**Day 2 – Finalizing Tech Stack and Approach**

**Project Scope Recap**

* Two roles: Admin and Employee
* Features: Employee Management, Salary Calculation, Salary Slip Generation
* Backend + Frontend

**Tech Stack**

**Backend**

|  |  |
| --- | --- |
| **Spring Boot (Java)** | Production-ready, fast development, great for REST APIs |
| **MySQL** | Lightweight, easy to manage, great for relational data like employees and salaries |
| **Hibernate / JPA** | ORM for smooth DB interaction with Java objects |
| **Maven** | Dependency management |

**Frontend**

|  |  |
| --- | --- |
| **React JS** | Component-based, fast, great community support |
| **Axios** | API calls to backend |
| **React Router** | Role-based routing (Admin vs Employee) |

**Tools**

|  |  |
| --- | --- |
| **MySQL Workbench** | GUI for DB |
| **VS Code & Eclipse** | IDE |
| **Git + GitHub** | Version control |

**Project Architecture**

**Approach**

**1. Authentication & Authorization**

* Admin and Employee log in via a common login endpoint
* Checks the data in the particular table.
* If data found redirected to dashboard and login details stored in local storage

**2. Employee Management (Admin only)**

* REST endpoints:
  + POST /admin/addEmployees – add employee
  + GET /admin/allEmployees – list all
  + PUT /admin/updateEmployees/{id} – update
  + DELETE /admin/deleteEmployees/{id} – delete

**3. Salary Calculation**

* Endpoint: POST /admin/salary/{employeeId}
* Server calculates Gross, Tax based on Indian tax regime (as of FY 2023-24), Net Salary

**4. Salary Slip Generation**

* POST /admin/salary-slip/{employeeId}
* Data stored in salary\_slips table
* Format response as PDF

**5. Employee Views Salary Slip**

* GET /employee/salary-slip – shows only their own slip(s)
* Token used to identify logged-in employee

**Summary**

|  |  |
| --- | --- |
| Backend | Spring Boot + MySQL + Hibernate |
| Frontend | ReactJS + Axios |
| Tools | GitHub, MySQL Workbench |