**Software Configuration**

**Management Plan**

**For the Development of FlyAway**

**Created by GreyMatter**

[**Introduction**](#_n0cv87s0g60f) **3**

[Purpose](#_khuh5bejp38e) 3

[Overview](#_a95vbopxe8e5) 3

[Scope](#_d53o4tjebn5e) 3

[Configuration Items](#_nqrfwexwaidf) 3

[Other Software](#_24esmd5ol1sl) 3

[Limitations](#_cgbeoksqqt5y) 4

[Key Terms](#_cfg3zsgvqeq0) 4

[References](#_pr7pts80s843) 4

[**SCM Management**](#_8vouh29iw2dc) **4**

[Organization](#_xx7h8yr5q3e5) 4

[Responsibilities](#_6syvkwtuikxx) 4

[Applicable Policies, Directives, and Procedures](#_83uvne4menhq) 5

[**SCM Activities**](#_jrym0cgs3e7) **5**

[Configuration Identification](#_mivdunn9mszf) 5

[Configuration Control](#_p389i94qazka) 6

[Configuration Status Accounting](#_s2g9nono73wr) 6

[Configuration Evaluation and Reviews](#_669znllr04k9) 7

[Interface Control](#_yaneoxivffui) 7

[Subcontractor/Vendor Control](#_woq1o2o1fc9h) 7

[Release management and Delivery](#_ig7q90wfxmu2) 7

[**SCM Schedules**](#_hpeiz8s3ijcv) **7**

[Sequence and coordination of SCM activities](#_j9uvnm9pr5za) 8

[Relationship of Key SCM Activities to Project Milestones or Events](#_1n2y2qgzuc7) 8

[Schedule Relative to Meetings and Sprint Progression](#_ppju5das19m0) 8

[**SCM Resources**](#_cvy07j6df447) **9**

[Environment](#_pm1ragjvjues) 9

[Infrastructure](#_o5vbwbaxbf31) 9

[Software Tools](#_ltyxlpvahymz) 9

[Techniques](#_b5apkvnb5jqo) 9

[Equipment](#_tq8564j7s8e3) 10

[Personnel](#_3eug07oogtpo) 10

[Training](#_vazsmp2csuda) 10

[**SCM Plan Maintenance**](#_m9lq5z3fnoz6) **10**

[Plan Monitoring](#_8k81jk9hi7uc) 10

[Plan Update Schedule](#_6lx09imgydq8) 10

[Plan Changes](#_vbgk3582b3mt) 11

[Change History](#_wl4haveaas80) 11

# Introduction

## Purpose

The purpose of this Configuration management Plan (CMP) document is to describe the Configuration Management (CM) activities that are to be performed during the development of the FlyAway web application. This document is intended for the use of the development team GreyMatter, who will update the document as development continues..

## Overview

FlyAway is a free, web-based application that allows pilots to connect with a market of individuals looking to take a flight. Users can sign up as a Pilot to connect with passengers or a Passenger to browse and book flights with Pilots.

*For more information about the application, view the ‘FlyAway’ documentation.*

## Scope

The scope of this CMP applies to everything that is developed for the FlyAway application. Such things include any code, documentation, presentations, and such that are relevant and created during the development process. The CMP documentation identifies the management and procedures that are to be followed by the team members throughout development and maintenance of the FlyAway system.

## Configuration Items

* + 1. Source code
    2. Software Requirements Specification Document
    3. Configuration Management Plan Document
    4. Sprint Backlog
    5. Product Backlog
    6. User Stories Document
    7. Project Description Document
    8. Sprint Review Documents
    9. Use Case Model Document
    10. Sequence Diagram Document
    11. Domain Model Document
    12. Software Architecture Document
    13. Focus Group Document

## Other Software

NodeJS - Server Environment

ReactJS - Web Framework

Visual Studio Code - Code Editor

Git - Version Management

GitHub - Issue Tracking

ZenHub - project management

Firestore - backend support

## Limitations

The time for development is the most significant limitation of the project. The development time is over the course of a semester, which is only about 15 weeks, therefore only a portion of the planned feature and user stories will be implemented by the end of the production time frame.

In addition, other limitations are the small development team of only 5 students and a lack of experience in most of the skills needed for most of the team members that needs to be during the process.

## Key Terms

CMP - Configuration Management Plan

CM - Configuration Management

FlyAway - the system under development

GreyMatter - the team creating FlyAway

CI - Configuration Item

## References

References used in creating this document include the 3 example documents provided from class.

# SCM Management

## Organization

The CM activities are mainly organized by the team members of GreyMatter assigned to Scrum Master and Product Owner of the sprint specific week and Allison Lupien, but the full team is expected to give insight and provide assistance or clarification when necessary. Most of the decisions pertaining to CM will be made during the period scrum meeting held by the team.

## Responsibilities

Responsibility is assigned at the beginning of each sprint. The different responsibilities are:

Scrum Master: Oversees that the development team is on track with the story tasks assigned to them and that they have everything they need to accomplish their tasks.

Product Owner: Tracks the completion of story tasks and goals, making sure all are entirely completed before marked as such.

Developer: It is in charge of implementation of the story tasks and the documentation that may be needed with it.

## Applicable Policies, Directives, and Procedures

* + 1. All documents will follow the appropriate standards that are expected from the class material.
    2. Due to the limited development time period, the project will have partial production of the software portion.
    3. Communication between developers for the project is done through online resources due to the remote delivery of the course.

# SCM Activities

## Configuration Identification

Configuration Items (CI) will primarily be documentation, code files or other components created during development that can be tracked by version management.

* + 1. Identifying Configuration Items

Configuration items are identified by a couple scenarios: documents that are required as course material, items that are determined necessary due to analysis of the created user stories or system requirements, or discussion with peers or potential users.

* + 1. Naming Configuration Items

The naming convention used mainly applies to the documentation, and it is based on the sprint number, the document or configuration item being identified and the type of file. Words are generally separated by underscores to allow easy reading of the title. The naming convention for CI documents generally follow the form GreyMatter\_Deliverable\_i\_Name.type where i stands for the sprint number, Name stands for the document name and type stands for the file type. For example, this Configuration Management document that is developed for sprint 4 and will be turned in as a word document will be named GreyMatter\_Deliverable\_4ConfigManagment.docx.

On the other hand, there is no strict naming convention for code files. Code files should generally have a name that suggests the contents of the file and should be marked with the file type. To combat this ambiguity in the naming conventions of code files, files should contain a comment within the file that gives a general description of what it does.

## Configuration Control

As FlyAway has a small development team, therefore changes can be made in an informal manner. If a developer feels a change is needed, they can propose it during a scrum meeting or over messaging. The rest of the team will then decide if the change is necessary. If it is, the implementation of the change is added to the schedule.

* + 1. How to Request a Change

To request a change to a document, code file, or other file, an individual can mention the issue during a meeting or more informally over messaging. A formal change request would be reflected in the documentation such as the Kanban board on ZenHub and the sprint backlog.

* + 1. How to Assess Change Viability

The Viability of a change can be assessed during a meeting, or if a small change over messaging. The team will discuss the viability of the change by assessing the time the implementation might take, the challenges it may create and the overall payoff of the implementation at the end.

* + 1. How Changes are Implemented

After a change has been confirmed viable, the implementation should begin soon after. These changes that are made to the code are logged through Git commits.

* + 1. How Changes are Logged

After changes have been completed, they should be marked done within the documentation, such as the Kanban board and necessary spreadsheets.

## Configuration Status Accounting

The FlyAway development will be tracked using GitHub projects and documentation within multiple spreadsheets such as spring backlogs and the product backlog as well as a Kanban board. The status of issues from waiting, in progress to done should be tracked on said documentation and board. The product backlog tracks user stories and records when they are implemented into the project. The sprint backlogs record which user stories have been implemented for each sprint along with a more in-depth summary of who worked on what and how much was finished.

## Configuration Evaluation and Reviews

Due to the small development crew of FlyAway, reviewing and evaluating documents and code can be done in an informal process. When a team member feels that a component, whether code or documentation is completed, they can alter the full company and the team can review said component. If the component is deemed finished, then the project can be updated in GitHub and within the documentation.

## Interface Control

FlyAway interface externally across web browsers on multiple operating systems. Changes that affect this interface, such that might change the visibility, design or behavior of the interfacing, should be handled carefully. There exists no active protocol, but it is important to take caution while implementing changes deemed explicitly necessary to the interfacing so as to not disrupt the functionality of the application.

## Subcontractor/Vendor Control

In future development, FlyAway will be connected to Paypal. The integration will need to be monitored to ensure that the functionality of payment transfer throughout the life of FlyAway stays consistent. Changes will need to be made if necessary to keep functionality.

## Release management and Delivery

Major release and delivery of the application occurs at the end of each sprint when all code and documentation appearing on the GitHub repository is fully up to date. These sprint end dates are decided by the course deadlines. Minor release and delivery is ongoing throughout each sprint as development team members push content from their own remote repositories to the main Git repository.

# SCM Schedules

| **CM Activity** | Frequency |
| --- | --- |
| CM Management |  |
| Plan CM activities | At the start of each sprint |
| Report CM Status | At the end of each sprint (SprintReview) |
| Monitor CMP | ongoing |
| Update CMP | At the end of the sprint, to be inplace for next sprint |
| Configuration Items |  |
| Identify CI | ongoing |
| Maintain CI List | ongoing |
| Assess CI List Coverage | At the end of each sprint |
| Change Control |  |
| Propose Change Request | ongoing |
| Assess Change Request Viability | ongoing |
| Assess Change Effectiveness | At the end of each sprint |

## Sequence and coordination of SCM activities

In general, SMC activities follow a similar flow of requesting a change within a meeting, checking viability, implementing the changes, getting approval to commit, and then committing to the GitHub repository. All SMC activities that are then accomplished at the end of a sprint are documented within the sprint backlog and other various documentations.

## Relationship of Key SCM Activities to Project Milestones or Events

* + 1. A change request is presented.
    2. It gets informally assigned to one or more team members.
    3. Development occurs on local repositories to implement change
    4. Change gets approval by the team.
    5. The team member(s) is allowed to push the changes to the main repository.
    6. The change is marked in the documentation.

## Schedule Relative to Meetings and Sprint Progression

* + 1. First Meeting: beginning of sprint
       1. Identifying User Stories to be worked on.
       2. Goals are established for the sprint.
       3. Presenting change requests and checking viability.
       4. Tasks are assigned to team members.
    2. In Between Meetings: during sprint
       1. Implementation of changes to code and documentation occurs.
       2. Informal communication through messaging
    3. Middle Meetings: meetings during sprint
       1. Discuss progress made on goals.
       2. Additional change requests can be made.
       3. Approvals for commits can be made.
    4. Last Meeting: last day of sprint
       1. Make last approved commits.
       2. Review what has been accomplished.
       3. Log the changes in the documentation.

# SCM Resources

## Environment

FlyAway will be developed using a React application and the language JavaScript as well as HTML and CSS.

## Infrastructure

The development of FlyAway takes place over GitHub and the documentation over a shared Google Drive.

Meetings take place over Zoom and informal meetings and communication occurs over a group discord server.

The back end of the development is supported by Firestore.

## Software Tools

The majority of the SCM coordination and planning for software development will be done through GitHub.

VisualStudioCode is the code editor used by all team members.

Google Drive acts as a collaborative space to work on documentation.

## Techniques

The development is done in sprint cycles where user stories are selected to be worked on for a sprint and certain tasks and goals are set to be completed by the end of that sprint for each user story. Sprints are generally 2 weeks, sometimes only 1, and each sprint a set of user stories and tasks are assigned to be implemented by the end of the sprint.

A Kanban board is used to track the progression, which allows for a more agile flow of development than a stricts sprint and scrum related development technique. This allows users stories to be assigned to multiple sprints, but with different tasks assigned.

## Equipment

All team members are required to have a functional computer that can access the internet in order to access GitHub and other online resources, that also has the ability to access a code editor and other necessary tools to develop software.

## Personnel

All team members are expected to be aware of the CM activities, but the majority of the oversight of the CM activities and the CMP documentation is managed by Allison Lupien. Specific questions about the document should be directed toward her.

## Training

The development of FlyAway has no official training, but it has required much informal training outside of the development in order to accomplish what is needed. All team members needed to personally learn React and Javascript/HTML/CSS in order to start development.

Other ‘training’, such as relating to documentation should be learned along the way of the development process through classwork or outside work.

# SCM Plan Maintenance

## Plan Monitoring

The plan will be monitored by the acting Scrum master and Product Owner of the sprint as well as Allison Lupien, the rest of the development team will be part of the decision making, but they don’t have to do any monitoring.

## Plan Update Schedule

The plan should be monitored throughout each sprint and if any revisions are needed they can be updated at the end of the sprint to be in place by the next sprint.

## Plan Changes

Changes to the CMP should be proposed like any other change proposal and be validated, assigned to a team member, implemented, approved and committed.

## Change History

The change history will be tracked through GitHub.

<https://github.com/weeksForDays/GreyMatter#workspaces/da-bois-and-girl-62059f2f4a5ae00014cb8c71/board>