Restaurant Reservation System

Objective

- Design a complete object-oriented system.
- Draw a UML Class Diagram.
- Practice on the concept of inheritance and polymorphism.
- Get to know files and exceptions.
- More practice on GUI.

Description

This project is almost a complete simulation for a real-life restaurant reservation system. The system's users are restaurants' owners [Mangers], customers, waiters and cooks. **<u>First</u>**, we have the restaurant reservation system itself with a complete set of customers and employees. **<u>Second</u>**, each restaurant has a set of tables, seats, orders, ...etc. **<u>Third</u>**, employees are divided into waiters, cooks, restaurant managers, each have a username and password and can perform specific actions. **<u>The manager of the restaurant</u>** can view all tables, waiters and cooks of the restaurant and see some statistics of the restaurant. He should also be able to see total money earned by the restaurant today. **<u>The Customer</u>** should be able to see the restaurant and check whether a table exists with the required number of seats, choose whether he wants a table in the smoking or non smoking areas, give his orders via the system and reserve a table.
You are required to develop a simulator for such a system in Java using the OO concepts you have covered in the course till now. **However**, you are also

required to be creative and do some brainstorming with your partner about how a real system works, remember, this is mainly a design course, **so a good use of object-oriented concepts like inheritance, polymorphism, abstract classes and interfaces will be graded significantly.** You are also required to develop a GUI that should support two modes of operations for two main actors: customers and employees. You should work with files too to store the restaurant's information, customer's list for each table, …etc. You will be required to deliver a Class Diagram for your work and the detailed division of labor among the two of you.

Scenario

Your program should at least follow the following scenarios (feel free to add any new features as you see fit):

1. When starting the program you get a prompt to sign in to the application with a username and a password. The user should enter correct credentials to be able to login to his dashboard.
2. If the entered username and password belong to a customer, he should be navigated to the customer dashboard. From the dashboard, he should be able to reserve a table in the restaurant, First he should determine the number of seats required, whether the table is in smoking or non smoking areas, if there is an available table, he is promoted to make an order from a list of predefined dishes in the restaurant, he may order more than one dish from the same type and finally he should be able to see the total price of the selected dishes.

3. If the entered username and password belong to a waiter, he should be navigated to the waiter dashboard. From the dashboard, he should be able to see the scheduled reservations for today each one with the customer name and table number.

4. If the entered username and password belong to a cooker, he should be navigated to the cooker dashboard. From the dashboard, he should be able to see all ordered meals for today so that he can prepare them on time. He shouldn't be able to see the customer name, he is only allowed to see each dish with the table ordering it.

5. If the entered username and password belong to a manager, he should be navigated to the manager dashboard. From the dashboard, he should be able to see full details for today's reservations including customer name, ordered dishes, table number and amount paid by the customer. He should also be able to see the total money earned by the restaurant today.

6. Each user should be allowed to logout his dashboard, this should navigate back to the login screen.

Tasks

You should concentrate on the inheritance and polymorphism part, take your time in design phase so you reach a good object-oriented model supporting at least:

1. Three types of dishes **[Appetizer (10% taxes), Main Course (15% taxes), Desert (20% taxes)]**.
2. Three types of employees **[Manager, Waiter, Cooker]**.
3. One type of customer or more. **[For example: Premium Customers]**

4. Two types of tables or more **[For example: Smoking vs Non-Smoking]**.

- You are encouraged to add more classes and extend more types in your project.
- Your code should read and write to files to save and load data. You should load some data from some files at the program startup, so you will have some employees, customers, dishes, tables.
- You can save the updates automatically to the files or provide a save button to save changes to the files.
- You should provide a good GUI which will ease the use of the system, you are free to use any GUI framework, and you are free to make any design you like.
- You should draw a class diagram of your project.
- You should handle errors and show error messages to the user, use try and catch to help you make your program stable.
- You should apply all the object-oriented concepts like encapsulation, …etc. And don't use static unless you really need it. **Be creative!** The required features are only the beginning of what you can do, add more features or spice up the required ones, bonus marks will be given to those with eye-catching extra features and user-friendly interfaces.
- The input file to the project will be in xml format, it is used to initialize the project data including users, tables and dishes, You should load initial data

from the file on program setup and append any needed data to the file to be loaded later, The file will look like this:

```
<restaurant>
  <users>
    <user>
      <name>Adam Smith</name>
      <role>Manager</role>
      <username>adam</username>
      <password>adam_manager</password>
    </user>
    <user>
      <name>John Doe</name>
      <role>Client</role>
      <username>john</username>
      <password>john_doe</password>
    </user>
    <user>
      <name>Brian Johnes</name>
      <role>Client</role>
      <username>brian</username>
      <password>mdir@admj%ar5qX2</password>
    </user>
    <user>
      <name>George Robben</name>
      <role>Waiter</role>
      <username>george</username>
      <password>k6987_#LpQ</password>
    </user>
    <user>
      <name>Tomas Hobbens</name>
      <role>Cooker</role>
      <username>hobbens_tom</username>
      <password>cooker_pass</password>
    </user>
  </users>
```

```
<tables>
    <table>
        <number>1</number>
        <number_of_seats>5</number_of_seats>
        <smoking>false</smoking>
    </table>
    <table>
        <number>2</number>
        <number_of_seats>12</number_of_seats>
        <smoking>false</smoking>
    </table>
    <table>
        <number>3</number>
        <number_of_seats>12</number_of_seats>
        <smoking>true</smoking>
    </table>
    <table>
        <number>4</number>
        <number_of_seats>4</number_of_seats>
        <smoking>false</smoking>
    </table>
    <table>
        <number>5</number>
        <number_of_seats>4</number_of_seats>
        <smoking>true</smoking>
    </table>
    <table>
        <number>6</number>
        <number_of_seats>7</number_of_seats>
        <smoking>true</smoking>
    </table>
    <table>
        <number>7</number>
        <number_of_seats>6</number_of_seats>
        <smoking>true</smoking>
    </table>
```

```
</tables>
<dishes>
    <dish>
        <name>Grilled Chicken</name>
        <price>75</price>
        <type>main_course</type>
    </dish>
    <dish>
        <name>Greek Salade</name>
        <price>35</price>
        <type>appetizer</type>
    </dish>
    <dish>
        <name>Fried Potatos</name>
        <price>30</price>
        <type>appetizer</type>
    </dish>
    <dish>
        <name>Apple Pie</name>
        <price>50</price>
        <type>desert</type>
    </dish>
    <dish>
        <name>Molten Cake</name>
        <price>60</price>
        <type>desert</type>
    </dish>
    <dish>
        <name>Mushroom Soup</name>
        <price>60</price>
        <type>main_course</type>
    </dish>
    <dish>
        <name>Beef Steak</name>
        <price>80</price>
        <type>main_course</type>
    </dish>
  </dishes>
</restaurant>
```

- This project will consume some time in developing, but it has minor difficult parts, so enjoy what you are doing and try to learn as much as you can. It is meant to be easy and to help you learn.
- It is better to use a git version control service like Gitlab to coordinate with your teammate. Interface can be used as a contract between the two of you, it will make the teamwork much smoother.
- You should maintain a clean code for your program, like: variables, classes, functions naming, code indentation, function length, …etc. Again, google is your best friend, the debugger is your second-best friend.

Deliverables

- You should work in groups of two (at maximum).
- You should develop this assignment in java.
- You are free to use any GUI framework you like.
- You should submit this assignment online on **GitLab**, you should include the following:
    - Your code folder.
    - A **self-executable jar file (JRE 1.8)** : The program should be executable by simply double clicking the icon, given that you have a running JRE.
    - **Class diagram**: You are permitted to use any software that allows you to draw the diagram itself not to automatically generate it from java class files.

    - A **readme.txt** file with any assumptions you need to run the program, and with the detailed division of labor among the two of you.

- **[Cheating Policy]:** Delivering a copy will be severely penalized for both parties, so delivering nothing is better than delivering a copy.
- Check these resources for file handling:
    - https://www.vogella.com/tutorials/JAXB/article.html
    - https://howtodoinjava.com/jaxb/jaxb-exmaple-marshalling-and-unmarshalling-list-or-set-of-objects/
    - https://stackoverflow.com/questions/5258159/how-to-make-an-executable-jar-file
    - https://www.youtube.com/watch?v=mE3rbtKm-pk
- Feel free to ask any questions on Piazza as your colleges may have the same question.