<Assignment Name>

Analysis and Design Document

Student: Neagu Lorena

**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 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.

# Functional Requirements

* Authenticate user and login
* Input and view grocery lists
* Input and view grocery list items
* Set goals
* Receive reminder for high waste levels
* Receive notification before items expire
* Generate reports

# Non-functional Requirements

* Use a layered architecture
* Use the abstract factory pattern for creating weekly/monthly reports
* The data will be stored in a database
* All the inputs of the application will be validated against invalid data before submitting

the data and saving it in the database.

* Use an ORM and a DI Container

2. Use-Case Model

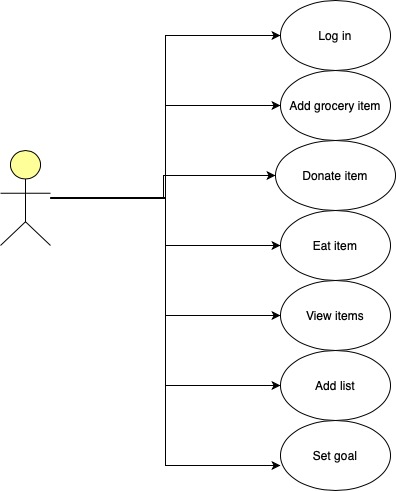
*Use case: Add Grocery List Item*

*Level: User-goal level*

*Primary actor: User*

*Main success scenario: First the authenticated user can input a new grocery item in an existing grocery list or in a newly created one. The data introduced is checked against validity constraints. He must not enter null fields, the purchase date must be only past or present and the dates must follow the date format specified in the text field. For the dates he can select the date from a drop-down calendar. If the user introduced the right item data, the item will be added to the database in the respective list and it can be seen in the list seen below.*

*Extensions: In case the data introduced is invalid or null, the page may display an error and the data won’t be added to the database. The user can the introduce a valid item.*



3. System Architectural Design

**3.1 Architectural Pattern Description**

The application follows a layered architecture, more precisely a 3-tier architecture. The levels of the 3-tier architecture are data, business and presentation layer, which are subleveled into a Spring specific component layering: Repository, Web, Service.

The presentation layer contains the Web component of the app. It interacts with the end user. It shows data, takes user input or takes a command from it. The Business Layer contains the services and is based on the User command and the data captured from the user. It takes a domain-specific decision, like what to do with the data and how to manipulate the data which comes from the database, so it can be presented into the user interface. Data layer contains the repositories and captures the data and persists it, captures any change of the state of the data, so this layer is maintaining a state of the application data.

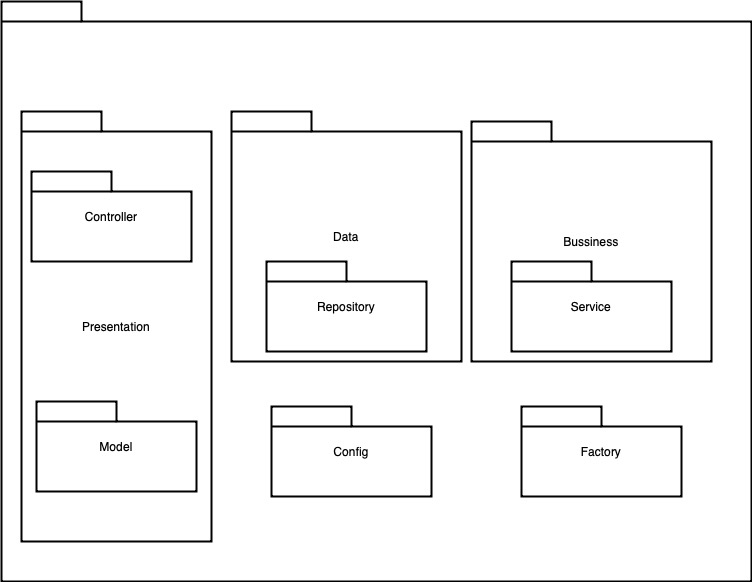
From the Spring point of view, the Web layer, which contains the Controllers and the Model classes, is the uppermost level and represents the entry point of the app. It is responsible of processing user’s input and returning the correct response back to the user.

The service layer resides below the web layer. It acts as a transaction boundary and contains both application and infrastructure services. The application services provide the public API of the service layer. They also act as a transaction boundary and are responsible of authorization. The infrastructure services contain the “plumbing code” that communicates with external resources such as file systems, databases, or email servers. Often these methods are used by more than a one application service.

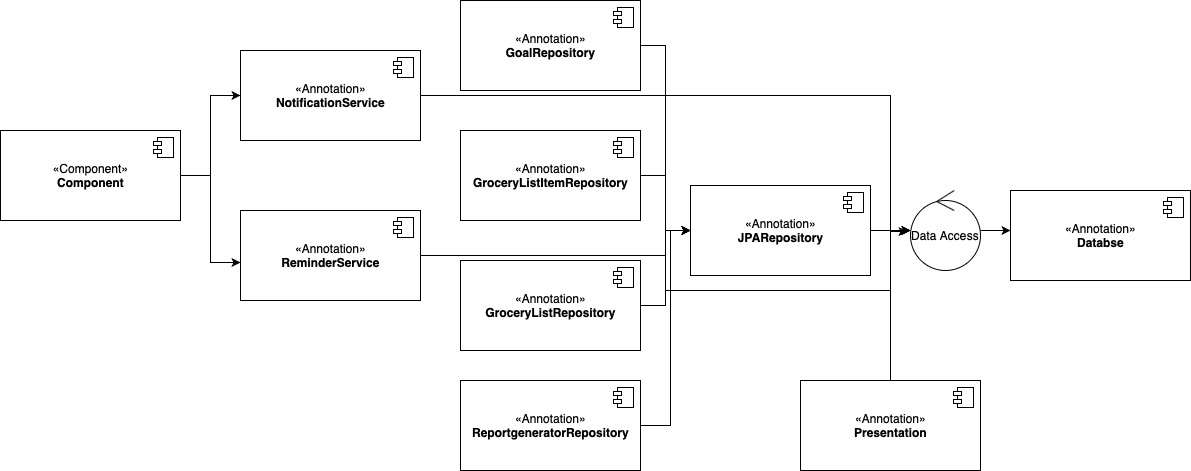
The repository layer is the lowest layer of a web application. It is responsible of communicating with the database.

**3.2 Diagrams**

*Package Diagram*



*Component Diagram*

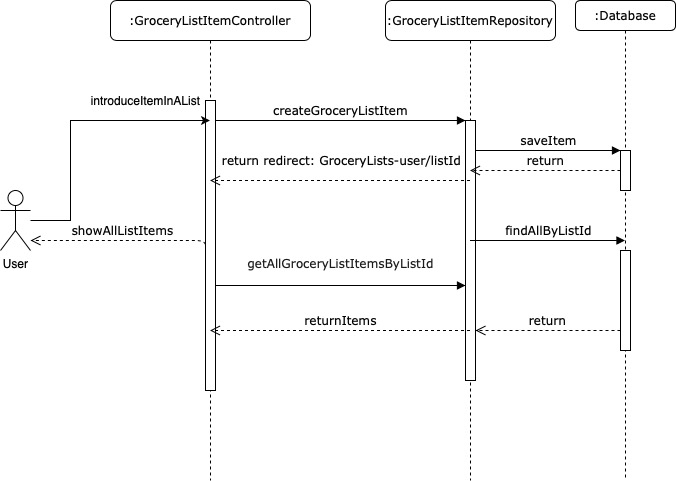


*Deployment Diagram*



4. UML Sequence Diagrams

Scenario: insert a new Grocery List Item



5. Class Design

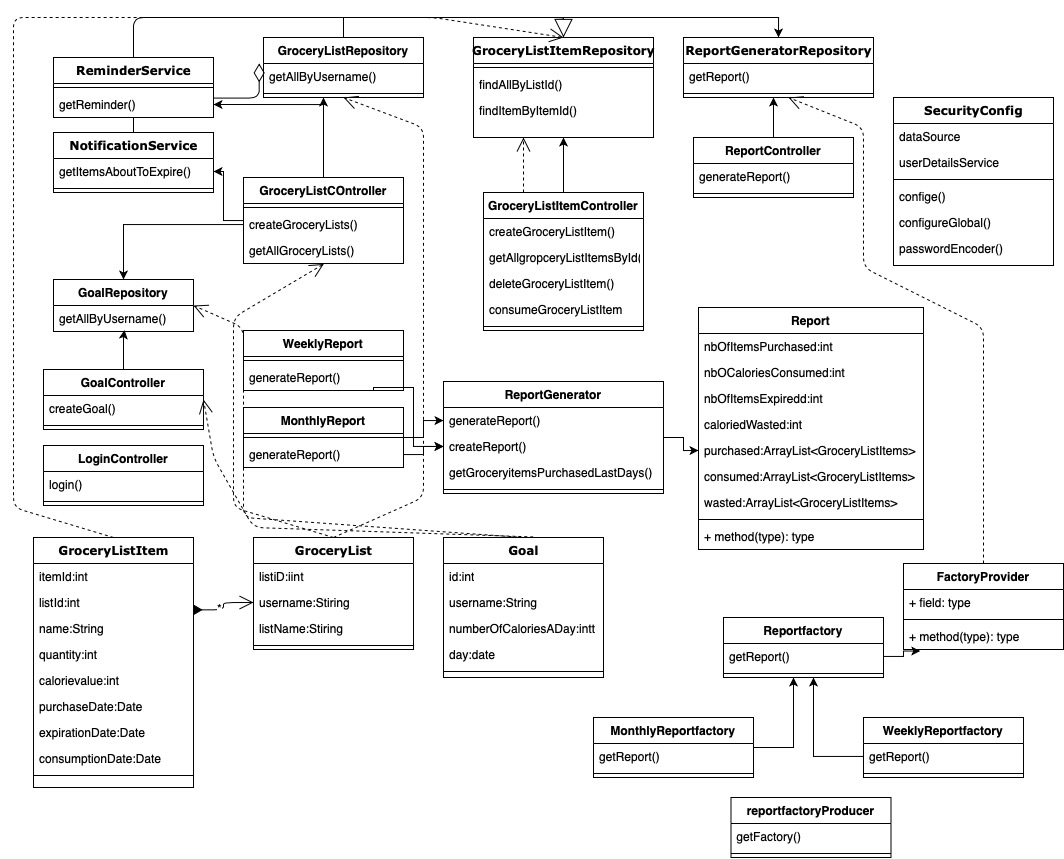
**5.1 Design Patterns Description**

Wasteless web application uses an Abstract Factory design pattern. It is one of the Creational pattern. Abstract Factory pattern works around a super-factory which creates other factories.

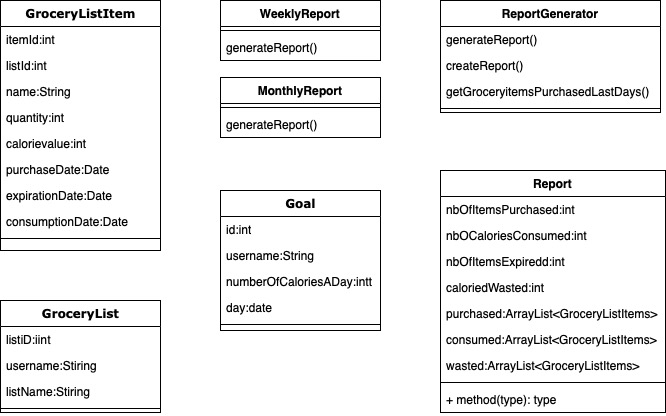
Abstract factory pattern implementation provides a framework that allows us to create objects that follow a general pattern. So, at runtime, abstract factory is coupled with a desired concrete factory which can create objects of desired type.

In this case the pattern is used for Monthly/Weekly type reports generation. Having two enums MONTHLY, WEEKLY, we will use a Report Factory as an abstract class that is inherited by the two factories WeeklyReportFactory and MonthlyReportFactory, which both need to implement the abstract method “getReport”. The factory producer will give the type of report chosen by the user and will be used in the report generator repository which will communicate with the database in order to extract the needed data and generate the report desired by the user, based on all the existing items that belong to the user logged in, which are bought last week or last month.

**5.2 UML Class Diagram**



6. Data Model



7. System Testing

The application uses Junit Framework for unit testing. Junit 5 is used for creating test cases to test the saving of the entities GroceryList, GroceryListItem and Goal in the H2 database and check the results are correct. The testing system contains 3 test cases for each main component of the system. Each case checks for the successful creation of a list, an item and a goal, which represent the three main objects used in the application, as well as checking if they were saved correctly in the database. The user entity doesn’t need to be checked because it is created by default by the Spring Security Framework, that uses the default schema.

8. Bibliography

<https://javabydeveloper.com/spring-boot-h2-in-memory-database-example/>

<https://www.javacodegeeks.com/2018/10/layered-architecture-running-just-5-minutes-spring-boot-part-1.html>

<https://www.tutorialspoint.com/design_pattern/abstract_factory_pattern.htm>

<https://www.javaguides.net/2018/06/guide-to-dependency-injection-in-spring.html>

<https://www.javaguides.net/2018/06/guide-to-dependency-injection-in-spring.html>