# 1. Estructura general y flujo de la app

## ### ¿Qué es este sistema?

Es una aplicación web para reservar turnos en una cooperativa. Permite a los usuarios:

- Seleccionar sucursal, fecha y horario.

- Completar sus datos personales.

- Reservar un turno (que se guarda en un backend).

- Consultar o cancelar un turno existente por ID.

## ### ¿Cómo está organizada la app?

Tu proyecto tiene tres grandes componentes:

### 1. \*\*HTML (index.html)\*\*

Define la estructura visual y los elementos de la página:

- Encabezado con logo y título.

- Formulario para solicitar turnos.

- Sección para consultar/cancelar turnos.

- Mensajes y enlaces.

### 2. \*\*CSS (estilos.css)\*\*

Aplica los estilos visuales:

- Colores, fuentes, tamaños, márgenes, bordes.

- Layout responsive (adaptable a móvil y escritorio).

- Apariencia de botones, inputs, encabezados, etc.

### 3. \*\*JavaScript (main.js)\*\*

Controla la lógica y la interacción:

- Carga sucursales y horarios desde el backend.

- Valida los datos del formulario.

- Envía y recupera turnos usando la API.

- Muestra mensajes de éxito/error.

- Permite consultar y cancelar turnos.

## ### ¿Cómo es el flujo típico para un usuario?

1. El usuario entra a la página y ve el logo, el título y el formulario.

2. Selecciona sucursal, fecha y horario (solo opciones válidas).

3. Completa sus datos personales.

4. Hace clic en “Reservar Turno”.

5. El sistema valida los datos y, si todo está bien, guarda el turno en el backend.

6. El usuario recibe un comprobante y puede consultarlo o cancelarlo más tarde usando su ID.

## ### ¿Cómo se conectan los archivos?

- El HTML carga el CSS para los estilos y el JS para la lógica.

- El JS manipula el DOM (elementos HTML) y aplica la lógica de negocio.

- El CSS asegura que todo se vea profesional y coherente con la identidad de la cooperativa.

# 2. Análisis del HTML (index.html)

## a) Estructura básica y cabecera (<head>)

<!DOCTYPE html>: Indica que es un documento HTML5.

<html lang="es">: El idioma principal es español.

<head>:

Metadatos (charset, viewport para responsive).

Título de la página.

Enlaces a hojas de estilo (CSS propio y librerías externas como SweetAlert2 y Toastify para notificaciones).

Favicon (icono de la pestaña).

## b) Cuerpo principal (<body>)

<div class="container">:

Un contenedor central que agrupa todo el contenido principal, centrado y con fondo blanco.

### 1. Encabezado (<header>)

Logo:

El logo es clickeable y lleva al sitio oficial.

Título principal:

<h1>RESERVA DE TURNOS</h1>

Instrucción:

<h2>Para gestionar un turno complete todos los datos y haga click en <span>"Reservar Turno"</span></h2>

Enlace de consulta/cancelación:

Un párrafo con un link para consultar o cancelar turnos existentes.

### 2. Separador visual

<hr style="margin: 30px 0;">:

Línea horizontal para separar el encabezado del formulario.

### 3. Formulario de solicitud de turno

<h3>Datos Personales</h3>:

Subtítulo para la sección de datos personales.

<form id="turnoForm">:

El formulario principal donde el usuario ingresa sus datos:

Nombre completo

DNI

Teléfono

Email

<h3>Sucursal</h3>:

Subtítulo para la selección de sucursal (y luego fecha y horario).

Aquí seguirán los campos para seleccionar sucursal, fecha, horario, y el botón para reservar.

### 4. Sección de consulta/cancelación

Más abajo (no mostrado en el fragmento), suele haber un formulario o sección para ingresar el ID del turno y consultarlo o cancelarlo.

### 5. Mensajes y feedback

Espacios reservados para mostrar mensajes de éxito, error o información al usuario (por ejemplo, después de reservar un turno).

Resumen visual de la estructura

<html>

<head> ... </head>

<body>

<div class="container">

<header>

[Logo] [Título] [Instrucción] [Enlace consulta]

</header>

<hr>

<h3>Datos Personales</h3>

<form id="turnoForm">

[Inputs de datos personales]

<h3>Sucursal</h3>

[Inputs de sucursal, fecha, horario]

[Botón reservar]

</form>

[Sección consulta/cancelación]

[Mensajes]

</div>

</body>

</html>

# 3. Análisis del formulario de turnos

## a) Sección de Sucursal

h3: Título de la sección.

select#sucursal: Menú desplegable para elegir la sucursal.

Las opciones se cargan dinámicamente con JS desde el backend.

div#direccionSucursal: Muestra la dirección de la sucursal seleccionada (se actualiza automáticamente).

## b) Sección de Fecha y Horario

h3: Título de la sección.

h4/h5.mensaje-fecha: Mensajes informativos sobre la selección de fechas y advertencias.

input#fecha: Campo de texto para elegir la fecha.

Se convierte en un selector de calendario (flatpickr) con días hábiles y feriados bloqueados.

select#horario: Menú desplegable para elegir el horario disponible.

Se actualiza según la sucursal y la fecha elegida, bloqueando horarios ocupados.

## c) Botón de envío

Envía el formulario para reservar el turno.

El JS valida los datos y, si todo está correcto, guarda el turno en el backend.

## d) Sección de consulta/cancelación de turno

h3: Título de la sección.

form#formConsultaTurno:

Permite ingresar el ID de un turno para buscarlo.

input#idConsulta: Campo para el ID del turno.

button: Envía la consulta.

div#resultadoConsulta:

Aquí se muestra el resultado de la búsqueda (datos del turno o mensaje de error) y la opción de cancelarlo si corresponde.

## e) Scripts y librerías externas

Al final del archivo se cargan:

Librerías para notificaciones y calendario (SweetAlert2, Toastify, flatpickr).

El archivo principal de JS (main.js), que contiene toda la lógica.

# 4. Análisis del CSS (estilos.css)

## a) Estilos globales y tipografía

Se usa la fuente Roboto para todo el sitio, lo que da un aspecto moderno y legible.

El color de texto principal es oscuro (#222), sobre fondo claro.

El fondo general es una imagen institucional, con un color de respaldo suave (#f7fafd).

## b) Layout y contenedores

.container:

Caja blanca centrada, con bordes redondeados y sombra suave para destacar el contenido sobre el fondo.

El contenido es mobile first: el ancho máximo está en unidades relativas (em), y se adapta a pantallas grandes con media queries.

## c) Encabezados y títulos

h1, h3:

Color de texto rojo oscuro (#b71c1c).

Fondo rojo claro (#fdeaea).

Borde fino rojo oscuro.

Bordes redondeados y padding para dar efecto de “tarjeta”.

h1 ocupa todo el ancho del contenedor, centrado; h3 es más pequeño y suele estar alineado a la izquierda.

h2, h4, h5:

Colores neutros o informativos.

Usados para instrucciones y advertencias.

## d) Formularios e inputs

Inputs y selects:

Bordes rojos, fondo blanco, bordes redondeados.

Sombra muy sutil para dar profundidad.

Al enfocar, el borde se resalta en rojo.

Si hay error, el fondo se tiñe de rojo claro.

Botones:

Fondo rojo principal, texto blanco, bordes redondeados.

Al pasar el mouse, el fondo se oscurece (#b71c1c).

Sombra sutil para resaltar.

## e) Selects personalizados

El select tiene un ícono de flecha personalizado en rojo.

Las opciones seleccionadas o en hover tienen fondo rojo claro y texto blanco.

## f) Mensajes y enlaces

.mensaje-pdf, .mensaje-fecha:

Fondos claros, bordes rojos, padding y bordes redondeados.

.consulta-link a:

Enlaces rojos, subrayados y con efecto hover.

## g) Responsive design

Con media queries, se ajustan tamaños de fuente, paddings y anchos para que el sitio se vea bien en escritorio y móvil.

Resumen visual de la experiencia

Todo el sistema tiene una estética limpia, profesional y coherente con el sitio principal.

Los colores institucionales (rojo y rojo oscuro) predominan en títulos, botones y detalles.

El layout es espacioso, centrado y fácil de usar en cualquier dispositivo.

# 5. Análisis detallado de main.js

## ### 1. Estructura general

Todo el código está dentro de un `DOMContentLoaded`, así se ejecuta solo cuando el HTML ya está cargado.

## ### 2. Manejo de feriados y días hábiles

- \*\*Variables:\*\*

- `feriados`: array donde se guardan las fechas de feriados.

- `hoy`: fecha actual.

- `maxFecha`: fecha máxima permitida (hoy + 30 días).

- \*\*Función `esDiaHabil(fecha)`\*\*

Devuelve `true` si la fecha NO es sábado, domingo ni feriado.

- Usa `getDay()` para ver si es sábado (6) o domingo (0).

- Convierte la fecha a formato `YYYY-MM-DD` y verifica si está en el array de feriados.

- \*\*Función `obtenerFeriados()`\*\*

Pide los feriados del año actual a la API de nolaborables.com.ar. (utilizo para probar un proxy publico temporal paa evitar errores cors, deberia manjerase con proxy propio)

- Si falla, usa un array de feriados locales como fallback.

## ### 3. Inicialización tras cargar feriados

- \*\*Variables de elementos del DOM:\*\*

- `sucursalSelect`, `horarioSelect`, `direccionDiv`, `fechaInput`: referencias a los campos del formulario.

- `fechaSeleccionada`, `sucursalSeleccionada`: para guardar la selección actual.

- \*\*Flatpickr en el input de fecha:\*\*

- Permite elegir solo fechas hábiles (no fines de semana ni feriados).

- Al cambiar la fecha, actualiza los horarios disponibles.

## ### 4. Conexión con el backend (jsonbin.io)

- \*\*Constantes:\*\*

- `URL\_SUCURSALES`, `URL\_TURNOS`: URLs de los bins de sucursales y turnos.

- `API\_KEY`: clave de acceso a la API.

- \*\*Funciones auxiliares:\*\*

- `getTurnosConfirmados()`: obtiene todos los turnos guardados.

- `setTurnosConfirmados(turnos)`: guarda el array de turnos actualizado.

---

## ### 5. Carga dinámica de sucursales

- Hace un fetch a `URL\_SUCURSALES` y llena el select de sucursales.

- Cada opción tiene atributos extra: dirección, teléfonos, id.

- Si falla, muestra un Toast de error.

- \*\*Al cambiar la sucursal:\*\*

- Muestra la dirección y teléfonos.

- Llama a `actualizarHorarios()` para refrescar los horarios disponibles.

## ### 6. Actualización de horarios disponibles

- \*\*Función `actualizarHorarios()`\*\*

- Limpia el select de horarios.

- Busca la sucursal y la fecha seleccionada.

- Obtiene los turnos ya tomados para esa sucursal y fecha.

- Deshabilita los horarios ocupados en el select.

## ### 7. Validación y alta de turno

- \*\*Evento submit del formulario principal:\*\*

- Toma los datos del usuario y los normaliza (quita espacios, deja solo números donde corresponde).

- Valida:

- DNI: solo números, 7-9 dígitos.

- Teléfono: solo números, 8-15 dígitos.

- Email: formato básico.

- Fecha: debe ser hábil.

- Si hay error, muestra Toast y marca el campo en rojo.

- \*\*Chequeo de turnos pendientes por DNI:\*\*

- Si el usuario ya tiene un turno futuro, muestra un mensaje con los datos y permite cancelarlo.

- Si no, pide confirmación para reservar el turno.

- \*\*Al confirmar:\*\*

- Genera un ID único para el turno.

- Abre la constancia en una nueva pestaña.

- Guarda el turno en el backend y en localStorage.

- Muestra Toast de éxito y resetea el formulario.

## ### 8. Consulta y cancelación de turno por ID

- \*\*Formulario de consulta:\*\*

- El usuario ingresa el ID de su turno.

- Busca el turno en el backend.

- Si lo encuentra, muestra los datos y un botón para cancelar.

- Si cancela, elimina el turno del backend y muestra mensaje de éxito.

## ## Resumen de variables y funciones clave

- \*\*Variables:\*\*

- Elementos del DOM (`getElementById`), datos de sucursales, turnos, feriados, selecciones actuales.

- \*\*Funciones:\*\*

- `esDiaHabil`, `obtenerFeriados`, `getTurnosConfirmados`, `setTurnosConfirmados`, `actualizarHorarios`.

- \*\*Condiciones:\*\*

- Validaciones de campos, chequeo de turnos pendientes, control de fechas hábiles, manejo de errores y confirmaciones.