

ITECH 7201

Assignment 1

Part B - Group Task

Human Resource Management System

Employee Management System

Leave Management System

Attendance Management System

Details of Team members

Student ID	Name	Sub system responsible for	Contribution for Part B	Color code
30432670	Syeda Tamanna Sheme	Employee Management System	33.34%	
30437681	Tanvir Iqbal	Leave management System	33.34%	
30423990	Hansi Nipunika Panwillaarachchi Kotudura Bandanage	Attendance Management System	33.34%	

Table of Contents

Details of Team members	2
1. Introduction.....	4
1.1. System Scenario.....	4
1.2. Overview of Purpose.....	5
1.3. Overview of Scope.....	6
2. System Analysis and Design Background	7
3. Consolidated class diagram.....	7
4. Justification of Classes.....	10
4.1. Class Justification	10
4.2. Identify the subsystem for the attributes and classes	14
5. Codes for Class definition and method definition	16

1. Introduction

1.1. System Scenario

The purpose of the Human Resource Management System (HRMS) is to enable the comprehensive management of HR operations across many and diverse geographic locations by strategically integrating various HR functional subsystems. By centralizing and automating the administration of crucial HR elements including leave, attendance, and personnel records, this system seeks to greatly improve operational effectiveness, increase transparency, and guarantee data accessibility across the entire company.

The HRMS solves the challenges of managing a dynamic and distributed workforce by consolidating employee management, leave management, and attendance management into a single, unified platform. This integration lowers the possibility of errors related to human data entry, streamlines overall HR procedures, and helps remove unnecessary activities that may result from managing various systems. For HR departments to effectively handle massive amounts of employee data and handle a range of HR demands, these improvements are essential.

Moreover, the integration of various subsystems into a single platform ensures that all HR-related data is easily accessible to HR managers as well as to staff members, managers, and upper management. Better departmental communication, well-informed decision-making, and a strong corporate culture are all facilitated by this accessibility. Better analytics and reporting capabilities are also made possible by improved data flow and centralized information systems, which help the company keep a closer eye on HR metrics and make strategic decisions based on updated, reliable data.

Furthermore, the HRMS is built to be highly adaptable, meeting the requirements of both small teams and operations at the enterprise level. The HRMS may be expanded and modified as businesses develop and their needs change without sacrificing security or functionality. This scalability guarantees that the HRMS will continue to offer a dependable basis for all HR operations, supporting the business's growth and strategic objectives if it enters new markets or hires more employees.

1.2. Overview of Purpose

The primary goal of the Human Resource Management System (HRMS) is to augment the capacities of human resources departments to a great extent by creating a consolidated platform that facilitates and optimizes diverse human resources activities throughout the enterprise. The latest technology was carefully developed to manage huge quantities of employee data, improve the leave application process, as well as track attendance.

The HRMS reduces the risks associated with manual processing errors and improves operational efficiency by combining many operations into a single, cohesive system.

The HRMS plays a key role in the strategic management of human resources by establishing a structured, legal HR environment that is quick to adjust to changing business requirements. It provides a strong structure that supports HR operations, makes policy enforcement easier, and raises employee engagement. It is an essential component in the optimization of human resource management. The system helps the organization grow and function more efficiently by freeing up HR experts to concentrate on more strategic projects like talent development, performance management, and succession planning. It does this by automating repetitive operations and centralizing data management.

1.3. Overview of Scope

The HRMS's scope includes a number of essential subsystems, each of which is intended to handle a particular facet of human resources management

Employee Management system :

The main component of the HRMS, this subsystem manages all aspects of employee information management, from processing terminations to onboarding new employees. It guarantees that every piece of employee data—from more intricate job-related information to fundamental personal details—is carefully recorded. All employee records are centrally stored by the Employee Management System, which makes it simple for HR staff to access and update information as needed. This helps to guarantee accuracy and uniformity of data throughout the company.

Leave Management System:

This subsystem manages the processing of leave encashment, tracking of leave balances, and management of the whole leave lifecycle for employees, from the original application to the final approval. It makes sure that all transactions are documented for future reference, automates the leave application process, and implements organizational leave policy consistently. This solution fosters justice and improves employee happiness by giving employees an intuitive and clear interface to manage their vacation entitlements.

Attendance Management System:

This subsystem makes it easier to record time and attendance data by automating the tracking and monitoring of employee attendance. It gives managers the tools they need to oversee daily attendance, analyze past data, and identify the absence or late trends for both staff members and managers. In addition to supporting correct payroll processing, the system offers insightful data on labor costs and worker productivity, all of which are critical for efficient staff administration.

2. System Analysis and Design Background

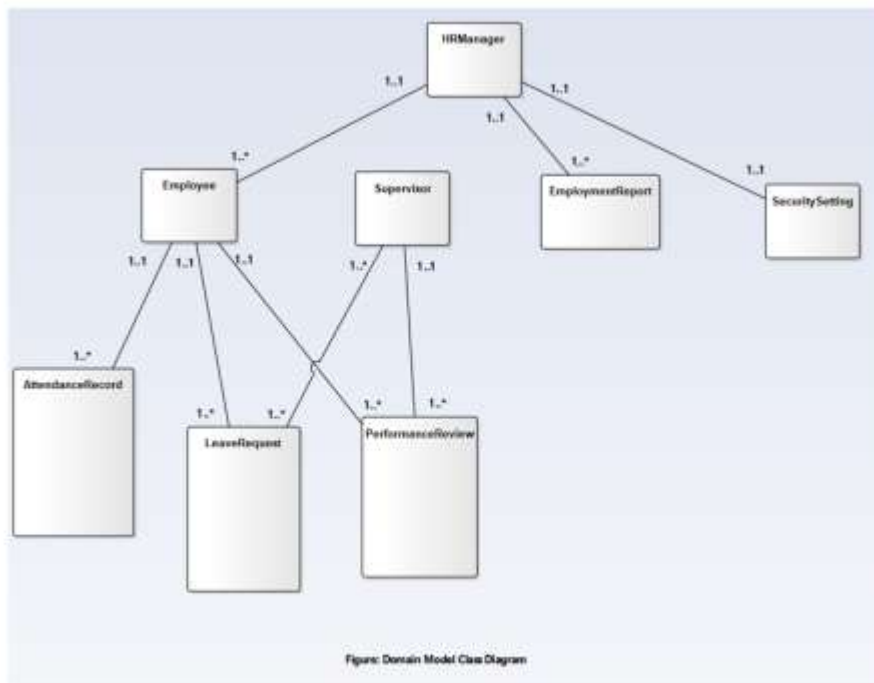
Analyzed System

System Analysis A comprehensive analysis of the current HR processes was conducted, taking into account the information flow between various HR functions, the interdependencies among HR tasks, and the workflow of HR tasks. The goal of the system design was to establish a single, centralized platform that would consolidate many processes and improve accessibility, data integrity, and process efficiency. In addition to the immediate operational needs, the design considers the strategic HR goals of supporting growth and guaranteeing flexibility.

3. Consolidated class diagram

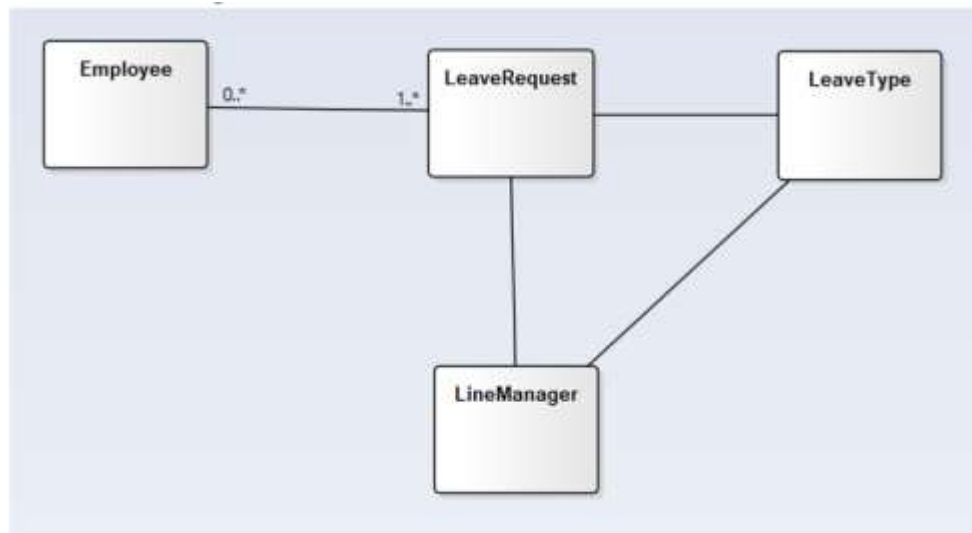
Domain model of Employee Management System

This domain model manages relationships from hiring through removing and contains classes like Employee, Supervisor, and HRManager. The main class, Employee, interacts with PerformanceReview, LeaveRequest, and AttendanceRecord. In the context of performance management and leave approval procedures, the Supervisor class represents the supervisory role by mediating PerformanceReviews and supervising LeaveRequests.



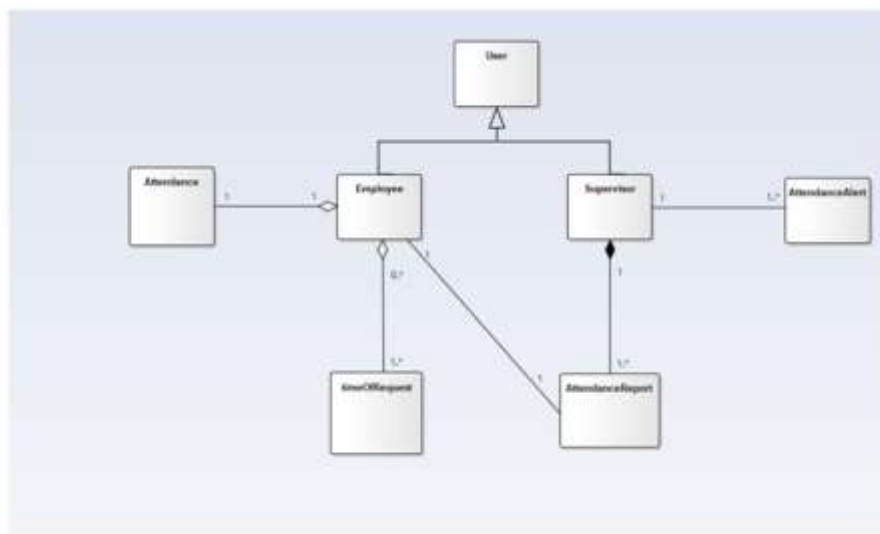
Domain model of Leave Management System

Attributes like LeaveRequest, Employee, LeaveType, and LineManager are used to show the Leave Management System. Workers may submit more than one LeaveRequest, and each LeaveRequest is connected to a particular LeaveType. Because of its connection to LeaveRequest, the LineManager class has a role in the approval procedure. For data consistency, this subsystem easily interfaces with the Employee Management System.



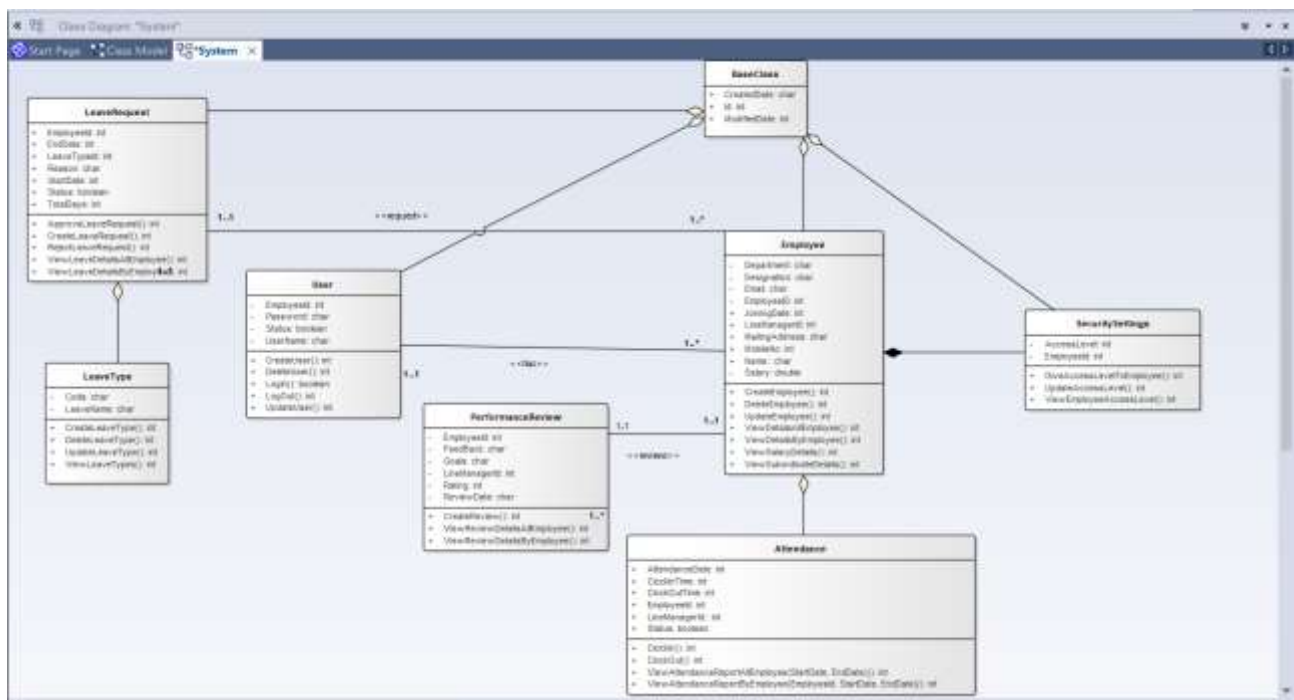
Domain model of Attendance Management System

The classes AttendanceAlert, AttendanceReport, Employee, timeOffRequest, Supervisor, and Attendance are included in this model. It outlines the procedures for tracking and documenting staff attendance, responding to leave requests, and issuing alerts and reports about attendance. By directly connecting with the Employee class and offering vital attendance data that affects other domains like payroll and compliance, it integrates with the larger HRMS.



Consolidated class diagram

The domain models of the three subsystems are integrated to form the consolidated class diagram. In order to maintain consistent interactions with other classes like LeaveRequest, Supervisor, and AttendanceRecord, it is necessary to align the Employee class across all models. To prevent duplication and guarantee that data is efficiently shared throughout the system, the consolidation process necessitates a detailed mapping of associations and properties.



4. Justification of Classes

4.1. Class Justification

Every class in the consolidated diagram has a distinct function within the HRMS. One key example is the Employee class, which contains information about an employee's employment and personal details. The workflow authority and organisational structure are reflected in the Supervisor and HRManager classes. Associations that show the possibility of many leave instances per employee include the one-to-many relationship between Employee and LeaveRequest. The direct connection between PerformanceReview and Supervisor indicates managerial oversight of the review process. Together, these classes and their affiliations capture the many relationships that exist within an HRMS and provide all-encompassing HR management.

1. Employee

Attributes :

Department , Designation , Email, EmployeeID, JoiningDate, LinemanagerId,
MailingAddress, MobileNo, Name, Salary

Methods :

CreateEmployee()
UpdateEmployee(EmployeeId)
DeleteEmployee(EmployeeId)
ViewDetailsAllEmployee()
ViewDetailsByEmployee(EmployeeId)
ViewSalaryDetails()
ViewSubordinateDetails()

Association :

- LeaveRequest: Most likely a one-to-many relationship, since LeaveRequests are tied to Employees but can exist separately.
- performance reviews: usually one-to-one relationship, although they are connected to employees, they are not 'part-of' them.
- Attendance: Aggregation, as records of attendance are linked to specific employees but are not exclusively dependent on those employees' presence.
- User: When an Employee 'has-a' User account for system access, this indicates one to many relationships and points to a connection in which the Employee and the User account are related but not directly related.

2. LeaveRequest

Attributes :

EmployeeId, LeaveTypeId, StartDate, EndDate, Reason, TotalDays, Status

Methods :

CreateLeaveRequest()

ApproveLeaveRequest()

RejectLeaveRequest()

ViewLeaveDetailsAllEmployee()

ViewLeaveDetailsByEmployee()

Association :

- Employee:it has one to many relationship. Since one employee can have many leave requests.
- LeaveType: Aggregation since LeaveRequests might exist independently but are categorised according to LeaveType.

3. PerformanceReview

Attributes :

EmployeeId, LineManagerId, ReviewDate , Feedback, Rating , Goals

Methods :

CreateReview()

ViewReviewDetailsAllEmployee()

ViewReviewDetailsByEmployee()

Association :

- Employee has a one to one relationship between PerformanceReview. One Employee can have one Performance Review at the same time one performance review belongs to only one employee

4. Attendance

Attributes :

EmployeeId, LineManagerId, AttendanceDate, ClockInTime, ClockOutTime
,Status

Methods :

ClockIn()

ClockOut()

ViewAttendanceReportAllEmployee(StartDate, EndDate)

ViewAttendanceReportByEmployee(EmployeeId, StartDate, EndDate)

Association :

- Employee: Aggregation, so although attendance records are a separate concept, they are linked to employees.

5. User

Attributes :

UserName , Password, EmployeeId, Status

Methods :

CreateUser()

UpdateUser()

DeleteUser()

LogIn()

LogOut()

Association :

- Employee, User has one to many relationships between the Employee.

6. LeaveType

Attributes :

Code, LeaveName

Methods :

CreateLeaveType()

UpdateLeaveType()

DeleteLeaveType()

ViewLeaveTypes()

Association :

- LeaveRequest: Since LeaveTypes classify LeaveRequests, aggregation is more likely to be the case here.

-

7. SecuritySettings

Attributes :

EmployeeId, AccessLevel

Methods :

GiveAccessLevelToEmployee()

UpdateAccessLevel()

ViewEmployeeAccessLevel(EmployeeId)

Association :

- Employee: If SecuritySettings cannot exist without an Employee, then this could be an example of composition—a 'part-of' connection with strict lifecycle dependency.

8. BaseClass

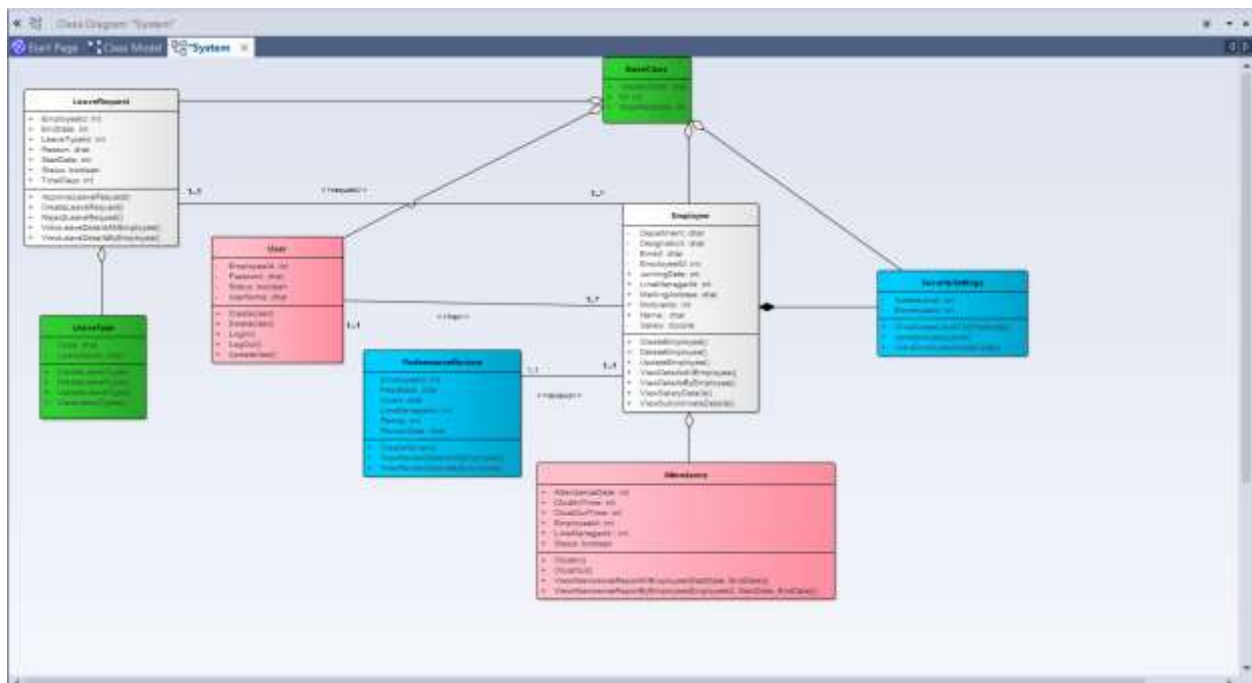
Attributes :

Id, CreatedDate, ModifiedDate

Association :


















- LeaveRequest, Security Settings, Employee, PerformanceReview,: BaseClass is a superclass in this inheritance example, and other classes inherit its attributes .

















4.2. Identify the subsystem for the attributes and classes



- Employee Management System –
- Leave Management System –
- Attendance Management System –
- Common Classes from all the subsystems -



Employee	
-	Department: char   
-	Designation: char 
-	Email: char 
-	EmployeeID: int   
+	JoiningDate: int 
+	LineManagerId: int 
+	MailingAddress: char 
+	MobileNo: int 
+	Name: char   
-	Salary: double  
+ CreateEmployee() + DeleteEmployee() + UpdateEmployee() + ViewDetailsAllEmployee() + ViewDetailsByEmployee() + ViewSalaryDetails() + ViewSubordinateDetails()	

LeaveRequest	
+	EmployeeId: int  
+	EndDate: int   
+	LeaveTypeId: int  
+	Reason: char  
+	StartDate: int   
+	Status: boolean   
+	TotalDays: int 
+ ApproveLeaveRequest() + CreateLeaveRequest() + RejectLeaveRequest() + ViewLeaveDetailsAllEmployee() + ViewLeaveDetailsByEmployee()	

5. Codes for Class definition and method definition

- **Base Class**

Code :

```
package hrmanagement.  
  
import java.util.Date.  
  
public abstract class BaseClass {  
    protected int id;  
    protected Date createdDate;  
    protected Date modifiedDate;  
}
```

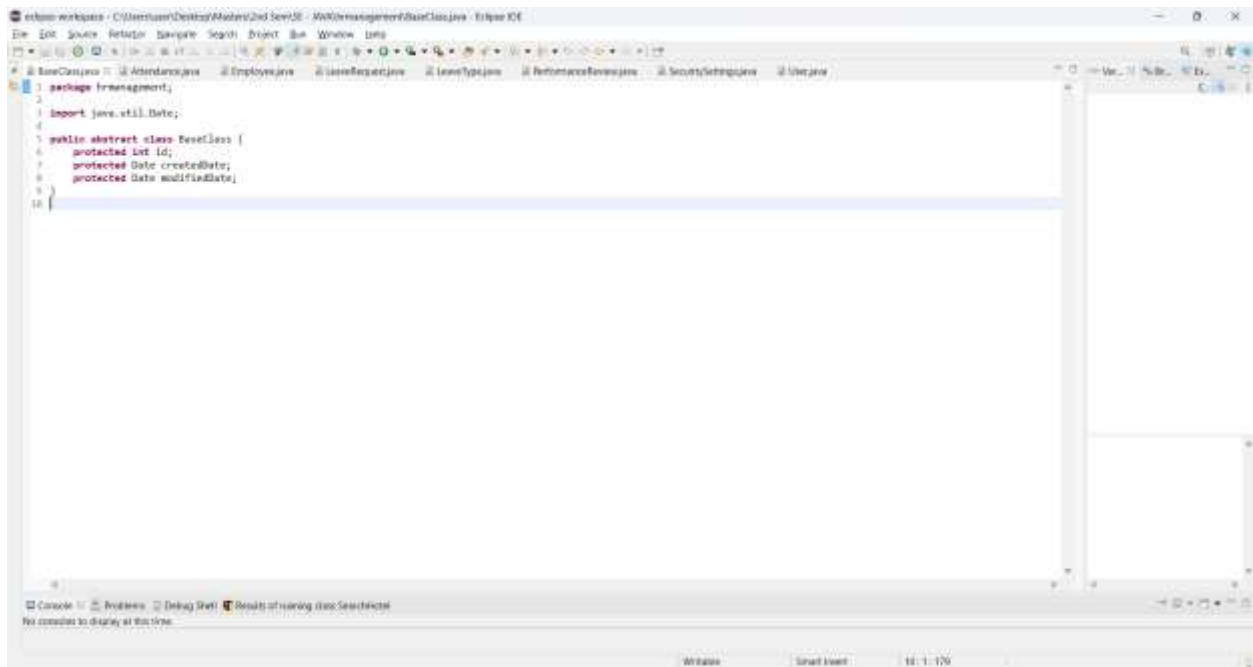


Figure 1(Base Class)

- **Attendance Class**

Code :

```
package hrmanagement;

import java.util.Date;

public class Attendance extends BaseClass {
    private int employeeId;
    private int lineManagerId;
    private Date attendanceDate;
    private Date clockInTime;
    private Date clockOutTime;
    private String status;

    // Constructor
    public Attendance() {

    }

    public void clockIn() {
        // Implementation
    }

    public void clockOut() {
        // Implementation
    }

    public void viewAttendanceReportAllEmployee(Date startDate, Date endDate) {
        // Implementation
    }

    public void viewAttendanceReportByEmployee(int employeeId, Date startDate,
Date endDate) {
        // Implementation
    }
}
```

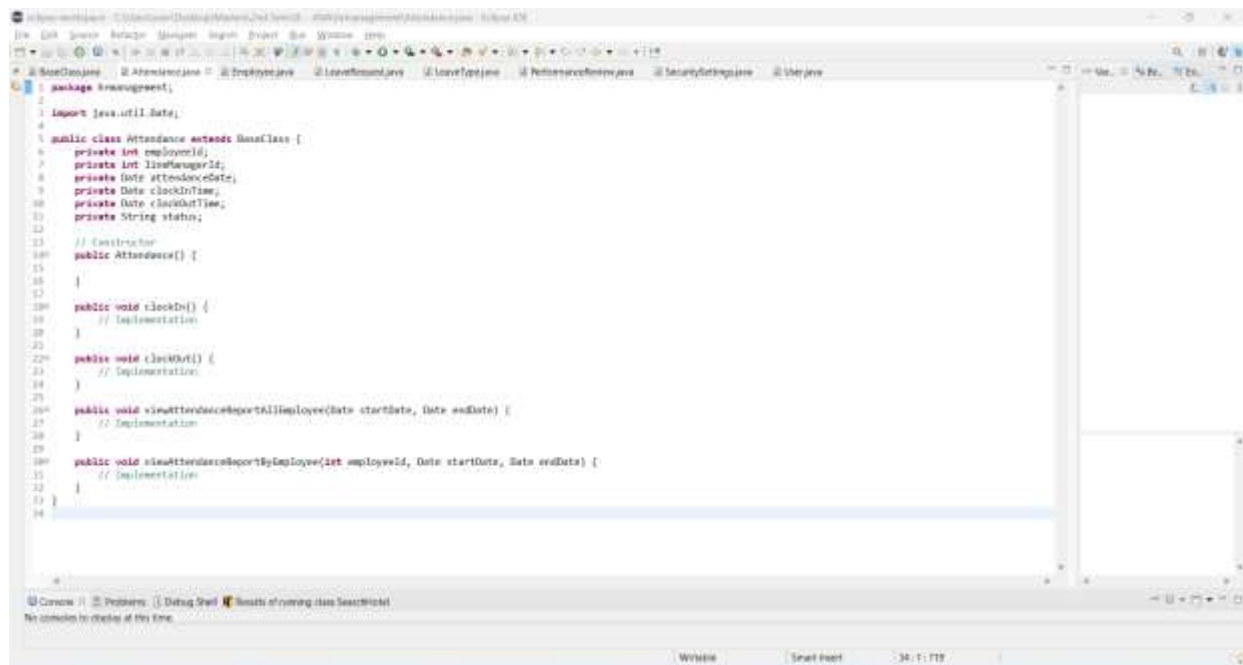


Figure 2 (Attendance class)

- **Employee class**

Code :

```
package hrmanagement;

import java.util.Date;

public class Employee extends BaseClass {
    private String employeeNo;
    private String name;
    private String mobileNo;
    private String mailingAddress;
    private String email;
    private String designation;
    private String department;
    private Date joiningDate;
    private double salary;
    private int lineManagerId;

    // Constructor
    public Employee() {

    }

    // Methods
    public void createEmployee() {
        // Implementation
    }

    public void updateEmployee(int employeeId) {
        // Implementation
    }

    public void deleteEmployee(int employeeId) {
        // Implementation
    }

    public void viewDetailsAllEmployee() {
        // Implementation
    }

    public void viewDetailsByEmployee(int employeeId) {
        // Implementation
    }

    public void viewSalaryDetails() {
        // Implementation
    }

    public void viewSubordinateDetails() {
        // Implementation    }}
}
```

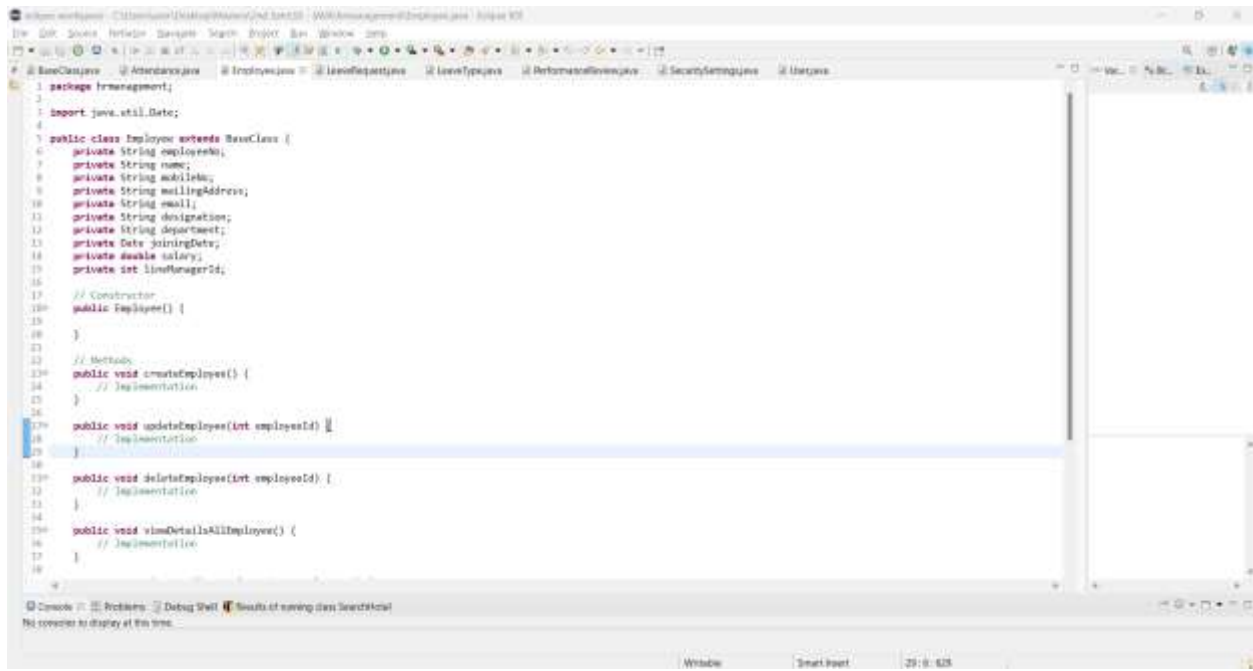


Figure 3 (Employee class)

- **Leave Request class**

Code :

```

package hrmanagement;
import java.util.Date;
public class LeaveRequest extends BaseClass {
    private int employeeId;
    private int leaveTypeId;
    private Date startDate;
    private Date endDate;
    private String reason;
    private int totalDays;
    private String status;
    // Constructor
    public LeaveRequest() {

    }
    // Methods
    public void createLeaveRequest() {
        // Implementation
    }
    public void approveLeaveRequest() {
        // Implementation
    }
    public void rejectLeaveRequest() {
        // Implementation
    }
}
  
```

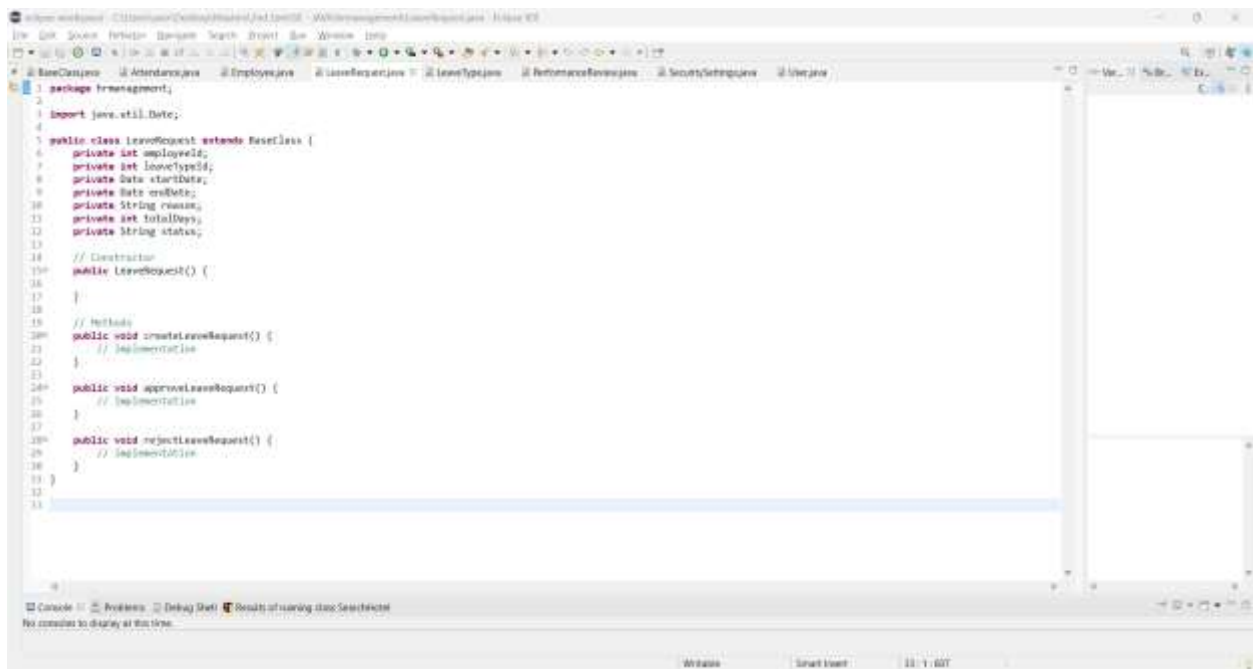


Figure 4(leaveRequest class)

- **leaveType class**

Code :

```

package hrmanagement;
import java.util.Date;
public class LeaveType extends BaseClass {
    private String code;
    private String leaveName;
    // Constructor
    public LeaveType(int id, Date createdDate, Date modifiedDate) {
        this.id = id;
        this.createdDate = createdDate;
        this.modifiedDate = modifiedDate;
    }
    // Methods
    public void createLeaveType() {
        // Implementation
    }
    public void updateLeaveType() {
        // Implementation
    }
    public void deleteLeaveType() {
        // Implementation
    }
    public void viewLeaveTypes() {
        // Implementation
    }
}

```

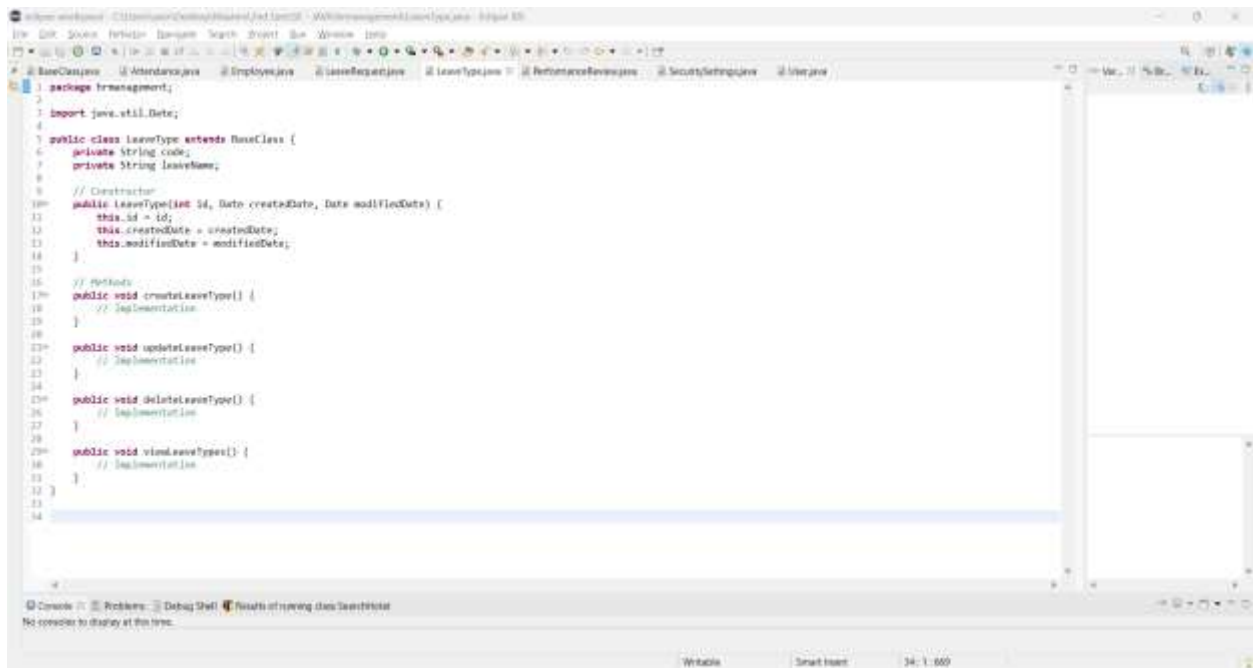


Figure 4 (leaveType class)

- **PerformanceReview class**

Code :

```

package hrmanagement;

import java.util.Date;

public class PerformanceReview {
    private int employeeId;
    private int lineManagerId;
    private Date reviewDate;
    private String feedback;
    private int rating;
    private String goals;

    // Methods
    public void viewReviewDetailsAllEmployee() {
        // Implementation
    }

    public void viewReviewDetailsByEmployee(int employeeId) {
        // Implementation
    }
}
  
```

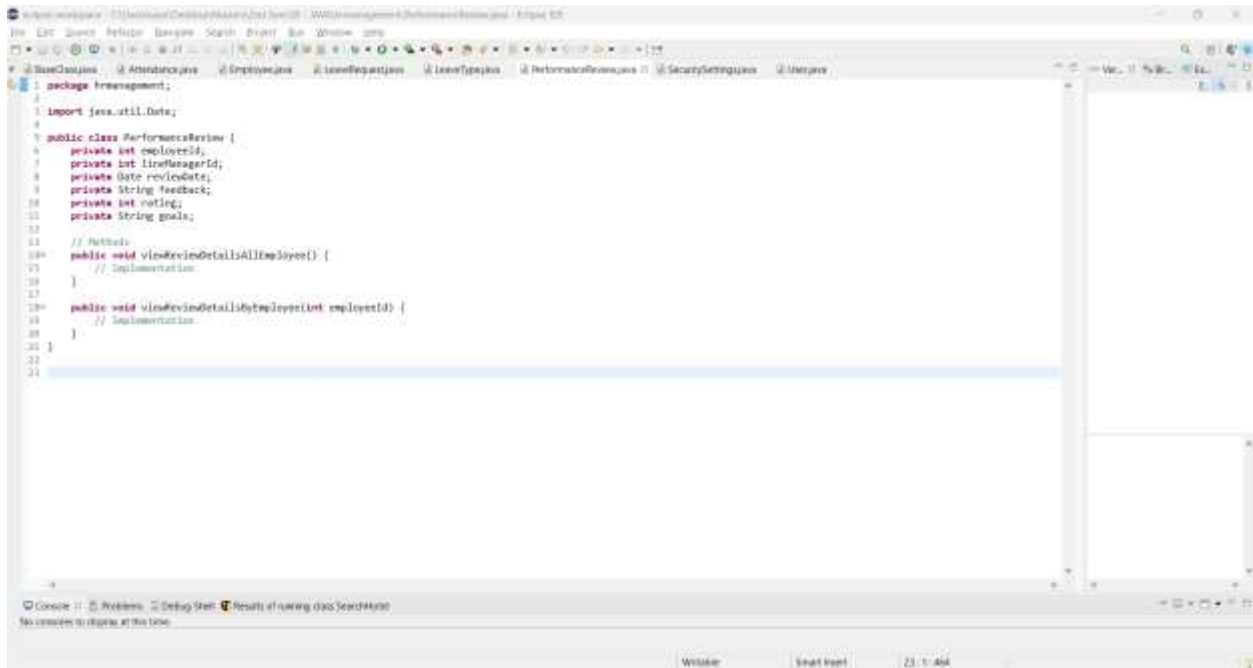


Figure 5(performanceReview class)

- **SecuritySettings class**

Code :

```

package hrmanagement;

public class SecuritySettings {
    private int employeeId;
    private String accessLevel;

    // Methods
    public void giveAccessLevelToEmployee() {
        // Implementation
    }

    public void updateAccessLevel() {
        // Implementation
    }

    public void viewEmployeeAccessLevel(int employeeId) {
        // Implementation
    }
}
  
```

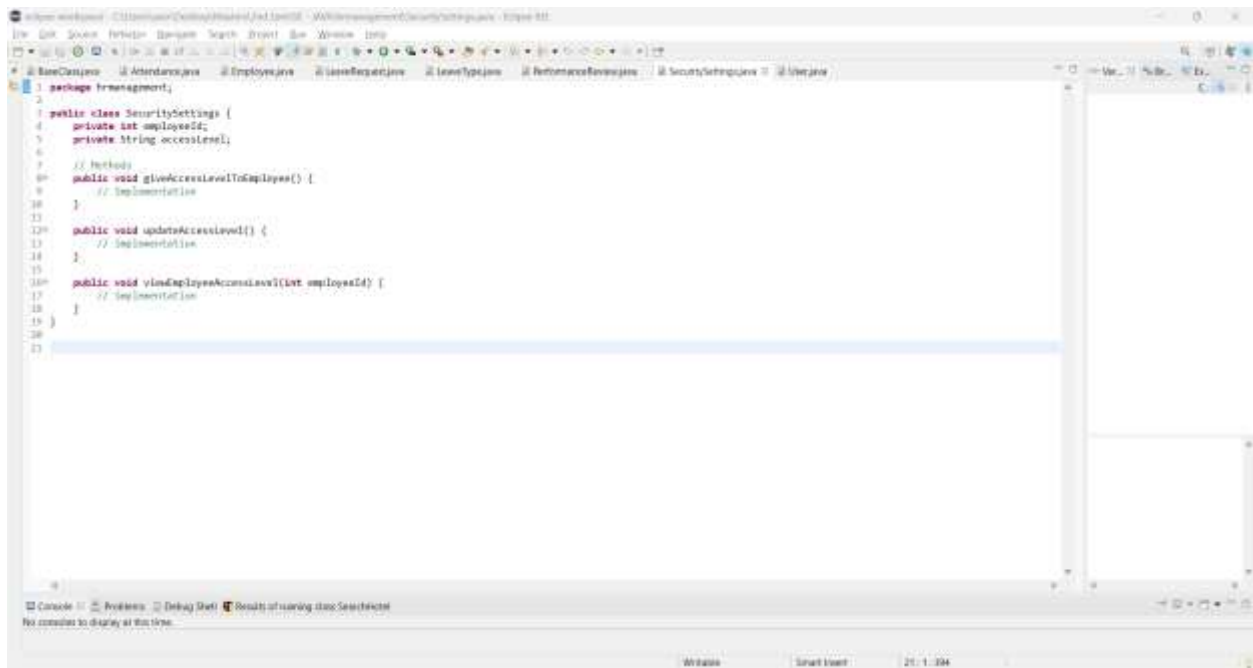


Figure 6(SecuritySettings class)

- **User class**

Code :

```

package hrmanagement;
import java.util.Date;
public class User extends BaseClass {
    private String userName;
    private String password;
    private int employeeId;
    private String status;
    // Constructor
    public User() {
    }
    // Methods
    public void createUser() {
        // Implementation
    }
    public void updateUser() {
        // Implementation
    }
    public void deleteUser() {
        // Implementation
    }
    public void logIn() {
        // Implementation
    }
    public void logOut() {
        // Implementation
    }
}
  
```

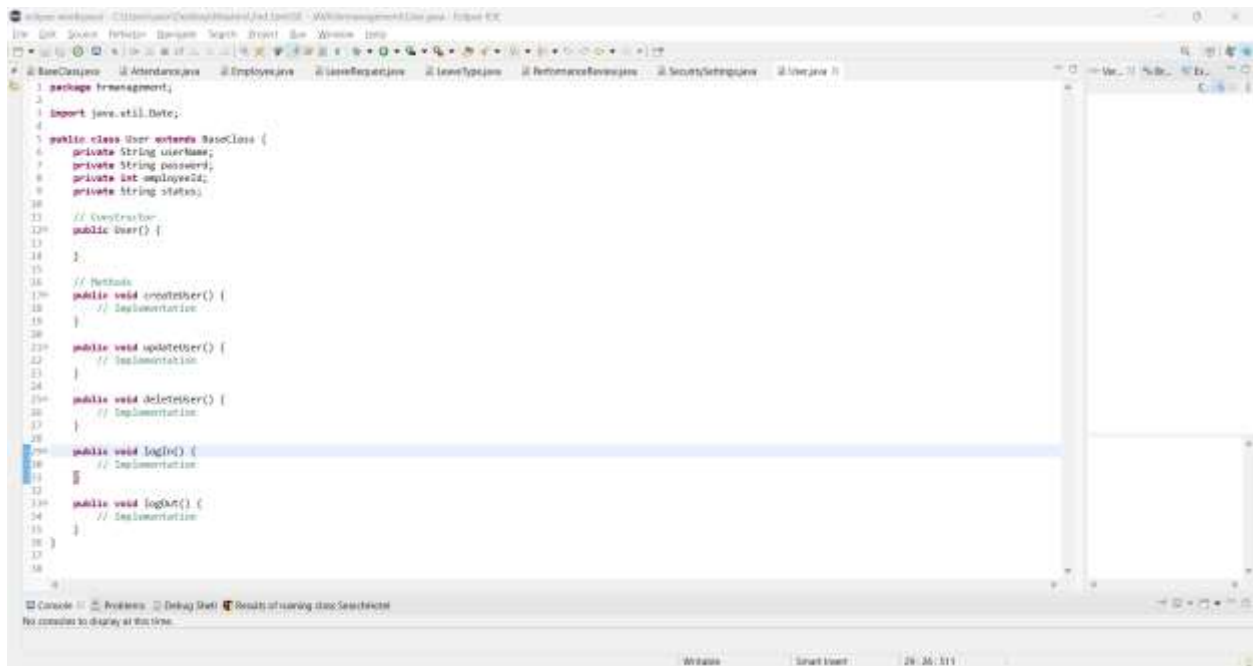



Figure 7(User class)