Microcontroladores
Prática Final 2 - Aplicativo web para receber e mostrar dados vindos de equipamentos IOT

Átila Camurça Alves

[1]Instituto Federal do Ceará (IFCE)

Abstract. *The Wireless Monitor app has the goal to allow a embed systems developer to send data to the cloud collected by his IOT device and preview in a browser.*

Resumo. *O aplicativo Wireless Monitor tem o objetivo de permitir que um desenvolvedor de sistemas embarcados possam enviar para a nuvem os dados obtidos por seu equipamento IOT e visualizá-los no navegador.*

1. Introdução

O aplicativo Wireless Monitor tem o objetivo de permitir que um desenvolvedor de sistemas embarcados possam enviar para a nuvem os dados obtidos por seu equipamento IOT e visualizá-los no navegador.

2. Objetivos

A ideia principal é criar uma *api* leve e simples, visto que equipamentos IOT são limitados. Tendo isso em vista, vejamos os passos para criar um projeto.

2.1. Cadastro do desenvolvedor

O desenvolvedor inicialmente deve fazer um cadastro simples na ferramenta. Esse cadastro irá criar para ele uma `api_key`, ou seja, uma chave única no formato UUIDv4.

2.2. Criar um *Monitor*

Um *Monitor* é um componente interno do sistema criado pelo desenvolvedor de acordo com sua necessidade, é o instrumento que caracteriza os dados coletados e os apresenta na interface web.

Imagine que o desenvolvedor queira medir a temperatura de um ambiente e acompanhar suas variações. Para isso ele deve criar um *Monitor* de Temperatura, que apenas recebe um valor a um certo intervalo de tempo. Dessa forma o desenvolvedor pode acompanhar as variações ou ainda ver em forma de gráfico um conjunto de variações de um período de tempo anterior.

Da mesma forma que uma chave UUID é criada para o desenvolvedor, uma chave é criada para o Monitor - `monitor_key`.

2.3. Autenticação do equipamento

Para autenticar e identificar o desenvolvedor e seu *monitor* é preciso enviar a `api_key` e a `monitor_key` via método *POST* para o *endpoint* `/api/authenticate`. Em caso positivo o sistema irá retornar um *token*. Esse *token* servirá para qualquer troca de informações futuras entre o equipamento IOT e o sistema. Esse método de autenticação é chamado de JWT ou *JSON Web Token*, um padrão internacional *RFC 7519* para intercâmbio de dados entre entidades.

Após ter o *token* o desenvolvedor deve passá-lo através da *Header HTTP* denominada *Authorization* usando *schema Bearer*. Algo do tipo:

```
Authorization: Bearer <token>
```

Um *token* é formado pelas seguintes informações:

- *Header*
- *Payload*
- *Signature*

Essa é uma forma segura e com pouco custo de memória. Além de ser uma forma de autenticação *stateless*, em que não são usadas sessões e nem mesmo *cookies*.

2.4. Envio dos dados

Além do cabeçalho contendo o *token* o usuário deve passar os valores coletados pelo equipamento e enviar para o sistema. Para isso ele deve enviar uma requisição *POST* para o *endpoint* `/api/send`, com o atributo `data` contendo um JSON com os dados.

No exemplo do *Monitor* de temperatura é necessário enviar apenas o valor, algo do tipo:

```
{
    "value": 23.89
}
```

Sumarizando o código final seria algo como:

```
HTTPClient http;
http.begin("https://wireless-monitor.site/api/send");
http.addHeader("Content-Type", "application/json");
http.addHeader("Authorization", "Bearer <token>");
http.POST("data={value:23.89}");
http.writeToStream(&Serial);
http.end();
```

2.5. Visualização dos dados

Após captar e enviar dados do IOT para a nuvem é possível acompanhar os resultados pelo sistema. A forma de visualização será como mostra a Figura 1

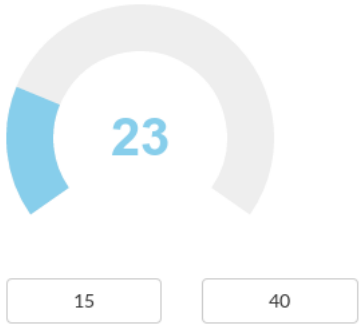

Figura 1. Visualização dos dados na web

3. Cronograma

Tarefas	Semana 1	Semana 2	Semana 3	Semana 4
Protótipo Inicial	X			
Programação	X	X	X	
Testes			X	X
Relatório	X			X

Tabela 1: Cronograma