

IRRIGA +

Sistema de Irrigação Automático

Discentes:

Fernando Oliveira

Joel de Araújo

Kauany Miranda

Mateus Souza

Mateus Viana

Rafaela Pereira

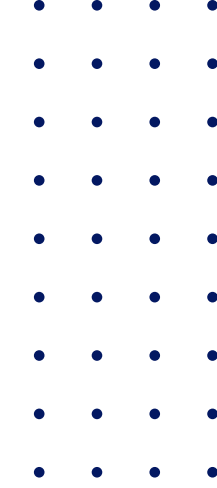
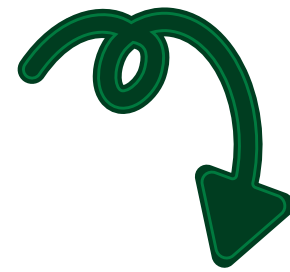
William Matsunaga



Introdução

Proposta de desenvolvimento de um sistema que automatiza a irrigação por meio da detecção da umidade do solo, acionando a água de forma autônoma quando necessário. A solução garante maior praticidade no cuidado com as plantas, economia de tempo, otimização do uso da água e facilidade na manutenção de jardins domésticos.

Tema



No contexto da automação residencial, a IoT possibilita o monitoramento em tempo real das condições do solo, permitindo que sistemas atuem de forma autônoma e eficiente. Com a crescente busca pelo uso consciente da água, soluções de irrigação inteligente vêm ganhando destaque.

Este projeto se insere nesse cenário, utilizando sensores e microcontroladores para desenvolver um sistema de irrigação automatizado e sustentável.

Problemática

Cuidar de plantas pode ser desafiador, especialmente para iniciantes. A irrigação incorreta é comum e pode causar fungos, podridão das raízes e até a morte das plantas.

A rega manual é imprecisa e falha com frequência, principalmente em rotinas atribuladas. Surge, então, a necessidade de um sistema automatizado que controle a umidade do solo com precisão.

A automação se apresenta como solução prática para otimizar o cuidado com jardins domésticos e evitar problemas causados por erros de irrigação.

Objetivo Geral

Desenvolver um sistema inteligente de irrigação para jardins que monitore a umidade do solo e acione a irrigação de forma automática a partir das condições do solo e climáticas

Aspectos Técnicos



Sensor de umidade

Realiza a leitura do nível da água no solo, retornando o valor em porcentagem.



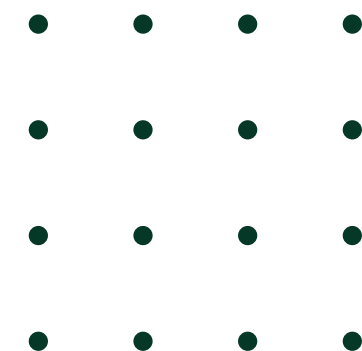
Consulta à API

Obtém a previsão meteorológica do dia, retornando todos os valores da api, especificamente a possibilidade de chuva.



Sensor de chuva

Identifica a presença ou a ausência de precipitação em tempo real no ambiente.



Aspectos Técnicos

Linguagens e ferramentas



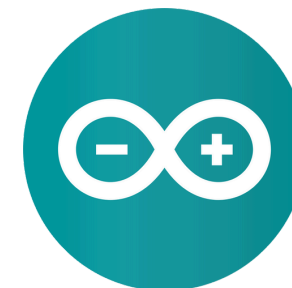
C++



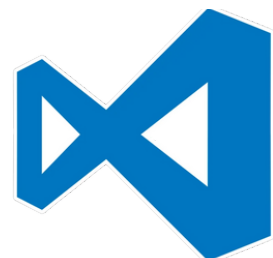
NODE.JS



NEST.JS



ARDUINO IDE



VS CODE



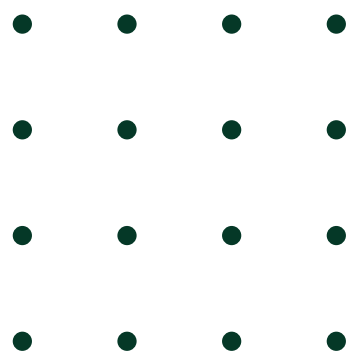
FRITZING



FIGMA

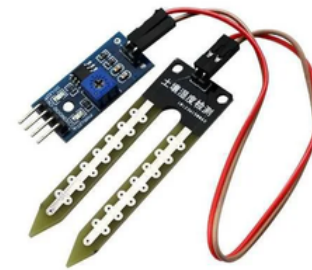


OPENWEATHER
MAP API

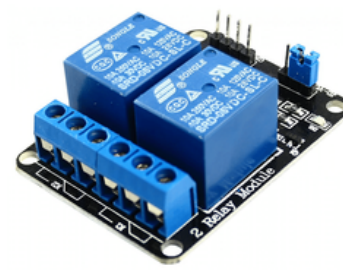


Aspectos Técnicos

Periféricos



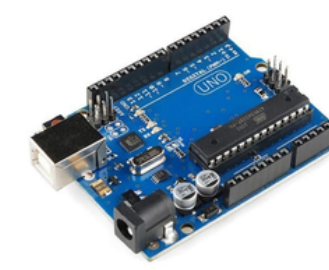
SENSOR DE
UMIDADE



MÓDULO
RELÉ



SENSOR DE
FLUXO



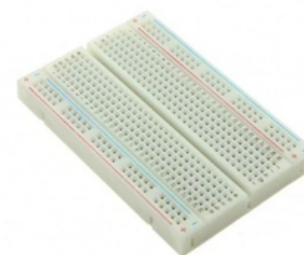
ARDUINO
UNO



MCU ESP12E



BOMBA DE
ÁGUA



PROTOBOARD



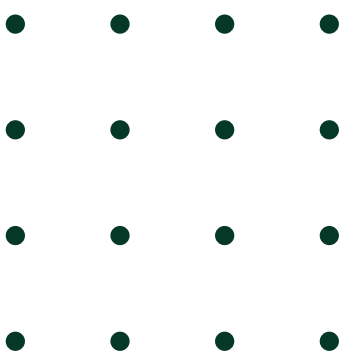
SENSOR DE
CHUVA



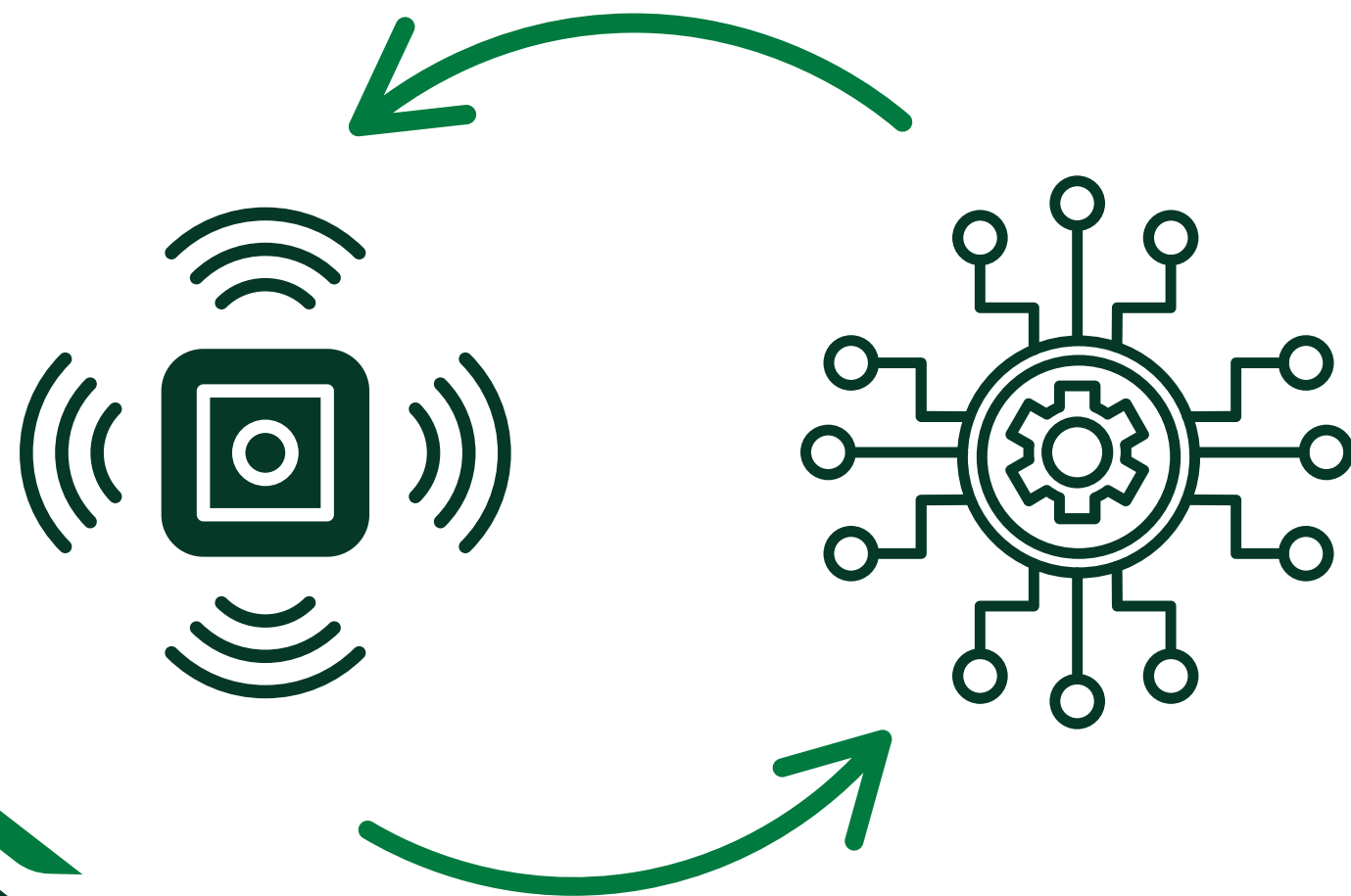
MANGUEIRA



JUMPERS



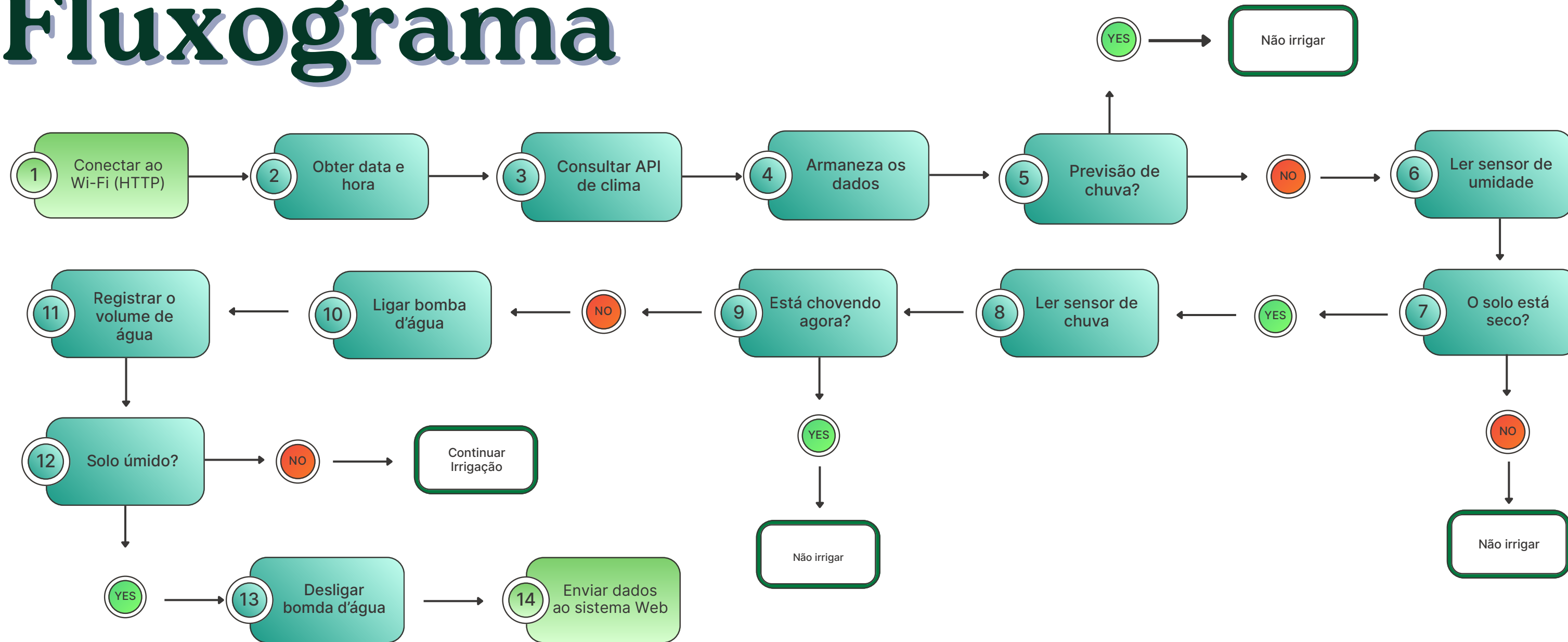
Metodologia



O dispositivo conecta-se ao Wi-Fi, obtém data e horário e realiza consultas periódicas à API de clima, utilizando essas informações a tomada de decisão.

O sistema monitora continuamente os sensores de umidade do solo e chuva local para verificar a real necessidade de irrigação.

Fluxograma



Código Arduino

```
#include <SoftwareSerial.h> // Para comunicar com o ESP
#include <ArduinoJson.h>     // Para criar o JSON de envio

// --- Pinos de Comunicação com o ESP-12E ---
// Conecte o pino 2 do Uno no TX do ESP
// Conecte o pino 3 do Uno no RX do ESP
#define ESP_RX_PIN 2
#define ESP_TX_PIN 3
SoftwareSerial espSerial(ESP_RX_PIN, ESP_TX_PIN);

// --- Pinos dos Sensores (Entradas Analógicas) ---
#define pino_sensor_chuva A1
#define pino_sensor_umidade A2
#define pino_sensor_fluxo A5 // Corrigido (original estava em A4)

// --- Pinos do Atuador (Saída Digital) ---
#define pino_valvula 5 // Corrigido (original estava em A5)

// --- Variáveis de Controle ---
unsigned long ultimoEnvio = 0;
const long intervaloEnvio = 5000; // Envia dados para o ESP a cada 5 segundos
String comandoESP = "";           // Armazena comandos vindos do ESP
```

```
void setup() {
  // Inicia o Serial Monitor (para debug no PC)
  Serial.begin(9600);
  Serial.println("Arduino Uno - Hub de Sensores - Iniciado.");
  Serial.println("Aguardando comandos do ESP...");

  // Inicia a comunicação com o ESP-12E
  espSerial.begin(9600);

  // Configura o pino da válvula como SAÍDA
  pinMode(pino_valvula, OUTPUT);
  digitalWrite(pino_valvula, LOW); // Garante que comece desligada

  // Pinos analógicos (A0-A5) são INPUT por padrão

  comandoESP.reserve(32); // Reserva 32 bytes para a string de comando
}

void loop() {
  // 1. Verifica se recebeu comandos do ESP
  receberComandosESP();

  // 2. Verifica se está na hora de ler e enviar os dados dos sensores
  unsigned long agora = millis();
  if (agora - ultimoEnvio >= intervaloEnvio) {
    ultimoEnvio = agora; // Reseta o timer
    lerEEenviarSensores();
  }
}
```

Código Arduino

```
63  /**
64   * @brief Lê todos os sensores e envia via JSON para o ESP.
65   */
66  void lerEEenviarSensores() {
67      // 1. Lê os valores brutos dos sensores
68      int umidade = analogRead(pino_sensor_umidade);
69      int chuva = analogRead(pino_sensor_chuva);
70      int fluxo = analogRead(pino_sensor_fluxo); // (Leitura de
71
72      // 2. Monta o JSON (usando ArduinoJson)
73      // StaticJsonDocument<200> reserva 200 bytes na memória p
74      StaticJsonDocument<200> doc;
75      doc["id_dispositivo"] = "uno_planta";
76      doc["umidade"] = umidade;
77      doc["chuva"] = chuva;
78      doc["fluxo"] = fluxo;
79
80      // 3. Envia o JSON para o ESP-12E
81      Serial.print("Enviando para ESP: ");
82      serializeJson(doc, Serial); // Mostra no PC o que vai env
83      Serial.println();
84
85      serializeJson(doc, espSerial); // Envia DE VERDADE para o
86      espSerial.println(); // Envia uma quebra de linha (LF) pa
87  }
```

```
90  * @brief Fica ouvindo a porta serial do ESP por comandos.
91  */
92  void receberComandosESP() {
93      while (espSerial.available() > 0) {
94          char c = espSerial.read();
95
96          if (c == '\n') { // '\n' (newline) marca o fim do comando
97              // Comando recebido, vamos processar
98              Serial.print("Comando recebido do ESP: ");
99              Serial.print(comandoESP);
100              Serial.println("");
101              comandoESP.trim();
102              // 2. Executa a ação
103              if (comandoESP == "LIGAR_VALVULA") {
104                  Serial.println("Acao: Ligando a valvula.");
105                  digitalWrite(pino_valvula, HIGH);
106              }
107              else if (comandoESP == "DESLIGAR_VALVULA") {
108                  Serial.println("Acao: Desligando a valvula.");
109                  digitalWrite(pino_valvula, LOW);
110              }
111
112              // 3. Limpa a variável para o próximo comando
113              comandoESP = "";
114
115          } else {
116              // Vai montando a string do comando caractere por caractere
117              comandoESP += c;
118          }
```

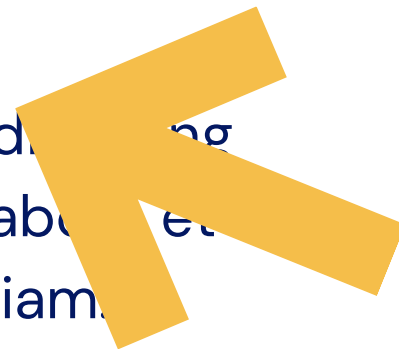

Sistema Web

Interpretation of Results

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Practical Implication

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.



**BOTA UMA BREVE
EXPLICAÇÃO DO
CÓDIGO AQUI**



Demonstração



OBRIGADO

