Digital Commerce Empowerment Ecosystem in rural areas (DCEE)

Mini Project Report

Submitted by

Josna Mary Thomas

Reg. No.: AJC20MCA-I041

In Partial fulfillment for the Award of the Degree Of

MASTER OF COMPUTER APPLICATIONS

(INTEGRATED MCA)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS

KANJIRAPPALLY

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

2024-2025

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, "**Digital Commerce Empowerment Ecosystem in rural areas (DCEE)"** is the bona fide work of **JOSNA MARY THOMAS (Regno: AJC20MCA-I041)** 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 2024-25.

Ms. Sona Maria Sebastian	**Mr. Binumon Joseph**
Internal Guide	**Coordinator**

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report **“Digital Commerce Empowerment Ecosystem in rural areas (DCEE)”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the **Integrated Master of Computer Applications (INMCA)** from **APJ Abdul Kalam Technological University**, during the academic year **2024-2025**.

Date:	**JOSNA MARY THOMAS**
KANJIRAPPALLY	**Reg: AJC20MCA-I041**

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 Director (Administration) **Rev. Fr. Dr. Roy Abraham Pazhayaparampil** 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 **Mr. Binumon Joseph (Assistant Professor)** for his 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 **Ms. Sona Maria Sebastian (Assistant Professor)** 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.

JOSNA MARY THOMAS

ABSTRACT

The Digital Commerce Empowerment Ecosystem (DCEE) project is a transformative initiative aimed at digitally uplifting small and local Indian businesses, empowering them to thrive in an increasingly competitive online market. The platform offers a comprehensive suite of tools and features designed to enhance digital presence, streamline operations, and foster customer relationships. By providing a customizable digital storefront, secure payment gateways, and an intuitive user interface, DCEE enables businesses to cultivate trust and effectively engage with their customers.

At the core of DCEE's functionality is its advanced analytics module, powered by the RoBERTa model, which facilitates predictive analytics for inventory management. This feature allows businesses to accurately forecast market demands, optimize stock levels, and reduce waste, ultimately leading to improved revenue and profitability. The platform is structured to cater to various user roles—storefront owners, customers, and administrators—each with tailored functionalities such as profile management, transaction oversight, and real-time data insights.

Moreover, DCEE incorporates a robust chatbot feature that provides users with immediate assistance, guiding them to appropriate resources and helping to resolve common issues. This promotes a seamless user experience, encouraging greater engagement and retention. DCEE also addresses critical challenges such as regulatory compliance, payment facilitation, and digital literacy gaps, ensuring that small businesses are equipped to navigate the complexities of the digital landscape.

In addition to these features, DCEE fosters community resilience by enabling small businesses to leverage technology for growth. By prioritizing accessibility and user-friendliness, the platform aims to create a supportive ecosystem where local entrepreneurs can compete effectively and contribute to economic development in their communities. Ultimately, DCEE not only serves as a catalyst for individual business success but also enriches the entrepreneurial spirit that is vital to India's economic landscape. Through this holistic approach, DCEE is poised to redefine how small businesses operate and succeed in the digital age, making a significant impact on the broader economy.

CONTENT

List of Abbreviation

- DCEE - Digital Commerce Empowerment Ecosystem
- CRUD - Create, Read, Update, Delete
- ARIMA - Autoregressive Integrated Moving Average
- LSTM - Long Short-Term Memory
- RoBERTa - Robustly optimized BERT approach
- NLP - Natural Language Processing
- ORM - Object-Relational Mapping
- SaaS - Software as a Service
- HTTPS - HyperText Transfer Protocol Secure
- IDE - Integrated Development Environment
- MVC - Model View Controller
- REST - Representational State Transfer
- JWT - JSON Web Token

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The Digital Commerce Empowerment Ecosystem (DCEE) is an innovative platform designed to empower small and local Indian businesses by offering essential tools to navigate the digital economy. Focusing on bridging the digital divide, DCEE provides solutions for establishing a digital presence, fostering customer trust, managing rural payments, optimizing inventory, and supporting digital literacy. Key features include a predictive analytics module powered by the RoBERTa model for inventory insights, a chatbot for user guidance, and a payment module for streamlined transactions. Built with a scalable tech stack, DCEE equips business owners, customers, and admins with robust tools to manage profiles, storefronts, and transactions, creating a dynamic and accessible environment for local business growth.

1.2 PROJECT SPECIFICATION

The Digital Commerce Empowerment Ecosystem (DCEE) supports small Indian businesses by providing digital tools to enhance visibility, optimize operations, and build customer trust. The platform features three 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.

DCEE aims to drive digital transformation, support small businesses, and promote economic growth in India’s local markets.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The Digital Commerce Empowerment Ecosystem (DCEE) is designed to revolutionize small and local Indian businesses by addressing the specific challenges they face in the digital landscape. Many of these enterprises struggle with limited access to technology and digital tools, which inhibits their ability to engage customers, streamline operations, and enhance their market presence. DCEE offers a holistic solution that empowers storefront owners with essential tools for online management, including customer profiles, product listings, and secure payment processing. This integration not only simplifies their operations but also enhances their ability to reach a wider audience.

DCEE features an analytics module that harnesses predictive insights, helping businesses manage inventory effectively and avoid the risks associated with overstocking and stockouts. The use of advanced machine learning tools, such as the RoBERTa model, enables accurate demand forecasting, enhancing decision-making capabilities. Additionally, the platform incorporates a user-friendly chatbot that assists users with inquiries and navigational support, significantly reducing the digital literacy barrier that many small businesses owners face.

By facilitating rural payment options and ensuring compliance with regulatory requirements, DCEE fosters customer trust and accessibility. This ecosystem not only streamlines operations but also nurtures entrepreneurial spirit by providing training and resources that promote digital literacy. Ultimately, DCEE aspires to bridge the digital divide, empowering small businesses to thrive, contribute to local economies, and sustain their growth in a rapidly evolving digital marketplace.

2.2 LITERATURE REVIEW

The Digital Commerce Empowerment Ecosystem (DCEE) project addresses key challenges faced by small and medium-sized enterprises (SMEs), especially in emerging economies like India, by incorporating predictive analytics and digital transformation solutions to bridge gaps in digital literacy, rural payments, and inventory management. This review summarizes the relevant literature to frame the DCEE's objectives and proposed solutions.

Inventory Forecasting with ARIMA and LSTM Models

Research on predictive analytics for inventory management highlights the effectiveness of time-series models like ARIMA and deep learning models like LSTM in predicting demand and reducing inventory costs. Choi and Lee (2019) review recent advances in ARIMA and LSTM models, noting their complementary strengths—ARIMA for linear trends and LSTM for complex, long-term dependencies—making them ideal for dynamic SME environments [1]. Similarly, Wang and Zhao (2020) emphasize that machine learning models, including ARIMA and LSTM, allow businesses to optimize stock levels and minimize costs [5].

Digital Transformation and SME Empowerment

Digital technologies have proven to be pivotal in boosting productivity and market access for SMEs. The World Bank (2020) reports that digital transformation improves SMEs' competitiveness by enhancing their operational efficiency and access to digital markets [2]. In line with this, the OECD (2021) underscores the role of digital tools in transforming business processes, helping SMEs remain competitive in a digital era [6].

Predictive Analytics and Inventory Optimization

Fildes and Goodwin (2020) highlight predictive analytics as a cost-saving mechanism for inventory management, enabling businesses to make more informed stock decisions. Their findings stress the importance of using accurate forecasts to avoid over- or under-stocking, thereby reducing waste and improving financial performance [3]. Moreover, AI-driven models for inventory management, as reviewed by Wang and Thompson (2022), enhance stock management by dynamically adjusting to demand patterns [22].

Addressing the Digital Divide in Emerging Economies

The digital divide remains a significant barrier for SMEs in underserved regions. The ITU (2021) and World Economic Forum (2020) stress digital inclusion strategies to support economic participation for rural businesses [4, 17]. Through digital literacy initiatives, UNESCO's (2020) report finds that digital skills improve SMEs' market reach, empowering them to leverage digital tools effectively [15].

Enhanced Customer Engagement and Digital Literacy

NLP models such as RoBERTa are being increasingly applied to improve customer engagement and support through automated systems. RoBERTa's capabilities for sentiment analysis and query handling enhance customer interactions, which is crucial for small businesses that may lack dedicated customer service teams [16]. Johnson and Davis (2021) discuss how digital literacy also plays a critical role in helping SMEs harness technology to improve competitiveness and sustain economic growth [18].

In summary, the literature supports the DCEE's integrative approach of leveraging predictive analytics for inventory management and enhancing digital literacy for SMEs. By combining

ARIMA and LSTM models for accurate inventory forecasting and utilizing RoBERTa for customer engagement, the DCEE project aligns with global findings that emphasize digital transformation as a pathway to growth for small businesses in emerging economies.

2.3 PROPOSED SYSTEM

The proposed system for small and local Indian businesses aims to transform their digital presence and operational efficiency. It will provide a user-friendly platform that allows businesses to establish and manage online storefronts, significantly enhancing visibility and customer reach. Integrated predictive analytics tools will enable accurate forecasting of inventory needs, minimizing the risks associated with overstocking and understocking. Additionally, the system will streamline payment solutions, offering secure and simplified transaction methods for both customers and business owners. To ensure compliance with local regulations, built-in features will guide users through the necessary requirements. Furthermore, the system will include resources and training programs focused on improving digital literacy, empowering users to leverage the platform effectively and enhance their overall business operations.

2.5 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 Digital Commerce Empowerment Ecosystem (DCEE) project by examining its technical, economic, operational, legal, and market aspects. It assesses whether the chosen technology stack (Flask for backend, Bootstrap for frontend, and MongoDB for database) is suitable and compatible with project requirements. The economic analysis focuses on development costs, potential revenue, and return on investment. Operational feasibility investigates how the system will integrate into current business practices and address user needs. Legal considerations ensure compliance with data protection regulations and payment processing laws. Lastly, market feasibility assesses the demand for the system among small businesses, ensuring it aligns with market trends and customer expectations. This comprehensive analysis aims to determine the project's overall viability and sustainability.

3.1.1 Economical Feasibility

The economic feasibility study for the Digital Commerce Empowerment Ecosystem (DCEE) assesses the project's financial viability, focusing on costs, revenue potential, and return on investment. Initial costs encompass development, hosting, and salaries, while ongoing expenses include maintenance and support. Revenue sources 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 Digital Commerce Empowerment Ecosystem (DCEE) evaluates the project's technological requirements and capabilities to ensure successful implementation. It examines the software, hardware, and network infrastructure necessary to develop and operate the platform effectively.

Key components include:

- Technology Stack: Utilization of Bootstrap for frontend development, Flask API for backend services, and MongoDB for database management.
- System Integration: Ensuring seamless integration between various modules, including authentication, payment processing, analytics, and chatbot functionalities.
- Scalability: Assessing the system's ability to handle increased user traffic and data as the platform grows.
- Security Measures: Implementing robust security protocols to protect user data and transactions.
- User Support: Providing necessary training and documentation to users for effective system utilization.

The study concludes that the DCEE project is technically feasible, leveraging existing technologies and best practices to create a reliable and secure digital commerce 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 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 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
Backend	-	FLASK, Cloud Mongo

Client on PC	-	Windows 7 and above.
Technologies used	-	PYTHON, JS, HTML5, AJAX, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 Eg. FLASK

Flask is a lightweight and flexible web framework written in Python, designed for creating web applications quickly and with minimal overhead. It follows a modular approach, offering only essential tools and allowing developers to add extensions as needed for database connections, form handling, authentication, and more. Flask operates on the WSGI (Web Server Gateway Interface) standard and uses Jinja2 for templating, providing robust tools for rendering dynamic HTML with minimal effort.

Being “micro,” Flask is unopinionated, meaning it does not enforce any project structure or dependency requirements, giving developers control over the app’s architecture and behavior. Its simplicity and adaptability make it ideal for small to medium-sized applications, while its scalability supports more complex setups when combined with additional components. Flask is a powerful yet minimalist web framework written in Python, ideal for developers who prefer simplicity and control in web application development. As a "micro-framework," Flask includes only the essential components for handling requests, routing, and templating, making it exceptionally lightweight and flexible. It leverages WSGI (Web Server Gateway Interface) for handling web requests and Jinja2, a fast and expressive templating engine, for creating dynamic HTML responses.

One of Flask’s standout features is its extensibility—developers can choose specific libraries or extensions for functionalities like database integration, form validation, and authentication without carrying any unnecessary dependencies. This "plug-and-play" approach allows developers to build both simple applications and scalable, production-level systems. Flask is also highly compatible with other Python libraries, making it popular for projects that include machine learning, data visualization, or complex business logic. Its straightforward syntax and clear documentation make Flask a top choice for beginners, while its flexibility and powerful extensions appeal to experienced developers who need a tailored, high-performing application stack.

3.3.2Eg. MongoDB

MongoDB is a NoSQL database known for its scalability, flexibility, and document-based structure. Unlike traditional relational databases that store data in tables with rows and columns, MongoDB stores data in BSON (Binary JSON) documents, allowing it to handle unstructured or semi-structured data efficiently. Each document is a self-contained data unit, making MongoDB well-suited for applications that require rapid data integration, real-time analytics, and frequent updates to data structure without disrupting the entire system.

MongoDB’s schema-less nature allows developers to add or modify fields easily without predefined data schemas. This flexibility makes it ideal for projects that evolve quickly or handle large volumes of diverse data types, like social media applications, e-commerce platforms, and IoT solutions. It also supports a rich query language, indexing, aggregation, and powerful features like horizontal scaling (sharding) and replication for high availability, which enhances its performance and fault tolerance. MongoDB integrates well with modern tech stacks, making it a popular choice for full-stack developers and organizations adopting cloud-native, distributed application architectures.

CHAPTER 4
SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process, or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance, and accuracy levels. The design phase is a transition from a user-oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. UML stands for Unified Modeling Language. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it complete. UML includes 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 represent specific functionalities or actions the system performs, such as "Login," "Register," "Purchase Product," or "View Orders."
- Associations are the lines connecting actors and use cases, indicating interactions or relationships.
- System Boundary is a rectangle that encapsulates all the use cases, defining the scope of the system.

Use Case Diagrams are useful for understanding user requirements, identifying main functionalities, and defining roles within a system. They also serve as a basis for further detailed analysis and design, helping ensure the system aligns with user needs.

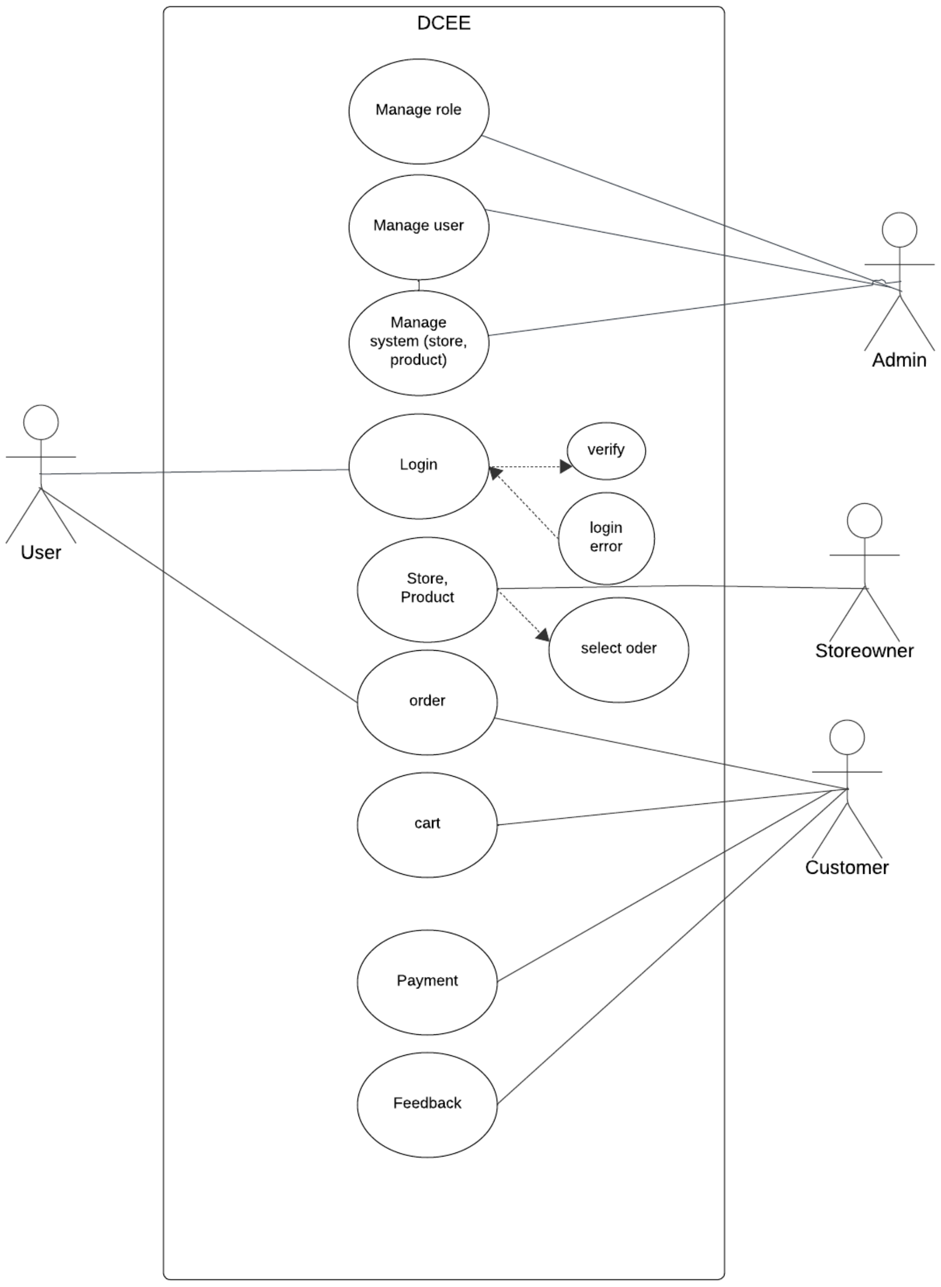


Figure 1. Use Case Diagram

4.2.1 SEQUENCE DIAGRAM

A sequence diagram is a type of interaction diagram in Unified Modeling Language (UML) that shows how objects interact in a particular sequence within a system. It visually represents the flow of messages or events between objects over time, illustrating how and in what order operations are carried out.

Key components of a sequence diagram include:

Actors: 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.
- System Design and Debugging: Developers use sequence diagrams to refine system architecture, identify bottlenecks, and troubleshoot issues by understanding exact message flow.

Sequence diagrams support collaboration by showing the workflow in a way that is easy to understand and highly detailed. They help developers, stakeholders, and testers confirm requirements and understand how different components should interact, assisting in debugging, performance optimization, and overall design clarity.

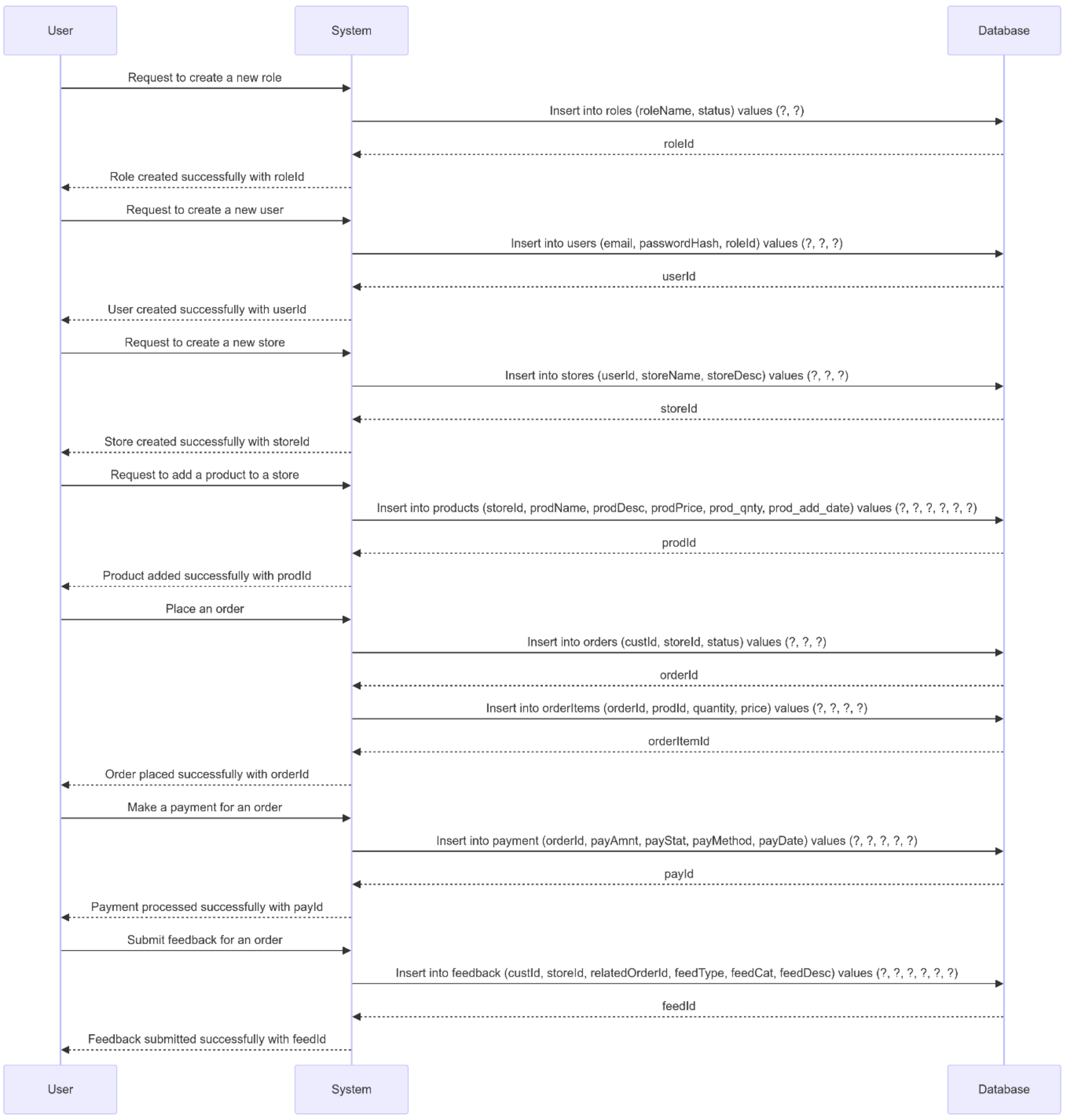

User
System
Database
Request to create a new role
Insert into roles (roleName, status) values (?, ?)
roleId
Role created successfully with roleId
Request to create a new user
Insert into users (email, passwordHash, roleId) values (?, ?, ?)
userId
User created successfully with userId
Request to create a new store
Insert into stores (userId, storeName, storeDesc) values (?, ?, ?)
storeId
Store created successfully with storeId
Request to add a product to a store
Insert into products (storeId, prodName, prodDesc, prodPrice, prod_qnty, prod_add_date) values (?, ?, ?, ?, ?, ?)
prodId
Product added successfully with prodId
Place an order
Insert into orders (custId, storeId, status) values (?, ?, ?)
orderId
Insert into orderItems (orderId, prodId, quantity, price) values (?, ?, ?, ?)
orderItemId
Order placed successfully with orderId
Make a payment for an order
Insert into payment (orderId, payAmnt, payStat, payMethod, payDate) values (?, ?, ?, ?, ?)
payId
Payment processed successfully with payId
Submit feedback for an order
Insert into feedback (custId, storeId, relatedOrderId, feedType, feedCat, feedDesc) values (?, ?, ?, ?, ?, ?)
feedId
Feedback submitted successfully with feedId

Figure 2. Sequence Diagram

4.2.2 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.

Entry and Exit Actions: Actions taken when entering or leaving a state.

Composite States: Hierarchical states that contain sub-states for complex behaviors.

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.

4.2.2 Activity Diagram

An activity diagram is a behavioral diagram in Unified Modeling Language (UML) that illustrates the flow of activities or actions within a system, highlighting the dynamic aspects of processes. It features key elements such as activities, represented by rounded rectangles, which indicate tasks performed within the system. The diagram begins with a start node, depicted as a filled circle, and concludes with an end node, a circle encased within another circle, marking the process's completion. Transitions, represented by arrows, connect activities to demonstrate the flow from one task to another. Decision nodes, illustrated as diamonds, indicate points where the flow can diverge based on specific conditions, while forks and joins (depicted as bars) represent parallel activities. Activity diagrams are particularly valuable for modeling complex workflows, use case scenarios, and business processes, as they clarify the sequence of operations, identify potential bottlenecks, and facilitate communication among stakeholders by visualizing activities and their interdependencies.

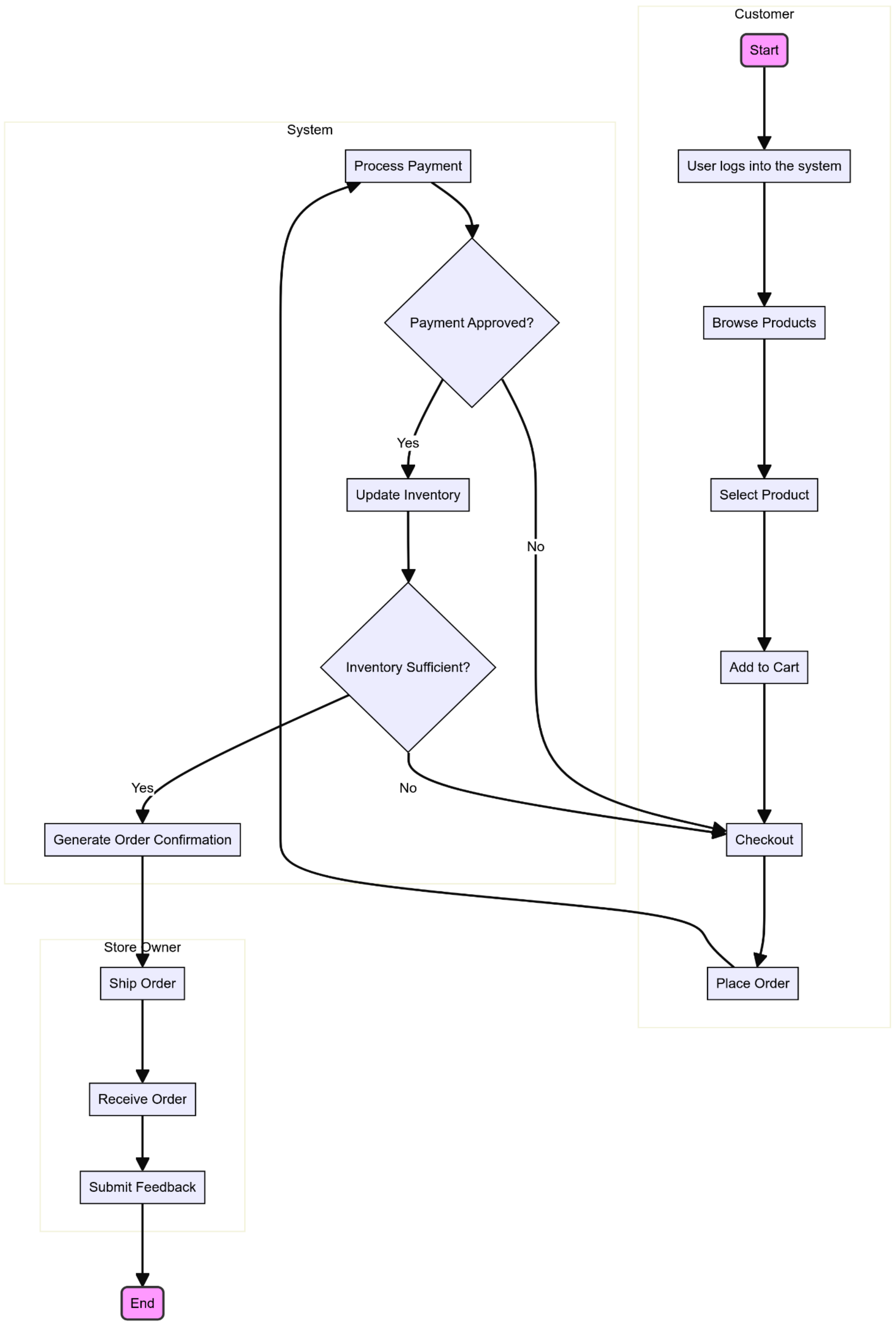


Figure 3. Activity Diagram

4.2.3 Class Diagram

A class diagram is a static structure diagram in Unified Modeling Language (UML) that represents the system's classes, their attributes, methods, and the relationships among them. It provides a visual blueprint of the system's architecture, highlighting how different classes interact and how data is organized. Each class is depicted as a rectangle divided into three sections: the top section contains the class name, the middle section lists its attributes, and the bottom section shows its methods (or operations). Relationships between classes are illustrated through lines, with various notations to indicate the type of relationship, such as inheritance (generalization), associations, aggregations, and compositions. Class diagrams are essential for object-oriented design, as they help in understanding the system's structure, guiding the implementation of classes, and ensuring that all components interact correctly. They also serve as a communication tool among developers, designers, and stakeholders by providing a clear representation of the system's data model and its relationships.

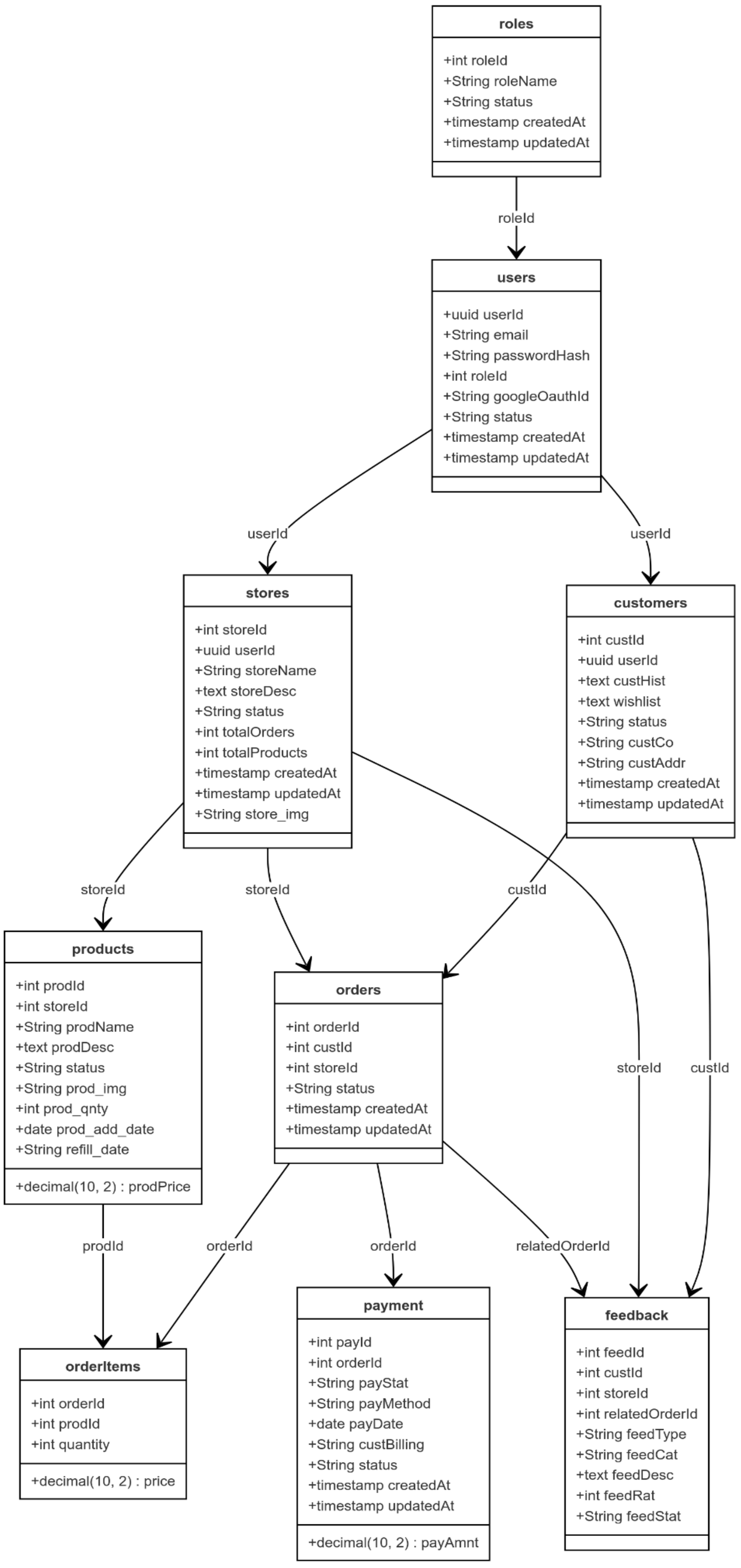


Figure 4. Class Diagram

4.2.4 Object Diagram

An object diagram is a type of static structure diagram in Unified Modeling Language (UML) that provides a snapshot of the instances of classes (objects) and their relationships at a specific moment in time. Unlike class diagrams, which focus on the blueprint of classes and their relationships, object diagrams emphasize the concrete examples of those classes in a particular state.

Each object is represented by a rectangle, similar to a class in a class diagram, but it includes the object's name and its current state or attribute values. Lines connecting the objects illustrate their relationships, such as associations, aggregations, or compositions. Object diagrams are particularly useful for visualizing complex relationships and interactions among objects in a system, demonstrating how they collaborate to perform tasks or represent data.

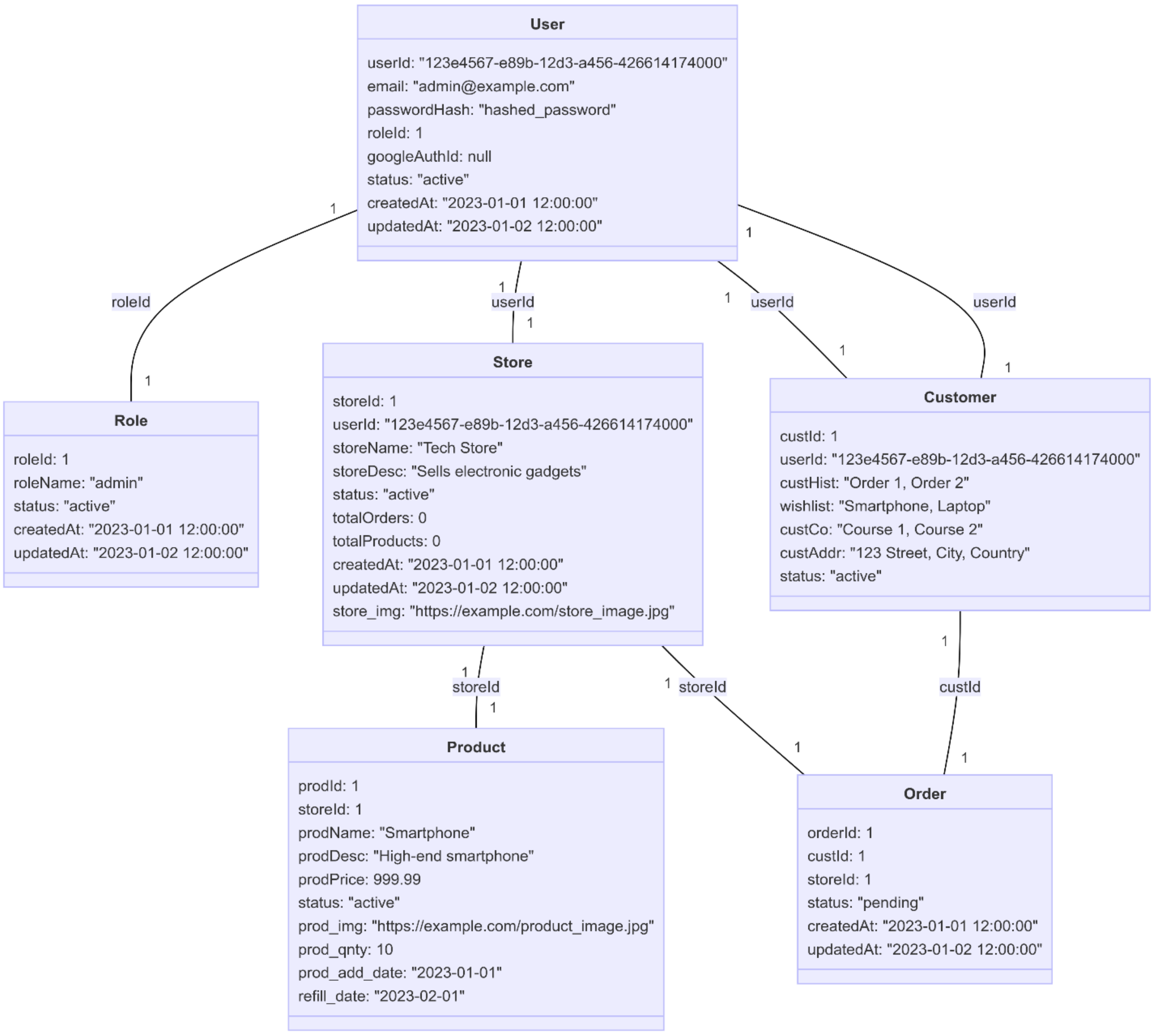

User
userId: "123e4567-e89b-12d3-a456-426614174000"
email: "admin@example.com"
passwordHash: "hashed_password"
roleId: 1
googleAuthId: null
status: "active"
createdAt: "2023-01-01 12:00:00"
updatedAt: "2023-01-02 12:00:00"
roleId
userId
userId
userId
1
Role
roleId: 1
roleName: "admin"
status: "active"
createdAt: "2023-01-01 12:00:00"
updatedAt: "2023-01-02 12:00:00"
Store
storeId: 1
userId: "123e4567-e89b-12d3-a456-426614174000"
storeName: "Tech Store"
storeDesc: "Sells electronic gadgets"
status: "active"
totalOrders: 0
totalProducts: 0
createdAt: "2023-01-01 12:00:00"
updatedAt: "2023-01-02 12:00:00"
store_img: "https://example.com/store_image.jpg"
Customer
custId: 1
userId: "123e4567-e89b-12d3-a456-426614174000"
custHist: "Order 1, Order 2"
wishlist: "Smartphone, Laptop"
custCo: "Course 1, Course 2"
custAddr: "123 Street, City, Country"
status: "active"
storeId
storeId
custId
Product
prodId: 1
storeId: 1
prodName: "Smartphone"
prodDesc: "High-end smartphone"
prodPrice: 999.99
status: "active"
prod_img: "https://example.com/product_image.jpg"
prod_qnty: 10
prod_add_date: "2023-01-01"
refill_date: "2023-02-01"
Order
orderId: 1
custId: 1
storeId: 1
status: "pending"
createdAt: "2023-01-01 12:00:00"
updatedAt: "2023-01-02 12:00:00"

Figure 5. Object Diagram

4.2.5 Component Diagram

A component diagram is a type of structural diagram in Unified Modeling Language (UML) that represents the physical components in a system, such as software applications, libraries, and packages. It illustrates how these components interact with one another and their dependencies, providing a high-level view of the system's architecture.

Components are depicted 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 the operations available for other components to use. Component diagrams are particularly useful for modeling complex systems, as they help to visualize the relationships between various software components, their functionalities, and how they fit into the overall system architecture. This aids in understanding the system’s modular structure, facilitates better planning for system integration, and enhances the communication between developers.

4.2.8 Deployment Diagram

A deployment diagram is another type of UML diagram that shows the physical deployment of artifacts on nodes. It illustrates how software is distributed across hardware and how different components communicate within the system infrastructure. Deployment diagrams are particularly useful in visualizing the physical arrangement of software components in a distributed system, such as client-server architectures or cloud-based applications.

In a deployment diagram, nodes represent physical devices (like servers, computers, or mobile devices), and artifacts (such as executables, libraries, or database schemas) represent the software deployed on those nodes. Connections between nodes depict communication paths, showing how information flows within the system. This diagram is essential for understanding the system’s architecture and can help in assessing performance, scalability, and reliability.

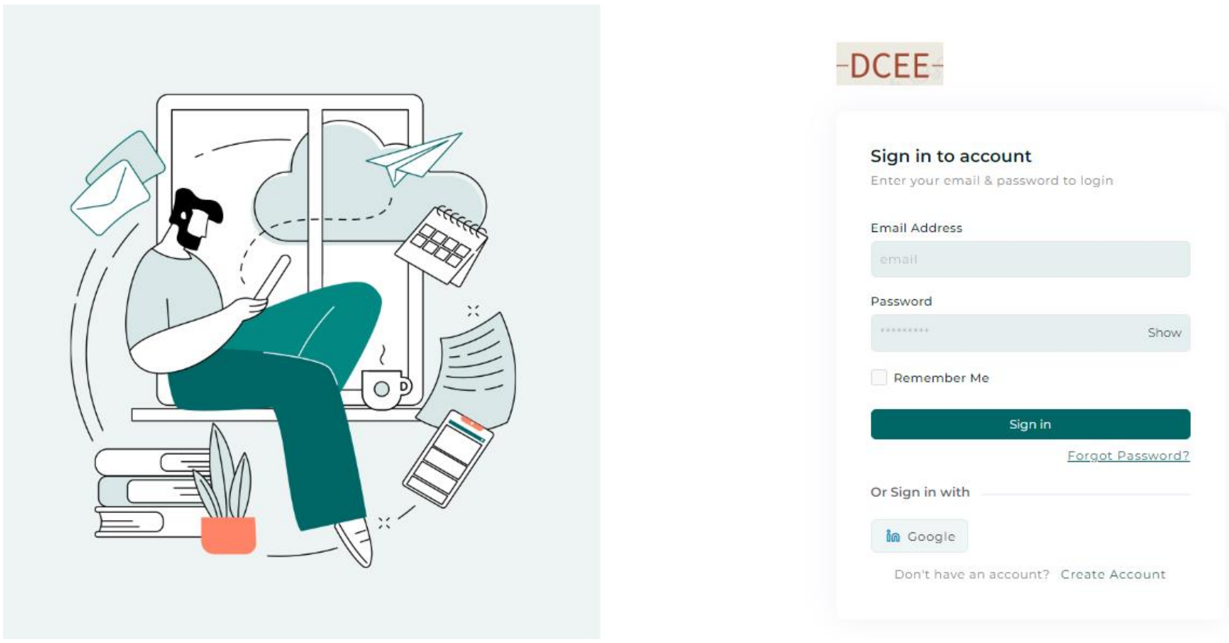
4.2.9 Collaboration Diagram

A collaboration diagram, or communication diagram, is a type of UML diagram that illustrates the interactions between objects in a system. It emphasizes the relationships and associations among objects while depicting the messages exchanged between them. Each object is represented as a rectangle, and lines connect these objects to indicate their relationships.

Messages are numbered to show the sequence of interactions, providing a clear view of how different components work together to achieve a specific task. Collaboration diagrams help visualize dynamic behavior, clarify roles, and enhance communication among team members during development, making them a valuable tool in system design.

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?

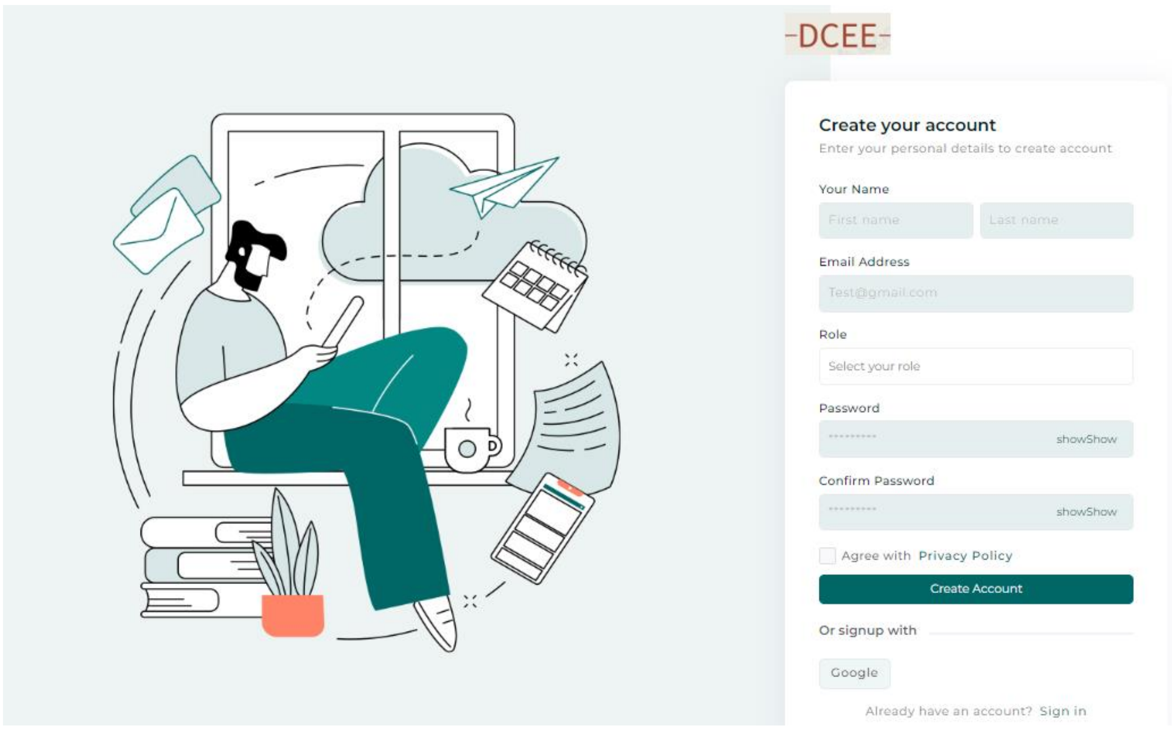
Or Sign in with

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

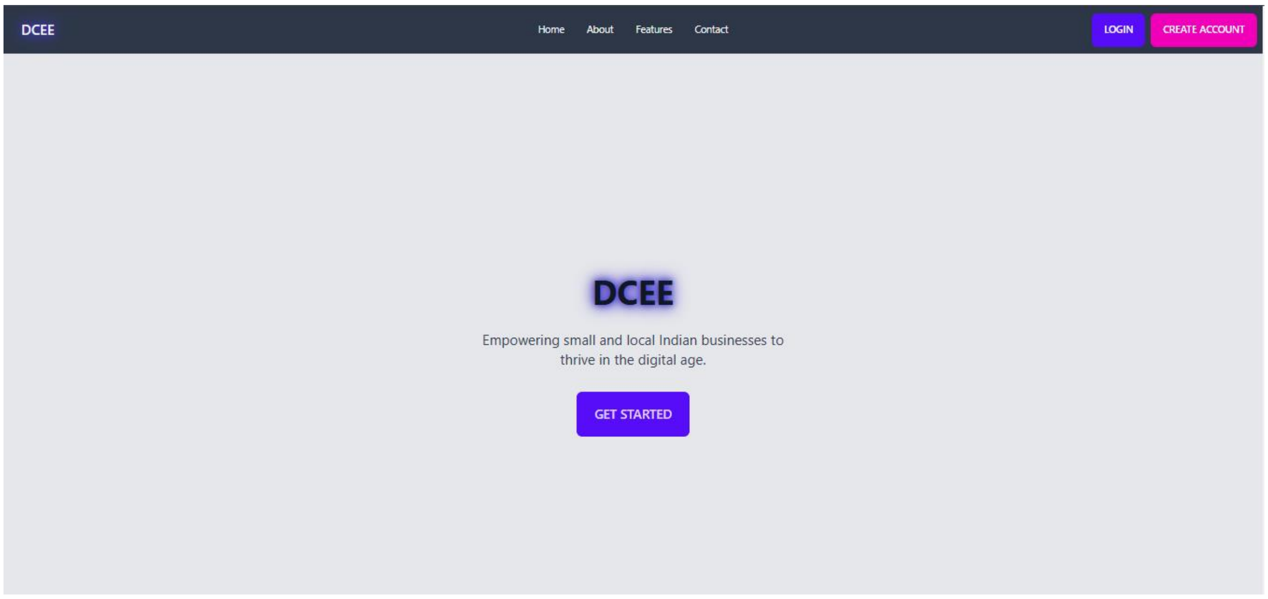
Or signup with

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

4.4 DATABASE DESIGN

4.4.1 Non - Relational Database Management System (RDBMS)

In a non-relational database (NoSQL), data is organized in collections and documents rather than tables and rows, allowing for flexible schemas where each document in a collection can have a unique structure. 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.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys

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. All the tables have been normalized up to the third normal form. As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

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 4NF, we need to ensure that all multi-valued dependencies are removed from the table, and that each table contains only attributes that are functionally dependent on the primary key.

Fifth Normal Form

5NF is indeed the highest level of normalization in relational database design, and it deals with complex data models that involve multiple overlapping multi-valued dependencies. In 5NF, tables are decomposed into smaller tables in order to eliminate any possible redundancy caused by overlapping dependencies, while ensuring that there is no loss of data.

The goal of 5NF is to ensure that each table represents a single entity or relationship, and that the data is organized in a way that minimizes redundancy, eliminates anomalies, and improves data integrity.

4.4.3 Sanitization

An automated procedure called "sanitization" is used to get a value ready for use in a SQL query. This process typically involves checking the value for characters that have a special significance 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)

CHAPTER 5
SYSTEM TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the term's verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers-based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met. There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation, and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

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

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover, differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested with all forms, code, modules, and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs.
- Output Screen Designs

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data.

5.2.5 Automation Testing

A test case suite is executed using specialized automated testing software tools as part of the software testing technique known as automation testing. The test stages are meticulously carried out by a human performing manual testing while seated in front of a computer. Additionally, the automation testing software may generate thorough test reports, compare expected and actual findings, and enter test data into the System Under Test. Software test automation necessitates significant financial and material inputs.

5.2.6 Selenium Testing

Selenium is a free and open-source tool for testing web applications across multiple browsers and operating systems. Selenium Test Scripts can be written in different programming languages, including Java, C#, JavaScript, Python, etc. Automation performed using the

Selenium framework is referred to as Selenium Automation testing.

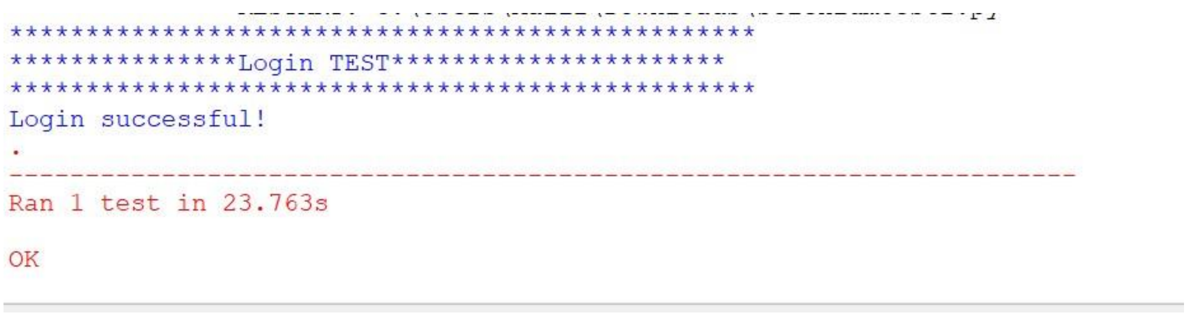
Example:

Test Case 1

Code

```
from selenium import webdriver
import time
from selenium.webdriver.common.by import By
from selenium.webdriver.edge.service import Service
from webdriver_manager.microsoft import EdgeChromiumDriverManager # Clear cached
EdgeDriver
import os
import shutil
cache_dir = os.path.expanduser("~/.cache/selenium")
if os.path.exists(cache_dir):
# Set up the driver using WebDriverManager and Service object for Edge service = Service
(EdgeChromiumDriverManager().install())
driver = webdriver. Edge (service=service)
username_input = driver.find_element(By.ID, "email")
username_input.send_keys("josna@gmail.com")
password_input = driver.find_element(By.ID, "password")
password_input.send_keys("Josna@123")
submit_button = driver.find_element(By.ID, "submit") submit_button.click()
time.sleep(10)
driver.quit()
```

Eg. Screenshot



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

OK
```

Eg. Test Report

Test Case 1

Project Name: Digital Commerce Empowerment Ecosystem (DCEE)					
Login Test Case					
Test Case ID: Test_1			**Test Designed By: Josna Mary Thomas**		
Test Priority (Low/Medium/High):			**Test Designed Date: 20/10/2024**		
Module Name: Registration Page			**Test Executed By: Sona Maria Sebastian**		
Test Title: Registration of user account			**Test Execution Date: 22/10/2024**		
Description: Registration user account					
Pre-Condition: User have unique email id					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/ Fail)**
1	Navigate to the Registration page	http://127.0.0.1:5000/auth/register	Registration page should be displayed	Registration page was loaded	Pass
2	Enter first name	“Josna”	The first name field should be filled with "Josna"	First name entered	Pass
3	Enter last name	“Thomas”	The last name field should be filled with "Thomas"	Last name entered	Pass
4	Enter role	“Customer”	The role field is selected as customer	Role selected	Pass
5	Enter email	“josnamarythomas25@gmail.com”	The email field should be filled with “josnamarythomas25@gmail.com”	Email entered	Pass
6	Enter password	“Josna@123”	The password field should be filled with “Josna@123”	Password inserted	pass

7	Enter confirm password	"Josna@123"	The confirm password field should be filled with "Josna@123"	Confirm password inserted	Pass
8	Click on the register button	N/A	If registration is successful, user should be redirected to login page	User was redirected to login page	Pass
Post-Condition: First name, last name, email, password, confirm password can be registered					

Test Case 2:

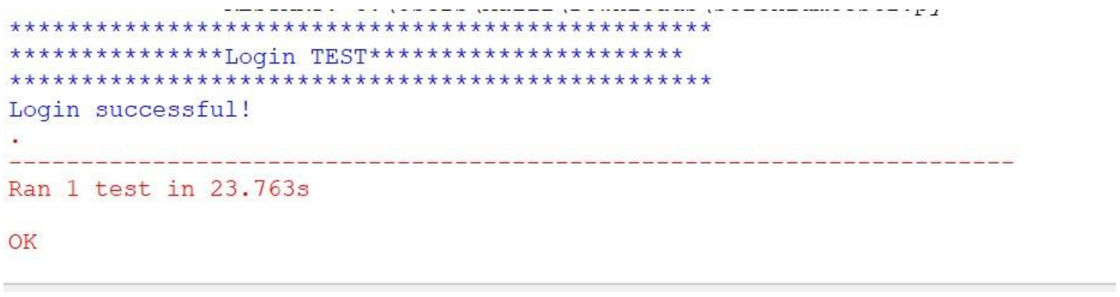
Code:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from time import sleep
import unittest
class TestService(unittest.TestCase):
   def setUp(self):
      self.driver = webdriver.Chrome()   # Ensure you have ChromeDriver set up
properly
      self.driver.get("http://127.0.0.1:8000/login1/")
      sleep(2)
def test_add_place(self):
   driver = self.driver
   username_input = driver.find_element(By.ID, "username")
   username_input.send_keys("mathew@gmail.com")
   sleep(2)
   # ... existing code ...
password_input            =            driver.find_element(By.ID,            "password")
password_input.send_keys("admin")
sleep(2)
submit_button = driver.find_element(By.ID, "submit") submit_button.click()
sleep(2)
driver.get("http://127.0.0.1:8000/admin/addservice1")
```

```
sleep(4)
service_name_input = driver.find_element(By.ID, "txt1")
service_name_input.send_keys("Plumbing Service")
sleep(2)
description_name_input = driver.find_element(By.ID, "txt2")
description_name_input.send_keys("At Expert Homecare, our expert plumbers offer
fast, reliable solutions for all plumbing issues. ")
sleep(2)
upload_input = driver.find_element(By.ID, "inputfileupload")
upload_input.send_keys("D:/mathew/miniproject-s9/uploads/1.jpg")
sleep(2)
submit_button = driver.find_element(By.ID, "register") submit_button.click()
sleep(3)
if driver.current_url == "http://127.0.0.1:8000/admin1/displayserviceadmin/":
else:
print("Place Added successful!")
print("Place Added Failed!")
def tearDown(self):

if __name__ == "__main__":  # Corrected the if statement
    self.driver.quit()  # Indented properly
    unittest.main()
```

Screenshot:



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

OK
```

Test Case 2	
Project Name: Digital Commerce Empowerment Ecosystem (DCEE)	
Login Test Case	
Test Case ID: Test_2	**Test Designed By: Josna Mary Thomas**
Test Priority (Low/Medium/High):	**Test Designed Date: 20/10/2024**

Module Name: Login Page			**Test Executed By: Sona Maria Sebastian**		
Test Title: Login to user account			**Test Execution Date: 22/10/2024**		
Description: Login to user account					
Pre-Condition: User have valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/ Fail)**
1	Navigate to the login page	http://127.0.0.1:5000/auth/login	Login page should be displayed	Login page was loaded	Pass
2	Enter email	"josna@gmail.com"	The email field should be filled with "josna@gmail.com"	email entered	Pass
3	Enter Password	"Josna@123"	The password field should be filled with "Josna@123"	Password entered	Pass
4	Click on the submit button	N/A	If login is successful, user should be redirected to user login page	User was redirected to user login page	Pass
Post-Condition: Email and password are checked with database values for successful login.					

Test Case 3:

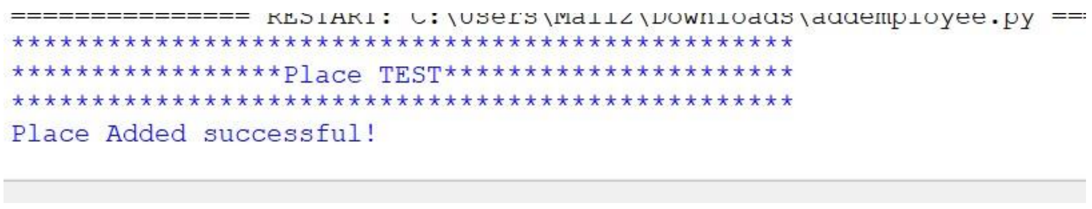
Code:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from time import sleep
import unittest
class TestService(unittest.TestCase):
def setUp(self):
self.driver = webdriver.Chrome() # Ensure you have ChromeDriver set up properly
self.driver.get("http://127.0.0.1:8000/login1/")
sleep(2)
def test_add_place(self):
```

```
driver = self.driver
username_input                =               driver.find_element(By.ID,               "username")
username_input.send_keys("josna@gmail.com")
sleep(2)
password_input                =               driver.find_element(By.ID,               "password")
password_input.send_keys("admin")
sleep(2)
submit_button = driver.find_element(By.ID, "submit") submit_button.click()
sleep(2)
driver.get("http://127.0.0.1:8000/admin/addservice1")
sleep(4)
service_name_input                =               driver.find_element(By.ID,               "txt1")
service_name_input.send_keys("Plumbing Service")
sleep(2)
description_name_input = driver.find_element(By.ID, "txt2")
description_name_input.send_keys("At DCEE. ")
sleep(2)
upload_input = driver.find_element(By.ID, "inputfileupload")
upload_input.send_keys("D:/josna/miniproject-s9/uploads/1.jpg")
sleep(2)
submit_button = driver.find_element(By.ID, "register") submit_button.click()
sleep(3)
if driver.current_url == "http://127.0.0.1:8000/admin1/displayserviceadmin/":
else:
print("Store Added successful!")
print("Store Added Failed!")
    # Fixing the if statement and __main__ block
    def tearDown(self):
        if self.driver:  # Check if driver is initialized
            self.driver.quit()

if __name__ == '__main__':  # Correctly format the __main__ block
    unittest.main()
```

Screenshot:



```
============== RESTART: C:\Users\Mal12\Downloads\addemployee.py ==
*************************************************
*****************Place TEST**********************
*************************************************
Place Added successful!
```

Test Case 3

Project Name: Digital Commerce Empowerment Ecosystem (DCEE)					
Add store place Test Case					
Test Case ID: Test_3			**Test Designed By: Josna Mary Thomas**		
Test Priority (Low/Medium/High):			**Test Designed Date: 20/10/2024**		
Module Name: Store page			**Test Executed By: Sona Maria Sebastian**		
Test Title: Add store place			**Test Execution Date: 22/10/2024**		
Description: Add store place					
Pre-Condition: All fields are required					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/ Fail)**
1	Navigate to the login page	http://127.0.0.1:8000/login	Login page should be displayed	Login page was loaded	Pass
2	Enter email	“josna@gmail.com”	The email field should be filled with “josna@gmail.com”	email entered	Pass
3	Enter password	“Josna@123”	The password field should be filled with “Josna@123”	Password entered	Pass
4	Click on the submit button	N/A	If login is successful, admin should be redirected to admin dashboard page	Admin was redirected to admin dashboard page	Pass
5	Navigate to the Add store place Page	http://127.0.0.1:8000/admin1/addservice1	Add store page should be displayed	Add store page was loaded	Pass

6	Enter store	tech Service	The service field should be filled with "Tech Service"	Service entered	Pass
7	Enter Description	"This requires the products related to technical"	The Description field should be filled with a description about the tech service	Description Entered	Pass
8	Click on the Add button	N/A	If store is added successful, then admin should be redirected to admin dashboard page	Admin was redirected to admin dashboard page	Pass
Post-Condition: Store and the place is added.					

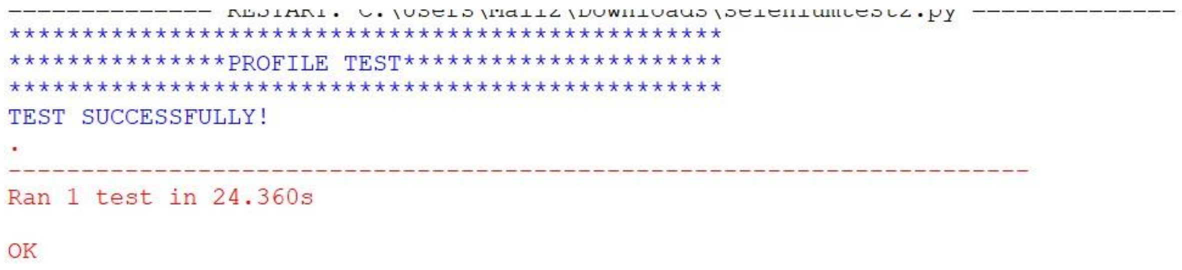
Test Case 4:

Code:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys from time import sleep
int()
driver = webdriver.Chrome()
driver.get("http://127.0.0.1:8000/login1/")
username_input = driver.find_element (By.ID, "email")
username_input.send_keys("teena@gmail.com")
sleep(2)
password_input = driver.find_element(By.ID, "password")
password_input.send_keys("Teena@123")
submit_button = driver.find_element (By.ID, "submit") submit_button.click()
sleep(2)
driver.get("http://127.0.0.1:8000/updatepassword")
sleep(4)
```

```
username_input              =              driver.find_element(By.ID,              "password")
username_input.send_keys("Teena")
sleep(2)
username_input              =              driver.find_element(By.ID,              "cpassword")
username_input.send_keys("Teena")
sleep(2)
submit_button = driver.find_element (By.ID, "register") submit_button.click()
sleep(3)
element = driver.find_element(By.ID, "txt1") value = element.get_attribute("Teena")
print(value)
sleep(3)
print ("************)
************************
print("
"**************PROFILE TEST*****************************")
print("
*****************
*****
if value == "Teena":
else:
| print("Tested successfully")
print("Testing failed")
```

Screenshot:



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

OK
```

Test Case 4

Project Name: Digital Commerce Empowerment Ecosystem (DCEE)	
Add profile Test Case	
Test Case ID: Test_4	**Test Designed By: Josna Mary Thomas**

Test Priority (Low/Medium/High):	**Test Designed Date: 20/10/2024**
Module Name: Profile page	**Test Executed By: Sona Maria Sebastian**
Test Title: Add profile	**Test Execution Date: 22/10/2024**
Description: Add profile	

Pre-Condition: All fields are required

Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/ Fail)**
1	Navigate to the login page	http://127.0.0.1:8000/login	Login page should be displayed	Login page was loaded	Pass
2	Enter email	“josna@gmail.com”	The email field should be filled with “josna@gmail.com”	email entered	Pass
3	Enter password	“Josna@123”	The password field should be filled with “Josna@123”	Password entered	Pass
4	Click on the submit button	N/A	If login is successful, admin should be redirected to admin page	Admin was redirected to admin dashboard page	Pass
5	Navigate to the Add profile Page	http://127.0.0.1:8000/admin1/profile	Add profile page should be displayed	Add profile page was loaded	Pass
6	Enter profile	Mobile shops	The service field should be filled with “Mobile shop”	profile entered	Pass
7	Enter Description	“Contain mobile appliances”	Description field should be filled with a description about the mobile shops	Description Entered	Pass

8	Click on the Add button	N/A	If store is added successful, then admin should be redirected to dashboard page	Admin was redirected to admin dashboard page	Pass
Post-Condition: Store profile added					

CHAPTER 6
IMPLEMENTATION

6.1INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover. The implementation state involves the following tasks:

- Careful planning
- Investigation of system and constraints
- Design of methods to achieve the changeover.

6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we must ensure that the resistance does not build up, as one must make sure that:

- The active user must be aware of the benefits of using the new system. Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process will not take place

6.2.1 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 the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered.

6.2.3 System Maintenance

System maintenance is an effective component such that System maintenance refers to the ongoing activities required to ensure that a system or application operates effectively and efficiently after it has been implemented. It involves regular updates, bug fixes, and performance optimizations to keep the system running smoothly and securely.

System maintenance is essential to ensure that a system remains operational and effective after implementation. By establishing maintenance procedures and following them consistently, project teams can ensure that the system operates smoothly, remains secure, and continues to meet the needs of the end-users.

6.2.4 Hosting

Hosting allows a website to be accessible online by storing its files on a server. Different hosting types cater to various needs: shared hosting is budget-friendly, VPS offers more control, dedicated hosting provides full server access, and cloud hosting improves scalability.

Managed hosting handles technical tasks for you, while self-hosting allows complete control for tech-savvy users. Good hosting ensures website reliability, speed, and security, impacting user experience and SEO.

Render Hosting

Render hosting often includes features like server-side rendering (SSR), static site generation (SSG), and content delivery networks (CDNs) to optimize load times and improve scalability. By pre-rendering pages or using SSR, the server sends fully constructed HTML to users, boosting speed and SEO performance.

Procedure for hosting a website on 000Webhost:

Step 1: Set up a Render Account

• Visit Render's website and sign up for a new account or log in if you already have one.

Step 2: Link GitHub Repository

• Once logged in to Render, navigate to the Dashboard and click on the New button to create a new service.

• Select Web Service as the type of service.

• When prompted, connect your GitHub account to Render if you haven't already done so.

• After linking your GitHub account, choose the DCEE repository from the list of available repositories.

Step 3: Configure Deployment Settings

• Select the appropriate branch for deployment (usually main or master).

• Under Environment, choose the correct runtime for your project (for example, Python 3.x for Flask applications).

• Set the Build Command to install dependencies: pip install -r requirements.txt.

• Set the Start Command to run the flask application: gunicorn DCEE.wsgi.

Step 4: Set up the Database (MongoDB)

• Since the project uses mongodb as the database, no complex database configuration is required on Render. However, make sure the database file is included in the repository
and is properly configured within your project.

• If you plan to migrate your database, ensure that the correct MongoDB commands are run
during deployment to apply migrations (e.g., python manage.py migrate).

Step 5: Deploy the Project

- Once everything is set up, click Create Web Service to begin the deployment process.
- Render will automatically build the project, install dependencies, and start the web service.
- After the deployment is complete, your project will be live at the provided URL.

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

Hosted QR Code:

Screenshot

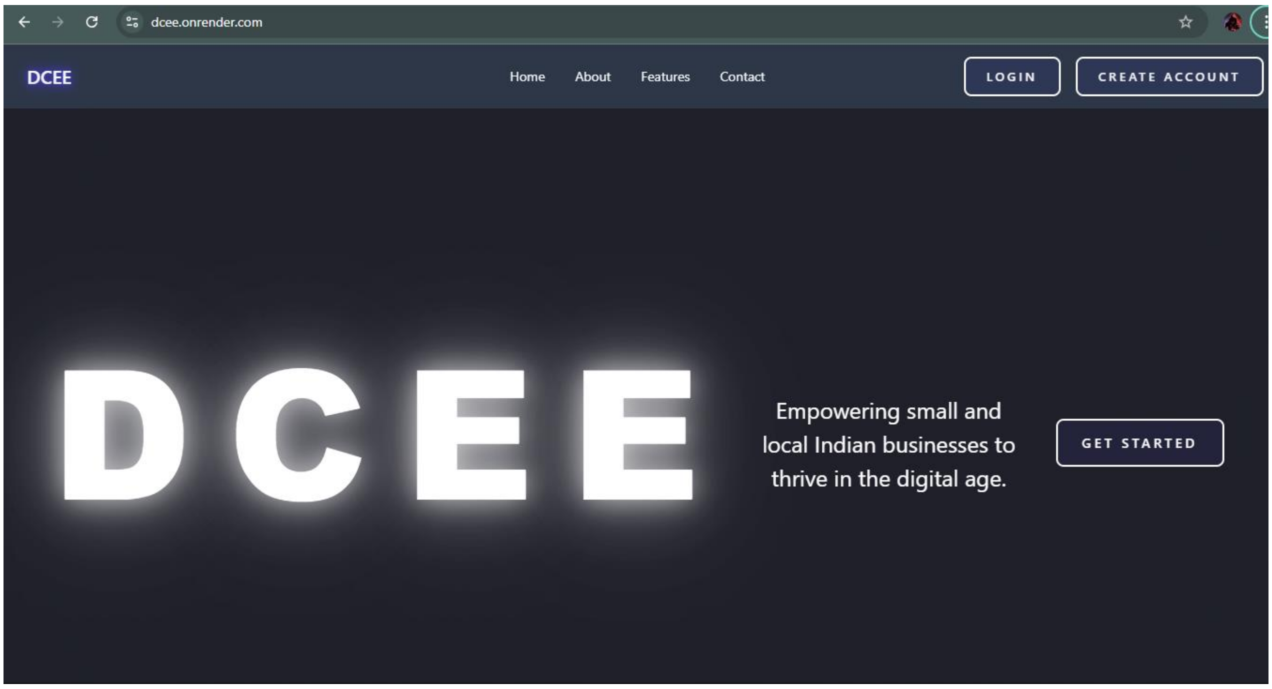


Figure 1. Hosted Site

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The DCEE project is a comprehensive digital empowerment platform designed to bridge critical gaps faced by small and local Indian businesses. Through its intuitive interface, DCEE streamlines essential tasks, from inventory management and secure payment transactions to customer engagement and compliance support. The project integrates a predictive analytics tool powered by the RoBERTa model, providing insights that help businesses forecast demand, optimize stock, and navigate market trends. DCEE's commitment to digital inclusivity and user-friendly tools equips entrepreneurs with the resources to strengthen customer trust, improve operational efficiency, and foster growth. By encouraging digital literacy and facilitating rural payment solutions, DCEE is an all-encompassing tool aiming to sustain local commerce, empower business owners, and nurture an interconnected economic ecosystem.

7.2 FUTURE SCOPE

The future scope of the DCEE project is promising, with opportunities to integrate advanced machine learning models for even more precise demand forecasting and customer behavior insights. Additional features could include multilingual support, catering to a diverse user base across India, and more sophisticated digital literacy modules to help users become proficient in the platform’s tools. As digital payments continue to grow, DCEE could expand its financial offerings with options like microfinancing and lending solutions tailored for small businesses. Furthermore, the platform could incorporate blockchain technology for enhanced transaction transparency and security, building trust with customers and streamlining compliance processes. DCEE's future evolution will continue to empower small businesses to adapt, compete, and grow sustainably in an increasingly digital marketplace.

.

CHAPTER 8
BIBLIOGRAPHY

REFERENCES:

- Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
- Chodorow, K. (2013). MongoDB: The Definitive Guide. O'Reilly Media.
- Norman, D. (2013). The Design of Everyday Things. Basic Books.

WEBSITES:

- https://flask.palletsprojects.com/
- https://arxiv.org/abs/1907.11692.
- https://www.mckinsey.com/industries/small-business

CHAPTER 9
APPENDIX

9.1 Sample Code

Registration Page

```
<!DOCTYPE html>
<html lang="en">
<head>
              <div class="form-group">
                <label class="col-form-label pt-0">Your Name</label>
                <div class="row g-2">
                  <div class="col-6">
                    <input class="form-control" type="text" name="first_name" required
placeholder="First name" oninput="capitalize(this)">
                  </div>
                  <div class="col-6">
                    <input class="form-control" type="text" name="last_name" required
placeholder="Last name" oninput="capitalize(this)">
                  </div>
                </div>
              </div>
              <div class="form-group">
                <label class="col-form-label">Email Address</label>
                <input class="form-control" type="email" name="email" id="email" required
placeholder="Test@gmail.com">
                <div class="invalid-feedback">Please enter a valid Gmail address.</div>
              </div>
              <div class="form-group">
                <label class="col-form-label">Role</label>
                <select class="form-control" name="role" id="role" required>
                  <option value="" disabled selected>Select your role</option>
                  <option value="storefrontowner">Storefront Owner</option>
                  <option value="customer">Customer</option>
                  <!-- <option value="instructor">Instructor</option> -->
                </select>
              </div>
              <div class="form-group" id="gstin-group" style="display: none;">
```

```
                <label class="col-form-label">GSTIN</label>
                <input class="form-control" type="text" name="gstin" id="gstin"
placeholder="22AAAAA0000A1Z5" pattern="^\d{2}[A-Z]{5}\d{4}[A-Z][1-9A-Z][Z][0-9]$">
                <div class="invalid-feedback">Please enter a valid GSTIN (e.g.,
22AAAAA0000A1Z5).</div>
              </div>
                </div>
                <button class="btn btn-primary btn-block w-100" type="submit">Create
Account</button>
              </div>
              <h6 class="text-muted mt-4 or">Or signup with</h6>
              <div class="social mt-4">
                <div class="btn-showcase"><a class="btn btn-light"
href="https://accounts.google.com" target="_blank"><i class="txt-linkedin" data-
feather="linkedin"></i> Google </a></div>
              </div>
              <p class="mt-4 mb-0 text-center">Already have an account?<a class="ms-2"
href="{{ url_for('auth.login') }}">Sign in</a></p>
            </form>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
</script>
</body>
</html>
```

Login Page

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="Riho admin is super flexible, powerful, clean &amp; modern
responsive bootstrap 5 admin template with unlimited possibilities.">
  <meta name="keywords" content="admin template, Riho admin template, dashboard template, flat
admin template, responsive admin template, web app">
  <meta name="author" content="pixelstrap">
  <link rel="icon" href="{{ url_for('static', filename='images/favicon.png') }}" type="image/x-icon">
  <link rel="shortcut icon" href="{{ url_for('static', filename='images/favicon.png') }}" type="image/x-
icon">
  <title>DCEE</title>
  <!-- Google font-->
</head>
<body>
  <!-- login page start-->
  <div class="container-fluid">
    <div class="row">
      <div class="col-xl-5"><img class="bg-img-cover bg-center" src="{{ url_for('static',
filename='images/login/1.jpg') }}" alt="login page"></div>
      <div class="col-xl-7 p-0">
        <div class="login-card login-dark">
          <div>
            <div>
              <a class="logo text-start"><img class="img-fluid for-dark" src="{{ url_for('static',
filename='images/logo/1.png') }}" alt="login page"><img class="img-fluid for-light" src="{{
url_for('static', filename='images/logo/1.png') }}" alt="login page"></a>
            </div>
            <div class="login-main">
              <form class="theme-form" method='post'>
                <h4>Sign in to account</h4>
                <p>Enter your email & password to login</p>

                <!-- Display flash messages -->
                {% with messages = get_flashed_messages(with_categories=true) %}
                  {% if messages %}
                    <div>
                      {% for category, message in messages %}
                        <p style="color: red;">{{ message }}</p>
                      {% endfor %}
                    </div>
                  {% endif %}
                {% endwith %}

                <div class="form-group">
                  <label class="col-form-label">Email Address</label>
                  <input class="form-control" type="email" name="email" required
placeholder="email">
                </div>

  <!-- latest jquery-->
  <script src="{{ url_for('static', filename='js/jquery.min.js') }}"></script>
  <!-- Bootstrap js-->
  <script src="{{ url_for('static', filename='js/bootstrap/bootstrap.bundle.min.js') }}"></script>
  <!-- feather icon js-->
  <script src="{{ url_for('static', filename='js/icons/feather-icon/feather.min.js') }}"></script>
  <script src="{{ url_for('static', filename='js/icons/feather-icon/feather-icon.js') }}"></script>
  <!-- scrollbar js-->
```

```
    <script src="{{ url_for('static', filename='js/config.js') }}"></script>
    <!-- Theme js-->
    <script src="{{ url_for('static', filename='js/script.js') }}"></script>
    <!-- Toggle Password Visibility -->
    <script>
      document.addEventListener('DOMContentLoaded', function () {
        const togglePassword = document.getElementById('togglePassword');
        const password = document.getElementById('password');

        togglePassword.addEventListener('click', function () {
          const type = password.getAttribute('type') === 'password' ? 'text' : 'password';
          password.setAttribute('type', type);
          this.textContent = type === 'password' ? 'Show' : 'Hide';
        });
      });
    </script>
</body>
</html>
```

Storeowner Dashboard

```
<!DOCTYPE html>
<html lang="en">
  <head>

            <div class="welcome-content d-xl-block d-none"><span class="text-truncate col-12">Here's what's
happening with your store today. </span></div>
          </div>
          <div class="nav-right col-xxl-7 col-xl-6 col-md-7 col-8 pull-right right-header p-0 ms-auto">
            <ul class="nav-menus">

              <li>
                <div class="mode"><i class="moon" data-feather="moon"> </i></div>
              </li>

<!-- End Product Registration Modal -->

          <!-- footer start-->
          <footer class="footer">
            <div class="container-fluid">
              <div class="row">
                <div class="col-md-12 footer-copyright text-center">
                  <p class="mb-0">Copyright 2024 © DCEE  </p>
                </div>
              </div>
            </div>
          </footer>
        </div>
      </div>
      <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

      <!-- DataTables JS -->
      <script type="text/javascript"
src="https://cdn.datatables.net/1.11.5/js/jquery.dataTables.min.js"></script>
      <script type="text/javascript"
```

```
src="https://cdn.datatables.net/1.11.5/js/dataTables.bootstrap5.min.js"></script>
   <script>
    $(document).ready(function() {
      let storeTable;

      function initDataTable() {
        if ($.fn.DataTable.isDataTable('#storeTable')) {
          storeTable.ajax.reload();
        } else {
          storeTable = $('#storeTable').DataTable({
            "ajax": {
              "url": "{{ url_for('storeowner.get_stores') }}",
              "dataSrc": "",
              "error": function(xhr, error, thrown) {
                console.error('Error loading store data:', error);
              }
            },
            "columns": [
              { "data": "store_name" },
              { "data": "store_type" },
              { "data": "store_address" },
              { "data": "store_gstin" },
              { "data": "store_owner_id" },
              { "data": "store_established_date" }
            ],
            "initComplete": function(settings, json) {
              console.log('DataTable initialization complete');
              console.log('Received data:', json);
            }
          });
        }
      }

      $('#storeDetailsModal').on('shown.bs.modal', function() {
        console.log('Store Details modal shown, initializing DataTable...');
        initDataTable();
      });
    });
    </script>
      function validateProductField($field) {
        var fieldId = $field.attr('id');
        var value = $field.val();
        var isValid = true;
        var errorMessage = '';

        switch(fieldId) {
          case 'product_name':
            if (!/^[A-Za-z0-9\s]+$/.test(value)) { // Allow alphabets and numbers
              isValid = false;
              errorMessage = 'Product name should contain only alphabets and numbers';
            }
            break;
          case 'product_price':
            if (isNaN(value) || parseFloat(value) <= 0) { // Ensure it's a positive number
              isValid = false;
```

```
                        errorMessage = 'Price must be a positive number';
                    }
                    break;
                case 'product_quantity':
                    if (isNaN(value) || parseInt(value) < 0) { // Ensure it's a non-negative integer
                        isValid = false;
                        errorMessage = 'Quantity must be a non-negative integer';
                    }
                    break;
            }

            if (isValid) {
                $field.removeClass('is-invalid').addClass('is-valid');
                $field.siblings('.invalid-feedback').hide();
            } else {
                $field.removeClass('is-valid').addClass('is-invalid');
                $field.siblings('.invalid-feedback').text(errorMessage).show();
            }

            return isValid;
        }

        function validateProductForm() {
            var isValid = true;
            $('#productRegistrationForm input[required], #productRegistrationForm
select[required]').each(function() {
                if (!validateProductField($(this))) {
                    isValid = false;
                }
            });
            return isValid;
        }
    });
    </script>
    <!-- End Product Registration Modal JS -->
<!-- End Product Details Modal JS -->

<!-- Store Registration Form AJAX Submission -->
<script>
$(document).ready(function() {
    // Real-time validation for store registration
    $('#store_name, #store_type, #store_gstin, #store_address').on('input change', function() {
        validateField($(this));
    });

    $('#storeRegistrationForm').on('submit', function(e) {
        e.preventDefault();
        var form = $(this);
        var url = form.attr('action');

        // Final validation before submission
        var isValid = validateForm();

        if (isValid) {
            $.ajax({
                type: 'POST',
                url: url,
                data: form.serialize(),
                success: function(data) {
                    if (data.success) {
                        $('#registrationMessage').html('<div class="alert alert-success">' + data.message +
'</div>');
                        $('#registrationMessage').show();
                        form[0].reset(); // Reset the form
                    } else {
                        $('#registrationMessage').html('<div class="alert alert-danger">' + data.message + '</div>');
                        $('#registrationMessage').show();
                    }
                },
                error: function() {
                    $('#registrationMessage').html('<div class="alert alert-danger">An error occurred. Please try
again.</div>');
                    $('#registrationMessage').show();
                }
            });
        }
    });

<script>
$(document).ready(function() {
  // Load product overview when the page loads
  loadProductOverview();

  function loadProductOverview() {
    $.ajax({
      url: "{{ url_for('storeowner.get_product_overview') }}",
      type: 'GET',
      success: function(response) {
        $('#productOverview').html(response);
      },
      error: function(xhr, status, error) {
        console.error('Error loading product overview:', error);
        $('#productOverview').html('<p>Error loading product overview. Please try again later.</p>');
      }
    });
  }

  function showProductDetails() {
    // Hide the product overview
    $('#productOverview').hide();

    // Show the existing product details section
    $('#productDetails').show();

    // If you need to refresh the product details, you can add that logic here
    // For example, you might call a function to reload the product data
    // reloadProductDetails();
  }
});
</script>
```

```
<!-- User Profile Modal -->
<div class="modal fade" id="userProfileModal" tabindex="-1" aria-labelledby="userProfileModalLabel"
aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="userProfileModalLabel">My Profile</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <p><strong>First Name:</strong> <span id="userFirstName"></span></p>
        <p><strong>Email:</strong> <span id="userEmail"></span></p>
        <p><strong>Phone Number:</strong> <span id="userPhone"></span></p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-primary" id="editProfileBtn">Edit Profile</button>
        <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>
      </div>
    </div>
  </div>
</div>
 // Handle click on My Profile link
 $('#myProfileLink').on('click', function(e) {
   e.preventDefault();
   $('#userProfileModal').modal('show');
 });
});
</script>

   </body>
 </html>
```

Customer Dashboard

```
    <!DOCTYPE html>
    <html lang="en">
      <head>
                <li class="profile-nav onhover-dropdown pe-0 py-0">
                  <div class="media profile-media">
                      <div class="media-body"><span>{{ user.first_name }}</span>
                        <p class="mb-0 font-roboto">Customer <i class="middle fa fa-angle-down"></i></p>
                      </div>
                  </div>
                  <ul class="profile-dropdown onhover-show-div">
                    <li><a href="#" id="myProfileLink"><i data-feather="user"></i><span>My Profile
</span></a></li>
                    <li><a href="{{ url_for('auth.login') }}"><i data-feather="log-out"> </i><span>Log
out</span></a></li>
                  </ul>
                </li>
              </ul>
            </div>
            <script class="result-template" type="text/x-handlebars-template">
              <div class="ProfileCard u-cf">
            <div class="ProfileCard-avatar"><svg xmlns="http://www.w3.org/2000/svg" width="24" height="24"
viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2" stroke-linecap="round" stroke-
linejoin="round" class="feather feather-airplay m-0"><path d="M5 17H4a2 2 0 0 1-2-2V5a2 2 0 0 1 2-2h16a2 2
0 0 1 2 2v10a2 2 0 0 1-2 2h-1"></path><polygon points="12 15 17 21 7 21 12 15"></polygon></svg></div>
            <div class="ProfileCard-details">
            <div class="ProfileCard-realName">{{name}}</div>
            </div>
            </div>
           </script>
           <script class="empty-template" type="text/x-handlebars-template"><div class="EmptyMessage">Your
search turned up 0 results. This most likely means the backend is down, yikes!</div></script>
          </div>
         </div>

        <!-- Edit Profile Modal -->
        <div class="modal fade" id="editProfileModal" tabindex="-1" aria-labelledby="editProfileModalLabel"
aria-hidden="true">
          <div class="modal-dialog">
            <div class="modal-content">
              <div class="modal-header">
                <h5 class="modal-title" id="editProfileModalLabel">Edit Profile</h5>
                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
              </div>
              <div class="modal-body">
                <form id="editProfileForm" novalidate>
                  <div class="mb-3">
                    <label for="editFirstName" class="form-label">First Name</label>
                    <input type="text" class="form-control" id="editFirstName" required>
                    <div class="invalid-feedback" id="firstNameError"></div>
                  </div>
                  <div class="mb-3">
                    <label for="editLastName" class="form-label">Last Name</label>
                    <input type="text" class="form-control" id="editLastName" required>
                    <div class="invalid-feedback" id="lastNameError"></div>
                  </div>
                  <!-- Removed email input field -->
                  <p><strong>Email:</strong> <span id="editProfileEmail" class="text-muted"></span></p>
<!-- Display email -->
                </form>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-primary" id="saveProfileBtn">Save Changes</button>
                <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Cancel</button>
              </div>
            </div>
          </div>
        </div>
        <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
      </body>
    </html>
```

9.1 Screen Shots

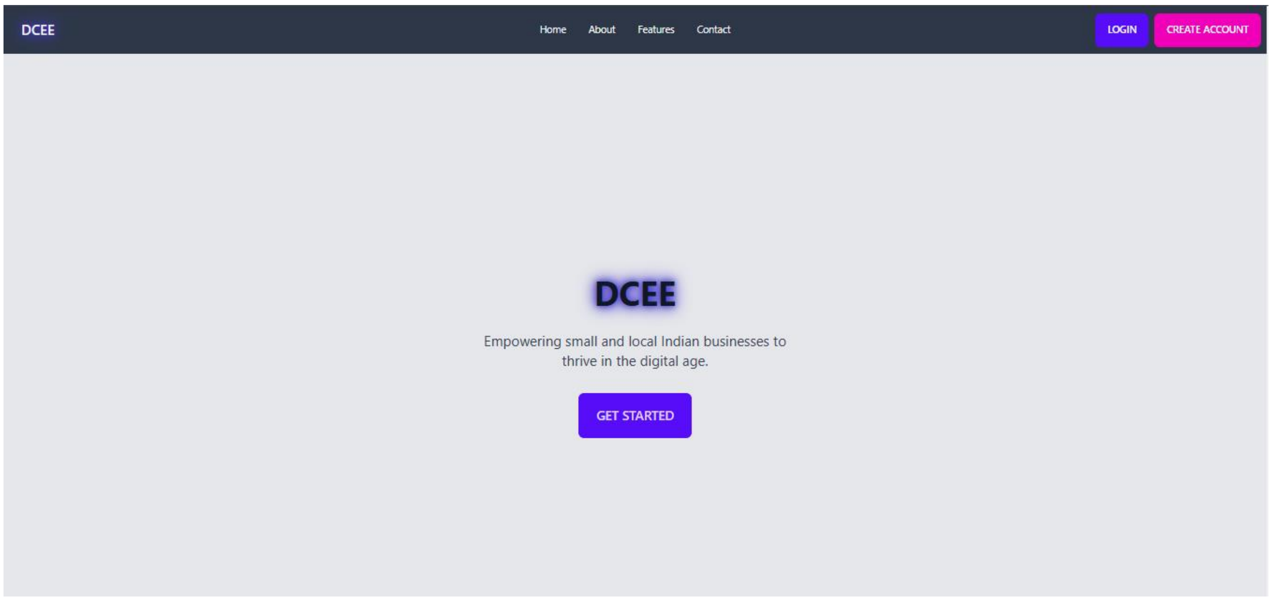

DCEE
Home
About
Features
Contact
LOGIN
CREATE ACCOUNT
DCEE
Empowering small and local Indian businesses to thrive in the digital age.
GET STARTED

Figure 1. Landing Page

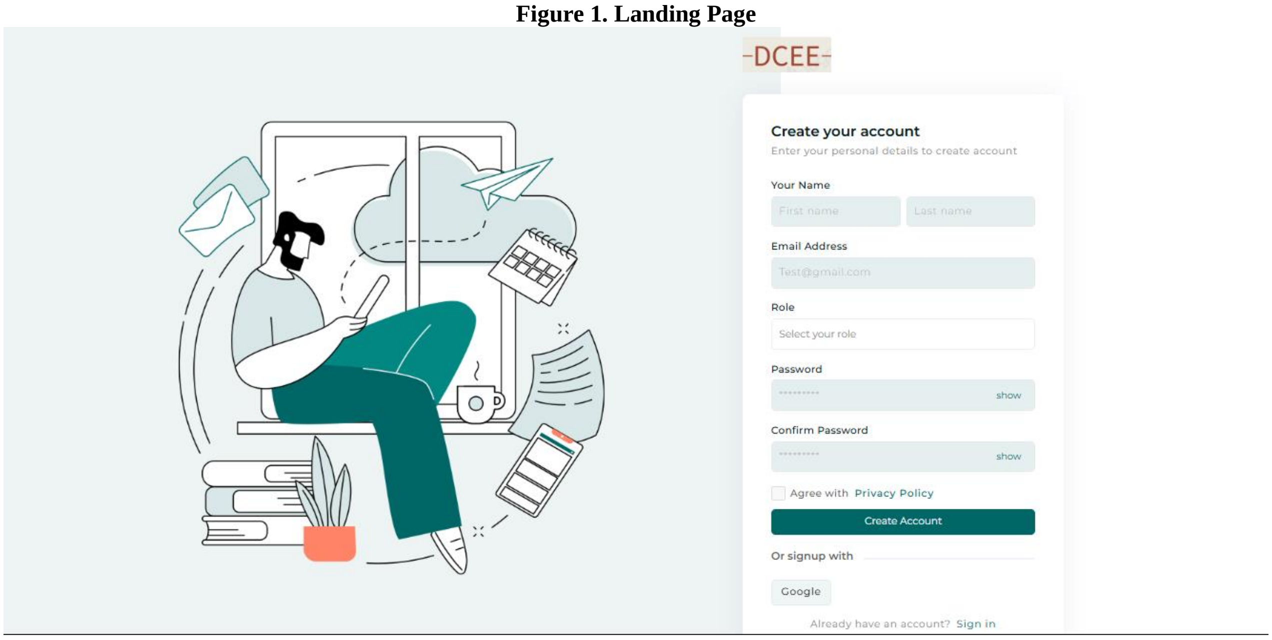
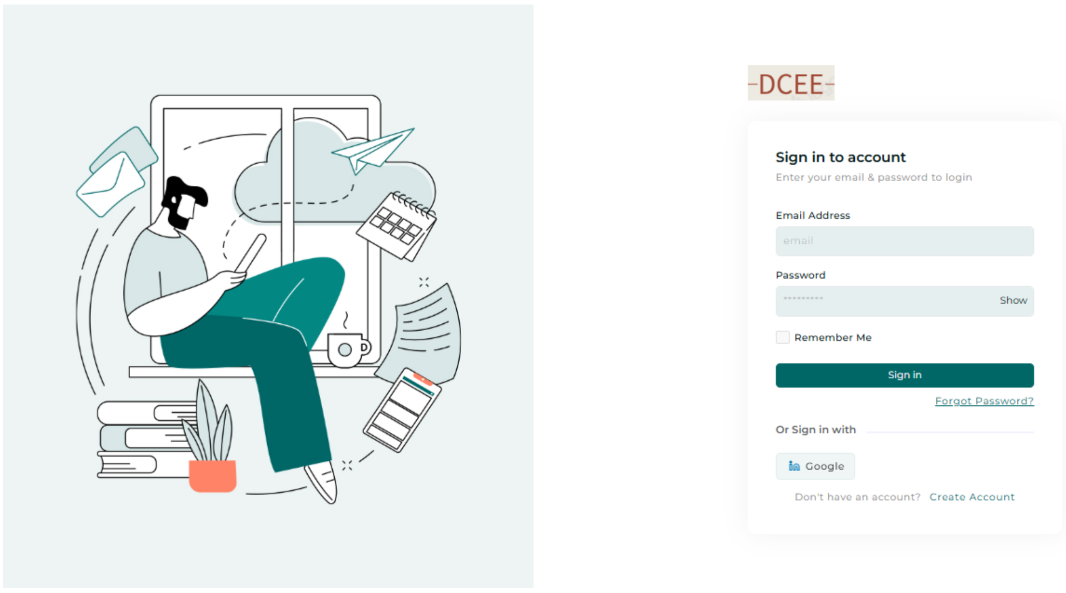


Figure 2. Register


DCEE

Sign in to account

Enter your email & password to login

Email Address

email

Password

Show

Remember Me

Sign in

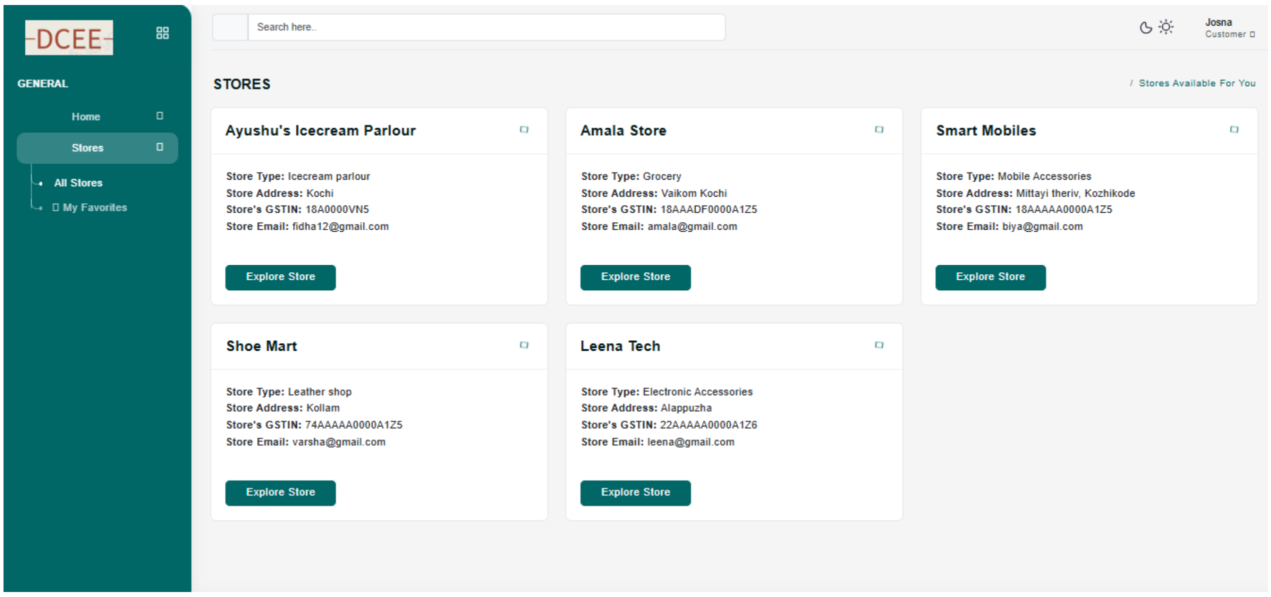
Forgot Password?

Or Sign in with

Google

Don't have an account? Create Account

Figure 3. Login


DCEE

Search here..

Josna
Customer

GENERAL

Home

Stores

All Stores

My Favorites

STORES

/ Stores Available For You

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

Figure 4. Customer Dashboard

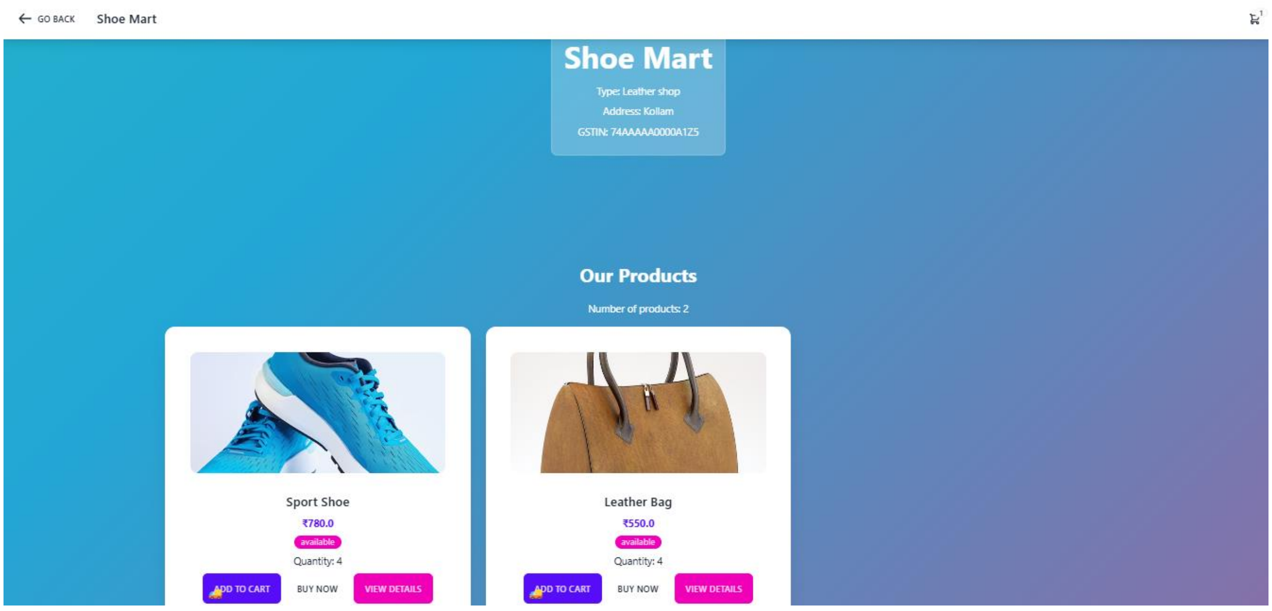
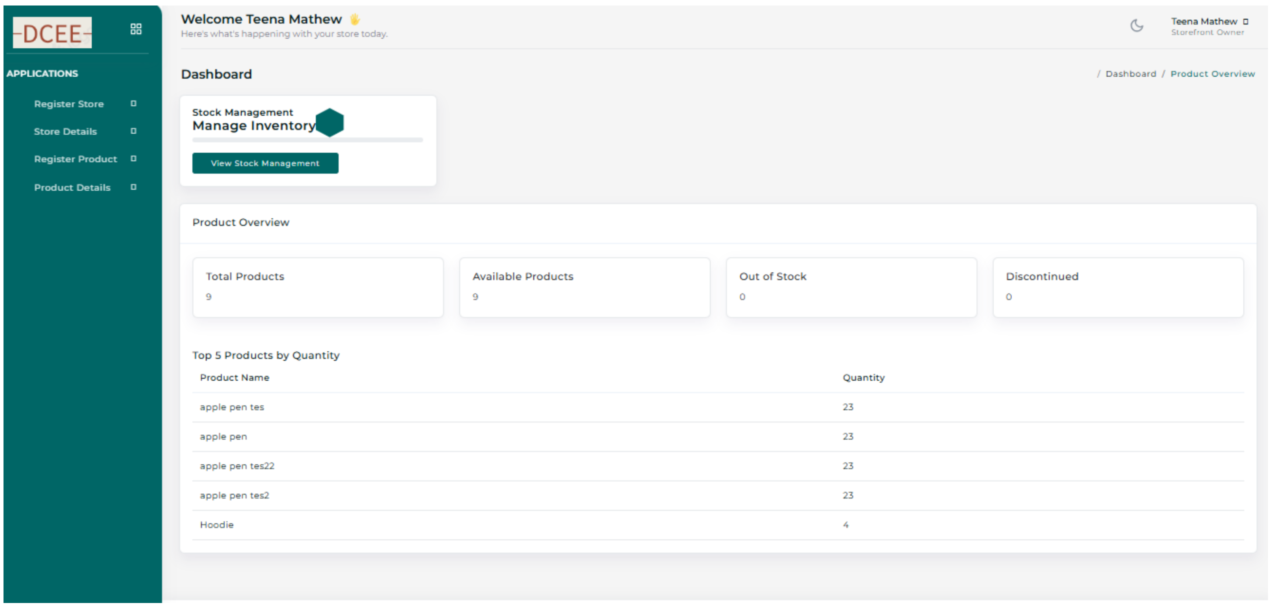


Figure 5. Product Display



Welcome Teena Mathew

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

Dashboard

Stock Management

Manage Inventory

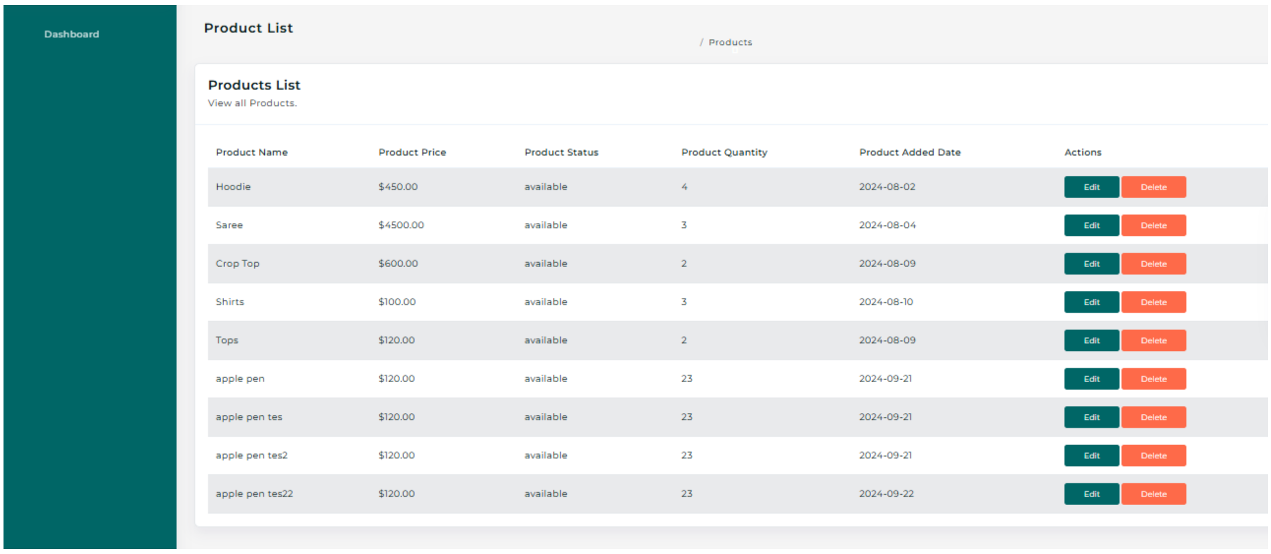
Product Overview

Total Products	Available Products	Out of Stock	Discontinued
9	9	0	0

Top 5 Products by Quantity

Product Name	Quantity
apple pen tes	23
apple pen	23
apple pen tes22	23
apple pen tes2	23
Hoodie	4

Figure 6. Storefront Dashboard



Products List
View all Products.

Product Name	Product Price	Product Status	Product Quantity	Product Added Date	Actions
Hoodie	$450.00	available	4	2024-08-02	Edit Delete
Saree	$4500.00	available	3	2024-08-04	Edit Delete
Crop Top	$600.00	available	2	2024-08-09	Edit Delete
Shirts	$100.00	available	3	2024-08-10	Edit Delete
Tops	$120.00	available	2	2024-08-09	Edit Delete
apple pen	$120.00	available	23	2024-09-21	Edit Delete
apple pen tes	$120.00	available	23	2024-09-21	Edit Delete
apple pen tes2	$120.00	available	23	2024-09-21	Edit Delete
apple pen tes22	$120.00	available	23	2024-09-22	Edit Delete

Figure 7. Product List

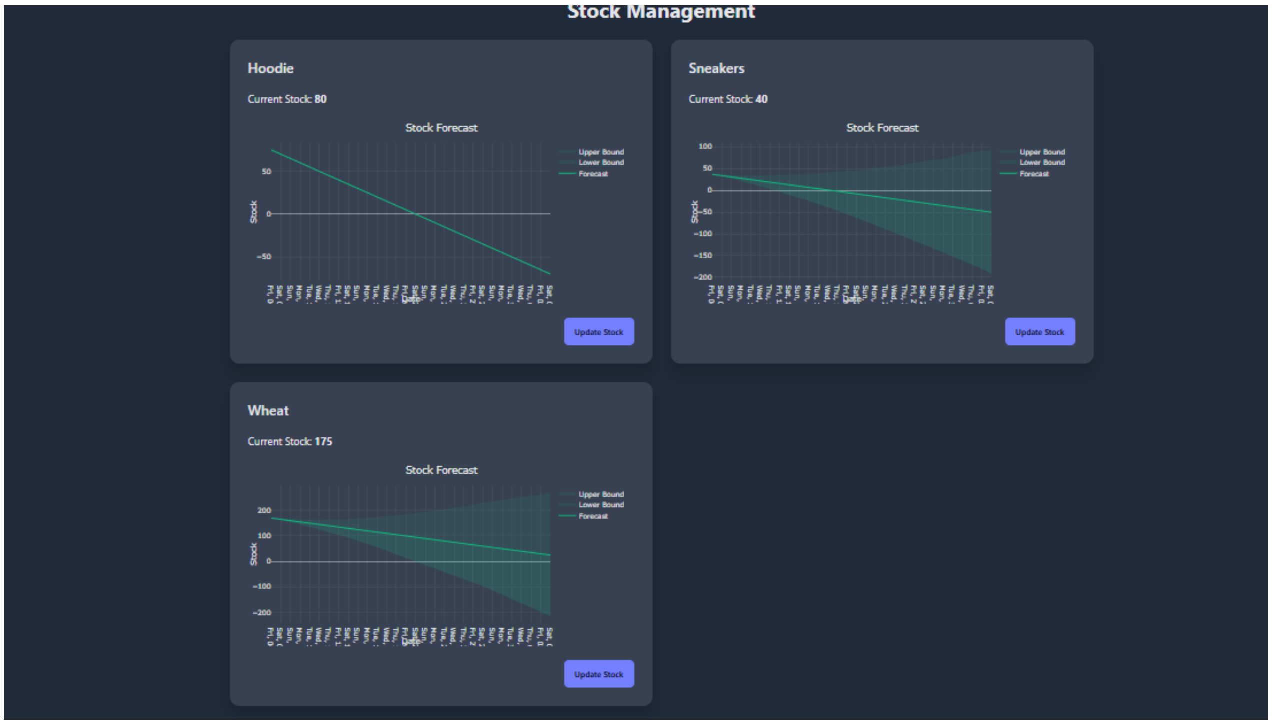


Figure 8. Stock Prediction

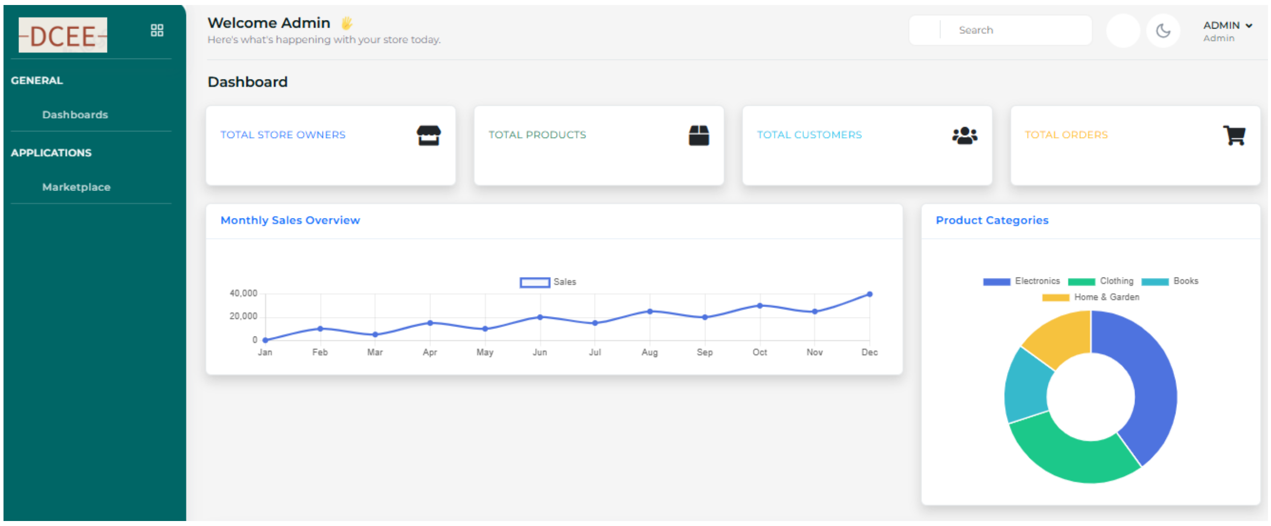
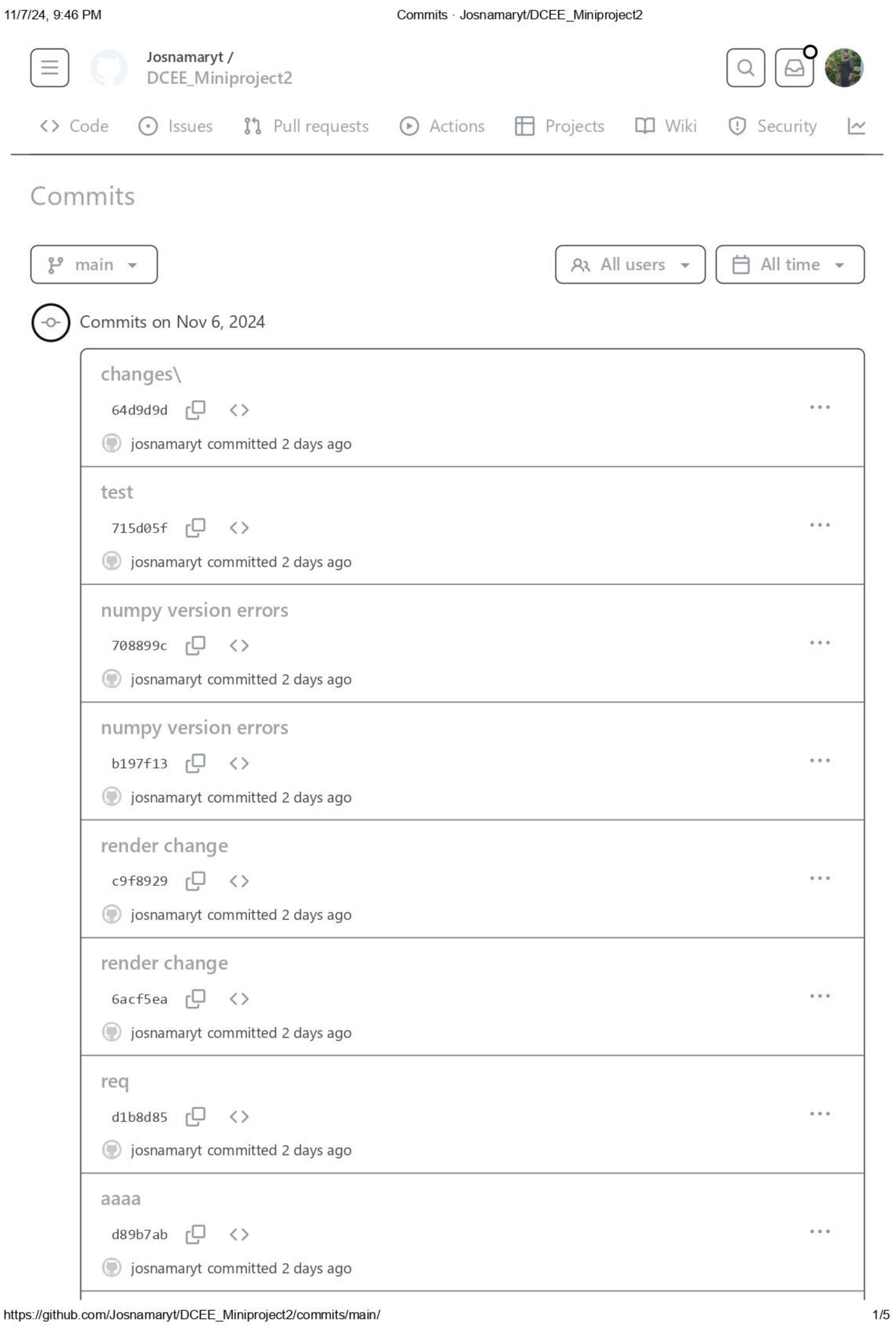


Figure 9. Admin Dashboard

9.3 GIT LOG


11/7/24, 9:46 PM Commits · Josnamaryt/DCEE_Miniproject2

Commits

Commits on Nov 6, 2024

changes\
64d9d9d
josnamaryt committed 2 days ago

test
715d05f
josnamaryt committed 2 days ago

numpy version errors
708899c
josnamaryt committed 2 days ago

numpy version errors
b197f13
josnamaryt committed 2 days ago

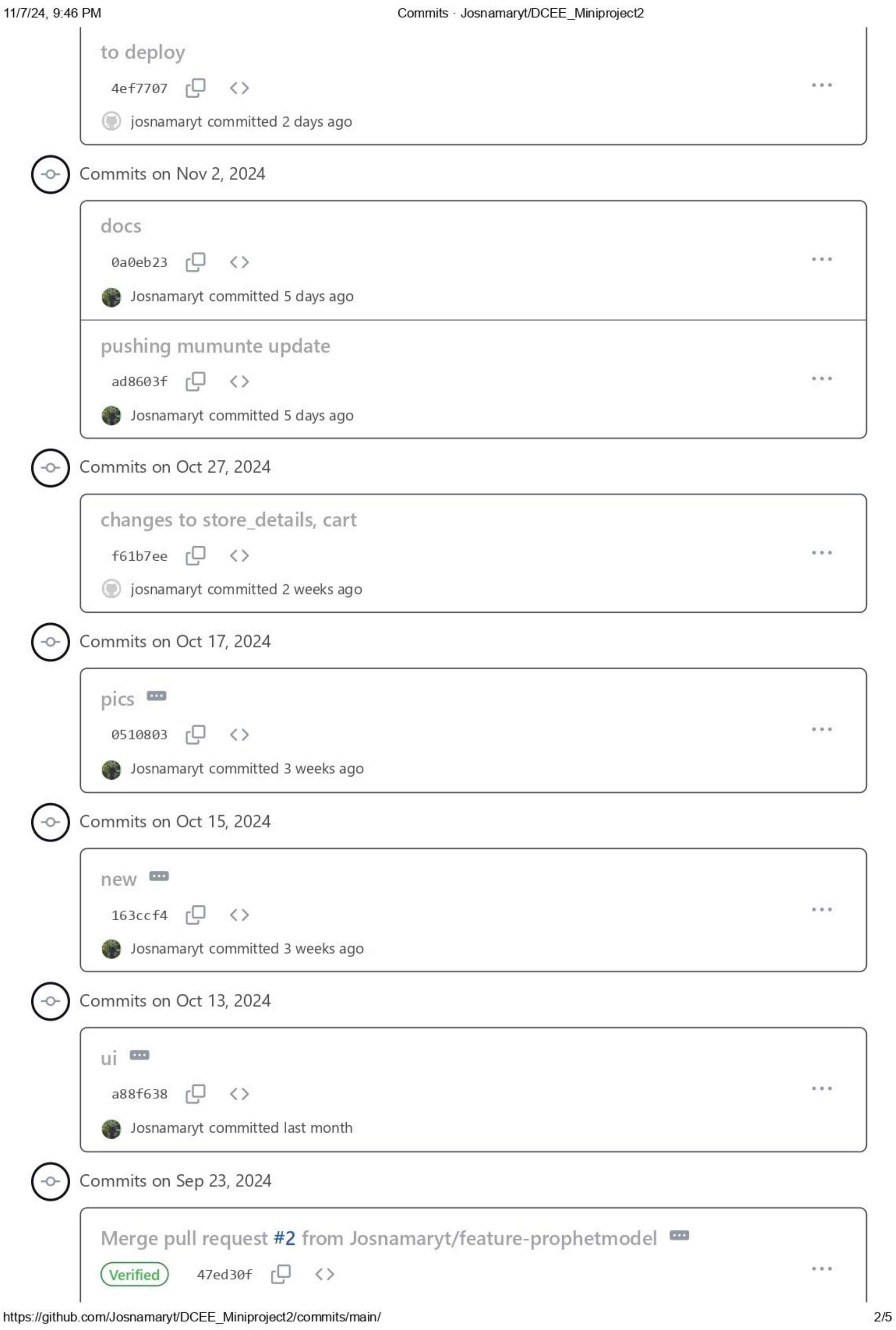
render change
c9f8929
josnamaryt committed 2 days ago

render change
6acf5ea
josnamaryt committed 2 days ago

req
d1b8d85
josnamaryt committed 2 days ago

aaaa
d89b7ab
josnamaryt committed 2 days ago

https://github.com/Josnamaryt/DCEE_Miniproject2/commits/main/ 1/5


11/7/24, 9:46 PM Commits · Josnamaryt/DCEE_Miniproject2

to deploy
4ef7707
josnamaryt committed 2 days ago

Commits on Nov 2, 2024

docs
0a0eb23
Josnamaryt committed 5 days ago

pushing mumunte update
ad8603f
Josnamaryt committed 5 days ago

Commits on Oct 27, 2024

changes to store_details, cart
f61b7ee
josnamaryt committed 2 weeks ago

Commits on Oct 17, 2024

pics
0510803
Josnamaryt committed 3 weeks ago

Commits on Oct 15, 2024

new
163ccf4
Josnamaryt committed 3 weeks ago

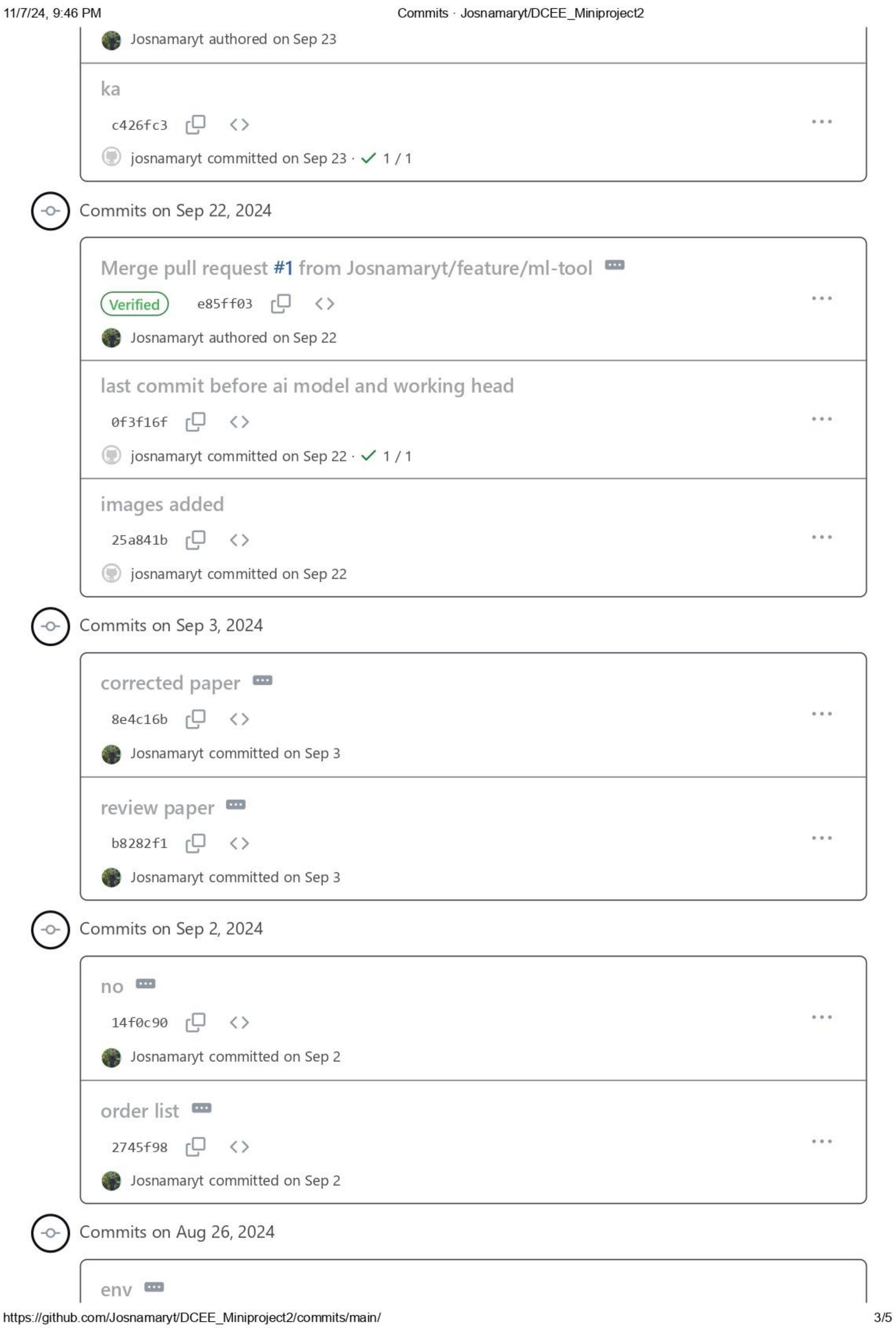
Commits on Oct 13, 2024

ui
a88f638
Josnamaryt committed last month

Commits on Sep 23, 2024

Merge pull request #2 from Josnamaryt/feature-prophetmodel
Verified 47ed30f

https://github.com/Josnamaryt/DCEE_Miniproject2/commits/main/ 2/5


11/7/24, 9:46 PM Commits · Josnamaryt/DCEE_Miniproject2

Josnamaryt authored on Sep 23

ka
c426fc3
josnamaryt committed on Sep 23 · ✓ 1 / 1

Commits on Sep 22, 2024

Merge pull request #1 from Josnamaryt/feature/ml-tool
Verified e85ff03
Josnamaryt authored on Sep 22

last commit before ai model and working head
0f3f16f
josnamaryt committed on Sep 22 · ✓ 1 / 1

images added
25a841b
josnamaryt committed on Sep 22

Commits on Sep 3, 2024

corrected paper
8e4c16b
Josnamaryt committed on Sep 3

review paper
b8282f1
Josnamaryt committed on Sep 3

Commits on Sep 2, 2024

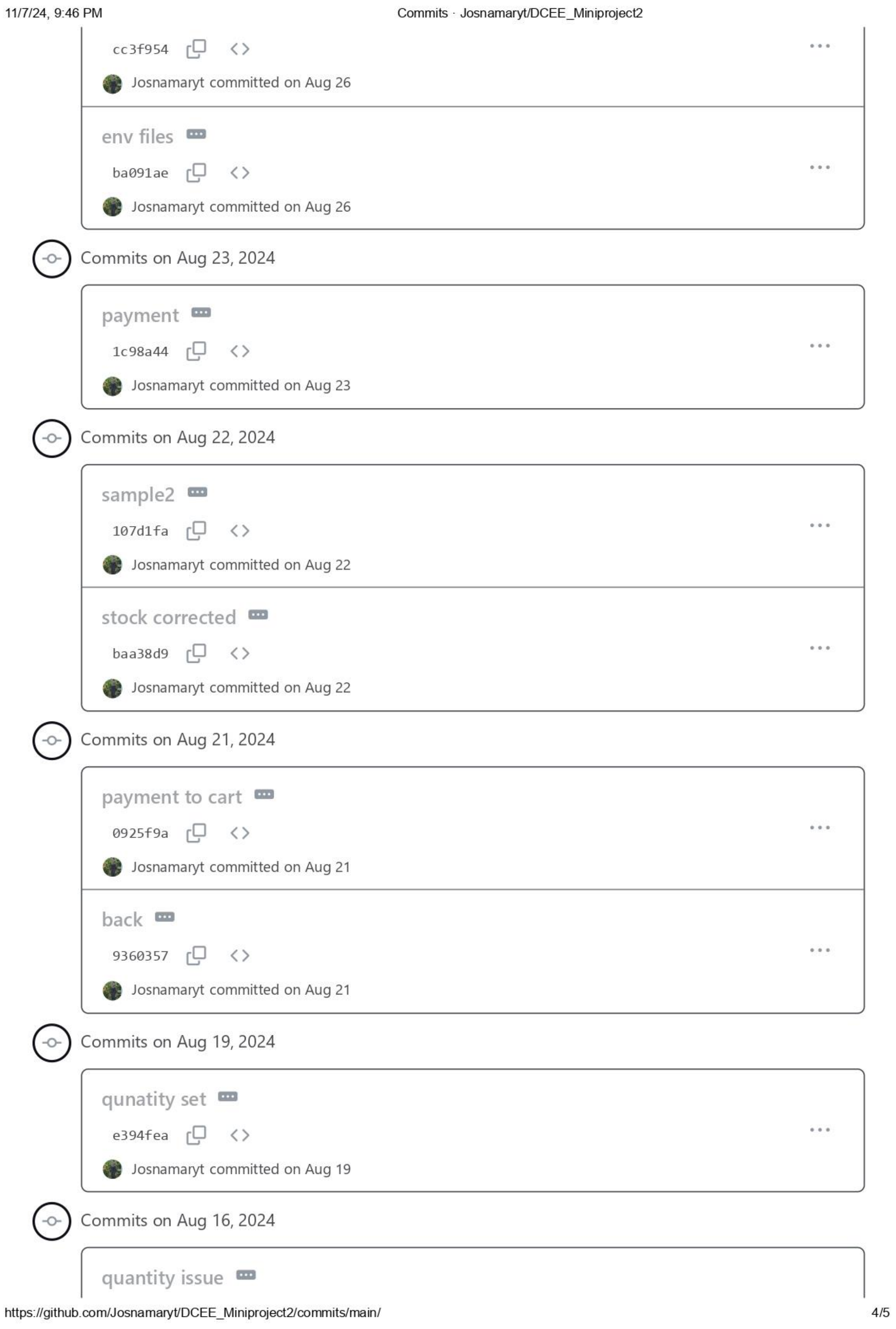
no
14f0c90
Josnamaryt committed on Sep 2

order list
2745f98
Josnamaryt committed on Sep 2

Commits on Aug 26, 2024

env

https://github.com/Josnamaryt/DCEE_Miniproject2/commits/main/ 3/5



11/7/24, 9:46 PM Commits · Josnamaryt/DCEE_Miniproject2

cc3f954

Josnamaryt committed on Aug 26

env files

ba091ae

Josnamaryt committed on Aug 26

Commits on Aug 23, 2024

payment

1c98a44

Josnamaryt committed on Aug 23

Commits on Aug 22, 2024

sample2

107d1fa

Josnamaryt committed on Aug 22

stock corrected

baa38d9

Josnamaryt committed on Aug 22

Commits on Aug 21, 2024

payment to cart

0925f9a

Josnamaryt committed on Aug 21

back

9360357

Josnamaryt committed on Aug 21

Commits on Aug 19, 2024

qunatity set

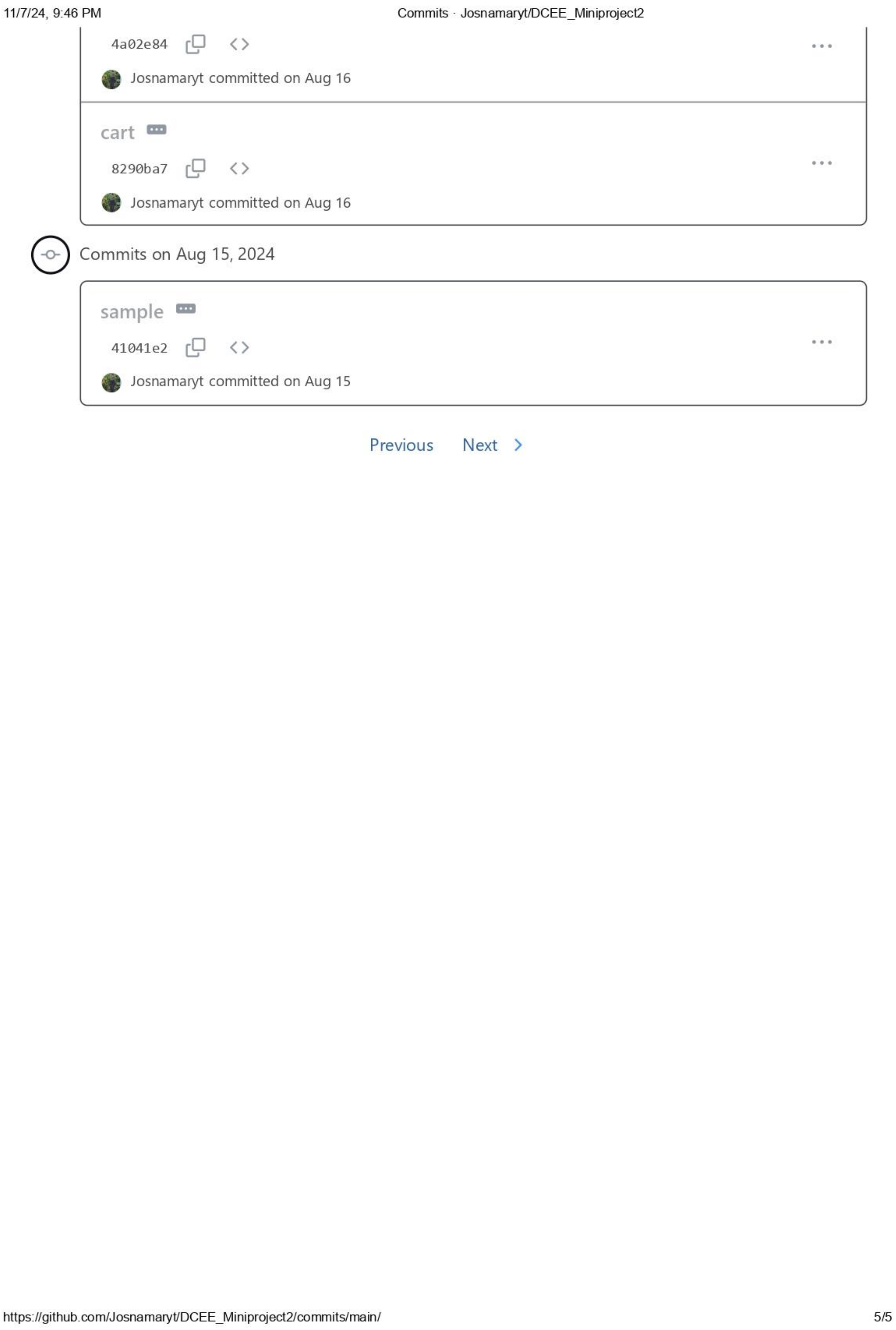
e394fea

Josnamaryt committed on Aug 19

Commits on Aug 16, 2024

quantity issue

https://github.com/Josnamaryt/DCEE_Miniproject2/commits/main/ 4/5


11/7/24, 9:46 PM Commits · Josnamaryt/DCEE_Miniproject2

4a02e84

Josnamaryt committed on Aug 16

cart

8290ba7

Josnamaryt committed on Aug 16

Commits on Aug 15, 2024

sample

41041e2

Josnamaryt committed on Aug 15

Previous Next >

https://github.com/Josnamaryt/DCEE_Miniproject2/commits/main/ 5/5