ADDIS ABABA UNIVERISITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

FOOD DELIVERY ANDROID APPLICATION

CoSc4411_: FINAL PROJECT

BY

NAME OF STUDENT	ID
NAHOM TESFAYE ……………………………………………….UGR/1797/12	
SOLOMON ASHAGRE ………………………………………..UGR/5156/12	
YOHANNES FIKIRE ……………………………………………UGR/5191/12	
TADESSE TSEGA …………..UGR/7317/12	

PROJECT ADVISOR: Mrs. FIKIRTE G.

FEB 2023

This is to declare that this project work is done under the supervision of Mrs. FIKIRTE G and having the title "Food delivery android application" is the sole contribution of

1. NAHOM TESFAYE

2. SOLOMON ASHAGRE

3. YOHANNES FIKIRE

4. TADESSE TSEGA

No part of the project work has been reproduced illegally (Copy and Paste) which can be considered as plagiarism.

We will be responsible and liable for any consequence if violation of this declaration is proven.

Date _____

Group Members

Full Name   Signature

_____   _____

_____   _____

_____   _____

CERTIFICATE

I certify that this final project entitled "Food delivery android application" by:

1. NAHOM TESFAYE

2. SOLOMON ASHAGRE

3. YOHANNES FIKIRE

4. TADESSE TSEGA

is approved by me for submission. I certify further that, to the best of my knowledge, the report represents work carried out by the students.

_____ _____

Date Name and Signature

ACKNOWLEDGMENT

ACRONYM AND ABBRIVATION

SRS -- Software Requirements Specification.

HTTPs -- Hypertext Transfer Protocol

SSL --- Secure Sockets Layer

SSH --- Secure Shell

Table of contents

List of tables

List of figure

1. CHAPTER ONE-INTRODUCTION

1.1.   INTRODUCTION AND OVERVIEW

With the emergence of modernity, technology is playing a vital role in helping people accomplish a certain work. Nowadays, most people rely on internet for a large number of reasons. The apps have made a lot of things convenient for the users. At present, we can't neglect the fact that people are having a busy lifestyle. So, most of the times people end up ordering food from restaurants that have delivery services. Every other person chose the faster way of accomplishing a task and one of this is they prefer to order food online rather than cooking at home.

Online Food delivery system is the process of ordering food neither having to go to the restaurant nor calling to the restaurant. It is a simple and convenient way for customers to purchase food. This is why; Food Ordering android App is going to be developing. The drive of this dissertation is to discover the possibility of coming up with a solution for people to be able to order food online and get delivery services. An online food delivery system is an application that stimulates the foodies (customers) to put food orders through internet by locating the restaurant It is known globally that, in today's market, it is extremely difficult to start a new Small-scale business and live-through the competition from the well-established and settled owners. In fast paced time of today, when everyone is squeezed for time, the majority of people are finicky when it comes to placing a food order. The customers of today are not only attracted because placing an order online is very convenient but also because they have visibility into the items offered, price and extremely simplified navigation for the order.

Mobile Food Ordering Application is the key to solve the problem regarding to manual listing of orders. Using this application, the customers need not go to the restaurant by themselves, but they can order the dishes through   Android mobiles anywhere.

Online ordering system here, greatly simplifies the ordering process for both the customer and the restaurant. System presents interactive and up-to-date menu with all available options in an easy-to-use manner. Customer can choose items to place an order which will land in the Cart. Customer can view all the order details in the cart before checking out. At the end, customer gets order confirmation details. Once the order is placed it is entered in the database and retrieved in pretty much real time.

This allows Restaurant Employees to quickly go through the orders as they are received and process all orders efficiently and effectively with minimal delays and confusion. The project is mainly meant to explore the feasibility of solving the given problem through the implementation of an android application that will allow for users to order food online from different restaurant using the android app and get delivery services.

1.2.   STATEMENT OF THE PROBLEM

 When peoples want to get a food, they should either visit the nearby hotels and restaurants physically to know about food items, place the order or make the payment. or over direct phone call. This type of getting foods and drinks causes various problems for both the customers and the restaurant owners. For that there should be enough labor to take the order over phone call to offer rich dining experience and process the payment. In today's market, labor rates are increasing day by day making it difficult to find employees. In this system the orders may not be prepared timely and this frustrates customers who have tight schedule. Ordering via phone call can cause receiving of incorrect orders because orders can be mixed up.in this system when customers call there is a chance that their calls is not answered this could be troublesome for the customers.   it is a waste of time for the customer to go and get a service.   The restaurants should have some spaces   to give their services for their customers; this by itself raises two main issues. the leading one is it requires a large space and the restaurants should have many employees. The second one is the profits the restaurants can get is highly depends on the no of rooms and number of spaces they have despite their quality services.

This project   gives the customer comfort easy, fast and secure services from the restaurants. This changes telephony system to online and virtual system. It gives the restaurants   unlimited customers to increase their profits. Make the process a one click away. it decreases a land resource which is needed otherwise, by making the system virtual. This project decreases the number of employee and rooms the restaurant's needs. The system greatly simplifies the ordering process for both the customer and the restaurant and also greatly lightens the load on the restaurants.

1.3. OBJECTIVE OF THE PROJECT

1.3.1 GENERAL OBJECTIVE

The general objective of this project is to design and develop food delivery android application.

1.3.2 SPECIFIC OBJECTIVES

Its specific aim is to simplify and improve the efficiency of the ordering process for both customer and restaurant, minimize manual data entry and ensure data accuracy and security during order placement process. Customers will also be able to view product menus and there ingredients and be able to have a visual confirmation that the order was place correctly. This website is developed to help computer science students to learn about the Web application to build a complete working application. Following givens are the specific aims of this project:

- ➢ Reduce time-consuming phone orders and eliminate illegible fax orders.
- ➢ No more busy phones or the requirement for extra phone lines.
- ➢ An edge over the competition at an affordable price.
- ➢ Broader customer reaches across regions.
- ➢ Provides a channel for marketing and promotion lowering your advertising cost.
- ➢ Builds a customer database.
- ➢ Helps in improved service
- ➢ Greater customer satisfaction
- ➢ It shows the correct menu and enables the customers to order items that are available.
- ➢ To avoid long queues at the counter.
- ➢ To accommodate huge amount of orders at a time.
- ➢ To improve the communication between the client and the server
- ➢ To allow users to view the food catalogue and search for food items.
- ➢ Enable people to install the application on their smartphones.
- ➢ To register new users or customers.
- ➢ To allow customers to order using the application.
- ➢ To allow customers to view or check their delivery status.

1.4. PROJECT SCOPE

The scope of this project is to develop android mobile application that will help everyone in Addis Ababa to order food online with easy and secure way. The project to be developed will address searching of restaurant and ordering of the foods. Anyone who has an account can search restaurant nearby and order the food and drinks what they want. It is a mobile app which can be downloaded to one's phone and the owner can create an account for further process of selecting

and ordering of favorite foods and beverages. The customer or the user of this food delivery app can rate see the restaurants on the integrated application.

In this project the customers cannot track the cars which come towards them. It does not support local languages. And also, it's only an android app it has not web version.

1.5. SYSTEM DEVELOPMENT METHODOLOGY

In this project, an object-oriented system analysis and design methodology is applied because it has got many advantages compared with structured analysis and design. The most important ones are flexibility to change, easy transition from one phase to the next, easy traceability and it suggest greater user involvement. We use Iterative software development model. While this project starts, the client's system requirements are clearly defined and fully understood. However, because additional requirements may arise during the design and development process, an iterative software development method is chosen and followed to provide flexibility by closely working with the customer. We select this model because it is less costly to change scope and requirements that can be easy to handle new requirements which arise during system development.

1.6. INVESTIGATION (FACT-FINDING) METHODS

We will use different methods of investigation or fact-finding methods to collect information from the user and hotel owners.

Document Review:

To obtain further information about our project context, we will review existing documents, such as current worldwide food delivery services, procedures and processes that the client is following. That will help us understand the client's requirements and needs in a more organized and strategic way.

Interview: we'll interview peoples about the problems they face when they want to get the services of the restaurants and hotels.

Discussion: we shall discuss and ask the enterprises (restaurants, hotels) about how they run and manage the system

1.7.   SYSTEM DEVELOPMENT TOOLS

The problems stated so far will be addressed from writing the documentation to implementing the project on android phone. And we'll use the following programming language and /or technologies.

Tools used for documentation

- ➢ Microsoft office 2019
- ➢ Figma: user interface maker
- ➢ Tools for implementation
- ➢ Flutter: Flutter is an open-source UI software development kit created by Google. It is used to develop cross-platform applications for Android, iOS, Linux, macOS, Windows, Google Fuchsia, and the web from a single codebase. First described in 2015, Flutter was released in May 2017
- ➢ Laravel: Laravel is a free and open-source PHP web framework, created by Taylor OTWELL and intended for the development of web applications following the model–view–controller architectural pattern and based on SYMFONYREDUX: a predictable state container designed to help write JavaScript apps that behave consistently across client, server, and native environments, and are easy to test.
- ➢ Google API: it's an API for map
- ➢ Firebase: it's one of Google service which enables us to access database and authentication.

1.8.   SIGNIFICANCE OF THE PROJECT

- ➢ The significance of this project is listed below
- ➢ it provides a complete sales channel for the restaurant.
- ➢ restaurant can use it as a tool for generating more profits and
- ➢ It also allows restaurant owners to save on labor costs and restaurant space needed to serve such customers.
- ➢ the distant of the restaurants and customers will be minimized.
- ➢ It organizes or brings many restaurants together for the better
- ➢ It saves time and labor for both customers and restaurants owners
- ➢ Make the service more reliable and flexible by considering changes
- ➢ greatly lightens the load on the restaurants

- it will lower expenses, boost profits, improve customer satisfaction and Improved Marketing Cycle

Our group will learn some new programming languages and/or technologies for developing mobile apps. It develops sense of team work and interpersonal communication

1.9. BENEFICIARY OF THE PROJECT

- Any customer who wants to order food online: anyone who has an app and account is benefitted from this project
- hotels and restaurants owners: this project create a hub and brings interested restaurants to one place and give them an opportunity to give their services through the channel and make profit by cheap cost (labor, time)
- Government: government can collect tax from the services to be delivered. It lowers the controlling and monitoring task of the government.

1.10. TIME SCHEDULE OF THE PROJECT

This Android Application development project work is intended to be started on 25th Dec 2022 and end on 30th of Jun 2023. The detail plan is provided here below.

		Dec 25/22 - Dec 29/22	Jan 01/23 - Jan 20/23	Feb 02/23 - Mar 15/23	Mar15/23 Jun01/23	Jun 02/23 – Jun 15/23	Status of the project (%)
1	Project proposal	▬					
2	Requirement Analysis		▬				
3	Design			▬			
4	Implementation				▬		
5	Installation and testing					▬	

Table 1.1 Time schedule

2. CHAPTER TWO - REQUIREMENT ANALYSIS
2.1. INTRODUCTION

Online food ordering system targets to solve the problems existed between customers and restaurants owners. Because of its ability to create a sales channel for the restaurants it enables them to reduce labor cost and restaurant space needed to serve such customers. Restaurants can use the system to organize and generate more profits than ever. This enables them to bypass the competition and getting closer to customers. This system is beneficial for food aggregators or food marketplace. Because this system is a hub for various restaurants and restaurants can share and post their food outlets to order from. When the customers visit the platform they can explore and get restaurant. And then order the restaurant based on what they want. This saves the life of the customers because instead of ordering the food from individual restaurant system it's better to go a restaurants hub, choose and order the food.it simplifies the process of receiving orders and eliminate human related errors and long delivery times. This extremely saves the time and day of the customers. This system enables the customers to browse menu items, choose based on their prefer and make payment to the restaurants app. Upon receiving the order, the restaurant prepares the food and delivers to a specific location. This helps the restaurants in reaching a big share without requiring them to come to the establishment.

2.2. CURRENT SYSTEM

In the current system most of the restaurants throughout Addis Ababa have no online delivery system. Anyone who wants to get food can do either of the two methods. the first one is they can directly go to the preferred restaurant, check the menus with corresponding prices and then tell the waiter based on what they prefer. After a while the waiter comes back with the ordered meal and drinks. The second one is that the customer can call via telephony and check whether or not the favorite foods and drinks are there. Then the customer can either go to the restaurant and get served or the meal is ready to be picked up upon arrival. There is no means of knowing the existed foods and drinks at the specific restaurants. Restaurants should have a sufficient space for their services to be accessed. The customer has no knowledge of restaurants around. Instead, randomly choose and serviced by a restaurant

2.3. PROBLEMS IN THE EXISTED SYSTEM

Placing orders via phone calls or actually present and get served in the restaurant has shown to be cumbersome task and become an additional burden to the customers. Marked problems of this system are as follows:

Even though orders were called beforehand Common issue is food is not prepared or ready for pickup upon arrival. This could be a frustrating thing for the customers who have tight schedule.

Most of the time customers expressed their frustration of receiving incorrect orders. Although expressing the orders explicitly vials phone calls, there is a chance of orders to be mixed up. This is usual during peak hours which are troublesome because no one is interested in receiving incorrect orders.

Most of the times customers complained that their calls are not answered. And this is not only for customers but it is inconvenient for the restaurants. Customers can't get their favorite meal and restaurants are missing out on sale.

It might not seem like it takes staff too much time to answer orders coming from customers. After all it takes 5 minutes on average per order. But when these calls come frequently, they can create an almost full-time job. And this can distract from other restaurant works.

2.4. REQUIREMENTS GATHERING

Requirement gathering methodology

We've used some methods to an insight about our system. We observe to the current system and tried to understand what and how the current system works. We had interviews with restaurant owners who have both telephony system and online system. We also had some tips and fact from customers. We reviewed existing systems that found in abroad such as McDonald's Mobile ordering App and within our country such as Deliver Addis Ordering App and users comments on these systems on the internet. Comments from Google Play store and review others work like us were our major source of requirements that our Food Delivery Android Application fulfill.

Results found

From our observation, reviews of current apps and interviews of customers and restaurant owners on existing systems   we have found that this kind of system is somehow troublesome, not efficient and accurate. It's not organized and it's time consuming for either of the sides. There is no way to sure that customer's orders are placed correctly and not mixed up with other orders. Customer's call for the orders may not be responded well because of overwhelming phone calls coming from the customers. It requires quite large labor forces and spaces to run the system.

2.5. PROPOSED SYSTEM

Overview

The proposed system is an android application which solves shortcomings of the current working trends of restaurant system in Addis Ababa.

2.6. FUNCTIONAL REQUIREMENTS

In this section we'll define the functional requirements of our proposed system which is the relationships of the system and the environment.

For the customers

- Create an account
- Manage their account
- Log in to the account
- Choose a restaurant
- Navigate the restaurant's menu
- Select an item from the menu
- Customize options for a selected item
- Add an item to their current order
- Review their current order
- Remove an item/remove all items from their current order.
- Provide delivery and payment details
- Place an order
- Receive confirmation in the form of an order customer sacrifice

For the app owner

- Add a new/update/delete vendor to/from the menu.

For the restaurants

- Add a new/update/delete food category to/from the menu.
- Add a new/update/delete option for a given food item
- Update price for a given food item
- Update default options for a given food item
- Update additional information (description, photos, etc.) for a given food item.

2.7. NON-FUNCTIONAL REQUIREMENT

Nonfunctional requirements describe features of the system that are not directly related to the functional behavior of the system. Instead, non-functional requirement deals with the quality. of the application needed to be developed from different evaluation point of view. Accordingly, the non-functional requirements of the system are listed below. Non-functional requirements are requirements that define 'how' the app must perform a certain function. In essence, they are the quality attributes of an app that define the user experience of the app. Non-functional requirements of a food delivery android application could include:

- Performance requirements (e.g. response time, throughput)

- Scalability (e.g. ability to handle a large number of users and orders)

- Security (e.g. protection of sensitive user data)

- Compatibility (e.g. compatibility with various versions of android)

- Usability (e.g. ease of use for customers and delivery drivers)

- Maintainability (e.g. ability to easily update and fix bugs in the application)

- Reliability (e.g. availability of the application during peak usage times)

- Compliance (e.g. adherence to relevant laws and regulations)

2.8. USER INTERFACE AND HUMAN FACTORS

User Interface: The app should have an intuitive, easy-to-learn user interface that allows users too quickly and easily place orders for food delivery. A user interface (UI) for a food delivery app has prioritized ease of use and a smooth ordering process for the user. This can be achieved through a clean design, clear and intuitive navigation, and simple and straightforward ordering forms. Additionally, the app should provide real-time updates on the status of the delivery and allow users to track their order

In terms of human factors, the app should take into account the user's context, such as their location and previous orders, to make personalized recommendations and streamline the ordering process. The app should also be designed to minimize errors and allow for easy corrections if mistakes are made during the ordering process. Additionally, providing user support and allowing for easy communication with the delivery driver can improve the overall user experience..

2.9. DOCUMENTATION

The documentation is SRS documentation. The purpose of this SRS is to outline both the functional and non-functional requirements the whole development process starting from the proposal will be well documented. While implementing the system, the source code will also be

well commented so that maintainers will be able to understand easily how the whole Application works.

This document specifies the requirements for a restaurant paper menu and ordering replacement strategy to alleviate the problems associated with the current archaic method, his SRS should convey and confirm the required functionality and represent contractual agreement between the involved parties.

2.10. HARDWARE CONSIDERATION

When we developing this food delivery android application, some hardware considerations to take into account include:

- Processor: The device will have a powerful processor to handle multiple tasks and run the application smoothly.

- Memory: The device will have enough memory to store the application and handle large amounts of data, such as menus and customer information.

- GPS: The device should have GPS capabilities to track the location of delivery drivers and provide accurate estimated delivery times to customers.

- Camera: The device should have a good camera to take pictures of food items for the menu.

- Battery life: The device should have a long battery life to ensure it can be used for an entire delivery shift without needing to be charged.

- Durability: The device should be durable enough to withstand the rigors of daily use in a delivery setting.

- Network Connectivity: The device should have good network connectivity to ensure the application can function properly and transmit data quickly.

- Screen size: A larger screen size would be beneficial for navigation and tracking deliveries

2.11. PERFORMANCE CHARACTERISTICS

Performance characteristics for a food delivery android application may include:

1. Fast loading times: The app should quickly load menus, customer information, and other data to ensure a smooth user experience.

2. Reliability: The app should function properly and consistently, with minimal downtime or errors.

3. Ease of use: The app should be easy to navigate and use, with a clear and intuitive interface.

4. Security: The app should protect sensitive customer information, such as credit card details, to ensure data privacy and security.

5. Real-time tracking: The app should provide real-time tracking of orders, so customers can see when their food is expected to arrive.

6. Push notifications: The app should send push notifications to customers to keep them informed about their order status.

7. Scalability: The app should be able to handle a large number of orders and customers without slowing down or crashing.

2.12. ERROR HANDLING AND EXTREME CONDITIONS

- Error handling:

1. Frontend error handling – This could include catching incorrect input in registration forms, handling network errors in the app, or debugging issues related to the UI.

2. Backend error handling – This could include catching and resolving server errors, responding to failed API requests, or detecting errors related to the database.

3. Notifications service error handling – This could include retrying failed notifications, managing rate limits and backpressure, or monitoring notifications for quality control.

- Extreme conditions:

1. Offline processing – The ability to queue up orders or other requests and process them in the background when a user is offline.

2. Low latency response times – Ensuring that the app responds quickly even in areas with spotty reception or high levels of network traffic.

3. Real-time data updates – Implementing systems that ensure orders and other data is up to date in real time, even when the device is offline.

4. Network failure recovery – Building in automatic retries to requests that failed due to network issues and robust error handling policies.

5. Automated testing – Automating the testing process to ensure the app works correctly in a variety of environments and use cases.

2.13. QUALITY ISSUES

Food delivery applications have quality assurance methods in place to ensure that all food delivered is safe and secure. The app should have adequate security protocols that prevent fraud and protect customers' data. The app should also be tested thoroughly to ensure that all functions of the app are working properly before they are released to the public. Additionally, there should be a system to ensure that all orders are fulfilled with high quality and fresh ingredients in a

timely manner. Finally, customer feedback should be taken into consideration when making changes or updates to the app.

The application should be available 24 hours a day and 7 days a week unless Internet connection or electric power is down.

System Modification

It is possible to modify a food delivery android application to offer additional features to customers. Here are some potential system modifications that could be made:

1. Allow customers to schedule deliveries in advance: Customers can plan and schedule their delivery times in advance of ordering their meals.

2. Adding a Loyalty program: Loyalty programs can be implemented that reward customers with discounts, vouchers, or other rewards for repeat orders.

3. Offer customized meal plans: Customers can choose from pre-defined meal plans or create their own with specific dietary preferences or dietary restrictions.

4. Allow customers to track their orders: Real-time tracking and notifications via text, email, or push notifications can be implemented to keep customers informed of the status of their orders.

5. Allow customers to rate and review restaurants and dishes: Customers can rate and review their orders and the restaurants to which they ordered from.

6. Add an automated payment system: Automated payment systems such as Apple Pay and PayPal can be integrated into the system to simplify the payment process for customers.

2.14. PHYSICAL ENVIRONMENT

The physical environment for food delivery android applications includes the location of the restaurants and customers, as well as the transportation infrastructure that connects them. This could include things like roads, sidewalks, and bike lanes for delivery drivers, as well as the availability of GPS technology for tracking and navigation. Additionally, the reliability and speed of internet connectivity may also be important factors in the overall functionality of the app.

2.15. SECURITY ISSUES

Security and Privacy: The app should ensure the privacy and security of customer personal information, payment information and order information. Appropriate encryption, authentication and other security measures should be implemented on user data.

There are several security issues that can arise with food delivery Android applications. Some of the main concerns include:

1. Data breaches: Personal information, such as credit card details and home addresses, can be stolen if the app's security is compromised.

2. Unauthorized access: Hackers may gain access to the app's backend systems and manipulate order details or steal sensitive information.

3. Insecure communications: If the app does not use secure communication protocols, such as HTTPS, it may be vulnerable to eavesdropping and man-in-the-middle attacks.

4. Malicious software: The app may be infected with malware or other malicious software that can steal personal information or damage the device.

5. Unsecured third-party components: If the app uses third-party libraries or components, they may have security vulnerabilities that can be exploited by attackers.

Solution:

To mitigate these risks, it is important for food delivery apps to use secure coding practices, regularly update their security measures and use encryption to protect sensitive data. Also, they should conduct regular security audits and penetration testing

2.16. RESOURCE MANAGEMENT

Resource management for food delivery android applications includes managing and optimizing various resources such as network connections, battery usage, device storage, and memory usage. Here are a few best practices for resource management in food delivery android applications:

1. Use efficient network connections: Use libraries such as Retrofit or Volley (i.e. **Retrofit** is a networking library that has made it easier for us to use API or web services in our android application) to manage network connections and minimize data usage.

2. Battery optimization: we will Use the battery saver feature in Android to optimize battery usage while the app is running.

3. Device storage management: we will Use libraries such as Glide or Picasso to efficiently manage images, and reduce the storage space required by the app.

4. Memory management: we try to Use libraries such as Leak Canary to detect and fix memory leaks, and optimize memory usage.

5. Reduce App Size: we will Use Pro-guard to reduce the size of the app.

6. Use Caching: we will Use caching to save data and reduce the number of network requests.

7. Use Background Services: we will Use background services to perform tasks in the background, so that the app does not consume too much memory.

8. Use a sync Task: we will Use A sync Task to perform background operations without freezing the UI.

By implementing these best practices, we can ensure that your food delivery android application runs efficiently and does not drain the device's resources.

2.17.   SYSTEM MODELS

2.17.1. Online Food Delivery Android Application Project Use Case Diagram
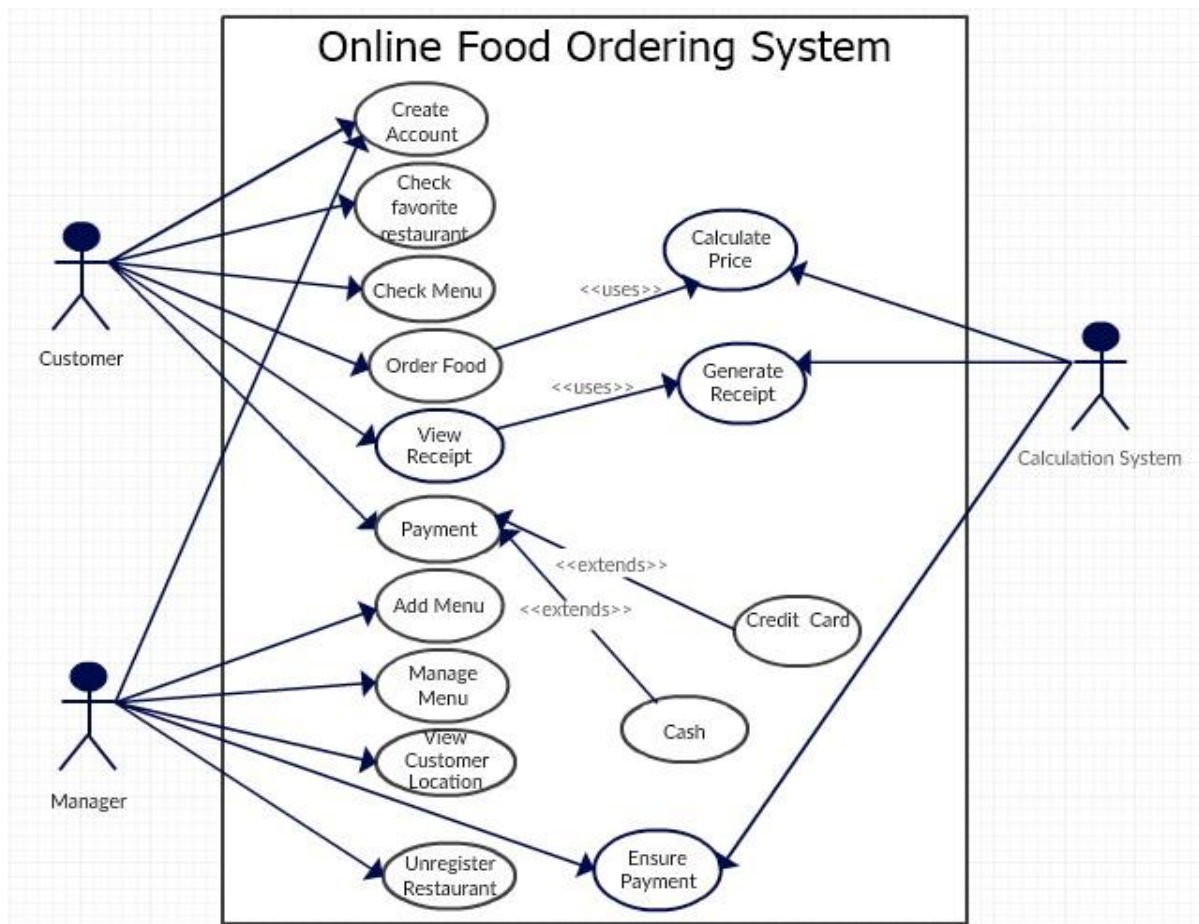
Figure 2-1 Use-case diagram

Use Case No:	1
Use Case Name:	Create Account
Actors	Manager, System administration
Description:	The create account use case allow manager and customer to create their accounts and become a registered member.
Preconditions:	None
Normal Course:	1. The manager and system admin enter the full name, email address, password and name of the restaurant. 2. The system will ask them to choose strong password. 3. The system will ask to reenter the password. 4. The account will be created after click on sign up button.

Alternative Courses:	1. Invalid entry of information
	
	  1.1. The system will show the message to reenter the invalid data.
	
	2. weak password
	
	  2.1. The system will show the message to enter strong password.

	3. Password not match 3.1. The system will show the message to reenter the password.
Post conditions:	1. The manager and customer are now registered. 2. The system displays all features to which customer and manager are associated with as defined in their account. None

Table 2-2 create account

Use Case No:	2
Use Case Name:	Log in
Actors	Manager, system admin
Description:	The Log in use case allow manager and system admin to login into their respective account.
Preconditions:	All should be registered
Normal Course:	1. The actor enters the username and password 2. The system checks the input information with the stored credential in the database 3. The system redirects the user to their respective page

17

Alternative Courses:	2. Invalid entry of information 2.1. The system will show the message to reenter the invalid data.
Post conditions:	The Actors are now logged in

Table 3-2 login

Use Case No:	3
Use Case Name:	Generate Report
Actors	System admin
Description:	Allow the System Administrator to generate report
Preconditions:	The system administrator should be logged in
Normal Course:	1. An actor clicks on "Generate Report" link 2. The System displays generate report button 3. An actor clicks on generate report button 4. The system generates report needed and displays submit report button 5. An actor clicks on the submit report button 6. The system saves the report.
Alternative Courses:	1. If there is no available report 1.1. The system displays message

Post conditions:	

Table 2-4 manage account

Use Case No:	4
Use Case Name:	Manage restaurant
Actors	System admin
Description:	Allow the System Administrator to add and delete restaurant
Preconditions:	The system administrator should be logged in
Normal Course:	1. An actor clicks on manage restaurant button 2. The system display add restaurant and unregister restaurant options 3. The admin clicks on whatever he/she want to accomplish 4. The system redirects to the proper management page
Alternative Courses:	none
Post conditions:	The restaurant is created or removed from the system

Table 2-5 manage restaurant

Use Case No:	5
Use Case Name:	Check Favorite Restaurant

Actors	Customer
Description:	This use case allow customer to search for their favorite restaurants.
Preconditions:	1. The customer should open the android app
Normal Course:	1. Search bar appear on homepage. 2. The customer enters the restaurant name or place name. 3. Now customer click on the search button. 4. The system displays the available restaurant in that area or the restaurant the customer looking for.
Alternative Courses:	1. No Restaurant available 1.1. The system displays the message "sorry no restaurant available in that area." 1.2. The system will show the available restaurants to customer nearby that area.
Post conditions:	1. The system displays the restaurant that customer searched. 2. The system display the nearby restaurant incase searched restaurant not available.
Exceptions:	None

Table 2-6 check favorite restaurants

Use Case No:	6
Use Case Name:	Check Menu
Actors	Customer

Description:	This use case allow customer to check the menu of restaurant
Preconditions:	1. The customer should open the application 2. The customer should search for restaurant.
Normal Course:	1. The customer search for restaurant. 2. The system displays the restaurant section. 3. The check menu button appears at top of restaurant section. 4. The customer clicks on the button. 5. The system displays the menu.
Alternative Courses:	1. No Menu available 　1.1. The system displays the message sorry no menu available for this restaurant.
Post conditions:	The system displays the menu of restaurant.
Exceptions:	None

Table 2-7 check menu table

Use Case No:	7
Use Case Name:	Order food
Actors	Customer
Description:	This use case allow customer to order the food
Preconditions:	The customer should open the android app
Normal Course:	1. The plus + sign available on left side of each item of menu. 2. The customer clicks on + sign. 3. The system ask customer to select quantity of that

	item.
	4. The customer selects the quantity.
	5. The system adds that item to cart.
	6. The customer selects more item from menu and system add them in cart.
	7. The order food button appears on cart.
	8. The customer clicks on the button.
	9. The system displays the message your food has been ordered
Alternative Courses:	7.Deselect item from cart 7.1. The minus – sign appears on the left side of every item in the cart. 7.2. The customer clicks on the – sign. 7.3. The item removed from cart. 7.4. The system displays the message the item is removed.
Post conditions:	1.The system display a message the food has been ordered.
Exceptions:	None

Table 2-8 order food

Use Case No:	8
Use Case Name:	Payment
Actors	Customer
Description:	This use case allow customer to pay for food.
Preconditions:	1. The customer should open the android app 2. The customer should order the food. 3. The customer should have the receipt that calculation system generates for them.

Normal Course:	1. The let's pay button appears on the right side below the receipt. 2. The customer clicks on the button. 3. The system displays tele birr and PayPal option 4. The system ask customer to choose one of them. 5. The customer choose any option.

Alternative
 	
Courses:
	1. pay pal
	   1.1. The system asks for card number, card name and CVV from customer.
	   1.2. The customer enter the credit card info.
	   1.3. The system display payment for verification.
	   1.4. System send payment request to external authorization service system.
	   1.5. System receive payment approval.
	   1.6. System records credit payment

Post conditions:	1. Payment information is saved. 2. Tax calculated. 3. Payment authorization approval recorded. 4. The system displays the message to customer that payment is done.
Exceptions:	None

Table 2-9 payment

Use Case No:	9
Use Case Name:	Add Menu
Actors	Manager
Description:	This use case allow manager to add menu in the table
Preconditions:	1. Manager should be registered. 2. Manager should login with specific username and password.

Normal Course:	1. The manager enters to the add menu form by clicking the Add menu button. 2. The manger will place the new food item in menu box. 3. The manager will set the price of the food item. 4. The manager will set the availability area of food item. 5. The manager will place the picture of the food item. 6. The manager will set any deal offer for that item. 7. The manager will press the enter button to complete the information and to store information in the data base.

Alternative Courses:	1. The other offer
	   1.1. The manager does not allow for any deal of the product.
	   1.2. The manger adds the product with another product in a deal.
	2. The reset buttons
	   2.1. The reset button will erase all the information of the product by the manager

Post conditions:	1. The system should enter the new product item in database. 2. The manager should see all the products
Exceptions:	None

Table 2-10 add menu

Use Case No:	10
Use Case Name:	Manage menu
Actors	Manager
Description:	This use case allow manager to manage the menu of his restaurant.
Preconditions:	1. The manager should be registered. 2. The manager should be entered to the system by his username and password. 3. The manager should be on the front page of the system.

Normal Course:	1. The manager will click on the manage menu button and he/she will be directed to the manage menu form. 2. There he/she will be allow access to choose the option of update or delete food item. 3. The manager will return back to its main page.

Alternative
 	
Courses:
	1. Update Information
	
	    1.1. The system displays the message "Select the item to update information".
	
	    1.2. The manager will select the food item to update its info.
	
	    1.3. The manager will update the information according to his requirements.
	
	    1.4. The manager will enter the update button to complete the update process.
	
	    1.5. The system displays update message to the manger.
	
	2. Delete Information
	
	    2.1. The system asks for item which is to be deleted.
	
	    2.2. The manager will select the desired product to delete its information
	
	    2.3. The manager will click the delete button.
	
	    2.4. The system will display the message "deletion completed''.

Post conditions:	The system will complete the deletion or update process.
Exceptions:	None

Table 2-11 manage menu

Use Case No:	11

Use Case Name:	View customer location
Actors	Manager
Description:	This use case allow manager to view the location of the customer
Preconditions:	1. The manager should be registered. 2. The manager should be entered to the system by his login and password. 3. The customer should order a food.
Normal Course:	1. The manager will click on the view customer info button and he/she will be directed to the customer info page. 2. A notification will be sent to the customer to access his location 3. Then he/she will be allowing access to customer current location

Alternative Courses:	1. Customer response "yes":
	    1.1. The system will send a notification to customer to access his location.
	    1.2. The customer will receive the notification
	
	    1.3. The customer will choose the yes option to give access to the manager
	
	2. Customer response "no":
	    2.1. The system will send a notification to customer to access his location
	
	    2.2. The customer will receive the notification
	
	    2.3. The customer will choose the yes option to give access to the manager.

Post conditions:	1. The system will show the customer location.
Exceptions:	None

Table 2-12 view customer location

Use Case No:	12
Use Case Name:	Unregister Restaurant
Actors	Manager, system admin
Description:	This use case allow manager to unregister the restaurant
Preconditions:	1. The manager should be registered. 2. The manager should be entered to the system by his username and password.

Normal Course:	For the manager
	1. The manager will click on the settings button and he/she will be directed to the settings page.
	2. The manager will click on to the unregister button in the list of different settings
	3. The system will open a dialog box to renter your password.
	 4. The manager will enter the password.
	5. The system will display form with a display message "are you sure to unregister the account".
	6. The system will direct the manager to main page of the system.
	
	For the system admin
	    1. The admin goes to his admin dash board
	    2. Click on restaurants
	    3. Click the three dots of the intended restaurant
	    4. Click on unregister the restaurant
	    5. The system will pop up a message "are you sure you

	want to unregister this restaurant" with yes or cancel option 6. The admin clicks on yes option 7. The restaurant will be deleted from the system
Alternative Courses:	3.1 Enter your Password 1. The manger will enter his password again 2. The manager will click on the submit button 3. The system will send his information for authorization to database to match the record. 1. The system will authorize the manager and let it move further 2. The system deny it access and ask him to reenter his password again a. Customer response "yes": 1. The system will display form with a display message "are you sure to unregister the account". 2. The manager will click on the yes button 3. The system will display the message "account unregistered". a. Customer response "no": 1. The system will display form with a display message "are you sure to unregister the account". 2. The manager will click on the yes button 3. The system will display the message "account unregistered".
Post conditions:	The system will perform the deletion of account action completely.
Exceptions:	None

Table 2-13 unregistered restaurants

Use Case No:	12
Use Case Name:	Log out

Actors	manager, system admin
Description:	Allow the System Administrator and manager to logout from the system
Preconditions:	The manager and the system admin must be logged in to the system
Normal Course:	1. Once the actor is decided to exit from the system, he/she clicks on the logout button 2. The system reassures the actor to logout 3. The system redirects the actor to the default Homepage and terminates the session
Alternative Courses:	1. The user doesn't confirm the logout 1.1. The system cancels the logout process
Post conditions:	The actor is not logged out from the system

Table 2-14 logout

2.17.2.   USE CASE SCENARIOS

A scenario is a tool used during requirements analysis to describe a specific use of a proposed system. Scenarios capture the system, as viewed from the outside, e.g., by a user, using a specific example. Or it's a deliberately informal, open ended and fragmentary narrative depiction of key usage situations happening over time. A usage scenario is a description of a way someone uses an existing product or system.

Scenario No	1
Scenario name	Create Account
Actors:	• manager (Alemayehu)
Flow of events:	1. Alemayehu install and open the android app on their phone 2. The app launches its default page and register tap is appeared 3. Abebe tap on the register and fills on the first name and last name section 4. Then he moves downwards and enters email and password 5. The system ask him to choose strong password. 6. The system will ask to reenter the password. 7. Alemayehu reenter the password 8. He enters the name and the address of the restaurant 9. The account will be created after click on sign up button

Table 2-15 Create account scenarios

Scenario No	2
Scenario name	Log-in

Actors:	• manager (Alemayehu)
Flow of events:	1. Alemayehu open the android app 2. The system opens the very first page 3. Alemayehu navigates to login button and tap it 4. He presented with login page which contains input spaces for entering credentials 5. Alemayehu fills out username and password and clicks the login button 6. The system compares the provided information with the stored credentials 7. If the username and password are correct the system redirects Alemayehu to Its respective page 8. If the username and password is incorrect the system denies access.

Table 2-16 Login scenarios

Scenario No	3

Scenario name	Check favorite Restaurants
Actors:	Customers (Abebe)
Flow of events:	Abebe logged in using his username and password Search bar appear on homepage Abebe enter the restaurant name or the place where the restaurant located Abebe clicks on the search button The system displays the available restaurant in that area or the restaurant the customer looking for If the searched restaurant not available the system displays the message "no restaurant available" The system show the available restaurants to customers nearby that area

Table 2-17 Check favorite Restaurants scenarios

Scenario No	4

Scenario name	Check menu
Actors:	Customers (Abebe)
Flow of events:	Abebe search for restaurant and clicks on it
	The system displays the restaurant section
	The menu button appears on the top section
	Abebe clicks on the button
	The system displays the menu
	If no menu available the system displays the message sorry no menu available for this restaurant

Table 2-18 Check menu scenarios

Scenario No	5
Scenario name	Order food

Actors:	Customers (Abebe)
Flow of events:	Abebe logged in to the system
	He searches for the preffered restaurant
	He again browses through the item's menu
	The plus + sign available on left side of each item of menu
	The customer clicks on the + sign
	The system ask abebe to selecet quantity of that item
	Abebe selet the quantity
	Abebe add that item to cart
	Abebe selects more item and the system add them in the virtual cart
	The order button appears on cart
	Abebe clicks on the button
	The system display the message "your food has been ordered"
	If Abebe wants to deselect the items
	The minus – sign appears on the left side of the item
	Abebe clicks on it
	The item removed from the cart
	The system shows the message the "item is removed"

Table 2-19 Order food

Scenario No	6

Scenario name	payment
Actors:	Customer(Abebe)
Flow of events:	Abebe logged in to the system, select restaurant and menu item
	The customer add the items to the cart and the calculation system calculates and generate the receipt
	The lets pay button appears on the right side of the receipt
	The customer clicks on this button
	The system displays the options of credit card and cash on delivery
	The system ask to choose from
	Abebe choose one option
	If Abebe chooses cash on delivery the system the confirm order button appears at below
	The customer clicks on button
	The system display a waiting message for the customer
	If Abebe chooses credit cart Abebe is asked for card number, card name and cvv
	Abebe enter the credit info
	The system depicts payment verivication
	The sytem reach out authorization service rof payment request
	The system receive payment approval
	The system stores credit payment

Table 2-20 payment

Table 2-21 add menu

Scenario No	8
Scenario name	Add menu
Actors:	Manager (Alemayehu)
Flow of events:	Alemayehu logged in to the system using his credentials Alemayehu clicks the Add menu button and access the add menu form Alemayehu place the new food item He set the price of the new food item He set the area wher the item is available He set the picture of the new item He set any deal offer for the new item He presses the enter button for completion and store the data on the database If he wants, he presses the reset button and eliminate all the data from the database and see what he enters

Scenario No	9
Scenario name	Manage menu
Actors:	Manager (Alemayehu)
Flow of events:	Alemayehu clicks on manage menu button
	The system directed Alemayehu to the manage menu form
	The system offers the option of update menu item and delete menu item
	If Alemayehu clicks on update menu item the system ask him to press the item to update
	He selects the item to update the info
	Update the information's of the items based on the requirements
	The manager clicks on update the item
	The system displays the update message to the manager
	If Alemayehu clicks on delete button
	The system ask him the item which is to be deleted
	The manager selects the item to be deleted
	The manager clicks on delete button

	The system displays the message deletion completed

Table 2-22 manage menu

Scenario No	10
Scenario name	View the location of the customer
Actors:	Manager (Alemayehu)
Flow of events:	The manager clicks on the view customer info button and directed to the customers info page
	A notification sent to a customer with option of yes and no to access the location
	If the customer's response is yes, the manager has an access of customer location
	If the customer's response is no the manager can't access or know the location of the customer

Table 2-23 view location of the customer

Scenario No	11
Scenario name	Unregister restaurant
Actors:	Manager (Alemayehu)
Flow of events:	The manager enters in to the settings of his profile
	The manager selects unregister button among other list of different settings
	The system asks the manager to enter the password
	Alemayehu enter the password
	The system show Alemayehu a message are "are you sure unregister the account"
	The manager responses with yes
	The system cross check the entered information with the stored credentials in the database
	If the entered and the stored informations are match the system remove the account
	If the informations are not match the system ask him to reenter the password again

Table 2-24 unregistered restaurants

Table 2-25 track order

Scenario No	12
Scenario name	Track orders
Actors:	Customer (Abebe)
Flow of events:	Abebe opens the food delivery app and chooses the "Track Order" option.
	Abebe is presented with a list of orders placed by him.
	Abebe selects the desired order from the list.
	Abebe is presented with a detailed description of the order, including order status, delivery time, etc.
	He can also make changes to the order, if required.
	He taps the "Track Order" button to track the order.
	He can track the delivery status of the order in real time.

Table 2-26 logout scenarios

Scenario No	13
Scenario name	Logout
Actors:	Customer (Abebe)
Flow of events:	Once a customer is ready to exit the system. He/she press on the logout button The system redirects him/her to the home page where the user's session is terminated

2.18. DATA DICTIONARY

TABLE	FIELD	DESCRIPTION	TYPE
Customer	CustomerId	Identification of the customer	string
	phone	Phone of the customer	string
	Username	Name of the customer	string

	Email	E-mail using	string

Table 2-27 customer dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
Manager	Id	Identification of manager	string
	Username	Name of manager	string
	Address	Address of manager	string
	RestName	Name of Restaurant	string

Table 2-28 manager dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
Payment	PaymentID	Identification of payment	string
	OrderID	Identification of order	string
	CustomerID	Identification of customer	string

Table 2-29 payment dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
Delivery Person	DeliveryId	Identification of delivery	string
	Item	Item	string
	Quantity	Quantity of product	Int
	Customer	Customer	string
	Datereceivable	Date of receivable	Date

Table 2-30 delivery dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
	OrderId	Identification of order	string
	ItemtId	Identification of the product	string

	Orderdate	Date of order	Date
ORDER	Customername	Name of customer	string
	Quantity	Quantity	Int
	Itemname	Name of Item	string
	Companyname	Name of company	string
	Deliverydate	Date of delivery product	Date
	Date	Date of order	Date

Table 2-31 order dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
	OrderId:	Id of the order	string
	Order name	Name of the order	string
ORDER DETAIL	Item id	Id of the item	String
	Item name	Name of the item	String
	Unitcost	Price of the unit item	decimal
	Subcost	Price of total item	decimal

Table 2-32 ORDER DETAIL dictionaries

TABLE	FIELD	DESCRIPTION	TYPE
	menuId	Identification of menu	string
	menucategory	Category of menu	string
MENU	menuname	Name of menu	string
	menudescription	Description of menu	string

Table 2-33 Menu dictionaries

2.18.1. Online Food Delivery Android Application Project Class Diagram

Class diagram

A class diagram is a unified modelling language that describes the structure of the system by showing the system's class, their attributes, operations and the relationships among objects
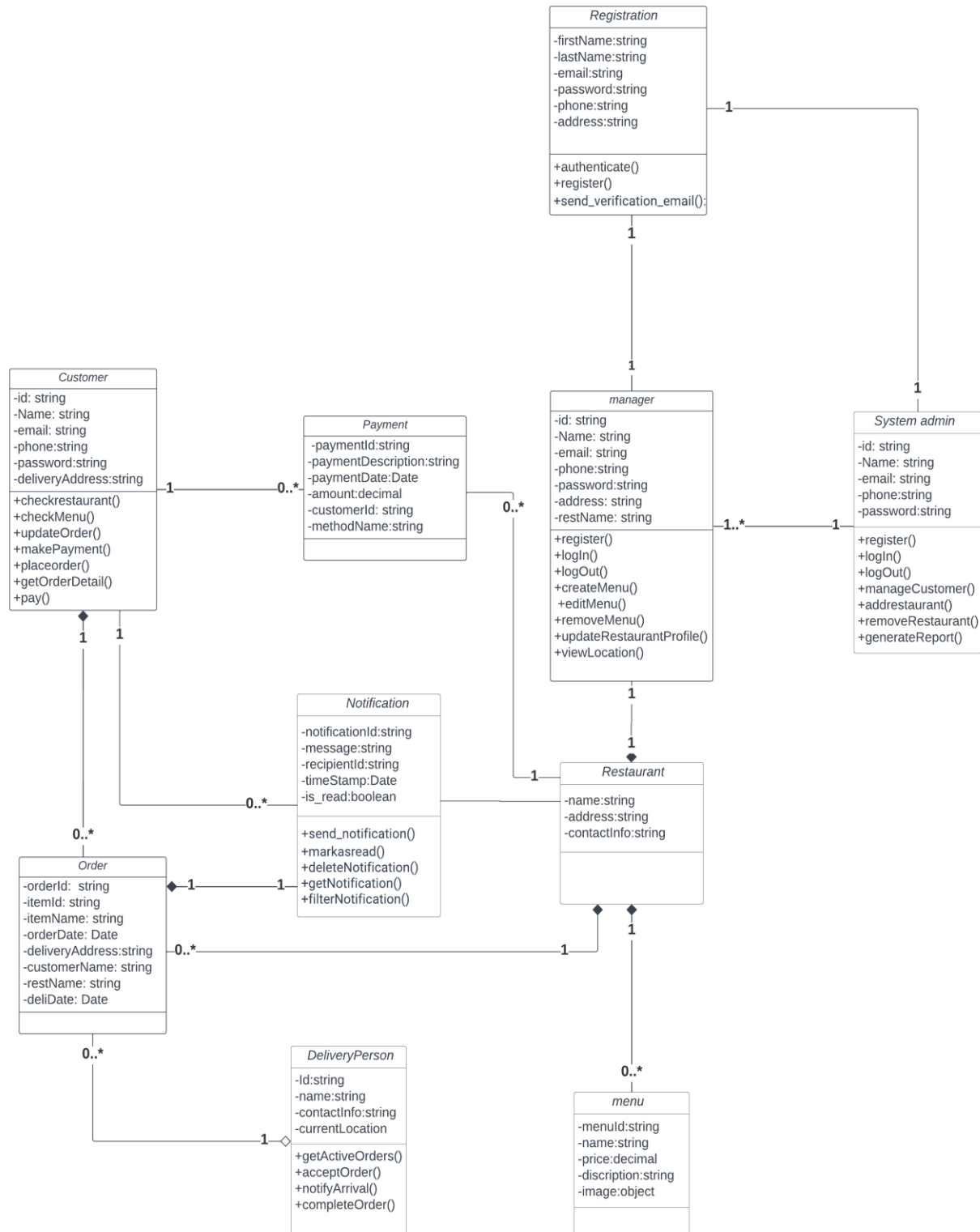
Figure 2-2 Class diagram

2.19. DYNAMIC MODELLING

2.19.1. Online Food Delivery Android Application Project Sequence Diagram

A sequence diagram is an interaction diagram that shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations.
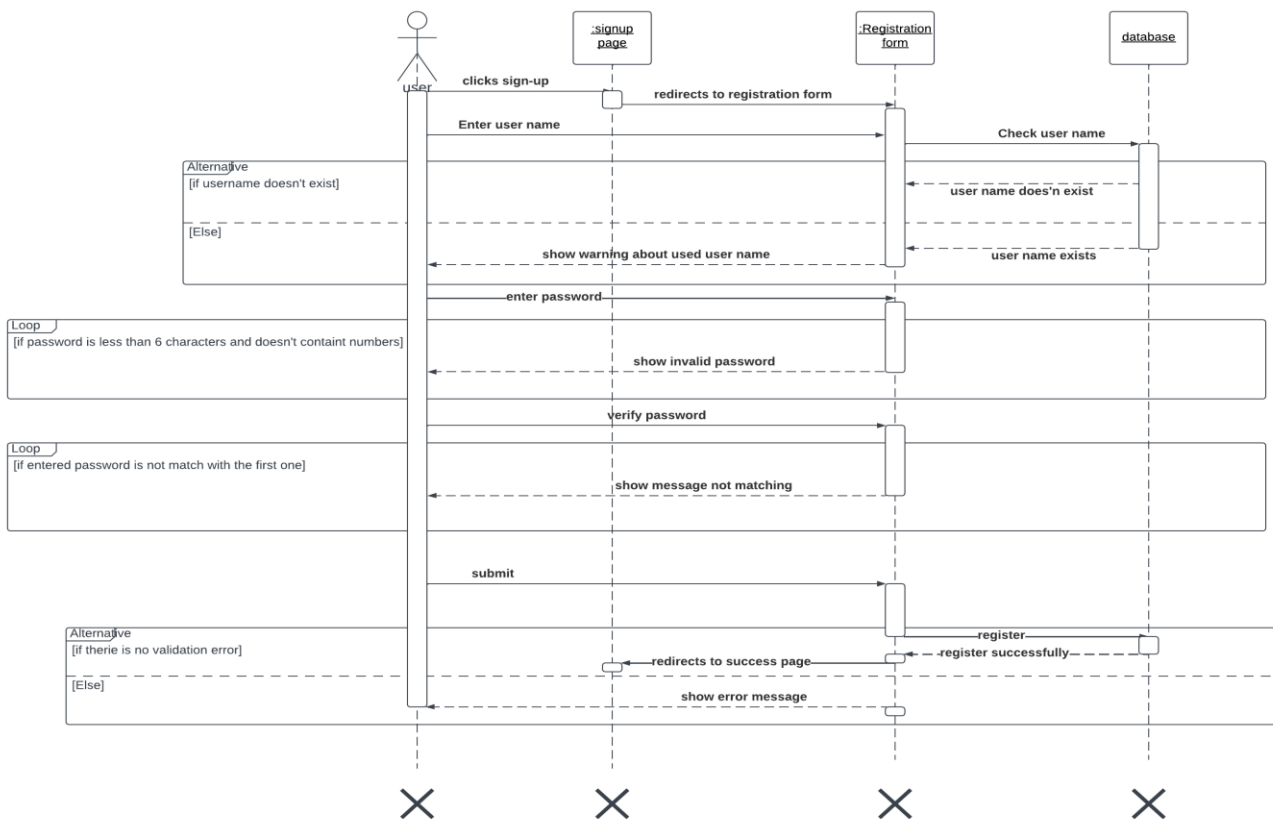
Figure 2-3 CREATE ACCOUNT USE CASE sequence diagram
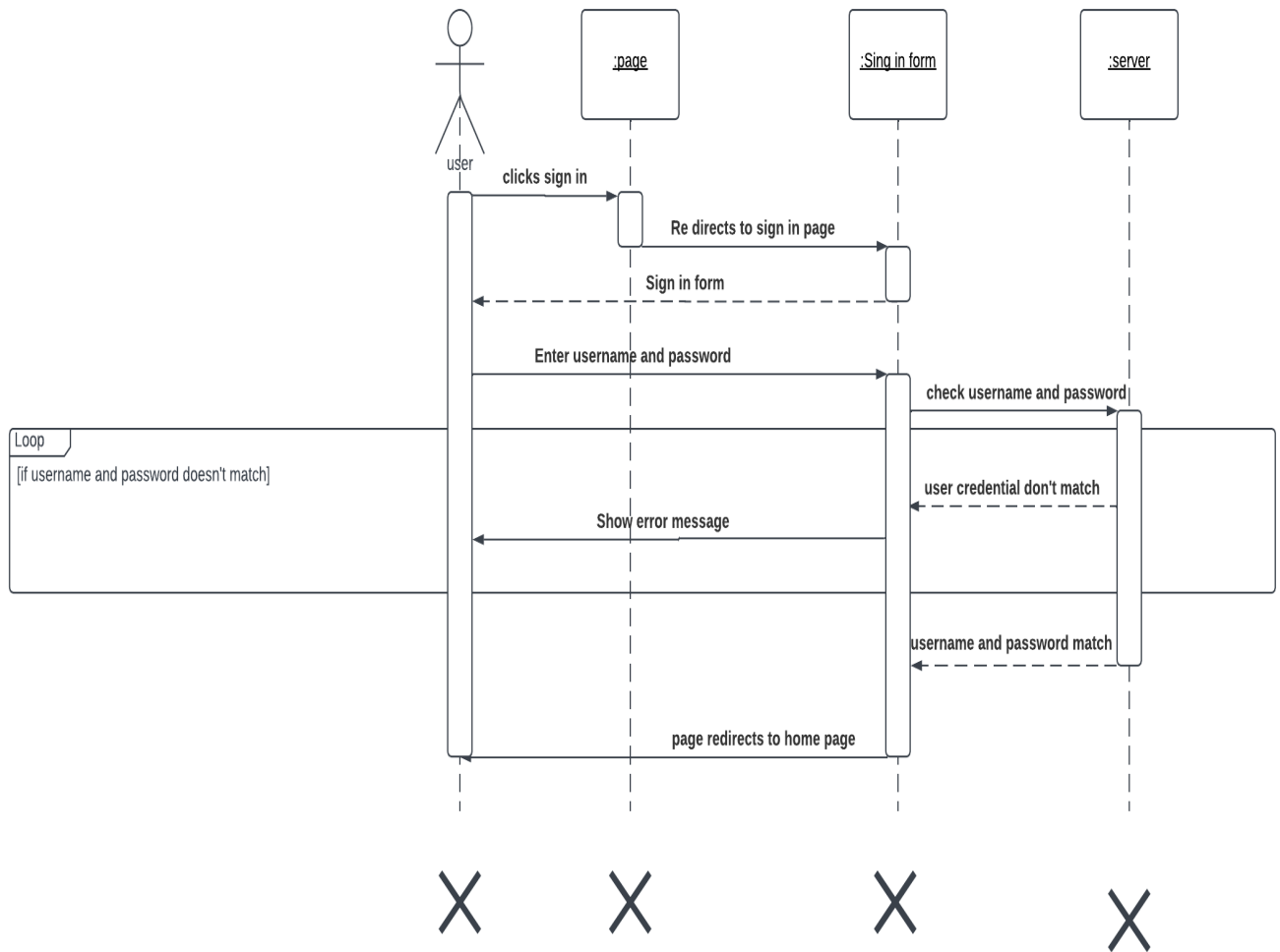
Figure 2-4 LOGIN USE CASE sequence diagram
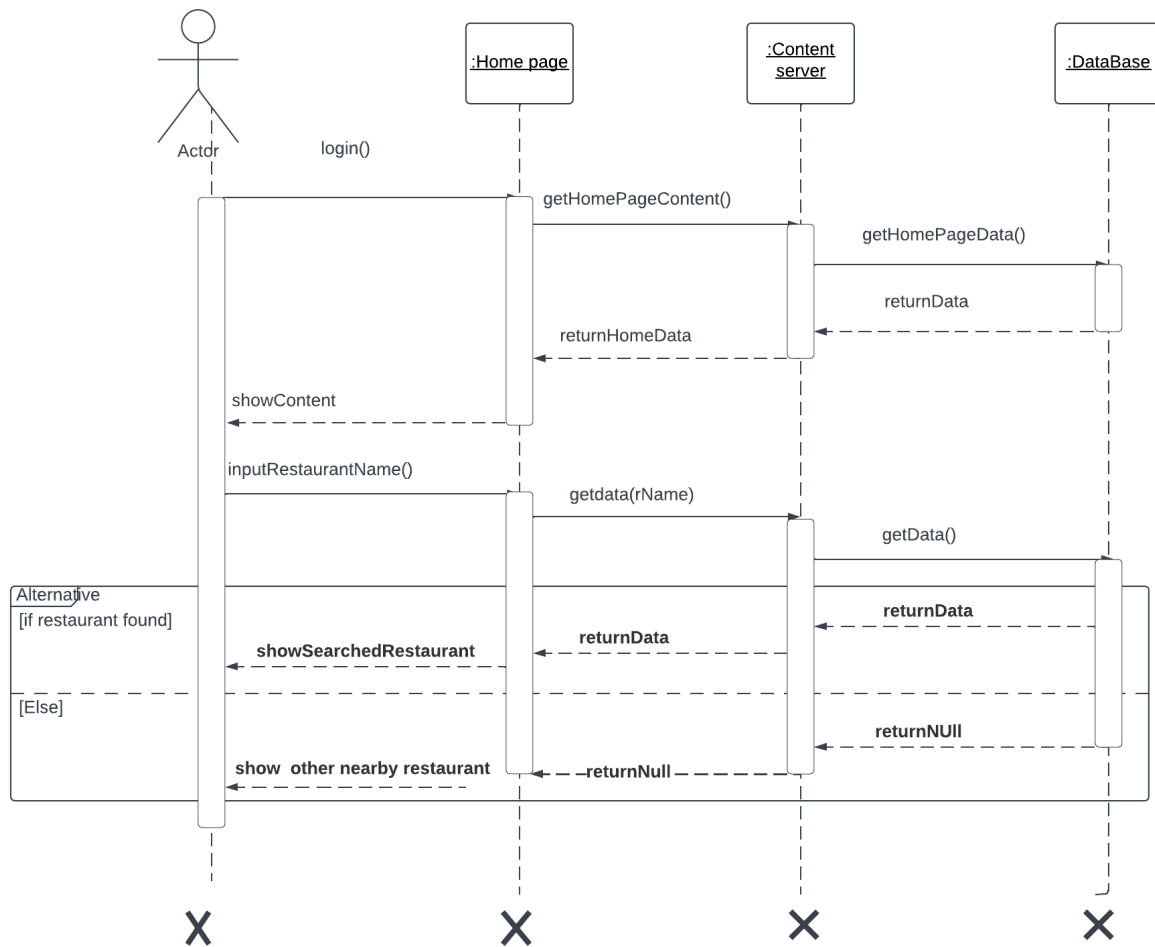
Figure 2-5 SEARCH RESATAURANT USE CASE sequence diagram
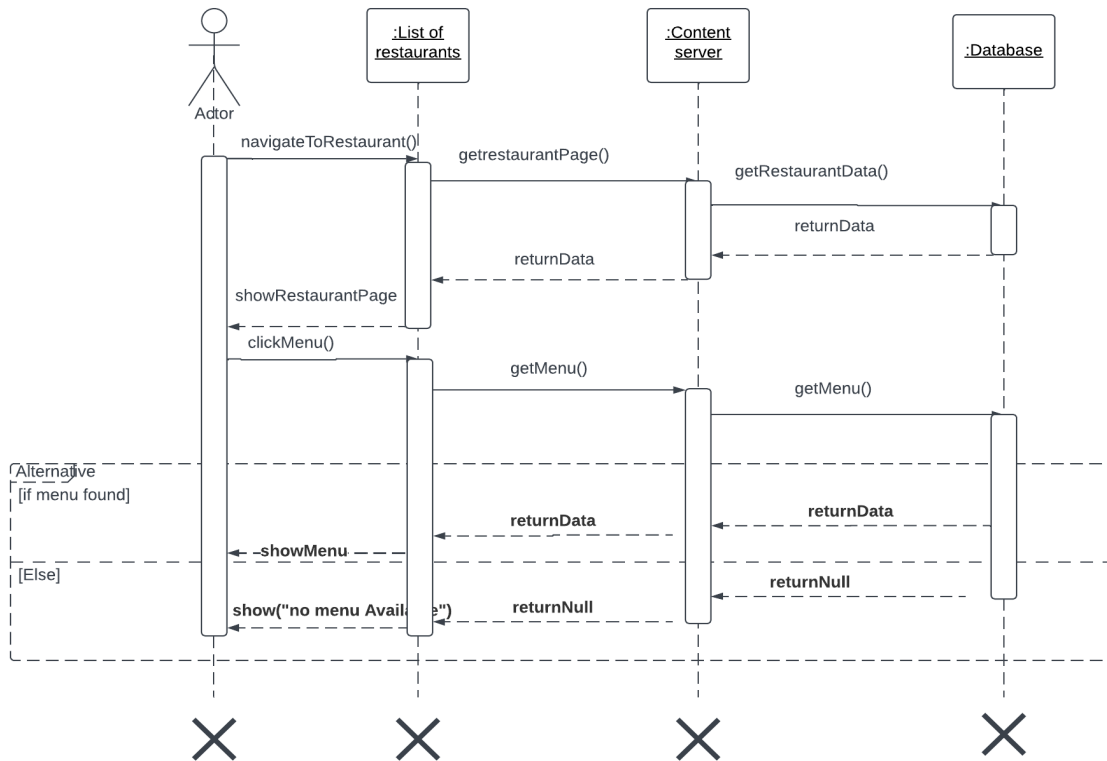
Figure 2-6 CHECK MENU USE CASE sequence diagram
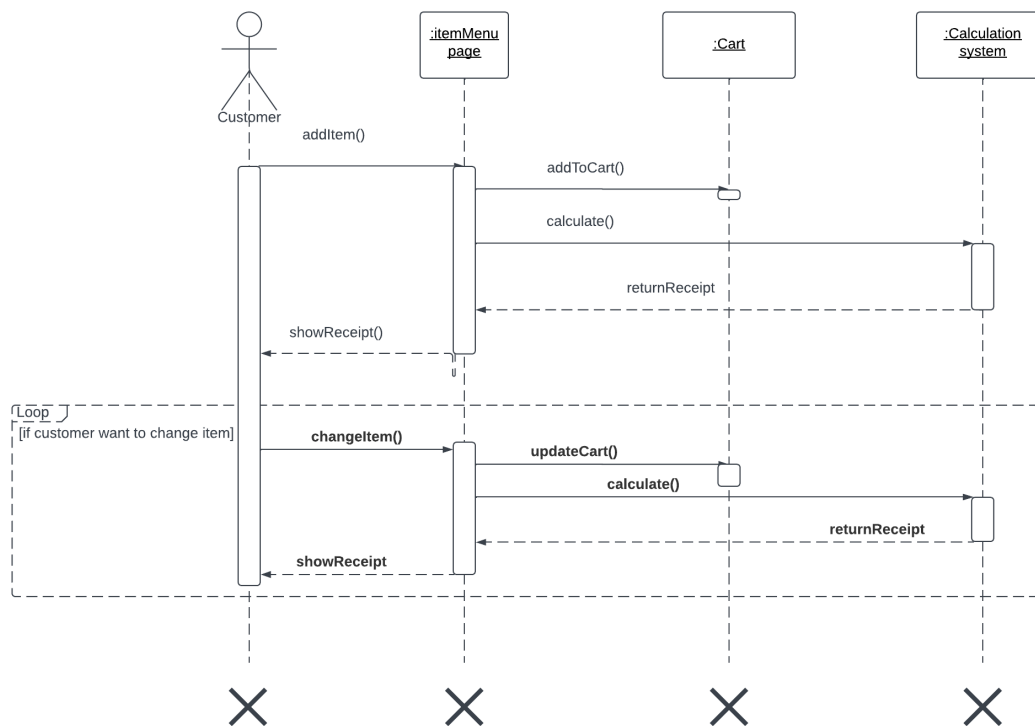
Figure 2-7 ADD TO CART USE CASE sequence diagram
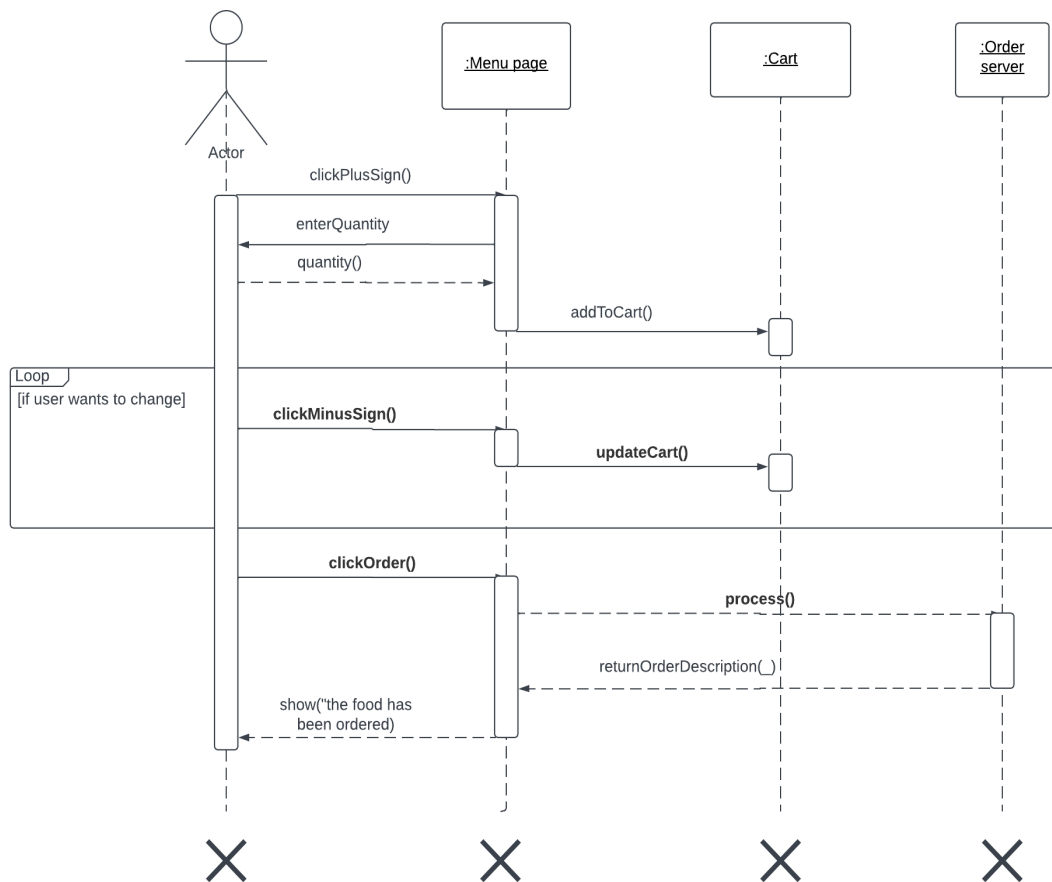
Figure 2-8 ORDER USE CASE sequence diagram
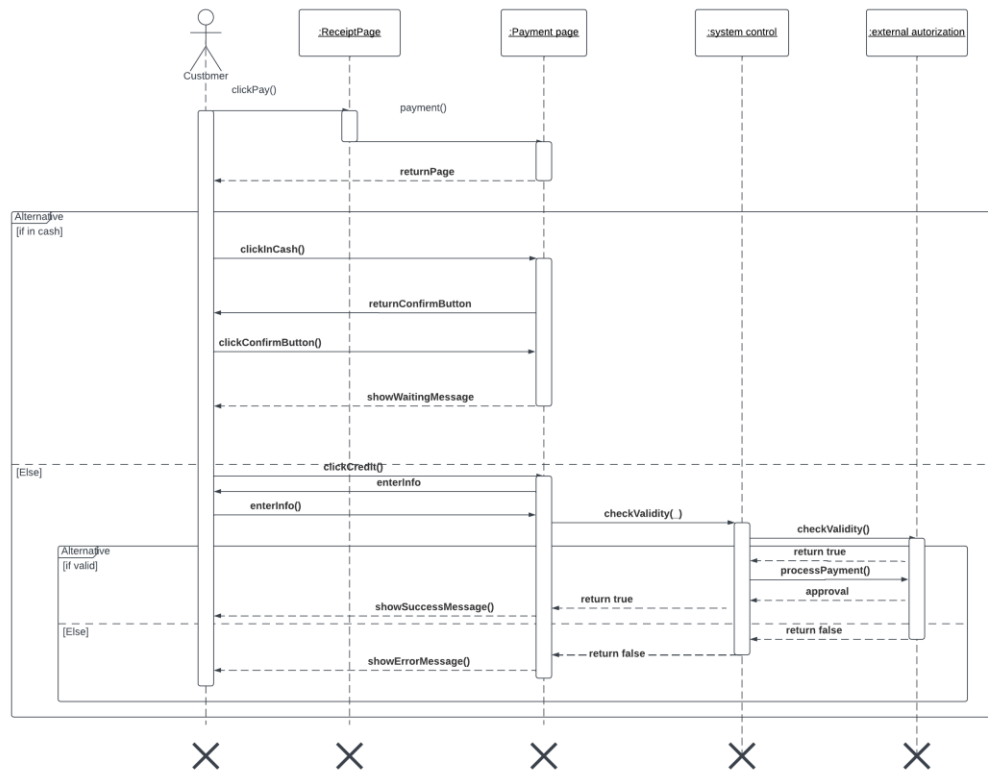
Figure 2-9 PAYMENT USE CASE sequence diagram
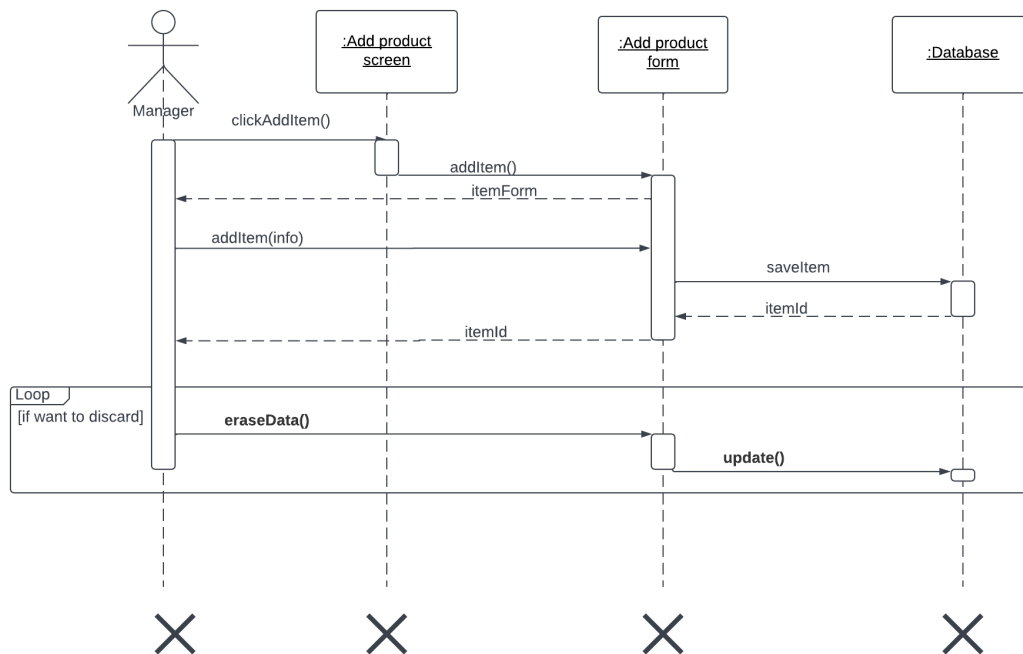
Figure 2-10 ADD MENU USE CASE sequence diagram
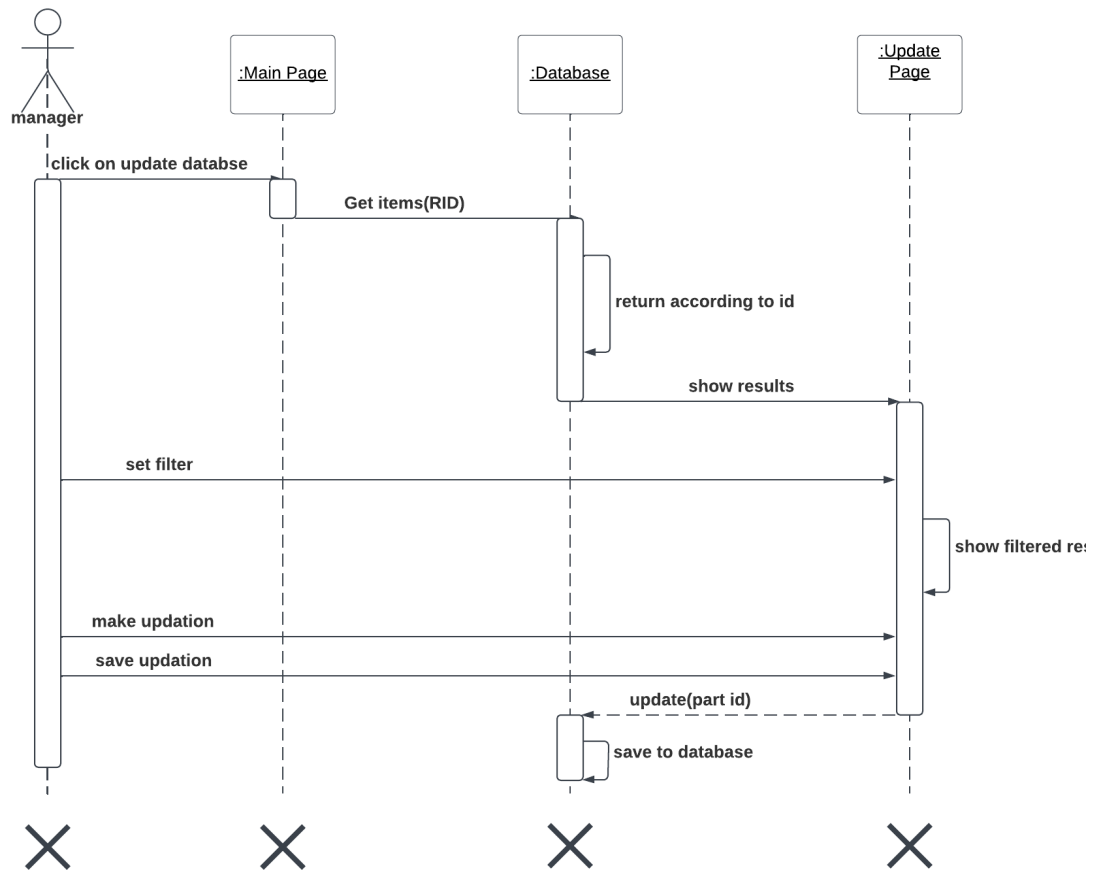
Figure 2-11 MANAGE MENU USE CASE sequence diagram
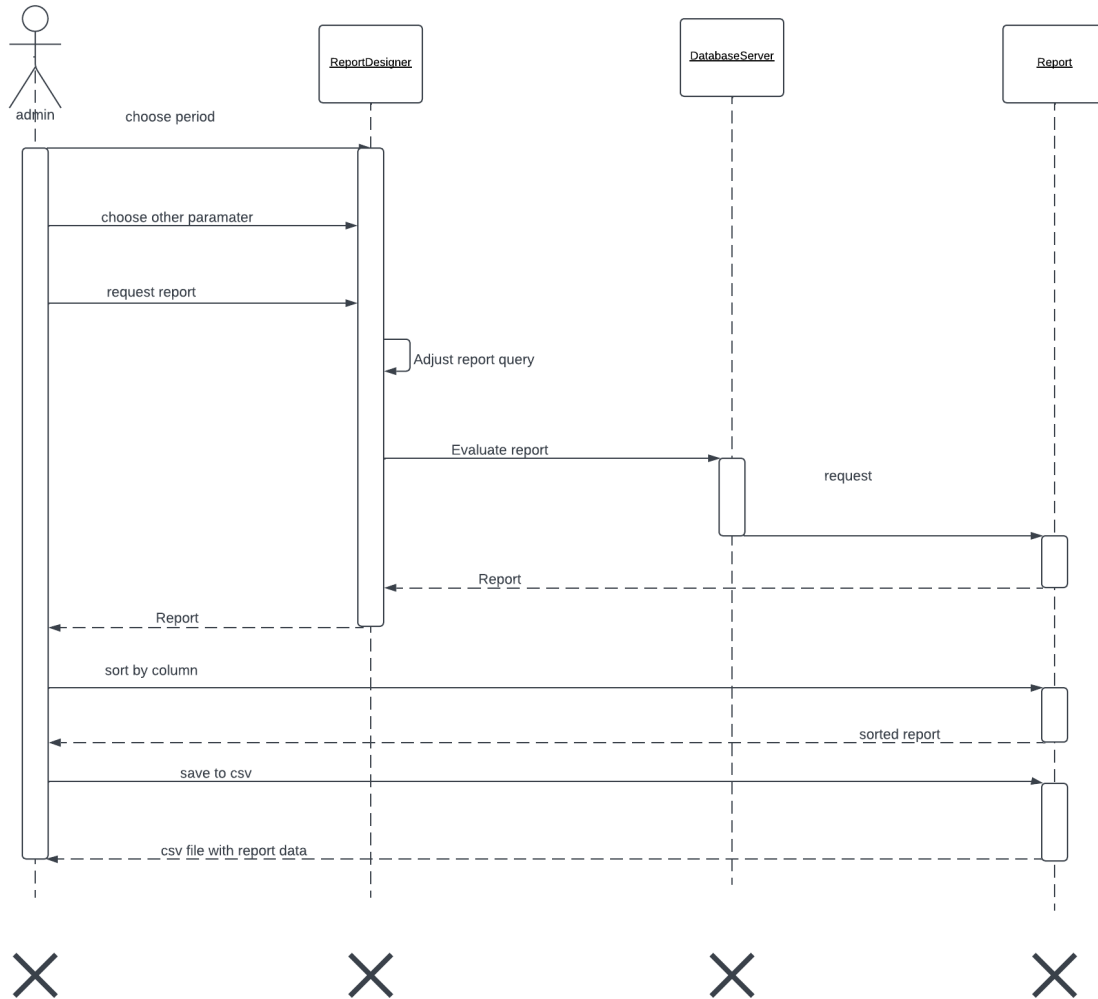
Figure 2-12 GENERATE REPORT USECASE sequence diagram
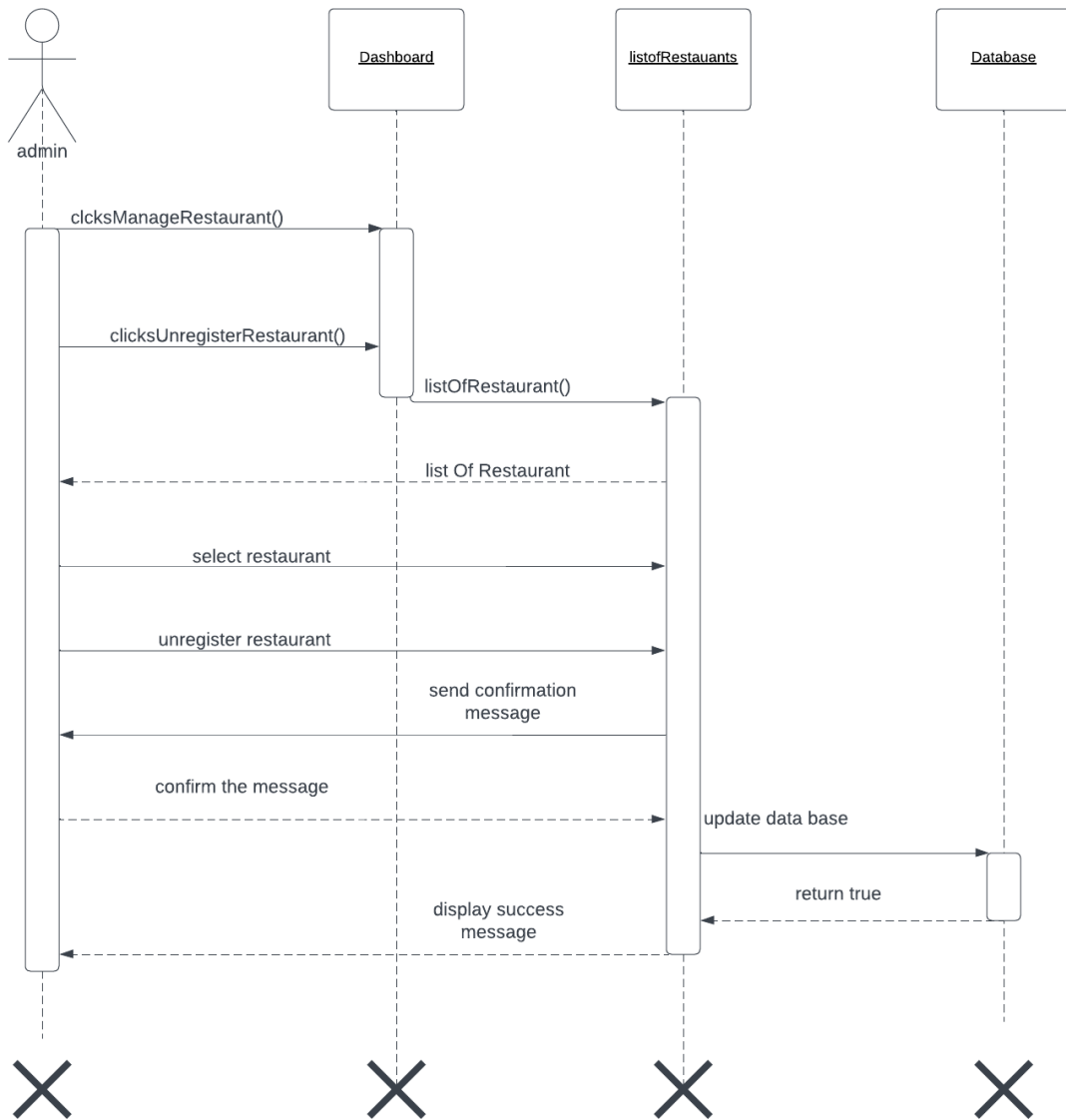
Figure 2-13 UNREGISTER RESTAURANT USE CASE sequence diagram

2.21. ACTIVITY DIAGRAM
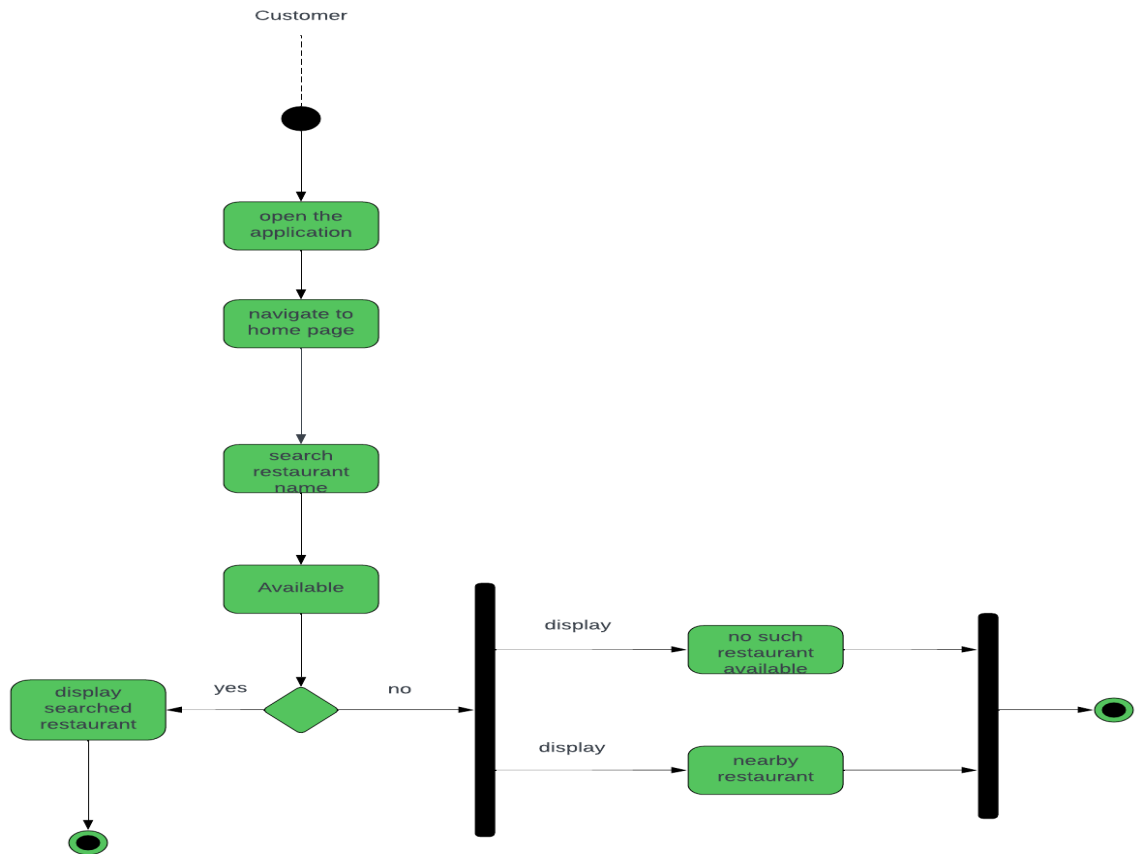
Figure 2-14 SEARCH RESTAURANT activity diagram
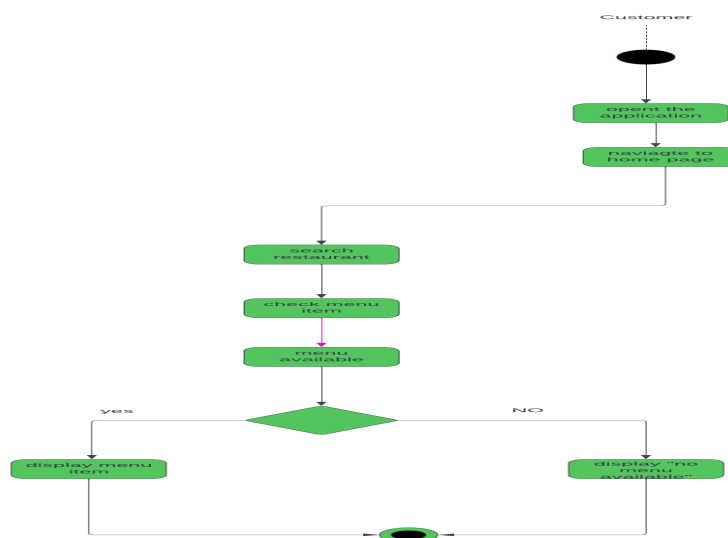
Figure 2-15 CHECK MENU activity diagram
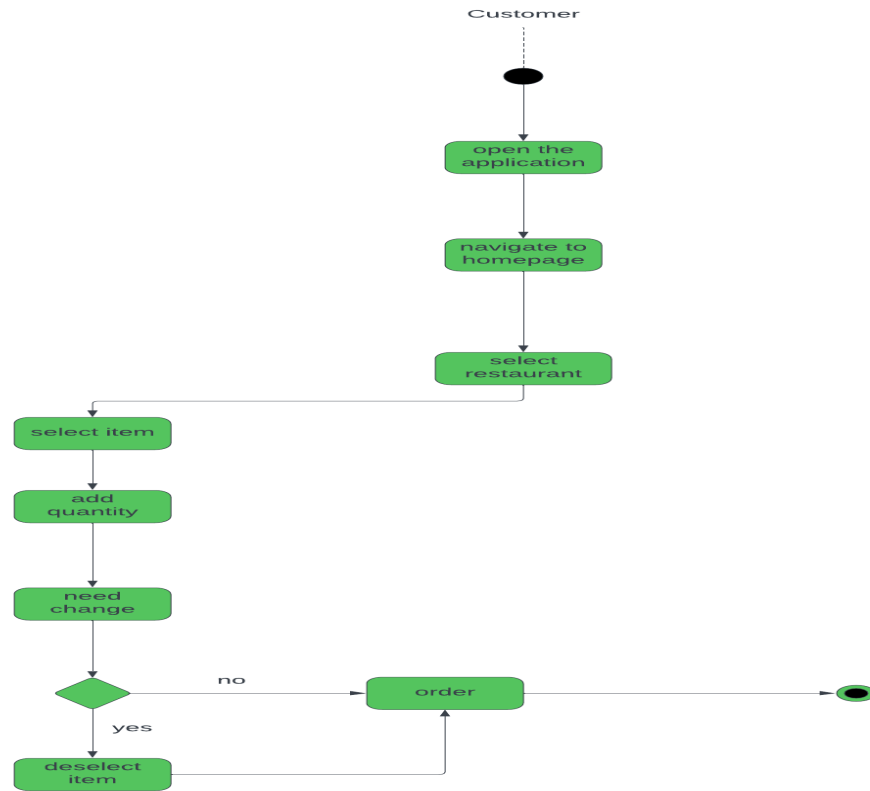
Figure 2-16 ORDER activity diagram

2.22. USER INTERFACE

User interface is a mechanism by which the actor interacts with the system either for query, input or update of information. Figure 2.17 – figure 2.22; is used to illustrate the user interface of the system.
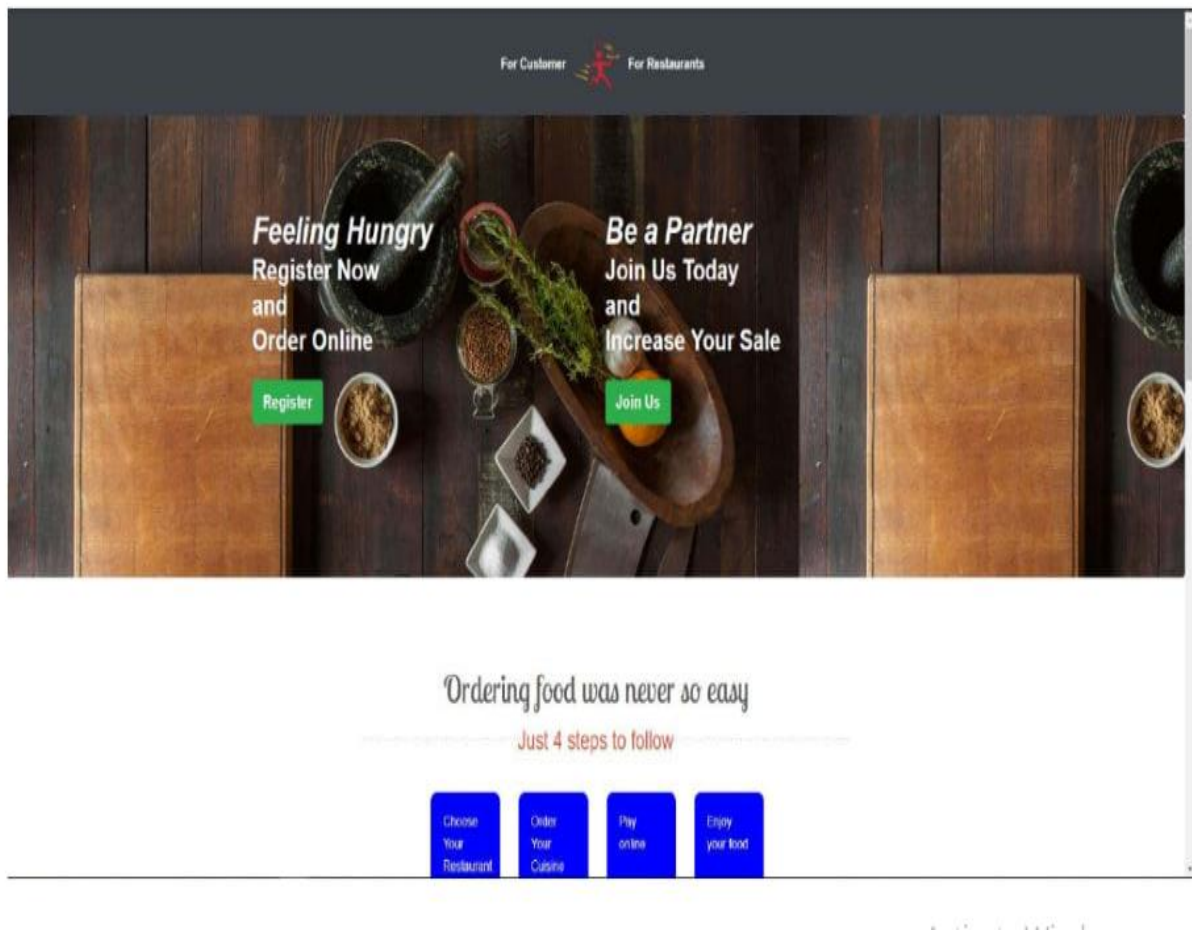
Figure 2-17 Home Page Interface

Figure 2-18 Registration page Interface

Figure 2-19 Login Page Interface
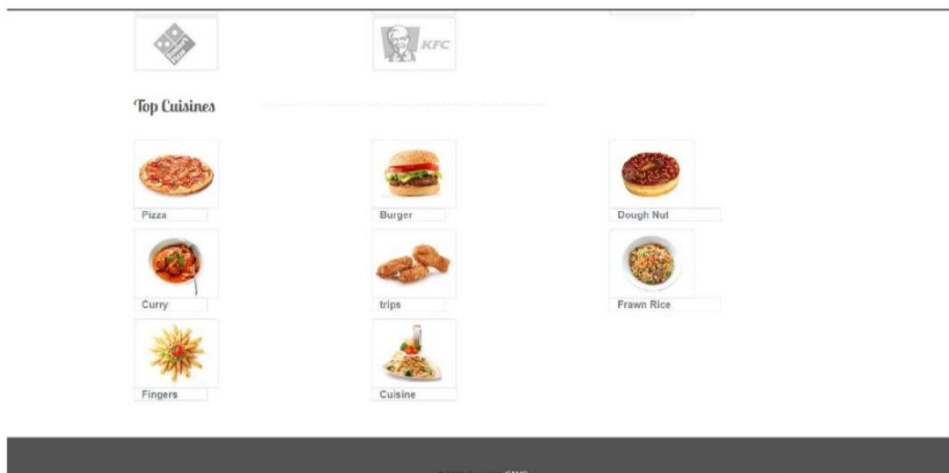
Figure 2-20 Top Cuisines Page Interface
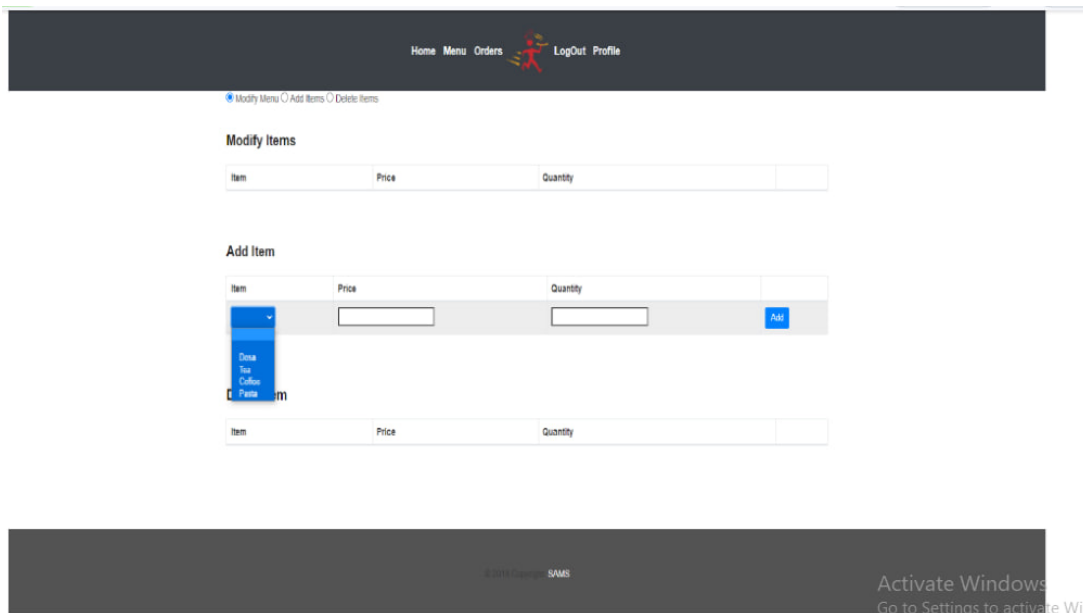
Figure 2-21 update Items Page Interface

Figure 2-22 Confirmation Page Interface

3. CHAPTER THREE - SYSTEM DESIGN

3.1. INTRODUCTION

This Part describes the proposed design for the food delivery android application. The system design concerns with the overall view of the application from the functional viewpoint. It is the process of defining the components, modules, interfaces, and data for the application to satisfy specified requirements. This section provides a clear description of the overall design parts of the "Food delivery android application" in organized way.

3.2. DESIGN GOALS

Design goals describe the qualities of the Application in which, developers should optimize that derived from the nonfunctional requirements. The following major quality perspectives are used to identify the design goals of the application.

3.3. DESIGN PERFORMANCE:

The design performance of a food delivery Android app should be focused on creating an intuitive user experience. The app should be easy to navigate and have a clear layout that allows users to quickly find what they are looking for. The app should also be visually appealing and have a modern design that is easy to understand.

Dependability:

The dependability of a food delivery Android app should be focused on ensuring that the app is reliable and secure. The app should be tested thoroughly to ensure that it is free from bugs and other issues. The app should also have a secure payment system that is encrypted and protected from hackers.

Maintainability:

The maintainability of a food delivery Android app should be focused on ensuring that the app is updated regularly with new features and bug fixes. The app should also be monitored for any potential security issues and have a system in place to quickly address any issues that arise. The app should also have a system in place to quickly respond to customer feedback and requests.

- ✓ Create an intuitive and user-friendly interface that allows users to easily navigate the app and find the food they are looking for.
- ✓ Ensure that the app is secure and that all user data is protected.
- ✓ Allow users to quickly and easily place orders and pay for their food.
- ✓ Provide users with accurate information about the restaurants and food items available.

- ✓ Allow users to track their orders in real-time.
- ✓ Provide users with a variety of payment options.
- ✓ Allow users to rate and review restaurants and food items.
- ✓ Provide users with helpful customer service and support.

3.4. END USER CRITERIA

 Easy to use interface: The app will have an intuitive and user-friendly interface that is easy to navigate and understand; the app loads quickly and is responsive to user input. The app should provide secure payment options for users to pay for their orders. The app has accurate order tracking and provides accurate order tracking so users can easily keep track of their orders. The app should provide push notifications to alert users of order updates and other important information. The app should allow users to customize their orders with various options such as toppings, sides, and more.

The App will provide an easy checkout process that is quick and efficient. Variety of restaurants: The app should offer a variety of restaurants to choose from so users can find the food they want. The app should provide ratings and reviews from other users to help users make informed decisions. It has Customer support service in which the app should provide customer support to help users with any issues they may have.

3.5. PROPOSED SOFTWARE ARCHITECTURE

The proposed software architecture for a food delivery Android application would include the following components:

User Interface: This component will provide the user with a graphical interface to interact with the application. It will include features such as a search bar, a list of restaurants, and a map view.

Database: This component will store all the data related to the application, such as user information, restaurant information, and order information for our food delivery app we will use firebase real-time database.

Backend Server: This component will handle all the requests from the user interface and will communicate with the database to retrieve and store data.

Payment Gateway: This component will handle all the payments related to the application.

Delivery Service: This component will handle all the logistics related to the delivery of the food orders.

3.6.   SUBSYSTEM DECOMPOSITION

Subsystem decomposition describes the division of the system into subsystems that are collection of classes, associations, operations, events, and constraints that are closely interrelated with each other and the responsibilities of each subsystem.

The "Food Delivery Android Application" has the following sub systems.

- ✓ **User Interface Subsystem**: the user interface subsystem of a food delivery application would be responsible for providing an intuitive and user-friendly experience for customers. This would include designing the user interface, creating navigation menus, and developing features such as search and filter functions be able to view menus, order history, and other information.
- ✓ **User Authentication Subsystem**: This subsystem is responsible for user authentication and authorization. It allows users to create accounts, log in, and manage their profiles.
- ✓ **Order Management Subsystem**: This subsystem is responsible for managing orders, including creating, editing, and canceling orders.
- ✓ **Payment Subsystem**: This subsystem is responsible for processing payments, including credit card, debit card, and other payment methods.
- ✓ **Delivery Subsystem**: This subsystem is responsible for managing the delivery of orders, including tracking orders, assigning drivers, and managing delivery routes.
- ✓ **Notification Subsystem**: This subsystem is responsible for sending notifications to users, such as order confirmations, delivery updates, and promotional messages.

The system decomposition and the interaction among the subsystems are shown in figure 3.1
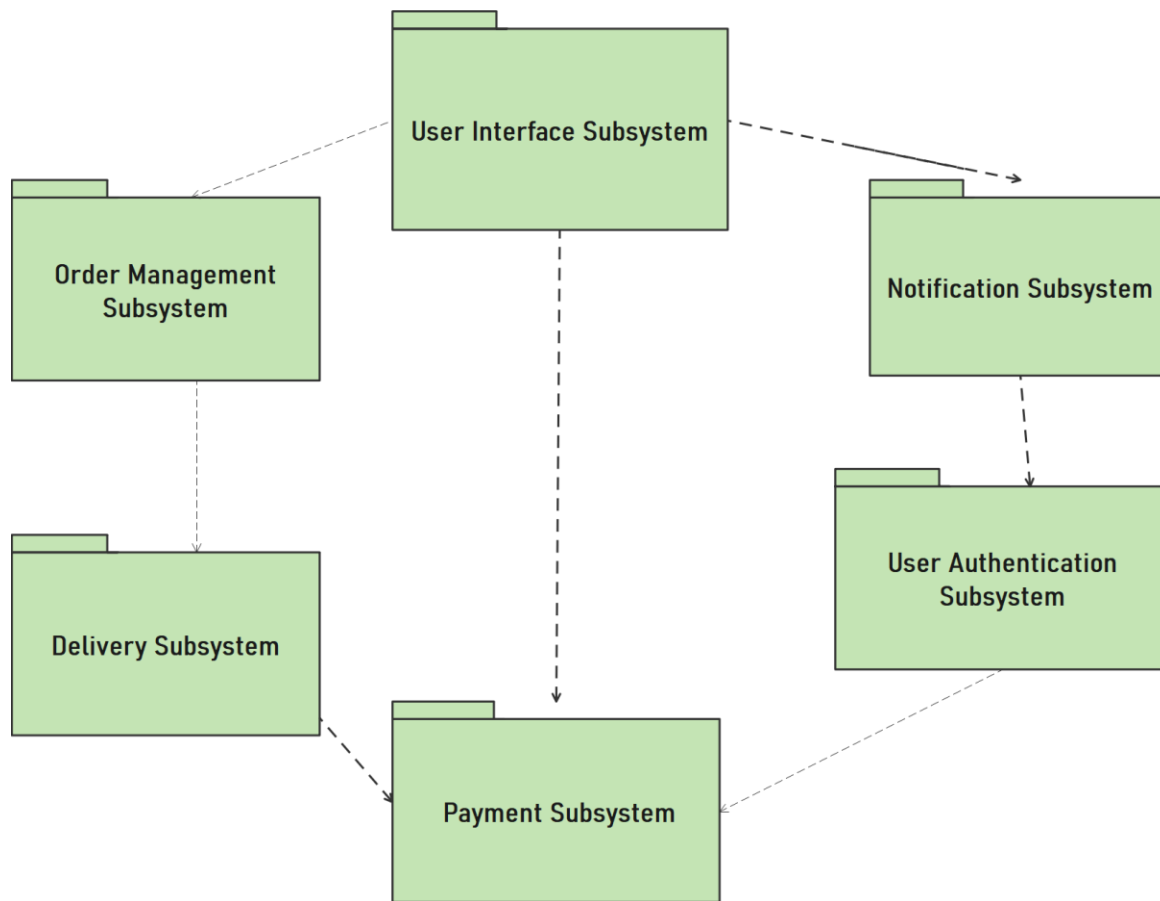
Figure 3-1 System Decomposition

User Interface Subsystem:

- ✓ **Login/Registration Subsystem**: This subsystem allows users to create an account and log in to the application. It also allows users to reset their passwords and manage their account settings.
- ✓ **Search/Browse Subsystem**: This subsystem allows users to search for restaurants and browse through menus. It also allows users to filter their search results based on various criteria such as cuisine type, price range, and location.
- ✓ **Ordering Subsystem**: This subsystem allows users to place orders for food items from the restaurants they have selected. It also allows users to customize their orders and add special instructions.
- ✓ **Customer Support Subsystem**: This subsystem allows users to contact customer support in case of any queries or issues. It also allows users to provide feedback and rate their experience with the application.
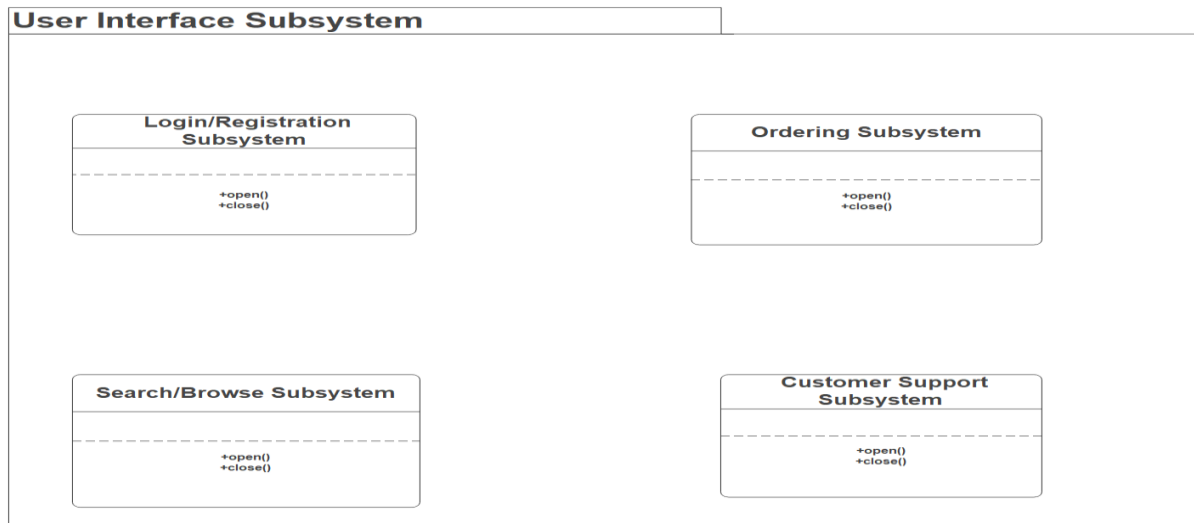
Figure 3-2 user interface subsystem

Report generating sub system

The report generating subsystem in a food delivery Android app can be divided into two main parts, which are generated by the restaurant and the system admin respectively.

1. Restaurant Report Generation - This part of the subsystem allows the restaurant to generate reports on their own performance and activity. Some common reports generated by the restaurant include:
   - ➢ Sales report - This report shows the total sales made by the restaurant for a particular time period.
   - ➢ Menu item report - This report shows the popularity of different menu items, helping the restaurant to understand which items are in high demand and which ones are not.
   - ➢ Customer report - This report shows the demographic and purchase data of the restaurant's customers, which can be used to better target marketing efforts.
2. System Admin Report Generation - This part of the subsystem allows the system administrator to generate reports on the overall performance and activity of the food delivery app. Some common reports generated by the system admin include:
   - ➢ Order and delivery report - This report shows the total number of orders received, the number of deliveries made, and any delivery-related issues that occurred.
   - ➢ User report - This report shows the number of active users on the app, the demographics of those users, and any feedback or complaints received from users.

63

> ➢ Financial report - This report shows the revenue generated by the app, the expenses incurred, and the overall profitability of the app.

The report generating subsystem is an important aspect of any food delivery app, as it allows the restaurant and system administrator to gain valuable insights into the performance and activity of the app, helping them to make data-driven decisions to improve the app's functionality and overall success.
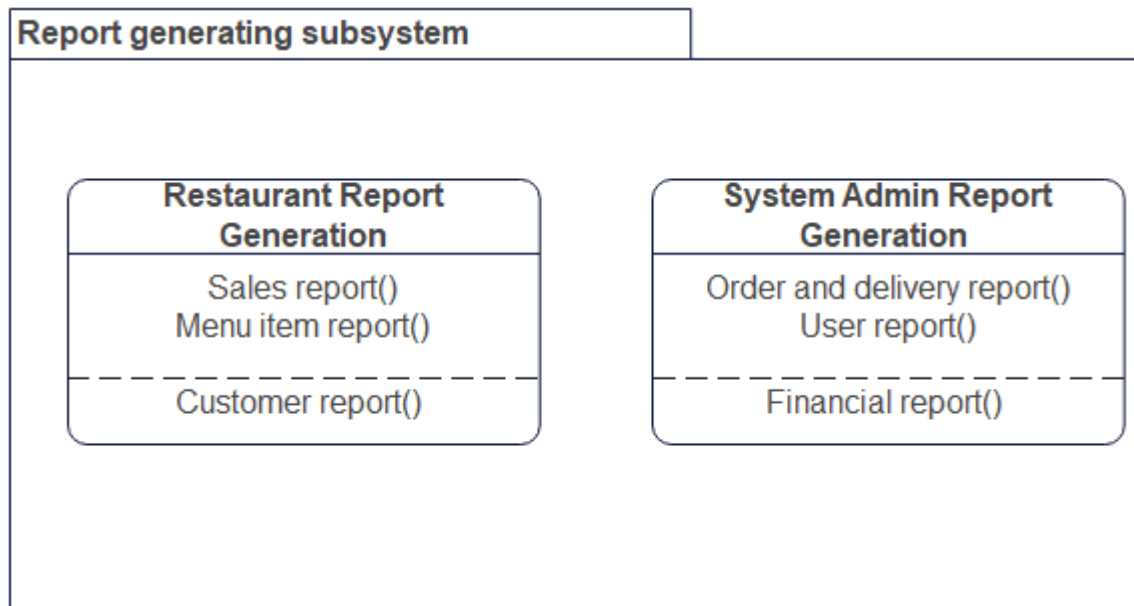
Figure 3-3 Report generating subsystem

Order Management Subsystem:

- ✓ **Order Placement Subsystem**: This subsystem allows customers to place orders for food items from the restaurant. It includes features such as selecting items from the menu, adding items to the cart, and making payment.
- ✓ **Order Tracking Subsystem**: This subsystem allows customers to track their orders in real-time. It includes features such as displaying the current status of the order, estimated time of delivery, and estimated cost.
- ✓ **Delivery Management Subsystem**: This subsystem allows restaurants to manage their delivery operations. It includes features such as assigning orders to delivery
- ✓ **Customer Support Subsystem**: This subsystem allows customers to get help with their orders. It includes features such as providing customer support via phone, email, and chat.
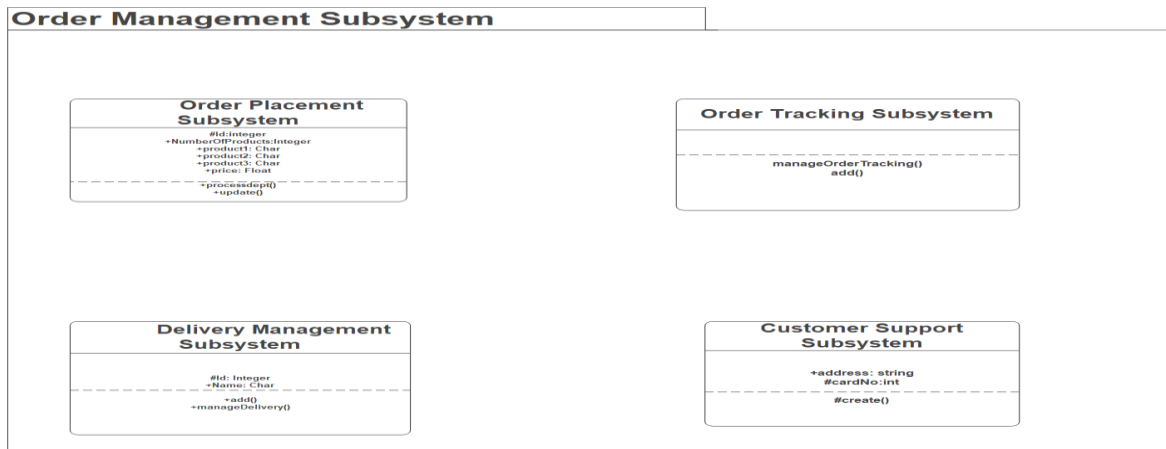
Figure 3-4 order management subsystem

Delivery Subsystem

1. Order Management System: This system is responsible for managing the orders placed by customers. It includes features such as order tracking, order history, order cancellation, and order modification.

2. Delivery System: This system is responsible for delivering the food to the customers. It includes features such as route optimization, delivery tracking, delivery history, and delivery cancellation.

3. Customer Support System: This system is responsible for providing customer support to customers. It includes features such as customer feedback, customer complaints, customer queries, and customer support.

4. Analytics System: This system is responsible for providing insights into the performance of the food delivery application. It includes features such as customer segmentation, order analysis, and delivery analysis.
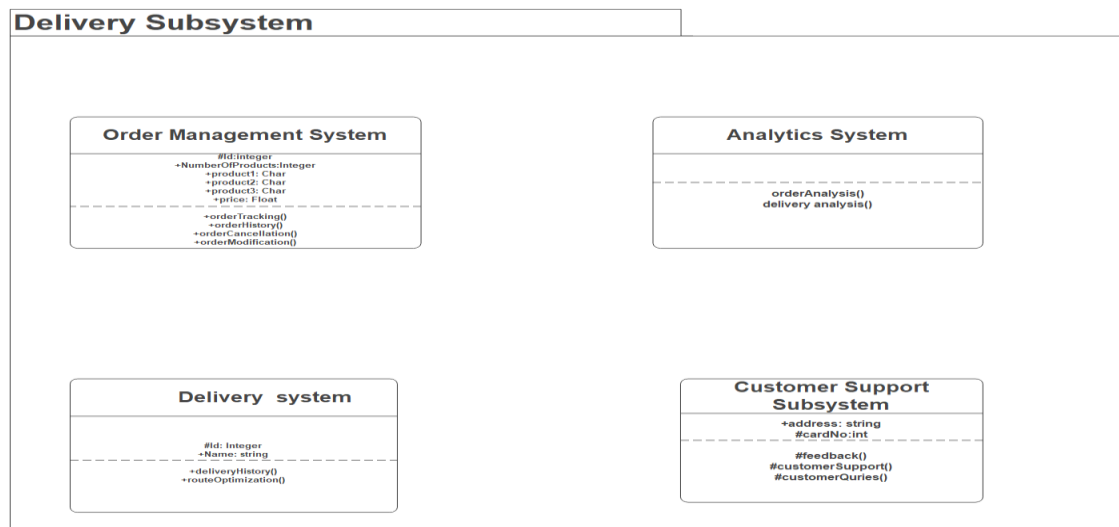
Figure 3-5 delivery subsystem

Notification Subsystem

1. User Analytics: This type of analytics helps to track user behavior and engagement with the application. It can be used to identify user preferences, track user activity, and measure user retention.
2. Payment Analytics: This type of analytics helps to track payment trends, analyze payment methods, and identify areas of improvement.
3. Inventory Analytics: This type of analytics helps to track inventory levels, analyze inventory trends, and identify areas of improvement.
4. Performance Analytics: This type of analytics helps to track application performance, analyze application usage, and identify areas of improvement.
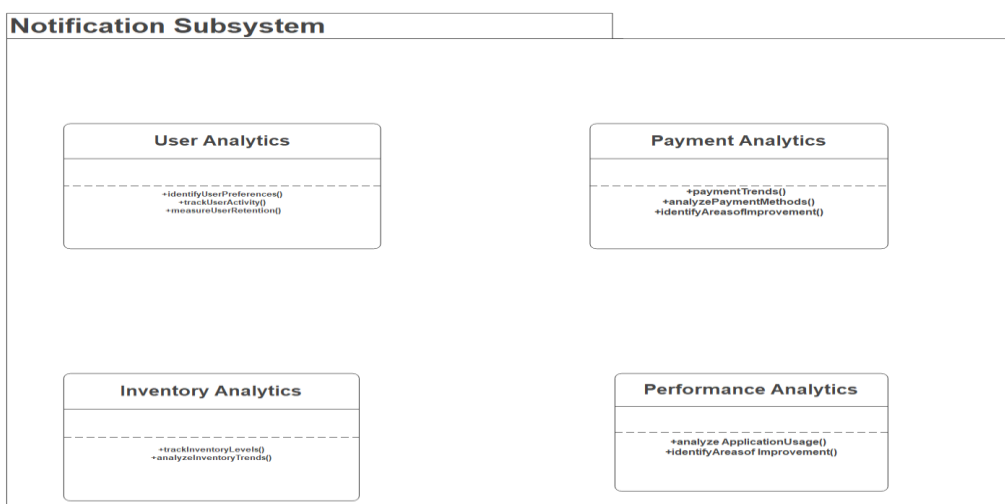
Figure 3-6 notification subsystem

Payment Subsystem:

1. PayPal: PayPal is a popular online payment system that allows customers to pay for their orders using their PayPal accounts.

Figure 3-7 payment subsystem

3.7. HARDWARE/SOFTWARE MAPPING

Hardware:

- ➤ Mobile devices (Smartphone, tablet) with Android operating system.
- ➤ GPS enabled device for location tracking.
- ➤ Internet connectivity (WiFi, cellular data).

Software:

- ✓ Android Studio or other Android app development tools.
- ✓ Google Maps API for location-based services.
- ✓ Payment gateway API for online transactions.
- ✓ Push notifications for order updates.
- ✓ Database management system for storing order details and customer information.
- ✓ Backend server for data management and processing.
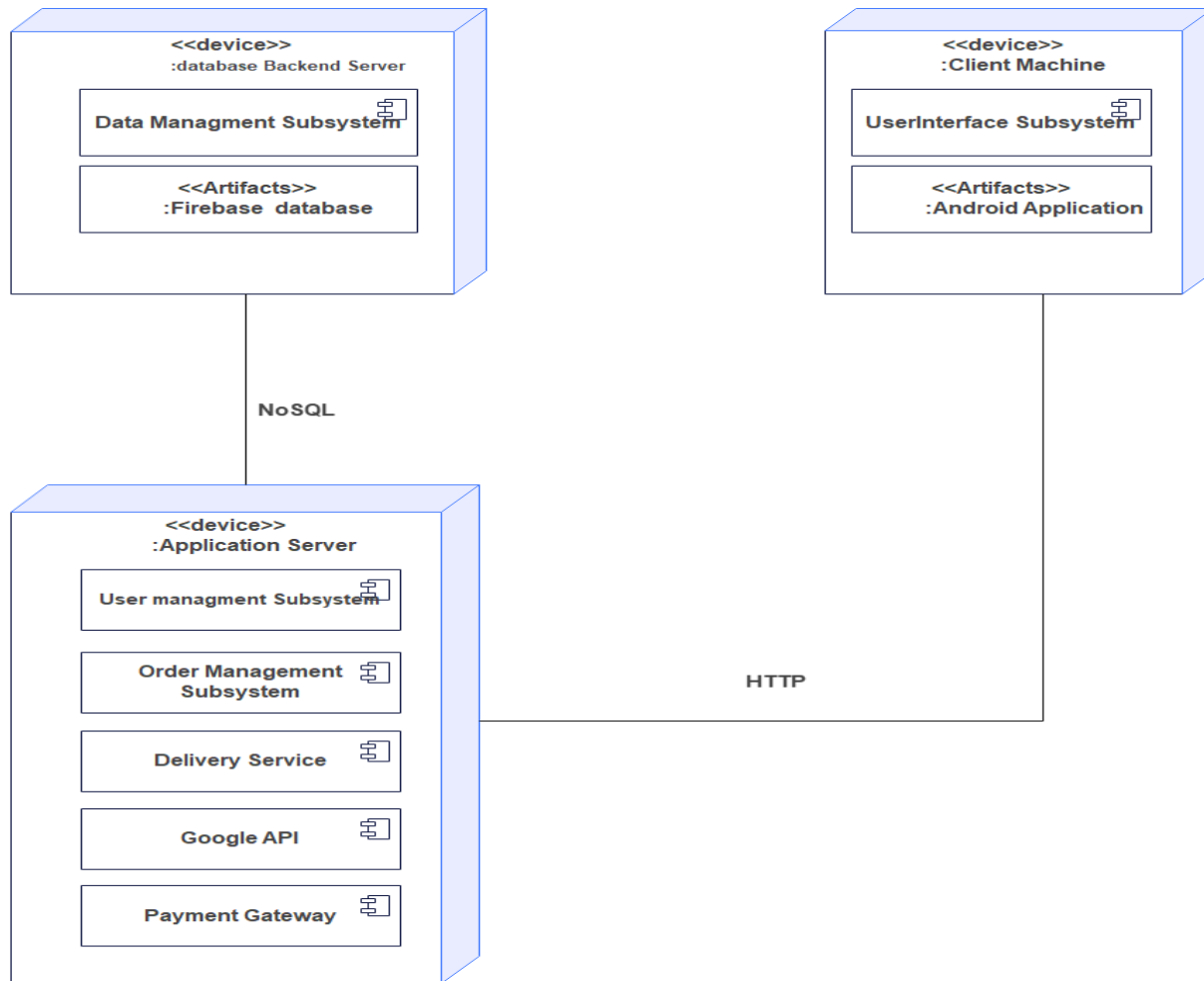
Figure 0-1 hardware software mapping

3.8. PERSISTENT DATA MANAGEMENT

Persistent data management describes the persistent data stored by the Application and the data management infrastructure required for it. The Application uses the Firebase database; a NoSQL database. Specifically, it is a document-oriented database that stores data as collections of documents, which are stored in JSON format. NoSQL databases differ from traditional relational databases in that they do not use a schema, allowing for greater flexibility in data modeling and storage. Firebase is part of the Google Cloud Platform and provides a real-time database and other services, such as hosting, authentication, and analytics.

Firebase is a popular platform for developing web and mobile applications, and it offers several advantages over other development platforms. Here are some of the key advantages of using Firebase:

- ✓ Real-time database: Firebase offers a real-time database that allows developers to build responsive and collaborative applications. The real-time database syncs data across multiple devices in real-time, which means that users can see changes instantly.
- ✓ Serverless architecture: Firebase is a serverless platform, which means that developers don't have to worry about managing servers or infrastructure. This allows developers to focus on building the application itself, rather than worrying about server maintenance.
- ✓ Authentication: Firebase provides a built-in authentication system that supports several authentication methods, such as email/password, social login, and phone number verification. This makes it easy for developers to add authentication to their application without having to build it from scratch.
- ✓ Cloud Functions: Firebase also offers Cloud Functions, which are serverless functions that can be triggered by events in the Firebase Realtime Database, Firebase Authentication, or other Firebase services. This allows developers to write custom logic and business rules without having to manage servers.
- ✓ Analytics: Firebase offers a powerful analytics tool that allows developers to track user behavior, measure the effectiveness of marketing campaigns, and gain insights into user engagement.
- ✓ Hosting: Firebase offers web hosting for static assets, such as HTML, CSS, and JavaScript files. This makes it easy for developers to host and deploy their application without having to set up and manage a separate web server.
- ✓ Integration with other Google services: Firebase is part of the Google Cloud Platform, which means that it can easily integrate with other Google services, such as Google Cloud Storage, Cloud Functions, and Cloud Messaging.

Overall, Firebase offers a comprehensive set of tools and services that make it easy for developers to build high-quality, real-time applications without having to worry about infrastructure or server maintenance. Its real-time database, serverless architecture, authentication system, Cloud Functions, analytics, hosting, and integration with other Google services make it a powerful platform for building modern web and mobile applications.

When building a food delivery Android application, data management is critical to ensure that the application is efficient, reliable, and user-friendly. Here are some examples of data management object models that can be used in a food delivery application:

- ➢ Customer object model: This object model represents the customers using the food delivery application. It includes properties such as the customer's name, contact information, delivery address, payment details, and order history. This object model is used to manage customer-related data, such as order history and customer profile information.
- ➢ Order object model: This object model represents the orders placed by customers using the food delivery application. It includes properties such as the order ID, order status, delivery time, delivery address, and order items. This object model is used to manage

order-related data, such as tracking the status of orders, managing delivery time, and ensuring that the correct order items are delivered.

➢ Menu object model: This object model represents the menu items offered by the food delivery application. It includes properties such as the menu item ID, name, description, price, and image. This object model is used to manage menu-related data, such as updating the menu items and ensuring that the correct prices are charged.

➢ Driver object model: This object model represents the drivers responsible for delivering the food to customers. It includes properties such as the driver's name, contact information, driver ID, and vehicle information. This object model is used to manage driver-related data, such as assigning drivers to specific orders and tracking driver location.

➢ Payment object model: This object model represents the payment methods accepted by the food delivery application. It includes properties such as the payment method ID, name, and description. This object model is used to manage payment-related data, such as processing payments and ensuring that the correct payment methods are accepted.

 Overall, these object models help to organize the different types of data that are managed by a food delivery Android application. By using a data management object model, developers can ensure that the application is efficient, reliable, and user-friendly.
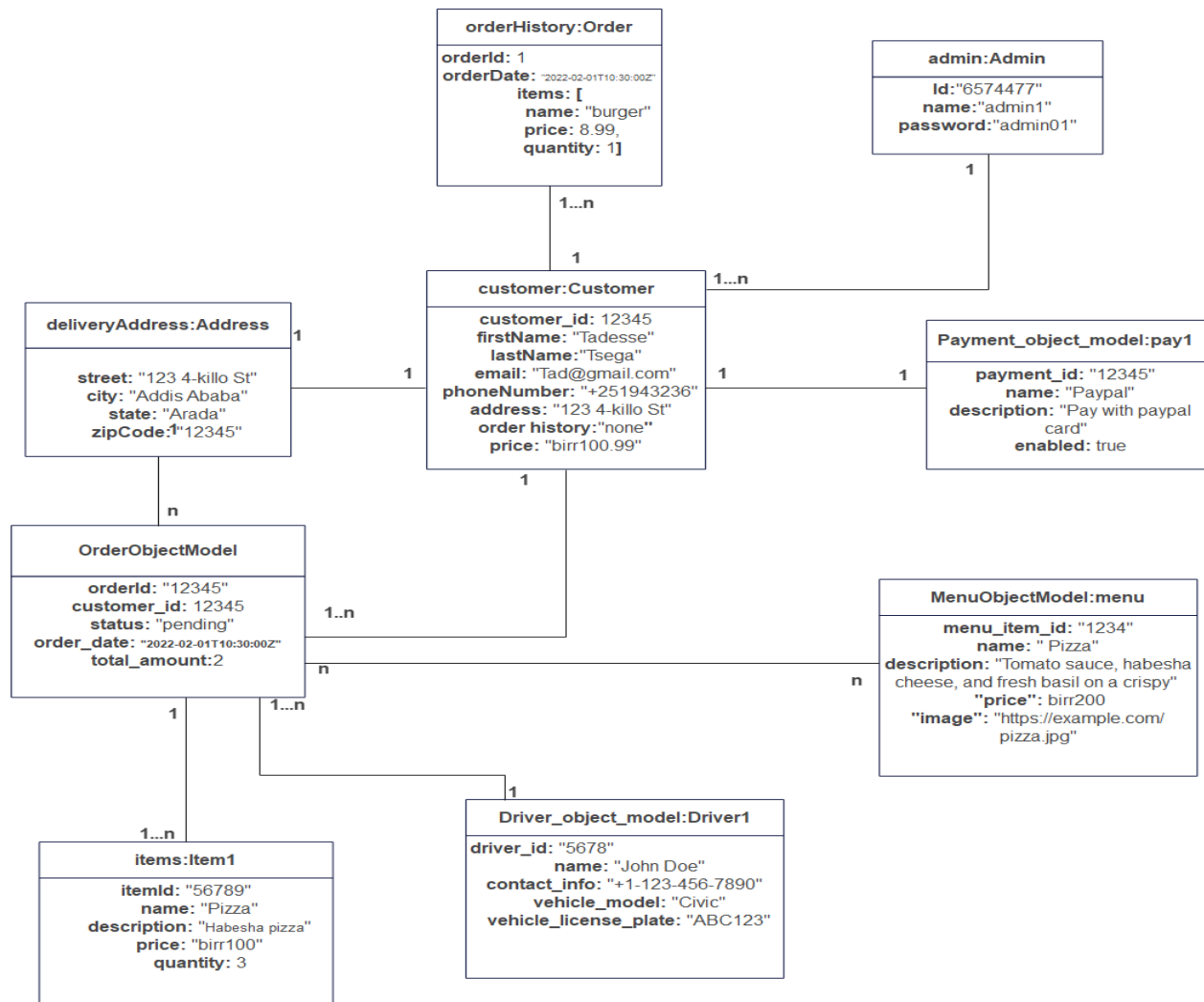
Figure 0-2 data management object model

Object Diagram

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
System admin	Adminid	Identification of the admin	Varchar(15)	Primary key
	Status	Status of the system admin account	Varchar(15)	Not null
	Name	Name of the Admin	Varchar(14)	Not null
	Email	E-mail using	Varchar(15)	Not null

Table 3-34 System admin

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Manager	Id	The Identification of the manager	Varchar(15)	Primary key
	Name	Name of the manager	Varchar(15)	Not null
	contact	Contact detail of the manager	Varchar(15)	Not null
	email	Email of the manager	Varchar(15)	Not null
	Name of restaurant	Name of restaurant managed by the manager	Varchar(15)	Not null

Table 3-35 Manager

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
RESTAURANT	Restid	Identification of the Restaurant	Varchar(15)	Primary key
	Name	Name of the Restaurant	Varchar(14)	Not null
	Email	E-mail using	Varchar(15)	Not null
	deliveryid	Identification of the delivery person	Varchar(15)	Primary key
	Address	Address of the restaurant	Varchar(15)	Not null
	Contact	Contact details of the restaurant	Varchar(15)	Not null
	Opening/Closing hour	opening and closing hours	Varchar(15)	Not null
	Menu	The menu item of the restaurant	Varchar(15)	Not null

Table 3-36 RESTAURANT

TABLE	FIELD	DESCRIPTION	TYPE	
CUSTOMER	CustomerId	Identification of the customer	Varchar(15)	Primary key
	Username	Name of the customer	Varchar(14)	Not null

	Email	E-mail using	Varchar(15)	Not null
	address	Address of the restaurant	Varchar(15)	Not null
	Payment	Payment information	Varchar(15)	Not null

Table 3-37 customers

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
	MenuItemId	Identification of MenuItem	Varchar(10)	Primary Key
Menu	Menucategory	Category of Menu	Varchar(15)	Not null
	Menuname	Name of Menuitem	Varchar(15)	Not null
	Menudescription	Description of Menu	Varchar(15)	Not null
	ItemPrice	Description of item price	Varchar(15)	Not null

Table 3-38 menu table

TABLE	FIELD	DESCRPTION	TYPE	CONSTRAINTS
	OrderId	Identification of order	Varchar(15)	Primary Key
ORDER	ItemId	Identification of the product	Varchar(15)	Foreign Key
	Orderdate	Date of order	Date	Not null
	Customername	Name of customer	Varchar(15)	Not null
	Quantity	Quantity	Int	Not null
	Itemname	Name of Item	Varchar(15)	Not null
	Companyname	Name of company	Varchar(15)	Not null

		Deliverydate	Date of delivery product	Date	Not null
		Restaurantdetails	Detail of the restaurant	Varchar(15)	Not null
		Paymentdetails	Detail of the payment	Varchar(15)	Not null

Table 3-39 order table

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
DELIVERY	DeliveryId	Identification of delivery	Varchar(15)	Not null
	Item	Item	Varchar(15)	Not null
	Quantity	Quantity of product	Int	Not null
	Datereceivable	Date of receivable	Date	Not null

Table 3-40 delivery tables

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Cart	ItemID	Identification of item	Varchar(15)	Not null
	ItemName	Name of the Item	Varchar(15)	Not null
	Itemdescription	Description of Item	Varchar(15)	Not null
	Itemprice	Price of the Item	Varchar(15)	Not null
	Itemquantity	Quantity of the item	Varchar(15)	Not null

Table 3-41 cart

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Payment	PaymentID	Identification of payment	Varchar(15)	Primary key
	OrderID	Identification of order	Varchar(15)	Foreign key
	CustomerID	Identification of customer	Varchar(15)	Foreign key
	Currency	Types of currency used	Varchar(15)	Not null

	Paymentmethod	Method of payment it used	Varchar(15)	Not null

Table 3-42 payment table

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Report generating	ReportId	Indentification of the report	Varchar(15)	Primary key
	Reporttype	The type of report generrated	Varchar(15)	Not null
	Reportdate	The date report generated	Varchar(15)	Not null
	Reportdata	The actual report data	Varchar(15)	Not null
	Reportaccess	The level of access of the report	Varchar(15)	Not null
	Reportowner	The owner of the report	Varchar(15)	Not null
	Reportformat	The format of the report	Varchar(15)	Not null

Table 3-43 report generating

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Notification	NotifyId	Identification of notification	Varchar(15)	Not null
	Type	Type of notification	Varchar(15)	Not null
	Date and time	Date of notification	Varchar(15)	Not null
	sender	Sender of notification	Varchar(15)	Not null
	Receiver	Receiver of notification	Varchar(15)	Not null
	Message	Message of notification	Varchar(15)	Not null

Table 3-44 notification

TABLE	FIELD	DESCRIPTION	TYPE	CONSTRAINTS
Complaint	ComplaintId	The identification of complaint	Varchar(15)	Not null
	OrderId	The identification of the order man	Varchar(15)	Not null
	CustomerId	The identification of customer	Varchar(15)	Not null
	Actiontaken	The action to be taken	Varchar(15)	Not null

75

	date	Complaint date	Varchar(15)	Not null
	type	Types of the complaint	Varchar(15)	Not null

Table 3-45 complaint

SystemAdmin:SystemAdmin1

Adminid:0112
Status:"Active"
Name:"GRT"
Email:GRT12@gmail.com

Manager:manager1

Id:Gr123
Name:"Cbrtyp"
contact:"+123453"
email:"+12783289"
Name of restaurant:"sky-light"

RESTAURANT:RESTAURANT1

Restid:vr2123
Name:"Sky Light"
Email:"SkyL@gmail.com"
deliveryid:DR3245
Address:"bole road123"
Contact:"+1511346789"
Opening/Closing hour:"24hours"
Menu:"Starters "

Report generating:Report g1

ReportId:Rp8965
Reporttype:"Customer report"
Reportdate:12/07/2022
Reportdata:"overall customer report"
Reportaccess :"readonly"
Reportowner :"Sadmin"
Reportformat:"file format"

menu:menu1

MenuItemId:Mn4632
Menucategory:"Appetizers"
Menuname:"Margherita Pizza"
Menudescription:"fresh tomato sauce, mozzarella cheese, and fresh basil leaves"

Notification:notify1

NotifyId:notf156
Type:"order status"
Date and time:"08/10/2021 7:30 PM"
sender:"grt restaurant"
Receiver :"ordinary one"
Message :"stay tuned"

delivery:deliver one

DeliveryId:12345
Item:"Cheese burger"
Quantity:1
Datereceivable:"6:00pm"

Cart:Cart1

ItemID:7549
ItemName :"Pizza"
Itemdescription:"made of fresh or canned tomatoes, olive oil, garlic, salt, and herbs such as oregano and basil. The sauce is usually cooked"
Itemprice:"birr200"
Itemquantity:3

Figure 0-3 System Admin

Customer:customer0

CustomerId:ur567
Username:"tadesse tad"
Email:"tad@gmail.com"
address:"4-killo"
Payment:"birr200"

Complaint:complaint1

ComplaintId:ur567
OrderId:"5467"
CustomerId:ur567
Actiontaken:"not delivered yet"
date:"12/03/12 4:30"
type:"order complaint"

Manager:manager1

Id:mg2345
Name:"bluepoint12"
contact:"+12334554"
email:"t@gmail.com"
Name of
restaurant:"blue nile"

Notification:notifycustomer

NotifyId:NF3327
Type:"notify delvery"
Date and time:"12/03/12 4:30"
sender:"blue-nile"
Receiver :"4-killo main road"
Message:"Order already arrived"

Restaurant:restaurant1

Restid:rd1256
Name:"blue-nile"
Email:"gprt@gmail.com"
deliveryid:"4-killo"
Address:"sky-light"

Menu:menu2

MenuItemId:5643
Menucategory:"Desserts"
Menuname:"Pizza menu"
Menudescription:"fresh
tomato sauce"
ItemPrice:"birr100"

Order:order1

OrderId:or543
ItemId:I9876
Orderdate:"2/4/2023 2:30"
Customername:"hellopi"
Quantity:"6"
Itemname:"burger,piza,,,"
Companyname:"gprt"
Deliverydate:"12/3/2022
9:00"
Restaurantdetails:" You
can get everything that you
want"
Paymentdetails:"pay by
any currency"

Cart:cart1

ItemID:ur5654
ItemName :"sauce
pizza"
Itemdescription"it
made from tomato
sauce"
Itemprice:"birr200"
Itemquantity:"2"

payment:pay1

PaymentID:4554
OrderID:"8765"
CustomerID:ur567
Currency:"you can pay by any
currency"
Paymentmethod:"paypal"

Delivery:delivery1

DeliveryId:"87654"
Item:"lazagna,chicken,beef"
Quantity:6
Datereceivable:" at6:00"

Figure 0-4 customer0

Customer:customer1

CustomerId:ur967
Username:"solomon sol"
Email:"sol@gmail.com"
address:"6-killo"
Payment:"birr300"

Complaint:complaint1

ComplaintId:ur067
OrderId:"5467"
CustomerId:ur967
Actiontaken:"not delivered yet"
date:"12/03/12 4:30"
type:"order complaint"

Manager:manager1

Id:mg2045
Name:"whitepoint12"
contact:"+12334554"
email:"op@gmail.com"
Name of restaurant:"skylight"

Notification:notifycustomer

NotifyId:NF3327
Type:"notify delvery"
Date and time:"12/03/12 4:30"
sender:"sky light"
Receiver :"6-killo main road"
Message:"Order already arrived"

Restaurant:restaurant1

Restid:rd1256
Name:"sky light"
Email:"gp@gmail.com"
deliveryid:"4-34"
Address:"bole road"

Menu:menu2

MenuItemId:5643
Menucategory:"Desserts"
Menuname:"Pizza menu"
Menudescription:"fresh tomato sauce"
ItemPrice:"birr100"

Order:order1

OrderId:or543
ItemId:l9876
Orderdate:"2/4/2023 2:30"
Customername:"ppop"
Quantity:"6"
Itemname:"burger,piza,,,"
Companyname:"gprt"
Deliverydate:"12/6/2022 5:00"
Restaurantdetails:" You can get everything that you want"
Paymentdetails:"pay by any currency"

Cart:cart1

ItemID:ur5654
ItemName :"sauce pizza"
Itemdescription"it made from tomato sauce"
Itemprice:"birr200"
Itemquantity:"2"

payment:pay1

PaymentID:4554
OrderID:"8765"
CustomerID:ur567
Currency:"you can pay by any currency"
Paymentmethod:"paypal"

Delivery:delivery1

DeliveryId:"87654"
Item:"lazagna,chicken,beef"
Quantity:6
Datereceivable:" at 7:00"

Figure 0-5 sample customer1

Customer:customer2

CustomerId:ur97
Username:"nahom"
Email:"nahom@gmail.com"
address:"6-killo"
Payment:"birr300"

Complaint:complaint1

ComplaintId:ur07
OrderId:"5467"
CustomerId:ur97
Actiontaken:"not delivered yet"
date:"12/03/12 4:30"
type:"order complaint"

Manager:manager2

Id:mg2045
Name:"whitepoint12"
contact:"+12334554"
email:"op@gmail.com"
Name of restaurant:"skylight"

Notification:notifycustomer

NotifyId:NF3327
Type:"notify delvery"
Date and time:"12/03/12 4:30"
sender:"sheraton"
Receiver :"bole main road"
Message:"Order already arrived"

Restaurant:restaurant1

Restid:rd1256
Name:"sky light"
Email:"gp@gmail.com"
deliveryid:"4-34"
Address:"bole road"

Menu:menu3

MenuItemId:5643
Menucategory:"Desserts"
Menuname:"Pizza menu"
Menudescription:"fresh tomato sauce"
ItemPrice:"birr100"

Order:order2

OrderId:or543
ItemId:l9876
Orderdate:"2/4/2023 2:30"
Customername:"ppop"
Quantity:"6"
Itemname:"burger,piza,,,"
Companyname:"gprt"
Deliverydate:"12/6/2022 5:00"
Restaurantdetails:" You can get everything that you want"
Paymentdetails:"pay by any currency"

ItemID:ur5654
ItemName :"sauce pizza"
Itemdescription"it made from tomato sauce"
Itemprice:"birr200"
Itemquantity:"2"

payment:pay1

PaymentID:4554
OrderID:"8765"
CustomerID:ur567
Currency:"you can pay by any currency"
Paymentmethod:"paypal"

Delivery:delivery1

DeliveryId:"87654"
Item:"lazagna,chicken,beef"
Quantity:6
Datereceivable:" at 7:00"

Figure 0-6 sample customer1

Customer:customer2

CustomerId:ur97
Username:"nahom"
Email:"nahom@gmail.com"
address:"6-killo"
Payment:"birr300"

Complaint:complaint1

ComplaintId:ur07
OrderId:"5467"
CustomerId:ur97
Actiontaken:"not delivered yet"
date:"12/03/12 4:30"
type:"order complaint"

Manager:manager2

Id:mg2045
Name:"whitepoint12"
contact:"+12334554"
email:"op@gmail.com"
Name of
restaurant:"skylight"

Notification:notifycustomer

NotifyId:NF3327
Type:"notify delvery"
Date and time:"12/03/12 4:30"
sender:"sheraton"
Receiver :"bole main road"
Message:"Order already arrived"

Restaurant:restaurant1

Restid:rd1256
Name:"sky light"
Email:"gp@gmail.com"
deliveryid:"4-34"
Address:"bole road"

Menu:menu3

MenuItemId:5643
Menucategory:"Desserts"
Menuname:"Pizza menu"
Menudescription:"fresh
tomato sauce"
ItemPrice:"birr100"

Order:order2

OrderId:or543
ItemId:I9876
Orderdate:"2/4/2023 2:30"
Customername:"ppop"
Quantity:"6"
Itemname:"burger,piza,,,"
Companyname:"gprt"
Deliverydate:"12/6/2022
5:00"
Restaurantdetails:" You
can get everything that you
want"
Paymentdetails:"pay by
any currency"

ItemID:ur5654
ItemName :"sauce
pizza"
Itemdescription"it
made from tomato
sauce"
Itemprice:"birr200"
Itemquantity:"2"

payment:pay1

PaymentID:4554
OrderID:"8765"
CustomerID:ur567
Currency:"you can pay by any
currency"
Paymentmethod:"paypal"

Delivery:delivery1

DeliveryId:"87654"
Item:"lazagna,chicken,beef"
Quantity:6
Datereceivable:" at 7:00"

Figure 0-7 sample customer2

Customer:customer3

CustomerId:ur0097
Username:"John"
Email:"jho@gmail.com"
address:"6-killo"
Payment:"birr300"

Complaint:complaint2

ComplaintId:ur07
OrderId:"5467"
CustomerId:ur0097
Actiontaken:"not delivered yet"
date:"12/03/12 4:30"
type:"order complaint"

Manager:manager3

Id:mg2045
Name:"whitepoint12"
contact:"+12334554"
email:"op@gmail.com"
Name of restaurant:"skylight"

Notification:notifycustomer

NotifyId:NF3327
Type:"notify delvery"
Date and time:"12/03/12 4:30"
sender:"hilton"
Receiver :"bole main road"
Message:"Order already arrived"

Restaurant:restaurant2

Restid:rd1256
Name:"hiltone"
Email:"g@gmail.com"
deliveryid:"4-34"
Address:"bole road"

Menu:menu4

MenuItemId:5643
Menucategory:"Desserts"
Menuname:"Pizza menu"
Menudescription:"fresh tomato sauce"
ItemPrice:"birr100"

Order:order3

OrderId:or543
ItemId:I9876
Orderdate:"2/4/2023 2:30"
Customername:"ppop"
Quantity:"6"
Itemname:"burger,piza,,,"
Companyname:"gprt"
Deliverydate:"12/6/2022 5:00"
Restaurantdetails:" You can get everything that you want"
Paymentdetails:"pay by any currency"

Cart:cart3

ItemID:ur5654
ItemName :"sauce pizza"
Itemdescription"it made from tomato sauce"
Itemprice:"birr200"
Itemquantity:"2"

payment:pay1

PaymentID:4554
OrderID:"8765"
CustomerID:ur567
Currency:"you can pay by any currency"
Paymentmethod:"paypal"

Delivery:delivery1

DeliveryId:"87654"
Item:"lazagna,chicken,beef"
Quantity:6
Datereceivable:" at 7:00"

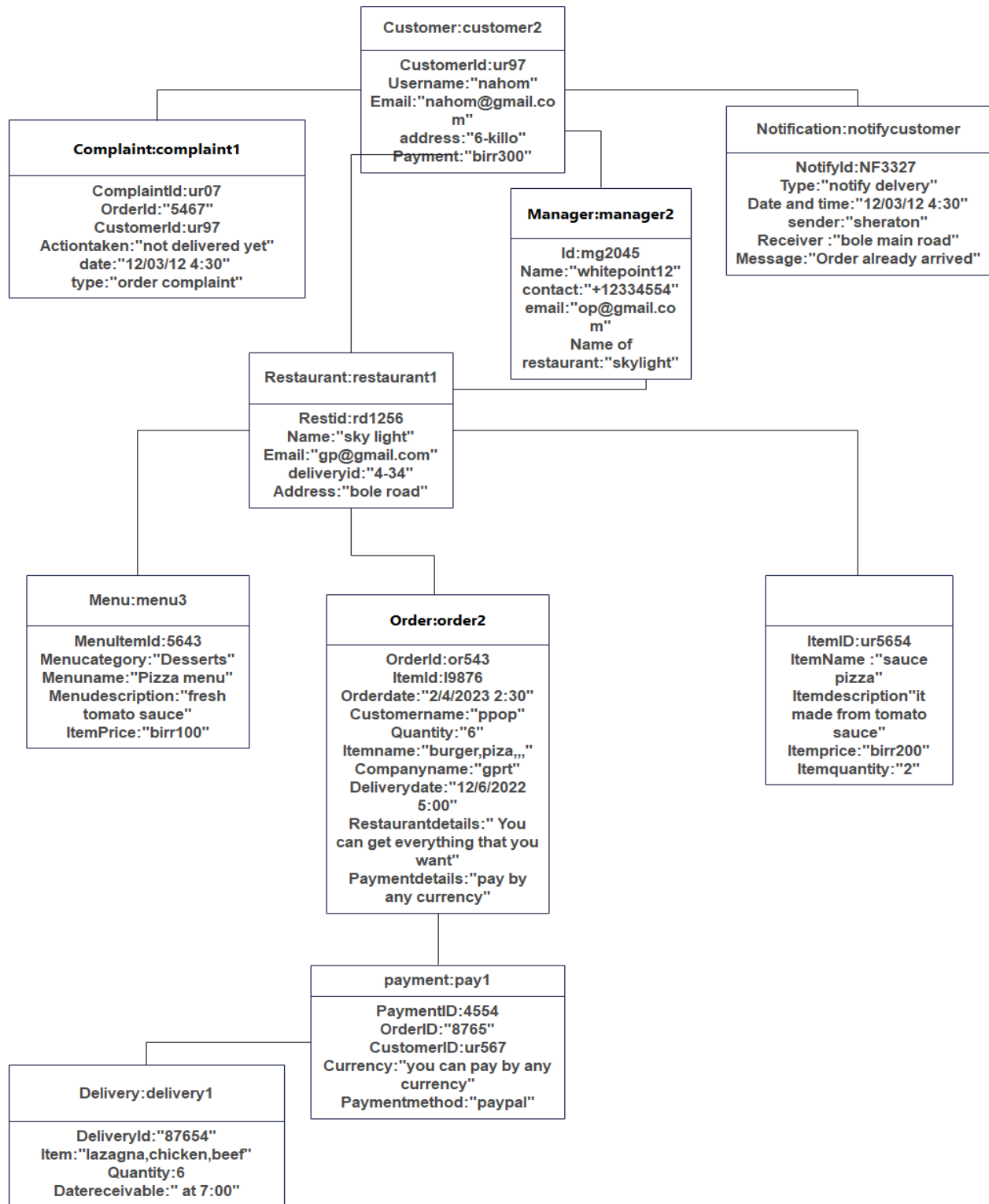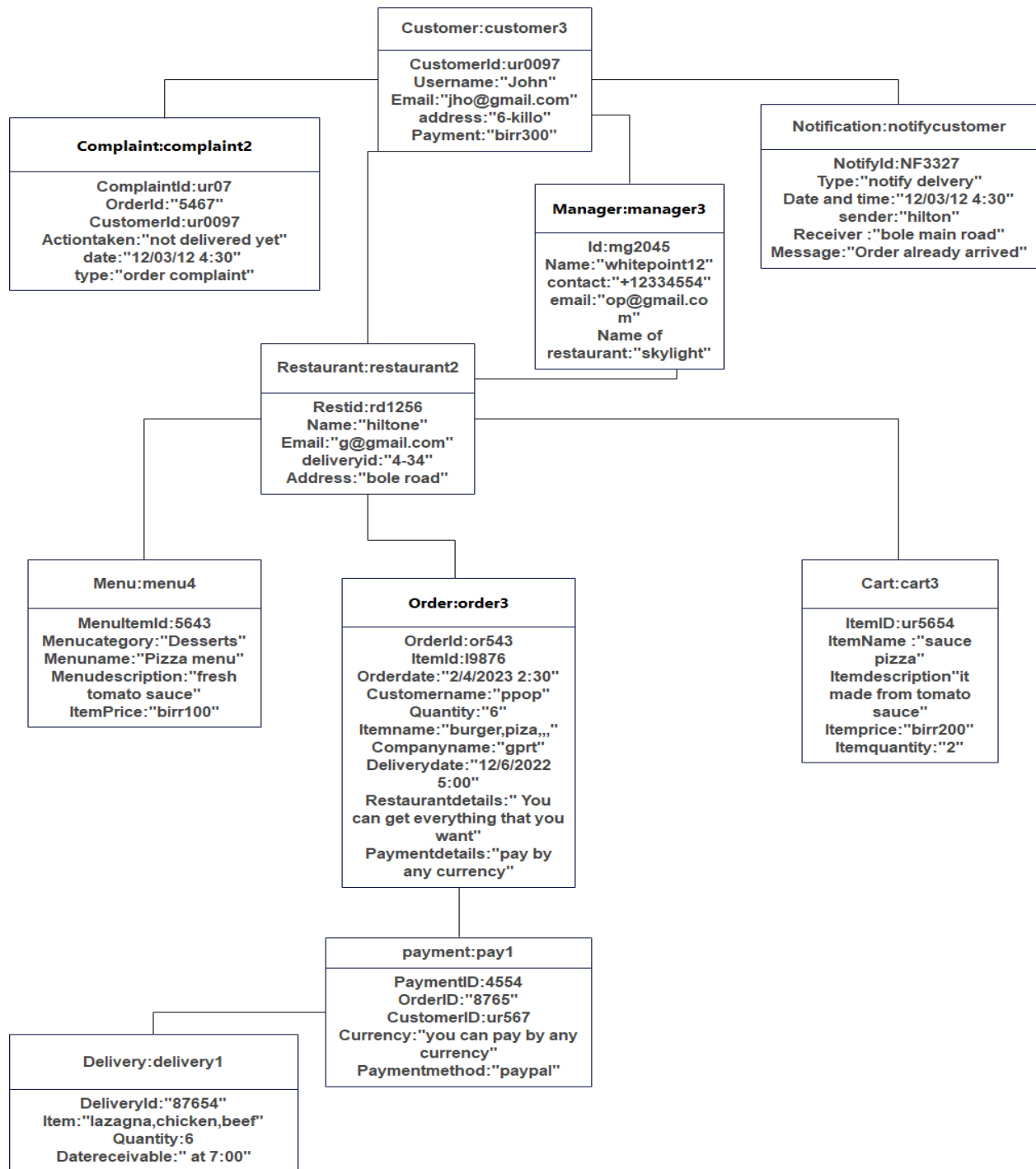**Figure 0-8 sample customer3**

3.9. ACCESS CONTROL AND SECURITY

Access control and security are important considerations when developing a food delivery Android application. Some common access control and security measures that can be implemented include

Access control is a security technique that regulates user grants to access the system's functionalities. And make availability of a system to limit what elements can be viewed or modified by a public or private member and who is allowed to access certain resources of the system. and restrict the modification of data or documents to prevent unauthorized changes, malicious modification of data, and accidental introduction of consistency.

Access control is a security technique that governs user permissions to access system functions. And make a system available to limit what elements can be viewed or modified by a public or private member and who is allowed to access specific system resources. And restrict data or document modification to prevent unauthorized changes, malicious data modification, and unintentional consistency introduction.

Some common access control and security measures that can be implemented include:

- ➢ Secure authentication: Food delivery applications should use secure authentication methods such as two-factor authentication, biometric authentication, and password protection to ensure that only authorized users can access the application.
- ➢ Authorization: This is the process of determining whether a user is allowed to access specific resources or perform specific actions within the application. This can be done by assigning roles and permissions to users, or by using access control lists (ACLs) to specify which users are allowed to access specific resources.
- ➢ Data encryption: All data stored in the application should be encrypted to protect it from unauthorized access.
- ➢ Secure payment processing: Food delivery applications should use secure payment processing methods such as tokenization and encryption to protect customer payment information.
- ➢ Secure data storage: All data stored in the application should be stored securely in a cloud-based server to prevent unauthorized access.
- ➢ Secure communication: All communication between the application and its users should be encrypted to protect it from eavesdropping.
- ➢ Secure access control: Food delivery applications should use access control methods such as role-based access control and user authentication to ensure that only authorized users can access the application.
- ➢ Network security: This is the process of protecting the application and its data from unauthorized access over the network. This can be done by using secure protocols such

as HTTPS, SSL, and SSH, and by implementing firewalls and intrusion detection/prevention systems.

➢ Secure key management: It is important to manage the encryption keys used to encrypt the sensitive data. This can be done by using a Key Management System (KMS) or Hardware Security Module (HSM) to securely store and manage encryption keys.

➢ Regular security auditing: It is important to regularly review and audit the application's security measures to ensure that they are functioning as intended and that any vulnerabilities have been identified and addressed.

➢ Compliance: It's important to follow the regulations, industry standards, and laws that are relevant to your application, such as the Payment Card Industry Data Security Standards (PCI DSS) and the General Data Protection Regulation (GDPR)

➢ Regular software updates: Keeping your application and its dependencies up-to-date can help fix any known vulnerabilities in the software.

These are some of the common security measures that can be implemented in a food delivery android application, but it's important to note that the specific security measures implemented will depend on the requirements of the application and the sensitive data it handles.
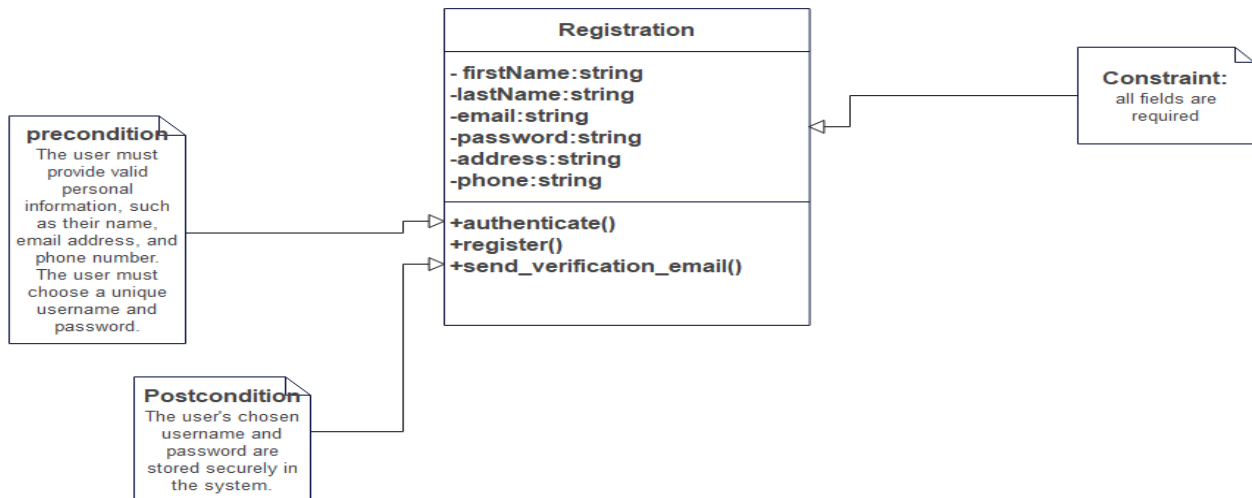
3.10.   DETAILED CLASS DIAGRAM
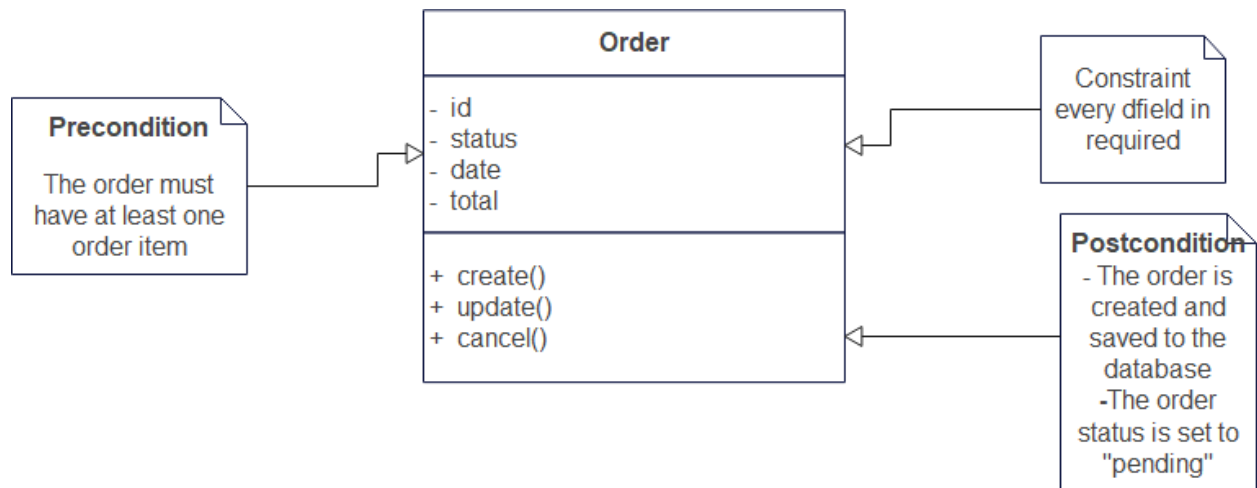

Figure 0-9 Order Detailed Class Diagram

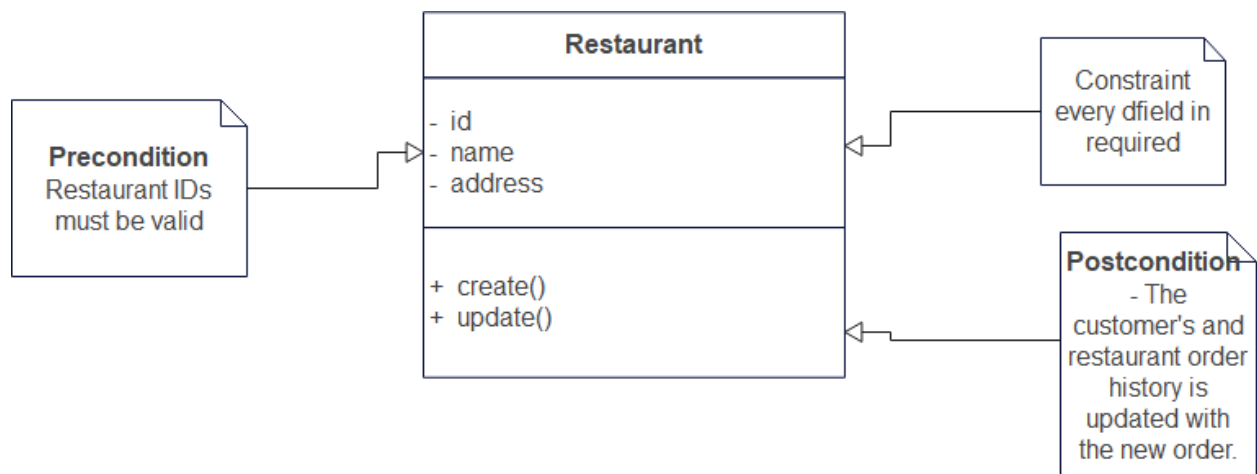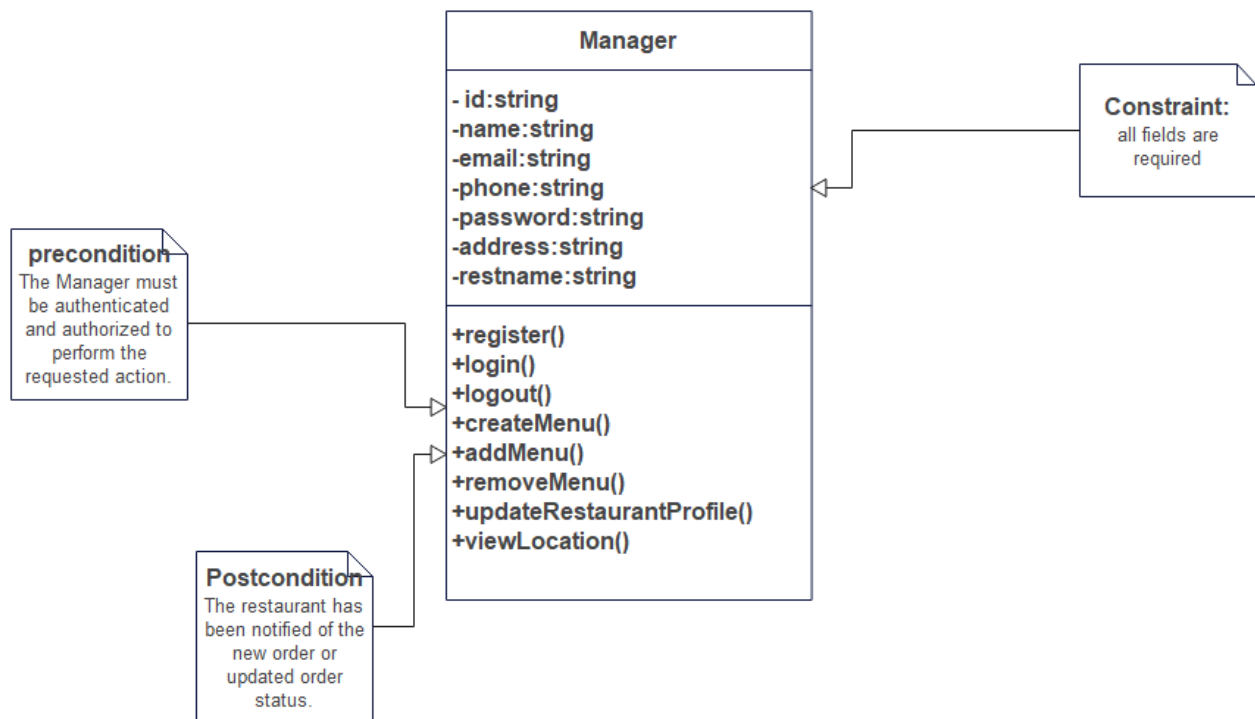**Figure 0-10 Order Detailed Class Diagram**

Figure 0-11 Restaurant Detail Diagram

Figure 0-12 Restaurant Detail Diagram

contraint

userId,userName
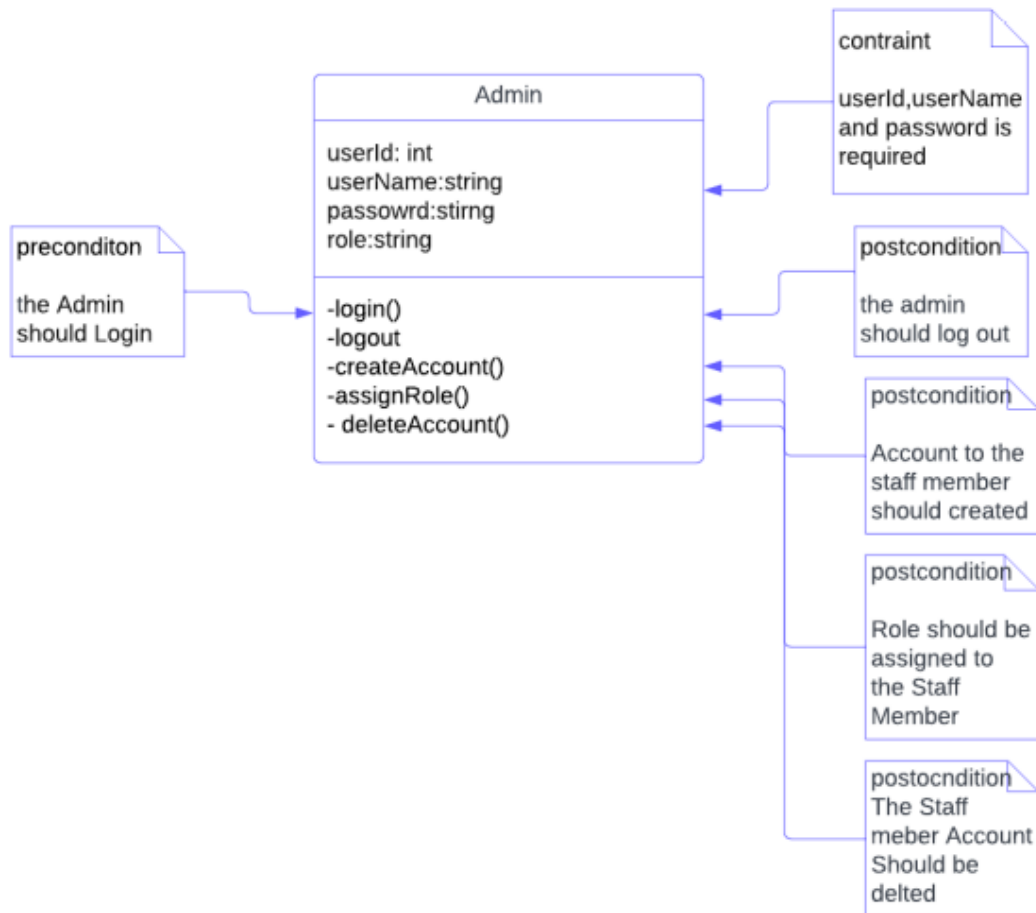and password is
required

Admin

userId: int
userName:string
passowrd:stirng
role:string

-login()
-logout
-createAccount()
-assignRole()
- deleteAccount()

preconditon

the Admin
should Login

postcondition

the admin
should log out

postcondition

Account to the
staff member
should created

postcondition

Role should be
assigned to
the Staff
Member

postocndition
The Staff
meber Account
Should be
delted

Figure 0-13 Admin Detailed Class Diagram

Customer

- id: Int
- firstName: String
- lastName : String
- phoneNumber : Int
- email : String

+ viewServiceDescription()
+ trackIssueStatus()
+ submitIssue()
+ getNotified()

Constraint
All Fields are Required.

Precondition
None

Precondition
Customer should Submit an Issue
Customer should have Ticket number

Precondition
Customer should Insert personal information
which are listed on the class properties of
Customer class and required documents for
the service needed.

Precondition
Customer should Submit an Issue or
customer's Issue is get done.

Post-condition
Information about Services Displayed

Post-condition
Status of the service Displayed

Post-condition
Customer will get notified with Ticket
number via email.

Post-condition
When a new issue is submitted, the
customer will be notified via email with a
Ticket number or with service information
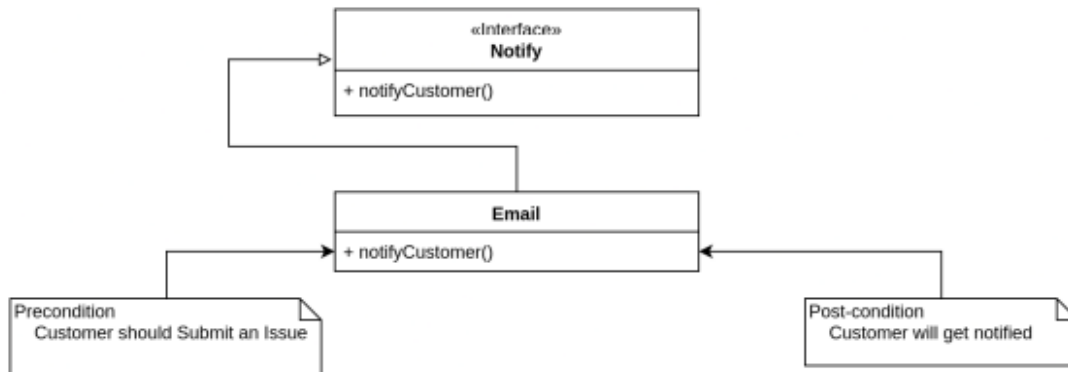that the service has been completed.

Figure 0-14 Customer Detailed Class Diagram

**Figure 0-15 Customer Detailed Class Diagram**

3.11.
3.12. PACKAGES

A package diagram in UML (Unified Modeling Language) is a type of structure diagram that shows the organization of packages and their dependencies. A package diagram for a food delivery Android application may include the following packages:

- ➢ User Interface Package: This package contains the graphical user interface (UI) elements and layouts that the user interacts with.
- ➢ Location Package: This package handles location-based services and maps integration, using APIs like Google Maps.
- ➢ Payment Package: This package implements the payment gateway integration and handling of online transactions.
- ➢ Order Package: This package manages the ordering process, including menu display, ordering, and order tracking.
- ➢ Notification Package: This package implements push notifications for order updates, delivery status, and other relevant information.
- ➢ Database Package: This package manages the database, including storing and retrieving customer information, order details, and delivery information.
- ➢ Backend Package: This package communicates with the backend server for data processing, management, and storage.
- ➢ These packages may have dependencies on each other, for example, the Order Package may depend on the User Interface Package for displaying the menu and ordering process, and on the Database Package for storing and retrieving order information.
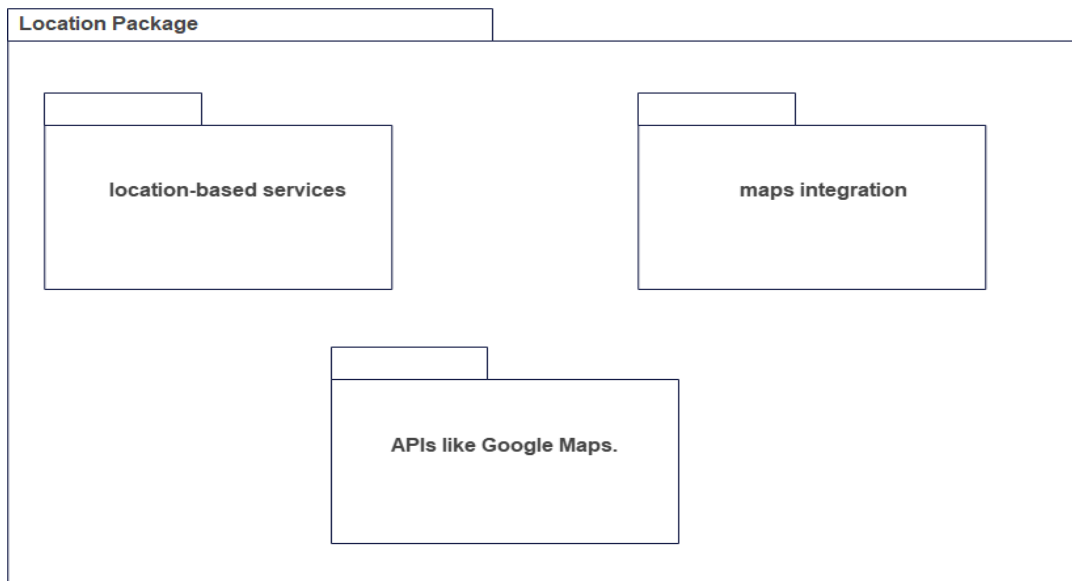 **User Management Package**

This package is responsible for administrator, customer, restaurant and delivery person interaction on user interface to login and It includes account creation and edit/update/delete.
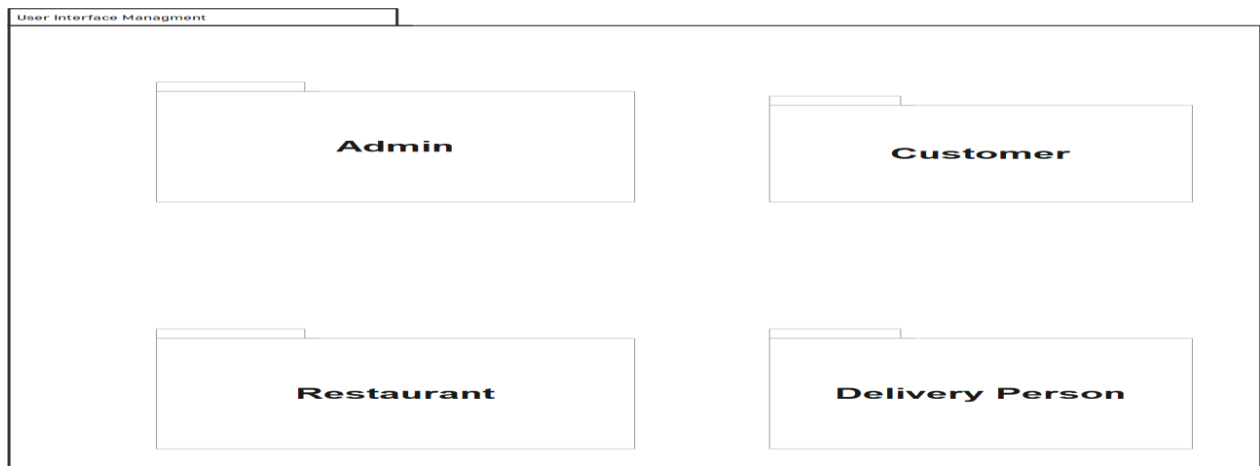
Figure 0-16 User Management Package Diagram

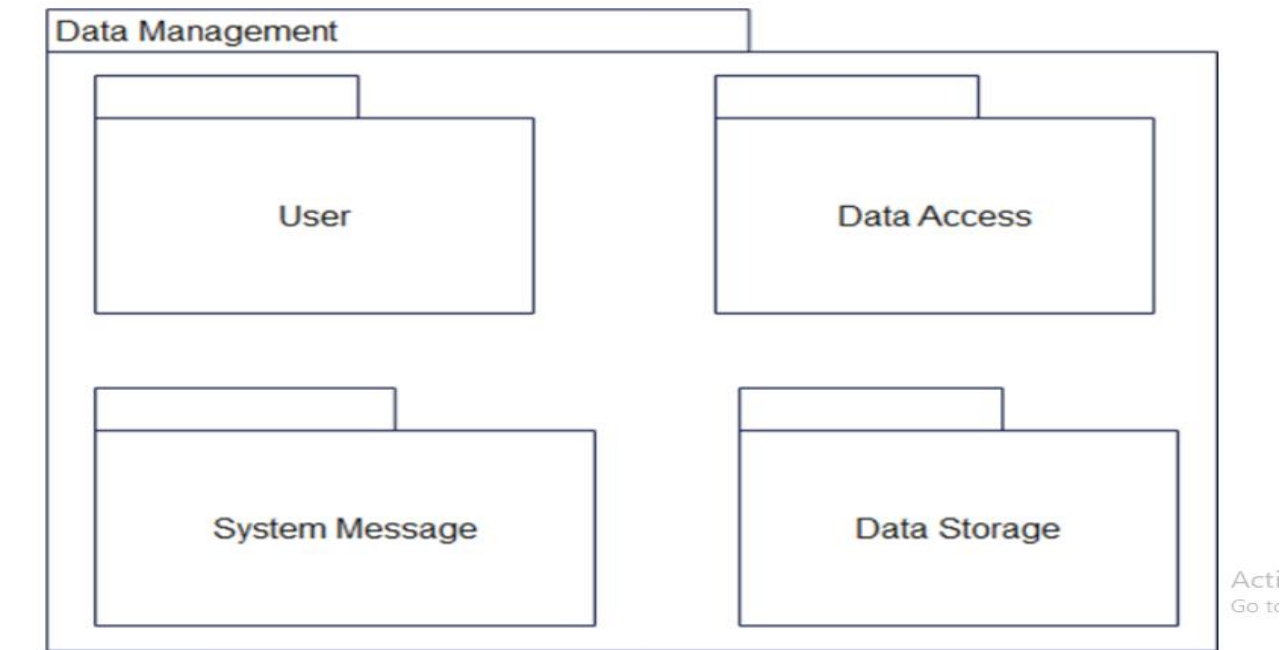## Location management Package

Location Package: This package handles location-based services and maps integration, using APIs like Google Maps.

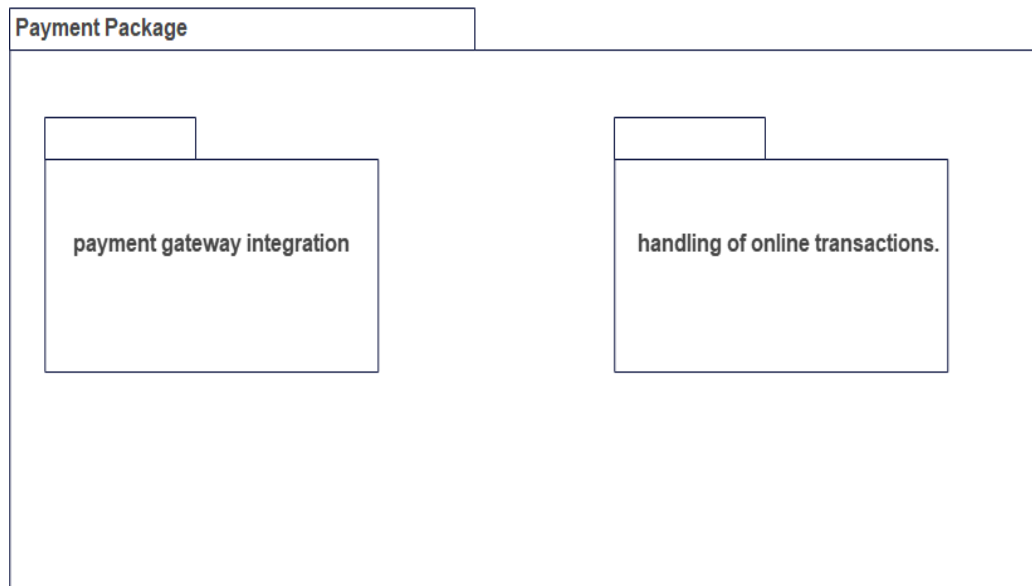**Figure 0-17 Location management Package**

Data Management Package

The data management package contains classes responsible for data storage and information retrieval triggered by the subsystems. The following diagram is data management package of the system

Figure 0-18 Data Management Package
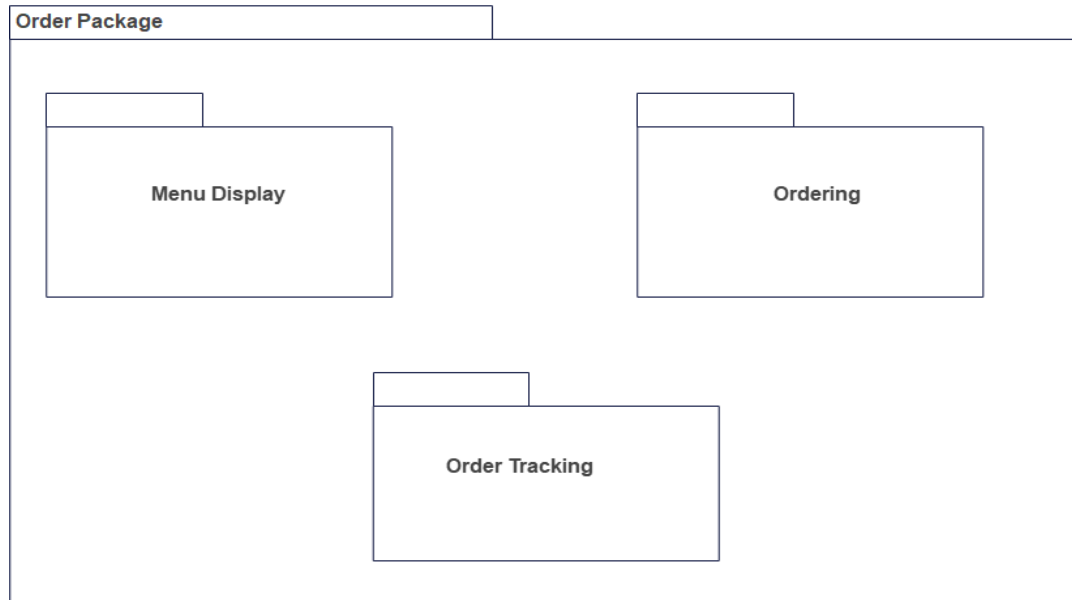
Payment Package

Payment Package: This package implements the payment gateway integration and handling of online transactions

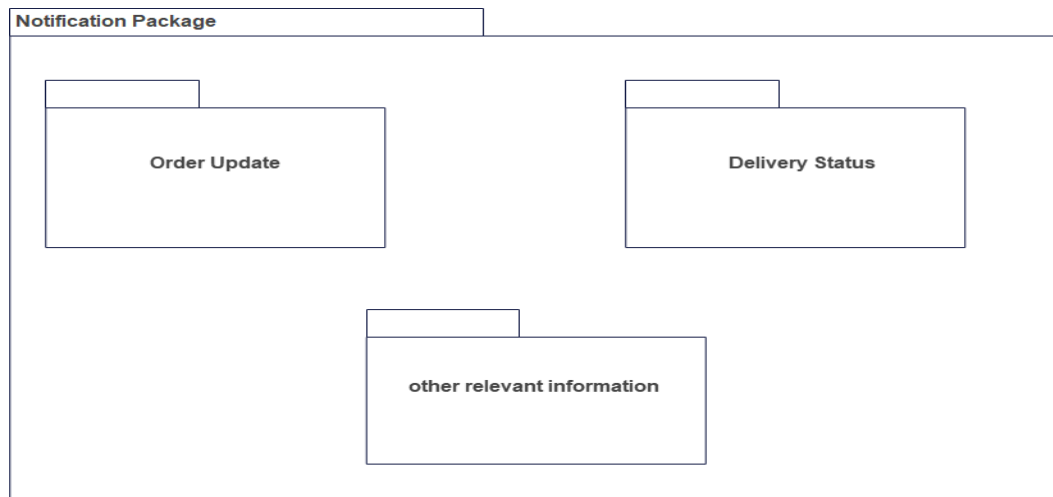**Figure 0-19 Payment Package**

Order Package

Order Package: This package manages the ordering process, including menu display, ordering, and order tracking.

Figure 0-20 Order Package
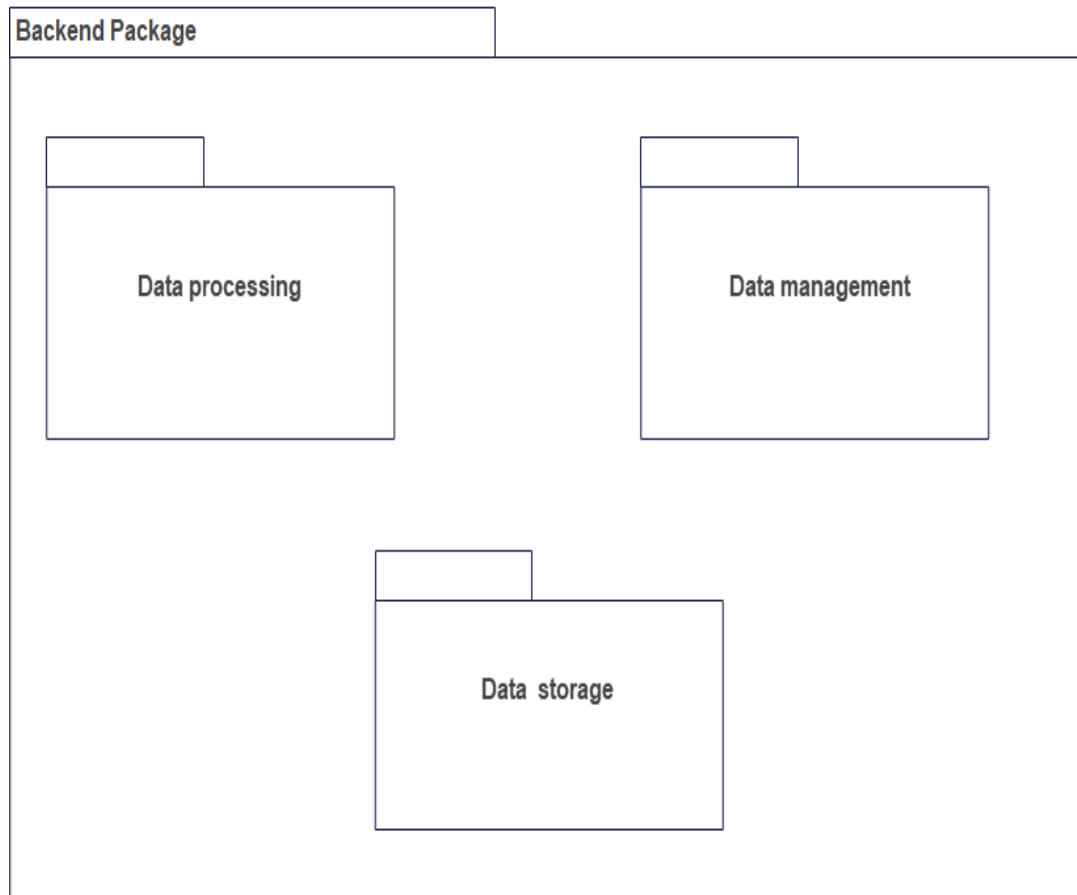
Notification Package

Notification Package: This package implements push notifications for order updates, delivery status, and other relevant information.

Figure 0-21 Notification Package

Backend Package

This package communicates with the backend server for data processing, management, and storage

Figure 0-22 Backend Package

3.13. DEPENDENCY AMONG THEM

- ➢ A package diagram for a food delivery Android application may include the following dependencies:
- ➢ User Interface Package depends on the Order Package, Notification Package, and Location Package for displaying relevant information and updates to the user.
- ➢ Location Package depends on the Google Maps API or other location-based services for mapping and location tracking functionality.
- ➢ Payment Package depends on the Payment Gateway API for handling online transactions.
- ➢ Order Package depends on the User Interface Package for displaying menu and order information, the Payment Package for processing payments, and the Notification Package for sending updates.
- ➢ Notification Package depends on the Order Package and Backend Package for getting information on order status and delivery updates.
- ➢ Database Package depends on the Backend Package for data management and processing, and the Order Package for storing and retrieving order information.
- ➢ Backend Package depends on the Database Package for data storage and management, and the Order Package for processing order information.

These dependencies show the relationships between packages and how they interact and depend on each other to provide a seamless user experience for the food delivery application.
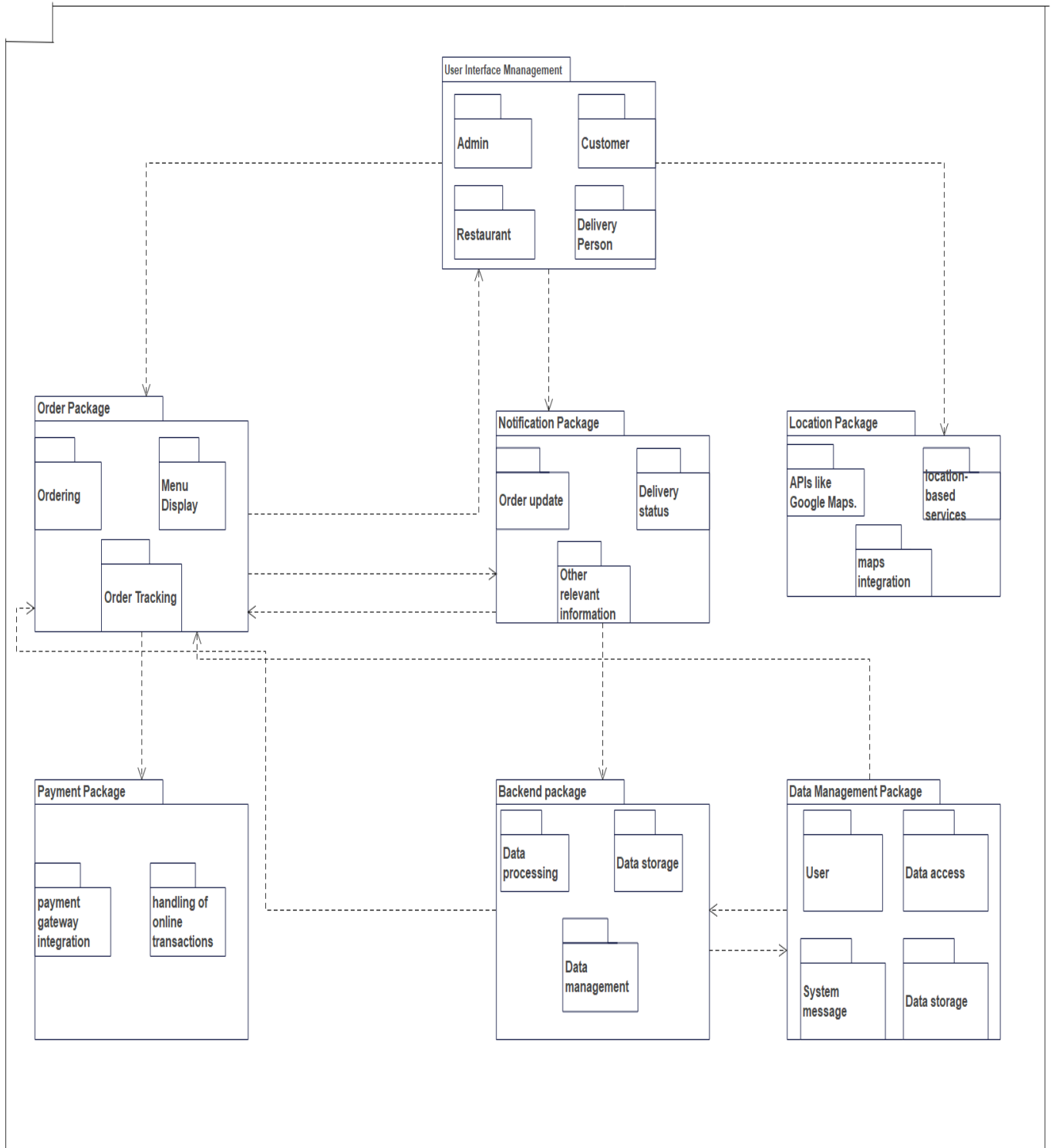
Figure 0-23 Dependency of package