BSIT-F20-015

03-135172-009 UMAR SAEED

03-135172-047 HASSAN HAMID

Autonomous Restaurant Service Robot & Food Ordering App

In partial fulfilment of the requirements for the degree **of Bachelor of Science in Information Technology**

Supervisor: Dawood Akram

Department of Computer Sciences
Bahria University, Lahore Campus

July 2021

Certificate

We accept the work contained in the report titled

"Autonomous Restaurant Service Robot and Food Ordering App"

Written by

UMAR SAEED

HASSAN HAMID

As a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Information Technology.

Approvedby:

Supervisor: Dawood Akram

(Signature)

July 26, 2021

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations, which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	**Name**	**Signature**
03-135172-009	**UMAR SAEED**	
03-135172-047	**HASSAN HAMID**	

Date : July 26, 2021

Specially dedicated to

My beloved mother and father

(UMAR SAEED)

My beloved mother and father

(HASSAN HAMID)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to my project supervisor, Mr. Dawood Akram for his invaluable advice, guidance and his/her enormous patience throughout the development of the research.

In addition, we would also like to express my gratitude to our loving parent and friends who had helped and given us encouragement.

UMAR SAEED
HASSAN HAMID

Autonomous Restaurant Service Robot & Food Ordering App

ABSTRACT

Nowadays, service robots are very popular in robot family. Service Robots have been widely used in many fields. We are designing a restaurant service robot. This restaurant service robot can transfer dishes in the restaurant. This robot could be capable of serving food to the required table of the customer in the restaurant. This robot is based on the characteristics of wheeled robot. The final positioning accuracy of the robot would be 10cm from the table. It is autonomous, has higher payload and is able to travel and dock at the target table.

Our objective is to implement this restaurant service robot in the restaurant. This robot will facilitate the customers. It's a wheeled base robot that will travel on the pre-defined path to the required table. The position of the robot can be determined and path planning can be done.

Target table and roll out the serving tray. With this robot, an android base mobile application will be beneficial for the restaurant owners. Customer would not be wait for the waiter to come on his table, they can simply place their order through app and robot will bring their food on their table.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

We are going to create an autonomous restaurant service robot and a mobile application for taking orders in the restaurant. This robot will take dishes to the customers waiting for their food in the restaurant. When the customers will come into the restaurant, they will open their order taking app and through that app they will order their food. The chef in the kitchen will note their order and cook it for them and then food will be placed on the robot and robot will serve that food to the customers. When the customer logins into App they will see complete food menu and then they can add food into their cart and the total amount will also be visible with the food and once they checkout, the order will be placed. This robot is a onetime investment and that will save many rupees of restaurant owner from giving monthly salaries to the waiters. The Cook will put the dishes in the hands of the robot and the robot will travel on the predefined path and serve on the table.

1.1 Background

An autonomous restaurant service robot are being used in different developed regions of the world like China and Japan. We are implementing a robot in the restaurant which is wheel based and will take dishes/food to the required table. Previously, many restaurant owners face the problem of the availability of the waiters. There are many monthly and annually expenses of the waiters in the restaurant.

This Robot and Application will neglect this problem of the owner. Customers don’t have to wait for the waiter to take their order, they can simply use mobile application for that and the food preparation time will be shown on the application. From this application and robot both the restaurant owner and the customers will be beneficial.

1.2 Problem Statements

Many restaurant owners are facing the problem shortage of waiters. Moreover, many restaurants face the problem of affordability of waiters. They can't afford waiters and their monthly salaries and other expenses. Because of that, they are not making enough profit. To solve this problem, we are introducing this robot and order taking mobile application. It is a onetime investment and might take maintenance expense after some time but this will surely save many monthly expenses of owners. Customers don't have to wait for the waiter to take their order, they can simply use mobile application for that and the food preparation time will be shown on the application. This will save huge amount from the restaurant owner's pocket.

1.3 Aims and Objectives

The objectives and aims of this project are shown as following:

i) To implement robot in a restaurant.

ii) To solve the problems of restaurant owners (Shortage of waiters and their affordability)

iii) To design a mobile application for taking orders in the restaurant.

iv) To design an application which will be beneficial for restaurant owner and customers.

1.4 Scope of Project

This is going to be a complete working autonomous restaurant service robot and android base mobile application. The customer will login into the application and then see the menu on the app and add things into cart and order it, the order will go into the kitchen the chef will note down it and give them preparation time and start preparing their food and lately it will be served through robot. This project will facilitate the customers by taking order through application and providing food on their table in a restaurant. Also provide comfort to the restaurant owner and staff. Robot will save the expenses of the owner and will attract the customers

1.5 Reference

- Proposal Template (FYP OFFICE)

1.6 Intended Audience and Reading Suggestions

- Typical Restaurant users such as their owners and their customers.

CHAPTER 2

SRS & LITERATURE REVIEW

2. Overall Description

This section would provide an overview of the overall system, it also details how the system works and associated system functionality. It will also describe the use cases and characteristics, operating environment, a different type of constraints the assumptions, and dependencies.

2.1 User Classes and Characteristics

The users can be divide into two classes:

- Administrator
- Customer

2.1.1 Administrator

Admin will be the developer who will be able to train the system. The administrator will perform the following activities

- SignIn/Signup
- Change credentials
- View query
- Sigout

2.1.2 Customer

Any person who wants to order from our menu

- Sign in/Signup
- Registration
- Update profile
- Search for item
- Add or remove item in cart
- Check out cart
- Place order
- Payment
- Add or remove item in Wishlist
- Check order status
- Sign out

2.2 Operating Environment

Operating environment for Restaurant Mobile Application & Robot.

- **Operating System:** Windows 7,8,8.1 and 10
- **Software:** Android Studio/ Visual Studio Code /Arduino IDE

2.3 Design and Implementation Constraints

It is an android-based application, which will be developed using Android studio having json, Java, php. Moreover, it will be developed using FDD (Feature driven development) where requirements divided into multiple standalone modules of the software development cycle.

In which first we will build an overall model than build a feature list and then we will plan, design and develop by feature. The project is only implemented to make mobile app so it might take time to open because of synchronizing database with internet on hardware. The internet connection is also a constraint for the application. Since the application fetches data from the database over the internet, it is crucial that there is an internet connection for the application to function.

2.4 Assumptions and Dependencies

Assumptions and dependencies:

- Internet connection must be connected

2.5 System Use Case

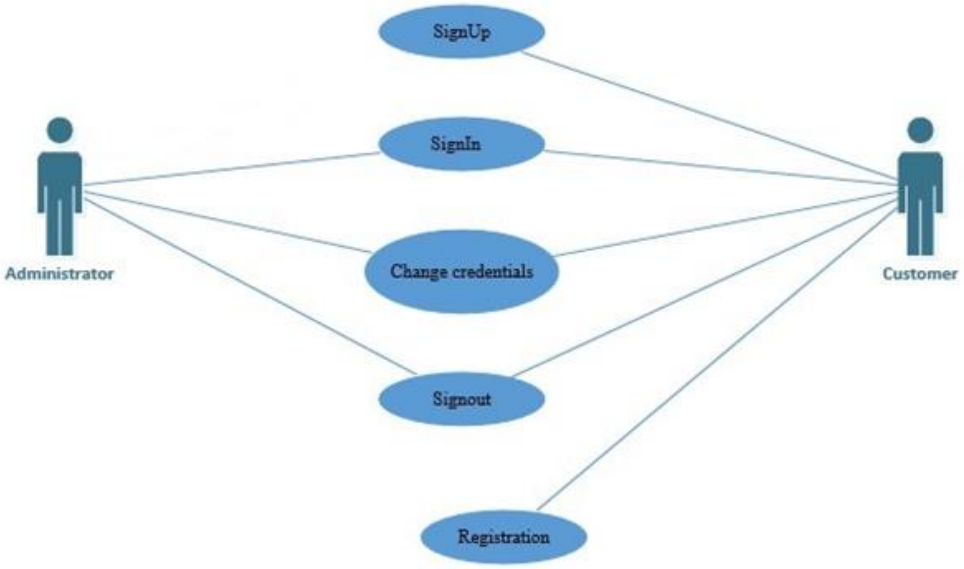


Figure 1: SIGNUP & REGISTRATION

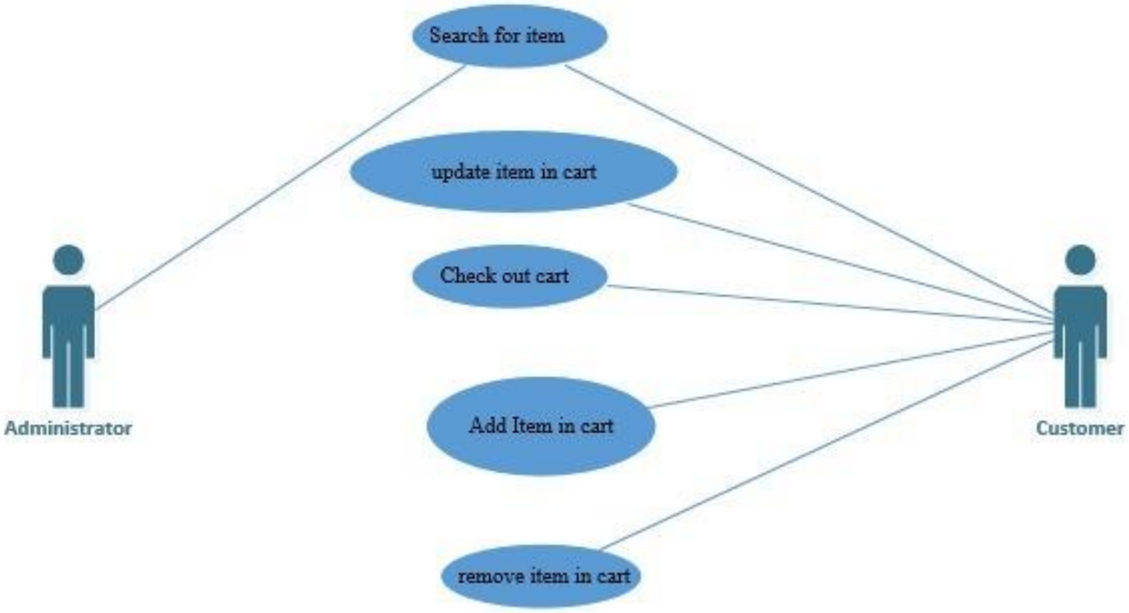


Figure 2: Cart

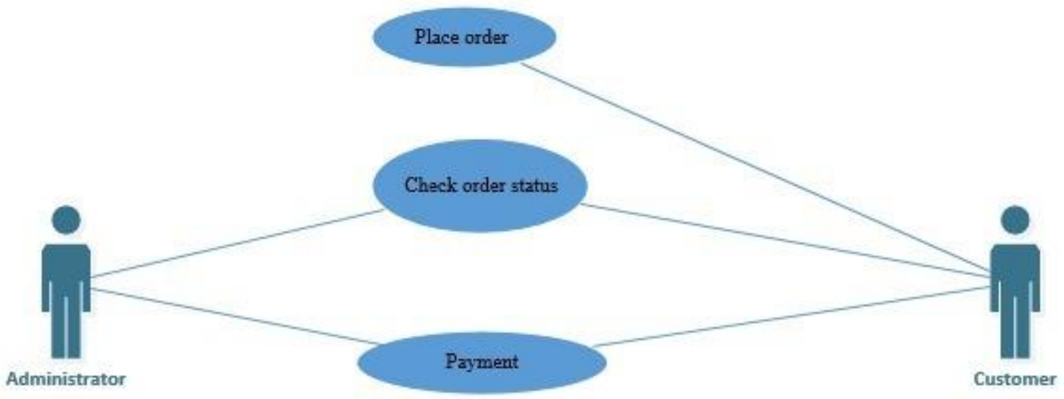


Figure 3: Order & Payment

Administrator:

Table 1: 2.5.1: Sign-in (U1)

	Name	**Sign-in**
1.	**Use-Case ID**	U1
2.	**Objective**	Admin will sign-in with the credentials. (Credentials will be given)
3.	**Priority**	High
4.	**Source**	Hassan Hamid (Developer)
5.	**Actors**	Administrator
6.	**Flow of Events**	1. Admin open the app 2. Enter admin panel URL 3. Enter Username and Password 4. Click on Sign-in Button to access the Administrator Panel
6.1	**Basic Flow**	After successful sign-in Administrator will go to U2, U3, U4, U5 and U6, U7
6.2	**Alternate Flow(s)**	No alternate flow, Administrator must sign-in to proceed further
6.3	**Exception Flow(s)**	• Invalid Username • Invalid Password
7.	**Includes**	No other use case
8.	**Preconditions**	Administrator must be connected to web service via an internet connection (e.g., LAN, WAN)
9.	**Post conditions**	Administrator will use Administrator panel
10.	**Notes/Issues**	If Administrator will sign-in with right credentials no problem will occur

Table 2: 2.5.2 Change Credentials (U2)

	Name	**Change Credentials**
1.	**Use-Case ID**	U2
2.	**Objective**	In this use case Admin can change his/her credentials.
3.	**Priority**	Medium
4.	**Source**	Admin is the main source of this use case. Admin cares most about this functionality.
5.	**Actors**	Admin
6.	**Flow of Events**	1. Login to this Application. 2. Select Change Credentials Option. 3. Enter old password. 4. Enter new password. 5. Re-Enter new password. 6. Click change button.
6.1	**Basic Flow**	After changing credentials Admin will go to U3, U4.
6.2.	**Alternative Flow(s)**	If the Admin does not want to change credentials, he/she will go to U5 and U6.
6.3.	**Exception Flow(s)**	• Invalid old password • Miss match new password.
7.	**Includes**	No
8.	**Preconditions**	Admin must be logged in to perform U2.
9.	**Post conditions**	Credentials are successfully changed.

10.	**Notes/Issues**	No

Table 3: 2.5.3 View Query (U3)

	Name	**View Query**
1.	**Use-Case ID**	U3
2.	**Objective**	In this use case Admin/ view query which is sent by customer.
3.	**Priority**	High.
4.	**Source**	Admin cares most about this functionality and is main source to this use case.
5.	**Actors**	Admin
6.	**Flow of Events**	1. Must fulfill U1. 2. Click view query option.
6.1	**Basic Flow**	Admin will able to view query.
6.2.	**Alternative Flow(s)**	If the Admin does not want to view query, he/she will go to U2 and U4.
6.3.	**Exception Flow(s)**	No.
7.	**Includes**	No Includes.
8.	**Preconditions**	Admin must be logged in.
9.	**Post conditions**	Queries Will be shown to Admin.
10.	**Notes/Issues**	No Notes.

Table 4: 2 .5.4 Sign Out (U4)

	Name	**Logout**
1.	**Use-Case ID**	U4
2.	**Objective**	In this use case Admin Will able to logout.
3.	**Priority**	Medium
4.	**Source**	Admin cares most about this functionality and is main source to this use case.
5.	**Actors**	Admin
6.	**Flow of Events**	1. Must be fulfil U1.
6.1	**Basic Flow**	Logout from Admin panel.
6.2.	**Alternative Flow(s)**	No.
6.3.	**Exception Flow(s)**	No.
7.	**Includes**	No Includes
8.	**Preconditions**	Admin must be logged in this application.
9.	**Post conditions**	Logout from Admin panel.
10.	**Notes/Issues**	No Notes.

Customer:

Table 5: 2.5.5 Sign Out (U4)

	Name	**Customer Sign-in**
1.	**Use-Case ID**	U8
2.	**Objective**	In this use case Customer will be able to sign-in
3.	**Priority**	High
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. User open this app 2. Select sign-in 3. Enter Username and Password 4. Click on Sign-in Button to sign-in
6.1	**Basic Flow**	After successful sign-in Customer will go to U9, U10, U11, U12, U13, U14, U15, U16, U17, U18.
6.2	**Alternate Flow(s)**	No alternate flow, Customer must sign-in to proceed further
6.3	**Exception Flow(s)**	• Invalid Username • Invalid Password
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must have fulfilled U8 and must be connected to via an internet connection.
9.	**Post conditions**	Customer will use Customer panel
10.	**Notes/Issues**	If Customer will sign-in with right credentials no problem will occur

Table 6: 2.5.6 Registration (U9)

	Name	**Registration**
1.	**Use-Case ID**	U 9
2.	**Objective**	In this use case Customer will be able to register
3.	**Priority**	High
4.	**Source**	Customer cares most about this functionality and is main source to this use case.
5.	**Actors**	Customer
6.	**Flow of Events**	1. User open this app 2. Enter Customer registration 3. Select register as Customer 4. Fill out details 5. Click on Register Button to send details to Administrator.
6.1	**Basic Flow**	After successful registration Customer will go to U9, U10, U11, U12, U13, U14, U15, U16, U17, U18.
6.2	**Alternate Flow(s)**	If the Customer is already registered, then go to U8
6.3	**Exception Flow(s)**	• Invalid Details
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be connected to an internet connection
9.	**Post conditions**	Customer will be registered and will go to Sign-in page
10.	**Notes/Issues**	If customer will registration with right credentials no problem will occur

Table 7: 2.5.7 Maintain Profile (U10)

	Name	**Maintain Profile**
1.	**Use-Case ID**	U 10
2.	**Objective**	In this use case Customer will be able to maintain profile
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Do the desired changes 3. Save changes
6.1	**Basic Flow**	Customer will maintain the profile and will go to U11, U12, U13 and U14
6.2	**Alternate Flow(s)**	If the Customer does not want any changes then will go to U11, U12, U13, U14, U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exceptions
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be signed in to perform U10
9.	**Post conditions**	Changes saved
10.	**Notes/Issues**	No notes

Table 8: 2.5.8 Search for item (U11)

	Name	Search for item
1.	**Use-Case ID**	U11
2.	**Objective**	In this use case Customer will be able to search item
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Click on search bar 3. Enter item name customer want to search 4. Get search results
6.1	**Basic Flow**	Customer will Search for item and then will go to U12
6.2	**Alternate Flow(s)**	If the Customer does not want to receive any order, then will go to U12, U13, U14, U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exceptions
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be signed in to perform U11
9.	**Post conditions**	Item searched
10.	**Notes/Issues**	No notes

Table 9: 2.5.9 Add or remove item in cart (U12)

	Name	Add or remove item in cart
1.	**Use-Case ID**	U12
2.	**Objective**	In this use case Customer will be able add or remove item from cart
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Click on item 3. Then click to add to cart button 4. Then click on cart icon 5. Then remove item from cart
6.1	**Basic Flow**	Customers add or remove items from cart and will go to U8, U11, U13 and U14, U15, U16, U17, U18.
6.2	**Alternate Flow(s)**	If the Customer does not want to accept any order, then will go to U13, U14, U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exceptions
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be signed in to perform U12
9.	**Post conditions**	Item added or removed from cart
10.	**Notes/Issues**	No notes

Table 10: 2.5.10 Check out cart (U13)

	Name	Check out cart
1.	**Use-Case ID**	U13
2.	**Objective**	In this use case Customer can Check out cart
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Open item 3. Place order 4. Then check out from shopping cart
6.1	**Basic Flow**	Customer will be able to Check out cart
6.2	**Alternate Flow(s)**	If the Customer does not want to check out from cart, then will go to U14, U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exception
7.	**Includes**	No Includes
8.	**Preconditions**	Must be signed in
9.	**Post conditions**	Successfully check out from cart
10.	**Notes/Issues**	No Notes

Table 11: 2.5.11 Place order (U14)

	Name	Place order
1.	**Use-Case ID**	U14
2.	**Objective**	In this use case Customer can place order
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Open item 3. Click on place order button 4. Fill out details for placing order 5. Then your order is placed
6.1	**Basic Flow**	Customer will be able to place order
6.2	**Alternate Flow(s)**	If the Customer does not want to place, then will go to U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exception
7.	**Includes**	No Includes
8.	**Preconditions**	Must be signed in
9.	**Post conditions**	Order successfully placed
10.	**Notes/Issues**	No Notes

Table 12: 2.5.12 payment (U15)

	Name	payment
1.	**Use-Case ID**	U15
2.	**Objective**	In this use case Customer can pay
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Open item 3. Click on place order button 4. Fill out details for placing order 5. Select payment method 6. Fill out payment information 7. Save changes
6.1	**Basic Flow**	Customer will be able to pay online then will go to U16, U17, U18.
6.2	**Alternate Flow(s)**	If the Customer does not want to pay, then will go to U9, U10, U11, U12 and U13, U14, U15, U16, U17, U18.
6.3	**Exception Flow(s)**	No exception
7.	**Includes**	No Includes
8.	**Preconditions**	Must be signed in
9.	**Post conditions**	Paid successfully

10.	**Notes/Issues**	No Notes

Table 13: 2.5.13 Add or remove item in Wishlist (U16)

	Name	Add or remove item in Wishlist
1.	**Use-Case ID**	U16
2.	**Objective**	In this use case Customer will be able add or remove item from Wishlist
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Food Lancer
6.	**Flow of Events**	1. Must fulfill U8 2. Click on item 3. Then click to heat sign to add or remove item in Wishlist 4. Item added or removed from Wishlist
6.1	**Basic Flow**	Customers add or remove item in Wishlist and will go to U17, U18.
6.2	**Alternate Flow(s)**	If the Customer does not want to Add or remove item in Wishlist, then will go to U17, U18.
6.3	**Exception Flow(s)**	No exceptions
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be signed in to perform U16
9.	**Post conditions**	Item added or removed from Wishlist
10.	**Notes/Issues**	No notes

Table 14: 2.5.14 Check Order Status (U17)

	Name	Check Order Status
1.	**Use-Case ID**	U17
2.	**Objective**	In this use case Customer will be able Check Order Status
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8 2. Click on order button 3. Click on your order to see status
6.1	**Basic Flow**	Customers Check Order Status and will go to U18.
6.2	**Alternate Flow(s)**	If the Customer does not want to Check Order Status, then will go to U18.
6.3	**Exception Flow(s)**	No exceptions
7.	**Includes**	No other use case
8.	**Preconditions**	Customer must be signed in to perform U17
9.	**Post conditions**	Order Status Checked
10.	**Notes/Issues**	No notes

Table 15: 2.5.15 Sign-out (U18)

	Name	Sign-out
1.	**Use-Case ID**	U18
2.	**Objective**	In this use case Customer will be able to sign-out
3.	**Priority**	Medium
4.	**Source**	Customer cares most about this functionality and is main source to this use case
5.	**Actors**	Customer
6.	**Flow of Events**	1. Must fulfill U8
6.1	**Basic Flow**	Sign-out from Customer panel
6.2	**Alternate Flow(s)**	No
6.3	**Exception Flow(s)**	No
7.	**Includes**	No Includes
8.	**Preconditions**	Customer must be signed into Customer Targeted E Commerce Store
9.	**Post conditions**	Sign-out from Customer panel
10.	**Notes/Issues**	No Notes

2.6 Other Nonfunctional Requirements

2.6.1 Performance Requirements

This mobile application comprises on one android base hardware. The hardware should be capable for connecting with the Wi-Fi or cellular data for the successful communication/ordering.

2.6.2 Safety Requirements

The application should remember the presets set and their values of every functionality. The application is develop for human computer interaction and one device would only allow one login at a time.

2.6.3 Security Requirements

The application will not allow any user to access the app without complete authorize login. On entering wrong password, the application will show the message and will not allow the access.

2.6.4 Software Quality Attributes

Following quality attributes will going to be consider while designing and developing the software.

- The application is easily editable.
- The application interface is aesthetic to use which provide easy functionalities and its UI is right according to the universal HCI principles.

2.7 Literature Review

2.7.1 Background

Restaurant Serving robots are recently used in China and Japan. Japan does not have that much a population compare to Pakistan, still, they became the most innovative country in terms of robotic technology, then why not Pakistan. These countries are using robotic services in restaurants in many ways; they are serving sushi through a conveyor belt and serving, robots are used in place of waiters that are navigating through the fixed path to serve.

Increasing labor cost is the main issue in the restaurants that this robotic evolved industry is focusing on. There are different types of robots that different restaurants are using in their particular environment. Moreover, those restaurants are getting fame because of their serving robots around the globe. A study by the Center for an Urban Future found that the automation potential for waiters and waiters is 77%. That figure increases to 87% when you factor in workers that prep food.

An autonomous restaurant service robot is being used in different developed regions of the world like China and Japan. We are implementing a robot in the restaurant which is wheel based and will take dishes/food to the required table. Previously, many restaurant owners face the problem of the availability of the waiters. There are many monthly and annually expenses of the waiters in the restaurant.

This Robot and Application will neglect this problem of the owner. There will be buttons on the back of the robot, which will tell the table number to the robot. The robot will follow black lines as its path. Robot will contain infrared sensors that detect light and help robot to keep on its track. From this robot, both the restaurant owner and the customers will be beneficial.

2.7.2 Tools and Techniques

2.7.2.1 Arduino UNO:

The Arduino UNO is the best board to get started with electronics and coding. If this is your first experience tinkering with the platform, the UNO is the most robust board you can start playing with. The UNO is the most used and documented board of the whole Arduino family.

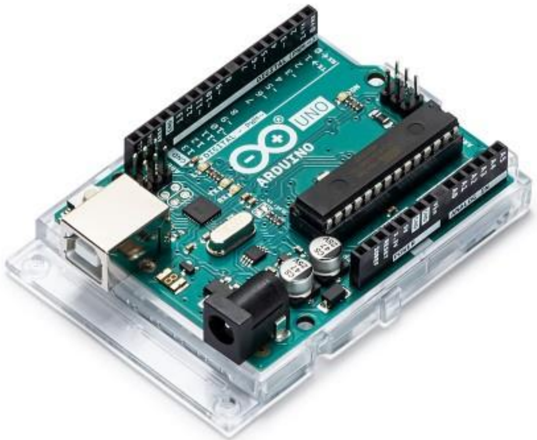
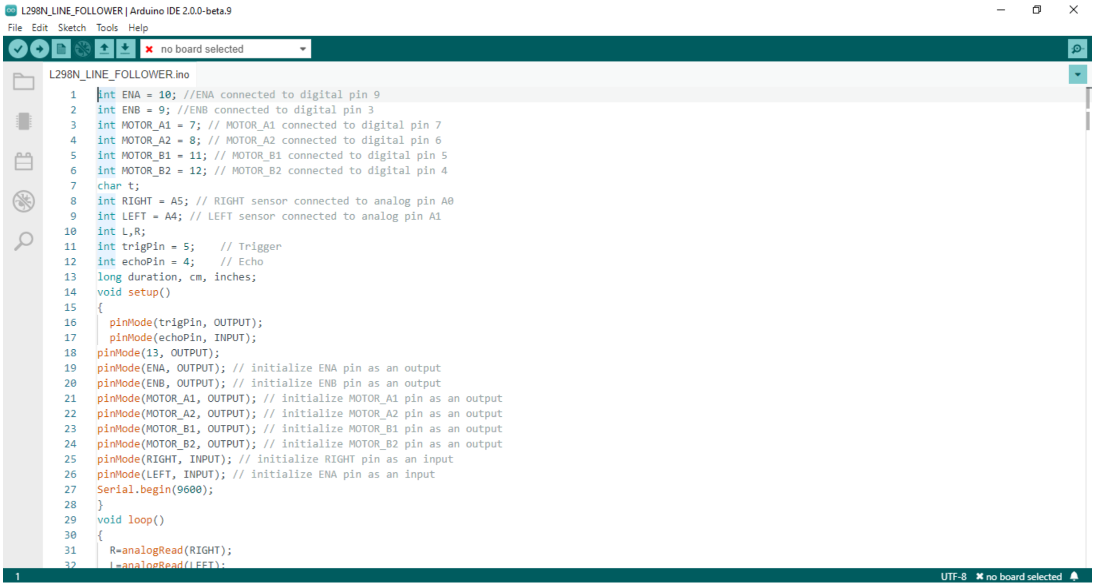


Figure 4: Arduino UNO

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. [1] It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your Uno without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

2.7.2.2 Arduino IDE

The Arduino Integrated Development Environment is a cross-platform application that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but with the help of third-party cores, other vendor development boards. We will use this software for the coding of the robot.



```
int ENA = 10; //ENA connected to digital pin 9
int ENB = 9; //ENB connected to digital pin 3
int MOTOR_A1 = 7; // MOTOR_A1 connected to digital pin 7
int MOTOR_A2 = 8; // MOTOR_A2 connected to digital pin 6
int MOTOR_B1 = 11; // MOTOR_B1 connected to digital pin 5
int MOTOR_B2 = 12; // MOTOR_B2 connected to digital pin 4
char t;
int RIGHT = A5; // RIGHT sensor connected to analog pin A0
int LEFT = A4; // LEFT sensor connected to analog pin A1
int L,R;
int trigPin = 5;    // Trigger
int echoPin = 4;    // Echo
long duration, cm, inches;
void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
pinMode(13, OUTPUT);
pinMode(ENA, OUTPUT); // initialize ENA pin as an output
pinMode(ENB, OUTPUT); // initialize ENB pin as an output
pinMode(MOTOR_A1, OUTPUT); // initialize MOTOR_A1 pin as an output
pinMode(MOTOR_A2, OUTPUT); // initialize MOTOR_A2 pin as an output
pinMode(MOTOR_B1, OUTPUT); // initialize MOTOR_B1 pin as an output
pinMode(MOTOR_B2, OUTPUT); // initialize MOTOR_B2 pin as an output
pinMode(RIGHT, INPUT); // initialize RIGHT pin as an input
pinMode(LEFT, INPUT); // initialize ENA pin as an input
Serial.begin(9600);
}
void loop()
{
  R=analogRead(RIGHT);
  L=analogRead(LEFT);
```

Figure 5: Arduino IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuine hardware to upload programs and communicate with them.

2.7.2.3 Arduino BlueControl:

Arduino Bluetooth Control is an application that allows you to control your arduino board (and similar boards) via Bluetooth, and so to create awesome and fully customized projects, with the new features available within the app.

The settings section allows you to adapt the application to your needs, through a very simple and intuitive interface.

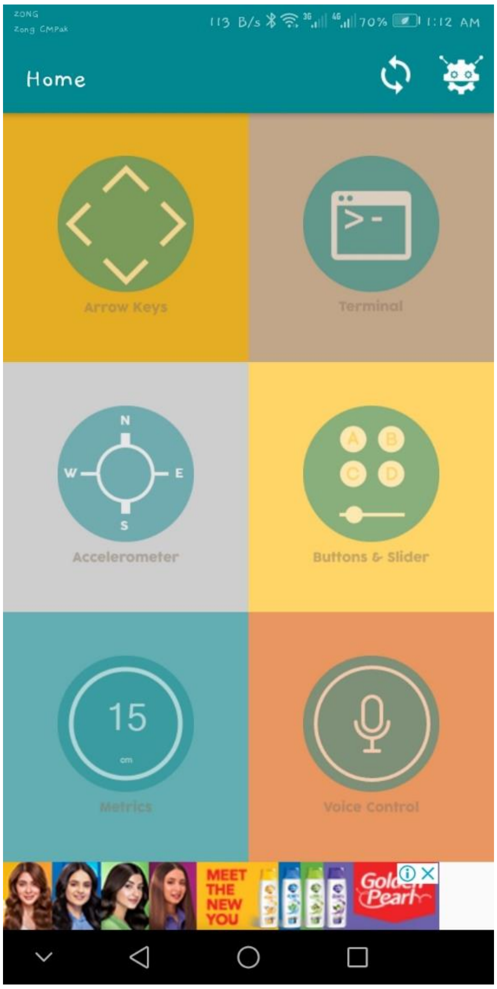


Figure 6: Arduino BlueControl

2.7.2.4 IR Sensor:

An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. We will be using IR Module is sensor circuit that consists IR LED/photodiode pair, potentiometer, LM358, resistors and LED. IR sensor transmits Infrared light and photodiode receives the infrared light. [2]

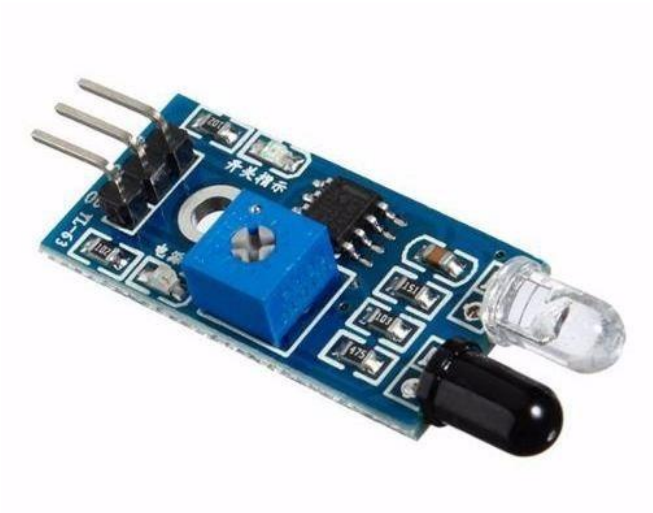


Figure 7: IR Sensors

There are two types of infrared sensors: active and passive. Active infrared sensors both emit and detect infrared radiation. Active IR sensors have two parts: a light emitting diode (LED) and a receiver. When an object comes close to the sensor, the infrared light from the LED reflects off of the object and is detected by the receiver.

2.7.2.4.1 Working Principle of IR Sensor:

IR LED is one kind of transmitter that emits IR radiations. This LED looks similar to a standard LED and the radiation which is generated by this is not visible to the human eye. Infrared receivers mainly detect the radiation using an infrared transmitter. These infrared receivers are available in photodiodes form. IR Photodiodes are dissimilar as compared with usual photodiodes because they detect simply IR radiation. Different kinds of infrared receivers mainly exist depending on the voltage, wavelength, package, etc.

Once it is used as the combination of an IR transmitter & receiver, then the receiver’s wavelength must equal the transmitter. Here, the transmitter is IR LED whereas the receiver is IR photodiode. The infrared photodiode is responsive to the infrared light that is generated through an infrared LED. The resistance of photo-diode & the change in output voltage is in proportion to the infrared light obtained. This is the IR sensor’s fundamental working principle.

2.7.2.5 Ultrasonic Sensor:

Ultrasonic sensors can be used to complete the most complex tasks involving the detection of objects. Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. [3] Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

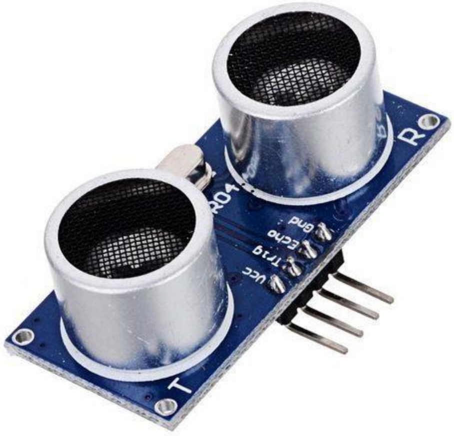


Figure 8: Ultrasonic Sensor

2.7.2.5.1 Features

The following list shows typical characteristics enabled by the detection system.

1. [Transparent object detectable]

   Since ultrasonic waves can reflect off a glass or liquid surface and return to the sensor head, even transparent targets can be detected.
2. [Resistant to mist and dirt]

   Detection is not affected by accumulation of dust or dirt.
3. [Complex shaped objects detectable]

Presence detection is stable even for targets such as mesh trays or springs.

2.7.2.6 L298N Motor Driver

At the heart of the module is the big, black chip with chunky heat sink is an L298N.

The L298N is a dual-channel H-Bridge motor driver capable of driving a pair of DC motors. [4] That means it can individually drive up to two motors making it ideal for building two-wheel robot platforms.

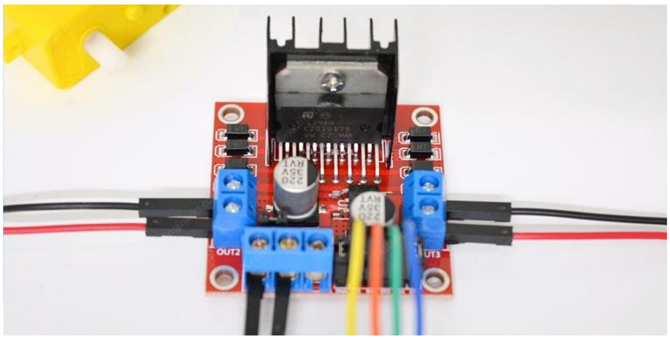


Figure 9: L298N Motor Driver

2.7.2.7 DC Motor

Power battery unit is the central unit of the robot. It consists of DC motors. A DC motor converts DC electrical energy into mechanical energy. When a current-carrying conductor is placed in a magnetic field, it experiences a torque and has a tendency to move. In other words, when a magnetic field and an electric field interact, a mechanical force is produced. The DC motor or direct current motor works on that principle. [5] This is known as motoring action.

2.7.2.8 LITHIUM ION (LI-ION) BATTERY:

LITHIUM ION (LI-ION) BATTERY 3 CELL 11.1V 2200 MAH. [6] These are high performance, high current discharge batteries best for all-terrain robots, combat robots, autonomous robots and other electronic projects.

2.7.2.8.1 Specification:

The battery Manufacturer is Amptex, Battery Voltage is 11.1V and The Battery capacity is 2.2 Ah. It has Barrel-Jack end Connector.

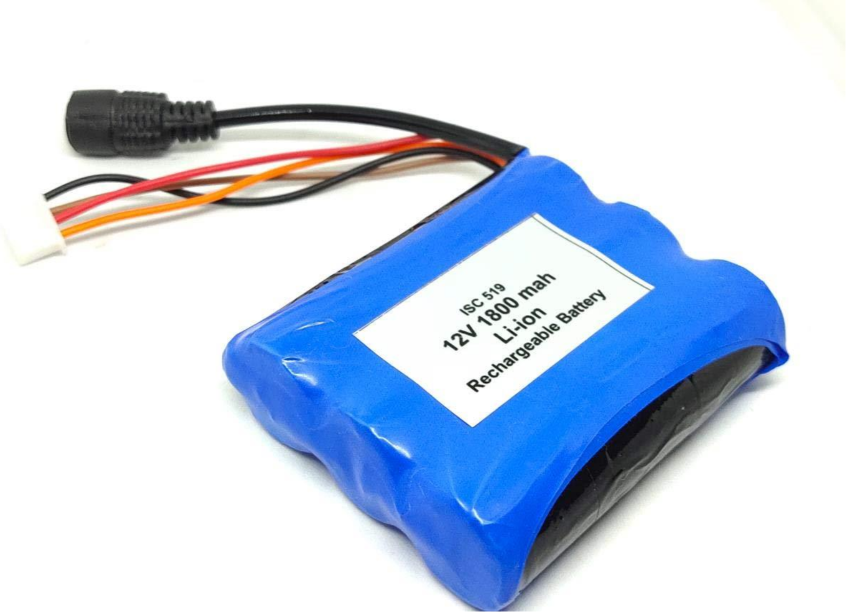


Figure 10: LITHIUM ION (LI-ION) BATTERY 3 CELL 11.1V 2200 MAH.

2.7.2.9: HC 05 Bluetooth Module

HC-05 Bluetooth Module TTL Module (With EN Pin) Breakout is the latest Bluetooth wireless serial Module. This version of the popular Bluetooth module uses the HC-05. These modems work as a serial (RX/TX) pipe. [7] Any serial stream from 9600 to 115200bps can be passed seamlessly from your computer to your target. The remote unit can be powered from 3.3V up to 6V for easy battery attachment. All signal pins on the remote unit are 3V-6V tolerant. No level shifting is required.

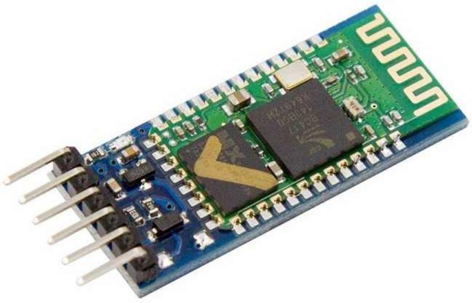


Figure 11: HC 05 Bluetooth Module

2.7.3 Miscellaneous

2.7.3.1 Jumper Wires:

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed. Fairly simple. In fact, it doesn't get much more basic than jumper wires.

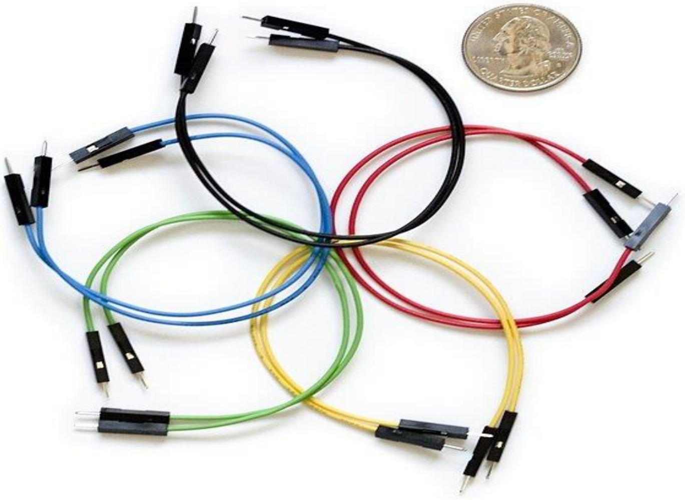


Figure 12: Jumper Wires

CHAPTER 3

SOFTWARE DESIGN AND METHODOLOGY

3.1 Domain Model

Domain Modeling is a way to describe and model real world entities and the relationships between them, which collectively describe the problem domain space.

Below Fig. shows the domain model of Autonomous Restaurant Service Robot & Food Ordering App.

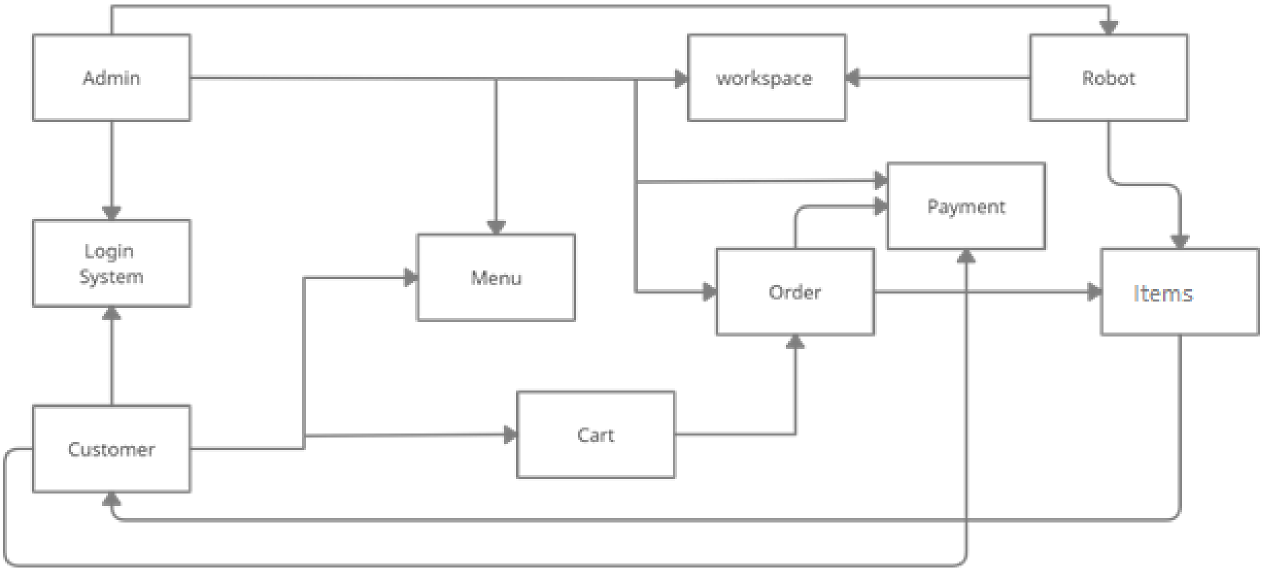


Figure 13: Domain Model

3.2 Sequence Diagram

A sequence diagram, in the context of UML, represents object collaboration and is used to define event sequences between objects for a certain outcome. A sequence diagram is an essential component used in processes related to analysis, design and documentation. A sequence diagram is also known as a timing diagram, event diagram and event scenario. The main purpose of a sequence diagram is to define event sequences that result in some desired outcome. The focus is less on messages themselves and more on the order in which messages occur; nevertheless, most sequence diagrams will communicate what messages are sent between a system's objects as well as the order in which they occur.

The diagram conveys this information along the horizontal and vertical dimensions: the vertical dimension shows, top down, the time sequence of messages/calls as they occur, and the horizontal dimension shows, left to right, the object instances that the messages are sent to.

Following figures show the Sequence Diagrams of the Food Ordering App:

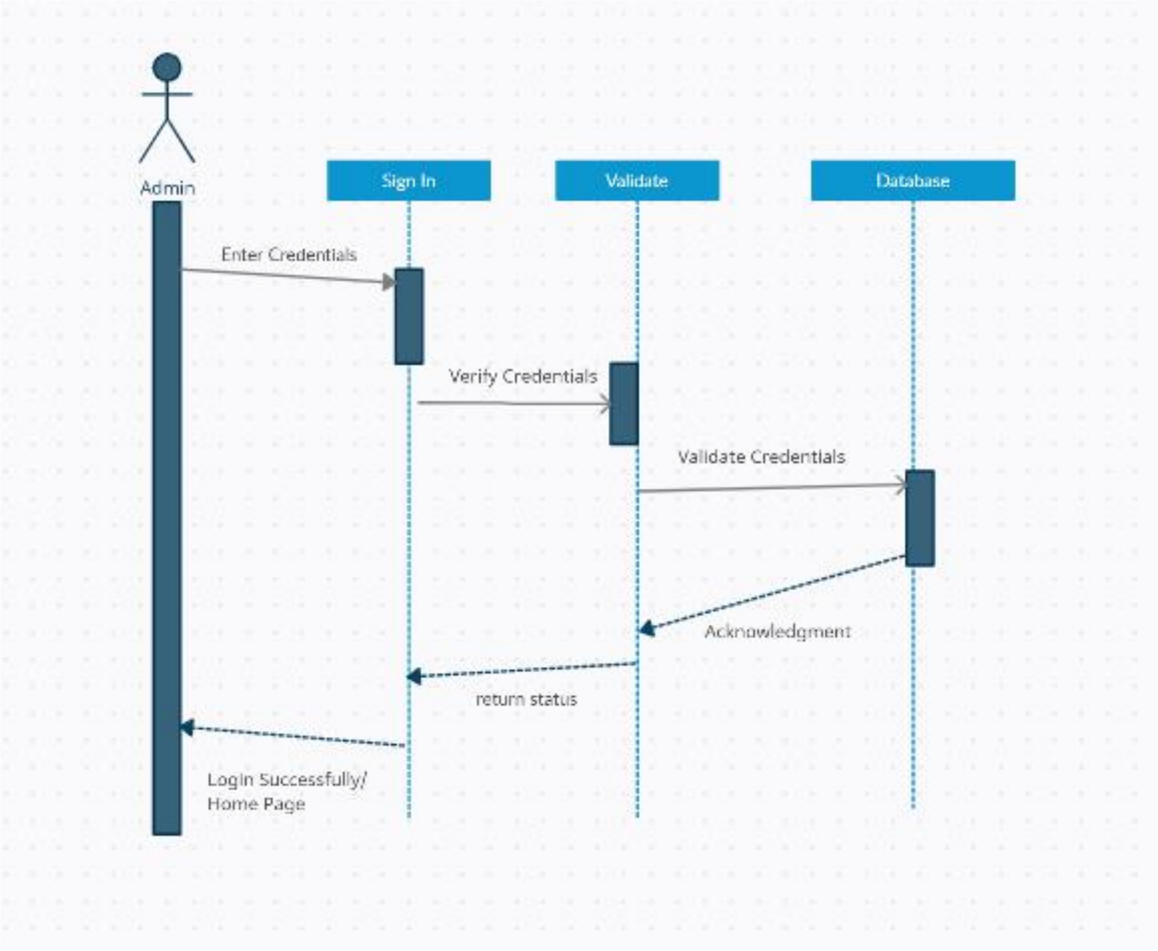


Figure 14: Admin Login Sequence Diagram

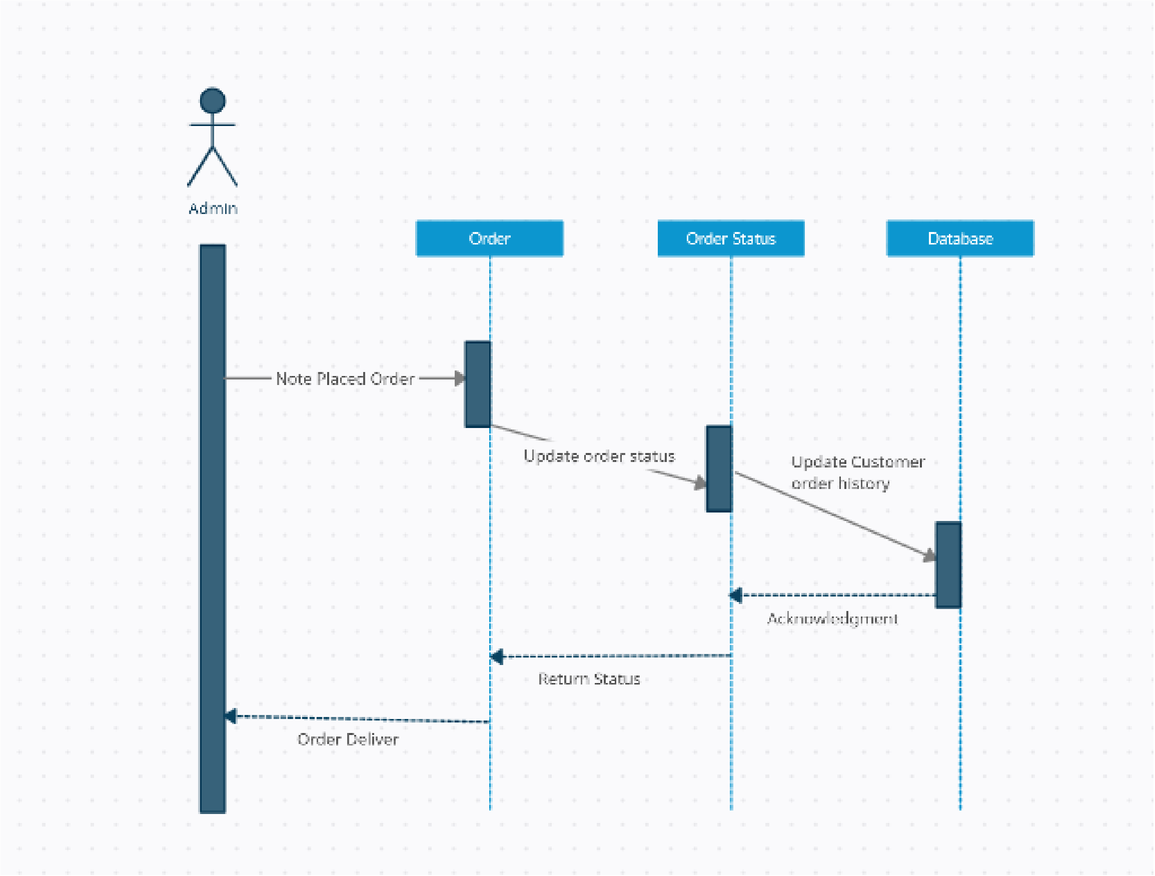


Figure 15: Admin Order Management Sequence Diagram

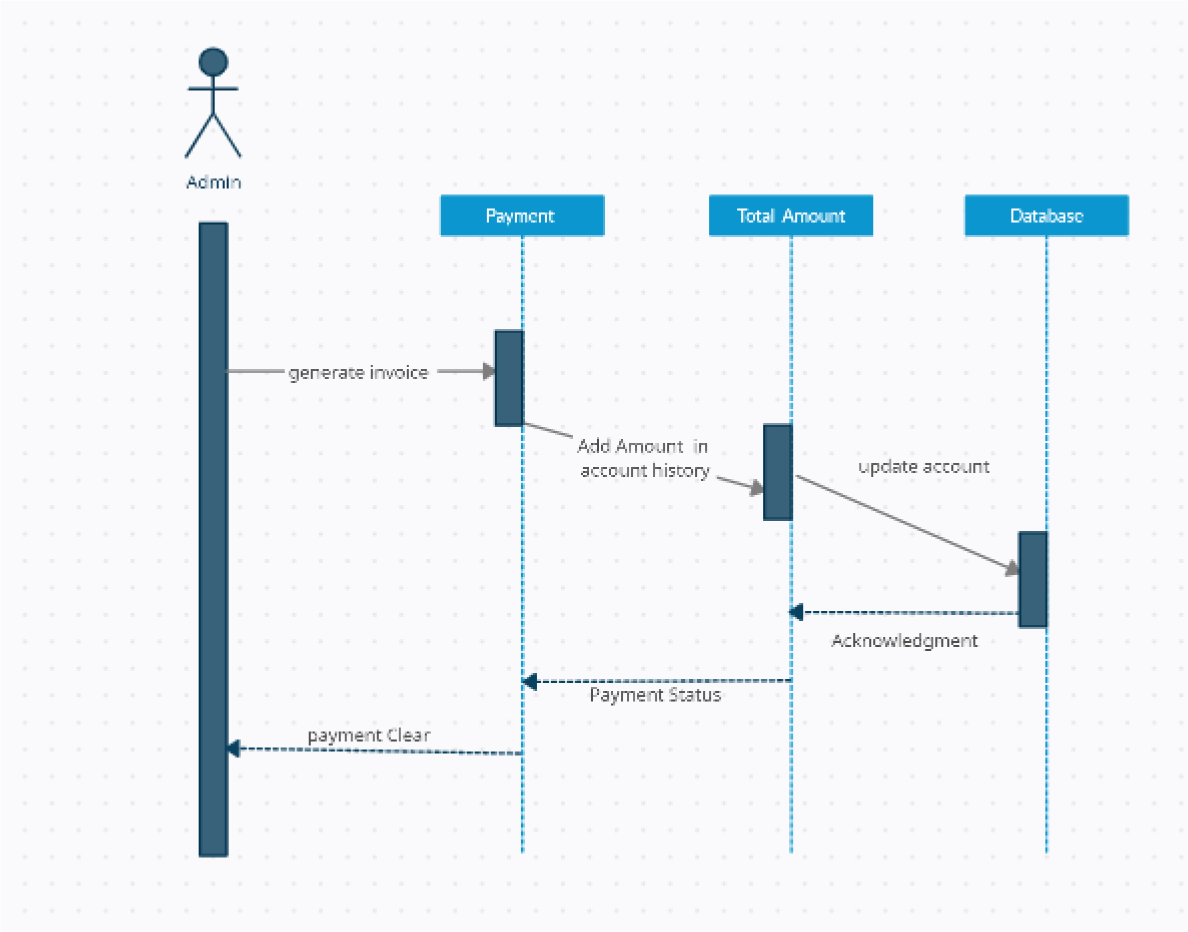


Figure 16: Admin Payment Sequence Diagram

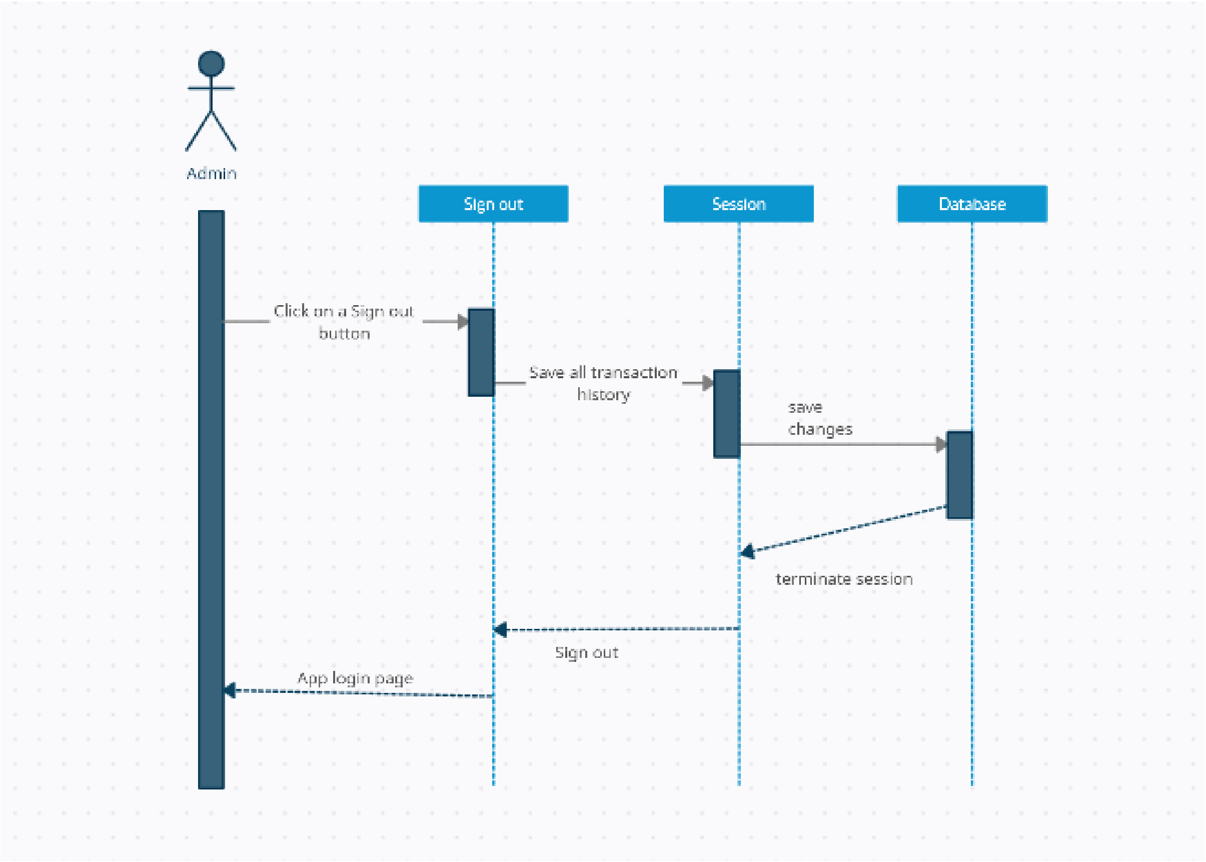


Figure 17: Admin Sign out Sequence Diagram

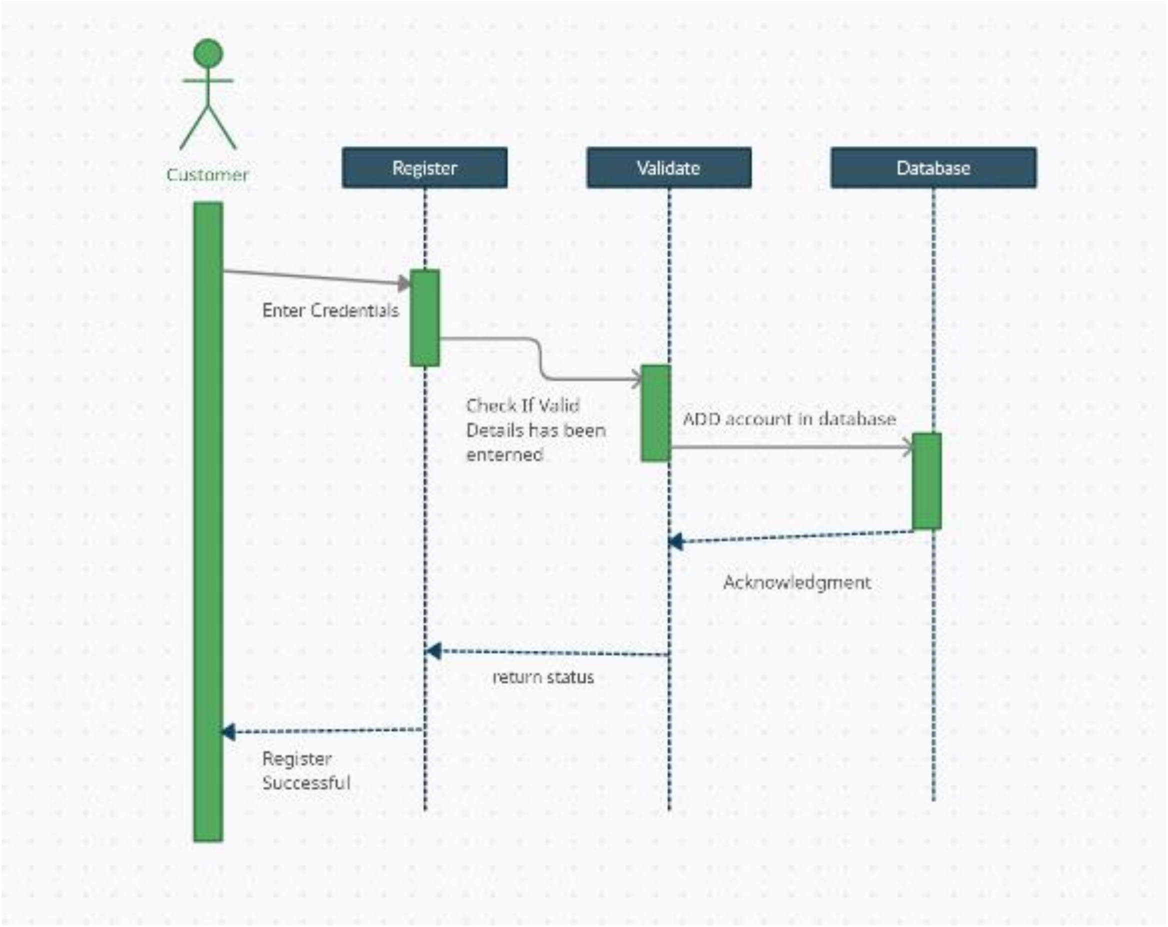


Figure 18: Customer Registration Sequence Diagram

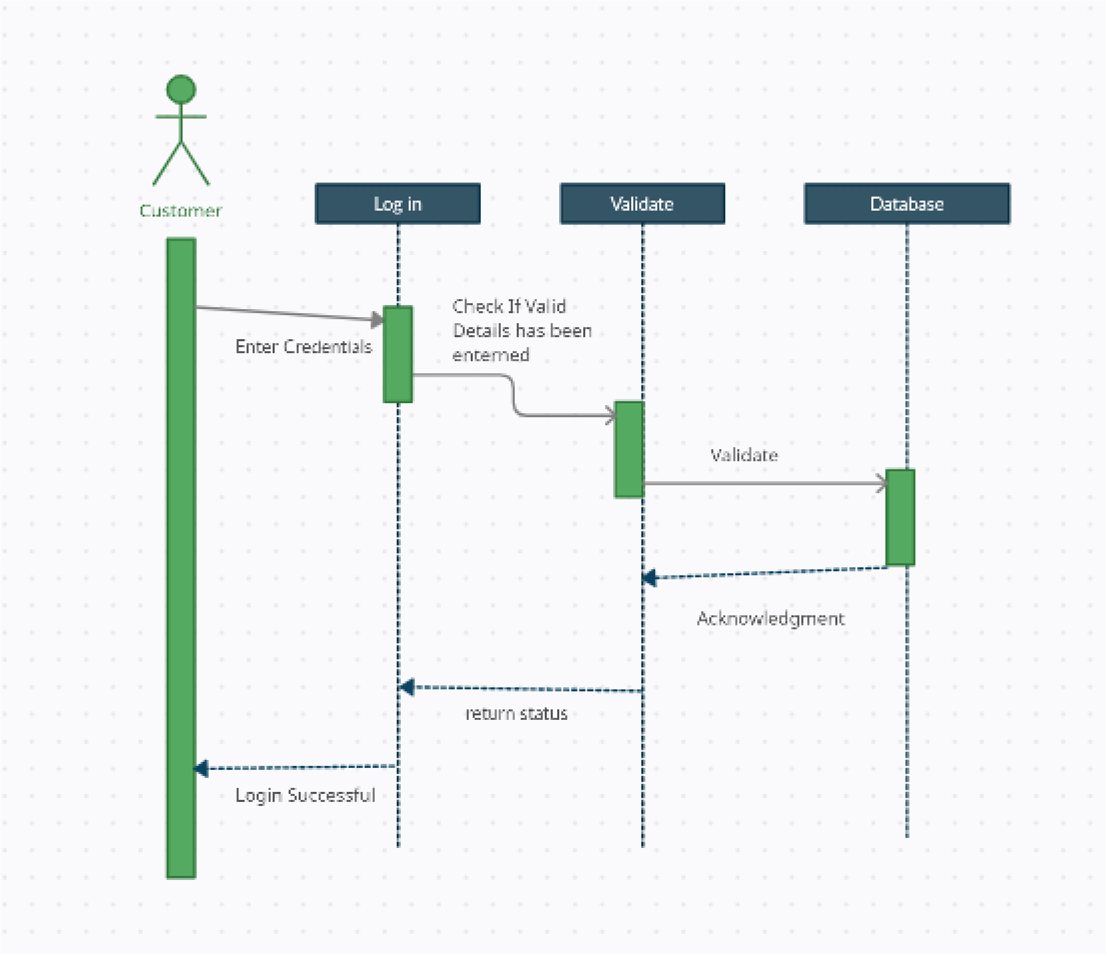


Figure 19: Customer Login Sequence Diagram

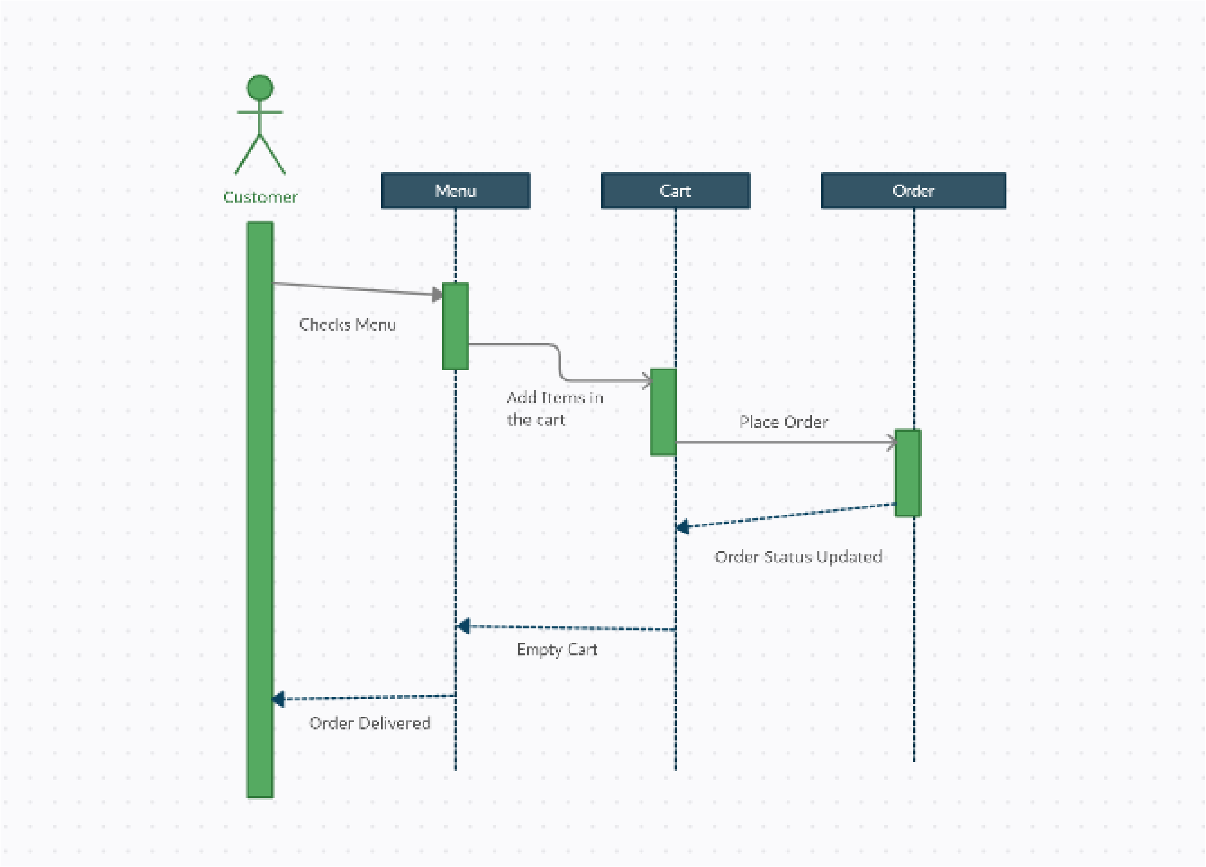


Figure 20: Customer Order Placing Sequence Diagram

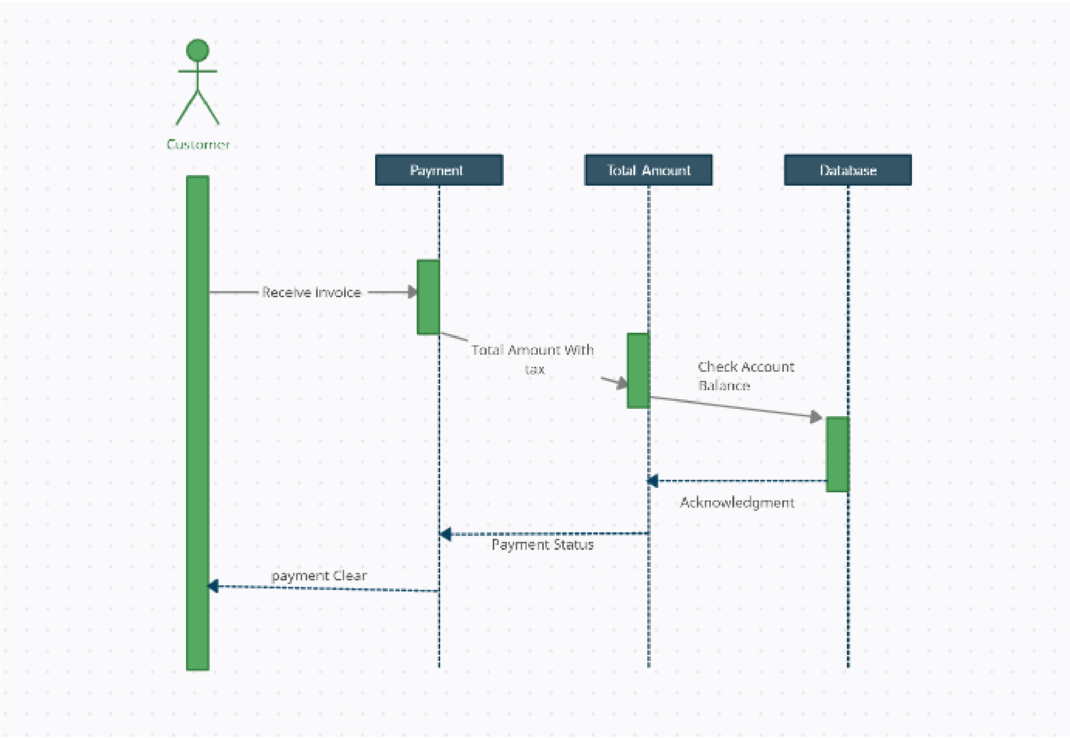


Figure 21: Customer Payment Sequence Diagram

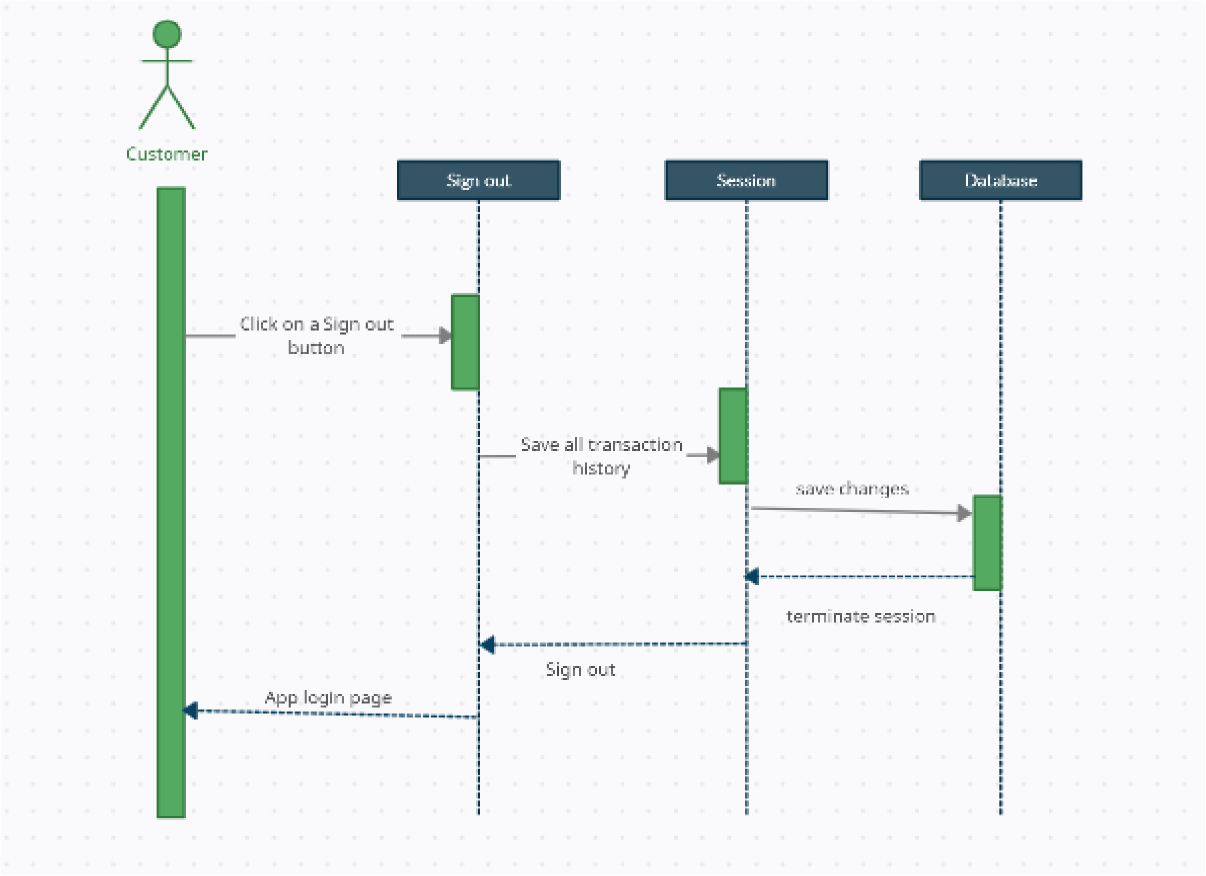


Figure 22: Customer Sign Out Sequence Diagram

3.3 Collaboration Diagram

A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. Objects are shown as rectangles with naming labels inside. These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles. The messages between objects are shown as arrows connecting the relevant rectangles along with labels that define the message sequencing.

Collaboration diagrams are best suited to the portrayal of simple interactions among relatively small numbers of objects. As the number of objects and messages grows, a collaboration diagram can become difficult to read. Several vendors offer software for creating and editing collaboration diagrams.

Fig. Below is the collaboration diagram of the Autonomous Restaurant Service Robot & Food Ordering App.

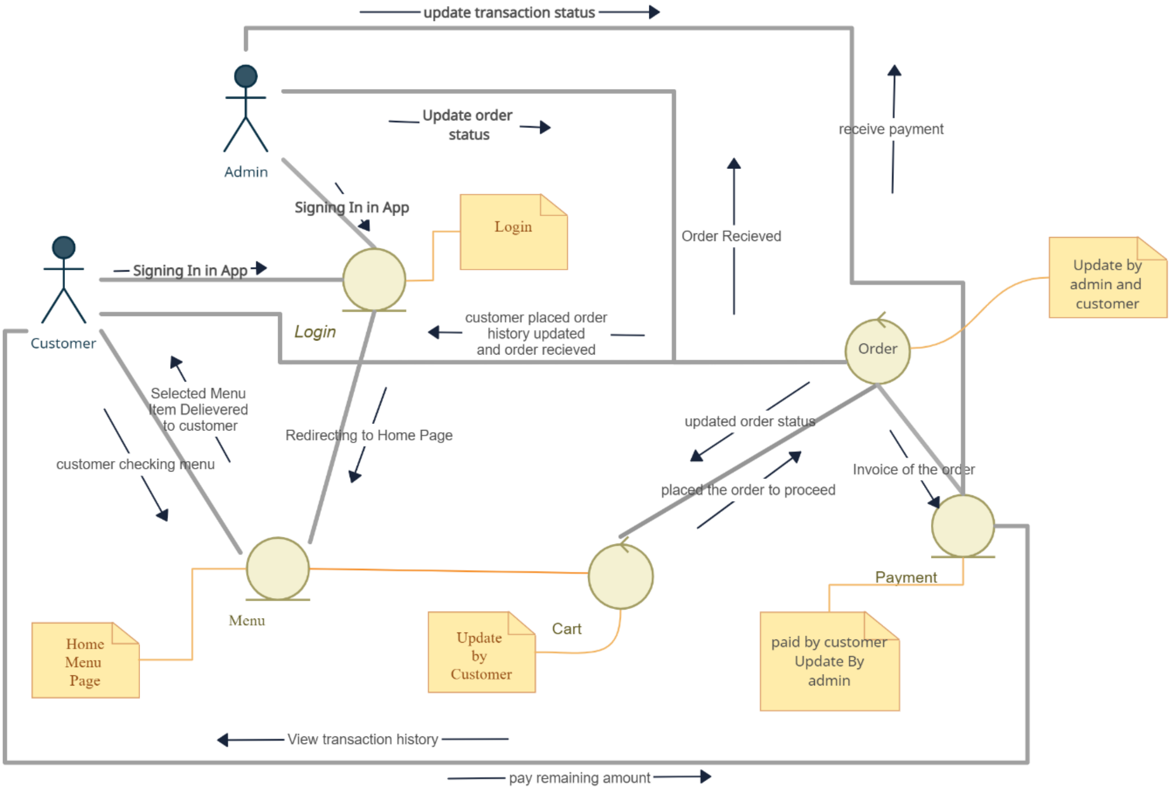


Figure 23: Collaboration Diagram

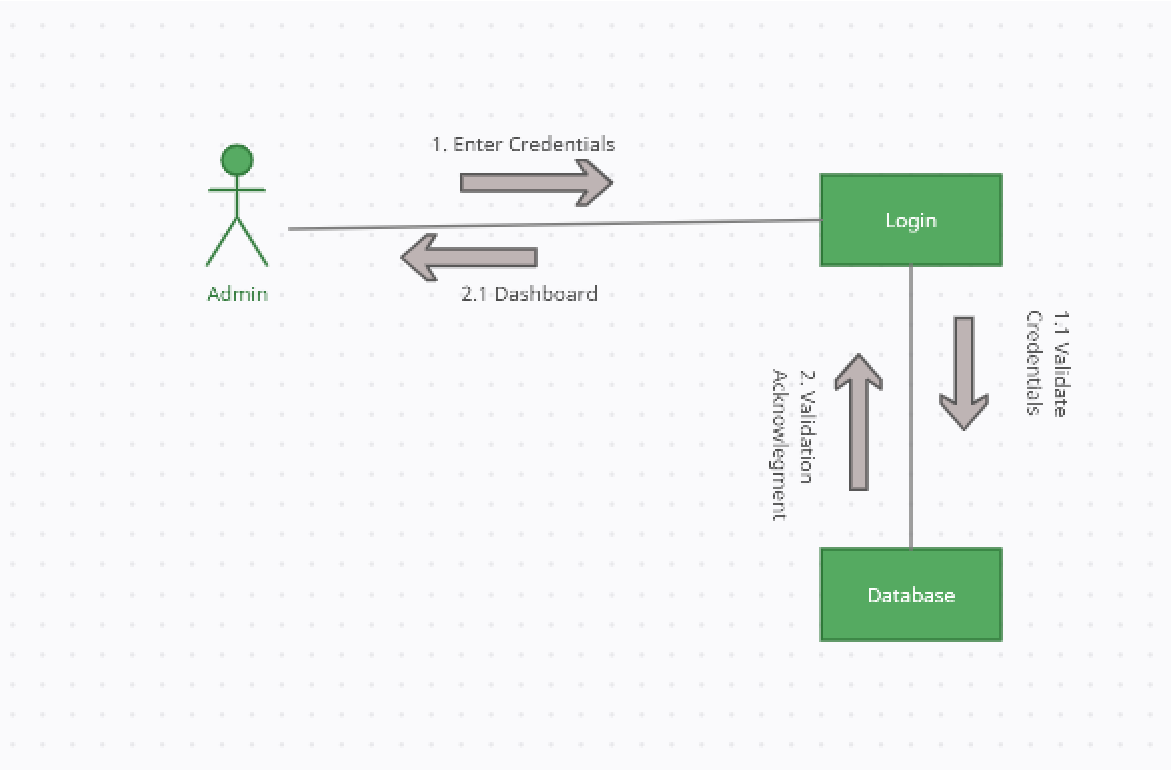


Figure 24: Admin Login

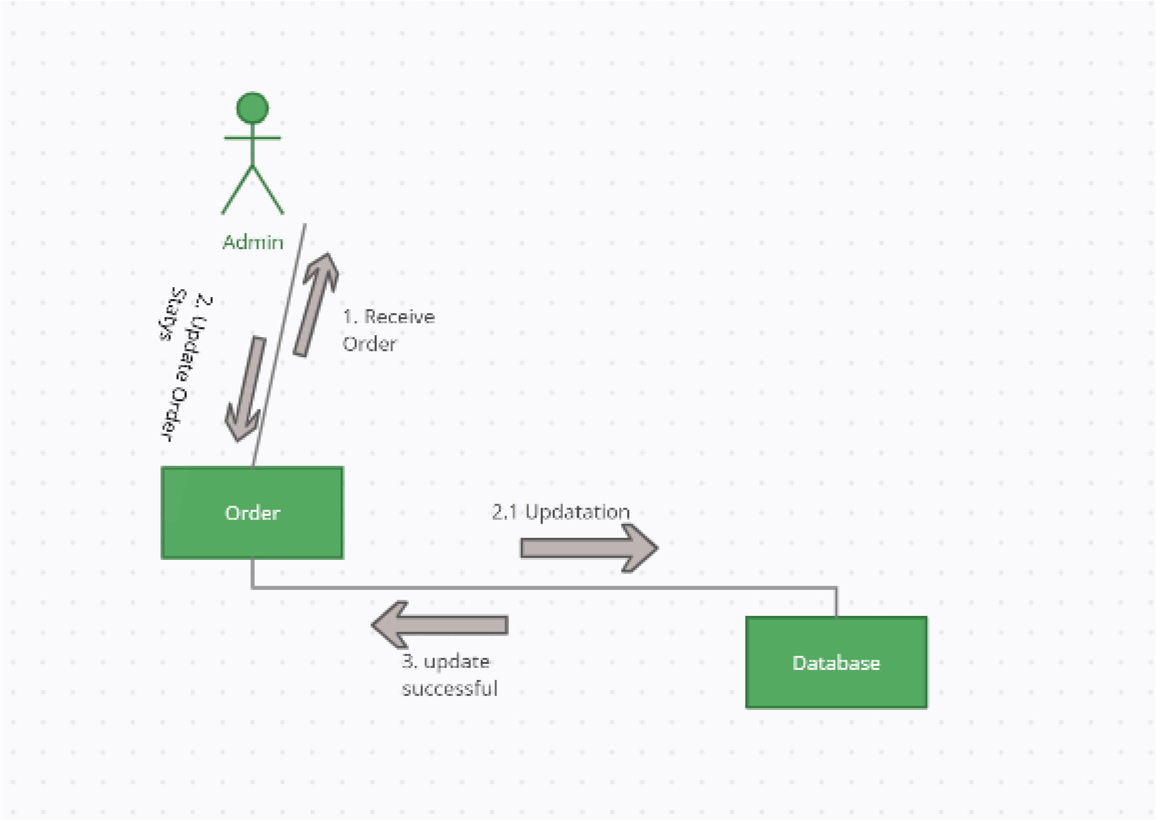


Figure 25: Admin Order

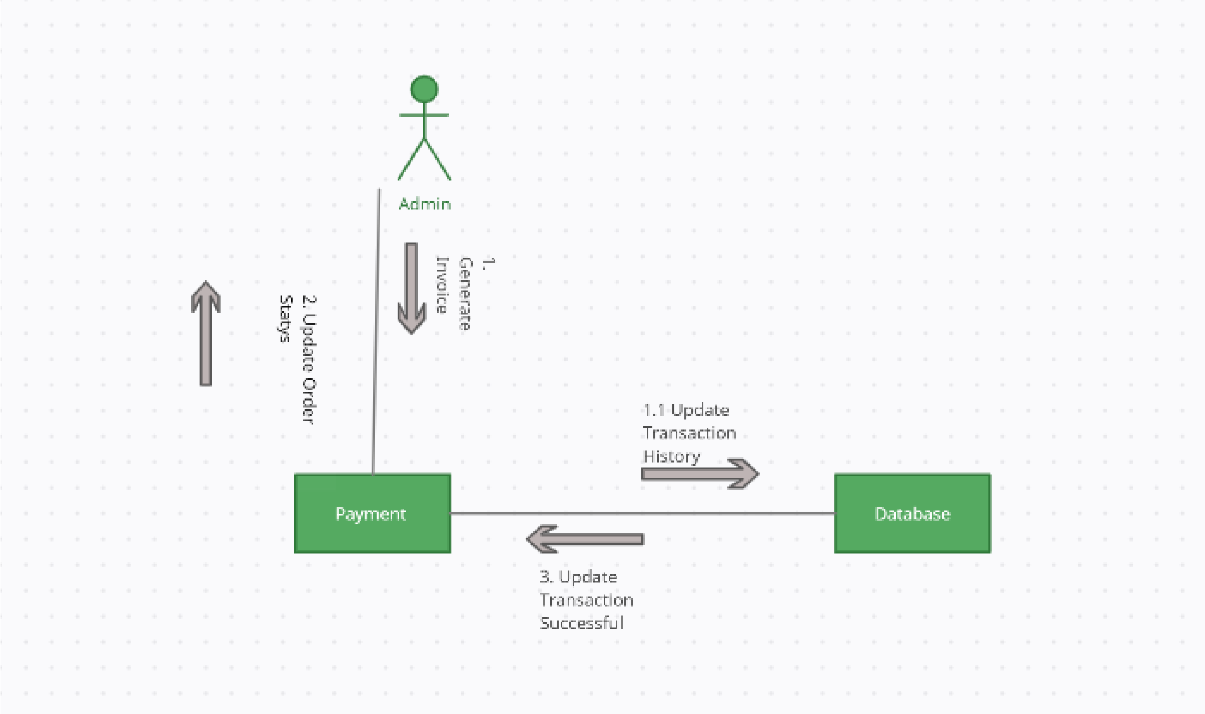

Admin
1. Generate Invoice
2. Update Order Statys
1.1 Update Transaction History
Payment
Database
3. Update Transaction Successful

Figure 26: Admin Payment

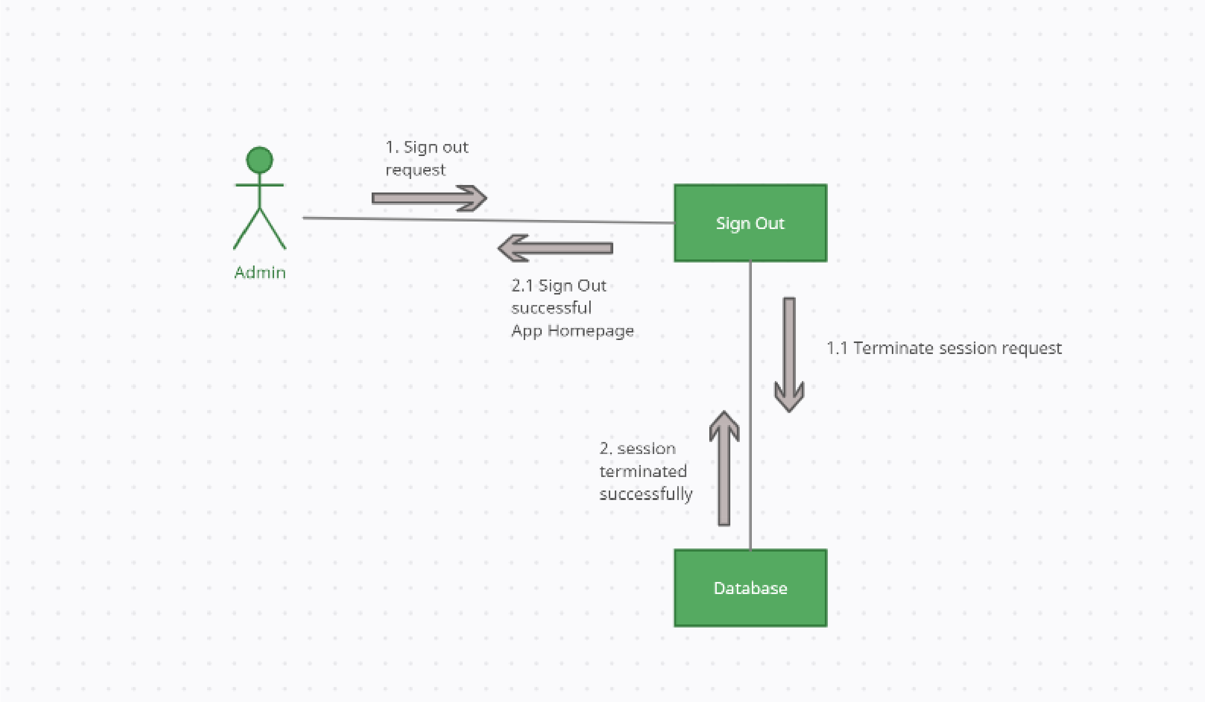

1. Sign out request
Admin
Sign Out
2.1 Sign Out successful App Homepage
1.1 Terminate session request
2. session terminated successfully
Database

Figure 27: Admin Sign Out

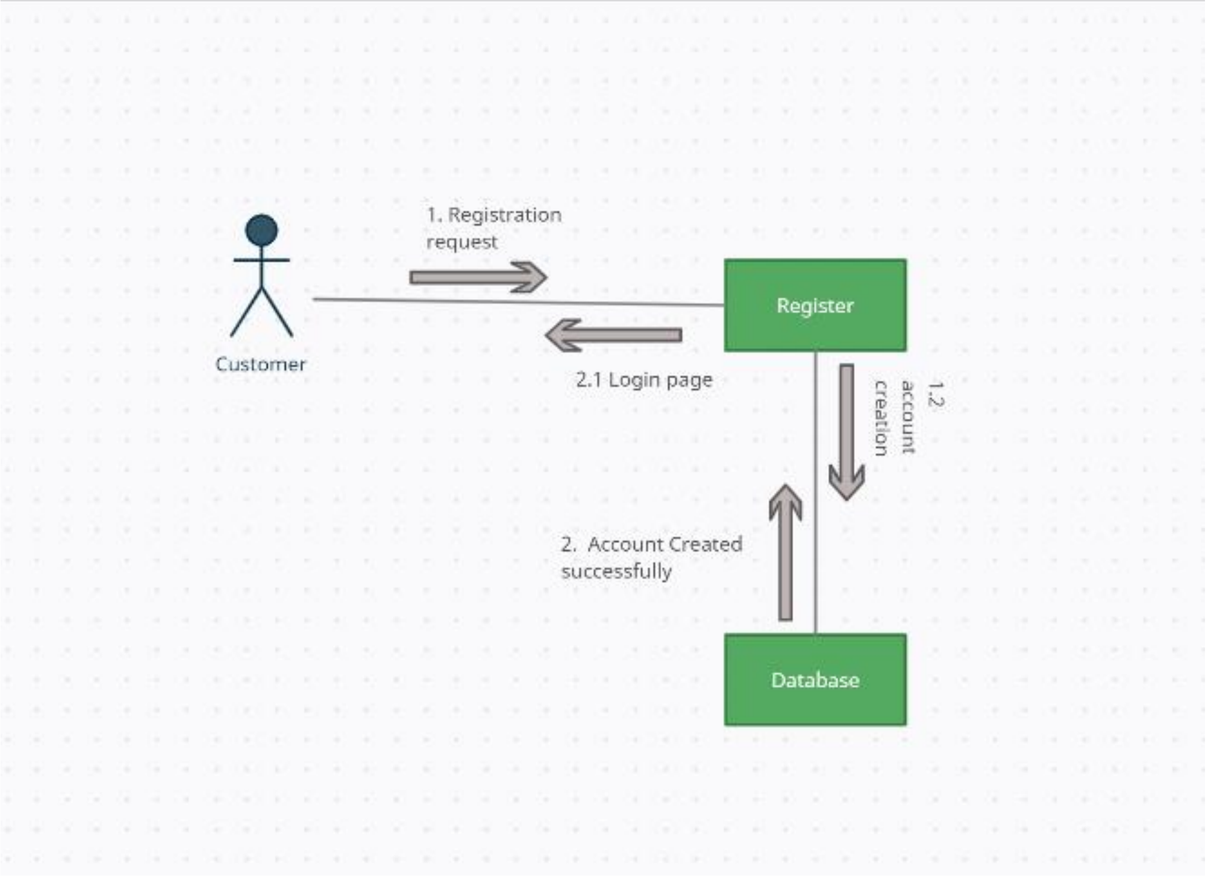


Figure 28: Customer Registration

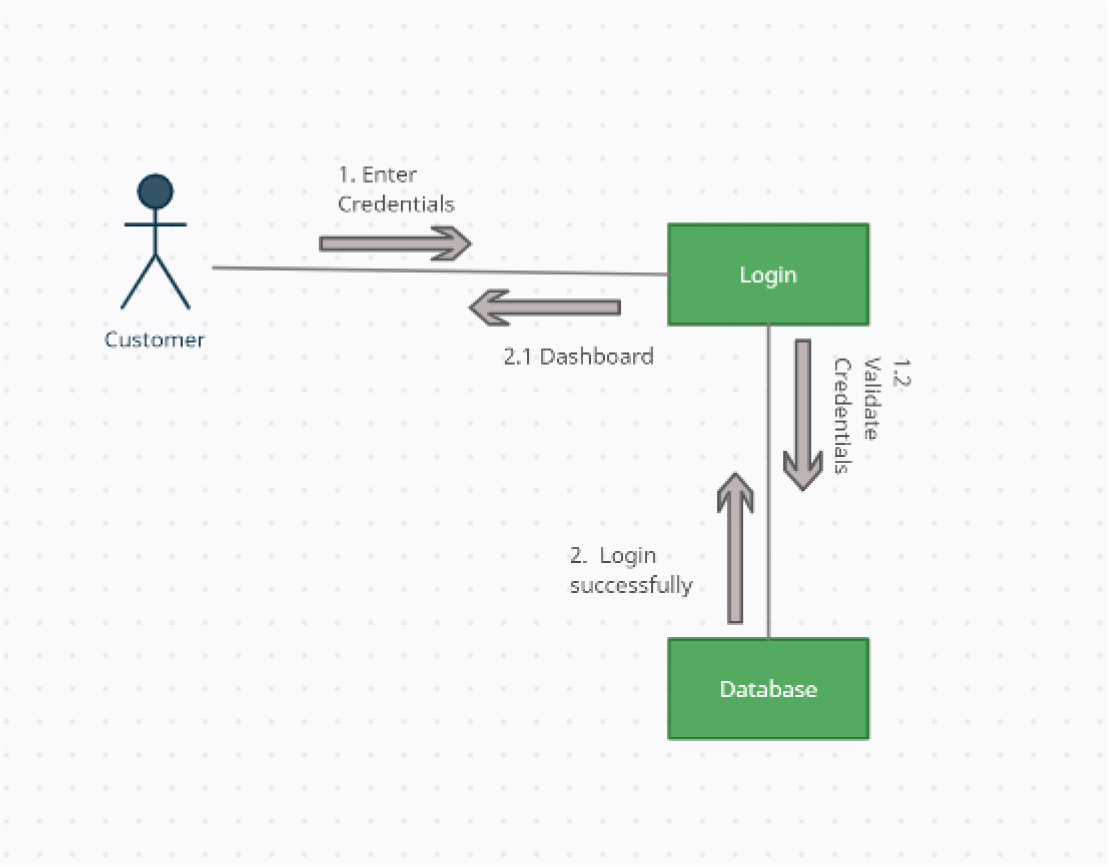


Figure 29: Customer Login

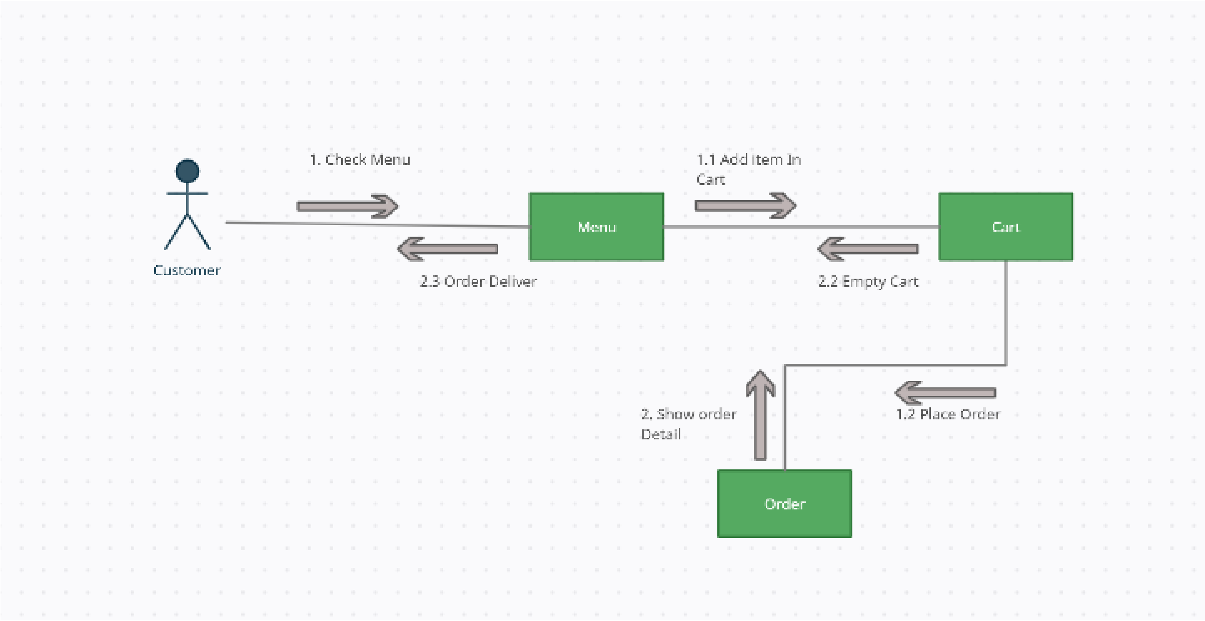


Figure 30: Customer Order

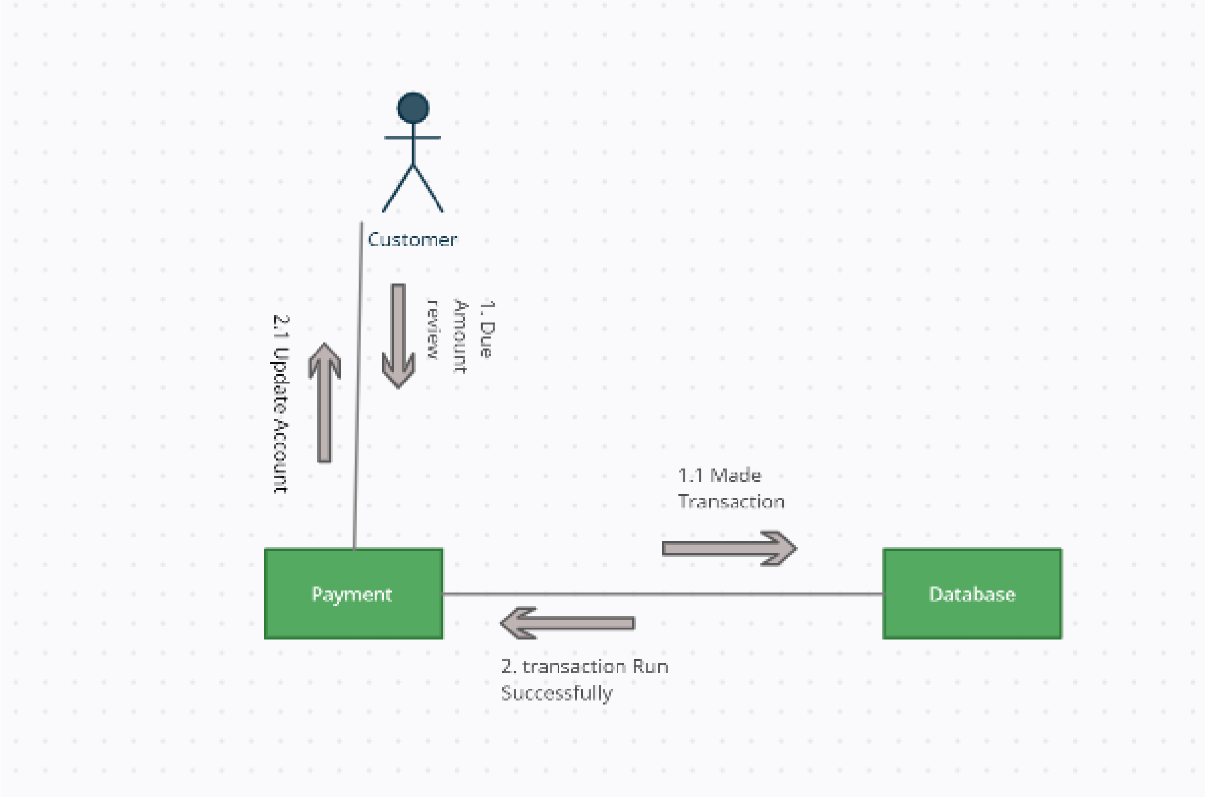


Figure 31: Customer Payment

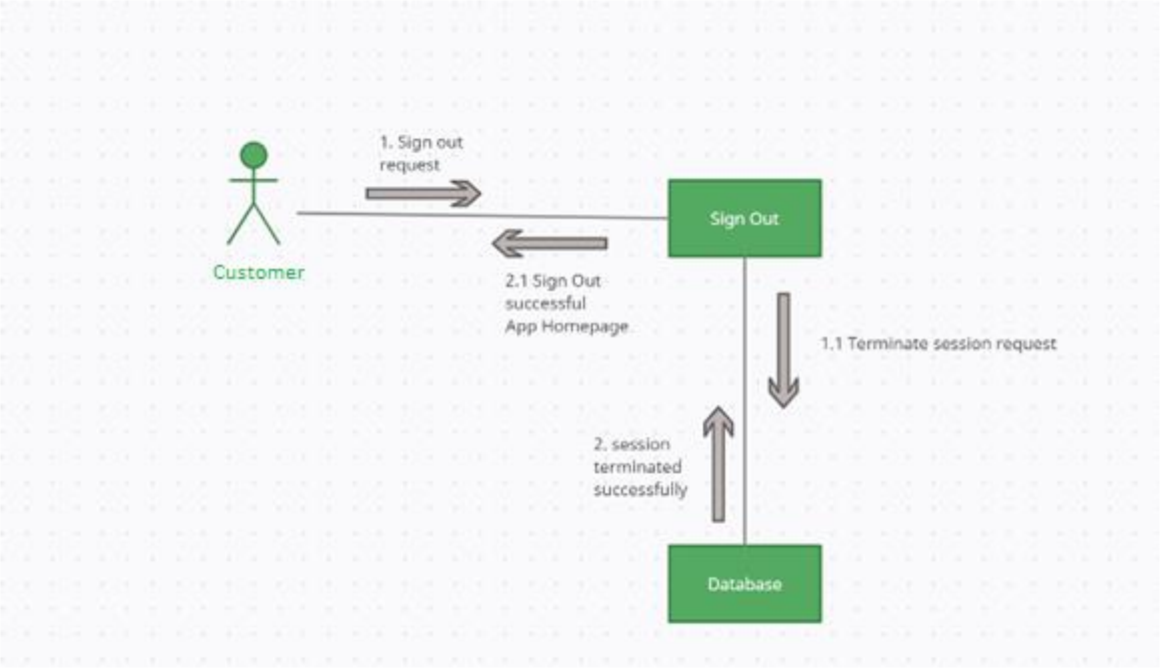

Customer
1. Sign out request
Sign Out
2.1 Sign Out successful App Homepage
1.1 Terminate session request
2. session terminated successfully
Database

Figure 32: Customer Sign Out

3.4 Operation Contracts

A UML Operation contract identifies system state changes when an operation happens. Effectively, it will define what each system operation does. An operation is taken from a system sequence diagram. It is a single event from that diagram. A domain model can be used to help generate an operation contract.

Operation Contract Admin login

Name: admin

Responsibilities: To register admin first himself

Cross References: U1

Exceptions: entered wrong credentials

Preconditions: Admin must open the app.

Post conditions: Admin is registered

Operation Contract Admin order

Name: order

Responsibilities: to admin the order update

Cross References: U2

Exceptions: entered wrong credentials

Preconditions: Admin must open the app.

Post conditions: Admin is registered

Operation Contract Admin payment

Name: payment

Responsibilities: to admin the update transaction

Cross References: U3

Exceptions: entered wrong credentials

Preconditions: Admin must open the app.

Post conditions: Admin is registered

Operation Contract Admin Log out

Name: Log out

Responsibilities: To allow the admin, log out from app

Cross References: U4

Exceptions: Admin entered wrong credentials

Preconditions: Admin must logged in.

Post conditions: Admin is log out

Operation Contract Customer Registration

Name: Registration

Responsibilities: To register user first himself

Cross References: U5

Exceptions: User entered wrong credentials

Preconditions: User must open the app.

Post conditions: User is registered.

Operation Contract Customer Login

Name: Login

Responsibilities: To allow user login to website

Cross References: U6

Exceptions: User entered wrong credentials

Preconditions: User must register himself first

Post conditions: User is logged in.

Operation Contract Customer order

Name: order

Responsibilities: to user the add items in cart

Cross References: U7

Exceptions: entered wrong credentials

Preconditions: User must open the app.

Post conditions: User is registered

Operation Contract Customer payment

Name: payment

Responsibilities: to user the made transaction

Cross References: U8

Exceptions: entered wrong credentials

Preconditions: User must open the app.

Post conditions: User is registered

Operation Contract Admin Log out

Name: Log out

Responsibilities: To allow the user, log out from app

Cross References: U9

Exceptions: User entered wrong credentials

Preconditions: User must logged in.

Post conditions: User is log out

3.5 Design Class Diagram

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.

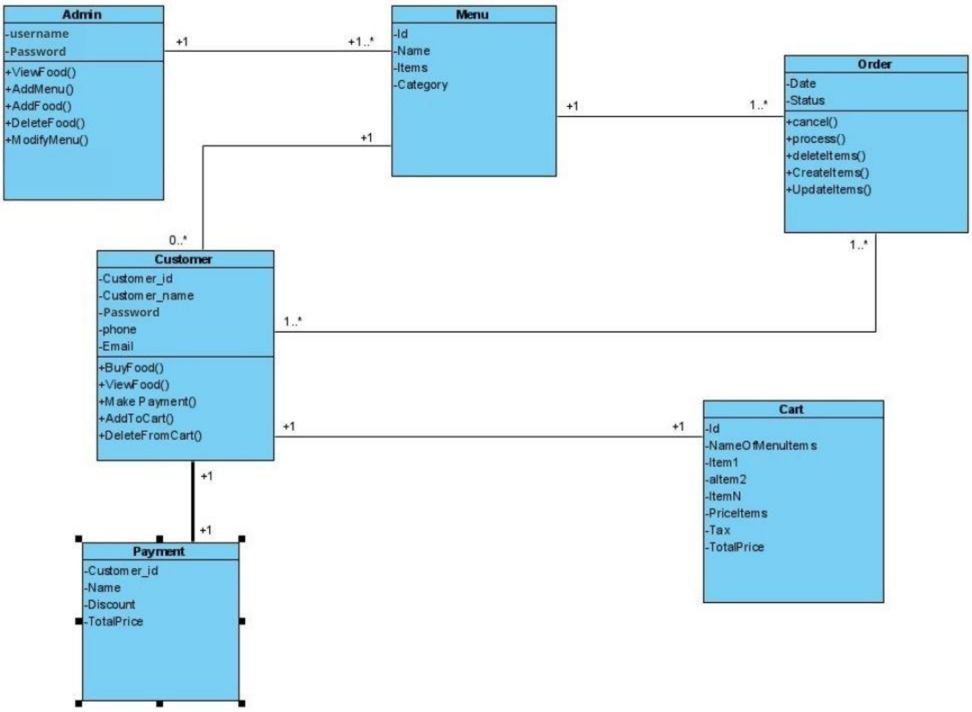


Figure 33: Class Diagram

Class diagram in figure 24 gives the description of the classes interacting within the Website. Classes are also showing the functions used in them including the variables used in the functions.

3.6 Data Model

3.6.1 Entity Relationship Diagram (ERD)

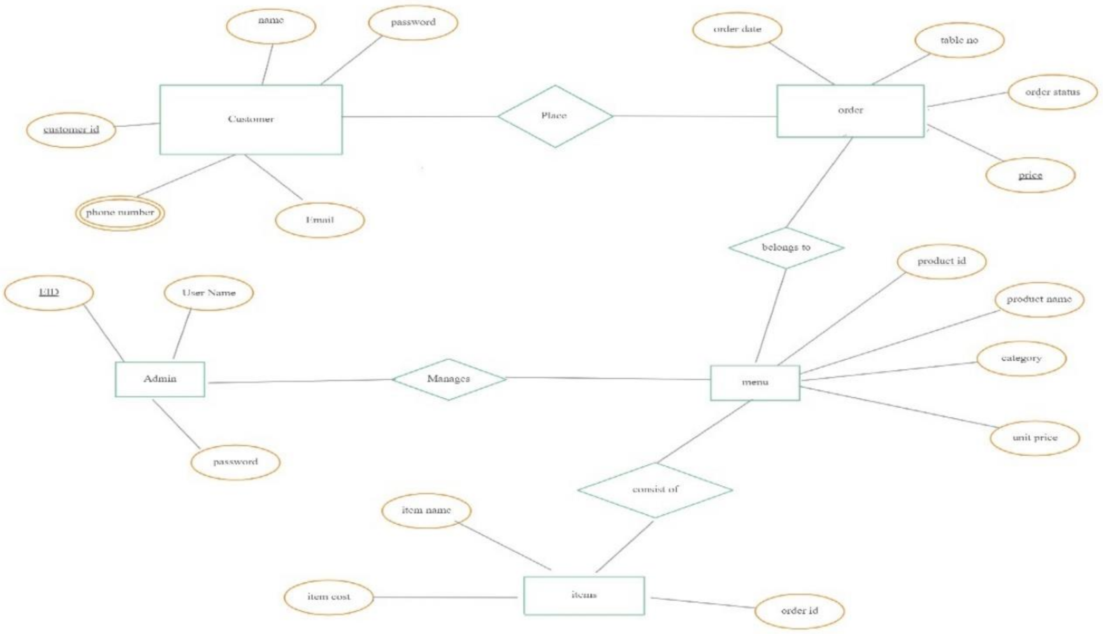


Figure 34: Data Model (ERD)

The ERD Entity Relationship Diagram in Fig 25: gives the description of the entities and their relationships with each other following are the entities in this project:

- Customer
- Admin
- Order
- Menu
- Items

Along with the entities their attributes and relationships are also listed which makes a complete ERD.

CHAPTER 4

DATA AND EXPERIMENTS (IMPLEMENTATION)

4.1 Tools & Technologies

Following are the tools and technologies used in Autonomous Restaurant Service Robot & Food Ordering App.

4.2 Hardware Implementation

In this project the main components which are used are as follow; Arduino Uno, Lithium-Ion 3 Cell Battery (11.1V, 2200 MAH), HC 05 Bluetooth Module, DC Gear Motors, Motor Driver Module (L298N), Infrared Sensors, Ultrasonic Sensors and Buzzer. All the components are integrating or interlink to get the actual output. Components are attached through Jumper Wires and configurations are made on Arduino Uno. Pulses coming from ultrasonic sensor and infrared sensor are passing to the motor driver module through Arduino. Arduino BlueControl App is configuring with Arduino which made the working of the robot proper.

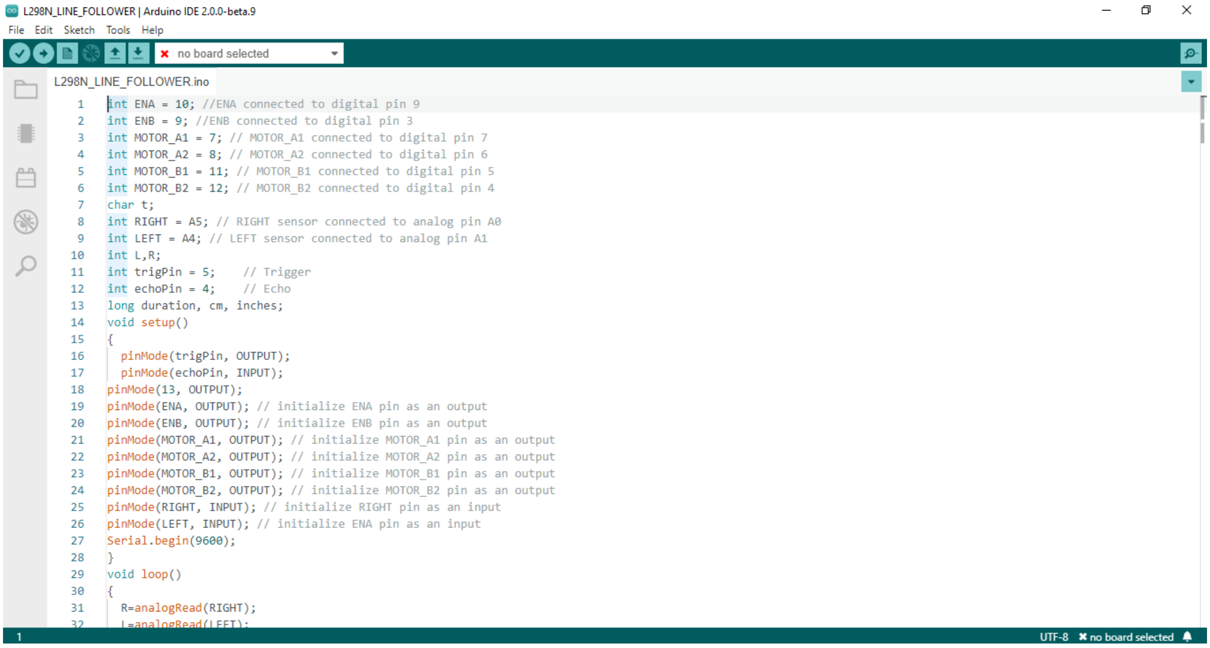
4.2.1 Arduino Uno

Specifications are:

- Microcontroller: Microchip ATmega328P
- Operating Voltage: 5 Volts
- Input Voltage: 7 to 20 Volts
- Digital I/O Pins: 14 (of which 6 can provide PWM output)
- UART: 1
- I2C: 1

- SPI: 1
- Analog Input Pins: 6
- DC Current per I/O Pin: 20 mA
- DC Current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB of which 0.5 KB used by bootloader
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz
- Length: 68.6 mm
- Width: 53.4 mm
- Weight: 25 g

Open-source hardware and software are Arduino, used for designing and for making digital and interactive objects, which will be sensing and controlling the objects. An IDE is used to write and load the code on physical board.



```
int ENA = 10; //ENA connected to digital pin 9
int ENB = 9; //ENB connected to digital pin 3
int MOTOR_A1 = 7; // MOTOR_A1 connected to digital pin 7
int MOTOR_A2 = 8; // MOTOR_A2 connected to digital pin 6
int MOTOR_B1 = 11; // MOTOR_B1 connected to digital pin 5
int MOTOR_B2 = 12; // MOTOR_B2 connected to digital pin 4
char t;
int RIGHT = A5; // RIGHT sensor connected to analog pin A0
int LEFT = A4; // LEFT sensor connected to analog pin A1
int L,R;
int trigPin = 5;    // Trigger
int echoPin = 4;    // Echo
long duration, cm, inches;
void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
pinMode(13, OUTPUT);
pinMode(ENA, OUTPUT); // initialize ENA pin as an output
pinMode(ENB, OUTPUT); // initialize ENB pin as an output
pinMode(MOTOR_A1, OUTPUT); // initialize MOTOR_A1 pin as an output
pinMode(MOTOR_A2, OUTPUT); // initialize MOTOR_A2 pin as an output
pinMode(MOTOR_B1, OUTPUT); // initialize MOTOR_B1 pin as an output
pinMode(MOTOR_B2, OUTPUT); // initialize MOTOR_B2 pin as an output
pinMode(RIGHT, INPUT); // initialize RIGHT pin as an input
pinMode(LEFT, INPUT); // initialize ENA pin as an input
Serial.begin(9600);
}
void loop()
{
  R=analogRead(RIGHT);
  L=analogRead(LEFT);
```

Figure 35: Figure 4. 1 Arduino IDE representing the code

4.2.2 Ultrasonic Sensor (hr-Sr04)

Specifications are:

- Provides 2-400 cm non-contact measurement function.

- Ranging accuracy can reach to 3mm and effectual angle is < 15°.
- Powered from 5V power supply.

It’s used for detecting the distance to an object using sonar. Generate pulses, which move to motor driver module to affect the movement of the suit case.

4.2.3 Motor Driver Module (L298N)

Specifications are:

- Double H Bridge Drive Chip: L298N
- Logical Voltage: 5V
- Drive Voltage: 5V-35V
- Logical Current: 0-36 mA
- Drive current: 2A (MAX single bridge)
- Max Power: 25 W
- Dimensions: 43 x 43 x 26 mm

The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time on the instructions coming from Arduino.

4.2.4 Bluetooth Module (Hc-05)

Specifications are:

- Bluetooth protocol: Bluetooth Specification v2.
- Frequency: 2.4GHz ISM band.
- Modulation: GFSK (Gaussian Frequency Shift Keying).
- Emission power: ≤4dBm, Class 2.
- Sensitivity: ≤-84dBm at 0.1% BER.
- Speed: Asynchronous: 2.1Mbps(Max) / 160 kbps, Synchronous: 1Mbps/1Mbps.
- Security: Authentication and encryption.

Its use to provide the theft alters when the suitcase will be far away from the passenger more than 10 meters.

4.2.5 DC Gear Motor (9V)

Specifications are:

- Voltage: 9 V.
- Speed at no load: 190 RPM.
- Current at no load: 0.05 A.
- Speed max. efficiency: 147 RPM.
- Current max. efficiency: 0.125 A.
- Torque max. efficiency: 0.0122 NM.
- Current at stall: 0.35 A.
- Torque at stall: 0.0784 NM.

DC motors have higher starting torque, quick starting and stopping, reversing, variable speeds with voltage input.

4.2.6 Lithium-Ion Cell Battery

Specifications are:

- Voltage: 11.1 V
- Battery Capacity: 2.2 MAH
- End Connector: Barrel Jack Connector
- Manufacturer: Amptex

These are high performance, high current discharge batteries best for all-terrain robots, combat robots, autonomous robots and other electronic projects.

4.2.7 Integration of Components

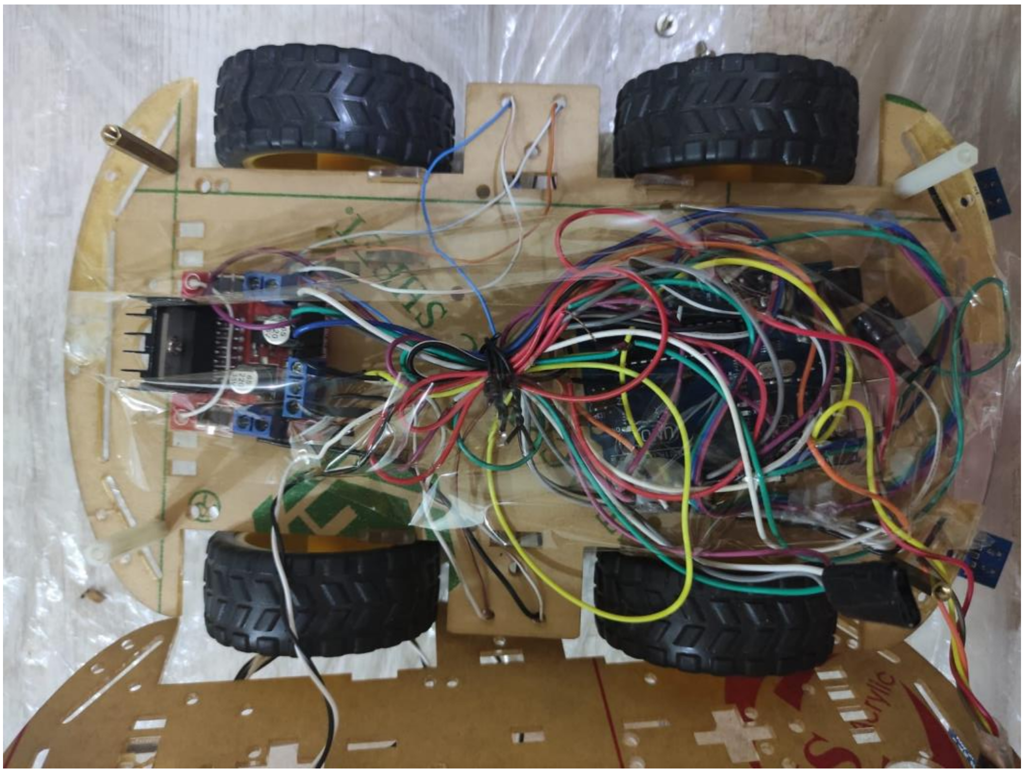


Figure 36: Integrated Components

4.3 Software Implementation

These are the main technologies used in the Application Development.

4.3.1 Java

Java is a general-purpose, class-based, object-oriented programming language designed for having lesser implementation dependencies. It is a computing platform for application development. Java is fast, secure, and reliable, therefore. It is widely used for developing Java applications in laptops, data centers, game consoles, scientific supercomputers, cell phones.

4.3.2 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. [8] It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

4.3.3 PHP

PHP is a general-purpose scripting language geared towards web development. PHP is a server-side scripting language. that is used to develop Static websites or Dynamic websites or Web applications. PHP stands for Hypertext Pre-processor, that earlier stood for Personal Home Pages. PHP scripts can only be interpreted on a server that has PHP installed. [9] The client computers accessing the PHP scripts require a web browser only. A PHP file contains PHP tags and ends with the extension ".php"

4.3.4 Laravel

Laravel is an open-source web MVC framework for PHP. Laravel is a robust framework that provides easy development of PHP web applications with features like a modular packaging system with a dedicated dependency manager, access to relational databases, and other utilities for application deployment and maintenance. [10]

4.3.5 C Language

C programming is a general-purpose, procedural, imperative computer programming language. C is the most widely used computer language. [11] It keeps fluctuating at number one scale of popularity along with Java programming language, which is also equally popular and most widely used among modern software programmers.

CHAPTER 5

RESULTS AND DISCUSSIONS (USER MANUAL)

5.1 General Information

Autonomous Restaurant Serving Robot is an Arduino Based Robot which serves the food in the restaurant and the android-based Food Ordering Mobile Application is used to take orders from customers and it will show on admin panel which is web based panel where the admin can deal with the orders to perform desired operations.

5.2 Overview

The Restaurant app will take the orders from the customers and in the admin panel the administrator will handle the orders and all the queries. And when the food gets ready, he will place the food on the robot which will complete the part of serving to the customer.

5.3 Result

Results of the system with UI and Robot are following:

5.3.1 Robot Body

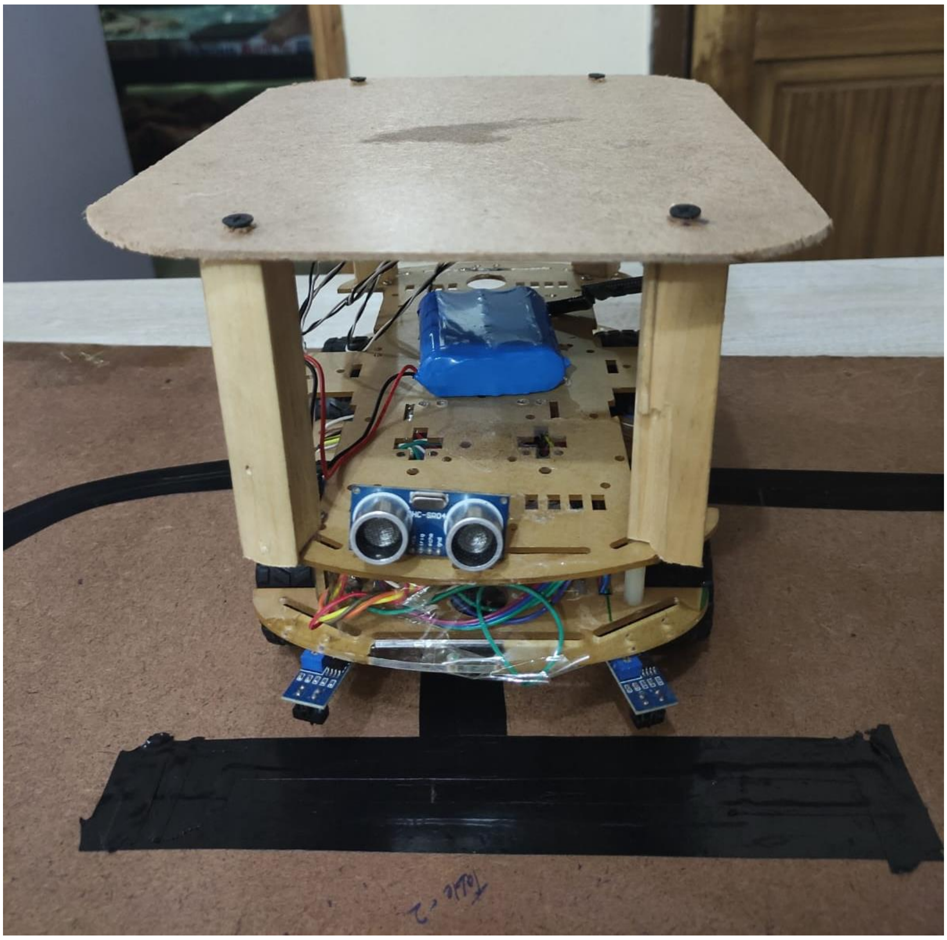


Figure 37: Robot Body

5.3.2 Robot Path

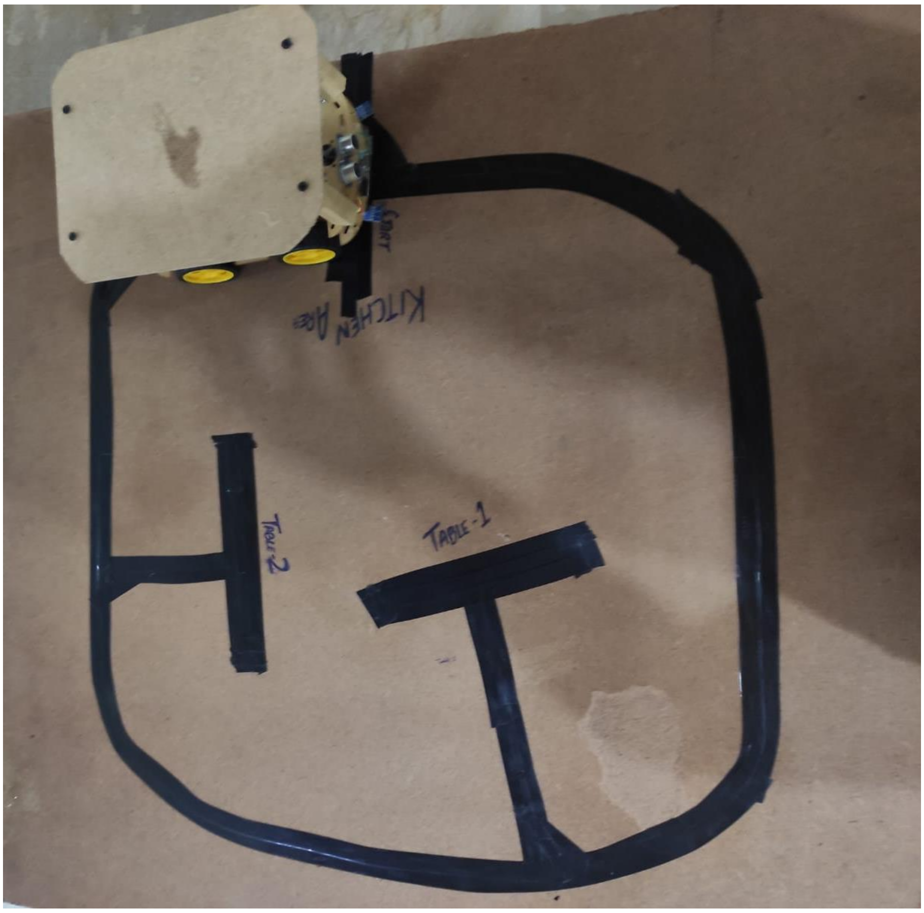


Figure 38: Robot Path

5.3.3 LOGIN PAGE

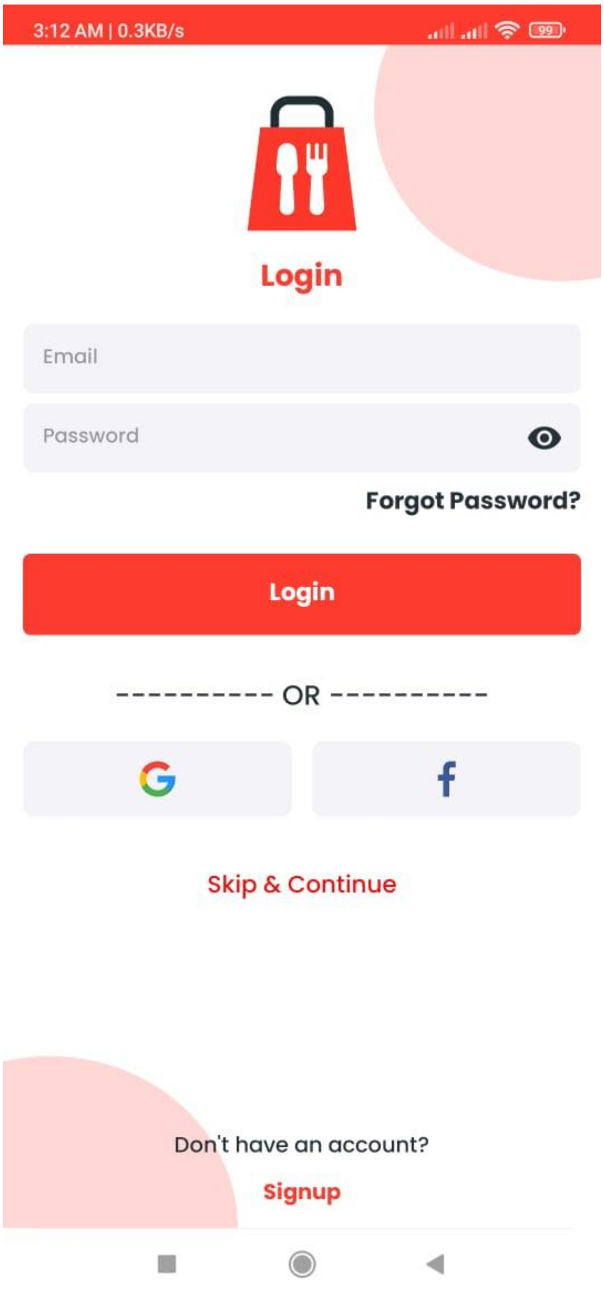


Figure 39: Login Page

5.3.4 Home Page UI

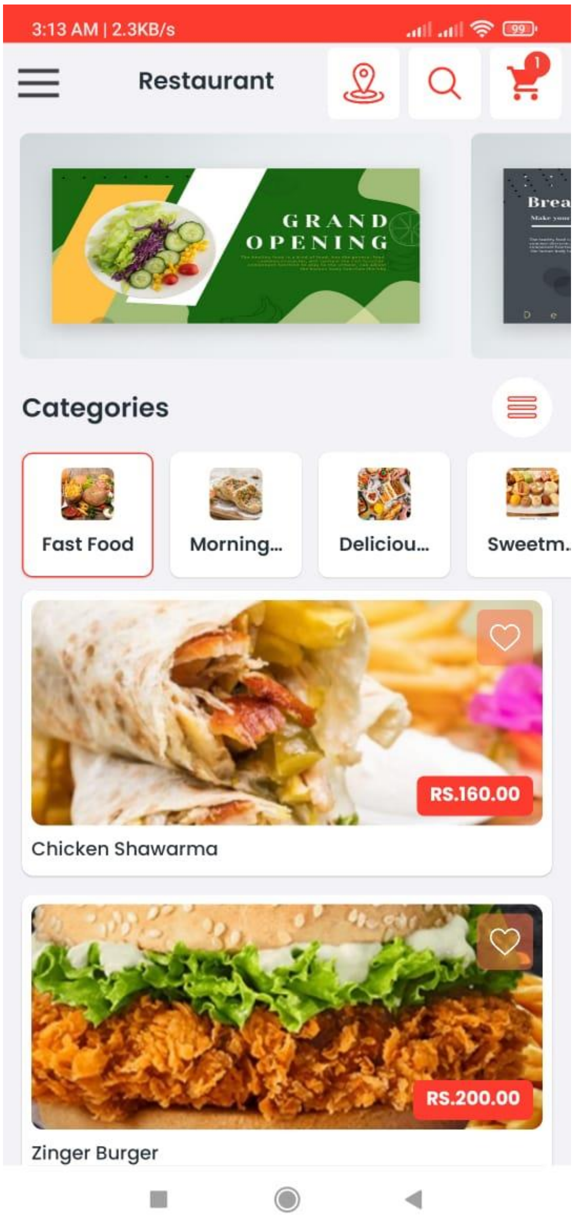
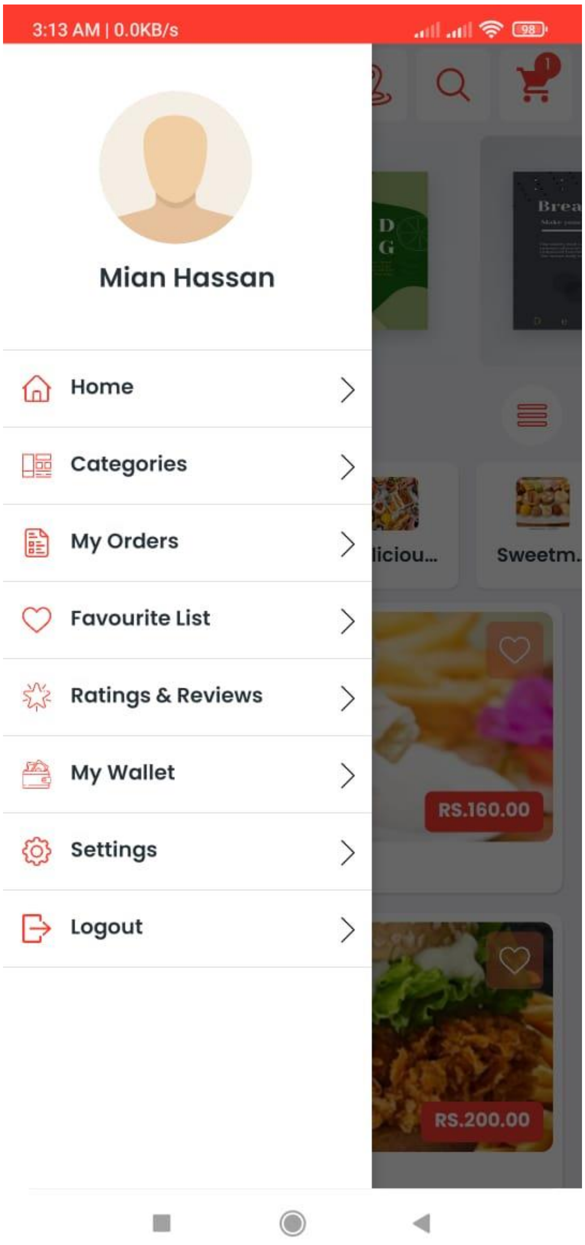


Figure 40: Home Page UI

5.3.5 User Menu



3:13 AM | 0.0KB/s

Mian Hassan

Figure 41: User Menu

5.3.6 Categories

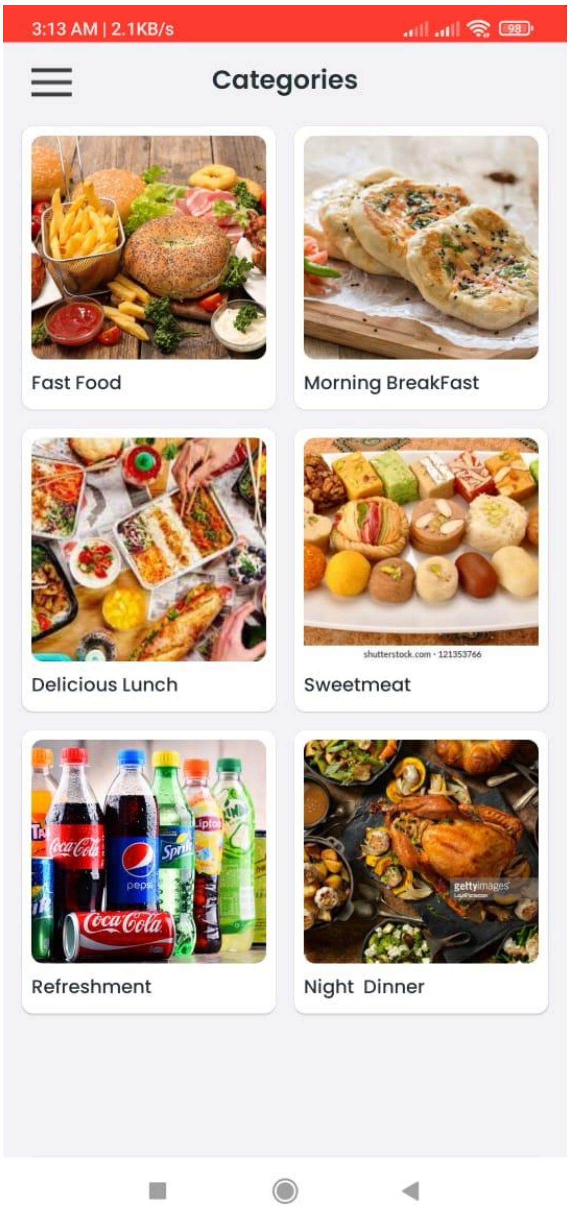
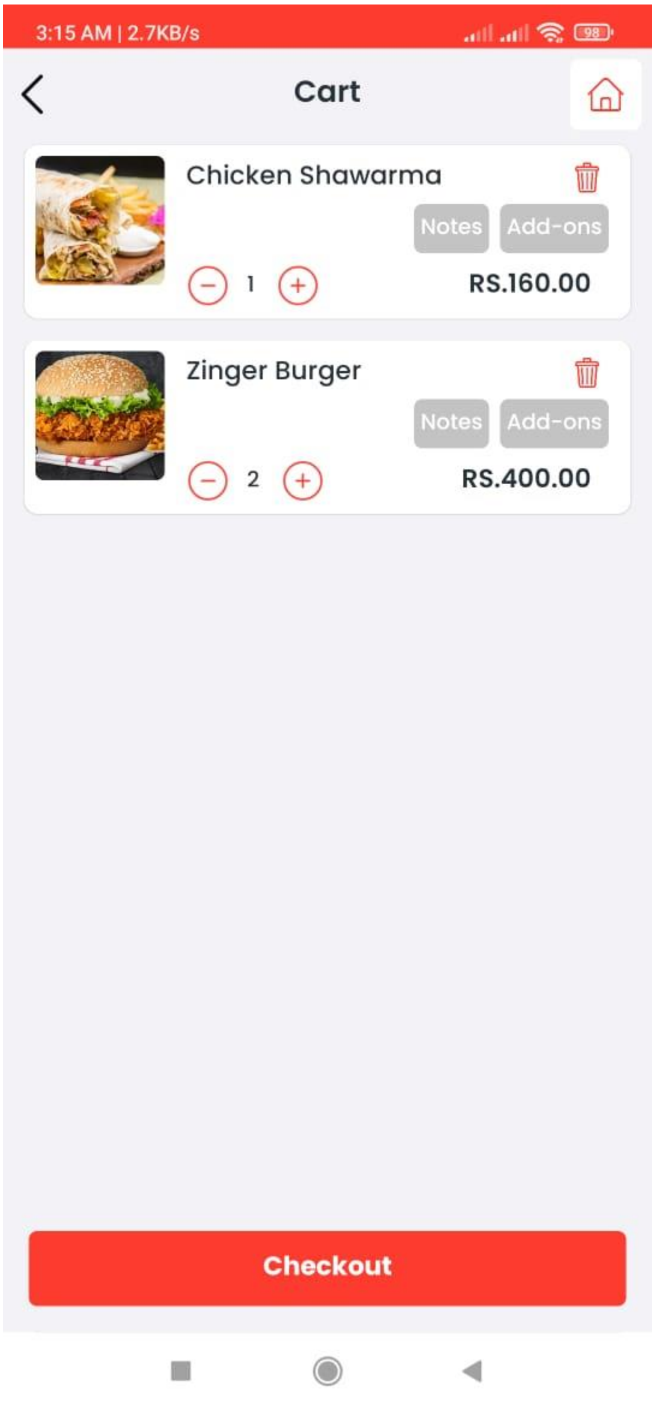


Figure 42: Categories

5.3.7 Cart



3:15 AM | 2.7KB/s

Cart

Chicken Shawarma

1

RS.160.00

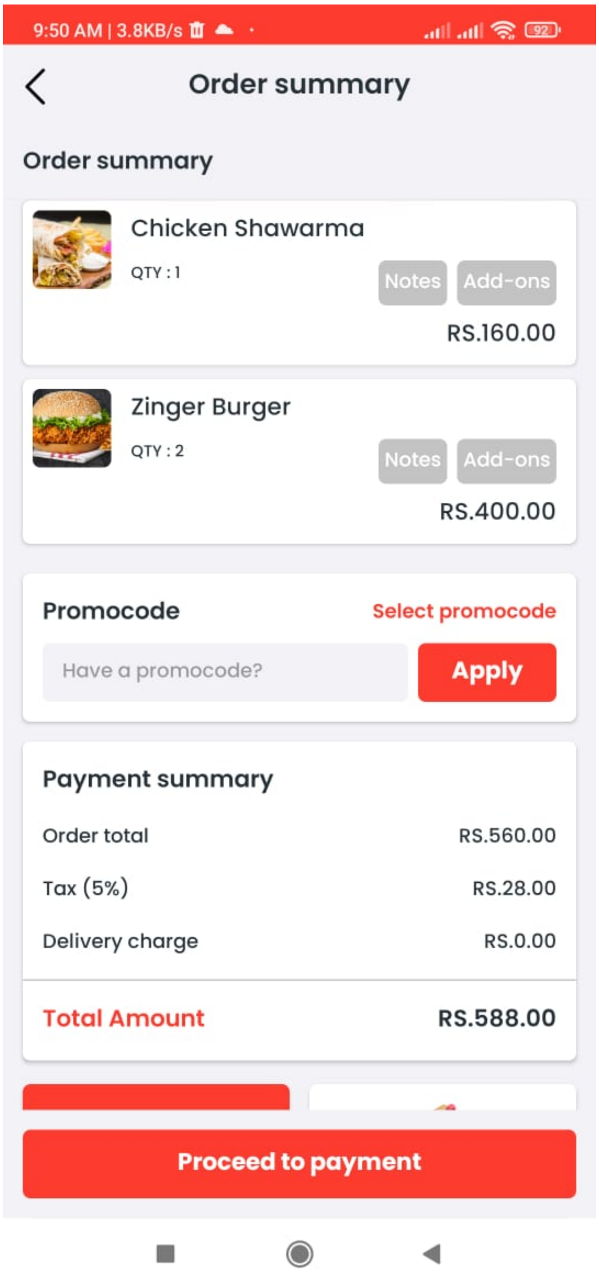
Zinger Burger

2

RS.400.00

Figure 43: Cart

5.3.8 Order Summary



Order summary

Chicken Shawarma
QTY : 1
RS.160.00

Zinger Burger
QTY : 2
RS.400.00

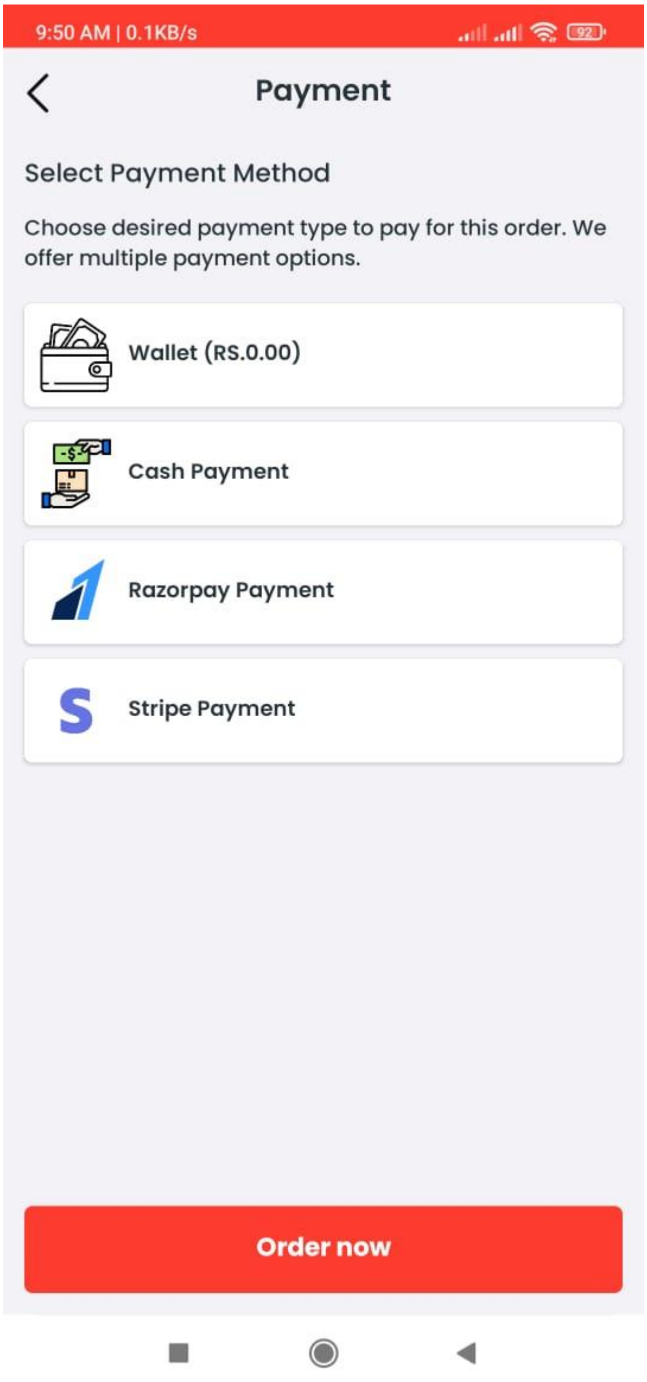
Promocode

Payment summary

Order total	RS.560.00
Tax (5%)	RS.28.00
Delivery charge	RS.0.00
Total Amount	RS.588.00

Figure 44: Order Summary

5.3.9 Payment Method



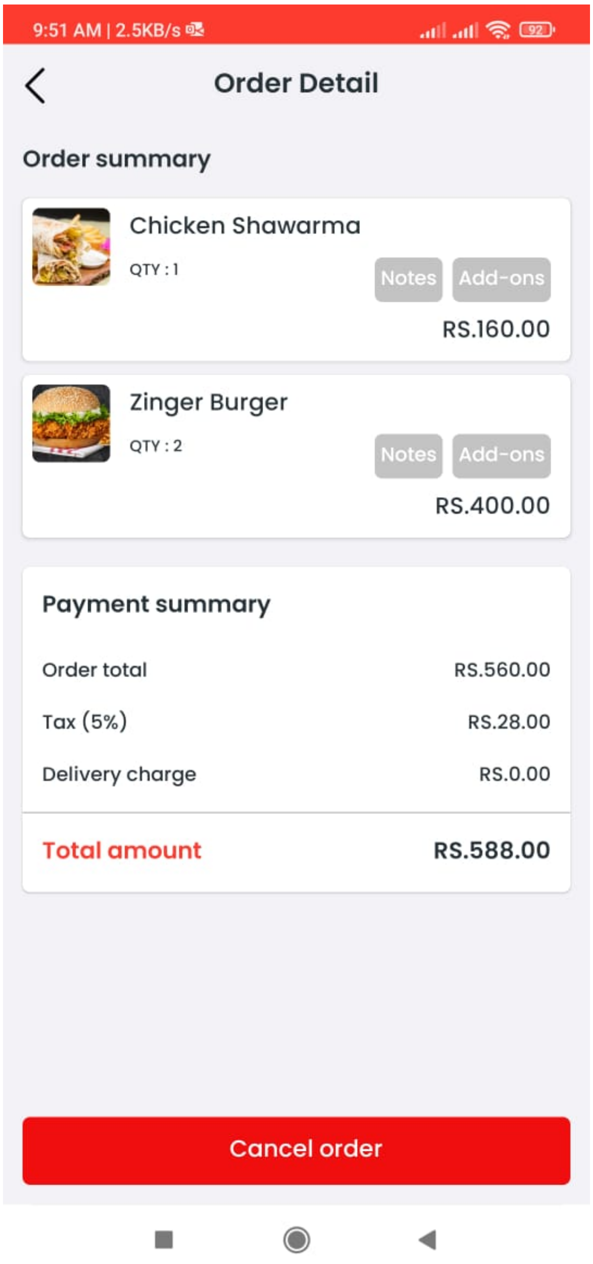
Payment

Select Payment Method

Choose desired payment type to pay for this order. We offer multiple payment options.

Figure 45: Payment Method

5.3.10 Order Details



9:51 AM | 2.5KB/s

Order Detail

Order summary

Chicken Shawarma

QTY : 1

RS.160.00

Zinger Burger

QTY : 2

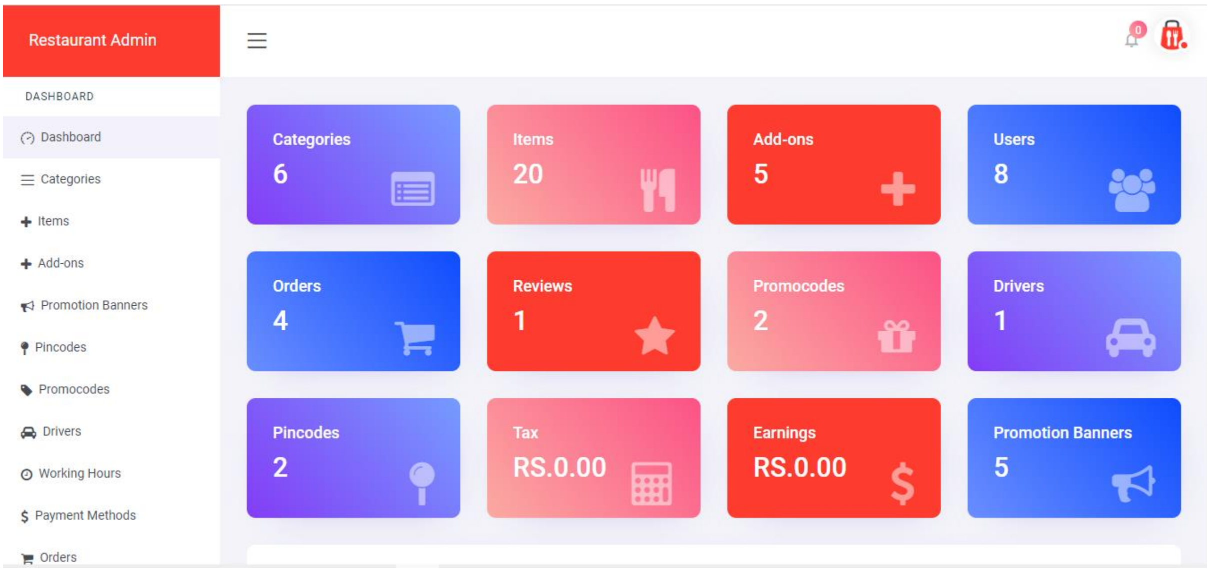
RS.400.00

Payment summary

Order total	RS.560.00
Tax (5%)	RS.28.00
Delivery charge	RS.0.00
Total amount	RS.588.00

Figure 46: Order Details

5.3.11 Admin Panel UI



Categories 6

Items 20

Add-ons 5

Users 8

Orders 4

Reviews 1

Promocodes 2

Drivers 1

Pincodes 2

Tax RS.0.00

Earnings RS.0.00

Promotion Banners 5

Figure 47: Admin Panel UI

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

The Autonomous Restaurant Serving Robot which comprises IOT solution which resolve the problem of restaurant owners. This robot neglects the concept of waiters in the restaurants and along with that our food ordering app provides all the facilities to the customers. In the app, when the customer Log-in into there account they see all the products and from those products they can order by adding products into their cart. This Autonomous Restaurant Serving Robot and Mobile Application makes the contact less service possible. It minimizes the risks of human interaction for the betterment of hygiene and health.

The customers will order their food through Mobile Application and The Admin in the kitchen will note down the order and cook it for them and once the food is prepared it will be served to the customer through the robot. This robot is a onetime investment and that will save many rupees of restaurant owner from giving monthly salaries to the waiters.

6.2 Future Recommendations

In The Autonomous Restaurant Serving Robot, there is a large way of advancement hence improving its functionality and efficiency using better sensors and better devices like “Raspberry Pi” to make it more automated using Machine Learning. To not provide any path to the robot and it will find its own path through Artificial Intelligence. If it detects any object in Infront of it, it changes its path and finds the shortest path to the required table. In addition, to make the robot with less power and resource-consuming hardware, light and portable batteries, and the mobile Application containing the aesthetic controls and other essential features flawlessly.

Also To add a driver in the food ordering application so the customer can enjoy their meal from their home and have a better experience of the meal.

REFERENCES

[1] Arduino Store, "Arduino Store," Arduino, 2016. [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3.

[2] Elprocus, "What is an IR Sensor : Circuit Diagram & Its Working," 2020.

[3] Questex LLC, "Ultrasonic Sensor," 2021.

[4] LastMinuteEngineers, "Interface L298N DC Motor Driver Module with Arduino".

[5] N. Hemati and M. C. Leu, "A complete model characterization of brushless DC motors," 1992.

[6] Yoshio, Masaki, Ralph J. Brodd, and Akiya Kozawa, "Lithium-ion batteries.," 2009.

[7] "Interfacing of AT Command based HC-05 Serial Bluetooth Module with," 2014.

[8] Craig, Clifton, and Adam Gerber, "Learn Android Studio," 2015.

[9] R. e. a. Lerdorf, Programming Php., 2002.

[10] M. Stauffer, Laravel: Up & running: A framework for building modern php apps., O'Reilly Media,, 2019.

[11] P. M. B. K. a. J. F. B. Embree, "C language algorithms for digital signal processing.," 1991.