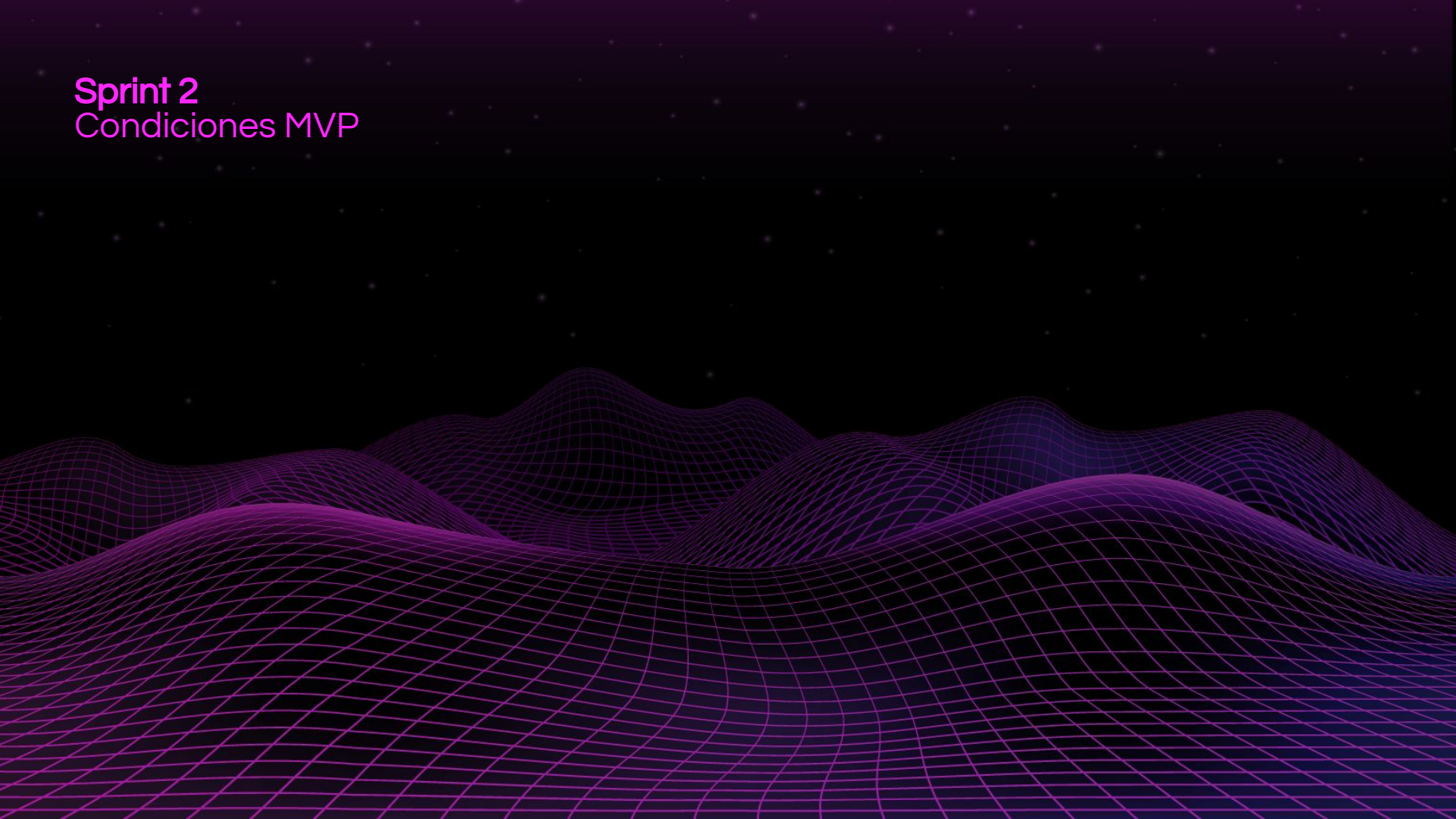
BIENVENIDOS

MIND
HUB.

Sprint 2
Condiciones MVP



ÍNDICE

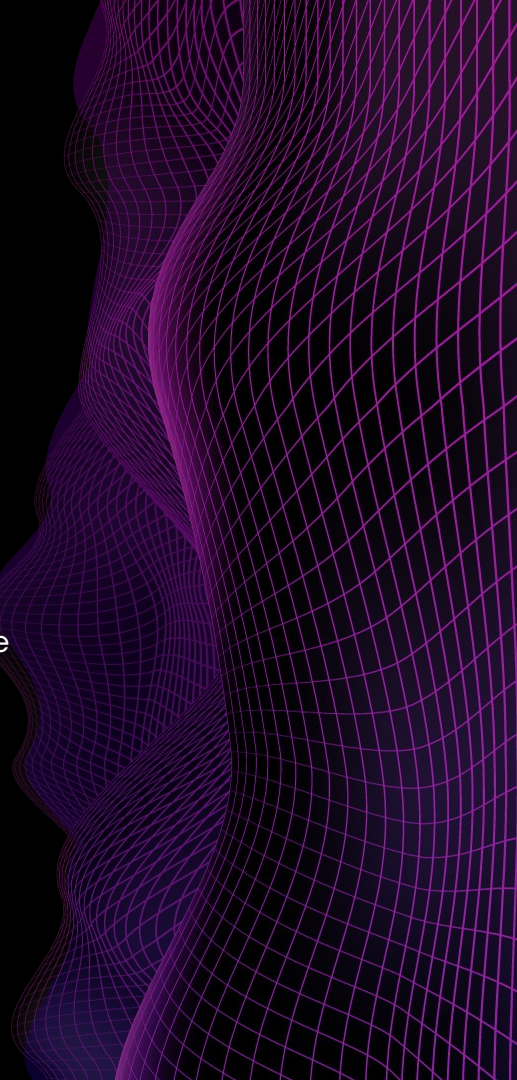
1 Pautas generales

2 Requisitos mínimos

Pautas generales

> Pautas generales

- El objetivo de este sprint es desarrollar el componente (página) Cities.
- Se sigue utilizando el mismo repositorio que para el sprint 1 (Front).
- Crear un nuevo repositorio con el nombre mytinerary-back-nombreApellido (Back).
- El repositorio debe ser pusheado antes del horario de entrega, los mentores realizarán el último pull y el estado evaluable del repositorio será el de ese momento. Se sugiere realizar pusheos diarios para incorporar Git a la rutina de trabajo e ir identificando problemas y errores en el uso de la herramienta.
- Durante el día se procederá a la corrección de cada trabajo. Para obtener un MVP se deben cumplimentar todas las consignas aquí especificadas. Incumplir algunos de los criterios evaluatorios implica un NO-MVP.
- La interfaz de toda la aplicación de **MyTinerary** debe estar en idioma **inglés** (verificar el correcto uso de palabras, expresiones, etc; evitar traducciones literales no verificadas). El código (nombre de funciones, variables, comentarios, etc) puede estar en inglés o en español (**evitar la mezcla**).
- El no respeto a los requisitos del MVP por falta de lectura, falta de cross-check (o hecho de manera deficiente), etc. los hará incurrir en un NO-MVP.



Requisitos mínimos

Requisitos mínimos

A nivel frontend:

- Desarrollar el componente ***Cities***. El mismo deberá fetchear de forma dinámica a la API, obtener todas las ciudades (nombre + foto) y a través de un input posibilitar al usuario a que filtre aquellas ciudades que le interesen.
- Las ciudades deben ser 15.
- El filtro debe funcionar de la siguiente manera: si el usuario ingresa "**bue**" deben aparecer las ciudades que comienzan con "**bue**" (*indistintamente si pone mayúsculas o minúsculas o una combinación*). Si borra la "**e**", deben aparecer ciudades que comienzan con "**bu**". No debe aparecer "**Paris**" por ejemplo si el usuario pone "**ri**", solo ciudades que comienzan con "**ri**". Incluso si la persona pone espacios en blanco al comienzo y/o al final de lo que busca, debe reconocerlo. Si no hubiera ciudades con las letras buscadas, la app lo debe informar de forma agradable y con buena UX.
- Al hacer click en una ciudad me debe dirigir a un componente reutilizable (obviamente no un componente por ciudad) que me indique en qué ciudad ingresé (a través de texto y foto) y debajo tenga la leyenda "*Under construction*" (que es donde irán los itinerarios y las actividades).
- Debe haber un botón para retornar a Cities (además del link del menú del header y footer).

Requisitos mínimos

A nivel backend:

Se debe crear:

- El **servidor**
- La **base de datos**
  - La **colección cities**
- El modelo **City** que incluya como mínimo:
  - **Nombre** de ciudad
  - **Foto** (link web o ruta/nombre archivo local)
  - **País** al que pertenece.
  - Si quieren agregar más propiedades para dar otras funcionalidades, adelante.
- El **router**
- Los **controladores** necesarios para:
  - **Cargar** una ciudad
  - **Obtener** todas las ciudades
  - **Obtener** una ciudad específica por su ID
  - **Modificar** una ciudad
  - **Borrar** una ciudad

(*Aunque los controladores de modificar, cargar y borrar no tengan en el frontend como ser utilizados, deben estar disponibles para usarse desde, por ejemplo, Insomnia/Postman*).

¡MUCHAS GRACIAS!