollo de aplicaciones móviles. Universidad del Magdalena, Página 27.

[Mazumder, 2010] Mazumder, A. (2010). Mobile application and its global impact. International Journal of Engineering & Technology IJET-IJENS Vol. 10 No. 06, Page 1.

[NetBeans, 2015a] NetBeans (2015a). Bienvenido a netbeans. https://netbeans.org/indexes.html.

[NetBeans, 2015b] NetBeans (2015b). Netbeans ide features,base ide. https://netbeans.org/features/ide/index.html.

[NetBeans, 2015c] NetBeans (2015c). Netbeans platform. https://netbeans.org/features/platform/index.html.

[Nice Agency, 2014a] nice Agency (2014a). Creating mobile applications with purpose. Nice Agency, Page 2.

[Nice Agency, 2014b] nice Agency (2014b). Creating mobile applications with purpose. Nice Agency, Page 1.

[Riehle, 2013a] Riehle, H. (2013a). Restaurant technology: Critical for tomorrow's success. National Restaurant Association, Page 39.

[Riehle, 2013b] Riehle, H. (2013b). Restaurant technology: Critical for tomorrow's success. National Restaurant Association, Page 34.

[Roopa, 2005] Roopa, T. A. . Y. (2005). Mobile computing. Technology, Applications and Service creation. McGraw Hill. 2005. ISBN-13: 978-0-07-058807-3.

[Santiago, 2007] Santiago, C. (2007). Fundamentos de sistemas operativos: teoría y ejercicios resueltos. Editorial Paraninfo.

[Soft, 2016a] Soft, N. (2016a). Soft restaurant móvil ficha técnica. versión profesional 2016.

[Soft, 2016b] Soft, N. (2016b). Soft restaurant professional, renta electronica.

[Zulma Cataldi, 2012] Zulma Cataldi, F. J. L. (2012). Entornos de aprendizaje personalizados en dispositivos móviles. Universidad de Buenos Aires, Facultad de Ingeniería.