1) Develop a Django app that displays current date and time in server

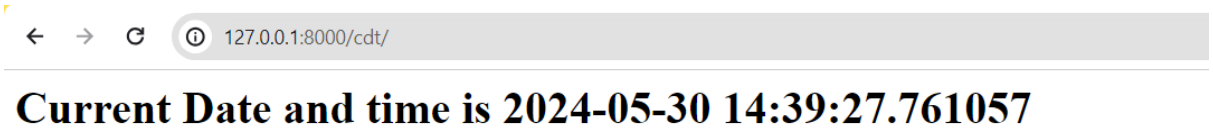
views.py

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

# Create your views here.
def current_date_time(request):
    now=datetime.datetime.now()
    result="<html><body><h1>Current Date and time is %s" %(now)
    return HttpResponse(result)
```

urls.py

```
from django.contrib import admin
from django.urls import path
from ap1.views import current_date_time,four_hours_ahead,four_hours_before
from ap2.views import showlist
urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time ),
```

OUTPUT:



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.

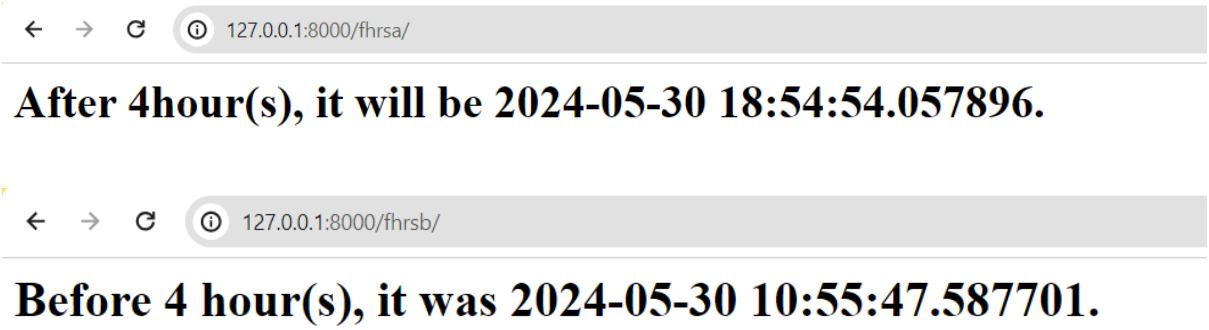
Views.py

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

# Create your views here.
def current_date_time(request):
    now=datetime.datetime.now()
    result="<html><body><h1>Current Date and time
is %s" %(now)
    return HttpResponse(result)

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)
```

urls.py

```
from django.contrib import admin
from django.urls import path
from ap1.views import
current_date_time,four_hours_ahead,four_hours_befor
e

urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time ),y
    path('fhrsa/',four_hours_ahead),
    path('fhrsb/',four_hours_before),
]
```

OUTPUT:



3) Develop a simple Django app that displays an unordered list of fruits and ordered list of selected students for an event

Create an another ap folder:

```
python manage.py startapp ap2
```

Create a folder "templates" in ap2
Create a file "showlist.html" as mentioned below in templates folder

ap2/templates/showlist.html

```
<html>
    <style type="text/css">
        #i1 {background-color: lightgreen;color:brown;display:table}
        #i2 {background-color: black;color:yellow;display:table}
```

```
    </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>
```

views.py

```
from django.shortcuts import render

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

urls.py

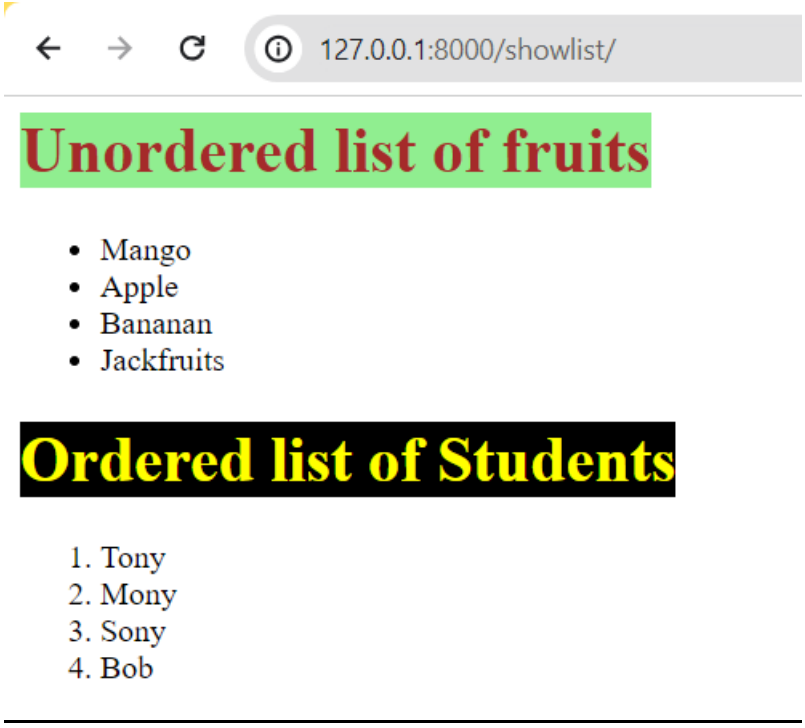
```
from django.contrib import admin
from django.urls import path
from ap1.views import current_date_time,four_hours_ahead,four_hours_before
from ap2.views import showlist
urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time ),
    path('fhrsa/',four_hours_ahead),
    path('fhrsb/',four_hours_before),
    path('showlist/', showlist),
]
```

Lab/settings.py

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,'ap2/templates'),],
```

```
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

OUTPUT:



4) 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 django.shortcuts import render


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.urls import path
from ap2.views import aboutus,home,contactus

urlpatterns = [
    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; Developed by Sir MVIT, Bengaluru
        </footer>
    </body>
</html>
```

home.html

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

aboutus.html

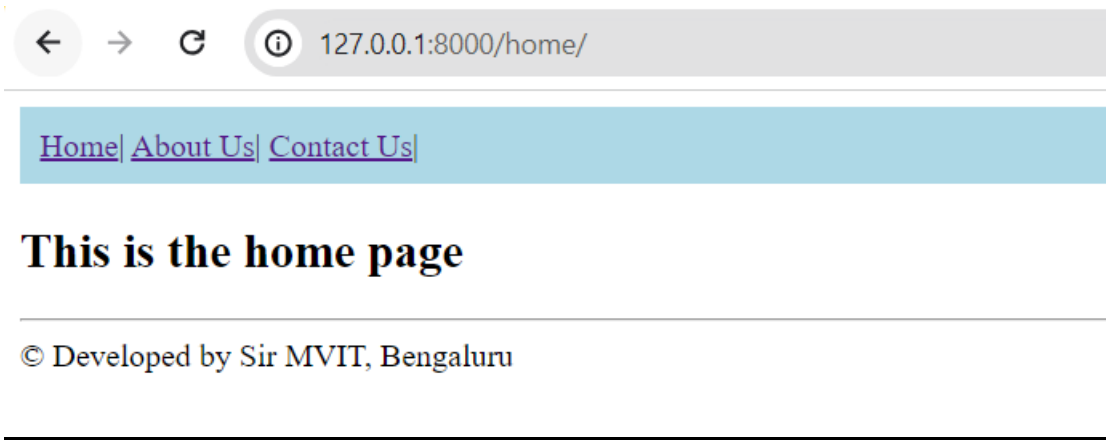
```
{% extends 'layout.html' %}
{% block title %}
About Us
{% endblock %}
{% block content %}
<h2>We are DJango developers</h2>
{% endblock %}
```

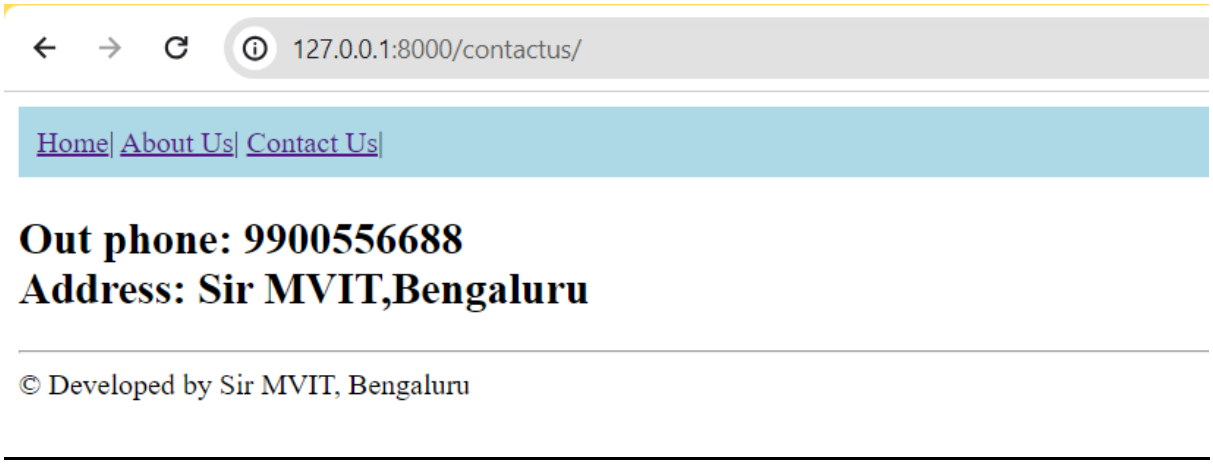
contactus.html

```
{% extends 'layout.html' %}
{% block title %}
Contact us
{% endblock %}
{% block content %}
<h2> phone: 9900556688 <br> Address: Sir
MVIT,Bengaluru</h2>
```

```
{% endblock %}
```

OUTPUT:




127.0.0.1:8000/contactus/
Home| About Us| Contact Us|
Out phone: 9900556688
Address: Sir MVIT,Bengaluru
© Developed by Sir MVIT, Bengaluru

5) 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.

WAMP Server link

https://sourceforge.net/projects/wampserver/files/latest/download

During installation process, many files will be missing and system asks to install it. Hence download the files from this website:

https://wampserver.aviatechno.net/

After successful installation of WAMP server, start it and then go to http://localhost/phpmyadmin

Usename: root

Password- empty

Use phpMyAdmin

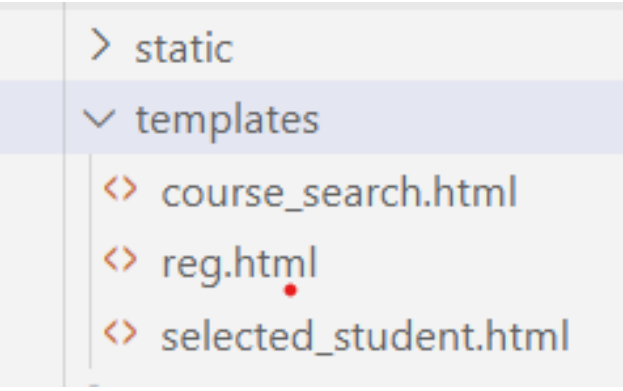
Create a new database "studentreg"

PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment> python manage.py startapp ap3

Install mysqlclient from VS Code terminal:

pip install mysqlclient

static and template folder creation and 3 files inside this folder



course_search.html

```
<html>
    <body>
        <form method="POST" action="">
            Courses
            {% csrf_token %}
            <select name="cname">
                {%for course in courses %}
                <option value="{{course.id}}">{{course.course_name}}</option>
                {% endfor %}
            </select>
            <input type="submit" value="Search">
        </form>
    </body>
</html>
```

reg.html

```
<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>
```

selected_student.html

```
<html>
    <body>
        <table border>
            <tr>
                <th>Student Name</th>
                <th>Student USN</th>
                <th>Sem</th>
            </tr>
            {% for student in student_list %}
            <tr>
                <td>{{student.student_name}}</td>
                <td>{{student.student_usn}}</td>
                <td>{{student.student_sem}}</td>
            </tr>
            {% endfor %}
        </table>
    </body>
</html>
```

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)
```

views.py

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

from ap3.models import Course, Student

# Create your views here.
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})

def course_search(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_student.html",{"student_list":student_list})
    else:
        courses=Course.objects.all()
        return render(request,"course_search.html",{"courses":courses})
```

urls.py

```
from ap3.views import reg, course_search

urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time ),
    path('fhrsa/',four_hours_ahead),
    path('fhrsb/',four_hours_before),
    path('showlist/', showlist),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    path('reg/', reg),
    path('course_search/',course_search),
]
```

settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'ap3'
]

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,'ap3/templates'),],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'studentreg',
        'USER': 'root',
        'PASSWORD': '',
        'HOST':'localhost',
        'PORT':'3306',
    }
}

STATIC_URL = 'static/'
STATICFILES_DIRS=[os.path.join(BASE_DIR, 'ap3/static')]
```

Perform Migrations

python manage.py makemigrations ap3

python manage.py migrate

python manage.py runserver

***Note: migration should be done every time as models.py change or any database table changes*.**

Insert into tables in phpmyadmin

http://localhost/phpmyadmin

OR

http://localhost/phpmyadmin/index.php?route=/sql&pos=0&db=studentreg&table=ap3_course

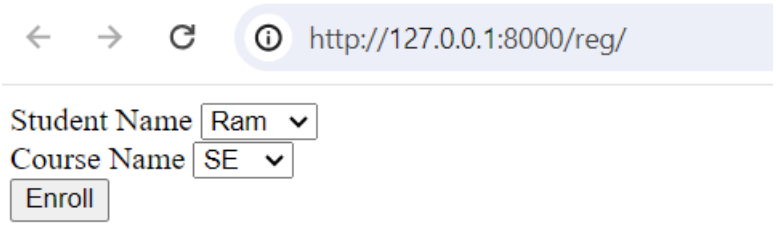
http://localhost/phpmyadmin/index.php?route=/sql&pos=0&db=studentreg&table=ap3_student

ap3_student and ap3_course

OUTPUT

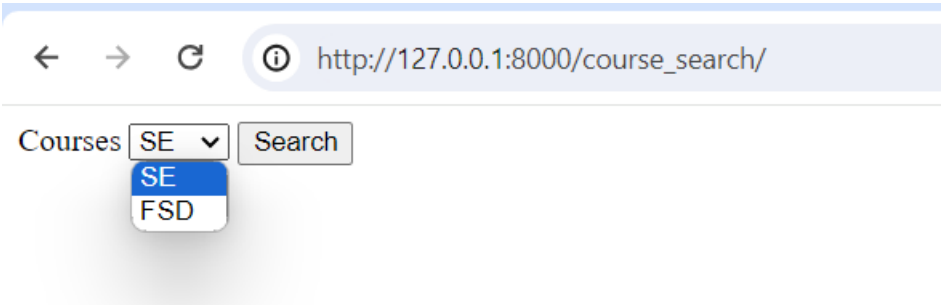
Run the url

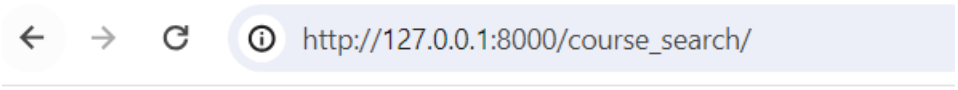
http://127.0.0.1:8000/reg/



---

http://127.0.0.1:8000/course_search/





Student Name	Student USN	Sem
Ram	100	6
Sham	101	6
Seeta	102	6

Module 3:

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

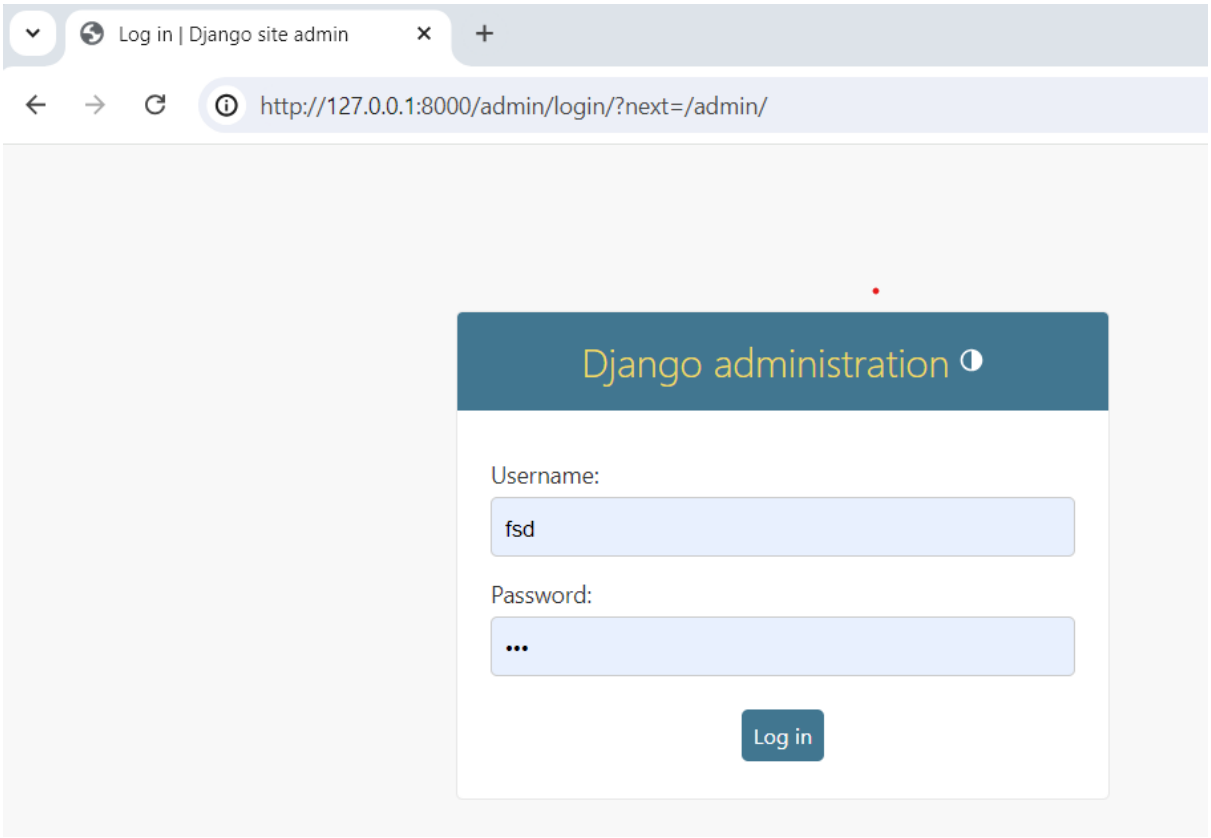
1)admin.py

```
from django.contrib import admin
from ap3.models import Course, Student
# Register your models here.
admin.site.register(Student)
admin.site.register(Course)
```

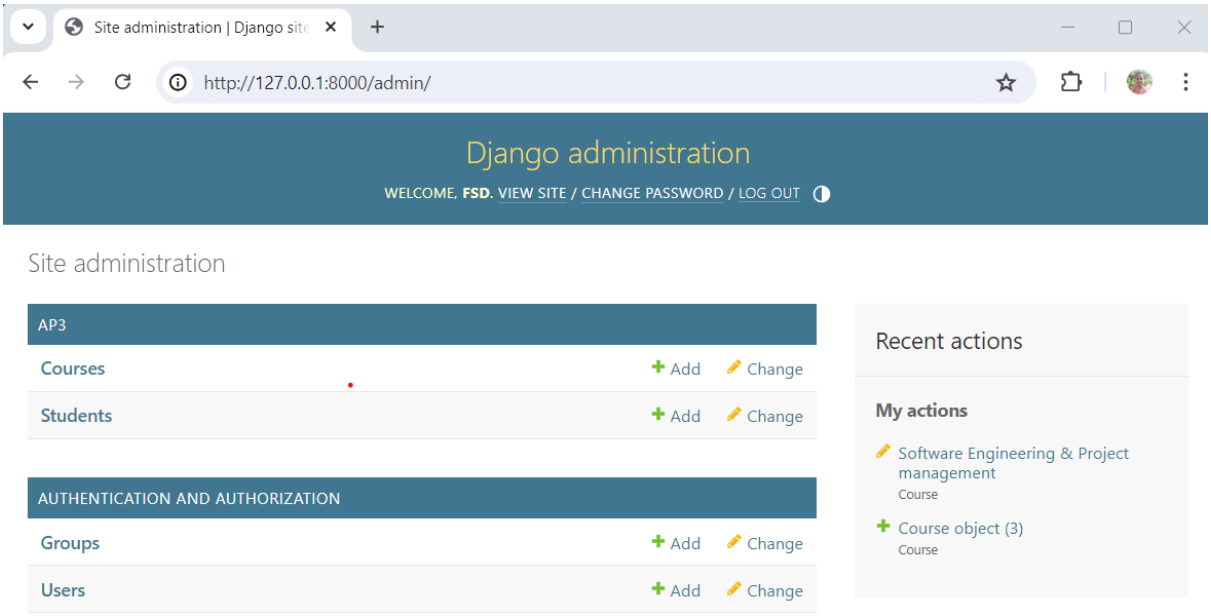
```
PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment>
PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment> python manage.py createsuperuser
Username (leave blank to use 'savitacb'): fsd
Email address: fsd@sirmvit.edu
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment>
```

PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment> python manage.py runserver

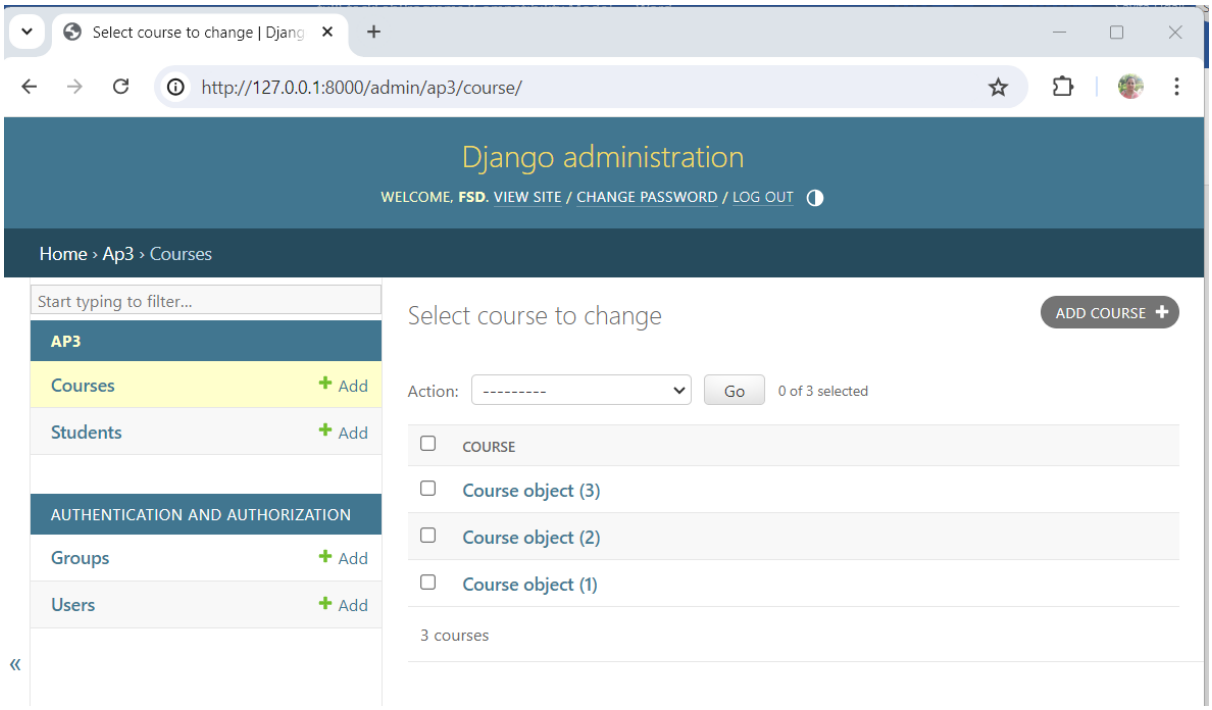
➔ Login with your superuser credentials



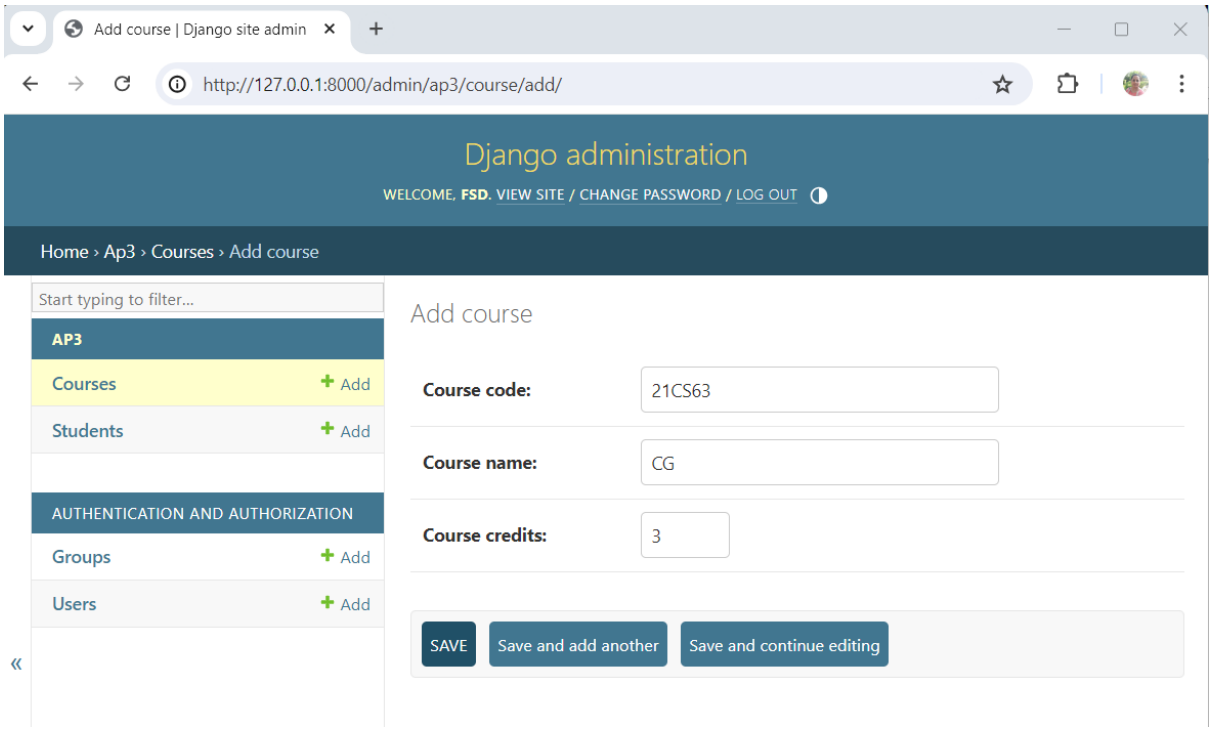
➔ Index page should be visible

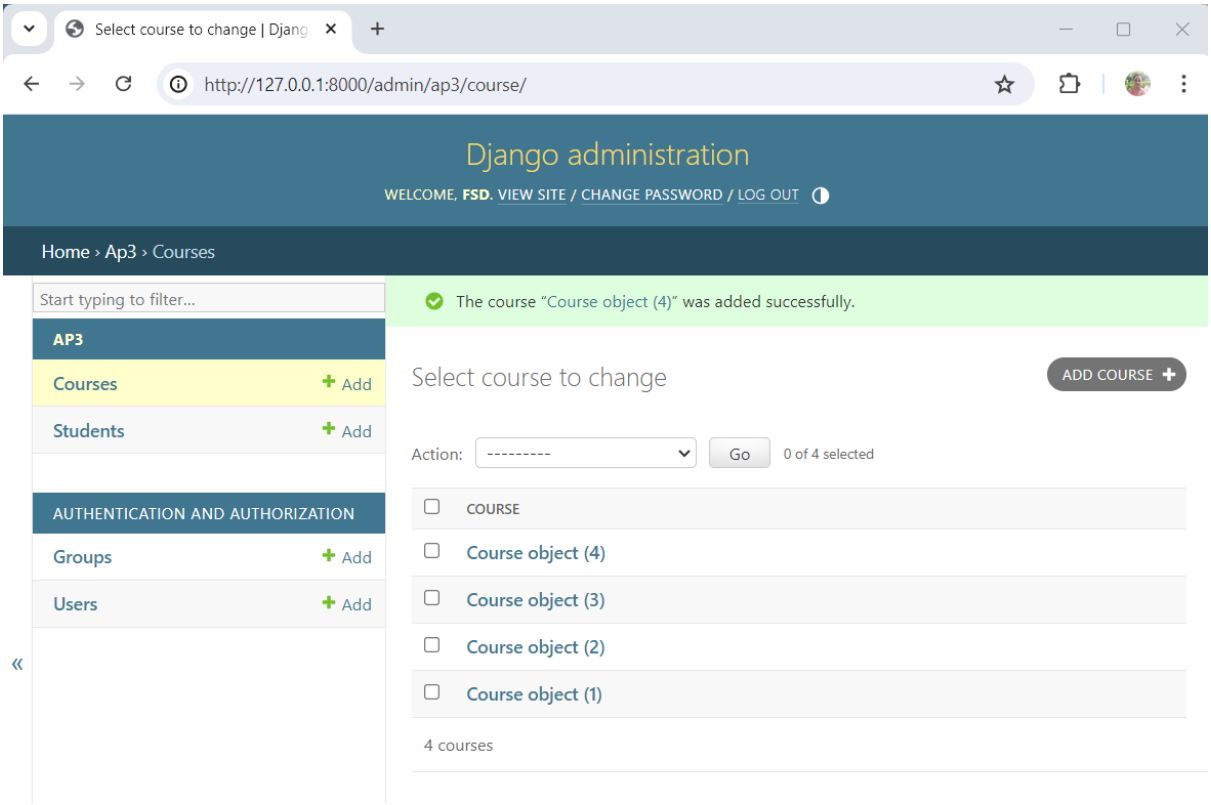


➔ Go to Courses



➔ Add course





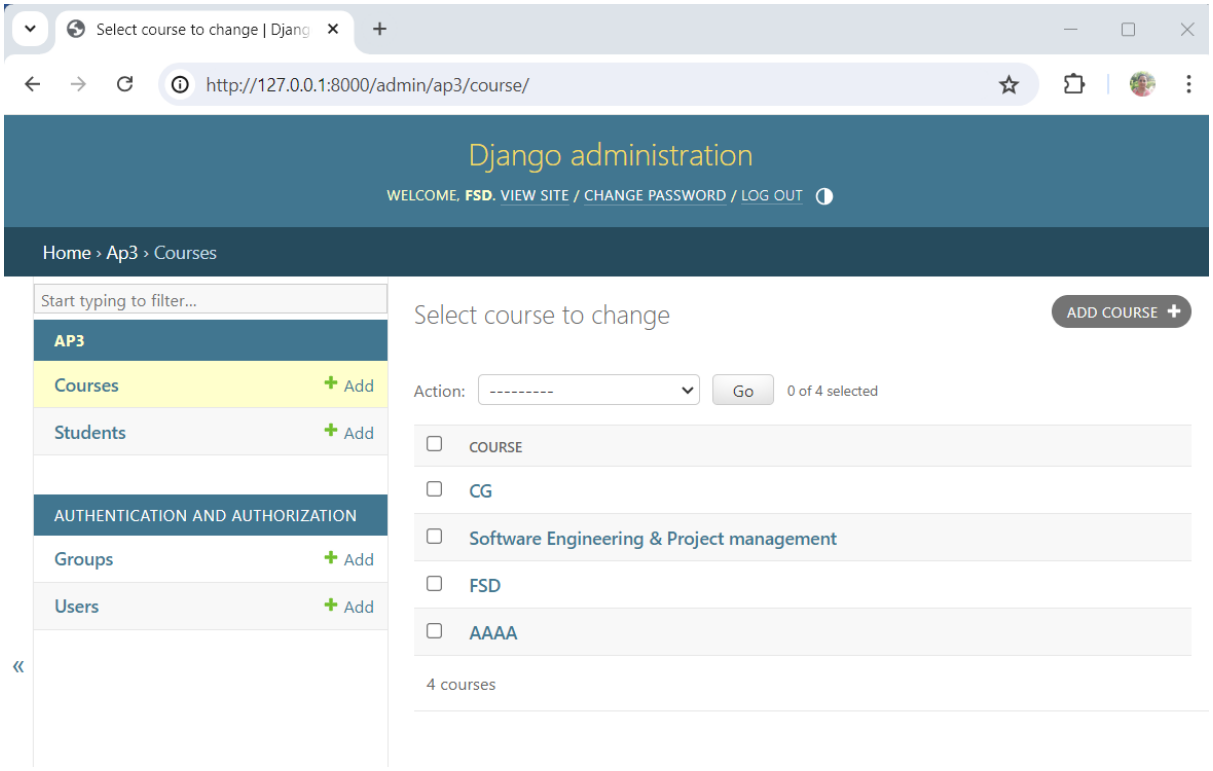
➔ We see a very vague display of objects. To fix it, make the following changes in models.py

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()
    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+")"
```

Run the code:

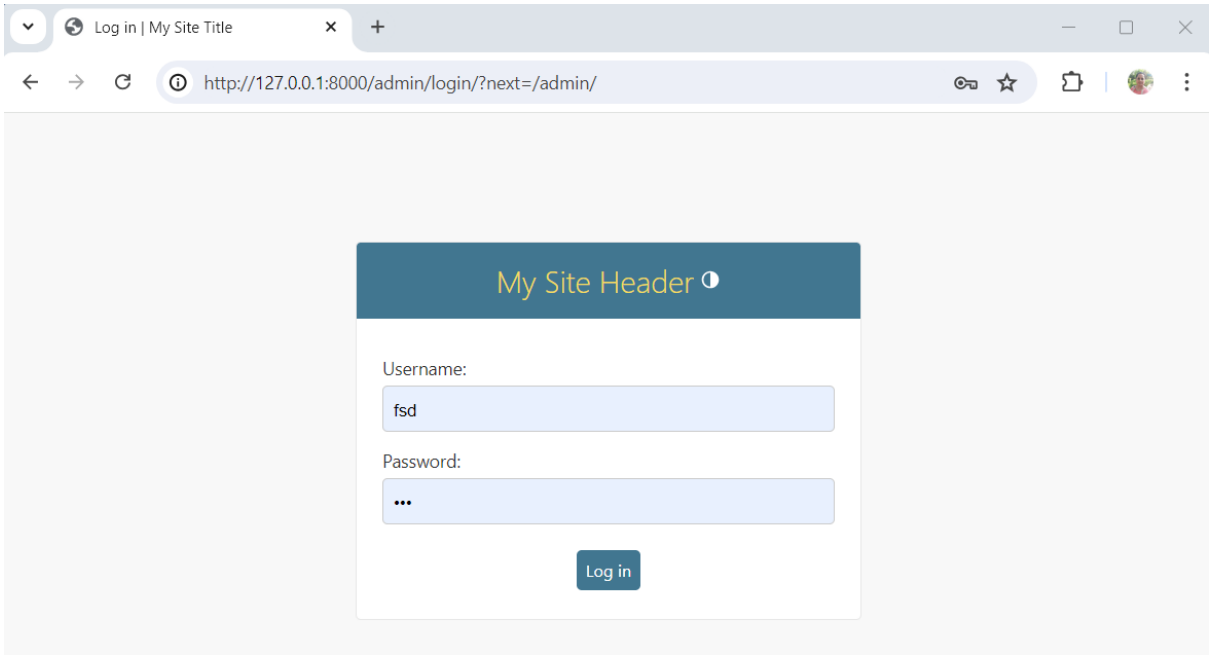
```
PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment> python manage.py runserver
```

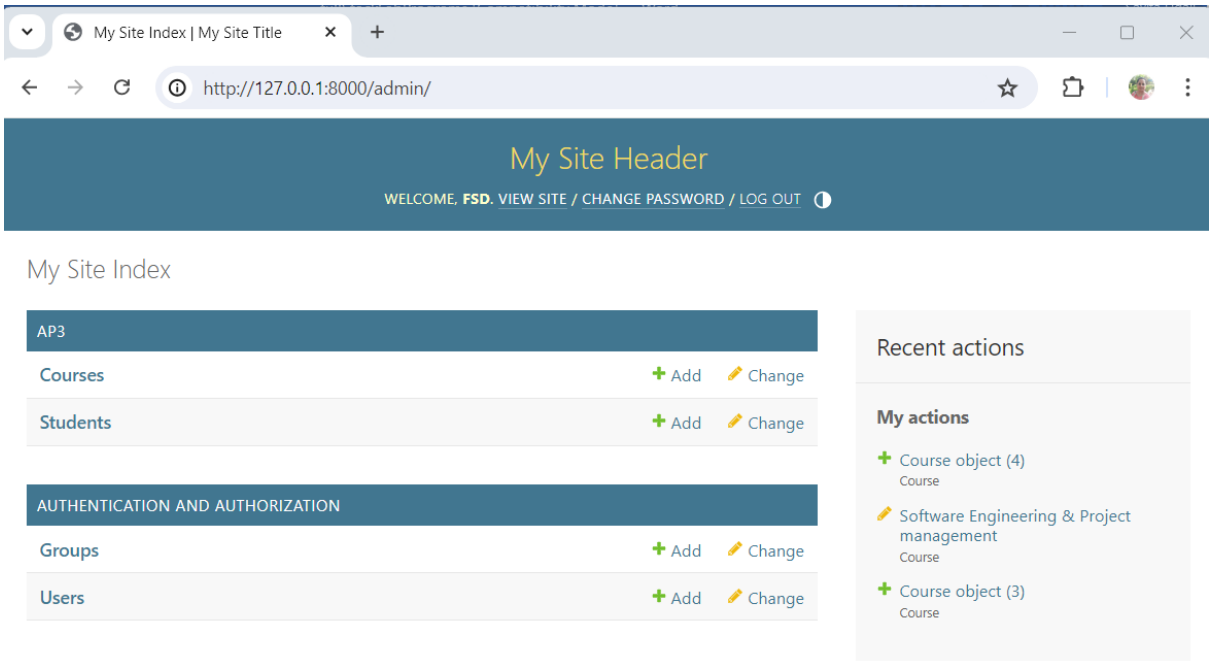


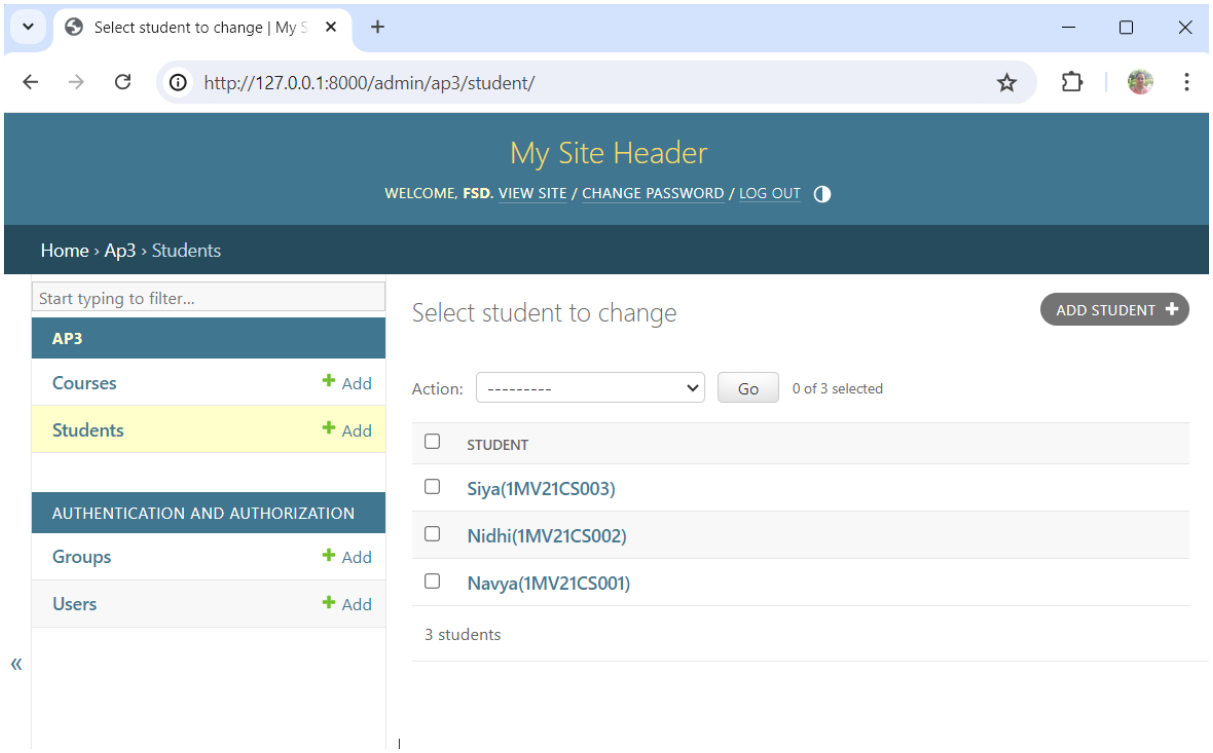
Updating urls.py

```
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"
```

PS D:\SirMVIT\MY_SUBJECTS\FullStackDevelopment> python manage.py runserver


Log in | My Site Title
http://127.0.0.1:8000/admin/login/?next=/admin/
My Site Header
Username:
fsd
Password:
Log in


My Site Index | My Site Title
http://127.0.0.1:8000/admin/
My Site Header
WELCOME, FSD. VIEW SITE / CHANGE PASSWORD / LOG OUT
My Site Index
AP3
Courses
Add
Change
Students
Add
Change
AUTHENTICATION AND AUTHORIZATION
Groups
Add
Change
Users
Add
Change
Recent actions
My actions
Course object (4)
Course
Software Engineering & Project management
Course
Course object (3)
Course



Updating models.py (removing compulsory field)
In course

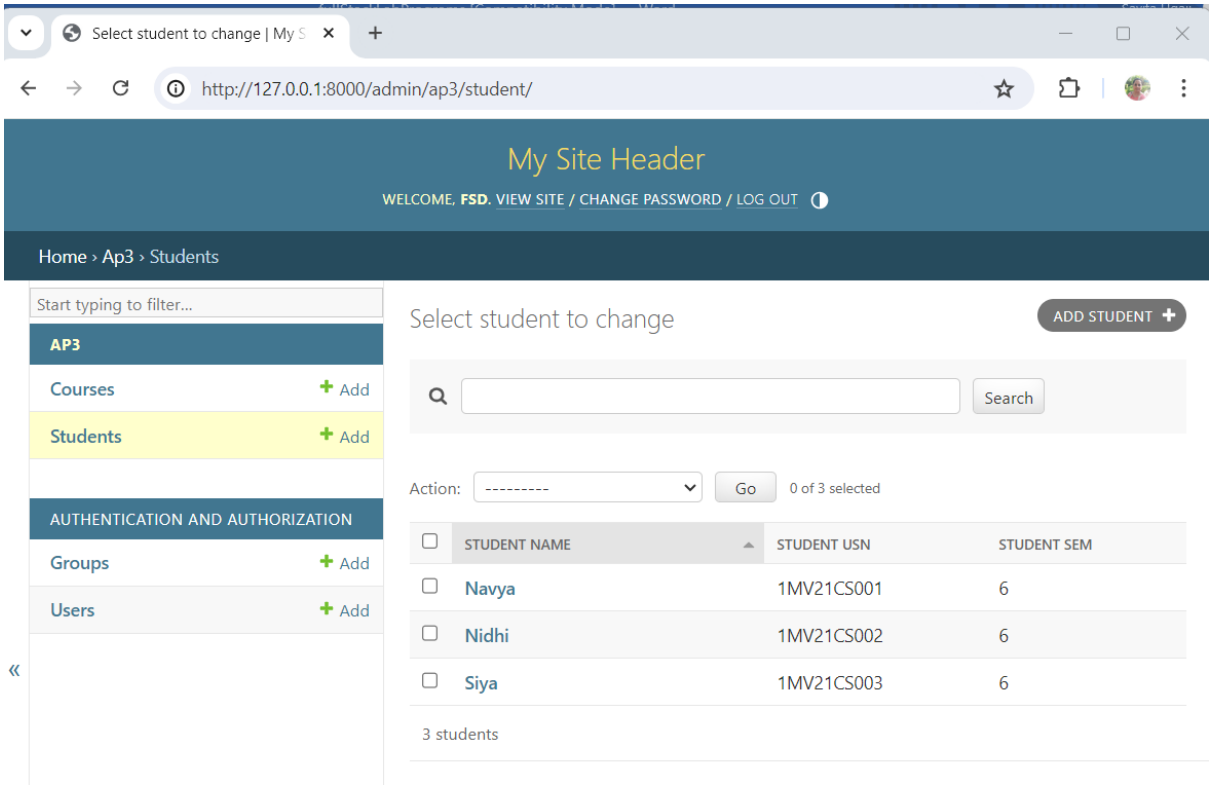
```
course_credits=models.IntegerField(blank=True, null=True)
```

and run the migration commands

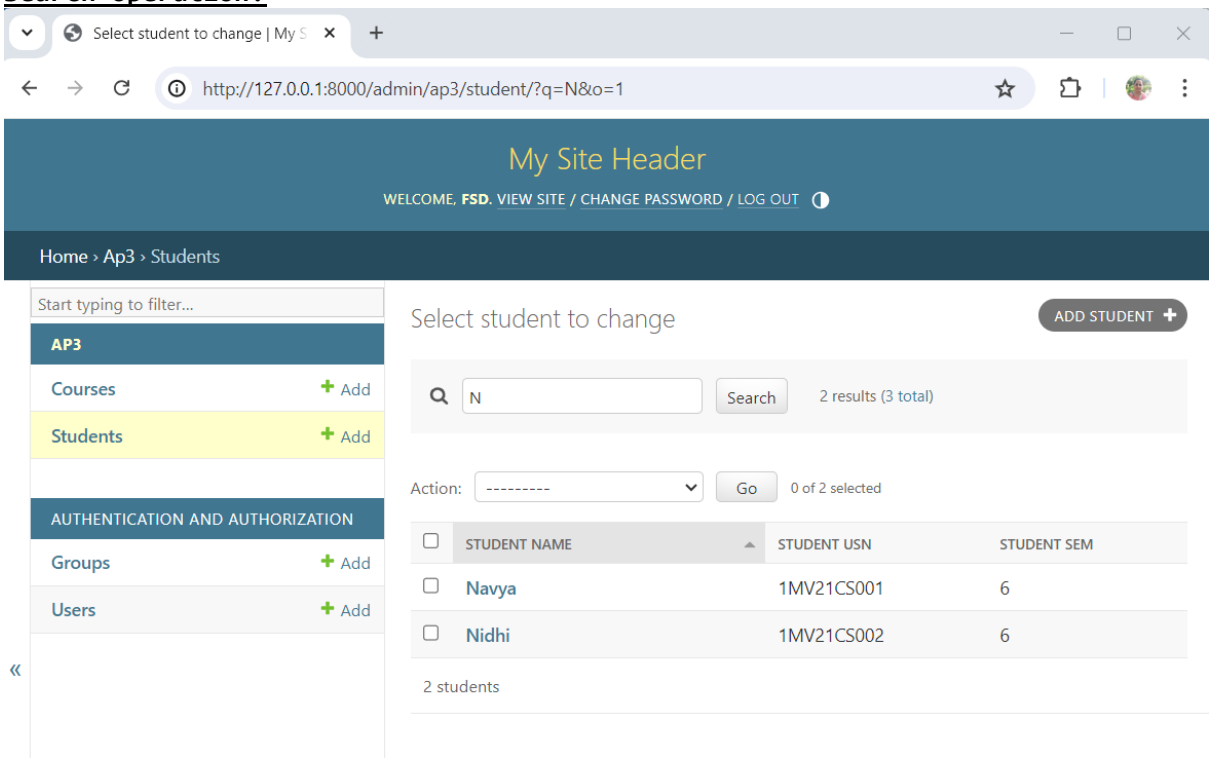
```
python manage.py makemigrations ap3
python manage.py migrate
```

customizing admin
in admin.py

```
#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',)
```



Search Operation:



7) 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

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 ap3.models import Course, Student, ProjectReg

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 (need to be created in 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>
```

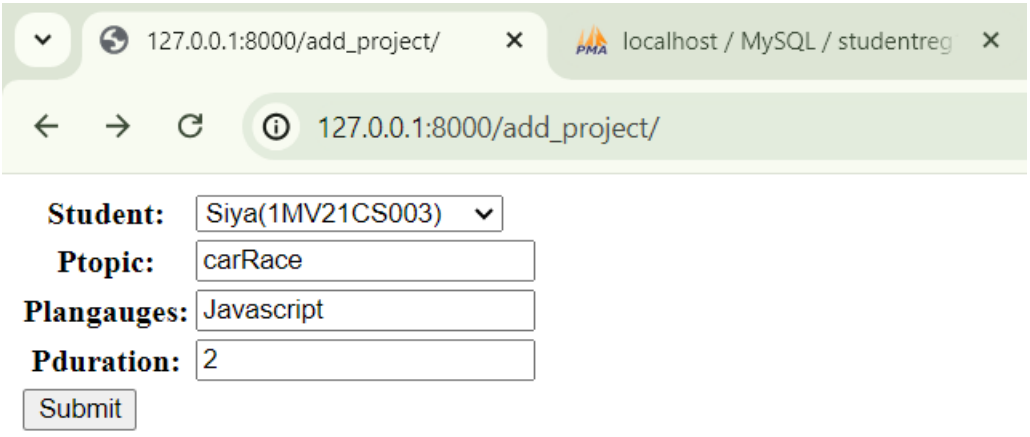
urls.py

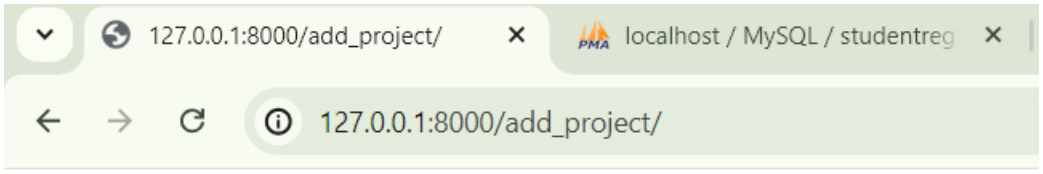
```
from ap3.views import reg, course_search, add_project


urlpatterns = [
    path('admin/', admin.site.urls),
    path('cdt/', current_date_time ),
    path('fhrsa/',four_hours_ahead),
    path('fhrsb/',four_hours_before),
    path('showlist/', showlist),
    path('aboutus/', aboutus),
    path('home/', home),
    path('contactus/', contactus),
    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

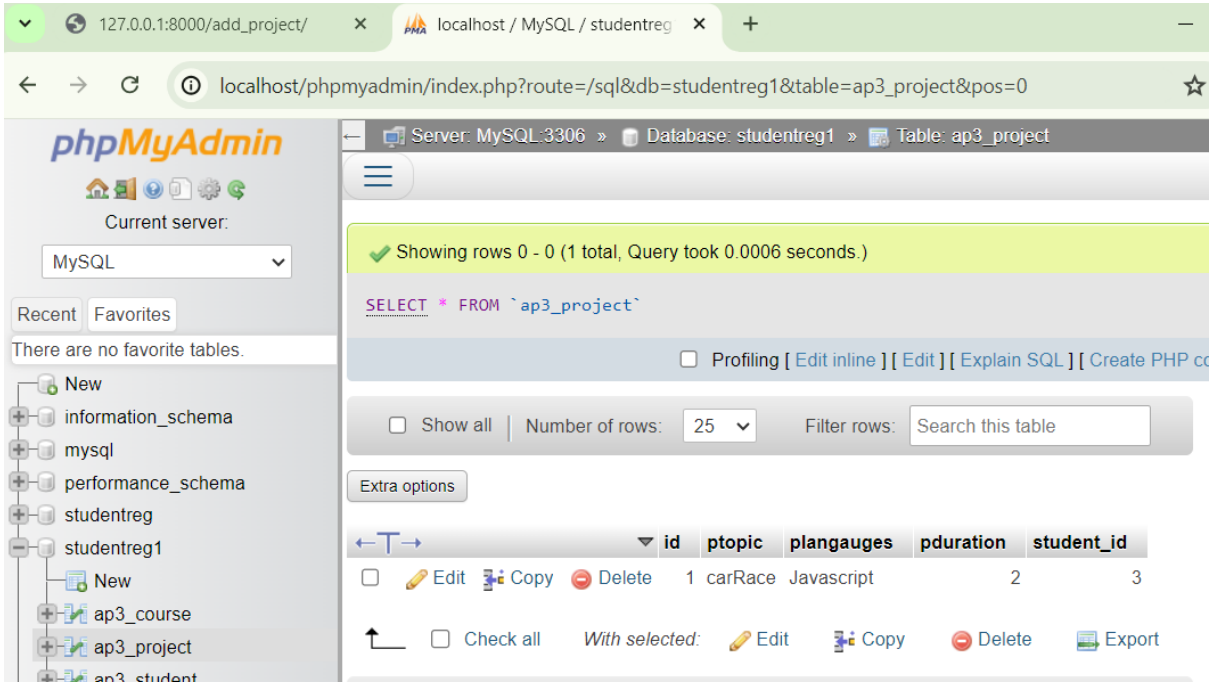
python manage.py runserver

OUTPUT:





Record inserted successfully



8) 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.

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

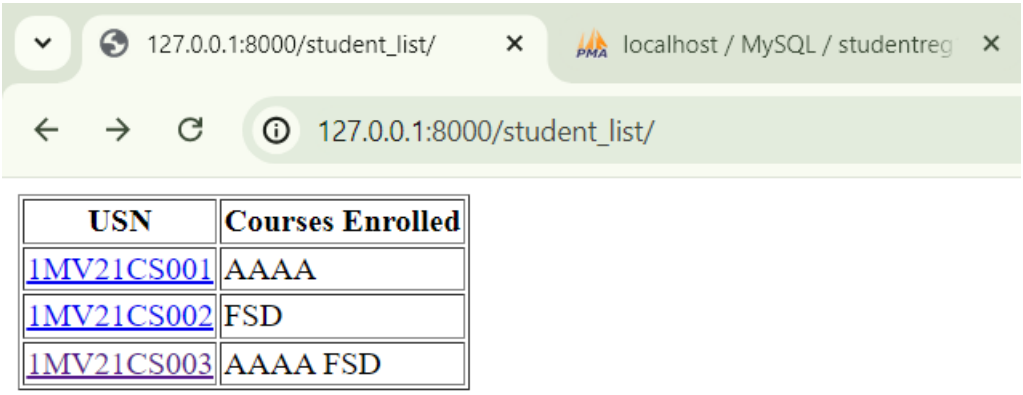
```
<html>
    <body>
        {% if student_list %}
        <table border>
            <tr>
                <th>USN</th>
                <th>Courses Enrolled</th>
            </tr>
            {% for student in student_list %}
            <tr>
                <td><a href="/student_detail/{{student.pk}}">{{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

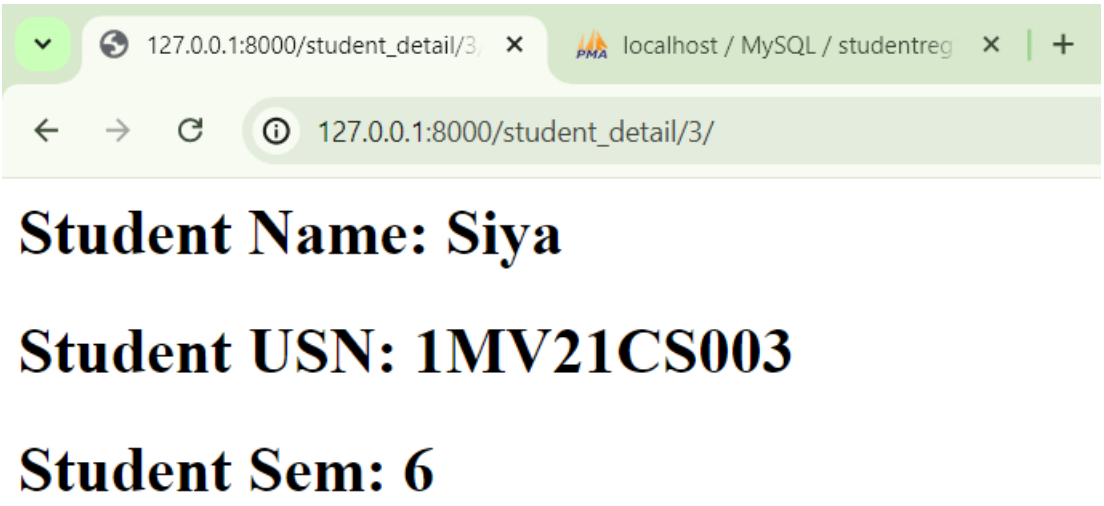
student_detail.html

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

urls.py

```
from ap3.views import StudentListView,StudentDetailView
urlpatterns = [
    path('add_project/', add_project),
    path('student_list/', StudentListView.as_view()),
    path('student_detail/<int:pk>/', StudentDetailView.as_view()),
]
```





9) Develop example Django app that performs CSV and PDF generation for any models created in previous laboratory component.

In the terminal:
pip install reportlab

views.py

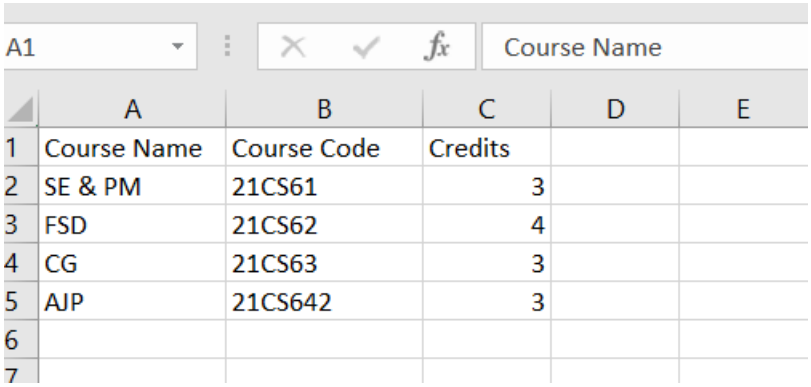
```
def construct_csv_from_model(request):
    courses=Course.objects.all()
    response=HttpResponse(content_type="text/csv")
    response['Content-Disposition'] = 'attachment;filename="courses_data.csv"'
    writer=csv.writer(response)
    writer.writerow(["Course Name","Course Code","Credits"])
 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

def construct_pdf_from_model2(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 ap3.views import construct_csv_from_model, construct_pdf_from_model2


urlpatterns = [
    path('construct_course/', construct_csv_from_model),
    path('construct_pdf_from_model2/', construct_pdf_from_model2),
]
```

CSV file downloaded



A1 | Course Name

	A	B	C	D	E
1	Course Name	Course Code	Credits		
2	SE & PM	21CS61	3		
3	FSD	21CS62	4		
4	CG	21CS63	3		
5	AJP	21CS642	3		
6					
7					

PDF file downloaded

Course Name	Course Code	Credits
SE & PM	21CS61	3
FSD	21CS62	4
CG	21CS63	3
AJP	21CS642	3

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

views.py

```
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

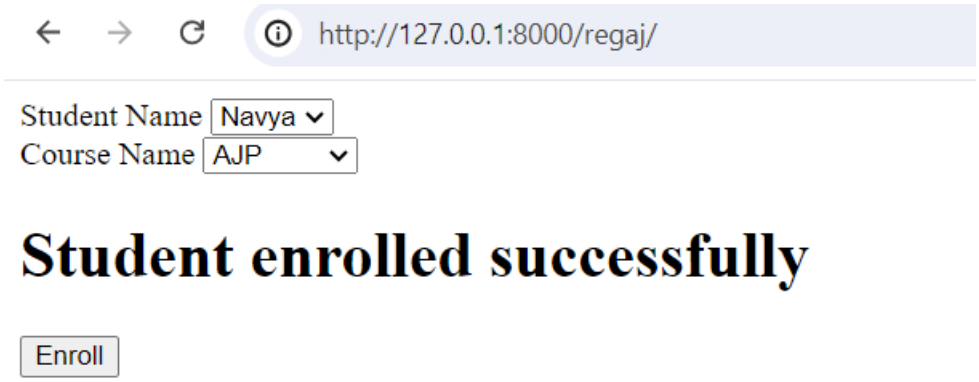
```
{% load static %}
<html>
    <body>
        <form method="post" action="">
            {% csrf_token %}
            Student Name
            <select name="sname" id="sname">
                {% for student in students %}
                <option value="{{ student.id }}">{{ student.student_name
}}</option>
                {% 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 ap3.views import regaj

urlpatterns = [
    path('regaj/',regaj),
]
```

OUTPUT



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

views.py

```
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();
            $.ajax({
                url: "/course_search/",
                type: "POST",
                data: { cname: cname, csrfmiddlewaretoken: "{{ csrf_token
}}"},
                success: function(response){$("#result").html(response);}
            });
```

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

urls.py

```
from ap3.views import regaj, course_search_ajax
urlpatterns = [
    path('course_seacrh_ajax/',course_search_ajax),
]
```

OUTPUT