LAB\_3:

Section A: Create a JavaFX Application for Simple HR Management (150 Marks)

Create a new JavaFX application project in your preferred Java IDE (for example, IntelliJ IDEA).

Name the project as "HR Management Your Name."

A screenshot of a computer program

AI-generated content may be incorrect.

***I kept my project name as \_lab3***

***Section B: Create Four GUI Pages for Login, Dashboard, Admin, and Employee (150 Marks)***

* + 1. ***Login Page:***
       - 1. ***Email and Password Fields***
         2. ***Login button***
    2. ***Dashboard Page:***
       - 1. ***Four buttons: Admin, Employee, Logout, Exit***
    3. ***Admin Page:***
       - 1. ***Table View***
         2. ***Buttons for Create, Update, Delete, View, and Back***
    4. ***Employee Page:***
       - 1. ***Table View***
         2. ***Buttons for Create, Update, Delete, View, and Back***

***1. A screenshot of a computer login screen

AI-generated content may be incorrect.***

***2. A screenshot of a computer

AI-generated content may be incorrect.***

***3. A screenshot of a computer

AI-generated content may be incorrect.***

***4. A screenshot of a computer

AI-generated content may be incorrect.***

***Section C: Create Database Tables (150 Marks)***

1. ***In your MySQL database, create tables for storing employee data, such as "Employee Detail," Admin, and "Salary."***
2. ***Define the table structures to accommodate employee information and salary details.***
3. ***Define the table structures to accommodate Admin information and salary details.***

***1) Creating a database SQL commands***

***CREATE DATABASE IF NOT EXISTS hr\_management;***

***USE hr\_management;***

***-- Admin Table***

***CREATE TABLE IF NOT EXISTS Admin (***

***id INT AUTO\_INCREMENT PRIMARY KEY,***

***name VARCHAR(100) NOT NULL,***

***email VARCHAR(100) UNIQUE NOT NULL,***

***password VARCHAR(100) NOT NULL***

***);***

***-- Employee Table***

***CREATE TABLE IF NOT EXISTS Employee (***

***id INT AUTO\_INCREMENT PRIMARY KEY,***

***name VARCHAR(100) NOT NULL,***

***position VARCHAR(100),***

***salary DOUBLE***

***);***

***-- Salary Table (Optional for future extensions like JUnit tests)***

***CREATE TABLE IF NOT EXISTS Salary (***

***employee\_id INT PRIMARY KEY,***

***base\_salary DOUBLE,***

***bonus DOUBLE,***

***FOREIGN KEY (employee\_id) REFERENCES Employee(id)***

***);***

***-- Sample Admin User for Testing Login***

***INSERT INTO Admin (name, email, password) VALUES ('Admin User', 'admin@example.com', 'password123');***

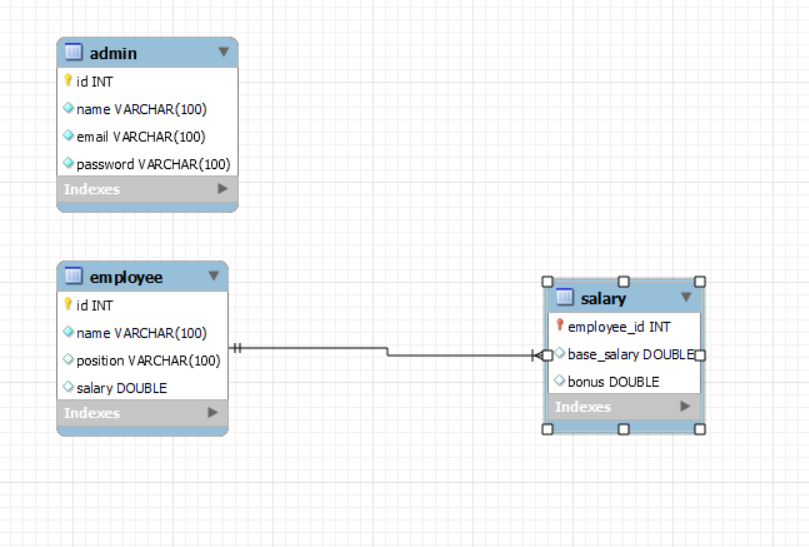
***A screenshot of a computer code

AI-generated content may be incorrect.***

***A screenshot of a computer program

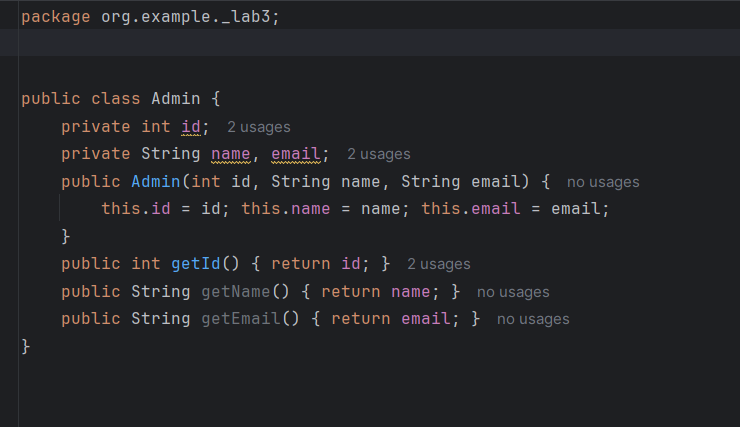
AI-generated content may be incorrect.***

***EER diagram for relationship between employee and salary***

***Section D: GitHub and Documentation (360 Marks)***

1. ***Create Java classes representing the structure of the database tables (for example, Employee and Admin).***
2. ***These classes will be used to model the data retrieved from and inserted into the database.***

***Admin.java***

******

***Employee.java***

***A screen shot of a computer code

AI-generated content may be incorrect.***

***Section E: Write Database Connection Code (100 Marks)***

1. ***Implement the database connection code in your JavaFX application.***
2. ***Ensure it includes the necessary database URL, username, and password.***
3. ***Take a screenshot of the code where you establish the database connection.***

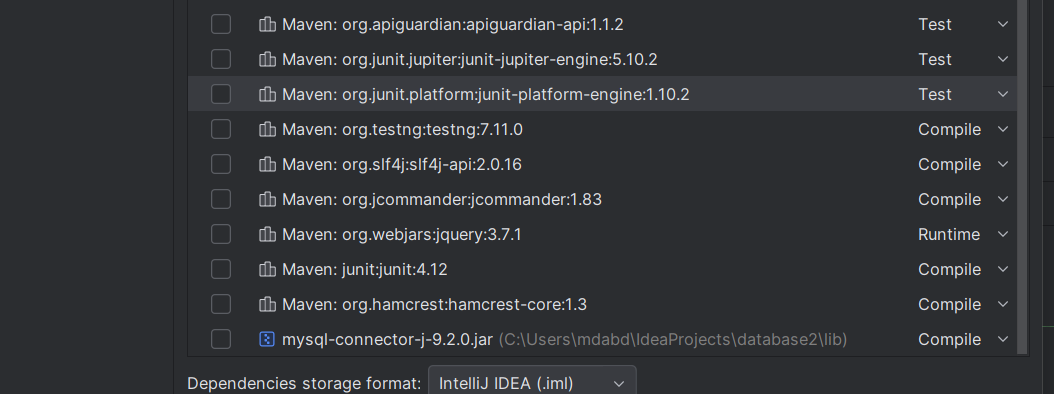
***A computer screen shot of a program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.***

***Section F: Use Open JDBC Jar to Connect Database (50 Marks)***

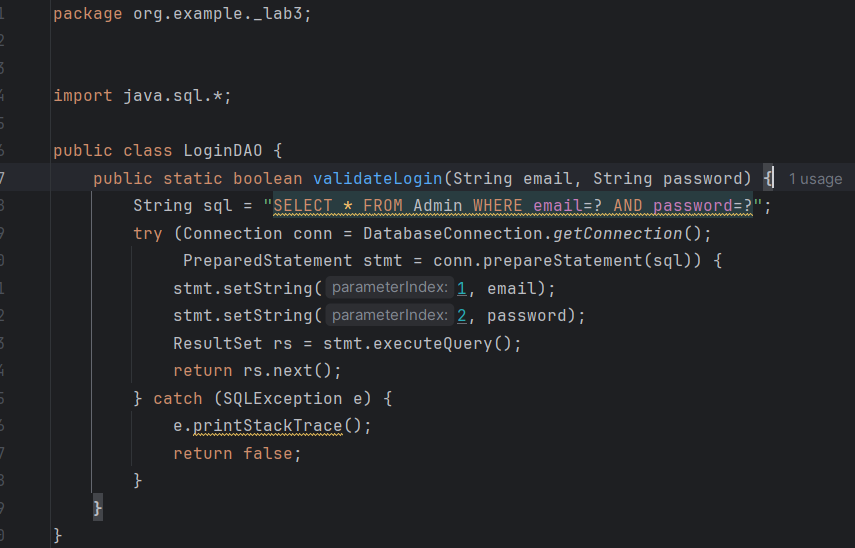
1. ***Download and include the appropriate JDBC driver (for example, MySQL Connector/J) in your project.***
2. ***Ensure your project's build path includes the JDBC driver JAR file.***

******

***Section G: Create DAO Classes and Use ORM (100 Marks)***

1. ***Create Java models that map to the database tables (for example, Employee DAO, Admin DAO, and Login DAO).***
2. ***Use Object-Relational Mapping (ORM) techniques to simplify database interactions.***

***Login DAO***

******

***AdminDAO***

***A computer screen shot of a program

AI-generated content may be incorrect.A computer screen shot of a program code

AI-generated content may be incorrect.A screen shot of a computer code

AI-generated content may be incorrect.***

***EmployeeDAO***

***A screenshot of a computer program

AI-generated content may be incorrect.***

***A computer screen shot of a program code

AI-generated content may be incorrect.***

***A screen shot of a computer code

AI-generated content may be incorrect.Section H: Implement the Login Module (75 Marks)***

1. ***Implement a login module to restrict access to HR data.***
2. ***Users should log in using a username and password.***

***A screenshot of a computer error message

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.***

***Section I: Implement the Dashboard Module (100 Marks)***

1. ***The dashboard page will have four buttons: Admin, Employee, Logout, Exit.***
2. ***The Logout button will take the user to the Login page.***
3. ***The Exit button will exit the application.***
4. ***Display a welcome message, "Welcome, Username," and the current date.***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Section J: Implement the Admin Module (125 Marks)***

1. ***Create, update, delete, and view Admin records.***
2. ***Include functioning Logout and Back buttons.***

***Create:***

***A screenshot of a computer screen

AI-generated content may be incorrect.***

***Update***

***A screenshot of a computer screen

AI-generated content may be incorrect.***

***Delete***

***\ A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.***

***BACK***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Section K: Implement the Employee Module (125 Marks)***

1. ***Create, Update, Delete, and View Employee Records.***
2. ***Include functioning Logout and Back buttons***

***Create:***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Update***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Delete***

***A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.***

***BACK***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Section L: Implement JUnit Testing for Calculating Yearly Salary (300 Marks)***

1. ***Write JUnit test cases to calculate yearly salaries for employees based on the provided data.***
2. ***Ensure the tests cover various scenarios, including different employee roles and salary structures.***

***A screenshot of a computer program

AI-generated content may be incorrect.***

***Section M: GitHub Repository and Documentation (125 Marks)***

* + - 1. ***Upload your JavaFX HR Management project to a GitHub repository.***
      2. ***Create a well-documented README file in your repository, providing instructions on setting up and running your application, including login details.***
      3. ***Submit a DOCX or PDF document that includes:***

***Screenshots of your GUI layout***

***Screenshots of your database table structures***

***Screenshots of relevant portions of your code***

***Screenshots of your JUnit test cases and results***

***A link to your GitHub repository***

***A screenshot of a computer

AI-generated content may be incorrect.***

***A screenshot of a computer

AI-generated content may be incorrect.***