ABSTRACT

.

This project (DCEE) is an initiative of transformation focused on digitally uplifting small and local Indian companies and emphasizes them to prosper on an increasingly competitive online market. The platform offers a comprehensive set of tools and functions designed to strengthen digital presence, streamline operations and support customer relationships. By providing customizable digital facades, secure payment gateway and intuitive user interface DCEE, businesses allow businesses to cultivate confidence and effectively participate in your customers. The core of the DCEE function is its advanced analytical module powered by the Robert model, which makes it easier for predictive analysts of supplies. This function allows businesses to accurately predict market requirements, optimize stock levels and reduce waste, which eventually leads to improvement in revenue and profitability.

The platform is structured to satisfy various user roles, customers and custom -made customers, such as profile management, transaction supervision and real -time data management. In addition, DCEE includes a robust Chatbot function that provides immediate assistance to users, leads them to suitable sources and helps solve common problems. This promotes user experience and supports greater involvement and maintenance. DCEE also solves critical challenges such as compliance with regulations, facilitating payments and gaps in the field of digital literacy, which ensures that small businesses are equipped to navigate the complexity of the digital environment. In addition to these DCEE functions, it supports community resistance by allowing small businesses to use technology for growth.

The aim of the availability and friendship of the user is to create a supporting ecosystem in which local entrepreneurs can compete effectively and contribute to economic development in their communities. Finally, DCEE serves not only as a catalyst for individual business success, but also enriches the business spirit that is vital to the economic landscape of India. Through this holistic approach, DCEE is ready to redefine how small businesses work and succeed in Digital Age, which has a significant impact on the wider economy.

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The digital ecosystem of digital trade (DCEE) is an innovative platform designed to strengthen small and local Indian businesses by offering basic tools for navigation in the digital economy. DCEE focuses on bridging the digital abyss and provides solutions to determine digital presence, support customer confidence, management of rural payments, optimize inventory and support digital literacy. The key features include the Gemini Thinking Sales Power Sales Power Analysis Model for Market Information, Chatbot for User Instructions and Payment Module for Effective Transactions. DCEE Equips Equips Business, customers and managers for management, interpretations and transactions, creates a dynamic and accessible environment for local business growth, creates a scalable technological magazine and creates a dynamic and accessible environment for local business growth.

1.2 PROJECT SPECIFICATION

The digital ecosystem of digital trade (DCEE) is an innovative platform designed to strengthen small and local Indian businesses by offering basic tools for navigation in the digital economy. The key features include the Gemini Thinking Sales Power Sales Power Analysis Model for Market Information, Chatbot for User Instructions and Payment Module for Effective Transactions. DCEE Equips Equips Business, customers, administrators and instructors that are based on a scalable technological magazine, for management of profiles, interpretations and transactions and create a dynamic and accessible environment for local business growth.

The platform features four user roles:

• Digital Storefront Owner: Manages business profiles, adds products, views customer interactions, and processes payments.

• Customer: Registers, shops, manages cart, and completes secure transactions, interacting smoothly with local businesses.

• Admin: Oversees user profiles, manages transactions, and ensures platform security.

• Instructor: Provides educational resources, conducts training sessions, and supports digital literacy efforts for small business owners.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The digital ecosystem is designed to revolutionize small and local Indian societies by dealing with specific challenges faced in the digital environment. Many of these companies are fighting limited access to technologies and digital tools and prevent their ability to involve customers, streamline operations and increase their market presence. DCEE offers a holistic solution that strengthens the owners of basic online management tools, including customers' profiles, product lists and secure payments. This integration not only simplifies their operations, but also increases their ability to reach a wider audience.

DCEE contains an analytical module that uses predictive knowledge, helps businesses effectively manage supplies and avoid risks associated with excessive evaluation and storage. Using advanced machine learning tools, such as the Gemini Thinking model, allows you to accurately analyze the performance of sales and increase decision -making options. In addition, this platform includes users a friendly chatbot that helps users with questions and support for navigation, which significantly reduces the digital literacy barrier that many small business owners face. By facilitating the possibilities of payments in the countryside and ensuring compliance with the DCEE regulatory requirements, it supports the customer's confidence and accessibility. This ecosystem not only makes operations more efficient, but also supports business spirit by providing training and resources that support digital literacy.

Finally, DCEE is trying to bridge the digital gap and entertain small businesses to prosper, contribute to local economies and maintain their growth on the rapidly developing digital market.

2.2 EXISTING SYSTEM

Small and local Indian businesses struggle with limited digital adoption, fragmented tools, and inefficiencies in managing sales, payments, and inventory. Existing e-commerce platforms are complex and not tailored to their needs, especially in rural areas. Businesses face challenges like poor demand forecasting, lack of digital literacy support, and regulatory compliance issues. A unified, user-friendly solution is needed to simplify digital operations and drive sustainable growth.

NATURAL SYSTEM STUDIED

The DCEE project draws inspiration from traditional Indian markets, where local businesses prosper through direct interactions with customers, transactions based on the trust and community involvement. These natural ecosystems work on the principles of availability, personalized services and dynamic modifications of demand for demand. At digital age, however, these businesses face technological barriers and fragmented tools. The aim of the DCEE is to replicate and improve this dynamics of organic business in digital format, ensure availability, effective operations and customer focusing on the integration of modern analytical and payment solutions.

DESIGNED SYSTEM STUDIED

The DCEE platform is designed as an all-in-one digital trade solution for small and local Indian companies. Unlike the existing fragmented DCEE systems, it integrates basic functions such as digital facade management, predictive sales analysis, secure payment processing, and chatbot assistance into a single user -friendly ecosystem. The system uses the Gemini Thinking model for data -based sales knowledge, allowing businesses to be optimized and improving decision -making. DCEE prefers to prioritize the availability, scalability and compliance and compliance with regulations ensures trouble -free digital adoption and seizes businesses to expand their reach and prosperous on the competitive market.

2.2.1 DRAWBACKS OF EXISTING SYSTEM

- **Limited Digital Adoption** – Many small businesses lack the technical expertise to establish an online presence.
- **Fragmented Tools** – Businesses rely on multiple disjointed platforms for payments, inventory, and customer management, leading to inefficiencies.
- **Poor Demand Forecasting** – Existing systems lack advanced analytics, causing issues like overstocking and stockouts.
- **High Operational Costs** – Managing multiple digital tools increases expenses and complexity for small businesses.

2.3 PROPOSED SYSTEM

The aim of the proposed system for small and local Indian companies is to transform their digital presence and operational efficiency. It provides a user -friendly platform that allows businesses to set up and manage internet shop windows, which significantly increases the visibility and reach of customers. Integrated predictive analytical tools will allow precise prediction of the needs of stocks, minimizing the risks associated with overvaluation and understanding. In addition, the system streamlines payments and offers secure and simplified transaction methods for customers and owners of enterprises. To ensure compliance with local regulations, the function of the built -in user will lead according to the necessary requirements. The system will also include resources and educational programs aimed at improving digital literacy, allowing users to use the platform efficiently and improve their total business operations.

2.3.1 ADVANTAGES OF PROPOSED SYSTEM

- Enhanced Digital Presence
- Improved Inventory Management
- Simplified Payment Processes
- Streamlined Regulatory Compliance
- Increased Digital Literacy and Support
- Greater Customer Trust and Engagement
- Scalable and Flexible Solutions

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

The feasibility study evaluates the viability of the ecosystem of the digital trade (DCEE) by exploring its technical, economic, operational, legal and market aspects. They will assess whether the selected technology magazine (Backend, Bootstrap for Frontend and Mongodb for database) is suitable and compatible with project requirements. Economic analysis focuses on the cost of development, potential income and return on investment. Operational feasibility examines how the system will integrate into current business practices and solve user needs. Legal considerations ensure compliance with data protection regulations and laws on the processing of laws. Finally, the feasibility in the market evaluates the demand for the system between small businesses and ensures that it is in line with the trends on the market and the expectation of customers. The aim of this comprehensive analysis is to determine the overall viability and sustainability of the project.

3.1.1 Economical Feasibility

The economic feasibility study for the ecosystem of digital trade (DCEE) evaluates the financial viability of the project with a focus on costs, income potential and investment return. Initial costs include development, hosting and salaries, while continuing expenses include maintenance and support. Resources of income are expected from subscription fees, transaction fees and advertising. Key components include:

- Market Demand: Evaluating the need for digital solutions.
- Cost-Benefit Analysis: Comparing costs with anticipated benefits.
- Scalability: Potential for service expansion.
- Risk Assessment: Identifying and mitigating financial risks.
- Funding Sources: Exploring grants and investments.
- The study indicates that DCEE can deliver substantial financial benefits and support local business growth.

3.1.2 Technical Feasibility

The technical feasibility study for the ecosystem of digital trade (DCEE) evaluates the technological requirements and capabilities of the project to ensure successful implementation. It examines software, hardware and network infrastructure necessary for efficient development and operation of the platform.

Key components include:

- Technology Stack: Use of bootstrap for the development of quends, cupping API for backend services and Mongodb for database management.

- System Integration: Ensuring trouble -free integration between different modules, including verification, processing of payments, analysis and cottage functions.
- Scalability: Assessing the ability of the system to manage the increased operation of users and data with increasing platform.
- Security Measures: Implementation of robust security protocols for protection of user data and transactions.
- User Support: Providing the necessary training and documentation to users for efficient system use

The study concludes that the DCEE project is technically feasible and uses existing technologies and proven procedures to create a reliable and secure digital trade platform.

3.1.3 Behavioral Feasibility

The behavioral feasibility study for the Digital Commerce Empowerment Ecosystem (DCEE) examines user acceptance, training needs, and potential behavioral changes among digital storefront owners, customers, and administrators. Key aspects include:

- User Acceptance: Assessing the willingness of small businesses to adopt the platform.
- Training Needs: Identifying necessary support for effective platform navigation.
- User Engagement: Understanding expectations to enhance loyalty and usage.
- Behavioral Change: Evaluating how the platform can shift purchasing and management practices.
- Feedback Mechanisms: Establishing channels for continuous improvement based on user input.

This study indicates that with adequate support and engagement strategies, the DCEE project is poised for high user acceptance and positive behavior changes in digital commerce.

3.1.4 Feasibility Study Questionnaire

Project Overview:

The DCEE project is an innovative platform designed to mainly empower small and local Indian businesses to thrive in digital age. It tackles the challenge of the digital divide by offering a comprehensive suite of problems.

Problem: Small businesses lack the resources and know-how to compete effectively online.

Objectives:

- Bridge the digital divide.
- Enhance online presence.
- Build customer trust.
- Facilitate rural payments.
- Streamline support processes.
- Optimize inventory management.
- Simplify regulatory compliance.
- Bridge digital literacy gaps.

System Scope:

The DCEE project is mainly proposed for a prototype like

Focus on the functionalities: The project focuses the functionalities and features of the platform in detail.
Lack of implementation details: There's no mention of the specific implementation plans and, hardware requirements, or deployment strategies.
Emphasis on user needs: The project includes a user questionnaire.

Industry/Domain:
E-commerce industry

Data Collection Contacts:
Name: Shibu
Phone No: 9847510157
Role: Owner of J J Communication, Ranni

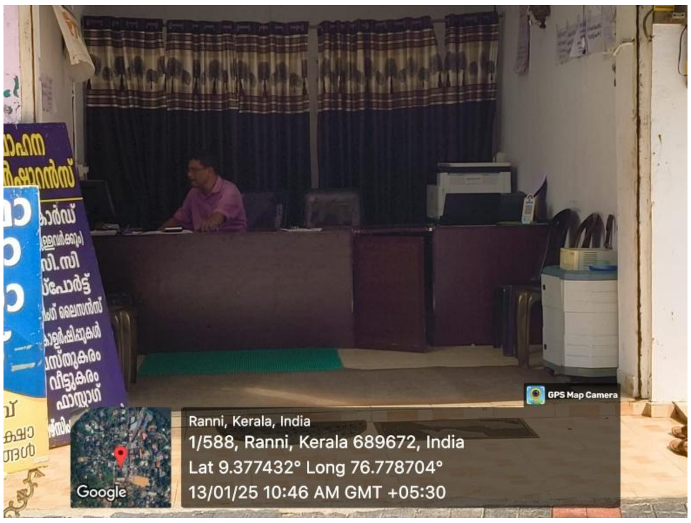
Study Questionnaire

1. What are the biggest challenges you face in running your business online (if applicable)?
    - They do not have a website yet, and managing social media for sales is overwhelming and get a lot of questions through email, and it's hard to keep track of everything.
2. Do you currently have an online store? If yes, what platform do you use?
    - No, they do not have any online platform
3. What are your biggest concerns about adopting a new platform like DCEE? (e.g., cost, learning curve, security)
    - Will DCEE be affordable for a small business like them and they are not very tech-savvy,

so they are worried it will be difficult to learn a new platform.

4. What features are most important to you in an e-commerce platform? (e.g., ease of use, secure payment processing, customer relationship management tools)
   - A user-friendly platform is crucial for them. Safe and secure transactions are essential for building customer trust.
5. How comfortable are you with using technology?
   - They are somewhat comfortable with technology, but not an expert
6. Would you be interested in educational resources to help you navigate the DCEE platform and improve your online presence?
   - Educational resources would be a huge help for the business and any resources that can help improve online presence
7. What is your typical budget for marketing and online tools?
   - They have not allocated a specific budget
8. How many employees do you have in your business?
   - There are no employees only the owner and his wife
9. What are your long-term goals for growing your business online?
   - The goal is to establish a strong online presence and increase my customer base
10. Would you be interested in a free trial of the DCEE platform to see if it meets your needs?
    - Yes, they are interested to free trial in DCEE

3.1.5 Geotagged Photograph


GPS Map Camera
Google
Ranni, Kerala, India
1/588, Ranni, Kerala 689672, India
Lat 9.377432° Long 76.778704°
13/01/25 10:46 AM GMT +05:30

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- Intel Core i3
RAM	- 8 GB
Hard disk	- 256 GB SSD or higher

3.2.2 Software Specification

Front End	-	BOOTSTRAP
Back End	-	FLASK, Cloud Mongo
Database	-	MongoDB Compass
Client on PC	-	Windows 7 and above.

Technologies used - PYTHON, JS, HTML5, AJAX, FLASK, Pytorch (for deploy model), Lang Chain (for AI Workflow), Google AI Studio (model inference)

3.3 SOFTWARE DESCRIPTION

3.3.1 Eg.FLASK

The flask is a lightweight and flexible web frame written in Python, designed to quickly create web applications. It monitors a modular approach, offers only basic tools, and allows developers to add extensions as needed to connect to database, forms processing, verification and more. The flask works on the WSGA (Interface Web Server Gateway Interface) and uses for the Jinja2 Templing and provides robust tools for rendering a dynamic HTML with minimal effort. Being a "micro" is a non -optionate flask, which means that it does not force any project structure or addiction . The flask is a powerful but minimalist web frame written in Python, ideal for developers who prefer simplicity and checking in the development of web applications. As a "micro line", the flask includes only basic components for the requirements for handling, routing and Templing, which is exceptionally light and flexible. To process web requirements and Yinja2, fast and expressive Templing Engine uses to create dynamic answers HTML WSGA (Interface Web Server interface Web Server Gateway). One of the Blacker features is its extensibility - developers can choose specific libraries or extensions for functions such as database integration, form and verification verification without unnecessary dependencies. This "plug-and-play" approach allows developers to create simple applications and scalable systems at the production level. The flask is also highly compatible with other Python libraries that are popular for projects that include machine learning, data visualization or comprehensive business logic. Thanks to their direct syntax and clear documentation, they make the best choice for beginners, while its flexibility and powerful extensions attract experienced developers who need adapted high performance magazine.

3.3.2Eg. MongoDB

MongoDB is a nosql database known for its scalability, flexibility and structure based on documents. Unlike traditional relational databases that store data in tables with rows and columns, Mongodb stores data in Bson (binary JSon) documents, allowing it to effectively manage unstructured or semi -structured data. Each document is a separate data unit, making Mongodb suitable for applications that require rapid data integration, real -time analytics, and frequent data structure updates without disrupting the entire system. Nature without the Mongodb scheme allows developers to easily add or modify the fields without predefined data schemes. Thanks to this flexibility, it is ideal for projects that develop or manage large volumes of different data types such as social media applications, electronic trading platforms and IoT solutions. It also supports the rich language of queries, indexing, aggregation and powerful functions such as horizontal scaling and high availability replications, which increases its performance and tolerance of failures. Mongodb integrates well with modern technological piles, making it a popular choice for developers and organizations with a full stack and accepting a cloud genus, distributed architecture of applications.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any technical product or system. Design is a creative process. Good design is the key to an efficient system. The term "design" is defined as "the process of using different techniques and principles to define a process or system in sufficient detail to allow its physical realization". It can be defined as the process of using different techniques and principles to define a device, process, or system in a sufficient detailed system to make it possible to make it physical. Software design sits in the technical core of the software engineering process and is applied regardless of the developmental paradigm used. The design of the system develops architectural details needed to create a system or product. As with any systematic approach, this software has also undergone the best possible design phase of fine tuning all efficiency, performance and accuracy. The design phase is a transition from a user -oriented document to a document for programmers or database staff. The system design goes through two phases of development: logical and physical design.

4.2 UML DIAGRAM

UML is a standard language for specification, visualization, design and documenting artifacts of software systems. UML was created by a group for managing objects (OMG) and designing the UML 1.0 specification was designed to OMG in January 1997. UML means a uniform modeling language. UML differs from other common programming languages such as C ++, Java, Cobol, etc. UML is a picture language used to create software plans. UML can be described as a language of visual modeling for visualization, specification, design and software design. Although UML is generally used to model software systems, it is not limited at this limit. It is also used to model systems without software. For example, the process flow in the manufacturing unit, etc. UML is not a programming language, but the tools can be used to generate code in different languages using UML diagrams. UML has a direct relation to object -oriented analysis and design. After some standardization, UML has become the standard of OMG. All elements, relationships are used to create a complete UML diagram and the diagram is a system. The visual effect of the UML diagram is the most important part of the process. All other elements are used to complete. UML contains the following nine diagrams.

- Class diagram
- Object diagram

- Use case diagram
- Sequence diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

A Use Case Diagram is a visual representation of a system's functionality, highlighting the interactions between users (actors) and the system itself. It serves as a high-level overview, showing how users achieve their goals by utilizing the system's various functions, represented as "use cases." These diagrams are commonly used in the early stages of software design to communicate system requirements, focusing on what the system does rather than how it does it.

In a Use Case Diagram:

- Actors represent the users or external systems that interact with the system, such as admins, customers, or third-party services.
- Use Cases of use represent the specific functions or actions that the system performs, such as "login", "register", "shopping product" or "display orders".
- Associations are lines connecting actors and cases of use, indicating interactions or relationships.
- The system limit is a rectangle that encapsulates all cases of use and defines the range of the system.

Use diagrams are useful for understanding user requirements, identifying main features and defining roles in the system. It also serves as a basis for further detailed analysis and design and helps ensure that the system is in line with the user needs.

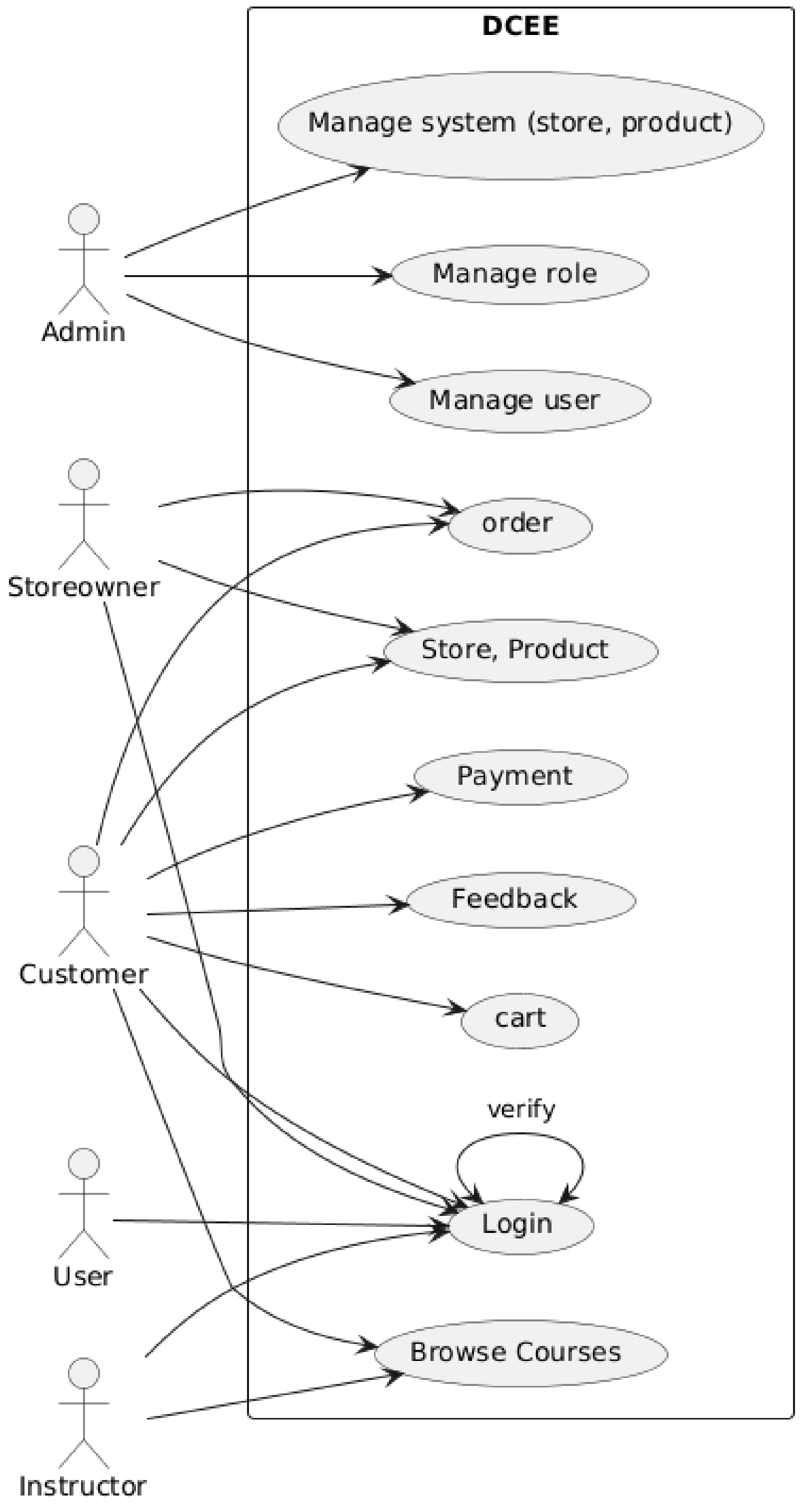


Figure 1. Use Case Diagram

4.2.2 SEQUENCE DIAGRAM

The sequential diagram is the type of interaction diagram in the uniform modeling language (UML), which shows how objects interact in a specific sequence in the system. Visually represents the flow of messages or events between objects over time and shows how and in what order the operations are performed. The key components of the sequence diagram include:

Actors: They represent users or external systems interacting with the system.

Objects or Classes: Represent entities within the system that perform actions.

Lifelines: Vertical lines showing the life span of an object during the sequence.

Activation Bars: Rectangles on the lifelines indicating the period when an object is active.

Messages: Arrows between objects indicating communication; they can be synchronous or asynchronous.

Return Messages: Dashed lines that show the return or response after a message is processed.

Use and Advantages:

Sequence diagrams are integral to modelling the dynamic aspects of systems and are especially useful during the design phase for:

- Clear Communication: They provide a visual, step-by-step breakdown of processes, improving communication among team members.
- Requirement Validation: Diagrams help validate requirements by demonstrating how processes should flow and helping stakeholders confirm intended functionality.

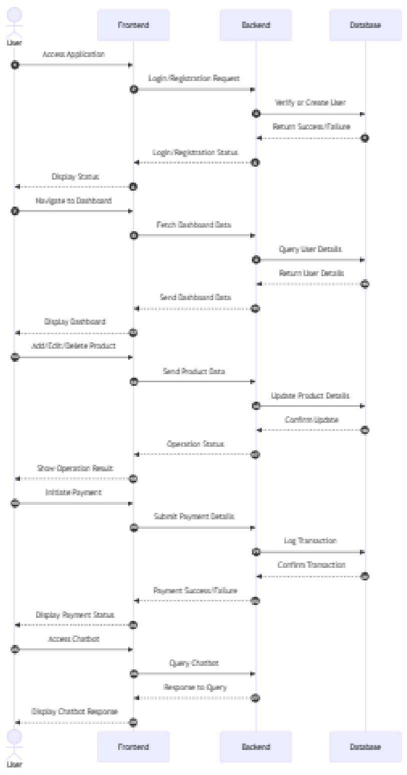


Figure 2. Sequence Diagram

4.2.3 State Chart Diagram

A state chart diagram, or state diagram, illustrates the various states an object or system component can undergo throughout its lifecycle, along with the transitions between those states triggered by events. Each state is represented as a rounded rectangle, while transitions are shown as arrows connecting the states, labeled with the events that cause the transitions.

Key Elements:

States: Conditions an object can be in (e.g., Idle, Processing).

Transitions: Arrows indicating changes between states, triggered by events.

Events and Actions: Triggers for state changes and activities that occur during transitions.

Use and Benefits:

State chart diagrams model dynamic behavior, simplify complex systems, and validate that all expected states and transitions are covered. They are commonly used in applications with state-dependent behavior, such as user interfaces and workflow management systems, providing a clear view of how an object responds to various events.

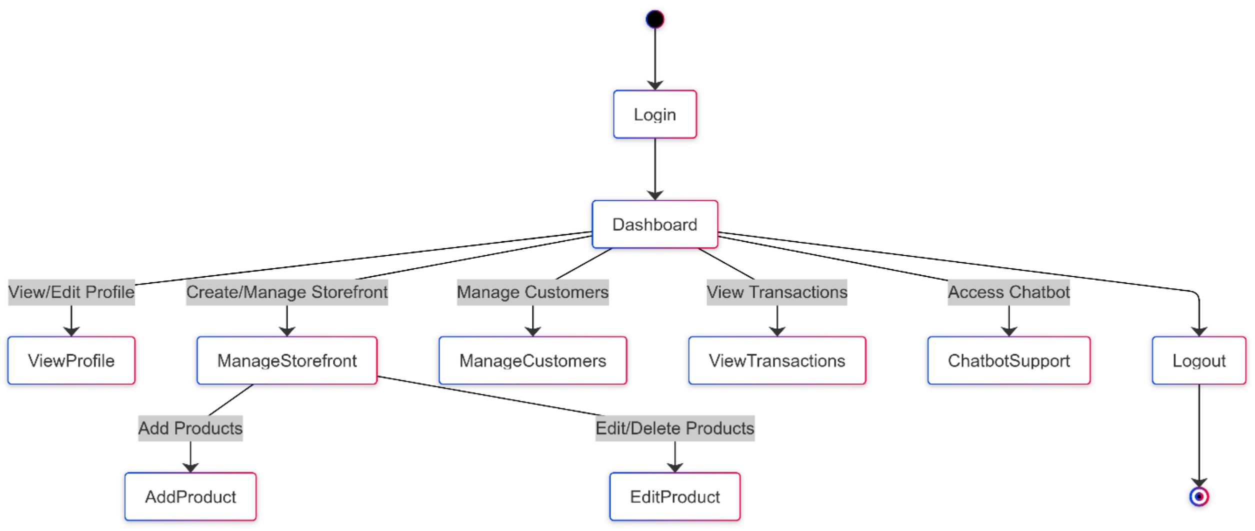


Figure 3. State Chart Diagram

4.2.4Activity Diagram

The activity scheme is a diagram of behavior in the language of unified modeling (UML), which illustrates the flow of activities or actions in the system and emphasizes the dynamic aspects of processes. It is equipped with key elements such as activities represented by rounded rectangles that indicate tasks performed in the system. The scheme begins with the initial node displayed as a filled circle, and closes the end node, a circle enclosed in another circle, indicating the completion of the process. Transitions represented by arrows connect activities to demonstrate the flow from one task to another. The decision nodes, illustrated as diamonds, indicate points where the flow may vary on the basis of specific conditions, while the fork and connections (displayed as rods) represent parallel activities. Activities schemes are particularly valuable for modeling complex workflows, scenarios of use and business processes because they clarify sequence of operations, identify potential obstacles and facilitate communication between the parties involved in visualization of activities and their interdependence.

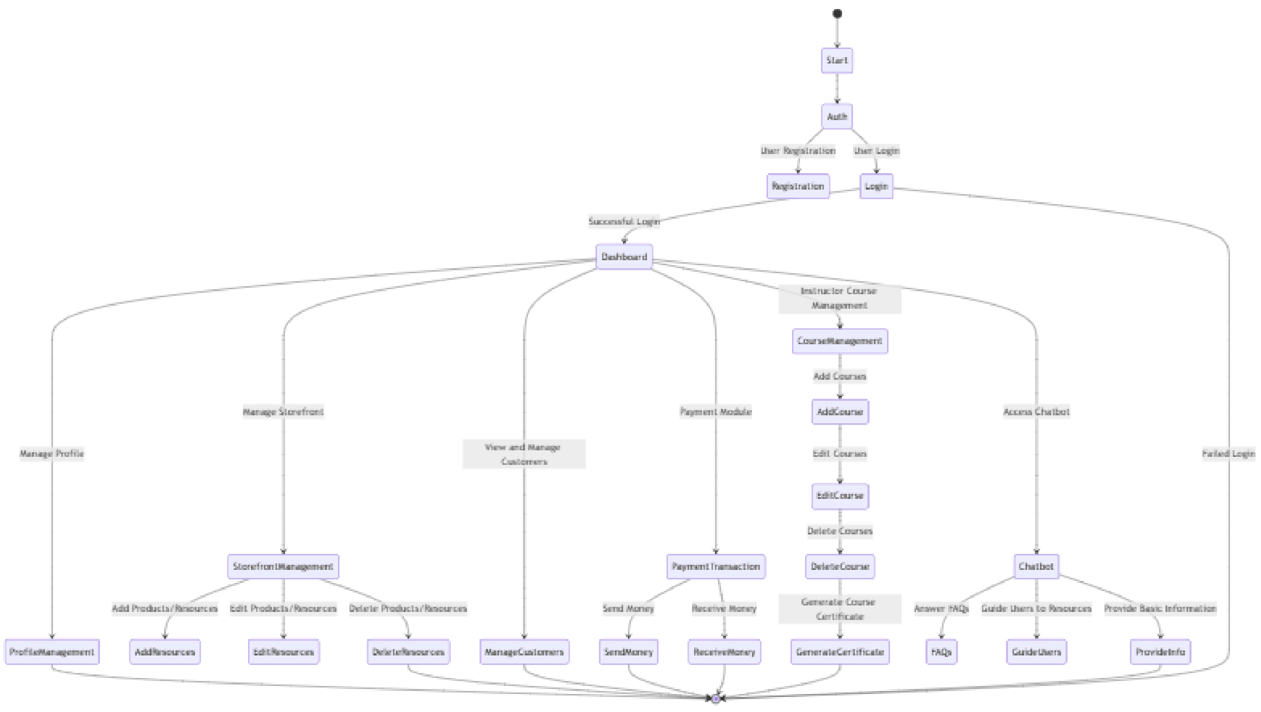


Figure 4. Activity Diagram

4.2.5 Class Diagram

The class scheme is a static structural diagram in a unified modeling language (UML), which represents the system classes, their attributes, methods and relationships between them. It provides a visual plan of the system architecture and emphasizes how different classes interact and how data is organized. Each class is displayed as a rectangle divided into three sections: the upper part contains the name of the class, the middle part states its attributes and the lower part shows its methods (or operations). The relationships between classes are illustrated by lines, with different notations that indicate a type of relationship such as inheritance (generalization), associations, aggregation and composition. Class diagrams are essential for object -oriented design because they help understand the system structure, lead class implementation and ensure that all components interact correctly. It also serves as a communication tool between developers, designers and parties involved in providing a clear representation of the data model of the system and its relationships.

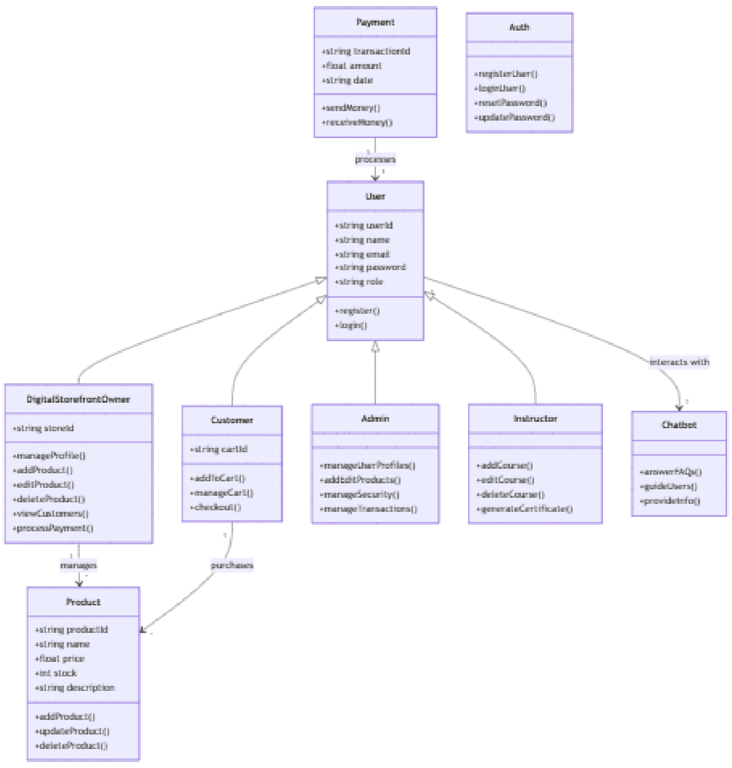


Figure 5. Class Diagram

4.2.6 Object Diagram

The object scheme is a type of static structure diagram in the language of unified modeling (UML), which provides a picture of the instances of classes (objects) and their relations at a certain point. Unlike class diagrams that focus on the class and their relationship plan, object diagrams emphasize specific examples of these classes in a particular condition. Each object is represented by a rectangle, a similar class in a class diagram, but contains the name of the object and its current state values or attributes. The lines connecting objects illustrate their relationships such as associations, aggregations or compositions. Object diagrams are particularly useful for visualizing complex relationships and interactions between objects in the system, showing how they work together to perform tasks or represent data.

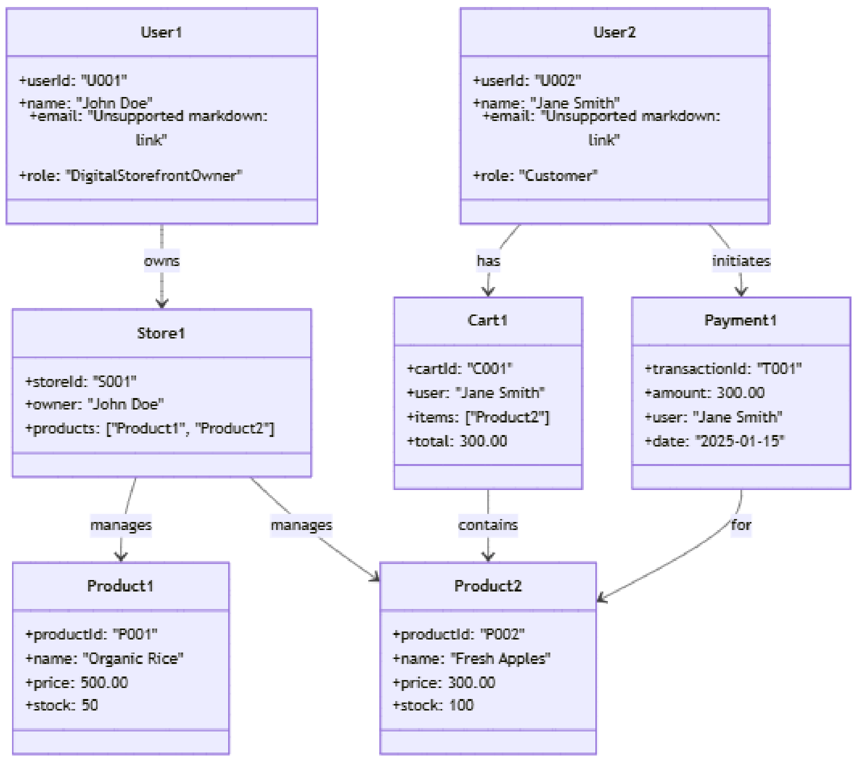


Figure 6. Object Diagram

4.2.7 Component Diagram

The component scheme is a type of structural diagram in a uniform language of unified modeling (UML), which represents physical components in the system such as software applications, libraries and packages. It illustrates how these components interact and their dependence, and provides a high level of view of the system architecture. The components are displayed as rectangles with two smaller rectangles on the left side, indicating that they can be independently developed and deployed. They may also include interfaces that define operations available for other components. Components schemes are particularly useful for modeling complex systems because they help visualize relationships between different software components, their functions and how they fit into the overall architecture of the system. This helps understand the modular system structure, makes it easier to plan system integration and increase communication between developers.

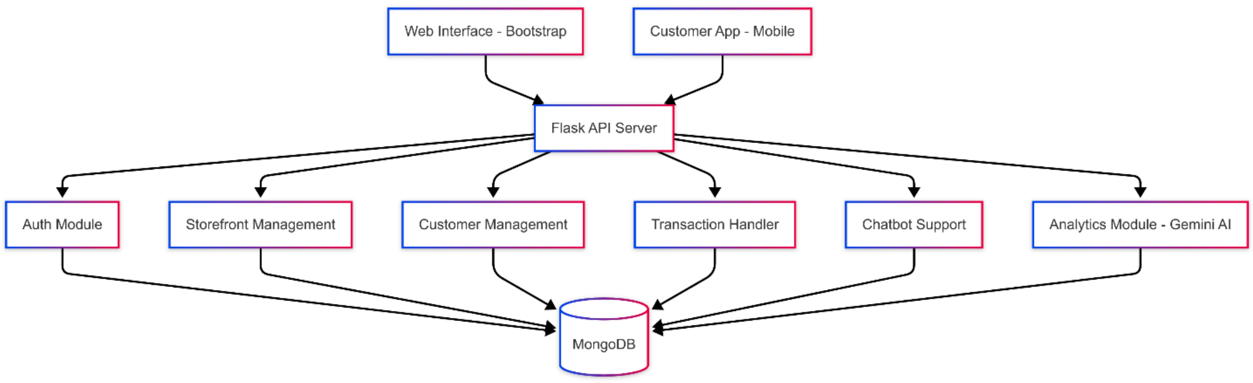
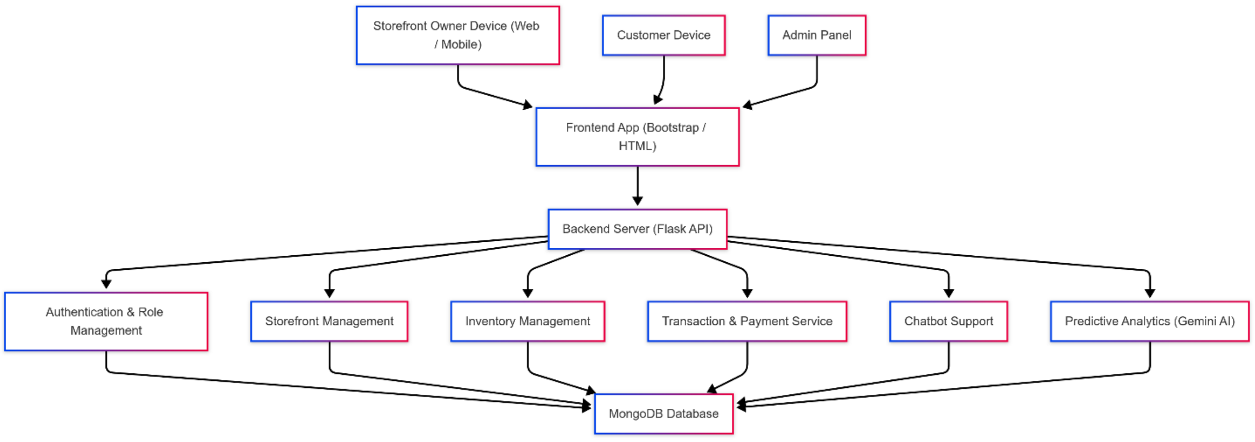


Figure 7. Component Diagram

4.2.8 Deployment Diagram

The deployment diagram is another type of UML diagram that shows the physical deployment of artifacts on the knots. It illustrates how software is distributed across hardware and how different components communicate within system infrastructure. Deployment schemes are particularly useful in visualizing the physical arrangement of software components in a distributed system, such as client-server architectures or cloud applications. In the deployment diagram, the nodes of the physical device (such as servers, computers or mobile devices) are, and artifacts (such as executable files, libraries or database schemes) are software deployed in these nodes. The connection between the nodes shows the communication paths and shows how the information flows in the system. This diagram is necessary to understand the system architecture and can help in assessment of performance, scalability and reliability.

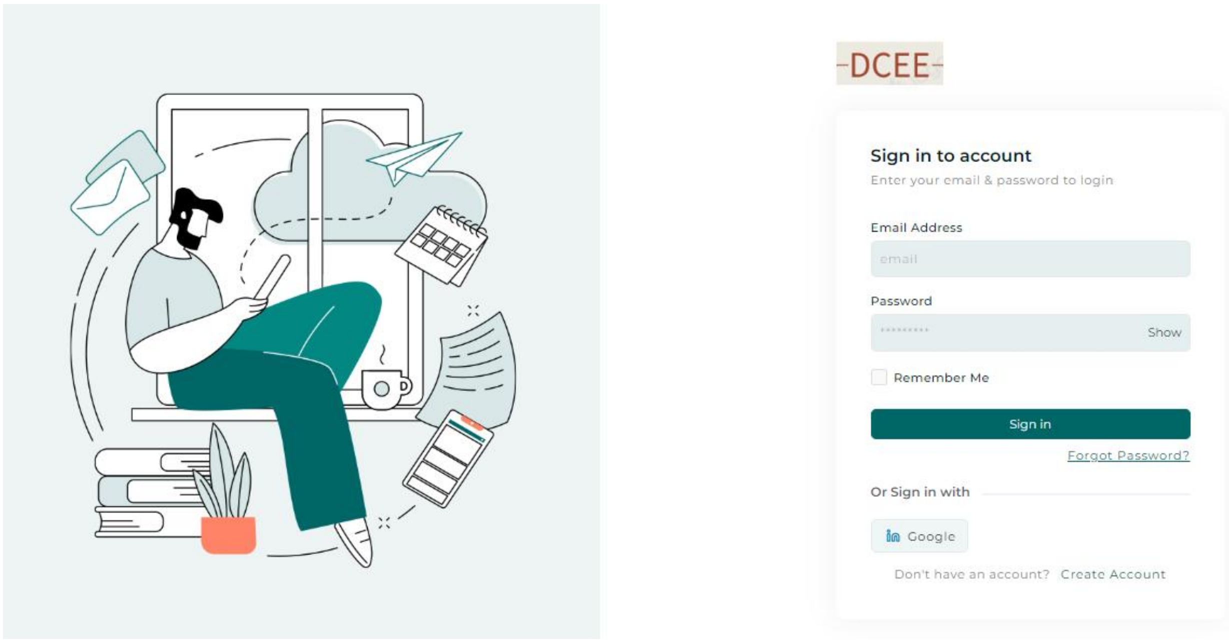


Storefront Owner Device (Web / Mobile)
Customer Device
Admin Panel
Frontend App (Bootstrap / HTML)
Backend Server (Flask API)
Authentication & Role Management
Storefront Management
Inventory Management
Transaction & Payment Service
Chatbot Support
Predictive Analytics (Gemini AI)
MongoDB Database

Figure 8. Deployment Diagram

4.3 USER INTERFACE DESIGN USING FIGMA

Form Name: Login



-DCEE-

Sign in to account

Enter your email & password to login

Email Address

email

Password

********* Show

Remember Me

Sign in

Forgot Password?

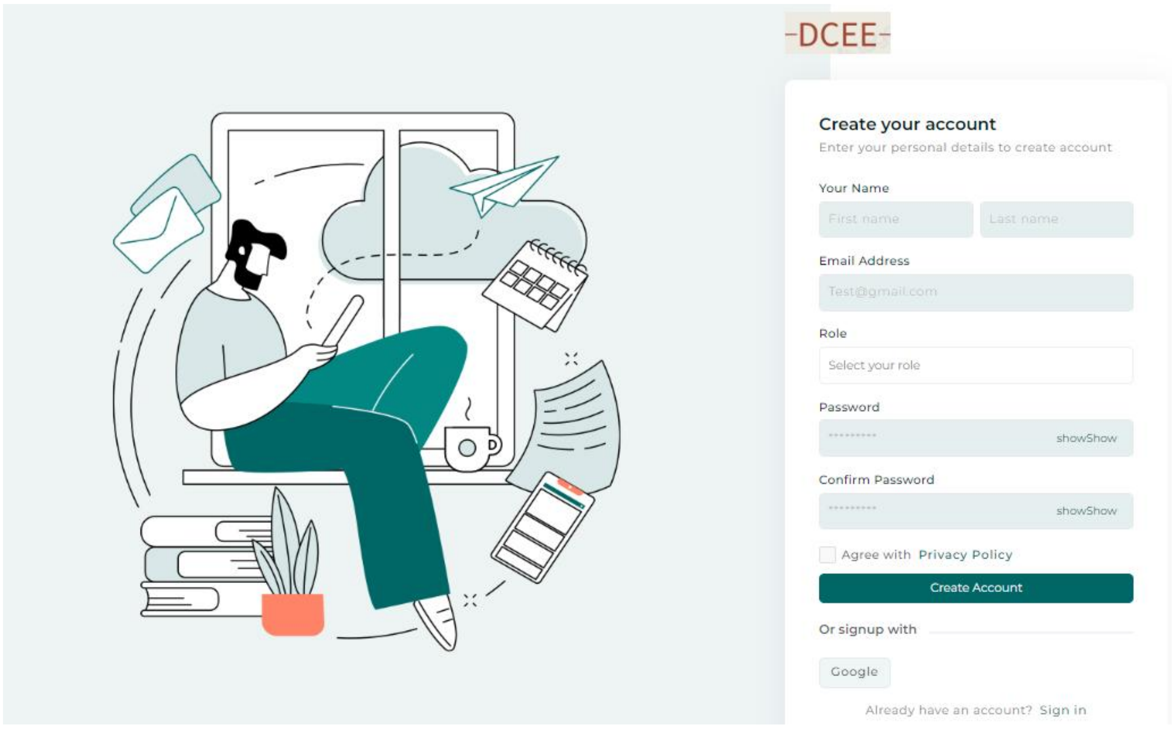
Or Sign in with

Google

Don't have an account? Create Account

Figure 1. Login Page

Form Name: Register



-DCEE-

Create your account

Enter your personal details to create account

Your Name

First name Last name

Email Address

Test@gmail.com

Role

Select your role

Password

********* showShow

Confirm Password

********* showShow

Agree with Privacy Policy

Create Account

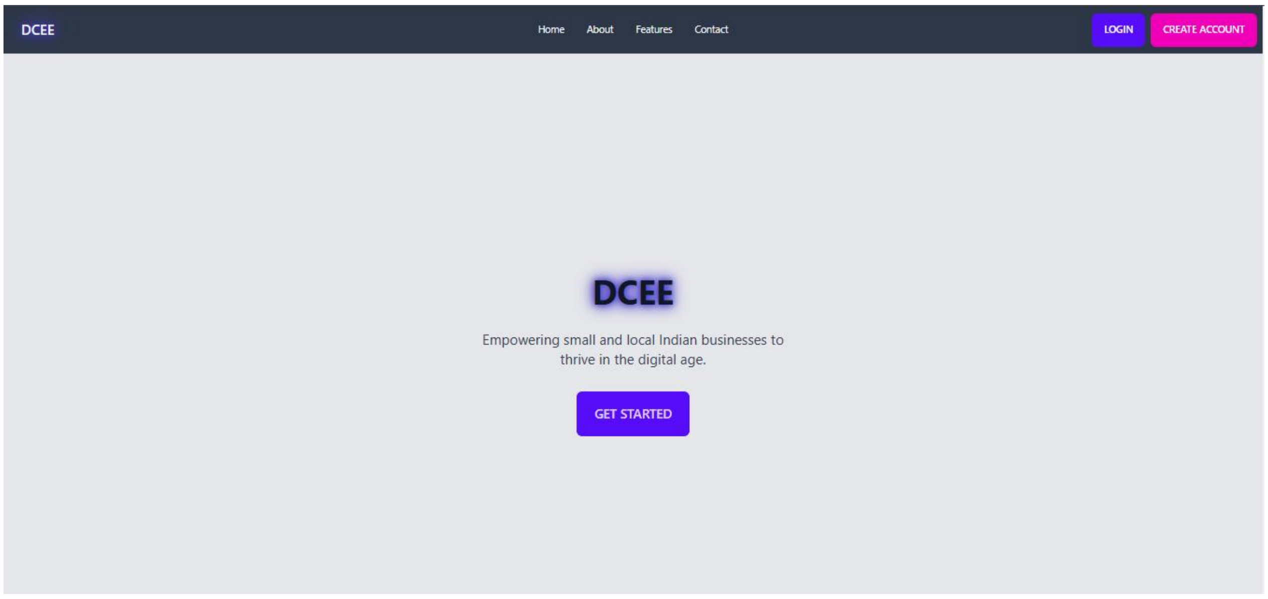
Or signup with

Google

Already have an account? Sign in

Figure 2. Register Page

Form Name: Home Page



Empowering small and local Indian businesses to thrive in the digital age.

Figure 3. Landing Page

Form Name: Customer Dashboard

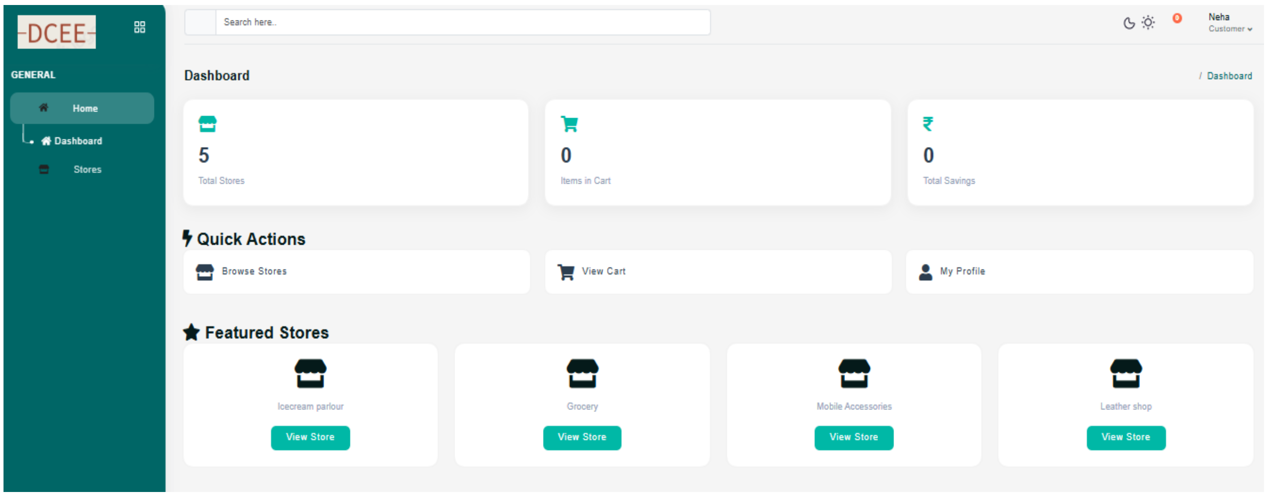
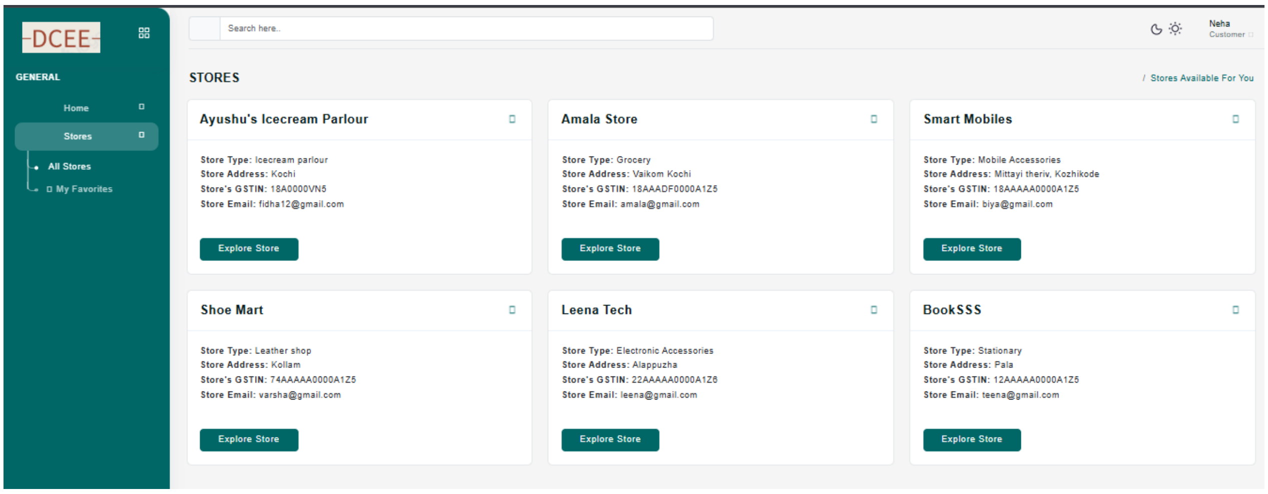


Figure 4. Customer Dashboard

Form Name: Stores Page



Ayushu's Icecream Parlour
Store Type: Icecream parlour
Store Address: Kochi
Store's GSTIN: 18A0000VN5
Store Email: fidha12@gmail.com
Explore Store

Amala Store
Store Type: Grocery
Store Address: Vaikom Kochi
Store's GSTIN: 18AAADF0000A1Z5
Store Email: amala@gmail.com
Explore Store

Smart Mobiles
Store Type: Mobile Accessories
Store Address: Mittayi theriv, Kozhikode
Store's GSTIN: 18AAAAA0000A1Z5
Store Email: biya@gmail.com
Explore Store

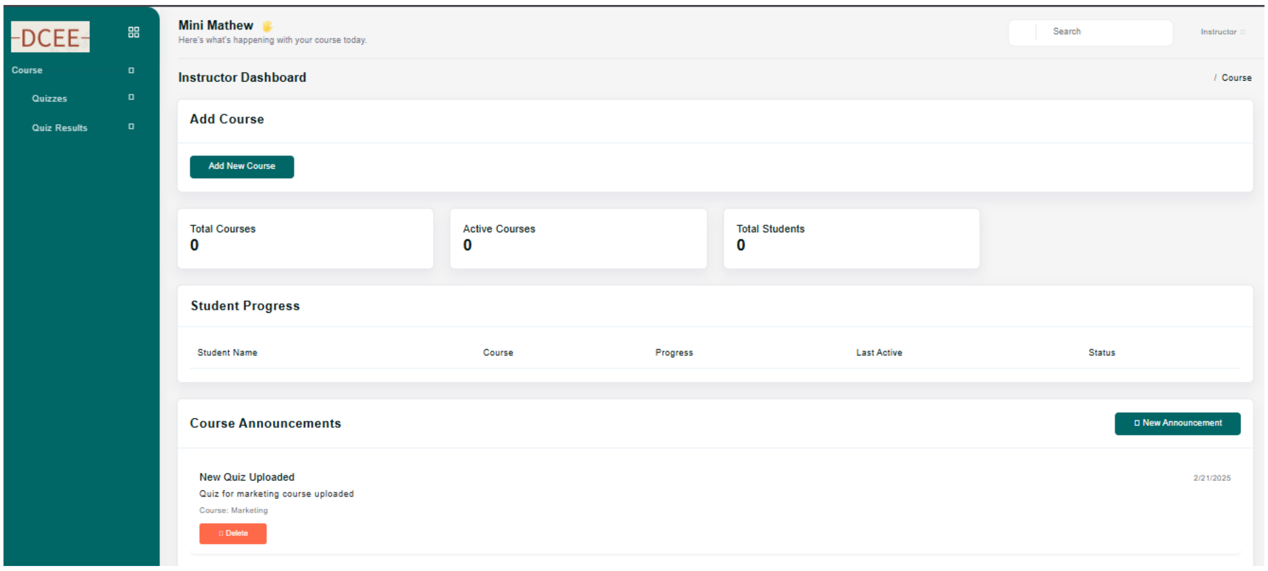
Shoe Mart
Store Type: Leather shop
Store Address: Kollam
Store's GSTIN: 74AAAAA0000A1Z5
Store Email: varsha@gmail.com
Explore Store

Leena Tech
Store Type: Electronic Accessories
Store Address: Alappuzha
Store's GSTIN: 22AAAAA0000A1Z6
Store Email: leena@gmail.com
Explore Store

BookSSS
Store Type: Stationary
Store Address: Pala
Store's GSTIN: 12AAAAA0000A1Z5
Store Email: teena@gmail.com
Explore Store

Figure 5. Stores Display

Form Name: Instructor Dashboard



Mini Mathew
Here's what's happening with your course today.

Instructor Dashboard

Add Course
Add New Course

Total Courses	Active Courses	Total Students
0	0	0

Student Progress

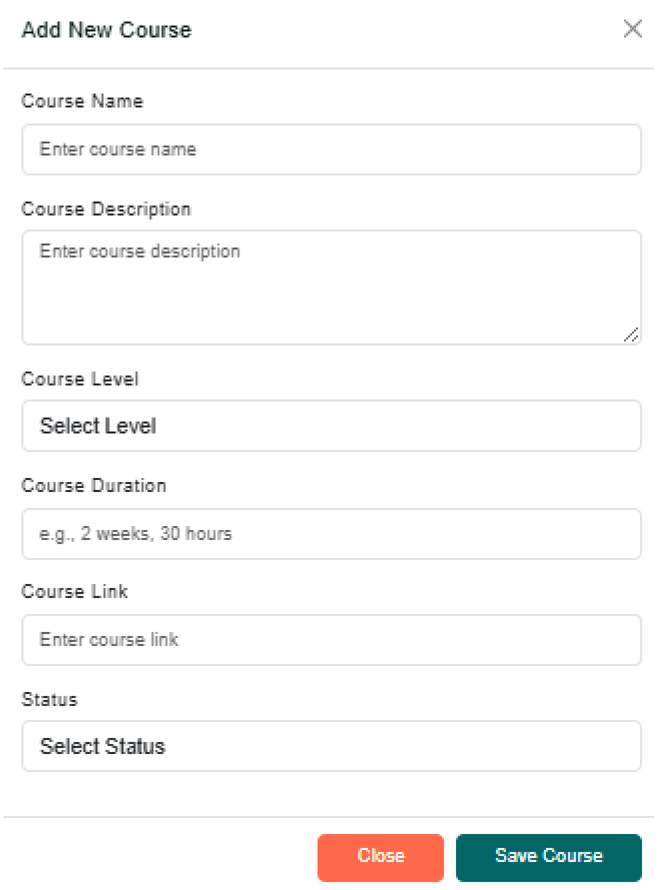
Student Name	Course	Progress	Last Active	Status

Course Announcements
New Announcement

New Quiz Uploaded — 2/21/2025
Quiz for marketing course uploaded
Course: Marketing
Delete

Figure 6. Instructor Dashboard

Form Name: Add new course


Add New Course

Course Name

Enter course name

Course Description

Enter course description

Course Level

Select Level

Course Duration

e.g., 2 weeks, 30 hours

Course Link

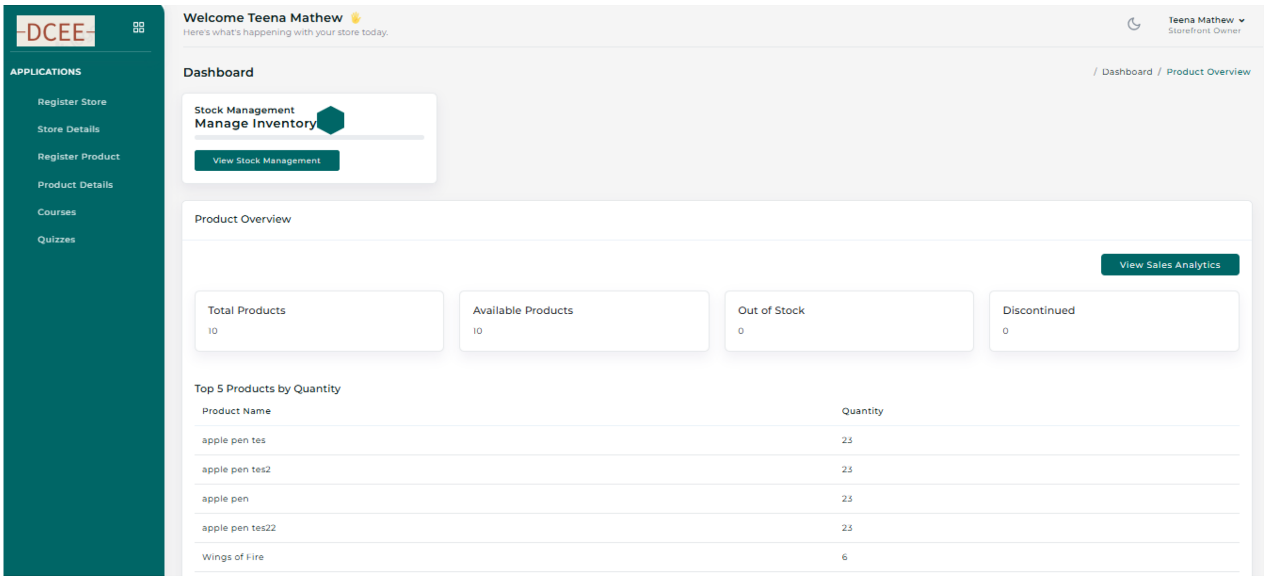
Enter course link

Status

Select Status

Figure 6. New Course Add

Form Name: Storeowner Dashboard



Welcome Teena Mathew

Here's what's happening with your store today.

Dashboard

Stock Management

Manage Inventory

Product Overview

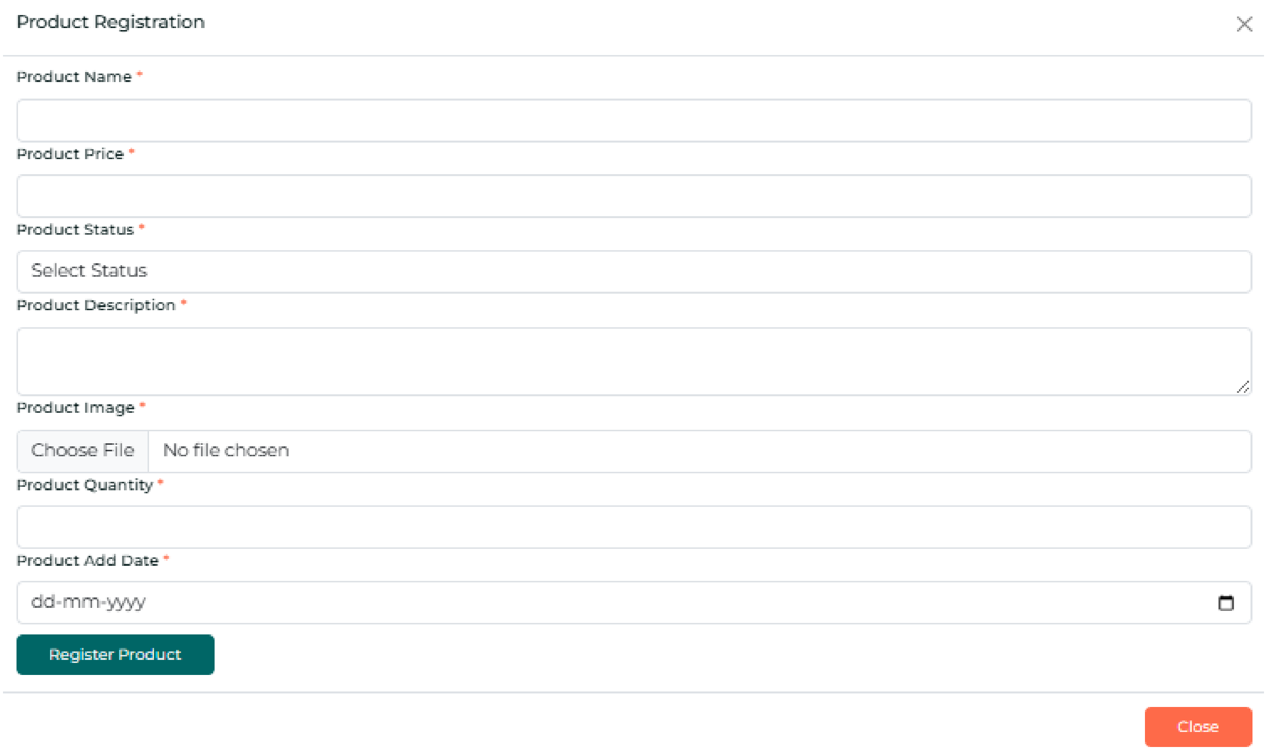
Total Products	10
Available Products	10
Out of Stock	0
Discontinued	0

Top 5 Products by Quantity

Product Name	Quantity
apple pen tes	23
apple pen tes2	23
apple pen	23
apple pen tes22	23
Wings of Fire	6

Figure 7. Storeowner Dashboard

Form Name: Product Registration


Product Registration

Product Name *

Product Price *

Product Status *

Select Status

Product Description *

Product Image *

Choose File No file chosen

Product Quantity *

Product Add Date *

dd-mm-yyyy

Figure 8. Product Registration

4.4 DATABASE DESIGN

Database design is a data structure properly to ensure efficient storage, search and management. DCEE is a system supported by RDBMS and the data is stored in tables with primary and foreign keys to ensure relationships between tables. The central design points are reduced redundancy, maintain data integrity and optimize performance for managing user accounts, products, rent and transactions.

4.4.1 Non - Relational Database Management System (RDBMS)

In the non-relay database (NOSQL), the data is organized more in collections and documents than in tables and lines, allowing flexible schemes where each document in the collection can have a unique structure. Documents, similar to objects, contain fields that can store various types of data, including fields or nested documents, allowing rich and hierarchical data structures. Documents, similar to objects, contain fields that can store diverse data types, including arrays or nested documents, enabling rich and hierarchical data structures. Relationships are managed through embedding related data directly within a document or referencing other documents by ID, rather than enforcing strict referential integrity. Fields and data types are not strictly bound, allowing for flexibility in handling various data formats. Atomicity applies at the document level, ensuring that operations on a document are consistent as a whole. This flexibility and schema-less nature make NoSQL databases ideal for applications requiring scalability and rapid data access across large volumes of unstructured or semi-structured data.

4.4.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys, and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on another non-key attribute.

Fourth Normal Form

The fourth normal form (4NF) is a database normalization rule that further refines data modeling by addressing multi-valued dependencies. When a table contains multiple independent sets of repeating data, we can break it down into smaller tables, with each table containing only one set of related data.

This reduces data redundancy and improves data consistency by ensuring that each table represents a single, well-defined concept or entity. To achieve a 4NF, we must ensure that all more valuable dependencies are removed from the table and that each table contains only attributes that are functionally dependent on the primary key.

Fifth Normal Form

Fifth normal form 5NF is truly the highest level of normalization in the design of a relational database and deals with complex data models that include more overlapping more valuable addictions. In 5NF, the tables are spread over smaller tables to eliminate any possible redundancy caused by overlapping addiction, ensuring that data is not lost. The aim of the 5NF is to ensure that each table represents a single entity or relationship and that data is organized in a way that minimizes redundancy, eliminates anomalies and improves data integrity.

4.4.3 Sanitization

An automated procedure called "disinfection" is used to readiness for use in a SQL query. This process usually involves checking the value for characters that have a special meaning for the target database. To prevent a SQL injection attack, you must sanitize(filter) the input string while processing a SQL query based on user input. For instance, the user and password input is a typical scenario. In that scenario, the server response would provide access to the 'target user' account without requiring a password check.

4.4.4 Indexing

By reducing the number of disk accesses needed when a query is completed, indexing helps a database perform better. It is a data structure method used to locate and access data in a database rapidly. Several database columns are used to generate indexes. The primary key or candidate key of the table is duplicated in the first column, which is the Search key. To make it easier to find the related data, these values are kept in sorted order. Recall that the information may or may not be kept in sorted order.

4.5 TABLE DESIGN

1.Tbl_roles

Eg.Primary key: roleId

Eg.Foreign key: roleId references table Tbl_user

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	roleId	int	PRIMARY KEY	Unique identifier for the role
2.	roleName	varchar	NOT NULL	Name of the role (e.g., admin, customer)
3.	status	varchar	NOT NULL	Status of the role (e.g., active, inactive)

2.Tbl_ users

Eg.Primary key: userId

Eg.Foreign key: userId references table Tbl_roles

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	userId	int	PRIMARY KEY	Unique identifier for the user

2.	email	varchar	NOT NULL	User's email address
3.	passwordHash	varchar	NOT NULL	Hashed password for the user
4.	roleId	int	FOREIGN KEY	User's assigned role ID (references roles)
5.	googleAuthId	int	NOT NULL	Google authentication ID (if applicable)
6.	status	varchar	NOT NULL	Status of the role (e.g., active, inactive)

3.Tbl_ stores

Eg.Primary key: **stores**

Eg.Foreign key: **stores** references table Tbl_products

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	storeId	int	PRIMARY KEY	Unique identifier for the store
2.	userId	int	FOREIGN KEY	User ID who owns the store (references users)
3.	storeName	varchar	NOT NULL	Name of the store
4.	storeDesc	varchar	NOT NULL	Description of the store
5.	status	varchar	NOT NULL	Status of the store (e.g., active, inactive)
6.	totalOrders	Int	NOT NULL	Total number of orders placed for the store
7.	totalProducts	int	NOT NULL	Total number of products in the store
8.	store_img	varchar	NOT NULL	URL or path to the store image

4.Tbl_ products

Eg.Primary key: **products**

Eg.Foreign key: **products** references table Tbl_ **orderItems**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	prodId	int	PRIMARY KEY	Unique identifier for the product
2.	storeId	varchar	FOREIGN KEY	Store ID where the product belongs (references stores)
3.	prodName	varchar	NOT NULL	Name of the product
4.	prodDesc	varchar	NOT NULL	Description of the product

5.	prodPrice	int	NOT NULL	Price of the product
6.	status	varchar	NOT NULL	Status of the product (e.g., active, inactive)
7.	prod_img	varchar	NOT NULL	URL or path to the product image
8.	prod_qnty	int	NOT NULL	Quantity of the product in stock
9.	prod_add_date	Date	NOT NULL	Date the product was added to the store
10.	refill_date	date	NOT NULL	Date the product needs to be restocked (optional)

5.Tbl_ orderItems

Eg.Primary key: orderId

Eg.Foreign key: orderId references table Tbl_ **orders**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	orderId	int	FOREIGN KEY	Unique identifier for the order
2.	prodId	int	FOREIGN KEY	Product ID of the ordered item (references products)
3.	quantity	int	NOT NULL	Quantity of the product ordered
4.	price	int	NOT NULL	Price of the ordered product

6.Tbl_ orders

Eg.Primary key: orderId

Eg.Foreign key: orderId references table Tbl_products

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	orderId	int	PRIMARY KEY	Unique identifier for the order
2.	custId	varchar	FOREIGN KEY	Customer ID who placed the order (references customers)
3.	storeId	varchar	NOT NULL	Store ID from where the order was placed (references stores)
4.	status	varchar	NOT NULL	Status of the order (e.g., pending, processing, shipped)

7.Tbl_ customers

Eg.Primary key: custId

Eg.Foreign key: custId references table Tbl_products

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	custId	int	PRIMARY KEY	Unique identifier for the customer
2.	userId	varchar	FOREIGN KEY	User ID associated with the customer (references users)
3.	custHist	varchar	NOT NULL	Customer's order history
4.	wishlist	varchar	NOT NULL	Customer's wishlist of products
5.	custCo	varchar	NOT NULL	Courses enrolled in by the customer
6.	custAddr	varchar	NOT NULL	Customer's shipping address
7.	status	varchar	NOT NULL	Status of the customer

8.Tbl_ payment

Eg.Primary key: payId

Eg.Foreign key: payId references table Tbl_order

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	payId	int	PRIMARY KEY	Unique identifier for the payment
2.	orderId	int	FOREIGN KEY	Order ID associated with the payment (references orders)
3.	payAmnt	int	NOT NULL	Amount paid for the order
4.	payStat	varchar	NOT NULL	Payment status (e.g., pending, completed, failed)
5.	payMethod	varchar	NOT NULL	Payment method used (e.g., credit card, debit card)
6.	payDate	date	NOT NULL	Date the payment was made
7.	custBilling	varchar	NOT NULL	Customer's billing address
8.	status	varchar	NOT NULL	Status of the payment (e.g., active, refunded)

9.Tbl_ instructor

Eg.Primary key: instId

Eg.Foreign key: userId references table Tbl_customer

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	instId	int	PRIMARY KEY	Unique identifier for the instructor

2.	userId	int	FOREIGN KEY	User ID associated with the instructor (references users)
3.	instName	varchar	NOT NULL	Full Name of the Instructor
4.	instBio	varchar	NOT NULL	Instructor's Biography or Description
5.	instExpertise	varchar	NOT NULL	Instructor's areas of expertise
6.	status	varchar	NOT NULL	Status of the instructor (e.g., active, inactive)
7.	createdAt	timestamp	NOT NULL	Date and time the instructor record was created

10.Tbl_ course

Eg.Primary key: courseId

Eg.Foreign key: userId references table Tbl_customer

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1.	courseId	int	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for the course
2.	instId	int	FOREIGN KEY REFERENCES Instructor(instId),	Instructor ID associated with the course (references Instructor)
3.	courseName	varchar(255)	NOT NULL	Name of the course
4.	courseDesc	varchar(255)	NOT NULL	Description of the course
5.	courseLevel	varchar(50)	NOT NULL	Level of the course (e.g., Beginner, Intermediate, Advanced)
6.	courseDuration	varchar(50)	NOT NULL	Duration of the course (e.g., "2 weeks", "3 months")
7.	coursePrice	decimal(10, 2)	NOT NULL	Price of the course
8.	courseImg	varchar(255)	NOT NULL	URL or path to the course image
9.	status	varchar(50)	NOT NULL	Status of the course (e.g., active, inactive, draft)
10.	createdAt	timestamp	NOT NULL, DEFAULT: CURRENT_TIMESTAMP	Date and time the course was created

CHAPTER 5
SYSTEM TESTING

5.1 INTRODUCTION

Software Testing is the software implementation process in a controlled way to answer the question - does the software behave as mentioned? Software testing is often used in conjunction with verification and verification of the term. Validation is inspection or testing of items, includes software for compliance and consistency with associated specification. Software testing is just one type of verification that also uses techniques such as reviews, analysis, inspection and passages. Validation is a process of control that what was specified is what the user wanted. Other activities that are often associated with software testing are static analysis and dynamic analysis. Static analysis examines the source code of the software, looks for problems and collects metrics without code. Dynamic analysis focuses on software behavior, providing information such as traces of implementation, timing profiles and test coverage information. Testing is a set of activity that can be planned in advanced and systematically performed. Testing starts at the module level and works on the integration of the entire computer -based system. Without testing, there is nothing complete, because it is a vital success of the system testing goals, there are several rules that can serve as testing goals. They are:

- Testing is the process of implementing the program with the intention of finding an error.
- A good test case is a case that has a high opportunity to find an undiscovered error.
- A successful test is a test that reveals an undiscovered error If testing is successfully performed according to the above objectives, it would reveal the software errors.
- Testing also shows that the software function seems to work according to the specification, the performance requirement seems to have been met.

5.2 TEST PLAN

The test plan means a number of required procedures that need to be followed when performing various test methods. The test plan works as a blue print for the event to be followed. Software engineers create a computer program, its documentation and related data structures. Software developers are always responsible for testing individual units of programs and ensure that everyone performs the function for which it was designed. There is an independent test group (ITG) to eliminate natural problems associated with the builder to try what was created. Specific testing objectives should be given in a measurable way. In order for the average failure time, the cost of finding and repairing defects, the remaining density of the defect or frequency of occurrence, and working hours testing for the regression test should be listed within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Testing units focuses on verification efforts on the smallest software design unit - software component or module. Using the component level design as a wizard, important control paths are tested to detect errors in the module limit. Before starting any other test, data flow tests across the module interface are required. If the data do not enter and leave correctly, all other tests are deepened. Selective trial testing is a necessary task during the unit test. Good design dictates that error conditions and set paths of handling errors set to redirect or clean processing are expected when an error occurs. Border testing is the last task of the step testing step. Software often fails on its borders.

5.2.2 Integration Testing

Integration testing is a systematic technique for the construction of the program structure and at the same time perform tests to detect errors associated with the connection. The aim is to take the components tested to the unit and create a program structure that was dictated by the proposal. The whole program is tested as a whole. Correction is difficult because the insulation of causes is complicated by the huge area of the entire program. Once these errors are fixed, a new one appears and the process continues to see the seemingly endless loop. After testing the units in the system, all modules were integrated for testing any inconsistency in interfaces. In addition, the differences in program structures were removed and the unique structure of the program has developed.

5.2.3 Validation Testing or System Testing

This is the last step in testing. In this, the whole system was tested with all forms, code, modules and class modules. This form of testing is known as a black box test or system tests. The black box test method focuses on functional software requirements. This means that the Black Box testing allows you to deduce a set of input conditions that fully perform all functional requirements for the program.

5.2.4 Output Testing or User Acceptance Testing

The system under consideration is tested to receive the user; Here it should satisfy the need for the company. Software should be in contact with a perspective system; The user at the time of development and making changes whenever necessary. This happened with respect to the following points:

- Input screens Designs
- Output screen designs

The above testing is performed with the reception of different types of test data. The preparation of test data plays an essential role in system testing. After testing the test data, the monitored system is tested using these test data.

5.2.5 Automation Testing

The Suite test case is performed using specialized automated test software tools as part of the software testing technique known as automation testing. The test phases are carefully performed by human performing manual testing while sitting in front of the computer. In addition, automation testing software can generate thorough test messages, compare the expected and actual findings, and enter test data into the test system. The automation of the software test requires significant financial and material inputs.

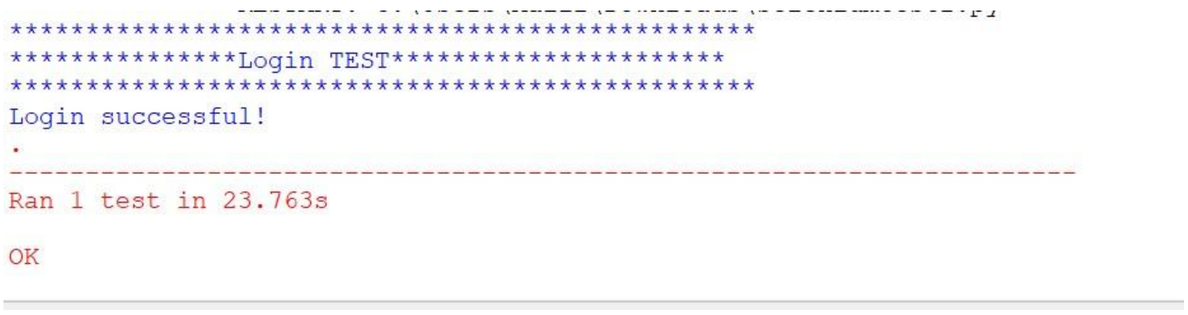
5.2.6 Selenium Testing

Selenium is a free and open source tool for testing web applications in multiple browsers and operating systems. Selenia test scripts can be written in various programming languages, including Java, C#, JavaScript, Python, etc. Automation performed by selenium frame is referred to as selenium automation testing.

Example:

Test Case 1

Eg.Screenshot

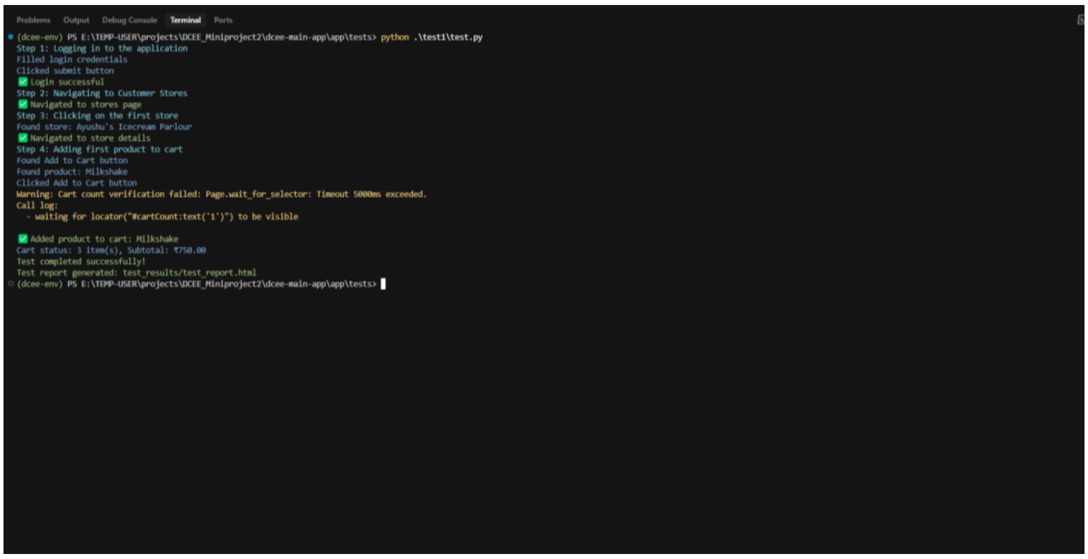


```
**************************************************
***************Login TEST**********************
**************************************************
Login successful!
.
----------------------------------------------------------------------
Ran 1 test in 23.763s

OK
```

Test Case 2:

Screenshot

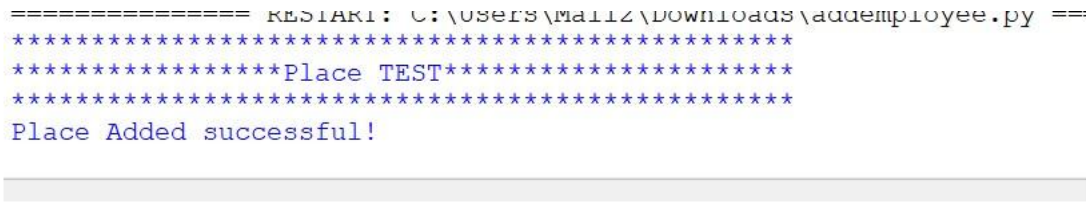


```
● (dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests> python .\test1\test.py
  Step 1: Logging in to the application
  Filled login credentials
  Clicked submit button
  ✅ Login successful
  Step 2: Navigating to Customer Stores
  ✅ Navigated to stores page
  Step 3: Clicking on the first store
  Found store: Ayushu's Icecream Parlour
  ✅ Navigated to store details
  Step 4: Adding first product to cart
  Found Add to Cart button
  Found product: Milkshake
  Clicked Add to Cart button
  Warning: Cart count verification failed: Page.wait_for_selector: Timeout 5000ms exceeded.
  Call log:
    - waiting for locator("#cartCount:text('1')") to be visible

  ✅ Added product to cart: Milkshake
  Cart status: 3 item(s), Subtotal: ₹750.00
  Test completed successfully!
  Test report generated: test_results/test_report.html
○ (dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests> █
```

Test Case 3:

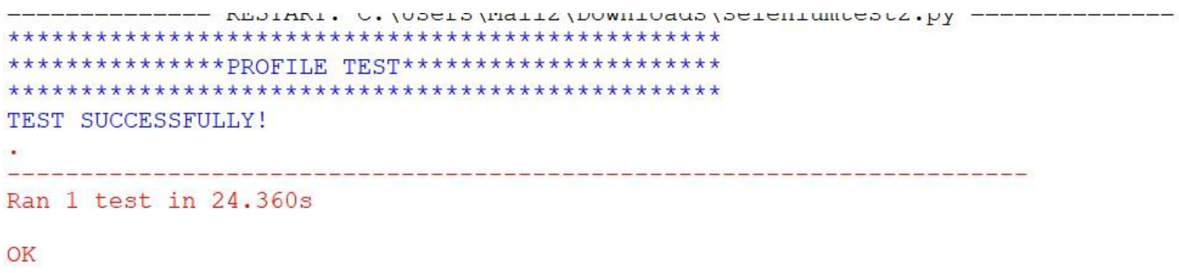
Screenshot



```
=============== RESTART: C:\Users\Mali2\Downloads\addemployee.py ==
**************************************************
*****************Place TEST**********************
**************************************************
Place Added successful!
```

Test Case 4:

Screenshot

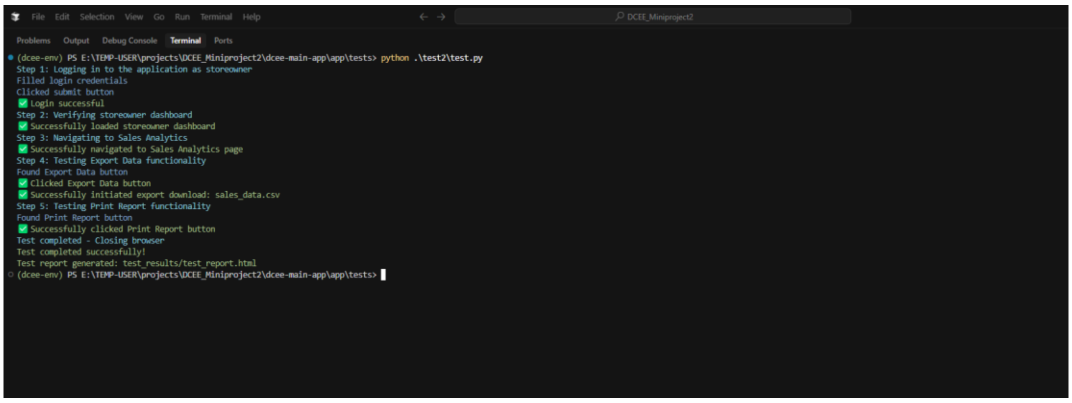


```
-------------- RESTART: C:\Users\Mali2\Downloads\seleniumtest2.py --------------
************************************************
***************PROFILE TEST**********************
************************************************
TEST SUCCESSFULLY!
.
----------------------------------------------------------------------
Ran 1 test in 24.360s

OK
```

Test Case 5:

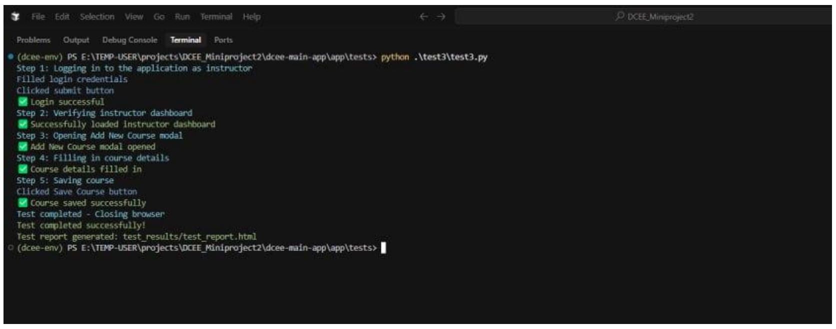
Screenshot


File Edit Selection View Go Run Terminal Help
DCEE_Miniproject2
Problems Output Debug Console Terminal Ports

```
(dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests> python .\test2\test.py
Step 1: Logging in to the application as storeowner
Filled login credentials
Clicked submit button
✅ Login successful
Step 2: Verifying storeowner dashboard
✅ Successfully loaded storeowner dashboard
Step 3: Navigating to Sales Analytics
✅ Successfully navigated to Sales Analytics page
Step 4: Testing Export Data functionality
Found Export Data button
✅ Clicked Export Data button
✅ Successfully initiated export download: sales_data.csv
Step 5: Testing Print Report functionality
Found Print Report button
✅ Successfully clicked Print Report button
Test completed - Closing browser
Test completed successfully!
Test report generated: test_results/test_report.html
(dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests>
```

Test Case 6:

Screenshot


File Edit Selection View Go Run Terminal Help
DCEE_Miniproject2
Problems Output Debug Console Terminal Ports

```
(dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests> python .\test3\test3.py
Step 1: Logging in to the application as instructor
Filled login credentials
Clicked submit button
✅ Login successful
Step 2: Verifying instructor dashboard
✅ Successfully loaded instructor dashboard
Step 3: Opening Add New Course modal
✅ Add New Course modal opened
Step 4: Filling in course details
✅ Course details filled in
Step 5: Saving course
Clicked Save Course button
✅ Course saved successfully
Test completed - Closing browser
Test completed successfully!
Test report generated: test_results/test_report.html
(dcee-env) PS E:\TEMP-USER\projects\DCEE_Miniproject2\dcee-main-app\app\tests>
```

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

Implementation is a project phase where theoretical design turns into a working system. It can be the most important phase in achieving a successful new system that gains users' confidence that the new system will work and will be efficient and accurate. It mainly deals with the training and documentation of users. Conversion usually takes place at about the same time when the user is trained or later. Implementation simply means convening a new system design into operation, a process of converting a new revised system design to an operating system. At this stage, the main working load, the greatest shocks and the main impact on the existing system move to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

The system can only be implemented after testing and if it is found to work according to specifications. System employees check system feasibility. The implementation includes all those activities that are taken from the existing system to a new system. The new system can be a brand new, replacing an existing manual or automated system or can be edited by an existing system. The more complicated the implemented system is, the more the system analysis and design efforts needed to implement the three main aspects: education and training, testing and transition in the system. The state of implementation includes the following tasks:

- Careful planning
- System investigation and restriction
- Design of methods to achieve transition.

6.2 IMPLEMENTATION PROCEDURES

The implementation of the software refers to the final installation of the package in its real environment, to the satisfaction of the intended use and operation of the system. In many organizations someone who will not operate it will introduce a software development project. In the initial phase, people doubt the software but we have to ensure that the resistance does not create because it is necessary to make sure that:

- The active user must be aware of the benefits of using the new system. Their confidence in the software is created.
- The correct instructions are provided to the user to be convenient to use the application.

Before we continue and display the system, the user must know that the server program should be launched on the server to view the result.

6.2.1 User Training

User training is designed to prepare a user for testing and converting a system. To achieve the goal and the benefits of the computer system, it is essential that people who are involved will convince their role in the new system. As the system becomes more complex, training is more important. User, the user learns how to enter data, respond to error messages, interrogate a database and call a routine that will create messages and perform other necessary functions.

6.2.2 Training on the Application Software

After providing the necessary basic training of computer awareness, the user will have to be trained on new application software. This provides a basic philosophy of using a new system, such as the screen flow, the type of help on the screen, the type of data entering errors, corresponding to the verification check with each item and the method of repairing the specified date.

6.2.3 System Maintenance

The maintenance of the system is an effective part, so the maintenance of the system applies to the ongoing activities needed to ensure that the system or application works efficiently and efficiently after implementation. It includes regular updates, bug fixes and performance optimization to keep the system smooth and safely in operation. The maintenance of the system is necessary to ensure that the system remains functional and effective after implementation. By introducing maintenance procedures and their consequence, project teams can ensure that the system works smoothly, remains safe and continues to suit the needs of end users.

6.2.4 Hosting

Hosting allows the site to be accessible online by saving its files on the server. Different types of hosting ensure different needs: shared hosting is a budget friendly; VPS offers more control, reserved hosting provides full server access and cloud hosting increases scalability. Managed hosting processes technical tasks for you, while self -management allows complete inspection for technically proficient users. Good hosting ensures reliability, speed and security of websites, impact on user experience and SEO.

Eg. Render Hosting

Hosting the rendering often includes features such as server rendering (SSR), generating a static website (SSG) and content supply network (CDN) to optimize the load time and improve scalability. The server sends a fully designed HTML server by pre -rendering or using SSR, increasing SEO speed and performance.

Procedure for hosting a website on Render Hosting:

Step 1: Set up a Render Account

- Visit the render website and register for a new account or log in if you already have it.

Step 2: Link GitHub Repository

- After logging in to plot, go to the dashboard and click on the new button to create a new service.
- Select a web service as the type of service.
- If there is a prompt, connect your Github account to render if you haven't done it yet so.
- After connecting the GitHub account, select DCEE storage from the list Available repositories.

Step 3: Configure Deployment Settings

- Select the appropriate branch for deployment (usually the main or main).
- Select the correct run for your project (for example, Python 3.x for flask applications).
- Set the assembly command to install addiction: PIP install -R requests.txt.
- Set the start command to start the app for the flask: Gunicorn DCEE.wsgi.

Step 4: Set up the Database (MongoDB)

- Since the project uses as a Mongodb database, no comprehensive database configuration is required to render. However, make sure the database file is part of the repository And it is correctly configured in your project.
- If you plan to migrate your database, make sure the correct Mongodb commands are running During deployment to use migration (eg Python Manage.Py migrate).

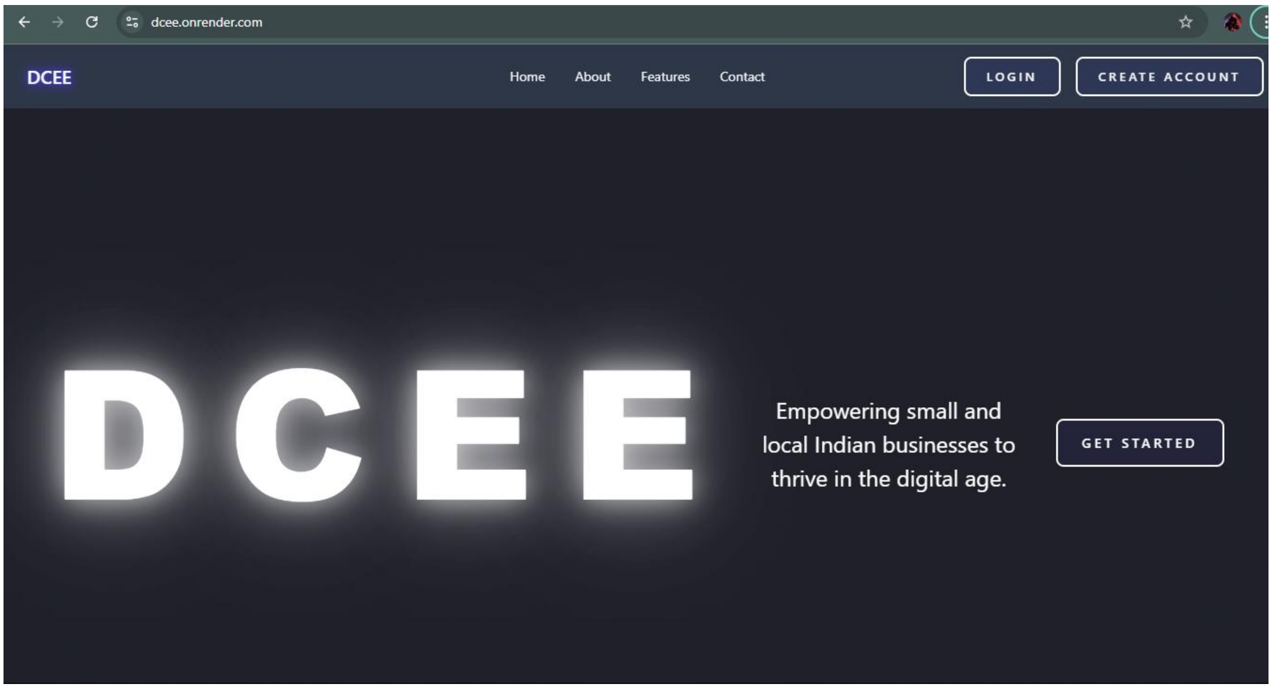
Step 5: Deploy the Project

- After setting everything, click Create a web service to start the deployment process.
- Drawing automatically creates a project, installs addiction and starts the web service.
- After the deployment is completed, your project will be alive on the URL provided.

Hosted Link: https://dcee.onrender.com/

Hosted Link QR Code

Screenshot


dcee.onrender.com
DCEE
Home
About
Features
Contact
LOGIN
CREATE ACCOUNT
DCEE
Empowering small and local Indian businesses to thrive in the digital age.
GET STARTED

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The DCEE project is a comprehensive digital strengthening platform to overlap critical gaps facing small and local Indian companies. Through its intuitive interface, DCEE streamlines basic tasks, from inventory management and secure payment transactions to customer involvement and support for compliance. The project integrates a predictive analytical tool powered by the Gemini AI thinking model and provides information that helps businesses to predict demand, optimize stocks and navigate market trends. DCEE's commitment to digital inclusivity and user -friendly tools is equipped with entrepreneurs to strengthen customers' confidence, improve operational efficiency and growth support. DCEE is supporting digital literacy and facilitating solutions in the countryside with a country tool to maintain local business, seize business owners and educate the interconnected economic ecosystem.

7.2 FUTURE SCOPE

The future range of DCEE project is promising, with the opportunity to integrate advanced machine learning models for an even more accurate demand forecast and knowledge of customer behavior. Other features could include multilingual support, diet of diverse user base throughout India, and more sophisticated digital literacy modules that will help users to become proficient in platform instruments. As digital payments continue to grow, DCEE could expand its financial offers with possibilities such as microfinance and credit solutions adapted to small businesses. In addition, the platform could include blockchain technology for increased transaction transparency and security, build confidence with customers and streamline compliance processes. The future development of DCEE will continue to seize small businesses to adapt, compete and sustainable growth in the increasingly digital market