**DOKUZ EYLÜL UNIVERSITY**

**ENGINEERING FACULTY**

**DEPARTMENT OF COMPUTER ENGINEERING**

**CME 2210**

**Object Oriented Analysis and Design**

**LARGE RESTAURANTS OR CAFETERIAS**

**by**

**Yiğit Can - 2021510021**

**Eylül Pınar Yetişkin - 2021510069**

**Evrim Gizem İşci - 2020510091**

**CHAPTER ONE**

**INTRODUCTION**

This project will be helpful for restaurant and café owners. We are aiming to make a program that will organize their place's menu, employees, budget, payment system, seating plan, etc. We hope that all types of restaurants- from boutique cafes to large chain restaurants- will be able to use this program.

**Access**

With this program employees can add new food to a menu or remove food from the menu, the boss can see employee's information and place's income and outcome prices, the cashier sees how much money will tables pay and see which table is empty, chiefs can see the ordering queue, etc.

On the main screen of the program, the parts that the boss and employees can see will be in separate sections. So if the boss wants to see the information about the place where he works, he will be directed to the boss screen. In this way, the boss will be able to directly access the income of the cafe, information about its employees, the payment system, and the seating arrangement. Employees will be directed to the screen they will use only in their own department.

In the employees category, the chef will be able to access the places that the chef can access, including the menu, and the cashier will be able to access the menu as a subset of the parts that the boss can access, such as the prices, seating plan, payment system. In this way, a simple use will be presented for the use of employees and the boss.

**Food**

In the food menu section, basic categories such as dishes, drinks, and desserts will be displayed. By adding or removing these categories and the contents of the categories, the program user can easily make the menu suitable for their own cafe.

**Screen**

Since we will display the program screen as minimally and categorized as possible, the owner and staff of the cafe can easily adapt to this program.

**Reading Input Menu**

The program will get the menu and prices of the items in the menu as a file which will be used as input. The owners of the restaurant will just enter the menu into the program and it will be ready to use in a few minutes.

**CHAPTER TWO**

**REQUIREMENTS**

**Main Project**

We will use chief, table, food and drink, menu, boss, cashier, employee, and budget class in this project. We will use our interface and abstract class information while creating these classes. We will use inheritance principle within our chief, boss, and cashier classes. In this way, some of the things that the boss can access and change will be easily accessible to the chef and cashier.

**Classes**

Attributes such as age, name, gender, and salary will be kept in the employees class. Values such as meals, drinks, and desserts will be kept in the food and drink class. In the table class, which table has spent how much and the numbers of the tables will be kept. In the budget class, the monthly income and expenses of the cafe and the profit from these incomes and expenses will be calculated and kept.

**Functions**

In our classes, we will use showPrice() for showing food's price, showCost() for showing food's cost, getSalary() for getting employees' salary, getTotalCost(Table table\_number) for getting the total cost of a table, setNewFoodandDrink(FoodandDrink food\_name) for adding new food to the menu, etc. ReadMenu(String menu) will contain all the input menu reading part of the program.

**Structures**

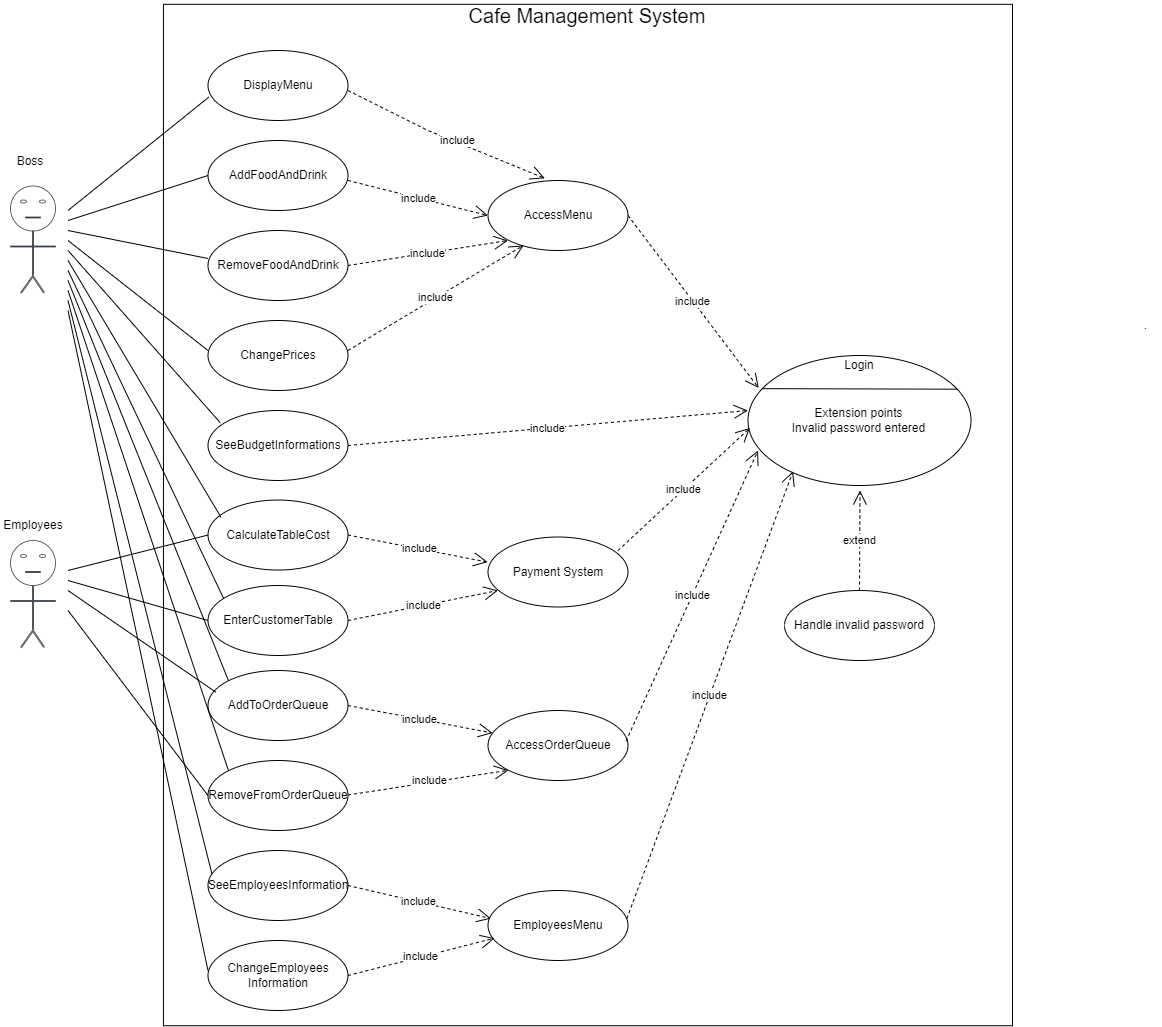
In our project we will use queue structure for food waiting queue, list structure for our food menu and employees. List structure will also be used for storing the orders of each table.

**CHAPTER THREE**

**UML DIAGRAMS**

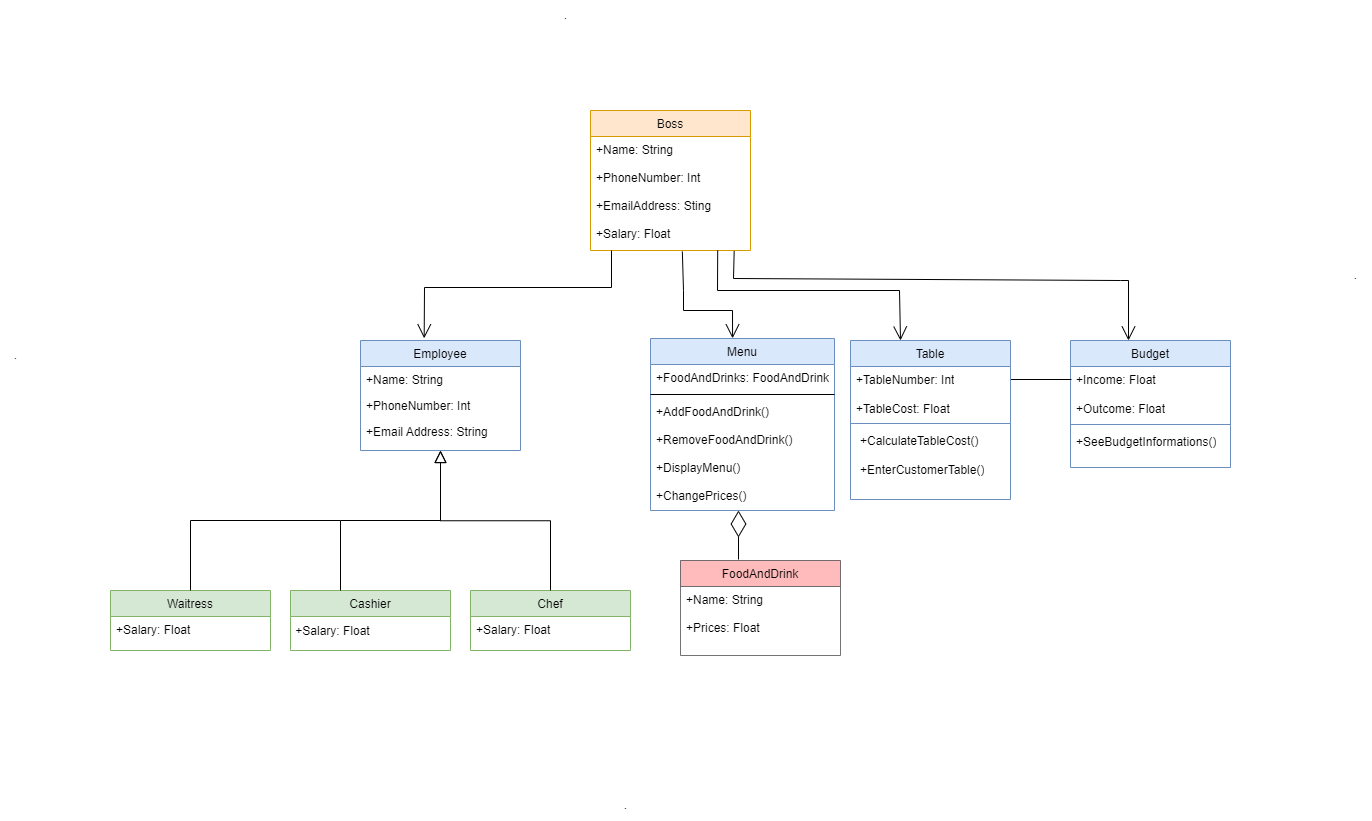
Use Case Diagram:

The boss has 11 functions to use in the restaurant system but only 4 of them are usable by employees. To use a function the client must log in to define their identity. After that, the system has 5 options that some lead to sub-options, and the client can choose from them.



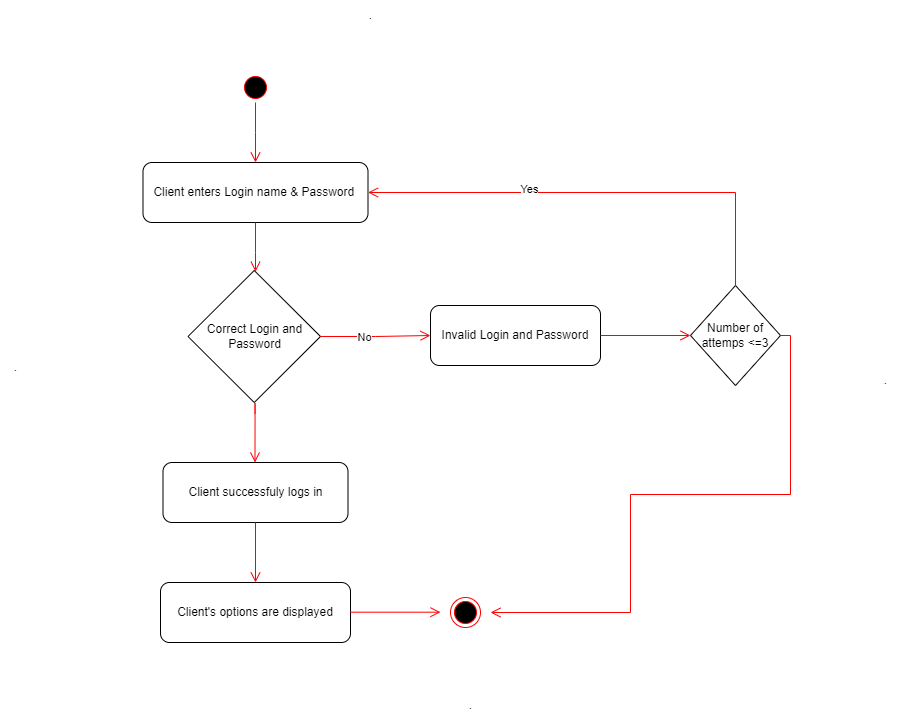
Class Diagram:

Boss class can access all other classes. Employee class has three sub-classes that inherit from it. Menu class has 4 functions and it uses FoodAndDrink Object as its attribute. Table class has 2 methods and it works together with Budget class which has 1 method.

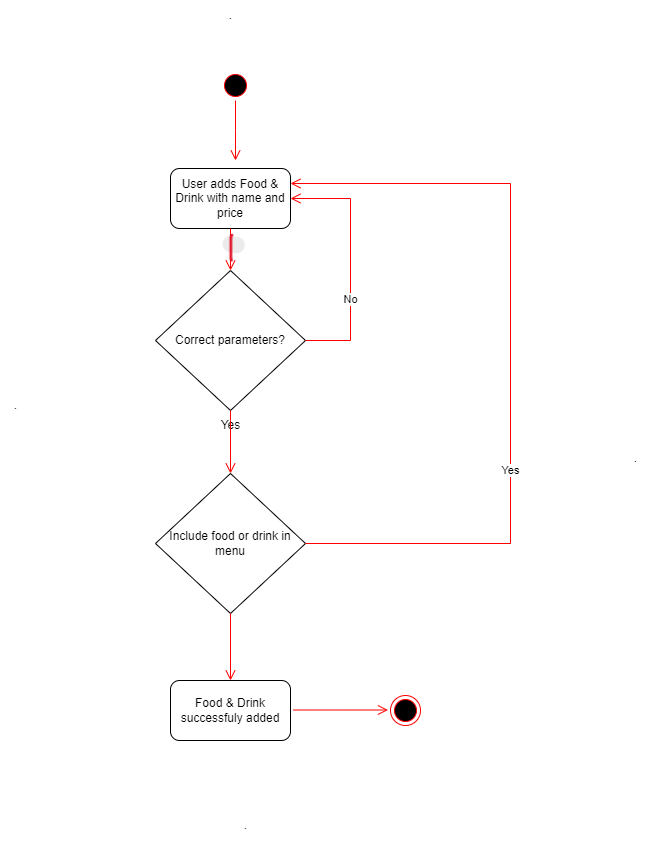


Activity Diagram:

The client enters the name and password to log in. The client has to enter the correct password in less than 3 attempts otherwise the system will terminate itself. If the password is correct the client successfully logs in.

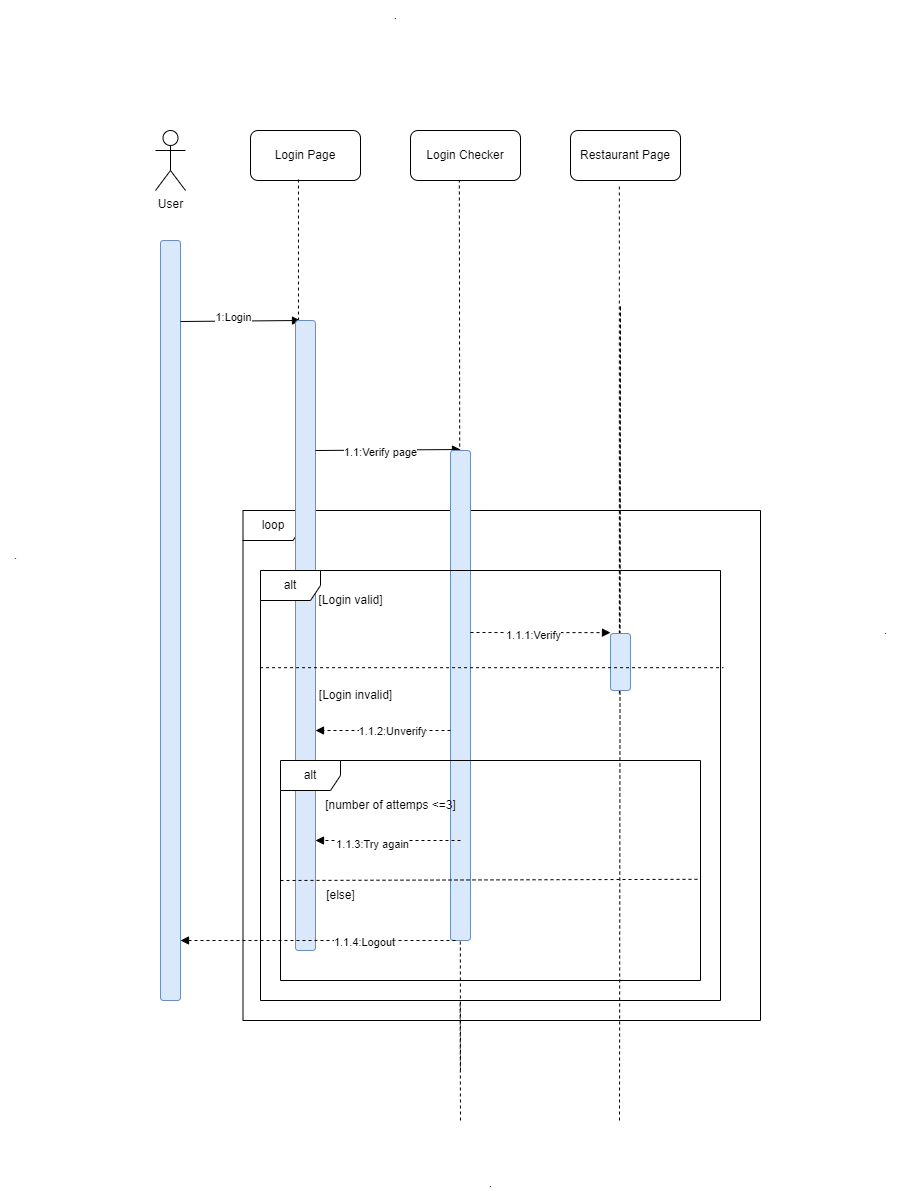


When the client wants to add a new item to the menu, the system checks its parameters first. If they are not correct, the system asks for new parameters. But if they are correct, the next step is searching for the item in the menu. If the item is in the menu, the system wants a new parameter; if it is not in the menu, the system adds it to the menu with the given parameters.



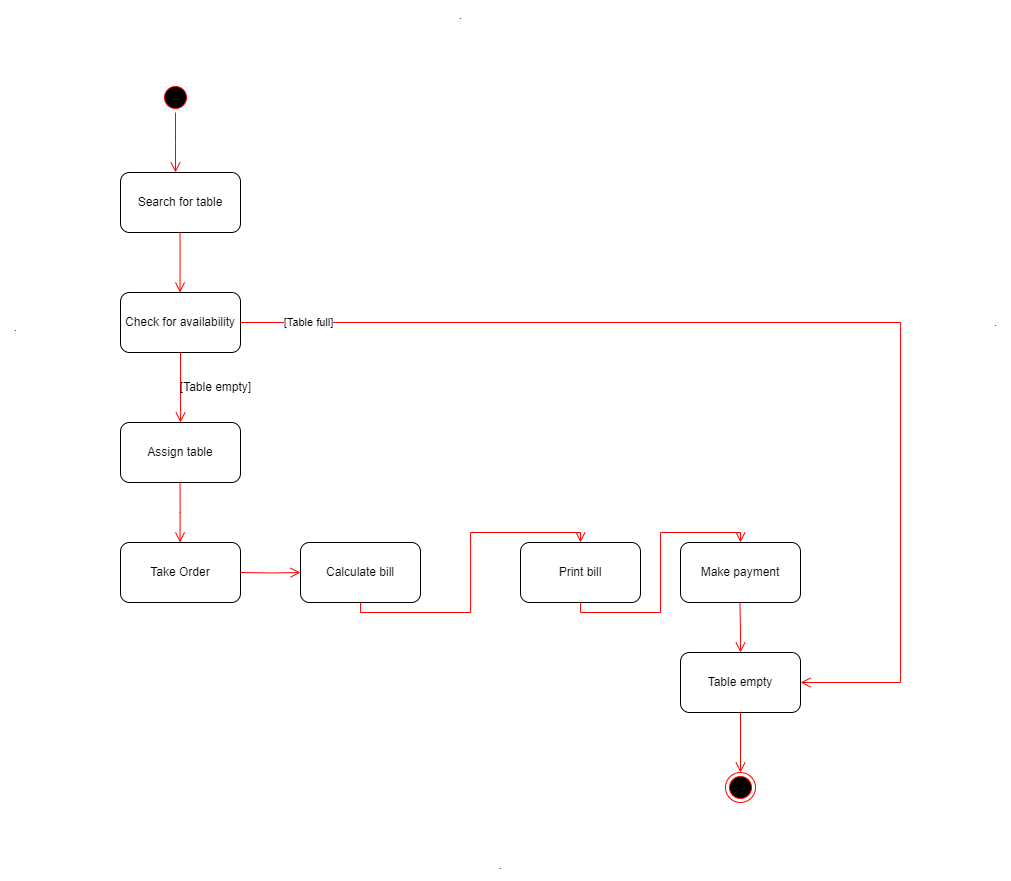
Sequence Diagram:

The user enters login information such as name and password to the login page. Then the system sends them to the login checker to verify. The login checker checks the information, and if it is correct sends the user to the restaurant page. But if it is not correct, the user has 2 more chances to log in. After a total of 3 wrong attempts, the system logs out the user.



State Diagram:

When a new customer comes to the restaurant, the system checks the available table and assigns it to the new customer. After the customer gives an order, the system calculates and prints the bill. After the customer makes the payment, the table becomes available.



**CHAPTER FOUR**

**IMPLEMENTATION**

The Cafe System Project consists of 10 classes. It uses JFrame to open the system page. Chef, Cashier, and Waitress classes extend the Employee class because they have all the same properties. These properties are name, salary, phone number, password, and email. The only difference between these employees is the amount of their salary which is according to their position.

The project uses Singleton Pattern which helps us create one instance from a class. If the instance is created, the system uses that. If it is not created before, the system creates the instance and uses it. The singleton pattern is used in main, menu, and table classes. At the start, there are no instances and they are being made when one is needed. There is only one instance per each, and these instances are used to perform the functions.

Project code starts with creating employee objects such as boss, cashier, chef, and waitress. Then the login screen that is made with JFrame opens. After setting the screen details, the code detects the input name and password. Then opens a suitable page for the user. For example, if the boss has logged in, the boss page that can access “menu, budget, table, and employee” opens.

The program chooses the page to be shown by comparing the input with page owners’ names, then forwards the code to that function. Examples of that function are openBossScreen(), openCashierScreen(), openChefScreen(), and openWaitressScreen(). If the input can not match with these, an error is shown on the screen. Or if failed attempts are 3, the program closes itself.

Information about employees can be changed if the boss page is open. On the boss page, some options let the boss see what they can do. The user enters what they want to do and the code opens the page for that operation.

**CHAPTER FIVE**

**CONCLUSION AND FUTURE WORKS**

In conclusion, The Cafe System Project is done. But like everything in life, it is open to updates and changes. It can be customized according to the requests of the clients or restaurants.

In summary, this project is a cafe system that includes menu and employee information. It starts with the login page and goes to employee-specialized pages according to the entered name. Also, the system has password control, so unauthorized entry is not allowed.