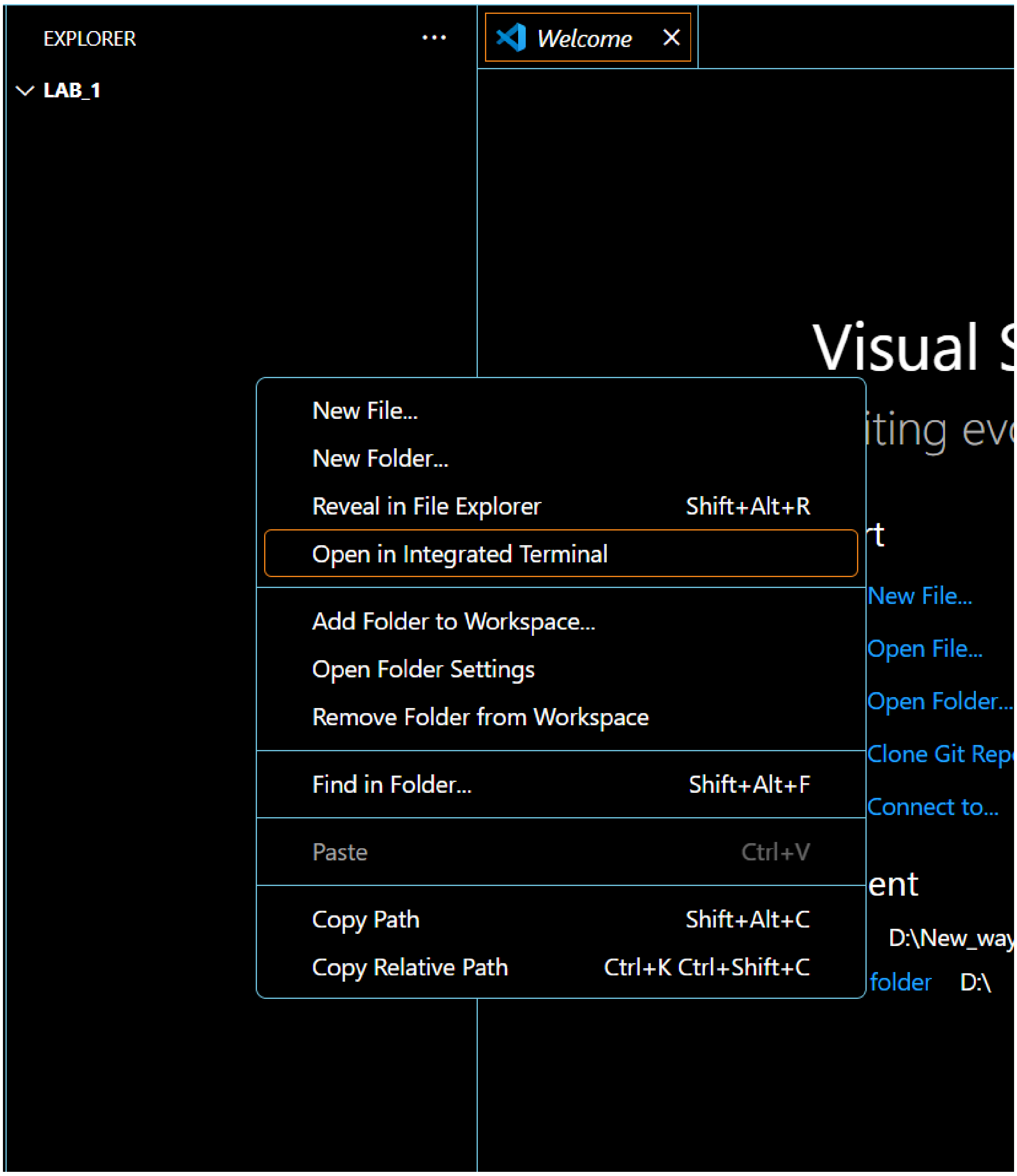
<u>Initial setup guide</u>

Step 1: Create Folder and open in vscode

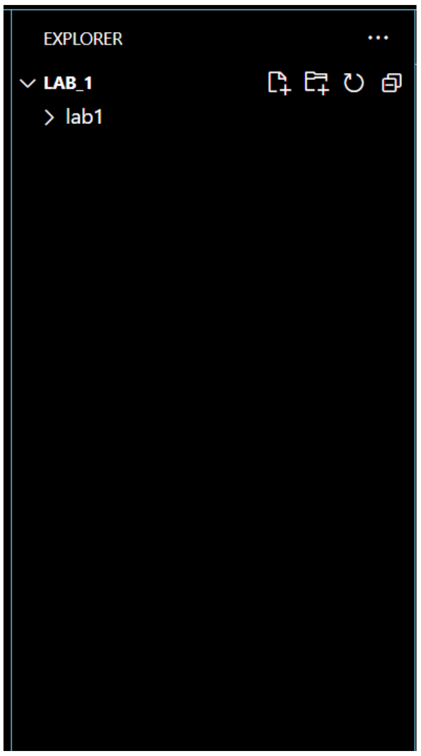
Step 2: Open Terminal



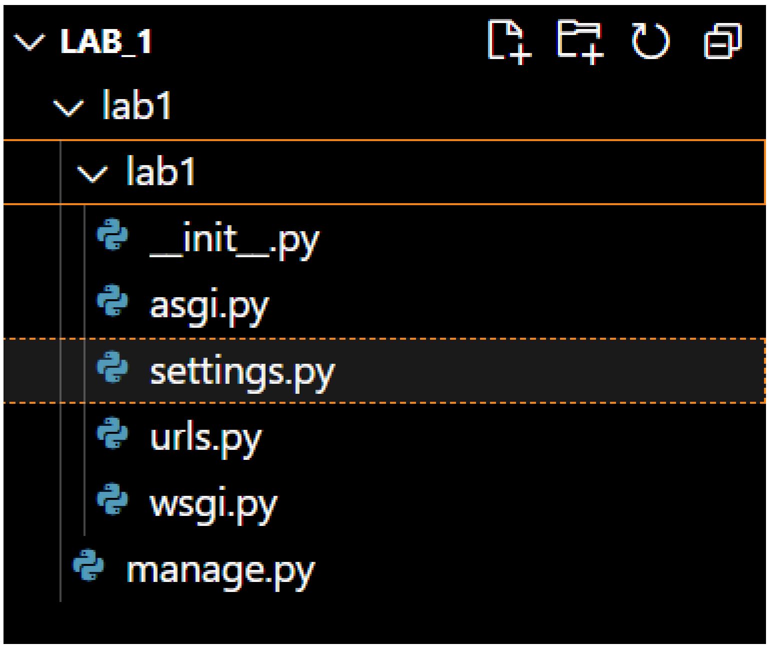
Step 3: For Creating Project use: **django-admin startproject <u>lab1</u>**

(Note Here lab1 is project name you can give any name)

You will get your project like below **lab1**



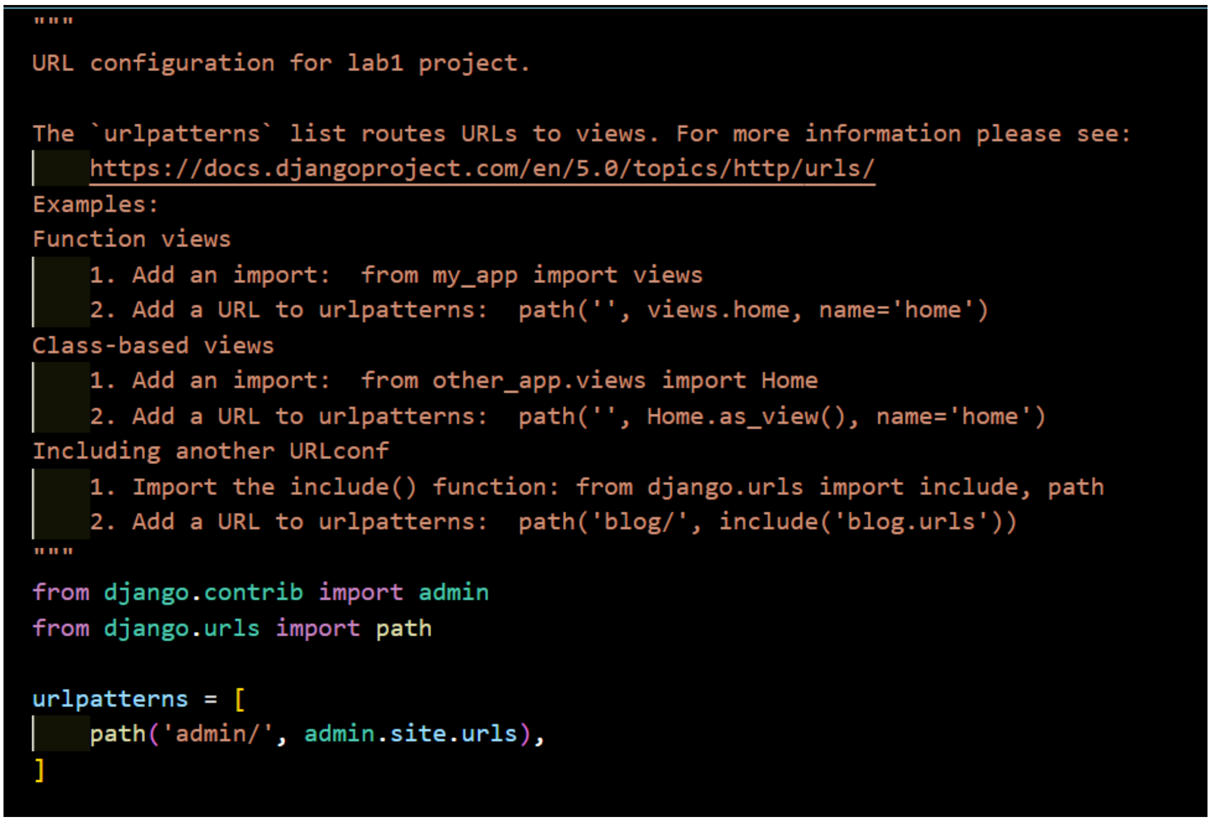
Step 4: Click that folder lab1. Again you click folder lab1



Step 5: Create python file name it **views.py**

Step 6: Click on urls.py

Step 7: Initial Look



```
"""
URL configuration for lab1 project.

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/5.0/topics/http/urls/
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

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

a) Import views for that type: **from . import views**

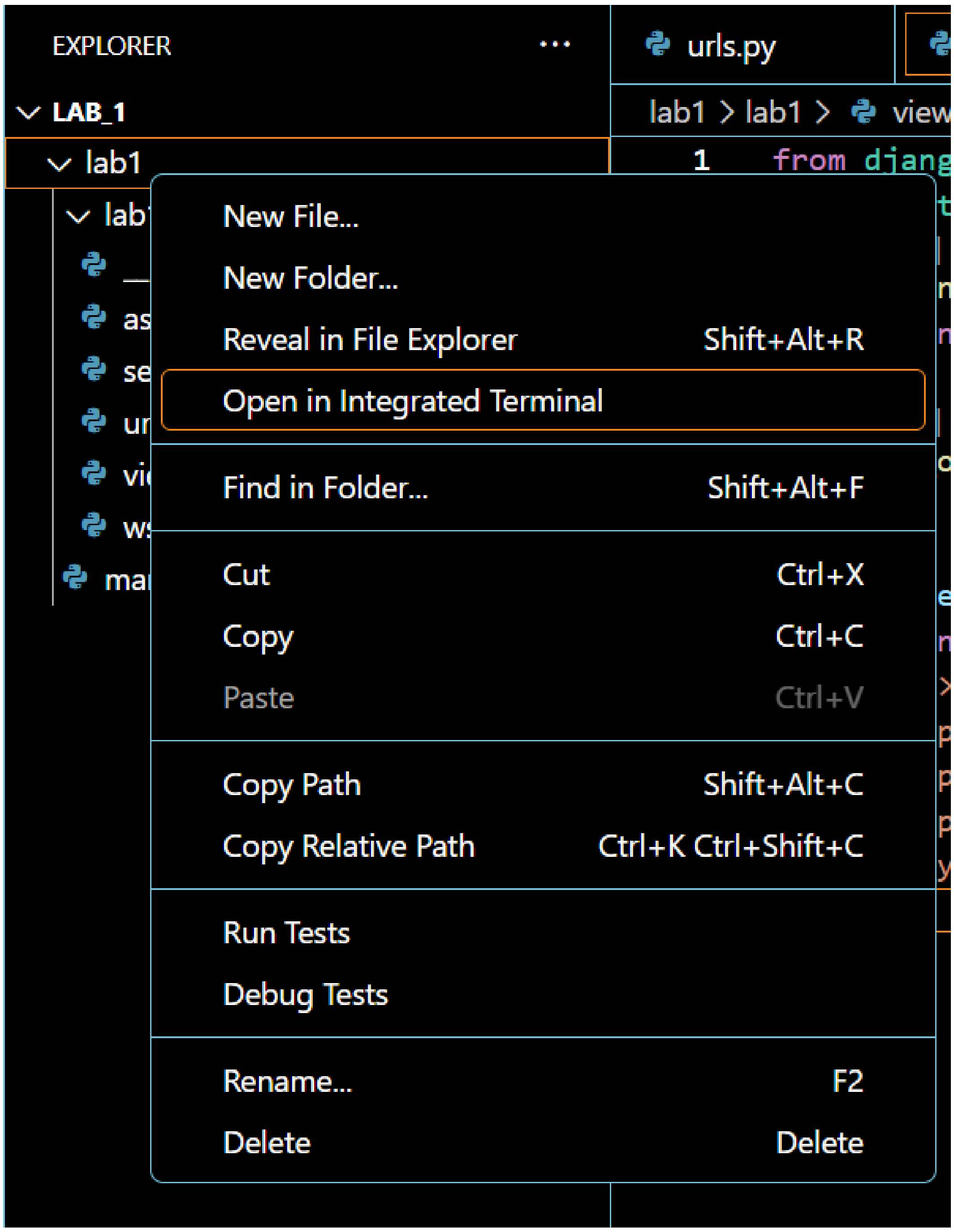
b) Update **urlpatterns**

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('time/', views.current_datetime),
    path('time-offsets/', views.time_offsets),
]
```

(Note:- Here **time/** should anything this is your path for url and **views.current_datetime**

In this **views** which you have created and .current_datetime or any after the . is the function name which you have created in views)

Step 8: Again open Integrated Terminal this time you right click in the folder which you have created in step 3 (lab1)



Step 9: Run the server use below to run:-

python manage.py runserver

Module 1

Question 1: **Develop a Django app that displays current date and time in server**

Step 1: Create Project :- **django-admin startproject Mod1_P1**

Step 2: Goto Your Project Directory:- **cd Mod1_P1**

Step 3: create views.py (Mod1_P1\Mod1_P1\views.py)

```
from django.http import HttpResponse
from datetime import datetime, timedelta
def current_datetime(request):
    return HttpResponse({datetime.now()})
```

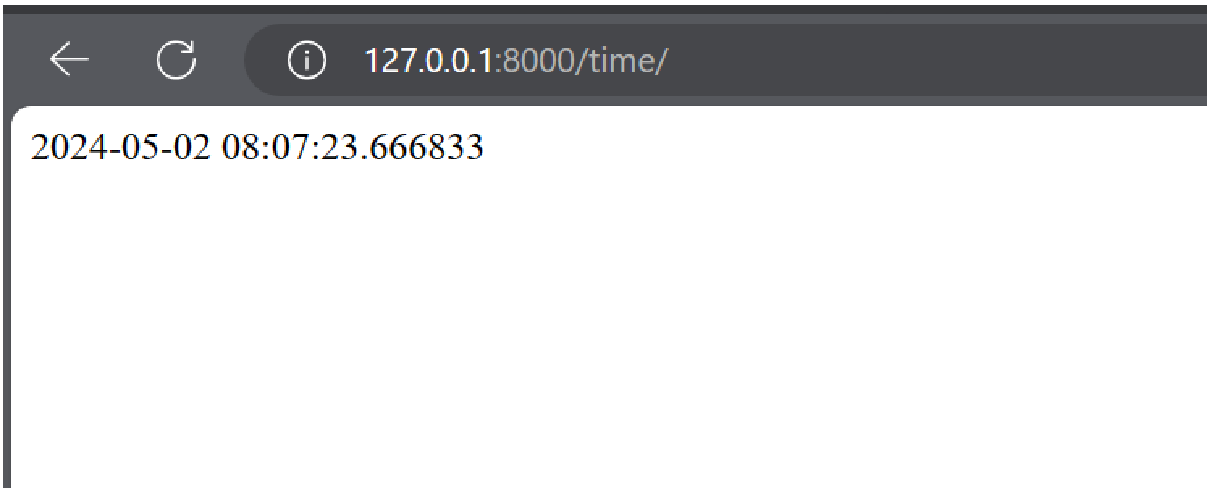
Step 4: Mod1_P1\Mod1_P1\urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('time/', views.current_datetime),
]
```

Output:-

Type URL : 127.0.0.1:8000/time/



127.0.0.1:8000/time/

```
2024-05-02 08:07:23.666833
```

Question 2: **Develop a Django app that displays date and time four hours ahead and four hours before as an offset of current date and time in server.**

Program:

Step 1: Create Project :- **django-admin startproject Mod1_P2**

Step 2: Goto Your Project Directory:- **cd Mod1_P2**

Step 3: create views.py (Mod1_P2\Mod1_P2\views.py)

```
from django.http import HttpResponse
from datetime import datetime, timedelta

def time_offsets(request):
    now = datetime.now()
    ahead = now + timedelta(hours=4)
    before = now - timedelta(hours=4)
    return HttpResponse(f"""
    <html><body>
        <p>Current Time: {now.strftime('%Y-%m-%d %H:%M:%S')}</p>
        <p>Time Four Hours Ahead: {ahead.strftime('%Y-%m-%d %H:%M:%S')}</p>
        <p>Time Four Hours Before: {before.strftime('%Y-%m-%d %H:%M:%S')}</p>
    </body></html>
    """)
```

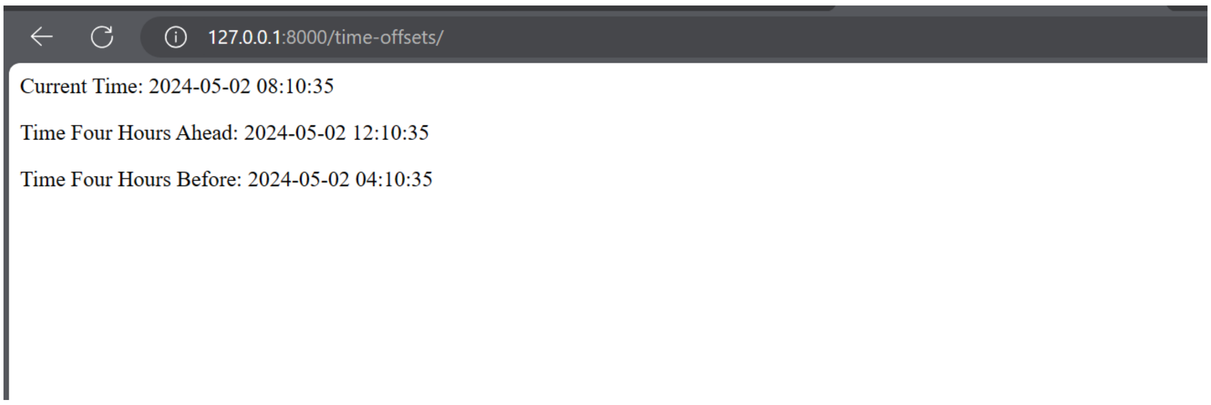
Step 4: Mod1_P2\Mod1_P2\urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('time-offsets/', views.time_offsets),
]
```

Output:

Type URL : 127.0.0.1:8000/time-offsets/


127.0.0.1:8000/time-offsets/

```
Current Time: 2024-05-02 08:10:35

Time Four Hours Ahead: 2024-05-02 12:10:35

Time Four Hours Before: 2024-05-02 04:10:35
```

Module 2

<u>Question 1:</u> **Develop a simple Django app that displays an unordered list of fruits and ordered list of selected students for an event.**

<u>Follow:</u>

Step 1: Create Project :- **django-admin startproject Mod2_P1**

Step 2: Goto Your Project Directory:- **cd Mod2_P1**

Step 3: Create app: **python manage.py startapp listing**

Step 4: Define Models In the listing app, edit the models.py

```
from django.db import models

class Fruit(models.Model):
    name = models.CharField(max_length=100)

    def __str__(self):
        return self.name

class Student(models.Model):
    name = models.CharField(max_length=100)
    selected_for_event = models.BooleanField(default=False)

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

Step 5: Register Models in Admin Modify the admin.py file in the listing

```
from django.contrib import admin
from .models import Fruit, Student
admin.site.register(Fruit)

def select_for_event(modeladmin, request, queryset):
    queryset.update(selected_for_event=True)
select_for_event.short_description = "Select students for the event"

def deselect_for_event(modeladmin, request, queryset):
    queryset.update(selected_for_event=False)
deselect_for_event.short_description = "Deselect students from the event"

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ('name', 'selected_for_event')
    list_filter = ('selected_for_event',)
    actions = [select_for_event, deselect_for_event]
```

Step 6: Configure URLs

a) settings.py under INSTALLED_APPS:

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'listing',  # Here register your app using this line
]
```

b) Configure the URLs in Mod2_P1/urls.py:

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

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

c) Create a urls.py in your listing app:

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

urlpatterns = [
    path('', display_lists, name='display_lists'),
]
```

Step 6: Create Views In the **views.py** file of the **listing** app

```
from django.shortcuts import render
from .models import Fruit, Student

def display_lists(request):
    fruits = Fruit.objects.all()
    selected_students =
Student.objects.filter(selected_for_event=True).order_by('name')
    return render(request, 'listing/display_lists.html', {'fruits': fruits,
'selected_students': selected_students})
```

Step 7: Create Templates in inside your listing app.

Create folder **listing** inside templates

display_lists.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>List of Fruits and Selected Students</title>
</head>
<body>
    <h1>Fruits List</h1>
    <ul>
        {% for fruit in fruits %}
        <li>{{ fruit.name }}</li>
        {% endfor %}
    </ul>

    <h1>Selected Students for Event</h1>
    <ol>
        {% for student in selected_students %}
        <li>{{ student.name }}</li>
        {% endfor %}
    </ol>
</body>
</html>
```

Step 7: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y
d) **Start the server**: python manage.py runserver

Step 8: Access Your Application

Admin Interface: Go to http://127.0.0.1:8000/admin/

View Lists: http://127.0.0.1:8000/

Question 2: **Develop a layout.html with a suitable header (containing navigation menu) and footer with copyright and developer information. Inherit this layout.html and create 3 additional pages: contact us, About Us and Home page of any website.**

Follow:

Step 1: Create Project :- **django-admin startproject Mod2_P2**

Step 2: Goto Your Project Directory: - **cd Mod2_P2**

Step 3: Create app: **python manage.py startapp myweb**

Step 4: Create a **templates** folder inside **myweb➔** inside **templates** create layout.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{% block title %}My Site{% endblock %}</title>
</head>
<body>
    <header>
        <h1>My Website</h1>
        <nav>
            <ul>
                <li><a href="{% url 'home' %}">Home</a></li>
                <li><a href="{% url 'about' %}">About Us</a></li>
                <li><a href="{% url 'contact' %}">Contact Us</a></li>
            </ul>
        </nav>
    </header>

    <main>
        {% block content %}
        {% endblock %}
    </main>

    <footer>
        <p>Copyright © 2024 by Rohan. All rights reserved.</p>
        <p>Developed by Rohan. </p>
    </footer>
</body>
</html>
```

Step 5: Create Child Pages

a) **home.html**

```
{% extends 'layout.html' %}

{% block title %} Home Page {% endblock %}


{% block content %}
<h2>Welcome to My World</h2>
{% endblock %}
```

b) **about.html**

```
{% extends 'layout.html' %}

{% block title %} About Us{% endblock %}

{% block content %}
<h2>Now lets know about me!</h2>

{% endblock %}
```

c) **contact.html**

```
{% extends 'layout.html' %}

{% block title %}Contact{% endblock %}

{% block content %}
<h2>Now lets call me!</h2>

{% endblock %}
```

Step 6: Define URLs

In **myweb/urls.py**, map URLs (create urls.py if not there)

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

urlpatterns = [
    path('', views.home, name='home'),
    path('about/', views.about, name='about'),
    path('contact/', views.contact, name='contact'),
]
```

Step 7: Create Views

In **myweb/views.py**

```
from django.shortcuts import render

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

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

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

Step 8: Configure Settings (go to project folder & open settings.py)

a)

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'templates'],  # Add this line
        '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',
            ],
        },
    },
]
```

b)

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'myweb',  # add your app.
]
```

Step 9: Map the URL In **Mod2_P2**/urls.py

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

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

Step 10: Start the server: python manage.py runserver

Question 3: Develop a Django app that performs student registration to a course. It should also display list of students registered for any selected course. Create students and course as models with enrolment as ManyToMany field.

Follow:

Step 1: Create Project :- **django-admin startproject Mod2_P3**

Step 2: Goto Your Project Directory: - **cd Mod2_P3**

Step 3: Create app: **python manage.py startapp registration**

Step 4: Create a **templates** folder inside that create **registration.html**

```
<!DOCTYPE html>
<html>
<body>
    <h1>Student Registration</h1>
  <form method="post">{% csrf_token %}
    <input type="hidden" name="action" value="register_student">
    USN: <input name="usn"><br>
    Name: <input name="name"><br>
    Department: <input name="department"><br>
    <button>Register Student</button>
  </form>
    <h2>Course Registration</h2>
  <form method="post">{% csrf_token %}
    <input type="hidden" name="action" value="register_course">
    Code: <input name="code"><br>
    Name: <input name="name"><br>
    <button>Register Course</button>
  </form>
    <h2>Enroll Student</h2>
  <form method="post">{% csrf_token %}
    <input type="hidden" name="action" value="enroll_student">
    <select name="student">{% for s in students %}
      <option value="{{ s.id }}">{{ s }}</option>{% endfor %}
    </select>
    <select name="course">{% for c in courses %}
      <option value="{{ c.id }}">{{ c }}</option>{% endfor %}
    </select>
    <button>Enroll</button>
  </form>
  <h2>Enrolled Students and Their Courses</h2>
  <table border='1'>
    <tr><th>Student</th><th>Courses</th></tr>
    {% for item in student_course_info %}
    <tr>
      <td>{{ item.student }}</td>
      <td>{{ item.courses }}</td>
    </tr>
    {% endfor %}
  </table>
</body>
</html>
```

Step 5: \Mod2_P3\registration\models.py

```
from django.db import models
class Student(models.Model):
    usn = models.CharField(max_length=15, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)
    def __str__(self):
        return f"{self.usn} - {self.name}"
class Course(models.Model):
    code = models.CharField(max_length=10, unique=True)
    name = models.CharField(max_length=100)
    students = models.ManyToManyField(Student, related_name='courses')
    def __str__(self):
        return f"{self.name} ({self.code})"
```

Step 6: \Mod2_P3\registration\views.py

```
from django.shortcuts import render, redirect
from .models import Student, Course
def register_and_enroll(request):
    if request.method == 'POST':
        action = request.POST.get('action')
        if action == 'register_student':
            Student.objects.create(
                usn=request.POST['usn'],
                name=request.POST['name'],
                department=request.POST['department']
            )
        elif action == 'register_course':
            Course.objects.create(
                code=request.POST['code'],
                name=request.POST['name']
            )
        elif action == 'enroll_student':
            student = Student.objects.get(id=request.POST['student'])
            course = Course.objects.get(id=request.POST['course'])
            course.students.add(student)
        return redirect('register_and_enroll')
    context = {
        'students': Student.objects.all(),
        'courses': Course.objects.all(),
        'student_course_info': [
            {'student': s, 'courses': ", ".join([str(c) for c in
s.courses.all()])}
            for s in Student.objects.prefetch_related('courses')
        ]
    }
    return render(request, 'registration.html', context)
```

Configuration

Step 7: \Mod2_P3 \registration\urls.py

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

urlpatterns = [
    path('', register_and_enroll, name='register_and_enroll'),
]
```

Step 8: \Mod2_P3 \college\urls.py

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

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

Step 9: \Mod2_P3 \college\settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'registration', # Add your app here
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:

   python manage.py makemigrations

   python manage.py migrate

b) python manage.py runserver

<u>**Module 3**</u>

<u>**Question 1:**</u> **For student and course models created in Lab experiment for Module2, register admin interfaces, perform migrations and illustrate data entry through admin forms**

<u>**Follow:**</u>

Step 1: Create Project :- **django-admin startproject Mod3_P2**

Step 2: Goto Your Project Directory: - **cd Mod3_P2**

Step 3: Create app: **python manage.py startapp registration**

Step 4: Create a **templates** folder inside that create **display.html**

```
<!DOCTYPE html>
<html>
<head>
    <title>Enrolled Students and Their Courses</title>
</head>
<body>
    <h1>Enrolled Students and Their Courses</h1>
    <table border="1">
        <tr>
            <th>Student</th>
            <th>Courses</th>
        </tr>
        {% for item in student_course_info %}
        <tr>
            <td>{{ item.student }}</td>
            <td>{{ item.courses }}</td>
        </tr>
        {% endfor %}
    </table>
</body>
</html>
```

Step 4: Mod3_P1\registration\models.py

```
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=15, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)

    def __str__(self):
        return f"{self.usn} - {self.name}"

class Course(models.Model):
    code = models.CharField(max_length=10, unique=True)
    name = models.CharField(max_length=100)
    students = models.ManyToManyField(Student, related_name='courses')

    def __str__(self):
        return f"{self.name} ({self.code})"
```

Step 5: Mod3_P1\registration\admin.py

```
from django.contrib import admin
from .models import Student, Course

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ['usn', 'name', 'department']

@admin.register(Course)
class CourseAdmin(admin.ModelAdmin):
    list_display = ['code', 'name']
    filter_horizontal = ['students']
```

Step 6: Mod3_P1\registration\views.py

```
from django.shortcuts import render
from .models import Student

def display_data(request):
    students = Student.objects.prefetch_related('courses')
    student_course_info = [
        {'student': s, 'courses': ", ".join([str(c) for c in
s.courses.all()])}
        for s in students
    ]

    return render(request, 'display.html', {
        'student_course_info': student_course_info
    })
```

Configure Settings

Step 7: Mod3_P1\settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'registration', # Add your app here
]
```

Step 8: Mod3_P1\ Mod3_P1\urls.py

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

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

Step 9: Mod3_P1\registration\urls.py

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

urlpatterns = [
    path('', display_data),
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver

Question 2: **Develop a Model form for student that contains his topic chosen for project, languages used and duration with a model called project.**

Follow:

Step 1: Create Project :- **django-admin startproject Mod3_P2**

Step 2: Goto Your Project Directory: - **cd Mod3_P1**

Step 3: Create app: **python manage.py startapp registration**

Step 4: Create a **templates** folder inside that create **project_form**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Project Registration</title>
</head>
<body>
    <h1>Register a Project</h1>
    <form method="post">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Submit</button>
    </form>

    <h2>Registered Projects</h2>
    <table border="1">
        <tr>
            <th>Student USN</th>
            <th>Student Name</th>
            <th>Project Topic</th>
            <th>Languages Used</th>
            <th>Duration (Months)</th>
        </tr>
        {% for project in projects %}
        <tr>
            <td>{{ project.student.usn }}</td>
            <td>{{ project.student.name }}</td>
            <td>{{ project.topic }}</td>
            <td>{{ project.languages }}</td>
            <td>{{ project.duration }}</td>
        </tr>
        {% endfor %}
    </table>
</body>
</html>
```

Step 5: Mod3_P2\registration\admin.py

```python
from django.contrib import admin
from .models import Student

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ['usn', 'name', 'department']
    search_fields = ['usn', 'name']
```

Step 6: Mod3_P2\registration\forms.py

```python
from django import forms
from .models import Project

class ProjectForm(forms.ModelForm):
    class Meta:
        model = Project
        fields = '__all__'
```

Step 7: Mod3_P2\registration\models.py

```python
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=10, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)

    def __str__(self):
        return f"{self.usn} - {self.name}"

class Project(models.Model):
    student = models.OneToOneField(Student, on_delete=models.CASCADE)
    topic = models.CharField(max_length=200)
    languages = models.CharField(max_length=200)
    duration = models.IntegerField(help_text="Duration in months")

    def __str__(self):
        return f"{self.topic} - {self.student.name}"
```

Step 8: Mod3_P2\registration\views.py

```
from django.shortcuts import render, redirect
from .forms import ProjectForm
from .models import Project

def project_form(request):
    if request.method == 'POST':
        form = ProjectForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('project_form')
    else:
        form = ProjectForm()

    projects = Project.objects.all()
    return render(request, 'project_form.html', {'form': form, 'projects':
projects})
```

Configure Settings

Step 9: Mod3_P2\settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'registration', # Add your app here
]
```

Step 10: Mod3_P2\ Mod3_P2\urls.py

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

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

Step 11: Mod3_P2\registration\urls.py

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

urlpatterns = [
    path('', project_form, name='project_form'),
]
```

Step 12: Run Migrations and Start the Server

a) **Run migrations**:
```
python manage.py makemigrations
python manage.py migrate
```
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver

Module-4

Question 1: For students enrolment developed in Module 2, create a generic class view which displays list of students and detailview that displays student details for any selected student in the list.

Follow:

Step 1: Create Project :- **django-admin startproject Mod4_P1**

Step 2: Goto Your Project Directory: - **cd Mod4_P1**

Step 3: Create app: **python manage.py startapp enrollment**

Step 4: Create a **templates** folder inside that create **html files:-**

a) student_detail.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Student Detail</title>
</head>
<body>
    <h1>Student Detail</h1>
    <p><strong>USN:</strong> {{ student.usn }}</p>
    <p><strong>Name:</strong> {{ student.name }}</p>
    <p><strong>Department:</strong> {{ student.department }}</p>
    <a href="{% url 'student_list' %}">Back to List</a>
</body>
</html>
```

b) student_list.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Student List</title>
</head>
<body>
    <h1>Student List</h1>
    <table border="1">
        <tr>
            <th>USN</th>
            <th>Name</th>
            <th>Action</th>
        </tr>
        {% for student in students %}
        <tr>
            <td>{{ student.usn }}</td>
            <td>{{ student.name }}</td>
            <td><a href="{% url 'student_detail' student.pk %}">View
Details</a></td>
        </tr>
        {% endfor %}
    </table>
</body>
</html>
```

Step 5: Mod4_P1\enrollment\admin.py

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

admin.site.register(Student)
```

Step 6: Mod4_P1\enrollment\models.py

```
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=15, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)

    def __str__(self):
        return f"{self.usn} - {self.name}"
```

Step 7: Mod4_P1\enrollment\views.py

```
from django.shortcuts import render
from django.views.generic import ListView, DetailView
from .models import Student

class StudentListView(ListView):
    model = Student
    template_name = 'student_list.html'
    context_object_name = 'students'

class StudentDetailView(DetailView):
    model = Student
    template_name = 'student_detail.html'
    context_object_name = 'student'
```

Configure Settings

Step 8: Mod4_P1\settings.py

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

Step 9: Mod4_P1\urls.py

```
from django.contrib import admin
from django.urls import path
from enrollment.views import *

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', StudentListView.as_view(), name='student_list'),
    path('student/<int:pk>/', StudentDetailView.as_view(),
name='student_detail'),
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver

<u>**Question 2:**</u> **Develop example Django app that performs CSV and PDF generation for any models created in previous laboratory component.**

<u>**Follow:**</u>

Step 1: Create Project :- **django-admin startproject Mod4_P2**

Step 2: Goto Your Project Directory: - **cd Mod4_P2**

Step 3: Create app: **python manage.py startapp students**

Step 4: Create a **templates** folder inside that create **student_list.html**

```
<!DOCTYPE html>
<html>
<body>
    <h1>Student List</h1>
    <table border="1">
        <tr>
            <th>USN</th>
            <th>Name</th>
            <th>Department</th>
        </tr>
        {% for student in students %}
        <tr>
            <td>{{ student.usn }}</td>
            <td>{{ student.name }}</td>
            <td>{{ student.department }}</td>
        </tr>
        {% endfor %}
    </table>
    <br>
    <a href="{% url 'generate_csv' %}"><button>Download CSV</button></a>
    <a href="{% url 'generate_pdf' %}"><button>Download PDF</button></a>
</body>
</html>
```

Step 5: Mod4_P2\students\models.py

```
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=15, primary_key=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)

    def __str__(self):
        return f"{self.usn} - {self.name}"
```

Step 6: Mod4_P2\students\views.py

```
import io
import pandas as pd
import matplotlib.pyplot as plt
from django.http import FileResponse, HttpResponse
from django.shortcuts import render
from .models import Student

def student_list(request):
    return render(request, 'student_list.html', {'students':
Student.objects.all()})

def generate_csv(request):
    df = pd.DataFrame(Student.objects.values())
    response = HttpResponse(df.to_csv(index=False), content_type='text/csv')
    response['Content-Disposition'] = 'attachment; filename="students.csv"'
    return response

def generate_pdf(request):
    df = pd.DataFrame(Student.objects.values())
    fig, ax = plt.subplots(figsize=(12, 2))
    ax.axis('off')
    ax.table(cellText=df.values, colLabels=df.columns, loc='center')
    buffer = io.BytesIO()
    plt.savefig(buffer, format='pdf')
    buffer.seek(0)
    return FileResponse(buffer, as_attachment=True, filename='students.pdf')
```

Step 7: Mod4_P2\students\admin.py

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

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ('usn', 'name', 'department')
```

Configure Settings

Step 8: Mod4_P2\settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'students', # add your app
]
```

Step 9: Mod4_P2\settings.py

```
from django.contrib import admin
from django.urls import path
from students import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.student_list, name='student_list'),
    path('generate-csv/', views.generate_csv, name='generate_csv'),
    path('generate-pdf/', views.generate_pdf, name='generate_pdf'),
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate

b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)

c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver

Module 5

Question 1: Develop a registration page for student enrolment as done in Module 2 but without page refresh using AJAX.

Follow:

Step 1: Create Project :- **django-admin startproject Mod5_P1**

Step 2: Goto Your Project Directory: - **cd Mod5_P1**

Step 3: Create app: **python manage.py startapp enrollment**

Step 4: Create a **templates** folder inside that create **enrollment.html**

```
<!DOCTYPE html>
<html>
<head>
    <title>Student Enrollment</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <h2>Enroll Student</h2>
    <form id="enrollForm">
        {% csrf_token %}
        <select name="student" id="student">
            {% for s in students %}
                <option value="{{ s.id }}">{{ s.name }}</option>
            {% endfor %}
        </select>
        <select name="course" id="course">
            {% for c in courses %}
                <option value="{{ c.id }}">{{ c.name }}</option>
            {% endfor %}
        </select>
        <button type="submit">Enroll</button>
    </form>
    <div id="message"></div>

    <script>
        $(document).ready(function() {
            $('#enrollForm').on('submit', function(e) {
                e.preventDefault();
                $.ajax({
                    type: 'POST',
                    url: '',
                    data: $(this).serialize(),
                    success: function(response) {
                        $('#message').text(response.message);
                    }
                });
            });
        });
    </script>
</body>
</html>
```

Step 5: Mod5_P1\enrollment\models.py

```python
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=15, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)
    enrollment = models.CharField(max_length=255, blank=True, default='')

    def __str__(self):
        return f"{self.usn} - {self.name}"

class Course(models.Model):
    code = models.CharField(max_length=10, unique=True)
    name = models.CharField(max_length=100)

    def __str__(self):
        return f"{self.name} ({self.code})"
```

Step 6: Mod5_P1\enrollment\admin.py

```python
from django.contrib import admin
from .models import Student, Course

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ['usn', 'name', 'department', 'enrollment']
    exclude = ['enrollment']

@admin.register(Course)
class CourseAdmin(admin.ModelAdmin):
    list_display = ['code', 'name']
```

Step 7: Mod5_P1\enrollment\views.py

```python
from django.shortcuts import *
from django.http import *
from .models import *
```

```
def enroll_student(request):
    if request.method == 'POST':
        student = get_object_or_404(Student, id=request.POST.get('student'))
        course = get_object_or_404(Course, id=request.POST.get('course'))
        student.enrollment = f"{student.enrollment}, {str(course)}" if
student.enrollment else str(course)
        student.save()
        return JsonResponse({'success': True, 'message': 'Enrollment
successful!'})

    return render(request, 'enrollment.html', {
        'students': Student.objects.all(),
        'courses': Course.objects.all()
    })
```

Configuration
Step 8: Mod5_P1\settings.py

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

Step 9: Mod5_P1\urls.py

```
from django.contrib import admin
from django.urls import path
from enrollment.views import enroll_student

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', enroll_student, name='enroll_student'),
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver

Question 2: Develop a search application in Django using AJAX that displays courses enrolled by a student being searched.

Follow:

Step 1: Create Project :- **django-admin startproject Mod5_P2**

Step 2: Goto Your Project Directory: - **cd Mod5_P2**

Step 3: Create app: **python manage.py startapp enrollment**

Step 4: Create a **templates** folder inside that create **enrollment.html**

```
<!DOCTYPE html>
<html>
<head>
    <title>Student Course Search</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <h2>Search Courses</h2>
    <input type="text" id="student_usn" placeholder="Enter USN">
    <button onclick="searchCourses()">Search</button>
    <div id="courseList"></div>

    <script>
    function searchCourses() {
        $.getJSON('/search/', {usn: $('#student_usn').val()}, function(data) {
            $('#courseList').html(data.courses.join(', ') || 'No courses 
found');
        });
    }
    </script>
</body>
</html>
```

Step 5: Mod5_P2\enrollment\admin.py

```
from django.contrib import admin
from .models import Student, Course

@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ['usn', 'name', 'department']
    filter_horizontal = ('enrollment',)

@admin.register(Course)
class CourseAdmin(admin.ModelAdmin):
    list_display = ['code', 'name']
```

Step 6: Mod5_P2\enrollment\models.py

```
from django.db import models

class Student(models.Model):
    usn = models.CharField(max_length=15, unique=True)
    name = models.CharField(max_length=100)
    department = models.CharField(max_length=50)
    enrollment = models.ManyToManyField('Course', blank=True)

    def __str__(self):
        return f"{self.usn} - {self.name}"

class Course(models.Model):
    code = models.CharField(max_length=10, unique=True)
    name = models.CharField(max_length=100)

    def __str__(self):
        return f"{self.name} ({self.code})"
```

Step 7: Mod5_P2\enrollment\views.py

```
from django.shortcuts import render
from django.http import JsonResponse
from .models import Student

def search_view(request):
    usn = request.GET.get('usn')
    student = Student.objects.filter(usn=usn).first()
    courses = [course.name for course in student.enrollment.all()] if student
else []
    return JsonResponse({'courses': courses})

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

Configuration

Step 8: Mod5_P2\settings.py

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

Step 9: Mod5_P2\ urls.py

```
from django.contrib import admin
from django.urls import path
from enrollment.views import home, search_view

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', home, name='home'),
    path('search/', search_view, name='search_view'),
]
```

Step 10: Run Migrations and Start the Server

a) **Run migrations**:
   python manage.py makemigrations
   python manage.py migrate
b) python manage.py createsuperuser (Only provide Username & Password, press Enter for others)
c) Bypass password validation and create user anyway? [y/N]: y

Start the server: python manage.py runserver