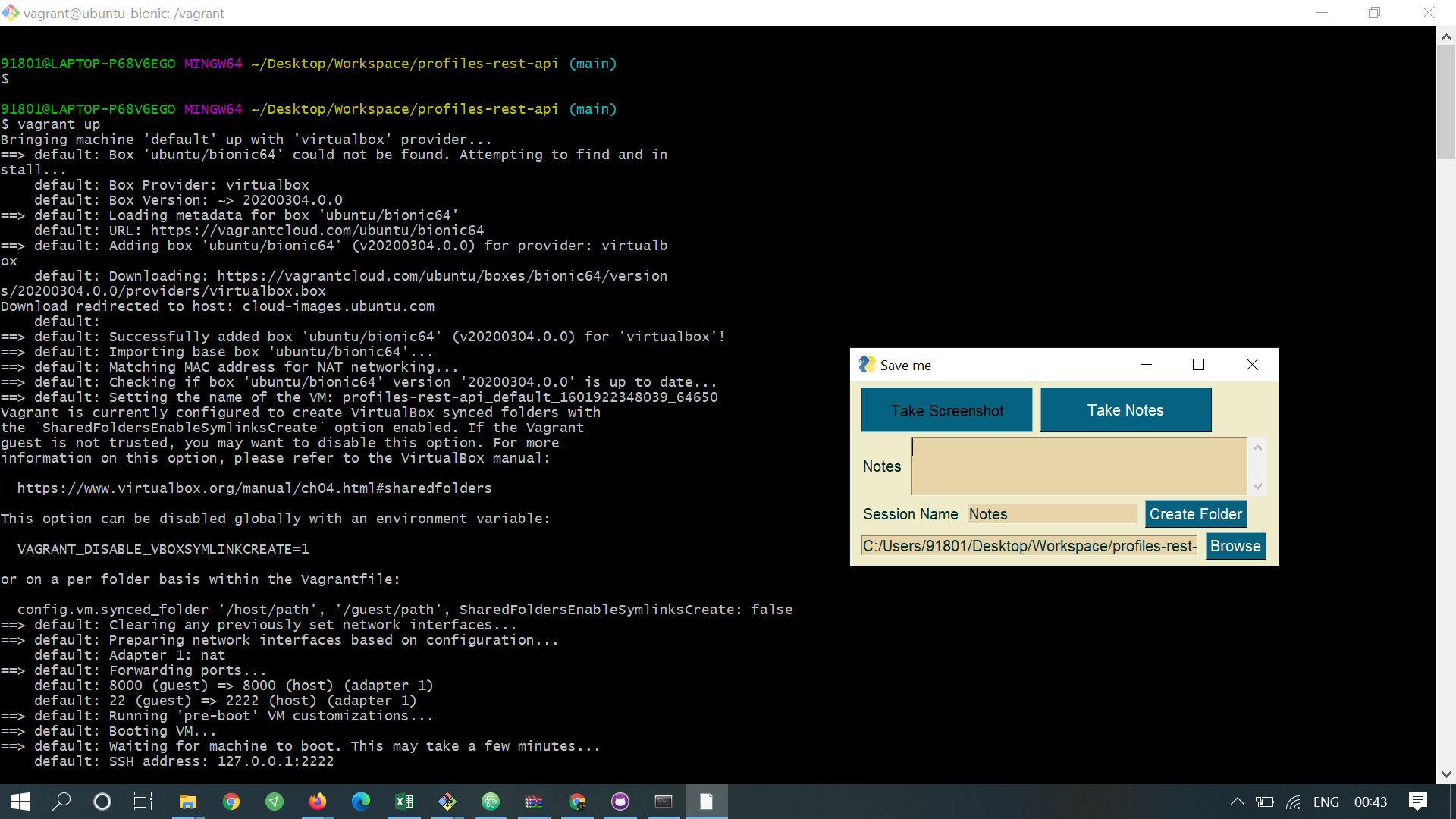
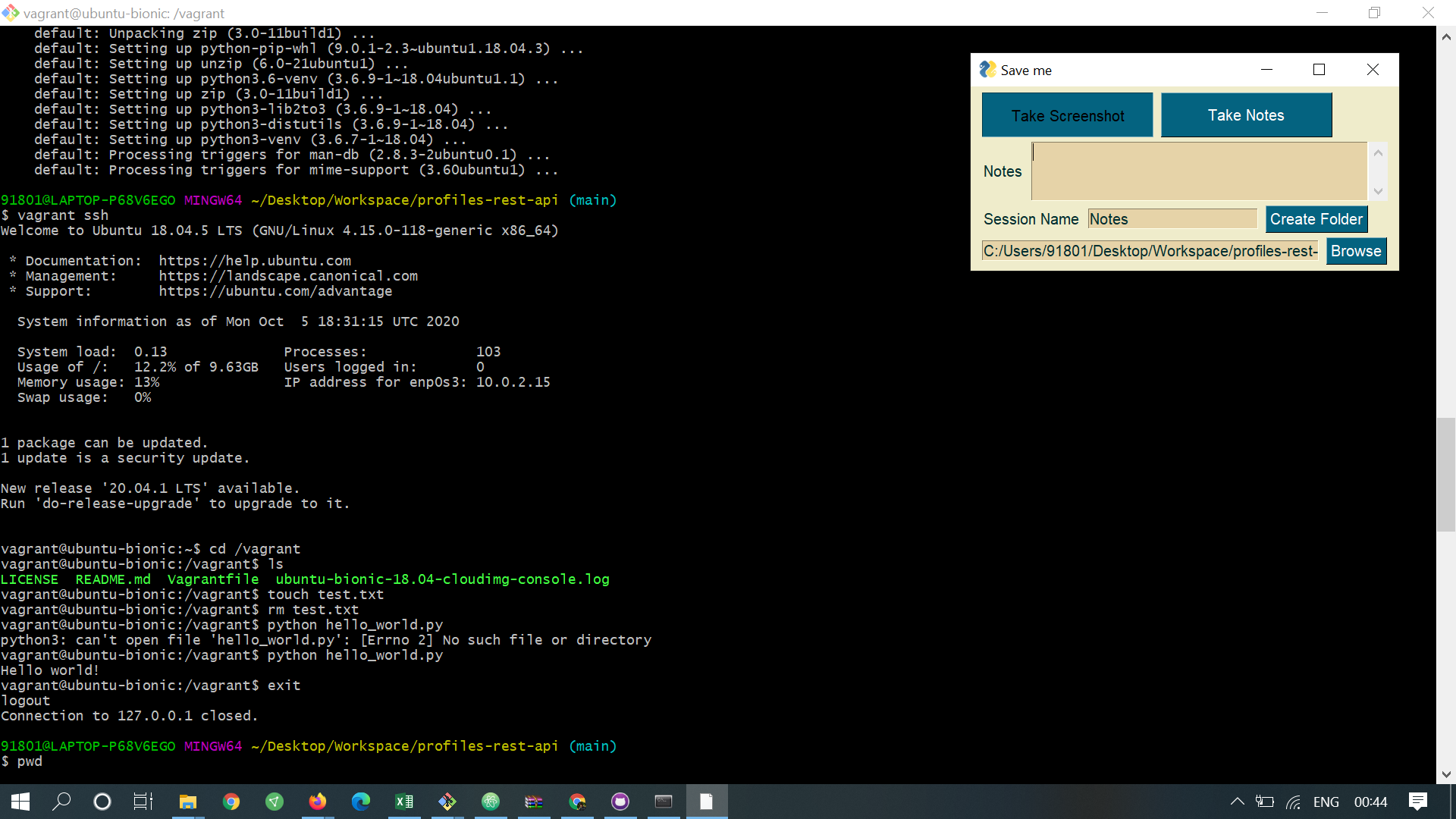
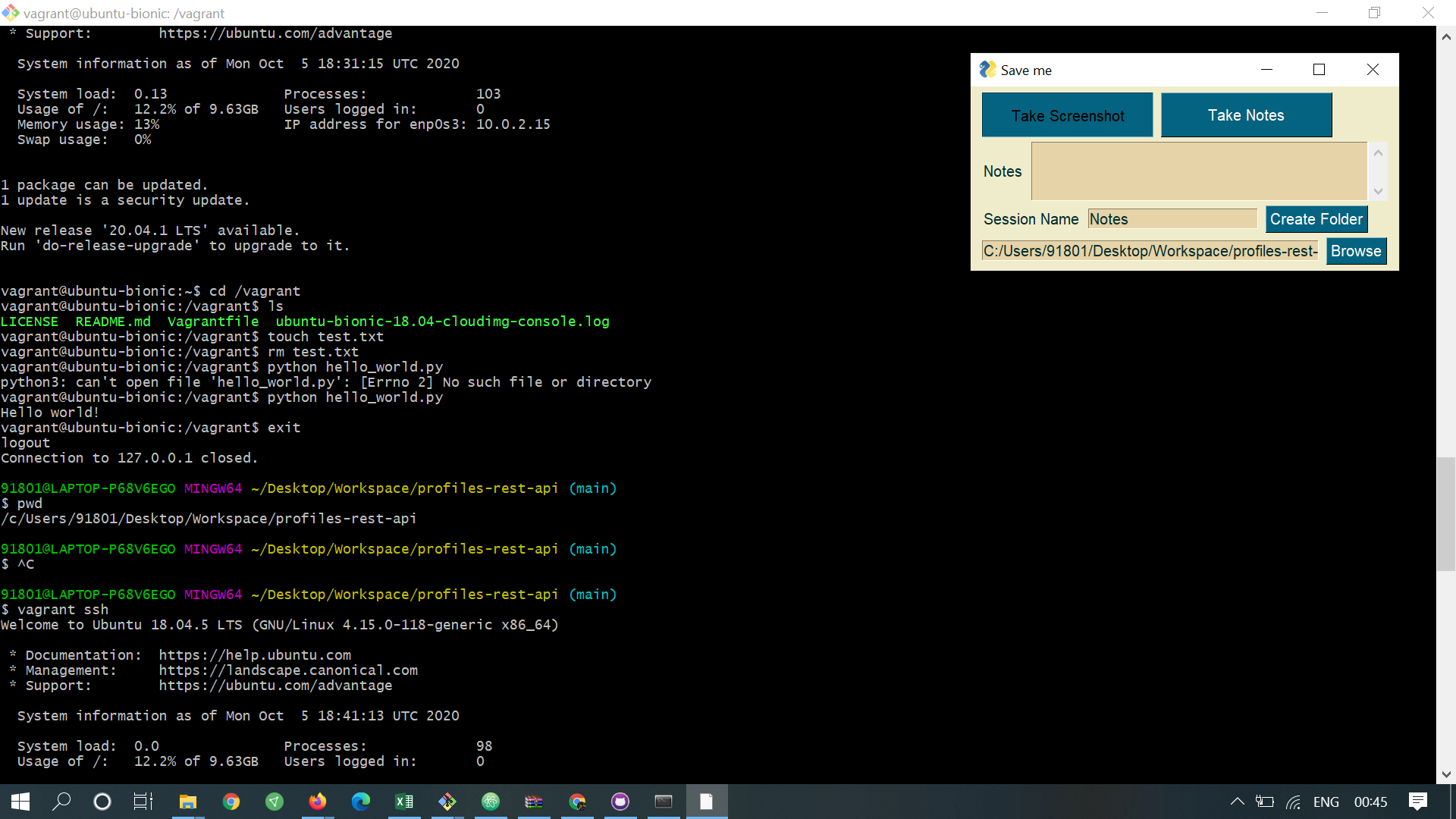
1.Activate Vagrant by typing in   
>>vagrant up



2.Connect by using vagrant by giving this command.  
>>vagrant ssh



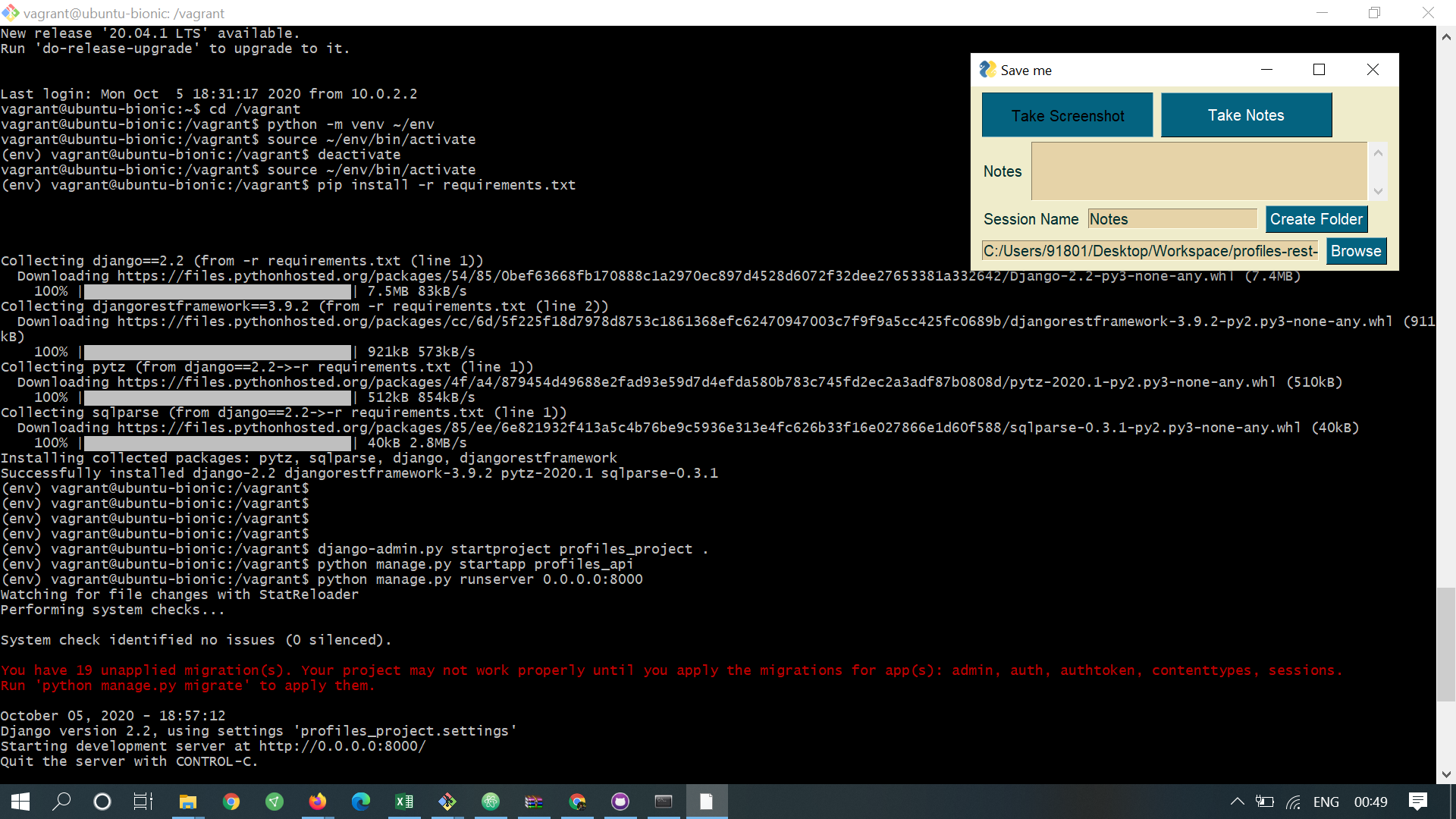
3. The /vagrant folder syncronizes with your root directory.



4. you can create a virtual environment in vagrant usingg -   
>>python ~/env/bin/activate

To deactivate - just type in >>deactivate

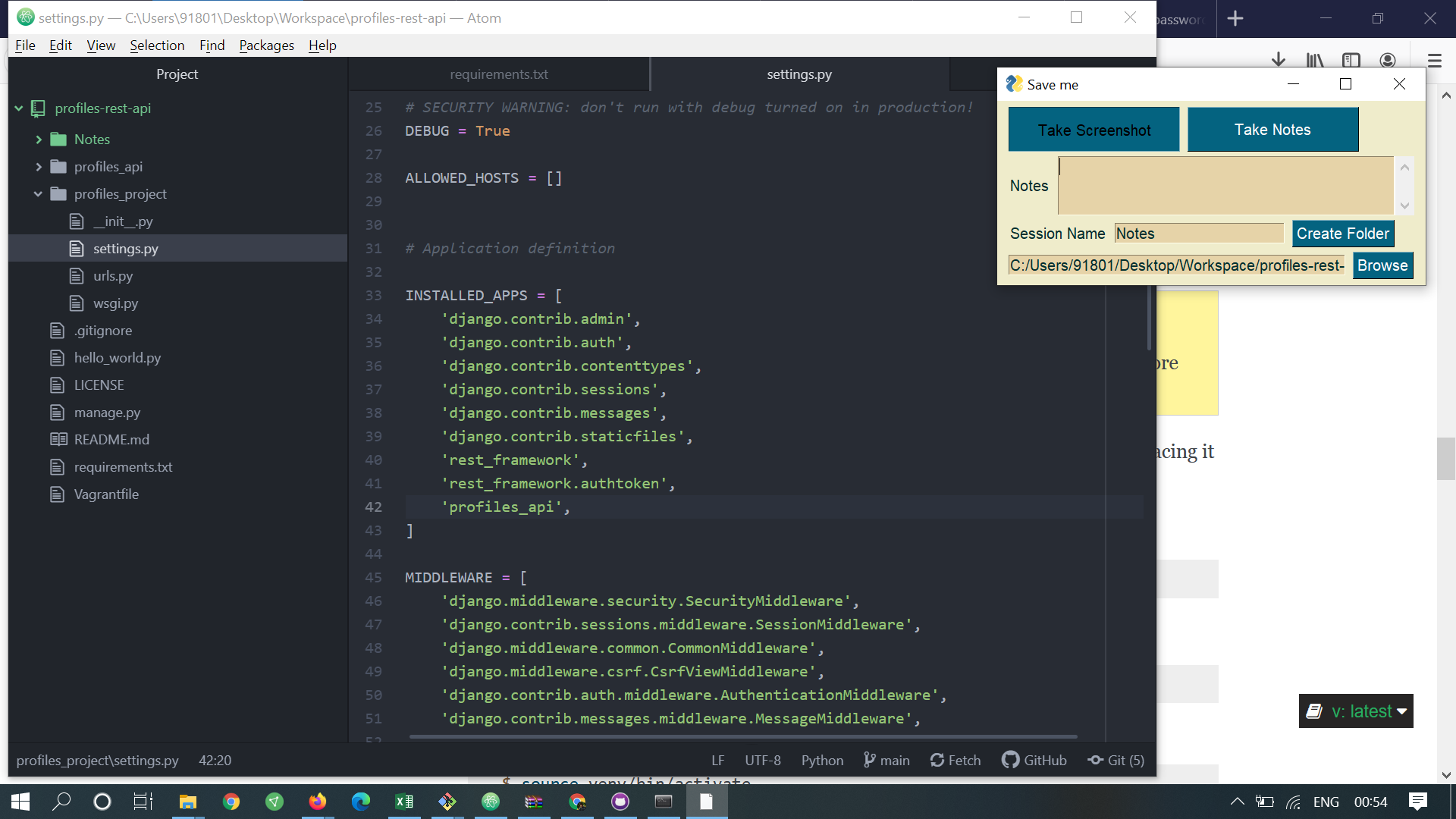
5.To install django , type -  
>>pip install -r requirements.txt



6. Create a Django project with the command (. in the below command saves the profiles\_project within the root directory)-  
>>django-admin.py startproject profiles\_project .

7. To create a Django APP, type -   
>>python manage.py startapp profiles\_api  
  
8. To run the django server (0.0.0.0:8000 is the home )-   
>>python manage.py runserver 0.0.0.0:8000

8. You need to add the APPs created in the project settings file for Django.



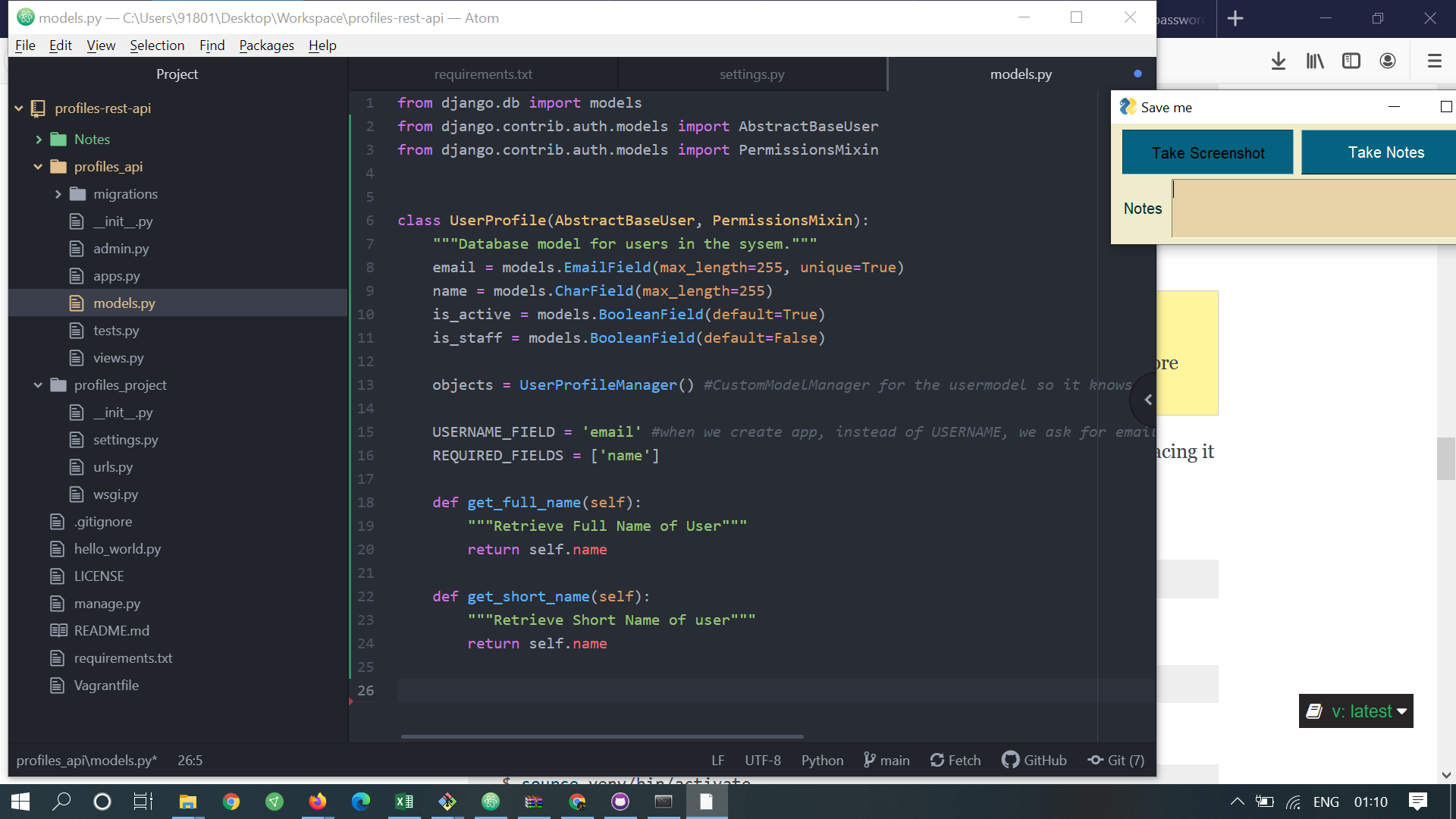
Last 4 rows in the INSTALLED\_APPS have been added for our purpose

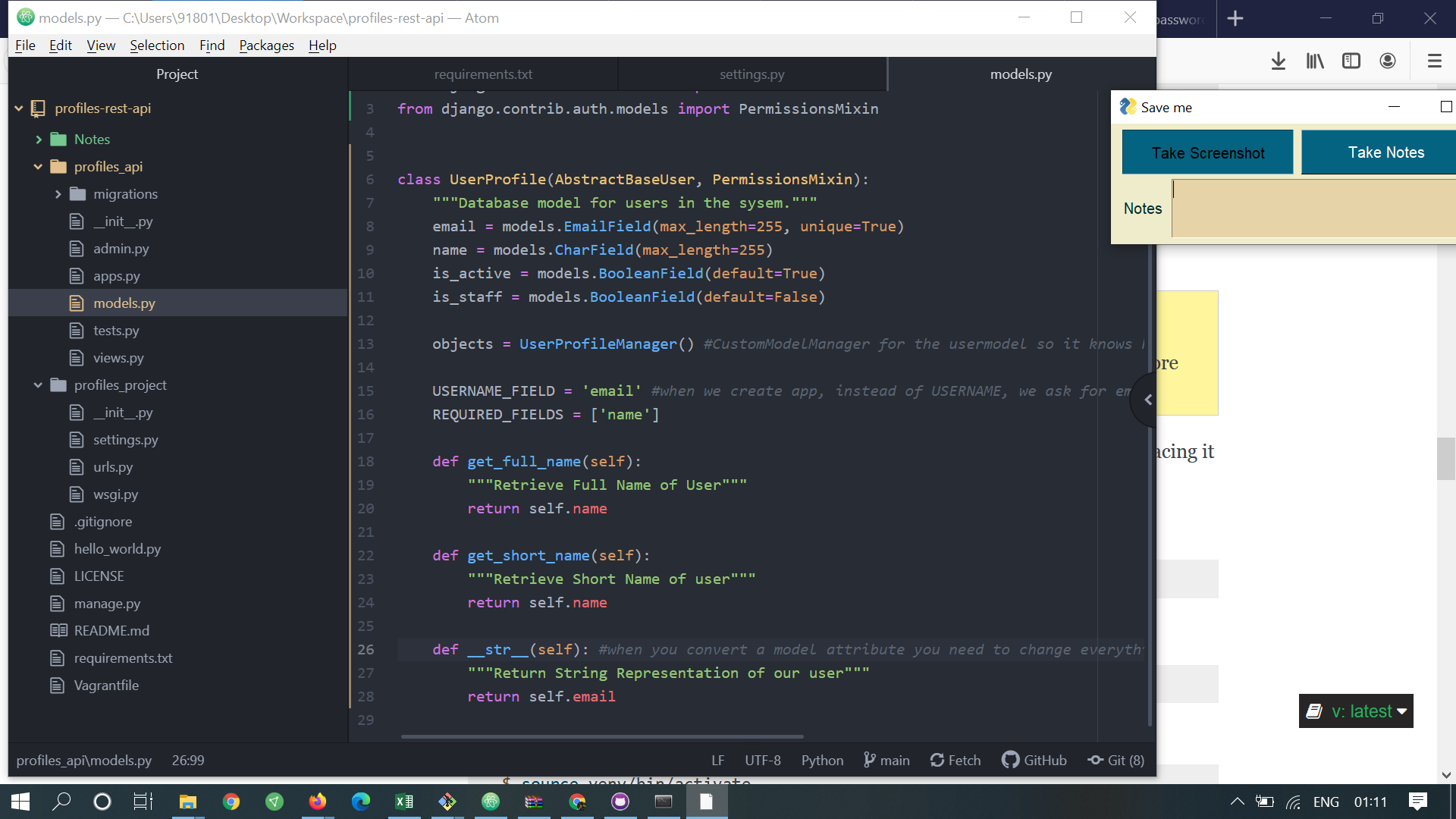
10. Usually Django has some models concept. (Need to learn more)

Django comes with a default database model, We customise this model for our purpose

We are adding our model code in the models.py in profiles\_api folder.  
  
In the code we need to import the following default packages when we overwrite the standard models.  
  
from django.contrib.auth.models import AbstractBaseUser  
from django.contrib.auth.models import PermissionsMix

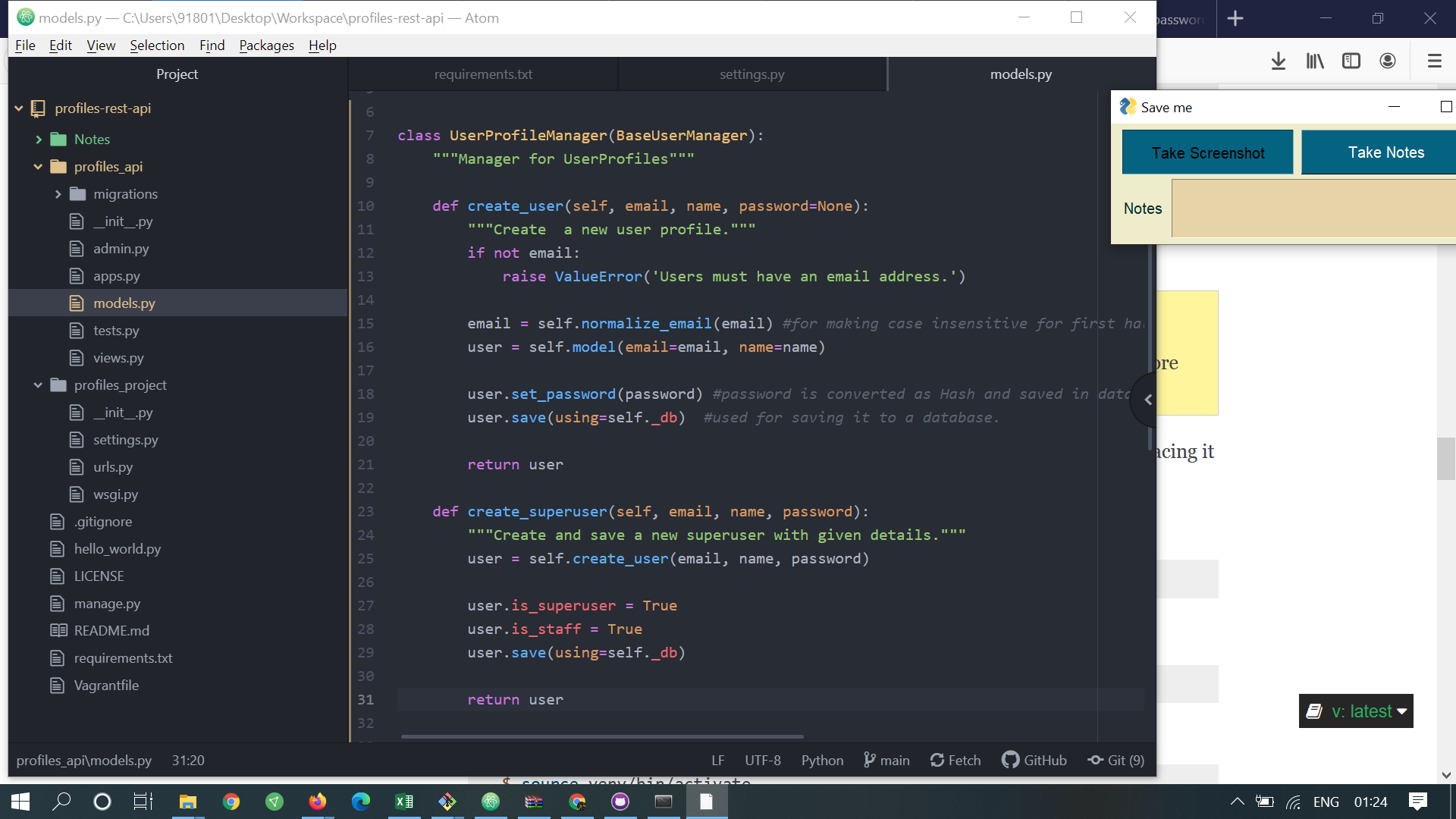
\*mixin





Django Manager has full control and manages all our apps.

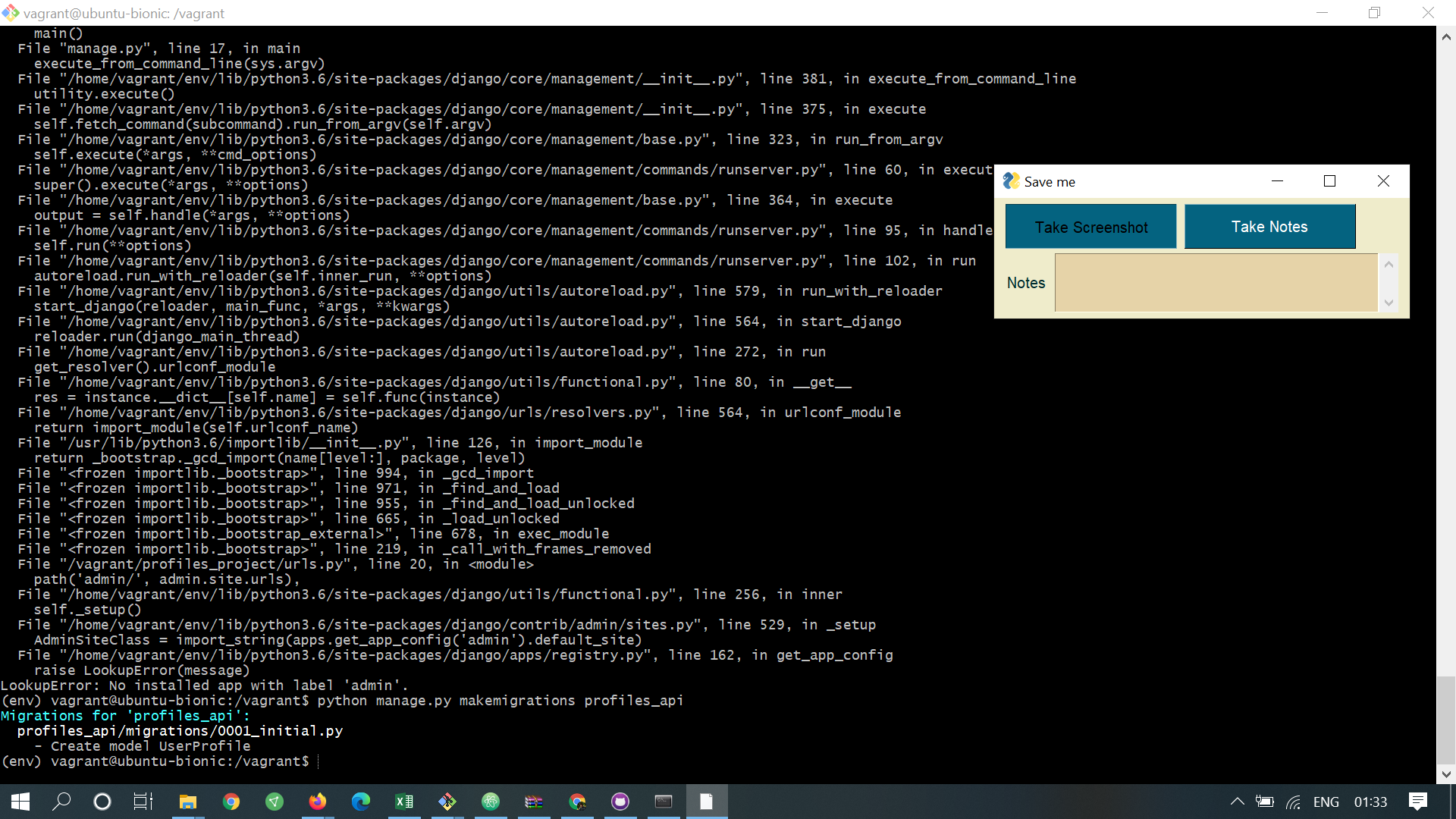
We use create SuperUser command to create a manager to customize our models.



After writing model manage code in models.oy  
  
we need to add a new line in settings.py in profile\_project folder -  
>> AUTH\_USER\_MODEL = 'profiles\_api.UserProfile'

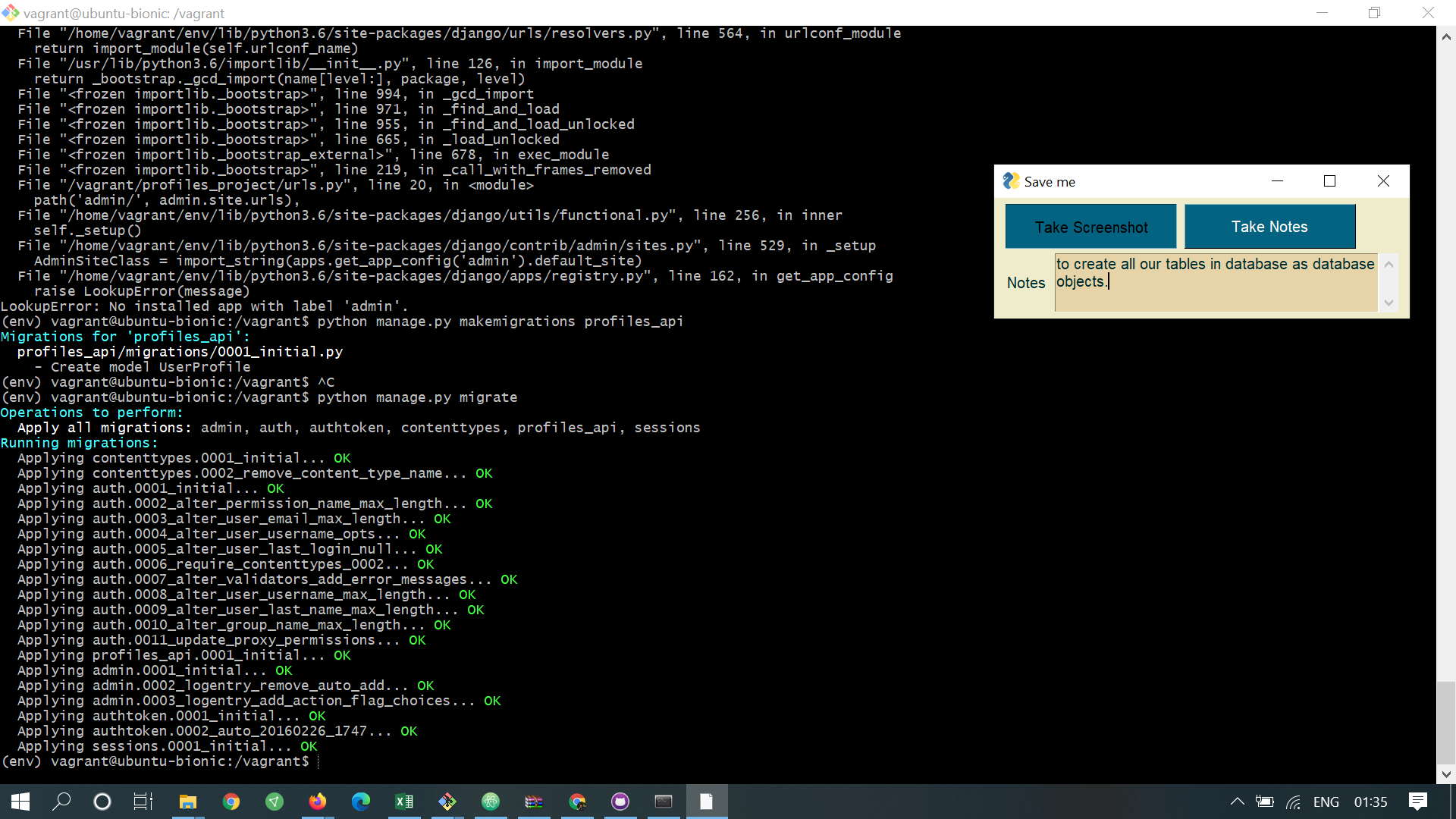
This tells our Django app to search for profiles\_api and pick the custom user model in the folder.

# We create migrations for our models.  
Django manages its database by creating Migration files.  
Migration files stores all the steps of required to make our database match our Django Models.  
  
So everytime you change or additional models, we need to create a new migration file.



python manage.py makemigrations profiles\_api

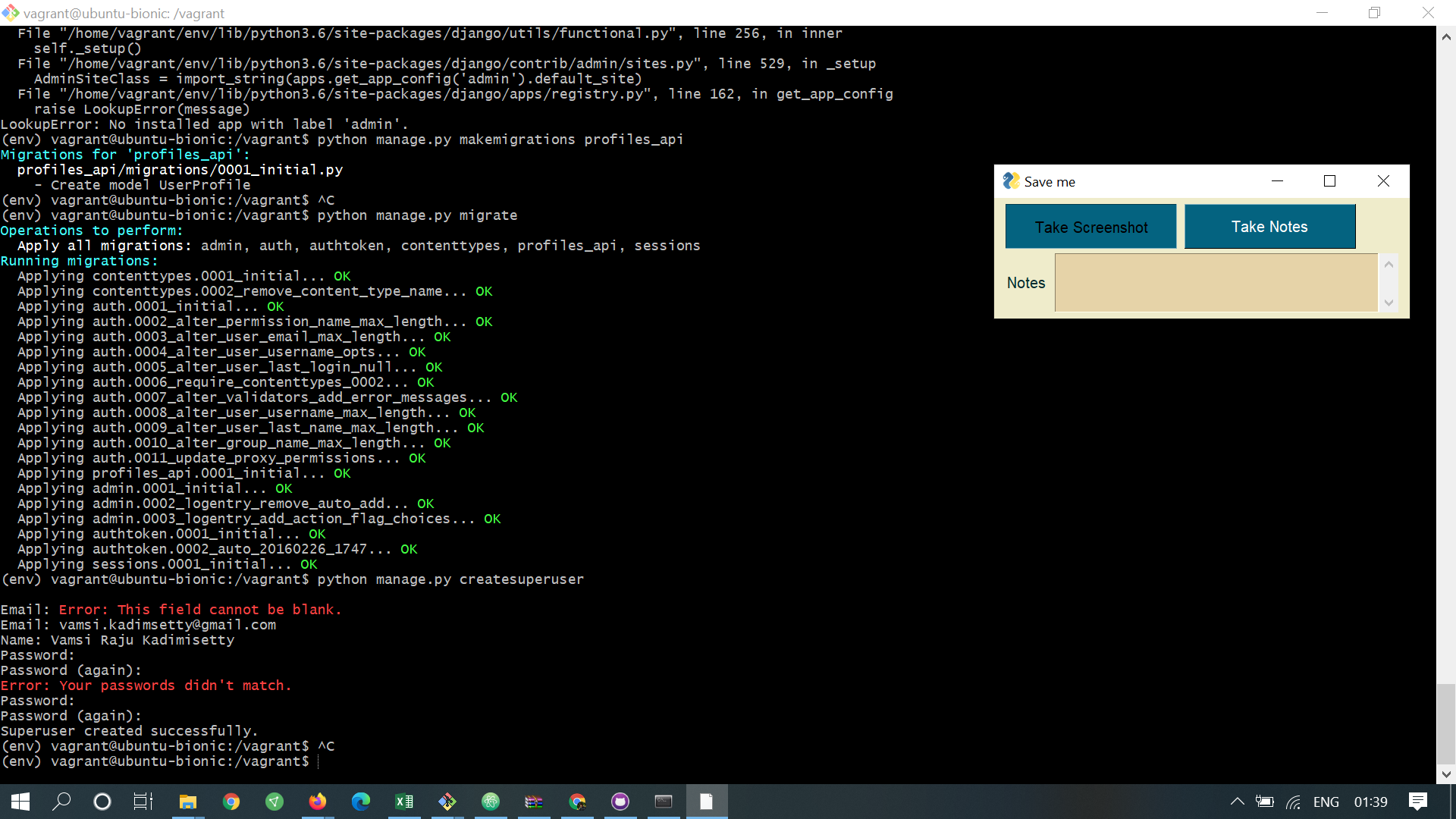
once migrations file is created, type -   
>>python manage.py migrate



to create all our tables in database as database objects.

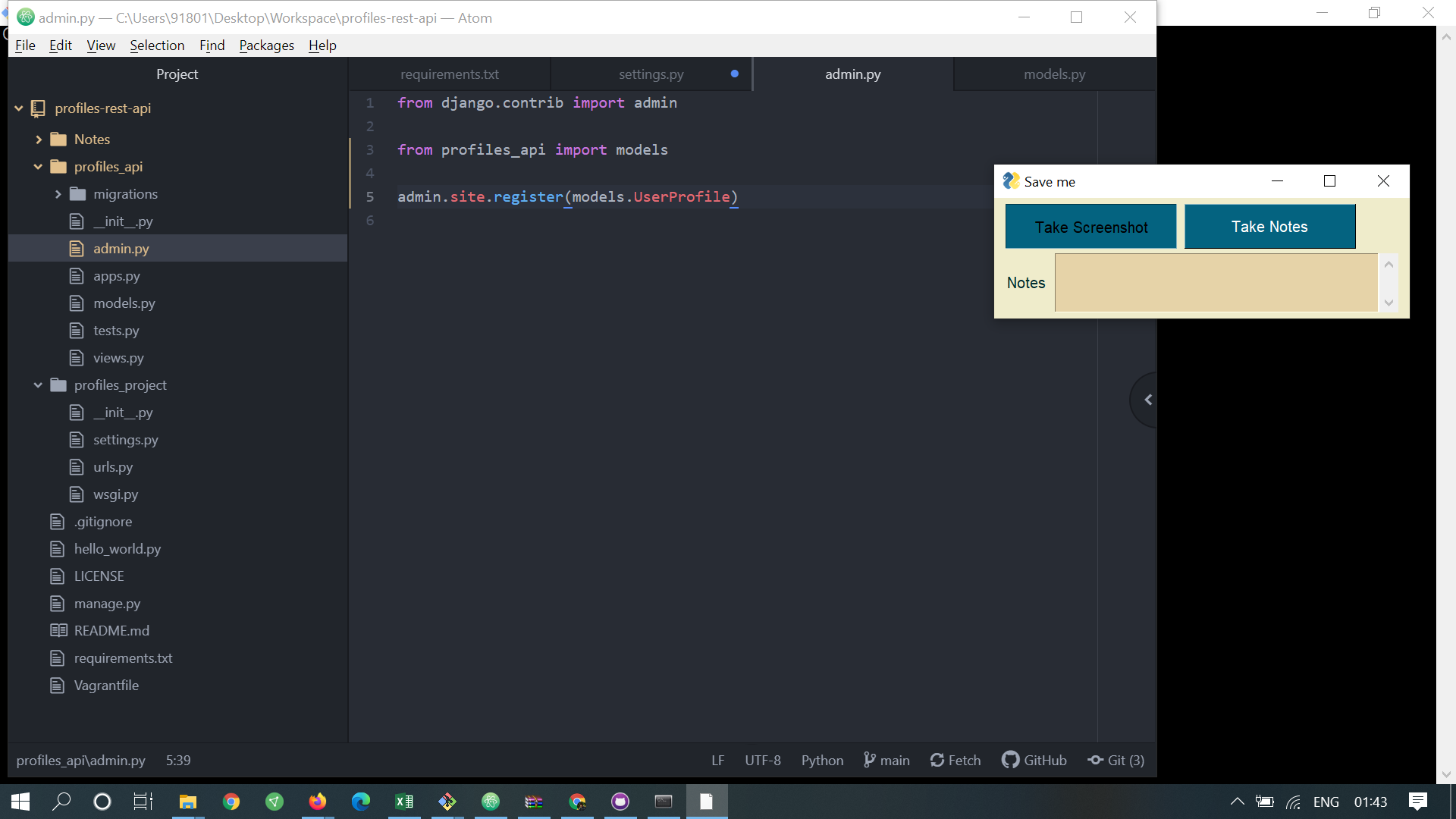
Created a SuperUser.

>> python manage.py createsuperuser



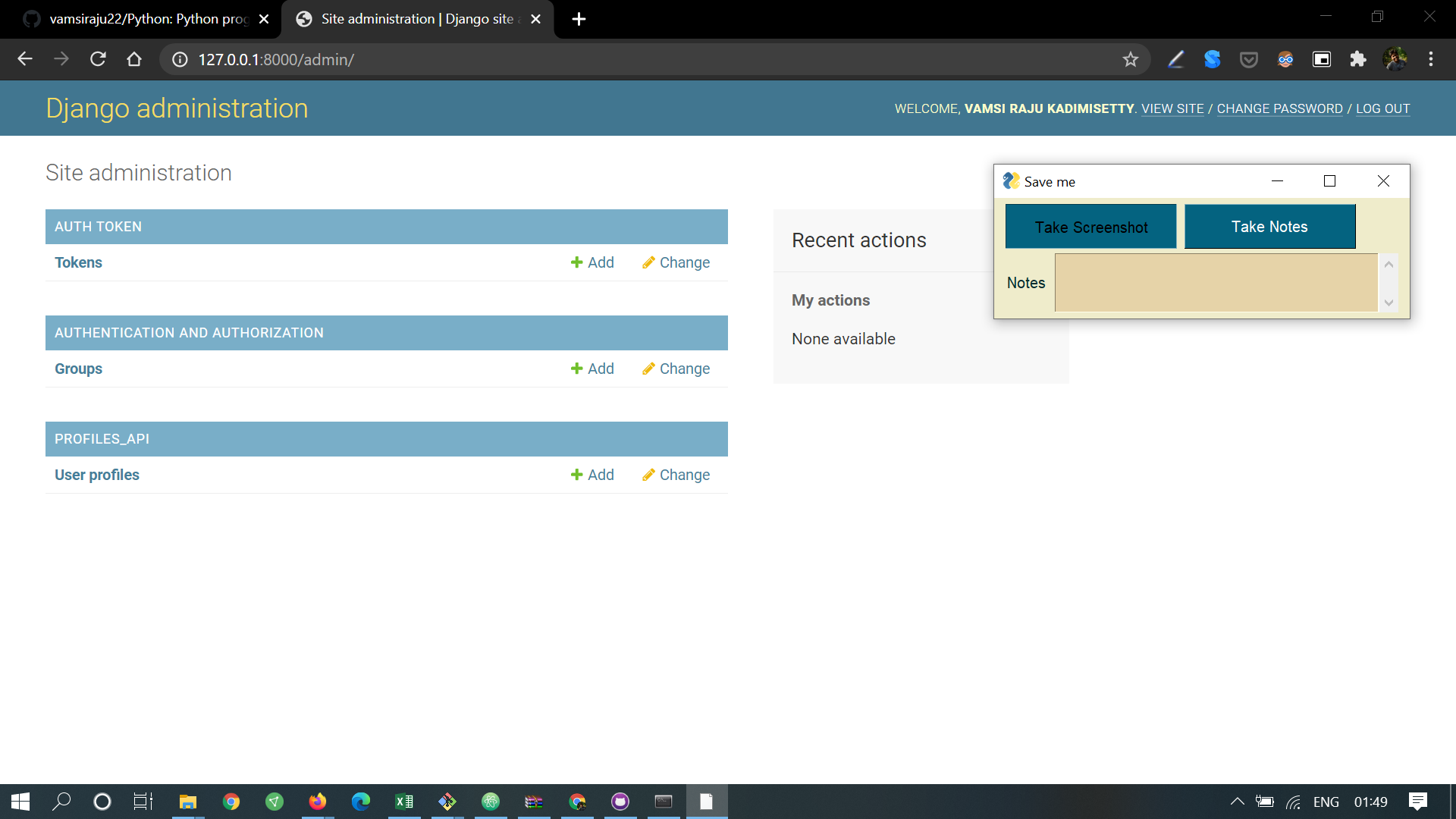
#ByDefault Django Admin is enabled for all the projects.

Any newly created models has to be registered with Django Admin, so that it knows you want to display the model in the admin interface.



>>admin.site.register(models.UserProfile) in admin,py

To run the Django Server,  
>> python manage.py runserver 0.0.0.0:8000  
and type the url in the browser - > 127.0.0.1:8000  
  
127.0.0.1:8000/admin -> takes you to admin page.



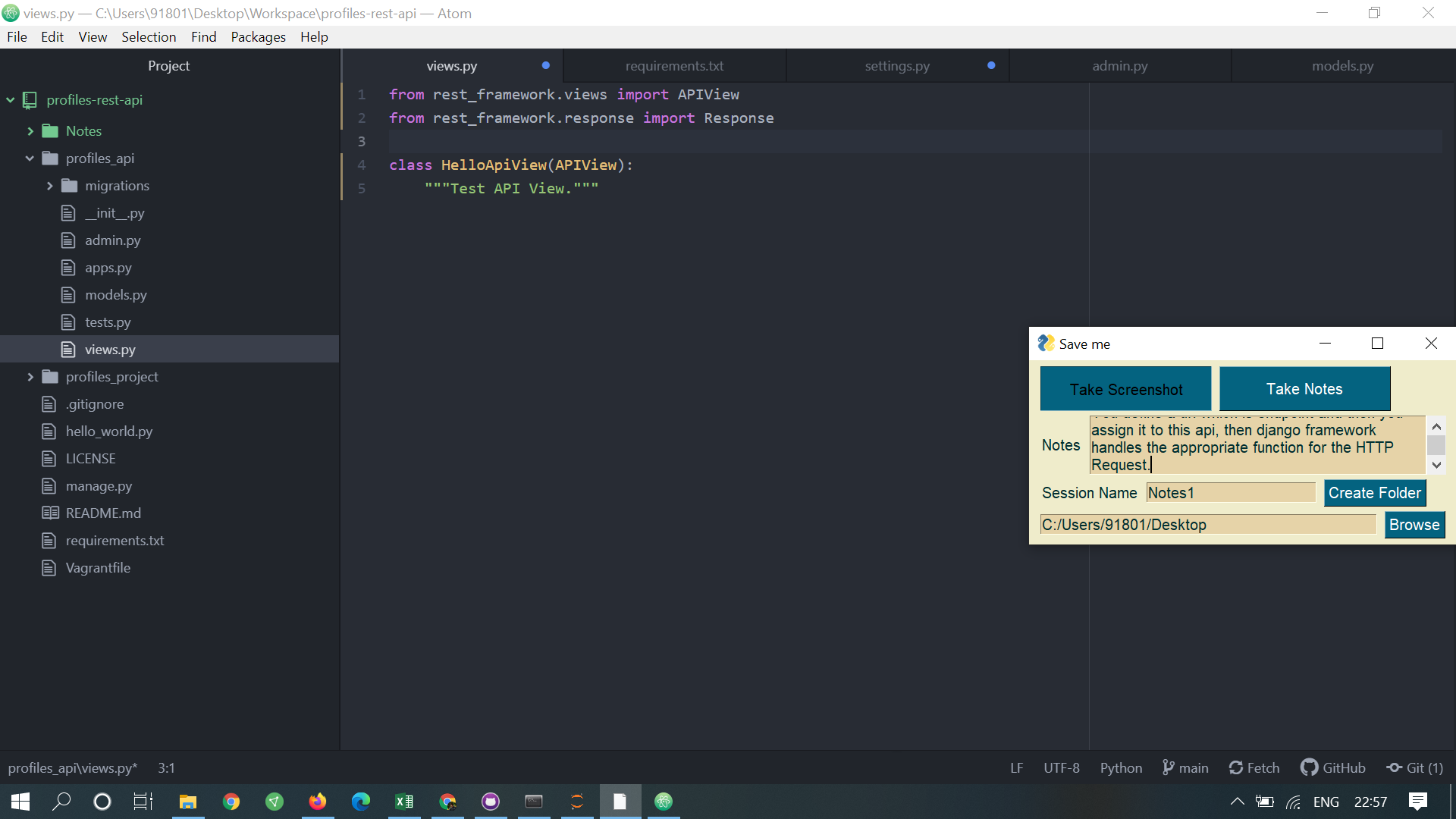
AUTH TOKEN AND   
AUTHENTICATION AND AUTHORIZATION by default.  
 Our App PROFILES\_API is displayed.

User Profiles Model that we created is displayed in our APP.

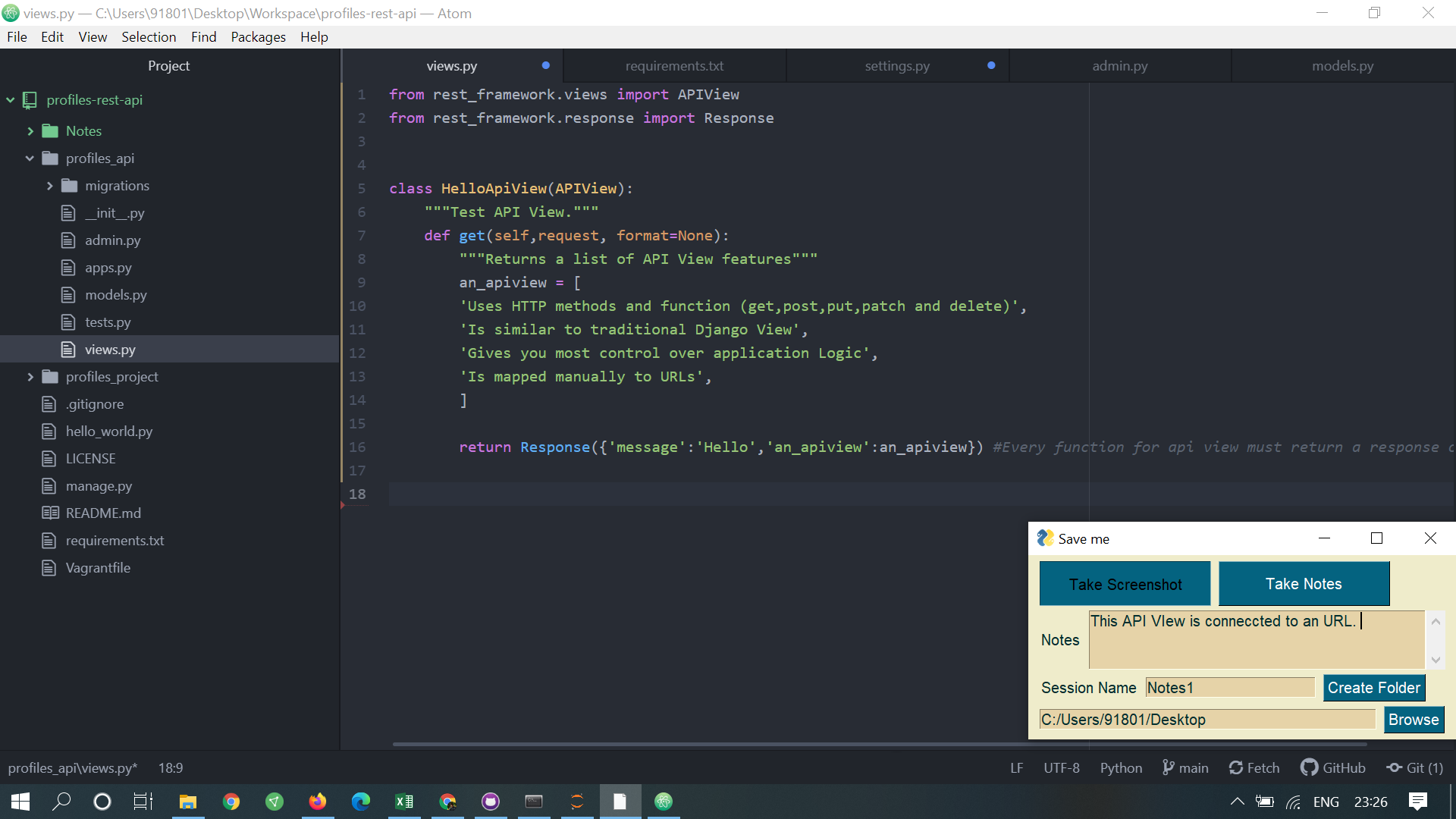
APIS

APIVIEW - basic type. enables to describe logic for our api. Allows us to define functions that match the standard http methods like GET, POST,PUT,PATCH,DELETE

used for implementing complex logic, calling other apis, usage is personal preference.   
eg - whenever you need full control of logic, processing files and rendering a synchronous response. Or when you are calling other APIs/services. or Accessing local files.

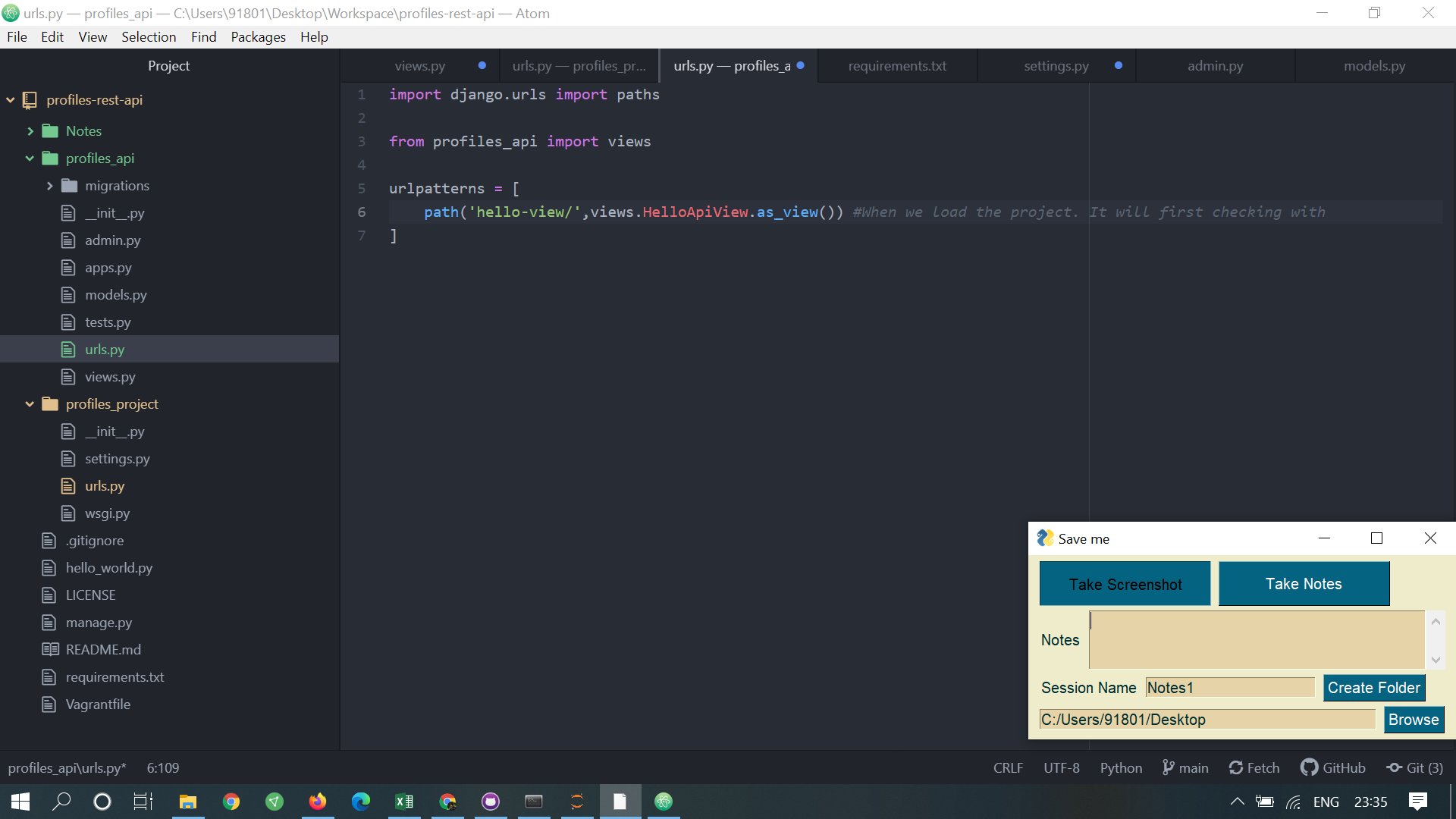


You create a class for API View,   
You define a url which is endpoint and then you assign it to this api, then django framework handles the appropriate function for the HTTP Request.



This API VIew is conneccted to an URL.

1.We add a new url path -   
path('api/',include('profiles\_api.urls')) in Main project folder urls.py file.  
  
2. We create a urls.py file in the profiles\_api folder and then add a urls.py file in this folder



We add the value for urlspatterns= [path('hello-view/',views.HelloApiView.as\_view())]  
  
This means that the Django will look for urls from the main project folder and then navigate to the urls in the api folders. Then to a url path, we pass our View Class objects as parameters.

Then, follow the previous steps in activating the python environment in virtual machine and start the server.  
  
>>vagrant ssh  
>>cd /vagrant  
>>source ~/env/bin/activate  
>>python manage.py runserver 0.0.0.0:8000  
  
Then open browser and enter the path -  
http://127.0.0.1:8000/api/hello-view/

