Wasteless A1

Student:Sas Cosmin Andrei

**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

The user will first register, then he can add lists of groceries. The application will then display the list on the account page, along with the number of edible calories, the calories that will expire soon and the expired calories. The user can request a monthly or weekly report regarding the number of calories.

# Functional Requirements

*As I said, the user is able to add items. He can throw, donate or consume them. He can request reports of the items*

# Non-functional Requirements

* *Implement and test the application*
* *Use an ORM and a DI Container*
* *Commit the work you do on your Git repository. Do it iteratively as you progress, not all at once (this will incur a penalty on your final mark)*
* *Use any OOP language you like. Non-exhaustive: Python, C#, Java, Ruby, C/C++, JS+Typescript*
* *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.*

2. Use-Case Model

*Use case: the users can see the amount of food they have and manage it Primary actor: everyone that has food Main success scenario: register, add a list, throw expired items and donate the ones that soon expire, consume items, get reports Extensions: the failures lead to informative pop-ups*

3. System Architectural Design

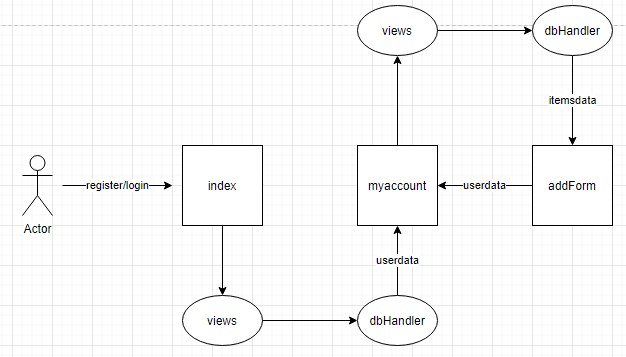
**3.1 Architectural Pattern Description**

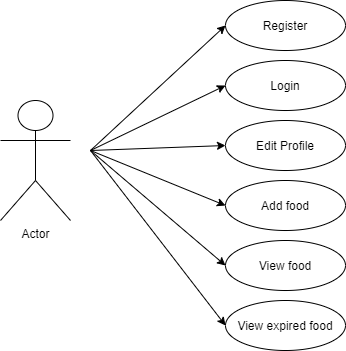
*I used the layered architecture to access the database.*

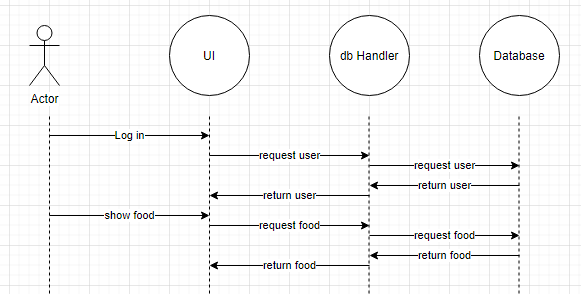
**3.2 Diagrams**

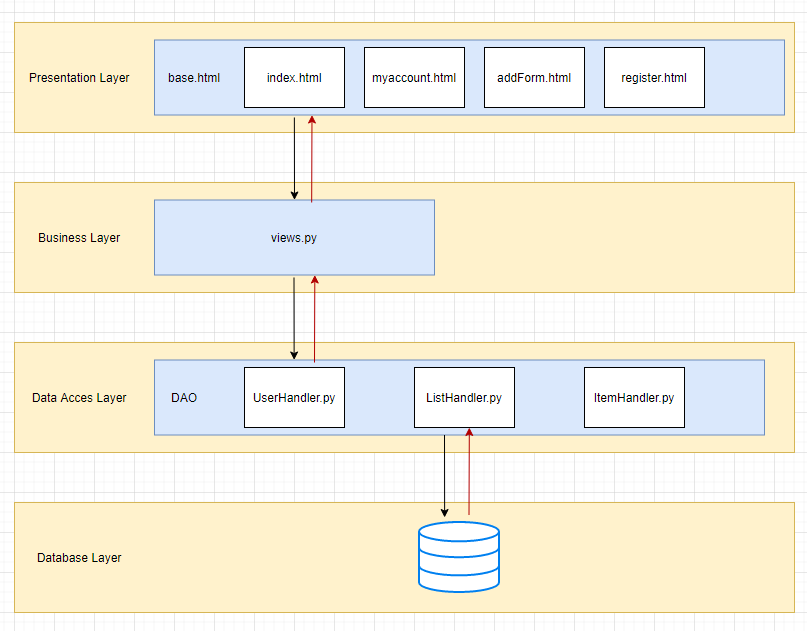
*[Create the system’s conceptual architecture; use architectural patterns and describe how they are applied. Create package, component and deployment diagrams]*

4. UML Sequence Diagrams

**



**



5. Class Design

**5.1 Design Patterns Description**

*I used a layered design pattern. The app will use a mediator to access the database.*

**5.2 UML Class Diagram**

6. Data Model

*Profile: name(char), password(char), calories(int)*

*List: user, name(char), date(datetime)*

*FoodItem: list, name(char), expiration date(datetime), calories(int)*

7. System Testing

*Each component has been tested individually and after assuring functionality they have been put together. After that, data-flow testing was done to ensure that all components function as expected.*

8. Bibliography