LONDON METROPOLITAN UNIVERSITY

informatics college • pokhara

Module Code & Module Title

CS6P05NP & Project

Level 6 - Project

Assessment Type

FYP Interim Report

5th Semester

2023/24 Autumn

Student Name: Manish Dhamala

London Met ID: 22069006

College ID: NP04CP4A220070

Project Name: Malt (Food Ordering Web Application)

Assignment Due Date: Wednesday, January 8, 2025

Assignment Submission Date: Wednesday, January 8, 2025

Submitted To: Mr. Sandeep Gurung

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Malt is a **web application** that allows users to order food from restaurants online. There are two reasons for naming this web application as "Malt". First, malt typically refers to grain (usually barley) that has been processed for brewing beer or distilling alcoholic beverages, such as whiskey, vodka, rum, gin, and tequila. The second reason is a combination of 'salt,' an ingredient found in almost every dish, and the first letter of my name, **M**anish resulting in the name **"Malt."**

In the quick evolving digital world, online services have become an essential part of everyday life, especially in urban areas. In this world, where technological adoption is rapidly increasing, the demand for online food ordering platforms has grown considerably. Traditional methods of ordering food over the phone or in person are gradually being replaced by more efficient, user-friendly web applications which relate to the fast-paced lifestyle of today's consumers (Gautam, 2021). However, the existing food ordering solutions often lack the customization, simplicity, and localized features required to meet the specific needs and preferences of the people.

This project aims to provide a comprehensive and user-friendly food ordering web application. The program will not only simplify the ordering process for customers, but it will also give a platform for **restaurants** to reach a larger audience and manage orders more efficiently. By using modern web technologies, the project seeks to address the gap in the current market, providing a solution that is both user-friendly and perfectly aligned with the technological capabilities and cultural context of Nepal.

1.1 Scope of the project

This project is about creating a simple and easy to use food ordering web application called Malt. The goal is to make it easier for customers to order food online and for restaurants to manage orders. The app will focus on providing a smooth and quick experience for users with helpful features. The main features of the Malt app include:

- **User Registration and Login**: Users can create an account, log in securely, and reset their password if needed.
- **Restaurant Listings**: A list of restaurants with basic details like the name and location.
- **Menu Display**: Restaurants will show their menu with food items, descriptions, prices, and pictures to help users choose.
- **Search and Filter**: Users can search for specific restaurants or dishes and filter results by price or food type.
- **Order Placement**: Users can add items to their cart, check the order summary, and complete the checkout process.
- **Payment Options**: The app will offer safe payment methods like digital wallets and cash on delivery.
- **Order Confirmation**: Users will get an immediate confirmation of their order through the app, email, or SMS.
- **Order Tracking**: Users can track the status of their order in real time, from order acceptance to delivery.
- **Favorites**: Users can save their favorite restaurants for quick access later.
- **Promotions and Offers**: Restaurants can share special events or discounts to attract customers.
- **Restaurant Dashboard**: Restaurant owners can manage their menus, track orders, and handle promotions.

1.2 Aims and Objectives

- ➢ The main Aim of this project is:

  To create a user-friendly web application that simplifies the online food ordering process for customers and provides a robust, efficient platform for restaurants to effectively manage and increase their online presence and operation.

1.2.1 Project Objectives

- ➢ The objectives of the project are as follows:
  - **To create a fully functional application** using **React** for front-end development and **Java** for the back-end development.
  - **Develop a user-friendly interface** that makes it easy for customers to browse, select, and order food online.
  - **Create a reliable and secure platform** for handling online orders, payments, and customer data.
  - **Enable restaurants to easily manage orders** through a simple dashboard.
  - **Implement features for order tracking** so, customers can follow their order status in real-time.
  - **Provide customization options for restaurants**, allowing them to update restaurant's details, menus and prices.
  - **Optimize the application for fast loading times** to enhance user experience, even on slower internet connections.

1.2.2 Personal Objectives

- ➢ My personal objectives for this project are as follows:
  - To do research on several related applications and understand how they operate and their limitations.
  - To create the application with the appropriate software methodology.
  - To develop the capacity to stick to the project schedule and address issues as they arise in a timely manner.
  - To challenge my own ability for developing applications that are practical in the real world.

1.3 Resource Required

These are the list of resources that are required for building this project:

1) **Frontend Technologies**
    - **React:** React is a popular JavaScript library used to build user interfaces, especially for single-page applications (SPAs). It allows developers to create reusable components, manage the application's state, and render updates efficiently.
    - **Tailwind CSS:** Tailwind CSS is a utility-first CSS framework that provides predefined CSS classes to style elements directly in the HTML markup. Unlike traditional CSS frameworks that offer predefined components like buttons or forms.
2) **Backend Technologies**
    - **Java:** Java is a popular, high-level, object-oriented programming language that is used to build a wide range of applications, from mobile and desktop applications to large-scale enterprise systems. It was known for its platform independence, meaning Java applications can run on any device or operating system that has a Java Virtual Machine (JVM).
    - **Spring Boot:** Spring Boot is a framework that simplifies the process of building Java-based web applications and microservices. It is part of the larger Spring framework and provides tools to quickly create production-ready applications with minimal configuration.
3) **Database**
    - **PostgreSQL:** PostgreSQL is an open-source, relational database management system (RDBMS) that is known for its stability, scalability, and support for advanced data types and features. It is used to store and manage data in a structured format, using tables, rows, and columns.

4) **Version Control**
   - **Git:** Git is a distributed version control system (VCS) used to track changes in source code during software development. It allows developers to collaborate, manage different versions of code, and rollback to previous states if needed.
   - **GitHub:** GitHub is a cloud-based platform that hosts Git repositories and facilitates collaborative software development. It provides a web interface to manage Git repositories, track issues, and collaborate with other developers on code.
5) **Development Environment**
   - **IntelliJ IDEA**: IntelliJ IDEA is a popular integrated development environment (IDE) developed by JetBrains, primarily designed for Java development. This is my primary IDE for developing the backend with Spring Boot.
   - **Visual Studio Code**: Visual Studio Code (VS Code) is a lightweight, open-source code editor developed by Microsoft that supports a wide variety of programming languages through extensions. I am using this editor for frontend development with React.
6) **Design Tool**
   - **Figma:** Figma is a cloud-based design tool primarily used for UI/UX design, prototyping, and collaboration. It allows designers to create interactive prototypes, wireframes design for websites and mobile applications. I am using Figma for designing prototypes for my application.
7) **Diagram Tool**
   - **Draw.io:** Draw.io (now known as diagrams.net) is a free, web-based tool used for creating various types of diagrams, such as flowcharts, organizational charts, UML diagrams, network diagrams, wireframes, and more. It is designed to help users visualize information and processes in an easy-to-understand way.

8) **Testing Tool**
   - **Postman:** Postman is a popular **API** (Application Programming Interface) testing tool that simplifies the process of developing, testing, and managing APIs. It provides a user-friendly interface for sending HTTP requests to RESTful services and analyzing the responses.
   - **JUnit:** JUnit is a widely used testing framework for Java programming, designed to help developers write and run **unit tests**. Unit tests are small, isolated tests that verify individual components or methods of an application to ensure they work as expected.
9) **Third party services**
   - **Payment Gateway:** Integration with services like E-Sewa or Khalti for processing payments.
   - **Notification Services:** SMS or email services for order updates.
10) **Documentation**
    - **Microsoft Word:** Microsoft Word for writing and organizing project documentation, including reports and design details.

2. Background

The way people order food has evolved significantly with the rise of online platforms. In many parts of the world, people now prefer ordering food online due to its convenience, speed, and variety (Restolabs, 2024). However, in Nepal, many food delivery services are still in the early stages of development, and existing platforms often face issues such as limited options, complicated user interfaces, and slow delivery times.

This project aims to develop "Malt," a user-friendly web application that will provide an easy and efficient way for customers to order food online. The goal is to offer a wide selection of restaurants, a smooth and hassle-free ordering process, and reliable delivery services. The platform will allow users to browse menus, customize orders, and make payments securely, all from the comfort of their homes.

"Malt" will help to bridge the gap in the food delivery market by addressing common challenges and providing an improved online food ordering experience in Nepal. The project will focus on creating a platform that is easy to use, fast, and accessible to a wide range of users, making food ordering a simple and enjoyable experience.

2.1 Review of Journals/Articles

➢ **An empirical study of online food delivery services from applications perspective**

The study *"An Empirical Study of Online Food Delivery Services from Applications Perspective"* examines the rise and impact of online food delivery apps in the food industry (R. Ramesh, et al., 2021). The key highlights of the documents are explained below:

- **Growth and Impact**: Online food delivery services have shifted from telephone-based ordering to digital platforms, offering convenience and accessibility that have driven increased adoption and industry growth.
- **Theoretical Framework**: Consumer attitudes toward online food shopping are influenced by perceived social norms, compatibility, complexity, and risk.
- **E-service Quality**: Key to the success of online food delivery is the functionality of websites, accuracy of deliveries, and security of data, which bolster customer loyalty.
- **Business Models**:
    - **Order and Supply Model**: Collaborates with logistics firms for delivery.
    - **Integrated Model**: Manages own apps and delivery systems for enhanced efficiency and personalized service.
- **Adoption Factors**: Consumer decisions are shaped by performance expectancy, social influence, hedonic motivation, price value, and online reviews.
- **Recommendations**: Companies should expand into underserved areas, improve delivery efficiency, and create more user-friendly interfaces to boost adoption.
- **Conclusion:** Online food delivery services are reshaping consumer behaviors and restaurant strategies, highlighting the importance of continuous innovation and personalized experiences.

- **Designing Web-based Food Ordering Information System in Restaurant**

  This article discusses the design and implementation of a web-based food ordering system for restaurants. The study aims to create an application that allows customers to place orders online, reducing the need to queue. The research involved direct observation, interviews with customers, and data collection on existing ordering systems. The results showed that while there is some doubt about the effectiveness of the system, there is significant enthusiasm among customers and employees for its implementation (S M Noersidik & L Warlina , 2018).

- **Online Food Ordering System**

  The paper, *"Online Food Ordering System,"* presented at the International Conference on Emanations in Modern Engineering Science & Management explores the development and benefits of an online food ordering system for restaurants and customers (Dhakulkar, et al., 2018). The key highlights of the documents are explained below:

  - **Introduction:**
    The online food ordering system simplify ordering processes, reduce operational costs, and expand restaurant reach. It meets the needs of modern customers seeking convenience and efficiency, addressing challenges like long waiting times and limited food visibility in traditional methods.
  - **Objective:**
    Provide a web-based platform for restaurants to manage orders and menus. Enhance customer experience by allowing users to browse menus, place orders, and track deliveries with ease.
  - **Features:**
    Web Ordering System enables customers to select items, add them to a cart, and confirm orders. Menu Management allows restaurants to update food items, prices, descriptions, and images. Order Retrieval System simplifies order management for restaurants, ensuring timely and accurate processing.

- **Advantages:**
  Reduces labor costs and operational overhead for restaurants. Improves order accuracy, customer satisfaction, and business scalability. Supports small-scale and established businesses in supporting digital platforms for growth.
- **Conclusion:**
  The system provides a streamlined, efficient solution for online food delivery, enhancing customer convenience and aiding restaurants in managing operations effectively.

➢ **A Study on the Effectiveness of Online Food Applications on Registered Restaurants**

The study *"A Study on the Effectiveness of Online Food Applications on Registered Restaurants"* investigates the impact of online food applications on restaurant operations (Abraham, 2021). The key highlights of the documents are explained below:

- **Introduction:**
  The rise of the internet has revolutionized various industries, including food services. Online food applications enable restaurants to expand their reach beyond traditional physical locations.
- **Scope and Methodology:**
  The study focuses on registered restaurants in Kerala, analyzing cost management, sales impact, and corporate image improvement due to online platforms. Data collection involved surveys using questionnaires and secondary sources like publications and reports.

- **Findings:**
  Restaurants, particularly smaller ones, have seen improved sales and profitability by using platforms like Zomato and Swiggy. Online delivery systems have increased service costs but are offset by higher revenues and customer base expansion. Zomato is the preferred platform due to its lower commission rates and promotional strategies. Adoption of online platforms has enhanced corporate image and overall business output.
- **Suggestions:**
  Ensure timely updates and upgrades of applications. Improve communication between app executives and restaurant owners. Train delivery personnel for better punctuality and accuracy. Enhance app usability, including payment systems and order tracking.
- **Conclusion:**
  Online food applications are critical for modern restaurants to remain competitive. These platforms increase accessibility and convenience for customers and serve as vital tools for business expansion and brand-building.

2.2 Comparison with Similar Applications

The following are the similar applications that are comparable to the one I aim to develop:

- **Foodmandu**

Figure 1: Foodmandu logo

Foodmandu is a **food delivery application** primarily operating in **Nepal**. It is a popular platform that connects customers with restaurants, enabling them to order food online and have it delivered to their doorsteps.

Application Link: https://foodmandu.com/

- **Pokhara Food Delivery**

Figure 2: Pokhara Food Delivery logo

The **Pokhara Food Delivery** application is a local **food delivery service** operating primarily in **Pokhara, Nepal**. It is designed to connect residents and visitors in Pokhara with nearby restaurants, providing them with a convenient way to order and enjoy food at their location.

Application link: https://apps.pokharafooddelivery.com/

- **Bhojdeals**

Figure 3: Bhojdeals logo

Bhojdeals is a **food delivery application** that operates primarily in **Nepal**, targeting cities like Kathmandu and Pokhara. It is unique in combining food delivery services with special discounts and offers.

Application link: https://www.bhojdeals.com/

- **FoodMood**

Figure 4: Foodmood logo

The **FoodMood** application is a **food delivery platform** operating primarily in **Nepal**, providing users with a convenient way to order food online from local restaurants. Its features focus on improving the food ordering experience by offering a wide variety of cuisines.

Application Link: https://foodmood.com.np/

2.3 Comparison Table

Table 1: Comparison Table

Features	**Malt** (My Application)	**Food Mandu**	**Food Mood**	**Bhoj Deals**	**Pokhara Food Delivery**
User Registration & Login	Yes	Yes	Yes	Yes	Yes
Food Image	Yes	No	No	No	No
Restaurant Listings	Yes	Yes	Yes	Yes	Yes
Food Description	Yes	Yes	No	No	No
Search and Filters	Yes	Yes	Limited	Yes	No
Add to Cart Functionality	Yes	Yes	Yes	No	Yes
Order Summary	Yes	Yes	Yes	Yes	Limited
Order Confirmation	Yes	Yes	Yes	Yes	Yes
Order Tracking	Yes	Yes	No	No	No
Favorites Feature	Yes	No	No	No	No
Promote Offers/Events	Yes	Yes	No	Yes	No
Responsive Design	Yes	Yes	Yes	Limited	Yes
Live Notifications	Yes	Yes	Yes	Limited	No

3. Methodology

A software development methodology is a framework used to structure, plan, and manage the software development process. Its goal is to ensure the delivery of software in accordance with project specifications, within time and budget, and with minimized project risks (Nikitin, 2024). Each methodology offers a unique approach to managing the software development life cycle, defining project steps, roles, responsibilities, activities, communication standards, and deliverables.

3.1 Different Methodologies

Below, I have explained some of the popular methodologies and models used for software development.

1) Waterfall model

The first SDLC methodology was the waterfall model. This model divides the software development process into many stages, each of which has a specific set of tasks and goals. Only until the prior phase is complete does the development of the next phase begin. The waterfall model's phases are all fairly accurate and well defined as a result of this nature (Gallagher, et al., 2019). It is known as the waterfall model because the phases flow from an upper level towards a lower level, just like a waterfall.

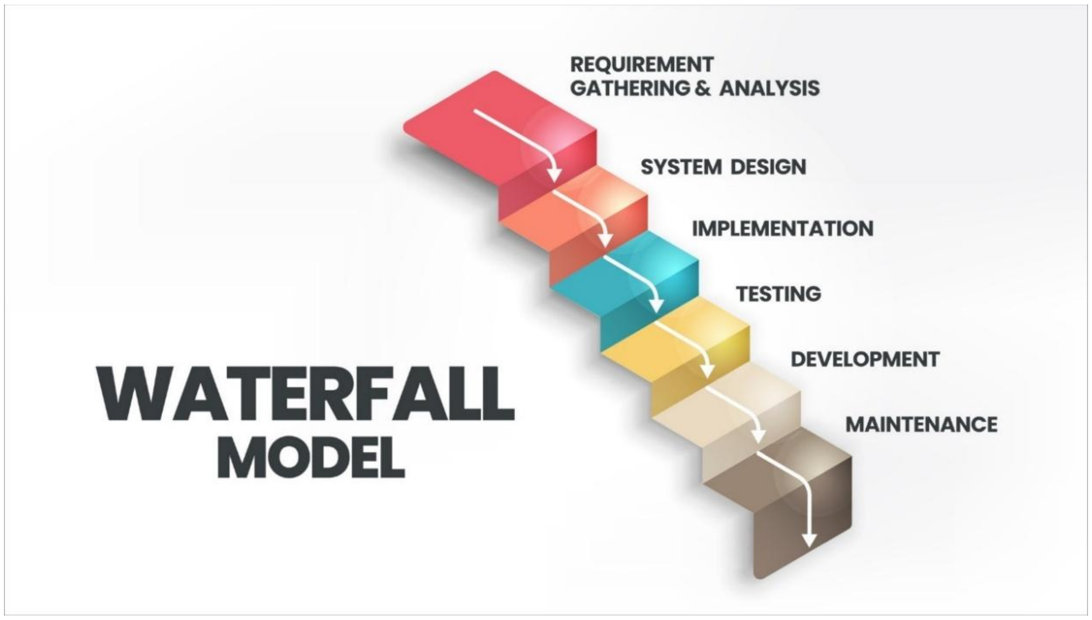


Figure 5: Waterfall Model

- **Advantages:**
  - This model is straightforward, understandable, and practical.
  - Phases are handled and finished one at a time in this model. They do not overlap.
  - This model functions effectively for smaller projects with well-defined criteria.

- **Disadvantages:**
  - It is quite challenging to fix anything that was poorly thought through at the idea stage after
  - an application has entered the later phases.
  - High levels of risk and uncertainty are involved in this model.
  - It is not a suitable model for complicated and object-oriented projects.

2) Prototype model

Prototype model is one of the SDLC models, where a prototype is constructed according to predetermined requirements. After the initial prototype has been reviewed and updated in response to consumer feedback, a complete prototype with all necessary functionality is then produced. The final prototype also serves as the foundation for the finished product (Rana, 2021).

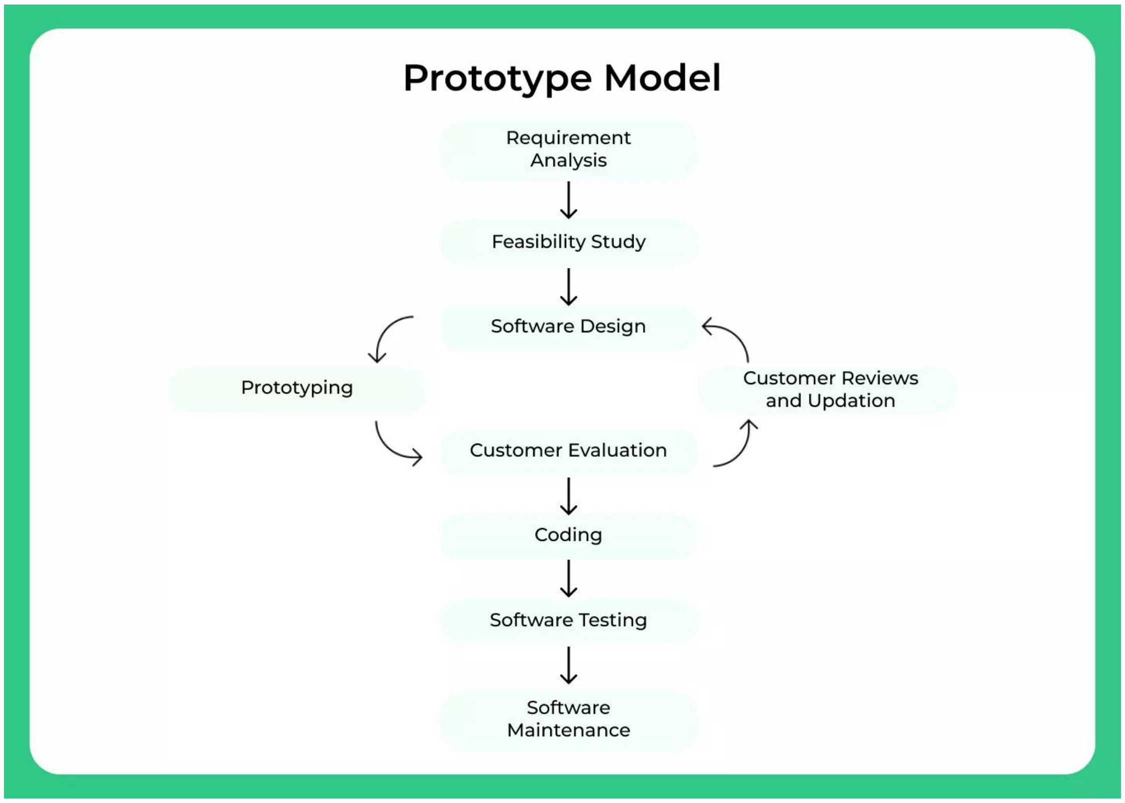


Figure 6: Prototype Model

- **Advantages:**
  - The development process is sped up by the quick client response.
  - Error and missing functionality are detected early.
  - This model is helpful when the client's needs are unclear.
- **Disadvantages:**
  - It takes time since several prototypes are made before the client is satisfied with the product.
  - This approach requires significant customer participation and interaction.
  - Practically, this process could make the system more complicated since the system's scope might go beyond initial projections.

3) Kanban

Kanban is a visual project management methodology that focuses on managing work by visualizing the flow of tasks, limiting work in progress (WIP), and ensuring continuous delivery (Bhaskar , 2024). It uses a Kanban board to visualize tasks, typically represented as cards that move through different stages such as "To Do", "In Progress", and "Done".

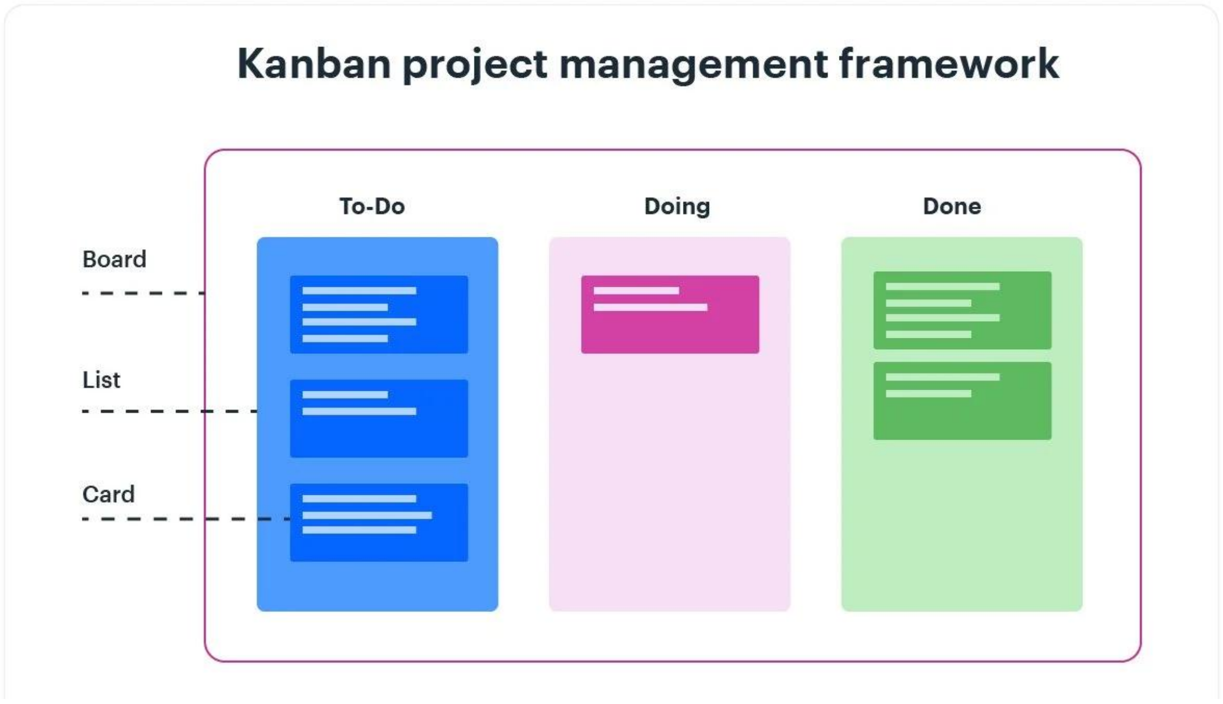


Figure 7: Kanban Framework

- **Advantages:**
    - Easy to track tasks and quick identification of bottlenecks.
    - No fixed iterations which allow teams to adjust priorities.
    - Limiting the work in progress (WIP) to helps team complete tasks faster.
    - It makes a steady flow of work without delays.
    - This model transparency promotes communication and early issue detection.

- **Disadvantages:**
    - The absence of defined roles can lead to confusion and inconsistency.
    - The ongoing work may result in expanding project scope.
    - This model requires continuous monitoring, and adjustments are needed to keep the process smooth.

3.2 Selected Methodology (Incremental Model)

After evaluating various software methodologies, the **incremental model** seems most suitable for this project. The Incremental Model in software engineering is a method where the project is divided into smaller, manageable parts or increments (Sachan, 2024). Each increment goes through the processes of gathering requirements, designing, testing, and implementation in this approach. These increments are built upon one another, gradually adding functionality to the software until the final product is complete.

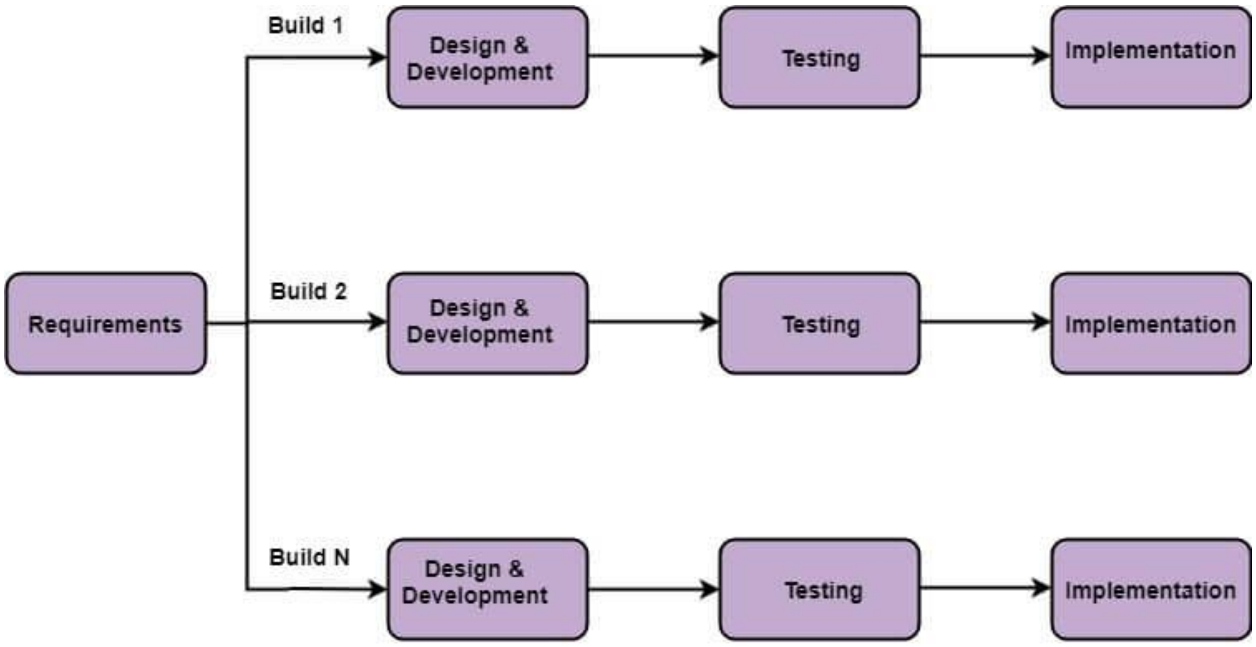


Figure 8: Incremental model

The **reason for choosing incremental model** is because it allows for adjustments at any point in the development process. If something needs to be added, it can be included in the next version or increment. Each iteration adds new functionality, allowing for the completion of essential features first. To achieve the final product, the project must implement several functionalities. Thus, completing a few features in each iteration is the most effective approach. Here are some of the **advantages** of using Incremental model:

- It is easy for breakdown of tasks because of divide and conquer approached used.
- It is good to use when requirements are known up-front.
- It is good to use when projects having lengthy developments schedules.
- It generates working software quickly and early during the software life cycle.
- It is easy to manage risk because of iterations.

According to my plan, the project will be completed in **three increments.** The initial requirement gathering must be finished before any increments can begin. The first increment will include the completion of the prototype design, database design, user login and registration page, home page, and the release of the initial version. The second version will be released after the cart page, user profile page, restaurant detail page, and payment gateway integration have been built in the second increment. The admin panel will be developed in the final increment. Then, the full version will be released along with the final documentation.

The various phases of the incremental model that **I will be following** while developing the application are as follows:

i. **Requirement Analysis:** First, I will gather all the necessary details about what the web app needs to do. This includes features like user registration, browsing menus, placing orders, and payment options. I will break these requirements down into smaller, manageable parts that I can develop one by one.

ii. **Design and Development:** Once I have a clear set of requirements, like the user registration feature, I will move on to design. I will plan out how this feature will look and function, including the layout and user interactions. After that, I will start coding to develop the feature. I will repeat this process for each feature.

iii. **Testing:** After developing a feature, such as user registration, I will thoroughly test it to ensure that it works as expected. This includes checking for bugs and making sure all data is correctly handled. If there are any issues, I'll fix them before moving forward.

iv. **Implementation:** Once a feature is tested and working properly, I'll implement it by adding it to the live application. This means it's now ready for users to interact with. I'll then move on to the next feature, like menu browsing, and continue the process until the entire app is complete.

3.3 Work Break Down Structure (WBS)

A Work Breakdown Structure (WBS) is a project management tool that breaks down a project into smaller, more manageable parts (Organ, 2024).

- Here is a **previous** Work Break Down Structure of my Project:

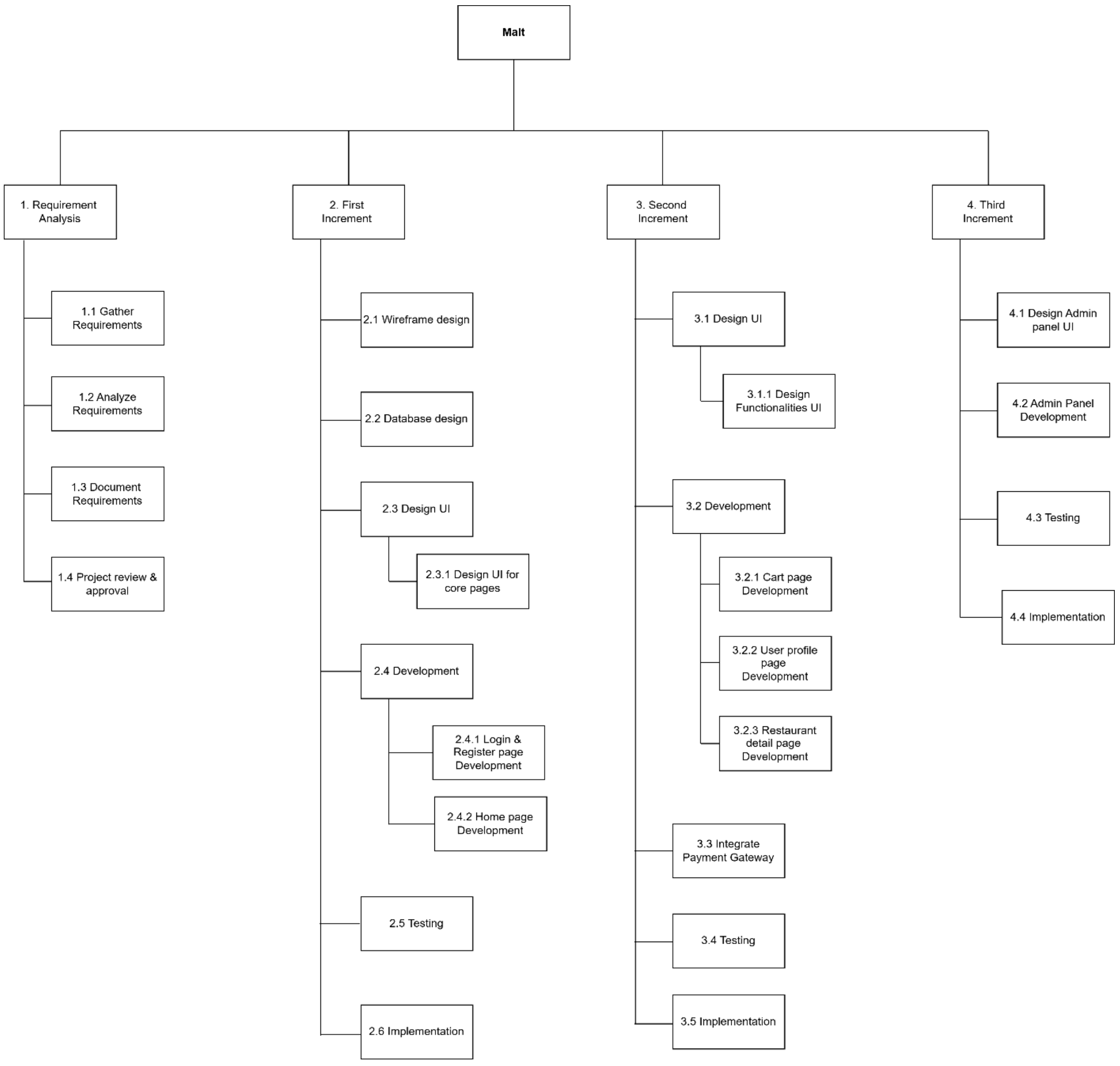

Malt
1. Requirement Analysis
1.1 Gather Requirements
1.2 Analyze Requirements
1.3 Document Requirements
1.4 Project review & approval
2. First Increment
2.1 Wireframe design
2.2 Database design
2.3 Design UI
2.3.1 Design UI for core pages
2.4 Development
2.4.1 Login & Register page Development
2.4.2 Home page Development
2.5 Testing
2.6 Implementation
3. Second Increment
3.1 Design UI
3.1.1 Design Functionalities UI
3.2 Development
3.2.1 Cart page Development
3.2.2 User profile page Development
3.2.3 Restaurant detail page Development
3.3 Integrate Payment Gateway
3.4 Testing
3.5 Implementation
4. Third Increment
4.1 Design Admin panel UI
4.2 Admin Panel Development
4.3 Testing
4.4 Implementation

Figure 9: Work Break Down Structure (Previous)

➢ Here is a **revised** Work Break Down Structure of my Project:

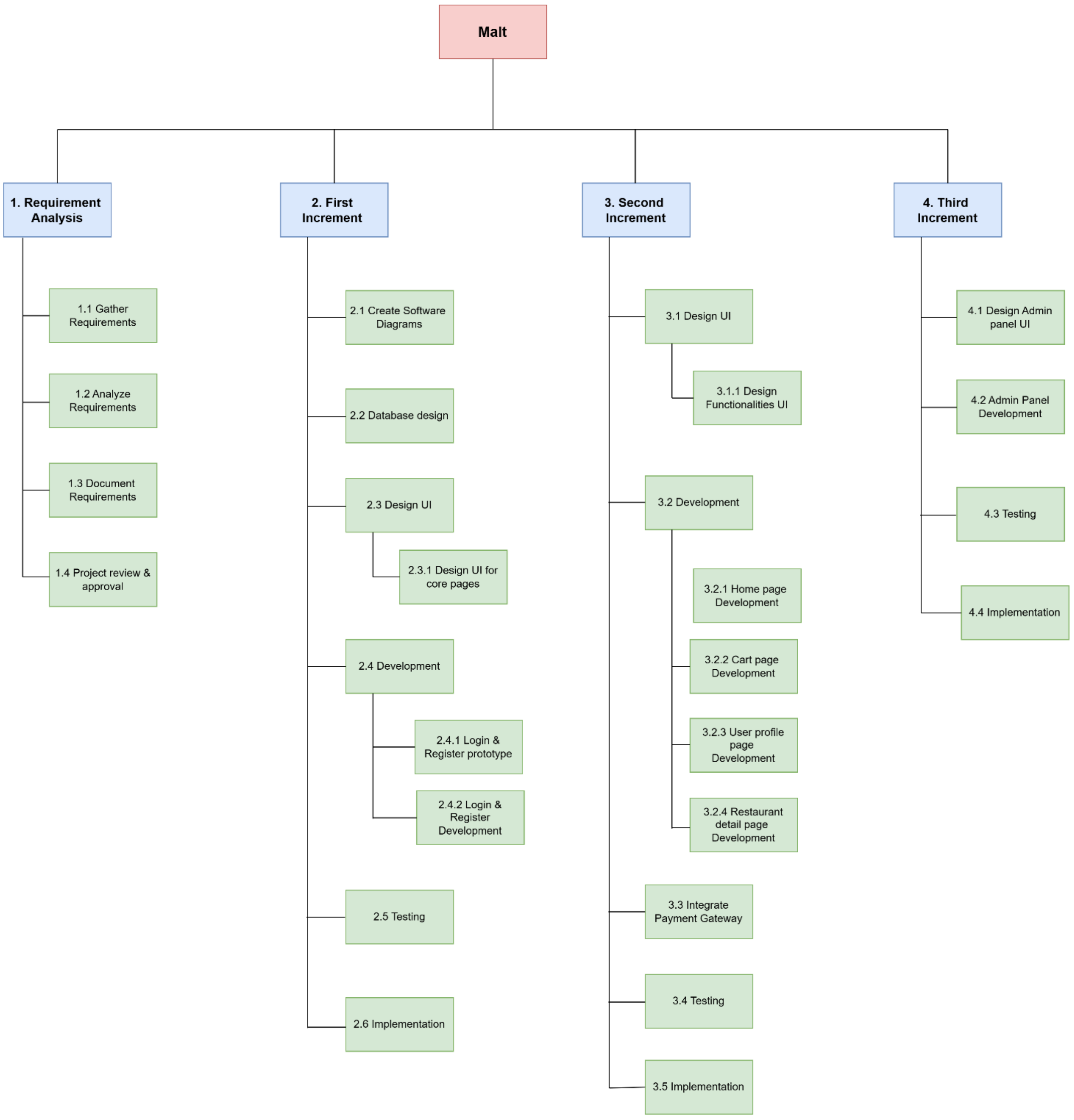

Malt
1. Requirement Analysis
1.1 Gather Requirements
1.2 Analyze Requirements
1.3 Document Requirements
1.4 Project review & approval
2. First Increment
2.1 Create Software Diagrams
2.2 Database design
2.3 Design UI
2.3.1 Design UI for core pages
2.4 Development
2.4.1 Login & Register prototype
2.4.2 Login & Register Development
2.5 Testing
2.6 Implementation
3. Second Increment
3.1 Design UI
3.1.1 Design Functionalities UI
3.2 Development
3.2.1 Home page Development
3.2.2 Cart page Development
3.2.3 User profile page Development
3.2.4 Restaurant detail page Development
3.3 Integrate Payment Gateway
3.4 Testing
3.5 Implementation
4. Third Increment
4.1 Design Admin panel UI
4.2 Admin Panel Development
4.3 Testing
4.4 Implementation

Figure 10: Work Break Down Structure (Revised)

The Work Breakdown Structure (WBS) for the **'Malt'** project provides a well-organized approach to incremental development. By dividing the project into different phases such as Requirement Analysis, First, Second, and Third Increment the WBS ensures clear planning, efficient execution, and complete testing.

3.4 Milestone

A milestone is a specific point within a project’s life cycle used to measure the progress toward the ultimate goal. Milestones in project management are used as signal posts for a project's start or end date, submission of a major deliverable, etc (shopdev, 2023).

➢ Here is a **previous** milestone of my Project:

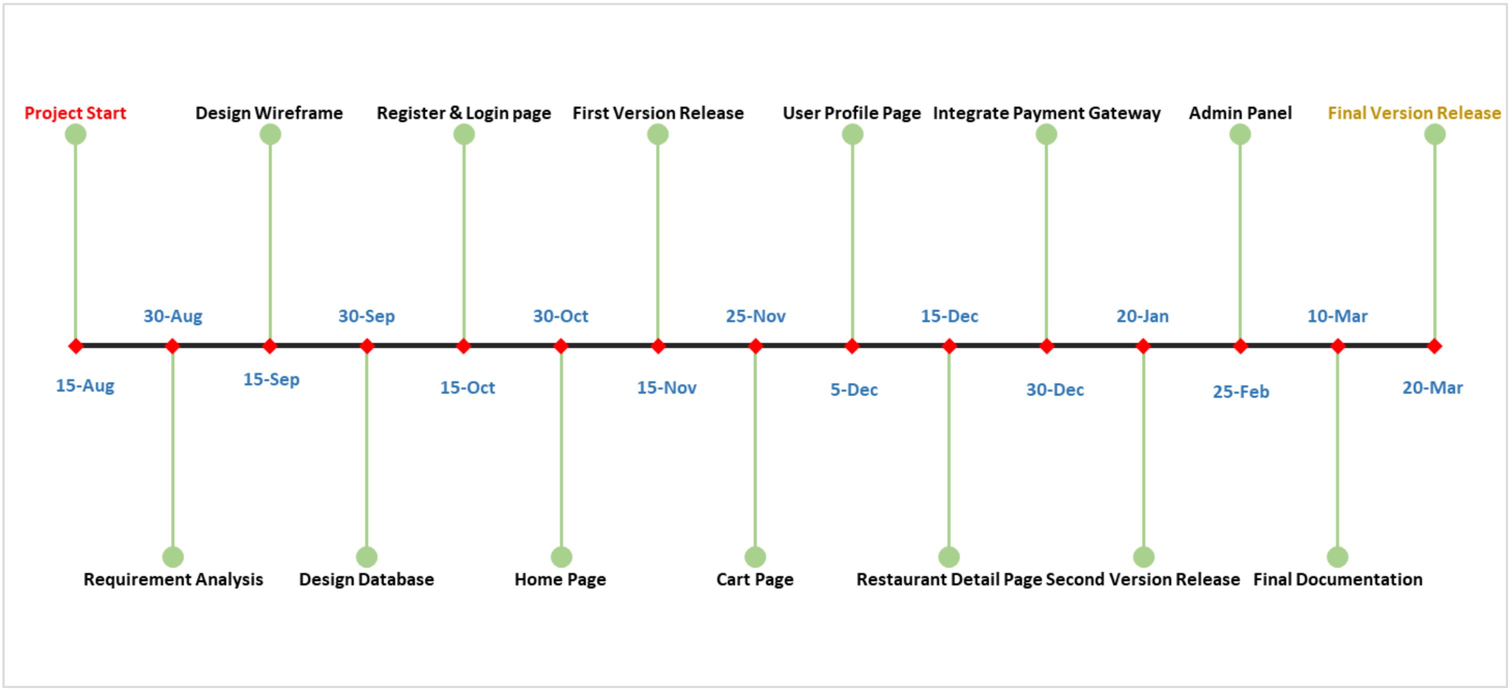


Figure 11: Milestones (Previous)

➢ Here is a **revised** milestone of my Project:

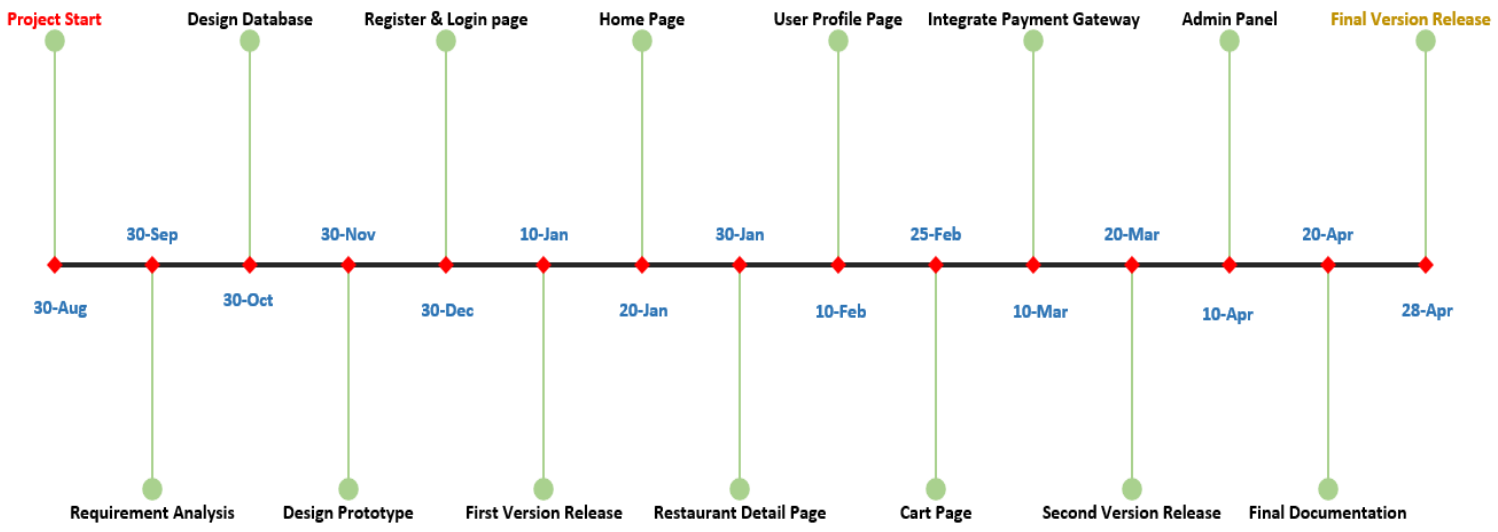


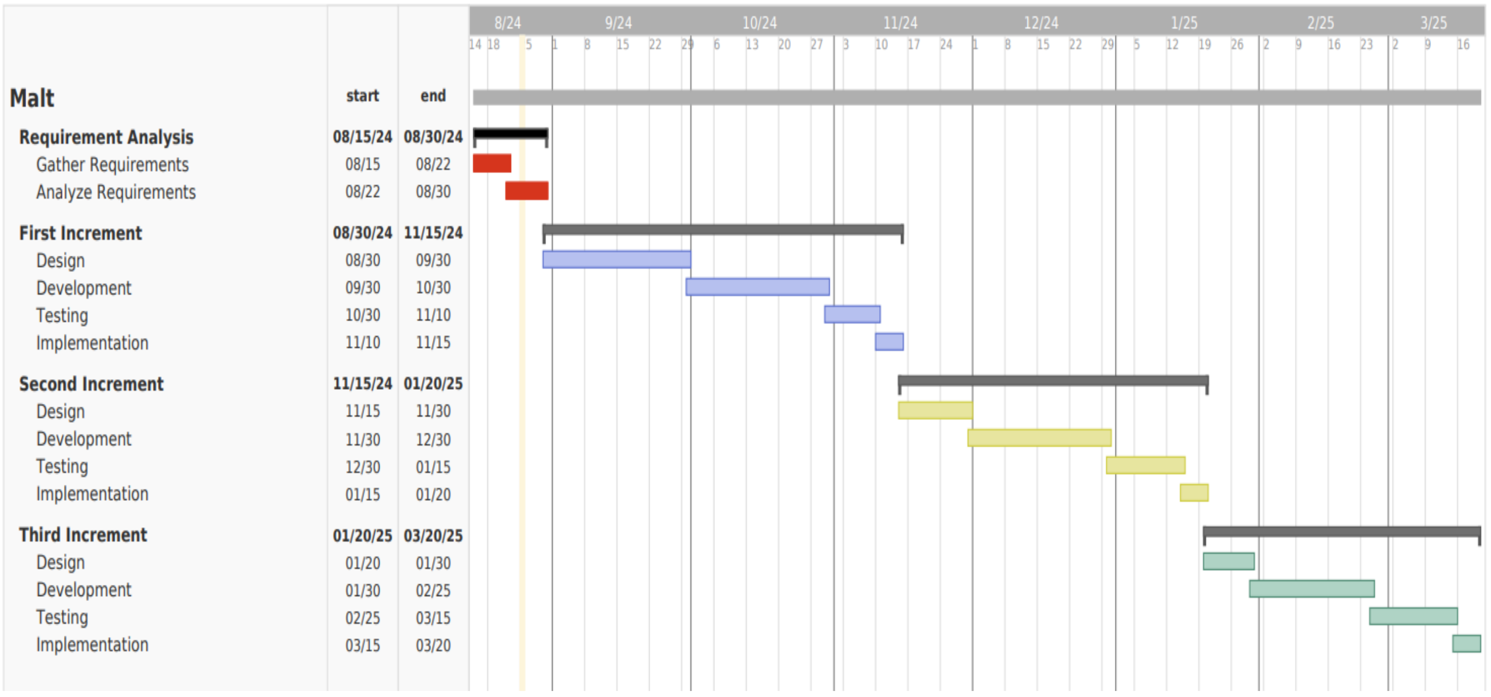
Figure 12: Milestones (Revised)

The above project **milestones** provide a clear roadmap for the development of the 'Malt' food ordering web application. Starting from **August 30, 2024** and concluding on **April 28, 2025** these milestones ensure a structured approach for completing each phase of the project.

3.5 Gantt Chart

A Gantt chart is a visual project plan that lists tasks and milestones on the vertical axis, with time plotted on the horizontal axis. Gantt charts are commonly used in project management to schedule, track, and communicate deliverables, deadlines, dependencies, and resource assignments (Adobe Communications Team, 2022).

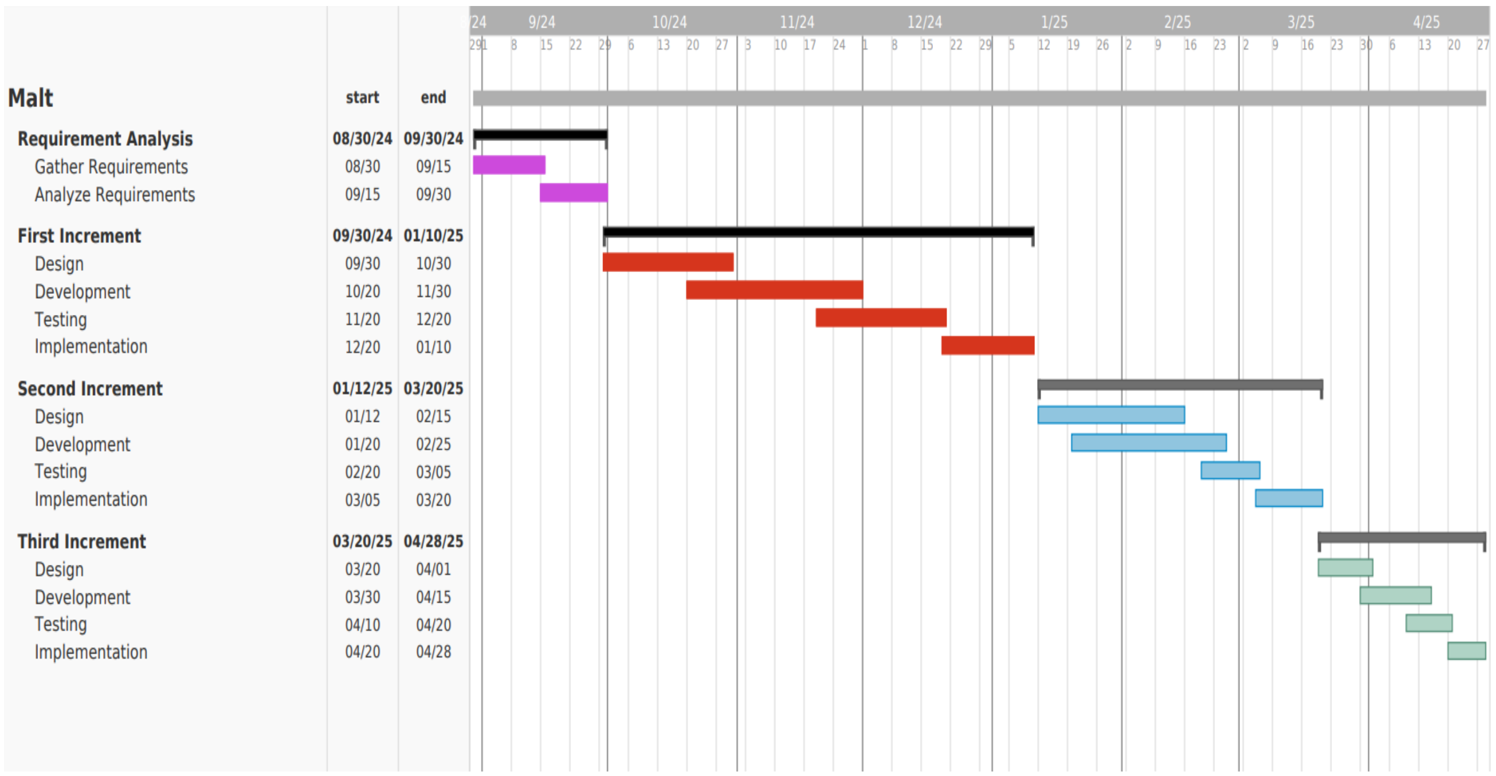
➢ Here is a **previous** Gantt Chart of my Project:



Malt	start	end
Requirement Analysis	**08/15/24**	**08/30/24**
Gather Requirements	08/15	08/22
Analyze Requirements	08/22	08/30
First Increment	**08/30/24**	**11/15/24**
Design	08/30	09/30
Development	09/30	10/30
Testing	10/30	11/10
Implementation	11/10	11/15
Second Increment	**11/15/24**	**01/20/25**
Design	11/15	11/30
Development	11/30	12/30
Testing	12/30	01/15
Implementation	01/15	01/20
Third Increment	**01/20/25**	**03/20/25**
Design	01/20	01/30
Development	01/30	02/25
Testing	02/25	03/15
Implementation	03/15	03/20

Figure 13: Gantt Chart (Previous)

➢ Here is a **revised** Gantt Chart of my Project:



Malt	start	end
Requirement Analysis	**08/30/24**	**09/30/24**
Gather Requirements	08/30	09/15
Analyze Requirements	09/15	09/30
First Increment	**09/30/24**	**01/10/25**
Design	09/30	10/30
Development	10/20	11/30
Testing	11/20	12/20
Implementation	12/20	01/10
Second Increment	**01/12/25**	**03/20/25**
Design	01/12	02/15
Development	01/20	02/25
Testing	02/20	03/05
Implementation	03/05	03/20
Third Increment	**03/20/25**	**04/28/25**
Design	03/20	04/01
Development	03/30	04/15
Testing	04/10	04/20
Implementation	04/20	04/28

Figure 14: Gantt Chart (Revised)

The above Gantt chart shows a clear step-by-step plan for building the project using the incremental model. Each increment focuses on specific functionalities, allowing for continuous feedback and improvement. This approach helps to keep the project on track, ensuring all tasks are completed on time for a successful finish.

4. Work Done

The project has already been going on for a while. The incremental software development process was used to build the project. This approach involves building and delivering the project in manageable sections or increments. The project is divided into three increments, with each increments contributing progressively towards the final product. The first increment of the project is almost finished. Below, I have described the progress of the project, and the work completed till now.

4.1 Software Requirement Specification (SRS)

Malt is a web application where people can order food online. It aims to make ordering food easier for customers and for restaurants to manage orders. The Software Requirement Specification explains the features the Malt app will have as well as its intended use and users.

4.1.1 Intended Use and Users

➢ The intended use of the application are as follows:

1. **Convenient Food Ordering**: Users can browse menus, view details about dishes, and order food online without needing to visit the restaurant physically.
2. **Efficient Delivery Management**: Helps restaurants streamline order processing and delivery tracking.
3. **Customizable Menus**: Restaurants can manage their menus, add new dishes and update prices.
4. **User Profiles**: Enables users to save preferences, order history, and payment information for a personalized experience.
5. **Secure Transactions**: Supports multiple payment options to allow users to pay securely using various methods like digital wallets and cash on delivery.
6. **Promotions and Discounts**: Restaurants can promote events and discounts to retain customers and attract new ones.

7. **Real-Time Order Tracking**: Provides customers with real-time updates on the status of their orders.

- The intended users of the application are as follows:

1. Customers

Customers are the primary users of food-ordering applications, and their reasons for using these platforms include:

- **Convenience:**
  They can order food from the comfort of their homes or offices without needing to visit the restaurant.
- **Timesaving:**
  The process is faster than physically going to a restaurant, especially with features like saved addresses and payment details.
- **Variety:**
  Access to a wide range of cuisines and restaurants in one place.
- **Order Tracking:**
  Real-time updates on order preparation and delivery status provide transparency.

2. Restaurant Owners/Managers

Restaurant owners use food-ordering applications to expand their customer base and online operations.

- **Increased Reach:**
  They can reach customers who may not be aware of their restaurant.
- **Ease of Management:**
  The platform provides tools for managing menus, orders, and promotions easily.

- **Sales Growth:**
  Online ordering opens up a new revenue channel, especially for delivery and takeout orders.

4. Admins (Platform Owners)

Admins use food-ordering applications to oversee and grow the platform.

- **Revenue Generation:**
  They earn commission from restaurants and delivery fees from customers.
- **Market Expansion:**
  The application enables partnerships with multiple restaurants and attracts more users.
- **Operational Efficiency:**
  Automated workflows reduce the need for manual interventions, making the business scalable.

4.1.2 System Features and Requirements

➢ **Functional Requirements**

The functional requirements for the application are as follows:

1. User Registration and Login

- **Secure Authentication:** Users can create an account using their email. They can log in securely after registering.
- **Password Reset:** If users forget their password, they can reset it through an easy process, like receiving a link via email.

2. Restaurant Listings

- **Restaurant Information:** The application shows a list of restaurants with important details, like:
  - Name of the restaurant.
  - Location (address or area).

3. Menu Display

- **Detailed Food Menus:** Each restaurant's menu is displayed with:
  - Food items and their descriptions.
  - Prices for each dish.
  - Images to help users decide what to order.

4. Search and Filters

- **Search Option:** Users can search for specific restaurants or dishes by typing keywords.
- **Filter Options:** Users can narrow their search based on:
  - Price range
  - Food Category

5. Order Placement

- **Add-to-Cart:** Users can add multiple food items to a cart for a single order.
- **Order Summary:** Before placing the order, users can review the items they added, along with the total cost.

6. Basic Payment Options

- **Payment Gateways:** The application supports easy payment methods, including:
  - E-Sewa (Digital wallet in Nepal)
  - Cash on delivery for convenience.

This ensures users have multiple ways to pay for their order.

7. Order Confirmation

- **Instant Notification:** After placing an order, users immediately get a confirmation:
  - Inside the app (e.g., "Your order is confirmed").
  - Via email or SMS with order details.

8. User Profile Management

- **Profile Editing:** Users can update their personal information, such as name or phone number.
- **Address Management:** Add or change saved delivery addresses.
- **Payment Methods:** Save preferred payment options for faster checkout.

9. Responsive Design

- Optimized for both desktop and mobile devices.

10. Order Tracking

- Live status updates (e.g., order accepted, preparing, out for delivery, delivered).

11. Add to Favorites

- Users can save their favorite restaurants for quick access.

12. Promote events or offers

- Restaurants can promote their special events and offers.

13. Restaurant Dashboard

- Dedicated panel for restaurant owners to manage menus and orders.

➢ Non-Functional Requirements

The non-functional requirements for the application are as follows:

1. **Response Time**:
   The application web pages should load within 3 seconds for a seamless user experience and search results must be displayed within 2 second.
2. **Access Control**:
   Only authorized admins should be allowed to manage restaurants and menus. Regular users should not have access to these controls.
3. **Data Storage**:
   The application should securely store information about users, restaurants, menus, orders, and payments, ensuring no data is lost.
4. **Feature Updates**:
   The application must be designed so that adding new features or improving existing ones can be done easily without breaking the current functionality.
5. **Performance**:
   Pages should load quickly, with minimal delay, even on slower internet connections or mobile networks.
6. **Security**:
   User data, including payment details, passwords, and personal information, must be protected with proper encryption and secure coding practices.
7. **Cross-Platform Compatibility**:
   The application should work smoothly on different devices (e.g., smartphones, tablets, desktops) and operating systems like Android, iOS, and Windows because of its responsive design.
8. **Error Handling**:
   The app should show clear error messages if something goes wrong, like failed payments or network issues, without crashing.
9. **Version Control**: All code changes must be tracked and managed using a version control system (e.g., Git) to ensure efficient future collaboration, rollback options, and proper history tracking.

10. **Database Optimization**: The database should be designed and optimized for quick data retrieval and minimal latency, even when handling large datasets or complex queries.
11. **Code Quality**: The codebase should follow clean coding principles, be well-documented, and stick to standard guidelines to make it maintainable, readable, and error-free.

4.2 Use Case Diagram

The purpose of a use case diagram in UML (Unified Modeling Language) is to demonstrate the different ways that a user might interact with a system. It offers a broad overview of the system's functionality without going into the specifics of its implementation (Creately, 2022). In a use case diagram, the primary elements include actors (users or external systems), use cases (specific tasks or functionalities the system provides), and the relationships between them. The use case diagram for this project is given below:

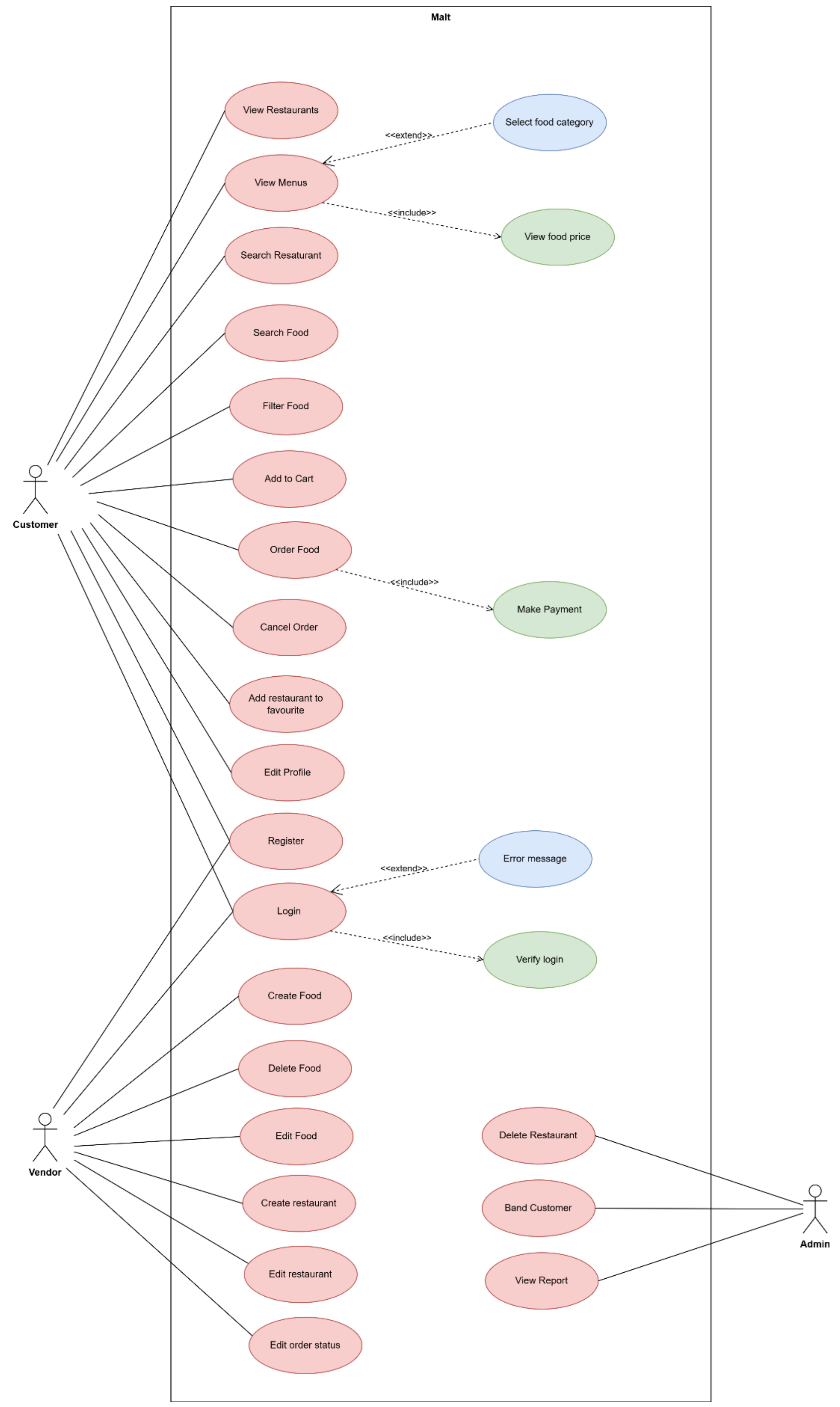


Figure 15: Use Case Diagram

➢ High level use case description for Login

Table 2: High level use case description for Login

Use case	Login
Actor	User
Description	Users provide login details, and a successful login redirects them to the home page.

➢ High level use case description for Order Food

Table 3: High level use case description for Order Food

Use case	Order Food
Actor	Customer
Description	Customers can view a variety of food items and place an order if they are interested.

- **Expanded use case description for Login**

Use case: Login

Actor: User

Description: Users provide login details, and a successful login redirects them to the home page.

Typical Course of Events:

Table 4: Expanded use case description for Login

Actor Action	System Response
1. User navigate to the login page.	2. System displays the login form requesting the user’s email and password.
3. User enters his/her login details and clicks the Login button.	4. System redirect the user to the home page.

Alternative Course:

Line 3 IF the login details are incorrect. The user receives login error messages.

- **Expanded use case description for Order Food**

Use case: Order Food

Actor: Customer

Description: Customers can view a variety of food items and place an order if they are interested.

Typical Course of Events:

Table 5: Expanded use case description for Login

Actor Action	System Response
	1. System displays the available food items and its details.
2. Customer select the food items they want to order.	3. System prompts the customer to specify the food quantity.
4. Customer enters the quantity.	5. System requests the customer to provide card details (e.g. Debit card and Credit card)
6.Customer pays the amount by credit/debit card.	7. System sends an order confirmation to the customer.

Alternative Course:

Line 2 The customer does not find a desirable food item. Use case ends.

4.3 Collaboration/Communication Diagram

Collaboration diagrams, also known as Communication Diagrams in UML, are used to illustrate how objects interact to carry out the behavior of a particular use case or a section of it. It is used to specify and clarify the roles of the objects involved in a particular flow of events of a use case (Carter, 2024). The collaboration diagram for login and order food use case are given below:

- **Login Collaboration Diagram**

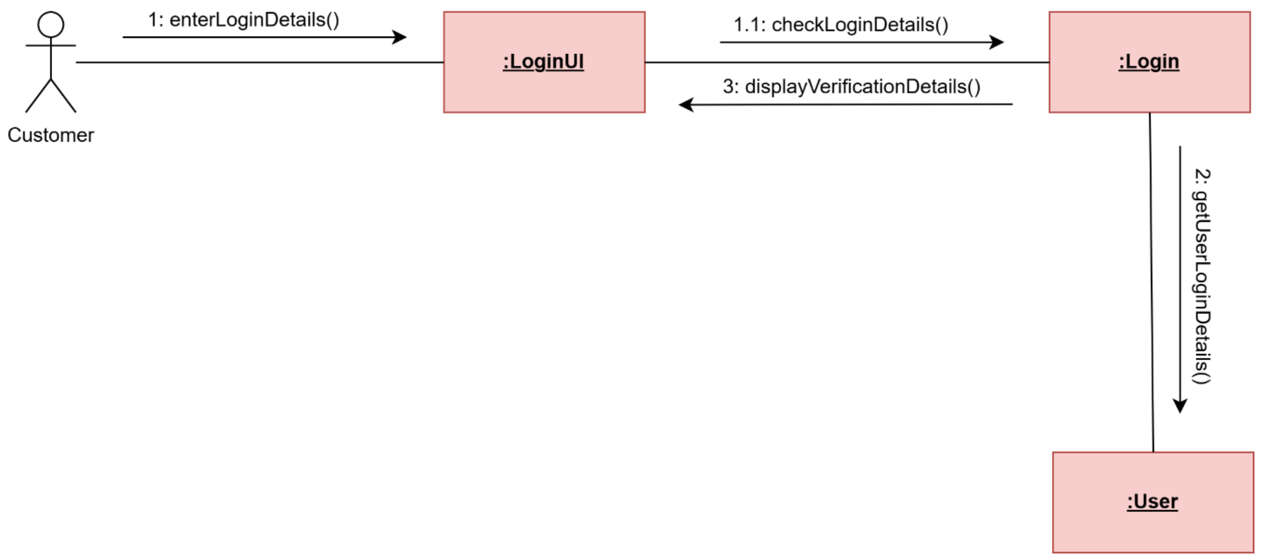


Figure 16: Login Collaboration Diagram

➢ **Order Food Collaboration Diagram**

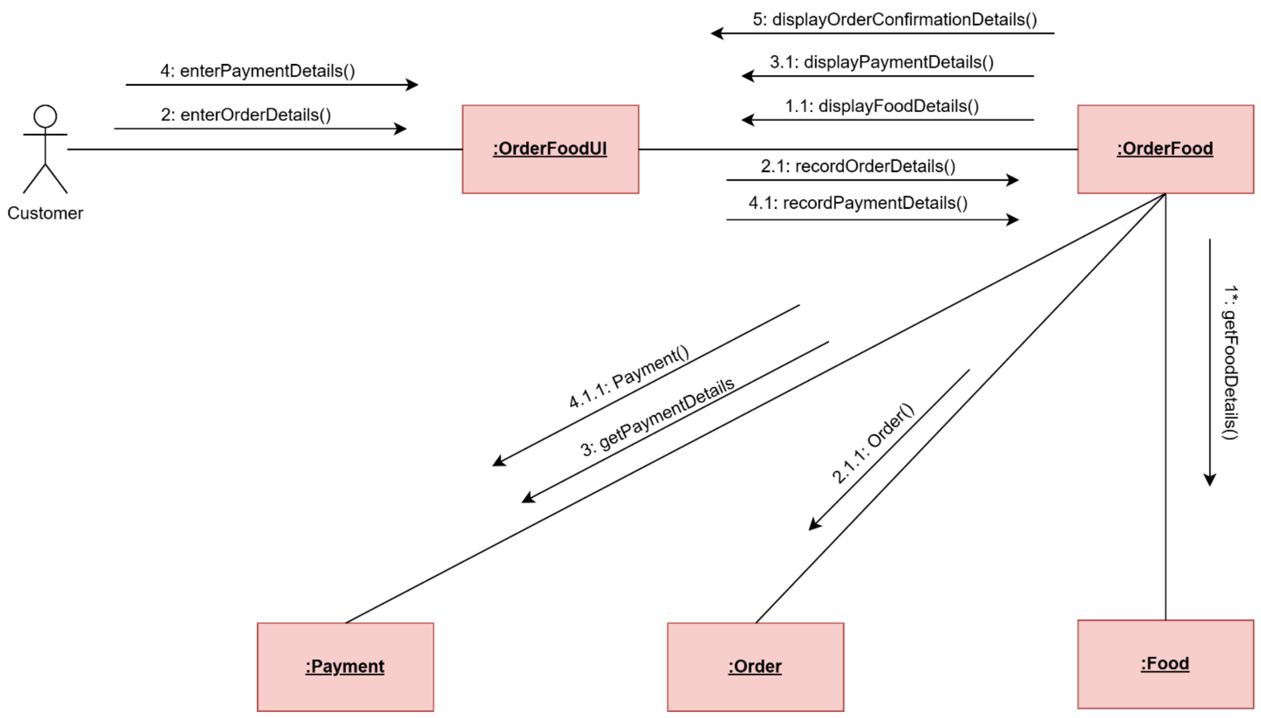


Figure 17: Order Food Collaboration Diagram

4.3 Sequence Diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios. These can help to predict how a system will behave and discover responsibilities a class may need to have in the process of modeling a new system (Medium, 2018). The sequence diagram for login and order food use case are given below:

- **Login Sequence Diagram**

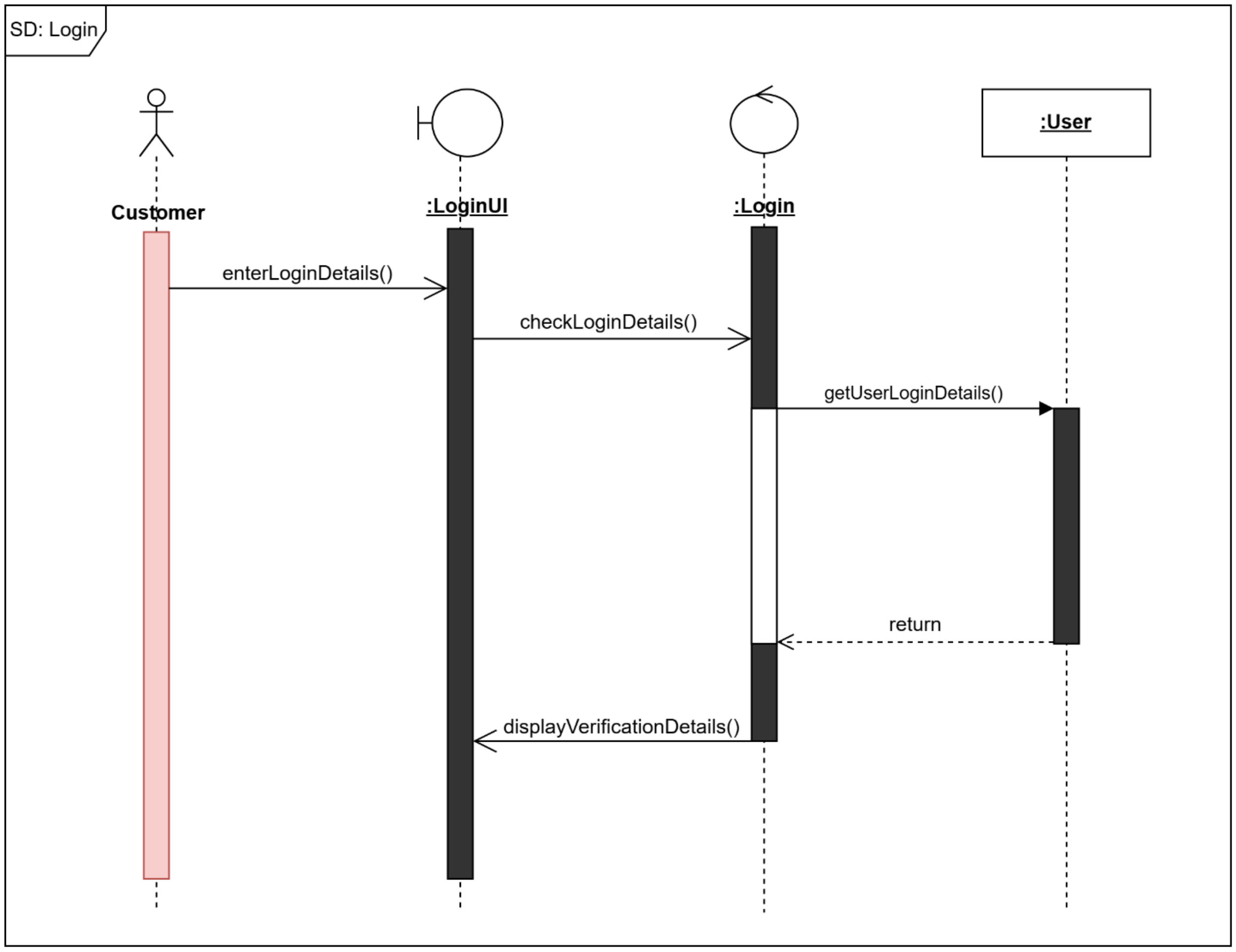


Figure 18: Login Sequence Diagram

➢ Order Food Sequence Diagram

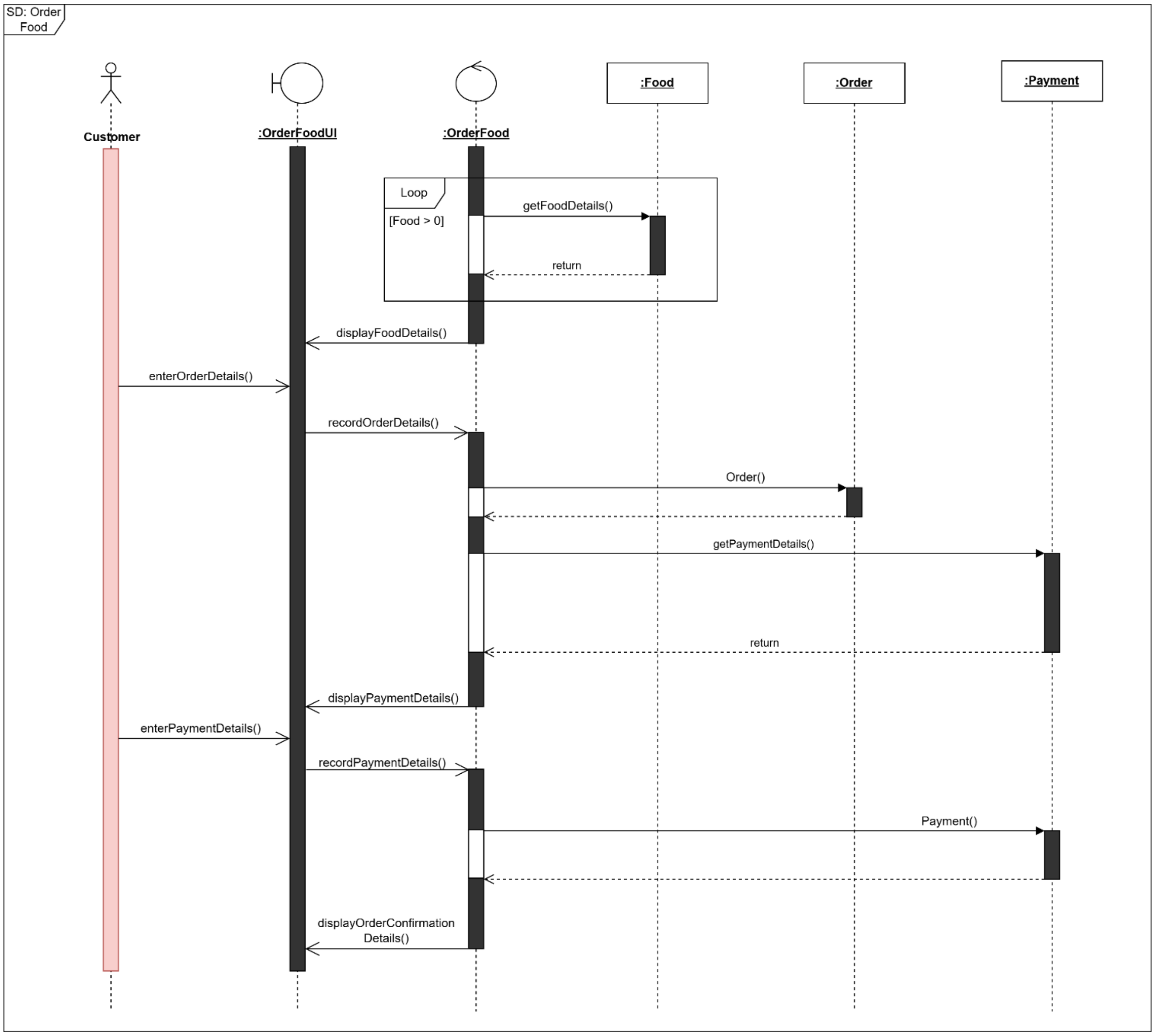


Figure 19: Order Food Sequence Diagram

4.4 Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) shows how data moves through a system or process. It simplifies complex systems, making them easier to understand. DFDs are useful in software development, system analysis, improving processes, and business management (Lindemulder, 2024). Data Flow Diagram levels 0, 1, and 2 for this project are given below.

4.4.1 DFD Level – 0

The context level diagram for the Malt food ordering web application is shown below:

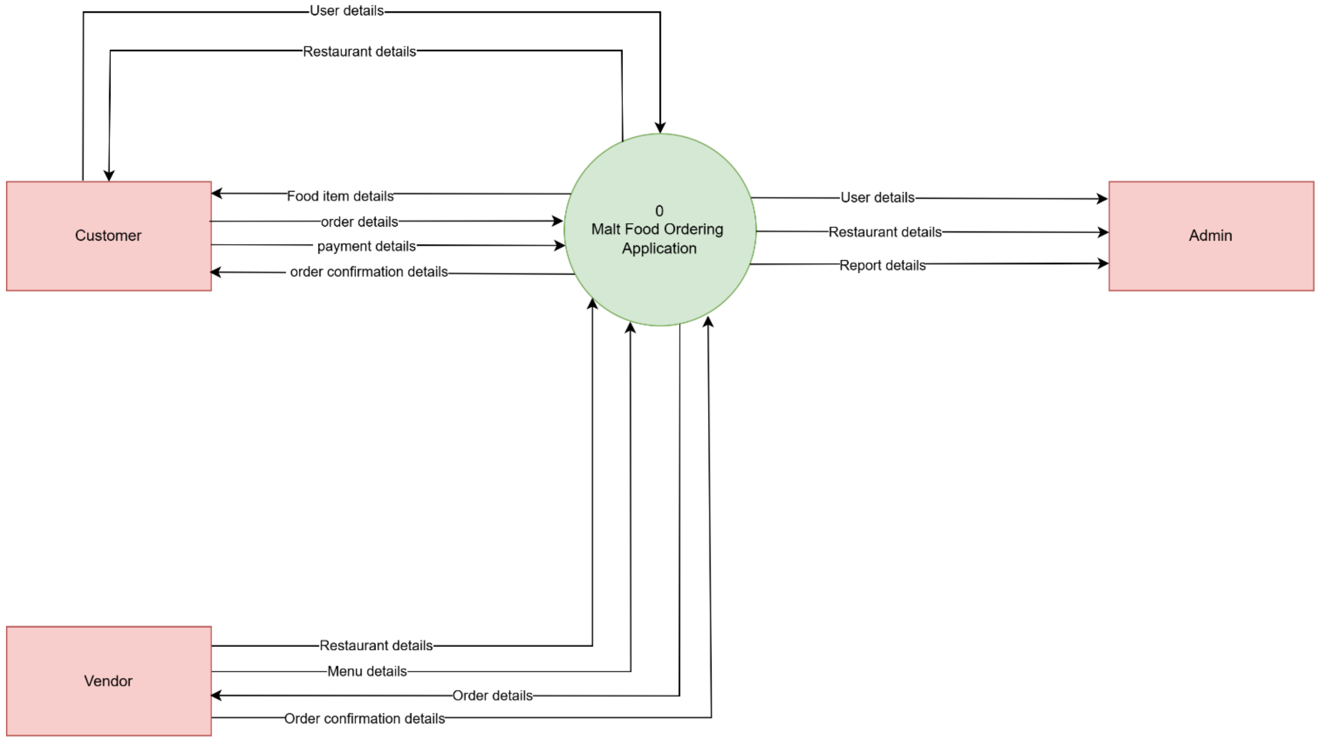


Figure 20: DFD Level – 0

4.4.2 DFD Level – 1

Level 1 DFDs are still a general overview, but they go into more detail than a context diagram. In level 1 DFD, the single process node from the context diagram is broken down into sub-processes. As these processes are added, the diagram will need additional data flows and data stores to link them together. The DFD level 1 for the Malt food ordering web application is shown below:

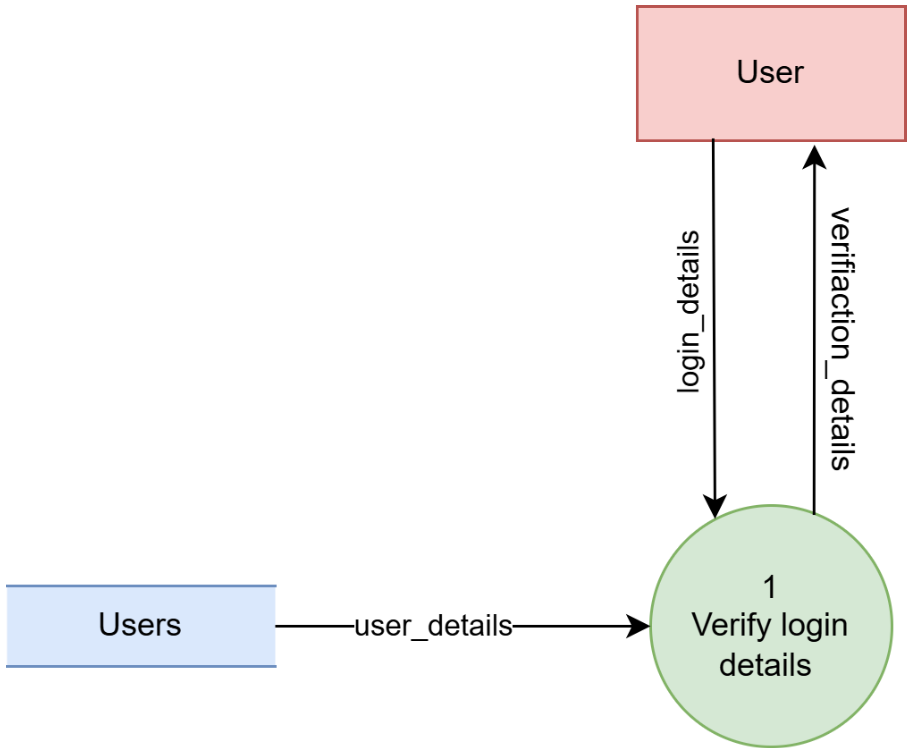


Figure 21: Login DFD Level – 1

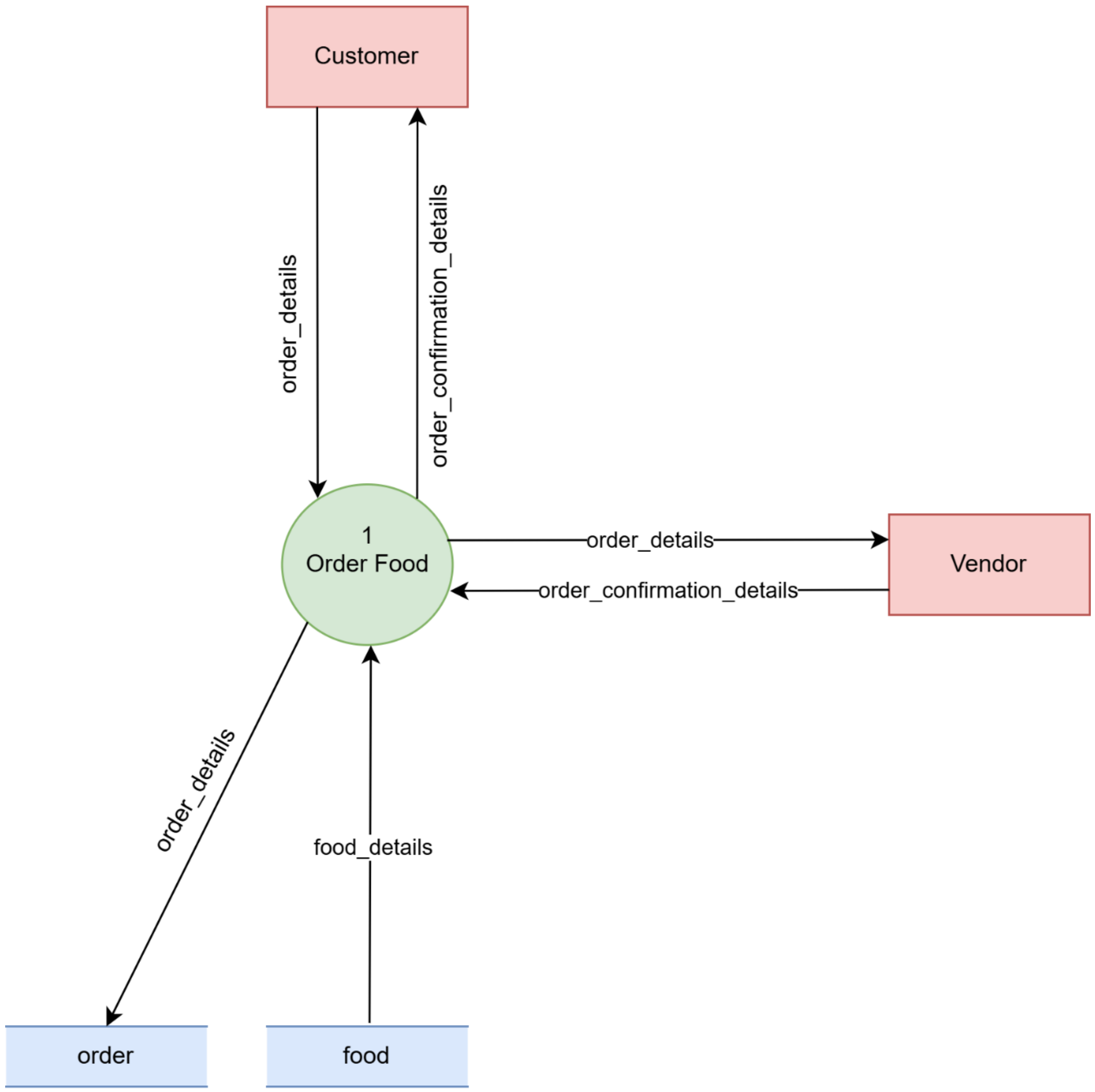


Figure 22: Order Food DFD Level – 1

4.4.3 DFD Level – 2

Level 2 DFDs break down the processes from Level 1 into even more detailed sub-processes. While Level 1 provides an overview of main activities, Level 2 dives deeper, showing specific tasks within those activities. At this level, you'll add more data stores and data flows to show how the processes interact with each other and the system. It makes the flow of data much clearer and helps identify any further details needed for development. The DFD level 2 for the Malt food ordering web application is shown below:

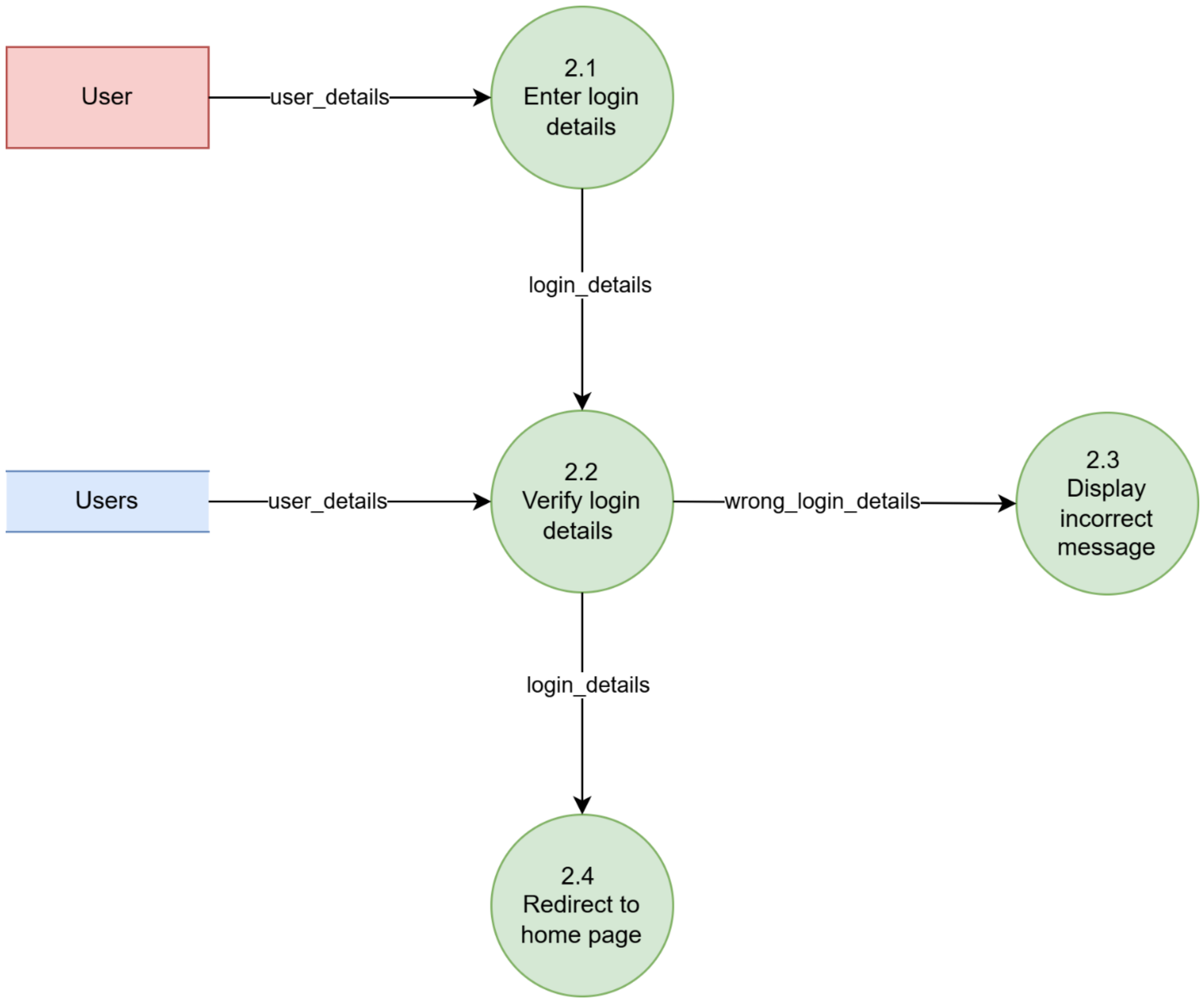


Figure 23: Login DFD Level – 2

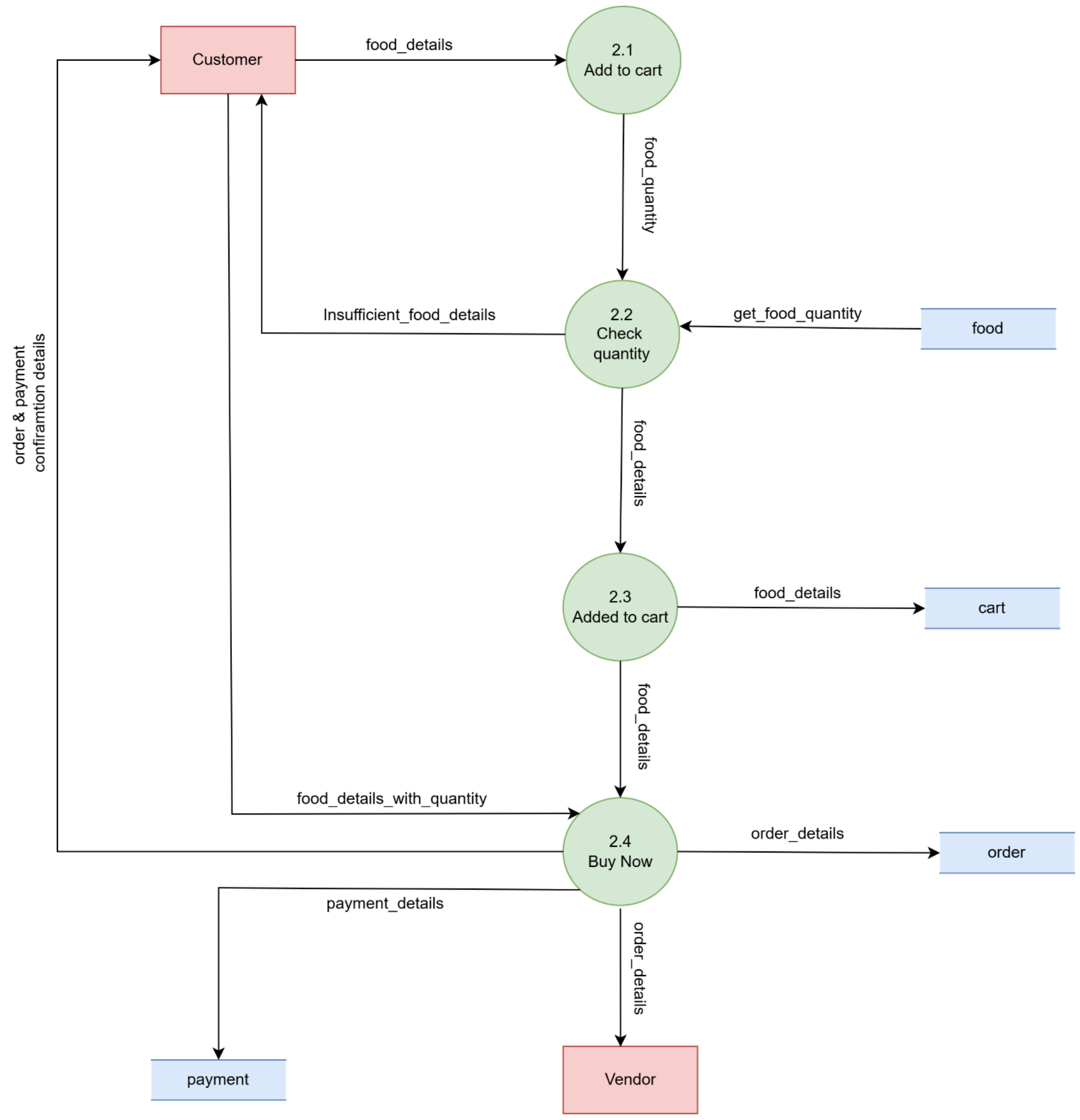


Figure 24: Order Food DFD Level – 2

4.5 Initial ERD

Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships (Brown, 2024). The Initial Entity Relationship Diagram for this project is shown below:

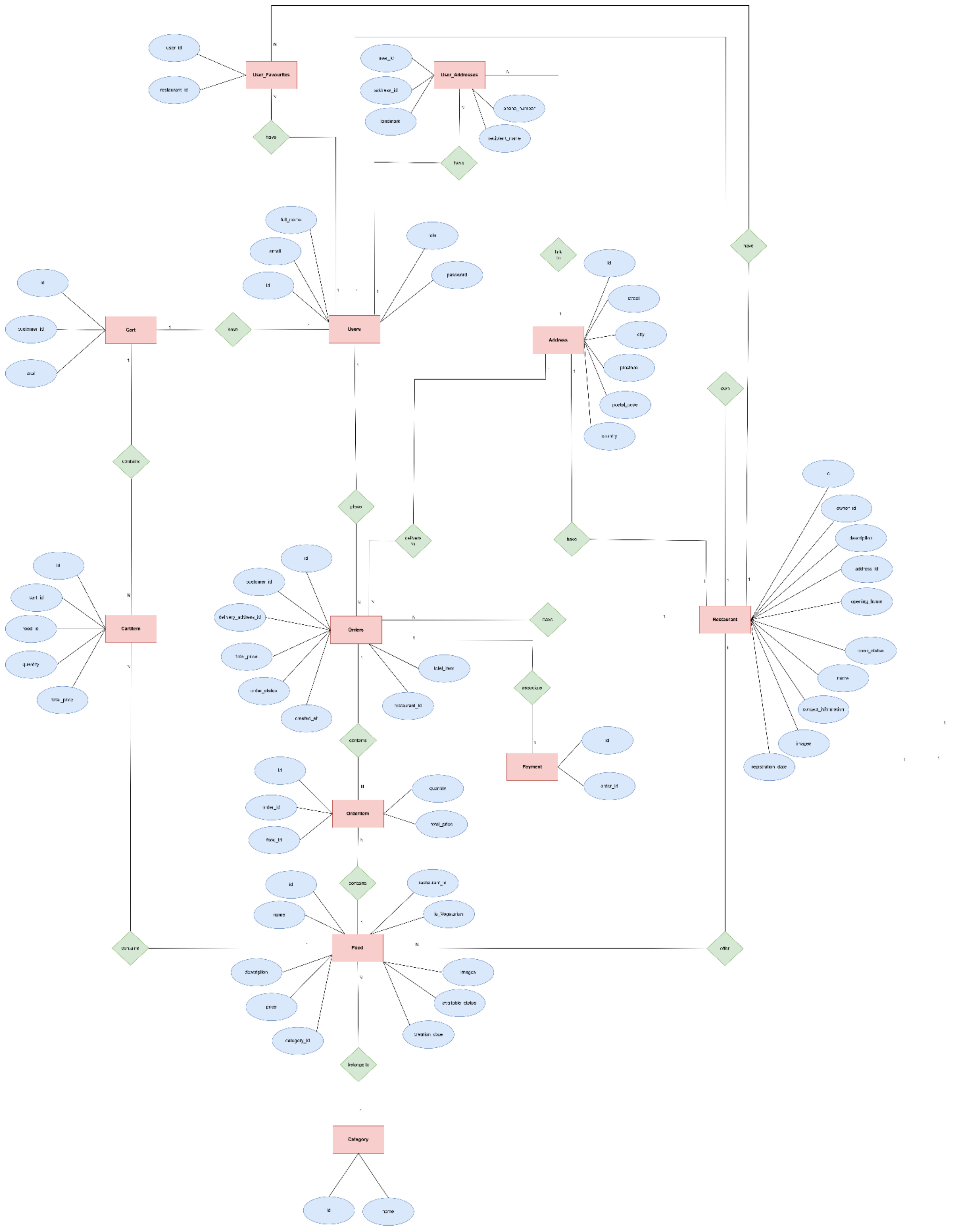


Figure 25: Initial ERD

4.6 Class Diagram

A class diagram is a UML diagram type that describes a system by visualizing the different types of objects within a system and the kinds of static relationships that exist among them. It also illustrates the methods and attributes of the classes (Creately, 2024). Class Diagram for this project is shown below:

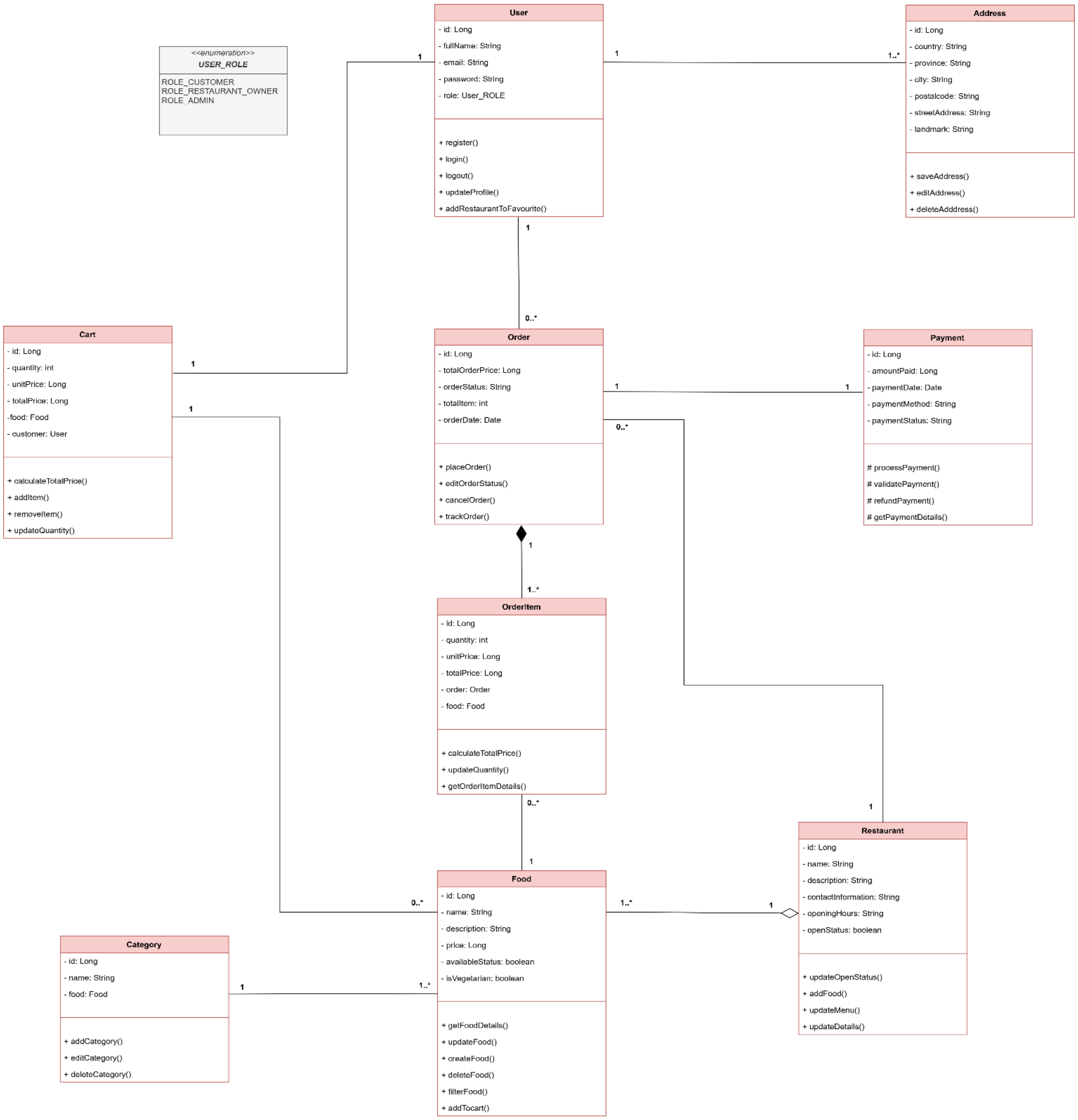
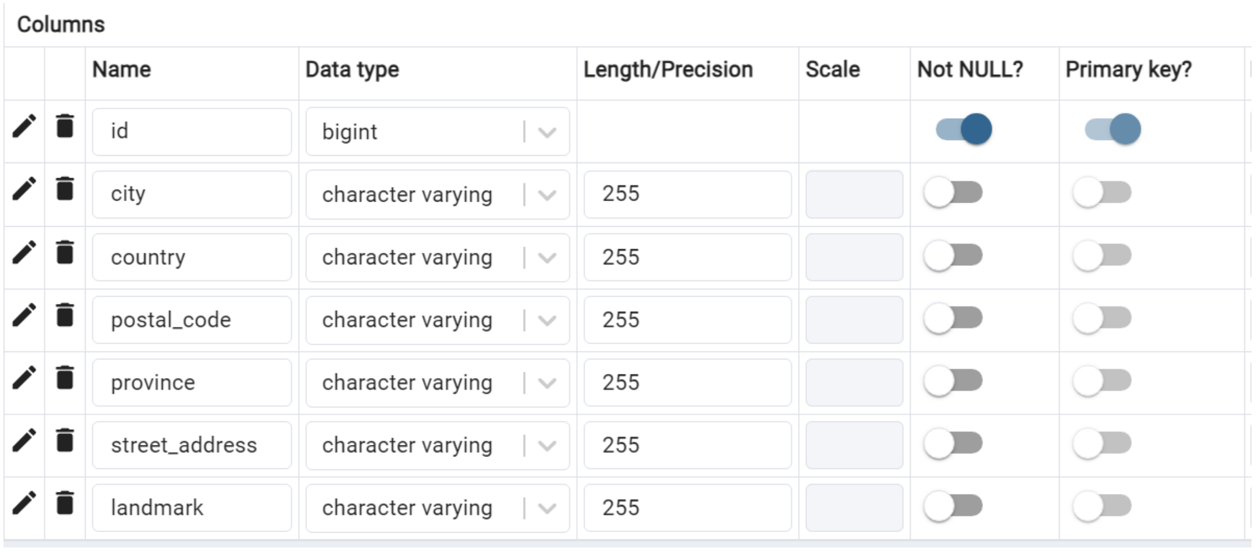


Figure 26: Class Diagram

4.7 Data Dictionary

A Data Dictionary is a detailed list of all the data elements used in a system, explaining their meaning, relationships, format, and usage. It serves as a reference guide for developers, analysts, and stakeholders to understand what each piece of data represents and how it is used within the system (Rouse, 2020). The main objective of the data dictionary is to figure out the entire schema of the **Malt** database. The contents of the data dictionary are shown below:

- **Data Dictionary for Address Table**

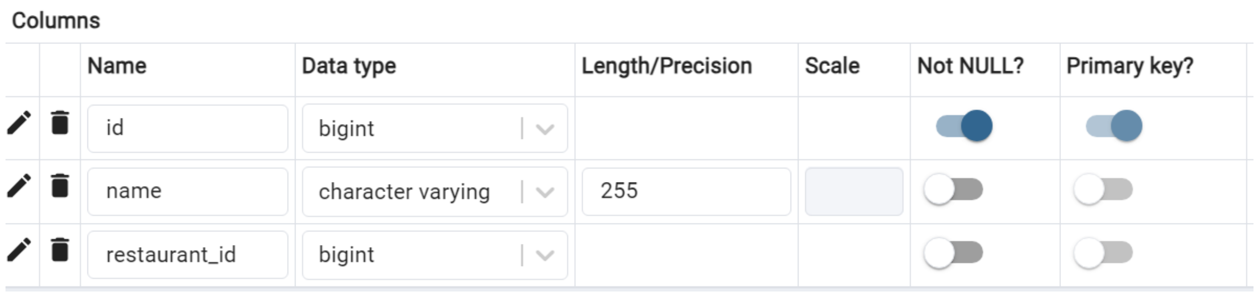


Columns

		Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
✎	🗑	id	bigint			✓	✓
✎	🗑	city	character varying	255			
✎	🗑	country	character varying	255			
✎	🗑	postal_code	character varying	255			
✎	🗑	province	character varying	255			
✎	🗑	street_address	character varying	255			
✎	🗑	landmark	character varying	255			

Figure 27: Data Dictionary for Address Table

- **Data Dictionary for Category Table**

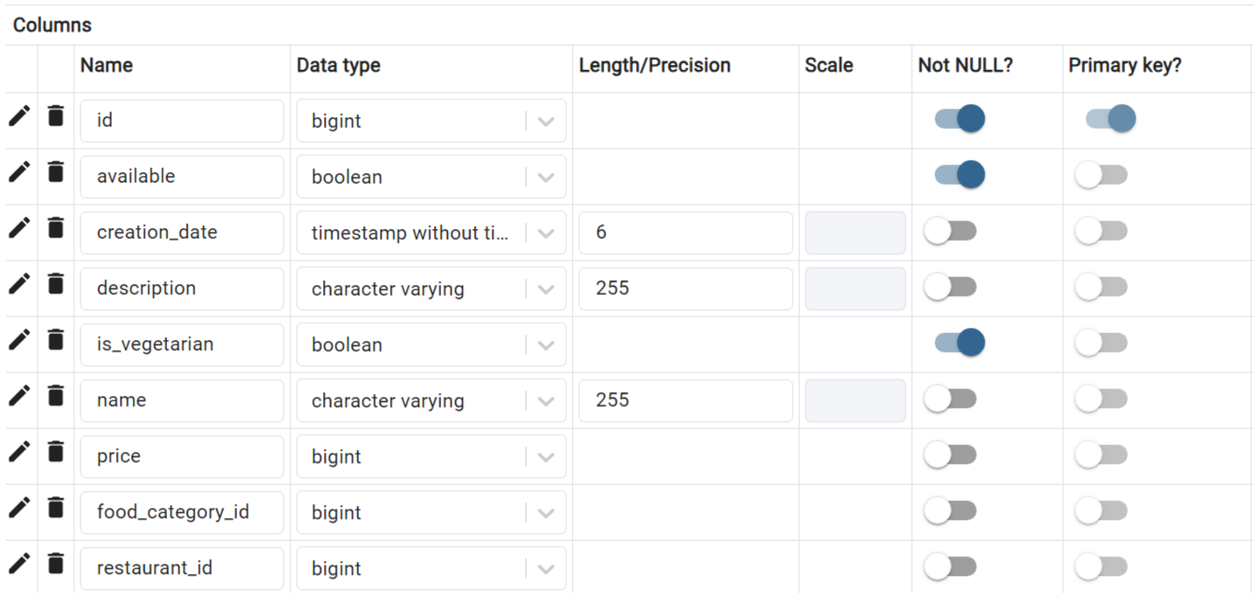


Columns

		Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
✎	🗑	id	bigint			✓	✓
✎	🗑	name	character varying	255			
✎	🗑	restaurant_id	bigint				

Figure 28: Data Dictionary for Category Table

- **Data Dictionary for Food Table**

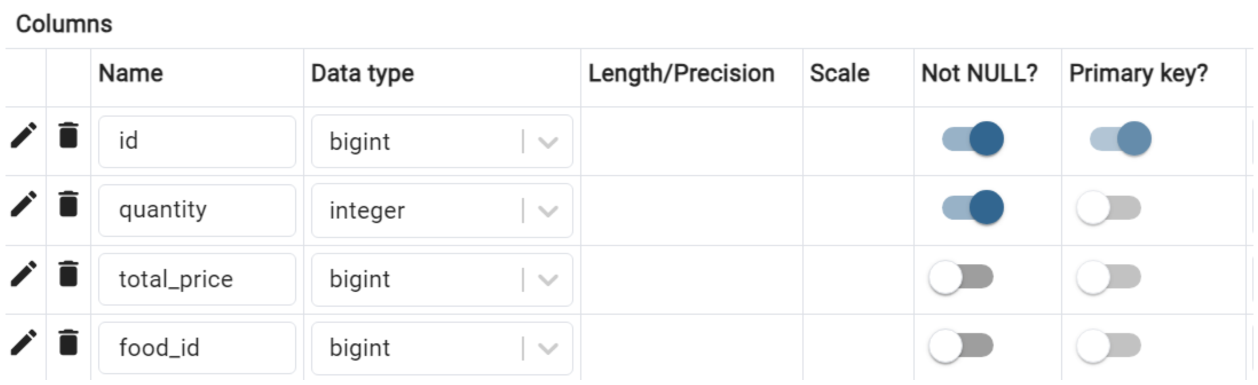


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
available	boolean			✓	
creation_date	timestamp without ti...	6			
description	character varying	255			
is_vegetarian	boolean			✓	
name	character varying	255			
price	bigint				
food_category_id	bigint				
restaurant_id	bigint				

Figure 29: Data Dictionary for Food Table

- **Data Dictionary for Order Item Table**

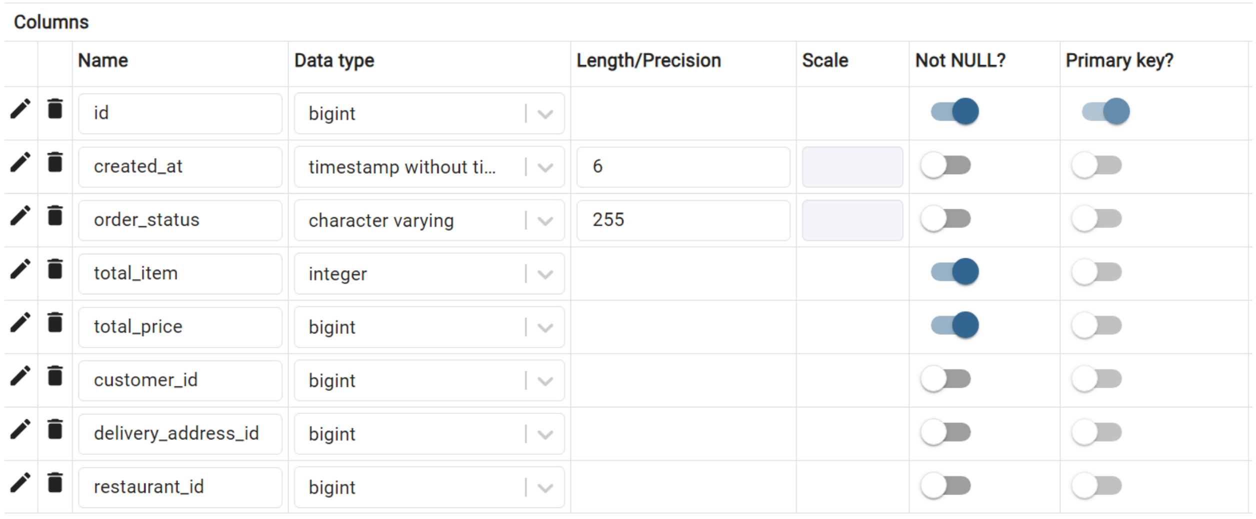


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
quantity	integer			✓	
total_price	bigint				
food_id	bigint				

Figure 30: Data Dictionary for Order Item Table

➢ Data Dictionary for Orders Table

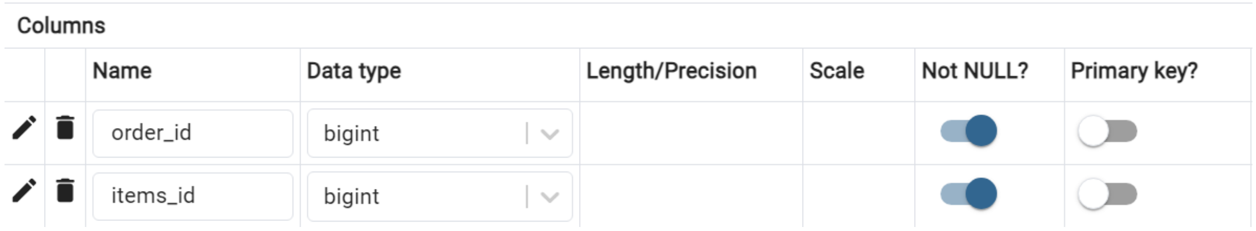


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
created_at	timestamp without ti...	6			
order_status	character varying	255			
total_item	integer			✓	
total_price	bigint			✓	
customer_id	bigint				
delivery_address_id	bigint				
restaurant_id	bigint				

Figure 31: Data Dictionary for Orders Table

➢ Data Dictionary for Orders Items Table

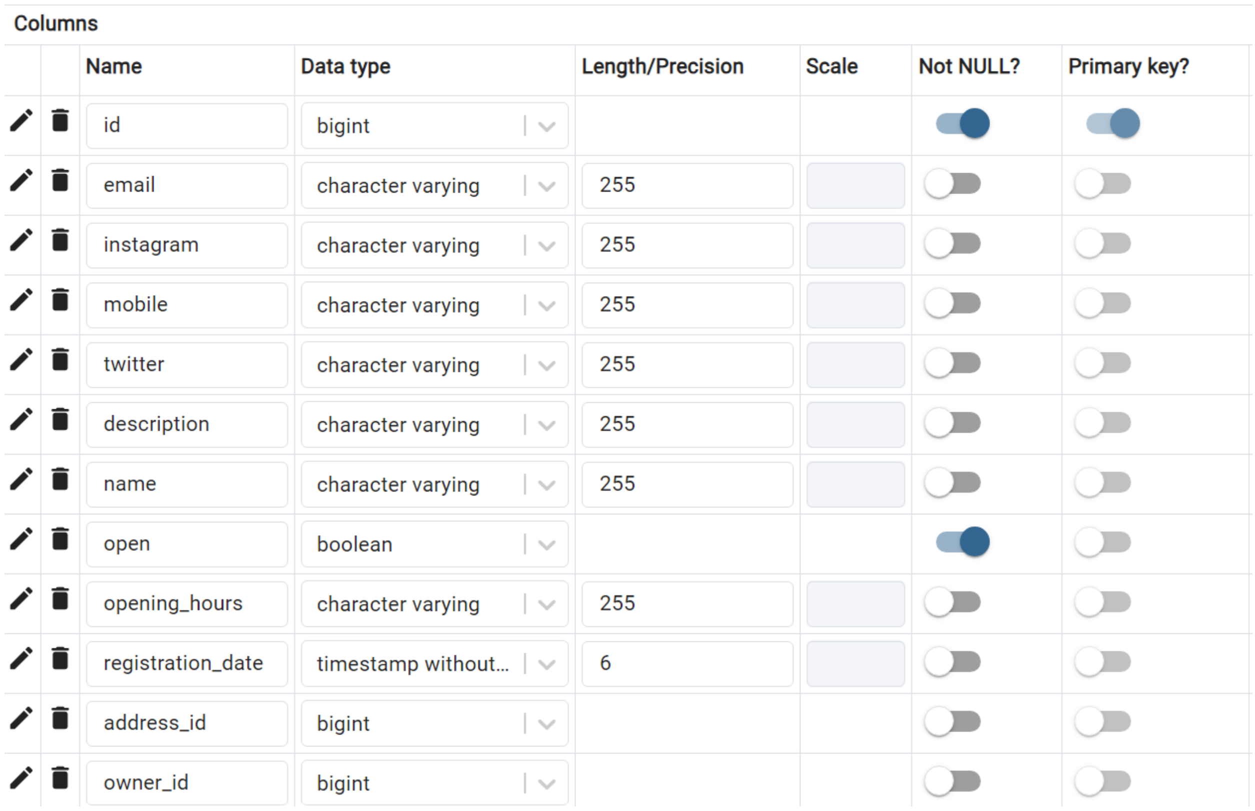


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
order_id	bigint			✓	
items_id	bigint			✓	

Figure 32: Data Dictionary for Orders Items Table

➢ Data Dictionary for Restaurant Table

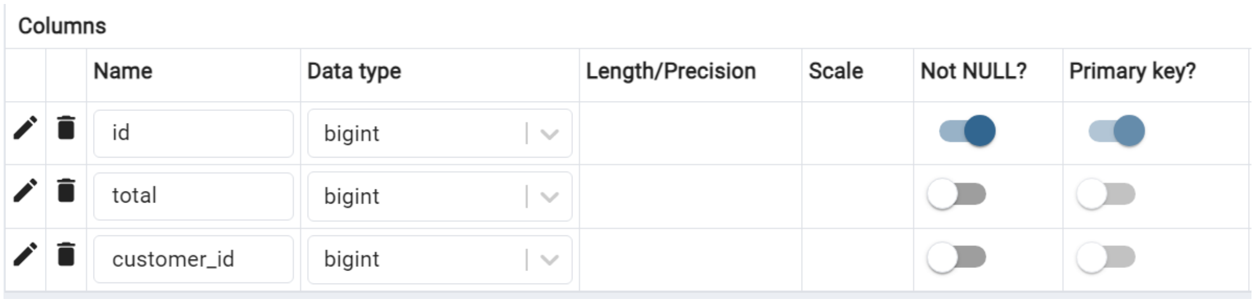


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
email	character varying	255		✗	✗
instagram	character varying	255		✗	✗
mobile	character varying	255		✗	✗
twitter	character varying	255		✗	✗
description	character varying	255		✗	✗
name	character varying	255		✗	✗
open	boolean			✓	✗
opening_hours	character varying	255		✗	✗
registration_date	timestamp without...	6		✗	✗
address_id	bigint			✗	✗
owner_id	bigint			✗	✗

Figure 33: Data Dictionary for Restaurant Table

➢ Data Dictionary for Cart Table

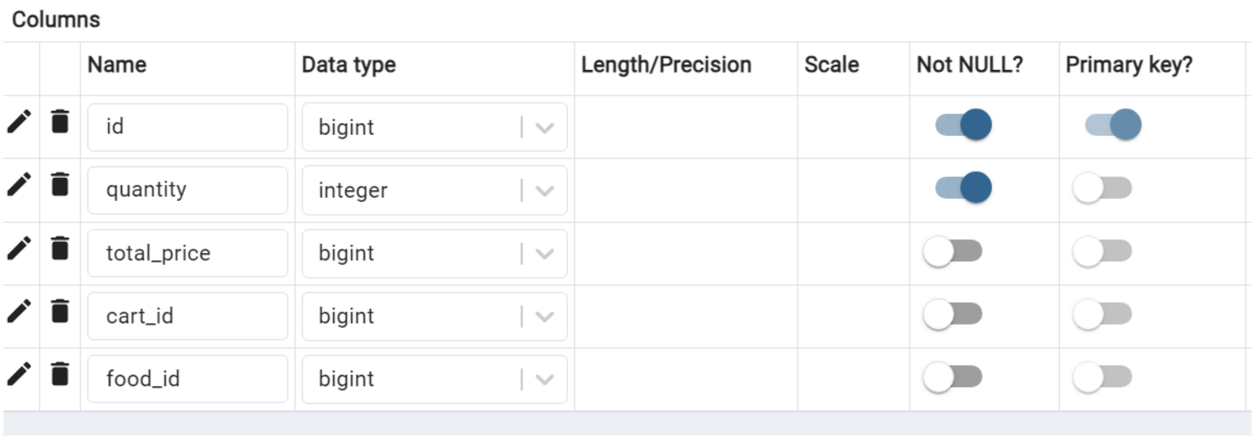


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
total	bigint			✗	✗
customer_id	bigint			✗	✗

Figure 34: Data Dictionary for Cart Table

➢ Data Dictionary for Cart Item Table

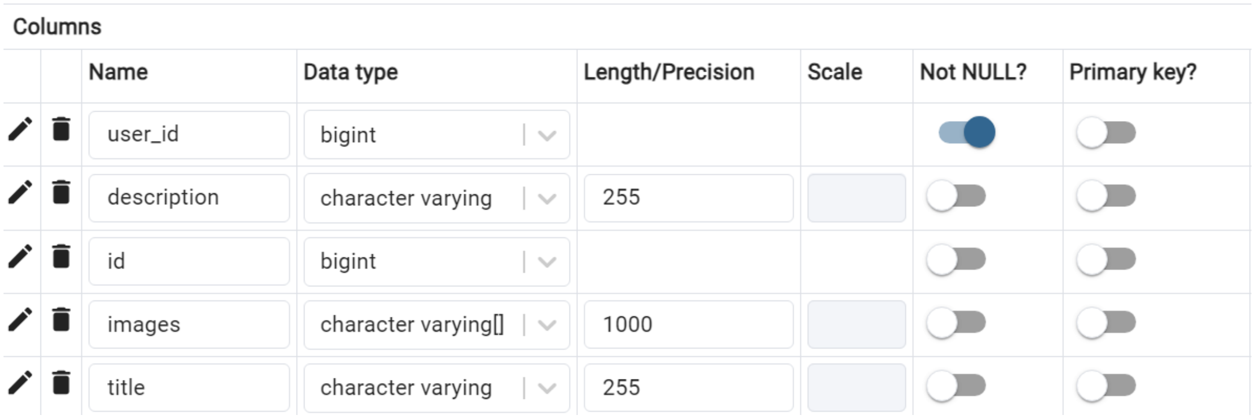


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			✓	✓
quantity	integer			✓	
total_price	bigint				
cart_id	bigint				
food_id	bigint				

Figure 35: Data Dictionary for Cart Item Table

➢ Data Dictionary for User Favorites Table

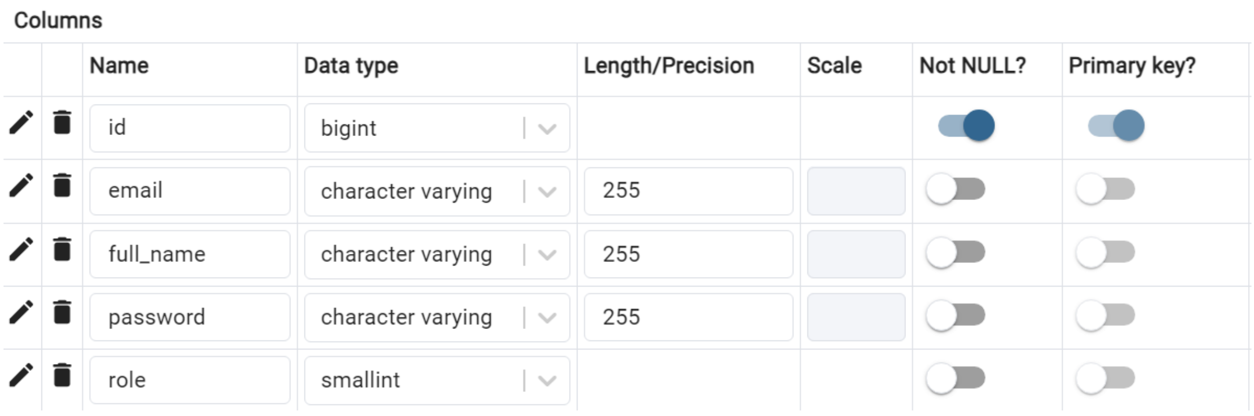


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
user_id	bigint			✓	
description	character varying	255			
id	bigint				
images	character varying[]	1000			
title	character varying	255			

Figure 36: Data Dictionary for User Favorites Table

➢ **Data Dictionary for Users Table**

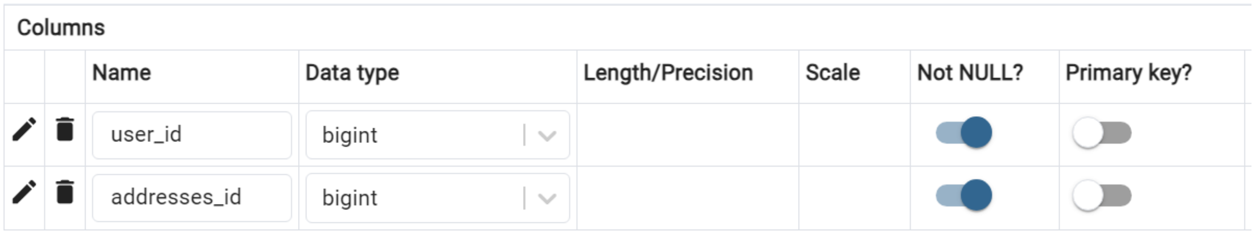


Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			●	●
email	character varying	255		○	○
full_name	character varying	255		○	○
password	character varying	255		○	○
role	smallint			○	○

Figure 37: Data Dictionary for Users Table

➢ **Data Dictionary for Users Addresses Table**



Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
user_id	bigint			●	○
addresses_id	bigint			●	○

Figure 38: Data Dictionary for Users Addresses Table

4.8 Prototype Design

Prototype Design refers to creating an early model or mockup of a user interface (UI) to visualize and test the design before full development. This prototype demonstrates how the frontend will look and function, allowing designers, developers, and stakeholders to review and provide feedback. It helps to identify potential issues and improve the user experience. Below, I have provided login and registration page prototype design.

➢ Registration Page Prototype

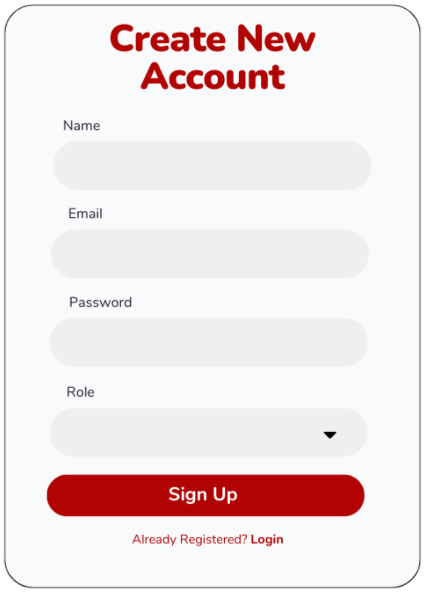


Figure 39: Registration Page Prototype

➢ Login Page Prototype

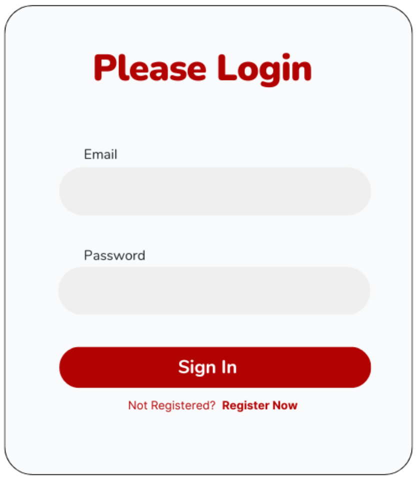


Figure 40: Login Page Prototype

4.9 Development

4.9.1 Software Architecture Pattern

In my project, I have employed the Model-View-Controller (MVC) architecture pattern. This approach organizes the application into three interconnected components:

1. **Model:** This layer is responsible for representing the data and encapsulating the business logic of the application. It handles data storage, retrieval, and manipulation, interacting with the database through the repository.
2. **Service:** The service layer acts as an intermediary between the controller and the repository. It contains the core business logic and manages operations that involve multiple repositories or external services, ensuring a clear separation of concerns.
3. **Repository:** This component provides an abstraction over the data access layer, handling database interactions. By using a Hibernate ORM (Object-Relational Mapping) tool, the repository simplifies data persistence and retrieval tasks.
4. **Controlle**r: The controller layer manages incoming HTTP requests, processes user inputs, and generates the appropriate responses. It communicates with the service layer to execute operations and retrieve data, facilitating user interaction with the system.

Below, I have provided the folder structure of my project backend following MVC architecture pattern.

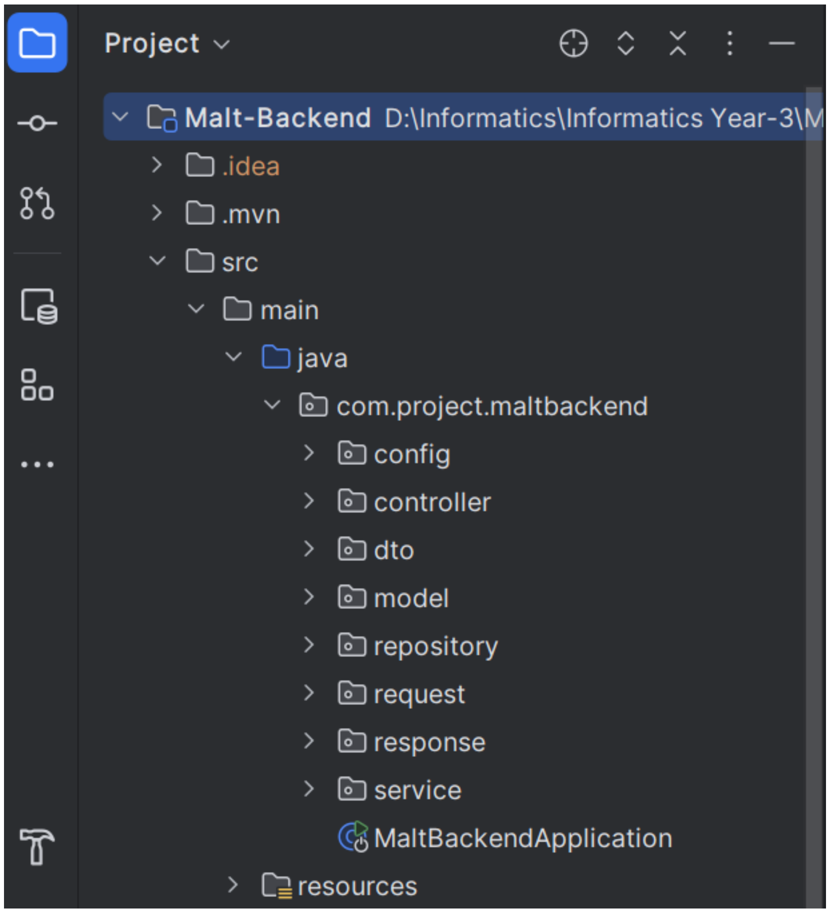
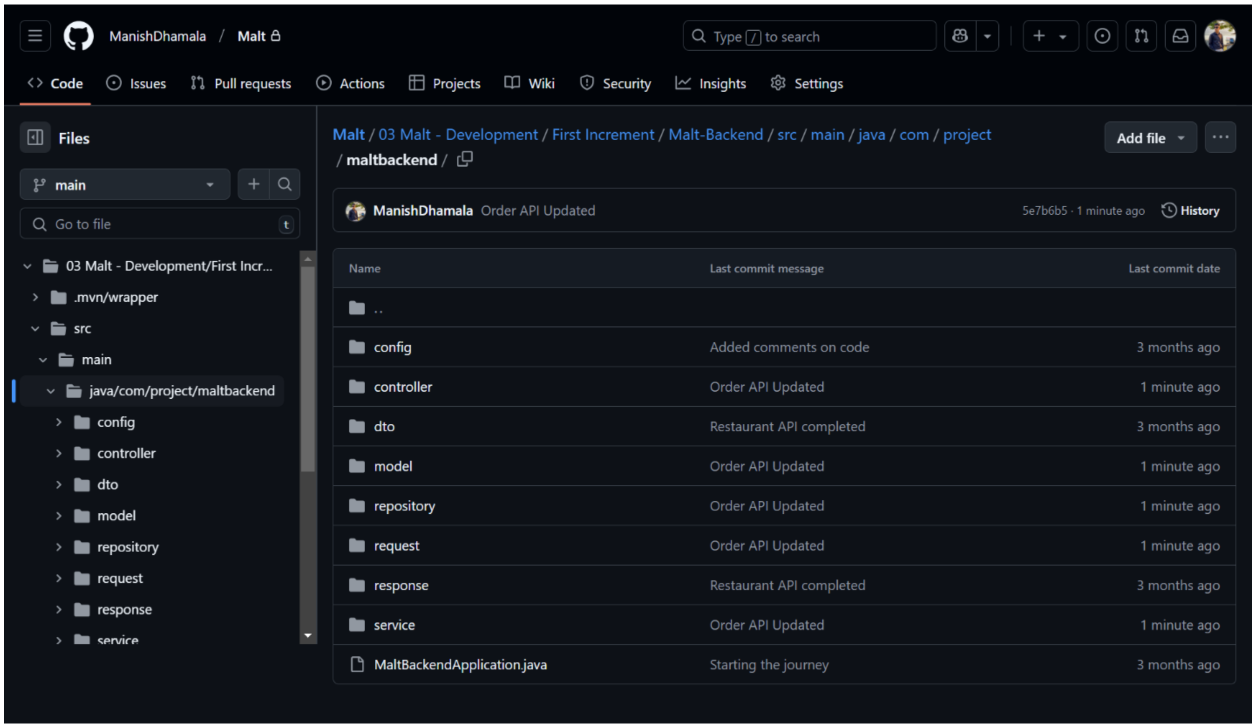

Project
```
Malt-Backend  D:\Informatics\Informatics Year-3\M
  .idea
  .mvn
  src
    main
      java
        com.project.maltbackend
          config
          controller
          dto
          model
          repository
          request
          response
          service
          MaltBackendApplication
      resources
```

Figure 41: Folder Structure of Malt project

By adopting the MVC pattern, I have ensured a clean separation of concerns within my project. This has made the codebase more modular, maintainable, and scalable, thereby enhancing the overall development process and future extensibility.

4.9.2 Version Control

In my project, I use Git for version control to manage and track code changes efficiently and GitHub serves as a backup for my project ensuring that the code is securely stored and accessible from any location. By using Git and GitHub, I have maintained a well-organized and structured development workflow, which has been crucial for the development phase of my project. Below, I have provided a screenshot of my GitHub Repository.



Malt / 03 Malt - Development / First Increment / Malt-Backend / src / main / java / com / project / maltbackend /

ManishDhamala Order API Updated 5e7b6b5 · 1 minute ago

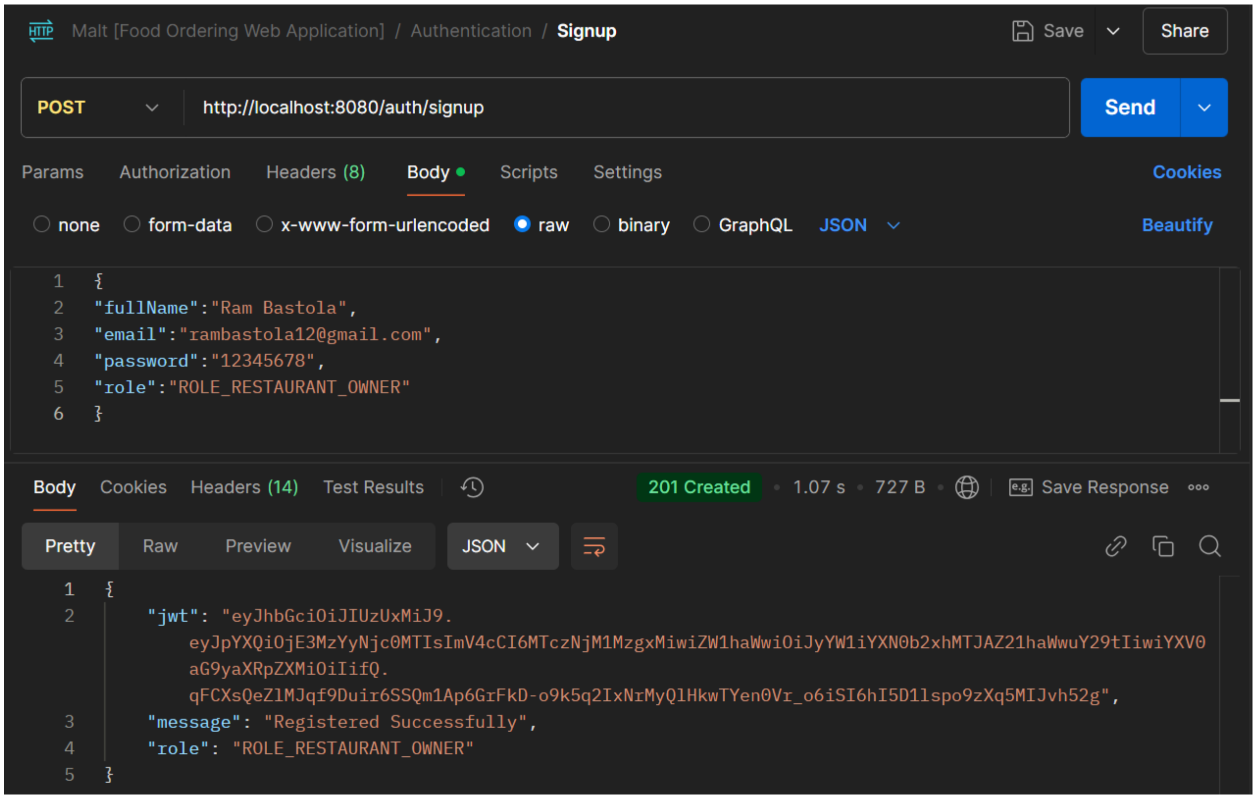
Name	Last commit message	Last commit date
..		
config	Added comments on code	3 months ago
controller	Order API Updated	1 minute ago
dto	Restaurant API completed	3 months ago
model	Order API Updated	1 minute ago
repository	Order API Updated	1 minute ago
request	Order API Updated	1 minute ago
response	Restaurant API completed	3 months ago
service	Order API Updated	1 minute ago
MaltBackendApplication.java	Starting the journey	3 months ago

Figure 42: GitHub Repository of Malt Project

4.9.3 Login and Register API Development

I have completed the development of the Login and Register API for my project, implementing **JSON Web Token (JWT)** for authentication and security. JWT is used to securely transmit user authentication data between the client and the server. Upon successful login, the server generates a JWT, which is then sent to the client and used for subsequent requests to verify the user's identity. This ensures that sensitive data is protected and that only authenticated users can access certain features of the application. The integration of JWT has improved both the security and efficiency of the authentication process. Below, I have provided the evidence of Login and Register API development.

- **Register API**



POST http://localhost:8080/auth/signup

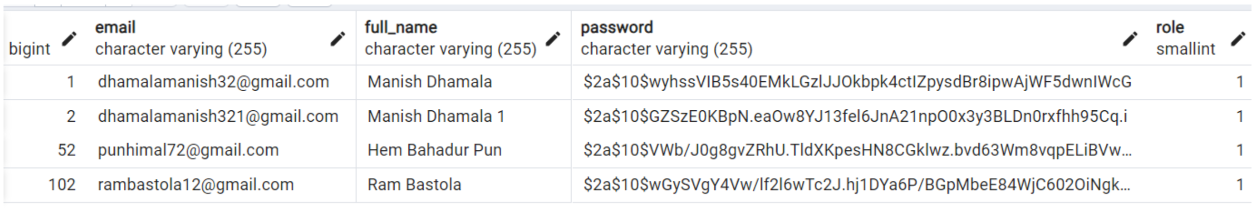
```
{
"fullName":"Ram Bastola",
"email":"rambastola12@gmail.com",
"password":"12345678",
"role":"ROLE_RESTAURANT_OWNER"
}
```

201 Created

```
{
    "jwt": "eyJhbGciOiJIUzUxMiJ9.
        eyJpYXQiOjE3MzYyNjc0MTIsImV4cCI6MTczNjM1MzgxMiwiZW1haWwiOiJyYW1iYXN0b2xhMTJAZ21haWwuY29tIiwiYXV0
        aG9yaXRpZXMiOiIifQ.
        qFCXsQeZlMJqf9Duir6SSQm1Ap6GrFkD-o9k5q2IxNrMyQlHkwTYen0Vr_o6iSI6hI5D1lspo9zXq5MIJvh52g",
    "message": "Registered Successfully",
    "role": "ROLE_RESTAURANT_OWNER"
}
```

Figure 43: Register API Test

- **Registered User Information Save in Database**



bigint	email character varying (255)	full_name character varying (255)	password character varying (255)	role smallint
1	dhamalamanish32@gmail.com	Manish Dhamala	$2a$10$wyhssVIB5s40EMkLGzlJJOkbpk4ctIZpysdBr8ipwAjWF5dwnIWcG	1
2	dhamalamanish321@gmail.com	Manish Dhamala 1	$2a$10$GZSzE0KBpN.eaOw8YJ13fel6JnA21npO0x3y3BLDn0rxfhh95Cq.i	1
52	punhimal72@gmail.com	Hem Bahadur Pun	$2a$10$VWb/J0g8gvZRhU.TldXKpesHN8CGklwz.bvd63Wm8vqpELiBVw...	1
102	rambastola12@gmail.com	Ram Bastola	$2a$10$wGySVgY4Vw/If2l6wTc2J.hj1DYa6P/BGpMbeE84WjC602OiNgk...	1

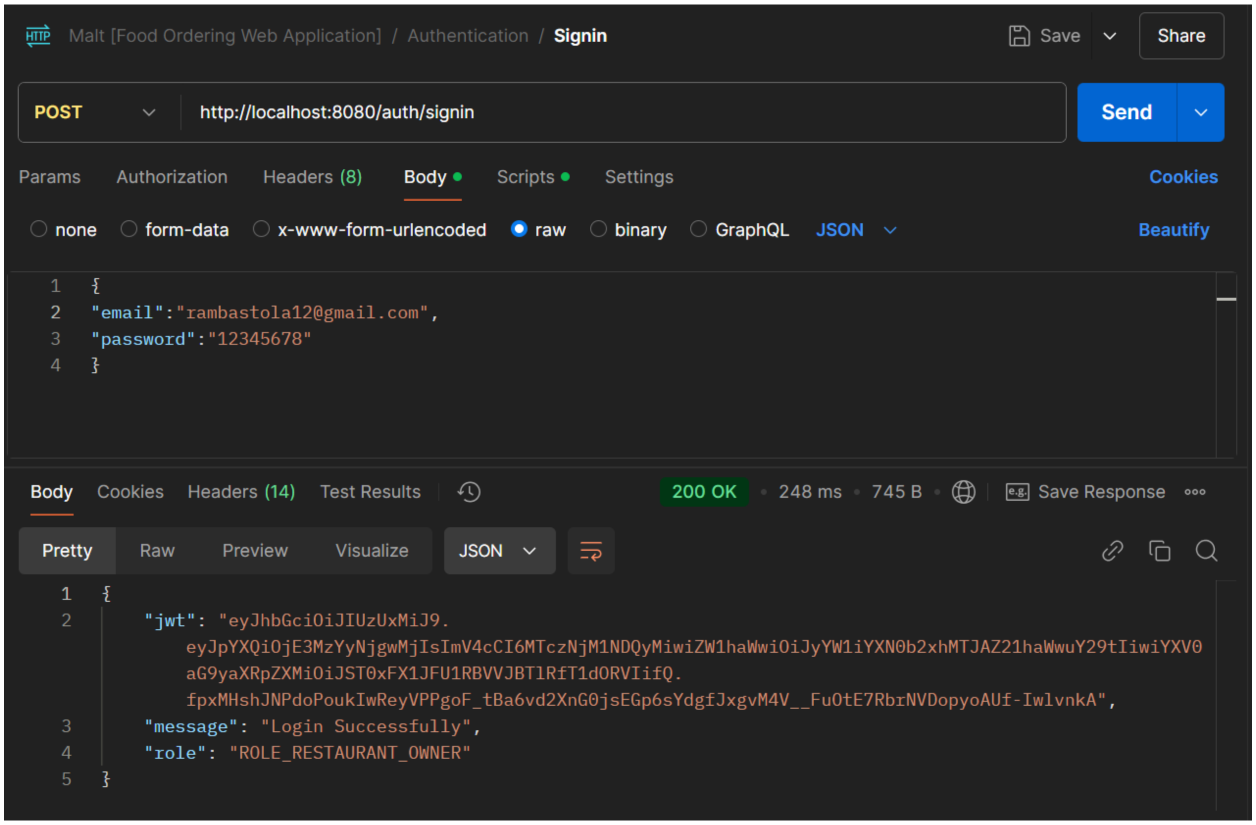
Figure 44: Registered User Information in Database

- **Test case for Register API**

Table 6: Test case for Register API

Objective	To register a new user and store their information in the database securely.
Action	Provide user details such as full name, email, password, and role in the registration request.
Expected Output	A JWT token must return along with the success message, “Registered Successfully” and the user information should be store in the database securely, with the password saved in an encrypted form.
Actual Output	A JWT token is returned along with the success message, “Registered Successfully” and the user information is saved in the database, with the password encrypted.
Conclusion	Test Success

➢ Login API



POST http://localhost:8080/auth/signin

```
{
"email":"rambastola12@gmail.com",
"password":"12345678"
}
```

200 OK • 248 ms • 745 B

```
{
    "jwt": "eyJhbGciOiJIUzUxMiJ9.
        eyJpYXQiOjE3MzYyNjgwMjIsImV4cCI6MTczNjM1NDQyMiwiZW1haWwiOiJyYW1iYXN0b2xhMTJAZ21haWwuY29tIiwiYXV0
        aG9yaXRpZXMiOiJST0xFX1JFU1RBVVJBTlRfT1dORVIifQ.
        fpxMHshJNPdoPoukIwReyVPPgoF_tBa6vd2XnG0jsEGp6sYdgfJxgvM4V__FuOtE7RbrNVDopyoAUf-IwlvnkA",
    "message": "Login Successfully",
    "role": "ROLE_RESTAURANT_OWNER"
}
```

Figure 45: Login API Test

➢ Test case for Login API

Table 7: Test case for Login API

Objective	To authenticate and log in a registered user.
Action	Provide valid login credentials, including email and password in the login request.
Expected Output	A JWT token must return along with a success message, "Login Successfully."
Actual Output	A JWT token is returned along with a success message, “Login Successfully”.
Conclusion	Test Success

5. Further Work

The Malt project is being developed using the incremental model, and as the first increment nears completion, basic progress has been made. However, significant work remains in the second and third increments to ensure the successful delivery of a comprehensive and user-friendly food ordering web application. This section outlines the planned activities and tasks for the remaining development phases, elaborating on the objectives, deliverables, and importance of each stage.

1. Second Increment: Core Functionalities Development

The second increment focuses on implementing the foundational features of the application that facilitate its primary purpose enabling users to order food smoothly from restaurants. The planned work includes:

- **Restaurant Listings and Menu Management**
    - Develop and integrate backend APIs for managing restaurant data and their respective menus.
    - Implement a user interface that displays restaurant details and menu items with filtering and sorting options.
    - Ensure real-time synchronization of menu updates with the database.
- **Order Placement System**
    - Develop the “Add to Cart” functionality, allowing users to select multiple food items for a single order.
    - Create an order summary page where users can review their selections, view itemized costs, and confirm their orders.
- **Payment Gateway Integration**
    - Integrate secure payment gateways such as e-Sewa and enable a "Cash on Delivery" option.

    - Implement backend APIs to handle payment processing, error handling, and transaction status updates.

- **Order Confirmation and Notifications**
    - Develop an instant notification system to confirm orders, both in-app and through email.
    - Ensure accurate and detailed order confirmation messages with necessary order details.
- **User Profile Management**
    - Build a user profile section allowing users to manage personal information, saved addresses, and preferred payment methods.

These tasks will lay the groundwork for a fully functional ordering system, addressing the core requirements outlined in the Software Requirement Specification (SRS) section.

2. Third Increment: Advanced Features and Optimization

The third increment will focus on enhancing user experience, optimizing performance, and implementing additional features that differentiate Malt from existing food ordering platforms. The tasks for this stage are as follows:

- **Order Tracking System**
    - Develop a real-time order tracking feature that updates users on their order status, such as preparation, ready for delivery, and delivered.
- **Restaurant Dashboard**
    - Build a dedicated dashboard for restaurant owners to manage their menus, view order details, and analyze customer interactions.
    - Include functionality for restaurants to promote special events or offers through the dashboard.

- **Saving Favorites Restaurant**
    - Implement a “Favorites” feature, enabling users to save their preferred restaurants for quick access.
- **Responsive Design Enhancements**
    - Ensure the application is fully responsive and provides a smooth user experience across various devices and screen sizes.
    - Perform cross-platform testing to identify and fix compatibility issues on Android, iOS, and desktop browsers.
- **System Optimization and Security**
    - Optimize the database for faster queries and efficient data handling.
    - Conduct thorough testing and code reviews to ensure secure user authentication, data encryption, and protection against vulnerabilities like SQL injection or XSS attacks.
- **Error Handling and User Support**
    - Enhance error-handling mechanisms to provide users with clear messages during issues such as failed transactions or connectivity problems.
    - Develop a basic user support system, such as FAQs or a chatbot, to assist users with common queries.
- **Enhance Performance**
    - Optimize the application to meet non-functional requirements, ensuring fast response times, even under high traffic or slower internet conditions.
    - Conduct load testing and stress testing to evaluate system behavior under various conditions.

3. Testing, Feedback, and Iterations

As the second and third increments progress, continuous testing and iterative improvements will play a critical role in ensuring a robust and user-friendly application. Planned activities include:

- **Integration Testing**
  - Test the interaction between various components, such as the user interface, APIs, and database, to ensure smooth data flow.
- **User Acceptance Testing (UAT)**
  - Gather feedback from potential users, including customers and restaurant owners, to identify areas of improvement and prioritize enhancements.
- **Performance and Security Testing**
  - Conduct rigorous testing to validate performance metrics and security protocols, addressing any issues before the final release.
- **Documentation**
  - Maintain up-to-date technical and user documentation to facilitate future development and ease of use for end-users.

The further work to be done from this point of time along with the estimated completion date is presented in the following table:

Table 8: Further Work

S. N	**Task to be Done**	**Estimated Task Completion Date**
1	Release First Version	10 Jan, 2025
2	Design prototype for Home page, Cart page, User Profile Page and Restaurant Detail Page	20 Jan, 2025
3	Develop backend APIs for restaurant listings and menu management	30 Jan, 2025
4	Implement "Add to Cart" functionality	10 Feb, 2025
5	Develop the order summary and confirmation page	15 Feb, 2025
6	Integrate payment gateways (e-Sewa, Cash on Delivery)	25 Feb, 2025
7	Implement order confirmation and notification system	5 Mar, 2025
8	Build user profile management section	10 Mar, 2025
9	Complete testing and second increment features	18 Mar, 2025
10	Release Second Version	20 Mar, 2025
11	Build restaurant dashboard for menu and order management	5 Apr, 2025
12	Develop Admin Panel	12 Apr, 2025
13	Implement "Favorites" feature	15 Apr, 2025
14	Optimize responsive design for all devices	20 Apr, 2025
15	Conduct performance and security testing	25 Apr, 2025
16	Finalize documentation and release final version	28 Apr, 2025

Conclusion

The remaining work for the Malt project is significant but well-defined, with a clear roadmap for delivering an efficient food ordering web application. By sticking to the incremental model, the project ensures that each phase builds upon the previous one, resulting in continuous development and scalable solution. Completing the outlined tasks in the second and third increments will not only fulfill the functional and non-functional requirements but also delivers an efficient food ordering application for the online food ordering industry.

References

S M Noersidik & L Warlina , 2018. *Designing Web-based Food Ordering Information,* s.l.: Purpose LED Publishing.

Abraham, A., 2021. A Study on the Effectiveness of Online Food Applications on Registered Restaurants. *International Journal of Creative Research Thoughts (IJCRT),* 9(1), p. 4694–4700.

Adobe Communications Team, 2022. *business.adobe.com.* [Online]
Available at: https://business.adobe.com/blog/basics/what-is-a-gantt-chart
[Accessed 5 Januray 2025].

Bhaskar , S., 2024. *NimbleWork.* [Online]
Available at: https://www.nimblework.com/kanban/what-is-kanban/
[Accessed 4 January 2025].

Brown, F., 2024. *GURU99.* [Online]
Available at: https://www.guru99.com/er-diagram-tutorial-dbms.html
[Accessed 5 January 2025].

Carter, M., 2024. *boardmix.* [Online]
Available at: https://boardmix.com/tips/collaboration-diagram/
[Accessed 5 January 2025].

Creately, 2022. *creately.com.* [Online]
Available at: https://creately.com/guides/use-case-diagram-tutorial/
[Accessed 5 January 2025].

Creately, 2024. *creately.com.* [Online]
Available at: https://creately.com/blog/software-teams/class-diagram-tutorial/
[Accessed 5 January 2025].

Dhakulkar, A., Girde , S. & Taywade, P., 2018. *Online Food Ordering System,* Nagpur: International Journal of Emerging Technologies (IJET).

Gallagher, A., Dunleavy, J. & Reeves, P., 2019. *IBM.* [Online]
Available at: https://developer.ibm.com/articles/waterfall-model-advantages-disadvantages/
[Accessed 4 January 2025].

Gautam, . N., 2021. *techlekh.com.* [Online]
Available at: https://techlekh.com/food-tech-companies-nepal-growth/
[Accessed 2 January 2024].

Lindemulder, G., 2024. *IBM.* [Online]
Available at: https://www.ibm.com/think/topics/data-flow-diagram
[Accessed 5 January 2025].

Medium, 2018. *medium.com.* [Online]
Available at: https://medium.com/thousand-words-by-creately/the-ultimate-guide-to-sequence-diagrams-a78e0e516886
[Accessed 5 January 2025].

Nikitin, V., 2024. *itransition.* [Online]
Available at: https://www.itransition.com/software-development/methodologies
[Accessed 3 January 2025].

Organ, C., 2024. *www.forbes.com.* [Online]
Available at: https://www.forbes.com/advisor/business/what-is-work-breakdown-structure/#:~:text=A%20work%20breakdown%20structure%20(WBS,deliverables%20int

o%20a%20single%20tool.
[Accessed 4 January 2025].

R. Ramesh, et al., 2021. *An empirical study of online food delivery services from applications perspective,* Amsterdam: Materials Today Proceedings.

Rana, K., 2021. *Art of Testing.* [Online]
Available at: https://artoftesting.com/prototype-model
[Accessed 4 January 2025].

Restolabs, 2024. *www.restolabs.com.* [Online]
Available at: https://www.restolabs.com/blog/advantages-food-ordering-system-restaurants
[Accessed 2 January 2025].

Rouse, M., 2020. *Techopedia.* [Online]
Available at: https://www.techopedia.com/definition/27752/data-dictionary
[Accessed 6 January 2025].

Sachan, D., 2024. *scaler.com.* [Online]
Available at: https://www.scaler.com/topics/incremental-model-in-software-engineering/
[Accessed 4 January 2025].

shopdev, 2023. *shopdev.co.* [Online]
Available at: https://shopdev.co/blog/software-development-milestones
[Accessed 4 January 2025].