WasteLess

Analysis and Design Document

Student:Rus Tudor

**Group:30431**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

WasteLess is an app that helps users manage food waste.

# Functional Requirements

Once a user is authenticated he can input grocery lists and see reports of how much food is wasted weekly and monthly. A grocery list item has a name and a quantity as well as a calorie value, purchase date, expiration date and consumption date.

The system also allows users to track goals and minimize waste by sending reminders if waste levels are too high based on ideal burndown rates.

The system provides you with options to donate excess food to various local food charities and soup kitchens and notifies you of them prior to item expiration.

# Non-functional Requirements

The non-functional requirements are:

* The user should not have access to another user’s data
* The user should not be allowed to have access to any data without logging in

2. Use-Case Model

Use case: User login

Level: user-goal level

Primary actor: user

Main success scenario: the user introduces the correct username and password and is redirected to the main app page

Extensions: user introduces a wrong username or password => error message is displayed



3. System Architectural Design

**3.1 Architectural Pattern Description**

The application is split into two parts: frontend and backend, and it represents a client-server architecture.

The backend implements the Layered Architecture Pattern.

**Layered Architecture Pattern**

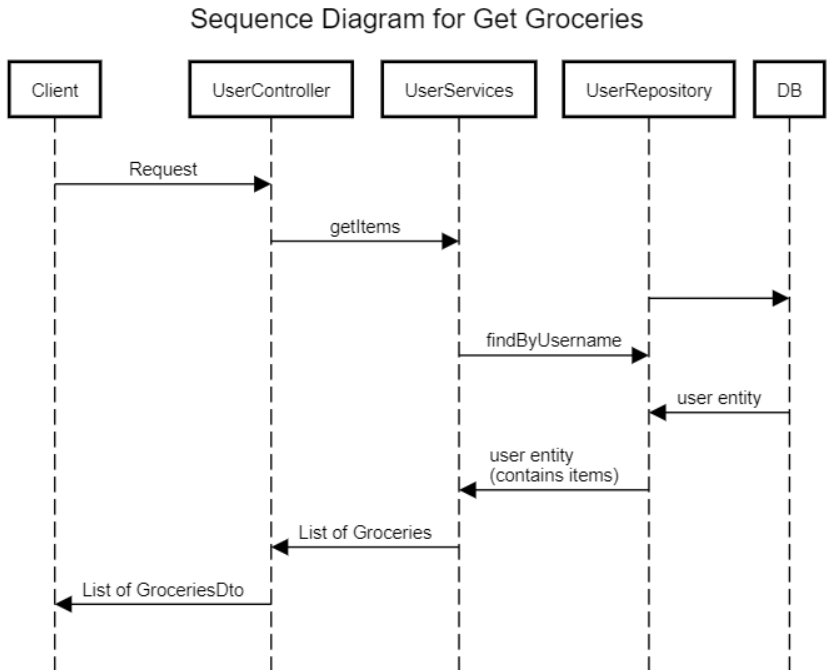
Components within the layered architecture pattern are organized into horizontal layers, each layer performing a specific role within the application (e.g., presentation logic or business logic). Although the layered architecture pattern does not specify the number and types of layers that must exist in the pattern, most layered architectures consist of four standard layers: presentation, business, persistence, and database In some cases, the business layer and persistence layer are combined into a single business layer, particularly when the persistence logic (e.g., SQL or HSQL) is embedded within the business layer components. Thus, smaller applications may have only three layers, whereas larger and more complex business applications may contain five or more layers.

**3.2 Diagrams**

The main architecture used is the layered architecture.



4. UML Sequence Diagrams



6. Data Model

Item:

Id,name,quantity,calories,purchaseDate,expirationDate,consumptionDate,userFK

Login:

Id,username,password,userFk,role

User:

Id, name, goal, email, groceryList, login

7. System Testing

Simple unit tests

8. Bibliography