CS471 – Web Technologies (Laboratory)	Qassim University College of Computer | كلية الحاسب جامعة القصيم	Lab 8 Django Models (Part 2)

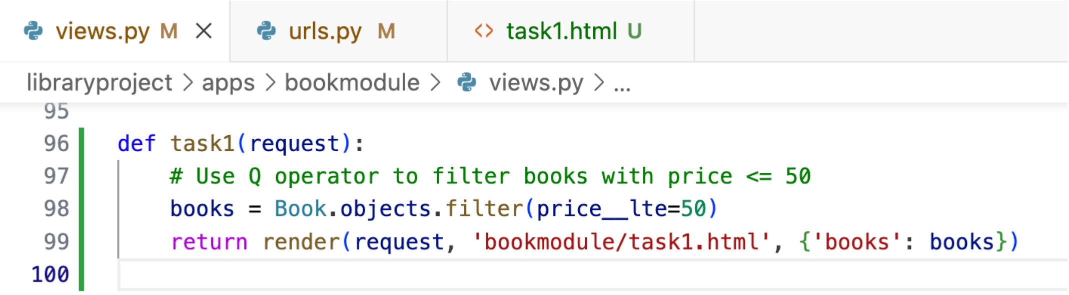
This lab session covers Django Models (Part 2 – Q operator, sorting objects, aggregation functions, and multi-models). By the end of this session, students should have a very good understanding of how to build advanced queries with multi-model query in Django.

Pre-lab Preparation:

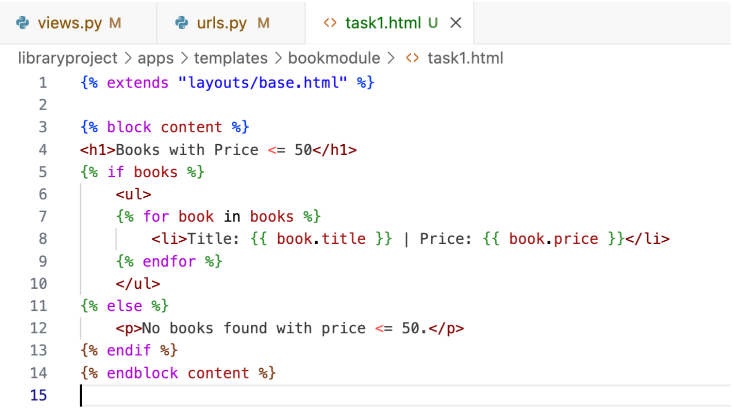
1. The table book as in lab7.
2. The fake data from mockaroo.com website or any other methods.

Lab Activities:

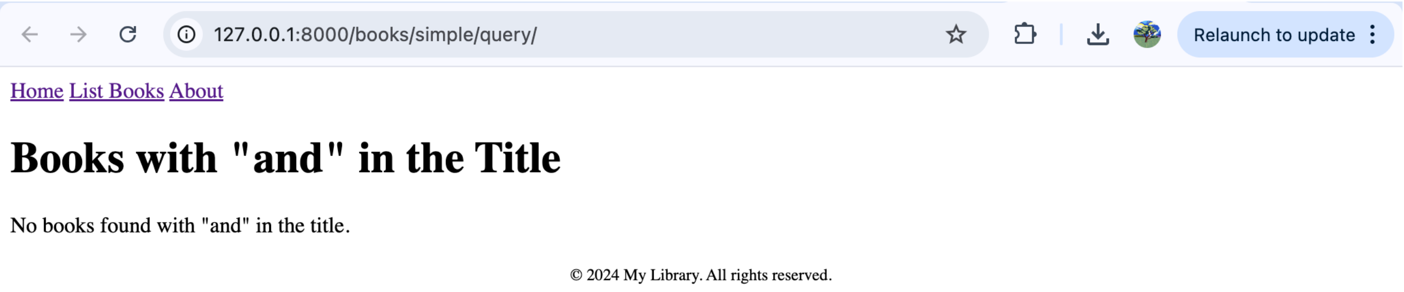
Task 1: Create a URL '/books/lab8/task1' with any necessary HTML file and view function to list books that have price less than or equal 50 using Q operator.



```python
def task1(request):
    # Use Q operator to filter books with price <= 50
    books = Book.objects.filter(price__lte=50)
    return render(request, 'bookmodule/task1.html', {'books': books})
```



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

{% block content %}
<h1>Books with Price <= 50</h1>
{% if books %}
    <ul>
    {% for book in books %}
        <li>Title: {{ book.title }} | Price: {{ book.price }}</li>
    {% endfor %}
    </ul>
{% else %}
    <p>No books found with price <= 50.</p>
{% endif %}
{% endblock content %}
```



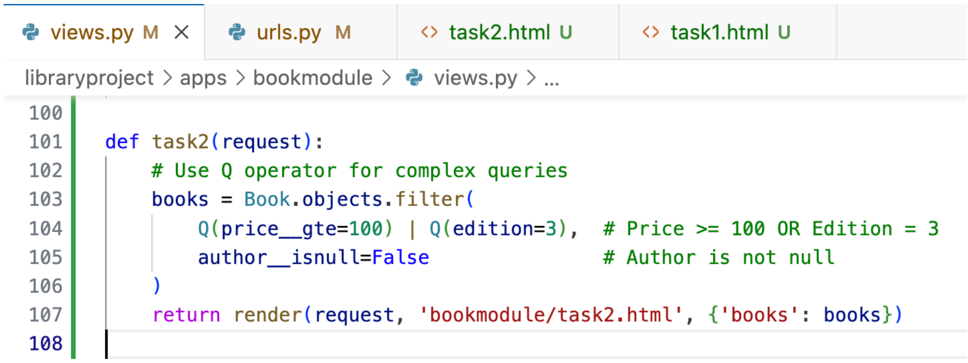
Home List Books About

Books with "and" in the Title

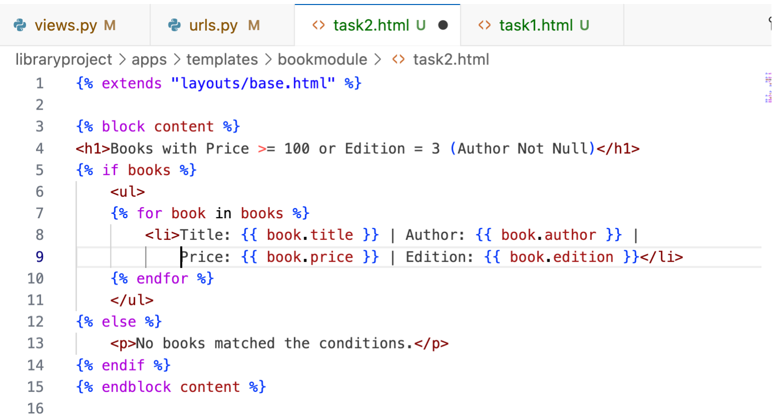
No books found with "and" in the title.

© 2024 My Library. All rights reserved.

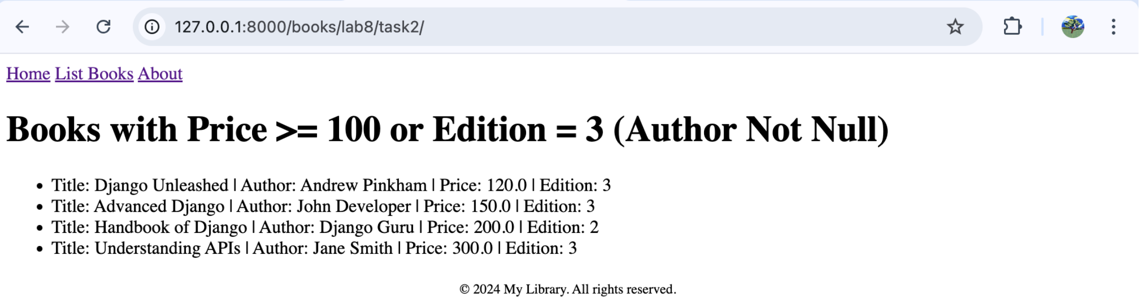
Task 2: Create a URL '/books/lab8/task2', HTML file, and view function to list books that have editions higher than two and either the title or author of the book contains the two adjacent letters 'qu' by combining two or more Q operator (through "&" or " | ").



```python

def task2(request):
    # Use Q operator for complex queries
    books = Book.objects.filter(
        Q(price__gte=100) | Q(edition=3),  # Price >= 100 OR Edition = 3
        author__isnull=False               # Author is not null
    )
    return render(request, 'bookmodule/task2.html', {'books': books})

```



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

{% block content %}
<h1>Books with Price >= 100 or Edition = 3 (Author Not Null)</h1>
{% if books %}
    <ul>
    {% for book in books %}
        <li>Title: {{ book.title }} | Author: {{ book.author }} |
            Price: {{ book.price }} | Edition: {{ book.edition }}</li>
    {% endfor %}
    </ul>
{% else %}
    <p>No books matched the conditions.</p>
{% endif %}
{% endblock content %}

```



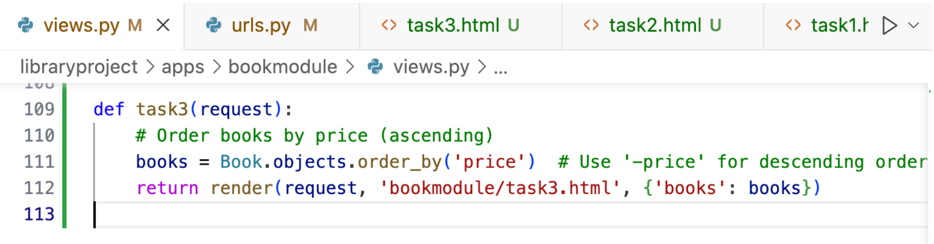
Home List Books About

Books with Price >= 100 or Edition = 3 (Author Not Null)

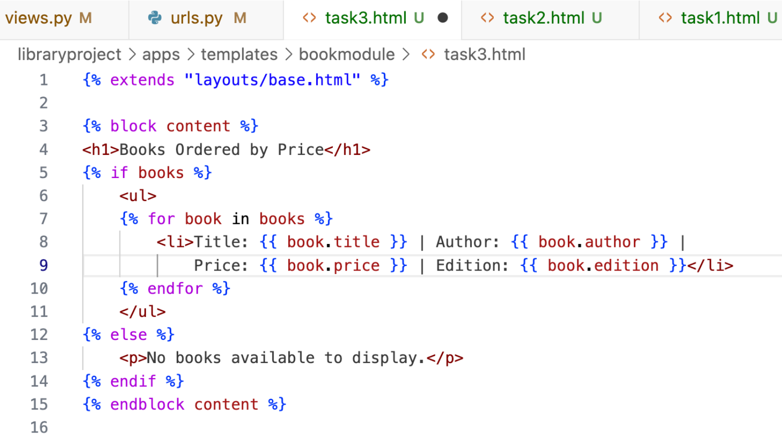
- Title: Django Unleashed | Author: Andrew Pinkham | Price: 120.0 | Edition: 3
- Title: Advanced Django | Author: John Developer | Price: 150.0 | Edition: 3
- Title: Handbook of Django | Author: Django Guru | Price: 200.0 | Edition: 2
- Title: Understanding APIs | Author: Jane Smith | Price: 300.0 | Edition: 3

© 2024 My Library. All rights reserved.

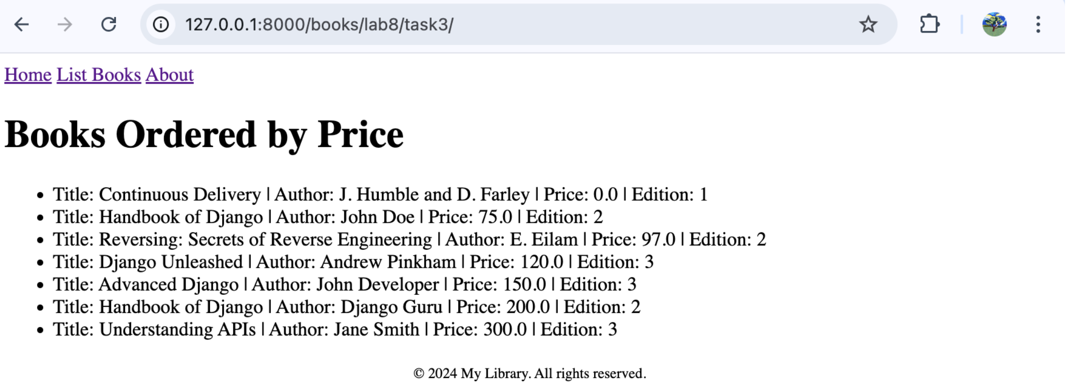
Task 3: (opposite to Task 2) Create a URL '/books/lab8/task3', HTML file, and view function to list books that have no editions higher than two and either the title or author of the book that does not contain the two adjacent letters "qu" by combining two or more Q operator (through "&" or "|", beside ~ operator).



```python
def task3(request):
    # Order books by price (ascending)
    books = Book.objects.order_by('price')  # Use '-price' for descending order
    return render(request, 'bookmodule/task3.html', {'books': books})
```



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

{% block content %}
<h1>Books Ordered by Price</h1>
{% if books %}
    <ul>
    {% for book in books %}
        <li>Title: {{ book.title }} | Author: {{ book.author }} |
            Price: {{ book.price }} | Edition: {{ book.edition }}</li>
    {% endfor %}
    </ul>
{% else %}
    <p>No books available to display.</p>
{% endif %}
{% endblock content %}

```



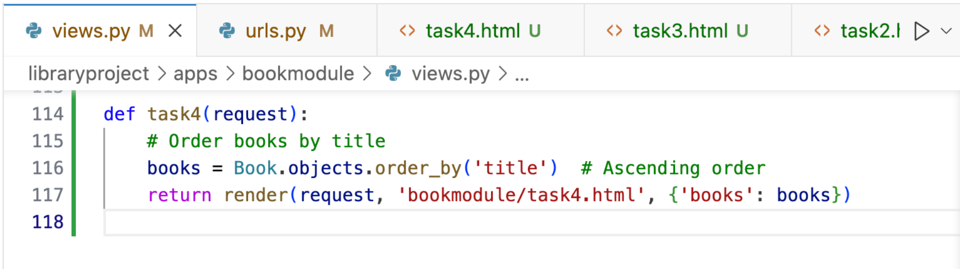
Home List Books About

Books Ordered by Price

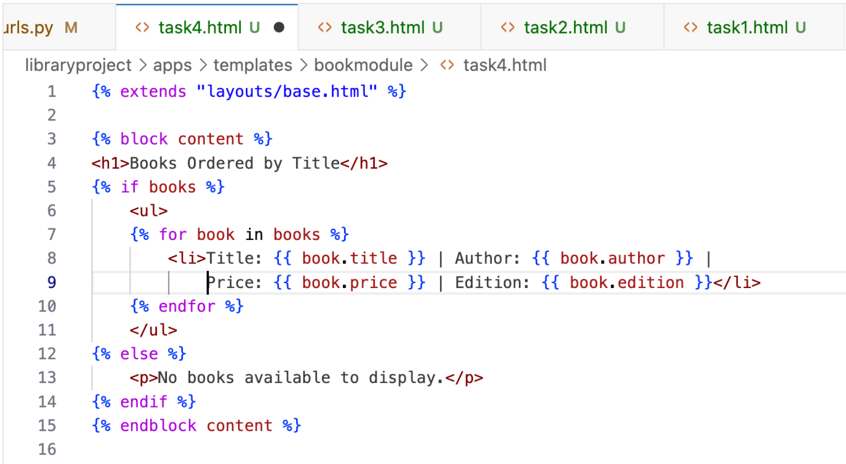
- Title: Continuous Delivery | Author: J. Humble and D. Farley | Price: 0.0 | Edition: 1
- Title: Handbook of Django | Author: John Doe | Price: 75.0 | Edition: 2
- Title: Reversing: Secrets of Reverse Engineering | Author: E. Eilam | Price: 97.0 | Edition: 2
- Title: Django Unleashed | Author: Andrew Pinkham | Price: 120.0 | Edition: 3
- Title: Advanced Django | Author: John Developer | Price: 150.0 | Edition: 3
- Title: Handbook of Django | Author: Django Guru | Price: 200.0 | Edition: 2
- Title: Understanding APIs | Author: Jane Smith | Price: 300.0 | Edition: 3

© 2024 My Library. All rights reserved.

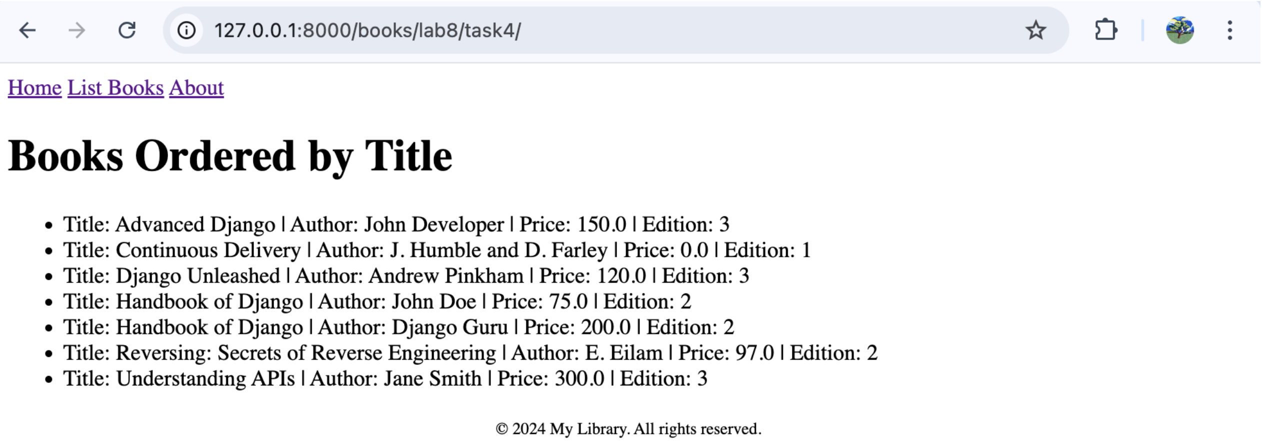
Task 4: Create a URL '/books/lab8/task4', HTML file, and view function to list books and order by their titles using the function "order_by".



```python
def task4(request):
    # Order books by title
    books = Book.objects.order_by('title')  # Ascending order
    return render(request, 'bookmodule/task4.html', {'books': books})
```



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

{% block content %}
<h1>Books Ordered by Title</h1>
{% if books %}
    <ul>
    {% for book in books %}
        <li>Title: {{ book.title }} | Author: {{ book.author }} |
            Price: {{ book.price }} | Edition: {{ book.edition }}</li>
    {% endfor %}
    </ul>
{% else %}
    <p>No books available to display.</p>
{% endif %}
{% endblock content %}

```



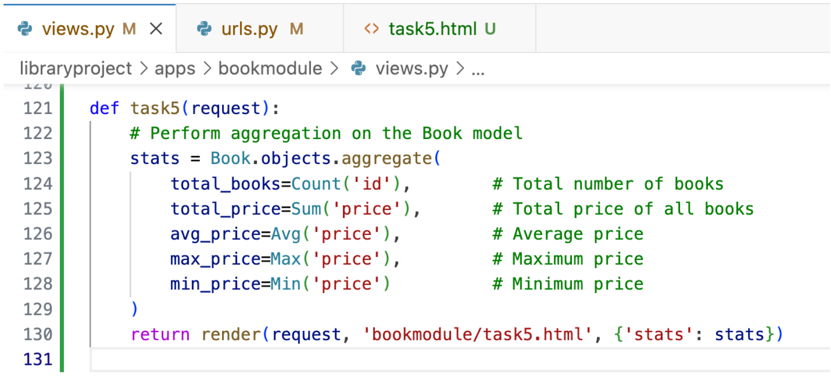
Home List Books About

Books Ordered by Title

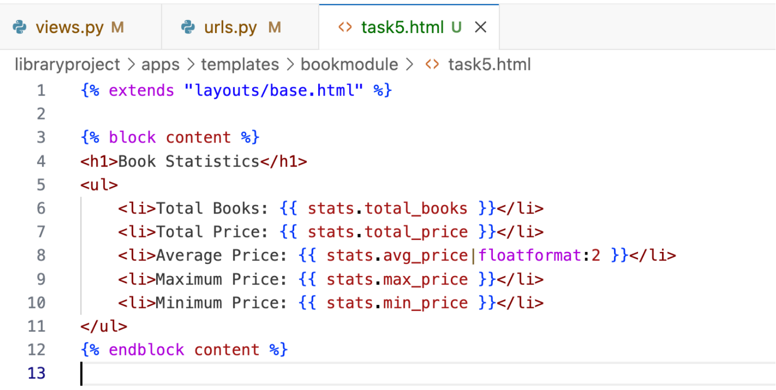
- Title: Advanced Django | Author: John Developer | Price: 150.0 | Edition: 3
- Title: Continuous Delivery | Author: J. Humble and D. Farley | Price: 0.0 | Edition: 1
- Title: Django Unleashed | Author: Andrew Pinkham | Price: 120.0 | Edition: 3
- Title: Handbook of Django | Author: John Doe | Price: 75.0 | Edition: 2
- Title: Handbook of Django | Author: Django Guru | Price: 200.0 | Edition: 2
- Title: Reversing: Secrets of Reverse Engineering | Author: E. Eilam | Price: 97.0 | Edition: 2
- Title: Understanding APIs | Author: Jane Smith | Price: 300.0 | Edition: 3

© 2024 My Library. All rights reserved.

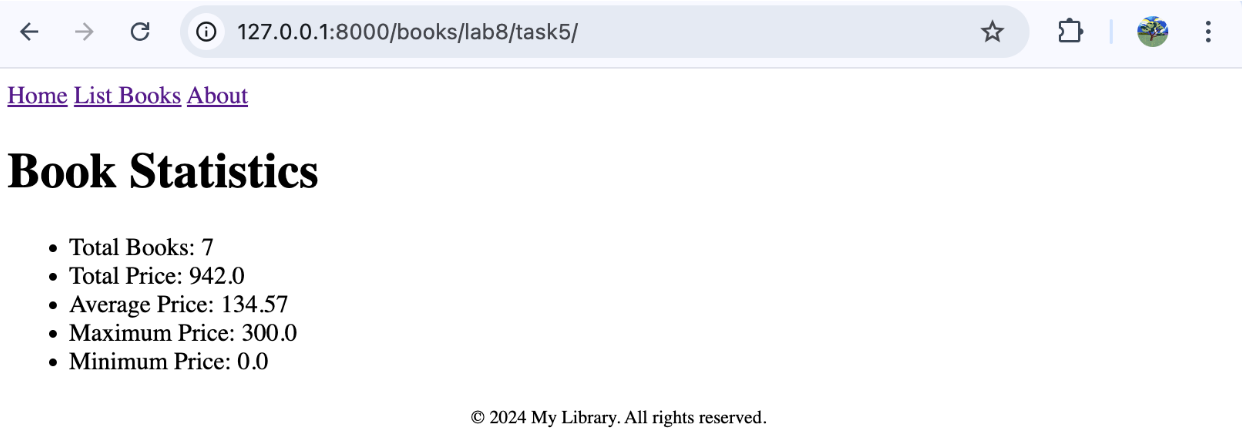
Task 5: Create a URL '/books/lab8/task5', HTML file, and view function to display the number of books, total price of all books, average price, maximum price, and minimum price using aggregation functions.



```python
def task5(request):
    # Perform aggregation on the Book model
    stats = Book.objects.aggregate(
        total_books=Count('id'),        # Total number of books
        total_price=Sum('price'),       # Total price of all books
        avg_price=Avg('price'),         # Average price
        max_price=Max('price'),         # Maximum price
        min_price=Min('price')          # Minimum price
    )
    return render(request, 'bookmodule/task5.html', {'stats': stats})
```



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

{% block content %}
<h1>Book Statistics</h1>
<ul>
    <li>Total Books: {{ stats.total_books }}</li>
    <li>Total Price: {{ stats.total_price }}</li>
    <li>Average Price: {{ stats.avg_price|floatformat:2 }}</li>
    <li>Maximum Price: {{ stats.max_price }}</li>
    <li>Minimum Price: {{ stats.min_price }}</li>
</ul>
{% endblock content %}
```



Home List Books About

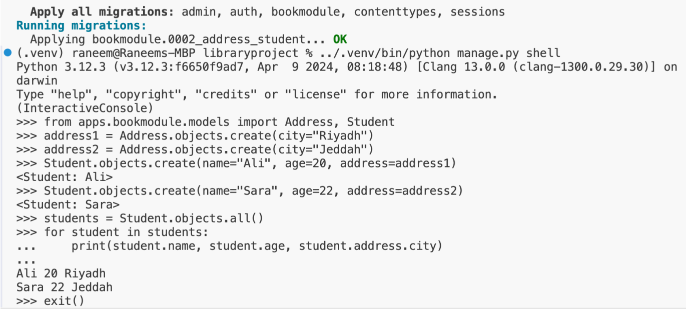
Book Statistics

- Total Books: 7
- Total Price: 942.0
- Average Price: 134.57
- Maximum Price: 300.0
- Minimum Price: 0.0

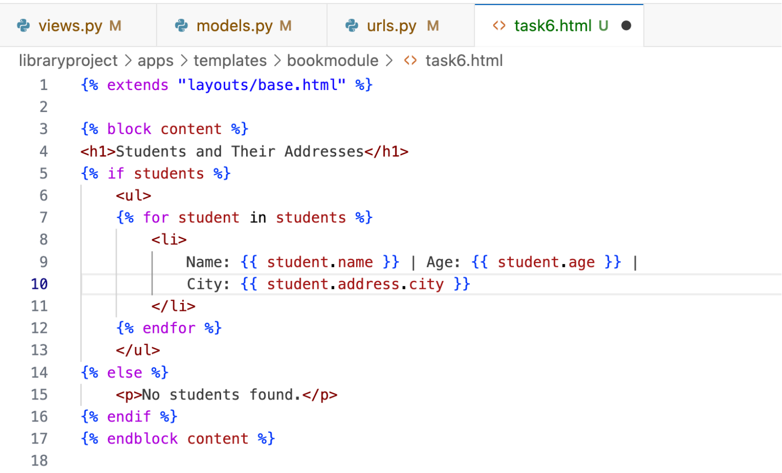
© 2024 My Library. All rights reserved.

Task 6: Create a database scheme (through models in Django)

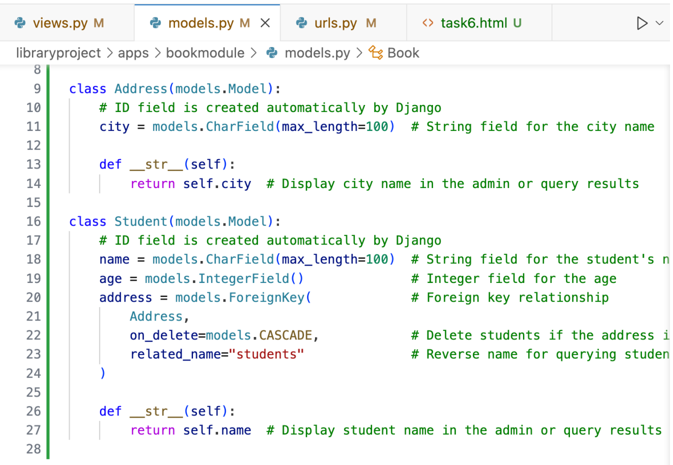
Note that instead of creating another app to handle student applications, you may exploit existing apps, like user or book modules



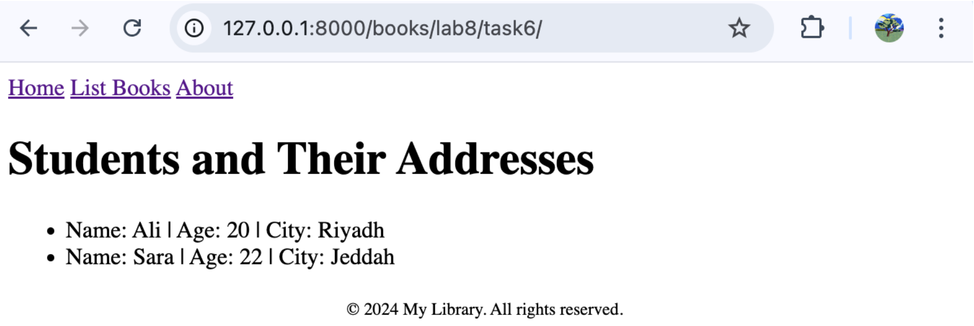
```
  Apply all migrations: admin, auth, bookmodule, contenttypes, sessions
Running migrations:
  Applying bookmodule.0002_address_student... OK
(.venv) raneem@Raneems-MBP libraryproject % ../.venv/bin/python manage.py shell
Python 3.12.3 (v3.12.3:f6650f9ad7, Apr  9 2024, 08:18:48) [Clang 13.0.0 (clang-1300.0.29.30)] on
darwin
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from apps.bookmodule.models import Address, Student
>>> address1 = Address.objects.create(city="Riyadh")
>>> address2 = Address.objects.create(city="Jeddah")
>>> Student.objects.create(name="Ali", age=20, address=address1)
<Student: Ali>
>>> Student.objects.create(name="Sara", age=22, address=address2)
<Student: Sara>
>>> students = Student.objects.all()
>>> for student in students:
...     print(student.name, student.age, student.address.city)
...
Ali 20 Riyadh
Sara 22 Jeddah
>>> exit()
```



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

{% block content %}
<h1>Students and Their Addresses</h1>
{% if students %}
    <ul>
    {% for student in students %}
        <li>
            Name: {{ student.name }} | Age: {{ student.age }} |
            City: {{ student.address.city }}
        </li>
    {% endfor %}
    </ul>
{% else %}
    <p>No students found.</p>
{% endif %}
{% endblock content %}

```



```

class Address(models.Model):
    # ID field is created automatically by Django
    city = models.CharField(max_length=100)  # String field for the city name

    def __str__(self):
        return self.city  # Display city name in the admin or query results

class Student(models.Model):
    # ID field is created automatically by Django
    name = models.CharField(max_length=100)  # String field for the student's n
    age = models.IntegerField()              # Integer field for the age
    address = models.ForeignKey(             # Foreign key relationship
        Address,
        on_delete=models.CASCADE,            # Delete students if the address i
        related_name="students"              # Reverse name for querying studen
    )

    def __str__(self):
        return self.name  # Display student name in the admin or query results

```



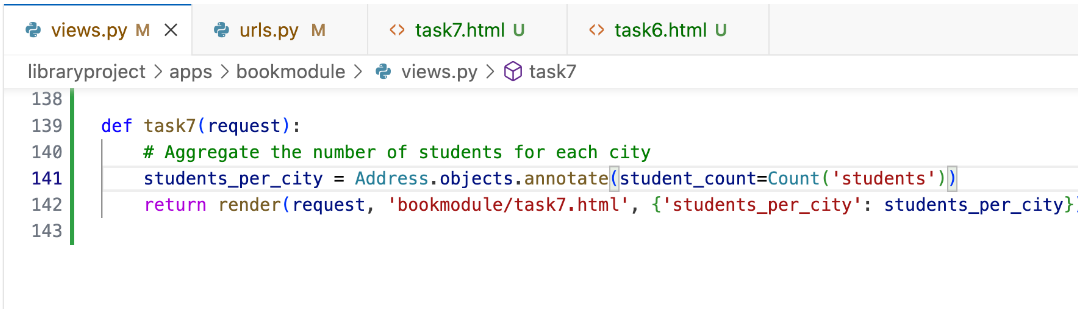
Home List Books About

Students and Their Addresses

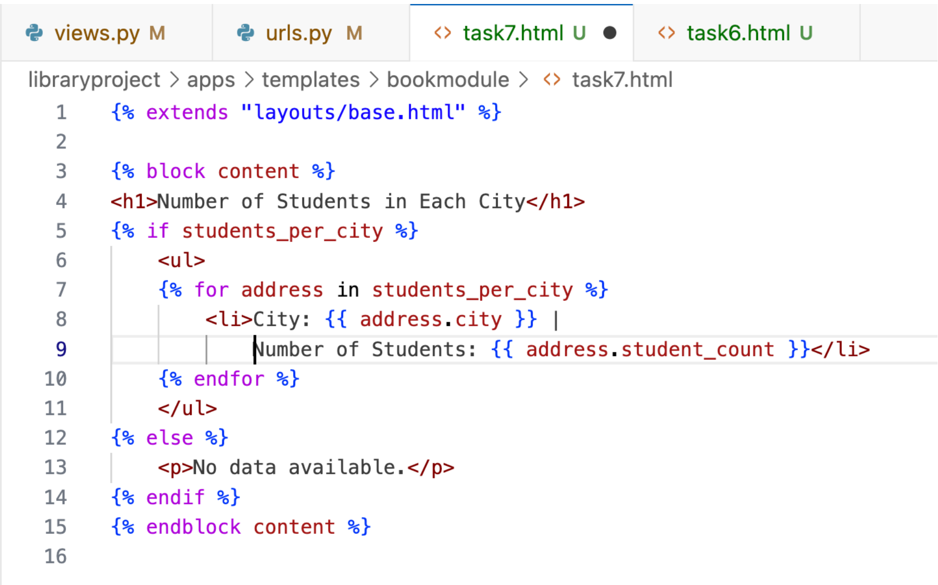
- Name: Ali | Age: 20 | City: Riyadh
- Name: Sara | Age: 22 | City: Jeddah

© 2024 My Library. All rights reserved.

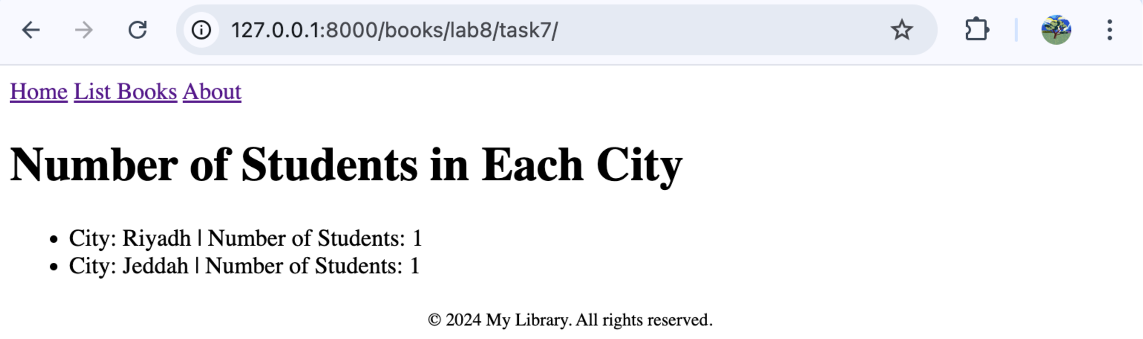
Task7: Create a page that shows the number of students in each city.



```python

def task7(request):
    # Aggregate the number of students for each city
    students_per_city = Address.objects.annotate(student_count=Count('students'))
    return render(request, 'bookmodule/task7.html', {'students_per_city': students_per_city}

```



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

{% block content %}
<h1>Number of Students in Each City</h1>
{% if students_per_city %}
    <ul>
    {% for address in students_per_city %}
        <li>City: {{ address.city }} |
            Number of Students: {{ address.student_count }}</li>
    {% endfor %}
    </ul>
{% else %}
    <p>No data available.</p>
{% endif %}
{% endblock content %}

```



Home List Books About

Number of Students in Each City

- City: Riyadh | Number of Students: 1
- City: Jeddah | Number of Students: 1

© 2024 My Library. All rights reserved.