Documentación General del Proyecto

Sistema de Gestión de Turnos para  
la Biblioteca de la UTN

Tabla de contenido

[Requerimientos funcionales 3](#_Toc202029824)

[Requerimientos no funcionales 4](#_Toc202029825)

[Requerimientos derivados del negocio 5](#_Toc202029826)

[Diagrama de casos de uso 6](#_Toc202029827)

[Descripciones de Casos de Uso 7](#_Toc202029828)

[CU-01: Registrar Turno 7](#_Toc202029829)

[CU-02: Registrar Invitados 8](#_Toc202029830)

[CU-03: Ver Invitaciones Recibidas 8](#_Toc202029831)

[CU-04: Aceptar o Rechazar Invitaciones 9](#_Toc202029832)

[CU-05: Registrar Asistencia Real 10](#_Toc202029833)

[CU-06: Crear Sala 11](#_Toc202029834)

[Diagrama de clases 12](#_Toc202029835)

[Proceso: Registrar turno 13](#_Toc202029836)

[Proceso: Registrar respuesta a invitación 14](#_Toc202029837)

[Diagramas de secuencia 15](#_Toc202029838)

[Proceso: Registrar turno 15](#_Toc202029839)

[Proceso: Responder a invitación 15](#_Toc202029840)

[CU: Registrar usuario 16](#_Toc202029841)

[CU: Iniciar sesión 16](#_Toc202029842)

[CU: Cerrar sesión 17](#_Toc202029843)

[CU: Crear sala 17](#_Toc202029844)

[CU: Editar sala 18](#_Toc202029845)

[CU: Eliminar sala 18](#_Toc202029846)

[CU: Ver turnos propios 19](#_Toc202029847)

[CU: Ver invitaciones recibidas 19](#_Toc202029848)

# Requerimientos funcionales

Estudiantes:

1. El usuario puede registrarse con DNI, nombre, Email y contraseña.
2. El usuario puede iniciar sesión con su DNI y contraseña.
3. El sistema permite recuperar y cambiar la contraseña del usuario.
4. El sistema permite verificar y actualizar los datos personales.
5. El usuario puede acceder a la sección de turnos.
6. El sistema permite crear tunos.
7. El usuario puede seleccionar el área a reservar y la temática.
8. El usuario puede elegir una fecha y un rango horario disponible.
9. El usuario puede agregar DNI de invitados al turno.
10. El sistema valida la disponibilidad de la sala según capacidad total y ocupación en el horario seleccionado.
11. El sistema notifica por correo y plataforma a los invitados del turno.
12. Los invitados pueden aceptar o rechazar la invitación al turno.
13. El creador del turno puede finalizarlo manualmente para liberar espacio en el horario.
14. El sistema actualiza automáticamente la disponibilidad de la sala al finalizar un turno.
15. El sistema registra el tiempo real de uso del turno (opcional).
16. El sistema debe permitir consultar las reservas realizadas por el estudiante.
17. El usuario puede dejar una solicitud en modo "espera" si no hay lugar disponible.
18. El sistema monitorea la disponibilidad y notifica al usuario cuando hay espacio.
19. El usuario tiene un tiempo límite para aceptar la notificación y confirmar su turno.

Encargados:

1. El sistema debe permitir al encargado iniciar sesión.
2. El sistema debe mostrar todas las reservas existentes.
3. El sistema debe permitir filtrar reservas por Usuario, temática, DNI, área o estado.
4. El sistema debe permitir modificar el estado de una reserva.

Comunes:

1. El sistema debe enviar notificaciones por correo o internamente al realizar o modificar una reserva.
2. El sistema debe gestionar sesiones según el rol del usuario.

# Requerimientos no funcionales

1. El sistema debe tener una interfaz simple, visualmente agradable y responsiva.
2. La interfaz debe ser intuitiva y accesible en dispositivos móviles.
3. El sistema debe validar la identidad del usuario al iniciar sesión (seguridad).
4. El sistema debe estar protegido contra accesos no autorizados y manejo incorrecto de datos.
5. El sistema debe ser compatible con los navegadores modernos (Chrome, Firefox, Edge).
6. El sistema debe guardar un historial de reservas por usuario.
7. La información del usuario debe almacenarse cifrada (especialmente la contraseña).

# Requerimientos derivados del negocio

1. El sistema debe estar alojado en los servidores de la universidad o plataforma autorizada.
2. Debe poder escalarse en el futuro para incluir más tipos de recursos (proyectores, computadoras, etc.).

# Diagrama de casos de uso

# Descripciones de Casos de Uso

## CU-01: Registrar Turno

* **Actor Principal:** Usuario
* **Precondiciones:** El usuario debe estar autenticado en el sistema
* **Postcondiciones:** Se crea un nuevo turno en el sistema
* **Flujo Principal:**
  1. El usuario accede a la funcionalidad "Crear Turno"
  2. El sistema muestra el formulario de creación de turno
  3. El usuario selecciona la sala disponible
  4. El usuario ingresa fecha y hora (inicio y fin)
  5. El usuario especifica la temática del turno
  6. El usuario indica la cantidad de integrantes
  7. El usuario puede agregar observaciones (opcional)
  8. El sistema valida que la sala esté disponible en el horario seleccionado
  9. El sistema valida que la fecha no sea anterior a la actual
  10. El sistema crea el turno con estado "pendiente"
  11. El sistema confirma la creación exitosa
* **Flujos Alternativos:**

8a. Si la sala no está disponible:

* + El sistema muestra mensaje de error
  + El usuario debe seleccionar otro horario

9a. Si la fecha es inválida:

* + El sistema muestra mensaje de error
  + El usuario debe corregir la fecha

## CU-02: Registrar Invitados

* **Actor Principal:** Usuario
* **Precondiciones:** El usuario tiene un turno creado
* **Postcondiciones:** Se registran invitados al turno
* **Flujo Principal:**
  1. El usuario selecciona un turno existente
  2. El sistema muestra la opción "Invitar personas"
  3. El usuario ingresa email del invitado
  4. El usuario ingresa nombre del invitado
  5. El sistema valida el formato del email
  6. El sistema crea la invitación con estado "pendiente"
  7. El sistema envía notificación al invitado (si implementado)
  8. El sistema confirma que la invitación fue enviada
* **Flujos Alternativos:**

5a. Si el email es inválido:

* + El sistema muestra mensaje de error
  + El usuario debe corregir el email

## CU-03: Ver Invitaciones Recibidas

* **Actor Principal:** Usuario
* **Precondiciones:** El usuario está autenticado
* **Postcondiciones:** El usuario visualiza sus invitaciones
* **Flujo Principal:**
  1. El usuario accede a "Invitaciones Recibidas"
  2. El sistema consulta las invitaciones del usuario
  3. El sistema muestra lista de invitaciones con:
  + Información del turno (fecha, hora, sala, temática)
  + Estado de la invitación
  + Opciones para aceptar/rechazar
  1. El usuario puede ver detalles de cada invitación
* **Flujos Alternativos:**

2a. Si no tiene invitaciones:

* + El sistema muestra mensaje "No tienes invitaciones pendientes"

## CU-04: Aceptar o Rechazar Invitaciones

* **Actor Principal:** Usuario
* **Precondiciones:** El usuario tiene invitaciones pendientes
* **Postcondiciones:** Se actualiza el estado de la invitación
* **Flujo Principal:**
  1. El usuario selecciona una invitación pendiente
  2. El usuario elige "Aceptar" o "Rechazar"
  3. El sistema actualiza el estado de la invitación
  4. El sistema registra la fecha de respuesta
  5. El sistema confirma la acción realizada
  6. El sistema notifica al creador del turno (si implementado)

## CU-05: Registrar Asistencia Real

* **Actor Principal:** Administrador
* **Precondiciones:** Existe un turno programado
* **Postcondiciones:** Se registra la asistencia real al turno
* **Flujo Principal:**
  1. El administrador accede a "Gestión de Turnos"
  2. El sistema muestra turnos del día actual
  3. El administrador selecciona un turno específico
  4. El sistema muestra lista de participantes (creador + invitados aceptados)
  5. El administrador marca la asistencia de cada participante:
  6. Presente
  7. Ausente
  8. El administrador puede agregar observaciones
  9. El sistema actualiza la asistencia en InvitadoTurno
  10. El sistema confirma el registro
* **Flujos Alternativos:**

2a. Si no hay turnos para el día:

* + El sistema muestra mensaje "No hay turnos programados para hoy"

## CU-06: Crear Sala

* **Actor Principal:** Administrador
* **Precondiciones:** El administrador está autenticado
* **Postcondiciones:** Se crea una nueva sala en el sistema
* **Flujo Principal:**
  1. El administrador accede a "Gestión de Salas"
  2. El administrador selecciona "Crear Nueva Sala"
  3. El sistema muestra formulario de creación
  4. El administrador ingresa nombre de la sala
  5. El administrador especifica la capacidad
  6. El administrador indica si está disponible
  7. El sistema valida que no exista otra sala con el mismo nombre
  8. El sistema crea la sala
  9. El sistema confirma la creación exitosa
* **Flujos Alternativos:**

7a. Si ya existe una sala con ese nombre:

* + El sistema muestra mensaje de error
  + El administrador debe cambiar el nombre

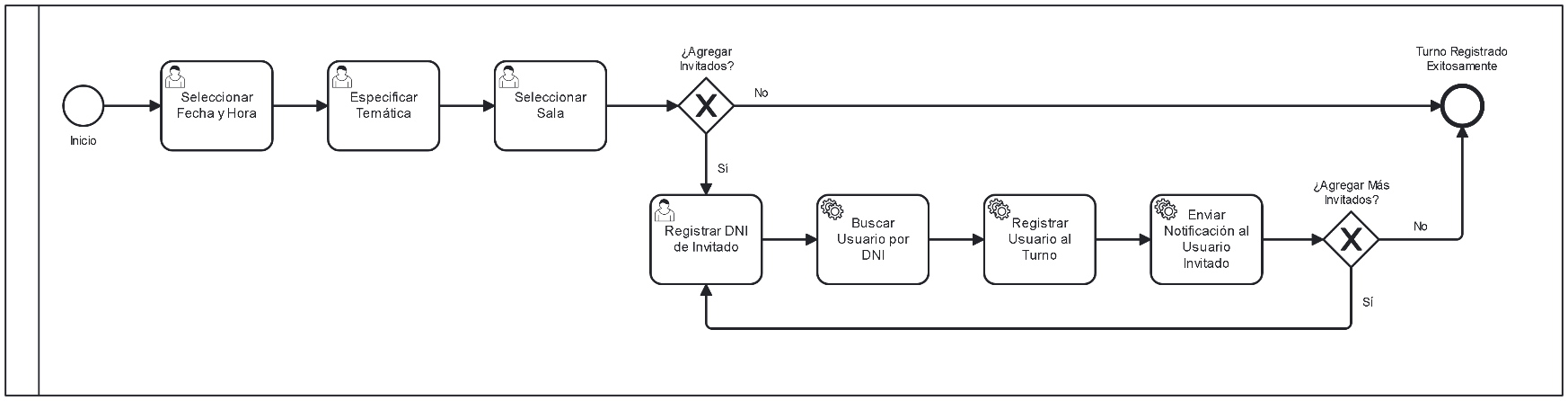
# Diagrama de clases

# Proceso: Registrar turno

Objetivo:

Permitir que el usuario seleccione área, temática, tipo de asistencia, fecha, hora y cantidad de personas.

Requerimientos involucrados:

1. El usuario puede seleccionar el área a reservar y la temática.
2. El usuario puede elegir una fecha y un rango horario disponible.
3. El usuario puede agregar DNI de invitados al turno.

# Proceso: Registrar respuesta a invitación

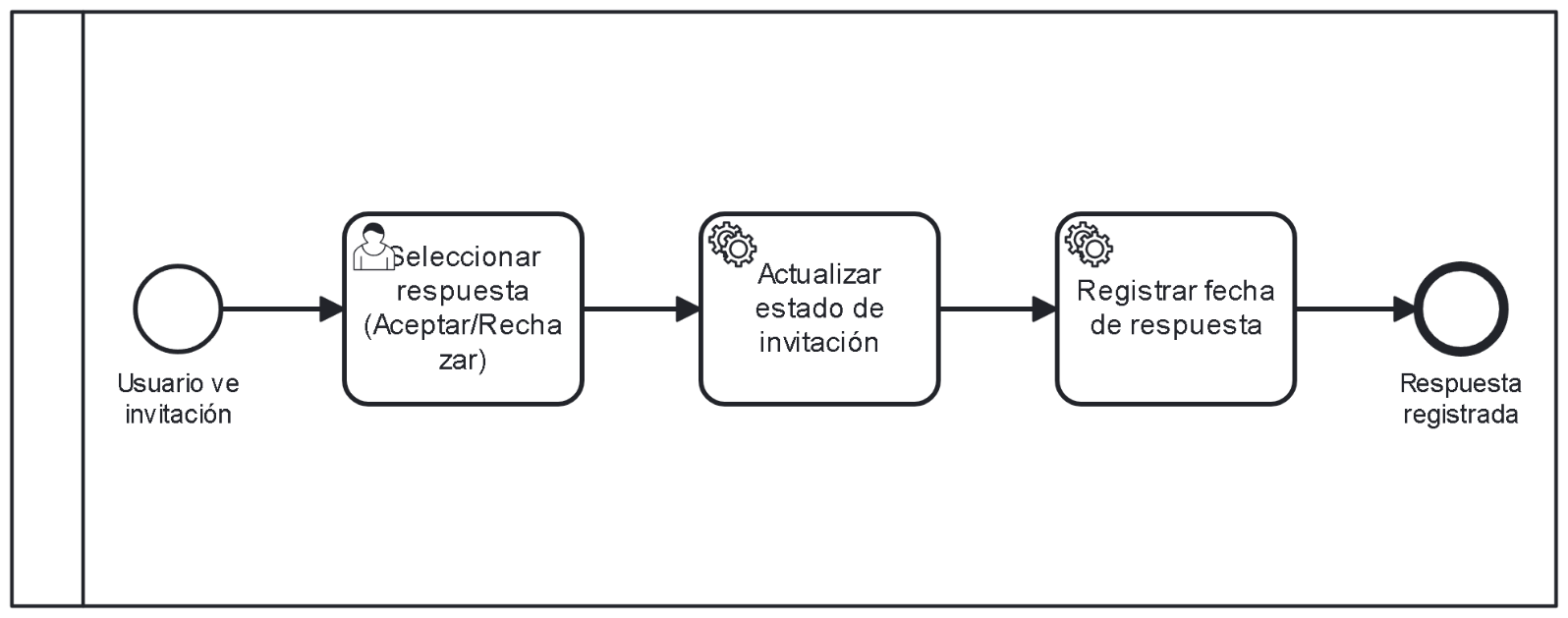
Objetivo:

Notificar a invitados y permitir aceptar/rechazar invitación.

Requerimientos involucrados:

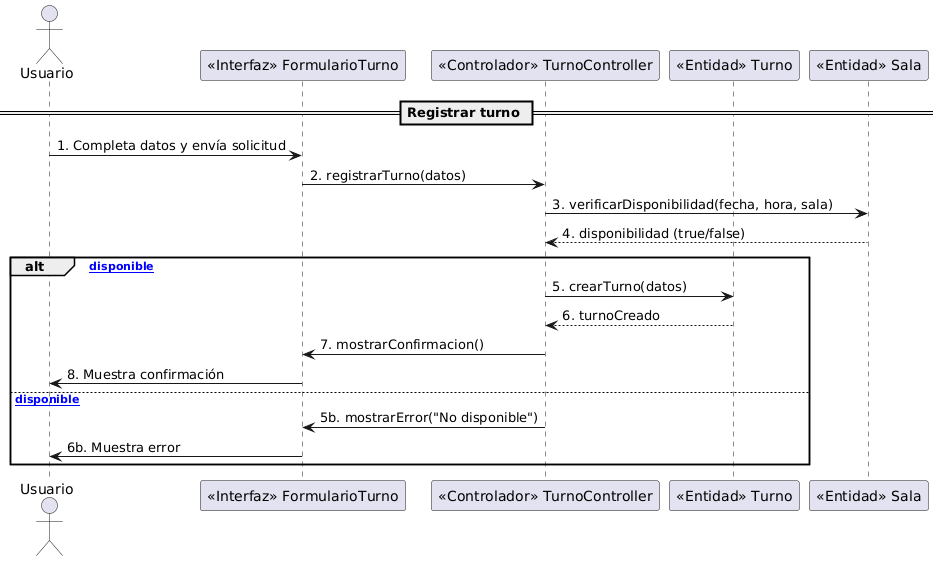
1. El sistema notifica por correo y plataforma a los invitados del turno.
2. Los invitados pueden aceptar o rechazar la invitación al turno.

Interfaces:

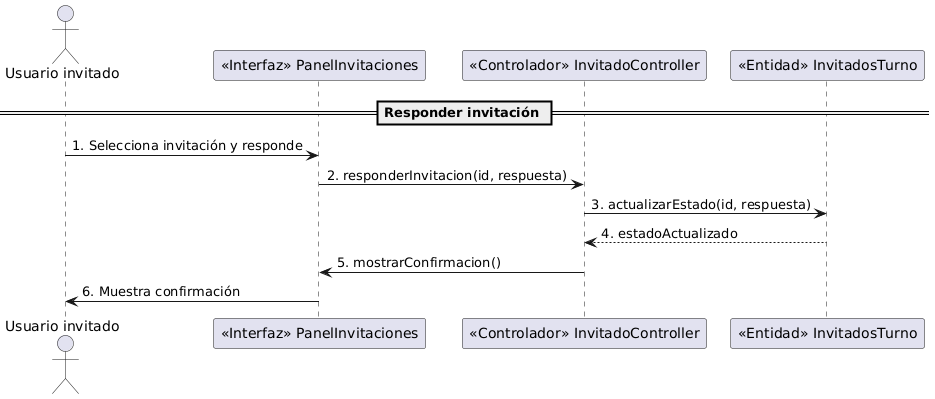
* Panel de invitaciones.

# Diagramas de secuencia

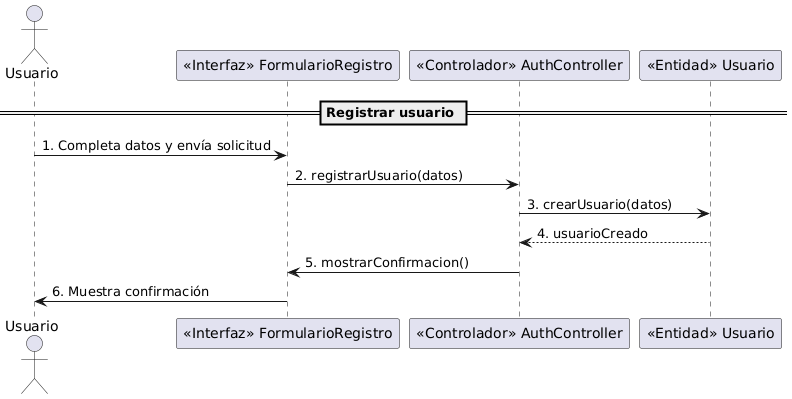
### Proceso: Registrar turno



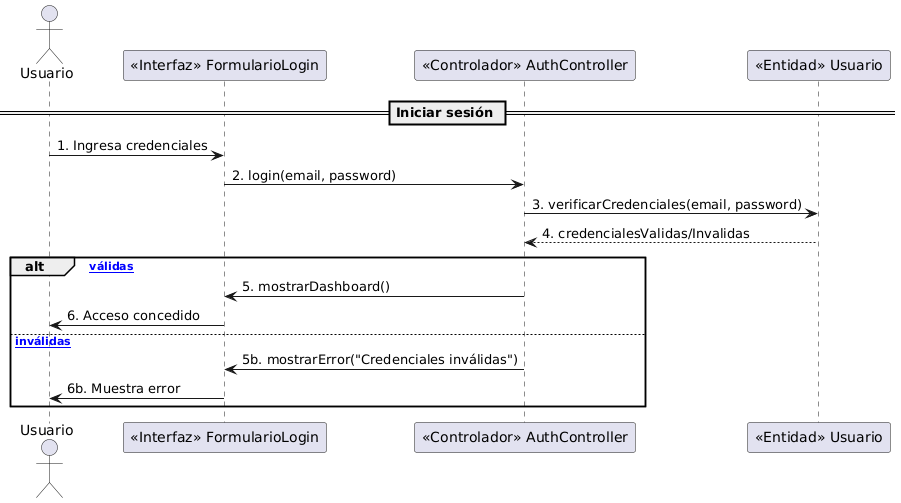
### Proceso: Responder a invitación



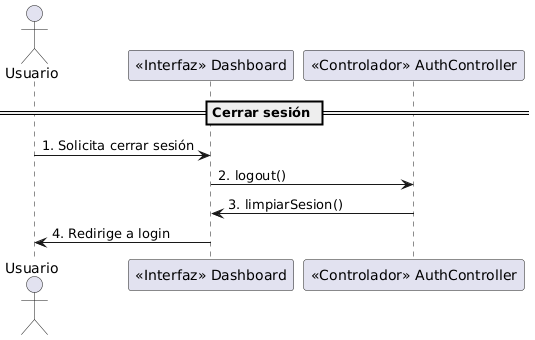
### CU: Registrar usuario



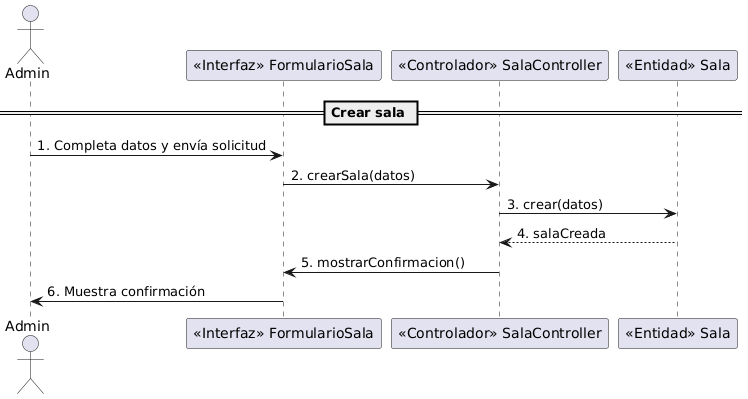
### CU: Iniciar sesión



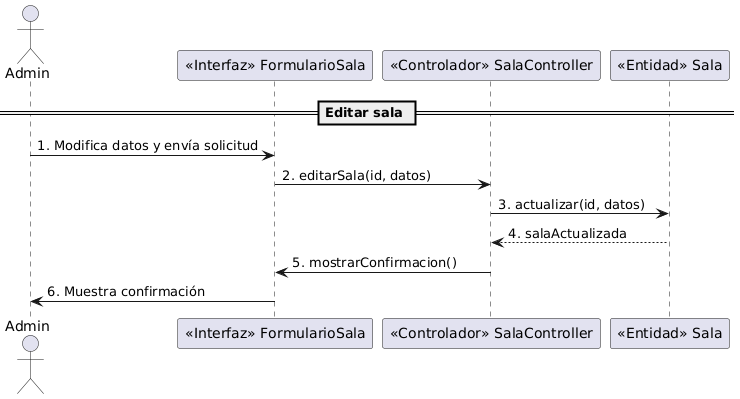
### CU: Cerrar sesión



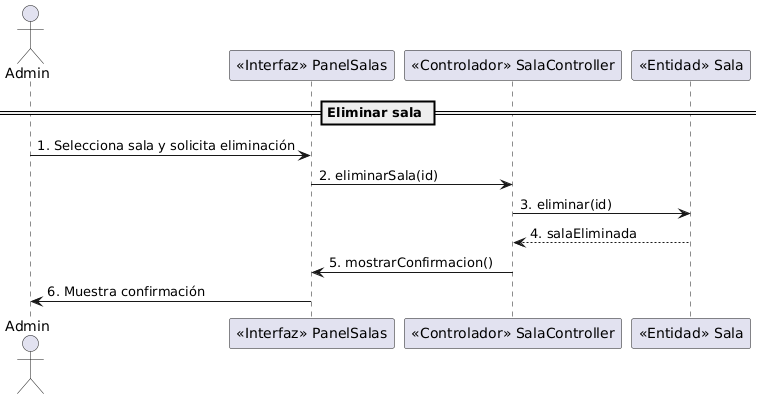
### CU: Crear sala



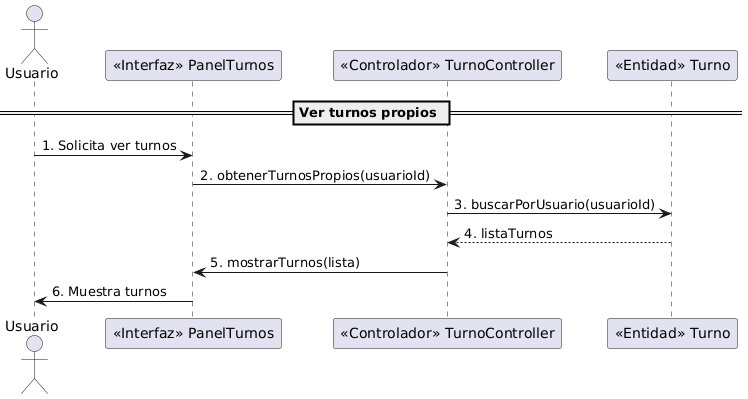
### CU: Editar sala



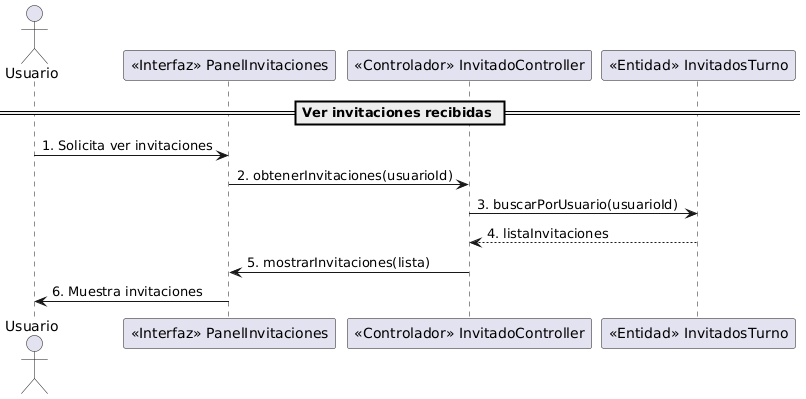
### CU: Eliminar sala



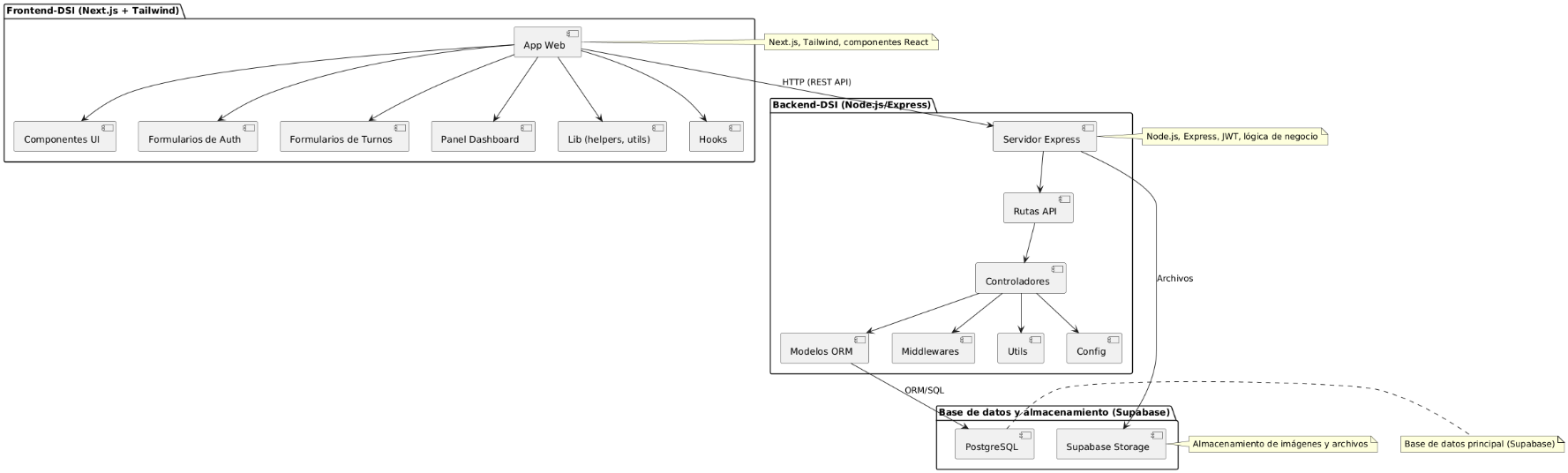
### CU: Ver turnos propios



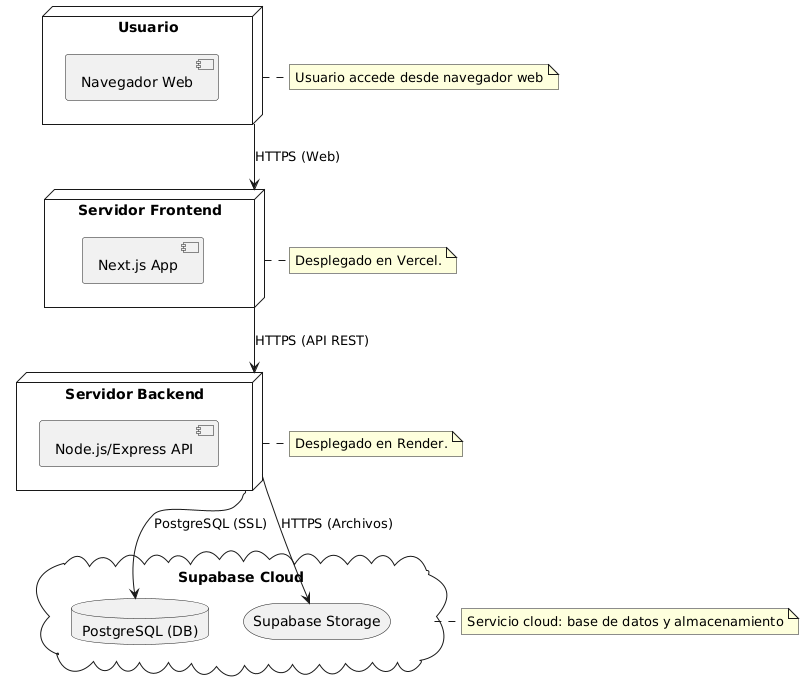
### CU: Ver invitaciones recibidas



# Diagrama de componentes



# Diagrama de despliegue



## Tabla El contenido generado por IA puede ser incorrecto.Test Case 1

## Test Case 2

## Tabla El contenido generado por IA puede ser incorrecto.Test Case 3

## Test Case 4

## Test Case 5

# 1. Gestión insegura de contraseñas

## Problema:

Las contraseñas de los usuarios pueden estar almacenadas en texto plano o con algoritmos de hash inseguros, exponiendo la información ante una brecha de datos.

## Contramedida:

Utilizar algoritmos de hash robustos como bcrypt, Argon2 o scrypt, configurando un nivel de dificultad adecuado. Nunca almacenar la contraseña original.

## Implementación:

* Instalar la librería “bcrypt” (“npm install bcrypt”).
* Modificar el backend para hashear la contraseña antes de guardarla y comparar hashes al autenticar.
* Revisar el modelo de usuario y los endpoints de registro/login.

# 2. Falta de validación y sanitización de entradas

## Problema:

La ausencia de validación permite ataques como SQL Injection y Cross-Site Scripting (XSS).

## Contramedida:

Validar y sanitizar todos los datos recibidos del usuario. Utilizar ORM seguro (por ejemplo, Sequelize) y escapar datos en el frontend.

## Implementación:

* Usar librerías como “zod” o “joi” para validar datos en backend y frontend.
* Revisar todos los formularios y endpoints para agregar validaciones.
* Configurar el ORM para evitar consultas inseguras.

# 3. Gestión deficiente de sesiones y tokens

## Problema:

Sesiones inseguras pueden ser secuestradas por atacantes.

## Contramedida:

Utilizar cookies seguras (“httpOnly”, “secure”, “SameSite”), implementar expiración y rotación de tokens, y evitar almacenar tokens en localStorage.

## Implementación:

* Configurar cookies en el backend con los flags mencionados.
* Usar JWT con expiración y refresco.
* Revisar el flujo de login/logout y almacenamiento de sesión.

# 4. Falta de control de acceso y autorización

## Problema:

Usuarios pueden acceder a recursos o acciones que no les corresponden.

## Contramedida:

Implementar middleware de autorización en el backend y verificar el rol y permisos del usuario en cada endpoint sensible.

## Implementación:

* Usar middlewares en Express/Next.js para chequear permisos.
* Definir roles y permisos en la base de datos y en el código.
* Revisar todos los endpoints protegidos.

# 5. Exposición de información sensible en errores o logs

## Problema:

Mensajes de error detallados pueden revelar información interna del sistema.

## Contramedida:

Mostrar mensajes genéricos al usuario y loguear detalles solo en el servidor. Configurar el entorno para no mostrar stack traces en producción.

## Implementación:

* Revisar el manejo de errores en backend y frontend.
* Usar variables de entorno para diferenciar entre desarrollo y producción.
* Configurar herramientas de logging como “Winston” o “bunyan”.

# Fases e iteraciones

Iteración 0: Iniciación y Fundamentos  
Duración: 2 semanas  
Fechas tentativas: 13–24 Oct 2025  
Enfoque: Refinar requerimientos, definir arquitectura, configurar entornos.

Iteración 1: Autenticación y Onboarding  
Duración: 3 semanas  
Fechas tentativas: 27 Oct – 14 Nov 2025  
Enfoque: Implementar registro, login, manejo de sesiones y base del frontend.

Iteración 2: Ciclo de Reservas (Estudiante)  
Duración: 4 semanas  
Fechas tentativas: 17 Nov – 12 Dic 2025  
Enfoque: Desarrollar flujo completo de reserva, notificaciones y pruebas.

Iteración 3: Consola de Encargado y Notificaciones  
Duración: 3 semanas  
Fechas tentativas: 5–23 Ene 2026  
Enfoque: Desarrollar panel de administración y refinar sistema de avisos.

Iteración 4: Endurecimiento y Preparación para Lanzamiento  
Duración: 2 semanas  
Fechas tentativas: 26 Ene – 6 Feb 2026  
Enfoque: Pruebas de rendimiento, documentación, capacitación y despliegue.

Iteración 5: Post-lanzamiento y Escalabilidad  
Duración: 2 semanas  
Fechas tentativas: 9–20 Feb 2026  
Enfoque: Monitoreo, soporte, mantenimiento y planificación de expansión.

# Recursos y herramientas

Equipo de trabajo:

* Product Owner (0.5 FTE): Definir prioridades y validar entregables.
* Tech Lead / Arquitecto (1 → 0.5 FTE): Supervisar diseño técnico y revisiones de código.
* Frontend Engineer (1 FTE): Desarrollo de interfaz (Next.js, Tailwind).
* Backend Engineer (1 FTE): API, lógica de negocio, base de datos (Node.js, Supabase).
* QA Engineer (0.5 → 1 FTE): Diseño y ejecución de pruebas automáticas.
* UX/UI Designer (0.5 FTE): Diseño de experiencia e interfaz accesible.
* DevOps (0.25 FTE): CI/CD, infraestructura, monitoreo.
* Technical Writer (0.25 FTE): Documentación técnica y manuales.

Infraestructura y herramientas:

* Control de versiones: GitHub (ramas protegidas, PR obligatorios).
* Gestión del proyecto: Jira / Linear.
* Diseño: Figma con librería de componentes.
* CI/CD: GitHub Actions (lint, test, build, deploy).
* Pruebas: Jest, React Testing Library, Cypress, Postman/Newman.
* Hosting: Supabase (auth y DB), Vercel/Render (frontend), Kubernetes (producción).
* Monitoreo: Sentry, Grafana, Loki.
* Seguridad: Dependabot, Snyk, gestión de secretos cifrados.

# Detalle de iteraciones y entregables

Iteración 0 – Iniciación y Fundamentos  
Objetivo: Establecer bases técnicas y de gestión.  
Actividades clave:

* Talleres con stakeholders y definición de requisitos.
* Diseño de arquitectura y decisión de tecnologías.
* Configuración de entornos de desarrollo y CI/CD.
* Creación de guía de estilos y sistema de diseño (RNF1–RNF2).  
  Entregables: Documento de arquitectura, backlog priorizado, entorno funcional inicial, pipeline configurado.

Iteración 1 – Autenticación y Onboarding  
RF involucrados: RF1–RF5, RF25  
RNF asociados: RNF3–RNF5  
Entregables:

* Registro, inicio y recuperación de sesión con Supabase Auth.
* Pantallas de perfil y navegación principal.
* Base de accesibilidad y responsividad.
* Pruebas unitarias e integrales para flujo de login.
* Documentación técnica (API Auth, manual de instalación).

Iteración 2 – Ciclo de Reservas (Estudiante)  
RF involucrados: RF6–RF19  
RNF asociados: RNF1–RNF7  
Entregables:

* Creación, edición y cancelación de reservas.
* Sistema de notificaciones (correo e in-app).
* Lógica de listas de espera y estados de reserva.
* Pruebas automáticas (unitarias, e2e, rendimiento).
* Guía de uso para estudiantes y documentación técnica.

Iteración 3 – Consola de Encargado y Notificaciones  
RF involucrados: RF20–RF25  
Entregables:

* Panel administrativo con filtros y reportes.
* Roles y permisos (control de acceso).
* Mejoras al sistema de notificaciones.
* Manual de uso para encargados.
* Tests de carga y regresión.

Iteración 4 – Hardening y Preparación de Lanzamiento  
Actividades clave:

* Auditorías de seguridad y rendimiento.
* Pruebas de compatibilidad multi-dispositivo.
* Documentación final del sistema.
* Capacitación interna y validación de backups.
* Ensayo de despliegue y rollback planificado.

Iteración 5 – Post-lanzamiento y Escalabilidad  
Objetivos:

* Monitorear métricas de uso y rendimiento.
* Resolver incidencias y aplicar hotfixes.
* Redactar informe de mantenimiento y lecciones aprendidas.
* Planificar mejoras y posibles integraciones futuras (RB2).

# DER