KIM'S CONVENIENCE MODERN CINEMA (KCMC)
CONFIGURATION MANAGEMENT PLAN

Version 1.0

10th October 2021

By:
Hussain Khozema Kheriwala (U1822071C)
Jeremy Book Kay Yip (U1822479G)
Wong Ying ()
Desmond ()
Duc ()
Parth ()

VERSION HISTORY

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
1.0	Hussain, Jeremy	10th October 2021	Wong Ying	11th October 2021	Initial Software Configuration Management Plan draft
1.1					
1.2					

TABLE OF CONTENTS

1 Identification

This document contains the Software Configuration Management Plan (SCMP) of Kim's Convenience Modern Cinema, developed by team Kim's Convenience.

Configuration management aims to establish, maintain and ensure the integrity of the project throughout the Software Development Life Cycle (SDLC) as part of the quality control process. It encompasses identifying and controlling the configuration items and changes for the project which is accomplished by recording the statuses of such configuration items and reporting their changes in activity statuses.

1.1 Document overview

The table below lists the sections that will be documented in this document as well as the contents within each section:

#	Section Name	Section Description
1	Identification	Explain and elaborate on the abbreviations and technical terms used in the project
2	Organization	Describes the organisation of the team to perform configuration management. The tasks of each team member are listed.
3	Configuration Identification	Describes how the Software Configuration Items will be identified.
4	Configuration Control	The changes in configuration management process in the Software Configuration Items in Kim's Convenience Modern Cinema will be described here.
5	Configuration Support Activities	The activities which evaluate the degree of compliance towards the procedure and standards in implementing changes to the Software Configuration Items (established within the Software Configuration Management Plan) as demonstrated by the development team will be described here.

1.2 Abbreviations and Glossary

1.2.1 Abbreviations

Abbreviation	Description
SCMP	Software Configuration Management Plan
SCM	Software Configuration Management
SCI	Software Configuration Item
SCR	Software Change Request

SRS	System Requirements Specification
SDLC	Software Design Life Cycle
KCMC	Kim's Convenience Modern Cinema

1.2.2 Glossary

Glossary Term	Description
Software Configuration Management (SCM)	The discipline of identifying the configuration of a system for controlling changes to this configuration as well as maintaining the integrity and traceability throughout the Software Development Life Cycle.
Software Configuration Management Plan (SCMP)	A reference document for the SCM process that includes the work breakdown structure and description of each section of each of the respective processes.
Software Configuration Item (SCI)	Any service component, infrastructure element, or other item that needs to be managed in order to ensure the successful delivery of services.
Software Change Request (SCR)	Software change request is a tool used for requesting, approving, and documenting changes to the project. A change request is a declarative document, stating what needs should be accomplished while leaving out how changes should be carried out.
Software Development Life Cycle (SLDC)	Software Development Life Cycle is the application of standard business practices to building software applications

1.3 References

1.3.1 Project References

#	Document Identifier	Document Title
1	[R1]	Movie prediction techniques Available at: https://towardsdatascience.com/the-4-recommendation-engines-that-can-predict-your-movie-tastes-109dc4e10c52

1.3.2 Standard and regulatory References

KCMC will be storing multiple users' personal and movie booking data. Hence, it is important to protect the privacy of all users and their data and Team Kim's Convenience shall be adopting the

International Organization for Standardization ISO/IEC 27000 information technology security technique.

#	Document Identifier	Document Title
1	[STD1]	IEEE Standard for Software Configuration management Plans - IEEE Std 828 -1998
2	[STD2]	ISO/IEC 27000:2018 Information technology — Security techniques — Information security management systems — Overview and vocabulary. (2019, February 4). International Organization for Standardization. https://www.iso.org/standard/73906.html https://standards.iso.org/ittf/PubliclyAvailableStandards/c073906_ISO_IEC_27000_2018_E.zip

1.4 Conventions

Typographical convention.
Any other convention.

2 Organization

Team Kim's Convenience will be managing the KCMC application. The tasks of Configuration Management will be shared between the Project Manager, Release Engineer as well as the Quality Assurance Manager & Engineers.

The development team will be using Github for source code management and version control in order to track the history of changes and who was responsible for the changes. This allows the team to traceback easily in the event where there is an issue found in any of the releases.

The tables below shows the activities as well as who will be using these management tools:

Software Tools	Users & Activity
Git	**Lead, Front-end & Back-end developers:** Push and Pull the source code for development of KCMC. Work on the issues and complete the milestones **Release Engineer:** Create branches for each release / development phase for version control. These branches will also help to narrow down the location of problematic components and bugs. **QA Manager & Engineers:** Pull source code from different branches and carry out quality assurance testing to discover bugs.
Github Issues/Milestones	**Project Manager:** Track software development and milestone progress of

	KCMC. **Release Engineer:** Track version release status of KCMC development. **Lead:** Create tickets for developers to work on **Front-end & Back-end developers:** Work on the tickets to achieve the milestones

2.1 Activities and responsibilities

The tables below shows the activities as well as who will be responsible for them:

Activities when setting up the project	**Person responsible**
Identification of Software Configuration Items (SCI's)	Quality Assurance Manager & Engineer
Install the bug repository tool and set up the database	Quality Assurance Manager & Engineer
Install the software configuration repository tool and set up the database	Project Manager
Setup of documentation archives and repository	Release Engineer
Define the configuration processes	Project manager & Release Engineer

Activities during the project lifecycle	**Person responsible**
Export components for modification, test or delivery	Quality Assurance Manager & Engineer
Set under control validated components	Release Engineer
Create version, write version delivery document	Release Engineer
Approve reference configurations	Project manager
Verify version to be delivered and authorise deliveries	Project manager
Creating Backup Spaces	Release Engineer
Do configuration audits	Quality Assurance Manager

Inspect configuration records	Quality Assurance Manager
Archive reference version	Release Engineer

Management activities	**Person responsible**
Manage versions and archives for software	Release Engineer
Manage configuration records	Release Engineer
Manage archives for documents	Release Engineer
Manage reference space and its access control list	Release Engineer
Manage spaces backup and archive media	Release Engineer
Manage quality reports	Quality Assurance Manager & Engineer

2.1.1 Decisions process and responsibilities

Responsibilities during reviews, audits and approvals that will be made through the Configuration Management Plane are listed in the table below:

At the end of an activity of the project

Activities	**Person Responsible**
Do a configuration freeze	Release Engineer
Present a configuration state of the components impacted by the activity	Release Engineer
Present a documentation state of the components impacted by the activity	Release Engineer

During a configuration management process audit:

Activities	**Person Responsible**
Do the configuration management process audit	Project Manager
Present the records of the configuration management process	Release Engineer
Present the quality records of the configuration management process	Quality Manager
Present the records of the documentation management process	Release Engineer

3 Configuration identification

The purpose of Configuration Identification will determine the way software configuration items will be created and identified throughout the Software Development Life cycle (SDLC) of KCMC project.

3.1 Identification rules

3.1.1 Identification rules of configuration items

Software Configuration Items (SCIs) will be identified via the pre-defined format which will be used for the entire SDLC of the KCMC project. The following SCIs are expected to be identified within the product:

- Web Frontend Source Code (HTML, CSS Files)
- Web Backend Source Code (Javascript files
- Django Backend Source Code (Python Files)
- SQLite Database (Relational Database)
- Github (Version Control using Git)

The format of nomenclature for these items shall be as follows:

[File Name]_[Version Number].[File Format]

Moreover, source code files are further organized into different folder
directories to separate logic, frontend and backend functions

Moreover, source code files are further organized into different folder directories to separate logic, frontend and backend functions during the SDLC. This helps to facilitate configuration item identification.

3.1.1.1 Identification of a configuration item

A configuration item can be identified by following its format:

[File Name]_[Version Number].[File Format]
File name and version number can be used to identify the configuration item name and its version respectively.

3.1.1.2 Version number of a configuration item

The usage of a version number is a prerequisite when providing a configuration item. This number shall be incremented when there are any modifications to the document.

The assignment of unique version numbers will reflect the type of changes made, and their corresponding severity. For each major, minor and revision change, the version number assigned will increase by 1.0, 0.1 and 0.001 respectively. This is further elaborated below:

Major	- Additions and changes made to the main features of KCMC
Minor	- Bug fixes relating to the main features of KCMC - Additions and changes made to the minor features of the KCMC - Increments of the Minor version number will be reset after each Major version increment.

Revision	● Bug fixes, additions and changes made to the minor features of KCMC ● Increments of the Revision version number will be reset after each minor version increment.

3.1.2 Identification rules of SOUPs

3.1.2.1 Identification of a SOUP

Use you own ID like above or take ID of SOUP manufacturer

3.1.2.2 Version number of a SOUP

Use you own ID like above or take ID of SOUP manufacturer

3.1.3 Identification rules of documents

3.1.3.1 Description of documents identifiers

The documents of the KCMC project serve throughout the development life cycle of the project. The document name and their respective revision histories shall determine the format for identifying documents. On that note, the comprehensive list of documents associated with the KCMC project are listed as follows:

1. Project Proposal
2. Use Case model
3. System Requirement Specification (SRS)
4. Quality Plan
5. Project plan
6. Risk Management Plan
7. Design Report On Software Maintainability
8. Configuration Management Plan
9. Change Management Plan
10. Developer test cases
11. Test Plan
12. Release plan
13. Requirement Test Coverage Report

The document type can be inferred from the document name, while the version number shall encapsulate the revision history of the document. The format of the document is as follows:

[Document Name]_[Version Number].[File Format]

Further, these documents shall be carefully organised into their respective folder directories based on their purposes throughout the SDLC, which shall further facilitate the document identification process.

3.1.3.2 Definition and evolution of the revision index

The assignment of unique version numbers will reflect the severity of the changes made. For each major and minor change, the version number assigned will increase by 1.0 and 0.1 respectively. This is further defined below:

Major	- Significant content modification made to the documentation. These include document content alterations in at least one section. - Must be approved by the Project Manager.
Minor	- Insignificant modifications made to the documentation. These include formatting of the document and grammatical corrections. - Need not be approved by the Project Manager. - Increments in the Revision version number will be reset after each major version number increment.

3.1.4 Identification rules of a media

The media in the KCMC project consists of the images of movie posters. These movie posters can be identified and inferred from the image name. The format of the media is as follows:

[Image Name].[File Format]

3.1.4.1 Internal identification

The media can be found inside the media directory in the source code.
directory structure: modern_cinema/media/<image name>
where:
"media" is the media directory,
"image name" is to identify the movie poster

3.2 Reference configuration identification

Document Name	Version Number

Project Proposal	3.1
Use Case Description	1.1
System Requirement Specification	2.1
Quality Plan	2.1
Project Plan	1.4
Risk Management Plan	1.1
Configuration Management Plan	1.1
Design Report On Software Maintainability	1.1
Change Management Plan	1.1
Release Plan	1.1
Developer Test cases	1.0
Test Plan	1.0
Requirement Test Coverage Report	1.0

3.3 Configuration Baseline Management

Multiple baselines have been defined for the KCMC project to ensure proper management and project control throughout the lifecycle of the project.

The purpose of a Configuration Baseline is to serve as a fixed reference throughout the Software Development Lifecycle Cycle (SDLC). It will be documented on the basis of defining change control for various configuration items. It will be considered as a baseline for all the changes that follow.

These baselines is defined as follows:

Baseline	**Baseline Description**	**Period of Establishment**	**Components**
Functional	Defines the functionality requirements based on the system specifications. It will document the system's functionality and capabilities at the minimum benchmark.	During Project Planning phrase and after Deployment & Maintenance Phase	1. Project Proposal 2. System Requirements Specification (SRS) 3. Quality Plan 4. Project Plan 5. Risk Management Plan

Allocated	Defines the design of the functional and interface characteristics that composes the system.	After Design Phase	1. Use Case Model 2. Database Design 3. User Interface Design
Product	Defines the completed and accepted system components and documentation that identifies these products	After Integration and Testing Phase	• Source Code • Test Cases • Developer Test Cases • Release Plan • Requirement Test Coverage Report

4 Configuration control

During the Software Development Lifecycle (SDLC), part of the Configuration Management domain is Change Management within the Software Configuration Items. With proper control, the team will have to give consent and acknowledge any changes made to the system to avoid any unsupervised changes.

The steps to Configuration Control are:

- Recognising that a change is necessary
- Discuss the change request submitted
- Approval or Disapproval of the change request
- Verification, Implementation and Release of the change request

Only people external to the development team may propose changes, and all requests must use the Change Request Form (CRF). The Change Request Form must be filled up and submitted to the Change Control Board (CCB) for approval. The form must be reviewed by the Lead Developer and the Quality Assurance Manager & Engineers to carry out impact analysis for the requested changes. Change assessment will also be carried out to set a priority of the change, finding out the cost to implement the changes and the budget of the company to implement these changes.

The Change Request Form must include the change, explanations on why it is implemented, and affected source codes.

The project manager will have a meeting with the Lead Developer and Quality Assurance Manager & Engineers to discuss the feasibility of these changes and how they will benefit the software and write down their analysis on the CRF.

After the changes have been approved by the CCB, a new separate branch will be created from the main branch. The separate branch will contain the new changes and will be tested to ensure it is clear of bugs before merging back to the main branch.

4.1 Change Management

Upon change approval from the Change Control Board(CCB), the change request will be accepted by the project manager and a branch will be created in the Software Configuration Management System (SCM). Software Configuration Items (SCI) will be “Checked out” of the SCM for changes to be made by the assigned development team. When changes are completed by the assigned development team, the codes will be reviewed and audited to make sure that implementation meets the updated

requirement. The new SCI will be "Checked in" into the SCM with the Baseline for testing being established. Finally the QA team will carry out the final testing to ensure that the product meets the expected quality before the changes can be merged with the main branch.

After QA testing, if there are no bugs reported, the development team can proceed to merge and integrate the new changes with the original branch and the software version will be modified based on the impact of the change (Major, Minor or Minor Revision). The entire development team will then be updated of the new changes made to the application.

To provide details on the changes, the branch name must identify the part of the software being changed, and proper commit details should be added.

The branch with the new changes must be tested and be cleared of bugs before being merged with the main branch, in order for the main branch to have the latest code.

All commit messages from the new branch must clearly state the changes made and the reason for the modifications. Software version of the main branch will be updated based on the scale of the changes.

4.2 Evolutions control of SOUP items

Explain how you manage evolutions of SOUP.
A simple solution is to freeze SOUPS at the beginning of the project, not always feasible.

5 Configuration support activities

5.1 Configuration Status Accounting

Configuration Status Accounting (CSA) is the process to record, store, maintain and report the statuses of Software Configuration Items during the Software Development Life Cycle. In order to ensure traceability, all SCIs and relevant documentation must be tracked. To track changes in the system, version numbers of the SCIs are used. They are responsible for:

- Elements that have to be tracked and reported for baseline changes.
- The types of Status Account Reports that are to be generated and the frequency of generating such reports.
- The collection, storage and reporting of information.
- Access control to the Configuration Management data of this project.

5.1.1 Evolutions traceability

The traceability of modifications of items given their types:

- Document: The modification sheet number identifies the origin of the modification. The modified paragraphs in the document are identified, if possible, by revision marks. Any changes made to the documentation will be saved with a different file name, based on the version number
- Source file: The software configuration management tool records, for each source file or group of source files, a comment where the modification is described.
- Configuration item: The Version Delivery Description of the article identifies the modification sheet included in the current version.

The modification sheet describes the modifications done to the components with enough precision to identify the modified parts.

5.1.2 Setting up Configuration status

Throughout the development lifecycle of this project, the Lead Developer will take on the role of the Software Configuration Manager and write the Version Description Document.

Being the Software Configuration Manager, the Lead Developer will set up the state of all versions of each Software Configuration Item consisting of the label, version number, and creation date of the Version Description Document.

5.1.3 Configuration status diffusion

Both the Quality Assurance Manager and the Lead Developer will write the Version Description Document. The Project Manager must then vet and approve the document in order for it to be considered valid.

5.1.4 Configuration status records storage

The records of each SCI are stored in a configuration folder, which contains the following:

- Software Change Requests, sorted by Request ID
- Software Documentation
- Version Description Documents
- Configuration States sorted chronologically

5.2 Configuration audits

The majority of the Configuration Audits in this project are baseline audits. These baselines are an agreed description and definition of attributes of the product, KCMC at a given point in time that serves as a foundation for future changes.

The most recent status of the SCIs is reviewed and tracked using a standardized checklist so as to ensure that each of their performances is in line with the System Requirement Specifications. The Quality Assurance Manager and Engineer will perform formal reviews of test results of the functional configuration of SCIs to verify that CashTrack has met the requirements identified.

The other types of configuration audits used in this project as as follows:

Configuration Audits	Audit Description	Key Personnel Involved
Functional	Examines the functional characteristics of a configuration item to verify that its functionality is consistent with the functional requirements as specified in the System Requirement Specification	● Project Manager ● Software Configuration Manager ● Developer ● Quality Assurance Manager

Software	Ensures that the software product KCMC satisfies the baseline needs so that the product built is the product delivered	● Project Manager ● Software Configuration Manager ● Developer ● Stakeholder

5.3 Reviews

The baseline audit will serve as a foundation for changes to be made in the future, but before the baseline can be utilised, it has to go through thorough review..

A project management tool that helps to set up these baselines is known as a review. The table below lists the personnel involvement and their responsibilities relating to reviews:

Personnel	Responsibilities
Lead Developer	- Select the objectives and Software Configuration Items for review - Provide approval criteria
Quality Assurance Manager & Engineer	- Planning and deciding a schedule for the review based on procedures - Formulate the procedures to document the flaws within the project as well as the solutions to these flaws
Technical Reviewer (An internal assignment within the project team)	• Have the appropriate domain knowledge or subject matter expertise relating to the work product • Provides technical feedback (including possible defects) on the work product
Recorder	• Documents the issues and key points discussed during the review in a short and concise manner

5.4 Configuration management plan maintenance

Throughout the whole Software Development process, the Software Configuration Management plan will be periodically updated. This way, it can be used to help identify the processes that will be used to implement changes to the identified requirement specifications.

The monitoring of these plans is to be done by the Quality Assurance Manager & Engineer. As for the updating of the Software Configuration Management plan, any changes for the document will have to be assessed and approved by the Quality Assurance Manager & Engineer. Every week, the Quality Assurance Manager & Engineer will collate the approved changes and update the Software Configuration Management plan accordingly.

Once approval for the plan has been obtained, the latest plan is then saved with a newer version number. All older plans will be saved in the configurations folder, this folder can be accessed to help improve traceability as well as facilitate changes to the project schedule.