PUSL3119 Computing Individual Project

Project Interim Report

Food Desire

An Online Food Ordering System

Supervisor: Mr. Pramudya Thilakarathne

Name: Rathnayake M Rathnayake

Plymouth Index Number: 10747887

Degree Program: BSc (Hons) Computer Science

Table of Contents

TABLE OF FIGURE

Chapter One
INTRODUCTION

1.1 INTRODUCTION

Online shopping has become a convenient and popular way of purchasing goods in the modern world. It offers many benefits such as speed, security, variety and comfort. One of the main advantages of online shopping is that it saves time for busy consumers who do not want to waste time going to physical stores. Online shopping allows them to order anything they need from their doorsteps and receive it quickly and safely.

Online food ordering is the latest trend of service that can be offered by fast-food restaurants. It allows customers to order food online and have it delivered to their location. It also enables electronic payment systems that make transactions convenient and secure. Online food ordering can benefit both customers and restaurants by saving time, reducing costs and increasing customer satisfaction.

These systems allow fast-food restaurants to focus more on food preparation and faster delivery, while reaching a wider customer base through the internet. Online food ordering systems are a valuable business opportunity for the food industry, as they leverage the high demand and convenience of internet usage.

1.1 PROBLEM DEFINITION

The food industry is expanding rapidly and constantly innovating to meet the diverse needs and preferences of consumers. One of the emerging trends is the customization of food products, which allows customers to tailor their meals according to their tastes and dietary requirements. However, this option is still limited in scope and availability, especially in online food ordering systems. For example, most pizza restaurants only offer a fixed menu of toppings and crusts, without giving customers the freedom to adjust the quantity or quality of ingredients. This restricts customers from creating their ideal pizza and prevents them from knowing the nutritional value of their order. Therefore, online food ordering systems should enhance their features to enable more flexibility and transparency in food customization.

1.3 PROJECT OBJECTIVES

The online food ordering system is a convenient and efficient way for customers to order their meals from various restaurants. However, some customers may have specific preferences or dietary restrictions that limit their choices of meals. To address this issue, the proposed solution is to add a feature that allows customers to customize their meals by adjusting the amount of ingredients in each meal item. This feature will enhance the customer experience by giving them more control and flexibility over their orders. To implement this feature, we will also need to modify the inventory management system of the restaurant end, so that they can keep track of the available ingredients and update the list of ingredients for each meal item accordingly. This will also improve the restaurant end experience by reducing waste and optimizing inventory.

This solution will update both customer and restaurant end to enable this feature. Customers will be able to select their preferred meal item, before going to the checkout or cart, they will be presented with a page that includes the meal information. The restaurant can provide customers with which ingredients are used in each dish so that customers have full transparency when ordering. Additionally, customers can edit the ingredients as they prefer - for example if a customer would like more chicken pieces in their fried rice then they can increase it from the ingredient list accordingly. This allows for greater control over what goes into your meals while also providing restaurants with an easy way of informing consumers about what is included within each dish on offer.

Moreover, a recommendation system will be implemented to suggest meal items to the users based on their preferences and previous orders. This system will use a content-based filtering machine learning algorithm that will analyze the data collected from each user and match it with the attributes of the meal items. The recommendation system will improve the user experience by providing personalized and relevant suggestions instead of showing the default meals list on the home page. Moreover, the recommendation system will also benefit the restaurant by increasing customer satisfaction and loyalty, as well as boosting sales and revenue.

Chapter 02
SYSTEM ANALYSIS

2.1 FACTS GATHERING TECHNIQUES

2.1.1 RESEARCH

2.1.1.1 Online Food Ordering Systems

Self-ordering systems are systems that allow customers to place orders by themselves without the need for human intervention. These systems have been shown to have various benefits for both customers and businesses. According to (Ratto *et al.*, 2008), self-ordering systems can achieve a state of maximal overall 'success' by optimizing the trade-off between customer satisfaction and operational efficiency. (Gao and Su, 2018) also found that self-ordering techniques can significantly reduce waiting cost and increase demand, as customers prefer faster and more convenient ways of ordering their needs. Self-ordering technology can be implemented through various channels, such as text messages, emails, telephone calls and internet websites or applications.

This system allows customers to place their orders online or through a digital device without the need for human intervention. According to (Thomas and Davis, 1978), this system can replace the traditional ordering technique and save time and resources. Moreover, self-ordering systems can enhance customer satisfaction by offering convenience, speed and delivery options. From a business perspective, self-ordering systems can increase sales volume and revenue by reaching more customers and reducing operational costs. Therefore, self-ordering systems are a beneficial tool for both businesses and customers in today's competitive market.

One of the most modern methods to implement a self-ordering system for a restaurant is online food ordering. This method allows customers to order food from their devices without having to wait for a waiter or a menu. Online food ordering has many benefits for both customers and restaurants. According to (R. *et al.*, 2017), online food ordering systems are efficient and easy to use, as customers can browse through the food items that the restaurant offers and order

the food with few steps. (Daim *et al.*, 2013) also found that customers value the efficiency of the website, speed of response duration on online services, and the quality of the personnel when choosing an online food service. Therefore, online food ordering can enhance customer satisfaction and loyalty, as well as increase sales and revenue for restaurants.

Despite the popularity and convenience of online food ordering services, there have been few significant innovations in this domain. The current system can be improved in various aspects to enhance customer satisfaction and loyalty. According to (Zulkarnain *et al.*, 2015), website quality and service quality are crucial factors that influence the success of online food ordering services. They suggest that online platforms should provide clear and accurate information, easy navigation, fast delivery, and responsive feedback. Moreover, (Goffe *et al.*, 2021) propose that online food ordering services should adopt a more human-centered design approach. They argue that customers should have more autonomy and flexibility in choosing their food options, such as customizing their ingredients, portions, or preferences.

2.1.1.2 Recommendation System

Artificial intelligence (AI) and machine learning (ML) are powerful technologies that can enhance various aspects of e-commerce. E-commerce is a business model that involves buying and selling goods and services online or through other computerized systems. AI and ML can help e-commerce businesses optimize their operations, improve customer experience, increase sales and revenue, and gain competitive advantage. Some examples of AI and ML applications in e-commerce are product recommendation systems, chatbots, fraud detection, inventory management, pricing optimization, and sentiment analysis.

Recommendation systems are a type of artificial intelligence model that are widely used in e-commerce domains. They aim to provide personalized suggestions to users based on their preferences and past interactions with products or services. According to (Ahrens, 2012), recommendation systems can enhance customer satisfaction and loyalty by increasing the time spent on the site, the number of products viewed, and the likelihood of purchase or engagement.

2.2 EXISTING SYSTEM

The online food ordering system is a convenient and efficient way for customers to order food from various restaurants. The system allows customers to browse through a food menu online and select the items they want to order. The system also provides information about the availability, price, and delivery time of the food items. Customers can track their orders online and see when they will receive their food(R. *et al.*, 2017). The system also has a feedback mechanism that enables customers to rate and review the food items and the restaurants. This helps other customers to make informed choices and helps the restaurants to improve their service and quality. The system also has a recommendation feature that suggests food items based on the customer's preferences and previous orders. The system aims to increase the transparency and trustworthiness of the restaurants by providing accurate and reliable information. Customers can pay for their orders online using various payment methods or opt for cash on delivery if they prefer. The system also ensures the security and privacy of each customer by creating separate accounts with unique ID and password(R. *et al.*, 2017).

Online food ordering systems are platforms that allow customers to order food from various restaurants and pay securely online. These systems can include both mobile and web applications that improve the user experience by offering features such as menus, ratings, reviews, delivery tracking and loyalty programs. Online food ordering systems benefit both customers and restaurants by providing convenience, efficiency and transparency.

2.3 DRAWBACKS OF THE EXISTING SYSTEM

One of the drawbacks of current online food ordering systems is that they do not allow customers to customize their food ingredients according to their preferences, dietary needs, or allergies. This can limit customer satisfaction and loyalty, as well as reduce sales opportunities for restaurants that offer such options.

Another drawback is that most online food ordering systems do not have a separate inventory management system for restaurants, which means that restaurants must manually track their stock levels, order supplies, and avoid wastage. This can increase operational costs, errors, and inefficiencies for

restaurants that rely on online orders. A food ordering system that can customize food ingredients and have a separate inventory management system for restaurants would solve these problems by providing more flexibility, convenience, and efficiency for both customers and restaurants.

Chapter 03
REQUIREMENTS SPECIFICATION

3.1 FUNCTIONAL REQUIREMENTS

- Online Food Ordering System
  - Customers can browse through different food categories and items.
  - Customers can customize their orders by changing the amount of ingredients.
  - Customers can see the price of their orders based on their customization.
  - Customers can place their orders online and pay with various methods.
  - Customers can track their orders status and delivery time.
  - Customers can rate and review their orders after delivery.
  - Customers can view their order history and recommendations.

- Inventory Management System
  - Admins can register new employees, suppliers, chefs, and deliverers.
  - Admins can assign roles and permissions to different employees.
  - Suppliers can supply ingredients to the restaurant based on demand.
  - Chefs can create recipes for different food items using ingredients.
  - Chefs can update recipes based on customer feedback or availability of ingredients.
  - Deliverers can take out orders for delivery based on location and priority.
  - Admins can pay for maintenance, salaries, taxes, etc.

- Recommendation System
  - The system collects data from customer orders, ratings, reviews, preferences, etc.
  - The system analyzes data using machine learning algorithms to find patterns and trends.
  - The system suggests food items for customers based on their most consumed food and individual ingredients.

- The system also considers factors such as seasonality, availability, popularity, nutrition, etc.
- The system displays suggestions on the website's home page.

3.2 NON-FUNCTIONAL REQUIREMENTS

- Online Food Ordering System
  - The system has a user-friendly interface with clear buttons, colors, fonts, etc.
  - The system supports English.
  - The system has a high performance.
  - It loads fast.
  - It responds quickly.
  - It handles concurrent requests.
  - The system has a high security.
  - It encrypts sensitive data such as payment information.

- Inventory Management System
  - The system has a reliable database that stores all information related to inventory, employees, suppliers, etc.
  - The system has a backup mechanism that saves data regularly in case of failure or data loss.

- Recommendation System
  - The system has a scalable architecture that allows it to handle increasing numbers of customers, orders, data, etc.
  - The system has an adaptive algorithm that updates itself based on new data or feedback.

3.3 USE CASE DIAGRAM

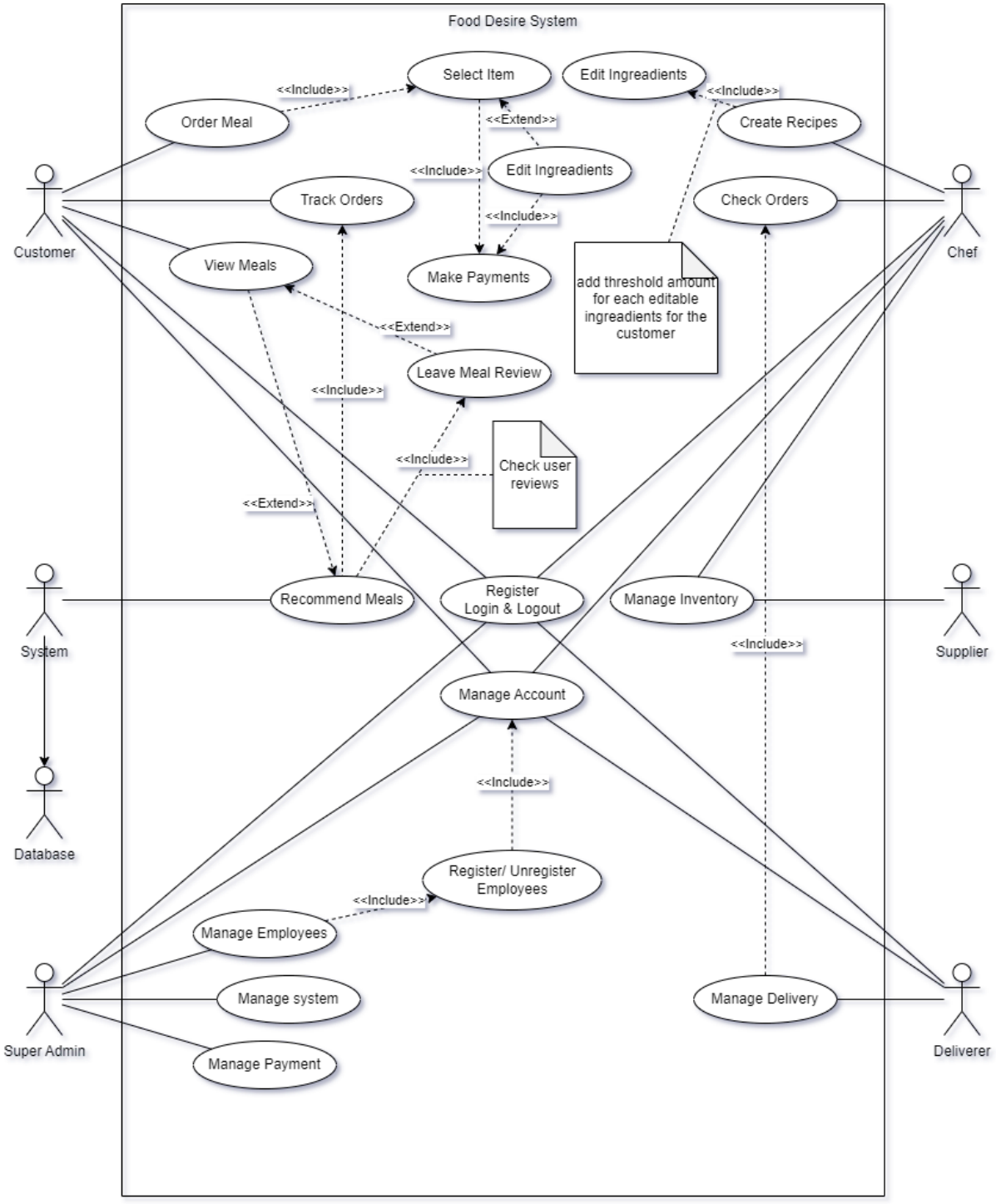


Figure 3. 1 Use case diagram.

3.4 HARDWARE / SOFTWARE REQUIREMENTS

3.4.1 HARDWARE REQUIREMENTS

The online food ordering system is structured like a standard web application, and as such requires no special hardware and only basic software, namely web and database servers, to function properly. The system can run on any device that supports a web browser, such as desktops, laptops, tablets, or smartphones.

The inventory management system is a software application that helps businesses track and manage their inventory levels, orders, sales, and deliveries. The system can also generate reports and insights to optimize inventory performance and reduce costs.

One of the requirements for the inventory management system is the hardware configuration. Depending on the size and complexity of the business, it may need few PCs or a single PC that can be shared with multiple users at the same time with separated display. This option can save space and money by using one PC as a server and connecting multiple monitors, keyboards, and mice to it. Each user can have their own session and access to the inventory management system.

However, this option also has some drawbacks. It may affect the performance and reliability of the system if there are too many users or tasks running on one PC. It may also pose security risks if the PC is not properly protected from unauthorized access or malware. Moreover, it may limit the scalability and flexibility of the system if there is a need to add more users or features in the future.

Therefore, another option is to use a single PC for each user. This option can ensure better performance and security of the system as each user has their own dedicated hardware and software resources. It can also allow more customization and integration of the system with other applications or devices. However, this option also has some challenges. It may require more space and money to purchase and maintain multiple PCs. It may also require more coordination and synchronization among different users to avoid data inconsistency or duplication.

3.4.2 SOFTWARE REQUIREMENTS

To use either option in Chapter 3.4.1, the PC must meet some minimum specifications to run .NET 7.

- Operating system: Windows 10 version 1809 or later.
- Processor: 1.8 GHz or faster processor.
- Memory: 4 GB of RAM; 8 GB of RAM recommended.
- Disk space: Minimum of 800 MB up to 5 GB of available space, depending on features installed; typical installations require 1 GB of free space.
- Network: Internet access (fees may apply).

3.5 NETWORK REQUIREMENTS

The system should have a reliable and secure network connection to communicate with the WinUI 3 desktop application, the Web API, and the Blazor web UI. The network bandwidth should be sufficient to handle concurrent requests and data transfers without delays or errors.

The system should use HTTP protocol for encryption and authentication of data between the client and server. The system should also implement SSL certificates for verifying the identity of the server and preventing man-in-the-middle attacks.

The system should use a cloud service provider such as Azure Cloud to host the Web API and the Blazor web UI. The cloud service provider should offer scalability, availability, backup, and disaster recovery features for the system. The cloud service provider should also comply with relevant data protection laws and regulations.

Chapter 04
FEASIBILITY STUDY

4.1 OPERATIONAL FEASIBILITY

To ensure that the system meets the needs of stakeholders, surveys and interviews can be conducted to gather information on their requirements, preferences, and expectations. These can be conducted with potential customers, restaurant owners, chefs, deliverers, and suppliers. This information can be used to design a user-friendly and efficient system that meets their needs.

Additionally, the benefits and costs of the system for each stakeholder group can be evaluated. This can include estimating the time and money the system can save customers by providing them with online ordering and fast delivery, as well as estimating the revenue and profit the system can generate for restaurant owners by reducing food waste and optimizing inventory management.

The risks and challenges of implementing the system in the real world should also be assessed. This can include identifying legal, ethical, and social issues that may arise from collecting and processing personal data, as well as potential threats to the system's security and reliability.

4.2 TECHNICAL FEASIBILITY

Reviewing the technical resources available for developing the system using .NET is necessary. This includes the necessary tools, technology, materials, labor, and logistics to create a WinUI 3 desktop application for inventory management system (IMS), a Web API to consume in a blazorUi (web app), and an Azure cloud deployment.

Additionally, the technical capabilities of the system should be evaluated in terms of functionality, performance, scalability, and compatibility. This can include testing how well the system can handle different types of inputs, outputs, processes, and data formats, as well as testing how quickly it can respond to user requests and how many concurrent users it can support.

Exploring technical innovations that can be incorporated into the system to enhance its value proposition is also necessary. This can include researching content filtering machine learning algorithms to provide personalized recommendations based on customer feedback, as well as artificial intelligence techniques such as natural language processing or computer vision to improve user experience by enabling voice or image recognition features.

Chapter 05
SYSTEM ARCHITECTURE

5.1 ER DIAGRAM

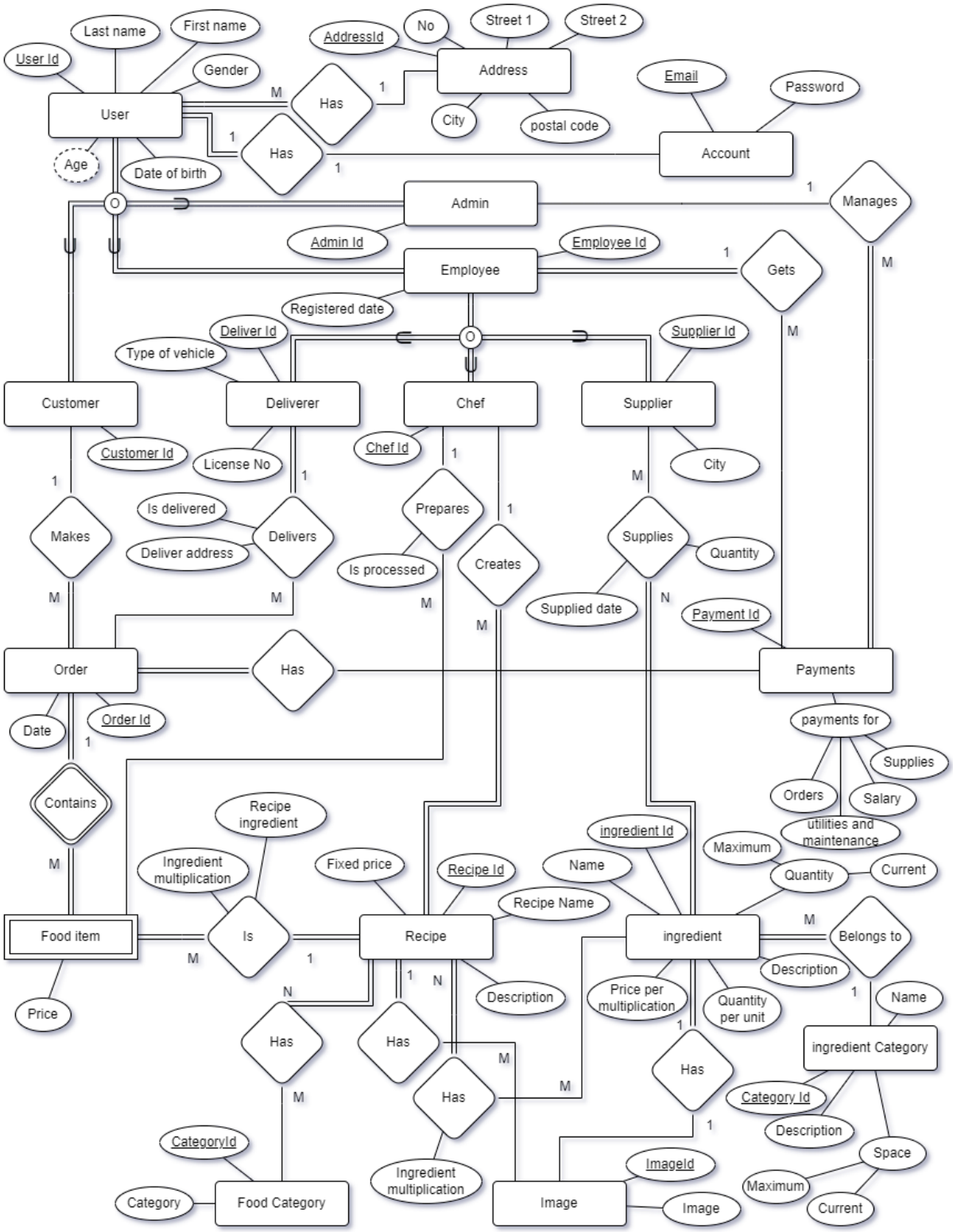


Figure 5. 1 ER Diagram.

5.2 CLASS DIAGRAMS

5.2.1 MODELS

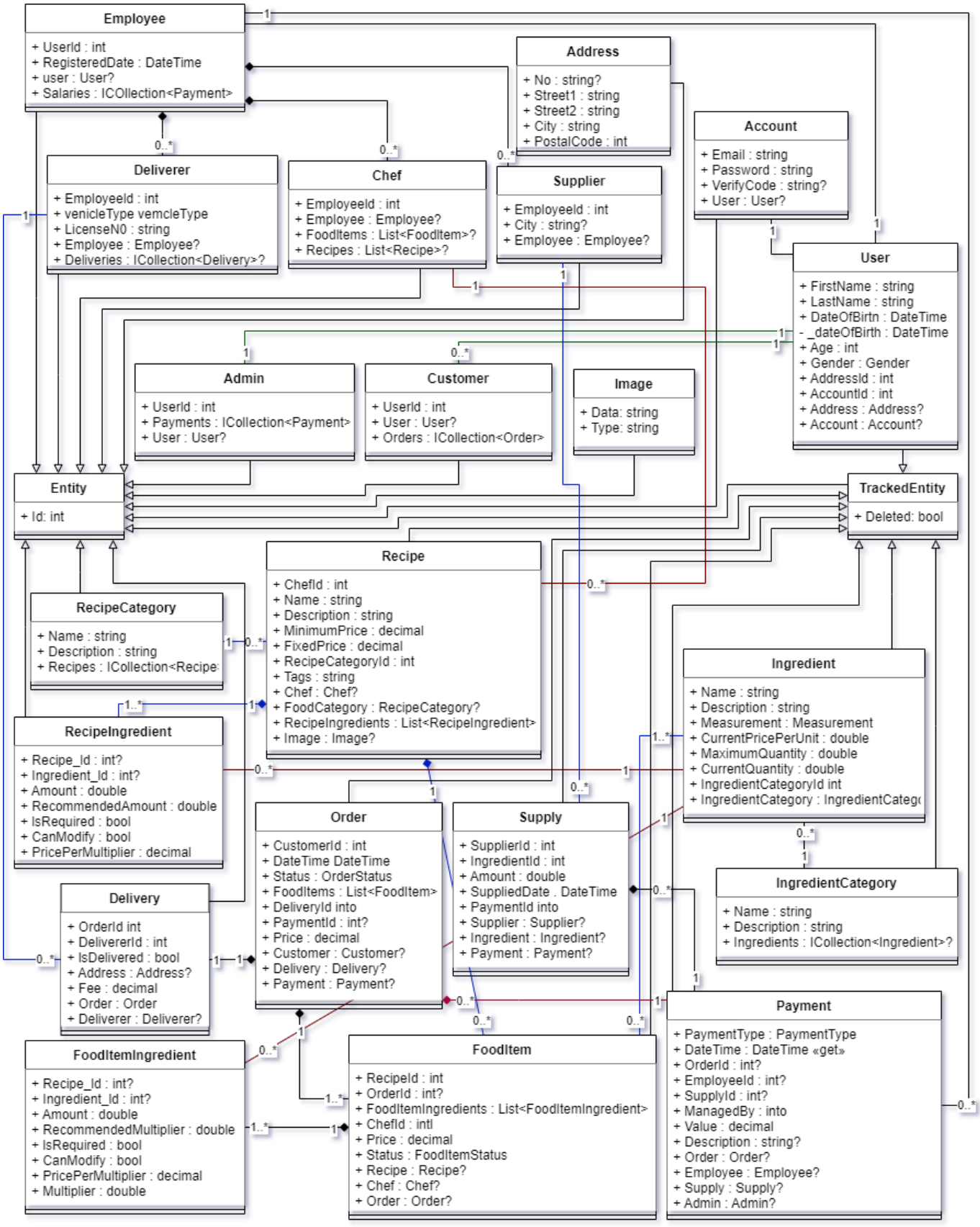


Figure 5. 2 Models Class Diagram.

The class diagram represents a model for managing each entity in figure 5.1 in the system, which is built using Entity Framework Core. This is a class library that can be used across the system.

The Entity class serves as a base class for all entities in the project, and provides a single property, Id, which represents the primary key for each entity. By inheriting from this base class, all entities in the project can access this property, simplifying the implementation of the generic repository pattern which will be further discuss in the document. In the context of the generic repository pattern, exposing the primary key property allows for easy execution of queries that are specific to a given entity. For example, a query to retrieve a single entity by its primary key can be executed using a simple LINQ expression Where “T” is the entity type being queried. T can be any entity in figure 5.1/5.2

```
public T GetById(int id) {
    return _dbContext.Set<T>().Find(id);
}
```

Overall, the Entity class serves as a useful tool for implementing the generic repository pattern and simplifying database operations across all entities in the project.

TrackedEntity is a class that can be used to implement soft delete functionality for any entity class that inherits from it. Soft delete is a pattern used to mark a record as deleted instead of deleting it from the database. This allows the record to be easily restored if needed and makes it possible to track when a record was deleted. TrackedEntity contains a Boolean property named IsDeleted, which is set to true when the entity is deleted. This property can be used to filter out deleted records when querying the database.

```
public T DeleteById(int id) {
    T entiy = _dbContext.Set<T>().Find(id);
    entiy.Deleted = true;
    _dbContext.SaveChanges();
}
```

5.2.1 DATA ACCESS LAYER (DAL)

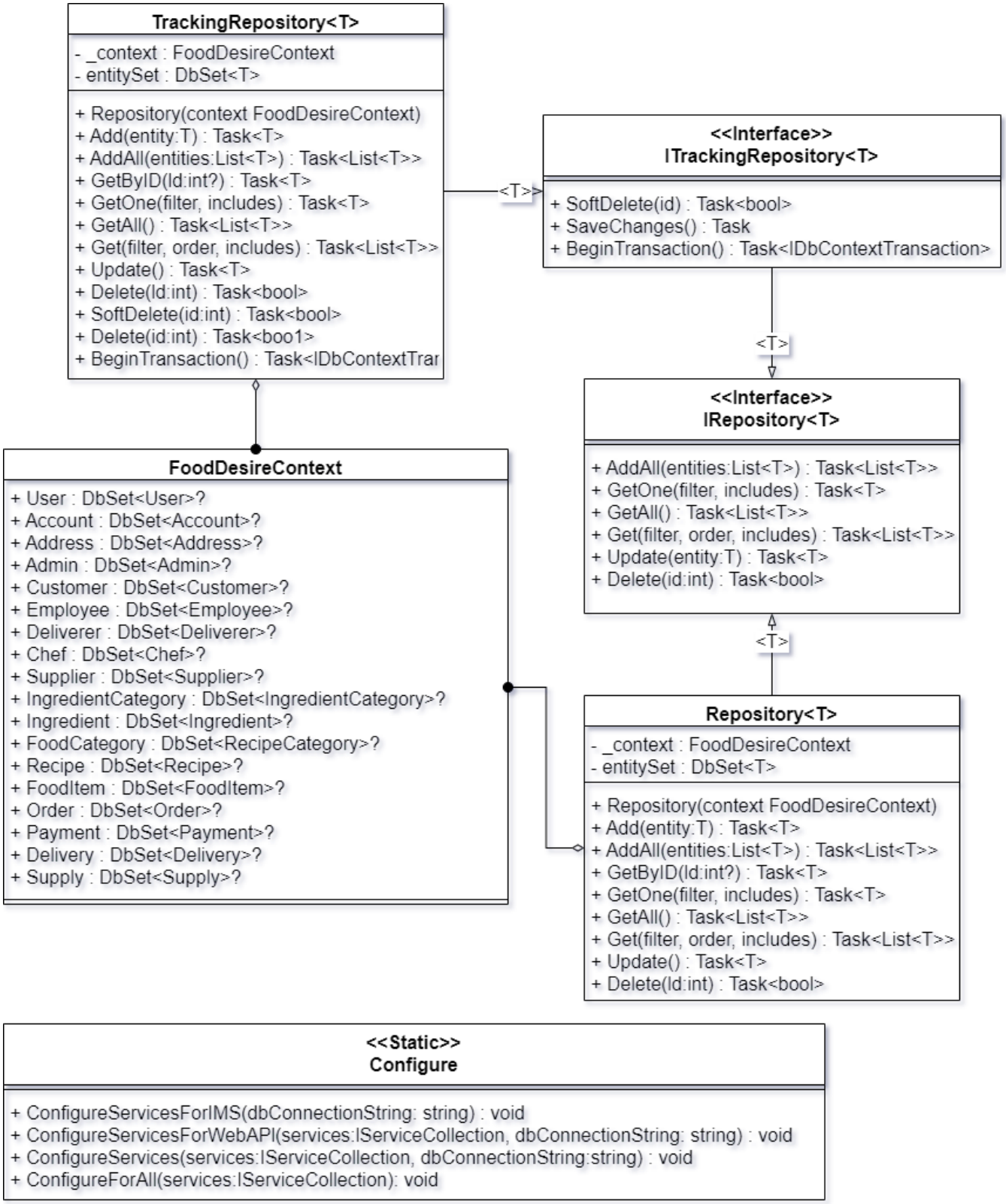


Figure 5. 3 DAL Class Diagram.

The class diagram (Figure 5.3) represents the data access layer (DAL) for a food ordering system. The FoodDesireContext class is for Entity Framework and contains properties for each entity (e.g., User, Account, Address, Admin, etc.) that can be persisted in the database.

Repositories are used as an abstraction layer between the data access logic and the business logic of an application, which is shown by the IRepository<T> interface and it is implemented in the Repository<T> class. The Repository<T> class provides methods for CRUD (Create, Read, Update, Delete) operations on the entities, which are executed on the DbContext instance.

There is also a ITrackingRepository<T> interface and it is implemented in the TrackingRepository<T> class that extends the IRepository<T> interface and provides additional methods for soft delete and transaction management.

Finally, the Configure class is used for dependency injection to configure the services that the application needs. This allows for loose coupling and better testability of the code. It defines the mappings between interfaces and implementations, which allows the application to configure dependencies at runtime.

5.2.2 CORE LAYER

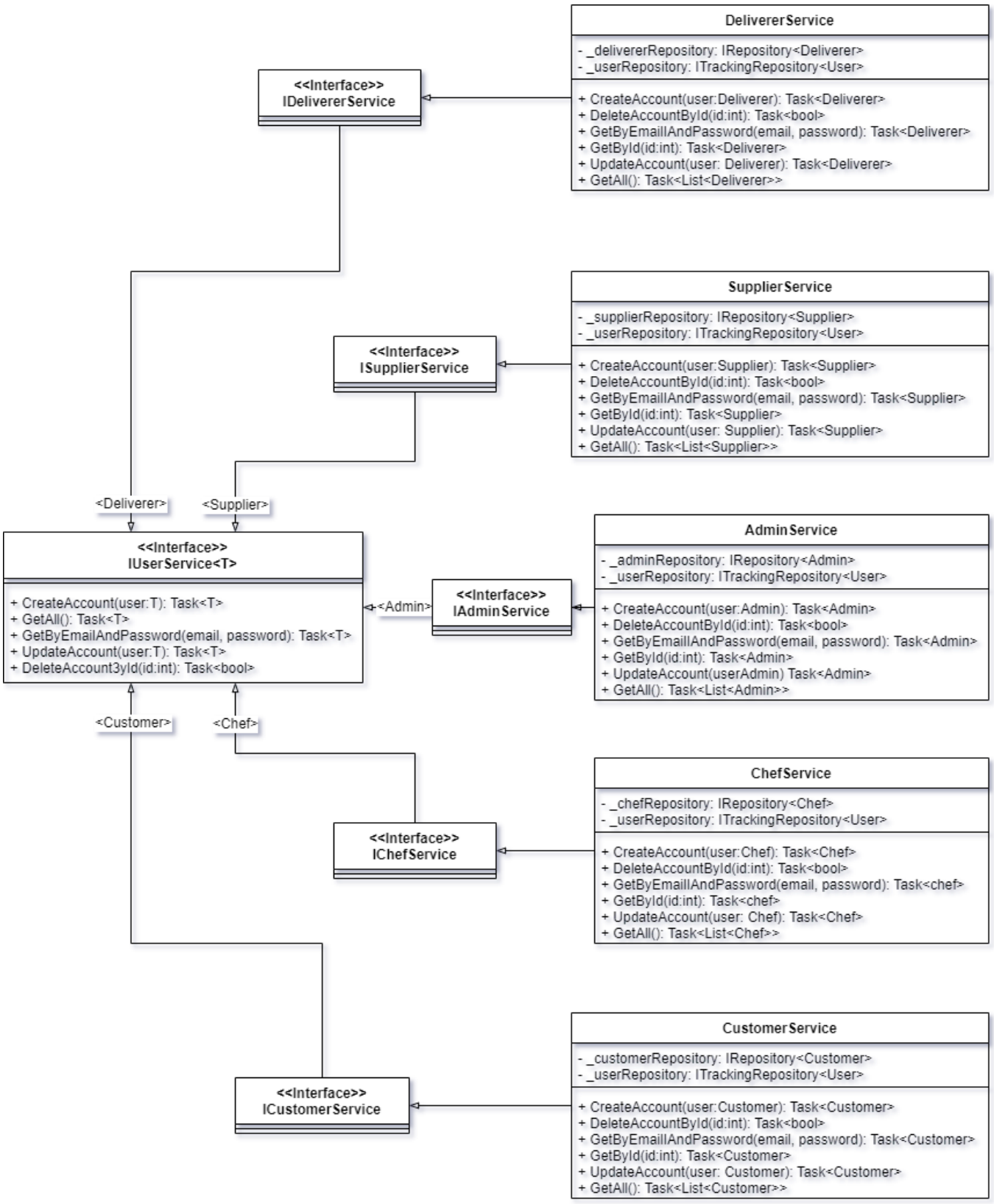


Figure 5. 4 User Services Diagram. (Inheritance only).

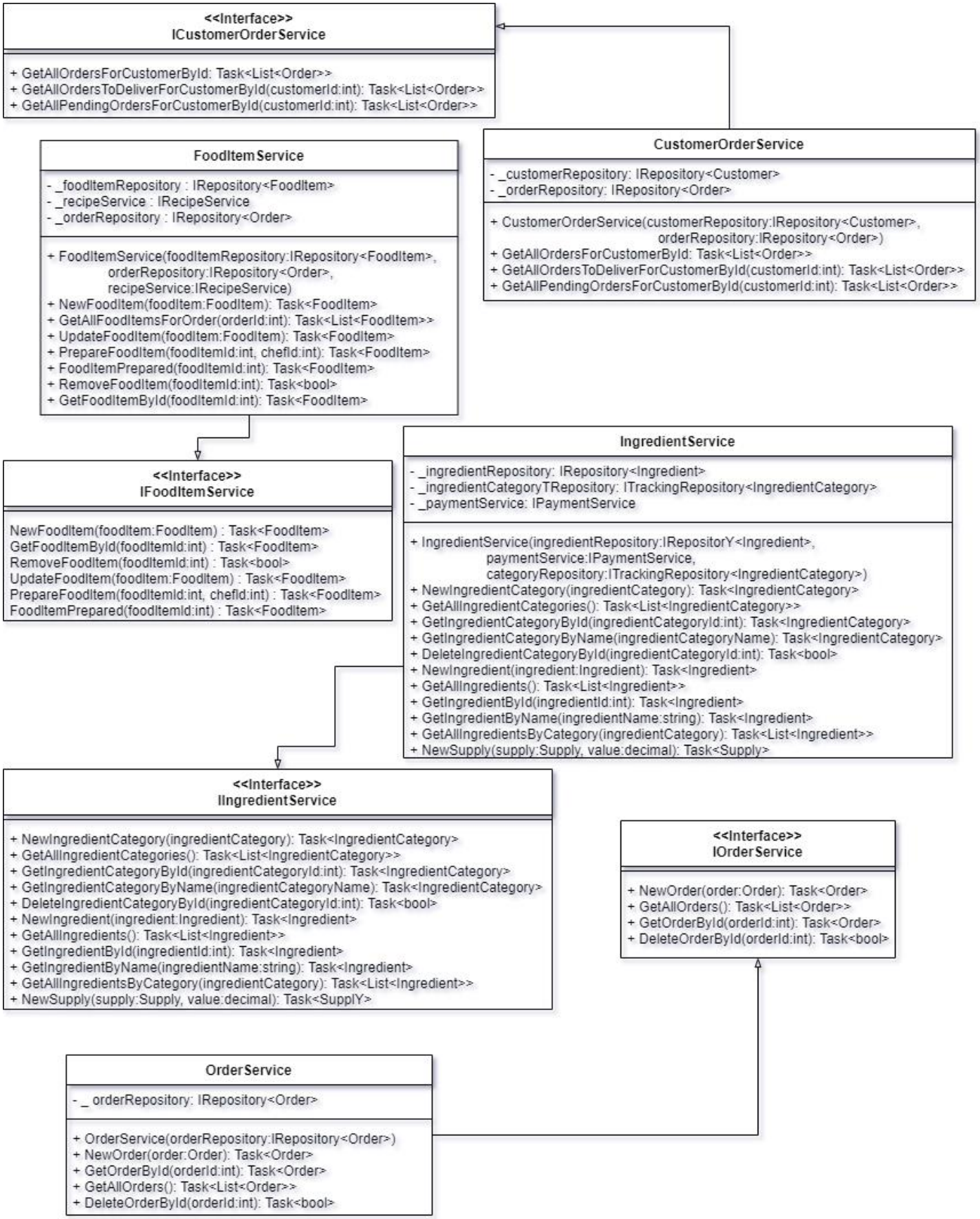


Figure 5. 5 Core Services 1. (Inheritance only).

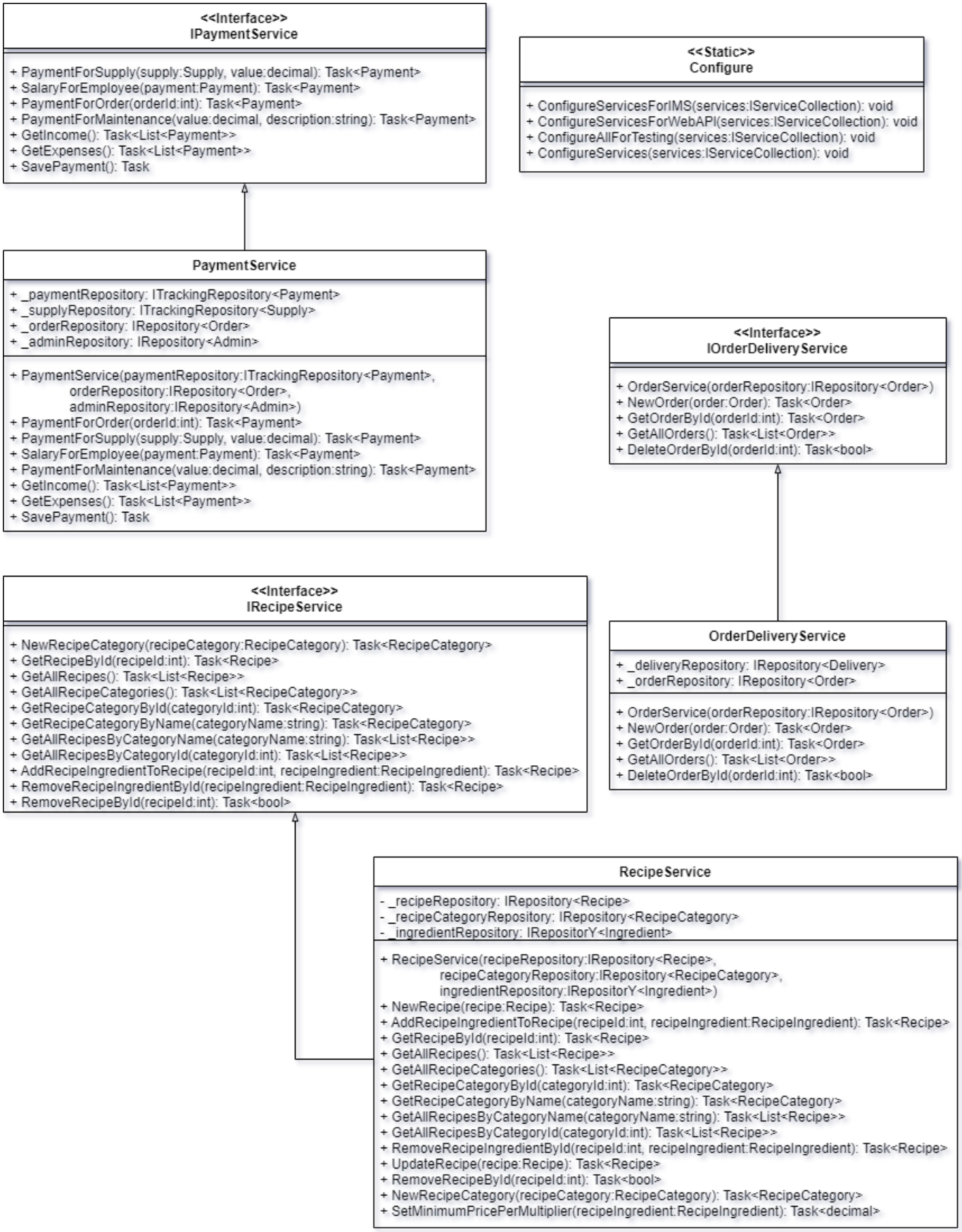


Figure 5. 6 Core Services 2. (Inheritance only).

Associations and dependencies for figure 5.4, 5,5, 5,6 as follows.

- AdminService depends on IRepository and ITrackingRepository interfaces and has a composition relationship with the Admin entity.
- ChefService depends on IRepository and ITrackingRepository interfaces and has a composition relationship with the Chef entity.
- CustomerService depends on IRepository and ITrackingRepository interfaces and has a composition relationship with the Customer entity.
- DelivererService depends on IRepository and ITrackingRepository interfaces and has a composition relationship with the Deliverer entity.
- SupplierService depends on IRepository and ITrackingRepository interfaces and has a composition relationship with the Supplier entity.
- FoodItemService depends on IRepository and IRecipeService interfaces and has a composition relationship with the FoodItem entity.
- CustomerOrderService depends on IRepository interface and has a composition relationship with the Customer and Order entities.
- IngredientService depends on IRepository, ITrackingRepository, and IPaymentService interfaces, and has a composition relationship with the ingredient entity.
- OrderDeliveryService depends on IRepository interface and has a composition relationship with the Delivery entity.
- OrderService depends on IRepository interface and has a composition relationship with the Order entity.
- PaymentService depends on IRepository, ITrackingRepository, and IPaymentService interfaces, and has a composition relationship with the Payment entity.
- RecipeService depends on IRepository interface, and has a composition relationship with the Recipe, RecipeCategory, and Ingredient entities.

The class diagram (figures 5.4, 5,5, 5,6) depicts several services and their respective interfaces that are used in the system. Each of these services implements a corresponding service interface, which defines the contract for the service and allows for loosely coupled code.

There are several associations and dependencies between the classes in the diagram. For example, the AdminService and ChefService both depend on some Repositories, which is used to access data storage. Additionally, the IngredientService depends on the PaymentService, which provides payment

functionality. The OrderDeliveryService and FoodItemService both depend on the Order Repository, indicating that they need to access and manipulate order data of the order entity in the data store.

The Configure class is a key component in setting up the dependency injection framework for the application, allowing for loose coupling between components and promoting code reuse and maintainability. (As discussed in Chapter 5.2.1)

5.3 HIGH-LEVEL ARCHITECTURAL DIAGRAM

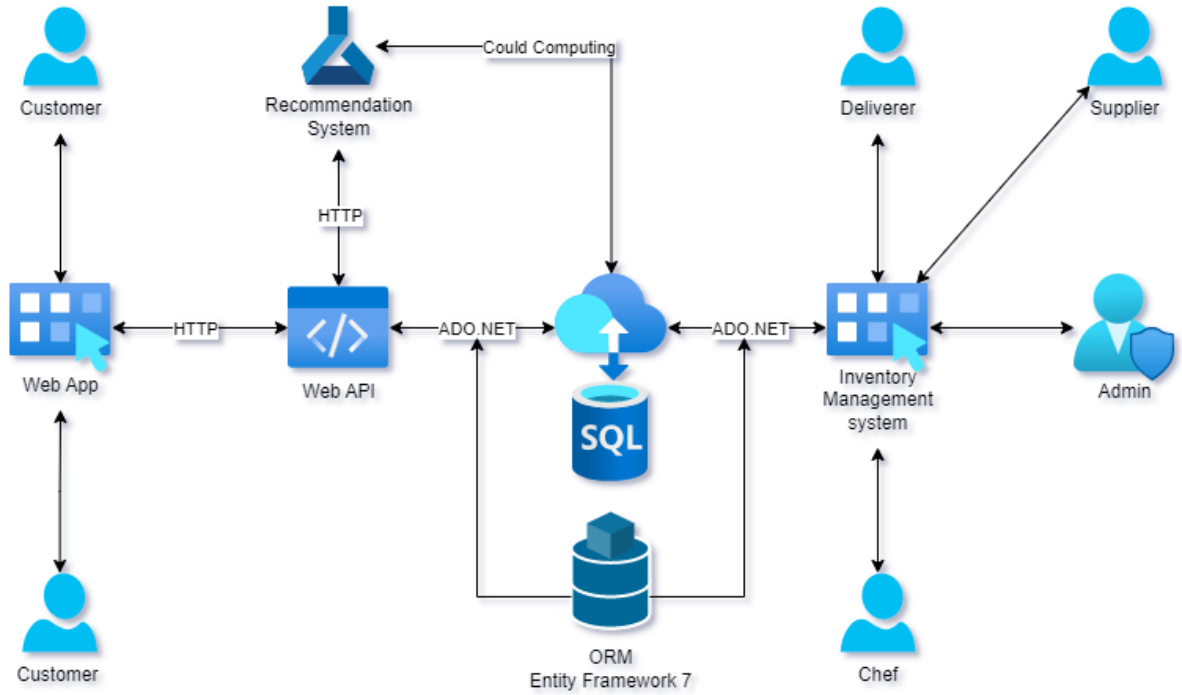


Figure 5. 7 High-level Architecture Diagram.

The architecture diagram (Figure5.7) consists of a distributed system of several components. A Web Application is hosted on Azure, which communicates with an API Layer to access the Cloud Storage. The Web Application serves customers who place orders, edit recipes, and leave reviews. The IMS is managed by Admin, Supplier, Chef, and deliverer. Additionally, there is a Recommendation System that communicates with the Web API over HTTP. Azure Cloud SQL is used as a database to store data related to employees, inventory, orders and recipes. This architecture allows for efficient management of inventory, orders, and employees in a restaurant setting.

5.4 PROJECT STRUCTURE

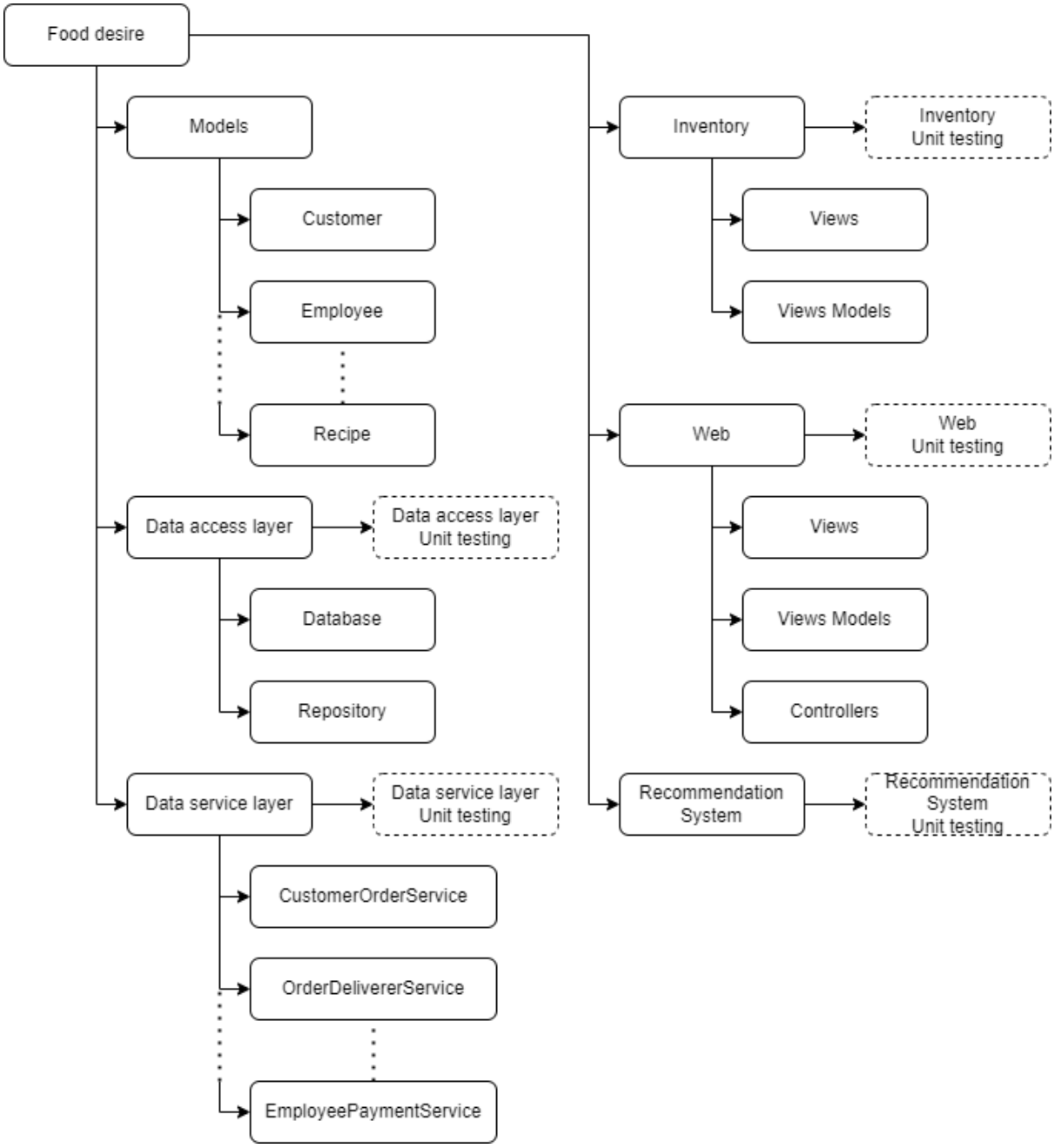


Figure 5. 8 Project Structure.

Chapter 06
DEVELOPMENT TOOLS AND TECHNOLOGIES

6.1 DEVELOPMENT METHODOLOGY

Requirements gathering is a crucial step in any software development project. It involves communicating with the stakeholders who have an interest in the system, such as customers, users, managers, and developers. The goal is to be ensure of their needs, expectations, and constraints for the system and document them clearly. A requirements specification document should be created that defines the scope, objectives, functionality, quality attributes, and acceptance criteria of the system. (Ref: Project initialization document, Chapter 03.)

Design is the next step after requirements gathering. It involves creating a high-level plan for how the system will be structured and implemented. The design should specify the system architecture and components, such as classes, interfaces, modules, and patterns. A high-level design document should be created that describes the rationale behind the design decisions and how they meet the requirements. (Chapter 05)

Implementation is the step where the actual coding of the system takes place. The development developer should follow the design document and use best practices for writing clean and maintainable code. The code should be well-commented and documented to facilitate understanding and reuse. The code should also adhere to coding standards and conventions to ensure consistency and quality. Correct use of good design pattern will help for the further development of the system.

Testing is the step where the system is verified and validated against the requirements. Testing should be done throughout the development process to ensure that each feature works as expected and meets its acceptance criteria. Testing can involve both manual and automated methods, such as unit testing, integration testing, system testing, performance testing, usability testing etc.

Integration and deployment are the step where all the features are combined into a complete system and delivered to a production environment. Integration involves ensuring that all components work together seamlessly without any

errors or conflicts. Deployment involves installing or uploading the system to a server or platform where it can be accessed by end-users.

Maintenance is the final step in software development lifecycle. It involves providing ongoing support for the system after it has been deployed. Continues integration and continues development techniques can be implemented for the system to improve the maintainability. Maintenance can include,

- Fixing bugs.
- Applying security patches.
- Adding new features.
- Enhancing existing features or improving performance.

This development methodology is based on the classic waterfall model, but it can be adapted to incorporate agile principles if desired. The key is to have a clear process in place for each stage of development, from requirements gathering to maintenance.

6.2 PROGRAMMING LANGUAGES AND TOOLS

The FoodDesire application is developed using the C# programming language, and its backend was created using the .NET 7 and Entity Framework 7 frameworks. For coding and debugging purposes, Visual Studio and Visual Studio Code being used for as development environments (IDEs).

Database management and querying are carried out using either SQL Server Management Studio or Azure Data Studio.

To ensure efficient version control, Git is used as the version control system.

To facilitate API documentation, Swagger is implemented, which provided a user-friendly interface for developers to explore and understand the different endpoints available in the application.

For project management and CI/CD pipelines, Azure DevOps being used to streamline the software development process and automate the deployment of the application. GitHub Action will also be used in the CI/CD pipelines.

In summary, the FoodDesire application was built using C# programming language and the .NET Core and Entity Framework Core frameworks, along with Visual Studio or Visual Studio Code for IDEs, SQL Server Management Studio or Azure Data Studio for database management, Git for version control, Azure DevOps for project management and CI/CD pipelines, and Swagger for API documentation.

6.3 THIRD PARTY COMPONENTS AND LIBRARIES

The system leverages several third-party components and libraries to deliver a robust and user-friendly solution. Entity Framework 7 (EF7) is the core component of the system as it provides a lightweight and extensible way to access data from various sources using C#. EF7 supports different types of databases, such as relational, non-relational, and in-memory. EF7 also enables code-first development, migrations, change tracking, and query optimization.

Microsoft.Extensions.Hosting, which is a library that simplifies the initialization and configuration of applications. Microsoft.Extensions.Hosting allows dependency injection to register services and components that can be resolved by the application at runtime.

The system also uses NUnit testing to ensure the quality and reliability of the code. NUnit is a unit-testing framework for .NET that supports parameterized tests, assertions, attributes, and test runners.

The IMS uses WinUI 3 library, which is a modern and fluent user interface framework for Windows desktop applications. WinUI 3 provides a consistent and adaptive design across various devices and platforms. The IMS also follows the Model-View-ViewModel (MVVM) design pattern to ensure a clear separation of concerns between the data model, the user interface, and the business logic. To facilitate the implementation of MVVM, the system uses MVVM Community Toolkit, which is a collection of helpers, extensions, converters, behaviors, and controls for MVVM development.

Another key component of the system is the Web UI, which allows customers to browse and order products online. The Web UI is built using Blazor framework, which enables web development using C#. Blazor offers several benefits such as

full-stack development with a single language, fast performance with Web Assembly runtime, rich interactivity with JavaScript interoperability, and code reuse with .NET libraries.

Swagger is a library that helps developers design, build, document, and test RESTful APIs. Swagger provides tools for generating interactive documentation from API specifications, validating API requests and responses, and testing API endpoints.

ML.NET is a machine learning framework for .NET developers that can be used to build recommendation systems. This will be used to develop the recommendation system of the Food Desire application.

By using these third-party components and libraries, the system achieves a high level of functionality and usability for both internal and external users.

6.4 ALGORITHMS

Content-based filtering: This algorithm can be used to recommend food items to users based on their previous orders or other preferences. The algorithm analyzes the content of food items and recommends similar items to users. For instance, if a user orders a lot of spicy food, the system may recommend other spicy food items to that user.

Collaborative filtering: This algorithm analyzes user behavior and interactions to recommend food items. The system can use this algorithm to identify patterns in user behavior and suggest food items that are popular among similar users.

Hybrid algorithm: A combination of content-based filtering and collaborative filtering can be used to provide more accurate recommendations to users. This algorithm analyzes both user preferences and similar user behavior to recommend food items.

Search algorithm: The system can use a search algorithm to allow users to search for specific food items or ingredients.

Overall, the algorithms used in the food desire application should be selected based on the specific requirements of the system and the data available. Entity Framework 7 can be used to retrieve data from the Azure Cloud SQL database and perform the necessary calculations to generate recommendations or perform searches.

Chapter 07
DISCUSSION

7.1 OVERVIEW OF THE INTERIM REPORT

The project aims to develop a web-based application using .NET framework that allows customers to order food online from the restaurants and customize their meals. The project also intends to implement a recommendation system that suggests meal items to the users based on their preferences and previous orders.

7.2 SUMMARY OF THE REPORT

Using a waterfall model methodology, the report outlines six stages of development that are crucial for the successful implementation of an online food ordering system. The first stage is requirements gathering, where the developer identifies and documents the functional and non-functional requirements for both customer and restaurant ends of the system. In the design stage, the developer uses the requirements to create detailed design specifications for the system, including the database schema, user interface design, and system architecture.

During the implementation stage, the developer begins coding the system using .NET framework, following best practices, and coding standards. The system is then tested in the next stage, where the developer ensures that all requirements have been met and that the system is functioning as expected. In the integration and deployment stage, the developer deploys the system to a web server using tools such as Apache software.

Finally, the developer must ensure that the system is maintained, by fixing bugs, applying security patches, adding new features, and ensuring the system is running smoothly. The report also provides some details on how each stage can be performed using various tools and techniques. For instance, in the testing stage, the developer can use tools such as NUnit and GitHub action to perform unit and integration testing, respectively.

7.3 CHALLENGES FACED

- Finding reliable sources of information on existing online food ordering systems

- Defining clear and consistent requirements for both customer end and restaurant end
- Testing the currently developed features thoroughly to ensure quality and functionality.
- Integrating currently developed components smoothly into one coherent system

7.4 FUTURE PLANS / UPCOMING WORK

Great progress has been made on the project with the completion of the DAL and CORE layers, as well as the unit testing for both layers. The next phase of the project will involve implementing the IMS UI, which will provide a user-friendly interface for managing inventory. Additionally, the developer will work on creating the web API and web UI, which will enable users to interact with the system and place orders online. Moreover, more research should be conducted to implement the recommendation system.

Overall, these upcoming works represent an exciting opportunity to bring the online food ordering system project to finished. The goal is to deliver a high-quality system that meets the needs of customers and restaurant.

References

Ahrens, S. (2012) ‘Recommender Systems’, in.

Daim, T.U. *et al.* (2013) ‘Exploring technology acceptance for online food services’, *Int. J. Bus. Inf. Syst.*, 12, pp. 383–403.

Gao, F. and Su, X. (2018) ‘Omnichannel Service Operations with Online and Offline Self-Order Technologies’, *Manag. Sci.*, 64, pp. 3595–3608.

Goffe, L. *et al.* (2021) ‘Appetite for Disruption: Designing Human-Centred Augmentations to an Online Food Ordering Platform’, in *34th British Human Computer Interaction Conference Interaction Conference, BCS HCI 2021*. BCS Learning and Development Ltd., pp. 155–167. Available at: https://doi.org/10.14236/ewic/HCI2021.16.

R., A. *et al.* (2017) ‘Online Food Ordering System’, *International Journal of Computer Applications*, 180(6), pp. 22–24. Available at: https://doi.org/10.5120/ijca2017916046.

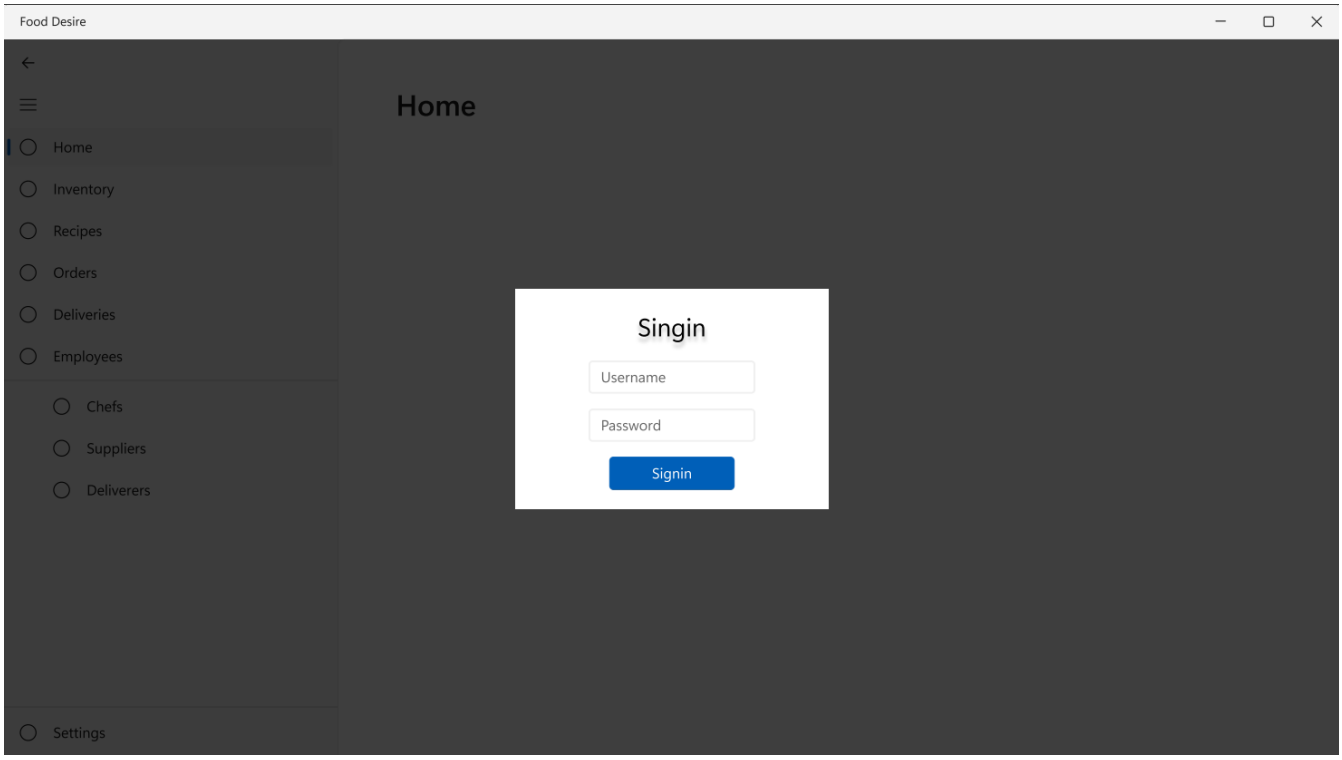
Ratto, F. *et al.* (2008) ‘A numerical approach to quantify self-ordering among self-organized nanostructures’, *Surface Science*, 602, pp. 249–258.

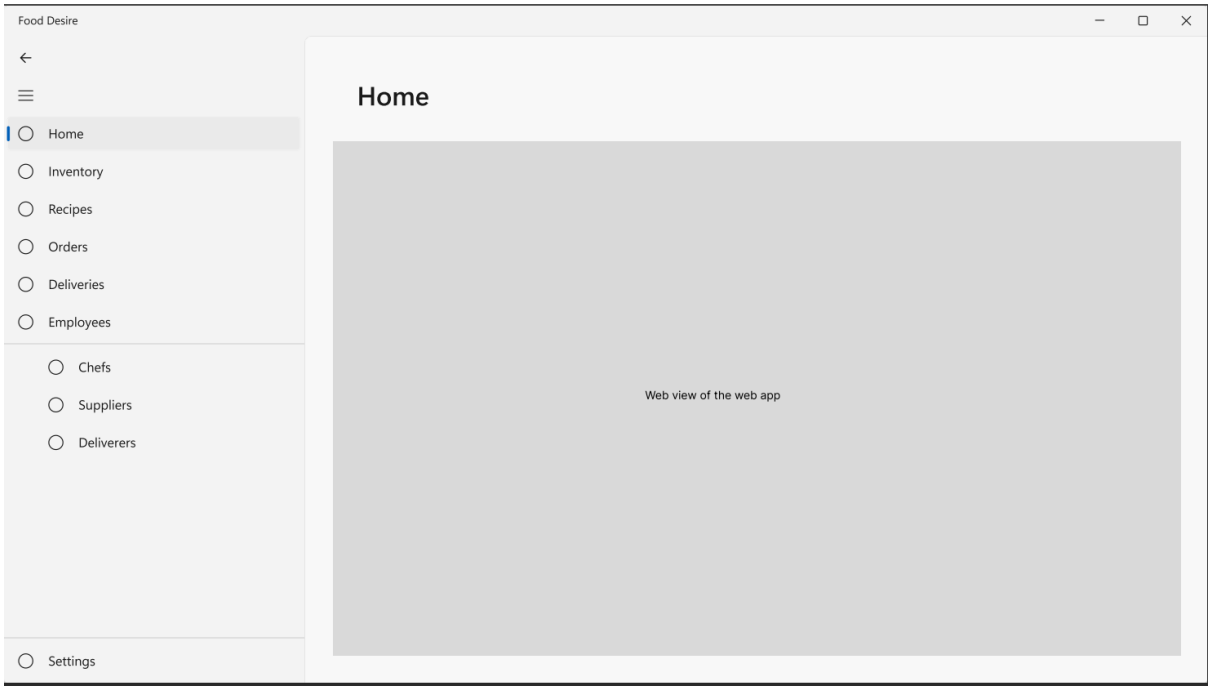
Thomas, A. and Davis, S.E. (1978) ‘Laboratory ordering: a system which eliminates paperwork.’, *The American journal of medical technology*, 44 10, pp. 1026–7.

Zulkarnain, K. *et al.* (2015) ‘Key success factors of online food ordering services:an empirical study’, in.

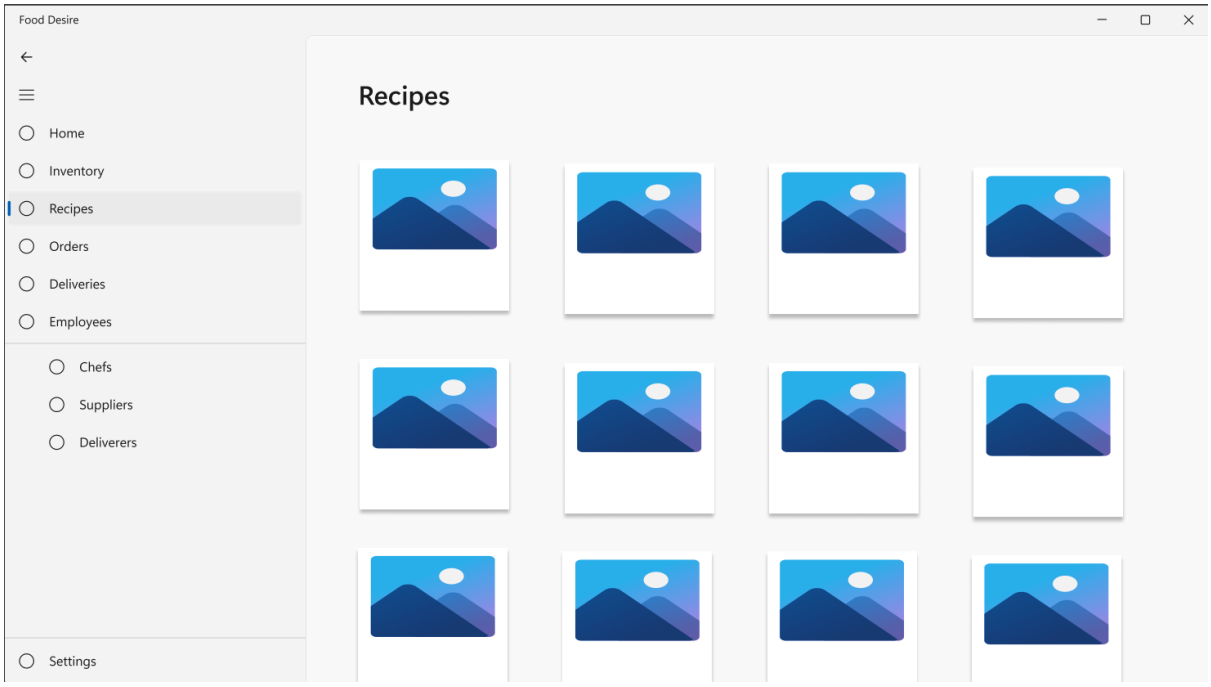
Appendixes

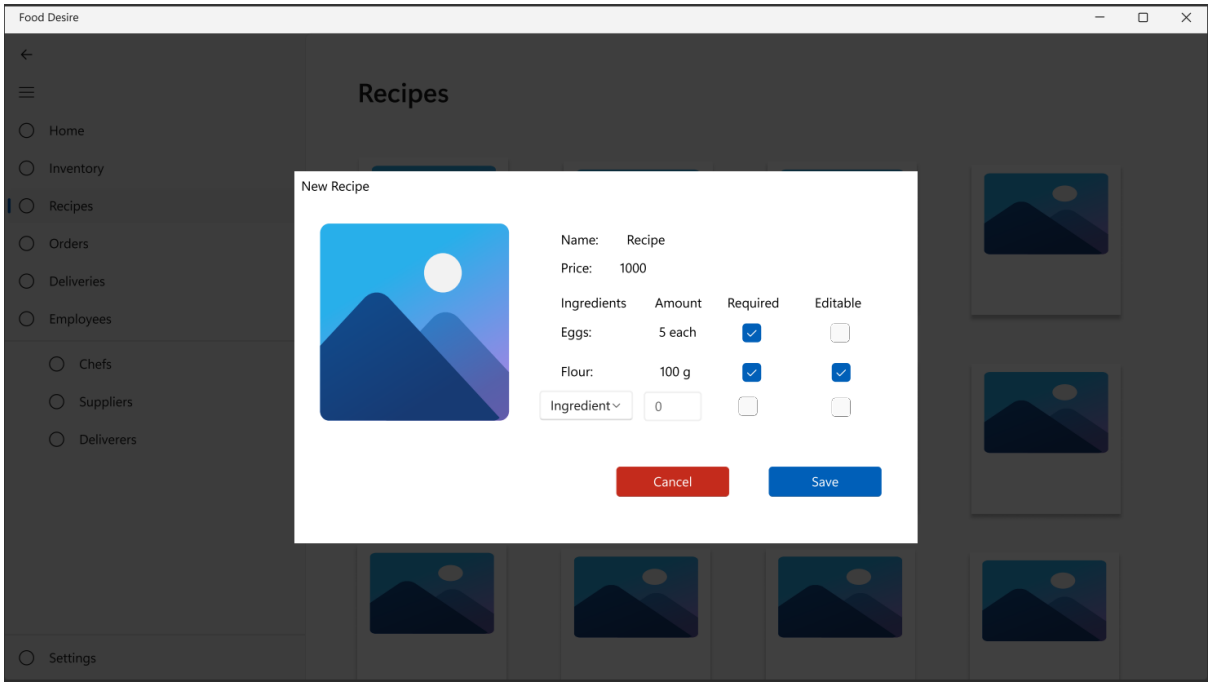
INVENTORY MANAGEMENT SYSTEM WIREFRAME.

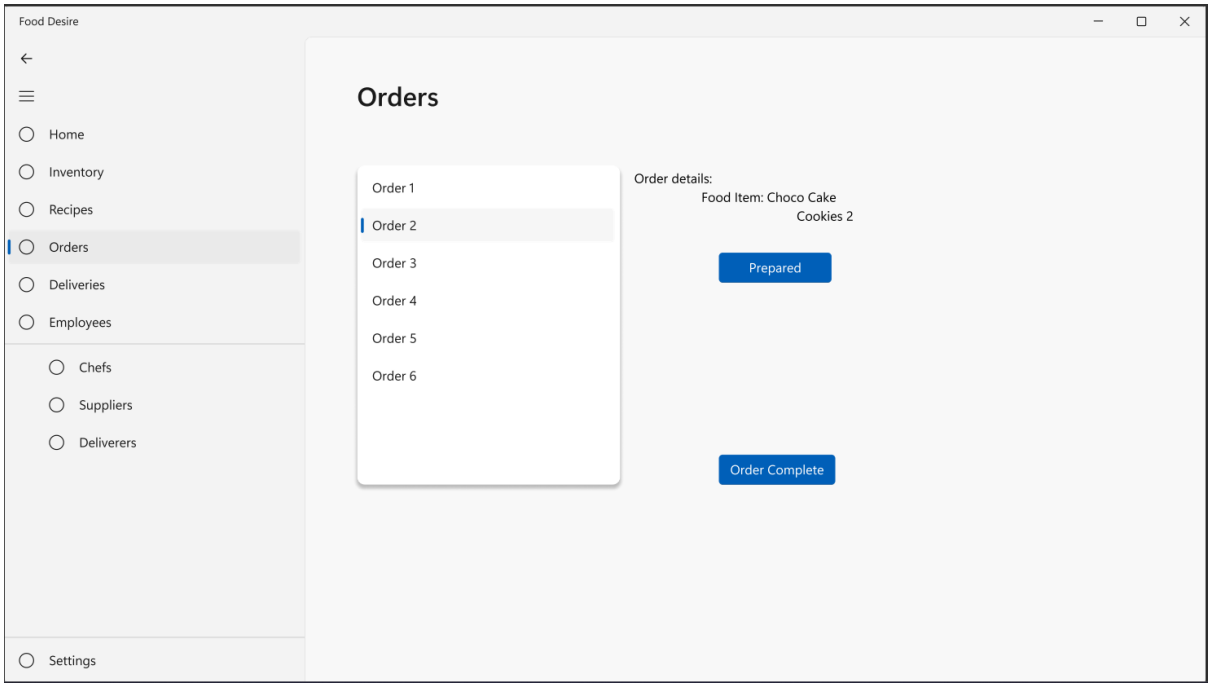


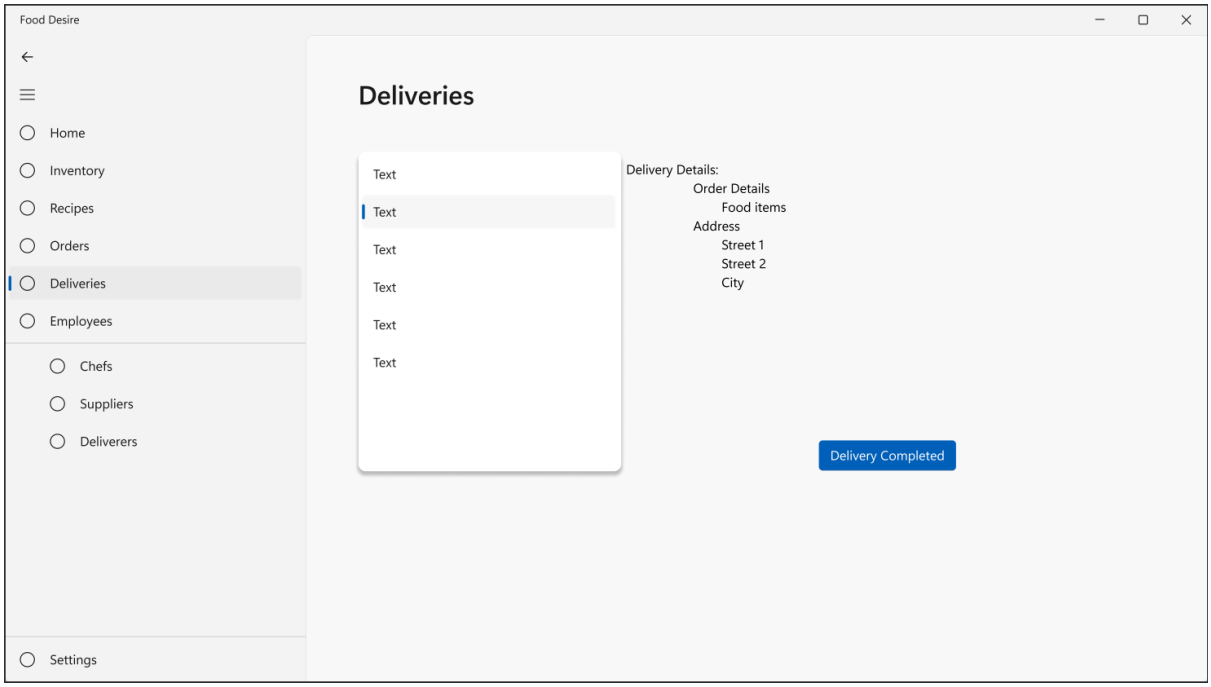


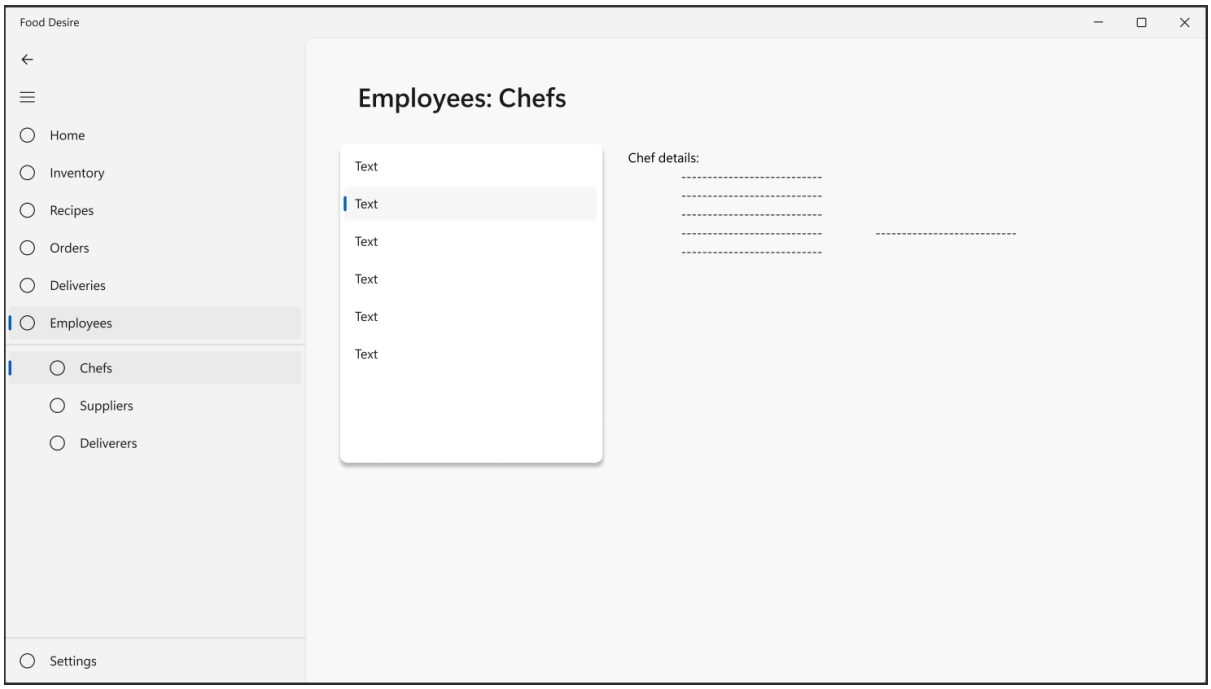

Food Desire
Inventory
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings


Food Desire
Recipes
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings


Food Desire
Recipes
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings
New Recipe
Name: Recipe
Price: 1000
Ingredients
Amount
Required
Editable
Eggs:
5 each
Flour:
100 g
Ingredient
0
Cancel
Save


Food Desire
Orders
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings
Order 1
Order 2
Order 3
Order 4
Order 5
Order 6
Order details:
Food Item: Choco Cake
Cookies 2
Prepared
Order Complete


Food Desire
Deliveries
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings
Text
Text
Text
Text
Text
Text
Delivery Details:
Order Details
Food items
Address
Street 1
Street 2
City
Delivery Completed


Food Desire
Employees: Chefs
Home
Inventory
Recipes
Orders
Deliveries
Employees
Chefs
Suppliers
Deliverers
Settings
Text
Text
Text
Text
Text
Text
Chef details:

END