FilmFlix Python App with database

Introduction

You have been asked to develop a command line python application for FilmFlix, to manage their film database. As a start-up FilmFlix have no more than 40 records of films in their database. A database developer volunteered to work with FilmFlix a while back to develop their database and for ease of use and integration implemented SQLite3 SQL database. However, due to unforeseen circumstance the developer is unable to connect the database with python application, but a copy of the SQLite database has been made available for you to work with if you chose to. Otherwise, you can create the database using SQLite.

Note:
CRUD: Create Read Update Delete.
Only one table in the database with records, which is all you need for this project.
Database name **filmflix.db**
Table name **tblfilms**
filmID(integer), **title**(text), **yearReleased**(integer),**rating**(text),**duration**(integer),**genre**(text)

You have now been tasked with developing a python application to manipulate the FilmFlix database.
Perform CRUD operation on the database from the python command line

1. Print all records in tblfilms in database filmflix.db
2. Allows users to add, update or delete records in the filmflix.db database (**CRUD**)
3. Print a selection of **reports**, these functions demonstrate different techniques of writing sql commands and printing reports

Hint: for FilmFlix **CRUD**: create Options menu to include
Options menu

1. Add a record	SQL INSERT INTO tblfilms(title, yearReleased, rating, duration, genre) VALUES (,,,,) - Don't provide the filmID as its AUTO_INCREMENT
2. Delete a record	SQL DELETE FROM tblfilms WHERE filmID = "film id"
3. Amend a record	SQL UPDATE tblefilms SET title = "Die Hard" WHERE title = "Di Hard"
4. Print all records	SQL SELECT * FROM tblfilms – Python to display info **for** loop
5. Reports	SQL SELECT – Give the user freedom to choose query parameters.
6. Exit	- Python exit()

Hint: for FilmFlix **report** create Options menu to include
1. Print details of all films | same as print all records
2. Print all films of a particular genre (where genre "Action")
3. Print all films of a particular year
4. Print all films of a particular rating
5. Exit

Task process breakdown

Model-View-Controller

1. Create SQL connections - Model the part of the program that handles data access
    a. Build a class to connect to the database.
    b. Test that the connections work.
    c. Write some simple fuctions to create tables, insert records, or select data into the SQL class.
2. GUI or UI (text-based) - View is the part that the user sees, feels, hears etc.
    a. How many UIs do you need? - At least 6.
    b. Decide on GUI or text-based UI.
    c. GUI
        i. Input boxes
        ii. Buttons
        iii. Radio or tick boxes
        iv. Labels to describe inputs
    d. UI
        i. Show user options: 1,2,3...
        ii. How will they choose: Type a number
        iii. Validate inputs to ensure correct datatype and range of values
    e. Test to ensure you can move between menus/UI elements
3. Controller or coordinators – Controller sends messages between the model and the view
    a. How will the (G)UI and the talk to the database?
        i. If it's a class, you need to instance the SQLConnector.
        ii. If only functions, passing parameters.
        iii. Import statements – Not just files you created but also things like random, time, etc.
    b. How will your controller tell the user if something went wrong?
        i. Error handling – TRY-EXCEPTIONS
        ii. If GUI need messageboxes
        iii. If UI then print()