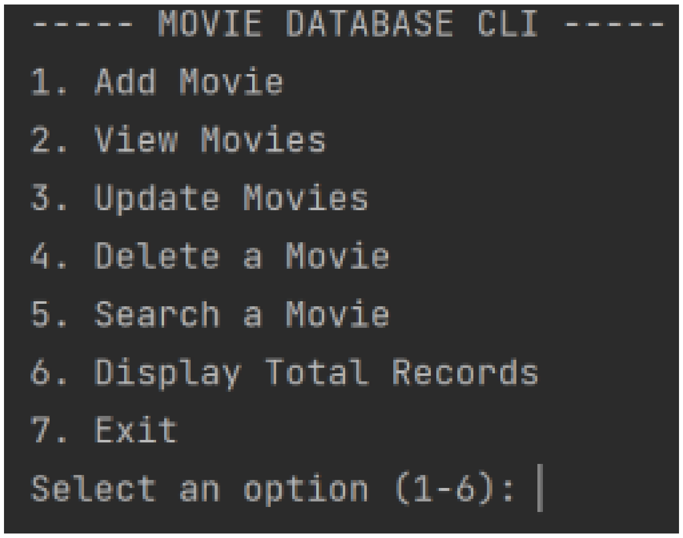
Finals Lab Task 5. CLI using Mysql and Python

1. Make sure you have installed the following pre-requisites before proceeding:
   a. Mysql-connector
   b. Mysql-connector-python
   c. Xampp is running along with Apache and Mysql in the background
2. Create the following database in Mysql;
   a. Database name: **moviesDB** with the ff: fields:

      movie_id int(10) Primary Key
      title varchar(50) NOT NULL
      main_actor varchar(50) NOT NULL
      director varchar(50) NOT NULL
      genre varchar(25) NOT NULL
      gross_sales float
      ratings (G, PG, R13, R16,X) varchar(5)

   b. Insert at_least 5 records
   c. Create a user named **test_user** and assign a **password** and give it an admin access by checking necessary SQL functions

3. Guided by the Demo code attached in this task. test_DemoDB.py 4. Kindly continue working on the code that will allow the user to navigate through the Database and perform simple CRUD operations. Follow the following **CLI Menu Options:**



```
----- MOVIE DATABASE CLI -----
1. Add Movie
2. View Movies
3. Update Movies
4. Delete a Movie
5. Search a Movie
6. Display Total Records
7. Exit
Select an option (1-6): |
```

5. The user should be able perform the ff: in your program.

MOVIE DATABASE CRUD APP

1- Add New Record
2- View all records,
3- Update a Record and show the updates,
4- Delete a record
5- Search A Record

6- Display **Total Numbers** of Movies stored in the database
7- Exit

6. For additional challenge, Task – You are to add a **SEARCH option** in the MENU that will allow the user to search by Name or emp_id, then display the information about the record being search. You may use Like statement and fetchOne method in my SQL to do this,
7. You are also going to add a method that will display the the **total number of records** in your database – You may use SQL count statement for this.
8. What to submit:
   a. UI Menu
   b. Sample Output
   c. Source Code
   d. Exported sql file

TestDb_Demo.py

```
import mysql.connector
# Connect to the SQLite database (it will create the DB file if not exists)

conn= mysql.connector.connect(
 host="localhost", # Replace with your MySQL host (e.g., IP address or hostname)
 user="root", # Replace with your MySQL username
 password="", # Replace with your MySQL password
 database="testdb" # Replace with the name of your database



cursor = conn.cursor()
```

#Insert a New Record

```
def add_employee():
 name = input("Enter name: ")
 emp_id = input("Enter employee ID: ")
 salary = float(input("Enter salary: "))
 cursor.execute("INSERT INTO employees (name, emp_id, salary) VALUES(%s,%s,%s)",
(name, emp_id, salary))
 conn.commit()
 print("Employee added successfully!\n")
 view_employees()

# View all employees
def view_employees():
 cursor.execute("SELECT * FROM employees")
 rows = cursor.fetchall()
 if rows:
 print("\nEmployees List:")
 for row in rows:
 print(row)
 else:
 print("\nNo employees found.")

# Update employee
def update_employee():
 emp_id = input("Enter employee ID to update: ")
 name = input("Enter new name: ")
 salary = float(input("Enter new salary: "))
 cursor.execute("UPDATE employees SET name=%s, salary=%s WHERE emp_id=%s",
(name, salary, emp_id))
 conn.commit()
 print("Employee updated successfully!\n")

# Delete employee
def delete_employee():
 emp_id = input("Enter employee ID to delete: ")
 cursor.execute("DELETE FROM employees WHERE emp_id=%s", (emp_id,))
conn.commit()
 print("Employee deleted successfully!\n")

#Test the methods
 if __name__ ==
'__main__':  #
add_employee()
 # update_employee()
# delete_employee()
view_employees()
```