Configuration management plan

Clothing Store

Document History

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

# Contents

[1. Overview 6](#_Toc448133047)

[1.1. Project Background 6](#_Toc448133048)

[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)

[2.4.6. Tools Used 8](#_Toc448133062)

[3. Configuration Items 9](#_Toc448133066)

[3.1. Configurable Item Identification(Name Conventional) 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)

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

[4.1. Change Management](#_Toc448133074)

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

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

5. Baseline

6.Naming convention

# Overview

The Online Clothes Store Configuration Management Plan aims to provide a solid foundation for handling the variability involved in software development and implementation. Throughout the project lifecycle, this document acts as a guide to ensure the integrity and traceability of configuration objects.   
• Goal: This plan's main objective is to guarantee that all system modifications are carried out in a coordinated and regulated way.  
• Scope: The plan includes all capabilities of the Online Clothes Store web portal, such as user account management, client and supplier interfaces, and transaction history features.   
• Objectives: We aim to ensure the dependability of the online platform, maintain system consistency with project requirements, and systematically track modifications.

## Project Background

*With the goal of revolutionizing the shopping experience by offering a seamless interface between suppliers and customers in the clothing business, the Online clothing Store project is an enormous undertaking.*

* *Project Inception: The project was started with the goal of developing a comprehensive platform for clothing transactions in response to the increasing need for online retail solutions.*
* *Business demand: The project addresses the demand for a centralized platform that enables suppliers to effectively manage their sales and customers to easily buy clothing and accessories.*
* *Project Objectives: We want to create an intuitive online gateway that facilitates safe login, account administration, and a flexible marketplace for purchasing and vending products.*
* *Strategic Alignment: By utilizing technology to improve consumer engagement and expedite corporate processes, this project is in accordance with the strategic vision.*
* *Benefits: Higher sales volume, better operational efficiency, and higher customer satisfaction are among the anticipated benefits.*

# Configuration Management Strategy

## Configuration Management Strategy Overview

Our configuration management strategy defines a clear path to maintaining consistency and control of project configurations across two primary branches:

1. The **‘main'** branch:

* serves as the primary line of development.
* It is where integration of features and bug fixes occurs.

1. The **'master'** branch:

* production-ready branch.
* Only thoroughly validated and approved changes from the 'main'.

## Documentation Repositories

The repository plays a pivotal role in ensuring transparency, collaboration, and knowledge sharing among team members.

Our overall strategy revolves around the following key principles:

1. **Main Repository:**

* We maintain a repository accessible to all project stakeholders.
* This repository includes requirements, design documents, technical specifications, user manuals, and any other relevant materials.

1. **Repository Hierarchy:**

This hierarchy includes folders for different types of documents, such as:

* + **Project Management:** Including project plans, schedules, and status reports.
  + **Technical Documentation:** Including architecture diagrams, API specifications, and system documentation.
  + **User Documentation:** Including user guides, tutorials, and FAQs.
  + **Quality Assurance:** Including test plans, test cases, and testing reports.

## Tools Used for Environment Management

| Tool used to manage technical environments | Tool Description and Functions |
| --- | --- |
| IntelliJ IDE | IntelliJ IDEA is an integrated development environment (IDE) primarily used for Java development. It provides advanced code editing, debugging, and refactoring tools, along with support for various programming languages and frameworks. |
| Chrome browser | Google Chrome is a popular web browser known for its speed, simplicity, and security features. It is used for testing web applications and accessing web-based tools and resources. |
| Excel | Microsoft Excel is a spreadsheet software used for organizing, analyzing, and visualizing data. It is commonly used for test case management, data manipulation, and reporting in software testing. |
| Figma | Figma is a collaborative interface design tool used for creating, prototyping, and sharing user interface designs and interactive prototypes. It allows teams to collaborate in real-time and streamline design workflows. |
| Git / GitHub | Git is a distributed version control system used for tracking changes in source code during software development. GitHub is a web-based Git repository hosting service that facilitates collaboration and code sharing among developers. |

**Table 3: Tools Used for Environment Management**

# Configuration Items

## What is Configuration Item

## A configuration item (CI) is any component or asset within a system or software environment that is managed and controlled as part of the configuration management process. CIs are typically identified, documented, and tracked to ensure consistency, reliability, and traceability throughout the development, deployment, and maintenance lifecycle of a system or software product.

## Configurable Item Identification and Management for the clothing store project

## For front-end code: HTML files, CSS stylesheets, JavaScript files.

## For back-end code: Server-side scripts, API endpoints.

## For databases: Database schema, SQL scripts.

## For images: Logo files, product images.

## For documentation: User manuals, design specifications.

## How to define the configuration level

* **Baseline Configuration:**

Represents stable settings and parameters foundational to the software, established early and undergo minimal changes.

* **Frequently Changing Configuration:**

Includes dynamic configurations for development, testing, and deployment, subject to frequent updates to support agile development practices.

* **Infrequently Changing Configuration:**

Encompasses settings for production, release-specific parameters, or compliance configurations, updated less frequently but still requiring careful management.

# Configuration Management System

1. **Project Documentation Configuration Management System:**

* **Tool Name**: GitHub
* **Purpose:** GitHub serves as the central repository for all project documentation, including requirements documents, design specifications, test plans, and user manuals. Its primary purpose is to ensure version control, access control, and traceability of project documents throughout the project lifecycle.
* **Directory Structure Standards:** Documents are organized into folders based on their categories, with subfolders for each document type. The directory structure follows a hierarchical arrangement for easy navigation and retrieval.
* **Owner:** The Project Manager is responsible for the administration and maintenance of the Configuration Management System.
* **Access and Version Controls:** All the team have access to all files. Version control is managed using a check-in/check-out system, with each document version tracked and recorded in the system.

1. **Backup and Recovery Approach:**

* **Frequency of Backup**: Full backups of the GitHub repository are performed weekly.
* **Recovery Exercised and Validation**: Regular recovery exercises are conducted weekly to ensure the integrity and availability of backup data.

1. **Tools and Techniques for Updating Controlled Documents:**

* Changes to controlled documents are initiated through an approved change request process, where stakeholders submit change requests detailing the proposed modifications.
* Upon approval, the Project Manager is notified to update the relevant document in GitHub, ensuring that the latest version is available to project stakeholders.

1. **Approach for ignoring of Project Documents:**

* **Document Selection Process:** Documents that have undergone changes through approved change requests and are deemed no longer necessary are marked as "ignored".
* **Responsibilities**: The Project Manager oversees these process.

# Baseline

* **process**

before each deliverable the project manager does the following:

1. ensure that each task is reviewed and approved.
2. ensure that there is evidence on review and approval and the version is the latest one.
3. close completed tasks on taskade.
4. take baseline.
5. send email for coach and quality representative.

# Naming convention

* The guidelines listed below must be adhered to:  
    
  CI Naming Format: A structured format containing the project identifier, CI type should be used to name each CI. OCS\_[CI Type] is one example.

**Naming Convention for Code, Baselines, and Branches**

*the format: Train-Case*

Document Naming Convention

* **Format**: OCS\_Document\_[ Document Name]
* **Example**: OCS\_Document\_ProjectProposal

Breakdown of components:

Code Naming Convention

* **Format**: OCS\_Code\_[ file name]
* **Example**: OCS\_Code\_Payment

**Baselines** Naming Convention

* **Format**: OCS\_Baseline\_[Milestone]\_[Date]
* **Example**: OCS\_Baseline\_Launch\_20240328

Versioning: A version number should be in the name of every CI and should take the following format: X is the main version and Y is the minor version, abbreviated as vX.Y.   
OCS\_Documentation\_v1.2 is one example.

In our project: X stands for baseline number, and Y stands for normal versions.

# 