Back-end Technical Case | Ruby on Rails Developer

BEFORE YOU GET STARTED:

- Return this test by the time indicated in the email used to send this challenge.
- Once you are ready to submit, post your work to a private GitHub repository and reply to the email with the link, or simply send me the files in a zip directly.
- Follow the case instructions; they are intentionally vague to allow for your interpretation. However, be sure to add tests and validation to your coding exercise.
- Do not hesitate to contact us if you need any clarification or have questions.

---

1. Introduction

Consider an existing application that has the following REST endpoints:

- Get users: https://cgjresszgg.execute-api.eu-west-1.amazonaws.com/users
- Get a user: https://cgjresszgg.execute-api.eu-west-1.amazonaws.com/users/fd282131-d8aa-4819-b0c8-d9e0bfb1b75c
- Get teams: https://cgjresszgg.execute-api.eu-west-1.amazonaws.com/teams
- Get a team: https://cgjresszgg.execute-api.eu-west-1.amazonaws.com/teams/7676a4bf adfe-415c-941b-1739af07039b

1.1. Data Modeling

- The **teams** have a list of **users**;
- Each **user** can be part of zero or more **teams**;
- Each **team** has one user as a **team lead**;

2. Main task

Create a new **Roles** service that enhances the Users and Teams services, by giving us the concept of team roles and the ability to associate them with **team members**.

At minimum three roles should be pre-defined:

1) Developer, Product Owner, and Tester.
2) Developer should be the default role.

The new Roles service should be able to do the following actions via REST:

- Create a new role
- Assign a role to a team member
- Look up a role for a membership

3) A membership is defined by a user ID and a team ID

- Look up memberships for a role

e-core

3. Must Have

- **README with:**

○ Description of how you approached the problem and the solution.

○ Information on how to run the code

○ Suggestion for improvement in the Team or User services

- Code, README, and any supporting documentation must be written in English
- Write tests (unit and integration tests)
- Add validation and think about edge cases
- Avoid logic where it does not belong, think about architecture as if the application was big.
- Must be written using Ruby on Rails

4. Recommended:

○ docker and docker-compose

○ What happens if the data you are using gets deleted?

○ Rails

Good luck! 👾