Aplicaciones web orientada a servicios

Proyecto: Plastic Recycler

Universidad Tecnológica Fidel Velázquez

Desarrollo de software en multiplataforma

Integrantes

Jovanny Aldahir Martínez Escudero

Jonathan Yael Jiménez Flores

Maria Fernanda Cañongo Martínez

Carlos Saúl Moreno Delgado

Estrella Lizeth Blancas Vázquez

Tipo de sitio

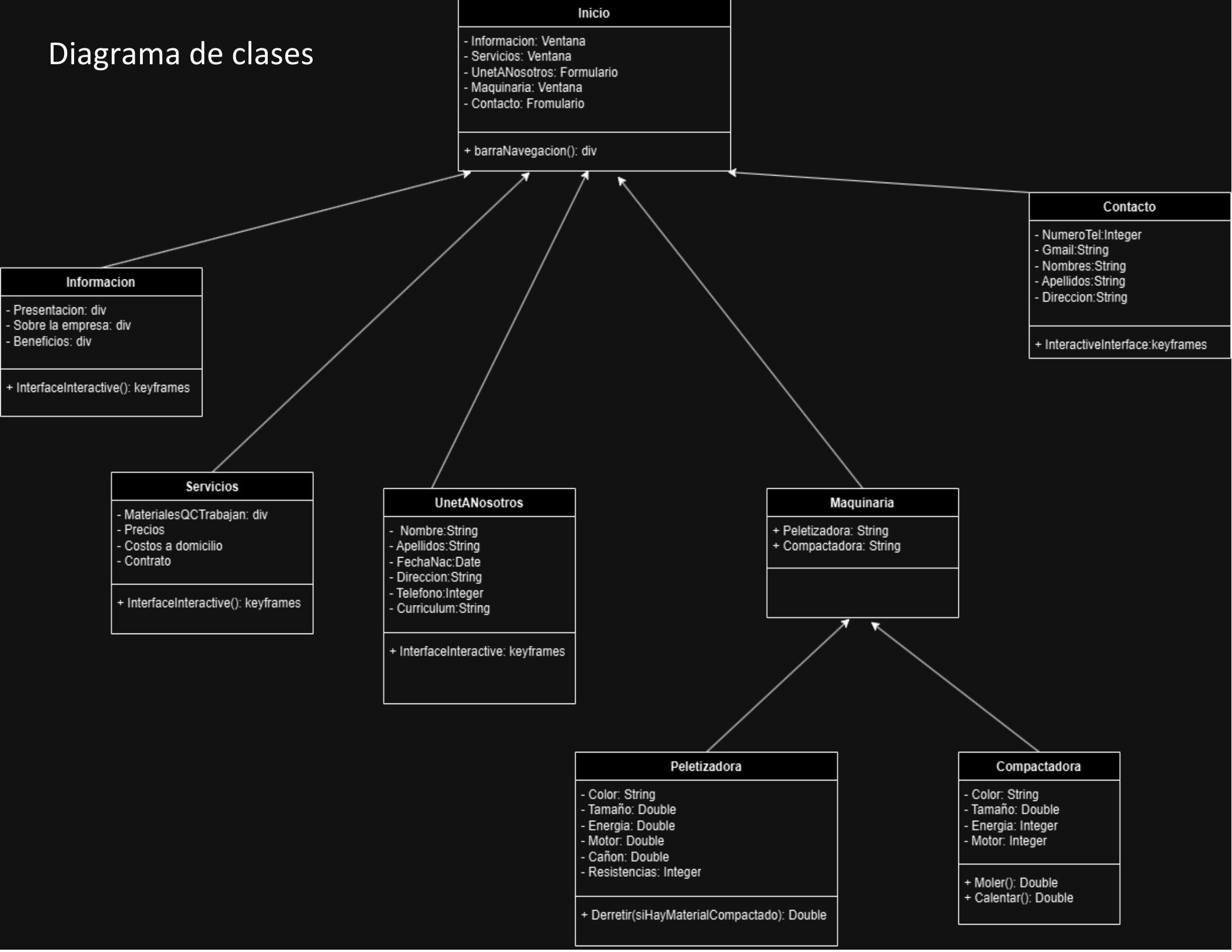
Sitio web tradicional con React en partes específicas

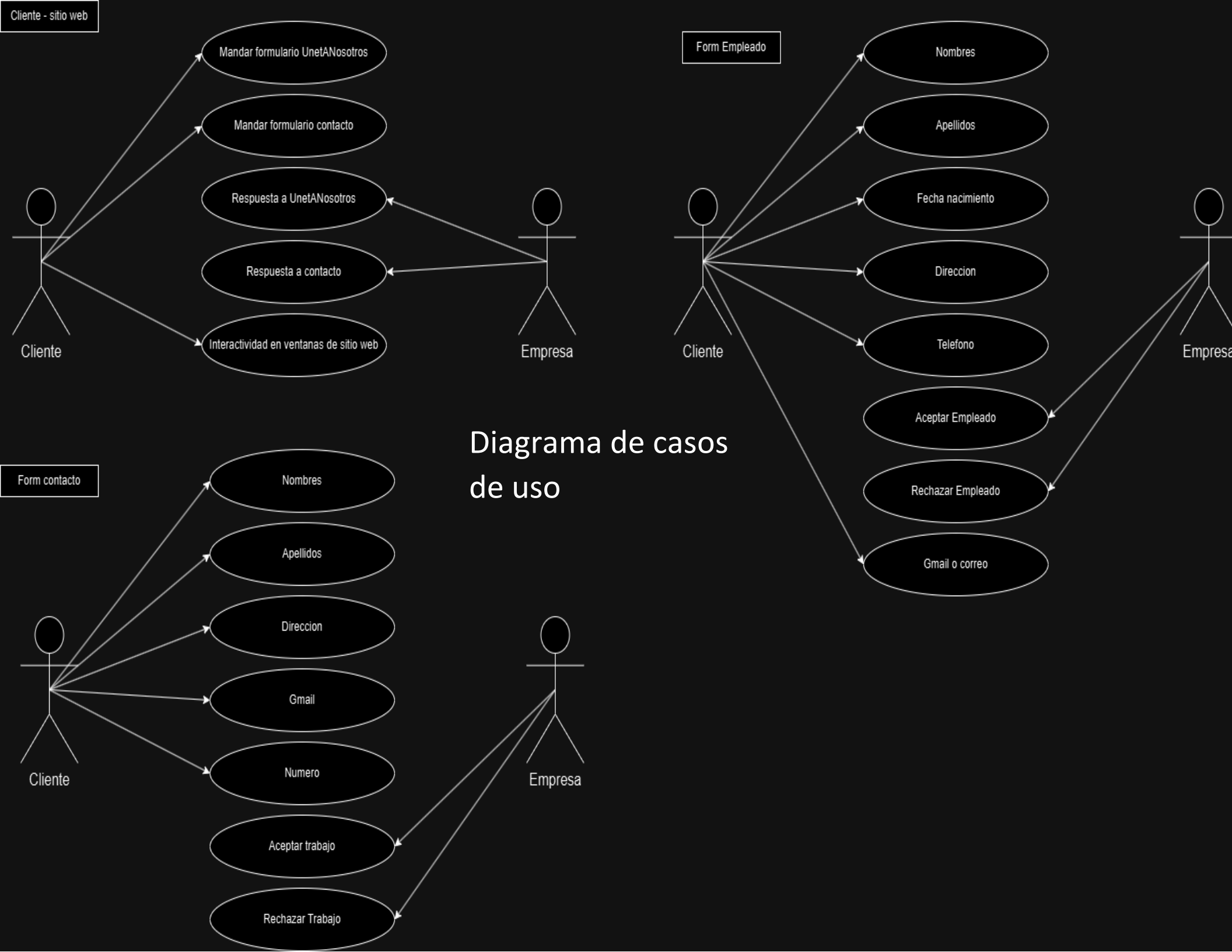
- Cada página es un archivo .html separado.
- Se usa JavaScript y React en algunas secciones para mejorar la interactividad.
- Puede incluir <script> en cada HTML para cargar React de forma independiente.

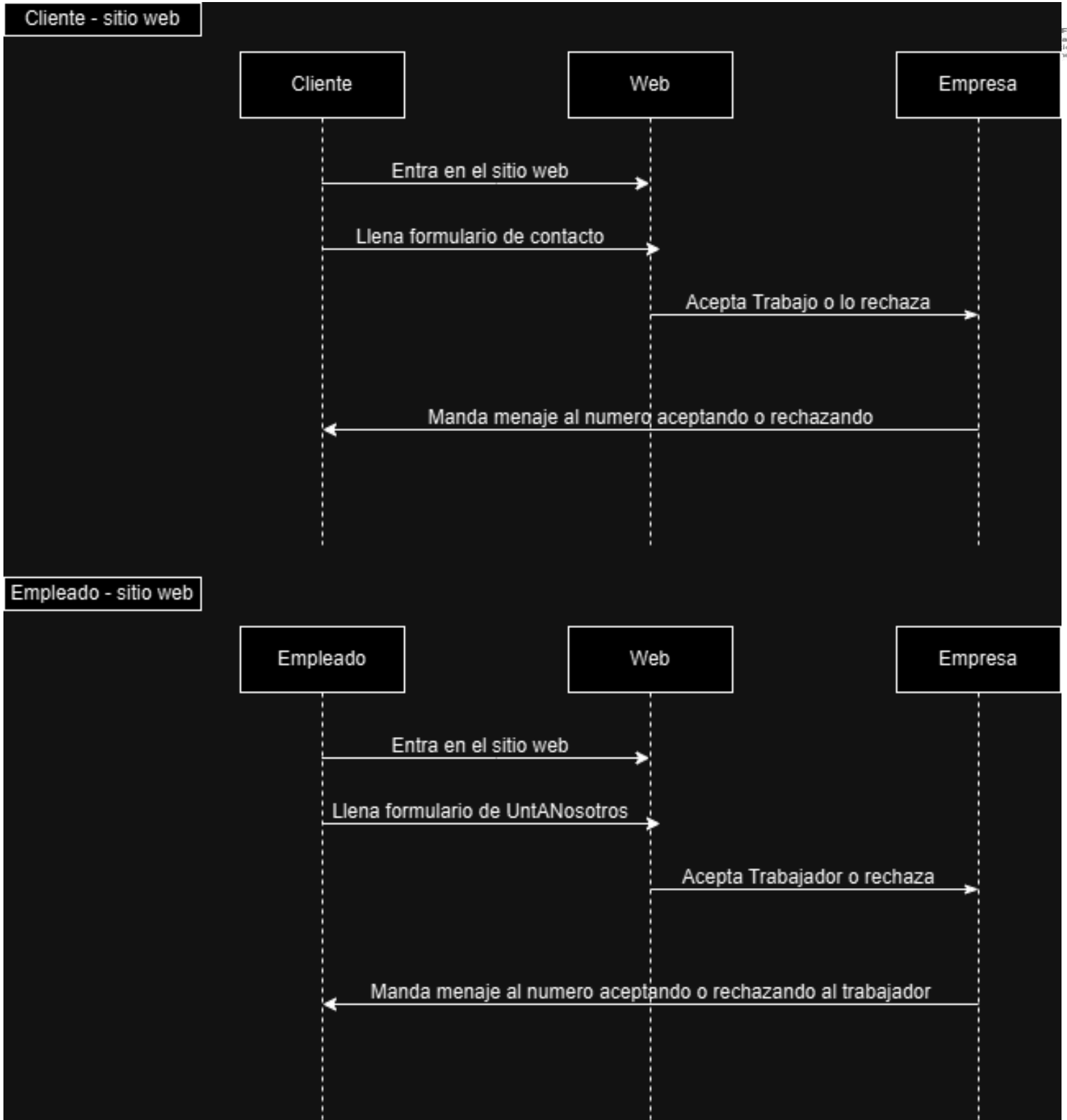
Desarrollo del Back-End con Node.js, Express y MySQL

El Back-End se desarrollará utilizando Node.js con el framework Express.js para manejar las solicitudes HTTP y la comunicación con una base de datos MySQL. Se implementará una API REST, permitiendo que el Front-End (por ejemplo, una aplicación en React) interactúe con la base de datos de manera eficiente.

Diagrama de clases




Cliente - sitio web
Mandar formulario UnetANosotros
Mandar formulario contacto
Respuesta a UnetANosotros
Respuesta a contacto
Interactividad en ventanas de sitio web
Cliente
Empresa
Form Empleado
Nombres
Apellidos
Fecha nacimiento
Direccion
Telefono
Aceptar Empleado
Rechazar Empleado
Gmail o correo
Cliente
Empresa
Diagrama de casos de uso
Form contacto
Nombres
Apellidos
Direccion
Gmail
Numero
Aceptar trabajo
Rechazar Trabajo
Cliente
Empresa


Cliente - sitio web
Cliente
Web
Empresa
Entra en el sitio web
Llena formulario de contacto
Acepta Trabajo o lo rechaza
Manda menaje al numero aceptando o rechazando
Empleado - sitio web
Empleado
Web
Empresa
Entra en el sitio web
Llena formulario de UntANosotros
Acepta Trabajador o rechaza
Manda menaje al numero aceptando o rechazando al trabajador

Mapa de navegación

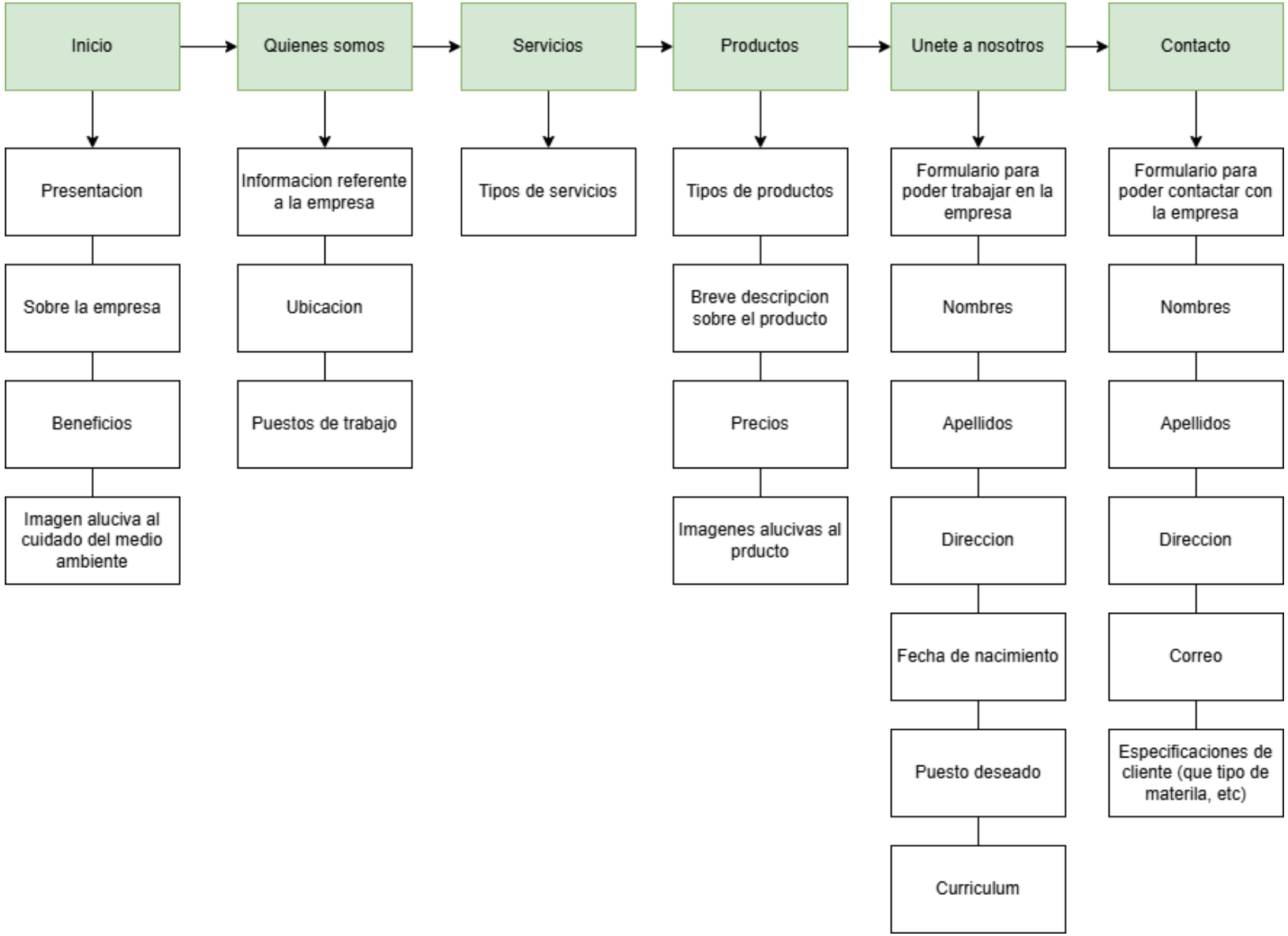
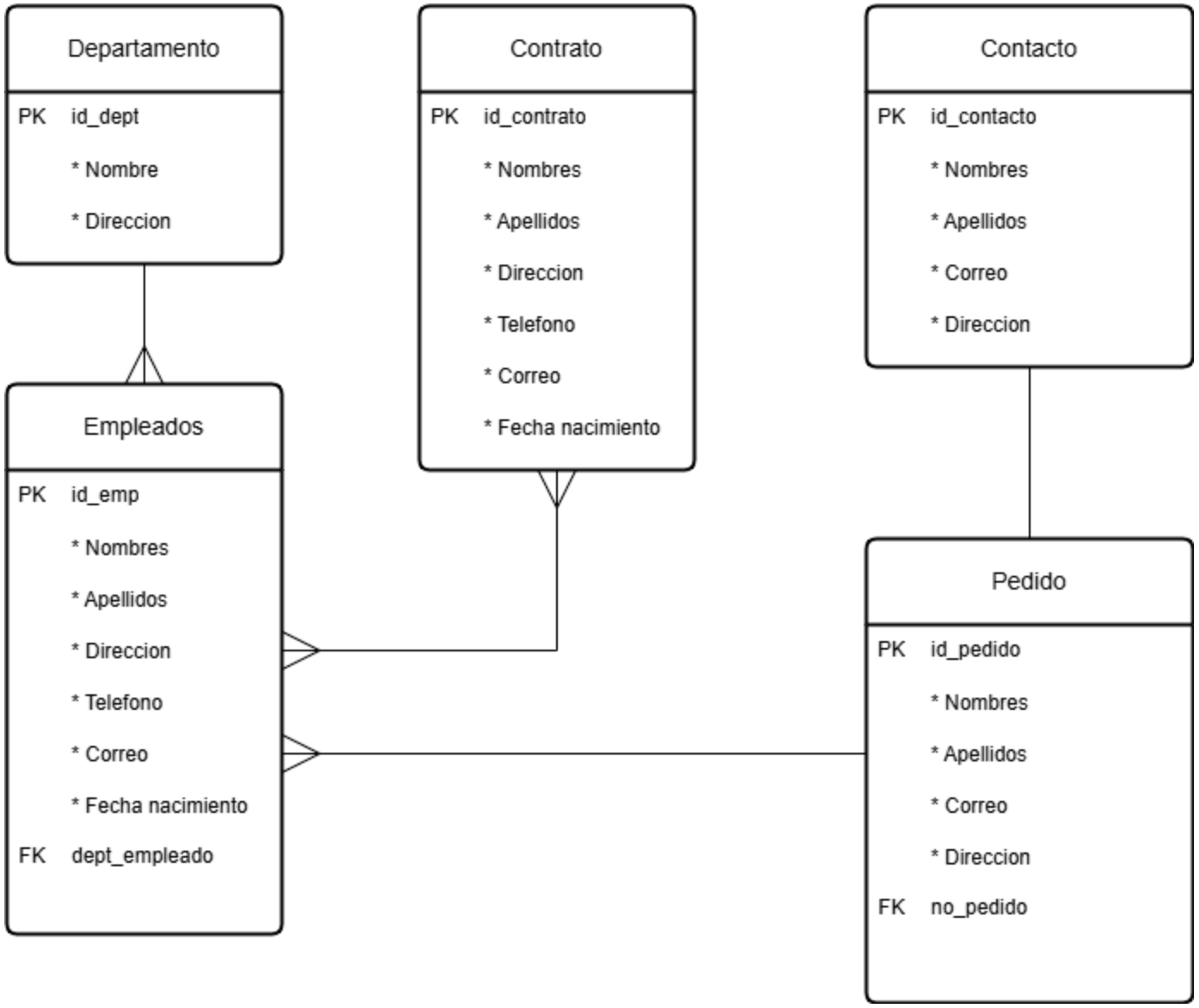


Diagrama entidad relación



Diseño GUI

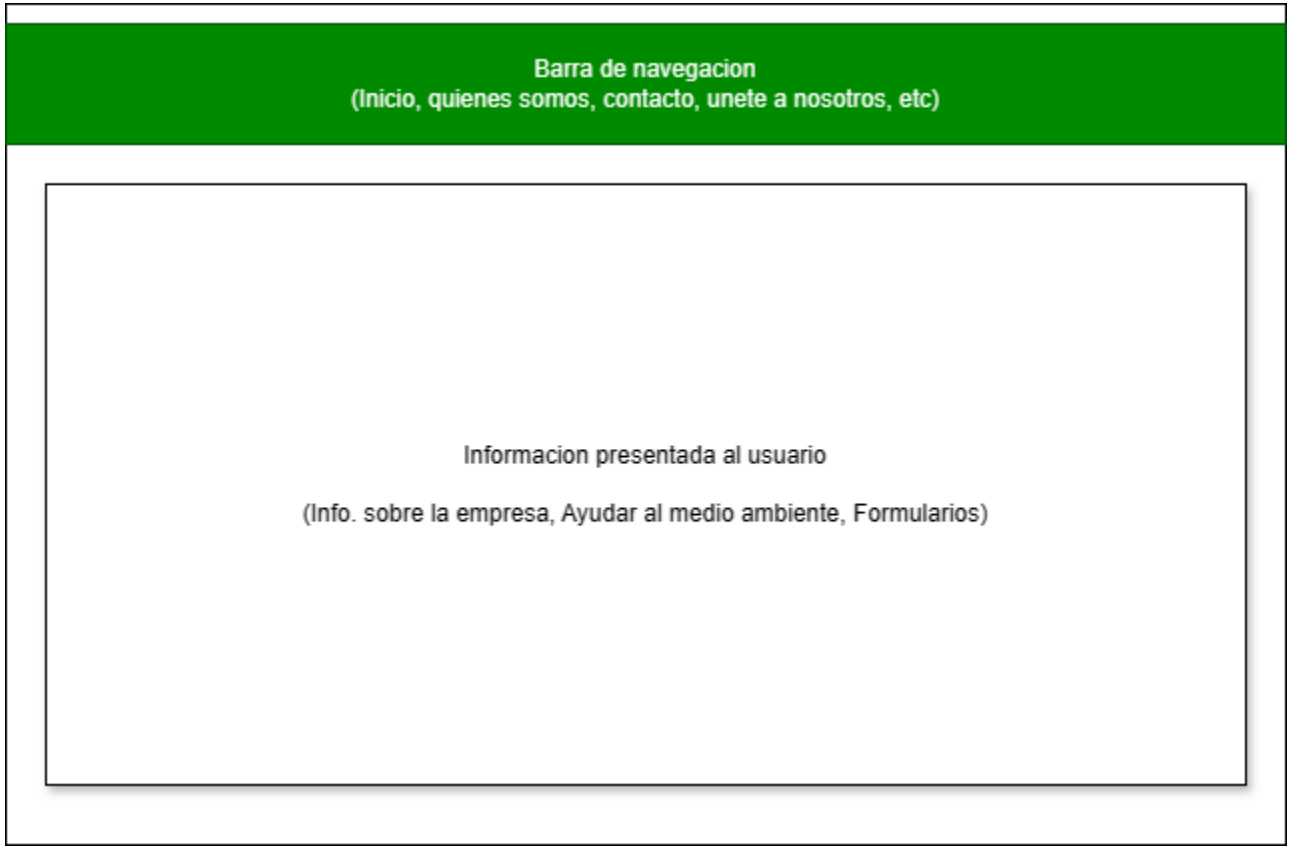

Barra de navegacion
(Inicio, quienes somos, contacto, unete a nosotros, etc)
Informacion presentada al usuario
(Info. sobre la empresa, Ayudar al medio ambiente, Formularios)

Diagrama de componentes

Frontend: Interfaz de usuario (páginas web y formularios).

Backend: Lógica de servidor (procesamiento de datos, validaciones).

Base de datos: Almacenamiento de datos de usuarios, testimonios, y contactos.

Tecnologías Utilizadas (Back-End)

Node.js: Entorno de ejecución para JavaScript en el servidor.

Express.js: Framework para crear el servidor y manejar rutas.

MySQL: Base de datos relacional para almacenar la información.

Sequelize o mysql2: Librerías para conectar Node.js con MySQL.

JWT (JSON Web Token): Para autenticación de usuarios.

2. Arquitectura del Proyecto

El Back-End seguirá la arquitectura MVC (Modelo-Vista-Controlador) para mantener un código estructurado y escalable.

Rutas: Definen los endpoints de la API.

Controladores: Manejan la lógica de las peticiones.

Modelos: Representan las tablas de la base de datos en código.

Base de datos: MySQL almacena los datos de la aplicación.

3. Funcionamiento de la API REST

La API REST permitirá que el Front-End realice operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre la base de datos.

Ejemplo de Endpoints:

GET /users → Obtener todos los usuarios.

GET /users/:id → Obtener un usuario por ID.

POST /users → Crear un nuevo usuario.

PUT /users/:id → Actualizar datos de un usuario.

DELETE /users/:id → Eliminar un usuario.

5. Despliegue y Hosting

El servidor se podrá desplegar en plataformas como Railway, Render o un VPS, asegurando su disponibilidad y acceso desde el Front-End.

Desarrollo del Front-End con JavaScript, CSS, HTML y React

El Front-End de la aplicación se desarrollará utilizando tecnologías modernas para crear una interfaz de usuario interactiva, responsiva y eficiente.

1. Tecnologías Utilizadas (Front-End)

HTML: Estructura principal de la aplicación.

CSS: Estilos y diseño responsivo para mejorar la apariencia.

JavaScript: Lógica de la aplicación y manipulación del DOM.

React: Biblioteca de JavaScript para construir interfaces dinámicas y componentes reutilizables.

2. Arquitectura del Proyecto

El código estará organizado de manera modular, utilizando la estructura de componentes de React.

Componentes: Bloques reutilizables de la interfaz.

Estado y Props: Manejo de datos internos y comunicación entre componentes.

Rutas con React Router: Para manejar la navegación entre páginas sin recargar la web.

Consumo de API: Con fetch o axios para conectar con el Back-End en Node.js.

3. Funcionamiento y Características

Interfaz intuitiva: Diseñada con HTML y CSS, asegurando una buena experiencia de usuario.

Estilos modernos: Uso de Flexbox, Grid y frameworks como Tailwind o Bootstrap si es necesario.

Dinamismo: Uso de eventos, hooks (useState, useEffect) y estado global si la aplicación lo requiere.

4. Conexión con el Back-End

El Front-End se conectará a la API REST creada en Node.js y Express para enviar y recibir datos desde la base de datos MySQL.

Ejemplo de solicitud con fetch:

javascript

Copiar

Editar

```
fetch('https://api.example.com/users')
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

5. Despliegue y Hosting

El Front-End podrá alojarse en InfinityFree, Netlify, Vercel o GitHub Pages, dependiendo de los requerimientos del proyecto.

¿Se retomara algún proyectro?

Si, se retomara el proyecto: Plastic Recycler, y se agrgara una nueva función que es un inicio de sesión además de actualizar las tecnologías con las que estaba hechas anteriormente (html, javascript. Css y php)

1. API de Autenticación y Usuarios (Necesaria para cualquier aplicación con usuarios)

Funcionalidad: Registro, inicio de sesión y gestión de usuarios.

Cómo implementarla:

Usar JWT para autenticación basada en tokens.

Guardar usuarios en MySQL con contraseñas encriptadas (bcrypt.js).

2. API de Productos o Servicios (Si la app maneja elementos como productos, artículos, etc.)

Funcionalidad: Gestión de productos, precios y descripciones.

Endpoints Ejemplo:

GET /products → Obtener todos los productos.

POST /products → Agregar un nuevo producto.

PUT /products/:id → Editar un producto.

DELETE /products/:id → Eliminar un producto.