TASK 4: REPORT

Matthew Els u21715191

Dominique da Silva u21629944

Tlhalefo Dikolomela u21507792

Tinashe Austin u21564176

Christopher Katranas u19154853

Contents

[INTRODUCTION 2](#_Toc149728178)

[4.1 RESEARCH 2](#_Toc149728179)

[4.2 DESIGN DECISIONS 4](#_Toc149728180)

[System Requirements 4](#_Toc149728181)

[Customer Management. 4](#_Toc149728182)

[Order Management. 4](#_Toc149728183)

[Billing 4](#_Toc149728184)

[Table Management 4](#_Toc149728185)

[Satisfaction 4](#_Toc149728186)

[Order 4](#_Toc149728187)

[4.3 WRITE UP OF DESIGN PATTERNS 5](#_Toc149728188)

[4.3.1 BUILDER 5](#_Toc149728189)

[4.3.2 CHAIN OF RESPONSIBILITY 5](#_Toc149728190)

[4.3.3 STATE 5](#_Toc149728191)

[4.3.4 STRATEGY 5](#_Toc149728192)

[4.3.5 DECORATOR 5](#_Toc149728193)

[4.3.6 COMMAND 5](#_Toc149728194)

[4.3.7 TEMPLATE METHOD 5](#_Toc149728195)

[4.3.8 5](#_Toc149728196)

[4.3.9 5](#_Toc149728197)

[4.3.10 6](#_Toc149728198)

[4.4 ASSUMPTIONS 6](#_Toc149728199)

[4.5 SUPPORTING UML DIAGRAMS 7](#_Toc149728200)

# INTRODUCTION

Our (Concrete pass) implementation of the given assignment focuses on the implementation of a restaurant simulation system. Customers are seated at a table, orders are taken by the waiter, and the food is prepared and served.

The primary components we have focused on, in our system, is the floor where customers interact with the waiters and are managed by the staff, as well as the kitchen where food orders are prepared and dispatched.  
Customer satisfaction also plays a role in this simulation, influencing factors such as tipping of their assigned waiter.

Customers also can create customized orders from a menu and may choose to start tabs for deferred payment.

# 4.1 RESEARCH

Similar project were found online and was used as reference as to what patterns to consider in our system design (as referenced below).

GitHub. (n.d.). *Grokking-OOD/object-oriented-design-case-studies/design-a-restaurant-management-system.md at master · wyaadarsh/Grokking-OOD*. [online] Available at: https://github.com/wyaadarsh/Grokking-OOD/blob/master/object-oriented-design-case-studies/design-a-restaurant-management-system.md#class-diagram [Accessed 1 Nov. 2023].

‌

Shahid, M. (2023). *Restaurant-Management-System*. [online] GitHub. Available at: https://github.com/mabbia706/Restaurant-Management-System/tree/master [Accessed 1 Nov. 2023].

‌

www.c-sharpcorner.com. (n.d.). *Food Delivery Application Using Design Patterns*. [online] Available at: https://www.c-sharpcorner.com/article/food-delivery-application-using-with-design-patterns/.

‌

Doyle, W. (2023). *Design Patterns Explained with Food 🥕*. [online] GitHub. Available at: https://github.com/wesdoyle/design-patterns-explained-with-food/tree/main [Accessed 1 Nov. 2023].

‌

Kothari, A. (2018). *Java Builder Design Pattern Example - Java Code Geeks*. [online] Examples Java Code Geeks. Available at: https://examples.javacodegeeks.com/java-development/core-java/java-builder-design-pattern-example/ [Accessed 1 Nov. 2023].

‌

# 4.2 DESIGN DECISIONS

## System Requirements

### Customer Management.

The system had to have a mechanism for the customers to be able to request, whether that is by reservation or by walking in, a seat. It should mimic the role of a Maitre D and assign customers to an available table. Waitstaff must be assigned to the table as to manage and serve the customers that are seated at the table.

### Order Management.

Waiters should be able to take a table’s order, then communicate to the kitchen. The waiter should also be the one to serve to the table with the meal. Waiters have the role of mediator between the kitchen and the floor.

### Billing

The implemented system, should at the end of the dining experience be able to generate a bill for the table. The table should be able to request for bill splitting, i.e. should be able to pay the bill in multiple payments. In addition to that, the table should be able to defer the payment, offering to pay the bill at a later date.

### Satisfaction

Customers should be able to rate their dining experience or complain. Should the experience be rated badly, there will be implications for the tipping of the waiter. Complaints should be handled by the manager.

### Order

Customer should be able to build their own order, selecting items from a set menu. Customers should also be able to specify any additions and preparation methods for their requested food. The order is taken by the waiter and delivered to the kitchen. The kitchen will notify the waiter when the food is done and thereafter the waiter will serve the table their order.

# 4.3 WRITE UP OF DESIGN PATTERNS

## 4.3.1 BUILDER

The Builder design pattern's basic intent is to separate the construction of a complex object from its representation, allowing the same construction process to create different representations.

A pizzeria has many different types of pizzas (representations), however the basic construction remains the same up until toppings are to be chosen for the final product.  
Thus we have chosen the builder design pattern to be able to create many different types of pizzas while simplifying the construction process.

## 4.3.2 CHAIN OF RESPONSIBILITY

The Chain of Responsibility design pattern is intended to create a chain of objects, where each object can process a request and decide whether to pass it to the next object in the chain or to stop processing it. It allows you to decouple the sender of a request from its receiver, providing multiple objects the opportunity to handle the request.  
With this in mind, the decision was made to pass the order made by the table through a chain. Multiple classes have to handle the request in different ways.  
In our design the order made (the request made by the customer table) is passed from the customer table, to the waiter. The waiter then passes it to the kitchen, and eventually the order is passed to the head chef to plate, and then sent back to the waiter to serve the table with their meal.

## 4.3.3 STATE

## 4.3.4 STRATEGY

## 4.3.5 DECORATOR

## 4.3.6 COMMAND

## 4.3.7 TEMPLATE METHOD

## 4.3.8

## 4.3.9

## 4.3.10

# 4.4 ASSUMPTIONS

# 4.5 SUPPORTING UML DIAGRAMS

(The idea is to include all the design patterns under this heading and refer to the figures in 4.3)

(We can just instant reverse our code and use a snipping tool to get each pattern’s UML class diagram)

Possibly also include a link to the entire system design UML Class diagram.