1. **Proceso de login y creación de usuario:**
2. El usuario se loguea en el Portal e ingresa al Chat.
3. **Request:** *get('/user', 'UserController@getUserLoggedJWT');*
   1. Si el usuario ya existe → devuelve el objeto Usuario
   2. Si el usuario NO existe → devuelve 404
      1. **Request:** *Route::post('/newUser', 'UserController@createUserJWT');*
         1. Si el **user\_name** enviado por JWT NO corresponde a un usuario existente, devuelve el objeto Usuario recién creado.
         2. Si el **user\_name** enviado por JWT corresponde a un usuario existente, devuelve un error 422, con el siguiente mensaje:

{

"error": "Ya existe el usuario con el user\_name: vblanco"

}

1. **Proceso de actualización de usuario por parte de un ADMIN – Designación de un usuario como Administrador:**
2. El usuario ingresa al Chat, se valida que el usuario sea Administrador.
3. **Request:** *get('/admin/user/{user\_id}', 'UserController@getUserManagedJWT');*
   1. Se obtiene la información del usuario solicitado.
4. ***Request:*** *post('/admin/user/{user\_id}', 'UserController@updateUserManagedJWT');*
   1. El Administrador podrá actualizar los datos personales de un usuario (NO podrá actualizar el user\_name, ya que el mismo fue tomado del JWT cuando éste se creó) y también designarlo como Administrador o viceversa.
5. **Proceso de actualización de usuario por parte del mismo:**
6. El usuario ingresa al Chat con un usuario existente.
7. **Request:** *get('/user', 'UserController@getUserLoggedJWT');* 
   1. Se obtiene la información del usuario logueado.
8. ***Request:*** *post('/admin/user/{user\_id}',* [*'UserController@updateUserManagedJWT*](mailto:'UserController@updateUserManagedJWT)*');*
   1. El usuario podrá actualizar sus datos personales.
   2. NO podrá actualizar el user\_name, ya que el mismo fue tomado del JWT cuando éste se creó)
   3. NO podrá cambiar el dato de si es o no Administrador.
9. **APIs de Autogestión de Usuarios**
10. ***Route::get('/user', 'UserController@getUserLoggedJWT');*** //Esta API también se llama después del login contra el portal, si EXISTE el usuario -> devuelve el objeto usuario, SI NO existe -> devuelve 404
    1. El token DEBE traer el user\_name cuando la API se llama inmediatamente después del login (todavía el front NO sabe el user\_id)
    2. El token DEBE traer el user\_id cuando la API se llama para que un usuario se autoadministre/visualice su cuenta.
11. ***Route::put('/user', 'UserController@updateUserLoggedJWT');***
    1. El token DEBE traer el user\_id, la API se llama para que un usuario se autoadministre/actualice su cuenta.
    2. *Valores pasados por el body:*

'name' => ['sometimes','string', 'max:255'],

'surname' => ['sometimes','string', 'max:255'],

'grade' => ['sometimes','nullable','string'],

'dni' => ['sometimes','integer'],

'location\_id' => ['sometimes','integer', 'exists:locations,id'],

1. ***Route::post('/user', 'UserController@createUserJWT');***
   1. El token DEBE traer el user\_name del usuario que desea crearse.
   2. *Valores pasados por el body:*

'name' => ['required','string', 'max:255'],

'surname' => ['required','string', 'max:255'],

'grade' => ['sometimes','nullable','string'],

'dni' => ['required','integer'],

'location\_id' => ['required','integer', 'exists:locations,id']

* 1. SALIDA:
     1. Para una petición correctamente ejecutada, se obtendrá el código de estado “**200 OK”** y la estructura de datos que devolverá el JSON es la siguiente:
     2. Si el **user\_name** enviado por JWT corresponde a un usuario existente, devuelve un error 422, con el siguiente mensaje:

{

"error": "Ya existe el usuario con el user\_name: vblanco"

}

* + 1. Si el JWT no contiene el valor de user\_name, devuelve un error 422, con el siguiente mensaje:

{

"error": "El token NO contiene el user\_name."

}

1. **APIs de Administración de Usuarios**
2. ***Route::get('/admin/user/{user\_id}', 'UserController@getUserManagedJWT');***
   1. El token DEBE traer el user\_id del ADMIN que se validará si es un usuario Administrador.
   2. En la URL se envía el ID del usuario del cual se está pidiendo información.
3. ***Route::post('/admin/user/{user\_id}', 'UserController@updateUserManagedJWT');***
   1. El token DEBE traer el user\_id del ADMIN que se validará si es un usuario Administrador.
   2. En la URL se envía el ID del usuario que se desea actualizar.
   3. *Valores pasados por el body:*

'name' => ['sometimes','string', 'max:255'],

'surname' => ['sometimes','string', 'max:255'],

'grade' => ['sometimes','nullable','string'],

'dni' => ['sometimes','integer'],

'location\_id' => ['sometimes','integer', 'exists:locations,id'],

'admin' => ['sometimes','boolean'],