**Software Architecture Design**

**(SAD)**

***DaClothes***

Author: Rodrigo Bedoya

**Introduction:**

This Software Architecture Document serves as a comprehensive guide to the architectural choices and strategies I will use in the development of the online marketplace project. It outlines the fundamental structure of the solution and provides a roadmap for its construction. The primary goal is to create a robust, scalable, and secure platform that meets the diverse requirements of the client.

**Architecture Constraints and Design Decisions**

I have chosen the technology stack for my project to align with specific architectural constraints and design considerations. Here is why I have opted for Spring Boot, React and MySQL:

**Spring Boot (Backend)**

* Fast development: Spring Boot gives an easier, quicker path to set up, configure, and run apps. It eliminates the heavy lifting of configuration that is required for setting up most Spring-based apps.
* Robust Ecosystem: The Spring ecosystem offers a rich set of libraries and tools for building enterprise-grade applications.

**React (Frontend)**

* Component-Based Architecture: The component-based model React uses, allows me to create reusable UI components, enhancing development efficiency and maintainability.

**MySQL (Database)**

* Data modeling Capabilities: MySQL excels in data modeling, supporting complex relationships, constraints, and indexing, making it a perfect match for our project’s data storage requirements.

**System Context (C1)**

A diagram of a user

Description automatically generated

This System Context Diagram provides a high-level view of how my online marketplace system interacts with users and external services. It serves as a foundation for understanding the broader system architecture and the flow of information between different users.

**Container (C2):**

A diagram of a software application

Description automatically generated

This diagram illustrates the major containers within the online marketplace system, highlighting the interaction between the front-end and back-end components. This Level 2 (Container) diagram offers a more detailed perspective of the system's architecture by focusing on the major components and their interactions. It serves as a valuable reference for me and stakeholders involved in the project.

**Component (C3):**

A diagram of a product

Description automatically generated

This diagram provides a more detailed view of the major components within the Front-End and Back-End containers of the online marketplace system. The Level 3 (Component) diagram serves as a blueprint for the internal structure of my application, helping me understand the relationships and responsibilities of each component.

**Code Diagram:**

A diagram of a product

Description automatically generated

This is my UML Class Diagram. This helps me while implementing the application. OrderItem and CartItem have a one to one relationship with the Product, because there exists only one product, there are no multiple products of the same kind, since a user is selling a piece of clothing they no longer use.

**CI Setup Diagram**

The diagram below illustrates the flow of the CI pipeline, detailing the interactions between different nodes and tools in the process.

A diagram of a software process

Description automatically generated

Nodes:

1. Developer’s Machine:

Here is where the development happens. Developers write tests, and commit on the machine and that triggers the CI pipeline.

1. GitLab Server:

This is the repository for the project and hosts the source code and CI/CD configuration. It also detects changes on the repository and triggers the CI pipeline.

1. Pipeline Runner:

Executes the CI pipeline defined by the projects configuration, fetches the lates code from the repository and runs predefined stages.

1. SonarQube Instance:

Integrates with the CI pipeline to perform code analysis, checks code quality, identifies bugs and ensures coding standards, and provides feedback to developers and maintains a code quality dashboard.

Also Docker is running on my local machine. Its being used for the pipeline runner and SonarQube.

The CI setup plays a crucial role in automating development workflows, ensuring code quality and facilitating the deployment process.