

**REPUBLIC OF CAMEROON**

**Peace-Work-Fatherland**

**MINISTER OF HIGHER EDUCATION**

**FACULTY OF ENGINEERING**

**AND TECHNOLOGY**

**REPUBLIQUE DU CAMEROON**

**PAIX-Travail-Patrie**

**MINISTRE DE L’ENSEIGNEMENT SUPERIEUR**

**FACULTE D’INGINERIE**

**ET TECHGNOLOGIE**

**CEF440: INTERNET AND MOBILE PROGRAMMING**

**Project3: FOOD WASTE MANAGEMENT SYSTEM**

**COLLECTION AND ANALYSIS OF THE FINAL REQUIREMENTS**

**Presented by Group1:**

**BITO’O BI HIEGA II DYLAN FLORENT FE20A024**

**CHEZEM DONGMEZA MIGUEL FE20A026**

**DJIOTSA DJOUAKE CHRISTIAN DARYN FE20A029**

**DJOUKENG NOUGNING YANNICK IVAN FE20A030**

**MADADJO KUITCHE MONICK CYBELLE FE20A060**

**INSTRUCTOR**: **Dr. NKEMENI VALERY**

**April 2023**

Table of Contents

[**1.** **INTRODUCTION:** 4](#_Toc132372938)

[a) **Purpose:** 4](#_Toc132372939)

[b) **Scope** 4](#_Toc132372941)

[2. **OVERALL DESCRIPTION:** 4](#_Toc132372943)

[2.1. **Product perspective:** 4](#_Toc132372944)

[2.1.1. **System Interface** 4](#_Toc132372946)

[2.1.2. **User interface** 4](#_Toc132372948)

[2.1.3. **Hardware Interface** 4](#_Toc132372950)

[a. **Server side** 4](#_Toc132372951)

[b. **Client side** 4](#_Toc132372953)

[2.1.4. **Software interface** 5](#_Toc132372960)

[a- **Server side** 5](#_Toc132372961)

[b- **Client side** 5](#_Toc132372963)

[2.1.5. **Communication interfaces** 5](#_Toc132372965)

[2.2. **Products functions:** 5](#_Toc132372967)

[2.2.1. **Context Diagram** 5](#_Toc132372968)

[2.2.2. **Use case diagrams** 6](#_Toc132372970)

[2.2.2.1. **User log in** 6](#_Toc132372971)

[2.2.2.2. **Discussion Threads** 6](#_Toc132372973)

[2.3. **User characteristics** 8](#_Toc132372974)

[2.3.1. **System administrator** 8](#_Toc132372975)

[2.3.2. **Normal users** 8](#_Toc132372977)

[3. **SPECIFIC REQUIREMENTS:** 9](#_Toc132372979)

[3.1. **External interface** **Error! Bookmark not defined.**](#_Toc132372980)

[3.2. **Functional requirements** 9](#_Toc132372981)

[3.2.1. **Use case scenarios** 9](#_Toc132372982)

[3.2.1.1. **Use case scenario 1: User log in.** 9](#_Toc132372983)

[3.2.1.2. **Scenario 2- Posting food** 9](#_Toc132372984)

[3.2.1.3. **Scenario 3- Discussion, Comments and Ratings.** 10](#_Toc132372985)

[3.2.1.4. **Scenario 4- Search result.** 10](#_Toc132372986)

[3.2.1.5. **Scenario 5- Notification** 11](#_Toc132372988)

[3.3. **Performance requirement** 11](#_Toc132372989)

[3.4.  **requirements** 11](#_Toc132372990)

[3.5. **Design constraint** 11](#_Toc132372992)

[3.6. **Non Functional Requirements** 12](#_Toc132372996)

[3.6.1. **Reliability** 12](#_Toc132372997)

[3.6.2. **Availability** 12](#_Toc132372999)

[3.6.3. **Security** 12](#_Toc132373001)

[3.6.4. **Maintainability** 12](#_Toc132373003)

[3.6.5. **Portability** 12](#_Toc132373005)

1. **INTRODUCTION:**
2. **Purpose:**

The purpose of this document is to capture, in natural language and at a functional level, the description and requirements of a food waste management system. This is a functional description of the different features and requirements that will be necessary to build the mobile application.

1. **Scope**

This application will be a **web mobile application** that will be used by any user in order to post a food he wants to get rid of for the other user to get to him and take an appointment to receive the food. The system shall provide features for the user to create an account, post his food, view food post, send messages, comment, like or unlike food post, send his localization.

1. **OVERALL DESCRIPTION:**
   1. **Product perspective:**

Food waste management system (FWMS) is meant to serve as a platform where individuals in a community can upload the food stuffs they are no longer in need of for other users to see and demand the food.

* + 1. **System Interface**

Django server will be used our web server. The user inputs data via the web server using HTML forms (React JS). The actual program that will perform the operations is written in Python (Django).

* + 1. **User interface**

The system shall provide a very intuitive and simple interface to the user and the administrator, so that the user can easily navigate through pages, create post, delete post, chat with other users and use GPS to join a meal point

* + 1. **Hardware Interface**

1. **Server side**

The web application will be hosted on a web server which is listening on the web standard port, port 80.

1. **Client side**

**Computers:**

**Monitor screen** – the software shall display information to the user via the monitor screen

**Mouse** – the software shall interact with the movement of the mouse and the mouse buttons. The mouse shall activate areas for data input, command buttons and select options from menus. **Keyboard** – the software shall interact with the keystrokes of the keyboard. The keyboard will input data into the active area of the database.

* **Phones and tablets:**

**Mobile screen:** – the software shall display information to the user via the screen

**Touch pad:** – the software shall interact with the touch pad of the phone**.**

* + 1. **Software interface**

1. **Server side**

An Apache web server will accept all requests from the client and forward it accordingly. A database will be hosted centrally using Django.

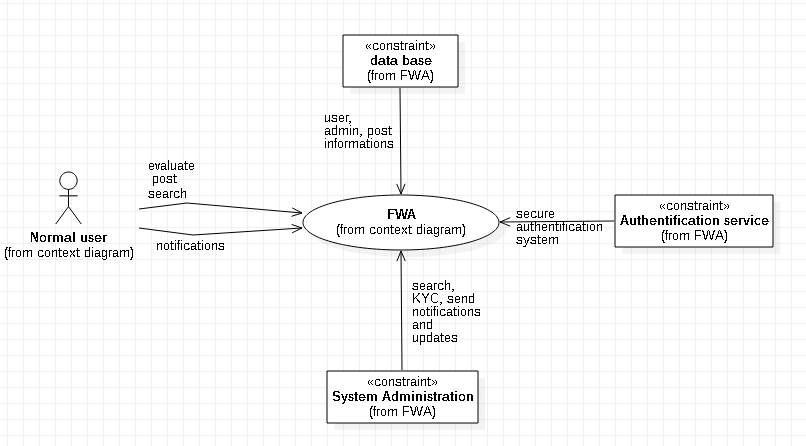
1. **Client side**

The application is will run on all the mobile Operating Systems, given that it is a web application.

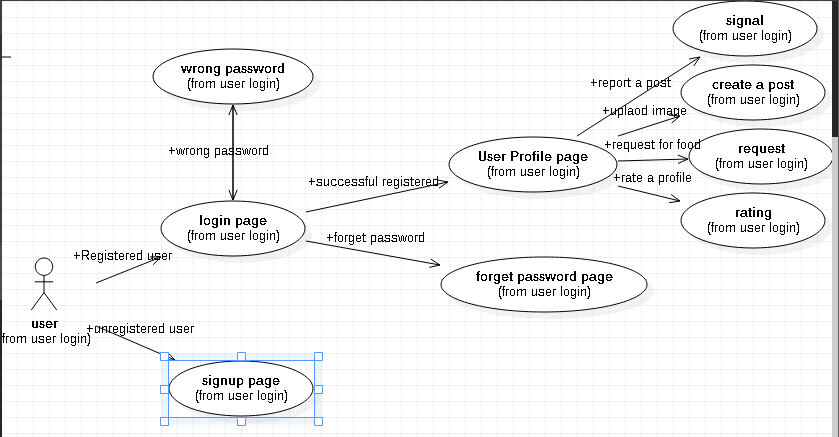
* + 1. **Communication interfaces**

The HTPP or HTTPS protocol(s) will be used to facilitate communication between the client and server.

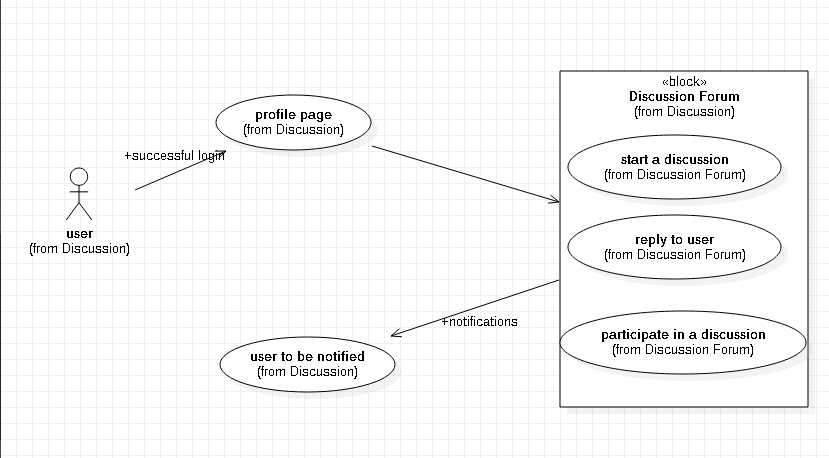
* 1. **Products functions:**
     1. **Context Diagram**

****

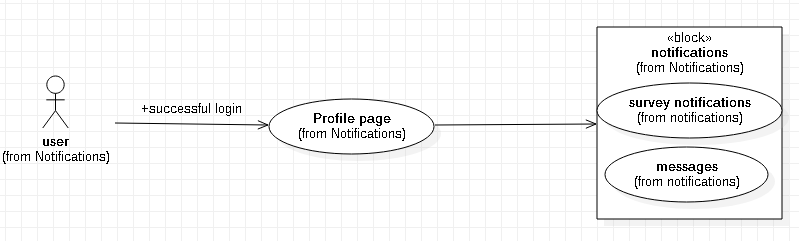
* + 1. **Use case diagrams**
       1. **User log in**



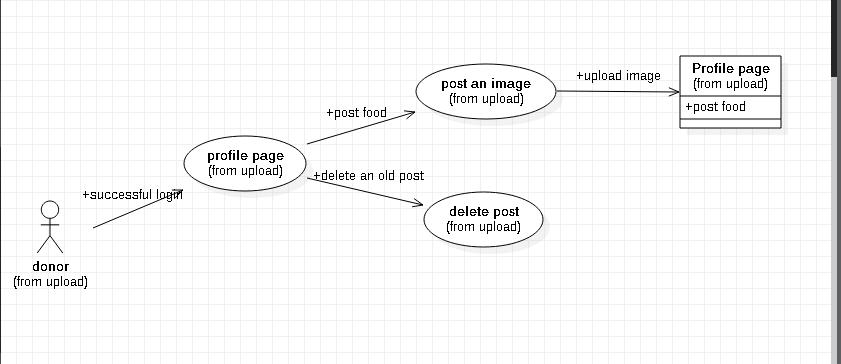
* + - 1. **Discussion Threads**



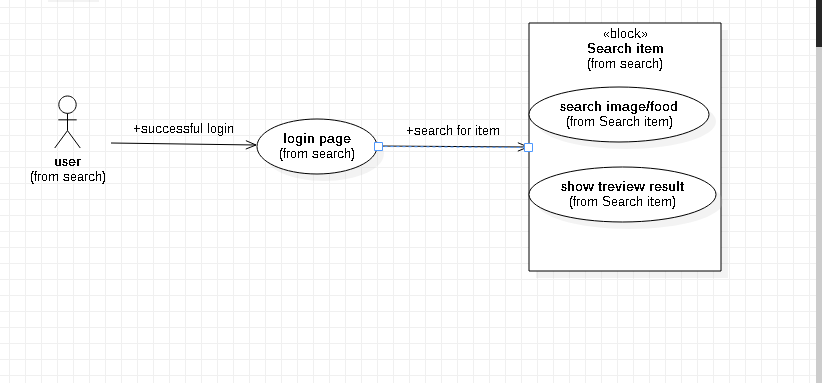
* + - 1. **Notifications**

****

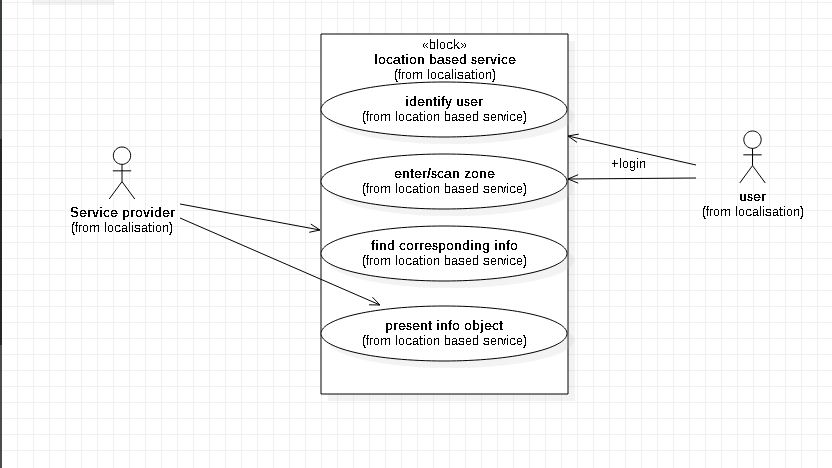
* + - 1. **Make a post**

****

* + - 1. **Search**



* + - 1. **Location**



* 1. **User characteristics**
     1. **System administrator**

The admin has the possibility to ban a user, to provide support for users, to verify the identity of the users (KYC: Know Your Customers). He can be able to delete a user’s post

* + 1. **Normal users**

They are the main users of the system. They can create an account, log into their account make/delete a post, chat, and they can provide ratings to other users.

1. **SPECIFIC REQUIREMENTS:**
   1. **Functional requirements**
      1. **Use case scenarios**
         1. **Scenario 1: User log in.**

|  |  |
| --- | --- |
| **Purpose** | User logs in to system using existing profile. |
| **User** | A user with an existing profile. |
| **Input Data** | Profile username and password. |
| **Output Data** | Corresponding page data. |
| **Invariants** | Profile table data and user information. |
| **Pre-Conditions** | User is not logged in to a profile, input  profile exists in data base, user  password matches profile |
| **Post-conditions** | User's computer has been supplied  with appropriate cookie, page data is  appropriate for selected profile |
| **Basic Flow:** | Webpage looks up profile data and returns the matching cookie. Webpage is updated to match new user data. |
| **Alternative Flow(s):** | Invalid password, invalid username,  Or mismatched username and password redirect to error message and previous page. |
| **Business Rules:** | This allows users to log in to their profile from anywhere. |

* + - 1. **Scenario 2- Posting food**

**A user Logs into the system and creates a post to publish available meal**

|  |  |
| --- | --- |
| **Purpose** | User wants to let others know he has available food |
| **User** | A legitimate(KYC checked) user that is logged into the system |
| **Input Data** | Picture Of the food, Nature ( cooked or uncooked ), Location, Quantity |
| **Output Data** | Post shared to other users |
| **Invariants** | The post cannot be modified |
| **Pre-Conditions** | User is Logged in and verified through KYC |
| **Post-conditions** | Any person in the same locality should be notified of the post and be able to register for the donation |
| **Basic Flow:** | A user creates a post by taping on the post button, taking a picture of the food, giving  his location and all other users in his town can view his post and apply to have the meal |

* + - 1. **Scenario 3- Discussion, Comments and Ratings.**

|  |  |
| --- | --- |
| **Purpose** | User Discuss with other users of the system, takes rendezvous with food donors and sellers, comment posts and also rate their quality of Service |
| **User** | A user that is logged into the system |
| **Input Data** | Message to be send |
| **Output Data** | Message send to targeted user or set of users |
| **Invariants** | The message sent cannot be modified but can be deleted |
| **Pre-Conditions** | User is Logged into the system |
| **Post-conditions** | The person(s) to who the message is destined to should be the only ones to be able to view its content |
| **Basic Flow:** | A user sends a message to another user on his profile page or sends a message in the main forum for all users to see or even comments a post or rate a user on the quality of service he received from a previous donation, this messages are stored on the server and send specifically to the target |

* + - 1. **Scenario 4- Search result.**

A user wants to find a particular food in the system

|  |  |
| --- | --- |
| **Purpose** | User wants to search for a particular meal |
| **User** | Any logged in user |
| **Input Data** | Name of meal, Filter by Category or Location |
| **Output Data** | All the corresponding posts |
| **Invariants** | The user and the search result |
| **Pre-Conditions** | User is Logged in |
| **Post-conditions** | Search result should be precise, explicit and usable |
| **Basic Flow:** | User logs in, Enters the name of the meal in the search box, clicks the search button and gets the search results. He can also filter the results by Category and by Location |

* + - 1. **Scenario 5- Notification**

|  |  |
| --- | --- |
| **Purpose** | To get Notifications of posts of foods you like the most, and permanent notifications for post monitoring |
| **User** | Any User of the system |
| **Input Data** | User enable notifications from his device settings |
| **Output Data** | The Notification |
| **Invariants** | Notification Content |
| **Pre-Conditions** | User has an account in the system and have granted notification permission to the app |
| **Post-conditions** | User receives only desired notifications |
| **Basic Flow:** | User enables notifications, and receives notification anytime food is published in his town  Notifications are also send to posters for them to monitor their posts without necessarily opening the app to indicate that the food they posted is finished for example |

* 1. **Logical database requirements**

All data will be saved in the database: user accounts and profiles, discussion data, messages etc. (except files which are stored on the disk.) The database allows concurrent access and will be kept consistent at all times, requiring a good database design.

* 1. **Performance requirements**

The system should support a large number of users. It is important that a substantial number of users be able to access the system at the same time

* 1. **Design constraint**

1. The communication between the portal software and the database will be in **Postgresql**.

2. The portal layout will be produced with **React JS**.

3. The product will be written in **Django**.

* 1. **Non Functional Requirements**
     1. **Reliability**

The reliability of the overall program depends on the reliability of the separate components.

* + 1. **Availability**

The system should be available at all times,

* + 1. **Security**

Passwords will be saved encrypted in the database in order to ensure the user's privacy.

* + 1. **Maintainability**

**Postgresql** is used for maintaining the database and the Apache server takes care of the site. In case of a failure, a re-initialization of the program is recommended.

* + 1. **Portability**

We are building a web application so it will be compatible with all the different Operating Systems.

* + 1. **Scalability**