Movie and Book Recommendation Desktop App
Software Design Document

Table of Contents

1. Introduction

The Movie and Book Recommendation Desktop App is a software application designed to supply personalized movie and book recommendations to users based on keyword searches. This document outlines the design and architecture of the desktop application, supplying a comprehensive view of its components, functionalities, and technical specifications.

Link to the figma project

2. Purpose

The purpose of this document is to supply a detailed description of the software design for the Movie and Book Recommendation Desktop App. It serves as a reference for developers involved in the project, ensuring a mutual understanding of the system's architecture and functionality.

3. Scope

- The Movie and Book Recommendation Desktop App will include the following key features:
    - A simple user registration and log-in.
    - Keyword-based search for movies and books.
    - Filtering search results by genre and release year.
    - Detailed information about the books and movies, including ratings.

    - Integration with External APIs (TMDb API, Google Books API)

4. Architecture Overview

The application will follow the MVC architecture, which includes:

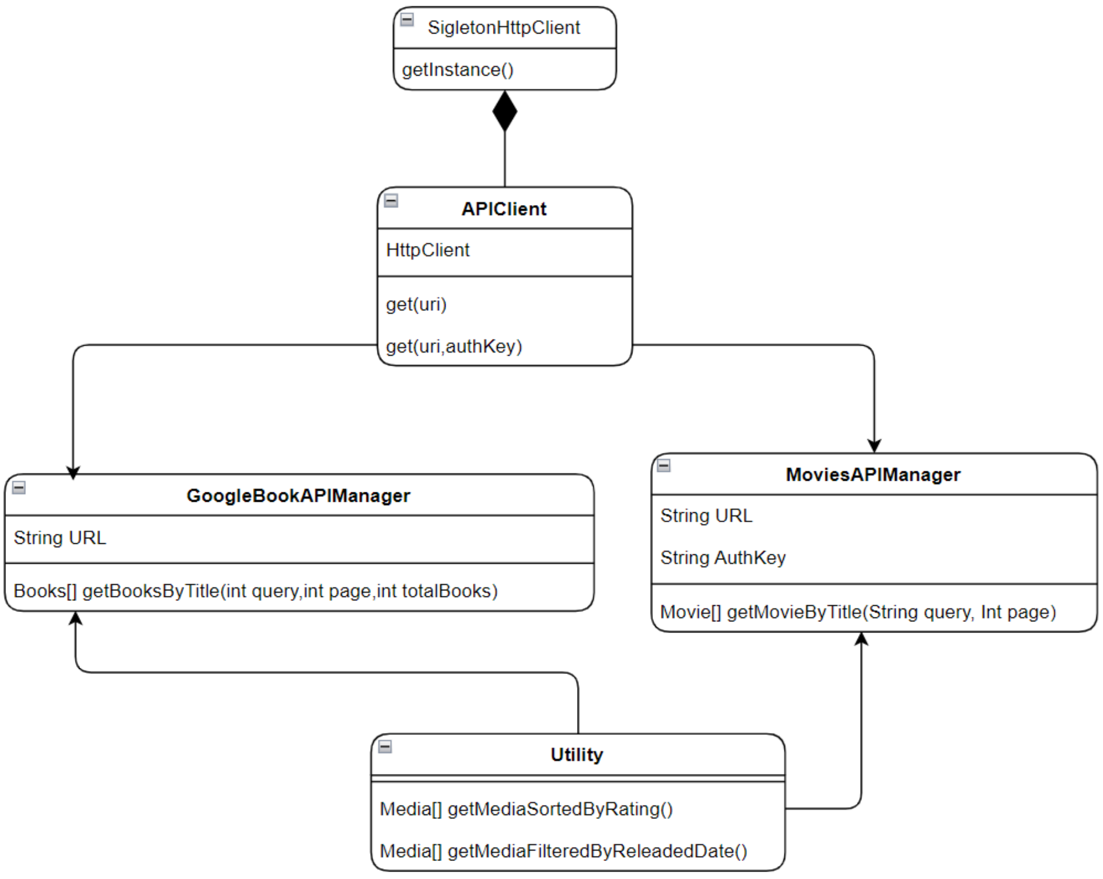
- **View**: The View stands for the user interface (UI) of the application. It displays data from the Model to the user and sends user commands to the Controller.
    - **Responsibilities**:
        - Display movie and book information in a user-friendly manner.
        - Supply UI elements for user interactions, such as search bars, filters, and detailed view pages.
        - Reflect any changes in the Model to ensure the displayed data is always current.
- **Controller**: The Controller acts as an interface between Model and View. It takes the user input from the View, processes it (with possible updates to the Model), and returns the display output to the View.
    - **Responsibilities**:
        - Handle user input from the View, such as keyword searches or filter selections.
        - Interact with the Model to retrieve or update data based on user input.
        - Update the View to reflect changes in the Model or to display new data to the user.
- **Model**: The Model stands for the data and the business logic of the application. It directly manages the data, the logic, and the rules of the application.
    - **Responsibilities**:
        - Communicate with external APIs (TMDb API, Google Books API) to fetch updated movies and book data.
        - Process data according to business rules, such as filtering search results based on genre and release year.
        - Notify the View of any changes to the data to ensure the user interface is up to date.

5. System Components

The major components of the Movie and Book Recommendation Desktop App include:

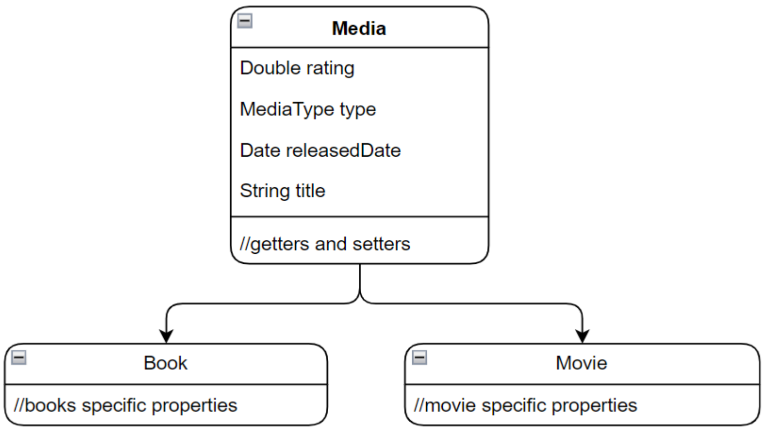
- **Search Engine**: Performs keyword-based searches and retrieves relevant movie and book data.

- **External API Integrations**: Connects with external sources to fetch updated movie and book information.

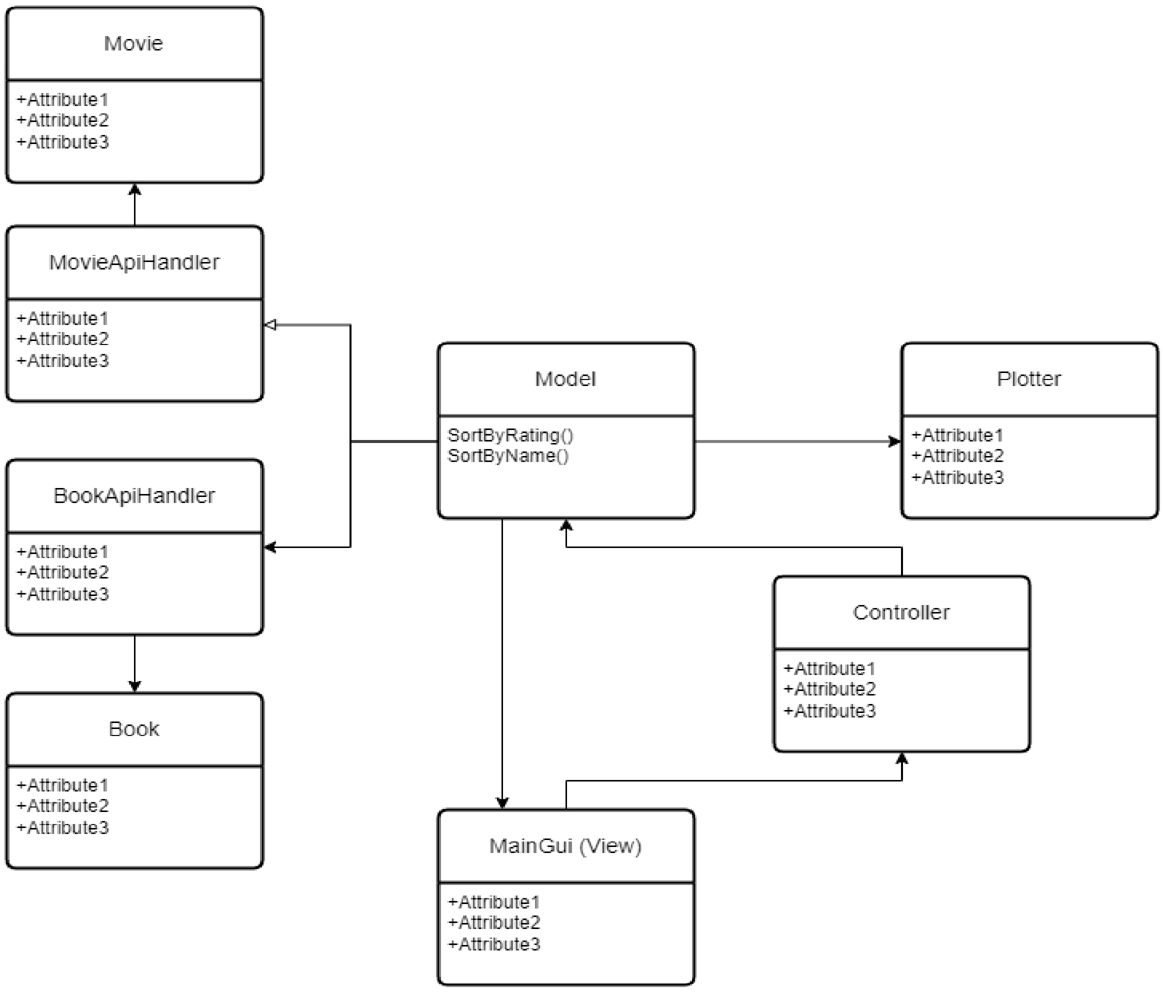


SigletonHttpClient
getInstance()
APIClient
HttpClient
get(uri)
get(uri,authKey)
GoogleBookAPIManager
String URL
Books[] getBooksByTitle(int query,int page,int totalBooks)
MoviesAPIManager
String URL
String AuthKey
Movie[] getMovieByTitle(String query, Int page)
Utility
Media[] getMediaSortedByRating()
Media[] getMediaFilteredByReleadedDate()

- **Book and Movie Classes:** These will be the books and movies in the system



- **Overall structure of the app**



6. User Interface

The desktop application will provide a user-friendly graphical interface for users to interact with the system. Key features will encompass:

- **Keyword Search Bar**: A search bar that allows the user to search for books and movies through keywords.
- **Search Results Display**: Displays the 20 first results retrieved from the APIs. 10 books and 10 movies. More can be displayed if the user wishes so.
- **Sorting by Rating**: Users will have the option to sort search results by movie or book ratings, allowing them to quickly find the most highly rated content.
- **Filtering by Release Date**: Users will be able to apply filters to search results based on the release date of movies or books, refining their results to include content released within a specified period.

- **Summary Analysis with Bar Graphs**: Users will have the ability to visualize the summary of their search results using bar graphs. These graphs will supply visual insights into factors such as ratings, release dates, and other relevant data, enabling users to make more informed decisions about their selections.
- Movie and Book Details

7. Search Algorithm

The search engine will use a combination of keyword matching to return relevant movie and book results. It will consider factors such as title, description and release year.

After the result has been fetched from the API sources it will be processed according to filter and sort selection made by the user and then it will be displayed.

There are few differences between API responses which we need to consider while displaying the search result.

For e.g., Movie API returns rating out of 10 while Books API returns rating out of 5 as a result, we need to map the ratings on correct scale to sort the result in terms of ratings.

7.1 Google Books API

Performing a search

You can perform a volume/book search by sending an HTTP GET request to the following URI:

https://www.googleapis.com/books/v1/volumes?q=search+terms

This request has a single required parameter:

- q - Search for volumes that have this text string. There are special keywords you can specify in the search terms to search fields, such as:

Request

Here is an example of searching for Daniel Keyes' "Flowers for Algernon":

GET
https://www.googleapis.com/books/v1/volumes?q=flowers+inauthor:keyes&key=API Key

Response
If the request succeeds, the server responds with a 200 OK HTTP status code and the volume results:

```
{
 "kind": "books#volumes",
 "items": [
  {
   "kind": "books#volume",
   "id": "_ojXNuzgHRcC",
   "selfLink": "https://www.googleapis.com/books/v1/volumes/_ojXNuzgHRcC",
   "volumeInfo": {
    "title": "Flowers",
    "authors": [
     "Vijaya Khisty Bodach"
    ],
   ...
  },
  {
   "kind": "books#volume",
   "id": "RJxWIQOvoZUC",
   "selfLink": "https://www.googleapis.com/books/v1/volumes/RJxWIQOvoZUC",
   "volumeInfo": {
    "title": "Flowers",
    "authors": [
     "Gail Saunders-Smith"
    ],
    ...
  },
 ],
 "totalItems": 3
}
```

7.2 TMDb API

Performing a search
You can perform movie search by sending an HTTP GET request to the following URI:

https://api.themoviedb.org/3/search/movie?query=moviename&page=number

This request has a single required parameter:

- query - Search for movies that have this text string.:

Request
Here is an example of searching for “horror” :

GET https://api.themoviedb.org/3/search/movie?query=horror&page=1

Response

If the request succeeds, the server responds with a 200 OK HTTP status code and the movie results:

```
{
 "page": 1,
 "results": [
  {
   "adult": false,
   "backdrop_path": "/q0q0nISnX4Z8V2wIfhIYfzaVX3b.jpg",
   "genre_ids": [
    18,
    9648,
    27,
    53
   ],
   "id": 301325,
   "original_language": "en",
   "original_title": "#Horror",
 },
  ...
 ],
 "total_pages": 58,
 "total_results": 1142

}
```

8. Performance

Performance improvements will be made as necessary to ensure reasonable efficiency and ease of use.

9. Conclusion

This Software Design Document provides an overview of the architecture and functionality of the Movie and Book Recommendation Desktop App. It serves as a reference for the development and maintenance of the application, ensuring that it meets the needs and expectations of its users and stakeholders in a desktop environment.