****

**Project Report**

**Title:**

Go4Food

**Group Name:**

Go4Food

**Group Members:**

Mohammad Tamanna | Mohammad.Tamanna@outlook.com

JJ Papeleras | justinejpapeleras@gmail.com

Adnan Uddin Mohammed | ami949@uregina.ca

**Department of Computer Science**

**CS 476-02: Software Development Project**

Spring/Summer 2023

Sirvan Parasteh

**21st July 2023**

****

**Table of Contents**

Table of Contents 2

1. Project Title 3

2. Problem Description 3

3. Problem Statement 3

3.1 Problem Definition 5

3.2 Project Motivations 5

3.3 Application Benefits 5

4. Functional Requirements 5

4.1 Functional Requirements List and Table 5

4.2 Implemented Requirements 5

4.3 Use Case Diagrams 5

4.3.1 Customer Diagram 5

4.3.2 Restaurant Diagram 5

4.4 Activity Diagram 5

5.4.1 Customer 5

5.4.2 Restaurant Registration and Menu Management 5

5.4.3 Restaurant Order Process 5

4.5 Software Qualities 5

5.5.1 Correctness 5

5.5.2 Time-Efficiency 5

5.5.3 Robustness 5

5. Non-Functional Requirements

6. Top-level and Low-level Software Design

6.1 MVC Architecture

****

6.2 Design Patterns

6.2.1 Observer

6.2.2 Factory

7.3 Comprehensive Class Diagram

7. Acceptance Testing

7.1 Correctness

7.2 Time-Efficiency

7.3 Robustness

8. Development Environment

8.1 Software tools

8.2 Programming languages

8.3 Project folders

9. Project Members

9.1 Roles and Responsibilities

9.2 Project Reporting

10. Project Timeline

10.1 Project Milestones

10.2 Gantt Chart

11. Risks and Constraints

11.1 Risks

11.2 Constraints

12. High Level Diagram

****

**1. Project Title**

Go4Food - an online food ordering system application.

**2. Project Description**

Develop a web-based application that allows users to order food from various restaurants. Implement features such as user registration, menu browsing, placing orders, and order tracking. Highlight benefits like convenience, a wide range of restaurant options, and user-friendly interfaces.

**3. Problem Statement**

Traditional food ordering methods often involve lengthy phone calls, miscommunications, and limited menu visibility, leading to inconvenience for customers and operational challenges for restaurant owners. Additionally, managing and tracking orders manually can result in errors, delays, and reduced customer satisfaction. The Go4Food project addresses these issues by providing a user-friendly platform within the e-commerce domain that empowers customers to explore menus, customize orders, and place food orders effortlessly and through unique order frequencies decided and customized by the user; for instance, allowing customers to set an automatic order given by their input time and date.

**3.1 Problem Definition**

Traditional food ordering methods often involve lengthy phone calls, miscommunications, and limited menu visibility, leading to inconvenience for customers and operational challenges for restaurant owners. Additionally, managing and tracking orders manually can result in errors, delays, and reduced customer satisfaction. The Go4Food project addresses these issues by providing a user-friendly platform within the e-commerce domain that empowers customers to explore menus, customize orders, and place food orders effortlessly and through unique order frequencies decided and customized by the user.

****

Go4Food’s online food ordering system project aims to address the challenges and limitations present in the traditional food ordering process. The primary objective is to develop a user-friendly web-based platform that streamlines the ordering experience and enhances customer satisfaction. By providing an online platform for customers to order from various restaurants, the project seeks to overcome the inconvenience and inefficiencies associated with traditional methods such as phone calls or physical visits.

**3.2 Project Motivations**

One of the key motivations behind this project is to offer convenience to customers. The current food ordering process an important issue is addressing customers' need for convenience. By implementing an online system, customers will have the ability to effortlessly browse through menus, customize their orders, and place food orders conveniently from their own homes or any location with internet access. This improved accessibility and convenience will enhance customer satisfaction and attract a wider customer base.

**3.3 Application Benefits**

Go4Food’s online food ordering system offers several significant benefits compared to existing systems such as SkipTheDishes and Uber Eats. By leveraging advanced features and functionalities, the application aims to enhance the customer experience, improve convenience, and provide added value to both customers and restaurants.

One of the standout benefits offered by the Online Food Ordering System is the feature of pre-planning customized orders, which enhances the convenience and efficiency of the food ordering experience. This feature enables customers to effortlessly set up recurring orders based on their preferences, eliminating the need to manually select the same items every time they place an order. By allowing customers to create personalized meal plans that cater to their specific tastes and dietary requirements, the system revolutionizes the way people engage with online food ordering.

With such orders, customers have the flexibility to establish a schedule for their favorite meals, ensuring that they can enjoy their preferred dishes consistently without the hassle of repetitively placing the same order. For instance, customers can create a standing order to be delivered every Monday with their favorite meal or select specific days of the week for different culinary

****

experiences. This feature caters to individuals who appreciate routine and those who wish to save time and effort by automating their regular food orders.

By offering this feature, the Online Food Ordering System fosters a more seamless and personalized customer experience. It recognizes that customers have unique preferences and routines when it comes to their meals and seeks to accommodate those preferences efficiently. Whether it is a weekly meal plan, a daily breakfast order, or a monthly treat, the system allows customers to tailor their food orders according to their specific needs, bringing a sense of customization and convenience to their dining routines.

Moreover, this feature not only benefits customers but also provides advantages to restaurants. By encouraging customers to set up recurring orders, restaurants can enjoy improved operational efficiency and enhanced customer loyalty. Predictable order volumes enable restaurants to streamline their production processes, optimize ingredient inventory management, and allocate resources more effectively. This streamlined approach results in cost savings and increased operational productivity, benefiting both the restaurant and the customer. Another one of the key benefits is the implementation of two-factor authentication (2FA) for every order placed. This security feature provides an additional layer of protection in the event that unauthorized access occurs on a user's account, safeguarding their saved credit card information and personal details.

Two-factor authentication, also known as multi-factor authentication (MFA), is a robust security measure that requires users to provide two separate forms of identification to access their accounts. In the context of the Go4Food, this means that customers will be prompted to enter a second authentication factor, such as a unique verification code sent to their registered mobile device, in addition to their regular login credentials when placing an order.

By implementing 2FA for every order, the online food ordering system significantly reduces the risk of unauthorized usage of a customer's account and ensures the protection of sensitive information. Even if an attacker gains access to a user's account credentials, they would be unable to complete an order without the secondary authentication factor. This extra security layer adds a crucial level of defense against unauthorized access attempts and fraudulent transactions.

****

**4. Functional Requirements**

**5.1 Functional Requirements List and Table**

\*\*TO DO\*\*

**5.2 Implemented Requirements**

• User Registration: Customers can create user accounts by providing their personal information, such as name, email, and password. This allows customers to register and securely access their accounts.

• Browse Restaurants: Customers have the ability to browse through a list of available restaurants on the platform. This feature provides customers with an overview of the restaurants available for ordering.

• Search for Restaurants: Customers can search for specific restaurants based on criteria such as name, cuisine, or location. This functionality enables customers to find restaurants that meet their preferences.

• View Menu: Customers can access the menu of each restaurant, including the list of available dishes and their prices. This allows customers to browse through the menus of restaurants they are interested in.

• Place Order: Customers can place food orders by selecting desired dishes, specifying quantity, and choosing delivery or pickup options. This feature allows customers to complete the order process and submit their requests.

• Track Order: Customers can track the status of their orders in real-time, including updates on preparation, delivery, or pickup. This provides customers with visibility and keeps them informed throughout the order process.

• Order History: Customers have access to their order history, where they can review past orders, reorder previous meals, or rate their dining experiences. This feature allows customers to track their ordering history and make informed decisions for future orders.

• User Reviews: Customers can read and provide reviews and ratings for restaurants and individual dishes to share their dining experiences. This feature promotes transparency and helps customers make informed choices based on other users' feedback.

****

• Pre-Planned Order: Customers can set up pre-planned orders, enabling them to receive the same order on specific days or at regular intervals. This feature allows customers to save time and effort by automating their recurring orders.

• Order Support: Customers have access to customer support channels, such as contacts to the restaurants, to address any inquiries, issues, or complaints related to orders. This feature ensures that customers can receive assistance and resolve any concerns they may have.

• Restaurant Registration: Restaurant owners can register their establishments within the system by providing relevant information such as name, address, contact details, and cuisine type. This allows restaurant owners to create their profiles and join the online food ordering platform.

• Menu Management: Restaurant owners have the ability to manage their menus, including adding, updating, or removing dishes, along with their names, descriptions, prices, and available quantities. This feature empowers restaurant owners to maintain and update their menus easily.

• Order Notification: Restaurant owners receive real-time notifications of incoming orders, providing details of the ordered dishes, customer information, and delivery/pickup instructions. This ensures prompt communication and enables timely order processing.

• Order Processing: Restaurant owners have a system to process orders efficiently, including order confirmation, preparation status updates, and order completion notifications. This feature facilitates smooth order management and timely delivery or pickup.

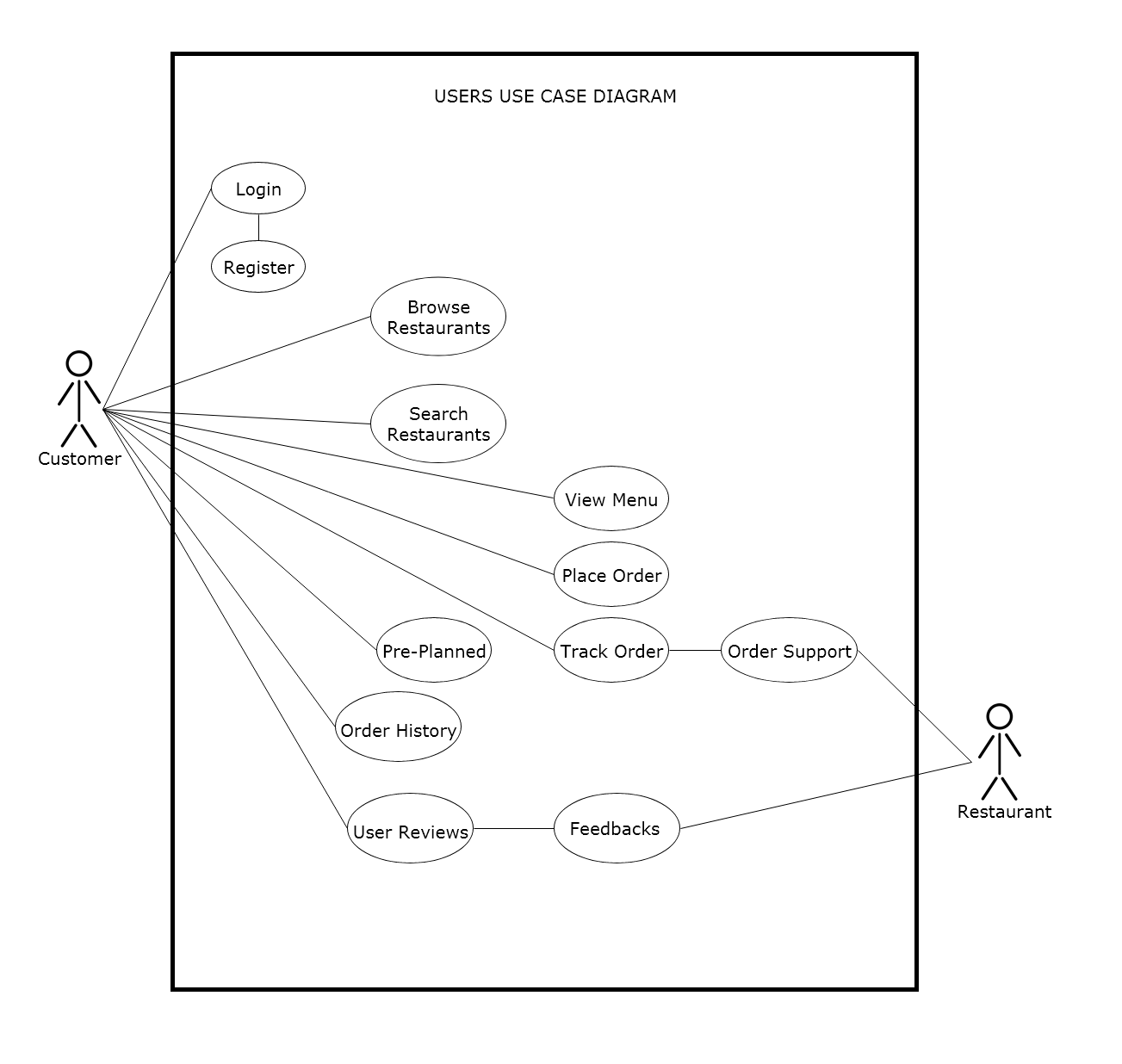
• Restaurant Analytics: Restaurant owners have access to analytics and reports, including order statistics and customer ratings to gain valuable business insights. This enables restaurant owners to make informed decisions and improve their operations.

• Customer Communication: Restaurant owners can communicate with customers regarding order clarifications, special requests, or any other necessary updates. This feature allows for effective communication and ensures a seamless customer experience.

****

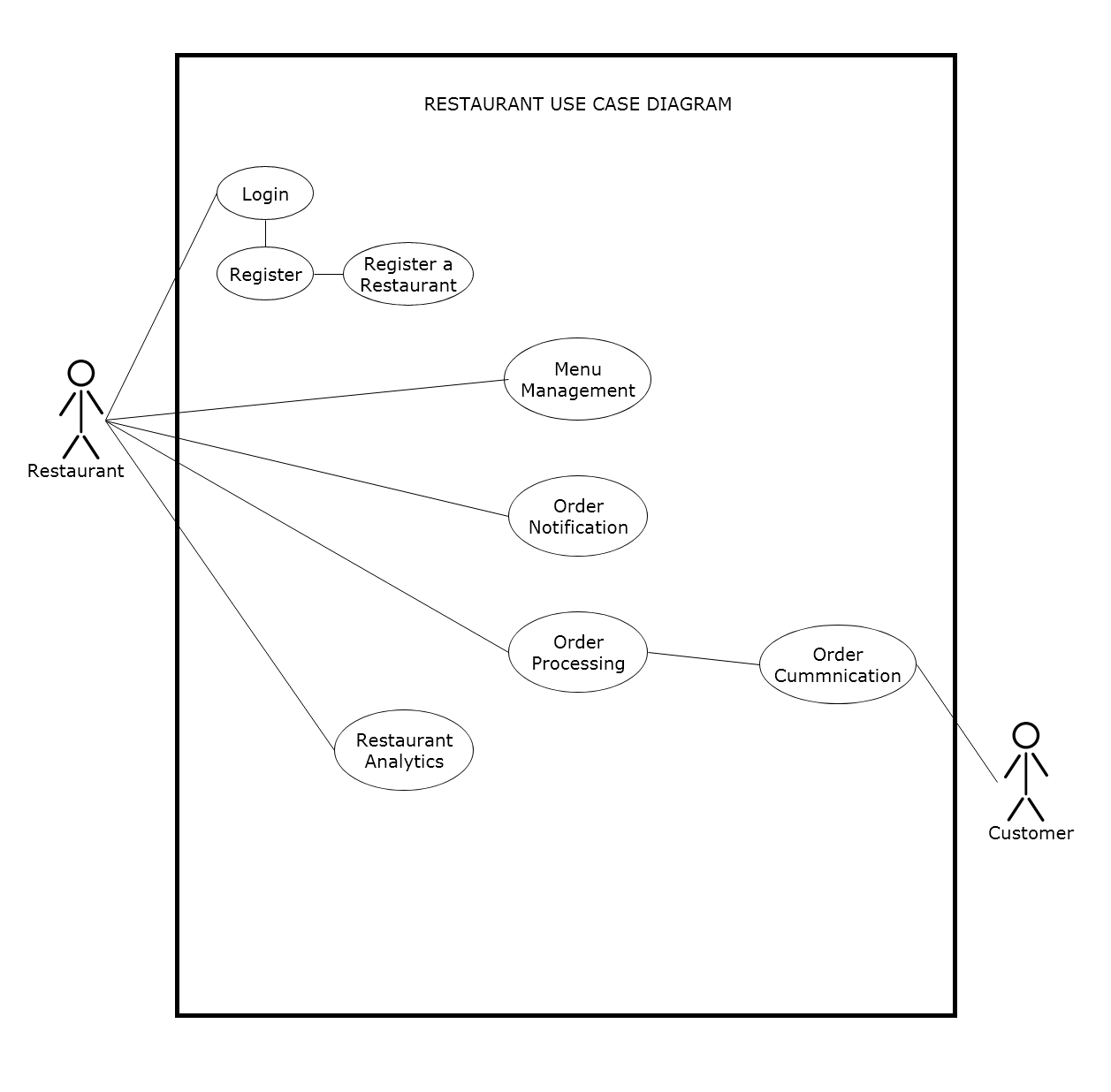
**5.3 Use case diagram**

**5.3.1 Customer Diagram**

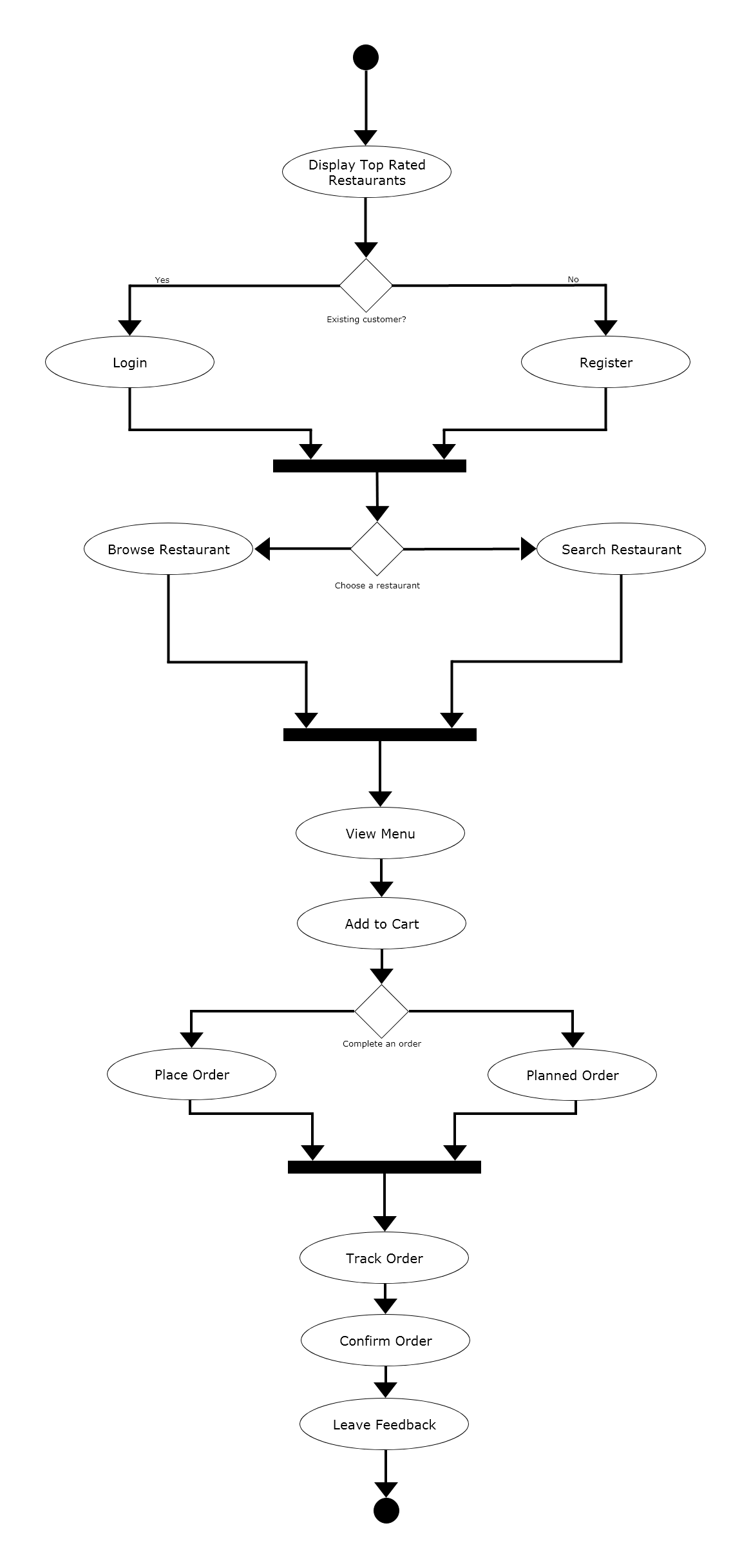


****

**5.3.2 Restaurant Diagram**



****



**5.4 Activity diagram**

**5.4.1 Customer**

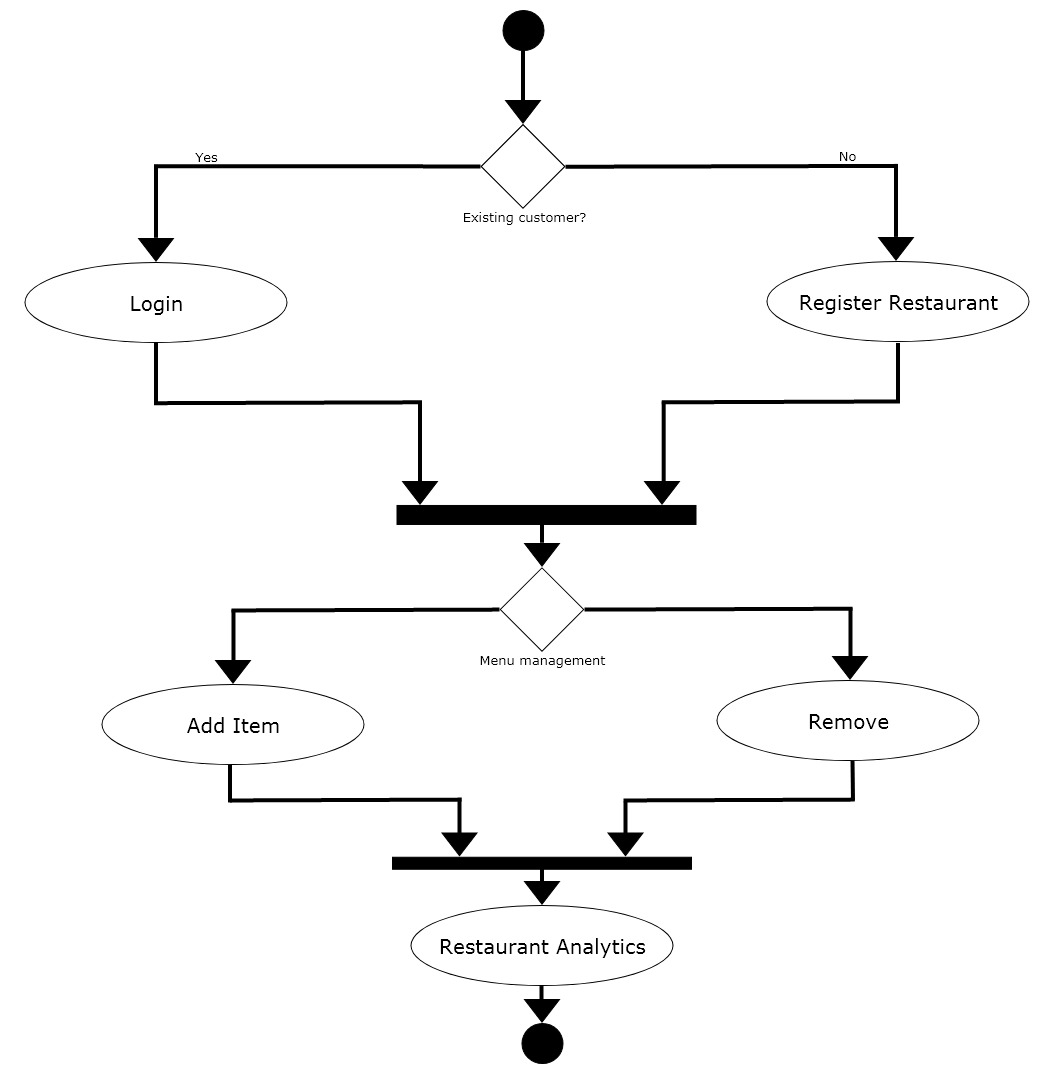
****

The customer/users will land on the landing page of the web application where top rated restaurants will be shown, then the user will have to login or create a profile in order to gain further access into the application and order food. Once it is done they will proceed and land on the main page where they get to search a specific restaurant or browse around. Then once a user has chosen a specific restaurant, they will be able to view that restaurant’s full menu item and be able to add to cart. Finally once the user has decided what to order they will be able to place an order or have it as a pre-planned order where they can decided a specific time and dates and the order will go through automatically given by the users input. Immediately after placing the order, the user will then go to the track order page where they will be able to track the orders full process such as the restaurant confirming the order, preparations, and courier pickups. The user and restaurant will then be able to communicate regarding the order through order communication if they so chooses. Lastly when the items are delivered to the given address, the user can confirm the order and leave a feedback directly to the restaurant and even be able to share the feed through their chosen social media platforms.

**5.4.2 Restaurant Registration and Menu Management**

The diagram shows the process of a restaurant login in if they are an existing customer and if not they will prompt to register the restaurant. Once a restaurant has registered and completed all the required information, they can procced through the application by signing in. They will land to the main page of the application, where the full menu and profile of the restaurant if visible. The restaurant can customize the menu in the menu management where they can remove or add items. Another option for the restaurant is the analytics board where it shows their top rated items, most ordered items and such.

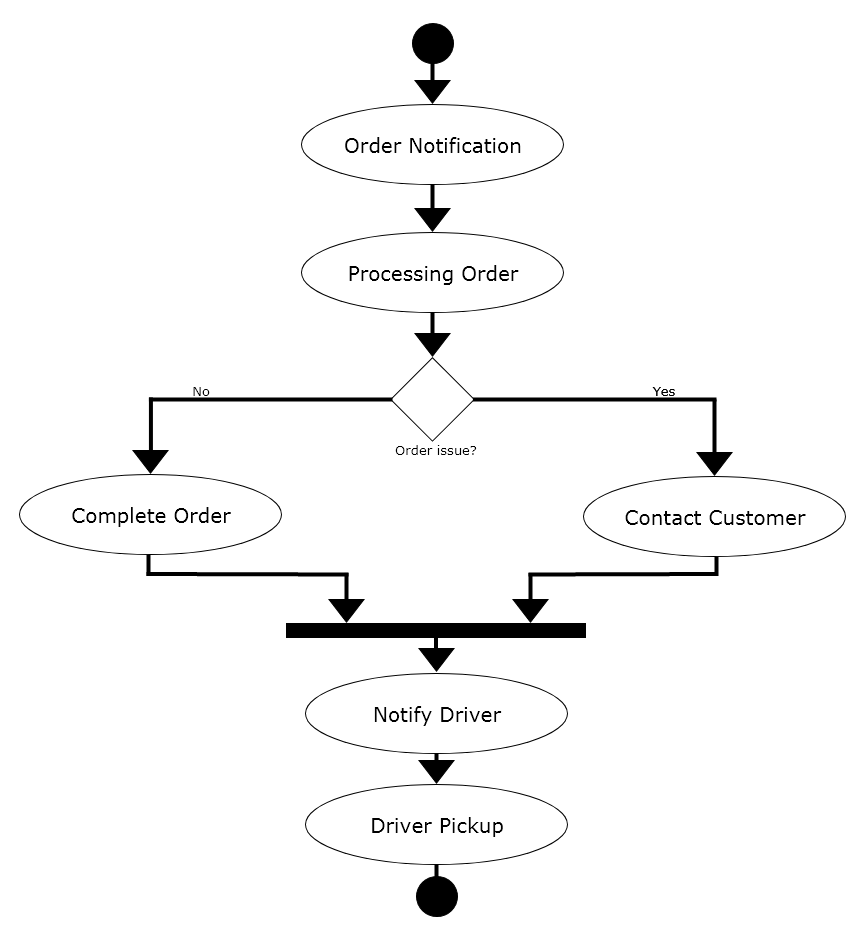
****



****

**5.4.2 Restaurant Order Process**

The diagram below shows the restaurants order process. The restaurant will get an order notification whenever an order is made by the customer. Restaurant can view the extent of the order and they will move on to the processing of the order. If there is any problem with the order they can decide to contact the customer or the customer can contact them if there are any concerns or question regarding the order process. Once they are done fulfilling the order of the customer, they can complete the order and will notify the couriers and then wait for them to arrive in the restaurant to pick up.



****

**5.5 Software Qualities**

**5.5.1 Correctness**

**5.5.2 Time-Efficiency**

**5.5.3 Robustness**

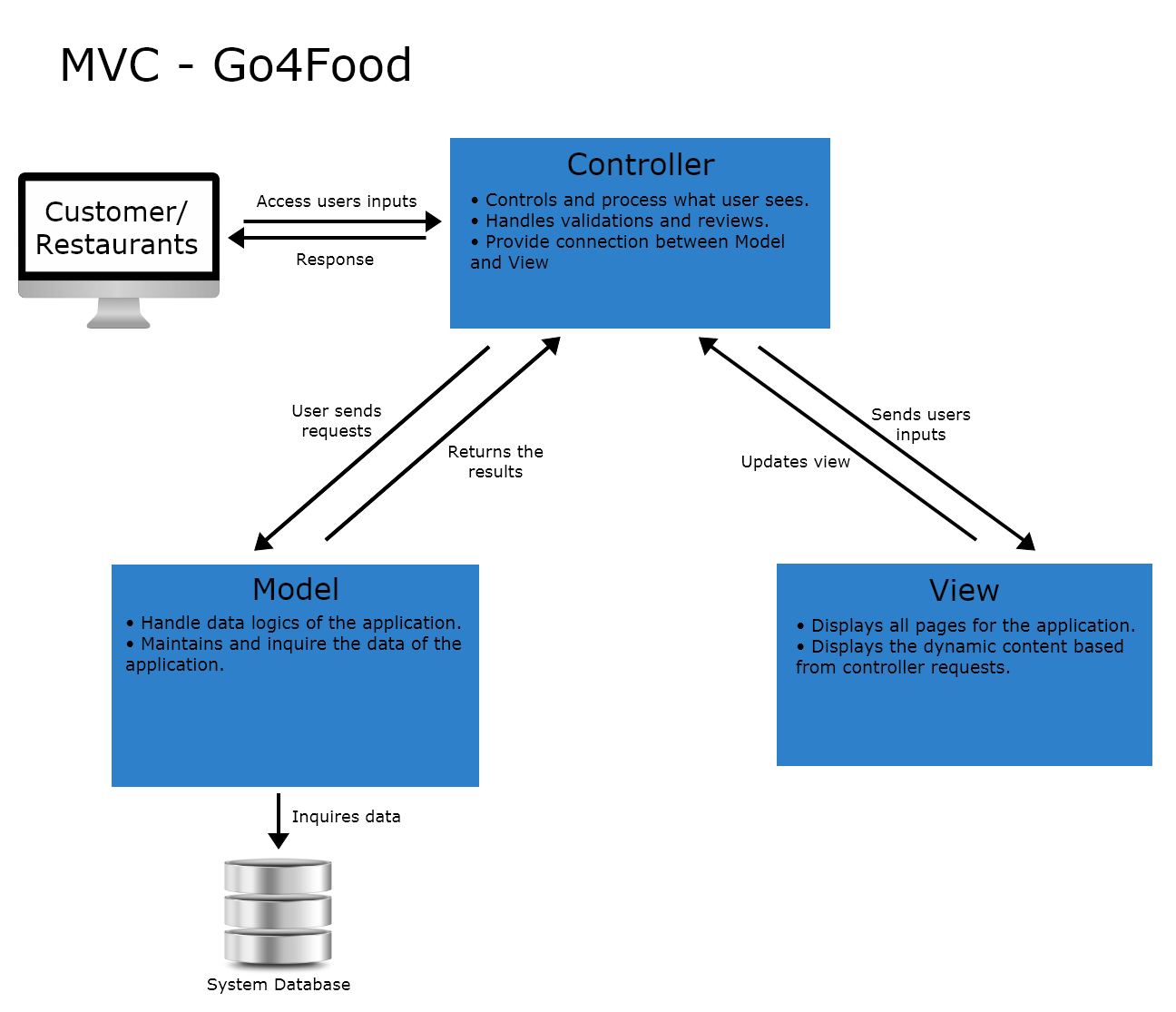
**6. Non-Functional Requirements**

\*\*TO DO\*\*

****

**7. Top-level and Low-level Software Design**

**7.1 MVC Architecture**



**7.1.1 Benefits**

Faster Development Process - since in the MVC architectures the codes are separated into three sections (View, Controller, and Data), it is possible for the work to be divided among group members. Thus, expediting the whole development process of the application. While maintaining efficiency as group members work on preferred components of the development.

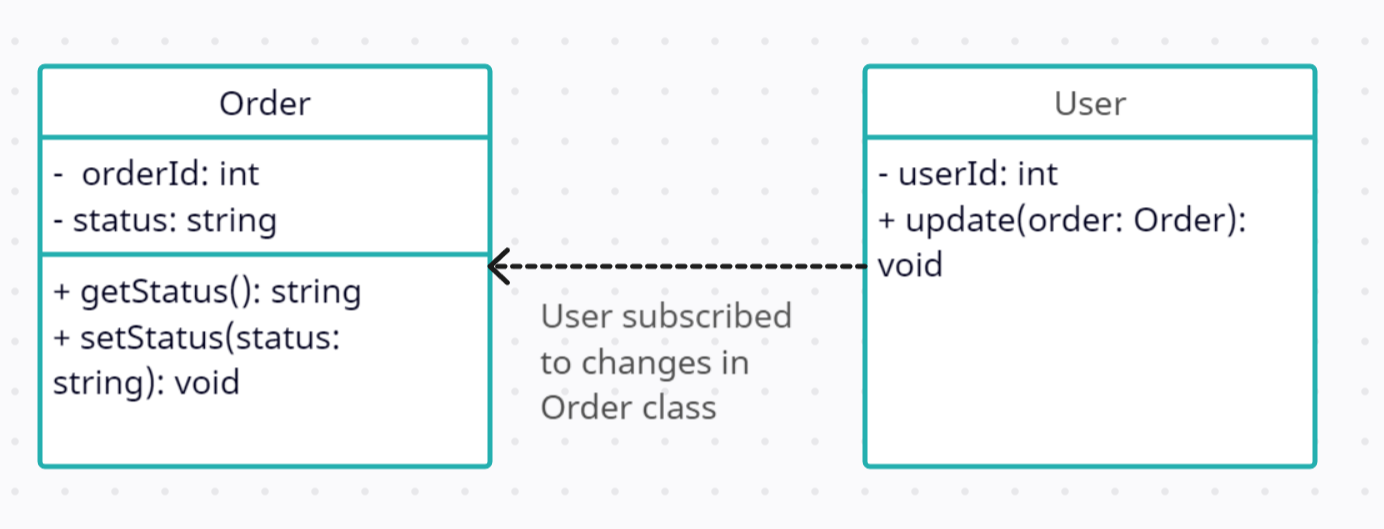
Modifiable - the MVC architecture allows straightforward modifications as such when adding new or modifying functionality it does not affect the overall architecture of the application. Locating certain codes to be improved or revised can be quickly done so due to having sections. By doing so, the applications overall scalability and flexibility is much more improved.

Decoupling of Codes - using the MVC architectures ensures decoupling of codes through the Model, View and Controller separation method. The View acts as the user interface of the application. Secondly, The Controller handles all user interactions. And, the Model provides the data part of applications. This separation allows code to be maintained efficiently and understood.

**7.2 Design Patterns - Observer and Factory**

**Observer Design Pattern**

The observer design pattern is used in Go4Food concerning new changes to the order status. Parties that are affected by this should therefore be notified of such changes. For example, as each order status changes, the observers which would be the users in our case would be notified of the order status change. Below is a class diagram of this implementation of the observer design pattern within this application.



The order class represents a single order. It has properties such as order id and order status. The user class represents a single user, it has a property for user id. Note that in the final product, the user may have more properties in the database such as username. However, for the purposes of illustrating the design pattern, only relevant properties have been included.

There’s a one-to-many relationship between the user class and the order class. This means that an order can make multiple orders.

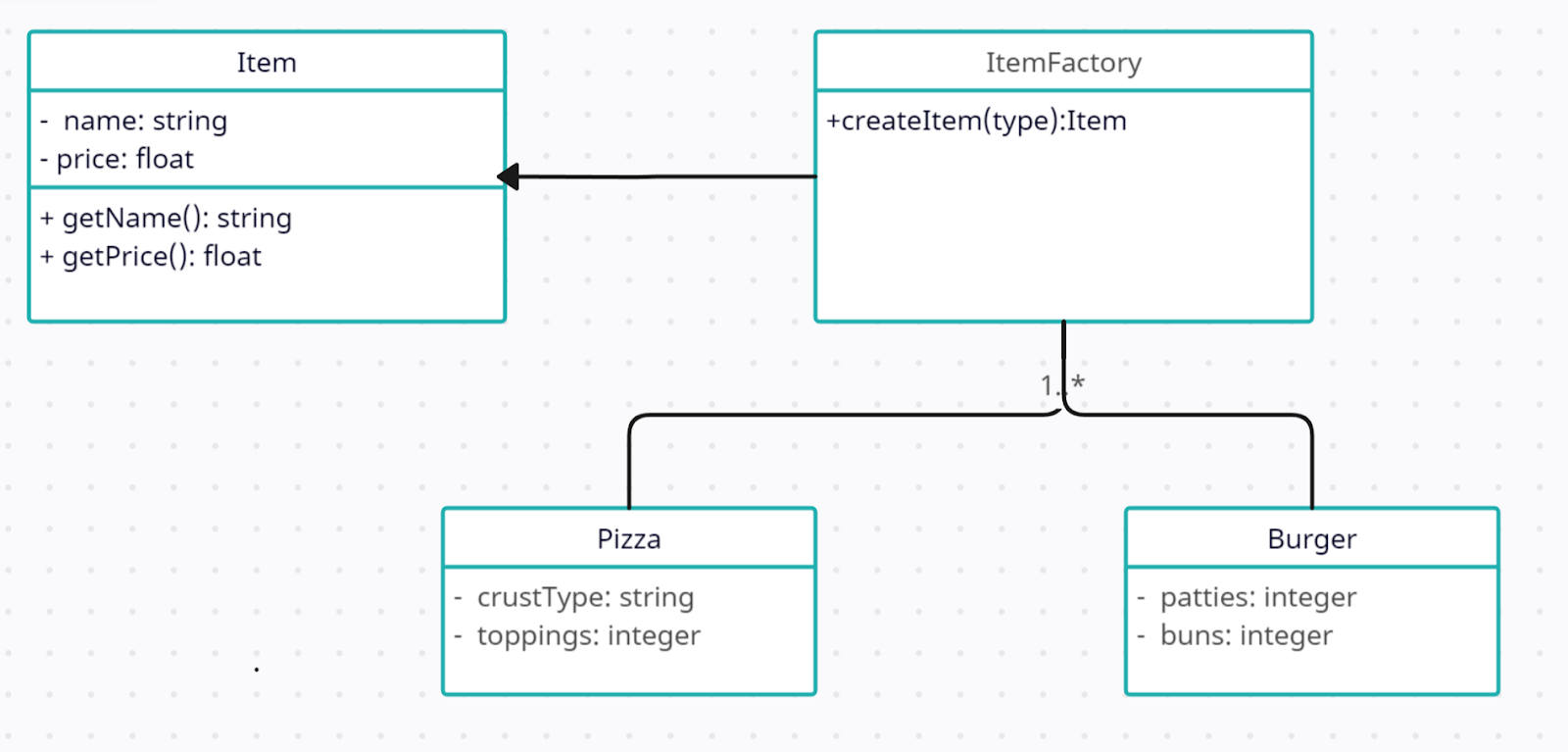
The user also has an update(order) method. This method is basically called by the order class when an order status changes.

The algorithm follows that in the order class, the getStatus() method returns the status of an order. The setStatus() method is used to update the status of a given order. It will also notify the observers.

In the user class, the update(order) receives the updated order and then takes the appropriate actions.

**Factory Design Pattern**

The factory design pattern is used in Go4Food to provide a design functionality where there exist generic food items, from which there are concrete examples of a particular item. The following class diagram shows how this is achieved. For example, considering a restaurant that has food items such as pizzas and burgers, a factory design pattern can be used to create different object types of burgers and pizzas based on their unique properties, etc. For each of these, we may have different sets of toppings, crust types, etc. The factory class is responsible for creating these specific pizza instances based on user selections. Consider the class diagram below:



The item class is a generic class for an item that has properties for its name (string) and price (float). It also has two methods getName and getPrice which both return strings and floats respectively. The item factory class is responsible for creating particular item objects. It has a createItem(type) method which takes type as the type of item. Based on this type, it returns a concrete type of restaurant. Pizza and Burger are concrete examples of how the item class extends.

The algorithm based on the class diagram above follows that if type is pizza, an instance of the item class will be created (demonstrated by concrete class pizza). It sets this particular instance’s attributes, such as name and price, and then returns its object. The same logic would apply if the type were to be something different, for example burger. Notice that the same factory design pattern could be implemented for the type of pizza itself as explained above.

The key feature here is that each pizza class or burger class inherits key properties from the generic item class, such as name and price, but with the factory design it shows how it is able to introduce new properties tailored to its own implementation.

**7.3 Comprehensive Class Diagram**

\*TO DO\*

****

**7. Project Members**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **SID** | **Email** | **GITID** |
| Adnan Uddin Mohammed | 200463605 | ami949@uregina.ca | adnan1420 |
| Justine Jhon Papeleras | 200405197 | justinejpapeleras@gmail.com | jmp808 |
| Mohammad Tamanna | 200376962 | mohammad.tamanna@outlook.com | tamannamo |

**7.1 Roles and Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Name** | **Roles** | **Responsibilities** |
| Adnan Uddin Mohammed | • Full Stack  • Requirements Documentation | • Modified the server and front end to fit with the backend  • Created the backend for both Customer and Restaurant users  • Presentation PPT Design patterns |
| Justine Jhon Papeleras | • Project Manager  • Full Stack  • Requirements Documentation | • Implemented the server  • Static pages for index, sign in and registration pages  • Creation of diagrams, charts, and tables  • GitHub organizations  • UI designs and flow  • Presentation PPT and Project Report  • In charge of tasks managements and allocations |
| Mohammad Tamanna | • Project Manager  • Front End  • Requirements Documentation | • In charge of tasks managements and allocations  • Created static pages for the customer  • Presentation PPT and Project Report |

****

**7.2 Project Reporting**

Project reporting structure was completed through the GitHub Projects section, arranged and organize resembling a functioning Kanban Board. The Project section is a form of a task-board spreadsheet and road map. It aided through visualization in showcasing and coordinating the group's strategic planning, task assignment and completion, and the project's overall efficiency. Group members meets twice a week, after the class lectures and every Fridays afternoons. The after class meeting is a quick touch based meeting where tasks were assigned, status reporting, and general questions/concerns. Friday meetings were a group work meeting where team members collaborate and work together on their tasks. This projects reporting structure approach falls under the Agile Scrum Methodology.

**9. Development Environment**

**9.1 Software Tools**

The software tools that were used to complete the project are VS Code, GitHub, Express.js,

**9.2 Programming languages**

Project’s application programming languages were HTML, CSS, JavaScript, JavaScript, and EJS.

**9.3 Project folders**

The folders in the project roots in the GitHub are the Documents, config, controllers, middleware, models, public, utils, and views. The Documents folder contains the key project documentations. The config folders contains the database JavaScript. The controllers, models, and views are folders that follows and contains the respective MVC architectural patterns. Middleware folders contains user authorizations for both the Customer and Restaurants users. Public folders has the f

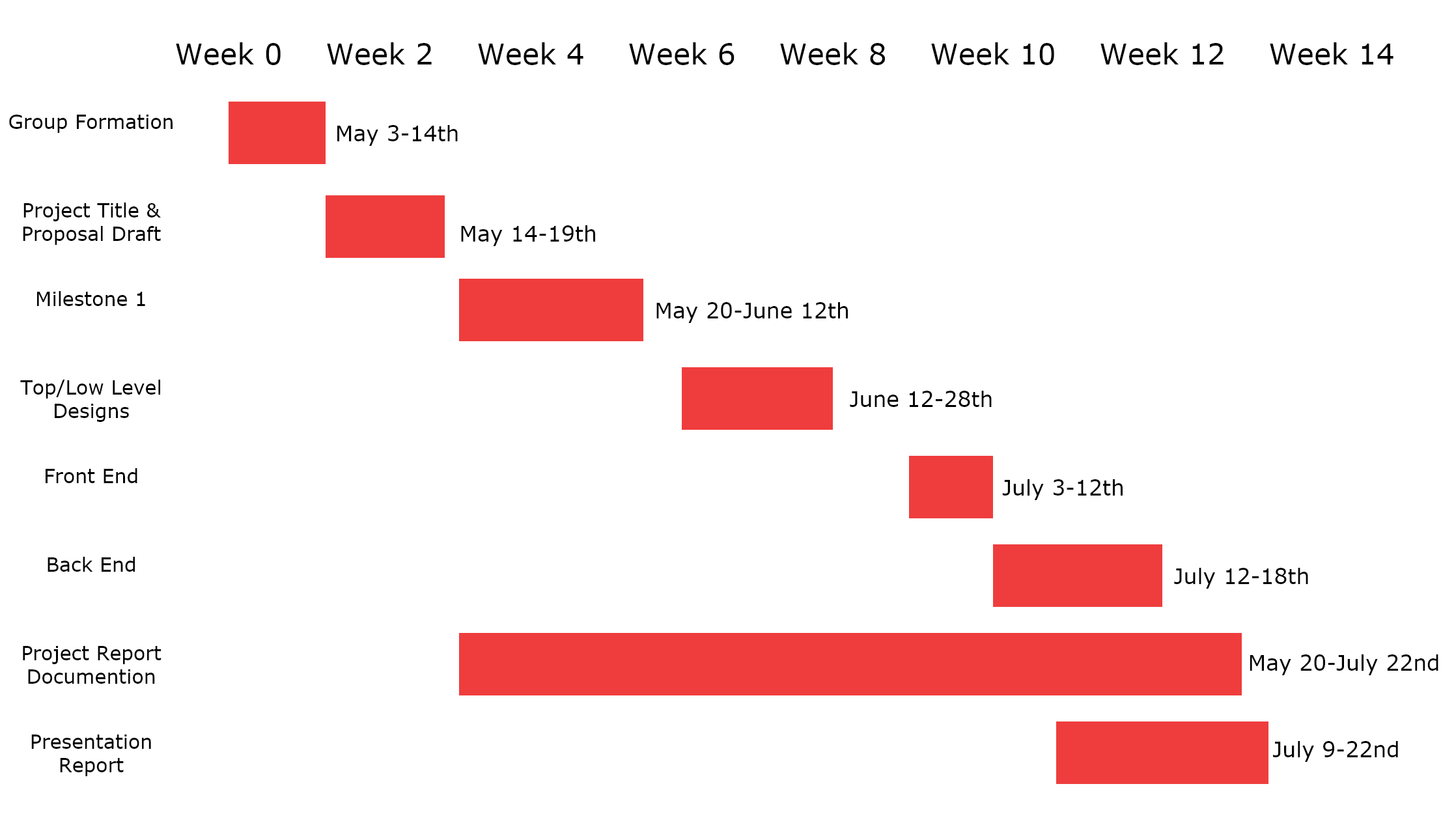
****

**10. Project Timeline**

**10.1 Project Milestones**

|  |  |
| --- | --- |
| **Milestones** | **Dates** |
| Group Formation | 14th May 2023 |
| Group’s Project Proposal | 15th May 2023 |
| Project Title and Proposal Draft | 19th May 2023 |
| Milestone 1 | 12th June 2023 |
| Hi-Fi UI and UX Design Completion | 28th June 2023 |
| Top-Level and Low-Level Software Design | 28th June 2023 |
| Front End Completion | 12th July 2023 |
| Back End Completion | 18th July 2023 |
| Project Report Completion | 22nd July 2023 |
| Presentation Report Completion | 22nd July 2023 |
| Project Deadline | 21st July 2023 |
| Presentation Opens | 26th July 2023 |
| Presentation | TBA |

**10.2 Gantt Chart**



**11. Expected Outcome and Products**

**11.1 Deliverables**

**11.2 Risks and Constraints**

Due to the Spring/Summer semester the time constraints of roughly three months for the projects formation and proposal, to designing phase, coding of the application, implementing and testing, and lastly the final Project Reports and presentation were significant. The three person group instead of four member means that extra tasks for each team member. In addition the Scope constraint,

The risks of the development project are group members not being able to complete given tasks with due dates leads to delaying other timed planned tasks. Project scope being to extensive and labour consuming given the limited time constraint. Furthermore, members lacks of

**12. High Level Diagram**