GIFTIFY: FIND THE PERFECT GIFT IN A SNAP!

A Project Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Lavanya Dhanendra Mithbaokar

3021200

Under the esteemed guidance of

Mr. Farhan Shaikh
Lecturer
&

Mrs. Supritha Bhandary
Lecturer

DEPARTMENT OF INFORMATION TECHNOLOGY

M. L. DAHANUKAR COLLEGE OF COMMERCE

(*Affiliated to University of Mumbai*)

MUMBAI, 400057

MAHARASHTRA

2024-2025

Parle Tilak Vidyalaya Association's

M.L.DAHANUKAR COLLEGE OF COMMERCE

(VILE PARLE (E), MUMBAI 400 057

DEPARTMENT OF INFORMATION TECHNOLOGY

APPROVAL OF PROJECT PROPOSAL

PRN No._____ Roll No. _____

Name of the Student: _____

Title of the Project: _____

_____

Name of the Guide: _____

Is this your first submission Yes ☐ No ☐

_____ _____

(Signature of Student) (Signature of Guide)

_____

(COORDINATOR)

Parle Tilak Vidyalaya Association's

M.L.DAHANUKAR COLLEGE OF COMMERCE

(AFFILIATED TO UNIVERSITY OF MUMBAI)
VILE PARLE (E), MUMBAI 400 057.

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled

_____

is bonafide work undertaken by

Mr./Ms. _____ *Seat no.* _____

submitted in partial fulfillment of the requirement for the award of Degree of Bachelor of Science in Information Technology (2024-25) from University of Mumbai.

Date:_____

Internal Guide **External Examiner**

Coordinator

College Seal

ROLES AND RESPONSIBILITIES

I Lavanya Dhanendra Mithbaokar take the responsibility of completing the project. All the roles and responsibilities will be fulfilled by me in the given timeframe.

Roles and Responsibilities for Lavanya Dhanendra Mithbaokar

Front end: The front end consists for the user interface and several modules of the application. This will be developed by using flutter.

Back end: The backend consist of modules and database which will be developed by using Firebase.

ABSTRACT

In an age where gift-giving is an integral part of celebrating special occasions, the process of selecting the perfect gift has become increasingly complex and time-consuming. This project addresses the common challenges faced by consumers in finding thoughtful and suitable gifts amidst an overwhelming array of options. By developing an e-commerce application “GIFTIFY” it leverages data analysis of popular gifting items, user preferences, and trends, this project aims to simplify the gifting experience. The application is designed to provide personalized gift recommendations based on individual user interests, past gift history and specific occasions, thereby reducing decision-making time and enhancing the meaningfulness of each gift. Key features include curated gift sets, customizable gift baskets, and reminders for important dates, and secure online transactions. The app ensures convenience by allowing users to browse and compare options effortlessly, while also offering personalized gift-wrapping services.Giftify aims to revolutionize the gifting experience by offering tailored recommendations and a user-friendly platform for secure transactions, thereby enhancing consumer satisfaction and convenience.

ACKNOWLEDGEMENT

It is indeed with a great pleasure and immense sense of gratitude I acknowledge the help of our principal Prof. Dr. Vinay Bhole for the facilities provided to accomplish this project. I am extremely thankful to out co-ordinator Smt. Archana Talekar for her constant support and inspiration in completing this project. The project could not be completed without the support of Mr. Farhan Shaikh his support has been throughout the process and helped me clear even the smallest doubts related to documentation and helped me in the development of project by suggesting new features to be added that can improve my project. Finally, I express my sincere thanks to all the IT faculties, non-teaching staffs, and all my friends who directly or indirectly helped me in completion of project.

DECLARATION

I hereby declare that the project entitled, “GIFTIFY: FIND THE PERFECT GIFT IN A SNAP!” done at “M. L. DAHANUKAR COLLEGE OF COMMERCE”, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be submitted as final semester project as part of our curriculum.

Lavanya Dhanendra Mithbaokar

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Background

People often give gifts to mark special occasions such as birthdays, weddings and other milestones. But in today's world, finding the perfect gift can be a tedious task. There are way too many gift choices and very less time to explore them all which makes consumers often struggle with choosing thoughtful and suitable gifts for different events and occasions, leading to frustration. Even Google or different websites would provide only a list predefined products rather than products which are actually chosen by gift-givers as a gift. For e.g. It's your friend's birthday, you will find variety of options to select from which at times will be confusing and make gifting process difficult and tiring. But the gifting process would be simpler by only recommending most popular gift items across a wide range of product categories. Hence there is a need for an e-commerce app with only popular recommended gift options that are chosen based on analysis of gifting product dataset, where user data such as user interests, past gift history and preferences will be analyzed to provide tailored and meaningful gift suggestions.

1.2 Objectives

The main objective of this project is to simplify the gifting process and make it convenient and enjoyable for users.

Other objectives of the project are:

- Suggest gifts suitable for different occasions based on trends and user preferences.
- Reduce the time users spend searching for gifts.

- Offer curated combos, thematic gift sets and customizable gift basket options.
- Send reminders for important dates and special occasions.
- Enhance user convenience, allowing them to browse and compare options efficiently.
- Facilitate secure and convenient online transactions for gift purchases.

1.3 Purpose, Scope and Applicability

1.3.1 Purpose

The main purpose of this project is to develop an e-commerce application that will enhance gifting process. This will help users simplify the gift selection process by providing user reviews, ratings, offers and reducing decision-making time. It will suggest gifts suitable for different occasions based on trends and user preferences. It will have gifts according to different categories of events and occasions such as birthday, anniversary, etc. It will offer customizable gifting options across various categories and price ranges and facilitate secure and convenient online transactions for gift purchases.

1.3.2 Scope

Consumers can look for and order gifts whenever they want at their ease. It will also offer personalized gift wrapping options and customizable message to add a personal touch to each gift. Users will be provided with easy access to customer support for assistance with orders, recommendations, or technical issues. The app will send reminders for important dates and special occasions.

Assumption:

- User should have internet connection.
- User should have android (version 12 +).
- User should login into the app to order gift.

Limitation:

- The app won’t function without internet connection.
- User cannot order gifts without logging in the app.

1.3.3 Applicability

- Offers 24/7 service. User can search for and buy products anytime.
- Reduce decision-making time.
- Enhance user convenience.
- Provides combos and customizable gift baskets.
- Provide seamless and enjoyable user experience.

1.4 Achievements

During the research and development of this project, I learned several programming languages, databases, and technologies essential for creating an application. I explored workings and backend processes of e-commerce applications. Given that the app deals with product data analysis, I also learned diverse techniques for data analysis. To enhance the application's efficiency, I learned how to seamlessly integrate various services into its framework.

This experience was challenging and enriching, pushing me beyond my comfort zone and significantly expanding my skill set.

1.5 Organization of Report

- **Chapter 1: Introduction**

  This chapter gives an overview of the project and explains the purpose, scope and availability of the project.

- **Chapter 2: Survey of Technologies**

  This chapter compares various programming language and technologies, explaining the reason behind the choices made for project.

- **Chapter 3: Requirement and analysis**

  The project concept will be defined in this chapter, along with its features and requirements. Additionally, planning and scheduling will be completed. The use case diagram, class diagram, flow chart, DFD, and ERD are all included in the conceptual model part.

- **Chapter 4: System Design**

  This chapter will cover the fundamentals of data module layout design as well as user interface and user experience details. We will also address security-related problems, which is another crucial requirement for this project.

- **Chapter 5: Implementation and Testing**

  This chapter will outline the numerous testing strategies used to uphold product quality along with the project execution process. To determine how the final product will behave, test cases will be produced. Additionally, we will make adjustments and changes to this project.

- **Chapter 6: Results and Discussions**

  Each of the test cases outlined in chapter 5 will be put to the test in this chapter, and the actual behaviour of the product will be documented. Test cases can be verified for the right outcome in accordance with the expectations and results

- **Chapter 7: Conclusion**

  This final chapter defines the completion of the project. Everything covered by the project is listed along with the project limits. All current requirements and possible future improvements are shown. This chapter concludes the documentation for this project.

CHAPTER 2

Survey of Technologies

2.1 Available Technologies

Technologies in app development refer to the tools, frameworks, languages and systems used to create software applications. They are essential for building, deploying and maintaining apps and each plays a specific role in the development process.

For creating an e-commerce android app following technologies can be considered:

Android is an operating system primarily designed for mobile devices. It provides a robust framework and set of tools for developers to create apps that can run on Android-powered devices. The most recent version of Android is 14.

The **frontend** also known as the client-side refers to the user interface and user experience (UI/UX) of the application. Java and Kotlin are the primary languages used for Android development, but there are also other options like C/C++, Python & JavaScript.

There are several **frameworks** used for developing mobile application. These frameworks provide ready-made components, libraries, and tools to streamline the app development process. Some popular frameworks are React Native, Flutter, etc.

The **backend** of an app, also known as the server-side, is responsible for managing data, performing complex calculations, etc. Relational databases like MySQL, Oracle, etc and non-relational databases like MongoDB, Cassandra, etc. are used as backend support for app development.

An **IDE** (Integrated Development Environment) and a **text editor** are both software applications used for writing and editing code. Commonly used IDEs and text editors for app development are Android Studio, Visual Studio, Eclipse, Visual Studio Code, etc

2.2 List of Technologies

1. JavaScript:

JavaScript is a high-level, dynamic, and interpreted programming language that is primarily used for client-side scripting on the web.JavaScript supports object-oriented programming (OOPs) concepts.JavaScript is often used in conjunction with HTML and CSS to create web pages, and is an essential tool for any web developer. It's also popular for developing desktop and mobile applications.

2. Python:

Python is a high-level, interpreted programming language that is widely used for various purposes such as web or app development, artificial intelligence and machine learning, Data analysis, etc. It is known for its simplicity and readability and large community support.

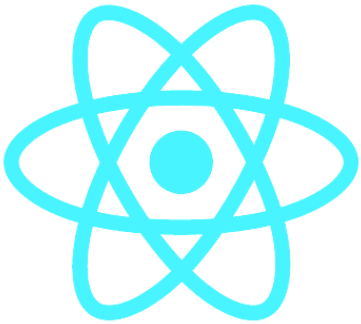
3. Kotlin:

Kotlin is a modern, statically typed programming language that runs on the Java Virtual Machine (JVM). It's designed to be more concise, safe, and interoperable with Java than Java itself. It is used for android app development, backend development, etc.

4. Flutter:

Flutter is an open-source mobile app development framework created by Google. It allows developers to build natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language and is known for building fast, beautiful, and cross-platform apps.

5. React:



React is a popular JavaScript library for building user interfaces (UIs) and can be used for developing web, mobile, and desktop applications. It's ideal for creating reusable UI components and managing state changes. React Native, a framework built on top of React, allows developers to build native mobile apps for Android and iOS using React and JavaScript.

6. MongoDB:

MongoDB is a popular NoSQL, document-based database that allows for flexible and dynamic schema design. It's designed for high performance, scalability and ease of use. It is used for big data storage, real-time analysis and data processing, e-commerce platforms, mobile and web application.

7. Firebase:

Firebase is a comprehensive platform developed by Google for building web and mobile applications. It provides a suite of tools and services for developing, managing, and scaling applications. Firebase provides detailed documentation and cross-platform app development SDKs, to help you build and ship apps for iOS, android, web, flutter, unity, C++, etc.

8. NodeJS:

Node.js is a free, open-source, cross-platform JavaScript runtime environment and libraries that lets developers create servers, web apps, command line tools and scripts. It is built on Chrome's V8 JavaScript engine that allows developers to run JavaScript on the server-side. It is mostly used for server-side or backend development.

9. Android Studio:

Android Studio is the official Integrated Development Environment (IDE) for Android app development. It's a free tool built by Google to help developers create, debug, and test Android applications. Android Studio provides a comprehensive set of tools and features to simplify the app development process

10. Visual Studio Code:

Visual Studio Code (VS Code) is a lightweight, open-source code editor developed by Microsoft. It's a free, cross-platform tool that provides a flexible and customizable development environment for a wide range of programming languages.

2.3 Comparative Study

Technology	Features	Advantages	Disadvantages
JavaScript	• Light weight • Object oriented programming language • Platform Independent	• Saves time and bandwidth • Easily send HTTP request • Open source • Easy to use	• Code is always visible to everyone. • JavaScript DOM is slow
Kotlin	• Statistically typed • Avoid null pointer • Extension functionality	• Less chances of bugs • More reliable • Easy maintenance	• Compilation speed is slow • Minimal learning opportunity
React	• Component base • Virtual DOM • Declarative • Uses JSX	• Reusable components • SEO friendly • JavaScript Libraries	• Poor documentation • Only covers UI • JSX as barrier
Flutter	• Widget-based UI	• Cross-platform • Fast development	• Limited third-party libraries • Large app size

	• Dart programming language • Customizable	• Fast performance	• Limited support
Firebase	• Real-time Database • Cloud Firestore • Cloud storage • Authentication	• Scalability • Security • Seamless integration	• Limited control • Additional cost • Limited support for complex queries
Android Studio	• Android emulator • Gradle build system • Advanced code editing features • Regular updates and bug fixes	• Free and open-source • Fast and efficient • Real-time debugging	• Complex interface • Resource-intensive • Occasional bugs and glitches

Table 2. 1 Comparative Study

2.4 Selected Technology

List of selected technologies and why?

1. Flutter
2. Firebase
3. Android Studio

1. Flutter

Flutter allows you to build apps for multiple platforms. Its "hot reload" feature allows you to experiment, build, and test your app quickly. With its rich set of pre-built widgets, flutter enables you to create beautiful, visually appealing apps. Even developers new to mobile app development find Flutter accessible due to its low barrier to entry. It provides fast performance on both iOS and Android. It allows for extensive customization, enabling you to create unique and branded apps. Additionally, its rich libraries and tools simplify the integration of essential features such as authentication, storage, and networking.

2. Firebase

Firebase provides a fully managed backend infrastructure, saving time and resources. It simplifies app development by providing a suite of tools and services that reduce the complexity of app creation and maintenance. Its Real-time Database or Cloud Firestore ensures seamless real-time data synchronization across all devices. Firebase Authentication provides an easy-to-use authentication system. It provides enterprise-grade security features, such as data encryption and access controls. Firebase Analytics provides detailed insights into app usage and user behaviour. Firebase offers flexible pricing options a free plan and a pay-as-you-go pricing model, reducing costs for small and large projects alike.

3. Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, developed by Google. It is both free and open-source, making it a cost-effective option for app development. It provides a fast and efficient development experience, with features like code completion, code analysis, and debugging. Android Studio's debugging tools allow you to debug your app in real-time. Visual design tools like the Layout Editor and Android NDK simplify UI design and development within Android Studio.

CHAPTER 3

Requirements and Analysis

3.1 Problem Definition

Gifting apps are very useful as they save the time and provide convenience to user. But while being incredibly useful, current apps often face a range of issues that can impact user experience & satisfaction. Here are some common problems with old system:

Overwhelming or Cluttered Design:

- Users may find the app's interface complex and difficult to navigate.
- This leads to frustration and abandoned transactions.

Irrelevant Recommendations:

- Users may be frustrated by irrelevant product suggestions.
- Users may abandon the app if they can't easily locate the gifts they are need.

Limited Gift Selection

- Restricted range of gift options or failure to offer popular and trending items.
- This leads to a less engaging shopping experience.

Limited Product Customization

- Users may find it challenging to personalize gifts to their liking, whether it's through custom messages, unique designs, or specific product features.

Payment Issues

- Problems with transaction processing, including payment gateway errors, issues with refunds, or lack of payment method options.
- This can result in failed transactions, customer frustration, and potentially lost sales.

3.2 Requirement Specification

To address the challenges of gifting e-commerce app, several solutions can be implemented. An ideal gifting app should be user-friendly and provide all the necessary information and functionalities to satisfy the needs of its target audience. Here are solutions to address the common problems:

Overwhelming or Cluttered Design:

- Focus on intuitive and user-friendly design that makes navigation and shopping easy.
- Test the app with users of different age groups and choose the most effective design based on user feedback.

Irrelevant Recommendations:

- Implement better search algorithms that understand user intent and provide relevant results.

Limited Gift Selection:

- Continuously update the inventory with new and trending gifts to keep the selection fresh and appealing.
- Collaborate with various suppliers and artisans to diversify the range of available gifts.
- Collect and analyze user feedback to identify desired custom features and incorporate them.

Payment Issues:

- Simplify the checkout process.
- Offer a variety of payment options, including credit/debit cards, digital wallets, etc.
- Use robust encryption and security measures to protect payment information and ensure smooth transaction processing.

3.3 Planning and Scheduling

3.3.1 GANTT Chart

A Gantt chart is a type of bar chart that illustrates a project schedule, showing the tasks, dependencies, and timelines for a project. It's a visual representation of the project plan, making it easier to understand and track progress.

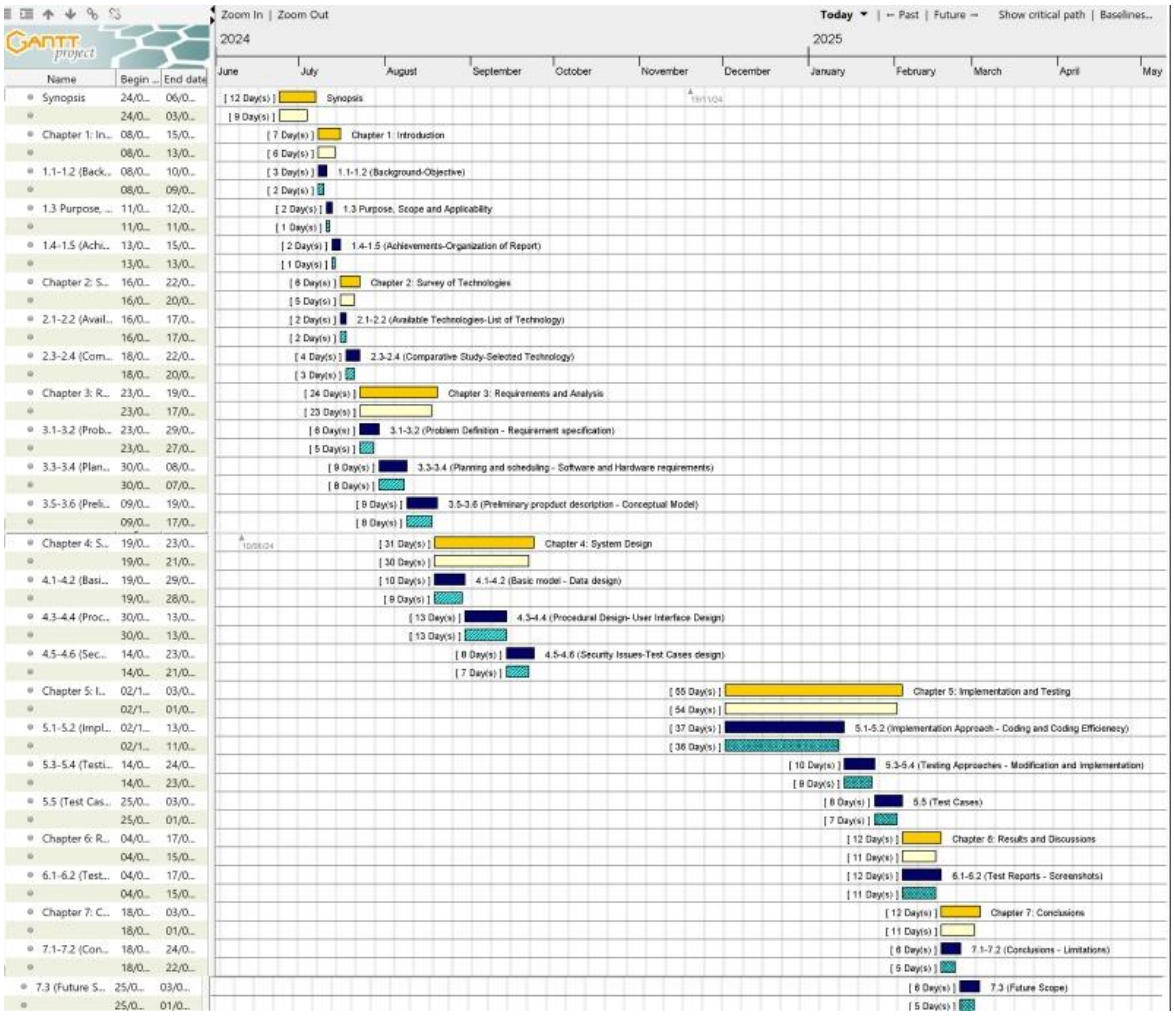


Figure 3. 1 GANTT Chart

3.2.2 PERT Chart

A PERT (Program Evaluation and Review Technique) chart is a type of project management tool used to plan, organize, and coordinate tasks within a project.

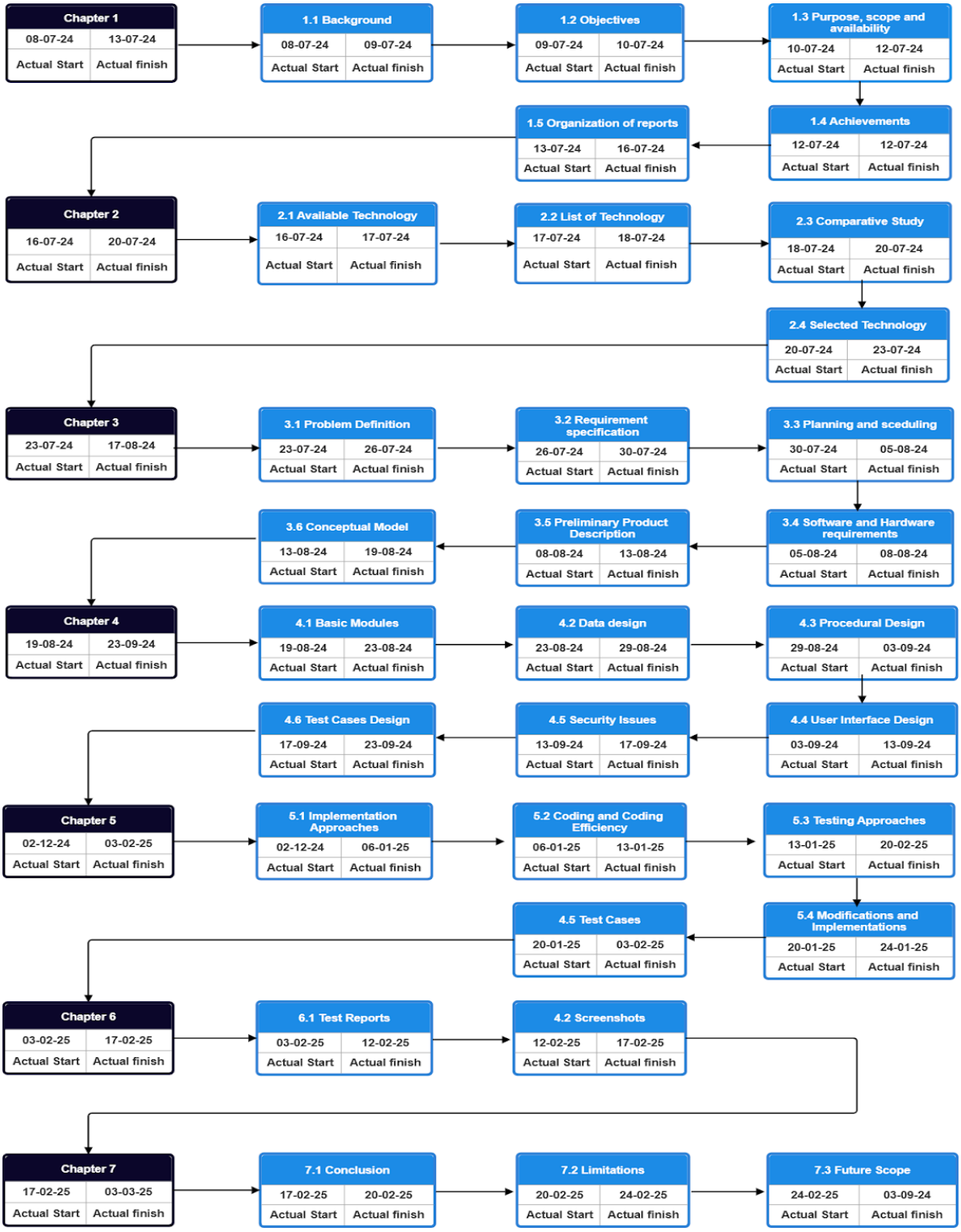


Figure 3. 2 Pert Chart

3.4 Software and Hardware Requirements

The software and hardware requirements for an e-commerce app for developer and user are as follows:

3.4.1 Developer requirements

1. Hardware requirements:

- A computer or laptop with sufficient processing power and memory for coding and running the app with proper electricity supply.
- Reliable and fast internet connection.
- Ram should be 8 GB or above.

2. Software requirements:

- Operating system: Windows 10+
- Frontend: Flutter with Dart.
- Database: Firebase
- IDE: Android Studio

3.4.2 User requirements

1. Hardware requirements:

- User needs a smart phone.
- Reliable and stable internet connection.

2. Software requirements:

- Email address to login to the app.

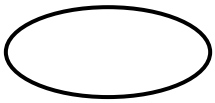
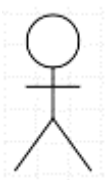
3.5 Preliminary Product Description

This Gifting e-commerce application is designed to enhance and simplify the gift selection process. It will provide user with reviews, ratings and offers. It will suggest gifts suitable for different occasions based on trends and user preferences. It will have different categories of gifts such as birthday, anniversary, etc. hence will enhance the user convenience and reduce the decision-making time user spends searching for gifts.

3.6 Conceptual Model

3.6.1 Use Case Diagram

A Use Case Diagram is a visual representation of the functional requirements of a system, showing how users (actors) interact with the system to achieve specific goals.

Symbol	Name	Description
	Use Case	These describe the actions performed by actors to achieve a specific goal.
	Actor	These are the users or external systems that interact with the system.
	Directed Association	It indicates a relationship between two entities and with its flow.
<<include>>	Include	It indicates one use case includes the behavior of another.

<<exclude>> ⋯▸	Exclude	It indicates one use case extends the behavior of another.
⟶▷	Generalization	It indicates that an actor or use case inherits properties from another.

Table 3. 1 Use Case Diagram

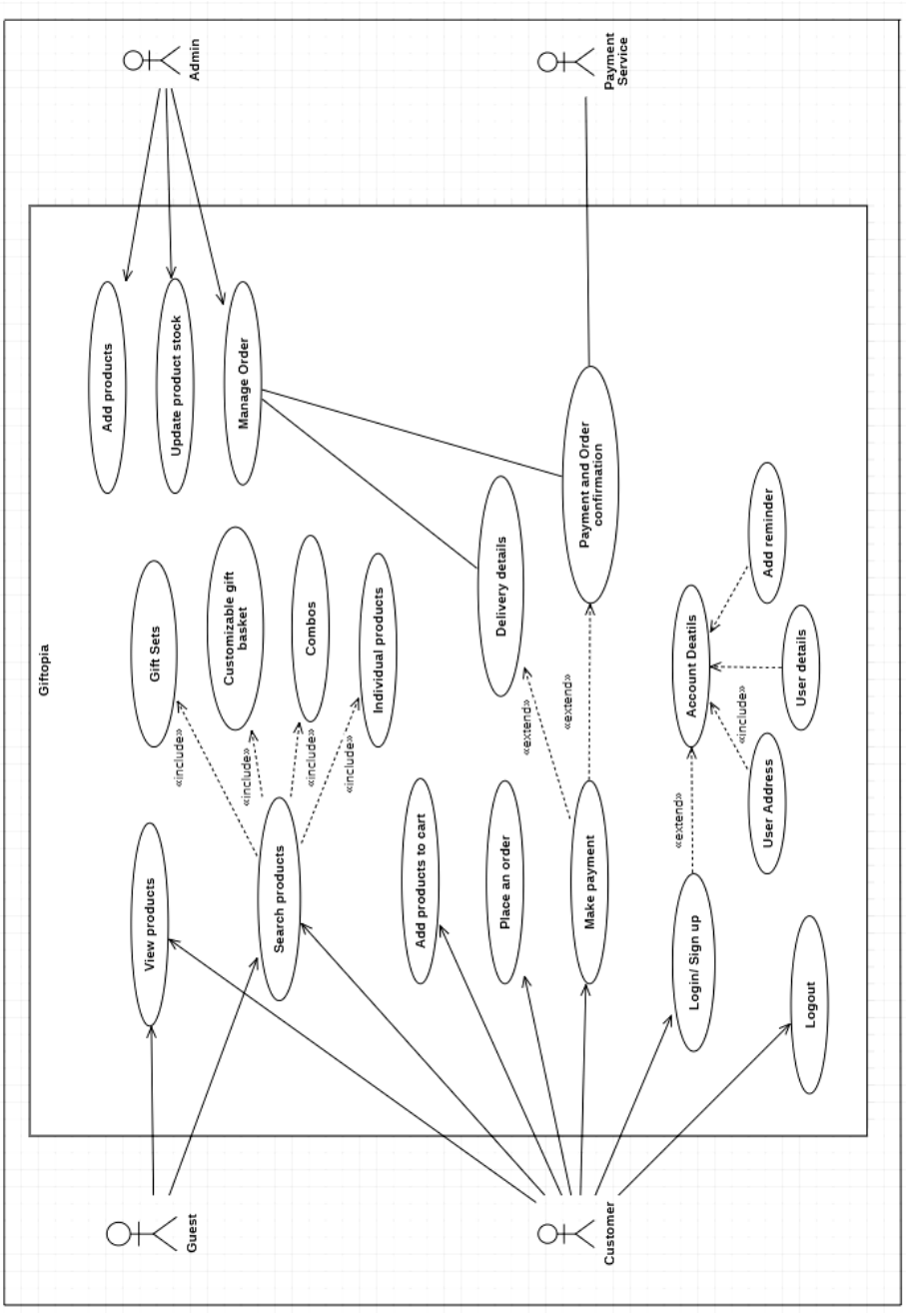
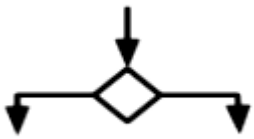


Figure 3. 3 Use Case Diagram

3.6.2 Activity Diagram

An Activity Diagram is a type of UML (Unified Modelling Language) diagram that shows the dynamic behaviour of a system, highlighting the flow of control and data between activities. It's used to model the workflow, business processes, and operational procedures of a system.

Symbol	Name	Description
	Action	These are the actions or tasks performed within the system.
	Initial State	Activity Diagrams start from this step. Also known as the entry state.
	Final State	This is the last stage of the activity diagram. This is where the activity ends in a software system.
	Decision Node	It represents flow of control is determined allowing the diagram to branch into different paths based on specific conditions.
	Merge Node	It represents the convergence of multiple flows into a single path. It's used to reunite branches that were split

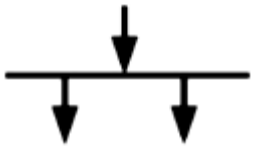
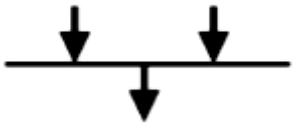
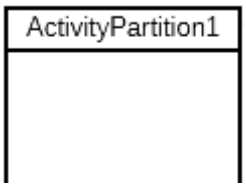
		by a Decision Node or a Fork Node.
	Fork	Two processes execute either concurrently or in parallel at this location. It generally includes a single input but may or may not get one output.
	Join	A join is one where two results of concurrent activities add and form a single result.
	Control Flow	Connectors between two states or two actions to depict the flow. Shows the sequence of execution.
	Swimlanes	Optional, used to organize activities by roles or responsibilities.

Table 3. 2 Activity Diagram

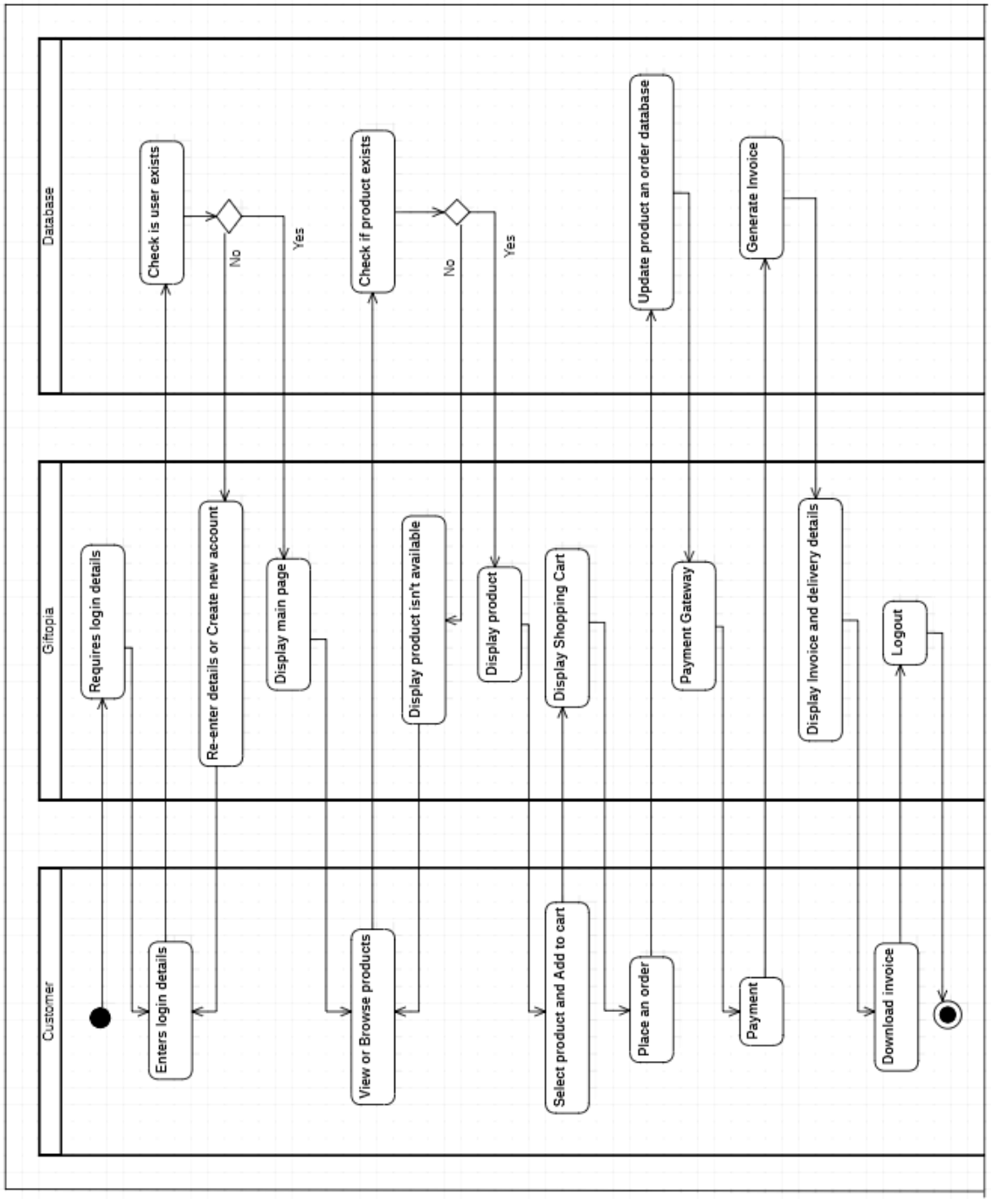


Figure 3. 4 Activity Diagram

3.6.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a visual representation of the flow of data through a system, highlighting the inputs, processing, and outputs. DFDs make it easy to depict the business requirements of applications by representing the sequence of process steps by step. It shows how data enters and leaves the system, what changes the information, and where data is stored.

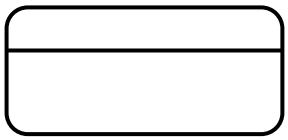
Symbols	Name	Description
	External Entity	These are sources or destinations of data outside the system, such as users, organizations, or other systems.
	Process	Process is a procedure that manipulates the data and its flow by taking incoming data, changing it, and producing an output.
	Data Store	Data stores are files or repositories that hold data for later access such as databases, files, or documents.
	Data Flow	Arrows showing data movement between processes, data stores, and external entities, labeled with the data being transmitted.

Table 3. 3 Data Flow Diagram

- Context Level (Level 0)

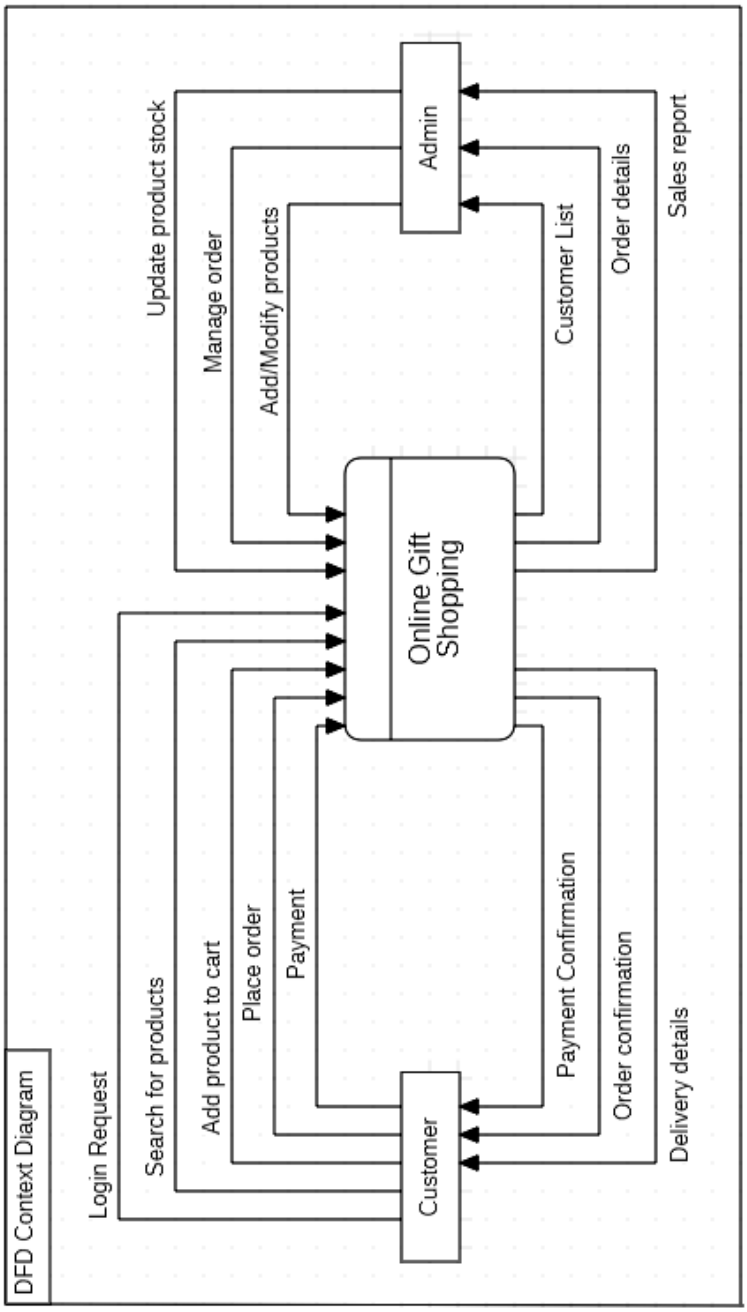


Figure 3. 5 DFD Level 0

- Level 1

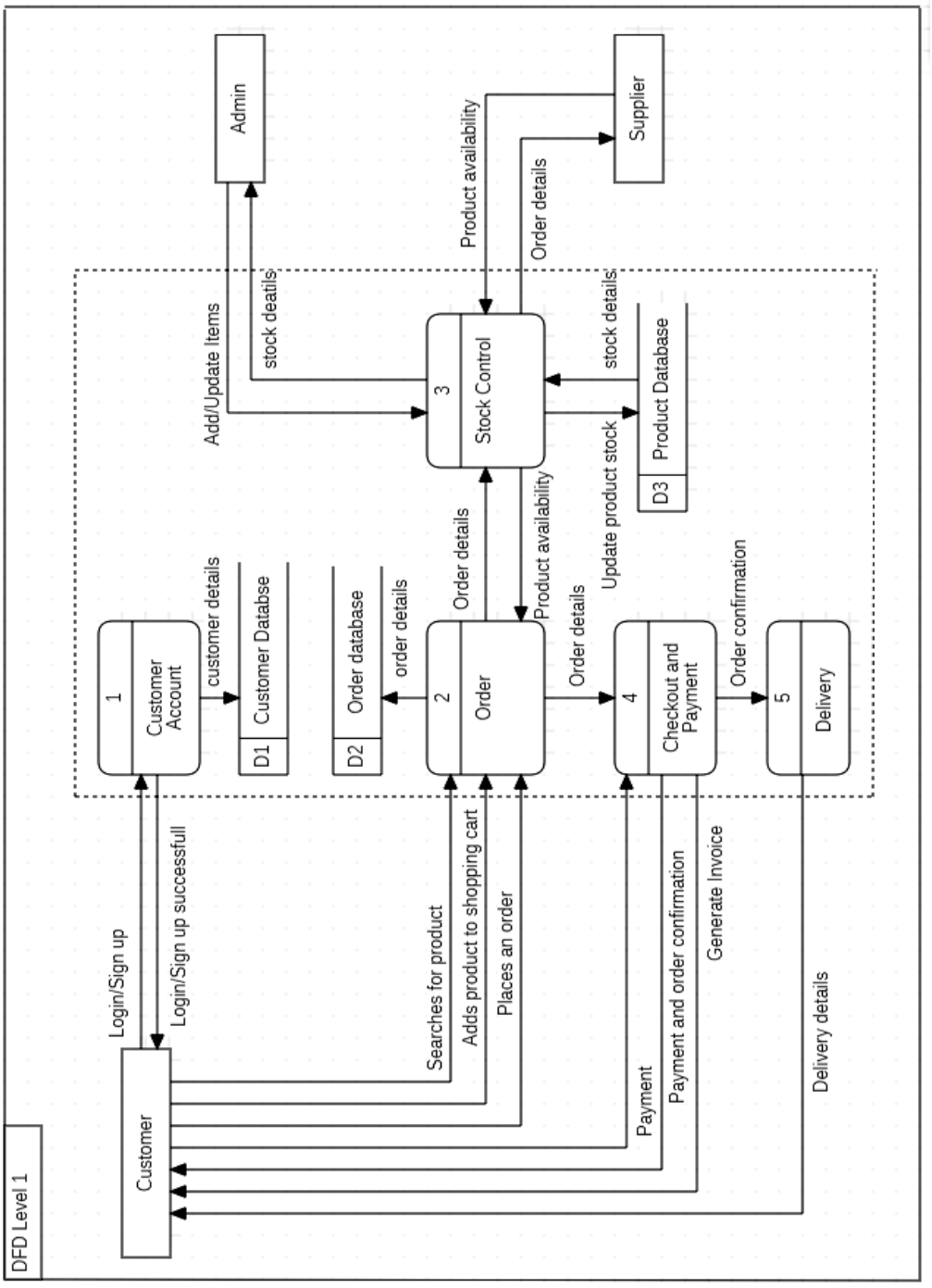


Figure 3. 6 DFD Level 1

- Level 2

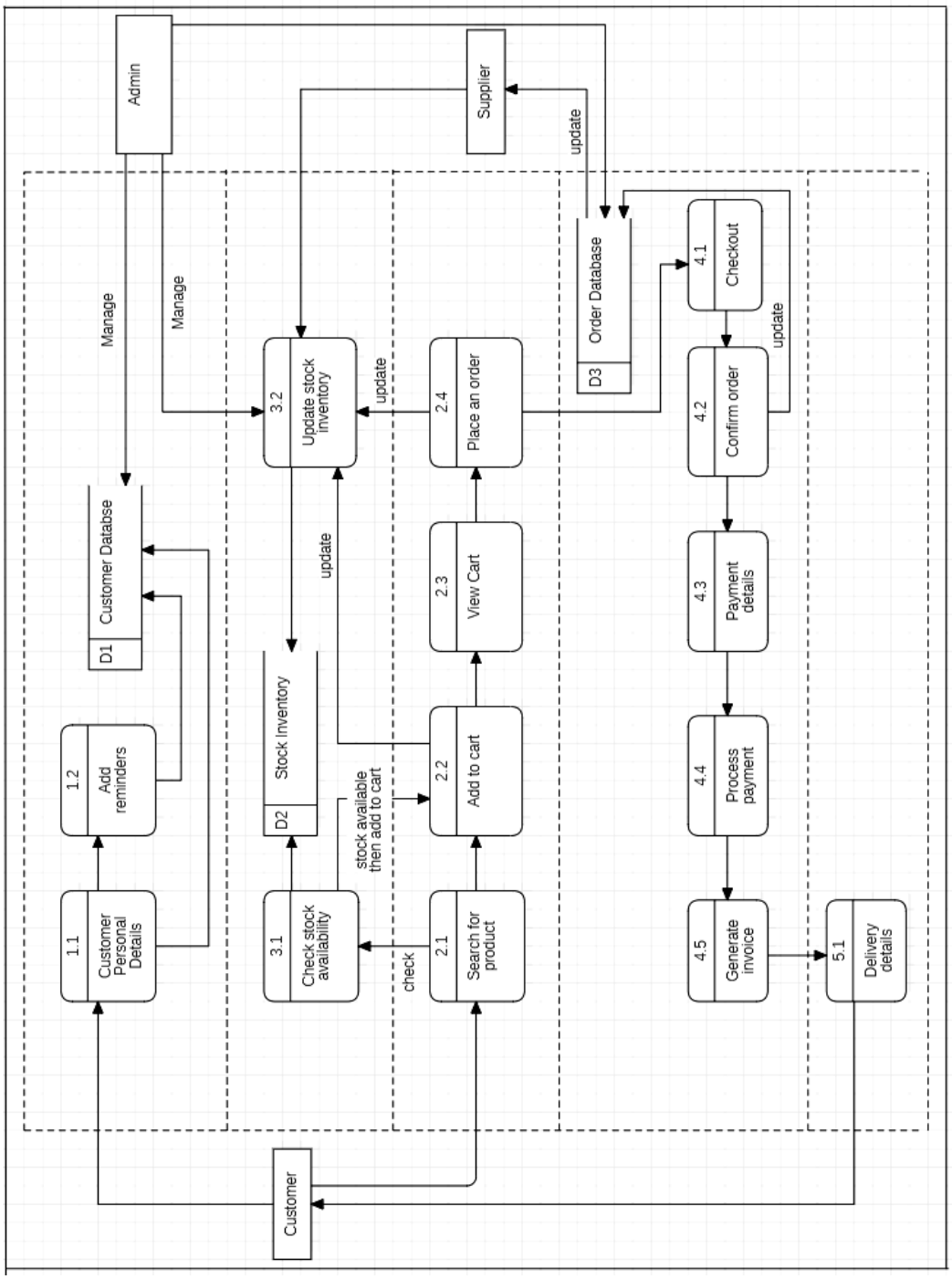


Figure 3. 7 DFD Level 2

3.6.4 ER Diagram

An Entity-Relationship (ER) diagram is a visual representation used to model the structure of a database. ER diagrams use symbols to represent entities, attributes and relationships. ER diagrams are particularly useful for designing and understanding databases and their relationships.

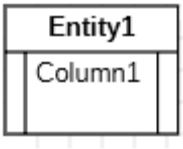
Symbol	Name	Description
	Entity	These are the tables that have data stored about them in the database.
	One-to-one relationship	Single Entity is related to another single Entity.
	One-to-many (1..*) relationship	Single instance of Entity A can be related to multiple instances of Entity B, but each instance of Entity B is related to only one instance of Entity A.
	One-to-many (0..*) relationship	Single instance of Entity A can be related to multiple instances of Entity B, but each instance of Entity B is related to only one instance of Entity A.
	Many-to-Many relationship	Multiple instances of an Entity can be related to multiple instances of another Entity.

Table 3. 4 ER Diagram

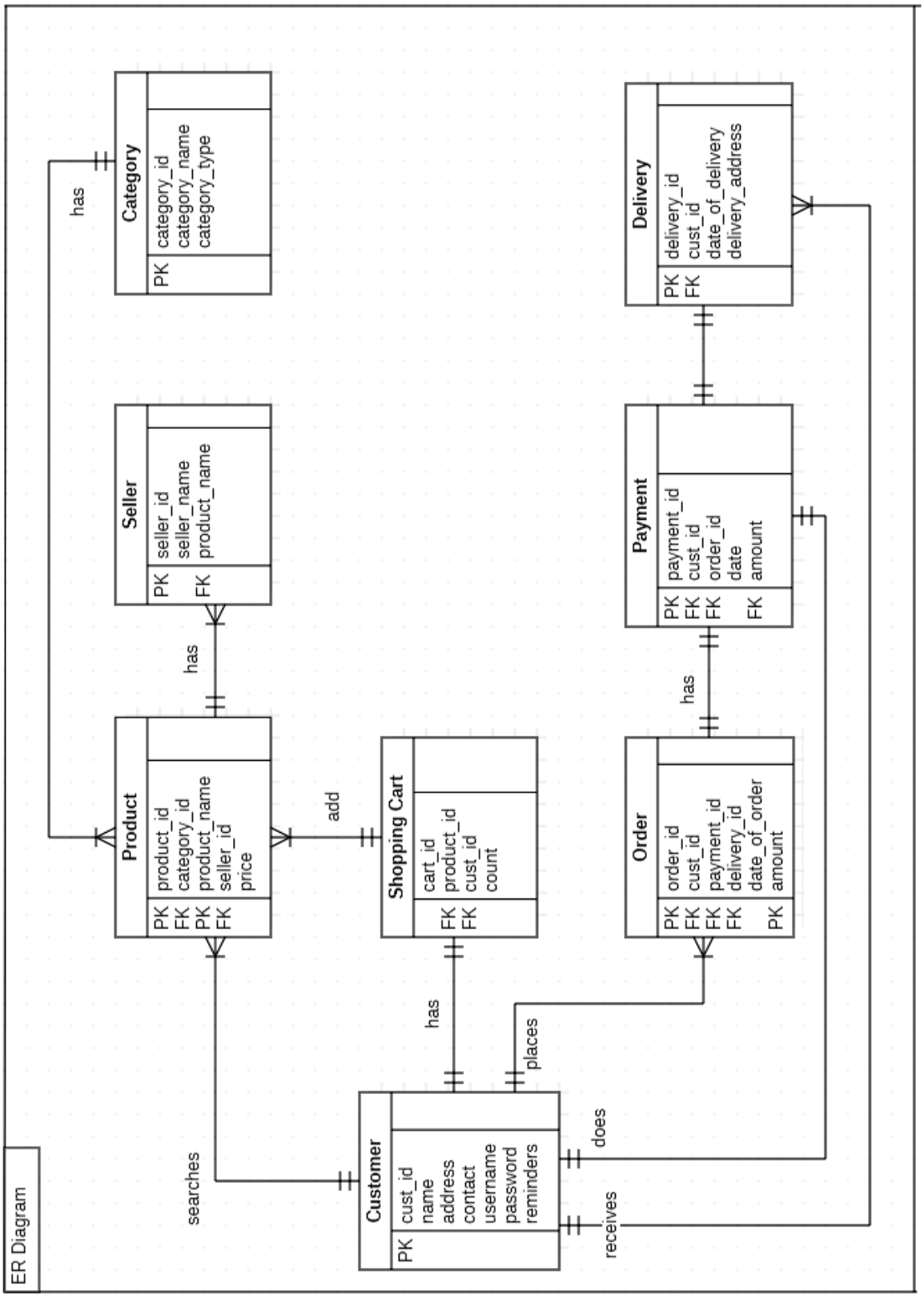


Figure 3. 8 ER Diagram

CHAPTER 4

SYSTEM DESIGN

4.1 Basic Modules

In system design, basic modules refer to the fundamental components or building blocks that make up a larger system. By creating modules, developers can work on individual parts of the system independently, which simplifies updates and enhances scalability. These modules help in organizing the system, making it easier to understand and maintain.

This project has a number of modules some of them are as follows:

User Management Module

- Manages user registration, login and user profile.

Product Catalogue Module

- Display a list of available gifts with filters and Search gifts by category, price, etc.

Gift Recommendation Module

- Suggest gifts based on user preferences.

Shopping Cart Module

- Allow users to add gifts for purchase, modify quantities, and review their selections.

Payment Gateway Modules

- Integrate payment gateways and handle transactions securely, including payment methods and order confirmation.

4.2 Data Design

4.2.1 Schema Design

Database is a fundamental tool for storing and managing information efficiently. It's a place where you can store and organize information in a way that makes it easy to find and use.

Database Schema is like a blueprint for a database. It defines the structure of the data stored within it.

Schema design refers to the process of planning and organizing the structure of a database. It involves specifying the types of data that will be stored, the relationships between different data elements, and the rules about how data can be entered, modified and accessed. A schema defines how data is stored, managed, and retrieved within a database management system.

- **Database Name**
  - ❖ GIFTIFY

- **List of Tables**
  - ❖ Customer
  - ❖ Products
  - ❖ Category
  - ❖ Order
  - ❖ Payment

4.2.2 Data Integrity and Constraints

Data integrity refers to the accuracy, consistency, and reliability of data within a database and **constraints** are rules or restrictions imposed on data to maintain data integrity.

- **Database Name**
  - ❖ GIFTIFY

- **List of Tables**

I. Customer

Field	Type	Null	Key	Default	Extra
Cust_id	INT	NO	PRI	NULL	
Cust_name	VARCHAR(50)	YES		NULL	
Username	VARCHAR(50)	YES		NULL	
Phone_no	INT	NO		NULL	
Password	VARCHAR(50)	YES		NULL	

Table 4. 1 Customer

II. Products

Field	Type	Null	Key	Default	Extra
Product_id	INT	NO	PRI	NULL	
Product_name	VARCHAR(50)	YES		NULL	
Price	INT	YES		NULL	
Category_id	INT	YES	FR	NULL	

Table 4. 2 Product

III. Category

Field	Type	Null	Key	Default	Extra
Category_id	INT	NO	PRI	NULL	
Category_name	INT	YES	FR	NULL	
Product_id	VARCHAR(50)	YES	FR	NULL	

Table 4. 3 Category

IV. Order

Field	Type	Null	Key	Default	Extra
Order_id	INT	NO	PRI	NULL	
Customer_id	INT	YES	FR	NULL	
Product_id	INT	YES		NULL	
count	INT	YES		NULL	
amount	INT	YES	PR	NULL	

Table 4. 4 Order

V. Payment

Field	Type	Null	Key	Default	Extra
Payment_id	INT	NO	PRI	NULL	
Customer_id	INT	YES	FR	NULL	
Order_id	INT	YES	FR	NULL	
Date	DATE	YES		NULL	
Amount	INT	YES	FR	NULL	

Table 4. 5 Payment

4.3 Procedural Design

Procedural design is a software development methodology that focuses on breaking down a problem into a series of steps or procedures that are executed sequentially. It emphasizes the flow of control and the order in which instructions are executed.

4.3.1 Logic Diagrams

A logic diagram is a visual representation of the logical flow and decision-making processes within a software program. By using logic diagrams, developers can improve the clarity, efficiency, and maintainability of their code.

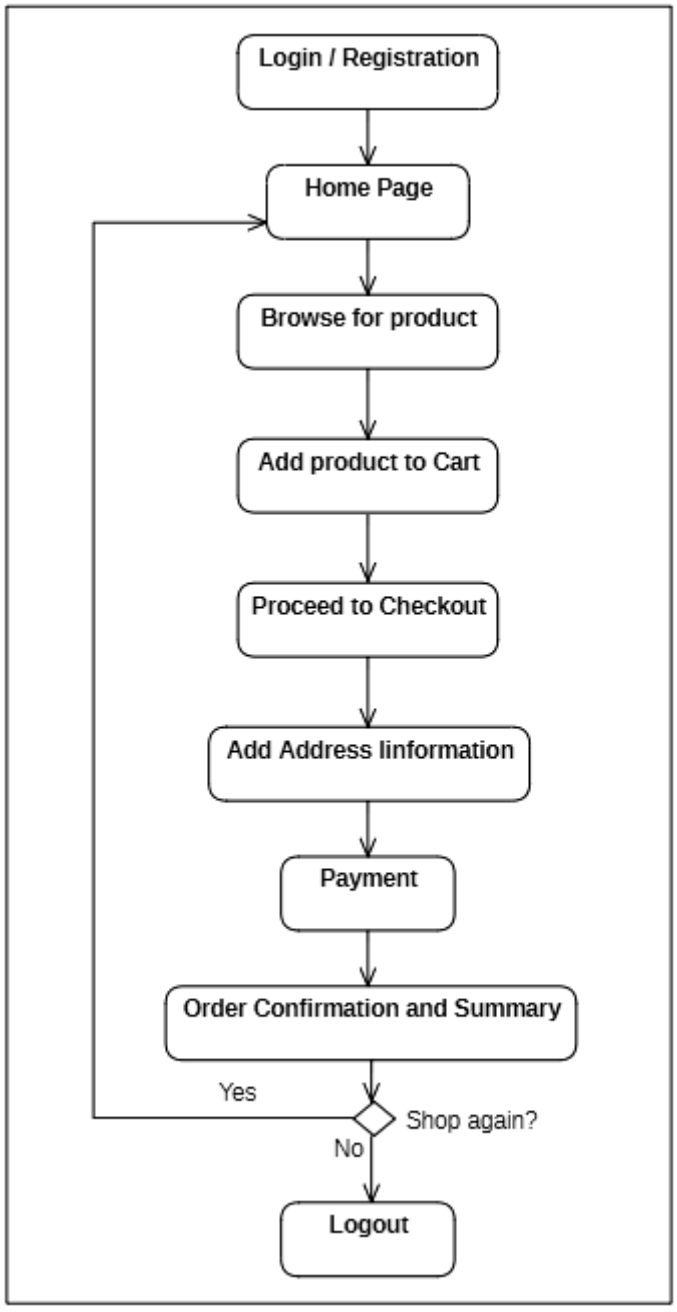


Figure 4. 1 Logic Diagram

4.3.2 Data Structures

List of common Data Structures

- **Array:** A collection of elements of the same data type accessed using a numerical index.
- **Linked List:** A linear collection of elements, where each element (node) points to the next element.
- **Stack:** A LIFO (Last-In-First-Out) data structure where elements are added and removed from the same end.
- **Queue:** A FIFO (First-In-First-Out) data structure where elements are added to one end (rear) and removed from the other end (front).
- **Tree:** A hierarchical data structure where each node can have zero or more child nodes which is used for efficient searching, sorting, and data organization.
- **Binary Tree:** A tree where each node has at most two children and is used for indexing and decision processes.
- **Graph:** A collection of nodes (vertices) connected by edges and useful for networks, social graphs and transportation systems.
- **Heap:** A complete binary tree that satisfies the heap property (e.g., min-heap or max-heap) which is used in priority queues and algorithms like heap sort.

To efficiently manage and organize the user, product and order database, I would consider using data structures like:

- **Graphs:** Implementing a social graph for user interactions, recommendation systems based on user preferences and gift relationships.
- **Heap:** Implementing a priority queue for sorting gift recommendations based on relevance or popularity.
- **Tree:** Organizing gift categories, implementing a search tree for efficient product lookup.
- **Arrays:** Manage and store a list of gifts, categories, or user preferences.

4.4 User Interface Design

1. Login Page

Figure 4. 2 Login Page

On the login page, user will enter credentials, such as a username and password, to access a specific app. If the correct credentials are entered and verified, the user is granted access to the app. If a new user wants to access the app, then user must click on Sign Up button.

2. Registration Page

Figure 4. 3 Registration Page

A signup page or registration page allows users to create a new account to access the app. Once a user fills out the signup form and submits it, the system will create a new account based on the provided information. The user can then use their newly created account to access the app.

3. Home Page

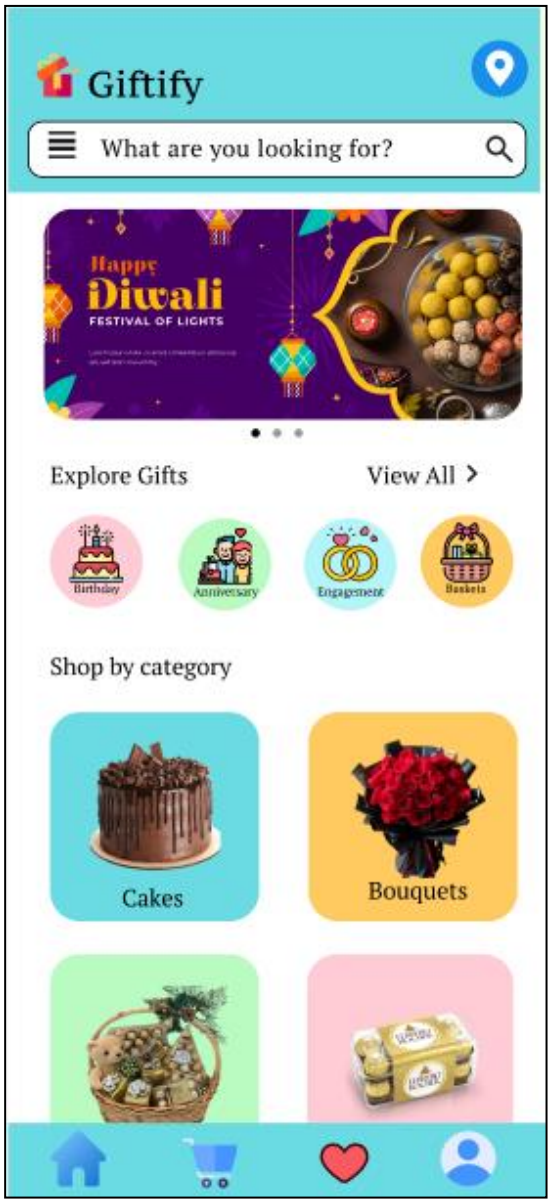


Figure 4. 4 Home Page

The homepage features a search bar, a banner, and various sections.

Search bar allows users to search for specific gifts.

Banner displays offers, upcoming events and redirects users to relevant gift options.

Section lets user browse gift based on occasions such as birthdays, anniversary, etc and categories such as cakes, bouquets, etc.

4. Product Page

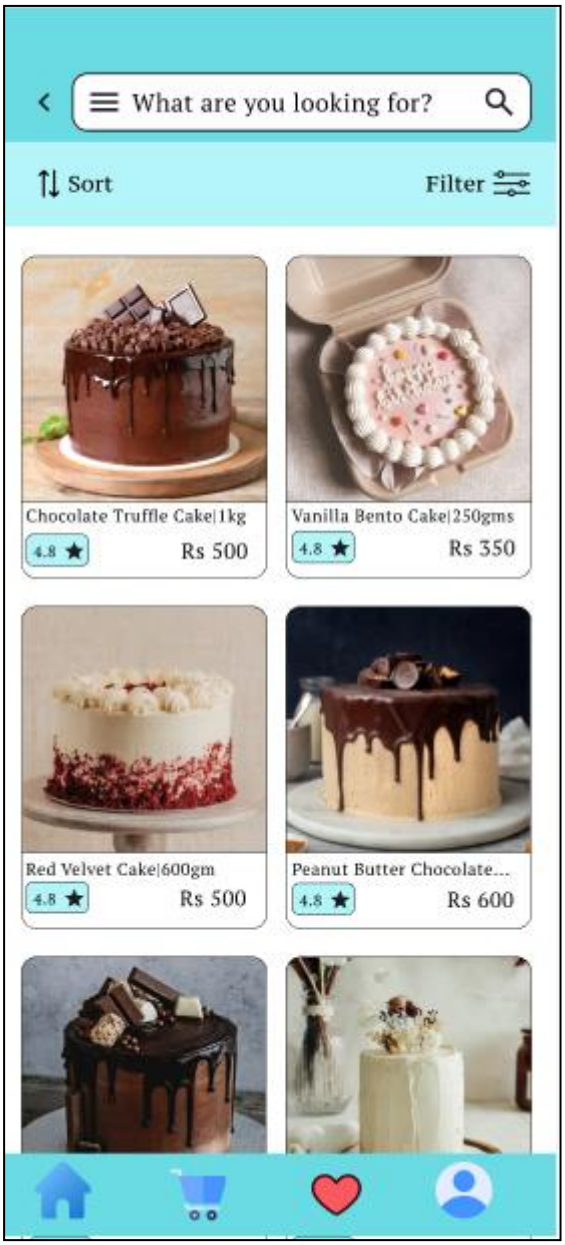


Figure 4. 5 Product Page

Product page shows a list of products (here cake) with their names, descriptions, ratings, and prices. Users can sort and filter the results to find their desired product. To view detailed information about a product, user needs to click on it.

5. Detailed Product Page

Figure 4. 6 Detailed Product Page

Here, the product is displayed with a detailed description, price, rating and an option to add it to the cart.

6. Cart

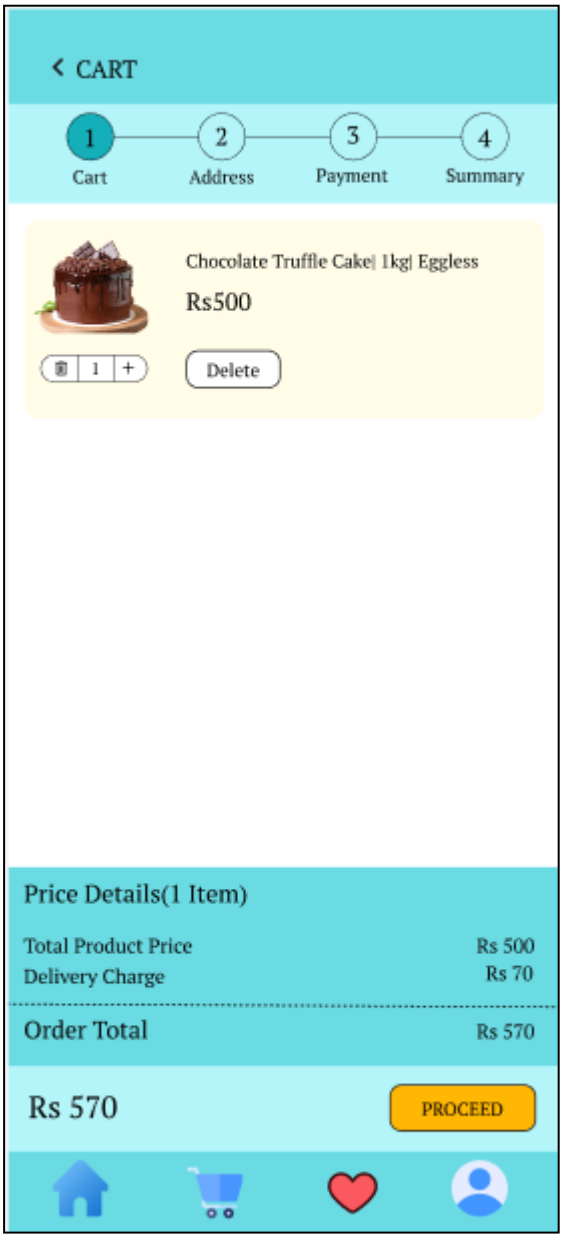


Figure 4. 7 Cart

Cart stores the items a user intends to purchase. It acts as a temporary holding area for products before the user proceeds to checkout and completes their order. The product's total price and delivery charge are displayed, along with a button to proceed to checkout.

7. Checkout Process: Address Information, Payment and Order Confirmation

i. Address Information

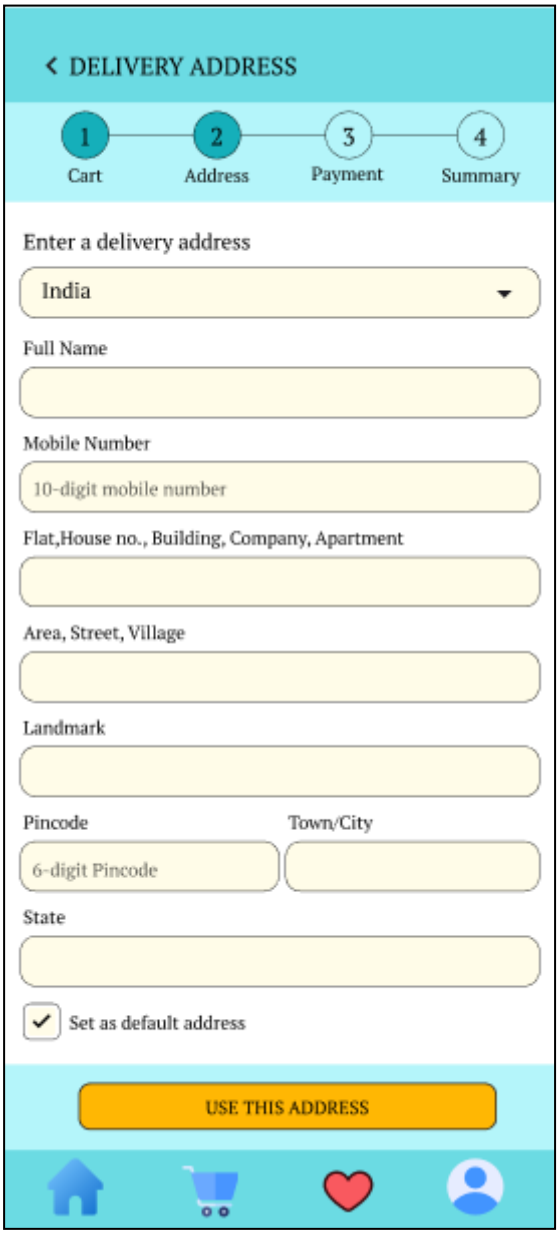


Figure 4. 8 Address Information

The address information form is part of the checkout process, allowing users to specify where their order should be delivered.

ii. Payment

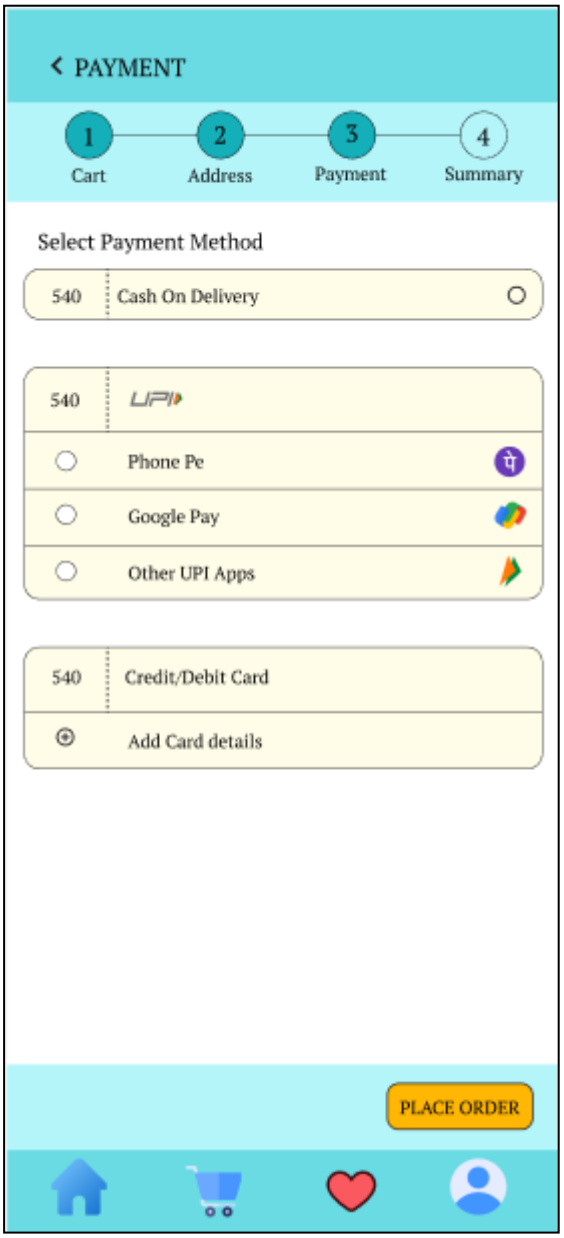


Figure 4. 9 Payment

The user can choose to pay for the order using Cash on Delivery, UPI payments (PhonePe, Google Pay, or other UPI apps), or Credit/Debit Card. After selecting a payment method and completing the transaction, the user can proceed to place the order.

iii. Order Summary/Conformation

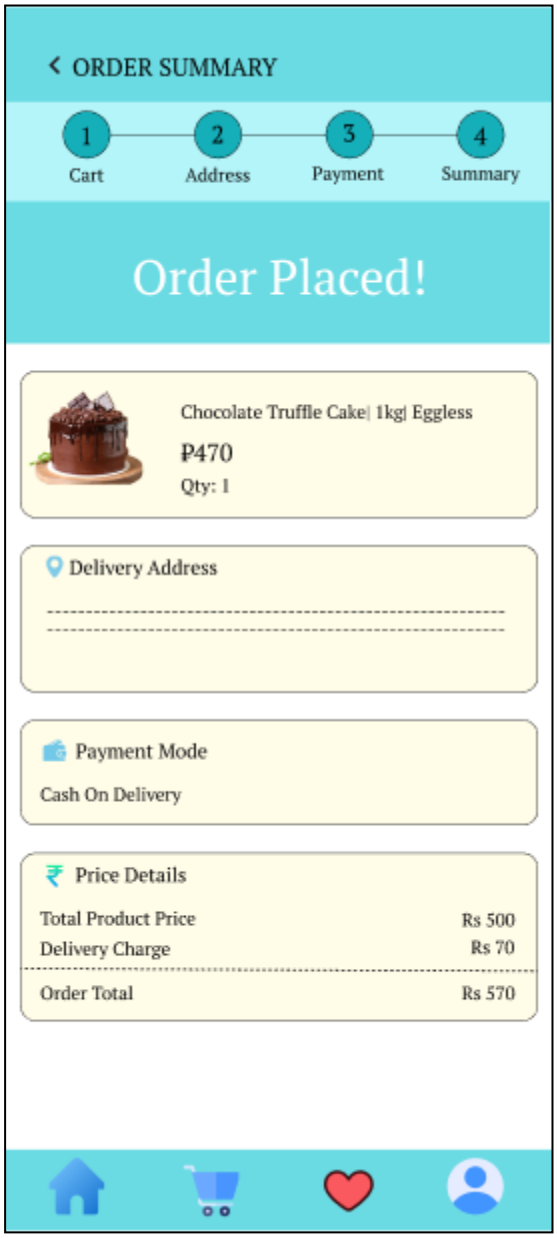


Figure 4. 10 Order Summary

The order summary page displays the completed order details, including the product, quantity, delivery address, payment mode, and total price. The user can now view their order history or continue shopping.

4.5 Security Issues

1.Data Exposure

- **Issue**:

  Sensitive user data (emails, addresses, preferences) may be exposed due to insecure APIs or improper data handling.

- **Solution**:

  Implement proper access controls, use secure API and ensure sensitive data is only accessible to authorized users. Use encryption for data at rest and in transit.

2.Insecure Payment Processing

- **Issue**:

  Payment information may be intercepted or mishandled, leading to fraud.

- **Solution**:

  Integrate with reputable payment gateways that comply with PCI DSS. Use HTTPS for all transactions and do not store sensitive payment information unless absolutely necessary.

3.Inaccurate Recommendation

- **Issue:**

  Users might get poor or irrelevant gift suggestions due to limited data or flawed algorithms.

- **Solution:**

  Enhance recommendations with machine learning, personalize suggestions based on user profiles, and regularly update algorithms with new data.

4.Complex Navigation

- **Issue:**

  A complicated interface can confuse users and lead to a frustrating experience, diminishing overall satisfaction and usability.

- **Solution:**

  Simplify navigation with clean layout, user-friendly design and intuitive menus to improve usability.

4.6 Test Cases Design

1. Login Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Login Success	User enters valid credentials and logs in	User must have an account	User successfully logs in and is redirected	Pass/ Fail
TC-02	Invalid Login Attempt	User enters invalid credentials	-	Error message: "Invalid credentials" is displayed	Pass/ Fail
TC-03	Empty Username or Password	User leaves username or password blank	-	Error message: "Please fill in required fields"	Pass/ Fail

Table 4. 6 Login Page Test Case

2. Registration Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Successful Registration	User enters valid data and registers	-	User account is created and redirected to homepage.	Pass/ Fail

TC-02	Password Mismatch	User enters non-matching passwords	-	Error message: "Passwords doesn't match" is displayed	Pass/ Fail
TC-03	Mandatory Field Validation	User skips mandatory fields	-	Error message: "Fill required fields" is displayed.	Pass/ Fail
TC-04	Invalid phone no. Format	User enters invalid phone no. format		Error message: "Invalid phone no. format" is displayed	Pass/ Fail

Table 4. 7 Registration Page Test Case

3. Home Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Search Functionality	User searches for a gift	User logged in	Search results relevant to the query are displayed	Pass/ Fail
TC-02	Category Selection	User selects a gift category to browse	-	Gifts within the selected category are displayed	Pass/ Fail
TC-03	Banner Navigation	User clicks on banner showing offers	-	User is redirected to the offer details page	Pass/ Fail

TC-04	Browse Popular Items	User clicks on a section for popular items	-	List of popular gifts is shown	Pass/ Fail
TC-05	Invalid Search Input	User enters a nonsensical search query	-	Message: "No results found" is displayed	Pass/ Fail

Table 4. 8 Home Page Test Case

4. Detailed Product Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Add to Cart	User clicks "Add to Cart" button	Product available	Product is added to the cart	Pass/ Fail
TC-02	Product Out of stock	System prevents adding out-of-stock products to the cart.	User attempts to add an out-of-stock product to their cart.	System displays an "Out of Stock" message and prevents the product from being added.	Pass/ Fail
TC-03	Add Quantity	User increases product quantity in the cart	Product added to cart	Quantity is updated and total price recalculated	Pass/ Fail

Table 4. 9 Detailed Product Page Test Case

5. Cart Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Update Cart Quantity	User modifies the quantity of a product	Product in cart	Quantity is updated, and the total price is recalculated	Pass/ Fail
TC-02	Remove Item from Cart	User removes a product from the cart	Product in cart	Product is removed from the cart	Pass/ Fail
TC-03	Proceed to Checkout	User clicks "Proceed to Checkout"	Product in cart	User is redirected to the address information page	Pass/ Fail
TC-04	View Cart Summary	User views total price and delivery charges	Product in cart	Correct total price and delivery charge are displayed	Pass/ Fail
TC-05	Empty Cart Notification	User tries to proceed with an empty cart	Cart is empty	Message: “Your cart is empty” is displayed	Pass/ Fail

Table 4. 10 Cart Page Test Case

6. Checkout Process Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Invalid Shipping Address	User enters an invalid shipping address.	User enters address where selected product isn't deliverable.	Error message: "Product not deliverable to selected address" is displayed.	Pass/ Fail
TC-03	Successful Order Placement	User has successfully placed an order with valid information.	User has products in the cart, valid shipping address, and payment method.	Order is placed successfully, and user receives an order confirmation page with order details.	Pass/ Fail

Table 4. 11 Checkout Process Test Case

7. Payment Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Select Payment Method	User selects a valid payment method	User in checkout process	Payment method is successfully processed	Pass/ Fail

TC-02	Invalid Payment Details	User enters invalid payment details	User in checkout process	Error message: "Invalid payment details" is shown	Pass/ Fail
TC-03	Payment Gateway Timeout	Payment gateway times out during transaction	User in checkout process	Error message: "Payment failed, please try again"	Pass/ Fail
TC-04	Payment Confirmation	User completes the payment successfully	Valid payment details	Payment confirmation and order summary are displayed	Pass/ Fail
TC-05	Cash on Delivery Option	User selects Cash on Delivery	User in checkout process	Order confirmation message is displayed	Pass/ Fail

Table 4. 12 Payment Page Test Case

8. Order Confirmation Page Test Case

Test Case ID	Test Case Name	Description	Precondition	Expected Result	Status
TC-01	Order Summary Display	User views the order summary after placing order	User has completed payment	Order details, including product, address, and payment method are displayed	Pass/ Fail

Table 4. 13 Order Confirmation Page Test Case

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1. Implementation Approach

This project is developed using the **Incremental Model**.

The **Incremental Model** follows an iterative approach where the system is built and delivered in increments. Each increment adds new functionality until the complete system is developed.

1. **Requirement Gathering**
   - Here I defined the complete set of requirements.
   - This included technologies used like Flutter, Firebase, VS Code etc.
   - Then I Divided requirements into multiple increments and prioritized features and functionalities for each increment.
   - Output of this phase is Software Requirement Specification (SRS) document.
2. **Design & Development:**
   - After the requirements were clear, I started designing the core functionalities.
   - I've used Figma to create the basic design of the system and define module interaction.
   - Once the design was finalized, I proceeded with the development phase and began coding the application.
3. **Testing**
   - Once the app was developed, I performed unit and integration testing for the developed components to ensure everything was working as expected.
4. **Implementation**
   - Deploy the initial version of the system with core features.

This project was completed in 10 increments:

Increment 1:

- In the first increment, I created a Flutter project in VS code and set up an Android project in Firebase.
- I also designed the user sign-up and login page and integrated Firebase Authentication with it.
- Then I implemented and tested the authentication flow to ensure the functionality.

Increment 2:

- In the second increment, I designed the homepage of the app with UI components like Carousel Slider, List of Events and Gift Category Cards.
- Then I implemented the Home page and verified if all the components were displayed as expected.

Increment 3:

- In the third increment, I designed and implemented Product Page, Detailed Product page, Cart Page, Checkout Page and Order Summary page.
- I also worked on seamless navigation between the different pages of the app.

Increment 4:

- In the fourth Increment, I designed and implemented the Profile page of the user.
- I also designed and added My orders and Wishlist sections under Profile page.

Increment 5:

- In the Fifth increment, I integrated Firebase Firestore with my project to store the user data and product details in the database.
- I also stored Products across multiple categories in the database
- Implemented the changes and Verified data storage and retrieval from Firebase Firestore.

Increment 6:

- In the sixth increment, I created a function to fetch product data from Firestore and display it on the Product page.
- I also developed function for fetching the Product details and display them on Detailed Product Information page.
- I implemented these functions and verified correct data retrieval and display.

Increment 7:

- In the seventh increment, I worked on Cart management.
- I developed a function for adding product to cart and displaying them on the Cart page.
- I integrated RazorPay with my app as a payment gateway for seamless transactions.
- Implemented and conducted successful testing of cart and payment functionality.

Increment 8:

- In the eight increment, I developed a search function to allow users to find products easily.
- Implemented the search function and verified the accuracy and efficiency of search results.

Increment 9:

- In the ninth increment, I developed the reminder feature, implemented and tested it.

Increment 10:

- Refined the **UI/UX** and made final **logic improvements** for a smoother experience.
- Final testing of the app.

5.2. Coding and Coding Implementation

```
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:flutter_timezone/flutter_timezone.dart';
import 'package:timezone/data/latest.dart' as tz;
import 'package:timezone/timezone.dart ' as tz;

class NotiService {
  final notificationPlugin = FlutterLocalNotificationsPlugin();

  bool _isInitialized = false;

  bool get isInitialized => _isInitialized;

  Future<void> initNotification() async {
    if (_isInitialized) return;

    tz.initializeTimeZones();
    try {
      final String currentTimezone = await FlutterTimezone.getLocalTimezone();
      tz.setLocalLocation(tz.getLocation(currentTimezone));
    } catch (e) {
      print("Error setting timezone: $e");
    }

    const initSettingsAndroid =
        AndroidInitializationSettings("@mipmap/ic_launcher");

    const initSettings = InitializationSettings(android: initSettingsAndroid);

    await notificationPlugin.initialize(initSettings);

    _isInitialized = true; //
  }

  NotificationDetails notificationDetails() {
    return const NotificationDetails(
      android: AndroidNotificationDetails(
        'daily_channel_id',
        'Daily Notifications',
        channelDescription: 'Daily Notification Channel',
        importance: Importance.max,
        priority: Priority.high,
        playSound: true,
        enableVibration: true,
      ),
    );
  }
```

```
  Future<void> showNotification({
    int id = 0,
    String? title,
    String? body,
  }) async {
    return notificationPlugin.show(id, title, body, notificationDetails());
  }

  Future<void> scheduledNotification({
    int id = 1,
    required String title,
    required String body,
    required int hour,
    required int minute,
  }) async {
    if (!_isInitialized) {
      await initNotification();
    }

    final now = tz.TZDateTime.now(tz.local);

    var scheduledDate =
        tz.TZDateTime(tz.local, now.year, now.month, now.day, hour, minute);

    if (scheduledDate.isBefore(now)) {
      scheduledDate = scheduledDate.add(Duration(days: 1));
    }

    await notificationPlugin.zonedSchedule(
      id,
      title,
      body,
      scheduledDate,
      notificationDetails(),
      uiLocalNotificationDateInterpretation:
          UILocalNotificationDateInterpretation.absoluteTime,
      androidScheduleMode: AndroidScheduleMode.inexactAllowWhileIdle,
    );
  }
}
```

5.3. Testing Approach

5.3.1 Unit Testing

The following components underwent unit testing during application development:

1. **Authentication Module**
    - Ensured only valid users could access the system.
    - Tested authentication and validation in Firebase.
2. **Login Module**
    - Verified that correct credentials led to successful login.
    - Ensured new user registration details were stored correctly.
3. **Payment Module**
    - Verified transactions were processed only with valid payment details.
    - Ensured RazorPay integration worked as expected.
4. **Product Management**
    - Verified product details were fetched and displayed correctly.
    - Ensured cart functionality worked as expected.
5. **Order Management**
    - Checked that orders were placed and stored properly.
    - Verified order summary and history were updated correctly.
6. **Search Module**
    - Ensured search returned relevant products.
    - Verified efficiency and response time.
7. **Wishlist & Profile Management**
    - Checked user profile updates and Wishlist additions.
8. **Reminder Feature**
    - Verified reminders were created and triggered at the correct time.

5.3.2 Integration Testing

Following components were integrated and tested:

1. **Authentication & Login Module**
    - Ensured authentication seamlessly transitioned into login.
2. **Cart & Payment Integration**
    - Verified that products added to the cart are proceeded correctly to checkout and payment.
3. **Product Management & Firebase**
    - Ensured product data was correctly fetched, stored, and updated from Firestore.
4. **User Profile & Order History**
    - Ensured seamless integration between user orders and profile management.

5.3.3 System Testing

- Performed **end-to-end testing** on the entire application.
- Ensured **seamless navigation** between all modules.
- Verified all **database operations** worked correctly.
- Conducted **stress testing** on large data operations

5.3.4 Usability Testing

- Evaluated the application's **user experience and interface design**.
- Collected user feedback on ease of navigation and accessibility.
- Made necessary improvements to **enhance UI/UX**.
- Conducted usability testing with real users to assess ease of navigation.
- Ensured UI elements were **intuitive and user-friendly**.
- Addressed feedback for improving **design and user experience**.

5.3.5 Compatibility Testing

- Tested the application on different **Android devices and screen sizes**.
- Ensured compatibility with different **network conditions and device performance levels**.

5.4. Modification and Implementation

1. Authentication Issues
   - **Error:** Authentication was failing for some users due to incorrect or incomplete details being stored.
   - **Solution:** Implemented validation checks to ensure authentication details are correctly stored and retrieved before verifying the user.

2. UI Alignment Issues
   - **Error:** UI elements such as buttons and text fields were misaligned on different screen sizes.
   - **Solution:** Applied responsive design principles in Flutter, utilizing `MediaQuery` and `Flexible` widgets for better adaptability.

3. User Session Management Issues
   - **Error:** Users had to log in every time they reopened the app, even after logging in previously.
   - **Solution:** Integrated shared preferences to store user details persistently, allowing automatic login after the first authentication.

4. Cart Management Bugs
   - **Error:** Items were not displaying when users added products to the cart.
   - **Solution:** Implemented real-time listeners on Firestore to ensure instant synchronization of cart changes.

5. Reminder Feature Notification not triggering
    - **Error:** Users were not receiving timely reminders due to background execution limits.
    - **Solution:** Integrated Flutter Local Notification for push notifications and ensured support for background execution.

5.5. Test Cases

1. Authentication Module Test Case

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Blank Authentication Key	-	Error
2	Invalid Authentication Key	random_invalid_key	Error
3	Valid Authentication Key	correct_key	Proceed to Homepage

Table 5. 1 Authentication Module Test Case

2. Login Test Case

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Blank username, password	username=, password=	Error
2	Blank username, correct password	username=, password= valid_password	Error
3	Correct username, blank password	username=valid_user, password=	Error
4	Incorrect username, incorrect password	username=wrong, password=wrongpass	Error
5	Correct username, password (User)	username=user, password=user123	Open User Dashboard
6	Correct username, password (New User)	username=newuser, password=newpass	Create New User Account

Table 5. 2 Login Test Case

3. Home Page

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Carousel Slider Display	-	Slider should rotate smoothly
2	Event List Display	-	List should be populated with events
3	Gift Category Cards Display	-	Cards should be correctly displayed

Table 5. 3 Home Page Test Case

4. Product Section

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	No Quantity Set	0	No Action
2	Quantity Set to Negative Number	-10	Error
3	Quantity Set Greater Than Stock	1000	Warning and Stop Checkout
4	Valid Quantity Set	3	Proceed to Checkout

Table 5. 4 Product Section Test Case

5. Cart and Checkout

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	No Item in cart	-	Display No Item in cart
2	Add Item to Cart	Select product	Item added to cart
3	Remove Item from Cart	Select remove option	Item removed from cart
4	Proceed to Checkout	Click checkout	Move to checkout page

Table 5. 5 Cart and Checkout Test Case

6. Payment Gateway

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Blank Payment Details	-	Error
2	Invalid Payment Details	Incorrect card/ UPI/ Netbanking info	Error

3	Valid Payment Details	Correct card/ UPI/ Netbanking info	Payment Successful
4	RazorPay Payment Failure	Payment incomplete	Back to checkout page
5	RazorPay Payment Success	Correct payment details and payment done	Order Confirmed

Table 5. 6 Payment Gateway Test Case

7. User Profile

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Edit Profile Details	Modify user details	Details should be updated
2	View Order History	Click on orders	Order history should be displayed
3	View Wishlist	Click on Wishlist	Wishlist should be displayed
4	View Reminders	Click on Reminders	Reminders will be displayed

Table 5. 7 User Profile Test Case

8. Search Functionality

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Search for Existing Product	Valid product name	Correct results displayed
2	Search for Non-Existing Product	Invalid name	No results found

Table 5. 8 Search Functionality Test Case

9. Reminder Feature

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT
1	Set Reminder	Set valid reminder	Reminder saved
2	Invalid Reminder Format	Incorrect date/time	Error
3	Trigger Reminder	Time reached	Notification displayed

Table 5. 9 Reminder Feature Test Case

CHAPTER 6

RESULT AND DISCUSSIONS

6.1. Test Reports

1. Authentication Module

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Blank Auth Key	-	Error	Same as Expected	Pass
2	Invalid Auth Key	invalid_key	Error	Same as Expected	Pass
3	Valid Auth Key	correct_key	Proceed to Homepage	Same as Expected	Pass

Table 6. 1 Authentication Module Test Report

2. Login

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Blank username, password	uname=, password=	Error	Same as Expected	Pass
2	Blank username, correct password	uname=, password=valid	Error	Same as Expected	Pass
3	Correct username, blank password	uname=valid_user, password=	Error	Same as Expected	Pass
4	Incorrect username, incorrect password	uname=wrong, password=wrongpass	Error	Same as Expected	Pass
5	Correct username, password	uname=user, password=user123	Open User Dashboard	Same as Expected	Pass
6	Correct username, password (New User)	uname=newuser ,password=newpass	Create New User Account	Same as Expected	Pass

Table 6. 2 Login Test Report

3. Home Page

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Carousel Slider Display	-	Slider should rotate smoothly	Same as Expected	Pass
2	Event List Display	-	Should display events	Same as Expected	Pass
3	Gift Category Cards Display	-	Should be correctly displayed	Same as Expected	Pass

Table 6. 3 Home Page Test Report

4. Product Section

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	No Quantity Set	0	No Action	Same as Expected	Pass
2	Quantity Set to Negative Number	-10	Error	Same as Expected	Pass
3	Quantity Set Greater Than Stock	1000	Warning and Stop Checkout	Same as Expected	Pass
4	Valid Quantity Set	3	Proceed to Checkout	Same as Expected	Pass

Table 6. 4 Product Section Test Report

5. Cart and Checkout

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	No Item in cart	-	Display No Item in cart	Same as Expected	Pass
2	Add Item to Cart	Select product	Item added to cart	Same as Expected	Pass
3	Remove Item from Cart	Select remove option	Item removed from cart	Same as Expected	Pass
4	Proceed to Checkout	Click checkout	Move to payment page	Same as Expected	Pass

Table 6. 5 Cart and Checkout Test Report

6. Payment Gateway

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Blank Payment Details	-	Error	Same as Expected	Pass
2	Invalid Payment Details	Incorrect payment info	Error	Same as Expected	Pass
3	Valid Payment Details	Correct payment info	Payment Successful	Same as Expected	Pass
4	RazorPay Payment Failure	Payment incomplete	Back to checkout page	Same as Expected	Pass
5	RazorPay Payment Success	Payment done	Order Confirmed	Same as Expected	Pass

Table 6. 6 Payment Gateway Test Report

7. User Profile

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Edit Profile Details	Modify user details	Details should be updated	Same as Expected	Pass
2	View Order History	Click on orders	History should be displayed	Same as Expected	Pass
3	View Wishlist	Click on Wishlist	Wishlist should be displayed	Same as Expected	Pass
4	View Reminders	Click on Reminders	Reminders will be displayed	Same as Expected	Pass

Table 6. 7 User Profile Test Report

8. Search Functionality

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Search for Existing Product	Valid product name	Correct results displayed	Same as Expected	Pass
2	Search for Non-Existing Product	Invalid name	No results found	Same as Expected	Pass

Table 6. 8 Search Functionality Test Report

9. Reminder Feature

SR NO	TEST CASE DESCRIPTION	TEST VALUES	EXPECTED RESULT	ACTUAL RESULT	TEST CASE RESULT
1	Set Reminder	Set valid reminder	Reminder saved	Same as Expected	Pass
2	Invalid Reminder Format	Incorrect date/time	Error	Same as Expected	Pass
3	Trigger Reminder	Time reached	Notification displayed	Same as Expected	Pass

Table 6. 9 Reminder Feature Test Case

6.2. Screenshots

1. Login Screen

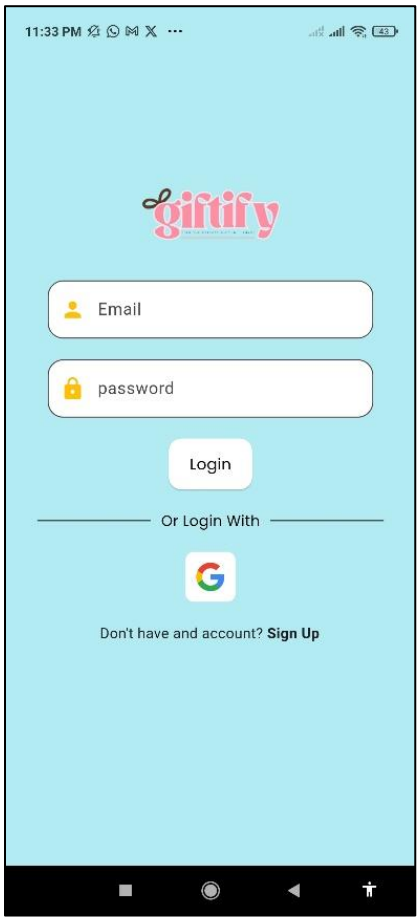


Figure 6. 1 Login Screen

2. Sign up Page

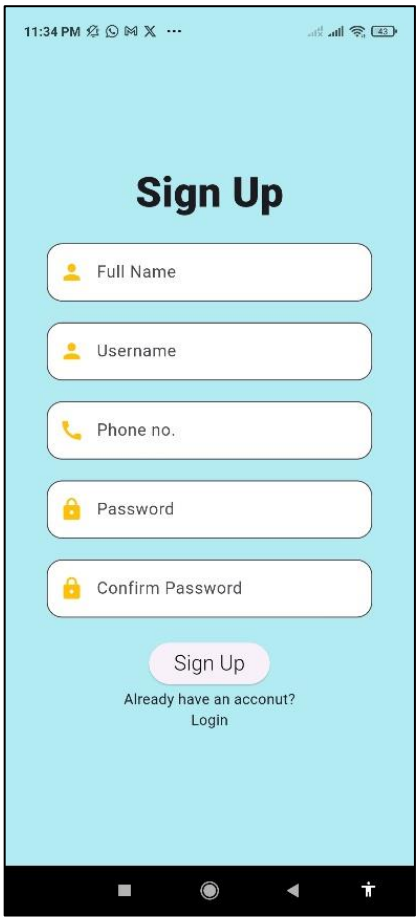


Figure 6. 2 Sign Up Page

3. Home Page

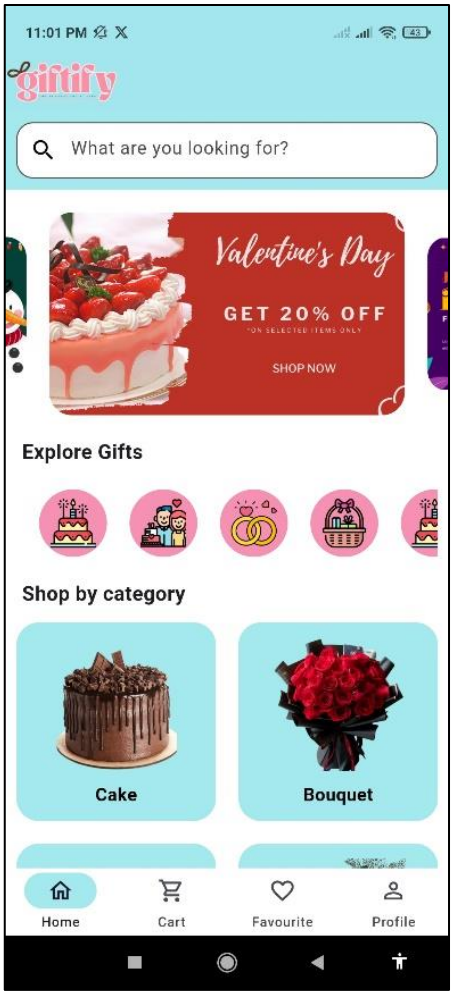


Figure 6. 3 Home Page

4. Product Page

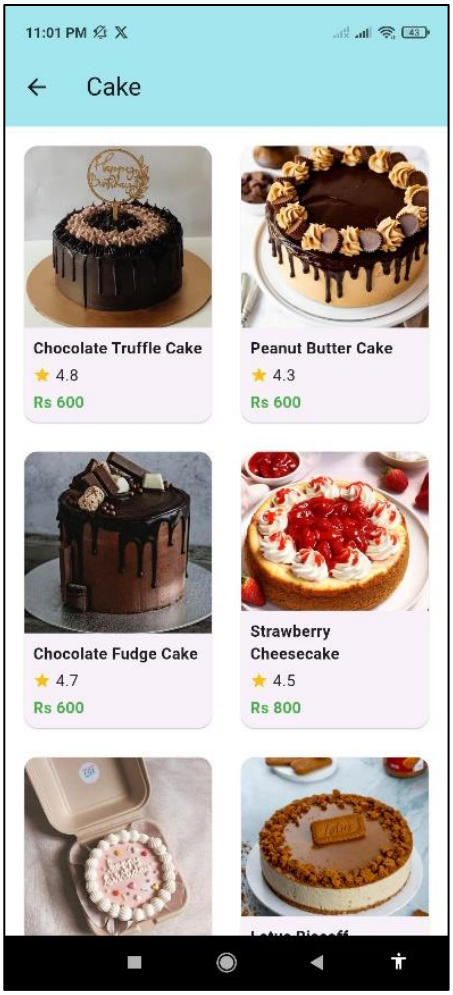


Figure 6. 4 Product Page

5. Product Detail Page

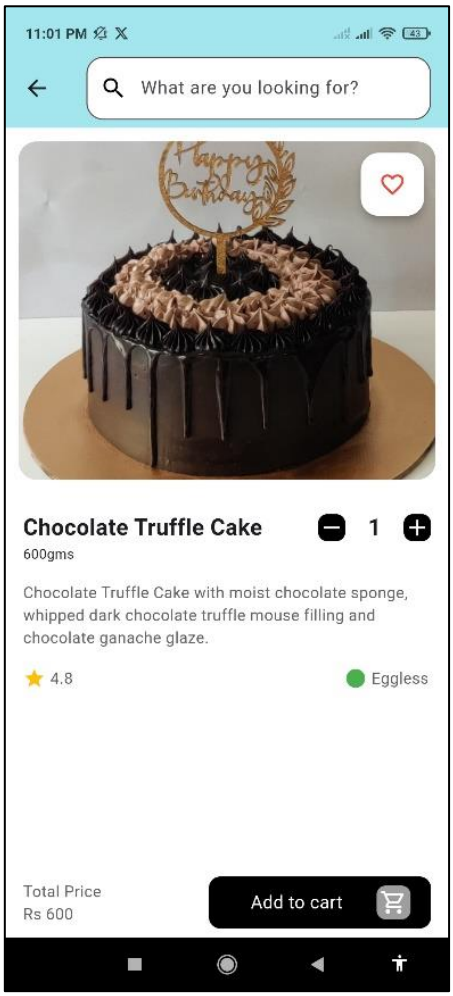


Figure 6. 5 Product Detail Page

6. Wishlist

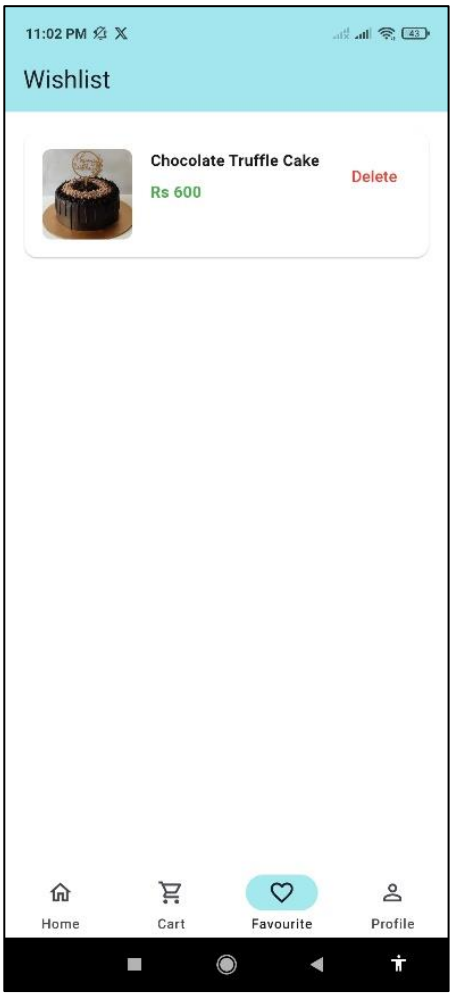


Figure 6. 6 Wishlist

7. Cart

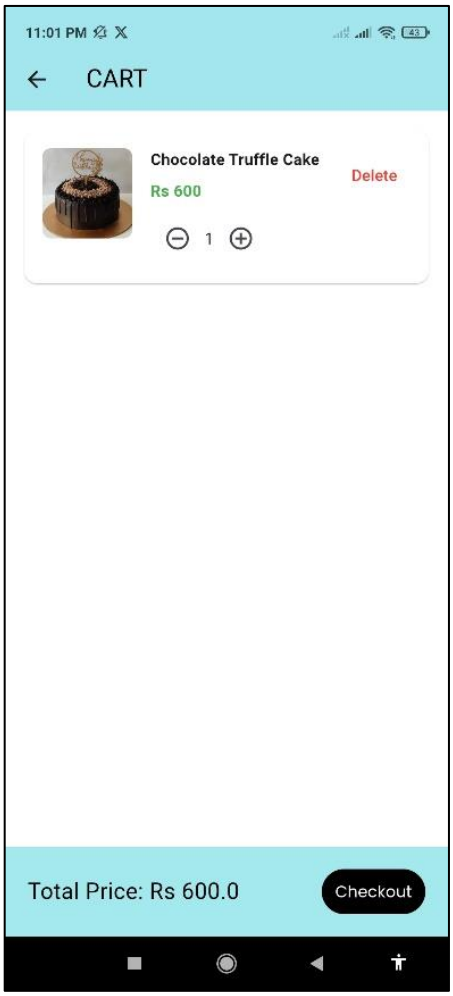


Figure 6. 7 Cart

8. Address Details

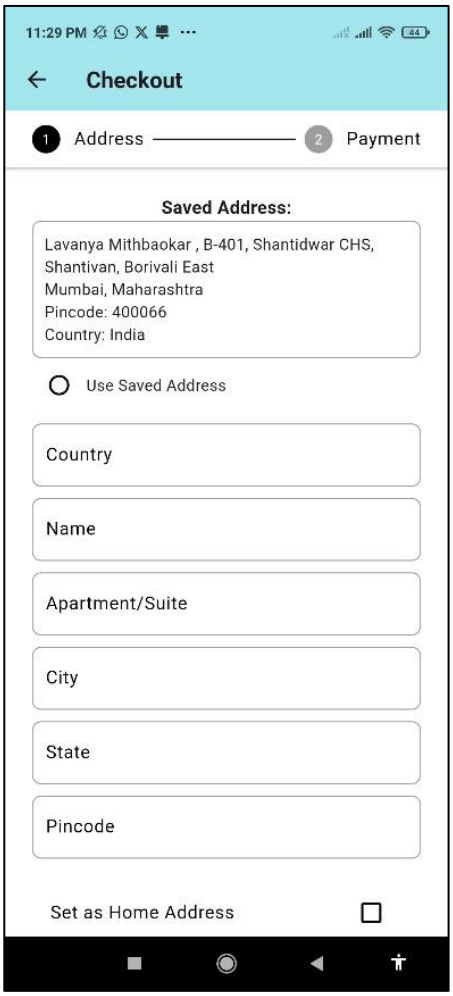


Figure 6. 8 Address Details

9. Payment Gateway

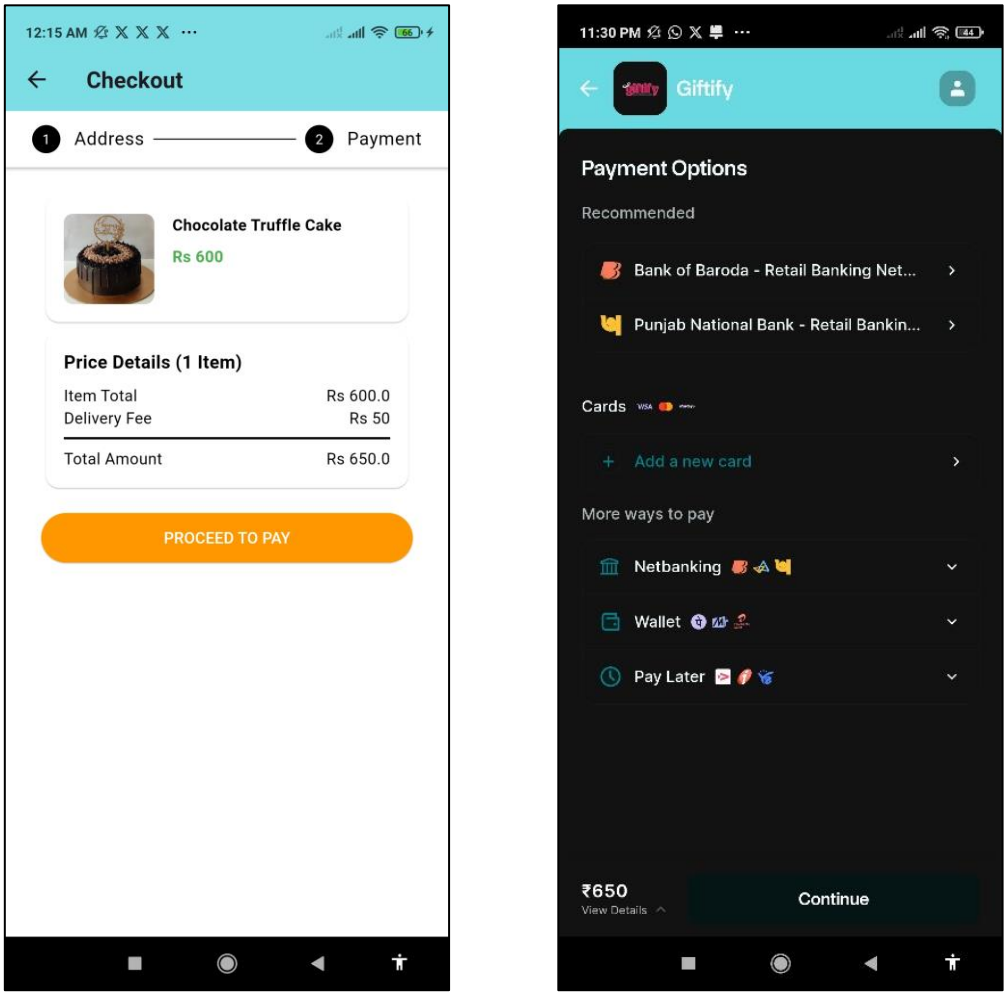


Figure 6. 9 Payment Gateway

10. Order Confirmation

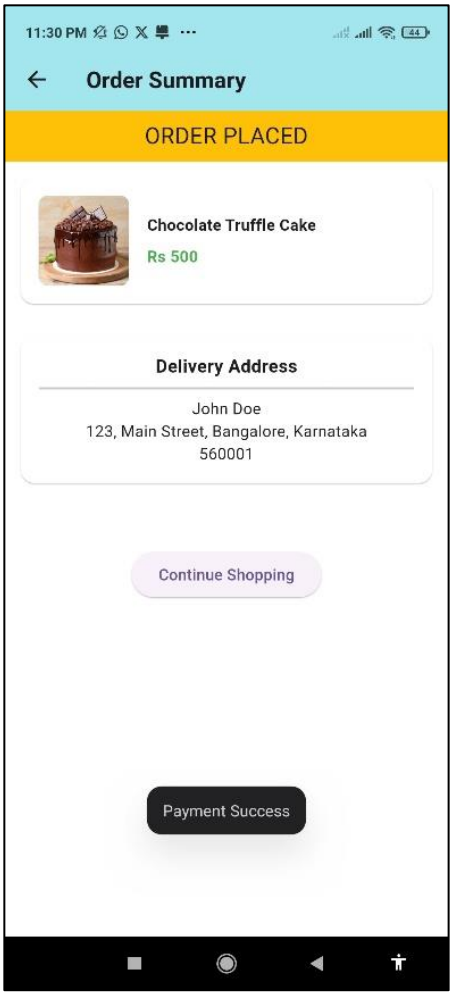


Figure 6. 10 Order Confirmation

11. Search Feature

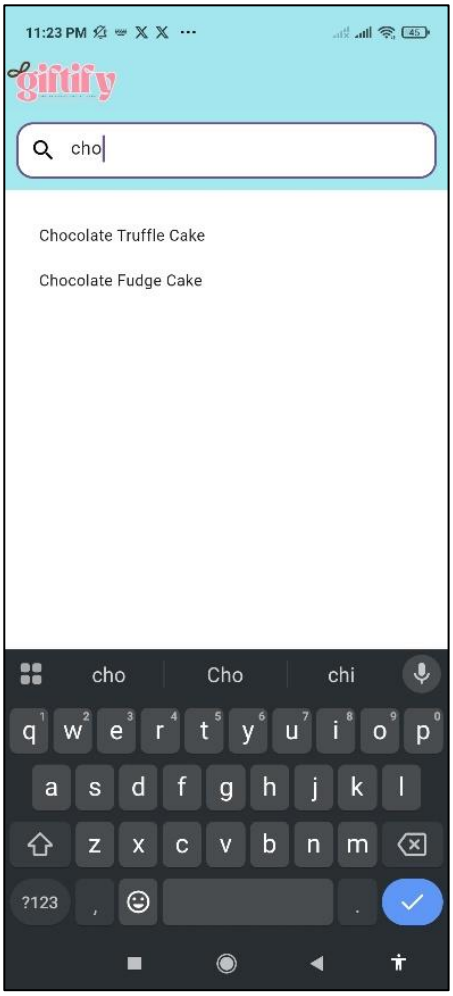


Figure 6. 11 Search Feature

12. Profile

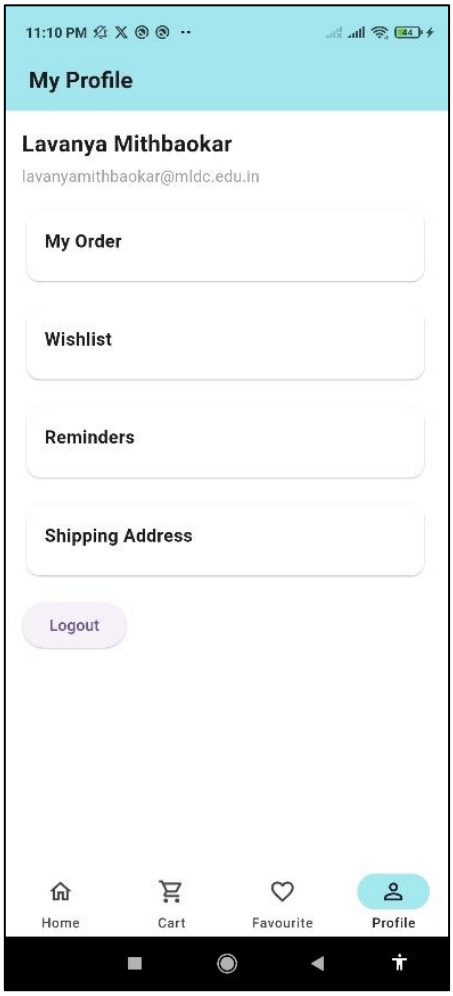


Figure 6. 12 Profile Page

13. My Orders

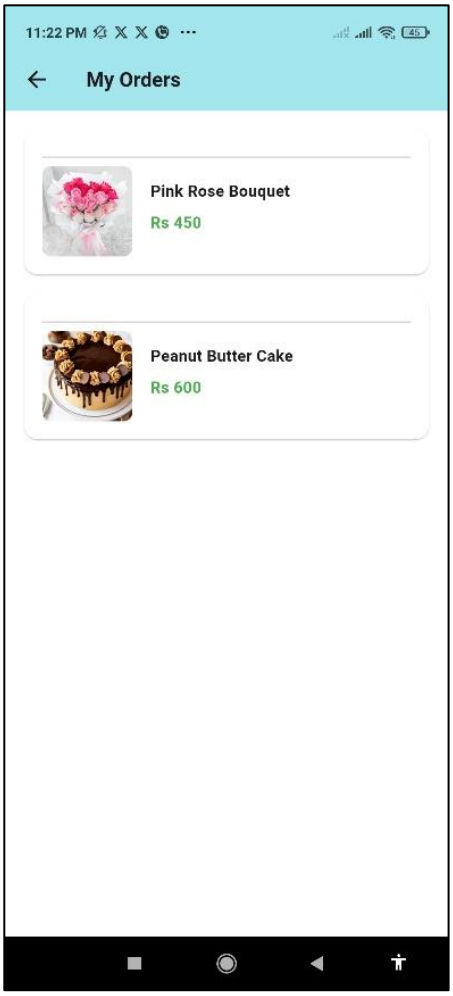


Figure 6. 13 My Orders

14. Reminders

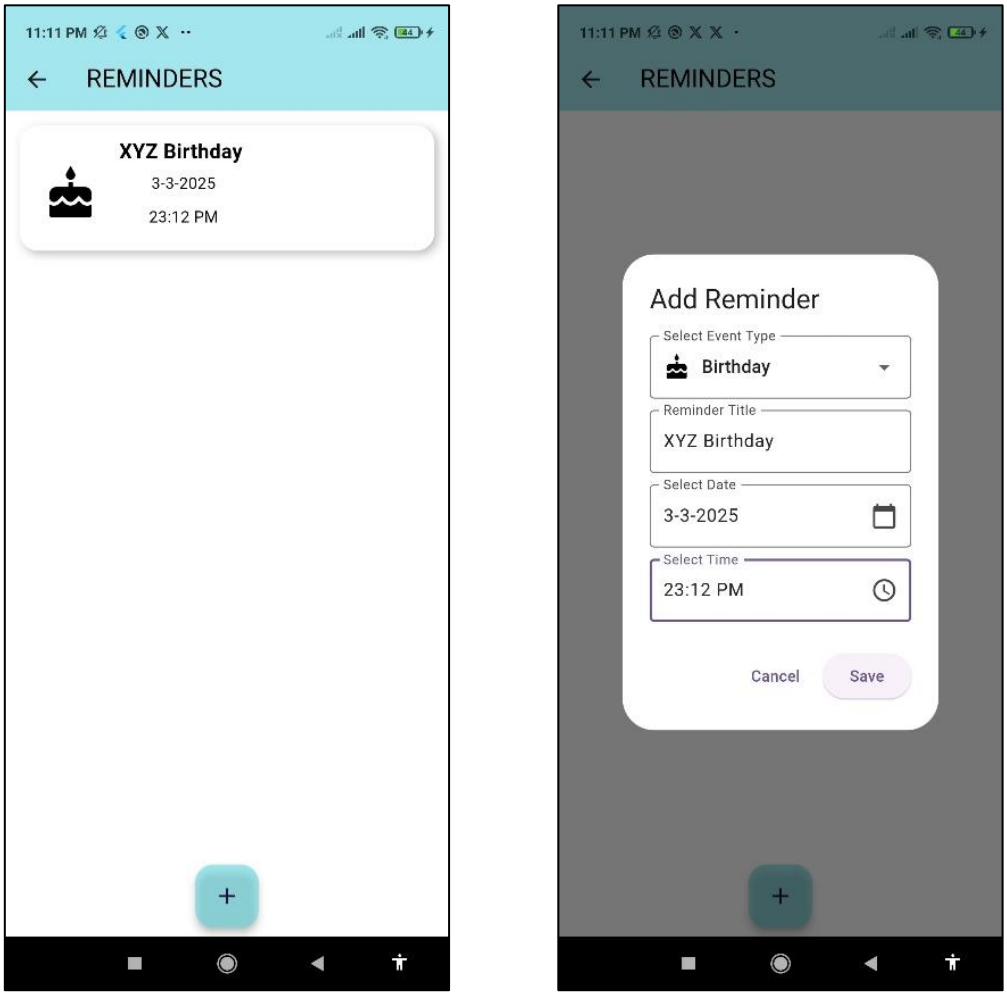


Figure 6. 14 Reminders

CHAPTER 7

CONCLUSION

7.1 Conclusion

- Giftify aims to simplify the gifting process.
- It will significantly reduce the time users spend searching for gifts.
- It will enhance user convenience, allowing them to browse and compare options efficiently.
- It enhances user satisfaction and ensures that every gift chosen is meaningful.
- The app intends to become the go-to platform for thoughtful and memorable gift-giving occasions.

7.2 Limitations

- **No Personalized Gifts:** Users prefer personalization of gifts (e.g., engraved items, custom messages, photo-printed gifts) which is not currently available.
- **Lack of customizable Reminders:** Currently, users cannot set personalized reminders for upcoming payments or product restocks.
- **No Web Version available:** The app is only accessible via mobile, restricting users who prefer browsing and shopping on larger screens.
- **Multi-Vendor Support:** The platform currently does not allow multiple sellers to list their products. This limits product variety and seller competition, which could improve pricing and quality.
- **No Multi-Language & Multi-Currency Support:** The app does not support multiple languages or currencies, making it less accessible for international users.

7.3 Future Scope

- **Personalized Gifts:** Introduce customization options like engraving, custom messages, and photo-printed gifts.
- **Customizable Reminders:** Allow users to set reminders for upcoming payments or product restocks.
- **Web Version of the App:** Expand beyond mobile by offering a web-based platform.
- **Multi-Vendor Support:** Allow different sellers to list products.
- **Multi-Language & Multi-Currency Support:** Make the app accessible worldwide.

SUMMARY

The main aim of this project is to simplify the gifting process by helping users to quickly find the perfect gift and get it delivered to their doorstep effortlessly. The app eliminates the hassle of manually searching for gifts by providing a smooth shopping experience.

This app has been implemented using Flutter version 3.27.1 and Firebase for backend services and authentication.

Firebase is an excellent choice as it provides a real-time, scalable, and secure backend without requiring complex server management. It enables seamless authentication, real-time order updates, and cloud storage for product images. With Firebase Cloud Messaging (FCM), users receive instant notifications for orders and reminders.

GLOSSARY

GUI- Graphical User Interface

API- Application Programming Interface

DOB- Date of Birth

INFO- Information

AUTH- Authentication

UNAME- Username

UI- User Interface

UX- User Experience

REFERENCES

https://dribbble.com/

https://pub.dev/

https://www.geeksforgeeks.org/flutter-stepper-widget/

https://firebase.google.com/codelabs/firebase-fcm-flutter#0

https://lottiefiles.com/blog/tips-and-tutorials/add-lottie-animations-websites-mobile-app-design-mockups

https://www.geeksforgeeks.org/flutter-schedule-local-notification-using-timezone/

https://www.geeksforgeeks.org/flutter-custom-bottom-navigation-bar/

https://www.geeksforgeeks.org/flutter-read-and-write-data-on-firebase/