Documentación del Proyecto: Sistema de Reservas

Índice



1. Descripción del Proyecto

El Sistema de Reservas es una aplicación web que permite a los usuarios reservar una sala en una fecha y duración específicas. El proyecto incluye:

- Backend con Node.js y Express.
- Frontend con HTML, CSS y JavaScript para la interfaz de usuario.
- Base de datos MongoDB para almacenar las reservas.

La aplicación sigue un esquema cliente-servidor, donde el frontend envía los datos de la reserva al backend, que a su vez los almacena en MongoDB.

2. Prerrequisitos

Antes de comenzar con la instalación, asegúrate de tener las siguientes herramientas instaladas en tu máquina:

1. Node.js (versión recomendada 14 o superior).
2. MongoDB Community Server
3. Editor de Código: Visual Studio Code (VS Code) es recomendado.

3. Instrucciones de Instalación

3.1 Backend

Paso 1: Crear y Configurar el Proyecto

1. Crear una carpeta para el backend:

mkdir backend

cd backend

2. Inicializar el proyecto:

npm init -y

Esto creará un archivo package.json para gestionar las dependencias del proyecto.

3. Instalar las dependencias necesarias

npm install express mongoose cors express-validator

1. Dependencias:
   - express: Framework para construir el servidor web.
   - mongoose: Biblioteca para conectarse y manejar MongoDB.
   - cors: Habilitar CORS para permitir solicitudes del frontend.
   - express-validator: Validación de datos en las rutas del backend.

Paso 2: Crear la Estructura del Proyecto

1. Crear el archivo principal del servidor

touch app.js

2. Crear la carpeta de modelos

mkdir models

touch models/Reserva.js

Paso 3: Escribir el Código del Backend

1. app.js (servidor): Copia y pega el siguiente código en app.js

2. models/Reserva.js (modelo MongoDB): Copia y pega el siguiente código en models/Reserva.js

3.2 Base de Datos (MongoDB)

Paso 1: Instalar MongoDB

1. Descargar e instalar MongoDB
2. Iniciar MongoDB:

- En Windows, MongoDB podría iniciar automáticamente como un servicio.
- En macOS o Linux, usa el siguiente comando para iniciar MongoDB desde la terminal

3. Verificar que MongoDB esté ejecutándose: MongoDB debería estar corriendo en mongodb://localhost:27017.

4. Modulos

1. Gestión de Salas

Este módulo se encargará de la creación, edición, y eliminación de salas de reuniones, y de asignar atributos relevantes a cada sala.

Características:

- Crear, editar y eliminar salas:
  - Los administradores podrán gestionar las salas a través de una interfaz sencilla.
  - Se requiere un formulario donde se introduzcan los detalles de cada sala.

Atributos de las salas:

- Capacidad: La capacidad máxima de personas que pueden usar la sala.
- Recursos disponibles: Ej. proyector, pizarra, equipo de videoconferencia.
- Ubicación: La ubicación física dentro del edificio o campus.

API Endpoints para Gestión de Salas:

- POST /salas: Crear una nueva sala.
- PUT /salas/:id: Editar una sala existente.
- DELETE /salas/:id: Eliminar una sala.
- GET /salas: Listar todas las salas disponibles.

2. 2. Sistema de Reservas

El sistema de reservas permite a los usuarios ver la disponibilidad de salas y reservarlas sin solapamientos de horario.

Características:

- Visualización de disponibilidad: Los usuarios pueden ver qué salas están disponibles en un calendario.
- Reservas sin conflictos: Las reservas deben verificar la disponibilidad de la sala para evitar conflictos de horario.

- Cancelar y modificar reservas: Los usuarios pueden cancelar o modificar sus reservas.

  API Endpoints para Sistema de Reservas:

- GET /reservas: Ver todas las reservas (con opción de filtro por sala o fecha).
- POST /reservas: Crear una nueva reserva.
- PUT /reservas/:id: Modificar una reserva existente.
- DELETE /reservas/:id: Cancelar una reserva existente.

Lógica para Evitar Conflictos de Horario:

Antes de confirmar una nueva reserva, el backend debe verificar si la sala ya está reservada en el período solicitado

3. Notificaciones

Este módulo gestionará las notificaciones por correo electrónico que se enviarán a los usuarios cuando ocurran eventos como la creación, modificación o cancelación de reservas.

Características:

- Enviar notificaciones al crear/modificar/cancelar: Cuando una reserva se crea o cambia, el sistema enviará un correo al usuario y a los administradores.
- Notificaciones de liberación de salas: Si una reserva es cancelada, se enviará una notificación a los usuarios interesados en esa sala.

Tecnologías para Envío de Correo:

Puedes utilizar una biblioteca como Nodemailer para enviar correos electrónicos

4. 4. Autenticación y Roles

El sistema requiere autenticación y la gestión de roles para diferenciar entre administradores y empleados regulares.

Características:

- Sistema de autenticación de usuarios: Los usuarios deberán registrarse e iniciar sesión para hacer reservas.
- Roles diferenciados:
    - Administradores: Pueden gestionar salas y ver todas las reservas.
    - Usuarios: Pueden crear, modificar y cancelar solo sus reservas.

Tecnologías para Autenticación:

Puedes implementar autenticación mediante JWT (JSON Web Tokens) para manejar las sesiones de usuario.

Login Endpoint

Middleware para Verificación de Roles
5. Reporte de Uso

Los administradores pueden generar informes que muestren el uso de las salas a lo largo del tiempo.

Características:

- Reporte de frecuencia de uso: Cuántas veces ha sido usada cada sala.
- Horas reservadas por sala: Total de horas reservadas en cada sala en un período determinado.

API Endpoint para Reportes:

- GET /reportes/uso: Obtener estadísticas sobre el uso de las salas.

Consulta de Datos:

La consulta para generar reportes podría agrupar las reservas por sala y calcular la suma de la duración de las reservas

3.3 Frontend

Paso 1: Crear y Configurar el Proyecto

1. Crear una carpeta para el frontend:

   mkdir frontend

   cd frontend

2. Crear los archivos HTML y JS necesarios:

touch index.html

touch script.js

Paso 2: Escribir el Código del Frontend

1. index.html: Copia y pega el siguiente código en index.html
2. script.js: Copia y pega el siguiente código en script.js

4. Configuración

4.1 Conexión a MongoDB

La conexión a MongoDB ya está configurada en el archivo app.js en la siguiente línea:

Asegúrate de que MongoDB esté corriendo en localhost en el puerto 27017

4.2 Configuración de CORS

CORS ya está configurado en el backend para permitir las solicitudes desde el frontend

app.use(cors());