Configuration management plan

[Foodies]

Version 1.0

[16, 03, 2024]

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Release No. | Date | Author | Revision Description |
| 1.0 |  |  | Initial Draft Version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

I have carefully assessed the Configuration Management Planforthe *<<Foodies >>*

MANAGEMENT CERTIFICATION - Please check the appropriate statement.

\_\_\_\_\_\_ The document is accepted.

\_\_\_\_\_\_ The document is accepted pending the changes noted.

\_\_\_\_\_\_ The document is not accepted.

We fully accept the content within this project artifact and associated tasks.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Mohamed Ibrahim>> <<16/03/2024>>*

*<<Project Manager>>*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Nourhan Ali>> <<16/03/2024>>*

*<<Quality Specialist>>*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Shimaa Shehata>> <<16/03/2024>>*

*<<Programmer>>*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Manar Ali>> <<16/03/2024>>*

*<<Programmer>>*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Mostafa Mohamed>> <<16/03/2024>>*

*<<Quality Specialist>>*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*<<Aalaa Adel>> <<16/03/2024>>*

*<<Programmer>>*

# Contents

\

[1. Overview 6](#_Toc448133047)

[1.1. Project Background 6](#_Toc448133048)

[1.2. Purpose and Scope 6](#_Toc448133049)

[1.3. Assumptions/Constraints 6](#_Toc448133050)

[1.4. Roles and Responsibilities 6](#_Toc448133051)

[2. Configuration Management Strategy 7](#_Toc448133052)

[2.1. Configuration Management Strategy Overview 7](#_Toc448133053)

[2.2. Documentation Repositories 7](#_Toc448133054)

[2.3. Technical Environments 7](#_Toc448133055)

[2.4. Development Environment 7](#_Toc448133056)

[2.4.1. Test Environment 7](#_Toc448133057)

[2.4.2. Training Environment 7](#_Toc448133058)

[2.4.3. Pre-Production Environment 7](#_Toc448133059)

[2.4.4. Production Environment 7](#_Toc448133060)

[2.4.5. Roles and Responsibilities for Environment Management 7](#_Toc448133061)

[2.4.6. Tools Used for Environment Management 8](#_Toc448133062)

[2.4.7. Environment Access 8](#_Toc448133063)

[2.4.8. Environment Schedule 8](#_Toc448133064)

[2.4.9. Environment Maintenance 8](#_Toc448133065)

[3. Configuration Items 9](#_Toc448133066)

[3.1. Configurable Item Identification and Management 9](#_Toc448133067)

[3.1.1. Configuration Item Class: Documentation 9](#_Toc448133068)

[3.1.2. Configuration Item Class: Software / Code 9](#_Toc448133069)

[3.1.3. Infrastructure Configuration Items 10](#_Toc448133070)

[3.1.4. Interface Configuration Items 10](#_Toc448133071)

[3.1.5. Database Configuration Items 10](#_Toc448133072)

[4. Configuration Management System 11](#_Toc448133073)

[4.1. Change Management 11](#_Toc448133074)

[4.2. Documentation Configuration Management System 11](#_Toc448133075)

[4.3. Software / Code Configuration Management 11](#_Toc448133076)

[4.4. Infrastructure Configuration Management 12](#_Toc448133077)

[4.5. Interface Configuration Management 12](#_Toc448133078)

[4.6. Database Configuration Management 12](#_Toc448133079)

[4.7. Library Configuration Management 13](#_Toc448133080)

[5. CM Plan Maintenance 14](#_Toc448133081)

[6. Configuration Management Automation Support 15](#_Toc448133082)

[6.1. Automated Tools 15](#_Toc448133083)

[6.2. Work Space Management 15](#_Toc448133084)

[6.3. Build Management 15](#_Toc448133085)

[Appendix A: Key Referenc 16](#_Toc448133086)

[Appendix B: Key Terms 16](#_Toc448133087)

# Overview

<<This document should be reviewed and approved by both the Provider and the Customer Program Managers.>>

## Project Background

*The Foodies Web App project aims to develop a comprehensive online platform designed to revolutionize the way users discover, order, and interact with nearby restaurants. The purpose of this new system is to provide a convenient and efficient solution for users to explore dining options, place orders, and enjoy a seamless dining experience.*

***Key functions of the Foodies Web App include:***

* *User registration and account creation using email authentication and password security measures.*
* *Displaying a dynamic home screen with a curated list of nearby restaurants, searchable by cuisine, location, and user preferences.*
* *Offering a dedicated section for promotions and special offers from partner restaurants.*
* *Implementing a credit-based loyalty system to incentivize user engagement and foster repeat business.*
* *Providing restaurant owners with a platform to promote their menus, services, and special events.*

*The business need for the Foodies Web App stems from the increasing demand for convenient and innovative solutions in the food industry. With the rise of online food delivery services and the growing reliance on digital platforms for everyday tasks, there is a clear opportunity to streamline the restaurant discovery and ordering process.*

*The benefits of this new capability are manifold. Firstly, it enhances user convenience by offering a centralized hub for restaurant information and ordering, eliminating the need for multiple apps or websites. Secondly, it empowers restaurant owners to reach a wider audience and showcase their offerings, thereby increasing visibility and potential revenue. Moreover, the Foodies Web App aligns with strategic goals by leveraging technology to enhance customer experiences and promote local businesses in the digital marketplace.*

## Purpose and Scope

The Configuration Management Plan (CMP) for the Foodies Web App project is designed to ensure the effective management of system software, hardware, and documentation changes throughout the project lifecycle. The primary purpose and objectives of the CMP are outlined below:

**Uniform CM Practices:** The CMP establishes uniform configuration management (CM) practices to ensure consistency and standardization in managing changes to project artifacts.

**Identification of Configuration Items:** It identifies the specific items placed under configuration control, including but not limited to software components, hardware configurations, and project documentation.

**Establishment of Identification Conventions:** The CMP establishes clear and concise system and document identification conventions to facilitate the accurate identification and tracking of configuration items.

**Baseline Determination:** It determines the baselines to be established, including initial baselines and subsequent configuration baselines, to effectively manage and track changes throughout the project lifecycle.

**Change Control Methods:** The CMP describes the methods for controlling, tracking, implementing, and reporting changes to configuration items. It ensures proper authorization, documentation, and communication of all changes made to project artifacts.

**Configuration Status Accounting:** Procedures are established for maintaining configuration status accounting, including the recording and reporting of changes made to baselines. This ensures the integrity and accuracy of configuration data throughout the project.

**Configuration Audits:** The CMP outlines the configuration audits to be performed to validate compliance with CM processes and identify any discrepancies or non-conformities. It also identifies the organizations responsible for conducting these audits.

**Utilization of CM Software Tools:** Documentation of how CM software tools will be used to support project-level CM activities, including version control, change tracking, and collaboration functionalities. This ensures the effective utilization of technology to streamline CM processes and enhance project efficiency.

## Assumptions/Constraints

**Assumption:** The assumption that users will have access to stable internet connectivity to utilize the Foodies Web App for discovering nearby restaurants and placing orders.

**Assumption:** The assumption that users will possess compatible devices (smartphones, tablets, or computers) with modern web browsers to access and interact with the Foodies Web App.

**Constraint:** The constraint of adhering to data privacy regulations and ensuring the secure handling of user information, including personal data and payment details, in compliance with relevant laws and standards.

**Assumption:** The assumption that participating restaurants will maintain accurate and up-to-date menu information on the Foodies Web App platform to facilitate user decision-making and ordering.

**Assumption:** The assumption that users will actively engage with the loyalty points system, contributing to the accumulation of points based on their usage of the Foodies Web App for ordering food.

**Assumption:** The assumption that the Foodies Web App will integrate seamlessly with third-party payment gateways to facilitate secure and convenient transactions for users placing orders.

**Constraint:** The constraint of budget limitations, requiring prudent resource allocation and cost management throughout the development and maintenance phases of the Foodies Web App project.

**Assumption:** The assumption that the Foodies Web App will attract a sufficient user base and generate revenue through partnerships with restaurants and promotional activities, ensuring long-term sustainability and growth.

**Assumption:** The assumption that the project team members will possess the necessary skills, expertise, and commitment to successfully develop, launch, and maintain the Foodies Web App according to project timelines and quality standards.

**Constraint:** The constraint of time, necessitating efficient project planning and execution to meet project milestones and deliverables within the designated timeframe.

## Roles and Responsibilities

<<Below is a table summarizing the key roles and responsibilities involved in executing the project's Configuration Management Plan. Modify or adjust to meet project-specific requirements>>

| Role | Responsibility |
| --- | --- |
| Configuration Manager | * Develop, implement, and maintain the Configuration Management Plan (CMP) based on project needs. identify and control system software, hardware, and documentation. Establish naming conventions and baselines. Track changes and ensure accurate documentation. Conduct configuration audits and maintain records. Use CM software tools for project support. |
| Project Manager | * Ensure CMP aligns with project goals. Allocate resources and monitor progress. Communicate with stakeholders. Approve baselines and changes. Provide support to Configuration Manager and team. |
| Development Team | * Follow CMP guidelines. Document changes properly. Use version control and labeling. Participate in audits and address issues. Collaborate with Configuration Manager. |
| Quality Assurance Team | * Review and verify CMP compliance. Conduct audits and report findings. Ensure integrity of configuration items. Provide improvement suggestions. |
| Stakeholders | * Provide feedback on CMP. Review and approve baselines and changes. Communicate concerns to Project Manager or Configuration Manager. Assist in audits and documentation. |

# Configuration Management Strategy

## Configuration Management Strategy Overview

*For our project, we have devised a robust Configuration Management Strategy aimed at facilitating the setup and management of project environments effectively. This strategy delineates a clear path for maintaining consistency and control over project configurations across two primary branches:* ***'main'*** *and* ***'master'****.*

*The* ***'main'*** *branch serves as the primary line of development. It is where integration of features and bug fixes occurs. All changes are thoroughly reviewed and tested before being merged into this branch, ensuring stability and reliability.*

*The* ***'master'*** *branch, on the other hand, is our production-ready branch. Only thoroughly validated and approved changes from the 'main' branch are merged into 'master'. This branch is strictly reserved for deployment purposes, ensuring that only thoroughly vetted code reaches our live environment..*

## Documentation Repositories

*The overall strategy for our project's usage of documentation repositories involves establishing a centralized and organized system for storing, managing, and accessing project documentation. We will utilize GitHub repositories to house various types of documentation, including technical specifications, design documents, user manuals, and project plans.*

***Key aspects of our strategy include:***

***Repository Structure:*** *We will define a clear structure for organizing documents within (****GitHub)*** *repositories, ensuring easy navigation and retrieval of information.*

***Version Control:*** *GitHub's version control features will be leveraged to track changes made to documents over time, enabling us to maintain a history of revisions and revert to previous versions if necessary.*

***Access Control:*** *Access to repositories and specific documents will be carefully managed to ensure that only authorized individuals can view, edit, or contribute to project documentation.*

***Collaboration: GitHub's*** *collaboration tools, such as pull requests and comments, will be utilized to facilitate collaboration among team members during the document creation and review process.*

***Documentation Standards:*** *We will establish and adhere to documentation standards to maintain consistency and quality across all project documentation.*

***Documentation Updates:*** *Procedures will be put in place to regularly review and update project documentation to reflect changes in requirements, design decisions, or project progress.*

## Technical Environments

***Environment Setup:***

*Dev Environment: This environment will be set up for developers to work on code changes, new features, and enhancements. It will closely mirror the production environment to facilitate accurate testing and development.*

*Test Environment: This environment will be dedicated to testing new features, bug fixes, and integrations. It will mimic the production environment as closely as possible to identify and resolve issues before deployment.*

*Prod Environment: The production environment is where the live application will be deployed and accessed by end-users. It must be stable, secure, and highly available to ensure optimal performance.*

***Configuration Management:***

*Version Control: Git, integrated with GitHub, will be used for version control, allowing for the tracking of changes across environments and enabling collaboration among team members.*

*Configuration Files: Configuration files for different environments (e.g., database connection strings, API endpoints) will be managed separately, with clear naming conventions to avoid confusion.*

***Deployment Pipeline:***

*Continuous Integration/Continuous Deployment (CI/CD): A CI/CD pipeline will be implemented to automate the build, testing, and deployment processes. This will ensure consistent and reliable deployments across all environments.*

*Automated Testing: Automated testing scripts will be integrated into the CI/CD pipeline to validate code changes in different environments, reducing the risk of introducing bugs into production.*

***Environment Isolation:***

*Each environment (Dev, Test, Prod) will be isolated from one another to prevent disruptions caused by changes made in one environment affecting others.*

*Access controls will be implemented to restrict access to production environments and sensitive data, ensuring security and compliance with data protection regulations.*

***Monitoring and Logging:***

*Monitoring tools will be set up to track the performance, availability, and health of each environment in real-time. This will help detect and resolve issues proactively.*

*Centralized logging will be implemented to capture and analyze logs from all environments, aiding in troubleshooting and root cause analysis.*

*Scalability and Disaster Recovery:*

*Scalability measures will be incorporated into the design of each environment to handle increases in traffic and workload.*

*Disaster recovery plans will be developed to ensure business continuity in the event of system failures or disasters, with regular backups and redundancy mechanisms in place.*

## Development Environment

*The development environment for our project is where developers work on coding, testing, and refining software before it's ready for use. Here's what it involves:*

***Purpose:*** *It's where developers write and test code, ensuring it works properly.*

***Configuration:*** *It's set up to match the final product closely, with the tools and resources developers need.*

***Isolation:*** *It's separate from other environments to avoid disruptions from unfinished work.*

***Collaboration:*** *Developers use tools like Slack and GitHub to work together and review each other's code.*

***Testing:*** *Basic testing is done to catch problems early and ensure the software works as expected.*

### Test Environment

***Purpose:*** *The test environment ensures our software works as intended before it goes live. It helps us find and fix any issues or bugs early on.*

***Setup:*** *Our test environment closely resembles the real thing, mimicking the same setup as our live system. This ensures accurate testing results.*

***Isolation:*** *We keep the test environment separate from the live one to avoid any disruptions. It's a safe place for testing without affecting users or data.*

***Testing Types:*** *We use the test environment for different types of testing, like checking if everything works correctly (functional testing), making sure it can handle lots of users (performance testing), and ensuring it's secure (security testing).*

***Tools:*** *We have tools to help with testing, like automated testing tools and programs that help us manage test data. These tools make testing faster and more efficient.*

### Tools Used for Environment Management

*<<Describe the tools used to support the configuration management activities in the table below.>>*

| Tool used to manage technical environments | Tool Description and Functions |
| --- | --- |
| GitHub | Version control system for code collaboration |
| Git | Distributed version control system |
| HTML/CSS | Markup and styling languages for web content |
| JavaScript | Programming language for dynamic web content |
| Java | Programming language for backend development |

**Table 2: Tools Used for Environment Management**

-

### Environment Schedule

| Environment | Build Time | Milestones |
| --- | --- | --- |
| Development | Week1 |  |
| Testing | Week2 |  |
| Staging | Week3 |  |
| Production | Week4 |  |

# Configuration Items

## Configurable Item Identification and Management

**Project Work Products:** This includes all project documentation, such as requirements documents, design documents, test plans, and user manuals.

**System Software:** The software components of the Foodies Web App, including the frontend developed using HTML, CSS, and JavaScript, as well as the backend developed using Java.

**System Hardware:** Any hardware components required for the system to function, such as servers, databases, and networking equipment.

**Commercial Off-The-Shelf (COTS) Software:** Any third-party software or libraries used in the development of the application, such as web frameworks or database management systems.

**Support Software:** Any additional software tools or utilities developed in-house or by contractors to support the development, testing, or deployment of the Foodies Web App.

**Provider Releases:** Releases of the hosted product by the service provider, including both major and minor releases, should be considered CIs.

Means of identifying changes to the CIs and related baselines will include:

**Version Control System:** Utilizing a version control system such as Git to track changes to code files and other project artifacts. Each change will be associated with a unique commit ID, providing a clear record of modifications.

**Change Control Board (CCB):** Establishing a Change Control Board responsible for reviewing and approving proposed changes to CIs. Any changes to CIs will require formal approval from the CCB before implementation.

**Configuration Management Tools:** Implementing configuration management tools that can track changes to CIs and maintain baselines. These tools will provide visibility into the history of changes and facilitate auditing and reporting.

### Configuration Item Class: Documentation

| # | Controlled Documentation (or class of documentation) | Description |
| --- | --- | --- |
| 1 | Requirements | Documents outlining the functional and non-functional requirements of the Foodies Web App, including user stories, use cases, and system specifications. |
| 2 | Design | Documents detailing the design architecture of the Foodies Web App, including system architecture diagrams, database schema, and API specifications. |
| 3 | Code | Documents containing the source code of the Foodies Web App, including HTML, CSS, JavaScript, and Java files. |
| 4 | Testing | Documents outlining the testing strategy, test plans, test cases, and test results for the Foodies Web App, including unit tests, integration tests, and acceptance tests. |
|  | PMP | Documents related to project management, including the Project Management Plan (PMP), project schedules, meeting minutes, and progress reports. |

**Table** 3**: Controlled Document Configuration Items**

### Configuration Item Class: Software / Code

| # | Controlled Software / Code | Configuration Management System |
| --- | --- | --- |
| 1 | Source Code | All source code files written in HTML, CSS, JavaScript, and Java |
| 2. | Libraries and Dependencies | External libraries, frameworks, and dependencies used within the project. |
| 3. | Database Scripts | SQL scripts or database migration scripts for managing the database schema. |

**Table 4 Controlled Code Configuration Items**

# Configuration Management System

*<<Not all interface may be known at the time of initial project planning, so this section should be updated to reflect environment planning decisions as they are made.>>*

## Change Management

*<<Define the process for controlling changes to the system baselines and for tracking the implementation of those changes. Usually a system change request (SCR) is used to provide information concerning the need to change a baseline system or system component (hardware, software, or documentation). If these processes are detailed in a separate, more detailed plan it should be referenced here.>>*

## Configuration Management System

***a) Project Documentation Management:***

***System Name:*** *Foodies*

***Purpose:*** *Foodies organizes and controls project documents to ensure everyone has the right version at all times.*

***Naming Standards:*** *Documents are named clearly so everyone knows what they are.*

***Directory Structure:*** *Documents are sorted into folders by type (like Requirements, Design, etc.) for easy access.*

***Access and Version Control:*** *Only authorized people can access documents, and we keep track of changes to make sure everyone has the latest version.*

***b) Backup and Recovery:***

***Frequency:*** *We back up documents every day to prevent data loss.*

***Recovery Exercise:*** *We practice restoring backups every three months to make sure it works when we need it.*

***c) Removing Old Documents:***

***Selection Process:*** *We regularly check for old or unnecessary documents and archive them.*

***Storage:*** *Archived documents are kept safe for a while before we delete them for good.*

***Responsibilities:*** *The person in charge of document management takes care of this.*

***d) Security and Clearance:***

***Controlled Access:*** *Only authorized people can see sensitive documents to keep them secure.*

***Clearance Requirements:*** *People who handle sensitive info must pass security checks.*

***e) Updating Documents:***

***Change Control Integration:*** *Our document system is linked to our change process, so when something changes, we know to update the documents.*

***Notifications:*** *The person in charge gets a heads-up when a document needs updating, so nothing gets missed.*

# CIL

For our project, the configuration item list is like a project checklist. It helps us keep track of all the important things we need, like software, documents, and hardware. This list makes sure we don't forget anything and helps us stay organized. It's like a roadmap that guides us through the project and makes sure everything runs smoothly.

For a detailed Configuration Item List template, you can find it **here**.

# Appendix A: Key Terms

Table below provides definitions and explanations for terms and acronyms relevant to the content presented within this document.

|  |  |
| --- | --- |
| Term | Definition |
| CM | Configuration Management - The process of managing and controlling changes to software, hardware, documentation, and other system components throughout the project lifecycle. |
| WBS | Work Breakdown Structure - A hierarchical decomposition of the project into manageable work packages. |
| CI | Configuration Item - A product or artifact that is subject to configuration management. |
| SCM | Software Configuration Management - A subset of configuration management focused specifically on software development projects. |
| COTS | Commercial Off-The-Shelf - Pre-built software or hardware components that are purchased or licensed for use in the project. |
| PMP | Project Management Plan - A formal document that outlines the project management approach, processes, and procedures for a specific project. |
| SPI | Schedule Performance Index - A measure of project schedule efficiency, calculated as the ratio of earned value to planned value. |
| CPI | Cost Performance Index - A measure of project cost efficiency, calculated as the ratio of earned value to actual cost. |

**Table 5: Key Terms**