FINAL REPORT

Friendly FOODS

CSC343 - System Analysis and Design

Group #1:

Shaden AlKhammash 441201139

Fahda Alqahtani 439201102

1 Nov 2022

CONTENTS

GLOSSARY OF TERMS

Functional requirements	Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. [1]
Stakeholders	A stakeholder is either an individual, group or organization who is impacted by the outcome of a project. They have an interest in the success of the project. [1]
A software requirements specification (SRS)	a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfil all stakeholders (business, users) needs. [1]
Non-functional requirements	Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc. [1]
Use case	A task that an actor needs to perform with the help of the system. [1]
Sequence Diagrams	Are part of the UML and are used to model the interactions between the actors and the objects within a system and between the objects themselves.it is also emphasis on time ordering.[1]
Class Diagram	The class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that depicts a system's structure by displaying the system's classes, attributes, operations\methods, and the relationships among objects. [1]
Structure diagram	A conceptual modeling tool that makes up a System such as a database or an application. [1]
State diagram	Show how the system reacts to external and internal events. [1]
Unit testing	a type of software testing where specific units or components of the software are tested. The purpose is to confirm that each unit of the software code performs as expected.[1]
Integration testing	a level of software testing where specific units/components are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units [1].
Acceptance testing	Formal testing with respect to user needs, requirements, and business processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers, or other authorized entity to determine whether to accept the system [1].
A software design document (SDD)	a written report of a software product describing its overall architecture. are usually written by software designers or project managers. It helps to guarantee the design specs of the software are understood and it is clear to all.[1].

CUSTOMER STATEMENT REQUIREMENTS

PROBLEM STATEMENT:

Now a day's we see more and more new restaurants opening but to the people with food allergies they still have limited options for the reason that most restaurants don't put them into consideration by providing some type of notation of what could trigger their allergies and what's not. FriendlyFoods will give allergic people more options.

Another issue people with allergies face is that most restaurants don't list detailed information on the menu, or listing the main ingredients only. the main ingredients aren't enough because people could be allergic of the smaller amount ingredient that most of the time don't get listed.

It could be inconvenient to see a list options without them being in a map. To make it easier and more beneficial, the listing and the mapping should be together in the same application.

Most often people with allergies feel that their opinions don't matter and that's caused by food businesses disregarding them. that's why leaving reviews and rating should be a part of FriendlyFoods.

Peers' reviews are essential to bring the sense of community and a lot of people rely on them to make decisions. it's also important for people with allergies to be able to suggest a place to other allergic individuals.

SYSTEM REQUIREMENTS

ENUMERATED FUNCTIONAL REQUIREMENTS:

REQ-ID	DESCRIPTION	PW
REQ-1	A user shall be able to log in/register.	5
REQ-2	A user shall be able to view food allergens.	5
REQ-3	A user shall be able to select food allergens.	5
REQ-4	The system shall display restaurants that offer user's allergens-free foods.	5
REQ-5	A user shall be able to select a restaurant.	5

REQ-6	The system shall display the locations of the selected restaurant.	5
REQ-7	A user shall be able to rate.	4
REQ-8	A user shall be able to leave a review.	4
REQ-9	A user shall be able to view reviews.	4
REQ-10	the system shall display a rating for each added place.	4
REQ-11	Each user shall register with a unique email.	5
REQ-12	A user shall add a menu with details to a place	5
REQ-13	The system shall display the menu with detailed ingredients.	5
REQ-14	The system shall save all the users registrations data to a database.	5
REQ-15	The system shall use google maps as an external software component to get the location of each place.	5
REQ-16	The system shall allow users to edit their profiles as desired easily.	5
REQ-17	The system shall be able to send the account's password to the user's email incase if users forgot their passwords.	4
REQ-18	The user shall be able to add a restaurant.	5

ENUMERATED NONFUNCTIONAL REQUIREMENTS:

REQ-19	The system should be user-friendly and easy to use.	4
REQ-20	The system's response time shall not exceed 5 seconds.	5
REQ-21	When a failure occurs, the system shall return to its default state.	3
REQ-22	The system shall be secure.	4
REQ-23	The system shall delete an account automatically after being dormant for more than 1 year.	3

FUNCTIONAL REQUIREMENTS

STAKEHOLDERS:

Restaurateur: a person who owns and runs a restaurant.

People with food allergies: individuals who are allergic to certain food ingredients.

ACTORS AND GOALS:

Actor	Type	Goal
Client	Initiating	The user of the app who use it to find suitable restaurant for their food allergens
Restaurateur	Initiating	Restaurateur is the user of the app who wants to promote for their allergy-free menu.
Database	Participating	The storage device that is used to store the information.
GoogleMapsAPI	Participating	An external component to display the location of a restaurant.

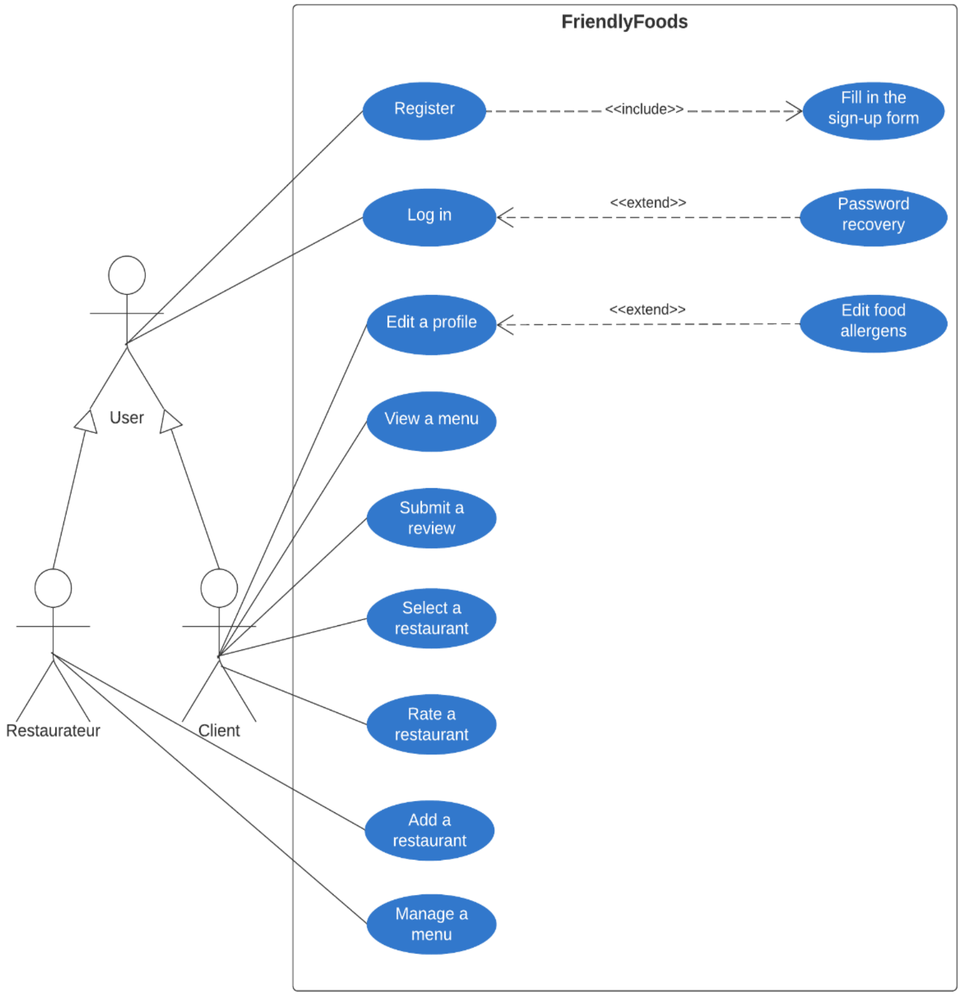
USE CASE CASUAL DESCRIPTION:

Use Case ID	**Name**	**Short Description**	**Corresponding REQ-ID**
UC1	Register	Allows the client\restaurateur to register a new account.	REQ-1, REQ-14
UC2	Log in	Allows the client\ restaurateur to log into their account.	REQ-1
UC3	Submit a review	Allows the client to write and submit a review about a restaurant.	REQ-8
UC4	Rate a restaurant	Allows the client to submit a rating for a specific restaurant.	REQ-7
UC5	Edit food allergens	Allows the client to select\deselect food allergens.	REQ-2, REQ-3
UC6	View a menu	Allows the client to view the menu for a restaurant.	REQ-13
UC7	Edit a profile	Allows the client to edit their profile.	REQ-16
UC8	Add a restaurant	Allows the restaurateur to add a restaurant.	REQ-18
UC9	Select a restaurant	Allows the client to select a restaurant and view its details.	REQ-5

UC10	Manage a Menu	Allows the restaurateur to add, delete, or edit a menu of their own restaurant.	REQ-12
UC11	Password recovery	An option that allows the client\restaurateur to recover their password in case of forgetting it.	REQ-17

INTERACTION DIAGRAMS

USE CASE DIAGRAM:



MADE WITH LUCIDCHART [2]

USE CASE FULLY DRESSED DESCRIPTION:

UC5<**Edit food allergens**>

Initiating Actor: Client.

Actor's Goal: select\deselect food allergens.

Participating Actor: Database.

Pre-Condition: The client must be logged in.

Post-Condition: The system shall save the changes made to the client's profile.

Flow of events for success scenario:

1. The client clicks on edit a profile.
2. The client select\deselect food allergens.
3. The client clicks on save.

UC9<**select a place**>

Initiating Actor: Client.

Actor's Goal: look up the menu and the location of the place.

Participating Actor: Database.

Pre-Condition: The client must be logged in.

Post-Condition: The system shall display the menu and the location with the rating.

Flow of events for success scenario:

1. The system display options of places.
2. The client clicks on one of the options.
3. The client views the menu and location with the rating.

UC1<**register**>

Initiating Actor: Client\Restaurateur.

Actor's Goal: sign up to the application.

Participating Actor: Database.

Pre-Condition: unique email.

Post-Condition: The system shall save the data to DB.

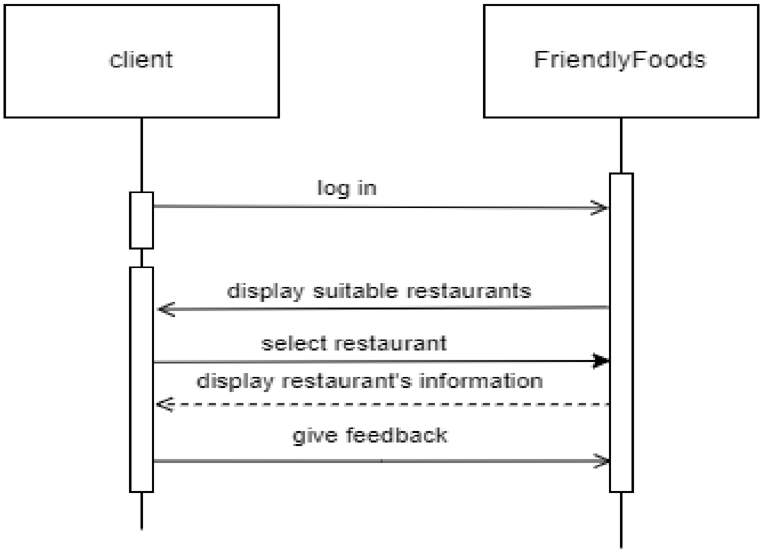
Flow of events for success scenario:

1. The user clicks on register.
2. The user enters unique email.
3. The user enters their username.
4. The user enters a password.
5. Update the DB.

Alternative scenario:

1. *The user entered an already existed email.*
2. *Error message will appear, the user should choose another username.*

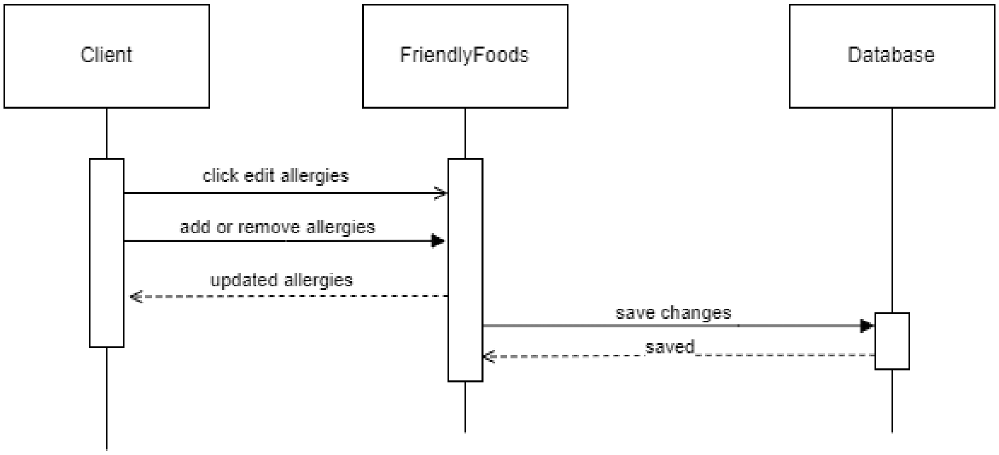
SYSTEM SEQUENCE:


client
FriendlyFoods
log in
display suitable restaurants
select restaurant
display restaurant's information
give feedback

Made with draw.io [3]

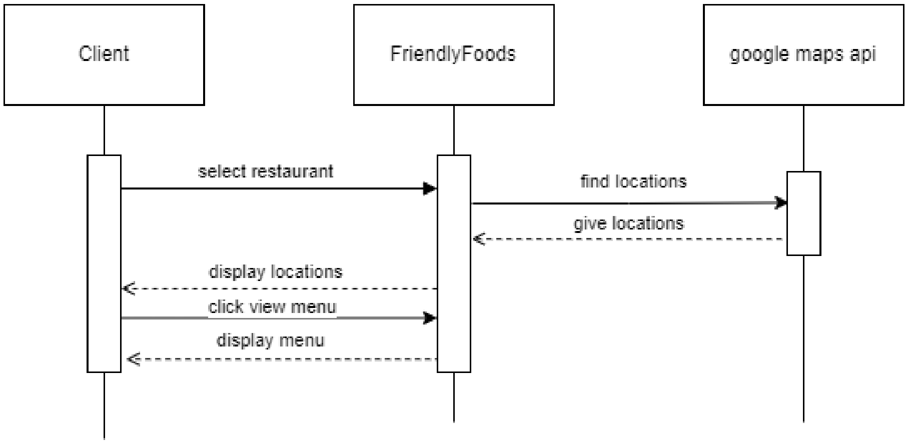
SEQUENCE DIAGRAM:

1.



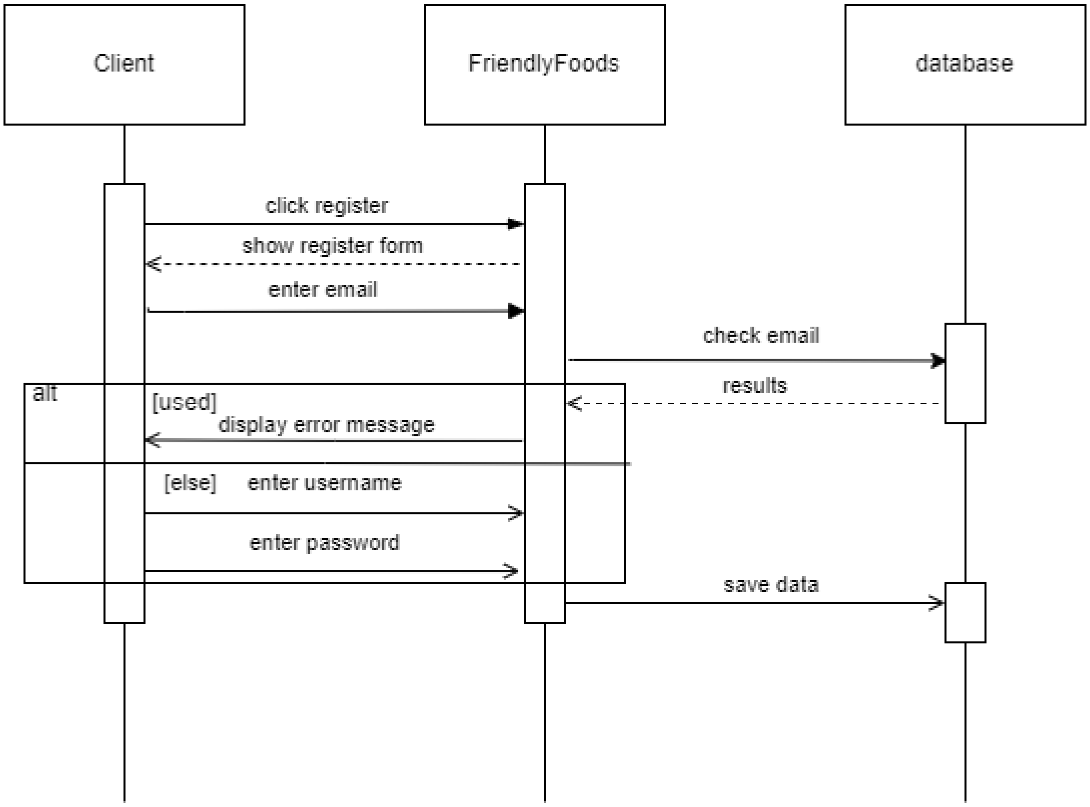
Made with draw.io

2.



Made with draw.io [3]

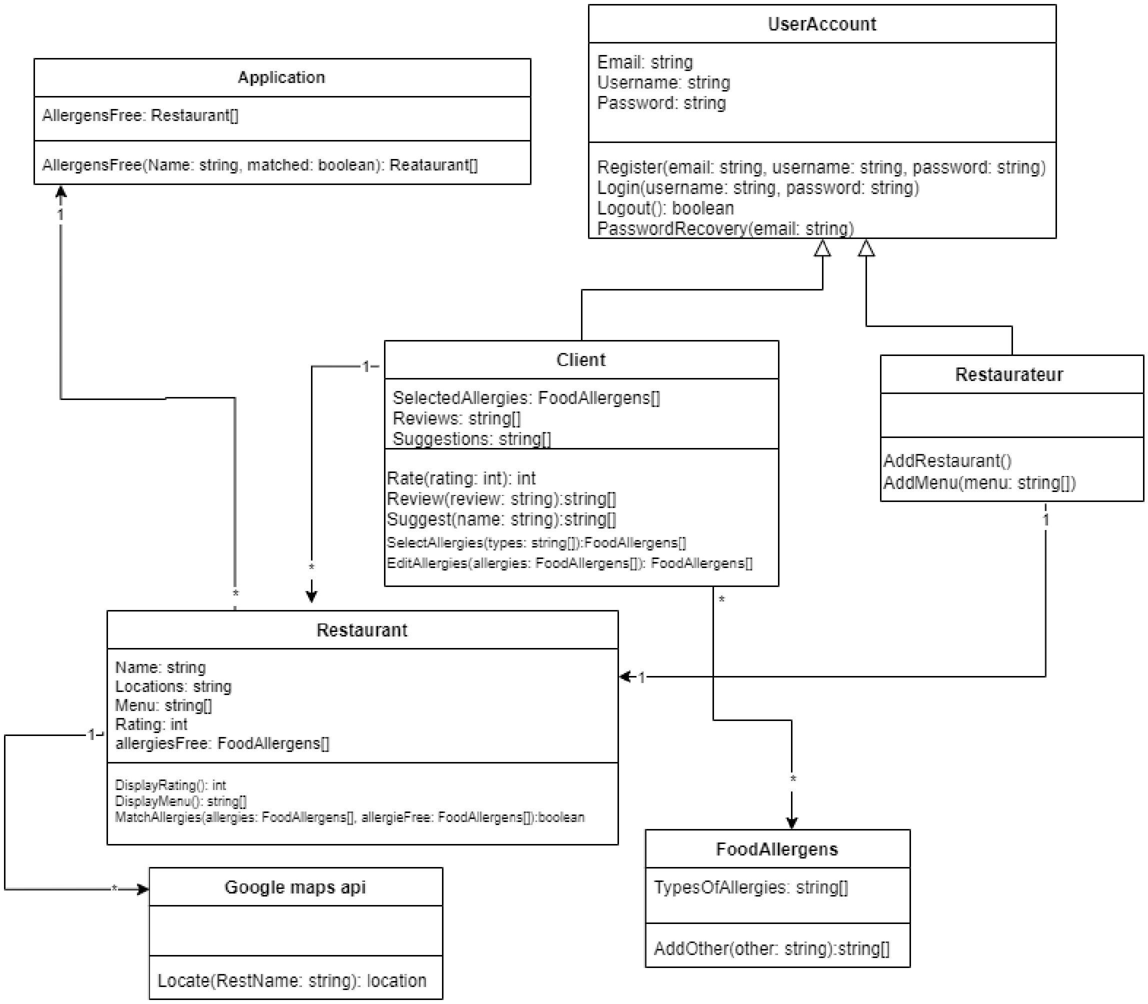
3.



Made with draw.io [3]

DOMAIN ANALYSIS

CLASS DIAGRAM:


UserAccount
Email: string
Username: string
Password: string
Register(email: string, username: string, password: string)
Login(username: string, password: string)
Logout(): boolean
PasswordRecovery(email: string)
Application
AllergensFree: Restaurant[]
AllergensFree(Name: string, matched: boolean): Reataurant[]
1
Client
SelectedAllergies: FoodAllergens[]
Reviews: string[]
Suggestions: string[]
Rate(rating: int): int
Review(review: string):string[]
Suggest(name: string):string[]
SelectAllergies(types: string[]):FoodAllergens[]
EditAllergies(allergies: FoodAllergens[]): FoodAllergens[]
1
*
*
Restaurateur
AddRestaurant()
AddMenu(menu: string[])
1
*
Restaurant
Name: string
Locations: string
Menu: string[]
Rating: int
allergiesFree: FoodAllergens[]
DisplayRating(): int
DisplayMenu(): string[]
MatchAllergies(allergies: FoodAllergens[], allergieFree: FoodAllergens[]):boolean
1
1
*
FoodAllergens
TypesOfAllergies: string[]
AddOther(other: string):string[]
*
Google maps api
Locate(RestName: string): location

Made with draw.io [3]

METHODS AND DESCRIPTIONS:

UserAccount:

Username: String
The user name.

Email: String
User's Email.

Password: String
User's password.

Login(Username: String, Password: String)
A method that receives the user's username and password, to get an access to their account.

Register(username: string, email: string, password: string)
A method that allows the user to create a new account.

Logout()
A method that allows the user to logout of their account.

PasswordRecovery(Email: String)
A method that receives the user's email, the main objective of this method is to help users re-establish access to their accounts by sending a new password to the user's email.

GoogleMapsAPI:
Locate(restName: String)
A method that receives the name of the restaurant and search for its locations using google maps.

FoodAllergens
TypesOfAllergies: string[]
Different types of food allergies.

AddOther(other: string)
A method to add an allergy that wasn’t on the list.

Restaurant
Name: String
Restaurant's name.

Locations: location[]
All of the restaurant’s locations.

Menu: string[]
The restaurant’s menu.

Rating: int
The rating for the restaurant that was left by the clients.

allergiesFree: FoodAllergens[]
the list of allergens that the restaurant doesn't have.
DisplayMenu()
A method to display the menu of the restaurant.

DisplayRating()
A method to display rating.

MatchAllergies(allergies: FoodAllergens[], allergiesFree: FoodAllergens[])
A method to match and check the client allergies with restaurant.

Client
SelectedAllergies: FoodAllergens[]
The client's allergies.

Review: string[]
Client's reviews.

Suggestion[]
Client's suggestions.

Rate(rating: int)
A method that allows the client to rate a restaurant.

Review(review: string)
A method that allows the client to review a restaurant.

Suggest(name: string)
A method that allow the client to submit a suggestion.

SelectedAllergies(types: string[])
A method to collect the client's allergy

EditAllergies(allergies: FoodAllergens[])
A method to edit the client's list of allergies.

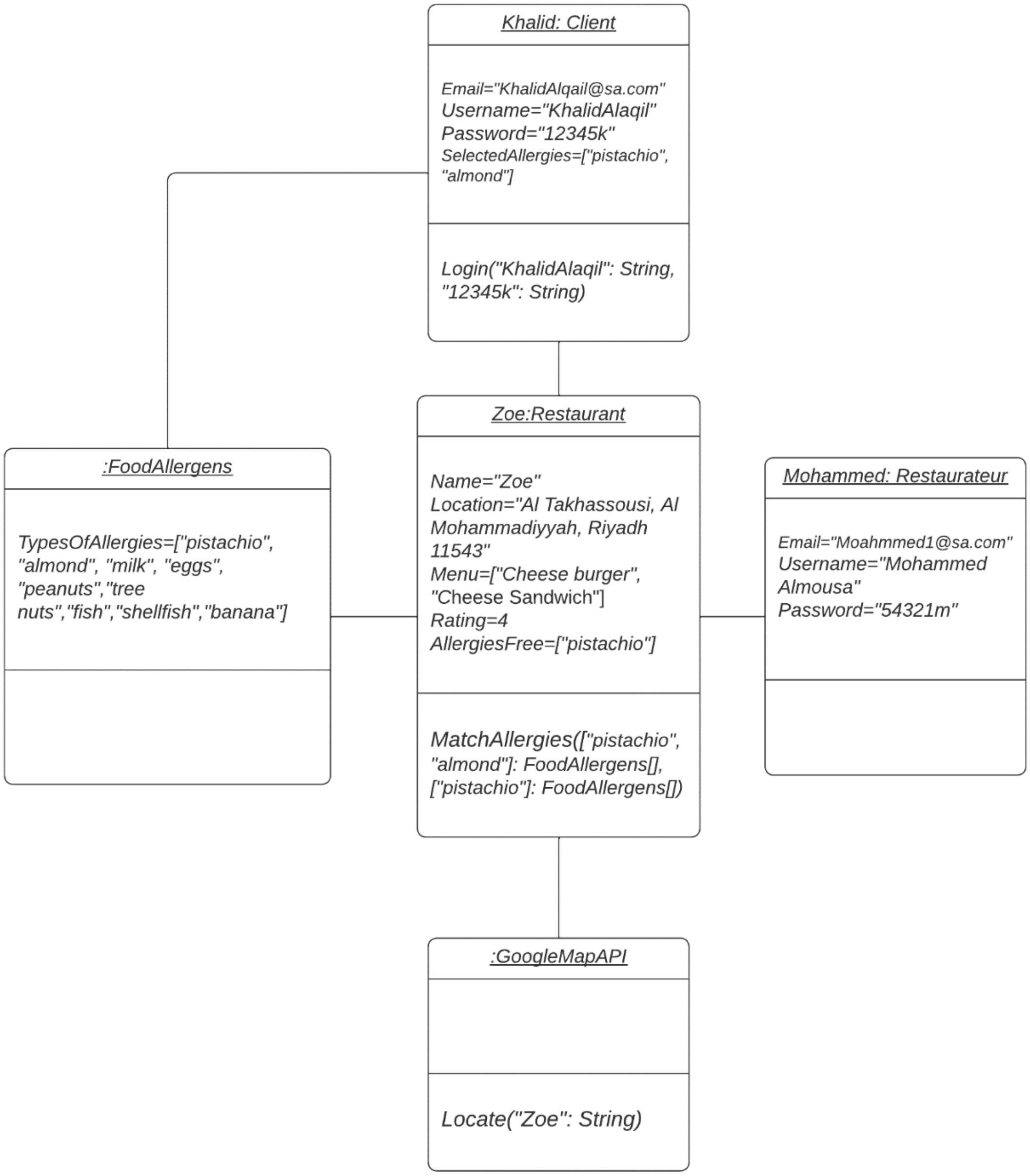
Restaurateur
AddRestaurant()
A method that allows the restaurateur to add a restaurant.

AddMenu(menu: string[])
A method that allows the user to add a menu for a restaurant.

Application
AllergensFree: Restaurant[]
List of restaurants that suit the client

AllergensFree(Name: string, matched: Boolean)
A method that collects the restaurants that suits the client.

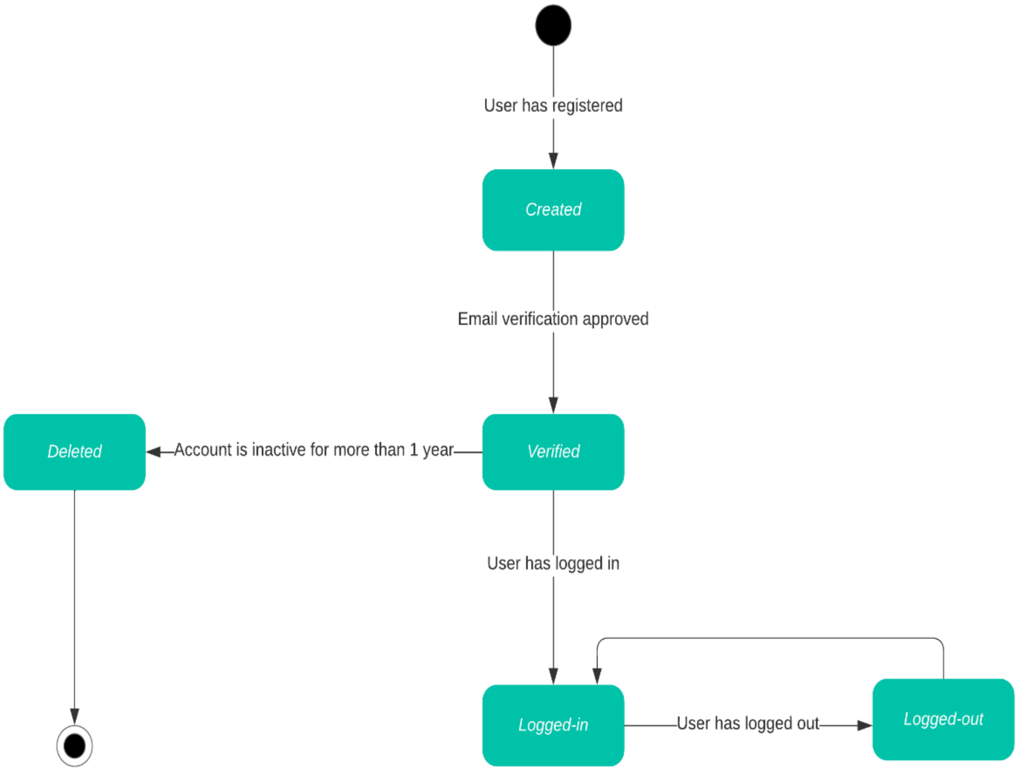
OBJECT DIAGRAM:


Khalid: Client
Email="KhalidAlqail@sa.com"
Username="KhalidAlaqil"
Password="12345k"
SelectedAllergies=["pistachio", "almond"]
Login("KhalidAlaqil": String, "12345k": String)
:FoodAllergens
TypesOfAllergies=["pistachio", "almond", "milk", "eggs", "peanuts","tree nuts","fish","shellfish","banana"]
Zoe:Restaurant
Name="Zoe"
Location="Al Takhassousi, Al Mohammadiyyah, Riyadh 11543"
Menu=["Cheese burger", "Cheese Sandwich"]
Rating=4
AllergiesFree=["pistachio"]
MatchAllergies(["pistachio", "almond"]: FoodAllergens[], ["pistachio"]: FoodAllergens[])
Mohammed: Restaurateur
Email="Moahmmed1@sa.com"
Username="Mohammed Almousa"
Password="54321m"
:GoogleMapAPI
Locate("Zoe": String)

Made with Lucidchart[2]

BEHAVIORAL DIAGRAMS

State Diagram of Account:



Made with Lucidchart[2]

SYSTEM ARCHITECTURE AND SYSTEM DESIGN

The System Organization:

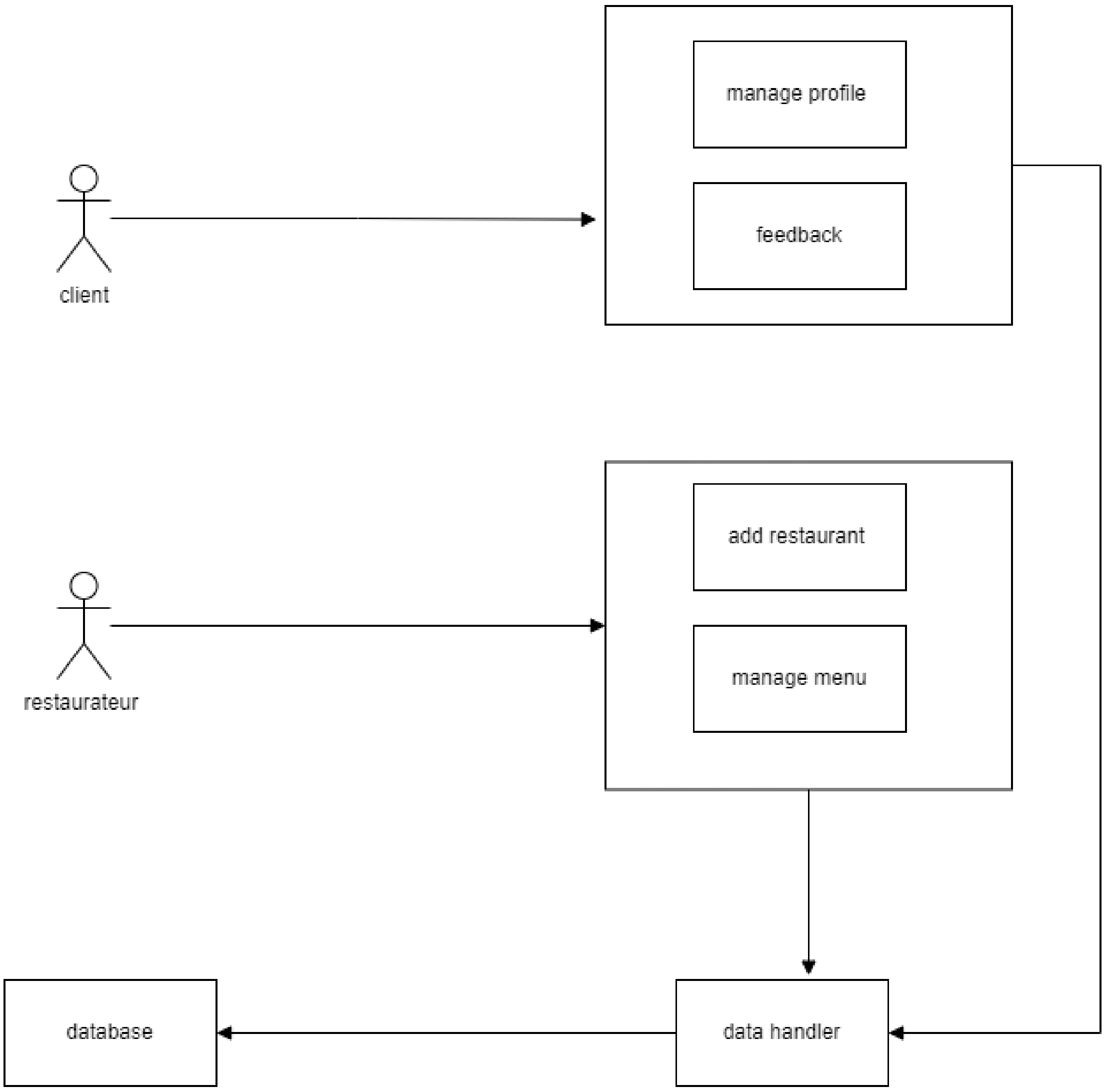
We chose **Client-server** architecture because our system needs permanent data storage to maintain the users and the restaurants information. Therefore, we require a server, and with that we need a client/server architecture. On the server side, we have the Database subsystem which contains the permanent data for our application. This server can handle the requests from the users and send or receive information as needed. The client side contains the data handler subsystem, which has access to the information in the Database. The system supports multiple users operating at the same time whenever they want.

We used **Object-Oriented model**; the object-oriented life cycle model considers objects as the basis of the software engineering process. The development team starts by observing and analyzing the system they intend to develop before defining the requirements. Once the process is over, they focus on identifying the objects of the system. And it's the same as what we did in developing our application.

CONTROL FLOW:

Our system is **event-driven** in that it waits for users' input and acts according to it. The system does not log anyone in until the initial event is driven by an external request. The user interacts with the system by first registering a new account. After logging into their account, the next step requires the user to enter the necessary data which includes food allergens for the client, and restaurant details for the restaurateur when adding a restaurant. This data gets stores into the database. Once the client updates their food allergens list, the system will display the restaurants according to the updated list. When the restaurateur wants to add a restaurant, they must enter the necessary data and after that the restaurant will be added to the application and it will be visible to the clients of the app.

IDENTIFYING THE SUBSYSTEMS:



Made with draw.io [3]

DESIGN OF TESTS

UNIT TESTING:

Test Case ID	Description	Test input	Expected output	Pass/Fail	Actual output	Status
TC1	Tests if the system allows the client /restaurateur to register a new account.	The User enters pre-existing email.	The system will display an error message "Email address already in use"	Pass if the user was unable to create the account and fail otherwise.	The system displayed an error message "Email address already in use"	Pass.
		The User enters a non-existent email.	User will register successfully.	Pass if the user was able to register a new account and fail otherwise.	User Registered successfully.	Pass.
TC2	Tests if the system allows the client /restaurateur to log into their account.	The User enters invalid login credentials which is email and password.	The system will display error message "Invalid email or password, please double check the information you provided and try again."	Pass if the user was unable to log into their account and fail otherwise.	The system displayed an error message "Invalid email or password, please double check the information you provided and try again."	Pass.
		The User enters valid login credentials which is email and password.	User will log into their account successfully.	Pass if the user was able to log into their account and fail otherwise.	User logged into their account successfully.	Pass.

TC3	Tests if the system Allows the client to write and submit a review about a restaurant.	The user writes and submits a review.	Review will be submitted successfully.	Pass if the user was able to submit a review and fail otherwise.	Review has been submitted successfully.	Pass.
TC4	Tests if the system Allows the client to submit a rating for a specific restaurant.	The client submits a rating for a restaurant.	Rating will be submitted successfully.	Pass if the client was able to submit a review and fail otherwise.	Rating has been submitted successfully.	Pass.
TC5	Tests if the system Allows the client to select\deselect food allergens.	The client select\deselect food allergens	The system will update the user's food allergens list.	Pass if the system updated the client's food allergens list and fail otherwise	The system updated the user's food allergens list	Pass.
TC6	Tests if the system Allows the client to view the menu for a restaurant.	The client clicks on View Menu of a restaurant.	The system will display the menu of the restaurant.	Pass if client was able to view the Menu of the restaurant and fail otherwise.	The system displayed the menu of the restaurant.	Pass.
TC7	Tests if the system Allows the client to edit their profile.	The client updates their profile information and clicks on save.	The system will update the client's profile.	Pass if the system updated the profile information and the client was able to see the changes made to their profile and fail otherwise.	The system updated the client's profile.	Pass.
TC8	Tests if the system Allows the restaurateur to add a restaurant.	The restaurateur clicks on add restaurant and enters restaurant information.	The system will add the new restaurant successfully.	Pass if the system added the restaurant successfully and fail otherwise.	The system has added the new restaurant successfully.	Pass.

TC9	Tests if the system Allows the client to select a restaurant and view its details.	The client clicks on a restaurant from the available free food allergens restaurants list.	The system will show the restaurant details.	Pass if the client was able to view the restaurant details and fail otherwise.	The system has showed the restaurant details.	Pass.
TC10	Tests if the system Allows the restaurateur to add, delete, or edit the menu of their restaurant.	The restaurateur clicks on manage menu.	The system will save the changes made to the menu.	Pass if the restaurateur was able to make changes to the menu of their restaurant and fail otherwise.	The system has saved the changes made to the menu.	Pass.
TC11	Tests if the system allows the client \restaurateur to recover their password in case of forgetting it.	The user enters non-existent email address.	The system will display an error message "An error occurred, please try again."	Pass if the system was unable to send the recovery instructions to the entered email and fail otherwise.	The system displayed an error message "An error occurred, please try again."	Pass.
		The user enters pre-existing email address.	The system will display "Email is on the way" and will send the recovery instructions to the entered email.	Pass if the system was able to send the recovery instructions to the entered email and fail otherwise.	The system displayed "Email is on the way" and sent the recovery instructions to the entered email.	Pass

INTEGRATION TESTING:

Test Case ID	Description	Test input	Expected output	Pass/Fail	Actual output	Status
TC12	Tests if the system allows the client to edit food allergens and edit profile information.	The client select\deselect the food allergens and then edit their profile information E.g., Username.	The system will save the changes made to the food allergens list and to the profile information.	Pass if the user was unable to create the account and fail otherwise.	The system has saved the changes made to the food allergens list and to the profile information.	Pass.
TC13	Tests if the client can select a restaurant and view its menu	The client selects a specific restaurant and click on view menu	The system will display the menu of the selected restaurant.	Pass if the client was able to view the menu of the selected restaurant and fail otherwise.	The system displayed the menu of the selected restaurant.	Pass.

ACCEPTANCE TESTING:

we asked four end-users to function our application as if they are clients or restaurateurs, after trying our application they've provided us with effective feedback.

UserName	UserType	User feedback
Abdulrahman	Client	It was a good experience, I recommend adding friendship lists and direct messaging features, which would allow clients to add each other as friends and communicate with each other through direct messages.
Sarah	Client	I suggest including a button that users may use to contact the app's developers about any problems they're facing with the app.
Mohammed	Restaurateur	The restaurant details are very limited, I recommend adding more details such as the type of food the restaurant offers E.g, Italian food etc.
Khalid	Restaurateur	Everything is fine but I recommend that the restaurateur gets notified when a client adds a review about their restaurant.

INTERFACE

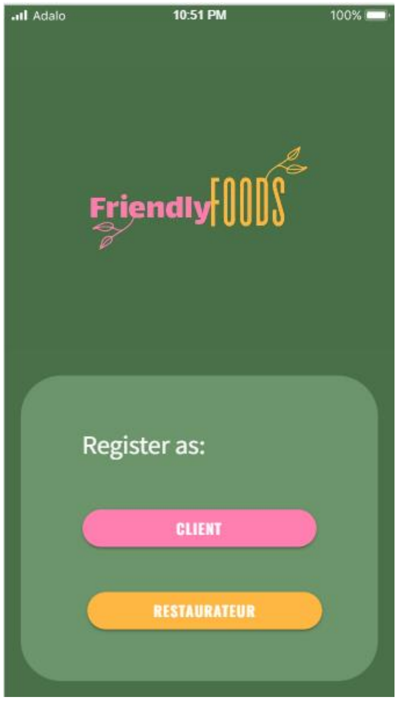


FIGURE 1 WHICH USER REGISTER

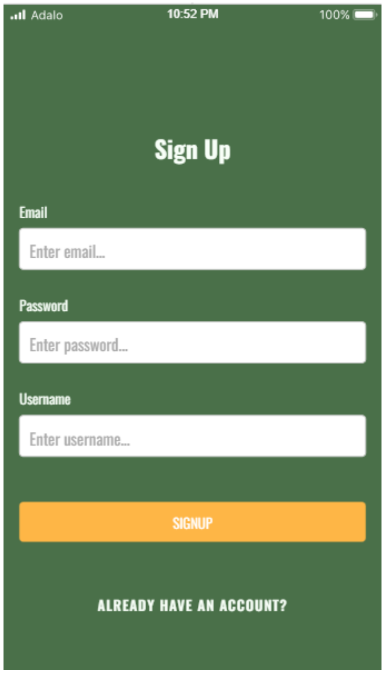


FIGURE 2 SIGN UP

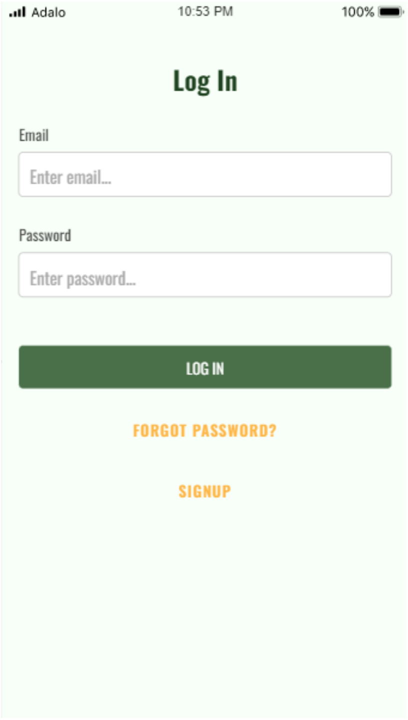


FIGURE 3 LOGIN IF ALREADY HAS ACOUNT

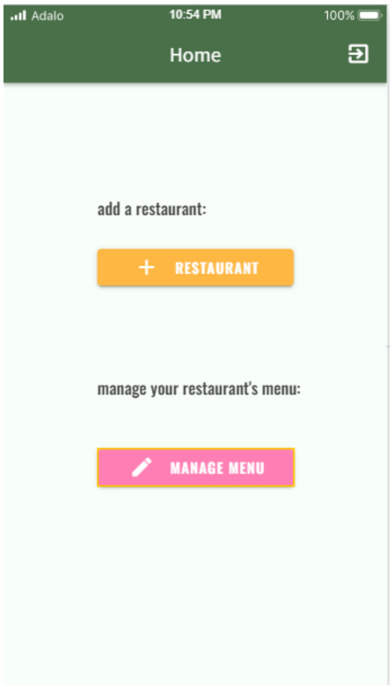


FIGURE 4 HOMEPAGE FOR RESTAURANTUER

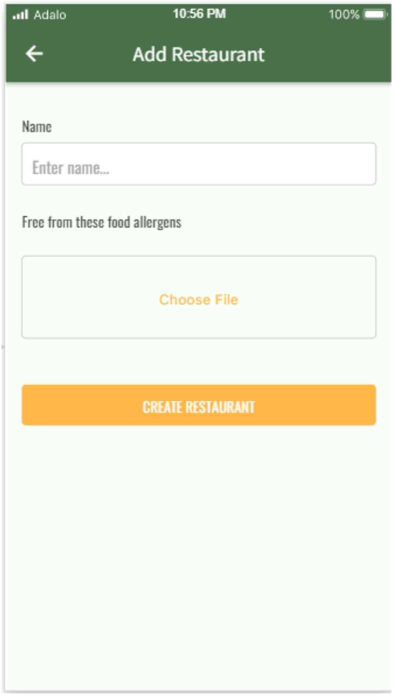


FIGURE 5 IF ADD RESTAURANT IS CLICKED

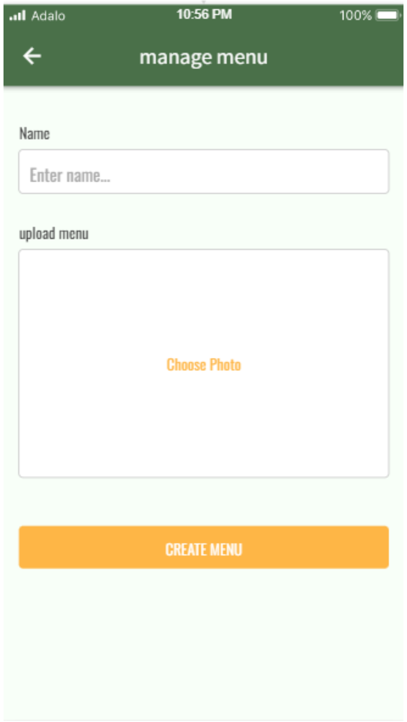
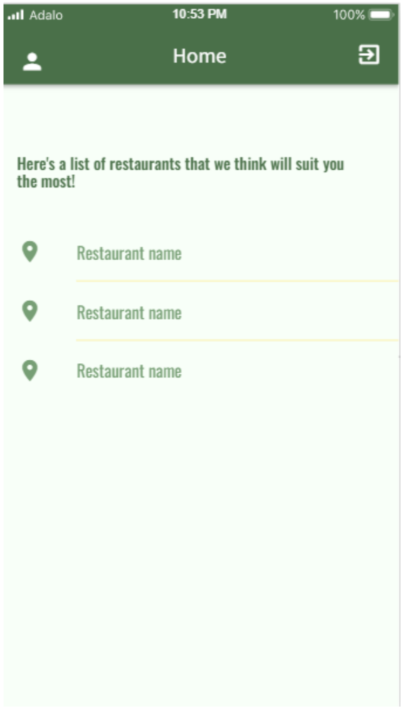


FIGURE 6 IF MANAGE MENU IS CLICKED



Here's a list of restaurants that we think will suit you the most!

FIGURE 7 HOMEPAGE FOR CLIENT

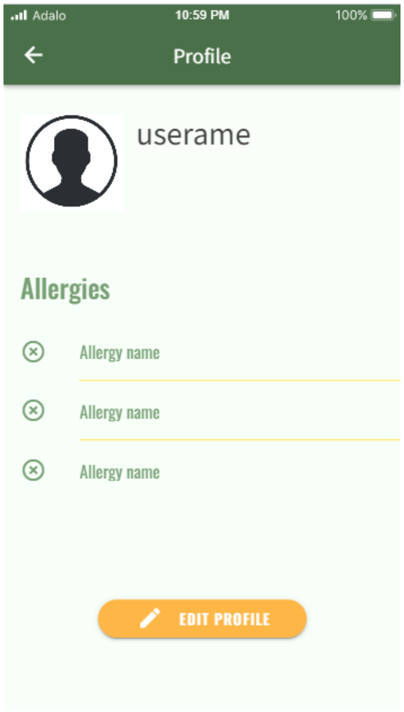
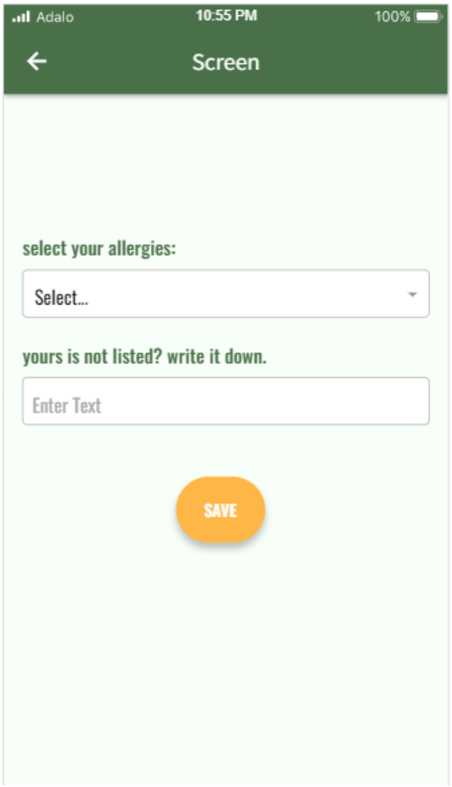


FIGURE 8 PROFILE PAGE AFTER CLICKING PROFILE ICON



select your allergies:

Select...

yours is not listed? write it down.

Enter Text

FIGURE 9 EDITING PROFILE

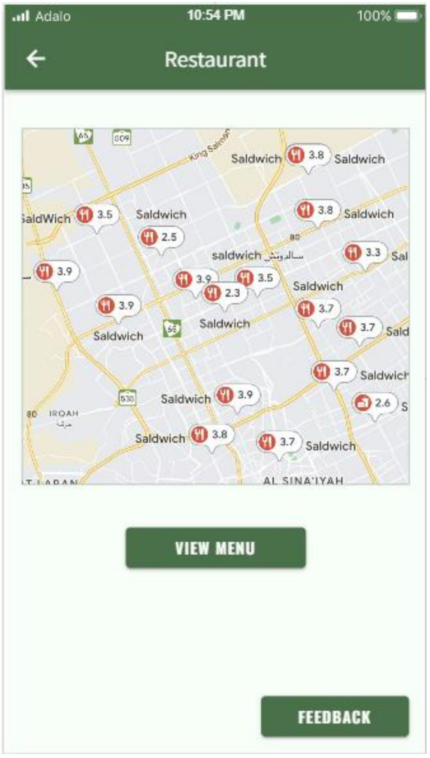
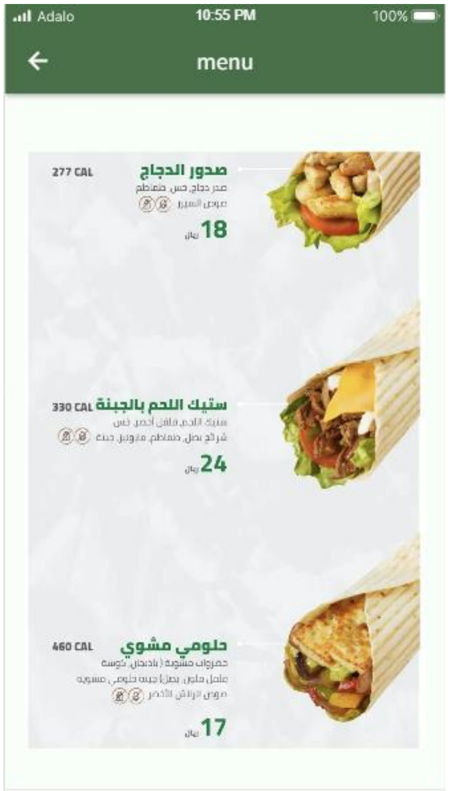


FIGURE 10 RESTAURANT FROM RECOMMENED IN CLIENT HOMEPAGE



Adalo 10:55 PM 100%

menu

صدور الدجاج 277 CAL

صدر دجاج، خس، طماطم

صوص السيزر

18 ريال

ستيك اللحم بالجبنة 330 CAL

ستيك اللحم، فلفل أخضر، خس

شرائح بصل، طماطم، مايونيز، جبنة

24 ريال

حلومي مشوي 460 CAL

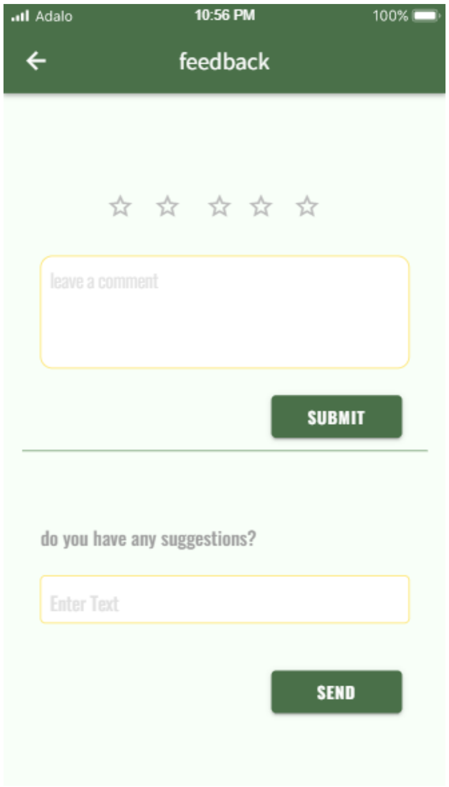
خضروات مشوية (باذنجان، كوسة

فلفل ملون، بصل) جبنة حلومي مشوية

صوص الرانش الأخضر

17 ريال

FIGURE 11 MENU OF RESTAURANT



Adalo 10:56 PM 100%

feedback

☆ ☆ ☆ ☆ ☆

leave a comment

SUBMIT

do you have any suggestions?

Enter Text

SEND

FIGURE 12 FEEDBACK FOR RESTAURANT

References

[1] Sommerville, I. (2010). Software Engineering: United States Edition (9th ed.). Uppersaddle River, NJ: Pearson.

[2] "Lucid visual collaboration suite: Log in," *Lucid.app*. [Online]. Available: http://lucid.app. [Accessed: 03-Nov-2022].

[3] "Flowchart maker & online diagram software," *Draw.io*. [Online]. Available: http://draw.io. [Accessed: 03-Nov-2022].

Individual contribution table

Name	**Contribution**
Fahda Alqahtani	Functional Requirements, Non-Functional Requirements, Stakeholders, Actors and Goals, Use Case Casual Description, Use Case Fully Dressed Description #1, Class Diagram, Object Diagram, State Diagram, The System Organization, Control Flow, Unit Testing, Integration Testing, Acceptance Testing.
Shaden alkhammash	Problem Statement, Functional Requirements, Non-Functional Requirements, Use Case Fully Dressed Description #2 #3, Class Diagram, Methods and Descriptions, Identifying the Subsystems, Unit Testing, Integration Testing, Acceptance Testing, Interface.