Project Report

Open source Lab

Prof. incharge : Swati Ringe

**Group Members :** Ryan D’Mello 7930

Aishwarya Sebin 7916

**Topic :** A community based forum website which allows user to ask questions, write answers and also view other people’s answers implemented with Django framework.

**Introduction:**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Some key features of Django framework :

* Ridiculously fast:

Django was designed to help developers take applications from concept to completion as quickly as possible.

* Reassuringly secure:

Django takes security seriously and helps developers avoid many common security mistakes.

* Exceedingly scalable:

Some of the busiest sites on the Web leverage Django’s ability to quickly and flexibly scale.

* Incredibly versatile:

Companies, organizations and governments have used Django to build all sorts of things — from content management systems to social networks to scientific computing platforms.

There are many multinational web services which have their website based on Django. They include :-

* Instagram
* Quora
* Pinterest
* NASA

**Our implementation of Django in our project :**

We have used Django Framework for the working of our website. Django framework creates all the files on its own so we just modify them according to our need.

We modified the following files:-

* views.py – This file is used to create views for templates and other website functionality.
* urls.py – This file does the url mappings i.e. connects your templates with the views.
* settings.py – This is the main file where you define the static urls, media urls, and also the BASE\_DIR for the proper working of the website.
* **models.py - It contains the essential fields and behaviors of the data you’re storing. Generally, each model maps to a single database table.**
* And also we created Templates folder, forms.py, urls.py in app folder, static files folder which contains the pictures and everything else required for the frontend.

Other than Django we have also implemented Css and bootstrap in the front end programming which makes the website look exquisite, very legit and out of this world.

The code written in the above mentioned files are as follows:

**models.py**

from django.db import models

from django.contrib.auth.models import User

from django.db.models.signals import post\_save

from django.utils import timezone

from django.urls import reverse

from django.template.defaultfilters import slugify

# Create your models here.

class UserProfile(models.Model):

user = models.OneToOneField(User,on\_delete=models.CASCADE)

description = models.CharField(max\_length=264, default='')

occupation = models.CharField(max\_length=20, default='')

email = models.EmailField(default='')

image = models.ImageField(upload\_to='profile\_image', blank=True)

def \_\_str\_\_(self):

return self.user.username

def create\_profile(sender, \*\*kwargs):

if kwargs['created']:

user\_profile=UserProfile.objects.create(user=kwargs['instance'])

post\_save.connect(create\_profile, sender=User)

class Question(models.Model):

title = models.CharField(max\_length=264)

slug = models.SlugField(max\_length=264)

content = models.TextField(default='')

author = models.ForeignKey(User, related\_name='questions', on\_delete=models.CASCADE)

published = models.DateTimeField(default=timezone.now)

def save(self, \*args, \*\*kwargs):

self.slug=slugify(self.title)

super(Question, self).save(\*args, \*\*kwargs)

def get\_absolute\_url(self):

return reverse('home:view\_question', args=[self.slug])

def \_\_str\_\_(self):

return self.title

class Answer(models.Model):

question=models.ForeignKey(Question, related\_name='answers', on\_delete=models.CASCADE)

user= models.CharField(max\_length=250)

email=models.EmailField()

body=models.TextField()

created = models.DateTimeField(auto\_now\_add=True)

approved = models.BooleanField(default=True)

# upvotes=models.BooleanField(default='0')

def approved(self):

self.approved =True

self.save()

def \_\_str\_\_(self):

return self.user

**views.py**

from django.shortcuts import render,redirect, get\_object\_or\_404

from django.http import HttpResponse

from django.views.generic import TemplateView

from django.contrib.auth import update\_session\_auth\_hash

from home.forms import (

RegistrationForm,

EditProfileForm,

AnswerForm,

AskQuestionForm,

)

from django.contrib.auth.models import User

from home.models import Question,Answer

from django.contrib.auth.forms import PasswordChangeForm

from django.contrib.auth.decorators import login\_required

from django.core.paginator import Paginator, EmptyPage, PageNotAnInteger

# Create your views here.

def welcome(request):

return render(request,'home/homepage.html')

def register(request):

if request.method=='POST':

form = RegistrationForm(request.POST)

if form.is\_valid():

form.save()

return redirect('/account/login')

else:

redirect('/account/register')

else:

form = RegistrationForm()

args={'form':form}

return render(request, 'home/register.html', args)

def about(request):

return render(request, 'home/about.html')

class testpage(TemplateView):

template\_name='home/dashboard.html'

class testpage2(TemplateView):

template\_name='home/homepage.html'

@login\_required

def view\_profile(request):

args= {'user': request.user}

return render(request, 'home/profile.html',args)

@login\_required

def edit\_profile(request):

if request.method== 'POST':

form= EditProfileForm(request.POST, instance=request.user)

if form.is\_valid():

form.save()

return redirect('/account/profile')

else:

return redirect('/account/profile/edit')

else:

form= EditProfileForm( instance=request.user)

args = {'form': form}

return render(request, 'home/edit\_profile.html', args)

@login\_required

def change\_password(request):

if request.method=="POST":

form= PasswordChangeForm(data=request.POST, user=request.user)

if form.is\_valid():

form.save()

update\_session\_auth\_hash(request, form.user)

return redirect('/account/profile')

else:

return redirect('/account/profile/edit/change-password')

else:

form= PasswordChangeForm(user=request.user)

args = {'form': form}

return render(request, 'home/change\_password.html', args)

def redirect\_password(request):

return redirect('/account/profile/edit/change-password')

def list\_of\_questions(request):

question=Question.objects.all()

paginator=Paginator(question, 3)

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

try:

questions=paginator.page(page)

except PageNotAnInteger:

questions=paginator.page(1)

except EmptyPage:

questions=paginator.page(paginator.num\_pages)

return render(request, 'home/dashboard.html', {'questions':questions, 'page':page} )

def view\_question(request, slug):

question = get\_object\_or\_404(Question, slug=slug)

args={'question':question}

return render(request, 'home/view\_question.html', args)

def add\_answer(request,slug):

question=get\_object\_or\_404(Question, slug=slug)

if request.method== 'POST':

form=AnswerForm(request.POST)

if form.is\_valid():

answer=form.save(commit=False)

answer.user=request.user

answer.email=request.user.email

answer.question=question

answer.save()

return redirect('home:view\_question',slug=question.slug)

else:

form=AnswerForm()

return render(request, 'home/add\_answer.html', {'form':form, 'question':question} )

def ask\_question(request):

if request.method == "POST":

form= AskQuestionForm(request.POST)

if form.is\_valid():

question=form.save(commit=False)

question.author=request.user

question.save()

return redirect('home:view\_question', slug=question.slug)

else:

form = AskQuestionForm()

args = {'form': form}

return render(request, 'home/ask\_question.html', args)

**forms.py**

from django import forms

from django.contrib.auth.models import User

from django.contrib.auth.forms import UserCreationForm, UserChangeForm

from home.models import Answer, Question

class RegistrationForm(UserCreationForm):

email = forms.EmailField(required = True)

class Meta:

model = User

fields = ('username',

'first\_name',

'last\_name',

'email',

'password1',

'password2'

)

def save(self,commit=True):

user = super(RegistrationForm, self).save(commit=False)

user.first\_name = self.cleaned\_data['first\_name']

user.last\_name = self.cleaned\_data['last\_name']

user.email = self.cleaned\_data['email']

if commit:

user.save()

return user

class EditProfileForm(UserChangeForm):

class Meta:

model = User

fields = (

'first\_name',

'last\_name',

'email',

'password',

)

class AskQuestionForm(forms.ModelForm):

class Meta:

model = Question

fields =(

'title',

'content',

)

class AnswerForm(forms.ModelForm):

class Meta:

model = Answer

fields = (

'body',

)

Some snapshots of our website :

























