Name:-

PRN: -

Class:- Ty (btech)

Batch:

PRACTICAL: 5

AIM: Design a django project for “Django Blog”

THEORY

INTRODUCTION:

In this practical we are going to build the blog application using django that will allow user to create, edit, delete their posts. The homepage will list of all blog posts, and there will be dedicated detail page for each individual post.

STARTING THE PROJECT

To initiate a project of Django on Your PC, open Terminal and Enter the following command

```
django-admin mysite
```

A New Folder with the name projectName will be created. To enter in the project using the terminal enter command

```
cd mysite
```

Now let’s run the server and see everything is working fine or not. To run the server, type the below command in the terminal.

```
python manage.py runserver
```

Creating an app named quiz

```
python manage.py startapp blog
```

- **Settings.py:**

  Add your app name in settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
```

```
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'home',
    'django.contrib.humanize'
]


DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

- **Urls.py:**

Now add your apps urls in the project

```
from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('home.urls')),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

- **Models.py:**

```
 from django.db import models
from django.contrib.auth.models import User
from django.urls import reverse
from django.utils.timezone import now
from django.db.models.signals import post_save
from django.dispatch import receiver

class Profile(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, blank=True,
null=True)
    image = models.ImageField(upload_to="profile_pics", blank=True, null=True)
    bio = models.TextField(blank=True, null=True)
    phone_no = models.IntegerField(blank=True, null=True)
    facebook = models.CharField(max_length=300, blank=True, null=True)
    instagram = models.CharField(max_length=300, blank=True, null=True)
    linkedin = models.CharField(max_length=300, blank=True, null=True)

    def __str__(self):
        return str.user.username
```

```
class BlogPost(models.Model):
    title=models.CharField(max_length=255)
    author= models.ForeignKey(User, on_delete=models.CASCADE)
    slug=models.CharField(max_length=130)
    content=models.TextField()
    image = models.ImageField(upload_to="profile_pics", blank=True, null=True)
    dateTime=models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return str(self.author) +  " Blog Title: " + self.title

    def get_absolute_url(self):
        return reverse('blogs')


class Comment(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    content = models.TextField()
    blog = models.ForeignKey(BlogPost, on_delete=models.CASCADE)
       parent_comment  =  models.ForeignKey('self',  on_delete=models.CASCADE,
null=True, blank=True)
    dateTime=models.DateTimeField(default=now)

    def __str__(self):
        return self.user.username +  " Comment: " + self.content


@receiver(post_save,  sender=User)
def create_user_profile(sender, instance, created, **kwargs):
    if created:
        Profile.objects.create(user=instance)
```

- **Admin.py:**

Here, we are registering our model to the admin site, so that we can update or access the database from the admin panel also. But we need a superuser to access the admin site.

```
from django.contrib import admin
from .models import *

admin.site.register(BlogPost)
admin.site.register(Comment)
admin.site.register(Profile)
```

Migrate the database and create the super user

```
Python manage.py makemigrations
Python manage.py migrate
Python manage.py createsuperuser
```

- **Apps.py:**

```
from django.apps import AppConfig

class HomeConfig(AppConfig):
    name = 'home'
```

- **Views.py:**

```
from django.shortcuts import render, redirect, HttpResponse
from django.contrib.auth.models import User
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.views.generic.edit import UpdateView
from django.contrib import messages
from .models import BlogPost, Comment, Profile
from .forms import ProfileForm, BlogPostForm
from .models import Profile

def blogs(request):
    posts = BlogPost.objects.all().order_by('-dateTime')
    return render(request, "blog.html", {'posts': posts})

def blogs_comments(request, slug):
    post = BlogPost.objects.filter(slug=slug).first()
    comments = Comment.objects.filter(blog=post)
    if request.method == "POST":
        user = request.user
        content = request.POST.get('content', '')
        comment = Comment(user=user, content=content, blog=post)
        comment.save()
        return render(request, "blog_comments.html", {'post': post, 'comments': comments})

def Delete_Blog_Post(request, slug):
    post = BlogPost.objects.get(slug=slug)
    if request.method == "POST":
        post.delete()
        return redirect('/')
    return render(request, 'delete_blog_post.html', {'posts': post})

def search(request):
    if request.method == "POST":
```

```
        searched = request.POST['searched']
        blogs = BlogPost.objects.filter(title__icontains=searched)
        return render(request, "search.html", {'searched': searched, 'blogs': blogs})
    return render(request, "search.html")

@login_required(login_url='/login')
def add_blogs(request):
    if request.method == "POST":
        form = BlogPostForm(request.POST, request.FILES)
        if form.is_valid():
            blogpost = form.save(commit=False)
            blogpost.author = request.user
            blogpost.save()
            alert = True
            return render(request, "add_blogs.html", {'form': form, 'alert': alert})
    else:
        form = BlogPostForm()
    return render(request, "add_blogs.html", {'form': form})

class UpdatePostView(UpdateView):
    model = BlogPost
    template_name = 'edit_blog_post.html'
    fields = ['title', 'slug', 'content', 'image']

def user_profile(request, myid):
    post = BlogPost.objects.filter(id=myid)
    return render(request, "user_profile.html", {'post': post})

# before (wrong)
def Profile(request):
    return render(request, "profile.html")

# after (correct)
def profile_view(request):
    return render(request, "profile.html")

def edit_profile(request):
    try:
        profile = request.user.profile
    except Profile.DoesNotExist:
        profile = Profile.objects.create(user=request.user)

    if request.method == 'POST':
        form = ProfileForm(request.POST, instance=profile)
        if form.is_valid():
            form.save()
            return redirect('profile')
    else:
        form = ProfileForm(instance=profile)
```

```
    return render(request, 'edit_profile.html', {'form': form})
def Register(request):
    if request.method == "POST":
        username = request.POST['username']
        email = request.POST['email']
        first_name = request.POST['first_name']
        last_name = request.POST['last_name']
        password1 = request.POST['password1']
        password2 = request.POST['password2']

        if password1 != password2:
            messages.error(request, "Passwords do not match.")
            return redirect('/register')

        user = User.objects.create_user(username, email, password1)
        user.first_name = first_name
        user.last_name = last_name
        user.save()
        return render(request, 'login.html')
    return render(request, "register.html")

def Login(request):
    if request.method == "POST":
        username = request.POST['username']
        password = request.POST['password']

        user = authenticate(username=username, password=password)

        if user is not None:
            login(request, user)
            messages.success(request, "Successfully Logged In")
            return redirect("/")
        else:
            messages.error(request, "Invalid Credentials")
            return render(request, "login.html")
    return render(request, "login.html")

def Logout(request):
    logout(request)
    messages.success(request, "Successfully logged out")
    return redirect('/login')
```

- **Urls.py:**
  Add the urls.py file in your app folder

```
from django.urls import path
from . import views
from .views import UpdatePostView
```

```
urlpatterns = [
#     blogs
    path("", views.blogs, name="blogs"),
    path("blog/<str:slug>/", views.blogs_comments, name="blogs_comments"),
    path("add_blogs/", views.add_blogs, name="add_blogs"),
                        path("edit_blog_post/<str:slug>/",              UpdatePostView.as_view(),
name="edit_blog_post"),
                         path("delete_blog_post/<str:slug>/",              views.Delete_Blog_Post,
name="delete_blog_post"),
    path("search/", views.search, name="search"),


#     profile
    path("profile/", views.profile_view, name="profile"),
    path("edit_profile/", views.edit_profile, name="edit_profile"),
    path("user_profile/<int:myid>/", views.user_profile, name="user_profile"),

#    user authentication
    path("register/", views.Register, name="register"),
    path("login/", views.Login, name="login"),
    path("logout/", views.Logout, name="logout"),
]
```

- **base.html:**
  Create a base.html in templates folder.

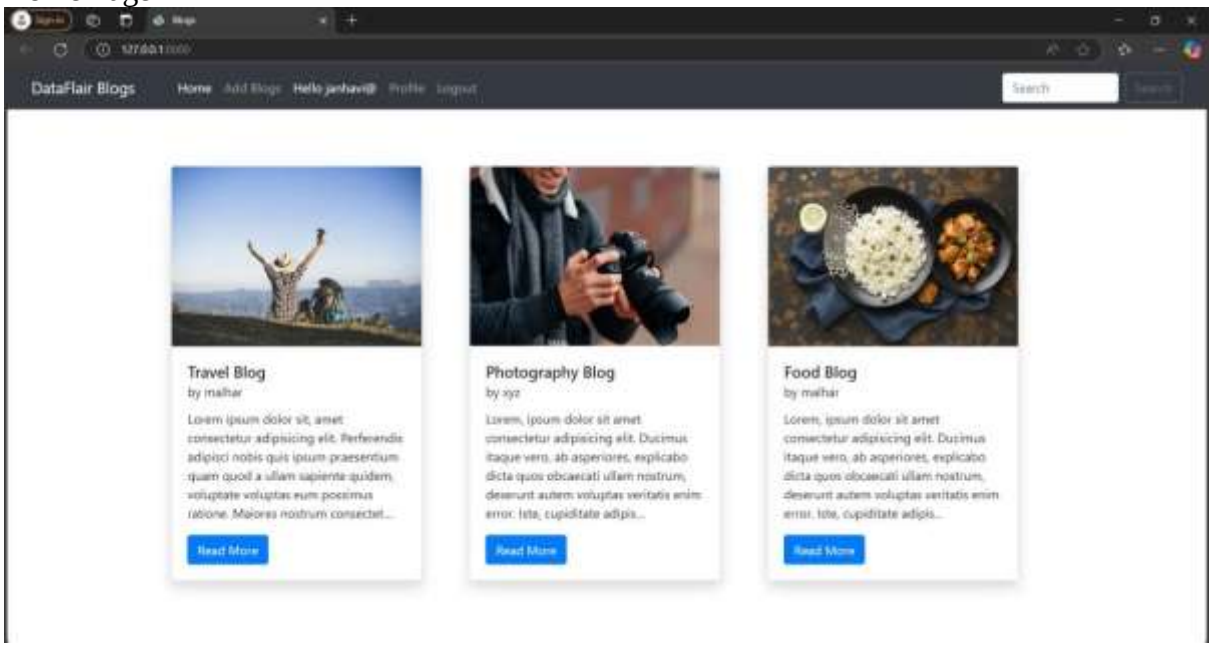
```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
       <link  href="//maxcdn.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">

    <!-- Bootstrap CSS -->
                                                                                                <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"                                                                integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmL
ASjC" crossorigin="anonymous">
                                                    <link                                rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS
6JXm" crossorigin="anonymous">

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

```
 <style>    {% block css %} {% endblock %} </style>
 </head>
 <body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark sticky-top">
     <div class="container-fluid">
      <a class="navbar-brand" href="#">DataFlair Blogs</a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <div class="container mx-3">
        <ul class="navbar-nav me-auto mb-2 mb-lg-0">
          <li class="nav-item">
           <a class="nav-link {% block homeactive %} {% endblock homeactive %}" aria-current="page" href="/">Home</a>
          </li>
          {% if user.is_authenticated %}
          <li class="nav-item">
                <a class="nav-link {% block add_blogs_active %} {% endblock add_blogs_active %}" href="/add_blogs/">Add Blogs</a>
          </li>
          <li class="nav-item">
            <a class="nav-link active" href="#">Hello {{request.user}}</a>
          </li>
          <li class="nav-item">
            <a class="nav-link {% block profileactive %} {% endblock profileactive %}" href="/profile/">Profile</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/logout/">Logout</a>
          </li>
          {% else %}
          <li class="nav-item">
            <a class="nav-link {% block registeractive %} {% endblock registeractive %}" href="/register/">Register</a>
          </li>
           <li class="nav-item">
            <a class="nav-link {% block loginactive %} {% endblock loginactive %}" href="/login/">Login</a>
          </li>
          {% endif %}
        </ul>
        </div>
        <form class="d-flex" method="POST" action="/search/"> {% csrf_token %}
            <input class="form-control me-2" type="search" placeholder="Search" name="searched" id="searched" aria-label="Search">
                    <button href="/search/" class="btn btn-outline-secondary" type="submit">Search</button>
```

```
          </form>
        </div>
      </div>
    </nav>
    {% block body %}
    {% endblock %}


                                                                    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIax
VXM" crossorigin="anonymous"></script>

                                                    <!--                    <script
src="//maxcdn.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"></script>   -->
<script  src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

   {% block js %}
   {% endblock %}
</body>
</html>
```

OUTPUT:

Home Page



CONCLUSION:

Hence, in this practical we learnt to make blog using django

Sign of Teacher