Ejercicio práctico

Haciendo uso de la Api de Dragon Ball Z https://web.dragonball-api.com/ se requiere desarrollar un sistema que consuma el api de dragon ball z, desarrollando un backend que orqueste el llamado al api y gestione las cuentas de usuario, además de un frontend que permite al usuario interactuar con el sistema.

Requerimientos Funcionales

Listado de personajes

- Listar los personajes de la serie en forma de galería
- Mostrar el detalle de un personaje seleccionado

Cuenta de usuario

- Registrar usuario
- Ingreso de usuario (Login: validación de email y password)

Lista de favoritos

- Agregar personaje a favoritos
- Listar personajes favoritos
- Quitar personaje de favoritos

Modelo de datos

Galería personajes

- Nombre personaje
- Genero
- Ki
- Imagen

Detalle del personaje

- Nombre
- Genero
- Raza
- Planeta de origen
- Ki
- MaxKi
- Imagen
- Descripción
- Afiliación

Usuario

- Nombre de usuario
- Email
- Password

El sistema en principio listara el listado de personajes de Dragon Ball, para eso se necesita orquestar el consumo del endpoint de la API **https://dragonball-api.com/api/characters** y transforma la trama original en una propia de la aplicación a criterio del desarrollador.

Al momento que el usuario interactúe con una carta del personaje (clic sobre la carta), se redireccionará a otra ruta donde se mostrará el detalle del personaje, de igual forma se necesita orquestar el consumo del endpoint de la API **https://dragonball-api.com/api/characters/{id_personaje}**

El sistema permitirá el registro de usuario de manera que cada usuario pueda tener una cuenta en la que tenga una lista de personajes favoritos. A esta lista el usuario podrá agregar o quitar personajes y por su puesto debe poder listarlos (mostrar el listado de personajes favoritos). Tanto el registro del usuario, como el manejo de su lista de favoritos debe ser persistido de alguna manera (criterio del desarrollador)

Para agregar un personaje a la lista de favoritos solo en caso de que el usuario interactúe con el sistema dentro de una sesión se debería habilitar un botón u otra manera para que el personaje pueda agregarse a la lista (queda a criterio del desarrollador)

Finalmente, el sistema debe restringir la cuenta de cada usuario mediante un componente de ingreso al sistema (un Login).

Funcionalidades que no requieren registro de usuario

- Ver listado de personajes
- Ver detalle de personajes

Funcionalidades que requieren registro de usuario

- Administración de lista de favoritos
- Validación de credenciales para acceso a cuenta de usuario

Entregables del proyecto

- **Diagrama de arquitectura de componentes**
- Artefacto(s) Backend
- Artefacto(s) Frotend
- Guía para levantamiento del sistema localmente

Adicionalmente es requerido que cada artefacto sea dockerizado es decir cuente con us archivo dockerfile el cual cree la imagen del artefacto, es deseable que se pueda dockerizar la solución completa incluyendo la persistencia median docker compose

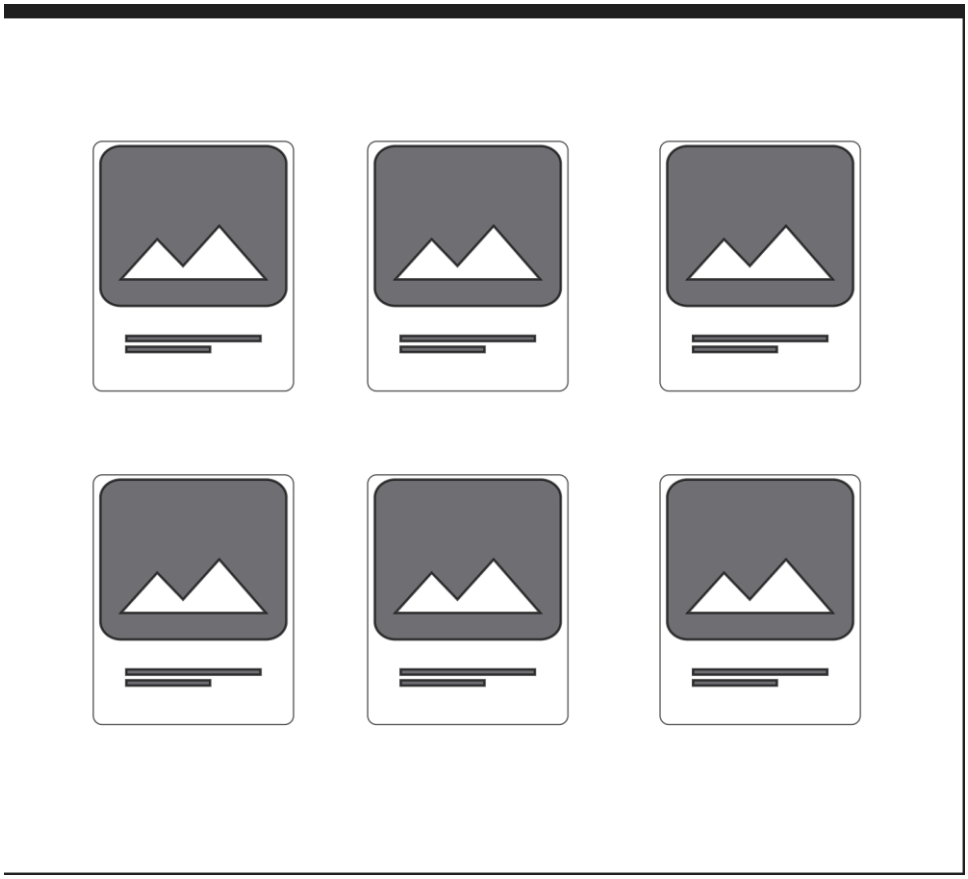
Nota: Es importante que el ejercicio pueda ser levantado facilmente desde una computadora local, por lo que es necesario que se redacte una guía para el levantamiento del sistema y sugerimos que se aseguren que la solución entregada no tenga errores de compilación o de ejecución al momento de levantar el sistema. Y que de ser necesario se faciliten credenciales para accesos a bases de datos u otros recursos de ser necesarios.

Interfaz de usuario (maqueta)

A continuación, solo como sugerencia de la presentación de la galería y el detalle del personaje, la pantalla de login y registro de usuarios es un componente tradicional del cual se puede tomar referencia de cualquier formulario.

Cabe recalcar que queda a criterio del desarrollador el uso de estilos y modelado y componentes (botones, labels, contenedores, etx.) para la implementación de las interfaces de usuarios. Sin embargo, es claro que buscamos una presentación estética además de funcional.

Galería de personajes



Detalle del personaje