Technical task for Visma EasyCruit Backend Team

Overview

The purpose of this task is to build a small REST API service for handling candidates. Service should support CRUD operations on a candidate record. A candidate is assumed to be supplying their first name, last name, email address and motivation letter text. As soon as the candidate information is updated the motivation letter should be stored as a text file, so it can be downloaded by HR. A link for accessing the file should be returned from the candidate read operation. We expect you to finalise the minimum set of requirements (listed at the end of this document). Any initiative shown to complete any additional requirements will be considered as an advantage.

Workflows

- When API consumer is calling GET HTTP method on http://<app_domain>/api/candidates/{candidate_id} it should return:
    - 4xx codes for invalid requests(invalid candidate ID, authentication issues, etc.)
    - 200 OK and the following response body if the candidate exists in the DB:

      ```
      {
         “candidate_id” : “<candidate_id>”,
         “first_name”: “<candidate_first_name>”,
         “last_name”: “<candidate_last_name>”,
         “email”: “<candidate_email>”,
         “motivation_letter”: “<link_to_motivation_letter_file>”
      }
      ```
- When API consumer is calling GET HTTP method on http://<app_domain>/api/candidates it should return:
    - 4xx codes for invalid requests(invalid request, authentication issues, etc.)
    - 200 OK and the following response body if there are any candidates in the DB:

      ```
      [{
         “candidate_id” : “<candidate_id1>”,
         “first_name”: “<candidate1_first_name>”,
         “last_name”: “<candidate1_last_name>”
      },
      ….
      {
         “candidate_id” : “<candidate_idN>”,
         “first_name”: “<candidateN_first_name>”,
         “last_name”: “<candidateN_last_name>”
      }]
      ```

- When API consumer is calling DELETE HTTP method on http://<app_domain>/api/candidates/{candidate_id} it should return:
    - 4xx codes for invalid requests(invalid candidate ID, authentication issues, etc.)
    - 204 No Content if candidate exists and also delete both data from the DB and any created motivation letter file.
- When API consumer is calling POST HTTP method on http://<app_domain>/api/candidates/ with the following body:

  ```
  {
     “first_name”: “<candidate_first_name>”,
     “last_name”: “<candidate_last_name>”,
     “email”: “<candidate_email>”,
     “motivation_letter”: “<motivation_letter_text>”
  }
  ```

  Service should create a candidate in the DB with the information sent in the request as well as create a file containing the motivation letter content in background. Service should return:
    - 201 Created with the candidate ID that can be used for accessing the candidate
    - 4xx codes for invalid requests(invalid request body, authentication issues, etc.)

Minimal set of requirements

- Service should be built in Perl/Python/PHP
- Modern Web framework should be used
- Code should be covered with unit tests with high level of coverage
- Clean code and logic will be highly valued
- Code should be uploaded to GitHub or any other public VCS
- Motivation letter file should be created through a background service

Additional requirements to consider

- Data persistency
- Service scalability (should support 1+ running instance of service)
- Designed to handle high load
- Authentication/authorization should be implemented
- Ways to deploy service to Docker or Serverless should be implemented
- Usage of cloud native services
- Build either part of service in C#/Java
- Documentation for deployment/run should be provided
- OpenAPI implemented for the service