Kingdom of Saudi Arabia Ministry of Education
Imam Abdulrahman bin Faisal University
College of Science and Humanities
Computer Science Department

جامعة الإمام عبدالرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY
كلية العلوم والدراسات الإنسانية - علوم الحاسب
College of Science and Humanities - Computer Science

Food to Door

Reem Shaker Almualem
Layan Saleh Algamdi
Narjis Al-Jumaia

Computer Science

1443هـ

Table of content

Table of figures

Introduction

With the emergence of modernity, computers and smartphones have become part of life for accessing almost all kinds of information. Life in this century is full of technological advancement. And in this technology era, it is very difficult for any organization to develop without utilizing technology.

In these days of fast food and take-out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Especially over the past year until now with the COVID-19 spread.

In this project, the proposed system is specially designed for the cafeteria of the College of Sciences and Humanities. In this application there are all the restaurants inside the cafeteria, this application makes it easier for those inside the college to buy from the cafeteria while they are there, so they can order from the application at any time and when the order arrives at the chosen restaurant, the order is prepared and delivered to them as quickly as possible. As well as it makes it easier for the employees who work in the cafeteria to add and remove products from the menu.

The problem statements

Developing this app was triggered by some issues that students and other members face when they go to the cafeteria. Some students and members do not like to go to the cafeteria to avoid getting the COVID-19 infection. Other students do not have enough time as they study and complete some assignments. Moreover, the employees face problems with the number of students who come to ask if the product is available or not. Another reason is that some students cannot buy from the cafeteria because of the crowdedness. As well as the college lacks the use of technology especially with the spread of it.

The target

The goal of this project is to make it easy and fast for the users to buy from any restaurant inside the cafeteria, with one click they find their order beside them. Moreover, the system will help the employee to edit the menu as quickly as possible, as well as to make the college advanced and in line with the age of technology. Furthermore, this app will help to reduce the crowd of students and members who go to the cafeteria. Also, users will be able to check the menu and the prices of all items online instead of asking for the price during walk-in buying. COVID-19 infection will be reduced as not too many students and members will enter the cafeteria.

UML

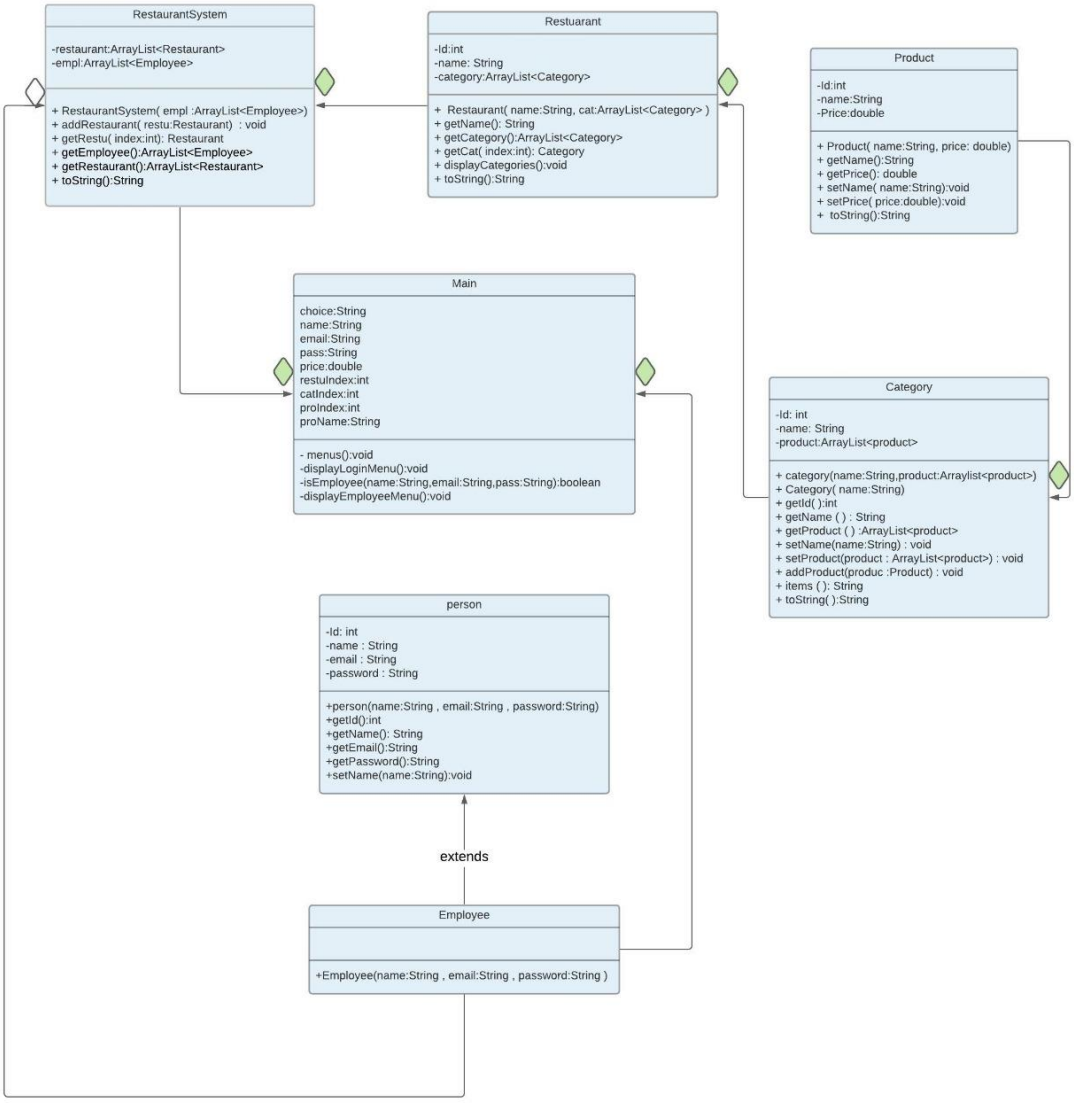

Figure (1) *UML Diagram*

Full code

```
import java.util.ArrayList;
import java.util.Scanner;
```

```
public class Main {

    private static RestaurantSystem sys;
    private static Employee e;


    public static void main(String[] args) {
        menus();

        displayLoginMenu();

    }


    private static void menus() {
        ArrayList<Product> s = new ArrayList<Product>();
        s.add(new Product("Chicken Shawarma", 8));
        s.add(new Product("Falafel", 6));
        s.add(new Product("Chicken burger", 8));
        s.add(new Product("Chicken kabab", 14));

        Category sandwiches1 = new Category("Sandwiches", s);

        ArrayList<Product> d = new ArrayList<>();
```

```
d.add(new Product("Espresso ", 7));
d.add(new Product("Cappuccino ", 8));
d.add(new Product("Americano  ", 7));
d.add(new Product("Hot chocolate ", 8));
d.add(new Product("Latte  ", 8));
d.add(new Product("Lemon mojito  ", 4));
d.add(new Product("Strawberry mojito  ", 14));

Category drinks1 = new Category("Drinks", d);

ArrayList<Category> menu1 = new ArrayList<>();
menu1.add(drinks1);
menu1.add(sandwiches1);

Restaurant rest1 = new Restaurant("Tawat Rahimah", menu1);

//second resturant
ArrayList<Product> s1 = new ArrayList<Product>();
s1.add(new Product("Club sandwich", 8));
s1.add(new Product("Omelette sandwich ", 4));
s1.add(new Product("Halloumi ", 6));

Category sandwiches2 = new Category("Sandwiches", s1);

ArrayList<Product> sa = new ArrayList<Product>();
sa.add(new Product("Green salad", 5));
sa.add(new Product("Shaw salad ", 8));
```

```
Category salads2 = new Category("Salads", sa);

ArrayList<Product> de = new ArrayList<Product>();
de.add(new Product("Cookies", 6));
de.add(new Product("Mango sandwich", 6));

Category desert2 = new Category("Deserts", de);

ArrayList<Product> cold = new ArrayList<Product>();
cold.add(new Product("Ice americano", 10));
cold.add(new Product("Ice mocha", 15));
cold.add(new Product("Orange juice", 4));
cold.add(new Product("Water ", 1));

Category colddrink2 = new Category("Cold drinks", cold);

ArrayList<Product> hot = new ArrayList<Product>();
hot.add(new Product("Tea", 3));
hot.add(new Product("Cappuccino", 12));
hot.add(new Product("Espresso ", 6));

Category hotdrink2 = new Category("Cold drinks", hot);

ArrayList<Category> menu2 = new ArrayList<>();
menu2.add(sandwiches2);
menu2.add(salads2);
menu2.add(desert2);
menu2.add(colddrink2);
```

```
        menu2.add(hotdrink2);

        Restaurant rest2 = new Restaurant("Shawmi", menu2);
        ArrayList<Employee> empl = new ArrayList<>();
        empl.add(new Employee("Reem", "Reem@gmail.com", "R20"));

        sys = new RestaurantSystem(empl);
        sys.addRestaurant(rest1);
        sys.addRestaurant(rest2);
    }

    private static void displayLoginMenu() {

        Scanner input = new Scanner(System.in);
        do {
            System.out.println("          WELCOME TO FOOD TO DOOR
APPLICATION     ");
            System.out.println("\nHere you will find all the resturants inside the College
of Science and Humanities \n");

            System.out.println("Enter '1' to Login as 'Employee'");

            System.out.println("Enter '2' to Exit");
            System.out.print("Your Choice : ");

            String choice = input.nextLine();
            String name, email, pass;

            switch (choice) {
```

```
            case "1":
                System.out.print("\nEnter your name : ");
                name = input.nextLine();
                System.out.print("Enter your email : ");
                email = input.nextLine();
                System.out.print("Enter your password : ");
                pass = input.nextLine();

                boolean isEmpl = isEmployee(name, email, pass);
                if (isEmpl) {
                    displayEmployeeMenu(name);
                } else {
                    System.out.println("Sorry!, You are not in the system as
employee.\n");
                }
                break;


            case "2":
                return;
            default:
                System.out.print("Please make your choice from the list.");

        }
    } while (true);
}

private static boolean isEmployee(String name, String email, String pass) {
    for (Employee e : sys.getEmployee()) {
```

```
            if (e.getEmail().equals(email) && e.getPassword().equals(pass) &&
e.getName().equals(name)) {
                return true;
            }
        }
        return false;
    }



    private static void displayEmployeeMenu(String name) {
        Scanner input = new Scanner(System.in);
        do {
            System.out.println("\n\nWelcome " + name + " in the restaurant
application\n\n");

            System.out.println("Enter '1' to Add Product to the menu");
            System.out.println("Enter '2' to Remove product from the menu");
            System.out.println("Enter '3' to Go Back");

            System.out.print("Your Choice: ");

            String choice = input.nextLine(), proName;
            double price;
            int restuIndex,
                catIndex, proIndex;
            switch (choice) {
                case "1":
                    System.out.println("\n" + sys);
```

```
                System.out.print("Enter Restaurant number: ");


                    restuIndex = Integer.parseInt(input.nextLine());
                    if (restuIndex < 0 || restuIndex >= sys.getRestaurant().size() + 1) {
                        return;
                    }
                    sys.getRestu(restuIndex - 1).displayCategories();
                    System.out.print("Enter Category number: ");
                    catIndex = Integer.parseInt(input.nextLine());
                    if (catIndex < 0 || catIndex >= sys.getRestu(restuIndex -
1).getCategory().size() + 1) {
                        System.out.print("Error: out of bound\n");
                        return;
                    }
                    System.out.print("Enter Name the product : ");
                    proName = input.nextLine();
                    System.out.print("Enter Price the product : ");
                    price = Double.parseDouble(input.nextLine());
                    sys.getRestu(restuIndex - 1).getCat(catIndex - 1).addProduct(new
Product(proName, price));
                    System.out.println();
                    sys.getRestu(restuIndex - 1).displayCategories();
                break;
            case "2":
                System.out.println(sys);
                System.out.print("Enter Restaurant number: ");
                    restuIndex = Integer.parseInt(input.nextLine());
                    if (restuIndex < 0 || restuIndex >= sys.getRestaurant().size() + 1) {
```

```
                    System.out.print("Error: out of bound");
                    return;
                }
                sys.getRestu(restuIndex - 1).displayCategories();
                System.out.print("Enter Category number : ");
                catIndex = Integer.parseInt(input.next());
                if (catIndex < 0 || catIndex >= sys.getRestu(restuIndex -
1).getCategory().size() + 1) {
                    System.out.print("Error: out of bound");
                    return;
                }

                System.out.print("Enter product number : ");
                proIndex = Integer.parseInt(input.next());
                if (proIndex < 0 || proIndex >= sys.getRestu(restuIndex -
1).getCat(catIndex - 1).getProduct().size() + 1) {
                    System.out.print("Error: out of bound");
                    return;

                }
                sys.getRestu(restuIndex - 1).getCat(catIndex -
1).getProduct().remove(proIndex - 1);
                System.out.println();

                sys.getRestu(restuIndex - 1).displayCategories();
            break;
        case "3":
            return;
        default:
```

```
                System.out.print("Please make your choice from the list.");

            }
        } while (true);

    }
}

class RestaurantSystem {

    private ArrayList<Restaurant> restaurant;

    private ArrayList<Employee> empl;

    public RestaurantSystem(ArrayList<Employee> empl) {

        this.restaurant = new ArrayList<>();
        this.empl = empl;
    }

    public void addRestaurant(Restaurant restu) {
        this.restaurant.add(restu);
    }

    public Restaurant getRestu(int index) {
        return restaurant.get(index);
    }
```

```
    public ArrayList<Employee> getEmployee() {
        return empl;
    }

    public ArrayList<Restaurant> getRestaurant() {
        return this.restaurant;
    }

    @Override
    public String toString() {
        String output = "";
        for (int i = 0; i < restaurant.size(); i++) {
            output += (i + 1) + "- " + restaurant.get(i).getName() + "\n";
        }
        return output;
    }

}


class Restaurant {

    private static int id = 0;
    private String name;
    private ArrayList<Category> category;
```

```
public Restaurant(String name, ArrayList<Category> cat) {
    id++;
    this.name = name;
    this.category = cat;

}

public String getName() {
    return name;
}

public ArrayList<Category> getCategory() {
    return category;
}

public Category getCat(int index) {
    return category.get(index);
}



public void displayCategories() {
    int i = 0;
    for (Category c : category) {
        System.out.println("||" + (++i) + "|| " + c);
    }
}
```

```
    @Override
    public String toString() {
        return name;
    }

}


class Product {

    private static int id = 0;
    private String name;
    private double price;

    public Product(String name, double price) {
        id++;
        this.name = name;
        this.price = price;
    }

    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }
```

```
    public void setName(String name) {
        this.name = name;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    @Override
    public String toString() {
        return "Name: " + String.format(" %-20s", name) + ", Price: " +
String.format("%.2fSR", price);
    }

}

class Category {

    private static int id = 0;
    private String name;
    private ArrayList<Product> product;

    public Category(String name, ArrayList<Product> product) {
        this.id++;
        this.name = name;
        this.product = product;
    }
```

```
public Category(String name) {
    this.id++;
    this.name = name;
    this.product = new ArrayList<>();
}

public static int getId() {
    return id;
}

public String getName() {
    return name;
}

public ArrayList<Product> getProduct() {
    return product;
}

public void setName(String name) {
    this.name = name;
}

public void setProduct(ArrayList<Product> product) {
    this.product = product;
}

public void addProduct(Product produc) {
```

```
        this.product.add(produc);
    }

    public String Items() {
        String it = "";
        int i = 0;
        for (Product p : product) {
            it += (++i)+"- " + p.toString() + "\n";
        }
        return it;
    }

    @Override
    public String toString() {
        return "Category{ " + name + " }\n" + this.Items();
    }

}

abstract class Person {

    private static int id = 0;
    private String name;
    private String email;
    private String password;

    public Person(String name, String email, String password) {
        id++;
```

```
        this.name = name;
        this.email = email;
        this.password = password;
    }

    public static int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getEmail() {
        return email;
    }

    public String getPassword() {
        return password;
    }

    public void setName(String name) {
        this.name = name;
    }

}

class Employee extends Person {
```

```
    public Employee(String name, String email, String password) {
        super(name, email, password);
    }

}
```

Conclusion

In brief, the program is aimed to serve both students and faculty members as well as people who work in the cafeteria. The program is divided into two main sections (Employee section and Customers section). For the employee section, the employee must enter a valid username, email, and password on the log-in page to access the program. The main feature for this section is to provide the employee access to modify the menu by adding or deleting information such as product name, product price. The employee section was developed and implemented using the JAVASCRIPT OOP program. For more advanced work, the customer section where the students and faculty members can log-in, make online orders, and pay using this program was left to be developed and implemented as a future recommendation.