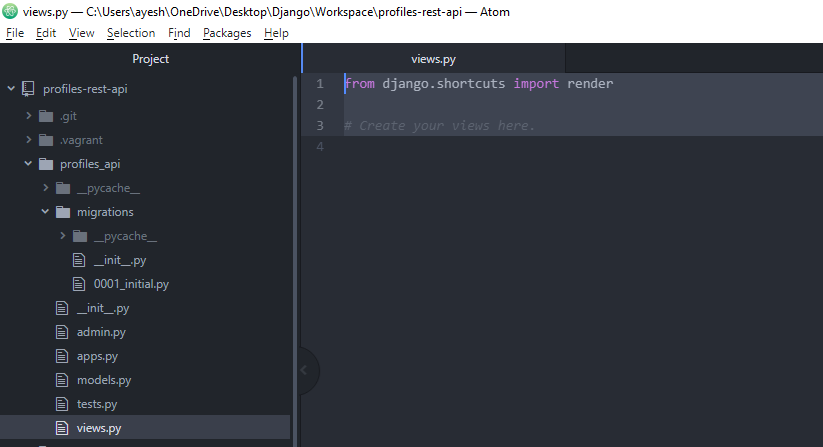
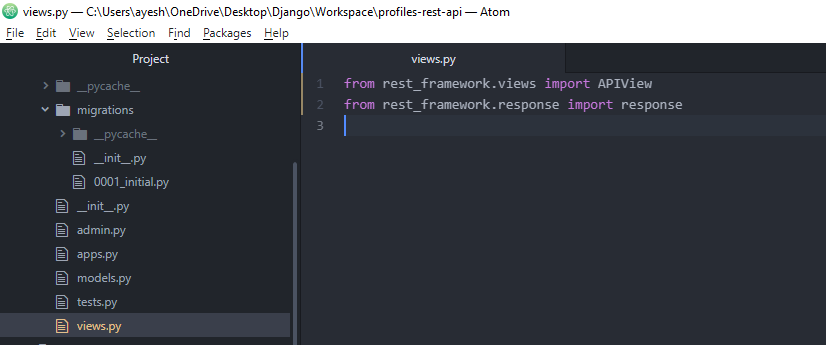
**Create API View:**

1. In Atom editor, go to views.py file (created under profiles\_api app in our project)



2. Perform Imports



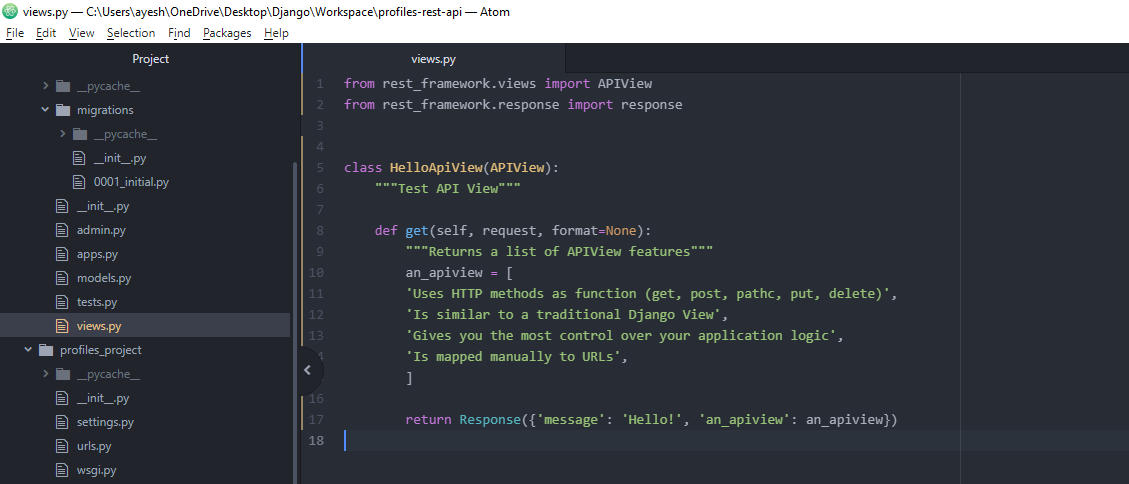
Imports API View class from the Django rest\_framework.views module

Imports response object which is used to return responses from the API View. So it’s the standard response object which gets called when we or DRF calls the API view.

3. Creating our API View class (to define the application logic for our endpoint)

Define a URL and assign it to the view and the DRF handles it by calling the appropriate function in the view for the HTTP request we make.

Accepting HTTP get request - used to retrieve a list of objects or a specific object.



request – contains details of the request being made to the API

format – used to add a format suffix to the end of the endpoint URL

get() - define a list which describes all of the features of an API view

Every function added to the API view (HTTP function) such as get, post, patch, put, delete must return a response which the DRF will send as output when the API is called.

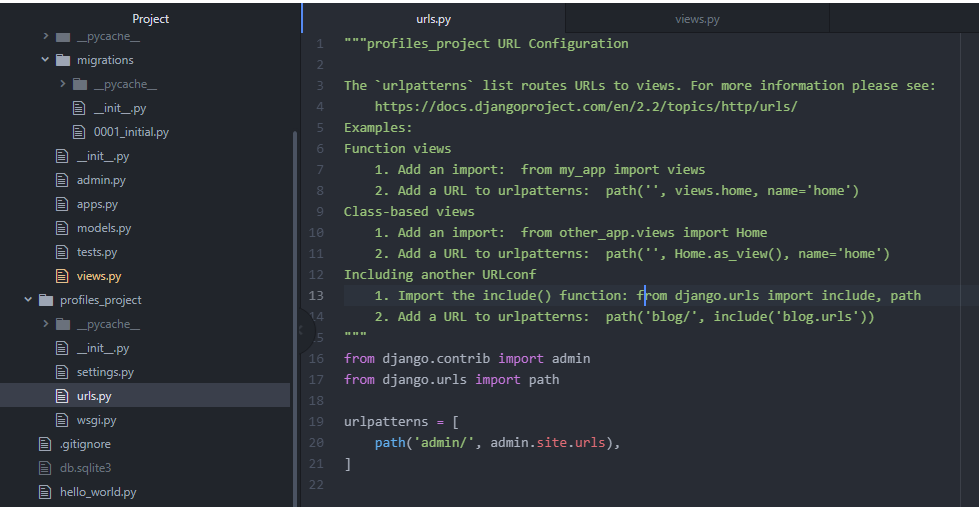
The Response needs to contain a dictionary or list which it will return when the API is called.

(converts the response object to json and in order to convert it to json it needs to be either a list or a dictionary)

**Configure View URL:**

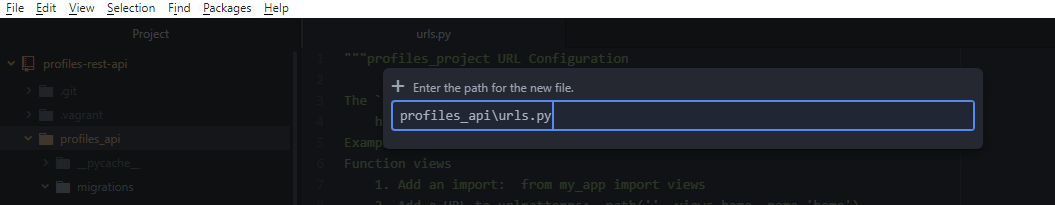
We need to map our API view with a Django URL.

1. In Atom editor, go to urls.py file



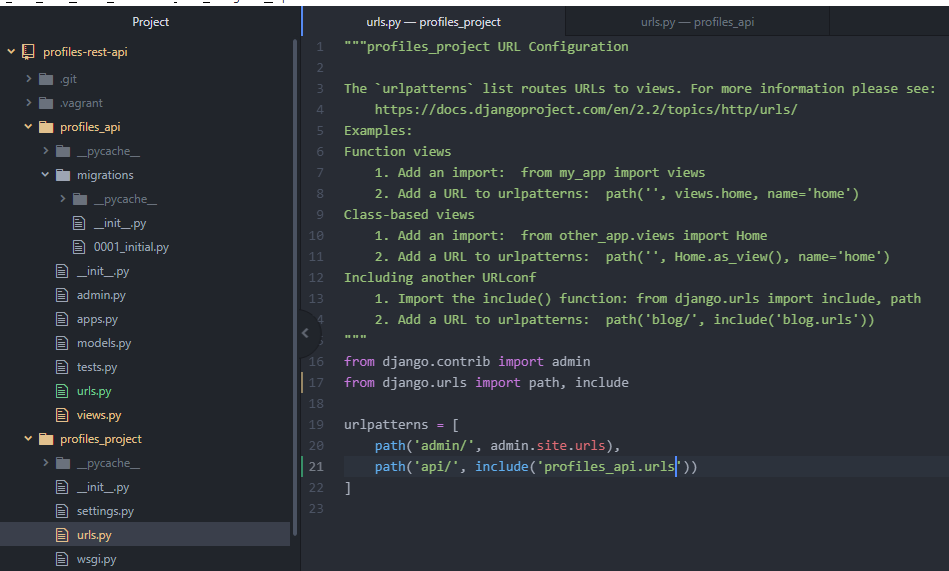
Links admin url to the admin app (created and enabled by default when we first created our project)

2. Similarly, we will create a new file called urls.py in our profiles\_api app



Here we will store the URLs of our API.

3. First we need to add our urls.py file in the project’s URL file in urlpatterns list

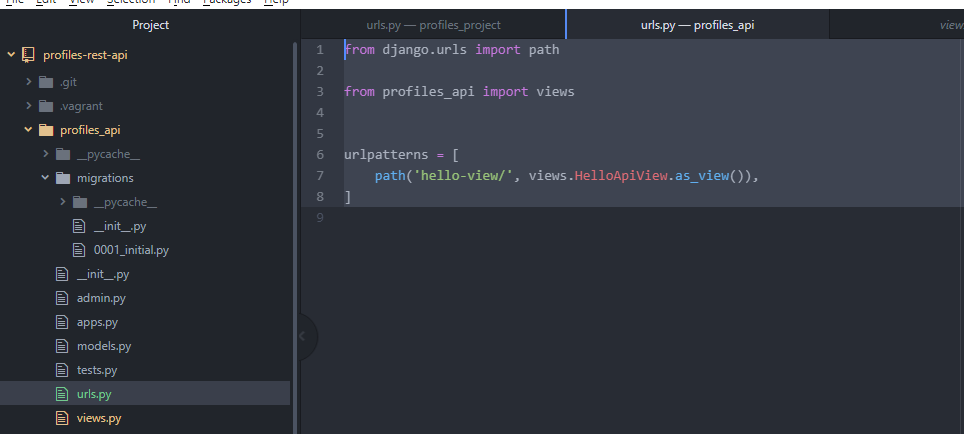


Import include – function that you can use to include URLs from other apps in the root project URLs file.

api/ - path

include(‘profiles\_api.urls’) - name of the app dot the module that contains the URLs

4. Now go to our newly created urls.py file in profiles\_api



Import path function, views module (views.py file which contains API View)

urlpatterns – new list variable. Create lists of path that map to views in our project.

hello-view/ - name of our url

views.HelloApiView – views.py file in profiles\_api which contains HelloApiView class

call as\_view() on our api class. Standard function to convert our api view class to be rendered by urls.

So DRF will call get function when a HTTP GET request is made to our URL.

**Testing our API View:**

1. Run the below commands in git bash to make the development server up and running.

vagrant up

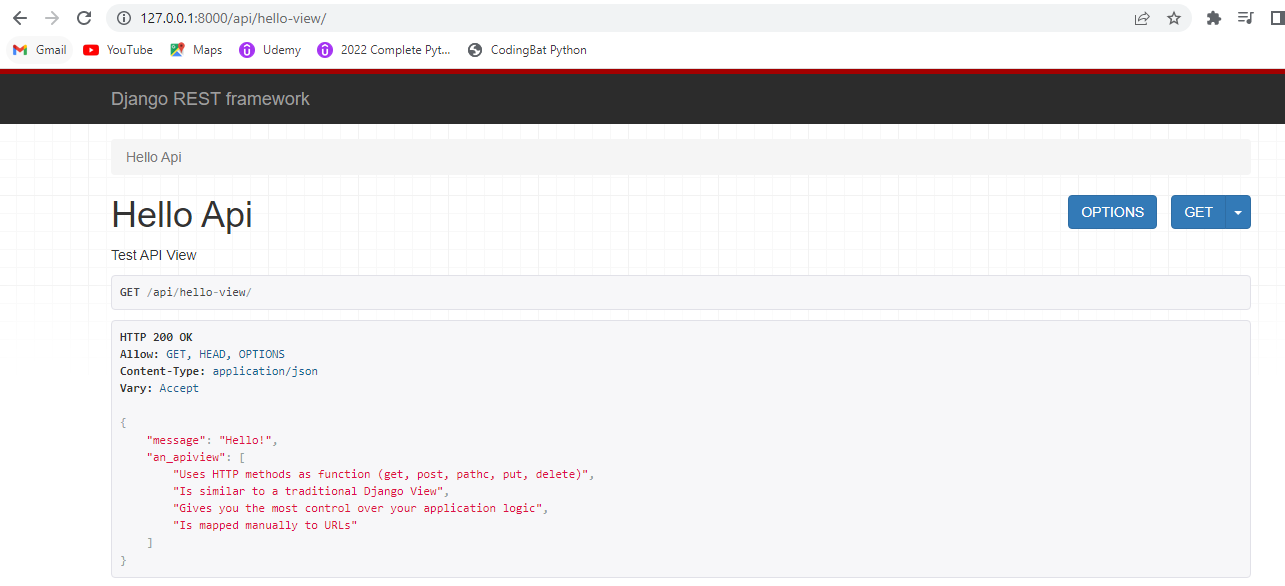
vagrant ssh

source ~/env/bin/activate

cd /vagrant

python manage.py runserver 0.0.0.0:8000

2. Go to any browser and provide address 127.0.0.1:8000/api/hello-view



3. Commit the changes in git.

