**Catch Movie!**

SSP8 - Group 4

*Casuarina, Dobrasiewicz Piotr, Godeliva Petrina M, Tan Jian Wei, Zhang Yuhan*

**Agenda**

● Introduction to Catch Movie

● Requirements Elicitation & Analysis

● Conceptual Models: Architecture & Class Design

● Dynamic Models: Sequence Diagram & Dialog Map

● Testing: Black box & White box

● Application Live Demo

● Learning Points & Challenges

● Possible Future Extensions

**Introduction to Catch Movie**

**About Catch Movie**

Provide a **unified platform** for users to **view movie information and book movie tickets from various cinemas**: Cathay, Golden Village and Shaw.

**Value Proposition**

● **Quick search and comparison** of movie schedules in one app

● **Eliminate trouble of opening multiple** cinema websites

● Convenient way to **find showtime that best suits our needs from today to next 4 days Product Scope & Targeted Audience**

The application is mainly targeted for people in Singapore as it utilizes publicly available movie data in Singapore only

**Product Functions**

● **Login** via gmail account

● Retrieving movie information of **Now Showing and Upcoming movies compiled across different cinemas** in Singapore:

○ **Movie details** (Synopsis, rating, casts etc)

○ **Showtimes** (Today to next 4 days)

● **Direct search** of movie information by providing **movie title and movie date as input**

● **Direct booking of movie tickets** via link to official cinema booking page

● Retrieving **ongoing promotions across different cinemas** in Singapore

● Viewing **support/basic FAQ** regarding how to use the application

● **Submitting rating** for the application

**Requirements Elicitation & Analysis**

**Requirements Design**

Take *ViewNowShowing* use case as example

● **Atomized** requirements

● Logically **structured** requirements

→ **Easily tested & verifiable** requirements

**Use Case Diagram**

***Initiating actor:*** *User*

***Participating actor:***

*● Firebase Authentication: Authenticates user account via gmail*

*● JSON Storage: Stores web scraping data*

*● Firebase Realtime Database: Stores users’ submitted ratings*

**Conceptual Models**

**Architecture Design: MVC**

*Defines presentation to user*

*Manages user interaction and captures user input*

*Contains data involved*

**Architecture Design Benefits**

● Separation of Model from View and Controller → allows **multiple views of same model**

○ Ex: Now Showing UI, Upcoming UI, Search UI all use same model but different presentation

● Views are **not tightly coupled** with core functionalities

○ Can **easily change UI design**

● **High cohesion** due to **logical grouping of related actions** together

○ Each View has its own Controller and grouped together with its specific model

**Class Diagram**

**Class Design Considerations**

● **Single Responsibility** → One class one responsibility/functionalities

○ Example:

● **Open-closed principle** → Open for extension but closed for modification

○ Example: Parent class StoreFragment with sub-class MovieFragment. StoreFragment is closed for modification but is open for extension by new subclasses in the future

● Implement **DAO pattern** → hide from the application logic the complexities involved in performing operations to database

○ Example: RatingDAO implements CatchMovieDAO to store ratings submitted by users to Firebase

○ Easy to add new database implementation (MySQL etc) for new DAO in the future

**Dynamic Models**

**Sequence Diagram: ViewNowShowingInfo**

**Dialog Map**

**Application Testing**

**Black Box Testing**

**Test Priority** (Low/Medium/High): High

**Test Title**: Movie Details

**Description**: To test movie details of a specific movie

**Steps Test Steps Test Data Expected Results Actual Results Status (Pass/Fail)**

1 User clicks on Movie logo on bottom

navigation

A list of now showing movies will be shown. The list contains an image of the movie and the movie title.

A list of now showing movies will be shown. The list contains an image of the movie and the movie title.

A list of now showing movies will be shown. The list contains an image of the movie and the movie title.

A list of now showing movies will be shown. The list contains an image of the movie and the movie title.

Pass

Pass

Pass

2 User clicks on a movie from the list *“Maleficent”* Movie details of

*“Maleficent”* will be shown: rating,language, synopsis, director, cast, duration and NA if not available. There will be a “VIEW SHOW TIMES” button at the bottom of the page for user to click.

Movie details of *“Maleficent”* will be shown: rating,language, synopsis, director, cast, duration and NA if not available. There will be a “VIEW SHOW TIMES” button at the bottom of the page for user to click.

Pass

Pass

**White Box Testing: Login**

**White Box Testing: SearchMovie**

**Application Live Demo**

**Learning Points**

● Get hands on experience on SDLC: Requirement elicitation, requirements analysis, conceptual & dynamic models, implementation and testing

● Learn to apply software engineering practices to develop Android application instead of normal web application

● Get the chance to develop an application that is useful in daily life, especially in Singapore **Challenges**

● Little to no experience in developing Android application

● Sometimes found it difficult to integrate codes since everyone is involved in code implementation

● Need to continuously update/refine design models

**Thank you! :)**