12-1

DJANGO -project.

Install python -> (Brew Install python 3)

Install pip -> Sudo easy Install PIP.

Download and install postgresql from org site.

Notedown the password given to the ~~postgre~~sql.

Notedown host, port. -> localhost, 5432.

Download Visual studio code.

we can also use terminal Instead of visual studio code

Install Django

terminal -> Sudo PIP Install Django==2.3.0.1

we can enter any version we need.

Install pyscopg2

terminal -> pip Install psycopg2.

create a new folder on desktop with what ever your project is

mine is HotelManagment

Navigate to this folder In terminal by using cd.

cd Hotel mangement

Now In terminal create a new Django project In the folder created.

terminal -> django-admin startproject projectname

mine is hotel.

Now go back to the folder created in terminal by usind
Project name.
cd , folder name

Now we have to add our app to the folder. by Giving below command.

```
Python manage.py startapp webapp.
```

This will create a folder web app in our folder.

Now go the project created ("hotel")

↓

It has all the inbuilt .py files.

↓

we will be using only setting.py and url's.py. and we will update both of them.

→ first open settings.py in the editor.

Now check for the database and edit the database as we are using postgresql. but not "sql lite3.

```
DATABASE = {
    'default': {
        'ENGINE' : 'django.db.backends.postgresql',
        'NAME'   : 'hotel',
        'USER'   : 'postgres',
        'PASSWORD' : 'Admin', ( the pasword given while installing
                                post gresqL)
        'HOST'   : 'localhost',
        'PORT'   : '5432',
    }
}
```

Now open the pgAdmin 4.

↓

Now create database by giving you project name (hotel). we will not have any tables here as we did not import any table into this.

Now open a new visual studio code terminal and in that ~~give a command.~~ navigate to the folder you created.

In that give the bellow command to migrate the tables from our code to pgAdmin.

```
Python manage.py migrate -> enter.
```

Now go to the pg admin and check the tables by refreshing.

Django will have inbuilt 10 tables. with our tables -> with our application name.

we create the tables in modles.py file and django will convert it to tables.

Now select the tables you want to see click on View or edit data and you can see the tables and Information here.

TO do all of this first we have to create our code.

→ when we open the folder created on desktop it should have the -init.py, asgi.py, setting.py, urls.py, wsgi.py. and manage.py in the project folder.

→ setting.py :- change the database details from sqlite to postg

Now go to the terminal and cd → foler then cd (project) ~~to~~ the project which contains manage.py file.

→ Now enter the comand to run the server i.e.,

" Python manage.py runserver".

→ we will get the server addres, Now we have to copy it and then paste it in the chrome and check for the ~~applicatio~~ web page (" it says worked succesfully").

→ to stop the server press control -c.

Now working on the code is pending.

<u>TO DO</u> <u>List</u> :-

1. understand the coding process for django.
2. complete the couse of django.
3. develope the code. for the web application.

moving to code :-

→ first copy the urls.py from the hotel and paste it in webapp.

→ Now go to the settings.py file in "hotel"

↓

look for the installed apps.

↓

under the list of installed apps. we have to add our app.

↓

'webapp'. (like this).

Now save, here we have successfuly deploied the webapp to our setting.py.

→ Go to the urls.py. in the hotle (project) and Here add a new path to include all the webapp.urls (app's urls)

path ('', include ('webapp.urls')),

↓

This my app.

So add "include" after django.urls import path, include.

→ Now open urls.py in the webapp (your app).

↓

start adding url patterns = [:

start writing code.

path (' ', Views.home, name = "welcome"). → welcome page code

login	path('login/', views. userlogindef, name = "userlogindef"),
signup	path('signup/', views. signupdef, name = "signupdef"),
usignup	path ('usignupaction/', views. usignupactiondef, name = "usignup actiondef"),
u login	path ('userloginaction/', views. userloginactiondef, name = "userloginactiondef"),
u home	path ('userhome/', views. userhomdef, name = "userhomedef"),
u logout	path ('userlogout/', views. userlogout def, name = "user logout def"),
Rooms	Path ('view rooms/', views. viewrooms, name = "view rooms"),
check	path ('check/', views. check, name = "check"),
booking	path (' booking/', views . bookingsdef, name = "bookingsdef"),
userhome2	path ('userhome 2/', views. userhomedef 2, name = "userhomedef2"),
aboutus	path ('aboutus/', views. aboutus, name = "aboutus"),
contactus	path ('contactus/', views. contactus, name = "contactus"),
alogin	path ('alogin/', views. alogin, name = "alogin"),
a loginaction	path ('adminlogindef/', views. adminlogindef, name = "admin login def"),
a home	Path ('admin home/', views. admin home, name = "adminhome"),
a logout	Path ('admin logout/', views. admin logout, name = "a logout"),
a view room	path (' aviewrooms/', views. aview rooms, name = "aviewrooms"),
a delt room	path (' adeleterooms/', views. adeleterooms, name = "adeleterooms"),
a add room	path ('a addrooms/', views. aadd rooms, name = "aaddrooms"),
a view userlist	path ('aviewuserlist def/', views. aviewuserlistdef, name = "aviewuserlist def"),
a view Bookings	Path ('aview bookingsdef/', views. aview bookingsdef, name = "aviewbookingdef"),

Now enter. and save urls.py.

above the url patterns = [, we have to import views

so type → "from . import views.

Now lets start working on 'views.py' as we are done with 'urls.py'

all the Http request will be forwarded to the views.py.

and the business logic from views.py will be executed.

→ Now ~~here~~ have to create templates folder in the webapp (app you created);

→ In templates we have to create our ~~[illegible]~~ HTML files.

→ Now first we have to deploy templates on the setting.py.

Go to settings.py.

↓

Search for templates.

↓

then we have to enter the Directory.

↓

'DIRS': [os.path.join(BASE_DIR, 'templates')],

(this path is mostly common for all the projects).

→ Now we have to create a New folder in the webapp

→ Name that folder as static → so here we will save all our static files like Images, JS, css. files.

· after adding the static data we have to deploy it to the settings.py as we did for the other files.

by entering the following code..

```
staticfiles_dir = [
    os.path.join(BASE_DIR, 'static')
    ]
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
STATIC_URL = '/static/'
```