**DJANGO WEB APP**

You need to have install python 3.6.5 at least

Open anaconda prompt and type: pip install django

Now you need to upgrade pip: python -m pip install –upgrade pip

Check Django version: python -m Django –version

Start Django project: Django-admin startproject ‘name’

A single project can contain multiple apps. You can take a single app and take it to multiple websites or projects.

You can create your app within the anaconda prompt like follows: python manage.py startapp ‘name’

You can run Django project like this: python manage.py runserver

You can create your views in the View.py file within the app created

For routing your project you will need to take a peek into app.py within your frontend and paste app config class title to the settings.py of you project backend within INSTALLED\_APPS

Django uses a template engine that allows us to write code within the template

You can create a super user (admin page) like: python manage.py createsuperuser (you need to first create the database for the project in order for this command to work) (we need to run some migrations command to apply changes to our database)

The first migration will create the database and add a bunch of default tables for us. The auth\_user is gonna get created, and so createsuperuser will be viable to run.

To do this run this command: python manage.py makemigrations (detects the changes and prepares Django to update the database).

In order to apply the migrations: python manage.py migrate (auth\_user table should now exist)

Run: python manage.py createsuperuser (and then input your user, email and password)

**DATABASE**

We will use and sqLite for development and posgres database for production.

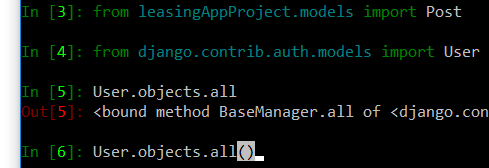
Within models.py you can create your models to map them as tables with Django orm

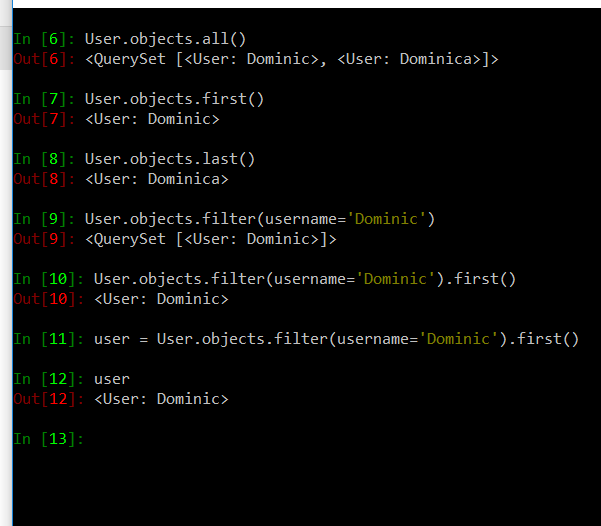
Once we have made some changes to our models.py we need to update the database with migrations: python manage.py makemigrations

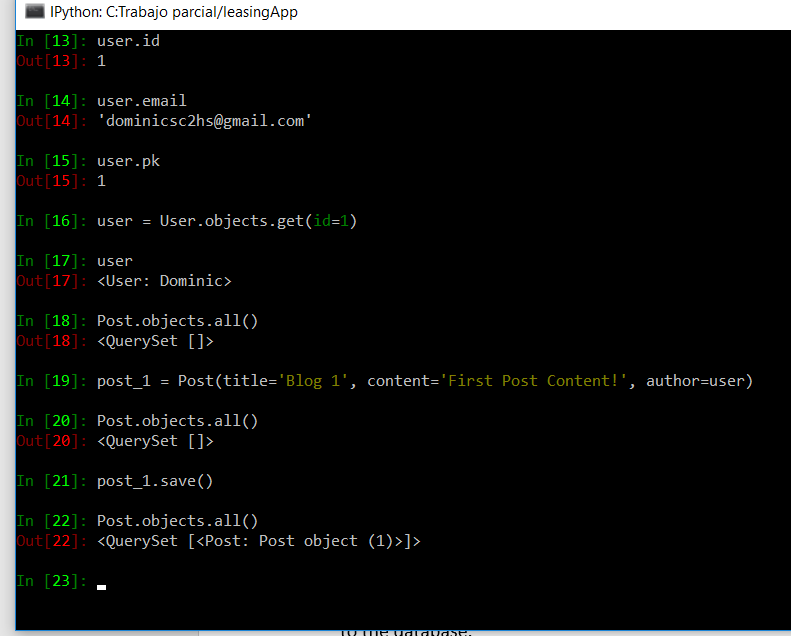
If you wanna take a look at the sql code that created the table run: python manage.py sqlmigrate leasingAppProject 0001 (this will print the sql code)

Now lets run the migrate command so it runs the migration and this effects will take place: python manage.py migrate

If we wanna query the database run: python manage.py shell (this will allow us to work with the models)

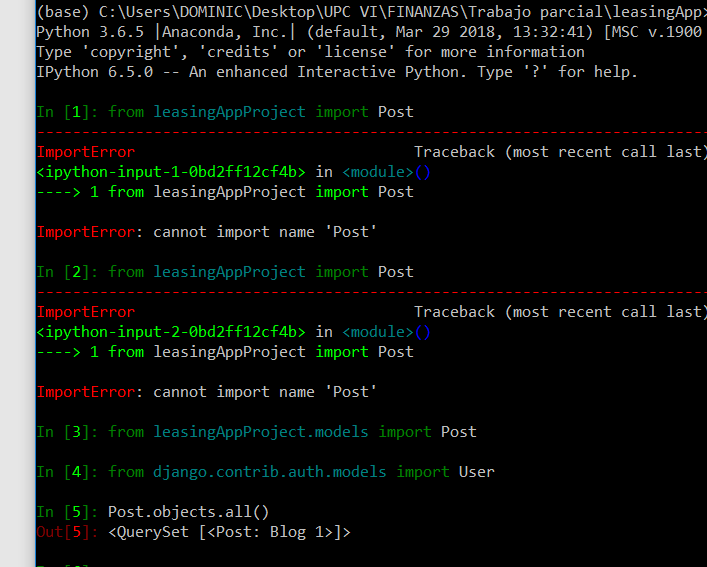




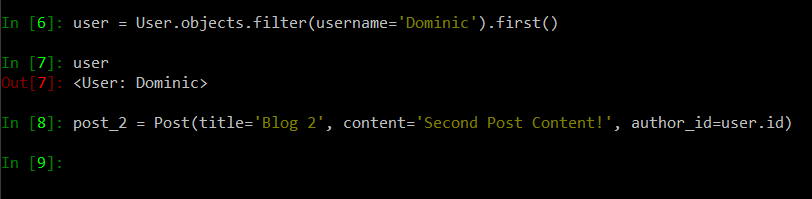


Now you can query the users and see details

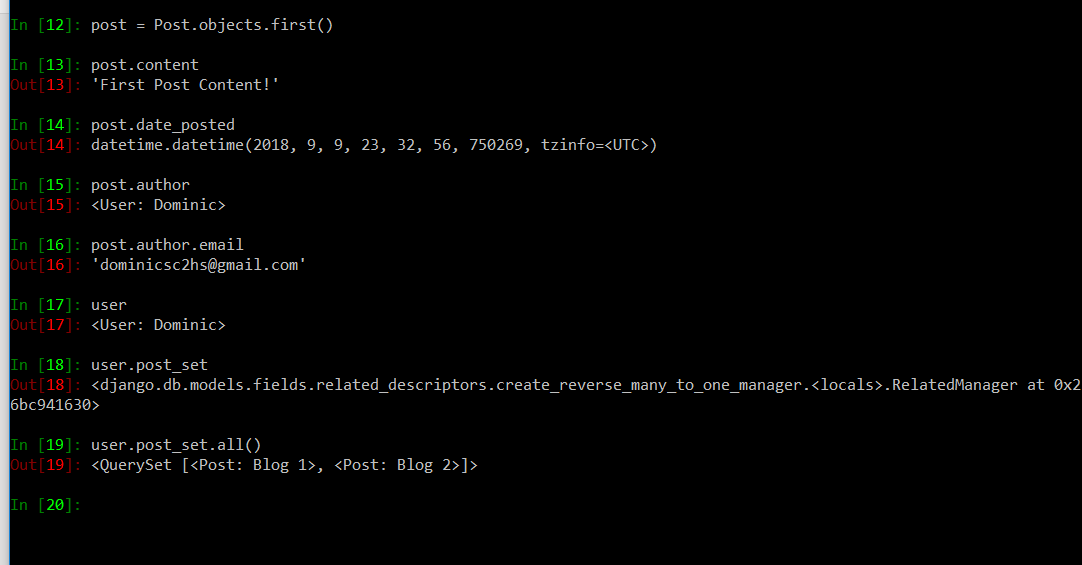
In line 19 we are creating a post for the first user and in line 21 we are saving that change to the database.



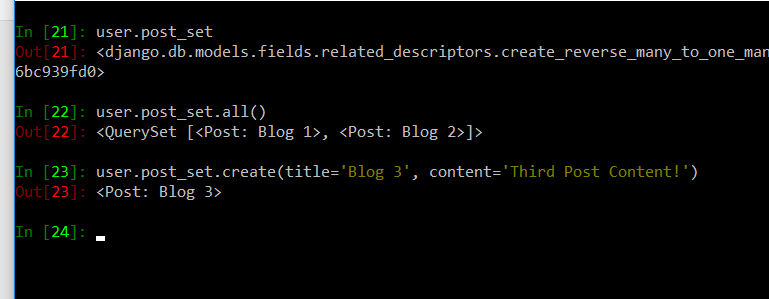
If we wanna have a more descriptive title to out post we need to set a function \_\_str\_\_ within out Post model and return a desired value. Then rerun the shell command and reimport the models and follow the image above.



We need also to get the user again in order to post another post, and you can set the post to be equal to a user.id like the image above. Save it



We can get access to all of the posts made by an user with a Django buil-in function shortcut like above.

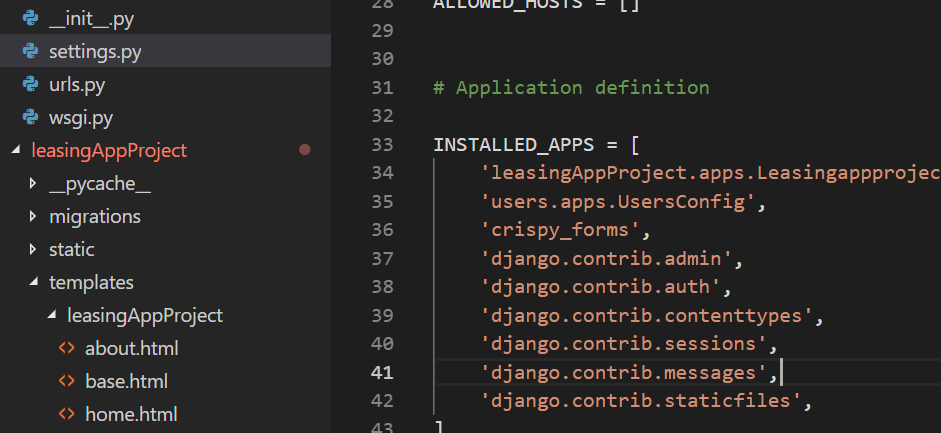


We can also create another post for this user with the help of post\_set like above.

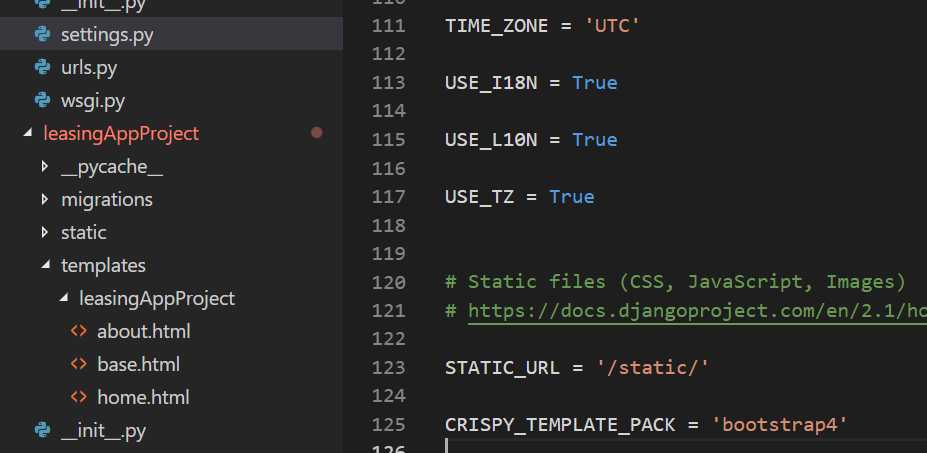
Now we can replace the dummy data from our View.py

If you wanna update the admin entries you should register your models within the admin.py which is in your frontend. We register our models, so they show in the admin page, and we can crud them.

Crispy forms will allow us to put simple tags in our template that will style our forms in a bootstrap fashion. Installation: pip install django-crispy-forms



Then include it in your setting.py



Then tell it you wanna use bootstrap 4

Django has a built in for getting the current user logged in

Pillow is a library for working with images in Django: pip install Pillow. It can also help resizing images for performance improvement.