5.NodeJS Assignment + Notes —-Making it perfect !

CENTRALOGIC

12/5/24

💡 **Github code reference using modular structure**

https://github.com/gauravwani127/NODEJS_training/tree/main/4modularStructure

Assignment

Assignment: Shift Tracking Module

Overview

The goal of this assignment is to create a backend system that allows employees to log in, start their shifts, and fill out their timesheets with project details. The backend will be built using Node.js, TypeScript, and PostgreSQL.

Functional Requirements

1. **Employee Register**:
2. **Employee Login**:
    - Employees can log in to start their shift.
    - Authentication should be implemented (JWT or session-based).
3. **Shift Management**:
    - Start shift when an employee logs in.
    - Track the actual hours worked.
    - Track the assigned shift hours.
4. **Timesheet Management**:
    - Employees can fill out a timesheet with project details.
    - Each timesheet entry should include the project name, task name, and the duration (from date to till date).
5. **Reporting**:
    - Generate reports on actual hours worked versus assigned shift hours.

Entities and Fields (Constructive changes in entities are welcomed)

1. **Employee**
    - `id` (UUID, Primary Key)
    - `name` (String)
    - `email` (String, Unique)
    - `password` (String, Hashed)
    - `assignedShiftHours` (Integer) - Number of hours assigned per shift
    - `role` **SuperAdmin, Manager, Employee**
2. **Shift**
    - `id` (UUID, Primary Key)
    - `employeeId` (UUID, Foreign Key)
    - `startTime` (Timestamp)
    - `endTime` (Timestamp, Nullable)
    - `actualHours` (Float, Computed)
3. **Timesheet**
    - `id` (UUID, Primary Key)
    - `employeeId` (UUID, Foreign Key)
    - `shiftId` (UUID, Foreign Key)
    - `projectName` (String)
    - `taskName` (String)
    - `fromDate` (Timestamp)
    - `toDate` (Timestamp)

4. Claims

- `id` (UUID, Primary Key)
- key (String) Eg . CanReceiveReport , Can AssignTasks
- value (String) Eg. true , false
- EmployeeId (UUID, Foreign Key)

Scenarios and Use Cases

1. **Employee Login**
    - An employee logs in using their email and password.
    - Upon successful login, a new shift record is created with the start time.
2. **Shift End**
    - When an employee logs out, the shift's end time is recorded, and the actual hours are calculated.
3. **Timesheet Entry**
    - The employee fills out a timesheet with details of the work done during the shift.
    - Timesheet entries include project name, task name, and the duration of the task.
4. **Reporting**
    - Generate a report comparing the actual hours worked to the assigned shift hours.
    - This could be a simple API endpoint that aggregates and returns the data.
    - An api that will return the report of actual hours worked by employees, assigned hours , date , time , in excel format..

Technical Specifications

- **Node.js** for the backend server.
- **TypeScript** for type safety.
- **Express.js** for the web framework.
- **PostgreSQL** for the database.
- **Sequelize** for ORM.
- **JWT** for authentication.

Step-by-Step Implementation Guide

1. Setting Up the Project

2. Define the Entities

3. Implement Authentication

4. **Frame accordingly the flow of the project**

Notes ——————————————————⟶

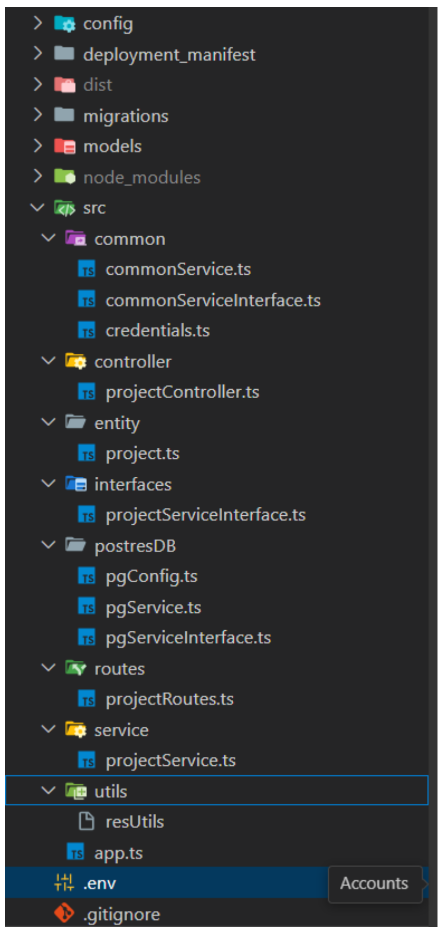
Best Practices in your NodeJS application

1. **Use of Environment variable (.env)**

> Environment variables are important to avoid security breaches and isolate environments

2. **Using Modular structure (Modularization)**

> Modular structure makes code maintainable
>
> Current Structure = app.ts ⟶ service.ts
>
> Expected Flow = app.ts ⟶Routers ⟶ controllers ⟶ service ⟶ dbService



```
> config
> deployment_manifest
> dist
> migrations
> models
> node_modules
v src
  v common
      commonService.ts
      commonServiceInterface.ts
      credentials.ts
  v controller
      projectController.ts
  v entity
      project.ts
  v interfaces
      projectServiceInterface.ts
  v postresDB
      pgConfig.ts
      pgService.ts
      pgServiceInterface.ts
  v routes
      projectRoutes.ts
  v service
      projectService.ts
  v utils
      resUtils
    app.ts
  .env
  .gitignore
```

3. **Use of CORS**

CORS errors happen **when a webpage makes a request to a different domain than the one that served the page**, and the server responds with an HTTP error because not allowed by server's configuration

4. **Normalization and associations in Database schemas**

Breaking databases into more for consistency and avoid redundancy.

— Use of Foreign Key

5. **Swagger**

   Use of swagger→

6. **Use of Logging in NodeJS**
7. **Using asynchronous functions**
8. **Using Error Handling**
9. Security Practices - **npm audit , snyk , helmet**
10. **Logging - Winston**
11. **Real Time Monitoring - PM2**
12. **Query Builder Libraries**:

Consider using query builders like Knex.js or ORM libraries like Sequelize, TypeORM, Prisma to abstract complex queries and prevent SQL injection

13. **SSL Connections**:

Ensure that your database connections use SSL, especially when connecting to a remote PostgreSQL server.

JavaScript basics

I suggest solving atleast one question every day-

https://exercism.org/tracks/javascript

https://javascript.info/

https://learning.postman.com/docs/sending-requests/requests/

How to submit assignment guidelines?

Theoretical questions are not to be submitted, Check documentations for finding answers to them

For 1. Coding question to be sent by pushing it into a public git respository . Exclude Node_Modules using gitignore

Maintain a differnt folder for Outputs , put here ss of the resposes of the apis given in the requirement

Submit a detailed :

- ***Screen capture (Record) the flow of the Project , using debugger***
- ***Put the code in a public git repository and share it***
- ***Attach screenshots of the outputs***
- ***Use best practices as they were followed in last session (Modular structure)***

Deadline: 10/6/2024

Note: Feel free to reach out for any clarifications or assistance during the assignment

+

Documentation is the key, Happy Learning---

CentraLogic

Gaurav Wani

Team Lead | CentraLogic Consultancy