Category: Technical Test
Position: Ruby/Rails Developer

Overview

- Create a web application that will extract large amounts of data from the Google search results page.
- Store this data and both display and report back this data in various ways
- Users must be authenticated to use the application

Application Requirements

1. Authenticated users can upload a CSV file of keywords. This upload file can be in any size from 1 to 100 keywords.

    Create a user interface to perform this action.

    Optional: create an API endpoint to perform this action.

2. The uploaded file contains keywords, each of those keywords will be used to search on http://www.google.com and they will start to run as soon as they're uploaded. You need to find a way to work around the limitations of repeated mass-searching keywords as Google prevents it e.g. a delay would suffice for instance.

3. Each search result / keyword result page on Google you will store the following information on the first page of results:

    a. Total number of AdWords advertisers on the page.
    b. Total number of links (all of them) on the page
    c. Total of search results for this keyword e.g. About *21,600,000* results (0.42 seconds)
    d. HTML code of the page/cache of the page.

4. Create a report from the stored information. This report will give back the details of each of the items stored in the database, sorted by keyword.

    Build a user interface to view past reports.

Technical requirements

- Use Ruby on Rails (6.x.x)

  Optional: Use our team Rails template as a boilerplate to save time:

  https://github.com/nimblehq/rails-templates

- Use PostgreSQL

- For the interface, a front-end frameworks such as Bootstrap or Tailwind CSS can be used. Use SASS or PostCSS as the CSS preprocessor.
  - Extra points will be provided to the neatness of the frontend.

- Use Git during the development process. Push to a public repository on Bitbucket, Github or Gitlab. Make regular commits and merge code using pull requests.

- Write unit tests using RSpec.

- *Optional*: deploy the application to Heroku or other PAAS or IAAS (AWS, Azure, Digital Ocean)