Final Project Documentation

Created by: Hrvoje Buljan (RIT Student)  
Date: 10.12.2023

# Project Overview

## Source Code

Frameworks and Libraries: Utilizes SwiftUI for the user interface, and the ActivityIndicatorView library for loading animations.

Resources: Incorporates images to enhance the user experience.

Code Repository: Available on myCourses, ensuring easy access .

## Presentation Summary

Brief Description: A Swift-based iOS application focused on movie discovery, integrating real-time data from the TMDb API to provide personalized recommendations and search functionality.

Challenges Faced: Balancing the complexity of API integration with the need for a seamless user experience was a significant challenge. Overcame this by adopting Swift's best practices and utilizing SwiftUI's robust features.

Third-party Libraries: ActivityIndicatorView for engaging loading animations; TMDb API for fetching up-to-date movie data.

# Product Definition Statement

The app aims to revolutionize how users discover movies on their iOS devices. It offers a unique combination of personalized recommendations, mood-based selections, and a comprehensive movie database, all wrapped in an intuitive and engaging user interface.

# Features Implemented

1. Movie Listing: Dynamically displays movies with details fetched from the TMDb API.

2. Personalized Recommendations: Suggests movies based on user preferences and viewing history.

3. Mood Selector: Offers movie suggestions tailored to the user's current mood.

4. Search Functionality: Enables users to efficiently search for movies by name.

5.The Detail Screen: should display the following data (this can be changed based on the API): title, poster, release date, duration, genre, directors, cast, ratings (IMDb), trailer.

# Wish List for Next Version

1. Dark Mode: To enhance visual comfort and accessibility.

2. Advanced Animations: To improve user engagement and interface dynamism.

# Self Evaluation and Documentation

Grade: A

Justification: The project successfully implements MVC architecture, ensuring a robust and scalable application. It excels in providing a seamless user experience, dynamic content integration, and adhering to best software development practices.

# Class Descriptions and Architecture

**ViewModel Classes:**

-Constants.swift - Centralizes all the constant values such as API keys and URLs, facilitating their management and usage.

-MovieDetailsViewModel.swift - Manages the fetching and processing of detailed information for individual movies, including error handling.

HomeViewModel.swift - Handles data for the home screen, managing the display of various movie categories.

MovieSearchViewModel.swift - Responsible for the search functionality, updating search results, and managing error states.

RecommendationViewModel.swift - Manages the logic for generating personalized movie recommendations based on user preferences.

**Model Classes:**

Movie.swift - Defines the data structure for movies, including properties for title, genres, and release date.

Genre.swift - Represents the genre of movies, aiding in categorizing and filtering movies.

Movies.swift - An alternative structure for movie data, used for simpler views or lists.

ErrorResponse.swift - Structures API error responses for effective error handling.

DiscoverResponse.swift - Manages responses from movie discovery API calls, important for handling paginated movie data.

**View Classes:**

HomeView.swift - The main interface of the app, providing navigation and layout for different sections like home, search, mood selection, and recommendations.

FeaturedMoviesView.swift - Showcases featured movies in an interactive carousel format.

MovieDetailsView.swift - Displays detailed information about a selected movie.

MoviesListView.swift - Presents a list of movies based on different categories or sorting criteria.

SearchView.swift - Implements a search interface with real-time updating of search results.

MoodSelectorView.swift - Allows users to select movies based on their mood.

WelcomeView.swift - Offers an introductory questionnaire to new users for setting up preferences.

RootView.swift - Decides whether to display the welcome view or the main app interface.

**ContentView.swift -** The primary container for the app's content, managing the initial loading state.

**FinalProjectApp.swift -** The main entry point of the application, initializing the ContentView.

# Installation Instructions

Detailed steps to install the app on iOS devices will be provided. In the case of remote presentation, a screen recording demonstrating the app's functionality will be available.

# Alignment with Grading Criteria

Scope: The project meets all defined objectives and user expectations, focusing on a key functionality—movie discovery.

Dynamic Data: Integration with TMDb API ensures real-time, dynamic data.

Software Design Patterns: Effectively uses MVC, Delegation, and other patterns for a robust architecture.

Software Engineering Practices: Adheres to DRY principles and effectively utilizes enums and defines to create clean, maintainable code.

UI/UX Design: The application makes appropriate use of SwiftUI components like NavigationView, TabView, and includes animations for a polished interface.

User Experience: Designed for optimal iPhone usage, including state saving and offline functionality.