python3 -m venv myenv

myenv\Scripts\activate.bat

pip install package\_name

deactivate

pip install django

**// creating a new project**

django-admin startproject name\_of\_the\_project

//mysite

//go to **name\_of\_the\_project<folder>** directory

cd name\_of\_the\_project

**//running it on local server**

python <directory of manage.py> runserver

// example

python .\mysite\manage.py runserver

**// create app**

python <directory of manage.py> startapp name\_of\_the\_app

//main

//example

python .\mysite\manage.py startapp main

// it should create name\_of\_the\_app <folder> in the name\_of\_the\_project<folder>

// if not make sure that name\_of\_the\_app <folder> is in name\_of\_the\_project<folder>

// try moving the folder

//go to main folder >>open **views.py**

from django.shortcuts import render

from django.http import HttpResponse

def index(response):

    return HttpResponse("Hey Hello")

# Create your views here.

//create new file named **urls.py** in **main<folder>**

from django.urls import path

from . import views

urlpatterns = [

    path("",views.index,name="index")

]

//to **link urls.py** of main<folder> with the actual project

//now go to **urls.py (in mysite <folder>)**

// import include from django.urls

"""

URL configuration for mysite project.

The `urlpatterns` list routes URLs to views. For more information please see:

    https://docs.djangoproject.com/en/4.2/topics/http/urls/

Examples:

Function views

    1. Add an import:  from my\_app import views

    2. Add a URL to urlpatterns:  path('', views.home, name='home')

Class-based views

    1. Add an import:  from other\_app.views import Home

    2. Add a URL to urlpatterns:  path('', Home.as\_view(), name='home')

Including another URLconf

    1. Import the include() function: from django.urls import include, path

    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))

"""

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include("main.urls")),

]

**DATABASE**

//go to **settings.py** in **mysite<folder>**

// add name\_of\_app.apps.Nmae\_of\_appConfig in INSTALLED\_APPS

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'main.apps.MainConfig',

]

// next **migrate**

python .\manage.py migrate

//now **create models** in **models.py**

from django.db import models

# Create your models here.

class ToDoList(models.Model):

    name=models.CharField(max\_length=200)

    def \_\_str\_\_(self):

        return self.name

class Item(models.Model):

    # connecting ToDoList to this item

    todolist=models.ForeignKey(ToDoList, on\_delete=models.CASCADE)

    text=models.CharField(max\_length=200)

    complete=models.BooleanField()

    def \_\_str\_\_(self):

        return self.text

// **to save this changes**, we need to tell python to migrate

// python <directory of manage.py> makemigrations name\_of \_the\_app

python .\manage.py makemigrations main

// now a**pply the change**

// python <directory of manage.py> migrate

python .\manage.py migrate

// now you can see migration <folder> in main<folder>

//to **check the functionality of database**

// **open shell** in python

python .\manage.py shell

*>>> from main.models import Item,ToDoList*

***//create name in the list***

*>>> t=ToDoList(name='Chummus List')*

*>>> t.save()*

***// show names in the list***

*>>> ToDoList.objects.all()*

*<QuerySet [<ToDoList: Chummus List>]>*

***// show names by id***

*>>> ToDoList.objects.get(id=1)*

*<ToDoList: Chummus List>*

***// show items in the list***

*>>> t.item\_set.all()*

*<QuerySet []>*

***// create item in the list***

*>>> t.item\_set.create(text="remember AK",complete=False)*

*<Item: remember AK>*

*>>> t.item\_set.all()*

*<QuerySet [<Item: remember AK>]>*

*>>> t.item\_set.get(id=1)*

*<Item: remember AK>*

***// filtering***

*>>> ToDoList.objects.filter(name\_\_startswith=”Ch”)*

*<QuerySet [<ToDoList: Chummus List>]>*

***// delete an object***

*>>> name\_obj=ToDoList.objects.get(id=1)*

*>>> name\_obj.delete()*

***//quit from shell***

*>>> quit()*

// **passing information into webpages** from **url** basics

urlpatterns = [

    path("<int:id>",views.index,name="index"),

]

def index(response, id):

    return HttpResponse("<h1>%s</h1>" %id)

// now **getting actual information** from **passing id in url**

from django.shortcuts import render

from django.http import HttpResponse

from .models import ToDoList,Item

def index(response, id):

    ls=ToDoList.objects.get(id=id)

    return HttpResponse("<h1>%s</h1>" %ls.name)

// **by name** and getting **items from list**

urlpatterns = [

    path("<str:name>",views.index,name="index"),

]

def index(response, name):

    ls=ToDoList.objects.get(name=name)

    item=ls.item\_set.get(id=1)

    return HttpResponse("<h1>%s</h1><br><p>%s<p>" %(ls.name,str(item.text)))

**ADMIN PAGE**

**// create superuser**

python .\manage.py createsuperuser

// give **access to database** to admin page

// go to admin.py

// import the models and register the database

from django.contrib import admin

from .models import ToDoList, Item

# Register your models here.

admin.site.register(ToDoList)

admin.site.register(Item)

**TEMPLATES & CUSTOM HTML**

// **adding home page via template**

// **add a new folder** named **templates** in **main<folder>**

// and add a **new main <folder>** in the above **templates<folder>** created

// **create templates** in this main folder

urlpatterns = [

    path("<int:id>",views.index,name="index"),

    path("",views.home,name="home")

]

def index(response, id):

    ls=ToDoList.objects.get(id=id)

    return render(response,"main/base.html",{})

    # item=ls.item\_set.get(id=1)

    # return HttpResponse("<h1>%s</h1><br><p>%s<p>" %(ls.name,str(item.text)))

def home(response):

    return render(response,"main/home.html",{})

<html>

  <head>

    <title>{% block title %}Base{% endblock %}</title>

  </head>

  <body>

    <div id="content">{% block content %} Base Page {% endblock %}</div>

  </body>

</html>

{% extends 'main/base.html' %}

{% block title %}Home{% endblock %}

{% block content %}

<h1>Hey Home</h1>

{% endblock %}

// **showing list via templates**

def index(response, id):

    ls=ToDoList.objects.get(id=id)

    return render(response,"main/list.html",{"ls":ls})

{% extends 'main/base.html' %}

{% block title %}List{% endblock %}

{% block content %}

    <h1>{{ls.name}}</h1>

    <ul>

        {% for item in ls.item\_set.all %}

        <li>{{item.text}}</li>

        {% endfor %}

    </ul>

{% endblock %}

**FORMS**

//create **forms.py** in mysite/main <folder>

// create **create.html** in templates/main<folder>

def create(response):

    form=CreateNewList()

    return render(response,"main/create.html",{"form":form})

urlpatterns = [

    path("<int:id>",views.index,name="index"),

    path("",views.home,name="home"),

    path("create/",views.create,name="create"),

]

{% extends 'main/base.html' %}

{% block title %}Create Form{% endblock %}

{% block content %}

    <form action="/create/" method="post">

{% csrf\_token %}

        {{form}}

// form.as\_table

//form.as\_p (vertically down)

//form.as\_ul (as list)

        <button type="submit">Create New</button>

    </form>

{% endblock %}

from django import forms

class CreateNewList(forms.Form):

    name=forms.CharField(label="Name", max\_length=200)

    check= forms.BooleanField(required=False)

// **POST METHOD form**

from django.http import HttpResponse, HttpResponseRedirect

def create(response):

    if response.method == "POST":

        form=CreateNewList(response.POST)

        if form.is\_valid():

            n=form.cleaned\_data["name"] #takes "name" into n <variable>

            t=ToDoList(name=n) # creates a list of above name

            t.save() # saves it

        return HttpResponseRedirect("/%i" %t.id) #returns to a seperate page where it shows the filled name

    else:

        form=CreateNewList()

    return render(response,"main/create.html",{"form":form})

**TO DO LIST APPLICATION**

// **to do list** application

{% extends 'main/base.html' %}

{% block title %}List{% endblock %}

{% block content %}

    <h1>{{ls.name}}</h1>

    <form action="#" method="post">

        {% csrf\_token %}

        <ul>

            {% for item in ls.item\_set.all %}

                {% if item.complete == True %}

                    <li><input type="checkbox" name="c{{item.id}}" value="clicked" checked >{{item.text}}</li>

                {% else %}

                    <li><input type="checkbox" name="c{{item.id}}" value="clicked">{{item.text}}</li>

                {% endif %}

            {% endfor %}

        </ul>

        <button type="submit" name="save" value="save">Save</button>

        <input type="text" name="new">

        <button type="submit" name="newItem" value="newItem">Add Item</button>

    </form>

{% endblock %}

def index(response, id):

    ls=ToDoList.objects.get(id=id)

    if response.method=="POST":

        print(response.POST)

        if response.POST.get("save"): #button\_name

            for item in ls.item\_set.all():

                if response.POST.get("c"+str(item.id))=="clicked": #input\_name

                    item.complete=True

                else:

                    item.complete=False

                item.save()

        elif response.POST.get("newItem"): #button\_name

            txt=response.POST.get("new") #input\_name

            if len(txt)>1:

                ls.item\_set.create(text=txt,complete=False)

            else:

                print("invalid text")

    return render(response,"main/list.html",{"ls":ls})

**USER REGISTRATION**

// create new app **register**

// so that we can use this in any other projects

python manage.py startapp register

// **define views** in **register <folder>**

from django.shortcuts import render

def register(response):

    return render()

// **link urls** in mysite<folder>

from django.urls import path, include

from register import views as v

urlpatterns = [

    path('admin/', admin.site.urls),

    path('register/', v.register, name="register"),

    path('', include("main.urls")),

]

// now **create templates** in register **by adding a templates folder** and **a register folder again**

{% extends "main/base.html" %}

{% block title %}

Create an Account

{% endblock %}

{% block content %}

    <form action="" method="post">

{% csrf\_token %}

        {{form}}

        <button type="submit">Register</button>

    </form>

{% endblock %}

from django.shortcuts import render

from django.contrib.auth import login,authenticate #inbuilt django tools

from django.contrib.auth.forms import UserCreationForm #inbuilt django tools

def register(response):

    form= UserCreationForm()

    return render(response,"register/register.html",{"form":form})

// add our **register app** in **setting.py**

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'main.apps.MainConfig',

    'register.apps.RegisterConfig',

]

// **add POST method**

from django.shortcuts import render,redirect

from django.contrib.auth import login,authenticate #inbuilt django tools

from django.contrib.auth.forms import UserCreationForm #inbuilt django tools

def register(response):

    if response.method=="POST":

        form= UserCreationForm(response.POST)

        if form.is\_valid():

            form.save()

        return redirect("/home")

    else:

        form= UserCreationForm()

    return render(response,"register/register.html",{"form":form})

// **adding our own form attributes** in user registration **like email** etc

// **create forms.py** in **register<folder>**

from django.contrib.auth import login,authenticate #inbuilt django tools

from django.contrib.auth.forms import UserCreationForm #inbuilt django tools

from django import forms

from django.contrib.auth.models import User

class RegisterForm(UserCreationForm):

    email = forms.EmailField(required=True)

    class Meta:

        model= User

        fields=["username","email","password1","password2"] # what order you things to show up

**// edit views.py** according to the function changes in **forms.py**

**//** change UserRegistrationForm into new function name

from django.shortcuts import render,redirect

from .forms import RegisterForm

def register(response):

    if response.method=="POST":

        form= RegisterForm(response.POST)

        if form.is\_valid():

            form.save()

        return redirect("/home")

    else:

        form= RegisterForm()

    return render(response,"register/register.html",{"form":form})

**ADDING INBUILT STYLING**

// pip install django-crispy-forms

// now **add this module** in **setting.py**

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'crispy\_forms',

    'main.apps.MainConfig',

    'register.apps.RegisterConfig',

]

STATIC\_URL = 'static/'

CRISPY\_TEMPLATE\_PACK ='bootstrap4'

// now **load crispy\_forms\_tags** in the html page

{% extends "main/base.html" %}

{% block title %}

Create an Account

{% endblock %}

{% load crispy\_forms\_tags %} //loaded

{% block content %}

    <form action="" method="post">

        {% csrf\_token %}

        {{form|crispy}} // changed |crispy

        <button type="submit">Register</button>

    </form>

{% endblock %}

**AUTHENTICATION**

//open **urls.py** in **mysite<folder>**

// add inbuilt path

urlpatterns = [

    path('admin/', admin.site.urls),

    path('register/', v.register, name="register"),

    path('', include("main.urls")),

    path("",include("django.contrib.auth.urls")),

]

// create **<new folder>** **registration** in **register <app>** and in **templates <folder>**

// now create **login.html & logout.html**

{% extends "main/base.html" %}

{% block title %}

Login Page

{% endblock %}

{% block content %}

    <form action="" method="post" class="form-group">

        {% csrf\_token %}

        {{form}}

        <button type="submit" class="btn btn-success">Login</button>

        <p>

            Don't have an account? <a href="/register">Click Here</a>

        </p>

    </form>

{% endblock %}

// add this in **settings.py**

LOGIN\_REDIRECT\_URL ='/'

LOGOUT\_REDIRECT\_URL ="/"

// change in **base.html** to show to **only authenticated users**

  {% if user.is\_authenticated %}

        {% block content %} Base Page {% endblock %}

      {% else %}

        <p>LOGIN<a href="/login">here</a></p>

      {% endif %}

    </div>

**CUSTOM USERS**

//open **models.py** in **main<folder>**

from django.db import models

from django.contrib.auth.models import User # new import for custom user

# Create your models here.

class ToDoList(models.Model):

    user=models.ForeignKey(User, on\_delete=models.CASCADE, related\_name=”todolist”,null=True) #changed

    name=models.CharField(max\_length=200)

    def \_\_str\_\_(self):

        return self.name

class Item(models.Model):

    # connecting ToDoList to this item

    todolist=models.ForeignKey(ToDoList, on\_delete=models.CASCADE)

    text=models.CharField(max\_length=200)

    complete=models.BooleanField()

    def \_\_str\_\_(self):

        return self.text

//  **do migrations**

// **delete everything except init.py** in **migrations<folder>** to delete database

python manage.py makemigrations

// open **views.py** in **main<folder>**

// edit **base.html** to create **view.html**

<div class="sidebar">

      <a href="/">Home</a>

      <a href="/view">View</a>

      <a href="/create">Create</a>

    </div>

// **add path** in url.py in main<folder>

urlpatterns = [

    path("<int:id>",views.index,name="index"),

    path("",views.home,name="home"),

    path("create/",views.create,name="create"),

    path("view/",views.view,name="view"),

]

// now **define view** in **views.py**

def view(response):

    return render(response."main/view.html",{})

// now **create view.html** in templates/main