**EMPLOYEE REFERAL SYSTEM**

A Low-Level Design (LLD) document provides a detailed view of the system components, data structures, and interactions. Here’s a general structure for an Employee Referral Management System LLD:

**1. Introduction**

**Purpose:** Describe the purpose of the system, its functionalities, and how it supports employee referrals.

**Scope:** Outline what the LLD covers and any limitations.

**2. Requirements Gathering**

**Stakeholder Meetings:** Gather requirements from stakeholders, including HR, IT, and potential users.

**Requirement Analysis:** Analyze and document functional and non-functional requirements.

Use Case Development: Create use cases and user stories to guide development.

**3. Technology Stack Selection**

**Frontend Technologies:** Choose technologies for the user interface (e.g., React).

**Backend Technologies:** Select technologies for server-side development (e.g., .NET).

**Database:** Decide on the database system (e.g., SQL Server).

**Other Tools:** Identify additional tools and services needed (e.g., notification services).

**4. System Overview**

**4.1 Architecture Diagram:** Include a diagram showing the system’s architecture, including all major components like frontend, backend, database, and external services.

The architecture diagram below illustrates the high-level components and their interactions within the Employee Referral Management System:

+----------------------+ +-------------------+ +--------------------+

| | | | | |

| | HTTP/S | | HTTP/S | |

| Frontend +<--------->+ Backend +<---------> + Database |

| (React, Redux) | | (.NET API) | | (SQL Server) |

| | | | | |

+--------+-------------+ +--------+----------+ +--------+---------+

| | |

| Web Sockets | REST API | SQL Queries

| | |

+--------+----------+ +--------+----------------+ +--------+----------+

| | | | | |

| Notification | | Authentication & | | |

| Service | | Authorization | | |

| (Email, SMS) | | (JWT, OAuth) | | |

| | | | | |

+-------------------+ +--------------------------+ +-------------------+

**5. Detailed Design**

**5.1 Frontend (React, Redux)**

**User Interface (UI):**

**Referral Form:** Allows employees to submit referrals with necessary details.

**Dashboard:** Displays referral status, history, and other relevant metrics.

**Profile Management:** Enables users to manage their profiles and view their referral history.

**5.2 State Management (Redux):**

**Actions & Reducers:** Handle user actions (e.g., submitting a referral) and update the application state.

**Middleware:** Manages asynchronous operations like API calls.

**5.3 Communication with Backend:**

**REST API Calls:** Fetch and update data through API endpoints.

**Web Sockets:** For real-time updates and notifications.

**5.4 Backend (.NET API)**

**API Endpoints Example:**

**Referral Management:** Endpoints for creating, updating, and retrieving referrals.

**User Management:** Endpoints for user registration, authentication, and profile management.

**Notification Management:** Endpoints for sending notifications to users.

**5.5 Communication with Database:**

**Data Access Layer:** Handles database interactions through ORM (e.g., Entity Framework).

**6. Security Considerations**

**6.1 Authentication & Authorization:** Securely authenticate users and manage their sessions. Describe how users will be authenticated and authorized (e.g., JWT, OAuth).

**6.2 Data Encryption:** Outline how sensitive data will be encrypted both in transit and at rest.

**6.3 Access Controls:** Define access control mechanisms for different user roles.

**6.3.1** **Role-Based Access Control (RBAC):** Define user roles and permissions to restrict access to certain functionalities.

**7. Error Handling and Logging**

**7.1** **Error Management:** Detail how errors will be logged and handled, including retry mechanisms and fallback procedures.

**7.2 Logging Strategy:** Define what information will be logged and how logs will be managed.

**8. Notification Service**

**8.1 Email Service:** Sends email notifications for referral status updates and other alerts.

**8.2 SMS Service:** Sends SMS notifications if configured.

**9. Testing Strategy**

**9.1 Unit Testing:** Outline the approach for unit testing frontend and backend components.

**9.2 Integration Testing:** Describe how different components will be tested together.

**9.3 End-to-End Testing:** Define how the system will be tested as a whole.

**10. Deployment**

**10.1** **Deployment Architecture:** Describe the deployment architecture (e.g., cloud services, CI/CD pipelines).

**10.2 Environment Setup:** Detail how different environments (development, staging, production) will be set up.

**11. Maintenance and Support**

**11.1 Update Procedures:** Outline procedures for system updates and patching.

**11.2** **Support Mechanisms:** Provide ongoing support, handle bug fixes, and perform regular updates.

**11.3** **Feedback Collection:** Gather feedback from users to identify areas for improvement.

**11.4** **Monitoring:** Implement monitoring tools to track system performance and detect issues.

**12. Documentation**

**12.1** **User Documentation:** Create user manuals and help guides.

**12.2 Technical Documentation:** Document system architecture, API endpoints, and deployment procedures.

**13. Training**

**13.1 End-User Training:** Train users on how to use the system effectively.

**13.2 Admin Training:** Train administrators on system management and troubleshooting.

**14. Appendices**

**14.1 Glossary:** Define terms used in the document.

**14.2 References:** List any references or related documentation.

**END**