**Securing APIs**

**Token-based Authentication**

* Tokens are used to secure APIs and decide if user is allowed to access resources or not.
  + If allowed will send the result.
  + If not allowed will send an error.
* Explanation:
  + when user create an account, they provide information such as username, email and password.
  + Next time they want to access the system they need to provide required information.
    - If this info is valid, a token will be sent to user.
    - If not valid will send an error.
  + When they try to access specific resources at endpoint, they need to provide the token for example /profile.
  + The user must provide the token, and when user is authenticated a profile info will be provided.
    - If not, will receive an error.

## Adding the Authentication Endpoints

* In order to use authentication in Django for APIs I need to use a library called Djoser.
  + url: <https://djoser.readthedocs.io/en/latest/index.html>
  + go to installation page to download it and use it (and authentication backends for full installation like urls and rest\_framework inside settings)
* There are two ways to add auth system:
  + Rest\_framework authentication: this approach store, the token in the database and on each request, I need to make a query to db to check everytime user make a request.
  + Jwt based auth: this approach does not require database because the jwt has its signature.
* So, I need to add urls for both the joser and the joser.jwt
* Also, I need to add to add it (authentication backends) in rest\_framework inside setting.
* And add the simple\_jwt setting also I can find it inside documentation (authentication backends)

URL to access my endpoint is: /auth/

## Registering Users

* **to** register a user, I need to go to /auth/users endpoint and fill required information.
* If I want to customize the signup required info, I need to go to Djoser serializers and find userCreate serializer.
* Then I go to the base serializer implementation and find how it was implemented.
* I will find class Meta and inside of it there is fields username, pass and email.
* I inherit inside my custom serializer create a meta class that should inherit from the base meta as well (if there is an update, I will inherit these functionalities.)
* Inside fields I define all the fields I want to implement.
* Then I go to setting and add DJOSER = {

‘SERIALIZERS’: {

‘user\_create’: ‘path to my serializer’

}

}

* And happy signup process :)

## Building the Profile API

* To create a profile, I need to create serializer + view + add url to my url list.
* A note about profile: I don’t want user to access list of users so I need to be specific on what user can do.
  + user can create, update and retrieve his profile, but can’t delete and can’t see list of users.

**Logging In**

* to login a user I need to use this url: /auth/jwt/create.
* I will receive two tokens: lifetime token and refresh toke.
  + Lifetime default is 5 minutes when it’s expired, I use refresh token to generate another one and refresh token is 1 day.
* To change the default setting I go to setting file and add SIMPLE\_JWT = { ‘ACCESS\_TOKEN\_LIFETIME’: timedelta(days=1 or minutes = 10)}

More information about simple jwt: <https://django-rest-framework-simplejwt.readthedocs.io/en/latest/index.html>

**Refresh Token**

* to refresh a token after being expired, I need to use this url: /auth/jwt/refresh.
* When access token is expired and I make a request I will receive 401 unauthorized then I send a request for refresh token.

**Get Current User**

* to get current user, I need to use this url: /auth/users/me.
* I need to provide the jwt to header and it’s a json object like this:

{“Authorization”: “JWT myToken”}

* If I want to customize my response, I need to first import UserSerializer and extend my class from it.
  + Then create my custom Meta (extends Base.Meta):
    - I redefine the fields property and add fields I want to restore.