Sistema para coleta e armazenamento de dados climáticos

Trabalho de Conclusão do Curso de
Tecnologia em Sistemas para Internet

Nome do Aluno: Bruno Trevissoi do Nascimento
Orientador(a): André Marcelo Schneider

[1]Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Campus Porto Alegre
Av Cel Vicente, 281, Porto Alegre – RS – Brasil

trevissoibruno@gmail.com, andre.schneider@poa.ifrs.edu.br

Resumo. *Dados meteorológicas constituem uma série de variáveis como temperatura, umidade, precipitação e vento que influenciam a todos nós, pode provocar doenças ou acidentes, esse mais presentes na mídia em decorrência da urbanização com construções feitas em áreas com terreno instável. Na agricultura onde o clima afeta constantemente, os produtores devem possuir informações sobre o clima e quando essas são imprecisas acabam tratando a área produtiva de forma homogênea, aplicando uniformemente corretivos e agrotóxicos. O presente trabalho tem como objetivo a construção de um protótipo de sistema de monitoramento climática, usando sensores para coleta de dados ambientais relevantes para o público. Variáveis climáticas que serão coletadas e a partir desses dados pode vir a ser uma ferramenta poderosa para recursos humanos.*

1. Introdução

O clima está passando por transformações. Em certas regiões, a temperatura média está aumentando as chuvas mais torrenciais estão afetando diretamente a população com inundações e deslizamentos. A temperatura, chuva, vento, umidade tem efeitos diretos na vida das pessoas. Monitorar a quantidade de chuva pode ajudar no deslocamento da população no momento em que atingir um patamar de perigo. Epidemias podem surgir em locais que jamais foram vistas, doenças transmitidas por mosquitos que se proliferam em regiões quentes, prova que a temperatura está acentuada. Saber quando está quente, frio ou ameno ao sair de casa pode ajudar as pessoas a se preparar para um dia com temperaturas altas ou baixas ou chuvoso, ajudando no bem-estar social da população. Na agricultura o número de empresas rurais aumenta e torna a agricultura um mercado altamente competitivo. Isso exige do produtor rural um alto grau de especialização e de profissionalismo. O produtor deve ter a capacidade de coletar dados da área produtiva para controlar os insumos que aplica na produção, não mais aplicando a mesma quantidade para toda a área, mas espalhando os insumos a uma área específica da produção. Então deve ter um controle a partir de dados climáticos provenientes da lavoura para determinar os insumos aplicados.

Esse trabalho é voltado para o público em geral com o intuito de captar variáveis climáticas relevantes para uso privado ou público, provendo um sistema para captação delas, exemplo de uso pode ser na agricultura onde as variáveis climáticas incidem mais

significativamente, pragas podem surgir com o aumento da temperatura o agricultor por experiencia conhece as pragas que causam prejuízos a sua plantação.

O objetivo desse artigo é propor um sistema de coleta de dados climáticos onde os dados podem ser visualizados em tempo real a partir de um smartphone ou computador com acesso a internet, protótipo de unidade de coleta de dados meteorológicos. O protocolo usado entre cliente e servidor e MQTT (*Message Queue Telemetry Transport*) que será descrito no decorrer do artigo.

2. Trabalhos relacionados

Nesta seção serão apresentados alguns sistemas de coleta de dados meteorológicos similares ao proposto na presente pesquisa.

2.1. Sistema embarcado para aquisição de dados agrometeorológicos (SDA)

O sistema apresentado por [Sabo et al. 2011] é uma microestação meteorológica com diversos sensores coletando dados agrometeorológicos relevantes para viticultura. Monitora o desenvolvimento da uva no campo e doenças fúngicas. É construído com sensor de temperatura, umidade ar e solo, pluviômetro, anemômetro, molhamento foliar, luminosidade e XBee para transmissão dos dados e gravados em um banco de dados. O microcontrolador Arduino é usado para captar os sinais que chegam dos sensores.

2.2. Aplicação da estação meteorológica portátil no combate a incêndio florestal no estado de Goiás(EMPIF)

O trabalho apresentado por [dos Santos 2015] propõe o uso de uma estação meteorológica portátil para medir os parâmetros que influenciam o comportamento do fogo em um incêndio florestal. São elas: temperatura, umidade relativa do ar, velocidade do vento e precipitação, pressão atmosférica. O artigo não trata da construção da estação, mas sim usando estações prontas de determinados fabricantes onde é apresentado no artigo. As estações serão usadas para ajudar os bombeiros no combate incêndios florestais que a partir dos dados, decisões especificas serão tomadas para segurança e combate ao fogo.

2.3. Desenvolvimento de sistema automatizado de baixo custo para coleta e armazenamento de dados de variáveis climáticas: aplicação no ambiente agrícola(DSB)

[Palmieri 2009] propõe a construção de uma estação meteorológica de baixo custo para monitoramento da lavoura, para reduzir a perdas que existem na agricultura que acarretam em grandes prejuízos não só para o produtor agrícola como também para o consumidor final. Os sensores de temperatura, umidade e radiação solar conectados ao microcontrolador Basic Step 1, Basic Step é um microcontrolador brasileiro criado pela Tato Equipamentos Eletrônicos ele foi baseado no famoso microcontrolador BASIC Stamp criado pela Norte Americana Parallax sua programação é feita em BASIC. O trabalho conclui que o sistema apresentou resultados satisfatórios no ambiente agrícola, o custo se apresentou baixo. Os sensores em longos períodos apresentaram erros, que podem ser desprezados por não alterar o resultado, diferenças de leitura em 16% comparados a estações meteorológicas governamentais. O sensor de LDR que ao receber luminosidade altera a resistência em seus terminais ou seja quanto maior a incidência de luz menor a resistência e na ausência dela, maior a resistência.

2.4. Estações meteorológicas de código aberto: Um projeto de pesquisa e desenvolvimento tecnológico(EMDCA)

O estudo apresentado por [Pezzi 2015] propõe desenvolver estações meteorológicas modulares de código aberto para formar uma rede cidadã de coleta, registro, armazenamento e compartilhamento de dados meteorológicos e ambientais em microclimas. O projeto de pesquisa e desenvolvimento conta com a participação de estudantes de diferentes níveis de ensino e áreas do conhecimento. O principal objetivo do projeto é introduzir estudantes universitários e da educação básica no estudo do tempo e do clima, visando o desenvolvimento de concepções e competências que os permitam compreender o ambiente em que vivem através do domínio de tecnologias livres e suas aplicações. O estudo conclui que vários desafios têm que ser superados como parcerias com escolas para definir os locais de instalação das estações e a principal contribuição do projeto é a formação de recursos humanos, envolvendo desde estudantes da Educação Básica a pesquisadores de diferentes áreas.

A figura 1 apresenta o equipamento desenvolvido.

Figura 1. Protótipo Meteorologico

2.5. Comparação entre os sistemas

A Tabela 1 apresenta uma comparação entre os sistemas citados anteriormente e o sistema proposto SCADV(Sistema para coleta e armazenamento de dados climáticos). Compara parâmetros de medição, portabilidade e como os dados serão resgatados: se via internet ou conectando algum outro dispositivo de armazenamento para coleta das informações contidas nos sistemas.

	SDA	EMPIF	DSB	EMDCA	SCADV
Temperatura do Ar	X	X	X	X	X
Umidade do Ar	X	X	X	X	X
Luminosidade	X		X		
Velocidade Do Vento	X	X			X
Direção do Vento	X				X
Molhamento foliar	X				
Pluviosidade	X				X
Umidade do Solo	X				
Altímetro		X			
Ponto De Orvalho		X			
Chuva Estado					X
Pressão Atmosférica		X		X	
Portátil		X			
Cartão SD	X	X			
Internet	X			X	X
Pen drive			X		

Tabela 1. Tabela apresenta comparação entre os sistemas

3. Sistema SCADV(Sistema para coleta e armazenamento de dados climáticos)

Neste capitulo é apresentada a estrutura do protótipo proposto, as funcionalidades: medição de temperatura do ar, umidade do ar, medição de quantidade de chuva e seu estado se esta ou não chovendo, velocidade do vento em Km/h e por ultimo a direção do vento, podem ser visualizados via internet podendo ser acessados de um smartphone ou computador conectado a internet. Os sensores de umidade, temperatura conectados ao NodeMcu e os sensore de vento, chuva e direção conectados ao Arduino, interpretam as informações enviadas por cada um dos sensores. Os microcontroladores enviam os dados para a plataforma Cayenne usando protocolo MQTT. O usuário requisita os dados a Cayenne que envia para o app do smartphone. O Cayenne possui, além de uma app para smartphones, um site que pode ser acessado pelo navegador.[MyDevices 2017]

3.1. NodeMcu

O NodeMCU é um modulo que combina o chip ESP8266 com a placa NodeMCU é programada via IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado) na linguagem C. O chip ESP8266 é um dispositivo que consiste de um microprocessador ARM de 32 bits com memória flash integrada e WIFI já incorporado a ele. Apenas o chip seria difícil criar uma plataforma de fácil desenvolvimento. Para isso existe o NodeMCU, que é uma plataforma de hardware e software que incorpora outros componentes e o chip ESP8266, que ajuda na programação dele facilitando o desenvolvimento.

3.2. Arduino UNO

O Arduino Uno é uma placa de microcontrolador baseado no ATmega328 facilmente conectada à um computador e programada via IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado) na linguagem C, foi criado em

2005 por um grupo de 5 pesquisadores : Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis, Depois de programado, o microcontrolador pode ser usado de forma independente.

3.3. W5100

Para prover acesso on-line para o Arduino o shield W5100 foi acoplado a ele, o Ethernet Shield é baseado no chip WIZnet ethernet W5100 que fornece acesso à rede (IP) nos protocolos TCP ou UDP.

3.4. DHT22

O RHT03 (também conhecido como DHT22) é um sensor de temperatura e umidade com comunicação digital de um fio[Mota 2017], utiliza um termistor e um sensor capacitivo para medir temperatura e umidade do ar ambiente[Robocore 2017]. Pode ser conectado a diversas plataformas NodeMCU, Arduino e Raspberry Pi entre outras é necessário instalar a biblioteca antes de utilizá-lo. As características dele são, tensões de 3,3 a 6V de trabalho, faixa de leitura de umidade 0 a 100%, faixa de temperatura -40 a 125 graus Celsius.

3.5. Sensor de chuva

O sensor de chuva LM393, possui ligação com uma placa composta por trilhas que percorre a placa sensor, ligado ao microcontrolador. O LM393 envia dois tipos de sinais, analógico que possui escala de 0 a 1024 ou 1023 valores e digital. Quando a placa esta seca o sinal digital é constante em nível lógico alto (HIGH) ou 1, a partir do momento que ela começa a receber pingos, o nível lógico vai para baixo (LOW) ou 0. Para o sinal analógico quando estiver seca o valor é máximo 1024, no momento que receber pingos a valor começa a decair conforme a quantidade de água estiver presente na placa sensor.

3.6. Pluviômetro

O Pluviômetro de Báscula é um módulo mecânico eletrônico, que mede a precipitação de chuvas em certa área em que está instalado. Conta com uma báscula para cálculo da precipitação da chuva, onde a cada 0,25mm o sensor presente no equipamento emite um pulso, o qual pode ser interpretado e lido pelo Arduino.

3.7. Biruta

O indicador de direção do vento ou biruta possui internamente um conjunto de 8 sensores onde cada um está posicionado em uma localização diferente, apresentando valores de resistências diferentes para cada posição.

3.8. Anemômetro

O Anemômetro possui copos conectados a um único eixo de rotação quanto mais rajadas de vento, mais voltas ele dará, e maior é a velocidade do vento, internamente junto ao eixo de rotação é conectado um imã que cada vez que passa pelo reed switch, manda um pulso para o Arduino e o mesmo realiza a conversão necessária para exibir a velocidade do vento, reed switch é um interruptor ou chave que pode ser acionado por um campo magnético, que neste caso é um imã.

A Figura 2 apresenta os sensores descritos.

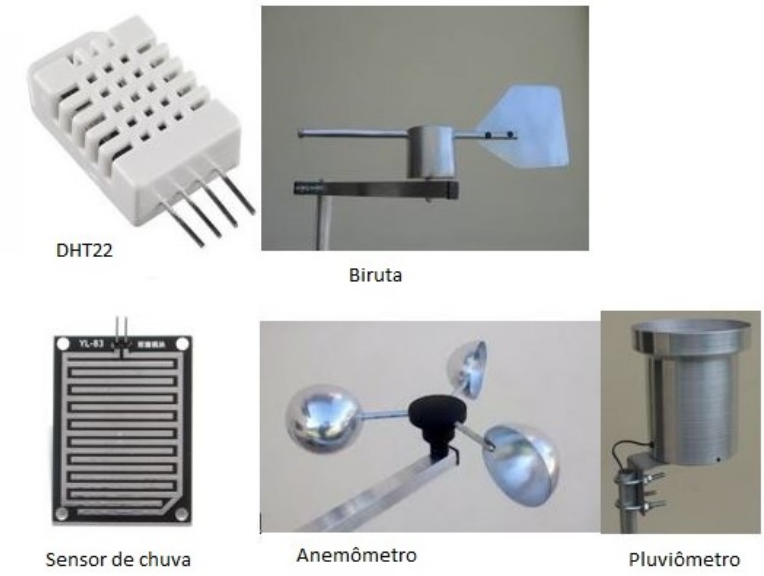

Figura 2. Sensores

3.9. Protocolo de Comunicação

Para envio dos dados foi usado o protocolo MQTT (*Message Queue Telemetry Transport*), criado pelo Dr. Andy Stanford-Clark da IBM e Arlen Nipper da Arcom em 1999. O protocolo foi desenvolvido pensando no padrão Publish/Subscribe com o propósito de ser simples, leve, que fosse capaz de ser utilizado até mesmo por dispositivos com restrições de memória e processamento e que estivessem em redes lentas, com alta latência ou pouco confiáveis. Com o protocolo MQTT, foi estabelecido a comunicação com o *Broker* (uma espécie de servidor), publicando as informações nele. Com as informações no Broker, o usuário, com um celular ou computador, realiza leitura, tendo acesso assim às informações de temperatura, umidade e chuva. O protocolo MQTT define dois tipos de conexão, um cliente que são os sensores usados neste artigo e um broker que é a Cayenne também usado neste trabalho, o broker recebe todos os dados dos cliente e os distribui a todos os clientes que requisitarem esses dados, cliente este que é o usuário na posse de um smartphone com a Cayenne instalado e recebe esses dados, mas como é feito esses envio e requisição e como ele sabe qual sensor esta enviado as informações. Os sensores enviam os dados fazendo um *publish* publicam a informação no broker e um tópico, um tópico com o nome "temperatura_data" o DHT22 publica os dados e junto desses dados o tópico e todos os que requisitarem fizerem um *subscribe* neste tópico "temperatura_data" terão acesso a esses dados.

A figura 3 apresenta o protocolo MQTT.

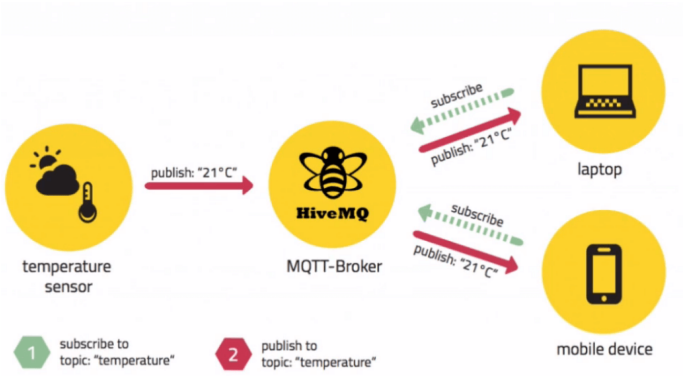

Figura 3. Protocolo MQTT

3.10. Cayenne: plataforma de prototipação

Cayenne é o intermediário entre o cliente e servidor, ele interpreta todas as requisições do cliente com ao servidor, que permite gerenciar e controlar sensores e atuadores a partir do smartphone com sistemas Android e IOS ou na internet de qualquer computador. Pertence a empresa myDevices para soluções em Internet das Coisas(Iot). O broker suporta diversos módulos e entre eles está o NodeMCU, usado no protótipo [MyDevices 2017].

A figura 4 apresenta a arquitetura do SCADV.

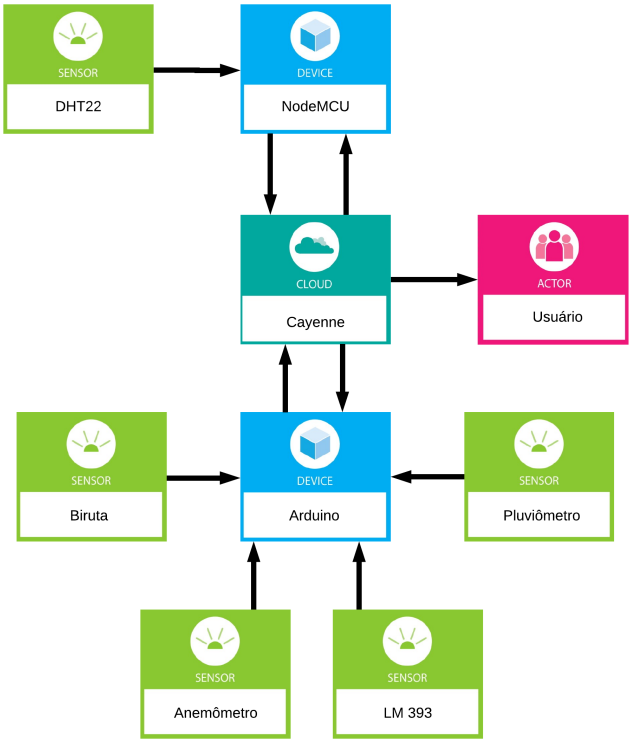

Figura 4. SCADV

4. Desenvolvimento

Para desenvolver o sistema de coleta de dados meteorológicos, foi usado as plataformas, dispositivos e sensores descritos anteriormente: Cayenne, Arduíno UNO, NodeMCU, DHT22, sensor de chuva LM393, pluviômetro, anemômetro e sensor de direção do vento. Arduíno e NodeMCU ambos programados em C usando o mesmo editor bastando apenas trocar o microcontrolador na hora de gravar o programa, por mais que eles sejam programados na mesma linguagem, são microcontroladores diferentes. O Arduino não estava no planejamento inicialmente e foi adicionado ao sistema por ser capaz de receber sinais PWM, característica não suportada pelo NodeMCU. Alguns dos sensores utilizados trabalham com sinais PWM. Desta forma, o W5100 foi adicionado para dar suporte de rede ao Arduino e enviar as leituras para o Cayenne. Optou-se por esta interface que usa cabo Ethernet por prover mais confiabilidade. O NodeMCU continuou sendo usado por apenas um sensor, o DHT22. Todos as leituras vindas dos sensores são processadas pelas plataformas Arduino e NodeMCU e posteriormente publicadas em tempo real na Cayenne que é o broker MQTT. A Cayenne permite a exibição dos valores diretos ou gráficos que são gerados conforme os dados enviados permitindo análise do histórico das medições. Não foi possível utilizar todos os sensores com o Arduino por questões de limitação da memória, não suportou o número de sensores. O Arduino UNO necessita de 10% de memória livre, caso esse valor seja menor pode haver problemas de estabilidade, todos os sensores usaram 96% de memória do Arduino, isso se deve as bibliotecas usadas. O Cayenne e o DHT22 para implementá-los é necessário incluir suas respectivas bibliotecas, essas que ocupam um espaço considerável, a divisão dos sensores , para dois microcontroladores foi a solução encontrada, com essa divisão o DHT22 foi implementado no NodeMCU, com a mudança o uso do Arduino caiu para 84% prova o quanto o Cayenne consome de recursos dos microcontroladores.

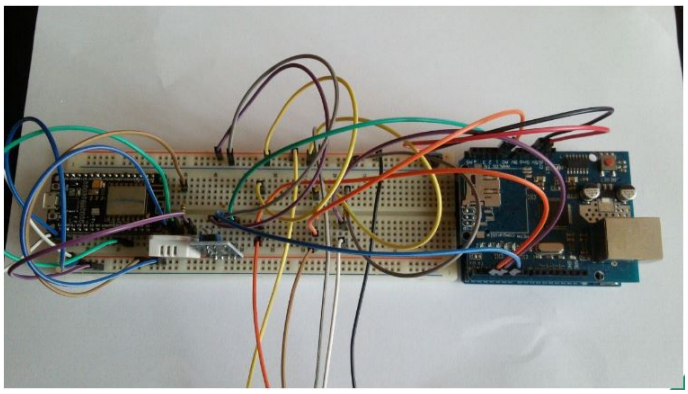

Figura 5. Sistema SCADV e seus controladores

5. Testes e Resultados Finais

Os testes finais foram realizados em ambiente fechado, DHT22 envia os dados de temperatura e umidade do ar para o NodeMCU, pluviômetro, biruta, anemômetro e sensor de chuva cada um envia a quantidade de chuva, direção do vento, velocidade do vento e estado da chuva respetivamente ao Arduino, o Arduino e NodeMCU processam os dados transmitem a Cayenne. Para testar as leituras, o estado dos sensores era alterado manualmente: forçando deslocamento de ar para simular vento e borrifando água para simular

chuva, o único sensor testado em ambiente aberto foi o sensor de chuva onde para medir a intensidade da chuva que caia em sua placa sensor. Após a confirmação das leituras, o W5100 foi acoplado ao Arduino e este foi conectado à Internet. A plataforma Cayenne é formada por canais de comunicação com cada sensor, onde cada um deve estar ocupando um canal. Essa configuração deve ser inserida na programação dos microcontroladores. No momento que o Arduino UNO e NodeMcu são conectados à Internet, a plataforma Cayenne passa a receber as leituras e insere no painel online. Após os testes com os microcontroladores e sensores acoplados. Os testes finais foram concluídos e os resultados foram considerados positivos, cada um dos componentes se comportou da maneira que se esperava. O DHT22 informou a temperatura e umidade do ar de forma correta, condizentes com a temperatura do ambiente de testes, não marcou nada que esteja fora dos limites reais como 100 graus ou 0 graus. O pluviômetro mediu exatamente a quantidade de água que foi borrifada em sua estrutura de captação. A biruta indicou a direção do vento naquele momento, mesmo alterando seu estado manualmente. O Anemômetro mediu a velocidade do vento, quando seu estado era alterado, e quando em repouso e medição indicava 0. O sensor de chuva indicava chuva quando havia chuva e não indicava quando não havia, seu funcionamento foi comprovado, na plataforma Cayenne suas leituras eram informadas. Problemas surgiram o Arduino, que perdia conexão constantemente a solução foi definir as configurações de IP(*Internet Protocol*) que é o endereço na rede a que ele está conectado, *Gateway* que representa o default da rede, MAC(*Media Access Control*) que representa a interface física de conexão que neste caso é o w5100 e DNS(*Domain Name System*) resolve os nomes na rede, cada um deles foi configurado diretamente no Arduino, estabilizando a conexão.

Figura 6. Plataforma Cayenne

Figura 7. Gráfico gerado pela plataforma

6. Considerações Finais

Este artigo descreveu o desenvolvimento de um sistema de coleta de dados climáticos para o uso do público em geral, o sistema pode ser usado de diversas formas, pode ser adicionado ou retirado algum sensor que não seja do interesse do usuário

coletar. Os benefícios são muitos como por exemplo a agricultura que sofre diretamente a variação climática, uso exagerado de agrotóxicos pode ser controlado se o agricultor possuir acesso a variáveis climáticas pragas que assolam a lavoura se desenvolvem numa determinada temperatura, o próprio agricultor sabe quando elas surgem, ele tendo acesso as variáveis climáticas dará a ele maior controle sobre sua produção. Uma pessoa que antes de sair de casa verifica as condições climáticas da sua região e a partir delas pode se preparar para um dia chuvoso. Em trabalhos futuros com a continuidade do sistema apresentado neste artigo pode ser desenvolvido um banco de dados que apresente os dados de uma forma diferente da Cayenne, um clima atípico estudos a partir dos dados coletados e armazenados para entender as mudanças bruscas do clima. Desenvolver um *broker* comunitário que qualquer pessoa pode e consultar a temperatura caso uma estação esteja próxima, e muitos outros experimentos podem ser feitos para ajudar a população.

Referências

dos Santos, Y. C. (2015). Aplicação da estação meteorológica portátil no combate a incêndio florestal no estado de goiás. *Corpo de Bombeiros Militar de Goias (GO)*, 5(2):01–12. https://goo.gl/iPX9BX.

Mota, A. (2017). Portal vida de silicio. `https://portal.vidadesilicio.com.br/sensores-dht11-dht22-biblioteca-arduino/`. Accessed: 2017-11-28.

MyDevices (2017). Cayenne. `https://mydevices.com/about/`. Accessed: 2017-10-20.

Palmieri, A. (2009). Desenvolvimento de sistema automatizado de baixo custo para coleta e armazenamento de dados de variáveis climáticas: aplicação no ambiente agrícula. *Piracicaba : Escola Superior de Agricultura Luiz de Queiroz, Universidade de São Paulo, 2009.*, 5(2):01–12. http://www.teses.usp.br/teses/disponiveis/11/11131/tde-18022010-142019/pt-br.php.

Pezzi, R. (2015). Estações meteorológicas de código aberto: Um projeto de pesquisa e desenvolvimento tecnológico. *Revista Brasileira de Ensino de Física*, 5(2):01–12. https://goo.gl/qymojK.

Robocore (2017). Robocore. `https://www.robocore.net/loja/produtos/nodemcu-esp8266-12-v2.html`. Accessed: 2017-11-28.

Sabo, P. H., Cruz, E. H. M., Martini, J. A., Gonçalves, P. C., and Leonardo, E. J. (2011). Sistema embarcado para aquisição de dados agrometeorológicos. *SBESC*, 5(2):01–12. https://goo.gl/uj7U13.

Anexo A CÓDIGO ARDUINO

```
#define CAYENNE_PRINT Serial
#include <CayenneMQTTEthernet.h>

char username[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
char password[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
char clientID[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";

byte arduino_mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress arduino_ip(x, x, x, x);
IPAddress dns_ip(x, x, x, x);
IPAddress gateway_ip(x, x, x, x);
IPAddress subnet_mask(x, x, x, x);

int interrupcao = 1;

// Pin definitions
# define Hall sensor 2          //  Pino digital 2

//Constantes LM
int pino_d = 5;
int pino_a = A1;

//Constantes Pluviometro
const int REED = 3;

//Constantes Anemometro
const float pi = 3.14159265;          // Numero pi
int period = 5000;                   // Tempo de medida(miliseconds)
int delaytime = 2000;                // Tempo entre amostras (miliseconds)
int radius = 147;
// Raio do anemometro(mm)

//Constantes Direcao do Vento
int pin=A0;
float valor =0;
int Winddir =0;

//Variaveis LM
String display_temp;
String display_humid;
String chuva;

// Variaveis Pluviometro
int val = 0;                         //Valor atual do switch
int old_val = 0;                     //Valor antigo do switch
```

```
int REEDCOUNT = 0;                //Esta e a variavel que contem a contage

// Variaveis Anemometro
unsigned int Sample = 0;    // Numero da amostra
unsigned int counter = 0; // Contador do sensor
unsigned int RPM = 0;           // Rotacoes por minuto

float speedwind = 0;          //vento em (m/s)
float windspeed = 0;            // vento em (km/h)

//--------------------------------------------------------------------------

void setup() {
  Serial.begin(9600);
  Cayenne.begin(username, password, clientID, arduino_ip, dns_ip, gatew

  //Anemometro
  pinMode(2, INPUT);
  digitalWrite(2, HIGH);

  //Pluviometro
  // initializa o pino do switch como entrada
   pinMode (REED, INPUT_PULLUP);
   //digitalWrite(2,HIGH);

//LM
pinMode(pino_d, INPUT);
pinMode(pino_a, INPUT);

attachInterrupt(interrupcao, funcionInterrupcao, FALLING);

}

void loop() {

  while(true){

//LM
getChuva();
delay(1000);

 //Direcao do Vento
  valor = analogRead(pin)* (5.0 / 1023.0);
  Cayenne.virtualWrite(3, valor);
  Serial.print("leitura do sensor :");
  Serial.print(valor);
```

```
  Serial.println(" volt");

  if (valor <= 0.27) {
  Winddir = 315;
  }
  else if (valor <= 0.32) {
  Winddir = 270;
  }
  else if (valor <= 0.38) {
  Winddir = 225;
  }
  else if (valor <= 0.45) {
  Winddir = 180;
  }
  else if (valor <= 0.57) {
  Winddir = 135;
  }
  else if (valor <= 0.75) {
  Winddir = 90;
  }
  else if (valor <= 1.25) {
  Winddir = 45;
  }
  else {
  Winddir = 000;
  }
   Serial.print("Direcao do vento :");
   Serial.print(Winddir);
   Cayenne.virtualWrite(4, Winddir);
   Serial.print(" graus");
   Serial.println();
   delay (1000);

   //Anemometro
    Sample++;
  Serial.print("Velocidade do Vento: ");
  windvelocity();

  RPMcalc();
   //*************************************************************
//print m/s
  WindSpeed();
 //*************************************************************
//print km/h
  SpeedWind();
```

```
  Serial.print(speedwind);
  Cayenne.virtualWrite(0, speedwind);
  Serial.print(" [km/h] ");
  Serial.println();
  Serial.print("**********************************************************
  Serial.println();

  //delay(delaytime);

  }
  Cayenne.loop();
}

//Funcoes Anemometro

void windvelocity(){
  speedwind = 0;
  windspeed = 0;

  counter = 0;
  attachInterrupt(0, addcount, RISING);
  unsigned long millis();
  long startTime = millis();
  while(millis() < startTime + period) {
  }
}

void RPMcalc(){
  RPM=((counter)*60)/(period/1000);  //Calcula rotacoes por minuto(RPM)
}

void WindSpeed(){
  windspeed = ((4 * pi * radius * RPM)/60) / 1000;
// Calcula velocidade em m/s

}

void SpeedWind(){
  speedwind = (((4 * pi * radius * RPM)/60) / 1000)*3.6;
// Calcula velocidade em km/h
}
void addcount(){
  counter++;
}
//--------------------------------------------------------------------------
//Funcoes Dht22
```

```
void getChuva(){
  int chuva_a = analogRead(pino_a);
  int chuva_d = digitalRead(pino_d);
  if(chuva_a < 300){
  Serial.print("Chuva forte");
  chuva = "Chuva forte";
  }
  else if(chuva_a < 500){
  Serial.print("Chuva moderada");
  chuva = "Chuva moderada";
  }
  else{
  Serial.print("Sem chuva");
  chuva = "Sem chuva";
  }
  Serial.print(" ");
  Serial.println(chuva_a);
  Cayenne.virtualWrite(1, chuva_a);
  Cayenne.virtualWrite(2, chuva_d);
}
void funcionInterrupcao() {
   //Pluviometro
   // ler o estado do switch pelo pino de entrada:
  //delay(5000);
  val = digitalRead(REED);

 if ((val == LOW) && (old_val == HIGH)){
//Verifica o status de foi alterado
   delay(10);                      // Delay para esperar leitura.
   REEDCOUNT = REEDCOUNT + 1;    //Adiciona 1 a contagem
   old_val = val;                //valor antifo igual a valor atual
        Serial.println("Medida de chuva (contagem): ");
        Serial.print(REEDCOUNT);//*0.2794);
        Serial.println(" pulso");
        Serial.print("Medida de chuva (calculado): ");
        Serial.print(REEDCOUNT*0.25);
        Cayenne.virtualWrite(5, REEDCOUNT*0.25);
        Serial.println(" mm");
        }
      else {

   old_val = val;                // Caso o status mude, entao nao faca na

}
  }
```

Anexo B CÓDIGO DHT22

```
#define CAYENNE_PRINT Serial
#include <CayenneMQTTESP8266.h>
#include <DHT.h>

#define DHTPIN D3
#define DHTTYPE DHT22
String display_temp;
String display_humid;
String chuva;

DHT dht(DHTPIN, DHTTYPE);

// Your network name and password.
char ssid[] = "NOME DA REDE";
char wifiPassword[] = "SENHA*";

// Cayenne authentication info. This should be obtained from the Cayenn
char username[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
char password[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
char clientID[] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";

void getTemperatura() {

float temperatura = dht.readTemperature(false);
if (isnan(temperatura)) {
Serial.println("Falha no sensor!");
return;
}
display_temp = temperatura;
Serial.print("Temperatura: ");
Serial.print(temperatura);

Cayenne.virtualWrite(0, temperatura, CELSIUS);
}

void getHumidade() {
float umidade = dht.readHumidity();

if (isnan(umidade)) {
Serial.println("Falha na leitura do sensor!");
return;
}
display_humid = umidade;
Serial.print("Umidade: ");
```

```
Serial.print(umidade);
Serial.println(" %");
Cayenne.virtualWrite(1, umidade);
}
void setup() {

Serial.begin(9600);
dht.begin();
Cayenne.begin(username, password, clientID, ssid, wifiPassword);
}
void loop() {
getTemperatura();
getHumidade();
delay(3000);
}
```