



BSCS-F21-018

**03-134182-087** Muhammad Umer Sheikh

**03-134182-032** Muhammad Ali

# **Milkoride**

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

Supervisor: Numan Aslam

Department of Computer Sciences  
Bahria University, Lahore Campus

June 2022



# Certificate



We accept the work contained in the report titled.

“Milkoride”.

written by

Muhammad Umer Sheikh

Muhammad Ali

as a confirmation to the required standard for the partial fulfilment of the degree of  
Bachelor of Science in Computer Science.

Approved by:

Supervisor: Numan Aslam

\_\_\_\_\_  
(Signature)

June 14, 2022

## 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
03-134182-087	Muhammad Umer Sheikh	
03-134182-032	Muhammad Ali	

Date: June 14, 2022

Specially dedicated to  
my beloved grandmother, mother and father  
(Muhammad Umer Sheikh)  
my beloved grandmother, mother and father  
(Muhammad Ali)

## **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 research supervisor, Mr Numan Aslam for his invaluable advice, guidance and his enormous patience throughout the development of the research.

Besides, we would also like to express our gratitude to our loving parent and friends who had helped and encouraged us.

Muhammad Umer Sheikh

Muhammad Ali

## **Milkoride**

### **ABSTRACT**

E-commerce application usage is increasing in our society on daily basis. Everyone likes and wants to do their shopping through e-commerce platforms. In specific individuals are buying their daily routine item through e-commerce. These items include food, clothing, and other items. Customers can purchase required items through e-commerce without any hassle. While online shopping and offline shopping both have their merits and demerits, comparing these two modes of shopping options we choose the best one from them based on our need and comfort level. Customers also give feedback about that product from which they purchased from the market or e-commerce site after using those things. Specifically in Pakistan, unpacked milk has a very huge market than organic packed milk. In Pakistan, e-commerce apps don't deal with the unpacked milk market. Our application fills this gap and links customer and unpacked milk providers with an e-commerce platform.

## TABLE OF CONTENTS

<b>DECLARATION</b>	<b>ii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>iv</b>
<b>ABSTRACT</b>	<b>v</b>
<b>TABLE OF CONTENTS</b>	<b>vi</b>
<b>LIST OF TABLES</b>	<b>viii</b>
<b>LIST OF FIGURES</b>	<b>x</b>

### CHAPTER 1

#### 1 INTRODUCTION

	Background	1
1.1	Problem Statements	2
1.2	Aims and Objectives	2
1.3	Scope of Project	2

### CHAPTER 2

#### 2 SOFTWARE REQUIREMENT SPECIFICATION (SRS)

	2.1 User Classes and Characteristics	4
2.2	Operating Environment	5
2.3	Design and Implementation Constraints	5
2.4	Assumptions and Dependencies	6
2.5	External Interface Requirements	6
2.6	Other Non-functional Requirements	7

### CHAPTER 3

#### 3 DESIGN AND METHODOLOGY

	3.1 System Development Methodology	10
3.2	Use Case Diagram	12
3.3	Use Case Description	14



3.4	Sequence Diagrams	28
3.5	Design Class Diagram	36
3.6	ERD Diagram	39
<b>CHAPTER 4</b>		
<b>IMPLEMENTATION</b>		
4.1	Technologies Used	40
<b>CHAPTER 5</b>		<b>43</b>
<b>USER MANUAL</b>		
<b>CHAPTER 6</b>		<b>53</b>
<b>CONCLUSION RECOMMENDATION</b>		
6.1	Conclusion	53
6.2	Recommendation	53
<b>REFERENCES</b>		<b>54</b>

**LIST OF TABLES**

<b>TABLE</b>	<b>TITLE</b>	<b>PAGE</b>
	<b>Table 2-1: Software required interfaces</b>	<b>7</b>
	<b>Table 3-1: Use Case Description of Sign-up</b>	<b>14</b>
	<b>Table 3-2: Use Case Description of Sign-in</b>	<b>15</b>
	<b>Table 3-3: Use Case Description of Sign-out</b>	<b>16</b>
	<b>Table 3-4: Use Case Description of Add Vendor</b>	<b>16</b>
	<b>Table 3-5: Use Case Description of Remove Users</b>	<b>17</b>
	<b>Table 3-6: Use Case Description of Edit Users</b>	<b>18</b>
	<b>Table 3-7: Use Case Description of Order Milk</b>	<b>18</b>
	<b>Table 3-8: Use Case Description of Cancel Order</b>	<b>19</b>
	<b>Table 3-9: Use Case Description of Check History</b>	<b>19</b>
	<b>Table 3-10: Use Case Description of Add to cart</b>	<b>20</b>
	<b>Table 3-11: Use Case Description of Checkout</b>	<b>21</b>
	<b>Table 3-12: Use Case Description of Payment</b>	<b>21</b>
	<b>Table 3-13: Use Case Description of Add Rider</b>	<b>22</b>
	<b>Table 3-14: Use Case Description of Check Delivery</b>	<b>23</b>
	<b>Table 3-15: Use Case Description of Check Payment</b>	<b>23</b>
	<b>Table 3-16: Use Case Description of Remove Rider</b>	<b>24</b>
	<b>Table 3-17: Use Case Description of Add Product</b>	<b>25</b>
	<b>Table 3-18: Use Case Description of Remove Product</b>	<b>26</b>

<b>Table 3-19: Use Case Description of Check Order</b>	26
<b>Table 3-20: Use Case Description of Update Status</b>	27
<b>Table 3-21: Use Case Description of Collect Payment</b>	28

## LIST OF FIGURES

<b>FIGURE</b>	<b>TITLE</b>	<b>PAGE</b>
	<b>Figure 3-1 Feature-driven development</b>	11
	<b>Figure 3-2: Use Case Diagram</b>	14
	<b>Figure 3-3 Admin Sequence Diagram 1</b>	29
	<b>Figure 3-4 Admin Sequence Diagram 2</b>	29
	<b>Figure 3-5 Admin Sequence Diagram 3</b>	30
	<b>Figure 3-6 Vendor Sequence Diagram</b>	31
	<b>Figure 3-7 Sequence Diagram For Product Management</b>	31
	<b>Figure 3-8 V Sequence Diagram For Payment Management</b>	32
	<b>Figure 3-9 Vendor Sequence Diagram of Rider Management</b>	32
	<b>Figure 3-10 Sequence Diagram For Order Management</b>	33
	<b>Figure 3-11 Vendor Sequence Diagram For Bill Management</b>	33
	<b>Figure 3-12 Customer Sequence Diagram For Signup</b>	34
	<b>Figure 3-13 Customer Sequence Diagram For Order Product</b>	35
	<b>Figure 3-14 Rider Sequence Diagram 1</b>	36
	<b>Figure 3-15: Class Diagram</b>	38
	<b>Figure 3-16: ER Diagram</b>	39
	<b>Figure 5-1: Sign In</b>	43
	<b>Figure 5-2: Sign up</b>	44
	<b>Figure 5-3: Sign In (Urdu)</b>	44

<b>Figure 5-4: Sign up (Urdu)</b>	45
<b>Figure 5-5: Customer Home Screen</b>	45
<b>Figure 5-6: Side Bar</b>	46
<b>Figure 5-7: Place Order Screen</b>	46
<b>Figure 5-8: Order History</b>	47
<b>Figure 5-9: Cart Screen</b>	47
<b>Figure 5-10: Order Screen (Urdu)</b>	48
<b>Figure 5-11 Customer Screen (Urdu)</b>	48
<b>Figure 5-12 Edit User</b>	49
<b>Figure 5-13 Admin Home Screen</b>	<b>49</b>
<b>Figure 5-14: Add Product</b>	50
<b>Figure 5-15: Create User</b>	50
<b>Figure 5-16: Select Rider</b>	51
<b>Figure 5-17: Product List</b>	51
<b>Figure 5-18: supplier screen (Urdu)</b>	52
<b>Figure 5-19 Rider Screen</b>	52

# CHAPTER 1

## INTRODUCTION

### Background

As we know that the whole world is gradually transferring from manual to digitizable. From Cars to Phones and from eating habits to the appointment of a visit to the doctor. All movement now is done digitally. It was not so long ago when we used pencil and paper for work, now we transfer those written works into digital through the computer or mobile phone. In our society using of e-commerce is increasing day by day. Individuals prefer to buy and sell their goods through e-commerce. Now especially people are buying their daily routine items from e-commerce through different apps. For example, using panda mart and daraz for grocery shopping and uber and cream for traveling purposes. The main theme of our project is to connect the unpacked milk market through e-commerce to reduce the efforts of customers by providing their milk at the doorstep rather than they are wasting their time by standing in the queue. We are doing this by creating an online platform through the app. The objective of our project is to reduce customer efforts in daily life. We will do it by providing unpacked milk at the doorstep instead of buying from the market which takes much time and effort. The secondary objective of our project is to provide enough benefits for the stakeholders who sell their products ranging from Rs 120 - Rs 150 per litter based on the order quantity. Those vendors are not having proper storage facilities, or they are not interested in selling their products at higher prices, so we found a way by using e-commerce technology to make them reachable to customers all over Pakistan.

## **1.1 Problem Statements**

The main theme of our project is to connect the unpacked milk market through e-commerce to reduce the efforts of customers by providing their milk at the doorstep rather than they are wasting their time by standing in the queue. We are doing this by creating an online platform through the app. The objective of our project is to reduce customer efforts in daily life. We will do it by providing unpacked milk at the doorstep instead of buying from the market which takes much time and effort. The secondary objective of our project is to provide enough benefits for the stakeholders who sell their products ranging from Rs 120 - Rs 150 per litter based on the order quantity. Those vendors are not having proper storage facilities, or they are not interested in selling their products at higher prices, so we found a way by using e-commerce technology to make them reachable to customers all over Pakistan.

## **1.2 Aims and Objectives**

The objectives of the thesis are shown as following:

- i. To provide ease to milk consumers.
- ii. Save customer time, money, and effort.
- iii. Provide standard quality and quantity
- iv. Providing Android based platform.
- v. Connect all stakeholders through e-commerce
- vi. To allow users to give order by using application.

## **1.3 Scope of Project**

The primary scope of our project is to connect all stakeholders through one platform. The benefits of our project will impact all participants. Customers don't need any more to stand in the queue. And riders are getting more job opportunists along with vendors

providing milk service. The main idea of our project is to connect the unpacked milk market with an e-commerce platform for the sake of automation and an increase in revenue. Our app is connected to all stakeholders including Customers, Riders, Vendors, Admin. In the customer module, the customer will register itself and make an order of milk daily, weekly basis and monthly basis with specific quantity and payments methods. Customer will check their past ordered and status history. In the riders' module, a rider will operate the order according to the situation and receive the payments from customers and update the status from its side. In the vendor's module, the vendor will register itself and its riders. And the vendor will list the milk quantity according to the customer's needs. The vendor will provide the ordered milk to its rider for delivery purposes and the vendor will check the payment status from the rider and customer side. Proposed application differs from conventional ecommerce applications in order of execution of buying procedure. Proposed application introduces recurrent order delivery systems and easy to use payment methods. Customers can update orders on the go. They can cancel orders or generate order any time.

### **1.3.1 Optional Scope**

- Integrate the digital payment.
- Tracking facility with customers and vendors.



## CHAPTER 2

### SOFTWARE REQUIREMENT SPECIFICATION (SRS)

#### 2.1 User Classes and Characteristics

Following are the user classes and characteristics:

- Admin
- Vendor
- Client/Customer
- Rider

**Administrator** – Admin will be the developer who will be able to perform the following activities.

- Admin Sign-in
- Change Credentials
- Maintain Record
- Sign-out

**Vendor** – Any person who wants a platform to sell his product.

- Sign-up
- Sign-in/Login
- Maintain Supply
- Order History
- Payments
- Receive Order
- Accept Order

- Sign-out

**Client/Customer** – Any person who wants to order a product.

- Sign-up
- Sign-in/Login
- Select Quantity
- Place order
- Order history
- Payments
- Sign-out

## 2.2 Operating Environment

Following are the operating environment are required:

### **Android-Based Requirements:**

- Android version (minimum Android Marshmallow-6.0)

### **Database Management System:**

- Firebase

### **Android Development System:**

- Android Studio 4.1.1
- VS Code 4.0
- Git (for version control)

## 2.3 Design and Implementation Constraints

Milkoride app is an Android-based application, and the front end of Application is designed using Flutter widgets which is powerful sdk build by google. Firebase is used for Database. It will be developed using Feature Driven Development which divides complex structures into subparts. This model recurrence last two phases i.e., Design

by feature and Develop by Feature until all features will complete. The internet connection is also a constraint for the application. Since the application fetches data from the database over the internet, there must be an internet connection for the application to function.

## **2.4 Assumptions and Dependencies**

Assumptions and dependencies are as follow, since its and android base application it requires the following features:

- Internet Connectivity

Without this service' application won't be able to run.

## **2.5 External Interface Requirements**

Following are the external interface requirements of this project:

### **2.5.1 User Interfaces**

Every user should have an Android smartphone with a minimum of the android version of 6.0 so that they can use our App.

### **2.5.2 Software Interfaces**

Following are the required software interfaces:

**Table 2-1: Software required interfaces**

Android Version	6.0+	This app will run on mobiles that are running Android version 6.0+
Database	Firestore	Firestore will be used to save records of the users.
Tools	Android Studio	Android Studio and Visual Studio Code are the development platform that'll be used to develop this application. Git for Version Control.

## 2.6 Other Non-functional Requirements

Non-functional requirements of this project are:

### 2.6.1 Performance Requirements

The mobile device requires a minimum of android version 6.0 to run this application. The system also requires a good internet connection for fast and better performance. The Hybrid application should perform all actions correctly and efficiently. The application should give a response within a time. Internet is very important for this android application to connect with the database. For the best performance of this Hybrid application, the mobile has a good processor and ram. By using these specifications of the mobile the Hybrid application performance for fetching the data from the database, manipulate the data will become fast and it will show the on time.

### 2.6.2 Safety Requirements

To provide the users with the best application experience we will bring time to time updates in our application to prevent any bugs and try to fix those bugs and errors.

### 2.6.3 Security Requirements

Once a customer creates an account their data will be kept secure and they can use any service securely without any security risk. The users provide their name, email, etc. for the register and for login the users provide the user name and password. The database must be secure because it contains all the data of the users. It should be prevented from the external user because a small changing in a database may cause loss of important information.

### 2.6.4 Software Quality Attributes

Software quality attributes of this project are:

- **Availability:** The application will be available for the user 24/7. But booking facility is available from 9 am to 11 pm.
- **Flexibility:** The application would be flexible for any type of user.
- **Usability:** The application should be user friendly for the user. The users easily understand how to use the application.
- **Testability:** The application should be easy to test at each level and find the bugs/defect at each level of development and remove the defects easily.
- **Reusability:** The application is divided into different modules of coding. These modules can be used in across the application.
- **Maintainability:** The application will be easy to maintain and removing bugs/errors and upgrade the application features, functionalities from time to time.

## **CHAPTER 3**

### **DESIGN AND METHODOLOGY**

Design and methodology refers to the development of a system or method for a unique situation. Term is most often applied to technological fields in reference to web design, software or information systems design [1]. Design and methodology stresses the use of brainstorming to encourage innovative ideas and collaborative thinking to work through each proposed idea and arrive at the best solution various degree programs involve design methodology, including those in the graphic and digital arts.

Following artefacts included in this Chapter:

1. System Development Methodology
2. Use case diagram
3. Use case description refined.
4. Sequence Diagram
5. Design Class Diagram
6. ERD Diagram

Now we are going to discuss these artefacts one by one as follows,

### 3.1 System Development Methodology

System Development Methodology is a methodology for systematically organizing the best ways to develop systems efficiently. It includes, for example, descriptions of work to be performed at each stage of the development process and drafted documents. We used the Feature Driven Development (FDD) as the methodology of our project.

#### 3.1.1 Design Model

Feature Driven Development (FDD) is an agile framework that, as its name suggests, organizes software development around making progress on features [2]. Features in the FDD context, though, are not necessarily product features in the commonly understood sense. They are, rather, more akin to user stories in Scrum. In other words, “complete the login process” might be considered a feature in the Feature Driven Development (FDD) methodology. FDD was designed to follow a five-step development process, built largely around discrete “feature” projects. That project lifecycle looks like this:

- Develop an overall model
- Build a features list
- Plan by feature
- Design by feature
- Build by feature

##### **Step 1: Develop an overall model**

One of the core principles of feature-driven development is to use domain object modelling. Domain object modelling is a way to represent connected concepts and the relationships between them. During the first step, an outline of the domain model is created. This object model will be the blueprint for the project.

##### **Step 2: Build a feature list**

The list of features identifies the elements necessary expressed as actions, results, or objects. Each feature should be built in less than two weeks. Features taking longer than two weeks to develop should be broken down into smaller features. For example,

if the feature you are building is “pay on-line bill”, some of the features on the list may include “validate account number” and “retrieve account balance”.

### **Step 3: Plan by feature**

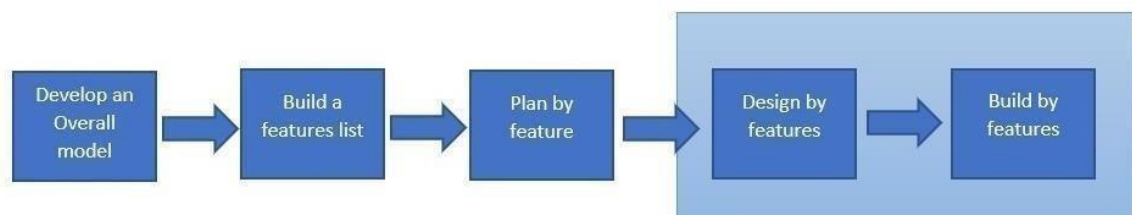
Once the feature list is created, you need to determine the order in which they need to be developed and implemented and which team member is responsible for each feature. All team members should take place in this process to identify potential risks, dependencies, workload constraints, and other obstacles.

### **Step 4: Design by feature**

A design package is created for each feature. The chief programmer identifies which features will be included in each two-week iteration, as well as identify the class owners and feature priorities. The design phase ends with a design review by the entire team. And the domain expert ensures what was built addresses the customer problem being solved.

### **Step 5: Build by feature**

After the design is approved, now it’s time to start writing code. All the items necessary to support the feature design are implemented. These elements can include front-end and back-end elements like user interfaces or database queries. Once everything is completed by individual team members the feature moves onto testing and the process begins again for the next feature on the list.



**Figure 3-1 Feature-driven development**



### 3.2 Use Case Diagram

The use case diagram is the representation of the user with our application which shows the relationship between the users and system use-cases. The use-case diagram for the analysis level is a refined high-level use-case diagram and is the explanation for the high-level use-case diagram. In this diagram, high-level use-cases are expanded to show how high-level use-cases reach their functionality [3]. It defines the roles of our actors, and their interaction with system use cases.

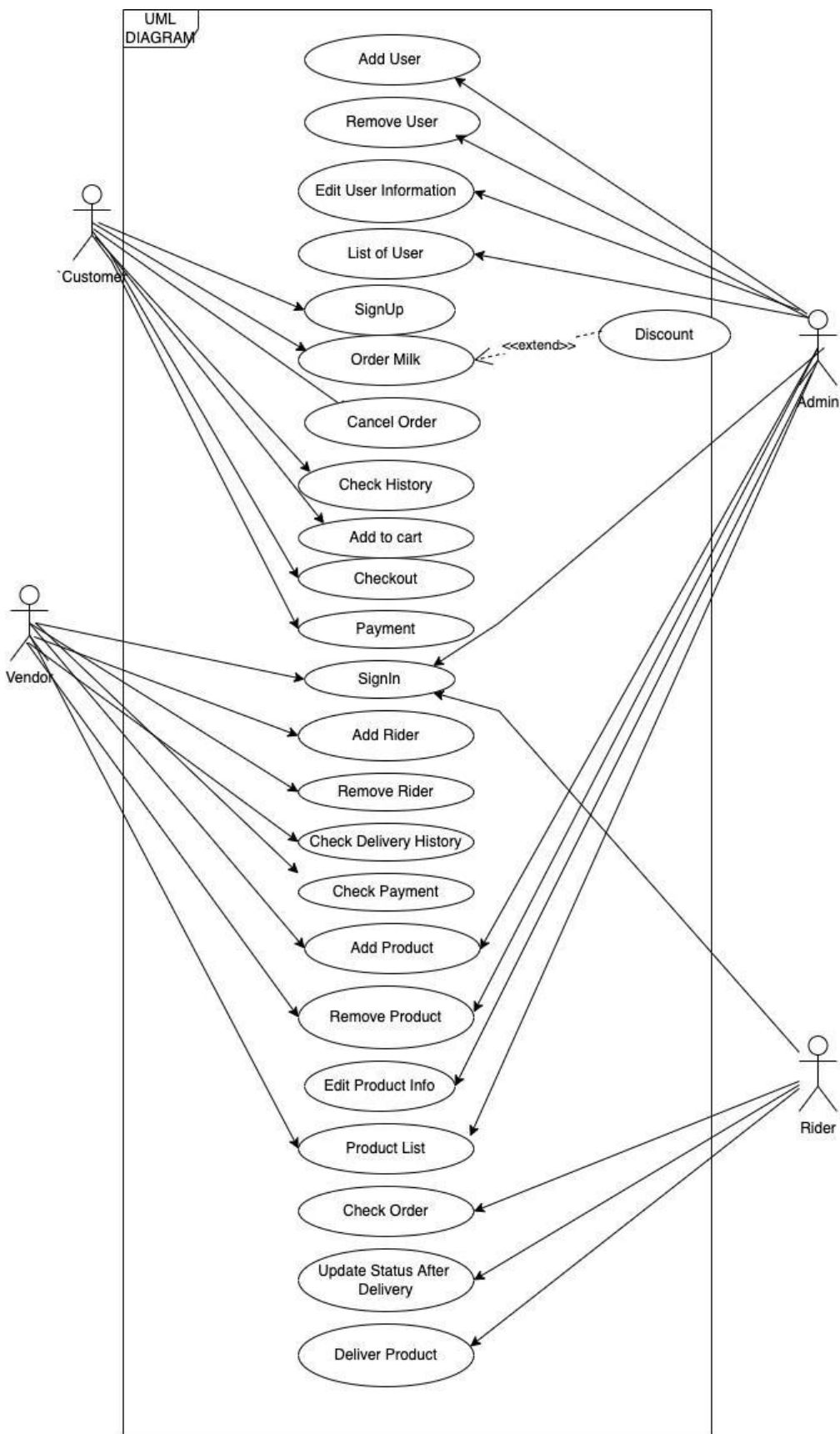
**Customer:** Customer can sign-up, sign-in and sign-out to our application, further he's the one to place order.

**Admin:** Admin has the rights of sign-in and sign-out as he has the role of record maintenance.

**Vendor:** vendor can also s sign-in and sign-out to our application, further he's the one to perform task of order conformation and product inventory maintenance.

System use case diagram of our application is shown in figure 3.1:

**Rider:** Rider has to deliver products to customers and collects the payments.and update status also.



### Figure 3-2: Use Case Diagram

### 3.3 Use Case Description

Following are the use-case description of our application:

#### 3.3.1 Sign-up Use Case Description

**Table 3-1: Use Case Description of Sign-up**

<b>Name and ID</b>	<b>Sign-Up (U-1)</b>
<b>Brief Description</b>	Users (Customer) can register themselves to our Application. During Sign-up application will get the necessary information and the user will provide the required information.
<b>Preconditions</b>	User must have an internet connection before registering/sign-up to our application.
<b>Basic Flow or Happy Path</b>	<ol style="list-style-type: none"> <li>1. User will provide their name.</li> <li>2. Users will provide their contact information.</li> <li>3. User will provide their email.</li> <li>4. Users will write their passwords.</li> <li>5. Admin will verify the required information for Sign-up/Registration.</li> <li>6. After registration user can Sign-in to the application according to his role.</li> </ol>
<b>Trigger</b>	The user wants an application which facilitates Customer.

<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Users might provide the email that already exists in the database. The system will generate an error message “Email already exists” and redirect to the login screen.
<b>Post Conditions</b>	Users (Customers) got Signed-up.

### 3.3.2 Sign-In Use Case Description

**Table 3-2: Use Case Description of Sign-in**

<b>Name and ID</b>	<b>Sign-In (U-2)</b>
<b>Brief Description</b>	Admin, Vendors, Riders and Customers, can log in to our application using their login credential (i.e., email, password).
<b>Preconditions</b>	User must be already registered.
<b>Basic Flow or Happy Path</b>	<ol style="list-style-type: none"> <li>1. User will enter their email.</li> <li>2. User will enter their password.</li> <li>3. After authentication users can log in to our application.</li> </ol>
<b>Trigger</b>	The user wants an application which facilitates Vendor/Customer/Rider/Admin.
<b>Alternate Flows</b>	<p><b>Alternative Flow 1</b> User can enter wrong email or password or that credentials which are not registered So, the application will pop-up an error message “non-valid credentials” and again redirect it to Sign-in screen.</p> <p><b>Alternative Flow 2</b> Users might forget their credentials (password). User can forget his password to renew his credentials (password).</p>
<b>Post Conditions</b>	User has access to the application and performs operations according to his role..

### 3.3.3 Sign-out Use Case Description

**Table 3-3: Use Case Description of Sign-out**

<b>Name and ID</b>	<b>Sign-Out (U-3)</b>
<b>Brief Description</b>	Admin, Vendors, Riders and Customers, can Sign-out to our application on one tap to protect his credential or to Sign-in with another user role.
<b>Preconditions</b>	User must be already registered (U-1) and Signed-in to the application (U-2).
<b>Basic Flow or Happy Path</b>	Must fulfil (U-2) and then can Sign-out to application.
<b>Trigger</b>	Sign-Out.
<b>Alternate Flows</b>	There is not any alternative flow.
<b>Post Conditions</b>	Sign-out from the application.

### 3.3.4 Add User Use Case Description

**Table 3-4: Use Case Description of Add Vendor**

<b>Name and ID</b>	<b>Add Users (U-4)</b>
<b>Brief Description</b>	After Signing-in to application Admin can add Users.
<b>Preconditions</b>	User must be Signed-In (U-2).
<b>Basic Flow or Happy Path</b>	<ol style="list-style-type: none"> <li>1. Admin must perform U-2 first.</li> <li>2. Admin add User.</li> </ol>

<b>Trigger</b>	Admin wants to add user
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Admin wants to add vendor but vendor is already added
<b>Post Conditions</b>	User added to application

### 3.3.5 Remove User Use Case Description

**Table 3-5: Use Case Description of Remove Users**

<b>Name and ID</b>	<b>Remove Users (U-5)</b>
<b>Brief Description</b>	After adding users to application Admin can remove users.
<b>Preconditions</b>	Admin must be Signed-In (U-2). Users must be Added (U-4)
<b>Basic Flow or Happy Path</b>	<ol style="list-style-type: none"> <li>1. Admin must perform U-2 first.</li> <li>2. Admin must perform U-4 first.</li> <li>3. Admin Remove Users.</li> </ol>
<b>Trigger</b>	Admin wants to remove user.
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Admin wants to remove users but Users is not added.
<b>Post Conditions</b>	Users removed from application

### 3.3.6 Edit Users Use Case Description

**Table 3-6: Use Case Description of Edit Users**

<b>Name and ID</b>	<b>Edit Users (U-6)</b>
<b>Brief Description</b>	After adding users to application Admin can edit users.
<b>Preconditions</b>	Admin must be Signed-In (U-2). Users must be Added (U-4)
<b>Basic Flow or Happy Path</b>	<ol style="list-style-type: none"> <li>4. Admin must perform U-2 first.</li> <li>5. Admin must perform U-4 first.</li> <li>6. Admin Edit Users.</li> </ol>
<b>Trigger</b>	Admin wants to edit users.
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Admin wants to edit vendor but users is not added.
<b>Post Conditions</b>	Users edited from application

### 3.3.7 Order Milk Use Case Description

**Table 3-7: Use Case Description of Order Milk**

<b>Name and ID</b>	<b>Place Order (U-7)</b>
<b>Brief Description</b>	Customers can place an order of milk
<b>Preconditions</b>	User must be Signed-In (U-2).
<b>Basic Flow or Happy Path</b>	After Signing-in. Customers can place an order of milk
<b>Trigger</b>	The user wants to order milk

<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Users can cancel the placed order before the order is locked by the vendor.
<b>Post Conditions</b>	User can proceed further to checkout

### 3.3.8 Cancel Order Use Case Description

**Table 3-8: Use Case Description of Cancel Order**

<b>Name and ID</b>	<b>Cancel Order (U-8)</b>
<b>Brief Description</b>	Customers can cancel order of milk
<b>Preconditions</b>	User must be Signed-In (U-2). Order must be placed (U-7).
<b>Basic Flow or Happy Path</b>	After Signing-in and Placing Order, Customers can cancel order of milk.
<b>Trigger</b>	The user wants to cancel order of milk
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Users wants to cancel order of milk but order is not placed yet.
<b>Post Conditions</b>	User can proceed further to add items to cart.

### 3.3.9 Check History Use Case Description

**Table 3-9: Use Case Description of Check History**

<b>Name and ID</b>	<b>Cancel Order (U-9)</b>
<b>Brief Description</b>	Customers can check History of orders
<b>Preconditions</b>	User must be Signed-In (U-2). Order must be placed (U-7).



<b>Basic Flow or Happy Path</b>	After Signing-in and Placing Order, Customers can check History
<b>Trigger</b>	The user wants to check History of orders.
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Users wants to check History but order is not placed yet.
<b>Post Conditions</b>	User can proceed further to checkout

### 3.3.10 Add To Cart Use Case Description

**Table 3-10: Use Case Description of Add to cart**

<b>Name and ID</b>	<b>Add to Cart (U-10)</b>
<b>Brief Description</b>	Customers can add to cart products
<b>Preconditions</b>	User must be Signed-In (U-2).
<b>Basic Flow or Happy Path</b>	After Signing-in, Customers can check products and add to cart.
<b>Trigger</b>	The user wants to add to cart product.
<b>Alternate Flows</b>	No Alternative flow
<b>Post Conditions</b>	User can proceed further to checkout

### 3.3.11 Checkout Use Case Description

**Table 3-11: Use Case Description of Checkout**

<b>Name and ID</b>	<b>Add to Cart (U-11)</b>
<b>Brief Description</b>	Customers can check out the added product
<b>Preconditions</b>	User must be Signed-In (U-2). Product must be added to cart (U-10)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• User signed in</li> <li>• User added product to cart</li> <li>• User proceed to checkout.</li> </ul>
<b>Trigger</b>	The user wants to proceed to checkout
<b>Alternate Flows</b>	No Alternative flow
<b>Post Conditions</b>	User can proceed further to payment

### 3.3.12 Payment Use Case Description

**Table 3-12: Use Case Description of Payment**

<b>Name and ID</b>	<b>Add to Cart (U-12)</b>
<b>Brief Description</b>	Customers can do payment of order
<b>Preconditions</b>	User must be Signed-In (U-2). Product must be added to cart (U-10) Product must be Checkout (U-11)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• User signed in</li> <li>• User added product to cart</li> <li>• User proceed to checkout.</li> </ul>

	<ul style="list-style-type: none"> <li>User do payment</li> </ul>
<b>Trigger</b>	The user wants to do payment
<b>Alternate Flows</b>	User skip payment for later.
<b>Post Conditions</b>	User can proceed further to add more products.

### 3.3.13 Add Rider Use Case Description

**Table 3-13: Use Case Description of Add Rider**

<b>Name and ID</b>	<b>Add Rider (U-13)</b>
<b>Brief Description</b>	After Signing-in to application Vendor can add rider.
<b>Preconditions</b>	Vendor must be Signed-In (U-2). Order must be placed (U-7).
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>User placed order (U-11)</li> <li>Vendor assign rider</li> </ul>
<b>Trigger</b>	Vendor wants to add rider
<b>Alternate Flows</b>	No alternative flow
<b>Post Conditions</b>	Rider added to application

### 3.3.14 Check Delivery Use Case Description

**Table 3-14: Use Case Description of Check Delivery**

<b>Name and ID</b>	<b>Remove Rider (U-14)</b>
<b>Brief Description</b>	Vendor can check delivery status of order
<b>Preconditions</b>	Vendor must be Signed-In (U-2). Order must be placed (U-7). Rider must be added (U-13)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Vendor signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Vendor can check delivery status</li> </ul>
<b>Trigger</b>	Vendor wants to check delivery status
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Vendor wants to check delivery status but rider not assigned yet.
<b>Post Conditions</b>	Vendor checked the delivery status

### 3.3.15 Check Payment Use Case Description

**Table 3-15: Use Case Description of Check Payment**

<b>Name and ID</b>	<b>Check Payment (U-15)</b>
<b>Brief Description</b>	Vendor can check payment status of order
<b>Preconditions</b>	Vendor must be Signed-In (U-2).

	Order must be placed (U-7). Rider must be added (U-13)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Vendor signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Vendor can check payment status</li> </ul>
<b>Trigger</b>	Vendor wants to check payment status
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Vendor wants to check payment status but rider not assigned yet.
<b>Post Conditions</b>	Vendor checked the payment status

### 3.3.16 Remove Rider Use Case Description

**Table 3-16: Use Case Description of Remove Rider**

<b>Name and ID</b>	<b>Remove Rider (U-16)</b>
<b>Brief Description</b>	After adding rider, vendor can remove rider.
<b>Preconditions</b>	Vendor must be Signed-In (U-2). Order must be placed (U-7). Rider must be added (U-13)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Vendor signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Vendor wants to remove rider</li> </ul>
<b>Trigger</b>	Vendor wants to remove rider

<b>Alternate Flows</b>	No alternative flow
<b>Post Conditions</b>	Rider removed from application

### 3.3.17 Add Product Use Case Description

**Table 3-17: Use Case Description of Add Product**

<b>Name and ID</b>	<b>Add Product (U-17)</b>
<b>Brief Description</b>	After signing, vendor can add products to application
<b>Preconditions</b>	Vendor must be Signed-In (U-2).
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Vendor signed in (U-2)</li> <li>• Vendor wants to add product</li> </ul>
<b>Trigger</b>	Vendor wants to add product
<b>Alternate Flows</b>	No alternative flow
<b>Post Conditions</b>	Product added to application

### 3.3.18 Remove Product Use Case Description

**Table 3-18: Use Case Description of Remove Product**

<b>Name and ID</b>	<b>Remove Product (U-18)</b>
<b>Brief Description</b>	After adding products vendor can remove products to application
<b>Preconditions</b>	Vendor must be Signed-In (U-2). Product must be Added (U-17)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Vendor signed in (U-2)</li> <li>• Vendor added product (U-17)</li> <li>• Vendor wants to remove product</li> </ul>
<b>Trigger</b>	Vendor wants to remove product
<b>Alternate Flows</b>	No alternative flow
<b>Post Conditions</b>	Product removed from application

### 3.3.19 Check Order Use Case Description

**Table 3-19: Use Case Description of Check Order**

<b>Name and ID</b>	<b>Check History (U-19)</b>
<b>Brief Description</b>	Rider can check Orders
<b>Preconditions</b>	Rider must be Signed-In (U-2). Order must be placed (U-7). Rider must be added (U-13)

<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Rider signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Customer placed order (U-7)</li> <li>• Rider can check order.</li> </ul>
<b>Trigger</b>	Vendor wants to check Order
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Rider wants to check order but rider not assigned yet.
<b>Post Conditions</b>	Rider can proceed to collect money

### 3.3.20 Update Status Use Case Description

**Table 3-20: Use Case Description of Update Status**

<b>Name and ID</b>	<b>Update Status (U-20)</b>
<b>Brief Description</b>	Rider can update status.
<b>Preconditions</b>	Rider must be Signed-In (U-2). Order must be Placed (U-7). Rider must be Added (U-13)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Rider signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Customer placed order (U-7)</li> <li>• Rider can update status.</li> </ul>
<b>Trigger</b>	Rider wants to update status
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Rider wants to update status but order not confirmed yet.



<b>Post Conditions</b>	Rider can proceed to collect payment.
------------------------	---------------------------------------

### 3.3.21 Collect Payment Use Case Description

**Table 3-21: Use Case Description of Collect Payment**

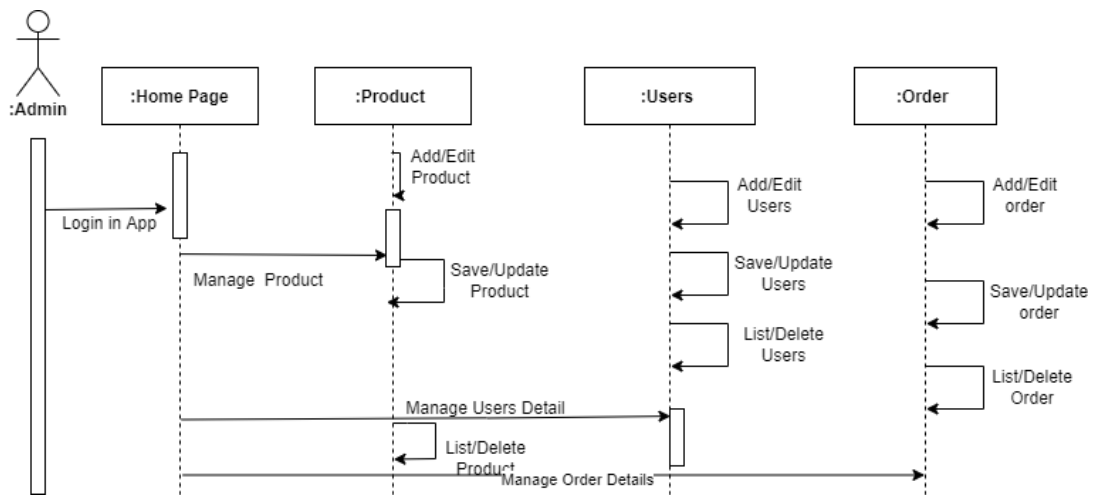
<b>Name and ID</b>	<b>Update Status (U-21)</b>
<b>Brief Description</b>	Rider can collect Payment
<b>Preconditions</b>	Rider must be Signed-In (U-2). Order must be Placed (U-7). Rider must be Added (U-13)
<b>Basic Flow or Happy Path</b>	<ul style="list-style-type: none"> <li>• Rider signed in (U-2)</li> <li>• Vendor added rider (U-13)</li> <li>• Customer placed order (U-7)</li> <li>• Rider can collect Payment</li> </ul>
<b>Trigger</b>	Rider wants to collect payment
<b>Alternate Flows</b>	<b>Alternative Flow 1</b> Rider wants to collect payment but payment not received yet.
<b>Post Conditions</b>	Rider collected the payment.

## 3.4 Sequence Diagrams

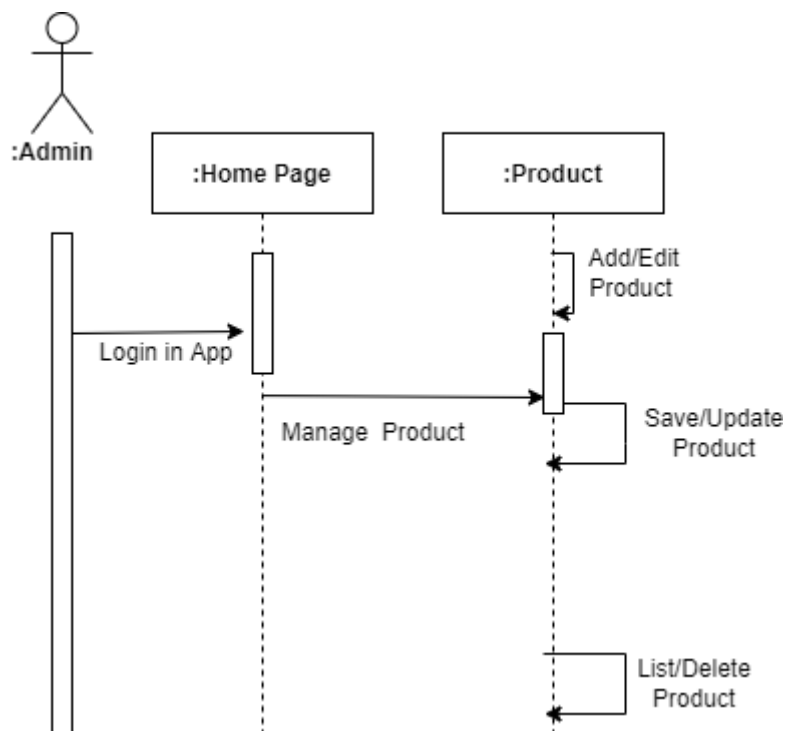
The series of actions that occur in a system is shown in a sequence diagram. In a sequence diagram, the invocation of methods for each object and the order in which the invocation occurs is captured. This makes the Sequence Diagram a very helpful tool for the dynamic behaviour of a system to be easily represented.

### 3.4.1 Admin Sequence Diagram

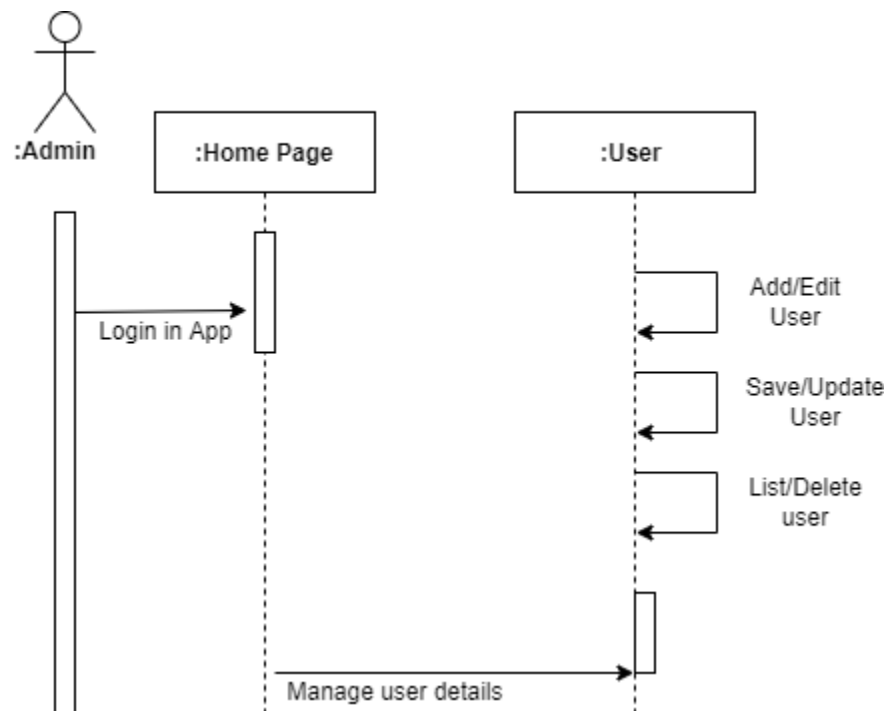
This sign-up sequence diagram defines the flow of actor and operations he performs. The user can send a request for registration with his valid credentials after that credential has been validated the sign-up record for a new user has been saved to the database.



**Figure 3-3 Admin Sequence Diagram 1**



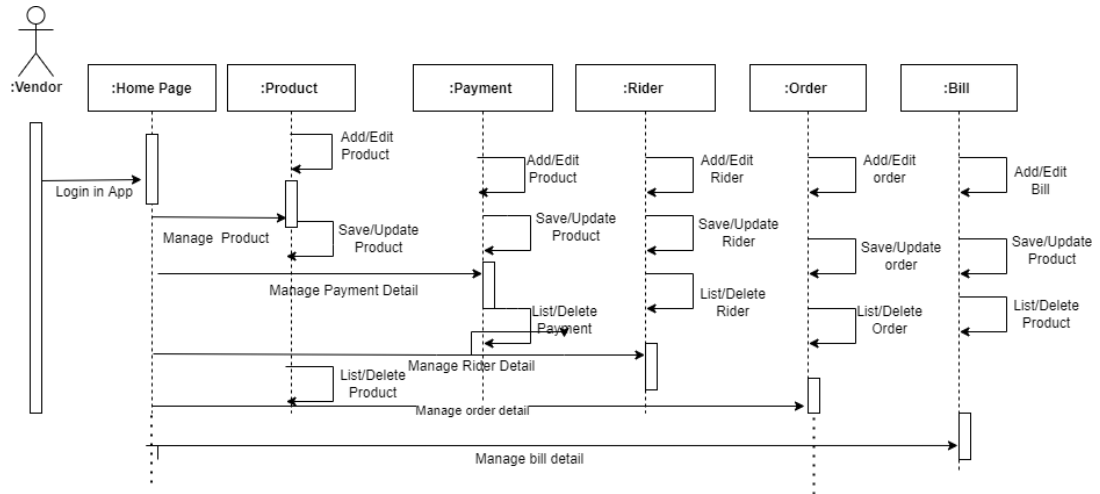
**Figure 3-4 Admin Sequence Diagram 2**



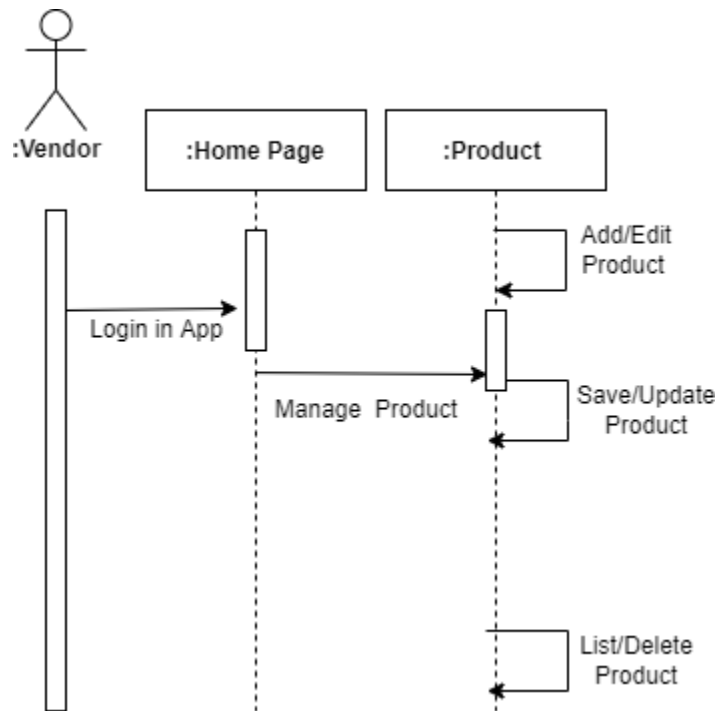
**Figure 3-5 Admin Sequence Diagram 3**

### 3.4.2 Vendor Sequence Diagram

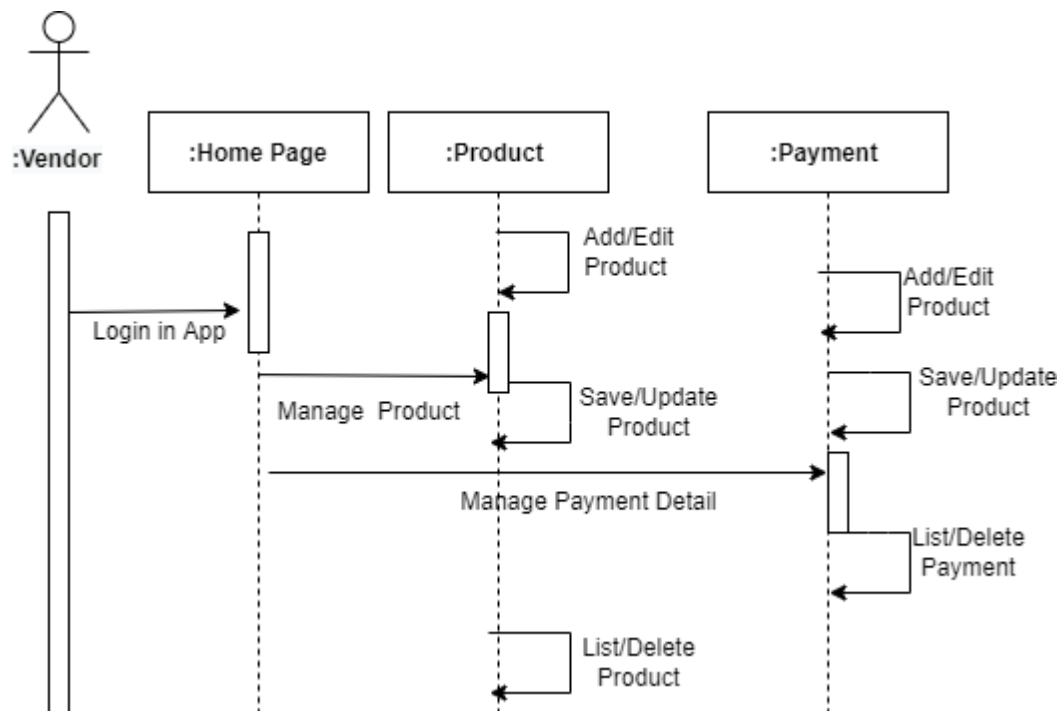
This sign-up sequence diagram defines the flow of actor and operations he performs. The user can send a request for registration with his valid credentials after that credential has been validated the sign-up record for a new user has been saved to the database. After then acknowledge has returned that a new user has been signed-up and the user navigated to the login screen.



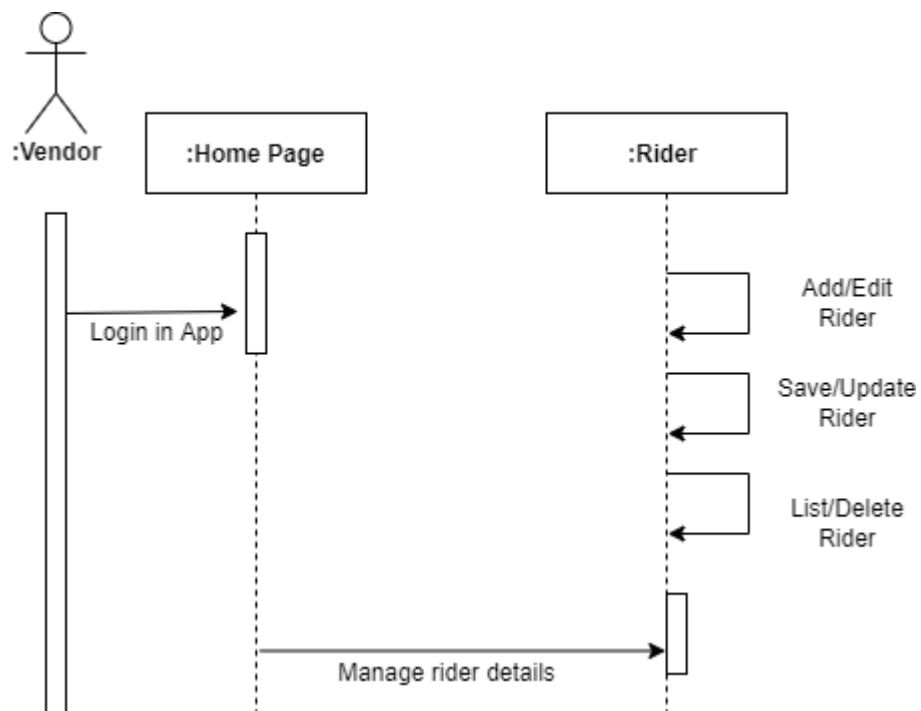
**Figure 3-6 Vendor Sequence Diagram**



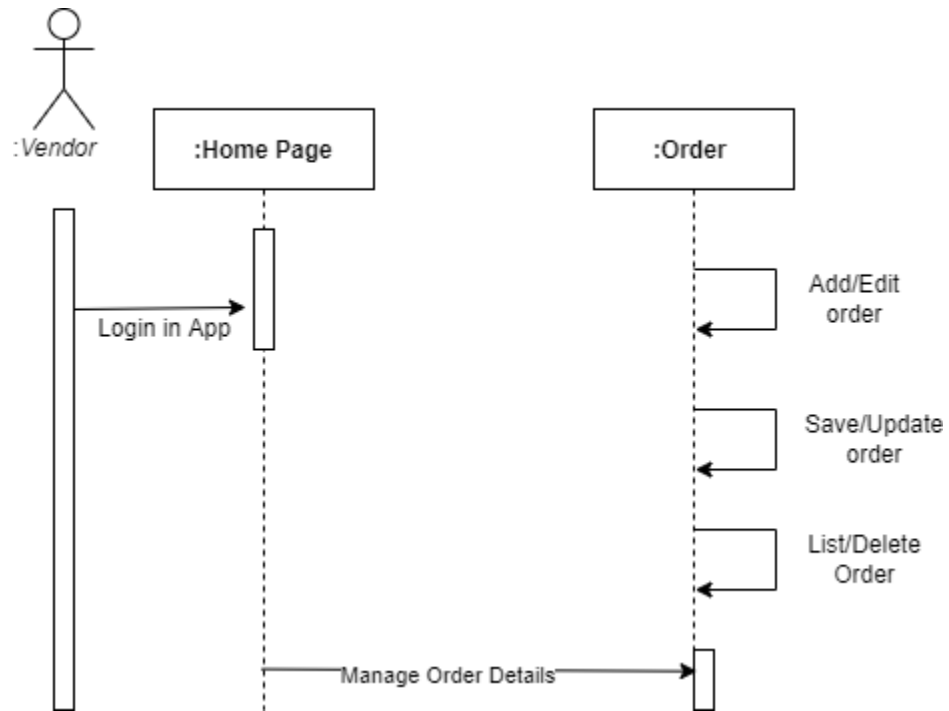
**Figure 3-7 Vendor Sequence Diagram For Product Management**



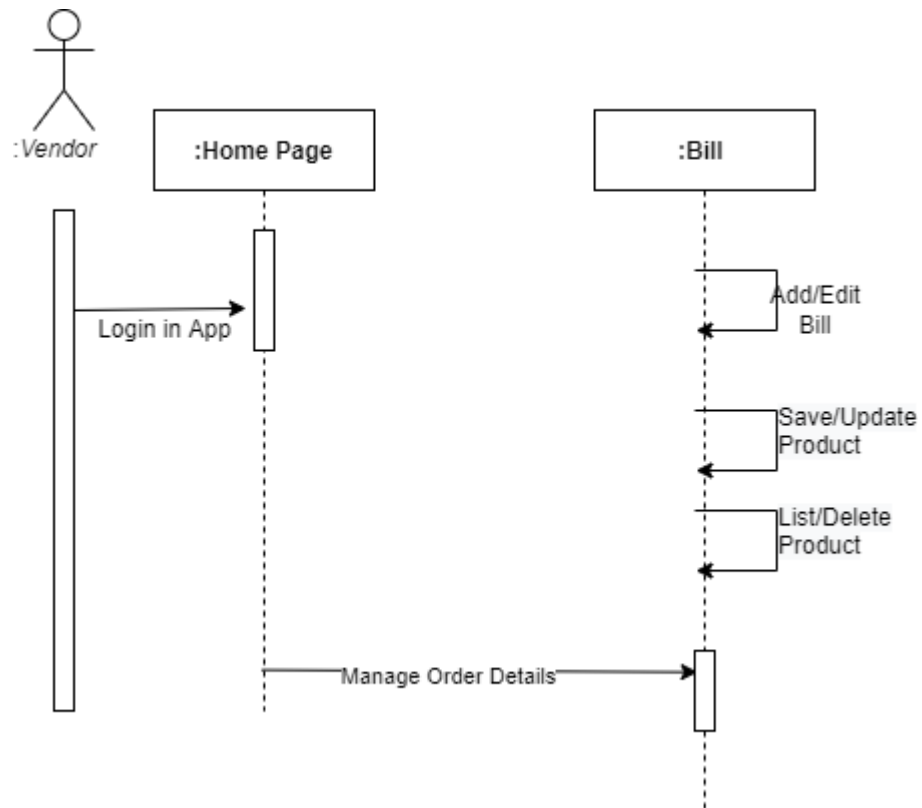
**Figure 3-8 Vendor Sequence Diagram For Payment Management**



**Figure 3-9 Vendor Sequence Diagram of Rider Management**



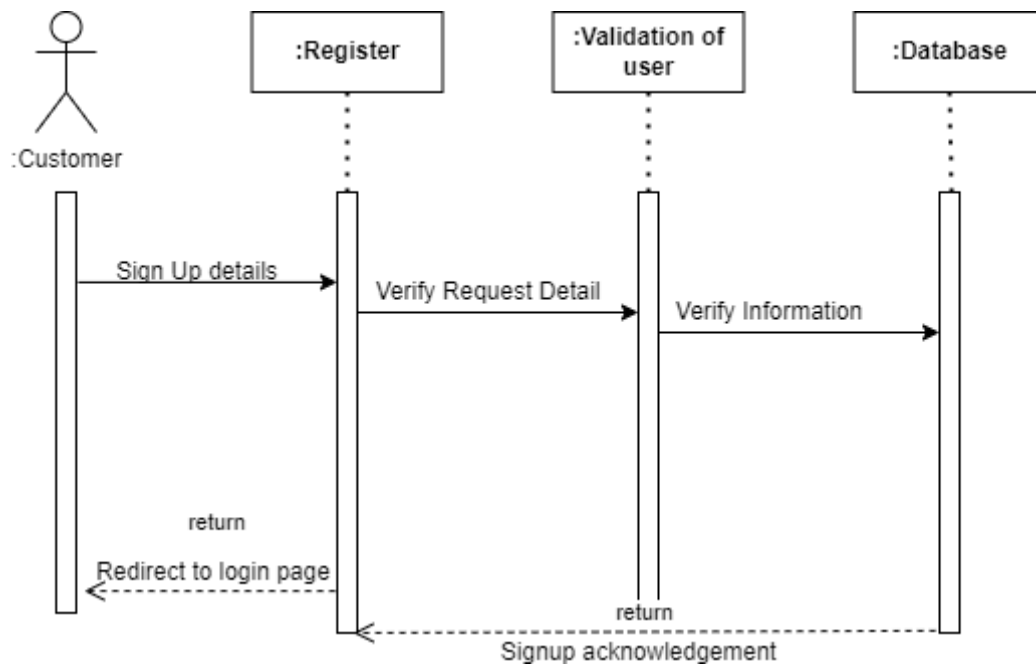
**Figure 3-10 Vendor Sequence Diagram For Order Management**



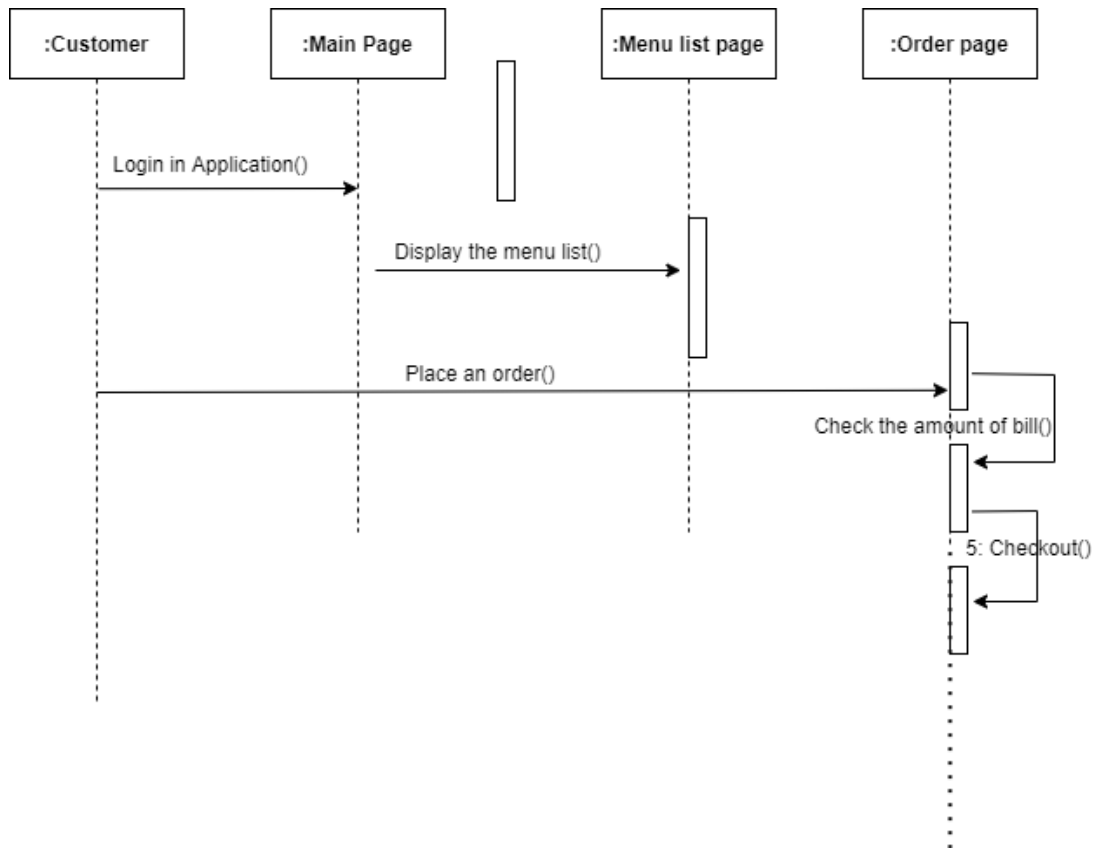
**Figure 3-11 Vendor Sequence Diagram For Bill Management**

### 3.4.3 Customer Sequence Diagram

This sign-in sequence diagram defines the flow of actor and operations he performs. The user can enter his valid credentials in fields after that the entered credential will be validated from the database that the user who is trying to sign-in is already registered or not. If credentials are validated successful sign-in acknowledgement has been returned and the user navigated to main HomeScreen.



**Figure 3-12 Customer Sequence Diagram For Signup**

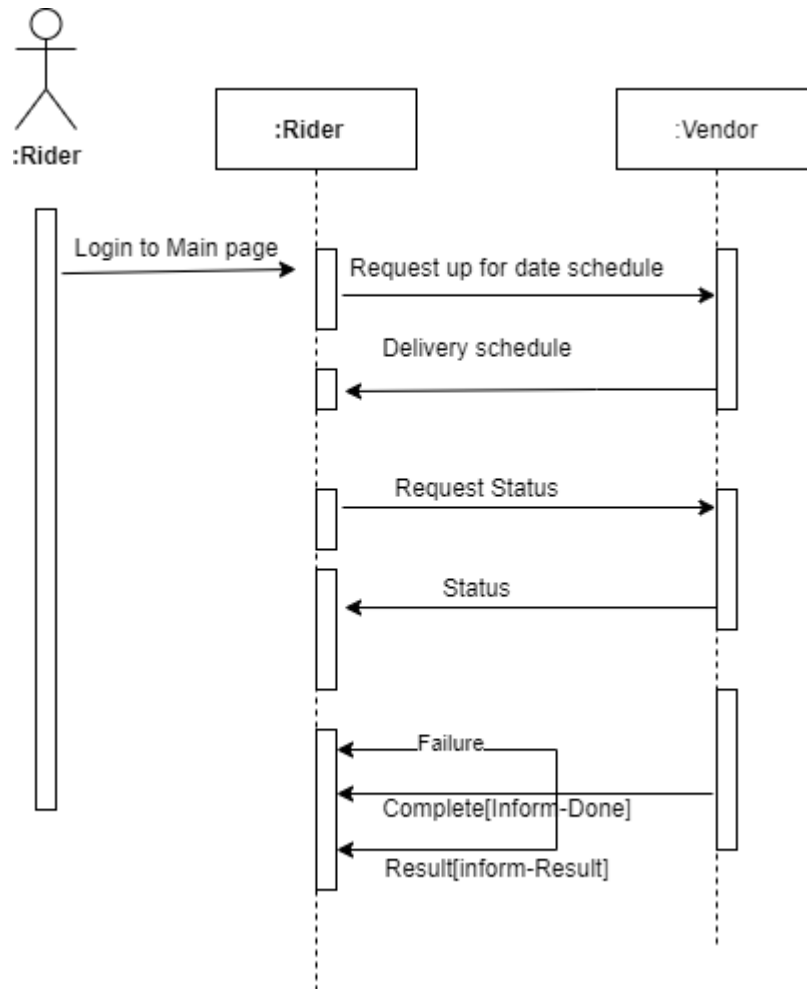


**Figure 3-13 Customer Sequence Diagram For Order Products**

### 3.4.4 Rider Sequence Diagram

The sign-out sequence diagram has defined the flow of signing-out from application the flow is as the user can request for sign-out by pressing sign-out button in the user profile after that a request has been sent to temporarily remove user's data from the application after request approved user has been signed-out.





**Figure 3-14 Rider Sequence Diagram 1**

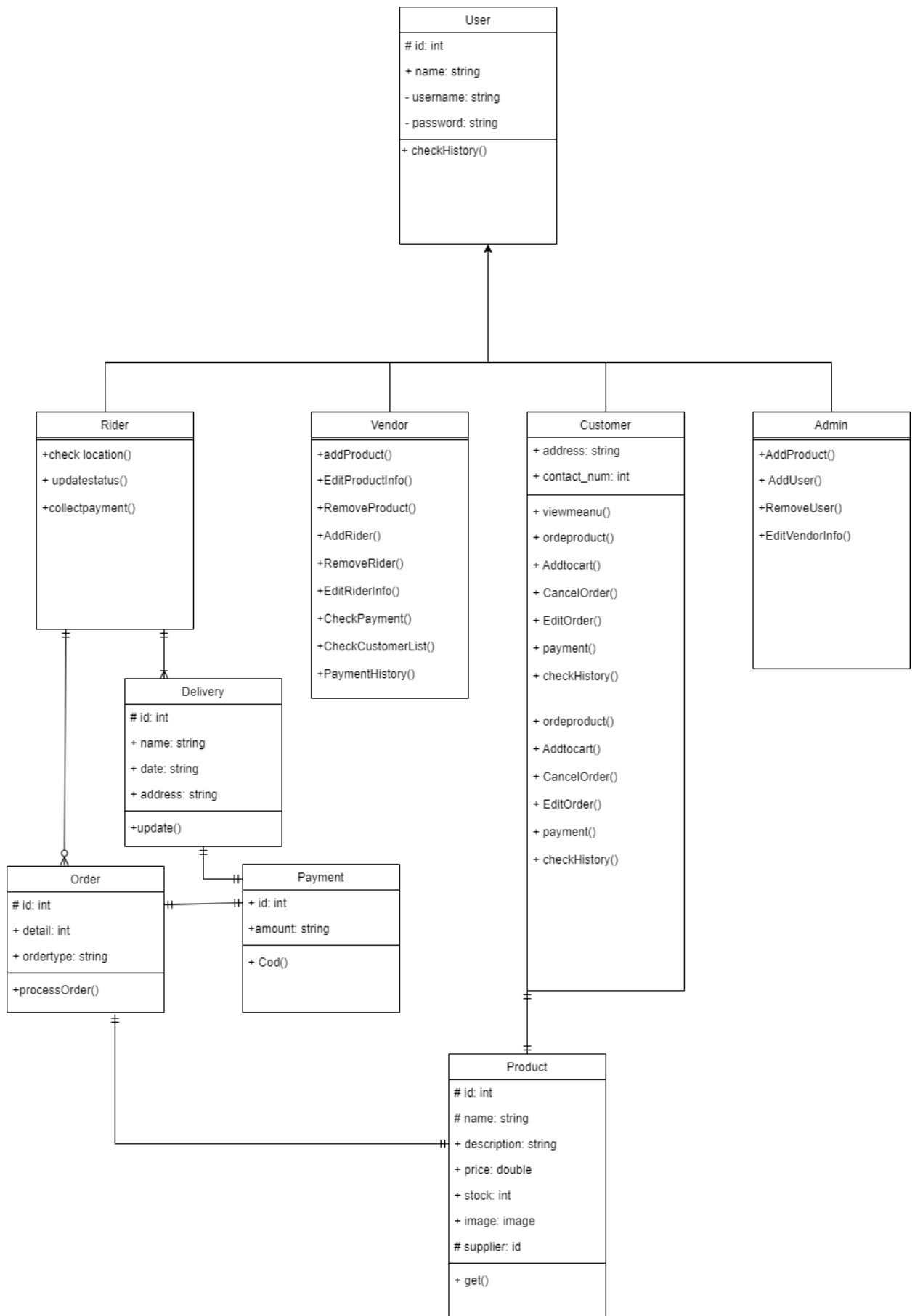
### 3.5 Design Class Diagram

Classes are the work-horses of the design effort, who carry out the system's real work. Subsystems, packages and collaborations, the other design components, simply explain how classes are grouped or how they interoperate.

The following concept class diagram is our application model's main building block, which shows the object, attributes, process, and the relationship between them. The following diagram depicts a certain object, characteristics and relationship that can be formed as the vendor has certain characteristics and activities and has a 1-to-many relationship with the customer and the customer has certain characteristics and a 1-to-1 relationship with the vendor as one customer can communicate with one

vendor at a time. Also, as a single product can be viewed at a single moment, the product has a 1-to-1 relationship with augmented reality. as admin is responsible for the multiple record management at a time, Admin has a 1-to-many relationship.

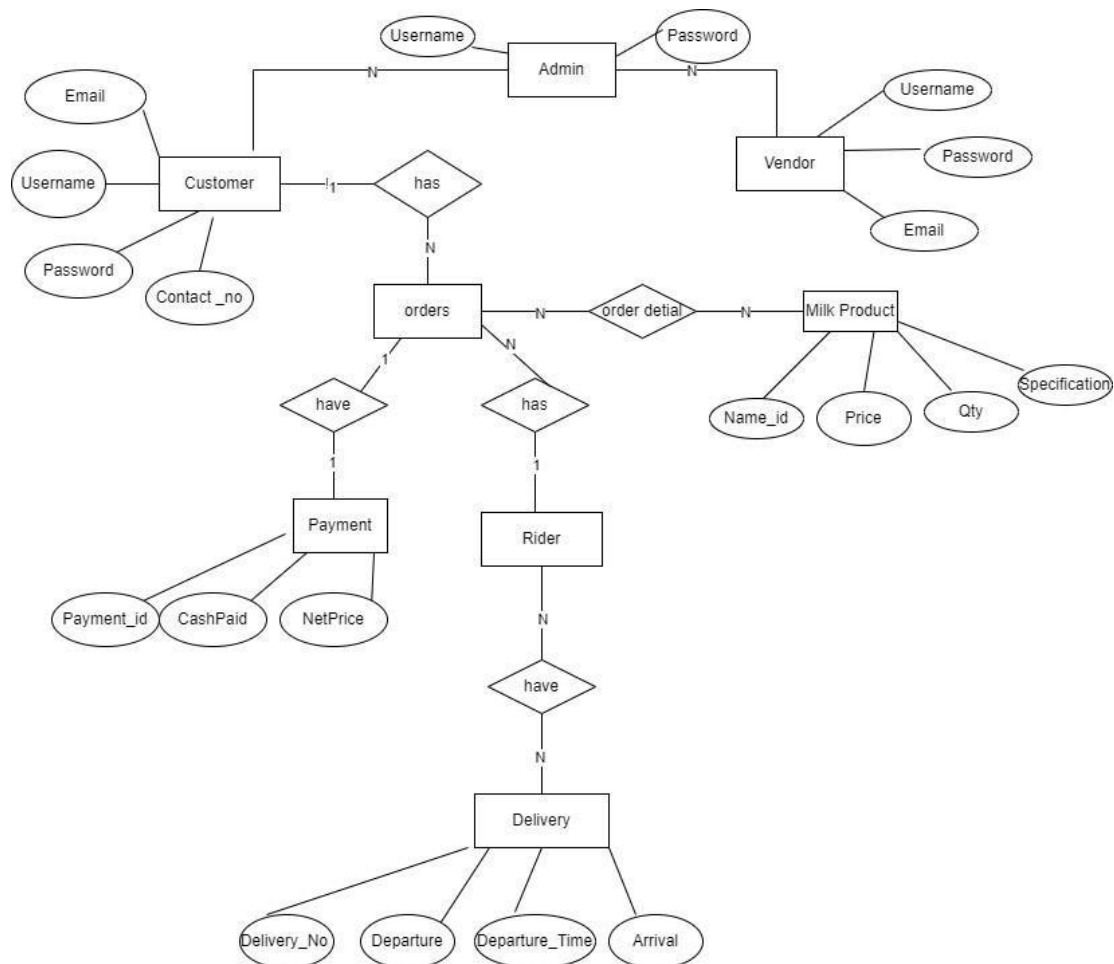
Class diagram of our application is given by the figure:



**Figure 3-15: Class Diagram**

### 3.6 ERD Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research [5]. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.



**Figure 3-16: ER Diagram**

## CHAPTER 4

### IMPLEMENTATION

This chapter describes the tools and technologies used to make Online Election System.

Explanation of this chapter is explained below:

#### 4.1 Technologies Used

The following technologies are used in this project

- Flutter
- Dart
- Firestore Database
- Firebase Authentication
- Cloud Storage

##### 4.1.1 Flutter

Flutter is a cross-platform, open-source mobile app development framework from Google. Flutter enables the creation of Android and iOS apps using the same codebase.

The project's front-end was built with version 2.5.1.0.

```
class MyApp extends StatelessWidget {  
  MyApp({Key? key}) : super(key: key);  
  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      translations: LocaleString(),  
      locale: const Locale('en', 'US'),  
      theme: ThemeData(  
        fontFamily: GoogleFonts.openSans().fontFamily,  
        visualDensity: const VisualDensity(vertical: 1, horizontal: 1),  
      ), // ThemeData  
      initialRoute: '/',  
      routes: {  
        '/': (context) => HomePage(),  
        '/login': (context) => LoginScreen(),  
      },  
      title: "Milkoride",  
    ); // MaterialApp  
  }  
}
```

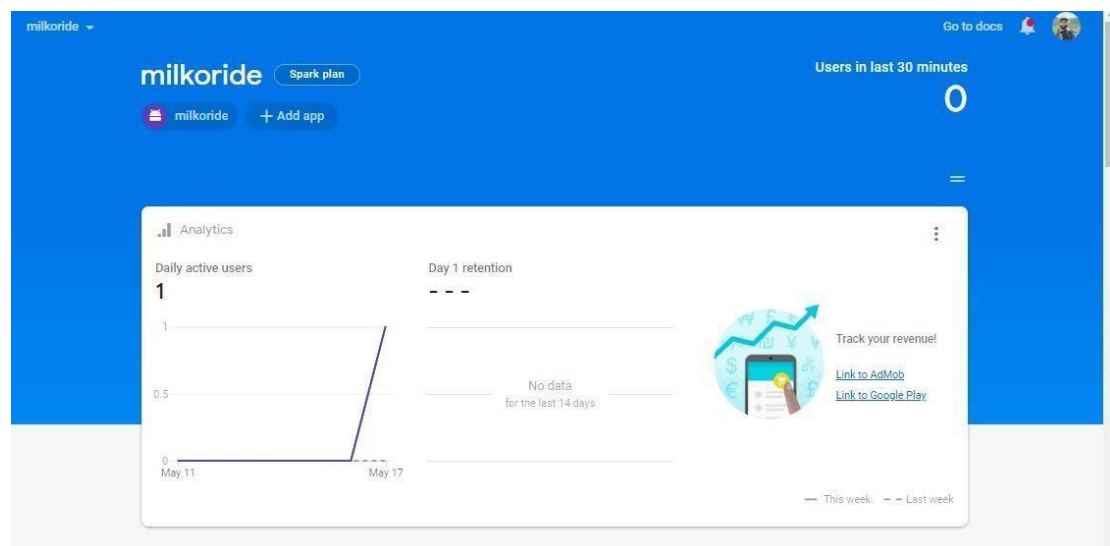
### 4.1.2 Dart

Flutter apps are written in Dart, a computer language. Dart is another Google product that was published in November, one month before Flutter. The Flutter community is still small in comparison to ReactNative, Ionic, or Xamarin. I just developed a fondness for JavaScript.

### 4.1.3 Firestore Database

The Firebase Realtime Database is a database stored in the cloud. Data is saved in JSON format and synced in real time across all connected clients. When you use our Apple, Android, and JavaScript SDKs to create cross-platform apps, all of your clients share a single Realtime Database instance and are immediately updated with the most recent data in Firebase Authentication

To authenticate users to our project, Firebase Authentication is used. It delivers UI framework, backend services and easy-to-use SDKs. It allows, Login, Logout, Signup and Reset Password etc.



### 4.1.4 Cloud Storage

Cloud storage is a cloud computing approach in which data is stored on the Internet and managed and operated by a cloud computing provider. It's on-demand, with just-in-time capacity and pricing, and it saves you money by not having to acquire and manage your own data storage infrastructure. With "anytime, everywhere" data access, you get agility, global scale, and durability.

The following technologies are used in this project

- Android Studio
- Android Emulator

#### **4.2.1 Android Studio**

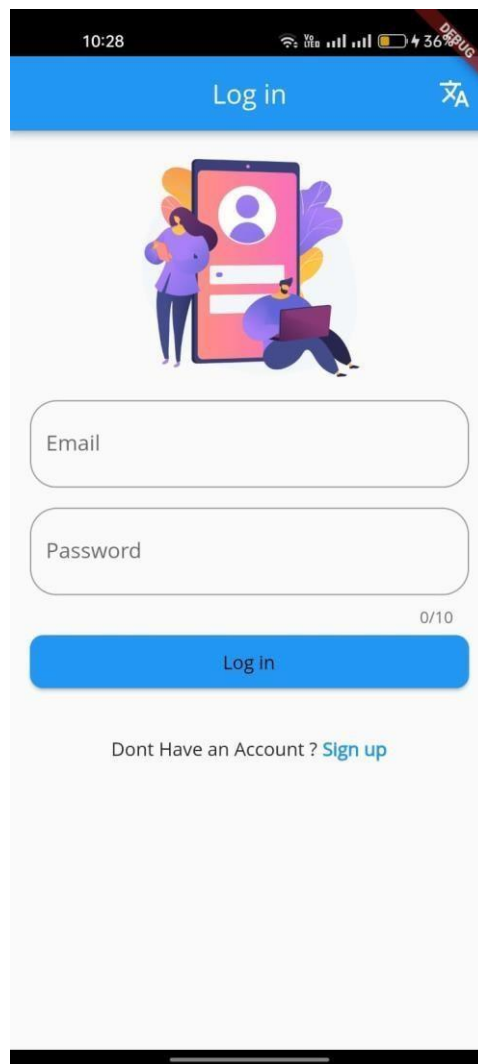
We used Android Studio Bumblebee 2022 for our project, it is a software that lets users to develop Android applications, although it involves coding understanding.

#### **4.2.2 Android Emulator**

We used Android Emulator version Nexus 6 API28 on our system to simulate an android device so that we could test our app on numerous devices

## CHAPTER 5

### USER MANUAL



**Figure 5-1: Sign In**



10:29

← Sign up

Name

Email

Password 0/10

Shipping Address

Sign up

**Figure 5-2: Sign up**

10:28

لاگ ان کریں

ای میل

پاس ورڈ 0/10

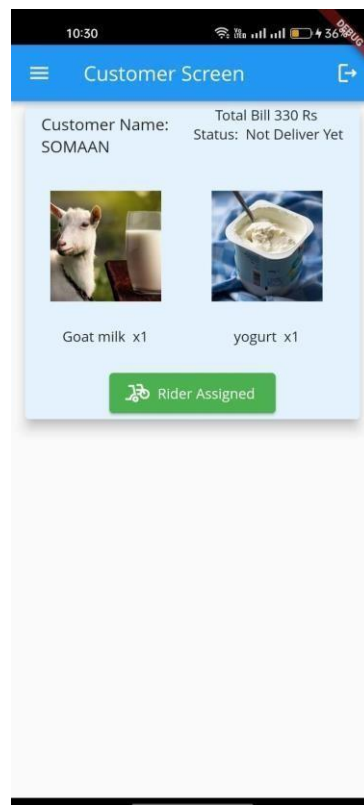
لاگ ان کریں

سائن اپ اکاؤنٹ نہیں ہے۔

**Figure 5-3: Sign In (Urdu)**



**Figure 5-4: Sign up (Urdu)**



**Figure 5-5: Customer Home Screen**

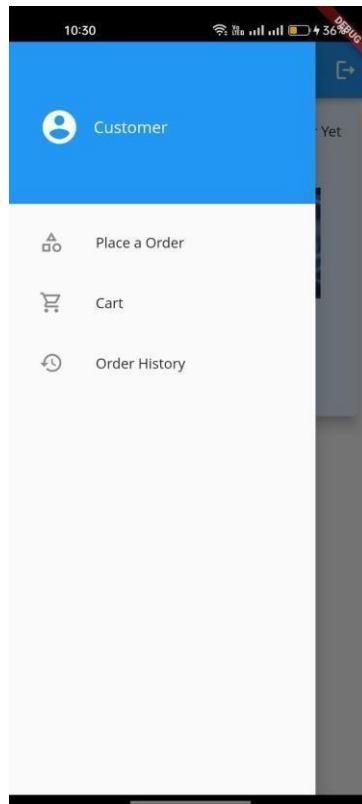


Figure 5-6: Side Bar

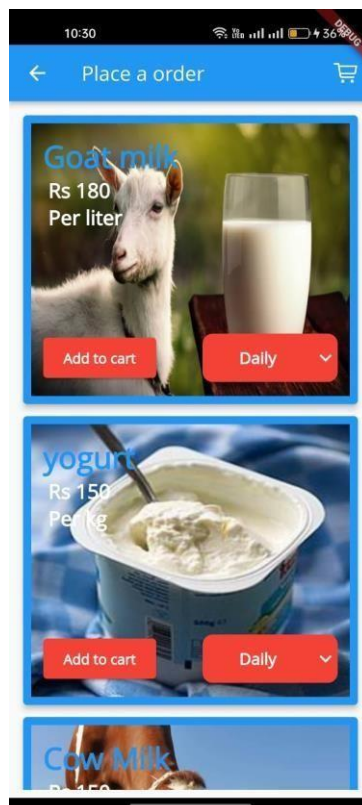
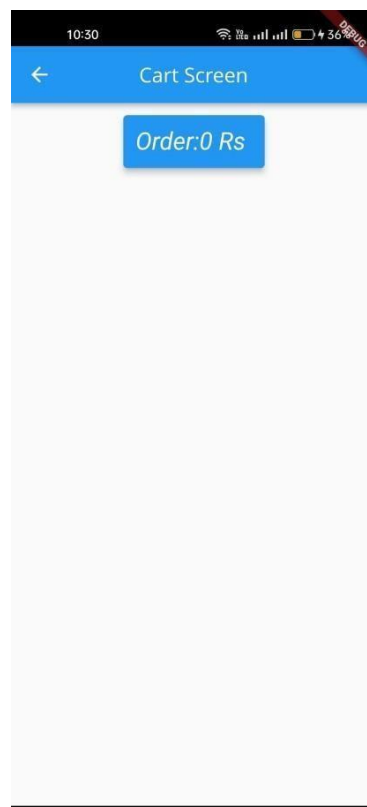


Figure 5-7: Place Order Screen



**Figure 5-8: Order History**



**Figure 5-9: Cart Screen**



Figure 5-10: Order Screen (Urdu)

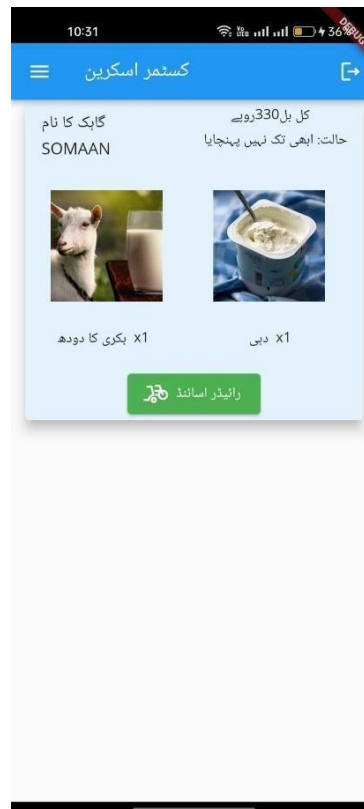


Figure 5-11 Customer Screen (Urdu)

10:33

← Update

### Edit User

user

somi@gmail.com

somaan

Shipping Address

h#67St#5,Multan Road ,Lahore

Update

Figure 5-12 Edit User

10:33

Admin Screen

### Admin Screen

Email

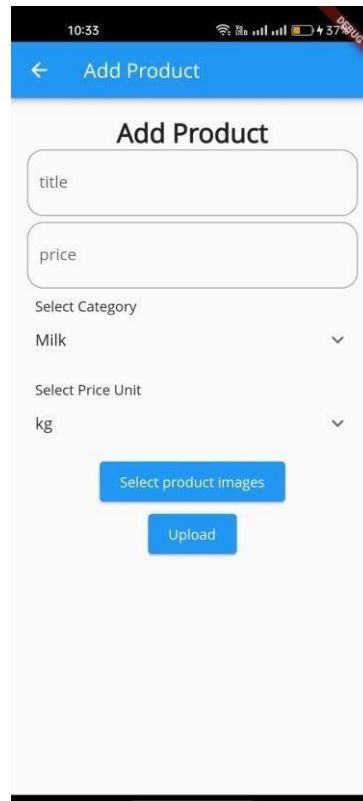
somi@gmail.com

Get User Data Edit User Delete User

#### User Data

Name : somaan  
Email : somi@gmail.com  
UID : VD5L6oZzuZefbgBICZeSrD6fu0v2  
Role : user  
Role : h#67St#5,Multan Road ,Lahore

Figure 5-13 Admin Home Screen



10:33

← Add Product

### Add Product

title

price

Select Category

Milk

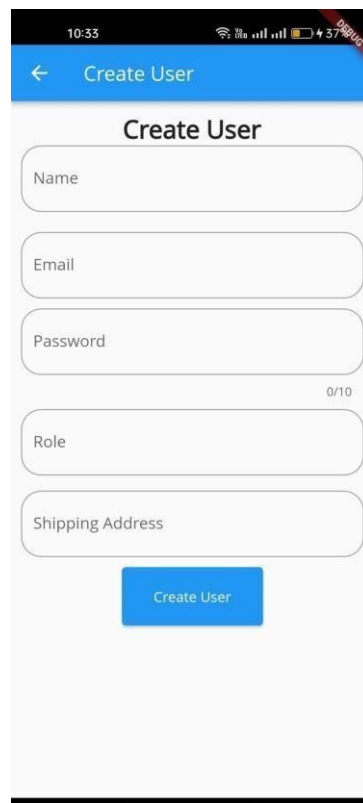
Select Price Unit

kg

Select product images

Upload

**Figure 5-14: Add Product**



10:33

← Create User

### Create User

Name

Email

Password

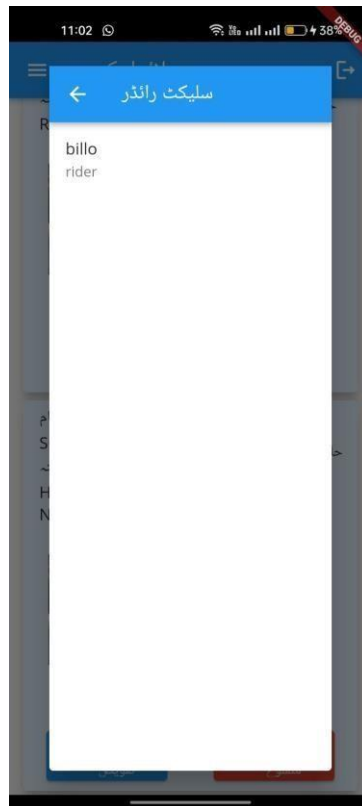
0/10

Role

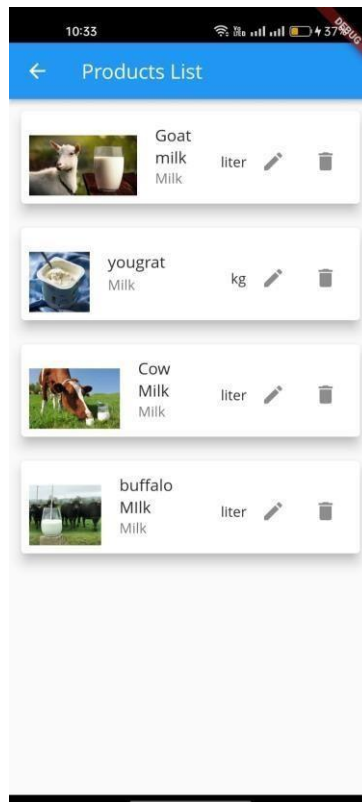
Shipping Address

Create User

**Figure 5-15: Create User**



**Figure 5-16: Select Rider**



**Figure 5-17: Product List**





Figure 5-18: supplier screen (Urdu)

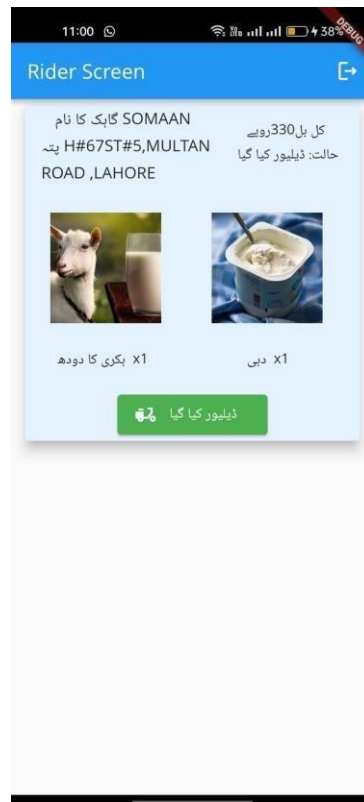


Figure 5-19 Rider Screen

## **CHAPTER 6**

### **Conclusion and Recommendation**

#### **6.1 Conclusion**

The main purpose of our project is to connect all stakeholders through one platform. The benefits of our project will impact all participants. Customers don't need any more to stand in the queue. And riders are getting more job opportunists along with vendors providing milk service. The main idea of our project is to connect the unpacked milk market with an e-commerce platform for the sake of automation and an increase in revenue. The secondary objective of our project is to provide enough benefits for the stakeholders who sell their products ranging from Rs 120 - Rs 150 per litter based on the order quantity. Those vendors are not having proper storage facilities, or they are not interested in selling their products at higher prices, so we found a way by using e-commerce technology to make them reachable to customers all over Pakistan.

#### **6.2 Recommendation**

In future it can be expended to other multiple platforms like IOS and Windows. For now the user should have Android phone to connect with the server.

## REFERENCES

- [1] G. Nadler, "Systems methodology and design," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 6, pp. 685-697, Nov.-Dec. 1985, doi: 10.1109/TSMC.1985.6313452.
- [2] Boehm, "Software Engineering," in *IEEE Transactions on Computers*, vol. C-25, no. 12, pp. 1226-1241, Dec. 1976, doi: 10.1109/TC.1976.1674590.
- [3] S. Sengupta and S. Bhattacharya, "Formalization of UML use case diagram-a Z notation based approach," 2006 International Conference on Computing & Informatics, 2006, pp. 1-6, doi: 10.1109/ICOICI.2006.5276507.
- [4] J. Yang, "Analyzing UML Sequence Diagrams with UTP," 2009 Fourth International Conference on Frontier of Computer Science and Technology, 2009, pp. 417-423, doi: 10.1109/FCST.2009.73.
- [5] Il-Yeol Song and K. Froehlich, "Entity-relationship modeling," in *IEEE Potentials*, vol. 13, no. 5, pp. 29-34, Dec. 1994-Jan. 1995, doi: 10.1109/45.464652.

## fyp umer

## ORIGINALITY REPORT

**16%**

SIMILARITY INDEX

**24%**

INTERNET SOURCES

**1%**

PUBLICATIONS

**21%**

STUDENT PAPERS

## PRIMARY SOURCES

<b>1</b>	<b>www.coursehero.com</b> Internet Source	<b>5%</b>
<b>2</b>	<b>launchdarkly.com</b> Internet Source	<b>4%</b>
<b>3</b>	<b>Submitted to Higher Education Commission Pakistan</b> Student Paper	<b>3%</b>
<b>4</b>	<b>dspace.daffodilvarsity.edu.bd:8080</b> Internet Source	<b>2%</b>
<b>5</b>	<b>Submitted to Universiti Teknologi MARA</b> Student Paper	<b>2%</b>
<b>6</b>	<b>Submitted to University of East London</b> Student Paper	<b>2%</b>
<b>7</b>	<b>www.lucidchart.com</b> Internet Source	<b>1%</b>
<b>8</b>	<b>Submitted to Middlesex University</b> Student Paper	<b>1%</b>
<b>9</b>	<b>startupstash.com</b> Internet Source	<b>1%</b>

10	<a href="http://educationdocbox.com">educationdocbox.com</a> Internet Source	1 %
11	<a href="http://answeregy.com">answeregy.com</a> Internet Source	1 %
12	Submitted to University of Bedfordshire Student Paper	<1 %
13	<a href="http://webapps.itc.utwente.nl">webapps.itc.utwente.nl</a> Internet Source	<1 %
14	Submitted to Kuala Lumpur Infrastructure University College Student Paper	<1 %
15	<a href="http://pujc.edu.pk">pujc.edu.pk</a> Internet Source	<1 %
16	<a href="http://literatureessaysamples.com">literatureessaysamples.com</a> Internet Source	<1 %
17	<a href="http://www.ijcttjournal.org">www.ijcttjournal.org</a> Internet Source	<1 %
18	<a href="http://yessilopez.com">yessilopez.com</a> Internet Source	<1 %
19	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1 %
20	<a href="http://nanopdf.com">nanopdf.com</a> Internet Source	<1 %
21	Submitted to University of Adelaide Student Paper	<1 %

---

		<1 %
22	Submitted to University of Bahrain Student Paper	<1 %
23	dspace.aus.edu:8443 Internet Source	<1 %
24	Submitted to University of Portsmouth Student Paper	<1 %
25	www.appsbee.com Internet Source	<1 %
26	Submitted to Federation University Student Paper	<1 %
27	Submitted to Institute of Research & Postgraduate Studies, Universiti Kuala Lumpur Student Paper	<1 %
28	Submitted to Segi University College Student Paper	<1 %
29	eie.uonbi.ac.ke Internet Source	<1 %

---