**Waterfall Project Part 3/3**

**Document 6- Please prepare a use case diagram, activity diagram and a use case specification document.**

**Use case Specification:**

**1. User Registration**

1. Use Case Name: Register User

2. Use Case Description: This use case allows new users to register in the system by providing their personal and contact information.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User accesses the registration page.

2. User enters required details: name, contact, email, and password.

3. User submits the registration form.

4. The system validates the entered data.

5. The system creates a new user account.

6. The system displays a successful registration message.

5. Alternate Flow:

Data Validation Failure: If the system detects invalid data (e.g., incorrect email format, weak password, duplicate email), it displays appropriate error messages and prompts the user to correct the information.

6. Exceptional Flows:

System Error: If a system error occurs during account creation, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user has access to a device with a web browser or the application installed.

The user has a valid email address.

8. Post-Conditions:

A new user account is created in the system.

The user can proceed to log in.

9. Assumptions:

The system has a functioning database to store user information.

Email service is properly configured for sending registration confirmations (if applicable).

10. Constraints:

Password complexity requirements are enforced (e.g., minimum length, special characters).

Email address format validation is implemented.

11. Dependencies:

* Database system.
* Email service (optional).

12. Inputs and Outputs:

Input: Name, contact details, email, password.

Output: Successful registration message, error messages (if any).

13. Business Rules:

* Unique email address requirement.
* Data validation rules for each field.
* Password policy.

14. Miscellaneous Information:

Consider implementing CAPTCHA to prevent bot registrations.

**2. User Login**

1. Use Case Name: Log In User

2. Use Case Description: This use case allows registered users to log in to the system using their email and password.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User accesses the login page.

2. User enters their registered email and password.

3. User submits the login form.

4. The system validates the credentials.

5. The system authenticates the user.

6. The system redirects the user to their dashboard.

5. Alternate Flow:

Invalid Credentials: If the system detects invalid credentials (incorrect email or password), it displays an error message and prompts the user to try again.

Account Locked: If the user enters incorrect credentials multiple times, the account may be locked for security reasons. Display an appropriate message.

6. Exceptional Flows:

System Error: If a system error occurs during authentication, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user has a registered account in the system.

The user knows their registered email and password.

8. Post-Conditions:

The user is logged in to the system and can access their account.

9. Assumptions:

The system has access to user credentials.

The system has a secure authentication mechanism.

10. Constraints:

Account lockout policy after multiple failed login attempts.

Password encryption for security.

11. Dependencies:

User database.

Authentication service.

12. Inputs and Outputs:

Input: Email, password.

Output: Successful login redirection, error messages (if any).

13. Business Rules:

Authentication mechanism.

Account lockout policy.

14. Miscellaneous Information:

Consider implementing multi-factor authentication for enhanced security.

**3. Upload Documents**

1. Use Case Name: Upload Documents

2. Use Case Description: This use case allows users to upload necessary documents to the system.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User navigates to the document upload section.

2. User selects the type of document to upload (e.g., educational certificate, PAN, Aadhar).

3. User selects the file from their device.

4. User uploads the document.

5. The system validates the file type and size.

6. The system stores the document.

7. The system displays a confirmation message.

5. Alternate Flow:

Invalid File Type/Size: If the system detects an invalid file type or size, it displays an error message and prompts the user to upload a valid document.

6. Exceptional Flows:

System Error: If a system error occurs during the upload process, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user is logged in to the system.

The user has the necessary documents available on their device.

8. Post-Conditions:

The document is uploaded and stored in the system.

9. Assumptions:

The system has sufficient storage space for documents.

The system has security measures to protect uploaded documents.

10. Constraints:

File size limits for uploads.

Allowed file types (e.g., PDF, JPG, PNG).

11. Dependencies:

File storage system.

12. Inputs and Outputs:

Input: Document file, document type.

Output: Confirmation message, error messages (if any).

13. Business Rules:

* File type validation.
* File size limits.
* Storage location for documents.

14. Miscellaneous Information:

Consider implementing progress bars during the upload process.

**4. HR Admin Review and Approve Documents**

1. Use Case Name: Review and Approve Documents

2. Use Case Description: This use case allows HR administrators to review and approve documents uploaded by users.

3. Actors:

Primary Actor: HR Admin

Secondary Actor: System

4. Basic Flow:

1. HR Admin logs into the system.

2. HR Admin navigates to the document review section.

3. HR Admin selects a document to review.

4. HR Admin views the document details.

5. HR Admin approves the document.

6. The system updates the document status.

7. The system notifies the user of the approval.

5. Alternate Flow:

(Rejection is a separate use case)

6. Exceptional Flows:

System Error: If a system error occurs during the approval process, the system displays a general error message and prompts the HR Admin to try again later.

7. Pre-Conditions:

The HR Admin has access to the system.

The document has been uploaded by a user.

8. Post-Conditions:

The document status is updated to "Approved."

The user is notified of the approval.

9. Assumptions:

The HR Admin has the necessary permissions to approve documents.

10. Constraints:

Access control to the document review section.

11. Dependencies:

Document storage system.

Notification system.

12. Inputs and Outputs:

Input: Document to review, approval action.

Output: Updated document status, notification to the user.

13. Business Rules:

Approval workflow.

HR Admin roles and permissions.

14. Miscellaneous Information:

Implement audit logging for document approvals.

**5. HR Admin Reject Documents**

1. Use Case Name: Reject Documents

2. Use Case Description: This use case allows HR administrators to reject documents uploaded by users and request re-upload.

3. Actors:

Primary Actor: HR Admin

Secondary Actor: System, User

4. Basic Flow:

1. HR Admin logs into the system.

2. HR Admin navigates to the document review section.

3. HR Admin selects a document to review.

4. HR Admin views the document details.

5. HR Admin rejects the document.

6. HR Admin provides a reason for rejection.

7. The system updates the document status.

8. The system notifies the user of the rejection and reason.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs during the rejection process, the system displays a general error message and prompts the HR Admin to try again later.

7. Pre-Conditions:

The HR Admin has access to the system.

The document has been uploaded by a user.

8. Post-Conditions:

* The document status is updated to "Rejected."
* The user is notified of the rejection and reason.
* The user is prompted to re-upload the document.

9. Assumptions:

* The HR Admin has the necessary permissions to reject documents.
* The system supports rejection reasons.

10. Constraints:

* Access control to the document review section.
* Mandatory rejection reason.

11. Dependencies:

Document storage system.

Notification system.

12. Inputs and Outputs:

Input: Document to review, rejection action, rejection reason.

Output: Updated document status, notification to the user, prompt to re-upload.

13. Business Rules:

Rejection workflow.

HR Admin roles and permissions.

14. Miscellaneous Information:

Implement audit logging for document rejections.

**6. View Attendance Calendar**

1. Use Case Name: View Attendance Calendar

2. Use Case Description: This use case allows users to view their attendance records in a calendar format.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the attendance section.

3. User selects the desired month and year.

4. The system displays the user's attendance records in a calendar format.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs while retrieving attendance data, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user is logged in to the system.

Attendance data exists for the user.

8. Post-Conditions:

The user views their attendance records in a calendar format.

9. Assumptions:

The system has access to the user's attendance data.

10. Constraints:

Access control to the attendance section.

11. Dependencies:

Attendance tracking system.

12. Inputs and Outputs:

Input: Month, year.

Output: Attendance calendar.

13. Business Rules:

Attendance record retrieval.

Calendar display format.

14. Miscellaneous Information:

Consider allowing users to export their attendance data.

**7. Submit Attendance Correction**

1. Use Case Name: Submit Attendance Correction

2. Use Case Description: This use case allows users to submit requests for correcting errors in their attendance records.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the attendance section.

3. User selects the date with the attendance error.

4. User submits a correction request with details.

5. The system saves the correction request.

6. The system notifies the department manager.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs while submitting the correction request, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user is logged in to the system.

The user has identified an error in their attendance record.

8. Post-Conditions:

The correction request is saved in the system.

The department manager is notified of the request.

9. Assumptions:

The system supports the submission of attendance correction requests.

The system can notify the department manager.

10. Constraints

Limited time frame for submitting correction requests.

Mandatory fields for correction requests (e.g., reason).

11. Dependencies:

Attendance tracking system.

Notification system.

12. Inputs and Outputs:

Input: Date with error, correction details, reason.

Output: Confirmation message, notification to the manager.

13. Business Rules:

Correction request submission workflow.

Notification rules.

14. Miscellaneous Information:

Implement a clear and user-friendly interface for submitting correction requests.

**8. Manager Approve Attendance Correction**

1. Use Case Name: Approve Attendance Correction

2. Use Case Description: This use case allows department managers to approve attendance correction requests submitted by users.

3. Actors:

Primary Actor: Department Manager

Secondary Actor: System

4. Basic Flow:

1. Department Manager logs into the system.

2. Department Manager navigates to the attendance correction requests section.

3. Department Manager selects a correction request to review.

4. Department Manager views the correction details.

5. Department Manager approves the correction.

6. The system updates the attendance record with the correction.

7. The system notifies the user of the approval.

5. Alternate Flow:

(Rejection is a separate use case).

6. Exceptional Flows:

System Error: If a system error occurs while approving the correction, the system displays a general error message and prompts the manager to try again later.

7. Pre-Conditions:

The Department Manager has access to the system.

A user has submitted an attendance correction request.

8. Post-Conditions:

The attendance record is updated with the correction.

The user is notified of the approval.

9. Assumptions:

The Department Manager has the authority to approve attendance corrections.

10. Constraints:

Access control to the attendance correction requests section.

11. Dependencies:

Attendance tracking system.

Notification system.

12. Inputs and Outputs:

Input: Correction request to review, approval action.

Output: Updated attendance record, notification to the user.

13. Business Rules:

Approval workflow.

Manager roles and permissions.

14. Miscellaneous Information:

Implement audit logging for attendance correction approvals.

**9. Manager Reject Attendance Correction**

1. Use Case Name: Reject Attendance Correction

2. Use Case Description: This use case allows department managers to reject attendance correction requests submitted by users.

3. Actors:

Primary Actor: Department Manager

Secondary Actor: System, User

4. Basic Flow:

1. Department Manager logs into the system.

2. Department Manager navigates to the attendance correction requests section.

3. Department Manager selects a correction request to review.

4. Department Manager views the correction details.

5. Department Manager rejects the correction.

6. Department Manager provides a reason for rejection.

7. The system updates the correction request status.

8. The system notifies the user of the rejection and reason.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs while rejecting the correction, the system displays a general error message and prompts the manager to try again later.

7. Pre-Conditions:

The Department Manager has access to the system.

A user has submitted an attendance correction request.

8. Post-Conditions:

* The correction request status is updated to "Rejected."
* The user is notified of the rejection and reason.
* The user can re-apply for the attendance correction.

9. Assumptions:

The Department Manager has the authority to reject attendance corrections.

The system supports rejection reasons.

10. Constraints:

Access control to the attendance correction requests section.

Mandatory rejection reason.

11. Dependencies:

Attendance tracking system.

Notification system.

12. Inputs and Outputs:

Input: Correction request to review, rejection action, rejection reason.

Output: Updated correction request status, notification to the user.

13. Business Rules:

Rejection workflow.

Manager roles and permissions.

14. Miscellaneous Information:

Implement audit logging for attendance correction rejections.

**10. View Leave Types and Balance**

1. Use Case Name: View Leave Types and Balance

2. Use Case Description: This use case allows users to view the different leave types available to them and their respective balances.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the leave section.

3. The system displays the leave types and balances available to the user.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs while retrieving leave information, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user is logged in to the system.

Leave data exists for the user.

8. Post-Conditions:

The user views their available leave types and balances.

9. Assumptions:

The system has access to the user's leave data.

10. Constraints:

Access control to the leave section.

11. Dependencies:

Leave management system.

12. Inputs and Outputs:

Input: User ID.

Output: Leave types and balances.

13. Business Rules:

Leave type definitions.

Leave balance calculation.

14. Miscellaneous Information:

Consider providing a history of leave taken.

**11. Apply for Leave**

1. Use Case Name: Apply for Leave

2. Use Case Description: This use case allows users to apply for leave, specifying the dates and reason for the leave.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the leave application section.

3. User selects the leave type.

4. User selects the start and end dates for the leave.

5. User provides a reason for the leave.

6. User submits the leave application.

7. The system saves the leave application.

8. The system notifies the manager.

5. Alternate Flow:

Insufficient Leave Balance: If the user's leave balance is insufficient for the requested leave period, the system displays an error message.

6. Exceptional Flows:

System Error: If a system error occurs while submitting the leave application, the system displays a general error message and prompts the user to try again later.

7. Pre-Conditions:

The user is logged in to the system.

8. Post-Conditions:

The leave application is saved in the system.

The manager is notified of the leave application.

9. Assumptions:

The system supports the submission of leave applications.

The system can notify the manager.

10. Constraints:

Minimum notice period for leave applications.

Mandatory fields for leave applications (e.g., reason).

11. Dependencies:

Leave management system.

Notification system.

12. Inputs and Outputs:

Input: Leave type, start date, end date, reason.

Output: Confirmation message, notification to the manager.

13. Business Rules:

Leave application submission workflow.

Leave balance validation.

14. Miscellaneous Information:

Implement a calendar for selecting leave dates.

**12. Manager Reject Leave Request**

1. Use Case Name: Reject Leave Request

2. Use Case Description: This use case allows managers to reject leave requests submitted by users before approval.

3. Actors:

Primary Actor: Manager

Secondary Actor: System, User

4. Basic Flow:

1. Manager logs into the system.

2. Manager navigates to the leave requests section.

3. Manager selects a leave request to review.

4. Manager views the leave details.

5. Manager rejects the leave request.

6. Manager provides a reason for rejection.

7. The system updates the leave request status.

8. The system notifies the user of the rejection and reason.

5. Alternate Flow:

N/A

6. Exceptional Flows:

System Error: If a system error occurs while rejecting the leave request, the system displays a general error message and prompts the manager to try again later.

7. Pre-Conditions:

The Manager has access to the system.

A user has submitted a leave request.

8. Post-Conditions:

The leave request status is updated to "Rejected."

The user is notified of the rejection and reason.

9. Assumptions:

The Manager has the authority to reject leave requests.

The system supports rejection reasons.

10. Constraints:

Access control to the leave requests section.

Mandatory rejection reason.

11. Dependencies:

Leave management system.

Notification system.

12. Inputs and Outputs:

Input: Leave request to review, rejection action, rejection reason.

Output: Updated leave request status, notification to the user.

13. Business Rules:

Rejection workflow.

Manager roles and permissions.

14. Miscellaneous Information:

Implement audit logging for leave request rejections.

**13. Manager Approve Leave Request**

1. Use Case Name: Approve Leave Request

2. Use Case Description: This use case allows managers to approve leave requests submitted by users.

3. Actors:

Primary Actor: Manager

Secondary Actor: System

4. Basic Flow:

1. Manager logs into the system.

2. Manager navigates to the leave requests section.

3. Manager selects a leave request to review.

4. Manager views the leave details.

5. Manager approves the leave request.

6. The system updates the leave request status.

7. The system updates the user's leave balance.

8. The system notifies the user of the approval.

5. Alternate Flow:

N/A (Rejection is a separate use case).

6. Exceptional Flows:

System Error: If a system error occurs while approving the leave request, the system displays a general error message and prompts the manager to try again later.

7. Pre-Conditions:

The Manager has access to the system.

A user has submitted a leave request.

8. Post-Conditions:

The leave request status is updated to "Approved."

The user's leave balance is updated.

The user is notified of the approval.

9. Assumptions:

The Manager has the authority to approve leave requests.

10. Constraints:

Access control to the leave requests section.

11. Dependencies:

Leave management system.

Notification system.

12. Inputs and Outputs:

Input: Leave request to review, approval action.

Output: Updated leave request status, updated leave balance, notification to the user.

13. Business Rules:

Approval workflow.

Manager roles and permissions.

14. Miscellaneous Information:

Implement audit logging for leave request approvals.

**14. Calculate Overtime**

1. Use Case Name: Calculate Overtime.

2. Use Case Description: This use case automatically calculates the overtime duration for employees based on their punch-in/out times.

3. Actors:

Primary Actor: System

Secondary Actor: None

4. Basic Flow:

1. The system retrieves employee punch-in and punch-out data.

2. The system calculates the time between the start and end of overtime.

3. The system stores the calculated overtime duration.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Missing Punch Data: If punch-in or punch-out data is missing, the system logs an error and may require manual intervention.

7. Pre-Conditions:

Employee punch-in and punch-out data is available in the system.

8. Post-Conditions:

The overtime duration is calculated and stored in the system.

9. Assumptions:

The system knows the employee's standard working hours.

10. Constraints:

Overtime calculation rules are defined in the system.

11. Dependencies:

Attendance tracking system.

12. Inputs and Outputs:

Input: Punch-in time, punch-out time, standard working hours.

Output: Overtime duration.

13. Business Rules:

Overtime calculation rules (e.g., after how many hours overtime starts).

14. Miscellaneous Information:

Consider allowing manual adjustments to overtime calculations.

15. Flag Late Arrivals

1. Use Case Name: Flag Late Arrivals

2. Use Case Description: This use case automatically flags late arrivals if employees exceed the scheduled login time.

3. Actors:

Primary Actor: System

Secondary Actor: None

4. Basic Flow:

1. The system retrieves the employee's scheduled login time.

2. The system retrieves the employee's actual login time.

3. The system compares the actual login time to the scheduled login time.

4. If the actual login time exceeds the scheduled login time, the system flags the employee as late

5. Alternate Flow:

N/A

6. Exceptional Flows:

Missing Scheduled Time: If the scheduled login time is not available, the system logs an error and may require manual intervention.

7. Pre-Conditions:

Employee scheduled login time is available in the system.

Employee actual login time is available in the system.

8. Post-Conditions:

The employee is flagged as late if their actual login time exceeds the scheduled login time.

9. Assumptions:

The system knows the employee's scheduled login time.

10. Constraints:

Late arrival threshold (e.g., how many minutes late is considered late).

11. Dependencies:

Attendance tracking system.

Roster management system.

12. Inputs and Outputs:

Input: Scheduled login time, actual login time, late arrival threshold.

Output: Late arrival flag.

13. Business Rules:

Late arrival policy.

Late arrival threshold.

14. Miscellaneous Information:

Consider providing a grace period before flagging an employee as late.

**16. Calculate Shift Hours**

1. Use Case Name: Calculate Shift Hours.

2. Use Case Description: This use case automatically calculates shift hours from employee punch-in and punch-out data.

3. Actors:

Primary Actor: System

Secondary Actor: None

4. Basic Flow:

1. The system retrieves employee punch-in and punch-out data.

2. The system calculates the difference between the punch-out time and the punch-in time.

3. The system stores the calculated shift hours.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Missing Punch Data: If punch-in or punch-out data is missing, the system logs an error and may require manual intervention.

7. Pre-Conditions:

Employee punch-in and punch-out data is available in the system.

8. Post-Conditions:

The shift hours are calculated and stored in the system.

9. Assumptions:

N/A

10. Constraints:

N/A

11. Dependencies:

Attendance tracking system.

12. Inputs and Outputs:

Input: Punch-in time, punch-out time.

Output: Shift hours.

13. Business Rules:

N/A

14. Miscellaneous Information:

N/A

**17. Calculate Break Duration**

1. Use Case Name: Calculate Break Duration.

2. Use Case Description: This use case automatically calculates the total break duration from employee punch-out and punch-in data.

3. Actors:

Primary Actor: System

Secondary Actor: None

4. Basic Flow:

1. The system retrieves employee break punch-out and punch-in data.

2. The system calculates the difference between each break punch-in time and the corresponding break punch-out time.

3. The system sums up the duration of all breaks to calculate the total break duration.

4. The system stores the calculated break duration.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Missing Punch Data: If break punch-in or punch-out data is missing, the system logs an error and may require manual intervention.

7. Pre-Conditions:

Employee break punch-out and punch-in data is available in the system.

8. Post-Conditions:

The total break duration is calculated and stored in the system.

9. Assumptions:

Employees consistently punch-in and punch-out for breaks.

10. Constraints:

N/A

11. Dependencies:

Attendance tracking system.

12. Inputs and Outputs:

Input: Break punch-out times, break punch-in times.

Output: Total break duration.

13. Business Rules:

N/A

14. Miscellaneous Information:

N/A

**18. Calculate Total Work Time**

1. Use Case Name: Calculate Total Work Time.

2. Use Case Description: This use case automatically calculates the total work time for employees by subtracting breaks from their total shift hours.

3. Actors:

Primary Actor: System

Secondary Actor: None

4. Basic Flow:

1. The system retrieves the employee's calculated shift hours.

2. The system retrieves the employee's calculated total break duration.

3. The system subtracts the total break duration from the shift hours to calculate the total work time.

4. The system stores the calculated total work time.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Missing Shift Hours or Break Duration: If shift hours or break duration data is missing, the system logs an error and may require manual intervention.

7. Pre-Conditions:

Employee shift hours are calculated and stored in the system.

Employee total break duration is calculated and stored in the system.

8. Post-Conditions:

The total work time is calculated and stored in the system.

9. Assumptions:

N/A

10. Constraints:

N/A

11. Dependencies:

Attendance tracking system.

12. Inputs and Outputs:

Input: Shift hours, total break duration.

Output: Total work time.

13. Business Rules:

N/A

14. Miscellaneous Information:

N/A

**19. View Yearly Holiday List**

1. Use Case Name: View Yearly Holiday List.

2. Use Case Description: This use case allows users to view a list of national and public holidays for the current year.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the holiday section.

3. The system displays a list of national and public holidays for the current year.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Holiday Data Not Available: If holiday data is not available, the system displays a message indicating that the holiday list is not yet available.

7. Pre-Conditions:

The system has access to the holiday list for the current year.

8. Post-Conditions:

The user views the list of national and public holidays.

9. Assumptions:

The holiday list is updated annually.

10. Constraints:

Access control to the holiday section.

11. Dependencies:

Holiday data source.

12. Inputs and Outputs:

Input: Year.

Output: List of holidays.

13. Business Rules:

Holiday definitions.

14. Miscellaneous Information:

Consider allowing users to filter the holiday list by type (e.g., national, public).

**20. View Holidays on Calendar**

1. Use Case Name: View Holidays on Calendar.

2. Use Case Description: This use case allows users to view holidays marked on a calendar.

3. Actors:

Primary Actor: User

Secondary Actor: System

4. Basic Flow:

1. User logs into the system.

2. User navigates to the calendar view.

3. The system displays the calendar with holidays highlighted.

5. Alternate Flow:

N/A

6. Exceptional Flows:

Holiday Data Not Available: If holiday data is not available, the system displays a message indicating that the calendar cannot be displayed.

7. Pre-Conditions:

The system has access to the holiday list.

8. Post-Conditions:

The user views the calendar with holidays highlighted.

9. Assumptions:

The system has a calendar component.

10. Constraints:

Access control to the calendar section.

11. Dependencies:

Holiday data source.

Calendar component.

12. Inputs and Outputs:

Input: Year, Month.

Output: Calendar with holidays highlighted.

13. Business Rules:

Calendar display format.

Holiday highlighting.

14. Miscellaneous Information:

Consider allowing users to navigate to different months and years.

**21. Use Case: View Assigned Work Rosters**

1. Use Case Name: View Assigned Work Rosters.

2. Use Case Description: This use case allows users to view their assigned work rosters in a calendar format, showing scheduled shifts and other work-related events.

3. Actors:

Primary Actor: Employee (User)

Secondary Actors: None

4. Basic Flow:

1. User logs into the system.

2. User navigates to the "Rosters" or "Calendar" section.

3. The system displays the user's assigned work roster in a calendar format.

4. User views their shifts, including date, time, location, and role.

5. Alternate Flows:

A. Viewing Rosters for a Different Period: User selects a different week or month to view. The system updates the calendar display with the selected period's roster.

B. Viewing Details of a Specific Shift: User clicks on a specific shift in the calendar. The system displays detailed information about the shift (e.g., specific tasks, break times, supervisor).

6. Exceptional Flows:

A. No Rosters Available: If no roster is assigned for the user, the system displays a message indicating that no schedule is currently available.

B. System Error: If the system encounters an error retrieving the roster, an appropriate error message is displayed to the user.

7. Pre-Conditions:

The user must be logged into the system.

The user must have an assigned work roster in the system.

8. Post-Conditions:

The user's assigned work roster is displayed in a calendar format.

System logs access to the roster.

9. Assumptions:

The system has a mechanism for assigning work rosters to users.

User roles and permissions are correctly configured.

10. Constraints:

The calendar display must be responsive and accessible on various devices.

Data security measures must be in place to protect sensitive roster information.

11. Dependencies:

Requires a functional user authentication module.

Requires a work roster management module.

12. Inputs and Outputs:

Input: User login credentials, date/period selection (optional).

Output: Calendar view of the user's work roster, shift details (upon selection).

13. Business Rules:

Work rosters must comply with labor laws and company policies.

Users should only be able to view their own assigned rosters.

14. Miscellaneous Information:

Consider adding features like exporting the roster to a personal calendar application.

**22. Use Case: Update and Assign Employee Rosters**

1. Use Case Name: Update and Assign Employee Rosters.

2. Use Case Description: This use case allows managers to update existing employee rosters and assign rosters to employees.

3. Actors:

Primary Actor: Manager

Secondary Actors: System (for validation and storage).

4. Basic Flow:

1. Manager logs into the system.

2. Manager navigates to the "Roster Management" section.

3. Manager selects an employee to assign/update roster.

4. Manager creates a new roster or modifies an existing one by adding/removing shifts.

5. Manager assigns the roster to the selected employee.

6. The system validates the roster details (e.g., no conflicts, compliance with labor laws).

7. The system saves the updated roster and notifies the employee (optional).

5. Alternate Flows:

A. Creating a New Roster Template: Manager creates a new roster template for future use.

B. Modifying an Existing Roster Template: Manager modifies an existing roster template.

C. Assigning a Roster to Multiple Employees: Manager selects multiple employees and assigns the same roster to all of them.

6. Exceptional Flows:

A. Roster Conflict: The system detects a conflict (e.g., overlapping shifts, employee unavailability). The system alerts the manager, and the manager must resolve the conflict before saving.

B. Validation Failure: The system detects a violation of business rules (e.g., exceeding maximum working hours). The system alerts the manager, and the manager must correct the issue before saving.

C. System Error: If the system encounters an error saving the roster, an appropriate error message is displayed to the manager.

7. Pre-Conditions:

The manager must be logged into the system with appropriate permissions.

Employee data must exist in the system.

8. Post-Conditions:

The employee's work roster is updated in the system.

The system logs the changes made to the roster.

The employee is notified of the updated roster (optional).

9. Assumptions:

Managers have the necessary training to use the roster management system.

The system has a mechanism for defining and enforcing business rules related to rostering.

10. Constraints:

The system must enforce data integrity and prevent unauthorized access to roster information.

The system should provide an audit trail of all roster changes.

11. Dependencies:

Requires a functional user authentication module.

Requires an employee management module.

May depend on a notification system.

12. Inputs and Outputs:

Input: Manager login credentials, employee selection, shift details, roster assignments.

Output: Updated work roster in the system, success/error messages, employee notification (optional).

13. Business Rules:

Rosters must comply with labor laws, company policies, and employee contracts.

Managers must have the authority to assign rosters to specific employees.

14. Miscellaneous Information:

Consider adding features like automated roster generation based on employee availability and demand forecasts.

Consider providing reporting capabilities to track roster utilization and labor costs.

**23. Use Case: View Monthly Salary Slips**

1. Use Case Name: View Monthly Salary Slips.

2. Use Case Description: This use case allows users to view their monthly salary slips after they have been released by HR or payroll.

3. Actors:

Primary Actor: Employee (User)

Secondary Actors: None.

4. Basic Flow:

1. User logs into the system.

2. User navigates to the "Salary Slips" or "Payroll" section.

3. The system displays a list of available salary slips, typically by month and year.

4. User selects a specific salary slip to view.

5. The system displays the selected salary slip with detailed information about earnings, deductions, and taxes.

5. Alternate Flows:

A. Downloading the Salary Slip: User selects an option to download the salary slip in a specific format (e.g., PDF).

B. Viewing a Summary of Year-to-Date Earnings: User selects an option to view a summary of their earnings and deductions for the current year.

6. Exceptional Flows:

A. No Salary Slips Available: If no salary slips are available for the user, the system displays a message indicating that no salary slips have been released yet.

B. Unauthorized Access: If the user attempts to access a salary slip that they are not authorized to view, the system displays an error message.

C. System Error: If the system encounters an error retrieving the salary slip, an appropriate error message is displayed to the user.

7. Pre-Conditions:

The user must be logged into the system.

Salary slips must have been generated and released by HR/payroll.

8. Post-Conditions:

The user's selected salary slip is displayed.

System logs access to the salary slip.

9. Assumptions:

Salary slips are generated and stored in a secure format.

User roles and permissions are correctly configured to restrict access to salary information.

10. Constraints:

Data security measures must be in place to protect sensitive salary information.

The system should comply with relevant data privacy regulations.

11. Dependencies:

Requires a functional user authentication module.

Requires a payroll processing module.

12. Inputs and Outputs:

Input: User login credentials, salary slip selection.

Output: Display of the selected salary slip, downloadable salary slip (optional), year-to-date summary (optional).

13. Business Rules:

Users should only be able to view their own salary slips.

Salary slips should only be released after payroll processing is complete.

14. Miscellaneous Information:

Consider adding features like the ability to request corrections to salary slips.

**24. Use Case: Access and Complete Assigned Training Sessions**

1. Use Case Name: Access and Complete Assigned Training Sessions.

2. Use Case Description: This use case allows users to access a list of their assigned training sessions and complete them online.

3. Actors:

Primary Actor: Employee (User)

Secondary Actors: None

4. Basic Flow:

1. User logs into the system.

2. User navigates to the "Training" or "Learning" section.

3. The system displays a list of the user's assigned training sessions.

4. User selects a training session to begin.

5. The system presents the training content (e.g., videos, documents, interactive modules).

6. User completes the training session.

7. The system records the user's progress and completion status.

5. Alternate Flows:

A. Resuming an Incomplete Training Session: User selects a training session that they previously started but did not finish. The system resumes the training session from where the user left off.

B. Viewing Training Session Details: User selects a training session to view details such as description, objectives, duration, and instructor.

6. Exceptional Flows:

A. No Assigned Training Sessions: If the user has no assigned training sessions, the system displays a message indicating that no training is currently assigned.

B. Training Session Unavailable: If a training session is unavailable (e.g., due to maintenance), the system displays an error message.

C. System Error: If the system encounters an error during the training session, an appropriate error message is displayed to the user.

7. Pre-Conditions:

The user must be logged into the system.

The user must have assigned training sessions.

Training content must be available in the system.

8. Post-Conditions:

The user's progress and completion status are updated in the system.

The user may receive a confirmation message or certificate of completion.

9. Assumptions:

Training content is designed to be accessible and engaging for users.

The system supports various types of training content (e.g., videos, documents, interactive modules).

10. Constraints:

The system should track user progress accurately.

The system should be able to handle a large number of concurrent users.

11. Dependencies:

Requires a functional user authentication module.

Requires a training management module.

12. Inputs and Outputs:

Input: User login credentials, training session selection, user interactions during the training session.

Output: Presentation of training content, progress updates, confirmation messages, certificates of completion.

13. Business Rules:

Users must complete all required training sessions within a specified timeframe.

Some training sessions may have prerequisites that must be completed first.

14. Miscellaneous Information:

Consider adding features like gamification to increase user engagement.

Consider providing reporting capabilities to track training completion rates and user performance.

**25. Use Case: Start Tests for Training**

1. Use Case Name: Start Tests for Training.

2. Use Case Description: This use case allows users to start tests or assessments associated with completed training sessions.

3. Actors:

Primary Actor: Employee (User)

Secondary Actors: None

4. Basic Flow:

1. User logs into the system.

2. User navigates to the "Training" or "Learning" section.

3. User selects a completed training session that has an associated test.

4. User clicks a button or link to "Start Test".

5. The system presents the test questions to the user.

5. Alternate Flows:

A. Reviewing Test Instructions: User clicks a link to view test instructions before starting the test.

B. Test Already Started: If the user has already started the test, the system asks if they want to resume or start a new attempt (if allowed).

6. Exceptional Flows:

A. No Test Available: If no test is associated with the selected training session, the system displays a message indicating that no test is available.

B. Test Already Completed: If the user has already completed the test, the system displays their results and prevents them from starting a new attempt (if limited attempts).

C. System Error: If the system encounters an error loading the test, an appropriate error message is displayed to the user.

7. Pre-Conditions:

The user must be logged into the system.

The user must have completed a training session that has an associated test.

8. Post-Conditions:

The test is presented to the user.

The system starts tracking the user's progress and time spent on the test.

9. Assumptions:

Tests are designed to assess the user's understanding of the training material.

The system supports various question types (e.g., multiple choice, true/false, short answer).

10. Constraints:

The system should enforce time limits for tests (if applicable).

The system should prevent users from accessing unauthorized resources during the test.

11. Dependencies:

Requires a functional user authentication module.

Requires a training management module.

Requires a test/assessment module.

12. Inputs and Outputs:

Input: User login credentials, training session selection, start test request.

Output: Presentation of the test questions.

13. Business Rules:

Users may have a limited number of attempts to complete a test.

The system should automatically grade objective questions and provide feedback to the user.

14. Miscellaneous Information:

Consider adding features like a progress bar to show the user how far they are into the test.

**26. Use Case: View Results and Feedback from HR for Training Tests**

1. Use Case Name: View Results and Feedback from HR for Training Tests.

2. Use Case Description: This use case allows users to view their results on completed training tests and any feedback provided by HR or trainers.

3. Actors:

Primary Actor: Employee (User)

Secondary Actors: None

4. Basic Flow:

1. User logs into the system.

2. User navigates to the "Training" or "Learning" section.

3. User selects a completed training session for which they want to see the test results.

4. The system displays the user's test score, answers, and any feedback provided by HR or trainers.

5. Alternate Flows:

A. Viewing Detailed Answers: User clicks a link to view the correct answers and explanations for the questions they answered incorrectly.

B. Printing Test Results: User selects an option to print their test results.

6. Exceptional Flows:

A. No Results Available: If the test has not been graded yet or results are not yet released, the system displays a message indicating that the results are not yet available.

B. Unauthorized Access: If the user attempts to access results for a test they did not take, the system displays an error message.

C. System Error: If the system encounters an error retrieving the test results, an appropriate error message is displayed to the user.

7. Pre-Conditions:

The user must be logged into the system.

The user must have completed a training test.

The test must have been graded, and results must be released to the user.

8. Post-Conditions:

The user's test results and feedback are displayed.

9. Assumptions:

The system stores test results and feedback securely.

HR or trainers are responsible for grading tests and providing feedback.

10. Constraints:

Data security measures must be in place to protect sensitive test results and feedback.

The system should comply with relevant data privacy regulations.

11. Dependencies:

Requires a functional user authentication module.

Requires a training management module.

Requires a test/assessment module.

12. Inputs and Outputs:

Input: User login credentials, training session selection.

Output: Display of test results, feedback from HR/trainers, correct answers and explanations (optional), printable test results (optional).

13. Business Rules:

Users should only be able to view their own test results and feedback.

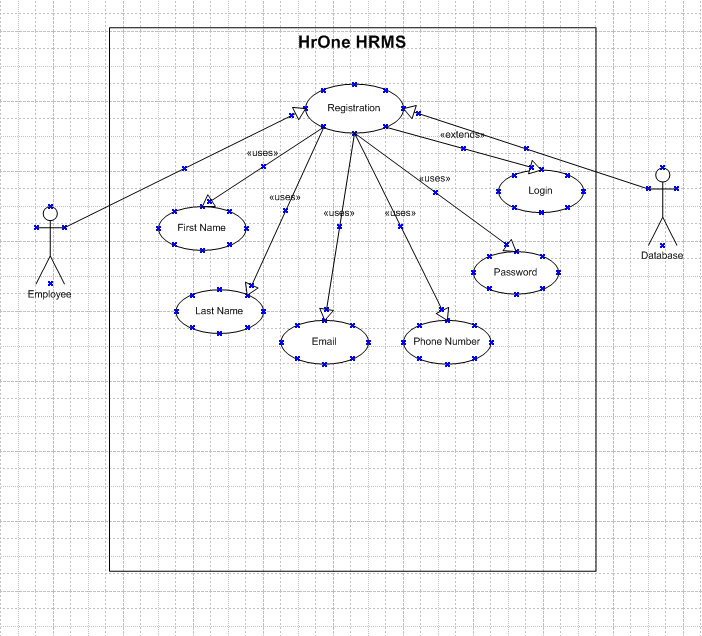
Test results and feedback should only be released after the grading process is complete.

14. Miscellaneous Information:

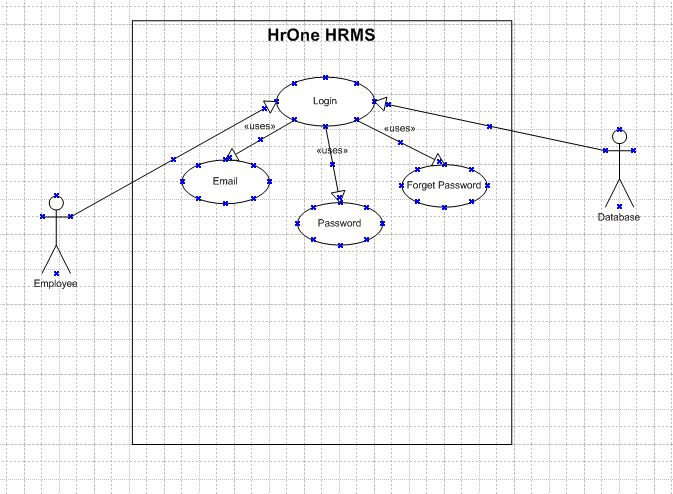
Consider adding features like the ability to appeal a test grade or provide feedback on the training session.

**Use Case Diagram:**

**1. User Registration**



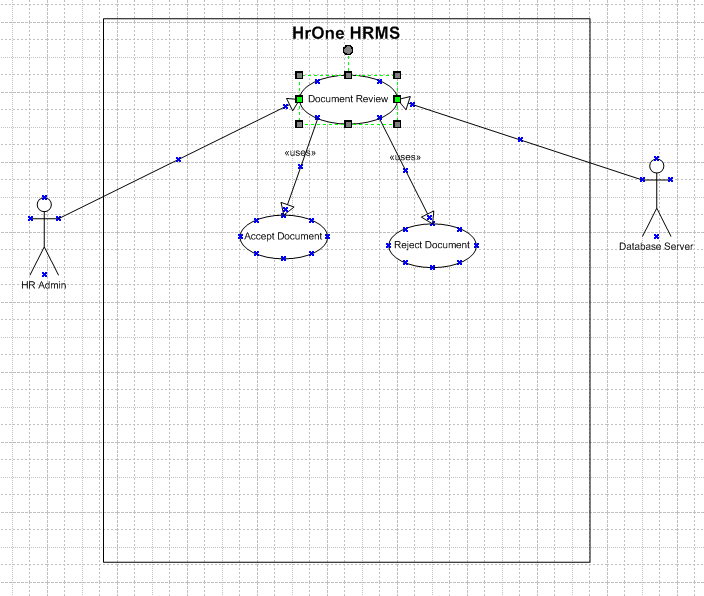
**2. User Login**



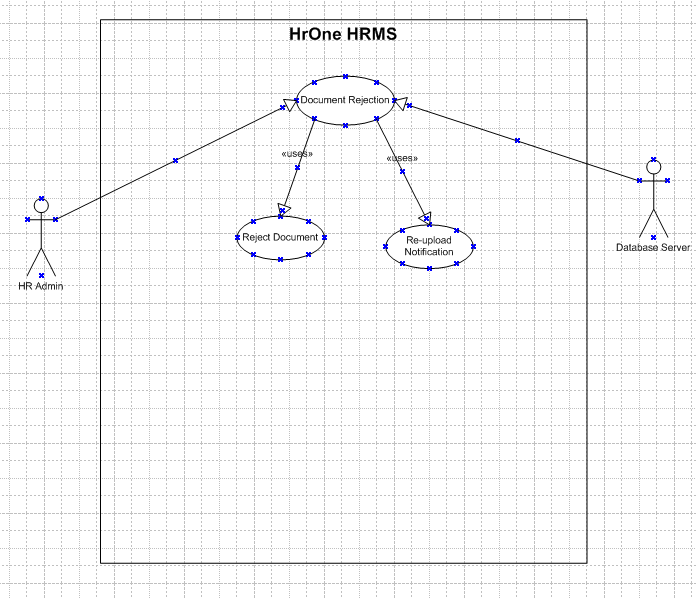
**3. Document Upload**



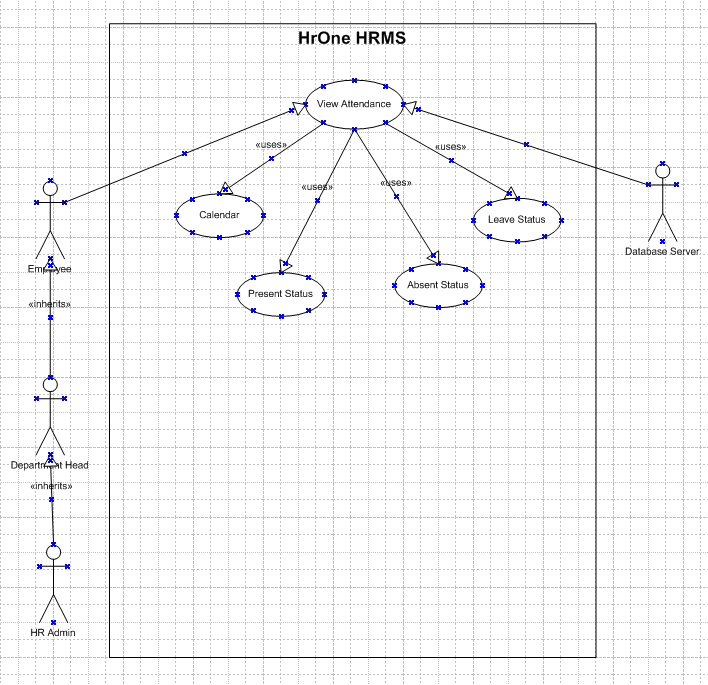
4. **Document Review and Approval**



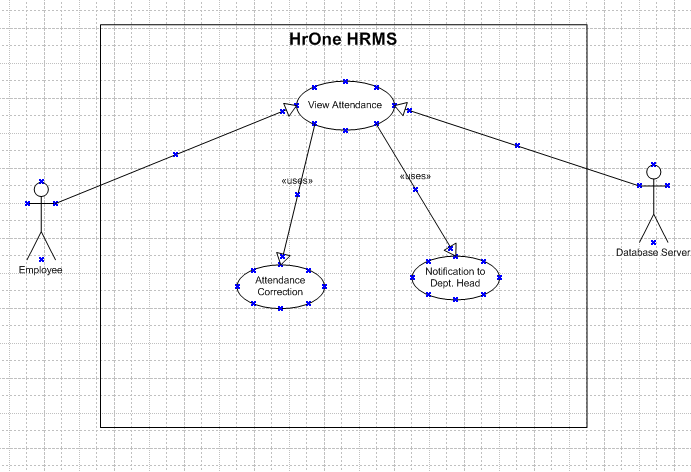
**5. Document Rejection**



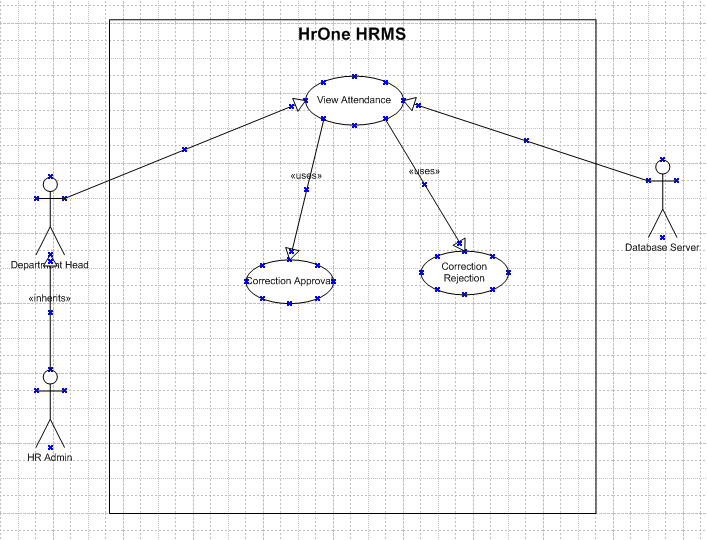
**6. View Attendance**



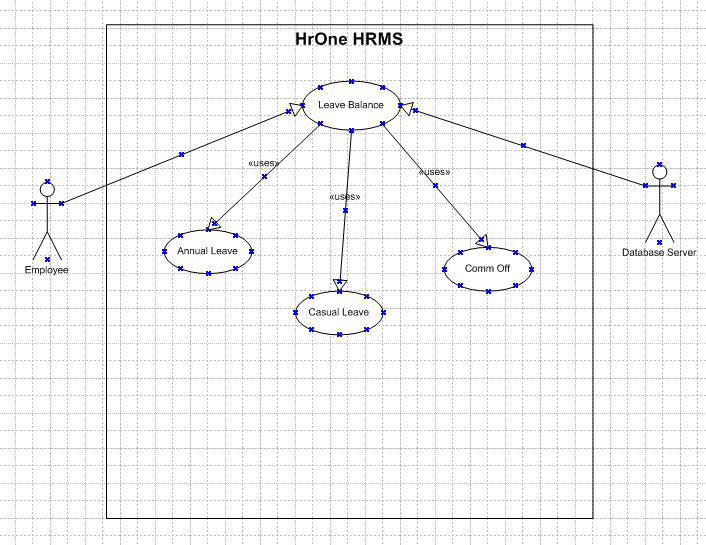
**7. Attendance Correction Submission**

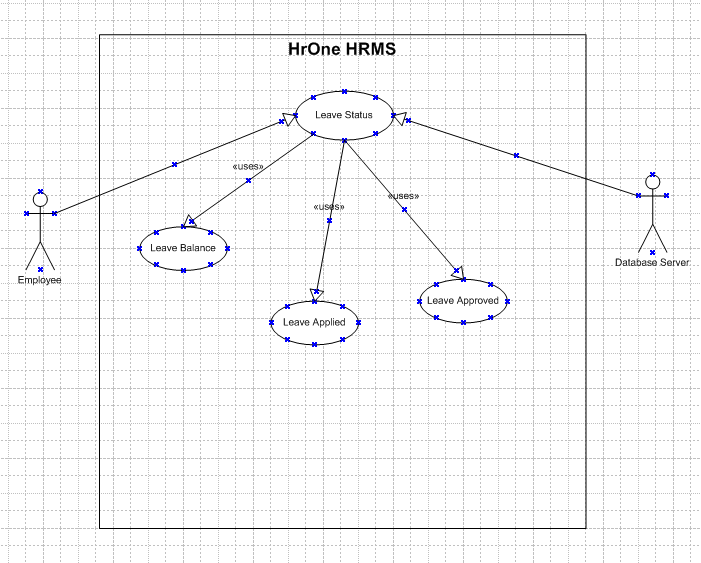


**8. Approve Attendance Corrections**

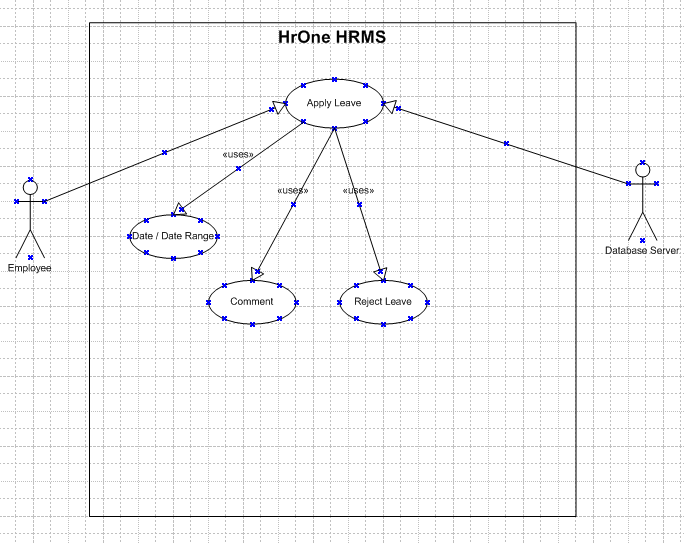


**09. View Leave Types and Balance**

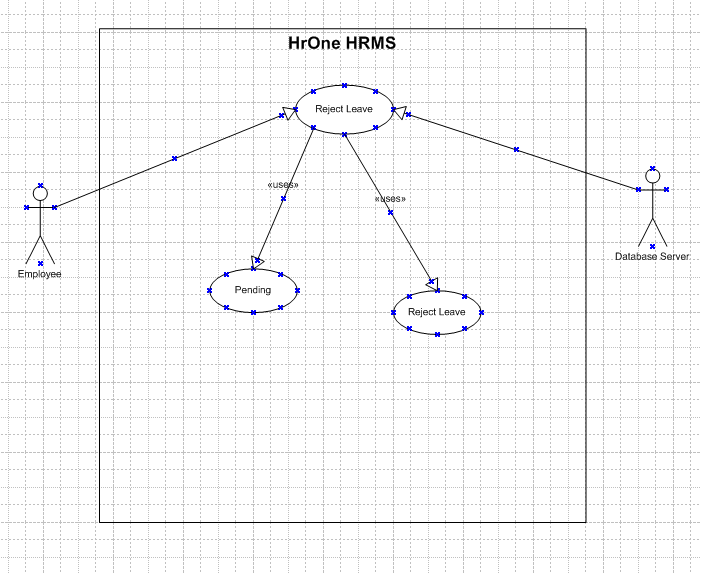




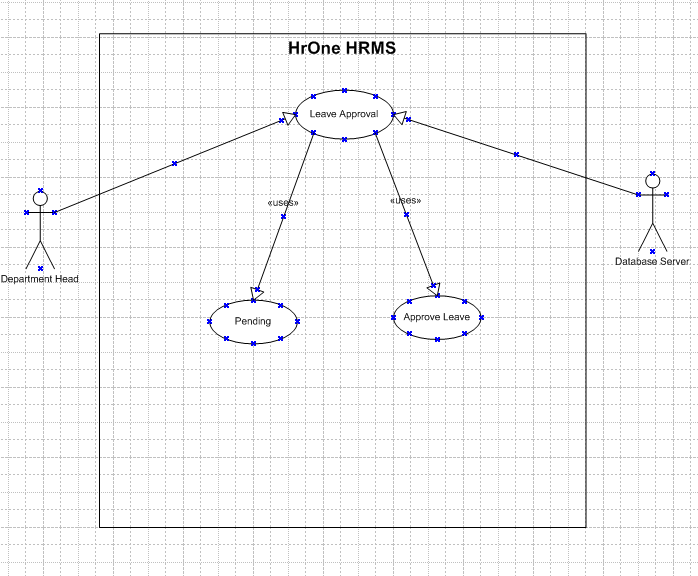
**10. Apply for Leave**

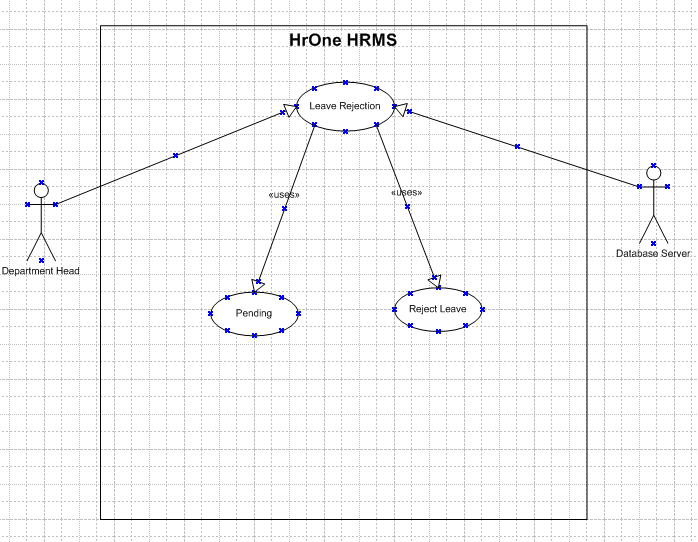


**11. Reject Leave Requests**

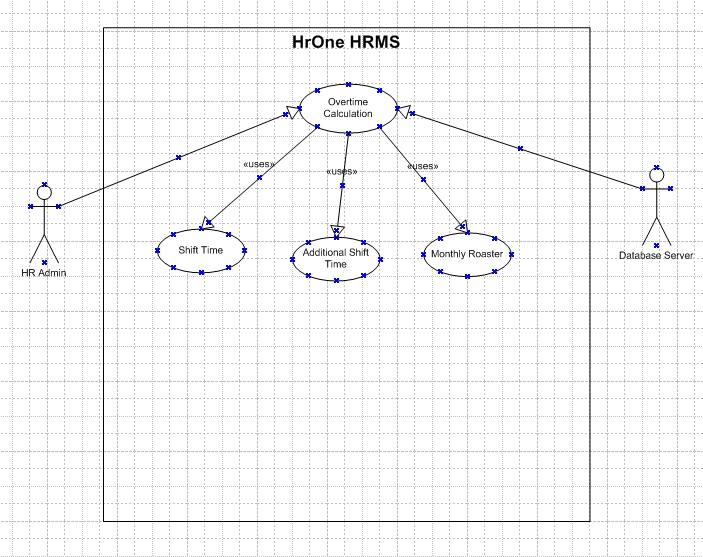


**12. Approve Leave Requests**

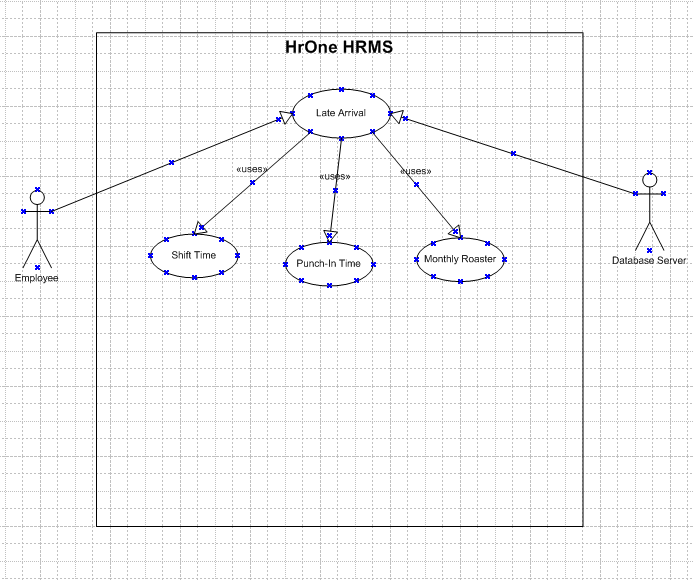




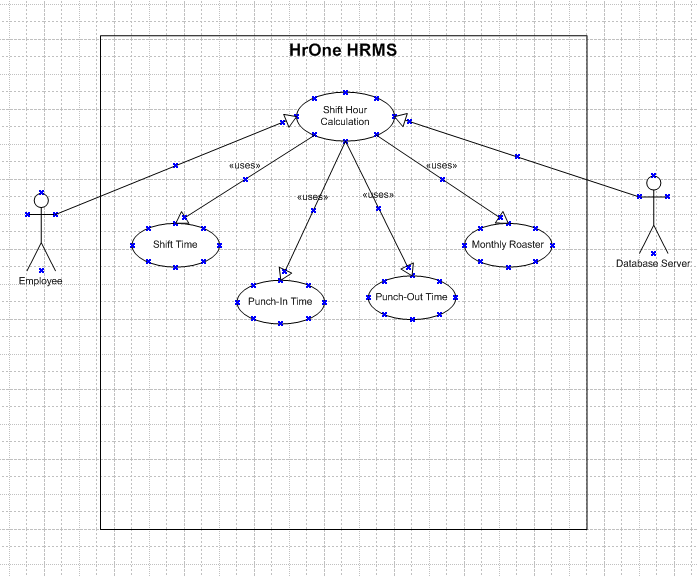
**13. Overtime Calculation**



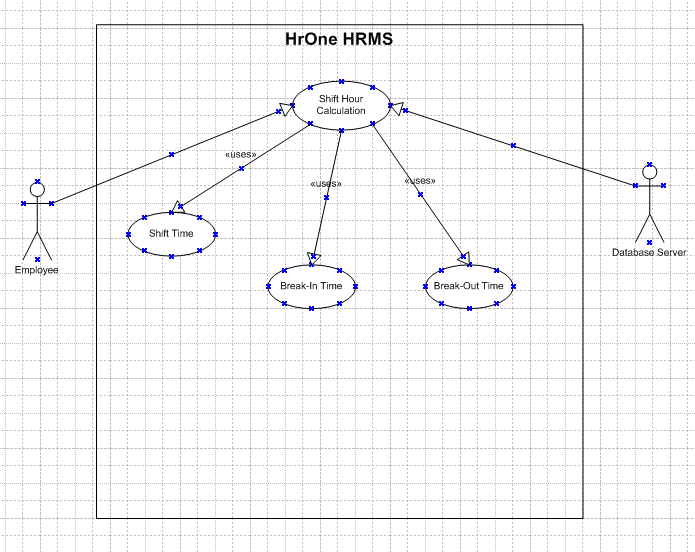
**14. Late Arrival Flagging**



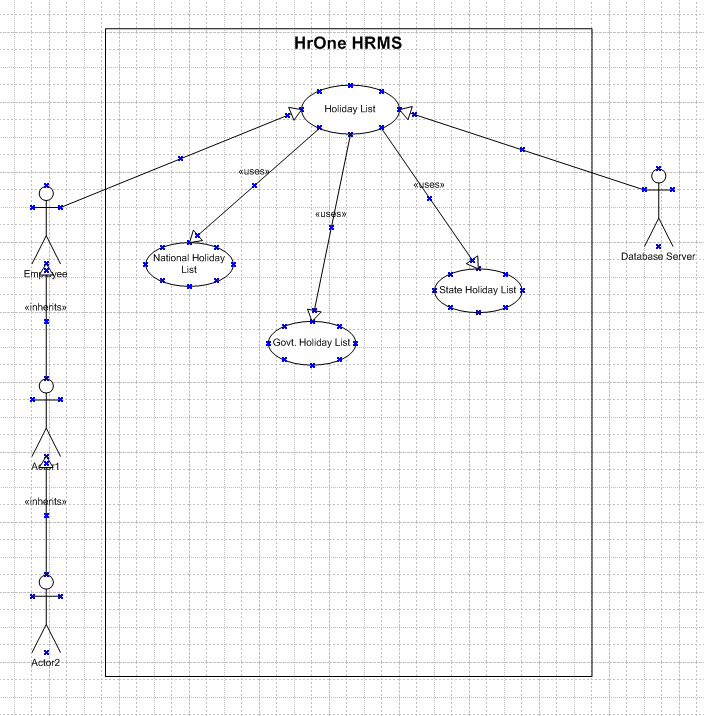
**15. Shift Hour Calculation**



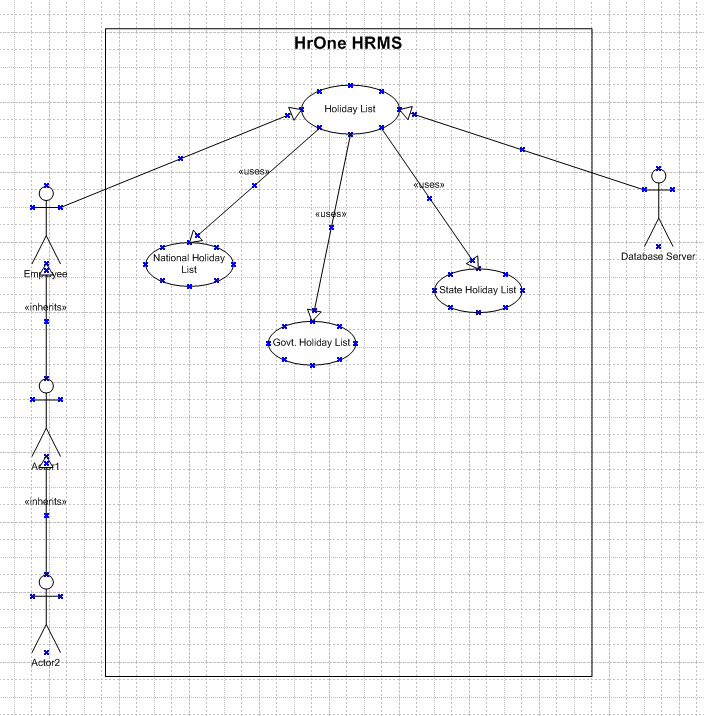
**16. Break Duration Calculation**



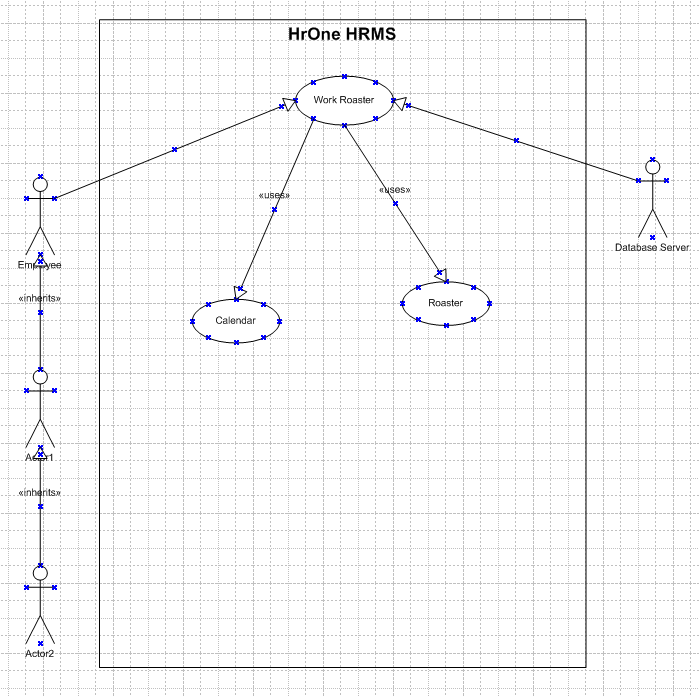
**17. View Yearly Holiday List**



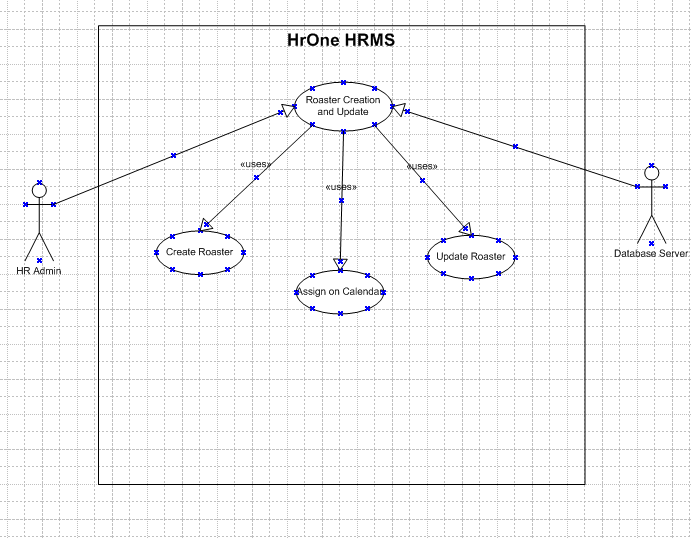
**18. View Holidays on Calendar**



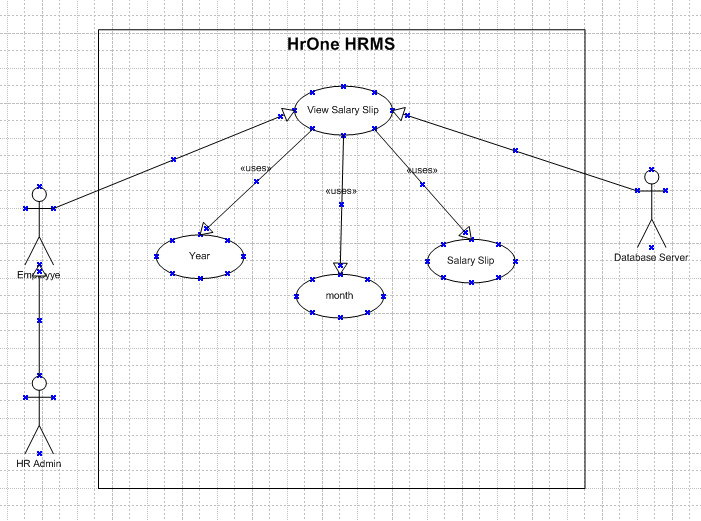
**19. View Work Rosters**



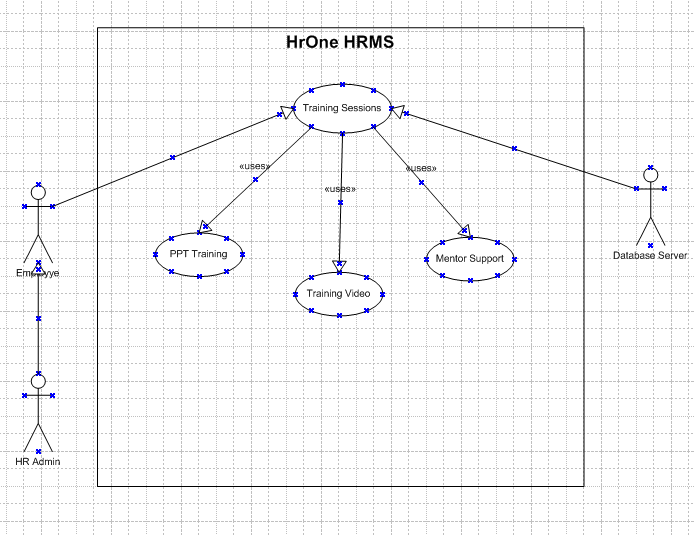
**20. Update & Assign Employee Rosters**



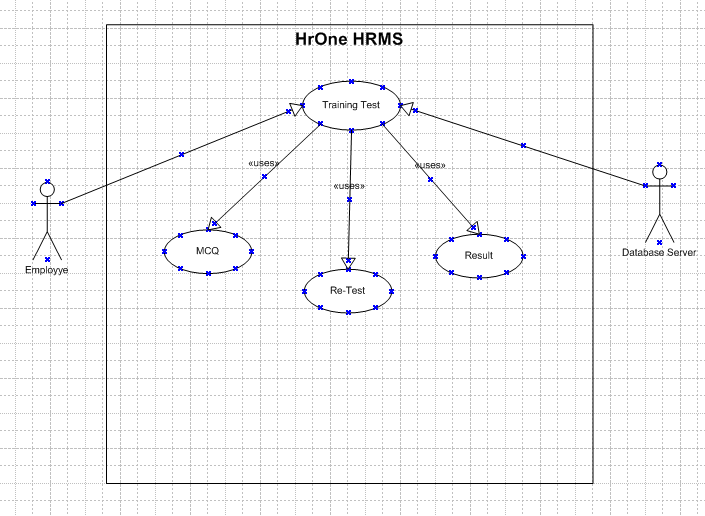
**21. View Monthly Salary Slips**



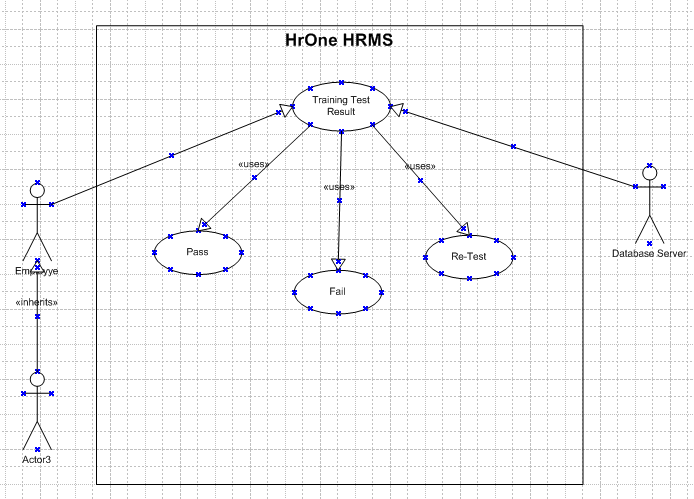
**22. Access Assigned Training Sessions**



**23. Start Training Tests**

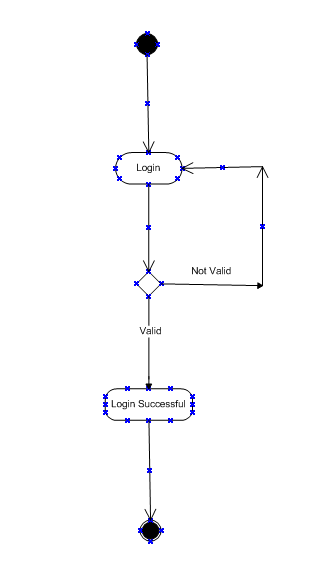


**24. View Training Test** **Results**

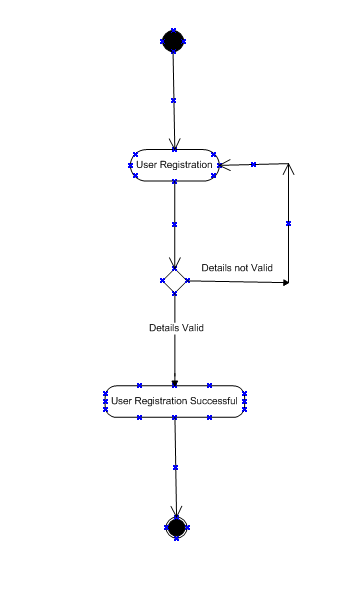


**Activity Diagram**

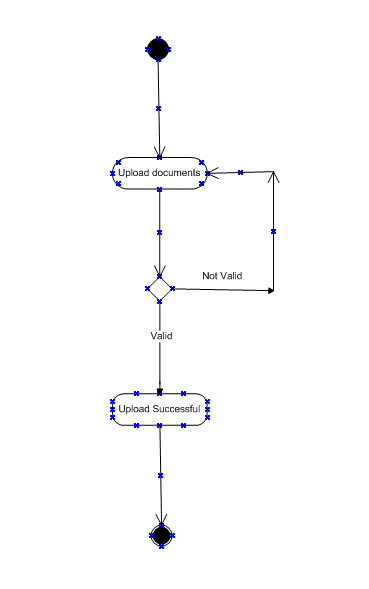
**1. Login**

****

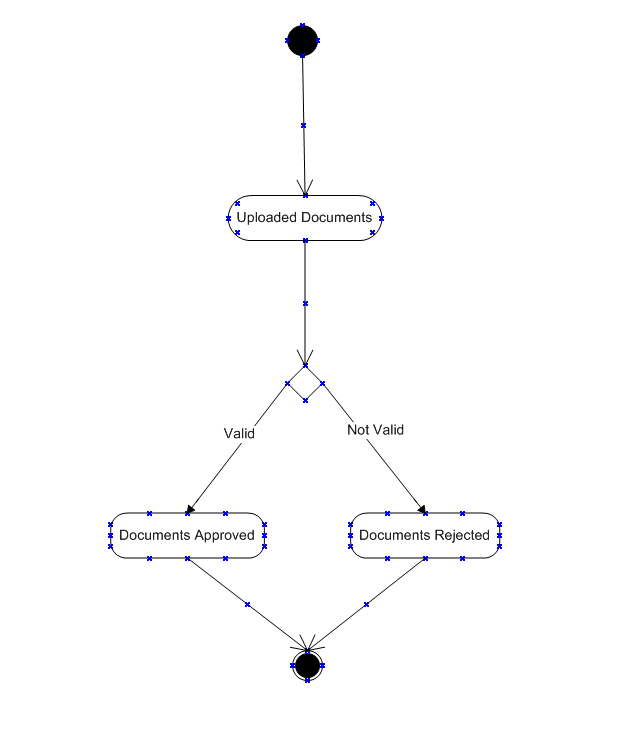
**2. Registration**

****

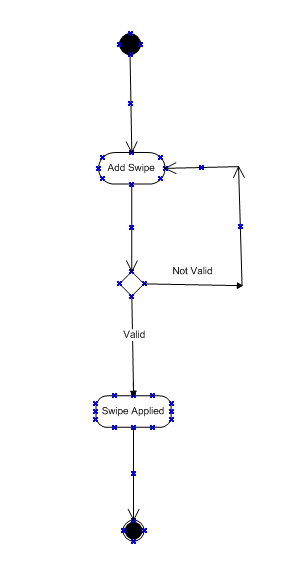
**3. Update documents**

****

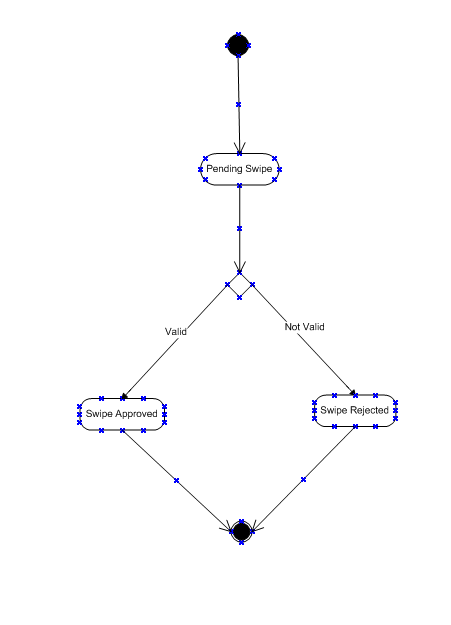
**4. Documents Approval**

****

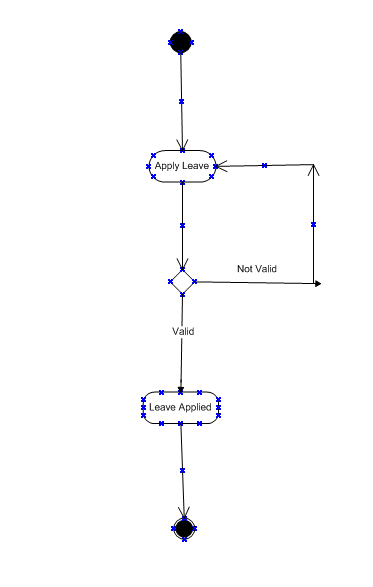
**5. Attendance Swipe Apply.**

****

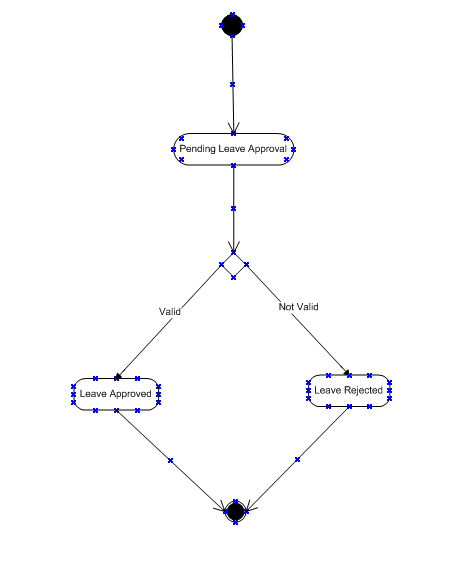
**6. Attendance Swipe Review.**

****

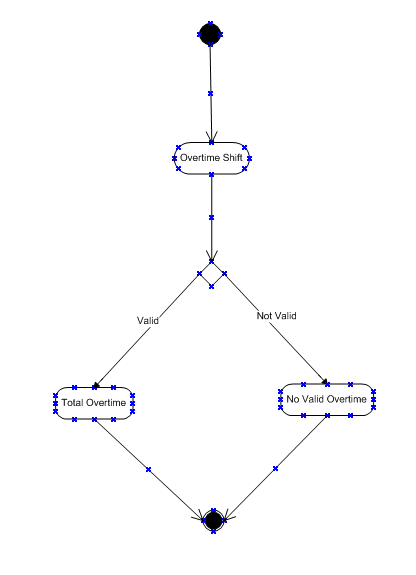
**7. Apply Leave.**

****

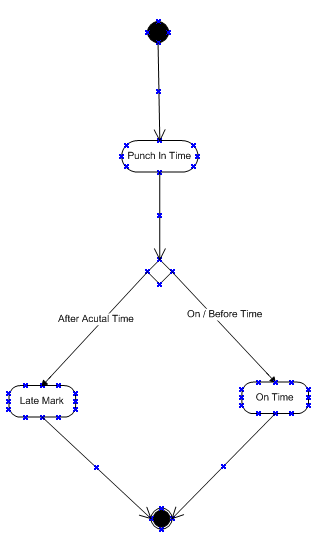
**8. Applied Leave Review.**

****

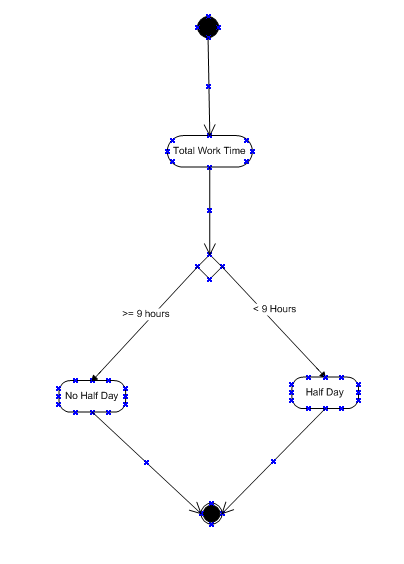
**9. Overtime Calculation.**

****

**10. Late Mark.**

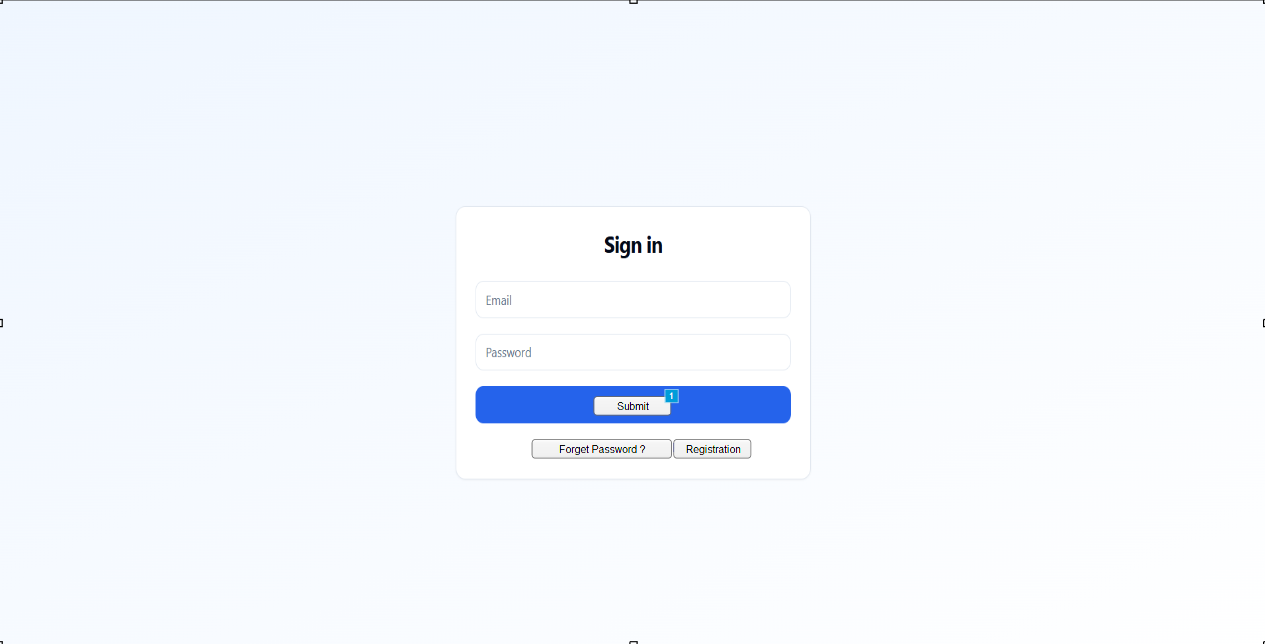
****

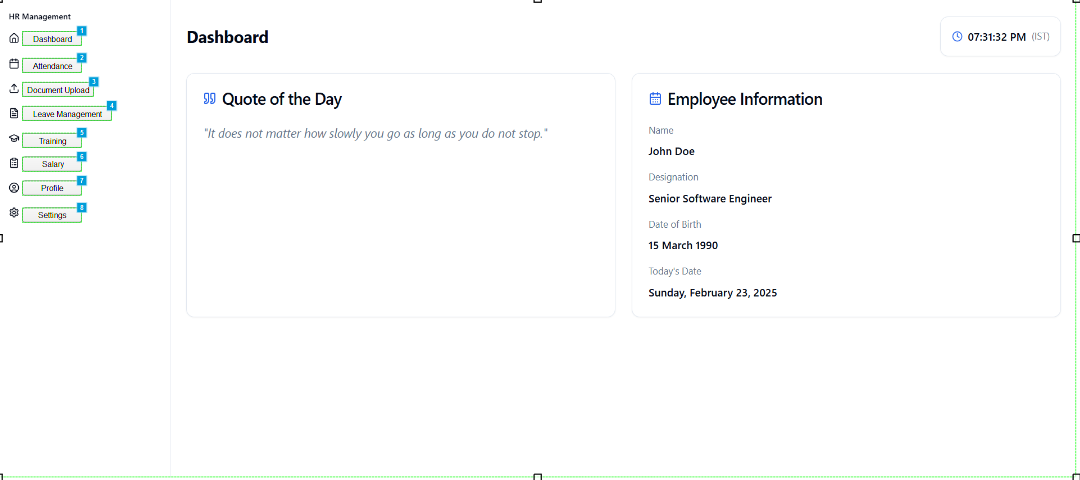
**11. Total Working Time.**

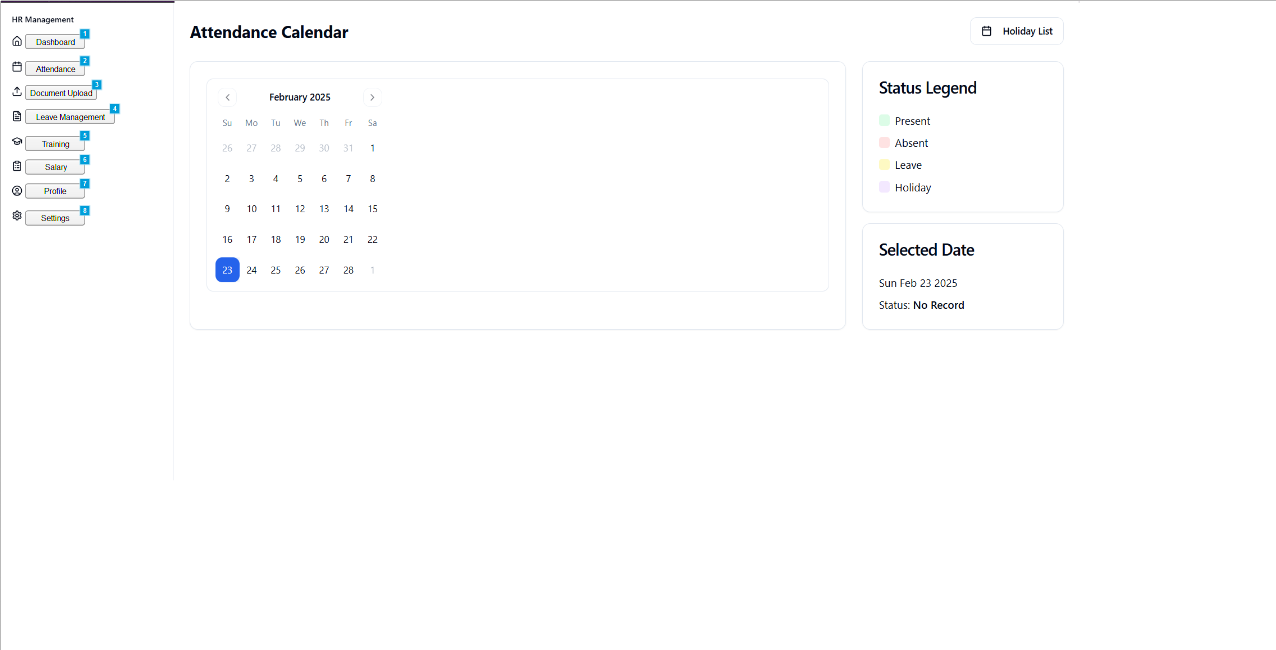
****

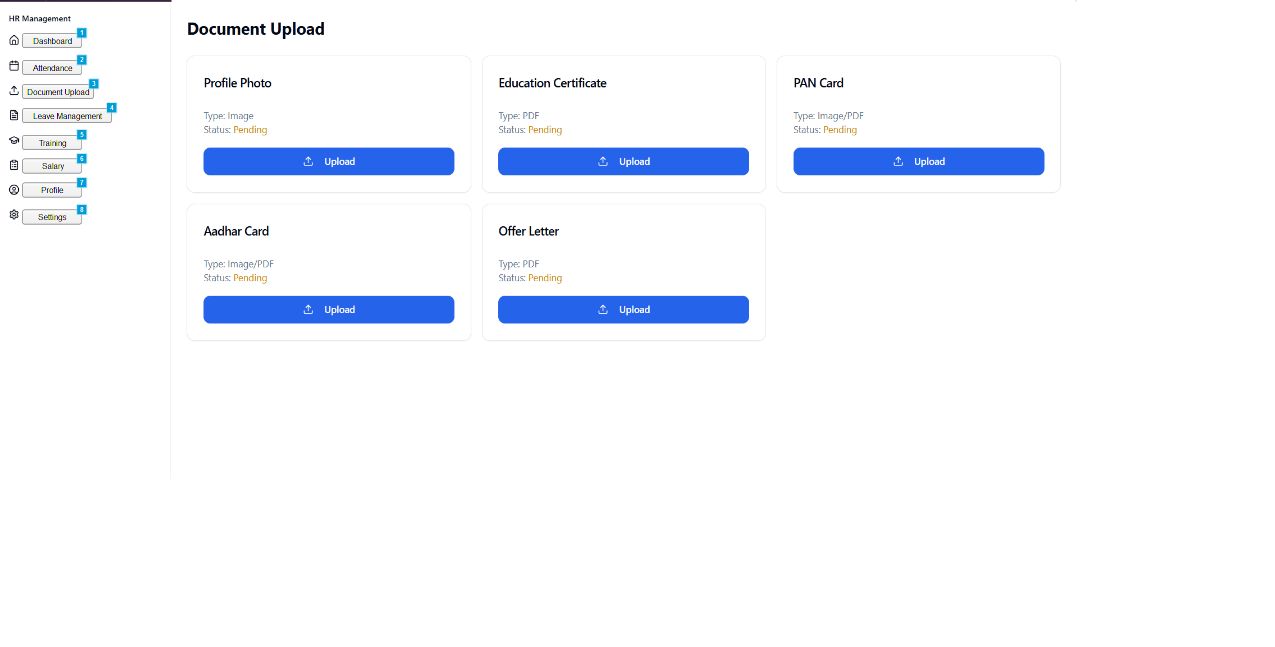
**Document 7- Screens and pages**

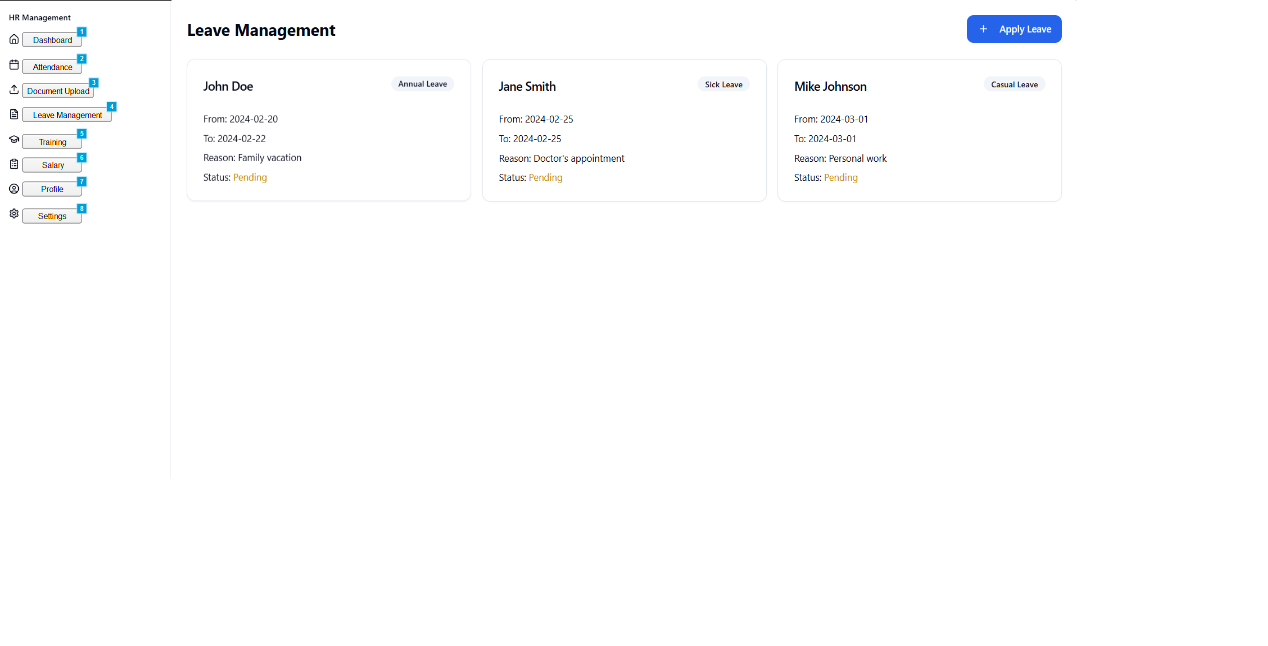
Screen pages.

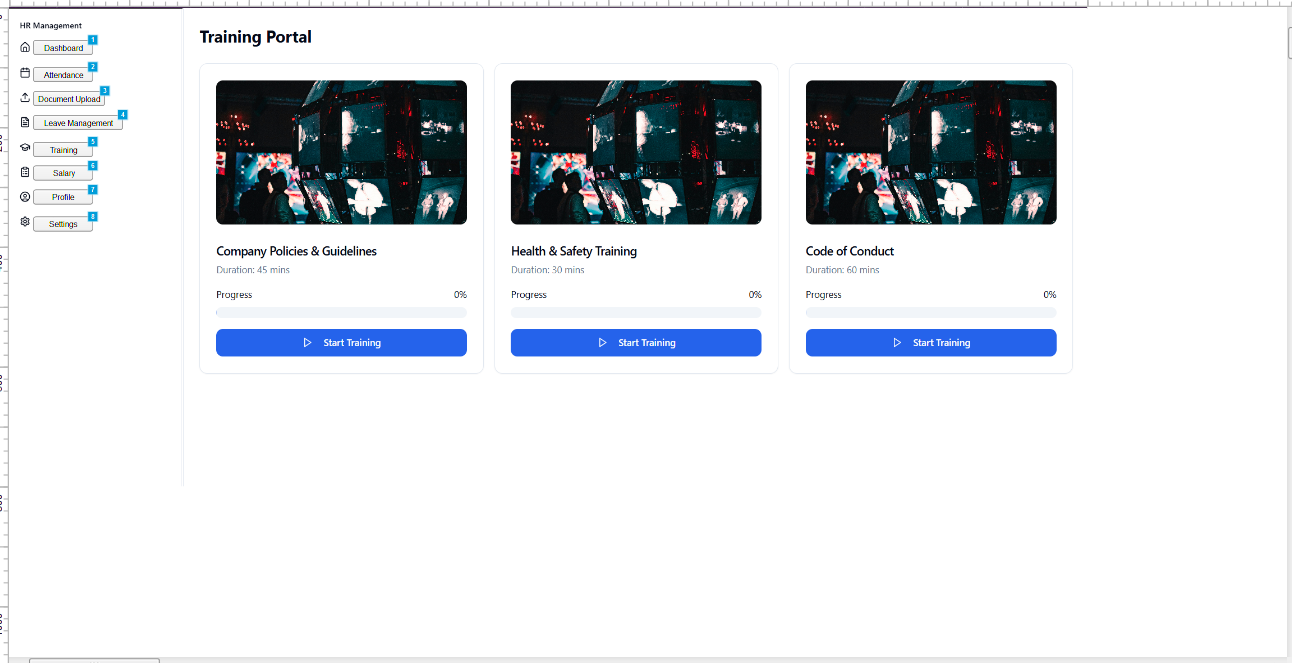


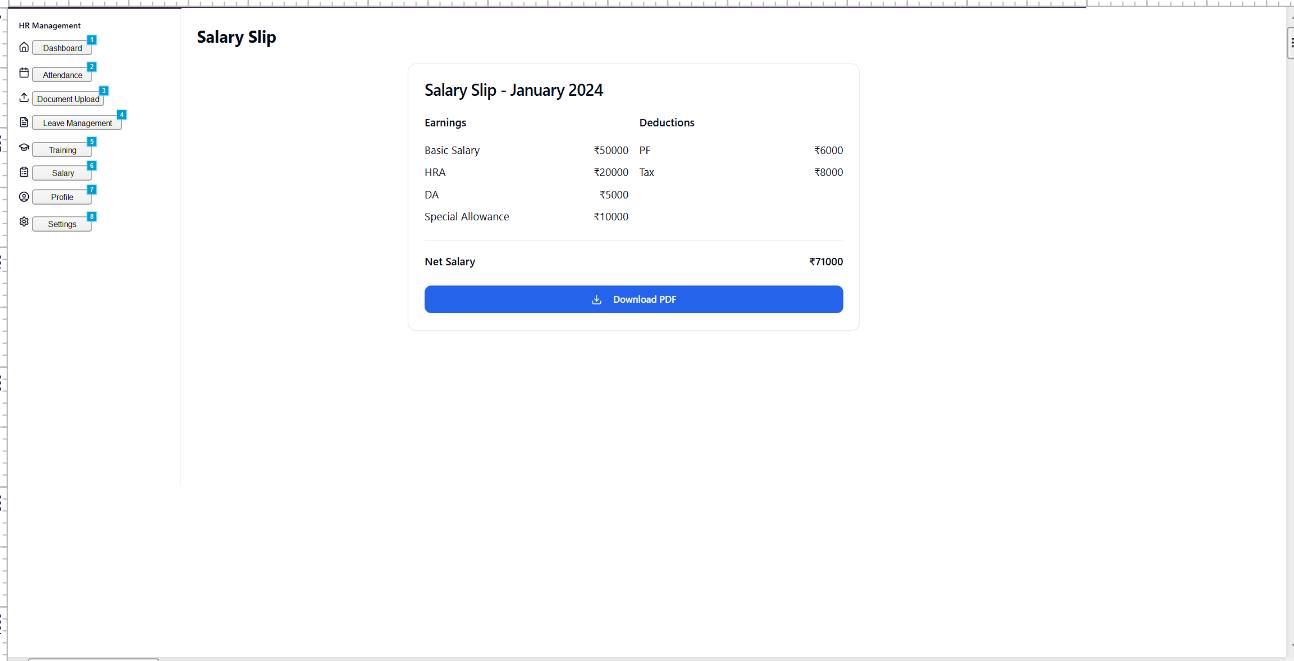


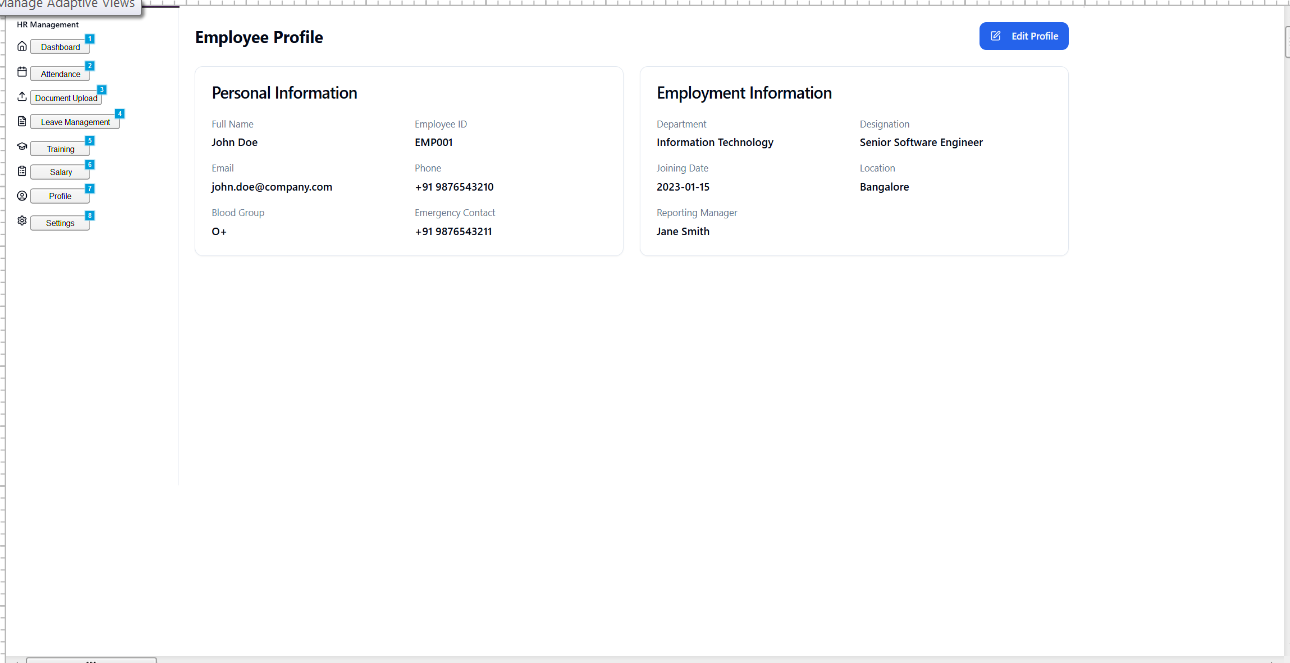


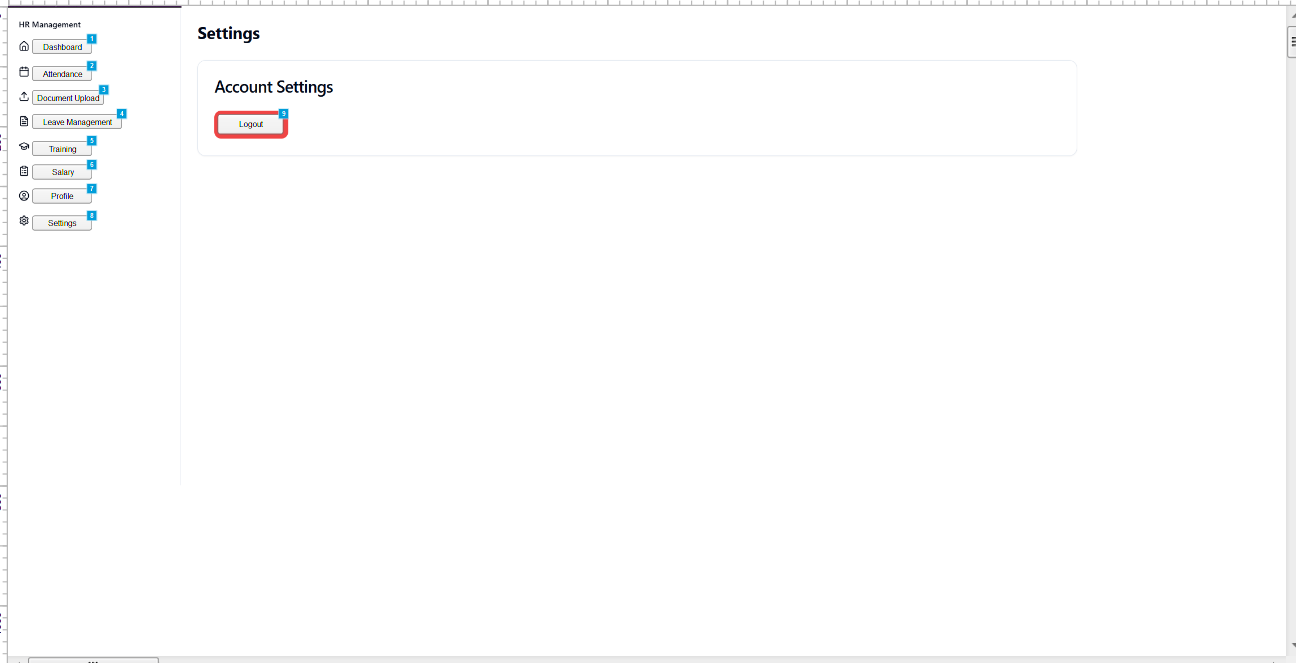












Please Open below link to see the sample working website  
  
<https://easy-hr-flow.lovable.app/login>

Please use ID and password:  
  
HR Admin: admin@hr.com / admin123  
Employee: user@hr.com / user123

**Document 8- Tools-Visio and Axure  
  
Microsoft Visio** is a powerful diagramming tool used to create professional and versatile visuals that simplify complex information. While working on my current project, I found it incredibly useful for transforming ideas and technical concepts into clear, easy-to-understand diagrams. Instead of explaining complex activities or processes step by step, Visio allows us to present them visually, making it easier for others to grasp the workflow efficiently. This enhances communication and helps in better decision-making.

**Axure RP** is a powerful tool for creating prototypes, wireframes, and interactive designs for websites and mobile applications. It simplifies the prototyping process by allowing seamless page alignment and interactive linking between screens, making previews highly engaging. This enables both the development team and clients to visualize the website’s basic functionality, ensuring a clear understanding of how the final product will work before development begins.

**Document 9- BA experience**

**My experience as BA in following phases:**

1. **Requirement gathering:**

* To gather requirement, we used Interview, Questionnaires and Prototype.
* SME were not available most of the time to get the details. So I utilize as mush time I can while meeting with SME.
* I validate the requirements using FURPS technique.
* Prototyping is used to give more specific requirements.

1. Requirement Analysis:

* I need to draw the UML diagrams to visually explain the requirements.
* Some of the points were disagreed by Developer Team, so we modified some pointers.
* Prepared BRD and FRD.
* Take Signoff from Client.

1. Design:

* Once I have prepared Use case diagrams, I prepared the test cases also.
* Communicated with HR Team to finalize the Design.
* Also prepared the both Positive Test case and Negative Test Case.
* Updating RTM so we make sure that all the requirements are met on time.

1. Development:

* Organized JAD sessions.
* Create the Health Environment so that developer Team can raise queries and as a BA I should solve them. By this we can save lot of time in coding once we start.
* Conducted regular meeting with team daily so we can solve any error or issues arise while developing / coding.
* Referred the prototype build by me so we don’t have to invest time in working on website.

1. Testing:

* Prepared Test case from use cases.
* Perform high level testing.
* Test data was gathered from existing employees.
* Updated RTM.
* Take Signoff from Client.
* Prepared UAT for Client.

1. Deployment:

* Shared RTM and project closure document.
* Organized training sessions.
* All HR members and HR Head attended the meetings.
* Project was fully functional after Training sessions.