



Offerize

Graduation Project, Part-I (SWE 496)

Software Engineering Department

CCIS, KSU

Project Advisor:

Dr.Achraf Gazdar

Submitted by

Salman Abdulaziz, 439101819

Nawaf Abdulaziz Alahmed, 437105083

Abdullah Aldayel, 439101185

Saleh Abdullah Alshehri, 436108011

Date submitted

05/23/2022

ABSTRACT

At the beginning we all want the best price for each item without going throw each supermarket company it will be waste of time, and effort. That the purpose of the project. the customer can browse all the supermarkets items, filters the items by their category and price, also searching for specific item. and the main goal is finding the cheapest supermarket company.

Furthermore, to develop this platform, it was decided to make the system to be a cross-platform mobile application for the customer. it will be available on Android and IOS devices and, there will be a website for the (supermarket company, data provider, admin) for entering the items and offers and controlling them.

Table of Contents

[1. Introduction 8](#_Toc104153920)

[2. Shortcut Table 9](#_Toc104153921)

[3. Domain Analysis 10](#_Toc104153922)

[3.1 Competitors 10](#_Toc104153923)

[**3.1.1 Carrefour** 10](#_Toc104153924)

[**3.1.2 Danube** 11](#_Toc104153925)

[**3.1.3 Flipp** 12](#_Toc104153926)

[**3.1.4 Basket** 13](#_Toc104153927)

[3.2- Competitors' comparison 14](#_Toc104153928)

[4. Risk/Constraints 15](#_Toc104153929)

[5. Project Plan 16](#_Toc104153930)

[6. Quality Assurance Plan 17](#_Toc104153931)

[7. Requirements 18](#_Toc104153932)

[7.1 Functional Requirements 18](#_Toc104153933)

[**7.1.1 Customer’s Functional Requirements** 18](#_Toc104153934)

[**7.1.2 Data Provider Functional Requirements** 18](#_Toc104153935)

[**7.1.3 Administrator Functional Requirements** 19](#_Toc104153936)

[**7.1.4 SMC Functional Requirements** 19](#_Toc104153937)

[7.2 Non-Functional Requirements 19](#_Toc104153938)

[8. Problem Complexity 20](#_Toc104153939)

[9. System Use-Cases 21](#_Toc104153940)

[9.1 Customer use case diagram 21](#_Toc104153941)

[9.1.2 Customer use cases description 22](#_Toc104153942)

[9.2 System background users use case diagram 28](#_Toc104153943)

[9.2.1 SMC use cases description 29](#_Toc104153944)

[9.2.2 Data provider use cases description 33](#_Toc104153945)

[9.2.3 Admin use cases description 35](#_Toc104153946)

[10. Analysis Class 37](#_Toc104153947)

[10.1 Customer’s Analysis Class Diagrams 37](#_Toc104153948)

[**10.1.1 Customer Sign Up** 37](#_Toc104153949)

[**10.1.2 {SCAD} login** 37](#_Toc104153950)

[**10.1.3 {SCAD}log out** 37](#_Toc104153951)

[**10.1.4 Browse All Items** 38](#_Toc104153952)

[**10.1.5 Search by name** 38](#_Toc104153953)

[**10.1.6 Filter by category** 38](#_Toc104153954)

[**10.1.7 Filter by price** 39](#_Toc104153955)

[**10.1.8 Add to list** 39](#_Toc104153956)

[**10.1.9 View List items** 40](#_Toc104153957)

[**10.1.10 Find best offers** 40](#_Toc104153958)

[**10.1.11 Find Cheapest supermarket to buy a list from** 41](#_Toc104153959)

[**10.1.12 Recover forgotten password** 41](#_Toc104153960)

[10.2 SMC’s Analysis Class Diagrams 42](#_Toc104153961)

[**10.2.1 SMC Sign Up request** 42](#_Toc104153962)

[**10.2.2 Add offers by uploading an excel file** 42](#_Toc104153963)

[**10.2.4 Add offers by scanning booklets** 43](#_Toc104153964)

[**10.2.5 Add an offer manually** 43](#_Toc104153965)

[10.3 Administrator’s Analysis Class Diagrams 44](#_Toc104153966)

[**10.3.1 Create Data Provider Account** 44](#_Toc104153967)

[**10.3.2 Approve SMC Sign Up request** 44](#_Toc104153968)

[11. Interaction Diagram 45](#_Toc104153969)

[11.1 Customer’s Interaction Diagrams 45](#_Toc104153970)

[**11.1.3 Filter by category** 46](#_Toc104153971)

[**11.1.4 Filter by price** 47](#_Toc104153972)

[**11.1.5 View list** 48](#_Toc104153973)

[**11.1.6 Find best offers** 49](#_Toc104153974)

[**11.1.7 Find cheapest SMC to shop list** 50](#_Toc104153975)

[11.2 SMC Interaction Diagrams 51](#_Toc104153976)

[**11.2.1 Signup Request** 51](#_Toc104153977)

[**11.2.2 Add offers using an excel file** 52](#_Toc104153978)

[**11.2.3 Generate API key** 53](#_Toc104153979)

[11.3 Data provider’s Interaction Diagrams 54](#_Toc104153980)

[*11.3.1 Add Offer by Scanning Booklet* 54](#_Toc104153981)

[**11.3.2 Add an offer manually** 55](#_Toc104153982)

[11.4 Administrator’s Interaction Diagrams 56](#_Toc104153983)

[**11.4.1 Create data provider account** 56](#_Toc104153984)

[**11.4.2 Approve SMC account** 57](#_Toc104153985)

[12. Design Class 58](#_Toc104153986)

[12.2 Customer’s Class Diagrams 59](#_Toc104153987)

[**12.2.1 Browse All Items** 59](#_Toc104153988)

[**12.2.2 Filter by category** 60](#_Toc104153989)

[**12.2.3 Filter by price** 61](#_Toc104153990)

[**12.2.4 View List items** 62](#_Toc104153991)

[**12.2.6 Find Cheapest supermarket to buy a list from** 64](#_Toc104153992)

[12.3 SMC’s Class Diagrams 65](#_Toc104153993)

[12.3.1 SMC Sign Up Request 65](#_Toc104153994)

[**12.3.2 Add offers by uploading an excel file** 66](#_Toc104153995)

[**12.3.3 Generate API Key** 67](#_Toc104153996)

[12.4 Data Provider’sClass Diagrams 68](#_Toc104153997)

[**12.4.1 Add offers by scanning booklet** 68](#_Toc104153998)

[**12.4.2 Add an offer manually** 69](#_Toc104153999)

[12.5 Administrator’s Class Diagrams 70](#_Toc104154000)

[**12.5.1 Create Data Provider Account** 70](#_Toc104154001)

[**12.5.2 Approve SMC account** 71](#_Toc104154002)

[13. System Architecture 72](#_Toc104154003)

[13.1 Architectural Styles 72](#_Toc104154004)

[13.2 Component Diagram 74](#_Toc104154005)

[14. User Interface Mock-up 75](#_Toc104154006)

[**14.1.1 Sign up screen** 75](#_Toc104154007)

[**14.1.2 Log in screen** 75](#_Toc104154008)

[**14.1.3 Home page screen** 76](#_Toc104154009)

[**14.1.4 Find best offers screen** 76](#_Toc104154010)

[**14.1.5 View list screen** 77](#_Toc104154011)

[**14.1.6 Find cheapest supermarket screen** 77](#_Toc104154012)

[14.2 Data provider’s screens 78](#_Toc104154013)

[**14.2.1 Main screen** 78](#_Toc104154014)

[**14.2.2 Validate extracted items screen** 78](#_Toc104154015)

[**14.2. Add offers manually screen** 79](#_Toc104154016)

[14.3 SMCs screens 79](#_Toc104154017)

[**14.3.1 Register screen** 79](#_Toc104154018)

[**14.3.2 Main screen** 80](#_Toc104154019)

[**14.3.3 Generate API key screen** 80](#_Toc104154020)

[14.4 Administrators’ screens 81](#_Toc104154021)

[**14.4.1 create data provider account** 81](#_Toc104154022)

[**14.4.2 Approve SMC’c accounts** 81](#_Toc104154023)

[16.Algorithms 83](#_Toc104154024)

[17. Expected Deployment 85](#_Toc104154025)

[18. Test Scenario 86](#_Toc104154026)

[19. Project Status 89](#_Toc104154027)

[20. Conclusion 90](#_Toc104154028)

[21. Reference 90](#_Toc104154029)

[22. Appendices 92](#_Toc104154030)

[22.1 Inspection Report 92](#_Toc104154031)

[22.2 Formal Review Reports 93](#_Toc104154032)

[22.3 use cases description 94](#_Toc104154033)

[22.4 Interaction Diagrams 100](#_Toc104154034)

[**22.4.1 Search item by name** 100](#_Toc104154035)

[**22.4.2 Add to list** 101](#_Toc104154036)

[**22.4.3 Customer sign up** 102](#_Toc104154037)

[**22.4.4 {SCAD} login** 103](#_Toc104154038)

[**22.4.5 {SCAD} log out** 104](#_Toc104154039)

[**22.4.6 {SCAD} Recover forgotten password** 105](#_Toc104154040)

[22.5 Class diagrams 106](#_Toc104154041)

[**22.5.1 Search item by name** 106](#_Toc104154042)

[**22.5.2 Add to list** 107](#_Toc104154043)

[**22.5.3 Customer Sign up** 108](#_Toc104154044)

[**22.5.4 {SCAD} login** 108](#_Toc104154045)

[**22.5.5 {SCAD} recover forgotten password** 109](#_Toc104154046)

Table of Figures

[*Figure 1 . Carrefour logo……………………………………………..Figure 2 . Carrefour user interface* 10](#_Toc104151534)

[*Figure 3 . Danube logo………………………………………………..Figure 4 . Danube interface* 11](#_Toc104151535)

[*Figure 5 . Flipp logo…………………………………………………….Figure 6 . Flipp interface* 12](#_Toc104151536)

[*Figure 7 . Basket logo……………………………………………….. Figure 8 . Basket interface* 13](#_Toc104151537)

[*Figure 9 . Project plan* 16](https://studentksuedu-my.sharepoint.com/personal/439101185_student_ksu_edu_sa/Documents/GP_SWE-496%20(1)%20(1)%20(1)%20(2)%20(1).docx#_Toc104151538)

[*Figure 10 . Application use case diagram* 21](#_Toc104151539)

[*Figure 11 . Web page use case diagram* 28](#_Toc104151540)

[*Figure 12 . Customer sign up analysis class diagram* 37](#_Toc104151541)

[*Figure 13 . Login analysis class diagram* 37](#_Toc104151542)

[*Figure 14 . Log out analysis class diagrams* 37](#_Toc104151543)

[*Figure 15 . Browse all items analysis class diagram* 38](#_Toc104151544)

[*Figure 16 . Search by name analysis class diagram* 38](#_Toc104151545)

[*Figure 17 . Filter by category analysis class diagram* 38](#_Toc104151546)

[*Figure 18 . Filter by price analysis class diagram* 39](#_Toc104151547)

[*Figure 19 . Add to list analysis class diagram* 39](#_Toc104151548)

[*Figure 20 . View list items analysis class diagram* 40](#_Toc104151549)

[*Figure 21 . Find best offers analysis class diagram* 40](#_Toc104151550)

[*Figure 22 . Find cheapest place to buy a list from analysis class diagram* 41](#_Toc104151551)

[*Figure 23 . Recover forgotten password analysis class diagram* 41](#_Toc104151552)

[*Figure 24 . Supermarket companies sign up analysis class diagram* 42](#_Toc104151553)

[*Figure 25 . Add offers by uploading an excel file analysis class diagram* 42](#_Toc104151554)

[*Figure 26 . Generate API Key analysis class diagram* 43](#_Toc104151555)

[*Figure 27 . Add offers by scanning booklets analysis class diagram* 43](#_Toc104151556)

[*Figure 28 . Add an offer manually analysis class diagram* 43](#_Toc104151557)

[*Figure 29 . Create data provider account Analysis class diagram* 44](#_Toc104151558)

[*Figure 30 . Approve SMC sign Up request analysis class diagram* 44](#_Toc104151559)

[*Figure 31 . Browse all items interaction diagram* 45](#_Toc104151560)

[*Figure 32 . Search by name interaction diagram* 46](#_Toc104151561)

[*Figure 33 . Filter by category interaction diagram* 46](#_Toc104151562)

[*Figure 34 . Filter by price interaction diagram* 47](#_Toc104151563)

[*Figure 35 . view list interaction diagram* 48](#_Toc104151564)

[*Figure 36 . Find best offers interaction diagram* 49](#_Toc104151565)

[*Figure 37 . Find cheapest SMC to shop list interaction diagram* 50](#_Toc104151566)

[*Figure 38 . SMC signup request interaction diagram* 51](#_Toc104151567)

[*Figure 39 . Add offers using an excel file interaction diagram* 52](#_Toc104151568)

[*Figure 40 . Generate API key interaction diagram* 53](#_Toc104151569)

[*Figure 41 . Add offer by scanning booklet interaction diagram* 54](#_Toc104151570)

[*Figure 42 . Add offer manually interaction diagram* 55](#_Toc104151571)

[*Figure 43 . Create data provider account interaction diagram* 56](#_Toc104151572)

[*Figure 44 . Approve SMC account interaction diagram* 57](#_Toc104151573)

[*Figure 45 . All used classes* 58](#_Toc104151574)

[*Figure 46 . Browse all items class diagram* 59](#_Toc104151575)

[*Figure 47 . Filter by category class diagram* 60](#_Toc104151576)

[*Figure 48 . Filter by price class diagram* 61](#_Toc104151577)

[*Figure 49 . View list items class diagram* 62](#_Toc104151578)

[*Figure 50 . Find best offers class diagram* 63](#_Toc104151579)

[*Figure 51 . Find cheapest place to buy a list from class diagram* 64](#_Toc104151580)

[*Figure 52 . Supermarket companies sign up class diagram* 65](#_Toc104151581)

[*Figure 53 . Add offers by uploading an excel file class diagram* 66](#_Toc104151582)

[*Figure 54 . Generate API Key class diagram* 67](#_Toc104151583)

[*Figure 55 . Add offers by scanning booklets class diagram* 68](#_Toc104151584)

[*Figure 56 . Add offers manually class diagram* 69](#_Toc104151585)

[*Figure 57 . Create data provider account class diagram* 70](#_Toc104151586)

[*Figure 58 . Approve SMC account class diagram* 71](#_Toc104151587)

[*Figure 59 . Component diagram* 74](#_Toc104151588)

[*Figure 60 . Sign up screen customer……………………………Figure 61 . Log in screen customer* 75](#_Toc104151589)

[*Figure 62 . Homepage screen customer…………………….Figure 63 . Find best offer screen customer* 76](#_Toc104151590)

[*Figure 64 . View list screen customer……………………….Figure 65 . Find cheapest supermarket screen customer* 77](#_Toc104151591)

[*Figure 66 . Data provider’s main screen* 78](#_Toc104151592)

[*Figure 67 . Validate extracted offers screen* 78](#_Toc104151593)

[*Figure 68 . Add offers manually screen* 79](#_Toc104151594)

[*Figure 69 . SMC Registration screen* 79](#_Toc104151595)

[*Figure 70 . SMC main screen* 80](#_Toc104151596)

[*Figure 71 . Generate API screen* 80](#_Toc104151597)

[*Figure 72 . Create data provider account screen* 81](#_Toc104151598)

[*Figure 73 . Approve SMC’s account screen* 81](#_Toc104151599)

[*Figure 74 . Database schema* 82](#_Toc104151600)

[*Figure 75 . System deployment diagram* 85](#_Toc104151601)

[*Figure 76 . Search item by name interaction diagram* 100](#_Toc104151602)

[*Figure 77 . Add to list interaction diagram* 101](#_Toc104151603)

[*Figure 78 . Add to list Interaction diagram* 102](#_Toc104151604)

[*Figure 79 . login interaction diagram* 103](#_Toc104151605)

[*Figure 80 . log out interaction diagram* 104](#_Toc104151606)

[*Figure 81 . Recover forgotten password interaction diagram* 105](#_Toc104151607)

[*Figure 82 . Search item by name class diagram* 106](#_Toc104151608)

[*Figure 83 . Add to list class diagram* 107](#_Toc104151609)

[*Figure 84 . Customer sign up class diagram* 108](#_Toc104151610)

[*Figure 85 . Login class diagram* 108](#_Toc104151611)

[*Figure 86 . Recover forgotten password class diagram* 109](#_Toc104151612)

List of Tables

[Table 1 . Shortcut table 9](#_Toc104151641)

[Table 2 . Competitors' comparison table 14](#_Toc104151642)

[Table 3 . Risk/Constraints table 15](#_Toc104151643)

[Table 4 . Filter by category use case description 22](#_Toc104151644)

[Table 5 . Filter by category use case description 23](#_Toc104151645)

[Table 6 . Filter by price use case description 24](#_Toc104151646)

[Table 7 . View list use case description 25](#_Toc104151647)

[Table 8 . Find best offer use case description 26](#_Toc104151648)

[Table 9 . Find cheapest supermarket to shop a list use case description 27](#_Toc104151649)

[Table 10 . Supermarket company signup use case description 29](#_Toc104151650)

[Table 11 . Add offers by uploading an excel file use case description 30](#_Toc104151651)

[Table 12 . Generate API key use case description 31](#_Toc104151652)

[Table 13 . Add offers by using system API use case description 32](#_Toc104151653)

[Table 14 . Add offers by scanning booklet use case description 34](#_Toc104151654)

[Table 15 . Add an offer manually use case description 35](#_Toc104151655)

[Table 16 . Create data provider account use case description 36](#_Toc104151656)

[Table 17 . SMC test scenario 86](#_Toc104151657)

[Table 18 . Data provider test scenario 87](#_Toc104151658)

[Table 19 . Admin test scenario 87](#_Toc104151659)

[Table 20 . Customer test scenario 88](#_Toc104151660)

[Table 21 . Project status 89](#_Toc104151661)

[Table 22 . Inspection report 92](#_Toc104151662)

[Table 23 . Formal review 93](#_Toc104151663)

[Table 24 . Search item by name use case description 94](#_Toc104151664)

[Table 25 . Add to list use case description 95](#_Toc104151665)

[Table 26 . Customer sign up use case description 96](#_Toc104151666)

[Table 27 . {SCAD} log in use case descrpition 97](#_Toc104151667)

[Table 28 . {SCAD} Log out use case description 98](#_Toc104151668)

[Table 29 . Recover forgotten password use case description 99](#_Toc104151669)

# Introduction

Occasionally, supermarkets offer discounts and special offers on their products in an effort to attract new customers or to increase the satisfaction of existing customers. One of the most important ways to make these offers to customers is to publish brochures in the streets or public places or through their own websites.

Market customers are primarily interested in getting the best price for a particular commodity, regardless of who is selling it, in order to maximize their savings and to get the best deal. Getting all the announced brochures in order to discover the lowest price for the commodity can be a difficult and time-consuming task and may lead to missing a much better offer in other markets.

In this report, we will document the software development life cycle of an offers comparator mobile application that will ease supermarkets' customers in the mentioned challenges. It will provide many features that can aid and ease their shopping.

The major feature of this application is to let the user be able to enter his/her grocery list for the application to compare this list between a variety number of supermarkets and finds the supermarket with the lowest price for that particular list, or if the user wants the application could provide him/her with the supermarket with the lowest price for each item “the most optimized path”.

To provide that feature we need a system that helps the application to know what the currently applicable offers are, for that problem we will provide a variety of solutions as intelligently let the system recognize the offers from a booklets/brochure by simply scanning them and other solutions that are discussed later on in this report.

# Shortcut Table

|  |  |
| --- | --- |
| Shortcut | Meaning |
| SMC | Supermarket Company |
| {SCAD} | Whenever the word SCAD appears in this document this means that the place where it appears is applied for all the actors (SMC, Customer, Admin, Data Provider). |
| Ref | Reference |

Table 1 . Shortcut table

# Domain Analysis

Domain analysis is the process of analysing related software systems in a

given domain to identify design patterns in software and data.[1]

We will study and analyse a group of software systems to learn which parts of the Systems are roughly equivalent and which parts are unique. Such analysis allows us to identify how the systems are used and to see the patterns within the domain. This type of analysis has also led us to identify improvements in the development of our system.

In this section, we give a brief description of each studied application and its main features.

## 3.1 Competitors

### **3.1.1 Carrefour**

Carrefour[2] is a supermarket service application where you can browse and shop for discounted items from only The Carrefour supermarket, like grocery, personal care, baby products, electronics, smartphones and look for their best offers. their main feature is identifying offers for given users and the support of both Arabic and English interfaces.

App logo: User interface:

*Figure 1 . Carrefour logo Figure 2 . Carrefour user interface*



Graphical user interface, website

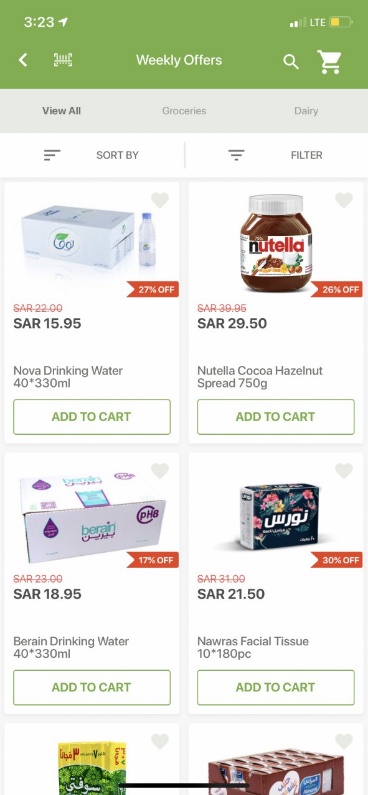
Description automatically generated

### **3.1.2 Danube**

Danube[3] is a mobile application that helps users look for the best offers from only the Danube supermarket. Its interface supports both English and Arabic users and has a fresh look. users can browse for discounted items, search for their best offers, and filter their prices and calories.

App logo: User interface:

*Figure 3 . Danube logo Figure 4 . Danube interface*

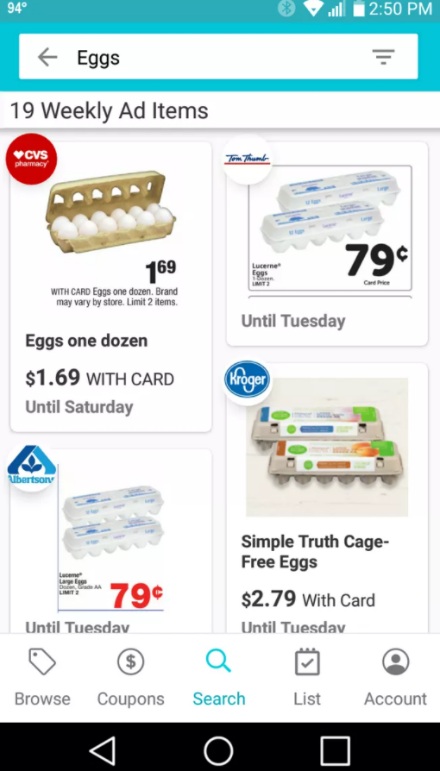


### **3.1.3 Flipp**

Flipp[4] is a North American application that allows you to browse weekly ads to find deals from multiple stores including Walmart, Dollar General, and Walgreens. their main feature is comparing prices between stores but the app also lacks Options of use for categories other than groceries

App logo: User interface:

*Figure 5 . Flipp logo Figure 6 . Flipp interface*

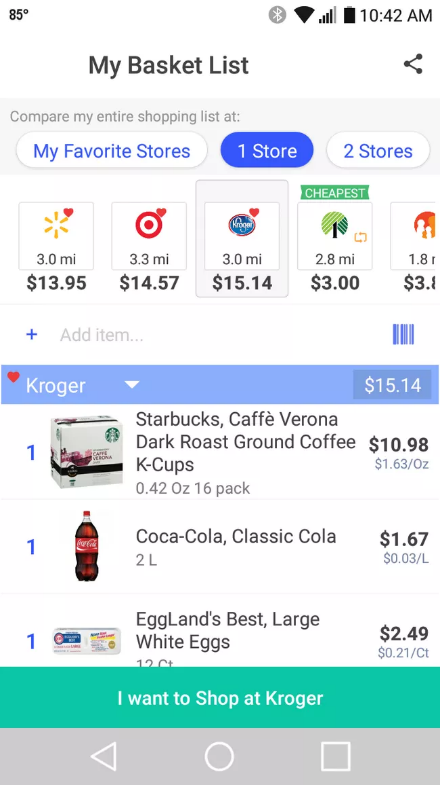


### **3.1.4 Basket**

Basket[5] is a North American application where you can create a shopping list and it’ll price checks shopping items between stores to see which stores have certain products and what the total price is at each store.

App logo: User interface:

*Figure 7 . Basket logo Figure 8 . Basket interface*



## 3.2- Competitors' comparison

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **features** | **Offerize** | **Carrefour** | **Danube** | **Flipp** | **Basket** |
| Provide offer comparison from multiple stores | ✓ |  |  | ✓ | ✓ |
| Available in Saudi Arabia | ✓ | ✓ | ✓ |  |  |
| Item Searching and finding sale prices or best deals | ✓ | ✓ | ✓ | ✓ | ✓ |
| Supporting Arabic/English. | ✓ | ✓ | ✓ |  |  |
| Filtering by prices (lowest to highest or highest to lowest) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Extracting the offers information from a booklet file | ✓ |  |  |  |  |

Table 2 . Competitors' comparison table

# Risk/Constraints

In this section we identify risks, constraints, types, severity, and probability of occurrence on projects/products, and how to avoid them.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Risk/Constraint** | **Type** | **Severity** | **Likelihood** | **Solution** |
| 1 | A misunderstanding of the requirements can lead to wasted efforts that adversely affect the schedule | Project | High | Medium | Our advisor and ourselves need to remain in constant communication to prevent this from occurring. |
| 2 | Limited knowledge of the technologies that will be used deep learning, cross-platform application and frameworks | Operational | High | low | Organize two-person groups focusing on specific technologies by taking online courses |
| 3 | There are multiple kinds of users, so the UI might not appeal to them. | Aesthetics | Medium | Medium | We will make the UI as simple as possible for the users, and we will allow them to give us feedback on our UI design. |
| 4 | Writing the terms and conditions, policy, and other documents. Since our system will be collecting data from multiple supermarkets and storing it on our servers | Business | High | Medium | To ensure a safe and legal environment, we will consult with legal counsel before writing these documents |
| 5 | Conflicts between project deliverables and other exams/assignments | Project | Medium | High | Organize and distribute the work and avoid delaying. |
| 6 | Limited marketing expertise makes it difficult to market the product | Marketability | low | High | Contact some social media advertisers. |
| 7 | team members time management | Project | Medium | Medium | Organize two weekly meetings |

Table 3 . Risk/Constraints table

# Project Plan

In this section we will follow a plan in developing the project to help us organize our time efficiently and avoid delays and other time-related issues. Figure 10. shows the phases of the project and the tasks assigned to team members.



*Figure 9 . Project plan*

# Quality Assurance Plan

The Quality Assurance Plan ensures the finished product meets all criteria making it the best possible quality.[6]

In this section, we define the criteria and processes that will ensure and verify that software meets specific requirement objectives throughout the Software Lifecycle.

- **Inspections**:

Inspections are aimed to find errors in the software documentation or code by having meetings between team members for inspecting errors. Meetings will be held weekly, At every meeting, all members will try to find any issues in grammar, spelling or any other error. (See section 22.1).

- **Formal Reviews**:

After every phase, we will show our work to our adviser and alter the project according to his comments. (See section 22.2).

- **Team training**:

Each team member will take online courses related to deep learning to improve our knowledge.

.

# Requirements

Requirements are the description of features and functionalities of the target product that meet the user's needs.[7] In this section, we gather Requirements that convey the expectations of users from the software product.

## 7.1 Functional Requirements

There are four user types of the proposed system, and each one has certain requirements.

### **7.1.1 Customer’s Functional Requirements**

7.1.1.1 Customer shall be able to create an account.

7.1.1.2Customer shall be able to recover his/her forgotten password.

7.1.1.3 Customer shall be able to log in to the system.

7.1.1.4 Customer shall be able to log out from the system.

7.1.1.5 Customer shall be able to browse all the supermarkets’ items.

7.1.1.6 Customer shall be able to search for an item by its name.

7.1.1.7 Customer shall be able to filter items by their category.

7.1.1.8 Customer shall be able to filter items by their offers’ new price.

7.1.1.9 Customer shall be able to add an item to his/her list.

7.1.1.10 Customer shall be able to find the cheapest SMC to buy each item on the list from.

7.1.1.11 Customer shall be able to find the cheapest SMC to buy all his/her list of items from.

7.1.1.12 Customer shall be able to view his/her List.

7.1.1.13 Customer shall be able to remove, increase or decrease the amount of every list item on his/her List.

### **7.1.2 Data Provider Functional Requirements**

7.1.2.1 Data provider shall be able to log in to the system.

7.1.2.2 Data provider shall be able to upload booklet PDF file to the system.

7.1.2.3 The system shall be able to extract the offers information from a provided booklet PDF file.

7.1.2.4 The system shall be able to store offers title, old price, new price, image, start date, and end date, and which SMC it belongs to.

7.1.2.5 The system shall be able to delete an expired offer from the system.

7.1.2.6 Data provider shall be able to validate offers extracted from booklet file by the system.

7.1.2.7 Data provider shall be able to add an offer manually during the validation of extracted offers.

7.1.2.8 Data provider shall be able to edit an offer during the validation of extracted offers.

7.1.2.9 Data provider shall be able to log out from the system.

### **7.1.3 Administrator Functional Requirements**

7.1.3.1 Administrator shall be able to log in to the system.

7.1.3.2 Administrator shall be able to log out from the system.

7.1.3.3 Administrator shall be able to create a data provider account.

7.1.3.4 Administrator shall be able to view all SMCs sign-up requests.

7.1.3.5 Administrator shall be able to accept or decline an SMC sign-up request.

### **7.1.4 SMC Functional Requirements**

7.1.4.1 SMC shall be able to log in to the system.

7.1.4.2 SMC shall be able to create a sign-up request.

7.1.4.3 SMC shall be able to generate only one API key.

7.1.4.4 SMC shall be able to add offers using systems API.

7.1.4.5 SMC shall be able to add offers by uploading an excel file containing offers.

## 7.2 Non-Functional Requirements

7.2.1 The accuracy of extracting the items shall be at least 85%

7.2.2 The system shall be available on both IOS and android devices.

7.2.3 The system load time should not be more than three seconds for users.

7.2.4 The system downtime shall not exceed 3 minutes per week.

7.2.5 The system should be able to handle 500 users at the same time.

7.2.6 The system shall be able to 100% recover from failures.

7.2.7 Users shall be able to understand the system in less than 3 minutes.

# Problem Complexity

Our system has many complex components as we are new to many of the required technologies that must be used to achieve the goals of this project, the causes of those complex components are illustrated below:

1. Some of the features of the system require deep learning technologies. It's a new field for us that requires a lot of knowledge of machine learning and very large data.
2. The persuasion of supermarkets to periodically publish their sale offers by using our application.
3. We must develop a mobile app for the customers and a website for every other actor at the same time but finally we need to connect them with the same database, so data synchronization could be a headache.
4. We will have to provide an API service for SMCs that would like to share their offers with us using their own systems.
5. The system requires the use of multiple processes which must be written in multiple programming languages, so we expect those processes' integration to be complex.

# System Use-Cases

A use case diagram is a graphical depiction of a user's possible interactions with a system.[8] A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams (sequence diagram) as well.As such, it describes the goals of the users, the interactions between the users and the system, and the required behaviour of the system in satisfying these goals.

use cases, use case descriptions are illustrated below:

## 9.1 Customer use case diagram

The following figure shows the use cases of the customer who searches for the best offers.

Diagram

Description automatically generated

*Figure 10 . Application use case diagram*

## 9.1.2 Customer use cases description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Browse All Items | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the customer browses all the items in the system grouped by their category. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **customer** | **System** | |
| 1. The customer asks the system to browse all items. | 2. The system gets all the items.  3. The system group the items by category,  4. The system displays all items grouped by category. | |
| **Alternative and exceptional flows:**  None | | |
| **Post-conditions:**   * All items are displayed and grouped by category | | |

Table 4 . Filter by category use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Filter by Category | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the customer filters the displayed items based on their category. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **Customer** | **System** | |
| 1. The customer asks the system to filter the items by category.  4. The customer chooses a specific category. | 2. The system displays all categories.  3. The system asks the customer to select a category.  5. The system gets all the items that is in the selected category.  6. The system displays all items in this category. | |
| **Alternative and exceptional flows:**  None | | |
| **Post-conditions:**   * All items from the selected category are displayed. | | |

Table 5 . Filter by category use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Filter by Price | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the customer filters the displayed items that have at least an offer within a selected price range by the customer. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **Customer** | **System** | |
| 1- The customer asks the system to filter by price.  3- The customer chooses the price range | 2- The system asks the customer to select the range price.  4. The system gets all the items that have at least an offer within the selected price range.  5. The system displays all gotten items. | |
| **Alternative and exceptional flows:**  **4a. The system finds no items which have an offer within the selected price range:**  4a1. the system displays an apology message. | | |
| **Post-conditions:**   * All items that include an offer from the selected price range are displayed. | | |

Table 6 . Filter by price use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** View List | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the customer can view the items that he/she added to his/her list. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The customer must be logged In. | | |
| **Customer** | **System** | |
| 1- Customer asks the system to view his/her list. | 2- The system gets the items that have been added to the customer list.  3- The system gets the best offer for each item in the list  4- The system displays the list with the best offer to the customer. | |
| **Alternative and exceptional flows:**  **2a. The system finds that the user hasn’t added any item to his/her list:**  2a1. The system displays a "no item has been added to the list" message.  **3a. The system doesn’t find any offer for an item or more in the list:**  3a1. The system displays all items that have offers.  3a2. The system displays all items that don’t have any offers with a "no offers" message attached to every item. | | |
| **Post-conditions:**   * All the items on his/her list are displayed. | | |

Table 7 . View list use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Find Best Offers | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the system provides the customer with the cheapest SMC to buy a certain item from. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **Customer** | **System** | |
| 1. The customer asks the system to provide the cheapest SMC to buy a certain item from. | 2. The system gets all the offers for this specific item.  3. The system sorts the offers of the selected item by price from cheapest to the most expensive.  4. The system displays the sorted offers along with the SMC that provide the offer. | |
| **Alternative and exceptional flows:**  **2a. The system finds no offers for the item:**  2a1. The system displays an apology message. | | |
| **Post-conditions:**   * All the offers of the selected items are displayed sorted by price from cheapest to the most expensive along with the SMC that provide the offer. | | |

Table 8 . Find best offer use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Find the Cheapest SMC To Shop a List | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how does the system provides the customer with the cheapest SMC to shop all his/her list from. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   1. The customer has at least one item added to his/her list. | | |
| **Customer** | **System** | |
| 1. The customer asks the system to get the cheapest SMC to buy his/her list. | 2. The system gets every SMC with offers for all the items in the Customer’s list.  3. The system calculates the sum of all the customer’s list’s items in every SMC in the system.  4. The system compares prices between SMCs.  5. The system displays a list of the SMCs sorted from the cheapest SMC to the most expensive. | |
| **Alternative and exceptional flows:**  **2a. The system finds no SMC with offers for the whole list’s items:**  2a1. The System displays an apology message. | | |
| **Post-conditions:**   * A list of the SMCs sorted from the cheapest SMC to the most expensive is displayed. | | |
|  |  |  |

Table 9 . Find cheapest supermarket to shop a list use case description

## System background users use case diagram

The following figure shows the use cases of the third-party companies that wishes to add their new offers to the system, data provider who browse the internet searching for new offers booklets/brochures to add them to the system, and administrator who manages the system users.

Diagram

Description automatically generated

*Figure 11 . Web page use case diagram*

## 9.2.1 SMC use cases description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:**  SMC Sign Up Request | | |
| **Primary actor:** SMC | **Other actors:** None | |
| **Description:** This use case describes how a SMC create a new account request. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **SMC** | | **System** |
| 1. The SMC asks the system to create an account.  3. The SMC fills out the form and submits it. | | 2. The system asks the SMC to fill out a form.  4. The system validates the given information.  5. The system creates a new account creation request.  6. The system displays a message that tells the user that his account is made but needs admin approval.  7. The system sends the request to an admin to approve the account. |
| **Alternative and exceptional flows:**  **4a. The information entered is invalid or the customer doesn’t exist.**  4a1. The system displays an error message.  4a2.The system returns to step 2. | | |
| **Post-conditions:**   * A new SMC account request is added to the system. | | |

Table 10 . Supermarket company signup use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Added offers by Uploading an Excel File | | |
| **Primary actor:** SMC | | **Other actors:** None |
| **Description:** The use case describes how an SMC upload items offers using excel file. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The SMC has Signed In. | | |
| **SMC** | **System** | |
| 1. The SMC ask the system to upload an excel file.  3. The SMC submit a compressed Folder that conforms to the right format. | 2. The system will ask the SMC to upload the compressed Folder that contains an excel file and all the offers’ images.    4. The system stores new offers.  5. The system displays a successful message. | |
| **Alternative and exceptional flows:**    3. T**he SMC submit a file that doesn’t conform to the right format:**  3a1. The system displays an error message. | | |
| **Post-conditions:**   * Offers are added to the system. | | |

Table 11 . Add offers by uploading an excel file use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Generate API Key | | |
| **Primary actor:** SMC | | **Other actors:** None |
| **Description:** This use case describes how the SMC creates an API key so they be able to upload offers using our API. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The SMC has Signed In. | | |
| **SMC** | **System** | |
| 1. The SMC asks the system to generate a new API key. | 2. The system validates if the SMC doesn’t already have an API key.  3. The system generates a new API key and attaches it to the SMC.  4. The system displays the new API key. | |
| **Alternative and exceptional flows:**  **2a. The system finds an API key attached to the SMC:**  2a1.The System displays the stored API key. | | |
| **Post-conditions:**   * A new SMC API key is created and attached to the logged in SMC. | | |

Table 12 . Generate API key use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Add Offers Using System’s API | | |
| **Primary actor:** SMC | | **Other actors:** None |
| **Description:** This use case describes how the SMC uses the system’s API to add their offer by their selfies. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The SMC has an API key. | | |
| **SMC** | **System** | |
| 1- The SMC sends the offers with their API Key | 2- The system validates the API key.  3- The system stores the valid offers attached to the SMC that the API key belongs to  4- The system responds with the stored offers | |
| **Alternative and exceptional flows:**  **2a. The system doesn’t validate the API key:**  2a1. The system responds with an error message. | | |
| **Post-conditions:**   * The valid offers are stored and attached to SMC the API key belongs to. | | |

Table 13 . Add offers by using system API use case description

## 

## 9.2.2 Data provider use cases description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Add offers by scanning booklet | | |
| **Primary actor:** Data provider | | **Other actors:** None |
| **Description:** This use case describes how the data provider adds booklets’ offers that an SMC has published. | | |
| **Relationships**   * **Includes:** Validate * **Extends:** None | | |
| **Pre-conditions:**   1. The Data provider has logged in. | | |
| **Data provider** | **System** | |
| 1.Data provider asks the system to scan a booklet’s offers.    3. Data provider selects an SMC and uploads the booklet pdf file.        8. Data provider validates offers extracted from booklet file by the system. | 2. System asks the data provider which SMC the booklet is for and to upload a PDF file of the booklet.      4. System scans the booklet file to identify each offer the booklet contains.  5. System extracts information from the identified offers.  6. System attach every offer to the corresponding item  7. System displays offers extracted.      9. System stores new offers.  10. System displays the stored offers. | |
| **Alternative and exceptional flows:**  **8a. Data provider asks the system to edit an offer:**  8a1. System asks data provider to select an offer to edit  8a2. Data provider selects an offer to be changed.  8a3. System displays offer field information to be filled.  8a4. Data provider fills offer information.  8b5. System asks for confirmation.  8b6. Data provider confirms.  8.b7. System validates required information.  8b8. Goes back to primary flow number 8. | | |
| **Post-conditions:**   * Offers with required information are added to the system. | | |
|  |  |  |

Table 14 . Add offers by scanning booklet use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** add an offer manually | | |
| **Primary actor:** Data provider | **Other actors:** None | |
| **Description:** This use case describes how the Data Provider manually add a single offer by providing the offer’s information by hand. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** Add offers by scanning booklet | | |
| **Pre-conditions:**   * The Data provider has logged in. | | |
| **Data provider** | | **System** |
| 1. The Data provider asks the system to add a new offer manually.  3. The Data provider provides the offer information. | | 2. The System asks the Data provider o provider offer information.  4. The system validates the provided information.  5. The system adds the offer to the system. |
| **Alternative and exceptional flows:**  **4a. The system doesn’t validate the provided information.**  4a1. The system displays an error message  4a2. Goes back to primary flow number 2. | | |
| **Post-conditions:**   * The offer is Added to the system. | | |

Table 15 . Add an offer manually use case description

## 9.2.3 Admin use cases description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:**  Create data provider account | | |
| **Primary actor:** Admin | **Other actors:** None | |
| **Description:** This use case describes how the admin creates a new data provider account for a newly joined data provider to the team. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The admin has logged in | | |
| **Admin** | | **System** |
| 1. The admin asks the system to a new data provider account.  3. The admin fills out the form and submits it. | | 2.The system asks the admin to fill the form  4. The system validates the filled information.  5. The system creates a new data provider account.  6. The system displays the new account’s username & password to the admin. |
| **Alternative and exceptional flows:**  **4a. The system doesn’t validate the filled information**  4a1. The system displays an error message.  4a2. Back to primary step 2. | | |
| **Post-conditions:**   * A new data provider account is added to the system. | | |

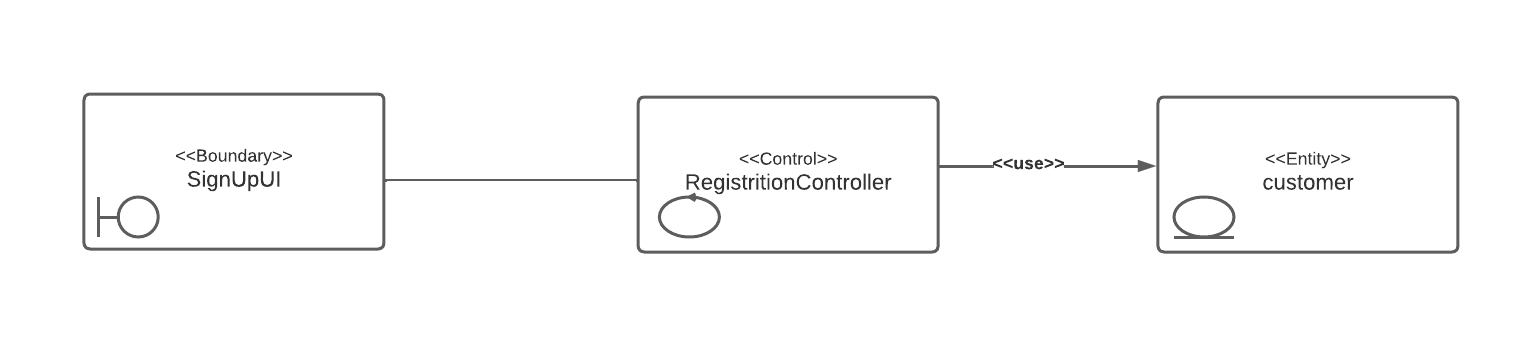
Table 16 . Create data provider account use case description

# 10. Analysis Class

This section describes the analysis class diagrams of the main system functionalities.

## 10.1 Customer’s Analysis Class Diagrams

### **10.1.1 Customer Sign Up**



*Figure 12 . Customer sign up analysis class diagram*

### **10.1.2 {SCAD} login**

Text, letter

Description automatically generated

*Figure 13 . Login analysis class diagram*

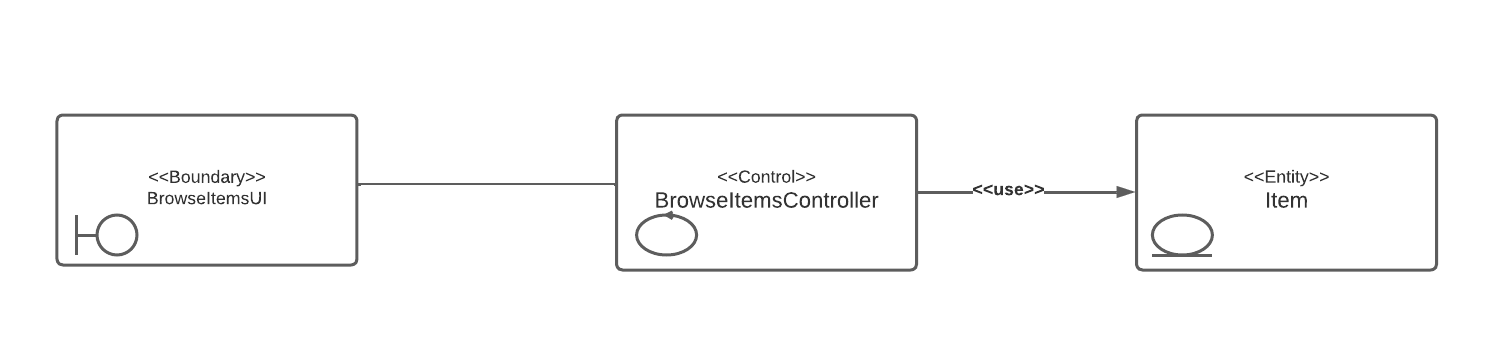
### **10.1.3 {SCAD}log out**

Diagram, box and whisker chart

Description automatically generated

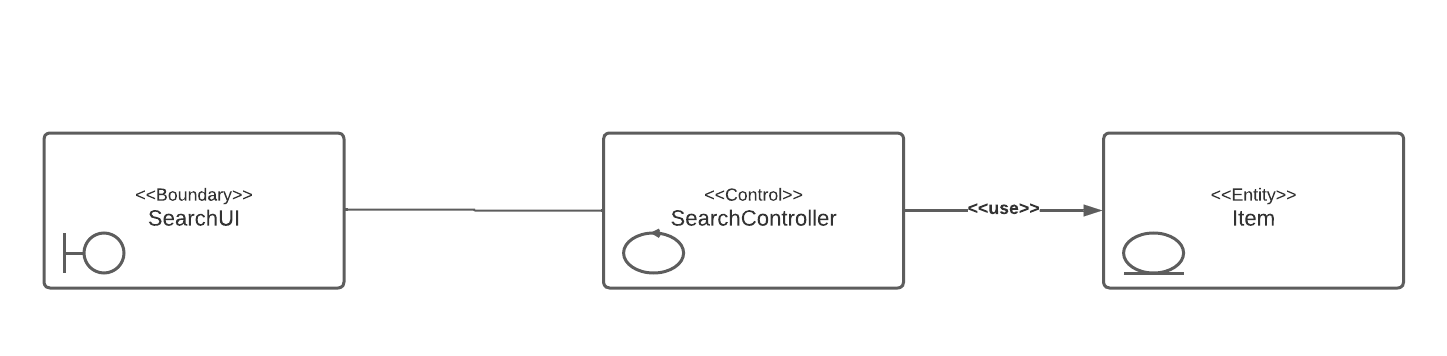
*Figure 14 . Log out analysis class diagrams*

### **10.1.4 Browse All Items**



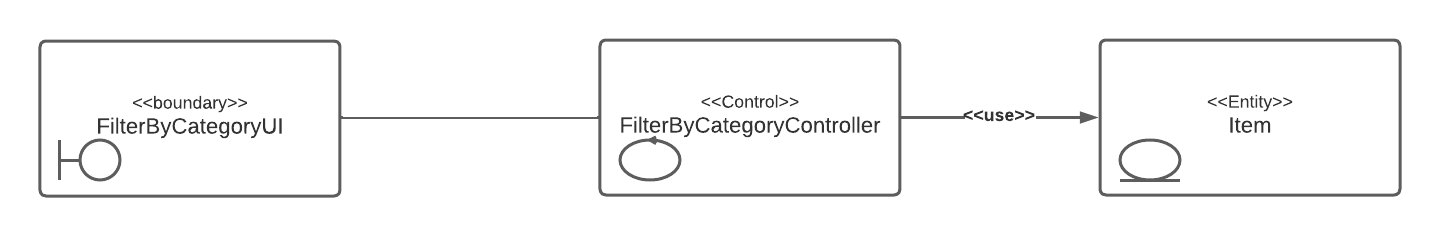
*Figure 15 . Browse all items analysis class diagram*

### **10.1.5 Search by name**



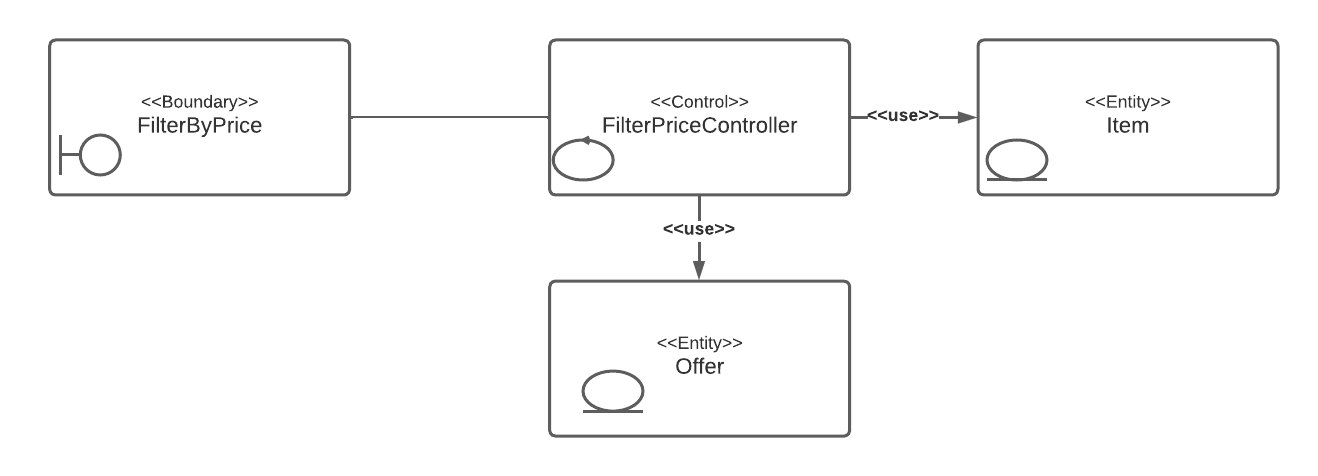
*Figure 16 . Search by name analysis class diagram*

### **10.1.6 Filter by category**



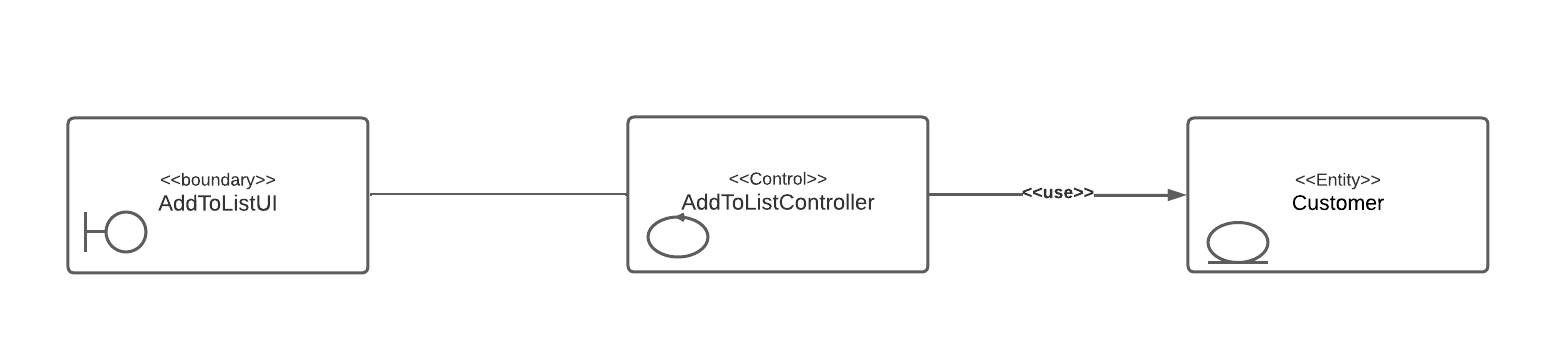
*Figure 17 . Filter by category analysis class diagram*

### **10.1.7 Filter by price**



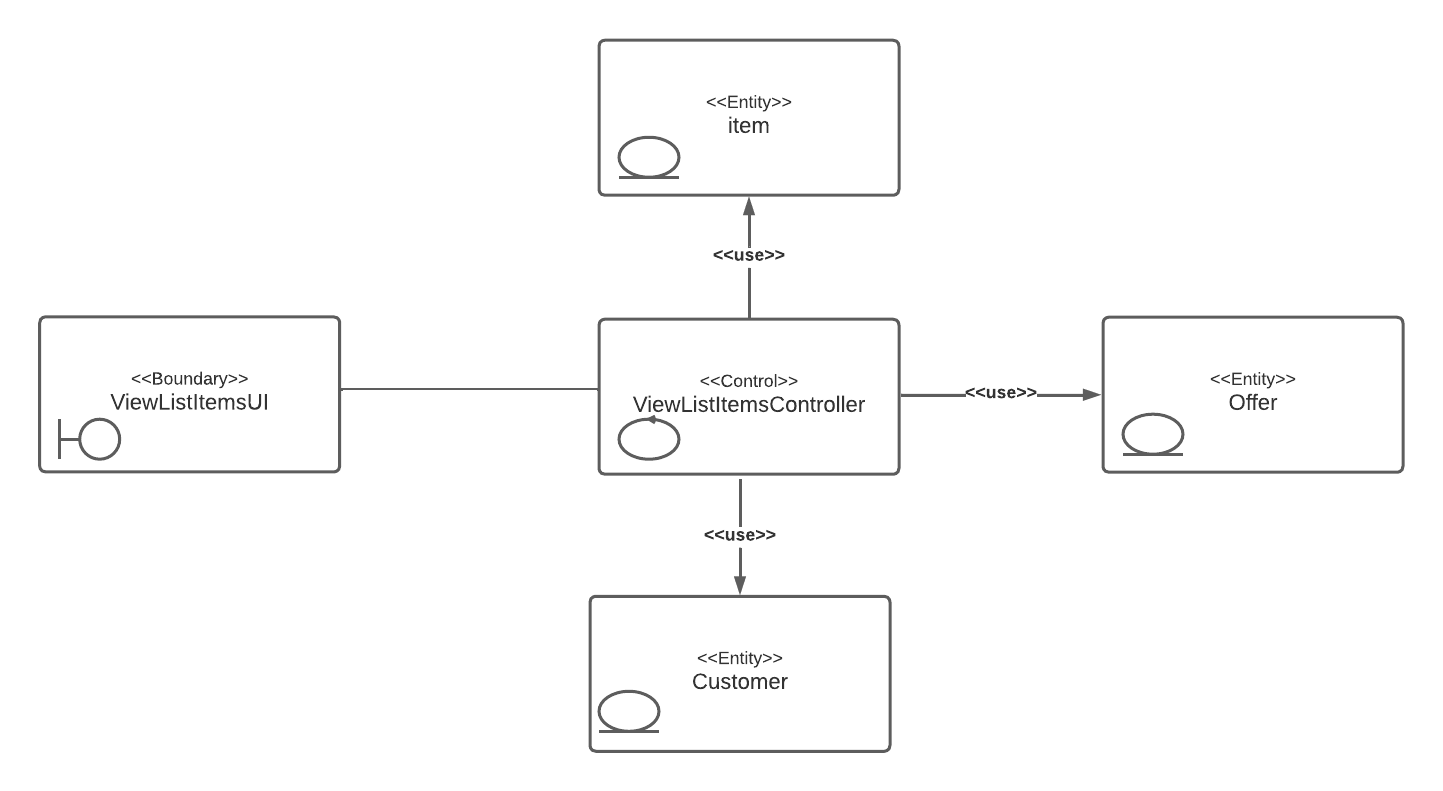
*Figure 18 . Filter by price analysis class diagram*

### **10.1.8 Add to list**



*Figure 19 . Add to list analysis class diagram*

### **10.1.9 View List items**



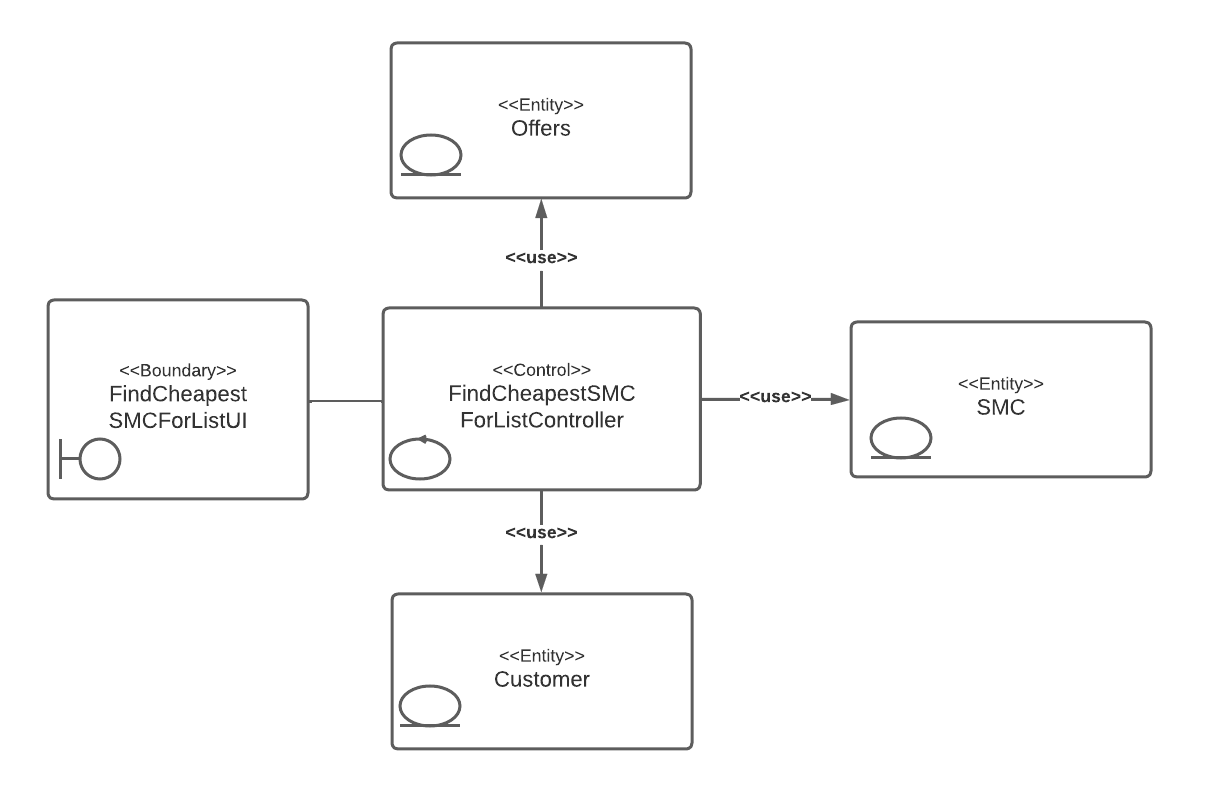
*Figure 20 . View list items analysis class diagram*

### **10.1.10 Find best offers**



*Figure 21 . Find best offers analysis class diagram*

### **10.1.11 Find Cheapest supermarket to buy a list from**



*Figure 22 . Find cheapest place to buy a list from analysis class diagram*

### **10.1.12 Recover forgotten password**

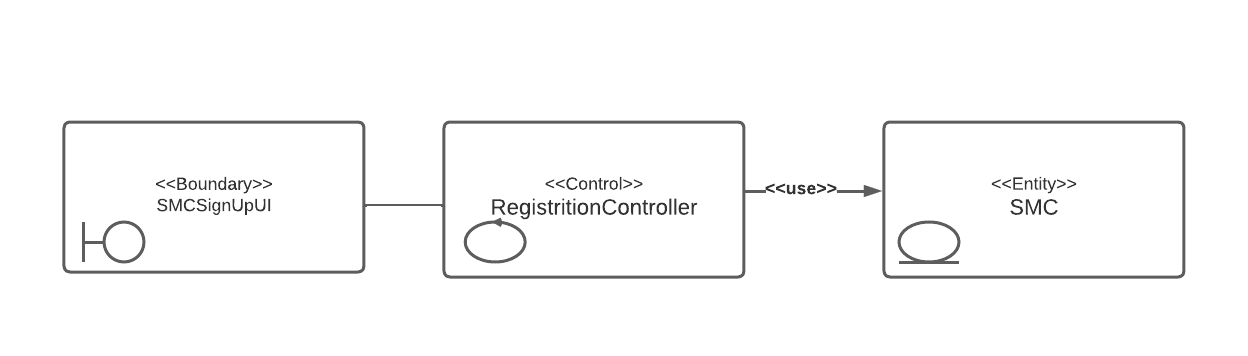
Text

Description automatically generated with low confidence

*Figure 23 . Recover forgotten password analysis class diagram*

## 10.2 SMC’s Analysis Class Diagrams

### **10.2.1 SMC Sign Up request**

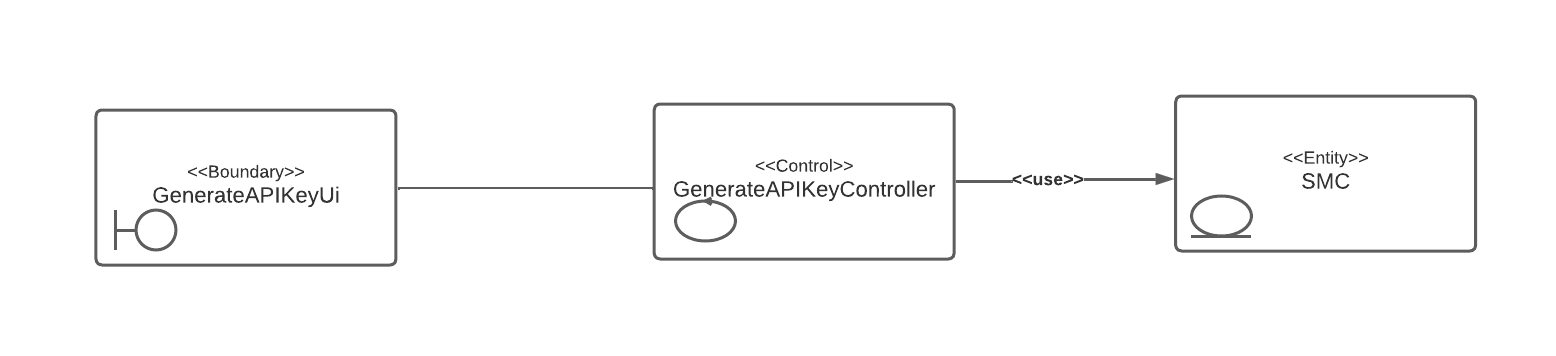


*Figure 24 . Supermarket companies sign up analysis class diagram*

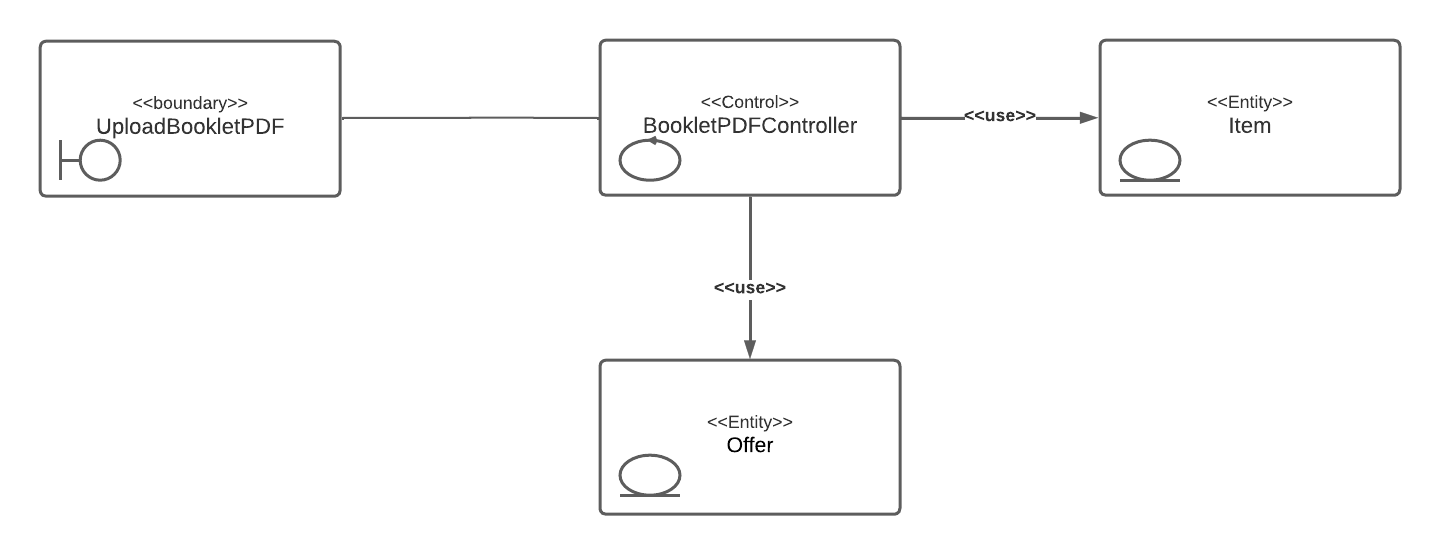
### **10.2.2 Add offers by uploading an excel file**

*Figure 25 . Add offers by uploading an excel file analysis class diagram*

10.2.3 Generate API Key

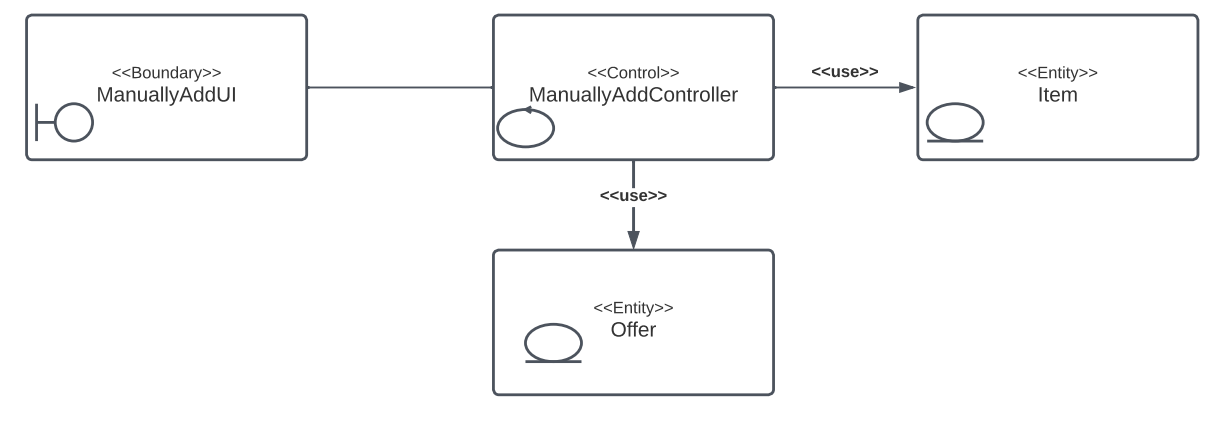
*Figure 26 . Generate API Key analysis class diagram*

### **10.2.4 Add offers by scanning booklets**



*Figure 27 . Add offers by scanning booklets analysis class diagram*

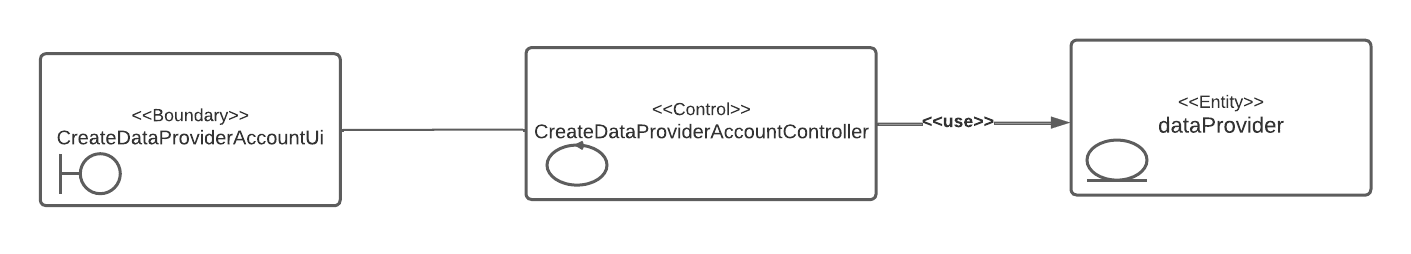
### **10.2.5 Add an offer manually**



*Figure 28 . Add an offer manually analysis class diagram*

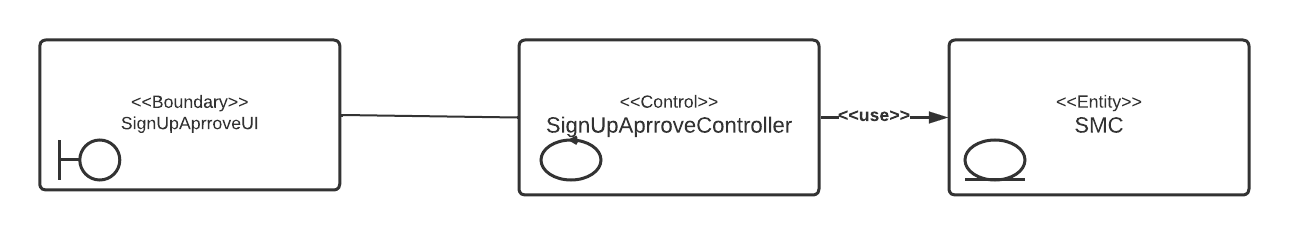
## 10.3 Administrator’s Analysis Class Diagrams

### **10.3.1 Create Data Provider Account**



*Figure 29 . Create data provider account Analysis class diagram*

### **10.3.2 Approve SMC Sign Up request**



*Figure 30 . Approve SMC sign Up request analysis class diagram*

# Interaction Diagram

Interaction diagrams show how objects interact with each other and the order in which those interactions occur. It’s important to note that they show the interactions for a particular scenario. The processes are represented vertically, and interactions are shown as arrows.

In this section, we will show the interaction diagram for the main system functionality.

## 11.1 Customer’s Interaction Diagrams

Diagram

Description automatically generated11.1.1 Browse All Items

*Figure 31 . Browse all items interaction diagram*

### 

11.1.2 Search by nameDiagram

Description automatically generated

*Figure 32 . Search by name interaction diagram*

### Diagram Description automatically generated**11.1.3 Filter by category**

*Figure 33 . Filter by category interaction diagram*

### **11.1.4 Filter by price** Diagram Description automatically generated

*Figure 34 . Filter by price interaction diagram*

### Diagram Description automatically generated**11.1.5 View list**

*Figure 35 . View list interaction diagram*

### Diagram Description automatically generated**11.1.6 Find best offers**

*Figure 36 . Find best offers interaction diagram*

### 

### Diagram Description automatically generated**11.1.7 Find cheapest SMC to shop list**

*Figure 37 . Find cheapest SMC to shop list interaction diagram*

## 11.2 SMC Interaction Diagrams

### Diagram Description automatically generated**11.2.1 Signup Request**

*Figure 38 . SMC signup request interaction diagram*

Diagram

Description automatically generated**11.2.2 Add offers using an excel file**

*Figure 39 . Add offers using an excel file interaction diagram*

### Diagram Description automatically generated**11.2.3 Generate API key**

*Figure 40 . Generate API key interaction diagram*

## 11.3 Data provider’s Interaction Diagrams

## 

## Diagram, schematic Description automatically generated*11.3.1 Add Offer by Scanning Booklet*

*Figure 41 . Add offer by scanning booklet interaction diagram*

### Diagram Description automatically generated**11.3.2 Add an offer manually**

*Figure 42 . Add offer manually interaction diagram*

## 11.4 Administrator’s Interaction Diagrams

### Diagram Description automatically generated**11.4.1 Create data provider account**

*Figure 43 . Create data provider account interaction diagram*

### **11.4.2 Approve SMC account**

Diagram

Description automatically generated

*Figure 44 . Approve SMC account interaction diagram*

# Design Class

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods) and the relationships among objects.[9] Classes are illustrated as the following:

A picture containing timeline

Description automatically generated12.1 All used classes *Figure 45 . All used classes*

12.2 Customer’s Class Diagrams:

### 

### Diagram Description automatically generated**12.2.1 Browse All Items**

*Figure 46 . Browse all items class diagram*

### 

### Diagram Description automatically generated**12.2.2 Filter by category**

*Figure 47 . Filter by category class diagram*

### **12.2.3 Filter by price**

Diagram

Description automatically generated

*Figure 48 . Filter by price class diagram*

### **12.2.4 View List items**

Diagram

Description automatically generated

*Figure 49 . View list items class diagram*

*Diagram

Description automatically generated12.2.5 Find best offers*

*Figure 50 . Find best offers class diagram*

### Diagram Description automatically generated**12.2.6 Find Cheapest supermarket to buy a list from**

*Figure 51 . Find cheapest place to buy a list from class diagram*

## 12.3 SMC’s **Class Diagrams**

### Diagram Description automatically generated12.3.1 SMC Sign Up Request

*Figure 52 . Supermarket companies sign up class diagram*

### Diagram Description automatically generated**12.3.2 Add offers by uploading an excel file**

*Figure 53 . Add offers by uploading an excel file class diagram*

### Diagram Description automatically generated**12.3.3 Generate API Key**

*Figure 54 . Generate API Key class diagram*

## 12.4 Data Provider’s**Class Diagrams**

### Diagram Description automatically generated**12.4.1 Add offers by scanning booklet**

*Figure 55 . Add offers by scanning booklets class diagram*

### 

### Diagram Description automatically generated**12.4.2 Add an offer manually**

*Figure 56 . Add offers manually class diagram*

## 

## 12.5 Administrator’s **Class Diagrams**

### Diagram Description automatically generated**12.5.1 Create Data Provider Account**

*Figure 57 . Create data provider account class diagram*

### Diagram Description automatically generated**12.5.2 Approve SMC account**

*Figure 58 . Approve SMC account class diagram*

# System Architecture

## 13.1 Architectural Styles

In this section, we describe the system architecture of our system. We provided the design decision, component diagram, and architectural style for our system.

Software architecture is the foundation of a software system, The system architecture is an important concept that shows the overall structure of the system and how it should be organized.

When designing an architecture, we define a solution to meet requirements, Enable and inhibit quality attributes, Impose implementation constraints. The result of this process is a model that shows the architectural design of the system in terms of communicating components.

As a result of careful consideration and analysis of other architectures, we have decided that MVC architecture is the most appropriate choice for implementation.

The Model-view-controller architecture splits the system into three partitions:

1. **Model:** Responsible for maintaining data, Handle data logically. The model is connected to the database so anything you do with data. Adding or retrieving data is done in the model component. The model talks to the database back and forth and then it responds and gives the needed data to the controller.

2. **View:** Data representation is done by the view component. It actually generates UI or user interface for the user and requests data through the controller from the model and displays it.

3. **Controller:** Enables the interconnection between the views and the model so it acts as an intermediary, it controls the data flow into a model object and updates the view whenever data changes.

**Justification of the selected architecture style:**

The reasons for the system to be implemented using MVC make it easier to manage the complexity of our system by dividing an application into the model, the view, and the controller. This way we can better manage the complexity of our recognition module. Also, one of the advantages of this architecture is that there are many frameworks and tools that support this architecture.

Furthermore, this architecture can be used to speed up development processes, as rapid and parallel development can be done. Lastly, it will simplify our testing by testing the components independently of each other.

**Other candidate architecture style:**

**Multi-Tier Architecture:**

As the name implies, a Multi-Tier Architecture is a software architecture characterized by the use of different software components, arranged in layers (tiers), to provide specific functionality. The multi-tiered architecture consists of three layers: the data management layer (database servers), the application layer (the business logic), and the client layer (the user interface).

The disadvantage of this architecture is that it has limited testability due to a lack of testing tools. It will also increase the complexity and the effort due to the complexity of communication between layers.

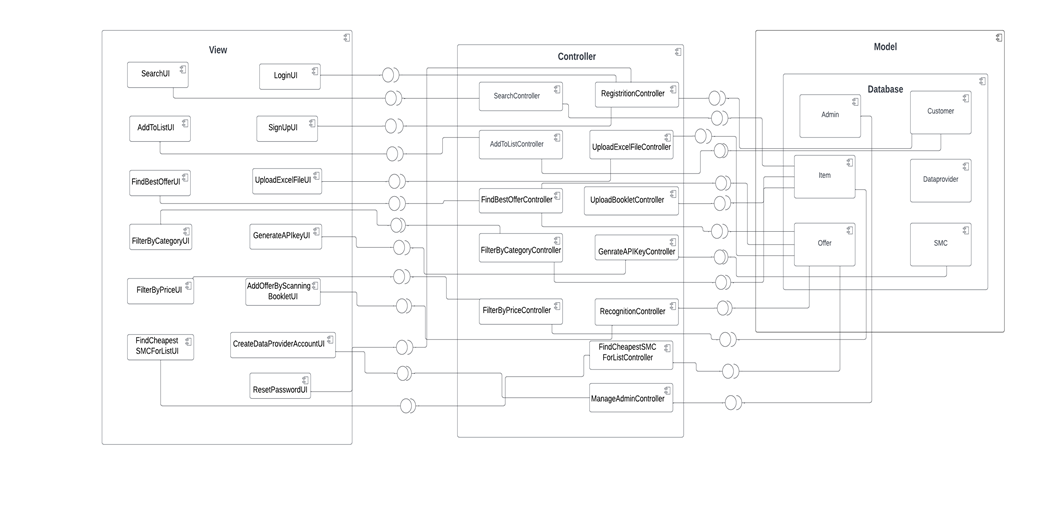
**Presentation-Abstraction-Control (PAC):**

The PAC architecture is similar to MVC, in the sense that the presentation module is like the view module of MVC. The abstraction module looks like the model module of MVC, and the control module is like the controller module of MVC, but the difference is in their flow of control and organization.

The disadvantages of this architecture style are that the system should be interactive and divided into multiple views rather than into multiple agents.

It's also hard to determine the right number of agents due to loose coupling and high levels of independence among agents. As well, separation of presentation and abstraction by control in each agent generates development complexity since communications between agents only happen between the controls of agents.

## 13.2 Component Diagram

The component diagram shows how components or software systems are wired together to form a larger system. They are used to visualize complex systems of any complexity.

*Figure 59 . Component diagram*

# 

# 

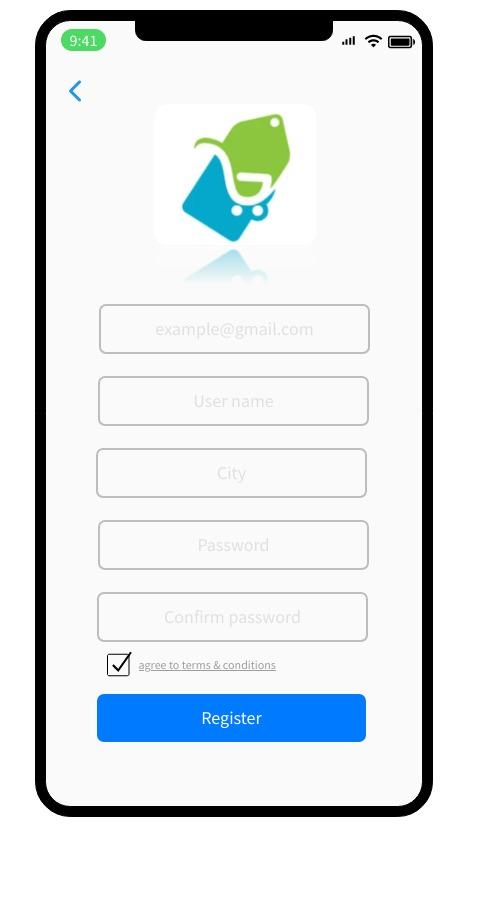
# User Interface Mock-up

A mock-up is a static representation of a product, showing users and stakeholders how it may look and be used.

This section shows some mock-ups of the system's main functionality. Ref 14.1 Customer’s mock-up screens:

### **14.1.1 Sign up screen**

### **14.1.2 Log in screen**

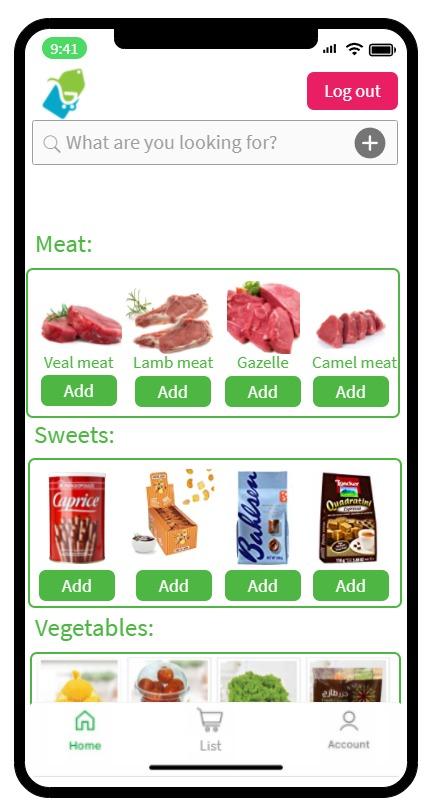
 Graphical user interface, application

Description automatically generated

*Figure 60 . Sign up screen customer Figure 61 . Log in screen customer*

### **14.1.3 Home page screen**

### **14.1.4 Find best offers screen**

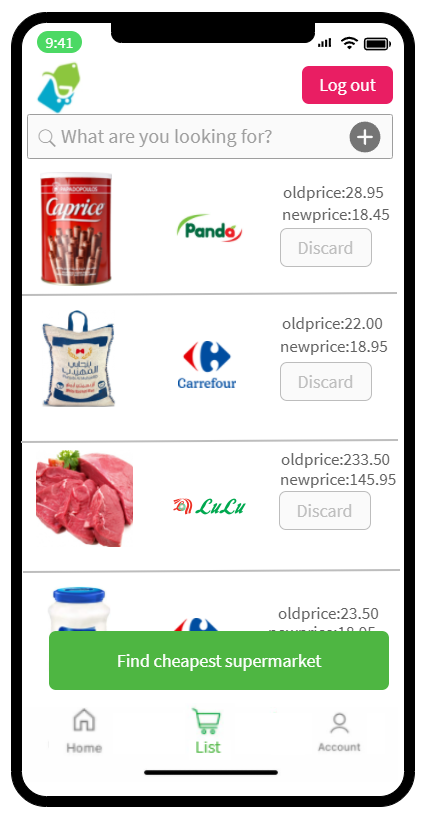
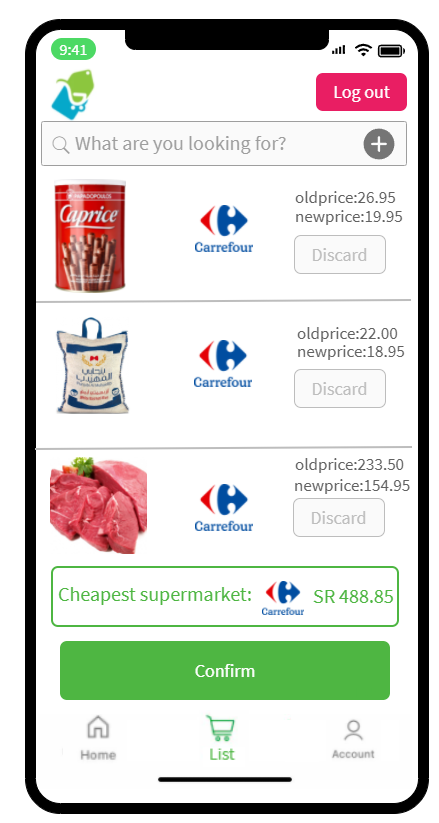
 Graphical user interface, application

Description automatically generated

*Figure 62 . Homepage screen customer Figure 63 . Find best offer screen customer*

### **14.1.5 View list screen**

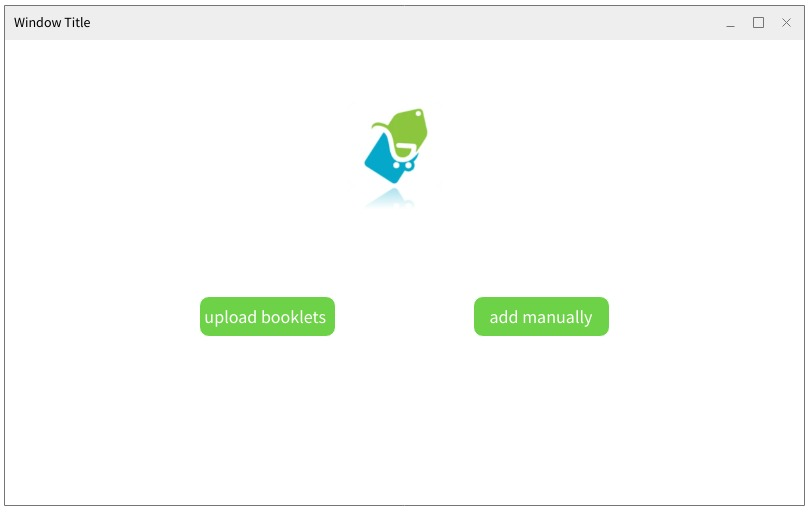
### **14.1.6 Find cheapest supermarket screen**

*Figure 64 . View list screen customer Figure 65 . Find cheapest supermarket screen customer*

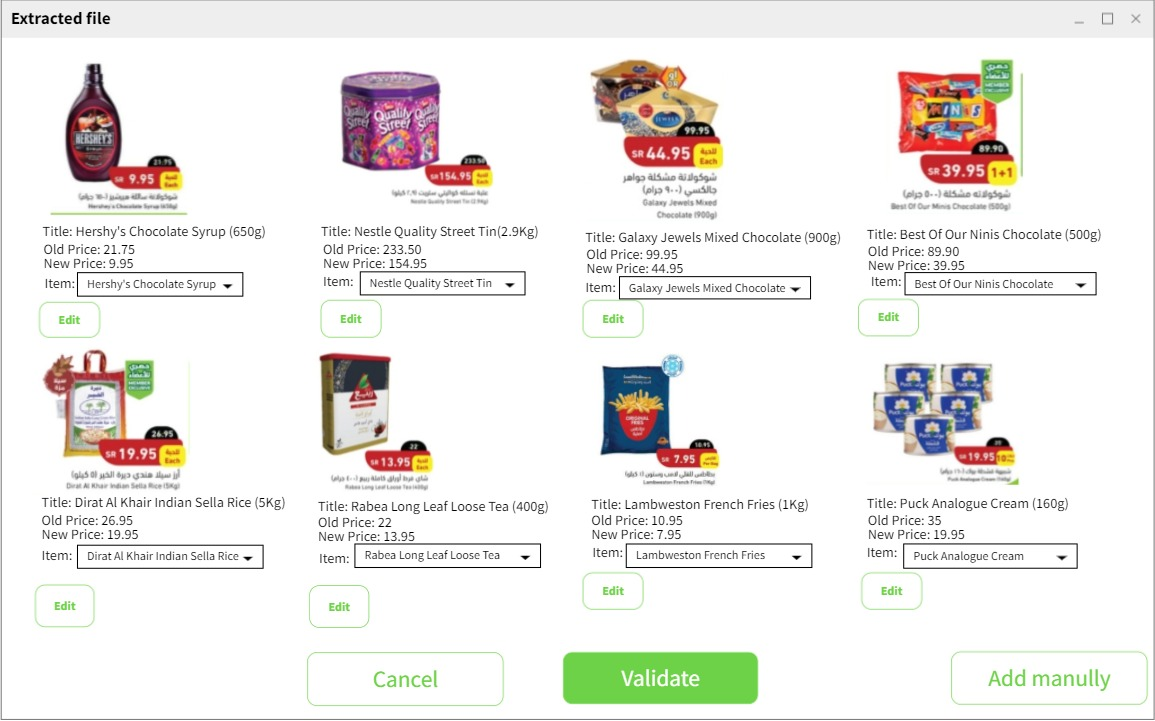
## 14.2 Data provider’s screens

### **14.2.1 Main screen**



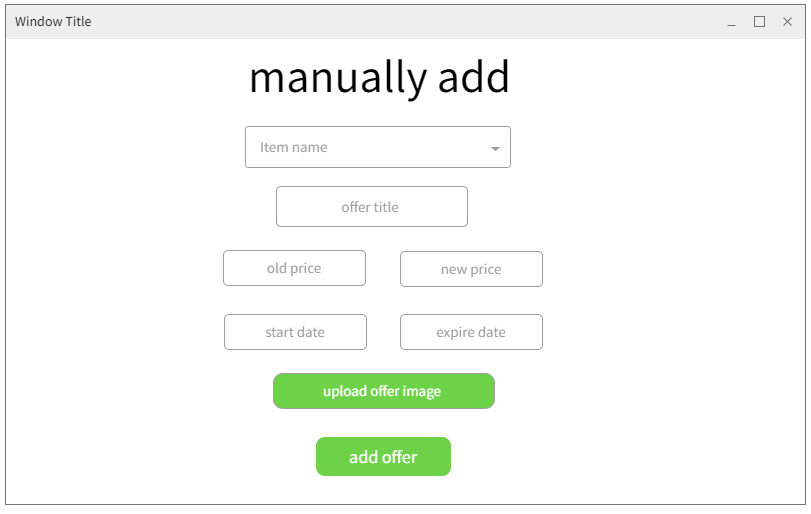
*Figure 66 . Data provider’s main screen*

### **14.2.2 Validate extracted items screen**



*Figure 67 . Validate extracted offers screen*

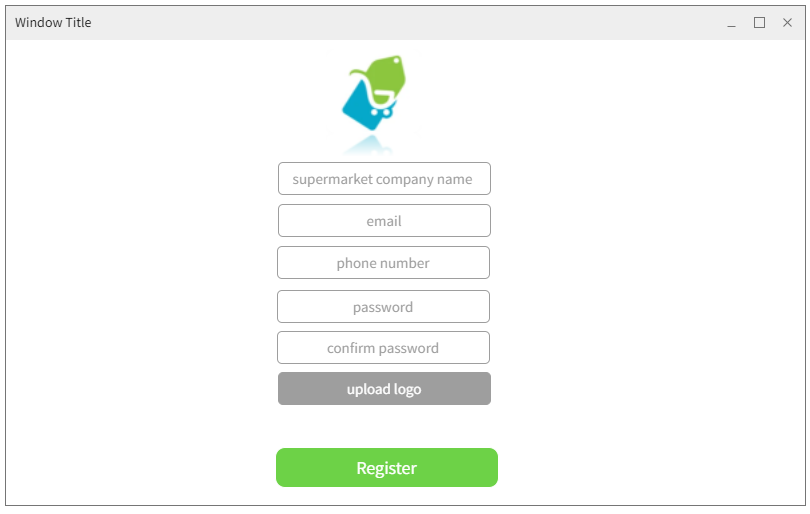
### **14.2. Add offers manually screen**



*Figure 68 . Add offers manually screen*

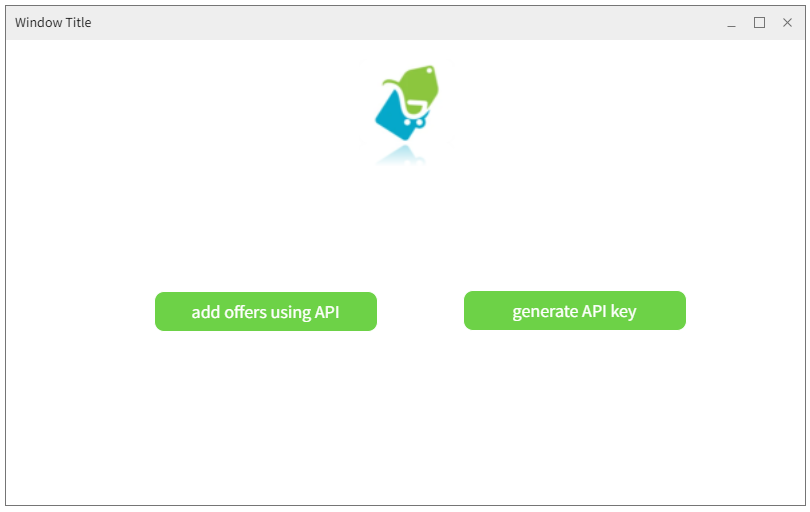
## 14.3 SMCs screens

### **14.3.1 Register screen**



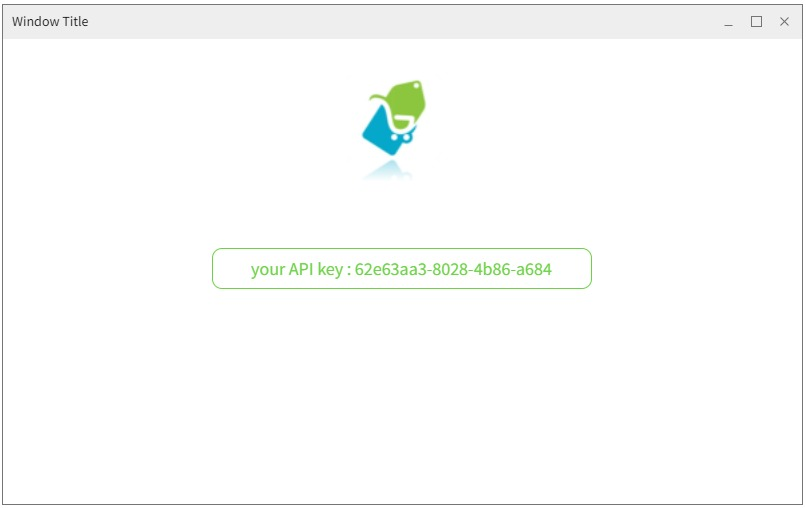
***Figure 69 . SMC registration screen***

### **14.3.2 Main screen**



*Figure 70 . SMC main screen*

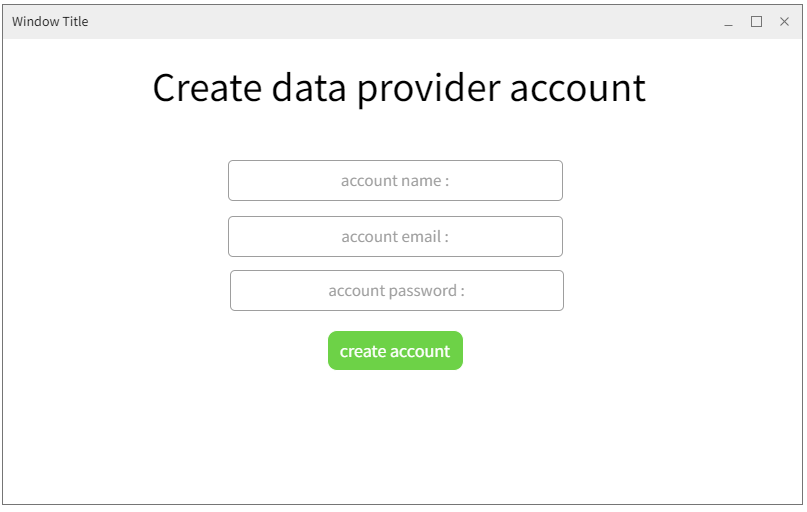
### **14.3.3 Generate API key screen**



*Figure 71 . Generate API screen*

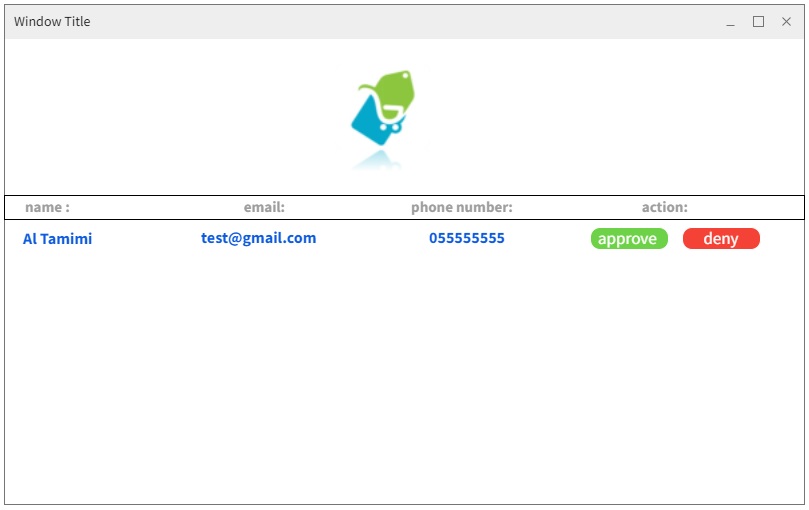
## 14.4 Administrators’ screens

### **14.4.1 create data provider account**



*Figure 72 . Create data provider account screen*

### **14.4.2 Approve SMC’c accounts**



*Figure 73 . Approve SMC’s account screen*

15. Database Schema

A database schema defines how data is organized within a relational database; this is inclusive of logical constraints such as, table names, fields, data types, and the relationships between these entities. [10] Schemas commonly use visual representations to communicate the architecture of the database. Figure illustrates database schema.

Diagram

Description automatically generatedRef <https://www.ibm.com/cloud/learn/database-schema>

*Figure 74 . Database schema*

# 

# 16.Algorithms

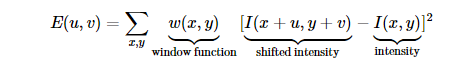
In mathematics and computer science, an algorithm is a finite sequence of rigorous well-defined instructions, typically used to solve a class of specific problems or to perform a computation. [11]

In software an algorithm is a set of instructions for solving a problem or accomplishing a task. One common example of an algorithm is a recipe, which consists of specific instructions for preparing a dish or meal. Every computerized device uses algorithms to perform its functions in the form of hardware- or software-based routines.

One of the main system features is extracting the offers from a booklet file by using deep learning algorithms. This feature includes an object boundry recognition algorithm.

For object recognition we use an algorithm that identifies features. Features are parts or patterns of an object in an image that help to identify it. For example a square has 4 corners and 4 edges, they can be called features of the square, and they help us humans identify it’s a square. Features include properties like corners, edges, regions of interest points, ridges, etc…

Corners are one of the features of an object and are regions in the image with large variation in intensity in all the directions. One early attempt to find these corners was done by Chris Harris & Mike Stephens in their paper A Combined Corner and Edge Detector in 1988, so now it is called the Harris Corner Detector. He took this simple idea to a mathematical form. It basically finds the difference in intensity for a displacement of (u,v) in all directions. This is expressed as below:



In coding terms Harris Corner Detector in OpenCV [12], OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy to utilize and modify the code. [13]

OpenCV has the function cv.cornerHarris(). Its arguments are:

img - Input image. It should be grayscale and float32 type.

blockSize - It is the size of neighbourhood considered for corner detection

ksize - Aperture parameter of the Sobel derivative used.

k - Harris detector free parameter in the equation.

See the example below:

**import numpy as np**

**import cv2 as cv**

**declare filename**

**filename = 'chessboard.png'**

**img =** [**cv.imread**](https://docs.opencv.org/3.4/d4/da8/group__imgcodecs.html#ga288b8b3da0892bd651fce07b3bbd3a56)**(filename)**

**gray =** [**cv.cvtColor**](https://docs.opencv.org/3.4/d8/d01/group__imgproc__color__conversions.html#ga397ae87e1288a81d2363b61574eb8cab)**(img,cv.COLOR\_BGR2GRAY)**

**gray = np.float32(gray)**

**dst =** [**cv.cornerHarris**](https://docs.opencv.org/3.4/dd/d1a/group__imgproc__feature.html#gac1fc3598018010880e370e2f709b4345)**(gray,2,3,0.04)**

**#result is dilated for marking the corners, not important**

**dst =** [**cv.dilate**](https://docs.opencv.org/3.4/d4/d86/group__imgproc__filter.html#ga4ff0f3318642c4f469d0e11f242f3b6c)**(dst,None)**

**# Threshold for an optimal value, it may vary depending on the image.**

**img[dst GRETAR THAN 0.01\*dst.max()]=[0,0,255]**

[**cv.imshow**](https://docs.opencv.org/3.4/df/d24/group__highgui__opengl.html#gaae7e90aa3415c68dba22a5ff2cefc25d)**('dst',img)**

**IF** [**cv.waitKey**](https://docs.opencv.org/3.4/d7/dfc/group__highgui.html#ga5628525ad33f52eab17feebcfba38bd7)**(0) AND 0xff IS EQUAL TO 27:**

[**cv.destroyAllWindows**](https://docs.opencv.org/3.4/d7/dfc/group__highgui.html#ga6b7fc1c1a8960438156912027b38f481)**()**

Deep learning techniques for feature extraction are more robust to scale, occlusion, deformation, rotation, etc and have pushed the limits of what was possible. As a result, this algorithm will be extremely useful for our system to use in order to recognize offer border images and automatically extract them.

# 17. Expected Deployment

**Global Overview**

These are the relics displayed in our system architecture diagram with a straightforward depiction, and each actor has his own system:

1. SMC: What the Supermarket Company will use to get to the system.

2. Data provider: What the data provider will use to get to the system.

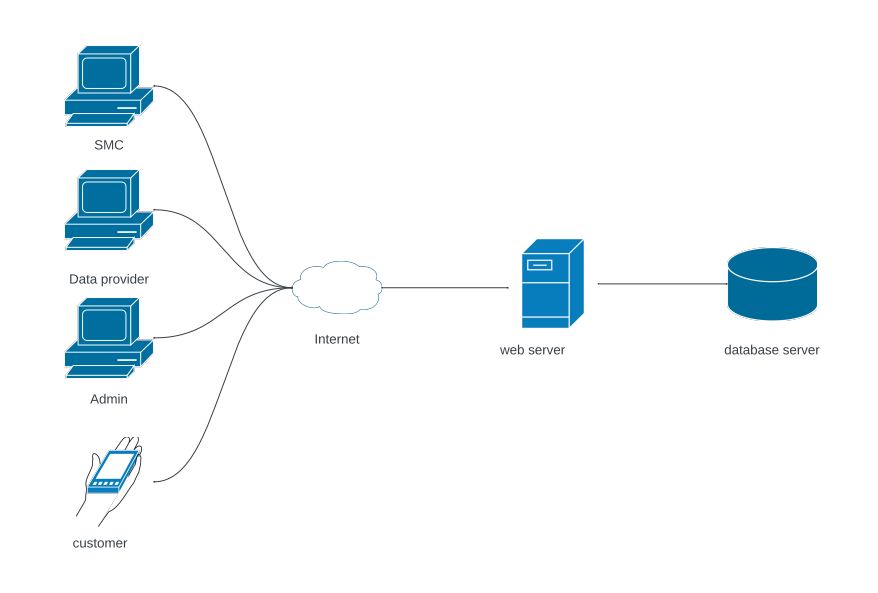
3. Admin: What the admin will use to get to the system.

4. Customer: What the customer will use to get to the system.

5. Internet: The connector between the user and the server.

6. Web server: Used to serve pages and static substance to the user through https demands.

7. Database server: A server that will speak with the database, it will be utilized to send/recover user-related information to/from the system.



*Figure 75 . System deployment diagram*

# 18. Test Scenario

A Test Scenario is defined as any functionality that can be tested. Scenario Testing in software testing is a method in which actual scenarios are used for testing the software application instead of test cases. [14] The purpose of scenario testing is to test end to end scenarios for a specific complex problem of the software. Scenarios help in an easier way to test and evaluate end to end complicated problems.

In this section we will use a variety of testing techniques as follows: Test Case ID, Test scenario, inputs, Expected Value, Prerequisites.

**18.1 Supermarket company**

* log in
* added offers by uploading an excel file
* generate API key

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test Scenario | Inputs | Expected Value | Prerequisites |
| 1 | Log in into the system | Valid username, password | Successful log in | -approve the account from the admin  -valid account |
| 2 | added offers by uploading an excel file | excel file | Offers are added | -user must be logged in |
| 3 | generate API key | - | API key | -user must be logged in |

Table 17 . SMC test scenario

**18.2 Data provider**

* log in
* add offers by scanning booklets
* validate

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test Scenario | inputs | Expected Value | Prerequisites |
| 4 | Log in into the system | Valid username, password | Successful log in | -valid account |
| 5 | Add offers by scanning booklets | booklet file | Offers are added | -user must be logged in |
| 6 | Validate the pdf booklets | - | New offers stores | -user must be logged in  -the booklets files scanned |

Table 18 . Data provider test scenario

**18.3 Admin**

* log in
* create data provider account

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test Scenario | inputs | Expected Value | Prerequisites |
| 7 | Log in into the system | Valid username, password | Successful log in | -valid account |
| 8 | Create a data provider account | Fill in the data provider username and password | New date provider account created | -user must be logged in |

Table 19 . Admin test scenario

**18.4 Customer**

* sign up
* log in
* find Best Offers
* Search item by name
* Filter by price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test Scenario | inputs | Expected Value | Prerequisites |
| 9 | Sign up into the system | Username  Password  email | Successful sign up | -valid email  -user name not already taken |
| 10 | Log in into the system | Valid username, password | Successful log in | -valid account |
| 11 | Find Best Offers | - | The cheapest offer from the supermarket is shown | -user must be logged in |
| 12 | Search item by name | Item name | Items with smaller name | -user must be logged in |
| 13 | filter by price | Price range | Items between range | -user must be logged in |

Table 20 . Customer test scenario

# 19. Project Status

In the beginning, we didn't have an unmistakable thought regarding the project yet, but Dr.Achraf made everything clear to us.

What's more, we made the Gantt graph and began chipping away at the Schedule until the midterms come and we needed to dial back the work and spotlight on our tests, after we finished the tests, we made more exertion into the task to get back on the timetable.

|  |  |  |
| --- | --- | --- |
| Issue | Description | How we solve it |
| Communication | It is so difficult to set a time to meet and everyone is not busy at that time. | We set the meeting on the weekend. |
| User Interface Mockup | Trying to find out the best interface that fits with all our functions. | We made a lot of changes to our mockups, until the last interface mockups we agreed on it. |
| Design Class | Attempting to cover all prerequisites, furthermore, figuring out how the system will work. | Dr.achraf helped us and gave us imagination how the system would be. |
| Reference | Attempting to know all the references that the group has profited from them. | We did a search for each reference we use |

Table 21 . Project status

# 20. Conclusion

Offerize is a cross-platform mobile application that allows supermarket customers to get all the offers from supermarkets without having to look through all booklets that the supermarket publishes. In terms of technology, the backend of the system will scan the offer booklet/brochures and use machine learning to automatically extract the item images and recognize text indicating the item name and the old/new price outlined in the booklet/brochure. Afterward, the extracted information is stored in a database so that it can be accessed by users of the mobile application.

In this report, we have presented a project plan that shows the steps we will take each week with the risks we could face, and we did a domain analysis to get a better idea of our competitors. Furthermore, we defined the system's functionality, and we illustrated many of those functionalities in order to get a better understanding of how the system operates. Furthermore, we have shown different components of our system and how they will be deployed.

In next semester, the team members plan to complete working on this project by developing and testing the functionalities. By the end of the next semester, the application will be ready to be installed and experimented. We hope that the second part of this project will be an enjoyable experience for all of us, we are looking forward to this second part as a way of extending our knowledge on different aspects of development.

# 21. Reference

[1] Neighbors, J.M. [*Software Construction using Components*](http://www.bayfronttechnologies.com/l02draco.htm#diss80). Technical Report 160, Department of Information and Computer Sciences, University of California, Irvine, 1980.

[2] Carrefour. [Online]. Available from: <https://www.carrefourksa.com/mafsau/ar/> [Accessed Feb. 4,

2022].

[3] Danube. [Online]. Available from:<https://danube.sa/> [Accessed Feb. 4,

2022].

[4] Flipp. [Online]. Available from:<https://flipp.com/home> [Accessed Feb. 4,

2022].

[5] Basket. [Online]. Available from:<https://basket.com/> [Accessed Feb. 4,

2022].

[6] Stebbing, L. (1993). Quality Assurance: The Route to Efficiency and Competitiveness (3rd ed.). Prentice Hall. p. 300. ISBN 978-0-13-334559-9.

[7] IEEE Computer Society (1990). "IEEE Standard Glossary of Software Engineering Terminology". IEEE Standard.

[8] Gemino, A., Parker, D.(2009) "Use case diagrams in support of use case modeling: Deriving understanding from the picture", Journal of Database Management, 20.

[9] Scott W. Ambler (2009) UML 2 Class Diagrams. Webdoc 2003-2009. [Online]. Available from: <http://www.agilemodeling.com/artifacts/classDiagram.htm> [Accessed Feb. 23,2022].

[10] Ashdown, Lance; Kyte, Tom (February 2010). Oracle Database Concepts 11g Release 2 (11.2). et al. Oracle Corporation. [Online]. Available from: <https://web.archive.org/web/20100129054802/http://download.oracle.com/docs/cd/E11882_01/server.112/e10713/tablecls.htm#CNCPT111> [Accessed Feb. 29,2022].

[11] Vault, Math. The Definitive Glossary of Higher Mathematical Jargon - Algorithm. [Online]. Available from: <https://2u.pw/QwR9R> [Accessed Mar. 6, 2022].

[12] Harris Corner Detection. [Online] <https://docs.opencv.org/3.4/dc/d0d/tutorial_py_features_harris.html> [Accessed Mar. 7, 2022].

[13] OpenCV. [Online] <https://opencv.org/about/> [Accessed Mar. 7, 2022].

[14] Gopalaswamy, Srinivasan Desikan. Software Testing:Principles and Practice.

\* All project diagrams were designed using the Lucidchart tool.

\* Project plan was designed using the Teamgantt tool.

\* Mockups were designed using Wireframe tool.

# 

# 22. Appendices

## 22.1 Inspection Report

|  |  |
| --- | --- |
| **Date of meeting** | **Results** |
| **29/1/2022** | **Fixing some grammar mistakes in the domain analysis.** |
| **12/2/2022** | **Adding actor Admin to use case diagram.** |
| **12/2/2022** | **Fixing use cases description names.** |
| **12/2/2022** | **Redone browse all items use case description.** |
| **6/3/2022** | **Fixing some spelling mistakes in class diagrams.** |
| **6/3/2022** | **Synchronized methods and names between class diagram, interaction diagram and Analysis diagram.** |
| **14/3/2022** | **Removed city option from Customer Sign Up mock up.** |

Table 22 . Inspection report

## 22.2 Formal Review Reports

|  |  |
| --- | --- |
| **Date of meeting** | **Results** |
| **4/2/2022** | **Reanalysed the domain** |
| **11/2/2022** | **Changed relationship type from extend to include between (view item list) and (find cheapest place for each item in list) use cases.** |
| **11/2/2022** | **Add the shortcuts definition in their first use.** |
| **17/2/2022** | **Removed any design details from use case description.** |
| **17/2/2022** | **Add more use cases description, interaction and analysis diagram.** |
| **3/3/2022** | **Added names to certain items in Customer mock-ups for differentiation.** |
| **3/3/2022** | **The actors (Customer, Data provider, SMC, Admin) in the database SCHEMA became inherited from an entity called user.** |

Table 23 . Formal review

## 22.3 use cases description

This section shows the rest of the use cases description of the system.

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Search item by name | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the customer finds an item that he/she is looking for by providing the item name. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **Customer** | **System** | |
| 1. The customer asks the system to search for an item.    3. The customer provides the item name. | 2. The system asks the customer for the item name.    4. The system selects all items that have a similar name to the provided name.    5. The system displays all selected items. | |
| **Alternative and exceptional flows:**  **3a. The customer provides an invalid or unavailable item name.**  3a2. The system displays an apology message. | | |
| **Post-conditions:**   * The items with a similar name to the provided name are displayed. | | |
|  |  |  |

Table 24 . Search item by name use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** Add to list | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** the use case describes how the customer adds an item to his/her list. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The customer must be logged in. | | |
| **customer** | **System** | |
| 1- The customer asks the system to add an item to his/her list. | 2- The system adds the item to the customer’s list.  3- The system tells the customer that the item has been added to the list. | |
| **Alternative and exceptional flows:**  None | | |
| **Post-conditions:**   * The selected item is added to the customer’s list. | | |

Table 25 . Add to list use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name: Customer Sign up** | | |
| **Primary actor:** Customer | | **Other actors:** None |
| **Description:** This use case describes how the Customer signs up to the system. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **Customer** | **System** | |
| 1.The Customer asks to sign up.    3. The Customer provides their information. | 2. The system asks the Customer for his information.    4. The system validates Customer information.  5. The system creates a record of the Customer in the database. | |
| **Alternative and exceptional flows:**  **4a. The Customer entered invalid/incomplete information.**  4a1. The system displays an error message.  4a2. The system returns to step two. | | |
| **Post-conditions:**   * The Customer is signed up to the system. | | |

Table 26 . Customer sign up use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:**{SCAD}log in | | |
| **Primary actor:** {SCAD} | | **Other actors:** None |
| **Description:** This use case describes how the {SCAD} logs in to the system. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * 1- The {SCAD}must be signed up to the system. | | |
| **customer** | **System** | |
| 1. The {SCAD} asks the system to login.  3. The {SCAD} provides their username and password. | 2.- The system asks for username and password.    4. The system validates the provided information.  5. The system logs in the {SCAD}.  6. The system redirects the {SCAD}to {SCAD}homepage. | |
| **Alternative and exceptional flows:**  **3a. The information entered is invalid or the {SCAD}doesn’t exist (or account is not Activated in case of SMC).**  3a1. The system displays an error message.  3a2.The system returns to step 2. | | |
| **Post-conditions:**   * The {SCAD} is logged in to the system. | | |

Table 27 . {SCAD} log in use case descrpition

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** {SCAD} Log out | | |
| **Primary actor:** {SCAD} | | **Other actors:** None |
| **Description:** This use case describes how the {SCAD} logs out from the system. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * The {SCAD} must be logged in. | | |
| **customer** | **System** | |
| 1. The {SCAD} asks the system to log out.  3. The {SCAD}confirms. | 2.- The system asks for confirmation.    4. The system logs out the {SCAD}.  5.The system redirects the {SCAD}to the log in page. | |
| **Alternative and exceptional flows:**  **3a. The user cancels the action.**  3a1. Exit the use case. | | |
| **Post-conditions:**   * The {SCAD} is logged out from the system and login page is displayed. | | |

Table 28 . {SCAD} Log out use case description

|  |  |  |
| --- | --- | --- |
| **Use Case Description** | | |
| **System**: Offerize | | |
| **Use Case name:** {SCAD} recover forgotten password | | |
| **Primary actor:** {SCAD} | | **Other actors:** None |
| **Description:** This use case describes how a Signed-Up {SCAD} actor recover his/her password after forgetting it. | | |
| **Relationships**   1. **Includes:** None 2. **Extends:** None | | |
| **Pre-conditions:**   * None | | |
| **{SCAD}** | **System** | |
| 1. The {SCAD}asks the system to recover his/her password.  3. the {SCAD}provide the required information | 2.The system ask {SCAD} to provide registered email  4.the system validates if the email is registered.  5.The system sends the password to the {SCAD}email.  6. the system displays a message that tells the user to check his email for the password | |
| **Alternative and exceptional flows:**  **4a. the system doesn’t find an account attached to the provided email:**  4a1. The system displays an error message.  4a2. Back to primary step 2. | | |
| **Post-conditions:**   * None | | |

Table 29 . Recover forgotten password use case description

## 22.4 Interaction Diagrams

This section shows the rest of the use cases Interaction diagram of the system.

### Diagram Description automatically generated**22.4.1 Search item by name**

*Figure 76 . Search item by name interaction diagram*

### Diagram Description automatically generated**22.4.2 Add to list**

*Figure 77 . Add to list interaction diagram*

### **22.4.3 Customer sign up**

*Figure 78 . Add to list Interaction diagram*

### **22.4.4 {SCAD} login**

Diagram

Description automatically generated

*Figure 79 . login interaction diagram*

### **22.4.5 {SCAD} log out**

*Figure 80 . log out interaction diagram*

### **22.4.6 {SCAD} Recover forgotten password**

*Figure 81 . Recover forgotten password interaction diagram*

## 22.5 Class diagrams

This section shows the rest of the use cases class diagram of the system.

### Diagram Description automatically generated**22.5.1 Search item by name**

*Figure 82 . Search item by name class diagram*

### Diagram Description automatically generated with medium confidence**22.5.2 Add to list**

*Figure 83 . Add to list class diagram*

### Diagram Description automatically generated with low confidence**22.5.3 Customer Sign up**

*Figure 84 . Customer sign up class diagram*

### Diagram Description automatically generated**22.5.4 {SCAD} login**

*Figure 85 . Login class diagram*

### Diagram Description automatically generated**22.5.5 {SCAD} recover forgotten password**

*Figure 86 . Recover forgotten password class diagram*