

STAFF PAYROLL MANAGEMENT SYSTEM

Table of Contents

| | |
|----------------------|----|
| Introduction | 2 |
| Problem Statement | 3 |
| Objectives | 4 |
| Program Scope | 5 |
| Detailed description | 7 |
| Screenshots | 15 |

Introduction

Staff payroll management is an integral component of human resource management within any organization. This crucial process involves the computation, distribution, and administration of employee salaries, wages, bonuses, and deductions. To streamline and expedite these complex tasks, organizations employ a Staff Payroll Management System (SPMS), which is a software application designed to automate and enhance the efficiency, accuracy, and reliability of the payroll process.

The primary objective of implementing a Staff Payroll Management System is to simplify the intricate payroll responsibilities carried out by HR departments and payroll administrators. By automating various payroll processes, organizations can achieve significant benefits such as time savings, cost reduction, improved data precision, heightened security, and increased overall productivity. This automation leads to enhanced efficiency by minimizing manual efforts, reducing errors, and expediting payroll processing times.

Moreover, the implementation of a Staff Payroll Management System results in increased accuracy by eliminating the need for manual calculations. The system ensures precise payroll calculations and prevents errors that may arise from human intervention. Additionally, organizations can experience cost savings by streamlining payroll operations, which reduces administrative costs, overhead expenses, and the reliance on manual labor. Consequently, these cost savings contribute to the financial well-being of the organization. Furthermore, timely and accurate payments facilitated by the system enhance employee satisfaction, morale, and retention rates, fostering a positive work environment. By storing and managing employee details in a centralized database, the staff payroll management system ensures data security and confidentiality through the implementation of role-based access controls and encryption.

Overall, the automation provided by the staff payroll management system enables organizations to optimize their payroll operations and improve overall business performance.

Problem Statements

- The current staff payroll management system in the company we are assisting lacks efficiency, which makes it time-consuming and difficult.
- HR staff members have difficulty maintaining and gaining access to employee records as there is restricted Access to Employee Records
- The current payroll management is complicated and error-prone, requiring manual calculations and data entry, leading to inaccuracies and delays in salary processing.
- The current staff payroll management does not have user authentication log in, which makes sensitive employee data at danger from unauthorised access and data breaches
- The current staff payroll management system makes it difficult to create and manage HR accounts as it is cumbersome and lacks user-friendly features, leading to delays in account setup and maintenance.
- Their current system lacks certain functionalities such as employee deletion and payroll management.
- Employee data is stored in a disorganized manner, making it difficult to retrieve and analyze information effectively.
- The current system may not be scalable to accommodate the growing needs of the organization, leading to potential performance issues and system limitations in the future.

Objectives

- To develop an HR management system with a user-friendly interface.
- To allow HR personnel to add new employees to the system.
- To enable viewing of existing employee records.
- To provide functionality for editing employee information.
- To implement the capability to delete employee records when required.
- To facilitate payroll management tasks within the system.
- To enable the program to easily calculate salaries for employees, especially including overtime and bonuses.
- To implement a secure login system to ensure only authorized HR personnel can access the system.
- To include functionality to create new HR accounts as needed.

Program Scope

Employee Information Management consists of modules for the seamless cataloging and administration of employee particulars, including full name, email address, department, title, salary, start date, scheduler, age, and city. It streamlines the retrieval and updating of employee records for HR purposes. Payroll Calculation and Processing automates the computation of workers' employee salaries, incorporating considerations for overtimes and bonuses. Authorized HR staff can modify employee details, such as full name, contact information, department, job designation, remuneration terms, and permanently delete staff profile from digital records when necessary. HR Account Administration provides a secure method for creating, validating, and storing HR user logins to improve security and compliance. The program has a friendly user interface of easy-to-understand graphical elements and patterns, like: login pages, data input forms, command buttons, and status display panels.

The system is designed to quicken HR operations, beginning with a login feature that allows HR personnel to gain entry into the system using the credentials that have been assigned to them. This gives them the ability to easily manage payroll and carry out other crucial duties, without wasting time. HR administrators can update employee details such as names, emails and salaries, as a way to ensure that the database remains current, as it ought to be. Authorized personnel are also allowed to delete employee records when needed. Payroll calculations are automated within the system, and so, with this feature working, the base salary, overtime, bonuses, and deductions of an employee are factored in to accurately compute their total. The software is also designed to enable the generation of new HR accounts, a feature that expands access to the software, meaning that the security and collaboration among HR professional can easily be enhanced.

This program has a user-friendly graphical user interface so that it can help HR admin users to easily manage the payroll related tasks. It allows them to manage payroll details. Reporting feature has been included for the generation of the payroll summaries which makes the access to payroll information handy and quick. The employee and HR account data is stored permanently in files (employers.csv and hr_accounts.txt) which means that data will be available across the sessions and it will be preserved as well. Following are the functional algorithms which make this program fully functional i.e. calculating the payroll, editing employee, HR account validation and all other important aspects, of the program to make it comprehensive solution for the HR Management.

File-based storage is used for employee and HR account data, a design that might limit scalability and performance with large datasets. There's a single HR administrator interface, so it won't manage multi-administrator scenarios easily. That said, the SPMS is intended to make life easier for HR admins while delivering all the functions they need at this scale, and leaving room to grow.

Detailed Description

Below is a detailed description of how the system works, including its inputs and outputs, key functions, file handling, application of object-oriented programming concepts and overall workflow

The system is designed with functionality that hinges on two main types of inputs to ensure smooth operation. Firstly, it requires inputs of employee information, including names, emails, departments, job titles, salaries, dates of hire, ages, and cities. This data is inputted by HR administrators into the system. Additionally, to access the system and engage in payroll management tasks, HR administrators must input valid HR account credentials, such as usernames and passwords.

As for the outputs, the system is equipped to generate detailed payroll calculations, including employee salaries, overtime pay and bonuses for a specific month. It also outputs employee information as display within the system's user interface. Moreover, the system outputs confirmation messages upon the successful completion of various tasks, such as logging in, editing employee details, deleting records, or creating new HR accounts. These outputs are critical for maintaining a smooth workflow in the staff payroll management system.

Next, the functionalities of the system and the overall workflow of the system. The system has got 6 main functionalities, which are to create an HR account for HR administrators, add an employee, view the employees' details, delete employees from the system, edit employee details and calculate employee salary and logging out of the HR account to prevent unauthorized access into the system. The overall flow of the system is that the HR administrator will create a new account, if they don't already have one, and use those details which they used to create the account to log in to the HR Dashboard. The HR Dashboard is where the main functionalities of the system are located. In that dashboard, HR Administrators can add, view, delete and edit employee details, in addition to calculating employees for any respective month. All of the details will be conveniently saved in .txt and .csv files which helps the system to easily process and help HR Administrators regarding staff payroll matters.

Applications of object-oriented programming concepts

GUI (Graphical User Interface) concepts are applied in the provided code snippets to create user-friendly interfaces for interacting with the HR management system. The code utilizes JFrame and JPanel classes from the Swing library to create the graphical windows and panels. For example, the EmployerForm class extends JFrame to create a window for adding new employers, while JPanel is used to organize and layout the components within the window. Various GUI components such as JLabel, JTextField, and JButton are employed to design the user interface. These components allow users to input data, display information, and interact with the application. For instance, text fields are used for entering employee details, labels provide descriptive text, and buttons trigger actions like adding new employers. The code utilizes layout managers to arrange GUI components within the window effectively. In the EmployerForm class, a null layout manager (setLayout(null)) is used, allowing precise positioning of components using absolute coordinates (setBounds(...)). Action listeners are attached to buttons to execute specific actions when clicked. For example, in the EmployerForm class, an action listener is added to the "Add" button to trigger the addition of a new employer when clicked. Additionally, background images, colors, button colors and text size and positioning have all been modified to provide the HR administrators with a visually good-looking system.

In addition to that, , inheritance is applied in the code as well. One clear example of inheritance can be found in Employer.java. The Employer class extends the Person class, indicating that it inherits all the attributes and methods defined in the Person class. This means that an Employer object not only possesses its specific attributes like department, jobTitle, salary, dateOfHire, age, and city, but also inherits attributes like id, name, and email from its superclass Person so that it does not need to redefine them. Additionally, another example of inheritance in our code is that the persons list in the HRDashboard class contains objects of both Person and Employer types. By treating all these objects as instances of the Person class, the code uses inheritance, as the Employer class inherits from Person. This allows the program to manage and manipulate both Person and Employer objects seamlessly.

Besides Inheritance, Polymorphism is also applied in the code. For example, in Employer.java, polymorphism is applied through method overriding. The toCSV method in the Employer class overrides the same method defined in its superclass, Person.

The superclass `Person` provides a generic implementation of `toCSV`, whereas the subclass `Employer` provides a more specific implementation to include additional attributes specific to an employer, such as `department`, `jobTitle`, `salary`, `dateOfHire`, `age`, and `city`. This demonstrates polymorphism by enabling the `toCSV` method to exhibit different behaviors based on whether it's dealing with a `Person` or `Employer`. Furthermore, in `HRDashboard.java`, polymorphism is utilized when iterating over a list of `Person` objects. The `persons` list contains objects of both `Person` and `Employer` types. When iterating through this list, the program treats each object the same as a `Person`. However, when specific methods like `toCSV` are called, the program knows to use the right method depending on whether it's dealing with `Person` or `Employer`.

Furthermore, another major and unique object-oriented programming concept that was used was Encapsulation. Encapsulation is enforced through the use of private access modifiers for class fields and methods. For example, in the `Employer` class, attributes like `name`, `email`, `department`, etc., are declared as private, restricting direct access from outside the class. Access to these attributes is provided through getter and setter methods, allowing controlled manipulation of the object's state. Additionally, the `setName` and `getName` methods in the `Employer` class provide controlled access to the `name` attribute. They use getter and setter methods to help ensure that data is accessed and modified through controlled interfaces. Another example of encapsulation is how the `HRDashboard` class interacts with `Employer` objects using public methods without needing to know how the `Employer` class is implemented internally.

File Handling

Various parts of the system, such as the `HRDashboard`, `HRManager`, and `LoginPage` classes, use file management mechanisms to facilitate operations from data storage to user authentication.

The `HRDashboard` and `HRManager` classes use file manipulation to efficiently manage employee information. The system stores the information of the employees including `ID`, `name`, `email address`, `department` and `salary` in a `.csv` file, created and updated by the program. The `loadEmployers()` method retrieves this information from the file and displays the information neatly in the system.

Similarly, the HRManager class uses file manipulation to manage HR Manager account information. The loadHRAccounts() method reads the HR account information from the hr_accounts.txt file (which is created and updated by the program) allows the system to authenticate users and grant the necessary rights.

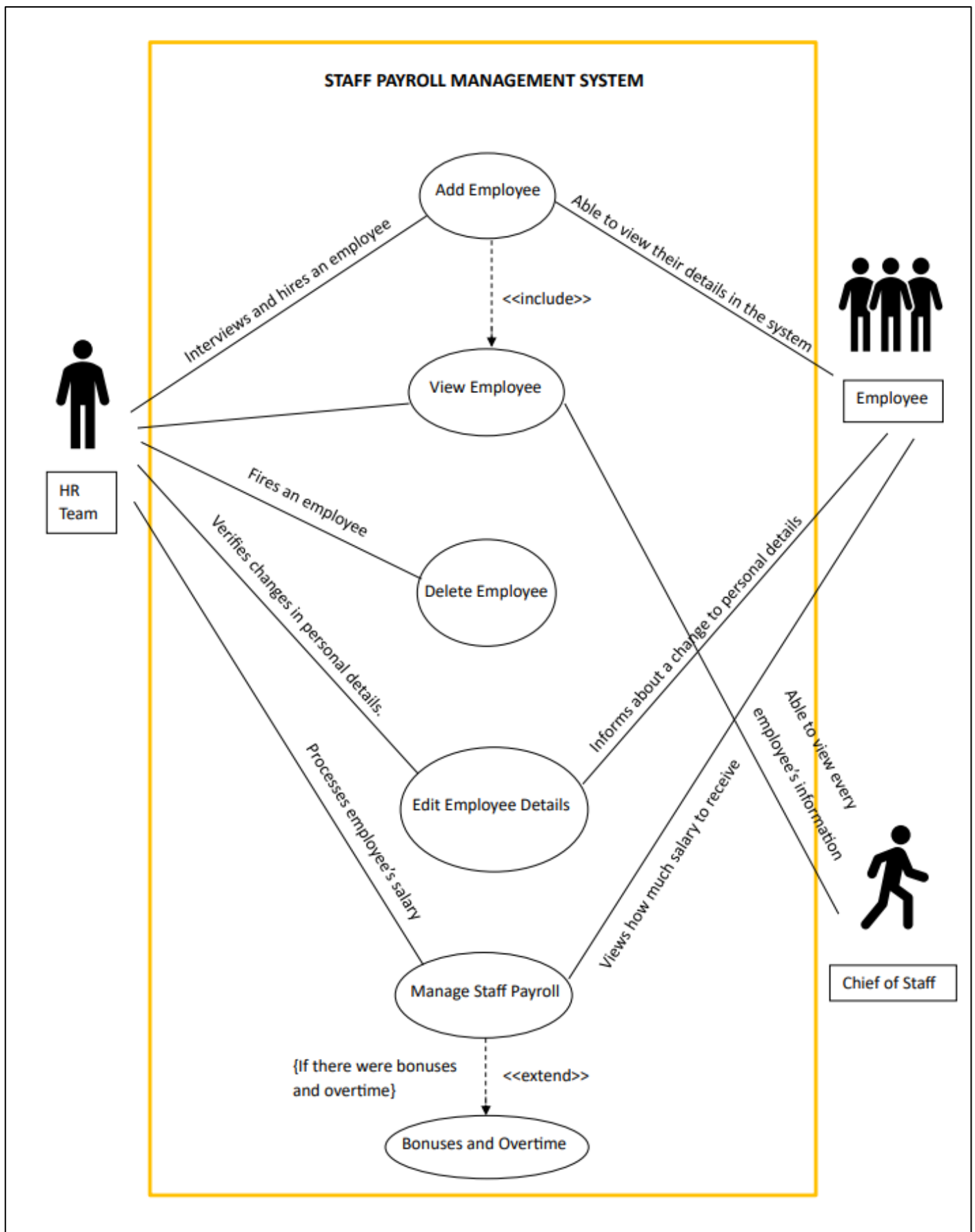
In the LoginPage class, file management plays an important role in user authentication processes. The program retrieves the HR manager's account information from the specified file and checks the authenticity of the entered credentials against existing accounts. This mechanism ensures system security by allowing authorized access and preventing unauthorized users from accessing sensitive information.

Additionally, the HRDashboard class uses file management to track payroll. The saveSalaryDetails() method writes salary details, including employee ID, month, year, and total salary, to the salary_details.txt. This payroll data repository allows the system to effectively track and manage financial transactions, increasing accountability and ease of use in the future.

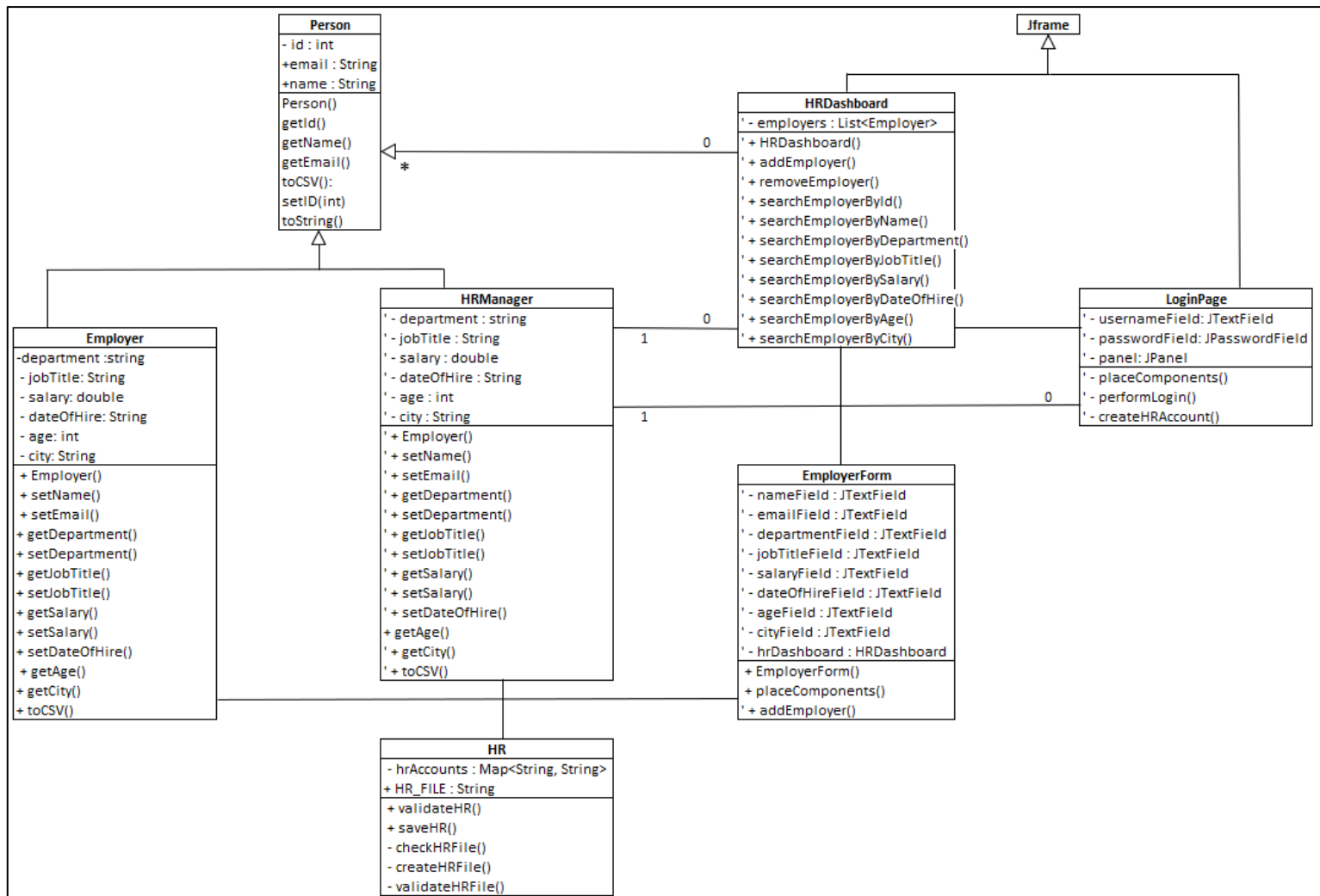
During the implementation of the file handling mechanisms, robust error handling strategies are used to handle potential problems such as file not found or I/O errors. Exception handling mechanisms are integrated into the system to handle such errors gracefully, preventing program crashes and ensuring uninterrupted operation. By providing informative feedback to users, the system improves usability and user experience while maintaining reliability and stability.

In short, it can be stated that efficient File Handling is an important part of the functionality and reliability of the employee payroll system. Through careful management of employee information, user authentication processes and payroll transactions, File Handling mechanisms ensure data integrity, security and persistence, contributing to the smooth operation and efficiency of the system.

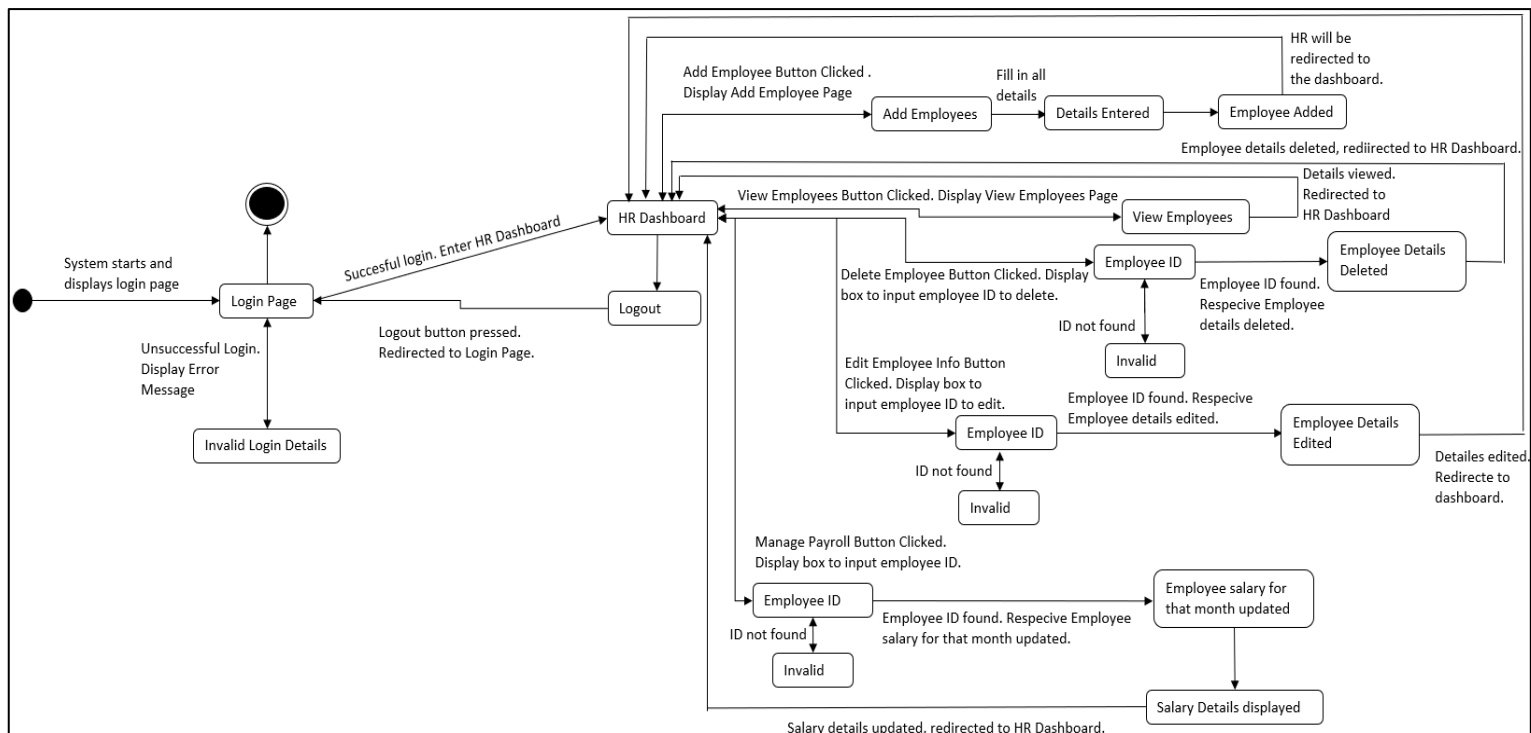
Use Case



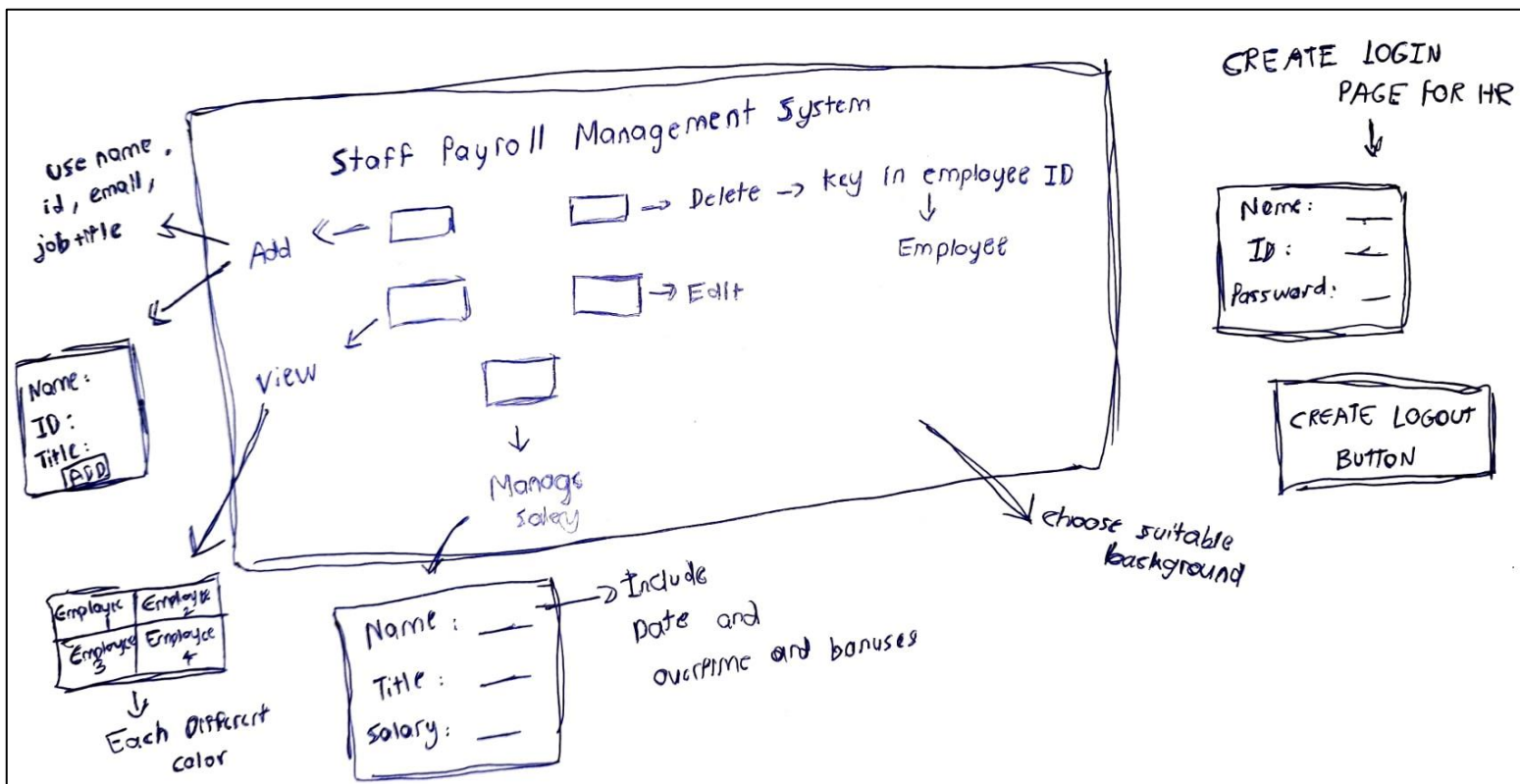
Class Diagram



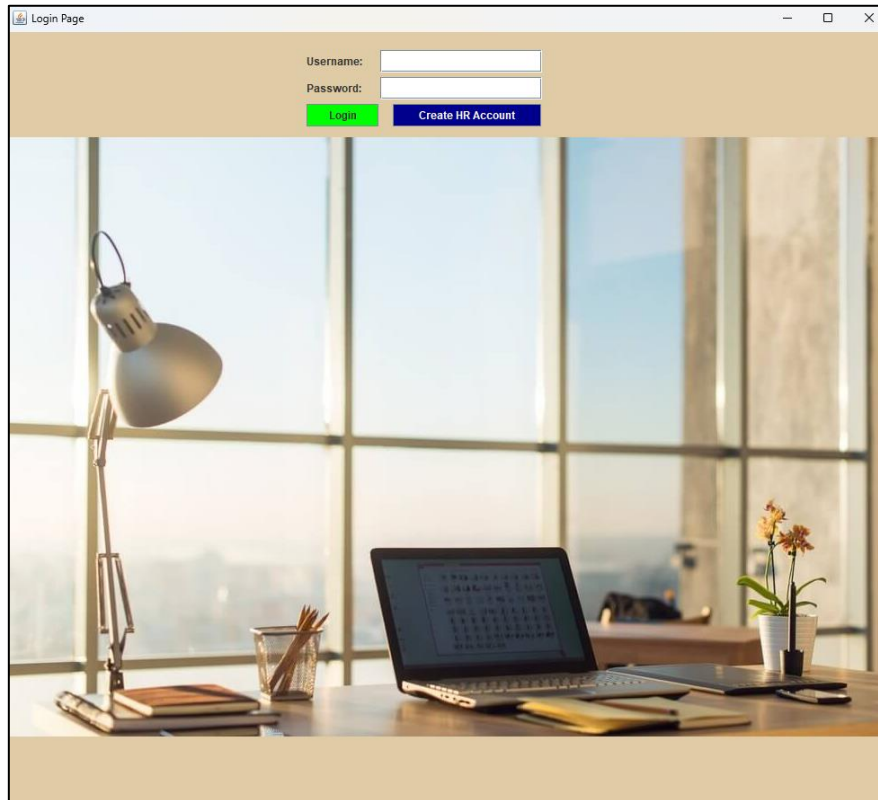
State Diagram



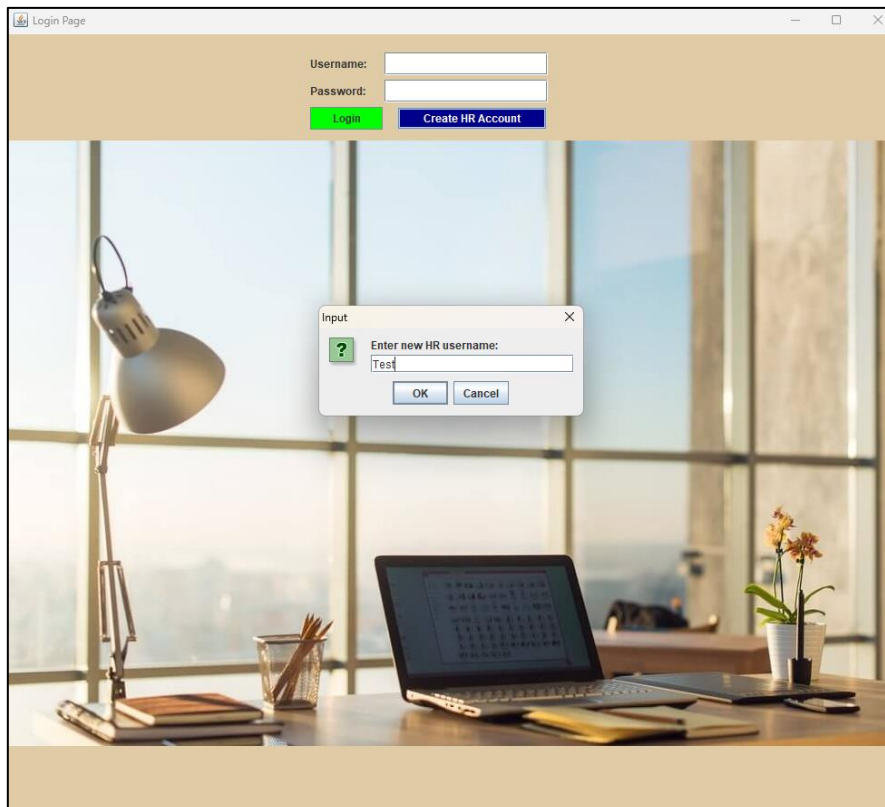
Draft Interface



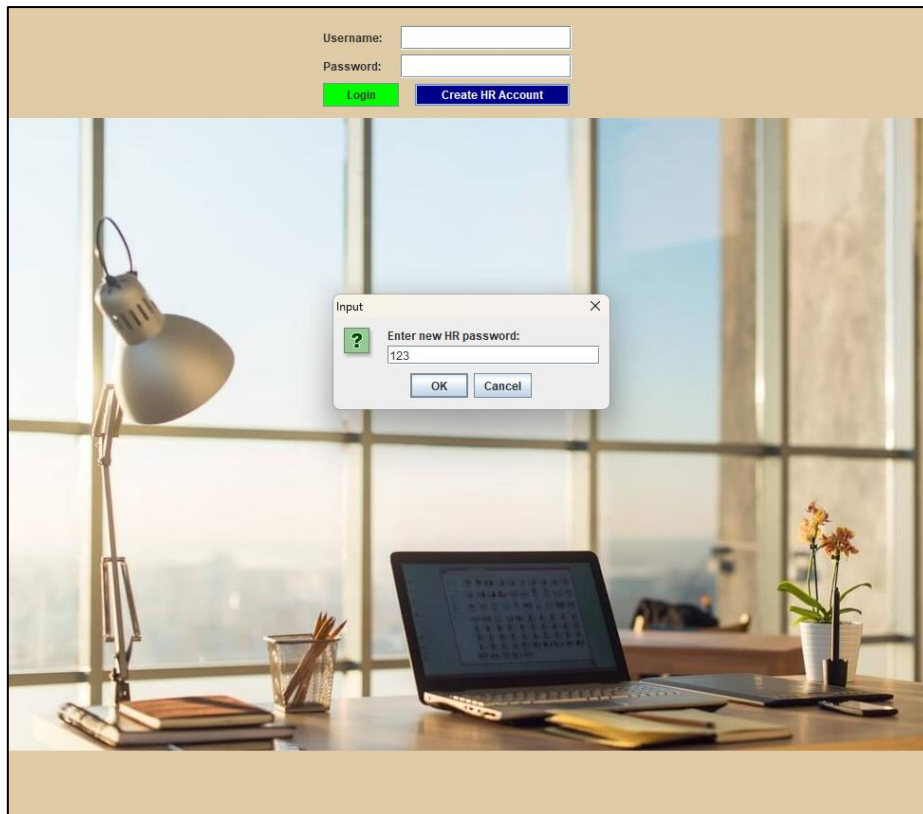
Screenshot of the program



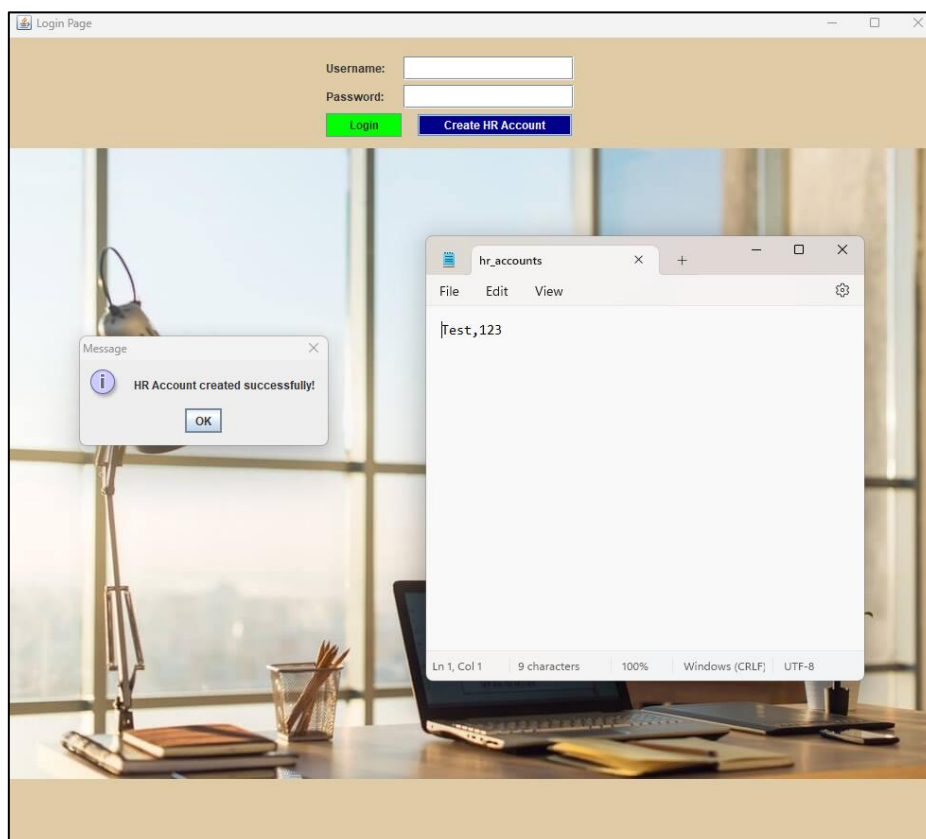
The Login Page and first page the HR staff will see when the system is accessed.



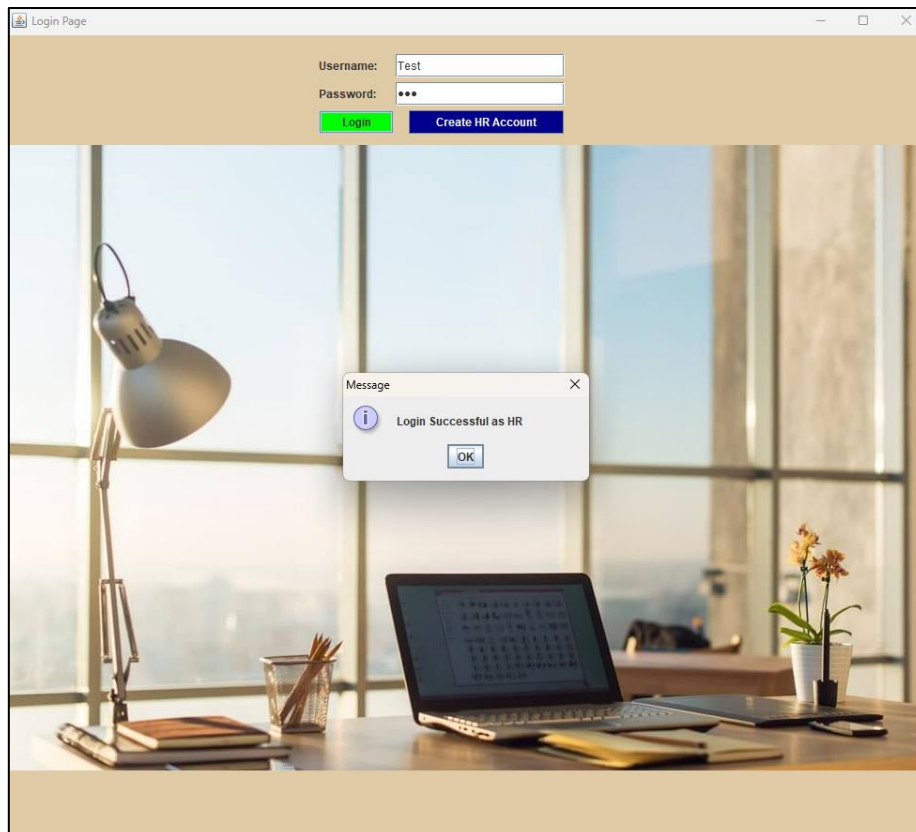
Since a HR account has not been created, it will be created first before we can log in. This pop up appears when the HR staff clicks the "Create HR Account" button.



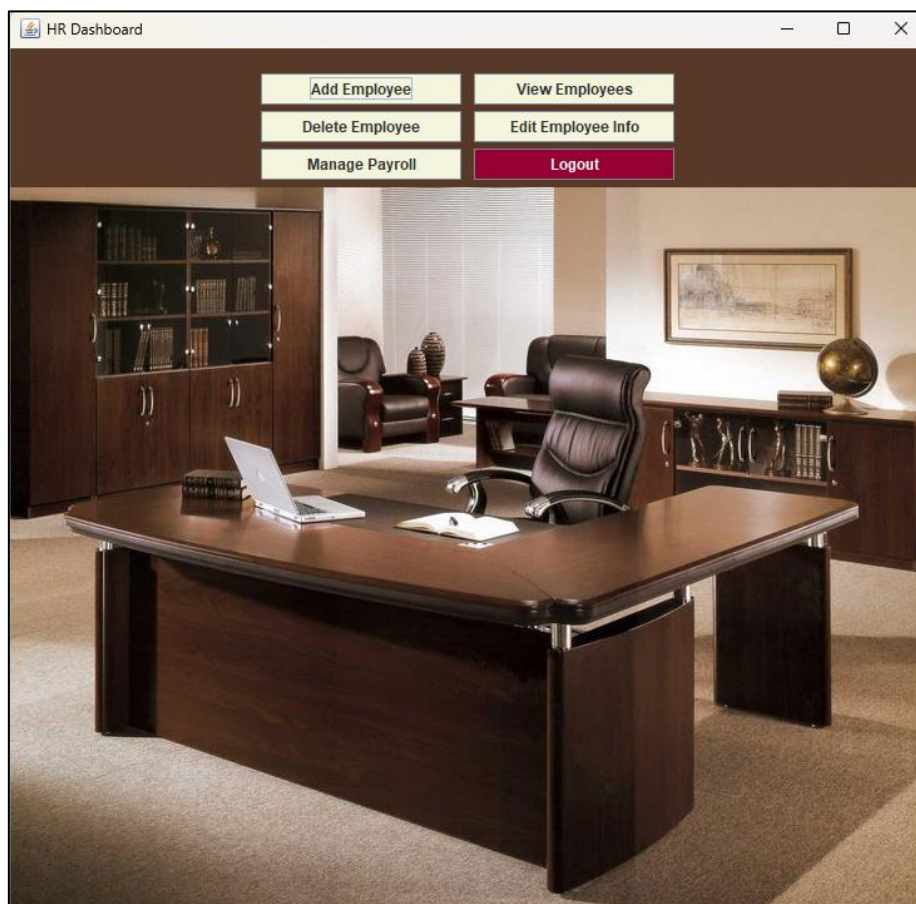
A password must be entered once a username has been entered for the new HR account.



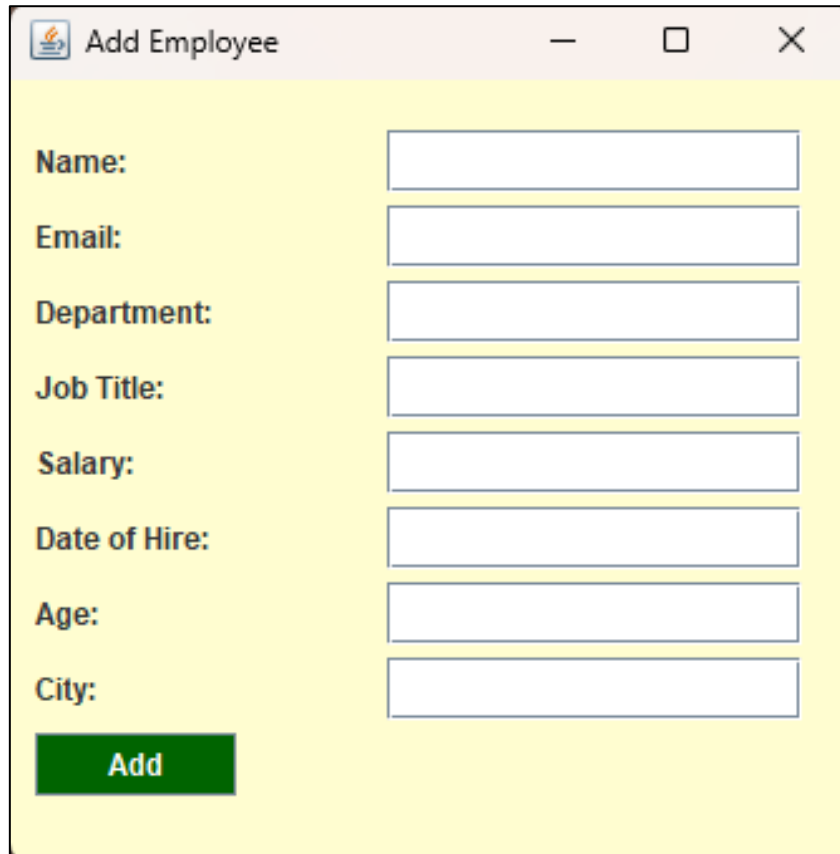
This shows that the HR account has successfully been created. A new .txt file will also be created by the program which saves the HR staff login details, as shown on the right.



The details that were used to create a new HR account will now be used to log in. This shows that the login has been successful.



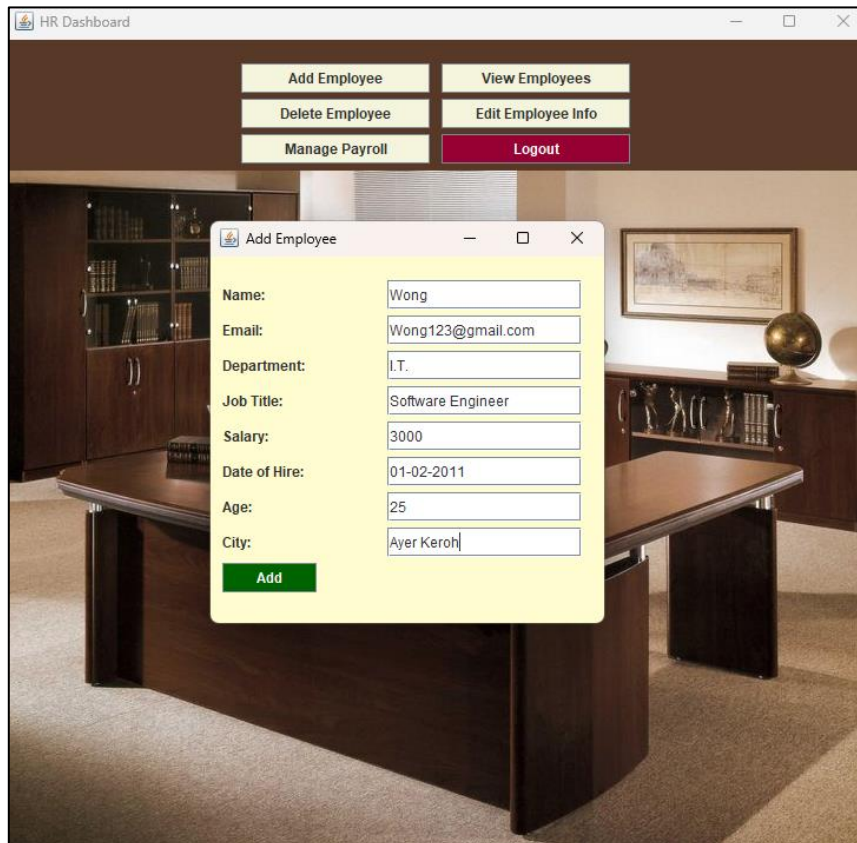
Once logged in, the HR staff will be brought to this screen, the HR Dashboard screen.



The image shows a window titled "Add Employee" with a yellow background. It contains eight input fields for employee details: Name, Email, Department, Job Title, Salary, Date of Hire, Age, and City. Each field is a white rectangle with a thin blue border. Below the fields is a green button with the text "Add" in white. The window has a standard title bar with a minimize button, a maximize button, and a close button.

| | |
|---------------|----------------------|
| Name: | <input type="text"/> |
| Email: | <input type="text"/> |
| Department: | <input type="text"/> |
| Job Title: | <input type="text"/> |
| Salary: | <input type="text"/> |
| Date of Hire: | <input type="text"/> |
| Age: | <input type="text"/> |
| City: | <input type="text"/> |

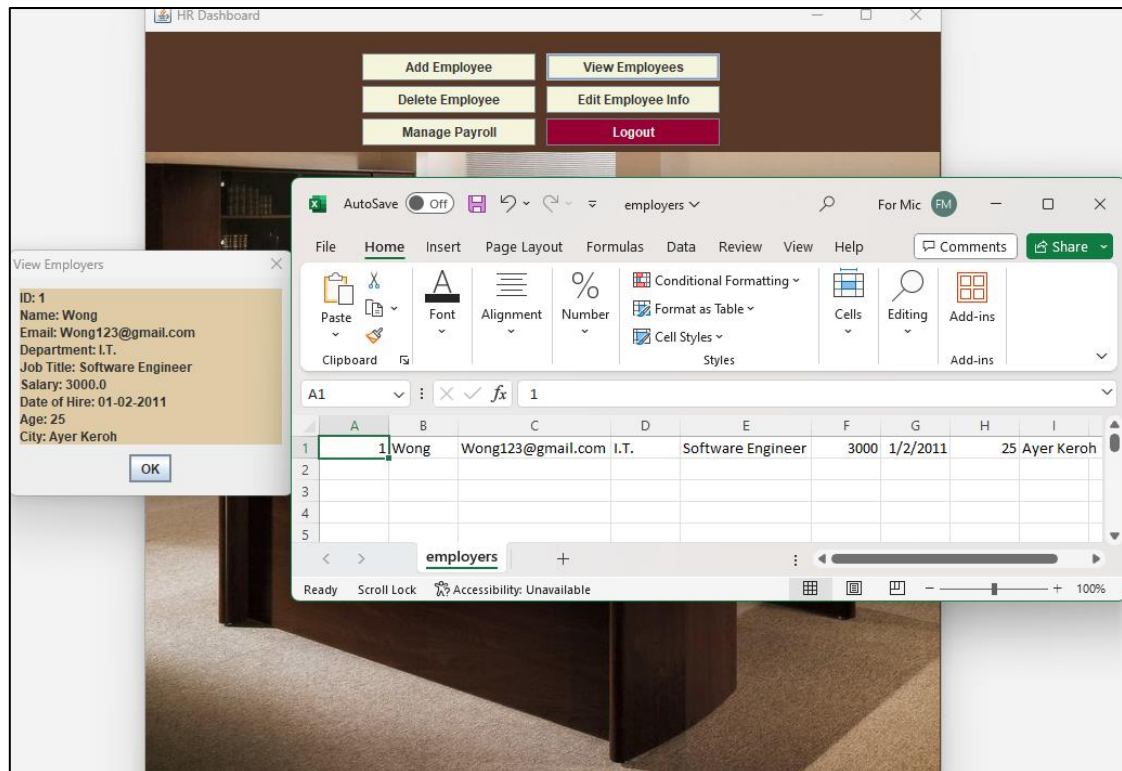
This screen appears when the “Add Employee” button is pressed. This is where the HR staff can add in details of a new employee.



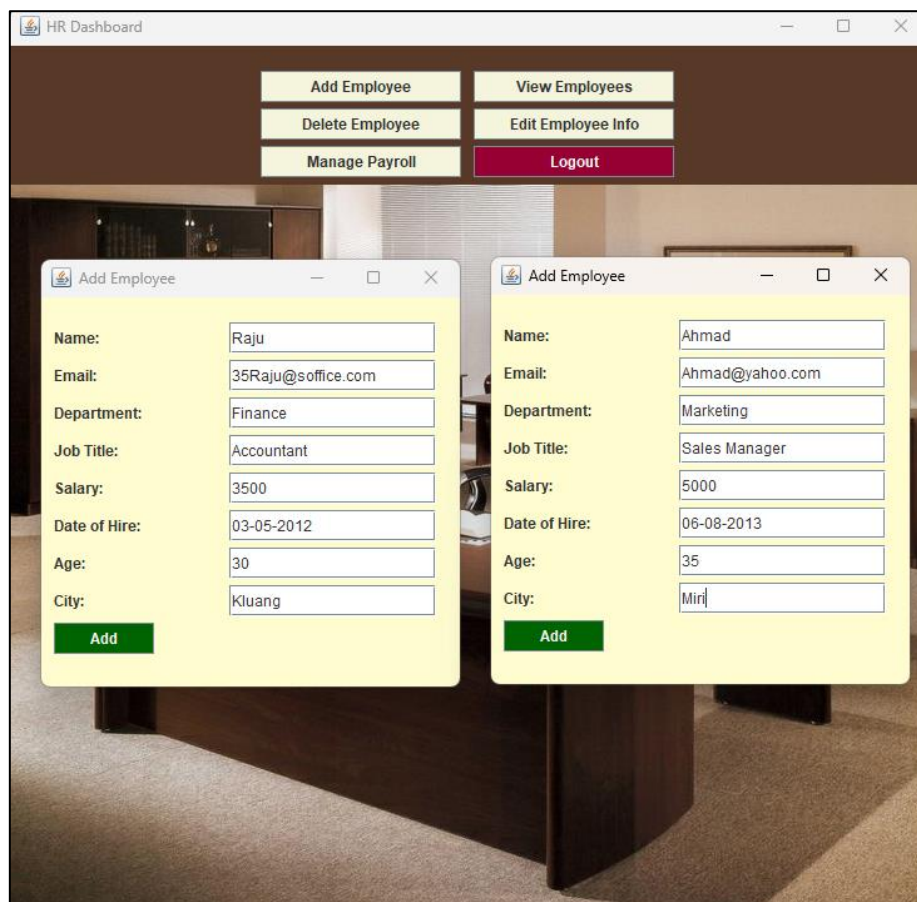
The image shows a window titled "HR Dashboard" with a brown header. It contains six buttons: "Add Employee", "View Employees", "Delete Employee", "Edit Employee Info", "Manage Payroll", and "Logout". The "Add Employee" button is highlighted. Below the buttons is a background image of an office desk. Overlaid on the desk is the "Add Employee" form window, which is the same as the one in the previous image, but with the following data entered:

| | |
|---------------|-------------------|
| Name: | Wong |
| Email: | Wong123@gmail.com |
| Department: | I.T. |
| Job Title: | Software Engineer |
| Salary: | 3000 |
| Date of Hire: | 01-02-2011 |
| Age: | 25 |
| City: | Ayer Keroh |

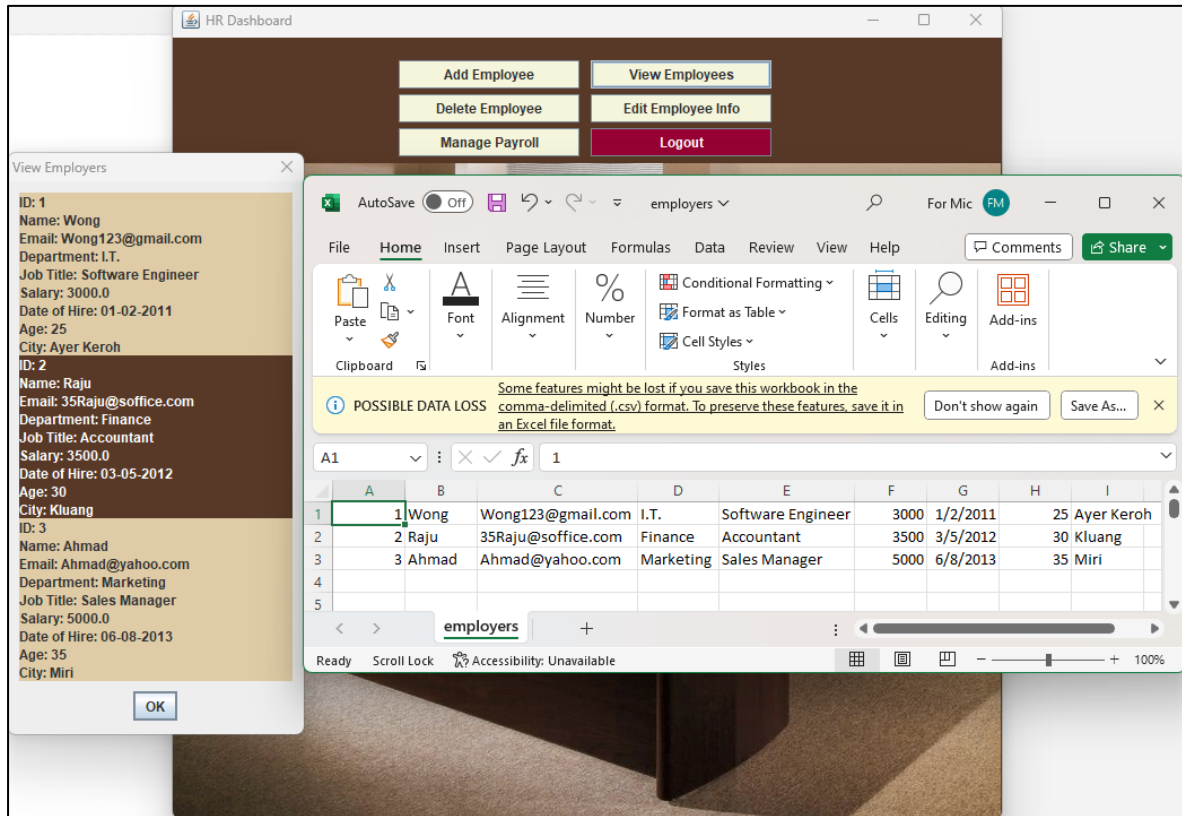
To test the ‘Add Employee’ functionality, details of a new employee has been added.



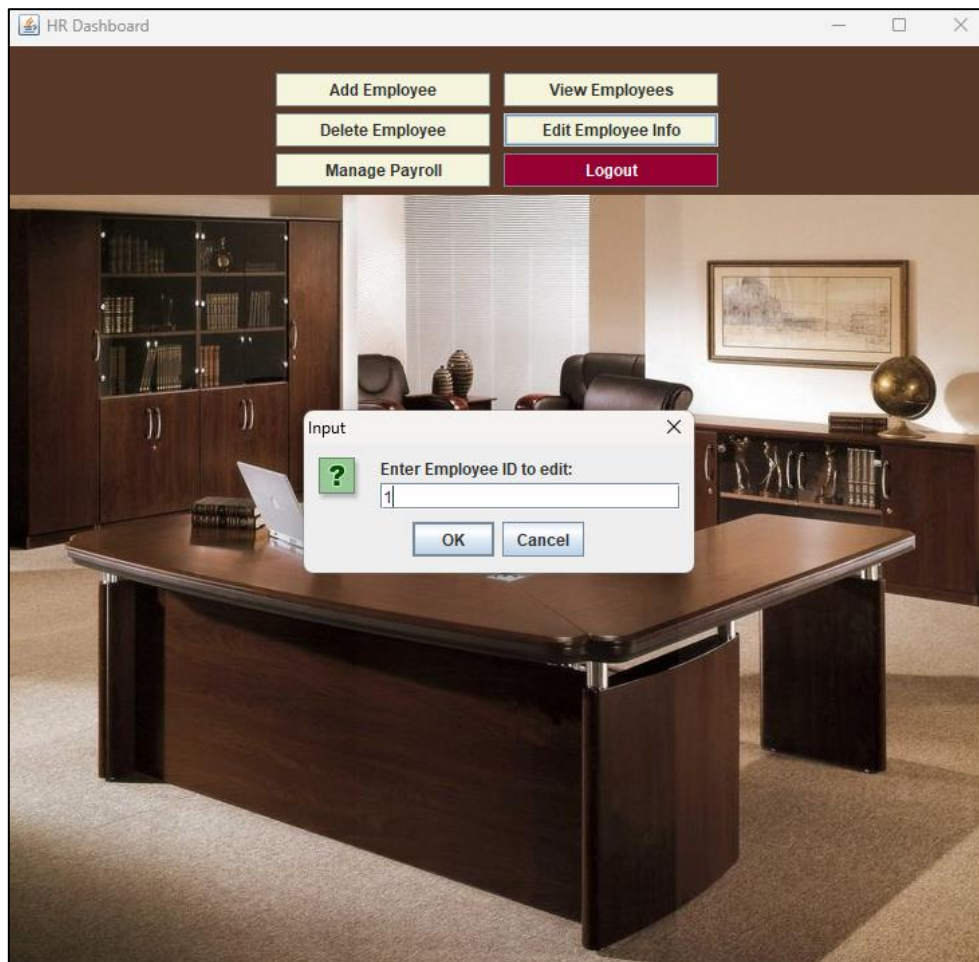
Once a new employee has been added, it can be viewed by pressing the 'View Employee' button which pops up another screen, as shown on the left. The details of the newly added employee will also be saved into a .csv file created by the program. The details are same.



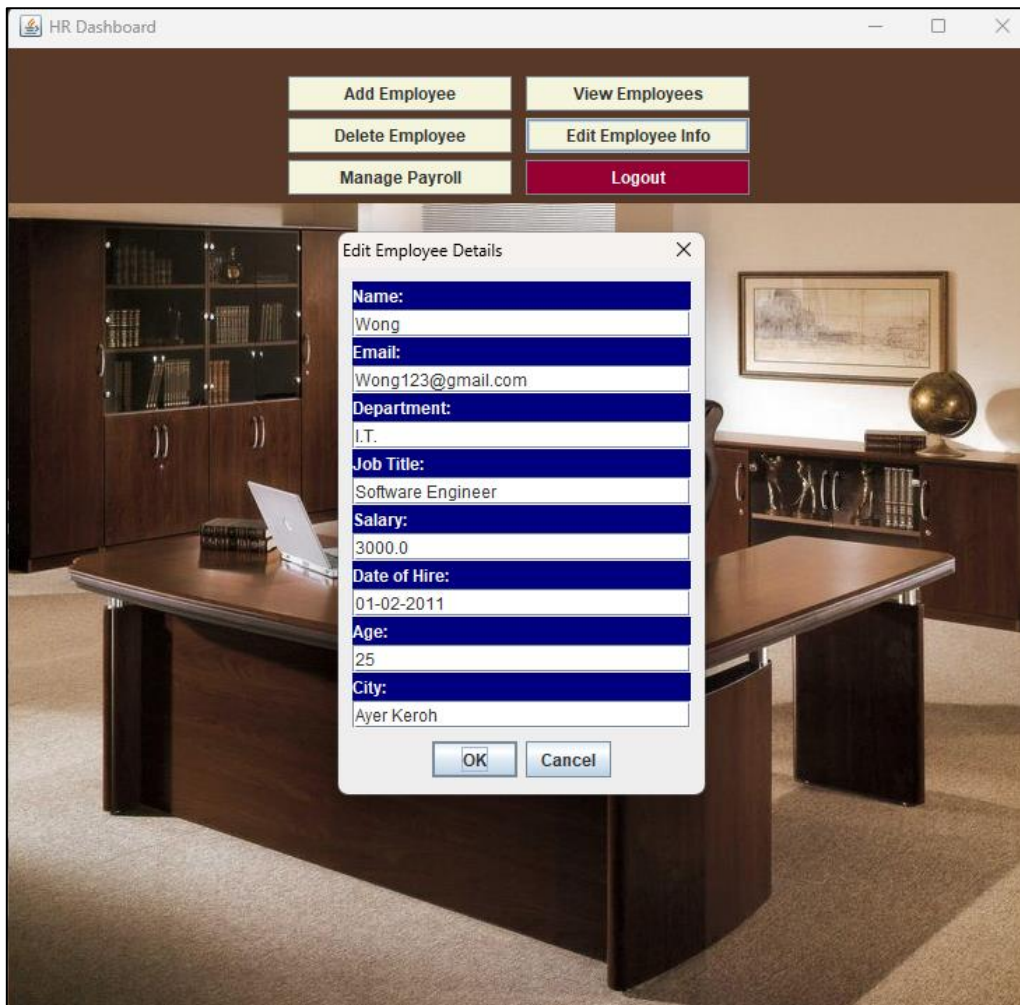
To test the other functionalities, two new employees have been added.



As a verification, these are the details of all three added employees in the 'View Employee' screen (left) and the .csv file which is created and updated by the program (right).



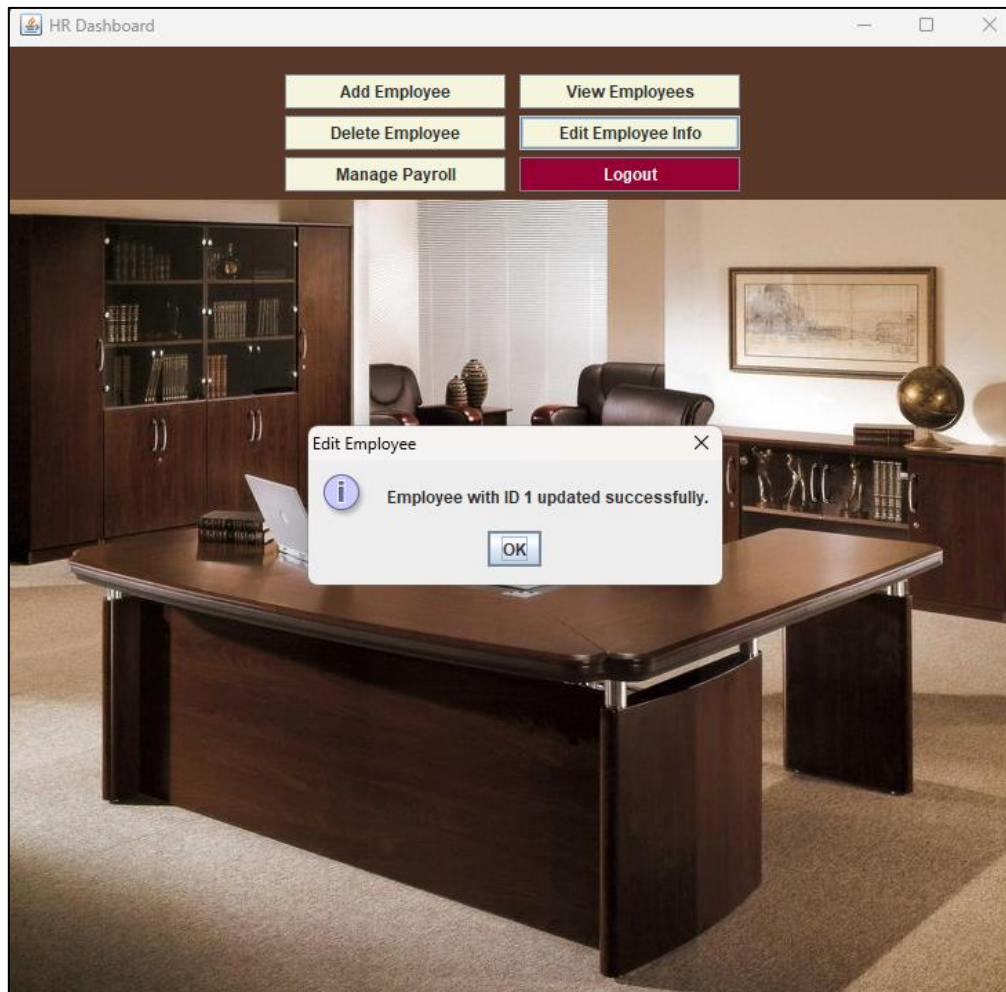
This appears after the 'Edit Employee Info' button is pressed.



Once the ID of the employee to be edited has been found, it then displays the details of the employee that were added. Any and as many of those details can be edited here.

This is a close-up of the 'Edit Employee Details' dialog box. The 'Salary' field, which previously contained '3000.0', now has '4000.0' entered. The cursor is positioned at the end of the new value. All other fields remain unchanged: Name: Wong, Email: Wong123@gmail.com, Department: I.T., Job Title: Software Engineer, Date of Hire: 01-02-2011, Age: 25, City: Ayer Keroh. The 'OK' and 'Cancel' buttons are at the bottom.

To demonstrate this functionality, we changed the salary from 3,000 to 4,000 as shown.

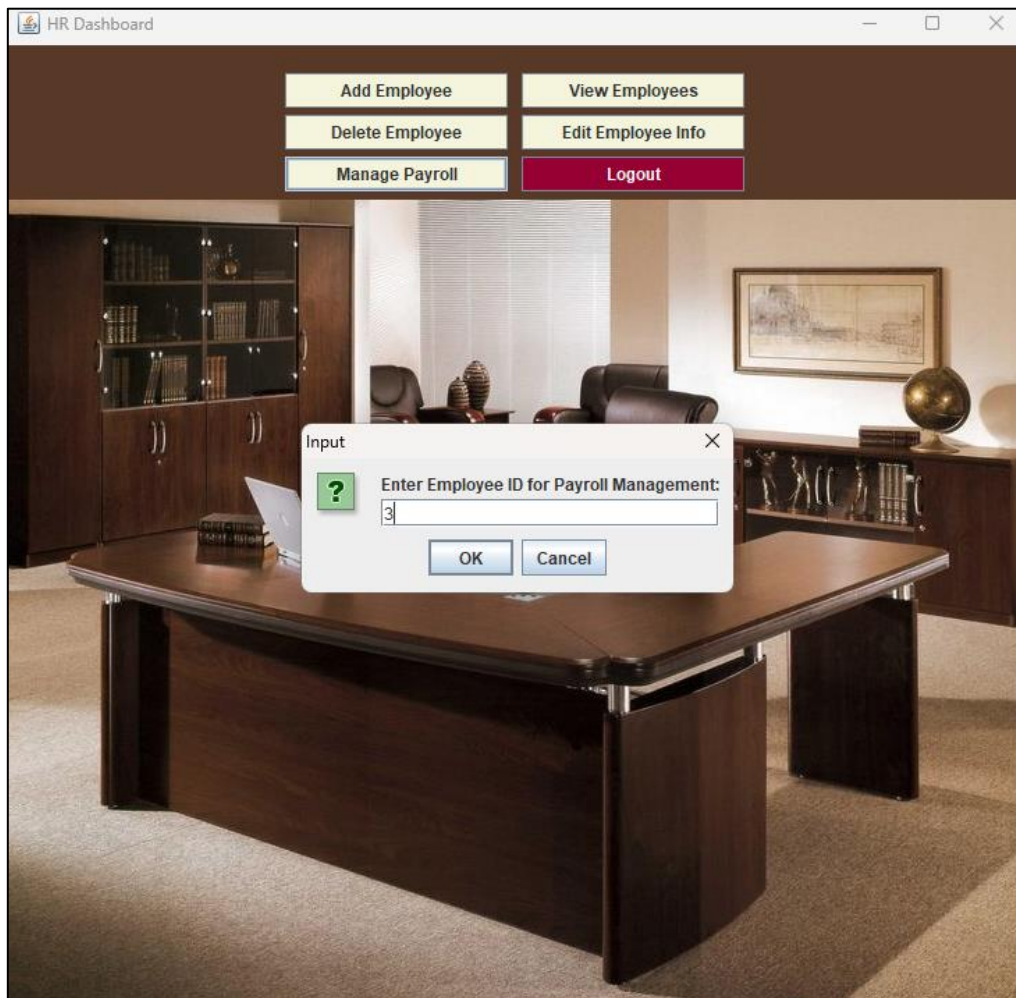


Once the 'OK' button earlier was pressed, a confirmation message is displayed.

The screenshot shows the 'View Employers' application window. On the left, there is a list of employee details for three employees. On the right, there is a Microsoft Excel spreadsheet titled 'employers' showing the same data. The salary for employee Wong (ID 1) is highlighted in green in the Excel spreadsheet.

| ID | Name | Email | Department | Job Title | Salary | Date of Hire | Age | City |
|----|-------|--------------------|------------|-------------------|--------|--------------|-----|------------|
| 1 | Wong | Wong123@gmail.com | I.T. | Software Engineer | 4000.0 | 01-02-2011 | 25 | Ayer Keroh |
| 2 | Raju | 35Raju@soffice.com | Finance | Accountant | 3500.0 | 03-05-2012 | 30 | Kluang |
| 3 | Ahmad | Ahmad@yahoo.com | Marketing | Sales Manager | 5000.0 | 06-08-2013 | 35 | Miri |

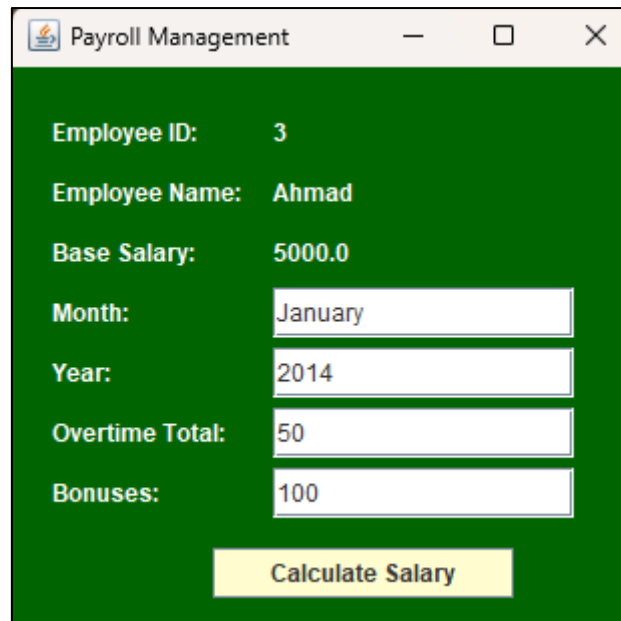
The salary for the employee named Wong will also be updated elsewhere in the system.



Once the 'Manage Payroll' button has been clicked, this prompt will be brought up. To test the functionality of the system's payroll, we will test it with employee with the ID 3.

The image shows a screenshot of a web application titled "Payroll Management". The screen has a green background. It displays the following information: "Employee ID: 3", "Employee Name: Ahmad", "Base Salary: 5000.0", "Month: [input field]", "Year: [input field]", "Overtime Total: [input field]", and "Bonuses: [input field]". At the bottom of the screen is a yellow button labeled "Calculate Salary".

Once the employee ID has been found, the Payroll Management screen is brought up. In this screen, the basic details of the respective employee (their name, ID number and base salary) **cannot** be changed as it is only there for reference purposes.

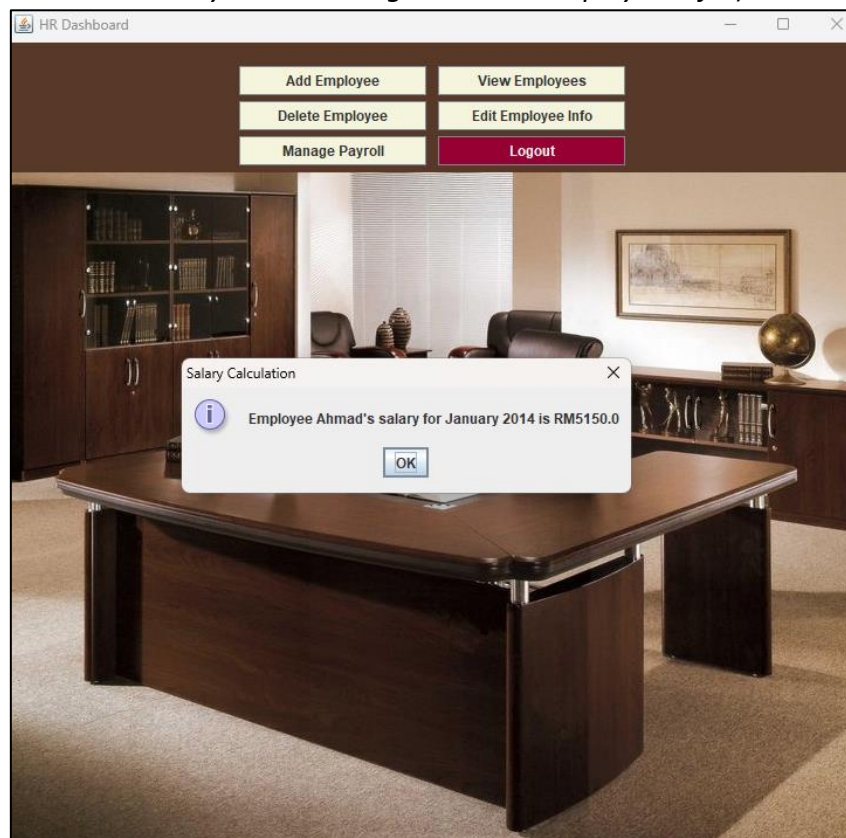


The screenshot shows a window titled "Payroll Management" with a green background. It contains the following fields and values:

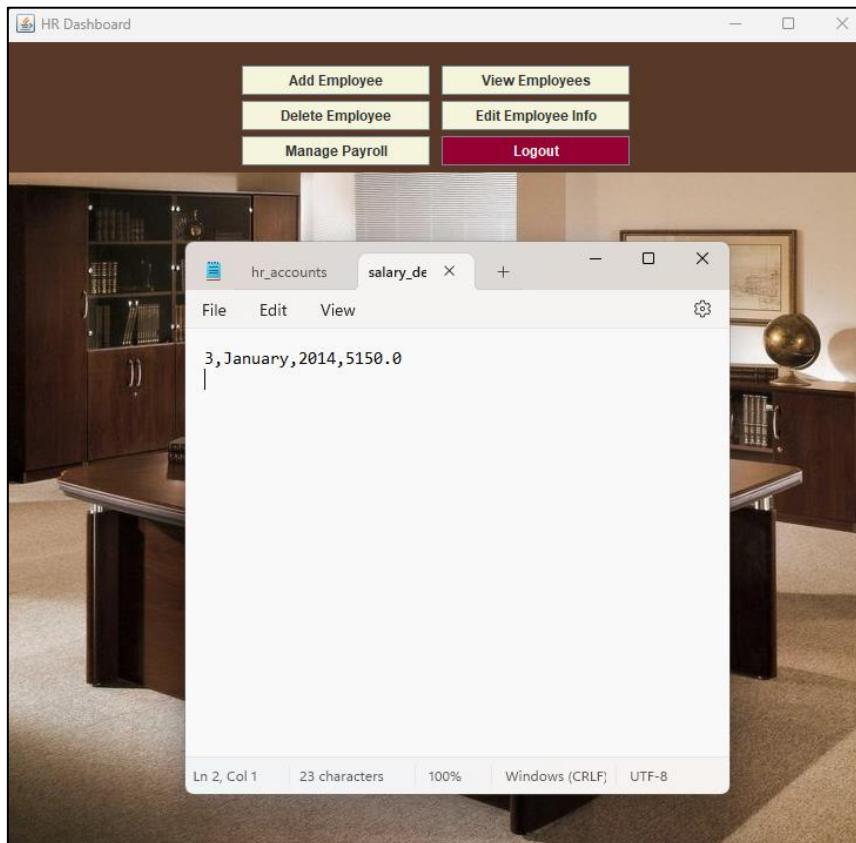
| Field | Value |
|-----------------|---------|
| Employee ID: | 3 |
| Employee Name: | Ahmad |
| Base Salary: | 5000.0 |
| Month: | January |
| Year: | 2014 |
| Overtime Total: | 50 |
| Bonuses: | 100 |

At the bottom, there is a yellow button labeled "Calculate Salary".

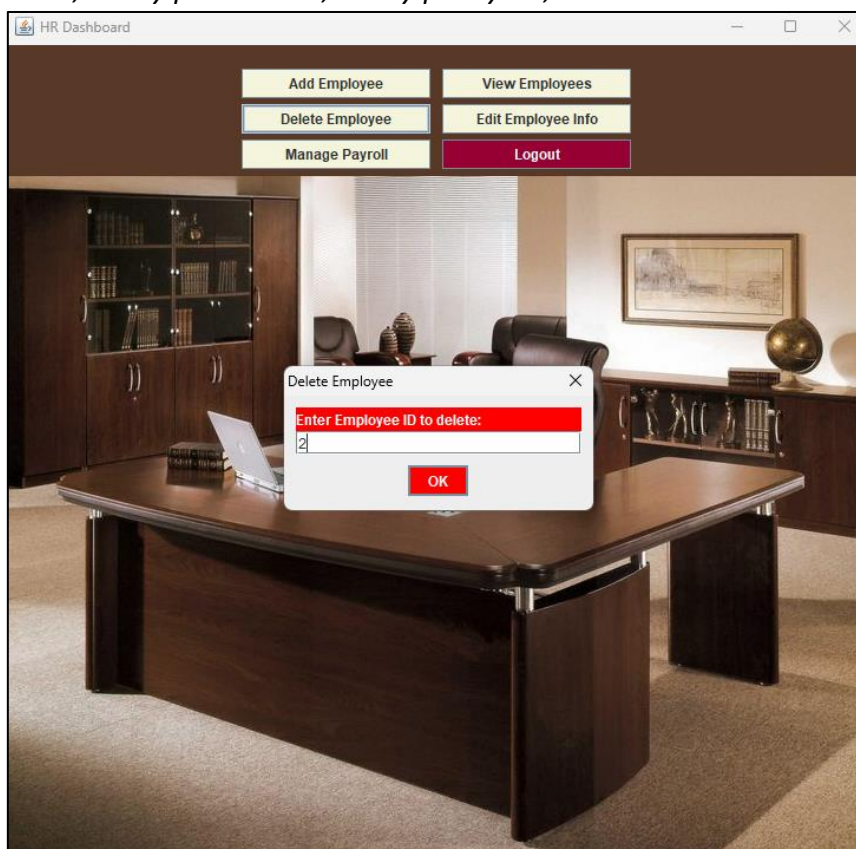
The month and year text fields refer to the respective month and year that the employee is to be paid. For example, the HR Staff is currently in the process of paying Ahmad's salary for January 2014. In that respective month, Ahmad had also earned 50 RM through overtime and 100 RM through bonuses, both of which have been calculated and processed externally by the HR. (*Note: Base salary can be changed at 'Edit Employee Info')



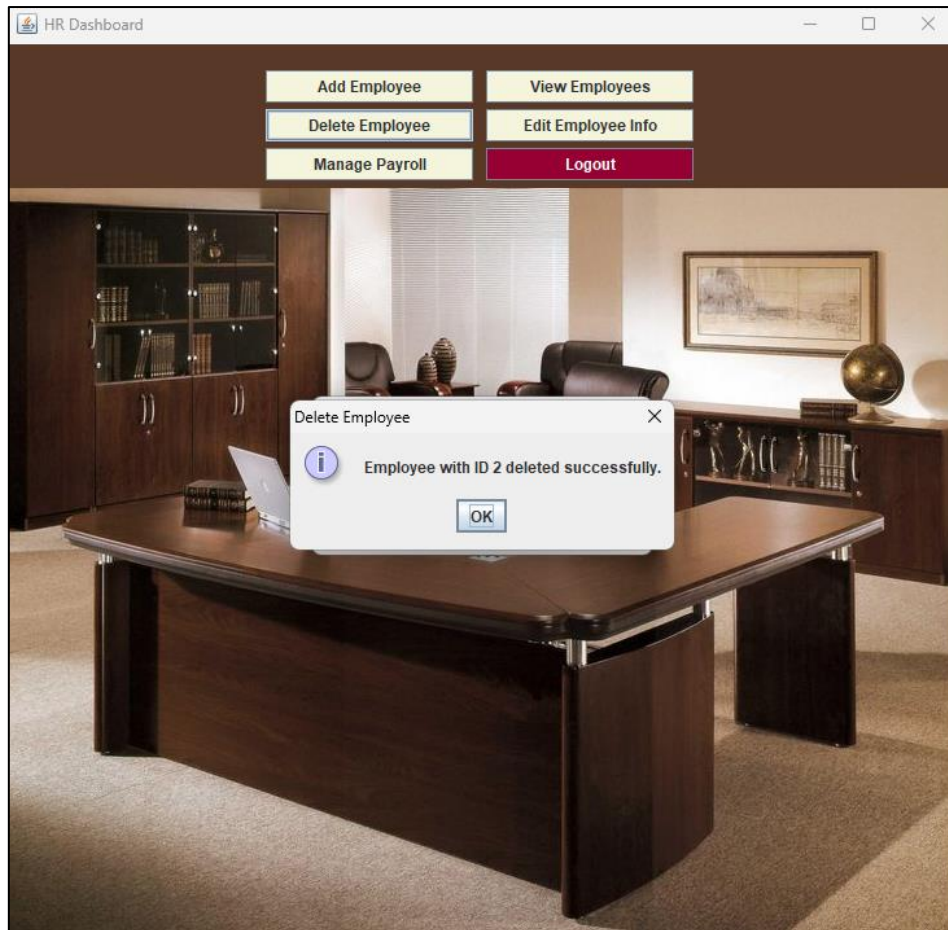
Once the 'Calculate Salary' button had been pressed, the program will calculate the respective employee's salary for that respective month and year, in addition to the base salary. It will then display a message stating that the salary for that respective employee in that year and month is X amount.



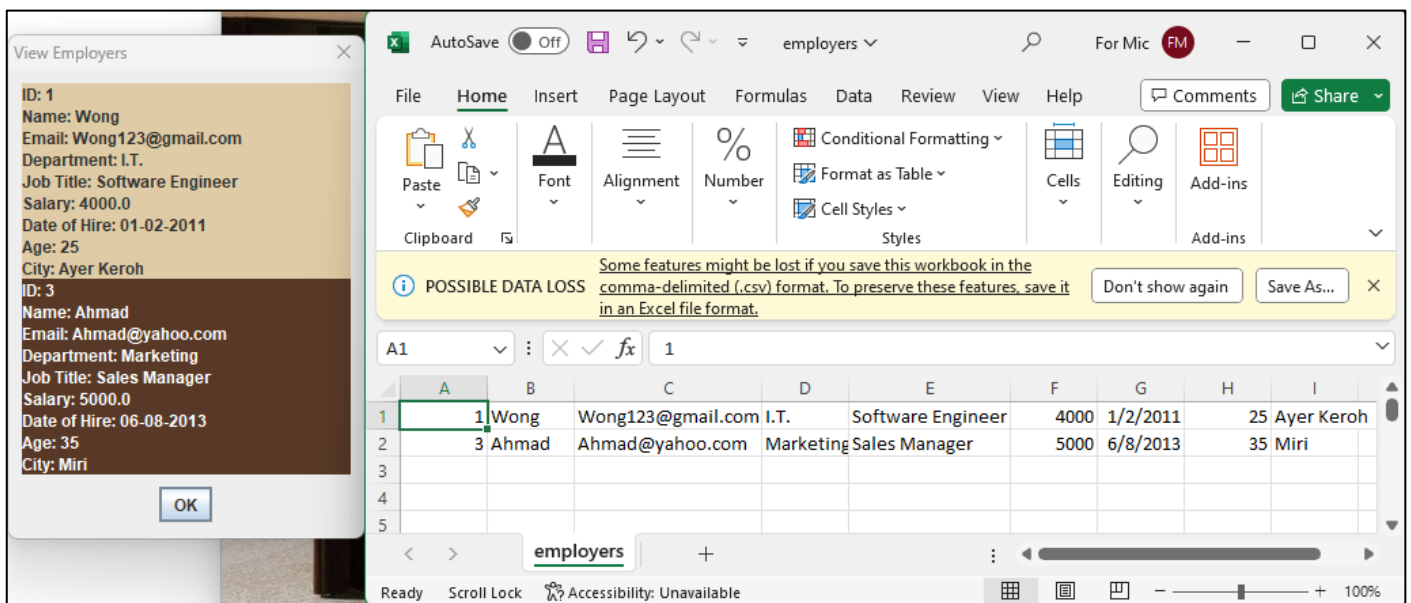
A new .txt file will be created by the program that saves the payroll details. **From left to right: Employee ID, salary paid month, salary paid year, amount.*



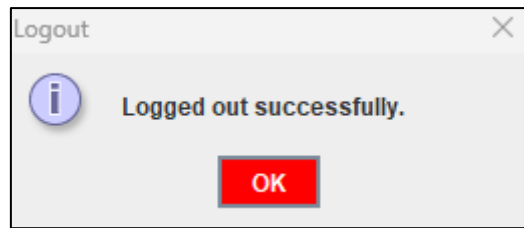
Once the 'Delete Employee' button has been pressed, this screen appears. To test this functionality, we will use employee with ID number 2



Once the 'OK' button had been pressed, and if the employee with the stated ID had been found, the system will display this message.



Once the employee detail has been deleted, it will also be deleted in other parts of the system, as shown.



Once the user presses the 'logout' button, this will be prompted, before the system is logged out.