Project:

1. Project Team Name and #. List of team members. Vision. Project description.

Team Members: Abdulrahman Alaraj, Chirag Kamat, Ganesh Byrandurga Gopinath.

Team Number: 40

Title: FitFoodie

Description: A web based system which recommends restaurants and allows the user to order food based on nutrition preferences\requirements.

Vision: Building a web based system by leveraging object oriented frameworks and applying object oriented design patterns.

2. List the features that were implemented (table with ID and title).

Features that were implemented

Use cases				
ID	**Description**	**Topic Area**	**Actor**	**Priority**
US-01	As an Admin, I need to be able to add a Restaurant.	User Account	Admin	High
US-02	As an Admin, I need to be able to add meals to a Restaurant.	User Account	Admin	High
US-03	As an Admin, I need to be able to delete a Restaurant.	User Account	Admin	Medium
US-04	As an Admin, I need to be able to delete meal from a Restaurant.	User Account	Admin	Medium
US-06	As an Admin, I need to be able to view Restaurant details in the database to have a collective view.	Documentation	Admin	High
US-07	As a Customer, I need to be able to order food based on nutritional preferences.	User Account	Customer, Guest	High
US-08	As a Customer, I need to be able to order food based my profile.	User Account	Customer	High
US-09	As a Customer, I need to be able to create my own profile.	User Account	Customer	Medium

US-10	As a Customer, I need to be able to remove my own profile.	User Account	Customer	Medium
US-11	As a Customer, I need to be able to update my profile preferences.	User Account	Customer	Medium

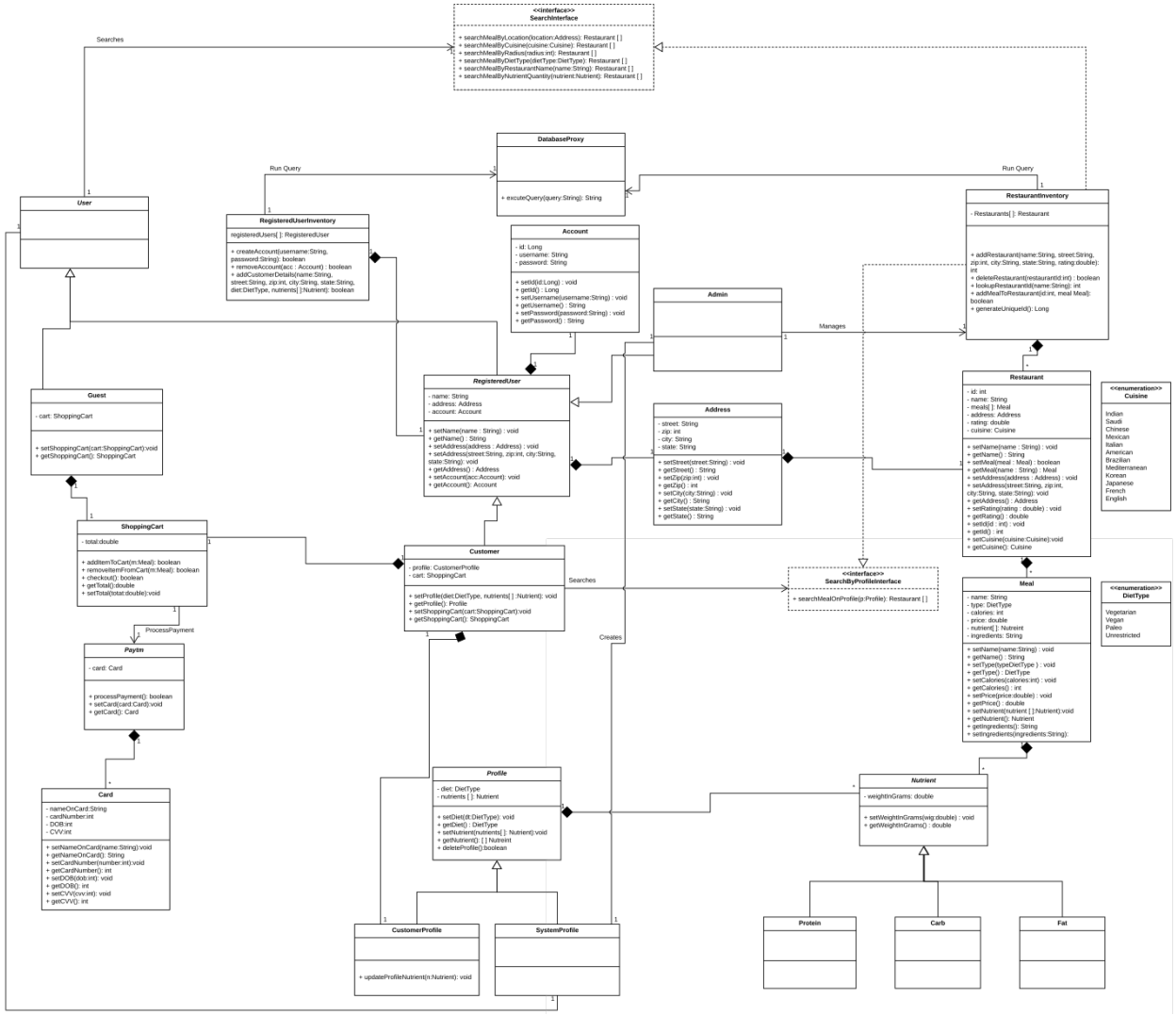
3. List the features were not implemented from Part 2 (table with ID and title).

Features that were not implemented

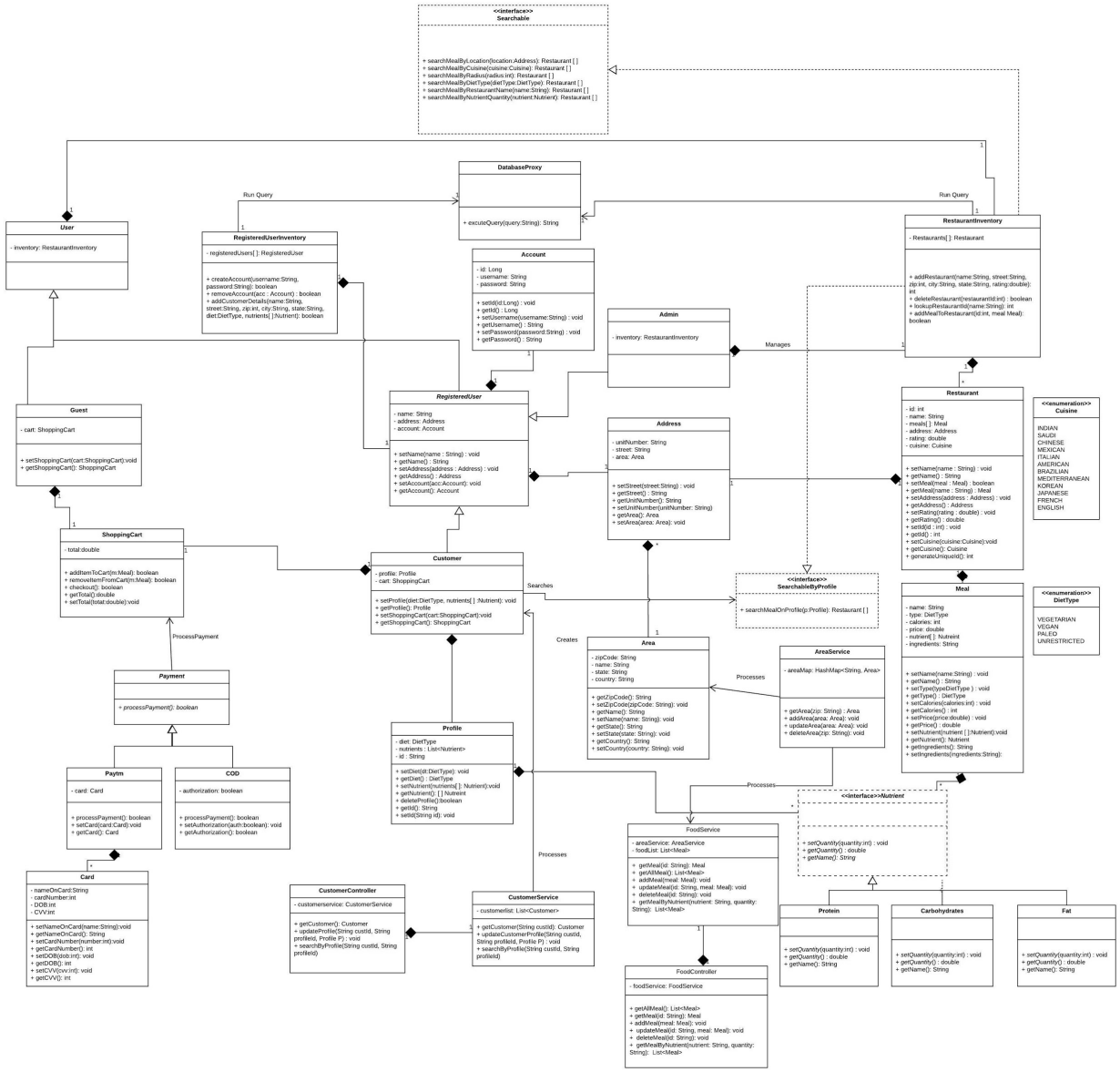
Use cases				
US-06	As an Admin, I need to be able to view Restaurant details in the database to have a collective view.	Documentation	Admin	High
US-12	As a Customer, I need to be able to pay for the order.	Payment	Customer, Guest	High

4. Show your Part 2 class diagram and your final class diagram.
What changed? Why? If it did not change much, then discuss how doing the design up front helped in the development.

Class diagram - part 2



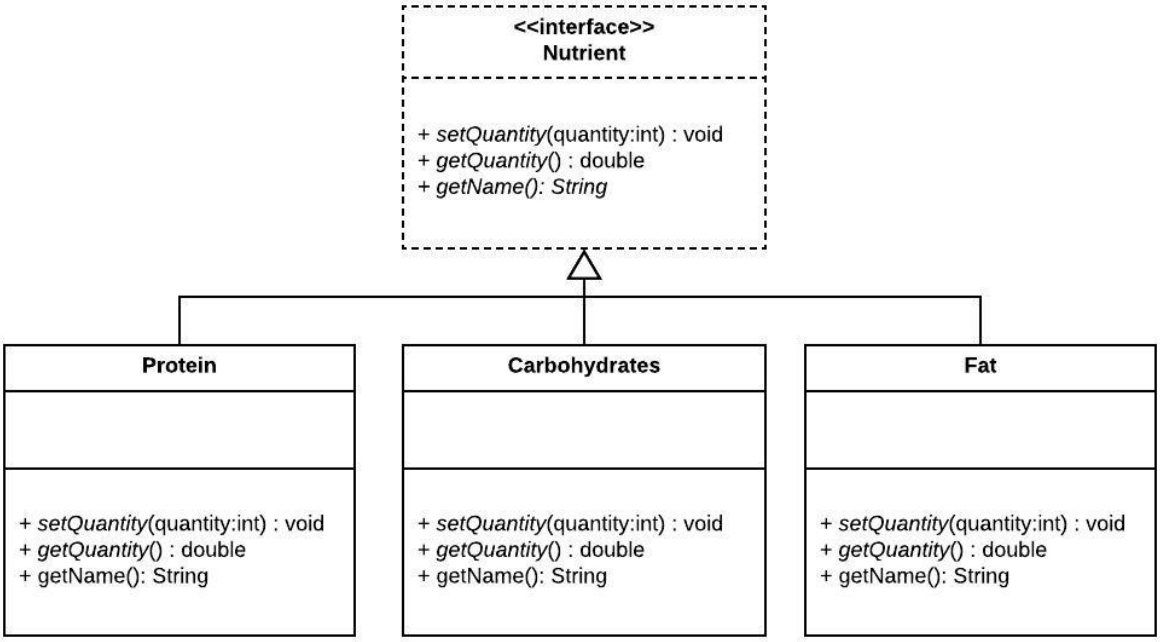
Class diagram - Final



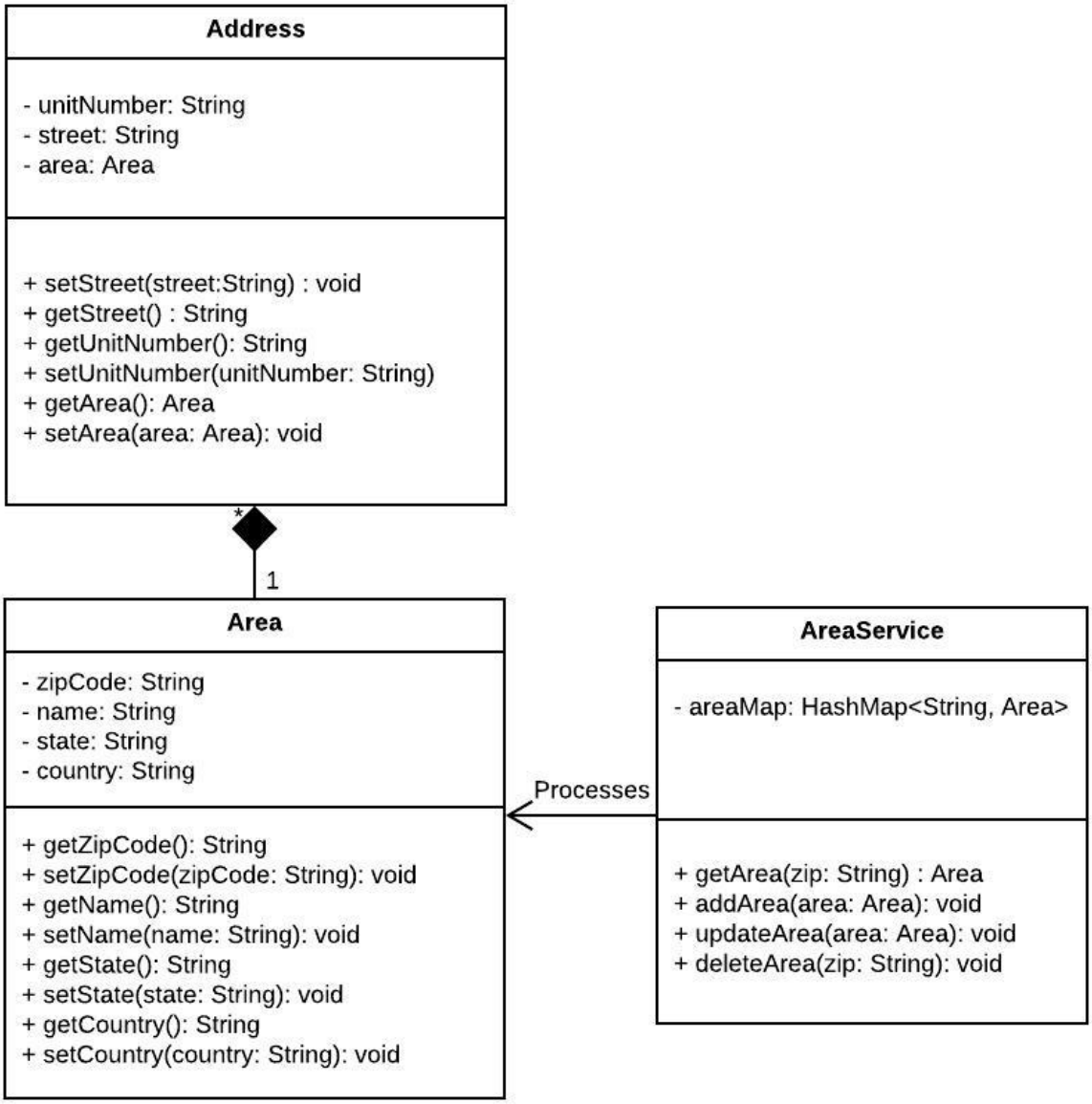
In part 2, when we began working on the class diagram, we had little knowledge about design patterns and web framework. For example, before we learn flyweight design pattern, we planned to have an Address object associated with each Meal object. This, in turn, necessitates that multiple Meal objects that belong to a single restaurant contain multiple instances of the same Address object. After going through flyweight design pattern, we changes the logic of the previous design to have a list of Area objects and assign each Meal object a specific Area object from the list, hence, the flyweight design pattern. Also, during the design of the class diagram, we did not consider web frameworks. For instance, before we learn about Java Spring Boot framework, we did not have a clear idea of how the customers can interact with the system. It might be clear in the class diagram from part 2 that we did not take customers interaction with the system into consideration. After getting acquainted with web frameworks, we adopted Java Spring Boot framework, and during its implementation, we realized that certain classes need to

be introduced in order to satisfy the framework conditions. Lastly, it might be important to note that during part 2, we were trying, probably excessively, to enforce encapsulation. This led to a relatively extensive class diagram which, during the implementation phase, became exhaustive to implement.

5. Show the classes from your class diagram that implement each design pattern
Class diagram - strategy design pattern



Class diagram - flyweight design pattern



6. What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?

We have done many projects in our coursework starting from our undergrad and working few years in the industry. Although, we have seen and implemented clean and robust code, this course helped us understand the importance of clean, robust and reusability of code. So now we have understood the importance, we also have implemented different design patterns. The way we see a problem and our approach has completely changed. We have also implemented certain design patterns in our other course projects this semester and in hackathons. A notable mention is that, we have started to think in a way such that, any code that we write should be extendable in the future rather than modifying the existing code.