**Install Python, PIP and Django on Windows**

**1.Install python latest version**

-check add to path

-check all user

-select C drive(Browse)

**2.python --version ->check on cmd**

-exit() ->to come out

++++++++++++++++++++++++++++++++++++++++++++++++++

**Install django on cmd (it install globally)**

1.pip install django

2.open new cmd

-pip freeze

+++++++++++++++++++++++++++++++++++++++++++++++++++

**Start project on django**

-create one folder for project

-go inside that folder from cmd or visual studio

-run command for new project->(django-admin startproject myfirstproject)

++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to run server(django project)**

-pyhon manage.py runserver

-http://127.0.0.1:8000/

-Ctlr+C to stop the server

-want to runserver other port command will->python manage.py runserver 4000

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Create folder for manage html pages**

C:\python\myfirstpro

-templates

-static

-media

-manage.py is main file to run your application

-setting.py is details of all configuration like database, templates ,static folder

-urls.py manage all urls

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to Migrate Default Migration(db.sqlite3)data**

-Django used default database qlite3

-Install Database Browser for SQLite in Windows

-python manage.py Migrate

with help of above command it default data is present in db.sqlite3

before it 0kb now its 128kb

-first install dbsqlite to see all data

-admin panel

http://127.0.0.1:8000/admin

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to Create Superuser**

You can log in on the admin panel once the superuser is created in the admin application in Django.

The command that is used to create a superuser in Django is:

python manage.py createsuperuser

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to Use URLs(Route) & Views in Django**

http://127.0.0.1:8000/

http://127.0.0.1:8000/about

http://127.0.0.1:8000/admin/

http://127.0.0.1:8000/homepage

http://127.0.0.1:8000/blog-single

**inside urls.py file load views folder**

from myfirstpro import views

**Inside urlpattern load urls and views**

urlpatterns = [

path('admin/', admin.site.urls),

path('', views.indexpage),

path('blog-single', views.blog),

path('homepage', views.homePage),

path('aboutus/', views.aboutUs),

]

path(urls,load view with function name)

**Views:**

base on URLs call view(function).

with help of http method prepare and return response data(HTML)

from django.http import HttpResponse

from django.shortcuts import render

def aboutUs(request):

return HttpResponse("<h1>Welconme to MCIT</h1>")

def homePage(request):

return render(request,"home.html")

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Creating a Dynamic URL in Django:(passing parameter(int,slug,str) on URLs)**

int:1,2,44,565656..........

str:blog,homepage,aboutus.............

slug:about-us,new-post-data,data-dt,post-menu-data-list..........

path('Course-details/<int:courseid>', views.CourseDetails),

path('Course-string/<str:coursestr>', views.Coursestring),

path('Course-slug/<slug:courseslug>', views.Courseslugdata),

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to Render an HTML Template as Response**

go inside setting.py

Inside TEMPLATES set this directory

'DIRS': [BASE\_DIR,"templates"],

inside urls.py create urls and views

path('homepage', views.homePage),

views(render take two parameter first request and web page)

def homePage(request):

return render(request,"index.html")

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to Pass Data From Django View to Template**

sending data on an HTML page without the use of a loop. Render function will use three functions to pass data from a Django view to template.

**views:**

def homePage(request):

data={

'title':'data render from home pae function',

'data':'welcome to MCIT',

'CourseList':['php','java','django','python'],

'numbers':[20,30,40,50,60,70,80],

'marks':[],

'student\_details':[

{'name':'seema','phone':989898998},

{'name':'rahul','phone':90098998}

]

}

return render(request,"home.html",data)

home.html ->Render data

<p>{{title}}<br />{{data}}<br />{{CourseList}}</p>

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**CSS, JavaScript & Images in Django**

**Managing CSS, JavaScript & Images static files in Django**

Inside setting.py place below code

STATICFILES\_DIRS = [

BASE\_DIR / "static"

]

Add path in html file

<link rel="icon" href="/static/img/favicon.png" />

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**How to create common header and footer in Django.**

Headers and footers are the same on every page and they should be on every page on the program.

Include is a tag in Django that is used in HTML. you can add a header and footer on every page.

{% include 'header.html' %}

.

.homepage

.

{% include 'footer.html' %}

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**URL Template Tags**

<li><a href="contact">Contact Us</a></li>

<li><a href="blog-single">Blog Details</a></li>

<li><a href="index">Home Page 1</a></li>

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Page Not Found when hit any URL(which is not in url list):Follow Step**

DEBUG = False

ALLOWED\_HOSTS = ['\*'] //work on any server

**Inside url.py**

create handler(project\_name.view.function\_name)

handler404='myfirstpro.views.error\_404\_view'

**Inside views**

def error\_404\_view(request,exception):

return render(request,'404.html')

**Redirect to Home page:404.html**

<a class="btn btn-primary btn-lg" href="index">Got to Home Page</a>

/++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**What is Model in Django**

The function that is used in this project are:

python manage.py startapp service

python manage.py makemigrations

python manage.py migrate

Models are used to make topics and headings dynamic on the site.

https://docs.djangoproject.com/en/5.0/topics/db/models/

**python manage.py startapp service**

**Create Model(service/models.py)**

class Service\_Member(models.Model):

first\_name = models.CharField(max\_length=30)

last\_name = models.CharField(max\_length=30)

service\_des=models.TextField()

**go inside setting.py**

add service(folder name) name in Installed\_APP=[]

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'service'

]

**Run migration command**

python manage.py makemigrations

migration create one file (0001\_initial.py)

**create table in database by this command:**

python manage.py migrate

**How to add model in admin panel**

go inside /service/admin.py

from service.models import Service\_Member

**from filename of model import class name of model**

make class, add all list to show in admin panel:

all field name:

from service.models import Service\_Member

class ServiceAdmin(admin.ModelAdmin):

    list\_display=('first\_name','last\_name','service\_des')

admin.site.register(Service\_Member, ServiceAdmin)

**admin.site.register(model\_class\_name, admin\_class\_name)**

https://docs.djangoproject.com/en/5.0/ref/contrib/admin/

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Get Data from Table in Django template:**

Go inside views:main folder(project folder)

import model:

from service.models import Service\_Member

def service(request):

serviceData=Service\_Member.objects.all()

return render(request,"service.html",alllist)

Display data in html page:

{% for n in serviceData %}

<p>

<span>{{n.first\_name}}</span>&nbsp; <span>{{n.last\_name}}</span>&nbsp;

<span>{{n.service\_des}}</span>

</p>

{% endfor %}

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**TinyMCE Integration with service app(it is editor)**

Install: tinymce for editor.

pip install django-tinymce

(https://pypi.org/project/django-tinymce/)

**Add tinymce to INSTALLED\_APPS in settings.py for your project:**

INSTALLED\_APPS = (

...

'tinymce',

)

create new app

python manage.py startapp news

create model: Add this(news app)

from tinymce.models import HTMLField

class News(models.Model):

title = models.CharField(max\_length=100)

description=HTMLField()

add news in install app:

INSTALLED\_APPS = [

'news',

'tinymce',

]

**Add in admin.py(news app)**

from django.contrib import admin

from news.models import News

# from filename.model import class nameof model

# Register your models here.

class NewsAdmin(admin.ModelAdmin):

list\_display=('title','descriptions')

admin.site.register(News, NewsAdmin)

**Now run command to create model and in admin panel**

python manage.py makemigrations

python manage.py migrate

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Marquee Tag for Display News in Django**

index.html

<marquee>

{% for news in latestnews%}

{{news.title}}::

{{news.descriptions}}

{% endfor %}

</marquee>

**\*\*\*\*\*\*\*\*\*\*\*\*\*views.py add model to dynamic content:**

from news.models import News

def index(request):

newsdata=News.objects.all()

print(newsdata)

latestnews={

'latestnews':newsdata,

}

return render(request,"index.html", latestnews)

**\*\*\*\*\*\*\*\*\*add field in model.py**

from tinymce.models import HTMLField

# Create your models here.

class News(models.Model):

title = models.CharField(max\_length=100)

descriptions=HTMLField()

++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Implement POST Method Form with CSRF Token**

<form class="form" method="post">

{% csrf\_token %}

++++++++++++++++++++++++++++++++++++++++++++++++

**Save Form data in database:**

Cross-Site Request Forgery (CSRF) is an attack that forces authenticated users to submit a request to a Web application against which they are currently authenticated.

Csrf requires a secret key by default, it uses the Flask app's Secret Key.

By using safe and secure HTTP, your data also stay secure. The post method goes through HTTP headers, which are known as QUERY\_STRING in Django.

If you have applied CSRF token form cookies, you can stop the hackers and save your website from potential threats. Django will check the information if you are using CSRF in the post method.

**step:1: create app**

python manage.py startapp enquiryform

**step2 :create model**

from django.db import models

# Create your models here.

class saveEnquiry(models.Model):

name = models.CharField(max\_length=50)

email = models.CharField(max\_length=50)

phone = models.CharField(max\_length=50)

subject = models.CharField(max\_length=50)

message = models.TextField()

**step3: create in admin panel(admin.py)**

from enquiryform.models import saveEnquiry

# Register your models here.

class contactEnquiry(admin.ModelAdmin):

list\_display = ('name', 'email', 'phone', 'subject', 'message')

admin.site.register(saveEnquiry, contactEnquiry)

**step4:create form action and generate csrf\_token(contact.html)**

<form class="form" method="post" action="{% url 'contactEnquiry' %}">

{% csrf\_token %}

**step5:set path in url**

path('contactEnquiry', views.contactEnquiry, name="contactEnquiry"),

step6:Create function in view function

from enquiryform.models import saveEnquiry

def contactEnquiry(request):

if request.method == "POST":

fname = request.POST.get('fname')

email = request.POST.get('email')

phone = request.POST.get('phone')

subject = request.POST.get('subject')

message = request.POST.get('message')

print(fname)

en = saveEnquiry(name=fname, email=email, phone=phone,

subject=subject, message=message)

en.save()

return render(request, "contact.html")

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Call template click on menu(home,service,blog)=>URL Template Tags**

set same path in html page /service -> it will take root path(/)

path('service', views.service),

<li><a href="/service">Services </a></li>

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Create Enquiry form**

create form using post method with CSRF in Django.

Create app enquiryform:

create Model:

create admin:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Create app enquiryform:**

python manage.py startapp enquiryform

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Create Model(enquiryform/models.py)**

class saveEnquiry(models.Model):

name = models.CharField(max\_length=50)

email = models.CharField(max\_length=50)

phone = models.CharField(max\_length=50)

subject = models.CharField(max\_length=50)

message = models.TextField()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Run migration command**

python manage.py makemigrations

migration create one file (0001\_initial.py) see in migrations folder

**create table in database by this command:**

python manage.py migrate

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**create admin:**

from enquiryform.models import saveEnquiry

# Register your models here.

class contactEnquiry(admin.ModelAdmin):

list\_display = ('name', 'email', 'phone', 'subject', 'message')

admin.site.register(saveEnquiry, contactEnquiry)

After that save, go to admin panel and refresh

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Contact.html=>(create form using post method with CSRF in Django.)**

<!--/ Start Form -->

<form class="form" method="post" action="{% url 'contactEnquiry' %}">

{% csrf\_token %}

<div class="row">

<div class="col-lg-6">

<div class="form-group">

<input type="text" name="fname" placeholder="Name" required=""/></div>

........

.........</div></div></div>

</form>

<!--/ End Form -->

**url.py**

path('contactEnquiry', views.contactEnquiry, name="contactEnquiry"),

**view.py**

from enquiryform.models import saveEnquiry

def contactEnquiry(request):

if request.method == "POST":

fname = request.POST.get('fname')

email = request.POST.get('email')

phone = request.POST.get('phone')

subject = request.POST.get('subject')

message = request.POST.get('message')

print(fname)

en = saveEnquiry(name=fname, email=email, phone=phone,

subject=subject, message=message)

en.save()

return render(request, "contact.html")

+++++++++++++++++++++++++++++++++++++++++++++++++++

**Upload section(create blog)**

Upload section(create blog)

**create app blog**

python manage.py startapp blog

**Create Model(blog/models.py):**

from tinymce.models import HTMLField

from autoslug import AutoSlugField

# Create your models here.

class Blog(models.Model):

blog\_title = models.CharField(max\_length=100)

blog\_desc = HTMLField()

blog\_slug = AutoSlugField(

populate\_from='blog\_title', unique=True, null=True, default=None)

blog\_image = models.FileField(

upload\_to="blog/", max\_length=250, null=True, default=None)

go inside setting.py

add blog app(folder name) name in Installed\_APP=[]

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'blog'

]

**Run migration command**

python manage.py makemigrations

migration create one file (0001\_initial.py) want to see go inside migrations folder

**create table in database by this command:**

python manage.py migrate

**How to add model in admin panel**

go inside /blog/admin.py

from django.contrib.admin.sites import site

from blog.models import Blog

(from filename of model import class name of model)

class BlogAdmin(admin.ModelAdmin):

list\_display = ('blog\_title', 'blog\_desc', 'blog\_image')

admin.site.register(Blog, BlogAdmin)

admin.site.register(model\_class\_name, admin\_class\_name)

**refresh the admin panel**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Follow steps for upload:**

Step1:create media folder, same root of manage.py

Step2:go inside setting.py

MEDIA\_ROOT = BASE\_DIR / "media"

MEDIA\_URL = "/media/"

Step3: go inside url.py, import

from django.conf import settings

from django.conf.urls.static import static

Add these line at the end

if settings.DEBUG:

urlpatterns += static(settings.MEDIA\_URL,

document\_root=settings.MEDIA\_ROOT)

**Now got to admin panel and refresh you can see upload file**

How to display image in forntend:

**url.py**

path('blogdata', views.blogdata),

**view.py**

def blogdata(request):

blogData = Blog.objects.all()

data = {

'blogData': blogData

}

return render(request, "blog.html", data)

**blog.html**

{% for blog in blogData %}

<h1>

<a href="/blogdetails/{{blog.blog\_slug}}"> {{blog.blog\_title}} </a>

</h1>

<p style="background-color: yellow">{{blog.blog\_desc | safe}}</p>

<p><img src="/media/{{blog.blog\_image}}" /></p>

{% endfor %}

+++++++++++++++++++++++++++++++++++++++++++++

**sending mail(enquiry)**

**sending mail(enquiry) from localhost**

https://docs.djangoproject.com/en/5.0/topics/email/

you need to configure an SMTP

need to give access to django

go to gmail account right side click account

search less secure app ->set allow

go to setting.py

EMAIL\_HOST = 'smtp.gmail.com'

EMAIL\_PORT = 587

EMAIL\_HOST\_USER = 'aaruaj14@gmail.com'

# 'less secure app' keep on gmail this generate password instance of gmail password

# <https://docs.djangoproject.com/en/4.2/topics/email/>

EMAIL\_HOST\_PASSWORD = 'fedk ejfl jwhw qbis'

EMAIL\_USE\_TLS = True

views.py

from django.core.mail import send\_mail

def sendmail(request):

send\_mail('testing mail', 'here is the message', 'aaruaj14@gmail.com',

['shejalesapna21@gmail.com'], fail\_silently=False,)

return render(request, "contactform.html")

url.py

path('sendmail/', views.sendmail),

https://docs.djangoproject.com/en/5.0/topics/email/

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Email Multi Alternatives Function**

views.py

from django.core.mail import send\_mail, EmailMultiAlternatives

def email\_multi\_alternatives(request):

subject = 'email\_multi\_alternatives testing'

from\_email = 'aaruaj14@gmail.com'

msg = '<p>welcome to <b>aaru</b></p>'

to = 'shejalesapna21@gmail.com'

msg = EmailMultiAlternatives(subject, msg, from\_email, [to])

msg.content\_subtype = "html"

msg.send()

return render(request, "contactform.html")

url.py

path('email\_multi\_alternatives/', views.email\_multi\_alternatives)

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**create mysq database**

connection

 'default': {

        'ENGINE': 'django.db.backends.mysql',

        'NAME': 'doctorweb',

        'PASSWORD' : "",

        'USER': 'root',

        'HOST' : 'localhost',

        'PORT': '3306',

    }

Phpmysql install

Start=> localhost/phpmyadmin/

Migrations(( python manage.py migrations)

Migrate ( python manage.py migrate)

Createsuperuser ( python manage.py createsuperuser)

Run->python manage.py runserver

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Add Pagination in Django**

Import library

from django.core.paginator import Paginator

#Paginator take two parameter first where to apply and second limit Paginator(serviceData, 3)

serviceData=Service\_Member.objects.all()

paginator = Paginator(serviceData, 3)

page\_number = request.GET.get('page')

pagedatafinal = paginator.get\_page(page\_number)

totalpage = pagedatafinal.paginator.num\_pages

alllist={

'pagedatafinal': pagedatafinal,

'lastpage': totalpage,

'totalpagelist': [n+1 for n in range(totalpage)]

}

return render(request,"service.html",alllist)

**service.html(template)**

<div class="pagination" style="margin-bottom: 20px">

{% if pagedatafinal.has\_previous %}

<a href="/service/?page={{pagedatafinal.previous\_page\_number}}"

>&laquo;</a

>

{% endif %} {% for n in totalpagelist %}

<a href="/service/?page={{n}}" style="padding: 30px">{{n}}</a>

{% endfor %} {% if pagedatafinal.has\_next %}

<a href="/service/?page={{lastpage}}">last page</a>

<a href="/service/?page={{pagedatafinal.next\_page\_number}}">&raquo;</a>

{% endif %}

</div>

+++++++++++++++++++++++++++++++++++++++++++++++++++++

**Login with student**

https://medium.com/@sinturana250/login-register-logout-and-create-a-student-with-crud-in-django-project-chapter-12-e8498effdd26