Compsoft Technologies
Bangalore,Karnataka

AN INTERNSHIP REPORT
ON

" Dream Team "

BACHELOR OF ENGINEERING
In
COMPUTER SCIENCE AND ENGINEERING

Submitted by: Pushpa Bohara (1CD18CS113)

Under the Supervision of: Sumukh S Jadhav

Cambridge Institute of Technology.

(Affiliated to Visvesvaraya Technological University and Approved by AICTE, New Delhi)Accredited By NAAC with 'A' Grade

ISO 9001-2015 and 14001-2015 certified Institute

Bengaluru,Karnataka 560036

2021-2022

ABOUT THE COMPANY

CST is a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses. At CST , we believe that service and quality is the key to success
We provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that you may have. Experience the service like none other!

Some of our services include:

- Development - We develop responsive, functional and super-fast websites.
- We keep User Experience in mind while creating websites. A website should load quickly and shouldbe accessible even on a small view-port and slow internet connection.
- Mobile Application - We offer a wide range of professional Android, iOS& Hybrid app development services for our global clients, from a start up to a large enterprise.
- Design - We offer professional Graphic design, Brochure design & Logo design. We are expertsin crafting visual content to convey the right message to the customers.
- Consultancy - We are here to provide you with expert advice on your design anddevelopment requirement.
- Videos - We create a polished professional video that impresses your audience.

Table of Contents

OVERVIEW OF THE PROJECT

Project Name: Quora for Engineering Students.

Team Members: Pushpa Bohara
Shobha S
Raksha G P
Kavya R S

QES is a question and answer website for Engineering students. We tend to create a flagship site similar to the Stack Overflow Site of the Stack Exchange Network, created in 2008 by Jeff Atwood and Joel Spolsky. QES features questions and answers on a wide range of topics in Engineering. Our notion of a better-connected engineering community to tackle all kinds of academic and non Academical questions and problems would make the engineering community better. Establishing a community-based platform like this would help bypass the location barrier for Engineering students.

Short Term GOALS

1. Solve queries related to/limited to the engineering community
2. Make a better Community that can start discussions/mini topic debates
3. Upvote/Downvote answers

SPECIFICATIONS to be present in your initial build-Build 0.0.1

1. A working front end site with appropriate CSS
2. A working backend database where user data is to be stored
3. Registering upvote/downvote to the question/answer and the same being dynamically updated on the front end
4. Use of JS to add conditional features to the site
5. Login/Signup feature

Tools you will need:
For performing the examples discussed in this tutorial, you will need a Pentium 200-MHz computer with a minimum of 64 MB of RAM (128 MB of RAM recommended).
You will also need the following software –

● Linux 7.1 or Windows xp/7/8/10 operating system

● Java JDK 8

● Any IDE(IntelliJ recommended)

Quora for engineering students.

Quora is a community-based questions and answers website and app. Users post questions on any topic and other users respond.

Users can sign up using Facebook, Twitter, Google or email. As part of the registration process, they are asked to select five areas of interest to follow. Registered users can post or respond to questions, post reviews, or add a blog about a topic. Quora also has social media features: users can upvote (like) a question, follow people, comment and leave private messages for others. Spaces is a feature that lets people form communities around shared interests and tastes.

The purpose of Quora is **to ask and answer questions**. It is NOT simply a social media platform, and don't use it that way. You can benefit by answering and asking lots of questions. the difference between Google and Quora is that an actual person can explain things to you.

That means you can use Quora to find advice, ideas, perspectives, explanations, and answers to things you've always wondered about.

TOOLS USED

Software Requirements

- Visual Studio Code 2019.
- Google Chrome or Microsoft Edge of latest version.
- Front End: HTML, CSS, JS
- Backend : Django,python,sqlite
- Linux 7.1 or Windows XP/7/8/10 OS or Mac OS

Hardware Requirements

- Pentium 200-MHz computer with a minimum of 64 MB of RAM (128 MB of RAM recommended).
- Monitor with a refresh rate of at least 40Hz for a smooth GUI experience (optional).

IMPLEMENTATION

1. User.html

```
  <center><h1 style="font-family:bold; margin-top:50px;"> ALL USERS </h1></center>
  <hr>
  <div class="row" style="margin-left:300px;">
  {% for au in all_users %}
 <div class="col-md-3 up" style="background-color:#212529;border-radius:10px;">
          <br>
          <a href="{% url 'DetailProfile' au.id %}"><img src="{{au.prof_pic}}" class="avatar avatar1" /></a>
          <h4 class="up1"><a href="{% url 'DetailProfile' au.id %}" style="text-
           decoration:none;color:white;">{{au.user.username|upper}}</a></h4>
          <h4 class="up1"><button class="btn btn-success" style="text-
           decoration:none;">{{au.level}}</button></h4>
         </div>{% endfor %}
   </div>
```

2. Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8" />
<link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<meta name="theme-color" content="#000000" />
<meta
name="description"
content="Web site created using create-react-app"/>

<link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.4.0/css/font-awesome.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
<link rel="stylesheet"  type="text/css"  href="{% static 'styling.css' %}">

  <title>Just-Ask</title>
 </head>
 <body style="zoom: 80%;">

  <div id="root"></div>

  <div class="wrapper fadeInDown">
    <div id="formContent">

      <div class="fadeIn first">
        <h1 style="font-family: bold; margin-top:20px;">LOGIN</h1>
      </div>

        <form action="login" method="POST">

          {% csrf_token %}
        <input style="margin-top:50px;" type="text" id="login" class="fadeIn second" name="username"
placeholder="username">
        <input style="margin-top:30px;" type="password" id="password" class="fadeIn third" name="password"
placeholder="password">
        <input style="margin-top: 20px;" type="submit" class="fadeIn fourth" value="Log In">

      </form>

      <div id="formFooter">
        <a class="underlineHover" href="register">Don't have an account ? </a>
      </div>

    </div>
  </div>


 </body>
</html>
```

3.Question.html

```
<div class="row">

<div class="col-md-12" style="background-color: #121212;padding-bottom: 50px;">

<img style="margin-top:300px;margin-left:20px ;" src="{{qs.prof_pic}}" class="avatar avatar2" />

</div>

</div>

{% if qs.user == request.user %}

<div style="margin-left:300px;margin-top:100px;font-family: bold;">

<h1 style="margin-left: 400px;">YOUR QUESTIONS</h1>
</div>

{% else %}
<div style="margin-left:300px;margin-top:100px;font-family: bold;">

<h1 style="margin-left: 400px;">{{qs.user.username|upper}}'s QUESTIONS</h1>

</div>

{% endif %}

{% for ans1 in qs.Questions.all %}

{% ifchanged ans1.id %}

<div class="row" style="margin-top: 100px;">


<div class="col-sm-8 setts1" style="padding:10px;margin-left:325px;background-color:#212529;border-
radius:10px;">
<div>
<div>
```

```
<a href="{% url 'DetailProfile' qs.id %}"><img src="{{qs.prof_pic}}" class="avatar avatar1"/></a>
<b style="margin-left: 20px;color:white;"> {{qs.user.username|upper}} </b><p style="color:white;margin-
left:820px;margin-top:-35px">{{ans1.dateTime}}
{% for fq in qs.FavQuestions.all %}

{% if fq.id == ans1.id %}
<img src="{% static 'star.png' %}"  style="height:30px;width:30px;margin-left:60px;" />
{% endif %}

{% endfor %}

</div>
<h3><a href="{% url 'DetailQuestion' ans1.id %}" style="color:#F77E7E;text-decoration:none"><b class="question-
title" style="margin-left:60px;">{{ans1.q_title}}</b></a></h3>
</div>
<div class="q-lbtm-content">
<div class="textStyle" style="margin-top:10px;margin-left:60px">
<h5 style="color:whitesmoke;"><b>{{ans1.content}}  ?</b></h5>
</div>
<div class="question-cat">
<ul class="ul1">
<li style="margin-left:15px; margin-top:20px">
<button class="btn btn-primary btn3">{{ans1.tag}}</button>
</li>
</ul>

</div>

<div class="question-cat" style="margin-left:16px;margin-top:40px;">
<ul class="ul1">
<li style="margin-top:-10px">
<button class="btn ansViewTag" style="border-radius:5px;color:black;">
<img  src="{% static 'eye.png' %}" class="ans-img" />
<b> Views {{ans1.views}} </b></button>

<button class="btn ansViewTag" style="border-radius:5px;color:black;">
<img  src="{% static 'ans1.png' %}" class="ans-img" />
<b> Answers {{ans1.answers}} </b></button>
<button class="btn btn-danger reportTag"> Report </button>
</li>
</ul>
</div>
</div>
</div>



</div>
<hr/>



{% if qs %}


<!-- <h3 style="color:black;font-family:bold;margin-left:350px;"><b>ANSWERS</b></h3>

{% for quest1 in an.all %}
```

```
<div class="row" style="margin-top:50px;">


<div class="col-sm-8" style="margin-left:325px;font-family:bold;">
<div class="q-lbtm-content">

<a href="UserProfile"><img src="{{an1.user.prof_pic}}" class="avatar avatar2" /></a>
<b style="margin-left:20px;">{{an1.user.user.username|upper}}</b><p style="color:grey; margin-
left:820px; margin-top:-35px">July 14, 2016</p>
<div>
<div style="margin-top:40px;margin-bottom:40px;">
</div>
<h4>
<b class="question-title" style="margin-left:60px;">{{quest1.question.q_title}}</b>
</h4>
</div>
<div class="q-lbtm-content" style="margin-top:30px;">
<div class="textStyle" style="margin-top:10px;margin-left:60px">
<p><b>{{quest1.content}}</b></p>
</div>
<div class="question-cat" style="margin-top:30px;">
<ul class="ul1">
<li style="margin-left:15px; margin-top:-10px">
<button class="btn btn-primary btn3" >Tag</button>
</li>
</ul>

</div>
<hr/>


</div>

</div>

</div>

</div>

<hr/>
{% endfor %}
-->

{% else %}

<center><h3 style="font-family:bold; margin:40px;">NO ANSWERS YET</h3></cen
```

4. Answer.html

```
<center><h1 style="font-family:bold; margin:100px;">ANSWER THIS QUESTION</h1></center>

{{ form.media }}

<form class="ui form" action="" method="POST" >

{% csrf_token %}

<center style="padding-left:150px;padding-right:150px">{{ form|crispy }}</center>

<center><button type="submit" class="btn btn-danger" style="margin:50px;font-family:bold;" >Post
Answer</button></center>

</form>
```

5.Ask Question.html

```
<h1>ASK QUESTION</h1>

{% load crispy_forms_tags %}


{{ form.media }}

<form class="ui form" action="" method="POST" enctype="multipart/form-data">

{% csrf_token %}

<center style="padding-left:150px;padding-right:150px">{{ form|crispy }}</center>

<center><button type="submit" class="btn btn-danger" style="margin:50px;font-family:bold;" >Post
Question</button></center>

</form>

</div>
</div>
</div>
```

6.User Answer.html

```
<div class="row">

<div class="col-md-12" style="background-color: #121212;padding-bottom: 50px;">

<a href="UserProfile"><img style="margin-top:300px;margin-left:20px ;" src="{{UP.prof_pic}}" class="avatar avatar2"
/></a>

</div>

</div>

{% if ans %}

<div style="margin-left:300px;margin-top:100px;font-family: bold;">

{% if UP.user == request.user %}
<h1 style="margin-left:400px;">Your Answers</h1>

{% else %}
<h1 style="margin-left:400px;">  {{UP.user.username|capfirst}}'s  Answers</h1>
{% endif %}
</div>


{% for ans1 in ans.all %}

<!-- {% ifchanged ans1.question.id %}
{% endifchanged %} -->
<div class="col-sm-8 setts1" style="margin-top:60px;margin-left:325px;background-color:#212529;border-
radius:10px;">
<div>
<div>
<a href="{% url 'DetailProfile' ans1.user.id %}"><img src="{{ans1.user.prof_pic}}" class="avatar avatar1"/></a>
{% if ans1.user.user == request.user %}

<b style="margin-left:20px;color:white;"> You </b><p style="color:white;margin-left:820px;margin-top:-
35px">{{ans1.dateTime}}
{% else %}
<b style="margin-left:20px;color:white;"> {{ans1.user.user.username|upper}} </b><p style="color:white;margin-
left:820px;margin-top:-35px">{{ans1.dateTime}}
{% endif %}
</div>

<div style="margin-top:40px;margin-bottom:40px;">
</div>
<h4>
<a href="{% url 'ShowAnswer' ans1.question.id %}" style="color:#F77E7E;text-decoration:none;margin-
left:60px;"><b class="question-title">{{ans1.question.q_title}}</b></a>
</h4>
</div>
<div class="q-lbtm-content" style="margin-top:30px;">
<div class="textStyle" style="margin-top:10px;margin-left:60px;color:white;">
<p><b>{{ans1.content}}</b></p>
</div>
<div class="question-cat" style="margin-top:30px;">
<ul class="ul1">
```

```
<li style="margin-left:15px; margin-top:-10px">
<button class="btn btn-primary btn3" >Tag</button>
</li>
</ul>

</div>
<hr/>


</div>

</div>
<hr/>



{% endfor %}


{% else %}

<center><h1 style="font-family:bold; margin-top:140px;padding-bottom:300px;">No Answers Yet</h1></center>
```

7.User Profile

```
div class="row">

<div class="col-md-12" style="background-color: #121212;padding-bottom: 50px;">

<img style="margin-top:300px;margin-left:20px ;" src="{{UP.prof_pic}}" class="avatar avatar2" />

</div>

</div>


<div style="margin-left:300px;margin-top:100px;font-family: bold;">

{% if UP.user == request.user %}

<h1 style="margin-left: 400px;">YOUR PROFILE</h1>

{% else %}

<h1 style="margin-left: 400px;">{{UP.user.username|upper}}'s PROFILE</h1>


{% if cond == "True" %}

<p class="btn btn-success">You are Following him</p>

{% else %}

<a href="{% url 'FollowUser' UP.id %}"><button class="btn btn-primary"
style="height:50px;width:120px;">Follow</button></a>
```

```
{% endif %}




{% endif %}

<div class="row" style="margin-top:130px;">

<div class="col-md-3 fol" style="border-radius:10px;">

<img src="{% static 'followers.png' %}" class="fol-avatar" />
{% if UP.user == request.user %}
<a href="UserFollowers" style="margin-left:50px;color:black;text-decoration:none;"><b>Followers</b></a>
{% else %}
<a href="{% url 'OtherUserFollower' UP.id %}" style="margin-left:50px;color:black;text-
decoration:none;"><b>Followers</b></a>
{% endif %}
<br>
<br>

{% if UP.followers == NULL %}

{% if UP.user == request.user %}
<center>You have no Followers.</center>
{% else %}
<center>{{UP.user.username|upper}} have no Followers.</center>
{% endif %}

{% else %}

{% if UP.user == request.user %}
<center>You have has {{UP.followers.count}} Followers.</center>
{% else %}
<center>{{UP.user.username|upper}} has {{UP.followers.count}} Followers</center>
{% endif %}

{% endif %}


</div>

<div class="col-md-3 fol" style="margin-left: 30px;border-radius:10px;">

<img src="{% static 'followers.png' %}" class="fol-avatar" />

{% if UP.user == request.user %}
<a href="UserFollowing" style="margin-left:50px;color:black;text-decoration:none;"><b>Following</b></a>
{% else %}
<a href="{% url 'OtherUserFollowing' UP.id %}" style="margin-left:50px;color:black;text-
decoration:none;"><b>Following</b></a>
{% endif %}

<br>
<br>

{% if UP.following == NULL %}

{% if UP.user == request.user %}
```

```
<center>You have no Following.</center>
{% else %}
<center>{{UP.user.username|upper}} is Following no one.</center>
{% endif %}

{% else %}

{% if UP.user == request.user %}
<center>You are following {{UP.following.count}} users.</center>
{% else %}
<center>{{UP.user.username|upper}} is following {{UP.following.count}} users</center>
{% endif %}

{% endif %}


</div>

<div class="col-md-3 fol" style="margin-left: 30px;border-radius:10px;">

<img src="{% static 'level.png' %}" class="lev-avatar" />
<b style="margin-left:50px">LEVEL</b>
<br>
<br>
<center>{{UP.level}} Level</center>

</div>

</div>

<hr style="margin-top: 40px;margin-bottom:40px;" />

<div class="row" style="margin-top:50px;border-radius:10px;">

<div class="col-md-3 fol">

<img src="{% static 'answers.png' %}" class="lev-avatar" />

<a  href="{% url 'UserAnswers' UP.id %}" style="margin-left:50px;color:black;text-
decoration:none;"><b>ANSWERS</b></a>
<br>
<br>
<center>{{answers}}</center>

</div>

<div class="col-md-3 fol" style="margin-left: 30px;border-radius:10px;">

<img src="{% static 'questions.png' %}" class="lev-avatar" />

{% if UP.user == request.user %}

<a  href="UserQuestions" style="margin-left:50px;color:black;text-decoration:none;"><b>QUESTIONS</b></a>

{% else %}

<a  href="{% url 'OUserQuestions' UP.id %}" style="margin-left:50px;color:black;text-
decoration:none;"><b>QUESTIONS</b></a>

{% endif %}
```

```
<br>
<br>
<center>{{qs}}</center>

</div>


<div class="col-md-3 fol" style="margin-left:30px;border-radius:10px;">

<img src="{% static 'heart.png' %}" style="margin-left:-15px" class="lev-avatar" />
<a href="{% url 'UserFavQuestions' UP.id %}" style="color:black;text-decoration:none;margin-
left:40px"><b>FAVORATE QUESTIONS</b></a>
<br>
<br>
<center>{{fav_qs}}</center>

</div>


</div>

<hr style="margin-top: 40px;margin-bottom:40px;" />

<div class="row" style="margin-top:50px;border-radius:10px;">

<div class="col-md-3 fol">

<img src="{% static 'heart.png' %}" style="margin-left:-15px" class="lev-avatar" />
<a href="{% url 'UserFavAnswer' UP.id %}" style="color:black;text-decoration:none;margin-
left:40px"><b>FAVORATE ANSWER</b></a>
<br>
<br>
<center>{{fav_ans}}</center>

</div>

<div class="col-md-3 fol" style="margin-left:30px;border-radius:10px;">

<img src="{% static 'questions.png' %}" class="lev-avatar" />

<a  href="{% url 'OtherUserBestQuest' UP.id %}" style="margin-left:50px;color:black;text-decoration:none;"><b>
BEST QUESTIONS</b></a>
<br>
<br>
<center>{{b_qs}}</center>

</div>


<div class="col-md-3 fol" style="margin-left:30px;border-radius:10px;">

<img src="{% static 'bestAnswer1.png' %}" style="margin-left:40px" class="lev-avatar" />
<a href="{% url 'UserBestAnswer' UP.id %}" style="color:black;text-decoration:none;"><b>BEST
ANSWERS</b></a>
<br>
<br>
<center>{{b_ans}}</center>

</div>
```

```
</div>



</div>

<hr style="margin-top: 100px;margin-bottom:100px;"/>
```

8.Admin.py

```
from django.contrib import admin
from home.models import UserProfile,Question,Answer,Notification

admin.site.register(UserProfile)
admin.site.register(Answer)
admin.site.register(Question)
admin.site.register(Notification)
```

9.APPs.py

```
from django.apps import AppConfig


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

10. Forms.py

```
from django import forms
from home.models import UserProfile,Question,Answer

class UserProfileForm(forms.ModelForm):

class Meta:
model = UserProfile

fields = [
'prof_pic'
]


class QuestionForm(forms.ModelForm):

class Meta:
model = Question

fields = [
'q_title',
'q_image',
'tag',
'content'
]

class AnswerForm(forms.ModelForm):
```

```
class Meta:
model =  Answer

fields = [
'content',
'tag',

]
```

10.Models.py

```
from django.db import models
from django.contrib.auth.models import User
from django.db.models.signals import post_save
from django.dispatch import receiver

showChoices = (

    ('Beginner' , 'Beginner'),
    ('Intermediate' , 'Intermediate'),
    ('advance' , 'advance'),
)

ReportChoices = (

    ('Invalid-content','Invalid-content'),

)


FavChoices = (

    ('Yes','Yes'),
    ('No','No')
)



class Question(models.Model):

    user = models.ForeignKey(User,on_delete=models.CASCADE)
    q_image = models.ImageField(upload_to='static',null=True)
    q_title = models.CharField(max_length=100)
    tag = models.CharField(max_length=20)
    content = models.TextField()
    answers = models.IntegerField(default=0)
    views = models.IntegerField(default=0)
    fav = models.CharField(max_length=50)
    report = models.CharField(choices=ReportChoices,max_length=50)
    dateTime = models.DateTimeField(auto_now=True)


class UserProfile(models.Model):

    user = models.ForeignKey(User,on_delete=models.CASCADE)
    prof_pic = models.ImageField(upload_to='static')
    level = models.CharField(choices=showChoices,max_length=50)
    followers = models.ManyToManyField('self',related_name='follow',null=True)
    following = models.ManyToManyField('self',related_name='followings',null=True)
    Questions = models.ManyToManyField(Question,null=True)
    FavQuestions = models.ManyToManyField(Question,related_name='FavQuestion',null=True)
    FavAnswers = models.ManyToManyField("Answer",related_name='FavAnswer',null=True)


class Answer(models.Model):

    user = models.ForeignKey(UserProfile,on_delete=models.CASCADE)
    questioner = models.ForeignKey(UserProfile,on_delete=models.CASCADE,related_name='questioner')
    question = models.ForeignKey(Question,on_delete=models.CASCADE,null=True)
    content = models.TextField()
    tag = models.CharField(max_length=20)
    dateTime = models.DateTimeField(auto_now=True)


class Notification(models.Model):

    message = models.TextField()
    user = models.ForeignKey(UserProfile,on_delete=models.CASCADE)
    from_user = models.ForeignKey(UserProfile,on_delete=models.CASCADE,related_name='from_User')
    viewed = models.BooleanField(default=False)
    Question_id = models.CharField(max_length=50)
    dateTime = models.DateTimeField(auto_now=True)


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

    def __str1__(self):
        return self.from_user.user.username
```

11.Views.py

```
  from django.shortcuts import render
from django.shortcuts import render, HttpResponseRedirect, Http404,redirect
from django.shortcuts import get_list_or_404, get_object_or_404
from django.contrib import messages
from django.forms.models import modelformset_factory
from django.conf.urls import url
from django.urls import reverse
from django.contrib.auth.models import User, auth
from django.contrib.auth import authenticate
from home.models import UserProfile,Question,Answer,Notification
from home.forms import QuestionForm,UserProfileForm,AnswerForm

def LoginPage(request):

    if request.method == "POST":

        username = request.POST['username']
        password = request.POST['password']

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

        if user is not None:
            auth.login(request,user)
            return redirect('UserProfile')
        else:
            return redirect('login')

    else:

        context = locals()

        return render(request,'login.html',context)


def RegisterPage(request):


    if request.method == "POST":

        username = request.POST['username']
        email = request.POST['email']
        password = request.POST['password']
        password1 = request.POST['password1']

        if password == password1:

            if User.objects.filter(username=username).exists():
                return redirect('register')

            elif User.objects.filter(email=email).exists():
                return redirect('register')

            user1 = User.objects.create_user(username=username,email=email,password=password)
            user1.save()
            friend , created = UserProfile.objects.get_or_create(user=user1,prof_pic='static/social.png',level='Beginner')

            return redirect('login')

        else:
            return redirect('register')

    else:

        context = locals()

        return render(request,'register.html',context)


def UserSettings(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count


    context = { "count_notis":total_Notis,'u1':users1 }
    return render(request,'UserSettings.html',context)

def Logout(request):

    auth.logout(request)
    return redirect('login')


def ProfileSettings(request,pk):
    users1 = UserProfile.objects.filter(id=pk).first()
    myprofileForm = UserProfileForm(request.POST or None, request.FILES or None , instance=users1 )
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    if myprofileForm.is_valid():
        myprofileForm.save()
        return redirect('UserProfile')

    context = { "count_notis":total_Notis,'form':myprofileForm,'u1':users1 }
    return render(request,'ProfileSettings.html',context)


def allUsers(request):

    users1 = UserProfile.objects.exclude(user=request.user).all()
    users2 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    return render(request,'allUsers.html',{'all_users':users1,"count_notis":total_Notis})


def AskQuestion(request):

    myquestionForm = QuestionForm(request.POST or None,  request.FILES or None )
    users1 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    if myquestionForm.is_valid():

        myquestionForm.instance.user = request.user
        myquestionForm.save()

        q_id = myquestionForm.instance.id

        qs = Question.objects.get(id=q_id)

        for q in UserProfile.objects.filter(user=request.user):
            q.Questions.add(qs)


            for usr in UserProfile.objects.exclude(user=request.user).all():

                for foll in usr.following.all():

                    if foll == users1:
                        notis , created = Notification.objects.get_or_create(user=usr,from_user=users1,message="had 
asked a Question",Question_id=q_id)

        return redirect('myQuestion')

    context = { 'form':myquestionForm,"count_notis":total_Notis }
    return render(request,'AskQuestion.html',context)


def UpdateQuestion(request,pk):

    users2 = UserProfile.objects.filter(user=request.user).first()
    qss1 = Question.objects.get(id=pk)
    myquestionForm = QuestionForm(request.POST or None, request.FILES or None, instance=qss1 )
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    if myquestionForm.is_valid():
        myquestionForm.save()

        q_id = myquestionForm.instance.id

        qs = Question.objects.get(id=q_id)

        for usr in UserProfile.objects.exclude(user=request.user).all():

            for foll in usr.following.all():

                if foll == qss1.user:
                    notis , created = Notification.objects.get_or_create(user=usr,from_user=qss1.user,message="had
updated his Question",Question_id=q_id)

        return redirect('myQuestion')

    context = { 'form':myquestionForm,"count_notis":total_Notis }
    return render(request,'AskQuestion.html',context)


def DeleteQuestion(request,pk):

    qss1 = Question.objects.get(id=pk)
    qss1.delete()

    Notis = Notification.objects.filter(Question_id=pk)
    Notis.delete()

    return redirect('myQuestion')



def UpdateProfile(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    return render(request,'UserProfile1.html',{'UP':users1,"count_notis":total_Notis })


def UserProfile1(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    all_ans = Answer.objects.filter(user=users1).all().count
    all_users = UserProfile.objects.filter().all()
    qs = Question.objects.filter(user=request.user).all().count
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count
    ans = Answer.objects.filter(user=users1).all()
    fav_ans = users1.FavAnswers.all().count()
    fav_qs = users1.FavQuestions.all().count()

    users = []

    for qns in users1.Questions.all():

        for n in all_users:
            for qns1 in n.FavQuestions.all():

                if qns1 == qns:

                    users.append(qns1)


    mylist = list(dict.fromkeys(users))
    m1 = len(users)

    counts = 0
    for items in range(len(mylist)):
        counts += 1


    usersFA = []

    for ans1 in ans:

        for n in all_users:

            for qns1 in n.FavAnswers.all():

                if qns1 == ans1:

                    usersFA.append(qns1)

    mylist2 = list(dict.fromkeys(usersFA))

    counts1 = 0
    for items in range(len(mylist2)):
        counts1 += 1


    context = {'UP':users1,'answers':all_ans,'qs':qs,
                "count_notis":total_Notis,'fav_ans':fav_ans,
                'fav_qs':fav_qs,'b_qs':counts,'b_ans':counts1}

    return render(request,'UserProfile1.html',context)


def DetailAnswer(request,pk):

    qs = Answer.objects.filter(id=pk).first()
    total_ans = Answer.objects.filter(question=qs).all().count
    users1 = UserProfile.objects.filter(user=request.user).first()

    if UserProfile.objects.filter(user=request.user,FavAnswers=qs).exists():
        condition = "True"
    else:
        condition = "False"

    condition1 = "False"
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context1 = {'aq':qs,'curr_user':users1,'cond':condition,'cond1':condition1,
                "count_notis":total_Notis,'total_ans':total_ans }

    return render(request,'DetailAnswer.html',context1)
```

```
def DetailQuestion(request,pk):

    qs = Question.objects.get(id=pk)
    user_q = UserProfile.objects.filter(user=qs.user).first()
    total_ans = Answer.objects.filter(question=qs).all().count
    users1 = UserProfile.objects.filter(user=request.user).first()

    if UserProfile.objects.filter(user=request.user,FavQuestions=qs).exists():
        condition = "True"
    else:
        condition = "False"

    if UserProfile.objects.filter(user=request.user,Questions=qs).exists():
        value = 'T'
    else:
        value = 'F'

    qs.views += 1
    qs.save()

    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count
    context1 = {'aq':qs,'user_q':user_q,'curr_user':users1,'val':value,
                'cond':condition,"count_notis":total_Notis,'total_ans':total_ans}

    return render(request,'DetailQuestion.html',context1)


def FavAnswer(request,pk):

    qs = Answer.objects.filter(id=pk).first()
    users1 = UserProfile.objects.filter(user=request.user).first()
    users2 = UserProfile.objects.filter(user=qs.user.user).first()
    q_id = qs.id

    for q in UserProfile.objects.filter(user=request.user):
        q.FavAnswers.add(qs)

        if users2.user != request.user:
            notis , created = Notification.objects.get_or_create(user=users2,from_user=users1,message="Liked Your
Answer",Question_id=q_id)

    return redirect('DetailAnswer',pk=pk)


def FavQuestion(request,pk):

    qs = Question.objects.filter(id=pk).first()
    users1 = UserProfile.objects.filter(user=request.user).first()
    users2 = UserProfile.objects.filter(user=qs.user).first()
    q_id = qs.id

    for q in UserProfile.objects.filter(user=request.user):
        q.FavQuestions.add(qs)

        if users2.user != request.user:
            notis , created = Notification.objects.get_or_create(user=users2,from_user=users1,message="Liked Your
Question",Question_id=q_id)

    return redirect('DetailQuestion',pk=pk)
```

```
def UserFavAnswer(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context = { 'qs':users1,"count_notis":total_Notis }
    return render(request,'FavAns.html',context)


def DetailAnswer(request,pk):

    qs = Answer.objects.filter(id=pk).first()
    users1 = UserProfile.objects.filter(user=request.user).first()
    users2 = UserProfile.objects.filter(user=qs.user.user).first()
    q_id = qs.id

    if UserProfile.objects.filter(user=request.user,FavAnswers=qs).exists():
        condition = "True"
    else:
        condition = "False"

    context = {'ans':qs,'cond':condition}

    return render(request,'DetailAns.html',context)


def FollowUser(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    curr_User = UserProfile.objects.filter(user=request.user).first()

    ans = Answer.objects.filter(user=users1).all().count
    qs = Question.objects.filter(user=users1.user).all().count


    for q in UserProfile.objects.filter(user=request.user):

        q.following.add(users1)

    for q1 in UserProfile.objects.filter(id=pk):

        q1.followers.add(curr_User)

        notis , created = Notification.objects.get_or_create(user=users1,from_user=curr_User,message="had started
Following You")

    return redirect('DetailProfile',pk=pk)


def DetailProfile(request,pk):

    users1 = UserProfile.objects.get(id=pk)
    curr_User = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=curr_User,viewed=False).all().count
    all_ans = Answer.objects.filter(user=users1).all().count
    ans = Answer.objects.filter(user=users1).all()
    qs = Question.objects.filter(user=users1.user).all().count
    all_users = UserProfile.objects.filter().all()
    fav_ans = users1.FavAnswers.all().count()
    fav_qs = users1.FavQuestions.all().count()


    if UserProfile.objects.filter(user=request.user,following=users1).exists():
        condition = "True"
    else:
        condition = "False"

    users = []

    for qns in users1.Questions.all():

        for n in all_users:

            for qns1 in n.FavQuestions.all():

                if qns1 == qns:

                    users.append(qns1)


    mylist = list(dict.fromkeys(users))
    m1 = len(users)

    counts = 0
    for items in range(len(mylist)):
        counts += 1


    usersFA = []

    for ans1 in ans:

        for n in all_users:

            for qns1 in n.FavAnswers.all():

                if qns1 == ans1:

                    usersFA.append(qns1)

    mylist2 = list(dict.fromkeys(usersFA))

    counts1 = 0
    for items in range(len(mylist2)):
        counts1 += 1


    context = {'UP':users1,'answers':all_ans,'qs':qs,
            'Curr_User':curr_User,'cond':condition,
            "count_notis":total_Notis,'fav_ans':fav_ans,
            'fav_qs':fav_qs,'b_qs':counts,'b_ans':counts1}

    return render(request,'UserProfile1.html',context)


def ShowAnswer(request,pk):

    users1 = Question.objects.get(id=pk)
    ans = Answer.objects.filter(question=users1).all()
    profile = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=profile,viewed=False).all().count

    if UserProfile.objects.filter(user=request.user,FavQuestions=users1).exists():
        condition = "True"
    else:
        condition = "False"


    context = { 'ans':ans,'qs':users1,"count_notis":total_Notis,
                'profile':profile,'cond':condition}

    return render(request,'ShowAnswer.html',context)


def AnswerQuestions(request,pk):

    qs1 = Question.objects.get(id=pk)
    profile = UserProfile.objects.filter(user=request.user).first()
    profile1 = UserProfile.objects.filter(user=qs1.user).first()
    total_Notis = Notification.objects.filter(user=profile,viewed=False).all().count
    q_id = qs1.id

    myAnsForm = AnswerForm(request.POST or None)

    if myAnsForm.is_valid():

        myAnsForm.instance.user = profile
        myAnsForm.instance.question = qs1
        myAnsForm.instance.questioner = profile1
        myAnsForm.save()

        qs1.answers += 1
        qs1.save()

        if profile1.user != request.user:
            notis , created = Notification.objects.get_or_create(user=profile1,from_user=profile,message="had
answered your question",Question_id=q_id)

        return redirect('myQuestion')

    context = { 'form':myAnsForm,"count_notis":total_Notis}
    return render(request,'AnswerQ.html',context)


def UpdateAnswer(request,pk):

    qs1 = Answer.objects.get(id=pk)
    profile = UserProfile.objects.filter(user=request.user).first()
    profile1 = UserProfile.objects.filter(user=qs1.user.user).first()
    total_Notis = Notification.objects.filter(user=profile,viewed=False).all().count
    q_id = qs1.id

    myAnsForm = AnswerForm(request.POST or None,instance=qs1)

    if myAnsForm.is_valid():
        myAnsForm.save()

        if profile1.user != request.user:
            notis , created = Notification.objects.get_or_create(user=profile1,from_user=profile,message="had updated
his Aswer",Question_id=q_id)

        return redirect('UserAnswers')

    context = { 'form':myAnsForm,"count_notis":total_Notis}
    return render(request,'AnswerQ.html',context)


def DeleteAns(request,pk):

    ans = Answer.objects.get(id=pk)
    ans.delete()

    Notis = Notification.objects.filter(Question_id=pk)
    Notis.delete()

    return redirect('UserAnswers')


def UserAnswers(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    users2 = UserProfile.objects.filter(user=request.user).first()
    ans = Answer.objects.filter(user=users1).all()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    context = { 'ans':ans,'UP':users1,"count_notis":total_Notis}
    return render(request,'UserAns.html',context)

def myQuestion(request):

    users1 = UserProfile.objects.filter().all()
    c_user = UserProfile.objects.filter(user=request.user).first()
    count_fav = c_user.FavQuestions.all().count
    total_Notis = Notification.objects.filter(user=c_user,viewed=False).all().count


    for us in users1:
        for qns in us.Questions.all():

            if UserProfile.objects.filter(user=request.user,FavQuestions=qns).exists():
                qns.fav = "True"
            else:
                qns.fav = "False"

    context = {'all_qs':users1,'curr_user':c_user,'count_fav':count_fav,"count_notis":total_Notis}
    return render(request,'Question.html',context)


def UserFavQuestions(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    qs = Answer.objects.filter(questioner=users1).all()
    users2 = UserProfile.objects.filter().all()
    users3 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users3,viewed=False).all().count

    context = { 'qs':users1, 'an':qs,"count_notis":total_Notis,'u2':users2 }
    return render(request,'FavQs.html',context)
```

```
def UserQuestions(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    qs = Answer.objects.filter(questioner=users1).all()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context = { 'qs':users1, 'an':qs,"count_notis":total_Notis }
    return render(request,'UserQs.html',context)

def UserBestQuestions(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    users2 = UserProfile.objects.filter(user=request.user).first()
    all_users = UserProfile.objects.filter().all()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    users = []

    for qns in users1.Questions.all():

        for n in all_users:

            for qns1 in n.FavQuestions.all():

                if qns1 == qns:

                    users.append(qns1)

    mylist = list(dict.fromkeys(users))

    context = { 'qs':mylist,'u1':users1 }

    return render(request,'UserBestQs1.html',context)


def UserBestAnswer(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    users2 = UserProfile.objects.filter(user=request.user).first()
    all_users = UserProfile.objects.filter().all()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count
    ans = Answer.objects.filter(user=users1).all()

    users = []

    for ans1 in ans:

        for n in all_users:

            for qns1 in n.FavAnswers.all():

                if qns1 == ans1:

                    users.append(qns1)

    mylist = list(dict.fromkeys(users))

    print(mylist.count)
    context = { 'qs':mylist,'u1':users1 }

    return render(request,'UserBestAns1.html',context)


def OtherUserQuestions(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    qs = Answer.objects.filter(questioner=users1).all()
    users2 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    context = { 'qs':users1, 'an':qs,"count_notis":total_Notis }
    return render(request,'UserQs.html',context)


def UserFollowers(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context = { 'curr_user':users1,"count_notis":total_Notis }
    return render(request,'UserFollowers.html',context)

def OtherUserFollowers(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    users2 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    context = { 'curr_user':users1,"count_notis":total_Notis }
    return render(request,'UserFollowers.html',context)


def OtherUserFollowing(request,pk):

    users1 = UserProfile.objects.filter(id=pk).first()
    users2 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users2,viewed=False).all().count

    context = { 'curr_user':users1 ,"count_notis":total_Notis }
    return render(request,'UserFollowing.html',context)


def UserNotifications(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    Notis = Notification.objects.filter(user=users1).all().order_by('-dateTime').order_by('viewed')
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context = { 'my_notis':Notis,"count_notis":total_Notis }
    return render(request,'UserNotifications.html',context)

def ReadNotifs(request,pk):

    Notis = Notification.objects.filter(id=pk).first()
    Notis.viewed = True
    Notis.save()
    return redirect('UserNotifications')
```

```
def UserFollowing(request):

    users1 = UserProfile.objects.filter(user=request.user).first()
    total_Notis = Notification.objects.filter(user=users1,viewed=False).all().count

    context = { 'curr_user':users1,"count_notis":total_Notis }

    return render(request,'UserFollowing.html',context)
```

SNAPSHOTS

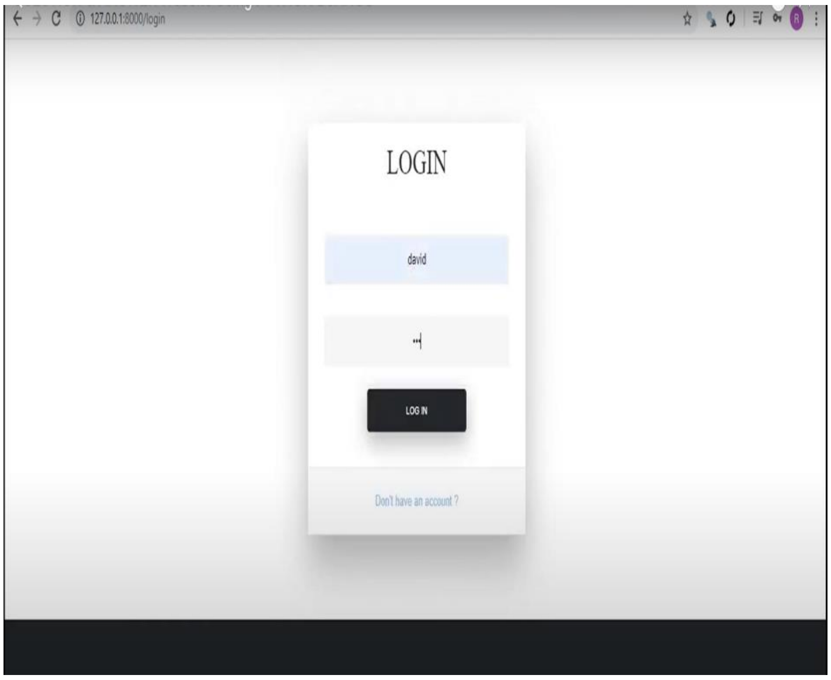


Fig 1.Login

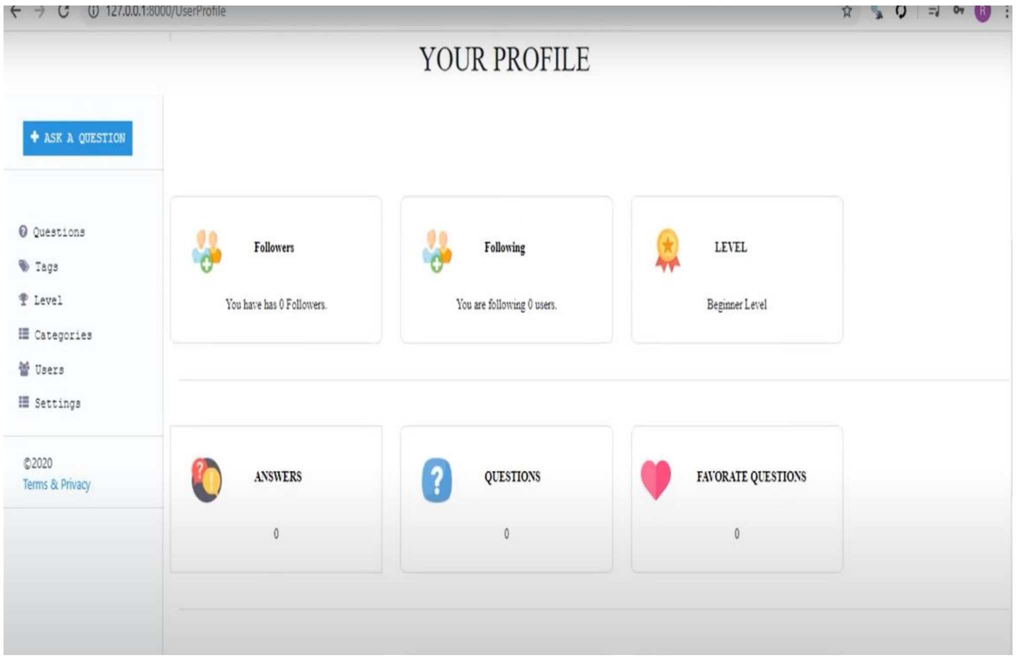


Fig2. Profile

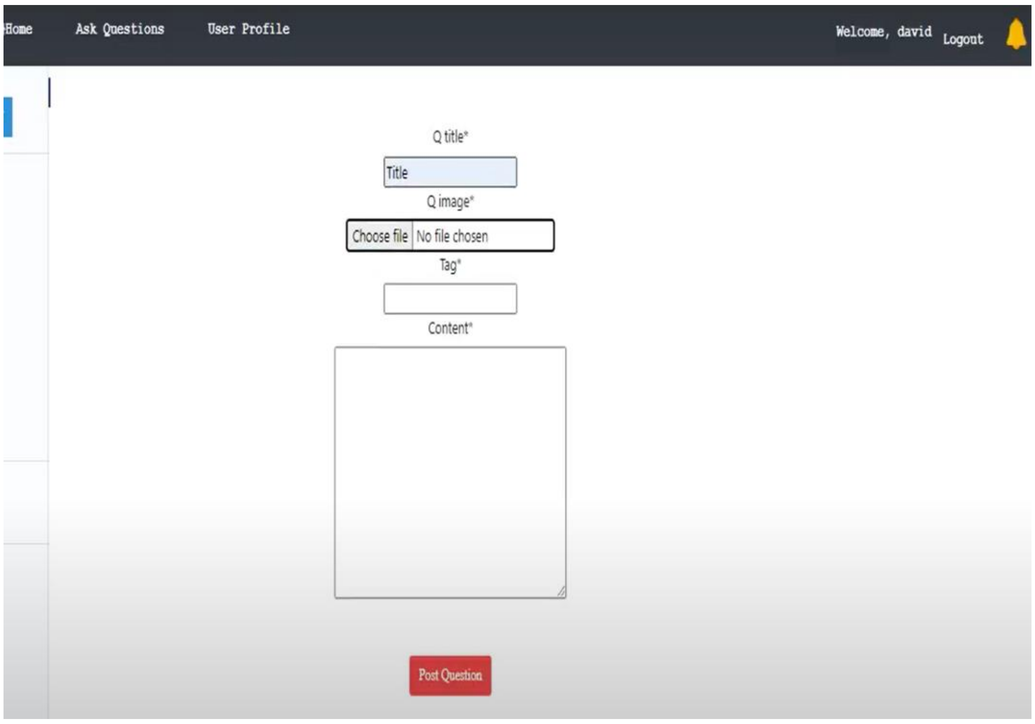


Fig 3: Question box

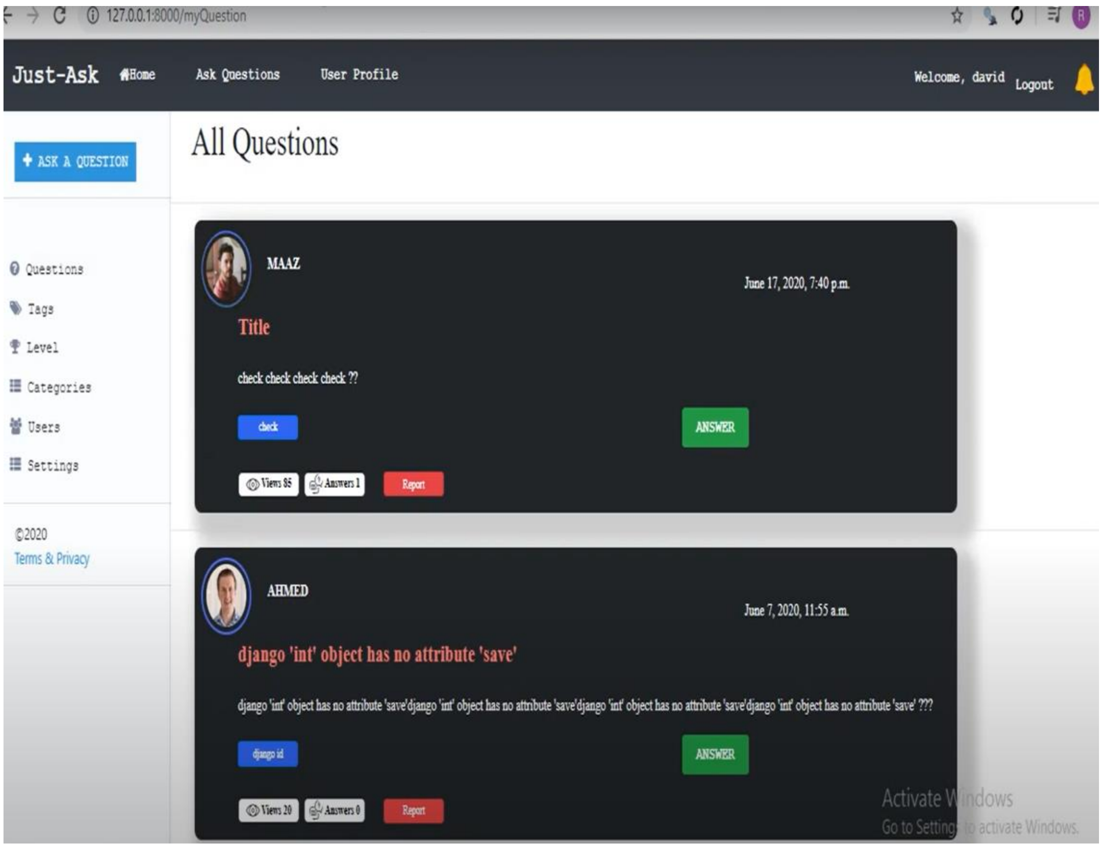
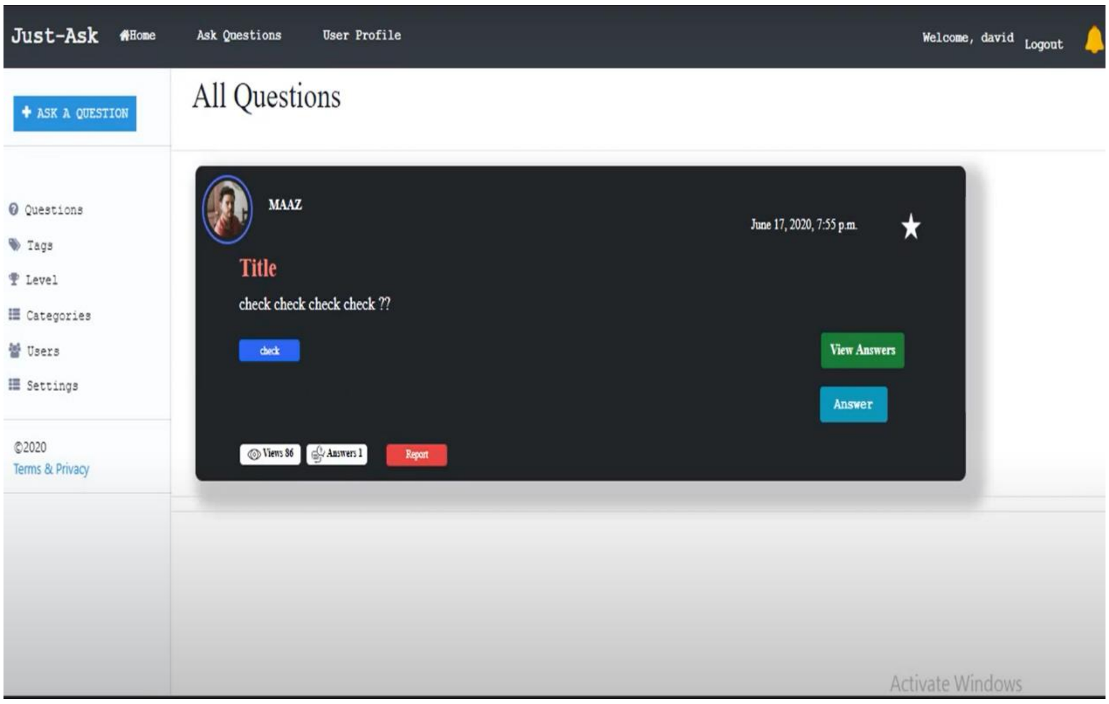


Fig 5: Home page



All Questions

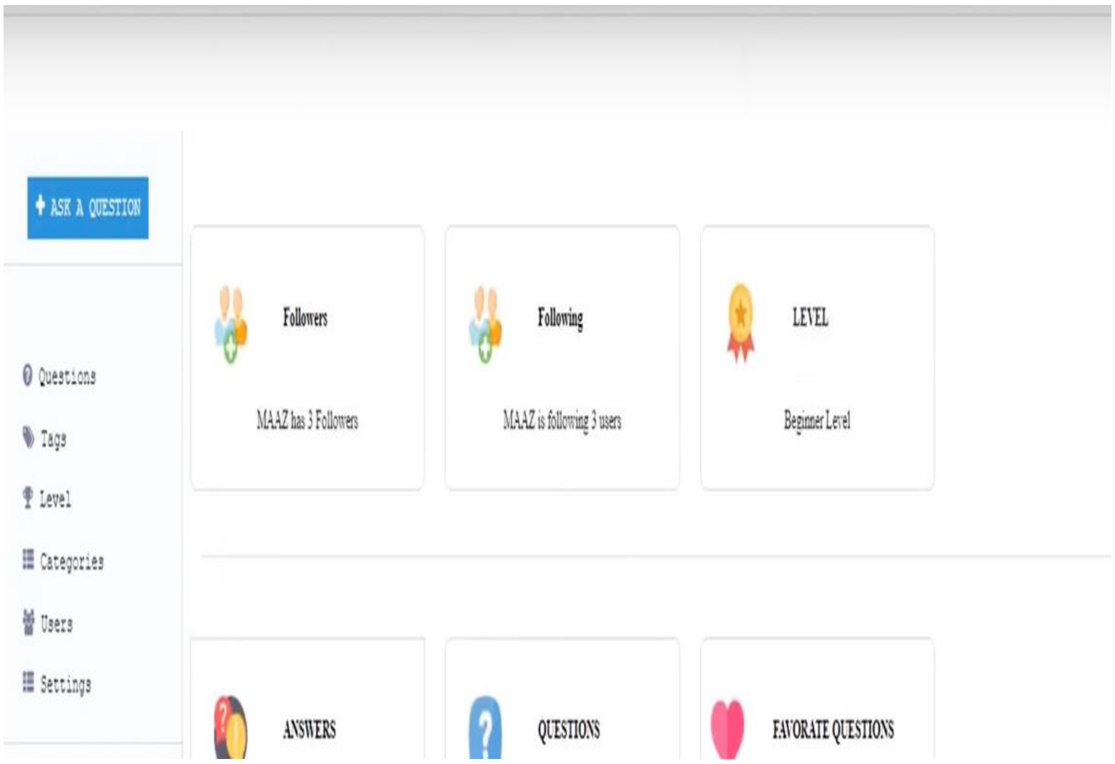
MAAZ

June 17, 2020, 7:55 p.m.

Title

check check check check ??

Fig 5: question page



Followers

MAAZ has 3 Followers

Following

MAAZ is following 3 users

LEVEL

Beginner Level

Fig 6: profile

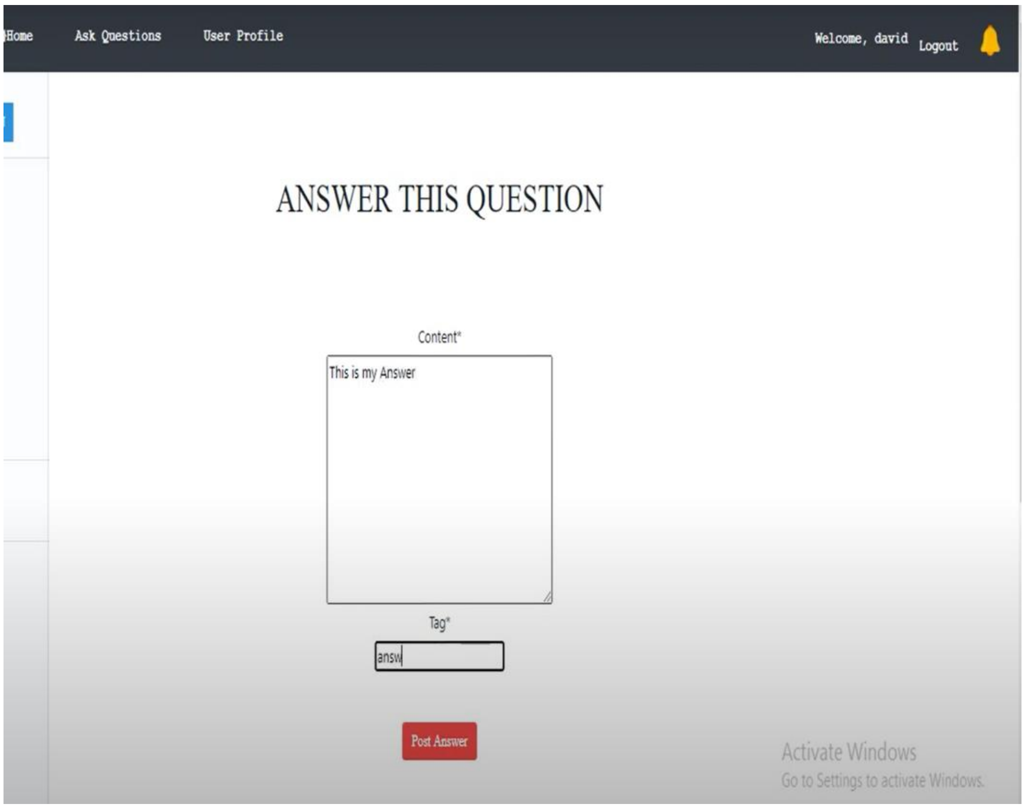


Fig 7: Answer page

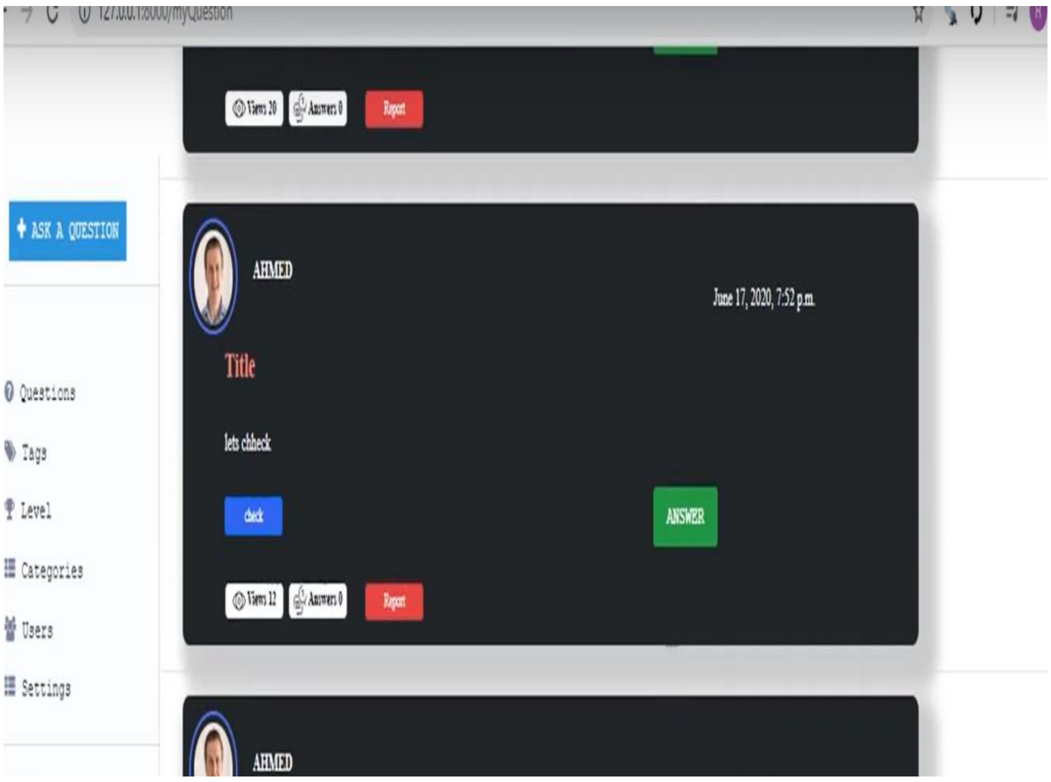


Fig 8: Answers for question

BIBLIOGRAPHY

- https://www.w3schools.com
- https://www.geeksforgeeks org
- https://freefrontend.com
- https://css-tricks.com/
- https://www.takeiteasyengineers.com
- https://dev.to/mychi_darko/php-tips-and-tricks-4kpn