**Design Document**

1. Introduction
   1. Purpose of the document

The software design document is meant to provide a clear description of the web application for the individual project of the 3rd semester. The document’s purpose is to provide a clear understanding of what needs to be built and how it is expected to look like at the end.

The design document will cover the frontend as well as the backend of the software, thus giving a clear insight of every aspect of the design of the application.

* 1. Document Overview

This document serves as a description of the functionality and software design for ”ComfyShop” application.

1. System Overview

For the 3rd semester of ICT & Software Engineering, students were asked to make a full-stack web application. The project must contain certain guidelines and requirement by using the agile software development methodology. The web application will work as an e-commerce platform where enable customers to browse, search, and purchase products from different categories online. Customers will be able to add products to their cart, view their order history. Additionally, the application will provide the company with access to manage product listings, update prices and descriptions.

1. System Architecture
   1. Architectural Design

The application is developed keeping in mind the SOLID principles and ensure that security is provided to the users. Interfaces are used to ensure the connection to the database in order to avoid repetition in the implementation. The backend of the application is connected with the front-end by using HTTP requests with AXIOS in order to connect to the REST endpoints. The C4 diagram below shows how the system works

C4 diagram consists of 4 levels: System Context(C1), Container (C2), Components (C3) and Code (C4).

Level 1: System Content

Diagram

Description automatically generated

The above diagram shows the various types of users of the application, namely the Site Visitor, Client and Administrator, each user has s distinct type of accessing the website.

The ComfyShop System is the main software part that enables the client to browse various categories and products, add them to their cart, and place an order. The administrator is responsible for managing the categories, products, and other related tasks.

Level 2: Containers

Diagram

Description automatically generated

The diagram describes the relationship between different users and systems, as well as the connection between the back-end and front-end with the database. The front-end container is responsible for providing the user interface (UI) and all website functionalities within the browser. Users and administrators use the UI to interact with the back-end, which provides all the data and logic for the website using Spring Boot and JSON. The back-end makes the front-end functionalities possible by processing the user requests and providing responses accordingly. Finally, the database stores all the data related to users/content and directly interacts with the back-end to ensure the availability of the required data.

Diagram

Description automatically generated

Level 3 represents components which are used in the functionality of the application. Controllers are used in order to connect the back end with front-end. With the help of interfaces, the connection with the service components is flexible. Service components are the logic which the controllers basically use. They are directly connected to data access layers and can CRUD data. Database is used to store all the data of all users/content and is directly interacting with the backend.

Diagram

Description automatically generated

1. Software Choices

In order to back up the software choices for this project, the DOT framework methodology has been used to see which technologies would fit best for this project.

* Spring Boot

According to (MULDERS, 2019) Spring Boot is an open-source micro framework, maintained by a company called Pivotal which provides Java developers a platform to get started with an auto configurable Spring application. Therefore, Spring Boot is a good framework to get started with the back-end part for full-stack web development, especially for beginners thanks to the fact that it reduces development process and increases efficiency by having a default setup for unit and integration tests. Therefore, thanks to its features, we are obliged to use Spring Boot this semester for this individual project.

Spring Boot offers some advantages such as productivity, reduction of development time and it’s easy to understand. The framework’s goals are to avoid complex XML configurations (this is a big plus especially for people who are beginning full stack web development) and developing Spring applications in an easier way.

The most important thing about Spring Boot would be that everything is auto configured, so there is no need for manual configurations. It is worth mentioning that the Java version used for this application is Java 11.

* React JS

Nowadays, there are a lot of frameworks to choose from, ranging from React JS, Angular, Vue JS, Django etc. each one having its unique functionalities depending on the complexity of the project. Since, ComfyShop is not a very complex project, it has been decided to use React JS. This framework works with components, each function being considered as a component. Therefore, separating concerns is crucial. For beginners, React is much more easier to get started with comparing to the other mentioned frameworks, this being the reason why we have been asked to use it for this semester by our teachers. Also, React is quick, efficient, works with an MVC template and makes creating front-end more easy. Thanks to the fact it has a large community, you can easily find anything about different problems you might encounter while creating your application.

* MySQL

MySQL is a popular and widely used relational database management system. It is known for its reliability, scalability, and ease of use.

Moreover, MySQL is free and open-source, everyone can use it for small business.

* CI/CD Pipeline Diagram

A picture containing text, screenshot, diagram, design

Description automatically generated

1. The code is pushed from the Intellij IDE to the GitLab repository.
2. GitLab CI/CD triggers the pipeline based on the push event.
3. Docker is introduced for containerization, represented as a component between GitLab CI/CD and the Pipeline Runner.
4. The Docker component handles the creation and management of Docker containers.
5. The Pipeline Runner executes the pipeline steps within the Docker container.
6. SonarQube performs code analysis on the code within the container.

Explanation of the setup:

1. Developer's Machine (IntelliJ IDE): This represents your local development environment where you write code using IntelliJ IDE.
2. Push to GitLab Repository: Once you have written and tested your code locally, you push the changes to the GitLab repository. This triggers the GitLab CI/CD pipeline.
3. GitLab CI/CD Pipeline: The code changes are detected in the GitLab repository, and the CI/CD pipeline is triggered. The pipeline is configured with different stages and jobs.
4. Pipeline Runner: The pipeline runner is responsible for executing the stages and jobs defined in the pipeline. It pulls the code from the GitLab repository and performs the specified actions.
5. SonarQube (Code Analysis): In the provided code, the "sonar" stage is included, which runs the SonarQube analysis on the code. The analysis helps identify code quality issues, bugs, vulnerabilities, and other metrics related to code health.