**1. pip install virtualenvwrapper-win** : command for downloading the virtual env things to our pc

**2. mkvirtualenv djangoLearning** : command for creating the virtual env into our pc i.e. mkvirtualenv NameOfVirtualEnvironment

**3. pip install django** : command for installation of django inside virtual environment

**4. django-admin --version** : command for checking the version of the django

**5. mkdir djangoProject(Folder Name)** : command for making directory/folder for managing the django project files inside one folder

**6. django-admin startproject djangoProject1(Project Name**) : command for creating first project of django, this command will create several .py files

**7. python manage.py runserver** : command for running local server to check the working, also manage.py will be created by the above command.

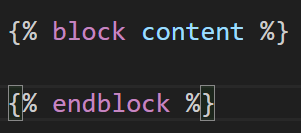
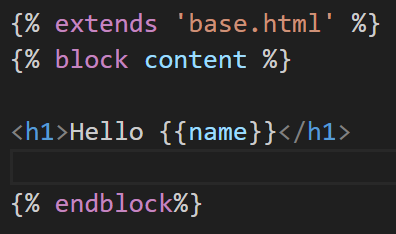
**Download VisualCode studio:**

1. For open the same virtual Environment into VS code terminal, command is : **cmd /k workon NameOfVirtualEnvironment**

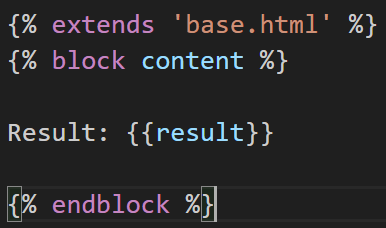
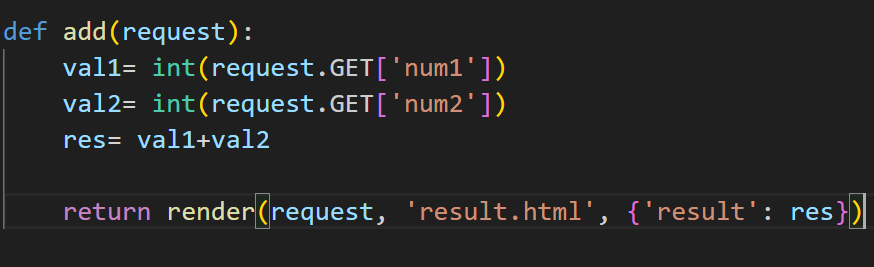
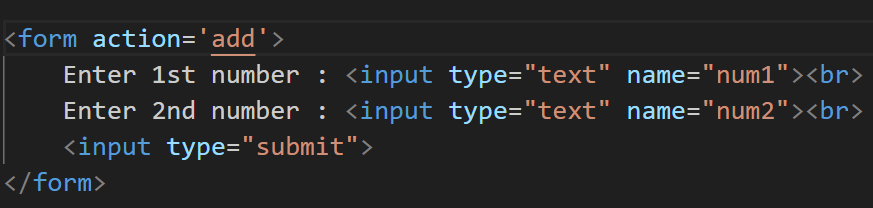
**DYNAMIC CONTENT OF HTML PAGE LOAD ON LOCAL SERVER: (Django Template Language (DTL))**

1. Create separate folder for html pages.
2. To run the content of html or any file we have to make changes in the setting.py file inside Template>DIR write this code: "**os.path.join(BASE\_DIR,'templates')**"
3. change content of home function of view.py to: " **return render (request,'home.html', {'name':'Kingo'}) "**
4. inside html file content is : **<h1>Hello {{name}}</h1>** (these double brackets represent it is a dynamic content).

**FIXING BACKGROUND AND CONTENT IS DYNAMIC OF HTML PAGE (LINKING THESE TWO)**

* Create base.html (background), change the color, then for linking write this code:  inside the body of html file.
* For applying the changes in dynamic content, write this code:  (this Hello {{name}} is our dynamic content.
* Run the server : **python manage.py runserver**.

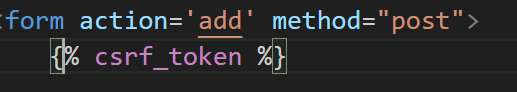
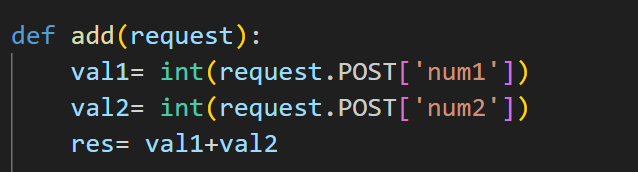
**ADDITION OF TWO NUMBER IN DJANGO**

* Create a result.html file, which will used for displaying result. 
* Inside the view.py, create a function with add name in which actual addition is done.  this is the way to get the number and sent the result to result.html file.
* Add the url of ‘add’ to the urls.py file. 
* And inside the home.html create a form for taking 2 number as input and one submit button. 

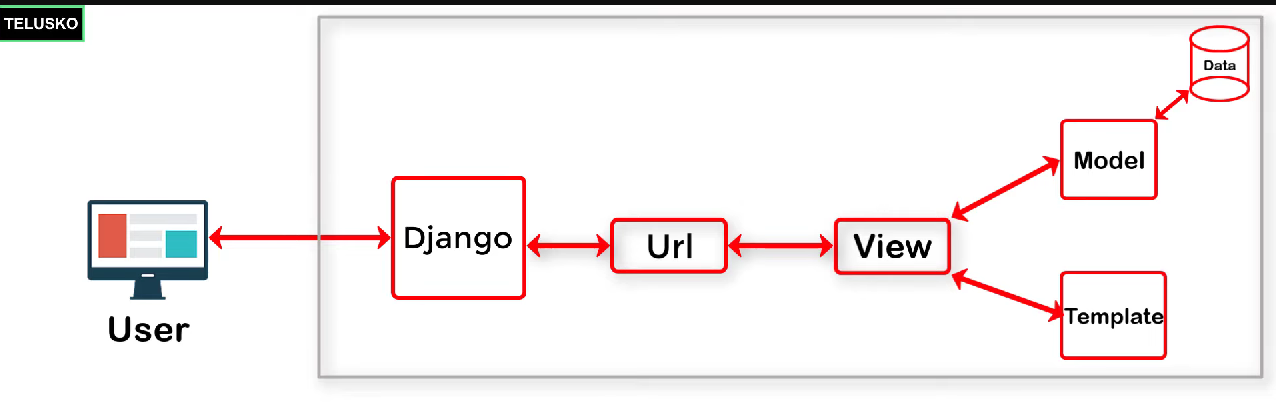
**GET VS POST (removing the data from the address bar)**

GET use when we need data from server like image, video or any text.

POST use when we upload data to server like image, number etc.

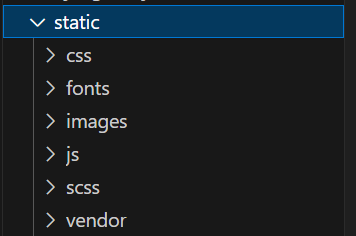
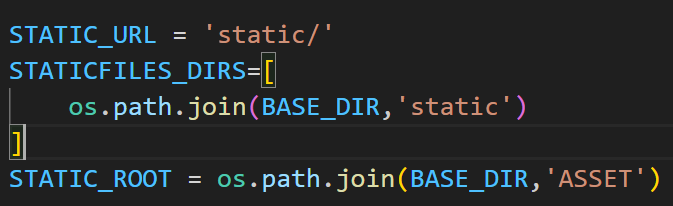
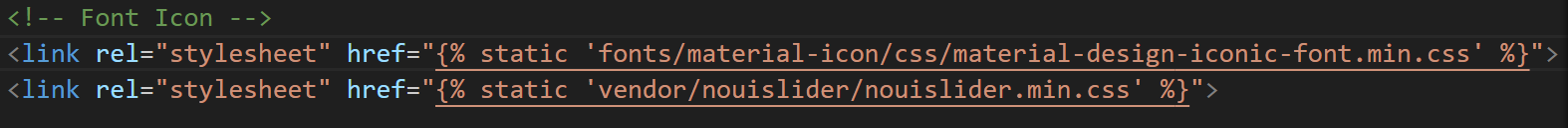
* Currently our data that we sending to server for addition of 2 number is showing on the address bar. 
* Do the changes in the home.html file , set the method to post and use this jinja csrf token.
* Also do the changes in add fxn in view file from GET to POST 

**Model View Template (MVT)**



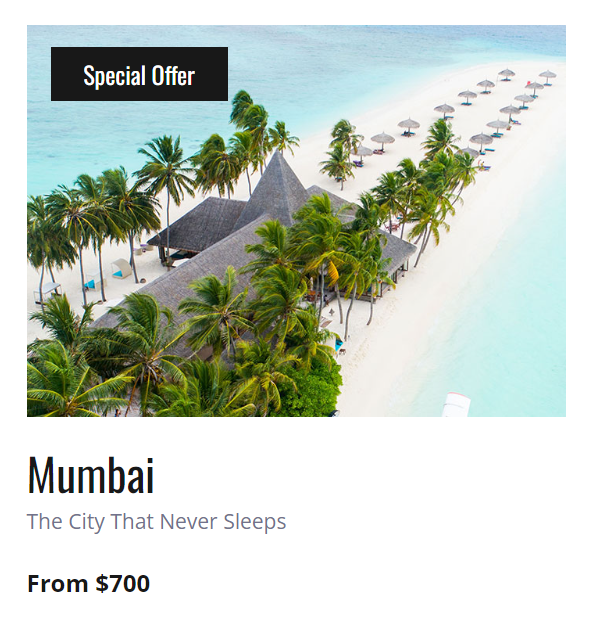
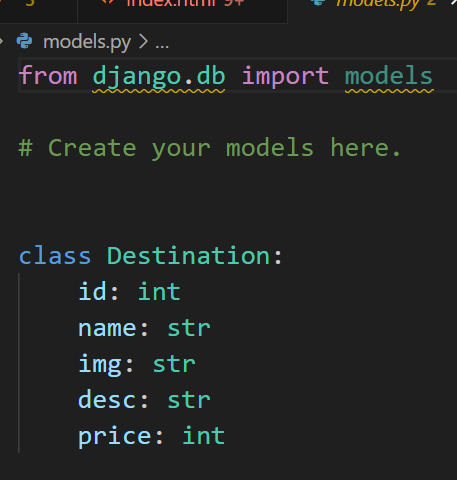
In MODEL our data part is there, in VIEW our logic/ function is there and in template is simply our html,css,js coding will be there (basically our front-end part is there). And in URL’s we gives the url for different pages (like while click on page it will take us to another url). Django is taking here as the server.

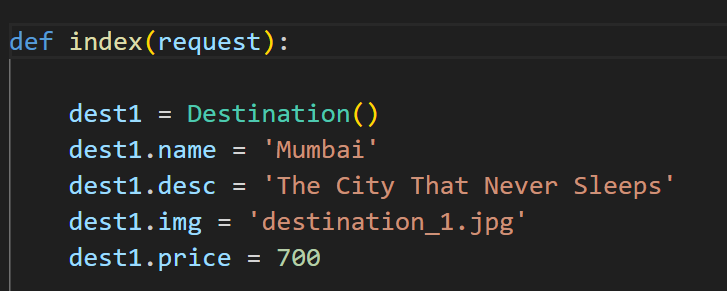
**#Video-11 (Run html website on to our Local Server)**

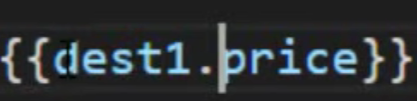
1. Copy one website template from google means its html,css,js and images that is used in that.
2. Copied all file in to one of the folder inside django. i.e. static folder 
3. **** This we write inside the setting.py file, this is used to refer the static files that where the actual static content is present and if want to use them with django than we create the STATIC\_ROOT and ASSET folder.
4. **** This command will create the ASSET folder which I mention above and for using the static content I refer it from here.
5. ****  This command is use to tell the html file (type of formatting) to load all static content (css,js,images) to the html file.
6. ****

We have to use this kind of formatting i.e. {% Folder\_name ‘link’ %}.

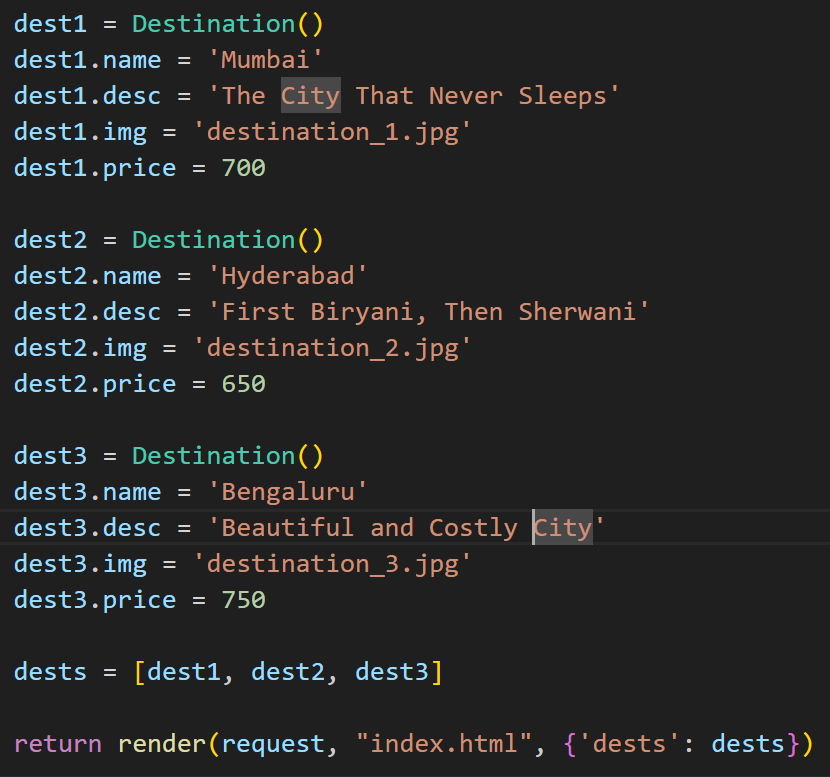
**#12 Changes the content of html file with the help of class and object**

1.  This is my content of html file i.e. image, city name, tag line,price.
2. So I will create one class inside model.py with these variables 
3. Then I call this class inside views.py



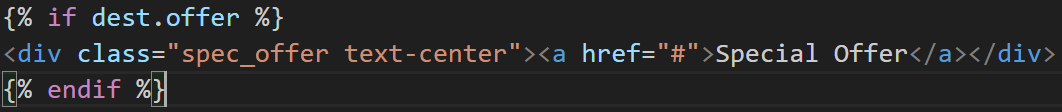
1.  return this (dest2 and dest3 are two other objects that I not shown here, similar to Mumbai).
2. At end, to fetch the data from view files that we send to .html file, we have to write this in jinja format in the place of everywhere, where we are using the data. 

**#13 Changing the content of html file where multiple same things are passes and we are updating the same content using for loop.**

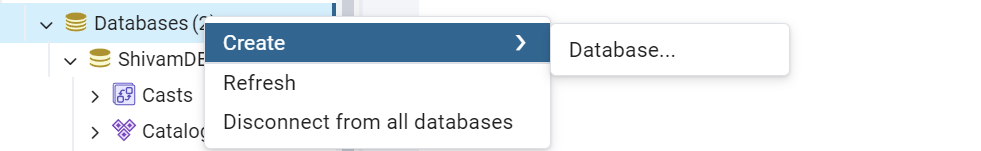
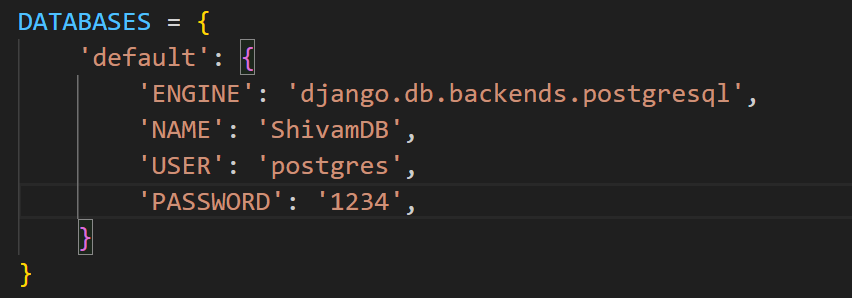
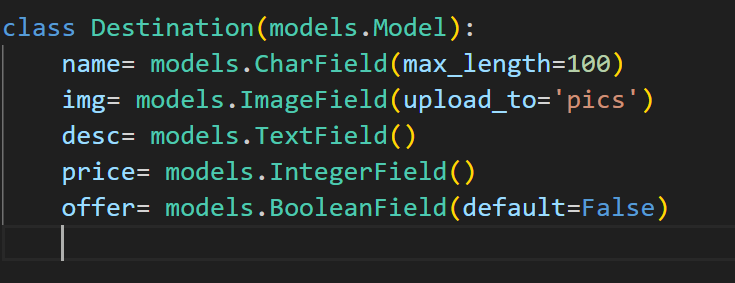
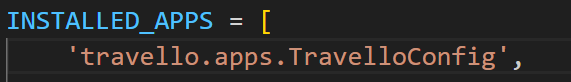
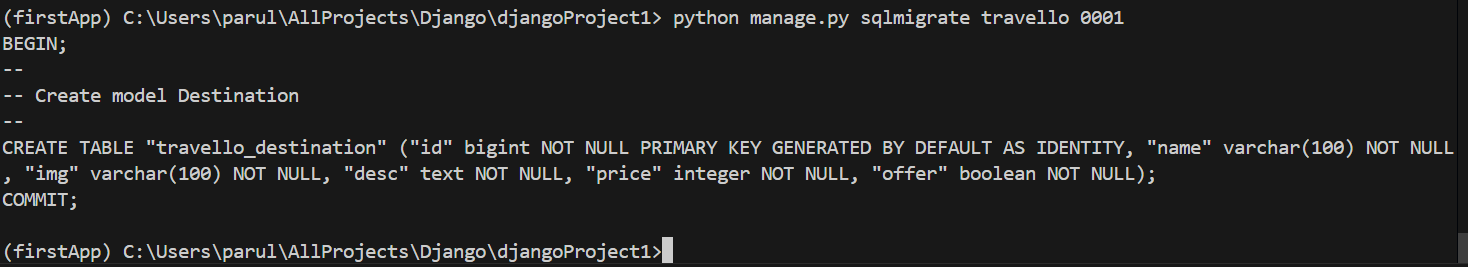
1.  Created different object and passing this object with the help of list to the html file.
2.  So this is the way to use the for loop in jinja format. Also u can see that in the place of destination name, description and price we use the jinja format , but u can see one thing also that for image showing we use different approach.
   1. This is the first approach that will come to our mind if we try to show the image using jinja format .(but this is wrong, because it is something like jinja inside jinja).
   2. Another approach is this, so this works perfectly.
   3. Another approach is like for image passing we will tell the image path to html file in jinja format  now in this ‘imagePath’ here is became the url/path of image folder where images are stored and by writing this we can give path to html file in jinja format in this case.
   4. Another way is this  is to display image.

**#14 if-else in Django**

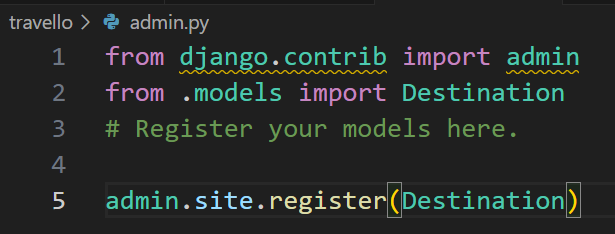
Now I want to apply special offer in the specific location instead of all places, so it would be achieved through one Boolean variable and if else statement.

* Create one bool variable in model.py file
* Initialize the value of this variable in views.py file, similarly initialize the variable for dest2 and dest3.
* Finale step, write the if else statement in index.html file in jinja format. 

**#17 Configuring Postgres and PgAdmin setup**

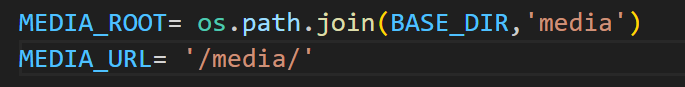
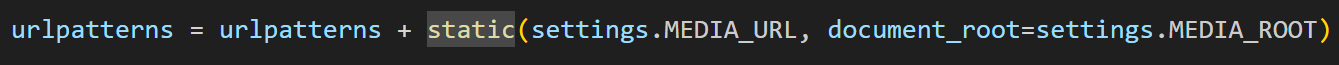
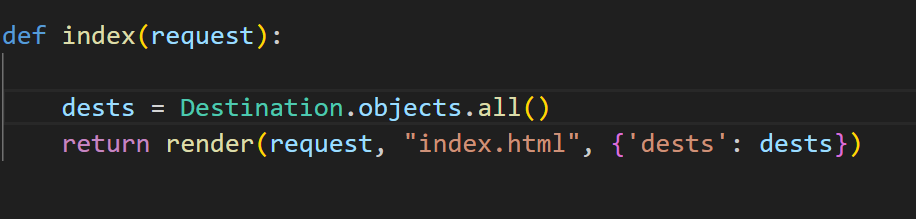
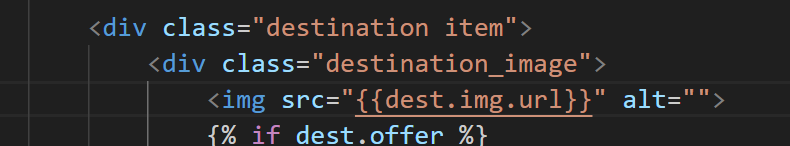
1. Create the DB in pgAdmin window app, by simply right clicking. 
2. Second step is to do changes in the setting.py of your project. 
3. Now for connecting these two app (postgres and python) we need one connector and i.e. psycopg2, (install this simply by ‘pip install psycopg2’).
4. Create models in the model.py file (here model means the columns of the table we have to specify) (<https://docs.djangoproject.com/en/5.1/ref/models/fields/)--> take the help of this.
5. Setting up your app inside setting.py , basically we have to tell our django to all our app.
6. Do the migration using command , this command will create one migration file inside migration folder, and that folder contains all the columns fields which we have created in model.py inside the class.
7.  now with the above command we can write the sql query without writing sql query, and it will create table for us.
8. this command will migrate the table to the postgre DB.

**#19 Working with Admin Panel**

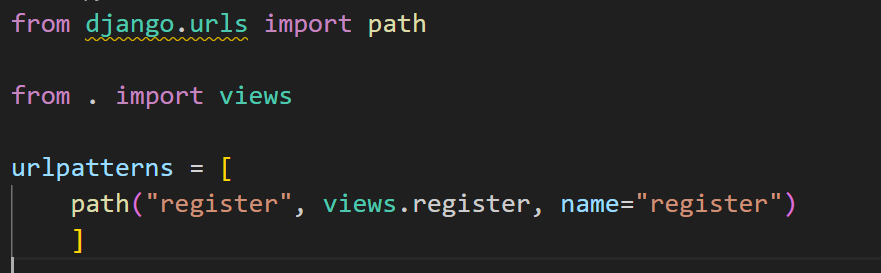
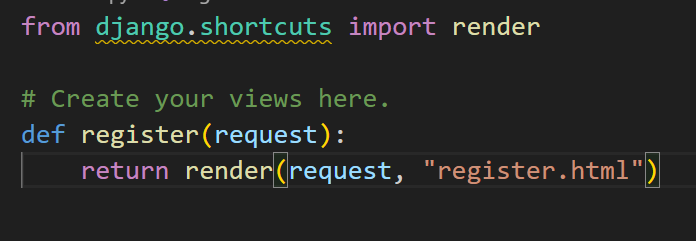
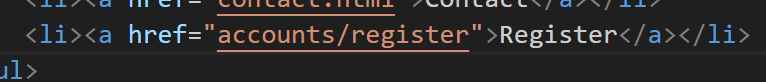
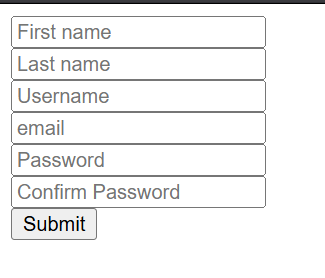
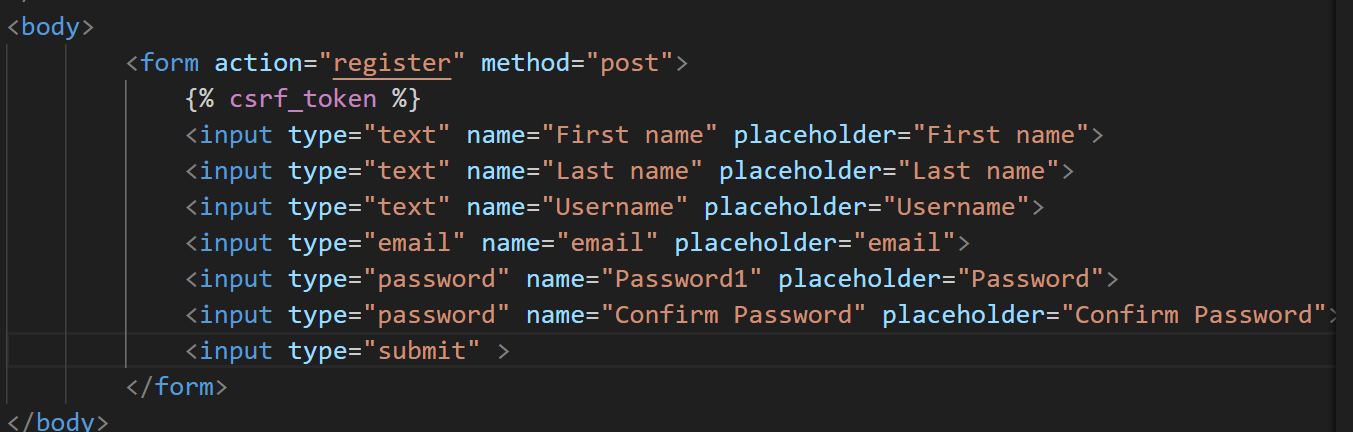
1. By default django create admin page for us and simply we can use it by http://127.0.0.1:8000/admin.
2. But before that we have to create the credentials of admin, so that thing we can do that by simple command, i.e. python manage.py createsuperuser.
3. Then to import destination model that we created in db, we can import simply by this (here in this case model is Destination). .
4. And our job is done. And now you can see the destination model in to your admin interface.

**#20 Add and Fetch data from DB**

Now here we will learn that how to store data in db and it will reflect into the output website, without touching the html code:

1. First we have to do some changes in settings.py file , i.e. we have to add the media\_url and media\_root.
2. Second we have to add url to url file. 
3. After that we have to add new data using add add destination button, using admin panel page, and after adding we can verify it by seeing the table.
4. After that go to views.py file, remove the static content, and write than simple one line command, 
5. Than we have to change the image src in index.html file . And our job is done, now this time if you add any new destination, it will directly reflect to the html page, without doing any changes in the html code.

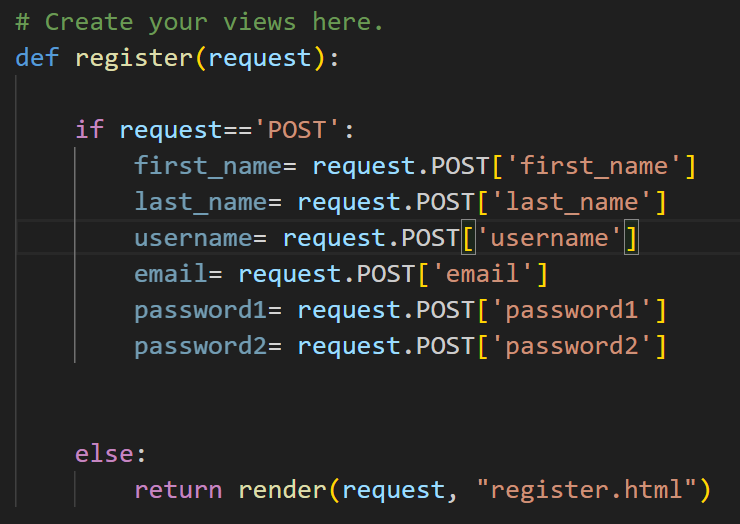
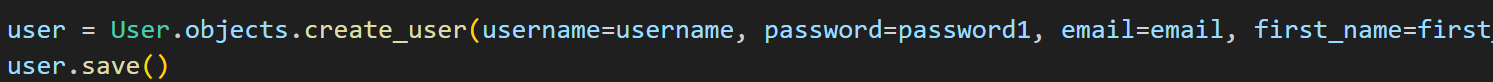
**#22 User Registration Form created and Stored data in database**

1. We are creating separate registration module for register, login, logout things.
2. We created separate module using .
3. In accounts modules we don’t have url.py file, so we have to create it, and copied the code from telusko url.py file, i.e. 
4. Also we have to create the function named register inside views.py file of accounts module. But before that we have to add url path of account module to telusko module, i.e. .
5. Created the register function inside views file of accnt module. 
6. Add the Register button in travello website . 
7. Create the register.html file and add following text areas.

**#22 User Registration Part 2**

Till now we create register.html , but if we enter the data and press submit button, then it will not go any where, so in this part we will submit our data to database.

Now here in register.html our action is register, so what the register is : so the register we are using in view file, so inside views file we have to do all POST request to the database, so that our data will be stored there.

1.  So we checked the request , if post then we post the first\_name to the first\_name of table column.
2. Now before saving this data to db we have to create the object, which handle the db, 
3. Now to validate that username already exist of gmail already register or password not matched, 