

eShopify.

SHOPPING APPLICATION

SOFTWARE ENGINEERING PROJECT REPORT

CONTENTS

1. Introduction	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Product Scope	1
2. Overall Description	2
2.1 Product perspective	2
2.2 Product Functions	2
2.3 User classes and Characteristics	2
2.4 Operating Environment	2
2.5 Design and Implementation Constraints	3
2.6 User Documentation	3
2.7 Assumption and Dependencies	3
3. External Interface Requirements	4
3.1 User Interfaces	4
3.2 Hardware Interface	4
3.3 Software Interface	4
4. System Features	5
4.1 Login and Registration	5
4.2 View Products	6
4.3 Add or remove products from cart	7
4.4 View cart	7
4.5 Buying the product from cart	8
4.6 Feedback	8
5. Other Non-Functional Requirements	9
5.1 Performance Requirements	9
5.2 Safety Requirements	9
5.3 Security Requirements	9
5.4 Software Quality Attributes	9
5.5 Business Rules	9
6. Risk Management	10
7. Function Point and Cost Estimation	11
7.1 Function Point Metric	11
7.2 Cost Estimation	13
8. Gantt Chart	14

9. Testing	15
9.1 White Box Method	15
9.2 Flow Graph	19
9.3 Cyclomatic Complexity	19
9.4 Test Cases	20
10. Analysis Models	21
10.1 E-R Diagram	21
10.2 Data Flow Diagram	22
10.3 Data Dictionary	25
10.4 Use Case Diagram	27
10.5 Sequence Diagram - Customer	28
10.6 Sequence Diagram - App Manager	29
10.7 Activity Diagram - Customer	30
10.8 Activity Diagram - App Manager	31
10.9 Navigation Diagram - Customer	32
10.10 Navigation Diagram - App Manager	33
10.11 Swimlane Diagram	34
Appendix Glossary	35
References	36

1.Introduction

1.1 Purpose

The purpose of this document is to give a detailed description of all the requirements for the Shopping Application Software. This SRS will provide a detailed description of all the functionalities of the application. It will also explain system constraints, interface and interactions with other external applications.

1.2 Document Conventions

This document uses the following conventions:

- User: Refers to all the people who will use the web application.
- App Manager: Refers to people who will update the application on weekly basis or as a new product gets launched.

These are the conventions that were used for editing the document

- The document follows some terminologies of IEEE Software Requirements Specification format.
- This document uses the Alegreya Sans Bold 20 for headings, Alegreya Sans Regular 18 for sub-headings and Arial Unicode 13 for inner body.

1.3 Intended Audience and Reading Suggestions

This document is intended for software developers who want to implement new features and testers who can run and test the software.

The remainder of this document includes three chapters:

- The second one provides an overview of the app functionality and app interaction with other systems. This chapter also introduces different types of stakeholders and their interaction with the system. Further, the chapter also mentions the system constraints and assumptions about the product.
- The third chapter provides a description of the different system interfaces.
- The fourth chapter deals with the prioritization of the requirements. It includes a motivation for the chosen prioritization methods and discusses why other alternatives were not chosen.

1.4 Product Scope

- The purpose of this Shopping Application is to find electronics gadgets of all brands under one application and to create a convenient and easy-to-use application for tech enthusiast to find their required gadget under the best price available.
- The goal is to merge all the electronics gadgets of all the brands in one application rather than finding them on their respective brand websites and get confused with the best price available.

2. Overall Description:

2.1 Product Perspective

The Shopping application is a web/mobile application that consists of various Electronic Gadgets that are in need of today's Era and their accessories. This application can be used by any user in the world who likes to buy gadgets more frequently. It can be used by people of any age group to buy any electronic product of any brand at the lowest price possible with assured after service. Since this is a data Centre product we need to store the data somewhere. For that DB will be used. The DB can be queried to store information about the user and stock that will be displayed in the app.

2.2 Product Functions

Functions for Users:

- Can have a thorough look for the product they need and wish list the product and compare product with other various brands.
- Can directly buy the product from the app by adding the product to cart and thereafter making the payment on the payment gateway.
- Can pay using UPI, Net Banking, cards as well Cash on Delivery.
- Can give feedback on purchase and provide suggestions for better improvement of app.
- Can view all the purchases made by them.

Functions for App Manager:

- Can add a new product as soon as a product gets launched.
- Can remove the product once its production terminates.
- Can update the price of items weekly as the best price that can be provided.

2.3 User Classes and Characteristics

There are two types of users that will interact with the system - customer and App manager.

- Customers will use the app to buy the electronic product they are willing to buy. S/He will be able to have a brief description of the product along with cost comparisons of product available on other apps. Customers can place the order directly from the app and make the payment via above stated methods and can cancel the order any time in case they change their mind. They can review the product on a scale of 1-5 after using it for few days as well.
- App manager can add, remove or update the product according to demand and availability of Stock.

2.4 Operating Environment

- Operating Systems: Windows, MacOS, Android, iOS.
- Web Browsers Google Chrome, Safari, Mozilla Firefox, Microsoft.

2.5 Design and Implementation Constraints

The web application will be constrained by the capacity of the DB. The DB may in some cases be forced to queue incoming requests and therefore increase the time it takes to fetch data.

2.6 User Documentation

The user documentation consists of the basic workflow of buying a product. It also consists of a detailed overview of how the payment system works using payment gateway where customer has the option to pay with UPI, Card, etc.

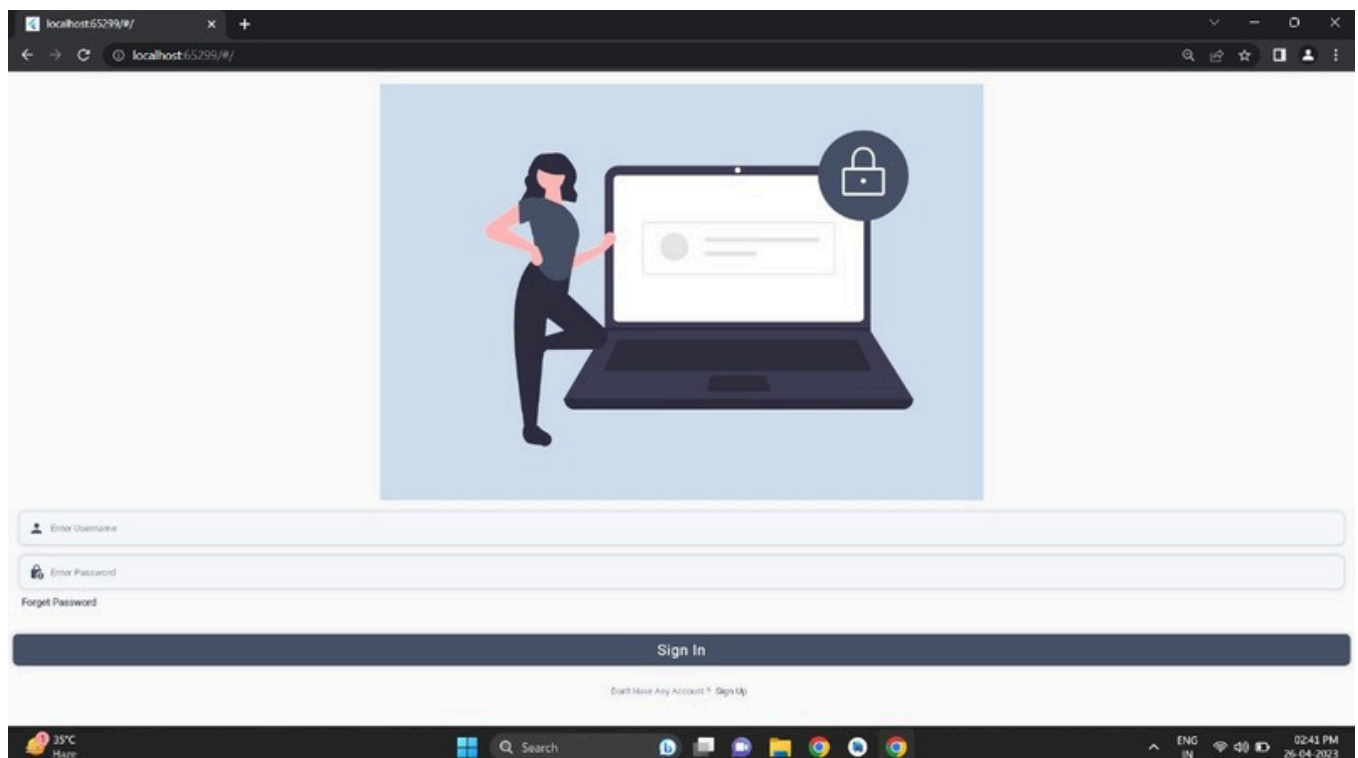
2.7 Assumptions and Dependencies

- The booking of a product is assumed to be completed only after receiving the payment if opted for online payment. If customer opts for cash on delivery he must confirm it by entering a captcha followed by a final confirmation.
- The customer needs to be logged in to the app in order to buy product.

3. External Interface Requirements

3.1 User Interface

- The customer can sign in or sign up in the application for conducting the search of various products available on the app or the product they are looking for.
- The App Manager can login through the web portal in which he/she can alter the products i.e., can add a new product, remove the old product or update the price of product if that gets reduced by the company.



3.2 Hardware Interface

- Since, the application runs on the internet, all the hardware shall require to connect internet will be hardware interface for the system. Example, Wi-Fi, Mobile data if using on mobile and WAN-LAN Ethernet if using on web.
- The hardware connection to the database server is managed by the underlying operating system of the computer and the web server
- As it is intranet communication, it will be through the TCP/IP protocol suite.

3.3 Software Interface

- Flutter - Dart framework will be used to develop the application which is a fast, secure and scalable framework for developing application to run on every platform and OS.
- SQLite DB will be used for this project as it is a robust and easy to use database and can be easily integrated with the framework.

4. System Features

This section describes the various features and functional requirements of the Shoooping Application

4.1 Login and Registration

4.1.1 Description and Priority

- This application contains a login page to authenticate the customer who will login to the application. The customer need to sign up for the first time or sign in if s/he gets logged out using the registration page.
- Priority for this feature is high as it prevents unauthorized access to the system and is required to purchase a product from the app.

4.1.2 Stimulus Response Sequences

- The customer needs to enter his/her login id and password which was used to register the account to login whereas the app manager can directly update the app from backend using the same code and IDE that was used during the development of the application. On entering valid credentials, the customer will be directed to their home page. A message will be shown to the customer if login was unsuccessful.
- Valid details must be entered for all the mandatory fields to register an account. The customer will then be directed to the login page. A message will be shown to the customer if registration was unsuccessful.
- If customer re-logs using the same id and forgets her/his password then s/he can reset the password using the Forget password option.

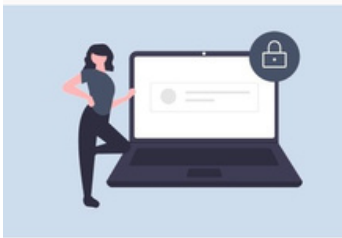
4.1.3 Functional Requirements

REQ-1: Every customer need to be associated with an account.

REQ-2: One account cannot be associated with multiple customers.

REQ-3: All mandatory fields in the login and registration pages should be filled.

REQ-4: The credentials entered should be verified with the DB and user should be allowed to proceed to next page only on successful authentication.



alex_123

[Forget Password](#)

Sign In

Don't Have Any Account ? [Sign Up](#)

4.2 View Products

4.2.1 Description and Priority

- Interface to display information about various available products and a search bar to search for the desired product so that the user can have a view of their product and can look for other available options on the same page.
- Priority for this feature is High as this is the main feature in the system whereafter user can proceed to buy the product.

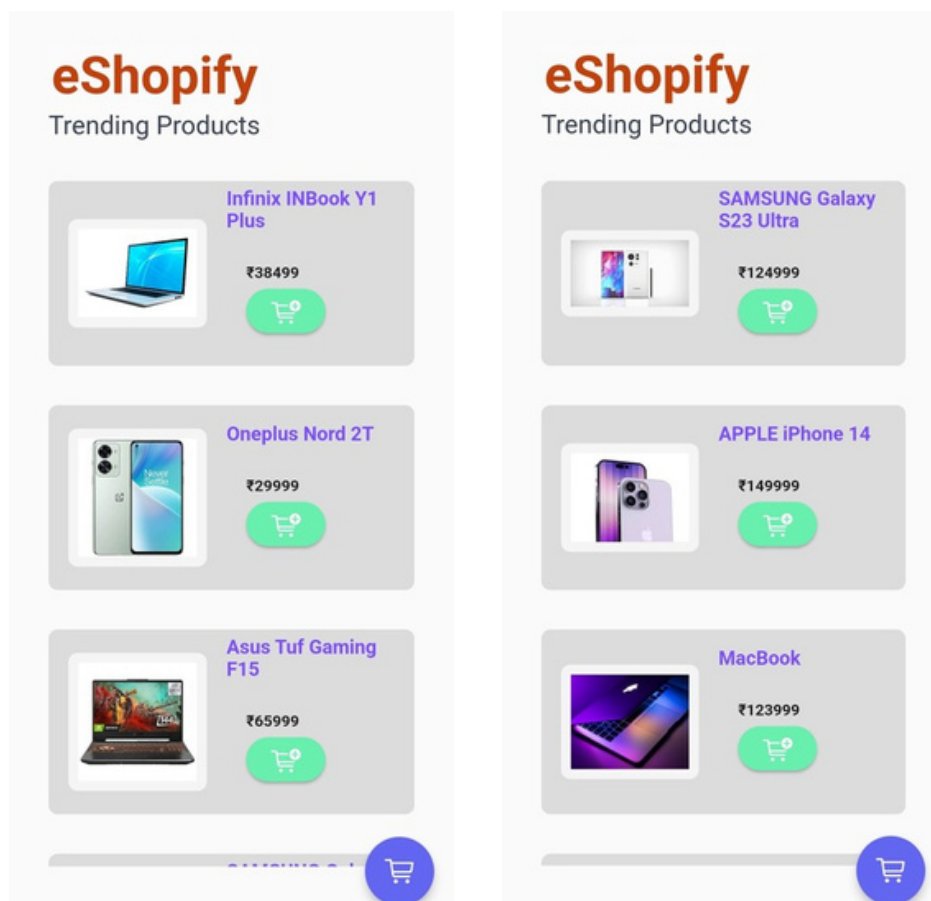
4.2.2 Stimulus/Response Sequences

- The product information for a particular product will be fetched from the DB and will be available to view in the next screen when user selects that product.
- On viewing the brief details of the product customer has the option to add the product to cart before buying it.
- Customer can compare his/her products with other similar options that are available on the app

4.2.3 Functional Requirements

REQ-1: When customer views a product then s/he will be able to read the reviews and feedbacks from the customers who have already bought that product.

REQ-2: If customer still feels to further enquire about the product then a ASK question box should be available in which either other customer or app manager can answer.



4.3 Add or remove product to cart

4.3.1 Description and Priority

- The app provides a facility to add product to cart and later can remove the product from cart if not required anymore.
- Priority for this feature is medium as customer may or may not necessarily buy the product

4.3.2 Stimulus Response Sequences

- The customer adds a product to cart for buying it and can remove the product if his/her mind changes to not buy the product

4.3.3 Functional Requirements

REQ-1: Only products that are in stock can be added to cart otherwise it will notify the customer that product will be available soon.

REQ-2: The addition and removal of products in cart should be reflected in the DB too.

4.4 View cart

4.4.1 Description and Priority

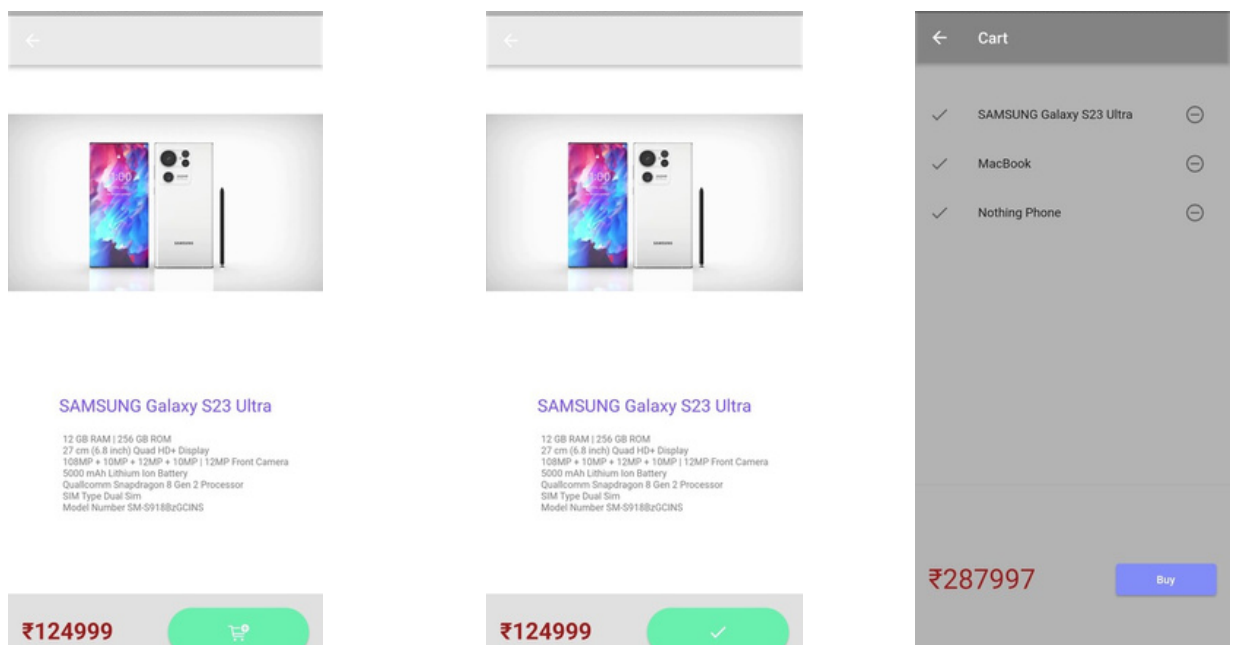
- The customers has the option of viewing the products they have added to the cart.
- Priority for this feature is medium as this feature may not be used by customers too often or they may directly checkout to buy all the items present in the cart.

4.4.2 Stimulus/Response Sequences

- The products and their final price are calculated in the cart after fetching the data from the DB and after applying all the necessary taxes and charges.

4.4.3 Functional Requirements

REQ-1: The customer should only be able to view his/her choice of products that are added to the cart.



4.5 Buying the product from cart

4.5.1 Description and Priority

- The customer checks the items in the cart and after making all the necessary decisions for purchasing the items s/he checkouts and finally goes to confirmation page where all the details are available and then proceeds to payment gateway.
- Priority for this feature is high as this is the only way to order or buy the products from app.

4.5.2 Stimulus/Response Sequences

- When the customer presses the final confirmation button s/he is redirected to the payment gateway if online payment option is chosen and they can pay the amount and after successful transaction their order gets placed.
- If they chose the option to pay on delivery then they are redirected to a page where they have to do a captcha verification and OTP verification and after successful completion of this step their order gets placed.

4.5.3 Functional Requirements

REQ-1: The customer should only pay from the trusted apps and banks regulated by RBI.

REQ-2: The customer should have an active number over which OTP verification can be done before finally placing the order

4.6 Feedback

4.6.1 Description and Priority

- The customer has the option to rate and review the products and may ask the app manager for the availability of that product s/he looking for.
- Priority for this feature is medium as not everyone want to rate and review the product and the feedback may be beneficial to some only.

4.6.2 Stimulus/Response Sequences

- The customer selects the product s/he have purchased and rates it on a scale of 1-5 where 1 denotes poor and 5 denotes Excellent.
- Customers can upload photos and videos of the product too and can provide the feedback based on that photos also.

4.6.3 Functional Requirements

REQ-1: The customer should only be able to give feedback only on that products s/he have purchased.

REQ-2: The average rating of a product must be displayed to all other customers while they are viewing any product.

REQ-3: A separate query box must be present where user can demand the availability of any product or can ask in detail about any product.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The eShopify Application requires a good internet connection, with a modern android version such as 11+, iOS version 12+ and browser such as Google Chrome, Mozilla Firefox, IEB or Microsoft Edge.
- The performance depends on internet speed, availability of product, demand of the product and number of database queries.

5.2 Safety Requirements

- There is a possibility that the user's login credentials might get compromised. So it is important to have a strong password, which needs to be enforced by the system.

5.3 Security Requirements

- To prevent unauthorized access to data, only authenticated customers can make purchase of products and only App manager who has the access of the app can make changes in the product.
- Proper form validation will be done throughout all pages to prevent erroneous data from being entered into DB.
- CSRF verification is done for all forms to prevent man-in-the-middle attacks
- The system shall use secure sockets in all transactions that include any confidential customer information.

5.4 Software Quality Attributes

- **Correctness information:** The product information displayed should be correct in terms of price and specifications given by the manufacturing company
- **Maintainability:** The application should be easy to extend and maintain
- **Portability:** The software should be easily portable and be accessible on all kinds of devices
- **Usability:** The interface should be user-friendly.

5.5 Business Rules

- **Customer:** Login and account registration, viewing products, adding product to cart, viewing cart, give feedback and purchasing the product.
- **App Manager:** Login and add or remove products.

6. Risk Management

SNo.	RISK	CATEGORY	PROBABILITY	IMPACT	RMMM PLAN
1	Some team members leave the project development in between.	Technical Risk	30%	2	Use backup Staffs who knows about this project its function and motive.
2	Delivery deadline preponed.	Project Risk	40%	1	Team have to extend the team size to complete the task on scheduled time.
3	Losing of all the project data. This may be caused by a hard disk being wiped out by a virus, hard disk failure etc.	Project Risk	20%	2	Carry out necessary backup of database data, source code and documentation
4	Team dissension/lack of cohesion.	Project Risk	15%	3	To build some common guidelines to be followed till the completion of project amongst team.

7.Function Point Metrics and Cost Estimation

7.1 Function Point

We have categorized external inputs on sub fields

- If <3-Simple
 <4-Medium
 >-4-Complex

For external output

- If 1 detail - Simple
 2 detail - Medium
 >2 detail - Complex

We take all logical fields as well as external enquiry as simple

- **External Input**
 1. Login credential -> Name, Password (medium)
 2. Signup credentials -> I'd, Password, Name, Phone number(complex)
 3. App Manager -> Name, Password, Key (medium)
 4. Product details add -> (medium)
 5. Product remove details -> (medium)
 6. Input query -> product search, cart (simple)
 7. Request -> Details (simple)
 8. Complaint -> Details (simple)
- **External Output**
 1. Result -> Product details + Price (medium)
 2. Request Ad -> Request at admin (simple)
 3. Complain Ad -> Complain at admin (simple)
- **Logical internal files (simple)**
 1. User login
 2. User signup
 3. Product info
 4. Reg. new product
 5. Reg complains
 6. Final Price of product calculated
 7. App Manager Login

8. Product remove
9. Product add
10. Reg. handling.
11. Complaint handling
12. Assign key

■ **External interface files**

0

■ **External enquiries**

1. Product details (result)
2. Query (user can write)
3. Request (user can write)
4. Feedback(admin can read)
5. Request (admin can read)

	Simple	Medium	Complex
External I/P	3	4	1
External O/P	2	1	0
Logical Interface	2	0	0
External Interface	0	0	0
External Enquiry	5	0	0

$$\begin{aligned}
 \text{UFP} &= 3 \times 3 + 4 \times 4 + 1 \times 6 + 2 \times 4 + 1 \times 5 + 12 \times 7 + 5 \times 3 \\
 &= 9 + 16 + 6 + 8 + 5 + 84 + 15 \\
 &= 143
 \end{aligned}$$

$$\begin{aligned}
 \text{CAF} &= 0.65 + 0.01 \times 14 \times 3 \\
 &= 1.07
 \end{aligned}$$

$$\begin{aligned}
 \text{FP} &= \text{UFP} \times \text{CAF} \\
 &= 143 \times 1.07 \\
 &= 153.01 \\
 &= 153 \text{ (approx.)}
 \end{aligned}$$

7.2 Cost Estimation of efforts

Screens - 6

Reports - 7

3GL - 1




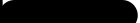

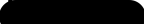


$$\begin{aligned}\text{Object Point} &= 6 \times 2 + 7 \times 5 + 1 \times 10 \\ &= 12 + 35 + 10 \\ &= 57\end{aligned}$$

$$\begin{aligned}\text{NOP Object Point} &\times (1 - \text{reuse}) \\ &= 57 \times (1 + 0) \\ &= 57\end{aligned}$$

$$\begin{aligned}\text{Efforts} &= \text{NOP} / \text{PROD} \\ &= 57 / 13 = 4.38\end{aligned}$$

We assume nominal developer experience

8. Gantt Chart

SN0.	TASK DESCRIPTION	START	FINISH	WEEK 1	WEEK 2	WEEK 3	WEEK 4
1	Establish Project	02/03/2023	06/03/2023				
2	Establish Customer Requirement	07/03/2023	09/03/2023				
3	Produce Software Specification Document	10/03/2023	12/03/2023				
4	Write test plans	13/03/2023	18/03/2023				
5	Write Codes	19/03/2023	22/03/2023				
6	Developer Testing	23/03/2023	29/03/2023				
7	System Testing	30/03/2023	04/04/2023				
8	Write Customer Documentation	05/04/2023	08/04/2023				

Project start date : 2 March 2023

9. Testing

9.1 White box method

Structured testing is a white box method for designing test cases. The method analyzes the control flow graph of a program to find a set of linearly independent paths of execution. The method normally uses cyclomatic complexity to determine the number of linearly independent paths and then generates test cases for each path thus obtained. Basis path testing guarantees complete branch coverage (all CFG edges), but achieves that without covering all possible CFG paths the latter is usually too costly. Basis path testing has been widely used and studied.

To measure the logical complexity of our software we consider the following procedure:

```
class LoginPage extends StatelessWidget {
  const LoginPage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SingleChildScrollView(
        child: SafeArea(
          child: Column(
            children: [
              Padding(padding: EdgeInsets.symmetric(vertical:20),
                child: Image.asset("assets/images/loginimg.png"),
              ),
              Container(
                margin: EdgeInsets.symmetric(horizontal: 20),
                padding: EdgeInsets.symmetric(horizontal:15 ),
                height: 55,
                decoration: BoxDecoration(
                  color: Color(0xFFF5F9FD),
                  borderRadius: BorderRadius.circular(10),
                  boxShadow: [
                    BoxShadow(
                      color: Color(0xFF475269).withOpacity(0.3),
                      blurRadius: 5,
                      spreadRadius: 1
                    )
                  ]
                ),
              child: Row(children: [
                Icon(
                  Icons.person,
                  size: 27,
                  color: Color(0xFF475269),
                ),
```

```

    SizedBox(width: 10,),
    Container(
      width: 250,
      child: TextFormField(
        decoration: InputDecoration(
          border: InputBorder.none,
          hintText: "Enter Username"
        ),
      ),
    ),
  ],),
),
SizedBox(height: 20,),
Container(
  margin: EdgeInsets.symmetric(horizontal: 20),
  padding: EdgeInsets.symmetric(horizontal: 15 ),
  height: 55,
  decoration: BoxDecoration(
    color: Color(0xFFFF5F9FD),
    borderRadius: BorderRadius.circular(10),
    boxShadow: [
      BoxShadow(
        color: Color(0xFF475269).withOpacity(0.3),
        blurRadius: 5,
        spreadRadius: 1
      )
    ]
  ),
),
child: Row(children: [
  Icon(
    Icons.lock_person,
    size: 27,
    color: Color(0xFF475269),
  ),
  SizedBox(width: 10,),
  Container(
    width: 250,
    child: TextFormField(
      obscureText: true,
      decoration: InputDecoration(
        border: InputBorder.none,
        hintText: "Enter Password"
      ),
    ),
  ),
],),
),
SizedBox(height: 10,),
Container(
  margin: EdgeInsets.only(left: 15),
  alignment: Alignment.centerLeft,
  child: TextButton(
    onPressed: (){
  },

```

```

        child: Text(
          "Forget Password",
          style: TextStyle(
            color: Color(0xFF475269),
            fontSize: 17,
            fontWeight: FontWeight.w600,
          ),
        ),

      ),
    ),
    SizedBox(height: 40,),
    InkWell(
      onTap: (){
        Navigator.pushNamed(context, MyRoutes.homeRoute);
      },
      child: Container(
        alignment: Alignment.center,
        margin: EdgeInsets.symmetric(horizontal: 8.0),
        padding: EdgeInsets.symmetric(horizontal: 15),
        height: 54,
        width: double.infinity,
        decoration: BoxDecoration(
          color: Color(0xFF475269),
          borderRadius: BorderRadius.circular(10),
          boxShadow:[
            BoxShadow(
              color: Color(0xFF475269).withOpacity(0.3),
              blurRadius: 5,
              spreadRadius: 1,
            )
          ]
        ),
      ),
      child: Text(
        "Sign In",
        style: TextStyle(
          fontSize: 25,
          fontWeight: FontWeight.w500,
          color: Colors.white,
          letterSpacing: 1,
        ),
      ),
    ),
    ),
    ),
    ),
    ),
    SizedBox(height: 30,),

    Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text("Don't Have Any Account ?",
          style: TextStyle(
            color: Color(0xFF475269).withOpacity(0.8),
            fontSize: 15,
          ),
        ),
      ],
    ),
  ),
)

```

```

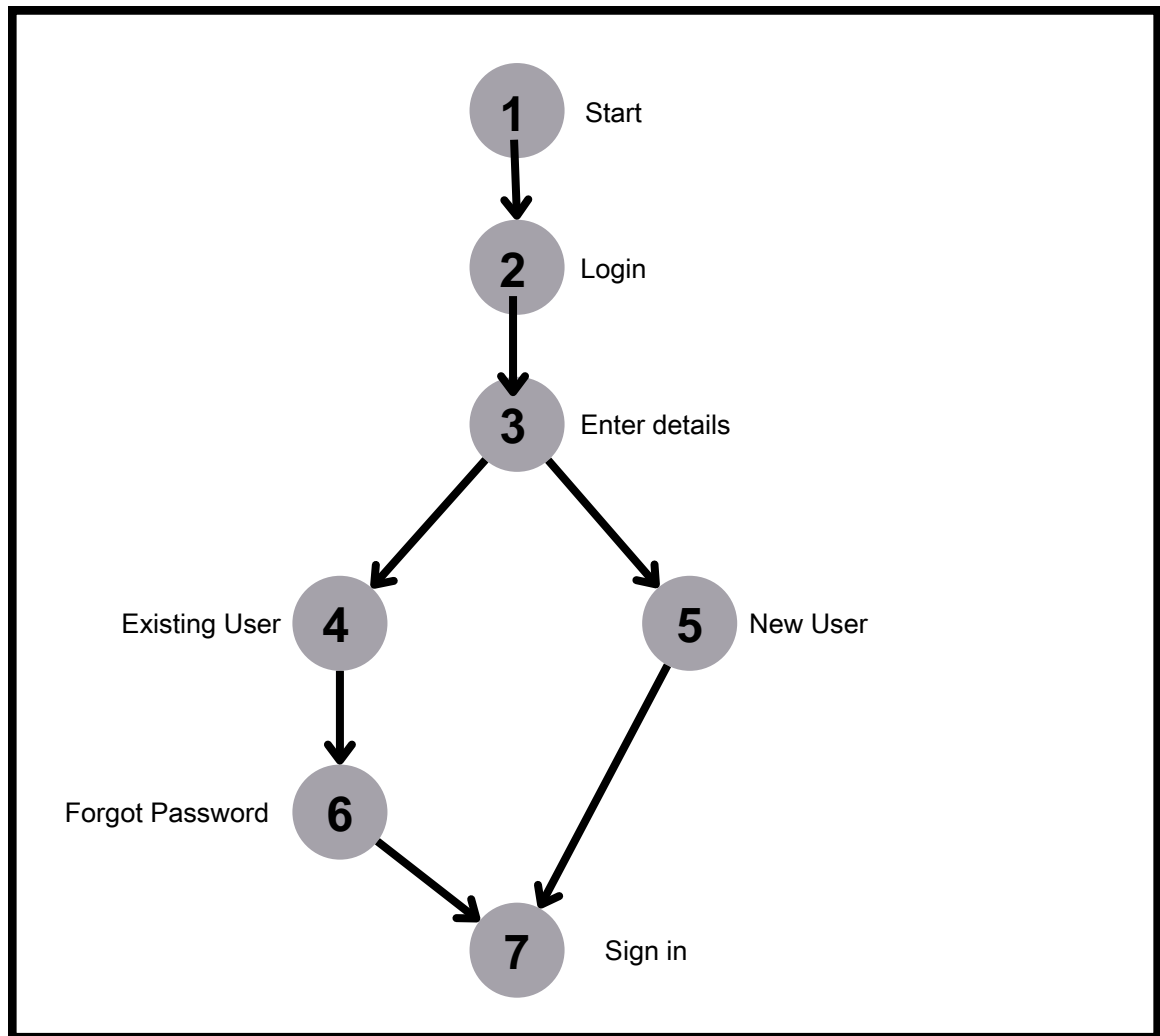
    TextButton(
      onPressed: (){

        },
        child: Text("Sign Up",
          style: TextStyle(
            color: Color(0xFF475269),
            fontSize: 16,
            fontWeight: FontWeight.w500,
          ),
        ),
      ),
    ),
  ),
  1,)

1,
),
),
),
),
);
}
}

```

9.2 Flow Graph of the above procedure:



9.3 Cyclomatic Complexity

Cyclomatic complexity $V(G)$ for a flow graph G is defined as

$$V(G) = E - N + 2$$

where E = No. of Edges

N = No. of Vertices

For the above graph $V(G) = 7 - 7 + 2 = 2$.

No of independent path is: 2

Path 1: 1-2-3-5-7

Path 2: 1-2-3-4-6-7

9.4 Test Cases:

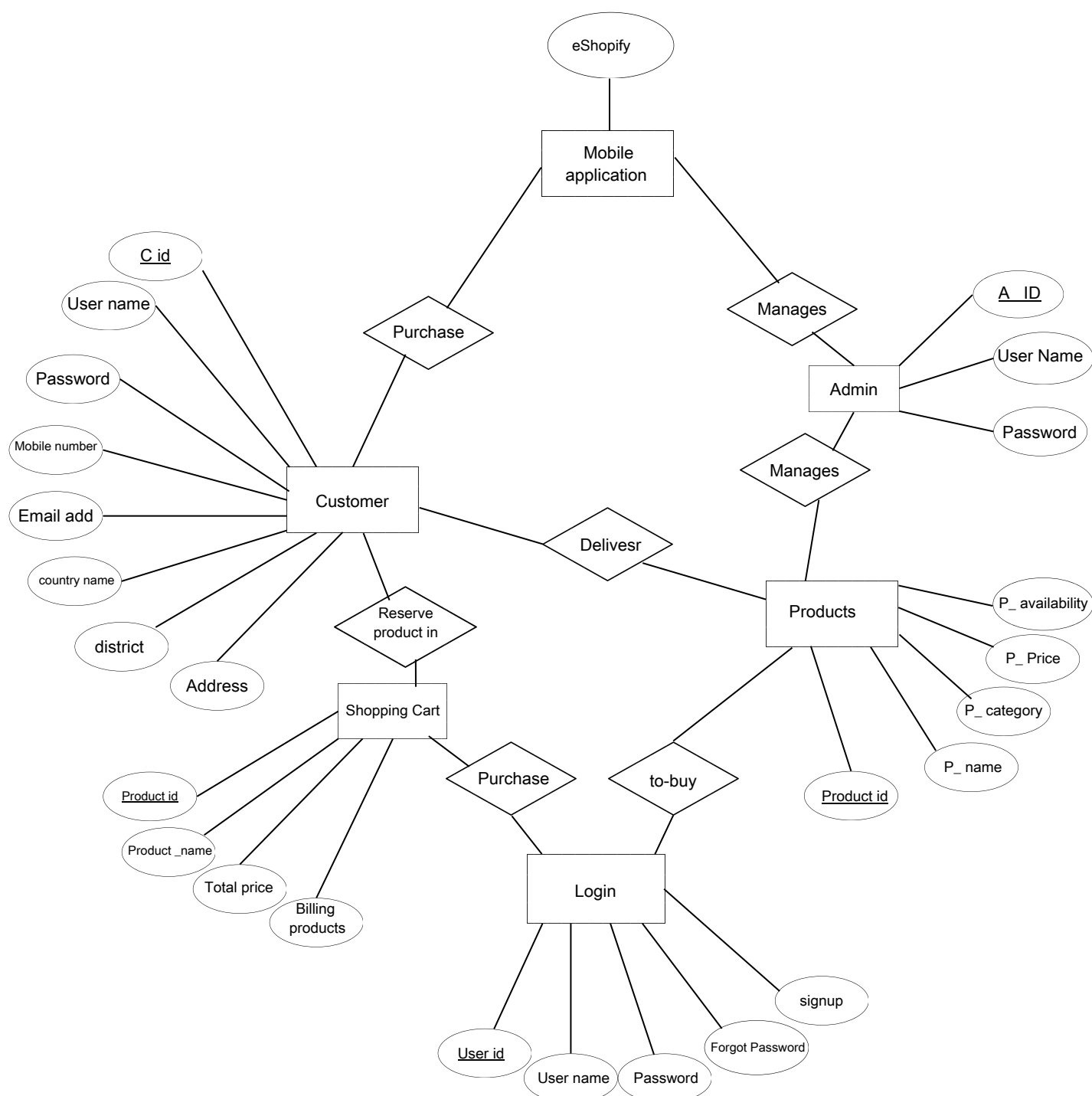
Test Scenario	Requirements	Test Cases	Test Data	Result
Check Login Functionality	✓ All correct combinations entered in login/password field will let user in.	Check response on entering valid username and password.	Username: alex_1234 Password: Temp@12345	Login Successful
	✓ All other combinations will be rejected.	Check response on entering invalid username and password.	Username: ALEX Password: Temp	Username is invalid
	✓ Login includes minimum 8 letters and 4 digits/special characters or both (except space) ✓ Login is not case sensitive. ✓ Password cannot be less than any 8 characters (except space character) ✓ Password is not case sensitive ✓ "Login" button is disabled unless both fields are typed in	Check response when username and password field is Empty and login button is pressed	Username: Password:	Username and password invalid

10: Analysis Models

10.1: E- R Diagram

An ER diagram can be used to design logical database schemas. An ER model is a high-level description of the data and the relationships among the data, rather than how data is stored. It focuses on identifying the entities and the relationship among the entities.

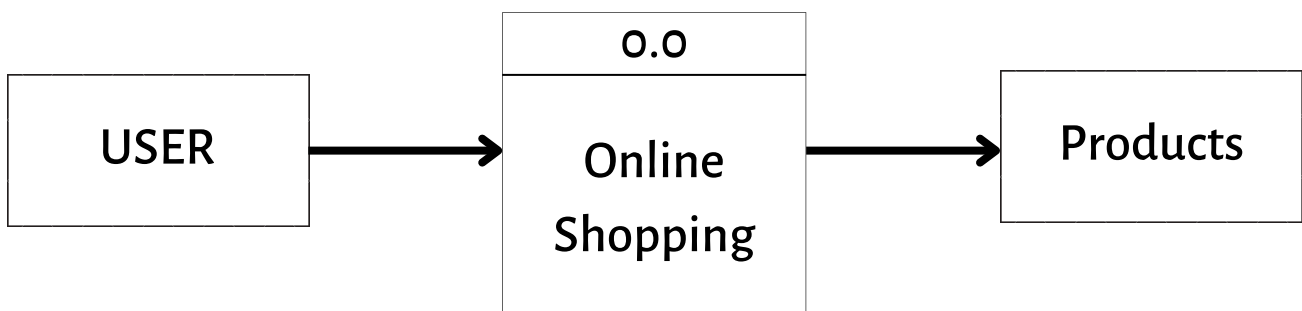
In the ER diagram of Online Shopping the relationship between the customer to product is one to many as same customer can buy multiple product, also for category to product is one to many as for a single category have different product. Also here mentioned the relationship between the tables, mention the primary key of the table. Here another important part is that its mention all the entities of the table.



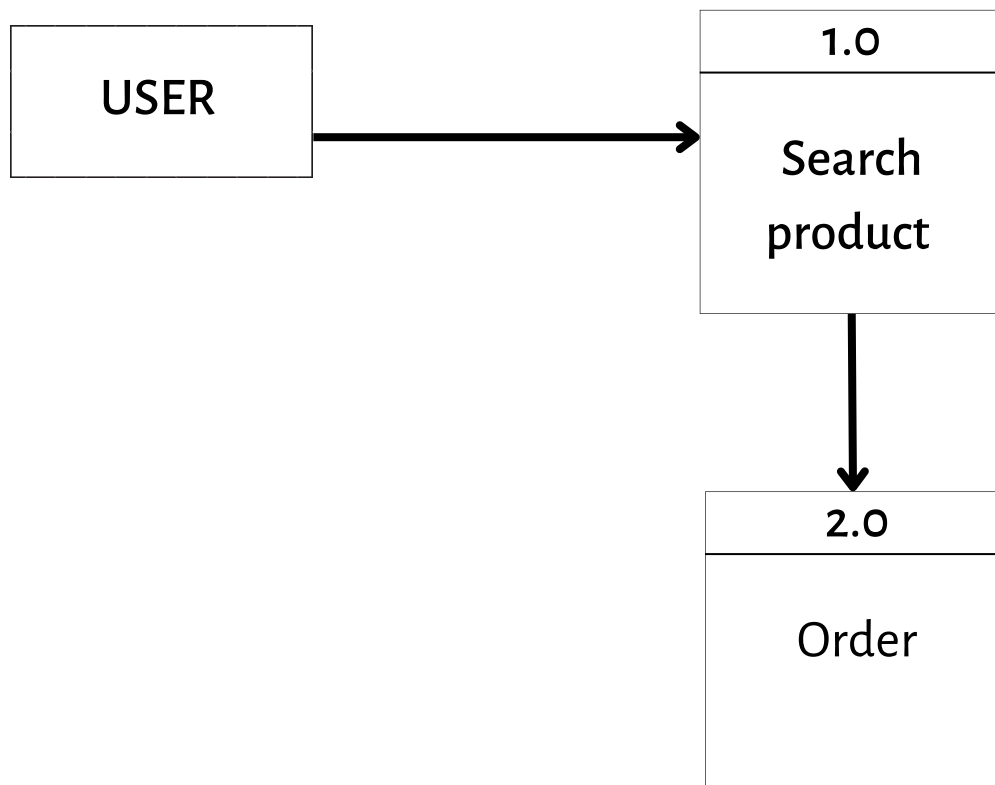
10.2: DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated

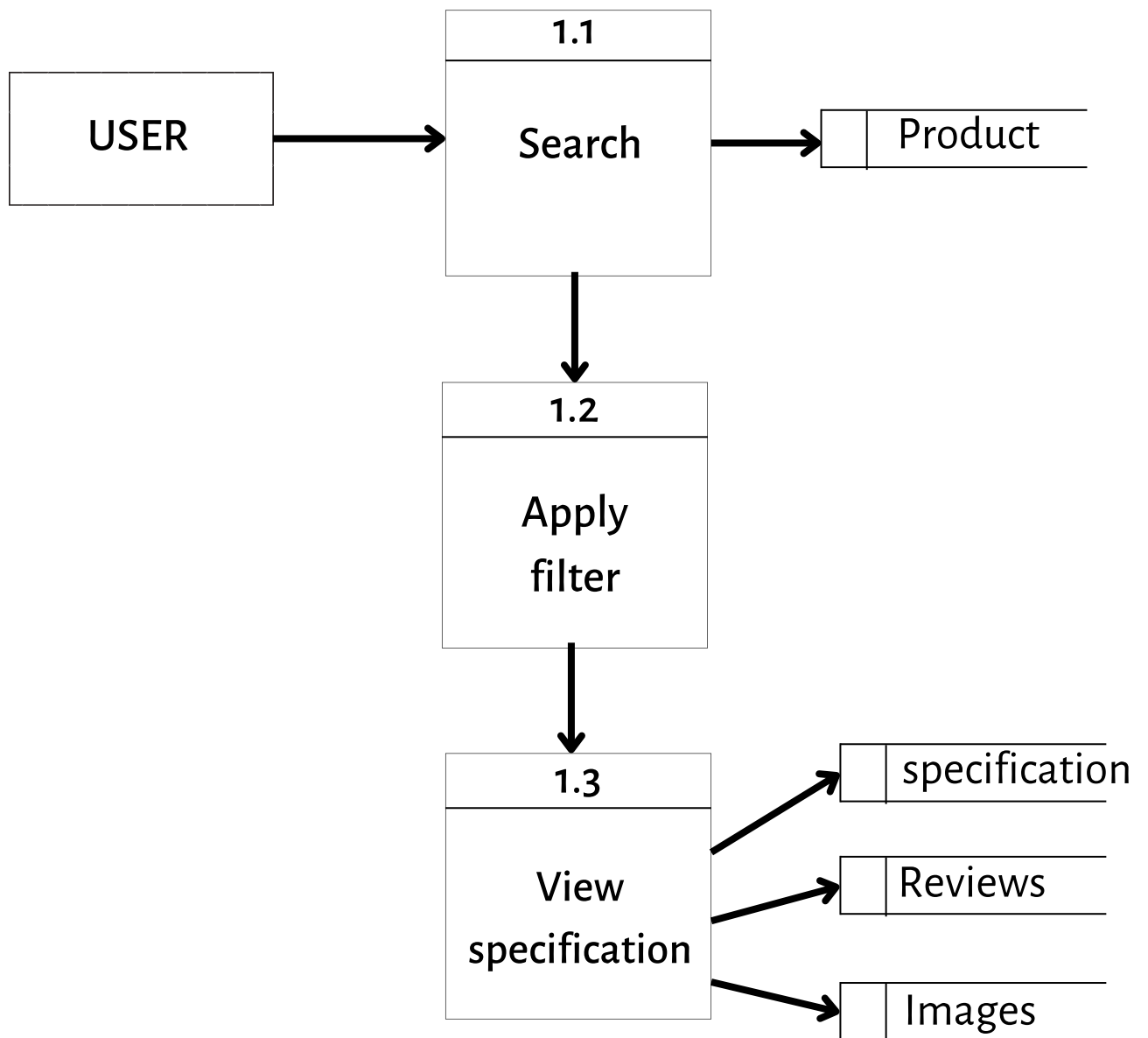
- Context Level Diagram



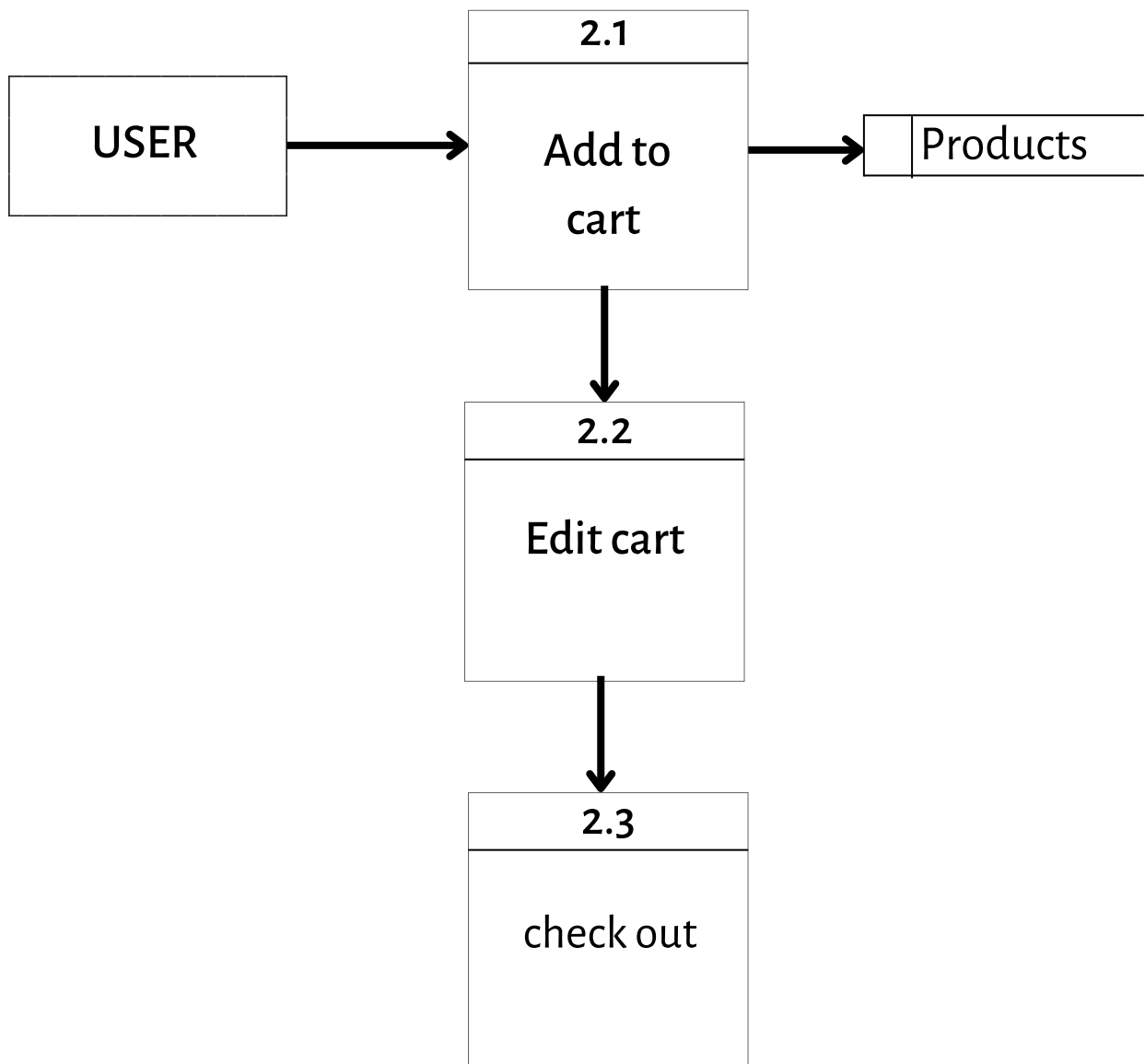
- First level DFD



- Second level DFD



- Second level DFD



10.3: DATA DICTIONARY

Table Name:- Reg_Cust

Primary Key : Reg_id

Foreign Key: -

Description: The information about Registration of Customer

No	Field_Name	Data Type	Constraint	Description
1	Reg_id	int	Primary Key	It stores Registration id
2	First_name	Nvarchar(30)	Not Null	It stores customer First name
3	Last_name	Nvarchar(30)	Not Null	It stores customer Last name
4	Email_id	Nvarchar(20)	Not Null	It store the Email_id
5	Address	Nvarchar(20)	Not Null	It store customer address
6	City	Nvarchar(20)	Not Null	It store customer city
7	Pincode	Nvarchar(6)	Not Null	It store customer Pincode
8	ContactNo	Nvarchar(10)	Not Null	It store customer ContactNo

Table Name:- Log_User

Primary Key : Uni_id

Foreign Key: -

Description: The information about Login

No	Field_Name	Data Type	Constraint	Description
1	Uni_id	int	Primary Key	It stores User ID
2	Username	Nvarchar(30)	Not Null	It stores Username
3	Password	Nvarchar(30)	Not Null	It stores Password
4	Type	Nvarchar(20)	Not Null	It store type of user(customer/manager)

Table Name:- Products

Primary Key : Pro_id

Foreign Key: -

Description: The information about Products

No	Field_Name	Data Type	Constraint	Description
1	Pro_id	int	Primary Key	It stores Product ID
2	PName	Nvarchar(30)	Not Null	It stores Product Name
3	PType	Nvarchar(30)	Not Null	It stores Product Type
4	No_of_items	Nvarchar(20)	Not Null	It store the stock of products

Table Name:- Order_Product

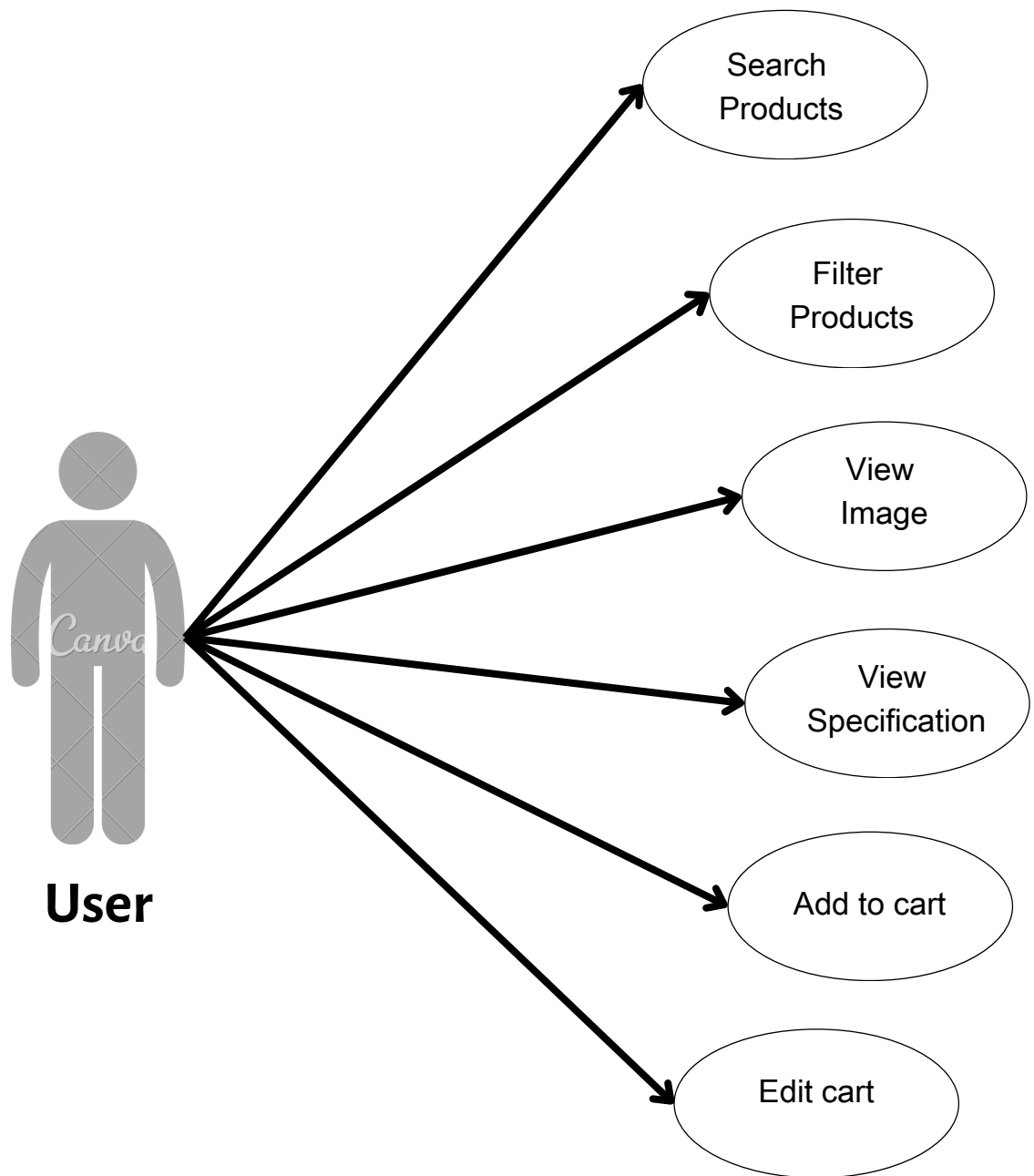
Primary Key : Order_id

Foreign Key: Reg_id, Pro_id

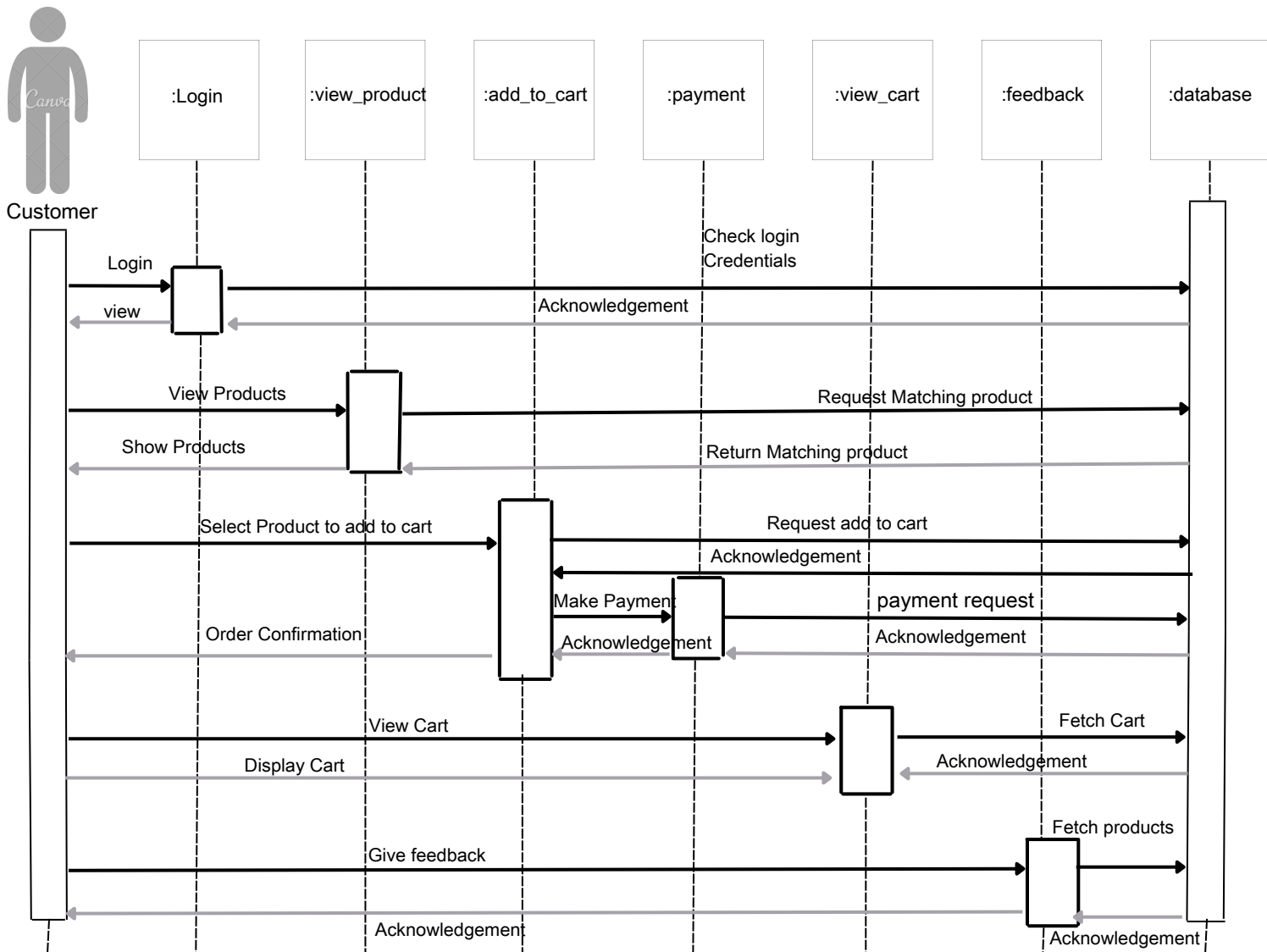
Description: The information about Ordering a product details

No	Field_Name	Data Type	Constraint	Description
1	Order_id	int	Primary Key	It stores Order id
2	Reg_id	int	Foreign Key	Gives reference to Reg_customer
3	Pro_id	int	Foreign Key	Gives reference to Products
4	First_name	Nvarchar(30)	Not Null	It stores First name
5	Last_name	Nvarchar(30)	Not Null	It stores Last name
6	Email_id	Nvarchar(20)	Not Null	It store the Email_id
7	Address	Nvarchar(20)	Not Null	It store customer address
8	City	Nvarchar(20)	Not Null	It stores city
9	Quantity	Nvarchar(6)	Not Null	It store No of items to be booked of any product
10	Transaction_id	Nvarchar(20)	Not Null	It store transaction Details

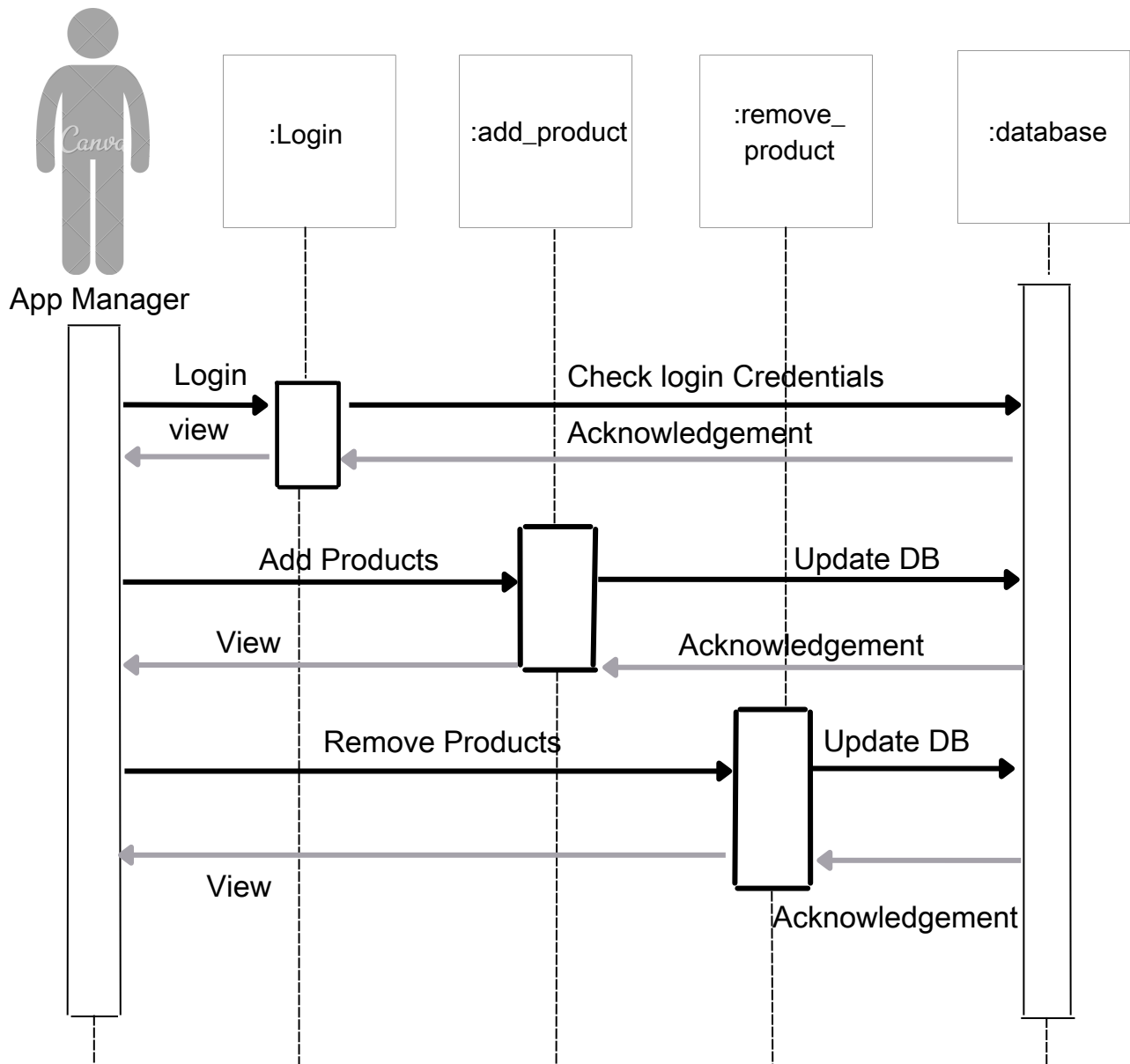
10.4: Use Case Diagram



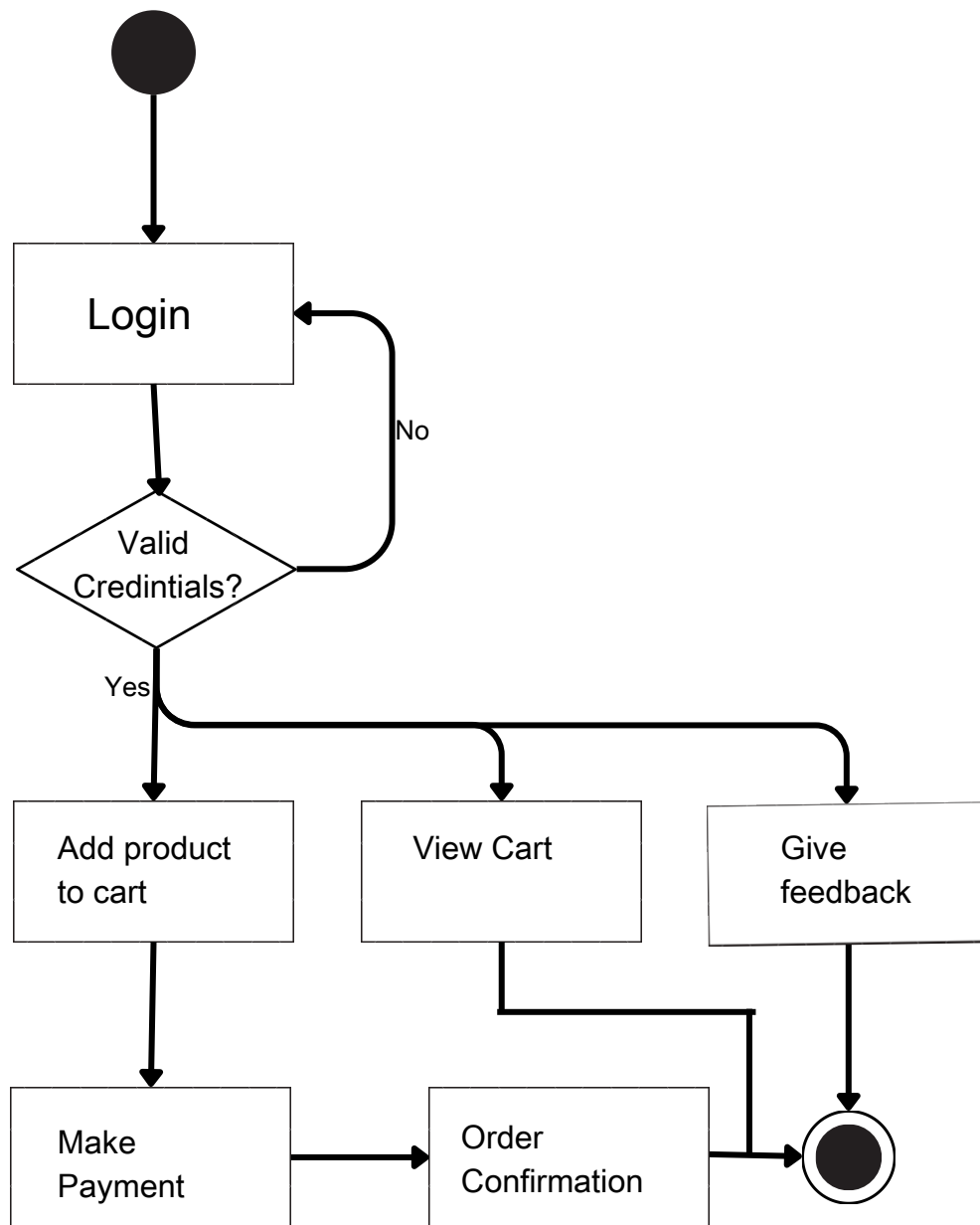
10.5: Sequence Diagram -- Customer



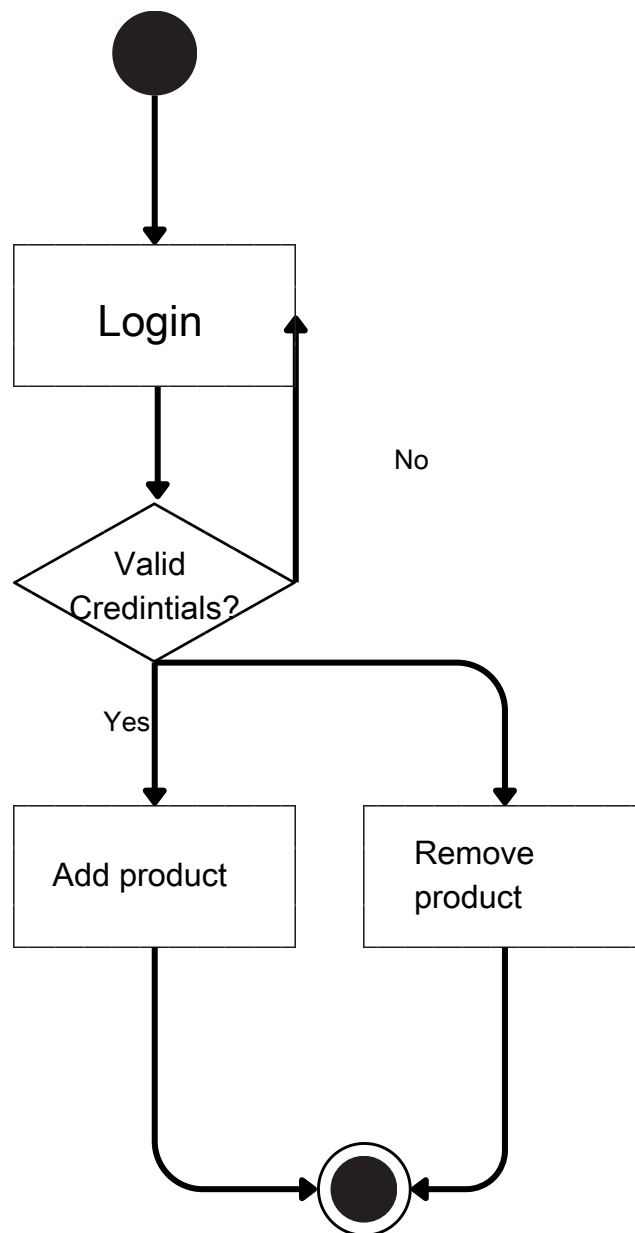
10.6: Sequence Diagram -- App Manager



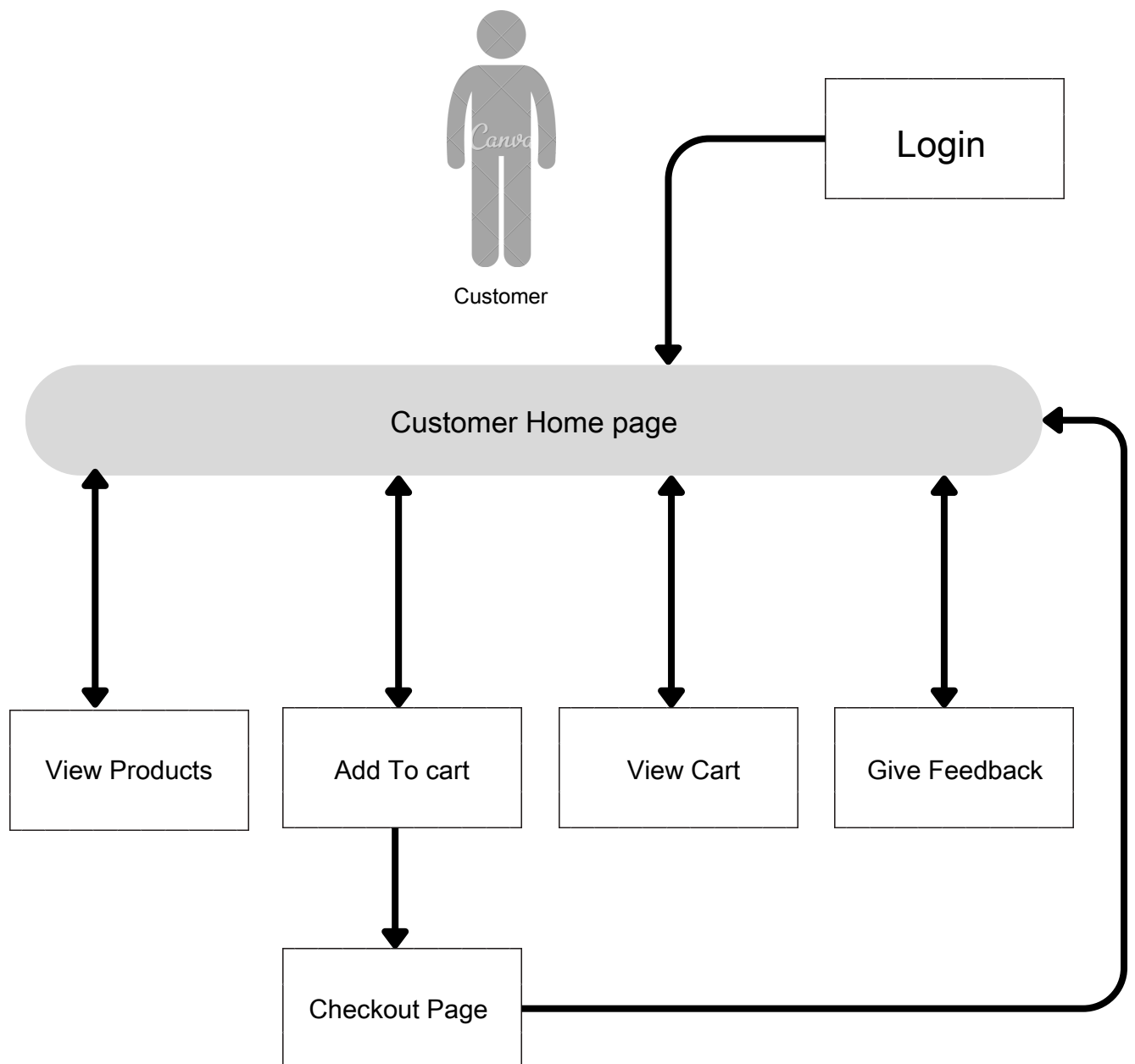
10.7: Activity Diagram -- Customer



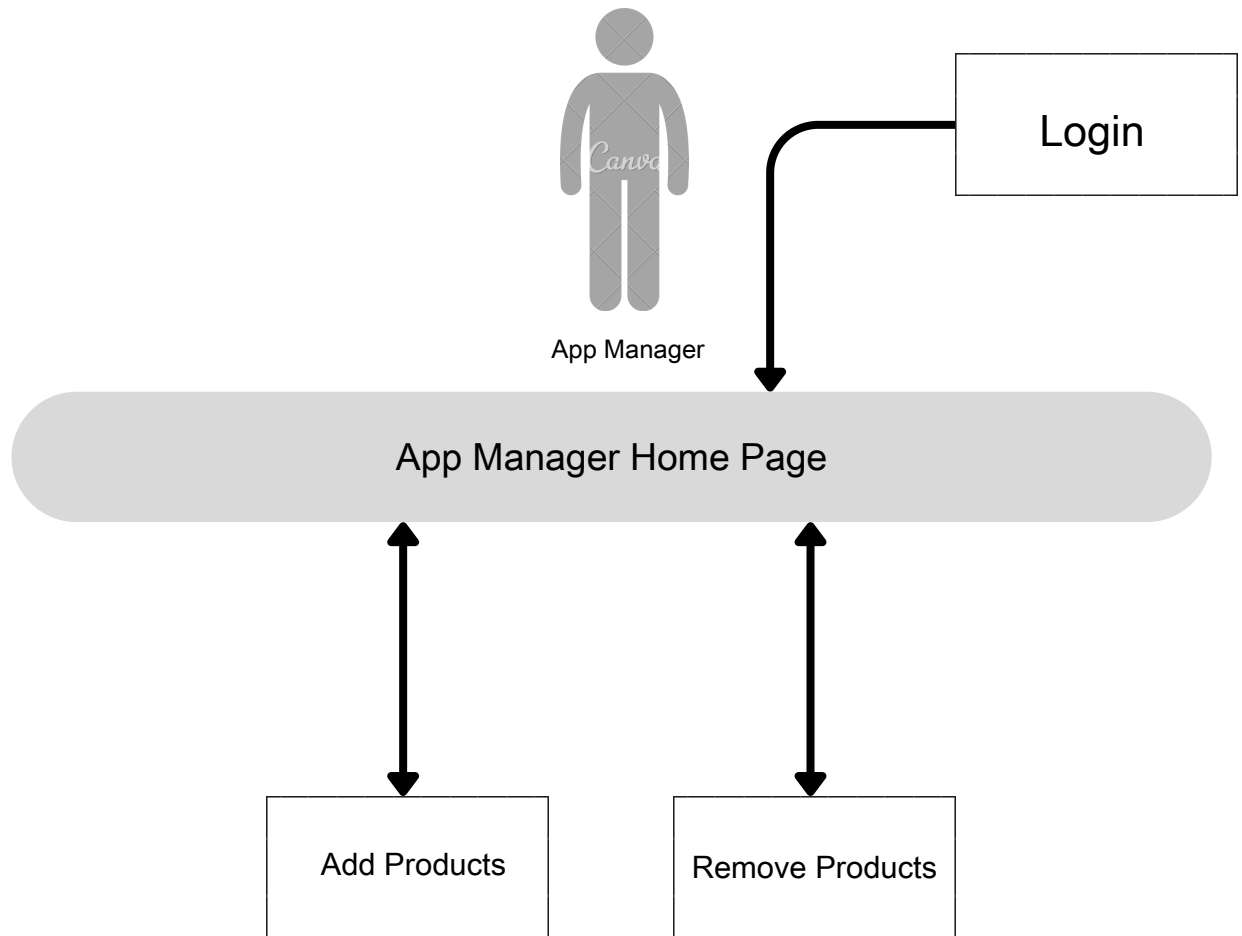
10.8: Activity Diagram -- App manager



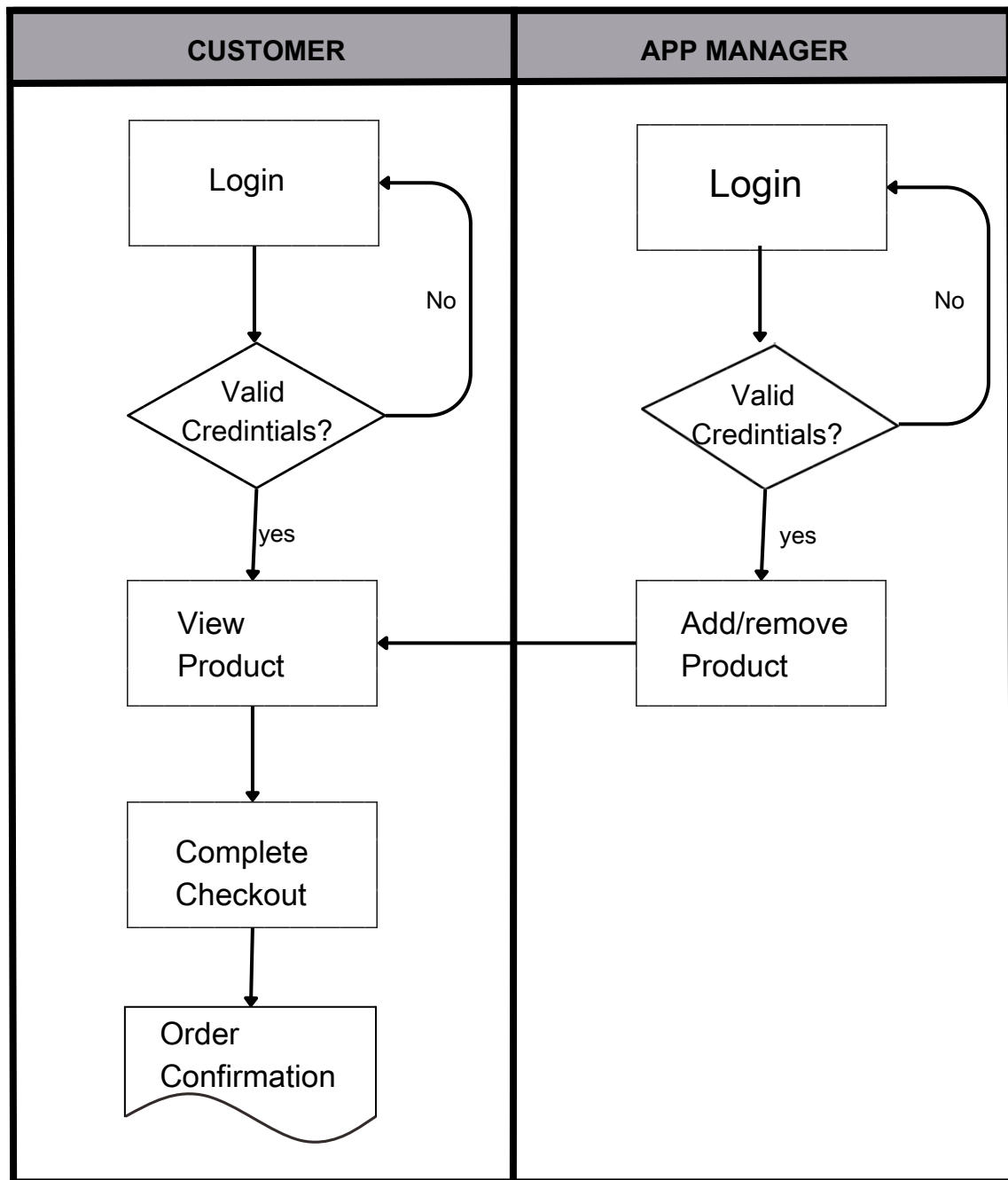
10.9: Navigation Diagram -- Customer



10.10: Navigation Diagram -- App manager



10.11: Swimlane Diagram



Appendix : Glossary

- **SRS:** Software Requirements Specification
- **DB:** Database
- **CSRF:** Cross-Site Request Forgery
- **IE8:** Internet Explorer 8
- **UFP:** Unadjusted Function Point
- **CAF:** Complexity adjustment factor
- **FP:** Function Point
- **NOP:** New Object Point
- **PROD:** Productivity
- **CFG:** Control Flow Graph
- **ER:** Entity Relationship

References

- [1] IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20,1998.
- [2] Pressman, R.S., & Maxim, B.R.(2015). Software Engineering: A Practitioner's Approach 8th edition. McGraw-Hill.
- [3] Aggarwal, K.K., & Singh, Y. (2007). Software Engineering, 3rd edition. New Age International Publishers.