CoderHack

Problem Statement

Develop a **RESTful API** service using **Spring Boot** to manage the Leaderboard for a Coding Platform while using **MongoDB** to persist the data.

Problem Description

While coding platforms usually host multiple contests while maintaining numerous leaderboards, this assignment requires you to design a service for managing the leaderboard of a specific contest. Assume the platform has only one contest with a single leaderboard. The platform also gives virtual awards to the users called Badges based on their score.

Requirements

- The **API** must handle CRUD operations for competing user registrations
- Each user has the following fields:
    - User ID (Unique Identifier)
    - Username
    - Score (0 <= Score <= 100)
    - Badges (Code Ninja, Code Champ, Code Master)
- User registration requests must have a User ID and Username
- The score must be 0, and the badges must be empty initially after the registration
- Updation through PUT requests is only allowed for Score
- Badges must be awarded based on the score:
    - 1 <= Score < 30 -> Code Ninja
    - 30 <= Score < 60 -> Code Champ
    - 60 <= Score <= 100 -> Code Master
- A user can only have a maximum of three unique badges
    - {Code Ninja, Code Champ, Code Master} -> Valid
    - {Code Ninja} -> Valid
    - {Code Ninja, Code Champ, Code Master, Code Ninja} -> Invalid
- User retrieval must be sorted based on the score
- Sorting should have the time complexity of O(nlogn)
- Include basic **JUnit** test cases to verify the operations

Validation and Error Handling

- Add basic **validation** for all fields (Ex. Score > 0)

- Handle common errors and return **appropriate HTTP codes** (Ex. 404, User not found)

Endpoints

- GET /users - Retrieve a list of all registered users
- GET /users/{userId} - Retrieve the details of a specific user
- POST /users - Register a new user to the contest
- PUT /users/{userId} - Update the score of a specific user
- DELETE /users/{userId} - Deregister a specific user from the contest

Publishing and Documentation

- Publish your code to a public **GitHub** repository
- Write meaningful commit messages (optional)
- Include a descriptive **README.MD** for your application codebase
- Create and add a public **Postman Collection** in the README.MD

Additional Notes

- Implement the solution using a **layered approach** - Ex. Entity, Controller, Service, Repository

What to Submit?

- You will be submitting your GitHub code repository for this assignment.
- Note: An activity will be part of your program to collect this submission.

Additional Resources

- Local Environment Setup - Backend - For setting up your local environment
- Seting Up Applications Using Spring Intializr - To learn about generating boilerplate code with Spring Intializr, adding dependencies, and Spring Boot best practices
- Template for Backend Takehomes
- Make sure to initialize a new repository for every project on GitHub. Use one of the below for the necessary steps:
    - Installing Git and Creating a Repository OR
    - How to Add a New Project to GitHub Repository with Visual Studio Code
- Postman Collections - Getting Started and Postman Collections - Learning More
- Basic writing and formatting syntax for README.MD and Markdown Cheatsheet