Building Neo4j Applications with Python

Project Setup

As part of this course, you will work on a pre-built repository(https://github.com/neo4j-graphacademy/app-python) for the fictional client **Neoflix**. The course is designed to be framework agnostic, so although we have chosen a specific framework, the tasks that you will perform will be the same regardless of your choice of framework.

In the early stages you will learn some of the theory required and then use that knowledge to implement a set of features in the API.

Repository Information

We have built a repository that takes care of the boiler plate, so you can focus on implementing the functionality

- The project is designed to work with Python version 3.9
- Packages can be installed with pip
- A web server has been built with Flask
  - Authentication is handled with JWT Tokens and PyJWT
  - Passwords are encrypted and verified with bcrypt
  - Testing is performed using pytest

Setup Python

If you haven't already installed Python, you must **install** *Python3* using the instructions on www.python.org. The project has been written to work wth Python version *v3.9.9*.

You can verify that the installation is successful by running the following command in the command line:

Verify Python Version

```
python --version
```

Clone the Repository

To clone the repository without logging in via HTTPS, you can run the following command to clone the repository:

```
git clone https://github.com/neo4j-graphacademy/app-python.git
```

Set up a new Environment

To create a virtual environment named sandbox, you can run the following:

```
python -m venv sandbox
```

To activate the virtual enviroment named sandbox, you can run the following:

```
source sandbox/bin/activate
```

Install Dependencies

Once you have cloned the repository, navigate to the folder in your terminal and run the following command to install the dependencies.

```
pip install -r requirements.txt
```

`$ requirements.txt`

```
txt
attrs==21.2.0
autopep8==1.6.0
backports.entry-points-selectable==1.1.1
bcrypt==3.2.0
certifi==2021.10.8
cffi==1.15.0
charset-normalizer==2.0.9
click==8.0.3
cryptography==36.0.0
distlib==0.3.3
docopt==0.6.2
filelock==3.4.0
Flask==2.0.2
Flask-JWT==0.3.2
idna==3.3
iniconfig==1.1.1
itsdangerous==2.0.1
Jinja2==3.0.3
MarkupSafe==2.0.1
neo4j==4.4.0
packaging==21.3
pipreqs==0.4.11
platformdirs==2.4.0
pluggy==1.0.0
py==1.11.0
pycodestyle==2.8.0
pycparser==2.21
PyJWT==1.4.2
pyparsing==3.0.6
pytest==6.2.5
python-dotenv==0.19.2
pytz==2021.3
requests==2.26.0
six==1.16.0
toml==0.10.2
urllib3==1.26.7
virtualenv==20.10.0
Werkzeug==2.0.2
yarg==0.1.9
```

Setting Environment Variables

This project will read environment variables from the `.env` file located in the project root.

The project contains an example file at `.env.example`. You can run the following command in your terminal window to copy the example file to `.env`.

```
cp .env.example .env
```

Your `.env` file should have the following settings:

```
env
FLASK_APP=api                       # (1)
FLASK_ENV=development               # (2)
FLASK_RUN_PORT=3000                 # (3)
JWT_SECRET=secret                   # (4)
SALT_ROUNDS=10                      # (5)

NEO4J_URI=neo4j://localhost:7687    # (6)
NEO4J_USERNAME=neo4j                # (7)
NEO4J_PASSWORD=password             # (8)
```

- `FLASK_APP` tells Flask to use the application in the `api/` folder
- Run Flask in Development Mode
- Run the server on port `3000`
- A secret key for signing JWT tokens - *This can be a random string of letters and numbers*
- The cost parameter used when hashing passwords
- The URI for your Neo4j Sandbox instance, we will set this in the next lesson
- The username for your Neo4j Sandbox instance
- The password for your Neo4j Sandbox instance

Start the Project

To **start the project**, run the following command:

```
flask run
```

You should see an output similar to the following confirming that the server has successfully started:

```
* Serving Flask app 'api' (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://127.0.0.1:53000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 566-751-036
```

The REST API will listen for requests on http://localhost:3000.

A Brief Tour of the Project

If you open up the listening address in your browser, you will see a Single Page Application (SPA) that communicates with the API served at http://localhost:3000/api. Currently, the responses are hardcoded, but as you progress through the course, you will learn how to query Neo4j to find this information.

Here are some of the important directories in the project:

- `example/` - Example code for working with the driver.
- `api/` - The application code:
  - `dao/` - Data Access Objects which will be modified to communicate with Neo4j
  - `middleware/` - Some custom middleware functions that are used by Flask throughout the request lifecycle
  - `routes/` - Route handlers that are registered on the server. You shouldn't need to edit these files. - `public/` - Minified build files for the SPA. **Do not edit these files.**