



Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Programación Web Avanzado - 6822

Proyecto Final

Instructores: Jhoan Carrero

Fecha: Mayo 26, 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": "Contraseña",  
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 [$\text{httpCode} \geq 400$]

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

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

Response: La respuesta contendrá la siguiente estructura

```
1 {  
2     "message": "Lista de avatars en la  
3     base de datos",  
4     "body": [  
5         "Avatar 1",  
6         "Avatar 2",  
7         ...  
8     ]  
9 }
```

(d) GET /api/v1/user/profile?**token**=\$2b\$10\$FlJVwqlK...

Response: La respuesta dependera si el token es valido o no.

i. Token correcto [httpCode=200]

```
1 {
2     "message": "Perfil del usuario con
3     toda su informacion",
4     "body": {
5         "name": "Nombre del usuario",
6         "email": "Email del usuario",
7         "avatar": "Nombre del avatar",
8         "image": "Ruta de la imagen al
9         avatar"
10    }
11 }
```

ii. Token incorrecto [httpCode \geq 400]

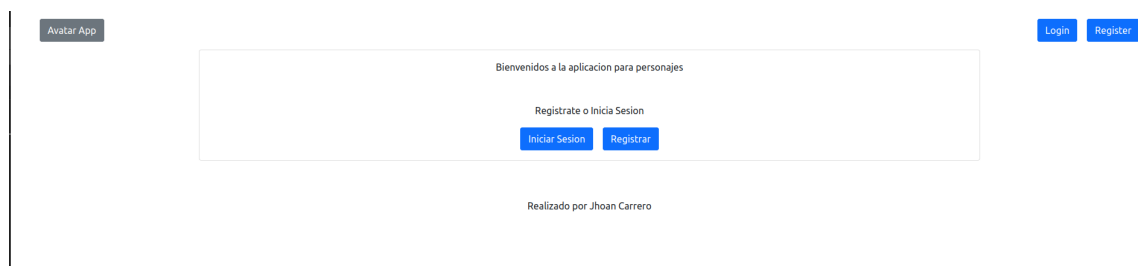
```
1 {
2     "error": "Token incorrecto"
3 }
```

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://pokeapi.co/docs/v2>).

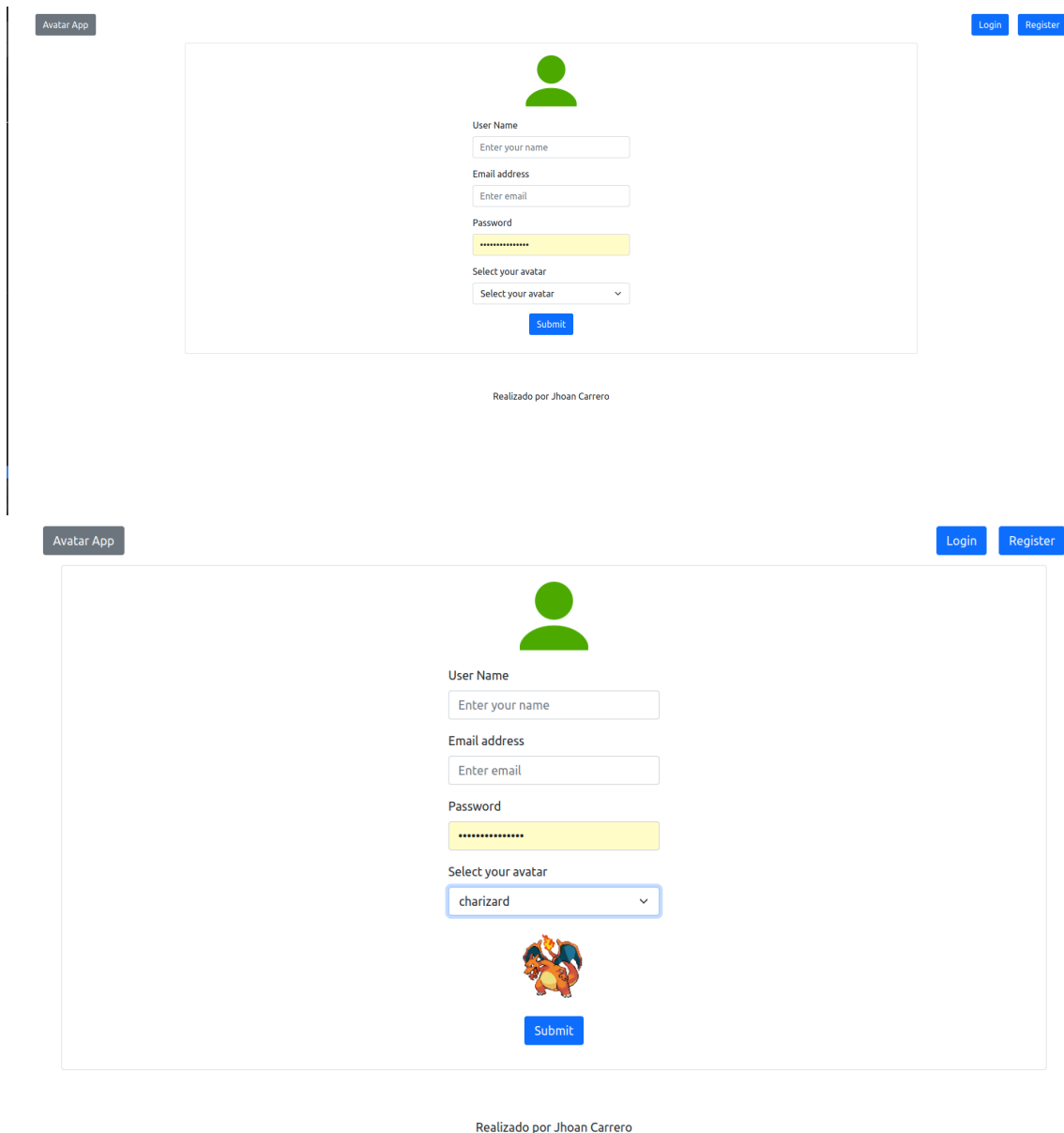
Debera contar con las siguientes vistas:

(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"]



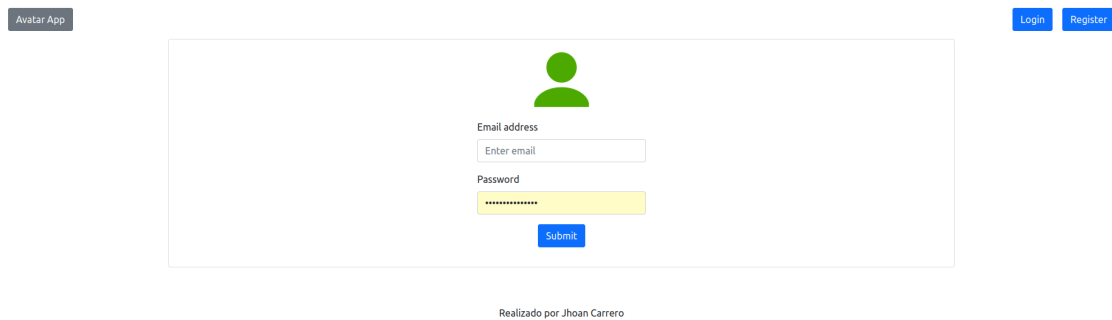
The image displays two screenshots of a web application's registration form, labeled 'Avatar App' in the top left corner. The form is centered and contains the following fields and elements:

- User Name:** A text input field with the placeholder 'Enter your name'.
- Email address:** A text input field with the placeholder 'Enter email'.
- Password:** A text input field with masked characters (dots).
- Select your avatar:** A dropdown menu with the placeholder 'Select your avatar'.
- Submit:** A blue button located below the avatar selection dropdown.

In the top screenshot, the avatar selection dropdown is open, showing a generic green circle icon. In the bottom screenshot, the dropdown menu is closed, and the selected avatar, 'charizard', is displayed as a small image below the dropdown. The text 'Realizado por Jhoan Carrero' is visible at the bottom of both screenshots.

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://pokeapi.co/docs/v2>), con el proposito de no repetir avatars entre usuarios.

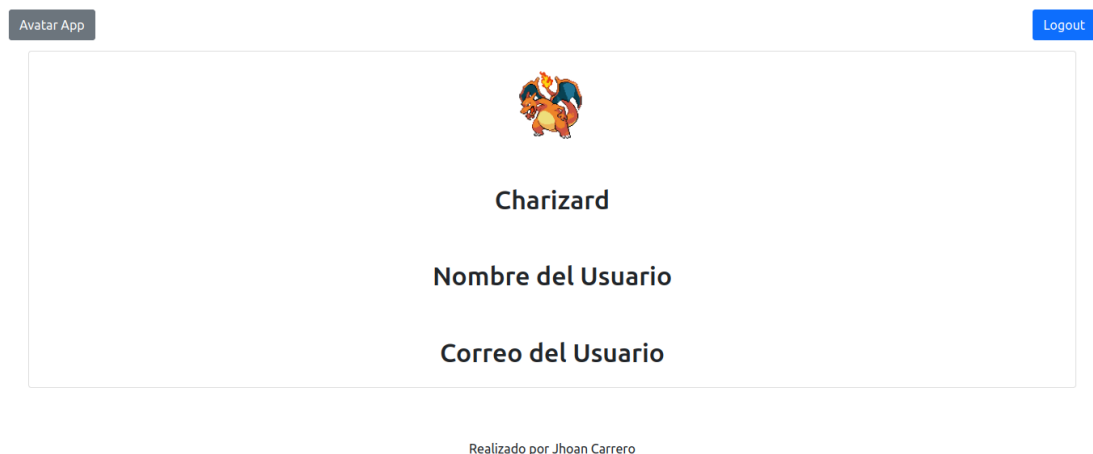
(c) Vista Login [Ruta = ”/login”]



The screenshot shows the login interface of the 'Avatar App'. At the top left is a dark grey button labeled 'Avatar App'. At the top right are two blue buttons labeled 'Login' and 'Register'. The main content area is a white box containing a green circular avatar icon. Below the icon are two input fields: 'Email address' with the placeholder text 'Enter email', and 'Password' with a masked password '*****'. A blue 'Submit' button is positioned below the password field. At the bottom center of the page, below the main content box, is the text '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”]



The screenshot shows the dashboard interface of the 'Avatar App'. At the top left is a dark grey button labeled 'Avatar App'. At the top right is a blue button labeled 'Logout'. The main content area is a white box containing a Charizard Pokemon image. Below the image is the text 'Charizard'. Underneath that is the text 'Nombre del Usuario'. At the bottom of the box is the text 'Correo del Usuario'. At the bottom center of the page, below the main content box, is the text '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. haciendo uso de la ruta (GET: api/v1/user/profile) pasando como parametro en la url el token del login. Asi como un boton de ”logout” que posteriormente solo invalidara el token.

Entrega

La fecha de entrega culmina el día **06/07/2022**, es importante que todos los cambios o dudas que tengan las aclaren antes de esta fecha.

El método de entrega del proyecto se realizará a través de Github en sus ramas correspondientes, respetando la estructura de carpetas.

```
utn
├── nombre_apellido
│   ├── lessons
│   ├── practices
│   └── project
```

Todo el código tanto del Backend como del Frontend se encontrará en la carpeta **project**. Luego de subido al repositorio se deberá pedir una Pull Request con el siguiente título:

Entrega Final - Nombre Apellido