

VIETNAM INTERNATIONAL UNIVERSITY – HO CHI MINH CITY
INTERNATIONAL UNIVERSITY



WEB APPLICATION DEVELOPMENT PROJECT

RESTAURANT WEB

By

Đỗ Gia Thụy - Student ID: ITCSIU21237

Nguyễn Thanh Tú - Student ID: ITITIU21338

Dương Minh Tuấn - Student ID: ITITWE21092

Advisor: Dr. Nguyen Van Sinh

A report submitted to the School of Computer Science and Engineering in
partial fulfillment of the requirements for the Final Project in Web
Application Development course - 2024

Ho Chi Minh city, Vietnam, 2024

Table of Contents

I. INTRODUCTION	3
1. ABOUT US.....	3
2. THE PRODUCT'S INFORMATION	4
3. WORK BREAKDOWN STRUCTURE.....	4
4. DEVELOPMENT PROCESS	7
Agile	7
SCRUM	7
5. DEVELOPMENT ENVIRONMENT	8
II. REQUIREMENT ANALYSIS AND DESIGN	9
1. REQUIREMENT ANALYSIS	10
USE CASE DIAGRAM	10
A. FUNCTIONAL REQUIREMENTS	13
1: QR Code Ordering System	13
2: Menu Management.....	13
3: Ordering System	14
4: Kitchen Management	15
5: Food Delivery System	15
6: Payment System.....	16
7: Food Status Notifications	16
8: Login/Signup System	17
9: Admin Management.....	18
10: Token Authentication System	18
B. NON-FUNCTIONAL REQUIREMENTS	19
a. Response Time Requirements (How quickly the system responds to user requests)	20
b. Throughput Requirements (The volume of tasks the system can handle within a certain time).....	20
c. Availability Requirements (Ensuring the system is available when users request it)	21
Description	21
2. DESIGN	24
Entity-Relationship Diagram (ERD)	26
Entities	27
Relationships	27
Class Diagram	28
Sequence Diagram 1: Login	30
Sequence Diagram 2: Process Order	31
III. IMPLEMENTATION	33
1. USER'S ACCOUNT MANAGEMENT FUNCTIONS	33
1.1. Login to the store	33
1.2. Register a new account:	34
1.3. Order process	35

1.4 Manage User	40
IV. DISCUSSION AND CONCLUSION	41
Acquired Experience	41
Conclusion	42
V. REFERENCES.....	43

I. INTRODUCTION

This section briefly introduces the background information of our software development team – 3T. More importantly, the basic information about our project is also mentioned in this part. Furthermore, the constraints during our project are included at the end of this section.

1. ABOUT US

<u>Company name:</u>	3T
<u>Team name:</u>	HKT Web Application Solution
<u>Business:</u>	TTT Restaurant Website
<u>Customer:</u>	Haidilao International Holding Ltd.
<u>Contact:</u>	1B Vo Van Ngan, Linh Trung ward, Thu Duc district, HCMC, Vietnam
<u>Email:</u>	3T@hcmiu.com
<u>Phone number:</u>	+84 (08) 3823 4567

Figure 1

2. THE PRODUCT'S INFORMATION

In today's digital era, the restaurant industry is increasingly relying on technology to improve customer experience and streamline operations. At 3T Company, we specialize in creating innovative web solutions, and we are proud to introduce our latest offering: a comprehensive website platform designed specifically to manage and optimize restaurant operations.

Our web-based system provides a complete solution for managing all aspects of a restaurant, from order taking and menu management to payment processing and kitchen coordination. The platform is designed to enhance both the customer and staff experience, making everyday operations smoother and more efficient.

3. WORK BREAKDOWN STRUCTURE

The structure of our project can be expressed in the *Figure 3.1*:

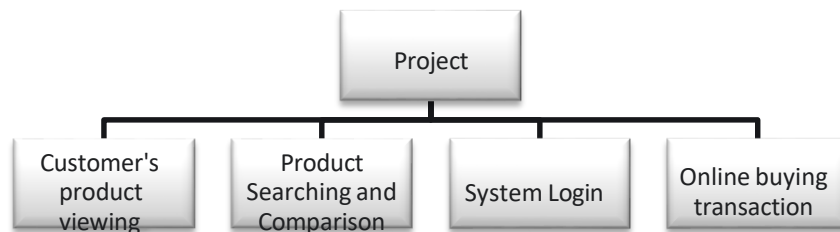


Figure 3.1

In each branch of this tree, we also have subtrees which describes the tasks needed to be accomplished of each team member.

The tasks for the Customer's product viewing is described in the *Figure 3.2*:

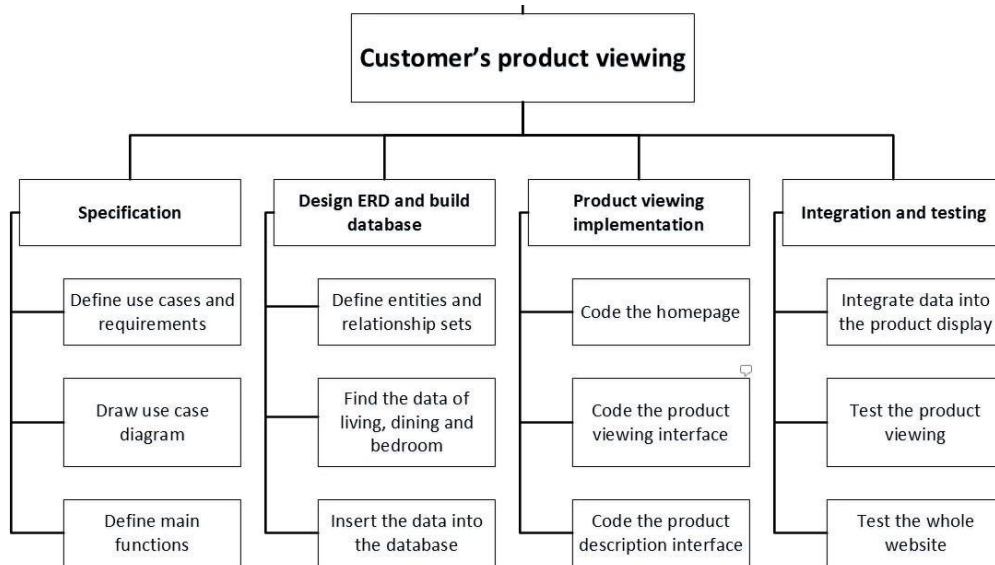


Figure 3.2

The tasks for the Product Searching and Comparison is described in the **Figure 3.3**:

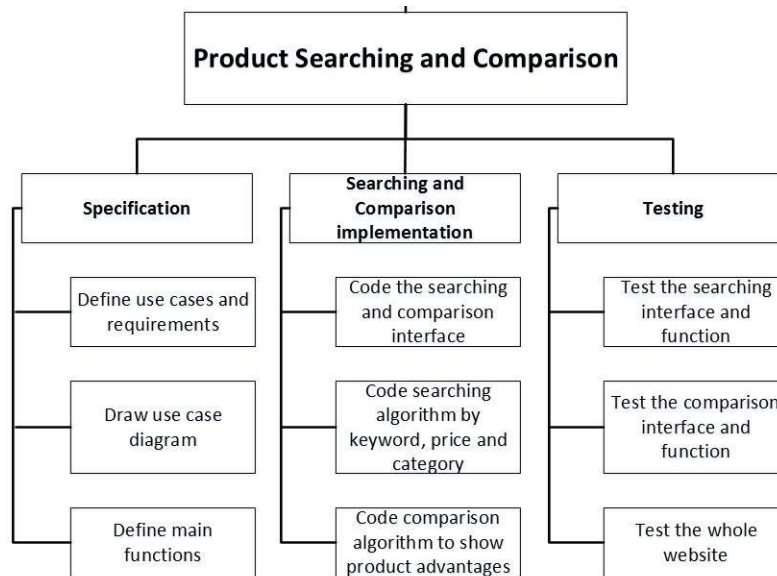


Figure 3.3

The tasks for the System Login is described in the **Figure 3.4**:

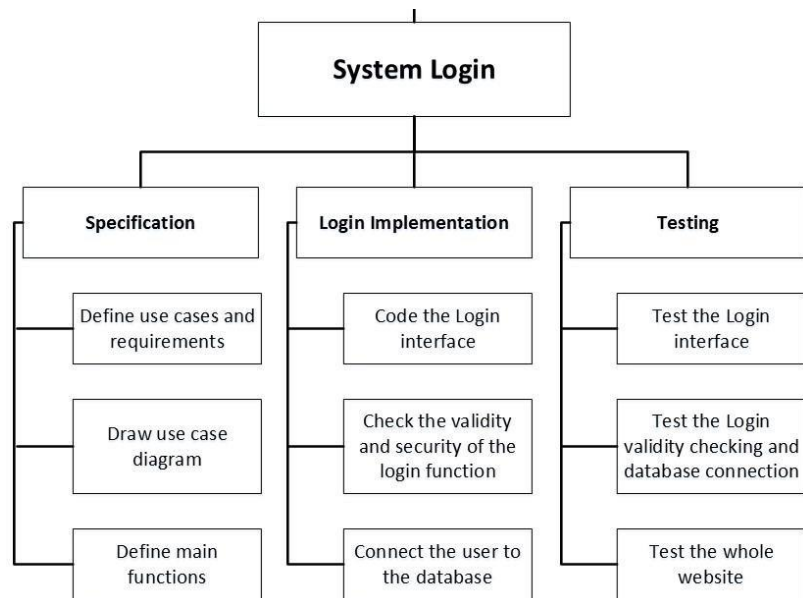
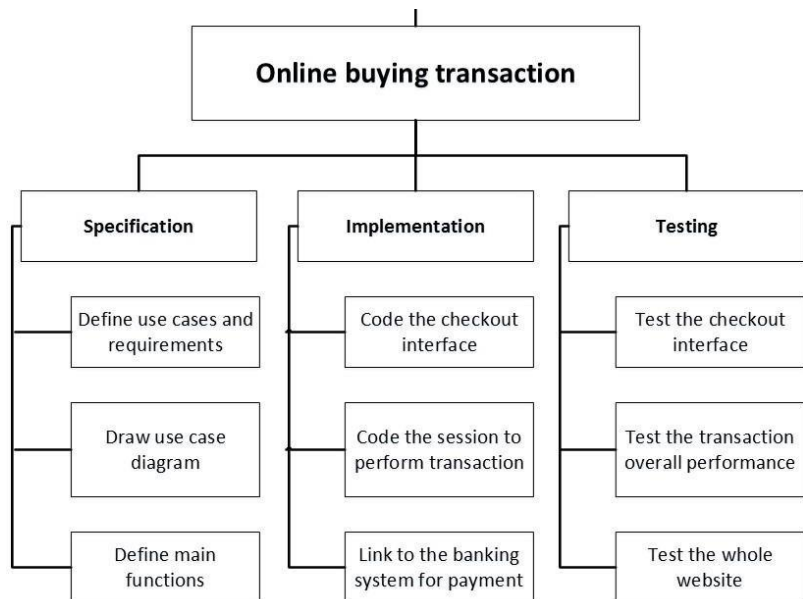


Figure 3.4

The tasks for the Online Buying Transaction is described in the *Figure 3.5*:



These tasks are expected to be completed and thoroughly tested after each iteration of the development process since we aim at applying the agile method rather than the traditional one which separates the steps.

4. DEVELOPMENT PROCESS

Agile

Agile is an approach to project management that centers around incremental and iterative steps to completing projects. The incremental parts of a project are carried out in short-term development cycles. The approach prioritizes quick delivery, adapting to change, and collaboration rather than top-down management and following a set plan.

In the Agile process, there is continuous feedback, allowing team members to adjust to challenges as they arise and stakeholders an opportunity to communicate consistently. Though originally created for software development, the Agile approach is now widely used in executing many different types of projects and in running organizations.

SCRUM

Scrum is an agile team collaboration framework commonly used in software development and other industries.

Scrum prescribes for teams to break work into goals to be completed within time-boxed iterations, called *sprints*. Each sprint is no longer than one month and commonly lasts two weeks. The scrum team assesses progress in time-boxed, stand-up meetings of up to 15 minutes, called *daily scrums*. At the end of the sprint, the team holds two further meetings: one sprint review to demonstrate the work for stakeholders and solicit feedback, and one internal sprint retrospective. A person in charge of a scrum team is typically called a scrum master.

Scrum's approach to product development involves bringing decision-making authority to an operational level. Unlike a sequential approach to product development, scrum is an iterative and incremental framework for product development. Scrum allows for continuous feedback and flexibility, requiring teams to self-organize by encouraging physical co-location or close online collaboration, and mandating frequent communication among all team members. The flexible approach of scrum is based in part on the notion of requirement volatility, that stakeholders will change their requirements as the project evolves.

Our method can be described in the *Figure 4.1*:

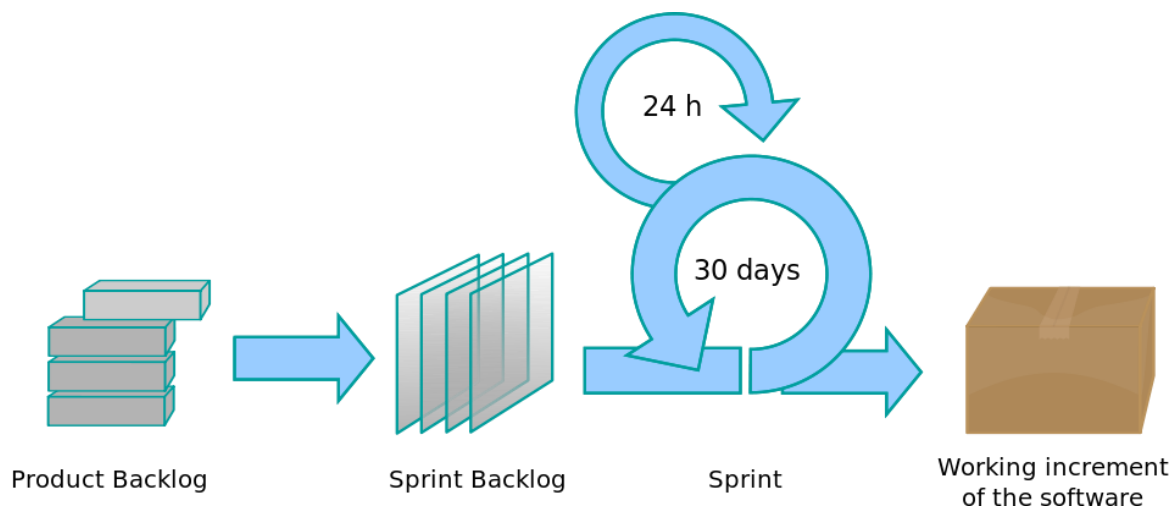


Figure 4.1

5. DEVELOPMENT ENVIRONMENT

Since this is a web-based product, the project is conducted using some Web Design and Programming Language under the model of MVC.

The following Programming Languages are used within our system:

1. **HTML:** to create an outline of the webpages
2. **CSS:** to design the looks of the webpages
3. **React.js:** Handles dynamic content, animations, and real-time updates on the frontend.
4. **Node.js:** Processes customer transactions and handles requests using **Express.js**.

5. **Express.js:** Manages backend logic, processes form data, and serves dynamic content.

6. **MySQL:** to store the database of the system

The code are implemented according to the MVC model in the *Figure 5.1*:

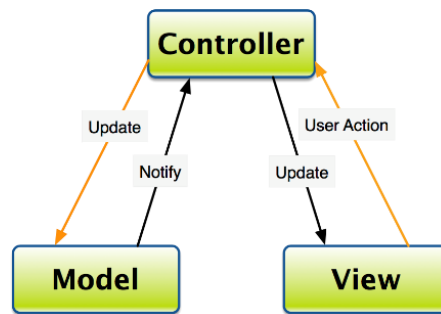


Figure 5.1

1. **Model:** the data of the system stored in the database and external system
2. **View:** the user interface
3. **Control:** the logics and algorithms used to develop a dynamic website with required functionalities

Moreover, our group has also utilized the cloud tools such as Google Docs to make it easier to incorporate the works of all the members in the team.

We also use UML tools to help us draw Entity Relationship Diagram as well Use Case needed in the project such as: Microsoft Visio, Edraw Max.

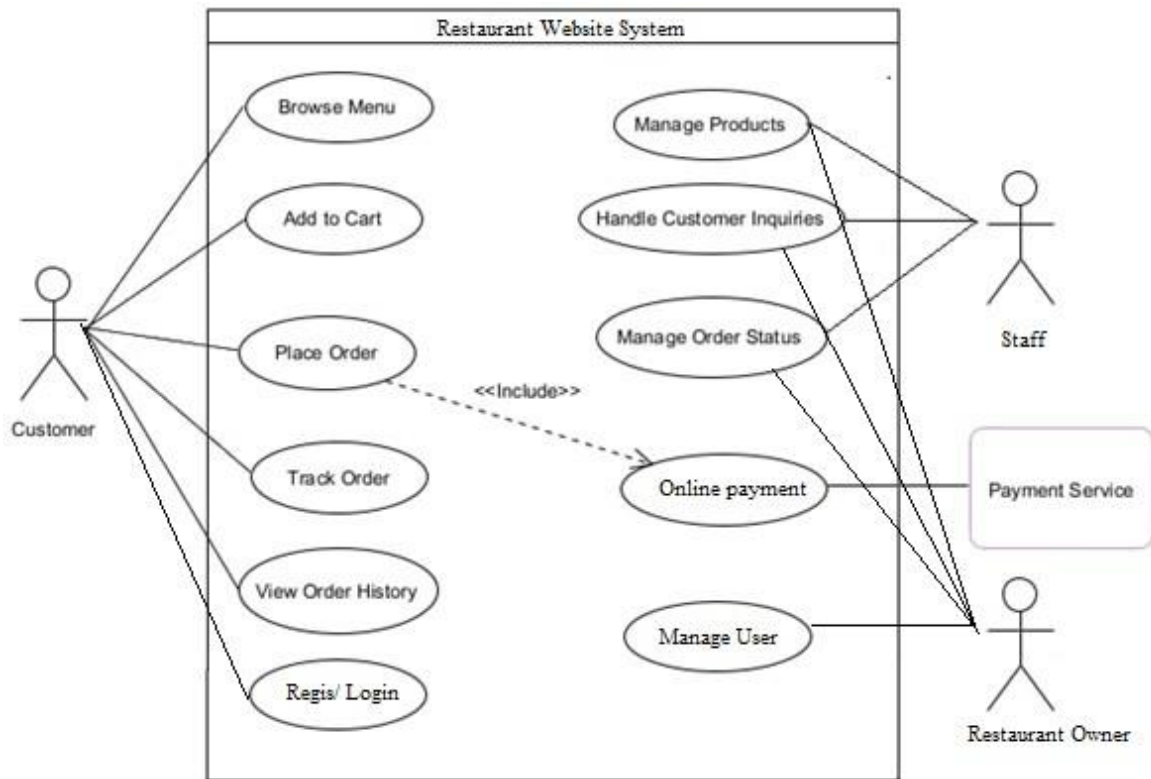
Moreover, we also make use of Microsoft Project - Management Tool - to keep track of the overall progress as well as each team member's work performance.

II. REQUIREMENT ANALYSIS AND DESIGN

This section briefly introduces the requirement analysis and design process. This version is created to provide the path for future implementation of the project. Based on this requirements specification, we will implement each function including all of the conditions as well as functional and non-functional requirements supplied by the customers. During our implementation, we will constantly revise and update the newer version so that we can always keep track of the progress of the project.

1. REQUIREMENT ANALYSIS

USE CASE DIAGRAM



Actor	Role
Customer	The primary user of the platform, who can browse food, make orders, make transactions, and track their order.
Restaurant Owner	The highest authority user can control every aspect of the site, including managing products, handling customer inquiries, managing order status and managing other users' account authorization.
Staff	The frequent admin of the application. Can managing products, handling customer inquiries, managing order status

The following use cases were identified with their goals:

Use case	Goal
Regis / Login	Register an account and Login to it
Browse Menu	Allow the customer to view the available food items
Add to Cart	Add selected items to the cart for checkout
Place Order	The customer placed the order

Use Case

Track Order	Allow the customer to track the status of their order
View Order History	Let the customer view past orders
Manage Products	Admin can manage the product list (add, remove, edit items)
Handle Customer Inquiries	Admin can manage customer support and inquiries
Manage Order Status	Admin can update the order status (processing, completed, canceled)
Manage User	Restaurant Owner can delete/add user account. They can also change the account authorization from customer to staff.
Make Payment	Handles the payment process once the customer places the order

Relationship:

- Use Case "Place Order" includes use case "Make Payment" since the activation of " Make Payment " requires the presence of at least one product ordered in the customer's cart.

A. FUNCTIONAL REQUIREMENTS

1: QR Code Ordering System

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 3 tasks are needed for this function.
- 15 hours of effort are needed for this function.

2. The Scope of the Product:

This function enables customers to scan a QR code at their table and be redirected to the restaurant's online ordering system.

3. Functional and Data Requirements

a. Functional Requirements

- Shall generate a unique QR code for each table.
- Shall direct customers to the corresponding table's ordering page upon scanning.
- Shall allow customers to view the menu and place orders directly from the page.

b. Data Requirements

- Table-specific data (table ID, order details) will be stored in the database.

2: Menu Management

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 4 tasks are needed for this function.
- 20 hours of effort are needed for this function.

2. The Scope of the Product:

This function allows restaurant admins and staff to manage the menu, including adding, updating, or removing menu items.

3. Functional and Data Requirements

a. Functional Requirements

- Shall allow admins to add, edit, or delete menu items.
- Shall allow menu items to be categorized (e.g., appetizers, main dishes, drinks).
- Shall allow updates to item descriptions, prices, and images.
- Shall display the menu with filtering and searching options for customers.

b. Data Requirements

- Menu item data (name, description, price, category, image) will be stored in the database.

3: Ordering System

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 5 tasks are needed for this function.
- 30 hours of effort are needed for this function.

2. The Scope of the Product:

This function allows customers to browse the menu, select dishes, enter quantities, view estimated completion times, and add items to their orders.

3. Functional and Data Requirements

a. Functional Requirements

- Shall allow the customer to select dishes from a categorized list.
- Shall allow the customer to enter the quantity of each selected dish.
- Shall display the estimated completion time for the selected items.
- Shall allow customers to add more items to their current order.
- Shall automatically calculate the total order value as the user adds/removes items.
- Shall generate an invoice with dish names, prices, and total order value.

b. Data Requirements

- Order data (dishes, quantities, total cost) will be stored in the database.
- Customer order data will be linked to the corresponding table number and customer session.

4: Kitchen Management

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 4 tasks are needed for this function.
- 25 hours of effort are needed for this function.

2. The Scope of the Product:

This function allows kitchen staff to receive customer orders, track progress, and manage the cooking schedule.

3. Functional and Data Requirements

a. Functional Requirements

- Shall automatically send customer orders to the kitchen.
- Shall allow kitchen staff to mark orders as “in progress” or “ready.”
- Shall allow the chef to prioritize orders based on estimated cooking time.
- Shall display the list of orders and their current status.

b. Data Requirements

- Kitchen order data (order details, status) will be stored in the database.

5: Food Delivery System

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 3 tasks are needed for this function.
- 15 hours of effort are needed for this function.

2. The Scope of the Product:

This function displays completed orders, showing the dish names and corresponding table numbers for delivery by staff.

3. Functional and Data Requirements

a. Functional Requirements

- Shall display a list of completed dishes with dish names and table numbers.
- Shall allow staff to mark dishes as “delivered” once served.
- Shall update the order status in real-time.

b. Data Requirements

- Delivery status data (dish name, table number, delivery status) will be stored in the database.

6: Payment System

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 6 tasks are needed for this function.
- 40 hours of effort are needed for this function.

2. The Scope of the Product:

This function allows customers to request payment, choose between cash or online payment options, and receive an electronic invoice.

3. Functional and Data Requirements

a. Functional Requirements

- Shall allow customers to request the bill.
- Shall allow customers to pay via cash or online (QR code for online payment).
- Shall generate an electronic invoice with order details.
- Shall verify online payments before confirming the transaction.

b. Data Requirements

- Payment data (payment method, amount, invoice details) will be stored in the database.

7: Food Status Notifications

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 3 tasks are needed for this function.
- 20 hours of effort are needed for this function.

2. **The Scope of the Product:**

This function sends notifications to customers when their food is ready or if the estimated completion time changes.

3. **Functional and Data Requirements**

a. **Functional Requirements**

- Shall notify the customer when their food is ready.
- Shall notify the customer if the estimated completion time is updated.
- Shall update the notification status in real-time.

b. **Data Requirements**

- Notification data (order status, estimated time) will be stored in the database.

8: Login/Signup System

1. **The Scope of the Work**

- This occurs in sprint [X] in the development process.
- 4 tasks are needed for this function.
- 15 hours of effort are needed for this function.

2. **The Scope of the Product:**

This function enables users (customers, staff, admins) to register accounts and log in.

3. **Functional and Data Requirements**

a. **Functional Requirements**

- Shall allow users to register an account.
- Shall allow users to log in with email and password.
- Shall support password recovery.
- Shall differentiate between user roles (customer, staff, admin).

b. **Data Requirements**

- User data (email, password, role) will be stored in the database.

9: Admin Management

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 3 tasks are needed for this function.
- 20 hours of effort are needed for this function.

2. The Scope of the Product:

This function allows admins to manage staff roles and access, and view system reports.

3. Functional and Data Requirements

a. Functional Requirements

- Shall allow admins to assign roles (staff, admin) to users.
- Shall allow admins to view reports on sales and order history.
- Shall allow admins to manage menu items and prices.

b. Data Requirements

- Role and access data (user roles, permissions) will be stored in the database.

10: Token Authentication System

1. The Scope of the Work

- This occurs in sprint [X] in the development process.
- 4 tasks are needed for this function.
- 20 hours of effort are needed for this function.

2. The Scope of the Product:

This function handles secure authentication for users using access tokens and refresh tokens.

3. Functional and Data Requirements

a. Functional Requirements

- Shall use JWT (JSON Web Tokens) for authentication.

- Shall store access tokens in local storage and refresh tokens in cookies.
- Shall allow token refresh when the access token expires.
- Shall securely verify user identity for accessing restricted areas.

b. Data Requirements

- Token data (access tokens, refresh tokens) will be stored in the system for authentication purposes.

B. NON-FUNCTIONAL REQUIREMENTS

1. Operational requirements:

- Must make sure all of the components of the software operate in good manner
- Must keep the host running licensed so that it can handle all of the requests properly
- Must keep an administrator to frequently update the software as well as checking for the system's errors

2. Legal requirements:

- Must cite all the components integrated into the system to avoid copyright violation
- Must keep the own-written source code confidential to avoid unauthorized use

3. Usability requirements:

- Usability is the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of system or component
- Well-structured user manuals
- Informative error messages – Error messages must state clear and might include hint to retrieve the solutions.
- Help facilities.
- Well-formed graphical user interfaces – easy to learn and navigate.

- Efficiency of use: goals are easy to accomplish and with few or no user error.

4. Humanity Requirements:

The users of the system typically have little to no background in using computers. Therefore, the development team must create comprehensive documentation that allows users to easily understand the system by simply reading it. The graphical user interfaces (GUIs) should be intuitively designed to ensure ease of learning and usability. Users should be able to quickly adapt to the system without needing extensive training. Additionally, the interface must be visually appealing and responsive to enhance the overall user experience.

5. Performance Requirements:

a. Response Time Requirements (How quickly the system responds to user requests)

- I. The login response time for both customers and managers must be quick.
- II. The system should update the database efficiently, with minimal delay when modifying multiple database tables simultaneously.
- III. The system must process services swiftly with no noticeable delays. It should be capable of handling 10 different transactions at the same time.
- IV. The website loading time must be fast, with all items correctly generated. Specifically, the system should load over 50 products per page in the shortest time possible.

b. Throughput Requirements (The volume of tasks the system can handle within a certain time)

- I. The system must handle a large number of simultaneous requests from customers without delay. It should support over 1000 visitors browsing the store page at once.
- II. The system must quickly process multiple database modifications without performance degradation.

c. Availability Requirements (Ensuring the system is available when users request it)

I. The system should utilize efficient memory management, including garbage collection in the programming language.

Description:

This is an e-commerce platform that may experience thousands of visitors daily, which places significant strain on the server if requests are not properly managed. The shopping cart system must be capable of processing orders simultaneously without delays or overload. The website must efficiently handle various types of requests from numerous customers on different operating systems, ensuring smooth operation without any interruptions.

6. Maintainability Requirements:

The **Online Store system** is designed to require continuous updates and changes. Since the users have limited knowledge about the internal workings of the system, the development team must implement it in a way that allows both users and future maintenance teams to easily manage and modify the system.

- Provide features that enable users to update and manage product and customer information.
- Write clear and well-structured source code, including detailed comments, to ensure better understanding and ease of modification.
- Design a straightforward and reliable database that facilitates easy maintenance and future updates.

7. Support Requirements

The system users, particularly the **Store Manager**, typically have limited technical knowledge regarding system maintenance. Therefore, in addition to developing the system, the development team should ensure that users can receive prompt support whenever needed.

- Provide a **hotline** for the store manager to report issues and receive immediate assistance.
- Offer a **remote maintenance tool** (such as TeamViewer or Remote Desktop) to provide direct support to the store manager.
- Perform regular system checks and maintenance to ensure stability, with updates scheduled monthly or every 2–3 months.

8. Security requirements:

- The web-based system shall ensure that data is safeguarded against unauthorized access.
- The system must maintain its integrity, protecting it from accidental or malicious damage.
- Login attempts shall be restricted: After five failed login attempts, access to the website will be blocked for 24 hours.
- Only the **administrator** shall have permission to make changes to system data.
- All modification events will be logged, with each log entry containing the following details: **date**, **time**, **user**, **action**, **object**, **prior value**, and **new value**.
- All communication between the system's data server and clients must be encrypted using secure protocols such as **SSL** or **SSH** for HTTP.
- A session should be initiated upon customer login, and a timeout should be applied after a period of inactivity.
- Customer information must be encrypted before being stored in the database.
- All system data, including product information and customer data (such as cookies and sessions), must be backed up every 24 hours, with one copy stored securely in a location separate from the primary system.
- A **privacy policy** must be provided to restrict third-party access to customer information.

9. Interface requirements:

- Describe all of the technical requirements that affect interfaces such as protocol management, scheduling, directory services, broadcasts, message types, error and

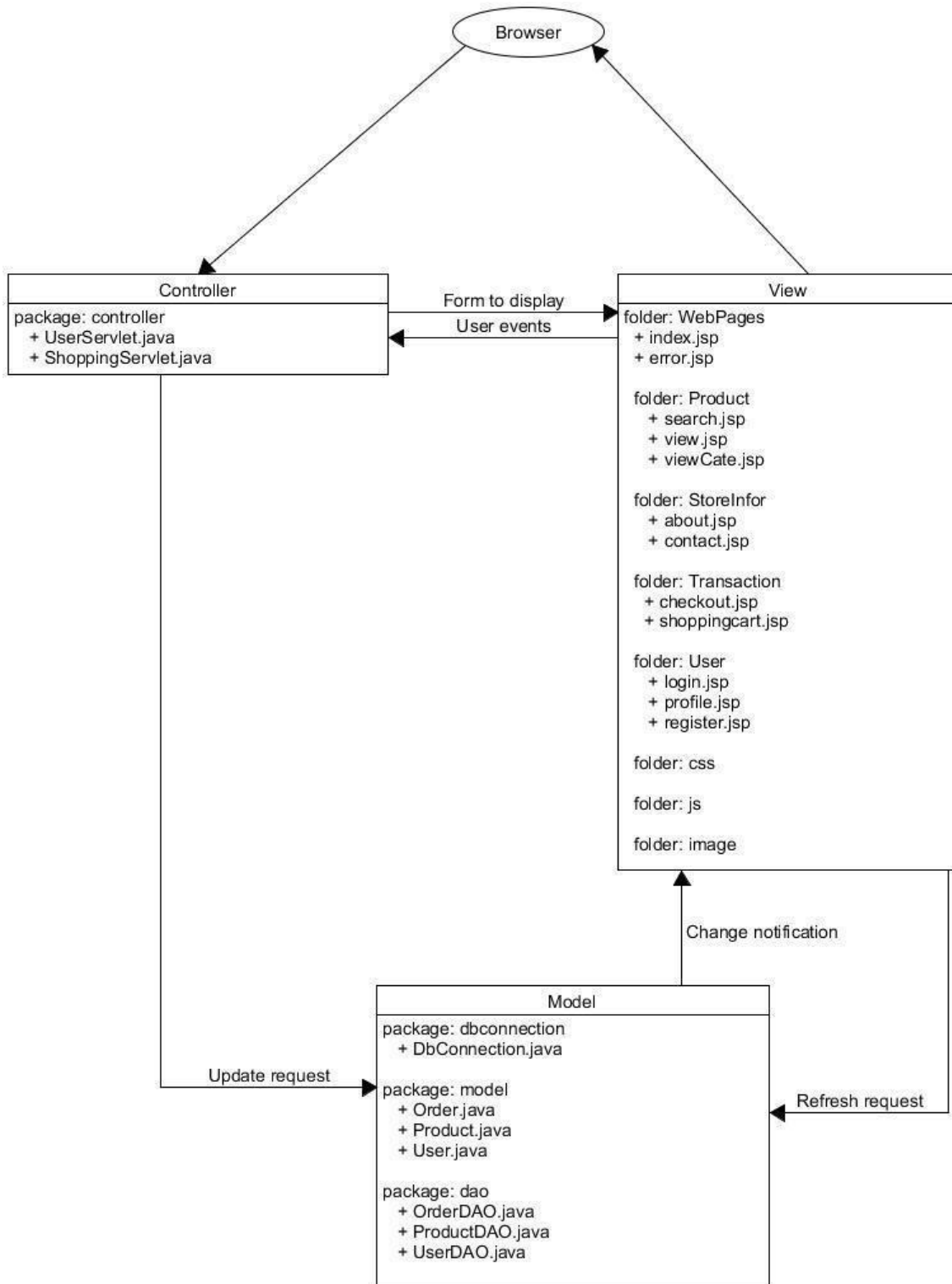
buffer management, security, etc. Assign a unique ID number to each requirement. □

Some non-functional requirements of interface:

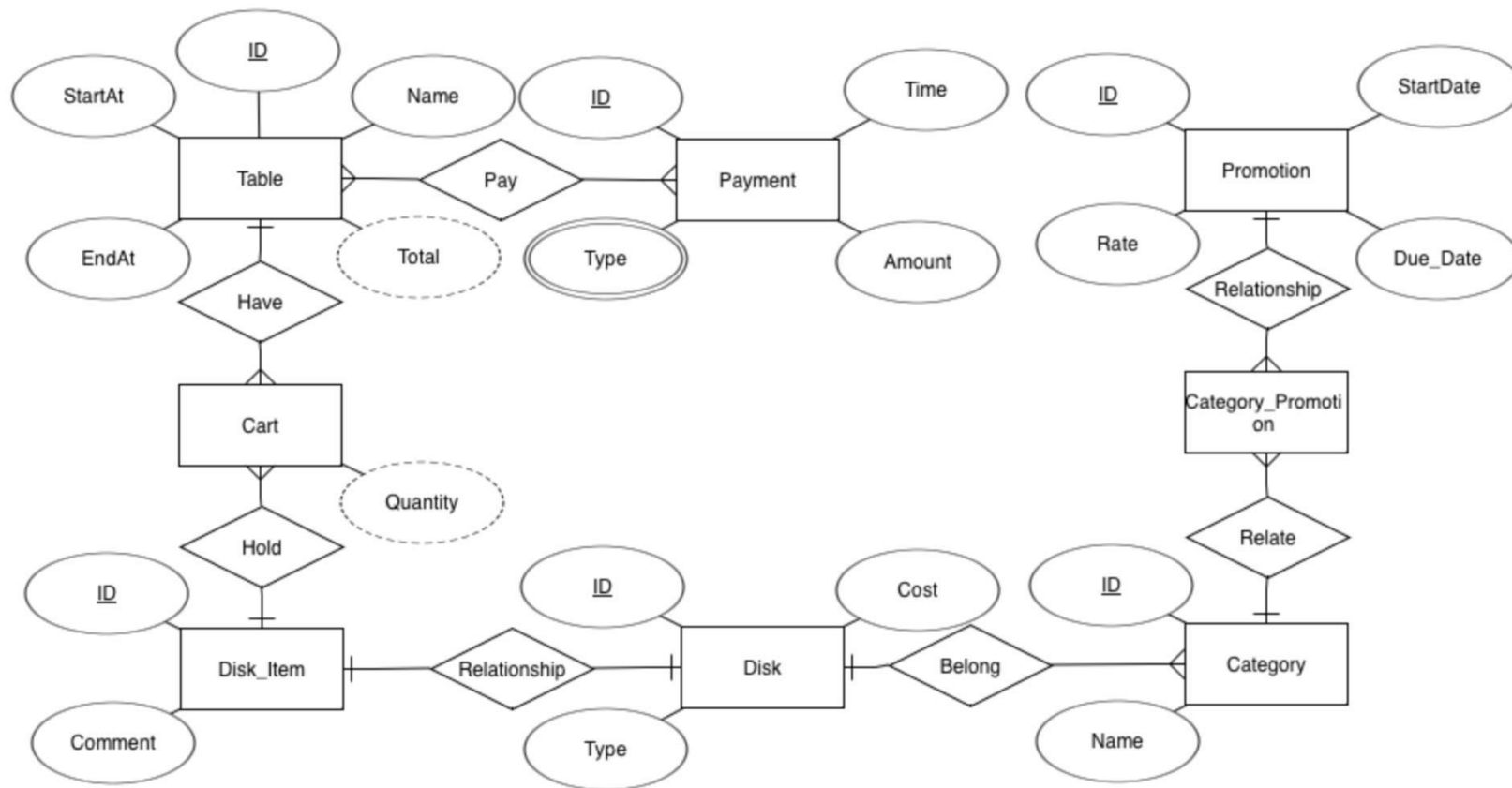
- +Buttons should include feature symbols equivalent to their functionalities .
- + Sales report included in every successful transaction.
- + Include language options for native and foreign visitors/customers

2. DESIGN

System Architecture Model



Entity-Relationship Diagram (ERD)



Entities

1. Table

- Attributes: ID, StartAt, EndAt, Name, Pay
- Weak Attribute: Total

2. Payment

- Attributes: ID, Time, Amount, Type

3. Promotion

- Attributes: ID, StartDate, Due_Date, Rate

4. Cart

- Attribute: Quantity

5. Disk_Item

- Attributes: ID, Comment, Relationship

6. Disk

- Attributes: ID, Cost, Type

7. Category

- Attributes: ID, Name

8. Category_Promotion

4. **Belong** (between Disk and Category)

5. **Relate** (between Category_Promotion and Category)

6. **Relationship** (between Promotion and Category_Promotion)

7. **Relationship** (between Promotion and Category_Promotion)

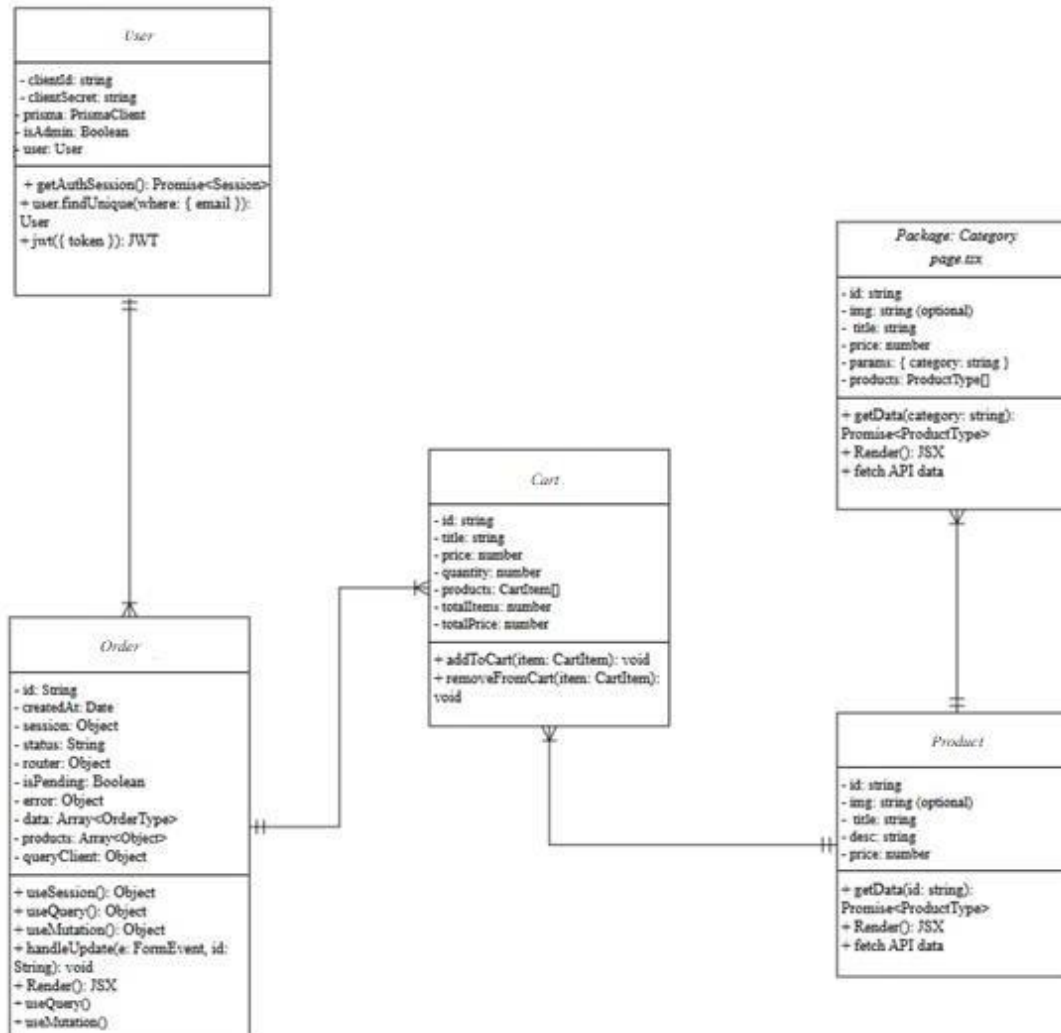
Relationships

1. **Have** (between Table and Cart)

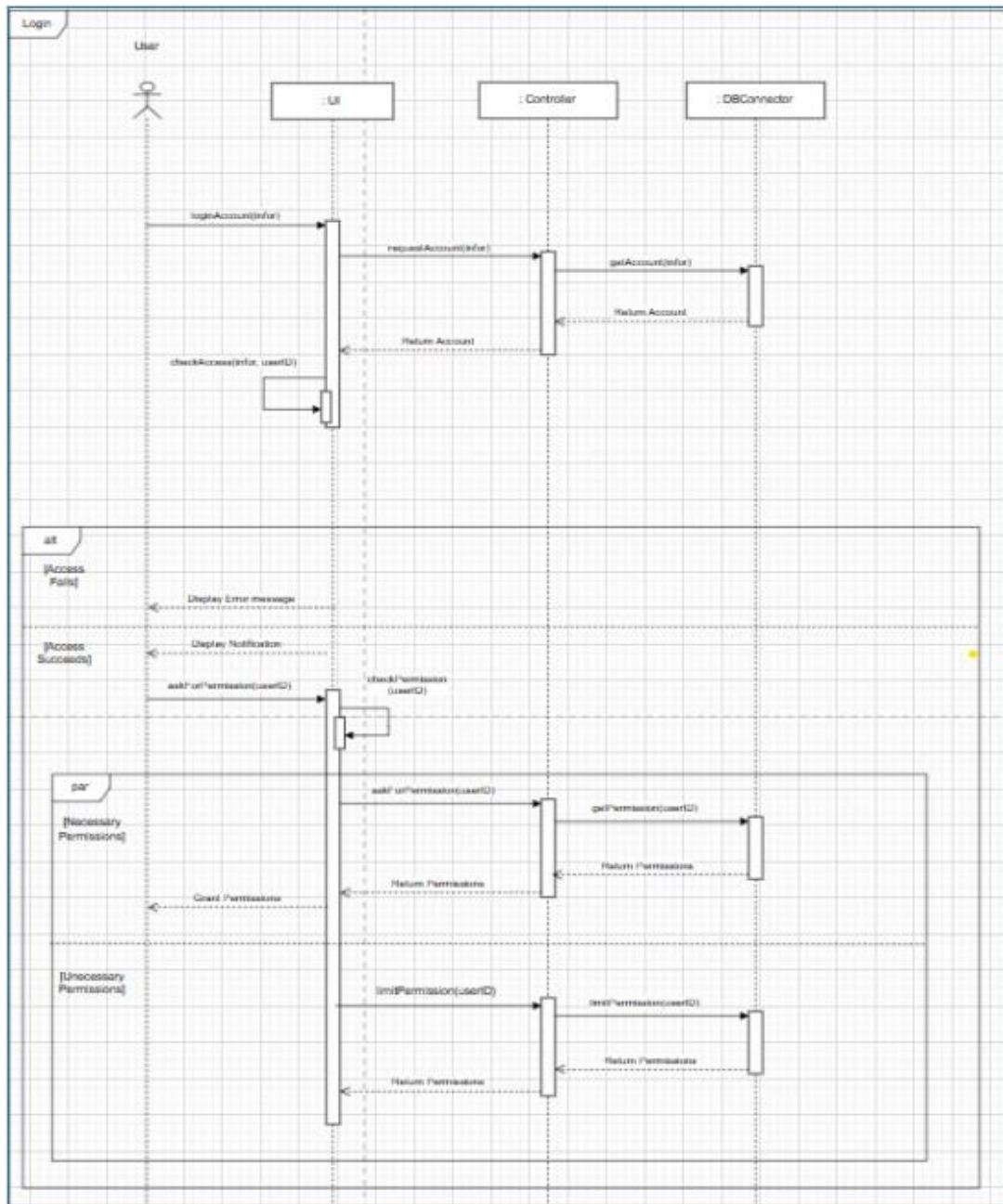
2. **Hold** (between Cart and Disk_Item)

3. **Relationship** (between Disk_Item and Disk)

Class Diagram



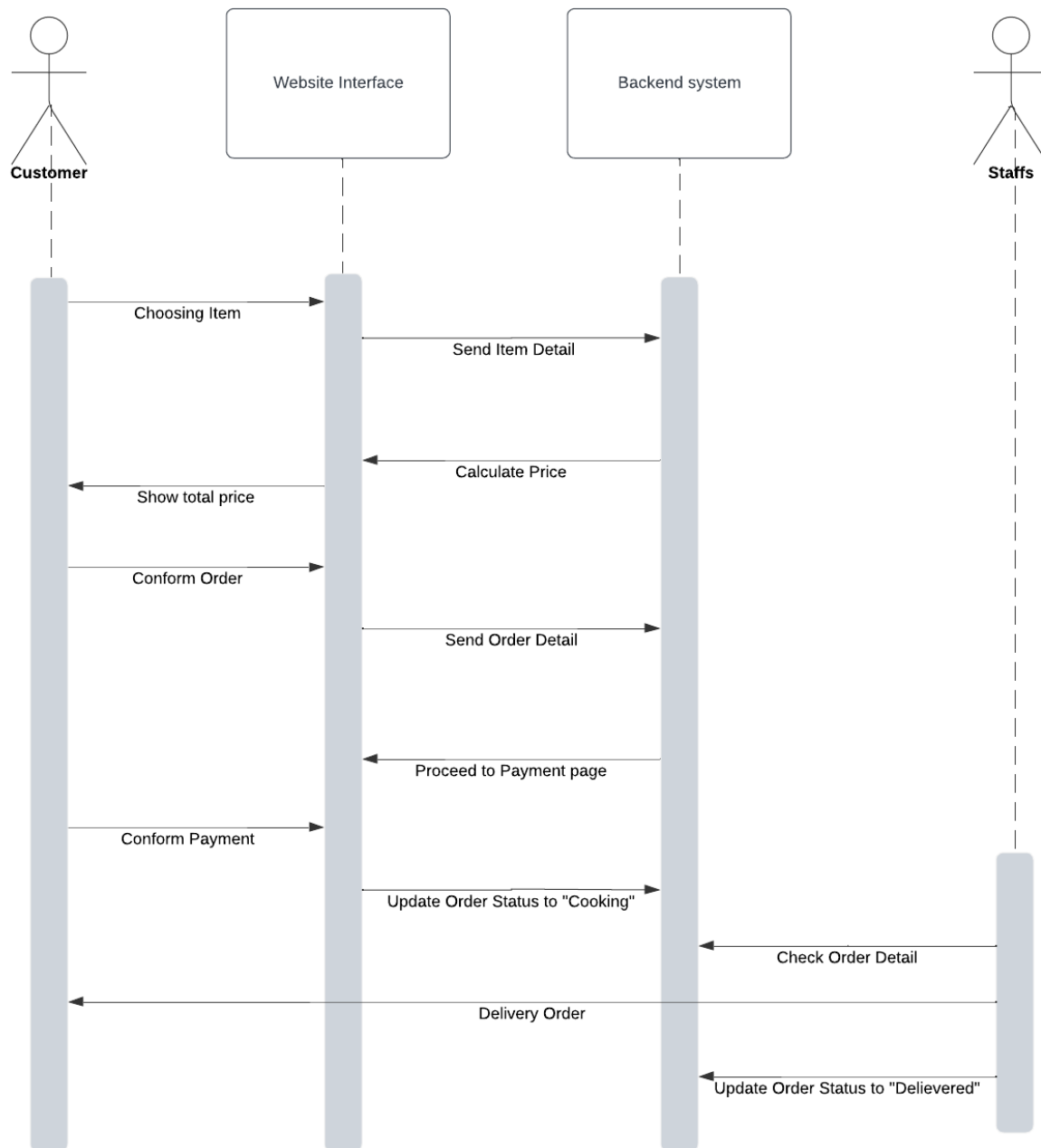
Sequence Diagram 1: Login



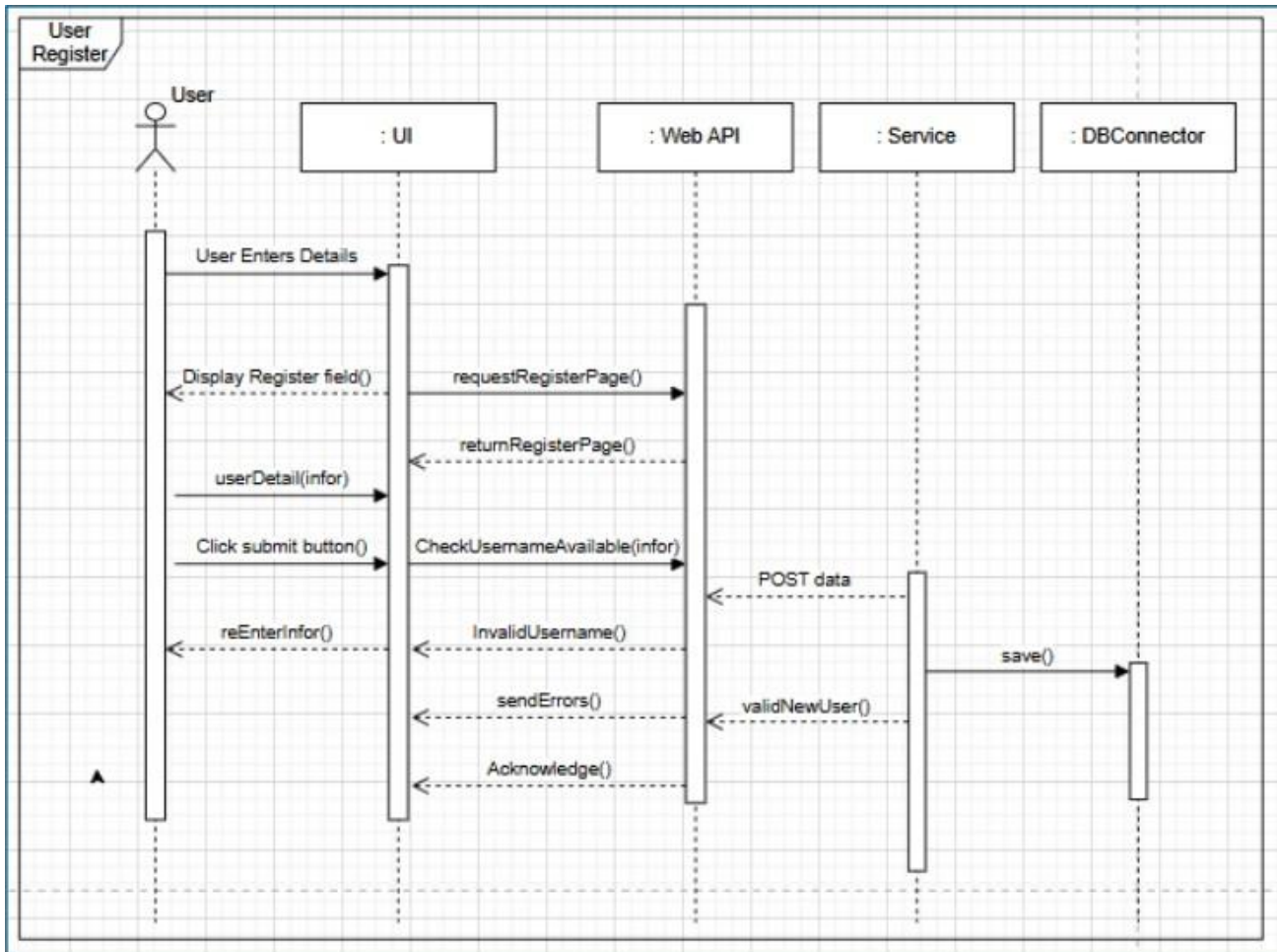
Sequence

Diagram 2:

Process Order



Sequence Diagram 3: User Register



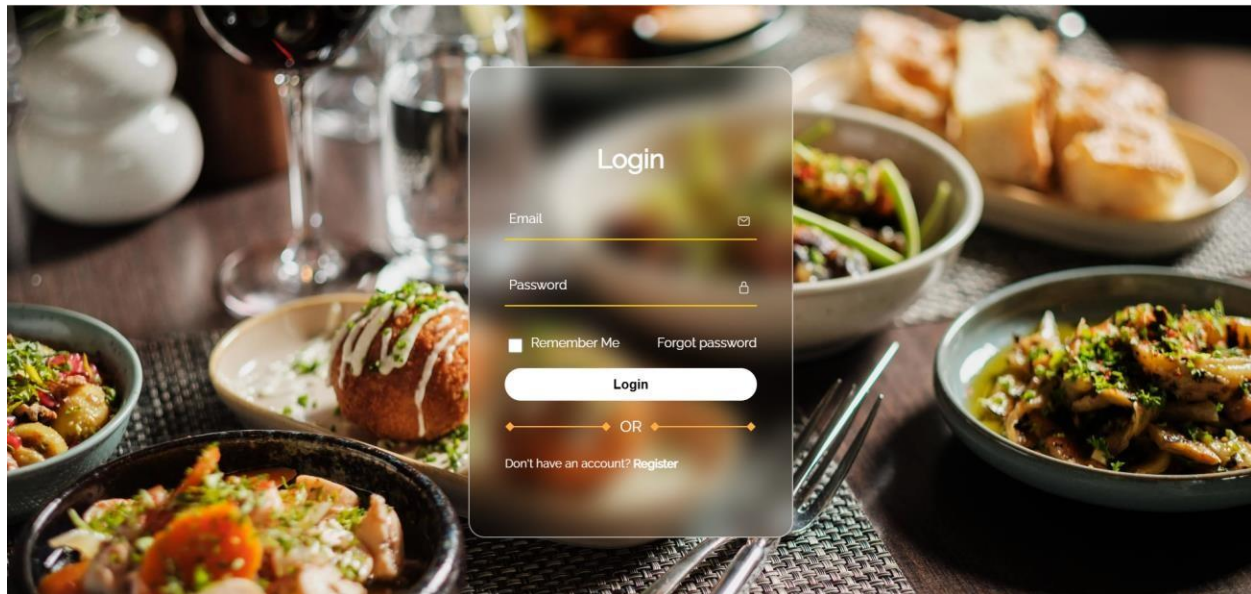
III. IMPLEMENTATION

1. USER'S ACCOUNT MANAGEMENT FUNCTIONS

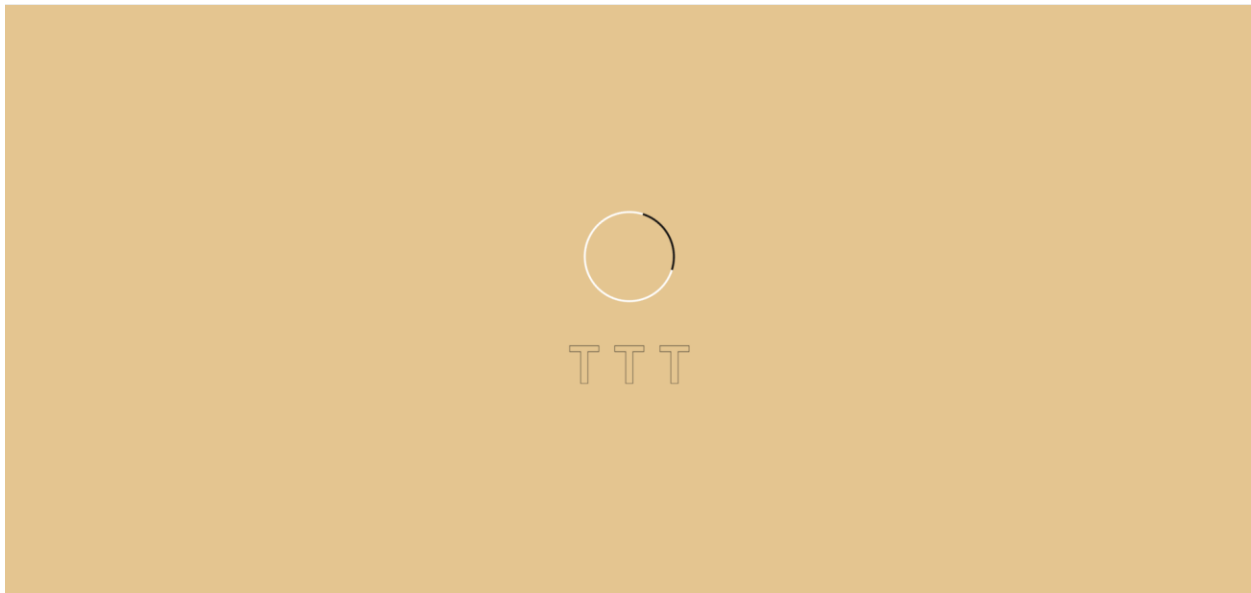
1.1. Login to the store:

Step 1: Go to the shopping page via the link: _

Click on **Login** button from the homepage, the login page will appear.



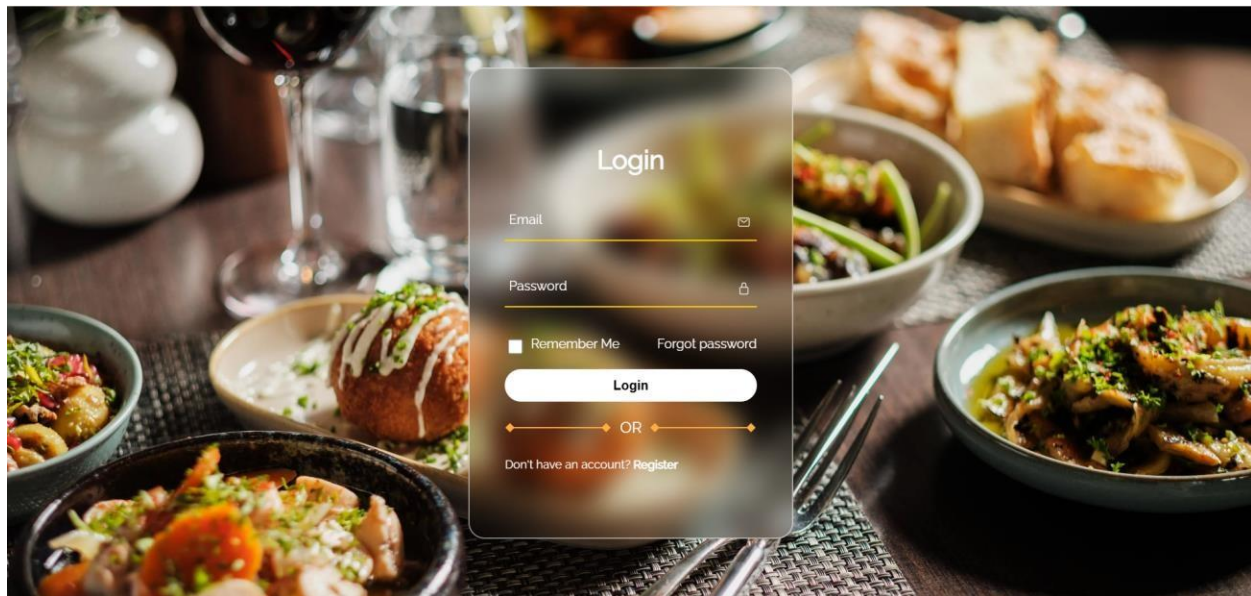
Step 2: Type user account information (email and password) in form and click on **Login** button.



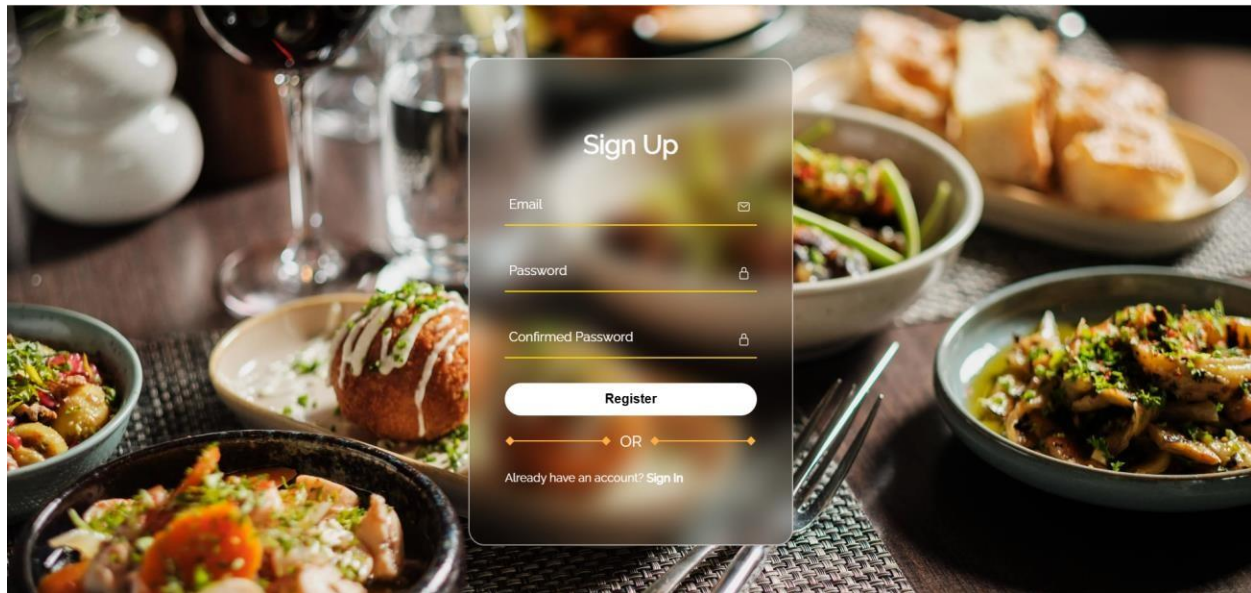
Loading page

1.2. Register a new account:

Step 1: In the login page, create **Register** button



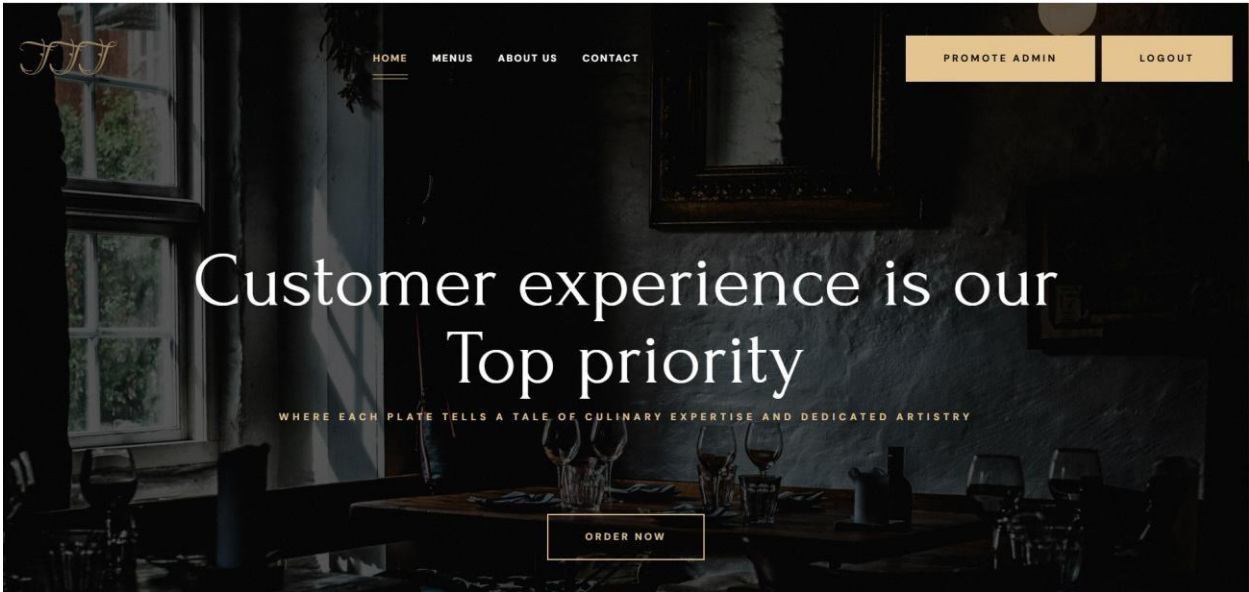
The register page will appear asking user for typing login information.



Step 2: Fill all the fields required and click **Register** to finish the registration

1.3. Order process

After successfully register, customer will need to log in again). Then they will be redirected to the homepage.



[HOME](#) [MENUS](#) [ABOUT US](#) [CONTACT](#)

[PROMOTE ADMIN](#)

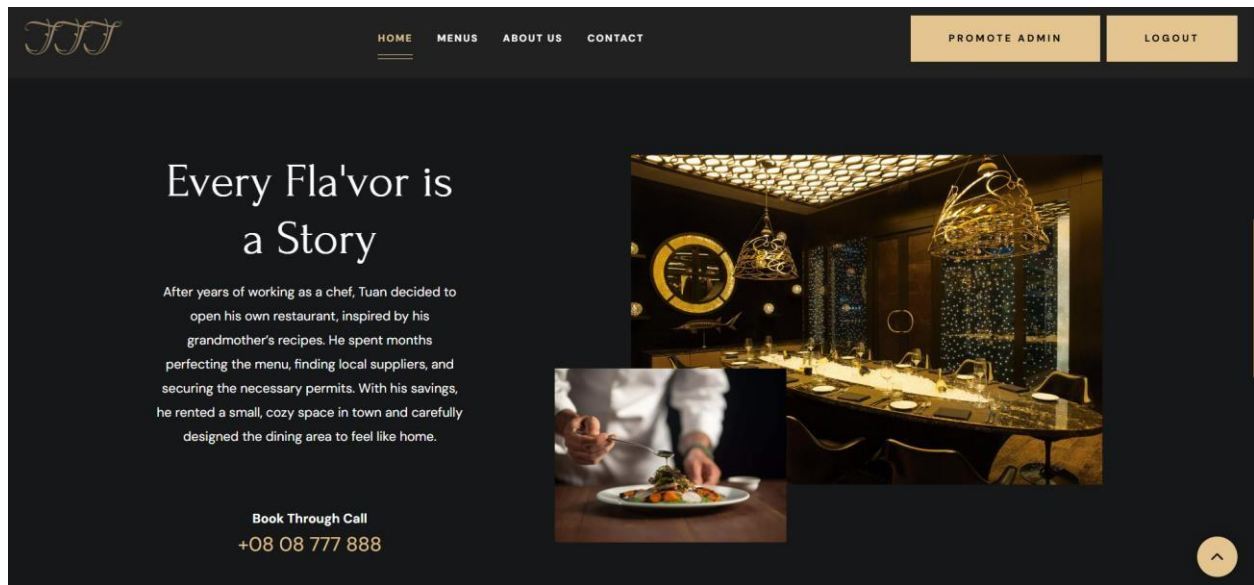
[LOGOUT](#)

Customer experience is our Top priority

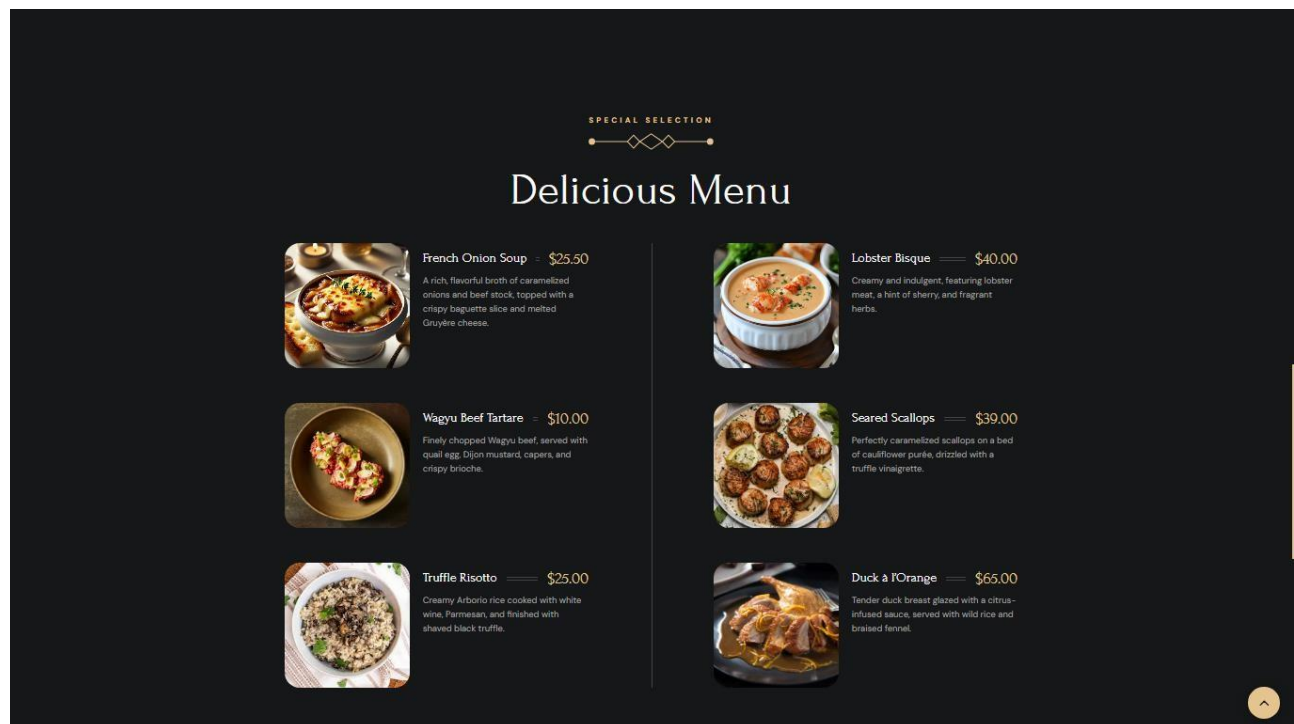
WHERE EACH PLATE TELLS A TALE OF CULINARY EXPERTISE AND DEDICATED ARTISTRY

[ORDER NOW](#)

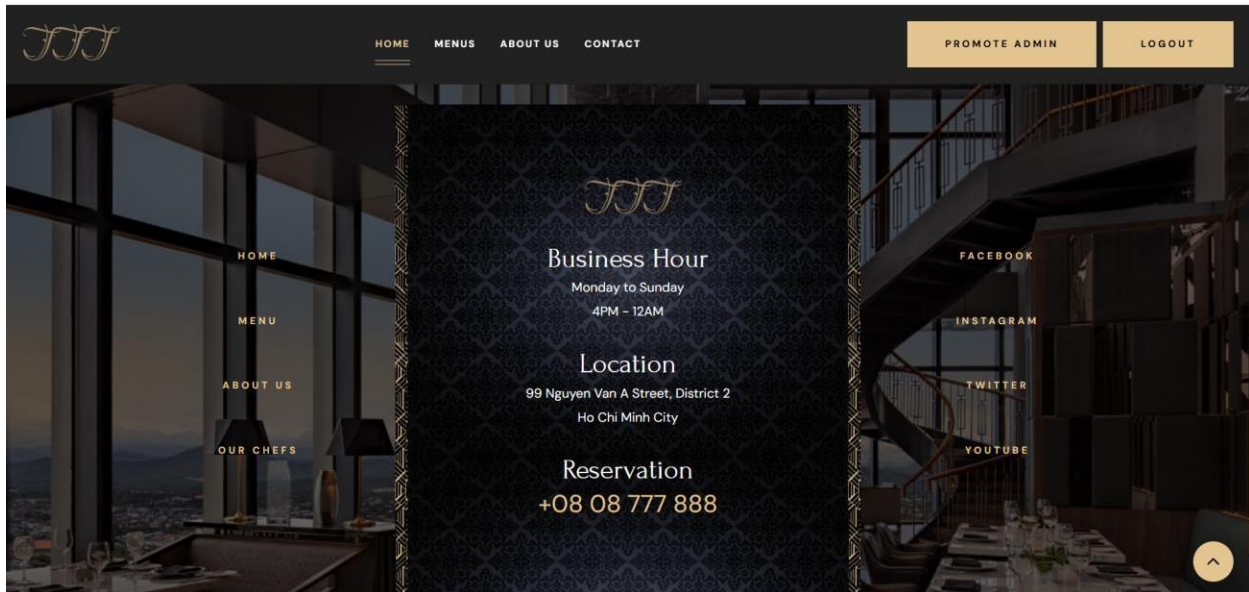
Step 1: Click on the **About Us** to know more about us



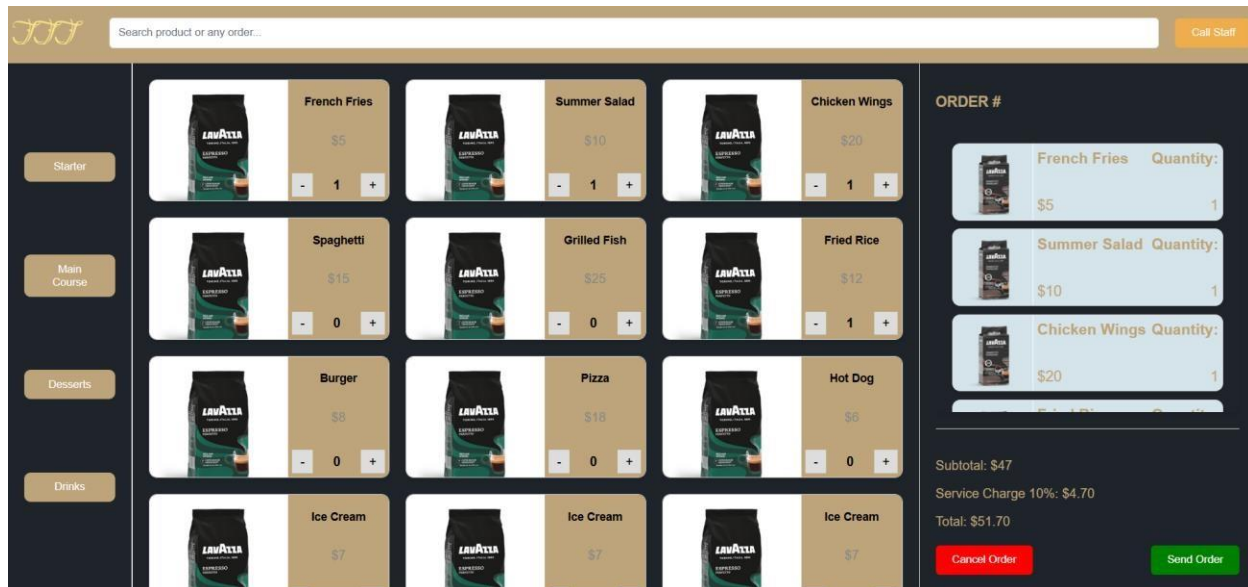
There you can browse our carefully chosen menu



If you want to contact, clicking on **Contact**

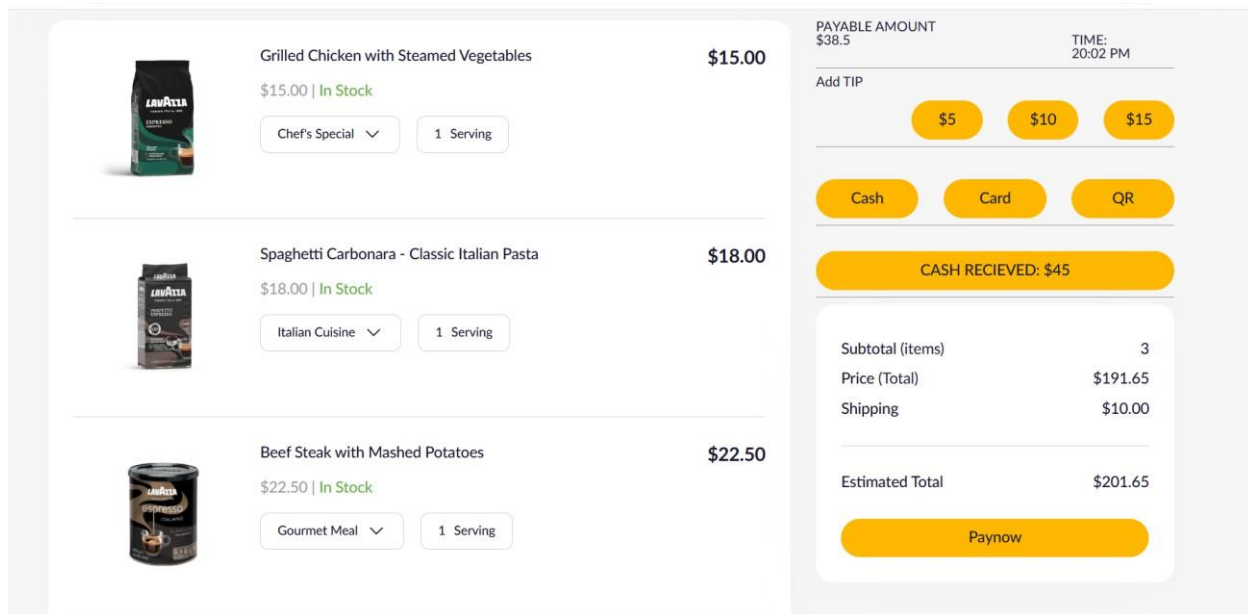


Step 2: Click on the **Cart** page to start order



Step 3: Click on button to choose, it will appear on your right **Step 4:**

Click on Send Order to go to check out and pay



1.4 Manage User

Users

Items







Tables

Message

Setting

Logout

http://localhost:3000/manage-user

ID	Avatar	Name	Email	Promote	Delete User
1		duong4	duong4@gmail.com	Promote	Delete
2		duong1	duong1@gmail.com	Promote	Delete
3		duong3	duong3@gmail.com	Promote	Delete
4		duong5	duong5@gmail.com	Promote	Delete
5		duong6	duong6@gmail.com	Promote	Delete
6		duong7	duong7@gmail.com	Promote	Delete

Delete Selected Users

IV. DISCUSSION AND CONCLUSION

Acquired Experience:

Challenges

During the project, the team faced several challenges, including:

1. **Integration of Payment Gateways:** Ensuring secure and seamless transactions required significant effort, particularly in adhering to industry security standards and managing various payment methods.
2. **Real-Time Order Updates:** Implementing real-time notifications posed technical difficulties, especially when synchronizing data between the frontend, backend, and database.
3. **Scalability Issues:** During peak load testing, the system initially exhibited slower response times, necessitating optimization of database queries and API endpoints.
4. **Team Coordination:** With a team of nine members, managing tasks, ensuring clear communication, and resolving overlaps in responsibilities were critical challenges.

Lesson Learned

- **The Value of Comprehensive Planning:**
 - Properly defining the project timeline and milestones from the start proved essential for staying on schedule. Tools like Google Calendar helped the team effectively allocate time to critical tasks.
- **Importance of Collaboration Tools:**

- Utilizing tools like Microsoft Teams for communication, GitHub for version control, and Figma for UI design significantly streamlined collaboration and ensured consistency across all project phases.
- **Adaptability in Problem-Solving:**
 - Unexpected challenges, particularly with technical integrations like payment gateways and WebSocket implementations, emphasized the importance of remaining flexible and open to alternative approaches.
- **User-Centric Development:**
 - Early and ongoing feedback from potential users proved invaluable for refining features. For example, adding dietary preference filters and improving mobile responsiveness addressed specific user needs effectively.
- **Technical Skills Development:**
 - The project provided opportunities for team members to deepen their expertise in tools and frameworks like ReactJS, PostgreSQL, and Docker, which will benefit future endeavors.
- **Effective Communication:**
 - Regular updates and clear documentation reduced misunderstandings and ensured all team members were aligned, even when schedules conflicted.

Conclusion:

The **Order Menu** project has provided us with a deeper understanding of the **Web Application Development process**. Now that the project is complete, there are several improvements and extensions that could be made in the future. Firstly, introducing a **product recommendation feature** would enhance the system's effectiveness and user engagement. This function would suggest related products to customers based on their previous orders, encouraging additional purchases and boosting overall sales for the store. Secondly, as the current system runs as a demo on a single machine's local host, it can only accommodate one user session at a time. To make the system scalable and deploy it in a real-world setting, the task of supporting **multiple user sessions** must be addressed. Lastly, for a fully integrated e-commerce experience, the system should incorporate **payment gateway systems**, allowing both customers and store managers to complete transactions smoothly within the platform.

This Web Application Development project has also provided valuable learning experiences. Our **coding skills** and ability to organize and structure code within a project have improved significantly. This will be crucial for maintaining and expanding the system in the future. The project also helped us strengthen our ability to **analyze requirements** and translate them into practical designs before beginning development. Furthermore, working on a project of this scale has enhanced our **teamwork** and **communication skills**, making it easier to collaborate and share ideas effectively.

In summary, the Order Menu project has not only allowed us to apply the knowledge gained in the course but also provided us with **hands-on experience** and **valuable skills** that will be beneficial for future projects.

V. REFERENCES

- **SCRUM**
- <https://www.scrum.org/>
- **Agile**
- <http://agilemethodology.org/>
- **Web-page template**
- <http://themeforest.net/>

- **Database Management System (DBMS)**
- <http://www.oracle.com/>
- **HTML, CSS, techniques**
- www.w3schools.com