Priyanandhana S
2313712080018

WEB APPLICATION WITH CRUD OPERATIONS

Aim:

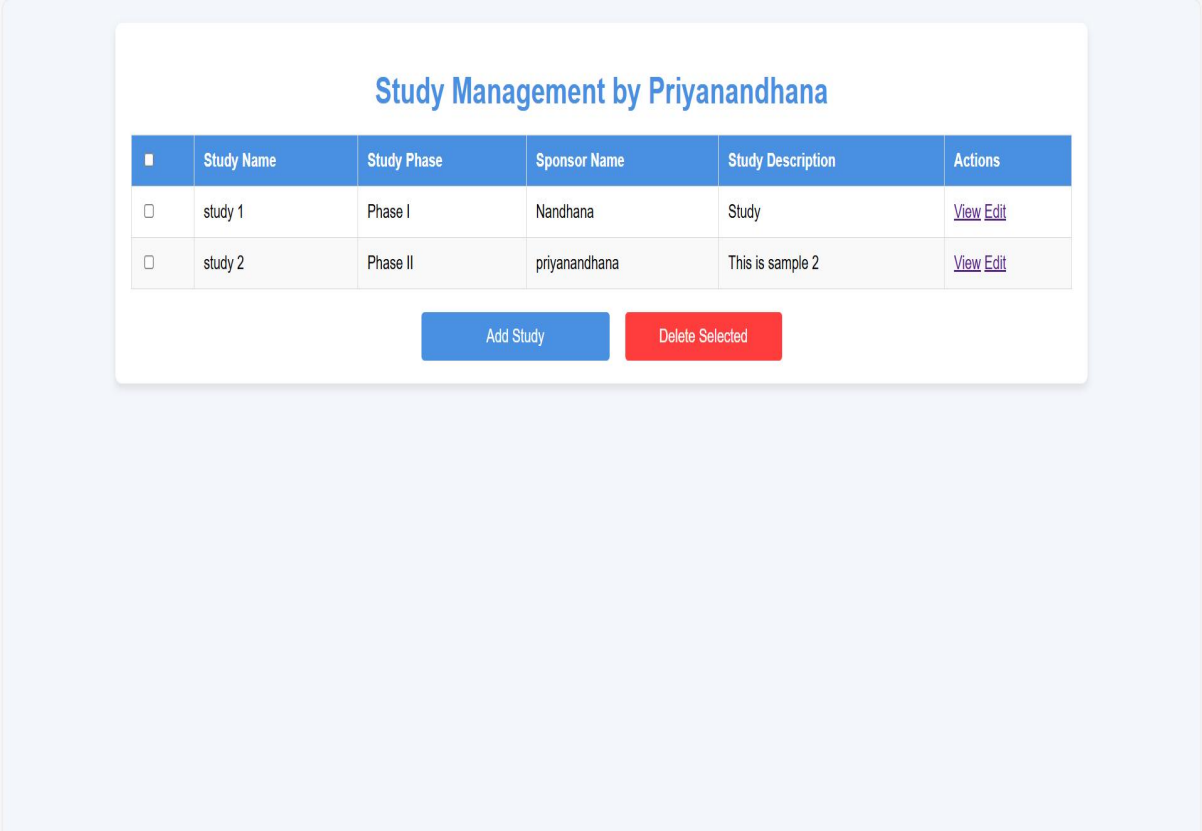
To build a simple and a small web application which can perform the below CRUD operations (CREATE, READ, UPDATE, DELETE).

Technical skills used :

- Language: Python, HTML, CSS
- Framework: Django
- Database: MySQL
- Logging
- Exception handling

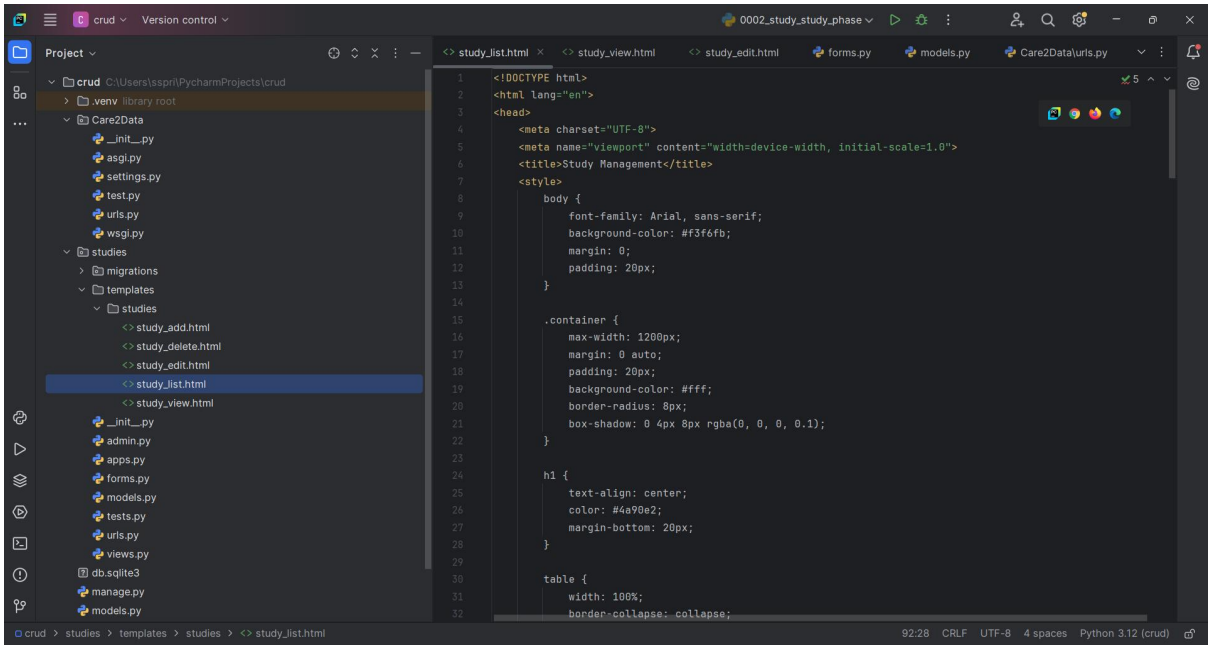
Test Duration: 6 hours

Main Page :



Priyanandhana S
2313712080018

IDE: PyCharm



Code:

Settings.py

```
"""
Django settings for Care2Data project.

Generated by 'django-admin startproject' using Django 5.1.3.

For more information on this file, see
https://docs.djangoproject.com/en/5.1/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/5.1/ref/settings/
"""

from pathlib import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/5.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-e@j+uw_^@1nd5v39xck4!2pv)ef)n7msye$&#3e(t0enn0xiat'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []
```

Priyanandhana S
2313712080018

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'studies',
]

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 = 'Care2Data.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [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 = 'Care2Data.wsgi.application'


# Database
# https://docs.djangoproject.com/en/5.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',  # Use MySQL backend
        'NAME': 'crud',  # Replace with your database name
        'USER': 'root',  # Replace with your MySQL username (default is 'root')
        'PASSWORD': 'root',  # Replace with your MySQL password
        'HOST': 'localhost',  # Or use your MySQL server hostname
        'PORT': '3306',  # Default MySQL port
    }
}
```

```
# Password validation
# https://docs.djangoproject.com/en/5.1/ref/settings/#auth-password-validators

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',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/5.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/5.1/howto/static-files/

STATIC_URL = 'static/'

# Default primary key field type
# https://docs.djangoproject.com/en/5.1/ref/settings/#default-auto-field

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

urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('studies.urls')),  # Include studies app URLs
]
```

Priyanandhana S
2313712080018

studies/templates/studies/study_list.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Study Management</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f3f6fb;
            margin: 0;
            padding: 20px;
        }

        .container {
            max-width: 1200px;
            margin: 0 auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
        }

        h1 {
            text-align: center;
            color: #4a90e2;
            margin-bottom: 20px;
        }

        table {
            width: 100%;
            border-collapse: collapse;
            margin-bottom: 20px;
        }

        table thead {
            background-color: #4a90e2;
            color: #fff;
        }

        table th, table td {
            text-align: left;
            padding: 12px;
            border: 1px solid #ddd;
        }

        table tr:nth-child(even) {
            background-color: #f9f9f9;
        }

        table tr:hover {
            background-color: #f1f1f1;
        }

        .actions a {
            text-decoration: none;
            padding: 8px 12px;
```

```
    border-radius: 4px;
    font-size: 14px;
}

.actions .view-btn {
    background-color: #4caf50;
    color: white;
}

.actions .edit-btn {
    background-color: #ff9800;
    color: white;
}

.buttons {
    display: flex;
    justify-content: center; /* Centers the buttons horizontally */
    gap: 20px;
    margin-top: 20px;
}

.buttons a, .buttons button {
    text-decoration: none;
    background-color: #4a90e2; /* Blue background for Add Study */
    color: white;
    padding: 12px 20px;
    border-radius: 4px;
    font-size: 16px;
    border: none;
    cursor: pointer;
    width: 200px; /* Same width for both buttons */
    text-align: center; /* Ensures text is centered */
}

.buttons a:hover, .buttons button:hover {
    background-color: #357abd; /* Darker blue for hover effect */
}

.buttons button {
    background-color: #ff3d3d; /* Red for delete button */
}

.buttons button:hover {
    background-color: #e53935;
}

/* Style for the "Add Study" button */
.buttons .add-btn {
    background-color: #4a90e2; /* Blue for Add Study */
}

.buttons .add-btn:hover {
    background-color: #357abd; /* Darker blue on hover */
}

/* Style for the "Delete Selected" button */
.buttons .delete-btn {
    background-color: #ff3d3d; /* Red for delete */
}
```

```
        .buttons .delete-btn:hover {
            background-color: #e53935; /* Darker red on hover */
        }
    </style>
</head>
<body>

    <div class="container">
        <h1>Study Management by Priyanandhana</h1>

        <!-- Form for bulk deletion -->
        <form method="POST" action="{% url 'study_delete_bulk' %}">
            {% csrf_token %}

            <table class="table">
                <thead>
                    <tr>
                        <th><input type="checkbox" id="select_all" onclick="selectAll()"></th>
                        <th>Study Name</th>
                        <th>Study Phase</th>
                        <th>Sponsor Name</th>
                        <th>Study Description</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    {% for study in studies %}
                        <tr>
                            <td><input type="checkbox" name="study_ids" value="{{ study.id }}"></td>
                            <td>{{ study.study_name }}</td>
                            <td>{{ study.study_phase }}</td>
                            <td>{{ study.sponsor_name }}</td>
                            <td>{{ study.study_description }}</td>
                            <td>
                                <a href="{% url 'study_view' study.id %}" class="btn btn-success">View</a>
                                <a href="{% url 'study_edit' study.id %}" class="btn btn-warning">Edit</a>
                            </td>
                        </tr>
                    {% empty %}
                        <tr>
                            <td colspan="6">No studies available</td>
                        </tr>
                    {% endfor %}
                </tbody>
            </table>

            <!-- Buttons to submit the selected studies for deletion -->
            <div class="buttons">
                <a href="{% url 'study_add' %}" class="add-btn">Add Study</a>
                <button type="submit" class="delete-btn">Delete Selected</button>
            </div>
        </form>
    </div>

    <script>
        // JavaScript for "Select All" functionality
        function selectAll() {
            const checkboxes = document.querySelectorAll('input[name="study_ids"]');
            checkboxes.forEach(function(checkbox) {
                checkbox.checked = document.getElementById('select_all').checked;
```

```
        });
    }
  </script>
</body>
</html>
```

studies/templates/studies/study_add.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Add Study</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f3f6fb;
      margin: 0;
      padding: 20px;
    }

    .container {
      max-width: 600px;
      margin: 0 auto;
      padding: 20px;
      background-color: #fff;
      border-radius: 8px;
      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    }

    h1 {
      text-align: center;
      color: #4a90e2;
      margin-bottom: 20px;
    }

    form {
      display: flex;
      flex-direction: column;
    }

    label {
      margin: 10px 0 5px;
      font-weight: bold;
    }

    input[type="text"], input[type="textarea"], select {
      padding: 10px;
      margin-bottom: 20px;
      border: 1px solid #ddd;
      border-radius: 4px;
    }

    button {
      background-color: #4a90e2;
      color: white;
      padding: 12px 20px;
```

```
            border: none;
            border-radius: 4px;
            font-size: 16px;
            cursor: pointer;
        }

        button:hover {
            background-color: #357abd;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Add Study</h1>
        <form method="POST">
            {% csrf_token %}
            {{ form.as_p }}  <!-- Render the form fields -->
            <button type="submit">Add Study</button>
        </form>
    </div>
</body>
</html>
```

studies/templates/studies/study_edit.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit Study</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f3f6fb;
            margin: 0;
            padding: 20px;
        }
        .container {
            max-width: 900px;
            margin: 0 auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
        }
        h1 {
            text-align: center;
            color: #4a90e2;
            margin-bottom: 20px;
        }
        label {
            font-size: 18px;
            color: #333;
            margin-bottom: 5px;
            display: inline-block;
        }
        input[type="text"], textarea, select {
```

```
            width: 100%;
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ddd;
            border-radius: 4px;
            font-size: 16px;
        }
        textarea {
            resize: vertical;
            height: 120px;
        }
        button {
            background-color: #4a90e2;
            color: white;
            padding: 12px 20px;
            border-radius: 4px;
            font-size: 16px;
            border: none;
            cursor: pointer;
            margin-right: 10px;
        }
        button:hover {
            background-color: #357abd;
        }
        .buttons {
            display: flex;
            justify-content: center;
            gap: 15px;
        }
        a {
            text-decoration: none;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Edit Study</h1>
        <form method="post" action="{% url 'study_edit' study.id %}">
            {% csrf_token %}

            <!-- Render the form fields using {{ form.as_p }} -->
            {{ form.as_p }}

            <div class="buttons">
                <button type="submit">Update</button>
                <a href="{% url 'study_list' %}"><button type="button">Cancel</button></a>
            </div>
        </form>
    </div>
</body>
</html>
```

studies/templates/studies/study_view.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

Priyanandhana S
2313712080018

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>View Study</title>
<style>
    body {
        font-family: Arial, sans-serif;
        background-color: #f3f6fb;
        margin: 0;
        padding: 20px;
    }

    .container {
        max-width: 900px;
        margin: 0 auto;
        padding: 20px;
        background-color: #fff;
        border-radius: 8px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    }

    h1 {
        text-align: center;
        color: #4a90e2;
        margin-bottom: 20px;
    }

    p {
        font-size: 18px;
        margin-bottom: 10px;
        color: #333;
    }

    strong {
        color: #4a90e2;
    }

    .buttons {
        display: flex;
        justify-content: center;
        margin-top: 20px;
    }

    .buttons a {
        text-decoration: none;
        background-color: #4a90e2;
        color: white;
        padding: 12px 20px;
        border-radius: 4px;
        font-size: 16px;
        border: none;
        cursor: pointer;
    }

    .buttons a:hover {
        background-color: #357abd;
    }

    /* Ensure paragraph content does not get cut off */
    .study-details p {
        word-wrap: break-word;
    }
```

```
    </style>
</head>
<body>

    <div class="container">
        <h1>View Study</h1>

        <div class="study-details">
            <p><strong>Study Name:</strong> {{ study.study_name }}</p>
            <p><strong>Study Description:</strong> {{ study.study_description }}</p>
            <p><strong>Study Phase:</strong> {{ study.study_phase }}</p>
            <p><strong>Sponsor Name:</strong> {{ study.sponsor_name }}</p>
        </div>

        <div class="buttons">
            <a href="{% url 'study_list' %}"><button type="button">Cancel</button></a>
        </div>
    </div>

</body>
</html>
```

studies/templates/studies/study_delete.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Delete Study</title>
</head>
<body>
    <h1>Delete Study</h1>
    <p>Are you sure you want to delete the study "{{ study.study_name }}"?</p>
    <form method="post" action="{% url 'study_delete' study.id %}">
        {% csrf_token %}
        <button type="submit">Yes, Delete</button>
    </form>
    <a href="{% url 'study_list' %}"><button type="button">Cancel</button></a>
</body>
</html>
```

apps.py

```
from django.apps import AppConfig

class StudiesConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'studies'
```

Priyanandhana S
2313712080018

forms.py

```
from django import forms
from .models import Study

class StudyForm(forms.ModelForm):
    class Meta:
        model = Study
        fields = ['study_name', 'study_phase', 'sponsor_name', 'study_description']
    phase = forms.ChoiceField(choices=Study.PHASE_CHOICES)
```

models.py

```
from django.db import models

class Study(models.Model):
    objects = None
    PHASE_CHOICES = [
        ('Phase I', 'Phase I'),
        ('Phase II', 'Phase II'),
        ('Phase III', 'Phase III'),
        ('Phase IV', 'Phase IV'),
    ]

    study_name = models.CharField(max_length=100, db_column='study_name')
    study_description = models.TextField(db_column='study_description')
    study_phase = models.CharField(max_length=50, choices=PHASE_CHOICES, db_column='study_phase')
    sponsor_name = models.CharField(max_length=100, db_column='sponsor_name')

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

studies/urls.py

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

urlpatterns = [
    path('', views.study_list, name='study_list'),
    path('add/', views.study_add, name='study_add'),
    path('view/<int:id>/', views.study_view, name='study_view'),
    path('edit/<int:id>/', views.study_edit, name='study_edit'),
    path('delete/<int:study_id>/', views.study_delete, name='study_delete'),
    path('delete_bulk/', views.study_delete_bulk, name='study_delete_bulk'),
]
```

Priyanandhana S
2313712080018

studies/views.py

```
from django.http import HttpResponse
from django.shortcuts import render, get_object_or_404, redirect
from .models import Study
from .forms import StudyForm

# View for displaying the list of studies
def study_list(request):
    studies = Study.objects.all()  # Get all studies
    return render(request, 'studies/study_list.html', {'studies': studies})


# View for adding a new study
def study_add(request):
    if request.method == 'POST':
        form = StudyForm(request.POST)
        if form.is_valid():
            form.save()  # Save the new study record
            return redirect('study_list')  # Redirect to the study list after adding
        else:
            print(form.errors)  # Log errors for debugging
    else:
        form = StudyForm()  # Show an empty form for a GET request
    return render(request, 'studies/study_add.html', {'form': form})


# View for viewing a study's details
def study_view(request, id):
    study = get_object_or_404(Study, id=id)  # Get the study object or return 404
    return render(request, 'studies/study_view.html', {'study': study})


# View for editing an existing study
def study_edit(request, id):
    # Get the study object based on the id
    study = get_object_or_404(Study, pk=id)

    if request.method == 'POST':
        form = StudyForm(request.POST, instance=study)  # Pre-populate the form with the study instance
        if form.is_valid():
            form.save()  # Save the updated study object
            return redirect('study_list')  # Redirect to the study list page
    else:
        form = StudyForm(instance=study)  # Populate form with current study data

    return render(request, 'studies/study_edit.html', {'form': form, 'study': study})



# View for deleting a study
def study_delete(request, study_id):
    study = get_object_or_404(Study, id=study_id)

    if request.method == 'POST':
        study.delete()  # Delete the study
        return redirect('study_list')  # Redirect to study list after deletion

    return render(request, 'studies/study_delete.html', {'study': study})
```

```
from django.shortcuts import render, redirect
from .models import Study

def study_delete_bulk(request):
    if request.method == 'POST':
        study_ids = request.POST.getlist('study_ids')
        if study_ids:
            Study.objects.filter(id__in=study_ids).delete()
        return redirect('study_list')  # Redirect to a page after deletion
```

models.py

```
# This is an auto-generated Django model module.
# You'll have to do the following manually to clean this up:
#   * Rearrange models' order
#   * Make sure each model has one field with primary_key=True
#   * Make sure each ForeignKey and OneToOneField has `on_delete` set to the desired behavior
#   * Remove `managed = False` lines if you wish to allow Django to create, modify, and delete the table
# Feel free to rename the models, but don't rename db_table values or field names.
from django.db import models


class AuthGroup(models.Model):
    name = models.CharField(unique=True, max_length=150)

    class Meta:
        managed = False
        db_table = 'auth_group'


class AuthGroupPermissions(models.Model):
    id = models.BigAutoField(primary_key=True)
    group = models.ForeignKey(AuthGroup, models.DO_NOTHING)
    permission = models.ForeignKey('AuthPermission', models.DO_NOTHING)

    class Meta:
        managed = False
        db_table = 'auth_group_permissions'
        unique_together = (('group', 'permission'),)


class AuthPermission(models.Model):
    name = models.CharField(max_length=255)
    content_type = models.ForeignKey('DjangoContentType', models.DO_NOTHING)
    codename = models.CharField(max_length=100)

    class Meta:
        managed = False
        db_table = 'auth_permission'
        unique_together = (('content_type', 'codename'),)


class AuthUser(models.Model):
    password = models.CharField(max_length=128)
```

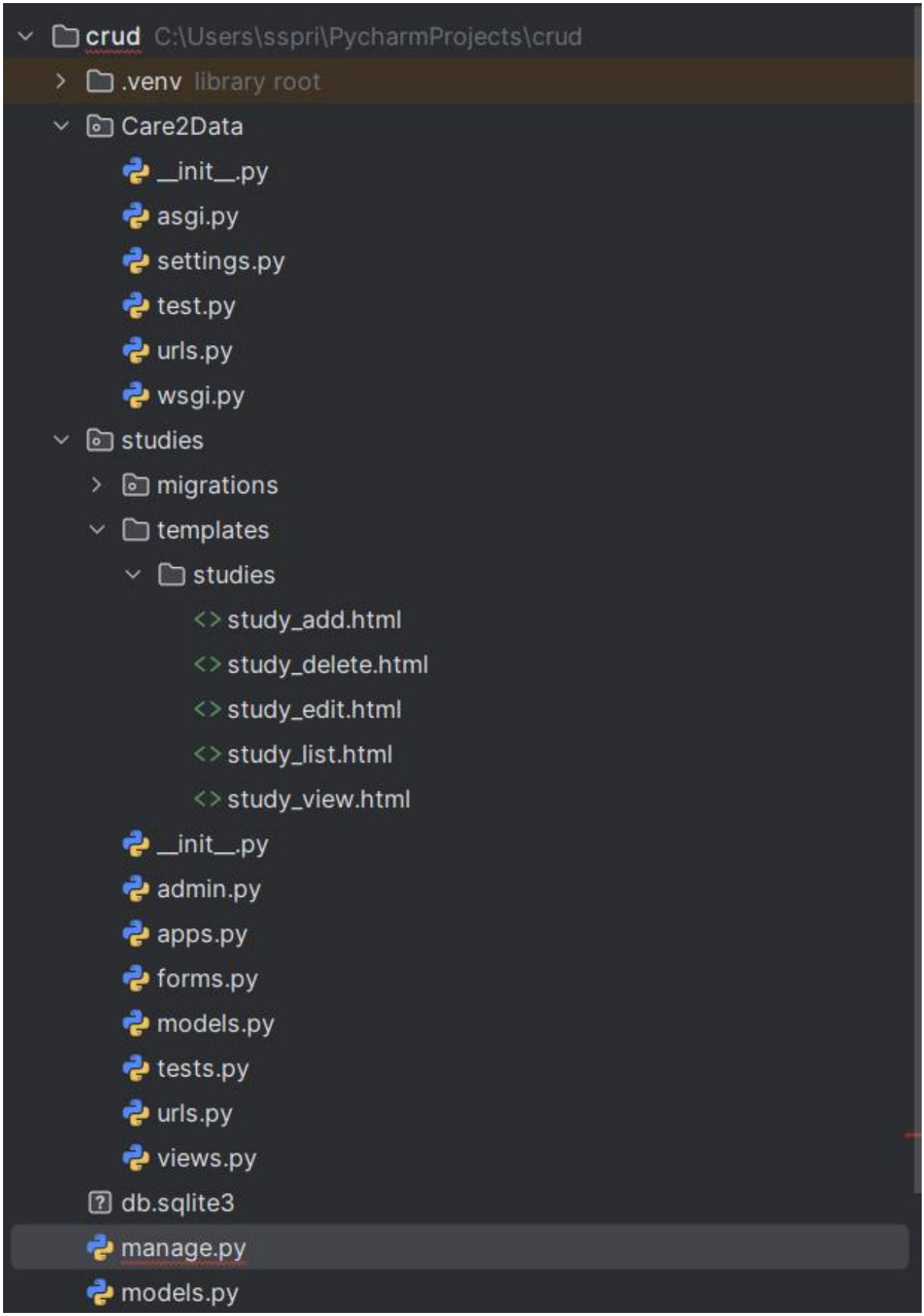
```
    last_login = models.DateTimeField(blank=True, null=True)
    is_superuser = models.IntegerField()
    username = models.CharField(unique=True, max_length=150)
    first_name = models.CharField(max_length=150)
    last_name = models.CharField(max_length=150)
    email = models.CharField(max_length=254)
    is_staff = models.IntegerField()
    is_active = models.IntegerField()
    date_joined = models.DateTimeField()

    class Meta:
        managed = False
        db_table = 'auth_user'


class AuthUserGroups(models.Model):
    id = models.BigAutoField(primary_key=True)
    user = models.ForeignKey(AuthUser, models.DO_NOTHING)
    group = models.ForeignKey(AuthGroup, models.DO_NOTHING)

    class Meta:
        managed = False
        db_table = 'auth_user_groups'
        unique_together = (('user', 'group'),)


class AuthUserUserPermissions(models.Model):
    id = models.BigAutoField(primary_key=True)
    user = models.ForeignKey(AuthUser, models.DO_NOTHING)
    permission = models.ForeignKey(AuthPermission, models.DO_NOTHING)

    class Meta:
        managed = False
        db_table = 'auth_user_user_permissions'
        unique_together = (('user', 'permission'),)


class DjangoAdminLog(models.Model):
    action_time = models.DateTimeField()
    object_id = models.TextField(blank=True, null=True)
    object_repr = models.CharField(max_length=200)
    action_flag = models.PositiveSmallIntegerField()
    change_message = models.TextField()
    content_type = models.ForeignKey('DjangoContentType', models.DO_NOTHING, blank=True, null=True)
    user = models.ForeignKey(AuthUser, models.DO_NOTHING)

    class Meta:
        managed = False
        db_table = 'django_admin_log'


class DjangoContentType(models.Model):
    app_label = models.CharField(max_length=100)
    model = models.CharField(max_length=100)

    class Meta:
        managed = False
        db_table = 'django_content_type'
        unique_together = (('app_label', 'model'),)
```

Priyanandhana S
2313712080018

```
class DjangoMigrations(models.Model):
    id = models.BigAutoField(primary_key=True)
    app = models.CharField(max_length=255)
    name = models.CharField(max_length=255)
    applied = models.DateTimeField()

    class Meta:
        managed = False
        db_table = 'django_migrations'


class DjangoSession(models.Model):
    session_key = models.CharField(primary_key=True, max_length=40)
    session_data = models.TextField()
    expire_date = models.DateTimeField()

    class Meta:
        managed = False
        db_table = 'django_session'


class StudiesStudy(models.Model):
    id = models.BigAutoField(primary_key=True)
    study_name = models.CharField(max_length=100)
    study_description = models.TextField()
    sponsor_name = models.CharField(max_length=100)
    study_phase = models.CharField(max_length=50, blank=True, null=True)

    class Meta:
        managed = False
        db_table = 'studies_study'
```

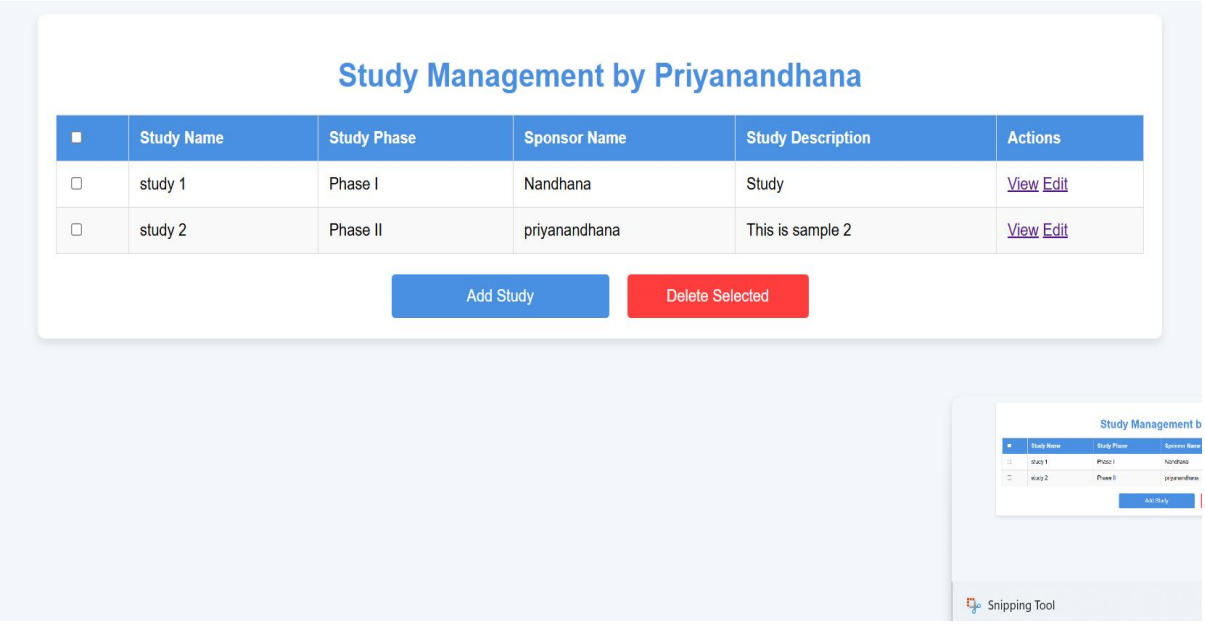
Priyanandhana S
2313712080018

My folders:

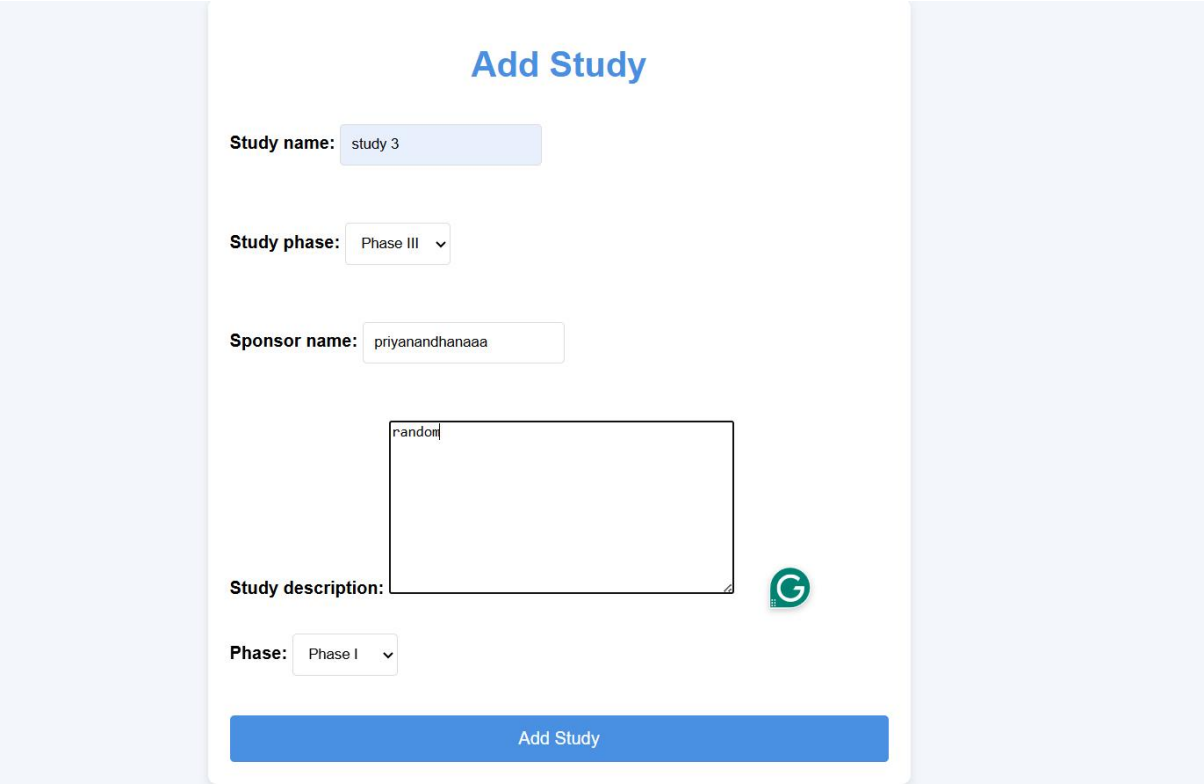


Priyanandhana S
2313712080018

Output :

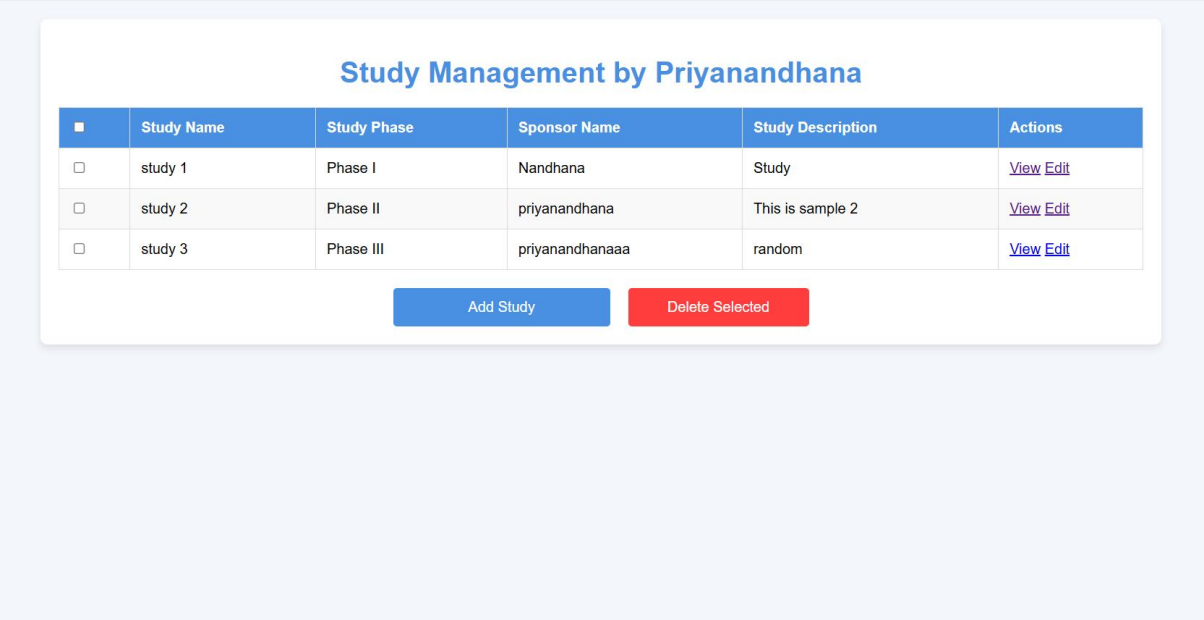


MAIN PAGE

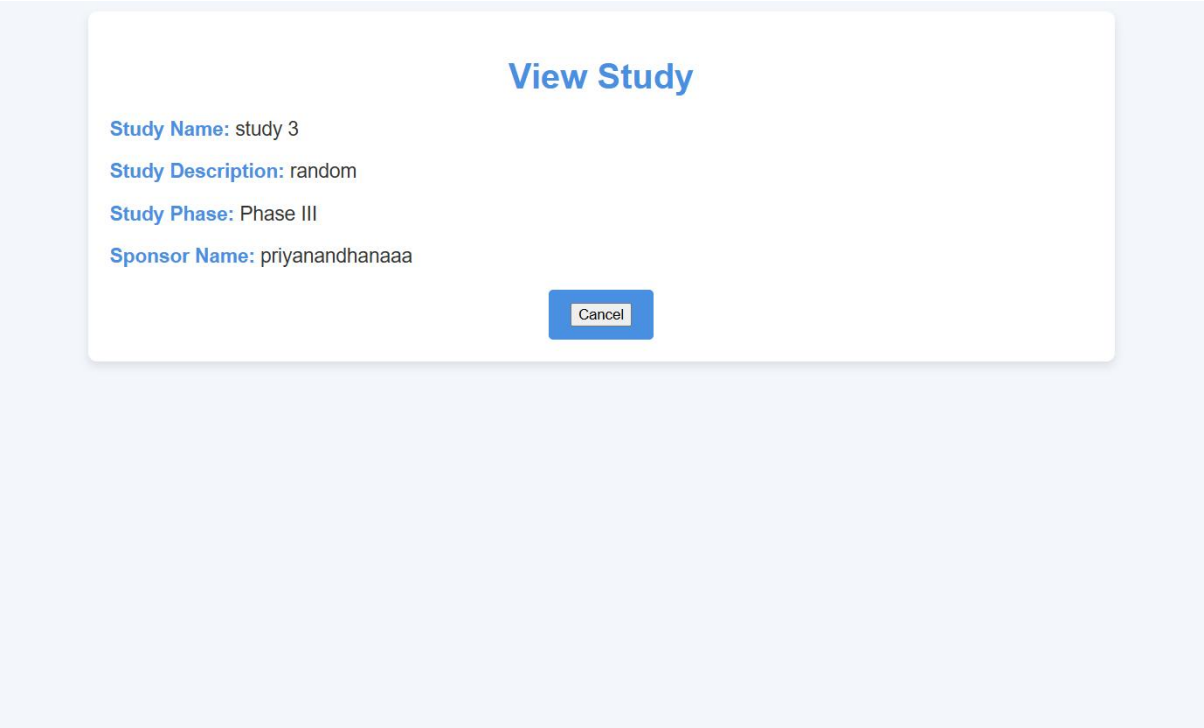


ADD STUDY

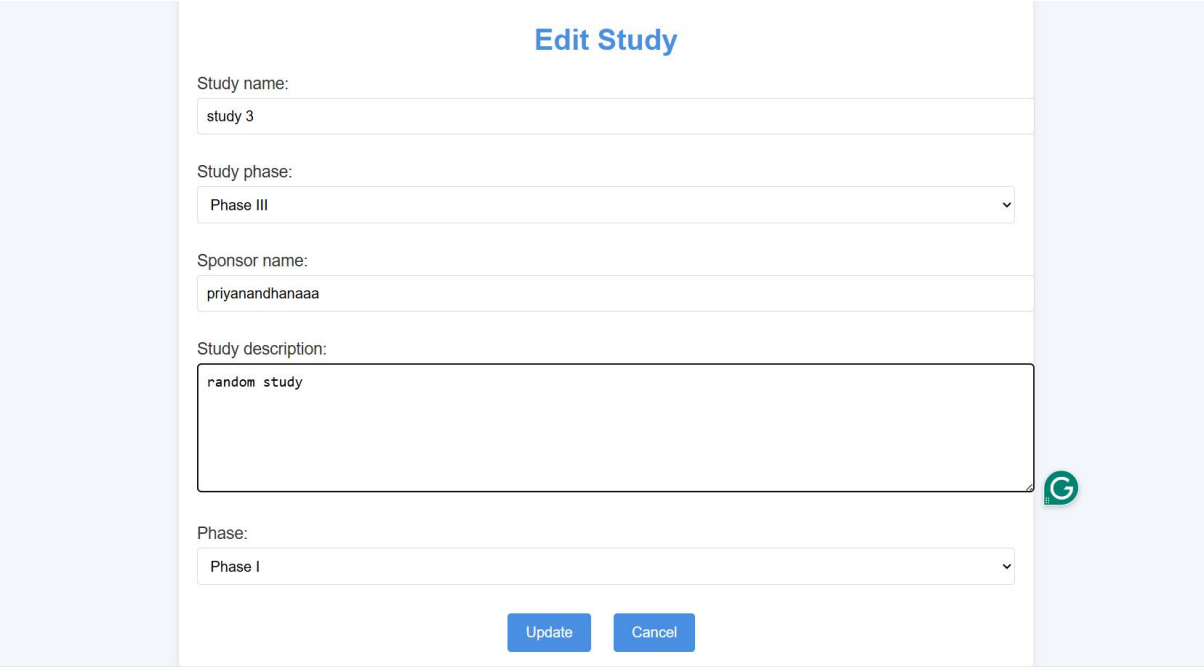
Priyanandhana S
2313712080018



ADDED



VIEW STUDY



EDIT STUDY

Conclusion:

The **Study Management System** is developed as user-friendly web application to streamline the management of studies, providing features for adding, editing, viewing, and deleting records. It's interface, responsive design, and robust functionality ensure efficient handling of study data, making it an effective tool for administrators. This project demonstrates the integration of simplicity and efficiency, offering a reliable solution for study management.

The project was started with the setup of a **Django framework**, where models were created to define the database structure for study data. Views were implemented to handle CRUD operations, and templates were designed for displaying and interacting with data.

CSS was applied to ensure an appealing and user-friendly interface. The system was thoroughly tested and debugged, resulting in a functional and efficient Study Management System. The Python exercise is successfully completed.