Django Tutorial

1. python manage.py runserver starts the server
2. python manage.py startapp app-name will make the app that is required
3. inside urls.py of the Django setting files, all the path has to be mentioned properly to connect to the app folder
4. make urls.py under the app folder and define the urlpatterns after importing the views directory
5. provide the paths as well as the necessary rendering function assosciated to that path
6. Always remember to add the class name of apps.py under installed apps of settings.py
7. In the views.py, create the necessary functions that are rendered on requesting the url.
8. Also create the dictionary object that is to be sent along while rendering the html
9. Under the templates folder keep all the html.
10. Make a base html whose code can be copied using extend in other html pages
11. {% is used to define any logic statement
12. {{ is used to access any variable value
13. Djaneiro is the package used for easy typing Django codes
14. If path is changed in urls.py, we have to change href also: better thing to do is to use the {% url ‘path name as defined in app>urls.py’ %} as the value of href

Creating Superuser

1. First make the migration using python manage.py makemigrations
2. Then migrate using python manage.py migrate
3. This will create a structure for all the installed apps
4. Now use python manage.py createsuperuser

Creating a post

1. Edit models.py and make a class say Post(models.Model)
2. Define all the variables assosciated to that class
3. Don’t forget to import User from Django.contrib.auth.models which will help to map the user of the post using the author variable
4. On\_delete=models.CASCADE will help delete info related to the user when the author variable is deleted
5. Post.objects.all() is an array of all the post objects that have been saved in the system
6. To create a new post, make a new object of the class;
7. Eg post\_object= Post()
8. Now fill all the attributes of this object

Accessing post of a particular user

1. Use user.post\_set.all() function for a particular user object to access all the post objects made by the user
2. user.post\_set.all() returns an array of objects of the user’s posts
3. user.post\_set.create() makes an object with preassigned author = user