ELEVATOR PITCH

La problemática que se presenta en las eps en Colombia con los sistemas de turnos, que muchas veces estos están desorganizados, se saltan turnos o los usuarios no sabemos en qué posición estamos o cuanto falta para que nos llamen, por lo que hemos decidido desarrollar un sistema de turnos que nos ofrecerá tanto la visión de los usuarios y como también la visión del agente que atiende.

USUARIOS

- **Afiliado**
- **Agente de atención al cliente**
- **Administrador**

CASOS DE USO

1. **MARCAR DILIGENCIA A REALIZAR**
   Actores: Cliente Activo
   Descripción: Permite al usuario activo en la entidad seleccionar que tipo de tramite quiere realizar para el cual necesita el turno (quejas, reclamos, citas, entre otros)
   Escenario:
   A. El actor inicia sesión colocando su documento de identidad
   B. El actor visualiza las opciones de tramites disponibles
   C. El actor selecciona la adecuada para su caso
2. **ESCOGER TIPO DE SOLICITUD A ATENDER**
   Actores: Agente
   Descripción: Permite al agente escoger entre todos los tipos de solicitudes activas cual quiere atender.
   Escenario:
   A. El actor inicia sesión con su usuario y contraseña
   B. El actor da click en la opción “solicitudes del día”
   C. El actor selecciona la solicitud a atender asignada.
3. **VISUALIZAR POSICION DEL TURNO**
   Actores: Cliente Activo
   Descripción: Permite al usuario visualizar en que posición se encuentra su turno para ser llamado, que turnos están antes y cual estará después.
   Escenario:
   A. El actor obtiene su turno del validador
   B. El actor da click en la opción “observar turnos anteriores”
   C. El actor visualiza a cuantas posiciones de ser llamado se encuentra
4. **GESTIONAR CUENTA**
   Actores: Administrador
   Descripción: El actor tendrá el acceso para visualizar las cuentas tanto de clientes activos como de agentes y las podrá gestionar según las condiciones de estas.
   Escenario:
   A. El actor inicia sesión en el sistema con su usuario y contraseña.

B. El actor podrá acceder a las cuentas a través del botón “gestionar cuenta de cliente” o “gestionar cuenta de agente”.
C. El actor bloquea cuentas inactivas o con actividad sospechosa.
D. El actor crea cuentas dando click en los botones “crear nuevo cliente” o “crear nuevo agente”.
E. El actor elimina cuentas sin actividad.

OBJETIVOS:

- Desarrollar una interfaz intuitiva para que usuarios de cualquier edad puedan utilizarla sin ayuda.
- Crear un aplicativo rápido y con poco margen de error.
- Automatizar la creación y gestión de turnos para reducir tiempos de espera.
- Reducir costos operativos relacionados con la gestión manual de turnos.
- Proporcionar un panel de administración que permita a la empresa visualizar métricas de atención, tiempos de espera y carga de trabajo del personal.
- Permitir la reserva de turnos en línea mediante dispositivos móviles o web, brindando comodidad al usuario.

NO OBJETIVOS:

- No priorizar usuarios por encima de otros, con el aplicativo no se busca crear algún tipo de preferencia o prioridad entre usuarios sea cual sea su situación.
- No sustituirá el servicio humano, con el aplicativo no se busca dar respuesta a solicitudes puntuales o que conlleven datos sensibles de los clientes para eso estarán los agentes.
- No funcionara sin verificaciones de seguridad, ya que los datos que se necesitan para la atención del público son de carácter personal y muchas veces delicados, el sistema debe de asegurar que la persona sea la correcta.
- No disminuye la cantidad de personas que se encuentran presentes, el sistema ayudara a bajar la congestión o la incertidumbre mas no el número de personas que asistan.

MÉTRICAS/KPIs DE EXITO

A través de las siguientes KPIs evaluaremos si nuestra web funciona de la manera correcta:

- **Pruebas de Rendimiento:** Utilizaremos varios dispositivos y sistemas operativos para evaluar la eficacia del sistema, que este funcione sin errores y haga lo que ha sido destinado para hacer.
- **Tiempo de Respuesta al Error:** Analizaremos el tiempo de respuesta del programa para detectar cuando ocurran errores y como este los manejara.

- **Velocidad de Desarrollo:** Evaluaremos la velocidad del equipo de trabajo para completar tareas planteadas en un tiempo determinado.
- **Tiempo Medio de Recuperación:** Mediremos el tiempo en el que el programa tardara en recuperarse después de haberse presentado una falla.

HISTORIAS PRIORIZADAS (MOSCOW)

1. **MUST HAVE**
   - Como cliente activo quiero poder tramitar mi turno desde mi computador o celular para evitarme las filas.
     Criterios de Aceptación:
     - ❖ El sistema debe permitir tramitar turnos desde un computador o un dispositivo móvil con acceso a internet.
     - ❖ El sistema debe mostrar al cliente los trámites o servicios disponibles para seleccionar el que desea realizar.
     - ❖ El sistema debe validar la disponibilidad de turnos antes de confirmar la solicitud.
     - ❖ El sistema debe generar una confirmación de turno con número, fecha, hora y lugar asignado.

   - Como cliente activo quiero poder registrar mis datos en el sistema para acceder al sistema de turnos.
     Criterios de Aceptación:
     - ❖ El sistema debe preguntar por nombre, número de identificación, celular y correo.
     - ❖ Se valida que los datos no existan en el sistema
     - ❖ El sistema envía confirmación por correo electrónico o celular, según el usuario elija.

   - Como cliente activo quiero recibir alertas de cuánto tiempo falta para mi turno.
     Criterios de Aceptación:
     - ❖ El sistema debe enviar notificaciones 30,15 y 5 minutos antes del turno.
     - ❖ Las alertas no podrán desactivarse a menos que se cancele o se reprograme el turno.

   - Como cliente activo quiero conocer cuántos turnos hay antes de mí.
     Criterios de Aceptación:
     - ❖ El sistema debe mostrar al cliente el número exacto de turnos que hay antes de él en la fila.
     - ❖ El sistema debe actualizar la información en tiempo real si los turnos avanzan o alguien cancela.
     - ❖ El sistema debe calcular y mostrar un tiempo estimado de espera basado en la duración promedio de los turnos anteriores.

- Como agente de servicio al cliente quiero poder seleccionar que tipo de gestión quiero atender al día.
  Criterios de Aceptación:
  - ❖ El sistema debe de mostrar una vista con las diferentes opciones de gestión para el agente.
  - ❖ El sistema debe permitir al agente seleccionar una o más opciones de gestión en el día.
  - ❖ El sistema debe permitir modificar la selección del agente durante la jornada si no hay muchas personas para una gestión.

- Como administrador quiero tener un acceso completo al sistema para gestionar las cuentas sin problemas.
  Criterios de Aceptación:
  - ❖ El sistema debe permitir al administrador crear cuentas nuevas de agente y cliente.
  - ❖ El sistema debe permitir al administrador bloquear cuentas ya sea de agente y/o cliente con actividad sospechosa o datos incorrectos o incompletos.
  - ❖ El sistema debe permitir al administrador hacer actualizaciones de datos o cambios de roles en cuentas existentes.

2. SHOULD HAVE

- Como cliente activo quiero poder cancelar o reprogramar mi turno ante cualquier eventualidad.
  Criterios de Aceptación:
  - ❖ El sistema debe permitir al cliente cancelar su turno hasta una hora antes.
  - ❖ El sistema debe permitir al cliente reprogramar los turnos de un solo cliente hasta 2 veces por día.
  - ❖ El sistema debe notificar la cancelación o reprogramación del turno.

- Como administrador quiero tener reportes de los tiempos de espera y cuál fue el tipo de trámite que más se realizó por día.
  Criterios de Aceptación:
  - ❖ El sistema debe permitir al administrador descargar un informe con los tiempos de espera de cada gestión al finalizar cada día.
  - ❖ El sistema debe de permitir al administrador filtrar los reportes por fechas, horas y tipo de trámite.

3. COULD HAVE

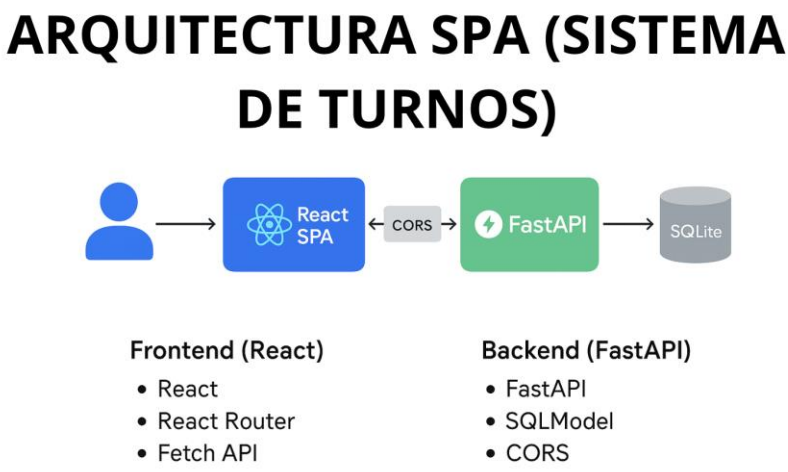
- Como medico quiero tener acceso a los turnos que se me fueron asignados.
  Criterios de Aceptación:
  - ❖ El sistema debe permitir al médico cancelar o reprogramar turnos.
  - ❖ El sistema debe de actualizar la agenda según la actividad del médico.

4. WON'T HAVE

- Como cliente activo quiero tener acceso a mi historia clínica.
- Como cliente activo quiero poder sacar turno para otra persona con mis datos.

ARQUITECTURA Y DECISIONES TÉCNICAS

El diagrama muestra el flujo principal de **Sistema de Turnos**: un usuario usa la **SPA en React** para gestionar su turno; la SPA realiza peticiones HTTP al **backend en FastAPI** protegido por configuración de **CORS** (permitiendo http://localhost:3000 durante desarrollo); el backend persiste y consulta datos en **SQLite** mediante SQLModel. Esta arquitectura permite ofrecer a los afiliados y agentes visibilidad en tiempo real sobre la posición del turno y reducir desorganización y tiempos de espera.



2. Librerías clave y justificación

Frontend

- **React / React-DOM**: núcleo de la SPA y renderizado de componentes.
- **React Router DOM**: gestión de navegación y rutas (Login, ClientePage, TrabajadorPage, AdminPage).
- **Bootstrap + React Bootstrap**: estilos y componentes UI listos para usar.
- **Testing Library**: pruebas unitarias y de integración.

Justificación: se eligieron librerías estándar, ligeras y ampliamente soportadas, que permiten rapidez en el desarrollo y una curva de aprendizaje baja.

Backend

- **FastAPI**: framework principal para endpoints REST.

- **Pydantic / SQLModel**: validación de datos y mapeo objeto-relacional.
- **SQLite**: base de datos ligera y embebida.
- **CORS Middleware**: habilita la comunicación entre frontend y backend.
- **Uvicorn**: servidor ASGI para desarrollo y producción.

Justificación: FastAPI y SQLModel permiten validaciones automáticas, documentación integrada y persistencia sencilla en SQLite, lo que asegura rapidez de desarrollo y confiabilidad.

3. Estrategia de estado en frontend

La aplicación utiliza **estado local con hooks nativos de React** (useState, useEffect) dentro de cada página o componente.

- El Login maneja el estado de credenciales y validaciones.
- En las páginas de cliente, trabajador y administrador, cada vista controla sus formularios y datos obtenidos del backend.
- La sincronización se realiza mediante llamadas HTTP (fetch/axios) dentro de useEffect, que actualizan el estado local con la respuesta de la API.

No se emplean librerías externas de gestión de estado (como Redux o Zustand), ya que la complejidad actual del sistema no lo requiere. La estrategia de hooks locales mantiene la aplicación ligera, clara y fácil de mantener.

4. Manejo de errores y patrones de respuesta

El sistema gestiona los errores más comunes en su API con FastAPI:

- **422 (Error de validación):** ocurre automáticamente cuando los datos enviados no cumplen el modelo definido. Ejemplo: falta el campo cliente al crear un turno.
- **404 (No encontrado):** ya implementado en DELETE /turnos/{id} y PUT /turnos/{id}/asignar. El mensaje devuelto es: *"Turno no encontrado"*.
- **409 (Conflicto):** aún no implementado, pero se planea añadirlo en la creación de turnos para evitar duplicados de cliente y hora.

Actualmente, los endpoints responden con objetos JSON simples (el turno creado, lista de turnos o confirmación). Como mejora futura se plantea un **patrón de respuesta uniforme** con los campos status, data y message, lo que permitirá al frontend manejar de forma consistente tanto casos de éxito como de error.