FARMERS MARKET MANAGEMENT SYSTEM

Amir Modibbo, Haowen Hou, Zihao Kang, Pradeep Kumar Detroja, Xu Wang, Xi Xiong

TEAM 10

Table of Contents

INTRODUCTION

SOFTWARE OVERVIEW

The Farmers' Market web application is designed to connect farmers and customers in a simple and easy-to-use platform. It aims to simplify the process of buying and selling fresh produce by connecting local farmers with consumers who are looking for fresh and healthy food. The Farmers' Market web application is designed to be user-friendly, and it offers a range of features that make it easy for farmers to list their produce and for customers to place orders.
The Farmers' Market web application is built on , Node.js, Express,Vue.js , and MySQL. It is a full-stack web application that offers a range of features, including user registration and login, product listing, product search, shopping cart, and payment processing. Farmers can easily create accounts and list their products, while customers can search for products, place orders, and make payments using a secure payment gateway.

BENEFITS

The Farmers' Market web application offers a range of benefits for farmers and customers, including:

1. Increased visibility: The application allows farmers to list their products and reach a wider audience. This makes it easier for farmers to sell their products and increase their revenue.
2. Simplified ordering: Customers can easily search for products, add them to their shopping cart, and place orders. This makes it easy for customers to buy fresh produce without having to leave their homes.
3. Secure payments: The application uses a secure payment gateway that ensures that all transactions are safe and secure. This gives customers peace of mind when making purchases.
4. Improved efficiency: The application is designed to be fast and efficient, which means that farmers and customers can complete transactions quickly and easily.
5. Better communication: The application allows farmers and customers to communicate easily and quickly. This makes it easy for farmers to answer customer questions and for customers to get the information they need.

PROJECT SCOPE AND OBJECTIVES

PROJECT SCOPE

Farmers' Market is a web-based software application designed to provide a convenient platform for farmers and customers to connect with each other. The software will enable farmers to create and manage their product listings, while customers will be able to browse and purchase products from their favourite farmers. Additionally, the software will facilitate communication between farmers and customers by allowing them to exchange messages and reviews.

The scope of the project includes developing a user-friendly interface that will allow customers and farmers to easily navigate the software. The software will be accessible via web browsers on desktop. The software will be designed to provide secure and reliable transactions between customers and farmers. The software will also include a user authentication system to ensure that only authorized users can access the platform.

OBJECTIVES: The objectives of the Farmers' Market software are as follows:

1. To provide a convenient platform for farmers to showcase their products and reach a wider audience.
2. To provide customers with easy access to fresh, locally sourced produce and other products.
3. To facilitate communication between farmers and customers, allowing for feedback and the building of a community.
4. To provide a secure and reliable platform for transactions between customers and farmers.
5. To provide a user-friendly interface that can be easily navigated by users with varying levels of technical expertise.
6. To ensure that the software is accessible via web browsers on desktop.
7. To provide a user authentication system to ensure that only authorized users can access the platform.

By achieving these objectives, the Farmers' Market software will help farmers to expand their businesses by reaching a wider audience, while also providing customers with easy access to fresh, locally sourced produce. Additionally, the software will help to build a community of farmers and customers, facilitating communication and feedback.

TEAM NAME AND LIST OF MEMBERS

NAME: TEAM-10

MEMBERS:

1. Amir Modibbo
2. Haowen Hou
3. Zihao Kang
4. Pradeep Kumar Detroja
5. Xu Wang
6. Xi Xiong

PRODUCT BACKLOG

1. As a farmer, I want to list my produce for sale, so that I can reach a wider customer base.
2. As a user, I want to register and log in, so that I can purchase food items from the application.
3. As a buyer, I want to purchase items from multiple farmers at the same time, so that I can enjoy a variety of fresh produce in a single transaction.
4. As a buyer, I want to search for food items based on pre-defined categories, so that I can easily find the items I need.
5. As a farmer, I want to view the details of purchases made and receive an email notification for each purchase, so that I can keep track of my sales.
6. As a first-time user, I want to validate my email address upon login, so that my account is secure and verified.
7. As a farmer, I want to bulk upload items for sale via a CSV or excel file, so that I can list multiple items efficiently.
8. As a farmer, I want my items that have run out to be automatically removed from the public page but be able to update the items left, so that my inventory is accurate and up to date.

9. As a buyer, I want to use PayPal or Stripe for payment, so that I can safely and conveniently complete my purchase.
10. As a user, I want the application to be secure and not store my credit or debit card details, so that my financial information is protected.
11. As a user, I want the application to use sandbox environments of PayPal and Stripe, so that my sensitive data is handled in a safe and secure manner.
12. As a buyer, I want to view the same item from multiple farmers with the cheapest item listed first, so that I can make cost-effective purchases.
13. As a user, I want to add items to a wish list, so that I can save them for future purchases.
14. As a site administrator, I want a dashboard where I can see a snapshot and graph of relevant metrics over a defined period, so that I can monitor and manage the application effectively.
15. As a user, I want the application to generate PDF reports for the various data stored, so that I can have a tangible record of my activities on the platform.

ANALYSIS & DESIGN

SYSTEM ARCHITECTURE

1. Frontend Layer
   - Framework: Vue.js for building a Single Page Application.
   - UI Libraries: Material-UI for a responsive and consistent User interface.
   - Authentication: Jason Web Tokens for handling user authentication and authorization.

2. Backend Layer: Language & Framework: A server with Node.js and express for building a RESTful API which include implementation of authentication/registration, managing database queries/transactions and services such as adding products, retrieving products for customers etc.

3. Database Layer Database: MySQL for storing user, farmer and product data.

4. Deployment and infrastructure: GitLab for development and deployment process

5. Code Documentation: Maintain clear and concise documentation for front end and backend.
6. Security

- Secure Storage: Use environment variables and secret management tool to store sensitive information.
- Input Validation & Sanitization: Prevent security vulnerabilities (e.g., SQL injection, XSS).

USE CASES

Use Case 1: Register as a user.
Actors: Customer, Farmer
Entry Conditions: User is not registered or logged into the system.
Normal Flow:
User selects the "Register" option. The User provides their name, email, and password. The User selects either "Customer" or "Farmer" role. If the authentication pass successfully, the system validates the input and creates the account.
Exceptional Flows:
User enters an email that is already registered. System shows an error message.
User enters invalid input, such as an incorrect email format or a too short password. System shows an error message.

Use Case 2: Login to the system.
Actors: Customer, Farmer
Entry Conditions: User has a registered account and is not logged in.
Normal Flow:
User enters their email and password; the system should first validate the input to ensure that the email and password are valid and match the user's account. If the validation is successful, the system can log the user in and redirect them to the appropriate dashboard based on their user type (Customer or Farmer).
Exceptional Flows:
User enters an incorrect email or password. System shows an error message.
Use Case 3: Browse products.
Actors: Customer
Entry Conditions: User is logged in as a customer.
Normal Flow:
User navigates to the "Products" section and system displays the list of available products with their basic information.
Exceptional Flows: None

Use Case 4: Search for a product.
Actors: Customer
Entry Conditions: User is logged in as a customer and browsing products.
Normal Flow:
User enters the name of the product in the search bar. System filters the name of the product based on the search criteria. System displays the matching products.
Exceptional Flows:
No products match the search criteria.

Use Case 5: View product details.
Actors: Customer
Entry Conditions: User is logged in as a customer and browsing products.
Normal Flow:
User selects a product from the product list and system displays detailed information about the product, including its description, price, and quantity.
Exceptional Flows: None

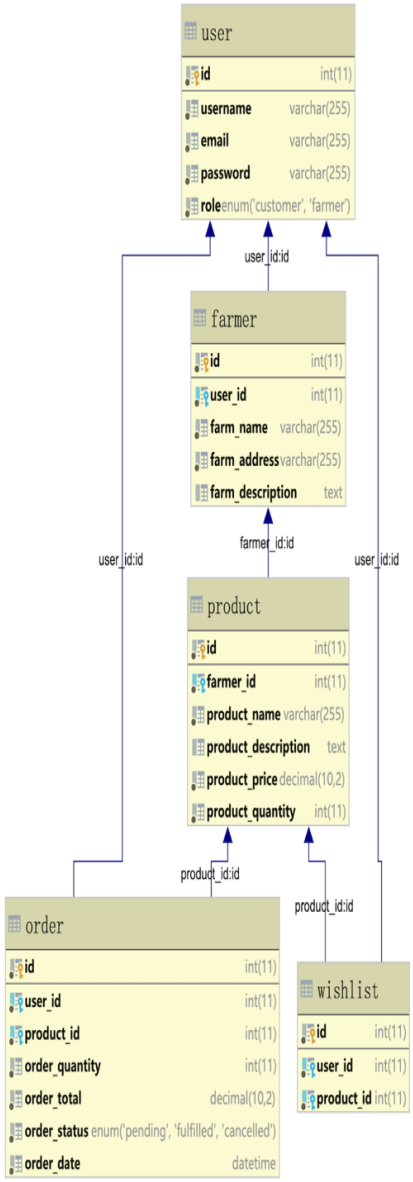
Use Case 6: Add a product to the cart.
Actors: Customer
Entry Conditions: User is logged in as a customer and viewing product details.
Normal Flow:
User selects the desired quantity and clicks the "Add to Cart" button.
System adds the product to the cart and updates the cart's total price.

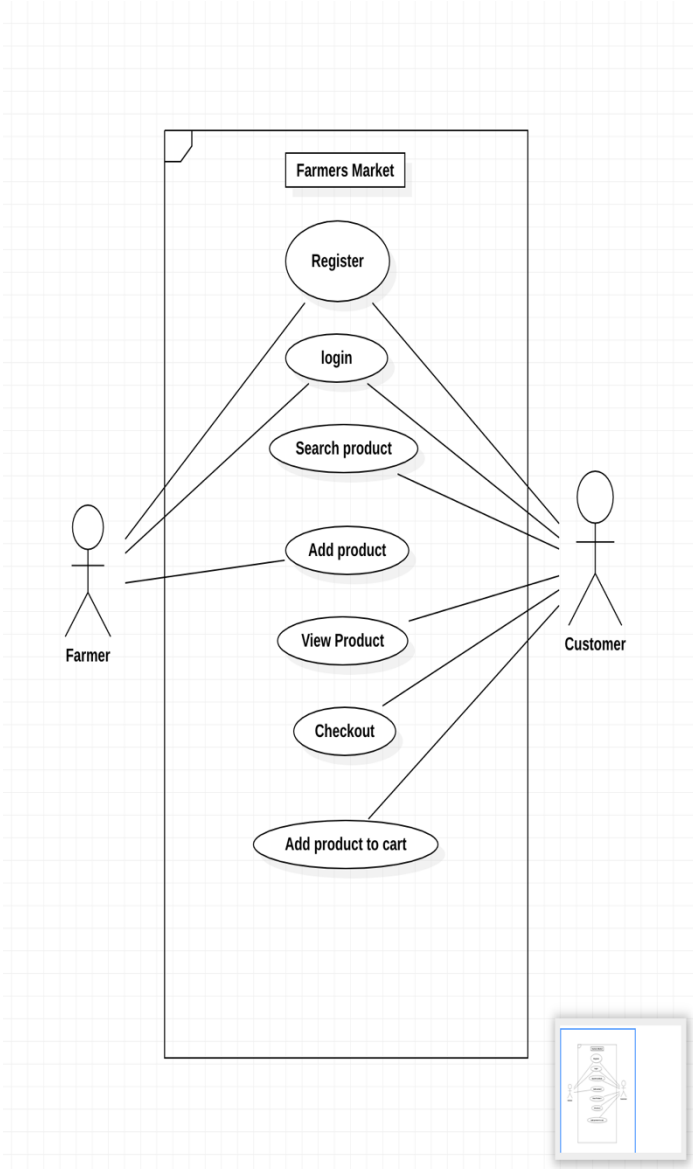
Exceptional Flows: None

Use Case 7: Update the cart.
Actors: Customer
Entry Conditions: User is logged in as a customer and has items in their cart.
Normal Flow:
User navigates to the "Cart" section.
User updates the quantity of a product or removes a product from the cart.
System updates the cart's total price.
Exceptional Flows: None

Use Case 8: Checkout
Actors: Customer
Entry Conditions: User is logged in as a customer and has items in their cart.
Normal Flow:
User navigates to the "Checkout" section.
User provides their shipping and billing information.
System validates the input and processes the payment.
System confirms the successful order.
Exceptional Flows: None

Use Case 9: Bulk Upload
Actors:
Entry Conditions: Farmer is registered, logged into the system and has a list of products ready for upload.
Normal Flow:
Farmer navigates to the "Import" section.
Farmer downloads the XLS template provided by the system.
Farmer fills the template with the product details and saves the file.
Farmer returns to the "Bulk Upload Products" section and uploads the filled XLS file.
System validates the uploaded XLS file and adds the products to the Farmer's product list.
System confirms the successful upload and provides a summary of the uploaded products.
Exceptional Flows:
Farmer uploads a CSV file in the wrong format or contains invalid data. System shows an error message.





Initial E. R. Diagram **UML Diagram**

TEAM MANAGEMENT & COMMUNICATION

PLANNING PHASE

Team Allocation Decision
The team was divided into two groups:
Front-end Team and backend Team

Back-end Team

1. Amir Modibbo
2. Haowen Hou

3. Pradeep Kumar Detroja

Front-end Team

1. Zihao Kang

2. Xu Wang (*)

3. Xi Xiong

(*) Xu Wang moved to the Back-end team at the end of the first sprint.

Agreement on the Technology Stack for the Project

Front-end: Vue.js
Back-end: Node.js and Express
Database: MySQL

Task completed in planning phase

1. Zihao Kang and Xi Xiong designed the Systems logo.
2. Haowen Hou and Xu Wang designed the Systems Prototype.
3. Amir Modibbo and Pradeep Detroja worked on the Product Backlog.
4. Git

SPRINT 1:

Client priorities

- Changes in the proposed website logo
- Conveyed the desired name for the platform- Farmers market.
- Conveyed his desire for the colour scheme to keep for the software.

1. Zihao Kang completed the development of the Vue front-end authentication page and registration page.
2. Amir Modibbo completed the development of the database architecture.
3. Pradeep Detroja and Amir Modibbo worked together on the development of the login and registration page.**Problem Encountered:**

Problems Encountered: The team faced issues with the integration of the front-end and back-end authentication and registration pages at the end of the first sprint which delayed the teams progress. These problems were resolved subsequently by Pradeep Kumar Detroja in the second sprint.

SPRINT 2:

Client priorities

- Functional Farmers page with single, bulk upload features
- Payment Integration
- Customers page- Market, where a customer can view the available farm produces.

Completed task

1. Xi Xiong started the development customer's front-end page.
2. Farmers page
3. Xu Wang and Haowen Hou started the development of the farmers and customers page back-end.

Problems Encountered:
The conflict between the current software and the previously developed version from the earlier iteration resulted in significant challenges for the project. These discrepancies stemmed from differences in features, design patterns, and architectural choices, ultimately leading to changes in the entire systems hierarchy. To address these issues, the team had to reassess the project's structure, paying close attention to dependencies and modules. By carefully analysing and understanding the discrepancies between the two versions, they were able to reconcile the differences and establish a coherent, unified system hierarchy. Consequentially, the team pivoted in its approach from a single server architecture to a dual server architecture. One for the Vue frontend, and one for the express Backend.

SPRINT 3:

Completed task.

1. The whole team collaborated closely and completed the development of the farmers, customers and admin page.
2. Team worked in completing the documentation and video walkthrough of the system.
3. Team worked towards the testing and assurance of the built platform.
4. Individual members worked towards making sure the code produced by them is of industry standards, by refactoring the codebase wherever necessary.

Problem Encountered:

During this sprint, we made a lot of progress, adding new features and functionalities. However, as our code base grew, we faced some challenges due to the team's inexperience with GitLab, our version control system. This resulted in some issues with merging code. Additionally, language differences within the team made it hard to communicate about what parts of the software each person was working on, what they had done so far, and where they were stuck.
To solve these problems, we started using an issue-based approach to development. Each team member created an issue on GitLab for the feature they were working on. They then created their own branches and worked on these until they had completed the feature, at which point they merged their changes back into the main branch. This system ensured everyone knew what others were working on, and using separate branches helped to reduce conflicts and keep the main branch clean and bug-free. This also allowed the team to assist each other when needed.

PLANNED & COMPLETED FEATURES

PLANNED FEATURES

Sprint 1 backlog

1. As a farmer, I want to list my produce for sale, so that I can reach a wider customer base.
2. As a user, I want to register and log in, so that I can purchase food items from the application.
3. As a first-time user, I want to validate my email address upon login, so that my account is secure and verified.
4. As a user, I want the application to be secure and not store my credit or debit card details, so that my financial information is protected.
5. As a user, I want the application to use sandbox environments of PayPal and Stripe, so that my sensitive data is handled in a safe and secure manner.

Sprint 2 backlog

1. As a buyer, I want to purchase items from multiple farmers at the same time, so that I can enjoy a variety of fresh produce in a single transaction.
2. As a buyer, I want to search for food items based on pre-defined categories, so that I can easily find the items I need.
3. As a farmer, I want to bulk upload items for sale via a CSV or excel file, so that I can list multiple items efficiently.
4. As a buyer, I want to use PayPal or Stripe for payment, so that I can safely and conveniently complete my purchase.
5. As a buyer, I want to view the same item from multiple farmers with the cheapest item listed first, so that I can make cost-effective purchases.

Sprint 3 backlog

1. As a farmer, I want to view the details of purchases made and receive an email notification for each purchase, so that I can keep track of my sales.
2. As a farmer, I want my items that have run out to be automatically removed from the public page but be able to update the items left, so that my inventory is accurate and up-to-date.
3. As a user, I want to add items to a wish list, so that I can save them for future purchases.
4. As a site administrator, I want a dashboard where I can see a snapshot and graph of relevant metrics over a defined period, so that I can monitor and manage the application effectively.
5. As a user, I want the application to generate PDF reports for the various data stored, so that I can have a tangible record of my activities on the platform.

COMPLETED FEATURES

Sprint 1 backlog

1. As a user, I want to register and log in, so that I can purchase food items from the application.
2. As a first-time user, I want to validate my email address upon login, so that my account is secure and verified.
3. As a user, I want the application to be secure and not store my credit or debit card details, so that my financial information is protected.

Sprint 2 backlog

1. As a farmer, I want to list my produce for sale, so that I can reach a wider customer base.
2. As a buyer, I want to purchase items from multiple farmers at the same time, so that I can enjoy a variety of fresh produce in a single transaction.
3. As a buyer, I want to search for food items based on pre-defined categories, so that I can easily find the items I need.
4. As a farmer, I want to bulk upload items for sale via a CSV or excel file, so that I can list multiple items efficiently.
5. As a buyer, I want to view the same item from multiple farmers with the cheapest item listed first, so that I can make cost-effective purchases.

Sprint 3 backlog

1. As a farmer, I want to view the details of purchases made and receive an email notification for each purchase, so that I can keep track of my sales.
2. As a farmer, I want my items that have run out to be automatically removed from the public page but be able to update the items left, so that my inventory is accurate and up-to-date.
3. As a user, I want to add items to a wish list, so that I can save them for future purchases.
4. As a site administrator, I want a dashboard where I can see a snapshot and graph of relevant metrics over a defined period, so that I can monitor and manage the application effectively.
5. As a user, I want the application to generate PDF reports for the various data stored, so that I can have a tangible record of my activities on the platform.

UNCOMPLETED FEATURES

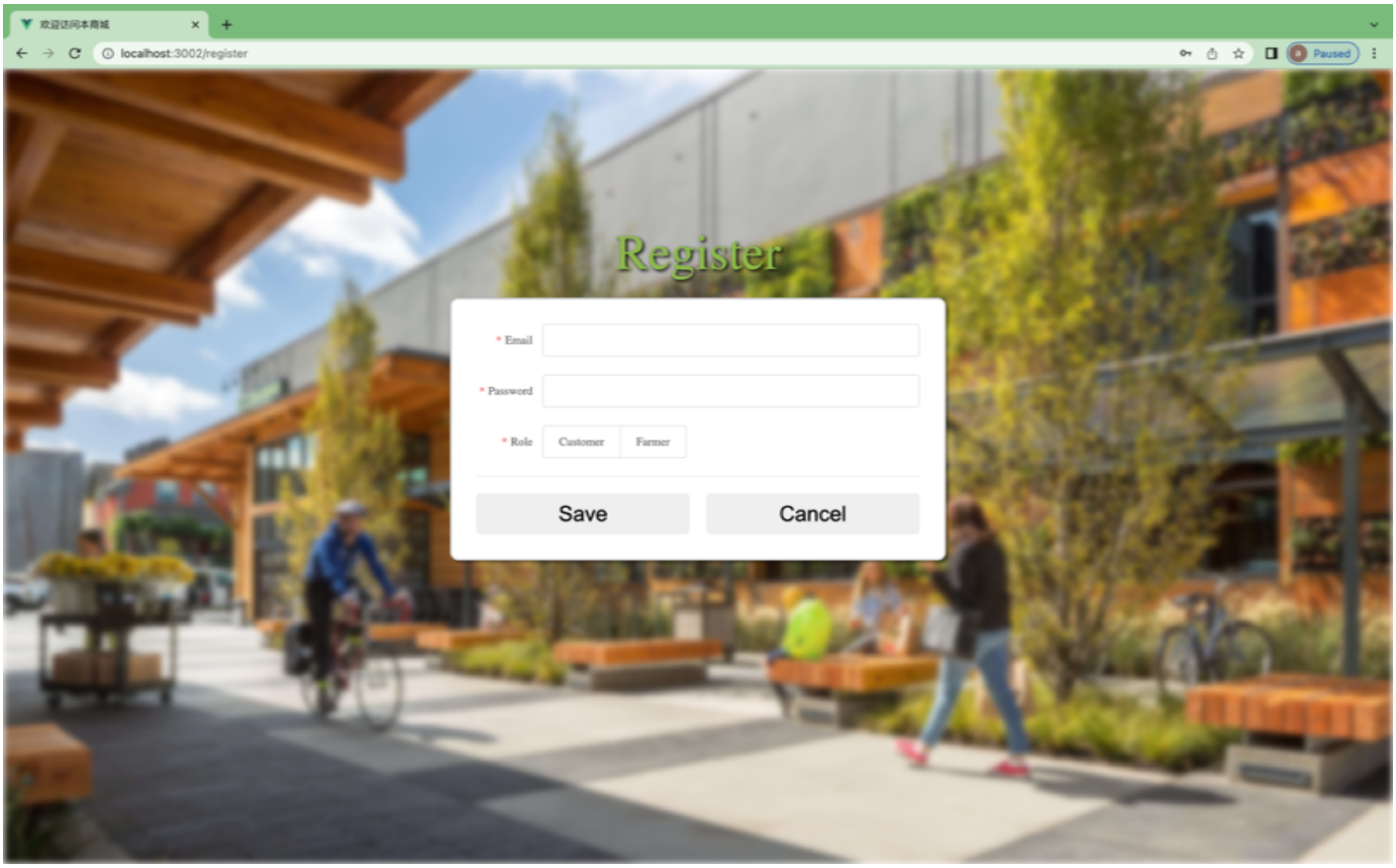
1. As a user, I want the application to use sandbox environments of PayPal and Stripe, so that my sensitive data is handled in a safe and secure manner.
2. As a buyer, I want to use PayPal or Stripe for payment, so that I can safely and conveniently complete my purchase.

Reason

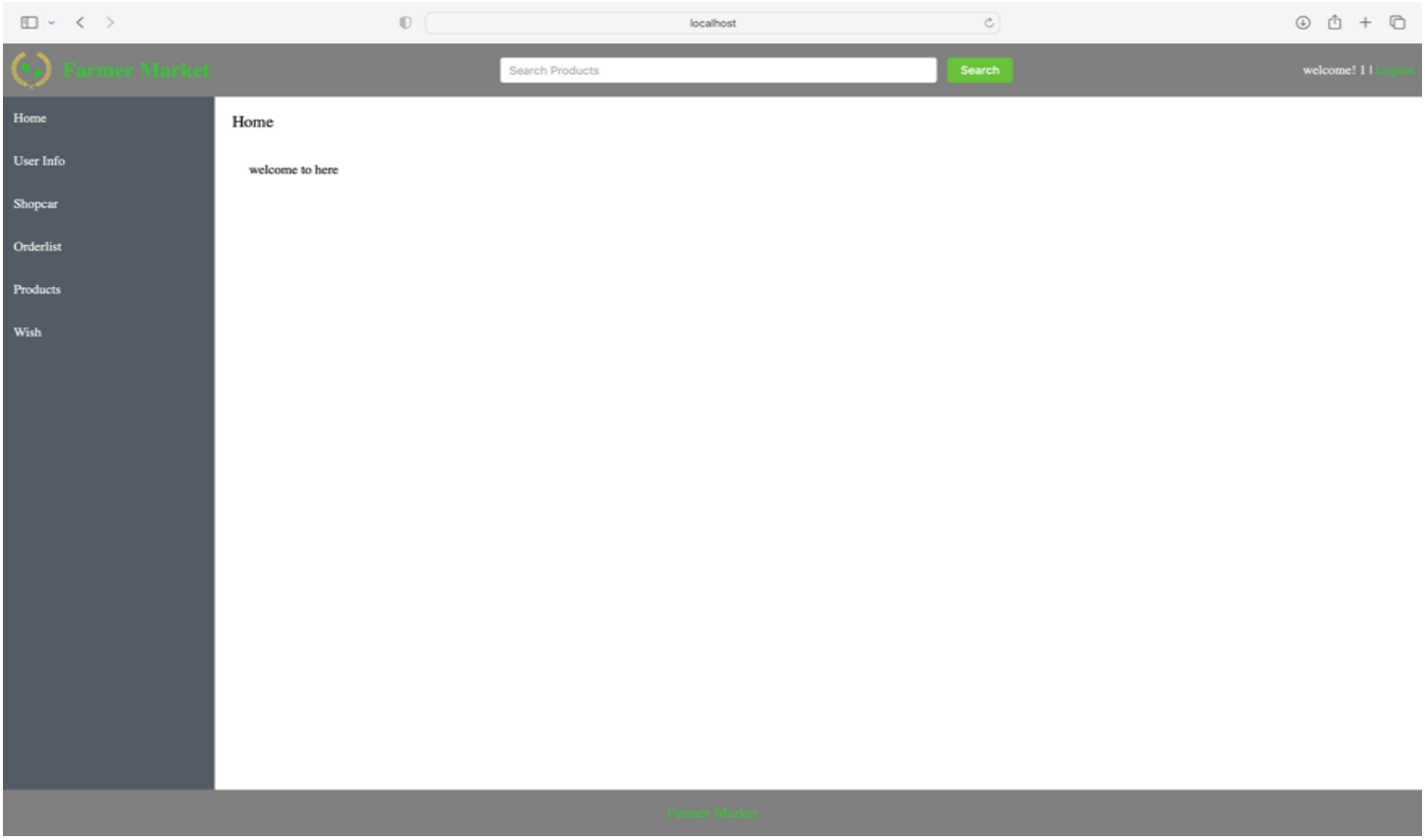
The team unanimously decided to delegate the PayPal payment integration task to the end of Sprint 3. However, unfortunately, due to unexpected time constraints, we were unable to complete this feature within the allocated timeframe. Despite our diligent planning and prioritization, circumstances beyond our control led to this delay.

SCREENSHOTS OF RELEVANT PAGES

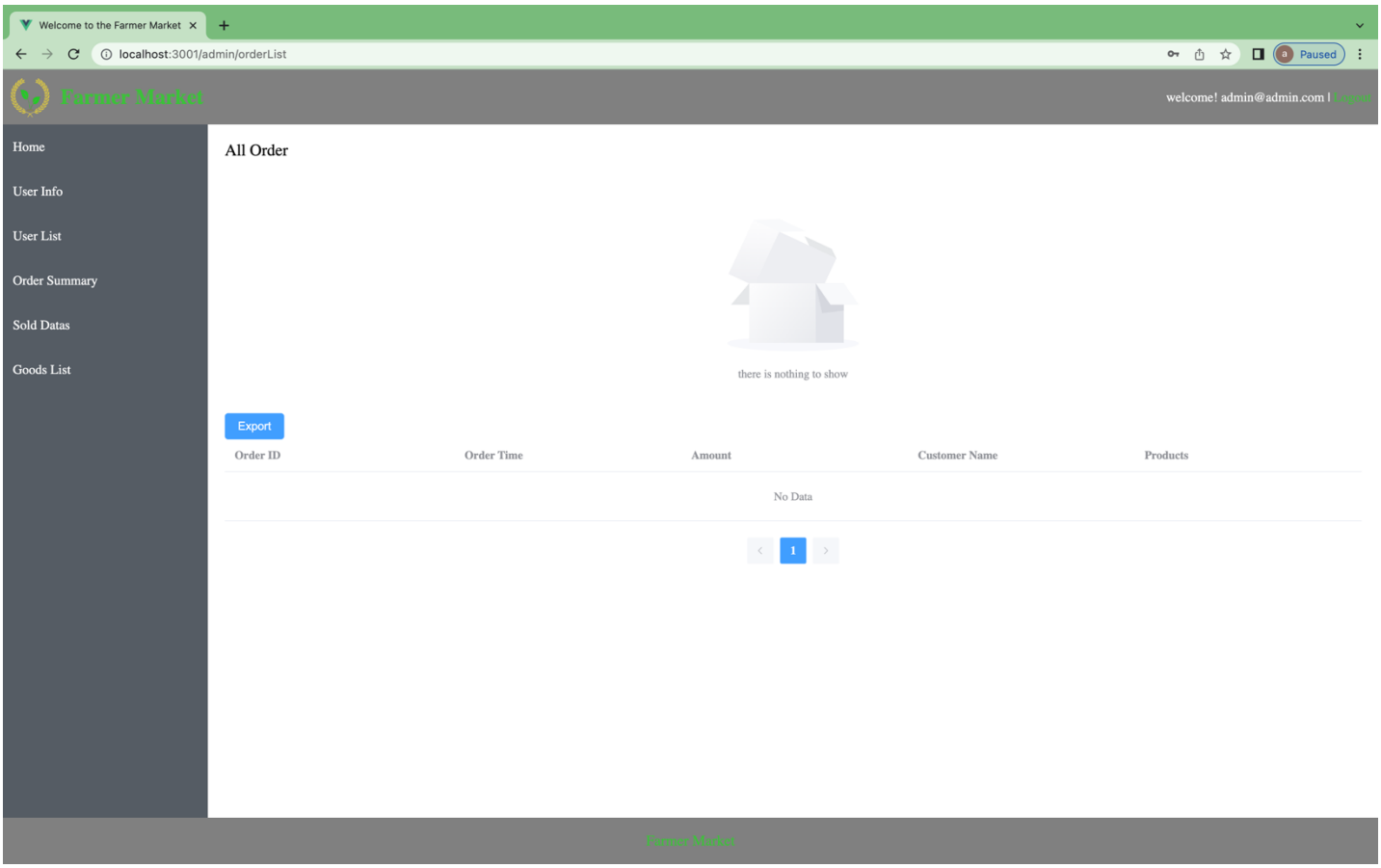
Home page and login



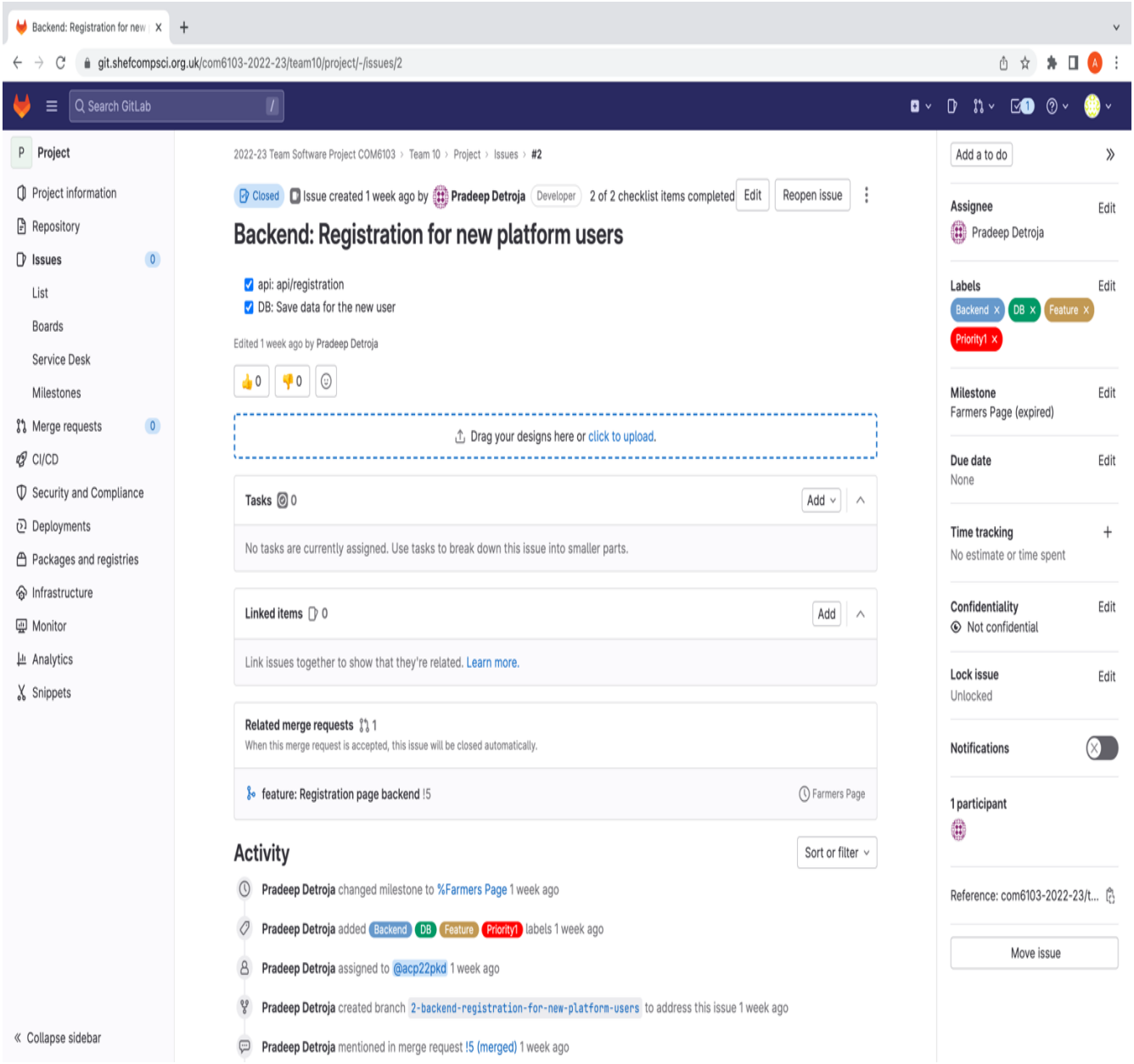
Registration Page



Customers Page



Admin page



Sample of GitLab repository Issues page

CONCLUSION

CHALLENGES

Throughout the development of the project, our team faced several challenges that impacted the overall progress and efficiency. These challenges can be broadly categorized into the following points:

1. GitLab issues: Our team experienced some difficulties using GitLab for version control and collaboration. Merge conflicts, branching issues, and lack of prior experience with Git led to confusion and delays which led the team collaborate closely and work offline. To resolve these problems, we held training sessions on Git best practices and sought help from experienced team members or online resources in the third sprint of the project.
2. Clients Requirement: The team was unable to meet the client's requirement at the end of the first sprint. To resolve this, we decided to work overtime during the sprint to expedite the work and satisfy the client's requirements.

3. Deciding on the technology stack: The team had differing opinions on which technology stack to use. They resolved this issue through a voting process, where the majority's choice was ultimately selected.
4. Difficulty in integrating front end and back end: Integrating the front end and back end of the project proved to be challenging, as each team member had their own approach to solving problems. This led to inconsistencies when attempting to merge the user interface with the server-side code between the front end and back-end components.
5. Splitting the work: Dividing tasks and responsibilities among team members was not an easy feat. It was important to consider each member's skill set and expertise to allocate tasks effectively. In some cases, the workload was unevenly distributed, which resulted in delays and increased pressure on certain team members. To mitigate this issue, regular project management and check-ins were essential to ensure a balanced distribution of work.
6. Team collaboration: Effective collaboration is the backbone of any successful project, and our team faced some difficulties in this area. Communication gaps, different working styles, and developers other commitments differences led to misunderstandings and conflicts. To overcome these challenges, we established regular meetings, used collaboration tools, and tried to be more understanding of each other's perspectives.
7. Understanding the programming languages: Some team members had limited experience with the programming languages used in the project. This led to a steep learning curve and slowed down the development process. To tackle this challenge, we organized knowledge-sharing sessions, provided learning resources, and encouraged team members to ask for help when needed.
8. Team meetings and time clashes: Scheduling team meetings was challenging due to personal commitments, and other obligations. This made it difficult for everyone to attend meetings, leading to a lack of alignment and cohesion within the team. To address this issue, we tried to schedule meetings at mutually convenient times and used asynchronous communication methods such as email and discord to keep everyone informed.
9. Developers' language communication barrier: As our team consisted of members from diverse linguistic backgrounds, communication barriers sometimes hindered collaboration and understanding. To overcome this challenge, we encouraged the use of simple and clear language to convey complex ideas.

   In conclusion, while our team faced numerous challenges during the project, we were able to overcome them through open communication, effective collaboration, and continuous learning. By addressing these issues, we were able to improve our processes and work more efficiently as a team.

LEARNING

During the development of the Farmers Market web application project, the team gained valuable experience and skills in various aspects of software development. These learnings not only enhanced their technical knowledge but also improved their ability to work in a professional environment.

1. End-to-end full-stack project: The team members learned to build a complete web application from scratch, involving frontend, backend, and database components. This experience gave them a comprehensive understanding of how different technologies and layers of an application interact with each other. They gained hands-on experience with frontend frameworks like Vue.js, backend technologies like Node.js, and databases like MySQL. This full-stack knowledge is vital for today's software developers, as it allows them to contribute to various aspects of a project and fosters a deeper understanding of the overall architecture.
2. Debugging large codebases: As the project progressed, the codebase grew larger and more complex. The team had to deal with various bugs and issues that arose during development. Debugging a large codebase requires systematic thinking and attention to detail, as well as the ability to understand and analyse the flow of the code. The team members honed their debugging skills, which are crucial in any software development project, as they ensure the final product works as expected.
3. Frontend and backend task splitting: The team learned how to efficiently split tasks between frontend and backend developers. This approach is commonly used in the industry, as it allows developers to focus on their areas of expertise, leading to more effective collaboration and faster development. By understanding

the best practices for dividing tasks and maintaining clear communication, the team members were able to work together more seamlessly and deliver a high-quality product.

4. Agile methods: The team adopted Agile methodologies, such as Scrum, to manage their project. Agile methodologies emphasize iterative development, collaboration, and flexibility in adapting to changing requirements. The team conducted regular stand-ups, sprint planning, and meetings to ensure that the project remained on track and to continuously improve their processes. This experience gave them a solid understanding of Agile principles, which are widely used in the software development industry.
5. Git and version control: The team learned to use Git, a popular version control system, to manage their codebase. Git allows developers to track changes in the code, collaborate more effectively, and maintain a clean and organized codebase. They learned to create branches for new features, merge changes, and resolve conflicts. Mastering Git is an essential skill for modern software developers, as it is widely used in the industry.
6. Teamwork: The project provided an excellent opportunity for the team members to learn the importance of effective teamwork. They worked together to solve problems, brainstorm ideas, and support each other's growth. Through open communication and a willingness to learn from each other, they fostered a strong team dynamic that contributed to the project's success.
7. Cultural understanding and diversity: The team members came from diverse backgrounds, which enriched their collaboration and broadened their perspectives. They learned to appreciate and respect each other's cultures, values, and work styles, which helped them create an inclusive and supportive work environment. This experience demonstrated the value of diversity in a team and the importance of embracing different perspectives to drive innovation.
8. Client interaction: The team members had interactions with the client, giving them real-world experience in managing client expectations and delivering a product that met their needs. They learned to communicate effectively, present their ideas clearly, and respond to feedback. These skills are crucial in any professional setting, as they ensure that the end product aligns with the client's vision and requirements.
9. Advisor interaction: Throughout the project, the team consulted with advisors who provided guidance and expertise in various aspects of software development. These interactions helped the team members refine their ideas, address technical challenges, and develop best practices. By actively seeking advice from experienced professionals, they were able to learn valuable lessons that went beyond textbook knowledge.
10. Time management and prioritization: The team learned to effectively manage their time and prioritize tasks to meet deadlines. They developed a clear understanding of the project's goals and identified which tasks were critical to the project's success. By setting realistic expectations and using tools like task boards and schedules, they ensured that work was completed on time and in an organized manner.
11. Continuous learning and skill development: The project provided ample opportunities for the team members to learn new technologies, tools, and techniques, as well as to improve their existing skills.
12. Documentation and code readability: The team learned the importance of creating clear and concise documentation, both for their code and for the overall project. They practiced writing well-structured and easily understandable code, as well as documenting their processes and design decisions. Good documentation and code readability make it easier for other developers to understand and contribute to the project, and they are crucial for long-term maintainability.

In conclusion, the Farmers Market web application project provided the team with invaluable experience in multiple areas of software development. The lessons they learned from this project will serve them well in their future careers, as they now possess a strong foundation in both technical and interpersonal skills. By embracing challenges, seeking guidance, and continuously learning, they were able to grow as developers and deliver a high-quality product.

APPENDIX

USER GUIDE

The Farmers Market web application is an easy-to-use platform developed using Vue.js, Node.js, and SQL, designed to connect farmers with customers. This user guide will help you understand how to navigate the application, create an account, manage products, and make purchases. Let's get started!

1. Registration and Login:

To begin using the Farmers Market application, you'll need to create an account. On the homepage, click on the 'Sign Up' button, and you'll be redirected to the registration page. Fill in your name, email address, password. Once you have registered, you can log in using your email and password.

2. Farmer's Dashboard:

As a farmer, you can manage your products and view customer orders through the farmer's dashboard. To access the dashboard, log in and click on the appropriate navigation menu.

2.1. Adding a Product:

To add a new product, click on the 'Add new' button on the 'Trade Manager' page. Fill in the required details, such as Trade name, description, price, stock and image. Once you're done, click the 'Save' button to add the product to your inventory.

2.2. Editing a Product:

To edit an existing product, click on the 'Modify' button next to the product on the 'Trade Manager' page. Update the necessary information and click the 'Save' button to save your changes.

View User Information

To view the farmers information, click on the 'User Info' button on the 'Trade Manager' page.

View Order List

To view order List, click on the 'Order List' button on the 'Trade Manager' page.

2.3.Bulk Upload Feature:

Generate Template File:

Log in and Click 'Import' then 'Generate Template.' download the generated excel file.

Open the excel file and fill in product details. Save the updated file.

Go back to 'Upload' on the dashboard. Click 'Import' then 'Upload Template.'

Choose and upload the filled excel file. Wait for the system confirmation of a successful upload.

3. Customer Dashboard:

As a customer, you can browse products, add items to your shopping cart, and complete purchases. To access the customer dashboard, log in and click on the 'Browse Products' link in the navigation menu.

3.1. Browsing Products:

On the 'Products' page, you can view all available products. You can filter products by searching for a specific product using the search bar.

3.2. Adding Products to the Shopping Cart:

To add a product to your shopping cart, click on the 'Add to Cart' button on the product listing. The product will be added to your cart, and you can view the contents of your cart by clicking the 'Cart' link in the navigation menu.

3.3. Completing a Purchase:

To complete a purchase, click on the 'Shopping Cart' link in the navigation menu and review your order. If you're satisfied with the contents of your cart, click the 'Checkout' button, and fill in your shipping and payment information. Once your order is complete.

3.4. Adding Products to the Wishlist:

To add a product to your wish list, click on the 'Add to Wishlist' button on the shopping cart page. The product will be added to your wish list, and you can view the contents of your wish list by clicking the 'Wishlist' link in the navigation menu.

4.Admin Dashboard
As an admin, you have a comprehensive view and control over all activities on the Farmers Market application. You can view users, summarize orders, check sold data, and manage the goods list. To access the Admin Dashboard, log in with your admin credentials and click on the 'Admin Dashboard' link in the navigation menu.
4.0. Admin login
The admin can login with
email : admin@admin.com
password: admin
4.1. Viewing User List:
In the Admin Dashboard, click on the 'User List' button.
You will be redirected to the page with the list of all registered users, including their name, email, role, and other necessary information.
4.2. Viewing Order Summary:
From the Admin Dashboard, click on the 'Order Summary' button.
Here, you can view all customer orders, their status, total cost, and other related details.
4.3. Checking Sold Data:
In the Admin Dashboard, select the 'Sold Data' button.
This page displays all the sold items, their quantity, selling price, and the total revenue generated.
4.4. Managing Goods List:
In the Admin Dashboard, click on the 'Goods List' button.
This page provides a list of all available and sold-out products. You can add, modify, or delete products as needed.
Remember, as an admin, you have the responsibility to manage and maintain the smooth operation of the Farmers Market application. Always ensure to keep data up-to-date and resolve any issues promptly.

The Farmers Market web application is a user-friendly platform that makes it easy for farmers and customers to connect and engage in commerce. By following this user guide, you'll be able to create an account, manage your inventory, and complete transactions with ease. Happy shopping!

SETUP GUIDE

Step-by-step guide on cloning the project from GitLab, setting up, and running the Vue.js frontend and Express.js backend servers.

1. Install Node.js and npm (npm is included with Node.js).
2. Install WebStorm.
3. Install MySQL.
4. Access to the GitLab repository with the Vue.js frontend and Express.js backend.

Steps:

1. Cloning the GitLab Project:
Open your terminal from the directory you want to store the project and use the following commands to clone the project:
git clone https://git.shefcompsci.org.uk/com6103-2022-23/team10/project

2. Open the project in WebStorm:
Launch WebStorm, click on "Open" and navigate to your newly cloned project directory (<your-project-name>). The IDE will automatically recognize and configure your project based on the existing configuration files.

3. Setting up the MySQL Database:
Use the SQL dump file in the MySQL folder of the project to set up the database. Open your MySQL command line or GUI (like MySQL Workbench or phpMyAdmin) and import the SQL dump file.

4. Running the Backend and Frontend Servers:
Front end
Step1: Open a new terminal tab or window, navigate to the frontend directory of project:
➔ **cd frontend**
step2: install npm by following the command
➔ **npm install**
step 3: start the frontend server:
➔ **npm run serve (windows)**
➔ **sudo npm run serve (Mac)**

Backend
Step1: Open another new terminal tab or window, navigate to the backend directory of project:
➔ **cd backend**
step2: install npm by following the command
➔ **npm install**
step 3: start the frontend server:
➔ **node server.js (windows)**
The backend and frontend servers should now be running, and you can access the application in your web browser.

5. Verifying the Setup:

To verify that everything is set up correctly, you can try to access the appliccation in your web browser at http://localhost:3001.

TEAM MEETING/COLLABORATION SAMPLE PICTURE