**DJANGO**

**To create a new Django project :** Navigate to the directory to which project is to be created.

*>> djngo-admin startproject projectname/(nameofproject)*

* This will create the project name folder inside which there is default files provided by Django.
* **@ \_\_init\_\_.py** : shows the parent folder is a package of python.
* **@settings.py:** Contain the entire setting of the website like database connectivity …
* **@url.py:** Table of content for the website. Looks at the url and performs the functionality.
* **@wsgi.py:** web server gateway interface.
* **@manage.py**:

**To start the server**

**>>** *python manage.py runserver*

* Should run this command from the project directory.
* This will give the base url to the website copy and run in web browser.

**To create new app.**

***>>*** *python manage.py startapp music/name of app (new page)*

* Will create some default files...as shown below.
* **@Migrations**.
* **@ admin** : gives the admin functionality
* **@ apps :** configuration file for this app/music
* **@ models** : Blue print of the app.
* **@ test** : to create test cases to check the bugs
* **@views** : python functions, takes user request and return them something.

[http://127.0.0.1:8000/**music/**](http://127.0.0.1:8000/music/)

When Music is requested the website URL is used to take the response.

* path(r'music/', include(**'music.urls'**)),
* This will send the request to music.urls.
* Next steps will be taken from music.urls program.
* In /music/ it is requesting nothing…
* path(r'', views.index, name='index'), # blank..goes to views.index method

**To update the database**

**>>** python manage.py makemigrations apps\_name(music)

>> python manage.py migrate

Will go to setting.py -> INSTALLED APPS and looks for all the required tables

**To add songs to the database**

* Import all the required classes to the shell
  + **from music.models import Album, Song**
* Create the object of class by using the song id/ reference to an album
  + Album1 = Album.objects.get(pk=1)
* Create the object of Song class / blank song object
  + Song = Song()
* Set the object to object of the particular album.
  + Song.album = album1
  + Song.file\_type = ‘mp3’
  + Song.song\_title **=**’I hate my boyfriend’
  + Song.save()
* **Accessing songs through set.**
  + Album1.song\_set.all()
    - Gives all the songs in the album1.
  + To create the songs in album1(pk=1)
    - Album1.song\_set.create(song\_title=’I love baycon’, song\_type=’mp3’)
    - It will automatically save the data in database.

In [1]: from music.models import Album, Song

In [2]: album = Album.objects.get(pk=1)

In [3]: album.artist

Out[3]: 'Taylor Swift'

In [4]: song = Song()

In [5]: song.album = album

In [9]: song.file\_type = 'mp3'

In [10]: song.save()

In [11]: album.song\_set.all()

Out[11]: <QuerySet [<Song: I hate my boyfriend>]>

In [13]: album.song\_set.create(song\_title='I love baycon', file\_type='mp3')

Out[13]: <Song: I love baycon>

In [14]: album.song\_set.create(song\_title='Bucky is lucky', file\_type='mp3')

Out[14]: <Song: Bucky is lucky>

In [15]: album.song\_set.create(song\_title='Ice cream', file\_type='mp3')

Out[15]: <Song: Ice cream>