Informe Desarrollo Web II

Sitio de Gestiones Internas

Versiones del documento

Fecha	Versión	Descripción	Autor
15/06/2025	1.0	Avance de documentación del proyecto.	1. Carlos Ordoñez 2. Cesar Reyes 3. Cristopher Enamorado 4. Joaquín Aguilar

Índice

Contenido

Informe Desarrollo de Aplicaciones Web II 4
1. Descripción de la solución 4
1.1 Resumen gerencial 4
1.2 Glosario 4
1.3 Restricciones y metas de arquitectura 5
1.4 Diagrama de Proceso 6
1.5 Requerimientos del sistema 7
1.5.1 Vista de Casos de uso de la aplicación 7
2. Tecnología 9
2.1 Desarrollo 9
2.2 Modelo Entidad Relación 11

Informe Desarrollo de Aplicaciones Web II

1. Descripción de la solución

Se desarrolló esta solución por su viabilidad para gestionar las solicitudes internas de una empresa de manera más eficiente enfocándose en la mejora de la atención, la organización y la versatilidad de los procesos.

1.1 Resumen gerencial

Este proyecto consiste en el desarrollo de una plataforma web interna para centralizar la gestión de tickets o solicitudes de tareas dentro de la empresa. A través del sistema, cualquier usuario autenticado podrá crear tickets para solicitar apoyo o intervención por parte de áreas específicas como Tecnología (IT), Administración o Recursos Humanos.

Los tickets se registrarán con información esencial como la descripción de la solicitud, el área correspondiente y el estado actual del ticket (pendiente, en proceso, finalizado o cancelado). No habrá personal dedicado exclusivamente a la asignación de tickets; únicamente los usuarios con rol de administrador tendrán la facultad de gestionar y reasignar estos tickets según sea necesario.

El sistema incluirá herramientas de seguimiento del estado de los tickets y una sección de reportes accesible solo por administradores. Los reportes permitirán filtrar por rangos de fechas y por áreas específicas, mostrando el conteo total de tickets según su estado, con el fin de ofrecer una visión general del volumen de solicitudes y su distribución.

1.2 Glosario

Término	Definición
Usuario	Cuenta o entidad registrada dentro de un sistema, generalmente con credenciales (nombre de usuario y contraseña), asociada a ciertos permisos o roles.
Usuario administrador	Usuario con privilegios elevados que le permiten gestionar, configurar y controlar diferentes aspectos del sistema, aplicación o plataforma.
Tickets	Un registro individual que documenta una solicitud, incidencia, tarea o problema reportado por un usuario, el cual debe ser gestionado o resuelto.

Reportes	Vista generada por el sistema que muestra información, generalmente con fines de análisis, seguimiento o auditoría.

1.3 Restricciones y metas de arquitectura

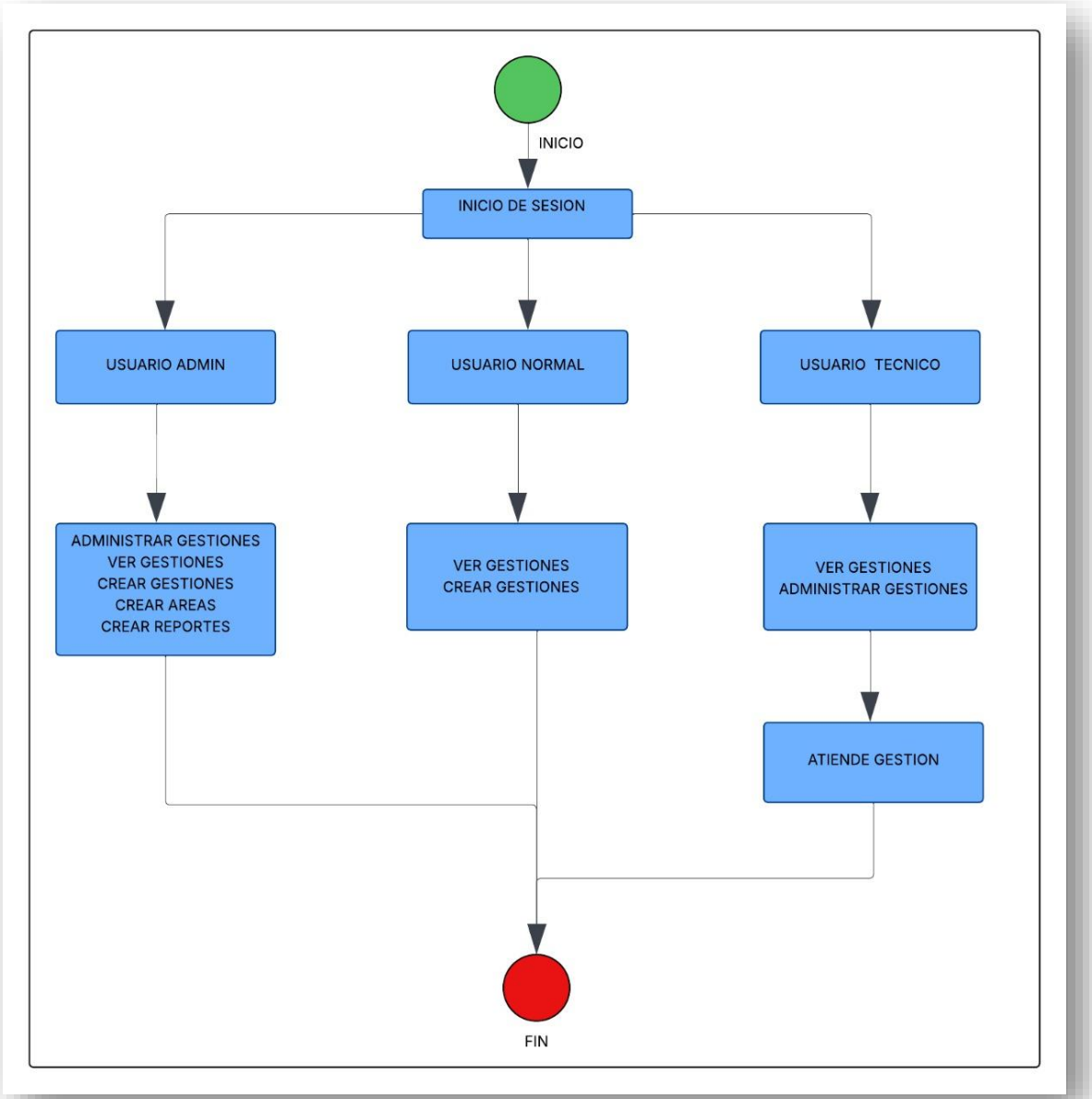
Restricciones:

- Infraestructura limitada: El sistema deberá ejecutarse sobre la infraestructura tecnológica actual de la empresa, sin posibilidad inmediata de migración a servicios en la nube o de escalar horizontalmente.
- Acceso restringido a la red interna: La aplicación está pensada para ser utilizada únicamente dentro de la red corporativa o mediante VPN, limitando su acceso desde ubicaciones externas.
- Usuarios no técnicos: Los usuarios finales no tienen formación técnica, por lo que la interfaz debe ser altamente intuitiva y el sistema debe requerir mínima capacitación.

Metas de arquitectura:

- Aplicar buenas prácticas de desarrollo seguro (uso de bcrypt, validación de entradas, control de sesiones).
- Implementar una arquitectura cliente-servidor desacoplada basada en API REST usando NestJS y Next.js.
- Garantizar una estructura modular que facilite el mantenimiento y escalabilidad del sistema.

1.4 Diagrama de Proceso



1.5 Requerimientos del sistema

- Inicio de sesión.
  - El sistema debe permitir a los usuarios autenticarse con su usuario y contraseña.
  - El sistema debe validar credenciales mediante JWT y Passport.
  - El acceso debe restringirse según el rol asignado (administrador, usuario estándar).
- Gestión de tickets.
  - Los usuarios podrán crear solicitudes o tickets desde un formulario en la plataforma.
  - Cada ticket debe contener: título, descripción y área destino.
  - Los encargados podrán actualizar el estado del ticket (pendiente, en proceso, resuelto, cerrado).
- Generación de reportes.
  - El sistema debe permitir la generación de reportes por área, estado y tiempos de resolución.

1.5.1 Vista de Casos de uso de la aplicación

Nombre del caso de uso	Descripción de la funcionalidad	Ruta del Caso de Uso
Inicio de sesión	Se debe permitir el inicio de sesión con contraseña y usuarios	Se muestra la pantalla de inicio de sesión en la que se encuentra un formulario donde se deben utilizar las credenciales asignadas por la empresa (correo y contraseña).
Creación de ticket	Permite al usuario crear un ticket indicando título, descripción, área, prioridad y fecha límite.	Una vez iniciada la sesión el usuario llega a la pantalla de tickets, en la barra de navegación se encuentra el botón “crear ticket”, botón que muestra un formulario en el que se debe ingresar la información correspondiente (título, descripción, área).

Visualización de tickets	Muestra al usuario los tickets relacionados con su cuenta. Los administradores pueden ver todos.	Luego de iniciar sesión se mostrará en la barra de navegación el botón de “mis tickets”. Dando clic en este botón el usuario podrá acceder a una pantalla donde podrá ver los tickets realizados por el mismo. En caso de ser usuario administrador se podrán ver todos los tickets que se han generado.
Actualizar tickets	El encargado o administrador puede cambiar el estado de un ticket (pendiente, en proceso, resuelto, cerrado).	En la pantalla de tickets, en caso de ser usuario administrador o técnico podrá visualizar los botones de actualizar y eliminar ticket. Al hacer clic en el botón de actualizar el usuario podrá cambiar el estado del ticket.
Generar reporte general por fechas de todas las áreas o una	Permite obtener un resumen del conteo de tickets por estado (pendiente, en proceso, finalizado, cancelado) en un rango de fechas específico.	1. Ingresar al módulo "Reportes" desde el menú de navegacion 2. Seleccionar fecha inicial y final y todas las áreas (por defecto) o un área. 3. Presionar "Consultar"

2. Tecnología

- NestJS
- TypeORM
- MySQL
- JWT (JSON Web Tokens)
- Passport (estrategias `local` y `jwt`)
- Bcrypt (hash de contraseñas)

2.1 Desarrollo

NestJs: Framework backend para Node.js basado en TypeScript que facilita la creación de APIs RESTful escalables y mantenibles mediante una arquitectura modular y orientada a controladores.

NextJs: Framework basado en React para la creación de aplicaciones web modernas, que permite renderizado híbrido (SSR - Server Side Rendering, y SSG - Static Site Generation) para mejorar el rendimiento y SEO.

API Rest: conjunto de principios arquitectónicos para el diseño de sistemas de software distribuidos. Estos principios se centran en la creación de servicios web que sean escalables, interoperables, y fáciles de mantener.

TypeScript: Superset de JavaScript con tipado estático utilizado en ambos entornos para mejorar la robustez y calidad del código.

JSON Web Token: estándar abierto (RFC 7519) que define un formato compacto y autocontenido para transmitir de manera segura la información entre dos partes como un objeto JSON. Está diseñado para ser compacto, seguro y fácilmente utilizable tanto para la transmisión de información entre partes como para la verificación de su autenticidad.

TypeORM: ORM (Object-Relational Mapping) para TypeScript y JavaScript, utilizado para interactuar con la base de datos de forma eficiente y segura, facilitando la gestión de entidades y migraciones.estándar abierto (RFC 7519) que define un formato compacto y autocontenido para transmitir de manera segura la información entre dos partes como un objeto JSON. Está diseñado para ser compacto, seguro y fácilmente utilizable tanto para la transmisión de información entre partes como para la verificación de su autenticidad.

MySQL: Sistema de gestión de bases de datos relacional, utilizado para el almacenamiento y consulta eficiente de la información del sistema.

Passport: Middleware de autenticación para Node.js, empleado para implementar estrategias de autenticación robustas y flexibles, incluyendo login con JWT.

Bcrypt: Librería para el hashing seguro de contraseñas, utilizada para proteger la información sensible de los usuarios mediante técnicas de encriptación.

Control de versiones: Git para la gestión y control del código fuente durante el desarrollo colaborativo.

2.2 Modelo Entidad Relación