

City University of Hong Kong

Department of Computer Science

CS3343 Software Engineering Practice

2024 - 25 Semester A

**CityU Cinema System**

**Project Plan**

**Group 2**

**Name SID**

CHAN Ka Hong 57845658

FUNG Wing Ho 56755919

LAW Lok Hang 57842515

NG Ka Ho 56652261

WONG Tsz Kin 57847086

Contents

[1. Project Plan 3](#_Toc183977959)

[1.1 Abstract 3](#_Toc183977960)

[1.2 Objectives 3](#_Toc183977961)

[1.3 Scope 4](#_Toc183977962)

[1.4 Constraints 5](#_Toc183977963)

[2. Project Team Organization 6](#_Toc183977964)

[3. Methodology 7](#_Toc183977965)

[4. Development Tools 8](#_Toc183977966)

[5. Configuration Management 9](#_Toc183977967)

[5.1 Version Control 9](#_Toc183977968)

[5.2 Build Management 9](#_Toc183977969)

[5.3 Change Control 9](#_Toc183977970)

[6. Work Breakdown Structure 10](#_Toc183977971)

[7. Team Gantt Chart 10](#_Toc183977972)

[8. Future Plans 11](#_Toc183977973)

[8.1 A Multi-threaded Design 11](#_Toc183977974)

[8.2 A Web Application 11](#_Toc183977975)

[8.3 More Movie Scheduling Strategies 11](#_Toc183977976)

# 1. Project Plan

## 1.1 Abstract

Watching movies at cinemas was once a popular leisure activity welcomed by many people. However, with the rise of many online streaming platforms, some people may have appeared to let go of this enjoyable entertainment and perhaps find themselves not doing any movie ticket ordering activities for some time already. Therefore, we propose to create the Movie Ticketing and Management System, a console-based Java program that aims to replicate and streamline the online movie ticket ordering experience, with functionalities ranging from viewing movie listings, selecting different movie sessions, viewing available seats and making seat booking, as well as simulating real-life payment procedures in a user-friendly manner. Moreover, we provide an admin panel for admin user to effectively manage movie sessions such as scheduling them in different houses with various scheduling strategies and have overall control and monitoring of different functions. Our design will hopefully address the common challenges faced by general movie-goers and cinema managers by providing an efficient and reliable platform for managing movie schedules and ticket reservations, thus reigniting their interests in watching movies in cinemas.

## 1.2 Objectives

Our design is a command line interface (CLI) Java program with a use-friendly menu that aims to comprehensively and efficiently achieve the following goals.

Firstly, we hope to streamline movie ticket booking services. We allow users to conveniently list all scheduled movie sessions at different houses, select a movie session they prefer, and book various seats based on their wishes and availability. They can also freely cancel their booking in the midst of the selection process by simply quitting from selection menu. Thus, our movie booking services are highly flexible and effortless, ensuring a satisfactory and smooth user experience.

Secondly, we provide user authentication and profiles. By default, all users access our system in the identity of guests, who can only enjoy limited functions such as listing available movies only. We allow them to login or register with a new profile, which will then grant them additional functions such as booking movies, saving previous booking records, and viewing their booking history afterwards, and checking their personal profiles. This helps provide a more personalized user experience.

Thirdly, we have integrated various payment methods. We have implemented various API payment gateways such as Octopus, credit cards, etc to simulate real-life secure payment procedures, providing users a complete movie ticketing and transaction experience.

Fourthly, our program is well equipped with error handling and support. We have incorporated robust exception handling mechanisms to ensure the validity and correctness of users’ input and in case of unacceptable input commands, our system will list out error messages and guidance to inform users on how to select their options and input suitable commands. This helps ensure a seamless and frustration-free experience.

Fifthly, we provide strategic movie scheduling methods to administrator. We have designed different movie scheduling algorithms that allow administrators to evenly or profitably schedule movies in different theaters based on their business needs.

Sixthly, we enable administrative inventory management services. To facilitate the efficient management of movie screenings, in addition to movie scheduling, we also allow administrator to add new movies and manage available seats, ensuring that the system remains up-to-date and accurate.

## 1.3 Scope

The Movie Ticketing and Management System consists of the following modules:

1. Guest Module
   * List movies/ movie sessions
   * List snacks/drinks
   * Search movies
   * Log in/ Register
2. Registered Customer Module
   * List movies/ movie sessions
   * List snacks/ drinks
   * Search movies
   * Buy movie tickets, snacks and drinks
   * Personal profiles
   * List payment records
   * Reset password
   * Log out
3. Payment Module
   * Shopping cart for selected items
   * Pay by different payment methods
4. Admin Module
   * List movies/ movie sessions
   * Add and list snacks/ drinks
   * List, search, add and delete users
   * Add and search movies
   * Schedule movies with different strategies
   * Personal profile
   * Reset password
   * Log out
5. Option Menu
   * Command interface for user input

## 1.4 Constraints

Despite our team’s concerted effort in developing the best possible solution for our target users, admittedly, our design still subjects to the following constraints:

1. Technical Constraints: For simplicity reasons and software testing concern, our program currently does not connect to an actual database e.g. a SQL database. Rather, we have implemented a Database class to simulate the effects of backend database and server. Moreover, we primarily used Java as the programming language for development, so our program is unlike other commonly seen and used web applications that involve other web technologies such as HTML, CSS, and JavaScript. As such, we currently only provide a command-line interface rather than a frontend web interface and all actions are to be performed locally on the client side without any remote backend services.
2. Usability Constraints: Our program is not intended to be multi-threaded so each time only one user can perform any operations. While no concurrent action is supported, we have implemented functionalities that save users’ actions and records so as long as the program is not terminated, users can retrieve and view any previously saved data. Of note, as our program is not connected to an actual database as discussed previously, the saved data could still not achieve the state of persistent storage as compared to most real-world systems with dedicated backend servers and databases.

Furthermore, our program is being designed in a way that assumes all movies that are being scheduled and all user actions related to the movies are all happening within a particular day, owing to the added complexity that may not be easily overcome within the limited time allowed for the project.

1. Time Constraints: This project has approximately 3-month time for completion, with tasks ranging from project planning, work allocation, program development, test case analysis, debugging, and writing various reports, which include class diagrams, use case analysis, discussion of adopted design principles and patterns. Given the extra tasks on top of the core program development, the time allowed for actual coding activities could, on estimation, be limited to approximately one month of the entire project duration. As such, continual updates, releases and software maintenance may be difficult to achieve within the short period of time.

# 2. Project Team Organization

A diagram of a project manager

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Name** | **Title** | **Roles** |
| CHAN Ka Hong | Project Manger | Management, Design, Development, Testing Support, Documentation |
| FUNG Wing Ho | Assistant Project Manager | Management, Design, Development, Testing Support, Documentation |
| LAW Lok Hang | Programmer/Tester | Design, Development, Testing, Documentation |
| NG Ka Ho | Programmer/Tester | Design, Development, Testing, Documentation |
| WONG Tsz Kin | Programmer/Tester | Design, Development, Testing, Documentation |

# 3. Methodology

Our team has adopted the Agile software development strategy, which is a methodology that emphasizes iterative, incremental progress and flexibility in response to change. Compared to traditional software development methodologies, such as Waterfall, Agile focuses on delivering small, workable pieces of software frequently and incorporating feedback from users and stakeholders throughout the development process.

In the initiation phase, we had our first kick-off meeting to discuss the possible topics. We narrowed down to the development of a system related to movie ticketing and aligned our expectations and objectives. Next, we planned and designed the potential features to be implemented. We detailedly documented our ideas and propositions, and drafted the flow, user stories and the object classes to outline the overall project direction. Preliminarily, we summarized into 5 major modules for development, namely input collection and the output interface, algorithms for movie scheduling, movie selection and booking services, the payment module, and lastly the administrative functionalities.

Following our agreement on requirement analysis, we proceeded to the development phase, with each member of our team focusing on one of the 5 key development modules. As Agile was adopted, from time to time, individual subparts of the whole software were released where each could function on its own without necessarily depending on other classes and functions. For example, the movie scheduling algorithms could work as an individual module when providing some dummy movie objects. The movie booking section could also be executed with dummy customer variables.

Even when certain modules had been released, our developers would gradually revisit their codes, debug and refactor if necessary to ensure the most up-to-date and bug-free codes were released and were ready for integration.

Meanwhile, unit test cases were written to assert that individual functions were working as expected. As more modules were released, continuous integration and development took place to incrementally build on the software. Additionally, following integration, integration test cases and acceptance test were also performed, while code refactoring and debugging were also iteratively realized as more progresses were made. With Agile, our program is actively and flexibly evolving alongside the changing constraints, ideas and requirements.

# 4. Development Tools

* Visual Paradigm
  + Class Diagram
  + Use Case Diagram
  + Sequence Diagram
* Visual Studio Code, IntelliJ IDE, Eclipse IDE
  + Programming
  + Testing
  + Debugging
* Office 365 Excel
  + Activity Log
  + Work Breakdown Structure
  + Gantt Chart
* Office 365 Word
  + Project Plan
  + Analysis & Design Report
  + Test Report
  + Bug Report
  + Self-assessment Report

# 5. Configuration Management

Configuration management is a process used to systematically handle changes in software systems, ensuring consistency, accuracy, and reliability throughout the development lifecycle. In our project, we strictly adhere to the following aspects of configuration management.

## 5.1 Version Control

Version control helps keep track of different versions of software components and ensuring that the most recent and correct versions are used during development and deployment. We primarily used Git to push, pull, merge changes, and keep a history of all modifications. To manage repositories and facilitate collaboration, GitHub was used as a central location to aggregate all our source codes and release documents.

## 5.2 Build Management

Build management tools helps automate the process of compiling source code into executable software, handle dependencies, and manage build environments. We mainly depended on Eclipse IDE build-in tools to manage our dependencies, with the main one being Junit, as well as to compile our Java source codes into executable jar files.

## 5.3 Change Control

Change control helps ensure proper management and documentation of changes to a software, including tracking the impact of changes, approvals, and implementation processes. In our project, considering its limited scale and duration of time of development, we had not used any professional tools to track our development changes. Rather, we opted a more conventional and easier approach i.e. using Microsoft Word, to record our changes and suggestions for development. Similarly, test analysis and bug cases were also recorded in Word documents.

# 6. Work Breakdown Structure

# 7. Team Gantt Chart

# 8. Future Plans

If more time is granted for the project and in future releases, our team would like to further improve and expand on the current functionalities. The following are some areas we are interested in exploring.

## 8.1 A Multi-threaded Design

One of the constraints of our program is that only one user is allowed to perform operations each time the program runs. A multi-threaded design that supports concurrent user connection and operations would make the software more usable and better reflect the functionalities of real-world cinema movie ticket booking system.

## 8.2 A Web Application

Instead of solely using Java to develop our program, a more optimal approach that aligns better with the current development trend and paradigm is to convert the software into a web application with a more user-friendly graphical interface built with JavaScript frameworks such as React.js and a more robust backend built with Java’s Spring Boot.

## 8.3 More Movie Scheduling Strategies

We would like to develop more automated movie scheduling algorithms to offer admin users more strategies to schedule the movies that can better suit their business needs. Alternatively, a manual schedule with no automation could also be implemented to provide the best flexibility for admin users to manage the movies.