**1. Páginas y Navegación**

* **Página Principal**:
  + Incluye un **Hero Section** con un buscador funcional para filtrar ciudades.
  + Muestra una cuadrícula de **DestinationCards**, cada una representando una ciudad destacada.
  + Las tarjetas contienen:
    - Imagen representativa.
    - Nombre de la ciudad.
    - Breve descripción.
    - Transición visual al pasar el cursor.
* **Detalles de la Ciudad**:
  + Página dinámica que carga detalles específicos de cada ciudad seleccionada desde la base de datos.
  + Información mostrada:
    - Imagen principal de la ciudad.
    - Descripción general.
    - Lista de atracciones turísticas con imágenes y descripciones.
    - Lista de guías turísticos disponibles en esa ciudad.
* **Perfil del Guía**:
  + Página protegida que solo los usuarios registrados pueden acceder.
  + Muestra información detallada de un guía turístico:
    - Nombre.
    - Experiencia.
    - Idiomas.
    - Botón para regresar.
* **Registro e Inicio de Sesión**:
  + **Registro**:
    - Formulario para que nuevos usuarios puedan registrarse.
    - Validaciones en el frontend para email, contraseñas, etc.
    - Envía datos al backend para guardarlos en la base de datos.
  + **Inicio de Sesión**:
    - Formulario para que los usuarios registrados inicien sesión.
    - Validaciones en el frontend.
    - Al iniciar sesión correctamente, el backend devuelve un **token JWT**, que se guarda en el localStorage.

**2. Backend**

* **Estructura Modular**:
  + Código dividido en carpetas para mantenerlo organizado:
    - **controllers**: Manejan la lógica de las rutas.
    - **routes**: Define las rutas de la API.
    - **config**: Contiene la configuración de la base de datos.
* **Rutas del Backend**:
  + **Ciudades**:
    - GET /api/ciudades: Obtiene todas las ciudades de la base de datos.
    - GET /api/ciudades/:id: Obtiene detalles de una ciudad específica (nombre, descripción, atracciones, guías).
  + **Guías**:
    - GET /api/guias/:id: Obtiene detalles de un guía turístico específico.
  + **Autenticación**:
    - POST /api/auth/register: Registra nuevos usuarios.
    - POST /api/auth/login: Inicia sesión y genera un **token JWT**.
* **Base de Datos**:
  + Tablas implementadas:
    - **ciudades**: Almacena información sobre las ciudades.
    - **atracciones**: Contiene las atracciones turísticas de cada ciudad.
    - **guias**: Información de los guías turísticos.
    - **usuarios**: Registra usuarios para autenticación.
* **Autenticación con JWT**:
  + Los usuarios deben iniciar sesión para acceder a páginas protegidas como el perfil del guía.
  + Los tokens se verifican en el backend para garantizar la autenticidad del usuario.

**3. Funcionalidades Adicionales**

* **Híbrido Backend-Datos de Prueba**:
  + Si el backend no responde, el frontend usa datos de prueba predefinidos.
* **Filtrado de Ciudades**:
  + Un buscador dinámico en la página principal que filtra las tarjetas de las ciudades en tiempo real mientras el usuario escribe.
* **Estilización con Tailwind CSS**:
  + Diseño responsive y moderno.
  + Componentes estilizados:
    - Hero Section.
    - DestinationCards con hover.
    - Formularios para inicio de sesión y registro.
* **Protección de Rutas**:
  + Implementación de un componente ProtectedRoute que redirige a los usuarios no autenticados a la página de inicio de sesión.

### AVANCES: Documento del Proyecto: ****GUIATOURS****

## **Descripción del Proyecto**

**GUIATOURS** es una plataforma web diseñada para conectar turistas con guías turísticos en diferentes ciudades. Los turistas pueden buscar guías según sus preferencias, ver sus perfiles y calificaciones, contactarlos mediante un chat integrado, y contratar sus servicios directamente en la plataforma.

## **Características Implementadas**

### ****Frontend****

1. **Registro e Inicio de Sesión**
   * Registro para usuarios y guías, con validaciones.
   * Diferenciación entre usuarios y guías mediante roles.
   * Sesiones persistentes utilizando tokens almacenados en localStorage.
2. **Navbar Dinámico**
   * Se adapta según el tipo de usuario que haya iniciado sesión (usuario, guía o visitante).
   * Oculta o muestra botones según el rol.
3. **Perfil del Guía (Vista Pública para Usuarios)**
   * Muestra información del guía como:
     + Nombre, especialidad, experiencia, idiomas.
     + Tarifas y descripción del tour.
     + Calificaciones y reseñas de usuarios.
   * Integra un botón para iniciar un chat con el guía.
   * Sistema de reseñas y calificaciones con validación de autenticación.
   * Integración de pagos para contratar guías.
4. **Dashboard del Guía**
   * Muestra métricas personalizadas:
     + Total de contrataciones.
     + Ingresos generados.
   * Buzón de mensajes (chat integrado con usuarios).
   * Botón para editar el perfil del guía.
5. **Chat en Tiempo Real**
   * Botón en el perfil del guía para iniciar un chat.
   * Permite a usuarios y guías intercambiar mensajes.
   * Interfaz amigable y sencilla de usar.
6. **Lista de Ciudades y Atracciones**
   * Página de inicio muestra destinos destacados con información de cada ciudad.
   * Atracciones turísticas relevantes en cada ciudad.

### ****Backend****

1. **Autenticación**
   * Implementación de JWT para proteger rutas.
   * Rutas separadas para el inicio de sesión y registro de usuarios y guías.
2. **Gestión de Perfiles**
   * Endpoints para obtener y actualizar información de usuarios y guías.
3. **Sistema de Chat**
   * Base de datos para almacenar mensajes con relaciones entre usuarios y guías.
   * Endpoints para enviar y recibir mensajes.
   * Validación de datos en cada operación.
4. **Reseñas**
   * Rutas para agregar y obtener reseñas de guías.
   * Validación de tokens para garantizar autenticidad.
5. **Manejo de Pagos**
   * Simulación de un sistema de pagos en el frontend.
   * Rutas para realizar operaciones de contratación.
6. **Estructura Escalable**
   * Dividido en controladores y rutas para mantener la modularidad.
   * Uso de MySQL para gestión de datos.

## **Estructura de la Base de Datos**

### ****Usuarios****

* **Campos:** id, nombre\_completo, email, contrasena\_hash, created\_at.

### ****Guías****

* **Campos:** id, ciudad\_id, nombre, experiencia, idiomas, foto, especialidad, sobre\_mi, tarifa, tour, calificacion, email, whatsapp, contrasena, created\_at.

### ****Reseñas****

* **Campos:** id, guia\_id, usuario\_id, texto, calificacion, fecha.

### ****Mensajes****

* **Campos:** id, usuario\_id, guia\_id, mensaje, timestamp.

## **Cosas Por Hacer**

### ****Frontend****

1. **Solución de Errores**
   * Resolver el problema con el envío de mensajes en el chat.
2. **Notificaciones**
   * Agregar notificaciones en tiempo real para nuevos mensajes.
3. **Optimización de Diseño**
   * Mejorar la responsividad del diseño para dispositivos móviles.
4. **Sistema de Contratación**
   * Implementar confirmaciones y estado de las contrataciones.

### ****Backend****

1. **Mejorar el Manejo de Errores**
   * Implementar respuestas de error más claras para facilitar la depuración.
2. **Real-Time Chat**
   * Integrar sockets para mensajes en tiempo real.
3. **Sistema de Contratación**
   * Rutas para manejar el estado de las contrataciones entre usuarios y guías.

## **Resumen General**

**Avance Total del Proyecto:**

* **Frontend:** 85% (pendiente optimización del chat y diseño).
* **Backend:** 90% (pendiente integración de sockets y mejoras en validación).
* **Base de Datos:** 100% (estructura funcional y relaciones establecidas).

Este proyecto está en las etapas finales de desarrollo, con la funcionalidad principal casi completa. Las próximas tareas se centran en mejorar la experiencia del usuario y garantizar un sistema robusto y escalable.