

# MEDIA LIBRARY PROJECT DOCUMENTATION

## Introduction

The Media Library is a desktop application built using PyQt5, Qt Designer and PyCharm. The application allows users to create personalized movie collections. Users can search movies via OMDb API, save movie posters to local storage, add personal notes and manage their library through an intuitive GUI. The application features user authentication, responsive UI navigation and offline poster viewing for a seamless experience.

## Technologies Used

**PyQt5 with Qt Designer:** The PyQt5 library and Qt designer application were instrumental in the creation of the frontend. The combination of different widgets, buttons and line edits are what makes up the responsive GUI the Media Library possesses.

**SQLite3:** The users records and movie data were all stored in a database made using SQLite3. This library also facilitates the lookups performed on the database which enables user authentication, signup and data retrieval when needed.

**OMDb API:** responsible for handling retrieval of movie data and posters according to user requests.

**Python 3.11:** Python 3.11 served as the core programming language, orchestrating all application logic from user authentication to database operations. It enabled seamless integration of PyQt5 GUI,

SQLite database queries, and external API calls through its rich standard library and third-party packages.

Image Handling: QPixmap, requests: The requests library downloaded movie posters from OMDb API URLs and saved them locally as JPG files in the project posters folder. QPixmap then loaded these local image files into QLabel widgets, scaling them perfectly to fit the movie details screen with aspect ratio preservation.

OS: Cross-platform (Windows focus)

Designed with `os.path.join()` for path compatibility, the app runs on Windows, Mac, and Linux by using platform-agnostic file handling. Windows-specific raw strings (`r"C:\path\to\posters"`) ensured reliable poster storage and retrieval on the target development platform.

File System: Posters are downloaded once from OMDb and permanently saved to MEDIA LIBRARY PROJECT/posters/ with filenames derived from movie titles (e.g., `forrest_gump.jpg`). This enables instant offline loading via full file paths stored in the database's PosterPath column, eliminating repeated API calls.

## Key Libraries

- PyQt5.QtWidgets - UI components
- PyQt5.uic - UI file loading
- sqlite3 - Database operations
- requests - API calls & poster downloads
- os.path - File path management
- re - Input validation

## Core Functionality

- User Authentication: Secure login/signup with regex validation
- Movie Search: Title/Genre/Year/Rating search
- OMDb Integration: Real-time movie data fetching
- Local Poster Storage: Downloads posters to MEDIA LIBRARY PROJECT/posters/
- Personal Notes: Save/delete notes per movie
- Responsive Navigation: Login → Splash → Dashboard → Details

## Database Schema

Users Table:

- User\_ID (Primary Key, Autoincrement)
- First\_name, Last\_name
- Gender, Dateofbirth
- Email (UNIQUE), Password

Movies Table:

- Movie\_ID (Primary Key, Auto)
- User\_ID (Foreign Key)
- Title, Genre, Year, Runtime, rating
- Plot, Actors, Director
- PosterPath (local file)
- Notes (personal)

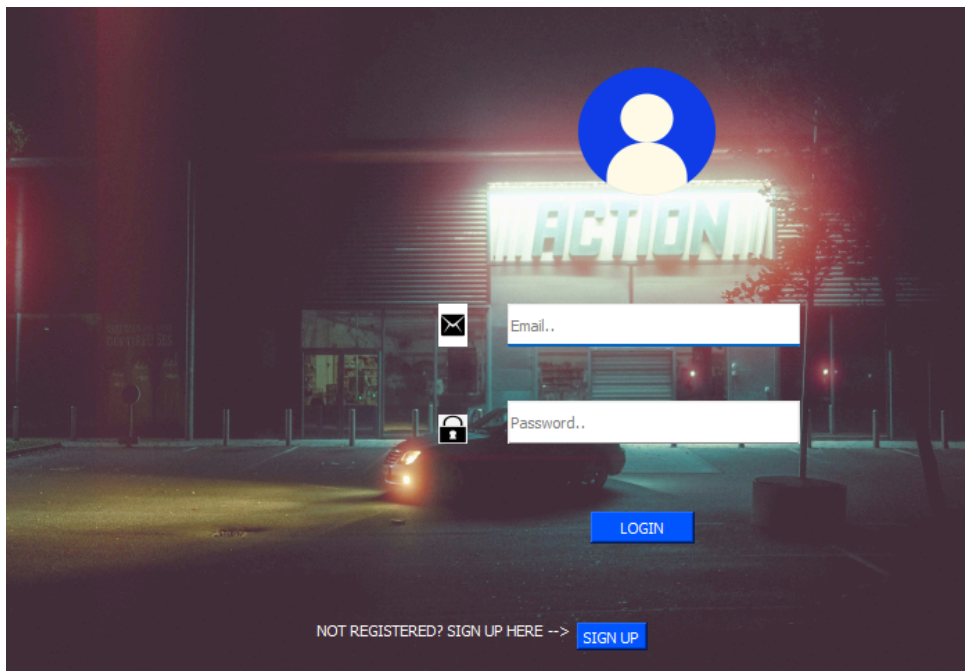
## UI Screens

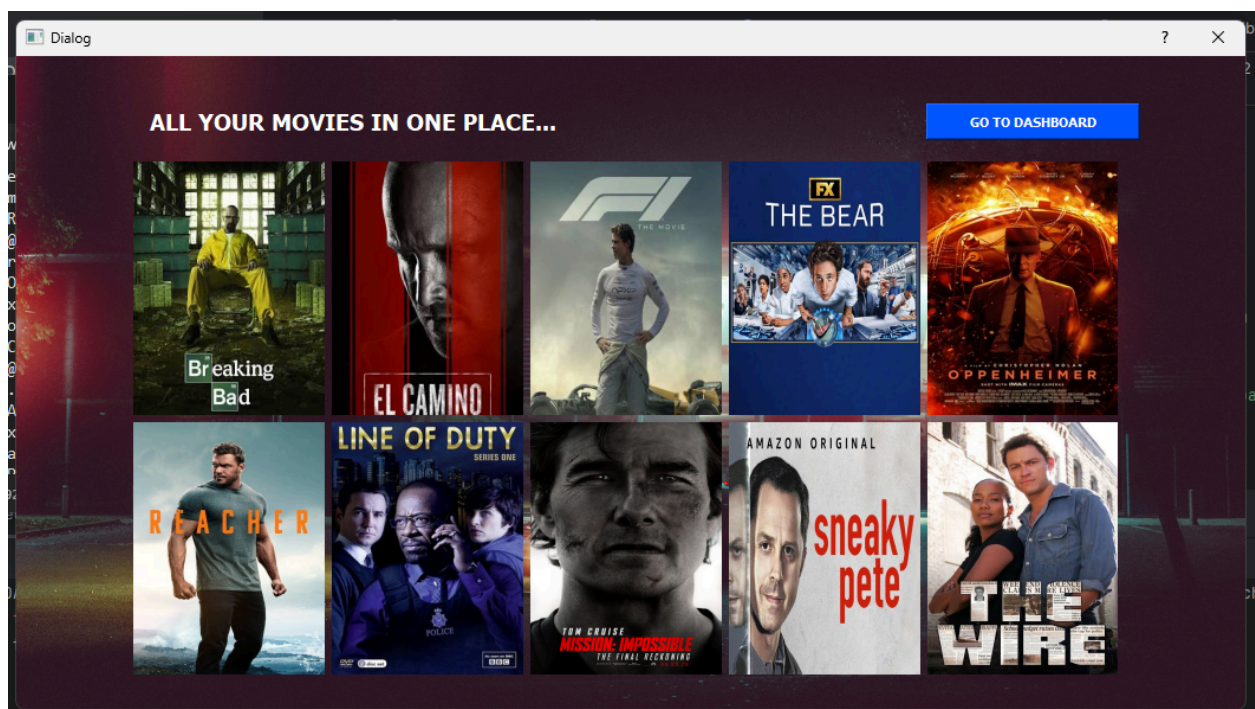
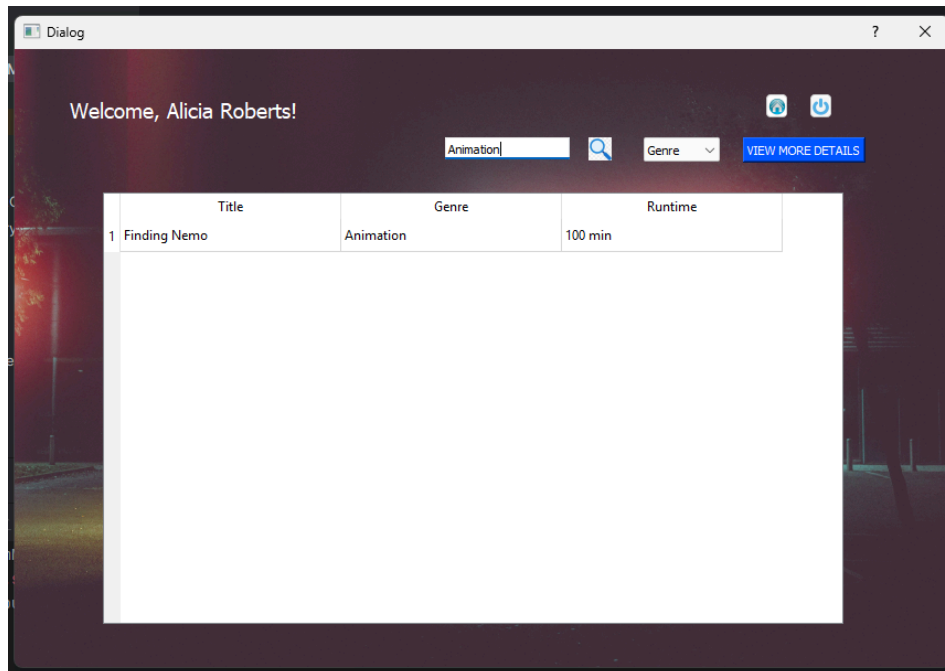
1. Login/Signup - Frameless modern design
2. Splash Page - Welcome animation
3. Dashboard - Search table + navigation
4. Movie Details - Poster, info, notes, CRUD buttons

## Challenges Faced

1. QTextEdit vs QLabel for Plot: In the Movie Details screen, QLabel was initially used to display plot data for the selected movie. However, QLabel could not handle long texts, and movie plots typically range from 100-400 words, so QTextEdit was employed instead as it is designed for longer content.
2. Poster Path Management: Managing the file path to store downloaded posters locally on the desktop was challenging. Upon discovering the `os.path.join()` method, this became feasible as it automatically generates the correct file path for the poster across different operating systems.

## Screenshots





## **Conclusion**

The Media Library GUI application successfully delivers a production-ready movie management system with modern desktop features.