Imagen de la pantalla de un computador portátil

Descripción generada automáticamente con confianza media

PROYECTO DE TITULO

PROYECTO DE TITULO

Título de Proyecto: *sistema web de RRHH*

MÓDULO: *PROYECTO DE TITULACIÓN/TÉCNICO EN INFORMÁTICA-100-CED*

Bimestre/Año: 2025

Semana: FINAL

Docente: Nicolas Sotelo

Estudiante: **Jonathan Solano Freire**

**Programa en ejecución** [**https://sackserver.ddns.net/**](https://sackserver.ddns.net/)

**PERFILES**

**-Administrador**

**Correo:** [**rrhh@correo.com**](mailto:rrhh@correo.com)

**Pass: admin123**

**-Usuario/ Trabajador**

**Correo:** [amartinez@correo.com](mailto:amartinez@correo.com)

**Pass: 1234**

Índice

1 Capítulo I: Introducción 5

1.1 Contexto y relevancia del proyecto 5

2 Capítulo II: Antecedentes del Proyecto 6

2.1 Planteamiento del problema 6

2.2 Justificación del proyecto 6

2.3 Propuesta de solución 6

2.4 Propuesta de valor 6

3 Capítulo III: Objetivos y Alcance 7

3.1 Objetivo general (1 objetivo general, aplicando metodología SMART) 7

3.2 Objetivos específicos (al menos 5 objetivos específicos, aplicando metodología SMART) 7

3.3 Límites y alcance del proyecto 7

4 Capítulo IV: Análisis del Proyecto 8

4.1 Requerimientos del sistema (funcionales y no funcionales) 8

4.2 Análisis de factibilidad 9

4.2.1 Factibilidad técnica mínima y óptima 9

4.2.2 Factibilidad operativa 10

4.3 Planificación del proyecto 10

4.3.1 Carta Gantt 11

5 Capítulo V: Diseño de Sistemas y Modelado 11

5.1 Tipos de bases de datos (SQL vs NoSQL) y sus aplicaciones. 11

5.2 Diagramas de flujo para representar procesos. 13

5.3 Diagramas BPMN para modelar procesos de negocio relacionados con la solución propuesta. 14

5.4 Árbol funcional y prototipo de interfaz de usuario (mockups aplicando principios de UX). 15

5.5 Metodologías de desarrollo de software. 21

6 Capítulo VI: Modelo de datos 22

6.1 Fundamento y elección del modelo de datos 22

6.2 Modelado de la base de datos 23

6.3 Diagrama Entidad-Relación (DER)- Normalización 24

6.4 Diccionario de Datos 25

7 Capitulo VII Desarrollo e implementación del sistema 25

7.1 Arquitectura del sistema en el contexto de Big Data 25

7.2 Seguridad en Big Data 26

7.3 API REST 27

7.4 Estructuración de Estilos Web 27

7.5 Eventos Web Dinámicos utilizando Tecnologías Open Source 27

8 Capítulo VIII: Plan de Prueba 28

8.1.1 Introducción del Plan de Prueba 28

8.1.2 Objetivo General del Plan de Pruebas 28

8.1.3 Alcance del Plan de Pruebas 28

8.1.4 Propósito del Plan de Pruebas 29

8.2.1 Características que serán probadas 29

8.2.2 Características que no serán probadas 30

8.2.3 Criterios de Aprobación y Fallo 31

8.2.4 Criterios de Suspensión y Reanudación 32

8.2.5 Casos de Prueba y Tareas (Pruebas Manuales) 33

8.2.6 Evidencias de Ejecución 34

9 Capítulo IX: Análisis de Resultados y Verificación de Objetivos 40

9.1 Análisis de Resultados 40

9.2 Verificación de Objetivos 41

10 Capítulo X: Conclusión y Reflexión 43

Referencias 44

**1 capítulo I: Introducción**

**1.1 Contexto y relevancia del proyecto**

Actualmente, los departamentos de recursos humanos en pequeñas y medianas empresas enfrentan dificultades para organizar y gestionar solicitudes de permisos, documentos laborales y consultas de los trabajadores. Muchas veces estos procesos son manuales, lo que genera demoras, errores administrativos y falta de trazabilidad en la información.

Este proyecto propone el desarrollo de un sistema web de gestión de recursos humanos que permita a los trabajadores realizar solicitudes de permisos y descargar certificados laborales, y al personal de RRHH revisar, gestionar y almacenar estos datos de forma eficiente.

Se utilizó una arquitectura cliente-servidor con tecnologías open source y una base de datos relacional. El sistema incorpora funcionalidades básicas como autenticación, gestión de usuarios, solicitudes y certificados, además de un módulo de visualización de liquidaciones. Todo esto aplicando buenas prácticas en desarrollo web, validación de formularios y control de accesos.

**2 capitulo II: Antecedentes del Proyecto**

**2.1 Planteamiento del problema**

En muchas organizaciones, el proceso para que un trabajador solicite permisos, acceda a sus certificados laborales o revise sus documentos es poco eficiente. En general, requiere una gestión manual por parte del departamento de RRHH, lo cual genera una carga administrativa elevada y limita la autonomía del trabajador.

Además, el historial de solicitudes, certificados o liquidaciones no siempre se encuentra digitalizado ni centralizado, dificultando la trazabilidad y seguimiento. Esta situación representa una barrera a la hora de generar estadísticas o visualizar datos agrupados de forma eficiente.

**2.2 Justificación del proyecto**

El desarrollo de este sistema nace de la necesidad de digitalizar y automatizar tareas que hoy en día siguen siendo manuales y repetitivas. La idea es mejorar la comunicación entre los trabajadores y el área de RRHH, facilitar el acceso a la información laboral de forma clara y segura, y además reducir los tiempos de respuesta en la gestión diaria.

Para llevarlo a cabo, se aplicarán varios de los conocimientos aprendidos durante la carrera, como el uso de bases de datos relacionales, diseño de interfaces web, manejo de formularios, creación de APIs REST, almacenamiento de archivos y control de sesiones de usuario.

**2.3 Propuesta de solución**

El sistema propuesto permite automatizar todo el proceso relacionado con solicitudes de permisos, emisión de certificados y manejo de liquidaciones de sueldo. Desde el panel web, el área de RRHH puede registrar trabajadores, revisar y aprobar solicitudes, y subir documentos en PDF. Por su parte, los trabajadores pueden gestionar sus permisos y descargar sus documentos laborales de forma sencilla y segura, sin depender directamente de terceros.

**2.4 Propuesta de valor**

Esta solución ayuda a las empresas a reducir los tiempos de gestión y mantener un mejor orden de la información. También permite dar mayor independencia a los trabajadores, quienes pueden acceder a sus documentos de forma rápida y sin intermediarios. Al estar construido con herramientas libres y con los conocimientos adquiridos en la carrera, su implementación resulta accesible y fácil de mantener.

**3 capitulo III: Objetivos y Alcance**

**3.1 Objetivo general**

Diseñar y desarrollar un sistema web funcional para el área de recursos humanos antes del 15 de mayo de 2025, que permita digitalizar solicitudes de permisos, certificados laborales y liquidaciones de sueldo, mejorando la trazabilidad, accesibilidad y eficiencia en la gestión interna.

**3.2 Objetivos específicos**

1. **Centralizar la administración de usuarios y accesos**, implementando una interfaz de inicio de sesión con control de roles diferenciados (trabajador / RRHH) antes del 30 de abril de 2025.

2. **Permitir a los trabajadores enviar solicitudes de permisos laborales automáticamente,** a través de un módulo web en su panel, reduciendo tiempos de gestión, antes del 2 de mayo de 2025.

3. **Administrar certificados laborales digitales**, habilitando su carga en PDF por parte de RRHH y consulta segura por los trabajadores, antes del 7 de mayo de 2025.

4. **Optimizar la entrega de información salarial**, mostrando liquidaciones ordenadas y descargables para los trabajadores, antes del 10 de mayo de 2025.

5. **Reducir los tiempos de gestión de información de trabajadores**, permitiendo a RRHH registrar, editar y eliminar registros desde un panel administrativo funcional antes del 12 de mayo de 2025.

**3.3 Límites y alcance del proyecto**

**Alcance:**

* Desarrollo de un sistema web responsive, accesible desde navegador web moderno.
* Implementación de login seguro con diferenciación de roles (trabajador y RRHH).
* Funcionalidades específicas por perfil de usuario:

 Trabajadores: envío de solicitudes, visualización y descarga de certificados y liquidaciones.

 RRHH: gestión de trabajadores, revisión de solicitudes, carga de documentos PDF.

**Límites:**

* No contempla módulos de asistencia, evaluaciones de desempeño ni notificaciones por correo.
* No incluye notificaciones automáticas por correo ni módulos de mensajería interna.
* Uso interno únicamente para trabajadores y personal RRHH, manteniendo un entorno controlado y seguro.

**Consideraciones técnicas:**

* Desarrollado con tecnologías open source: Node.js, Express, MySQL, HTML, CSS y JavaScript.
* El despliegue podrá realizarse de forma local con soporte DDNS o a través de hosting en servidores cloud como Railway/Render.

**4 capitulo IV: Análisis del Proyecto**

**4.1 Requerimientos del sistema.**

**4.1.1 Requerimientos funcionales.**

Estos son los requisitos relacionados con las funciones que el sistema debe cumplir para responder a las necesidades de trabajadores y del área de RRHH:

* Login con control de acceso por rol.
* Panel personalizado según perfil.
* Envío de solicitudes de permisos.
* Gestión de solicitudes por RRHH (aprobar/rechazar).
* Visualización y descarga de certificados y liquidaciones.
* Administración de trabajadores (registro, edición, eliminación).
* Subida de documentos en PDF (certificados, liquidaciones).

**4.1.2 Requerimientos no funcionales**

Estos requerimientos no están ligados directamente a funciones específicas, pero son esenciales para asegurar la usabilidad, seguridad y calidad general del sistema:

* Compatible con navegadores modernos.
* Sistema responsive para móvil y escritorio.
* Contraseñas cifradas usando bcrypt.
* Organización de archivos por tipo de documento y trabajador.
* Tiempo de respuesta esperado de hasta 3 segundos en red local, y un máximo de 5 segundos cuando el sistema esté desplegado en entorno web, considerando posibles variaciones por conexión, tráfico y recursos del servidor.
* Sistema desarrollado 100% con tecnologías open source.

**4.2 Análisis de factibilidad**

Para llevar a cabo este proyecto, se optó por el uso de tecnologías que fueran accesibles, conocidas y alineadas con los contenidos vistos durante la carrera. Esto permite un desarrollo más ágil, económico y práctico, sin sacrificar funcionalidad.

**Selección de tecnologías y herramientas:**

* El sistema es desarrollado utilizando herramientas de código abierto (open source), como Node.js, Express, MySQL, HTML, CSS y JavaScript puro. Estas tecnologías no requieren licencias pagadas y permiten construir una solución robusta y flexible, adecuada para proyectos académicos y de pequeña escala empresarial.

**Selección de lenguaje de programación:**

* Se trabaja principalmente con JavaScript, tanto en el backend (usando Node.js y Express) como en el frontend (HTML, CSS y JavaScript). Este lenguaje es altamente utilizado en la industria del desarrollo web, es multiplataforma y facilita una rápida implementación de nuevas funcionalidades.

**Selección de Sistema Gestor de Base de Datos (SGBD):**

* Para el manejo de la información se opta por MySQL, una base de datos relacional confiable y de uso extendido. Permite estructurar los datos mediante relaciones entre tablas (usuarios, trabajadores, solicitudes, certificados, liquidaciones y saldos) asegurando integridad y consistencia de la información.

**4.2.1 Factibilidad técnica mínima y óptima**

* **Factibilidad mínima:** El sistema puede ser desplegado localmente en una red interna mediante el uso de un servidor tipo XAMPP y un servicio de DDNS (por ejemplo, No-IP), lo que permite el acceso desde fuera de la red sin costos adicionales.
* **Factibilidad óptima:** Alternativamente, el sistema podría desplegarse en servicios de nube como Railway, Render o Vercel, ofreciendo mayor estabilidad, disponibilidad y facilidad de acceso sin necesidad de infraestructura propia.

Esta flexibilidad en el despliegue permite adaptar el sistema tanto a escenarios de pruebas locales como a ambientes productivos reales.

**4.2.2 Factibilidad operativa**

El sistema está diseñado para que cualquier persona del área de recursos humanos pueda utilizarlo sin necesidad de conocimientos técnicos avanzados. La interfaz gráfica es clara, intuitiva y amigable, organizada según el perfil de acceso del usuario:

* **Trabajador:** acceso a solicitudes, Documentos (certificados, contratos y liquidaciones).
* **RRHH:** acceso a gestión de trabajadores, aprobación de solicitudes y carga de documentos.

Además:

* El sistema garantiza un tiempo de respuesta ágil (menor a 3 segundos en red local) para una experiencia de usuario fluida.
* El acceso controlado por roles mejora la seguridad en el manejo de datos sensibles.
* El desarrollo basado en tecnologías open source minimiza costos y facilita su mantenimiento a largo plazo.
* Se considera que el sistema debe mantener estabilidad durante sesiones de uso prolongadas (mínimo 4 horas de operación continua sin fallos).

En consecuencia, se concluye que la implementación es operativamente viable, accesible para los usuarios finales, y fácilmente adaptable tanto en ambientes locales como en infraestructura de nube si se requiere escalar su uso en el futuro.

**4.3 Planificación del proyecto**

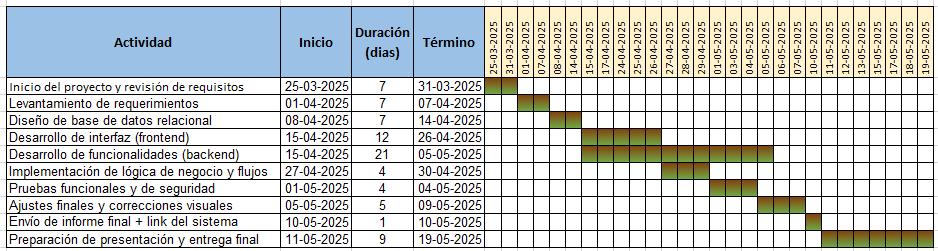
El proyecto fue planificado utilizando una Carta Gantt, que organiza las actividades de manera secuencial, estableciendo fechas específicas de inicio y término para cada tarea. Esto permite gestionar los tiempos de manera eficiente y visualizar claramente el progreso del proyecto.

Cada actividad tiene una duración determinada en días, lo que facilita el control del avance y el cumplimiento de los plazos establecidos.

**Responsable:**

Dado que el desarrollo se realiza de manera individual, **Jonathan Solano** es el responsable de la ejecución de todas las actividades planificadas.

**4.3.1Carta Gantt**

****

**5 capitulo V: Diseño de Sistema y Modelado**

**5.1 Tipos de bases de datos (SQL vs NoSQL) y sus aplicaciones.**

Para el desarrollo del sistema, se optó por una base de datos relacional (MySQL), ya que permite estructurar y mantener relaciones entre diferentes tipos de datos (usuarios, trabajadores, solicitudes, certificados, liquidaciones y saldos), garantizando la integridad referencial y facilitando consultas complejas.

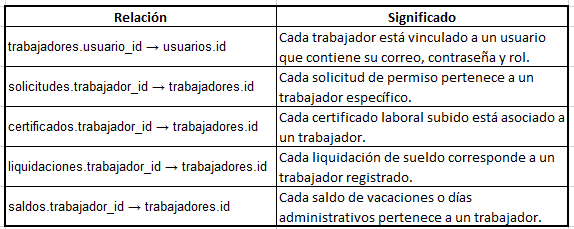
Las bases de datos NoSQL fueron descartadas en esta fase, dado que el sistema maneja información altamente estructurada, donde las relaciones son críticas.

**Modelo preliminar relacional:**

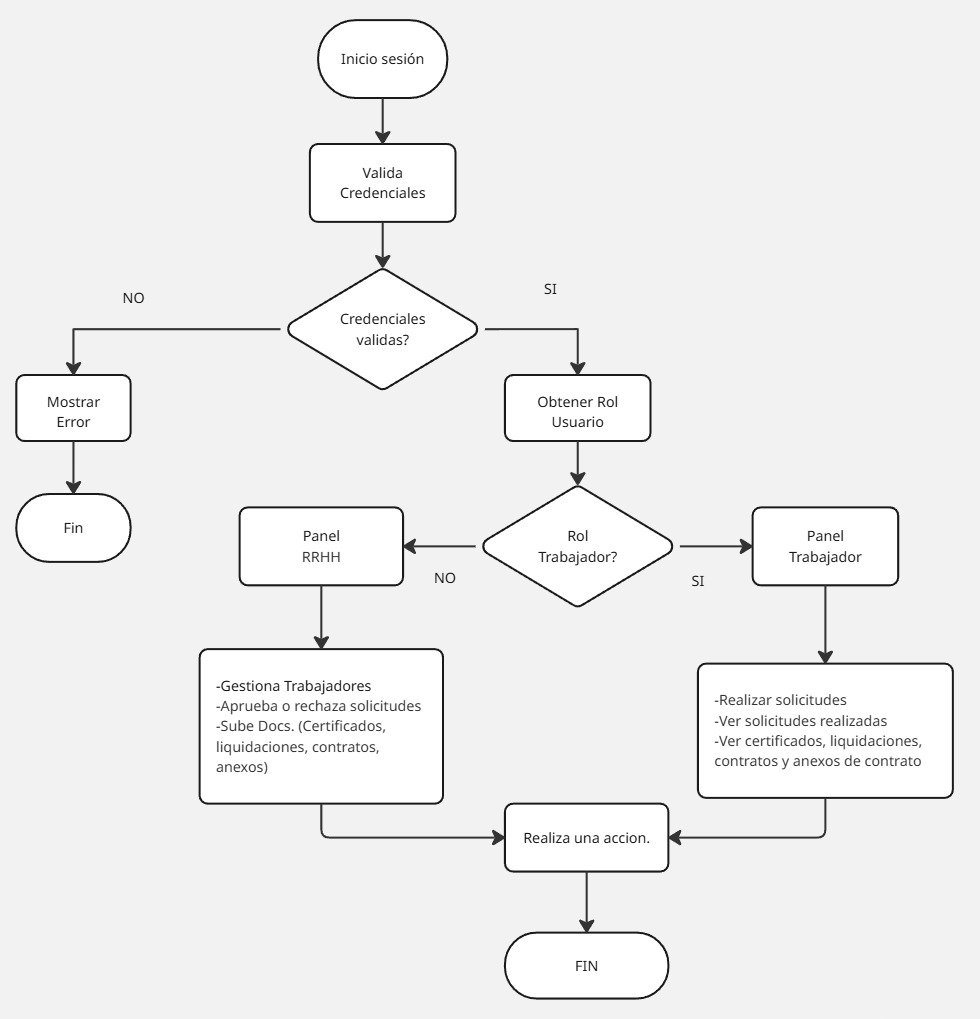
**Tablas principales:**

* Usuarios
* trabajadores
* solicitudes
* certificados
* liquidaciones
* saldos

**Relaciones:**



**5.2 Diagrama de flujo**

****

**Proceso del Trabajador:**

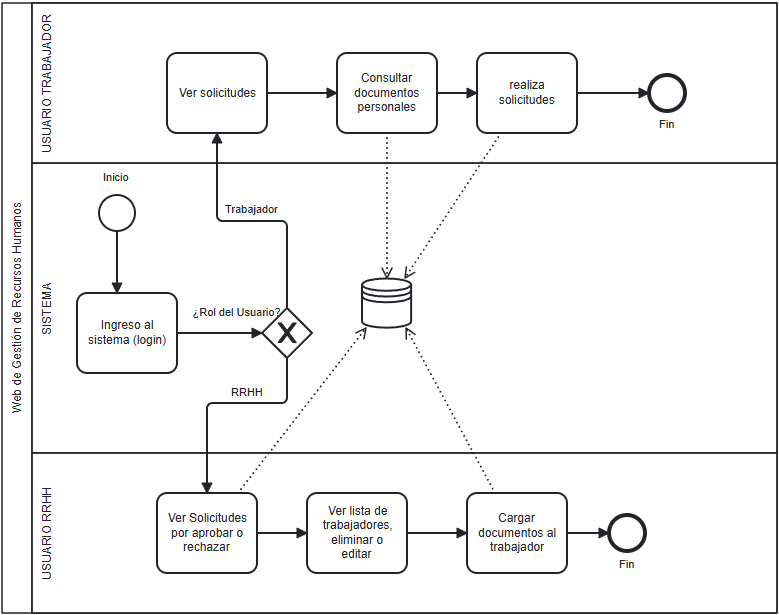
* Inicia sesión → Accede a su panel → Ingresa nueva solicitud de permiso → Envía → Visualiza el estado de su solicitud→ Visualiza documentación→ Descarga documentación→ Fin

**Proceso del RRHH:**

* Inicia sesión → Accede al panel RRHH → Visualiza solicitudes pendientes → Aprueba o rechaza → Actualiza el estado del trabajador→ Sube documentación del trabajador→ Fin.

**5.3 Diagrama BPMN**

A continuación, se presenta el diagrama BPMN del sistema de gestión de recursos humanos, que modela los flujos de acciones principales de los dos tipos de usuarios: Trabajador y RRHH.

****

**Actores involucrados:**

* **Trabajador:** Usuario que realiza solicitudes de permisos, consulta documentos personales y visualiza el estado de sus solicitudes.
* **RRHH:** Usuario que gestiona las solicitudes de los trabajadores, administra documentos laborales y mantiene el registro de trabajadores.

**Descripción del proceso:**

* Inicio del proceso: El usuario accede al sistema mediante el módulo de autenticación (login).
* Determinación de rol: Una vez autenticado, el sistema identifica si el usuario pertenece al rol Trabajador o RRHH.

**Si el usuario es Trabajador:**

* Puede ver sus solicitudes realizadas previamente.
* Puede consultar documentos personales como certificados, contratos y liquidaciones.
* Puede realizar nuevas solicitudes de permisos laborales.

**Si el usuario es RRHH:**

* Puede ver las solicitudes pendientes para aprobar o rechazar.
* Puede ver la lista de trabajadores, editarlos o eliminarlos según sea necesario.
* Puede cargar nuevos documentos asociados a trabajadores (certificados o liquidaciones).

**Fin del proceso:**

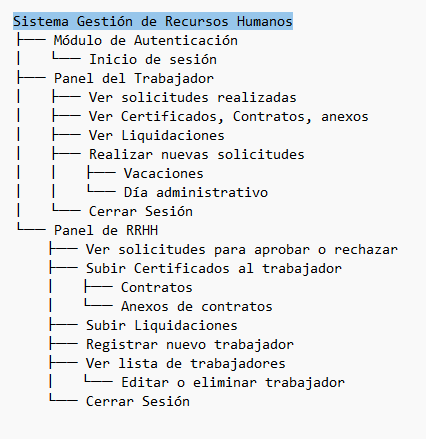
* Cada actor finaliza su interacción según las funciones utilizadas dentro del sistema.

Este diagrama permite visualizar de manera estructurada los flujos principales, asegurando una correcta separación de responsabilidades y garantizando la trazabilidad de las acciones realizadas en la plataforma.

**5.4 Árbol Funcional y prototipo de interfaz de usuario (Mockups aplicando principios de ux)**

El árbol funcional presentado a continuación describe la estructura jerárquica del sistema web de gestión de recursos humanos, diferenciando claramente los módulos y funcionalidades disponibles para cada tipo de usuario (trabajador o RRHH).

Esta estructura permite comprender la lógica de navegación propuesta, asegurando una experiencia de usuario (UX) ordenada, intuitiva y adaptada al perfil de cada usuario.

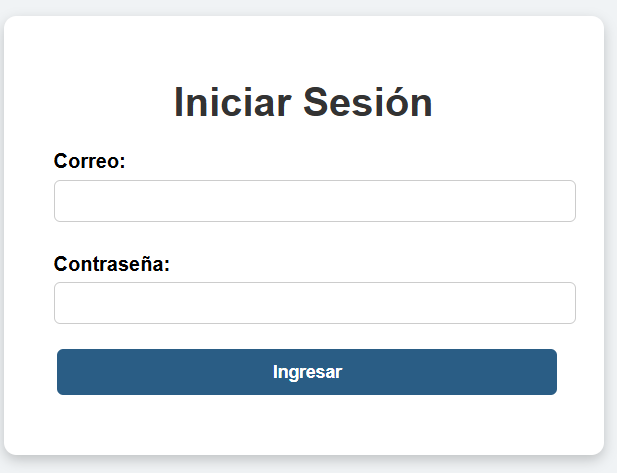
****

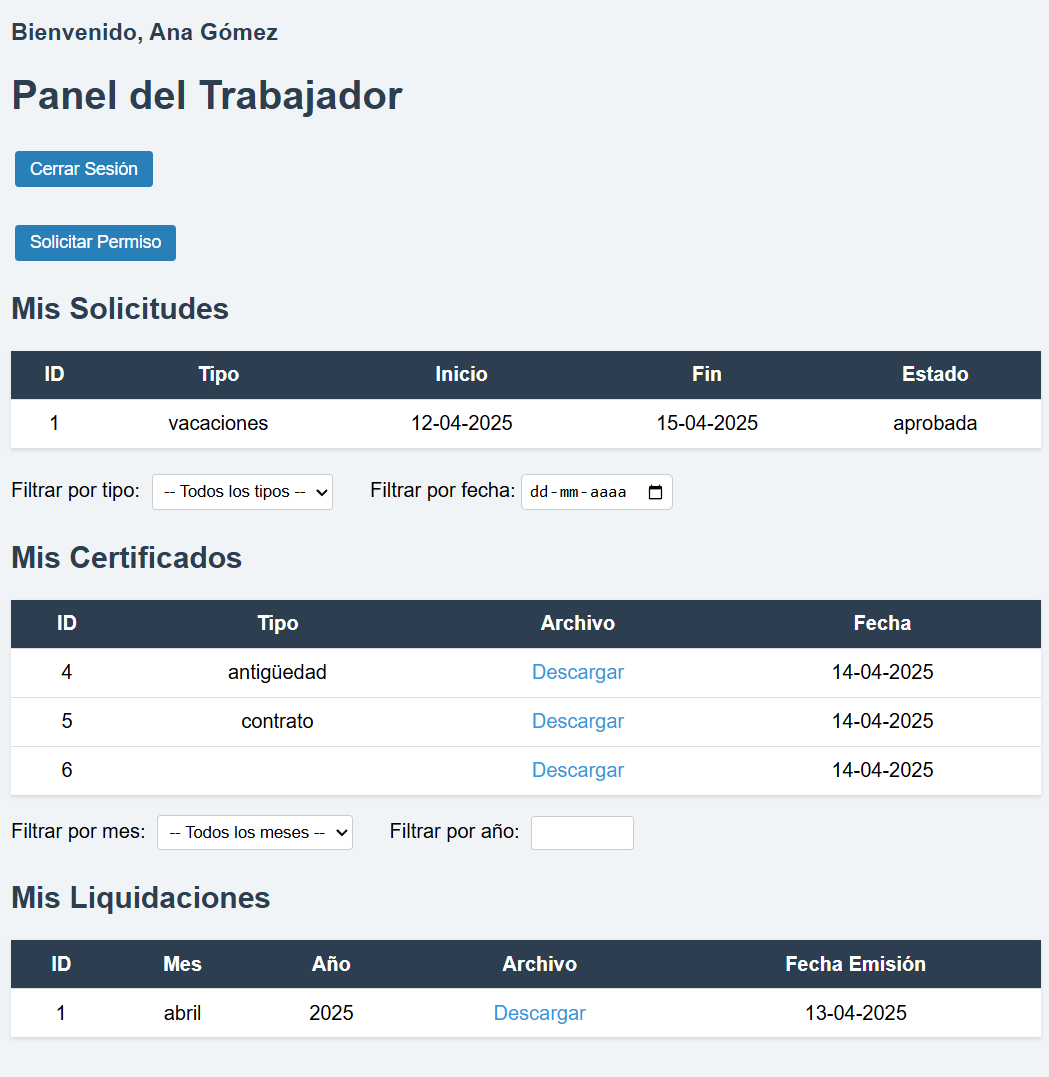
**Descripción general del árbol funcional.**

* **Módulo de Autenticación:** Punto de inicio del sistema, donde los usuarios deben autenticarse para acceder a su panel correspondiente.
* **Panel del Trabajador:**
* Visualizar solicitudes realizadas y su estado.
* Ver certificados, contratos u otros documentos anexos.
* Consultar liquidaciones de sueldo ordenadas por fecha.
* Crear nuevas solicitudes de permisos, diferenciadas en:
* Vacaciones
* Días administrativos
* Cerrar sesión de forma segura.
* **Panel del RRHH:**
* Visualizar solicitudes enviadas por los trabajadores y aprobar o rechazar.
* Subir certificados laborales vinculados a trabajadores.
* Subir liquidaciones de sueldo asociadas a cada trabajador.
* **Administrar trabajadores:**
* Registrar nuevos trabajadores.
* Ver la lista actual de trabajadores registrados.
* Editar o eliminar trabajadores existentes.
* Cerrar sesión de forma segura.

Cada panel muestra únicamente las funcionalidades disponibles según el tipo de usuario autenticado, siguiendo los principios de usabilidad y seguridad.

**5.4.2 Mockups**

****

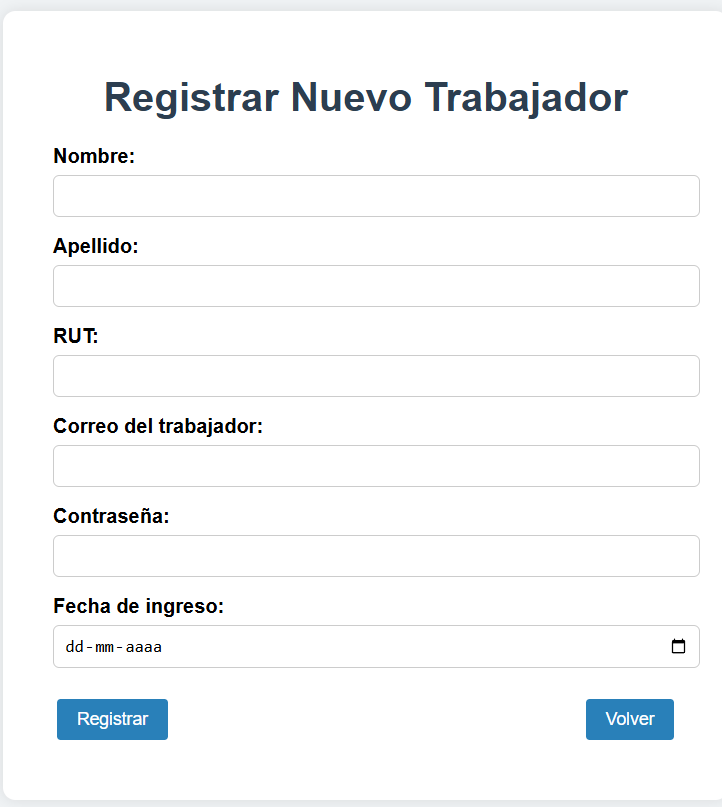
****

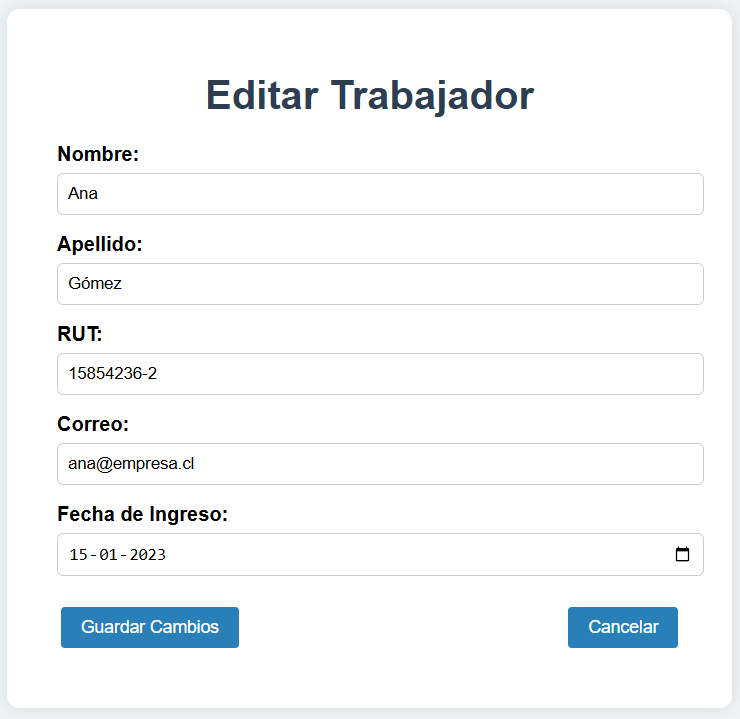




****

****

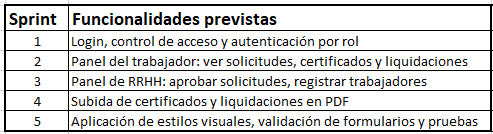
****

**5.5 Metodologia de desarrollo de sofware.**

Para el desarrollo del sistema de gestión de recursos humanos se decidió aplicar una metodología ágil, adaptada al contexto académico y basada en los principios de Scrum. Este enfoque permite organizar el trabajo en sprints, enfocándose primero en las funcionalidades más importantes y avanzando de forma iterativa y progresiva a lo largo del proyecto.

Aunque el desarrollo se realiza de manera individual, la estructura propuesta facilita entregas funcionales al finalizar cada sprint, permitiendo la revisión, la mejora continua y la rápida adaptación a cambios sugeridos durante el proceso.

**Planificación por etapas de desarrollo**



**Justificación del enfoque:**

* Permite organizar el desarrollo del sistema en partes lógicas, manejables y entregables.
* Facilita la detección temprana de errores y la aplicación de mejoras a lo largo del proceso.
* Se adapta bien a los tiempos académicos, permitiendo mostrar avances funcionales en entregas parciales.
* Mejora la planificación individual, asegurando el cumplimiento de plazos establecidos.
* Fomenta la flexibilidad ante cambios o correcciones requeridas durante el desarrollo.

**¿Por qué no se utilizó una metodología clásica (como Cascada)?**

Las metodologías tradicionales como Cascada requieren definir completamente todos los requerimientos, diseño y etapas desde el inicio, sin permitir modificaciones durante la ejecución. Esto no es adecuado para el contexto académico donde el proyecto debe adaptarse a nuevas observaciones y correcciones parciales.

Scrum, en cambio, permite avanzar por entregas funcionales y ajustar la planificación según las necesidades que vayan surgiendo, haciendo que el proceso de desarrollo sea más dinámico y efectivo.

**Herramientas previstas:**

El conjunto de herramientas seleccionadas respalda la implementación ágil del proyecto, permitiendo realizar entregas incrementales rápidas y aplicar correcciones inmediatas al final de cada sprint:

* Visual Studio Code (VS Code): Editor de código que facilita la gestión simultánea de múltiples archivos, integración con control de versiones y pruebas rápidas durante cada sprint.
* Servidor local para pruebas (XAMPP o equivalente): Permite simular un entorno real de ejecución, necesario para validar las funcionalidades desarrolladas en cada sprint antes de su entrega.
* Gestor de base de datos (phpMyAdmin + MySQL): Herramientas que permiten crear, administrar y modificar las bases de datos de forma ágil, adaptando el modelo de datos según los avances de cada sprint.

Estas herramientas permiten implementar los **principios ágiles de Scrum** en un entorno académico sencillo, donde la prioridad es la entrega funcional y la documentación exhaustiva.

**6 Capítulo VI: Modelo de datos**

**6.1 Fundamento y elección del modelo de datos**

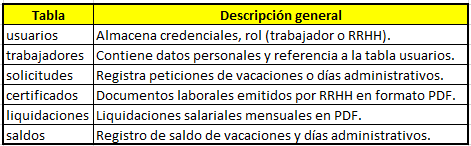
Para el desarrollo del sistema de Gestión de Recursos Humanos se optó por un modelo de base de datos **relacional (SQL),** implementado con **MySQL,** ya que permite garantizar la integridad referencial entre las entidades y una estructura de datos clara, normalizada y transaccional. Esta elección se fundamenta en la naturaleza **altamente estructurada** de los datos gestionados por el sistema, como usuarios, trabajadores, solicitudes, certificados, liquidaciones y saldos.

El uso de bases relacionales también permite aplicar reglas de integridad referencial, validaciones estrictas y normalización hasta tercera forma normal (3FN) o más, reduciendo así redundancias y errores humanos.

Las bases de datos **NoSQL** se descartaron en esta etapa, ya que, si bien son útiles para datos no estructurados, este sistema requiere relaciones claras y transacciones atómicas que se gestionan de mejor forma en un entorno SQL.

**6.2 Modelado de la base de datos**

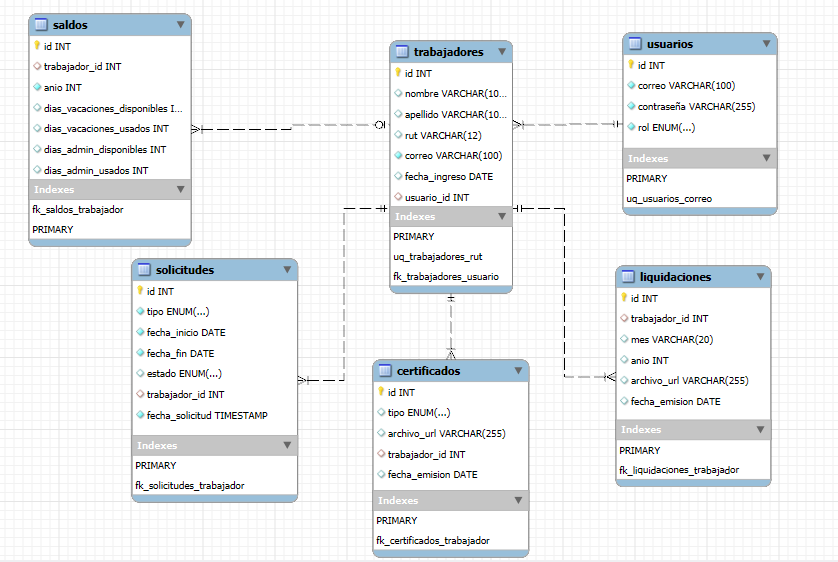
El sistema está compuesto por seis tablas principales: **usuarios**, **trabajadores**, **solicitudes**, **certificados**, **liquidaciones** y **saldos**, todas organizadas y normalizadas hasta la **Tercera Forma Normal (3FN)** para garantizar consistencia, eliminar redundancias y facilitar la escalabilidad. A continuación, se presenta una descripción general:



La relación principal es **1:N** entre trabajadores y sus solicitudes, certificados, liquidaciones y saldos, garantizando integridad en la trazabilidad de los registros.

**6.3 Diagrama Relacional:**

Imagen del modelo relacional creado en MySQL Workbench

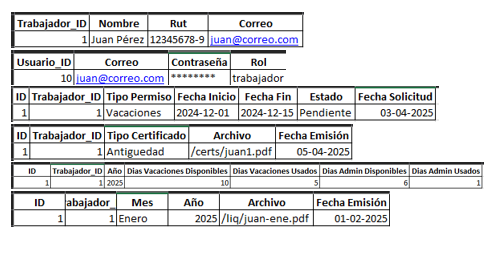


**Normalización:**

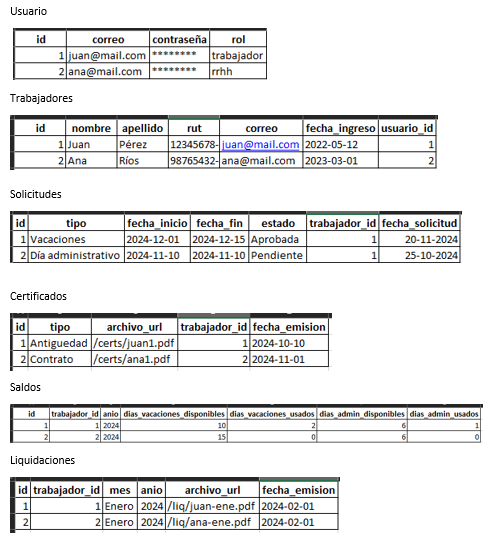
* **Primera Forma Normal (1FN):** Cada campo contiene datos atómicos sin repetición de grupos.



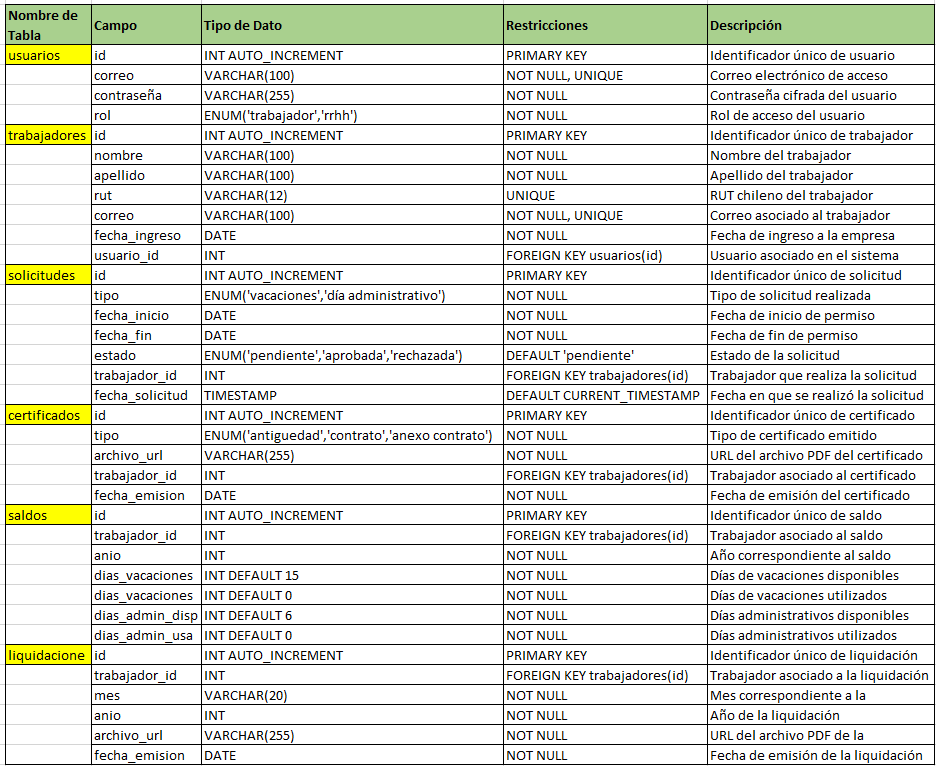
* **Segunda Forma Normal (2FN):** Todos los campos no clave dependen completamente de la clave primaria.



* **Tercera Forma Normal (3FN):** Eliminación de dependencias transitivas entre campos no clave.



**6.4 Diccionario de Datos:**



**7 capítulo VII: Desarrollo e implementación del sistema**

**7.1 Arquitectura del sistema en el contexto de Big Data**

El sistema está basado en una arquitectura **cliente-servidor**, compuesta por:

* **Frontend:** HTML5, CSS3 personalizado y JavaScript puro.
* **Backend:** Node.js con Express.js, desarrollando una API REST modular.
* **Base de datos:** MySQL como sistema gestor de bases de datos relacional.
* **Almacenamiento de documentos**: Archivos PDF guardados en carpetas del servidor

Esta organización ayuda a mantener el código ordenado, permite trabajar de forma independiente en las distintas partes del sistema y deja abierta la posibilidad de ampliarlo o adaptarlo a futuro si fuera necesario.

**7.2 Seguridad en Big Data**

Actualmente, Dentro de las medidas de seguridad básicas implementadas en esta etapa están:

* Control de acceso basado en roles, diferenciando entre usuarios de tipo trabajador y RRHH, restringiendo el acceso a funcionalidades sensibles según el perfil.
* Validación de credenciales mediante verificación de correo y contraseña en el proceso de inicio de sesión.
* Validación de campos obligatorios en formularios y rutas backend, para asegurar que los datos requeridos sean enviados correctamente y prevenir manipulaciones.
* Restricción de acciones por tipo de usuario, evitando accesos indebidos a paneles o funciones que no correspondan a su rol.
* Implementación de HTTPS mediante certificado SSL configurado en el servidor Nginx sobre Ubuntu, lo que asegura la transmisión cifrada de datos entre cliente y servidor.

**Como mejora futura se contempla:**

* Cifrado de contraseñas utilizando bcrypt en la base de datos para proteger credenciales de acceso.
* Aplicar límites de peticiones por IP mediante middleware (express-rate-limit) en el backend para reducir la exposición a ataques de denegación de servicio (DoS).
* Configuración de filtros IP personalizados y la eventual integración con Cloudflare, con el fin de añadir una capa adicional de defensa a nivel de red.
* Validaciones avanzadas de entrada de datos, reforzando la protección contra inyección SQL mediante sanitización de inputs, validaciones de esquema y eventualmente el uso de procedimientos almacenados.
* Implementación de tokens de sesión serán generados al momento del inicio de sesión y enviados al cliente, donde serán almacenados en el almacenamiento local del navegador (localStorage). Estos tokens permitirán validar la autenticación del usuario en cada solicitud, manteniendo su sesión activa mientras sea válida.

**7.3 API REST**

La implementación de una API REST en este proyecto permite desacoplar el frontend del backend, facilitando el mantenimiento, la escalabilidad y el desarrollo modular. Esta arquitectura favorece la reutilización de endpoints y la integración futura con otras plataformas o sistemas externos.

* **Tecnologías utilizadas:** Node.js y Express.js.
* **Endpoints principales:** /api/login, /api/usuarios, /api/trabajadores, /api/solicitudes, /api/certificados, /api/liquidaciones, /api/saldos.
* **Buenas prácticas aplicadas:**
  + Modularización de rutas y middlewares.
  + Validación básica de datos en solicitudes.
  + Manejo estructurado de errores y respuestas JSON estandarizadas.

**7.4 Estructuración de Estilos Web**

* El sitio web utiliza CSS personalizado.
* Se aplicaron principios de diseño responsivo para asegurar la compatibilidad en dispositivos móviles y de escritorio.
* Se diseñaron estilos específicos para diferentes módulos (login, panel trabajador, panel RRHH, formularios).

**7.5 Eventos Web Dinámicos utilizando Tecnologías Open Source**

* El sistema utiliza **JavaScript** para manejar la interacción del usuario, sin frameworks externos.
* Funcionalidades:
  + El inicio de sesión dinámico.
  + El envío y carga de solicitudes, certificados y liquidaciones.
  + La actualización dinámica de tablas según filtros de tipo, fecha, mes o año.
  + Subida y descarga de documentos desde el navegador.

**8 Capítulo VIII: Plan de Prueba**

**8.1.1 Introducción del Plan de Pruebas**

El presente capítulo tiene como finalidad documentar el plan de pruebas diseñado para evaluar el correcto funcionamiento del sistema web de gestión de recursos humanos desarrollado durante el proyecto. Este sistema contempla funcionalidades esenciales como la autenticación por roles, la gestión de solicitudes de permisos, la administración de trabajadores, y la visualización y descarga de documentos laborales (certificados y liquidaciones).

El proceso de pruebas permite identificar errores, verificar que cada módulo cumpla con los requerimientos definidos y asegurar que la experiencia del usuario sea fluida, estable y confiable. Además, este plan de pruebas busca detectar errores que puedan afectar la estabilidad del sistema o comprometer la seguridad de los datos, anticipándose a posiblesfallos antes de que el sistema entre en funcionamiento.

**8.1.2 Objetivo General del Plan de Pruebas**

El objetivo general de este plan de pruebas es verificar que cada funcionalidad del sistema web de gestión de recursos humanos opere correctamente bajo condiciones normales de uso, garantizando que los trabajadores y el personal de RRHH puedan interactuar con el sistema sin errores, interrupciones ni comportamientos inesperados.

Se busca asegurar que los módulos críticos como el login, la creación y revisión de solicitudes, el registro de trabajadores, la carga de documentos y la consulta de saldos cumplan con los requerimientos funcionales definidos, y que el sistema responda de manera coherente a los datos ingresados, validaciones y acciones de los usuarios.

**8.1.3 Alcance del Plan de Pruebas**

El alcance de este plan de pruebas comprende la verificación de las funcionalidades principales del sistema web de gestión de recursos humanos, tanto para el perfil de trabajador como para el perfil de RRHH. Las pruebas se enfocan en validar el comportamiento de los formularios, la correcta ejecución de las operaciones CRUD (crear, leer, actualizar y eliminar), y la interacción entre el frontend y el backend a través de las rutas definidas en la API REST.

Se incluyen pruebas manuales sobre:

* Inicio de sesión y redirección según el rol.
* Envío y gestión de solicitudes de permisos.
* Registro, edición y eliminación de trabajadores.
* Subida y visualización de certificados y liquidaciones.
* Visualización de saldos y filtros por tipo, fecha o año.

Este plan no contempla pruebas de rendimiento, pruebas automatizadas ni pruebas en condiciones extremas, ya que el enfoque es asegurar el funcionamiento general del sistema en un entorno controlado y con condiciones reales de uso.

**8.1.4 Propósito del Plan de Pruebas**

El propósito principal de este plan de pruebas es asegurar que el sistema web de gestión de recursos humanos funcione de manera estable, segura y conforme a los requerimientos funcionales definidos, evitando errores que puedan interrumpir el servicio o comprometer la experiencia del usuario final.

A través de la ejecución de pruebas manuales, se busca identificar fallos en la lógica del sistema, errores en formularios, problemas de validación o situaciones que puedan afectar la disponibilidad de las funcionalidades críticas. Esto permite mejorar la calidad del software antes de su uso final, brindando una plataforma confiable tanto para trabajadores como para el personal del área de RRHH.

Asimismo, este plan sirve como referencia para documentar el comportamiento esperado de cada módulo, establecer un control de calidad y facilitar futuras mejoras o mantenimientos.

**8.2.1 Características que serán probadas**

El plan contempla la evaluación manual de las siguientes funcionalidades clave del sistema:

* Inicio de sesión con validación de credenciales y redirección según el rol del usuario (trabajador o RRHH).
* Envío de solicitudes de permiso por parte de los trabajadores, con validación de fechas y tipo de solicitud.
* Visualización de solicitudes enviadas, incluyendo estado actualizado (pendiente, aprobada, rechazada).
* Aprobación o rechazo de solicitudes por parte del personal de RRHH desde su panel administrativo.
* Registro de nuevos trabajadores, incluyendo validación de campos y creación de usuario y saldo inicial en dias.
* Edición y eliminación de trabajadores desde el panel de RRHH.
* Carga de certificados y liquidaciones en formato PDF, asociados correctamente a cada trabajador.
* Visualización y descarga de certificados y liquidaciones por parte del trabajador autenticado.
* Consulta del saldo de días disponibles (vacaciones y administrativos) tanto por parte de RRHH como del trabajador.
* Filtros por tipo, fecha o año en los módulos de certificados, liquidaciones y solicitudes.

Cada una de estas características será validada a través de pruebas manuales, ejecutadas desde la interfaz del sistema en condiciones normales de uso.

**8.2.2 Características que no serán probadas**

Este plan de pruebas se enfoca en validar el funcionamiento general del sistema bajo condiciones normales de uso. Por lo tanto, se excluyen del alcance de las pruebas las siguientes características:

* Visualización del campo de contraseña con íconos tipo “mostrar/ocultar” (no implementado en el sistema actual).
* Pruebas de carga o estrés para evaluar el comportamiento del sistema ante múltiples solicitudes simultáneas.
* Validación de seguridad avanzada como cifrado de contraseñas con bcrypt o protección contra inyección SQL (definido como mejora futura).
* Pruebas automatizadas mediante herramientas como Selenium o Postman (el enfoque es 100% manual).
* Notificaciones por correo electrónico (no contempladas en esta versión del sistema).
* Accesibilidad web bajo normas WAI-ARIA (no considerada en esta etapa del desarrollo). se entienden como un conjunto de estándares que mejoran la accesibilidad de las aplicaciones web, permitiendo que usuarios con discapacidades puedan interactuar con tecnologías de asistencia como lectores de pantalla. Su aplicación futura dependerá del contexto de uso final del sistema.

Estas exclusiones responden a las limitaciones, alcance del proyecto y del tiempo disponible para su desarrollo y validación.

**8.2.3 Criterios de Aprobación y Fallo**

Para cada funcionalidad probada, se establecen los siguientes criterios generales de aprobación y fallo.

**Criterios de Aprobación.**

Una prueba se considera aprobada si cumple con todas las siguientes condiciones:

* El sistema responde correctamente a la acción del usuario.
* Los datos ingresados son procesados y almacenados correctamente en la base de datos.
* La interfaz muestra mensajes adecuados de éxito o validación.
* El contenido generado (documentos, solicitudes, saldos) refleja lo esperado según la acción realizada.
* No se generan errores visibles en la consola del navegador ni en la respuesta del servidor.

**Criterios de Fallo.**

Una prueba se considera fallida si ocurre alguna de las siguientes situaciones:

* El sistema no responde o genera errores al ejecutar una acción.
* Los datos no se guardan correctamente o presentan inconsistencias.
* La interfaz no muestra retroalimentación adecuada (ej. no hay mensajes de error o éxito).
* Se detectan comportamientos inesperados (como navegación incorrecta, archivos no descargables, o datos incompletos).
* La funcionalidad se interrumpe sin justificación (ej. formulario no envía nada o se bloquea).

**8.2.4 Criterios de Suspensión y Reanudación**

**Criterios de Suspensión**

Las pruebas serán suspendidas temporalmente si se presenta alguna de las siguientes condiciones:

* El servidor backend deja de estar disponible o presenta errores críticos que impiden continuar con las pruebas.
* La base de datos no responde o pierde conexión, impidiendo el guardado o consulta de datos.
* Alguna ruta API esencial devuelve errores permanentes (500 o fallos de conexión).
* Cambios inesperados en el entorno de pruebas (corte de red, cambios en configuración local o despliegue).

**Criterios de Reanudación**

Las pruebas podrán retomarse cuando:

* Se haya solucionado el problema técnico y verificado mediante una prueba rápida, como intentar iniciar sesión o consultar una ruta básica del sistema (ej: /api/trabajadores).
* La base de datos esté accesible nuevamente desde el entorno de desarrollo (por ejemplo, mediante conexión desde phpMyAdmin).
* El servidor backend se haya reiniciado correctamente y no arroje errores en consola al ejecutarse.
* Se verifique que las funcionalidades principales vuelven a responder desde el navegador, sin errores visibles.

**8.2.5 Casos de Prueba y Tareas (Pruebas Manuales)**

A continuación se presentan 10 casos de prueba realizados manualmente en el sistema, considerando las funcionalidades más relevantes:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nº | Funcionalidad | Entrada / Acción | Resultado Esperado | Resultado Obtenido |
| 1 | Inicio de sesión correcto | Correo y contraseña válidos | Redirección al panel según el rol | Correcto |
| 2 | Inicio de sesión fallido | Correo válido, contraseña incorrecta | Mensaje de error: “Credenciales incorrectas” | Correcto |
| 3 | Envío de solicitud de permiso | Formulario completado con fechas válidas y tipo seleccionado | Solicitud guardada como “pendiente” | Correcto |
| 4 | Envío de solicitud incompleta | Campo “fecha de fin” vacío | Validación y mensaje de error | Correcto |
| 5 | Aprobación de solicitud | RRHH aprueba solicitud desde el panel | Estado cambia a “aprobada” y se descuenta del saldo | Correcto |
| 6 | Registro de trabajador nuevo | Datos completos ingresados | Trabajador y usuario registrados, con saldo inicial | Correcto |
| 7 | Subida de certificado (PDF) | Archivo PDF, tipo y trabajador seleccionados | Certificado visible en el panel del trabajador | Correcto |
| 8 | Visualización de certificados | Trabajador autenticado accede a su panel | Lista de certificados con opción de descarga | Correcto |
| 9 | Edición de trabajador | Cambiar nombre y correo desde formulario de edición | Datos actualizados correctamente | Correcto |
| 10 | Filtros de certificados /liquidaciones | Selección de tipo o fecha | Resultados filtrados correctamente | Correcto |

**8.2.6 Evidencias de Ejecución**

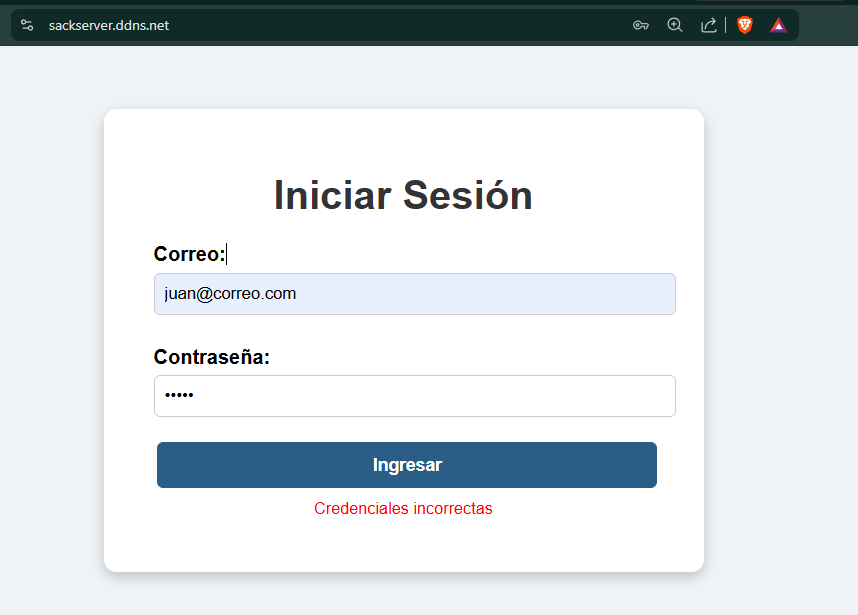
A continuación, se presentan capturas de pantalla que respaldan la ejecución de las pruebas manuales descritas en el apartado anterior. Estas evidencias permiten verificar que las funcionalidades del sistema fueron evaluadas correctamente en un entorno real de uso:

1. Inicio de sesión exitoso



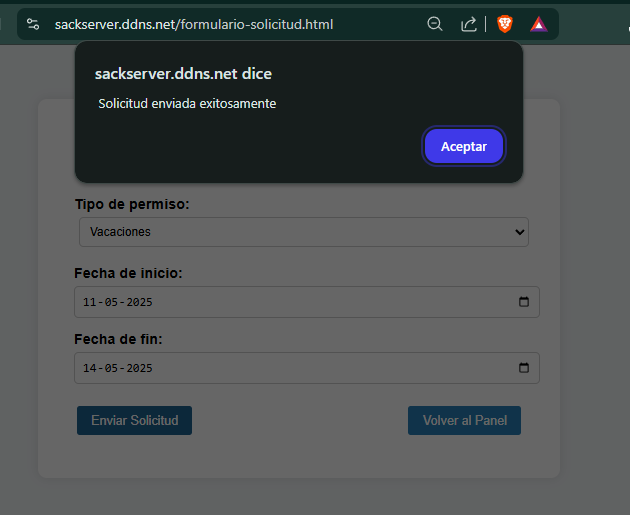
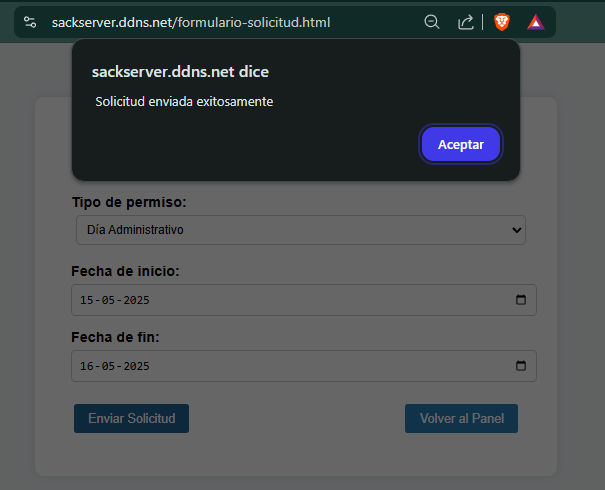
Usuario ingresa credenciales válidas y accede al panel.

2. Error en inicio de sesión



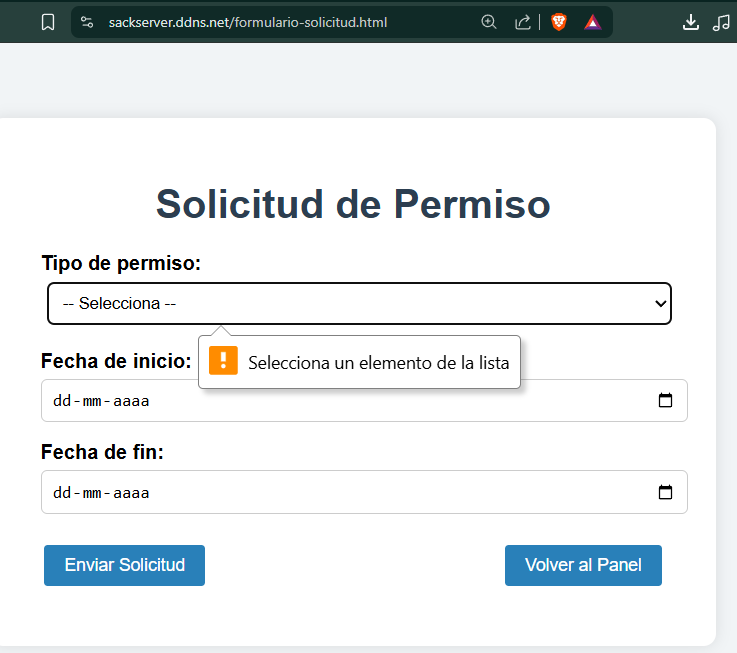
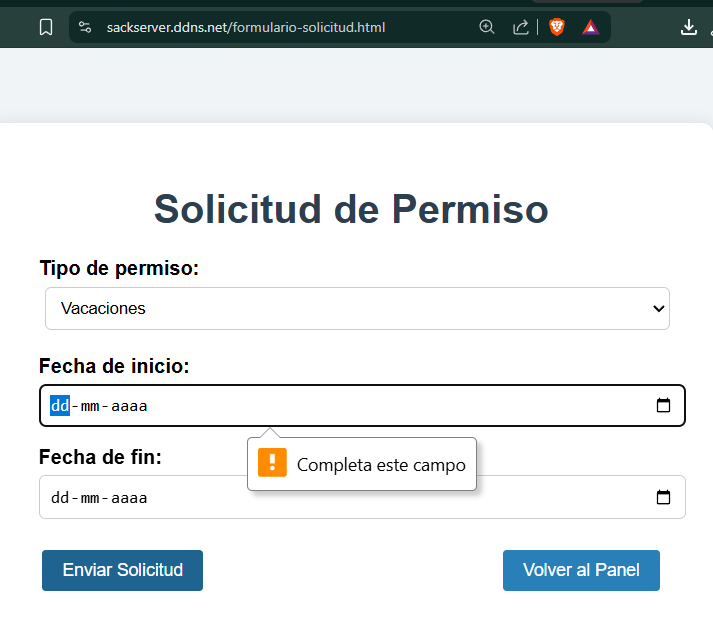
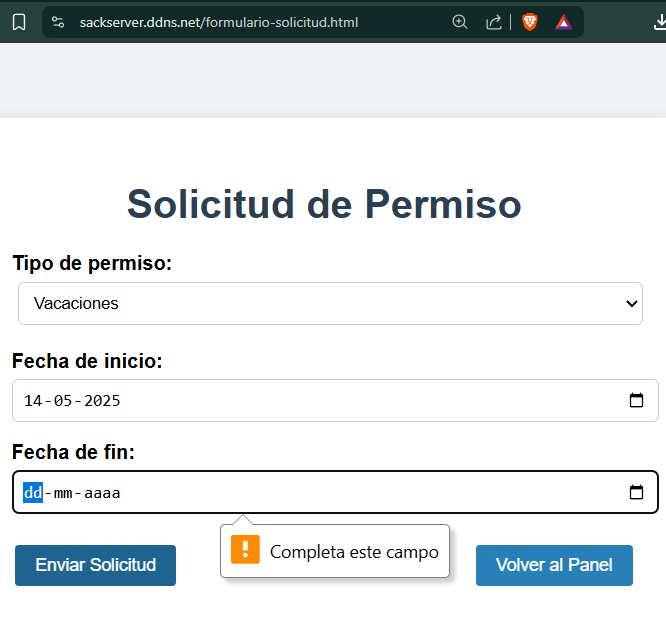
Mensaje de error al ingresar contraseña incorrecta.

3. Envío de solicitud correctamente

Formulario lleno y mensaje de éxito tras envío.

4. Validación de campos vacíos en solicitud

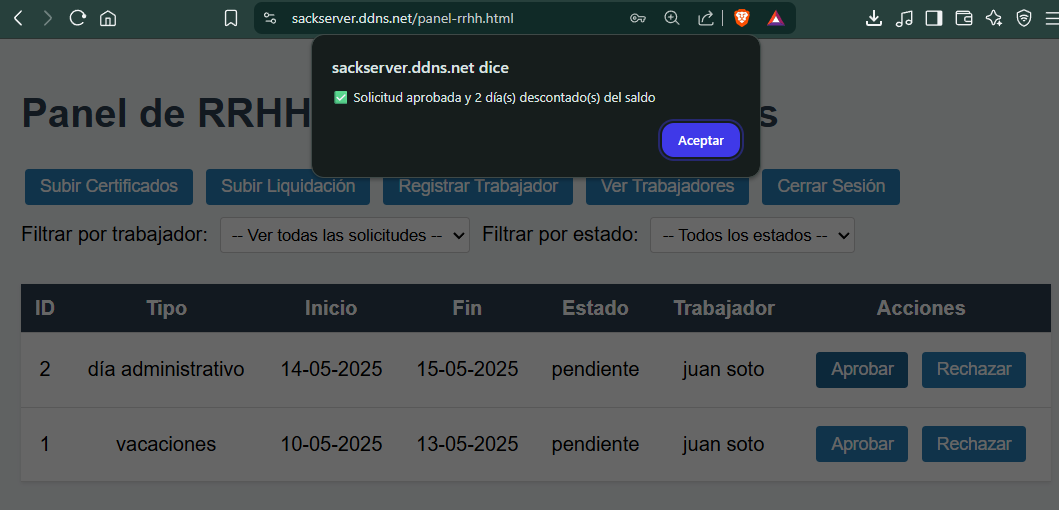
Formulario incompleto mostrando mensaje de validación.

5. Panel RRHH con solicitud pendiente



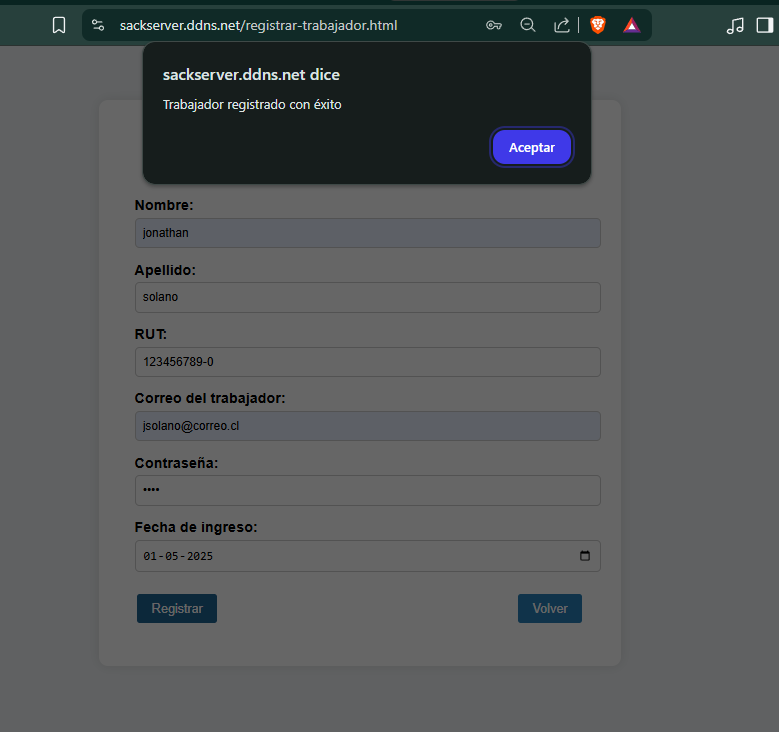
Vista de solicitud recibida antes de aprobarla.

6. Solicitud aprobada y cambio de estado



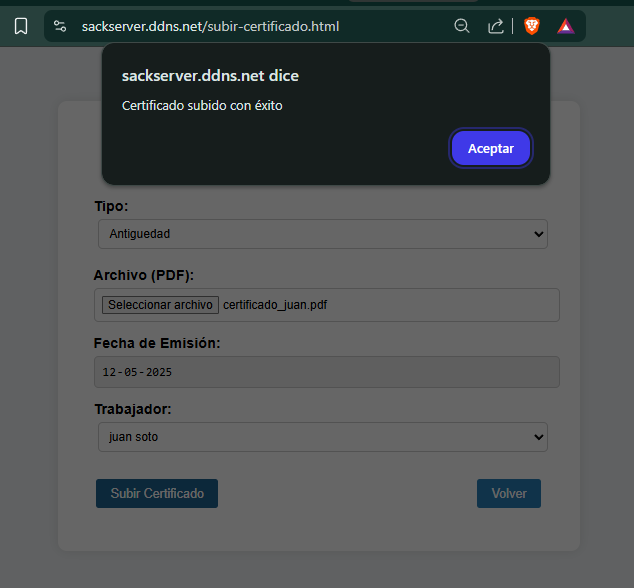
Solicitud con estado actualizado a “aprobada”.

7. Registro exitoso de trabajador

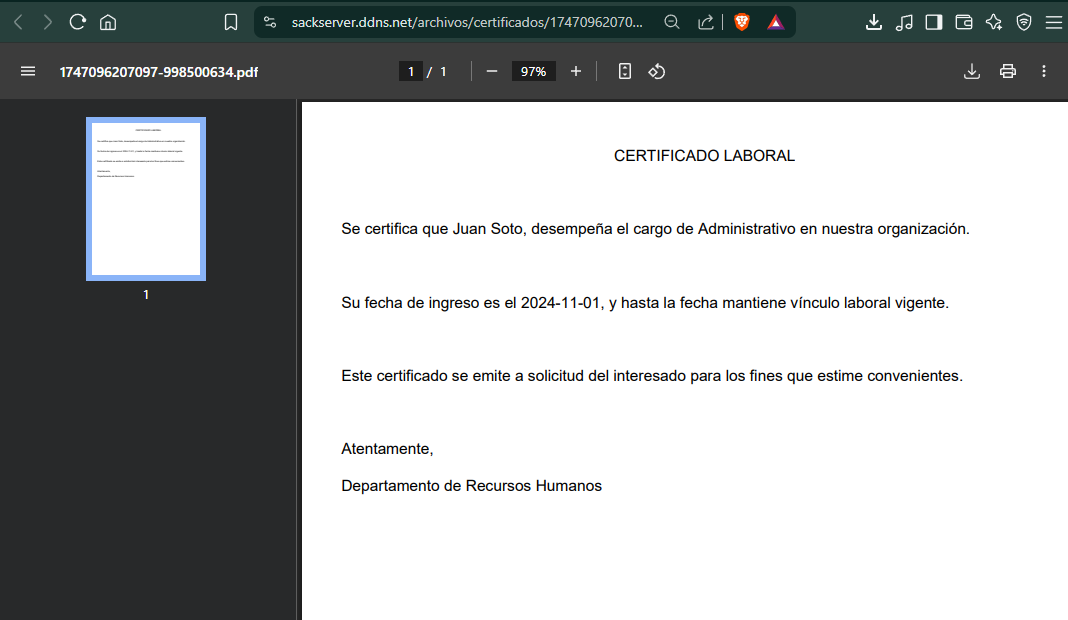
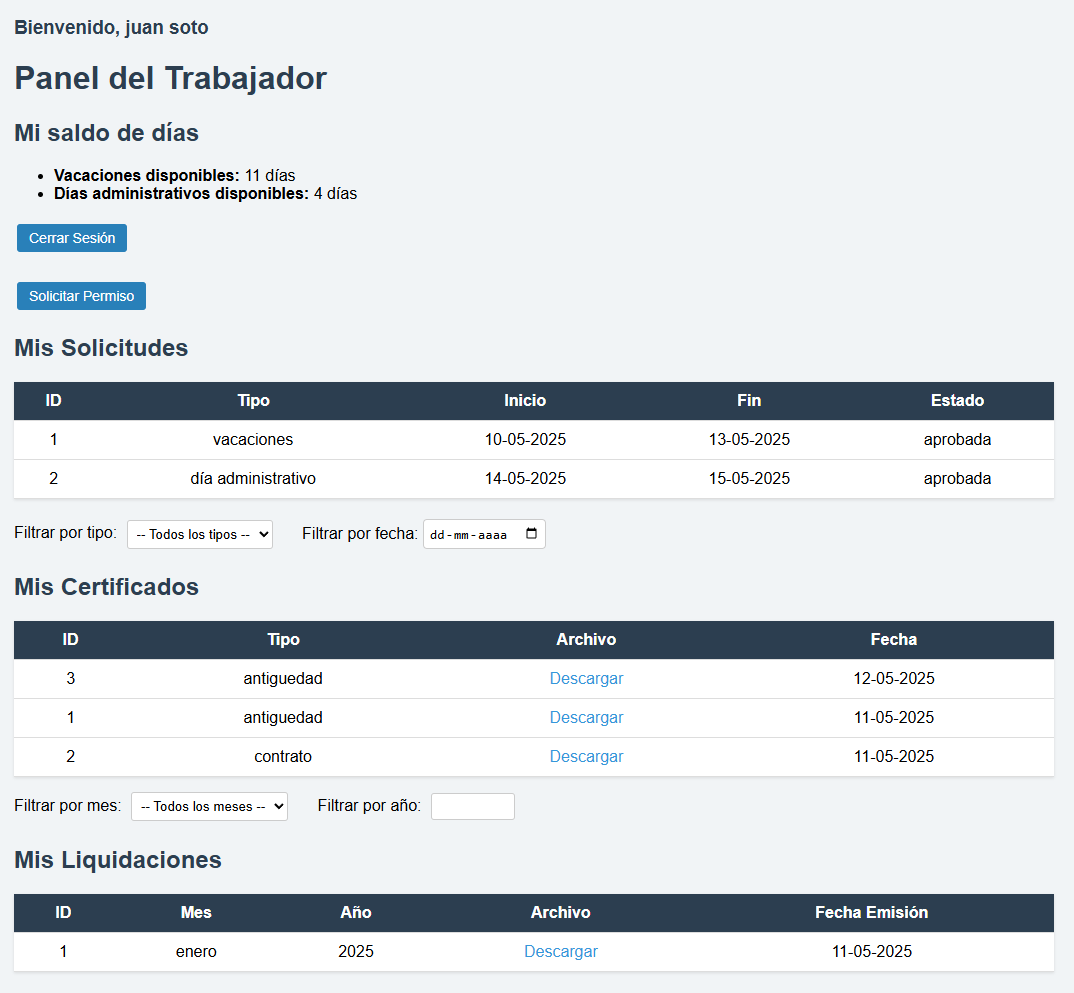


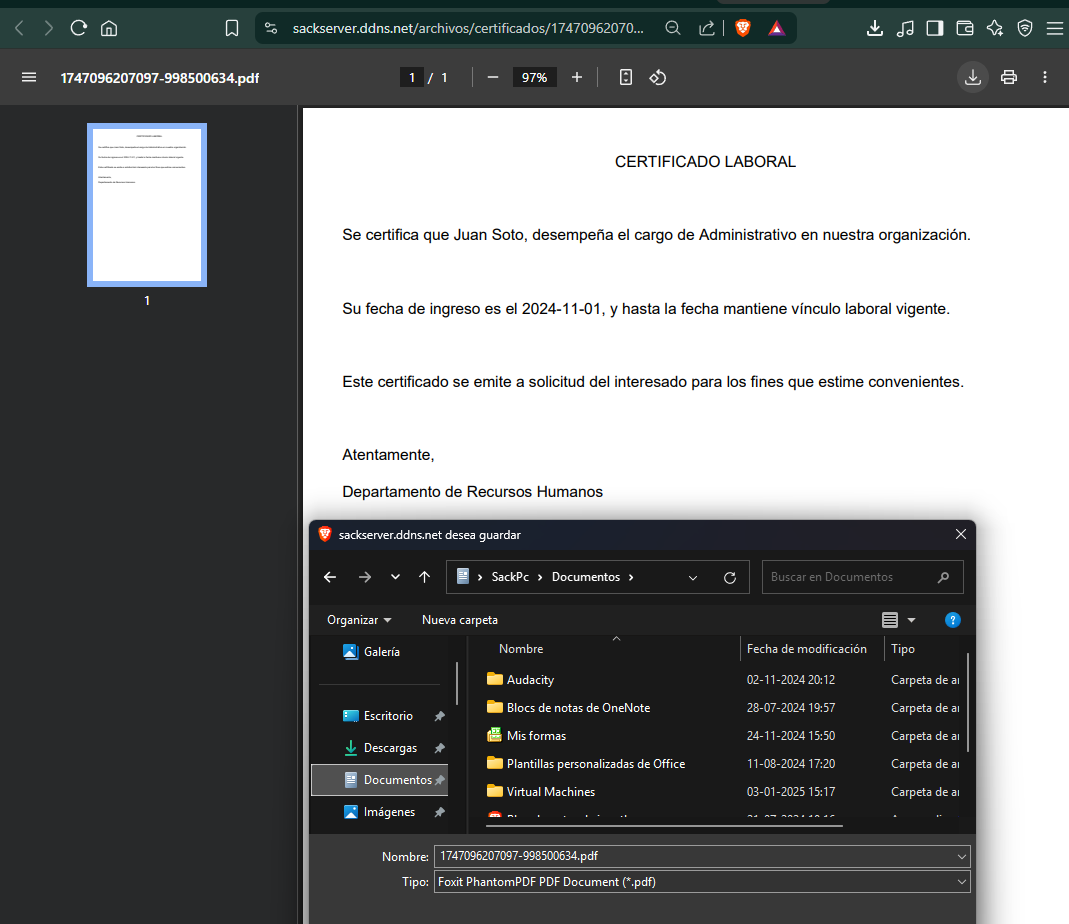
Formulario de registro completado y mensaje de éxito.

8. Subida de certificado por RRHH



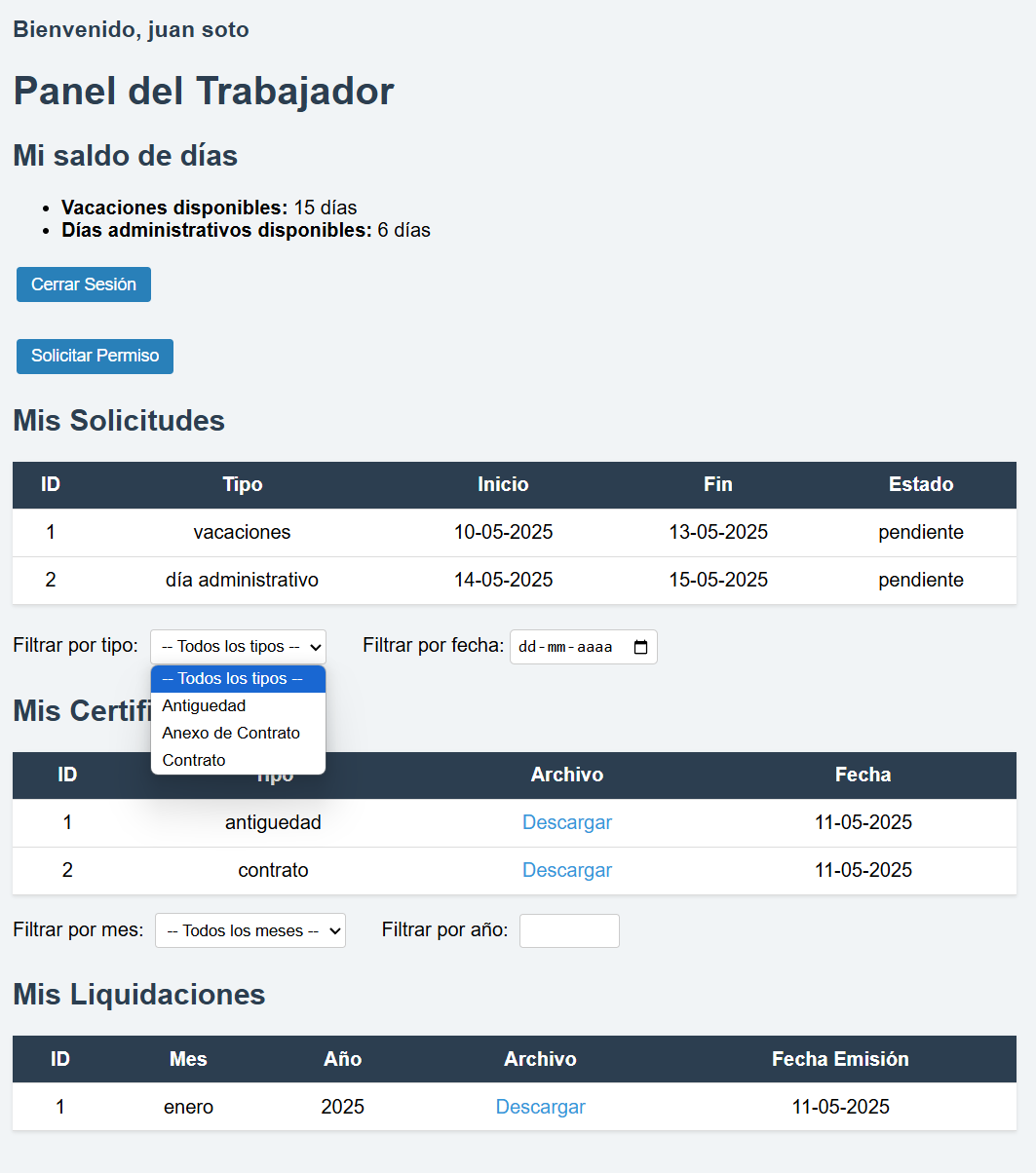
Formulario de carga y mensaje tras subir el archivo.

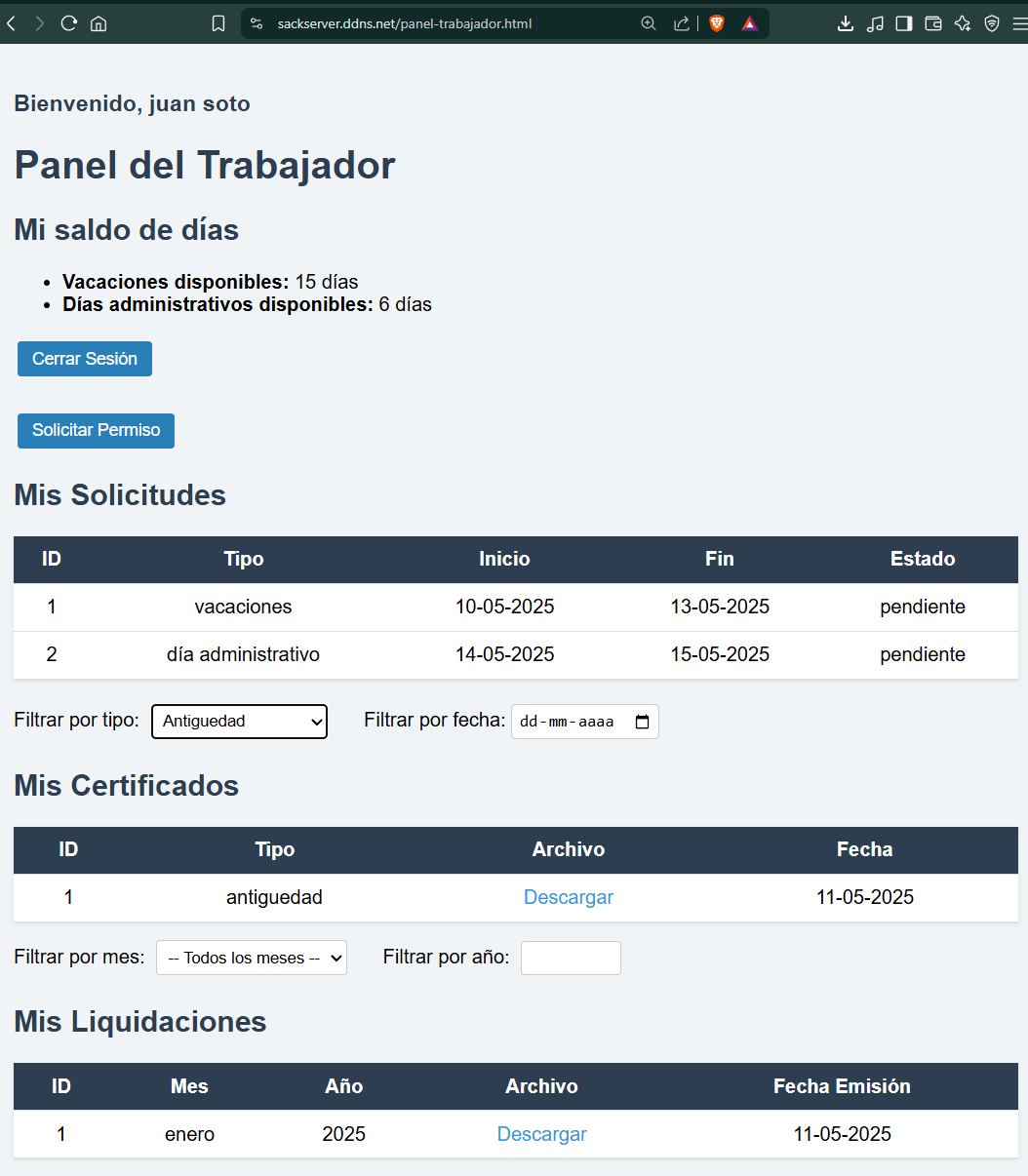
9. Visualización y descarga de certificado



Certificado listado en el panel del trabajador.

10. Filtros aplicados en certificados o liquidaciones





Resultados filtrados por tipo, fecha o año.

**Resumen del Capítulo VIII**

Este capítulo presentó el plan de pruebas diseñado para validar el correcto funcionamiento del sistema web de gestión de recursos humanos. Se definieron los objetivos, el alcance, el propósito del plan y los criterios necesarios para determinar si una funcionalidad es aprobada o rechazada durante su evaluación.

Además, se identificaron claramente las características que serán sometidas a pruebas manuales, así como aquellas que, por razones de alcance y tiempo, no serán consideradas en esta etapa. A través de diez casos de prueba se comprobó que las principales funcionalidades del sistema, como el inicio de sesión, gestión de solicitudes, carga de documentos y administración de usuarios, funcionan según lo esperado.

Finalmente, se incorporaron evidencias gráficas que respaldan la ejecución de cada prueba, lo cual permite demostrar que el sistema responde de manera estable y coherente en un entorno real de uso.

**9 Capítulo IX: Análisis de Resultados y Verificación de Objetivos**

**9.1 Análisis de Resultados**

Tras la finalización del desarrollo e implementación del sistema web de gestión de recursos humanos, se procedió a ejecutar el plan de pruebas definido previamente. Estas pruebas permitieron validar el comportamiento del sistema en condiciones reales de uso, asegurando su estabilidad y correcto funcionamiento.

Entre los principales resultados observados, se destaca la reducción significativa en los tiempos de gestión de solicitudes, certificados y liquidaciones, ya que los procesos antes realizados de forma manual, un trabajador debía esperar entre 24 a 72 horas para recibir una respuesta a su solicitud vía correo o papel. Con el sistema, el envío y la revisión puede realizarse en menos de 5 minutos, dependiendo de la disponibilidad de RRHH. Esta reducción refuerza el objetivo de eficiencia operativa planteado al inicio del proyecto.

Ahora pueden ejecutarse directamente desde el sistema, en cuestión de minutos. Además, la centralización de información y el acceso inmediato a documentos laborales mejoraron la trazabilidad y autonomía tanto para el trabajador como para el personal de RRHH.

El sistema respondió correctamente ante las interacciones del usuario, mostrando mensajes adecuados, realizando validaciones en formularios, y actualizando correctamente el estado de las solicitudes y los saldos correspondientes. El manejo de documentos PDF y la visualización de información por filtros también funcionaron según lo esperado.

Desde el punto de vista técnico, el sistema fue desplegado exitosamente en un entorno real utilizando una máquina virtual con Ubuntu Server sobre VMware Workstation Pro. Se configuró acceso externo mediante DDNS (sackserver.ddns.net) y conexión segura a través de certificados SSL proporcionados por Let's Encrypt, gestionados con Nginx Proxy Manager.

El backend desarrollado con Node.js se ejecuta directamente en el servidor mediante el comando node app.js, y la base de datos MySQL está instalada y funcionando como un servicio local del sistema operativo, permitiendo una integración directa y estable. La herramienta Docker se utiliza únicamente para el despliegue de Nginx Proxy Manager, lo que proporciona modularidad y seguridad en la gestión del acceso externo.

**9.2 Verificación de Objetivos**

Los objetivos definidos en la etapa inicial del proyecto fueron evaluados uno a uno, en base a los resultados obtenidos durante las pruebas manuales. A continuación, se verifica el cumplimiento de cada uno de ellos:

**Objetivo General:**

Diseñar y desarrollar un sistema web funcional para el área de recursos humanos antes del 15 de mayo de 2025, que permita digitalizar solicitudes de permisos, certificados laborales y liquidaciones de sueldo, mejorando la trazabilidad, accesibilidad y eficiencia en la gestión interna.

**Verificación:** Cumplido. El sistema fue desarrollado y desplegado correctamente dentro del plazo establecido, con todas las funcionalidades operativas. La trazabilidad de documentos, la accesibilidad remota y la eficiencia de los procesos se mejoraron significativamente.

**Objetivos Específicos:**

* Login con control de acceso por rol (trabajador/RRHH):
* **Cumplido**. El inicio de sesión redirige correctamente al panel correspondiente según el rol almacenado.
* Envío de solicitudes por parte de trabajadores:
* **Cumplido.** El formulario de solicitud permite enviar permisos que son almacenados, consultados y filtrados correctamente.
* Carga y consulta de certificados PDF:
* **Cumplido.** RRHH puede subir certificados en PDF y los trabajadores los visualizan y descargan desde su panel.
* Visualización de liquidaciones con filtros:
* **Cumplido.** Se cargan correctamente los archivos PDF de liquidaciones y el trabajador puede filtrar por mes y año.
* Gestión de trabajadores (registro, edición y eliminación):
* **Cumplido.** Desde el panel de RRHH se puede administrar completamente el listado de trabajadores, incluyendo la creación del usuario y saldo inicial.

Como cierre de proceso, todos los objetivos planteados en la planificación del proyecto fueron alcanzados satisfactoriamente, y el sistema funciona conforme a los requerimientos definidos. Las pruebas realizadas validaron el correcto desempeño de cada módulo, tanto a nivel funcional como técnico, permitiendo su uso en un entorno real y accesible externamente.

**10 Capítulo X: Conclusión y Reflexión**

**Conclusión**

El desarrollo del sistema web de gestión de recursos humanos permitió dar solución efectiva a una problemática común en muchas organizaciones: la falta de digitalización y trazabilidad en los procesos administrativos relacionados con los trabajadores. A través de la implementación de funcionalidades clave como la autenticación por roles, la gestión de solicitudes de permisos, el manejo de documentación laboral y el control de saldos de días disponibles, se logró construir una plataforma funcional, clara y accesible tanto para el personal de RRHH como para los trabajadores.

El sistema fue diseñado bajo criterios técnicos sólidos, utilizando tecnologías open source y buenas prácticas de desarrollo web. Su implementación en un entorno real mediante el despliegue en una máquina virtual con Ubuntu Server, acceso remoto vía DDNS y protección SSL demuestra su viabilidad para el uso práctico y su potencial para ser ampliado o mantenido en el tiempo.

El proyecto cumplió con todos los objetivos establecidos desde su planificación inicial, y las pruebas realizadas confirmaron la estabilidad, seguridad y eficiencia de cada uno de los módulos implementados.

**Reflexión**

El desarrollo de este proyecto significó una instancia valiosa para aplicar de forma práctica los conocimientos adquiridos durante la carrera, enfrentando un desafío real que integró diseño de bases de datos, programación backend y frontend, validación de formularios, control de usuarios, y despliegue en entornos reales de producción.

Uno de los aprendizajes más importantes fue la planificación del flujo de trabajo, la organización modular del sistema y la implementación de pruebas manuales para asegurar el correcto funcionamiento. También fue relevante enfrentar situaciones reales como la configuración de entornos en máquinas virtuales, la administración de dominios DDNS, y la implementación de certificados SSL para asegurar el acceso al sistema desde cualquier ubicación.

A pesar de las limitaciones de tiempo y recursos, se logró implementar un sistema estable y funcional, que cumple con todos los requerimientos definidos en la etapa inicial del proyecto. Esta experiencia fortalece no solo las competencias técnicas, sino también la capacidad de resolver problemas, adaptarse y tomar decisiones en contextos reales de desarrollo de software.

**Referencias:**

**Fuentes utilizadas para respaldar el diseño y desarrollo del proyecto**

Apache Friends. (s. f.). \*XAMPP: Servidor local de pruebas\*. Recuperado el 27 de abril de 2025, de https://www.apachefriends.org/es/index.html

Express.js. (s. f.). \*Framework web para Node.js\*. Recuperado el 27 de abril de 2025, de https://expressjs.com/es/

Mozilla Developer Network (MDN). (s. f.). \*CSS: Hojas de estilo en cascada\*. Recuperado el 27 de abril de 2025, de https://developer.mozilla.org/es/docs/Web/CSS

Mozilla Developer Network (MDN). (s. f.). \*HTML: Lenguaje de marcas de hipertexto\*. Recuperado el 27 de abril de 2025, de https://developer.mozilla.org/es/docs/Web/HTML

Mozilla Developer Network (MDN). (s. f.). \*JavaScript: Guía y documentación\*. Recuperado el 27 de abril de 2025, de https://developer.mozilla.org/es/docs/Web/JavaScript

Node.js. (s. f.). \*Documentación oficial de Node.js\*. Recuperado el 27 de abril de 2025, de https://nodejs.org/es/docs

Oracle Corporation. (s. f.). \*Manual de referencia del desarrollador MySQL\*. Recuperado el 27 de abril de 2025, de https://dev.mysql.com/doc/

Scrum.org. (s. f.). \*The Scrum Guide\*. Recuperado el 27 de abril de 2025, de <https://www.scrum.org/resources/scrum-guide>

Nginx Proxy Manager. (2024). Secure and Manage Nginx with a GUI. Recuperado de: <https://nginxproxymanager.com/>

DDNS - No-IP. (2024). Dynamic DNS Services. Recuperado el 10 de mayo de 2025 de: <https://www.noip.com/>

Ubuntu. (2024). Netplan - YAML network configuration. Recuperado el 10 de mayo de 2025 de: <https://netplan.io/>

VMware Workstation Pro. (2024). Documentation and Downloads. Recuperado el 10 de mayo de 2025 de: <https://www.vmware.com/products/workstation-pro.html>

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente