PROJECT REPORT - FINAL
PulseHR Employee Management System

1. Problem Statement:

The project's goal is to solve the difficulties small and medium-sized businesses (SMEs) have in effectively managing their human resources. Current employee management systems (EMS) are frequently too expensive or don't allow for the customization required to meet particularly business needs. This leaves a gap that makes it more difficult to implement HR procedures that are more efficient, such as payroll, timesheet management, performance reviews, onboarding, leave monitoring, and reporting.

2. Goals:

• Create an effective, adaptable, and scalable employee management system.
• Simplify HR procedures to boost productivity and cut down on manual labor.
• Offer an affordable option that is available to companies of all sizes.
• Boost user experience with a UI that is responsive and dynamic.

3. Alternative Approaches:

• Invest in and modify an already-existing EMS system: This could be expensive and might not completely satisfy each company's unique requirements.
• Create a unique solution from the ground up without relying on pre-existing frameworks: This method might take a while, and it might not have all the advantages of well-known frameworks like React.js and Spring Boot.

4. Chosen Approach:

Backend: Create RESTful APIs, connect databases, and implement business logic using Spring Boot.
Frontend: For a dynamic and responsive design, create the user interface with React.js.
Database: To store employee records, use a relational database (MySQL or PostgreSQL).

5. Justification:

Cost-effective Customization: React.js and Spring Boot strike a compromise between development efficiency and customization. They make customized solutions possible without the exorbitant prices of certain commercial EMS platforms.

Reliable Frameworks: Spring Boot streamlines backend development by effectively managing the creation of APIs and business logic. A responsive and contemporary user interface is guaranteed by React.js, which enhances the user experience.

Support from the Community: React.js and Spring Boot both have sizable and vibrant communities that offer a wealth of plugins, resources, and assistance with updates and troubleshooting.

Scalability: The EMS can expand with the company without sacrificing performance thanks to the selected technologies' well-known scalability.

Agile Development Methodology: Using this approach guarantees iterative development, testing, and deployment. This strategy lowers the possibility of significant setbacks by enabling ongoing improvement based on user feedback.