COMP-8347: Internet Applications and Distributes Systems

I/S 2022 LAB #1

Due Date: Sec. 4 Wed May 18, Sec. 5 Sun May 15

Submitted by: GROUP 3-

Vedangkumar Jotaniya ,

Jayesh Prajapati,

Dhruv Shah,

Dharam Horiya

PART 3:

1. import django

from myapp.models import Topic, Course, Student, Order

a. List all the courses in the db.

```
>>> courses = Course.objects.all()
>>> courses
<QuerySet [<Course: CCNA>, <Course: AWS>, <Course: CompTIA Security>, <Course: Oracle SQL>, <Course: Microeconomics>, <Course: Introduction to Economics>, <Course: Global Economics>, <Course: Web Development Bootcamp>, <Course: Web Design>, <Course: Web Development Masterclass>, <Course: Build Responsive Websites>, <Course: Innovation for Business>, <Course: Leadership Skills>, <Course: Project Management>]>
>>> students = Student.objects.all()
>>> students
<QuerySet [<Student: Mary Jones>, <Student: John Smith>, <Student: Alan George>, <Student: Josh Jones>, <Student: Chris Bill>]>
```

b. List all the students in the db.

```
>>> students = Student.objects.all()
>>> students
<QuerySet [<Student: Mary Jones>, <Student: John Smith>, <Student: Alan George>, <Student: Josh Jones>, <Student: Chris Bill>]>
```

c. List all the orders in the db.

```
>>> orders = Order.objects.all()
>>> orders
<QuerySet [<Order: order#1 courseName: Web Development Bootcamp StudentId: Mary Jones>, <Order: order#2 courseName: Project Management StudentId: Mary Jones>, <Order: order#3 courseName: CCNA StudentId: John Smith>, <Order: order#4 courseName: Microeconomics StudentId: Chris Bill>, <Order: order#5 courseName: Build Responsive Websites StudentId: Chris Bill>, <Order: order#6 courseName: CCNA StudentId: Alan George>, <Order: order#7 courseName: CompTIA Security StudentId: Alan George>, <Order: order#8 courseName: Project Management StudentId: Josh Jones>,
```

<Order: order#9 courseName: Innovation for Business StudentId: Josh Jones>, <Order: order#10 courseName: Build Responsive Websites StudentId: John Smith>, <Order: order#11 courseName: Global

Economics StudentId: Mary Jones>, <Order: order#12 courseName: Oracle SQL StudentId: Josh Jones>]>

2. Write queries to do the following.

a. List all students whose last name is 'Jones'

```
>>> Student.objects.filter(last_name='Jones')
```

<QuerySet [<Student: Mary Jones>, <Student: Josh Jones>]>

b. List all courses that for Topic 'Management'

```
>>> m_topic = Topic.objects.get(name='Management')
>>> Course.objects.filter(topic=m_topic.id)
```

<QuerySet [<Course: Innovation for Business>, <Course: Leadership Skills>, <Course: Project Management>]>

c. List all students that live on 'Sunset Avenue'.

#Here school is used instead of Address attribute as per db.

```
>>> Student.objects.filter(school__contains='Sunset Avenue')
```

<QuerySet [<Student: Mary Jones>, <Student: Josh Jones>]>

d. List all students that live on an 'Avenue' and live in 'Windsor' city.

```
>>> Student.objects.filter(school__contains='Avenue').filter(city='WS')
```

<QuerySet [<Student: John Smith>]>

e. List all the students that are interested in Topic 'Health & Fitness'

```
>>> Student.objects.filter(interested_in=hf.id)
```

<QuerySet []>

f. List the courses that cost more than $150.00

```
>>> Course.objects.filter(price__gte=150)
```

<QuerySet [<Course: CCNA>, <Course: Oracle SQL>, <Course: Microeconomics>, <Course: Introduction to Economics>, <Course: Web Development Bootcamp>, <Course: Web Design>, <Course: Innovation for Business>, <Course: Leadership Skills>]>

g. List the students that do NOT live in Windsor.

```
>>> Student.objects.exclude(city="WS")
```

<QuerySet [<Student: Mary Jones>, <Student: Alan George>, <Student: Josh Jones>]>

h. List the Orders placed by a student whose first_name is 'Chris'.

>>> Order.objects.filter(student_id=Student.objects.get(first_name='Chris').id)

<QuerySet [<Order: order#4 courseName: Microeconomics StudentId: Chris Bill>, <Order: order#5 courseName: Build Responsive Websites StudentId: Chris Bill>]>

i. List the courses that are currently NOT for_everyone.

>>> Course.objects.filter(for_everyone='False')

<QuerySet [<Course: CCNA>, <Course: AWS>, <Course: CompTIA Security>, <Course: Oracle SQL>, <Course: Microeconomics>, <Course: Introduction to Economics>, <Course: Global Economics>, <Course: Web Development Bootcamp>, <Course: Web Design>, <Course: Web Development Masterclass>, <Course: Build Responsive Websites>, <Course: Innovation for Business>, <Course: Leadership Skills>, <Course: Project Management>]>

j. Get the first name of the student of the Order with pk=1.

>>> Student.objects.get(id=Order.objects.get(id=1).student.id)

<Student: Mary Jones>

k. List all topics that the studentt with username 'john' is interested_in.

>>> Student.objects.get(username='john').interested_in.values()

<QuerySet [{'id': 1, 'name': 'Web Development', 'category': 'Development'}, {'id': 2, 'name': 'Management', 'category': 'Business'}, {'id': 3, 'name': 'Economics', 'category': 'Finance&Accounting'}, {'id': 4, 'name': 'IT certification', 'category': 'IT & Software'}]>

l. List all the courses with a price < $150 and is for_everyone.

>>> Course.objects.filter(price__lte=150, for_everyone=True)

<QuerySet []>

m. List the categories that the students who ordered a Web Dev Bootcamp is interested_in.

>>> for k in Student.objects.get(id=Order.objects.get(course_id=Course.objects.get(name='Web Development Bootcamp').id).student_id).interested_in.values():

... print(k['category'])

...

Development

Finance&Accounting

n. List the category of the topic that 'alan' is interested in. (You may assume that 'alan' is interested in exactly one topic.)

```
>>> for k in Student.objects.get(first_name="Alan").interested_in.values():
... print(k['category'])
...
Business
Finance&Accounting
IT & Software
```