🎬 Media Library Management System

A Java Swing Desktop Application for managing, searching, and reviewing Movies and TV Series.
This application demonstrates Object-Oriented Programming (OOP) principles, file persistence (Serialization), and a layered architecture (Model-View-Service).

Overview

The Media Library Management System allows users to browse a collection of media items (Movies and Series).
It differentiates between Subscribers, who consume content and write reviews, and Admins, who manage the content library.
The application features a graphical user interface (GUI) built with Java Swing.

Features

1. General

Authentication: Secure Login and Registration system.
Search Engine: Advanced filtering by Title, Actor, Category (Action, Drama, Comedy, etc.), Rating, and suitability for underage viewers.
Data Persistence: All data (users, items, reviews) is saved locally using Java Object Serialization (.dat files).

2. Subscriber Capabilities

Browse & Search: View details of movies and series.
Favorites: Add or remove items from a personal "Favorites" list.
Reviews: Write, edit, and delete personal reviews and ratings for media items.
Recommendations: See "Relevant Items" (similar movies/series) linked to the current selection.

3. Admin Capabilities

Content Management: Create, Update, and Delete Movies and Series.
Complex Data Entry:
  Movies: Set duration, year, cast, and link relevant movies.
  Series: Manage a tree structure of Seasons and Episodes (duration per episode).
Review Management: Admins can oversee content (logic allows for expansion here).

Architecture

The project follows a layered architecture to separate concerns:

1. Model (api.model): Contains the data classes (POJOs).
   Abstract base class MediaItem extended by Movie and Series.

User hierarchy: Admin and Subscriber.
Helper models: Season, Episode, Review, Category (Enum).

2. View (gui): Contains all Swing JFrame and JPanel classes responsible for the UI (e.g., Login, SearchPanel, ShowItemFrame).
3. Service/Controller (api.service): Handles business logic and data manipulation.
   UserService: Handles registration and login logic.
   SearchService: Filters lists based on user criteria using Java Streams.
   MediaItemFileHandler & UserFileHandler: Manage reading/writing data to files.

Data Persistence

The application uses Java Serialization to store data.
items.dat: Stores the list of all MediaItem objects (Movies/Series).
users.dat: Stores the list of all User objects (Admins/Subscribers).

Note: The Main.java class contains an initData() method to pre-populate the database with sample data (e.g., Harry Potter, The Office, The Crown) for testing purposes

Installation & Usage

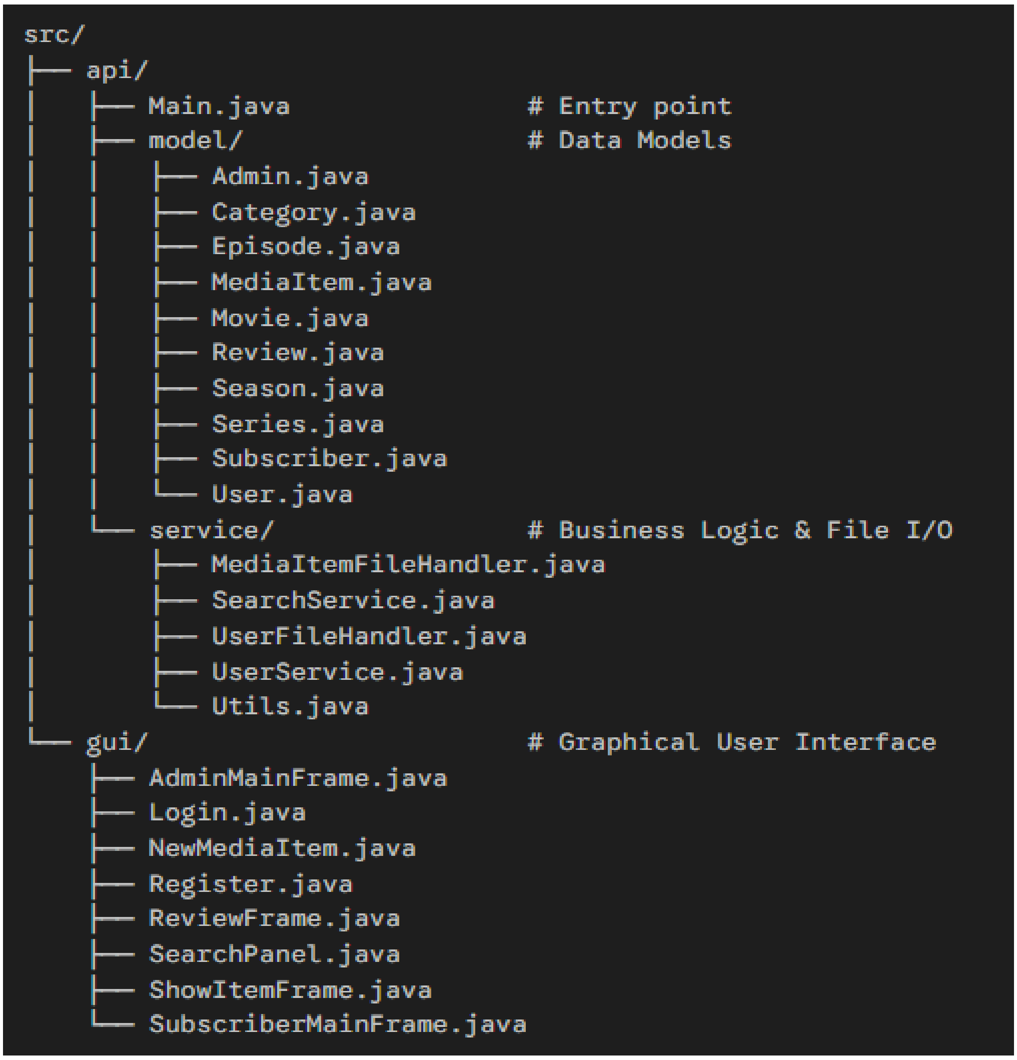
Prerequisites
Java Development Kit (JDK) 8 or higher.
An IDE (IntelliJ IDEA, Eclipse, or NetBeans).

How to Run

1. Clone the repository.
2. Open the project in your IDE.
3. Locate api.Main.java.
4. First Run: Uncomment the initData(); line inside main() to generate the initial .dat files with sample data.
5. Run the main method.
6. Subsequent Runs: Comment out initData(); to persist your changes.

Default Credentials (from Sample Data)
Admin: Username: admin1, Password: password1
Subscriber: Username: user1, Password: password1

Project Structure



```
src/
├── api/
│   ├── Main.java               # Entry point
│   ├── model/                  # Data Models
│   │   ├── Admin.java
│   │   ├── Category.java
│   │   ├── Episode.java
│   │   ├── MediaItem.java
│   │   ├── Movie.java
│   │   ├── Review.java
│   │   ├── Season.java
│   │   ├── Series.java
│   │   ├── Subscriber.java
│   │   └── User.java
│   └── service/                # Business Logic & File I/O
│       ├── MediaItemFileHandler.java
│       ├── SearchService.java
│       ├── UserFileHandler.java
│       ├── UserService.java
│       └── Utils.java
└── gui/                        # Graphical User Interface
    ├── AdminMainFrame.java
    ├── Login.java
    ├── NewMediaItem.java
    ├── Register.java
    ├── ReviewFrame.java
    ├── SearchPanel.java
    ├── ShowItemFrame.java
    └── SubscriberMainFrame.java
```