Jarryd Carelse 221267
Milestone Checkup Document
Functional Requirements Checklist

1. **User Authentication**

    - DONE- **User Registration**: Accounts can be created with roles such as farmer, consumer, and admin, allowing personalised access for each user type.

    - DONE- **Login/Logout**: Users can securely log in and out, with session management in place for secure access.

    - BUSY - **Profile Management**: Final adjustments to the profile section are still in progress for full CRUD functionality.

2. **CRUD Operations**

    - DONE-**Product Listings CRUD**: Farmers can create, view, update, and delete product listings. Basic functionalities are working, but I'm refining the user experience to make it more intuitive.

    - DONE- **Order Management**: Consumers and farmers can view and manage their orders. The core CRUD operations are in place, though I'm still testing for efficiency and usability.

    - BUSY - **Profile Updates**: User profile updates are partially functional, allowing for updates to basic details but still needing a few final adjustments.

3. **Shopping Cart and Checkout**

    - DONE- **Add to Cart**: Consumers can add items to their cart seamlessly. This feature has been tested across different user roles and seems to be working well.

    - DONE- **Checkout Process**: Order processing, including address capture, has been implemented. The payment setup is still a placeholder, and I'm considering options based on user feedback.

    - BUSY - **Order Tracking**: Basic order tracking is working, but I'm enhancing it for clearer status updates, so users know exactly where their order stands.

4. **Responsive Design**

   - DONE- **Mobile and Desktop Compatibility**: Core components are responsive, and the UI adapts well to mobile and desktop devices. I'm still testing for tablet compatibility and refining some of the layouts.

5. **Security Features**

   - DONE- **Data Validation**: Input validation is active across most fields, especially for registration and product entries.

System Documentation

1. Technical Architecture

- **System Overview**: The web app runs on a LAMP stack with Linux as the host, Apache as the server, MySQL for database management, and PHP for backend scripting. It has a well-defined structure that supports CRUD operations on user data, product listings, and orders.

- **Component Interaction**:

  - **PHP and MySQL**: PHP handles CRUD operations and securely interacts with the MySQL database for data management.

  - **Frontend (HTML, CSS, JavaScript)**: Built for usability, it uses a mix of HTML, CSS, and JavaScript to make the interface intuitive and responsive.

  - **Security Components**: Sessions and role-based access control are managed with PHP, ensuring that only authorized users access certain functionalities.

2. Database Schema

- **Entities and Relationships**: The following tables are set up and currently functional:

  - **Users Table**: Stores basic user information and role assignments.

  - **Products Table**: Holds product details, like name, price, and stock availability.

  - **Orders Table**: Manages links between users and products, including order details and statuses.

3. User Interface (UI) Design

- **Dashboard for Farmers**: Farmers have a dedicated dashboard where they can manage product listings, view orders, and edit profiles.
- **Product Listings for Consumers**: Consumers can browse through listings, add items to the cart, and proceed to checkout.
- **Mobile-Friendly Layout**: Screens have been designed for easy access on mobile and desktop devices, with ongoing testing for full tablet compatibility.

4. Instructions for Running the Application

- **Development Setup**:
    1. Install a LAMP stack (Linux, Apache, MySQL, PHP) on your local machine or server.
    2. Clone the repository and configure Apache settings.
    3. Import the SQL file to initialise the database in MySQL.
    4. Update database connection settings within PHP files.

Deployment: Configuration for a production environment is underway. This includes final security settings and permissions to ensure a stable live setup.

Problem Statement

Problem Identification: Many small-scale farmers lack a streamlined, affordable way to connect directly with local consumers and businesses, resulting in high levels of food waste and reduced profitability. Local consumers, meanwhile, face limited access to fresh produce at competitive prices.

Solution: This application addresses this gap by providing a direct-to-consumer marketplace, where farmers can sell their products without intermediaries. This approach not only reduces food waste but also increases the financial sustainability of small-scale farmers.

Context: This platform has the potential to strengthen local economies by supporting sustainable agriculture and reducing food waste. It also fosters community health by giving consumers easy access to fresh, locally-sourced food.