

PROJECT REPORT ON SUMMER INDUSTRIAL TRAINING

COURSE TITLE :

**WEB DEVELOPMENT WITH PYTHON &
DJANGO**

**(AUGUST 14th, 2019 to SEPTEMBER
14th ,2019)**

PROJECT TITLE :

QBRAIN

Submitted By:

Debanjan Bhattacharyya (Roll No. : 16800116076)

*(CSE, 4th Year, Saroj Mohan Institute Of
Technology)*

TABLE OF CONTENTS

Acknowledgement

Declaration

Introduction

Technologies Used

Description of The Project

Snapshots of Website

Snapshots of Codes

Conclusion

ACKNOWLEDGEMENT

This Project Titled Question Submitting Website (QBrain) was supported by WEBTEK LABS PVT. LTD. .

We are thankful to our mentor Ms. Deepa Ashok Kumar ,who provided expertise that greatly assisted the project completion. We are thankful to WEBTEK LABS for giving us an opportunity to work there and help us to gain real time job experience.

DECLARATION

We, Debanjan Bhattacharyya, Avik Ghosh, Rajat Saha, students of *CSE, 4th Year of <Saroj Mohan Institute Of Technology> <Kolkata> <West Bengal>* have completed summer training project from Webtek Labs Pvt. Ltd. (Duration: 14th August 2019 – 14th September 2019).

During the mentioned period we worked on Web Development with Django and completed the summer training under the guidance of Ms. Deepa Ashok Kumar.

Debanjan Bhattacharyya:

Avik Ghosh:

Rajat Saha:

DATE : _____

INTRODUCTION

Web development is a broad term for the work involved in developing a web site for the Internet (World Wide Web) or an intranet (a private network). Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications, electronic businesses, and social network services. A more comprehensive list of tasks to which web development commonly refers, may include web engineering, web design, web content development, client liaison, client-side/server-side scripting, web server and network security configuration, and e-commerce development. For larger organizations and businesses, web development teams can consist of hundreds of people (web developers) and follow standard methods like Agile methodologies while developing websites. Smaller organizations may only require a single permanent or contracting developer, or secondary assignment to related job positions such as a graphic designer and/or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department. There is open source software for web development like Django (web framework), GlassFish, LAMP (Linux, Apache, MySQL, PHP) stack and Perl/Plack. This has kept the cost of learning web development to a minimum. Another contributing factor to the growth of the industry has been the rise of easy-to-use WYSIWYG web-development software, such as Adobe Dreamweaver, Blue Griffon and Microsoft Visual Studio. Knowledge of Hyper Text Markup Language (HTML) or of programming languages is still required to use such software, but the basics can be learned and implemented quickly with the help of help files, technical books, internet tutorials, or face-to-face training.

TECHNOLOGIES USED

❖ Text editor software and local server installation :

- ✓ Details about the software requirement
- ✓ Download and installation process
- ✓ Download and installation of Python, Django, Notepad++
- ✓ Setting up database and creating database connections

❖ HTML Layouts :

- ✓ Div containers
- ✓ Form fields
- ✓ Tables
- ✓ Buttons

❖ CSS Designing :

- ✓ Navigation bar
- ✓ Footer
- ✓ Background images
- ✓ Div containers
- ✓ Responsiveness of overall website
- ✓ Buttons, Fields

❖ JAVASCRIPT Used :

- ✓ Responsiveness of overall website

❖ DJANGO Used :

- ✓ Connecting database
- ✓ Passing and retrieving values from database
- ✓ Server and Client side validations
- ✓ Displaying profile details
- ✓ Search
- ✓ Sessions
- ✓ Validations on signup and login pages

❖ Other Components Used :

- ✓ Bootstrap

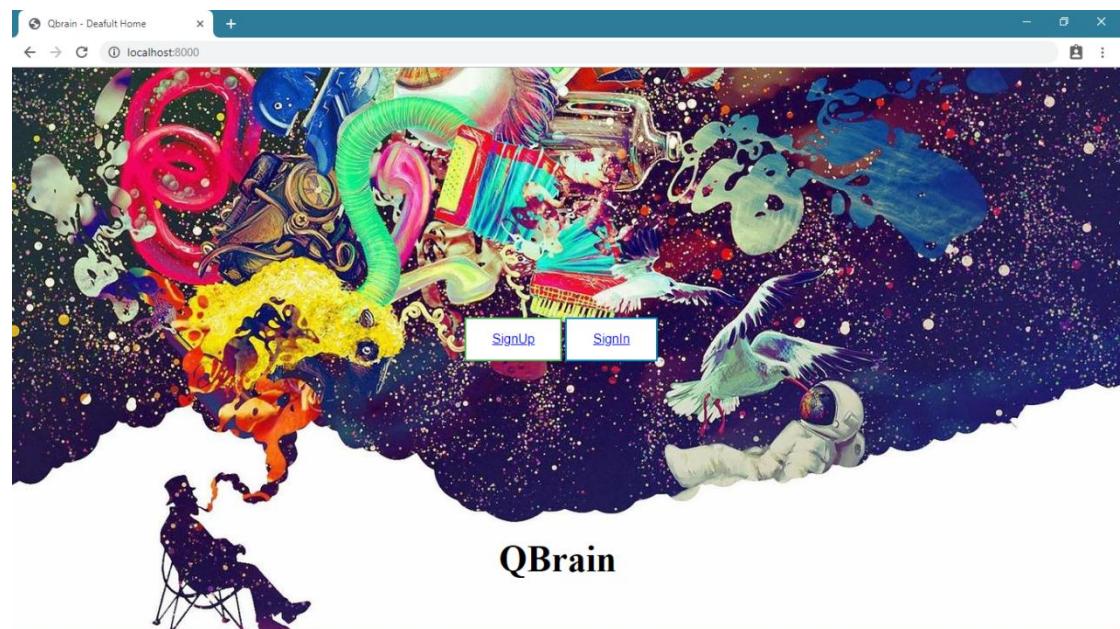
DESCRIPTION OF THE PROJECT

Humans are curious by nature and we have thousands of queries in our brain. QBrain is an online app where you can ask your queries and get answered by other users. Everything is saved in our database so don't worry about your question getting deleted, and you may also answer other's questions. All the questions are categorised by there categories and while answering others you can go to your category and answer.

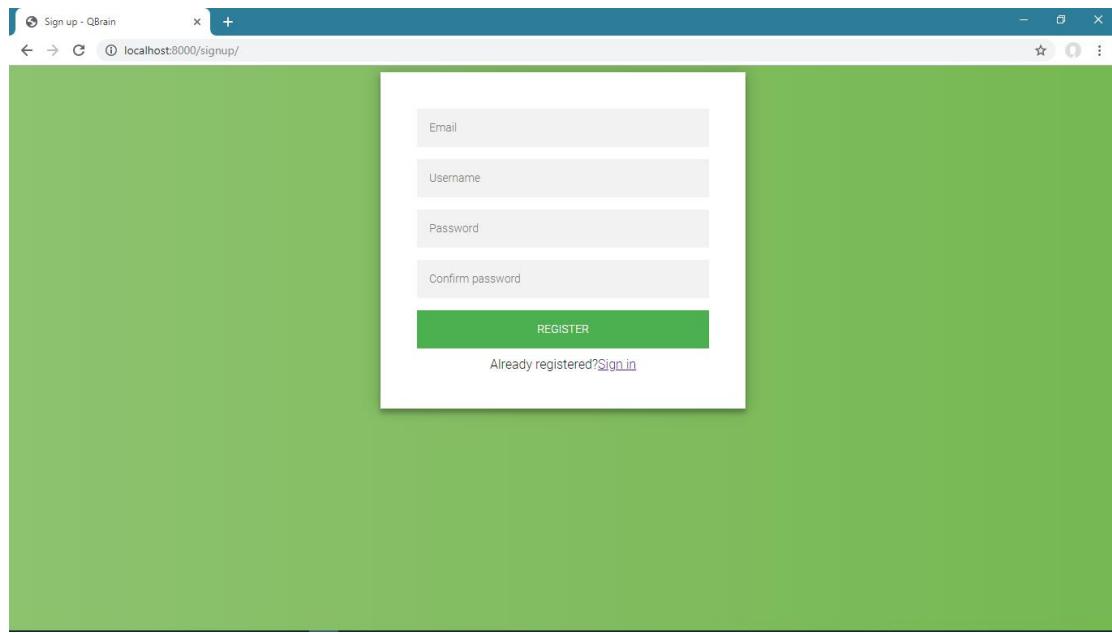
I made this project during the training I did few months back on web development using Django at WebTek.

SnapShots

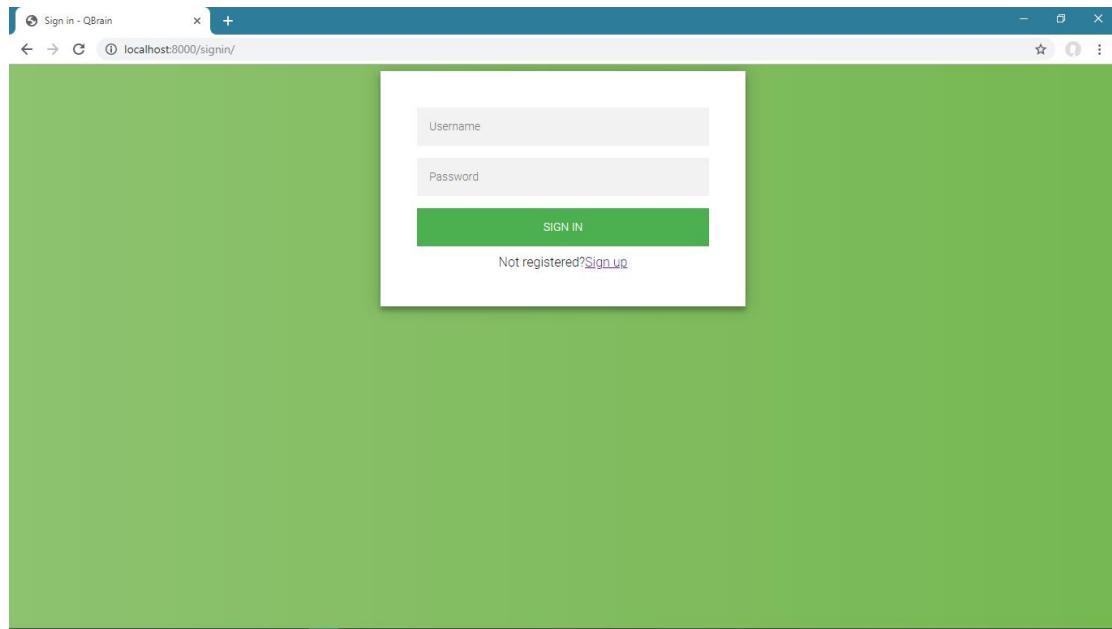
LANDING PAGE: CLICK ON SIGN UP IF YOU'RE NEW USER ELSE ON SIGN IN.



SIGN UP PAGE: REGISTER YOURSELF BY GIVING THE NEEDED INFORMATION



SIGN IN PAGE: CLICK ON SIGN IN IF YOU'RE ALREADY AN USER.



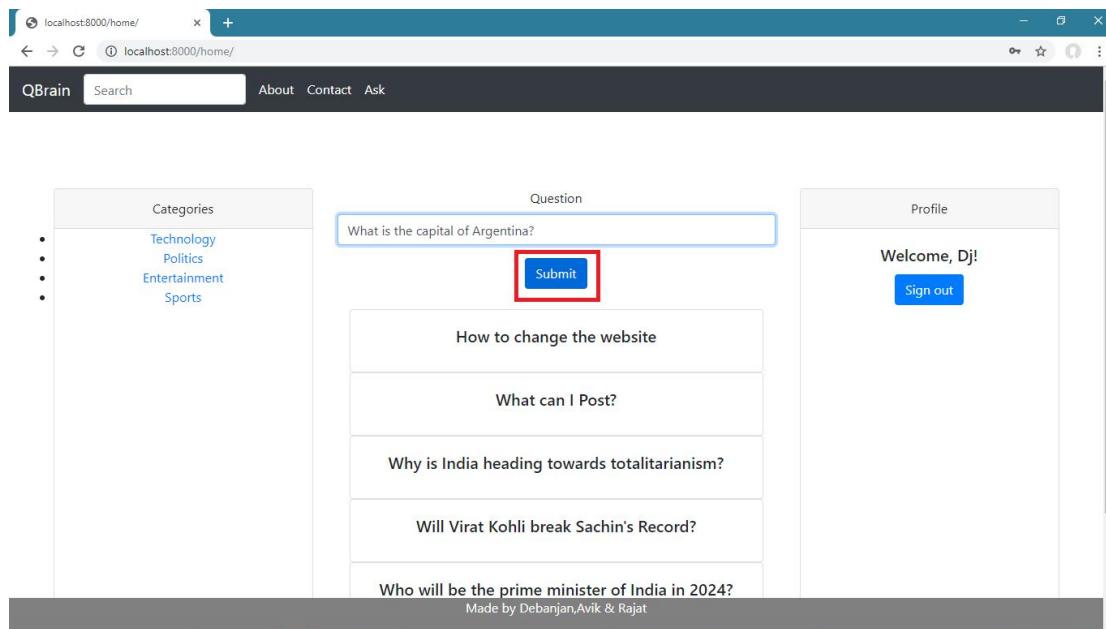
HOME PAGE: AFTER SUCCESSFULLY LOGGING IN YOU WILL COME TO THIS PAGE

The screenshot shows a web browser window for 'localhost:8000/home/'. The title bar says 'localhost:8000/home/'. The page has a dark header with 'QBrain' and a search bar. On the left, there's a sidebar with 'Categories' (Technology, Politics, Entertainment, Sports) and a list of bullet points. The main area has a 'Question' input field with a 'Submit' button. Below it is a list of questions: 'How to change the website', 'What can I Post?', 'Why is India heading towards totalitarianism?', 'Will Virat Kohli break Sachin's Record?', and 'Who will be the prime minister of India in 2024?'. The footer says 'Made by Debanjan, Avik & Rajat'. The right side shows a 'Profile' section with 'Welcome, Dj!' and a 'Sign out' button.

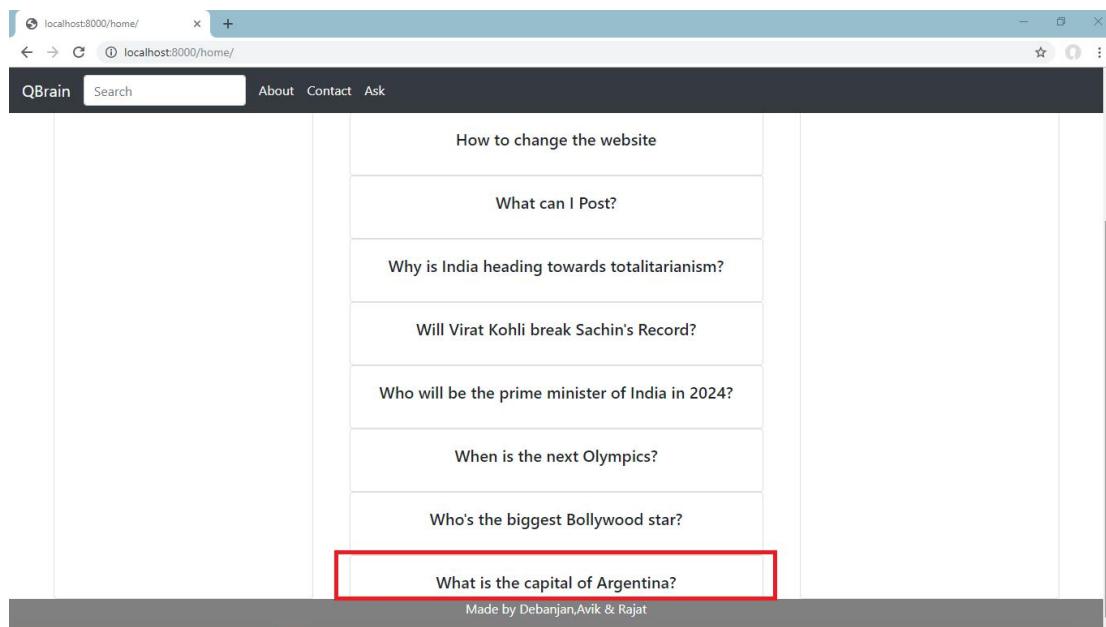
IN THE QUESTION BOX YOU CAN ASK YOUR QUESTION

The screenshot shows the same web browser window as before, but now the 'Question' input field is highlighted with a red border and contains the text 'What is the capital of Argentina?'. The rest of the page structure is identical to the first screenshot.

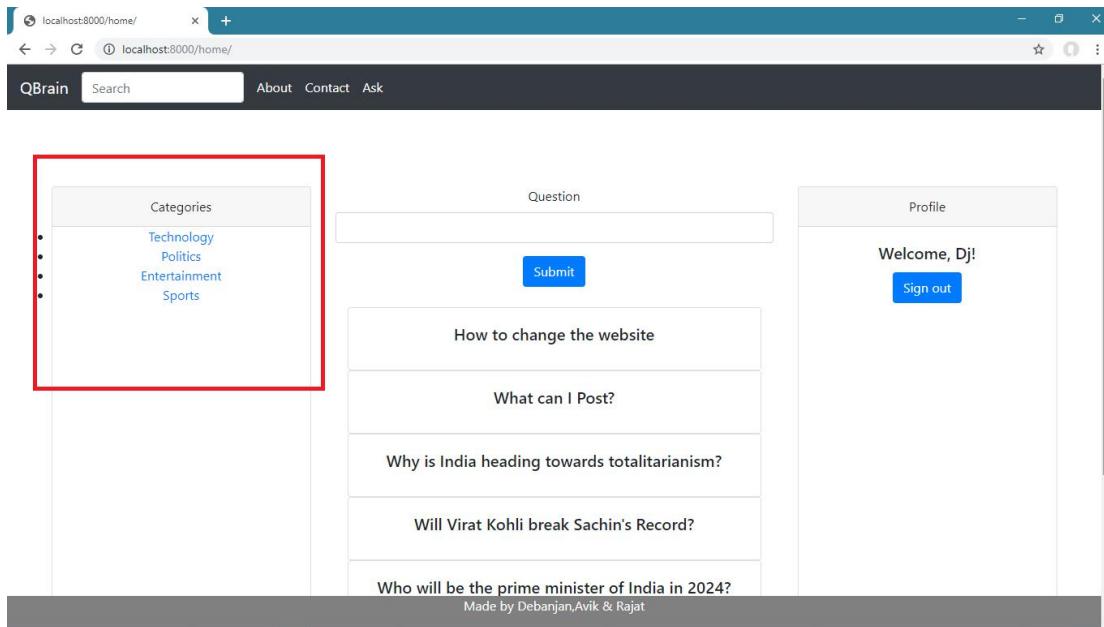
CLICK ON THE SUBMIT BUTTON TO POST YOUR QUESTION



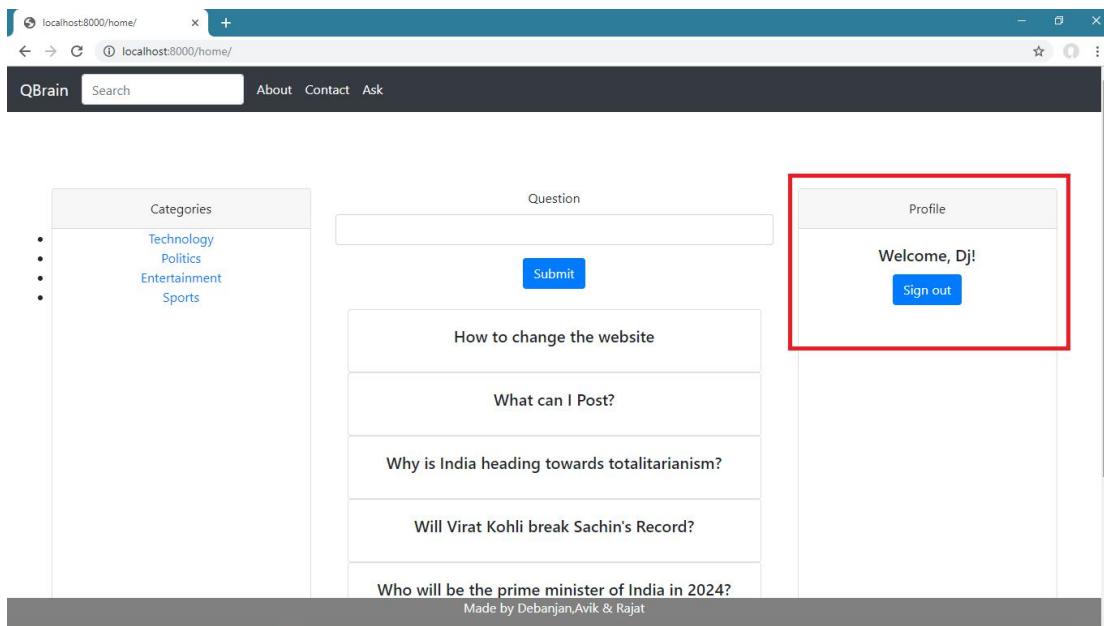
YOU CAN SEE YOUR QUESTION AT THE BOTTOM



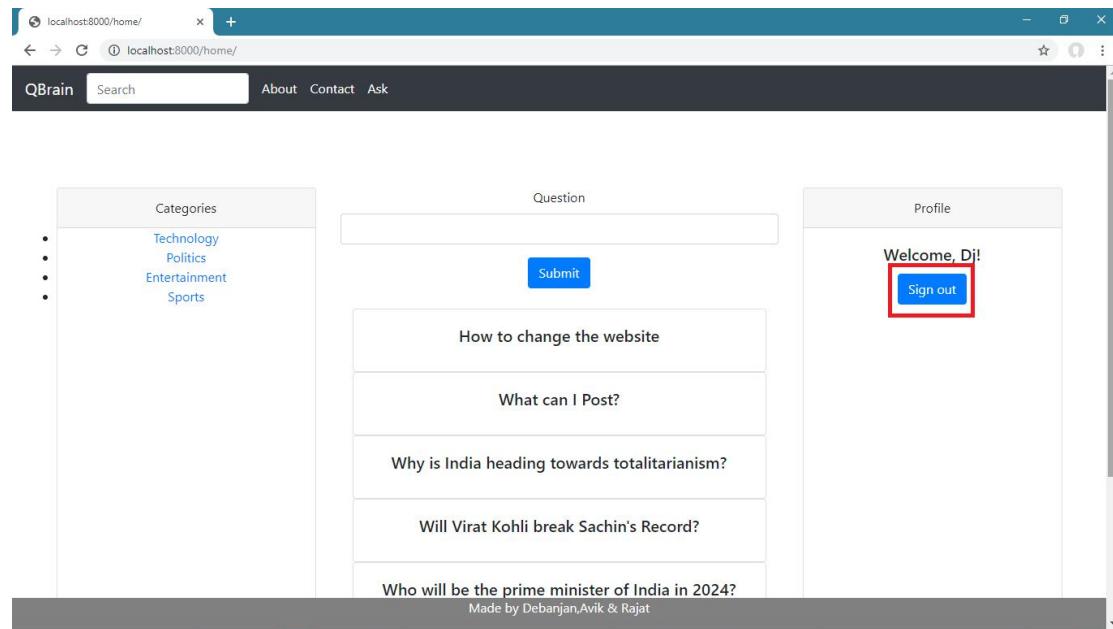
HERE ARE THE CATEGORIES



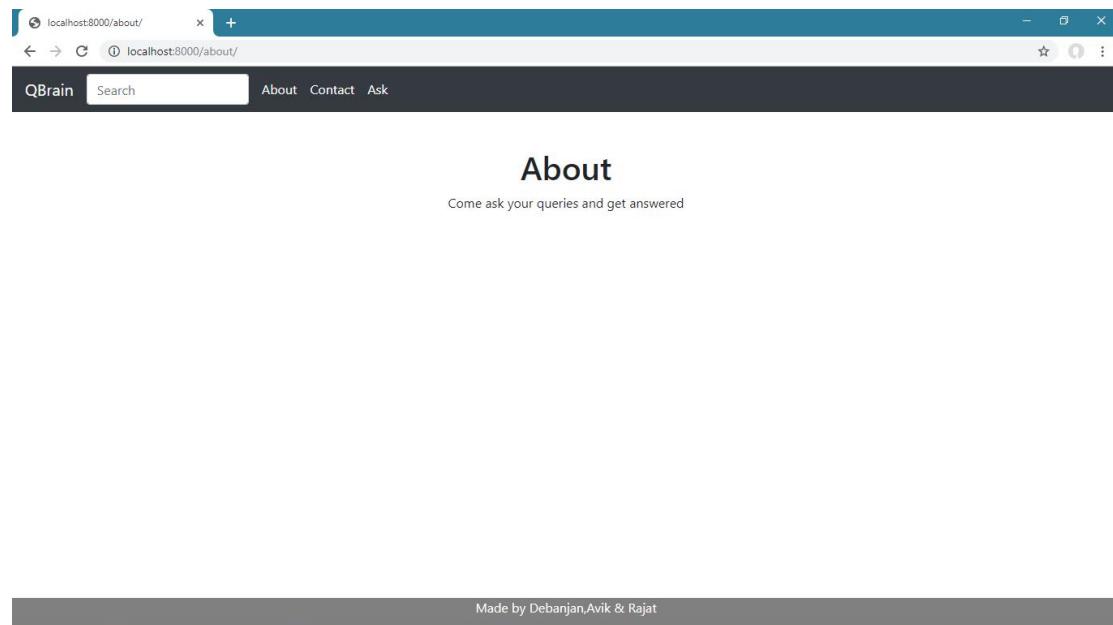
YOUR USERNAME WITH AN WELCOME MESSAGE WILL BE HERE



BY CLICKING ON SIGN OUT YOU WILL BE LOGGED OUT OF YOUR ACCOUNT



ABOUT PAGE: Come ask your queries and get answered



CONTACT PAGE: YOU CAN SEND US YOUR FEEDBACK OR ANY QUESTION ABOUT OUR WEBSITE

The screenshot shows a contact form on a website. The browser address bar displays "localhost:8000/contact/". The page title is "QBrain". The navigation menu includes "Search", "About", "Contact", and "Ask". The contact form fields are as follows:

- First Name: "Your name.."
- Last Name: "Your last name.."
- Country: "India" (selected in a dropdown menu)
- Subject: "Write something.."

A green "Submit" button is located at the bottom left of the form area. At the bottom right, there is a footer note: "Made by Debanjan, Avik & Rajat".

VIEWS.PY

```
views.py - D:\QBrain\qbrain\qbrainapp\views.py (3.7.3)
File Edit Format Run Options Window Help
from django.shortcuts import render, redirect
from qbrainapp.models import category,qbraindb
from qbrainapp.forms import qbraindbForm,UserForm
from django.contrib.auth.models import User
from django.contrib.auth import authenticate,login,logout
#from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.decorators import login_required

def landing(request):
    return render(request, "qbrain/landing-page.html")

""" def home(request):
    categories=category.objects.all()
    return render(request,"qbrain/home.html",{'categories':categories}) """

@login_required
def home(request):
    if request.method == "POST":
        form = qbraindbForm(request.POST)
        print("hhhhhhhhhhhh")
        if form.is_valid():
            print("yyyyyyyyyyyy")
            new = form.save(commit=False)
            new.author = request.user
            new.save()
            #form.save()
            return redirect("home")
    else:
        form = qbraindbForm()
        categories = category.objects.all()

Ln: 8 Col: 0
```

```
*views.py - D:\QBrain\qbrain\qbrainapp\views.py (3.7.3)*
File Edit Format Run Options Window Help
    return redirect('home')
else:
    form = qbraindbForm()
    categories = category.objects.all()
    questions = qbraindb.objects.all()
    return render(request, "qbrain/home.html", {'form': form, 'categories': categories, 'questions': questions}

def about(request):
    return render(request, "qbrain/about.html")

def signup(request):
    if request.method=="POST":
        form = UserForm(request.POST)
        if form.is_valid():
            form.save()
            username=request.POST['username']
            password=request.POST['password1']
            user=authenticate(request,username=username,password=password)
            if user is not None:
                login(request,user) #logs in the user
                return redirect("/home/")
    else:
        form = UserForm()
    return render(request, "registration/signup.html",{'form' : form})

@login_required
def SignOut(request):
    logout(request)
    return redirect('landing')

Ln: 57 Col: 0
```

```
views.py - D:\QBrain\qbrain\qbrainapp\views.py (3.7.3)*
File Edit Format Run Options Window Help
    if form.is_valid():
        form.save()
        username=request.POST['username']
        password=request.POST['password1']
        user=authenticate(request,username=username,password=password)
        if user is not None:
            login(request,user) #logs in the user
            return redirect("/home/")
    else:
        form = UserForm()
    return render(request, "registration/signup.html",{'form' : form})

@login_required
def SignOut(request):
    logout(request)
    return redirect('landing')

""" @login_required
def showquestion(request):
    questions=qbraindb.objects.get(author=request.user.username) """

@login_required
def showbycat(request,id):
    question=qbraindb.objects.select_related().filter(cat=id)
    return render(request, "qbrain/showbycat.html", { 'question':question})

@login_required
def contact(request):
    return render(request,"qbrain/contact.html")
```

Ln: 72 Col: 0

MODELS.PY

```
models.py - D:\QBrain\qbrain\qbrainapp\models.py (3.7.3)*
File Edit Format Run Options Window Help
from django.db import models
from django.contrib.auth.models import User

# Create your models here.
class category(models.Model):
    name=models.CharField(max_length=20)

class qbraindb(models.Model):
    cat = models.ForeignKey(category, on_delete=models.CASCADE, null=True)

    questions=models.CharField(max_length=100)

    author=models.ForeignKey(User,on_delete='CASCADE', null=True)

    answer=models.CharField(max_length=100000,default='')

    answer_date_posted=models.DateTimeField(auto_now_add=True)

    question_date_posted=models.DateTimeField(auto_now_add=True)

    like=models.IntegerField(default=None,blank=True,null=True)

class UserData(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    #bio = models.TextField(max_length=500, blank=True)
    #gender = models.CharField(max_length=30)
    #dob = models.DateField()
    #location=models.CharField(max_length=30)
```

Ln: 1 Col: 0

FORMS.PY

```
forms.py - D:\QBrain\qbrain\qbrainapp\forms.py (3.7.3)
File Edit Format Run Options Window Help
from django.forms import ModelForm
from django import forms
from qbrainapp.models import qbraindb,category
from django.contrib.auth.models import User
from qbrainapp.models import UserData
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm

class qbraindbForm(ModelForm):
    class Meta:
        model = qbraindb
        fields = ('questions', )

class UserForm(UserCreationForm):
    class Meta:
        model = User
        fields = ('email','username','password1','password2')

Ln: 1 Col: 0
```

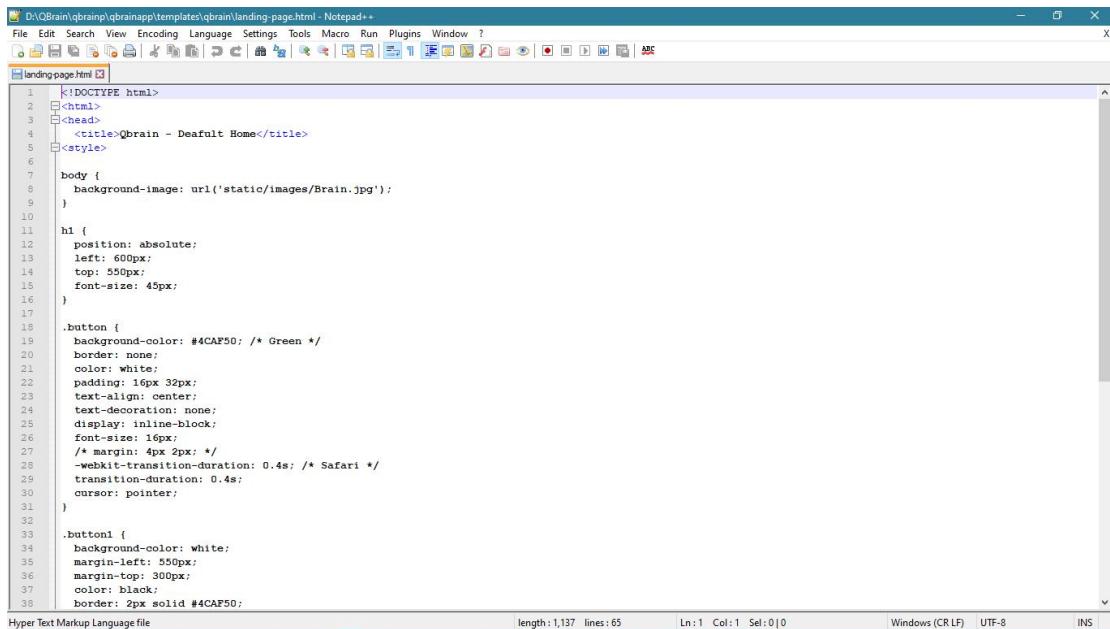
URLS.PY

```
urls.py - D:\QBrain\qbrain\qbrainapp\urls.py (3.7.3)
File Edit Format Run Options Window Help
Examples:
Function views
    1. Add an import: from my_app import views
    2. Add a URL to urlpatterns: path('', views.home, name='home')
Class-based views
    1. Add an import: from other_app.views import Home
    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
"""

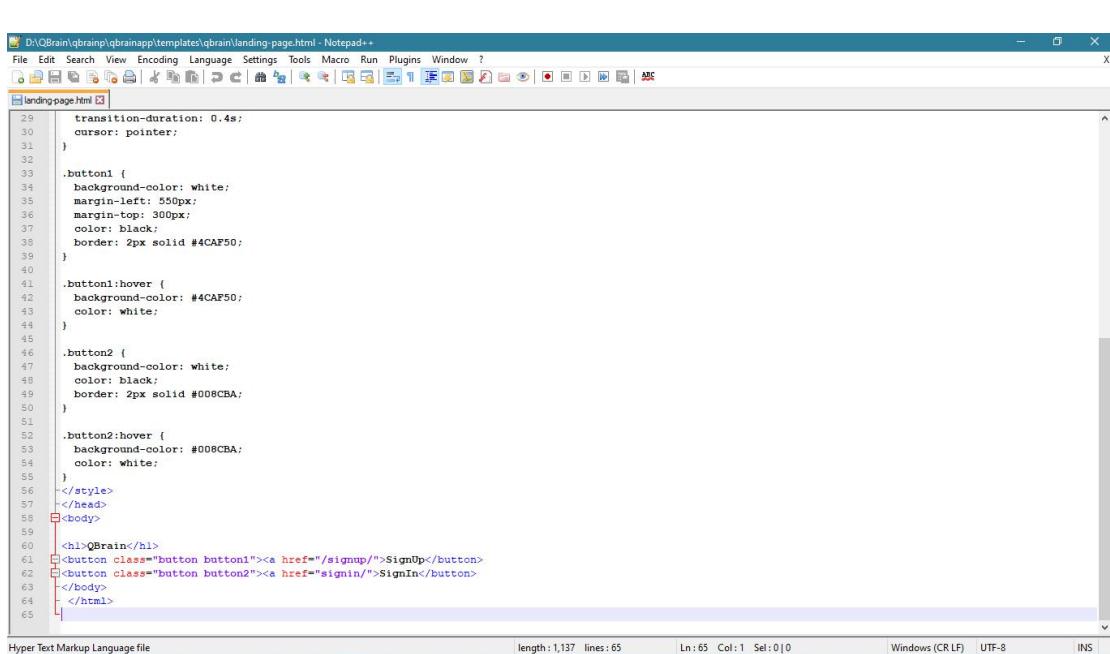
from django.contrib import admin
from django.urls import path
from qbrainapp import views
from django.contrib.auth.views import LoginView
from django.contrib.auth import views as auth_views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('',views.landing,name="landing"),
    path('home/',views.home, name="home"),
    path('showbycat/<int:id>',views.showbycat),
    path('signup/',views.signup, name="signup"),
    path('about/',views.about, name="about"),
    #path('signin/',LoginView.as_view(template_name='diary_app/signin.html', redirect_authenticated_user=True),
    path('signin/', LoginView.as_view(template_name='registration/signin.html', redirect_authenticated_user=True),
    path('contact/',views.contact, name="contact"),
    path('logout/',views.SignOut, name="signout")
    # path('signup/',views.signup, name="signup")
]
```

LANDING-PAGE.HTML

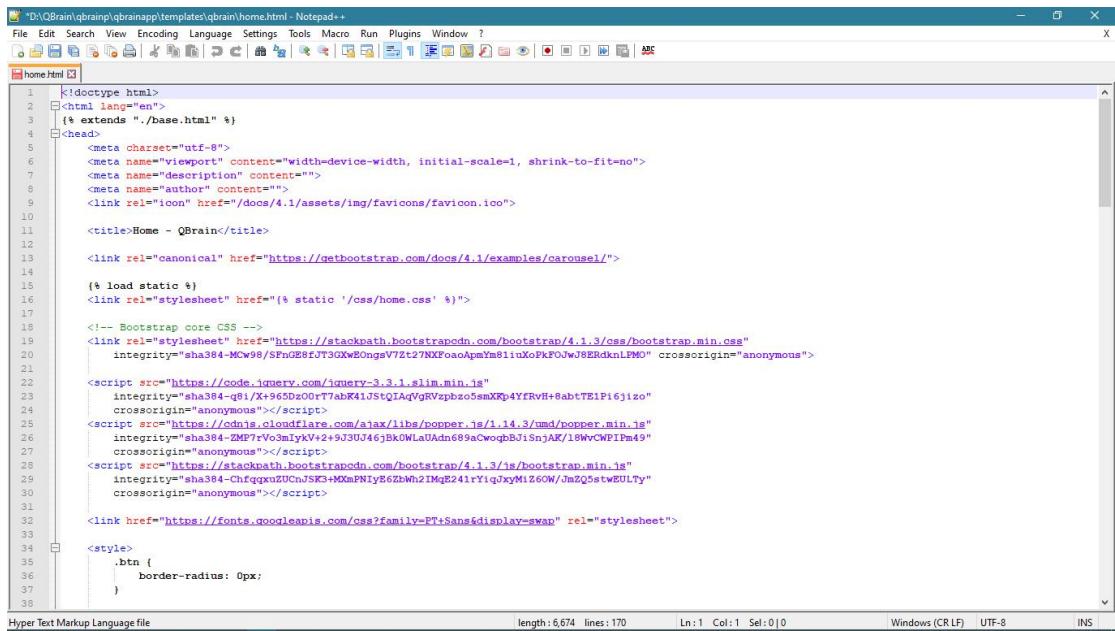


```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Qbrain - Default Home</title>
5 <style>
6
7 body {
8   background-image: url('static/images/Brain.jpg');
9 }
10
11 h1 {
12   position: absolute;
13   left: 600px;
14   top: 550px;
15   font-size: 45px;
16 }
17
18 .button {
19   background-color: #4CAF50; /* Green */
20   border: none;
21   color: white;
22   padding: 16px 32px;
23   text-align: center;
24   text-decoration: none;
25   display: inline-block;
26   font-size: 16px;
27   /* margin: 4px 2px; */
28   /*-webkit-transition-duration: 0.4s; /* Safari */
29   transition-duration: 0.4s;
30   cursor: pointer;
31 }
32
33 .button1 {
34   background-color: white;
35   margin-left: 550px;
36   margin-top: 300px;
37   color: black;
38   border: 2px solid #4CAF50;
39 }
40
41 .button1:hover {
42   background-color: #4CAF50;
43   color: white;
44 }
45
46 .button2 {
47   background-color: white;
48   color: black;
49   border: 2px solid #008CBA;
50 }
51
52 .button2:hover {
53   background-color: #008CBA;
54   color: white;
55 }
56 </style>
57 </head>
58 <body>
59
60   <h1>QBrain</h1>
61   <button class="button button1"><a href="/signup">SignUp</a></button>
62   <button class="button button2"><a href="/signin">SignIn</a></button>
63
64 </body>
65 </html>
```

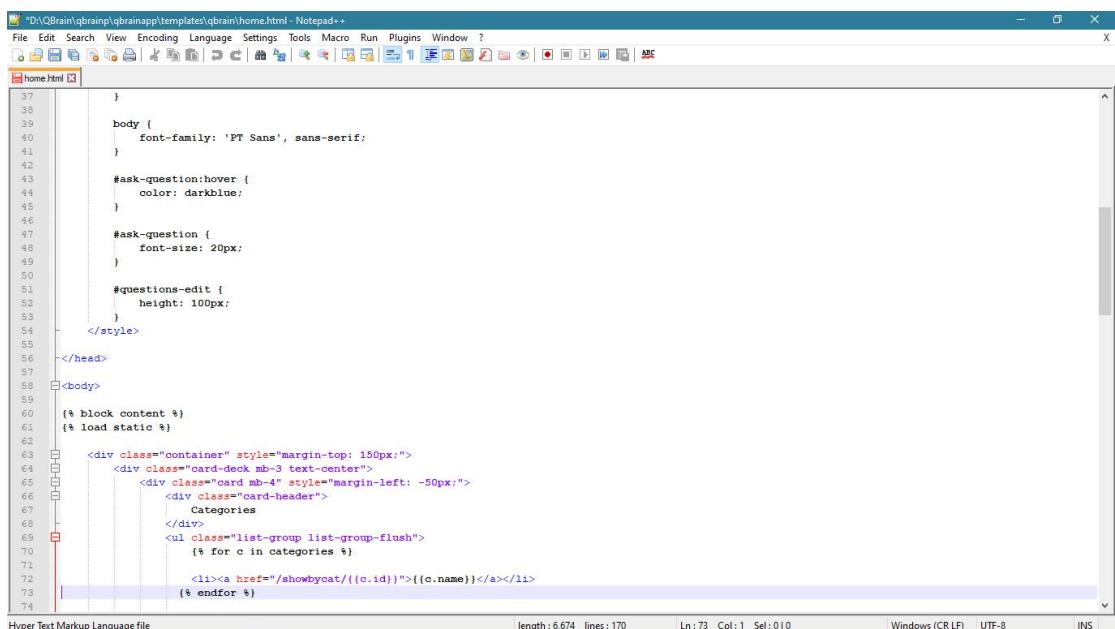


```
29   transition-duration: 0.4s;
30   cursor: pointer;
31 }
32
33 .button1 {
34   background-color: white;
35   margin-left: 550px;
36   margin-top: 300px;
37   color: black;
38   border: 2px solid #4CAF50;
39 }
40
41 .button1:hover {
42   background-color: #4CAF50;
43   color: white;
44 }
45
46 .button2 {
47   background-color: white;
48   color: black;
49   border: 2px solid #008CBA;
50 }
51
52 .button2:hover {
53   background-color: #008CBA;
54   color: white;
55 }
56 </style>
57 </head>
58 <body>
59
60   <h1>QBrain</h1>
61   <button class="button button2"><a href="/signin">SignIn</a></button>
62
63 </body>
64 </html>
```

HOME.HTML



```
1 <!DOCTYPE html>
2 <html lang="en">
3   (& extends "./base.html" *)
4   <head>
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7     <meta name="description" content="">
8     <meta name="author" content="">
9     <link rel="icon" href="/docs/4.1/assets/img/favicons/favicon.ico">
10
11   <title>Home - QBrain</title>
12
13   <link rel="canonical" href="https://getbootstrap.com/docs/4.1/examples/carousel/">
14
15   (& load static *)
16   <link rel="stylesheet" href="(% static '/css/home.css' *)">
17
18   <!-- Bootstrap core CSS -->
19   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
20     integrity="sha384-MCw98/SFnGE8OgI从来没做过这种类型的题目，但是根据你的描述，我猜测你可能是在问如何在Python中使用TensorFlow或其他深度学习库来实现一个简单的图像分类模型。由于你没有提供具体的代码或数据集，我将提供一个基本的示例，帮助你入门。假设我们有一个名为“猫狗识别”的项目，我们希望训练一个模型来识别输入图片中的猫和狗。
```



```
37
38
39   body {
40     font-family: 'PT Sans', sans-serif;
41   }
42
43   #ask-question:hover {
44     color: darkblue;
45   }
46
47   #ask-question {
48     font-size: 20px;
49   }
50
51   #questions-edit {
52     height: 100px;
53   }
54
55 </style>
56
57 </head>
58 <body>
59   (& block content *)
60   (& load static *)
61
62   <div class="container" style="margin-top: 150px;">
63     <div class="card-deck mb-3 text-center">
64       <div class="card mb-4" style="margin-left: -50px;">
65         <div class="card-header">
66           Categories
67         </div>
68         <ul class="list-group list-group-flush">
69           {& for c in categories *}
70
71             <li><a href="#">{{c.name}}</a></li>
72
73           {& endfor *}
74
75     </div>
76   </div>
77
78 </body>
79 </html>
```

```
119 <div class="col-6">
120
121     <form method="post" style="margin-bottom: 25px;">
122         {% csrf_token %}
123         <div class="form-group">
124             <label for="questions_edit">Question</label>
125             <input type="text" name="questions" class="form-control" id="questions_edit" ariaplaceholder="Drop your query here" required="true">
126         </div>
127         <button type="submit" class="btn btn-primary">Submit</button>
128     </form>
129
130     {% for q in questions %}>
131         <div class="card">
132
133             <div class="card-body">
134                 <h5 class="card-title">{{q.questions}}</h5>
135
136             </div>
137
138         </div>
139     {% endfor %}>
140
141         </div>
142         <div class="card mb-4" style="margin-right: -50px;">
143             <div class="card-header">
144                 Profile
145             </div>
146             <div class="card-body">
147                 <h5 class="card-title">Welcome, {{user.username}}!</h5>
148                 <a class="btn btn-md btn-primary" href="{% url 'signout' %}">Sign out</a>
149
150             </div>
151
152         </div>
153
154     </div>
155
156     <!-- Bootstrap core JavaScript -->
```

```
<div class="card-body">
    <h5 class="card-title">{{q.questions}}</h5>
</div>
</div>
{% endfor %}

</div>
<div class="card mb-4" style="margin-right: -50px;">
    <div class="card-header">
        Profile
    </div>
    <div class="card-body">
        <h5 class="card-title">Welcome, {{user.username}}!</h5>
        <a class="btn btn-md btn-primary" href="{% url 'signout' %}">Sign out</a>
    </div>
</div>
</div>


<!-- Placed at the end of the document so the pages load faster -->
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
    integrity="sha384-qJyWI9aIgFkWjXq+8abTTE1Pi6jizo"
    crossorigin="anonymous"></script>
<script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery-slim.min.js"></script>')</script>
<script src="../../assets/js/vendor/popper.min.js"></script>
<script src="../../dist/js/bootstrap.min.js"></script>
<!-- Just to make our placeholder images work. Don't actually copy the next line! -->
<script src="../../assets/js/vendor/loader.min.js"></script>

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

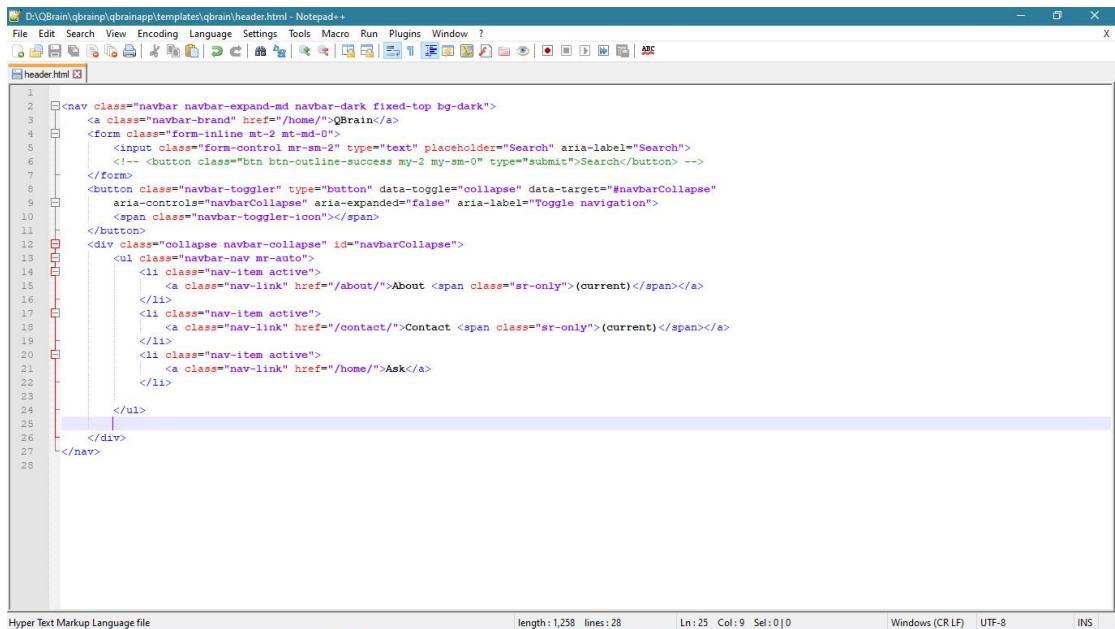
SIGNIN.HTML

```
<!DOCTYPE html>
<html>
<head>
<title>Sign in - QBrain</title>
<meta charset="utf-8">
<link rel="stylesheet" href="{% static '/css/signin.css' %}">
<script src="{% static '/js/signin.js' %}"></script>
</head>
<body>
<div class="signin-form">
<form method="POST">
    {% csrf_token %}
    <div class="form">
        <div class="form-group">
            <input type="text" class="form-control input-lg" name="username" placeholder="Username" required="required">
        </div>
        <div class="form-group">
            <input type="password" class="form-control input-lg" name="password" placeholder="Password" required="required">
        </div>
    {% form.errors %}
    <div class="form-group">
        <button type="submit" class="btn btn-success btn-lg btn-block signup-btn">Sign in</button>
    </div>
    <div class="text-center medium" style="margin-top: 10px;">Not registered?<a href="/signup/">Sign up</a></div>
    </div>
</div>
</body>
</html>
```

SIGNUP.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Sign up - QBrain</title>
<link rel="stylesheet" href="{% static '/css/signup.css' %}">
<script src="{% static '/js/signup.js' %}"></script>
</head>
<body>
<div class="signin-form">
<form method="post">
    {% csrf_token %}
    <div class="form">
        <div class="register-form">
            &#42;{{form.non_field_errors}}
            <div class="form-group">
                <input type="email" class="form-control input-lg" name="email" placeholder="Email" required="required">
            </div>
            <div class="form-group">
                <input type="text" class="form-control input-lg" name="username" placeholder="Username" required="required">
            </div>
            <div class="form-group">
                <input type="password" class="form-control input-lg" name="password1" placeholder="Password" required="required">
            </div>
            &#42;{{form.errors}}
            <div class="form-group">
                <input type="password" class="form-control input-lg" name="password2" placeholder="Confirm password" required="required">
            </div>
        &#42;{{form.errors}}
        <div class="form-group">
            <button type="submit" class="btn btn-success btn-lg btn-block signup-btn">Register</button>
        </div>
        <div class="text-center medium" style="margin-top: 10px;">Already registered?<a href="/signin/">Sign in</a></div>
    </div>
</div>
</body>
</html>
```

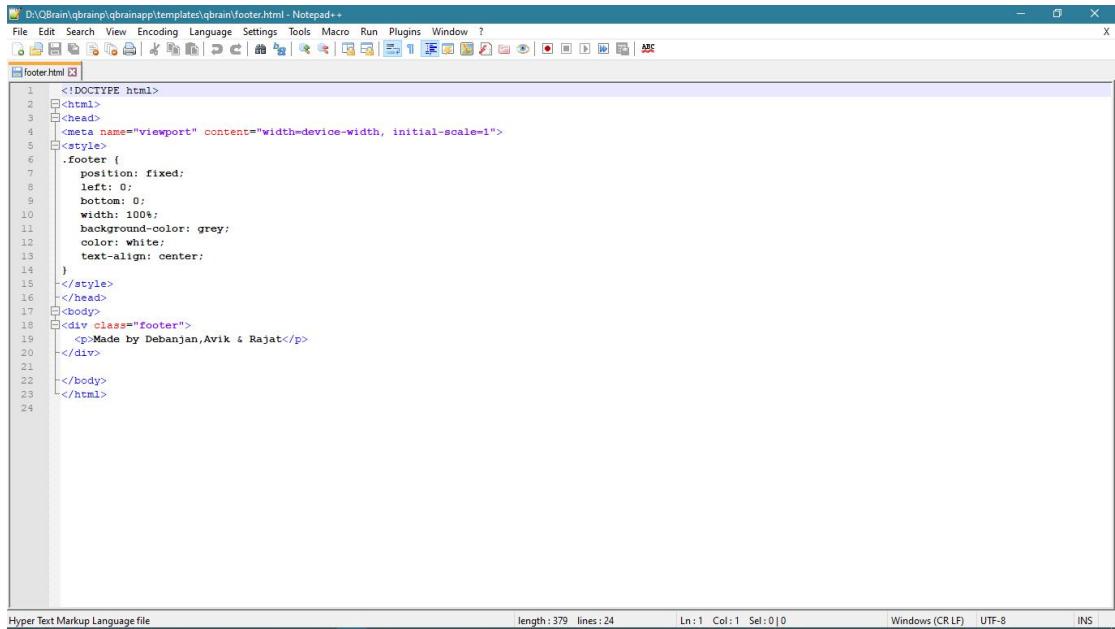
HEADER.HTML



A screenshot of the Notepad++ text editor showing the content of the header.html file. The code is an HTML navbar with a search bar and three menu items: About, Contact, and Ask.

```
1 <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
2   <a class="navbar-brand" href="#">QBrain</a>
3   <form class="form-inline mt-2 mr-3">
4     <input class="form-control mr-sm-2" type="text" placeholder="Search" aria-label="Search">
5     <!-- <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button> -->
6   </form>
7   <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse"
8     aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
9     <span class="navbar-toggler-icon"></span>
10  </button>
11  <div class="collapse navbar-collapse" id="navbarCollapse">
12    <ul class="navbar-nav mr-auto">
13      <li class="nav-item active">
14        <a class="nav-link" href="/about/">About <span class="sr-only">(current)</span></a>
15      </li>
16      <li class="nav-item active">
17        <a class="nav-link" href="/contact/">Contact <span class="sr-only">(current)</span></a>
18      </li>
19      <li class="nav-item active">
20        <a class="nav-link" href="/home/">Ask</a>
21      </li>
22    </ul>
23  </div>
24</nav>
25
26
27
28
```

FOOTER.HTML



A screenshot of the Notepad++ text editor showing the content of the footer.html file. It contains a single footer line with the text "Made by Debanjan, Avik & Rajat".

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta name="viewport" content="width=device-width, initial-scale=1">
5   <style>
6     .footer {
7       position: fixed;
8       left: 0;
9       bottom: 0;
10      width: 100%;
11      background-color: grey;
12      color: white;
13      text-align: center;
14     }
15   </style>
16 </head>
17 <body>
18   <div class="footer">
19     <p>Made by Debanjan, Avik & Rajat</p>
20   </div>
21 </body>
22 </html>
23
24
```

CONCLUSION

This website can be improved in future by many aspects. Each question can have a separate webpage and each user with their own profile page where all their questions and those questions they answered will be there. The website can be launched globally by taking a proper privacy policy and copyright and purchasing a domain/host name.