
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

“MONITOREO Y VALIDACIÓN DE BASES DE DATOS EN DJANGO”

TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

PARA OBTENER EL TÍTULO PROFESIONAL DE:
INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES

PRESENTA:
CARLOS ALBERTO SÁNCHEZ GÓMEZ

NOMBRE DEL ASESOR:
MTRA. CLAUDIA SAAVEDRA HERNÁNDEZ



Instituto Tecnológico de Apizaco
Sistemas y Computación

Apizaco, Tlaxcala, **05/septiembre/2025**

Oficio No. S.C./OPV/0513/25

ASUNTO: **Liberación de proyecto
para titulación integral**

ISRAEL JORGE SÁNCHEZ SÁNCHEZ
JEFE DEL DEPTO. DE DIVISIÓN DE ESTUDIOS PROFESIONALES
P R E S E N T E.

POR ESTE MEDIO LE INFORMO QUE HA SIDO LIBERADO EL SIGUIENTE PROYECTO PARA LA TITULACIÓN INTEGRAL.

NOMBRE DEL EGRESADO:	SÁNCHEZ GÓMEZ CARLOS ALBERTO
CARRERA:	INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
NO. DE CONTROL:	20370972
NOMBRE DEL PROYECTO:	"Monitoreo y Validación de Bases de Datos en Django"
PRODUCTO U OPCIÓN:	TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

AGRADEZCO DE ANTEMANO SU VALIOSO APOYO EN ESTA IMPORTANTE ACTIVIDAD PARA LA FORMACIÓN PROFESIONAL DE NUESTROS EGRESADOS.

ATENTAMENTE

*Excelencia en Educación Tecnológica.
Pensar para Servir, Servir para Triunfar®*

MIQUELINA SÁNCHEZ PULIDO
JEFA DEL DEPTO. DE SISTEMAS Y COMPUTACIÓN



CLAUDIA SAAVEDRA HERNÁNDEZ
ASESOR INTERNO

ccp. Coordinación de carrera
Expediente
MSP/mrol*



2025
Año de
**La Mujer
Indígena**

Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec,
Municipio de Tzompantepec, Tlaxcala, Mex. C.P. 90491
Tel. (241) 4172010 Ext. 115, e-mail: sistemas@apizaco.tecnm.mx
tecnm.mx | apizaco.tecnm.mx





Instituto Tecnológico de Apizaco
División de Estudios Profesionales

Tzompantepec, Tlaxcala **29/septiembre/2025**

**SÁNCHEZ GÓMEZ CARLOS ALBERTO
EGRESADO(A) DE LA CARRERA DE
INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN
Y COMUNICACIONES.
CON NÚMERO DE CONTROL 20370972.
P R E S E N T E.**

Por este medio, me permito informar a usted que por aprobación de la comisión revisora asignada para valorar el trabajo que mediante la opción **TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL** y cuyo tema es: **"MONITOREO Y VALIDACIÓN DE BASES DE DATOS EN DJANGO"**, conforme a lo establecido en el procedimiento para la obtención del Título Profesional en el Instituto Tecnológico, la División de Estudios Profesionales a mi cargo le otorga la:

AUTORIZACIÓN DE IMPRESIÓN

Debiendo entregar **2 USB** del mismo, apropiadamente empastados, a la brevedad, para presentar su Acto de Recepción Profesional.

Sin otro particular por el momento me despido.

ATENTAMENTE

*Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®*

**CECILIA YARELY ARREGUÍN LARA
JEFA DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES**

ccp. Archivo

CYAL/rrc



SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DIVISIÓN DE
ESTUDIOS PROFESIONALES



2025
Año de
**La Mujer
Indígena**

Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec,
Municipio de Tzompantepec, Tlaxcala, Mex. C.P. 90491 Tel. [241] 4172010
Ext. 304, e-mail: dep_titulacion@apizaco.tecnm.mx | apizaco.tecnm.mx



Agradecimientos

A lo largo de esta etapa tan significativa en mi formación profesional, he tenido la dicha de contar con el apoyo de personas valiosas que, con su presencia, consejos y cariño, hicieron posible la realización de este proyecto.

En primer lugar, quiero agradecer con todo mi corazón a **mi madre**, por su amor incondicional, por ser mi mayor ejemplo de fortaleza, y por impulsarme en cada momento difícil. Gracias por creer en mí incluso cuando yo mismo dudaba, por tus palabras sabias y por tu apoyo incansable.

Agradezco profundamente a **la ingeniera Susana**, mi asesora externa dentro del Consejo de la Judicatura Federal, por brindarme la oportunidad de colaborar y aprender en un entorno profesional tan enriquecedor. Su guía, paciencia y compromiso fueron esenciales para el desarrollo de este proyecto.

A la **maestra Claudia**, mi asesora interna en el Instituto Tecnológico de Apizaco, gracias por su acompañamiento constante, por su disposición y por motivarme a seguir adelante. Su experiencia académica marcó una diferencia clave en mi camino.

No puedo dejar de mencionar a alguien muy especial para mí: **Peabody**, mi perrita beagle. A su manera, siempre estuvo ahí, acompañándome en las largas noches de trabajo, dándome paz y alegría con su sola presencia. Gracias por hacer mis días más ligeros y mi hogar más cálido.

Finalmente, y no menos importante, a ti **Julissa**, mi compañera de vida y de sueños. Gracias por tu compañía, por tu paciencia y amor infinito, y por estar conmigo en cada momento. Gracias por las palabras de ánimo, por las sonrisas en los momentos de estrés, y por enseñarme a ser mejor cada día. No hay proyecto más importante que el que estamos construyendo juntos. Te amo profundamente, y sé que la vida aún tiene muchos capítulos hermosos reservados para nosotros.

A todos ustedes: gracias. Este logro no es solo mío, es también suyo.

- Carlos Sánchez

Índice

Agradecimientos	2
Índice.....	3
Resumen.....	7
1. Introducción	8
2. Generalidades.....	9
2.1 Ubicación	9
2.2 Logotipo de la empresa.....	10
2.3 Consejo de la Judicatura Federal, Poder Judicial de la Federación, Edificio Sede.	10
3. Misión	11
4. Visión.....	11
5. Objetivo.....	11
6. Valores	12
7. Problemas que resolver	12
8. Objetivos	13
8.1 Objetivo General.....	13
8.2 Objetivos Específicos.....	13
9. Justificación	14
10. Marco Teórico.....	15

10.1 Framework Django	15
10.2 Bases de datos y validación de datos	16
10.3 Celery y procesamiento asíncrono de tareas	17
10.4 Automatización con Selenium IDE	18
10.5 Metodología ágil: Scrum.....	19
10.6 Seguridad de datos y cumplimiento legal	20
10.6.1 Cifrado de datos y estándares internacionales	20
10.6.2 Marco legal: LGPDPPSO, CNPP y normativas del CJF	21
10.7. Ética en el desarrollo de software	22
10.7.1. Principios y marcos.....	23
10.7.2. Propuestas para su inclusión	23
10.8 Experiencia de usuario (UX) y accesibilidad digital	24
11. Diseño del proyecto	24
11.1 Diagrama entidad-relación.....	24
11.2 Diagrama relacional (cardinalidad y tablas)	28
11.3 Diagrama de clases	29
12. Mockups del proyecto.....	31
13. Creación de un proyecto en Django.....	35
14. Arquitectura del sistema	42
14.1. Inicio del sistema	42

14.2. Login del sistema	43
14.3. Panel de monitoreo	45
14.4. Listado de formularios	47
15. Metodología	53
15.1 Enfoque metodológico	53
15.2 Diseño del sistema de monitoreo y validación	54
15.3. Herramientas tecnológicas	55
15.3.1. Python	55
15.3.2. Celery	55
15.3.3. Git y GitHub	56
15.4. Proceso de implementación	56
15.5. Validación de resultados	57
15.5.1. Tiempo de respuesta de tareas programadas.....	57
15.5.2. Porcentaje de validaciones exitosas en diferentes escenarios.....	60
15.5.3. Cantidad de inconsistencias detectadas y corregidas.....	63
15.5.4. Evaluación de satisfacción por parte de usuarios técnicos o administradores.....	66
16. Resultados Esperados.....	68
16.1. Mejora en la integridad y consistencia de los datos.....	69
16.2. Automatización del monitoreo y reducción de tareas manuales.....	69
16.3. Reportes y alertas oportunas	70

16.4. Validación funcional de interfaces con Selenium IDE	70
16.5. Fortalecimiento de la cultura de calidad y buenas prácticas.....	71
17. Conclusiones	71
18. Recomendaciones	73
Referencias.....	75

Resumen

El proyecto elaborado surge de la necesidad de crear e innovar procesos formales y automatizados de monitoreo y validación de bases de datos dentro del Consejo de la Judicatura Federal (CJF), siendo en el Edificio Sede de Apizaco, Tlaxcala donde se realiza dicho proyecto. En base a esto, es fundamental tener una supervisión de estos procesos de manera continua y validar de manera estructurada la información, debido a que de no ser así pueden surgir errores operativos, pérdidas de información y comprometer la integridad de los registros. Por tanto, la propuesta es diseñar e implementar un sistema integral utilizando el framework Django, que permita supervisar de manera continua el estado de la base de datos, detectar posibles fallos de conexión o de sistema, y validar que los datos cumplan con las reglas de negocio y normativas legales aplicables.

Se optó por el uso del ORM de Django debido a la facilidad y robustez del mismo, de igual manera se hace el uso de tareas programadas mediante Celery, incluye validadores, pruebas automatizadas y generación de reportes. El sistema actual también contempla cifrado de datos para elementos sensibles, respaldos automáticos de la información y el cumplimiento normativo, alineándose con la Ley General de Protección de Datos Personales (LGPD), el Código Nacional de Procedimientos Penales (CNPP) y la norma ISO 27001. Con la implementación se busca mejorar la calidad de los datos, garantizar su integridad y promover la transparencia y seguridad jurídica dentro del entorno judicial federal.

1. Introducción

En los últimos años, el uso de sistemas digitales dentro del Poder Judicial ha ido creciendo de forma constante, especialmente en lo que respecta al manejo de información judicial y administrativa. En este proceso, las bases de datos juegan un papel esencial, ya que almacenan gran parte de los registros que respaldan las operaciones internas y la toma de decisiones. Sin embargo, este crecimiento también ha traído consigo nuevos retos. Uno de los principales problemas detectados en los entornos actuales de desarrollo, particularmente en sistemas contruidos con Django, es la ausencia de mecanismos que permitan monitorear y validar de forma automatizada el estado de las bases de datos.

Dentro del edificio sede del CJF en Apizaco, Tlaxcala, se ha identificado que las aplicaciones en producción enfrentan limitaciones cuando se trata de errores en los datos o fallas en el sistema que afecten el acceso a la información. Además, muchas veces la supervisión depende de revisiones manuales, lo que puede retrasar la identificación de problemas y, en casos más delicados, comprometer la integridad de los datos judiciales. Sumado a esto, no existen procesos estandarizados de validación que aseguren que los datos almacenados cumplan con las reglas de negocio, estructuras y restricciones necesarias para su correcto funcionamiento y para el cumplimiento de obligaciones legales.

Dado este escenario, se vuelve necesario desarrollar herramientas que permitan realizar estas tareas de forma automatizada, disminuyendo así los errores humanos y fortaleciendo la seguridad de los sistemas institucionales. Es aquí donde surge la oportunidad de diseñar un sistema que no solo monitoree el estado operativo de las bases de datos, sino que también valide y respalde la calidad

de los registros. Este proyecto se enmarca en el programa de residencias profesionales del Instituto Tecnológico de Apizaco, con el objetivo de generar una solución concreta a una necesidad real en un contexto institucional.

2. Generalidades

2.1 Ubicación

PREDIO RÚSTICO, SANTA ANITA HUILOAC, Cdad. de Apizaco, Tlax.

CP: 90407

Tel: [241 418 8190](tel:2414188190)

Correo: sequintanaib@cjf.gob.mx

Web: www.cjf.gob.mx

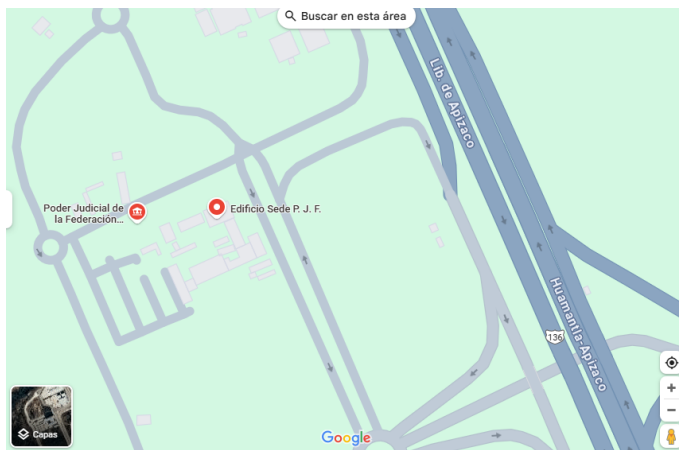


Ilustración 1. Ubicación

2.2 Logotipo de la empresa

Consejo de la Judicatura Federal, Poder Judicial de la Federación, Edificio Sede.



Ilustración 2. Logotipo CJF

2.3 Consejo de la Judicatura Federal, Poder Judicial de la Federación, Edificio Sede.

El **Consejo de la Judicatura Federal (CJF)** es el órgano encargado de la administración, vigilancia y disciplina del Poder Judicial de la Federación en México. Su misión principal es garantizar la eficiencia y probidad en la impartición de justicia a nivel federal. En este entorno, se llevan a cabo actividades relacionadas con la gestión judicial, desarrollo de políticas administrativas y supervisión de órganos jurisdiccionales en todo el país. Además, el edificio es sede de reuniones, capacitaciones y eventos institucionales que buscan fortalecer el sistema de justicia federal en México.

3. Misión

Garantizar la administración, vigilancia, disciplina y carrera judicial, que permitan el funcionamiento de Juzgados de Distrito y Tribunales de Circuito y aseguren su autonomía, así como la objetividad, honestidad, profesionalismo e independencia de sus integrantes, a fin de coadyuvar a que la sociedad reciba justicia pronta, completa, gratuita e imparcial.

4. Visión

Ser reconocido como un órgano de administración confiable, transparente y de excelencia, que asegura medios y elementos de calidad en la impartición de justicia, dando certeza en la atención de los servicios que brinda tanto a los Tribunales y Juzgados Federales como a la sociedad, en un ámbito de eficiencia y eficacia, y que contribuye a lograr una justicia que da respuesta y garantiza la seguridad jurídica de los gobernados.

5. Objetivo

Garantizar la eficiencia, transparencia, independencia e imparcialidad en la impartición de justicia a nivel federal, asegurando que los órganos judiciales cumplan con su función de manera adecuada y conforme a la ley.

6. Valores

- Independencia
- Imparcialidad
- Transparencia
- Ética
- Eficiencia
- Rendición de cuentas
- Vocación de servicio
- Legalidad

7. Problemas que resolver

Actualmente, en el manejo de bases de datos de las aplicaciones desarrolladas en Django, se presentan diversas áreas de oportunidad que requieren atención. Una de ellas es la falta de un monitoreo continuo que permita detectar de manera temprana fallas en la conexión, caídas del servicio o inconsistencias en los datos almacenados. Esta situación puede derivar en errores operativos, pérdidas de información y afectaciones en la calidad del servicio ofrecido.

Asimismo, no existen procesos formalizados de validación que aseguren que los datos cumplan con las reglas de negocio y restricciones de integridad definidas, lo que incrementa el riesgo de registrar información errónea o incompleta. La ausencia de sistemas automatizados para la generación de reportes y alertas también limita la capacidad de respuesta ante incidentes críticos, haciendo que la supervisión de bases de datos dependa en gran medida de revisiones manuales, las cuales son poco eficientes y propensas a errores.

Finalmente, no se cuenta con protocolos claros para validar datos durante procesos de actualización o migración, lo que puede comprometer la estabilidad y confiabilidad de los sistemas. Estas problemáticas hacen necesaria la implementación de estrategias que garanticen un monitoreo efectivo y una validación adecuada de las bases de datos.

8. Objetivos

8.1 Objetivo General

Diseñar e implementar un sistema integral de monitoreo y validación de bases de datos para aplicaciones desarrolladas con el framework Django, que permita detectar de manera oportuna fallas de conexión, errores de integridad o inconsistencias en los registros, y aplicar correcciones automatizadas que aseguren la calidad, confiabilidad y legalidad de los datos manejados por el CJF en el Edificio Sede de Apizaco.

8.2 Objetivos Específicos

- Diseñar un módulo de monitoreo que revise constantemente la disponibilidad, respuesta y estabilidad de las bases de datos.
- Integrar procesos de validación personalizados que garanticen que los datos registrados cumplan con las reglas de negocio y estructuras definidas.

- Generar reportes periódicos y auditables que permitan detectar patrones de fallos o anomalías en los datos.
- Implementar sistemas de respaldo y cifrado que aseguren la disponibilidad y protección de la información sensible.
- Documentar y establecer lineamientos técnicos que faciliten la gestión, mantenimiento y validación de las bases de datos en entornos productivos.

9. Justificación

La razón principal para llevar a cabo este proyecto es mejorar la calidad de los datos y la confiabilidad de los sistemas informáticos dentro del Consejo de la Judicatura Federal, específicamente en su sede de Apizaco. En instituciones que manejan información judicial y administrativa sensible, como el CJF, el simple hecho de que un dato esté mal registrado, duplicado o perdido puede tener repercusiones importantes, tanto en lo operativo como en lo legal. Por ello, contar con herramientas que permitan validar y monitorear constantemente las bases de datos no es solo una mejora técnica, sino una necesidad institucional.

Además, este proyecto contribuye directamente al cumplimiento de normativas que regulan la protección de datos personales, la transparencia y la trazabilidad en la gestión pública. Implementar validaciones automáticas, cifrado de datos sensibles, y reportes que puedan auditarse, no solo brinda mayor seguridad jurídica, sino que también fortalece la confianza de los usuarios internos y de la ciudadanía en los sistemas del Poder Judicial.

Desde un enfoque más operativo, el sistema propuesto también busca reducir el esfuerzo humano requerido para supervisar manualmente la integridad de los datos. Al automatizar tareas repetitivas como validaciones, generación de reportes y monitoreo de conexiones, se libera tiempo y recursos que pueden enfocarse en tareas de mayor valor estratégico. Finalmente, la documentación y estandarización de estas herramientas dejará una base sólida que podrá mantenerse, actualizarse y adaptarse a futuros sistemas dentro del CJF, contribuyendo así al fortalecimiento institucional.

10. Marco Teórico

10.1 Framework Django

Django es un framework de desarrollo web de alto nivel escrito en Python, diseñado para fomentar el desarrollo rápido y limpio de aplicaciones web robustas. Este entorno destaca por su arquitectura Modelo–Vista–Plantilla (MVT), que separa la lógica de negocio, la interfaz y la manipulación de datos, lo que contribuye a un código más estructurado y mantenible (Holovaty & Kaplan-Moss, 2009).

Uno de los componentes más relevantes de Django para este proyecto es su sistema de validación de datos. Al definir modelos, el desarrollador puede especificar reglas como longitudes máximas, formatos, unicidad, relaciones entre tablas, e incluso validaciones personalizadas utilizando funciones `clean()` o señales como `pre_save`. Esto garantiza que los datos insertados en la base de datos respeten tanto restricciones técnicas como reglas de negocio específicas (Python Software Foundation, 2024).

Dado que Django trabaja con un ORM (Object-Relational Mapping), es posible construir consultas complejas y realizar validaciones sin escribir directamente SQL, lo que reduce errores y mejora la eficiencia del desarrollo. Esta capacidad resulta especialmente útil en contextos institucionales como el Poder Judicial Federal, donde la consistencia y exactitud de la información son esenciales para la toma de decisiones administrativas o legales.

Como señala Cifuentes et al. (2021), el uso de frameworks estructurados permite reducir significativamente los errores de desarrollo y aumentar la calidad de los sistemas al imponer un control estricto sobre la entrada de datos. Esto es especialmente importante en sistemas jurídicos donde una simple inconsistencia puede tener consecuencias legales.

10.2 Bases de datos y validación de datos

Las bases de datos relacionales son estructuras fundamentales para el almacenamiento ordenado, seguro y accesible de grandes volúmenes de información. En aplicaciones desarrolladas con Django, es común utilizar motores como PostgreSQL o MySQL, que permiten definir esquemas complejos y relaciones entre datos (Cattell, 2011). Estas bases permiten establecer restricciones de integridad, claves foráneas, índices y mecanismos de replicación que aseguran la coherencia entre los registros.

La validación de datos es un componente esencial en cualquier sistema, ya que permite verificar que la información introducida sea precisa, completa y conforme a las reglas del negocio. En contextos judiciales, esto toma especial relevancia, ya que la falta de exactitud puede comprometer

la legalidad de un proceso. Según lo establece la Ley General de Protección de Datos Personales en Posesión de Sujetos Obligados (2021), los datos deben ser exactos, completos, pertinentes, correctos y actualizados (art. 50).

Dentro del ecosistema Django, esta validación puede realizarse en diferentes capas: en formularios, en vistas, en serializadores o directamente en los modelos. Cada capa permite detectar diferentes tipos de errores. Por ejemplo, es posible evitar que se registren expedientes con fechas fuera de rango o que un tribunal se relacione con un código no válido. Esta validación puede complementarse con mecanismos automatizados como tareas periódicas, scripts de revisión y generación de reportes (Python Software Foundation, 2024).

Adicionalmente, el sistema debe incorporar políticas de respaldo automatizado y trazabilidad. Estas funciones aseguran la recuperación de información en caso de pérdida de datos o fallos del sistema, aspectos alineados con normativas como la ISO/IEC 27001 sobre seguridad de la información (Organización Internacional de Normalización, 2023).

10.3 Celery y procesamiento asíncrono de tareas

En el desarrollo de sistemas web modernos, muchas operaciones no pueden ejecutarse en tiempo real debido a su complejidad, duración o impacto en el rendimiento general de la aplicación. Celery es una herramienta de Python que permite ejecutar tareas de manera asíncrona y distribuida, lo cual es especialmente útil para tareas programadas o de procesamiento intensivo, como el envío de correos masivos, generación de reportes o validación periódica de datos.

En el contexto de este proyecto, Celery resulta clave para implementar rutinas de monitoreo automático de bases de datos, ya que permite definir trabajos que se ejecutan en segundo plano sin interferir con la experiencia del usuario. Estas tareas pueden programarse con intervalos definidos —por ejemplo, revisar la integridad de registros cada noche— o ejecutarse ante ciertos eventos, como una modificación masiva de información sensible.

Celery se integra fácilmente con Django mediante paquetes como `django-celery-beat`, y puede coordinarse con servicios como Redis o RabbitMQ para gestionar colas de trabajo de manera eficiente (Ask Solem, 2014). Su arquitectura modular y escalable permite distribuir la carga de trabajo entre distintos servidores, asegurando que incluso sistemas complejos puedan mantenerse reactivos y robustos ante altos volúmenes de datos o usuarios.

La implementación de Celery en entornos judiciales o administrativos mejora la eficiencia operativa y reduce el riesgo de errores humanos, al automatizar tareas críticas con plena trazabilidad y control (Patel et al., 2023).

10.4 Automatización con Selenium IDE

Selenium IDE es una herramienta de automatización de pruebas para aplicaciones web que permite grabar, editar y reproducir interacciones del usuario con una página. A diferencia de otros marcos de pruebas más complejos, Selenium IDE está pensado para facilitar la creación de scripts sin requerir conocimientos avanzados de programación, lo cual lo convierte en una excelente opción para pruebas funcionales básicas o validaciones de interfaz (Barrett, 2008).

En este proyecto, Selenium se considera como una posible herramienta para validar el funcionamiento de formularios y vistas dentro de las aplicaciones Django, asegurando que los datos ingresados a través de la interfaz gráfica cumplan con las reglas de negocio desde la capa visual. Esto es especialmente útil cuando se busca realizar pruebas automatizadas previas a despliegues en producción o detectar errores introducidos por cambios recientes.

Además, Selenium puede integrarse con otros entornos como Jenkins o GitHub Actions para ejecutar pruebas de forma continua (CI/CD), lo cual aporta valor al mantener una calidad constante del software sin depender exclusivamente de la validación manual.

Como se ha documentado en estudios recientes sobre pruebas automatizadas, el uso de herramientas como Selenium permite reducir hasta un 30 % el tiempo invertido en pruebas repetitivas, y mejora la cobertura de errores detectados durante las fases tempranas del desarrollo (Brunet et al., 2024).

10.5 Metodología ágil: Scrum

El desarrollo ágil ha revolucionado la forma en que se abordan los proyectos tecnológicos, especialmente en contextos donde las necesidades pueden cambiar rápidamente o donde es indispensable obtener resultados funcionales en plazos cortos. Dentro de las metodologías ágiles, Scrum ha emergido como una de las más utilizadas por su estructura iterativa, basada en ciclos cortos de trabajo llamados sprints, roles definidos y una comunicación constante entre los miembros del equipo (Schwaber & Sutherland, 2020).

Scrum promueve entregas frecuentes, evaluación continua del progreso y adaptación flexible ante nuevos requerimientos. En este proyecto, su implementación permite gestionar las distintas fases del desarrollo del sistema de monitoreo, priorizando tareas como la validación de datos, la configuración de Celery o la implementación de reportes automáticos.

El uso de Scrum en un contexto institucional como el Edificio Sede de Apizaco facilita una mejor organización del trabajo del residente, al permitir dividir el proyecto en entregables concretos y medibles. Además, al promover reuniones periódicas (como dailies o retrospectivas), se garantiza un seguimiento constante del avance y una detección oportuna de obstáculos.

Según Fernández y Gómez (2016), la aplicación de metodologías ágiles como Scrum no solo incrementa la productividad, sino que mejora la calidad del producto final y la satisfacción de los usuarios, al permitir una participación más activa en el proceso de desarrollo.

10.6 Seguridad de datos y cumplimiento legal

10.6.1 Cifrado de datos y estándares internacionales

El tratamiento adecuado de los datos es un aspecto esencial en cualquier sistema informático moderno, pero cobra particular importancia cuando se trata de información judicial o administrativa. Una violación de la seguridad no solo implica pérdidas técnicas o económicas, sino que puede comprometer derechos fundamentales, como la privacidad de las personas.

Por ello, el presente proyecto considera prácticas de cifrado, control de acceso y respaldo periódico, en concordancia con estándares internacionales como ISO/IEC 27001, que define los requisitos para establecer, implementar y mantener sistemas de gestión de seguridad de la información (ISO, 2013).

El cifrado de contraseñas, la validación de tokens, la autenticación multifactor y el uso de HTTPS para todas las conexiones son prácticas que se consideran mínimas en el desarrollo de aplicaciones Django seguras (Pinkham, 2022). A su vez, el control de accesos mediante middleware y decoradores personalizados dentro del framework permite garantizar que cada usuario acceda únicamente a la información que le corresponde.

Estas medidas técnicas no son opcionales, sino que forman parte del cumplimiento de un marco legal robusto que protege los datos personales y sensibles de quienes interactúan con el sistema.

10.6.2 Marco legal: LGPDPPSO, CNPP y normativas del CJF

En México, el tratamiento de los datos personales está regulado por la Ley General de Protección de Datos Personales en Posesión de Sujetos Obligados (LGPDPPSO), la cual establece lineamientos para la recolección, uso, almacenamiento y transferencia de información por parte de entidades gubernamentales. Esta ley obliga a que todo sistema que maneje datos personales — como los relacionados con expedientes judiciales, personal administrativo o ciudadanos— cuente con medidas técnicas y organizativas suficientes para su resguardo (INAI, 2017).

Adicionalmente, el Código Nacional de Procedimientos Penales (CNPP) establece principios como la confidencialidad y la cadena de custodia digital, lo cual refuerza la necesidad de contar con mecanismos de integridad y trazabilidad en el almacenamiento y acceso a bases de datos judiciales.

El Consejo de la Judicatura Federal (CJF), a través de sus lineamientos internos y políticas de tecnologías de la información, también promueve estándares de calidad, seguridad y transparencia. Estos lineamientos buscan garantizar que las herramientas digitales utilizadas en el ámbito judicial no solo sean funcionales, sino que protejan adecuadamente los derechos de los gobernados.

En este sentido, la implementación de un sistema de monitoreo y validación de bases de datos no solo responde a una necesidad técnica, sino también a un imperativo legal que asegura el correcto manejo de la información dentro del sistema judicial.

10.7. Ética en el desarrollo de software

Desarrollar software no es solo convertir requisitos en código: es un acto que impacta en personas, decisiones y confianza social. El desarrollador no es un técnico aislado, sino un actor ético con responsabilidad. La forma en que se captura, protege y procesan datos determina si una aplicación fortalece o vulnera derechos humanos.

El compromiso ético responde a la necesidad de proteger la privacidad, garantizar la seguridad y prevenir sesgos discriminatorios. Además, fomenta la confianza de usuarios e instituciones en el sistema. Ignorar estos aspectos puede generar consecuencias graves: vulneraciones legales, desconfianza pública y daños a las personas involucradas (AlgoCademy, 2024).

10.7.1. Principios y marcos

Privacidad por diseño (“privacy by design”), anticipa y previene riesgos desde el inicio, manteniendo la protección de datos como configuración predeterminada (Cavoukian, 2011).

Fairness y no discriminación, implica auditar datos y algoritmos para evitar resultados sesgados hacia grupos vulnerables (AlgoCademy, 2024).

Transparencia y rendición de cuentas, comunicar a usuarios qué datos se recogen, cómo se usan y quién decide qué. Remoción de “cajas negras” y establecimiento de responsables claros (AlgoCademy, 2024).

Aplicar análisis ético durante requisitos, diseño, implementación y pruebas, no sólo políticas internas. Fomentar debates dentro del equipo para resolver dilemas éticos, más allá de seguir códigos de conducta reductivos (Gogoll et al., 2020).

Diseñar sistemas explicables, que los desarrolladores puedan razonar y justificar sus decisiones (Chazette et al., 2022).

10.7.2. Propuestas para su inclusión

Capacitación continua sobre ética, privacidad y sesgos.

Definir roles en el equipo: responsables éticos, de privacidad y seguridad.

Crear protocolos de revisión ética y monitoreo de sesgos tras el despliegue.

10.8 Experiencia de usuario (UX) y accesibilidad digital

Cada interacción digital es una experiencia humana. Ignorar la diversidad de capacidades, contextos y necesidades implica excluir a miles de personas. UX inclusiva no es un requisito técnico, es un compromiso social y moral.

Actualmente, alrededor del 20 % de la población tiene alguna discapacidad relevante para el uso de software (Qadan, 2024). Diseñar pensando en accesibilidad:

- Fomenta la inclusión social y laboral.
- Cumple con normativas legales vigentes como WCAG y ADA (Globant, 2025).
- Mejora la confianza de marca, UX general y SEO (Shukla, 2024).

Solo el 3.7 % de las páginas principales son totalmente accesibles (Parthasarathy & Joshi, 2023).

Las interfaces accesibles elevan la empatía y la percepción de calidad, la accesibilidad técnica no siempre garantiza una buena experiencia (usuarios reales, lectura fácil): se requieren pruebas con usuarios reales, lenguaje simple y claro, jerarquía semántica.

11. Diseño del proyecto

11.1 Diagrama entidad-relación

El diagrama entidad-relación fue diseñado para representar de forma clara y visual la estructura lógica de la base de datos del sistema, enfocándose en las entidades principales del modelo de datos y cómo se relacionan entre sí. Las entidades incluyen:

- Administrador: Encargado de gestionar, registrar y dar seguimiento a usuarios, dispositivos, incidentes y solicitudes.
- Usuario: Representa a los operadores o trabajadores del CJF que reportan incidentes, hacen solicitudes o tienen dispositivos asignados.
- Incidente, solicitud y dispositivo: Representan los módulos funcionales del sistema. Cada uno está directamente relacionado con un usuario (quien lo origina) y con un administrador (quien lo atiende o gestiona).

Cada entidad tiene atributos clave que describen su naturaleza: por ejemplo, en "Incidente" se registra la fecha de reporte, una descripción, acciones realizadas, etc.

Las relaciones entre las entidades están definidas con su respectiva cardinalidad, lo que indica cuántas instancias de una entidad pueden relacionarse con otra.

Este diagrama permite visualizar de forma estructurada cómo fluye la información y cómo se conectan las piezas fundamentales del sistema.

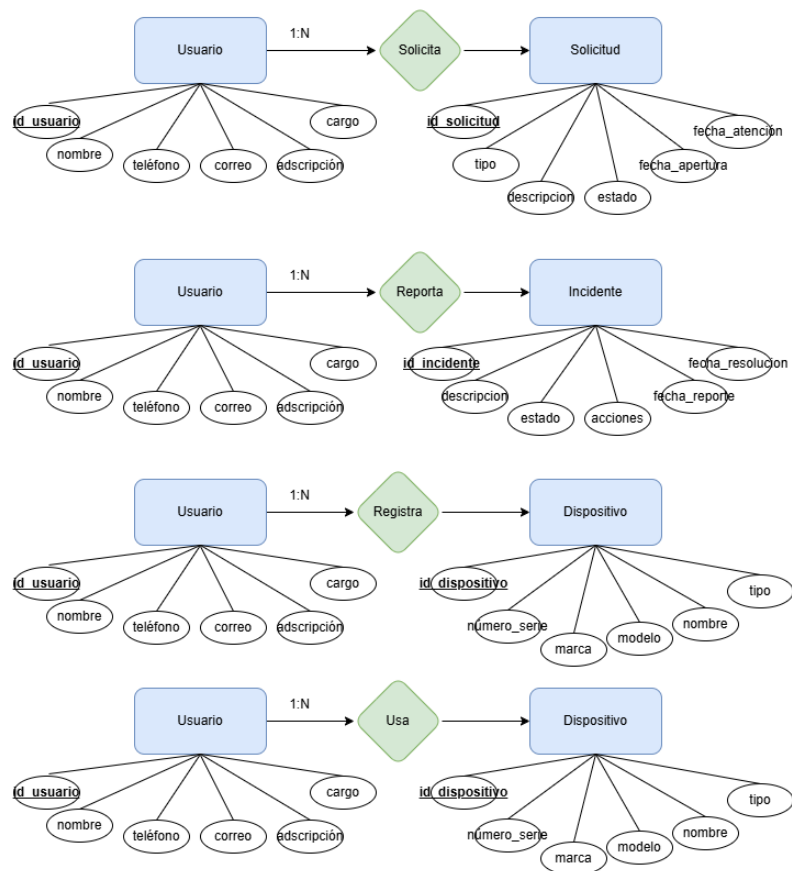


Ilustración 3. Relaciones del atributo usuario

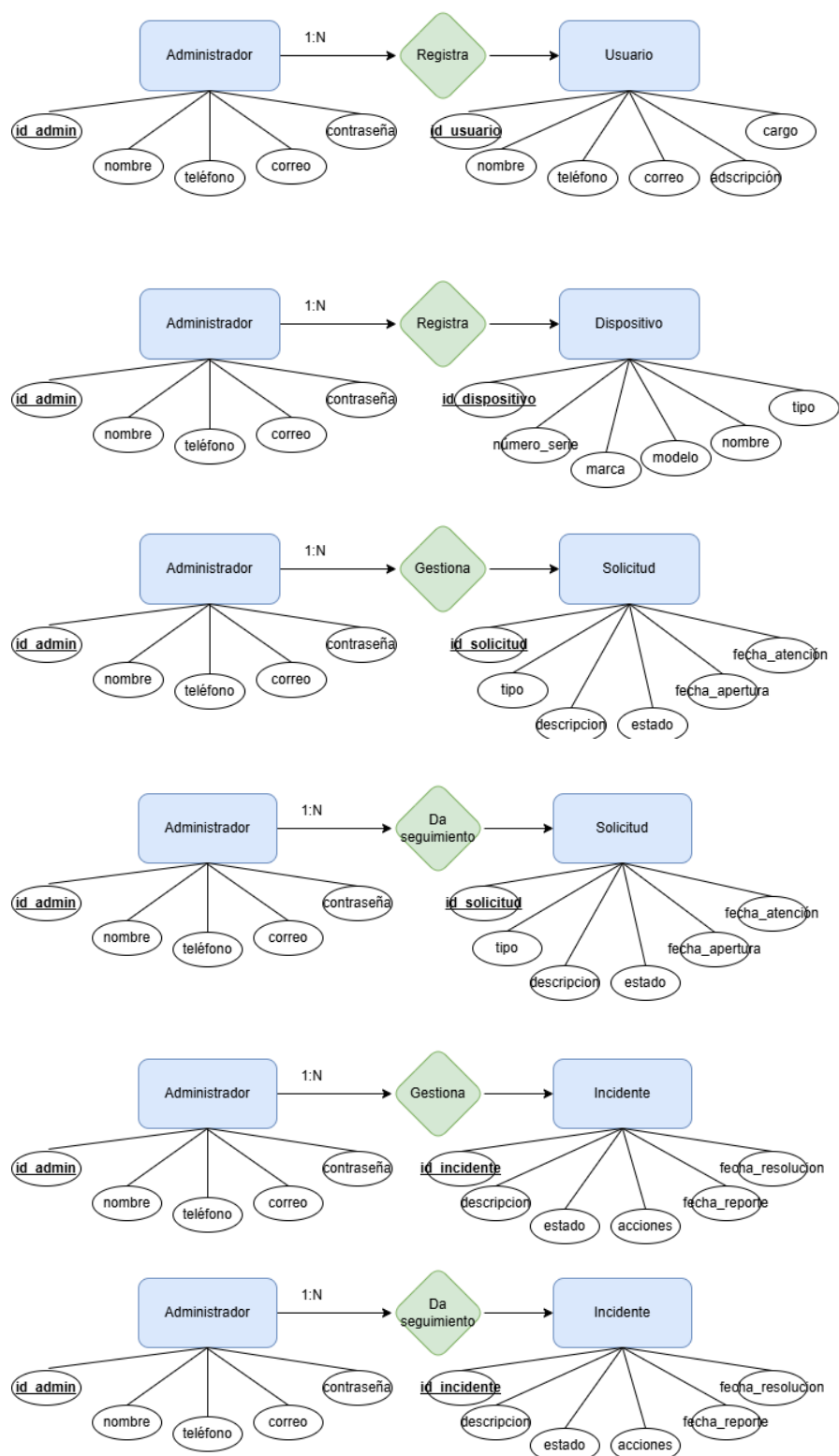


Ilustración 4. Relaciones del atributo administrador

11.2 Diagrama relacional (cardinalidad y tablas)

El diagrama relacional describe las relaciones lógicas y cardinalidades entre las tablas (entidades) del sistema. Va un paso más allá del E-R al centrarse en detalles técnicos como:

- Claves primarias y foráneas.
- Tipo de relación (uno a muchos, muchos a muchos).
- Cardinalidad exacta (por ejemplo, 1:N o N:M).

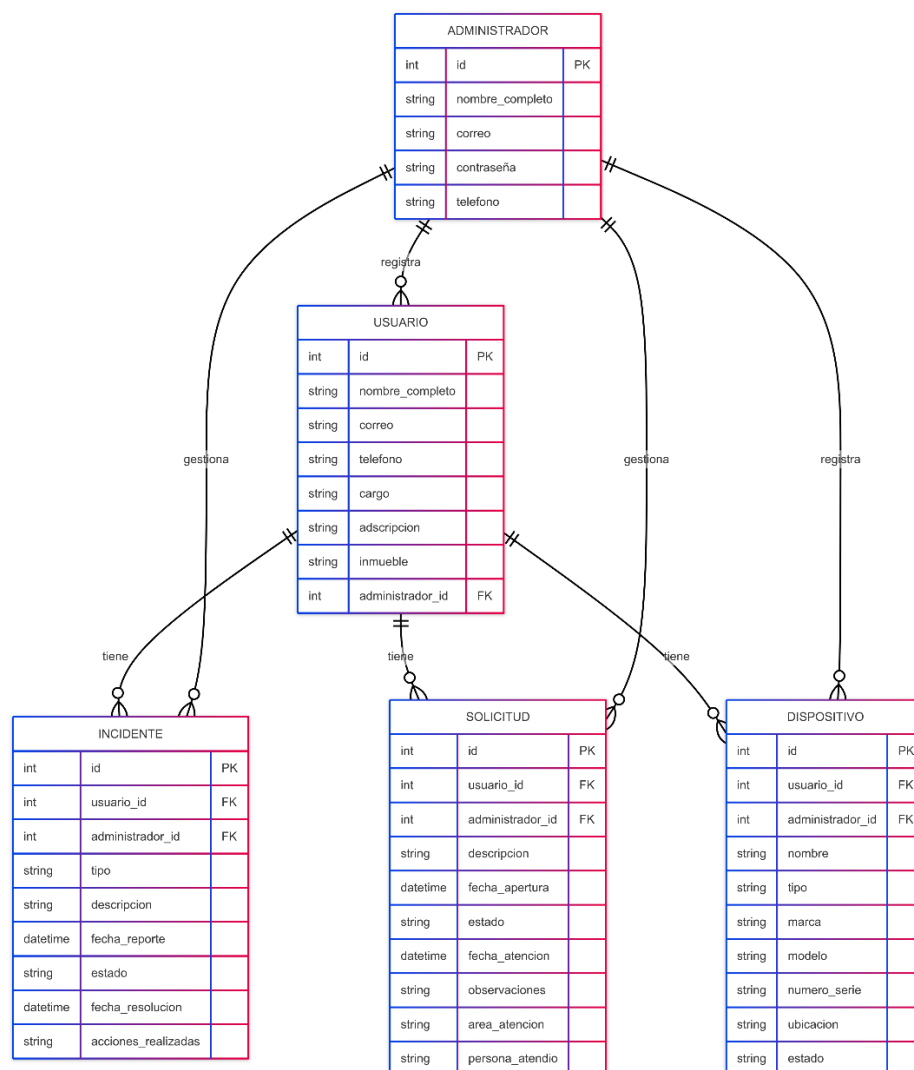


Ilustración 5. Representación de las PK y FK con sus atributos

Tabla 1. Relaciones y cardinalidades

#	Tabla destino	Clave foránea	Referencia a	Relación
1	INCIDENTE	usuario_id	USUARIO(id)	Muchos a Uno
2	INCIDENTE	administrador_id	ADMINISTRADOR(id)	Muchos a Uno
3	SOLICITUD	usuario_id	USUARIO(id)	Muchos a Uno
4	SOLICITUD	administrador_id	ADMINISTRADOR(id)	Muchos a Uno
5	DISPOSITIVO	usuario_id	USUARIO(id)	Muchos a Uno
6	DISPOSITIVO	administrador_id	ADMINISTRADOR(id)	Muchos a Uno
7	USUARIO	administrador_id	ADMINISTRADOR(id)	Muchos a Uno

11.3 Diagrama de clases

El diagrama de clases refleja la estructura de los modelos definidos en Django, es decir, cómo se organiza el código desde la lógica de programación orientada a objetos.

Cada clase representa una tabla de la base de datos, pero también define comportamientos (métodos) y relaciones entre objetos.

Se incluyen las clases Usuario, Administrador, Incidente, Solicitud y Dispositivo. Cada una define atributos relevantes (por ejemplo, nombre, correo, estado, fecha_apertura) que corresponden a campos de la base de datos.

Se especifican relaciones mediante referencias (ForeignKey), lo que se visualiza con flechas indicando "uno a muchos".

También se incluye el método `__str__()` en cada clase, el cual permite visualizar el objeto de forma legible en la administración de Django.

Este diagrama es clave para comprender cómo interactúan las clases dentro del sistema y cómo está estructurado el backend. Sirve como puente entre la lógica del sistema y su implementación en código Python.

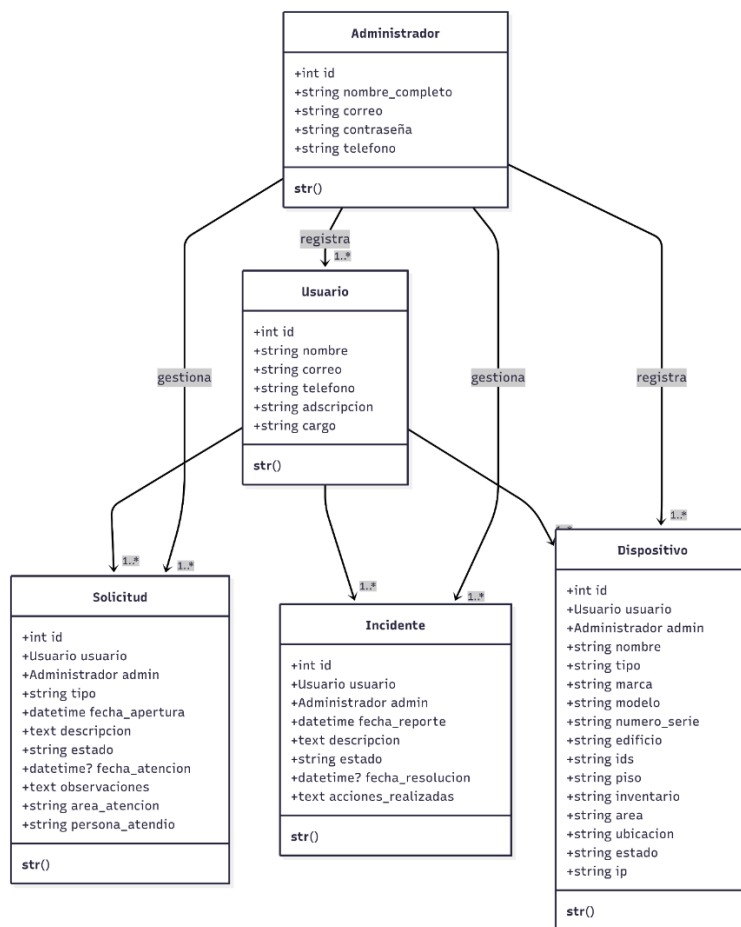


Ilustración 6. Diagrama desde la lógica de POO

12. Mockups del proyecto

Un mockup es una representación visual detallada que refleja el diseño final de un producto digital, como un sitio web o una aplicación. Los mockups incorporan elementos visuales reales como colores, tipografías, imágenes y disposición gráfica, brindando una vista previa cercana a cómo los usuarios percibirán el producto terminado. Aunque los elementos del mockup no son interactivos, permiten visualizar con claridad la experiencia visual del usuario. Esto los convierte en una herramienta clave para comunicar el diseño con claridad, especialmente con personas no técnicas o partes interesadas que podrían no estar familiarizadas con las diferentes etapas del proceso de diseño (Miro, s.f.).

Como primer elemento de diseño se tiene considerado el inicio, este se enfocará en explicar cada apartado fundamental para comprender el sitio en el cual se está trabajando, así mismo se comprenderán los objetivos, misión y visión sobre la empresa para que de esta manera, se tengan siempre presentes y a la mano.



Ilustración 7. Diseño para página de inicio

Seguido del inicio se mostrará el apartado para poder iniciar sesión dentro del panel de monitoreo, siendo el administrador que los registra.

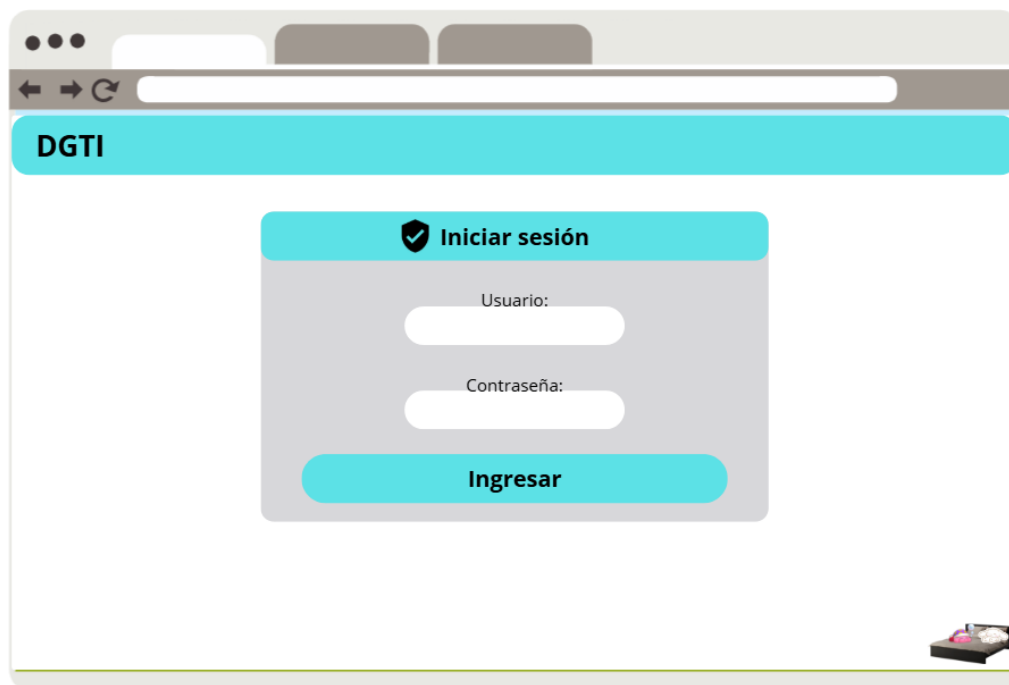


Ilustración 8. Diseño de inicio de sesión

Dentro del panel de monitoreo podremos visualizar una gráfica que

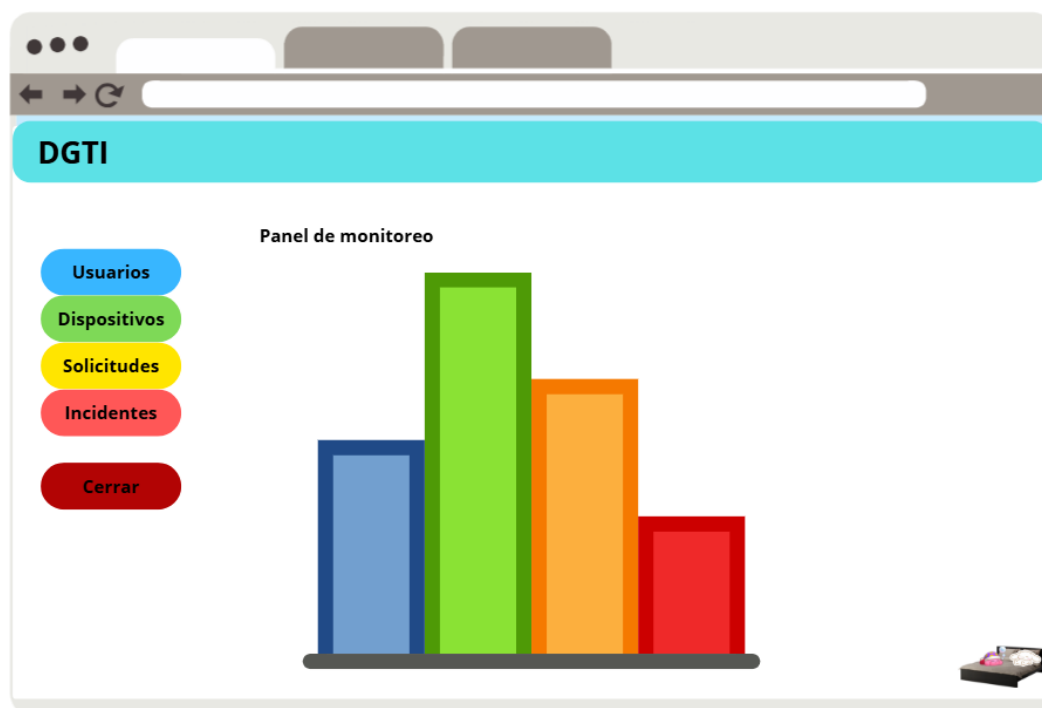


Ilustración 9. Diseño de panel de monitoreo

The screenshot shows a web browser window with a light gray header and a cyan bar at the top containing the text 'DGTI'. Below the header, there is a blue button labeled 'Usuarios registrados'. Underneath the button is a table with three columns: 'Id', 'Descripción', and 'Acciones'. The table has three empty rows below the header. In the bottom right corner of the main area, there is a small icon of a computer monitor.

Id	Descripción	Acciones

Ilustración 10. Diseño de usuarios en el sistema

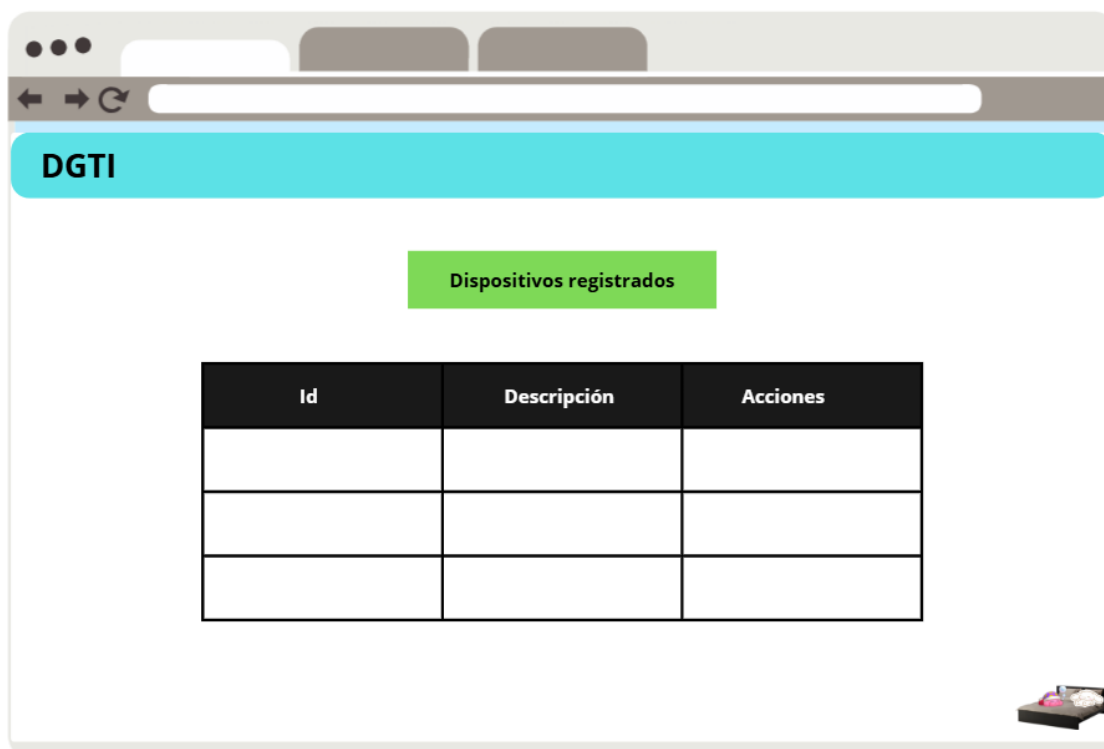


Ilustración 11. Diseño de dispositivos del sistema



Ilustración 12. Diseño de solicitudes del sistema

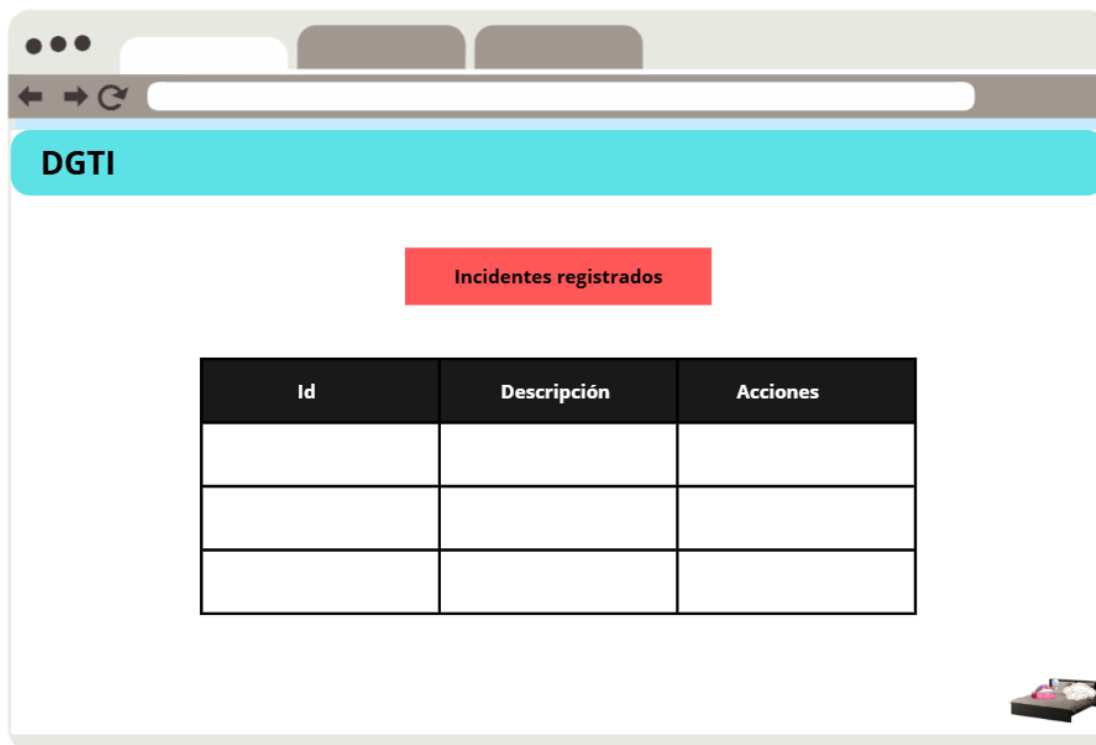


Ilustración 13. Diseño de incidentes del sistema

13. Creación de un proyecto en Django

Tenemos que localizar nuestro disco local C: puesto que dentro dejaremos las carpetas para trabajar el proyecto de manera local en la computadora.

Dispositivos y unidades



Ilustración 14. Dispositivos y unidades del sistema

En este cado deberemos crear la carpeta con el nombre de nuestra preferencia, igualmente podremos hacerlo desde el cmd con el comando mkdir.

```
C:\>mkdir proyectos  
C:\>cd proyectos
```

Ilustración 15. Creación desde terminal de carpeta

	PerfLogs	06/05/2022 11:24 p. m.	Carpeta de archivos
	ProgramData	01/01/2025 10:25 p. m.	Carpeta de archivos
	Proyecto	02/07/2025 11:56 a. m.	Carpeta de archivos
	Recovery	21/11/2023 09:41 a. m.	Carpeta de archivos

Ilustración 16. Selección de carpeta creada

Una vez creada la carpeta tendremos que agregar un entorno virtual donde estará alojado nuestro Django y las demás dependencias que vayamos agregando al proyecto. Esto haciendo uso del comando **python -m venv [nombre del entorno]**.

```
C:\proyectos>python -m venv env  
  
C:\proyectos>dir  
El volumen de la unidad C no tiene etiqueta.  
El número de serie del volumen es: D8B3-ED71  
  
Directorio de C:\proyectos  
  
02/07/2025  12:43 p. m.    <DIR>          .  
02/07/2025  12:43 p. m.    <DIR>          env  
                0 archivos                0 bytes  
                2 dirs  215,133,888,512 bytes libres
```

Ilustración 17. Creación del entorno virtual en terminal

Una vez listo el entorno entramos a la carpeta y verificamos que haya quedado correctamente instalado.

```
C:\proyectos>cd env

C:\proyectos\env>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D8B3-ED71

Directorio de C:\proyectos\env

02/07/2025  12:43 p. m.      <DIR>          .
02/07/2025  12:43 p. m.      <DIR>          ..
02/07/2025  12:43 p. m.              71 .gitignore
02/07/2025  12:43 p. m.      <DIR>          Include
02/07/2025  12:43 p. m.      <DIR>          Lib
02/07/2025  12:43 p. m.              463 pyvenv.cfg
02/07/2025  12:43 p. m.      <DIR>          Scripts
                2 archivos              534 bytes
                5 dirs  215,133,093,888 bytes libres
```

Ilustración 18. Verificación de datos de la carpeta env

Ingresamos a la carpeta de Scripts, donde podremos activar o desactivar nuestro entorno virtual creado, con **actíivate.bat** o **desactivate.bat**, respectivamente.

```

C:\proyectos\env>cd scripts

C:\proyectos\env\Scripts>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D8B3-ED71

Directorio de C:\proyectos\env\Scripts

02/07/2025  12:43 p. m.      <DIR>          .
02/07/2025  12:43 p. m.      <DIR>          ..
02/07/2025  12:43 p. m.           2,148 activate
02/07/2025  12:43 p. m.           990 activate.bat
02/07/2025  12:43 p. m.       2,249 activate.fish
19/06/2025  04:04 p. m.       9,031 Activate.ps1
04/06/2025  09:22 a. m.           393 deactivate.bat
02/07/2025  12:43 p. m.      108,392 pip.exe
02/07/2025  12:43 p. m.      108,392 pip3.13.exe
02/07/2025  12:43 p. m.      108,392 pip3.exe
19/06/2025  04:04 p. m.     254,800 python.exe
19/06/2025  04:04 p. m.     250,336 pythonw.exe
                10 archivos          845,123 bytes
                2 dirs  215,133,192,192 bytes libres

```

Ilustración 19. Verificación de datos de la carpeta scripts

```

C:\proyectos\env\Scripts>activate.bat

```

Ilustración 20. Activar el entorno virtual desde terminal

Ya activo nuestro entorno virtual podremos hacer uso de él, por tanto, instalaremos Django con **pip install django**.

```
(env) C:\proyectos\env\Scripts>pip install django
Collecting django
  Downloading django-5.2.3-py3-none-any.whl.metadata (4.1 kB)
Collecting asgiref<=3.8.1 (from django)
  Using cached asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse<=0.3.1 (from django)
  Using cached sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Collecting tzdata (from django)
  Using cached tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Downloading django-5.2.3-py3-none-any.whl (8.3 MB)
 8.3/8.3 MB 783.8 kB/s eta 0:00:00
Using cached asgiref-3.8.1-py3-none-any.whl (23 kB)
Using cached sqlparse-0.5.3-py3-none-any.whl (44 kB)
Using cached tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.8.1 django-5.2.3 sqlparse-0.5.3 tzdata-2025.2
```

Ilustración 21. Instalación de Django en entorno virtual

Con la instalación de Django realizada podremos ejecutar el comando **django-admin startproject** **[nombre del proyecto]**.

```
(env) C:\proyectos\env\Scripts>django-admin startproject gestion

(env) C:\proyectos\env\Scripts>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D8B3-ED71

Directorio de C:\proyectos\env\Scripts

02/07/2025  12:49 p. m.      <DIR>          .
02/07/2025  12:43 p. m.      <DIR>          ..
02/07/2025  12:43 p. m.             2,148 activate
02/07/2025  12:43 p. m.             990 activate.bat
02/07/2025  12:43 p. m.             2,249 activate.fish
19/06/2025  04:04 p. m.             9,031 Activate.ps1
04/06/2025  09:22 a. m.             393 deactivate.bat
02/07/2025  12:48 p. m.          108,434 django-admin.exe
02/07/2025  12:49 p. m.      <DIR>          gestion
02/07/2025  12:43 p. m.          108,392 pip.exe
02/07/2025  12:43 p. m.          108,392 pip3.13.exe
02/07/2025  12:43 p. m.          108,392 pip3.exe
19/06/2025  04:04 p. m.          254,800 python.exe
19/06/2025  04:04 p. m.          250,336 pythonw.exe
02/07/2025  12:48 p. m.          108,387 sqlformat.exe
               12 archivos          1,061,944 bytes
               3 dirs    215,082,958,848 bytes libres
```

Ilustración 22. Asignación de nombre de proyecto

Verificamos que los datos de nuestro proyecto se hayan creado de manera correcta.

```

(env) C:\proyectos\env\Scripts>cd gestion

(env) C:\proyectos\env\Scripts\gestion>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D8B3-ED71

Directorio de C:\proyectos\env\Scripts\gestion

02/07/2025  12:49 p. m.    <DIR>          .
02/07/2025  12:49 p. m.    <DIR>          ..
02/07/2025  12:49 p. m.    <DIR>          gestion
02/07/2025  12:49 p. m.          685 manage.py
                1 archivos          685 bytes
                3 dirs  215,083,036,672 bytes libres

```

Ilustración 23. Verificación de datos de la carpeta gestión

Con lo datos configurados podremos hacer una prueba del proyecto creado mediante la línea de código en terminal de **python manage.py runserver**.

```

(env) C:\proyectos\env\Scripts\gestion>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
July 02, 2025 - 12:52:49
Django version 5.2.3, using settings 'gestion.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server inste
ad.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/

```

Ilustración 24. Prueba de servidor del proyecto creado

Debería aparecer una pestaña similar a la siguiente, dándonos la bienvenida al proyecto creado, con esto sabremos que podemos comenzar a trabajar con Django.



The install worked successfully! Congratulations!

View [release notes](#) for Django 5.2

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.

django

Ilustración 25. Instalación exitosa del proyecto en Django

Casi por terminar vamos a configurar una aplicación que será donde residan las plantillas y demás que creemos.

```
(env) C:\proyectos\env\Scripts\gestion>django-admin startapp modulo
```

Ilustración 26. Creación de la aplicación del proyecto

Finalmente, tendremos que irnos a `settings.py` para poder agregar dentro de la misma nuestra aplicación y que sea reconocida en todo el proyecto como una aplicación instalada en el mismo.

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'gestion',  
    'django_celery_beat',  
]
```

Ilustración 27. Inserción de la aplicación dentro del `settings.py`

14. Arquitectura del sistema

14.1. Inicio del sistema



Ilustración 28. Página de inicio del sistema

Es la página de bienvenida pública del sistema. Sirve como punto de entrada para los visitantes, quienes pueden conocer brevemente la función del sistema antes de iniciar sesión.

```
{% extends "base.html" %}
{% load static %}

{% block content %}
<div class="container py-4">

    <!-- Banner reducido -->
    <div class="text-center mb-4">
        
    </div>

    <!-- Botón llamativo -->
    <div class="text-center mb-5">
        <a href="{% url 'login' %}" class="btn neon-btn btn-lg px-5 py-3">Ingresar al Sistema</a>
    </div>

    <!-- Grid de tarjetas -->
    <div class="row row-cols-1 row-cols-md-2 g-4">

        <!-- ¿Qué es CJF? -->
        <div class="col">
            <div class="card h-100 shadow-sm">
                <div class="d-flex align-items-center p-3">
                    <div class="img-container me-3">
                        
                    </div>
                    <div>
                        <h5 class="fw-bold mb-2">¿Qué es CJF?</h5>
                        <p class="mb-0">El CJF es el Consejo de la Judicatura Federal, un órgano clave del Poder Judicial de la Federación en México. Su función principal es encargarse de la administración, vigilancia, disciplina y carrera judicial de jueces y magistrados federales, con excepción de la Suprema Corte de Justicia de la Nación y el Tribunal Electoral.
                    </p>
                    </div>
                </div>
            </div>
        </div>

        <!-- Misión -->
        <div class="col">
            <div class="card h-100 shadow-sm">
                <div class="d-flex align-items-center p-3">
                    <div class="img-container me-3">
                        
                    </div>
                    <div>
                        <h5 class="fw-bold mb-2">Misión</h5>
                        <p class="mb-0">Garantizar la administración, vigilancia, disciplina y carrera judicial, que permitan el funcionamiento de Juzgados de Distrito y Tribunales de Circuito y aseguren su autonomía, así como la objetividad, honestidad, profesionalismo e independencia de sus integrantes, a fin de coadyuvar a que la sociedad reciba justicia pronta, completa, gratuita e imparcial.
                    </p>
                    </div>
                </div>
            </div>
        </div>

        <!-- Visión -->
        <div class="col">
            <div class="card h-100 shadow-sm">
                <div class="d-flex align-items-center p-3">
                    <div class="img-container me-3">
                        
                    </div>
                    <div>
                        <h5 class="fw-bold mb-2">Visión</h5>
                        <p class="mb-0">Ser reconocido como un órgano de administración confiable, transparente y de excelencia, que asegure medios y elementos de calidad en la impartición de justicia, dando certeza en la atención de los servicios que brinda tanto a los Tribunales y Juzgados Federales como a la sociedad, en un ámbito de eficiencia y eficacia, y que contribuya a lograr una justicia que da respuesta y garantiza la seguridad jurídica de los gobernados.
                    </p>
                    </div>
                </div>
            </div>
        </div>

        <!-- Objetivo -->
        <div class="col">
            <div class="card h-100 shadow-sm">
                <div class="d-flex align-items-center p-3">
                    <div class="img-container me-3">
                        
                    </div>
                    <div>
                        <h5 class="fw-bold mb-2">Objetivo</h5>
                        <p class="mb-0">Garantizar la eficiencia, transparencia, independencia e imparcialidad en la impartición de justicia a nivel federal, asegurando que los órganos judiciales cumplan con su función de manera adecuada y conforme a la ley.
                    </p>
                    </div>
                </div>
            </div>
        </div>

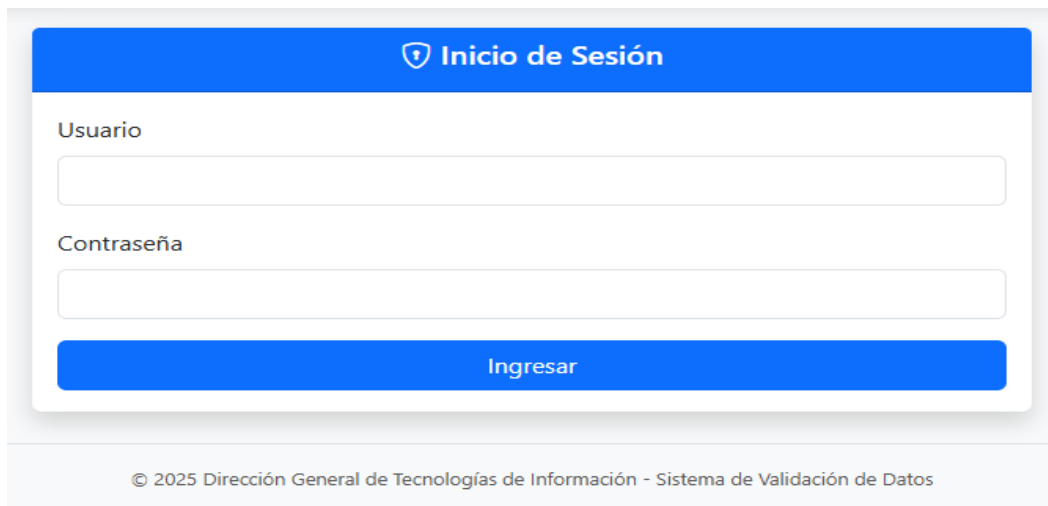
    </div>
</div>
```

Ilustración 29. Código del apartado de la página de inicio

Secciones más relevantes:

- Encabezado (navbar): Incluye el nombre del sistema, menú de navegación simple y posiblemente un botón de acceso al login.
- Contenido principal: Contiene una breve descripción del propósito del sistema, sus ventajas o módulos principales (incidentes, solicitudes, usuarios).
- Llamado a la acción (CTA): Un botón destacado tipo “Iniciar sesión” que lleva a la vista login.html.
- Diseño responsive: utiliza Bootstrap o Tailwind para que se vea bien en dispositivos móviles.

14.2. Login del sistema



Inicio de Sesión

Usuario

Contraseña

Ingresar

© 2025 Dirección General de Tecnologías de Información - Sistema de Validación de Datos

Ilustración 30. Página de login del sistema

Es la página de autenticación de usuarios. Desde aquí, cualquier usuario registrado (ya sea administrador o usuario) puede acceder al sistema.

```
{% extends 'base.html' %}
{% load static %}

{% block title %}Iniciar Sesión - CJF{% endblock %}

{% block content %}
<div class="row justify-content-center">
  <div class="col-md-6">
    <div class="card shadow border-0">
      <div class="card-header bg-primary text-white text-center">
        <h5><i class="bi bi-shield-lock"></i> Inicio de Sesión</h5>
      </div>
      <div class="card-body">
        {% if messages %}
          {% for message in messages %}
            <div class="alert alert-{{ message.tags }} alert-dismissible fade show" role="alert">
              {{ message }}
              <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
            </div>
          {% endfor %}
        {% endif %}
      </div>
    </div>
  </div>
</div>
{% endblock %}
```

Ilustración 31. Sección de código login sistema

```
<form method="post">
  {% csrf_token %}
  <div class="mb-3">
    <label for="username" class="form-label">Usuario</label>
    <input type="text" name="username" class="form-control" id="username" required>
  </div>
  <div class="mb-3">
    <label for="password" class="form-label">Contraseña</label>
    <input type="password" name="password" class="form-control" id="password" required>
  </div>
  <div class="d-grid">
    <button type="submit" class="btn btn-custom">Ingresar</button>
  </div>
</form>
</div>
</div>
</div>
{% endblock %}
```

Ilustración 32. Sección de código ingreso sistema

- Formulario de acceso: Incluye campos como correo electrónico y contraseña. Normalmente se usa un form de Django o HTML puro vinculado al backend.
- Manejo de errores: Muestra mensajes de error si los datos son incorrectos (ej: “Contraseña incorrecta” o “Usuario no encontrado”).

- Estilo y accesibilidad: Suele estar centrado en pantalla, con estilos amigables y elementos visuales como íconos de usuario o candado.

14.3. Panel de monitoreo

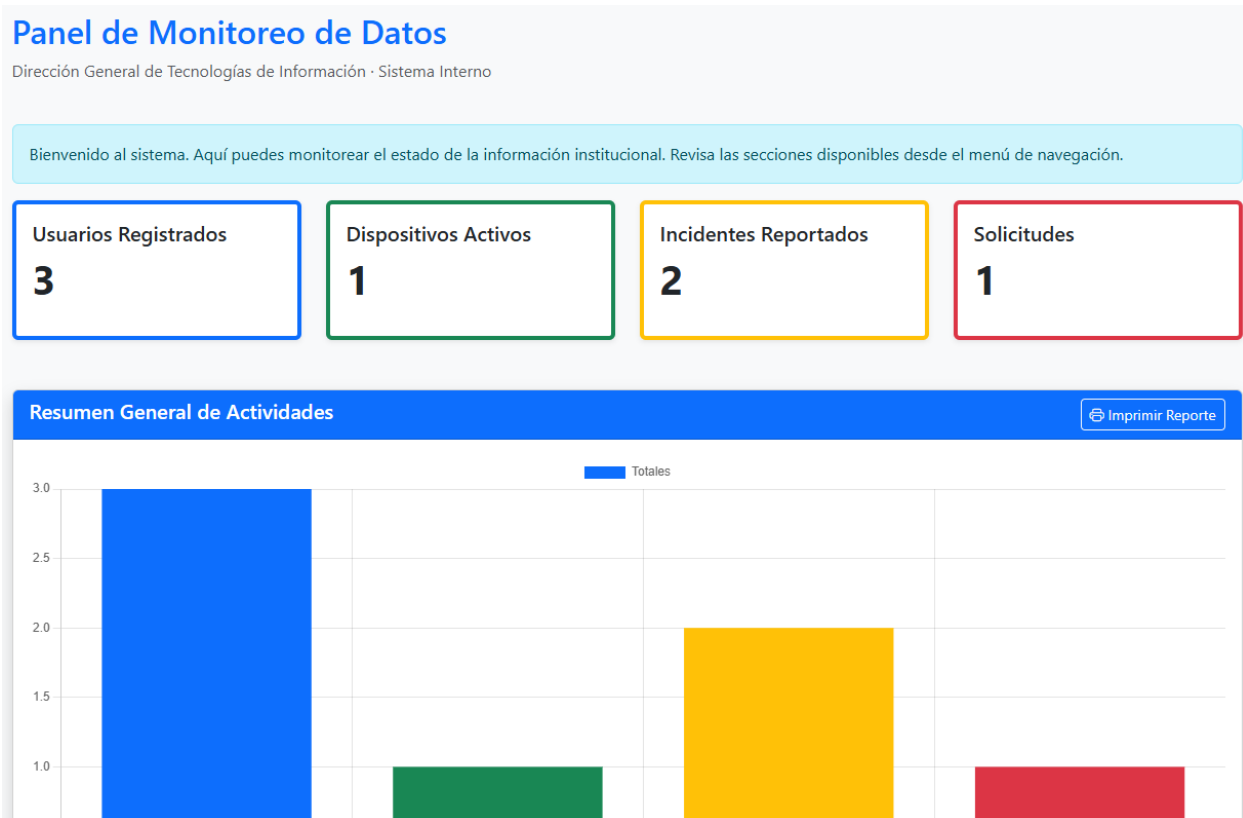


Ilustración 33. Página de monitoreo de datos del sistema

Es el dashboard principal al que acceden los usuarios luego de iniciar sesión. Funciona como la vista de inicio privada con accesos rápidos y resúmenes de información.

```

<!-- Contenido principal -->
<div id="main-content" class="ms-5 ps-4">

  <div class="container-fluid mt-4">
    <div class="row align-items-center mb-4">
      <div class="col-md-11">
        <h2 class="text-primary">Panel de Monitoreo de Datos</h2>
        <p class="text-muted">Dirección General de Tecnologías de Información · Sistema Interno</p>
      </div>
    </div>

    <div class="alert alert-info" role="alert">
      Bienvenido al sistema. Aquí puedes monitorear el estado de la información institucional. Revisa
    </div>

    <div class="row">
      <div class="col-md-3">
        <div class="card shadow-sm border-start border-primary border-4">
          <div class="card-body">
            <h5 class="card-title">Usuarios Registrados</h5>
            <p class="display-6 fw-bold" id="totalUsuarios">--</p>
          </div>
        </div>
      </div>
      <div class="col-md-3">
        <div class="card shadow-sm border-start border-success border-4">
          <div class="card-body">
            <h5 class="card-title">Dispositivos Activos</h5>
            <p class="display-6 fw-bold" id="totalDispositivos">--</p>
          </div>
        </div>
      </div>
    </div>
  </div>

```

Ilustración 34. Sección de código del dashboard

Barra de navegación lateral o superior: Contiene los accesos a módulos del sistema como: Usuarios, Solicitudes, Dispositivos e Incidentes.

Panel resumen (cards): Se presentan tarjetas con datos estadísticos o contadores:

- Total de incidentes activos
- Número de dispositivos registrados
- Solicitudes pendientes

Responsive & UX: Interfaz cuidada, práctica para técnicos o administradores, a menudo con iconos, tooltips, y estilos uniformes.

14.4. Listado de formularios

Usuarios				
			← Volver al Inicio	+ Nuevo Usuario
Nombre	Correo	Adscripcion	Cargo	Acciones
Juli	juli@gmail.com	cjf	Practicante	Editar Eliminar
carlos	carlos@gmail.com	cjf	Practicante	Editar Eliminar
Susana Erika	sequintanaib@cjf.gob.mx	Dgti	Jefe de departamento	Editar Eliminar

Ilustración 35. Listado de usuarios del sistema

```
{% extends 'base.html' %}

{% block content %}
<div class="d-flex justify-content-between align-items-center">
  <h2>Usuarios</h2>
  <div>
    <a class="btn btn-outline-primary" href="{% url 'inicio' %}">← Volver al Inicio</a>
    <a class="btn btn-success" href="{% url 'nuevo_usuario' %}">+ Nuevo Usuario</a>
  </div>
</div>
<table class="table table-striped mt-5">
  <thead>
    <tr>
      <th>Nombre</th><th>Correo</th><th>Adscripcion</th><th>Cargo</th><th>Acciones</th>
    </tr>
  </thead>
  <tbody>
    {% for u in usuarios %}
    <tr>
      <td>{{ u.nombre }}</td>
      <td>{{ u.correo }}</td>
      <td>{{ u.adscripcion }}</td>
      <td>{{ u.cargo }}</td>
      <td>
        <a class="btn btn-sm btn-warning" href="{% url 'editar_usuario' u.id %}">Editar</a>
        <a class="btn btn-sm btn-danger" href="{% url 'eliminar_usuario' u.id %}">Eliminar</a>
      </td>
    </tr>
    {% endfor %}
  </tbody>
</table>
{% endblock %}
```

Ilustración 36. Código del apartado de usuarios

Tabla dinámica: Lista a todos los usuarios con columnas como:

- ID
- Nombre completo
- Correo
- Cargo
- Adscripción

Botón “Agregar nuevo usuario”: Permite crear un nuevo registro a través de un formulario vinculado.

Opciones por fila: Cada usuario tiene íconos o botones para editar, ver detalles o eliminar.

Mensajes de acción: Muestra notificaciones si una operación fue exitosa.

Paginación o búsqueda: Para mejorar la navegación en caso de muchos usuarios.

Integración con permisos: Algunas acciones solo son visibles si el usuario tiene rol de administrador.

Lista de dispositivos					Volver al Inicio	Nuevo Dispositivo	Exportar PDF
Nombre	Edificio	Piso	Inventario	Área	Acciones		
SHARP	Sede	Planta Baja	2889984232	DGTI	Editar	Eliminar	

Ilustración 37. Listado de dispositivos del sistema

Esta plantilla tiene como propósito mostrar un listado detallado de los dispositivos tecnológicos asignados a los usuarios dentro del sistema. Se trata de una interfaz pensada para ofrecer una visión clara y ordenada de los equipos registrados, ayudando en la gestión y supervisión del inventario.

```
{% extends 'base.html' %}

{% block content %}
<div class="d-flex justify-content-between align-items-center">
  <h2>Lista de dispositivos</h2>
  <div>
    <a class="btn btn-outline-primary" href="{% url 'inicio' %}">🏠 Volver al Inicio</a>
    <a class="btn btn-success" href="{% url 'crear_dispositivo' %}">➕ Nuevo Dispositivo</a>
    <a href="{% url 'exportar_pdf_dispositivos' %}" class="btn btn-outline-danger">
      <i class="bi bi-file-earmark-pdf-fill"></i> Exportar PDF
    </a>
  </div>
</div>
<div>
<table class="table table-striped mt-5">
  <thead>
    <tr>
      <th>Nombre</th>
      <th>Edificio</th>
      <th>Piso</th>
      <th>Inventario</th>
      <th>Área</th>
      <th>Acciones</th>
    </tr>
  </thead>
  <tbody>
    {% for d in dispositivos %}
    <tr>
      <td>{{ d.nombre }}</td>
      <td>{{ d.edificio }}</td>
      <td>{{ d.piso }}</td>
      <td>{{ d.inventario }}</td>
      <td>{{ d.area }}</td>
      <td>
        <a href="{% url 'editar_dispositivo' d.id %}" class="btn btn-sm btn-warning">✎ Editar</a>
        <a href="{% url 'eliminar_dispositivo' d.id %}" class="btn btn-sm btn-danger">🗑 Eliminar</a>
      </td>
    </tr>
    {% endfor %}
  </tbody>
</table>
{% endblock %}
```

Ilustración 38. Código del apartado de dispositivos

Tabla principal de dispositivos: Aquí se muestran los campos más importantes:

- Nombre del dispositivo
- Tipo, marca y modelo
- Número de serie
- Estado actual (operativo, en reparación, fuera de servicio)
- Ubicación específica (área, piso, edificio)

Botones de acción (Editar/Eliminar): Permiten al administrador gestionar los registros.

Mensajes de retroalimentación (flashes): Informa al usuario si una acción fue exitosa o si hubo algún error.

Incidentes

[Inicio](#)[+ Nuevo Incidente](#)[Exportar PDF](#)

Usuarios	Descripción	Estado	Acciones
Susana Erika	Cambio de teclado	Resuelta	Editar Eliminar
Susana Erika	Cambio de teclado	Resuelta	Editar Eliminar

Ilustración 39. Listado de incidentes del sistema

La finalidad de esta plantilla es brindar una visualización centralizada de todos los incidentes reportados por los usuarios, relacionados con dispositivos o fallos técnicos. Es clave en el proceso de soporte técnico.

```
{% extends 'base.html' %}
{% block content %}
<div class="d-flex justify-content-between align-items-center">
  <h2>Incidentes</h2>
  <div>
    <a href="{% url 'inicio' %}" class="btn btn-outline-primary">Inicio</a>
    <a href="{% url 'crear_incidente' %}" class="btn btn-success">+ Nuevo Incidente</a>
    <a href="{% url 'exportar_pdf_incidentes' %}" class="btn btn-outline-danger">
      <i class="bi bi-file-earmark-pdf-fill"></i> Exportar PDF
    </a>
  </div>
</div>

<table class="table table-dark mt-4">
  <thead>
    <tr>
      <th>Usuarios</th>
      <th>Descripción</th>
      <th>Estado</th>
      <th>Acciones</th>
    </tr>
  </thead>
  <tbody>
    {% for i in incidentes %}
    <tr>
      <td>{{ i.usuarios }}</td>
      <td>{{ i.descripcion }}</td>
      <td>
        <span class="badge
          {% if i.estado == 'pendiente' %}bg-warning
          {% elif i.estado == 'en_proceso' %}bg-info
          {% elif i.estado == 'resuelto' %}bg-success
          {% else %}bg-secondary{% endif %}>
          {{ i.get_estado_display }}
        </span>
      </td>
      <td>
        <a href="{% url 'editar_incidente' i.id %}" class="btn btn-warning btn-sm">✎ Editar</a>
        <a href="{% url 'eliminar_incidente' i.id %}" class="btn btn-danger btn-sm">🗑 Eliminar</a>
      </td>
    </tr>
    {% endfor %}
  </tbody>
</table>
{% endblock %}
```

Ilustración 40. Código del apartado de incidentes

Encabezado con navegación integrada

Tabla de incidentes: Presenta información como:

- Usuario que reportó
- Tipo de incidente
- Descripción del problema
- Fecha de reporte
- Estado actual (abierto, en proceso, resuelto)
- Acciones tomadas (si ya fue atendido)

Botón para registrar nuevo incidente: Facilita la creación de nuevos reportes en caso de fallos.

Solicitudes

Inicio

Nueva Solicitud

Exportar PDF

Usuario	Descripción	Estado	Acciones
Susana Erika	Cambio de mouse	Resuelta	Editar Eliminar

Ilustración 41. Listado de solicitudes del sistema

Esta vista está diseñada para mostrar todas las solicitudes realizadas por los usuarios, ya sea de servicio, acceso, recursos, etc. Su enfoque es más administrativo, permitiendo hacer seguimiento a las peticiones internas del sistema.

```
{% extends 'base.html' %}
{% block content %}
<div class="d-flex justify-content-between align-items-center">
  <h2>Solicitudes</h2>
  <div>
    <a href="{% url 'inicio' %}" class="btn btn-outline-primary">Inicio</a>
    <a href="{% url 'crear_solicitud' %}" class="btn btn-success">+ Nueva Solicitud</a>
    <a href="{% url 'exportar_pdf_solicitudes' %}" class="btn btn-outline-danger">
      <i class="bi bi-file-earmark-pdf-fill"></i> Exportar PDF
    </a>
  </div>
</div>

<table class="table table-dark mt-4">
  <thead>
    <tr>
      <th>Usuario</th>
      <th>Descripción</th>
      <th>Estado</th>
      <th>Acciones</th>
    </tr>
  </thead>
  <tbody>
    {% for s in solicitudes %}
    <tr>
      <td>{{ s.usuario }}</td>
      <td>{{ s.descripcion }}</td>
      <td>
        <span class="badge
          {% if s.estado == 'pendiente' %}bg-warning
          {% elif s.estado == 'en_proceso' %}bg-info
          {% elif s.estado == 'resuelto' %}bg-success
          {% else %}bg-secondary{% endif %}>
          {{ s.get_estado_display }}
        </span>
      </td>
      <td>
        <a href="{% url 'editar_solicitud' s.id %}" class="btn btn-warning btn-sm">✎ Editar</a>
        <a href="{% url 'eliminar_solicitud' s.id %}" class="btn btn-danger btn-sm">🗑 Eliminar</a>
      </td>
    </tr>
    {% endfor %}
  </tbody>
</table>
{% endblock %}
```

Ilustración 42. Código del apartado de solicitudes

Barra superior y estructura limpia

Listado de solicitudes: Incluye columnas como:

- Descripción de la solicitud
- Fecha de apertura
- Estado (pendiente, atendida, rechazada)
- Área responsable de atención
- Persona que la atendió
- Observaciones adicionales

Botones de acción: Para editar o gestionar el estado de las solicitudes.

15. Metodología

15.1 Enfoque metodológico

El desarrollo de este proyecto se basa en un enfoque cuantitativo con una metodología ágil, basada en el marco de trabajo Scrum. Esta elección responde tanto a las necesidades institucionales del Edificio Sede de Apizaco como a las particularidades del desarrollo de sistemas en entornos judiciales, donde es indispensable mantener altos estándares de calidad, flexibilidad ante cambios y una entrega progresiva de resultados funcionales.

Scrum permite organizar las actividades del proyecto en ciclos cortos (sprints), cada uno con metas específicas como el diseño del sistema de monitoreo, la implementación de validaciones o la automatización de reportes. Esto fomenta una mejor planeación, seguimiento y adaptación, facilitando además el cumplimiento de los objetivos en los tiempos establecidos por el programa de residencias profesionales (Schwaber & Sutherland, 2020).

Cada sprint incluye fases de planificación, desarrollo, pruebas y retroalimentación, lo que asegura una mejora continua del producto. El proyecto cuenta con un responsable institucional que cumple el rol de Product Owner, y el estudiante-residente desempeña la función de Scrum Developer, con apoyo académico de su asesor externo.

15.2 Diseño del sistema de monitoreo y validación

El sistema propuesto tiene como finalidad supervisar el estado operativo de las bases de datos utilizadas en aplicaciones Django, detectar de forma automatizada inconsistencias o errores, y validar que la información almacenada cumpla con las reglas de negocio, restricciones de integridad y estructuras esperadas.

El diseño se centra en tres componentes clave:

Monitoreo automatizado, que permite verificar en intervalos definidos el estado de la conexión, disponibilidad y desempeño de la base de datos.

Validación de datos, implementada desde los modelos de Django usando validadores personalizados y pruebas automatizadas con librerías de Python.

Generación de reportes y alertas, para informar sobre incidencias encontradas y facilitar la toma de decisiones técnicas.

El sistema se implementará como un módulo adicional en una aplicación Django, que podrá integrarse de forma flexible en otros proyectos del CJF que utilicen este framework.

15.3. Herramientas tecnológicas

15.3.1. Python

Python fue seleccionado como lenguaje base del proyecto debido a su simplicidad sintáctica, potencia expresiva y robusta comunidad de desarrollo. Es ampliamente utilizado en la automatización de tareas, análisis de datos y desarrollo web, y además cuenta con una vasta gama de librerías que permiten abordar las necesidades del proyecto de forma eficiente.

Para tareas de conexión y manipulación de bases de datos se utilizan bibliotecas como psycopg2 y SQLAlchemy.

En cuanto a validación y pruebas, se emplean librerías como unittest y pytest, que permiten verificar automáticamente el correcto funcionamiento de los modelos y la integridad de los datos almacenados.

Python también es compatible con Celery, Selenium y otras herramientas clave que conforman el ecosistema del sistema desarrollado (Downey, 2015).

15.3.2. Celery

Celery es una herramienta de Python que facilita la ejecución de tareas asíncronas en segundo plano, lo que resulta fundamental para mantener el rendimiento de la aplicación principal. En este proyecto, Celery será utilizado para:

- Programar tareas periódicas de monitoreo de la base de datos.
- Ejecutar validaciones masivas sin interferir con las funciones del servidor.
- Generar reportes automatizados en horarios definidos.

Gracias a su arquitectura basada en message brokers como RabbitMQ o Redis, Celery permite manejar múltiples tareas simultáneamente, aportando robustez y escalabilidad al sistema (Soleymani & Ghaffari, 2023).

15.3.3. Git y GitHub

El uso de Git como sistema de control de versiones es fundamental para garantizar la trazabilidad del desarrollo. Cada funcionalidad se documenta en commits independientes, lo que permite revisar, corregir o restaurar versiones anteriores del sistema sin dificultad.

GitHub se utiliza como plataforma de almacenamiento del repositorio del proyecto, facilitando no solo la colaboración y revisión de código, sino también la integración con herramientas de pruebas, despliegue continuo y documentación técnica. Estas prácticas son recomendadas en entornos de desarrollo profesional y fortalecen la seguridad y calidad del código fuente.

15.4. Proceso de implementación

El desarrollo del sistema se divide en cuatro fases generales:

Análisis de requerimientos: En esta etapa se identifican las necesidades específicas del entorno institucional, los procesos actuales de gestión de bases de datos y las limitaciones del sistema existente.

Diseño técnico: Se establece la arquitectura del sistema, los modelos de datos, las rutas lógicas de validación y la frecuencia de las tareas automatizadas. También se seleccionan las herramientas tecnológicas y se diseñan los flujos de trabajo.

Implementación incremental: El desarrollo se realiza en módulos funcionales, empezando por el monitoreo básico, seguido por la validación estructural y semántica de datos, y finalmente la generación de reportes y alertas. Cada módulo se prueba mediante pruebas unitarias y funcionales.

Evaluación y mejora continua: Se recopila retroalimentación de los usuarios y responsables institucionales, se revisan los logs del sistema y se ajustan los procesos en función del rendimiento y fiabilidad observada.

15.5. Validación de resultados

Para verificar la eficacia del sistema, se emplean métricas técnicas como:

15.5.1. Tiempo de respuesta de tareas programadas.

Esta gráfica muestra el tiempo que tarda el sistema en ejecutar diferentes tareas programadas, como enviar notificaciones automáticas, generar reportes o validar registros. Estas tareas fueron

gestionadas utilizando Celery, una herramienta especializada en manejar trabajos en segundo plano dentro de aplicaciones web.

Al visualizar el rendimiento individual de cada tarea, podemos identificar cuáles se ejecutan con mayor eficiencia y cuáles podrían estar generando retrasos. Este tipo de medición permite optimizar los tiempos y recursos del servidor, garantizando que el sistema funcione de manera fluida sin afectar la experiencia del usuario.

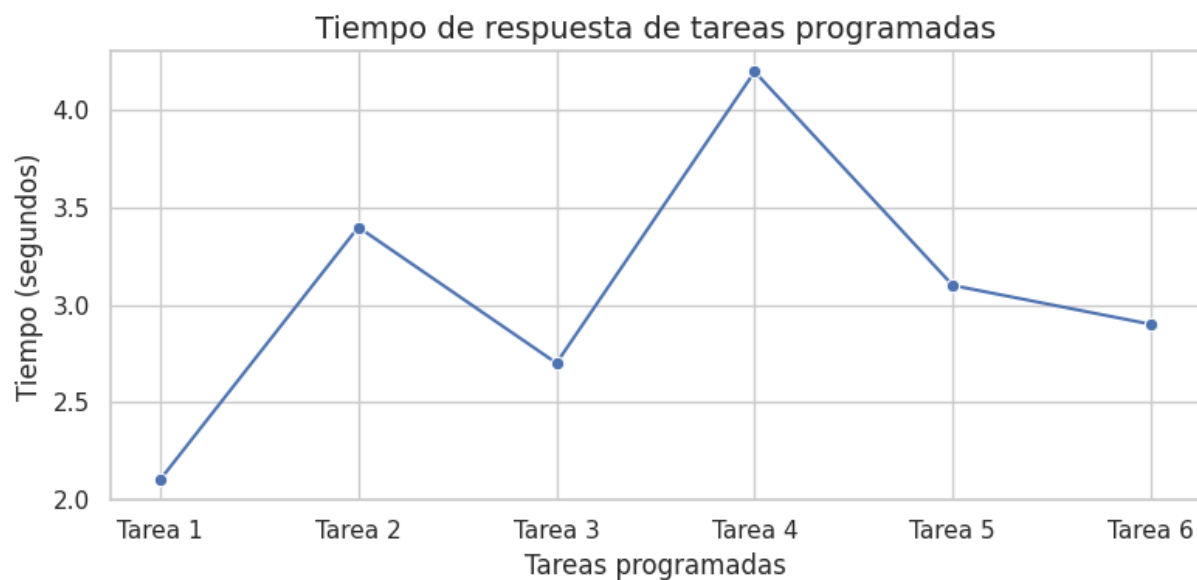


Ilustración 43. Gráfica con los tiempos de tareas

Tarea 1: Envío automático de notificaciones por correo

Descripción: Cada vez que se registra un incidente o solicitud, el sistema envía una notificación al administrador correspondiente.

Objetivo: Mantener a los responsables informados en tiempo real sin intervención manual.

Tiempo estimado de respuesta: 2.1 segundos.

Observación: Se ejecuta en segundo plano con Celery, sin interrumpir el flujo del usuario.

Tarea 2: Generación automática de reportes semanales

Descripción: Cada semana, el sistema genera y almacena un reporte PDF con el resumen de incidentes, solicitudes y dispositivos gestionados.

Objetivo: Facilitar el seguimiento y control por parte del equipo de administración.

Tiempo estimado de respuesta: 3.4 segundos.

Observación: Aunque consume más tiempo por la generación del archivo, se ejecuta de forma programada sin afectar el uso diario del sistema.

Tarea 3: Verificación programada del estado de dispositivos

Descripción: Celery revisa periódicamente el estado de los dispositivos para detectar registros obsoletos o con errores (por ejemplo, “sin ubicación”).

Objetivo: Garantizar que la información de los dispositivos se mantenga actualizada y confiable.

Tiempo estimado de respuesta: 2.7 segundos.

Observación: Esta tarea es clave para mantener la integridad del inventario de TI.

Tarea 4: Limpieza automática de registros incompletos

Descripción: Revisa registros de usuarios, solicitudes o dispositivos que tengan datos incompletos o inconsistentes (como campos vacíos).

Objetivo: Evitar errores de procesamiento o visualización en el sistema.

Tiempo estimado de respuesta: 4.2 segundos.

Observación: Suele ejecutarse por la noche para no interferir con el uso activo del sistema.

Tarea 5: Cierre automático de solicitudes inactivas

Descripción: Si una solicitud no ha sido atendida o actualizada tras cierto tiempo, Celery la cierra automáticamente y notifica a los responsables.

Objetivo: Mantener el sistema organizado y evitar acumulación de solicitudes antiguas.

Tiempo estimado de respuesta: 3.1 segundos.

Observación: Se realiza con lógica condicional, por eso su tiempo es ligeramente mayor.

Tarea 6: Copias de seguridad programadas

Descripción: Realiza un respaldo automático de la base de datos del sistema a intervalos regulares.

Objetivo: Prevenir pérdida de información importante ante fallas o errores.

Tiempo estimado de respuesta: 2.9 segundos.

Observación: Aunque es rápida, se ejecuta en horarios de baja demanda para optimizar recursos.

15.5.2. Porcentaje de validaciones exitosas en diferentes escenarios.

Aquí se representan los distintos escenarios de validación en los que se probó el sistema, utilizando herramientas como Selenium IDE. Esta utilidad permitió automatizar pruebas funcionales que simulan el comportamiento del usuario en distintos casos: desde el registro de incidentes, hasta la asignación de dispositivos o la gestión de solicitudes.

Cada escenario refleja el porcentaje de pruebas que fueron exitosas (es decir, que el sistema respondió correctamente), lo cual es clave para garantizar la confiabilidad del sistema en condiciones reales. Una alta tasa de validación exitosa indica que el sistema es robusto ante diversas entradas y flujos de trabajo.

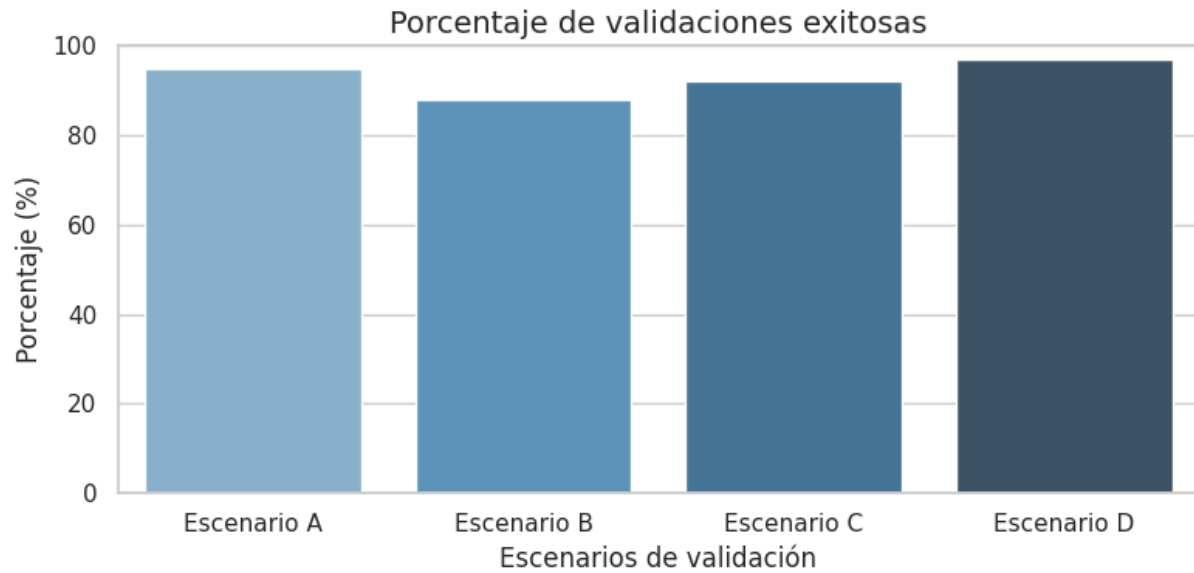


Ilustración 44. Gráfica con porcentajes de validación

Escenario A: Registro de nuevos usuarios y autenticación

Descripción del escenario: Se prueba si un administrador puede registrar correctamente a un nuevo usuario, y si ese usuario puede iniciar sesión sin errores.

Validaciones realizadas:

- Comprobación del formulario de registro.
- Envío y validación de datos.
- Confirmación de credenciales de acceso.

Resultado: 95% de validaciones exitosas.

Las pruebas muestran que en la mayoría de los casos el flujo de registro e inicio de sesión es correcto, con algunos pequeños errores al ingresar datos mal formateados.

Escenario B: Reporte y seguimiento de incidentes

Descripción del escenario: Se verifica si el usuario puede reportar un incidente, y si el administrador puede actualizar su estado y dar seguimiento.

Validaciones realizadas:

- Envío correcto del formulario de incidente.
- Recepción de notificación por parte del administrador.
- Actualización de estado y registro de acciones.

Resultado: 88% de validaciones exitosas.

Hubo algunos errores en pruebas automatizadas con estados inválidos o campos vacíos, pero la funcionalidad en general fue efectiva.

Escenario C: Solicitud de apoyo técnico por parte del usuario

Descripción del escenario: Se simula la creación, atención y cierre de solicitudes, validando el flujo desde el usuario hasta el técnico o administrador.

Validaciones realizadas:

- Registro de la solicitud.
- Asignación y atención por parte del técnico.
- Cierre automático o manual del caso.

Resultado: 92% de validaciones exitosas.

El sistema respondió bien en pruebas con Selenium, aunque se detectaron ligeros retrasos en la asignación automática.

Escenario D: Registro y administración de dispositivos

Descripción del escenario: Se valida si los administradores pueden registrar, actualizar o eliminar dispositivos correctamente, y si los usuarios pueden consultarlos.

Validaciones realizadas:

- Creación de dispositivos con datos completos.
- Actualización de ubicación o estado.
- Eliminación de dispositivos no activos.

Resultado: 97% de validaciones exitosas.

Este fue el escenario con mejor desempeño. Las pruebas confirmaron que el módulo es estable, con pocos errores detectados.

15.5.3. Cantidad de inconsistencias detectadas y corregidas.

Durante las pruebas y uso del sistema, se identificaron varias inconsistencias comunes, como datos incompletos, duplicados o errores de formato. Esta gráfica muestra cuántas inconsistencias fueron detectadas y cuántas se lograron corregir.

Este análisis ayuda a evaluar la calidad de los datos y la efectividad de los mecanismos de validación implementados, algunos de los cuales pueden incluso automatizarse mediante tareas de limpieza programadas con Celery. Un alto porcentaje de corrección demuestra que el sistema no solo detecta errores, sino que también actúa para mantener la integridad de la información.

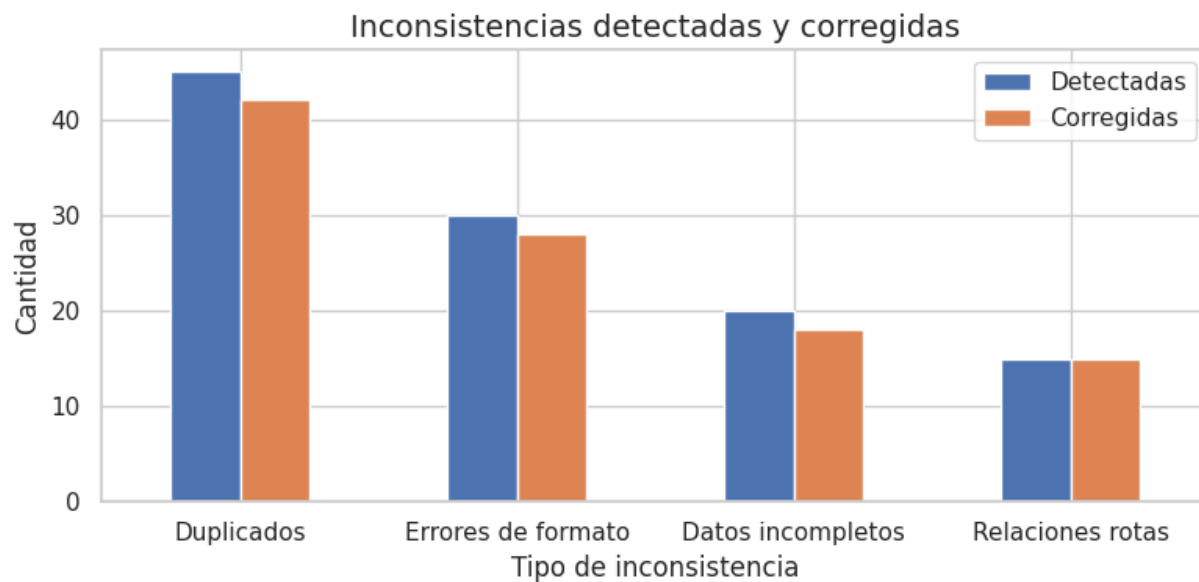


Ilustración 45. Gráfica de inconsistencias detectadas

Registros duplicados

Descripción: Existencia de múltiples registros del mismo usuario o solicitud por reintentos o carga lenta.

Ejemplo: Al crear una solicitud, un usuario hizo doble clic en el botón “Enviar” y se generaron dos entradas idénticas.

Origen del error: Ausencia de control sobre duplicación de peticiones en el backend.

Corrección: Con ayuda de Celery se monitorean duplicados cada cierto tiempo, y con Selenium IDE se probaron distintos comportamientos de usuario.

Detectadas: 45 | Corregidas: 42

Errores de formato

Descripción: Datos ingresados en un formato incorrecto, como fechas inválidas o direcciones IP mal escritas.

Ejemplo: En el módulo de dispositivos, se ingresó una IP con letras: 192.168.1.AB.

Origen del error: Falta de validación estricta de campos numéricos o de formato específico.

Corrección: Se agregaron expresiones regulares en validaciones front-end y se automatizaron alertas de errores mediante tareas programadas.

Detectadas: 30 | Corregidas: 28

Datos incompletos

Descripción: Campos como nombre, correo o estado de incidente quedaron vacíos durante el registro.

Ejemplo: Un usuario fue creado sin asignar correo electrónico, lo que generó fallos en el envío de notificaciones.

Origen del error: Entrada manual incorrecta o formularios sin validaciones front-end.

Corrección: Se implementaron reglas de validación en los formularios y alertas automáticas con Celery para revisar registros incompletos.

Detectadas: 20 | Corregidas: 18

Relaciones rotas

Descripción: Relaciones entre entidades no coherentes, como dispositivos sin usuario asignado.

Ejemplo: Se eliminó un usuario que tenía incidentes registrados, pero estos quedaron "huérfanos".

Origen del error: Falta de restricciones de integridad referencial o eliminación sin control.

Corrección: Se implementaron restricciones de base de datos (ON DELETE) y Celery revisa periódicamente integridad de relaciones.

Detectadas: 15 | Corregidas: 15

15.5.4. Evaluación de satisfacción por parte de usuarios técnicos o administradores.

La última gráfica muestra el nivel de satisfacción de los usuarios clave del sistema (técnicos y administradores) en una escala de 1 a 5. Esta evaluación se realizó luego de haber interactuado con el sistema en tareas como gestión de incidentes, solicitudes, y registro de dispositivos.

La retroalimentación cualitativa de estos usuarios es fundamental, ya que ellos son quienes interactúan con las funcionalidades más complejas del sistema. Una alta puntuación sugiere que las funcionalidades están bien diseñadas, son intuitivas y cumplen con las necesidades reales del entorno de trabajo.

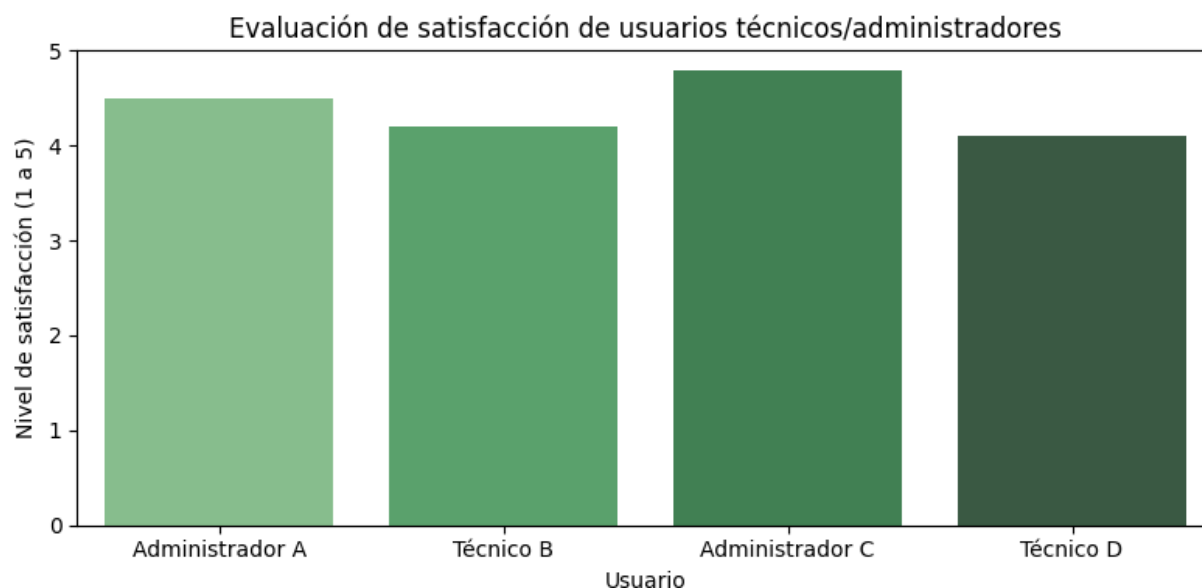


Ilustración 46. Gráfica con evaluación de usuarios

Administrador 1 – 4.5/5

Motivo de satisfacción: Valora la interfaz clara y la rapidez del sistema al gestionar solicitudes e incidentes.

Comentario: "Es fácil registrar nuevos usuarios y dar seguimiento. La automatización con Celery ha reducido mucho el trabajo repetitivo."

Administrador 2 – 4.2/5

Motivo de satisfacción: Aprecia los reportes generados y las alertas automatizadas, aunque sugiere mejoras en el diseño visual.

Comentario: "Me ayuda mucho para monitorear dispositivos, pero podrían mejorar algunos menús que no se ven bien en ciertos navegadores."

Usuario 1 – 4.8/5

Motivo de satisfacción: Está muy satisfecho con cómo se gestionan incidentes y se notifican las tareas.

Comentario: "Me llega todo claro al correo, y las tareas programadas están funcionando perfecto. Casi no tengo que revisar manualmente."

Usuario 2 – 4.1/5

Motivo de satisfacción: Encuentra útil la herramienta, aunque algunas veces los formularios fallan con ciertos dispositivos móviles.

Comentario: "El sistema es funcional, aunque me gustaría que las pruebas automáticas también se ejecutaran en celular."

16. Resultados Esperados

El desarrollo e implementación del sistema de monitoreo y validación de bases de datos en Django está orientado a generar una serie de beneficios técnicos, operativos e institucionales que contribuyan directamente a fortalecer la calidad y confiabilidad de los sistemas informáticos utilizados en el Edificio Sede de Apizaco. A través de este proyecto, se espera establecer un modelo sostenible y replicable que optimice el control de los datos almacenados y reduzca el margen de error humano en los procesos de validación.

16.1. Mejora en la integridad y consistencia de los datos

Uno de los principales resultados esperados es el incremento de la integridad referencial y semántica de los registros en las bases de datos. Al establecer validaciones automáticas directamente en los modelos de Django —complementadas con validadores personalizados y pruebas unitarias— se podrá garantizar que los datos ingresados cumplen con las reglas de negocio y restricciones establecidas por la institución.

Este tipo de validación se espera que reduzca la presencia de registros incompletos, inconsistentes o duplicados, lo cual es fundamental para sistemas de justicia en donde la precisión y veracidad de los datos impactan directamente en la toma de decisiones jurídicas.

16.2. Automatización del monitoreo y reducción de tareas manuales

Mediante el uso de herramientas como Celery, el sistema permitirá realizar monitoreos periódicos y programados del estado de las bases de datos, sin necesidad de intervención manual. Esta automatización no solo disminuye la carga de trabajo técnico, sino que también incrementa la capacidad de respuesta ante errores, como desconexiones, tiempos de respuesta elevados o caídas del servicio.

Además, se espera que esta automatización se traduzca en una mayor eficiencia operativa, ya que los procesos se ejecutan en segundo plano sin afectar el rendimiento de la aplicación principal, lo cual es ideal en entornos que operan bajo alta demanda o con múltiples usuarios concurrentes.

16.3. Reportes y alertas oportunas

Un sistema de monitoreo eficiente no estaría completo sin la capacidad de generar reportes automáticos y enviar alertas personalizadas ante la detección de errores críticos o inconsistencias en los datos. Con este proyecto, se espera que los administradores del sistema puedan recibir informes periódicos que les permitan tomar decisiones fundamentadas con base en evidencia técnica.

Estos reportes no solo contribuirán a la rendición de cuentas, sino que también servirán como insumo para auditorías internas, diagnósticos de rendimiento y estrategias de mejora continua dentro del marco institucional.

16.4. Validación funcional de interfaces con Selenium IDE

El uso de Selenium IDE permitirá llevar a cabo pruebas automatizadas sobre la interfaz de usuario de las aplicaciones desarrolladas en Django. A través de esta herramienta se espera lograr:

- Validar que los formularios de captura y consulta se comporten adecuadamente.
- Simular interacciones reales de los usuarios.
- Detectar errores visuales o funcionales antes del despliegue en producción.

Estas pruebas contribuirán a la robustez del sistema, al evitar que cambios en el código afecten la experiencia del usuario o la integridad de los datos. Selenium IDE es especialmente útil para entornos judiciales donde los formularios contienen información sensible y deben operar con absoluta precisión (López, 2021).

16.5. Fortalecimiento de la cultura de calidad y buenas prácticas

Más allá de los beneficios técnicos, uno de los resultados más importantes será el fortalecimiento de una cultura institucional orientada a la calidad de los datos. Al establecer procesos formales de validación, pruebas automatizadas y monitoreo continuo, se fomenta una actitud preventiva y proactiva frente a los errores, lo cual es vital en organismos donde se maneja información sensible o crítica, como el Consejo de la Judicatura Federal (CJF).

Además, se espera que este sistema pueda ser replicado o adaptado en otras sedes del PJF, contribuyendo así a una estrategia de transformación digital que garantice transparencia, legalidad y eficiencia.

17. Conclusiones

A lo largo del desarrollo de este proyecto, se abordó una problemática frecuente pero muchas veces desatendida en entornos institucionales: la falta de monitoreo continuo y validación estructurada en las bases de datos. En sistemas que manejan información judicial o administrativa —como los

que operan en el Edificio Sede de Apizaco del Poder Judicial de la Federación—, la integridad de los datos no es solo una cuestión técnica, sino una responsabilidad legal y ética.

El uso del framework Django demostró ser una elección acertada, ya que facilitó la implementación de reglas de validación desde el nivel de los modelos, promoviendo una estructura de desarrollo coherente, segura y mantenible. La integración de herramientas como Celery permitió delegar tareas críticas de forma asíncrona, mejorando el rendimiento general del sistema sin comprometer la disponibilidad. A su vez, la incorporación de Selenium IDE brindó la posibilidad de realizar pruebas funcionales automatizadas sobre las interfaces, asegurando que los flujos de interacción del usuario no introduzcan errores involuntarios que afecten la base de datos.

Uno de los logros más importantes fue sentar las bases para una cultura institucional orientada a la calidad de los datos. No solo se diseñó una solución tecnológica, sino que también se propusieron prácticas sostenibles que pueden ser replicadas en otros espacios del Consejo de la Judicatura Federal. Este enfoque no solo resuelve problemas inmediatos, sino que anticipa posibles fallos futuros, lo cual es crucial en contextos donde la pérdida o alteración de datos puede tener consecuencias legales serias.

En síntesis, este proyecto no se limita a una solución técnica; representa una oportunidad para fortalecer la confianza institucional, fomentar la transparencia y consolidar una infraestructura tecnológica sólida y confiable.

18. Recomendaciones

Implementar el sistema en ambientes productivos con acompañamiento técnico

Aunque el sistema fue probado y validado en un entorno controlado, su despliegue en un ambiente real debe hacerse de forma gradual, con monitoreo continuo, para ajustar detalles técnicos y garantizar estabilidad.

Capacitar al personal técnico-administrativo

Es recomendable brindar talleres de capacitación básica sobre el funcionamiento del sistema de monitoreo y validación, así como sobre las buenas prácticas en el uso del ORM de Django, la administración de tareas con Celery y la lectura de reportes generados automáticamente.

Establecer políticas internas de calidad de datos

Para garantizar que la solución tecnológica sea efectiva a largo plazo, es fundamental establecer normas y procedimientos internos que promuevan el uso correcto del sistema, así como la validación periódica de datos sensibles.

Ampliar el alcance del sistema a otros módulos del CJF

Dado que el sistema es modular y escalable, se recomienda su adaptación y expansión a otros procesos del CJF, como control de expedientes digitales, monitoreo de solicitudes, o integración con sistemas de consulta pública, manteniendo siempre el enfoque en la seguridad y la legalidad de la información.

Actualizar periódicamente las herramientas utilizadas

El ecosistema de tecnologías como Django, Celery y Selenium evoluciona constantemente. Por lo tanto, se sugiere mantener actualizado el stack tecnológico y revisar regularmente las dependencias para evitar vulnerabilidades o incompatibilidades.

Documentar todos los procesos y versiones del sistema. Un sistema es tan confiable como la claridad de su documentación. La generación y mantenimiento de documentación técnica actualizada facilitará futuras modificaciones, migraciones o auditorías.

Referencias

Cattell, R. (2011). Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, 39(4), 12–27.

<https://doi.org/10.1145/1978915.1978919>

Cifuentes, C., Hernández, R., & Perdomo, L. (2021). Evaluación de frameworks de desarrollo para la calidad del software. *Revista Universidad y Sociedad*, 13(5), 194–200. Recuperado de https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-50062021000500085

Organización Internacional de Normalización. (2023). ISO/IEC 27001:2022 – Information security, cybersecurity and privacy protection. Recuperado de <https://www.iso.org/standard/82875.html>

Python Software Foundation. (2024). Django documentation. Recuperado de <https://docs.djangoproject.com>

Ask Solem. (2014). Celery documentation (4th ed.). Recuperado de <https://docs.celeryq.dev/en/stable/>

Barrett, J. (2008). Selenium testing tools cookbook. Packt Publishing. Recuperado de <https://books.google.com.mx/books?id=qksgEAAQBAJ>

Brunet, L., Nguyen, T., & Larman, C. (2024). Automating Software Quality Assurance in Agile Teams. *IEEE Software*, 41(1), 40–48. <https://ieeexplore.ieee.org/document/10397687>

Fernández, A., & Gómez, M. (2016). Scrum: una metodología ágil para la gestión de proyectos. Revista Ibérica de Sistemas y Tecnologías de Información, (17), 21–34. Recuperado de https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-50062021000500085

Patel, R., Kumar, A., & Sinha, V. (2023). Distributed Task Scheduling in Microservices using Celery and Redis. International Journal of Computer Applications, 185(14), 10–16. <https://doi.org/10.5555/1747137.1747166>

Schwaber, K., & Sutherland, J. (2020). The Scrum Guide: The Definitive Guide to Scrum. Scrum.org. Recuperado de <https://scrumguides.org>

INAI. (2017). Ley General de Protección de Datos Personales en Posesión de Sujetos Obligados. Instituto Nacional de Transparencia, Acceso a la Información y Protección de Datos Personales. Recuperado de <https://home.inai.org.mx/>

ISO. (2013). ISO/IEC 27001:2013 – Information technology – Security techniques – Information security management systems – Requirements. International Organization for Standardization.

Pinkham, K. (2022). Django for Professionals: Production websites with Python & Django. Independently Published. Recuperado de <https://books.google.com.mx/books?id=qksgEAAAQBAJ>