ORANGE

ASSIGNMENT PRESENTATION

BY: DANYLO BONDARENKO

Introduction

Thank you so much for the opportunity! I've used a lot of new technology for me in my work. I faced many difficulties during the assignments because of the lack of practice and knowledge in the subject area. Also, not all tasks were covered. But even with this, I tried to give the best result. Further is a description of the work done.

Technical Background

Assignment

To create a simple decoder function in TypeScript to decode the payload of Dragino LHT65

In the work, it was necessary to develop a decoder for the LHT65 Temperature & Humidity Sensor. In order to get acquainted with the operation of the sensor the manual was studied. It is necessary to receive the HEX format and decode it.

Implementation

PHASE 01

Implement JavaScript/TypeScript decoder logic.
The task is to create a simple decoder function in TypeScript to decode the payload of Dragino LHT65 devices

PHASE 02

Implement a REST API for decoding HEX payload.
The task is to create a REST API that calls the decoder microservice from the first assignment.

PHASE 03

Dockerfile for the REST API.
The task is to create a Dockerfile that can be used to build and run the microservices.

Decoder creation

The task was to decode the HEX code to the values of temperature, humidity, battery, and external temperature. Unfortunately, during the development, there was no correct code to decode this sensor. All used variants did not provide a logical and correct result, comparable to the example. In the future, the solution will be finalized to get accurate results.

```
export function hexToBytes(hex: string): number[] {
  const bytes: number[] = [];
  for (let i = 0; i < hex.length; i += 2) {
    bytes.push(parseInt(hex.substr(i, 2), 16));
  }
  return bytes;
}

function extractTemperature(bytes: number[]): number {
  const temperatureBytes = (bytes[1] << 8) | bytes[2];
  let temperatureValue = temperatureBytes & 0x7FFF;

  if ((temperatureBytes & 0x8000) > 0) {
    temperatureValue = -temperatureValue;
  }

  return temperatureValue / 100;
}

function extractHumidity(bytes: number[]): number {
  const humidityByte = bytes[3];
  return humidityByte !== undefined ? humidityByte * 0.5 : 0;
}

function extractBatteryVoltage(bytes: number[]): number {
  const voltageByte1 = bytes[4];
  const voltageByte2 = bytes[5];
  if (voltageByte1 !== undefined && voltageByte2 !== undefined) {
    return (voltageByte1 << 8 | voltageByte2) * 0.01;
  }
  return 0;
}
```

Decoder creation

```
PS C:\Users\lipri\OneDrive\Documents\
e\TestProj> node test.js
Decoded data for payload 1:
{
  temperature: -133.21,
  humidity: 98,
  battery: 4.09,
  temperatureExt: 0
}
---------------------------
Decoded data for payload 2:
{
  temperature: -128.09,
  humidity: 39,
  battery: 4.23,
  temperatureExt: 0
}
---------------------------
Decoded data for payload 3:
{
  temperature: -245.87,
  humidity: 14,
  battery: 4.19,
  temperatureExt: 0
}
---------------------------
```

As a result, we get the following values. They are very different from the proposed ones. But the script works and is called. I assemble a Javascript file from the typescript file and then execute it.

Implement a REST API for decoding HEX payload

Using the resulting decoder, it is necessary to create a microservice on the REST API. Create a server.ts file, which will describe all the logic of the server. It will only be able to accept POST requests. Call the decoder from the previous task and use it.

```
// Routes
app.all('/decode', (req, res) => {
    if (req.method === 'POST') {
        // Get the HEX payload from the request body
        const hexPayload = req.body.payload;

        // Call the decoder function to decode the
        payload
        const decodedData = decodePayload(hexPayload);
            You, 1 hour ago • Initial commit …
        // Send the decoded data as a JSON response
        res.json(decodedData);
    } else {
        // Handle GET request for /decode
        res.send('This route only supports POST
        requests');
    }
});
```

Implement a REST API for decoding HEX payload

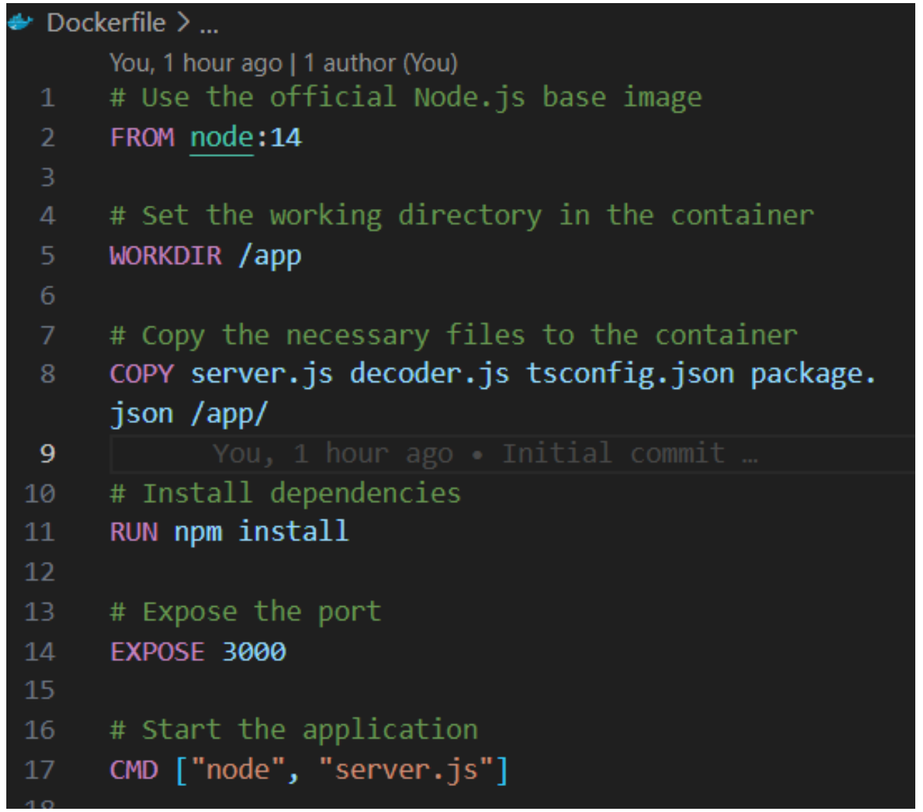
```
PS C:\Users\lipri\OneDrive\Documents\Programming\Orange\TestProj> Invo
ke-RestMethod -Method POST -Uri "http://localhost:3000/decode" -Header
s @{"Content-Type" = "application/json"} -Body '{"payload": "cbb409c40
1990109857fff"}'
>>

temperature humidity battery temperatureExt
----------- -------- ------- --------------
    -133.21       98    4.09              0
```

Let's start the server first. Then we pass the HEX code via Powershell. After that, we get the decoding according to the work of the written decoder.

Dockerfile for the REST API

The next step is to create a Dockerfile which should include all the results. First, let's install Docker. Then fill the Dockerfile with the necessary information. After that, we can build the container with the command "docker built" and start the server. The result is working and the result is the same as before.



Conclusion

The Conclusion Of The Work

The work to decode the HEX code from the sensor was done. Unfortunately, the obtained decryption code does not correspond to reality and it needs to be finalized. Also with the help of typescript, a server was deployed to transmit and decrypt data on it. And then the container was built using Docker. The final step was to deploy microservices to the cloud. But unfortunately, the knowledge available was not enough to create a CI/CD pipeline and deploy the microservices developed in the previous assignments to a cloud. Next, I will further study this task and the options for its implementation.

As a result of the work, I got a server on which the data is decrypted. Also, I have learned how to write the code in TypeScript and got acquainted with the logic of the sensors. There is a lot more to improve, so I won't stop there.

THANKS

FOR YOUR ATTENTION