Proyecto Sprint 2 Módulo 2 Curso Desarrollo Web Frontend

Se requiere desarrollo de una aplicación web de compra de boletos de pasajes aéreos utilizando React JS, React Router DOM v6, Vite como herramienta de compilación de código, sistema de control de versiones Git, GitHub como plataforma de alojamiento de código, administración de desarrollo y colaboración en equipos, así como opciones de implementación en la nube y despliegue como Render y Github pages, entre otras herramientas y tecnologías relevantes para la construcción de la aplicación.

La aplicación debe permitir a los usuarios buscar y seleccionar vuelos, completar el flujo de la compra de ticketes, ver el resumen de la compra y generar los boletos. Debe implementar funciones como autenticación de usuarios, seguridad de rutas y posiblemente integración con API externas como Open Skay Network API o FLIGHTSTATS. El enfoque principal del proyecto es desarrollar una interfaz utilizando ReactJS y SASS o Styled Components para estilos y diseño responsive y receptivo (Responsive Design y Adaptive Design) que cumpla con el diseño proporcionado a continuación:

https://www.figma.com/file/ujckGNXmGneONpKdignWvg/E-flight?type=design&node-id=0%3A1&mode=dev

El objetivo final del proyecto es adquirir experiencia en el desarrollo de aplicaciones frontend utilizando React JS junto con otras tecnologías y herramientas de interés, así como la implementación de despliegue y buenas prácticas de desarrollo y calidad del código.

A continuación, se detallan los requerimientos del proyecto:

Espacio de trabajo:

- Configurar un nuevo proyecto de React con Vite.
- Organizar la estructura de carpetas y archivos del proyecto de acuerdo con las mejores prácticas y el enfoque modular.

Diseño UI/UX:

- Cumplir con el diseño proporcionado por el área de diseño.
- Implementar diseño receptivo y responsivo (Responsive Design y Adaptive Design) .
- Realizar propuesta diseño responsivo para dispositivos como tablets y mobile.
- Escoger SASS o Styled Component para estilizar las vistas y de manera opcional, implementar cualquier framework CSS o librerías de estilos de su preferencia.

React JS:

- Utilizar Hooks de React, como useState, useEffect, useContext y useReducer para gestionar el estado y los efectos secundarios de la aplicación.
- Implementar Hooks personalizados y hooks de rendimiento como useCallback y useMemo.
- Implementar componentes funcionales de React para construir la interfaz de usuario y la lógica de la aplicación.

Sistema de control de versiones:

- Utilizar Git y GitHub para gestionar el control de versiones del proyecto.
- Implementar Git Flow para crear ramas para organizar el desarrollo y facilitar la colaboración con los compañeros de equipo.
- Realizar commits frecuentes para asegurar el código. Se recomienda realizar commit por cada feature completamente funcional y sin errores o bugs.

Autenticación y autorización de usuarios:

- Implementar sistema de autenticación para permitir el registro e inicio de sesión de usuarios.
- Proponer el diseño de la página de inicio de sesión y creación de cuentas de usuarios.
- Proteger las rutas y funcionalidades para garantizar que solo los usuarios autenticados puedan seleccionar un vuelo e iniciar y completar la compra de pasajes aereos. A usuarios no autenticados se les debe permitir solo la búsqueda de vuelos.

React Router DOM v6:

- Implementar la librería para permitir el enrutamiento de la apliación.
- Implementar rutas anidadas, componente Layout y hooks (useParams, useNavigate, useLocation, entre otros).

Implementación JSON server:

- Simular API REST con los endpoints necesarios que proporcionen información de los vuelos, programación de vuelos, listado de paises, aeropuertos, usuarios, reservas entre otros datos que considere de relevancia para la aplicación.
- Proponer una estructura de datos óptima en el JSON server que facililte las operaciones de lectura, filtrado, ordenamiento y creación o envío de datos a los diferentes enpoints.
- En caso de que se decida implementar una API externa con información veráz de vuelos, calendario de vuelos y aereopuertos, la API REST cómo mínimo debe contener la información de los usuarios activos y reservas realizadas.
- Para generar datos de prueba en formato JSON, es posible utilizar JSON Data AI.

Integración con API externa (opcional):

- Se recomienda utilizar APIs como Open Skay Network API o FLIGHTSTATS para obtener la información vuelos, programación de vuelos, listados de paises, aeropuertos y aerolíneas disponibles.

Búsqueda de vuelos:

- Realizar filtrado y ordenamiento de vuelos por los siguientes criterios:
  - Tipo de viaje: ida y regreso (Round trip) o solo ida (one way)
  - Cantidad de pasajeros: Adultos (mayor a 12 años), niños (2 a 12 años) y bebés (debajo de 2 años)
  - Clase de la cabina (Cabain class): Economy, premium economy, Bussiness y first class
  - Aeropuerto de origen y destino
  - Fecha de vuelo
  - Ordenar los vuelos por precio: de manera ascendente y descendente.
  - Número de escalas: directo, una escala, etc.
  - Tiempo de vuelo

Mostrar resultado de la búsqueda y selección de asientos:

- Realizar la operación de filtrado desde la operación disponible en JSON server o directamente desde la lógica del componente utilizando las funciones o métodos de arrays como .filter() o .sort()
- Mostrar los resultados y categorizarlos por fechas de salida y organizarlos de manera ascendente por hora.
- Al seleccionar el vuelo, debe mostrar un drawer con la información de detalle del vuelo y beneficios que indica el diseño.
- **No** es requerido incluir en la información de detalle del vuelo las opciones **Refund** ni **Reschedule**.

Flujo de compra de pasajes y generación de Tiquetes:

- Se debe solicitar la información personal de cada pasajero y mostrar el resumen del o los vuelos seleccionados.
- Seleccionar el método de págo (tarjetas o paypal), completar la información del pago y confirmar la compra.
- Mostrar compra exitosa de pasajes y generar los tiquetes según la cantidad de usuarios y vuelos reservados.

Despliegue de aplicaciones en la nube:

- Desplegar la aplicación en un servicio de alojamiento en la nube como GitHub pages, vercel, Netlify, entre otros.
- Desplegar API REST simulada en servicios de Hosting como Render, fly.oi, fl0.com, entre otros.

En la aplicación se debe evidenciar el uso e implementación de las siguientes herramientas o librerías:

1. React JS
2. Hooks de estado, de efecto, de rendimiento y personalizados
3. React Router DOM v6 y enrutamiento dinámico
4. Vite
5. Git Flow
6. Axios
7. SASS o Styled Component
8. Sweet Alert o react Toastify
9. JSON server
10. Despliegue en la nueve del JSON server y de la aplicación