Version overview

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Comment |
| 21.10.2024 | 2 | Gozman-Pop Maria-Eliza  Borzan Călina-Annemary | ……. |

System requirements for < Project Title>

[1 Introduction 2](#_Toc22892244)

[2 General view of actors and use cases 2](#_Toc22892245)

[3 Detailed description of actors and use cases 2](#_Toc22892246)

[3.1 Actors 2](#_Toc22892247)

[3.1.1 Administrator 2](#_Toc22892248)

[3.1.2 Customer 3](#_Toc22892249)

[Type: ……….. 3](#_Toc22892250)

[3.1.3 Provider 3](#_Toc22892251)

[3.2 Use cases 3](#_Toc22892252)

[*3.2.1* Use Case Diagram for *Actor1* 3](#_Toc22892253)

[3.2.2 Use Case Diagram for *Actor2* 3](#_Toc22892254)

[3.2.3 Use Case Diagram for *Actor3* 4](#_Toc22892255)

[*3.2.4* Use Case Diagram for *Actor*n 4](#_Toc22892256)

[4 Glossary – used terms 5](#_Toc22892257)

# Introduction

The project presented in this document is *YumCycle*, a fridge contents management application. This mobile app is designed around helping people of all ages and places in managing their fridge leftovers. The idea has started in our own dorm room, where every week, we buy a lot of unnecessary groceries and always end up throwing some of them away. This is how this project was born, we realized we could optimize the way we handle our aliments and also find new recipes to cook and eat healthier. The main focus of this app is to help the user keep track of what is in their fridge and generate recipes based on this information, so that we will minimize waste.

In more detail, we will implement a bar-code scanning mechanism based on pictures provided by the user, and we will also have the option to add them manually. Then, our application will provide helpful instructions on how to cook delicious meals, with no need to buy other products. The user can add their own recipes, right from their grandma’s cookbook, or they can benefit from the wide database that we will connect to: <https://www.barcodelookup.com/4056489662860>.

# General view of actors and use cases

|  |  |  |
| --- | --- | --- |
| ***Actor*** | ***Type*** | ***Description*** |
| *Administrator* | *Human* | *The person administrating the system, the ones that aprove recipes that can be posted, and the ones that choose who is banned from using the app.* |
| *User* | *Human* | *Performs general operations over the system, such that updating the virtual fridge contents, adding recipes, searching for food in the connected database.* |

# Detailed description of actors and use cases

## Actors

### Administrator

**Type**: human

**Detailed description**:

This actor is the person administrating the system. The administrator is the one responsible for the maintenance of the application. They are the ones managing the user’s activity in case of misbehaviour, the ones who choose who gets the opportunity to add their recipes to the common database, and enhancing the user experience based on public reviews.

**Where is used:**

This actor is used in the: “*Administrator page*” use case

. “*Administrator Log-in* “use case.

“*Administrator Log-out*” use case.

“*Order page*” use case

“*Deleting an article in the Database*”use case

“*Editing Article Info in the Database*” *u*se case

“*Entering a New Article into the Database*” use case

### User

### Type: human

**Detailed description:**

…………………………………………..

**Where is used:**

This actor is used in the: …………………………..

## Use cases

*This sub-chapter should describe how each actor would interact with the system. If the actor is a human, then describe typical ways he will use the system. System requirements captured in this way are called use cases.*

### **Use Case Diagram for *Actor1***

**………………….. / *put here the use case diagrams for Actor1/***

**Description:**

**Actor:** ……………..  
**Precondition:** -  
**Postcondition:** …………………..

**Behavior:** ………………………………

### **Use Case Diagram for *Actor2***

**………………….. / *put here the use case diagrams for Actor2/***

**Description:**

**Actor:** ……………..  
**Precondition:** -  
**Postcondition:** …………………..

**Behavior:** ………………………………

### **Use Case Diagram for *Actor3***

**………………….. / *put here the use case diagrams for Actor1/***

**Description:**

**Actor:** ……………..  
**Precondition:** -  
**Postcondition:** …………………..

**Behavior:** ………………………………

………………………………

### **Use Case Diagram for *Actor*n**

**………………….. / *put here the use case diagrams for Actor*n*/***

**Description:**

**Actor:** ……………..  
**Precondition:** -  
**Postcondition:** …………………..

**Behavior:** ………………………………

# Glossary – used terms

*This chapter should contain all the key items or terms or entities the application deals with.*

*Here is an example were the glossary terms are defined:*

|  |  |  |
| --- | --- | --- |
| **Term** | **Use cases** | **Description** |
| Administrator account  authentication | Administrator Log-in use case | Contains all the information needed by the administrator that wants to authenticate itself in the system using a user password schema |
| Provider account authentication | Provider Log-in use case | Contains all the information needed by a Provider that wants to authenticate itself in the system using a ticket schema |
| Maintenance file | Administrator page case | Contains all the information used by the system. |
| Order file | Order page | Latest orders. |
| Shopping Card | Adding an Article to the Shopping Cart,  Ordering, Delivery and payment | It is used for buying and ordering a product |
| …………….. | …………… | …………………………………………………………. |
| ………………. | ………… | …………………………………………………………. |

*Terms with related meanings and/or that are used in similar contexts can be grouped by term groups as depicted in the following table.*

|  |  |  |
| --- | --- | --- |
| **Term Groups** | **Included terms and term groups** | **Description** |
| Authentication | Administrator password authentication, Provider password authentication | Defines a general group of credentials used in the authentication process |