**BeauteCart**

**[Online Cosmetics Shop]**

*Mini Project Report*

*Submitted by*

**NANDANA R S**

**Reg. No.: AJC22MCA-2066**

*In Partial fulfillment for the Award of the Degree of*

**MASTER OF COMPUTER APPLICATIONS**

**(MCA TWO YEAR)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING**

**KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with ‘A’ grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

# 2023-2024

## DEPARTMENT OF COMPUTER APPLICATIONS

### AMAL JYOTHI COLLEGE OF ENGINEERING

**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**BEAUTECART”** is the bona fide work of **NANDANA RS(Regno: AJC22MCA-2066)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

**Dr. Bijimol TK Ms. Meera Rose Mathew**

**Internal Guide Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**

**Head of the Department**

**DECLARATION**

I hereby declare that the project report **“BEAUTECART”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

**Date: 30/11/2023 NANDANA R S**

**KANJIRAPPALLY Reg: AJC22MCA-2066**

# ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator  **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Dr. Bijimol TK** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

NANDANA R S

# ABSTRACT

The "Online Cosmetics Shopping" project is a comprehensive web-based platform designed

to provide users with an immersive and convenient online shopping experience for cosmetics

and beauty products . This project aims to provide a sophisticated yet user-friendly interface

for cosmetics enthusiasts to discover, explore, and purchase beauty products online. By

presenting a curated collection of products and incorporating user-centric features, the

platform seeks to redefine the cosmetics shopping experience in the digital age. It has various categories of beauty products that are available for shopping. The project has two modules: user ,sellers and admin. Unregistered users can view the website and check out the information about the parlour. There must be a login for admin , sellers and users. Admin can see the details of the above mentioned .Sellers add product. By using Artificial Intelligence application in terms of beauty products the site creates a Augmented reality enhances the online shopping experience with the ability to try lipstick and eye shadows virtually. The aim of this project is to create generic software that can be used by any company. It also offers its customers convenience. In addition, the program can include a large amount of summary data. It is created using Python Django Framework on the backend and HTML,CSS, and JavaScript on the frontend.

**CONTENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL. NO** | | **TOPIC** | **PAGE NO** | |
| **1** | | **INTRODUCTION** |  | |
| **1.1** | | **PROJECT OVERVIEW** |  | |
| **1.2** | | **PROJECT SPECIFICATION** |  | |
| **2** | | **SYSTEM STUDY** |  | |
| **2.1** | | **INTRODUCTION** |  | |
| **2.2** | | **EXISTING SYSTEM** |  | |
| **2.3** | | **DRAWBACKS OF EXISTING SYSTEM** |  | |
| **2.4** | | **PROPOSED SYSTEM** |  | |
| **2.5** | | **ADVANTAGES OF PROPOSED SYSTEM** |  | |
| **3** | | **REQUIREMENT ANALYSIS** |  | |
| **3.1** | | **FEASIBILITY STUDY** |  | |
| **3.1.1** | | **ECONOMICAL FEASIBILITY** |  | |
| **3.1.2** | | **TECHNICAL FEASIBILITY** |  | |
| **3.1.3** | | **BEHAVIORAL FEASIBILITY** |  | |
| **3.1.4** | | **FEASIBILITY STUDY QUESTIONNAIRE** |  | |
| **3.2** | | **SYSTEM SPECIFICATION** |  | |
| **3.2.1** | | **HARDWARE SPECIFICATION** |  | |
| **3.2.2** | | **SOFTWARE SPECIFICATION** |  | |
| **3.3** | | **SOFTWARE DESCRIPTION** |  | |
| **3.3.1** | | **PHP** |  | |
| **3.3.2** | | **MYSQL** |  | |
| **4** | | **SYSTEM DESIGN** |  | |
| **4.1** | | **INTRODUCTION** |  | |
| **4.2** | | **UML DIAGRAM** |  | |
| **4.2.1** | | **USE CASE DIAGRAM** |  | |
| **4.2.2** | | **SEQUENCE DIAGRAM** |  | |
| **4.2.3** | | **STATE CHART DIAGRAM** |  | |
| **4.2.4** | | **ACTIVITY DIAGRAM** |  | |
| **4.2.5** | | **CLASS DIAGRAM** |  | |
| **4.2.6** | | **OBJECT DIAGRAM** |  | |
| **4.2.7** | | **COMPONENT DIAGRAM** |  | |
| **4.2.8** | | **DEPLOYMENT DIAGRAM** |  | |
| **4.2.9** | | **COLLABORATION DIAGRAM** |  | |
| **4.3** | | **USER INTERFACE DESIGN USING FIGMA** |  | |
| **4.4** | | **DATABASE DESIGN** |  | |
| **5** | | **SYSTEM TESTING** |  | |
| **5.1** | | **INTRODUCTION** |  | |
| **5.2** | | **TEST PLAN** |  | |
| **5.2.1** | **UNIT TESTING** | |  |
| **5.2.2** | **INTEGRATION TESTING** | |  |
| **5.2.3** | **VALIDATION TESTING** | |  |
| **5.2.4** | **USER ACCEPTANCE TESTING** | |  |
| **5.2.5** | **AUTOMATION TESTING** | |  |
| **5.2.6** | **SELENIUM TESTING** | |  |
| **6** | **IMPLEMENTATION** | |  |
| **6.1** | **INTRODUCTION** | |  |
| **6.2** | **IMPLEMENTATION PROCEDURE** | |  |
| **6.2.1** | **USER TRAINING** | |  |
| **6.2.2** | **TRAINING ON APPLICATION SOFTWARE** | |  |
| **6.2.3** | **SYSTEM MAINTENANCE** | |  |
| **7** | **CONCLUSION & FUTURE SCOPE** | |  |
| **7.1** | **CONCLUSION** | |  |
| **7.2** | **FUTURE SCOPE** | |  |
| **8** | **BIBLIOGRAPHY** | |  |
| **9** | **APPENDIX** | |  |
| **9.1** | **SAMPLE CODE** | |  |
| **9.2** | **SCREEN SHOTS** | |  |

## List of Abbreviation

IDE - Integrated Development Environment

HTML - Hyper Text Markup Language.

CSS - Cascading Style Sheet

SQL - Structured Query Language

UML - Unified Modelling Language

JS - JavaScript

# CHAPTER 1

# INTRODUCTION

### PROJECT OVERVIEW

The "Online Cosmetics Shopping" project is a comprehensive web-based platform aiming to redefine the digital cosmetics shopping experience. It features a sophisticated yet user-friendly interface for enthusiasts to explore and purchase beauty products. Divided into mini and main projects, the former focuses on essential modules such as Admin, Users, and Cosmetics Sellers. After successful payment of the purchased product, user can track the orders. Administration of the system includes managing services, bookings, products, stocks and customers. In contrast, the latter introduces innovative elements like Virtual Try-On, Order Tracking, Packing Management, Beauty Consultation, Chatbot integration, and Sustainability Programs. Leveraging HTML, CSS, JavaScript for the front end, Python for the back end, Django as the framework, and MySQL as the database, the platform not only provides secure admin access, brand exploration, and user registration but also incorporates advanced features to enhance user engagement, streamline operations, and promote sustainable practices.

### PROJECT SPECIFICATION

The proposed system is an online cosmetics shop providing system. The system manages a variety of services that are displayed virtually by using a customer interface where it lets them to select their desired cosmetics products by filtering.. And the customers can view and buy beauty products offered by the beauty parlour administrator. Customers can track their order through the system .The system includes 3 actors. They are:

**1. Admin**

Admin has the overall control of the system. He/She administrates and enables the proper

functioning of the system. Admins functionality is to manage the whole website.

**2. Seller**

Registered Sellers add products to the website.

**3. User**

Users can view the system through an interface .They can view their profile,. Users can buy beauty products based on different category and they can pay money for purchasing products through online. And also, user can write review and rate the products.

# CHAPTER 2

# SYSTEM STUDY

### INTRODUCTION

Finding and analysing data, diagnosing problems, and applying data for any new system

suggestions are all parts of system analysis. During this troubleshooting procedure, there must be

substantial communication between system users and system developers. The system must be

analysed or researched during the system development process. The system is then efficiently

checked and evaluated. System analyst has the role of an investigator and crosschecks how the

current system works. For understanding the problem, important and relevant variables should be

identified, and each element should be evaluated and synthesized and thus find the best. Through

methodologies questionnaires and interviews the system is extensively studied. To draw a

conclusion, the information gathered needs to be carefully reviewed. In conclusion, it entails being aware of how the system functions. The system is known as the current system, and this system is then carefully evaluated to identify any problems that need to be fixed. The solutions are replaced with proposals. The best one is chosen from the proposals analytically compared to the current system. The user of the system can either approve or reject the suggestion. The preliminary study refers to the process of interpreting and gathering facts for preceding studies of the system.

### EXISTING SYSTEM

The existing method has data stores dispersed throughout the system infrastructure and

necessitates several paper forms. On forms, information is typically wrong or doesn't follow

management guidelines. There are chances for records frequently vanishing during the

computation, thus a strict auditing system is required to make sure that no important data is lost

during the period. The online marketplace offers several services that are listed under various

categories and thus resulting in a non-coordinated way of processing and does not provide users

an opportunity to book services that are relevant. Purchasing beauty products online can be

challenging for customers, as they are unable to physically try on products before making a

purchase.

**2.2.1 NATURAL SYSTEM STUDIED**

In the system studied people are directly concerned to Beauty Parlours or by some other way in order to book bridal makeup . It requires more time .This proposed system helps to overcome the disadvantages and helps the users to select the bridal makeup look and other accessories as per their requirements.

**2.2.2 DESIGNED SYSTEM STUDIED**

In the designed system, the designed system manages all the data online. So, the data become safe and secure, and easy to manage. It offers an online Bridal makeup booking system and online shopping of beauty products. So that customers take their appointment online and wastage of time can be reduced. And it offers shopping for beauty products with discounts. Customers can buy products at cheap rates and also it provides a try try-before-buy. After successful payment of the purchased product, the user can track the orders. Administration of the system includes managing services,bookings, products, stocks, and customers.

### DRAWBACKS OF EXISTING SYSTEM

* They provide a less-interactive user environment.
* Unable to physically try on products before making a purchase.
* The usage of less user-friendly or traditional interfaces that are hard to use with.
* Absence of proper online management.
* Requires human effort.
* Absence of Bridal makeup booking schedule
* Wastage of time

### PROPOSED SYSTEM

The proposed system, "Online Cosmetics Shopping," is an ambitious web-based platform set to redefine the digital cosmetics shopping experience. It encompasses an extensive range of functionalities aimed at providing users with a seamless and user-friendly interface for discovering, exploring, and purchasing beauty products online. The system is divided into several key modules. The Admin Module caters to administrators, offering secure login, password reset, and a comprehensive dashboard displaying crucial statistics. It also includes features for efficient product, user, and order management, as well as streamlining packaging processes. The Users Module focuses on customer interactions, allowing for hassle-free account creation, secure login, shopping cart management, and order tracking. Additionally, users can update their profile information, leave product reviews, and seamlessly process payments.

Furthermore, the system introduces a dedicated Cosmetics Brand Module, offering brand listings, brand pages, product showcases, and brand rankings. This module provides users with a curated selection of renowned cosmetic brands, enhancing their shopping experience. The proposed system also includes a Packing Management Module to ensure products are accurately and efficiently handled before shipping. With the integration of these modules, the "Online Cosmetics Shopping" system aspires to be a comprehensive platform that caters to the needs of both administrators and customers, ultimately elevating the online cosmetics shopping experience to new heights.

### ADVANTAGES OF PROPOSED SYSTEM

* Convenience and Accessibility
* Wide Product Range
* User-Friendly Interface
* Efficient Order Management
* Product Reviews and Ratings
* Instant Assistance with Chatbot
* Sustainable Practices

# CHAPTER 3

# REQUIREMENT ANALYSIS

## FEASIBILITY STUDY

Planning, organizing, and managing resources to ensure the achievement of particular project goals band objectives is the process of project management. A feasibility study is a preliminary

examination of a prospective project or end to determine its merits and viability. A feasibility study aims to provide an objective assessment of the technical, economic, financial, legal, and

environmental elements of a proposed project. The information can then be used by decisionmakers to decide whether to proceed with the project or not. The findings of the feasibility study can also be used to develop a practical project plan and budget. It cannot be simple to determine whether or not a proposed project is worthwhile pursuing without a feasibility study. The document provides the feasibility of the project that is being designed and lists. Various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibility. The following are its features:

### Economical Feasibility

Cost and benefit analyses are required to support the developing system. criteria to make sure that focus is on the project that will yield the best results and return the earliest. The price that would be involved in developing a new system is one of the variables. Some significant financial queries raised during the initial investigation include the following:

• The costs conduct a full system investigation?

➢ The proposed system is developed as part of project work, there is no manual cost to spend for

the proposed system.

• The cost of the hardware and software?

➢ Also all the resources are already available.

### Technical Feasibility

The system needs to be assessed first from a technical standpoint. The outline design of the system requirement in terms of input, output, programs, and procedures must serve as the foundation for the assessment of this feasibility. After determining an outline investigation must continue to identify the necessary equipment kind. Once the system has been designed, there are several ways to run it.

• Is the project feasible within the limits of current technology.

➢ YES

• Technical issues raised during the investigation are:

➢ NOTHING

• Can the technology be easily applied to current problems?

➢ YES

• Does the technology have the capacity to handle the solution?

➢ YES

### Behavioral Feasibility

The proposed system includes the following questions:

• Is there sufficient support for the users

➢ YES

• Will the proposed system cause harm?

➢ NO

The project would be beneficial because it satisfies the objectives when developed and installed.

All behavioral aspects are considered carefully and conclude that the project is behaviorally

Feasible

**3.1.4 Feasibility Study Questionnaire**

1. How is task allotted to handle the functioning of your electronic shop?

Task is divided among different sections like Accountant, cashier,

staffs, stock clerk,cleaners.

2. How is it made user friendly?

If a customer visit the shop for first time his crentials are feeded on tosystem so that they get e-bill, get notified on offers .

3. How is stock managed?

Reports are kept for each sold products and will refill out of stock products.

4. At what time do you open the store?

Around 7:30 am, since to arrange products

5. Does staff help out customers to find a relevant product?

Staff will just help to know about systems specification and they donot encourage to buy a particular product ,according to marketing strategy

6. How do you attract customers?

By keeping offers.

7. How is payment system managed?

Either as cash or online transaction (g pay phone pay)

8. What are the extra expenditures?

Maintainence cost , Fuel cost ,Electricity bill.

9. Is it government organized system?

It is organized by both public and private sector.

10. Is there any factors that affect sales?

Yes, public holidays, harthals and mainly area where shop is situated is a factor for natural

disaster like flood

## SYSTEM SPECIFICATION

### Hardware Specification

Processor - Intel core i5

RAM - 8 GB

Hard disk - 1 TB HDD

### Software Specification

Front End - HTML /CSS/JAVASCRIPT

Back End - Python Django

Database -SqLite

Client on PC - Windows 11

Technologies used - JS, HTML5. Python Django, CSS

## SOFTWARE DESCRIPTION

### PYTHON DJANGO

A well-known Python-based framework for web development is called Django. A high-level web framework, it enables the fast creation of secure and maintainable websites. Because it is an open-source framework, it is also free to use. Django also adheres to the Model-View-Controller (MVC)design, which is currently the norm for creating online applications. Additionally, it offers a thriving community, first-rate documentation, and a selection of both free and commercial support services. Django was initially created to create online applications for the Lawrence Journal- World, a media firm. As a result, it excels at managing projects with a lot of media files, text content, and traffic. This architecture can be used outside of the publishing industry, though. This architecture can be used outside of the publishing industry, though. Django gains additional functionality with each new release, making it effective enough to create any kind of web application. Any kind of online application, from an e-commerce site to a social media platform, may be created with Django. Django provides a wide range of capabilities that streamline andaccelerate the development process. It integrates a variety of features to create a comprehensive framework.

### SQLite

SQLite is a lightweight and self-contained relational database management system (RDBMS) widely used for its simplicity and ease of integration. One of its significant advantages lies in its simplicity and minimal configuration requirements, making it an ideal choice for small to medium-sized applications. As a serverless, file-based database, SQLite operates directly from a single local file, eliminating the need for a separate database server and reducing system overhead. This feature makes it particularly suitable for embedded systems, mobile applications, and scenarios where a compact, efficient database solution is preferred. Moreover, SQLite is ACID-compliant (Atomicity, Consistency, Isolation, Durability), ensuring the reliability and integrity of data transactions.

When integrated with Django, a high-level Python web framework, SQLite becomes a convenient and default choice for development. Django seamlessly supports SQLite as one of its database backends, allowing developers to quickly set up databases for their projects without complex configuration. This integration enhances Django's "batteries-included" philosophy, providing a

straightforward database solution for prototyping, testing, and deploying web applications. While SQLite may not be the optimal choice for large-scale or production-grade applications with high concurrency and heavy data processing requirements, its compatibility with Django makes it an excellent choice for smaller projects or scenarios where simplicity and rapid development are prioritized. The integration between Django and SQLite underscores the flexibility and versatility of both technologies, empowering developers to build web applications efficiently. Another noteworthy advantage of SQLite in its integration with Django is its seamless compatibility with version control systems. The single-file architecture of SQLite databases simplifies versioning and deployment processes, making it easier for developers to manage changes across different development stages. This compatibility ensures that database schema changes and updates can be smoothly tracked and implemented alongside the codebase within version control systems like Git. This cohesion streamlines collaborative development efforts, enhances team coordination, and facilitates the sharing of project updates. SQLite's straightforward integration with Django, coupled with its version control friendliness, reinforces its utility as a versatile database solution within the Django framework, catering to the needs of developers engaged in agile and collaborative web application development.

# CHAPTER 4

# SYSTEM DESIGN

* 1. **INTRODUCTION**

Any technical system or product is first created through the design phase. A creative process is

design. A system's success depends on a good design. Design is the process of outlining a system

or process in enough detail to allow for its practical execution while making use of a variety of

techniques and ideas. In order to make a device, technique, or system practical, it involves using a

variety of methodologies and concepts to describe it in detail. Regardless of the development

model used, software design is at the core of the software engineering process. The architectural

detail required to build a system or product is generated via system design. Our software goes

through the best design process possible, polishing every area of efficiency, performance, and

accuracy levels, much like any exacting method. A user-focused document becomes a document

for programmers or database specialists during the design phase. The two phases that make up the

system design development process are physical design and logical design.

## UML DIAGRAM

To specify, visualize, construct, and record software system artifacts, a standard language known

as UML is utilized. The Object Management Group (OMG), which was in charge of developing

UML, received a draughts of the UML 1.0 definition in January 1997. The acronym UML stands

for Unified Modelling Language. Unlike other well-known programming languages like C++, Java,and COBOL, UML has some unique features. Software designs are created using a visual language called UML. UML is a general-purpose visual modelling language that can be used for software system visualisation, definition, development, and documentation. Despite being the most popular application, UML is used for more than only defining software systems. Simulating non-software systems is another use for it. Think about the manufacturing facility's process flows as an illustration. Although UML is not a programming language, tools based on UML diagrams can be used to generate code in a variety of languages. UML is intimately tied to the analysis and design of object-oriented systems. UML has evolved to the point that it is now an OMG standard. A comprehensive UML diagram that depicts a system includes all the parts and connections. The

UML diagram's appearance is the most important component of the complete approach. It is

finished once all remaining components are included.

• Use case diagram

• Sequence diagram

• State chart diagram

• Activity diagram

• Class diagram

• Object diagram

• Component diagram

• Deployment diagram

## USE CASE DIAGRAM

A use case diagram is a visual representation of the relationships between system components. A use case is a method for locating, defining, and organising system needs. Here, the term UML (Unified Modelling Language) refers to something that is being built or operated, such as a website for mail-order service and product sales. Use case diagrams are used in UML (Unified Modelling Language), a standard language for modelling actual items and systems. System objectives include the formulation of general requirements, the verification of a hardware design, the testing and debugging of an under development software product, the production of an online help reference, and the execution of a task focused on customer service. Customer support, item ordering, cataloge updating, and payment processing are examples of use cases in a product sales scenario. A use case diagram has four components.

• The boundary separating the studied system from its surroundings.

• The actors, who are frequently system players who are recognised by their roles.

• The use cases, which are the specialised roles, are performed by the actors within and around the system.

• The relationships and exchanges between the use cases and the actors. Using use case diagrams, a system functional requirement is represented.

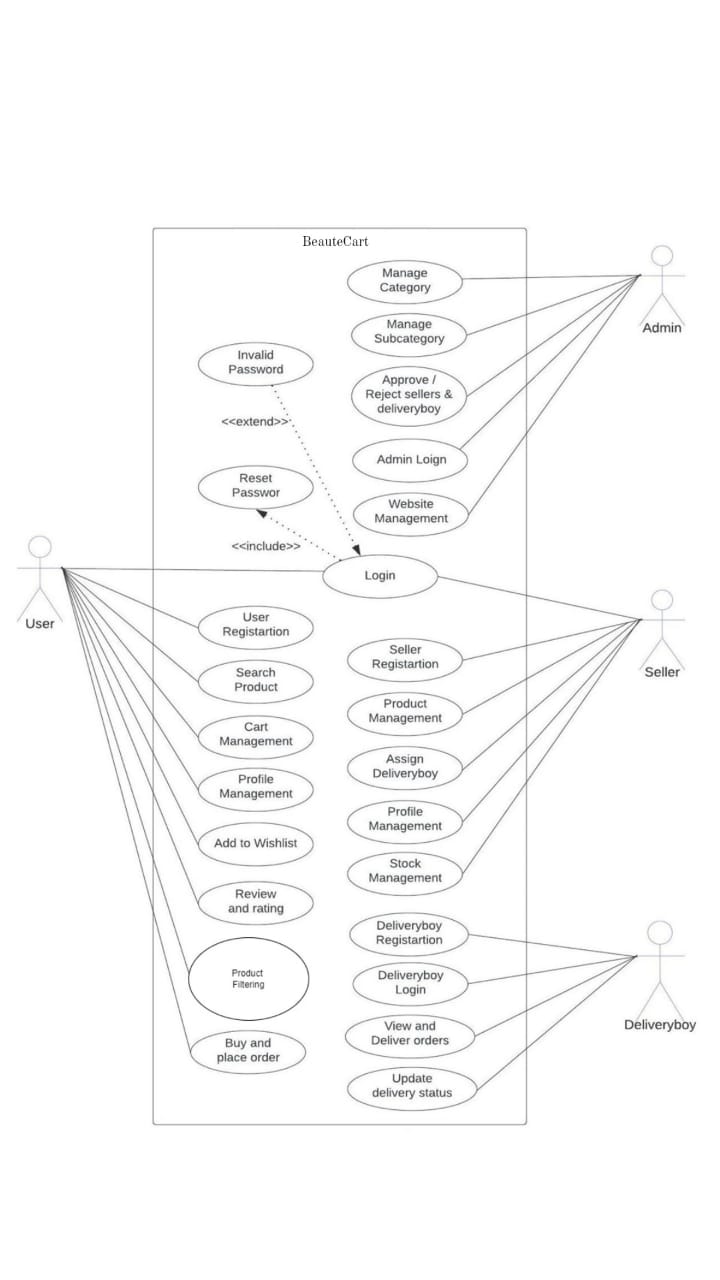
Following the identification of the aforementioned items, we must follow the guidelines below to create an effective use case diagram.

• A use case's naming is very significant. It is important to choose names that make it clear what tasks are being carried out.

• Assign appropriate names to the actors.

• Clearly show dependencies and links in the diagram.

• Keep in mind that the diagram's main purpose is to represent the needs; avoid attempting to show every relationship that might exist.



## SEQUENCE DIAGRAM

A sequence diagram shows the relationships between businesses in the chronological order in

which those relationships take place. A sequence diagram may also be referred to as an event

diagram or event scenario. The activities of system components are displayed in chronological

order on sequence diagrams. Businesspeople and software engineers regularly use these diagrams to establish and understand the requirements for new and current systems.

**Sequence Diagram Notations –**

**Actors** –

In a UML diagram, an actor represents a certain kind of role that interacts with the system's elements. An actor is never included in the UML diagram of the system we want to describe. We employ actors to perform a variety of parts, including those of external topics and human users. In a UML diagram, an actor is represented by a stick person. A sequence diagram could contain several actors.

**Lifelines** –

A lifeline is a recognisable component that stands in for a particular participant in a

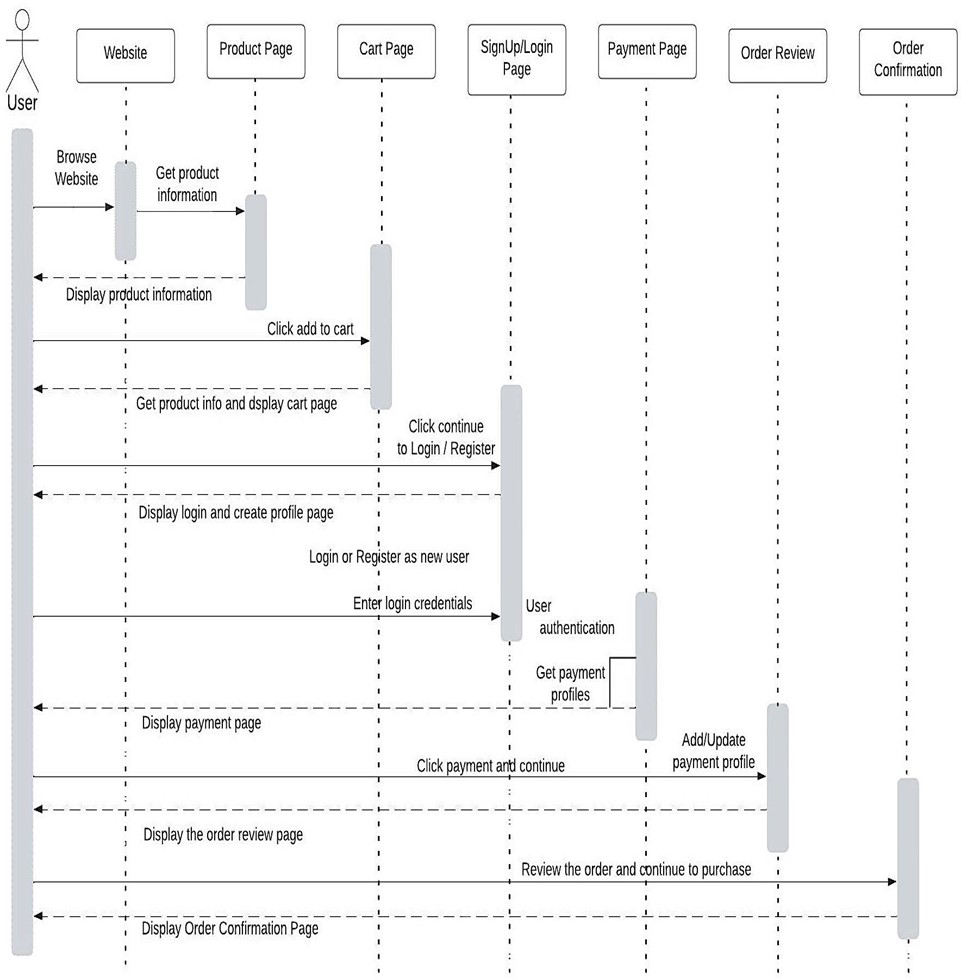
sequence diagram. Each episode in a sequence diagram is essentially represented by a lifeline. The lifeline elements of a sequence diagram are at the top.

**Messages** –

Messages are used to show object communication. The messages are presented

chronologically on the lifeline. Messages are represented by arrows. The basic components of a

sequence diagram are lifelines and messages.



## 4.2.2 State Chart Diagram

State-Chart diagram In order to explain how a software system works, diagrams are utilised.

A class, a subsystem, a package, or even an entire system's behaviour can be explained by a state

machine diagram in a UML model. It is also known as a state chart or a state transition diagram.

We may rapidly and clearly explain the interactions or interaction that takes place between external

elements and a system using state chart diagrams. The event-based system is represented in these

diagrams. An event can be used to modify an object's state. The application system uses state chart

diagrams to show the various states of an entity. A state chart diagram's principal objective is to

simulate an object's existence from creation to annihilation. State chart diagrams are used for both

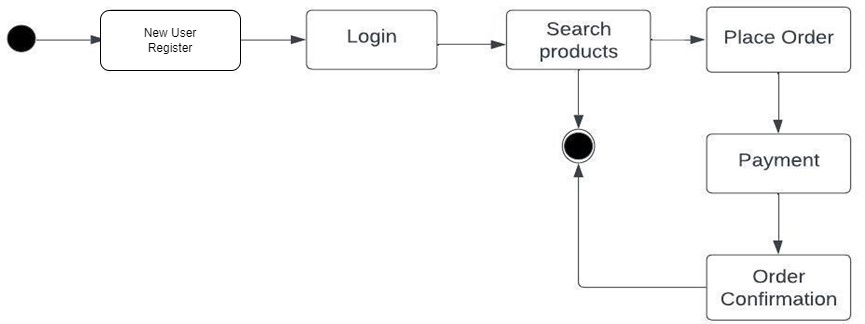
forward and backward engineering of a system. However, the main goal is to simulate the reactive

system. The major goals of using State chart diagrams are as follows:

• To simulate a system's dynamical behavior.

• To simulate a reactive system's lifetime.

• To characterize an object's various stages throughout its lifetime.



## Activity Diagram

Activity diagrams demonstrate how activities at various abstraction levels are combined to form

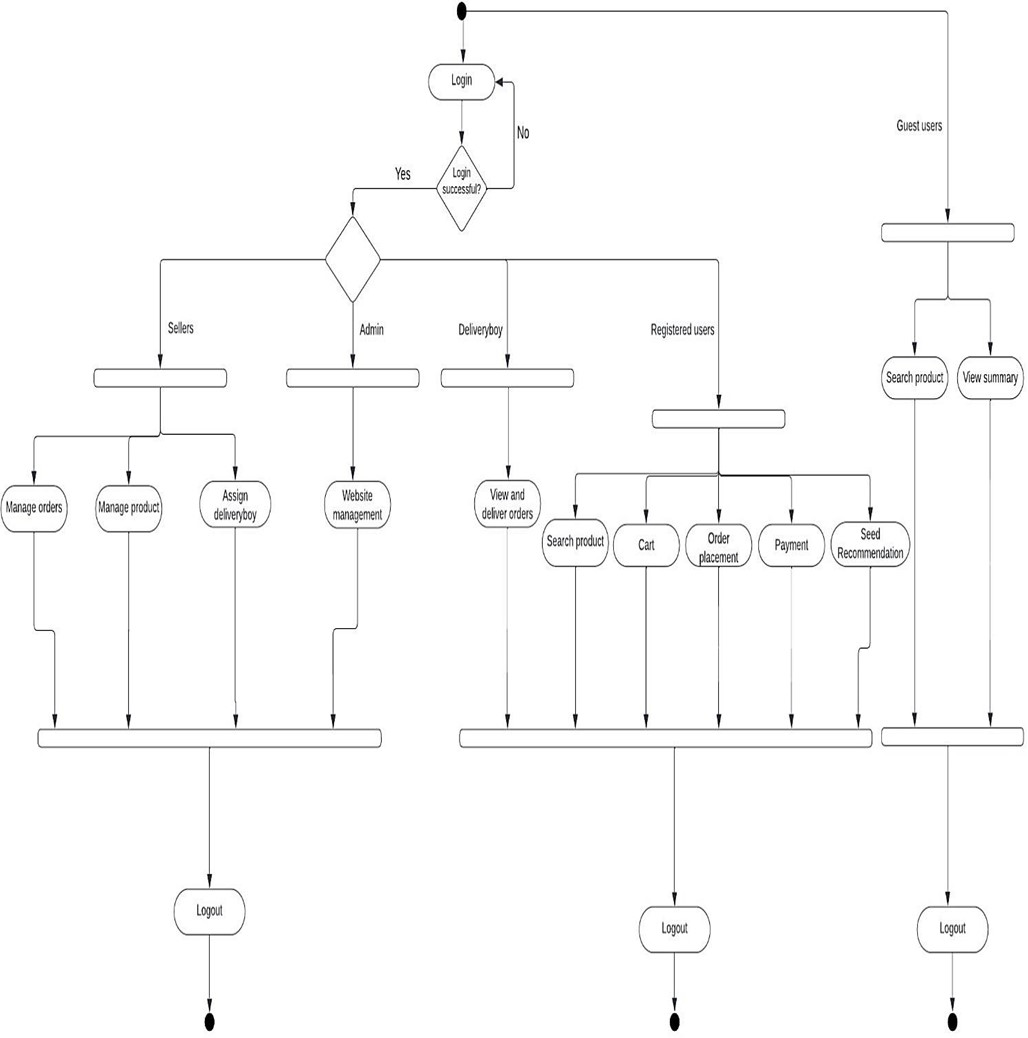
services. An event must typically be followed by various activities, especially if the operation is

meant to achieve several different goals that need to be coordinated. How events within a single

use case interact with one another is a typical requirement, particularly when tasks overlap and

require coordination. It is also excellent for forecasting how a group of use cases used to illustrate

company operations would coordinate.



## Class Diagram

Static diagrams include class diagrams. It represents the static view of the application. Class

diagrams are used to create software programmes that can be executed as well as to visually

represent, describe, and keep track of different system components. A class diagram outlines a

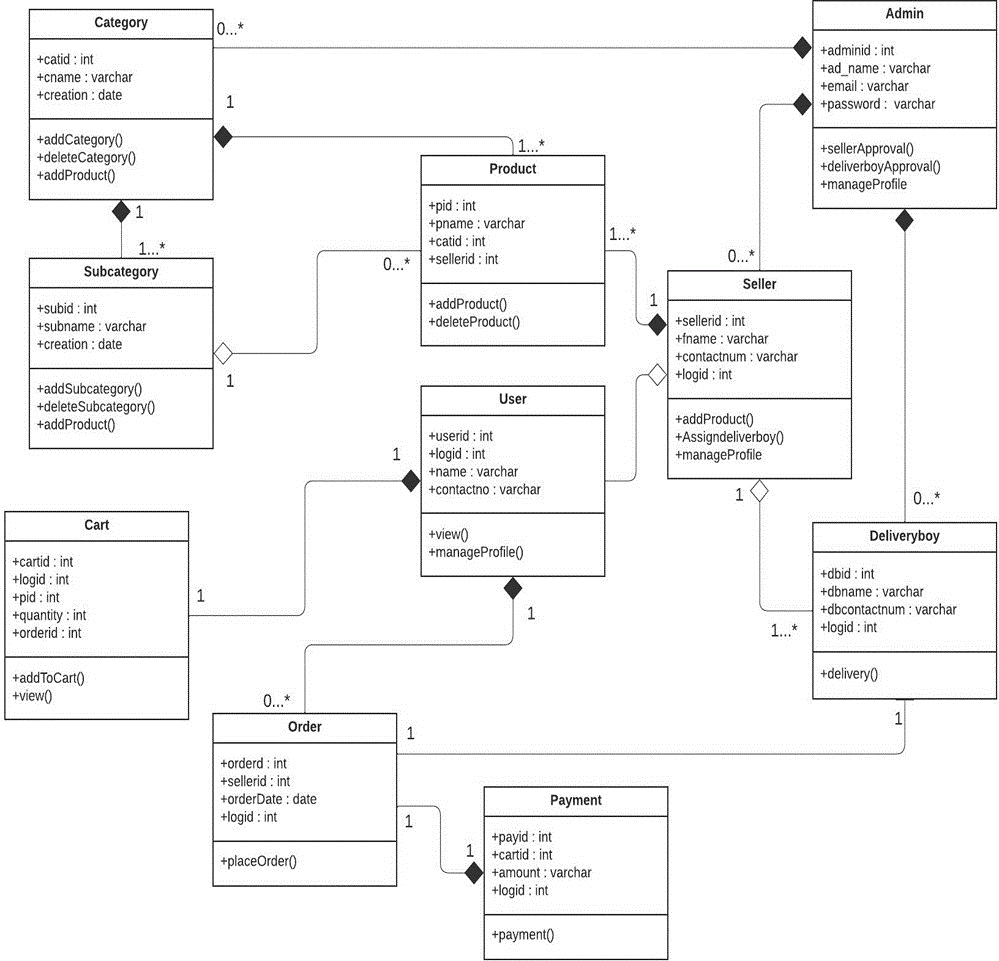
class's attributes, capabilities, and constraints. Since class diagrams are the only UML diagrams

that can be instantly translated using entity languages, they are frequently employed in the design

of object-oriented systems. A collection of classes, interfaces, affiliations, cooperative

relationships, and constraints are represented visually in a class diagram. It is also known as a

structural diagram.



## Object Diagram

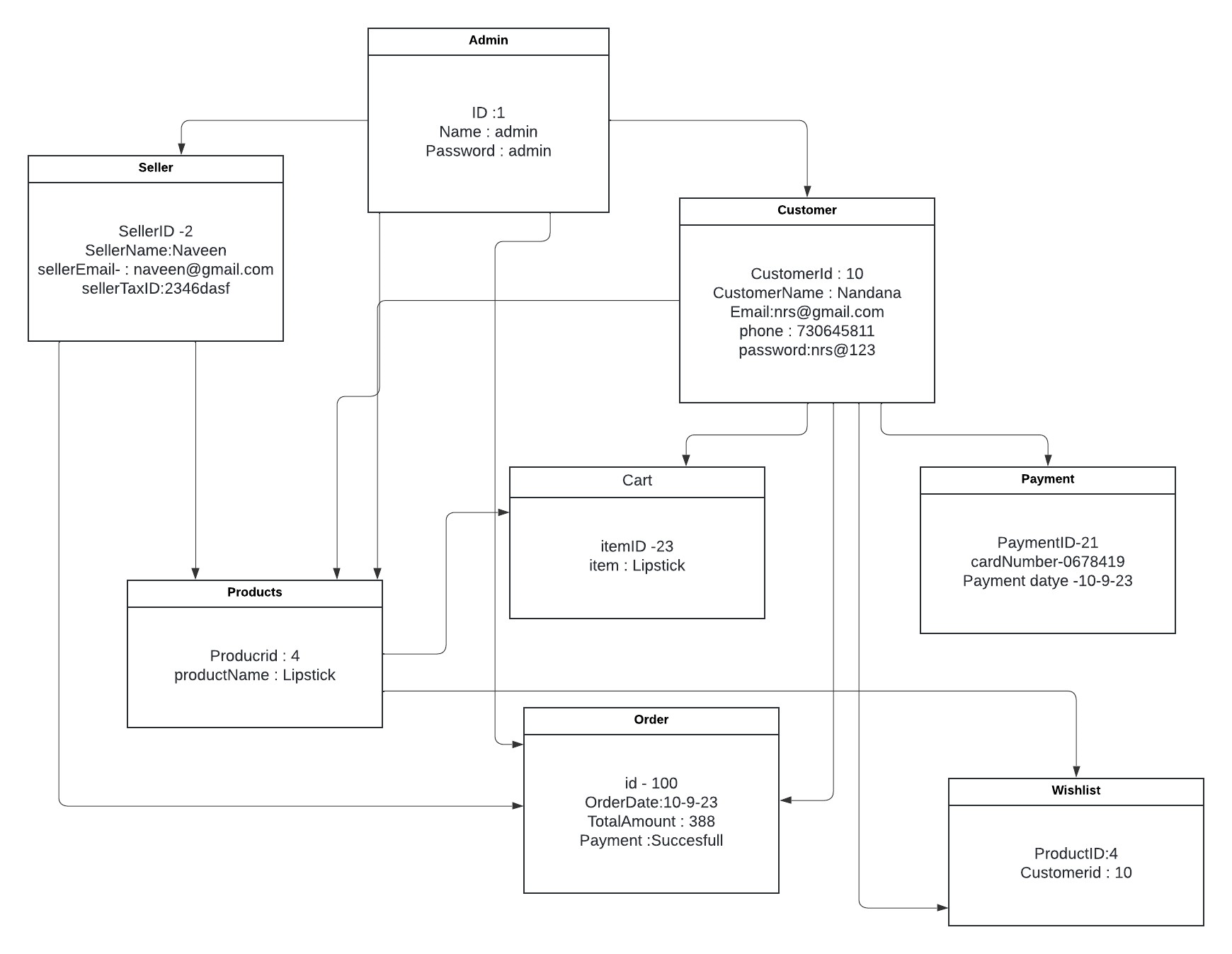
Class diagrams are necessary since they actually come from object diagrams. A class diagram

instance is shown in an object diagram. Class diagrams and object diagrams share the same core

concepts. The static view of a system can also be represented by object diagrams, although this

static view just depicts a screenshot of the system. You can visualize a gang of things and their

relationships using object diagrams.

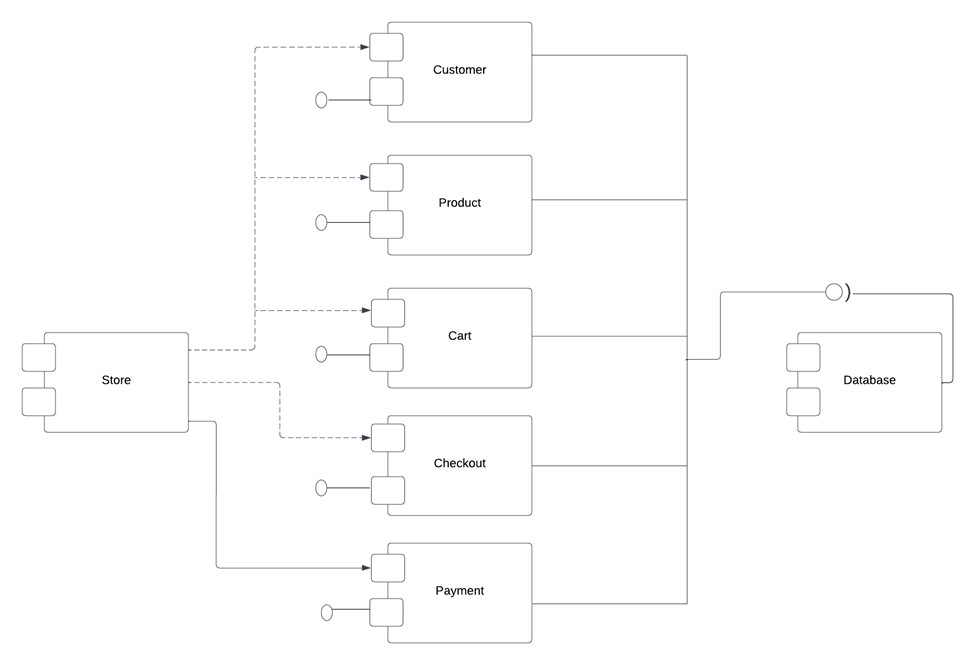


## Component Diagram

A component diagram shows the connections between smaller components to create larger

components or software systems. They serve as examples of the structure of systems that can be

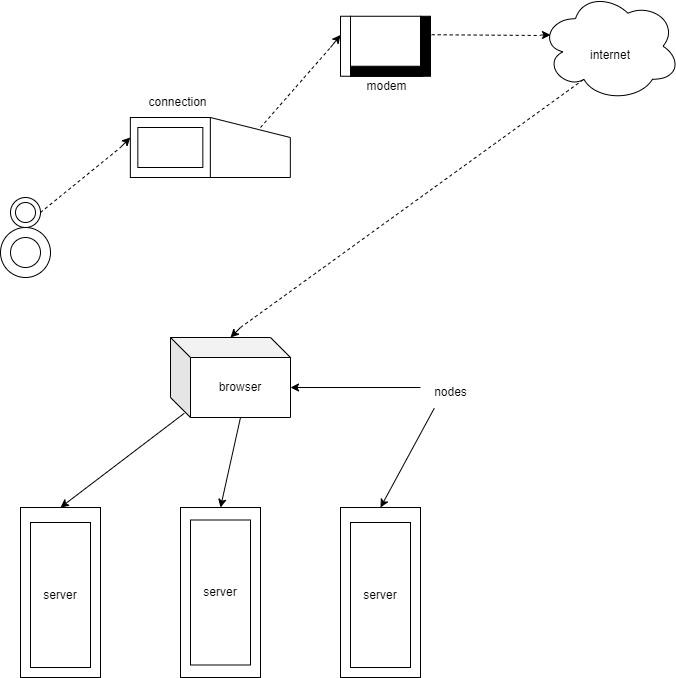
arbitrary complicated.



**4.2.8 Deployment Diagram**

A deployment diagram, a type of UML diagram, illustrates the execution architecture of a system,

which consists of nodes such as hardware or software execution environments and the middlewareconnecting them. Deployment diagrams are frequently used to show a system's actual hardware andsoftware. You can understand how the hardware will actually provide the system by utilising it.Deployment diagrams help represent the physical structure of a system, in contrast to other UMLdiagram types that primarily show the logical components of a system.



**4.2.9 Collaboration Diagram**

A collaboration diagram is a diagram that is used to represent the relationships between objects in a system. It is similar to a sequence diagram in that it represents the same information, but it does so in a different way. Instead of showing the flow of messages between objects, it depicts the structure of the objects in the system. This is because collaboration diagrams are based on object-oriented programming, where objects have various attributes and are connected to each other. Thus, collaboration diagrams are a visual representation of the object architecture in a system. A component diagram includes the following components:

• Objects: Objects are represented by symbols with their name and class underlined,

separated by a colon. In a collaboration diagram, objects are used to represent a class

instance and specify its name and class. It is not necessary for every class to have an object

representation, and a single class may have multiple objects. Objects are created first, and

their class is specified afterwards. Naming objects is important to differentiate them from

one another.

• Actors: Actors play a key role in the collaboration diagram as they invoke the interaction.

Each actor has its own name and role. In the diagram, one actor initiates the use case.

• Links: Links are instances of association that connect objects and actors. They represent

the relationship between objects through which messages are sent. Links are represented

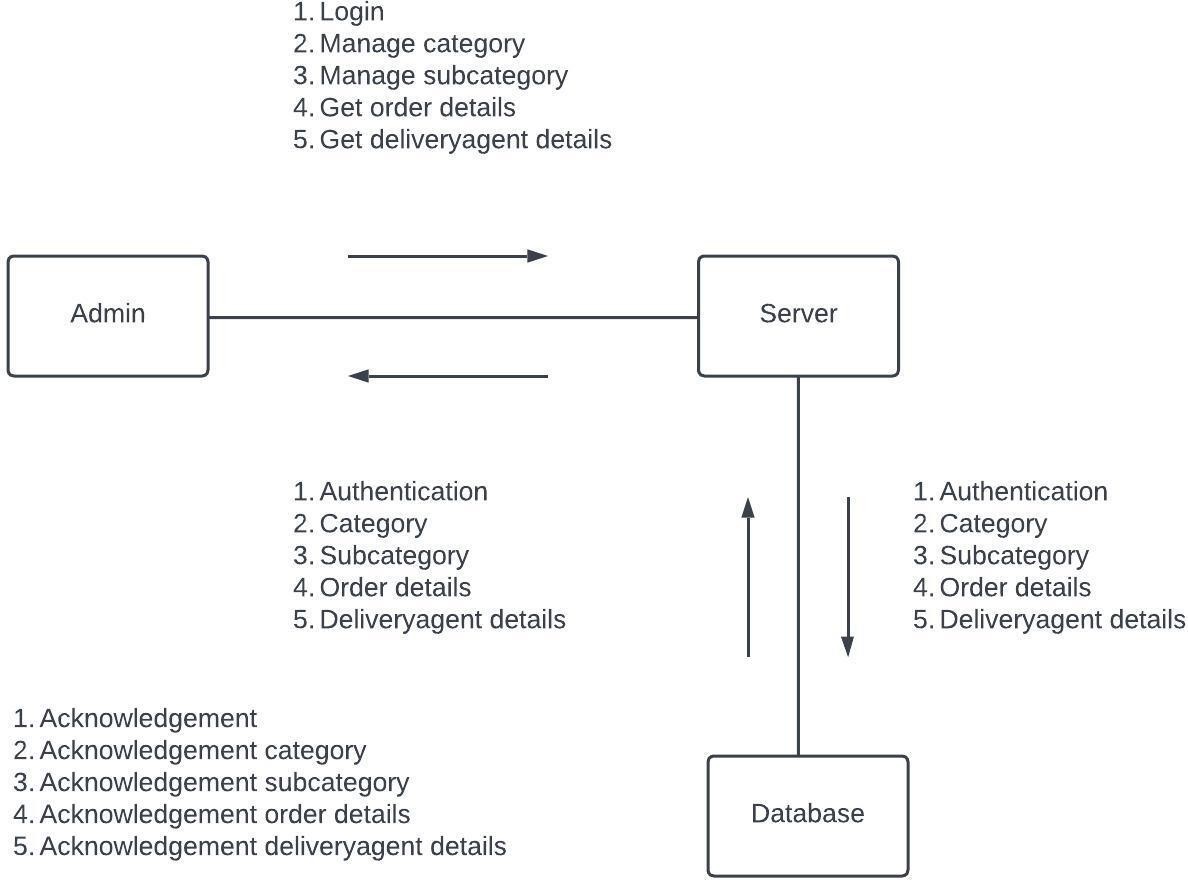
by solid lines and help objects to navigate to other objects.

• Messages: Messages represent communication between objects that carry information and

are identified by a sequence number. They are represented by labeled arrows placed near

the link and sent from the sender to the receiver. The direction must be navigable in that

specific direction, and the receiver must understand the messages.

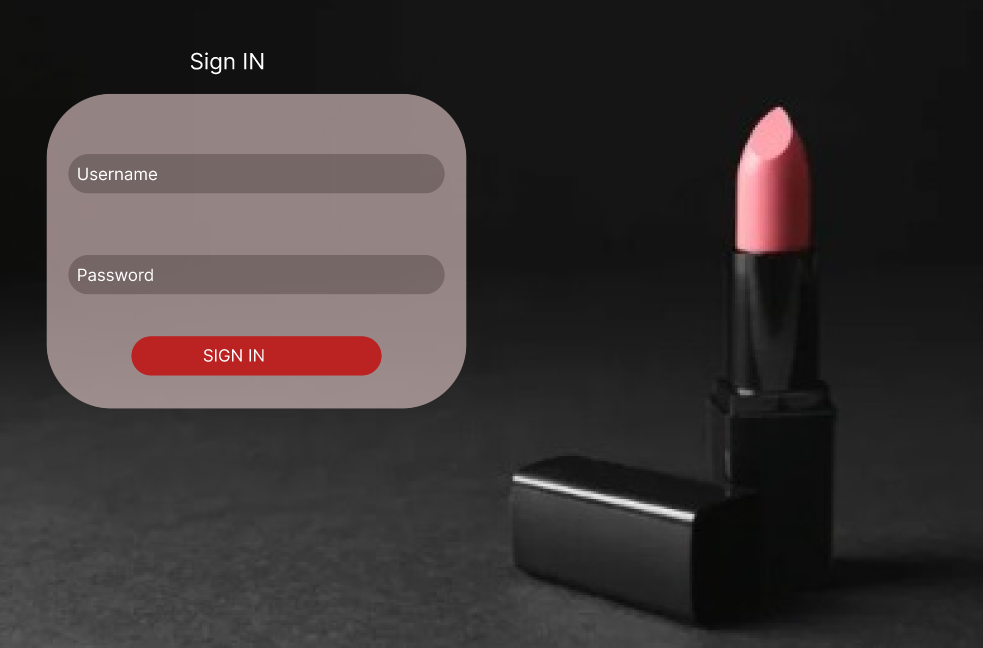


## 4.3 USER INTERFACE DESIGN USING FIGMA

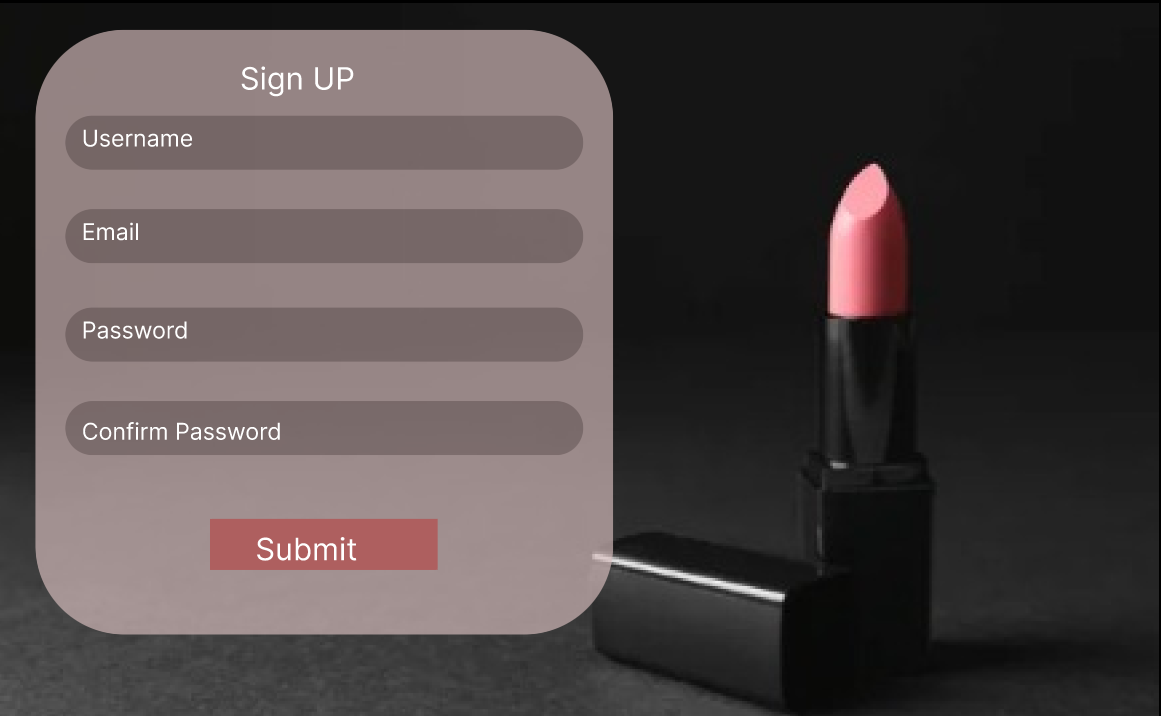
**Form Name: Home Page**



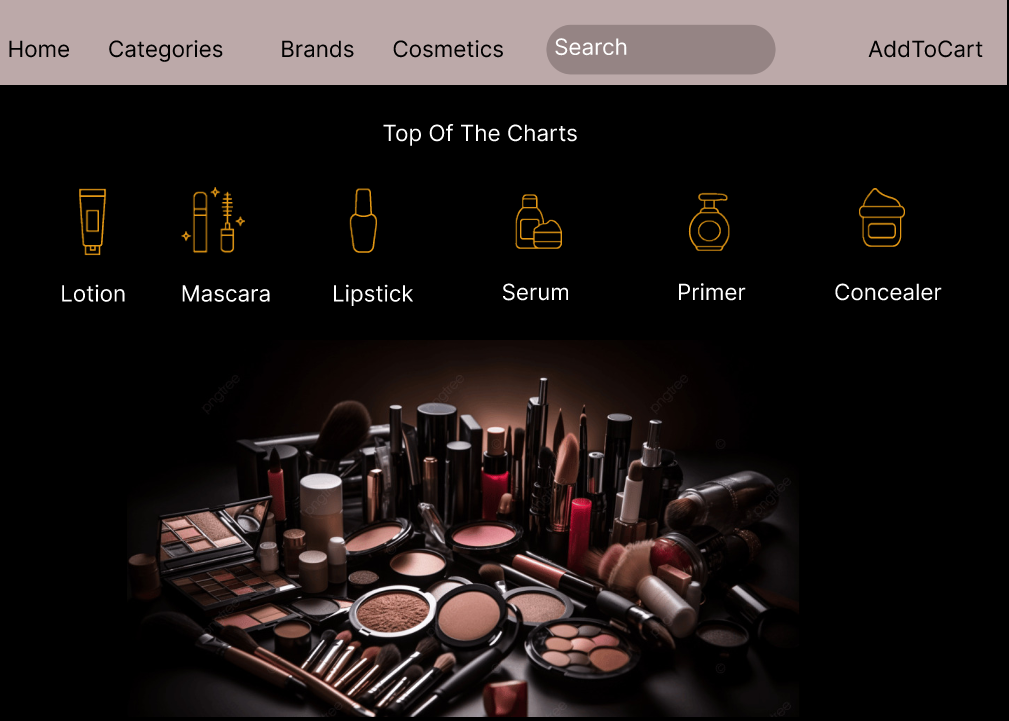
**Form Name: Login Page**



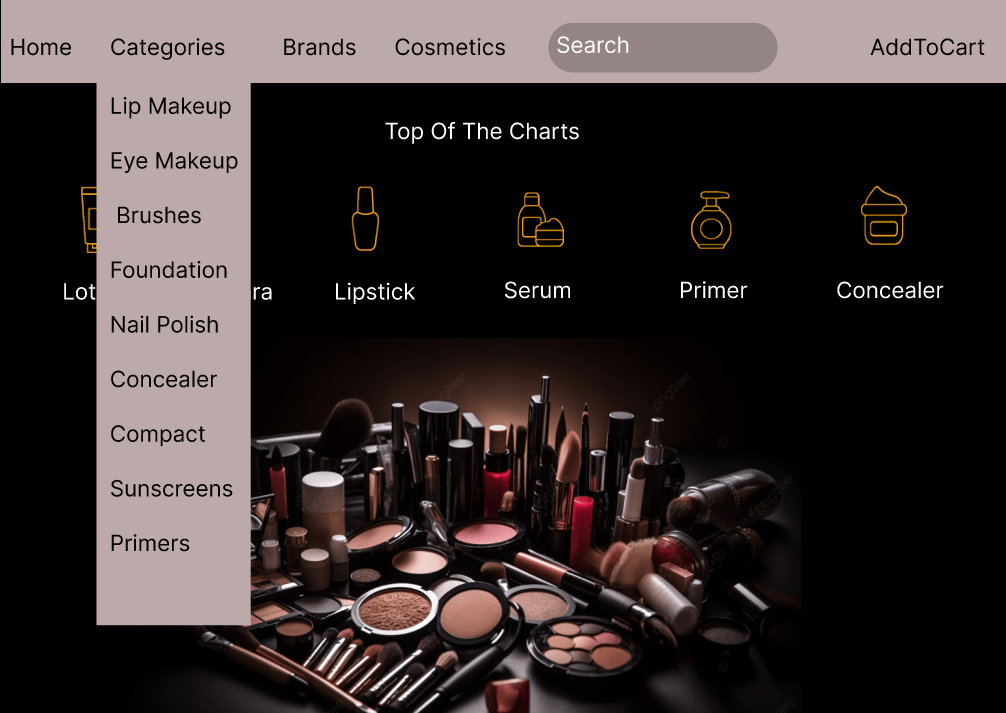
**Form Name: Registration Page**



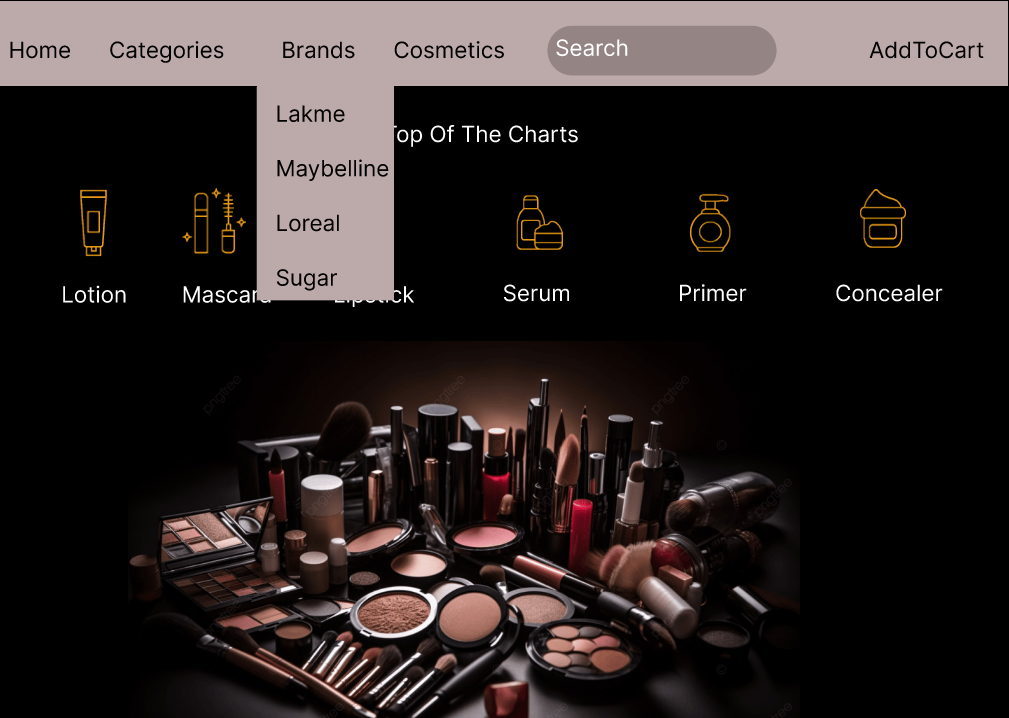
**Form Name: User Home Page**



**Form Name: Category Listing**



**Form Name: Brand Listing**



## 4.4 DATABASE DESIGN

Database design is a crucial aspect of information management, focusing on creating an organized structure to store and manage data efficiently. The process typically involves two main stages: information-level design and physical-level design. During information-level design, user requirements are gathered and translated into a conceptual database model that accurately reflects the intended structure and relationships among different data elements. This stage is independent of any specific Database Management System (DBMS) and aims to capture the essence of the information that the database will handle.

The second stage, physical-level design, involves converting the conceptual database model into a specific DBMS design. This step considers the characteristics and requirements of the chosen DBMS, incorporating details such as data types, indexing, and storage mechanisms. It ensures that the database can be implemented effectively within the selected system. The goal of this stage is to bridge the gap between the conceptual model and the practical implementation, optimizing performance and ensuring compatibility with the chosen technology.

In addition to system design, database design also strives to achieve two key objectives: data integrity and data independence. Data integrity ensures the accuracy and consistency of information within the database, while data independence allows for modifications to the database structure without affecting the applications that use the data. This separation between the logical structure of the database and the physical implementation enhances flexibility and adaptability, facilitating future updates or changes to the system without disrupting the overall functionality. Overall, a well-executed database design is fundamental for creating a robust, secure, and scalable information management system.

### 4.4.1 Relational Database Management System (RDBMS)

A relational database management system (RDBMS) is a popular type of database that organizes data into tables to facilitate relationships with other stored data sets. Tables can contain vast amounts of data, ranging from hundreds to millions of rows, each of which are referred to as records. In formal relational model language, a row is called a tuple, a column heading is an attribute, and the table is a relation. A relational database consists of multiple tables, each with its own name. Each row in a table represents a set of related values.

In a relational database, relationships are already established between tables to ensure the integrity of both referential and entity relationships. A domain D is a group of atomic values, and a common way to define a domain is by choosing a data type from which the domain's data values are derived. It is helpful to give the domain a name to make it easier to understand the values it contains. Each value in a relation is atomic and cannot be further divided.

In a relational database, table relationships are established using keys, with primary key and foreign key being the two most important ones. Entity integrity and referential integrity relationships can be established with these keys. Entity integrity ensures that no primary key can have null values, while referential integrity ensures that each distinct foreign key value must have a matching primary key value in the same domain. Additionally, there are other types of keys such as super keys and candidate keys.

### 4.4.2 Normalization

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are two different kinds of keys. Primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized.

Normalization is a process in database design that aims to organize data into proper tables and columns, making it easily correlated to the data by the user. This process eliminates data redundancy that can be a burden on computer resources. The main steps involved in normalization include:

* Normalizing the data
* Choosing appropriate names for tables and columns
* Choosing the correct names for the data

By following these steps, a developer can create a more efficient and organized database that is easier to manage and maintain.

### 4.4.3 Sanitization

Data sanitization is the process of removing any illegal characters or values from data. In web applications, sanitizing user input is a common task to prevent security vulnerabilities. PHP provides a built-in filter extension that can be used to sanitize and validate various types of external input such as email addresses, URLs, IP addresses, and more. These filters are designed to make data sanitization easier and faster. For example, the PHP filter extension has a function that can remove all characters except letters, digits, and certain special characters (!#$%&’\*+-=?\_`{|}~@.[]), as specified by a flag.Web applications often receive external input from various sources, including user input from forms, cookies, web services data, server variables, and database query results. It is important to sanitize all external input to ensure that it is safe and does not contain any malicious code or values.

**4.4.4 Indexing**

An index is a database structure that enhances the speed of table operations. Indexes can be created on one or more columns to facilitate quick lookups and efficient ordering of records. When creating an index, it's important to consider which columns will be used in SQL queries and to create one or more indexes on those columns. In practice, indexes are a type of table that store a primary key or index field and a pointer to each record in the actual table. Indexes are invisible to users and are only used by the database search engine to quickly locate records. The CREATE INDEX statement is used to create indexes in tables.

When tables have indexes, the INSERT and UPDATE statements take longer because the database needs to insert or update the index values as well. However, the SELECT statements become faster on those tables because the index allows the database to locate records more quickly

### 4.5 TABLE DESIGN

**1.Tbl\_Admin**

Primary key: **Adminid**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Admin\_id | Int | Primary key |
| Email\_id | Varchar(255) | notnull |
| Contact number | Varchar(255) | notnull |

**2.Tbl\_Users**

Primary key: **User\_id**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| User\_id | Int | Primary key |
| Username | Varchar(255) | notnull |
| Name | Varchar(255) | notnull |
| Email | Varcahr(255) | notnull |
| Address | Varchar(255) | notnull |
| Password | Varchar(255) | notnull |

**3.Tbl\_Brands**

Primary key: **Brand\_id**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Brand\_id | Int | Primary key |
| Brandname | Varchar(255) | notnull |
| Logo | Varchar(255) | notnull |
| Brand License | Varcahr(255) | notnull |

**4.Tbl\_Category**

Primary key: **cat\_id**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| cat\_id | Int | Primary key |
| Category\_name | Varchar(255) | notnull |

**5.Tbl\_SubCategory**

Primary key: **Subcat\_id**

Foreign key: **cat\_id** references table **Tbl\_Category**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Subcat\_id | Int | Primary key |
| cat\_id | int | Reference key |
| Subcat\_name | Varchar(255) | notnull |

**6.Tbl\_Products**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

Foreign key: **Brand\_id** references table **Tbl\_Brand**

Foreign key: **cat\_id** references table **Seller**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Product\_id | Int | Primary key |
| Subcat\_id | int | reference key |
| Brand\_id | int | reference key |
| Seller\_id | int | reference key |
| Name | Varchar(255) | notnull |
| Description | Varchar(255) | notnull |
| Price | Varchar(255) | notnull |

**7.Tbl\_cart**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| cart\_id | Int | Primary key |
| user\_id | int | reference key |
| product\_id | int | reference key |
| Quantity | Float(15) | notnull |

**8.Tbl\_order**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| order\_id | Int | Primary key |
| user\_id | int | reference key |
| product\_id | int | reference key |
| Quantity | Varchar(255) | notnull |
| Order\_date | date | notnull |
| Total\_amount | Varchar(255) | notnull |

**9.Tbl\_Payment**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| payment\_id | Int | Primary key |
| order\_id | int | reference key |
| paymentmethod\_id | int | reference key |
| amount | Decimal(10,2) | notnull |

**10.Tbl\_PaymentMethod**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Paymentmethod\_id | Int | Primary key |
| user\_id | int | reference key |
| Paymentmethod | Varchar(100) | notnull |
| Account\_no | Varchar(255) | notnull |
| Cvv | int | notnull |
| Expiry\_date | int | notnull |

**11.Tbl\_Seller**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| Seller\_id | Int | Primary key |
| Address | Varchar(255) | notnull |
| product\_id | int | reference key |
| Phone | Varchar(255) | notnull |
| Email | Varchar(255) | notnull |

**12.Tbl\_Review**

Primary key: **Adminid**

Foreign key: **Subcat\_id** references table **Tbl\_SubCategory**

|  |  |  |
| --- | --- | --- |
| **Field Type** | **Keytype** | **Data Constarints** |
| review\_id | Int | Primary key |
| user\_id | int | reference key |
| product\_id | int | reference key |
| rating | Decimal(2,1) | notnull |
| comment | text | notnull |

# CHAPTER 5

# SYSTEM TESTING

* 1. **INTRODUCTION**

Software testing involves executing a software program in a controlled manner to determine if it behaves as intended, often using verification and validation methods. Validation involves evaluating a product to ensure it complies with specifications, while verification can involve reviews, analyses, inspections, and walkthroughs. Static analysis examines the software's source code to identify issues, while dynamic analysis examines its behavior during runtime to gather information like execution traces, timing profiles, and test coverage details.

Testing involves a series of planned and systematic activities that start with individual modules and progress to the integration of the entire computer-based system. The objectives of testing include identifying errors and bugs in the software, ensuring that the software functions according to its specifications, and verifying that it meets performance requirements. Testing can be performed to assess correctness, implementation efficiency, and computational complexity.

A successful test is one that detects an undiscovered error, and a good test case has a high probability of uncovering such errors. Testing is crucial to achieving system testing objectives and can involve various techniques such as functional testing, performance testing, and security testing

## TEST PLAN

A test plan is a document that outlines the required steps to complete various testing methodologies. It provides guidance on the activities that need to be performed during testing. Software developers create computer programs, documentation, and associated data structures. They are responsible for testing each component of the program to ensure it meets the intended purpose. To address issues with self-evaluation, an independent test group (ITG) is often established.

Testing objectives should be stated in quantifiable language, such as mean time to failure, cost to find and fix defects, remaining defect density or frequency of occurrence, and test work-hours per regression test.

The different levels of testing include:

* Unit testing
* Integration testing
* Data validation testing
* Output testing

### Unit Testing

Unit testing is a software testing technique that focuses on verifying individual components or modules of the software design. The purpose of unit testing is to test the smallest unit of software design and ensure that it performs as intended. Unit testing is typically white-box focused, and multiple components can be tested simultaneously. The component-level design description is used as a guide during testing to identify critical control paths and potential faults within the module's perimeter.

During unit testing, the modular interface is tested to ensure that data enters and exits the software unit under test properly. The local data structure is inspected to ensure that data temporarily stored retains its integrity during each step of an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once, and all error handling paths are tested to ensure that the software can handle errors correctly.

Before any other testing can take place, it is essential to test data flow over a module interface. If data cannot enter and exit the system properly, all other tests are irrelevant. Another crucial duty during unit testing is the selective examination of execution pathways to anticipate potential errors and ensure that error handling paths are set up to reroute or halt work when an error occurs. Finally, boundary testing is conducted to ensure that the software operates correctly at its limits.

In the Sell-Soft System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. Any issues with the internal logic of the modules were fixed, and each module was tested and run separately after coding. Unused code was

eliminated, and it was confirmed that every module was functional and produced the desired outcome.

### Integration Testing

Integration testing is a systematic approach that involves creating the program structure while simultaneously conducting tests to identify interface issues. The objective is to construct a program structure based on the design that uses unit-tested components. The entire program is then tested. Correcting errors in integration testing can be challenging due to the size of the overall program, which makes it difficult to isolate the causes of the errors. As soon as one set of errors is fixed, new ones may arise, and the process may continue in an apparently endless cycle.

Once unit testing is complete for all modules in the system, they are integrated to check for any interface inconsistencies. Any discrepancies in program structures are resolved, and a unique program structure is developed.

### Validation Testing or System Testing

The final stage of the testing process involves testing the entire software system as a whole, including all forms, code, modules, and class modules. This is commonly referred to as system testing or black box testing. The focus of black box testing is on testing the functional requirements of the software. A software engineer can use this approach to create input conditions that will fully test each program requirement. The main types of errors targeted by black box testing include incorrect or missing functions, interface errors, errors in data structure or external data access, performance errors, initialization errors, and termination errors.

### Output Testing or User Acceptance Testing

User acceptance testing is performed to ensure that the system meets the business requirements and user needs. It is important to involve the end users during the development process to ensure that the software aligns with their needs and expectations. During user acceptance testing, the input and output screen designs are tested with different types of test data. The preparation of test data is critical to ensure comprehensive testing of the system. Any errors identified during testing are addressed and corrected, and the corrections are noted for future reference.

* + 1. **Automation Testing**

Automation testing is a software testing approach that employs specialized automated testing software tools to execute a suite of test cases. Its primary purpose is to verify that the software or equipment operates precisely as intended. Automation testing identifies defects, bugs, and other issues that may arise during product development.

While some types of testing, such as functional or regression testing, can be performed manually, there are numerous benefits to automating the process. Automation testing can be executed at any time of day and uses scripted sequences to evaluate the software. The results are reported, and this information can be compared to previous test runs. Automation developers typically write code in programming languages such as C#, JavaScript, and Ruby.

* + 1. **Selenium Testing**

Selenium is an open-source automated testing framework used to verify web applications across different browsers and platforms. Selenium allows for the creation of test scripts in various programming languages such as Java, C#, and Python. Jason Huggins, an engineer at Thought Works, developed Selenium in 2004 while working on a web application that required frequent testing. He created a JavaScript program called "JavaScriptTestRunner" to automate browser actions and improve testing efficiency. Selenium has since evolved and continues to be developed by a team of contributors.

In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It allows for the creation of executable specifications in a human-readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals.

Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web applications across different browsers and platforms, using a business-readable and maintainable format.

**Test Case 1 : User Login**

**Code**

from django.test import TestCase

# Create your tests here.

import unittest

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

import time

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

class LoginTest(unittest.TestCase):

def setUp(self):

# Start the Selenium WebDriver

self.driver = webdriver.Chrome() # Adjust based on your WebDriver configuration

self.driver.get("http://127.0.0.1:8000/login/") # Replace with the actual URL of your login page

def test\_login\_successful(self):

# Find the username, password, and login button elements

username\_input = self.driver.find\_element(By.NAME, "username")

password\_input = self.driver.find\_element(By.NAME, "password")

# You can use a more robust selector for the login button

login\_button = WebDriverWait(self.driver, 10).until(

EC.element\_to\_be\_clickable((By.CSS\_SELECTOR, "button[type='submit']"))

)

# Enter valid credentials

username\_input.send\_keys("Anna")

password\_input.send\_keys("Anna@123")

# Click the login button

login\_button.click()

# Wait for a while to see the result (you can adjust this based on your application's response time)

time.sleep(2)

# Assert that the URL matches the expected redirect

self.assertEqual(self.driver.current\_url, 'http://127.0.0.1:8000/home2/')

def tearDown(self):

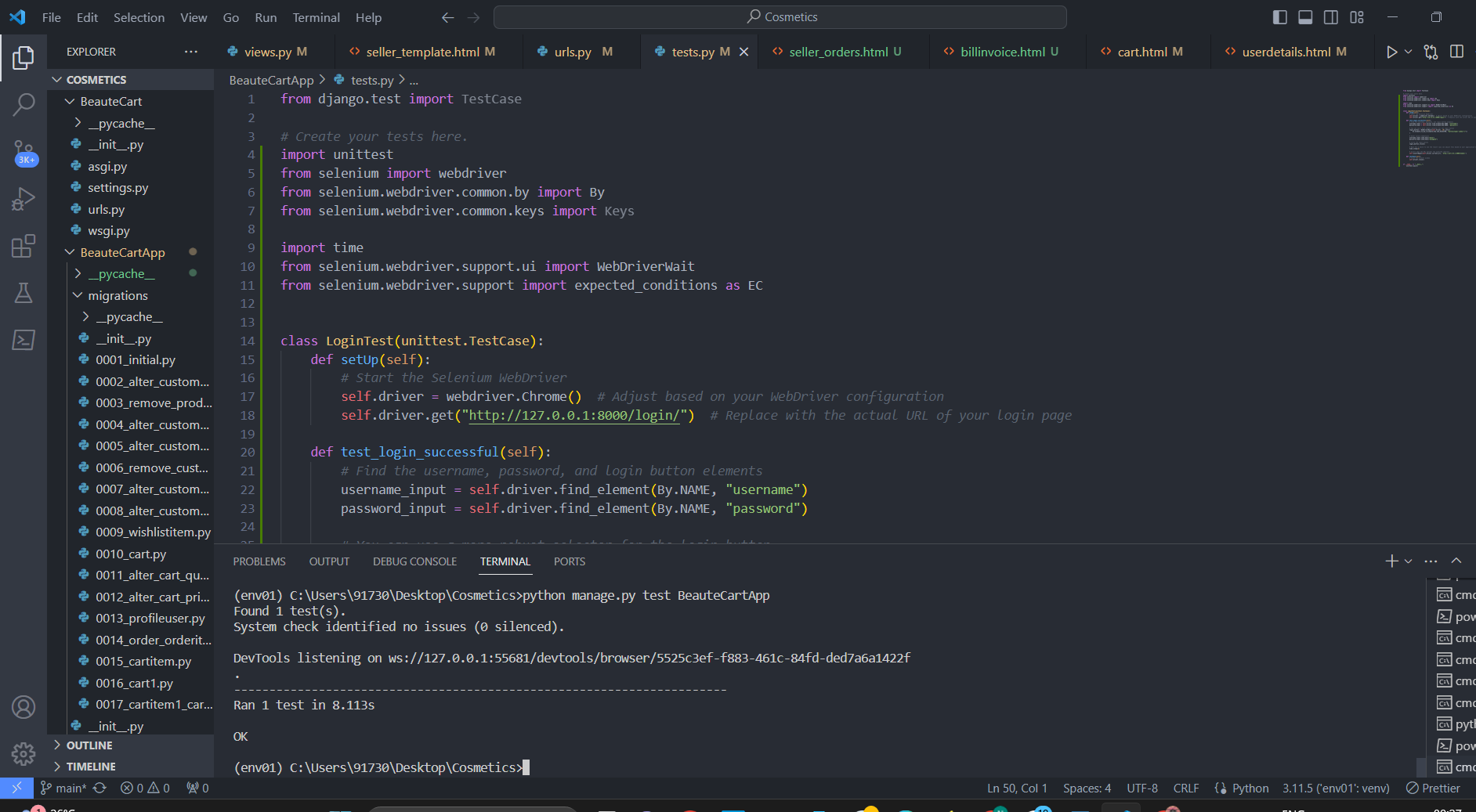
# Close the browser window

self.driver.close()

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

**Screenshot**



**Eg.Test Report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case 1** | | | | | |
| **Project Name:** | | | | | |
| **Login Test Case** | | | | | |
| **Test Case ID: Test\_1** | | | **Test Designed By:** Nandana R.S | | |
| **Test Priority(Low/Medium/High):** High | | | **Test Designed Date:** 3/12/2023 | | |
| **Module Name**: Login | | | **Test Executed By :** Dr Bijimol TK | | |
| **Test Title :** Customer Login | | | **Test Execution Date:** 4/12/2023 | | |
| **Description: :** Verify login with valid email and password | | |  | | |
| **Pre-Condition :**User has valid username and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/**  **Fai l)** |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |  |  |  |
|  |  |  |  |  |  |
| 6 |  |  |
| 7 |  |  |  |  |  |
|  |  |  |  |  |
| **Post-Condition:** | | | | | |

**Test Case 2:**

**Code**

**Screenshot**

**Test report**

**Minimum 4 test cases (1 login 3 functionalities)**

# CHAPTER 6

# IMPLEMENTATION

## INTRODUCTION

The implementation phase of a project is where the design is transformed into a functional system. It is a crucial stage in ensuring the success of the new system, as it requires gaining user confidence that the system will work effectively and accurately. User training and documentation are key concerns during this phase. Conversion may occur concurrently with user training or at a later stage. Implementation involves the conversion of a newly revised system design into an operational system.

During this stage, the user department bears the primary workload, experiences the most significant upheaval, and has the most substantial impact on the existing system. Poorly planned or controlled implementation can cause confusion and chaos. Whether the new system is entirely new, replaces an existing manual or automated system, or modifies an existing system, proper implementation is essential to meet the organization's needs. System implementation involves all activities required to convert from the old to the new system. The system can only be implemented after thorough testing is done and found to be working according to specifications. System personnel evaluate the feasibility of the system. Implementation requires extensive effort in three main areas: education and training, system testing, and changeover. The implementation phase involves careful planning, investigating system and constraints, and designing methods to achieve changeover.

## IMPLEMENTATION PROCEDURES

Software implementation is the process of installing the software in its actual environment and ensuring that it satisfies the intended use and operates as expected. In some organizations, the software development project may be commissioned by someone who will not be using the software themselves. During the initial stages, there may be doubts about the software, but it's important to ensure that resistance does not build up. This can be achieved by:

* Ensuring that active users are aware of the benefits of the new system, building their confidence in the software.
* Providing proper guidance to the users so that they are comfortable using the application. Before viewing the system, users should know that the server program must be running on the server. Without the server object up and running, the intended process will not take place.

### User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

### 6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness, it is essential to provide training on the new application software to the user. This training should include the underlying philosophy of using the new system, such as the flow of screens, screen design, the type of help available on the screen, the types of errors that may occur while entering data, and the corresponding validation checks for each entry, and ways to correct the data entered. Additionally, the training should cover information specific to the user or group, which is necessary to use the system or part of the system effectively. It is important to note that this training may differ across different user groups and levels of hierarchy.

### 6.2.3 System Maintenance

### The maintenance phase is a crucial aspect of the software development cycle, as it is the time when the software is actually put to use and performs its intended functions. Proper maintenance is essential to ensure that the system remains functional, reliable, and adaptable to changes in the system environment. Maintenance activities go beyond simply identifying and fixing errors or bugs in the system. It may involve updates to the software, modifications to its functionalities, and enhancements to its performance, among other things. In essence, software maintenance is an ongoing process that requires continuous monitoring, evaluation, and improvement of the system to meet changing user needs and requirements.

# CHAPTER 7

# CONCLUSION AND FUTURE SCOPE

## CONCLUSION

BeauteCart emerges as a premier online cosmetics shop, redefining the beauty retail landscape with its commitment to quality and customer-centric approach. This beauty destination is meticulously curated, offering a diverse range of top-tier cosmetics and skincare products to cater to the unique needs of beauty enthusiasts. BeauteCart not only provides a seamless and delightful shopping experience through its user-friendly interface but also prioritizes customer education by offering detailed information on products. The platform's dedication to authenticity, transparency, and a comprehensive beauty catalog positions it as a trusted and go-to destination for those seeking to elevate their beauty routines with genuine and luxurious products. BeauteCart isn't just an online shop; it's a beauty haven that empowers customers to make informed and confident choices in their beauty journey.

* 1. **FUTURE SCOPE**

BeauteCart, the Online Cosmetics Shop, holds significant potential for future growth and

enhancement within the dynamic beauty and skincare industry. With the rising trend of online

beauty shopping, BeauteCart can explore avenues to expand its product offerings by including a

wider range of beauty and skincare brands, emphasizing detailed product information, customer

reviews, and personalized recommendations to enhance the shopping experience. To foster a sense of community engagement, the platform could introduce features such as beauty forums, virtual beauty consultations, and user-generated content, allowing customers to share beauty tips, experiences, and product recommendations. Embracing emerging technologies like augmented reality for virtual try-ons and personalized beauty algorithms for tailored product suggestions can further elevate the customer experience. Despite these opportunities, addressing challenges related to color accuracy in product representation, ensuring a secure and user-friendly interface, and staying abreast of evolving beauty trends will be essential for BeauteCart's sustained success. As the beauty industry continues to evolve, BeauteCart is poised for an exciting future, promising continuous innovation and a seamless online shopping experience for beauty enthusiasts.

.

# CHAPTER 8

# BIBLIOGRAPHY

### REFERENCES:

* PankajJalote, “Software engineering: a precise approach”
* Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009
* Ken Schwaber, Mike Beedle, Agile Software Development with Scrum͟

Pearson (2008)

* Roger S Pressman, “Software Engineering”
* IEEE Std 1016 Recommended Practice for Software Design Descriptions
* Allen B.Downey ,“Think Python: How to think like a Computer Scientist, 2nd edition,

(O’Reilly, 2015)

### WEBSITES:

• www.w3schools.com

• www.bootstrap.com

• https://chat.openai.com/chat

• www.jquery.com

# CHAPTER 9

# APPENDIX

## Sample Code

Main functionalities

## Screen Shots