SENAI

Serviço Nacional de Aprendizagem Industrial

PELO FUTURO DO TRABALHO

Vaso Smart

SA- terceira fase

Cleandro Menezes de Oliveira
Leandro Ciysz de Oliveira
Mathias Murilio dos Santos

Maio/2023
Florianópolis/SC

SUMÁRIO

1. INTRODUÇÃO

Este relatório tem o intuito de descrever as etapas, os objetivos, explicar a ideia e mostrar os resultados do projeto de vaso smart com integração IOT para a SA. A internet das coisas é um assunto em alta atualmente e como forma de vivenciar uma introdução interessante nesse universo, houve a ideia de criar um projeto que engloba a parte de programação e também a conexão com a internet das coisas.

Dessa forma o projeto “Vaso Smart” utiliza a placa ESP32 que possui conectividade Wi-fi, um sensor de umidade do solo, leds, um relé, bomba de água, além de toda a parte de programação/configuração onde foi utilizado o programa Arduino e também o TagoIO. Portanto esse relatório servirá como uma forma de descrição de todo o projeto de forma detalhada e organizada.

2. OBJETIVOS

Além de aplicar conceitos das aulas de IoT e modelagem de sistemas, os principais objetivos deste projeto de vaso smart são:

1- Fazer um gráfico dos resultados obtidos com o sensor de umidade do solo no painel da plataforma TagoIO.

2- Montar o circuito do projeto de forma organizada e eficiente.

3- Aplicar a linguagem C ao código na plataforma Arduino e conectar ao TagoIO. O código deve ser escrito sistematicamente.

4- Regar as plantas automaticamente com base no nível de umidade do solo detectado e visualizar os resultados em tempo real em seu painel TagoIO.

3. DESENVOLVIMENTO

3.1. SENSORES UTILIZADOS

O principal sensor utilizado neste projeto foi um sensor de umidade do solo. Sua função era coletar o teor de umidade do solo e enviar esses dados para a placa esp32 utilizada no projeto. Este sensor tinha vários recursos que precisavam de pesquisa e modificação.

Embora seja chamado de sensor de umidade do solo, na verdade ele mede a resistividade do solo e, de acordo com nossa pesquisa, quanto maior a resistividade, mais seco fica o solo. Isso causou claramente uma discrepância, pois o teor de água no momento da medição atingia um valor máximo de 4095 (configuração padrão do sensor), mesmo em solo completamente seco. Esse valor não é apenas irracional no gráfico, mas na prática ele só diminui quando o solo está molhado, então a representação gráfica era contra-intuitiva.

Para resolver este problema, foram feitas adições ao código do programa Arduino para inverter intuitivamente os valores e estabelecer uma relação proporcional para exibir resultados para a faixa de 0-100° para umidade do solo.

Para instalar esse sensor é bem simples:

-O pino VCC do sensor deve ser alimentado com o 5V da placa ESP32;
-O pino GND do sensor deve ser conectado ao pino GND da placa ESP32;
-O pino analógico A0 do sensor ao pino G23 da placa ESP32 que foi o utilizado;

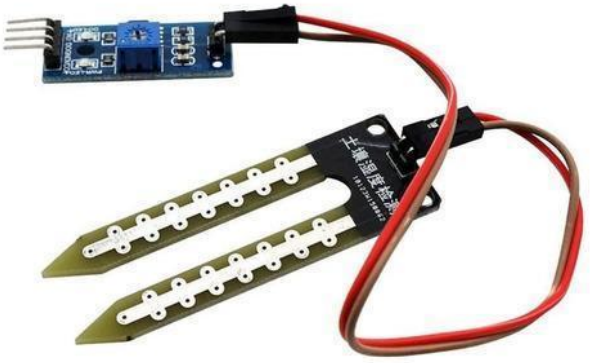


Foto 1: um sensor de umidade do solo.

Na parte de atuadores usamos o Relé. O relé é um componente elétrico que controla o fluxo de corrente em um circuito acionando um interruptor mecânico. Consiste em uma bobina e uma série de contatos que são acionados pela bobina quando a tensão é aplicada.

A operação do relé é relativamente simples. Quando uma tensão é aplicada à bobina do relé, é gerado um campo magnético que aciona uma série de contatos elétricos. Esses contatos podem abrir ou fechar um circuito dependendo do tipo de relé usado.

Sua função no projeto era acionar a bomba de água, no código arduino foi programado um comando que aciona o relé quando o sensor de umidade do solo atingia um nível predeterminado, a bomba então regava e ao chegar no nível desejado, o relé era desativado e consequentemente a bomba de água.

Um exemplo de ligação do Relé com a placa NodeMCU é o seguinte:

- Conectando um pino da placa ESP32 (G22 por exemplo) com o pino IN1 do Relé;

- Conectando o pino 5V da placa ESP32 com o VCC do Relé;

- Conectando o GND da placa ESP32 com o Gnd do Relé;

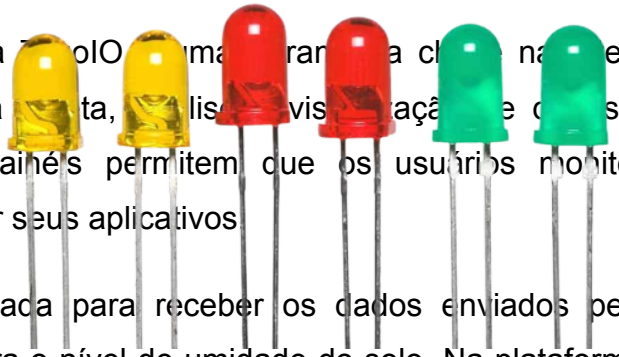
Foto 2- Relé utilizado no projeto

Ainda foram utilizados leds para monitorar o nível de água. Os leds são basicamente componentes que emitem luz.

Os pequenos leds eram acionados ou desligados de acordo com o nível de umidade do solo. Quando o solo estava seco, um led era acionado e após a

bomba de água regar o vaso e atingir o nível limite, outros 2 leds eram ligados enquanto esse primeiro era desligado, os mesmos permaneciam ligados enquanto a umidade estivesse acima desse nível.

3.2. USANDO O TAGO.IO



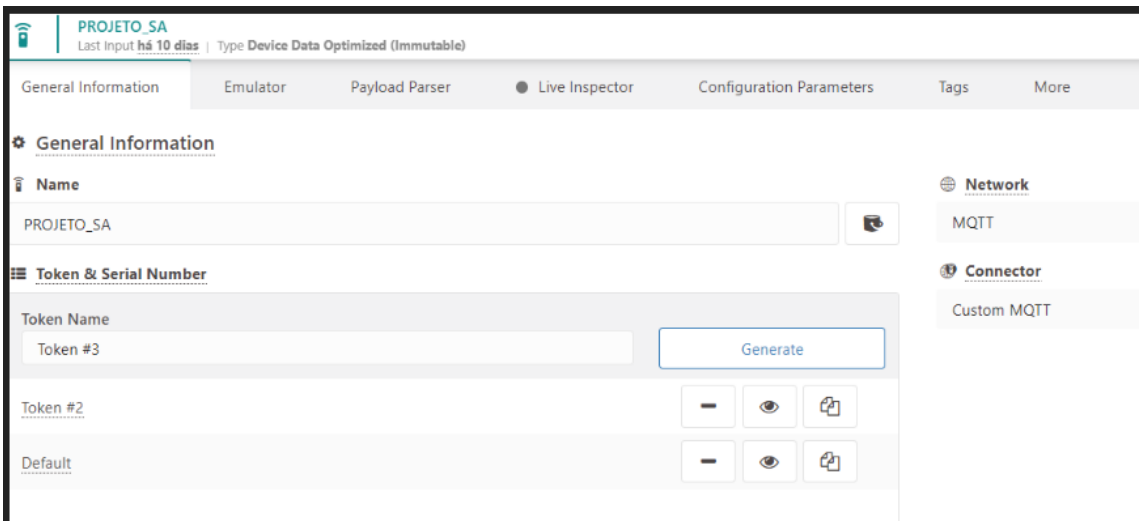
A plataforma TagoIO [illegible] na Internet das Coisas (IoT), permitindo a [illegible] de dispositivos conectados. Os painéis permitem que os usuários monitorem os dados coletados para criar seus aplicativos.

Ela foi utilizada para receber os dados enviados pelo programa do arduino, no caso era o nível de umidade do solo. Na plataforma foi criado uma dashboard com um widget semelhante há um velocímetro que mostrava a variação do nível de umidade do solo entre 0 e 100 ao mesmo tempo em que mudava as cores da variante de acordo com esse mesmo nível. Quando era feita a conexão os dados eram enviados e automaticamente apareciam na dashboard em tempo real.

A Tago suporta protocolos comuns, como MQTT e esse foi o protocolo usado na conexão com o programa arduino. Foram seguidas as seguintes etapas.

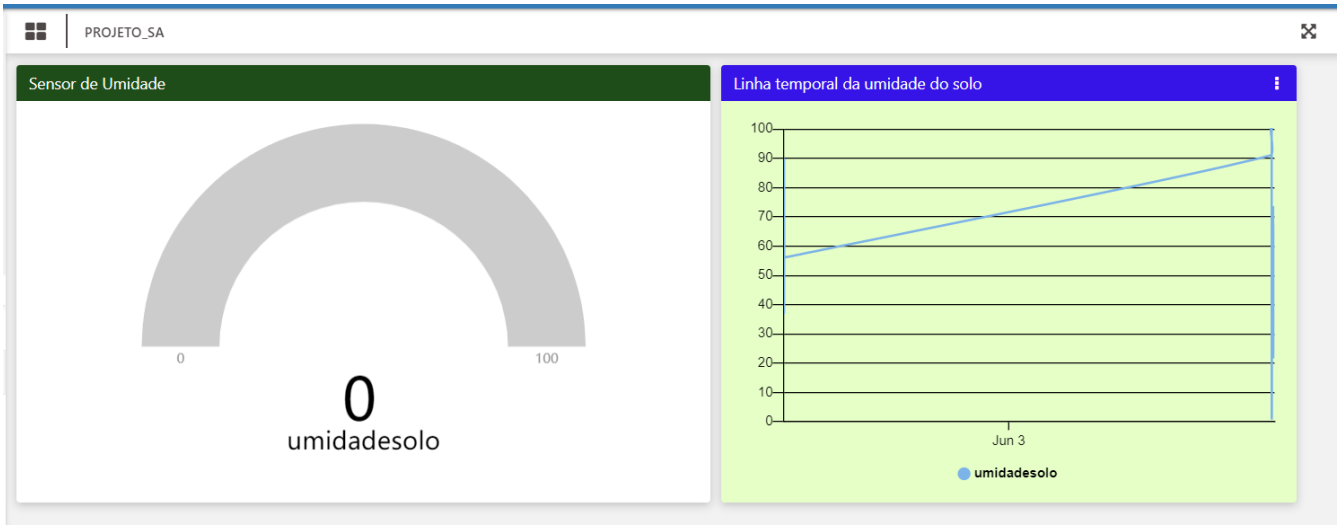
1- Criar conta na plataforma Tago;

2- Criar um “device” para o projeto como está no print abaixo:



Observe que foi gerado um token, o token será utilizado no código no arduino para fazer a conexão da placa com o Tago. O tipo de conexão é o MQTT.

3- Criar um dashboard. O dashboard será onde ficarão os widgets que servem para monitorar o vaso inteligente. Usamos dois widgets, o resultado ficou assim:



Como pode se observar são dois widgets, o primeiro monitora em tempo real o aumento ou diminuição do nível de umidade do solo, o widget é semelhante a um velocímetro de carro e diferencia as cores de acordo com o nível mostrado, e no outro há uma linha do tempo dos valores obtidos, para facilitar o monitoramento do usuário. Ainda há etapas mais específicas para o Tago, como a criação do bucket para armazenar os dados, a edição dos widgets e configurar o código no arduino.

Em resumo, a plataforma Tago desempenha um papel crucial na gestão de dados e aplicativos na IoT. Sua capacidade de integração com protocolos de rede populares, como MQTT, HTTP e TCP/IP, permite uma comunicação eficiente e segura entre dispositivos IoT e a plataforma em nuvem. Ao utilizar a Tago juntamente com protocolos de rede adequados, os usuários podem aproveitar ao máximo os recursos da IoT, possibilitando análises avançadas, visualizações personalizadas e tomada de decisões baseadas em dados em tempo real.

3.3. O CÓDIGO E A CONEXÃO COM O TAGO

Além das configurações dos sensores e atuadores, é preciso configurar no código a conexão com o Tago da seguinte forma:

```
// configurações da conexão MQTT
EspMQTTClient client( "NOME_DO_WIFI",              // nome da sua rede Wi-Fi
"123456",             // senha da sua rede Wi-Fi
 "mqtt.tago.io",          // MQTT Broker server ip padrão da tago "Token",                //
username "298c54c0-3268-4714-po7d-7812bf4ea412",   // Código do Token
"TestClient",             // Client name that uniquely identify your device 1883
// The MQTT port, default to 1883. this line can be omitted );
```

Além disso, tem toda a etapa de transformar os valores em objetos JSON e pegar esses valores obtidos pelo sensor. Por fim deve-se lembrar de colocar no final do código o seguinte:

```
void onConnectionEstablished() {
 client.subscribe("node/status", [] (const String &payload) {
Serial.println(payload);
processa_msg(payload); }); }
```

Essa função será ativada quando o wifi e o mqtt estiverem conectados, e é necessária quando se usa o EspMQTTClient.

O Código completo em “C” com os comentários ficou assim:

```
#include <ArduinoJson.h>
#include "EspMQTTClient.h"
#define LED1 16
#define LED2 17
#define LED3 18

int sensorPin = 33;        // define o pino analógico a ser usado para o sensor de
umidade do solo
int umidadeSolo;           // variável para armazenar o valor da umidade do solo
int umidadeSoloDesejada = 35; // valor desejado da umidade do solo
int relePin = 22;          // define o pino digital a ser usado para o relé

bool bombaLigada = false;  // indica se a bomba está ligada ou não

// variáveis para Json
char json_umd[100];

// variáveis internas
int valorMinimo = 0;       // Valor mínimo do sensor de umidade
int valorMaximo = 4095;    // Valor máximo do sensor de umidade (depende da
resolução do ADC)
```

```
// configurações da conexão MQTT
EspMQTTClient client(
  "FIESC_IOT",              // nome da sua rede Wi-Fi
  "C6qnM4ag81",             // senha da sua rede Wi-Fi
  "mqtt.tago.io",           // MQTT Broker server ip padrão da tago
  "Token",                  // username
  "288c54c0-3268-4713-b07d-7812bf4ea412",   // Código do Token
  "TestClient",             // Client name that uniquely identify your device
  1883                      // The MQTT port, default to 1883. this line can be omitted
);

void setup() {
  Serial.begin(9600);              // Inicia a comunicação serial
  pinMode(relePin, OUTPUT);        // Define o pino do relé como saída
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
}

void converte_json() {
  StaticJsonDocument<300> sjson_umd;

  int umidadeSoloMapeada = umidadeSolo;
  sjson_umd["variable"] = "umidadeSolo";
  sjson_umd["value"] = umidadeSoloMapeada;
  serializeJson(sjson_umd, json_umd);
}

void envia_msg() {
  client.publish("node/umd", json_umd); // You can activate the retain flag by setting
the third parameter to true
}

void loop() {
  int leituraSensor = analogRead(sensorPin);   // Lê o valor analógico do sensor de
umidade do solo
  umidadeSolo = map(leituraSensor, valorMinimo, valorMaximo, 100, 0); // Inverte a
escala do valor do sensor para a escala de 0 a 100

  Serial.print("Umidade do Solo: ");     // Envia uma mensagem pela porta serial
  Serial.print(umidadeSolo);             // Envia o valor da umidade do solo pela porta
serial
  Serial.println("%");                   // Envia o símbolo de porcentagem pela porta serial

  if (!bombaLigada && umidadeSolo < umidadeSoloDesejada) {
    digitalWrite(relePin, LOW);
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, HIGH);
        // Aciona o relé
    bombaLigada = true;
  } else if (bombaLigada && umidadeSolo >= umidadeSoloDesejada) {
```

```
    digitalWrite(relePin, HIGH);        // Desliga o relé
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, LOW);
    bombaLigada = false;

  }

  converte_json();
  envia_msg();
  delay(1000);                          // Aguarda um segundo antes de ler novamente

  client.loop();
}

void processa_msg(const String payload) {
  StaticJsonDocument<300> msg;
  DeserializationError err = deserializeJson(msg, payload);
  if (err) {
    Serial.print(F("deserializeJson() failed with code "));
    Serial.println(err.f_str());
  }
  Serial.print("var:");
  String var = msg["variable"].as<String>();
  Serial.println(var);
  if (var == "status") {
    Serial.print("value:");
    String val = msg["value"].as<String>();
    Serial.println(val);
  }
}

// This function is called once everything is connected (Wifi and MQTT)
// WARNING : YOU MUST IMPLEMENT IT IF YOU USE EspMQTTClient
void onConnectionEstablished() {
  client.subscribe("node/status", [] (const String &payload) {
    Serial.println(payload);
    processa_msg(payload);
  });
}
```

4. CONCLUSÃO

Em resumo, o projeto vaso smart com integração IoT utilizou placa ESP32, sensor de umidade do solo, LED, relé e bomba d'água, plataforma Arduino e TagoIO. Os principais objetivos foram alcançados, como a visualização de dados de umidade do solo em tempo real no painel da plataforma TagoIO e a automação da irrigação de culturas com base nesses dados.

Os sensores de umidade do solo são configurados e calibrados para exibir intuitivamente os resultados em uma proporção que representa a umidade em uma escala de 0 a 100. Relés foram usados para controlar a operação das bombas de água e garantir que as plantações fossem regadas quando necessitado.

A plataforma TagoIO desempenhou um papel fundamental na recolha e visualização de dados. Isso possibilitou a criação de painéis que exibiam graficamente os resultados de umidade do solo, possibilitando o monitoramento em tempo real. A conexão entre a placa ESP32 e a plataforma TagoIO é estabelecida via protocolo MQTT para garantir uma comunicação eficiente e segura.

Este projeto demonstrou que o vaso inteligente funciona bem e demonstra a aplicação prática do conceito IoT. No futuro, mais sensores e funções podem ser adicionados para tornar o sistema ainda mais completo e eficiente para um controle mais abrangente do ambiente da planta.

Em resumo, o projeto de vaso inteligente integrado à IoT automatizou com sucesso a irrigação de plantas com base na umidade do solo. A combinação de componentes eletrônicos, programação e plataforma TagoIO permitiu que as plantas fossem monitoradas e controladas com eficiência, proporcionando uma solução prática para o cuidado das plantas.

5. REFERÊNCIAS

MQTT

MQTT. **© 2022 MQTT.org.** Getting started

Disponível em: https://mqtt.org/ Acesso em 17 jun. 2023.

TAGO

KHOMP. **BLOG DA KHOMP, © 2023.** tecnologia para o desenvolvimento de soluções para Telefonia, IoT & Controle de Acesso.

Disponível em**:** **https://www.khomp.com/pt/tagoio-solucao-iot/** Acesso em 16 jun. 2023.

Imagem 1

Disponível em: **https://www.eletrogate.com/modulo-sensor-de-umidade-de-solo** Acesso em 1. 2023

Imagem 2

Disponível em: **https://www.eletrogate.com/modulo-rele-2-canais-5v** Acesso em 17 jun. 2023