Software Requirements Specification (SRS)

1. Introduction

Purpose

The purpose of this document is to outline the design and architecture of a Warehouse Management System (WMS). The system aims to streamline and optimize warehouse operations, including inventory management, order fulfillment, and shipment tracking, while distinguishing between the roles of the admin and warehouse users.

Scope

The scope of the Warehouse Management System (WMS) encompasses the functionalities available to both the admin and warehouse users. The admin, responsible for overseeing multiple warehouses, can add, delete, and update products across all warehouses, as well as create and manage warehouse assignments for users. On the other hand, warehouse users are tasked with managing inventory within their assigned warehouses, including viewing, exporting productsThe system also includes integration capabilities with ERP systems and e-commerce platforms. Non-functional requirements such as scalability, performance, reliability, usability, and security are also defined within the scope.

Definitions, Acronyms, and Abbreviations

- **Admin**: Administrator with privileges to manage all the warehouses.
- **User**: Standard user with access to his warehouse only.

Overview

This SRS document details the functional and non-functional requirements of the Warehouse Management System, including user roles, system features, data handling, and external interface requirements.

2. Overall Description

Product Perspective

The Warehouse Management System is an independent module that integrates with existing systems, providing comprehensive warehouse management functionalities.

Product Functions

- User authentication and role management
- Inventory management
- Activity tracking

User Classes and Characteristics

- **Admins**: Manages multiple warehouses (add, delete, increase).
- **Users**: Manages his own warehouse (reduce).

Operating Environment

The system will operate on web browsers and mobile devices, interfacing with a backend server and a MySQL database.

Design and Implementation Constraints

- Must support secure authentication mechanisms.
- Should handle high volumes of real-time data processing via Kafka.
- Must ensure data consistency and integrity within the MySQL database.

User Documentation

User manuals and online help guides will be provided, covering both Admin and Regular User functionalities.

Assumptions and Dependencies

- Users have basic internet access and web browser capabilities.
- The system relies on the proper functioning of the Kafka and MySQL services.

3. System Features

3.1 User Roles and Authentication

Description and Priority

Support two user roles: Admin and Regular User. High priority as it forms the basis of system access control.

Functionality

- **Admin**: Can add, delete and increase the products in the warehouse.
- **User**: Can reduce the quantity of products as he exports them.

3.2 Inventory Management

Description and Priority

Allow admins to manage products across warehouses and warehouse users to manage inventory within their assigned warehouses. Priority is High.

3.3 Activity Tracking

Description and Priority

After the activities of both admin and user, all the movements happened in the warehouses will be tracked and loaded into the db. So that we can have track of all the events.

Functionality

- Process the event data via Kafka.
- Load the processed event data into db.

4. External Interface Requirements

User Interfaces

- **Admin View**: Options for adding products, deleting products, increasing the quantity of the products.

- **User View**: Options for reducing the products quantity as he exports the products.

Hardware Interfaces

No specific hardware requirements, as the system will be web-based.

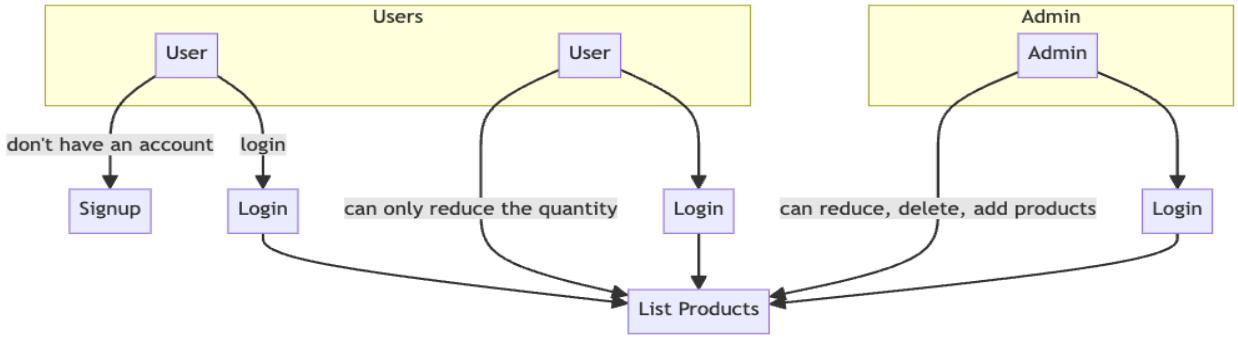
Software Interfaces

- **Backend Server**: Handles API requests.
- **MySQL Database**: Stores user, products and activity data.
- **Kafka**: Processes real-time data for recommendations.
- **GCP**: To host the website online.

Communications Interfaces

Standard HTTP/HTTPS protocols for communication between client and server.

5. System Architecture



6. Other Nonfunctional Requirements

User-Friendly Interface

- The system should have an intuitive and easy-to-use UI design for seamless navigation and operation.

Performance

- Fast response times and effective data processing,

Scalability

- The system should handle a growing database of products and users and increasing user interactions without performance degradation.

7. Detailed Requirements

Functional Requirements

- **FR1**: The system shall authenticate users using secure login credentials.
- **FR2**: The system shall allow Admins to add products.
- **FR3**: The system shall enable users to check the products in their warehouse and can reduce the quantity of products.
- **FR4**: The system shall track all the activities happened in the warehouses.
- **FR5**: The system shall list all the products in all the warehouses for the admin and warehouse specific products for the users.

Non-functional Requirements

- **NFR1**: The system shall process user data securely.
- **NFR2**: The system shall be accessible on various web browsers and mobile devices.
- **NFR3**: The system shall have an intuitive and user-friendly interface.
- **NFR4**: The system shall be scalable to handle a growing database and user interactions.

8. Validation Criteria

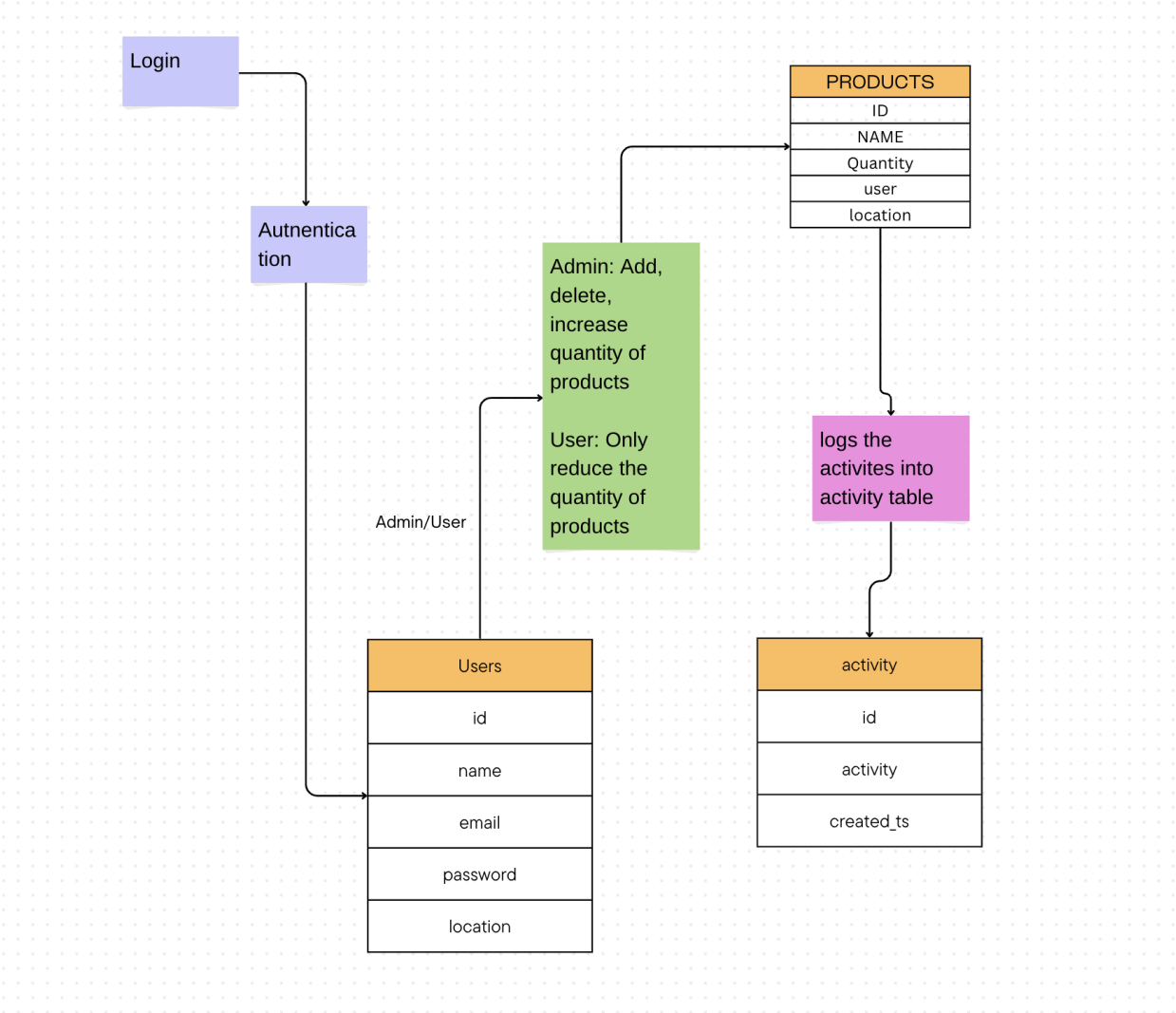
Validation Methods

- User acceptance testing to validate functionality.
- Performance testing to ensure response times.

Success Criteria

- Successful login for both Admin and Regular User roles.
- Admin should be able to add products, delete products, and increase the quantity of the products.
- Users should be able to reduce the quantity of the products.
- All the activities should be tracked and inserted into the db.
- Efficient inventory Management.

9. Database Design



10. Github URLs

WarehousingService Repo: https://github.com/srinadhd1999/WarehousingService

ActivityTrackingService Repo: https://github.com/srinadhd1999/ActivityTrackingService