NETWORKZ SYSTEM

MARTHANDAM

Name of Course: ***Python Full Stack-Developer Project***

NAME OF THE PROJECT

BRAIN NEURONS

Submitted by: ***Sajin S K***

Guided by: ***Shaline S***

Content

Abstract

Program

Output

Conclusion

Project Abstract:

Python is an interpreted high-level general-purpose programming language. In this project, we used the Python framework Django to develop a website and also utilized the sqlite3 database for authentication.

I developed the project using PyCharm, Python's official software development platform. The website is designed for memorization purposes and primarily targeted towards students. Anyone can visit this website with registration. The data will be stored on the sqlite3 database.

Program:

Django Installation:

pip install Django

Create project:

django-admin startproject project.

Create Application:

python manage.py startapp app

To Run:

python manage.py runserver

Program Files:

This is the main file to run the code here we have added a sqlite3 database connection

```
from pathlib import Path
import os

BASE_DIR = Path(__file__).resolve().parent.parent

SECRET_KEY = 'django-insecure-*7wkswj1w13u5wf3o#jph*+4n)dnf0m7#$!t!dot6f*qcf9*x9'

DEBUG = True

ALLOWED_HOSTS = []

INSTALLED_APPS = [
   'django.contrib.admin',
   'django.contrib.auth',
   'django.contrib.contenttypes',
   'django.contrib.sessions',
   'django.contrib.messages',
   'django.contrib.staticfiles',
   'authentication'
]
MIDDLEWARE = [
   'django.middleware.security.SecurityMiddleware',
   'django.contrib.sessions.middleware.SessionMiddleware',
   'django.middleware.common.CommonMiddleware',
   'django.middleware.csrf.CsrfViewMiddleware',
   'django.contrib.auth.middleware.AuthenticationMiddleware',
   'django.contrib.messages.middleware.MessageMiddleware',
   'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'gfg.urls'

TEMPLATES = [
   {
      'BACKEND': 'django.template.backends.django.DjangoTemplates',
      'DIRS': [os.path.join(BASE_DIR / 'templates')],
      'APP_DIRS': True,
      'OPTIONS': {
         'context_processors': [
            'django.template.context_processors.debug',
            'django.template.context_processors.request',
            'django.contrib.auth.context_processors.auth',
            'django.contrib.messages.context_processors.messages',
         ],
```

```
        },
    },
]

WSGI_APPLICATION = 'gfg.wsgi.application'

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


AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True


STATIC_URL = '/static/'

STATICFILES_DIRS = [
    BASE_DIR / "authentication/static",
]

MEDIA_URL = '/media/'
MEDIA_ROOT=os.path.join(BASE_DIR,'media')


DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

Project / urls.py:

This is the python file which is handling urls of the application.

```
from django.contrib import admin
from django.urls import path, include
from authentication import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('authentication.urls')),
]
```

Create urls.py file in Application

gfg/ urls.py:

This tells Django to search for URL patters in the file gfg/urls.py. For example, A URL request to authentication/login/ will match with the first URL pattern, As a result, Django will call the function views.

```
from django.urls import path
from . import views
from .views import save_label
from .views import view_labels

from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('', views.index, name='index'),
    path('homepage/', views.homepage, name='homepage'),
    path('registration/', views.registration, name='registration'),
    path('createfolder/', views.createfolder, name='createfolder'),
    path('save_label/', save_label, name='save_label'),
    path('view_labels/', view_labels, name='view_labels'),
    path('randomquestion/', views.randomquestion, name='randomquestion'),
    path('random_question/', views.random_question, name='random_question'),
    path('data_list/', views.data_list, name='data_list'),
    path('get_all_data/', views.get_all_data, name='get_all_data'),
    path('updateData/<int:id>', views.updateData, name='updateData'),
    path('deleteData/<int:id>', views.deleteData, name='deleteData'),
]

urlpatterns += static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)
```

Views.py:

As per Django Documentation, A view function is a Python function that takes a Web request and returns a Web response. This respone can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML, document, or an image, anything that a web browser can display

```
from django.shortcuts import redirect, render
from django.contrib.auth import login
from django.contrib import messages
from .forms import RegistrationForm
from .models import Registration
from .models import Label
from .forms import LabelForm
from django.http import JsonResponse
import random

def index(request):
    if request.method == 'POST':
        email = request.POST.get('email', '')
        password = request.POST.get('password', '')

        user = Registration.objects.filter(email=email, password=password).first()

        if user is not None:
            login(request, user)
            messages.success(request, 'Login successful!')
            return redirect('homepage')
        else:
            messages.error(request, 'Invalid email or password. Please try again.')
    return render(request, 'index.html')


def registration(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, 'Account created successfully.')
            return redirect('index')  # Redirect to a success page or another appropriate URL
        else:
            messages.error(request, 'This email already has an account, so the account was not created. Please try
another email.')
    else:
        form = RegistrationForm()

    return render(request, 'registration.html', {'form': form})

def homepage(request):
    return render(request, 'homepage.html')
```

```
def createfolder(request):
    return render(request, 'createfolder.html')

def randomquestion(request):
    return render(request, 'randomquestion.html')

def view_folder(request):
    # Your view logic here
    return render(request, 'view_folder.html')

def save_label(request, *args, **kwargs):
    if request.method == 'POST':
        form = LabelForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return JsonResponse({'status': 'success', 'message': 'Label data saved successfully.'})
        else:
            return JsonResponse({'status': 'error', 'message': 'Invalid form data.'})
    else:
        return JsonResponse({'status': 'error', 'message': 'Invalid request method.'})


def view_labels(request):
    labels = Label.objects.values_list('name', flat=True).distinct()
    return JsonResponse({'labels': list(labels)})




def random_question(request):
    selected_label = request.GET.get('label', None)

    # Fetch all questions and answers without using distinct()
    labels = Label.objects.filter(name=selected_label).values('question', 'answer', 'image') if selected_label else
Label.objects.values('question', 'answer', 'image')

    if labels:
        random_entry = random.choice(labels)
        return JsonResponse({
            'question': random_entry['question'],
            'answer': random_entry['answer'],
            'image': random_entry['image']  # Assuming you have a field named 'image' in the Label model
        })
    else:
        return JsonResponse({'question': 'No questions found.', 'answer': '', 'image': ''})


def data_list(request):
    labels = Label.objects.values_list('name', flat=True).distinct()
    return JsonResponse({'labels': list(labels)})


def get_all_data(request):
    labels = Label.objects.all()

    serialized_labels = []
```

```
    for label in labels:
        serialized_labels.append({
            'id': label.id,
            'name': label.name,
            'question': label.question,
            'answer': label.answer,
            'image': label.image.url if label.image else None,
        })

    return render(request, 'AllData.html', {'labels': serialized_labels})


def updateData(request, id):
    mydata=Label.objects.get(id=id)
    if request.method=='POST':

        name=request.POST['name']
        question=request.POST['question']
        answer=request.POST['answer']
        image=request.POST['image']

        mydata.name=name
        mydata.question=question
        mydata.answer=answer
        mydata.image=image
        mydata.save()

        return redirect('homepage')

    return render(request, 'update.html',{'label' :mydata})

def deleteData(request, id):
    mydata=Label.objects.get(id=id)
    mydata.delete()
    return redirect('homepage')
```

models.py:

Django web applications access and manage data through Python objects referred to as models. Models define the structure of stored data, including the field types and possibly also their maximum size, default selection list options, help text for documentation, label text for forms, etc.

```
from django.db import models
from django.contrib.auth.models import AbstractBaseUser
```

```python
class Registration(models.Model):
    name = models.CharField(max_length=100)
    email = models.EmailField(unique=True)
    password = models.CharField(max_length=100)
    last_login = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.name


class Label(models.Model):
    name = models.CharField(max_length=255)
    question = models.TextField()
    answer = models.TextField()
    image = models.ImageField(upload_to='label_images', blank=True, null=True)

    def __str__(self):
        return self.name
```

Create forms.py file in Application

forms.py

forms.py is where the django documentation recommends you place all your forms code; to keep your code easily maintainable.

```python
from django import forms
from .models import Registration
from .models import Label

class RegistrationForm(forms.ModelForm):
    class Meta:
        model = Registration
        fields = '__all__'

class LabelForm(forms.ModelForm):
    class Meta:
        model = Label
        fields = ('name', 'question', 'answer', 'image')
```

admin.py:

The admin.py file is used to display your models in the Django admin panel. You can also customize your admin panel.

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

# Register your models here.
admin.site.register(Label)
```

Create template directory file in Application:

Django template engine is used to separate the design from the python code and allows us to build dynamic web pages.

Create static directory file in Application:

Static files collects static files from each of your applications into a single location that can easily be served in production.

From template directory:

Inc:

registration.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Brain Neurons</title>
  {% load static %}
  <link rel="stylesheet" href="{% static 'css/style0.css'%}">
```

```
</head>
<body>

  <div class="main">
    <div class="navbar">
      <div class="icon">
        <h2 class="logo">BrainNeurons</h2>
      </div>

      <div class="menu">
        <ul>
          <li><a href="#">HOME</a></li>
          <li><a href="#">ABOUT</a></li>
          <li><a href="#">SERVICE</a></li>
          <li><a href="#">DESIGN</a></li>
          <li><a href="#">CONTACT</a></li>
        </ul>
      </div>
    </div>
    <div class="content">
      <h1>Brain Neurons &amp;amp; <br><span>Development</span> <br>Training</h1>
      <p class="par">To find answers, life depends on asking questions. <br> Asking questions is the
simplest and most effective way of learning. <br> - Richard Branson </p>
      {% if messages %}
        <ul class="messages">
          {% for message in messages %}
            <li{% if message.tags %} class="{{ message.tags }}"{% endif %}>{{ message }}</li>
          {% endfor %}
        </ul>
      {% endif %}

      <div class="form1">
        <h2>Registration Form</h2>
        <form autocomplete="off" method="POST" action="{% url 'registration' %}">
          {% csrf_token %}
          <input type="text" name="name" placeholder="Enter Your Name">
          <input type="email" name="email" placeholder="Enter Your Email">
          <input type="password" name="password" placeholder="Enter Your Password">
          <button class="btnn" type="submit">Register</button>
        </form>


        <div class="icons">
          <a href="#"><ion-icon name="logo-facebook"></ion-icon></a>
          <a href="#"><ion-icon name="logo-instagram"></ion-icon></a>
          <a href="#"><ion-icon name="logo-twitter"></ion-icon></a>
          <a href="#"><ion-icon name="logo-google"></ion-icon></a>
          <a href="#"><ion-icon name="logo-skype"></ion-icon></a>
        </div>
      </div>
    </div>
```

```
    </div>
    <script src="https://unpkg.com/ionicons@5.4.0/dist/ionicons.js"></script>

</body>
</html>
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Brain Neurons</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/style.css'%}">
</head>
<body>
    <div class="main">
        <div class="navbar">
            <div class="icon">
                <h2 class="logo">BrainNeurons</h2>
            </div>

            <div class="menu">
                <ul>
                    <li><a href="#">HOME</a></li>
                    <li><a href="#">ABOUT</a></li>
                    <li><a href="#">SERVICE</a></li>
                    <li><a href="#">DESIGN</a></li>
                    <li><a href="#">CONTACT</a></li>
                </ul>
            </div>
        </div>
        <div class="content">
            <h1>Brain Neurons &amp;amp; <br><span>Development</span> <br>Training</h1>
            <p class="par">To find answers, life depends on asking questions. <br> Asking questions is the
simplest and most effective way of learning. <br> - Richard Branson </p>

            <button class="cn" type="submit"> <a href="registration/">JOIN US</a></button>
            {% if messages %}
                <div class="popup">
                    <ul class="messages">
                        {% for message in messages %}
                            <li{% if message.tags %} class="{{ message.tags }}"{% endif %}>{{ message }}</li>
                        {% endfor %}
                    </ul>
                </div>
            {% endif %}

            {% if msg %}
```

```
        <p>{{ msg }}</p>
        {% endif %}

        <div class="form">
          <h2>Login Here</h2>
          <form autocomplete="off" method="POST" action="{% url 'index' %}">
            {% csrf_token %}
            <input type="email" name="email" placeholder="Enter Email Here">
            <input type="password" name="password" placeholder="Enter Password Here">
            <button class="btnn"><a>Login</a></button>
          </form>
          <p class="link">Don't have an account<br>
            <a href="#">JOIN US</a> here</p>
          <p class="liw">Log in with</p>

          <div class="icons">
            <a href="#"><ion-icon name="logo-facebook"></ion-icon></a>
            <a href="#"><ion-icon name="logo-instagram"></ion-icon></a>
            <a href="#"><ion-icon name="logo-twitter"></ion-icon></a>
            <a href="#"><ion-icon name="logo-google"></ion-icon></a>
            <a href="#"><ion-icon name="logo-skype"></ion-icon></a>
          </div>
        </div>
      </div>
    </div>

    <script src="https://unpkg.com/ionicons@5.4.0/dist/ionicons.js"></script>

</body>
</html>
```

homepage.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Brain Neurons</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/style1.css'%}">

</head>
<body>
    <div class="main">
      <div class="navbar">
        <div class="icon">
          <h2 class="logo">BrainNeurons</h2>
        </div>
```

```
        <div class="menu">
          <ul>
            <li><a href="#">HOME</a></li>
            <li><a href="#">ABOUT</a></li>
            <li><a href="#">SERVICE <i class="users"></i></a>
              <div class="sub-menu-1">
                <ul>
                  <li><a href="/createfolder/">Create Questions</a></li>
                  <li><a href="/randomquestion/">Practice</a></li>
                  <li><a href="/get_all_data/">Data</a></li>
                </ul>
              </div>
            </li>
            <li><a href="#">DESIGN</a></li>
            <li><a href="#">CONTACT</a></li>
          </ul>
        </div>
      </div>
      <div class="content">
        <h1>Brain Neurons &amp;amp; <br><span>Development</span> <br>Training</h1>
        <p class="par">To find answers, life depends on asking questions. <br> Asking questions is the
simplest and most effective way of learning. <br> - Richard Branson </p>
      </div>
      {% if messages %}
          <div class="popup">
            <ul class="messages">
              {% for message in messages %}
                <li{% if message.tags %} class="{{ message.tags }}"{% endif %}>{{ message }}</li>
              {% endfor %}
            </ul>
          </div>
      {% endif %}
    </div>
</body>
</html>
```

createfolder.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Brain Neurons</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/style2.css' %}">
    <script src="{% static 'js/createFolder.js' %}"></script>
    <script src="{% static 'js/viewLabels.js' %}"></script>
</head>
<body>
    <div class="navbar">
        <div class="icon">
```

```
        <h2 class="logo">BrainNeurons</h2>
      </div>

      <div class="note-form">
        <form>
          <textarea id="note-input" class="note-input" placeholder="Label Name"
maxlength="50"></textarea>
          <div id="label-container"></div>

        </form>
      </div>
    </div>

    <div class="main">
      <div class="scrollable">
        <div id="wall"></div>
        <div id="data-container"></div>
      </div>
    </div>

</body>
</html>
```

randomquestion.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Brain Neurons</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/style3.css' %}">
    <script src="{% static 'js/QuestionAnswer.js' %}"></script>
</head>
<body>
    <div class="navbar">
      <div class="icon">
        <h2 class="logo">BrainNeurons</h2>
      </div>
    </div>
    <div class="main">
      <div class="question-box">
        <p><span id="question-text">Question</span></p>

      </div>

      <div class="answer-box">
        <textarea id="user-answer" placeholder="Type your answer here" rows="5"></textarea>
```

```
        <button class="button" id="btn-next-question">Random Question</button>
        <button class="button" id="btn-check-answer">Check Answer</button>
        <button class="button" id="btn-show-answer">Show Answer</button>
        <p>Correct Answer: <span id="correct-answer-text"></span></p>
        <p class="correct-answer-text">Correct Answer: <span id="correct-answer-text"></span></p>
      </div>

      <div class="image-box">
        <img id="question-image" src="" alt="Question Image" width="300" height="400">

      </div>

      <select id="data-option">
      </select>
    </div>

    <script>

      document.getElementById('btn-next-question').addEventListener('click',
fetchRandomQuestionWithLabel);
      document.getElementById('btn-check-answer').addEventListener('click', checkAnswer);
      document.getElementById('btn-show-answer').addEventListener('click', showAnswer);

      fetchLabels();
    </script>
</body>
</html>
```

update.html

```
<!DOCTYPE html>
<html>
<head>
  <title>All Data</title>
  {% load static %}
  <!-- Add Bootstrap CSS -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

  <!-- Custom CSS -->
  <style>
    body {
      font-family: "Times New Roman", Times, serif;
    }

    em {
      font-style: italic;
    }
  </style>
</head>
<body>
```

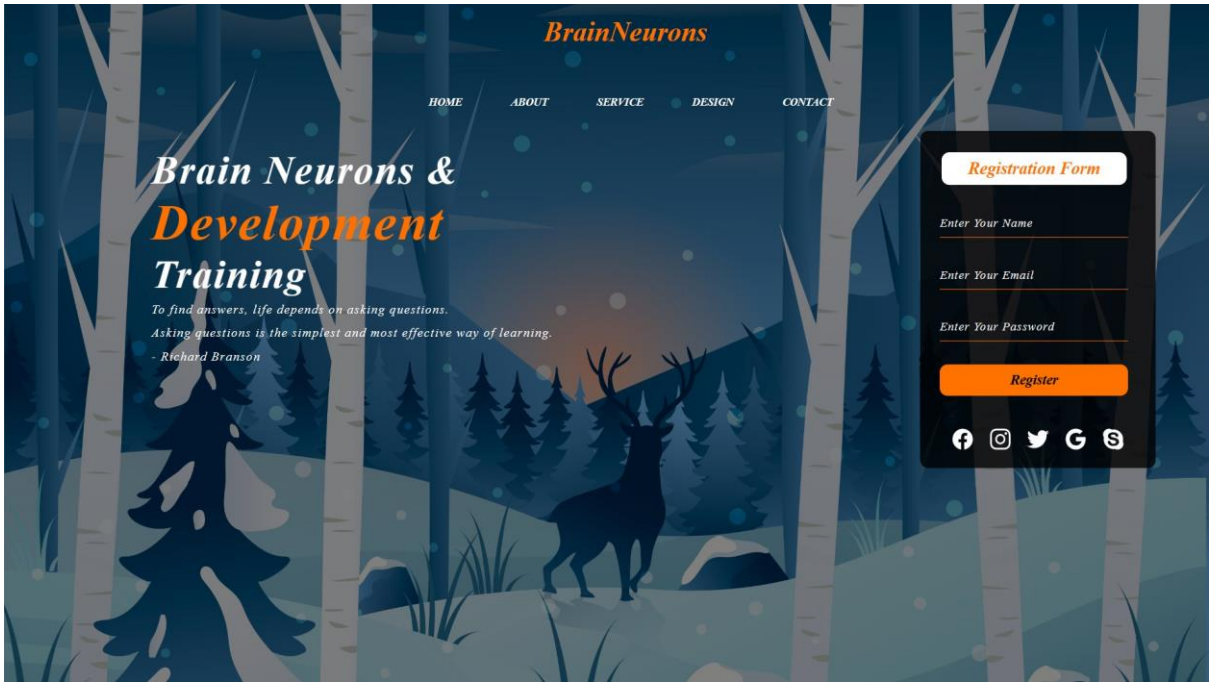
```
  <div class="container">
    <div class="col-md-offset-3 col-md-6">
      <form action="/updateData/{{label.id}}" method="POST" autocomplete="off" class="form-horizontal">
        {% csrf_token %}
        <h3 class="page-header text-primary text-center">Edit Data</h3>
        <div class="form-group">
          <label for="name" class="col-sm-2 control-label">Name</label>
          <div class="col-sm-10">
            <input type="text" class="form-control" value="{{ label.name }}" name="name" placeholder="Enter your name">
          </div>
        </div>
        <div class="form-group">
          <label for="question" class="col-sm-2 control-label">Question</label>
          <div class="col-sm-10">
            <textarea class="form-control" name="question" placeholder="Ask your question" rows="4">{{ label.question }}</textarea>
          </div>
        </div>
        <div class="form-group">
          <label for="answer" class="col-sm-2 control-label">Answer</label>
          <div class="col-sm-10">
            <textarea class="form-control" name="answer" placeholder="Enter the answer" rows="4">{{ label.answer }}</textarea>
          </div>
        </div>
        <div class="form-group">
          <label for="image" class="col-sm-2 control-label">Image</label>
          <div class="col-sm-10">
            <input type="file" class="form-control" name="image">
          </div>
        </div>
        <div class="form-group">
          <div class="col-sm-offset-2 col-sm-10">
            <input type="submit" value="Update" class="btn btn-primary btn-block">
          </div>
        </div>
      </form>
    </div>
  </div>

  <!-- Add Bootstrap JS -->
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</body>
</html>
```

AllData.html

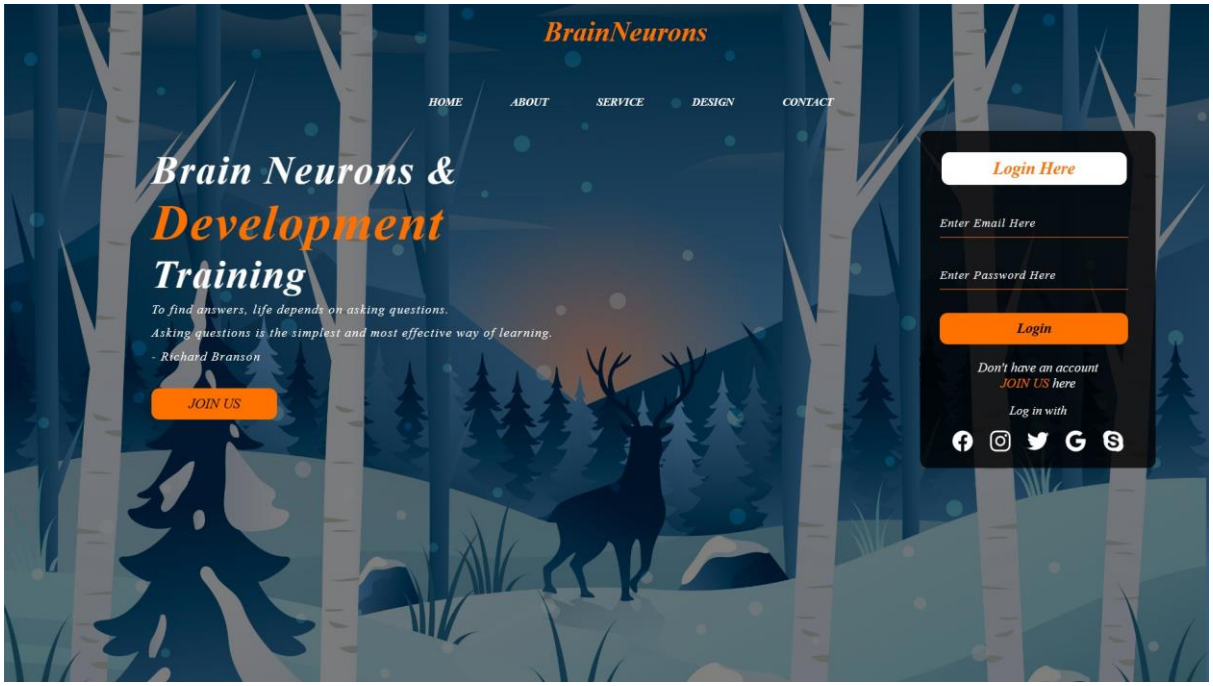
```
<!DOCTYPE html>
<html>
<head>
  <title>All Data</title>
  {% load static %}
  <link rel="stylesheet" href="{% static 'css/style4.css' %}">
</head>
<body>
  <h1>All Data</h1>
  <table>
    <tr>
      <th>ID</th>
      <th>Name</th>
      <th>Question</th>
      <th>Answer</th>
      <th>Image</th>
      <th>Update</th>
      <th>Delete</th>
    </tr>
    <!-- Assuming 'labels' is an array of objects with properties like 'id', 'name', 'question', 'answer',
and 'image' -->
    {% for label in labels %}
    <tr>
      <td>{{ label.id }}</td>
      <td>{{ label.name }}</td>
      <td>{{ label.question }}</td>
      <td>{{ label.answer }}</td>

      <td>
        {% if label.image %}
          <img src="{{ label.image }}" alt="{{ label.name }}" height="50">
        {% else %}
          No Image
        {% endif %}
      </td>
      <!-- Assuming there's a server-side endpoint to handle the update, e.g.,
'/updateData/{{label.id}}' -->
      <td><a class="btn btn-success" href="/updateData/{{label.id}}">Update</a></td>
      <td><a class="btn btn-danger" href="/deleteData/{{label.id}}">Delete</a></td>
    </tr>
    {% endfor %}
  </table>
</body>
</html>
```

Output:

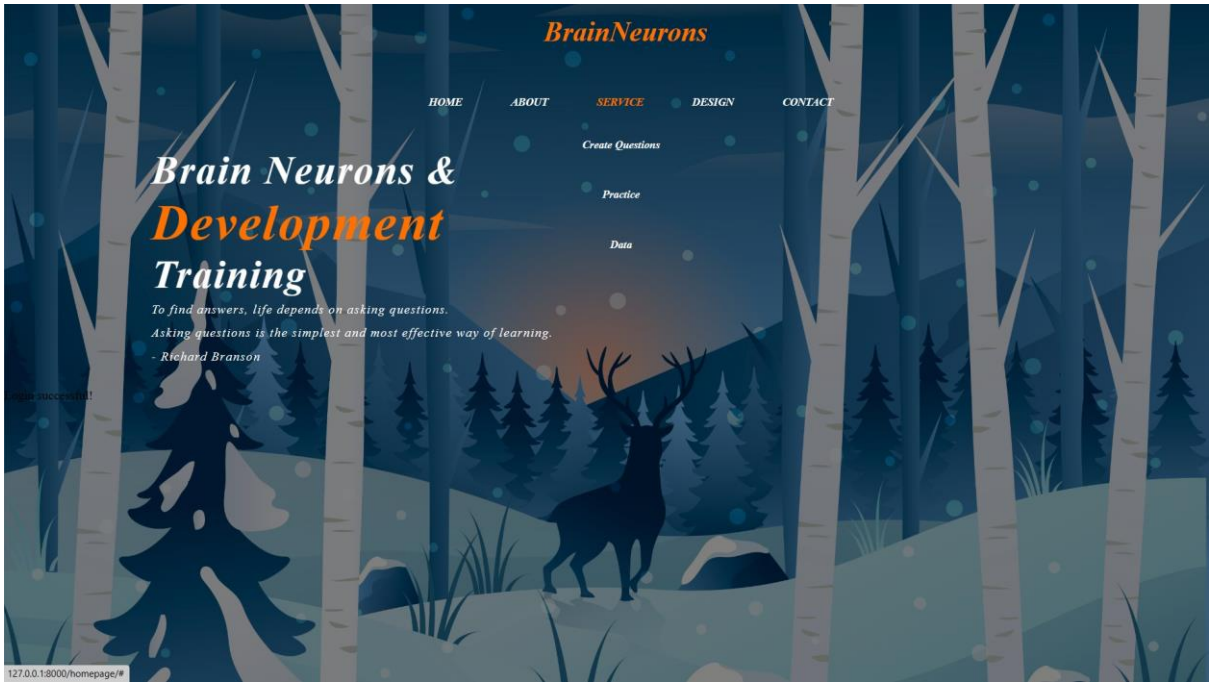
Registration page:



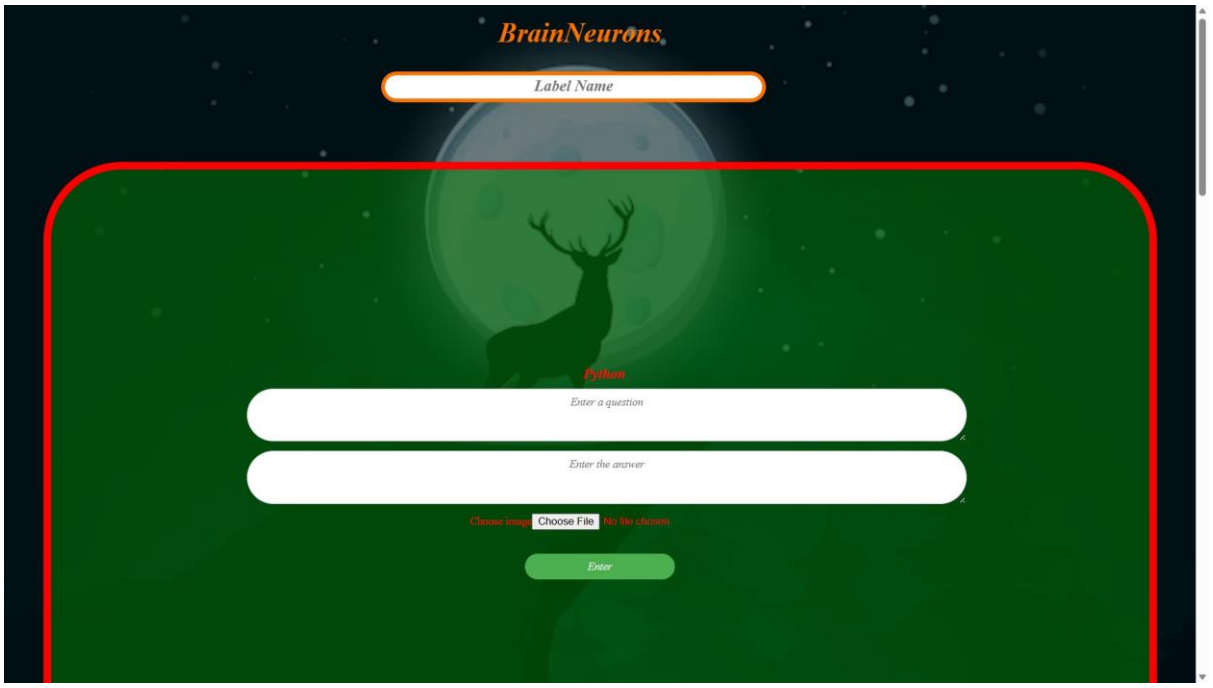
Login Page :

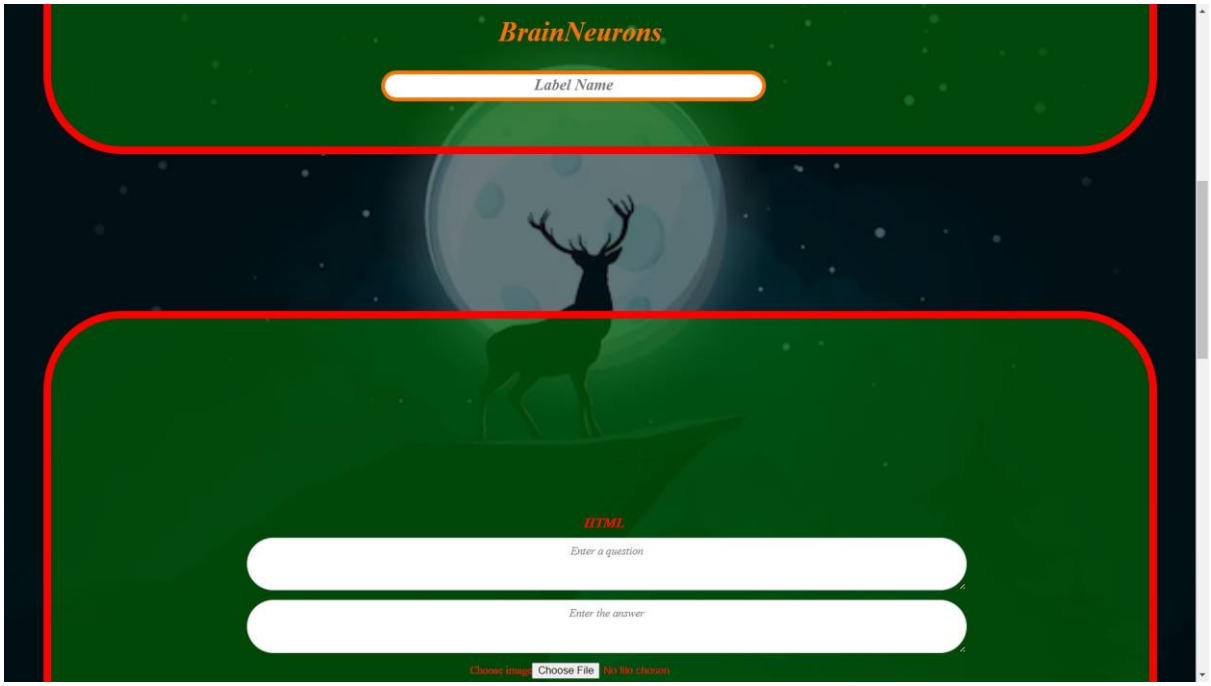


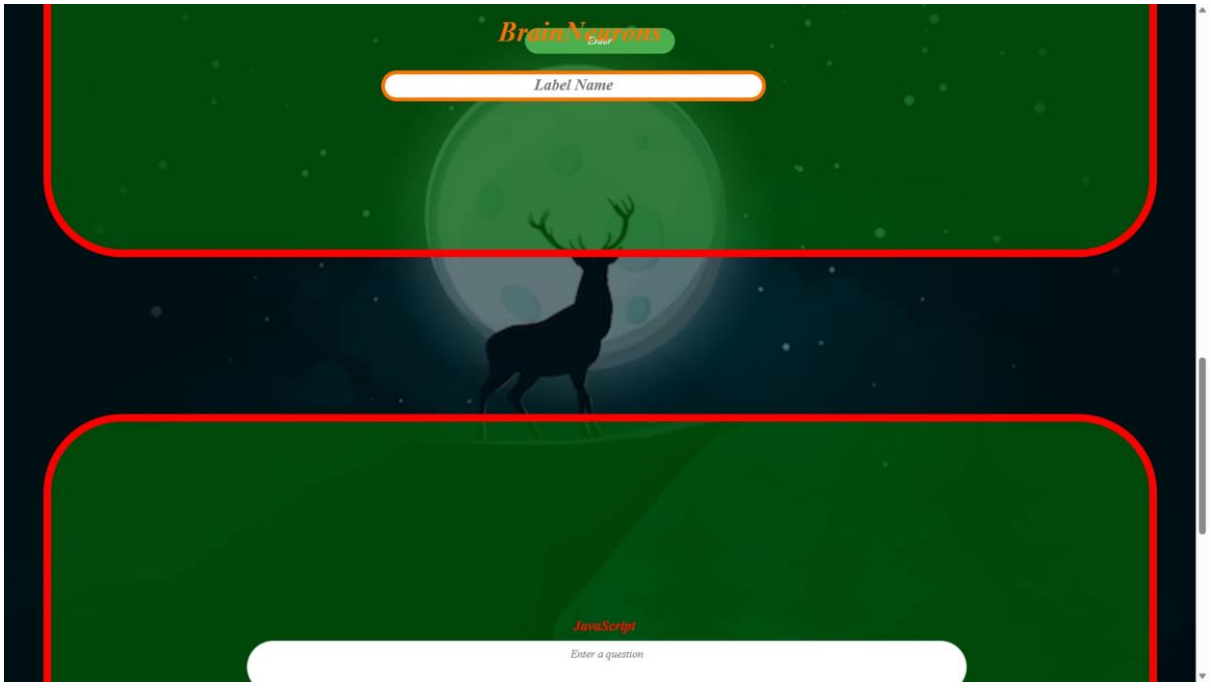
Home Page:

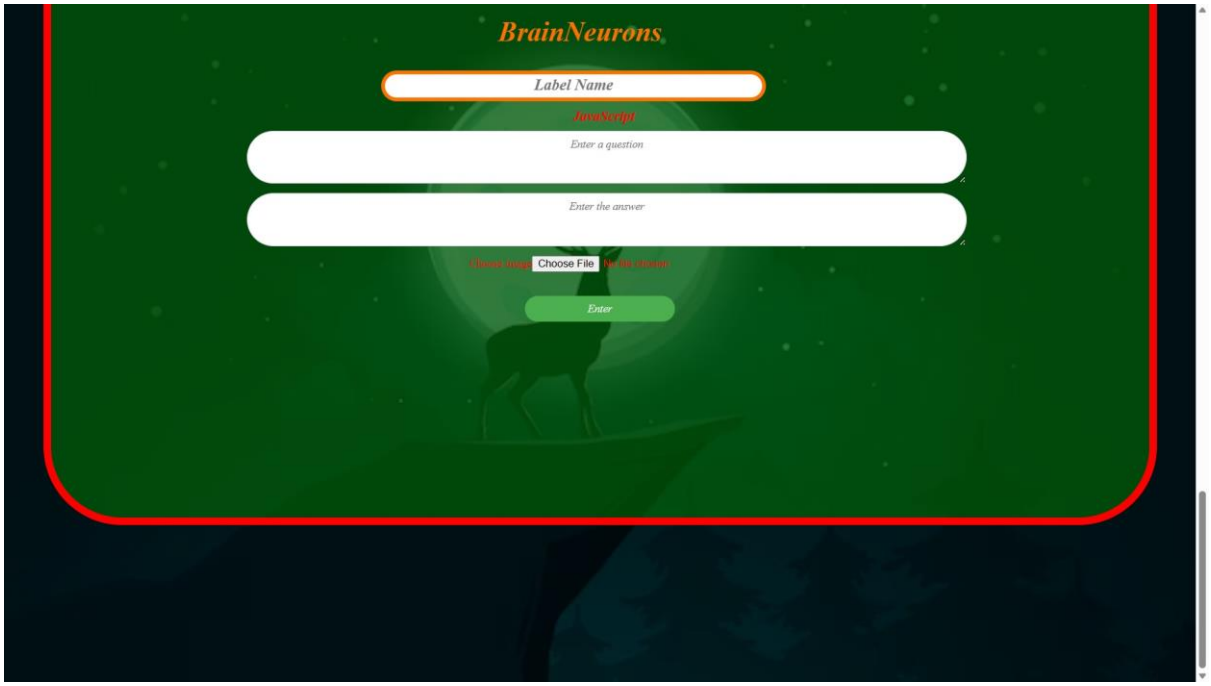


Create Question:

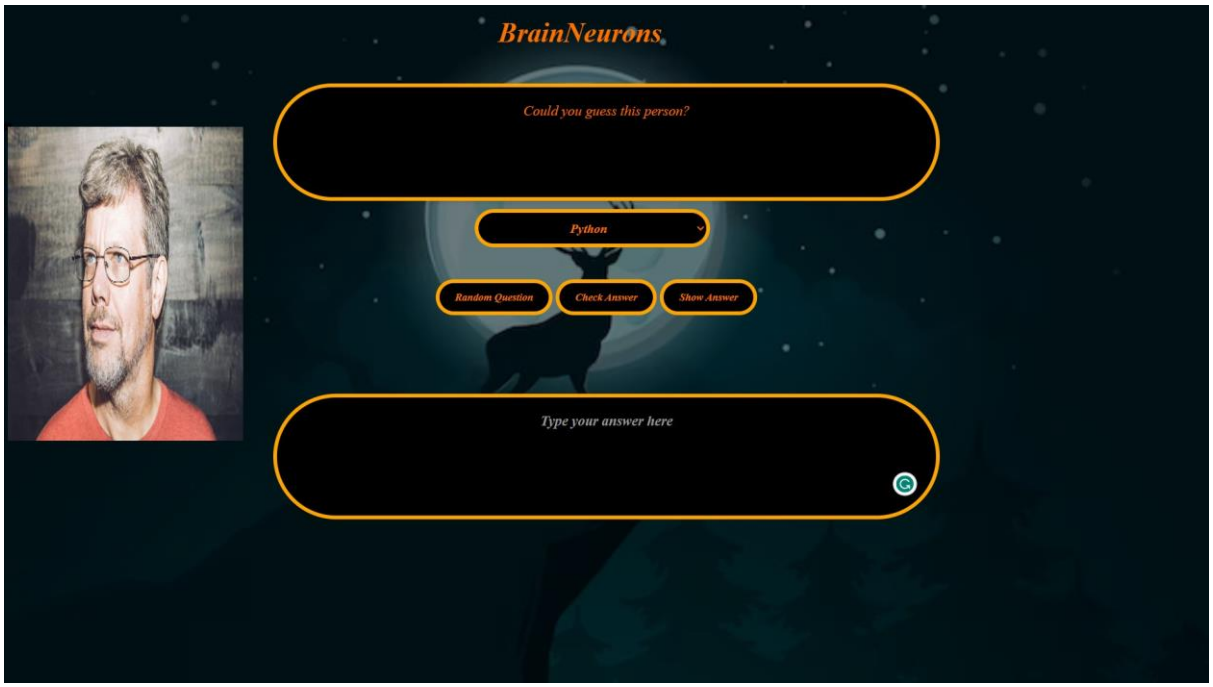



BrainNeurons
Label Name
HTML
Enter a question
Enter the answer
Choose image
Choose File
No file chosen


BrainNeurons
Label Name
JavaScript
Enter a question


BrainNeurons
Label Name
JavaScript
Enter a question
Enter the answer
Choose File
Enter

Practice:


BrainNeurons
Could you guess this person?
Python
Random Question
Check Answer
Show Answer
Type your answer here

Data:

All Data

ID	Name	Question	Answer	Image	Update	Delete
437	Python	Who is the founder of Python?	Guido van Rossum	No Image	Update	Delete
438	HTML	Who is the founder of HTML?	Tim Berners-Lee	No Image	Update	Delete
439	JavaScript	Who is the founder of JavaScript?	Brendan Eich	No Image	Update	Delete
440	Python	Could you guess this person?	Guido van Rossum		Update	Delete
441	HTML	Could you guess this person?	Tim Berners-Lee		Update	Delete
442	JavaScript	Could you guess this person?	Brendan Eich		Update	Delete
443	Python	In Python, which built-in function can be used to find the length of a list or a string?	len()	No Image	Update	Delete
444	Python	What will be the output of the following Python code?	4		Update	Delete
445	Python	In Python, which keyword is used to create a function?	def	No Image	Update	Delete
446	Python	What is the result of the following expression?	1		Update	Delete
447	Python	How do you open and read a file named "data.txt" in Python?	file = open("data.txt", "r")	No Image	Update	Delete
448	HTML	What does HTML stand for?	Hyper Text Markup Language	No Image	Update	Delete
449	HTML	Which HTML element is used to define the title of a webpage?	<title>	No Image	Update	Delete
450	HTML	What is the correct HTML tag for creating a hyperlink?	<a>	No Image	Update	Delete

Update :

Edit Data

Name Python

Question Who is the founder of Python?

Answer Guido van Rossum

Image Choose File No file chosen

Update

Conclusion:

The Django framework provides me with a simple and reliable way to create memorization. It offers powerful tools for dealing with the database and web pages, helping me learn a lot about website development using Django. Within the Django framework, I have achieved success. Once this system passes the testing phase, it can be used to serve people and instructors.