Overview of Repo

This repository contains an Employee Management System built using Node.js, Express.js, and PostgreSQL. It includes various models and routes to manage employee data effectively.

GIT Repo: https://gitrepo.valuelabs.com/nishant.malhotra/employeemanagementnodejs

Assignment 1: **Implementing Advanced Search and Filtering Capabilities**

Objective: Enhance the search and filtering capabilities of the employee management system to allow users to find employees based on multiple attributes such as name, department, title, and date range.

Steps:

1. Update the `employeeRoutes.js` file:

- Add a new route `GET /employees/search` that accepts query parameters for filtering and searching employees.

2. Implement the search logic:

- Use Sequelize's querying capabilities to filter employees based on the provided query parameters.

- Allow filtering by:

  - `first_name`

  - `last_name`

  - `department_name` (from the `Department` model)

  - `title` (from the `Title` model)

  - Date range (`hire_date`, `promotion_date`, etc.)

3. Return the filtered results:

- Return the filtered list of employees in JSON format.

4. Add validation for query parameters:

- Ensure that only valid query parameters are accepted and handle invalid inputs gracefully.

Evaluation Criteria:

- Functionality: The search functionality should return accurate results based on the provided filters.

- Query Parameter Validation: The system should handle invalid query parameters gracefully without crashing.

- Performance: Ensure that the search operation performs well even with large datasets.

- Documentation: Provide clear documentation for the new search API endpoint.

Objective:

By implementing advanced search and filtering capabilities, you will provide users with powerful tools to quickly locate specific employees, improving overall usability and efficiency of the system.

Assignment 2: Building a Reporting Dashboard

Objective: Build a reporting dashboard that provides insights into employee data, such as total number of employees, average salary, top-performing departments, and employee turnover rates.

Steps:

1. Create a new file named `reportRoutes.js` in the `routes` directory:
   - This file will handle all routes related to generating reports.

2. Define the following endpoints:
   - `GET /reports/totalEmployees`: Returns the total number of employees.
   - `GET /reports/averageSalary`: Returns the average salary across all employees.
   - `GET /reports/topDepartments`: Returns the top-performing departments based on average salary.
   - `GET /reports/turnoverRate`: Returns the employee turnover rate over a specified period.

3. Implement the logic for each endpoint:
   - Use Sequelize's aggregation functions to calculate the required metrics.
   - Join the necessary tables to gather comprehensive data.
4. Export the router from `reportRoutes.js` and include it in your main `index.js` file.
5. Design a simple HTML page:
   - Create a basic HTML page that displays the report data fetched from the API endpoints.
6. Style the page:
   - Use CSS to style the HTML page for better readability and aesthetics.

Evaluation Criteria:

- Accuracy: The reported data should accurately reflect the current state of the employee database.

- Performance: Ensure that the report generation process does not cause significant performance issues.

- Usability: The HTML page should be easy to navigate and understand.

- Scalability: The solution should be scalable to accommodate future reporting needs.

Objective

A reporting dashboard will provide valuable insights into the organization's human resources, helping managers make informed decisions and optimize resource allocation.