Recipe Wizard

Software Requirements Specification

<Date>

Senanur Güvercinoğlu 150119740

Melisa Durmuş 150119727

Cem Anaral 150119761

Ali Yetim 150119803

Nurefşan Yücel 150119802

Prepared for

CSE3044 Software Engineering Term Project

**Table of Contents**

**1. Introduction 3**

1.1 Purpose 3

1.2 Scope 3

1.3 Definitions, Acronyms, and Abbreviations 3

1.4 References 3

1.5 Overview 3

**2. General Description 4**

2.1 Product Perspective 4

2.2 Product Functions 4

2.3 User Characteristics 4

2.4 General Constraints 4

2.5 Assumptions and Dependencies 4

**3. Specific Requirements 4**

3.1 External Interface Requirements 5

*3.1.1 User Interfaces 5*

*3.1.2 Hardware Interfaces 5*

*3.1.3 Software Interfaces 5*

*3.1.4 Communications Interfaces 5*

3.2 Functional Requirements 5

*3.2.1 <Functional Requirement or Feature #1> 5*

*3.2.2 <Functional Requirement or Feature #2> 5*

3.3 Non-Functional Requirements 5

*3.3.1 Performance 6*

*3.3.2 Reliability 6*

*3.3.3 Availability 6*

*3.3.4 Security 6*

*3.3.5 Maintainability 6*

*3.3.6 Portability 6*

3.4 Inverse Requirements 6

3.5 Design Constraints 6

3.6 Logical Database Requirements 6

3.7 Other Requirements 6

**4. UML Diagrams 6**

4.1 Use Cases 6

*4.1.1 Use Case #1 6*

*4.1.2 Use Case #2 6*

4.2 Classes / Objects 6

*4.2.1 <Class / Object #1> 6*

*4.2.2 <Class / Object #2> 6*

4.3 Sequence Diagrams 7

4.4 Data Flow Diagrams (DFD) 7

4.5 State-Transition Diagrams (STD) 7

**A. Appendices 7**

A.1 Appendix 1 7

A.2 Appendix 2 7

# 1. Introduction

**1.1. Purpose**

The purpose of this document is to explain our project, recipe recommendation for leftover products and also you can share your recipes. This document describes the system's software requirements, system intent, and features. This document is prepared for both stakeholders and system developers. The purpose of developing this application is to provide convenience and savings for customers.

By filtering the ingredients in your home, it prevents users from wasting fruits and vegetables and offers users the opportunity to try new recipes. It also offers many options that make it easy to cook and share recipes.

**1.2. Scope of Project**

This system will be a website. This system aims to finding and presenting recipes that the user is searching and providing correct recommendations similar to what the user prefers.Also user share his/her recipes on the web site. The system contains a database which is providing lists of users, recipes, category of recipes, recipes’ ingredients. This database enables to keep the information of recipes and users.

In addition, this system also receives personal information such as age, height, weight from the user. Users can update this information and keep this information under control. Also some of ingredients’ calories will be found in the system. Users can choose recipe according to the amount of calories they need to take daily.

In addition, this system will save user’s time to find preferred recipes.

**1.3 Definitions, Acronyms and Abbreviations**

UI: User Interface

DBMS: Database Management System

React: A Javascript framework.

SQLite: A lightweight and relational DBMS engine.

Django Rest Framework: A framework built upon Python that enables writing services.

Python: An object-oriented programming language.

ARM: A computer architecture mainly used in embedded systems.

x86-64: A computer architecture mainly used in embedded systems.

HTML: Stands for HyperText Markup Language

Javascript: A programming language used on the client-side.

Admin:Someone who is in charge of the website. Which viewed the information of the users and validated the information of the users.

User: A person that can view or add recipes to the system.

**1.4 References**

IEEE Template for System Requirement Specification Documents:

https://goo.gl/nsUFwy

## 1.5 Overview

After that, we'll go through our requirements. We'll also display our case diagrams. We shall give a brief introduction to SRS in the first part. Then we'll go over our general descriptions, followed by a thorough discussion of the needs. Finally, we will present our Case Diagrams.

# 2. General Description

Recipe wizard is a platform where you can find the recipes you want and/or add the recipes you want. It also will give you an idea of what you can do with the materials you have. You can find the popular recipes. It allows you to try new tastes.

## 2.1 Product Perspective

Unsimilar to other recipe sites, the required ingredients and utensils are not hard coded as texts somewhere in the recipe. Our system uses labels for that purpose. Using labels enables our users to search for a recipe based on the ingredients / utensils they already have.

## 2.2 Product Functions

Briefly, our product's functions are creating and reading recipes.

## 2.3 User Characteristics

It is not necessary for the user to have a certain level of education or technical knowledge. It is sufficient to use the website with basic internet knowledge.

## 2.4 General Constraints

Time: Time is a main constraint for our project. We have to finish before the deadline.

Scope: This app will suggest recipes and calories of meals.

Limit of database: The size of the database where we keep users' information has a limit.

Software Platform: “Recipe Wizard” is a Website, so we can do what this platform allows.

## 2.5 Assumptions and Dependencies

• Internet: The system expects that you are connected to the internet.

• Operating System: The program is compatible with any operating system, including Windows, Linux, and Android, as well as any web browser.

• People: It is anticipated that everyone who wishes to cook would do so.

• Skill: The program verifies that the user is capable of using a computer, tablet, or smartphone.

# 3. Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### Front-end software: React

### Back-end software: Django Rest, Sqlite

### 3.1.2 Hardware Interfaces

### A browser which supports HTML & Javascript.

### 3.1.3 Software Interfaces

Operating System: We have chosen Windows operating system for its best support and user-friendliness.

Database: Sqlite

Django: To implement the project we have chosen Django language for its more interactive support

### 3.1.4 Communications Interfaces

This project is compatible with all web browsers. We use basic forms for sign-in, adding, and filtering recipes, among other things.

## 3.2 Functional Requirements

*This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.*

### 3.2.1 User Log In

3.2.1.1 Description

Users should be able to enter the system by providing a username and password.

3.2.1.2 Inputs / Display

There will be two textboxes for username and password and a button for submitting these fields.

3.2.1.3 Processing

The user data will be sent to the backend service, where they will be checked whether they are valid or not. A query will be executed for that in the DBMS.

3.2.1.4 Outputs

Users will be redirected to the home page based on the validness of credentials.

3.2.1.5 Constraints

Password textbox must be hidden.

3.2.1.6 Error/Data Handling

### If credentials are not valid, users will not be granted access to the home page and stay in the log in page.

### 3.2.2 Searching

3.2.2.1 Description

Users will be able to look for recipes by their names, ingredients or utensils. Users be able to apply these filters simultaneously.

3.2.2.2 Inputs / Display

There will be 3 search bars. One will be used for filtering based on the recipe name, others will be used for filtering the ingredients and utensils separately. Ingredient and utensil names will be shown along with their checkboxes.

3.2.2.3 Processing

A query on DBMS will be issued based on the filters sent from the user. Results will be sent back to the UI.

3.2.2.4 Outputs

Recipes will be shown according to the result of the database query.

3.2.2.5 Constraints

The user must enter a recipe name or at least one ingredient name. Both of them can not be empty.

3.2.2.6 Error/Data Handling

User must enter a name that we have in our system. Otherwise, results will be empty.

### 3.2.3 Adding Recipe

3.2.3.1 Description

Users can add new recipes. The recipe should include necessary ingredients and utensils.

3.2.3.2 Inputs / Display

A big textbox for typing the recipe will be shown. Also the filtering component from previous requirements will be shown here in order to label the recipe. Another textbox will be for adding calorie information.

3.2.3.3 Processing

All of the recipe data taken from the user will be inserted into the DBMS.

3.2.3.4 Outputs

User will be asked if they want to add a new recipe or return to the home page by the help of the choice box.

3.2.3.5 Constraints

User must check the ingredients and the utensils they used in the recipe from the ingredient and utensil list.

3.2.3.6 Error/Data Handling

If user presses the add key without entering any data we display an error message and the empty data will not be sent to DBMS.

### 3.2.4 Like / Dislike Recipe

3.2.4.1 Description

Users will be able to like / dislike recipes.

3.2.4.2 Inputs / Display

There will be two buttons for liking / disliking. Also the number of likes / dislikes will be shown on the recipe page.

3.2.4.3 Processing

When a user hits the like / dislike button a query will be executed on DBMS which increments the like / dislike field.

3.2.4.4 Outputs

Current like or dislike number will be incremented by one and this number will be shown to the user.

3.2.4.5 Constraints

Users can not like and dislike at the same time.

3.2.4.6 Error/Data Handling

Users can not like or dislike a recipe more than one time.

### 3.2.5 Ranking

3.2.5.1 Description

Recipes will be ranked according to their like / dislike numbers in a page.

3.2.5.2 Inputs / Display

A list of recipes will be shown.

3.2.5.3 Processing

Recipes on the DBMS will be sorted based on number of likes subtracted by number of dislikes.

3.2.5.4 Outputs

Recipes will be ranked according to their like / dislike numbers in a page.

**3.3 Non-Functional Requirements**

**3.3.1 Availability**

On the website, users should be able to quickly locate what they're searching for. Web design should be uncomplicated. In general, people should be able to access a website.

**3.3.2 Maintainability**

All functions, particularly complicated ones, should be checked.

**3.3.3 Performance**

The reaction time of the system to the user, as well as other minor time measures such as refresh time, are critical.

**3.3.4 Security**

Only the system's data administrator has the ability to modify access permissions for system data. Passwords must never be seen during the login process or at any other time.

**3.3.5 Portability**

The website may be accessed from any device that has an internet connection and is equipped with a web browser.

**3.3.6 Reliability**

Users should be able to access the recipes 98% of the time without failure.

## 3.4 Inverse Requirements

* Calories of recipes will not be calculated. It will be taken from the author as an input.

## 3.5 Design Constraints

* Our backend service will be guaranteed to run on x86-64 architecture. ARM support is not guaranteed.

## 3.6 Logical Database Requirements

* A relational DBMS called SQLite will be used. For holding the recipe instructions TEXT data type will be used. For like / dislike INTEGER will be used.

# 4. UML Diagrams

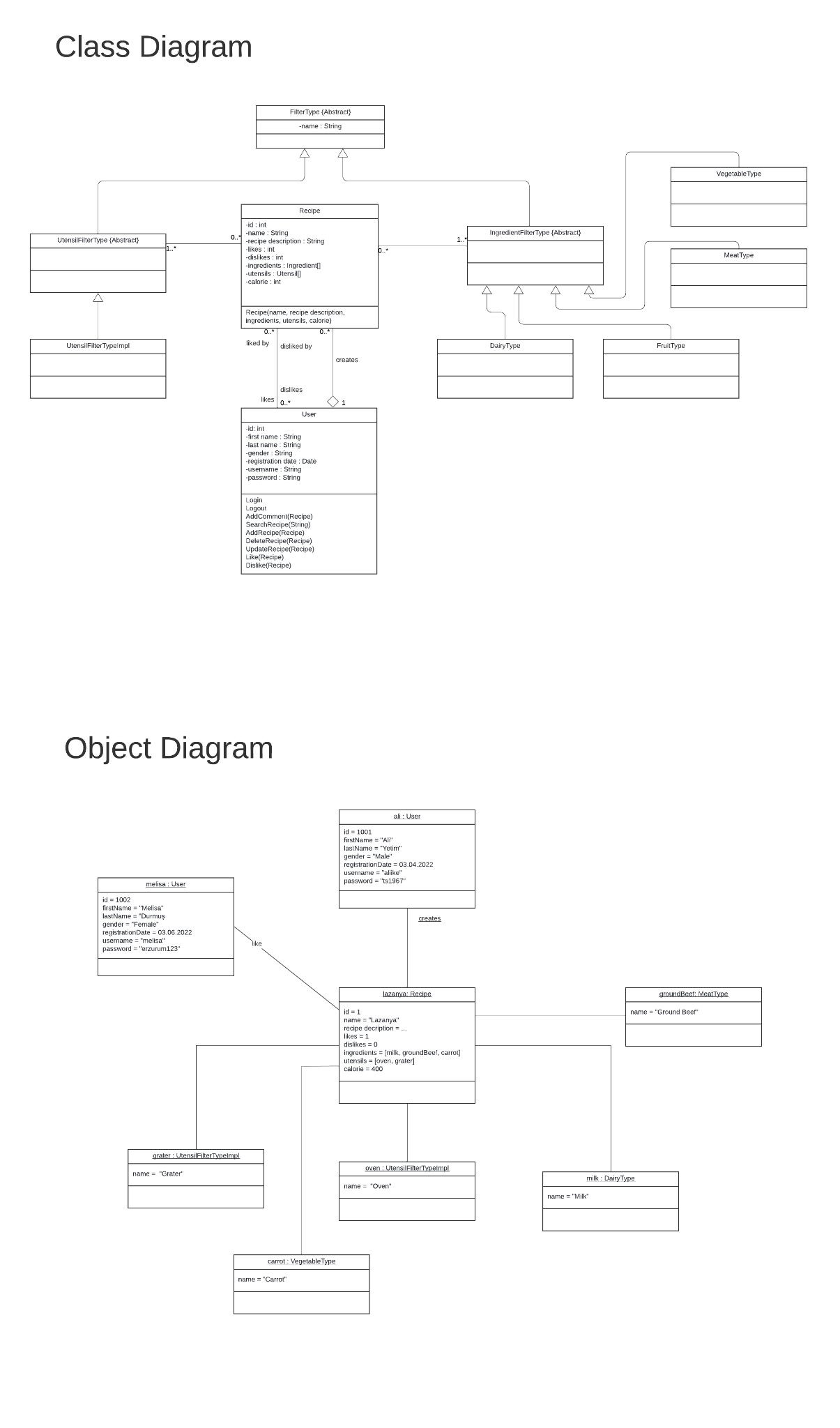
## 4.1 Use Cases

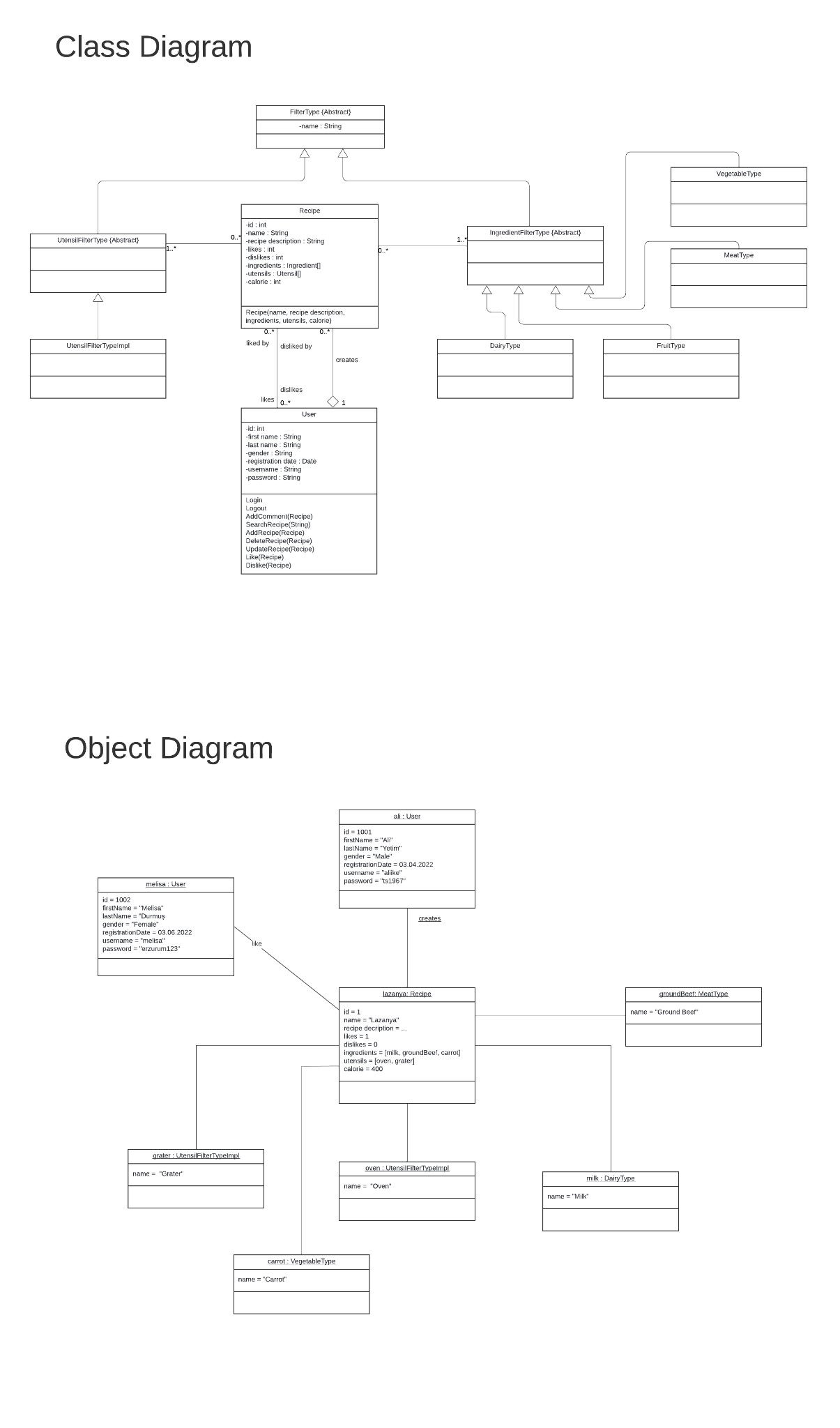
### 4.1.1 Use Case #1

### 4.1.2 Use Case #2

…

## 4.2 Classes / Objects





### 4.2.1 References to functional requirements and/or use cases in Class Diagram

* Class FilterType is used for filtering and labeling recipes. (see: 3.2.2)
* Class User is used in logging (see: 3.2.1), adding recipe (see: 3.2.3)

### 4.2.2 References to functional requirements and/or use cases in Object Diagram

* ali:User creates lazanya:Recipe (see: 3.2.3)
* melisa:User likes lazanya:Recipe (see: 3.2.4, see: 3.2.5)

## 4.3 Sequence Diagrams

## 4.4 Data Flow Diagrams (DFD)

## 4.5 State-Transition Diagrams (STD)

# A. Appendices

## A.1 Glossary

Acknowledgment: Sections of this document are based upon the IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984). The SRS templates of Dr. Orest Pilskalns (WSU, Vancover) and Jack Hagemeister (WSU, Pullman) have also be used as guides in developing this template.