Python Assignment

Table of Contents

Introduction

The purpose of this assignment is to assess your capacity to execute a specific request.

It is taking into account some of the software engineering aspects beyond programming, like

- Packaging
- Modularity
- Error/Exception management
- Documentation
- Deployment
- Testing
- Best Practices
- Clean and documented code
- …

Please pay attention to have clear documentation how to **deploy run** and **test** delivered application.

Target Frameworks, tools and services

Ideally, the application should be developed with the following:

- Python (flask or FastAPI, sync or async of your choice)
- Docker (for containerizing your application)
- Git (personal repository of your choice)

Specifications

1. You have to create REST API, that will generate passwords on demand.
2. It should be able to handle following configurations (params)
    a. Password length (if length is for example 10 it should generate and return random password with length of 10)
    b. Numbers flag (if enabled it should consider number symbols during password generation, like ‘1’, ‘2’, …, ‘9’)
    c. Lowercase chars flag (if enabled it should consider lowercase ascii characters also during password generation, like ‘a’, ‘b’, … , ‘z’)
    d. Uppercase chars flag (if enabled it should consider uppercase ascii characters also during password generation, like ‘A’, ‘B’, … , ‘Z’)
    e. Special symbols flag (if enabled it should consider special symbols also during password generation, like ‘%’, ‘$’, … , ‘@’)
3. Default password length and default flags should be configurable from the server.

4. Password length is limited to 200 characters max.
5. It should raise an exception and return formatted response correspondingly in case if user makes request with disabling all features.
6. Consider covering all edge cases that may be requested by the user.

Testing

1. All the functionalities of this application and any user interaction should be thoroughly tested. This is a great opportunity to show that you were taking into consideration any edge case(s) you might have found along the way!
2. Consider providing also Postman collections and tests, that are demonstrating all nice features and edge cases of your application.

Extras

It would be nice to cover the sections below.

Best Practices

It would be nice to have in documentation, the section that you describe what best practices you used and what kind of design decisions you made during your development of this assessment.

Packaging

With Docker, build a simple dockerfile in order to containerize your application. Consider having docker-compose file also.

Version Control System

Git history will also be reviewed. (Commits, messages, branches, tags, etc.)

Deployment

Create a CI pipeline on your personal repository that will automatically test, build and push your docker image to your own container registry (dockerhub).

Workflow Diagram

It would be nice to include in the documentation workflow diagram (consider using tools like flowchart maker or other.)

Additional Information

If you come across any issue, feel free to reach us back. It would be fine to post your solution in git and provide us access (or link if it is public) as soon as you finished the assessment.