COMP.SE.110 Software Design

Group Assignment Design Document

Link to the prototype

GROUP JMSK
Jani Ilkka, jani.ilkka@tuni.fi Markus Siitonen, markus.siitonen@tuni.fi Sviatoslav Vasev, sviatoslav.vasev@tuni.fi Kian Moloney, kian.moloney@tuni.fi

Document version history	
Document created	14.9.2024 (Jani Ilkka)
Added movie functionality-related documentation and a diagram	19.9.2024 (Jani Ilkka)
Updated Movies-table and the diagram	29.9.2024 (Markus Siitonen)

Table of Contents

Introduction

Application Idea

Our application is called Movie Night Planner.

Movie Night Planner allows users to select a movie to watch based on a search criteria. The application then tells which streaming service(s) has/have the movie in their catalog.

BONUS: As a potential addition to the aforementioned functionality, we have an idea about using a third-party API to get recommendations as to which snacks go with the theme of the movie.

Schedule and Division of Work

Course week	Tasks	Deadline	Responsible
1	Post a group formation message.	27.8.	Jani Ilkka, Markus Siitonen

2	Choose application idea.	10.9.	Whole team
3	Pivot the application idea.	18.9.	Whole team
4	First Prototype Views	20.9.	Markus Siitonen, Sviatoslav Vasev, Kian Moloney
5	JavaFX application creation, TMDB-related Java code, including relevant documentation.	20.9.	Jani Ilkka
6	Additional TMDB java code added. Documentation updated.	25.9.	Jani Ilkka
7	TA Meeting	1.10.	Whole team

Central Features

TBD

User Guide

TBD

APIs Utilized

API	**Functionality**
The Movie Database (Getting Started (themoviedb.org)	Provides movie and tv-show data.
Movie of the Night (Shows - Streaming Availability API Reference (movieofthenight.com))	Lists streaming services that can be used to watch the movie/tv-show selected in the first phase. We use this to get subtitle information. This data is not available on TMDB.
BONUS: Spoonacular (spoonacular recipe and food API)	Provides snack and recipe information. We search the Spoonacular database to get suggestions based on the movie theme.

Classes, Interfaces, and Their Responsibilities

Movies

Movies (TMDB API)		
Views	SearchView	Start view, initially showing a list of most popular movies in the database. Enables searching for movies based on a search criteria.
	MovieDetailsView	Displays detailed data about a selected movie.
Controllers	MovieDataController	Houses Application logic for all movie-related functionalities.
Models	Genre	Represents movie genre.
	GenresResponse	Response data from the "get all genres" API. (https://api.themoviedb.org/3/genre/movie/list)
	Movie	Non-detailed data about a single movie. Used by PopularMoviesResponse and MovieSearchByTitleRespose.
	MovieDetails	Detailed data about a single movie.
	MovieDetailQueryResponse	Response data for a data request about a single movie. (https://api.themoviedb.org/3/movie/{movie_id})
	PopularMoviesResponse	Response data for the API returning data about the currently most popular movies. (https://api.themoviedb.org/3/movie/popular)
	ProductionCompany	Represents production company information. (Inside MovieDetails.)
	ProductionCountry	Represents production country information. (Inside MovieDetails.)
	SpokenLanguage	Represents spoken language information. (Inside MovieDetails.)
	MovieSearchByTitleResponse	Response data for movie title-based search. (https://api.themoviedb.org/3/search/movie)
	StreamingProvider	Holds the information of a streaming provider
	StreamingResponse	Response data for a List of Streaming Providers (https://api.themoviedb.org/3/watch/providers/movie?language=en-US&watch_region=FI)

MovieDataController

Method	Functionality
getPopularMovies()	Returns a list of currently most popular movies.
getMovieDetails(movieID)	Returns detailed movie data.
searchMoviesByTitle(title)	Returns a movie list based on a title search string.
getAllGenres()	Returns a list of all available genres.

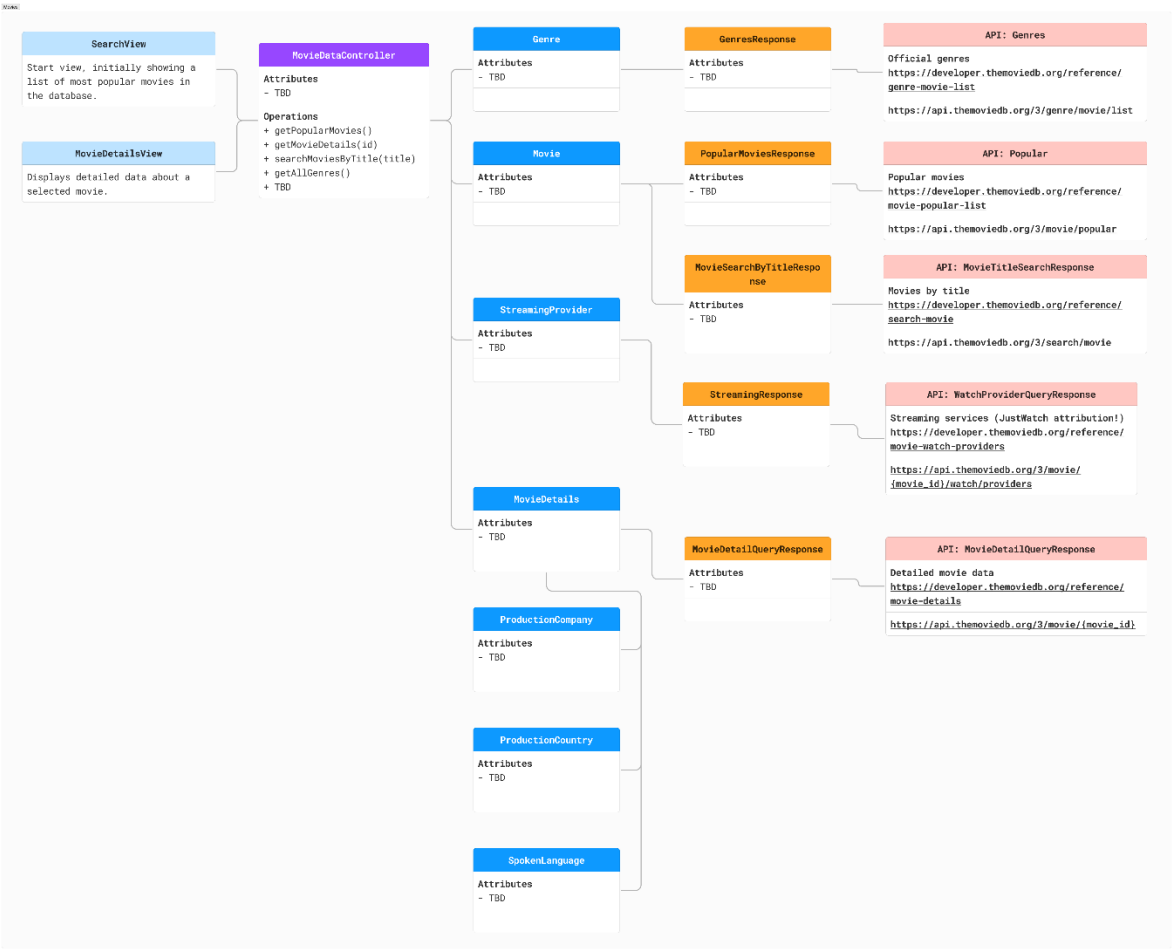

Figure 1 Movie functionality

Subtitles

NOTE! This is based on theoretical observations only. All must be verified after the creation of actual Java classes!

Subtitles (Movie of The Night API)		
Controllers	ShowDataController	
Models	Show	
	StreamingOptions	
	CountryCode	
	Service	
	Subtitles	

ShowDataController

Other

Application State and User Data		
Views	ProfileView	Displays user-specific data, including view history. Previous viewing data is used to display a collection of charts and figures.
Controllers	UserProfileController	Application logic for user information, search history, and application state management.
Models	User	
	History	
	ApplicationState	

BONUS: Snacks

NOTE! This is based on theoretical observations only. All must be verified after the creation of actual Java classes!

Snacks (Spoonacular API)		
Views		
Controllers		
Models		

Architecture

Architecture Patterns

The selected architecture pattern is MVC.

We spent considerable time researching[iiiiiiiv] and discussing this choice. As JavaFX does not dictate using a certain pattern, it boils down to what suits the application best and what are the preferences of the development team.

The application itself is rather simple. A classic, clear division between the view, application logic, and the model was deemed sufficient. We opted for a version, where the models are as lightweight as possible. In fact, they only represent data with appropriate getters and setters.

The length of previous experience and amount of knowledge of using architecture patterns varied greatly within the development team – starting with no experience and ending with professional work experience with multiple patterns. This impacted the selection process in favor of a simple pattern.

Design Patterns

TBD

Implementation Technologies

Below you'll find explanations for a few key technology selections.

Views

The user interface is implemented using JavaFX. More specifically, we opted for the XML-based approach whereby a baseline implementation of each view is created using Gluon Scene Builder. This will be later enhanced using the declarative Java approach, as needed due to view dynamics.

HTTP Client

We use Apache HttpClient (version 5). (Apache HttpComponents – HttpClient Quick Start)

AI Disclosure

Various AI services were used with several tasks during the project, as detailed below.

Idea

We used several prompts to create dozens of application ideas. The idea that was ultimately chosen was from ChatGPT. The chosen prompt-idea combination is:

"Suggest ideas for a programming group project in an university software design class. The application should use and combine data from at least two open REST application programming interfaces."

Movie Night Planner

- **Description**: A collaborative movie recommendation platform where users can select a movie and receive recommendations for streaming services and snacks.

- **APIs to Use**:
    - **The Movie Database (TMDb) API**: For fetching movie information and ratings.
    - **Utelly API**: For determining which streaming platforms have the selected movie.
    - **Spoonacular API**: For suggesting themed snacks and drinks based on the movie genre.

It was later revealed that we do not need Utelly API at all, but we actually need to get subtitle information from somewhere. Thus, we dropped Utelly and started to use the Movie of the Night API.

Code

All initial Java model class files were created by Copilot. For each API, we requested Copilot to create a Java class that corresponds to the JSON data that each individual API returns. The Author-tag in such Java classes include "Copilot", in addition to at least one human name.

The draft level Java code was refined by team members. This means adjusting property visibility settings and, e.g., refactoring the Movie-class out of two API responses into a class of its own.

Graphic Design Resources

All app logos, background images etc. Were originally created using DALL-E.

References

[i] Design pattern to use : r/JavaFX (reddit.com)
[ii] Implementing JavaFX Best Practices | JavaFX 2 Tutorials and Documentation (oracle.com)
[iii] Best practices for JavaFX architecture and patterns? - Oracle Forums
[iv] Implementing MVC in JavaFX - PragmaticCoding