



Course Code | Course Name

COMP8347-2-R-2023F | Internet Appl/ Distributed Sys

Document Type

Lab 4

Professor

Dr. Saja Al Mamoori

Graduate Assistant

Ms. Amangel Bhullar

Team - Members

Student ID

Manjinder Singh

110097177

Harbhajan Singh

110100089

Karan Vishavjit

110099867

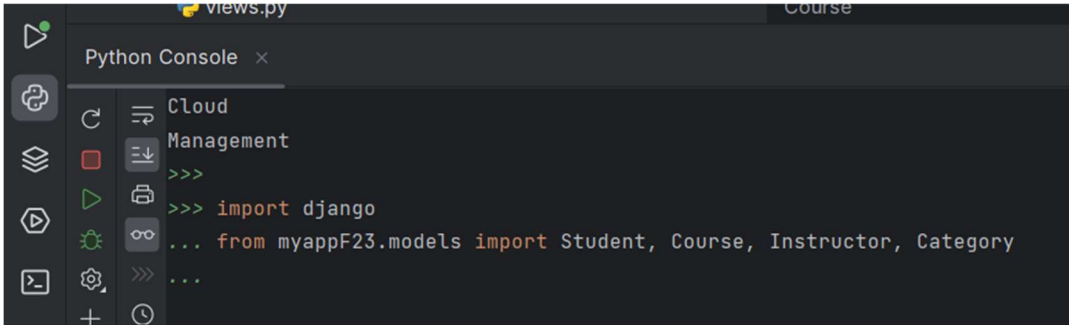
Khyati Shah

110090234

NOTES: For simplicity questions and content from lab manual 4 are mentioned in a box.

Part 3: Querying the database.

1. Open **Python Console**. In **Python console** import Django then models from *models.py*.
`import django`
`from myappF23.models import Student, Course, Instructor, Category`



Screenshot 1: Snippet of solution 1

2. Write queries to obtain the following information. Verify if your query generates the correct answer using the data entered in question 2.5. Can you use `annotate` for any of these queries? explain to the grader.

The `annotate` method in Django's ORM is used to perform aggregations on your data, like counting, summing, or averaging related objects. None of the given queries require aggregations. All queries listed below are simple retrieval operations without a need for aggregation. Thus, in this specific case, there's no need to use `annotate`.

2.1 Write a separate query to list all the students, instructors, courses, and categories in the db.

```
print("\nAnswer 2.1 ---")

students = Student.objects.all()

print("1. Enlisting student names:- ")

for student in students:

    print("-->",student.first_name, student.last_name)

instructors = Instructor.objects.all()

print("\n2. Enlisting instructor names:- ")

for instructor in instructors:

    print("-->",instructor.first_name, instructor.last_name)

courses = Course.objects.all()

print("\n3. Enlisting course names:-")

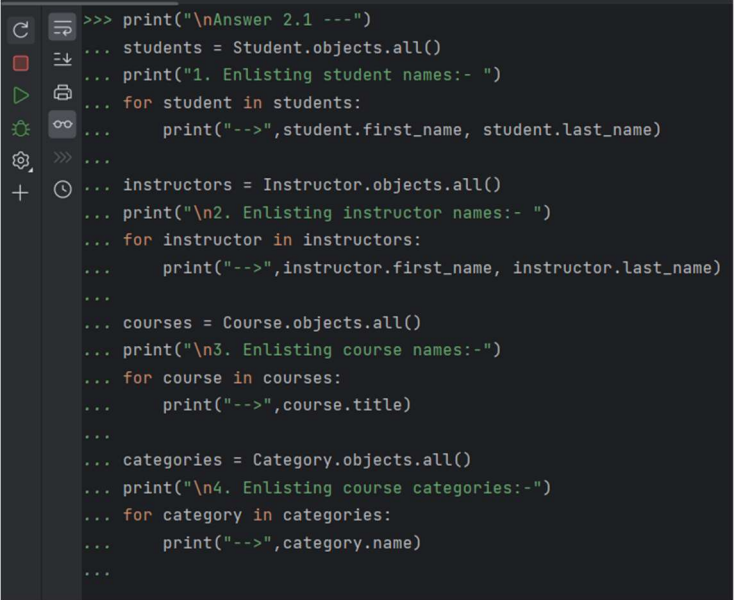
for course in courses:

    print("-->",course.title)

categories = Category.objects.all()

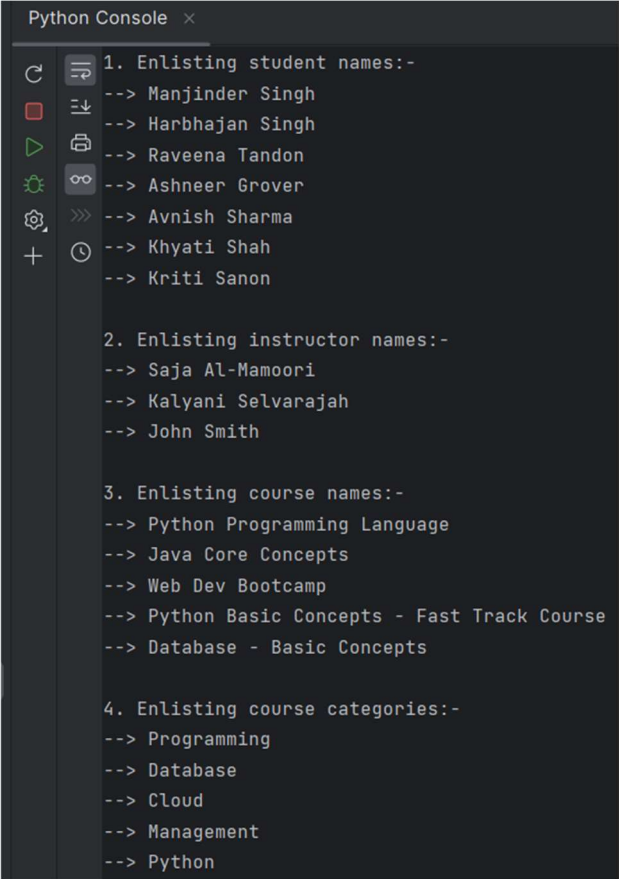
print("\n4. Enlisting course categories:-")
```

```
for category in categories:  
    print("-->",category.name)
```



```
>>> print("\nAnswer 2.1 ---")  
... students = Student.objects.all()  
... print("1. Enlisting student names:- ")  
... for student in students:  
...     print("-->",student.first_name, student.last_name)  
...  
... instructors = Instructor.objects.all()  
... print("\n2. Enlisting instructor names:- ")  
... for instructor in instructors:  
...     print("-->",instructor.first_name, instructor.last_name)  
...  
... courses = Course.objects.all()  
... print("\n3. Enlisting course names:-")  
... for course in courses:  
...     print("-->",course.title)  
...  
... categories = Category.objects.all()  
... print("\n4. Enlisting course categories:-")  
... for category in categories:  
...     print("-->",category.name)  
...  
...
```

Screenshot 2: Snippet of code solution of question 2.1



```
Python Console x  
1. Enlisting student names:-  
--> Manjinder Singh  
--> Harbhajan Singh  
--> Raveena Tandon  
--> Ashneer Grover  
--> Avnish Sharma  
--> Khyati Shah  
--> Kriti Sanon  
  
2. Enlisting instructor names:-  
--> Saja Al-Mamoori  
--> Kalyani Selvarajah  
--> John Smith  
  
3. Enlisting course names:-  
--> Python Programming Language  
--> Java Core Concepts  
--> Web Dev Bootcamp  
--> Python Basic Concepts - Fast Track Course  
--> Database - Basic Concepts  
  
4. Enlisting course categories:-  
--> Programming  
--> Database  
--> Cloud  
--> Management  
--> Python
```

Screenshot 3: Snippet of code output of question 2.1

2.2 List all courses in a specific category (e.g., 'Programming')

```
filter_by_category = Course.objects.filter(categories__name='Programming')  
  
print("\nAnswer 2.2 - Enlisting all courses of Programming category:-")  
  
for index, course in enumerate(filter_by_category, start=1):  
    print(f"{index}. {course.title}")
```

```
>>> filter_by_category = Course.objects.filter(categories__name='Programming')  
... print("\nAnswer 2.2 - Enlisting all courses of Programming category:-")  
... for index, course in enumerate(filter_by_category, start=1):  
...     print(f"{index}. {course.title}")  
...  
  
Answer 2.2 - Enlisting all courses of Programming category:-  
1. Python Programming Language  
2. Java Core Concepts  
3. Web Dev Bootcamp  
4. Python Basic Concepts - Fast Track Course
```

Screenshot 4: Snippet of code solution and output of question 2.2

2.3 List all courses taught by a specific instructor (e.g., 'John Smith')

```
filter_by_instructor = Course.objects.filter(instructor__first_name='John',  
instructor__last_name='Smith')  
  
print("\nAnswer 2.3 - Courses taught by John Smith instructor:-")  
  
for index, course in enumerate(filter_by_instructor, start=1):  
    print(f"{index}. {course.title}")
```

```
>>> filter_by_instructor = Course.objects.filter(instructor__first_name='John', instructor__last_name='Smith')  
... print("\nAnswer 2.3 - Courses taught by John Smith instructor:-")  
... for index, course in enumerate(filter_by_instructor, start=1):  
...     print(f"{index}. {course.title}")  
...  
  
Answer 2.3 - Courses taught by John Smith instructor:-  
1. Python Basic Concepts - Fast Track Course  
2. Database - Basic Concepts
```

Screenshot 5: Snippet of code solution and output of question 2.3

2.4 List all courses with a price greater than \$100.

```
filter_by_price = Course.objects.filter(price__gt=100)  
  
print("\nAnswer 2.4 - Enlisting courses that are priced above $100:")  
  
for index, course in enumerate(filter_by_price, start=1):  
    print(f"{index}. {course.title}")
```

```
>>> filter_by_price = Course.objects.filter(price__gt=100)
... print("\nAnswer 2.4 - Enlisting courses that are priced above $100:")
... for index, course in enumerate(filter_by_price, start=1):
...     print(f"{index}. {course.title}")
...

Answer 2.4 - Enlisting courses that are priced above $100:
1. Python Programming Language
2. Java Core Concepts
3. Web Dev Bootcamp
```

Screenshot 6: Snippet of code solution and output of question 2.4

2.5 List all students enrolled in a specific course (e.g., 'Web Dev Bootcamp').

```
filter_by_course_name = Student.objects.filter(course__title='Web Dev Bootcamp')
print("\nAnswer 2.5 - Students enrolled in Web Dev Bootcamp course:")
for index, student in enumerate(filter_by_course_name, start=1):
    print(f"{index}. {student.first_name} {student.last_name}")
```

```
>>> filter_by_course_name = Student.objects.filter(course__title='Web Dev Bootcamp')
... print("\nAnswer 2.5 - Students enrolled in Web Dev Bootcamp course:")
... for index, student in enumerate(filter_by_course_name, start=1):
...     print(f"{index}. {student.first_name} {student.last_name}")
...

Answer 2.5 - Students enrolled in Web Dev Bootcamp course:
1. Manjinder Singh
2. Raveena Tandon
3. Ashneer Grover
4. Avnish Sharma
```

Screenshot 7: Snippet of code solution and output of question 2.5

2.6 List all courses with start dates in the future.

```
from datetime import datetime

future_courses = Course.objects.filter(start_date__gt=datetime.now().date())
print("\nAnswer 2.6 - Enlisting all the courses that are scheduled in the future:-")
for index, course in enumerate(future_courses, start=1):
    print(f"{index}. {course.title}")
```

```
>>> from datetime import datetime
... future_courses = Course.objects.filter(start_date__gt=datetime.now().date())
... print("\nAnswer 2.6 - Enlisting all the courses that are scheduled in the future:-")
... for index, course in enumerate(future_courses, start=1):
...     print(f"{index}. {course.title}")
...

Answer 2.6 - Enlisting all the courses that are scheduled in the future:-
1. Java Core Concepts
2. Database - Basic Concepts
```

Screenshot 8: Snippet of code solution and output of question 2.6

2.7 Retrieve a specific student by email (e.g., 'john@gmail.com'):

```
get_name_from_email = Student.objects.get(email='lnu8@uwindsor.ca')

print(f"\nAnswer 2.7 - Student name with email id - { get_name_from_email.email} is having name {
get_name_from_email.first_name} { get_name_from_email.last_name}.")
```

```
Answer 2.7 - Student name with email id - lnu8@uwindsor.ca is having name Manjinder Singh.
>>>
... get_name_from_email = Student.objects.get(email='lnu8@uwindsor.ca')
... print(f"\nAnswer 2.7 - Student name with email id - { get_name_from_email.email} is having name {
get_name_from_email.first_name} { get_name_from_email.last_name}.")
...

Answer 2.7 - Student name with email id - lnu8@uwindsor.ca is having name Manjinder Singh.
```

Screenshot 9: Snippet of code solution and output of question 2.7

2.8 Retrieve a specific instructor by name (e.g., 'Saja Al-Mamoori'):

```
Inst_Name = Instructor.objects.get(first_name='Saja', last_name='Al-Mamoori')

print(f"\nAnswer 2.8 - Fetched instructor name:- {Inst_Name.first_name} {Inst_Name.last_name}")
```

```
>>> Inst_Name = Instructor.objects.get(first_name='Saja', last_name='Al-Mamoori')
... print(f"\nAnswer 2.8 - Fetched instructor name:- {Inst_Name.first_name} {Inst_Name.last_name}")
...

Answer 2.8 - Fetched instructor name:- Saja Al-Mamoori
```

Screenshot 10: Snippet of code solution and output of question 2.8

2.9 List all courses with a specific difficulty level (e.g., 'Intermediate').

```
courses_of_intermediate_level = Course.objects.filter(level='IN')

print("\nAnswer 2.9 - Enlisting courses of intermediate level:-")

for index, course in enumerate(courses_of_intermediate_level, start=1):

    print(f"{index}. {course.title}")
```



```
>>> courses_of_intermediate_level = Course.objects.filter(level='IN')
... print("\nAnswer 2.9 - Enlisting courses of intermediate level:-")
... for index, course in enumerate(courses_of_intermediate_level, start=1):
...     print(f"{index}. {course.title}")
...

Answer 2.9 - Enlisting courses of intermediate level:-
1. Python Programming Language
2. Java Core Concepts
```

Screenshot 11: Snippet of code solution and output of question 2.9

2.10 List all courses that have the word 'Python' in their title.

```
courses_with_word_python = Course.objects.filter(title__icontains='Python')
print("\nAnswer 2.10 - Enlisting courses containing 'Python' in title:-")
for index, course in enumerate(courses_with_word_python, start=1):
    print(f"{index}. {course.title}")
```

```
>>> courses_with_word_python = Course.objects.filter(title__icontains='Python')
... print("\nAnswer 2.10 - Enlisting courses containing 'Python' in title:-")
... for index, course in enumerate(courses_with_word_python, start=1):
...     print(f"{index}. {course.title}")
...

Answer 2.10 - Enlisting courses containing 'Python' in title:-
1. Python Programming Language
2. Python Basic Concepts - Fast Track Course
```

Screenshot 12: Snippet of code solution and output of question 2.10

2.11 List all students who were born after a certain date (e.g., after January 1, 2000).

```
students_year2000 = Student.objects.filter(date_of_birth__gt="2000-01-01")
print("\nAnswer 2.11 - Enlisting students born after January 1, 2000:- ")
for index, student in enumerate(students_year2000, start=1):
    print(f"{index}. {student.first_name} {student.last_name}")
```

```
>>> students_year2000 = Student.objects.filter(date_of_birth__gt="2000-01-01")
... print("\nAnswer 2.11 - Enlisting students born after January 1, 2000:- ")
... for index, student in enumerate(students_year2000, start=1):
...     print(f"{index}. {student.first_name} {student.last_name}")
...

Answer 2.11 - Enlisting students born after January 1, 2000:-
1. Avnish Sharma
2. Kriti Sanon
```

Screenshot 13: Snippet of code solution and output of question 2.11

2.12 List all courses that started after a specific date (e.g., after January 1, 2023).
--

```
courses2023 = Course.objects.filter(start_date__gt="2023-01-01")
```

```
print("\nAnswer 2.12 – Enlisting courses that started after January 1, 2023:-")
```

```
for index, course in enumerate(courses2023, start=1):
```

```
    print(f"{index}. {course.title}")
```

```
>>> courses2023 = Course.objects.filter(start_date__gt="2023-01-01")
... print("\nAnswer 2.12 – Enlisting courses that started after January 1, 2023:-")
... for index, course in enumerate(courses2023, start=1):
...     print(f"{index}. {course.title}")
...
```

```
Answer 2.12 – Enlisting courses that started after January 1, 2023:-
1. Python Programming Language
2. Java Core Concepts
3. Web Dev Bootcamp
4. Python Basic Concepts - Fast Track Course
```

Screenshot 14: Snippet of code solution and output of question 2.12