PROYECTO

FIN DE CICLO

TÍTULO DEL PROYECTO

Nombre: Jose Del Pino Castillo

DNI: 53916424P **Curso Académico: 2022 / 2024**

Profesor tutor del Proyecto : Manuel Arroyo

FPA FORMACIÓN PROFESIONAL ANDALUZA

Índice

1. Introducción
2. Análisis de requisitos.
3. Diseño.
4. Codificación.
5. Estructura del manual de instalación y configuración.
6. Conclusiones.
7. Trabajo futuro.
8. Bibliografía y webgrafía.

Introducción

El proyecto consiste en el desarrollo de una plataforma de reserva de hoteles, que permitirá a los usuarios buscar, comparar y reservar habitaciones de hotel de manera fácil y conveniente. La plataforma estará diseñada para ofrecer una experiencia intuitiva y eficiente tanto para los clientes como para los administradores de hoteles.

Los principales objetivos del proyecto son los siguientes:

- **Facilitar la reserva de habitaciones:** La plataforma permitirá a los usuarios buscar disponibilidad de habitaciones en diferentes hoteles, visualizar detalles de las habitaciones y realizar reservas de manera rápida y sencilla.
- **Optimizar la gestión de hoteles:** Para los administradores de hoteles, la plataforma ofrecerá herramientas para gestionar la disponibilidad de habitaciones, precios, promociones y reservas de manera eficiente.
- **Mejorar la experiencia del cliente:** Se priorizará la experiencia del usuario en la plataforma, asegurando un diseño atractivo, navegación intuitiva y procesos de reserva transparentes. Además, se ofrecerá soporte al cliente para resolver consultas y problemas de manera rápida y efectiva.

En resumen, el proyecto tiene como objetivo desarrollar una plataforma de reserva de hoteles completa y eficiente, que brinde una experiencia excepcional tanto para los clientes como para los administradores de hoteles, contribuyendo así a la transformación digital de la industria hotelera.

Análisis de requisitos

El desarrollo y funcionamiento del proyecto de reserva de hoteles requiere el cumplimiento de una serie de requisitos para garantizar su eficacia y satisfacción de los usuarios. A continuación, se detallan los requisitos necesarios:

1. **Registro de usuarios:** La plataforma debe permitir a los usuarios registrarse creando una cuenta con información básica como nombre, correo electrónico y contraseña.

2. **Inicio de sesión:** Los usuarios registrados deben poder iniciar sesión en la plataforma utilizando su correo electrónico y contraseña.
3. **Búsqueda de hoteles:** La plataforma debe ofrecer a los usuarios la posibilidad de buscar hoteles por ubicación.
4. **Visualización de detalles del hotel:** Una vez realizada la búsqueda, los usuarios deben poder ver detalles de los hoteles disponibles, como imágenes, descripción, servicios ofrecidos, precios y disponibilidad de habitaciones.
5. **Reserva de habitaciones:** Los usuarios deben poder seleccionar las fechas de entrada y salida, y realizar la reserva de manera segura.
6. **Gestión de reservas:** Los usuarios registrados deben poder acceder a su historial de reservas, modificar o cancelar reservas existentes, y recibir confirmaciones por correo electrónico.
7. **Diseño responsivo:** La plataforma debe ser compatible con dispositivos móviles y tabletas, garantizando una experiencia de usuario óptima en diferentes tipos de dispositivos y tamaños de pantalla.

Estos son los principales requisitos necesarios para el desarrollo y funcionamiento del proyecto de reserva de hoteles, los cuales deben ser tenidos en cuenta durante todas las etapas del proceso de desarrollo del software.

Diseño

1. Diseño de UI/UX:

El diseño de la interfaz de usuario (UI) y la experiencia de usuario (UX) es crucial para garantizar una interacción fluida, intuitiva y atractiva con la plataforma.

1.1 UI/UX

- **Página de inicio:**
  - Barra de búsqueda para poder buscar los hoteles en la ubicación que queramos
  - Banner principal con buscador de hoteles (ubicación, fechas, número de habitaciones).
  - Sección de promociones y ofertas destacadas.
- **Página de búsqueda:**
  - Resultados de búsqueda mostrados en tarjetas con imagen, nombre del hotel, ubicación, precio por noche y puntuación.
  - Botón para ver más detalles de cada hotel.

- **Página de detalles del hotel:**
    - Imágenes del hotel en un carrusel.
    - Descripción detallada del hotel y sus servicios.
    - Disponibilidad de habitaciones con precios.
    - Ubicación del hotel.
    - Botón para reservar.
- **Página de reserva:**
    - Formulario de reserva con fechas de estadía, número de habitaciones, precio total calculado.
    - Información del cliente pre-llenada si está registrado.
    - Botón para confirmar reserva.
- **Página de perfil de usuario:**
    - Historial de reservas con opciones para ver detalles, modificar o cancelar reservas.

(Propuesta de mejora)

- **Panel de administración:**
    - Dashboard con estadísticas y gráficos sobre reservas, ingresos y usuarios.
    - Gestión de hoteles: agregar, editar, eliminar hoteles y habitaciones.
    - Gestión de usuarios: ver y editar perfiles de usuarios.
    - Gestión de reservas: ver y administrar todas las reservas.

2. Datos

2.1 Modelos de datos:

- **Cliente:**
    - ID
    - Nombre
    - Email
    - Contraseña
    - Edad
    - Nacionalidad
    - Idioma
- **Hotel:**
    - ID
    - Nombre

    - Ubicación
    - Precio por noche
    - Nº Habitaciones Totales
    - Calificación
    - Imágenes
- **Reserva:**
    - ID
    - Fecha de inicio
    - Fecha de fin
    - Precio total
    - Número de habitaciones
    - Cliente (ID de cliente, referenciado)
    - Hotel (ID de hotel, referenciado)

2.2 Relación entre datos:

- Un **Cliente** puede tener múltiples **Reservas**.
- Un **Hotel** puede tener múltiples **Reservas**.
- Una **Reserva** está asociada a un **Cliente** y a un **Hotel**.

3. Almacenamiento

Para el almacenamiento de datos, se utiliza una base de datos relacional **PostgreSQL**. A continuación, se describe la estructura de las tablas en la base de datos.

3.1 Tablas de la base de datos:

- **Clientes:**
    - id (PK, AUTO_INCREMENT)
    - nombre (VARCHAR)
    - email (VARCHAR, UNIQUE)
    - contraseña (VARCHAR)
    - edad (INTEGER)
    - nacionalidad (VARCHAR)
    - idioma (VARCHAR)
- **Hoteles:**
    - id (PK, AUTO_INCREMENT)
    - nombre (VARCHAR)
    - ubicacion (VARCHAR)
    - precio_noche (INTEGER)

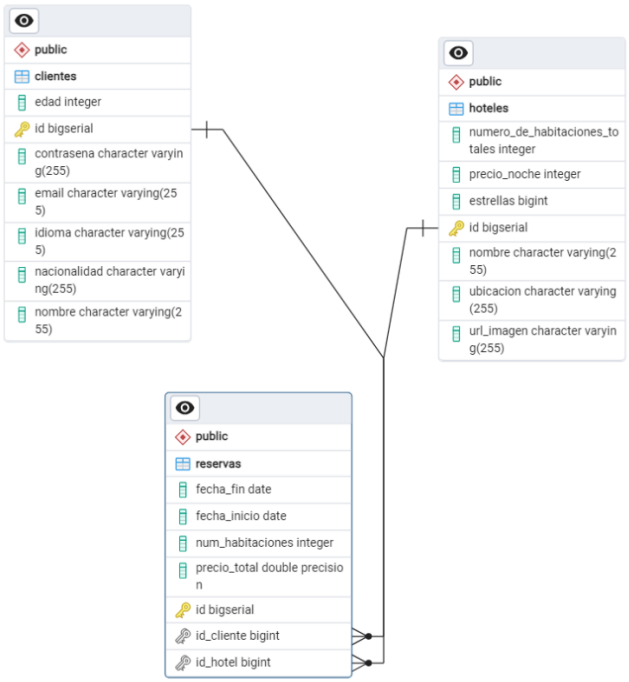
    - nº habitaciones totales (INTEGER)
    - calificacion (INTEGER)
    - imagenes (TEXT)
    - url_imagen (TEXT)
- **Reservas:**
    - id (PK, AUTO_INCREMENT)
    - fecha_inicio (DATE)
    - fecha_fin (DATE)
    - precio_total (INTEGER)
    - num_habitaciones (INTEGER)
    - cliente_id (FK, REFERENCES Clientes(id))
    - hotel_id (FK, REFERENCES Hoteles(id))

3.2 Almacenamiento de imágenes:

Las imágenes de los hoteles se almacenan como URLs en la columna url_imagenes de la tabla Hoteles. Estas URLs apuntan a las ubicaciones donde las imágenes están almacenadas, permitiendo su fácil acceso y gestión.

Estos elementos conforman el diseño integral del proyecto, asegurando una experiencia de usuario satisfactoria y un manejo eficiente y seguro de los datos.

Foto del diagrama UML de la base de datos utilizada para mi proyecto:



Codificación

La elección de lenguajes y herramientas es crucial para el desarrollo efectivo del proyecto. A continuación, se detallan las tecnologías seleccionadas y las razones detrás de su elección.

Lenguajes y Herramientas Utilizadas:

4.1 Backend:

Java con Spring Boot:

- **Razones para elegir Java con Spring Boot:**
  - **Robustez y Escalabilidad:** Java es conocido por su robustez y capacidad de manejar aplicaciones grandes y complejas. Spring Boot, una parte del ecosistema Spring, facilita la creación de aplicaciones independientes, productivas y listas para producción.
  - **Seguridad:** Spring Boot proporciona características integradas de seguridad y es compatible con una amplia gama de herramientas de seguridad.
  - **Ecosistema y Soporte:** La comunidad de Spring Boot es amplia, proporcionando abundante documentación y recursos. Además, Spring Boot se integra bien con otros componentes del ecosistema Spring.

4.2 Frontend:

Vue.js:

- **Razones para elegir Vue.js:**
  - **Componentización:** Vue.js permite dividir la interfaz de usuario en componentes reutilizables, facilitando la gestión y escalabilidad del código.
  - **Simplicidad y Facilidad de Uso:** Vue.js es conocido por su curva de aprendizaje suave y su sintaxis intuitiva, lo que lo hace accesible tanto para desarrolladores principiantes como experimentados.
  - **Performance:** Vue.js es ligero y ofrece un rendimiento rápido, lo que es crucial para una experiencia de usuario fluida.
  - **Popularidad y Comunidad:** Vue.js tiene una comunidad creciente y muchas bibliotecas y herramientas de terceros, lo que facilita encontrar soluciones y apoyo.

4.3 Base de Datos:

PostgreSQL:

- **Razones para elegir PostgreSQL:**
  - **Relacional y Open Source:** PostgreSQL es una base de datos relacional open source, conocida por su robustez y conformidad con el estándar SQL.
  - **Funciones Avanzadas:** Ofrece características avanzadas como transacciones ACID, replicación y concurrencia avanzada.
  - **Escalabilidad y Rendimiento:** PostgreSQL maneja eficientemente grandes volúmenes de datos y puede escalar según las necesidades del proyecto.
  - **Almacenamiento de Imágenes:** Las URL de las imágenes de los hoteles se almacenan en la misma tabla de hoteles, facilitando la gestión y acceso a las imágenes desde el frontend.

4.4 Herramientas de Desarrollo:

Postman:

- **Razones para elegir Postman:**
  - **Testing de APIs:** Postman es una herramienta poderosa para probar y documentar APIs, facilitando el desarrollo y debugging del backend.
  - **Colaboración:** Postman permite la colaboración entre desarrolladores, asegurando que todos tengan acceso a las mismas colecciones de pruebas y documentación de la API.

IntelliJ IDEA:

- **Razones para elegir IntelliJ IDEA:**
  - **Productividad:** IntelliJ IDEA ofrece una amplia gama de herramientas integradas para el desarrollo de aplicaciones Java, mejorando la productividad del desarrollador.
  - **Integraciones:** Se integra perfectamente con sistemas de control de versiones como Git, bases de datos y herramientas de construcción como Maven y Gradle.

GitHub:

- **Razones para elegir GitHub:**
  - **Control de Versiones:** GitHub ofrece un sistema de control de versiones basado en Git, que es esencial para el seguimiento y la gestión del código fuente.

- **Colaboración y Gestión de Proyectos:** GitHub facilita la colaboración entre desarrolladores y proporciona herramientas para la gestión de proyectos, incluyendo issues y pull requests.

Resumen

La combinación de **Java con Spring Boot** para el backend, **Vue.js** para el frontend y **PostgreSQL** para la base de datos proporciona una base sólida para desarrollar una aplicación web robusta y escalable. Las herramientas adicionales como **Postman**, **IntelliJ IDEA**, y **GitHub** complementan el desarrollo, asegurando una gestión eficiente del código y una colaboración efectiva entre los desarrolladores. Estas elecciones están motivadas por la robustez, escalabilidad, seguridad y el soporte comunitario que ofrecen estas tecnologías.

Estructura del manual de instalación y configuración

Un manual de instalación y configuración bien estructurado es crucial para asegurar que los usuarios puedan poner en marcha el proyecto de manera eficiente. A continuación se detalla la estructura recomendada para el manual, asegurando que incluya todas las partes necesarias y siga un estilo adecuado.

Manual de Instalación y Configuración

1. Introducción

- **Objetivo del Manual:**
  - Proveer una guía detallada para la instalación y configuración del sistema de reserva de hoteles.
- **Requisitos Previos:**
  - Lista de software y hardware necesarios.
  - Conocimientos previos requeridos.

2. Requisitos del Sistema

- **Hardware:**
  - Procesador, memoria, espacio en disco, etc.
- **Software:**
  - Sistema operativo soportado.
  - Versiones específicas de Java, Node.js, PostgreSQL, etc.

3. Descarga del Proyecto

- **Repositorio:**
  - URL del repositorio GitHub.

- **Clonación:**
    - Instrucciones para clonar el repositorio.

4. Instalación del Backend

- **1. Prerequisitos:**
    - Instalación de JDK.
    - Instalación de Maven.
- **2. Configuración de Spring Boot:**
    - Variables de entorno.
    - Archivo application.properties o application.yml.
- **3. Base de Datos:**
    - Instalación de PostgreSQL.
    - Creación de la base de datos.
    - Configuración de las credenciales de la base de datos en Spring Boot.

Comandos y Configuraciones:

Copiar código

Ejemplo de comandos

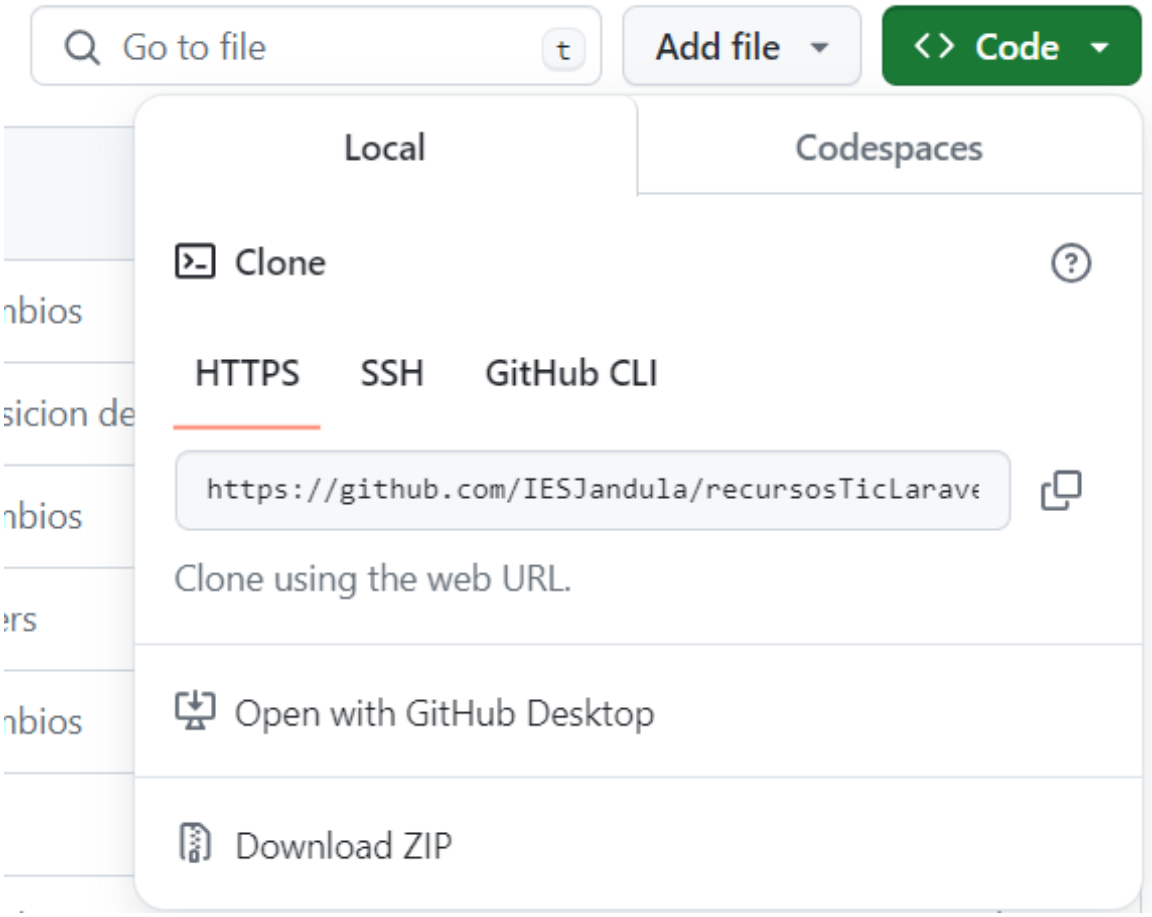
git clone https://github.com/IESJandula/destinoDeluxe.git

cd proyecto-reservas

mvn clean install

Ejemplo:

Para poder realizar el git clone debemos ir en github al proyecto que queramos clonar, pulsamos en clone y copiamos la url, como en la siguiente imagen:



Una vez copiada la url ya podemos hacer el clone de este proyecto.

Ejecución:

Una vez tengamos clonado el repositorio abrimos la carpeta back que es donde esta todo el back, y abrimos el proyecto con IntelliJ.

Una vez lo tengamos abierto lo único que debemos de hacer para arrancar el programa es darle a ejecutar arriba a la derecha.

5. Instalación del Frontend

- **1. Prerequisitos:**
  - Instalación de Node.js y npm.
- **2. Instalación de Dependencias:**
  - Comandos para instalar las dependencias del proyecto.
- **3. Configuración:**
  - Variables de entorno y archivos de configuración.
- **4. Construcción y Ejecución:**
  - Comandos para compilar y ejecutar el frontend.

Comandos y Configuraciones:

Copiar código

Ejemplo de comandos

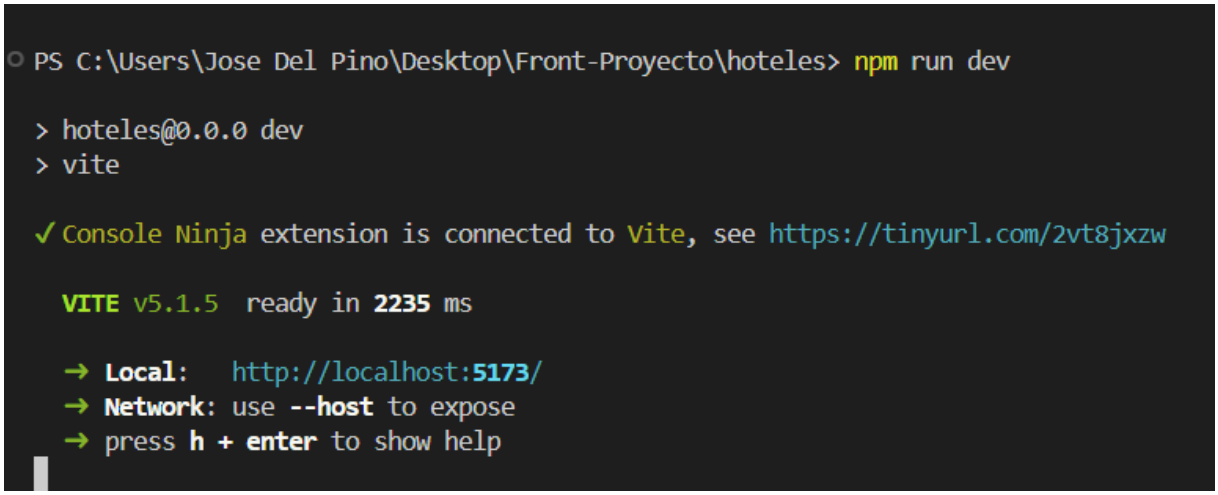
git clone https://github.com/IESJandula/destinoDeluxe.git

cd frontend

npm install

npm run dev

Ejemplo:



http://localhost::5173 es la url para poder visualizar el proyecto de manera local, no sin antes arrancar el back para que funcionen todas las funcionalidades del proyecto de manera correcta.

6. Configuración de Postman

- **1. Instalación de Postman:**
  - Instrucciones para descargar e instalar Postman.
- **2. Importación de Colecciones:**
  - Instrucciones para importar las colecciones de API.
- **3. Configuración de Entornos:**
  - Cómo configurar entornos en Postman para pruebas.

7. Ejecución del Proyecto

- **1. Iniciar Backend:**
  - Comandos para iniciar el servidor Spring Boot.
- **2. Iniciar Frontend:**
  - Comandos para iniciar el servidor Vue.js.
- **3. Verificación:**
  - Verificar que los servicios están corriendo correctamente.

Esta estructura asegura que el manual de instalación y configuración cubra todos los aspectos necesarios para que cualquier usuario pueda instalar, configurar y ejecutar el proyecto de manera efectiva.

Conclusiones

El desarrollo del sistema de reserva de hoteles ha sido un proyecto integral que ha abarcado diversas fases desde la concepción inicial hasta la implementación final. A continuación, se presentan las conclusiones sobre el desarrollo y desempeño del proyecto.

Desarrollo

1. **Planificación y Diseño:**
    - **Requisitos:** La fase de análisis de requisitos fue fundamental para definir las necesidades del sistema, asegurando que todas las funcionalidades esenciales fueran cubiertas.
    - **Diseño UI/UX:** La elección de Vue.js para el frontend permitió crear una interfaz de usuario moderna, dinámica e intuitiva, mejorando la experiencia del usuario final.
2. **Tecnologías Utilizadas:**
    - **Backend:** La implementación con Spring Boot proporcionó una estructura robusta y escalable, facilitando el manejo de las operaciones del servidor y la integración con la base de datos PostgreSQL.
    - **Frontend:** Vue.js demostró ser una excelente opción para desarrollar una interfaz de usuario interactiva, permitiendo una fácil integración con el backend.
    - **Base de Datos:** PostgreSQL utilizé como sistema de gestión de bases de datos, proporcionando un almacenamiento confiable y eficiente para las reservas, clientes y hoteles.
3. **Desafíos y Soluciones:**
    - **Integración:** La integración entre el frontend y el backend requirió atención detallada, especialmente en la configuración de las API y el

manejo de datos. Las pruebas exhaustivas con Postman aseguraron que todas las integraciones funcionaran correctamente.

- **Validación y Seguridad:** Se implementaron validaciones en el backend para garantizar la integridad de los datos, así como medidas de seguridad para proteger la información sensible de los usuarios.

Desempeño

1. **Eficiencia:**
   - **Rendimiento:** El sistema demostró un buen rendimiento en pruebas de carga, manejando múltiples solicitudes simultáneas sin problemas significativos.
   - **Optimización:** Las consultas a la base de datos y el manejo de datos en el servidor fueron optimizados para reducir tiempos de respuesta y mejorar la eficiencia general del sistema.
2. **Usabilidad:**
   - **Interfaz de Usuario:** Los usuarios encontraron la interfaz de usuario intuitiva y fácil de navegar, facilitando la gestión de reservas y la consulta de disponibilidad de hoteles.
   - **Accesibilidad:** Se tomaron en cuenta aspectos de accesibilidad para asegurar que el sistema pudiera ser utilizado por una amplia gama de usuarios, incluyendo aquellos con discapacidades.
3. **Escalabilidad:**
   - **Estructura Modular:** La arquitectura modular del proyecto permite una fácil escalabilidad, permitiendo agregar nuevas funcionalidades y mejorar las existentes sin afectar el rendimiento del sistema.
   - **Base de Datos:** PostgreSQL se adapta bien a un crecimiento futuro del sistema, permitiendo la gestión eficiente de un mayor volumen de datos.

Futuras Mejoras

1. **Nuevas Funcionalidades:**
   - **Integración con Servicios de Pago:** Implementar integración con servicios de pago en línea para completar el ciclo de reserva.

- **Administración:** Toda la parte de Administración como por ejemplo añadir hoteles, eliminar, etc..
  Que algunas cosas están implementadas en el backend pero no en el fronted.
- **Soporte Multilingüe:** Añadir soporte para múltiples idiomas para atender a una audiencia global.

2. **Mejoras en Seguridad:**
   - **Autenticación y Autorización:** Mejorar los mecanismos de autenticación y autorización para aumentar la seguridad del sistema con Spring Security
   - **Encriptación de Datos:** Implementar encriptación de datos tanto en tránsito como en reposo para proteger la información sensible.
3. **Optimización de Rendimiento:**
   - **Mejoras en Caché:** Implementar técnicas de caché para reducir la carga en la base de datos y mejorar los tiempos de respuesta.
   - **Optimización de Consultas:** Continuar optimizando las consultas SQL para asegurar un rendimiento óptimo incluso con un mayor volumen de datos.

Conclusión General

El sistema de reserva de hoteles desarrollado cumple con los requisitos y objetivos establecidos al inicio del proyecto. La elección de tecnologías como Spring Boot, Vue.js y PostgreSQL ha sido acertada, proporcionando un sistema robusto, eficiente y escalable. El proyecto está bien posicionado para futuras mejoras y expansiones, garantizando su relevancia y efectividad a largo plazo.

Durante el desarrollo de este proyecto me he encontrado con algunos problemas con el back, ya que escogí SpringBoot para aprenderlo por mi cuenta porque es una tecnología que se utiliza mucho en el mundo laboral, por ese lado ha sido algo más complicado, pero me ha gustado mucho realizar este proyecto por las cosas que he aprendido tanto de SpringBoot como de Vue.js y a ver resuelto los problemas que me he ido encontrando.

Trabajo Futuro

El sistema de reserva de hoteles desarrollado es una base sólida que puede ser expandida y mejorada para cubrir más necesidades y ofrecer una mejor experiencia a los usuarios. A continuación, se presentan algunas propuestas para futuras modificaciones, ampliaciones y extensiones del proyecto:

1. Integración con Servicios de Pago

Descripción: Implementar la integración con servicios de pago en línea como PayPal, Stripe, o sistemas de pago locales para permitir a los usuarios realizar pagos directamente desde la plataforma.

Beneficios:

- Facilita el proceso de reserva, ofreciendo una experiencia de usuario más completa.
- Aumenta la confianza de los usuarios al permitir pagos seguros y confiables.
- Genera nuevas oportunidades de ingresos a través de tarifas de transacción.

2. Soporte Multilingüe

Descripción: Añadir soporte para múltiples idiomas en la interfaz de usuario y en los mensajes de error y validación.

Beneficios:

- Atrae a una audiencia global, incrementando la base de usuarios.
- Mejora la accesibilidad y la experiencia del usuario para hablantes no nativos.

3. Sistema de Recomendaciones

Descripción: Desarrollar un sistema de recomendaciones basado en las preferencias y el historial de reservas de los usuarios.

Beneficios:

- Personaliza la experiencia del usuario, sugiriendo hoteles y ofertas que se adapten a sus preferencias.
- Aumenta la tasa de conversión al ofrecer recomendaciones relevantes.

4. Mejoras en Seguridad

Descripción: Implementar medidas avanzadas de seguridad, como la autenticación de dos factores (2FA), encriptación de datos sensibles y auditorías de seguridad regulares.

Beneficios:

- Protege la información sensible de los usuarios, aumentando la confianza en la plataforma.

- Cumple con las normativas de seguridad y privacidad, reduciendo el riesgo de brechas de datos.

5. Optimización de Rendimiento

Descripción: Continuar optimizando el rendimiento del sistema mediante la implementación de técnicas de caché, mejoras en las consultas SQL y la adopción de tecnologías de servidores más eficientes.

Beneficios:

- Mejora los tiempos de respuesta y la experiencia del usuario, especialmente durante picos de alta demanda.
- Reduce la carga en la base de datos y los servidores, permitiendo un mejor escalado del sistema.

6. Aplicación Móvil

Descripción: Desarrollar una aplicación móvil nativa para iOS y Android que permita a los usuarios acceder a todas las funcionalidades de la plataforma desde sus dispositivos móviles.

Beneficios:

- Aumenta la accesibilidad y la conveniencia para los usuarios, permitiendo reservas sobre la marcha.
- Extiende la presencia del sistema a una audiencia más amplia, que prefiere el uso de aplicaciones móviles.

7. Análisis de Datos y Reportes

Descripción: Integrar herramientas de análisis de datos que proporcionen informes detallados sobre el uso del sistema, las tendencias de reservas y otros indicadores clave de rendimiento (KPIs).

Beneficios:

- Proporciona información valiosa para la toma de decisiones estratégicas y la mejora continua del sistema.
- Ayuda a identificar patrones y oportunidades de negocio, optimizando el marketing y la gestión de recursos.

8. Ampliación de Funcionalidades de Gestión de Hoteles

Descripción: Añadir funcionalidades avanzadas para la gestión de hoteles, como la administración de disponibilidad de habitaciones, la gestión de tarifas dinámicas y la

integración con sistemas de gestión de propiedades (PMS). Además, incluir funcionalidades para la administración de hoteles como crear, editar y eliminar hoteles desde una interfaz administrativa.

Beneficios:

- Ofrece a los hoteleros herramientas más poderosas para gestionar sus propiedades de manera eficiente.
- Mejora la competitividad de la plataforma al ofrecer un conjunto completo de herramientas para la gestión hotelera.
- Permite una administración más flexible y controlada de los hoteles, facilitando la actualización y mantenimiento de la información.

Conclusión

Estas propuestas para el trabajo futuro del sistema de reserva de hoteles están orientadas a mejorar la funcionalidad, la seguridad, y la experiencia del usuario, así como a expandir el alcance y la capacidad de la plataforma. La implementación de estas mejoras y extensiones permitirá que el sistema siga siendo relevante y competitivo en el mercado, ofreciendo un servicio de alta calidad a usuarios y hoteleros por igual.

Bibliografía y webgrafía

Para el desarrollo de este proyecto de reserva de hoteles, se han consultado diversas fuentes de información y recursos técnicos. A continuación, se presenta una lista de las referencias bibliográficas y webgráficas utilizadas:

Webgrafía

1. **Spring Documentation**. *Spring Framework Reference Documentation*. Disponible en: https://docs.spring.io/spring-framework/docs/current/reference/html/
    - La documentación oficial del framework Spring, que proporciona guías, ejemplos y referencias detalladas sobre todas sus características y módulos.
2. **PostgreSQL Documentation**. *The PostgreSQL Global Development Group*. Disponible en: https://www.postgresql.org/docs/
    - La documentación oficial de PostgreSQL, cubriendo desde la instalación y configuración hasta características avanzadas y mejores prácticas.

3. **Vue.js Documentation**. *Vue.js - The Progressive JavaScript Framework*. Disponible en: https://vuejs.org/v2/guide/
    - La documentación oficial de Vue.js, que ofrece guías y ejemplos sobre cómo utilizar este framework para desarrollar interfaces de usuario interactivas y dinámicas.
4. **Baeldung**. *Spring Tutorials*. Disponible en: https://www.baeldung.com/
    - Un recurso en línea con numerosos tutoriales y artículos técnicos sobre el uso de Spring y otras tecnologías relacionadas con el desarrollo de aplicaciones Java.
5. **W3Schools**. *W3Schools Online Web Tutorials*. Disponible en: https://www.w3schools.com/
    - Una plataforma educativa que ofrece tutoriales y referencias sobre lenguajes de programación web y desarrollo front-end.

Otros Recursos

- **GitHub**. *Repositores de Código Fuente*. Disponible en: https://github.com/
    - Utilizado para gestionar y versionar el código fuente del proyecto, facilitando la colaboración y el control de versiones.

Estas fuentes han sido fundamentales para guiar el desarrollo del proyecto, ofreciendo la información técnica necesaria y mejores prácticas en cada una de las áreas abordadas.