# Company Management System – Back End

## 1. Overview

In this task, you are required to develop a Company Management System that encompasses various features for managing companies, departments, employees, and projects. The system will allow users to create, read, update, and delete (CRUD) records for each of these entities. Additionally, there will be a workflow for handling the performance review cycle of employees and implementing role-based access control to ensure secure data handling. The system should adhere to the requirements specified below to ensure all features are implemented correctly.

While there are mandatory requirements that must be completed, there are also bonus requirements. The bonus requirements are optional and should only be implemented if you have extra time and it does not compromise the quality of the mandatory requirements. Achieving the bonus requirements is not essential but will be considered a plus.

This task is designed to evaluate your technical skills, problem-solving abilities, and attention to detail. It also assesses your capability to work under stress and tight schedules, reflecting real-world scenarios where meeting deadlines is crucial.

## 2. Requirements

- **Data Models:**

  ***User Accounts***
  - Username
  - Email Address (Login ID)
  - Role

  ***Company***
  - Company Name
  - Number of Departments (auto-calculated)
  - Number of Employees (auto-calculated)
  - Number of Projects (auto-calculated)

  ***Department***
  - Company (Select)
  - Department Name
  - Number of Employees (auto-calculated)
  - Number of Projects (auto-calculated)

  ***Employee***
  - Company (Select)
  - Department (Select)
  - Employee Name
  - Email Address
  - Mobile Number
  - Address
  - Designation (Position/Title)
  - Hired On (optional)
  - Days Employed (auto-calculated; only if hired)

  ***Project***
  - Company (Select)

- Department (Select)
- Project Name
- Description
- Start Date
- End Date
- Assigned Employees (Multi-Select)

## 3. Workflow: Employee Performance Review Cycle

Develop a workflow to manage the Employee Performance Review Cycle, ensuring a structured process for scheduling, feedback collection, and approvals.

### *Stages and Allowed Transitions*

#### *Stages*

- Pending Review: Employee is flagged for performance review.
- Review Scheduled: A review meeting has been scheduled.
- Feedback Provided: Feedback from the review meeting has been documented.
- Under Approval: Feedback is under managerial review.
- Review Approved: Performance review is finalized and approved.
- Review Rejected: Feedback is rejected and requires rework.

#### *Transitions*

- Pending Review → Review Scheduled: Triggered when the review date is confirmed.
- Review Scheduled → Feedback Provided: Triggered when feedback is recorded after the review meeting.
- Feedback Provided → Under Approval: Triggered when feedback is submitted for managerial review.
- Under Approval → Review Approved: Triggered when the manager approves the feedback.
- Under Approval → Review Rejected: Triggered when the manager rejects the feedback.
- Review Rejected → Feedback Provided: Triggered when the feedback is updated after rejection.



## 4. Security & Permissions:

a. Implement role-based access control to ensure:
   i. Only authorized personnel can view and edit data.
   ii. Different roles have different levels of access (e.g., Admin, Manager, Employee).

b. Use secure authentication and authorization mechanisms of your choice. (e.g., Sessions, Tokens).

## 5. APIs:

a. Create a RESTful API that supports all CRUD operations for all models:

i. **Company**

- GET: Retrieve a single company or list all companies

ii. **Department**

- GET: Retrieve a single department or list all departments

iii. **Employee**

- POST: Create a new employee
- GET: Retrieve a single employee or list all employees
- PATCH: Update an existing employee
- DELETE: Delete an employee

iv. **Project (Bonus)**

- POST: Create a new project
- GET: Retrieve a single project or list all projects
- PATCH: Update an existing project
- DELETE: Delete a project

b. Ensure the API handles data securely

c. Ensure the API follows RESTful conventions (e.g., using proper HTTP methods like GET, POST, etc.)

d. If applicable, provide clear documentation on the API endpoints, parameters and expected responses



## 6. Testing:

a. Include unit tests to validate individual components and functions.

b. Include integration tests to ensure different parts of the application work together correctly.

## 7. Logging (Bonus):

a. Implement logging to track application behavior and capture errors.

b. Ensure logs are detailed enough to troubleshoot issues but do not expose sensitive information.

# Task Submission:

Applicants are required to submit their tasks via a dedicated public GitHub repository through the submission form shared with you in the email. Follow the steps below to ensure a seamless delivery:

1. **GitHub Repository Creation:**
    - Create a public GitHub repository
2. **Repository Structure:**
    - Organize your repository with clear folder structure for different aspects of the task
3. **Code Submission:**
    - Commit your application code to the designated repository, ensuring that it includes all necessary files and directories.
4. **Documentation:**
    - Include a detailed README.md file containing the following:
        1. Comprehensive documentation detailing your approach, implementation details and any considerations made during the development process.
        2. Instructions on how to set up and run your application. Include any prerequisites, installation steps, and additional information required for a smooth evaluation.
        3. Clearly indicate the completion of each task (e.g., Check List). Mention any assumptions or considerations made during the implementation.
        4. If applicable, specify the security measures implemented, especially concerning role-based access control and data protection.
        5. If applicable, provide clear documentation on the API endpoints, parameters and expected responses or instructions on how to access the API documentation externally.

5. **Bonus Requirements:**

    - If you choose to implement any of the bonus requirements, ensure that it does not affect the quality and completion of the mandatory requirements. The bonus requirements are optional and should only be done if you have additional time and wish to showcase extra skills. Achieving these will be considered a plus during the evaluation.

6. **Deadline & Completion:**

    - Even if you are unable to complete all the requirements, it is crucial to submit your task by the specified deadline. Completion of all requirements is important, but submitting on time is a crucial part of the evaluation process and not completing all requirements does not mean failing. Make sure to indicate which parts are done and which are not in the README file to help us understand your progress.

7. **Submission:**

    - Share the GitHub repository link through the submission form shared with you in the email