

# FullStack Development Lab Manual

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

urls.py (url file inside the app)

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

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

Views.py (view file inside the app)

```
from django.shortcuts import render
from django.http import HttpResponse
from django.utils.timezone import now, timedelta

# Create your views here.

def current_datetime(request):
    now_time = now()
    return HttpResponse(f"Current date and time: {now_time}")
```

urls.py(url file inside the mainproject)

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('myapp1.urls')),
]
```

## 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

urls.py(url file inside the app)

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

urlpatterns = [
    path('time/plus/<int:hours>', views.hours_ahead),
    path('time/minus/<int:hours>', views.hours_before),
]
```

views.py(view file inside the app)

```
from django.shortcuts import render
from django.http import HttpResponse
from django.utils.timezone import now, timedelta

# Create your views here.
```

```
def hours_ahead(request, hours):
    offset_time = now() + timedelta(hours=hours)
    return HttpResponse(f"time {hours} hours ahead: {offset_time}")

def hours_before(request, hours):
    offset_time = now() + timedelta(hours=-hours)
    return HttpResponse(f"time {hours} hours ahead: {offset_time}")
```

urls.py(url file inside the mainproject)

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

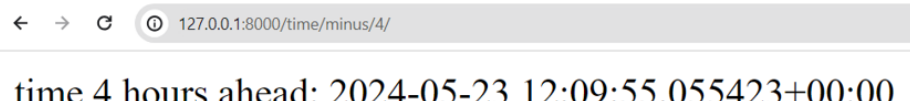
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('myapp1.urls')),
]
```

### Output



← → ↻ ⓘ 127.0.0.1:8000/time/plus/2/

time 2 hours ahead: 2024-05-23 18:07:40.002096+00:00



← → ↻ ⓘ 127.0.0.1:8000/time/minus/4/

time 4 hours ahead: 2024-05-23 12:09:55.055423+00:00

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

### **Urls.py(myapp)**

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

urlpatterns = [
    path('',views.showlist)
]
```

### **Views.py(myapp)**

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

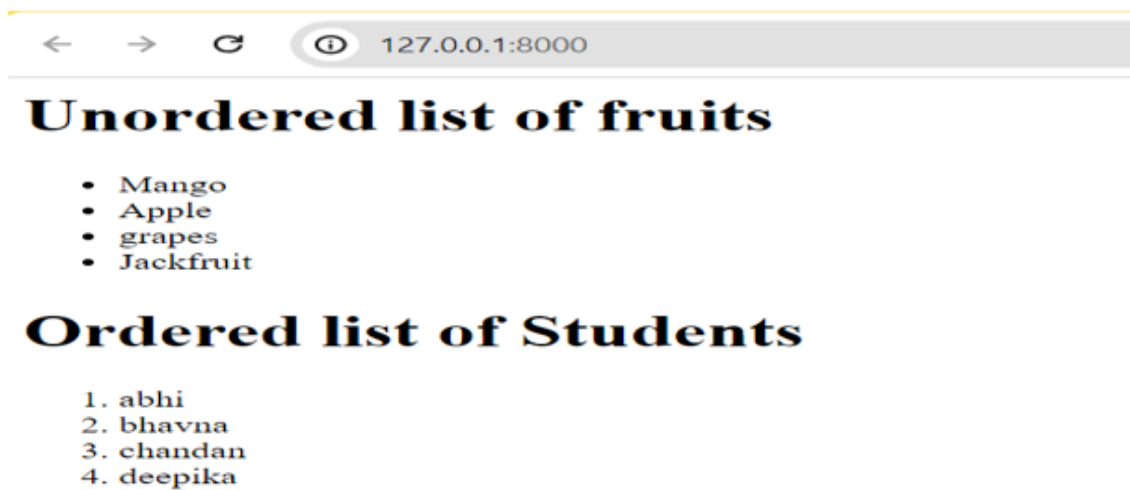
# Create your views here.
def showlist(request):
    fruits=["Mango","Apple","grapes","Jackfruit"]
    student_names=["abhi","bhavna","chandan","deepika"]
    context={"fruits":fruits,"student_names":student_names}
    return render(request,'showlist.html',context)
```

### **Showlist.html(template)**

```
<html>
<body>
<h1>Unordered list of fruits</h1>
<ul>
{% for fruit in fruits %}
<li>{{ fruit }}</li>
{% endfor %}
</ul>
```

```
<h1>Ordered list of Students</h1>
<ol>
{% for student in student_names %}
<li>{{ student }}</li>
{% endfor %}
</ol>
</body>
</html>
```

## 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.

Urls.py(myapp)

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

urlpatterns = [

    path('about/', views.about),
    path('home/', views.home),
    path('contact/', views.contact)

]
```

Views.py(myapp)

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

# Create your views here.
def home(request):
    return render(request, 'home.html')
def about(request):
    return render(request, 'about.html')
def contact(request):
    return render(request, 'contact.html')
```

## html files in template folder

about.html

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

{% block content %}
<h2>we are Django developers</h2>
{% endblock %}
```

Contact.html

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

{% block content %}
<h2>contact us mycem.edu.in</h2>
{% endblock %}
```

Home.html

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

{% block content %}
<h2>This is the home page</h2>
{% endblock %}
```

Layout.html

```
<html>
<body>
<nav>
<a href="/home/">Home</a>|
<a href="/about/">About Us</a>|
<a href="/contact/">Contact Us</a>|
</nav>

{% block content %}

{% endblock %}

<footer>
<hr>
&copy; 2024 Mycem Developed by digital india
</footer>
</body>
</html>
```

## OUTPUT

[Home](#)| [About Us](#)| [Contact Us](#)|

**we are Django developers**

---

© 2024 Mycem Developed by digital india

[Home](#)| [About Us](#)| [Contact Us](#)|

**This is the home page**

---

© 2024 Mycem Developed by digital india

[Home](#)| [About Us](#)| [Contact Us](#)|

**contact us mycem.edu.in**

---

© 2024 Mycem Developed by digital india



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.

### Create tables in mysql db

```
CREATE TABLE student (  
    usn INT PRIMARY KEY,  
    name VARCHAR(50) NOT NULL,  
    branch VARCHAR(20) NOT NULL,  
    phone BIGINT NOT NULL,  
    email VARCHAR(20) NOT NULL  
);
```

```
CREATE TABLE course (  
    cid INT PRIMARY KEY,  
    cname VARCHAR(50) NOT NULL  
);
```

```
CREATE TABLE enrollment (  
    eid INT PRIMARY KEY,  
    sid INT,  
    cid INT,  
    FOREIGN KEY (sid) REFERENCES student(usn) ON DELETE CASCADE,  
    FOREIGN KEY (cid) REFERENCES course(cid) ON DELETE CASCADE  
);
```

```
CREATE TABLE course_student (  
    course_id INT NOT NULL,  
    student_id INT NOT NULL,  
    FOREIGN KEY (course_id) REFERENCES course(cid) ON DELETE CASCADE,  
    FOREIGN KEY (student_id) REFERENCES student(usn) ON DELETE CASCADE  
);
```

## Settings.py(myproject)

```
DATABASES = {  
    'default':{  
        'ENGINE':'django.db.backends.mysql',  
        'NAME':'jangoDB',  
        'USER':'root',  
        'PASSWORD':'WWW@deep123',  
        'HOST':'localhost',  
        'PORT':'3306'  
    }  
}
```

## MODELS.PY

```
from django.db import models  
  
# Create your models here.  
  
class student(models.Model):  
    usn = models.IntegerField(primary_key=True)  
    name = models.CharField(max_length=50)  
    branch = models.CharField(max_length=20)  
    phone = models.BigIntegerField()  
    email = models.EmailField(max_length=20)
```

```
class Meta:
    db_table = "student"

def __str__(self):
    return self.name

class course(models.Model):
    cid = models.IntegerField(primary_key=True)
    cname = models.CharField(max_length=50)
    student = models.ManyToManyField('student',
through='Enrollment')

def __str__(self):
    return self.cname

class Meta:
    db_table = "course"

class Enrollment(models.Model):
    eid = models.IntegerField(primary_key=True)
    sid = models.ForeignKey(student, on_delete=models.CASCADE)
    cid = models.ForeignKey(course, on_delete=models.CASCADE)

class Meta:
    db_table = "enrollment"
```

## forms.py

```
from django import forms
from .models import student, Enrollment

class StudentForm(forms.ModelForm):
    class Meta:
        model = student
        fields = ['usn', 'name', 'branch', 'phone', 'email']

class EnrollmentForm(forms.ModelForm):
    class Meta:
        model = Enrollment
        fields = ['eid', 'sid', 'cid']
```

## urls.py(myapp)

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

urlpatterns = [
    path('students/register/', views.student_registration,
        name='student-registration'),
    path('students/enroll/', views.enroll_student, name='enroll-
student'),
    path('courses/', views.course_list, name='course-list'),
    path('courses/<int:course_id>', views.students_in_course,
        name='students-in-course'),
]
```

## Views.py

```
from django.shortcuts import render, get_object_or_404, redirect
from .models import student, course, Enrollment
from .forms import StudentForm, EnrollmentForm

def student_registration(request):
    if request.method == 'POST':
        form = StudentForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('course-list')
    else:
        form = StudentForm()
    return render(request, 'student_form.html', {'form': form})

def enroll_student(request):
    if request.method == 'POST':
        form = EnrollmentForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('course-list')
    else:
        form = EnrollmentForm()
    return render(request, 'enrollment_form.html', {'form': form})

def course_list(request):
    courses = course.objects.all()
    return render(request, 'course_list.html', {'courses': courses})

def students_in_course(request, course_id):
    selected_course = get_object_or_404(course, cid=course_id)
    enrollments = Enrollment.objects.filter(cid=selected_course)
    students = [enrollment.sid for enrollment in enrollments]
    return render(request, 'students_in_course.html', {'course':
selected_course, 'students': students})
```

## TEMPLATES

### student\_form.html

```
<html>
<head>
  <title>Student Registration</title>
</head>
<body>
  <h1>Register a Student</h1>
  <form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Register</button>
  </form>
</body>
</html>
```

### Course\_list.html

```
<html>
<head>
  <title>Courses</title>
</head>
<body>
  <h1>Courses</h1>
  <ul>
    {% for course in courses %}
      <li><a href="{% url 'students-in-course' course.cid %}">{{
course.cname }}</a></li>
    {% endfor %}
  </ul>
</body>
</html>
```

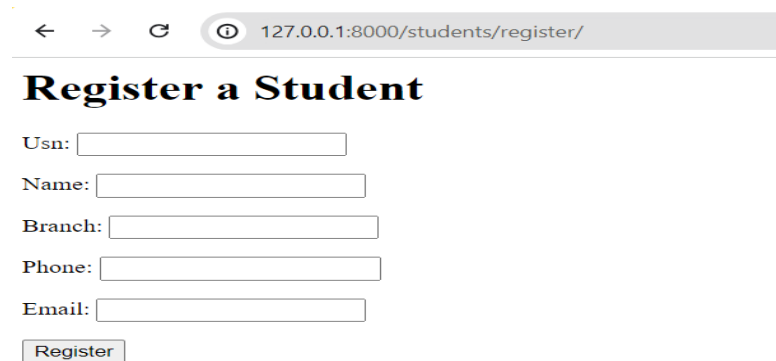
## Enrollment\_form.html

```
<html>
<head>
  <title>Enroll Student</title>
</head>
<body>
  <h1>Enroll a Student in a Course</h1>
  <form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Enroll</button>
  </form>
</body>
</html>
```

## Student\_in\_course.html

```
<html>
<body>
  <h1>Students in {{ course.cname }}</h1>
  <ul>
    {% for student in students %}
      <li>{{ student.name }} ({{ student.usn }})</li>
    {% endfor %}
  </ul>
  <a href="{% url 'course-list' %}">Back to Courses</a>
</body>
</html>
```

## OUTPUT



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/students/register/". The page title is "Register a Student". The form contains five input fields: "Usn:", "Name:", "Branch:", "Phone:", and "Email:". Below these fields is a "Register" button.

## Enroll a Student in a Course

Eid:

Sid:  ▼

Cid:  ▼

Enroll

## Courses

- [java full stack](#)
- [data Analytics](#)
- [.net](#)
- [AWS](#)
- [android development](#)

## Students in .net

- chaitra (3)
- deepika (89)
- madurya (22)

[Back to Courses](#)



6. 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

class student(models.Model):
    usn = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=50)
    branch = models.CharField(max_length=20)
    phone = models.BigIntegerField()
    email = models.EmailField(max_length=20)

    class Meta:
        db_table = "student"

    def __str__(self):
        return self.name

class course(models.Model):
    cid = models.IntegerField(primary_key=True)
    cname = models.CharField(max_length=50)
    student = models.ManyToManyField('student', through='Enrollment')

    def __str__(self):
        return self.cname

    class Meta:
        db_table = "course"

class Enrollment(models.Model):
    eid = models.IntegerField(primary_key=True)
    sid = models.ForeignKey(student, on_delete=models.CASCADE)
    cid = models.ForeignKey(course, on_delete=models.CASCADE)

    class Meta:
        db_table = "enrollment"

class Project(models.Model):
    student = models.ForeignKey(student, on_delete=models.CASCADE)
    topic = models.CharField(max_length=100)
    languages = models.CharField(max_length=200)
```

```
duration = models.CharField(max_length=50)

class Meta:
    db_table = "project"

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

### forms.py

```
from django import forms
from .models import student, Enrollment, Project

class StudentForm(forms.ModelForm):
    class Meta:
        model = student
        fields = ['usn', 'name', 'branch', 'phone', 'email']

class ProjectForm(forms.ModelForm):
    class Meta:
        model = Project
        fields = ['student', 'topic', 'languages', 'duration']
```

### urls.py

```
urlpatterns = [

    path('projects/register/', views.project_registration, name='project-
registration'),
    path('plist/', views.project_list, name='project-list'),

]
```

## Views.py

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

def project_registration(request):
    if request.method == 'POST':
        form = ProjectForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('course-list')
    else:
        form = ProjectForm()
    return render(request, 'project_form.html', {'form': form})

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

## Templates

### Project\_form.html

```
<html>
<head>
    <title>Project Registration</title>
</head>
<body>
    <h1>Register a Project</h1>
    <form method="post">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Register</button>
    </form>
</body>
</html>
```

## Project\_list.html

```
<html>
<body>
  <h1>projects</h1>
  <ul>
    {% for p in projects %}
      <li>{{ p.student_id }}</li>
      <li>{{ p.topic }}</li>
      <li>{{ p.languages }}</li>
      <li>{{ p.duration }}</li>

    {% endfor %}
  </ul>
</body>
</html>
```

## OUTPUT

← → ↻ ⓘ 127.0.0.1:8000/projects/register/

---

# Register a Project

Student:  ▼

Topic:

Languages:

Duration:

← → ↻ ⓘ 127.0.0.1:8000/plist/

---

# projects

- 8
- abcd
- python
- 1month

-----

- 11
- plant management
- c# and f#
- 1month

-----

7. 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.

## Models.py

```
from django.db import models

# Create your models here.
class student(models.Model):
    usn = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=50)
    branch = models.CharField(max_length=20)
    phone = models.BigIntegerField()
    email = models.EmailField(max_length=20)

    def __str__(self):
        return self.name

    class Meta:
        db_table = "student"
```

## Views.py

```
from . models import student
from django.http import HttpResponse
from django.urls import reverse_lazy
from django.views.generic import ListView,DetailView

class studentListView(ListView):
    model = student
    template_name = 'std_list.html'
    context_object_name = 'student'
    queryset = student.objects.all().order_by('-name')

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

## urls.py

```
from django.urls import path
from . import views
from .views import TemplateView, StudentDetailView

urlpatterns = [

    path('stdlist/', studentListView.as_view(), name='std_list'),
    path('std/<int:pk>/', StudentDetailView.as_view()),
]
```

## Templates

### Std\_list.html

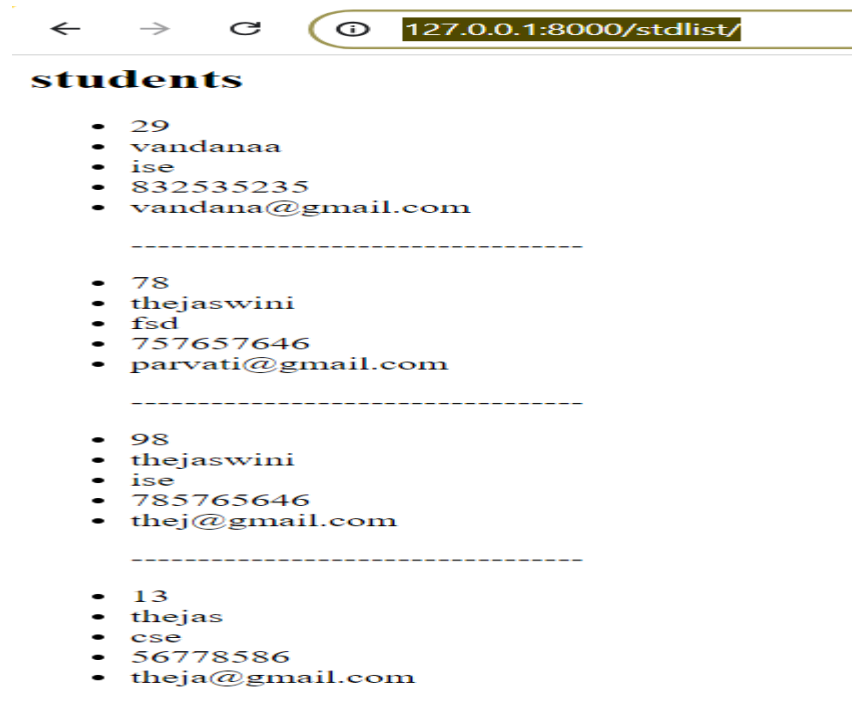
```
<html>
<body>
<h2>students</h2>

    <ul>
        {% for std in student %}
            <li>{{ std.usn }}</li>
            <li>{{ std.name }}</li>
            <li>{{ std.branch }}</li>
            <li>{{ std.phone }}</li>
            <li>{{ std.email }}</li>
            <p>-----</p>
        {% endfor %}
    </ul>
</body>
</html>
```

## detail.html

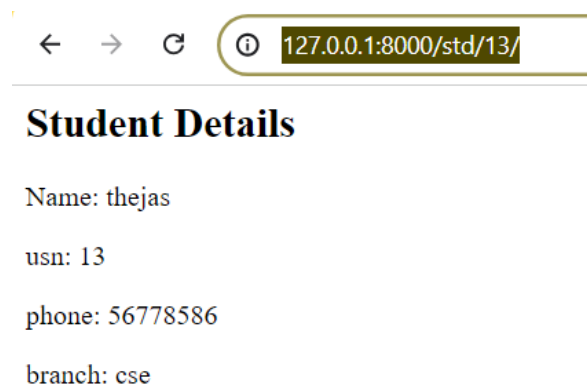
```
<html>
  <body>
    <h2>Student Details</h2>
    <p>Name: {{ student.name }}</p>
    <p>usn: {{ student.usn }}</p>
    <p>phone: {{ student.phone }}</p>
    <p>branch: {{ student.branch }}</p>

  </body>
</html>
```



A screenshot of a web browser displaying a list of students. The browser's address bar shows the URL `127.0.0.1:8000/stdlist/`. The page has a title **students** and contains a list of four student entries, each separated by a dashed line. Each entry includes a USN, a name, a phone number, and an email address.

USN	Name	Phone	Email
29	vandanaa	ise	832535235
78	thejaswini	fsd	757657646
98	thejaswini	ise	785765646
13	thejas	cse	56778586



A screenshot of a web browser displaying the details of a specific student. The browser's address bar shows the URL `127.0.0.1:8000/std/13/`. The page has a title **Student Details** and displays the following information:

Name: thejas

usn: 13

phone: 56778586

branch: cse

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

Install reportlab through terminal

Command → pip install reportlab

## Models.py

```
from django.db import models

# Create your models here.
class student(models.Model):
    usn = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=50)
    branch = models.CharField(max_length=20)
    phone = models.BigIntegerField()
    email = models.EmailField(max_length=20)

    def __str__(self):
        return self.name

    class Meta:
        db_table = "student"
```

## views.py

```
from models import student
from reportlab.pdfgen import canvas
import csv
from django.http import HttpResponse

def export_students_csv(request):
    # Create the HttpResponse object with the appropriate CSV header
    response = HttpResponse(content_type='text/csv')
    response['Content-Disposition'] = 'attachment;
filename="students.csv"'

    students = student.objects.all()

    writer = csv.writer(response)
```



```

writer.writerow(['usn', 'name', 'branch', 'phone', 'email'])

for s in students:
    writer.writerow([s.usn, s.name, s.branch, s.phone, s.email])

return response

def csvhome(request):
    return HttpResponse('''
        <html>
            <body>
                <h1>Download student List CSV From DB</h1>
                <a href="csv/">Download CSV</a>
            </body>
        </html>
    ''')

def getpdf(request):
    # Fetch all student objects from the database
    students = student.objects.all()

    # Create the HttpResponse object with the appropriate PDF header
    response = HttpResponse(content_type='application/pdf')
    response['Content-Disposition'] = 'attachment;
filename="students.pdf"'

    # Create a canvas object
    p = canvas.Canvas(response)

    # Set font and font size
    p.setFont("Times-Roman", 12)

    # Set initial y position
    y_position = 700

    p.drawString(100, y_position, "ID")
    p.drawString(150, y_position, "Name")
    p.drawString(300, y_position, "branch")

    y_position -= 25
    # Write student information to the PDF
    for s in students:
        # Write student ID, name, and description to the PDF

```

```

        p.drawString(100, y_position, f"{s.usn}")
        p.drawString(150, y_position, f"{s.name}")
        p.drawString(300, y_position, f"{s.branch}")

        y_position -= 25

    # If the y_position is too low, create a new page
    if y_position < 50:
        p.showPage()
        p.setFont("Times-Roman", 12)
        y_position = 700

    # Save the PDF
    p.save()

    return response

def pdfhome(request):
    return HttpResponse('''
        <html>
            <body>
                <h1>Download student List pdf</h1>
                <a href="pdf">Download pdf</a>
            </body>
        </html>
    ''')
```

## Urls.py

```

from django.urls import path
from . import views
from .views import StudentDetailView, studentListView

urlpatterns = [

    path('pdfhome/', views.pdfhome),
    path('pdf/', views.getpdf),
    path('csvhome/', views.csvhome),
    path('csv/', views.export_students_csv),

]
```

OUTPUT

127.0.0.1:8000

Download student List CSV From DB

[Download CSV](#)

Student.csv file

usn	name	branch	phone	email
3	chaitra	ise	98789899	chaitra@gmail.com
4	darshan	civil	9.01E+08	darshan@gmail.com
6	swetha	civil	5757655	swetha@gmail.com
7	deva	ise	67374883	dev@gmail.com
8	swetha	civil	6.58E+08	swetha@gmail.com
9	jayanthi	civil	67364803	jayanthi@gmail.com
10	ishanth	mech	97364803	ish@gmail.com
11	sneha	mech	96876876	sneha@gmail.com
12	neha	ise	8586565	neha@gail.com
13	thejas	cse	56778586	theja@gmail.com
14	sanjay	civil	6.45E+08	sanjay@gmail.com
17	sdf	cse	7.47E+08	sdf@gmail.com
18	hgf	fghg	8.69E+09	gfd@gmail.com
19	abhiii	mech	9.86E+08	abhi@gmail.com
21	paaru	civil	6.7E+09	parvati@gmail.com
22	madurya	cse	86497856	madurya@gmail.com
29	vandanaa	ise	8.33E+08	vandana@gmail.com
33	fsd	dfs	42334534	dfs@gmail.com
55	supriya	cse	4.69E+08	supriya@gmail.com
59	parvati	civil	7.67E+09	parvati@gmail.com
67	dfs	cse	56457567	dfs@gmail.com

127.0.0.1:8000

Download student List pdf

[Download pdf](#)

## Student.pdf

ID	Name	branch
3	chaitra	ise
4	darshan	civil
6	swetha	civil
7	deva	ise
8	swetha	civil
9	jayanthi	civil
10	ishanth	mech
11	sneha	mech
12	neha	ise
13	thejas	cse
14	sanjay	civil
17	sdf	cse
18	hgf	fghg
19	abhiii	mech
21	paaru	civil
29	vandanaa	ise
33	fsd	dfs