Contents & Index

[Getting Started 2](#_Toc124927374)

[Creating an Environment 4](#_Toc124927375)

[Downloading Django 5](#_Toc124927376)

[Start a Django Project in the Current Directory 5](#_Toc124927377)

[Start an Django App 6](#_Toc124927378)

[Migrate 8](#_Toc124927379)

[Creating Templates Folder 8](#_Toc124927380)

[Print the Contents of a HTML File 9](#_Toc124927381)

[Data Communication 10](#_Toc124927382)

[Call Data to the Template 10](#_Toc124927383)

[Send List Data to the Template 10](#_Toc124927384)

[Creating Themes 11](#_Toc124927385)

[Form Input 12](#_Toc124927386)

[SQL 14](#_Toc124927387)

[Initialization 14](#_Toc124927388)

[Migrations of Models 14](#_Toc124927389)

[Read &or Write Data 15](#_Toc124927390)

[Static Folder 17](#_Toc124927391)

[Initialization 17](#_Toc124927392)

[Create Resources Folders 18](#_Toc124927393)

[CSS 18](#_Toc124927394)

[JS 18](#_Toc124927395)

[IMG 18](#_Toc124927396)

[Fonts 18](#_Toc124927397)

[Collect Static 18](#_Toc124927398)

[Connect to HTML Page 19](#_Toc124927399)

[Project 20](#_Toc124927400)

[Refer 21](#_Toc124927401)

[HTML 21](#_Toc124927402)

[SQL 21](#_Toc124927403)

# Getting Started

|  |  |
| --- | --- |
| Create a New Folder for Django Projects: |  |
| Select the New Folder: |  |
| Open the New Folder: |  |
| Then Right Click & Open in Terminal |  |
| OR | |
| Click on the Path |  |
| and Type CMD |  |
| OR In Case That Doesn’t Work Copy Your Path and CD to that Directory in CMD |  |
| THEN We Have the CMD Open  in the Path of the Folder we Need to Work in |  |
| Type:  python --version  pip --version |  |
| In Case You Don’t Get the Desired Output Go to <https://www.python.org/downloads/> Download the Latest Version and During Installation Don’t Forget to **Add Python to Path** *&or* **Add Python to Environment Variables**. | |

# Creating an Environment

|  |  |
| --- | --- |
| pip install virtualenvwrapper-win  mkvirtualenv [nameofenv] |  |
| To Select the Environment:  workon [nameofenv] |  |
| Now We Install Django onto This Environment. | |

# Downloading Django

|  |  |
| --- | --- |
| pip install django  django-admin –version |  |

# Start a Django Project in the Current Directory

|  |  |
| --- | --- |
| django-admin startproject [ProjectName] |  |
| This Folder will be Created - |  |
| The Files Created in the Following Folder is = |  |
| Run the Server to Check if the Project Files are not Corrupted and the Django Project is Working:  cd [ProjectName]  python manage.py runserver |  |
| Go to Link <http://127.0.0.1:8000/> and If the Below Output is What You Get It Means Django Project is Working. |  |
| to End Server  Ctrl + C |  |

# Start an Django App

|  |  |  |
| --- | --- | --- |
| python manage.py startapp [AppName] | |  |
| Then The Files in the Project File will be Updated = | |  |
| Open the Base Directory Project Folder with your Editor | |  |
| in my Case I’m Using VS-Code Editor | |  |
| Add a New File Called *urls.py* to the ***[AppName] Folder*** | |  |
| In the *projectname/appname/views.py* Create a Function with the Page ID: | | |
| FROM | from django.shortcuts import render  # Create your views here. | |
| TO | from django.shortcuts import render  from django.http import HttpResponse  from django.template import loader  # Create your views here.  def pageID(request):      return HttpResponse("<p>Basic HTML Codes Can Go Here</p>") | |
| In the *projectname/appname/urls.py* | | |
| FROM |  | |
| TO | from django.urls import path  from . import views  urlpatterns = [      #This is Where You add the Path Code      path('',views.pageID, name = "Page Name" ), #Path for HomePage would be Empty  ] | |
| In the *projectname/projectname/urls.py* | | |
| FROM | from django.contrib import admin  from django.urls import path  urlpatterns = [      path('admin/', admin.site.urls),  ] | |
| TO | from django.contrib import admin  from django.urls import path, include  urlpatterns = [      path('admin/', admin.site.urls),      path('', include('appname.urls'))  ] | |
| Then RUNSERVER |  | |

# Migrate

|  |  |
| --- | --- |
| In the Project Folder Run a Command through CMD | |
| python manage.py migrate |  |

# Creating Templates Folder

|  |  |
| --- | --- |
| In the Base Directory Create a Folder Called ***templates*** |  |
| In the *projectname/projectname/settings.py* Edit the Following Code | |
| FIRST (Import the Important Libs) | import os |
| Then Edit the Settings Required  FROM |  |
| TO | 'DIRS': [os.path.join(BASE\_DIR, 'templates')], |
| Then Create a File in the ***templates*** Folder Called *index.html* fill it with Default HTML Code for Now: |  |

# Print the Contents of a HTML File

|  |  |
| --- | --- |
| Lets Print The Contents of *index.html* | |
| First Let’s go to  *projectname/appname/views.py*  and Create a Function to Call the HTML File | def HTMLPage(request):      '''      #This Syntax Can Also be Used.      template = loader.get\_template('index.html')      return HttpResponse(template.render())      '''      return render(request,'index.html') |
| and then Call the Function to  *projectname/appname/urls.py* | path('html/', views.HTMLPage, name = "HTML Page"), |
| add Something to the HTML Page  *projectname/templates/index.html* | <body>      This is the Contents of the HTML Page called index.html  </body> |
| Run the Server  & Go To  <http://127.0.0.1:8000/html/> |  |

# Data Communication

## Call Data to the Template

|  |  |
| --- | --- |
| In the File *projectname/appname/views.py* | |
| FROM | def HTMLPage(request):      '''      #This Syntax Can Also be Used.      template = loader.get\_template('index.html')      return HttpResponse(template.render())      '''      return render(request,'index.html') |
| TO | def HTMLPage(request):      return render(request,'index.html', {'KEY':'VALUE'}) |
| In the File *projectname/templates/index.html* | |
| FROM | <body>      This is the Contents of the HTML Page called index.html  </body> |
| TO | <body>      We are Calling <b><i><u>{{KEY}}</u></i></b> to the HTML Page.  </body> |
| Result Output |  |

## Send List Data to the Template

|  |  |
| --- | --- |
| In the File *projectname/appname/views.py* | |
| FROM | return render(request,'index.html', {'KEY': 'VALUE'}) |
| TO | List = ["Value1", "Value2", "Value3", "Value4", "Value5"]      return render(request,'index.html', {'KEY': List}) |
| In the File *projectname/templates/index.html* | |
| FROM | We are Calling <b><i><u>{{KEY}}</u></i></b> to the HTML Page. |
| TO | <ol type="1">  <lh>List Heading:</lh>             {% for x in KEY %}                  <li>{{x}}</li>                  {% endfor %}  </ol> |
| Result Output |  |

## Creating Themes

|  |  |
| --- | --- |
| In *projectname/appname/models.py* Create a Class Containing a Data Template | |
| FROM | from django.db import models  # Create your models here. |
| TO | from django.db import models  # Create your models here.  class theme:      dataTypeNameForStr:str      dataTypeNameForInt:int |
| In the File *projectname/appname/views.py* | |
| FIRST  Import the Class from the *models.py* File | from appname.models import theme |
| FROM | List = ["Value1", "Value2", "Value3", "Value4", "Value5"]      return render(request,'index.html', {'KEY': List}) |
| TO  ( Import the Class as an Object ) | Object1 = theme()      Object1.dataTypeNameForStr = "Jonathan"      Object1.dataTypeNameForInt = 1805      Object2 = theme()      Object2.dataTypeNameForStr = "Chacko"      Object2.dataTypeNameForInt = 2002      MasterListObject = [Object1, Object2]      return render(request,'index.html',{'KEY':MasterListObject}) |
| In the File *projectname/templates/index.html* | |
| FROM | <ol type="1">  <lh>List Heading:</lh>             {% for x in KEY %}                  <li>{{x}}</li>                  {% endfor %}  </ol> |
| TO | <table border = '1'>      {% for x in KEY %}      <tr>          {{x.dataTypeNameForStr}}          <td>              {{x.dataTypeNameForInt}}          </td>      </tr>      {% endfor %}  </table> |
| Result Output |  |

## Form Input

|  |  |
| --- | --- |
| In the *projectname/appname/views.py* | |
| FROM | Object1 = theme()      Object1.dataTypeNameForStr = "Jonathan"      Object1.dataTypeNameForInt = 1805      Object2 = theme()      Object2.dataTypeNameForStr = "Chacko"      Object2.dataTypeNameForInt = 2002      MasterListObject = [Object1, Object2]      return render(request,'index.html', {'KEY': MasterListObject}) |
| TO | if request.method == 'POST':          num1 = request.POST['num1'] # input1 = request.GET['num1']          num2 = request.POST['num2'] # input2 = request.GET['num2']          if 'add' in request.POST:              result = int(num1) + int(num2)              return render(request,'index.html',{'KEY':result})      return render(request,'index.html') |
| In the File *projectname/templates/index.html* | |
| FROM | <table border = '1'>      {% for x in KEY %}      <tr>          {{x.dataTypeNameForStr}}          <td>              {{x.dataTypeNameForInt}}          </td>      </tr>      {% endfor %}  </table> |
| TO | <form method="POST">      {% csrf\_token %}      <label for="num1">Enter Any Number: </label>          <input type="text" name="num1"/>      <label for="num2">Enter Any Number: </label>          <input type="text" name="num2"/>      <br/>      <button type="submit" name="add">+</button>  </form>  <br><br><br>  <h3>The Result is = </h3><br>  <h4>{{KEY}}</h4> |
| Result Output |  |
| Django's login form is returned using the POST method, in which the browser bundles up the form data, encodes it for transmission, sends it to the server, and then receives back its response.  GET, by contrast, bundles the submitted data into a string, and uses this to compose a URL. | |

## SQL

### Initialization

|  |  |
| --- | --- |
| In the *projectname/projectname/settings.py* Edit the Following Code: | |
| FROM |  |
| TO | INSTALLED\_APPS = [      'django.contrib.admin',      'django.contrib.auth',      'django.contrib.contenttypes',      'django.contrib.sessions',      'django.contrib.messages',      'django.contrib.staticfiles',      'appname',  ] |

### Migrations of Models

|  |  |
| --- | --- |
| Create a Class Containing a Model in In *projectname/appname/models.py* | class tablename(models.Model):      dtstr1 = models.CharField(max\_length=255)      dtstr2 = models.CharField(max\_length=255)      #This is to Sync SQLite3 |
| py manage.py makemigrations appname  py manage.py migrate |  |
| These are the Files Created: |  |
| This Should be the Content of the  *projectname/appname/migrations/0001\_initial.py* |  |
| Another Way to Check what the Migration Executed is:  py manage.py sqlmigrate appname 0001 |  |
| NOTE: | |
| [***https://sqliteonline.com/***](https://sqliteonline.com/) | ***USE THIS SITE TO CHECK THE SQLITE DATABASE.*** |
| Incase of Error in Creation of SQL Table Run  python manage.py migrate --fake appname zero |  |
| Then Rerun all the Above CMD Commands for Migrations | |

### Read &or Write Data

|  |  |
| --- | --- |
| In the *projectname/appname/views.py* | |
| FROM | if request.method == 'POST':          num1 = request.POST['num1'] # input1 = request.GET['num1']          num2 = request.POST['num2'] # input2 = request.GET['num2']          if 'add' in request.POST:              result = int(num1) + int(num2)              return render(request,'index.html',{'KEY':result})      return render(request,'index.html') |
| TO | if request.method == 'POST':          if request.POST.get('data1') and request.POST.get('data2'):              post = tablename()              post.dtstr1 = request.POST.get('data1')              post.dtstr2 = request.POST.get('data2')              post.save()              return render(request, 'index.html')      else:          objectName = tablename.objects.all()          return render(request, 'index.html', {'KEY': objectName}) |
| In the File *projectname/templates/index.html* | |
| FROM | <form method="POST">      {% csrf\_token %}      <label for="num1">Enter Any Number: </label>          <input type="text" name="num1"/>      <label for="num2">Enter Any Number: </label>          <input type="text" name="num2"/>      <br/>      <button type="submit" name="add">+</button>  </form>  <br><br><br>  <h3>The Result is = </h3><br>  <h4>{{KEY}}</h4> |
| TO | <h1>Create a Post </h1>  <form action="" method="POST">      {% csrf\_token %}      Title: <input type="text" name="data1"/><br/>      Content: <br/>      <input type="text" name="data2"/>      <input type="submit" value="Post"/>  </form>  <br><br>  <table border="1">      <tr>          <th>Data 1</th>          <th>Data 2</th>      </tr>      {% for x in KEY %}      <tr>          <td>{{ x.dtstr1 }}</td>          <td>{{ x.dtstr2 }}</td>      </tr>      {% endfor %}  </table> |
| Result Output | 1 2  ---  3 |

# Static Folder

## Initialization

|  |  |
| --- | --- |
| Create a ***static*** Folder in the Base Directory |  |
| & Add the Resources Folders in it. |  |
| In the *projectname/projectname/settings.py* Edit the Following Code | |
| FROM |  |
| TO | # Static files (CSS, JavaScript, Images)  # https://docs.djangoproject.com/en/4.1/howto/static-files/  STATIC\_URL = '/static/'  STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static')]  STATIC\_ROOT = os.path.join(BASE\_DIR, 'assets') |

## Create Resources Folders

### CSS

|  |  |
| --- | --- |
| Create a CSS File called ***first.css*** then | |
| open *projectname\static\css\first.css* & Add Script to It | body {      background-color: lightblue;      font-family: verdana;    } |

### JS

### IMG

### Fonts

## Collect Static

|  |
| --- |
| python manage.py collectstatic |
|  |
|  |

## Connect to HTML Page

|  |  |
| --- | --- |
| Open *projectname\template\index.html* & Edit It | |
| FROM | <!DOCTYPE html>  <html lang="en">  <head> |
| TO | {% load static %}  <!DOCTYPE html>  <html lang="en">      <link rel="stylesheet" href="{% static '\css\first.css' %}">  <head> |
| Result Output |  |
| ***Note: We Can Do the Same for The Other Resources Not Only CSS Files.*** | |

# Project

|  |  |
| --- | --- |
| First Lets Gets a Free Template- | <https://templatemo.com/tag/one-page> |
| Add All the Resources to the ***static*** Folder |  |
| Make Sure the Content in *projectname/projectname/settings.py* Make Sure All the Settings are Completed | |
| python manage.py collectstatic |  |
| These Files will be Created in the Assets Folder. |  |
| THEN FIRST | Copy All the HTML Files into the ***templates*** Folder |
| Add the Line in the First Line of all the ***html files*** | {% load static %} |
| Change All The Links of the Resources Folder to route through ***static*** inall the ***html files*** | |
| ***EXAMPLE CHANGE***  FROM | <link rel="stylesheet" href="projectname/static/css/bootstrap.min.css"> |
| TO | <link rel="stylesheet" href="{% static 'css/bootstrap.min.css' %}"> |
| DO THIS TO ALL THE LINKS WHICH ARE INTERNAL (NOT EXTERNAL .com’S) | |
| FINALLY | Create the Loops in the ***HTML Files*** and Load all the Data Through Templates Created Through models.py (Do this to Reduce the Number of Lines in the HTML Files). |

# Refer

## HTML

<https://www.w3schools.com/html/default.asp>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/js/default.asp>

<https://www.w3schools.com/howto/default.asp>

<https://www.w3schools.com/bootstrap/bootstrap_ver.asp>

## SQL

<https://www.w3schools.com/sql/default.asp>