Take A Byte

Group Members:

u18234039 – Jiahoa Hu

u21439321- Sonaly Ambelal

u21508322- Zandile Modise

u21516741- James Fitzsimons

u21525936- Victor Zhou

u21556416- Asakundwi Siphuma

u21785742- Itumeleng Moshokoa

GitHub Repository: https://github.com/ZandileModise/PA5-214-Repo

Task 4: Report

Research

How do Restaurants work?

Restaurants have many components to them. “These components involve everything from purchasing and preparing food to serving meals and cleaning up after customers.” [1] (Indeed.com, 2022). Restaurants usually have certain operations that keep the business afloat. “These processes include specific responsibilities, though they may sometimes overlap. For example, service represents one component of restaurant operations, and one of its tasks is to serve food to customers. Preparation represents another component, and it involves the creation of the meals served. The manager or owner of a restaurant oversees these various components to ensure successful operations.

There can be multiple positions within a restaurant, where there is a need, a position will provide a solution. When hiring people for certain positions, managers often look for “soft skills” which includes “Communication skills, teamwork, organizational skills, ability to work in fast-paced environments, and attention to detail” (Indeed.com, 2022)[2.] There are two main categories within a restaurant. There is the “front of house (FOH) and back of house (BOH)” (Indeed.com, 2022)[2.] The article, *How to Succeed In A Food Service Career* (Indeed.com, 2022)[2], lists eleven different restaurant jobs as follows: Dishwasher, Host, Pastry cook, Server, Bartender, Chef, Sous chef, Sommelier, Food and beverage manager, General manager, and Shift manager. Although some of these roles can seem mundane or rather simple, they all contribute to the success and growth of the restaurant. There are lots of benefits to working in a restaurant. In each position you learn new skills and develop new talents. Some benefits may include meeting new people and building relationships, teamwork, free discounts of food and beverages, learning about new cultures and culinary techniques, improve customer service skills, build on your resume, and gain more industry knowledge, and lastly learn how to work nontraditional hours. (Indeed.com, 2022)[2].

The most applicable roles in a restaurant that applies to our system are as follows: Serving Staff, Chefs, and Bartenders. Chefs are very important. Without them

there would be no food in a restaurant. Chefs' salaries depend on a few factors namely experience, location, training (such as culinary school) and the benefits package offered. (The Entrepreneur, 2016)[3] . There are different types of chefs for example pastry chefs, sous chefs, and executive chefs just to name a few.

Serving staff also play a vital role in the restaurant. Customers interact the most with the serving staff therefore it is important that they make a strong lasting impression on customers. "Servers must be able to work well under pressure, meeting the demands of customers at several tables while maintaining a positive and pleasant demeanor." (The Entrepreneur, 2016)[3] .

Lastly Bartenders play a smaller role however still important. "The bartender begins their day by prepping the bar, which includes preparing the condiments and mixers for the entire day as well as ordering supplies. The bartender also needs to check the liquor requisition sheet and the liquor inventory and restock the bar." (The Entrepreneur, 2016)[3] .

These three roles makeup our small-scale simulation of a restaurant with the Serving staff working the floor and the Chefs working the kitchen. The Bartenders simply add to the joy of our restaurant.

References

Resources:
[1] Indeed.com. (2022). *A Day in the Life of a Restaurant Manager*. [online] Available at: https://www.indeed.com/career-advice/career-development/what-are-restaurant-operations [Accessed 7 Nov. 2023].

[2] Indeed.com. (2022). *How to Succeed In A Food Service Career*. [online] Available at: https://www.indeed.com/career-advice/finding-a-job/restaurant-work [Accessed 7 Nov. 2023].

[3] The (2016). *The Staff You Need to Hire to Run a Restaurant*. [online] Entrepreneur. Available at: https://www.entrepreneur.com/starting-a-business/the-staff-you-need-to-hire-to-run-a-restaurant/282438 [Accessed 7 Nov. 2023].

[4] unsplash.com (for all the images)

Assumptions and Decisions:

- Each waiter assigned will only serve their assigned tables.
- The queue of customers will be known before starting to seat customers.
- Seating is based on tables that can accommodate multiples of 2 (2, 4, 6, 8, 10 etc.…).
- For example, a party of 7 will be seated at a table that seats 8.
- Once the maximum capacity has been reached, no more parties can be seated that exceed the maximum capacity.
- Chefs have designated jobs and will only handle their relevant job.
- Orders follow the first come first serve preparation process.
- The individual orders for each customer at a table will be completed all at once and served all at once.
- The bill is presented once the table has finished eating.
- The party can pay as a group.
- The party can decide to split the bill, if so, the bill is split evenly amongst each customer of that party.
- Once a party has finished their table becomes open to seat a new party.
- To simplify the restaurant setup, we will have waiters and chefs.
- The system uses the classes to accomplish other tasks such as seating customers instead of a host staff.
- User Interface Design
    - Used a command line interface in our design as it was less complex and gave us a better platform to showcase our design patterns and how they were used.
- Modularity
    - The code was organized into separate modules to promote code reusability and maintainability. Modularity allows for easier updates and additions to the system in the future.
- Also made a game loop that showcases our design:
    - Restaurant is a singleton.
    - Print out all customers in a queue.
    - Iterate to the customers we want to put in the store (also shows the flyweight.
    - Print out what tables have been taken/occupied.
    - Print out our menu.
    - Ask customers if they want to order now or wait(state) if not go to the other table our waiter is assigned to. (Used prototype to clone waiters)

    - Waiter goes back and asks for their order (memento is used to store orders)
    - Order is taken to headchef who is a mediator between the waiters and chefs.
    - Print out the chain of responsibilities as the order travels from one chef to another.
    - The dish is taken to headChef who will add garnish to the dish(decorator)
    - Also add drinks if ordered.
    - Order will be delivered to table.
    - Print out the bill.
- During the we will be indicating the state of the party as we go along

Application of Design Patterns

Chain of Responsibility

The Chain of Responsibility design pattern is a suitable choice for managing food preparation in the kitchen, where different chefs are responsible for specific parts of the order. In this context, the pattern allows you to ensure that the correct item in the order is created by the right chef by creating a chain of handlers.

Participants:

Handler - Italian

ConcreteHandler - PizzaDoughChef, PastaDoughChef, PizzaSauceChef, PastaSauceChef, PizzaToppingChef, PastaToppingChef, CheeseChef, PantryChef, HeadChef

Prototype

The Prototype pattern is used to clone waiters in the restaurant, which means that you create new waiter objects by copying an existing waiter object, thereby ensuring that the new waiters have the same initial attributes and behaviours as the original waiter. Cloning is often more efficient than creating new objects from scratch and avoids redundant setup operations.

Participants:

Prototype, ConcretePrototype - Waiter

Memento

The memento pattern is implemented for the restaurant class to allow backtrack of the order and state. If there is a mistake in an order or if a customer's request changes, the restaurant can use the Memento pattern to revert the order to a previous state, apply the necessary corrections, and update the state again. It is important for

maintaining customer satisfaction as that forms an essential part of a restaurant's reputation.

Participants:

Originator - Restaurant

Caretaker - RestaurantFacade

Memento - RestaurantMomento

Singleton

The Singleton pattern ensures that there is only one instance of the restaurant. This is useful for maintaining global settings and data for the entire establishment. There is no need to add multiple identical instances of the restaurant. There is only one point of control in the management of the restaurant.

Participants:

Singleton - Restaurant, RestaurantFacade

Flyweight

The Flyweight pattern is used to optimize memory usage by sharing common resources. In the restaurant context, it can be used to manage tables that are reused by different customers, ensuring efficient use of space. In cases where specific tables are rarely used, the Flyweight pattern avoids unnecessary resource allocation by reusing shared attributes and creating unshared flyweights only when needed.

Participants:

FlyweightFactory - Restaurant

Flyweight - Restaurant

ConcreteFlyweight - Table

Factory

The Factory pattern is used to generate different types of food such as drinks, pasta, and dessert. This pattern provides a structured and flexible way to create objects of various food types while abstracting the instantiation process from the client code. The Factory pattern allows for the addition of new food items to the menu without extensive code changes.

Participants:

Product - Order

ConcreteProduct - SaladOrder, DrinkOrder, PastaOrder, DesertOrder, PizzaOrder

Creator - OrderFactory

Template

Template pattern defines a way to execute functions. This pattern is used when you want to define a common algorithm structure but allow specific steps to be customized by derived classes. In this scenario, we use it to print the Receipt for each table, it just needs to follow printHeader, printBody, printFooter.

Participants:

AbstractClass - OrderFactory,

ConcreteClass - SaladOrder, DrinkOrder, PastaOrder, DesertOrder, PizzaOrder

Composite

Composite pattern is used to manage different types of Orders. In this scenario the totalOrder class represents the component class; it manages different types of orders.

Participants:

Component -TotalOrder

Leaf - Order

Composite - Restaurant

Facade

Facade pattern provides a simplified interface to a library. In this case we use restaurantFacade class to invoke the interfaces in restaurant class. The restaurantFacade class acts as an intermediary that encapsulates and simplifies interactions with the underlying subsystems and interfaces of the restaurant class. This ensures that clients can easily access and utilize restaurant services, without being exposed to the complexities of the underlying subsystem.

Participants:

Facade - RestaurantFacade

Subsystem - Restaurant

Iterator

The Iterator method lets you traverse elements of a collection without exposing its underlying representation. It is used to iterate the table to check the availability of tables in the restaurant, without revealing how the tables are organized.

Participants:

Aggregate - TableIterator

Iterator - Iterator

Command

The command pattern encapsulates a request as an object, thereby letting us parameterize other objects with different requests, queue or log requests, and support undoable operations, in this scenario, we created Restaurant Simulate command to simulate the restaurant activity.

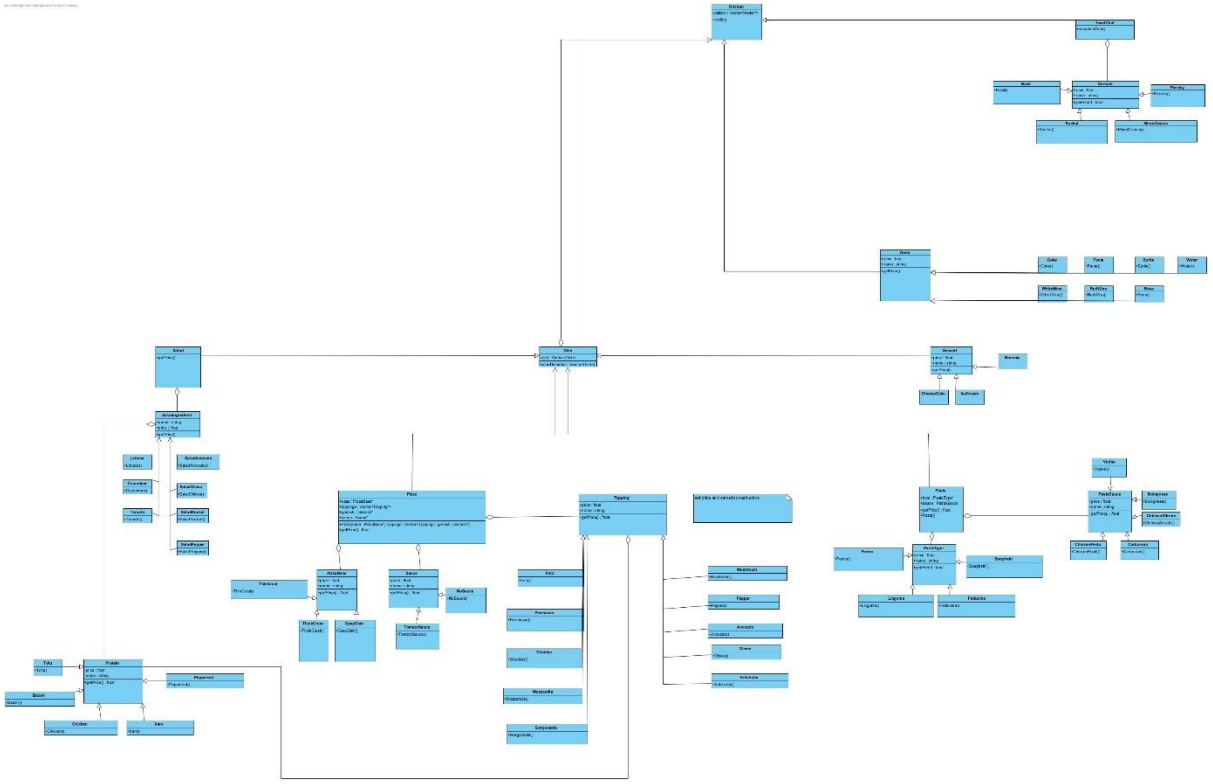
Participants:

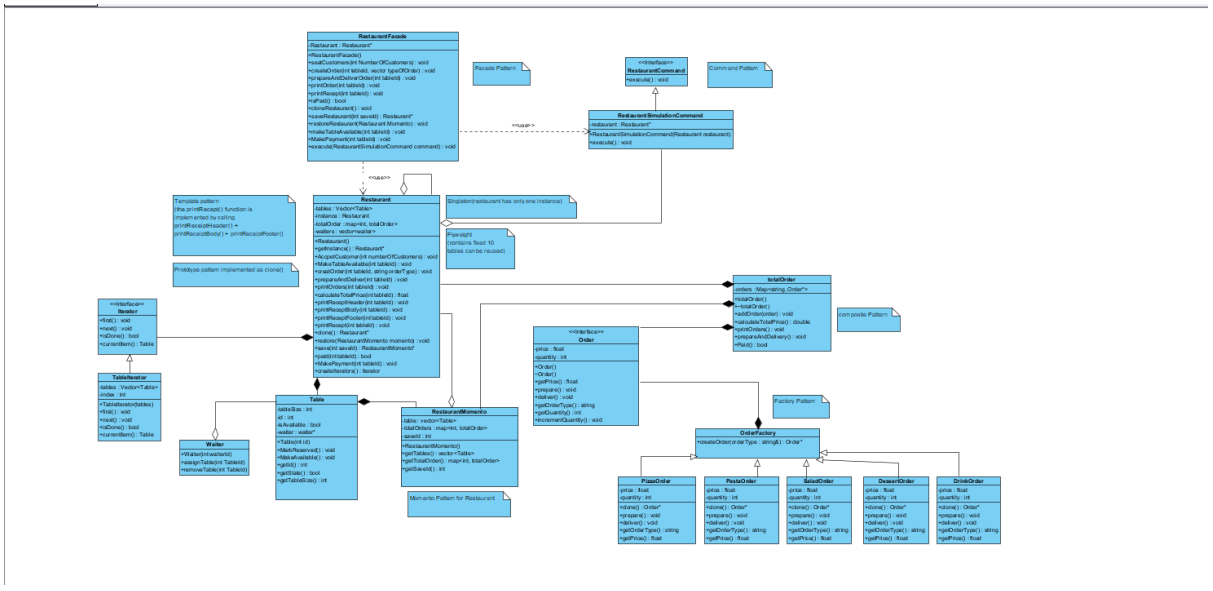
Invoker - Restaurant

Command - RestaurantCommand

ConcreteCommand - RestaurantSimulationCommand, Receiver - main

Final UML




RestaurantFacade
Facade Pattern
RestaurantCommand
Command Pattern
RestaurantSimulationCommand
Restaurant
Singleton(restaurant has only one instance)
Flyweight
totalOrder
Composite Pattern
Iterator
TableIterator
Table
RestaurantMomento
Memento Pattern for Restaurant
Order
Factory Pattern
OrderFactory
PizzaOrder
PastaOrder
SaladOrder
DessertOrder
DrinkOrder
Waiter

Our 214 game loop (suggestion)

0. Restaurant is a Singleton

1. Print out all customers in the queue

2. Iterate to the customers we wanna put in the store (maybe let's say there's only 3 tables left in the restaurant and this is a party of 6, therefore we have to put tables together (shows the flyweight)

3. Print out what tables they are seated at

4. Print out our menu (make it look good)

5. Ask customers if they want to order now or wait - if not go to the other table waiter is assigned (make an assumption that the other table the waiter goes to has started eating and they say their good for now) -> used prototype to make waiters since they have the same function

6. Waiter goes back and asks their order -> memento is used to store the order

7. Order is taken to the headChef who is a mediator between the waiters and the chefs

8. Print out the chain of responsibilities ask the order travels from one chef to another

9. The dish is taken to the headChef who will add garnish to the dish (decorator)

10. We'll also add drinks

11. Order will be delivered to the table (court they are eating)

12. Print out their bill

During all this we will also indicate the state of the party as we go along