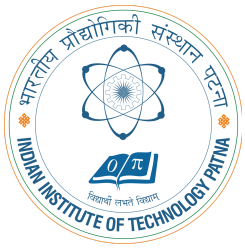


Executive MTECH in Cloud Computing

Semester - 1 Project

Batch - 2 (Session: 2023-25)

Subject: Virtualization & Cloud Computing

Project Name: Electronic MutiChannel-OWN eCommerce website

Team Name: Team - 21

Team Members:

SANJIB Saha
- Role: **Project Management and Product Owner**

Shaurya Pratap Singh
- Role: **Technology Lead Development (Fullstack)**

Shiv Kumar Sah
- Role: **Provisioning and Deployment**

Shivam Srivastava
- Role: **Database Management, Testing, and Documentation**

Document Version

Version	Author	Reason for change	Date
1	Team 21	Project Idea Submission	5th Oct 2023
2	Team 21	Final Project Submission	15th Dec 2023

Index:

1. Introduction
2. Functional specifications
3. Technical specifications
4. Code repository
5. Future landscape
6. Books and reference

1. Introduction:

An e-commerce platform for an electronics retail store, involving the deployment of the retail application to a virtual sandbox on AWS/GCP using a centralization technique. The application deployment is facilitated through Kubernetes and Docker technologies.

2. Functional Specification

We are in the process of creating an eCommerce platform with role-based functionalities. Upon logging in, consumers will be assigned a specific role that dictates their level of access. Depending on their access permissions, users will be directed to either the consumer page or the admin page.

For users with consumer role access, the platform allows them to browse and purchase electronic items. On the other hand, users with admin access have the additional capability to add or remove products and make modifications to product specifications.

3. Technical Specification

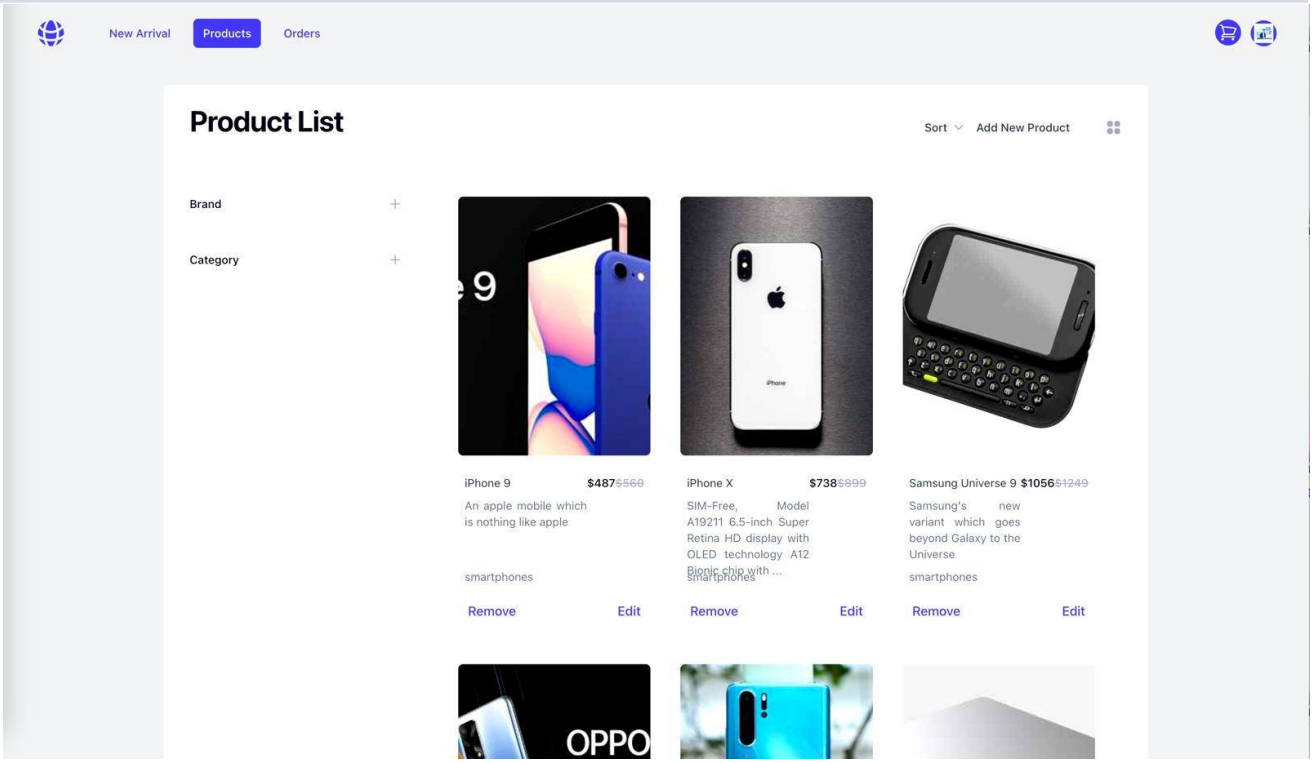
Front-end (React.js):

1. User Authentication:
    - Implemented user authentication using JWT for secure logins and stored tokens securely on the client side.

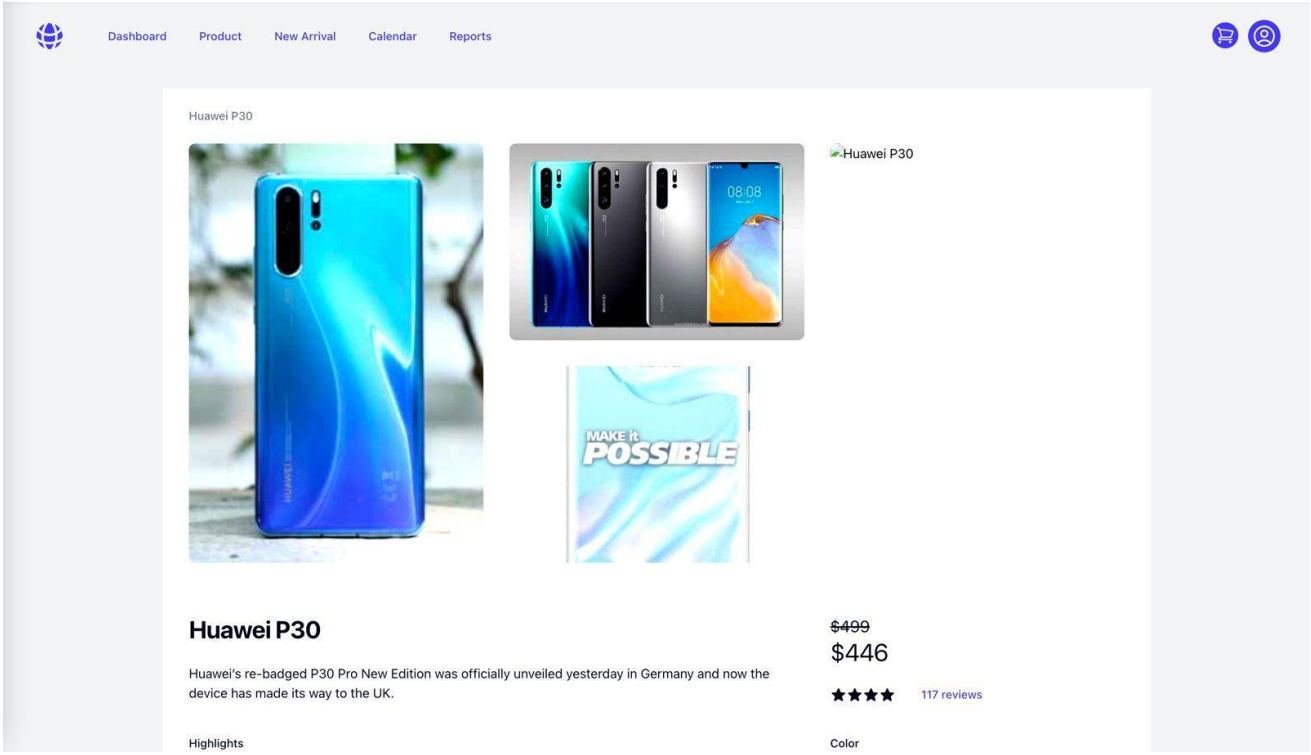
- Utilized React Context API or Redux for managing global state, including user authentication status and role.

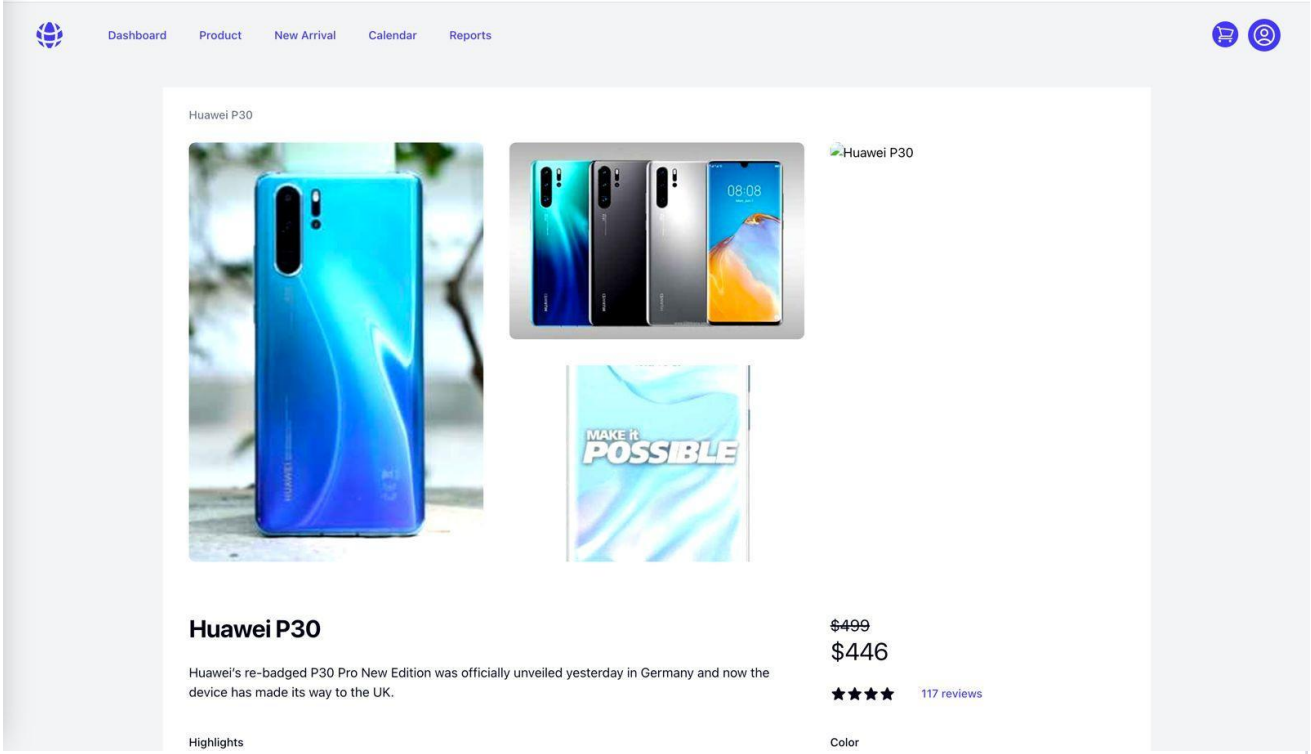
2. Role-Based Routing:
   - Integrated React Router to manage role-based navigation, directing users to the consumer or admin pages based on their assigned roles.
3. Consumer Page:
   - Developed a visually appealing consumer page featuring a product catalog, detailed product views, and a shopping cart component.
   - Utilized React hooks (e.g., useState, useEffect) for state management within functional components.

Consumer product page

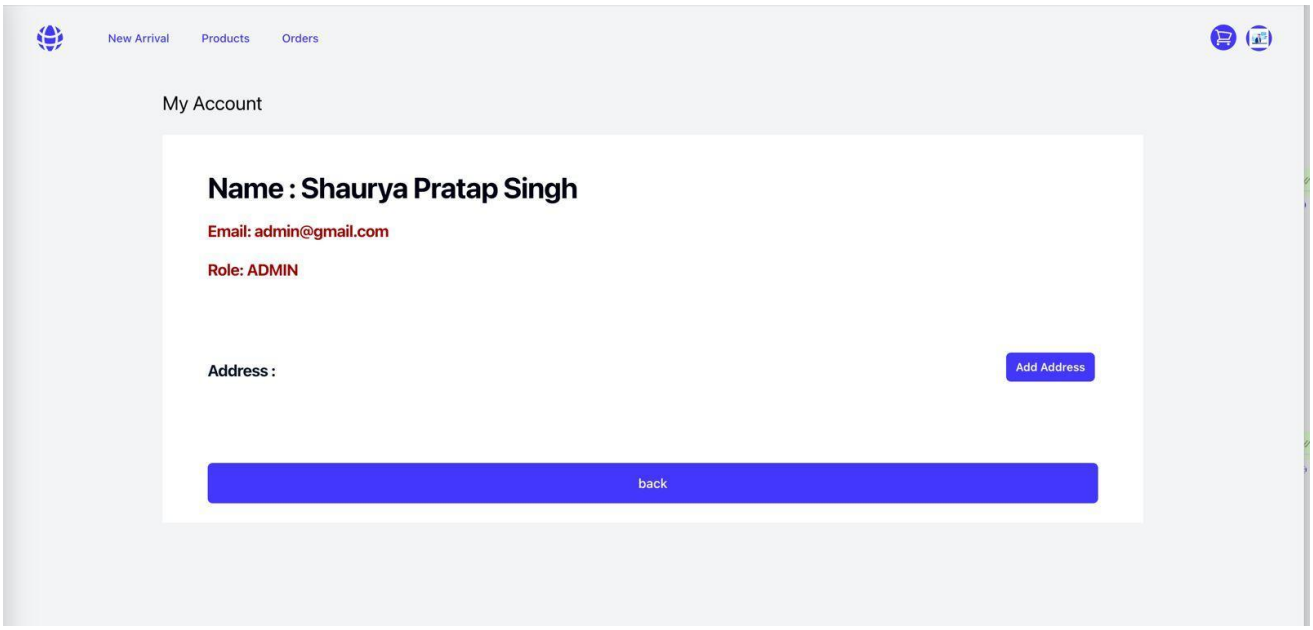


Product specifications page

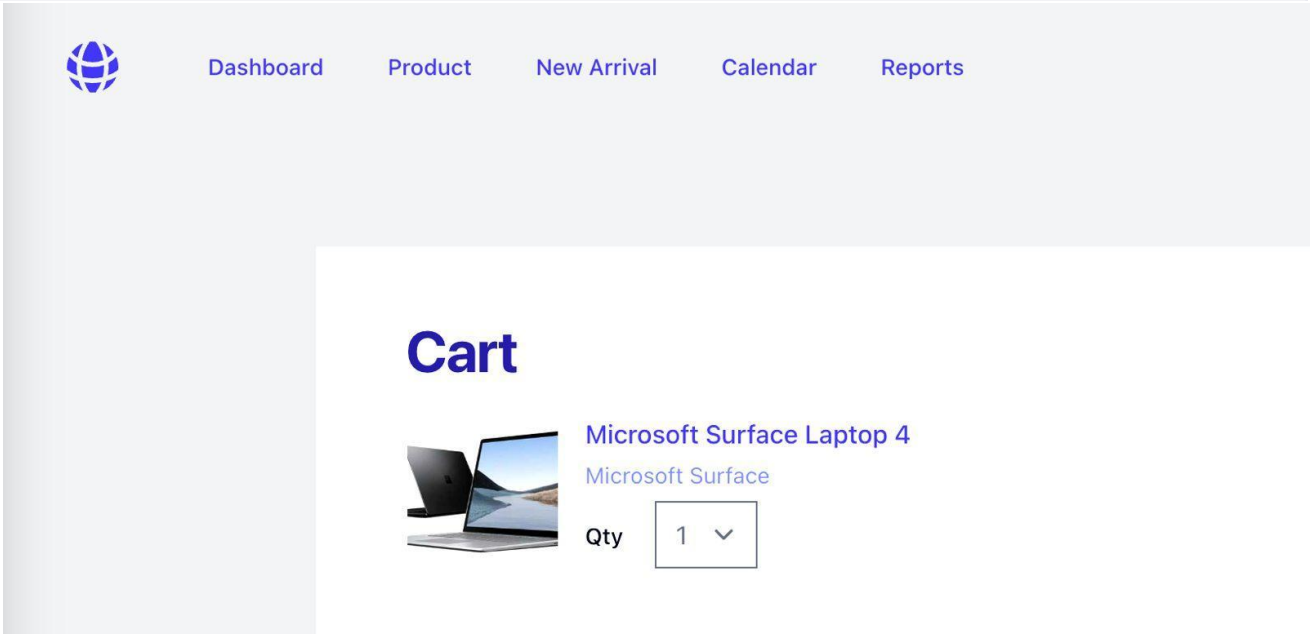




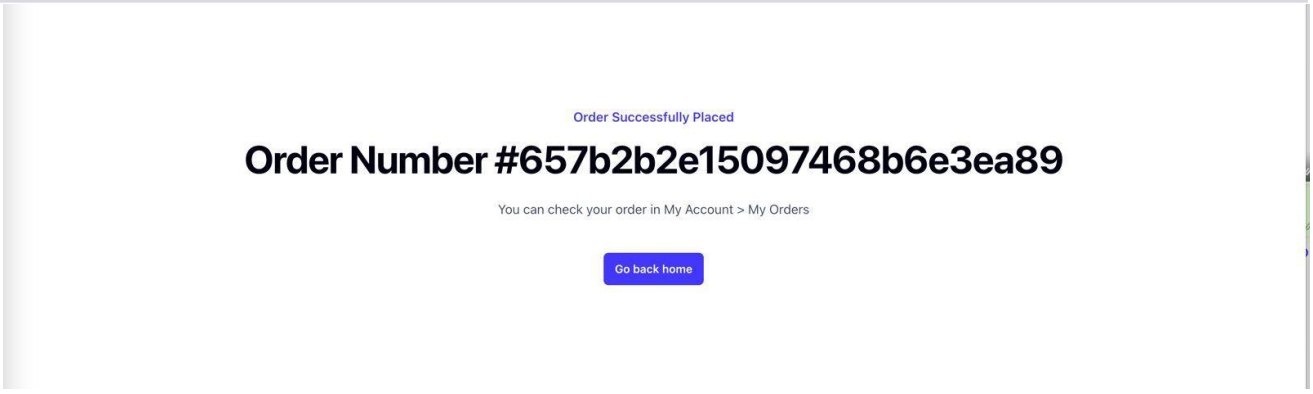
Consumer Profile page



Consumer Shopping cart



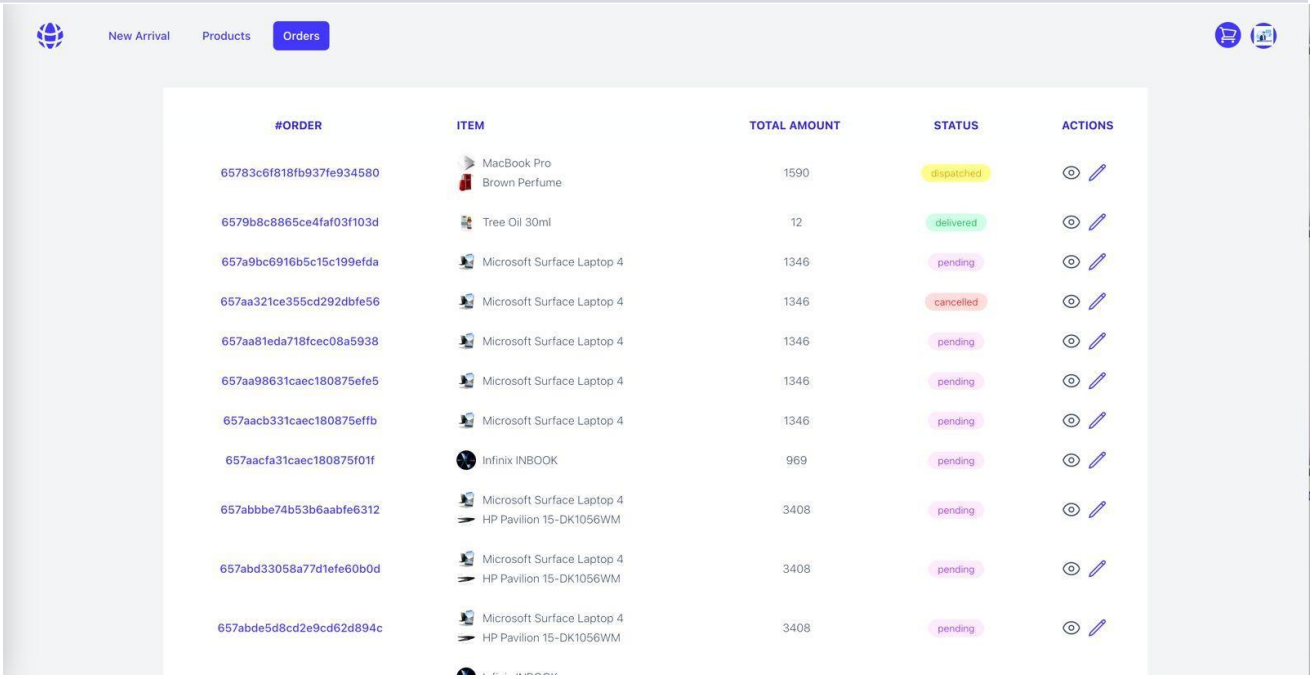
Order ID



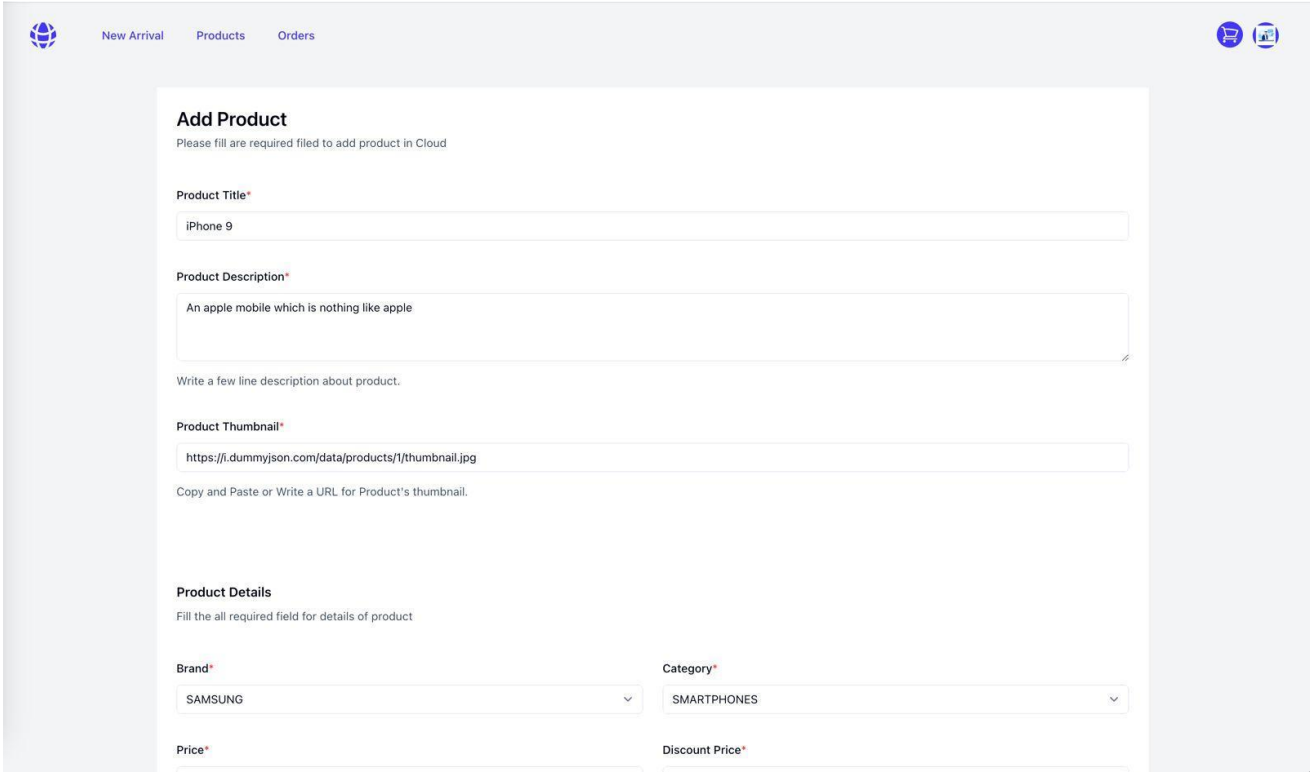
4. Admin Page:

- Designed a responsive admin dashboard with features for product management, including add, delete, and modify functionalities.
- Implemented conditional rendering to display admin-specific components based on the user's role.

Admin page to view orders



Admin page to add product



5. Responsive Design:
    - Ensured a responsive and mobile-friendly design using CSS Grid, Flexbox, or a responsive UI library (e.g., Bootstrap) for a seamless cross-device experience.

Back-end (Express.js/Node.js):

1. RESTful API Endpoints:
    - Created RESTful API endpoints for user authentication, product retrieval, purchase transactions, and admin functionalities using Express.js.
2. Middleware:
    - Implemented middleware functions for authentication and authorization, ensuring that only authorized users can access certain routes.
    - Employed error-handling middleware to gracefully handle and log errors.
3. Database Integration:
    - Connected Express.js with MongoDB using Mongoose for schema definition, data validation, and database operations.
    - Utilized Mongoose population to efficiently retrieve related data.
4. Security Measures:
    - Implemented secure password hashing for user credentials.
    - Configured CORS settings to control cross-origin resource sharing.
    - Employed Helmet.js middleware for HTTP security headers.

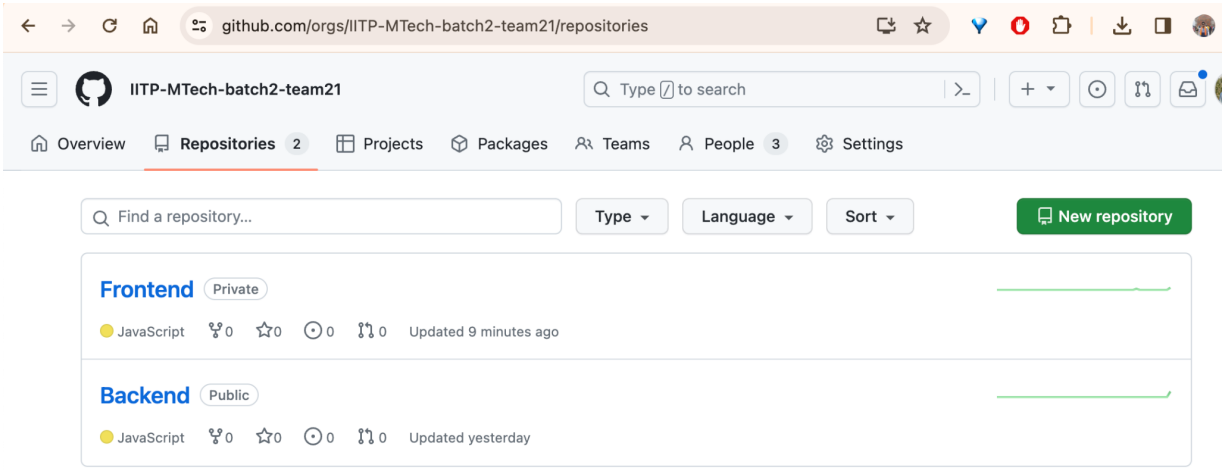
Database (MongoDB):

1. Data Schema:
    - Designed MongoDB schemas for user profiles, product details, and transaction records.
    - Established relationships between collections to represent associations (e.g., user roles).
2. Indexing:
    - Applied appropriate indexes to optimize query performance for common operations, such as user authentication and product retrieval.
3. Data Validation:
    - Implemented Mongoose validation to ensure data integrity and consistency.
    - Validated incoming data against predefined schemas.
4. Scalability Considerations:
    - Explored sharding strategies and optimized queries for potential future scalability.
    - Monitored database performance using tools like MongoDB Atlas monitoring.

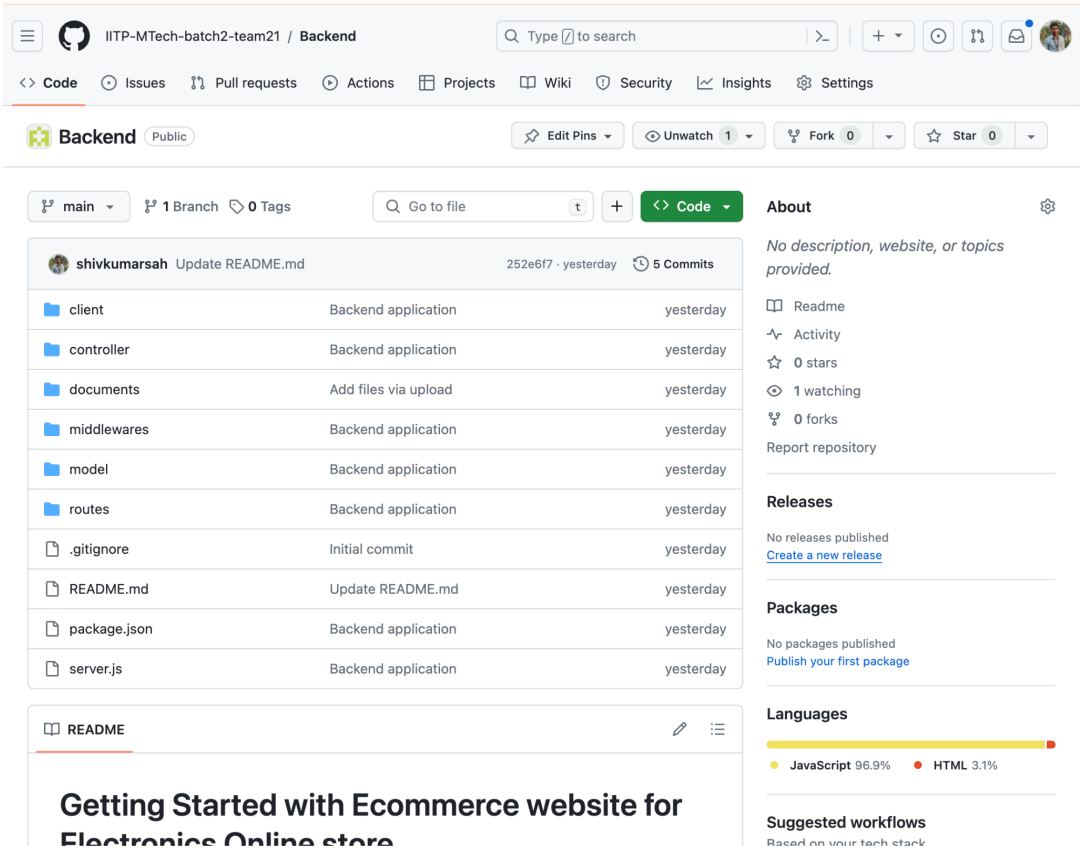
4. Code repository

Project code and articles are committed at the GitHub repository at the below URL.

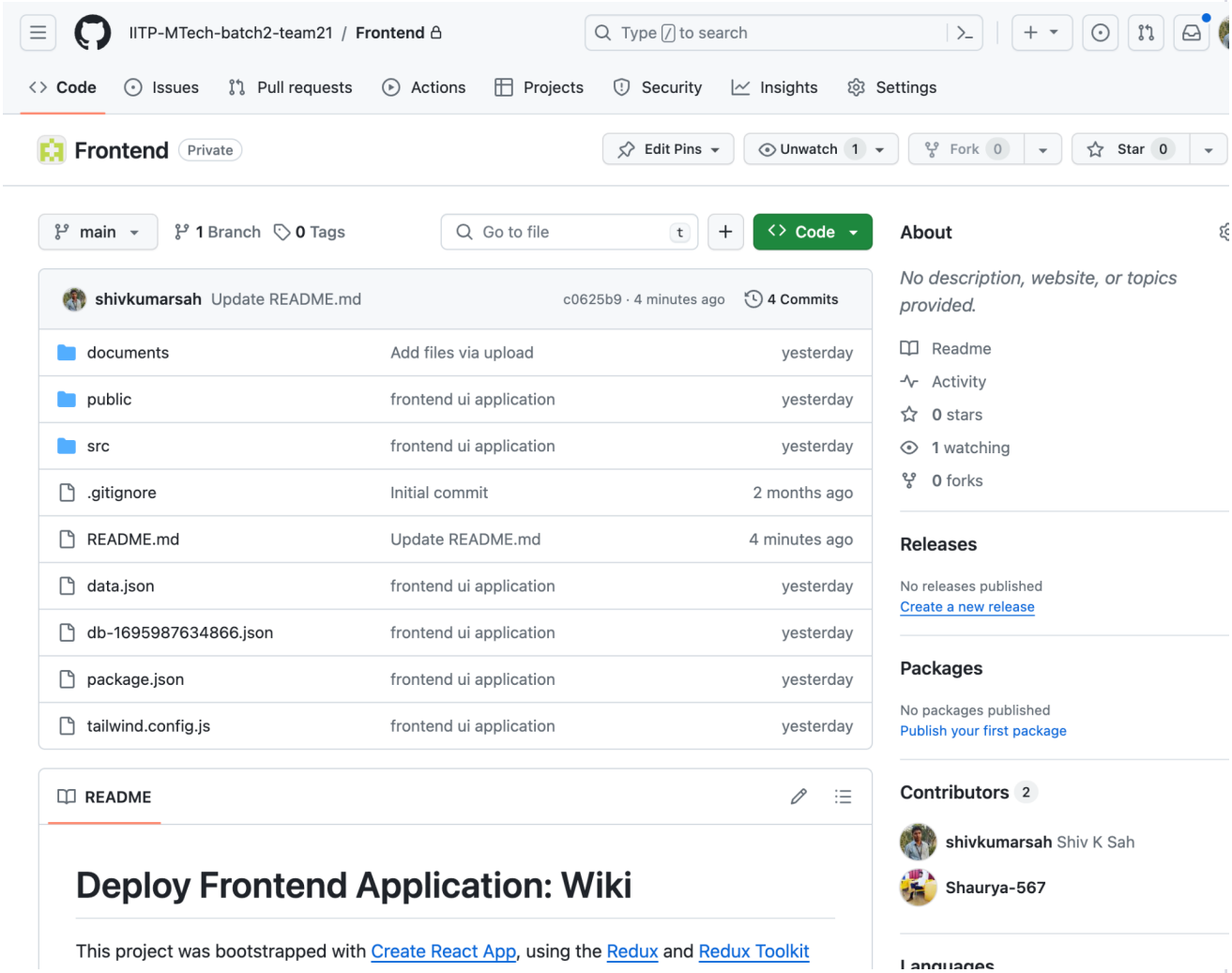
Team: https://github.com/IITP-MTech-batch2-team21



Backend Code: https://github.com/IITP-MTech-batch2-team21/Backend



Frontend: https://github.com/IITP-MTech-batch2-team21/Frontend



5. Future Landscape

The existing e-commerce website offers product suggestions primarily based on product price and features. In our project, we will enhance this by sharing version specifications of a product, including its relationship with predecessor or successor products. This added information will provide consumers with valuable insights, helping them make informed decisions when selecting the right product version.

Benefit/Impact

1. Microservice hosting on the cloud (AWS/GCP) using Containerization, Kubernetes and Docker enables us to Scalability, High Availability, Flexibility, and Cost Efficiency of the e-commerce application to accommodate increased traffic and demand. This ensures that the website can handle traffic surges during peak times, such as sales events or holidays.

2. Faster Time-to-Market: The use of CI/CD pipelines and containerization simplifies and accelerates the deployment process, enabling developers to quickly push updates and new features to the e-commerce site.
3. Improved Security: By following best practices for securing data and implementing role-based access control, the e-commerce site can protect sensitive customer data and maintain compliance with industry regulations.
4. Enhanced Performance: Performance optimization techniques like caching, load balancing, and auto-scaling ensure that the e-commerce site delivers a fast and seamless user experience, even during periods of high traffic.
5. Better User Experience: A user-friendly and responsive web interface ensures that customers can easily navigate and shop on the e-commerce site across various devices and platforms.
6. Seamless Payment and Shipping Integrations: By integrating popular payment gateways and shipping APIs, the e-commerce site can provide a streamlined and convenient shopping experience for customers.

References: eCommerce portals like Amazon and Flipkart