\*\*1. Instalación y configuración del proyecto\*\*

Primero, asegurémonos de tener instaladas las herramientas necesarias:

- \*\*Node.js\*\*: para correr nuestro servidor backend.

- \*\*MongoDB Atlas\*\* (o una instancia local de MongoDB): para la base de datos.

- \*\*npm\*\* o \*\*yarn\*\*: para gestionar las dependencias de nuestros proyectos (incluido con Node.js).

- \*\*React\*\*: para crear el frontend.

### Paso a Paso

1. \*\*Crear el Frontend\*\*:

- Crear una carpeta para el proyecto llamada \*\*pagina-registro-sitios\*\*.

- Dentro de la carpeta, crear una subcarpeta llamada \*\*frontend\*\*:

```sh

mkdir pagina-registro-sitios

cd pagina-registro-sitios

npx create-react-app frontend

```

Esto creará un proyecto básico de React dentro de la carpeta \*\*frontend\*\*.

- Dentro del proyecto React, vamos a modificar el componente \*\*App.js\*\* para incluir una estructura básica de la página:

```javascript

import React, { useState, useEffect } from 'react';

import './App.css';

function App() {

const [sitios, setSitios] = useState([]);

useEffect(() => {

// Aquí realizaremos la llamada al backend para obtener los sitios registrados

fetch('http://localhost:5000/sitios')

.then(response => response.json())

.then(data => setSitios(data))

.catch(error => console.error('Error fetching sitios:', error));

}, []);

return (

<div className="App">

<header className="App-header-expanded">

<h1>Explora y Descubre la Belleza de Colombia</h1>

<p>En esta página podrás registrar y explorar diversos sitios turísticos de Colombia, para que otros puedan conocer y disfrutar de las maravillas que ofrece nuestro país.</p>

</header>

<section className="descripcion">

<p>Esta plataforma te permite registrar lugares turísticos y compartir información detallada para ayudar a otros a conocer más sobre Colombia. ¡Comparte los sitios que más te apasionan!</p>

</section>

<section className="registro-formulario">

<h2>Registrar un nuevo lugar</h2>

<form>

<div className="form-group">

<label>Nombre del lugar:</label>

<input type="text" name="nombre" />

</div>

<div className="form-group">

<label>Ubicación:</label>

<input type="text" name="ubicacion" />

</div>

<div className="form-group">

<label>Descripción:</label>

<textarea name="descripcion" style={{ resize: 'none' }}></textarea>

</div>

<div className="form-group">

<label>Imagen URL:</label>

<input type="text" name="imagenUrl" />

</div>

<div className="form-group">

<label>Responsable:</label>

<input type="text" name="responsable" />

</div>

<div className="form-group">

<label>Correo:</label>

<input type="email" name="correo" />

</div>

<div className="form-group">

<label>Teléfono:</label>

<input type="tel" name="telefono" />

</div>

<button type="submit">Registrar</button>

</form>

</section>

<section className="sitios-registrados">

<h2>Sitios Registrados</h2>

<p>A continuación, encontrarás una lista de los sitios registrados por nuestra comunidad. Puedes explorar sus características, ubicación y quién es el responsable de cada uno de estos lugares maravillosos.</p>

<div className="tarjetas-sitios">

{sitios.map((sitio, index) => (

<div key={index} className="tarjeta-sitio">

<h3>{sitio.nombre}</h3>

<img src={sitio.imagenUrl} alt={`Imagen de ${sitio.nombre}`} style={{ width: '100%', height: '150px', objectFit: 'cover' }} />

<p>{sitio.descripcion}</p>

<p><strong>Ubicación:</strong> {sitio.ubicacion}</p>

<p><strong>Responsable:</strong> {sitio.responsable}</p>

<p><strong>Contacto:</strong> {sitio.telefono}</p>

</div>

))}

</div>

</section>

<section className="informacion-admin">

<h2>Información del Administrador</h2>

<p>Nombre: Administrador de la Plataforma Turística de Colombia</p>

<p>Correo: admin@ejemplo.com</p>

<p>Teléfono: 123456789</p>

<p>Gracias por visitar nuestra plataforma y contribuir al crecimiento del turismo en nuestro hermoso país. Estamos comprometidos en hacer que la experiencia sea lo más sencilla y enriquecedora posible para todos los usuarios.</p>

</section>

</div>

);

}

export default App;

```

- Crear y ajustar el archivo \*\*App.css\*\* para dar un estilo más organizado al formulario y las tarjetas:

```css

.App {

font-family: Arial, sans-serif;

max-width: 900px;

margin: 0 auto;

padding: 30px;

background-color: #f5f5f5;

}

.App-header-expanded {

text-align: center;

margin-bottom: 30px;

background-color: #4CAF50;

color: white;

padding: 20px;

border-radius: 10px;

}

.registro-formulario {

background: #f9f9f9;

padding: 25px;

border-radius: 10px;

margin-bottom: 25px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

}

.form-group {

margin-bottom: 20px;

}

.form-group label {

display: block;

font-weight: bold;

margin-bottom: 5px;

}

.form-group input,

.form-group textarea {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 5px;

}

button[type="submit"] {

background-color: #007bff;

color: white;

padding: 12px 24px;

border: none;

border-radius: 5px;

cursor: pointer;

transition: background-color 0.3s;

}

button[type="submit"]:hover {

background-color: #0056b3;

}

.tarjetas-sitios {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 25px;

}

.tarjeta-sitio {

background: #ffffff;

padding: 20px;

border: 1px solid #e0e0e0;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

}

.tarjeta-sitio img {

border-radius: 5px;

margin-bottom: 15px;

}

.informacion-admin {

margin-top: 30px;

background: #f1f1f1;

padding: 20px;

border-radius: 10px;

}

.informacion-admin p {

margin-bottom: 10px;

}

```

2. \*\*Crear el Backend con Node.js y Express\*\*:

- Dentro de la carpeta \*\*pagina-registro-sitios\*\*, crear otra subcarpeta llamada \*\*backend\*\* para contener nuestro servidor:

```sh

mkdir backend

cd backend

npm init -y

```

Esto creará un archivo `package.json` que permitirá gestionar las dependencias de Node.js.

- Instalar Express para crear el servidor y Mongoose para conectar con MongoDB:

```sh

npm install express mongoose cors

```

- Crear un archivo principal para el servidor llamado \*\*server.js\*\*:

```sh

echo > server.js

```

- Dentro de `server.js`, vamos a crear una configuración básica del servidor con Express:

```javascript

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const app = express();

const PORT = 5000;

// Middleware

app.use(cors());

app.use(express.json());

// Inicio del servidor

app.listen(PORT, () => {

console.log(`Servidor corriendo en el puerto ${PORT}`);

});

```

Este código inicia un servidor en el puerto 5000.

3. \*\*Conexión del Backend a MongoDB\*\*

Vamos a usar \*\*MongoDB Atlas\*\* para la base de datos, pero puedes optar por una instalación local si lo prefieres. Asegúrate de reemplazar `mongodb+srv://guerrerotechia:VapxCIoE0kJyuJuf@cluster0.5jw1f.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0` en `server.js` con la URI proporcionada por MongoDB Atlas.

- Actualizar `server.js` para incluir la conexión a MongoDB:

```javascript

mongoose.connect('TU\_MONGO\_URI', { useNewUrlParser: true, useUnifiedTopology: true })

.then(() => console.log('Conectado a MongoDB'))

.catch(err => console.error('Error conectando a MongoDB', err));

```

4. \*\*Extender el Backend para soportar la gestión de sitios\*\*

5. \*\*Definir el Modelo de Datos\*\*

- Crear un archivo dentro de la carpeta \*\*backend\*\* llamado \*\*models/Sitio.js\*\*:

```javascript

const mongoose = require('mongoose');

const sitioSchema = new mongoose.Schema({

nombre: String,

ubicacion: String,

descripcion: String,

imagenUrl: String,

responsable: String,

correo: String,

telefono: String

});

module.exports = mongoose.model('Sitio', sitioSchema);

```

6. \*\*Crear las Rutas para el Backend\*\*

- Crear un archivo dentro de la carpeta \*\*backend\*\* llamado \*\*routes/sitios.js\*\*:

```javascript

const express = require('express');

const Sitio = require('../models/Sitio');

const router = express.Router();

// Ruta para obtener todos los sitios

router.get('/', async (req, res) => {

try {

const sitios = await Sitio.find();

res.json(sitios);

} catch (err) {

res.status(500).json({ message: err.message });

}

});

// Ruta para crear un nuevo sitio

router.post('/', async (req, res) => {

const sitio = new Sitio(req.body);

try {

const nuevoSitio = await sitio.save();

res.status(201).json(nuevoSitio);

} catch (err) {

res.status(400).json({ message: err.message });

}

});

module.exports = router;

```

- Importar estas rutas en \*\*server.js\*\*:

```javascript

const sitiosRouter = require('./routes/sitios');

app.use('/sitios', sitiosRouter);

```