How To Create A Django Project:

(if in new environment)

Create and navigate to a folder named my\_environments

Create your environment:

------------------------------------------------------------------

| Mac/Linux: | python3 -m venv djangoPy3Env

-------------+----------------------------------------------------

| Windows (command prompt): | python -m venv djangoPy3Env

>------------------------------------------------------------------

**Activate your environment:**

------------------------------------------------------------------

| Mac/Linux: | source djangoPy3Env/bin/activate

------------------------------------------------------------------

| Windows (command prompt): | call djangoPy3Env\Scripts\activate

------------------------------------------------------------------

| Windows (git bash) : | source djangoPy3Env/Scripts/activate

------------------------------------------------------------------

**Install Django:**

(djangoPy3Env) Windows/Mac:| pip install Django

cd .. back out to main level and start a project

(if working in a previous environment)

**To Create a new project:**

>>> django-admin startproject your\_project\_name\_here

**To add an app to your project:**

>>> cd your\_project\_name\_here

your\_project\_name\_here> python manage.py startapp your\_app\_name\_here

**to run your project:**

your\_project\_name\_here> python manage.py runserver

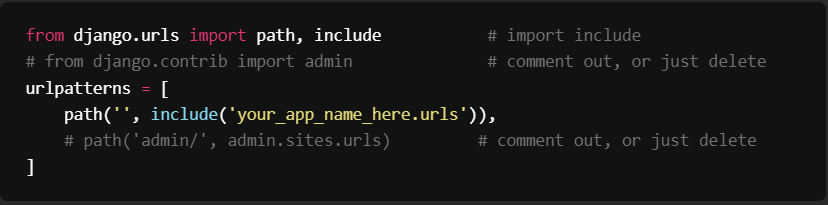
Go to Settings.py and add in your app name to the installed apps list

your\_project\_name\_here/your\_project\_name\_here/settings.py



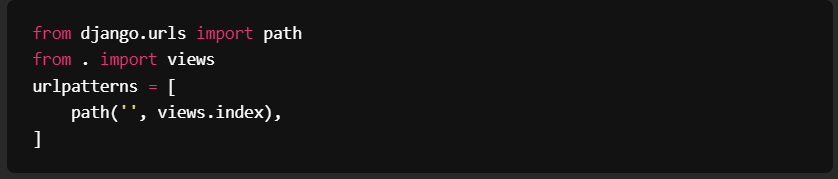
Go to urls.py and create a route to associate the app to the project

your\_project\_name\_here/your\_project\_name\_here/urls.py



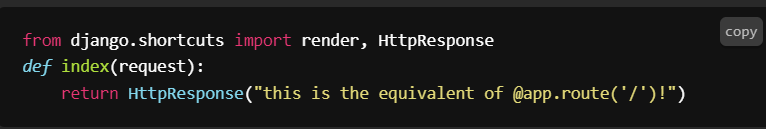
Next cd into your app and create a urls.py file and start with the following code

your\_project\_name\_here/your\_app\_name\_here/urls.py



Next add a function called index into the apps’ views.py file

your\_project\_name\_here/your\_app\_name\_here/views.py



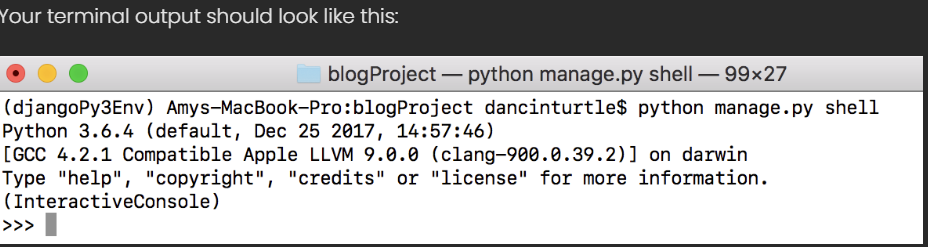
Run your app

>>> python manage.py runserver

Using Django Shell

To use the shell, we'll run the following command in our terminal from our project's root directory (where our manage.py file is located):

>>> python manage.py shell



**Import the table you want to work with** (your must always do this any time you open the shell)

>>> from your\_app\_name\_here.models import your\_model\_name

**OR** (warning – importing all can potentially cause conflicts, using the specific table is better practice)

>>> from your\_app\_name\_here.models import \*

**Overview of Commands**

**Creating a new record**

ClassName.objects.**create**(field1="value for field1", field2="value for field2", etc.)

**Reading existing records**

Methods that return a single instance of a class

ClassName.objects.**first**() - gets the first record in the table

ClassName.objects.**last**() - gets the last record in the table

ClassName.objects.**get**(id=1) - gets the record in the table with the specified id

**this method will throw an error unless only and exactly one record matches the query**

**Methods that return a list of instances of a class**

ClassName.objects.**all**() - gets all the records in the table

ClassName.objects.**filter**(field1="value for field1", etc.) - gets any records matching the query provided

ClassName.objects.**exclude**(field1="value for field1", etc.) - gets any records not matching the query provided

**Updating an existing record**

c = ClassName.objects.**get**(id=1)  
c.field\_name = "some new value for field\_name"  
c.**save**()

**Deleting an existing record**

c = ClassName.objects.**get**(id=1)  
c.**delete**()

**Other helpful methods**

**Displaying records**

ClassName.objects.get(id=1).**\_\_dict\_\_** - shows all the values of a single record as a dictionary

ClassName.objects.all().**values**() - shows all the values of a QuerySet (i.e. multiple instances)

**Ordering records**

ClassName.objects.all().**order\_by**("field\_name") - orders by field provided, ascending

ClassName.objects.all().**order\_by**("**-**field\_name") - orders by field provided, descending

UNREPAIRABLE DATABASE

If you have tried everything and hit a brick wall with your database, or corrupted your data beyond repair, as a last-ditch effort you can delete your database and start over from scratch. To do this, you will need to delete the following files/directories:

* db.sqlite3
* your\_app\_name/migrations
* your\_app\_name/\_\_pycache\_\_

After you delete these, you will need to re-run your makemigrations and migrate commands. When you make your migrations, you will need to include the app name:

python manage.py makemigrations your\_app\_name

python manage.py migrate

**IMPORTANT:**There is a reason that this is so difficult to do! Databases should as a rule never be deleted after creation, as this interferes with the reliability of your trusted data source. We typically want to store any changes ever made to a database so that we can audit it at any time, so this should only be used in an emergency.