A blue and white background

Description automatically generated

**Restaurant system**

*2024 - second semester Milestone 3*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sign** | **Role Participated in** | **ID** | **Name** |  |
| **✔** | **(Leader) All classes** | **2220004853** | **Hassan Mohammed Alzourei** | **1** |
| **✔** | **Account, Dashboard classes** | **2220002074** | **Abdulrahman Mohammed Al-mejna** | **2** |
| **✔** | **Manager, Login classes** | **2220003187** | **Ahmed Al-Shaikh Mohammed-** | **3** |
| **✔** | **Manager, Menu classes** | **2220002664** | **Ali Alkhars** | **4** |
| **✔** | **Cashier, Logout classes** | **2220001392** | **Abdullah Mohmmed Al-Battat** | **5** |
| **✔** | **Arraylist, DB Access classes** | **2220000148** | **Baqer Abdullah Alhaddad** | **6** |

**Description: Restaurant system:**

- The restaurant is divided into two parts: one side is managed by the staff taking orders and the other is managed by the manager who manages the staff, the menu content, and can make salary reports.

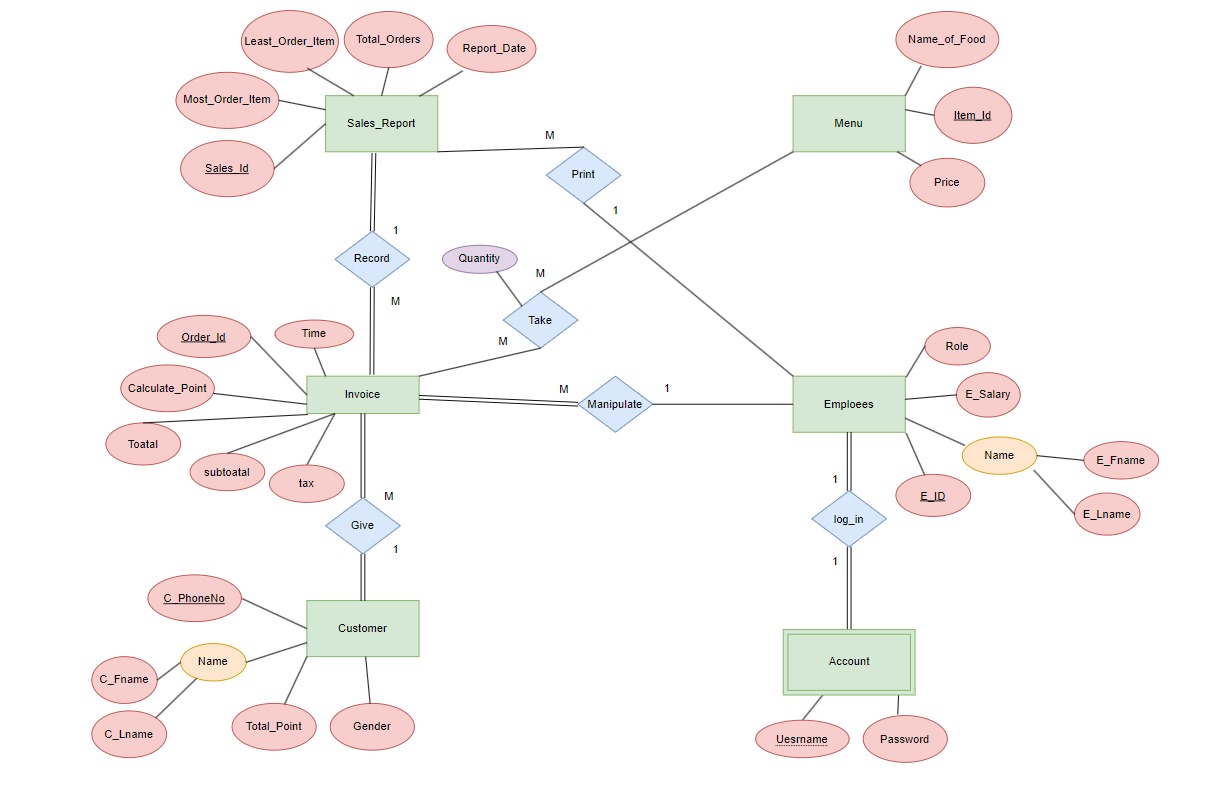
- The cashier employee takes customer orders and information to complete the order through the system and registers customers who want to earn points to make a discount after a while of orders.

- The customer information will assist us in sending promos and points to them.

- The invoice will have information of the order like the date of order, how many items have been ordered and information about customers with points they earned, and so on

- The manager can access sales reports which can also know the most order items, the least order items, and the total orders. The manager can also modify the menu, and it is possible to add items and set a price for each item, or it is possible to delete items.

**ER Diagram**



**Relation Schema**

A screenshot of a computer

Description automatically generated

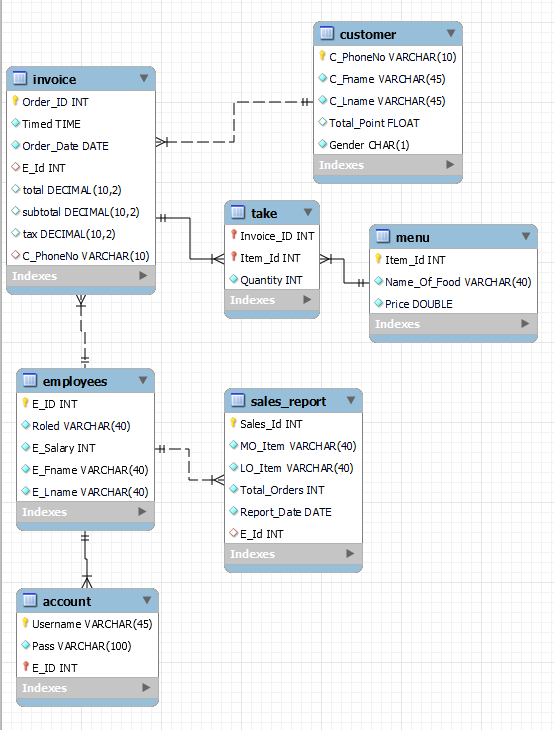
**Classes hierarchy**

A computer screen shot of a computer screen

Description automatically generated

**UML Link:** [**https://lucid.app/lucidchart/8167285c-e57a-4b75-85b6-c3e26a4c6e39/edit?view\_items=spuCvVJQvm9k%2CspuCge0xdjlT%2CspuC\_y7tQIc0%2CW1Epm3fYOYhh%2CspuC3.aNTyKv%2CspuCkeMOOM18%2Cv4EpPgvFBK13%2CspuCWs2Wfs~i%2CspuCY-**](https://lucid.app/lucidchart/8167285c-e57a-4b75-85b6-c3e26a4c6e39/edit?view_items=spuCvVJQvm9k%2CspuCge0xdjlT%2CspuC_y7tQIc0%2CW1Epm3fYOYhh%2CspuC3.aNTyKv%2CspuCkeMOOM18%2Cv4EpPgvFBK13%2CspuCWs2Wfs~i%2CspuCY-)

**Relational Schema Mapping**

****

**Employee**

A screenshot of a computer

Description automatically generated

**Account**

A screenshot of a computer

Description automatically generated

**Customer**

A screenshot of a table

Description automatically generated

**Invoice**

A screenshot of a table

Description automatically generated

**Sales\_Report**

A screenshot of a menu

Description automatically generated

**Menu**

A table with text on it

Description automatically generated

**Take**

A screenshot of a table

Description automatically generated

**Difficulties & How Problems were Solved:**

 **Look and Feel Configuration:**  
Difficulty: The code includes multiple look and feel configurations, such as FlatLaf and FlatMacLightLaf, which could lead to inconsistencies or conflicts.

Solution: Review and consolidate the look and feel configurations to ensure they are applied consistently throughout the application. Choose a single look and feel that best suits the application's design and stick with it.

 **Unused Variables and Imports:**  
Difficulty: The code contains unused variables and imports, which can clutter the codebase and make it harder to maintain.

Solution: Remove any unused variables and imports to improve code readability and maintainability. This can be done by identifying which variables are not used and safely removing them from the code.

 **Database Connectivity:**  
Difficulty: The code does not include the implementation of database connectivity, even though there are imports related to database operations.

Solution: Implement the necessary code for establishing database connectivity using the appropriate libraries and APIs. This may involve configuring the database connection, executing queries, and handling exceptions related to database operations.

 **User Interface Design and Functionality:**  
Difficulty: The code includes user interface elements and their associated actions, such as buttons and event listeners, but the actual functionality or behavior of these elements is not fully implemented.

Solution: Complete the implementation of the user interface elements by adding the necessary logic and functionality to handle user interactions. This may involve defining actions, validating user input, performing calculations, and updating the UI accordingly.

 **Error Handling and Exception Management:**  
Difficulty: The code lacks proper error handling and exception management. Exceptions are not caught, logged, or displayed to the user, which can make it difficult to identify and troubleshoot issues.

Solution: Implement appropriate error handling mechanisms by using try-catch blocks to catch and handle exceptions. Display meaningful error messages to the user, log exceptions for debugging purposes, and ensure graceful degradation or recovery from errors.

**Checklist of Requirements Fulfilled in the Project:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Criterion** | **Description** |  |
| 1 | Classes Management | Creation of superclass(s)/subclasses (e.g. products) are stored in a dynamic and editable hierarchy, which can easily be browsed and searched. | **✔** |
| 2 | Data Export/ Import | Export / import data from other files **(Bonus)** | **✔** |
| 3 | Professional Appearance | Multi-part forms, logo, appropriate navigation, login status/control. | **✔** |
| Rich Content: User generated and manipulated rich text and multimedia such as images, videos, or sound files |
| 4 | Data Access Tools and Techniques | * DB Design, Constraints, DDL, DML, and read Queries, Semantic Integrity Control of Data. * Login Facility : multiple users (and roles) and login control | **✔** |
| Implementing correct DB connections | **✔** |
| 5 | Error Handling | Expected users' errors should be handled well through:   * Using a proper GUI to limit the user's invalid entry. * Using try-Catch exception handling; throws as well for both checked and unchecked exceptions. * Avoid runtime-errors. * Limiting the user's entry mistakes to the least. | **✔** |

**How were the tasks distributed equally between members?**

1. Hassan Mohammed Alzourei (ID: 2220004853) (Leader):

Responsible for the Dashboard, ArrayList, and DatabaseAccess classes.

1. Abdulrahman Mohammed Al-mejna (ID: 2220002074):
   * Responsible for FormAddEmployee, FormCustomerView, FormDisplayReports,
   * FormEmployeeView,
   * FormInvoiceView,
   * FormRead, LogoutPAGE, CashierLogin, ManagerLogin, and animation for all classes including themes.
2. Ahmed Al-Shaikh Mohammed- (ID: 2220003187):
   * Responsible for the FormGenerateReport class.
3. Ali Alkhars (ID: 2220002664):

Responsible for the FormMenuView class and UML.

1. Abdullah Mohmmed Al-Battat (ID: 2220001392):
   * Responsible for the CustomerRegistration class and worked with Hassan in the class Dashboard.
2. Baqer Abdullah Alhaddad (ID: 2220000148):

PowerPoint and UML.

Tasks were distributed based on the different areas of functionality within the application. The leader (Hassan Mohammed Alzourei) have had a supervisory role.

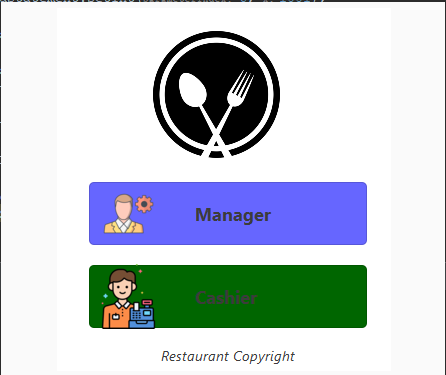
**General Comments:**

1. **UI Design and User Experience:**
   * The UI design, especially in the ManagerLogin class, is visually appealing with the use of images and colors.
   * The use of icons for buttons enhances usability.
   * The layout is generally intuitive, making it easy for users to understand and interact with the application.
   * However, some improvements can be made in terms of consistency in UI design across different frames or forms. For example, ensuring consistent spacing and alignment of components.
2. **Code Modularity and Readability:**
   * The code is well-organized into separate classes, each responsible for specific functionalities like login and logout.
   * Methods and variables are appropriately named, contributing to code readability.
   * However, there are areas where the code could be further modularized and abstracted to promote reusability and maintainability. For instance, database connection setup could be encapsulated in a separate class.
3. **Database Interaction:**
   * The code effectively interacts with the database using JDBC for user authentication.
   * Prepared statements are used, which helps prevent SQL injection attacks.
   * However, there is room for improvement in error handling and reporting to provide more informative feedback to users in case of database-related issues.
4. **Security:**
   * The code handles sensitive user information such as usernames and passwords.
   * Passwords are stored securely as the actual values are not displayed in the UI.
   * However, additional security measures such as password hashing could be implemented to enhance security further.
5. **Testing and Debugging:**
   * The code may require thorough testing to ensure all functionalities work as expected under various scenarios.
   * Debugging tools could be utilized to identify and fix potential runtime errors and logical issues.
6. **Overall Impact:**
   * The application provides essential functionalities for user authentication and access control, which are crucial for any management system.
   * With further refinements and enhancements, the application has the potential to become a reliable and user-friendly tool for managing restaurant operations.

**GUI Interfaces:**

**Choosing Role:**

When the code runs, the user has to choose whether they are a manager or a customer.



**Manager Choice:**

If the user chooses "manager", they will be asked to enter a username and password.

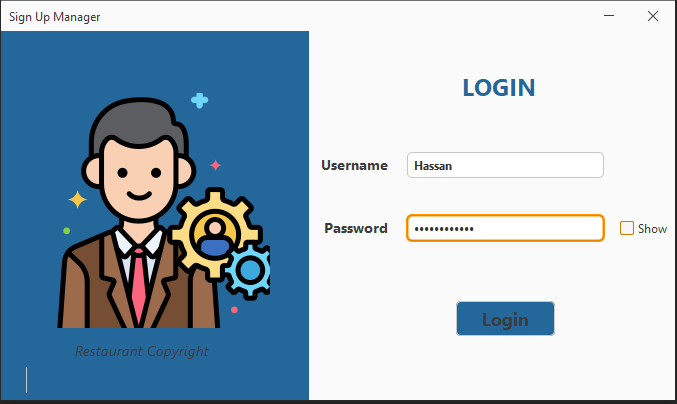


Figure 1: Maneger Login Page

**Lobby Page for Manager:**

"The lobby page for the manager"

Figure 2: Lobby Page for Manager

**Employee Category:**

The manager has two options in this category: either adding an employee or viewing the employees, with options to remove or update an employee, reset, and print employee information as a file.

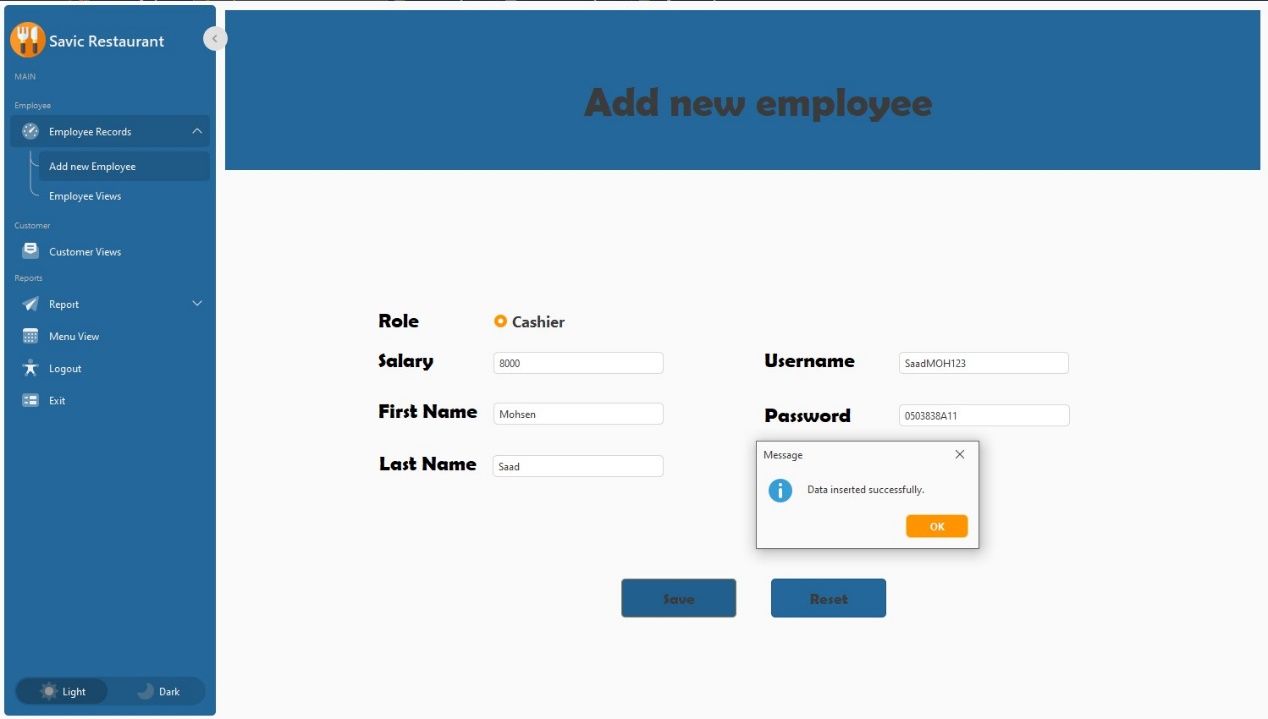


Figure 3: Add new employee interface

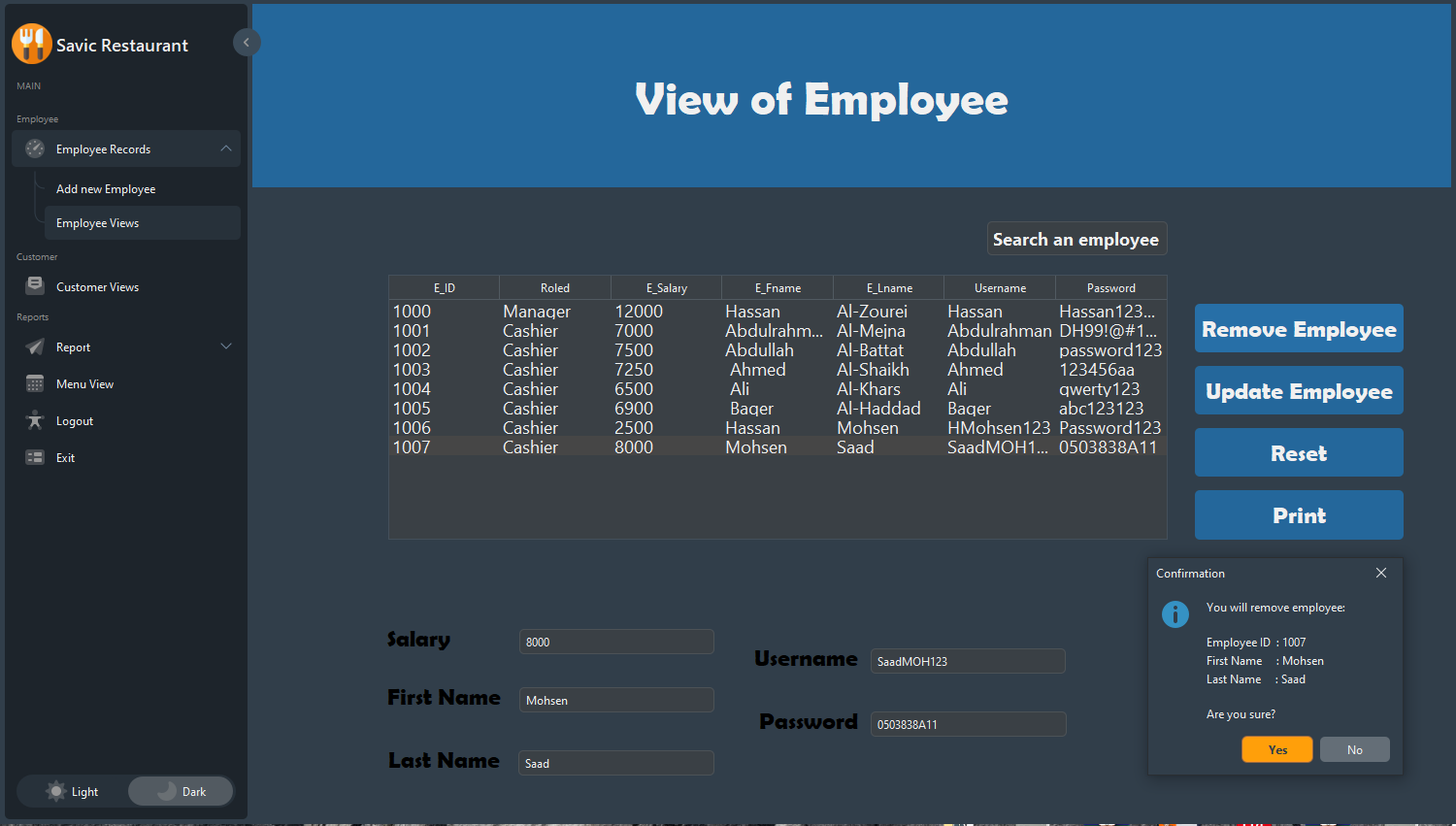


Figure 4: View of employees interface

**Cutsromer View:**

In this category manager can see the customer registered with their informations and can print it as a file.

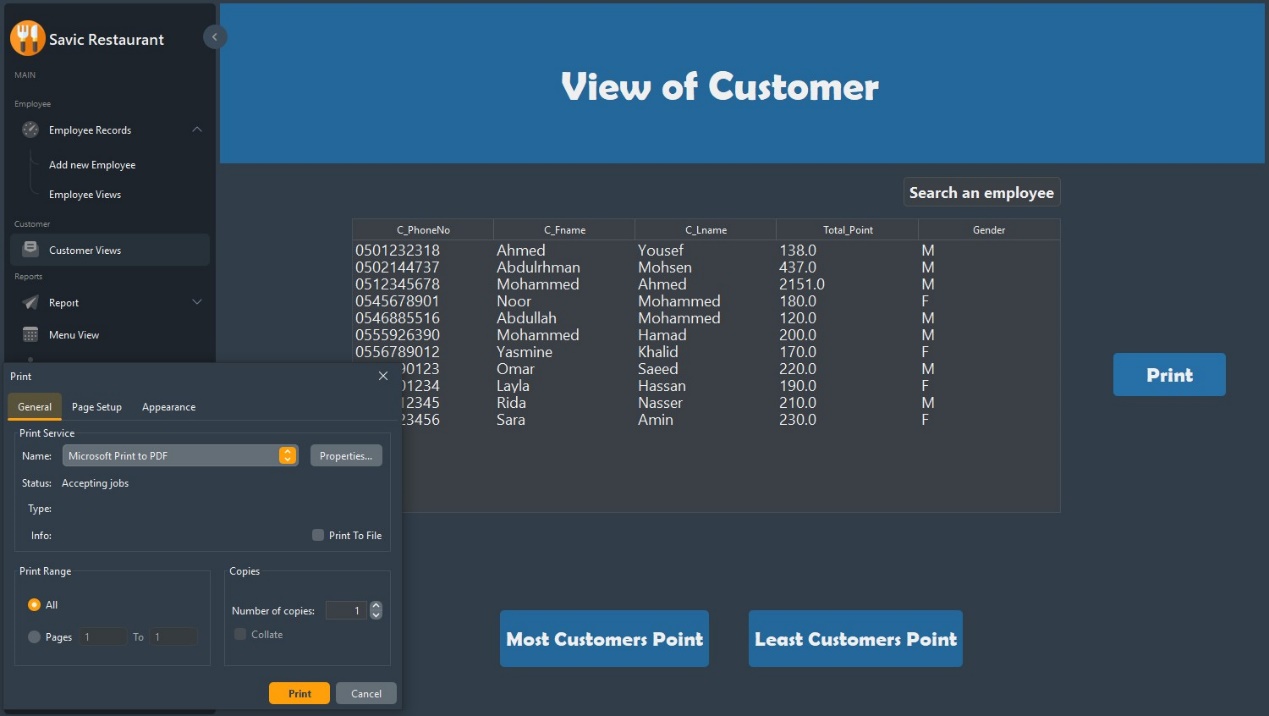


Figure 5: View of Customers

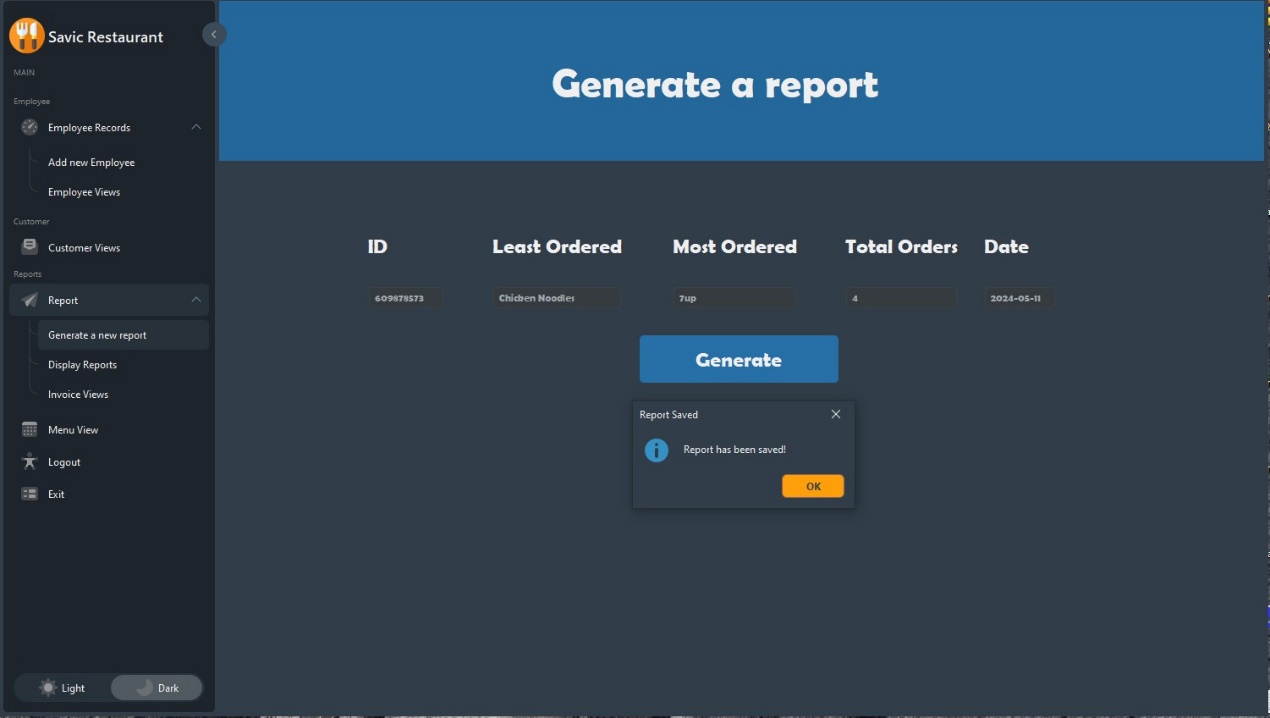
**Reports Category:**In this category, the manager can generate a report for a specific date, display reports with either descending or ascending order for the number of orders for items by selecting a date, and view invoices for today's orders.

Figure 6: Generate reports

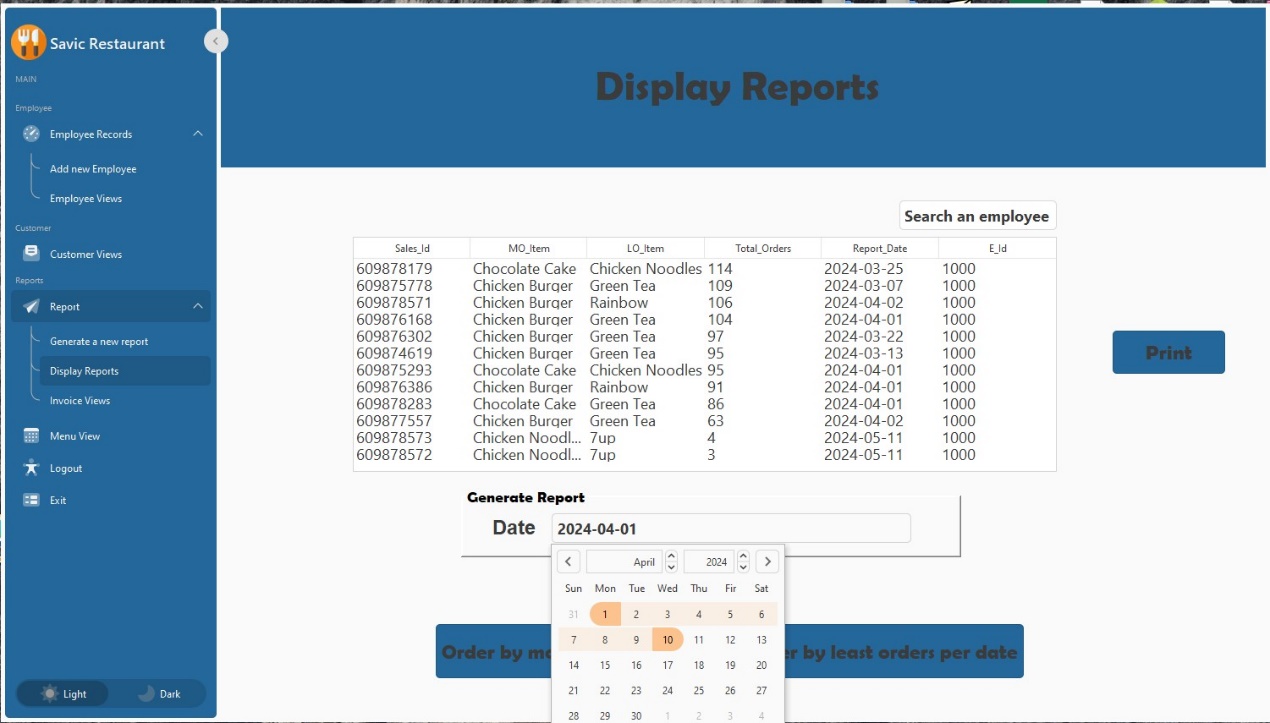


Figure 7: Display Reports

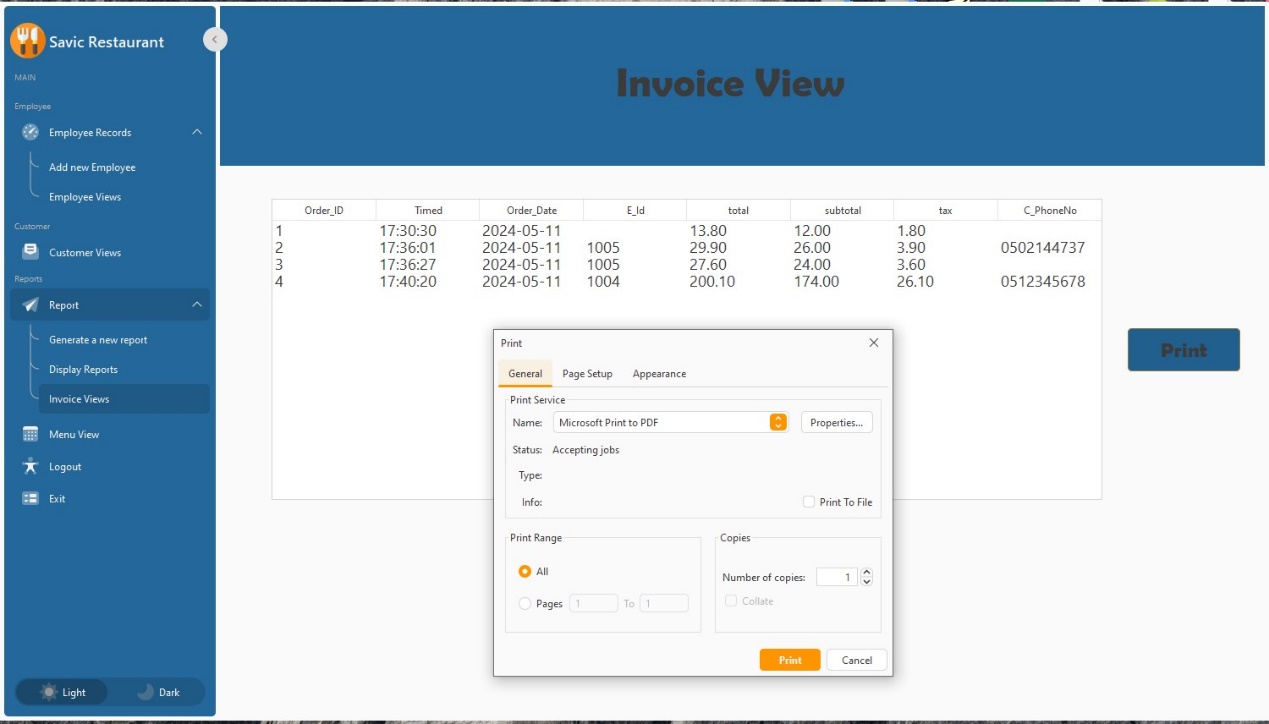


Figure 8: Invoive View

**Menu View Category:**

In this category, the manager can view the menu along with its information and also have the option to print it as a file.

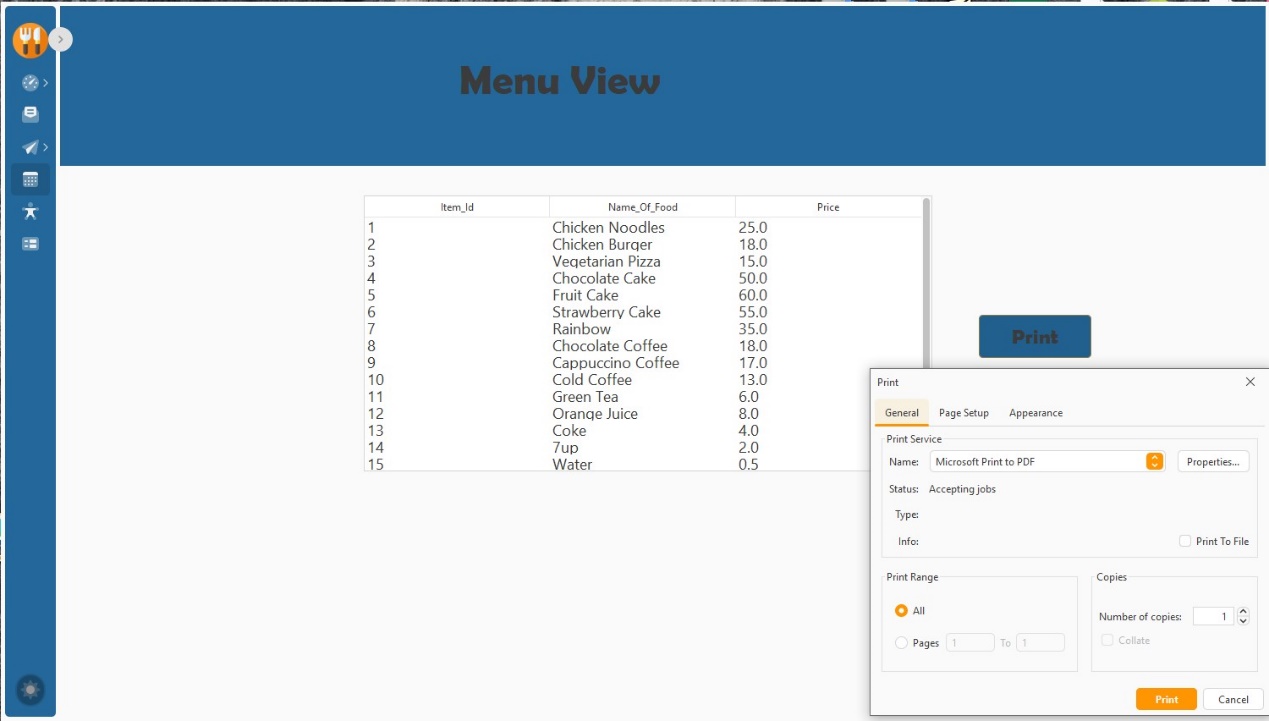


Figure 9: Menu View

When the manager wants to logout, he has two options: either logout and return to the login screen for manager and cashier, or exit and shut down the program.

(Red Lines).

A white background with a blue and white circle with text

Description automatically generated

Figure 9: Logout Options

**s**

**Cashier choice:**

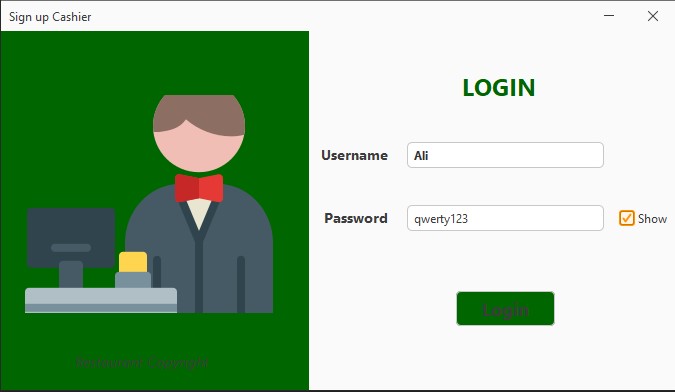
If the user chooses "Cashier", they will be asked to enter a username and password.

Figure 10: Cashier Login

**Lobby Page for Cashier:**

"The lobby page for the cashier"

A logo and text on a white background

Description automatically generated

Figure 11: Lobby Page for Cashier

**The Menu:**

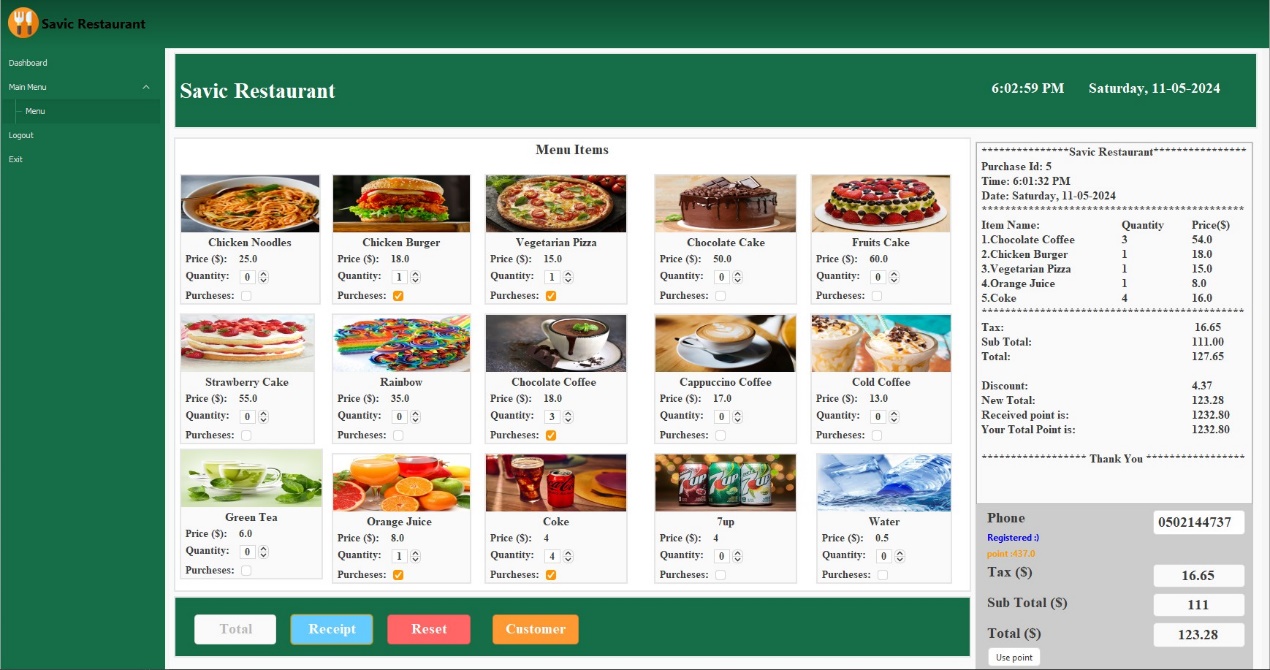
 rom here, the cashier can create orders with the provided invoice details (purchase ID, time, date, item name, quantity, etc.). Additionally, they can register new customers for the restaurant to grant them points for their next order and provide them with discounts.

Figure 12: Menu Dashboard

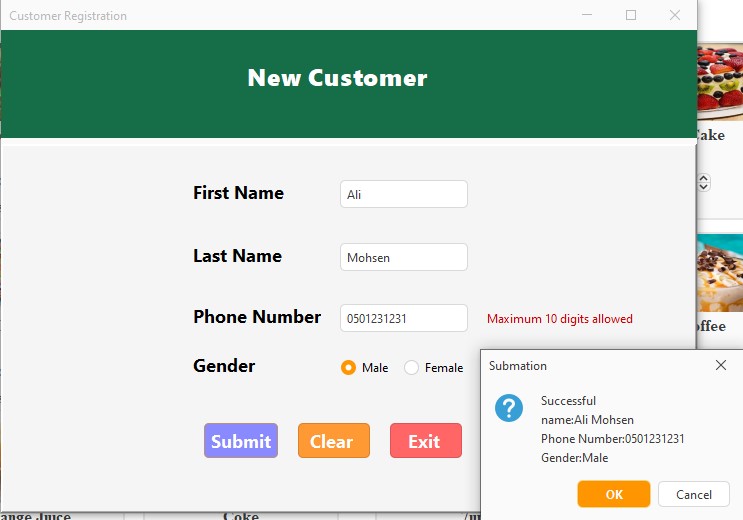


Figure 13: Cutomer Registration

When the cashier wants to logout, he has two options: either logout and return to the login screen for manager and cashier, or exit and shut down the program.

(Red Lines).

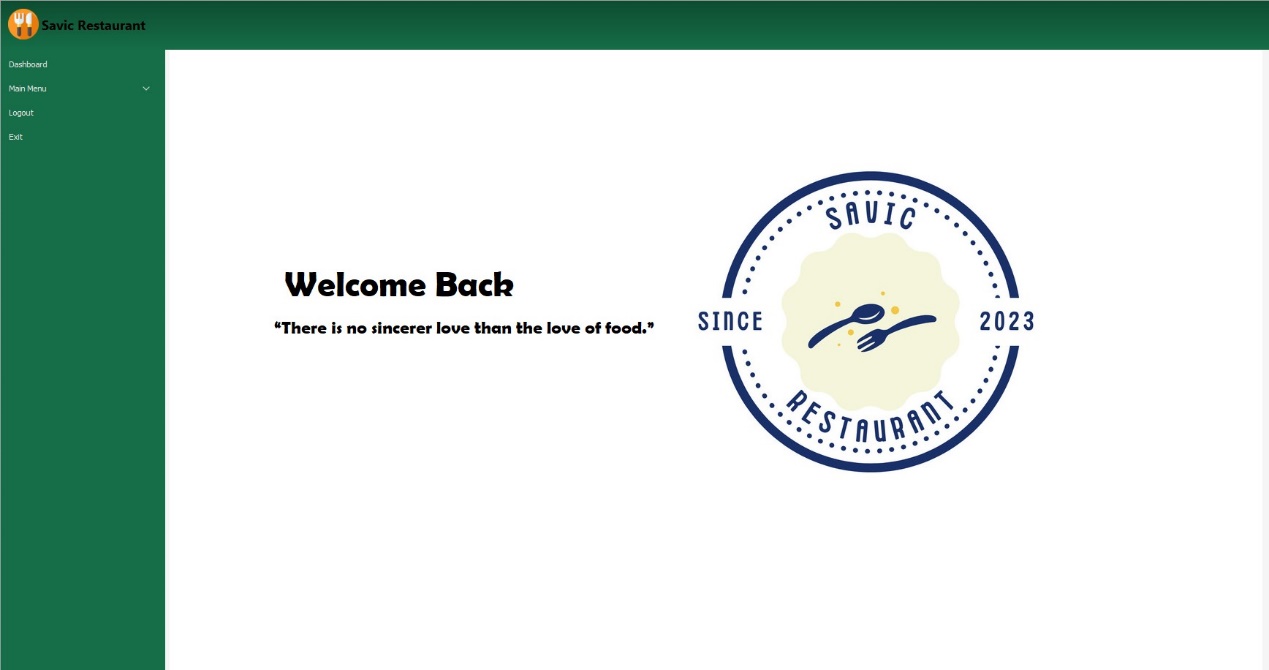




Figure 14: Logout Options