Department of Computer Science and Information Systems

Birkbeck, University of London

2020

Coursework

Konstantin Orlovskiy, Student ID: 13157188, MSc Data Science

Academic Declaration

I have read and understood the sections of plagiarism in the College Policy on assessment offences and confirm that the work is my own, with the work of others clearly acknowledged. I give my permission to submit my report to the plagiarism testing database that the College is using and test it using plagiarism detection software, search engines or meta-searching software.

Konstantin Orlovskiy  04/04/2020

Phase A [10 marks]
Install and deploy software in virtualised environments

For the purposes of this coursework the BBK virtual machine was used:
ssh student@193.61.36.109
password: 2020

In this VM the virtual environment “api-coursework” was created and activated. Inside this environment Django was installed as well as all the required packages (REST, oAuth toolkit) and added to installed apps.

The auctioning website is a Django project “mysite” which was created in the virtual environment. It contained the folder “mysite” this is why parent folder “mysite” was renamed to “src” for the better usability during development process.

Inside the folder “src” the application “users” was created and registered as application, migrations applied. Also, superuser was created:
superuser name: admin
superuser password: 2020

Using the credentials above the first endpoint can be accessed:
193.61.36.109:8000/admin/

Super user can now assess and edit the data using web browser.

193.61.36.109:8000/o/ # oAuth provider URL which is added to “urls.py” for the authentication purposes following the oAuth v2 protocol.

Phase B [20 marks]
Development of authorisation and authentication services

For the authentication services the RESTful API framework was installed and registered in settings.py. In order to have an option of creating more users the serializers.py file was populated with the code where class CreateUserSerializer created.

In order to make the actions with users accessible from the API the endpoints were added in file “users/views.py” to show the data and in file “mysite/urls.py” to link the views to URL. As a result the following endpoints created:

http://193.61.36.109:8000/authentication/register/ # New users registration.
http://193.61.36.109:8000/authentication/token/ # Request token.
http://193.61.36.109:8000/authentication/token/refresh/ # Refresh token.
http://193.61.36.109:8000/authentication/token/revoke/ # Revoke token.

Using the oAuth2 allows avoiding unauthorised users accessing the website resources. Firstly, the user needs to get registered. After the registration the user is

returned the token which is valid for 10 hours and can be used for accessing the internal resources by including the token in the request header. Examples are shown in test cases file.

The way oAuth is used can be followed in “mysite/settings.py” where the respected lines were added:

1. INSTALLED_APPS: This is where Django discovers your apps. Added OAuth2.
2. MIDDLEWARE: Each time a request is coming into Django server the middleware hooks into Django’s request/response processing. Added the line for OAuth2 here.
3. REST_FRAMEWORK: These are the settings for Django REST to accept OAuth2 as authentication.
4. AUTHENTICATION_BACKENDS: This specifies what authentication system should be used.

As next step a new application was created to let us create new users:
Home › Django OAuth Toolkit › Applications › Add application.
Choose user “admin” that was previously created and add Client id and Client secret.

Client id:
OXyeRyKdRrI8qaFb9Pn5AyHGr0jJICz3mfbduuII

Client secret:
9vM4RGaD2hUcVnzie5Pn36uvsFAYMqlSL2CGywdoWaSi21CSFjgoABE09PwEVwm48NcWgOwnzpeIYlaRb5LRuvAY84KcIEIWUKnrQPEDSShb0LLLwhnLC1TxboA9VFsp

Then the views API created to allow the following calls:

- Register new user
- Get token
- Refresh token
- Revoke token

In order to do so the code provided in Moodle for Lab 4 was used to paste into “users/views.py”. Additionally, urls.py updated to make the authentication urls available in browser.

Phase C [35 marks]
Development of Auctioning RESTful APIs

For the purpose of managing the resources the application “resources” was created inside the folder “src”.

As there’s a need to post items, keep track on auction status and bids, three classes were created in “resources/models.py”:

Item - it is connected to User
Auction - it is connected to Item
Bid - it is connected to Auction

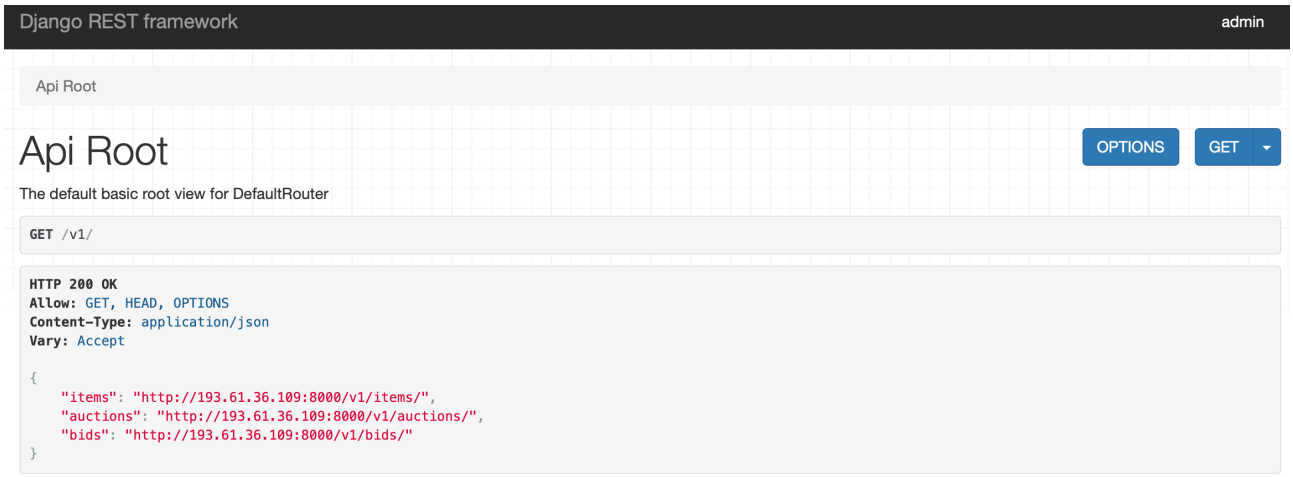
Then the respecting serializers were created as well as views. Views were connected by router to “mysite/urls.py” to make them assessable externally.

As using ForeignKey first argument requires the primary key which I was unable to get as response via accessing using API, the connections between Bid—>Auction are made with CharField as a temporary solution to continue the project development process.

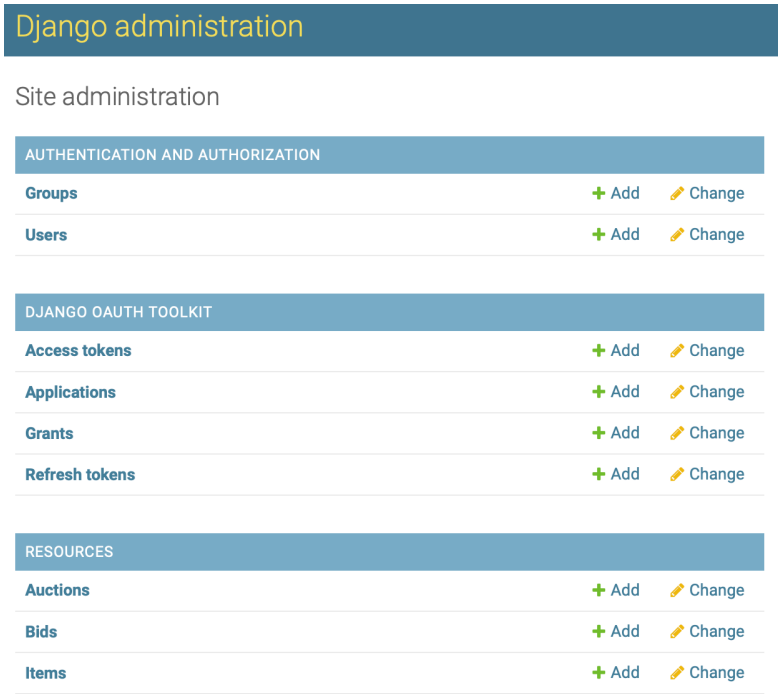
This is the first version of API so the following url was registered:
http://193.61.36.109:8000/v1/

It has three subfolders for accessing different resources respectively.

http://193.61.36.109:8000/v1/items/
http://193.61.36.109:8000/v1/auctions/
http://193.61.36.109:8000/v1/bids/



All resources are also available via admin panel.



Phase D [20 marks]
Development of a testing application

Per the coursework guidelines the test cases were developed outside the virtual environment and are saved as “CC_CW_Tester.ipynb” in the ZIP file uploaded in Moodle.

Annex

Current folders structure:

```
student@cc:~/api-coursework/src$ tree
.
├── db.sqlite3
├── manage.py
├── mysite
│   ├── asgi.py
│   ├── __init__.py
│   ├── __pycache__
│   │   ├── __init__.cpython-36.pyc
│   │   ├── settings.cpython-36.pyc
│   │   ├── urls.cpython-36.pyc
│   │   └── wsgi.cpython-36.pyc
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── resources
│   ├── admin.py
│   ├── apps.py
│   ├── __init__.py
│   ├── migrations
│   │   ├── 0001_initial.py
│   │   ├── __init__.py
│   │   └── __pycache__
│   │       ├── 0001_initial.cpython-36.pyc
│   │       └── __init__.cpython-36.pyc
│   ├── models.py
│   ├── __pycache__
│   │   ├── admin.cpython-36.pyc
│   │   ├── __init__.cpython-36.pyc
│   │   ├── models.cpython-36.pyc
│   │   ├── serializers.cpython-36.pyc
│   │   └── views.cpython-36.pyc
│   ├── serializers.py
│   ├── tests.py
│   └── views.py
└── users
    ├── admin.py
    ├── apps.py
    ├── __init__.py
    ├── migrations
    │   ├── __init__.py
    │   └── __pycache__
    │       └── __init__.cpython-36.pyc
    ├── models.py
    ├── __pycache__
    │   ├── admin.cpython-36.pyc
    │   ├── __init__.cpython-36.pyc
    │   ├── models.cpython-36.pyc
    │   ├── serializers.cpython-36.pyc
    │   ├── urls.cpython-36.pyc
    │   └── views.cpython-36.pyc
    ├── serializers.py
    ├── tests.py
    ├── urls.py
    └── views.py

10 directories, 43 files
```