**Used Oil Management**

**Functional Specification**

Version 1.0

Last modified 15/09/2023 13:25:00 AM

Number of pages 7

|  |  |  |
| --- | --- | --- |
| Approver list | | |
| Date | Our program possible users | Approved by |
|  | Ministry of Environment |  |
|  | FSIB - food safety inspection body of Armenia |  |
|  | The Government of Armenia |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Document History | | |
| Date | Description | Author(s) |
| 2023-09-15 | created | Ani |
| 2023-09-18 | finished | Ani |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[1 Introduction 4](#__RefHeading__3731_216754541)

[1.1 Document identifier](#__RefHeading__3733_216754541) 4

[1.2 Scope](#__RefHeading__3735_216754541) 4

[1.3 Definitions of Terms and Acronyms](#__RefHeading__3737_216754541) 4

[1.4 References](#__RefHeading__3739_216754541) 4

[2 Functional requirements](#__RefHeading___Toc7698_808326353) 5

[2.1 Non-Functional requirements](#__RefHeading___Toc2943_193171694) 7

[3 Dependencies, Risks](#__RefHeading___Toc3717_961417545) 7

[3.1 Dependencies](#__RefHeading___Toc3946_961417545) 7

[3.2 Risks](#__RefHeading__1996_465711201) 7

* **Introduction**
* ***Document identifier***

This document describes the functional specifications for the Used Oil Management project.

* ***Scope***
* Analysing accumulated oil resources gathering frequency, giving optimal routes for logistic company.
* Decreasing polluting of enviornment.
* ***Definitions of Terms and Acronyms***

**Term/Acronym Definition**

Admin Head of the oil evaluation system

Restaurants All kinds of public food services

* ***References***

Document referenced:

phpMyAdmin – database

https://www.phpmyadmin.co/index.php

* **Functional Components Specified**

**1․ int main(); -** the main function that starts the program․

**2․ Authentication**

* **void authPage(); -** presents the first page of the program, offers 4 options to start use it:

login(); - if you already have an account

registration(): - if you want to create an account

forgot(); - if you forgot your password

exit - if you don't want use the programm

* **void login();** - provides access to the user's page
* **void registration(); -** provides data reception, registration in the database, page creation for the user
* **void forgot(); -** generates a random code and sends it to the email added by the user, if he forgotten password.
* **void showProfilePage(); -** presents the user's profile page, offers options - for Menu Screen

for Ordering

for Analytics

EXIT

* **bool lettersAndNumbers(std::string& password); -** checks for the condition that the password must be composed of both letters and numbers.

Gets password as argument.

* **std::string generateUniqueID(); -** ganerates random numbers as an ID for each user

**3. Menu**

* **menu(); -** default constructor
* **void firstPage();**

presents the profile page of current user, offers 6 options to add and change menu.

* **fetchUser -** fegtchs user by ID from databese
* **void addMenu(std::string Current\_user\_ID);**

as argument gets Current\_user\_ID from database

gets dishes name, oil count and frying level from user, add them in Menu table with Current\_user\_ID.

* **void editMenu(std::string Current\_user\_ID);**

as argument gets Current\_user\_ID from database

edit current user's menu.

* **void deleteMenu(std::string Current\_user\_ID);**

as argument gets Current\_user\_ID from database

delete current user's menu

* **void ishowMenu(std::string Current\_user\_ID);**

as argument gets Current\_user\_ID from database

show current user's menu.

**4. Order**

* **Order(const std::string &dishName, const std::chrono::system\_clock::time\_point &date, const int &oilCount, const int &count, const std::string &RestNameOrderID); -** default constructor
* **void fillOrderData(std::string currentUserID); -** provides data reception, fetchs orders by ID from Menu table.
* **int takeOrder(Order order);** - adds orders in Order table
* **int getDishByID(std::string currentUserID, std::string dishName); -** gets currentUserID and dishName as an arguments, finds the menu row corresponding to them

**5. Calculations**

* **float calculateCoefficientOnOrder(bool post, std::string currentUserID); -** gets current user ID and post as arguments – Function calculates coefficient on order.
* **float calculateFrequencyOnOrder(bool post, std::string restID, std::string currentUserID); -** gets current user ID and post as arguments – Function calculates frequency on order.

**6․Analytics**

* **void firstPage(); -** offers options

Show Analytics

Go Back to Profile

EXIT

* **void showAnalytics(); -** gets frequency and coefficient results from according functions and show them.

**7. MySQL**

**library - mysql.h**

* **mysql\_real\_connect** - attempts to establish a connection to a MySQL server
* **mysql\_query** - provids selecting information by given table.
* **mysql\_store\_result** - provides a synchronous connection to the specified database.
* **mysql\_num\_fields -**  returns the number of columns by given result/MySQl/.
* **mysql\_fetch\_row** - f etches one row of data from the result set and returns it as an array of char pointers
* **INSERT INTO .... -** inserts the data into the table according to the given arguments
* **mysql\_free\_result** - frees the memory associated with a result set.
* **mysql\_close** - close database/MySQL/
* ***Non-Functional specifications***

*DataBase MySQL*

* **Platform Requirements**

Online database MySQL , bash terminal

* **Dependencies, Risks**
* Restaurants may provide inaccurate information of oil they use․ The goal is to present less than the amount consumed and reuse the oil․
* Human factor is uncontrollable