ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028

A T M E

College of Engineering

DEPARTMENT OF COMPUTER SCIENCE
&
ENGINEERING

(ACADEMIC YEAR 2023-24)

LABORATORY MANUAL

SUBJECT: FULL STACK DEVELOPMENT

SUBJECT CODE: 21CS62
SEMESTER: VI

Institute Mission & Vision

Vision

- Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

Department of Computer Science & Engineering

Vision of the Department

- To develop highly talented individuals in Computer Science and Engineering to deal with real world challenges in industry, education, research and society.

Mission of the Department

- To inculcate professional behavior, strong ethical values, innovative research capabilities and leadership abilities in the young minds & to provide a teaching environment that emphasizes depth, originality and critical thinking.

- Motivate students to put their thoughts and ideas adoptable by industry or to pursue higher studies leading to research.

Program Outcomes (POs)

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, andan engineering specialization to the solution of complex engineering problems.

2. Problem Analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

Design/development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of theengineering practice.

9. Individual and Team Work: Function effectively as an individual, and as a member or leader in diverseteams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and designdocumentation, make effective presentations, and give and receive clear instructions.

11. Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

PSO1: Ability to apply skills in the field of algorithms, database design, web design, cloud computing and data analytics.

PSO2: Apply knowledge in the field of computer networks for building network and internet-based applications.

Program Educational Objectives (PEOs):

1. Empower students with a strong basis in the mathematical, scientific and engineering fundamentals to solve computational problems and to prepare them for employment, higher learning and R&D.
2. Gain technical knowledge, skills and awareness of current technologies of computer science engineering and to develop an ability to design and provide novel engineering solutions for software/hardware problems through entrepreneurial skills.
3. Exposure to emerging technologies and work in teams on interdisciplinary projects with effective communication skills and leadership qualities.
4. Ability to function ethically and responsibly in a rapidly changing environment by applying innovative ideas in the latest technology, to become effective professionals in Computer Science to bear a life-long career in related areas.

Course Syllabi with CO's

Faculty Name:			Academic Year: 2023 - 2024				
Department: Computer Science & Engineering							
Course Code	**Course Title**	**Core/AEC/Elective**	**Pre-Requisite**	**Contact Hours**			**Total Hrs/ Sessions**
				L	**T**	**P**	
21CS62	**Full Stack Development**	**Core**	**Basics of Web and Python Programming**	**3**	**-**	**2**	**40T +20P**

Course objectives:

CLO 1. Explain the use of learning full stack web development.
CLO 2. Make use of rapid application development in the design of responsive web pages.
CLO 3. Illustrate Models, Views and Templates with their connectivity in Django for full stack web development.
CLO 4. Demonstrate the use of state management and admin interfaces automation in Django.
CLO 5. Design and implement Django apps containing dynamic pages with SQL databases.

Module-1: MVC based Web Designing

Web framework, MVC Design Pattern, Django Evolution, Views, Mapping URL to Views, Working of Django URL Confs and Loose Coupling, Errors in Django, Wild Card patterns in URLS.
Textbook 1: Chapter 1 and Chapter 3

Module-2: Django Templates and Models

Template System Basics, Using Django Template System, Basic Template Tags and Filters, MVT Development Pattern, Template Loading, Template Inheritance, MVT Development Pattern. Configuring Databases, Defining and Implementing Models, Basic Data Access, Adding Model String Representations, Inserting/Updating data, Selecting and deleting objects, Schema Evolution
Textbook 1: Chapter 4 and Chapter 5

Module-3: Django Admin Interfaces and Model Forms

Activating Admin Interfaces, Using Admin Interfaces, Customizing Admin Interfaces, Reasons to use Admin Interfaces. Form Processing, Creating Feedback forms, Form submissions, custom validation, creating Model Forms, URLConf Ticks, Including Other URLConfs.
Textbook 1: Chapters 6, 7 and 8

Module-4: Generic Views and Django State Persistence

Using Generic Views, Generic Views of Objects, Extending Generic Views of objects, Extending Generic Views. MIME Types, Generating Non-HTML contents like CSV and PDF, Syndication Feed Framework, Sitemap framework, Cookies, Sessions, Users and Authentication.
Textbook 1: Chapters 9, 11 and 12

Module-5: jQuery and AJAX Integration in Django

Ajax Solution, Java Script, XHTML HttpRequest and Response, HTML, CSS, JSON, iFrames, Settings of Java Script in Django, jQuery and Basic AJAX, jQuery AJAX Facilities, Using jQuery UI Autocomplete in Django
Textbook 2: Chapters 1, 2 and 7.

Course Outcome:

At the end of the course the student will be able to:
CO 1. Understand the working of MVT based full stack web development with Django.
CO 2. Designing of Models and Forms for rapid development of web pages.
CO 3. Analyse the role of Template Inheritance and Generic views for developing full stack web applications.
CO 4. Apply the Django framework libraries to render non-HTML contents like CSV and PDF.
CO 5. Perform jQuery based AJAX integration to Django Apps to build responsive full stack web applications,

The Correlation of Course Outcomes (CO's) and Program Outcomes (PO's)

Subject Code:	21CS62	TITLE: Full Stack Development						Faculty Name:				
List of Course Outcomes	Program Outcomes											Total
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	
CO-1	-	2	1	-	1	2	-	-	-	-	-	1	7
CO-2	2	2	2	3	1	2	-	-	-	-	-	-	12
CO-3	2	2	2	3	3	-	-	-	-	-	-	2	14
CO-4	2	1	2	3	1	2	-	-	-	-	-	-	11
CO-5	1	2	2	1	2	-	-	-	-	-	-	-	8
Total	7	9	9	10	8	6	-	-	-	-	-	3	52

The Correlation of Course Outcomes (CO's) and Program Specific Outcomes (PSO's)

Subject Code:	21CS62	TITLE: Full Stack Development	Faculty Name:	
List of Course Outcomes	Program Specific Outcomes			Total
	PSO-1	PSO-2		
CO-1	2	-		2
CO-2	3	1		4
CO-3	2	-		2
CO-4	3	1		4
CO-5	2	1		3
Total	12	3		15

Note: 3 = Strong Contribution, 2 = Average Contribution, 1 = Weak Contribution, - = No Contribution

Module-1

MVC based Web Designing

Laboratory Component:

1. **Installation of Python, Django and Visual Studio code editors can be demonstrated.**

   Python download and installation Link:
   https://www.python.org/downloads/

   Visual Studio Code download and installation link:
   https://code.visualstudio.com/

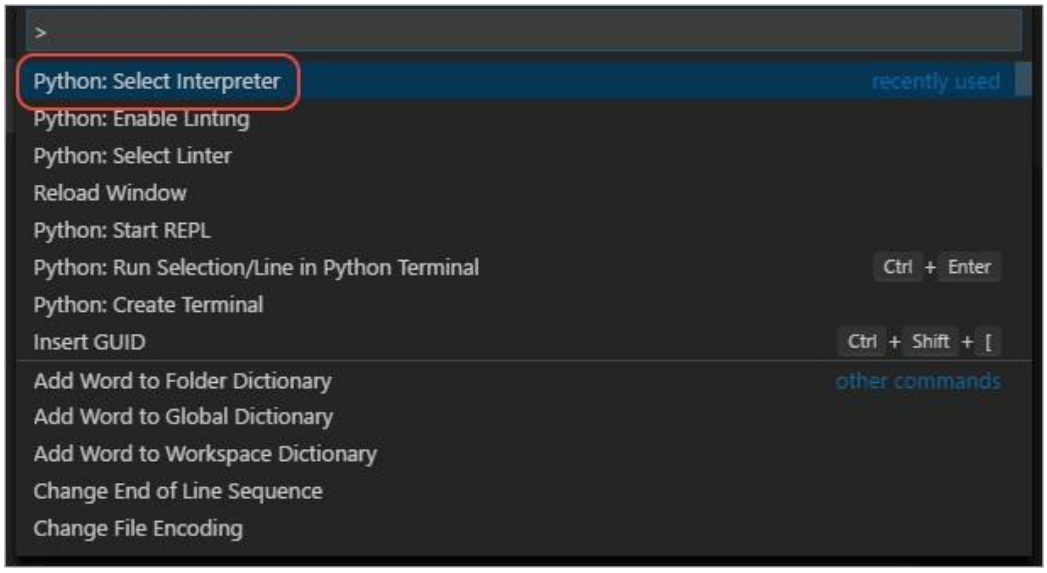
   Django installation:
   Open a command prompt and type following command:
   pip install django

2. **Creation of virtual environment, Django project and App should be demonstrated Follow these steps**

1. Install the Python extension.- Open VS Code IDE and click extensions there automatically u will be shown Python extension (Make sure you are connected to Internet)
2. On your file system, create a project folder
3. In that folder, use the following command (as appropriate to your computer) to create a virtual environment named `env` based on your current interpreter:

```
# Windows
python -m venv env
```

4. Open the project folder in VS Code by running `code .`, or by running VS Code and using the **File** > **Open Folder** command.
5. In VS Code, open the Command Palette (**View** > **Command Palette** or (Ctrl+Shift+P)). Then select the **Python: Select Interpreter** command:



6. The command presents a list of available interpreters that VS Code can locate automatically (your list will vary; if you don't see the desired interpreter, see Configuring Python environments). From the list, select the virtual environment in your project folder that starts with ./env or .\env:
7. Create a New Terminal : In Menu Terminal -> New Terminal option

Creating project:

1. Create a django project -

   Type following command in the terminal opened:

   `django-admin startproject p .`

   (dot following project name is important which refers to current directory)

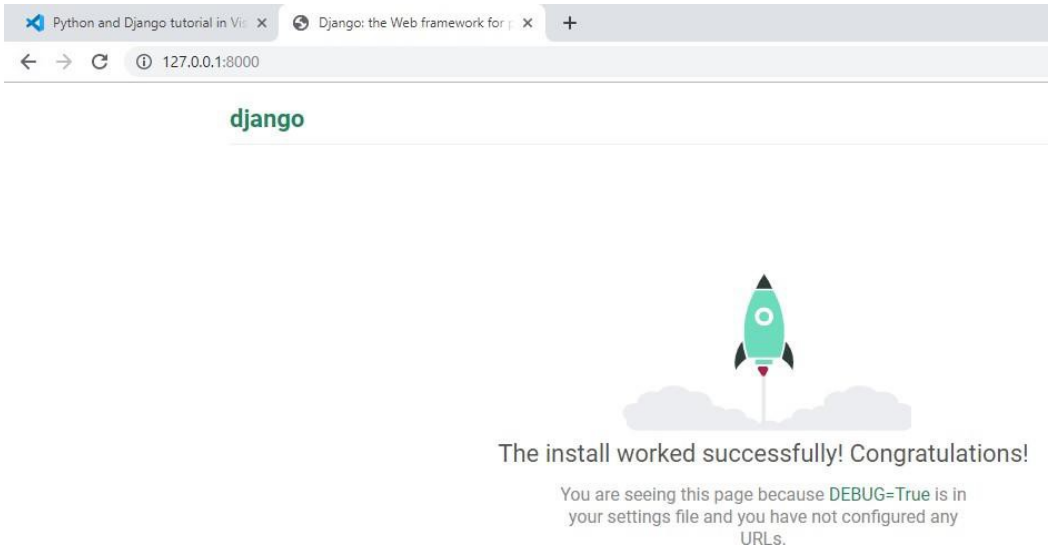
This startproject command assumes (by use of . at the end) that the current folder is your project folder, and creates the following within it:

- manage.py: The Django command-line administrative utility for the project. You run administrative commands for the project using python manage.py <command> [options].
- A subfolder named p which contains the following files:
  - **__init__.py:** an empty file that tells Python that this folder is a Python package.
  - **wsgi.py:** an entry point for WSGI-compatible web servers to serve your project. You typically leave this file as-is as it provides the hooks for production web servers.
  - **settings.py:** contains settings for Django project, which you modify in the course of developing a web app.
  - **urls.py:** contains a table of contents for the Django project, which you also modify in the course of development.

2. To verify the Django project, make sure your virtual environment is activated, then start Django's development server using the command **python manage.py runserver.** The server runs on the default port 8000, and you see output like the following output in the terminal window:

Verify server by typing:

python manage.py runserver



When you run the server the first time, it creates a default SQLite database in the file db.sqlite3, which is intended for development purposes but can be used in production for low-volume web apps. Also, Django's built-in web server is intended *only* for local development purposes. When you deploy to a web host, however, Django uses the host's web server instead. The wsgi.py module in the Django project takes care of hooking into the production servers.

If you want to use a different port than the default 8000, specify the port number on the command line, such as python manage.py runserver 5000.

3. When you're done, close the browser window and stop the server in VS Code using Ctrl+C as indicated in the terminal output window.
4. In the VS Code Terminal with your virtual environment activated, run the administrative utility's startapp command in your project folder (where manage.py resides):

python manage.py startapp lab1

5. The command creates a folder called lab1 that contains a number of code files and one subfolder. Of these, you frequently work with views.py (that contains the functions that define pages in your web app) and models.py (that contains classes defining your data objects). The migrations folder is used by Django's administrative utility to manage database versions. There are also the files apps.py (app configuration), admin.py (for creating an administrative interface), and tests.py (for unit tests).

3. Develop a Django app that displays current date and time in server

In lab1 subfolder, make following changes to views.py:

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
import datetime
def current_datetime(request):
    now = datetime.datetime.now()
    html = "<html><body><h1>It is now %s.</h1></body></html>" % now
    return HttpResponse(html)
```

In project named first, make following changes to urls.py

```
from django.contrib import admin
from django.urls import path
from lab11.views import current_datetime
urlpatterns = [

    path('cdt/', current_datetime),
```

Output:

It is now 2024-02-03 18:42:01.631243.

4. **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.**

In lab11 subfolder, make following changes to views.py:

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
import datetime
def current_datetime(request):
    now = datetime.datetime.now()

    html = "<html><body><h1>It is now %s.</h1></body></html>" % now
    return HttpResponse(html)
def four_hours_ahead(request):

    dt = datetime.datetime.now() + datetime.timedelta(hours=4)
    html = "<html><body><h1>After 4hour(s), it will be %s.</h1>"% (dt,)
    return HttpResponse(html)

def four_hours_before(request):

    dt = datetime.datetime.now() + datetime.timedelta(hours=-4)
    html = "<html><body><h1>Before 4 hour(s), it was %s.</h1>"% (dt,)
    return HttpResponse(html)
```

In project named first, make following changes to urls.py

```
from django.contrib import admin
from django.urls import path
from lab11.views import  current_datetime,four_hours_ahead,four_hours_before
urlpatterns = [

    path('cdt/', current_datetime),
    path('fhrsa/',four_hours_ahead),
    path('fhrsb/',four_hours_before),
```

Output:

After 4hour(s), it will be 2024-02-03 22:43:50.544397.

Before 4 hour(s), it was 2024-02-03 14:44:10.994024.

Module-2

Django Templates and Models

Laboratory Component:

1. **Develop a simple Django app that displays an unordered list of fruits and ordered list of selected students for an event**

Views.py

```
from datetime import date
from django.http import HttpResponse
from django.shortcuts import render
from django.template import Context, Template

# Create your views here.
def showlist(request):
   fruits=["Mango","Apple","Bananan","Jackfruits"]
   student_names=["Tony","Mony","Sony","Bob"]
   return
render(request,'showlist.html',{"fruits":fruits,"student_names":student_names}
)
```

URLS.py

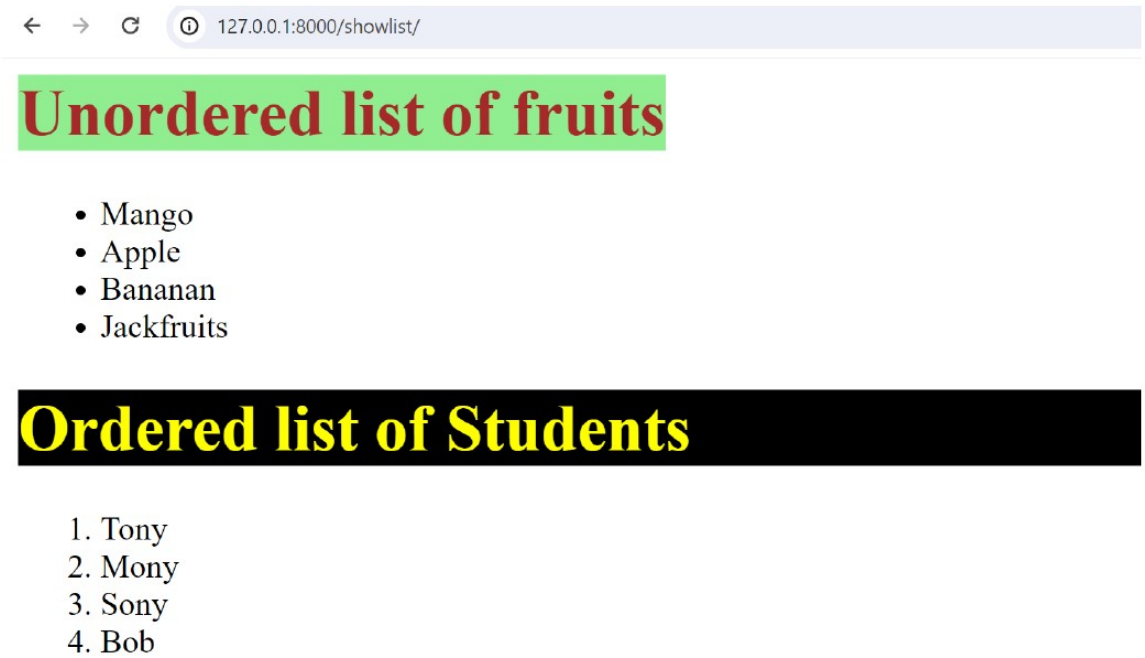
```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist
urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
    path('cts/<int:s>/<int:n>', create_table_of_squares),
    path('vc/<str:sentence>', vc),
    path('find_mode/<str:listofnum>', find_mode),
    path('template_test/', template_test),
    path('showlist/', showlist),
```

Template HTML file (inside ap2/templates subfolder)

showlist.html

```
<html>
    <style type="text/css">
        #i1 {background-color: lightgreen;color:brown;display:table}
        #i2 {background-color: black;color:yellow}
    </style>
    <body>
        <h1 id="i1">Unordered list of fruits</h1>
        <ul>
            {% for fruit in fruits %}
            <li>{{ fruit }}</li>
            {% endfor %}

        </ul>
        <h1 id="i2">Ordered list of Students</h1>
        <ol>
            {% for student in student_names %}
            <li>{{ student }}</li>
            {% endfor %}

        </ol>
    </body>
</html>
```

Output:



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.**

Views.py

```
from datetime import date
from django.http import HttpResponse
from django.shortcuts import render
from django.template import Context, Template
def home(request):
   return render(request,'home.html')

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

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

URLS.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus
urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
```

```
path('cts/<int:s>/<int:n>', create_table_of_squares),
path('vc/<str:sentence>', vc),
path('find_mode/<str:listofnum>', find_mode),
path('template_test/', template_test),
path('showlist/', showlist),
path('list_of_subjects/', list_of_subjects),
path('aboutus/', aboutus),
path('home/', home),
path('contactus/', contactus),
```

Template files:

layout.html

```
<html>
    <title>{% block title %} {% endblock %} </title>
    <style type="text/css">
        nav {background-color: lightblue;padding:10px}
    </style>
    <body>
        <nav>
            <a href="/home/">Home</a>|
            <a href="/aboutus/">About Us</a>|
            <a href="/contactus/">Contact Us</a>|
        </nav>
        <section>
            {% block content %}{% endblock %}
        </section>
        <footer>
            <hr>
            &copy; AIML, Developed by ABC, Inc.
        </footer>
    </body>
</html>
```

home.html

```
{% extends 'layout.html' %}
{% block title %}
Home
{% endblock %}
{% block content %}
<h2>This is the home page</h2>
{% endblock %}
```

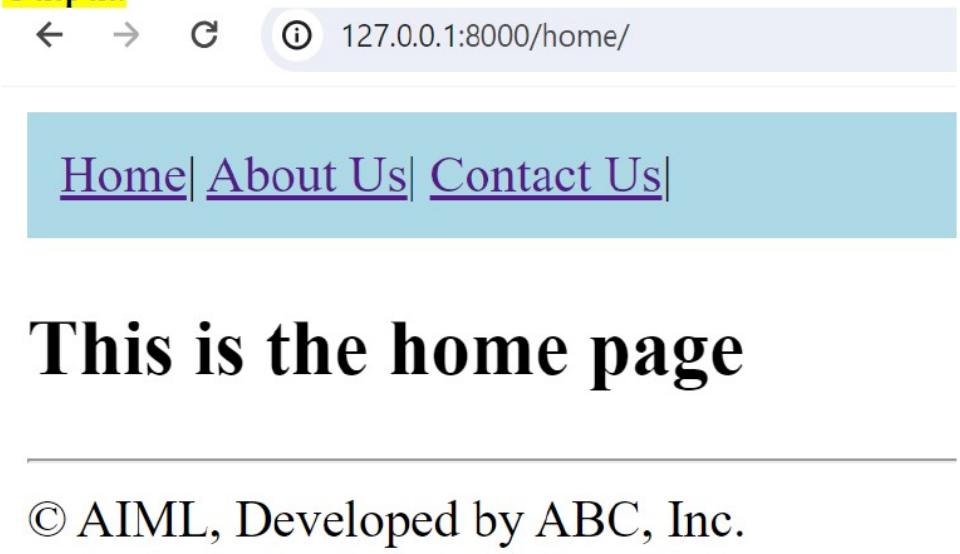
aboutus.html

```
{% extends 'layout.html' %}
{% block title %}
About Us
{% endblock %}
{% block content %}
<h2>We are DJango developers</h2>
{% endblock %}
```

contactus.html

```
{% extends 'layout.html' %}
{% block title %}
Contact us
{% endblock %}
{% block content %}
<h2>Out phone: 9900923050 <br>
Address: Navule JNNCE</h2>
{% endblock %}
```

Output:



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**

models.py

```
from django.db import models

# Create your models here.
class Course(models.Model):
    course_code=models.CharField(max_length=40)
    course_name=models.CharField(max_length=100)
    course_credits=models.IntegerField()

class Student(models.Model):
    student_usn=models.CharField(max_length=20)
    student_name=models.CharField(max_length=100)
    student_sem=models.IntegerField()
    enrolment=models.ManyToManyField(Course)
```

reg.html inside templates folder

```
<html>
    <body>
    <form method="post" action="">
        {% csrf_token %}
        Student Name
        <select name="sname">
        {%for student in students %}
        <option value="{{student.id}}">{{student.student_name}}</option>
        {% endfor %}
        </select><br>
        Course Name
        <select name="cname">
        {%for course in courses %}
        <option value="{{course.id}}">{{course.course_name}}</option>
        {% endfor %}

        </select><br>
        <input type="submit" value="Enroll">
    </form>
    </body>
</html>
```

views.py

```
from django.http import HttpResponse
from django.shortcuts import render

from ap3.models import Course, Meeting, Student

def reg(request):
    if request.method == "POST":
        sid=request.POST.get("sname")
        cid=request.POST.get("cname")
        student=Student.objects.get(id=sid)
        course=Course.objects.get(id=cid)
        res=student.enrolment.filter(id=cid)
        if res:
            return HttpResponse("<h1>Student already enrolled</h1>")
        student.enrolment.add(course)
        return HttpResponse("<h1>Student enrolled successfully</h1>")

    else:
        students=Student.objects.all()
        courses=Course.objects.all()

       return render(request,"reg.html",{"students":students,
"courses":courses})
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg
urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
```

```
path('cts/<int:s>/<int:n>', create_table_of_squares),
path('vc/<str:sentence>', vc),
path('find_mode/<str:listofnum>', find_mode),
path('template_test/', template_test),
path('showlist/', showlist),
path('list_of_subjects/', list_of_subjects),
path('aboutus/', aboutus),
path('home/', home),
path('contactus/', contactus),
path('getpos/', getpos),
path('stable/', stable),
path('insert_demo/', insert_demo),
path('update_demo/', update_demo),
path('delete_demo/', delete_demo),
path('retreive_demo/', retreive_demo),
path('reg/', reg),
```

Database input:
Insert student and courses record in phpMyAdmin

Browse | Structure | SQL | Search | Insert | Export

Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

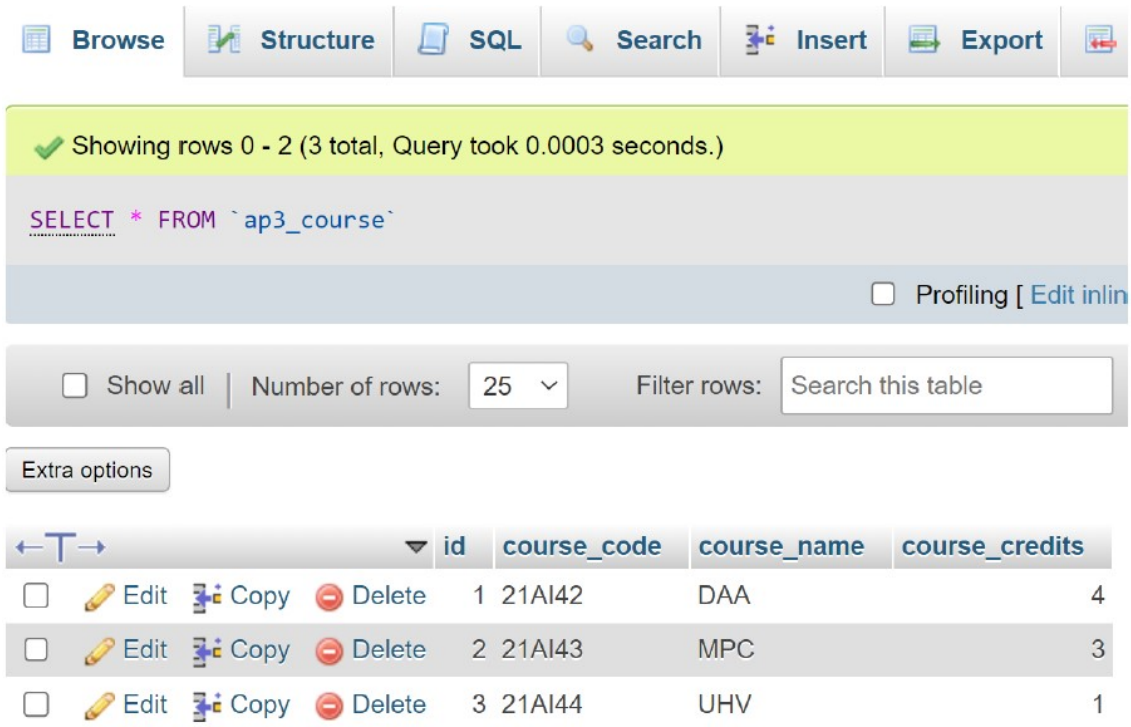
```
SELECT * FROM `ap3_student`
```

Profiling [Edit inlin

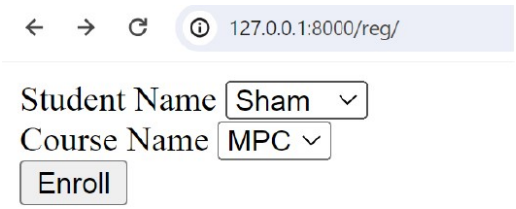
Show all | Number of rows: 25 Filter rows: Search this table

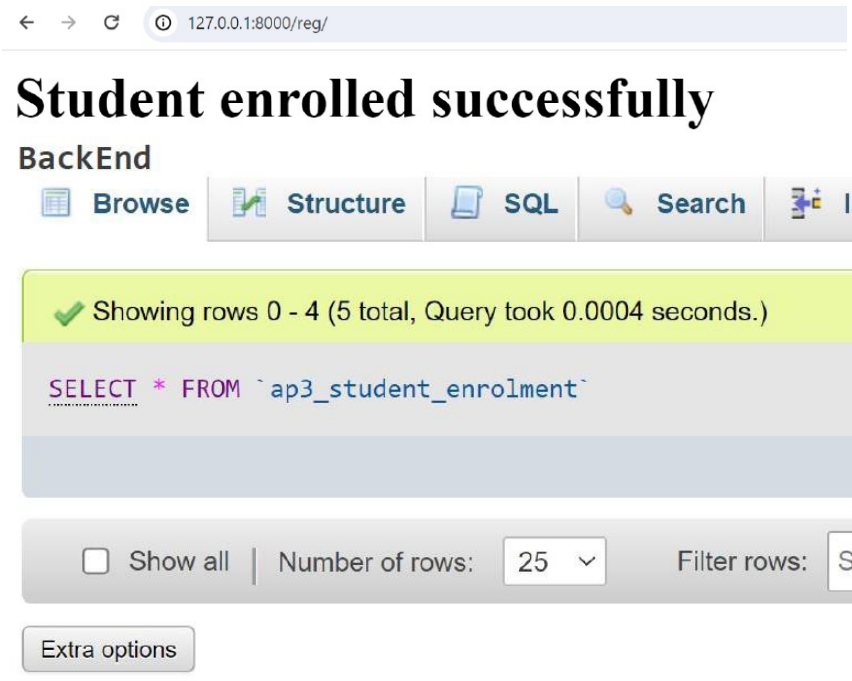
Extra options

	id	student_usn	student_name	student_sem
Edit Copy Delete	1	4JN22AI001	Sham	4
Edit Copy Delete	2	4JN22AI002	Manish	4
Edit Copy Delete	3	4JN22AI003	Suma	4


Browse
Structure
SQL
Search
Insert
Export
Showing rows 0 - 2 (3 total, Query took 0.0003 seconds.)
SELECT * FROM `ap3_course`
Profiling [Edit inlin
Show all
Number of rows: 25
Filter rows: Search this table
Extra options
id course_code course_name course_credits
Edit Copy Delete 1 21AI42 DAA 4
Edit Copy Delete 2 21AI43 MPC 3
Edit Copy Delete 3 21AI44 UHV 1

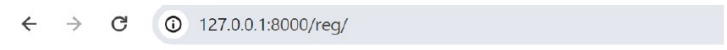
OUTPUT:


127.0.0.1:8000/reg/
Student Name Sham
Course Name MPC
Enroll


127.0.0.1:8000/reg/
Student enrolled successfully
BackEnd
Browse
Structure
SQL
Search
Insert
Showing rows 0 - 4 (5 total, Query took 0.0004 seconds.)
SELECT * FROM `ap3_student_enrolment`
Show all
Number of rows: 25
Filter rows:
Extra options

←T→				id	student_id	course_id
☐	Edit	Copy	Delete	1	1	1
☐	Edit	Copy	Delete	2	1	3
☐	Edit	Copy	Delete	3	2	2
☐	Edit	Copy	Delete	4	2	1
☐	Edit	Copy	Delete	5	1	2

If you try again, you will get



Student already enrolled

Module-3

Django Admin Interfaces and Model Forms

1. **For student and course models created in Lab experiment for Module2, register admin interfaces, perform migrations and illustrate data entry through admin forms**

python manage.py createsuperuser

make following changes to admin.py

```
from django.contrib import admin

from ap3.models import Course, Student

# Register your models here.
#admin.site.register(Student)
@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ('student_name','student_usn','student_sem')
    ordering=('student_name',)
    search_fields = ('student_name',)
admin.site.register(Course)
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search
admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"

urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
```

```
path('cts/<int:s>/<int:n>', create_table_of_squares),
path('vc/<str:sentence>', vc),
path('find_mode/<str:listofnum>', find_mode),
path('template_test/', template_test),
path('showlist/', showlist),
path('list_of_subjects/', list_of_subjects),
path('aboutus/', aboutus),
path('home/', home),
path('contactus/', contactus),
path('getpos/', getpos),
path('stable/', stable),
path('insert_demo/', insert_demo),
path('update_demo/', update_demo),
path('delete_demo/', delete_demo),
path('retreive_demo/', retreive_demo),
path('reg/', reg),
path('course_search/', course_search),
```

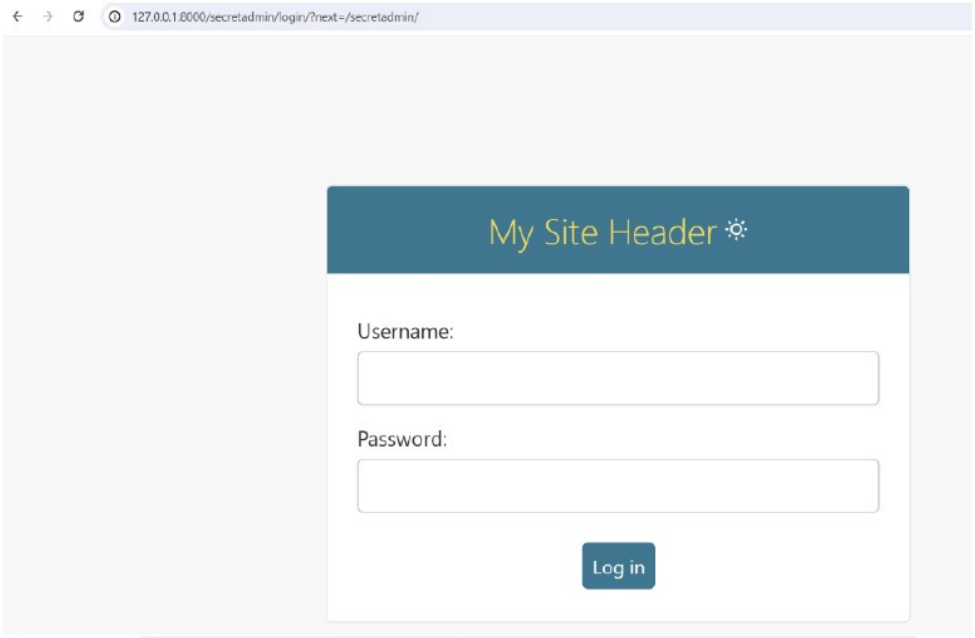
Changes to models.py

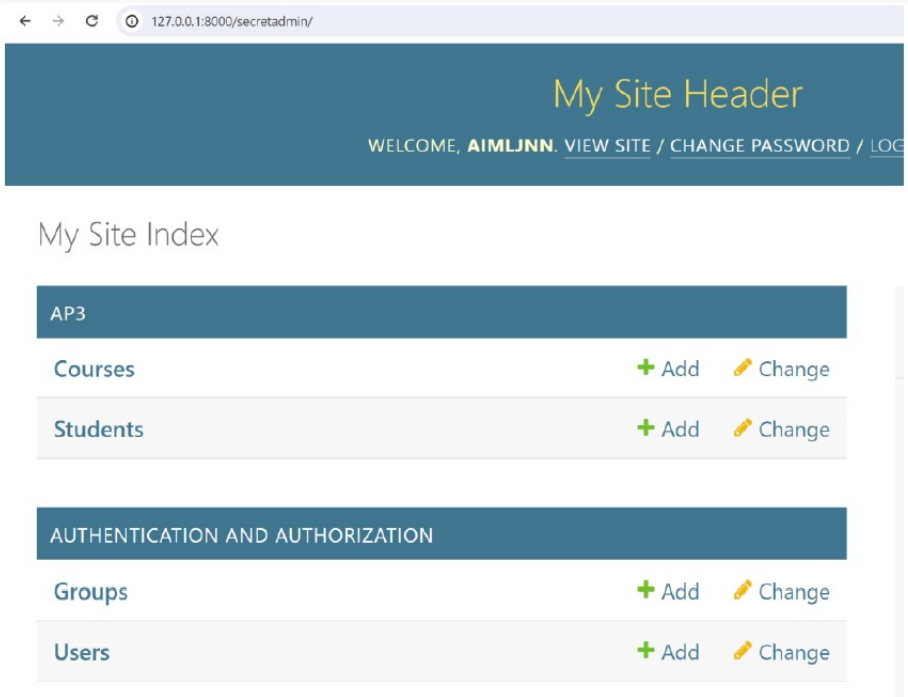
```
from django.db import models
from django.forms import ModelForm

# Create your models here.
class Meeting(models.Model):
    meeting_code=models.CharField(max_length=100)
    meeting_dt=models.DateField(auto_now_add=True)
    meeting_subject=models.CharField(max_length=100)
    meeting_np=models.IntegerField()

class Course(models.Model):
    course_code=models.CharField(max_length=40)
    course_name=models.CharField(max_length=100)
    course_credits=models.IntegerField(blank=True, null=True)
    def __str__(self):
        return self.course_name

class Student(models.Model):
    student_usn=models.CharField(max_length=20)
    student_name=models.CharField(max_length=100)
    student_sem=models.IntegerField()
    enrolment=models.ManyToManyField(Course)
    def __str__(self):
        return self.student_name+"("+self.student_usn+")"
```

Perform remigrations before running:

```
python manage.py makemigrations ap3

python manage.py migrate
```

Output:





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

models.py

```
from django.db import models
from django.forms import ModelForm

# Create your models here.
class Meeting(models.Model):
    meeting_code=models.CharField(max_length=100)
    meeting_dt=models.DateField(auto_now_add=True)
    meeting_subject=models.CharField(max_length=100)
    meeting_np=models.IntegerField()

class Course(models.Model):
    course_code=models.CharField(max_length=40)
    course_name=models.CharField(max_length=100)
    course_credits=models.IntegerField(blank=True, null=True)
    def __str__(self):
        return self.course_name

class Student(models.Model):
    student_usn=models.CharField(max_length=20)
    student_name=models.CharField(max_length=100)
    student_sem=models.IntegerField()
    enrolment=models.ManyToManyField(Course)
    def __str__(self):
        return self.student_name+"("+self.student_usn+")"
class Project(models.Model):
    student=models.ForeignKey(Student,on_delete=models.CASCADE)
    ptopic=models.CharField(max_length=200)
    plangauges=models.CharField(max_length=200)
    pduration=models.IntegerField()

class ProjectReg(ModelForm):
    required_css_class="required"
    class Meta:
        model=Project
        fields=['student','ptopic','plangauges','pduration']
```

views.py

```
from django.http import HttpResponse
from django.shortcuts import render

from ap3.models import Course, Meeting, ProjectReg, Student


def add_project(request):
    if request.method=="POST":
        form=ProjectReg(request.POST)
        if form.is_valid():
            form.save()
            return HttpResponse("<h1>Record inserted successfully</h1>")
         else:
            return HttpResponse("<h1>Record not inserted</h1>")
      else:
        form=ProjectReg()
        return render(request,"add_project.html",{"form":form})
```

add_project.html inside templates folder

```
<html>
    <form method="post" action="">
        {% csrf_token %}
        <table>
            {{ form.as_table}}
            <tr>
                <td>
                    <input type="submit" value="Submit">
                </td>
            </tr>
        </table>
    </form>
</html>
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
```

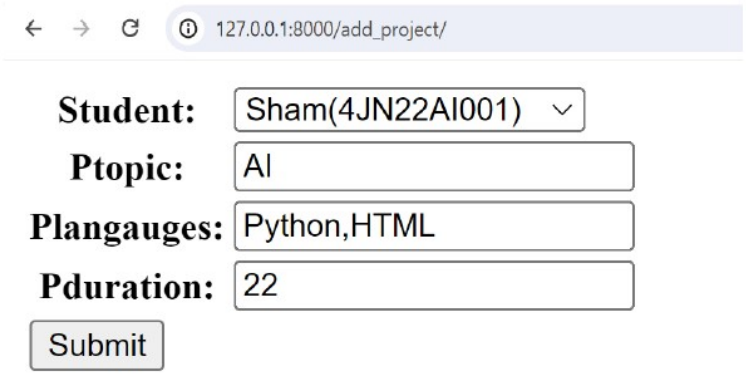
```
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project

admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
    path('cts/<int:s>/<int:n>', create_table_of_squares),
    path('vc/<str:sentence>', vc),
    path('find_mode/<str:listofnum>', find_mode),
    path('template_test/', template_test),
    path('showlist/', showlist),
    path('list_of_subjects/', list_of_subjects),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    path('getpos/', getpos),
    path('stable/', stable),
    path('insert_demo/', insert_demo),
    path('update_demo/', update_demo),
    path('delete_demo/', delete_demo),
    path('retreive_demo/', retreive_demo),
    path('reg/', reg),
    path('course_search/', course_search),
    path('add_project/', add_project),
```

Perform remigrations before running:

python manage.py makemigrations ap3

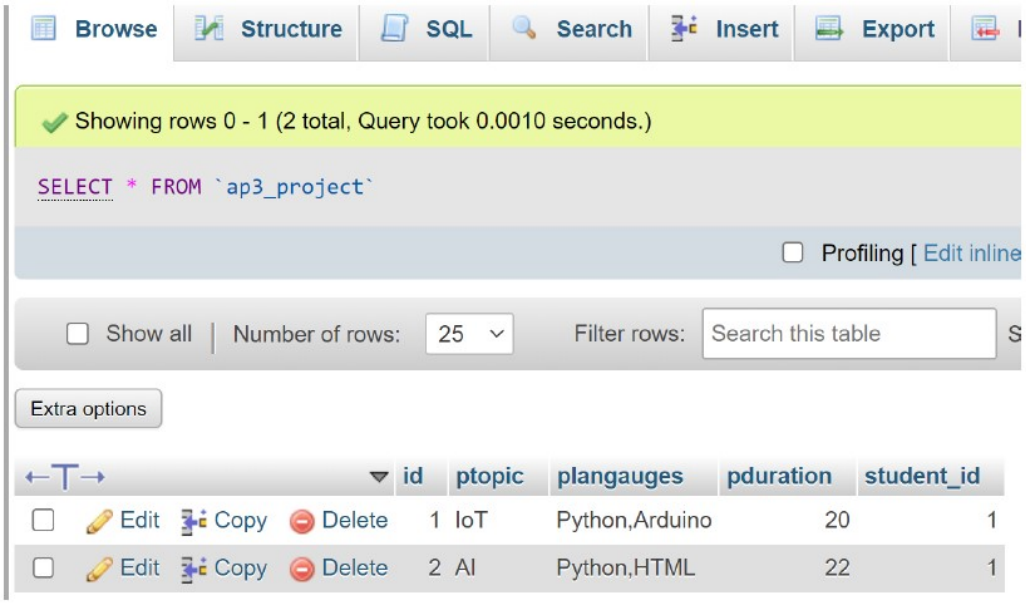
python manage.py migrate

Output:



Record inserted successfully

BackEnd



Module-4

Generic Views and Django State Persistence

1. **For student's 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**

views.py

```
from django.views import generic
class StudentListView(generic.ListView):
    model=Student
    template_name="student_list.html"

class StudentDetailView(generic.DetailView):
    model=Student
    template_name="student_detail.html"
```

student_list.html inside templates folder

```
<html>
    <body>
        {% if  student_list %}
        <table border>
            <tr>
                <th>USN</th>
                <th>Courses Enrolled</th>
            </tr>
            {% for student in student_list %}
            <tr>
                <td><a href="/student_detail/{{student.pk}}">{{
student.student_usn }}</a></td>
                <td>{% for course in student.enrolment.all %}
                    <span>{{ course.course_name }}</span>
                    {% endfor %}
                </td>
```

```
            </tr>
            {% endfor %}
        </table>
        {% else %}
          <h1>No Students Enrolled</h1>
        {% endif %}


    </body>
</html>
```

student_detail.html inside templates folder

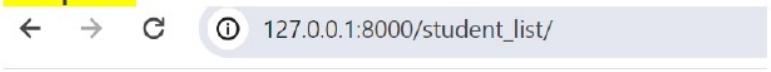
```
<h1>Student Name: {{ student.student_name }}</h1>
<h1>Student USN: {{ student.student_usn }}</h1>
<h1>Student Sem: {{ student.student_sem }}</h1>
```

urls.py

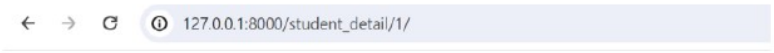
```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project
from ap3.views import StudentListView,StudentDetailView

admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
```

```
re_path('check_number/(\d){1,2}/',check_number),
path('cts/<int:s>/<int:n>', create_table_of_squares),
path('vc/<str:sentence>', vc),
path('find_mode/<str:listofnum>', find_mode),
path('template_test/', template_test),
path('showlist/', showlist),
path('list_of_subjects/', list_of_subjects),
path('aboutus/', aboutus),
path('home/', home),
path('contactus/', contactus),
path('getpos/', getpos),
path('stable/', stable),
path('insert_demo/', insert_demo),
path('update_demo/', update_demo),
path('delete_demo/', delete_demo),
path('retreive_demo/', retreive_demo),
path('reg/', reg),
path('course_search/', course_search),
path('add_project/', add_project),
path('student_list/', StudentListView.as_view()),
path('student_detail/<int:pk>/', StudentDetailView.as_view()),
```

Output:



USN	Courses Enrolled
4JN22AI001	DAA MPC UHV
4JN22AI002	DAA MPC
4JN22AI003	



Student Name: Sham

Student USN: 4JN22AI001

Student Sem: 4

2. **Develop example Django app that performs CSV generation for any models created in previous laboratory component**

views.py

```
from django.http import HttpResponse
from django.shortcuts import render

from ap3.models import Course, Meeting, ProjectReg, Student

import csv
def construct_csv_from_model(request):
    courses=Course.objects.all()
    response=HttpResponse(content_type="text/csv")
    response['Content-Disposition'] = 'attachment;
filename="courses_data.csv"'
    writer=csv.writer(response)
    writer.writerow(["Course Name","Course Code","Credits"])
    for course in courses:
        writer.writerow([course.course_name,course.course_code,course.course_c
redits])
    return response
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project
from ap3.views import StudentListView,StudentDetailView,construct_csv
from ap3.views import construct_csv_from_model

admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re path('check number/(\d){1,2}/',check number),
```

```
path('cts/<int:s>/<int:n>', create_table_of_squares),
path('vc/<str:sentence>', vc),
path('find_mode/<str:listofnum>', find_mode),
path('template_test/', template_test),
path('showlist/', showlist),
path('list_of_subjects/', list_of_subjects),
path('aboutus/', aboutus),
path('home/', home),
path('contactus/', contactus),
path('getpos/', getpos),
path('stable/', stable),
path('insert_demo/', insert_demo),
path('update_demo/', update_demo),
path('delete_demo/', delete_demo),
path('retreive_demo/', retreive_demo),
path('reg/', reg),
path('course_search/', course_search),
path('add_project/', add_project),
path('student_list/', StudentListView.as_view()),
path('student_detail/<int:pk>/', StudentDetailView.as_view()),
path('construct_csv/', construct_csv),
path('construct_csv_from_model/', construct_csv_from_model),
```

Output:

CSV file is generated and downloaded

3. **Develop example Django app that performs PDF generation for any models created in previous laboratory component**

views.py

```
from django.http import HttpResponse
from django.shortcuts import render

from ap3.models import Course, Meeting, ProjectReg, Student
from reportlab.pdfgen import canvas

def construct_pdf_from_model(request):
    courses=Course.objects.all()
    response=HttpResponse(content_type="application/pdf")
    response['Content-Disposition'] = 'attachment;
filename="courses_data.pdf"'
    c=canvas.Canvas(response)

    c.drawString(70,720,"Course Name")
    c.drawString(170,720,"Course Code")
    c.drawString(270,720,"Credits")
    y=660
    for course in courses:
        c.drawString(70,y,course.course_name)
        c.drawString(170,y,course.course_code)
        c.drawString(270,y,str(course.course_credits))
        y=y-60
    c.showPage()
    c.save()
    return response
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
```

```
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project
from ap3.views import StudentListView,StudentDetailView,construct_csv
from ap3.views import construct_csv_from_model
from ap3.views import construct_pdf_from_model
admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
    path('cts/<int:s>/<int:n>', create_table_of_squares),
    path('vc/<str:sentence>', vc),
    path('find_mode/<str:listofnum>', find_mode),
    path('template_test/', template_test),
    path('showlist/', showlist),
    path('list_of_subjects/', list_of_subjects),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    path('getpos/', getpos),
    path('stable/', stable),
    path('insert_demo/', insert_demo),
    path('update_demo/', update_demo),
    path('delete_demo/', delete_demo),
    path('retreive_demo/', retreive_demo),
    path('reg/', reg),
    path('course_search/', course_search),
    path('add_project/', add_project),
    path('student_list/', StudentListView.as_view()),
    path('student_detail/<int:pk>/', StudentDetailView.as_view()),
    path('construct_csv/', construct_csv),
    path('construct_csv_from_model/', construct_csv_from_model),
    path('construct_pdf_from_model/', construct_pdf_from_model),
```

Output:

PDF file is generated and downloaded

Module-5

jQuery and AJAX Integration in Django

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

views.py

```
from django.http import HttpResponse
from django.shortcuts import render

from ap3.models import Course, Meeting, ProjectReg, Student

def regaj(request):
    if request.method == "POST":
        sid=request.POST.get("sname")
        cid=request.POST.get("cname")
        student=Student.objects.get(id=sid)
        course=Course.objects.get(id=cid)
        res=student.enrolment.filter(id=cid)
        if res:
            return HttpResponse("<h1>Student already enrolled</h1>")
        student.enrolment.add(course)
        return HttpResponse("<h1>Student enrolled successfully</h1>")

    else:
        students=Student.objects.all()
        courses=Course.objects.all()

        return render(request,"regaj.html",{"students":students,
"courses":courses})
```

regaj.html

```
{% load static %}
<html>
    <body>
    <form method="post" action="">
        {% csrf_token %}
        Student Name
        <select name="sname" id="sname">
        {%for student in students %}
        <option value="{{student.id}}">{{student.student_name}}</option>
        {% endfor %}
```

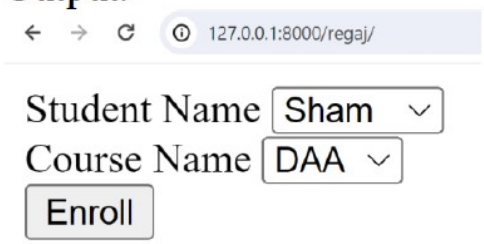
```
    </select><br>
    Course Name
    <select name="cname" id="cname">
    {%for course in courses %}
    <option value="{{course.id}}">{{course.course_name}}</option>
    {% endfor %}

    </select><br>
    <span id="ans"></span>
    <input type="button" value="Enroll" id="ebtn">
</form>
<script src="{% static 'jquery.min.js' %}"></script>
<script>
    $(document).ready(function(){
        $("#ebtn").click(function(){
            var sname = $("#sname").val();
            var cname = $("#cname").val();
            $.ajax({
                type: "POST",
                url: "/regaj/",
                data: {sname: sname, cname: cname,
                csrfmiddlewaretoken:"{{ csrf_token}}"
                },
                        success: function(response){
                            $("#ans").html(response)
                        }
                    });
                });
            });
      </script>
      </body>
  </html>
```

urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
```

```
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project
from ap3.views import StudentListView,StudentDetailView,construct_csv
from ap3.views import construct_csv_from_model,construct_pdf
from ap3.views import construct_pdf_from_model
from ap3.views import expr,cbox,regaj,course_search_ajax
admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [
    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
    path('cts/<int:s>/<int:n>', create_table_of_squares),
    path('vc/<str:sentence>', vc),
    path('find_mode/<str:listofnum>', find_mode),
    path('template_test/', template_test),
    path('showlist/', showlist),
    path('list_of_subjects/', list_of_subjects),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    path('getpos/', getpos),
    path('stable/', stable),
    path('insert_demo/', insert_demo),
    path('update_demo/', update_demo),
    path('delete_demo/', delete_demo),
    path('retreive_demo/', retreive_demo),
    path('reg/', reg),
    path('course_search/', course_search),
    path('add_project/', add_project),
    path('student_list/', StudentListView.as_view()),
    path('student_detail/<int:pk>/', StudentDetailView.as_view()),
    path('construct_csv/', construct_csv),
    path('construct_csv_from_model/', construct_csv_from_model),
    path('construct_pdf/', construct_pdf),
    path('construct_pdf_from_model/', construct_pdf_from_model),
    path('expr/', expr),
    path('cbox/', cbox),
    path('regaj/',regaj),
```

Output:



Student already enrolled

Enroll

Student enrolled successfully

Enroll

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

views.py

```
from django.http import HttpResponse

from django.shortcuts import render

from ap3.models import Course, Meeting, ProjectReg, Student

def course_search_ajax(request):
    if request.method=="POST":
        cid=request.POST.get("cname")
        s=Student.objects.all()
        student_list=list()
        for student in s:
            if student.enrolment.filter(id=cid):
                student_list.append(student)
        if len(student_list)==0:
            return HttpResponse("<h1>No Students enrolled</h1>")
        return
render(request,"selected_students.html",{"student_list":student_list})

    else:
        courses=Course.objects.all()
        return render(request,"course_search_aj.html",{"courses":courses})
```

course_search_aj.html

```
{% load static %}
<html>
    <body>
        <form method="POST" action="">
        Courses
        {% csrf_token %}
        <select name="cname" id="cname">
            {%for course in courses %}
            <option value="{{course.id}}">{{course.course_name}}</option>
            {% endfor %}
        </select>
        <input type="button" value="Search" id="serbtn">
        <span id="result"></span>
    </form>
    </body>
    <script src="{% static 'jquery.min.js' %}"></script>
    <script>
        $(document).ready(function(){
            $("#serbtn").click(function(){
                var cname = $("#cname").val();
                $.ajax({
                    url: "/course_search_ajax/",
                    type: "POST",
                    data: {cname:cname,csrfmiddlewaretoken:"{{csrf_token }}"},
                    success: function(response){
                        $("#result").html(response);
```

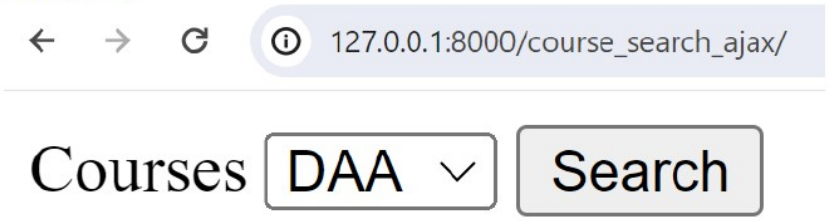
```
                    }
                });
            });
        });
    </script>
</html>
```

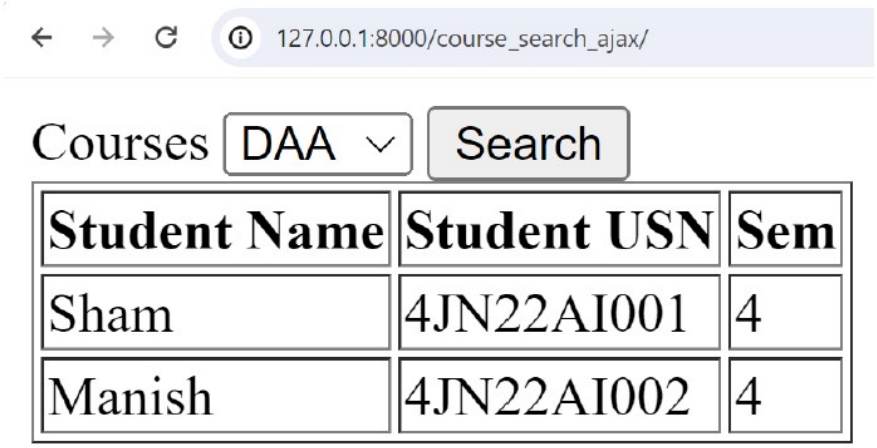
urls.py

```
from django.contrib import admin
from django.urls import path, re_path
from ap1.views import check_number, current_date_time
from ap1.views import four_hours_after, four_hours_before
from ap1.views import n_hours_after,display_string
from ap2.views import create_table_of_squares,vc,find_mode
from ap2.views import template_test,showlist,list_of_subjects
from ap2.views import aboutus,home,contactus,getpos,stable
from ap3.views import insert_demo,update_demo,delete_demo,retreive_demo
from ap3.views import reg,course_search,add_project
from ap3.views import StudentListView,StudentDetailView,construct_csv
from ap3.views import construct_csv_from_model,construct_pdf
from ap3.views import construct_pdf_from_model
from ap3.views import expr,cbox,regaj,course_search_ajax,cookie_demo
admin.site.site_header="My Site Header"
admin.site.site_title="My Site Title"
admin.site.index_title="My Site Index"
urlpatterns = [

    path('secretadmin/', admin.site.urls),
    path('cdt/', current_date_time),
    path('fha/', four_hours_after),
    path('fhb/', four_hours_before),
    path('nha/<int:num>', n_hours_after),
    path('display_string/<slug:sentence>', display_string),
    re_path('check_number/(\d){1,2}/',check_number),
    path('cts/<int:s>/<int:n>', create_table_of_squares),
    path('vc/<str:sentence>', vc),
    path('find_mode/<str:listofnum>', find_mode),
    path('template_test/', template_test),
    path('showlist/', showlist),
    path('list_of_subjects/', list_of_subjects),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    path('getpos/', getpos),
    path('stable/', stable),
    path('insert_demo/', insert_demo),
    path('update_demo/', update_demo),
    path('delete_demo/', delete_demo),
```

```
path('retreive_demo/', retreive_demo),
path('reg/', reg),
path('course_search/', course_search),
path('add_project/', add_project),
path('student_list/', StudentListView.as_view()),
path('student_detail/<int:pk>/', StudentDetailView.as_view()),
path('construct_csv/', construct_csv),
path('construct_csv_from_model/', construct_csv_from_model),
path('construct_pdf/', construct_pdf),
path('construct_pdf_from_model/', construct_pdf_from_model),
path('expr/', expr),
path('cbox/', cbox),
path('course_search_ajax/', course_search_ajax),
```

Output:





Student Name	Student USN	Sem
Sham	4JN22AI001	4
Manish	4JN22AI002	4