**DataBase Project**

Employee Enrollment System

Table of contents:

1. User-Define Requirements
2. Functional Requirements
3. Non-Functional requirements
4. Entities and their attributes
5. Relationships
6. ERD Diagram
7. Database Coding
8. Front end Screens
9. Front-End & Back-End attachment

User-Define Requirements:

* Calculate and display working hours
* Shows status (active, offline, etc.)
* Show monthly salary add also suggest deductions or bonus based on working hours
* Emailing system

-email should be sent to every user in case of any action

* Application system (application for leave, early salary, loan/bonus demand)

-application status should be provided to user (sent, under review, approved, cancel)

-Application details from that user should be provided to admin (previous applications, leave restriction, etc.)

-warnings should be implemented in case of application demand exceed the default range. (Like more than 3 eaves in month)

* Transaction (access admin to his bank account to transfer salary)

Functional Requirements:

1. User Authentication and Authorization:

- The system should have user authentication to ensure that only authorized personnel can access and perform actions within the system.

- Different user roles (e.g., employees, managers, admin) should be defined with specific access permissions.

2. Employee Enrollment:

- The system should allow the admin to add and maintain employee records, including personal information, contact details, and job-related data.

3. Working Hours Calculation:

- The system should track and calculate working hours for each employee based on clock-in and clock-out times.

4. Status Display:

- Employees' status (active, offline, etc.) should be displayed based on their login/logout activities.

5. Salary Calculation:

- The system should calculate monthly salaries for employees based on their working hours and predefined pay rates.

- It should suggest deductions or bonuses based on working hours and predefined rules.

6. Emailing System:

- The system should send automated emails to users for various actions (e.g., salary crediting, application status updates, leave approvals).

- It should store a record of sent emails for auditing purposes.

7. Application System:

- Employees should be able to submit applications for leave, early salary, or bonus/loan demands.

- Users should be able to track the status of their applications (sent, under review, approved, canceled).

- Admin should have access to application details for each user, including previous applications and leave restrictions.

8. Warning System:

- The system should implement warnings when certain conditions are met, such as when an employee exceeds the allowed number of leaves in a month.

9. Transaction Handling:

- Admin should have access to a bank account management module to initiate salary transfers to employee bank accounts.

10. Reporting:

- The system should generate reports for various purposes, including payroll reports, application statistics, and employee attendance records.

11. Data Backup and Recovery: //can be non-functional

- Implement a data backup and recovery system to ensure data integrity and availability.

12. Logging and Auditing: : //can be non-functional

- The system should maintain logs of user actions and system events for auditing and troubleshooting.

13. User Interface:

- Design a user-friendly interface for employees, managers, and admin to interact with the system efficiently.

14. Error Handling:

- Implement error handling mechanisms to gracefully handle system errors and provide informative error messages.

Non-Functional Requirements:

Non-functional requirements are essential aspects of a software project that describe how the system should perform rather than what it should do. They often define the quality attributes, constraints, and performance characteristics of the system. In the context Employee Enrollment System database project, here are some non-functional requirements to consider:

1. **Performance:**

* The system should respond to user interactions promptly, with low latency.
* It should handle a specified number of concurrent users without significant performance degradation.
* Database queries should execute efficiently, even with large datasets.

2. **Scalability:**

* The system should be designed to scale horizontally or vertically to accommodate future growth in terms of users and data volume.
* Scalability should not disrupt the system's availability or performance.

3. **Reliability:**

* The system should be highly reliable, with minimal downtime for maintenance or upgrades.
* It should handle errors gracefully and provide error messages that are informative to users and administrators.

4. **Security:**

* User data, including personal and financial information, should be stored securely and encrypted.
* Access to sensitive data and system functionality should be protected by user authentication and authorization mechanisms.
* The system should comply with relevant data privacy regulations and standards.

5. **Availability:**

* The system should be available 24/7, with minimal planned downtime for maintenance.
* Redundancy and failover mechanisms should be in place to ensure continuous availability.

6. Usability:

* The user interface should be intuitive and user-friendly, requiring minimal training for users to navigate and perform tasks.
* Accessibility features should be implemented to ensure that the system can be used by individuals with disabilities.

7. **Compatibility:**

* The system should be compatible with a range of web browsers and devices to ensure a broad user base.
* It should also be compatible with various email clients for the emailing system.

8. **Scalability:**

* The system should be designed to handle future increases in the number of employees and user accounts.
* Database and storage systems should scale to accommodate growing data volumes.

9. **Auditability and Logging:**

* The system should maintain detailed logs of user activities, including login attempts, data changes, and application submissions.
* Audit logs should be securely stored and accessible only to authorized administrators.

10. **Backup and Recovery:**

* Regular automated backups of the database and system configuration should be performed.
* There should be a well-defined disaster recovery plan in place to minimize data loss in case of system failures.

11. **Compliance:**

* Ensure that the system complies with relevant industry standards and regulations, especially those related to financial data and employee information.

12. **Documentation:**

* Comprehensive documentation should be provided for both users and administrators, including user manuals, system architecture diagrams, and API documentation.

13. **Load Testing:**

* The system should undergo load testinig to ensure it can handle peak usage without performance degradation or crashes.

14. **Maintainability:**

* The codebase should be well-structured and well-documented to facilitate future maintenance and updates.
* Changes and updates should be easy to implement without causing system instability.

15. **Cost and Resource Constraints:**

* The project should adhere to budget constraints, and resource usage (e.g., server resources) should be optimized to minimize operational costs.

These non-functional requirements are crucial to ensure that your Employee Enrollment System is not only functional but also performs well, is secure, and meets the expectations of users and administrators.

|  |
| --- |
| Working Time |
| Log-In Time |
| Log-Out Time |
| Arriving Status |
| Departure Status |
| Extra Start |
| Extra End |

Entities & their Attributes:

|  |
| --- |
| Employee |
| Employee ID |
| Employee first Name |
| Employee Last Name |
| Date of Joining |
| Employee CNIC |
| Employee Date of Birth |
| Employee Phone Number |
| Employee Gender |
| Employee Phone |
| Employee Bank\_ID |

|  |
| --- |
| Position |
| Position Id |
| Position Name |
| Salary Amount |
| Employee\_ID |

|  |
| --- |
| Office Hours |
| From\_Date |
| To\_Date |
| OfficeTime\_Start |
| OfficeTime\_End |

|  |
| --- |
| Application Status |
| Accepted |
| Denied |
| Under Review |
| Application ID |
| Employee ID |

|  |
| --- |
| Application/Request Form |
| Application ID |
| Application Type |
| Reason |
| Note |
| Apply Date |
| From Date |
| To Date |
| No\_of Applictions |
| Employee ID |

|  |
| --- |
| Projects |
| Project\_Id |
| Project Name |
| Client |
| From\_Date |
| To\_Date |
| Payment |
| No\_Projects |

Relationship:

ERD Diagram:

Data Base Coding:

Front-end Screens:

Front end and Backend Attachment: