**I. Estructura General del Proyecto Backend**

* **Propósito:** Organizar y estructurar la API del sistema de turismo inmersivo.
* **Tecnologías:** Node.js, Express.js y MySQL.

**Estructura de Carpetas:** roma-virtual-backend/

├── server.js // [1] Punto de entrada de la aplicación

├── db.js // [2] Conexión a la base de datos

├── config/ // [3] Configuración de la aplicación

│ └── dbConfig.js // (Opcional) Configuración específica de la DB

│ └── .env // (Opcional) Variables de entorno

├── models/ // [4] Modelos de datos (si se usa un ORM/ODM)

│ ├── Destino.js

│ └── Reserva.js

├── controllers/ // [5] Lógica de la aplicación

│ ├── destinosController.js

│ └── reservasController.js

├── routes/ // [6] Rutas de la API

│ ├── destinos.js // Rutas para destinos

│ ├── reservas.js // Rutas para reservas

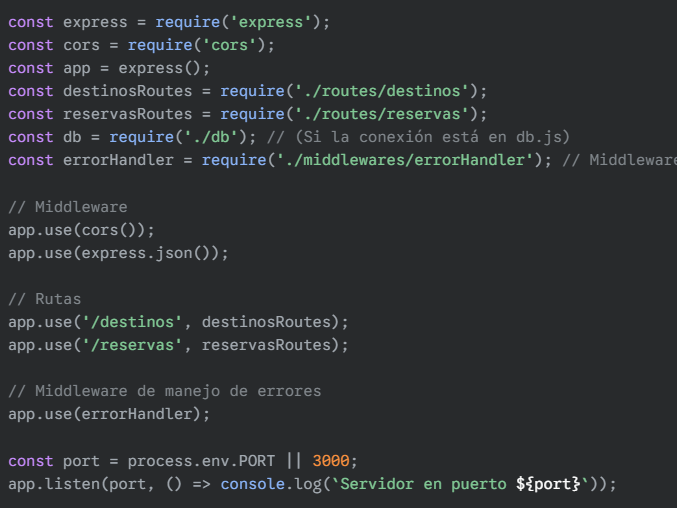
│ └── auth.js // Rutas de autenticación (si se implementa)

├── middlewares/ // [7] Middleware (autenticación, errores, etc.)

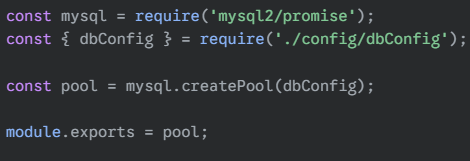
│ └── authMiddleware.js

└── package.json // Archivo de configuración de Node.js

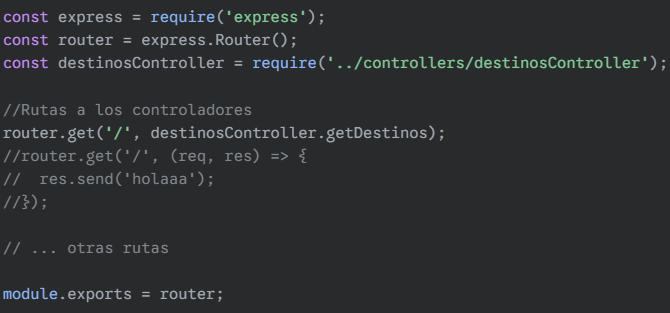
**II. Detalles de la Estructura**

1. **server.js (Punto de entrada):**
   1. **Responsabilidades:**
      * Inicializa Express.
      * Configura el middleware a nivel de aplicación (CORS, Express.json(), etc.).
      * Define las rutas principales o delega a los archivos en routes/.
      * Maneja la conexión a la base de datos.
      * Inicia el servidor (escucha en un puerto).
      * Código de ejemplo:
        1. 

2.**db.js (Conexión a la Base de Datos):**

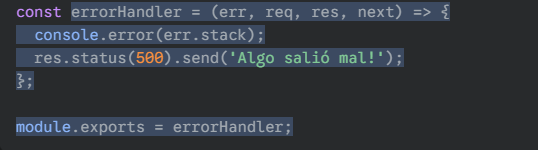
* **Responsabilidades:**
  + Establece la conexión con la base de datos.
  + Exporta el objeto de conexión (o pool) para que lo usen otros módulos.
* **Ejemplo (mysql2/promise):**
  + ****
* **Responsabilidades:** Contiene la lógica de la aplicación para cada recurso. Separa la lógica de las rutas para mejorar la mantenibilidad.

**3.routes/ (Rutas de la API):**

* **Responsabilidades:** Define los endpoints de la API y vincula cada ruta a una función de controlador (si usas controllers/) o contiene la lógica de la API directamente (para aplicaciones más simples).  
  + **Ejemplo (routes/destinos.js):**

**4.middlewares/ (Middleware):**

* **Responsabilidades:** Contiene el middleware personalizado para la aplicación. Por ejemplo, para autenticación, manejo de errores, registro (logging).  
  + **Ejemplo (middlewares/errorHandler.js):**



**ackend (Node.js/Express.js) Estructura:**

* **server.js** (Punto de entrada):  
  + Este es el archivo principal que inicializa Express, configura el middleware y define las rutas de la API.
  + Ejemplo simplificado:  
    
* **routes/destinos.js** (Rutas para destinos):  
  + Este archivo define los endpoints relacionados con los destinos.
  + Ejemplo simplificado:  
    - const express = require('express');
    - const router = express.Router();
    - const db = require('../db');
    - router.get('/', async (req, res) => { /\* ... Obtener todos los destinos \*/ });
    - router.get('/:id', async (req, res) => { /\* ... Obtener un destino por ID \*/ });
    - router.post('/', async (req, res) => { /\* ... Crear un nuevo destino \*/ });
    - router.get('/nombre/:nombre', async (req, res) => { /\* ... Obtener destino por nombre \*/ });
    - // …
    - module.exports = router;
    - module.exports = router;
* **routes/reservas.js** (Rutas para reservas):  
  + Este archivo define los endpoints relacionados con las reservas.
  + Ejemplo simplificado:
    - const express = require('express');
    - const router = express.Router();
    - const db = require('../db');
    - router.get('/', async (req, res) => { /\* ... Obtener todas las reservas \*/ });
    - router.post('/', async (req, res) => { /\* ... Crear una nueva reserva \*/ });
    - router.get('/:id', async (req, res) => { /\* ... Obtener una reserva por ID \*/ });
    - // ...
    - module.exports = router;

**Frontend (React) Estructura:**

* **src/App.js:**
  + Este archivo define las rutas de la aplicación React usando react-router-dom.
  + Ejemplo simplificado:
    - import React from 'react';
    - import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
    - import Home from './pages/Home';
    - import DestinoDetalle from './pages/DestinoDetalle';
    - import NuevaReserva from './pages/NuevaReserva';
    - function App() {
    - return (
    - <Router>
    - <Routes>
    - <Route path="/" element={<Home />} />
    - <Route path="/destinos/:id" element={<DestinoDetalle />} />
    - <Route path="/reservas/nuevo/:nombreDestino" element={<NuevaReserva />} />
    - </Routes>
    - </Router>
    - );
    - }
    - export default App;
* **src/pages/NuevaReserva.js:**
  + Este archivo contiene el componente React para crear una nueva reserva.
  + Incluye el formulario para capturar los datos de la reserva y la lógica para comunicarse con la API del backend.
  + Ejemplo simplificado:  
      
    - import React, { useState, useEffect } from 'react';
    - import { useParams, useNavigate } from 'react-router-dom';
    - import axios from 'axios';
    - import { getDestinoPorNombre } from '../services/destinos';
    - function NuevaReserva() {
    - const { nombreDestino } = useParams();
    - const [destino, setDestino] = useState(null);
    - const [fechaInicio, setFechaInicio] = useState('');
    - const [numeroVisitantes, setNumeroVisitantes] = useState(1);
    - const [nombreCliente, setNombreCliente] = useState('');
    - const [correoElectronico, setCorreoElectronico] = useState('');
    - const [telefono, setTelefono] = useState('');
    - useEffect(() => {
    - getDestinoPorNombre(nombreDestino)
    - .then(response => {
    - setDestino(response.data);
    - })
    - .catch(error => {
    - console.error('Error al obtener destino por nombre:', error);
    - });
    - }, [nombreDestino]);
    - const handleSubmit = async (event) => {
    - event.preventDefault();
    - // ... (Lógica para enviar la reserva a la API)
    - };
    - return (
    - <div>
    - {/\* Formulario de reserva \*/}
    - </div>
    - );
    - }
    - export default NuevaReserva;
* **src/services/destinos.js:**
  + Este archivo contiene las funciones que hacen llamadas a la API relacionadas con los destinos.
  + Ejemplo simplificado:
    - import axios from 'axios';
    - const API\_URL = 'http://localhost:3000'; // Define la URL base de la API
    - export const getDestinos = () => {
    - return axios.get(`${API\_URL}/destinos`);
    - };
    - export const getDestinoPorId = (id) => {
    - return axios.get(`<span class="math-inline">\{API\\_URL\}/destinos/</span>{id}`);
    - };
    - export const getDestinoPorNombre = (nombre) => {
    - return axios.get(`<span class="math-inline">\{API\\_URL\}/destinos/nombre/</span>{nombre}`);
    - };
    - // ... (Otras funciones)