FULL STACK WEB DEVELOPER COURSE PROJECT REPORT

PROJECT TITLE : ShopSmart:Your Digital Grocery Store Experience

Team Member Details

Team ID : LTVIP2025TMID54346

Team leader: Parchuru Venkata Kiran kumar
Team member:
Name: Neelapala Divyasree
Branch: Information Technology
College Registration Number: 23PA1A12E7

Team member:
Name: Nethala SusanaVesly
Branch: Information Technology
College Registration Number: 23PA1A12E9

PROJECT REPORT: Project Idea: ShopSmart: Your Digital Grocery Store Experience

our basic grocery-web app! Our app is designed to provide a seamless online shopping experience for customers, making it convenient for them to explore and purchase products. With user-friendly navigation and intuitive design, our grocery-webapp app allows customers to browse through various categories, view product details, add items to their cart, and securely complete the checkout process. We prioritize user satisfaction and aim to provide a smooth and hassle-free shopping experience.

Key Features:

1. Customer's Access to Visit :

- The application provides a simple and responsive login/signup interface.
- Users can securely create an account or log in to access the weather dashboard.

2. Customers do product ordering and cart management:

- Users can browse products by category and place orders with a single click.
- Ordered items are displayed on a common orders.html page along with total cost calculation.

Used Technologies:

- Node.js and Express.js: These technologies form the core of your web application, managing tasks like handling HTTP requests, directing traffic, and displaying web pages.
- MongoDB Authentication: Used MongoDB Atlas to securely store and verify user credentials during signup and login processes through the backend.
- MongoDB database: Utilized MongoDB Atlas to manage and retrieve data for users, enabling smooth interaction between frontend and backend..

- EJS: Embedded JavaScript
- Password-Hashing: Password hashing is a way to change a user's password into a special code that's hard to read and always the same length. This makes it safer to store passwords and protect people's accounts.

- Body-parse library: It is used to hide the user details in the HTTP requests. It can also used as security library. By this the data will be safe.

Structure of the project:

It consists of various routes and publics signup, login, etc. for handling the customer registration and events management.

Customer Access:

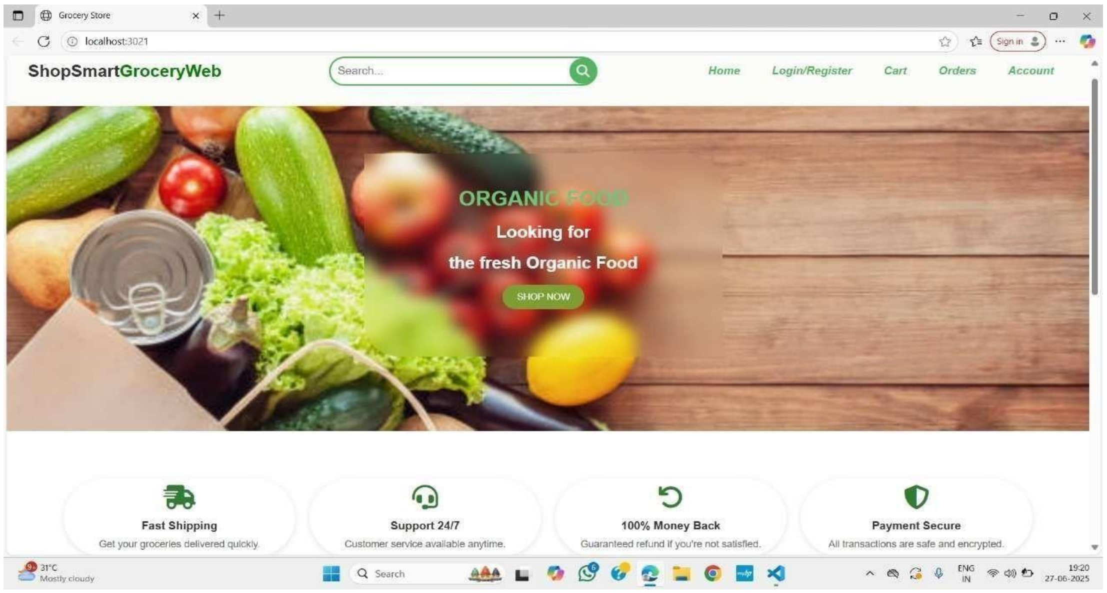
- Customer can Sign up and login with their usernames.
- Email will not be repeated more than one time.
- Passwords will be hashed while storing in the database.

User Authentication:

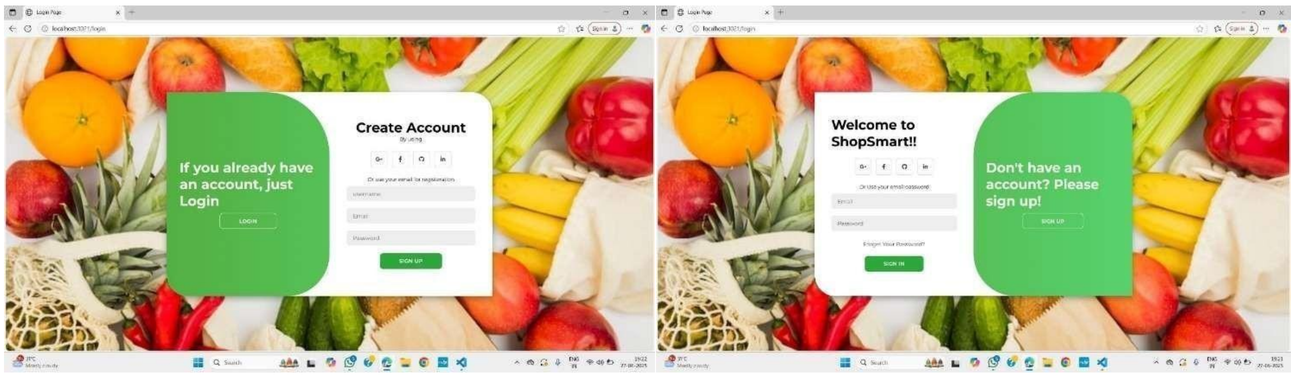
- Users can sign in using their email and password.
- Passwords are securely verified using the hashed password from the database.
- User can have the access to the website.

Images of the webpage: I.

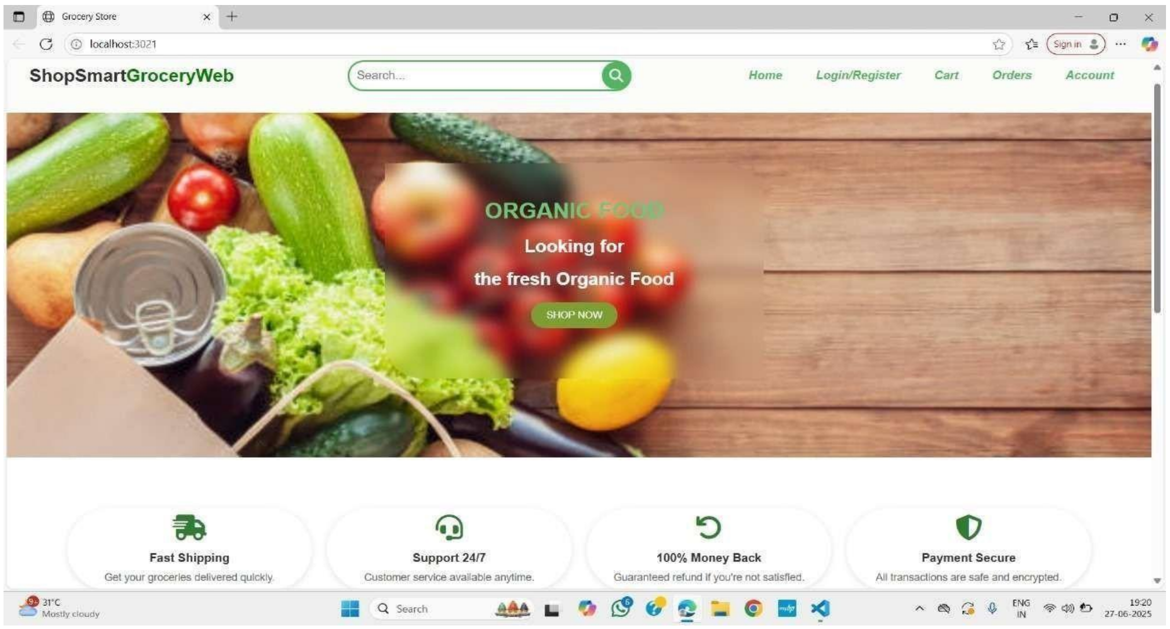
Home page:



II. Login and Signup page:

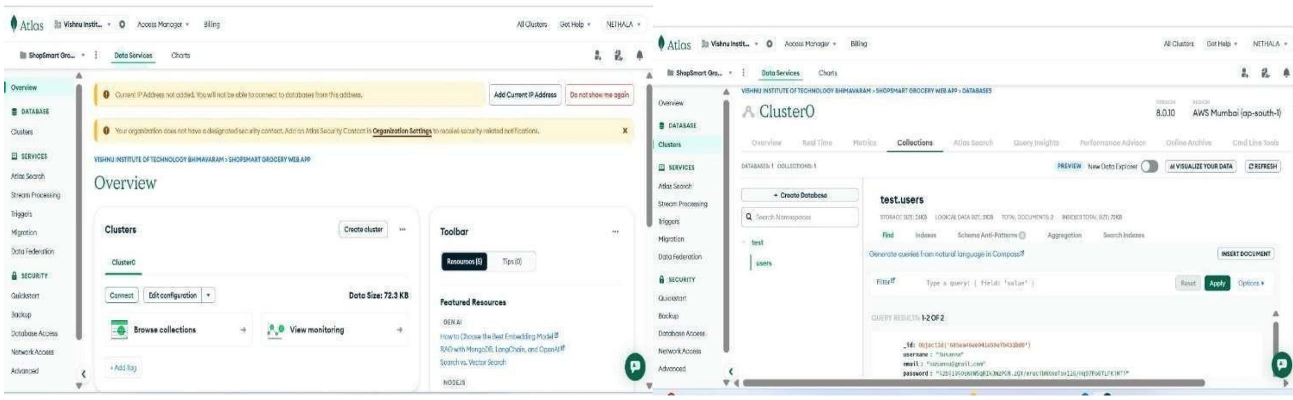


III. Main page:



Database:

- In this I have used the Mongodb Atlas database.
- We created the two types of collections
- One is Submit collection
- In submit collection we used to store the data of users signup details.
- In todo collection we used to store the data of event registration details.



Ideas:

1. Security: Ensure that sensitive data, such as passwords, is stored securely and consider implementing additional security measures, such as user authentication rules in Firebase.

2. Validation: Implement data validation and error handling for user inputs to enhance the robustness of your application.
3. Documentation: Provide clear and concise documentation for users and developers on how to use your application and its API.
4. User Experience: Enhance the user experience by adding features like password reset, better error messages, and validation on the frontend.
5. Frontend Enhancement: Consider improving the user interface and making it more userfriendly.
6. Testing: Implement unit and integration testing to ensure your application works as expected. 7. Scaling: Think about how the application can be scaled to handle a larger number of users and academic records.

Future Ideas:

- It can be implemented further for more features.
- Create a personalized dashboard where users can view past orders, track current orders, and manage account settings.
- Build an admin interface for sellers to add, update, or delete products, view orders, and manage inventory. • Show stock availability status for each product and notify users when items are back in stock.

Contribution from each team member:

1. Member 1: Neelapala Divyasree

- Designed the user interface including login and signup pages, category pages.
- Implemented user authentication and managed user data flow.
- Linked frontend and backend ,performed testing.

2. Member 2: Nethala SusanaVesly

- Designed the user interface, main page with the functionalities.

- Set up the Server using Node.js and Express.js.
- Connected to Mongodb Atlas for storing the user data.
- Implement javascript functionality, test and debug the flow.

References:

Source Code:

https://github.com/Nethala-Susana-vesly/SmartShop_capstone_groupproject.git

Youtube link:

https://www.youtube.com/watch?v=xtZB6FFCGEQ