Senior Data Engineer - Technical test

In order to monitor the health of the battery, we decided to send data from the battery to the cloud. This data is transmitted in hexadecimal format and received in our AWS account. As a Data Engineer, it is your role to make this data available to the user as soon as it arrives.

1). Python

Write a Python lambda function that takes as input an event:

```
{
  "device": string,
  "payload": string
}
```

and logs the data to stdout in the following format:

```
{
  "device": string,
  "time": integer,
  "state": string,
  "state_of_charge": float,
  "temperature": float
}
```

2). Terraform

Deploy the lambda function using Terraform

Encoding

The data is transmitted as a hexadecimal string. Every payload consists of 8 bytes. Due to space optimization, the information is not byte aligned. A field can start in the middle of a byte. We therfore need bit operations to decode the payload. The payload is not signed and encoded in little Endian. The following table describes the data fields contained in the payload and their bit positions.

0	7	6	5	4	3	2	1	0
	7	6	5	4	3	2	1	0
	time				type			
1	15	14	13	12	11	10	9	8
	7	6	5	4	3	2	1	0
	time							
2	23	22	21	20	19	18	17	16
	7	6	5	4	3	2	1	0
	time							
3	31	30	29	28	27	26	25	24
	7	6	5	4	3	2	1	0
	time							
4	39	38	37	36	35	34	33	32
	7	6	5	4	3	2	1	0

	state				time			
5	47	46	45	44	43	42	41	40
	7	6	5	4	3	2	1	0
	state of charge							
6	55	54	53	52	51	50	49	48
	7	6	5	4	3	2	1	0
	battery temperature							
7	63	62	61	60	59	58	57	56
	7	6	5	4	3	2	1	0

For instance, `type` is encoded on 4 bits in the first byte. `state of charge` is encoded on 8 bits (1 byte) on the 6th byte.

Time

`time` represents the timestamp of the data. It is defined in seconds since UNIX epoch.

State

`state` is a string, with the following corresponding values:

- 0: "power off"
- 1: "power on"
- 2: "discharge"
- 3: "charge"
- 4: "charge complete"
- 5: "host mode"
- 6: "shutdown"
- 7: "error"
- 8: "undefined"

State of charge

`state of charge` represents the charge of the battery. It is a float with values between 0 and 100 and a 0.5 precision. To store it as an integer it was multiplied by 2.

Battery temperature

`battery temperature` represents the temperature of the battery. Values can vary between -20 and 100. The precision is 0.5. To store it as an integer we added 20 and multiplied it by 2.

Test data

input:

`F1E6E63676C75000`

output:

```
{
  "time": 1668181615,
  "state": "error",
  "state_of_charge": 99.5,
  "temperature": 20.0
}
```

input:

`9164293726C85400`

output:


```
{
  "time": 1668453961,
  "state": "discharge",
  "state_of_charge": 100.0,
  "temperature": 22.0
}
```


input:

`6188293726C75C00`

output:


```
{
  "time": 1668454534,
  "state": "discharge",
  "state_of_charge": 99.5,
  "temperature": 26.0
}
```