CST8288_020 OOP

Food Waste Reduction Platform

Computer Programming

Rustom Trayvilla

Vaishali Jaiswal

Seyedeh Mahsa Seyed Danesh

Farock Nathanael Yandom Youmbi

March 20, 2024

1. Introduction

The Food Waste Reduction Platform (FWRP) aims to address the global issue of food waste by connecting food retailers, consumers, and charitable organizations. This document outlines the architecture, database design, and testing approach for the FWRP application.

2. Version History

Date: 2024-01-11

Version #: 0.0

Author name: Rustom Trayvilla, Farock Natanael Yandom Youmbi,Vaishali Jaiswal, Seyedeh Mahsa Seyed Danesh

3. Target Audience

The targeted audience for this document includes:

- Developers involved in building the FWRP application.
- Consumers who want to see reduced price products so that they can purchase by visiting the retailers.
- Charitable organizations who want to see free products so that they can contact retailers to get it for free.
- Project managers oversee the development.
- Quality assurance testers responsible for testing the application.
- Stakeholders interested in understanding the technical aspects of the FWRP.

4. Scope

4.1 In Scope

- High-level overview of the application architecture.
- Database design including entity-relationship diagrams and data models.
- Security considerations for user authentication, data encryption, and secure communication.
- Testing model including unit testing.

4.2 Out of Scope

- Detailed implementation of individual components or functionalities.
- The coding details and algorithm used in building the application.
- Deployment architecture covering infrastructure setup and deployment models.

5. Application Architecture

Presentation Layer: Create JSP pages for visitors, registration and login as a consumer, retailer or charitable organization, listing inventory.

Business Layer: Develop java classes with the use of Apache NetBeans to handle application logic. Use DAO design patterns to abstract and encapsulate all access to the data source. Implement DTOs to transfer data between the DAOs and the business logic classes which will be the servlet classes and java classes.

Database Layer: Set up RDBMS with the use of MySQL and create tables according to ERD. Use JDBC to connect to the database, execute queries, and handle transactions.

Testing: Write JUnit tests for Java methods, especially those handling business logic. Test JSP pages to ensure they correctly display data from the backend and handle user input as expected.

Version Control: GitHub for version control, regularly committing changes and collaborating among team members.

Documentation: MS Office for document creation.

Communication: MS Teams for team communication and collaboration.

UML: UMLet for creating UML diagrams.

6. Business Architecture

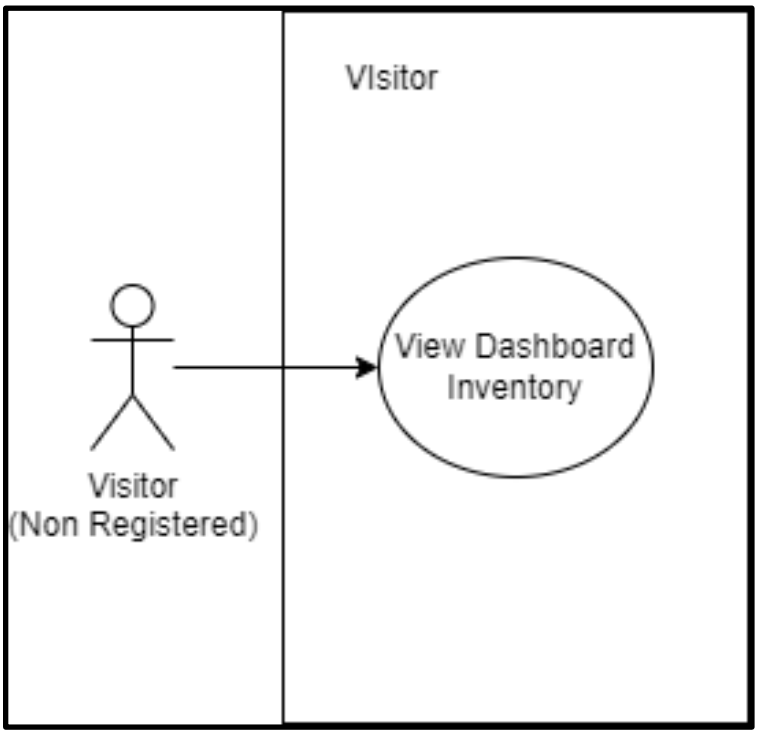

Figure 1: Visitor

Visitor: Any web non-registered and potential consumer/charity can come and check the inventory dashboard listed.

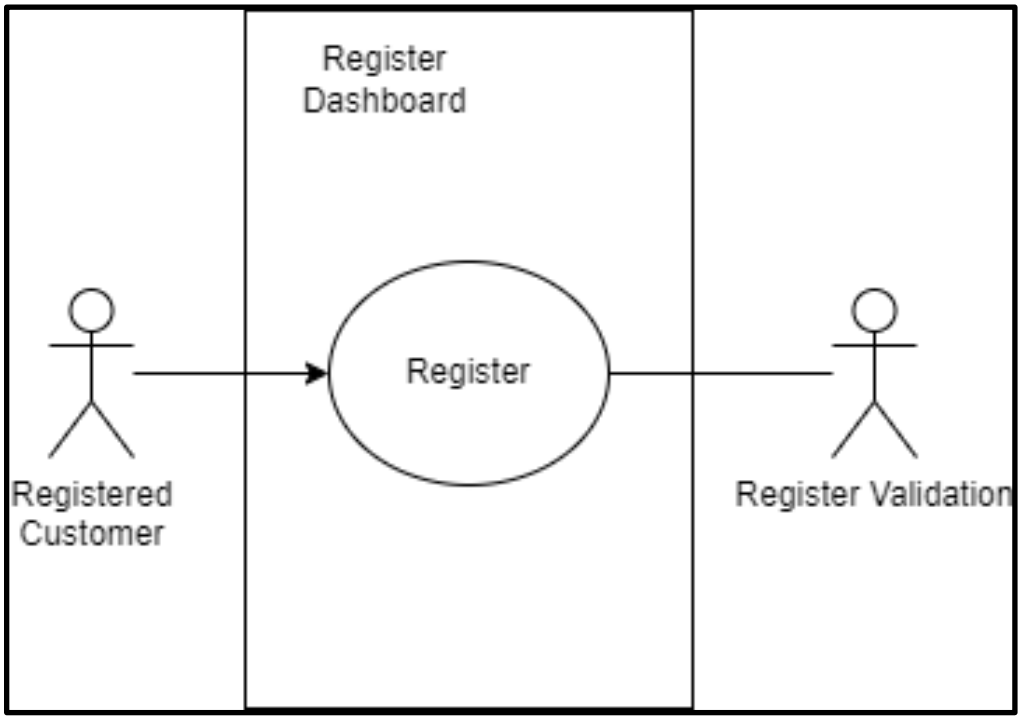

Figure 2: Register Dashboard

User Registration: Users can register on the platform by providing necessary details.

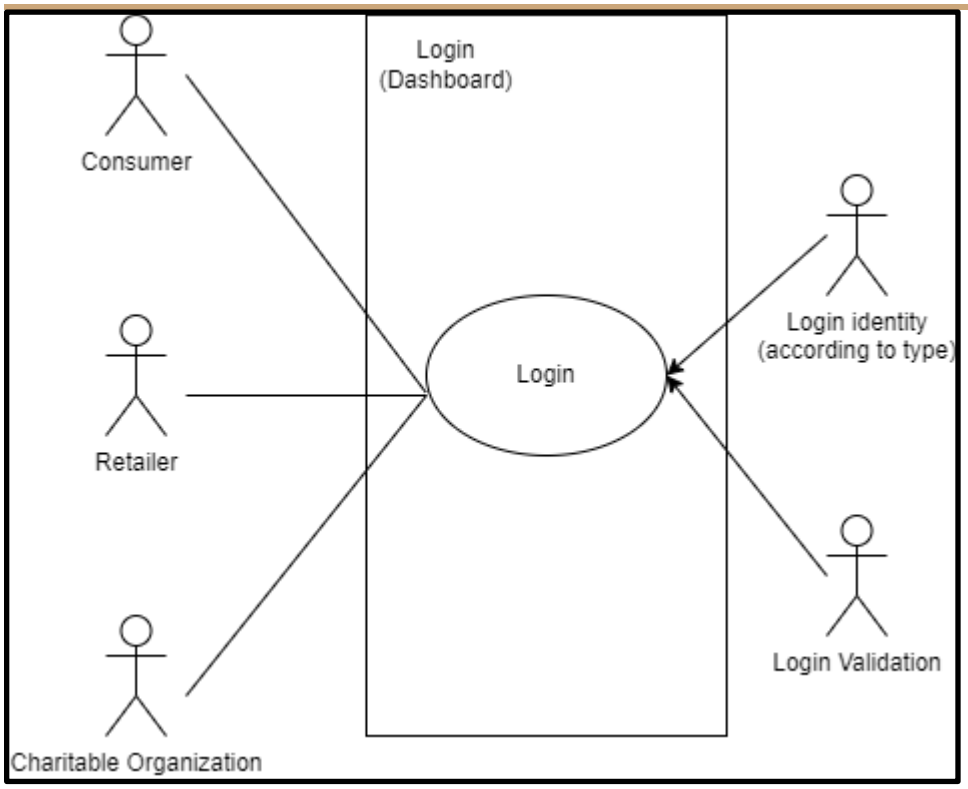

Figure 3: Login Dashboard

Login: Users can log in to access their accounts.

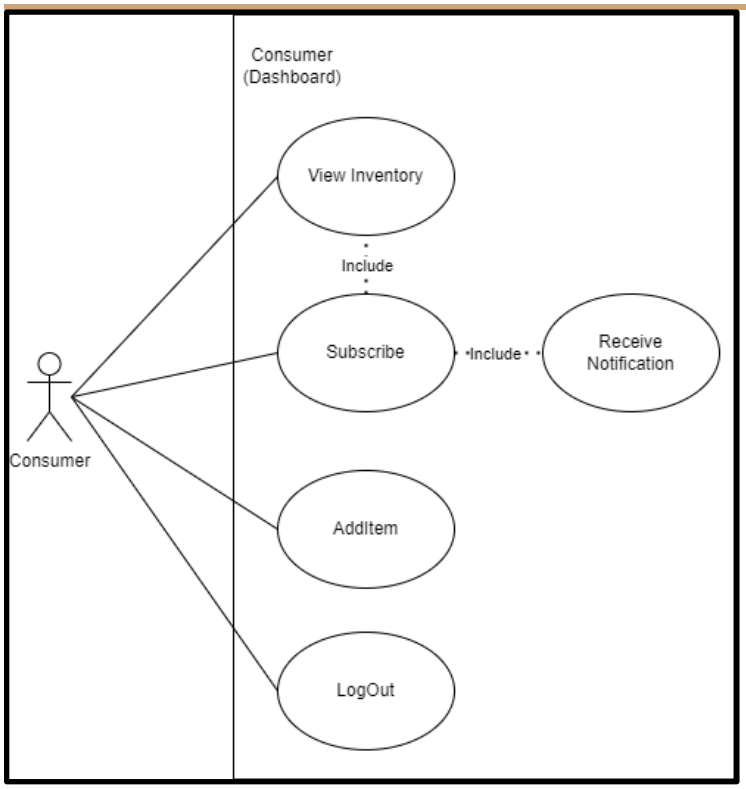

Figure 4: Consumer Dashboard

Subscribe to Alerts: Users can subscribe to receive surplus food alerts based on preferences.

Please note:- Notification component was not able to setup because of time-constraint

Add Item: Consumers can add surplus food items listed by retailers for purchase.

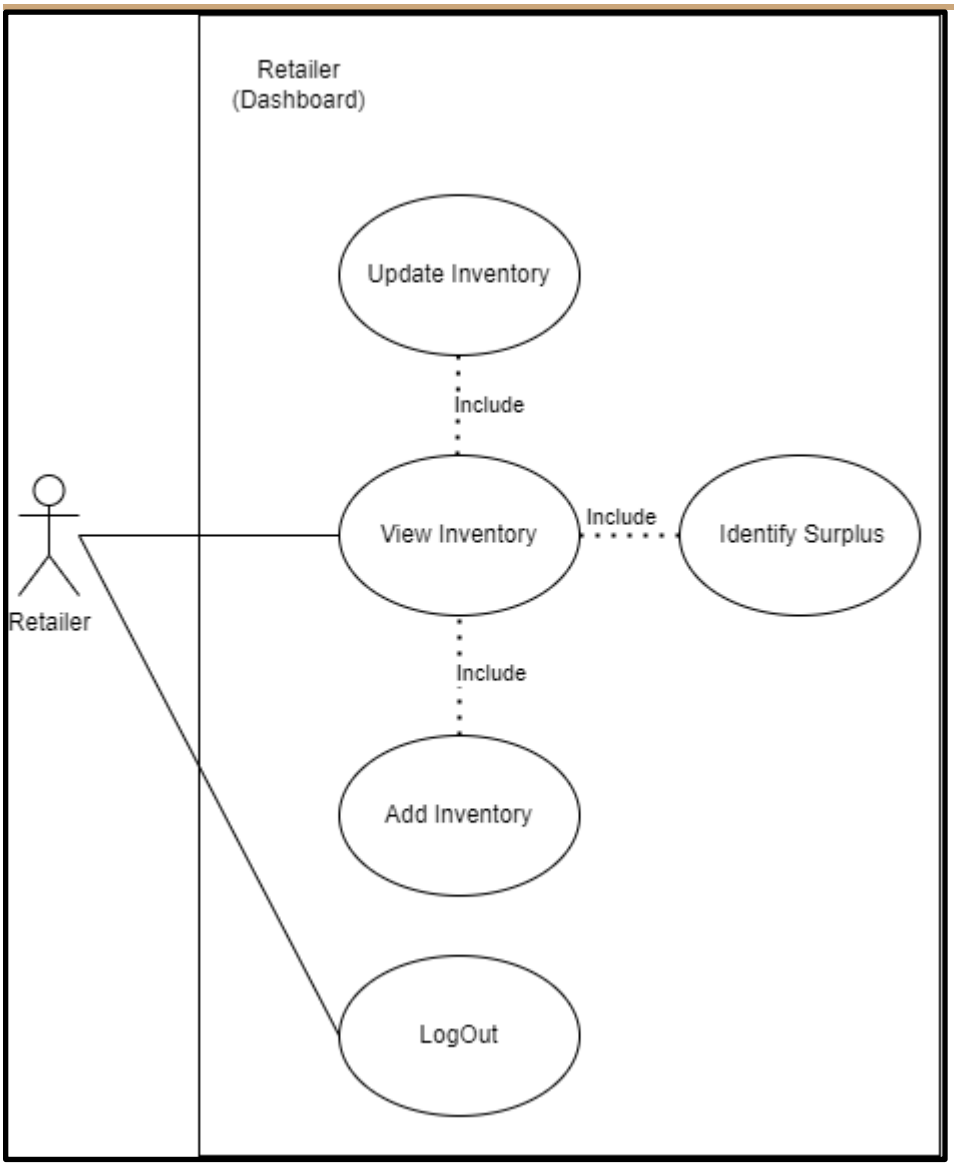

Figure 5: Retailer Dashboard

Manage Inventory: Retailers can manage their inventory by adding, updating, or removing food items.

List Surplus Food: Retailers can list surplus food items for donation or sale.

Identify Surplus Food: Retailers can identify surplus food items nearing expiration or more than demand manually, expiry date selection automation not setup.

.

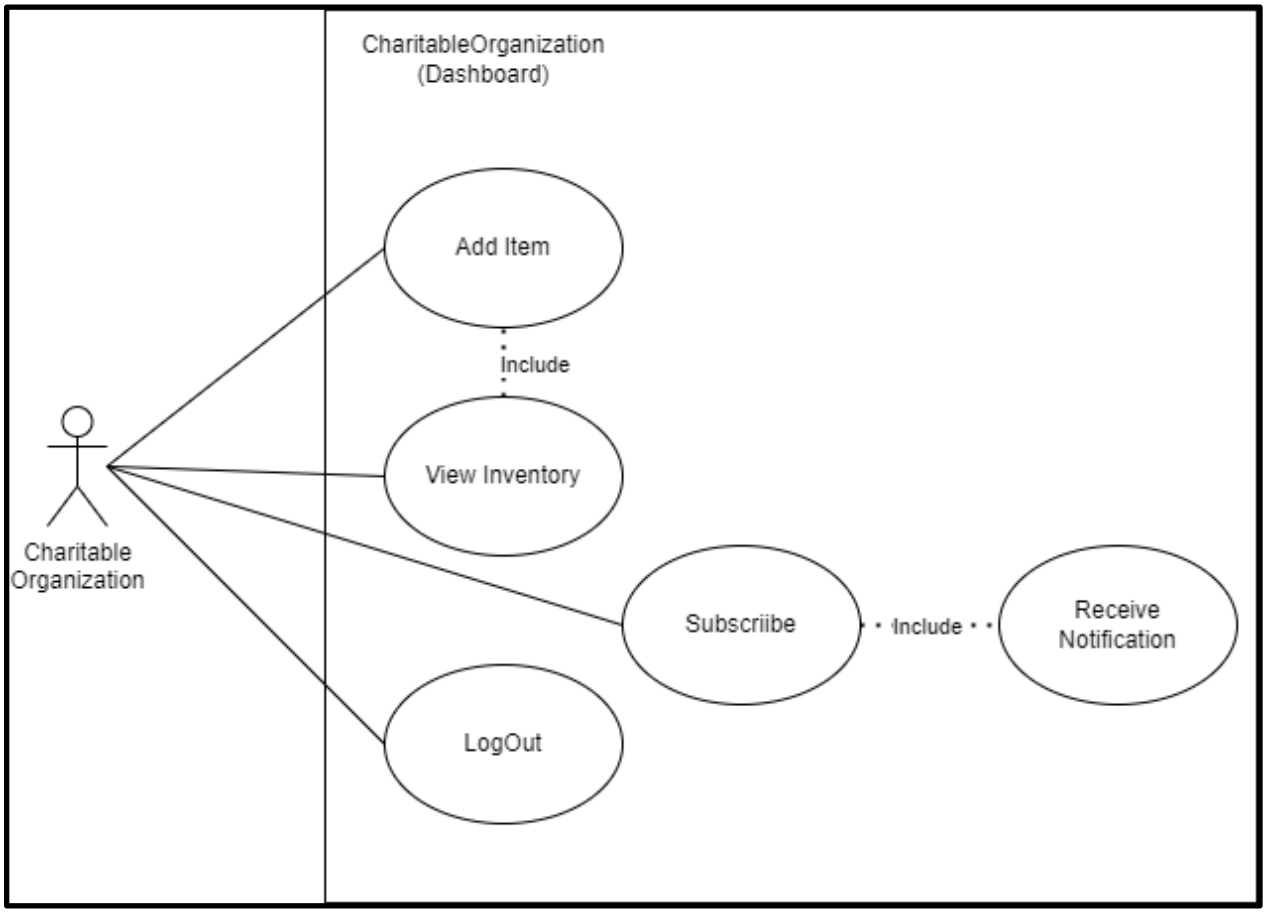

Figure 6: Charitable Organization Dashboard

Add Item: Charitable organizations can add free surplus food items listed by retailers.

Subscribe to Alerts: Users can subscribe to receive surplus food alerts based on item and retailer selection. Message comes showing the subscription is setup.

Receive Notifications: Subscriber's notification was not set up because of time-constraints.

7. Class Diagram

Users: Represents a user of the platform with attributes like name, email, password, and userType. Subclass of User, represents.

Consumer: Subclass of User, represents a consumer with additional attributes like purchase.

Retailer: Subclass of User, represents a retailer with additional attributes like business_name.

CharitableOrganization: Subclass of User, represents a charitable organization with additional attributes like organization name.

Inventory: Manages the inventory of a retailer and represents individual food items with attributes like name, quantity, and reduced price or free for donation.

Subscription: Represents a user's subscription to receive surplus food alerts.

Claims: Tracks claims made by charitable organizations for surplus food items.

UML diagram needs to be updated.

Current UML classes : DAO implementations merged into business logic as the application is small so the class diagram is modified.

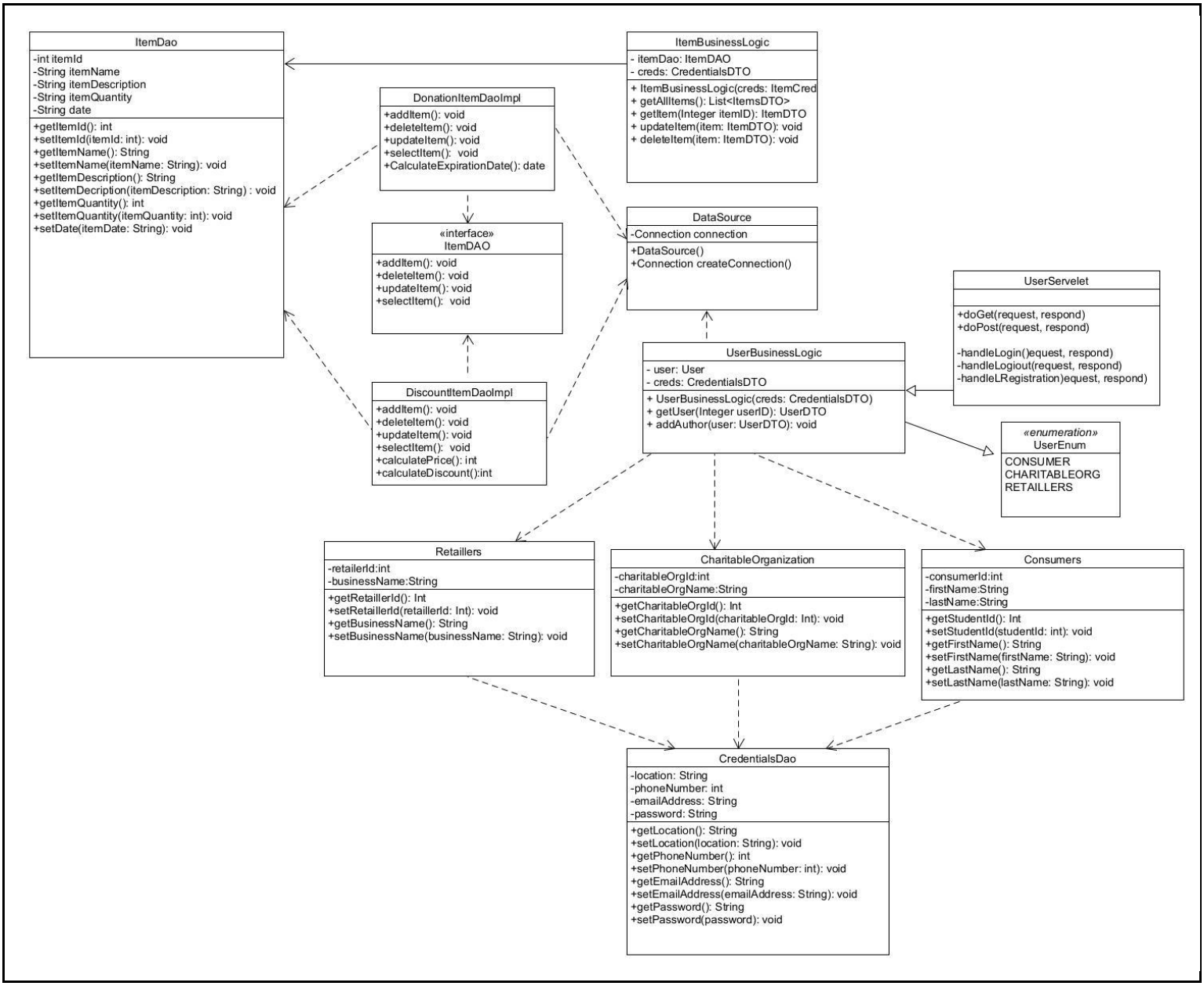

Figure 7: UML Class Diagram

8. Data Architecture(updated)

Users: User Tables hold specific details for each user type, such as consumer, charitable organization and retailers. Each type of user stores basic user information like name, email, password, address and phone number.

Inventory: Tracks food items managed by retailers, including item name, quantity, expiration date, and status.

Subscriptions: Manages user preferences for receiving surplus food alerts, including location, and food preferences.

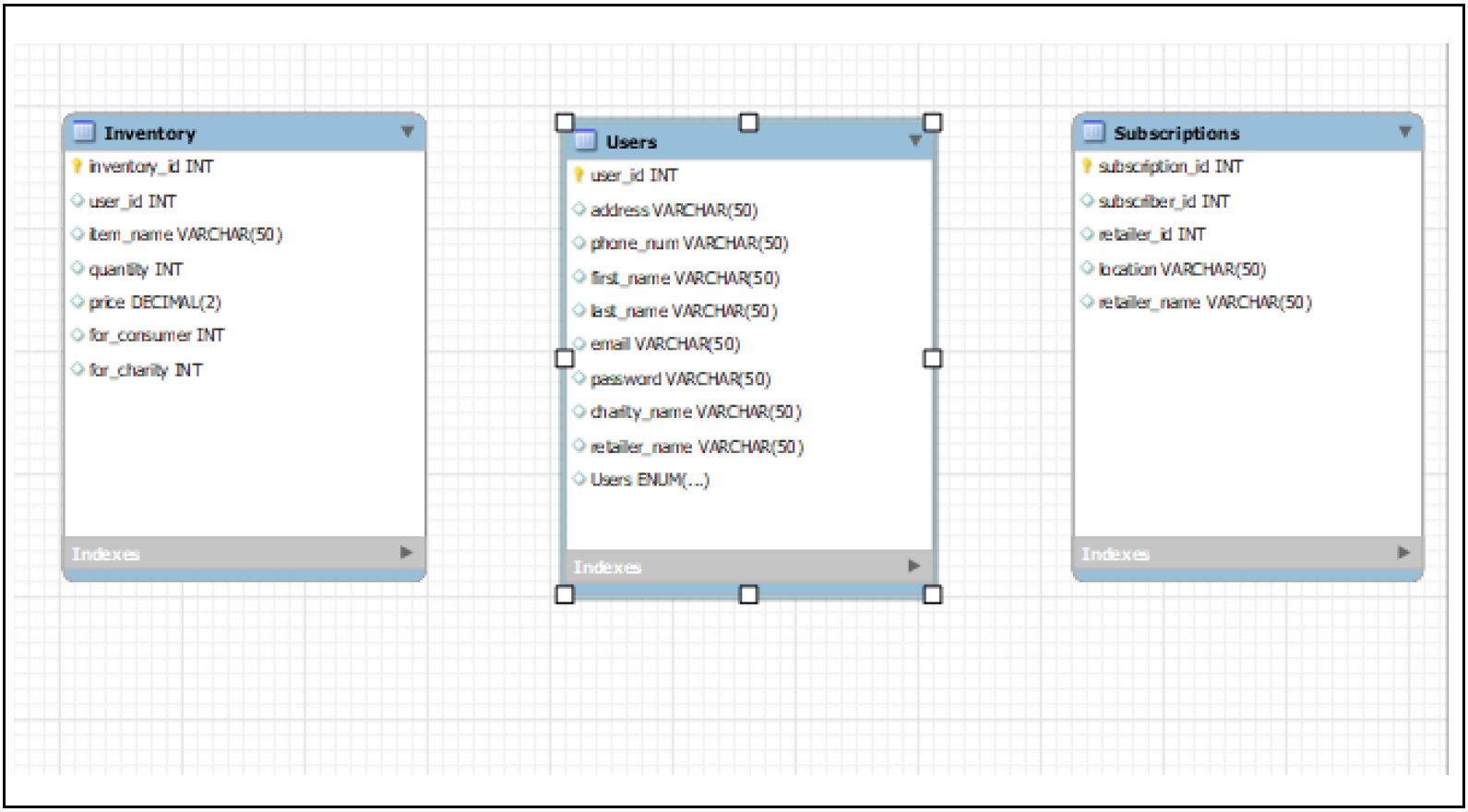

Figure 8: Database FWRP

9. Testing Model

Unit Testing with JUnit: Involves writing and executing unit tests using the JUnit framework to verify the correctness of individual components or units of code in isolation. This ensures that each unit behaves as expected and meets the specified requirements, helping to identify and address any defects early in the development process.