Prueba Técnica para Desarrollador Front-end.

Crear una aplicación web para el agendado de boletos de una aerolínea. En esta prueba técnica se evaluará la creatividad y diseño presentado, la organización de código, la lógica de creación de componentes y funcionalidad. El diseño y la experiencia de usuario deben ser preferentemente de tu autoría, evita utilizar frameworks de diseño (Bootstrap, Material, Bulma) lo importante no es que “se vea bonito” si no que demuestres un manejo de CSS y puedas crear layouts responsivas por tu cuenta.
El plazo de entrega es de 1 semana natural, si necesitas un par de días más para completar la prueba (entendemos que puedes tener otras responsabilidades) lo puedes solicitar sin problema. ¡Buena suerte!

Mi aerolínea.

Se necesita una aplicación para la reserva de boletos de vuelo. Se debe consultar de un API el listado de ciudades destino, los destinos ofrecen varios vuelos por día a distinta hora y el precio de cada vuelo es distinto.

Vista de Inicio

Se debe tener una vista donde el usuario elige el origen y destino de su viaje, el horario de su vuelo, el número de personas que van a viajar.
La lista de ciudades a desplegar **tiene que ser consultada de algún API,** (puedes usar un mock server, hacer un API en Django, Firebase, Node-Express o lo que más te facilite la implementación), la petición puede ser consultada por fetch, axios o lo que te facilite la consulta.
El catálogo de ciudades se debe almacenar en el store de Redux y ser consumido por tu aplicación.
Al seleccionar el destino se le debe presentar las opciones de horarios de vuelo de cada ciudad, esto puede ser por medio de una lista, tarjetas o como prefieras (uno de los puntos a evaluar es creatividad y diseño).
Una vez elegida la ciudad, el horario y el número de personas **se deberá almacenar la reserva** en un “carrito de compras” y se regresará a la página inicial donde el usuario podrá elegir reservar otros vuelos.

Mis reservas

Se requiere de una vista de *mis reservas* (el carrito de compras) donde se muestran todas las reservas que el usuario haya elegido, se debe desplegar la ciudad origen, destino, el horario, número de viajeros y precio de cada vuelo. Al final debe haber un total del costo de todas sus reservas. Desde ahí el usuario debe poder cancelar (borrar) alguna de las reservas que ya no deseé o limpiar su carrito.

Reserva de vuelos

Al final se le deben pedir sus datos al usuario:

- Nombres
- Apellidos
- Dirección
- Correo electrónico

Esto puede ser mediante una nueva vista, modal, tarjeta, etc. Al llenar estos se puede proceder a “Reservar” donde la selección de vuelos y demás datos se le presentarán al usuario con un “Gracias por tu reserva!” y se eliminarán del listado del carrito de reservas. No es necesario hacer algo con estos datos (en un aplicativo real se mandaría a un API la confirmación, más en esta prueba omitiremos este paso 😁)
El botón de “Reservar” no deberá estar activo si no se han llenado todos los campos.

Entregables

La prueba técnica deberá estar en un repositorio de tu preferencia (github, gitlab, bitbucket, etc.) y se deberá proporcionar:

1. La URL al repositorio para poder descargar el Código fuente de tu app.
2. Las instrucciones para poder ejecutar el proyecto en la computadora del evaluador.
3. Estar desplegada tu aplicación en el servicio de tu preferencia (heroku, firebase, aws, github pages) con el fin de compartir la aplicación funcional con el equipo de revisión.

Puntos a Evaluar en la Prueba:

1. Entregó un link con su aplicativo hosted en internet.
2. Proporcionó una liga con su código en Git.
3. Documentó cómo ejecutar su proyecto en local (Especialmente importante si utiliza herramientas o librerías extra que requieren ser instaladas por parte del usuario), esto puede ser directamente en el *readme* de tu proyecto.
4. Cumple con todos los requisitos de la prueba técnica.
    a. Casos de uso (vistas con funcionalidad completa).
    b. Consumo de un API.
    c. Manejo del store de Redux.
5. Diseño: es entendible el flujo, los elementos se ven claramente y es **responsivo.**
6. Abstracción: la separación de componentes es lógica, e hizo una buena separación de responsabilidades.
7. Tests: Cuenta con pruebas en al menos un componente.

¡Mucha suerte!