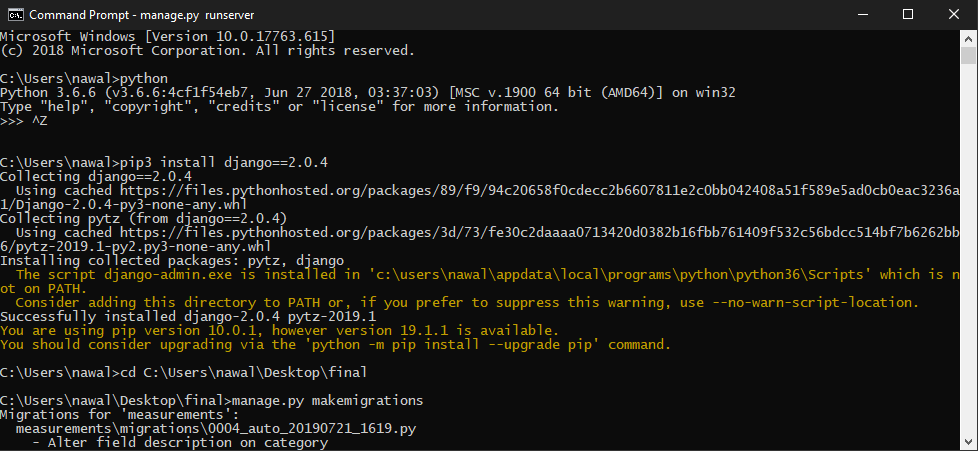
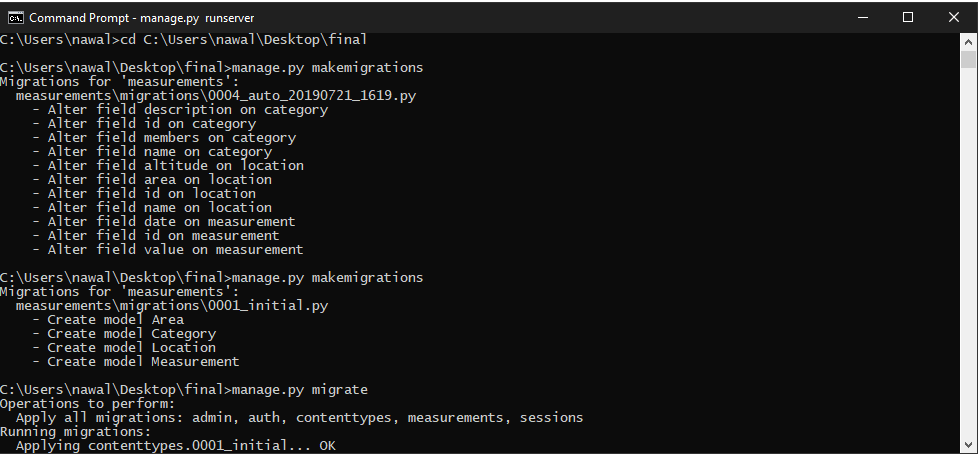
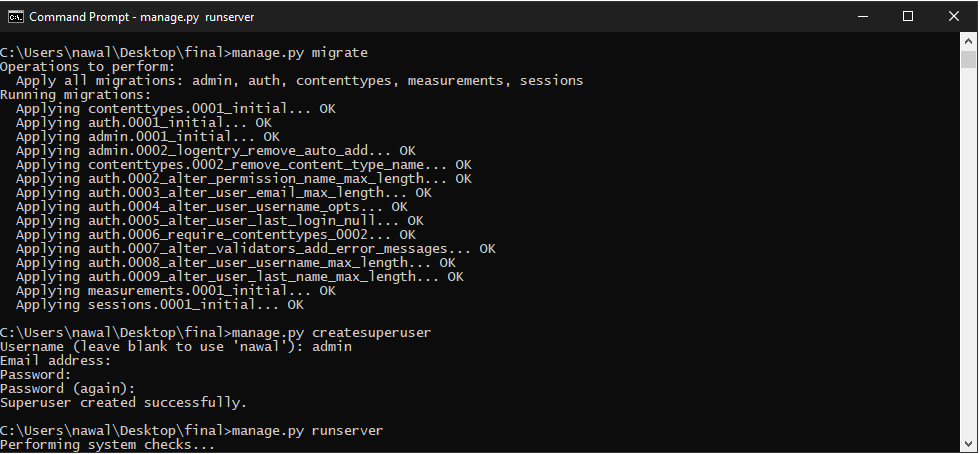
**SCREENSHOTS OF THE PROJECT RUNNING.**

****

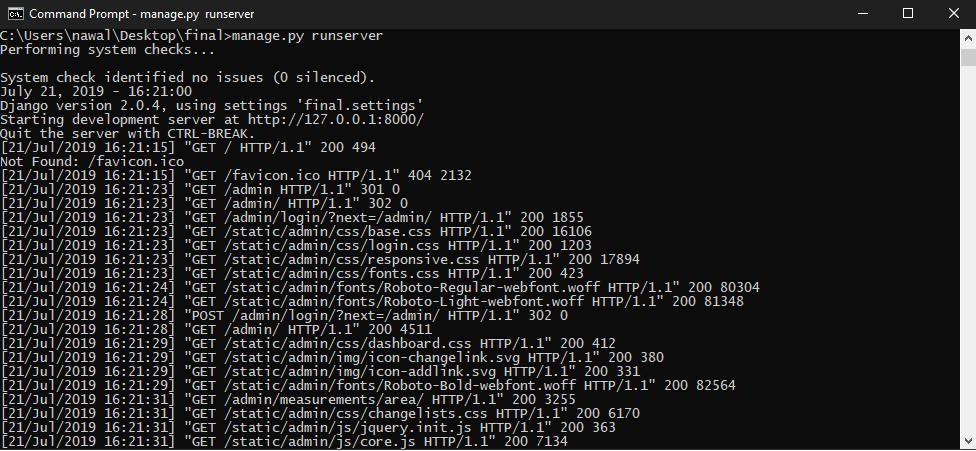
**1** - Install Django

****

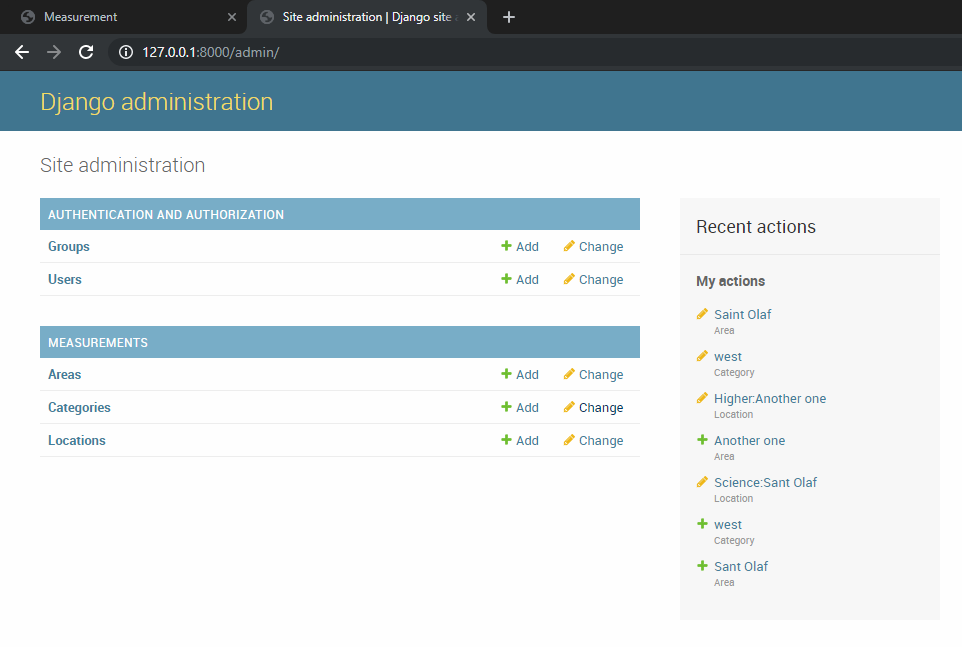
**2** - manage.py makemigrations



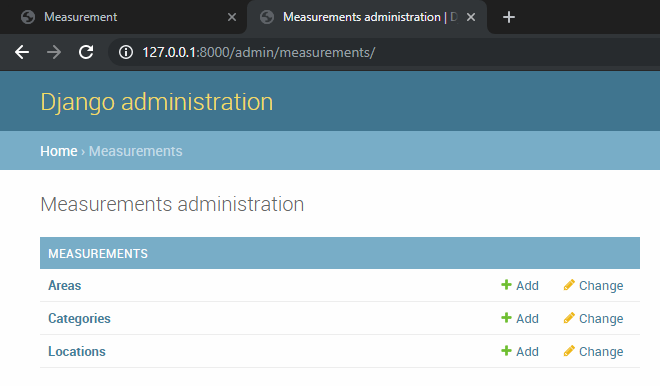
3 - manage.py migrate



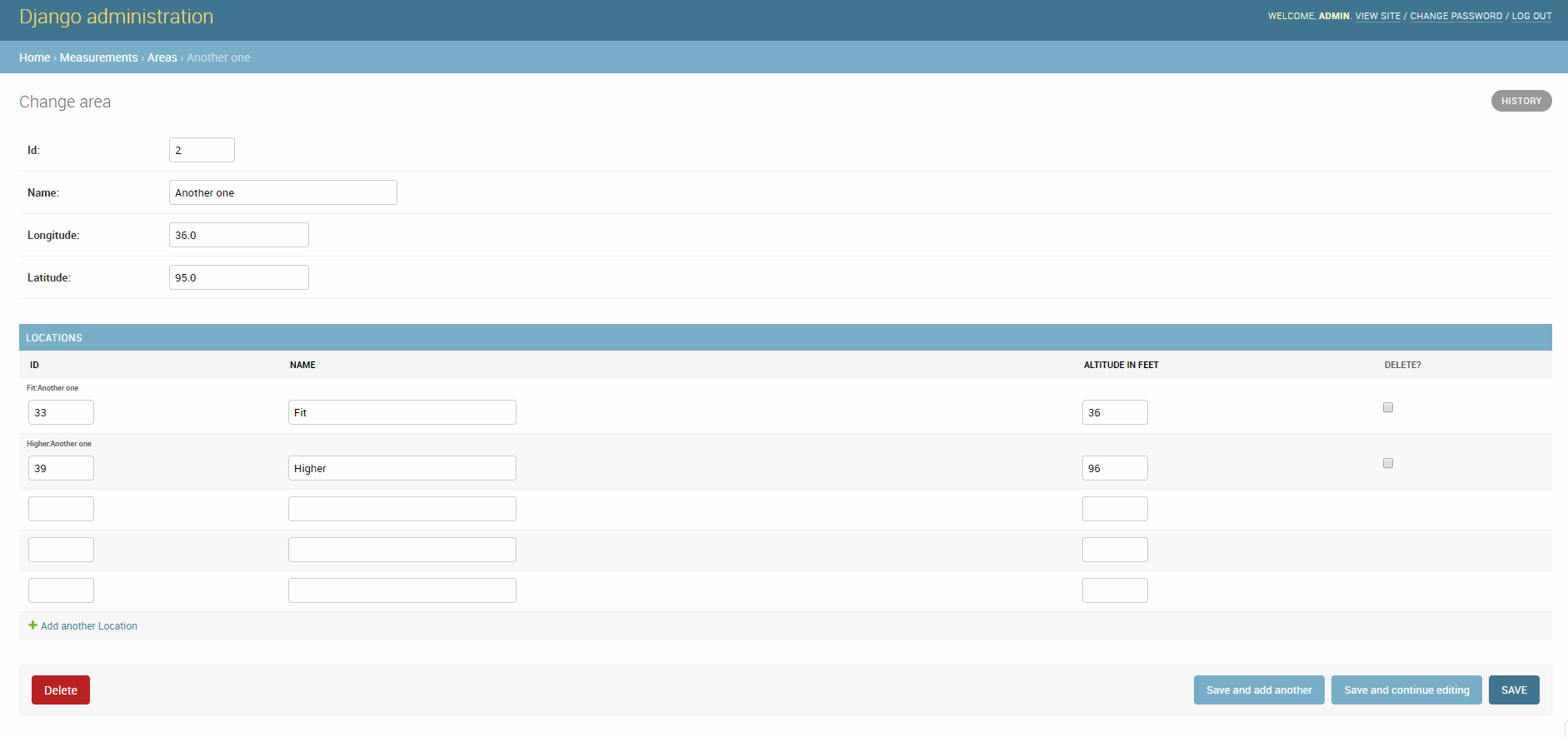
4 - manage.py runserver



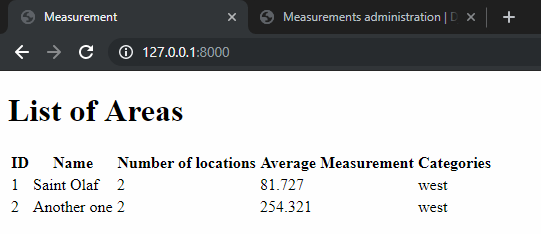
5 - Site Administration



6 - Measurements Administration



7 - Change Areas Example



8 - List of Areas Example

**REPORT DESCRIBING THE DETAILS OF THE PROJECT.**

The project is a web that serves contents saved in the database. The application was developed with Django web framework, a web application framework written in the Python programming language.

Django is a Model View Template (MVT) framework. Unlike the common Model View Controller (MVC), Django is structured to work from models to views then rendered to the template.

**MODELS.PY**

The model for the project defined in four database table which is:

**Area model:**

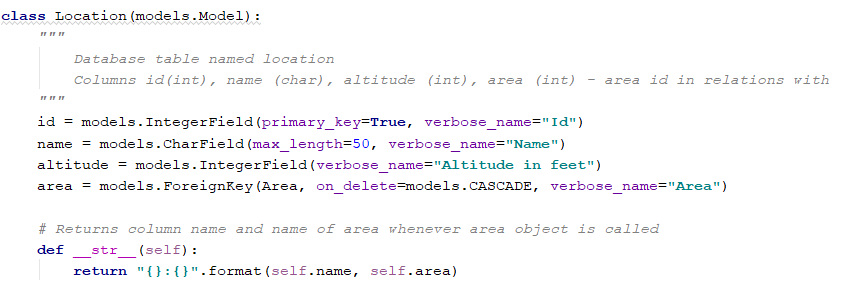
The Area database table has four columns, it contains, “id” which is the primary key in integers, “name” in characters, “longitude” and “latitude” in floats. It also contains column like methods, this include a method for returning the total number of locations for a certain area, average measurement for a certain area and category names for a certain area. Whenever the Area object is called it returns the name of the Area as the default column.



*Snippet from models.py showing the Area model*

**Location Model:**

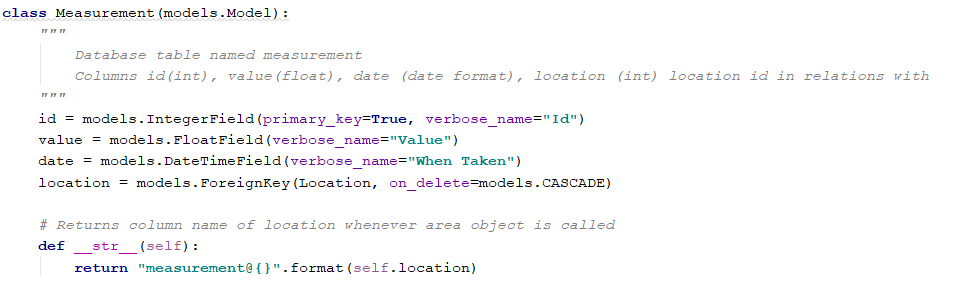
The Location database table has four columns, it contains “id” which is the primary key in integers, “name” in characters, “altitude” in integers and area in integers which is in relations with the Area’s table. Whenever the model is called, it returns a string in a certain format.



*Snippet from models.py showing the Location model*

**Measurement model:**

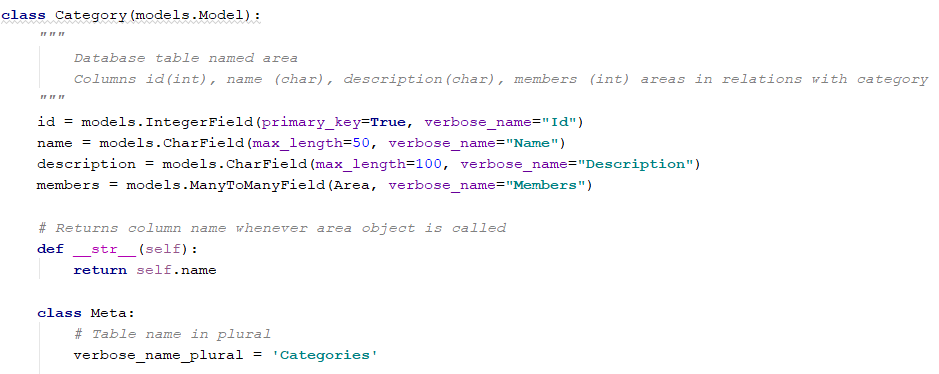
The Measurement table has four columns, it contains “id” which is the primary key in integers, “value” in float, “location” integers which is in relations with the Location’s table. Whenever the model is called, it returns a string in a certain format.



*Snippet from models.py showing the Measurement model*

**Category model:**

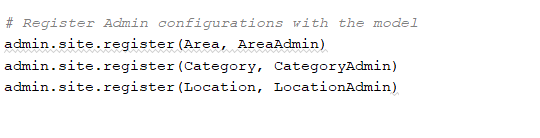
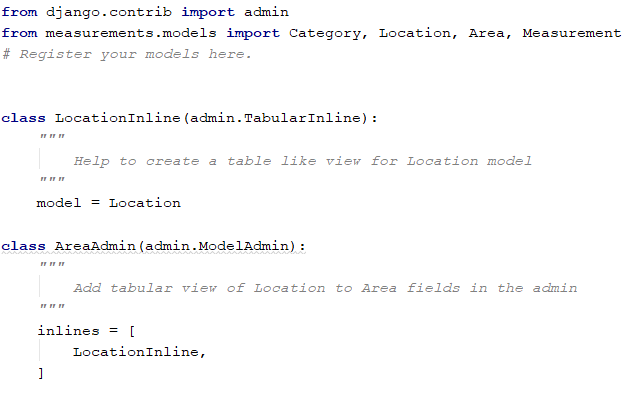
The Category database table has four columns, it contains “id” which is the primary key in integers, “name” in characters, “description” in characters and “members” which has a many to many relationship with Area.



*Snippet from models.py showing the Category model*

**ADMIN.PY**

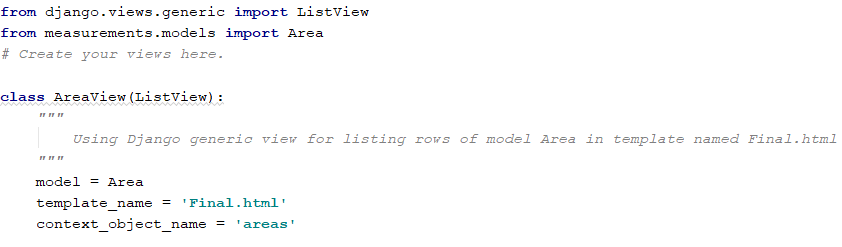
Models are configured registered into the admin interface in Django. The models Area, Location and Category are registered to be interfaced with in the admin section. Each model can be configured to display in preferred ways.



*Snippets from admin.py*

**VIEWS.PY**

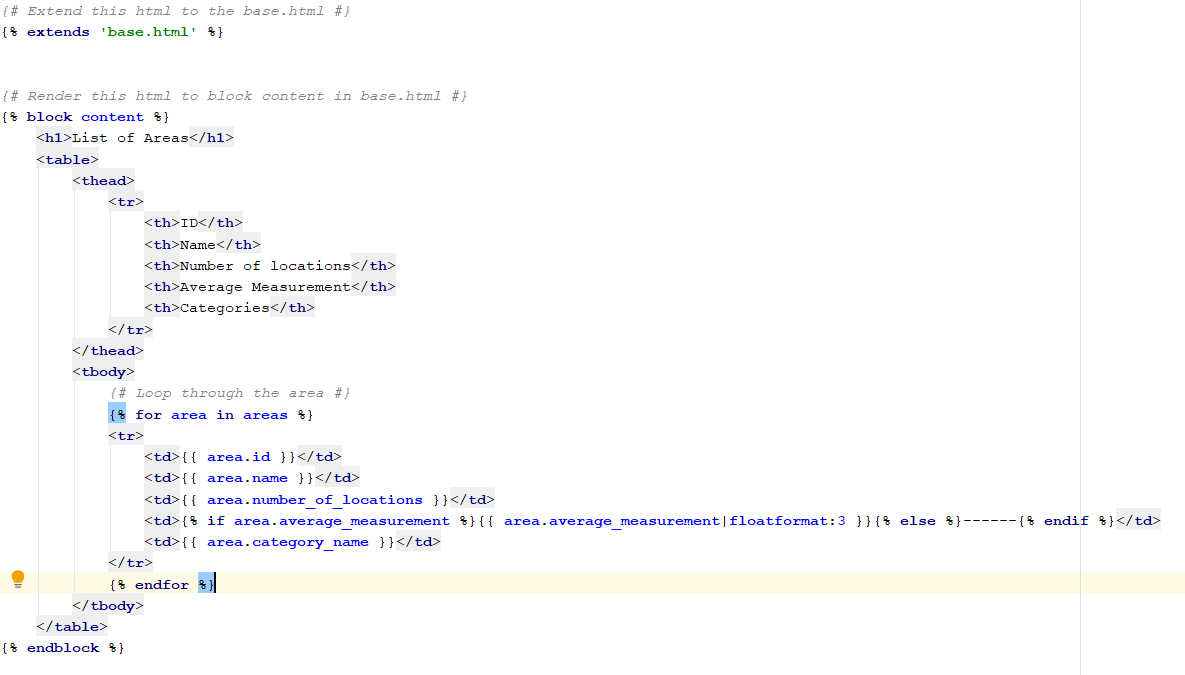
The views uses Django generic view “ListView” to renders the list of areas from the Area model to the final.html template.



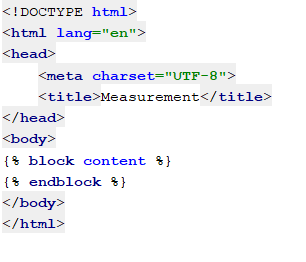
*views.py*

**TEMPLATE**

The final.html templates extends its view to the base.html. The final.html uses a table to list out row from the queried area model by and loop through it.



*Final.html*



*base.html*

**URLS.PY**

This routes url locations to certain views. The main urls.py is located in the project folder name, a sub folder of the main project name. With this the main urls.py can direct url locations to other urls.py files usally located in the app folder.

**SETTINGS.PY**

The settings.py located in the project named folder contains configuration/preferences for the application. This includes middleware, database, template and static configurations. Installed packages or apps are also configured in it. For instance the measurement app needs to be listed in the INSTALLED\_APP list, for Django to be able to find it.

**MANAGE.PY**

The manage.py behaves like a management button. It is used for running the server, making database migrations, creating super users, shell interaction and many more

**Running a server**

To run a server, you use a command prompt/terminal and run within the first project folder

manage.py runserver

**Database migrations**

manage.py makemigrations

manage.py migrate

**Create super user**

manage.py createsuperuser