Technical Exam -work in Japan-

To develop Simple search application

Description

- Let us know if you find strange points/have questions about this exam.
- There are four requirements (A, B, C and D).
- Only when you complete all the requirement, please submit the output.
- Our engineers will review your work and decide whether you will proceed to the next process.

Rule

- Write it in Java/ Scala/ Ruby/ Phython.
- Your code is to be managed by git, and you are to submit your project by sharing the link of git hosting like GitHub.
- Please make your project available to be checked its functionality by deploying to Heroku or any other servers/tools.

Expected time needed to complete

- Standard: 30 - 40 hours (* Of course you can spend more time if you want to finish all of section)
- Measure the time to finish.

Requirements

Table (model) structure

- There are three tables (models)
    - developers（Developer)
    - programming_languages (ProgrammingLanguage)
    - languages (Language)
- Relationships
    - “developers” “programming” and “languages” tables should be many-to-many (has many :through relationship)
    - “developers” table and “Interview” table should be one-to-many.

- Column information
    - developers
        - email（not null）
    - programming_languages table

        - name（not null） （e.g. php, *ruby, JavaScript, python, Scala, kotlin, swift*）
    - languages table
        - code（not null） （e.g. bd, vn, *en, ja*）
    - interviews table
        - score (1-5)
        - comment : to keep recording the comment after interview is done

Function

A. Search Website Search Website (with unit test and system test)

- by making tables of “developers” “programming_languages” “languages”, and Implement a page that fulfills the following search conditions.
    - Being able to search developers who can write ruby.
    - Being able to search developers who can write ruby and speak Japanese
    - Being able to search developers who can write ruby and javascript also speak Japanese
- Write unit test of the search class. (optional but will be a huge plus)
- Write system test for the search views, controllers, models. Etc.

※ You don’t have to pay much attention to the UI (view) this time. The essential part is the logic and the structure of the way of your coding.

- The UI should look like this.

email	programming_languages	languages
yamada.hana@example.com	ruby, javascript, php	jp, en
ktaro.tanaka@example.com	ruby, javascript, kotlin	jp

- Create a pull request.
- Make a unit test. Upload a dummy data(100+data of “developers” which links randomly with “programming_languages”, “languages”.) by using a command line interface (e.g. Rails seeder, rake, Laravel artisan.)
- Create a code to extract “programming_languages” which is not used in any “developers”.

B. DB Seeder

- Create a seeder mechanism.

  ○ Create dummy “developer”’s records (100 records) using a seeder that are associated with “programming_languages” and “languages” with another seeder

- Write tests for DB seeder.

C. Front end

- Make a form which enables adding, deleting and sorting “interviews” freely in a Admin page of “developers” without refreshing page.
- Using JS framework (e.g. Vue, React, Angular) will be better.

D . API

- Create an API for getting a developer’s detail (http GET) by JSON format.

  ※ Your controller should have Create, Update, Delete and Show methods which will take care of POST,PUT,DELETE and GET request accordingly.
- The associated information, such as”programming_languages”, “languages” should be included in the response.
- Write request tests for the API.
- Implement the API to be able to respond within 50 ms even if the number of access is 10 at the same time and 10 times in a row.