Java Programming

Report #1: Object-oriented programming

ARM (Application for Restaurant Management)

Class : 18CLC2-KTPM

Group 05:

Đinh Hoàng Dương – 18127084

Dương Trần Mẫn Duy – 18127087

Huỳnh Đức Lê – 18127126

Table of content

Revision History

[*Provide in this section a revision history table. A such sample table is given below*]

Date	Version	Description	Author
10/11/2020	0.0.1	Prototype login	Hoàng Dương, Mẫn Duy, Đức Lee

Introduction

- Our app will be used by restaurants' managers and their employees to support buy/sell, storage, statistic
- People will need an application to support their business. In this case, we provide solution for managing income, material, human resource, etc. We aim to an univeersial application that can be used not only for restaurants but many similar kinds of businesses
- Propose the requirements:

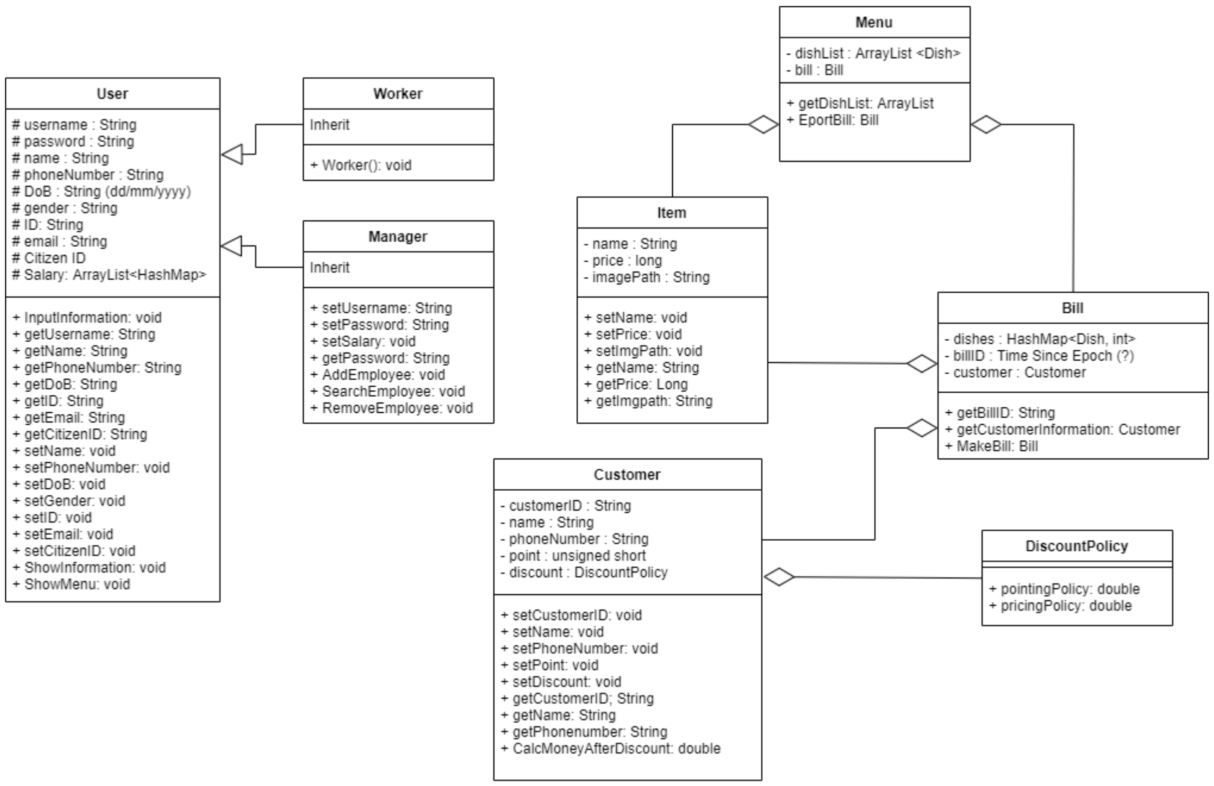
1. Sign-in and Sign-up
    - 1.1. Hash password
2. Sale
    - 2.1. Show menu
    - 2.2. Choose/change items to buy
    - 2.3. Export and store bills
3. Customers
    - 3.1. Membership registeration
    - 3.2. Accumulating points for discount and promotion
4. Statistics (low priority)
    - 4.1. List bills and periodic income
    - 4.2. Periodic saled products
    - 4.3. Storage management
5. Employee information

5.1. Display and modify employees' profile

5.2. Create/Modify account only available for manager

- This project is to be used
- Also present the expected outcome of this project

Analysis and design



Models	Views	ViewModels
Contain mainly data	Have UI elements	Connect/Handle data flow between Views and Models

- Users can be normal employees or managers. So we need class Users to sell as well as tracking assets because this app can be used for both employees and managers, and managers have higher privilege to manage the restaurant.

- Class Menu is responsible for showing Item and Bill
- Class Bill not only shows chosen Item but also discount
- Class Item represents for dishes that would be display to customers
- Restaurant promotes some kinds of special offer for familiar/membership so we need class Customer to store their data
- Different Customer have different offer, class DiscountPolicy will instruct how to calculate

MVVM (Model View ViewModel) pattern:

- Splitting UI elements and logic code behind, easy to maintain
- Front-end team and back-end team can work simultaneously

Implementation

```
// Containing UI elements of login page
// This is landing page when open app
Views.LoginView.java:
// Connection to data as mentioned above
private LoginViewModel loginViewModel = new LoginViewModel();
// Handle when user click Sign-in, send propriate data to LoginViewModel
private void onSignInButtonClick();
// Handle when user click Sign-up, send propriate data to LoginViewModel
private void onSignUpButtonClick();

// Do logic code behind
ViewModels.LoginViewModel.java:
// Receive signal when user click login
// Return User object if login successful
// Null otherwise
public CompletableFuture<User> loginAsync(String user, String pass);
// Receive signal when user click sign-up
// Return true
// False otherwise
public CompletableFuture<Boolean> signUpAsync(User user);

// Super class of Worker and Manager
Models.User.java:
public class User {
        protected String username, password;
        protected String name;
        protected String phoneNumber;
        protected String DoB;
        protected String gender;
        protected String ID;
        protected String email;
        protected String CitizenID;
        protected double salary;
```

```
       public User(){}

       public void InputInformation(){}

       //GETTER

       public String getUsername(){
           return username;
       }
       public String getName(){
           return name;
       }
       public String getPhoneNumber(){
           return phoneNumber;
       }
       public String getDoB(){
           return DoB;
       }
       public String getGender(){
           return gender;
       }
       public String getID(){
           return ID;
       }
       public String getEmail(){
           return email;
       }
       public String getCitizenID(){
           return CitizenID;
       }

       //SETTER
       public void setName(){}
       public void setPhoneNumber(){}
       public void setDoB(){}
       public void setGender(){}
       public void setID(){}
       public void setEmail(){}
       public void setCitizenID(){}

       //Function
       public void ShowInformation(){}
       public void ShowMenu(){}
}
```

```
// Containing employee data when they login
```

Models.Worker.java:

```
public class Worker extends User{
   public Worker(){}
}
```

// Containing manager data when they login, manager have right to deal with all employee's data

<u>**Models.Manager.java:**</u>

```
public class Manager extends User{
   public Manager(){}

   //SETTER
   public void setUsername(){}
   public void setPassword(){}
   public void setSalary(){}

   //GETTER
   public String getPassword(){
       return password;
   }

   //Function
   public void AddEmployee(){}
   public void SearchEmployee(){}
   public void RemoveEmployee(){}
}
```

// Containing all item for sale and provide it to customer, also as the bill

<u>**Models.Menu.java:**</u>

```
public class Menu {
   private ArrayList<Items> dishList = new ArrayList<>();
   private Bill bill = new Bill();

   public Menu(){}

   public ArrayList getDishList(){
       return dishList;
   }

   public Bill ExportBill(){
       return bill;
   }
}
```

// Containing item's data (name, price)

<u>**Models.Items.java:**</u>

```
public class Items {
   private String name;
   private Long price;
   private String imgPath;

   public Items(){}

   //SETTER
   public void setName(){}
   public void setPrice(){}
   public void setImgPath(){}
   //GETTER
   public String getName(){
       return name;
   }
   public Long getPrice(){
       return price;
   }
   public String getImgPath(){
       return imgPath;
   }
}
```

// Containing chosen items, price and total cost
Models.Bill.java:

```
public class Bill {
   private HashMap<Items, Integer> dishes = new HashMap<>();
   private String billID;
   private Customer customer;

   public Bill(){}

   //These two function will use on getBill()
   private String getBillID(){
       return billID;
   }
   private Customer getCustomerInformation(){
       return null;
   }

   public Bill MakeBill(){
       return null;
   }
}
```

// Containing customer’s data
Models.Customer.java:

```
public class Customer {
   private String customerID;
   private String name;
   private String phoneNumber;
   private Integer point;
   private DiscountPolicy discount;

   public Customer(){}

   //SETTER
   public void setCustomerID(){}
   public void setName(){}
   public void setPhoneNumber(){}
   public void setPoint(){/*Use function in Discount Policy*/}
   public void setDiscount(){/*Use function in Discount Policy*/}
   //GETTER
   public String getCustomerID(){
       return customerID;
   }
   public String getName(){
       return name;
   }
   public String getPhoneNumber(){
       return phoneNumber;
   }

   //Function
   public double CalcMoneyAfterDiscount(){
       return 0;
   }
}
```

```
// Instruct how to calculate bill for customer
```

Models.DiscountPolicy.java:

```
public class DiscountPolicy {
   public DiscountPolicy(){}

   public double PointingPolicy(Bill bill, Customer cus){
       return 0;
   }
   public double PricingPolicy(Bill bill, Customer cus){
       return 0;
   }
}
```

Result

Date	Task Done
10/11/2020	Login function prototype

Plan

Duration	Task	Author
28/10/2020 - 4/11/2020	Research for asynchronous programming	Dương
24/10/2020 (in day)	Decide functions Design UML	Dương, Duy, Lê
6/11/2020 (in day)	Specify login function Agree on design-pattern	Dương, Duy, Lê
10/11/2020 (in day)	Implement prototype Finish PA#1 report	Dương, Duy, Lê
	User Login, Register	
11/11/2020 - 24/11/2020	Implement Database for User, Item, Customer	Duy, Lê
	Design Home screen	Dương
	Implement ViewModel for Home screen	Duy, Lê
25/11/2020-1/12/2020	Design Member screen	Dương
	Implement ViewModel for Member screen	Duy, Lê
2/12/2020-8/12/2020	Design User Info screen	Dương
	Implement ViewModel for User Info screen	Duy, Lê
8/12/2020-21/12/2020	Design Statistic screen	Dương
	Implement ViewModel for Statistic screen	Duy, Lê

References

https://commons.apache.org/proper/commons-codec/download_codec.cgi - DigestUtils

CompletableFuture (Java Platform SE 8) - Asynchronous Code

https://www.flaticon.com/ - For UI elements