June 5, 2022

Chia yi quan

Student Number: 210220223

Advanced web development

report for coursework 1(rest api)

Table of Contents

**Type chapter title (level 1)1**

Type chapter title (level 2)2

Type chapter title (level 3)3

**Type chapter title (level 1)4**

Type chapter title (level 2)5

Type chapter title (level 3)6

# Introduction

The purpose of this report is to document down on the bioscience project. The report will be separated into 2 sections which is running of the projects and code explanation. The first section will cover setting up of the project, running migration and script, running of the project, running of test case and code explanation of the required api endpoint. The second section will explain the code in the scripts, models design, serializer and the api code.

# Running Of The Projects

## Setting Up Of The Project

Inside the folder that was submitted, there should be 2 items which is a requirements.txt and bioscience folder. Firstly, we have to setup the virtual environment before going to the migration. Open a terminal and navigate into the project folder(the folder where the requirements.txt and bioscience folder is in). You can create a new virtual environment with the following command.

**py -m venv env\_name** -> Replace the env\_name with the name you want.

Text

Description automatically generated

This will create a env folder in the directory.



In the terminal, change the directory to the Scripts folder which in my case will be cd bioscience\_env/Scripts and type in the following command, the command is different depending on the operating system that is used.

**activate.bat** -> For windows

**source activate.sh** -> For unix system

If done successfully, the env name should appear at the front of the directory.

Text

Description automatically generated

Now we have to change directory back to the project folder which is 2 directory above the current directory we are in, so in the terminal we can execute **cd ..** for 2 time. After changing the working directory to the project folder, we can run the following command to install the dependencies.

**pip install -r ./requirements.txt**

After running the command above you should see this screen

Text

Description automatically generated

After installing the dependencies, we can continue to the next step but do not close the terminal yet.

## Migration Of The Project

In the terminal(with the env activated), change directory into bioscience folder. After changing the directory to the folder, we can run the following command in the terminal to do a migration. For the python command it depend on the system global variable which can be python3 py etc..

**python manage.py migrate**

All the models will be migrated into the default database which is the sqlite.

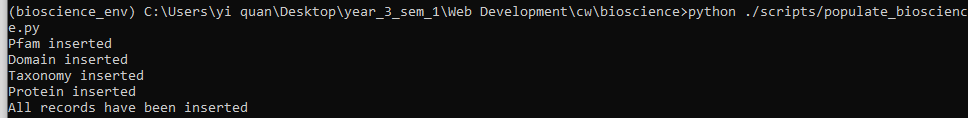
Text

Description automatically generated

Next, we will run the script to import all the csv data into the database, I will explain all the scripts codes at the later part of the report but for now we will run this command to execute the scripts.

**python ./scripts/populate\_bioscience.py**

It will take around 5 minutes if is on SSD and around 10 to 15 minutes for HDD for the scripts to execute finish. However, changing the database to postgres will improve the migration speed. If all data have been inserted in successfully, the screen will appear as the following screenshot.



## Running Of The Project

To start the project, simply just run the command below.

**python manage.py runserver**

Once the server started, the list of api is as listed:

**[POST]** <http://localhost:8000/api/protein/>

The structure to submit a new protein must be in this format

{

"protein\_id": "",

"sequence": "",

"taxonomy": {

"taxa\_id": null,

"clade": "",

"genus": "",

"species": ""

},

"length": null,

"domains": []

}

**Example:**

{

"protein\_id": "A0A016S8J7-duplicate",

"sequence": "MVIGVGFLLVLFSSSVLGILNAGVQLRIEELFDTPGHTNNWAVLVCTSRFWFNYRHVSNVLALYHTVKRLGIPDSNIILMLAEDVPCNPRNPRPEAAVLSA",

"taxonomy": {

"taxa\_id": 53326,

"clade": "E",

"genus": "Ancylostoma",

"species": "ceylanicum"

},

"length": 101,

"domains": [

{

"pfam\_id": {

"domain\_id": "PF01650",

"domain\_description": "PeptidaseC13family"

},

"description": "Peptidase C13 legumain",

"start": 40,

"stop": 94

},

{

"pfam\_id": {

"domain\_id": "PF02931",

"domain\_description": "Neurotransmitter-gatedion-channelligandbindingdomain"

},

"description": "Neurotransmitter-gated ion-channel ligand-binding domain",

"start": 23,

"stop": 39

}

]

}

The result of the post request will be like the following screenshot.

Graphical user interface, text, application

Description automatically generated

**[GET]** [http://localhost:8000/api/protein/[PROTEIN\_ID](http://localhost:8000/api/protein/%5bPROTEIN_ID)]

**Example:**

<http://localhost:8000/api/protein/A0A016S8J7-duplicate/>

The result of the get request will be like the following screenshot.

Graphical user interface, text, application, email

Description automatically generated

**[GET]** [http://localhost:8000/api/pfam/[PFAM\_ID](http://localhost:8000/api/pfam/%5bPFAM_ID)]

**Example:**

<http://localhost:8000/api/pfam/PF00360/>

The result of the get request will be like the following screenshot.

Graphical user interface, text, application, email

Description automatically generated

**[GET]** [http://localhost:8000/api/proteins/[TAXA\_ID](http://localhost:8000/api/proteins/%5bTAXA_ID)]

**Example:**

<http://localhost:8000/api/proteins/55661/>

The result of the get request will be like the following screenshot.

Graphical user interface, application

Description automatically generated

**[GET]** [http://localhost:8000/api/pfams/[TAXA\_ID](http://localhost:8000/api/pfams/%5bTAXA_ID)]

**Example:**

<http://127.0.0.1:8000/api/pfams/55661/>

The result of the get request will be like the following screenshot.

Text

Description automatically generated with low confidence

**[GET]** [http://localhost:8000/api/coverage/[PROTEIN\_ID](http://localhost:8000/api/coverage/%5bPROTEIN_ID)]

**Example:**

<http://localhost:8000/api/coverage/A0A016S8J7/>

The result of the get request will be like the following screenshot.

Table

Description automatically generated

## Running The Test Case

To run the test case, simply run the command.

**python manage.py test**

There are a total of 31 test case which will be like the following screenshot

Text

Description automatically generated