CS471 – Web Technologies (Laboratory)		Lab 9 / Django Models (Part 3)

This lab session covers Django Models (Part 3. The tasks include creating models like Student, Card, Department, and Course, establishing appropriate relationships between them, and using aggregation functions to retrieve and display meaningful data.

Pre-lab Preparation:

1. The fake data from mockaroo.com website or any other methods.

Lab Activities:

Part 1: Create two models: Student and Card as described below.

2) Create a relation between Student and Card.
2) Ensure that if you try to delete a Card that is linked to a Student, Django will raise an error and prevent the deletion.

Think carefully: What type of relationship (One-to-One, One-to-Many, Many-to-Many) is appropriate between a Student and a Card

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

    # Part 1: One-to-One relation with Card (prevent delete)
    card = models.OneToOneField(Card, on_delete=models.PROTECT, null=True, blank=True)

    # Part 2: ForeignKey to Department (delete students if department deleted)
    department = models.ForeignKey(Department, on_delete=models.CASCADE, null=True,
blank=True)

    # Part 3: ManyToMany with Course
    courses = models.ManyToManyField(Course, blank=True)

    def __str__(self):
        return self.name

class Card(models.Model):
    card_number = models.IntegerField()

    def __str__(self):
        return str(self.card_number)
```

Part 2: Create a model Department as described below.

2) Add a new field department to the Student model and link it to the Department model.
2) Make sure that if a department is deleted, all its students are deleted automatically.

Decide the appropriate relationship type between Student and Department.

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

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

Part 3: Create a model Course as described below.

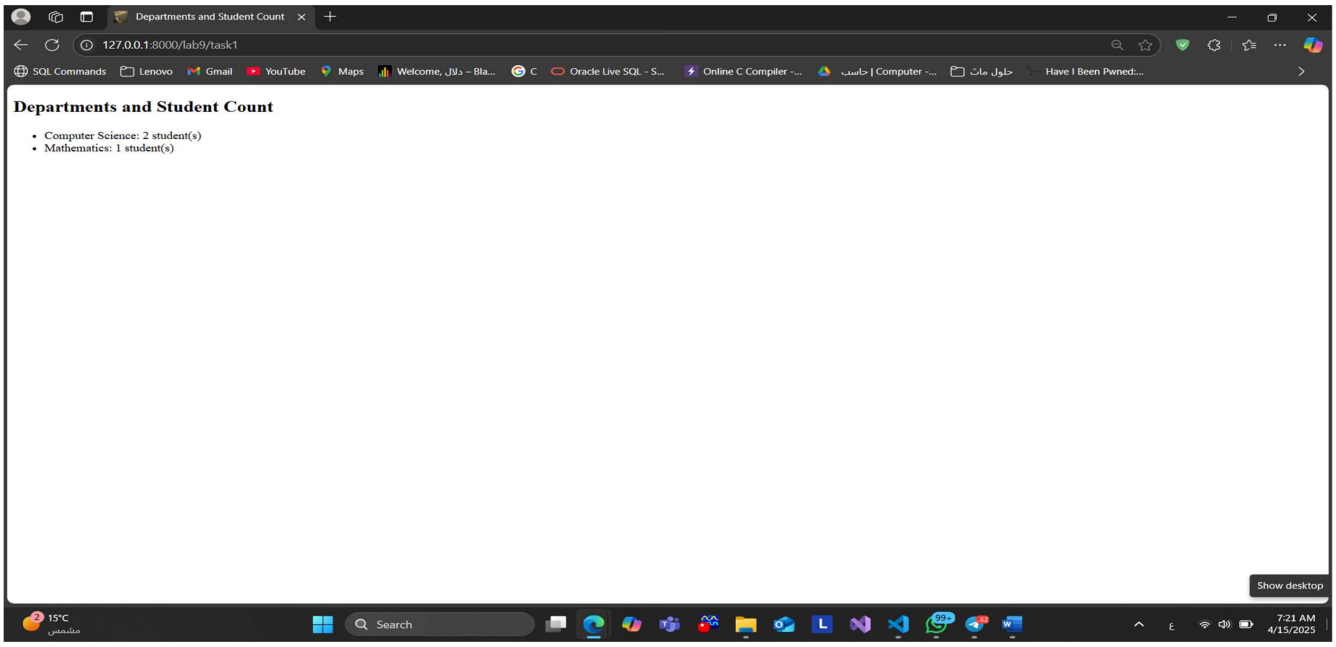
1) Add a new field course to the Student model and link it to the Course model.

Think carefully: Can a student have multiple courses? Can a course have multiple students? Choose the right relationship type.

```
class Course(models.Model):
    title = models.CharField(max_length=100)
    code = models.IntegerField()

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

Task 1: Create a URL /books/lab9/task1 with any necessary HTML file and view function to list each department with the number of students in it.

```
{% extends "layouts/base.html" %}

{% block title %} Departments and Student Count {% endblock %}

{% block content %}
<h2>Departments and Student Count</h2>
<ul>
    {% for dept in departments %}
        <li>{{ dept.name }}: {{ dept.student_count }} student(s)</li>
    {% endfor %}
</ul>
{% endblock %}
```

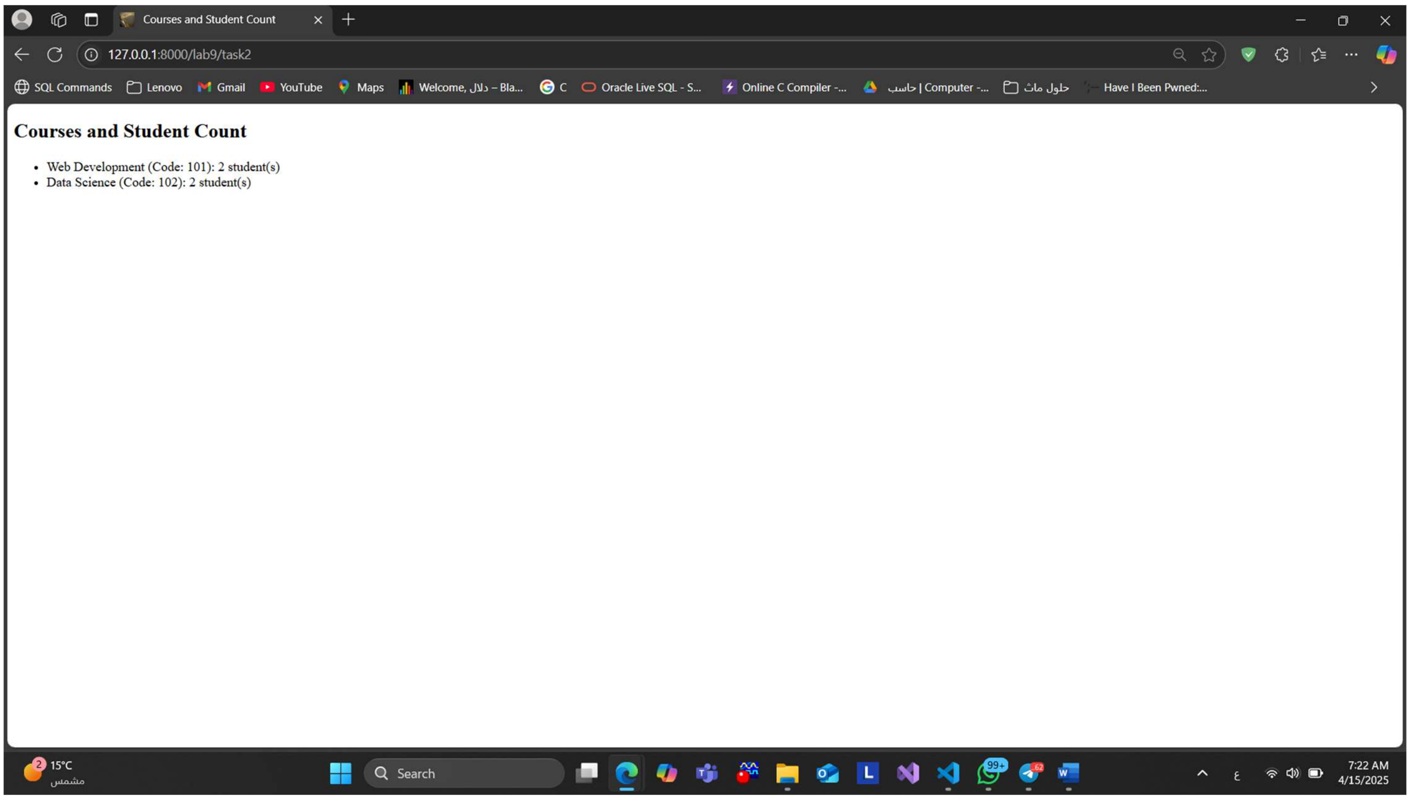

127.0.0.1:8000/lab9/task1

Departments and Student Count

- Computer Science: 2 student(s)
- Mathematics: 1 student(s)

Task 2: Create a URL /books/lab9/task2 with any necessary HTML file and view function to list each course with the number of students registered in it.

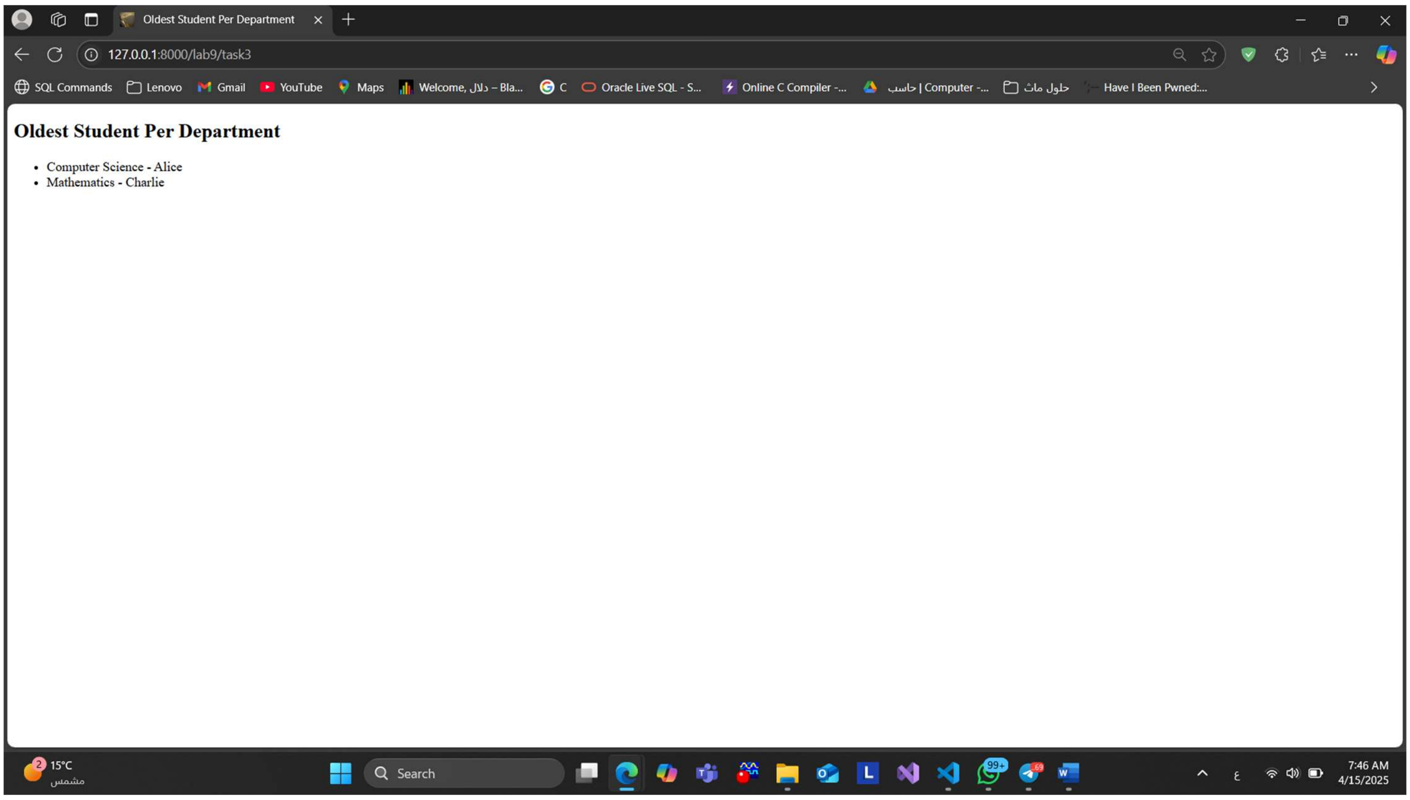
```
{% extends "layouts/base.html" %}
{% block title %} Courses and Student Count {% endblock %}

{% block content %}
<h2>Courses and Student Count</h2>
<ul>
    {% for course in courses %}
        <li>{{ course.title }} (Code: {{ course.code }}): {{ course.student_count }}
student(s)</li>
    {% endfor %}
</ul>
{% endblock %}
```



```
Courses and Student Count

  • Web Development (Code: 101): 2 student(s)
  • Data Science (Code: 102): 2 student(s)
```

Task 3: Create a URL /books/lab9/task3 with any necessary HTML file and view function to show the name of the oldest student (by ID) For each department.

```
{% extends "layouts/base.html" %}

{% block title %} Oldest Student Per Department {% endblock %}

{% block content %}
<h2>Oldest Student Per Department</h2>
<ul>
    {% for dept in departments %}
      <li>{{ dept.name }} - {{ dept.oldest_student_name }}</li>
    {% empty %}
      <li>لا توجد بيانات</li>
    {% endfor %}
</ul>
{% endblock %}
```


Oldest Student Per Department

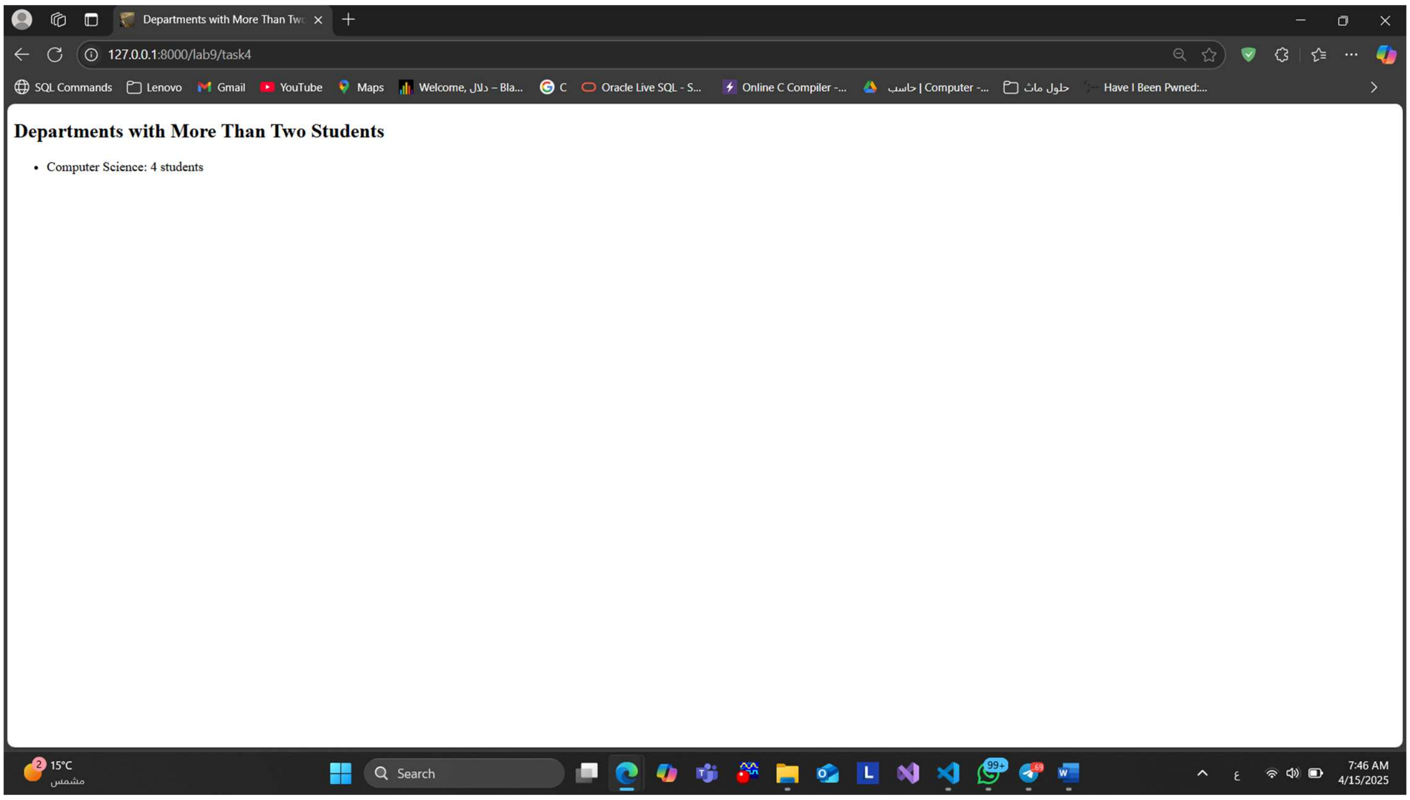
127.0.0.1:8000/lab9/task3

Oldest Student Per Department

- Computer Science - Alice
- Mathematics - Charlie

Task 4: Create a URL /books/lab9/task4 with any necessary HTML file and view function to list the departments that have more than two students, ordered by the number of students in descending order (from highest to lowest).

```
{% extends "layouts/base.html" %}
{% block title %} Departments with More Than Two Students {% endblock %}

{% block content %}
<h2>Departments with More Than Two Students</h2>
<ul>
    {% for dept in departments %}
        <li>{{ dept.name }}: {{ dept.student_count }} students</li>
    {% endfor %}
</ul>
{% endblock %}
```



Departments with More Than Two Students

- Computer Science: 4 students

Views.py

```python
from django.shortcuts import render
from django.db.models import Count, Min
from .models import Department, Course, Student
from django.db.models import Min, Subquery, OuterRef

def task1(request):
    departments = Department.objects.annotate(student_count=Count('student'))

    return render(request, 'lab9/task1.html', {'departments': departments})

def task2(request):
    courses = Course.objects.annotate(student_count=Count('student'))
    return render(request, 'lab9/task2.html', {'courses': courses})

def task3(request):

    departments = Department.objects.annotate(
        oldest_student_id=Min('student__id')
    )
    for dept in departments:

        oldest_student = Student.objects.filter(id=dept.oldest_student_id).first()
        dept.oldest_student_name = oldest_student.name if oldest_student else "No student"

    return render(request, 'lab9/task3.html', {'departments': departments})

def task4(request):
    departments =
Department.objects.annotate(student_count=Count('student')).filter(student_count__gt=2).order
_by('-student_count')
    return render(request, 'lab9/task4.html', {'departments': departments})
```

Urls.py

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

urlpatterns = [
    path('task1', views.task1, name='task1'),
    path('task2', views.task2, name='task2'),
    path('task3', views.task3, name='task3'),
    path('task4', views.task4, name='task4'),
]
```