

SWE321: Software Architecture and design
2nd Semester 1443 H

Project

Course Code / Title: SWE 321

STUDENT IDENTIFICATION:

Section: 46474

Group Number: 3

Name: Shahed Hamadmad(leader)	ID: 441203854
Name: Ruyuf Adil Albarrak	ID: 439200293
Name: Ruba abdullah Alfelaih	ID: 441201010
Name: Fatimah Dagriri	ID: 441201258
Name: Yomna Abdullah Almalike	ID: 439201224
Name: Sara Aboumahmoud	ID: 441203797

NAJEEA



Software Architecture and design 2022 fall

Table of contents

Revision Control History	4
1. Introduction to Najeaa	5
2. Problem Domain Analysis	6
2.1 System Glossary	7
3. System Context View	8
4. Functions Requirement (FR) & Non-Functional Requirement (NFR) of the System	9
4.a Functional requirements:	9
4.b Non-Functional Requirement (NFR)	9
5. Challenges	10
6. Projection	11
7. Introduction to architecture part	13
8. methodology	14
9. System Architecture	15
9.1 Design decisions	15
9.2 Domain model	16
9.3 Architectural Style	17
9.4 Structural model	18
9.4.1 use case diagram:	18
9.4.2 use case description :	19
9.4.2.1 use case #1	19
9.4.2.2 use case #2	20
9.4.2.3 use case #3	21
9.4.2.4 use case #4	22
9.4.3 sequence diagram	23
9.4.3.1 sequence diagram for use case #1 Add review	23
9.4.3.2 sequence diagram for use case #2 pay online	24
9.4.3.3 sequence diagram for use case #3 search for offers	25
9.4.3.4 sequence diagram for use case #4 Add offer	26
9.5 User interface	27
10 .Non-Functional Properties:	33
10.1 Performance:	33
10.2 Scalability:	33
10.3 Flexibility:	33
10.4 Useability:	33
10.5 Portability:	33

10.6 Security:	33
10.7 Design Constraints:	33
11. Quality Assurance:	34
11.a. Reviews.	34
11.b. Verification.	34
11.c. Validation	34
11.d.Acceptance Criteria.	34
12. Future Considerations:	35
13. Introduction to design part	36
14. Detailed design	37
14.a Detailed system architecture:	38
14.b Detailed structural model:	45
14.b.1 Class diagram:	45
14.b.2 VOPC Diagrams:	46
14.b.2.1 VOPC diagram for use case #1 Add review	46
14.b.2.2 VOPC diagram for use case #3 search for offers	46
14.b.2.3 VOPC diagram for use case #4 AddOffer	47
14.b.2.4 VOPC diagram for use case #2 pay online	48
14.c Dynamic model:	49
14.c.1 State Diagrams:	49
14.c.1.1 State diagram for use case #1 Add review	49
14.c.1.2 State diagram for use case #2 pay online	49
14.c.1.3 State diagram for use case #3 search for offers	50
14.c.1.4 State diagram for use case #4 AddOffer	50
15. Implementation Consideration:	51
16. Architecture Analysis/Testing:	53
16.a Reviews.	53
16.b verification.	53
16.c Validation	53
17. Deployment and Mobility:	54
18. Personal Reflection:	55
References:	56

Revision Control History

Version Number	Date	Reason
1	1 Mar 2021	Problem Definition
2	3 Apr 2021	System Architecture Document
3	12 May 2021	System Design Document
	14 May 2021	Final Architecture And Design Document
	15 May 2021	Changes to version 1 and version 2: 1- point 6 of functional requirements 2- methodology 3- clear photo for sequence diagram #2 and #4 4- Design modes (box and line diagram) 5- revision sequence diagram #1

1. Introduction to Najeea

Najeea is a mobile application that can be downloaded on any mobile device and used anywhere to help people maintain a healthy lifestyle. Najeea displays a summarized description that is machine generated based on the members reviews that has been analyzed and thoroughly inspected. The description depicts three major components: quality of the service, price, and additional services.

With rapid growth of review and opinion-based applications especially those related to locations, reading reviews can be cumbersome and overwhelming at large amount. Consequently, reviews can involve unrelated information and biased opinions that will influence the reader to form unvaluable opinion. Najeea aims to resolve this issue by mining members reviews and analyzing them then displays a summarized review that describes all the relevant details mentioned towards a POI, the summary as mentioned will contain the quality of the service, price, addition services, and an overview that describes what the location is specifically (e.g., gym, healthy restaurant, healthy market).

Living a healthy lifestyle extends not only life but also rejuvenates the body by remaining healthy, and the mind stays intuitive and fresh. But maintaining a healthy lifestyle is hard. And that's how Najeea come in useful.

To conclude, Najeea serves you the ability to read and write reviews that are text based and that are rated through a scale with the added benefit of reserving an offer for any subscription. This way member will always be aware of new healthy places around them and be the first to know new offers.

2. Problem Domain Analysis

Our project aims to help maintain a healthy lifestyle by finding the best healthy places around our members, which will make a healthy lifestyle easier and more. Practical. Many hours of our time are wasted due to looking for a good gym, cooking a healthy meal, or going to a far place because of not knowing the nearest healthy place which will leave us to read a lot of reviews that contain unrelated opinions or overdetailed information about a place. In our application, we intend to locate every close healthy place to spread the knowledge of every new healthy place and sharing all offers and subscriptions.

Moreover, there is another problem that our application intends to solve, which is reading a lot of reviews. Najeea will resolve this problem by analyzing all members' reviews then provide a summarized review about the place which will save time and effort.

2.1 System Glossary

Abbreviation	Description
Healthy Lifestyle	Is a way being one which helps to keep and improve people's health and well-being and overcome lots of stress.
Opinion Based	An impression based on judgment or belief not founded on certainty or proof.
Summarized Review	A combination of the member's opinion of a certain location that includes quality of the service, price, and additional services.
POI - Point of Interest	A location or specific spot in which a customer can find interesting.
Location	A particular place or position.
Member	A registered user that can access the system and has all the functions.
Offer	A presenting of something for acceptance.

3. System Context View

user scenario A:(Najeea search for a service”gym” functionality)

the user opens the Najeea”نجيع” Application, they register and fill out their information. the user will be one of our users after completion the registration, then from the home page he could see a menu for our Service provider then choose the tab “list of the available gyms” . it’s will display the gym around their location. They could search for specific services in the gym and working time. Also, in this tab the user could see the review of different people for different gyms , the user chooses The fitness time gym then it displays the gym page ,so he will be able to see the services(swimming pool, gacozy all exercise equipment) and open and close time(open 7AM- close 11PM) .So after choosing the desired gym we guide them to the fitness time branch that he want and subscribe in an easy way.

user scenario B:(Najeea view the review of list of service”restaurants” functionality)

The user opens the Najeea”نجيع” Application, and after completion of the registration, from the home page he can choose the tab “list of the restaurants”. Displays a list of restaurants near them, showing them in order by their location, then he could search for a restaurant, and select the category of restaurants. the user chooses The Health Box restaurant. Shown to them the restaurant page shows restaurant photos and menu and most importantly a brief review generated from user comments the review stated: “The Health Box is a healthy restaurant, the service is great, and often the restaurant is described as cool, comfortable and well suited for meal after an exercise”. User read the summary without having to look at other reviews and irrelevant comments. They made the decision to visit this restaurant after the exercise in the gym.

4. Functions Requirement (FR) & Non-Functional Requirement (NFR) of the System

4.a Functional requirements:

1. The Customer shall be able to register to the system by his/her (Name, phone number, E-mail, and a password).
2. The Customer shall be able to login using his/her (E-mail and a password).
3. The System shall display available Service Provider around user's location according to its category (healthy market, healthy restaurant, Gym).
4. The System shall display the user's reviews for each Service provider.
5. The Customer shall be able to subscribe to interested offers.
6. The Customer shall be able to add his/her review.
7. The admin shall be able to login using his/her employee ID and a password.
8. The admin shall be able to add the offers of the Service providers.
9. The Customer shall be able to reserve an offer by pay online 20% of the service cost using his /her credit card (pay-pal, mada, visa card).
10. The Customer shall be able to search for a service using the Service provider name.
11. The System shall be able to generate a summarized Review depending on the users reviews to specify (quality of the service, Price, addition Services).
12. The Customer shall be able to display his/her Upcoming/previous reservations.
13. The Customer shall be able to filter the offers according to (Reviews, price, location, type of the service).

4.b Non-Functional Requirement (NFR)

1. The system response time shall be less than 3 seconds. (Performance)
2. The system shall be able to accommodate 1000 simultaneous users
3. The system shall be able to Support (English, Arabic) languages.
4. The customer shall be able to subscribe to an offer in 3 clicks.

5. Challenges

To carry out the project successfully and to reach the plan, we tried to identify the problems and challenges that we might face throughout the project.

Some of the expected challenges: Commitment to the project scheduled time, work distribution, selection of the best architecture of the whole system and for the subsystems, assigning the appropriate tasks to each subsystem, connecting the subsystems, and unfamiliarity with AI technology.

6. Projection

By the end of this project, we hope to come out with an app that successfully covers all the previously mentioned features such as allowing users to find reviews and ratings for available gym or the ability to save favorite gyms to a user list. Moreover, each feature should take no more than three to four days to be able to adherence to the time limit. we believe that the most valuable and important skills to acquire by the end of the project will be teamwork skills and creative skills, and we also look forward to gaining experience and learning to work through any difficulties we may encounter and find solutions to them.

Outline of Tasks and The Time Limit:

	phase	Category	Task	Start Date	End Date
	1	Problem Definition	Cover page	15 Feb 2022	1 Mar 2022
			Frontal matter		
			Introduction		
			Problem Domain Analysis		
			The System Context View		
			Functions Requirement (FR) &(NFR) of the System		
			Challenges		
			Projection		
	2	System Architecture Document	Cover page	1 Mar 2022	29 Mar 2022
			Frontal matter		

			Introduction		
			Problem Domain Analysis		
			The System Context View		
			Functions Requirement (FR) &(NFR) of the System		
			Challenges		
			Projection		
	3	System Design Document	Cover page	30 Mar 2022	14 Apr 2022
			Frontal matter		
			Detailed Design		
	4	Presentation and Final Report	Project Presentation	15 Apr 2022	21 Apr 2022

7. Introduction to architecture part

In this phase we had to work as system architects , we will define the methodology used for the system as well as a high level design to understand the system clearly and to choose the right architecture, use case diagram and use case description and sequence diagram defining the architecture of the system .also we briefly talked about quality assurance of the system and quality attribute .finally, we talked about our future consideration and what is the plan to do with the system in term of modification and enhancement.

as we see Najeea is a mobile application that can be downloaded on any mobile device and used anywhere to help people maintain a healthy lifestyle. Najeea displays a summarized description that is machine generated based on the members' reviews that has been analyzed and thoroughly inspected. The description depicts three major components: quality of the service, price, and additional services.

With rapid growth of review and opinion-based applications, especially those related to locations, reading reviews can be cumbersome and overwhelming at large amount. Consequently, reviews can involve unrelated information and biased opinions that will influence the reader to form invaluable opinions. Najeea aims to resolve this issue by mining members reviews and analyzing them then displays a summarized review that describes all the relevant details mentioned towards a POI, the summary as mentioned will contain the quality of the service, price, addition services, and an overview that describes what the location is specifically (e.g., gym, healthy restaurant, healthy market).

Living a healthy lifestyle extends not only life but also rejuvenates the body by remaining healthy, and the mind stays intuitive and fresh. But maintaining a healthy lifestyle is hard . and that's how Najeea comes in useful .

To conclude, Najeea serves you the ability to read and write reviews that are text based and that are rated through a scale with the added benefit of reserving an offer for any subscription. This way members will always be aware of new healthy places around them and be the first to know new offers.

we will describe this idea as system architects in the following .

8. methodology

<u>Task</u>	<u>(Assigned member(s</u>	<u>Prototype demonstrati</u> <u>duration</u>	<u>Final demonstration</u> <u>duration</u>	<u>Execution Time</u>
Register	Fatimah Dagriri	1day	1day	days 2
Offers	Ruyuf Adel Albarrak	1day	1day	days 2
reservation	Yomna Almalki	2days	1day	days 3
Service provider	Sara Aboumahmoud	1day	1day	days 2
Reviews management	Shahed Hamadmad	1day	days 2	days 3
Subscription provider	Ruba Abdullah Alfelaih	1day	days 2	days 3

9. System Architecture

9.1 Design decisions

While our system “Najeea” aim to maximize satisfaction of customer by increase the efficiency and performance of system , and also we focus on scalability of the system ,we will use the MVC for many reasons.

Firstly, while we want to increase the **performance**, so separate the user interface from business logic it will be good style to increase it.^[1]

Second , while our system may increase the **functionality ability** the MVC is Easier to maintain or modify, so that fit will.^{[1][2]}

Third, the **connection** between parts should be Loosely Coupled MVC achieve that ,to make it not complex so as to improve performance .^[2]

Fourth , while our system deals with the **languages(Arabic, English)** support multiple view is important MVC Easy to plug-in new or change interface views,thus allowing updating the interface views with new technologies without overhang the rest of the system, so it supports Multiple views.^[7]

Fifth , scalability is important in our application, and cannot guarantee scalability. As we cannot scale only the parts relating to performance, the application needs to be scaled as a whole.which a limitation for as .^[3]

Sixth, MCV is not very popular in mobile applications. It's known for web applications, but nowadays we can consider it for mobile applications. ^[5]

Finally, while we mention all our target of the non functional that deals properly MVC so our system can increase the efficiency and performance of system , and also we focus on scalability of the system.

9.2 Domain model

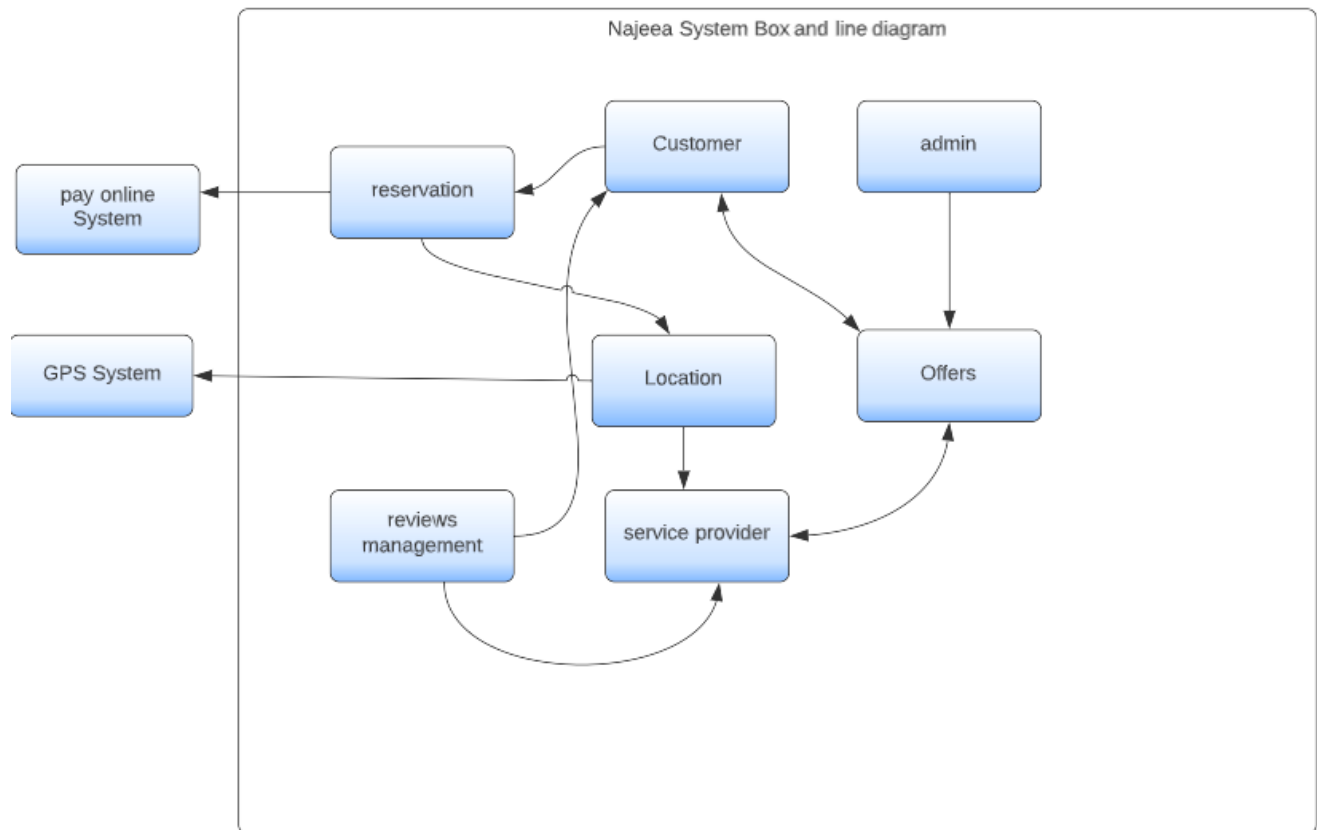


Figure 1: a box and line diagram that describes our system at a high level .

Customer : Registered user that can access the system functionalities.

reservation: it is private and only customers can see all its reservations that are kept in DB.

Location:A public space shows customer location to make the filter more efficient.

service provider: it represents our services in the System it can be(healthy market, healthy restaurant , gym).

offers: it is offers that admin set or offers that customers can subscribe with (Ex: 3 month by 2300 Riyals).

admin: Person that has an authentication to add an offer .

external systems (pay online System - GPS system):**GPS system:**navigation system consisting of a constellation of satellites for monitoring and control.**pay online System :** Fast & Easy Integration! Reduce Friction & Speed Up Checkouts. Increase Your ROI with a Seamless Payment Solution.

reviews management: manage its mean (add ,view)added by customer in specific service. and viewed by another customer.

9.3 Architectural Style

Our system can be composed into three major parts, **the first** part is basic functionality of mobile applications which could be represented in interface , Ex: registration, login ,search ,select offer ,writing review , which is all can implement as form. **the second** part of our system deals with more complex feature of our system and slightly deeper and work with background , this part will not have direct interaction with customer , Ex: service provider,subscription provider,reviews management, **The third** part is complex too , and should be secure and private and we can access it by a single model, its DB . while we want to increase our performance. That's why we separate the DB from the second part .[2]

The final summary must be well defined and structured and contain all the mentioned components , and functionality satisfaction , also non functional requirements, with all that has been selected we will adopt the MVC architectural style for our system.

How can we implement it , the MVC is short for model-view- controller, deals with our system perfectly as we divide our system into three parts and completely understandable and well defined, while MVC pattern decouples the system into three parts.

The model ,its central structure which obviously deals with DB in the system , and it provides you with service to query , it's for user information such as adding or retrieving some information.

The view, Data representation, It actually generates UI or user interface for the user and it is where my service gets displayed, in my cases it can be registration UI or log in and this is initiate the controller

The controller acts as an intermediary between view and controller, the controller doesn't have to worry about handling data logic models do ,the controller can be in my application .[2][1][11]

The MVC style provides us with separation of concerns therefore , it can be easily debugged , tested and maintained , also it is easily scalable and as we mention before it decouples the system and loosen any dependency between the components which allows for a modifiability.

We can use blackboard in some parts to deal with complexity, but the main architectural style is MVC.[11]

The last style was considered but we don't use it is SOA architecture, we decline it for many reasons, it's good in multiple aspect but Fail in two , first it not considerable for GUI-Based: SOA would not be suitable for applications with GUI functionality,second Real-time: SOA is not desirable to be used with require response times since the services communicate asynchronously.[4]

In conclusion , our main architecture is MVC which is decoupling smoothly and perfectly our system, we can use blackboard in some parts to deal with complexity , therefore , our system achieves functionality ,and non functional quality with this architecture.

9.4 Structural model

9.4.1 use case diagram:

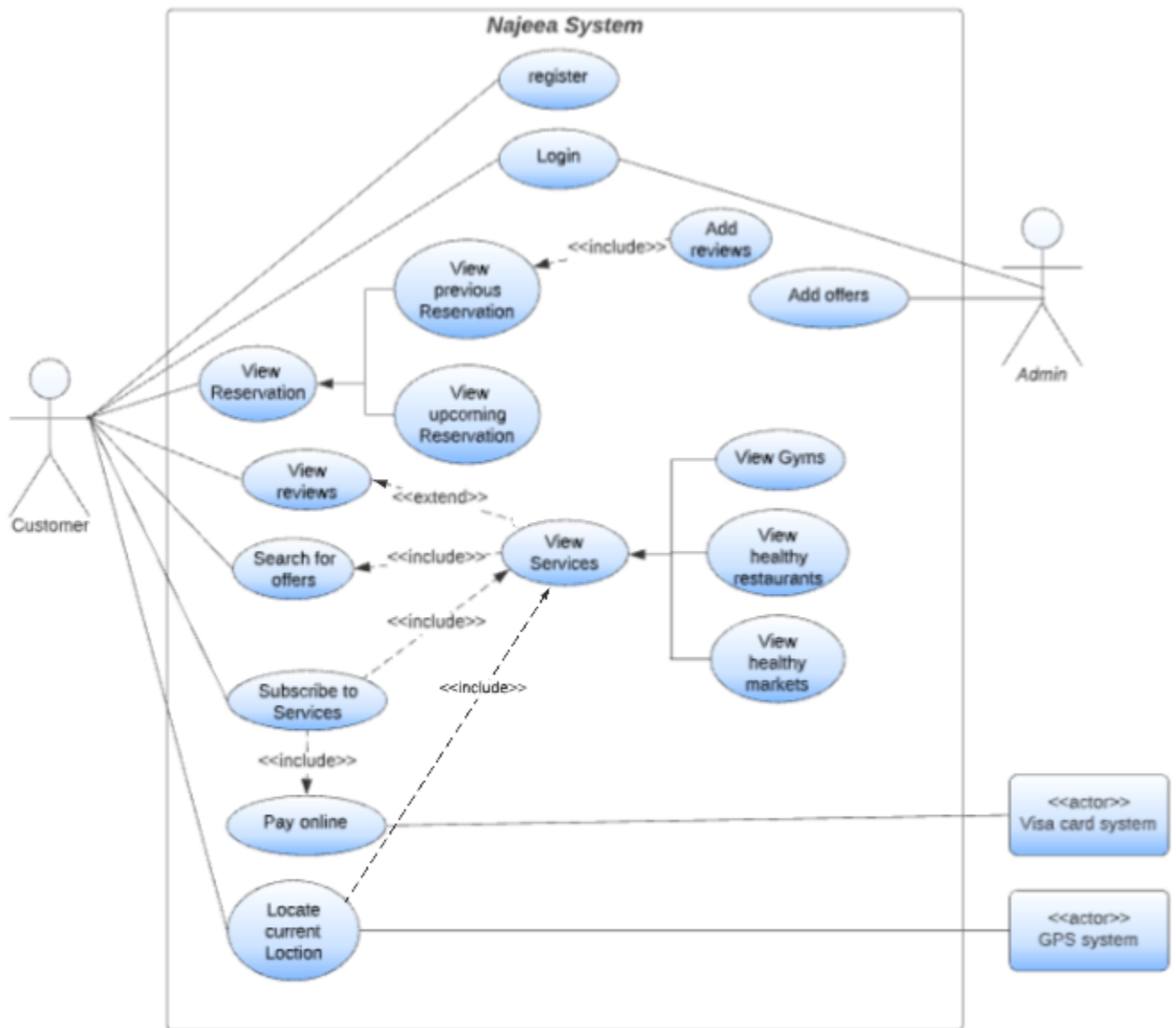


Figure 2: a use case diagram describing our system and the relation between our main functions, and also between our functions and external systems .

Description:

The use case diagram describes the general process of searching and viewing an offer of services. In this diagram the primary actor is the customer , it also shows the general and alternative flow of relationship. Customers must register / login before doing their actions.

[9.4.2 use case description :](#)

[9.4.2.1 use case #1](#)

<u>Use Case Description</u>	
System: Najeea	
Use Case name: Add Review.	
Primary actor: Customer	Secondary actor(s): -
Description: This use case describes how the customer add new reviews	
Relationships <ul style="list-style-type: none">■ Includes: None■ Extends: None	
Pre-conditions: The customer must be logged in.	
Primary Actor (Customer)	System
1- The customer select "Add Review".	2-The system displays text box.
3-The customer writes his/her review in the text box.	5- The system will add the new review.
4- The customer submits the review.	6- The system will display a successful message.
Alternative and exceptional flows: In step 3, if the customer forgot to enter a specific field, then step 4 the system display a message with the required field	
Post-conditions: Successful condition: The system shall add the new review. Failure condition: The system fails to add the new review.	

9.4.2.2 use case #2

Use Case Description		
System: Najeea		
Use Case name: pay for offer		
Primary actor: Customer	Secondary actor(s): Visa payment system	
Description: This use case describes how a customer pay for his/her chosen offer using visa card.		
Use Case Description		
Relationships Includes: none Extends: none		
Pre-conditions: 1. Customer is logged in successfully. 2. Customer choose an offer to subscribe with		
Steps:		
Primary Actor (Customer)	System	Secondary Actor(s): Visa Card payment system
1-the use case begin when the customer selects the“pay” option from the offer page.	2- The System displays a form that requires his/her Credit Card Information (Card Number, Card Verification Code (CVC), User Name (same as on the card), Expiry date).	
3-Customer enters Visa information and selects “Next”.	4-The system verifies the submitted form. 5- The system sends credit card information and the amount to the Payment System.	6- Bank system confirms Card details 7-The payment system withdraws the required amount.
	8- The system displays a success message.	
Alternative and exceptional flows:		
The use case ends with a failure condition. if the card system failed to verify the submitted form		
1-the system reject the payment process.		
2-The system display an error message.		
3- Back to step 2 .		
Successful condition: payment completed, and the offer subscribed successfully.		
Failure condition: payment failed, and the offer doesn’t subscribed successfully.		

9.4.2.3 use case #3

Use Case Description	
System: Najeea	
Use Case name: Search for offers.	
Primary actor: Customer	Secondary actor(s): -
Description: This use case describes how a customer searches for available offers.	
Relationships <ul style="list-style-type: none">▪ Includes: None▪ Extends: None	
Pre-conditions: None	
Steps:	
Primary Actor (Customer)	System
1. The use case begins when the customer selects “search” from the home page. 3. The customer fills the textbox with the service provider name. 4. The customer submits the name.	2.The system presents a textbox to type the service provider name 5.The system displays a list of available offers for the specified service provider, and the customer will be able to filter them.
Alternative and exceptional flows: 1.Invalid Service provider name. If in step 3 user specify an invalid service provider name <ul style="list-style-type: none">1.The system displays a message indicating that the name is invalid “Not a service provider name”2.Step 2 is resumed 2. Customer quits If at any step before step 4 the user leaves the search area <ul style="list-style-type: none">1. The use case ends with a failure condition	
Post-conditions: Successful condition: The system successfully display a reminder for a list of available offers for the specified service provider name. Failure condition: No offers will be displayed.	

9.4.2.4 use case #4

Use Case Description	
System: Najeea	
Use Case name: Add Offers	
Primary actor: Admin	Secondary actor(s): -
Description: This use case describes how the admin adds an offer to the service providers.	
Relationships Includes: None Extends: None	
Pre-conditions: Admin is successfully logged in	
Primary Actor (Admin)	System
<p>1-The use case begins when the admin selects the “Add Offer”.</p> <p>3-The admin fills the form with the desired information:</p> <ul style="list-style-type: none">• Category (determine the type of service provider)• Name• Location• Price• Photo• Offer details <p>4-The admin select “submit” option.</p>	<p>2-The system displays a form with required fields to add the offer.</p> <p>5- The system verifies that all required fields are filled.</p> <p>6- The system presents a message indicating that the offer information is added successfully.</p>
Alternative and exceptional flows: <ul style="list-style-type: none">• Submit the form with an empty required field:<p>If in step 4 the admin submits the form with a missing required field, then:</p><ol style="list-style-type: none">1. the system displays a message indicating an empty required field is missing.2. step 3 is resumed.• Admin quits:<p>If at any step before step 4 the admin selects “Cancel”, then: the use case ends with a failure condition.</p>	
Post-conditions:	
Successful condition: The offer is successfully added to the service provider.	
Failure condition: The offer isn't added to the service provider.	

9.4.3 sequence diagram

9.4.3.1 sequence diagram for use case #1 Add review

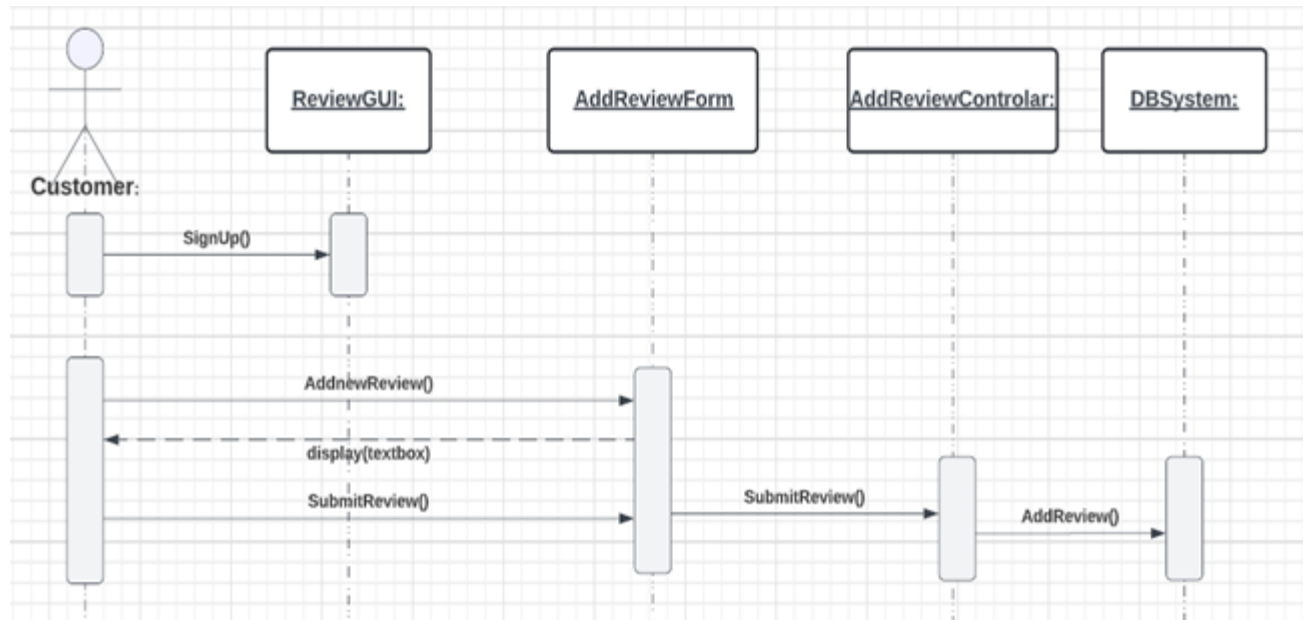


Figure 3: sequence diagram that describe the flow of add review use case

9.4.3.2 sequence diagram for use case #2 pay online

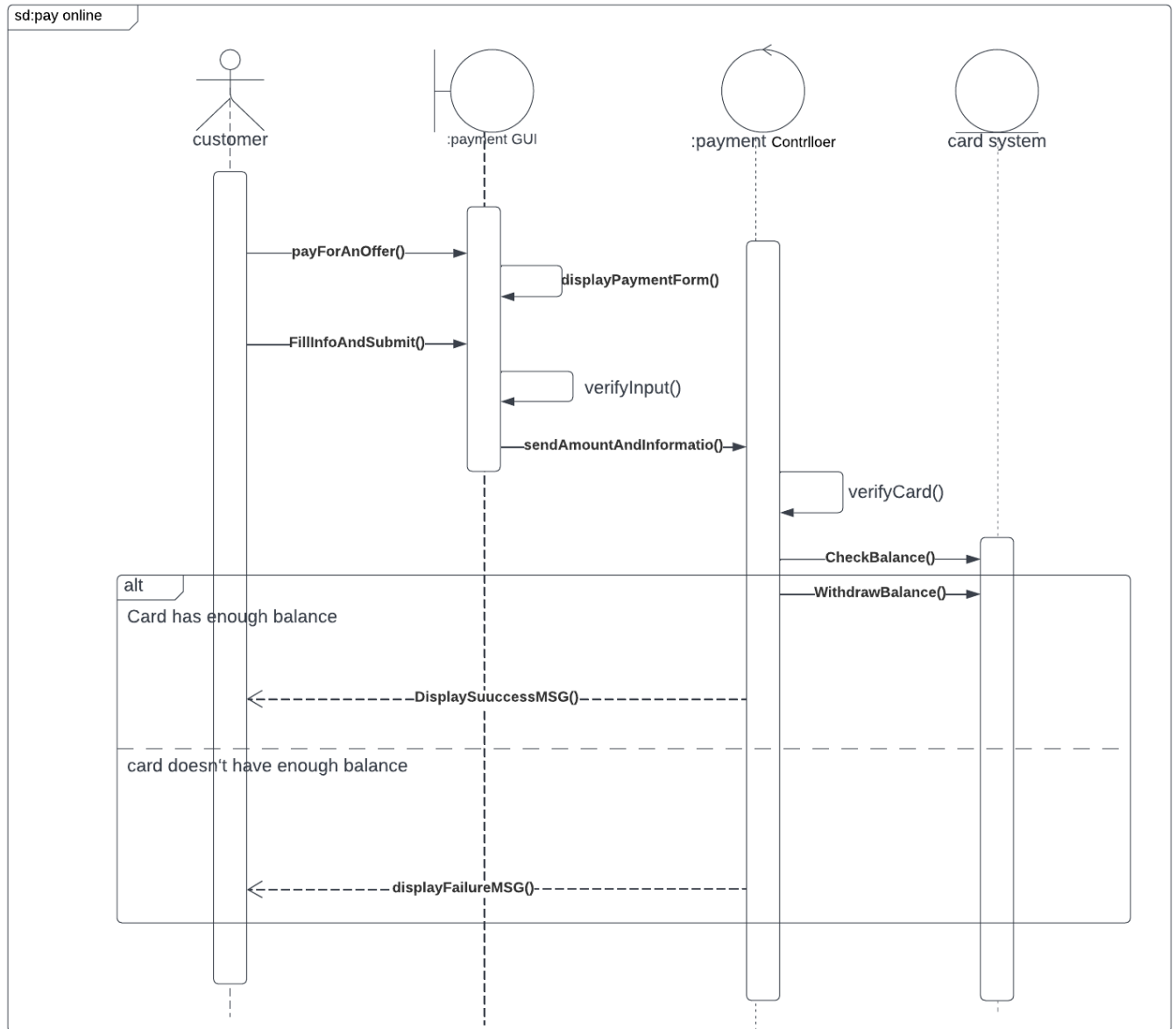


Figure 4: sequence diagram that describe the flow of pay online use case

9.4.3.3 sequence diagram for use case #3 search for offers

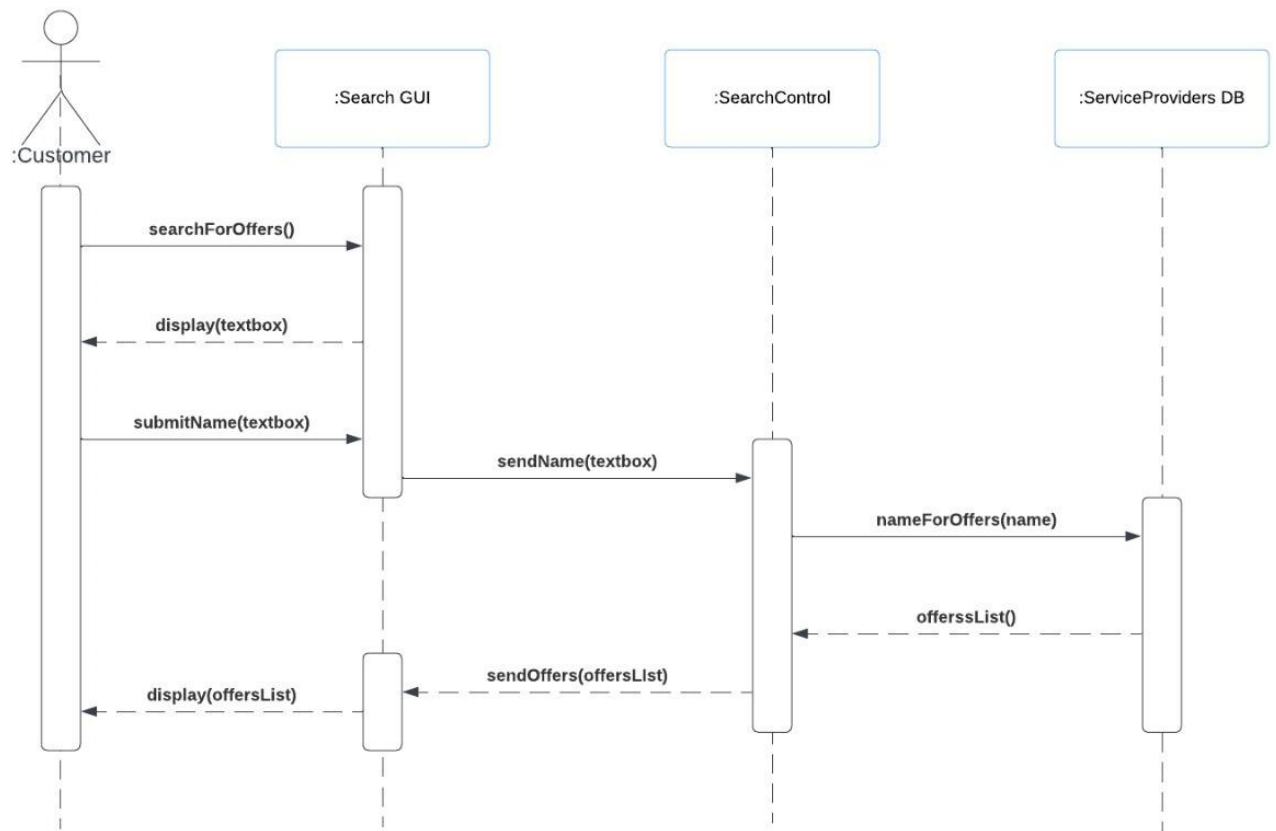


Figure 5: sequence diagram that describe the flow of search for offer use case

9.4.3.4 sequence diagram for use case #4 Add offer

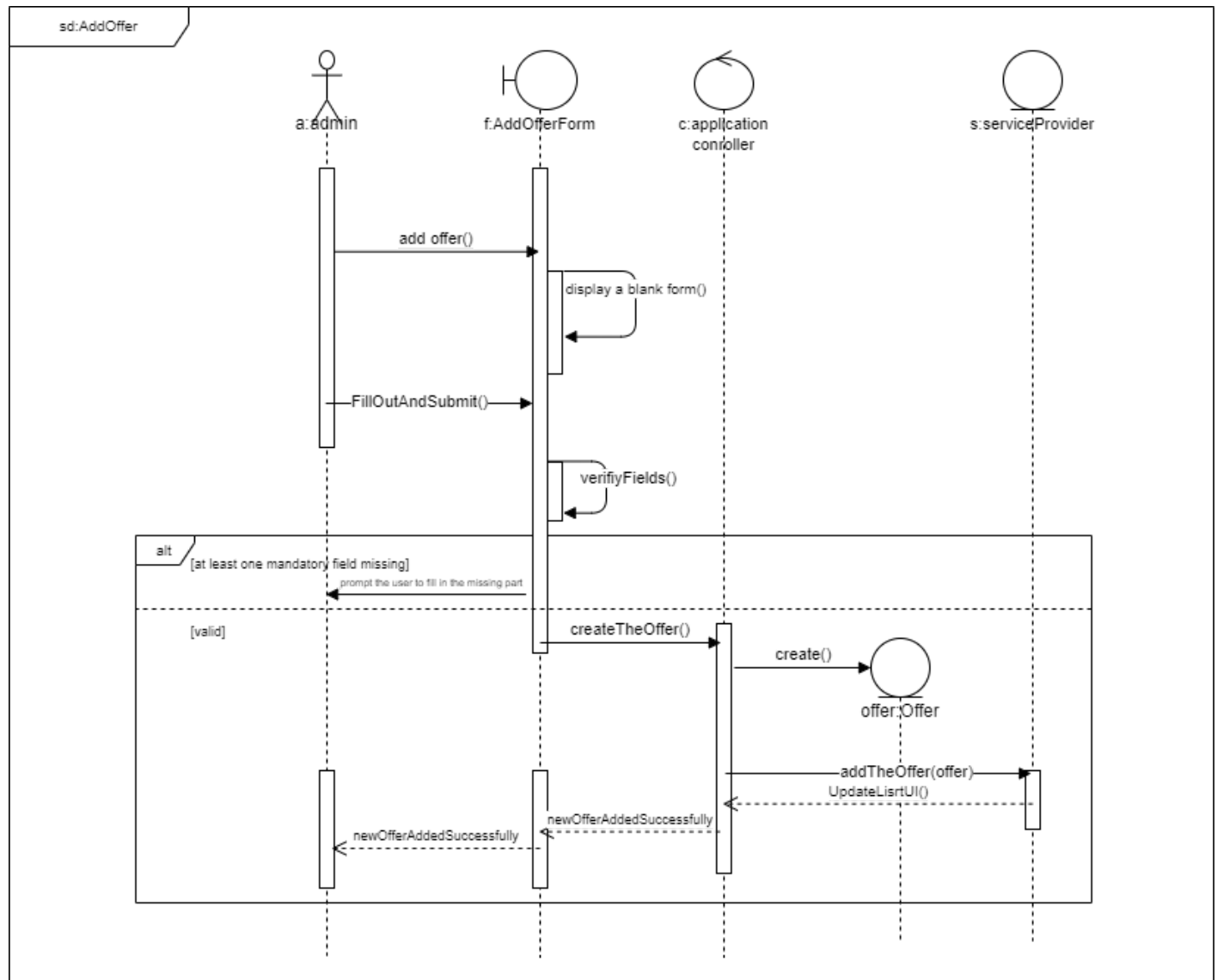


Figure 6: sequence diagram that describe the flow of add offer use case

9.5 User interface

- **Start UI**

This is the first page the user will see when he opens the application.



Figure 7: Start page user interface

- **Register UI**

To register to the system, user must fill their information then press “register”.

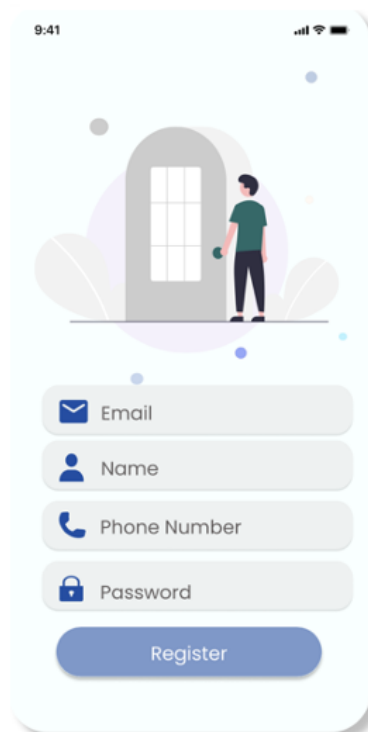


Figure 8: Registration user interface

- **Successful Register UI**

When the user fills their information then press “register” successfully

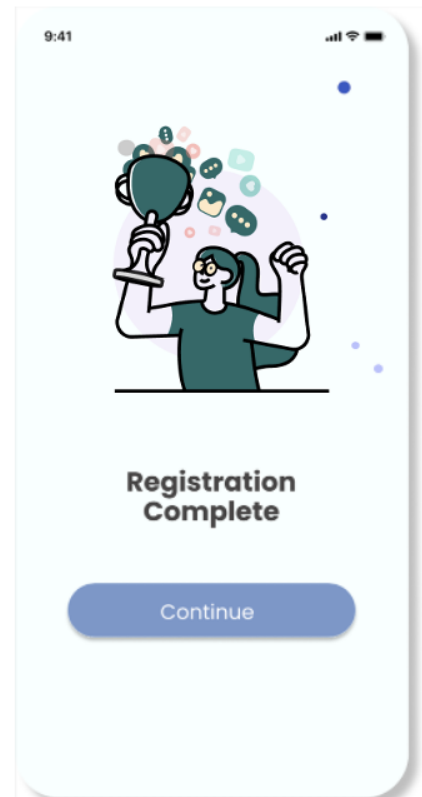


Figure 9: successful registration user interface

- **Login UI**

A member can login to the system by providing their email/phone number and password.

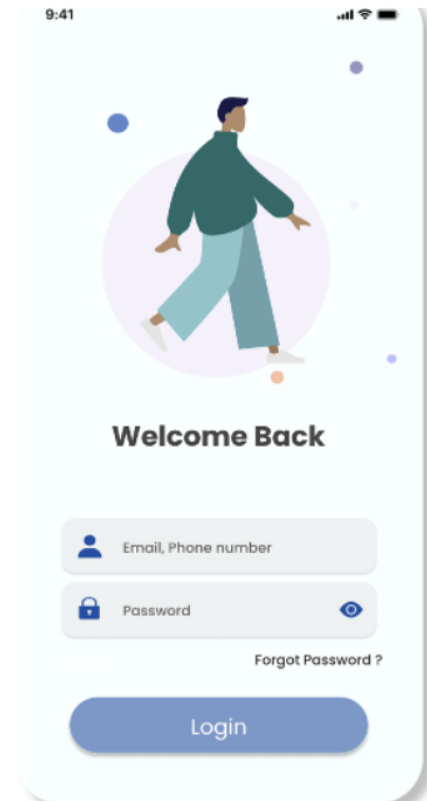


Figure 10: Login user interface

- **Home page UI**

Once a member logs in they can access all services, check their favorite places, and search for services from here.

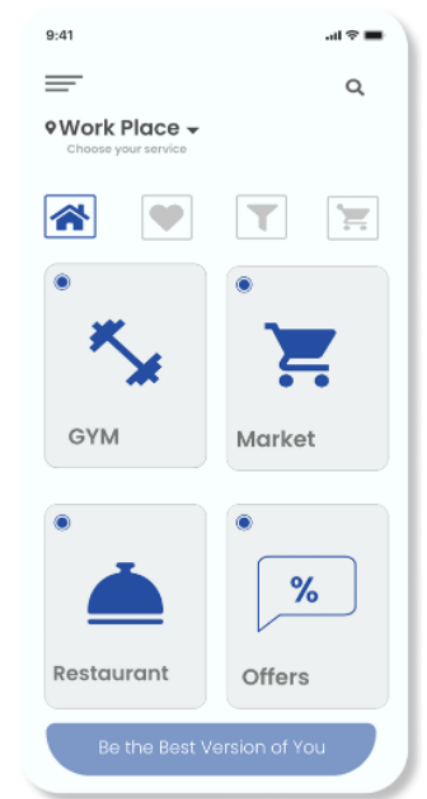


Figure 11: Home page user interface

- **Search UI**

A member can search for services or places using the search bar and they can filter the results by nearest first which is accessible via the location icon in the top right of the page.

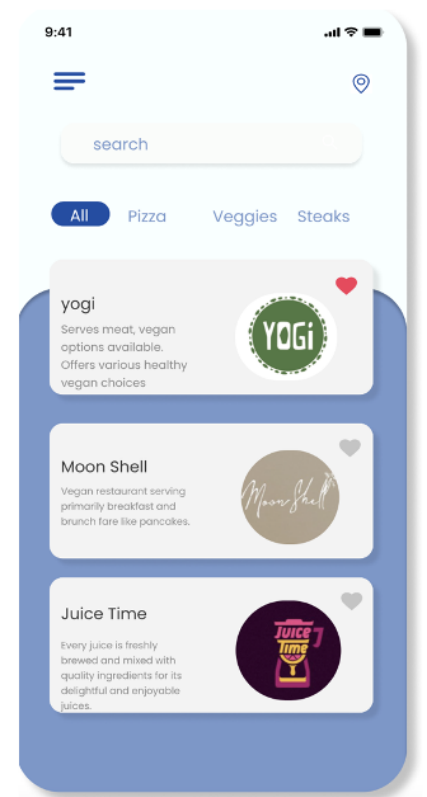


Figure 12: Search user interface

- **Menu UI**

A member will always access home page, recent offers, addresses, notifications, location, settings, and logout. From the menu icon in the top left of the page.

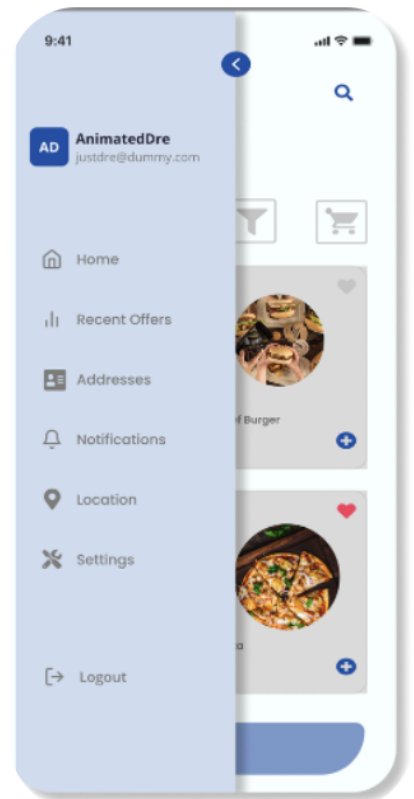


Figure 13: Menu user interface

- **Review UI**

When a member chooses a specific place, it would direct them to the review information interface. Which has all the important information.

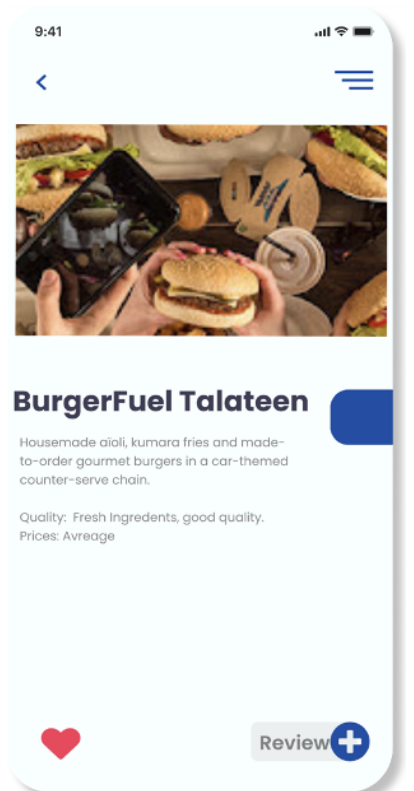


Figure 14: View reviews user interface

- **Add review UI**

If the member clicks on the icon bottom right, the system will show a popup window to add a new review.

A light gray rounded rectangle representing a popup window. At the top, it has the title "New Review" in bold black text. Below the title is a large rounded rectangle with a blue border, containing the text "Write your review here, please include quality of the service, Price, addition Services." in a smaller font. At the bottom of the popup is a blue rounded button with the text "Submit" in white.

Figure 15: Add a review user interface

- **Add offer UI**

If the admin wants to add a new offer, he will need to add a name, location, category, price, and other details.

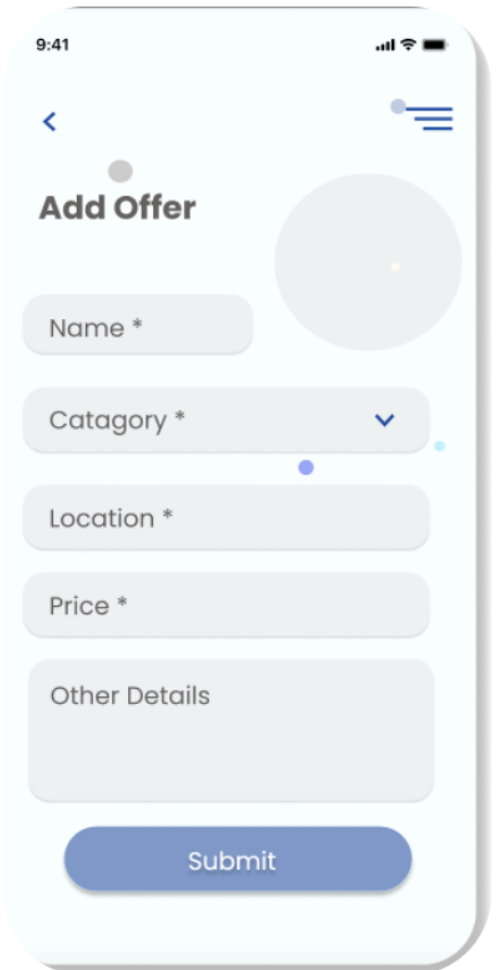
A mobile app screen titled "Add Offer" in bold black text. The screen has a light blue background. At the top, there is a status bar with the time "9:41" and signal icons. Below the title, there is a large gray circle with a small orange dot in the center. Below this are four input fields: "Name *" (a rounded rectangle), "Category *" (a rounded rectangle with a blue dropdown arrow), "Location *" (a rounded rectangle), and "Price *" (a rounded rectangle). Below these is a larger rounded rectangle labeled "Other Details". At the bottom of the screen is a blue rounded button with the text "Submit" in white.

Figure 16: Add an offer user interface

- **Payment UI**

The member will have two options to pay for an offer either pay with a saved card or add a new card.

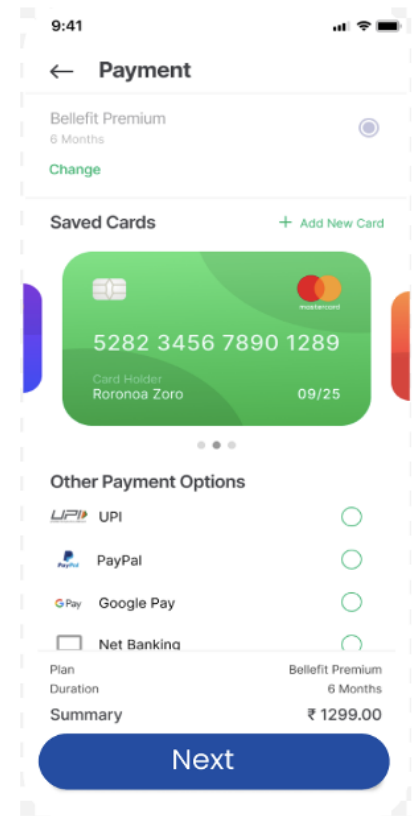


Figure 17: Payment user interface

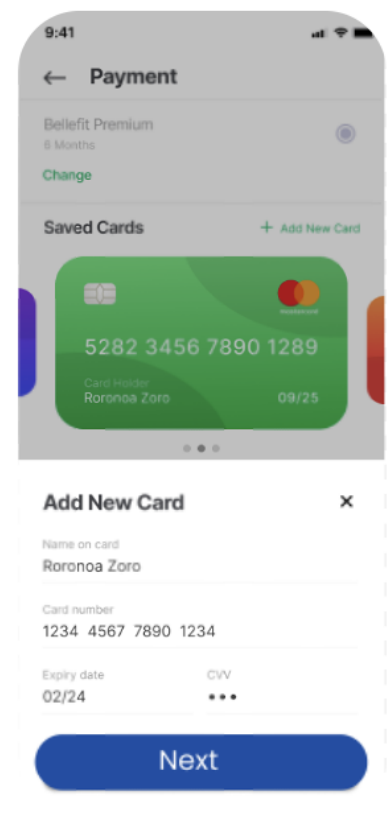


Figure 18: Add new card user interface

10 .Non-Functional Properties:

10.1 Performance:

Performance is essential in 'Najeea' application. Multiple users may subscribe to offers at the same time, so the system shall accommodate 1000 users accessing it at the same time. The operations must be done in a specific period of time which is not to exceed 3 seconds for each, this can be achieved by using caching (copying information) into a faster storage system.

10.2 Scalability:

Najeea must manage a large amount of information and handle an increasing amount of work. As mentioned previously, it will allow many users to use it at the same time and in different places, so we should use multiple servers and separate the users between them.

10.3 Flexibility:

Because we have future considerations, we use MVC and Blackboard architecture, when use them the changes do not affect the entire model, making it easy to modify and add features, resulting in a faster development process. It should take no more than three hours to add new functionality.

10.4 Useability:

Usability is important in Najeea since the customer shall be able to subscribe to any offer in 3 clicks. There are some customers who are not familiar with technology, lack of usability leads to customers' frustration and to probably rejection of the application. To avoid this, we will try to get feedback from the customers (representative users) to improve the design (iteration design).

10.5 Portability:

We will be developing two versions of the app, the first one is Android, then we will develop the iOS version. We will make our application run in the new environment ,the effort required to adapt it to the new environment is within reasonable limits.

10.6 Security:

The access permissions for system data may only be changed by the system's data administrator. And the passwords shall never be viewable at the point of entry or at any other time.

10.7 Design Constraints:

- The System shall run under Android operating system, and it should have the ability to transfer later to IOS operating system.
- The system shall be written in Java programming language.
- The system shall use a firebase(cloud-hosted) as DBMS.
- The system shall be compatible with the visa card system.
- The system must be implemented within 5 months.
- The system shall support (Arabic, English) language.

11. Quality Assurance:

11.a. Reviews.

Review is the principal method of validating the quality of a process or of a product. In our project we will conduct 2 types of reviews: walkthrough and checklists.

Walkthrough is an informal review performed by peers. It needs no prior preparation, so every piece of work will be given to one or more colleagues to be checked and to benefit from their comments.

Checklists are lists of items that should be checked during the review. Checklists are useful to support walkthroughs and they are Straightforward to use. We will prepare the needed checklists (for reviewing code, for reviewing software design. etc.) and we will assess them throughout the development process.

11.b. Verification.

Verification is the process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. It answers the question: Are we building the product right?

We will plan verification tests to ensure that, such as testing if the system throughput, performance, etc. are working as specified in non-functional requirements.

11.c. Validation

After completion of the verification test.

- We will use the contract review to ensure that each requirement is met.

In addition we will prepare a white box testing that concerns the input and output of the system so that internal testing is done by the developers and the testers to ensure if the requirements are met or not.

We will use checklists to assure that every test case meets its requirements.

- We will be confident in our security of the system by paying for Ethical Hacking to expose security flaws in the organization system.

- The system will follow rapid application development which will ensure the expandability of the system

- We will ensure the performance of the system by applying Spike testing in our system which is test how the system react if it receive a large spikes in the load by the users.^{[7][8][9]}

11.d. Acceptance Criteria.

Acceptance criteria are important to define boundaries and help us achieve a high quality there, the users will surely install our mobile app on Google Play or App store platforms depending on their Operating System.

For the testing we will provide a beta version to let the users interact with the system and collect their feedback.

The user will be trained by tutorials the first time he/she opens the app.

In the other hand no need for the technical documentation the user will be understand the system by the tutorials provided in the app.^[10]

12. Future Considerations:

Our system serves a huge domain and target users, future modifications are expected, and we are fully aware of that. One of the features we would like to add is the ability for users to get points from purchasing or joining a gym from our application, this way members will always try to get offers from the application and they will get something in return for helping others. With points members will get money, extra sales, or gifts.

Another feature is the ability for members to check-in and out in any location they go, which will be saved in their accounts as posts. Luckily, our architecture can accommodate these changes and will not cause other components or subsystems to be altered, all we need to do is define the model, view, and controller components and their interactions and behavior which will be relatively simple and similar to the use of other components and modules.

We plan on adding any necessary changes that will help members in navigating and using the system and that will also provide usability and additional ease to ensure that they are getting the best of our system.

13. Introduction to design part

Najea plays an important role for society and for our life, it has a big impact on human health, quality of human life, Fitness and other aspects of life.

We are building a design for a system that deals with other companies and service providers, and dealing with some data to support these goals.

The purpose of the design document is to describe the system architecture in a detailed manner, and contribute a classification of the design of a system sufficient to permit the development to progress by defining all system components and their connections and interfaces to be clear to the programmers to develop the system. it allows the software development to proceed with an understanding of what is to be built and what is not expected to be built. It assures that the system is built to meet the needs and is on par with what was agreed upon prior to the inception of the software purposes.

In this phase we will be concerned to refine some aspects of the previous system architecture and to modify some issues in the overall design. also, we will provide a clear specification of each component, description of the algorithms and data structures, describe the non-obvious implementation techniques, and detailed structural model and describe the dynamic behavior of the system.

14. Detailed design

Since our "Najeer" system performs many functions, maintenance and performance become our focus. Therefore, we chose to implement the Model View Control (MVC) architecture along with the blackboard architecture as suitable architectures for our system, thus increasing the efficiency and effectiveness of the system.

The Model-View-Controller (MVC) architecture is important for separating concerns, resulting in easy code maintenance. Also, it can be easily extended without affecting any other modules, which is in line with our plan as we may need to update our system in the future as our requirements change and additional features can be added. As for the blackboard architecture, it is best suited for the problem of complex revision summarization, since it is used for complex systems. We plan to use it as a subsystem, and it will only be responsible for giving the revision summary to the main system. Moreover, due to the autonomy and loose coupling of knowledge resources, it is ideal when additional components need to be added to the system, or when an aspect of the subsystem needs to be reused.

For the alternative architecture style, we excluded Layered architecture due to its low performance, besides the fact that Layered architecture is difficult to scale.

In Najea we decided to use Linked List data structure. It is a sequence of links which contains items. Each link contains a connection to another link. Linked list operations will help Najea to complete its functionality by using its operations like insertion and deletion. The algorithms that Najea uses are the Sorting Algorithm, Searching Algorithm, finally, Dijkstra.

In conclusion, since Najeer focus is on performance, maintainability, and scalability, our decision is to use the MVC architecture along with the blackboard architecture as a subsystem.

14.a Detailed system architecture:

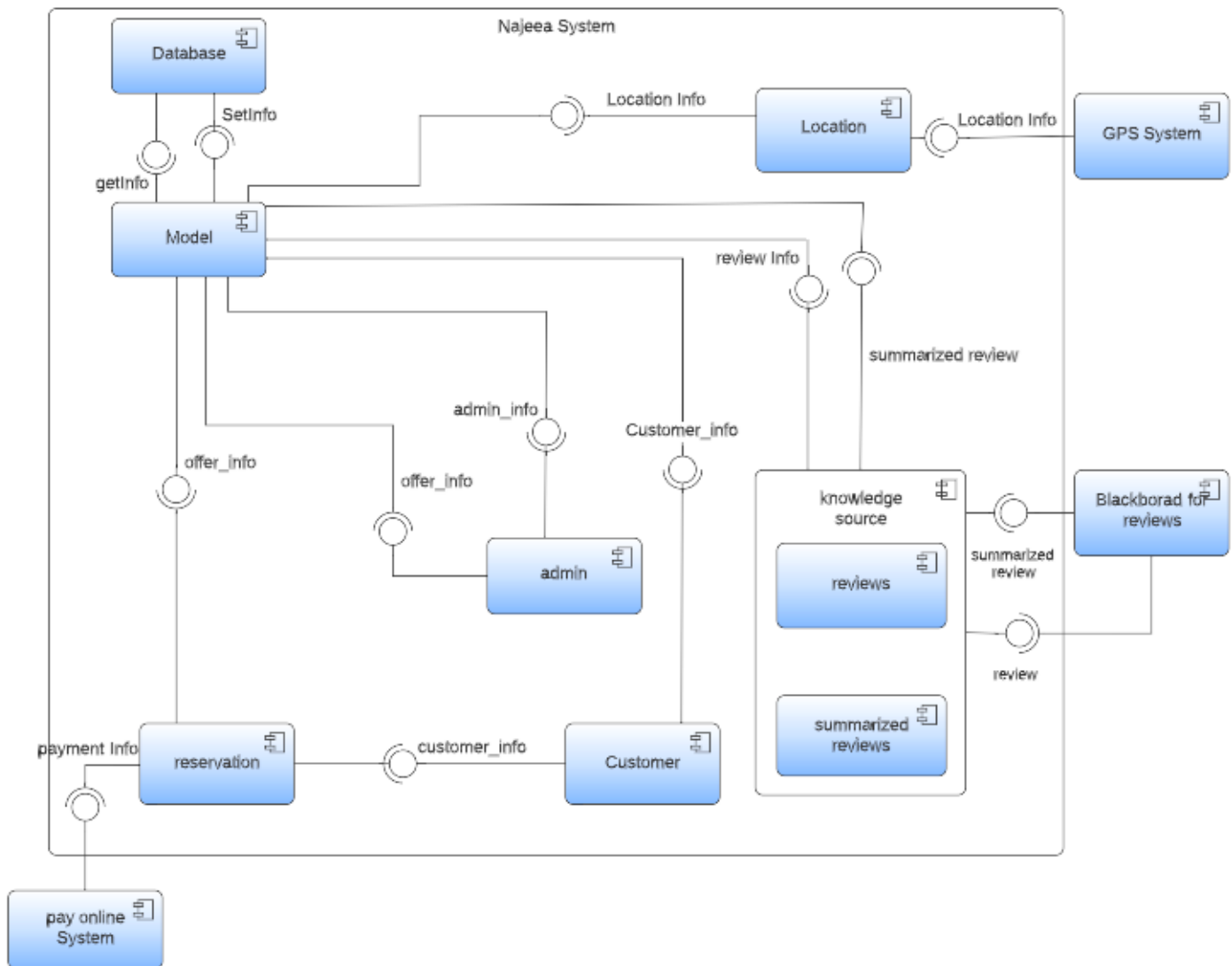


Figure 19: Component diagram that describes the component of our system .

Component Name	Customer
Description	Registered user that can access the system functionalities.
Properties/data	Can access the system functionalities,by providing it data which is (email,full name ,phone number).
Behavior/Functionality	Can view the model and reservation component.
Interface and Connectors	Required interface from model, required interface from reservation. The information will be received as a message since we will connect to the internet to receive it.
Dependencies	It depends on the Model component to get customer information.
Resources	HW device.

Component Name	Model
Description	Responsible for communicating with services (controllers) ,databases.
Properties/data	can access the database of the system,most application data will go through it.
Behavior/Functionality	Deals services and directs the business logic of the application.
Interface and Connectors	1-Required interface from knowledge source(summarized review) 2-Required interface from database(get_info) 3-provide interface to reservation (offer_info) 4-provide interface to customer(customer_info) 5-provide interface to admin(admin_info) 6- provide interface toLocation(location_info) 7-provide interface to database(set_info) 8-provide interface to knowledge source(review info) . dealing with database JDBC connectors,message networked connectors.
Dependencies	It depends on the knowledge source to get a summarized review.
Resources	-

Component Name	Database
Description	contain all system data
Properties/data	save all application data in an organized way Ex:(customer info, all offers made, our services and reviews).
Behavior/Functionality	responsible to provide this data to other components and make all systems related.
Interface and Connectors	Required interface from model(set_info),provide interface to model(get_info). JDBC connectors,massage networked connectors.
Dependencies	It depends on the Model component .
Resources	Database server

Component Name	Location
Description	A public space generally open and accessible to people.
Properties/data	make the application more efficient to give customers what they need(e.g., gym, healthy restaurant, healthy market) .
Behavior/Functionality	shows the customer location by GPS and gives you the coordination.
Interface and Connectors	Required interface from model, required interface from GPS System. The information will be received as a message since we will connect to the internet to receive it.
Dependencies	It depends on the Model component to get location information and the GPSSystem component to get the current location .
Resources	-

Component Name	reservation
Description	It is a reservation that customers can choose and pay for it .
Properties/data	ask customers for some information for reservation and deal with payment issues(price, period of subscribe,bank account)
Behavior/Functionality	The component should know the offer that customer asks for and get it with price and type of offer.
Interface and Connectors	Required interface from customer(customer_info),Required interface from model(offer_info),provide interface to pay online system(payment_info)..The information will be received as a message since we will connect to the internet to receive it.
Dependencies	It depends on the Model component to get offers information and the customer component to customer information and payonline system to get payment information.
Resources	-

Component Name	admin
Description	Person that has an authentication to add an offer .
Properties/data	can add an offer and update and add the application content rapidly (offer info) .
Behavior/Functionality	Can view the model.
Interface and Connectors	Required interface from model(admin_info) , provide interface to model(offer_info).The information will be received as a message since we will connect to the internet to receive it.
Dependencies	It depends on the Model component to get admin information.
Resources	HW device

Component Name	review
Description	It's a text written by customers reflecting their opinion about a certain service.
Properties/data	It's have a opinion about specific service which is we will use it to make the application more efficient (the data is the text of review it self)
Behavior/Functionality	This component saves all the opinions of customers .
Interface and Connectors	Required interface from model(review_info),provide interface to Blackboard for reviews(review).The information will be received as a message since we will connect to the internet to receive it
Dependencies	It depends on the Model component to get the customer's review.
Resources	-

Component Name	summarized review
Description	a combination of customer's opinion of a certain location that includes quality of the service, price.
Properties/data	make it easier for customers to choose by reading a combination of customer's opinions .
Behavior/Functionality	The component is responsible for analyzing customer reviews and generating a summary.
Interface and Connectors	Required interface from Blackboard for reviews(summarized review) ,provide interface to model (summarized review).The information will be received as a message since we will connect to the internet to receive it
Dependencies	It depends on the review component to get the customer's review
Resources	-

Component Name	GPS System
Description	navigation system consisting of a constellation of satellites for monitoring and control.
Properties/data	save the allocating location to location component .
Behavior/Functionality	The component is responsible for allocating location to a map.
Interface and Connectors	Provide an interface to location (location_info).The information will be received as a message since we will connect to the internet to receive it
Dependencies	-
Resources	-

Component Name	Blackboard for reviews
Description	a globally shared database which is used for storing the summarized review as it's being updated by knowledge source
Properties/data	Stores the summarized review.
Behavior/Functionality	Stores the summarized review while it's processed by a knowledge source.
Interface and Connectors	Required interface from knowledge source(review) ,provide interface to knowledge source (summarized review).The information will be received as a message since we will connect to the internet to receive it
Dependencies	It depends on the knowledge source to get reviews.
Resources	-

Component Name	pay online System
Description	Fast & Easy Integration! Reduce Friction & Speed Up Checkouts. Increase Your ROI with a Seamless Payment Solution.
Properties/data	make payment easier , it's to get customer data for payment (CVV ,bank account) depending on payment method.
Behavior/Functionality	The component deals with reservation to get information of method and price , service .
Interface and Connectors	provide interface to reservation (pyment_info).The information will be received as a message since we will connect to the internet to receive it
Dependencies	-
Resources	-

14.b Detailed structural model:

14.b.1 Class diagram:

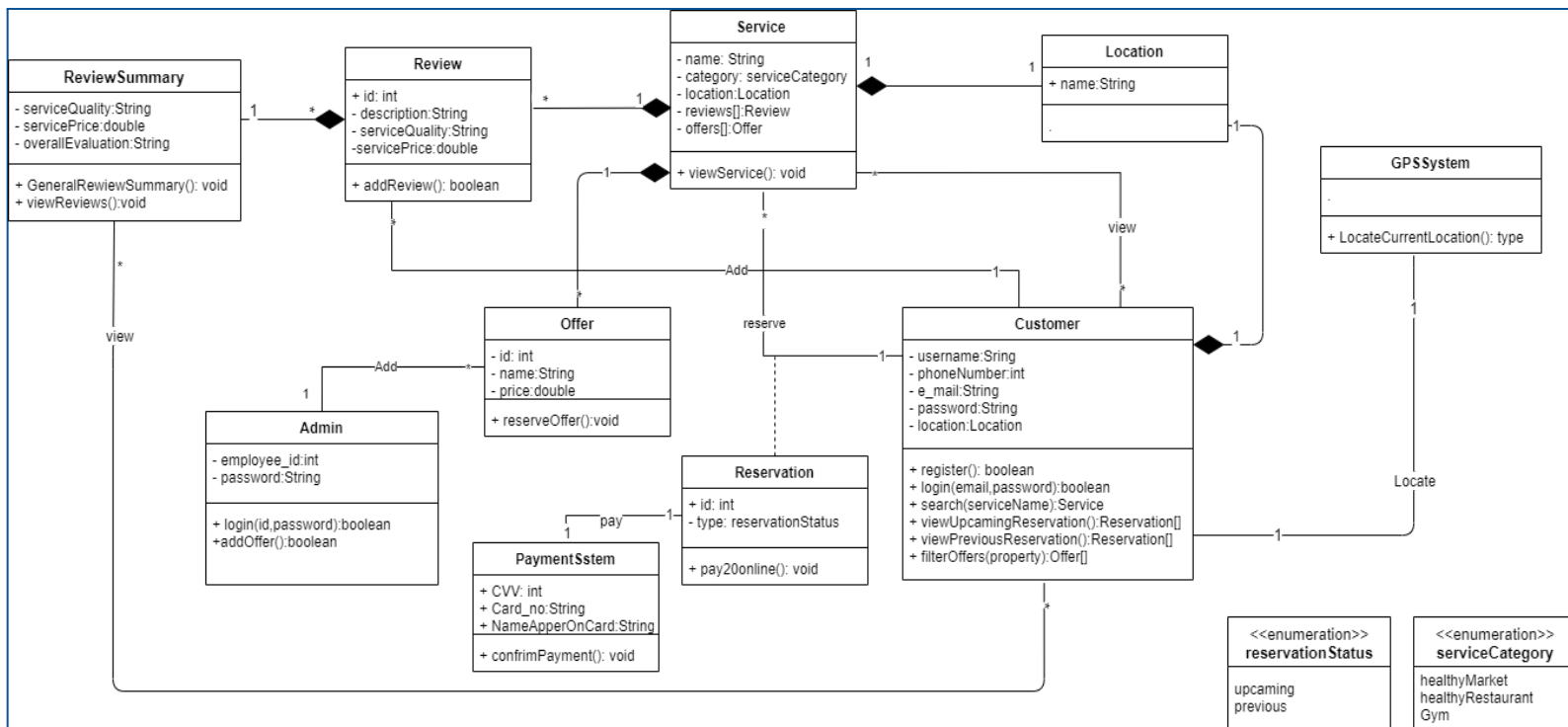


Figure 20: Class diagram that describes the classes of our system .

14.b.2 VOPC Diagrams:

14.b.2.1 VOPC diagram for use case #1 Add review

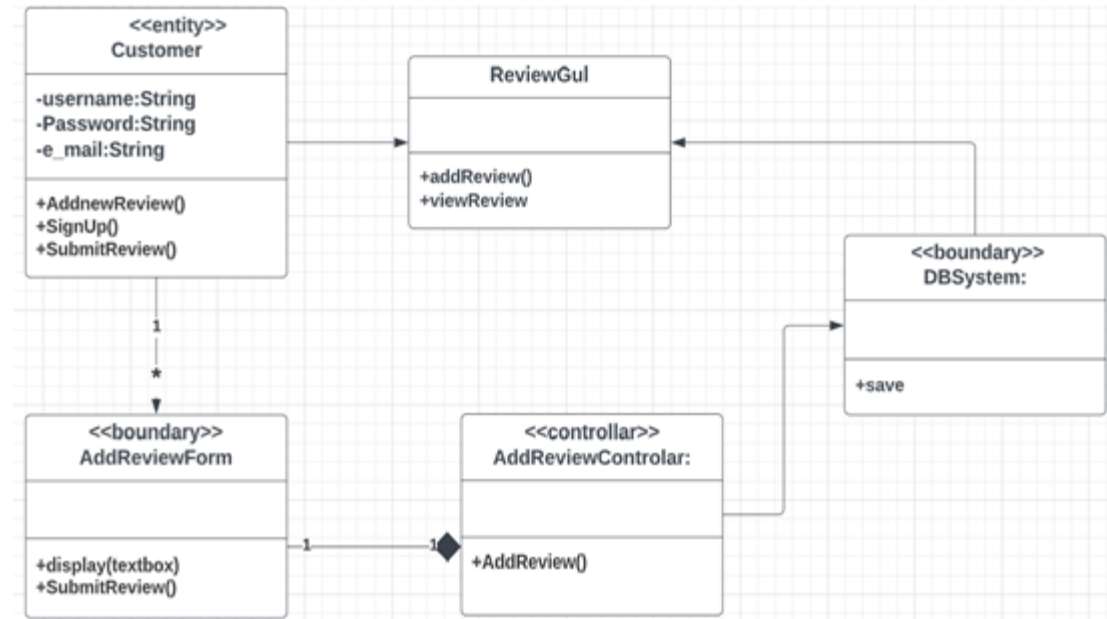


Figure 21: VOPC diagram for add review use case.

14.b.2.2 VOPC diagram for use case #3 search for offers

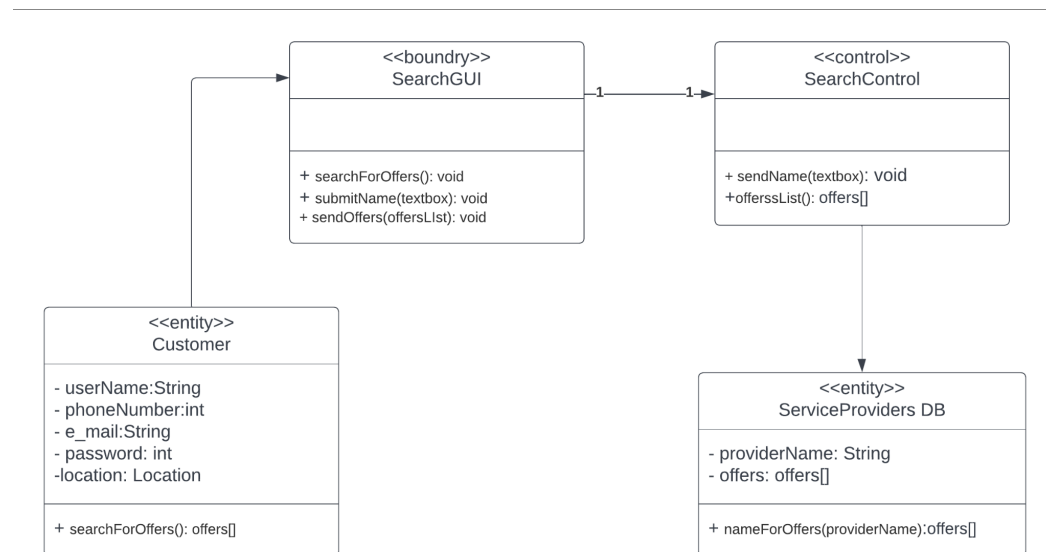


Figure 22:VOPC diagram for search for offers use case.

14.b.2.3 VOPC diagram for use case #4 AddOffer

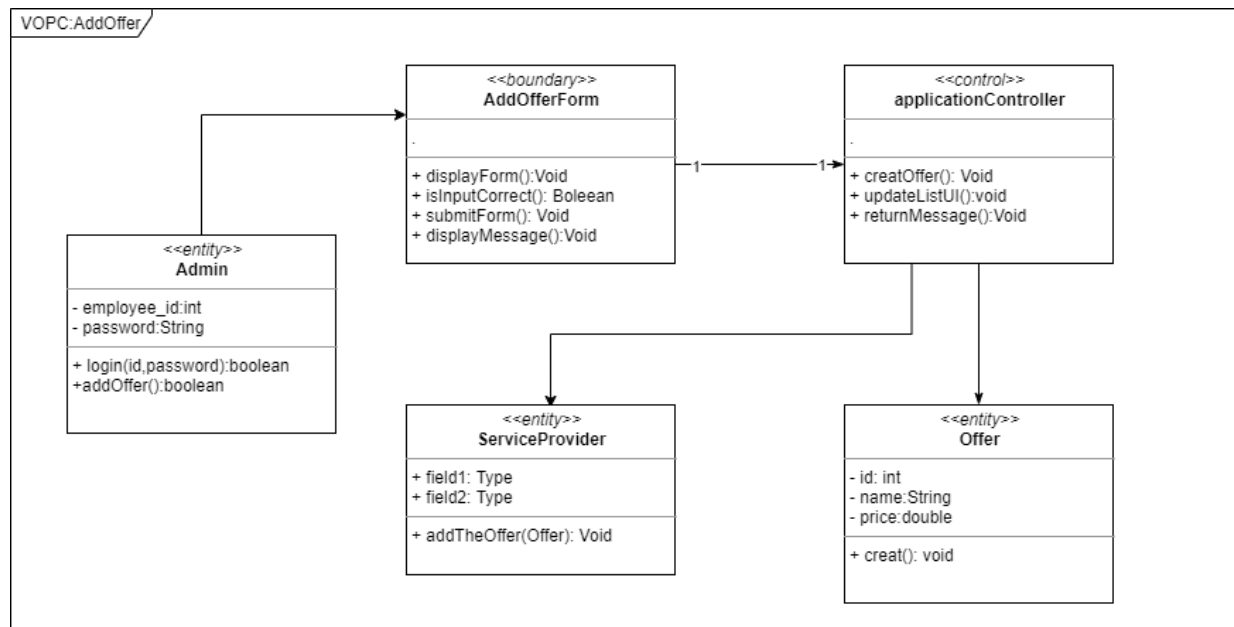


Figure 23:VOPC diagram for add offer use case .

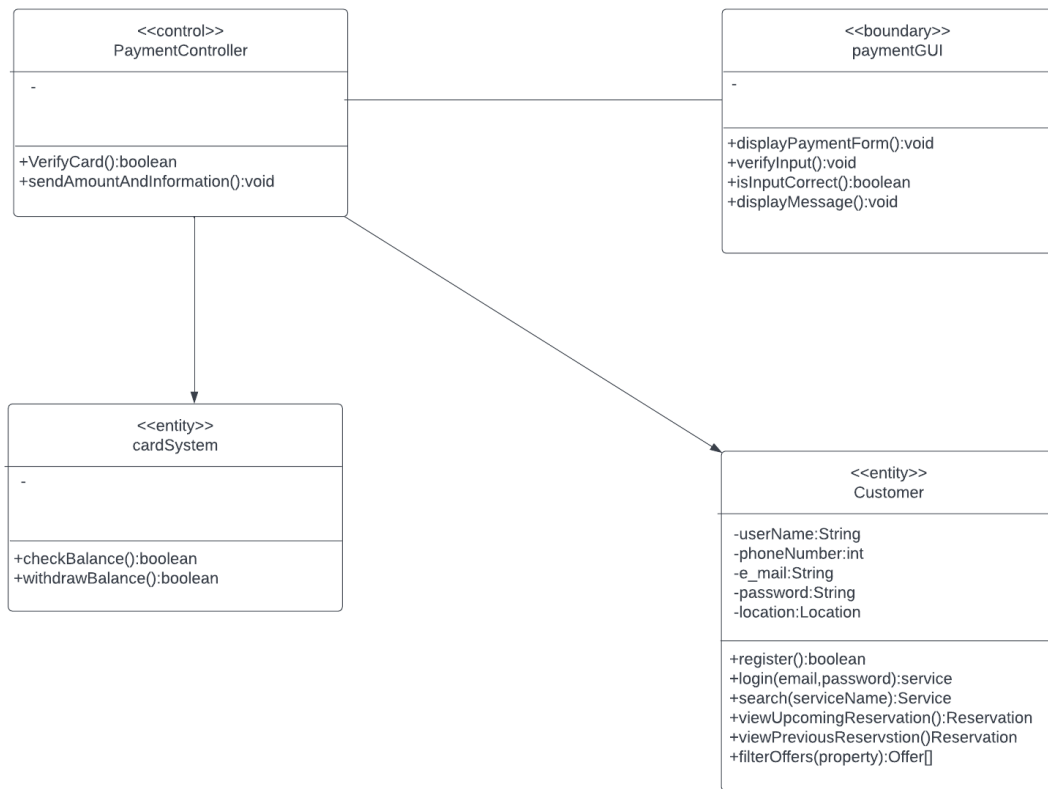


Figure 24: VOPC diagram for pay online use case .

14.c Dynamic model:

14.c.1 State Diagrams:

14.c.1.1 State diagram for use case #1 Add review

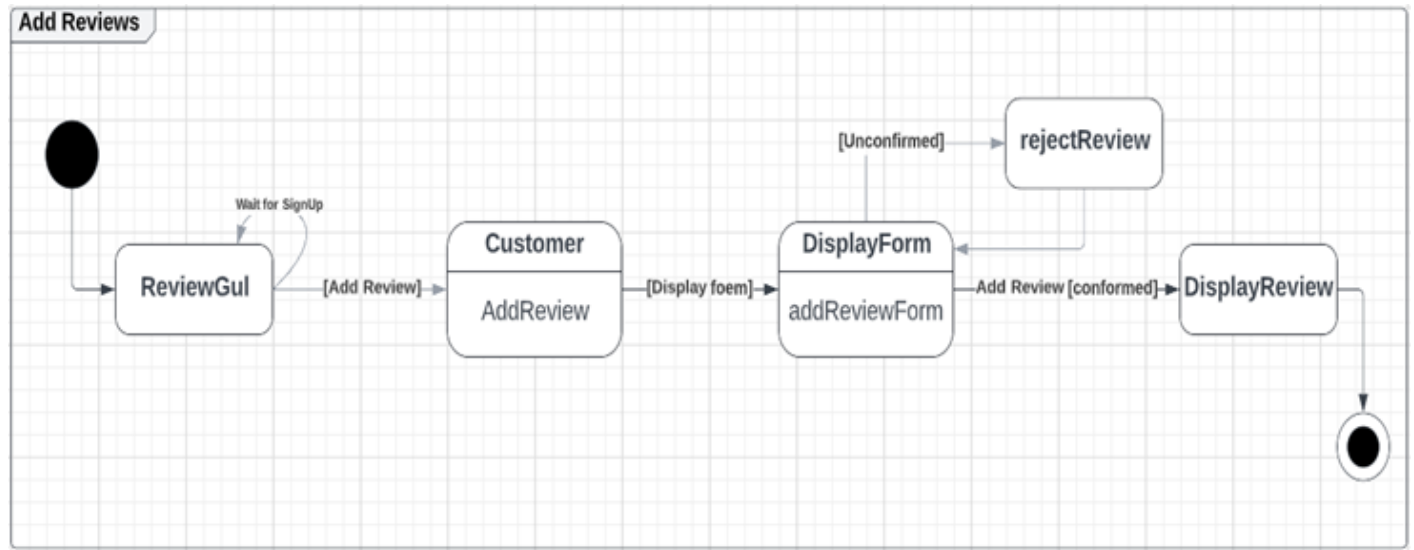


Figure 25: state diageam for add review use case.

14.c.1.2 State diagram for use case #2 pay online

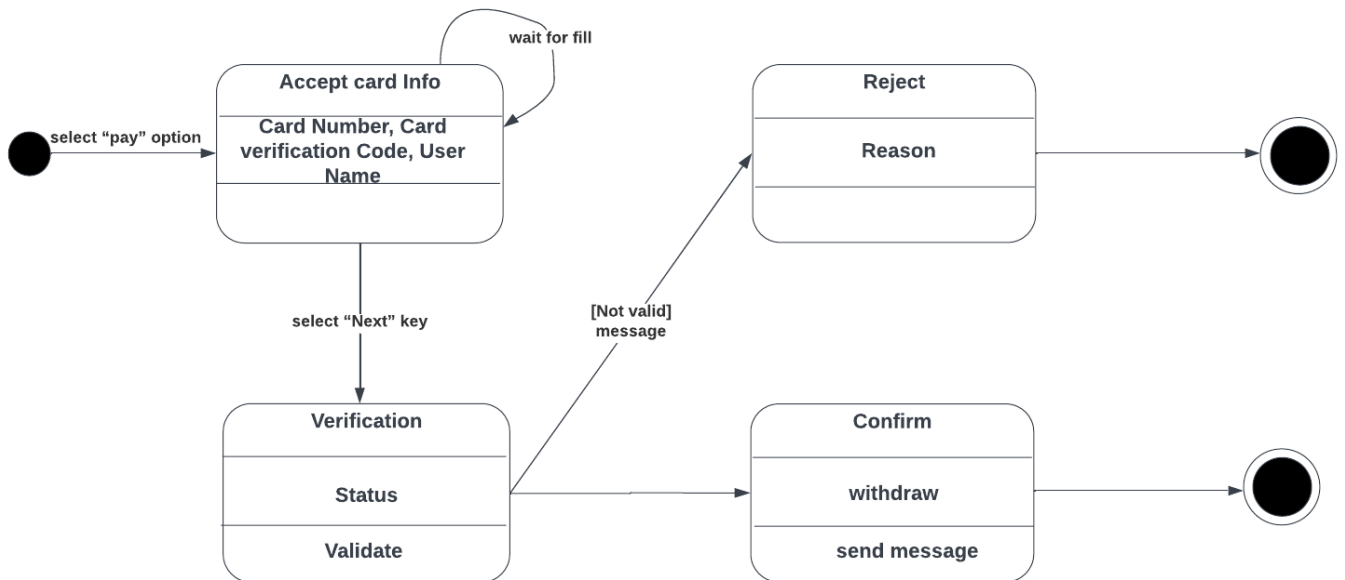


Figure 26: state diagram for pay online use case.

14.c.1.3 State diagram for use case #3 search for offers

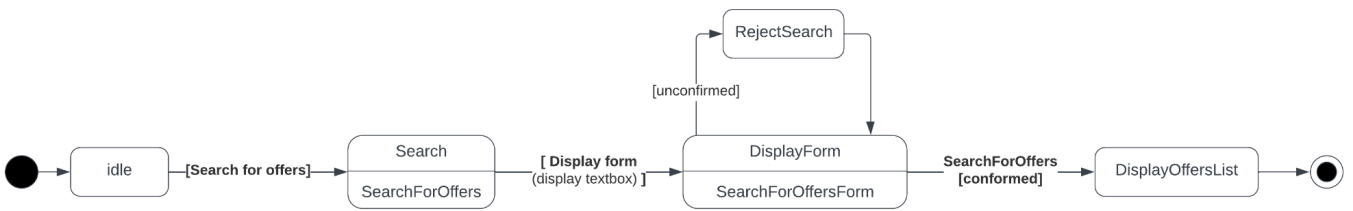


Figure 27: state diagram for search for offers use case..

14.c.1.4 State diagram for use case #4 AddOffer

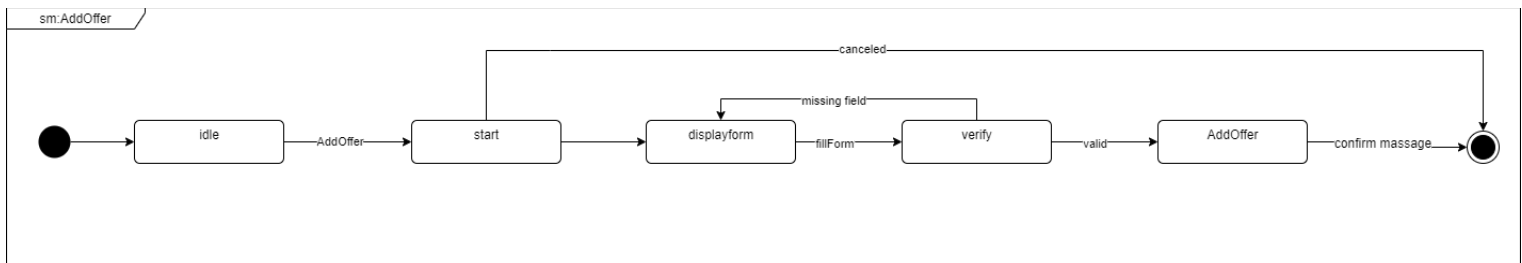


Figure 28: state diagram for add offer use case.

15. Implementation Consideration:

In this section, a mapping of system architecture into an implementation artifact is described. Since Najea application will initially be deployed on Android OS, the integrated development environment (IDE) to be used is the android studio, Java as the programming language, and Firebase as the database.

Each component in Najea is implemented as an Activity that may have multiple Fragments. Connectors between components are implemented as Intent used to start another activity while carrying any necessary data to that activity.

Considering that Najea will be available in both Arabic and English languages, all strings need to be translated to Arabic in XML files.

Java is an Object-Oriented programming language. It is used to develop desktop and mobile applications, big data processing, and embedded systems. Java is used in all kinds of applications like Mobile Applications (Android is Java-based), desktop applications, web applications, client-server applications, enterprise applications, and many more. As Najea we are a mobile application that makes java an excellent programming language. Compared with C++, Java codes are generally more maintainable and faster than python. ^[12]

Firebase is a mobile platform that helps you develop high-quality apps and grow your user base. Firebase comprises complementary features that you can customize to fit Najea's needs. It is easier to explore and integrate Firebase services in your app directly from Android Studio using the Assistant window. Firebase gives the tools and infrastructure from Google to help you develop the app, for example, Analytics, that will help measure user activity and engagement with free, easy, and unlimited analytics. Cloud Messaging delivers and receives messages and notifications reliably across the cloud and devices. Authentication to sign in and manage users easily, accepting emails, Google Sign-In, Facebook, and other login providers. Finally, Realtime Database Store and sync data in real-time across all connected clients. ^[13]

The Google Maps Platform is a set of APIs and SDKs that allows developers to embed Google Maps into mobile apps and web pages or retrieve data from Google Maps. In Najea, we will use maps SDK for Android and Places SDK for Android to locate the nearest place to the user. When the user wants to add their address, the application will display the maps using Google Maps SDK via getting an API key to be added in the Manifest XML file, and any configuration needed. ^[12]

The algorithm that Najea uses is the Sorting Algorithm, which will be used to rearrange and filter the search of a given array or list of elements according to the comparison operator on the features. It will also use a Searching Algorithm because it is designed to check for an element or retrieve a part from any data

structure where it is stored. Finally, Najea will use Dijkstra, it is used in google maps, to find the shortest paths from the source vertex to all other vertices in the graph. ^{[14][12]}

A linked list is a sequence of data structures connected via links, it is a sequence of links that contain items. Each link has a connection to another link. The Linked list is the second most-used data structure after array. In Najea we decided to choose a linked list data structure to complete the insertion and deletion operation that the Najea app Provides to give the admin the complete flexibility on adding or deleting an offer quickly. ^[15]

Finally Najea will use Text Summarizer Extractive Summarization for the review summary. In this process, we focus on the vital information from the input sentence and extract that specific sentence to generate a summary. There is no generation of new sentences for summary, they are exactly the same that are present in the original group of input sentences.

16. Architecture Analysis/Testing:

Based on the output quality assurance, we will analyze the system architecture and make sure the quality assurance tests. We have made sure that the test is not from the same person responsible for her work so that the evaluation is more accurate and correct. Each person from the team checks the work of the other

Quality assurance:

16.a Reviews.

Auditing is the primary way to check the quality of work and architecture so that the reason we chose the MVC architecture is consistent with the outcome. In our project we will conduct two types of reviews: walkthroughs

Step-by-step instructions are an informal review conducted by colleagues so that each person checks the other's work, for example, and whether it meets all the requirements.

Checklists are lists of items that must be checked during a review. Checklists are useful to support how to guide and are easy to use. We will prepare the required checklists (for code review, code is tracked and logical or contextual errors detected) and we will evaluate them throughout the development process

16.b verification.

Verification is the process of evaluating the program to determine whether the products of a particular development stage meet the imposed conditions, assuming that the whole team tracks the output from the diagrams, and makes sure that the desired is reached and the idea is clarified. In this way we answer the question: Are we building the product correctly?

One of the most important tests is to ensure that the nonfunctional requirements are met. For example, among the non-functional requirements, access permissions to system data may not be changed except by the system data administrator.

Passwords should never be viewable at the point of entry or at any other time.

We will, as a team, make sure that the system does not break passwords, and in the event that it displays an error, we must modify it.

We will review each requirement as such.

16.c Validation

After the validation test is completed. We will use a contract review to make sure each condition is met.

17. Deployment and Mobility:

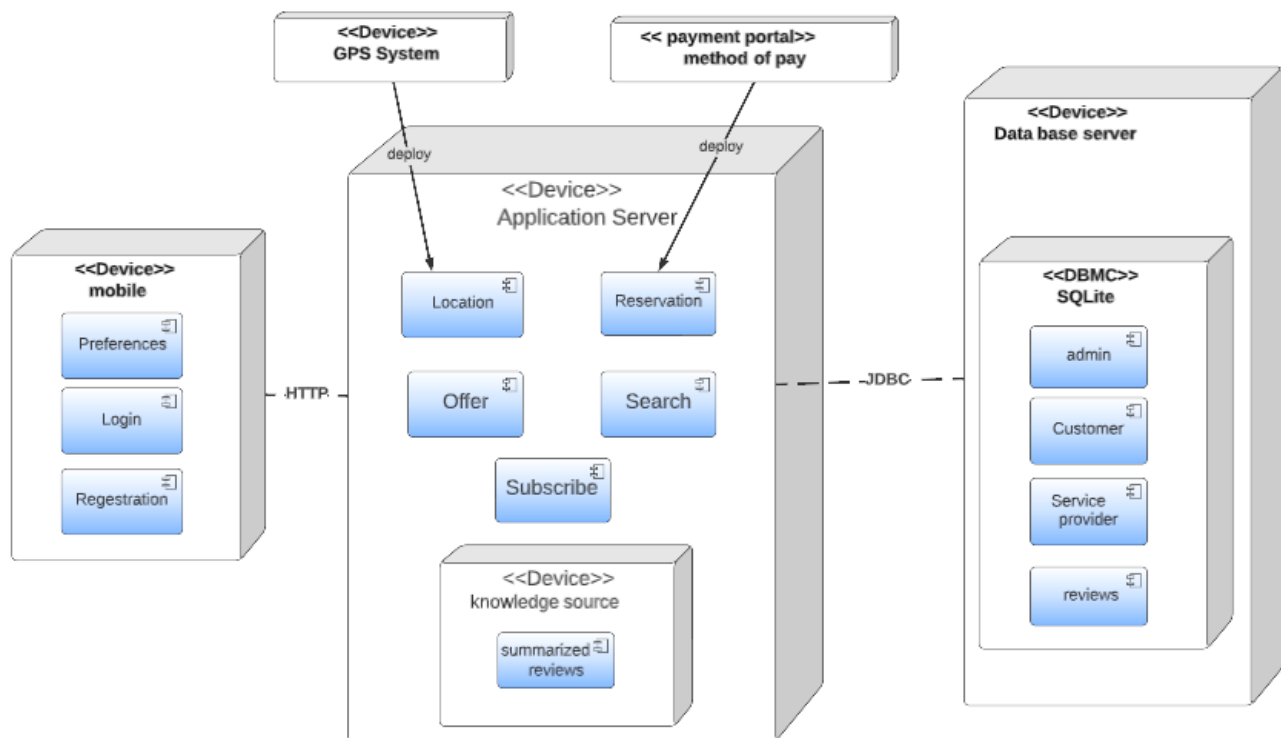


Figure 29: Deployment diagram that describes how to deploy our application. First we have a mobile device which will contain the first steps to go with the application . Second we have an application server which contains the most effective component in our system , third we have DB and we use an SQLite db with entity component.

18. Personal Reflection:

During this project the skills that we have acquired is the ability to be flexible, patient, and teamwork to motivate and influence the team not by adopting a specific decision, but we consult, and each of us tries to convince the other of his decision and reasons and then choose the appropriate one from them. This helped us a lot in expanding our ideas and perceptions. As software architects, we may have the authority to make a high-level decision. Hence, interpersonal skills to resolve conflicts as they arise helps resolve the blocker in the team. This enabled us to develop a Software Design Description(SDD) with no bias and opportunity to read widely on the techniques and practices of (SDD) practitioners.

We have learned that software architecture is a strategic role that builds the core foundation of systems. In addition to core technical skills, a software architect must possess soft skills to run the product engineering team smoothly. Following soft skills are required for the software architects to function effectively. In addition, we know that technology is changing at lightning speed so some of the quality attributes such as performance, scalability, and maintainability of software products are not good-to-have but must-have requirements in current times. A software architect needs to be adaptable as technology and market trends require in the course of product development.

We feel more comfortable architecting systems now than we did before, architecture stability and flexibility, this is a balance where we want an architecture to be stable (it is a foundation for the software product) but also flexible (to support future additions and enhancements). Refactoring is an agile technique used to continuously restructure your code to keep it maintainable. It will not change the external behavior of the product.

By choosing to follow the MVC architecture model we were able to develop a clear structure. In the beginning, this was something we struggled with as we want to increase the performance, and while our system may increase the functionality ability the MVC is Easier to maintain or modify, also the connection between parts should be Loosely Coupled MVC achieve that will improve performance . This is certainly something we have found challenging but it is something we have enjoyed and will continue to apply in future.

One skill we have learned in the course is paying attention to detail. We have learned that because of the nature of engineering practice, especially during drawing diagrams, it is crucial to pay attention to all details no matter how insignificant they may appear. Another skill that we have learned during the course is critical thinking. Critical thinking skills we have acquired include making observations, analyzing those observations, identifying problems and responding to emerging situations. Critical thinking skills also help when choosing the right architecture for a specific system.

Understanding Problem-solving is an important architectural, design, and development skill .Its your ability to handle challenging or surprising situations. Good problem-solvers can stay calm when they encounter obstacles and assess all their options to find the best solution.

References:

- [1]ZanfinaSvirca,2020 ,Everything you need to know about MVC architecture,[online],<<https://towardsdatascience.com/everything-you-need-to-know-about-mvc-architecture-3c827930b4c1>>
- [2]Erin Doherty, 2020 ,MVC Architecture in 5 minutes, [online],<<https://www.educative.io/blog/mvc-tutorial>>
- [3]2020,What are the challenges with MVC architecture?,[online],<<https://www.oreilly.com/library/view/design-patterns-and/9781786463593/777bb79f-6aa1-48f9-9b5d-855a4e744149.xhtml#:~:text=The%20challenges%20with%20MVC%20architecture%20are%20as%20follows%3A,be%20scaled%20as%20a%20whole.>>>
- [4]Advantages and Disadvantages of Service-oriented-architecture, [online],<<https://techspirited.com/advantages-disadvantages-of-service-oriented-architecture-soa>>
- [5]Paul Krill,2011,MVC development gets a new life in mobile apps,[online],<<https://www.infoworld.com/article/2625145/mvc-development-gets-a-new-life-in-mobile-apps.html>>
- [6]draw.io. (13.7.9), Seibert-media. Accessed: Mar. 29, 2022. [Online]. Available: <https://www.draw.io/index.html>.
- [7] Guru99. 2022. What is WHITE Box Testing? Techniques, Example & Types. [online] Available at: <<https://www.guru99.com/white-box-testing.html#2>> [Accessed 2 April 2022].
- [8]GeeksforGeeks. 2022. Software Testing | Security Testing - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/software-testing-security-testing/>> [Accessed 2 April 2022].
- [9]Guru99. 2022. Performance Testing Tutorial: What is, Types, Metrics & Example. [online] Available at: <<https://www.guru99.com/performance-testing.html#3>> [Accessed 2 April 2022].
- [10] AltexSoft. 2022. Acceptance Criteria for User Stories: Purposes, Formats, Examples, and Best Practices. [online] Available at: <<https://www.altexsoft.com/blog/business/acceptance-criteria-purposes-formats-and-best-practices/>> [Accessed 2 April 2022].
- [11]Kai Qian, Xiang Fu, Lixin Tao, Chong-wei Xu, 2010, Software Architecture and Design Illuminated.
- [12] GeeksforGeeks. 2022. *GeeksforGeeks | A computer science portal for geeks*. [online] Available at: <<https://www.geeksforgeeks.org/>> [Accessed 16 May 2022].
- [13] Firebase. 2022. *Firebase Realtime Database | Store and sync data in real time*. [online] Available at: <<https://firebase.google.com/products/realtime-database>> [Accessed 16 May 2022].
- [14] Eduonix Blog. 2022. *5 Algorithms Every App Developer Should Know and Understand*. [online] Available at: <<https://blog.eduonix.com/marketing/5-algorithms-every-app-developer-know-understand/>> [Accessed 16 May 2022].

[15] Tutorialspoint.com. 2022. *Data Structure and Algorithms - Linked List*. [online] Available at: <https://www.tutorialspoint.com/data_structures_algorithms/linked_list_algorithms.htm> [Accessed 16 May 2022].