Technical Assessment - Final (4 Hours)

Design and develop a **secure, role-based Employee & Project Management System** using **Spring Boot**, where a **REST API is consumed by a separate Web/UI layer** with **JWT authentication**, **claims-based authorization**, and **Hibernate (Database-First approach)**.

Mandatory Architecture

Web Application / Frontend
|
| JWT Token (Authorization: Bearer)
v
Spring Boot REST API
|
v
MySQL Database (Hibernate – Database First)

Technologies to Use (Strict)

- Spring Boot (REST API)
- Hibernate / JPA (Database-First)
- JWT Authentication
- Custom Authorization Filter (NOT annotations only)
- Claims-based Security
- SQL Database (MySQL / PostgreSQL / SQL Server)

Database Design (Given – Do NOT Modify)

Only **table definitions** are provided. No entity code.

Required Tables

1. **users**
    - user_id
    - username
    - password_hash
    - role (ADMIN / MANAGER / EMPLOYEE)
    - is_active
2. **employees**
    - employee_id
    - name
    - email
    - department_id
    - created_by (user_id)
3. **projects**
    - project_id
    - project_name
    - start_date
    - end_date
    - created_by (user_id)
4. **employee_projects**
    - employee_id
    - project_id
    - assigned_date

Technical Assessment - Final (4 Hours)

📌 **Important Constraints**

- Use **Hibernate reverse engineering** or JPA mapping
- ✕ No schema auto-generation
- ✕ No ddl-auto=create/update

Authentication Requirements (JWT)

1. Create a **Login REST API**:
    - Accepts username & password
    - Validates credentials from database
    - Generates JWT Token
2. JWT Token **must include claims**:
    - userId
    - username
    - role

🚫 Requests without required claims must be rejected

Authorization Rules (Custom Filter – Mandatory)

✕ Do NOT rely only on @PreAuthorize or @Secured

✓ Implement a **custom authorization filter**

Role Access Rules

Role	Permissions
ADMIN	Full access
MANAGER	Assign Project to Employee
EMPLOYEE	View only assigned projects

📌 Filter Responsibilities:

- Extract JWT from header
- Validate token
- Read claims
- Decide access dynamically
- Return **403 Forbidden** when unauthorized

REST API Requirements

Develop secured APIs for:

- Employee Management
- Project Management
- Assign Employee to Project
- View Assigned Project

Mandatory Rules

- All APIs must be protected
- No API should work without JWT
- Standardized error response format

Web/UI Layer Requirements

(UI technology of your choice)

1. User Registration

Technical Assessment - Final (4 Hours)

2. Login Page
    - Calls Login API
    - Stores JWT securely
3. After Login:
    - Token must be sent in every API call
    - UI elements must render based on **role claim**
4. Pages:
    - DashBoard
    - Employee List
    - Project List
    - Assign Employee to Project

    (Both Project and Employee Two dropdownlists and assigned date is the current date)
    - View Assigned Project

📌 **Direct DB access from UI is strictly forbidden**

▯ Advanced Security Constraints (Anti–ChatGPT)

1. ✕ No default Spring Security JWT samples
2. ✕ No hardcoded roles in controllers
3. ✕ No authorization logic in controllers
4. ✓ Authorization must happen in **filter layer**
5. ✓ Claims must be dynamically extracted
6. ✓ JWT expiry and invalid token handling required