



Universidad Tecnológica Nacional  
Facultad Regional Buenos Aires  
Programación Web Avanzado - 6522

## Proyecto Final

**Instructores:** Jhoan Carrero

**Fecha:** May 20, 2022

# Avatar App

Debido a las nuevas tendencias en el mundo web, se han creado innumerables aplicaciones, las cuales se han destacado por resolver bien sea un problema o una necesidad o bien ofrecer una idea creativa o algun recurso.

Es por este motivo que debemos estar abiertos a nuevas tendencias y poderlas llevar a cabo mediante el uso de la programación, ya que acá no se encuentran limites o barreras naturales en las cosas que se puedan construir.

Asi que llevemos a cabo nuestra primera aplicacion de desarrollo Full-Stack, una aplicacion donde podremos utilizarla en cualquier momento o usarla de base para nuevos proyectos o porque no? emprendimientos.



La idea es la siguiente, ud se encuentra en un equipo de desarrollo de alto nivel en el que se le ha dado unas tareas referidas a una aplicacion con dibujos animados que cumplen con estas consignas:

1. Backend como servicio:

Un backend como servicio sera el encargado de procesar las peticiones HTTP, bien sea para obtener, registrar, actualizar o borrar datos. Este backend sera encargado de procesar la informacion que se declara a continuacion usando conexion hacia una base de datos sea relacional o no relacional.

Debera estar configurado con las siguientes variables de entorno la conexion con la base de datos:

- (a) DB\_CONNECTION (Podra ser *mysql* o *mongodb*)
- (b) DB\_URI (Sera la uri para la conexion con la base de datos)

protocolo://usuario:contraseña@dominio:puerto/base\_de\_datos

## 1. Backend como Servicio

(a) POST /api/v1/auth/register (Registrar un nuevo usuario):

Request body: El body de la request contiene la informacion para registrar un nuevo usuario que debe cumplir la siguiente estructura

JSON representation

```
1 {  
2     "name": "Nombre del usuario",  
3     "email": "Email del usuario",  
4     "password": "Contrasena",  
5     "avatar": "Nombre del avatar",  
6     "image": "Ruta de la imagen al  
       avatar"  
7 }
```

Response: La respuesta dependera si fue registrado o no.

i. Registro correcto [httpCode=201]

```
1 {  
2     "message": "Creado  
       satisfactoriamente",  
3     "body": {  
4         "name": "...",  
5         "createdAt": "..."  
6     }  
7 }
```

ii. Registro incorrecto [httpCode  $\geq$  400]

```
1 {  
2     "error": "El Personaje ya se  
        encuentra registrado"  
3 }
```

(b) POST /api/v1/auth/login (Registrar login del usuario)

Request body: El body de la request contiene la informacion para registrar el login de un nuevo usuario que debe cumplir la siguiente estructura

JSON representation

```
1 {  
2     "email": "Email del usuario",  
3     "password": "Contrasena"  
4 }
```

Response: La respuesta dependera si fue logueado o no.

i. Login correcto [httpCode=201]

```
1 {  
2     "message": "Logueado  
        satisfactoriamente",  
3     "body": {  
4         "token": "$2b$10$F1JVwqlKBdigb1  
                MGgYpxXOUGjv8JRCw0mcdDf7fimPU  
                5R0n.9rM6e"  
5     }  
6 }
```

ii. Login incorrecto [httpCode  $\geq$  400]

```
1 {
2     "error": "Credenciales incorrectas"
3 }
```

(c) GET /api/v1/user/avatar

Request body: El body de la request contiene la informacion del token de autorizacion para la api

JSON representation

```
1 {
2     "token" : "$2b$10$F1JVwqlKBdigb1
3     MGgYpxXOUGjv8JRCw0mcdDf7fimPU5R0n.9
    rM6e"
}
```

Response: La respuesta dependera si fue autenticado anteriormente o no.

i. Actualizado correcto [httpCode=201]

```
1 {
2     "message": "Lista de avatars en la
3     base de datos",
4     "body": [
5         "Avatar 1",
6         "Avatar 2",
7         ...
8     ]
}
```

ii. Peticion incorrecta [ $\text{httpCode} \geq 400$ ]

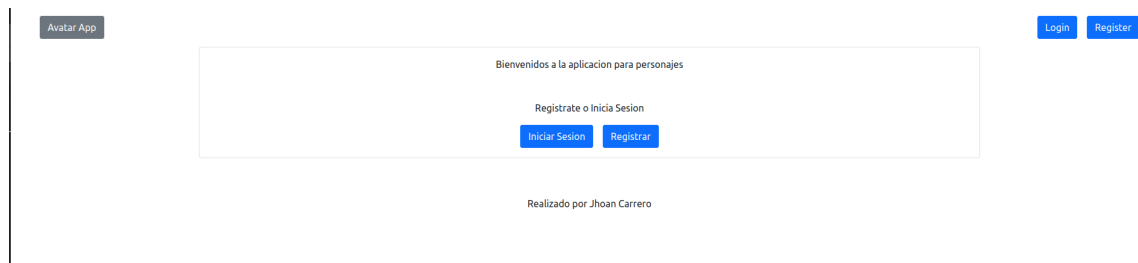
```
1 {  
2   "error": "Debe estar logueado para  
   este recurso"  
3 }
```

## 2. Frontend como Servicio

Un frontend como servicio sera el encargado de procesar las interacciones con el cliente, donde este sera el encargado de comunicarse con el backend para autorizar los recursos y con una api (<https://rickandmortyapi.com/>).

Debera contar con las siguientes vistas:

(a) Vista Principal [ Ruta = "/" ]



Esta vista sera la encargada de mostrar links o botones para navegar hacia '/login' '/register' '/dashboard' '/logout'

(b) Vista Registro [ Ruta = "/register" ]

Avatar App Login Register

User Name  
Enter your name

Email address  
Enter email

Password  
\*\*\*\*\*

Select your avatar  
Select your avatar

Submit

Realizado por Jhoan Carrero

Avatar App Login Register

User Name  
Enter your name

Email address  
Enter email

Password  
\*\*\*\*\*

Select your avatar  
Albert Einstein

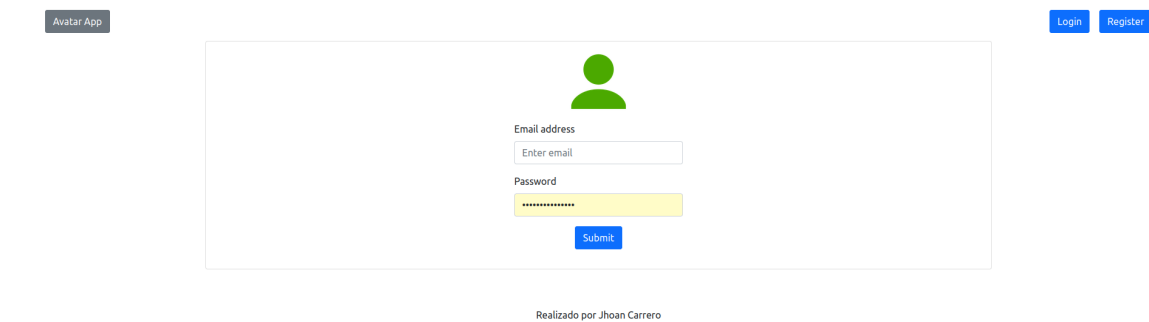
Submit

Realizado por Jhoan Carrero

Esta vista sera la encargada de recibir los datos del usuario a registrar (nombre, correo y contraseña), ademas de ello un avatar y la imagen del avatar que desea registrar, para ello debera consultar a su "Backend como servicio" que avatar hay en la aplicacion ( GET: `api/v1/user/avatar` ) y tambien consultar cuales hay disponibles con la API (`https://rickandmortyapi.com/`), con el proposito de no repetir avatars entre usuarios.




(c) Vista Login [ Ruta = "/login" ]



Avatar App

Login Register



Email address

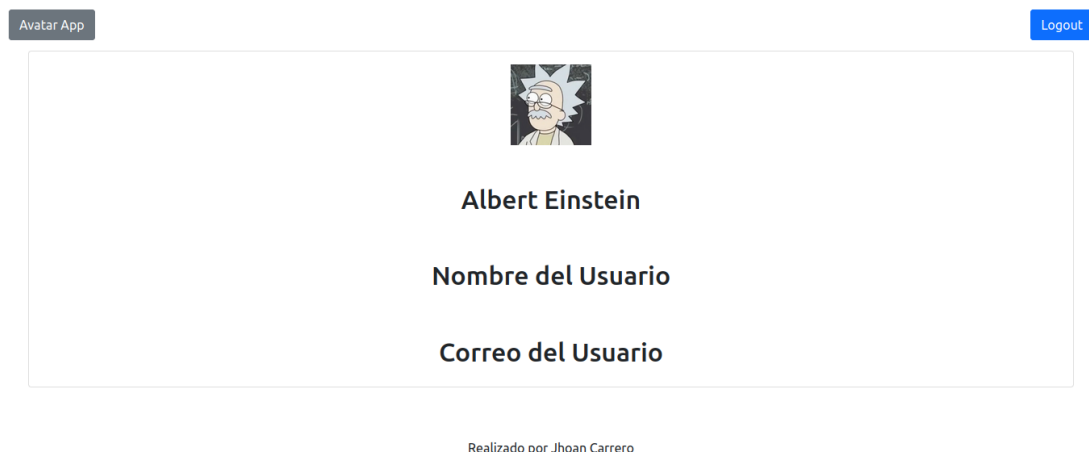
Password

Submit

Realizado por Jhoan Carrero


Esta vista sera la encargada de comunicarse con su "Backend como servicio" y validar el inicio de sesion para realizar la posterior redireccion al dashboard.

(d) Vista Dashboard [ Ruta = "/dashboard" ]



Avatar App

Logout



**Albert Einstein**

**Nombre del Usuario**

**Correo del Usuario**

Realizado por Jhoan Carrero

---

Esta vista sera la encargada de mostrar todos los datos del usuario, como el nombre, el correo, el avatar y su imagen. Asi como un boton de "logout" que posteriormente solo invalidara el token.