CS673 Team-5 Project

Software Configuration Management Plan (SCMP)

1. Introduction

This Software Configuration Management Plan (SCMP) outlines the processes and tools used to manage and control all software configuration items (SCIs) in the project. This includes the versioning and control of code, documentation, design documents, and other project artifacts.

2. SCM Management

2.1 Organization

The SCM process will be organized by the project team. Each team member will have specific responsibilities in managing software artifacts, ensuring version control, and maintaining the project repository.

2.2 Configuration Items

The key software configuration items (SCIs) will include:

- **Code (product and test):** All source code for the project.
- **User Documentation:** User manuals and guides.
- **Specification Documents:** Requirements and design documentation.
- **Supporting Software:** Libraries, compilers, and dependencies.

SCIs will be developed by team members and version-controlled using Git. All SCIs will be placed in relevant GitHub repositories.

2.3 Applicable Policies, Guideline, and Procedures

Code Commit Guideline and Git Branching Strategy

- The source code will be managed through Git and organized in the GitHub. The project will follow various code management techniques, such as version control and branching policies.

- Main branch: It serves as the main line of development and should only be used to pull changes from upstream.
- Iteration branch: We create a new branch for each iteration or feature development. Developers work on these branches and merge them into the main branch at the end of the iteration after review.
- Commits must be atomic, meaning each commit should represent a single, well-defined change. Before pushing changes to a remote repository, developers should ensure that their code is properly tested and passes all existing tests.
- Any changes to project files must go through a review process, with final approval from the SCM Manager before being merged into the main branch.

Document Management and Progress Tracking

- All SCIs will be on GitHub, however documents will also be stored in Google Drive with appropriate names and revision history to clearly indicate the changes made by any revisions. This includes all documents such as designs, requirements, plans and analyses, as well as user-specific documents such as guides and tutorials.
- A progress report that records the progress every week will be managed.
- A summary document of every time meeting will be managed.
- Any required document in future like a Test Case document etc. will be added to manage.
- Development tasks and progress will be managed in document and Jira.

3. Workflow Guideline

- Connect your local repository to the project repository hosted on GitHub.
- Before starting work, ensure your local main branch is updated with the latest changes from the remote repository.
- The development of system functions requires creating a new branch for development after reading the system design document.
- Stage and commit your changes locally, then push them to your remote iteration branch.
- Initiate a request on GitHub to merge your changes from the iteration branch into the main branch after completing your feature development.

- After completing the functional development, it needs to be recorded in the system design document, and the possible development directions in the future should also be recorded in the document.

4. Release Management and Delivery

Before each major project release, a tag will be created in GitHub representing the stable version. The SCM Manager will coordinate with the development team to ensure all changes are stable and documented.

Each release needs to be ensured to be in a runnable state. If there are any known bugs, they need to be recorded in detail in the release notes.

The delivery after the project is completed needs to provide a detailed deployment document, which will include the software version and operating environment used in the project. This document also provides a step-by-step introduction to help users successfully deploy the service on their own servers.

5. Tools and Resources

Git: Version control and source code management.

GitHub: Remote repository and issue dealing.

Docker: Back-end deployment and development environment synchronization tool.

Nginx: Web-Serve, front-end development tool.

Discord: Communication tool within team members.

Jira: Tasks and progress management.