Embedded BMS Applications Developer (EBAD)

Technical Assessment

Envio is an award- winning startup pioneering technologies that transform how we monitor, manage, operate, and interact with commercial real estate. We are more than 50 engineers, data scientists, product managers from over 27 different nationalities, working together to improve people's lives and create a more sustainable future through innovation.

As an EBAD we are looking for you to be a senior (Python) developer who is curious, innovative, can quickly pivot and utilize concepts from unfamiliar areas of research, new technologies, leading edge innovations to further improve solutions for our Building Management System Gateway.

We expect you to have the ability to architect and design full IoT Edge/Cloud applications. Additionally we look forward to your ability to assess and to add value to our BMS distributed applications and solutions.

We look forward to your ability to support the design and implementation of our products and to be a passionate engineer that has architected and delivered IoT Edge solutions in a fast-paced agile environment.

We expect you to have previous proven experience with continuous integration and continuous delivery process. We look forward to you leveraging the freedom and agency we provide to make calculated high level decisions, all while ensuring the quality and efficiency of the product.

We understand some of these questions may be quite specific to our business and you may not have the exact answer. We are looking for an understanding of your thought process and your reasoning. be *“Resourceful-* we believe workable solutions exist for every problem.”

We look forward to seeing your solutions.

Coding Task #1- Abstract

Develop a simple Linux CLI application that toggles *gpio* X based on the input from *gpio* Y.

Description

a) **The application should:**

- retrieve the state of *gpio* Y and control gpio X based on the state of gpio Y
- if gpio Y is in HIGH then *gpio* X need to be toggled this way (LOW-HIGH-LOW-HIGH…) every 1 second
- if *gpio* Y is LOW, *gpio* X should remain in LOW as well

Requirements:

- Program should accept options "*-i Y*" and "*-o X*" to set the *gpio* X and Y numbers.
- The target system is a Linux machine with *sysfs* available.
- Programming language you need to use is Python.
- Don't forget about unit tests.

b)**Provide *systemd* service file for your application.**

c) **(optional and not mandatory task)**
Implement simple logging mechanism that is activated by “--log“ option and logs only *gpio* Y state into .txt file with entries like

```
2020-01-01T18:00:00 gpio <Y> HIGH
2020-01-01T18:00:05 gpio <Y> LOW
```

NOTE:
Coding style and quality is more important than feature completeness.
What to submit: A git repository of the work and all commits made on the way.

Estimated Time to solve
3 hours.

Coding Task #2-Abstract

Develop a simple Linux or Windows CLI application that sorts the (name, age, height) tuples by ascending order where name is string, age and height are numbers. The tuples are input by console.
Description
The sort criteria is:

1. Sort based on name
2. Then sort based on age
3. Then sort by score

The priority is that name > age > score

If the following tuples are given as input to the program:

```
Tommy,19,80
John,20,90
```

```
Jony,17,91
Jony,17,93
Json,21,85
```

Then, the output of the program should be:

```
[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]
```

Hints:
In case of input data being supplied to the question, it should be assumed to be a console input.
We use itemgetter to enable multiple sort keys.

NOTE:
Coding style and quality is more important than feature completeness.
What to submit: A git repository of the work and all commits made on the way.
Estimated Time to solve
30 mins.

Coding Task #3- Abstract

Write a dummy 'greeting' web-app that waits for one second and sends a greeting (e.g. Hola, Pycon) to the browser.
Description

1. Write a script that requests greeting from ex1 on your local web-server and prints it to shell. Request time should result in around 1 second
2. Modify your code to call the greeting-server of your neighbour. Delay should be a bit more then 1 second.
3. Modify your code to get greetings of 3 neighbours in a loop. Since requests are sequential, time would be more then 3 seconds.

It's required to use *AsyncIO* + *aiohttp* library.

NOTE:
Coding style and quality is more important than feature completeness.
What to submit: A git repository of the work and all commits made on the way.
Estimated Time to solve
60 mins.

Coding Task #4- Abstract

Develop a simple Linux or Windows CLI application that computes the frequency of the words from the input. The output should output after sorting the key alphanumerically..

Description

Suppose the following input is supplied to the program:

```
New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3.
```

Then, the output should be:

```
2:2
3.:1
3?:1
New:1
Python:5
Read:1
and:1
between:1
choosing:1
or:2
```

`11`

`to:1`

Hints:

In case of input data being supplied to the question, it should be assumed to be a console input.

NOTE:
Coding style and quality is more important than feature completeness.
What to submit: A git repository of the work and all commits made on the way.

Estimated Time to solve

30 mins.