<Wasteless app assignment 2>

Analysis and Design Document

Student: Bozdog Ioana

**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 4

4. UML Sequence Diagrams 5

5. Class Design 5

6. Data Model 7

7. System Testing 7

8. Bibliography 7

1. Requirements Analysis

# Assignment Specification

Design and implement an application that helps users manage food waste.

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 ideal burndown rate for 100 calories worth of groceries due to expire in 5 days is

20 calories worth of groceries per day.

The system should provide you with options to donate excess food to various local

food charities and soup kitchens and notify you of them prior to item expiration.

For this Assignment I chose to use Angular as the client side and the previous assignment as the server side.

# Functional Requirements

The user can perform CRUD operations such as create a new account, create a new list, set a new goal, edit an item, see the lists and their items and can also delete lists, as well as items. Besides that, the user can choose to donate food to various charities.

# Non-functional Requirements

Use an ORM and a DI Container

Use any OOP language you like

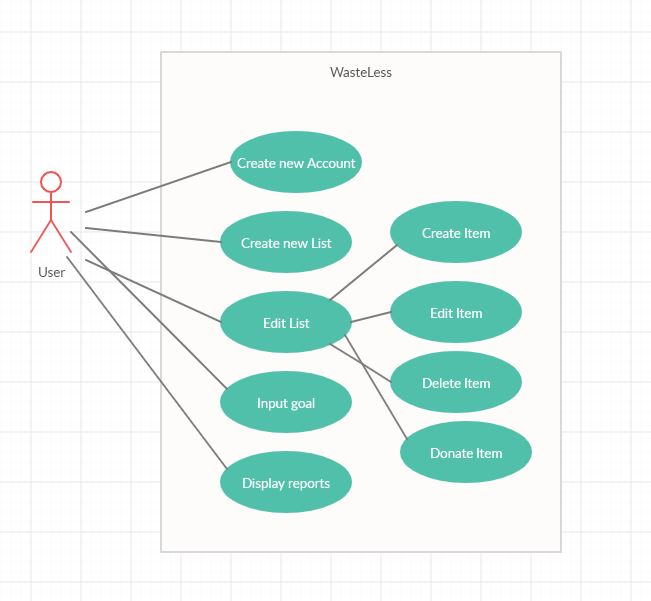
Use a client-server architecture

Use an observer for sending notifications to users about donation options when item

expiration is due

The data will be stored in a database

2. Use-Case Model



Use case: create new account

Level: user goal

Primary actor: User

Main success scenario: The User that wishes to create a new account needs to enter the login page and click on the “Create new user”. A new page will appear and the user will need to input the desired username and password, after which they will need to save the information.

Extensions: a fail case for this would be that the user could input an already used username, in which case they will be prompted to change the password

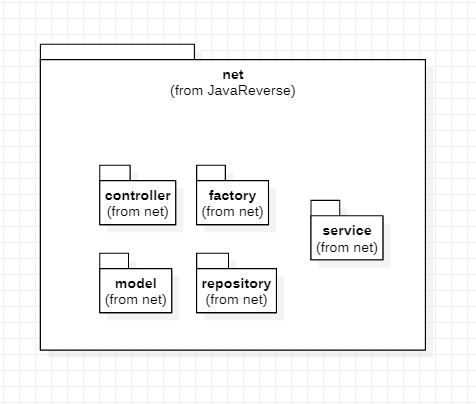
3. System Architectural Design

**3.1 Architectural Pattern Description**

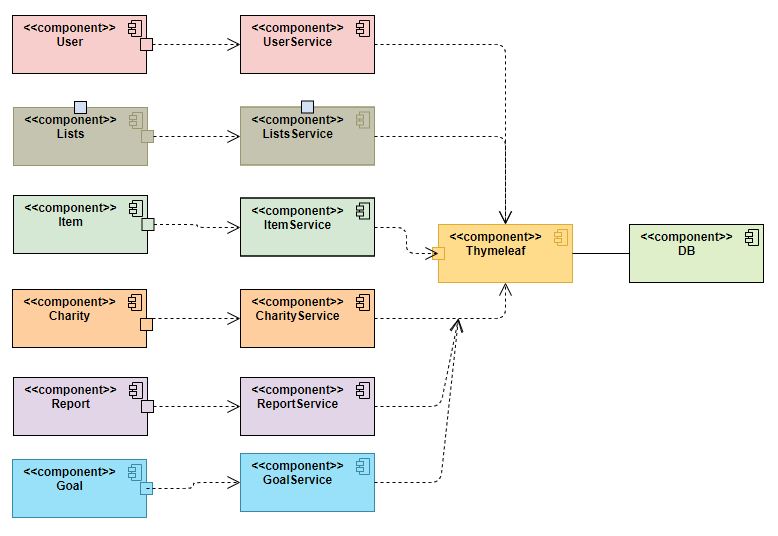
As you can see below, the Architecture used is a client-server one, having a model package where every entity class is stored, a controller package where every request is handled, a service package that makes use of the interfaces from the repository package and the factory package that creates two different types of reports, as well as a presentation layer that can be found in the templates package from resources that handles the visual elements of the application. The application layer is not together with the rest of them because of some restriction from Spring.

**3.2 Diagrams**

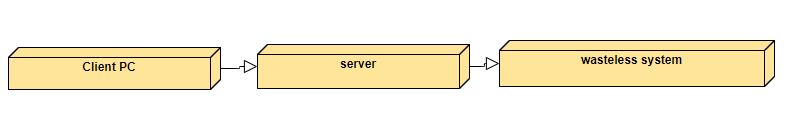
Package:



Component:

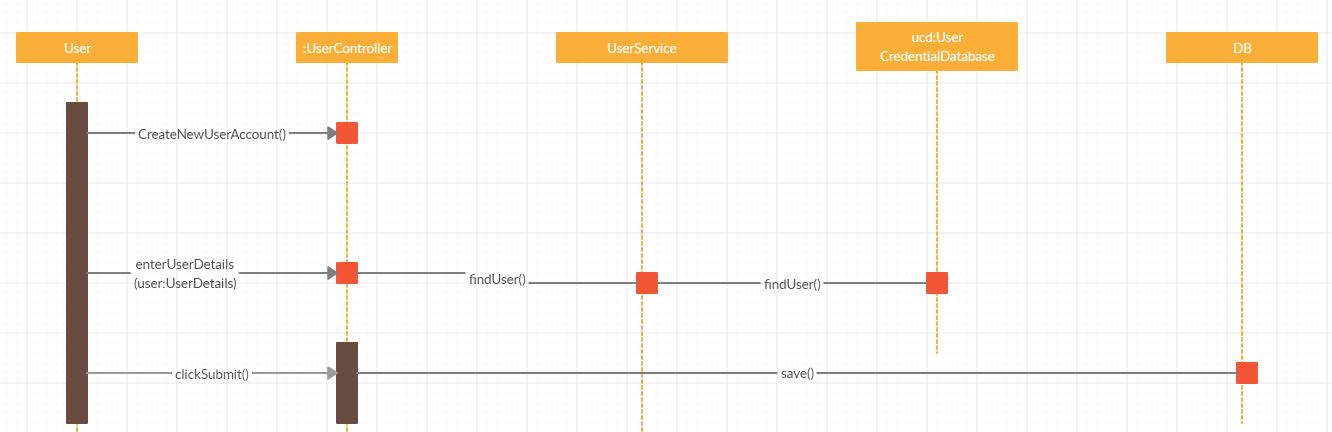


Deployment:



4. UML Sequence Diagrams

Creating a new Account:



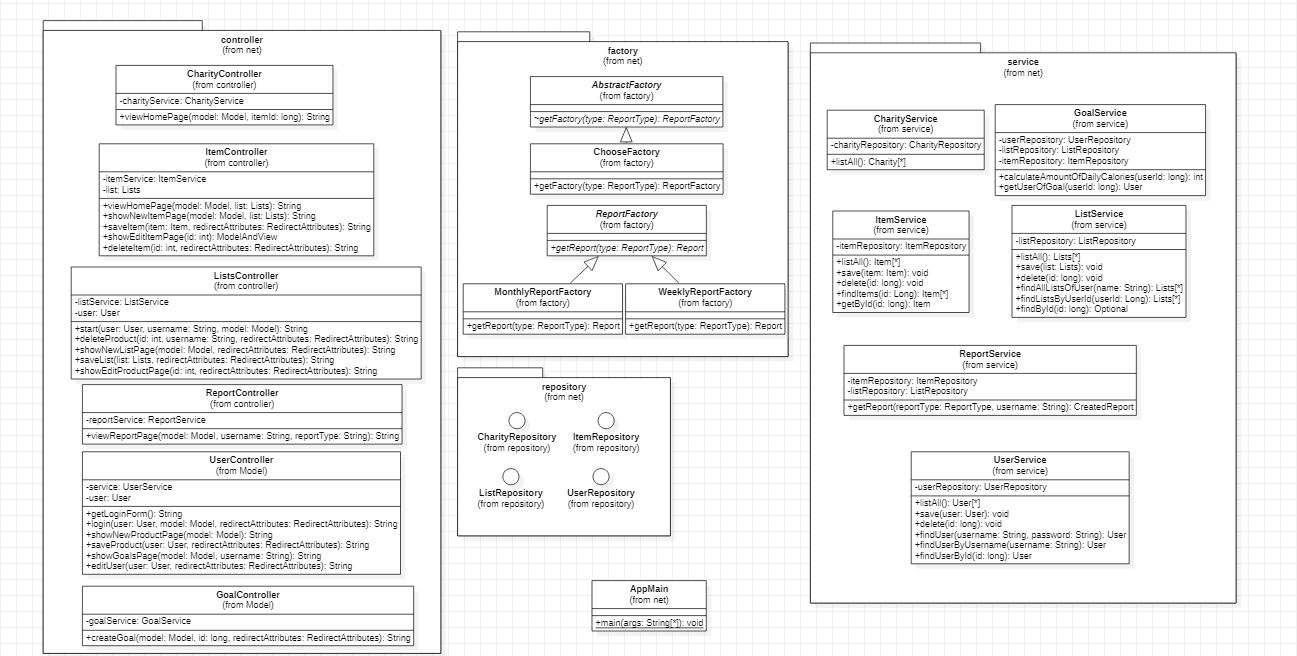
5. Class Design

**5.1 Design Patterns Description**

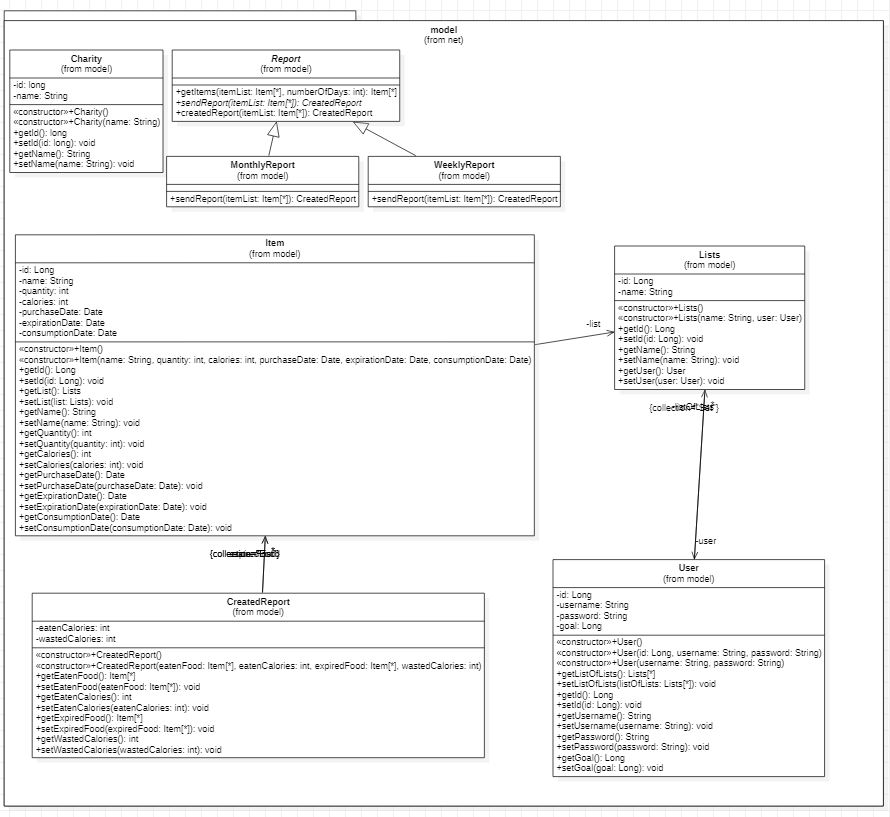
The design pattern used in this assignment is the Abstract Factory Pattern which basically is a factory of factories. This was used to create two different kinds of reports, a weekly report and a monthly report of wasted food, as well as eaten food. Since the requirements only specify two types of reports, there are two Factories, one that generates weekly reports, and another that generates monthly reports. Both these factories make use of two other classes, ReportFactory and ChooseFactory that basically decide which type of report to be implemented. There is also an enum that only contains the two types of reports that can be implemented, weekly or monthly.

I also needed to implement an observer for expired items. The hard part here was knowing when the day changed, but I used timer schedule and it was ok. In Angular I displayed a list of those expired items that refreshes every day.

**5.2 UML Class Diagram**



6. Data Model



7. System Testing

The test that I created was a simple sanity check test that would fail if the application context could not start. The test succeeds.

8. Bibliography

-stackoverflow

-<https://www.thymeleaf.org/doc/tutorials/2.1/thymeleafspring.html>

-<https://spring.io/guides/gs/testing-web/>

-baeldung

-<https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-test/2.1.1.RELEASE>

-YouTube indians