FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS – FATECS
ENGENHARIA DA COMPUTAÇÃO

FLÁVIA RESENDE PEIXOTO
2175029/5

MONITORAMENTO DE AMBIENTE IoT

BRASÍLIA
2020

FLÁVIA RESENDE PEIXOTO

MONITORAMENTO DE AMBIENTE IoT

Trabalho de Conclusão de Curso (TCC) apresentado como um dos requisitos para a conclusão do curso de Engenharia Civil, Elétrica ou de Computação do UniCEUB– Centro Universitário de Brasília

Orientador (a)**: Me. Ivandro da Silva Ribeiro**

BRASÍLIA
2020

FLÁVIA RESENDE PEIXOTO

MONITORAMENTO DE AMBIENTE IoT

Trabalho de Conclusão de Curso (TCC) apresentado como um dos requisitos para a conclusão do curso de Engenharia Civil, Elétrica ou de Computação do UniCEUB – Centro Universitário de Brasília

Orientador (a)**: Me. Ivandro da Silva Ribeiro**

Brasília, 2020.

BANCA EXAMINADORA

_____

Me. Ivandro da Silva Ribeiro Mestre.
Orientador (a)

_____

Dr. Honório Assis Filho Crispim.
Examinador (a)

_____

Me. Francisco Javier de Obaldia Diaz Mestre.
Examinador (a)

MONITORAMENTO DE AMBIENTE IOT
IoT environment monitoring

Flávia Resende Peixoto[1], Ivandro da Silva Ribeiro[2], Francisco Javier de Obaldia Diaz[3], Honório Assis Filho Crispim[4]

Resumo

Este projeto tem como principal objetivo desenvolver um protótipo, utilizando-se de sensores de monitoramento de ambiente, construído a partir de componentes de baixo custo. Para isso, foram utilizados a placa NodeMCU, o sensor PIR, sensor MQ-7 de detecção de fumaça e gás e o sensor de umidade e temperatura DHT11. Esse sistema irá captar os dados fornecidos pelos sensores e publicar as informações no broker MQTT para que o ambiente possa ser monitorado pelo usuário. Um sistema simples que proporcionará tranquilidade para moradores e, consequentemente, mais segurança e conforto ao usuário.

Palavras-chave: Casa inteligente, monitoramento, sistema de monitoramento de ambiente, sensor PIR, sensor MQ-7, sensor DHT11**.**

Abstract: This project has as main goal to develop a prototype, using environment monitoring sensors, built with low cost components. For this, we used the NodeMCU board, the PIR sensor, MQ-7 smoke and gas detection sensor and the DHT11 humidity and temperature sensor. This system will capture the data provided by the sensors and publish the information in the MQTT broker so the environment can be monitored by the user. A simple system that will provide tranquility for residents and, consequently, more safety and comfort for the user.

keywords: Smart home, monitoring, environment monitoring system, PIR sensor, MQ-7 sensor, DHT11 sensor.

1 INTRODUÇÃO

Garantia de segurança é atualmente uma das preocupações prioritárias, especialmente nas grandes cidades, que vêm apresentando um crescente aumento nos números nos últimos anos, pois invasões e furtos residenciais são os casos mais frequentes.(ESTADÃO, 2019) Este projeto tem por objetivo contribuir para a constante busca de mais tranquilidade e bem-estar da sociedade.

O Arduino, plataforma de hardware livre para prototipagem, é amplamente utilizado

[1] UniCEUB, aluno.
[2] UniCEUB, orientador.
3 UniCEUB, primeiro examinador.
[4] UniCEUB, segundo examinador.
Aqui você deve inserir os dados referentes ‘às instituições às quais os autores pertencem

no desenvolvimento de projetos de eletrônica que envolvam linguagem programação, microcontrolador e linhas de entrada e saída para conexão de periféricos externos. Alguns dos benefícios e motivos para escolha do Arduino é o baixo custo de implementação, a flexibilidade no desenvolvimento e o vasto conteúdo disponível na comunidade open source.(AVED, 2017)

Mesmo não contando com qualquer recurso de rede, o Arduino originalmente desenvolvido pela empresa italiana Smart Projects e a estadunidense SparkFun Electronics, possui shields capazes de adicionar recursos e expandir as possibilidades de desenvolvimento de protótipos.

Ao longo dos anos, diversos fabricantes também lançaram no mercado versões variantes do Arduino, como o NodeMCU que possui o módulo ESP8266, estendendo a capacidade de comunicação por Wi-Fi nativamente por meio de conexões TCP/IP e dispensando a necessidade de adicionar um shield para estabelecer a comunicação de rede.

Sendo a Internet das Coisas ou IdC (Internet of Things ou IoT em inglês) um conceito cada vez mais popular de interconexão digital de dispositivos do cotidiano com a rede, faz-se necessário adotar protocolos de comunicação que supram a necessidade dos dispositivos de transmitirem dados do ambiente, obtidos através de diferentes tipos de sensores. (AGÊNCIA BRASIL, 2019)

Como exemplo dessa aplicação, destaca-se o protocolo MQTT amplamente utilizado para permitir a comunicação de dispositivos na Internet das Coisas.

O protocolo MQTT (Message Queue Telemetry Transport) foi inventado e desenvolvido pela empresa IBM nos anos 90, tornando-se o padrão para comunicações de IoT ao suportar a comunicação assíncrona entre as partes, ou seja, desacopla o emissor do receptor da mensagem tanto no espaço quanto no tempo, viabilizando sua implementação em ambientes de rede não confiáveis. (IBM, 2017)

Sendo um protocolo de rede leve e flexível, a utilização do MQTT pode ser aplicada em dispositivos de hardware altamente limitados e em redes com baixa largura de banda e alta latência.

No monitoramento de ambiente, será utilizado o sensor de presença PIR, sensor de gás e fumaça MQ-7, e o sensor de umidade e temperatura DHT11 para que seja possível monitorar, por exemplo, o cômodo de entrada da residência.

Conforme exposto nos parágrafos introdutórios, este projeto tem como objetivo a construção de um protótipo capaz de monitorar um ambiente, como o cômodo de uma casa. Ele monitora um eventual vazamento de gás, observando o nível de CO no ambiente, e verifica a existência de movimento, com o sensor PIR. Além disso, controla a temperatura e umidade do local com o sensor DHT11, enviando essas informações para um protocolo MQTT e para um aplicativo Blynk, com interface para que o usuário possa monitorar seu ambiente de qualquer lugar com conexão de internet.

2 REVISÃO BIBLIOGRÁFICA

2.1 Trabalhos relacionados

No decorrer do trabalho de conclusão de curso, foi proposta uma solução que integre sensores e realize o monitoramento do ambiente de forma que seja capaz de notificar o usuário do sistema de acordo com gatilhos pré-programados, como variação de temperatura, umidade, gás CO, e quando houver algum movimento no local. (DE MELLO, 2016)

A cada dia que passa, milhares de dispositivos novos são conectados à Internet. Entre 2016 e 2017 a quantidade de dispositivos conectados à Internet ultrapassaram a quantidade de pessoas no mundo.” Pesquisas apontam que até o fim de 2020, a quantidade de dispositivos será equivalente a 6,5 vezes a quantidade de pessoas, não relacionada somente aos dispositivos pessoais. Atualmente, cerca de 75% dos dispositivos conectados à Internet são para usos pessoais e esse número tende a cair para 25%. “(TUNG, 2017).

Toda essa quantidade de dispositivos conectados se dá devido à Internet das Coisas. O termo Internet das Coisas é utilizado para definir o uso de sistemas embarcados conectados à Internet, onde se torna capaz de comunicar com outros dispositivos, serviços ou pessoas em escala global. (TECHTUDO, 2014)

2.2 IoT – Internet das Coisas

Utilizado pela primeira vez em 1999, por Kevin Ashton, o termo Internet das Coisas se refere ao modo em que objetos físicos se comunicam entre si na internet. Em relação ao agronegócio ao utilizar IoT obtêm-se vários benefícios, já que ao ter uma visão geral sobre seu negócio é possível organizá-lo de uma maneira mais eficaz. (COSTA, 2018)

Quando IoT é implementado em um campo, é possível estabelecer uma rede de sensores que são conectados uns aos outros. Os dados captados por esses sensores são apresentados para o agricultor, que consegue analisar e aplicá-los para obter o melhor resultado possível. (MUNDO MAIS TECH, 2020)

Figura 1 – Esquemático IoT

Fonte: Microcontroladores-c, 2018.

Na figura 1 mostra o esquemático da conexão do Broker MQTT com o smartphone, por exemplo, e o NodeMCU.

2.3 Sensores

O sensor é basicamente um dispositivo que tem a função de detectar e responder com eficiência algum estímulo. Existem vários tipos de sensores que respondem a estímulos diferentes como, por exemplo: calor, pressão, movimento, luz e outros. Depois que o sensor recebe o estímulo, a sua função é emitir um sinal que seja capaz de ser convertido e interpretado pelos outros dispositivos.

2.3.1 Sensor de presença PIR Modelo

Figura 2 – Sensor de Presença PIR

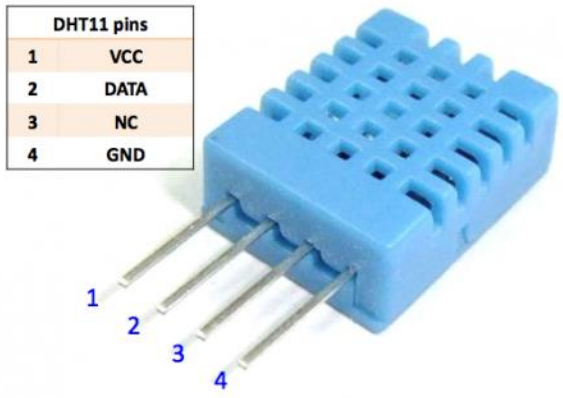
Fonte: FlipeFlop, 2020.

Um dos sensores mais comuns usados para detectar presença são os sensores PIR (Passive Infrared Sensor, ou Sensor Infravermelho Passivo). No Arduino, temos o módulo PIR DYP-ME003, que reúne numa mesma estrutura o sensor PIR, além dos circuitos necessários para seu ajuste e controle do sinal de saída. (ARDUINOECIA, 2014)
Esse módulo é composto internamente por duas faixas com material sensível ao infravermelho. Na parte externa há uma espécie de capa/tampa que na verdade é uma lente Fresnel.

2.3.2 Sensor de umidade DHT11

Este sensor é um dos componentes mais utilizados em projetos que envolvem medição de temperatura e umidade ambiente. Além disso, faz medições de temperatura de 0º até 50º Celsius e mede a umidade do ar nas faixas de 20% a 90%. A precisão (margem de erro) do sensor para medição de temperatura é de aproximadamente 2º Celsius e para umidade é de 5%.(FLIPEFLOP, 2019)

Figura 3 – Sensor DHT11



Fonte: Vida de Silício, 2020.

Na figura 3 está o sensor DHT11 com seus respectivos pinos.

2.3.3 Sensor de gás e fumaça MQ-7

É um módulo eletrônico desenvolvido com a finalidade de detectar a presença de gás inflamável/fumaça em determinado ambiente. A partir da detecção feita pelo sensor, o microcontrolador que estiver ligado ao módulo será notificado e poderá tomar uma ou várias ações que o usuário determinar. Dentre os gases que o sensor pode detectar, podemos destacar: gás natural, CO, metano, propano, butano, GLP e hidrogênio. Segue uma foto do componente na figura 4.

Figura 4 – Sensor MQ-7

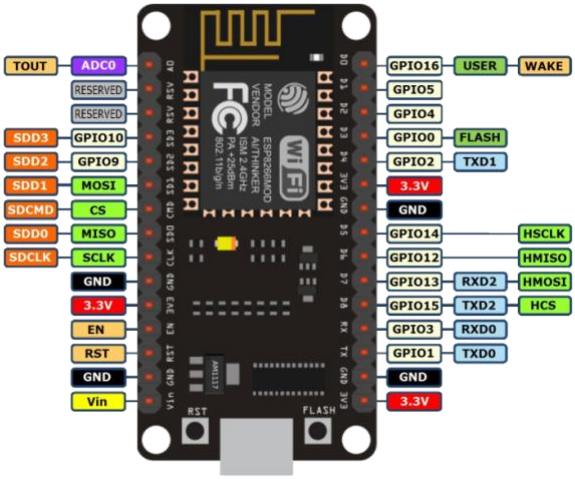
Fonte: Baú da eletrônica, 2020.

2.4 Placa NodeMCU

O módulo Wifi ESP8266 NodeMCU é uma placa de desenvolvimento que combina o chip ESP8266, uma interface usb-serial e um regulador de tensão 3.3V. A programação pode ser feita usando LUA ou a IDE do Arduino, utilizando-se a comunicação via cabo micro-usb. Esta placa foi criada em 2014 e é bem interessante, pois ao contrário

de alguns módulos desta família, que necessitam de um conversor USB serial externo para que haja troca de informações entre computador e o módulo, o NodeMCU já vem com um conversor USB serial integrado. Essa plataforma é composta basicamente por um chip controlador (ESP8266 ESP-12E), uma porta micro USB para alimentação e programação, e um conversor USB serial integrado. Ela já possui WiFi nativo.(MASTER WALKER, 2016)

Figura 5 - NodeMCU



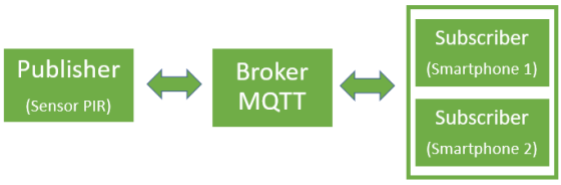
Fonte: Eletrogate, 2018.

Na figura 5 está o NodeMCU com sua pinagem indicada.

2.5 Protocolo de Comunicação MQTT

Seguindo o modelo de publicação e assinatura, o MQTT define dois tipos de entidades na rede: o broker e os clientes. O broker é um servidor que recebe todas as mensagens dos clientes (publisher) e as roteia para os clientes que tenham assinado (subscriber) o mesmo tópico pertencente à mensagem. O cliente pode ser definido como qualquer dispositivo que interage com o broker, como uma placa de desenvolvimento ou um aplicativo mobile. Um único broker pode receber interação de inúmeros clientes enquanto roteia mensagens de diversos tópicos distintos. (IBM, 2017). Exemplificado na figura 6.

Figura 6 - Esquemático da rede IoT utilizada



Fonte: Próprio Autor

Como se pode observar na Figura 6, a rede é composta por 1 agente publisher e 2 agentes (ou mais) subscribers MQTT. Cada agente da rede tem a seguinte função:

Publisher: Responsável pela publicação do valor do sensor de presença da área delimitada na residência;
Subscriber: Responsável por receber os valores que identificam a presença no smartphone que estiver conectado e inscrito.

2.7 Plataforma Cloud MQTT

A plataforma CloudMQTT disponibiliza instâncias virtuais hospedadas em centro de dados localizados em diferentes continentes. Portanto, foi definida a escolha da região “Estados Unidos” devido à sua maior proximidade física com os dispositivos e, consequentemente, por apresentar menor latência durante a comunicação.

3 METODOLOGIA DO TRABALHO

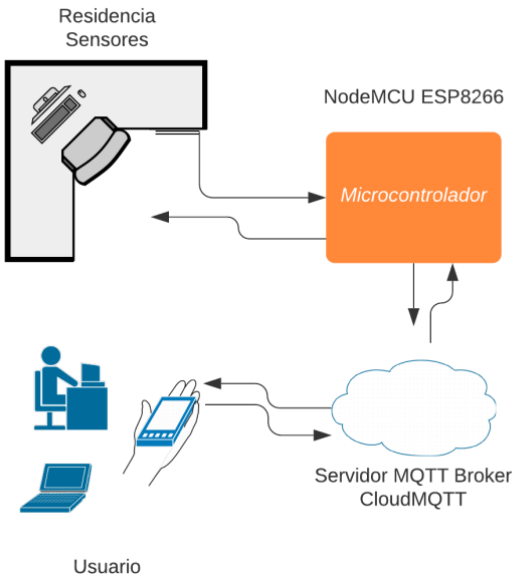
A metodologia do projeto foi feita como pesquisa aplicada, visando aplicar os conhecimentos adquiridos no decorrer do curso. Utilizando conceitos de Internet das Coisas, foram criados para a conexão com a internet - protocolo MQTT, sendo criada uma instância de MQTT, um código arduino IDE, e um aplicativo Blynk, software e circuito do

protótipo com sensores e microcontrolador, hardware, e a seguir estão as plataformas utilizadas, fazendo o papel de conexão dos dispositivos.

O desenvolvimento parte de uma base detalhada de um conjunto de hardware e software que permite monitorar um cômodo em uma casa residencial.

Na figura 7 é mostrado de forma simplificada a interação entre hardware e software. As informações dos sensores são atualizadas a cada segundo, devido à regra estabelecida. Os dados registrados pelos sensores são captados e enviados pelo o microcontrolador para um servidor MQTT. E a informação será disponibilizada no aplicativo Blynk em um smartphone ou na plataforma CloudMQTT Assim, o usuário poderá visualizar os dados registrados.

Figura 7 - Esquemático Geral Simplificado



Fonte: próprio autor

De acordo com o exposto acima, o projeto está dividido em etapas descritas a seguir:

Primeira etapa: o cômodo escolhido conta com sensores acoplados no sistema que enviará dados registrados para o microcontrolador. As informações coletadas serão tratadas devidamente nos próximos passos.

Segunda etapa: o microcontrolador receberá as informações emitidas pelos sensores, tratará esses dados e enviará para a próxima etapa. Observando que o microcontrolador também irá alimentar o sistema. Contudo, o microcontrolador tem como sua principal função controlar e gerenciar todos os outros sensores e seus respectivos processos.

Terceira etapa: no servidor MQTT os dados recebidos pela etapa anterior serão publicados e disponíveis para consulta pelo usuário devido à instância CloudMQTT criada pelo autor.

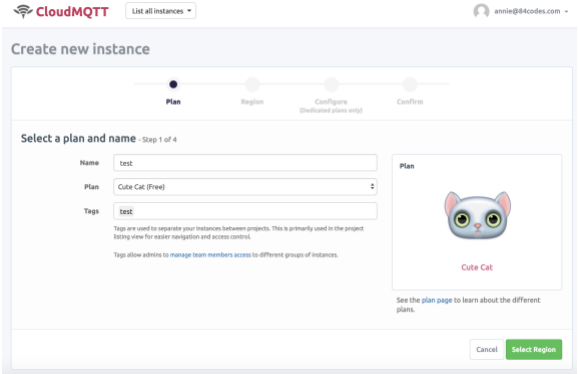
Quarta etapa: o usuário com a interação do navegador conseguirá visualizar as informações registradas e disponíveis no CloudMQTT ou no aplicativo Blynk.

3.1 Conexão com Internet – Protocolo

3.1.1 CloudMQTT

A primeira etapa para a utilização da referida plataforma se deu por meio do registro de uma conta de usuário e criação da instância virtual, sendo necessário preencher o nome da instância e selecionar o plano gratuito (free) conforme Figura 8.

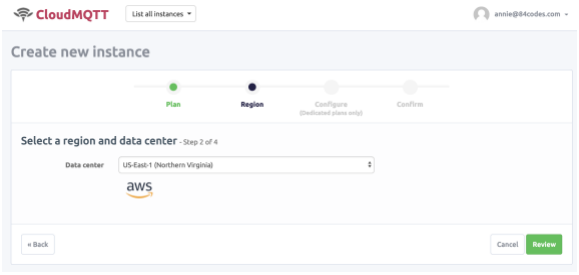
Figura 8 - Etapa de criação da instância virtual na plataforma CloudMQTT.



Fonte: CloudMQTT, 2019.

A segunda etapa consistiu na escolha da região onde a instância seria hospedada.

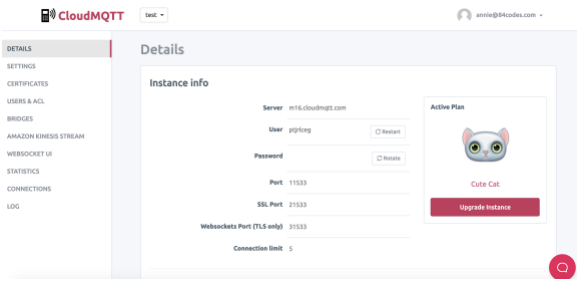
Figura 9 - Etapa de seleção da região onde a instância será hospedada no CloudMQTT.



Fonte: CloudMQTT, 2019.

Após finalização da segunda etapa, a instância foi imediatamente provisionada e seus detalhes exibidos. Ao ser criada a instância, a plataforma define automaticamente o servidor remoto, usuário, senha e porta que deverão ser utilizados para estabelecer a comunicação dos clientes com o servidor Broker conforme a figura 9.

Figura 8 - Informações da instância criada pelo CloudMQTT.



Fonte: CloudMQTT, 2019.

Para que o microcontrolador ESP8266 pudesse se conectar à internet através de uma rede wifi e publicar mensagens no Broker, fez-se necessário definir os parâmetros da rede sem fio e da instância virtual provisionada pelo CloudMQTT no código em C++ desenvolvido no software IDE Arduino. Após cada parâmetro ser declarado como uma variável constante do tipo char* (com exceção do parâmetro mqttPort que foi declarado como tipo int), definiu-se o nome da rede WiFi do roteador em que o microcontrolador estabeleceria conexão no parâmetro “ssid” e a senha da rede sem fio no parâmetro password. Em seguida, definiu-se os parâmetros do servidor Broker remoto fornecidos pela plataforma CloudMQTT: o host remoto no parâmetro mqttServer, a porta no mqttPort, o usuário no mqttUser e, por fim, a senha do Broker no parâmetro mqttPassword.

Figura 10 - Declaração das variáveis para comunicação com a rede Wifi e servidor Broker.

```
const char* ssid = "PINHONET";// wifi da minha casa
const char* password = "*******"; //Senha, colocamos a que usaremos no WIFI
const char* mqttServer = "tailor.cloudmqtt.com"; // servidor do broker
const int mqttPort = 17678; // porta
const char* mqttUser = "bvxcyhby"; // usuário criado para conexão com o broker
const char* mqttPassword = "Y3q7p47nJrMS"; // senha criada para conexão com o broker
```

Fonte: Próprio Autor.

A configuração com a rede WiFi utilizando os parâmetros anteriormente definidos foi iniciada por meio da função Wifi.begin(ssd, password), a qual retorna o status WL_CONNECTED enquanto a conexão com a rede sem fio estiver estabelecida, e WL_IDLE_STATUS enquanto a conexão não estiver estabelecida. Porém, com o módulo WiFi estando ainda ligado. A função Wifi.status() foi chamada dentro de uma condição while para que o microcontrolador exibisse o progresso da conexão com a rede WiFi no monitor serial do Arduino IDE. Ao retornar o status WL_CONNECTED, o microcontrolador exibe no monitor serial o endereço IP local recebido do DHCP do roteador através da função Wifi.localIP().

Figura 11 - Trecho do código responsável por realizar a conexão WiFi.

```
void setup_wifi() {

 delay(10);
 // Conectamos a WiFi network
 Serial.println();
 Serial.print("Connecting to ");
 Serial.println(ssid);

 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 Serial.print(".");
 }

 Serial.println("");
 Serial.println("WiFi connected");
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());
}
```

Fonte: Próprio Autor

Após estabelecida a conexão com a rede WiFi, o microcontrolador dará continuidade a sequência de instruções definidas no código. Para inicializar a comunicação com o servidor Broker, ao chamar a função client.connect("ESP8266Client",mqttUser, mqttPassword), onde "ESP8266Client" é o cliente ID do dispositivo. Este cliente será identificado no Broker. Os demais parâmetros se referem ao usuário e senha declarados nas variáveis para realizar a autenticação com o servidor. O host remoto, onde será estabelecida a conexão, é definido na função client.setServer(mqttServer, 17678), onde o parâmetro mqttServer é o host remoto tailor.cloudmqtt.com também declarado nas variáveis. E "17678" é a porta que será utilizada para estabelecer a comunicação com o Broker. Ao retornar o valor true na função client.connect(), o microcontrolador exibe uma mensagem no monitor serial do Arduino IDE, indicando que a conexão com o Broker foi estabelecida ou, caso contrário, retornará o código que representa o motivo da falha na conexão ao chamar a função client.state(). A tentativa de conexão com o host remoto irá persistir até obter êxito.

Figura 12 - Trecho do código responsável por realizar a conexão com o Broker remoto.

```
void reconnect() { // tentará reconectar ao broker, caso não tenha conseguido
 while (!client.connected()) {
    Serial.println("Connecting to MQTT...");
    WiFi.mode(WIFI_STA);
    if (client.connect("ESP8266Client", mqttUser, mqttPassword )) {

      Serial.println("connected");

    } else {

      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(5000);

    }
}
}
```

Fonte: Próprio Autor

3.1.2 Blynk

Ao considerar-se o universo maker, é quase impossível não citar os projetos baseados em Arduino. O surgimento de novos dispositivos, que também podem ser programados em Arduino, bem como a utilização de shields (placas que agregam funções aos dispositivos Arduino) ampliaram as possibilidades de projetos que podem ser desenvolvidos em Arduino. Paralelamente, o surgimento de serviços conectados à internet e o conceito de IoT (Internet Of Things) aumentaram a demanda por dispositivos que possuam conectividade e, assim, proporcionem o envio de dados à internet e o controle remoto destes dispositivos.

Para mostrar o nível de CO, o widget de variação do monóxido de carbono foi conectado ao pino A0, onde está conectado o sensor MQ-7, da placa NodeMCU. Os LCDs virtuais na plataforma utilizam pinos digitais criados no código do arduino, V1 e V2 para data e hora. E os pinos virtuais V3 e V4 para umidade e temperatura. O pequeno smartphone na tela do app, é um widget responsável por enviar as notificações de movimento ou gás detectado. O relógio é para sincronizar a hora com o local onde o

aparelho se encontra. Pode-se observar no trecho do código mostrado na figura 12, a configuração dos pinos V1 e V2 para display LCD virtual.

Figura 13 - Trecho do código responsável configurar o relógio e pinos virtuais.

```
BlynkTimer timer;

WidgetRTC rtc;

// Digital clock display of the time
void clockDisplay()
{
  // You can call hour(), minute(), ... at any time
  // Please see Time library examples for details

  String currentTime = String(hour()) + ":" + minute() + ":" + second();
  String currentDate = String(day()) + "/" + month() + "/" + year();
  Serial.print("Current time: ");
  Serial.print(currentTime);
  Serial.print(" ");
  Serial.print(currentDate);
  Serial.println();

  // Send time to the App
  Blynk.virtualWrite(V1, currentTime);
  // Send date to the App
  Blynk.virtualWrite(V2, currentDate);
}

BLYNK_CONNECTED() {
  // Synchronize time on connection
  rtc.begin();
}
```

Fonte: Próprio Autor

Figura 14 – Função responsável por medir a temperatura

```
void DHT_11(){

  Serial.print("\nTemperatura DHT11: ");
  tempa= dht.readTemperature();
  Serial.print(tempa);
  Serial.print(" *C \n");

  Serial.print("Humidade DHT11: ");
  humid = dht.readHumidity();
  Serial.print(humid);
  Serial.print(" % \n");

  Blynk.virtualWrite(V3, humid);
  Blynk.virtualWrite(V4, tempa);

}
```

Fonte: Próprio Autor

Na figura 14 pode-se ver a função que faz a medição da temperatura por pinos digitais utilizando as funções da biblioteca DHT.h, poupando assim o uso de um multiplexador para aumentar as portas analógicas, pois o NodeMCU só possui uma, que está ocupada com o sensor MQ-7 de gás.

Figura 15 - Aplicativo Blynk

Fonte: Próprio Autor

Portanto, o Blynk foi usado para controlar os pinos do NodeMCU, ou receber informações de tais pinos. Como pode ser visto na figura 15. Por fim, ao receber um sinal de estado HIGH do sensor PIR conectado a porta digital do NodeMCU, o microcontrolador armazena a mensagem "Movimento detectado" em uma variável temporária e, em seguida, a mensagem é publicada no Broker ao chamar a função client.publish("outTopic", msg). Onde "outTopic" é a identificação do tópico em que a mensagem será publicada no Broker e msg é o conteúdo (mensagem) da variável temporária. Caso o microcontrolador receba um sinal LOW do sensor PIR, a mensagem "Nenhum movimento detectado" será exibida no monitor serial do Arduino IDE. A placa recebe também um sinal do sensor de gás e fumaça MQ-7 que, por emitir sinal analógico, recebe um valor de quantidade de CO. Caso esse valor for maior que 400ppm

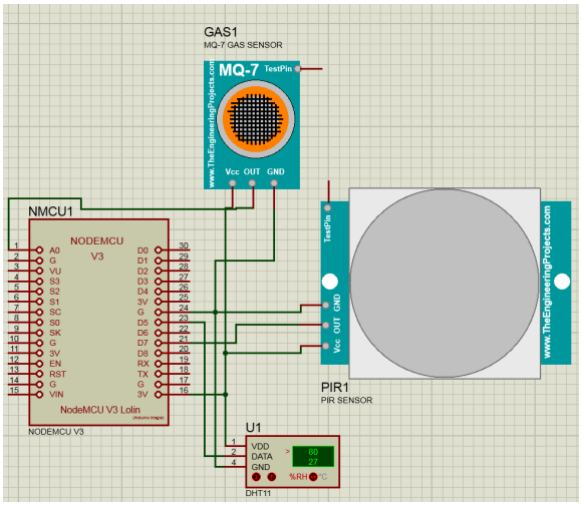
então é acionada a mensagem de gás detectado.

O protótipo foi programado para que se houver movimento e gás detectados ele apresentará a mensagem “movimento e gás detectados”. Caso sinalize presença no sensor PIR, mas não haja detecção de CO, apresentará a seguinte mensagem: “movimento detectado”. Se acontecer de somente haver uma variação alta de CO acima de 400ppm, apresentará a mensagem “gás detectado”. Finalmente, se não houver movimento ou gás detectado, não haverá mensagem para que não haja desperdício de memória, já que esta mensagem estaria sendo publicada frequentemente.

Essas mensagens serão publicadas em notificações em tempo real para que o usuário possa monitorar o ambiente desejado. Esse monitoramento pode ocorrer por meio do Broker MQTT ou pelo app criado Blynk, onde mostra notificações pelo smartphone.

Para alcançar a melhor performance do sistema proposto, uma rede baseada no modelo de arquitetura IoT é composta por 4 agentes MQTT. Ela foi constituída e um sensor de presença, gás, umidade e temperatura foram simulados. O diagrama contendo sensor e os agentes MQTT sensores foram utilizados para melhor entendimento e veracidade do cenário real. O sensor PIR publica a presença ao determinar um estado de valor HIGH. A partir dessa situação, mensagens periodicamente são enviadas para o Subscriber e com níveis de Qos= 0 e 1. Figura a seguir ilustra a rede física constituída para testes.

Figura 16 – Esquemático do circuito



Fonte: Próprio Autor

Na figura 16 é exemplificado o circuito do protótipo, para melhor compreensão do leitor.

No microcontrolador NodeMCU, foi conectado no pino digital D7 o sensor PIR, para aferir sinal HIGH ou LOW. O outro pino digital utilizado foi no sensor de temperatura DHT11 conectado no pino D5, para medição de temperatura com biblioteca DHT.h. O único pino analógico foi conectado ao sensor de gás MQ-7, com cada sensor conectado ao barramento GROUND e 3.3V.

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Como visto, o projeto tem como objetivo monitorar um cômodo e publicar esses dados para gerenciamento do usuário e identificar qualquer modificação remotamente, por meio de um dispositivo com conexão à rede, podendo ser um smartphone, tablet, notebook entre outros. Para o êxito do projeto, o usuário deverá manusear todas as funcionalidades descritas acima sem nenhuma dificuldade, e que haja veracidade e integridade nos resultados encontrados.

O sistema foi submetido por diversas etapas para verificar o seu funcionamento. A partir disso, foram coletados e registrados os para a análise.

Para a análise do funcionamento do sistema e dos dados, foram utilizados métodos específicos de testes. Esses testes serão utilizados para verificar a margem de erro, a tolerância a falhas do sistema e a veracidade dos resultados registrado pelo o mesmo.

Eles são:

O sensor de gás MQ-7 mede o nível de CO do ambiente, e mostra essas informações no aplicativo ou no servidor MQTT. O referido teste é demonstrado conforme a tabela 1.

Tabela 1. Testes de Gás, temperatura e umidade.

No quarto			
Hora	umidade (%)	Nível CO (MPa)	4 (°C)
19:47:30	51	256	25.8
19:48:32	51	257	25.7
20:02:08	53	327	25.5

Fonte: Próprio Autor (2020)

Os resultados foram obtidos a partir do protótipo citado nos parágrafos de metodologia. Sendo assim, ao medir os dados de nível de CO, a temperatura, umidade, e sinalização de presença são coletados e armazenados os resultados a seguir. Como representados na figura 15.

4.1 Resultados

O sensor de temperatura DHT11 mede temperatura e umidade do ambiente, e mostra essas informações no aplicativo ou no servidor MQTT. Assim, para testar o funcionamento do protótipo foram medidas a diferença de temperatura e de umidade do sensor, comparada à temperatura e umidade da cidade no site do Clima Tempo. O referido teste demonstrou êxito e um bom funcionamento do sistema conforme a tabela 2 e 3.

Foram realizados testes aumentando o nível de CO do local, fazendo movimento no ambiente e medindo a temperatura e umidade. Seguem os prints e tabelas com data e hora:

Tabela 2. Testes temperatura.

Data	Hora	Projeto (°C)	Clima (°C)	Erro (%)
16/06	21:30	18.2	18	1,09
16/06	22:30	16.8	15	10,71
16/06	23:30	14.5	14	3,44
17/06	13:30	24.4	22	9,83
17/06	16:30	22.6	22	2,65
17/06	19:30	18.9	18	4,76
17/06	21:10	17.9	17	5,02
17/06	23:50	16.6	16	3,61
18/06	12:15	22.1	20	9,50
18/06	14:10	27.5	26	5,45
18/06	17:30	25.4	24	5,51
18/06	22:10	17.8	17	4,49
19/06	10:20	18.5	17	8,10
19/06	13:10	21.6	23	6,48
19/06	23:20	19.2	18	6,25
20/06	01:00	17.9	17	5,02

Fonte: Próprio Autor (2020)

Nos horários mostrados, ao anoitecer a temperatura tende a diminuir, como exemplificado na tabela 2. O erro se deve ao fato de a temperatura ser medida pelo site Clima Tempo com a localização de Brasília inteira, diferentemente do sensor que está medindo em tempo real, na região administrativa Jardim Botânico.

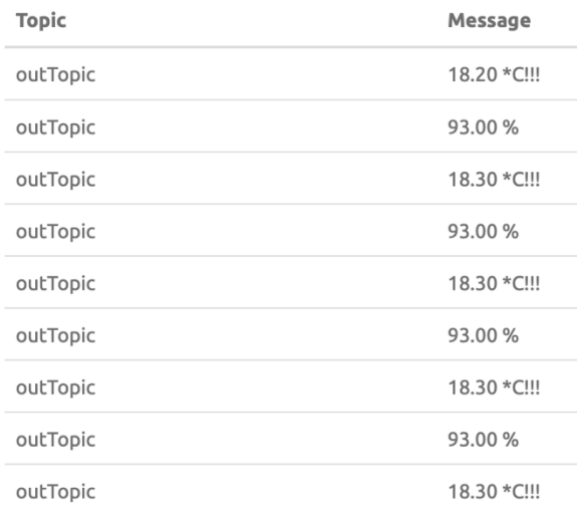
Tabela 3. Testes umidade.

Data	Hora	umidade (%)	Clima (%)	Erro (%)
16/06	21:30	81	79	2,40
16/06	22:30	92	80	13,04
16/06	23:30	93	82	11,83
17/06	13:30	64	67	4,68
17/06	16:30	71	73	2,81
17/06	19:30	93	89	4,30
17/06	21:10	92	90	2,17
17/06	23:50	92	90	2,17
18/06	12:15	80	77	3,75
18/06	14:10	52	56	7,69
18/06	17:30	62	59	4,83
18/06	22:10	92	88	4,34
19/06	10:20	93	89	4,30
19/06	13:10	85	79	7,05
19/06	23:20	92	90	2,17
20/06	01:00	93	92	1,07
12/07	14:02	44	48	9,09
12/07	15:12	48	45	6,25
12/07	16:15	49	46	6,12

Fonte: Próprio Autor (2020)

Nos horários mostrados, ao anoitecer a umidade tende a aumentar, como exemplificado na tabela 3. O erro se deve ao fato de a umidade ser medida pelo site Clima Tempo com a localização de Brasília inteira, diferentemente do sensor que está medindo em tempo real, na região administrativa Jardim Botânico.

Figura 17 – Resultado na plataforma CloudMQTT



Fonte: CloudMQTT, 2020.

A figura 17 representa os resultados na plataforma CloudMQTT mostrando a os resultados da comunicação do WiFi com o MQTT.

Como pode-se observar na figura 18, o protótipo mede a temperatura e umidade em tempo real, e verifica sinal de movimento ou presença de gás CO.

Figura 18 - Resultado Monitor Serial Arduino IDE

```
19:11:13.701 -> Temperatura DHT11: 24.30 *C
19:11:13.701 -> Humidade DHT11: 54.00 %
19:11:13.737 -> 436
19:11:13.737 -> GAS DETECTADO !!!
19:11:19.276 ->
19:11:19.276 ->
19:11:19.777 ->
19:11:19.777 -> Temperatura DHT11: 24.30 *C
19:11:19.813 -> Humidade DHT11: 54.00 %
19:11:19.851 -> 414
19:11:19.851 -> GAS DETECTADO !!!
19:11:25.357 ->
19:11:25.391 ->
19:11:25.890 ->
19:11:25.890 -> Temperatura DHT11: 24.30 *C
19:11:25.926 -> Humidade DHT11: 54.00 %
19:11:25.926 -> 402
19:11:25.926 -> GAS DETECTADO !!!
19:11:31.477 ->
19:11:31.477 ->
19:11:31.964 ->
19:11:31.964 -> Temperatura DHT11: 24.30 *C
19:11:32.001 -> Humidade DHT11: 54.00 %
19:11:32.040 -> 393
19:11:32.040 -> GAS AUSENTE !!! e Nenhum movimento detectado
19:11:37.497 ->
19:11:37.497 ->
19:11:37.991 ->
19:11:37.991 -> Temperatura DHT11: 24.30 *C
19:11:38.027 -> Humidade DHT11: 54.00 %
19:11:38.078 -> 387
19:11:38.078 -> GAS AUSENTE !!! e Nenhum movimento detectado
```

Fonte: Arduino IDE, 2020.

A figura 18 mostra o resultado dos testes citados anteriormente na plataforma Arduino IDE.

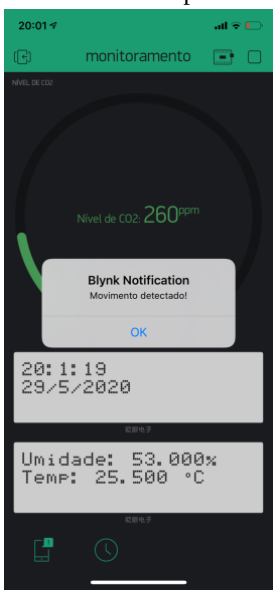
Em certos momentos há presença de gás, pois um isqueiro estava saindo gás próximo ao sensor, para podermos observar a detecção. Pode-se observar a partir das figuras 19 e 20 os mesmos resultados, porém no aplicativo Blynk, mostrando inclusive sinal de movimento e gás detectados.

4.2 Aplicativo

4.2.1 Funcionamento e visualização de resultados no aplicativo Blynk

O aplicativo foi configurado na plataforma Blynk com o intuito de monitorar o ambiente com uma interface mais intuitiva ao usuário. Para tal funcionamento, foram utilizados pinos virtuais na programação do arduino IDE.

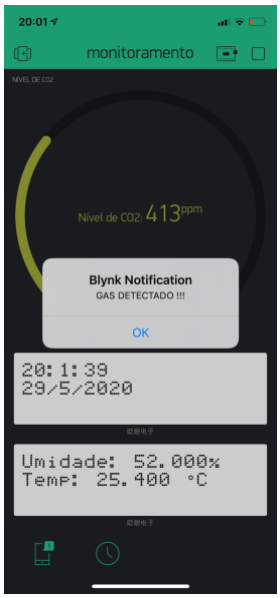
Figura 19 – Resultado aplicativo Blynk



Fonte: Blynk, 2020.

Como mostra a figura 19, quando houver qualquer movimento no ambiente o aplicativo irá notificar o usuário.

Figura 20 – Resultado do teste de gás CO no aplicativo Blynk



Fonte: Blynk, 2020.

Como mostra a figura 20, quando houver qualquer indício de Gás CO no local, o aplicativo irá notificar o usuário e fará tais notificações até mesmo com o aparelho em modo bloqueado.

5 CONSIDERAÇÕES FINAIS

O principal objetivo do projeto foi o de construir um protótipo de monitoramento de ambiente de baixo custo para trazer mais segurança às casas das pessoas. O NodeMCU se mostrou uma ótima opção, já que é capaz de funcionar como o arduino, mas possuindo um preço muito mais em conta e ainda disponibilizando acesso a redes sem fio. Segurança residencial custa muito, o que impossibilita o investimento para pessoas com menor poder aquisitivo. O projeto desenvolvido custou aproximadamente R$258,00 e mesmo se tratando de um protótipo, pode ser considerado uma alternativa. A tabela apresenta os gastos detalhados relacionados ao projeto.

Tabela 4. Tabela de Gastos com protótipo.

Material	Valor(R$)
NodeMCU	39,90
DHT11	13,00
Sensor MQ-7	34,90
Protoboard	19,90
Jumpers	19,80
PowerBank	130,00
Total	257,50

Fonte: Próprio Autor (2020)

Em relação à precisão dos dados, com os testes realizados, obteve-se ao calcular uma média dos erros com um valor médio de mais ou menos 5,03% para a umidade, e de aproximadamente 5,74% para temperatura. O que no geral pode ser considerado aceitável. Os testes com gás e fumaça ocorreram a partir da sensibilidade do sensor,

observando o tempo de resposta ao estimular o sensor com um isqueiro. Como apresentado anteriormente, esses testes foram realizados na RA Jardim Botânico, porém os dados de comparação foram do Clima Tempo da cidade de Brasília-DF. Dessa forma, pode ter ocasionado a diferença entre os dados obtidos. Sendo assim, o projeto apresentou no geral um bom desempenho.

Neste trabalho foram estudados os controladores, foi cogitado o uso da placa Wemos D1 R2, ou somente o módulo ESP8266, porém com melhores e simples resultados foi o NodeMCU. Houve dificuldades para formulação do código e aplicativo, porém ao fim de muito estudo e dedicação foi possível o término deste trabalho.

Para uma complementação e sequência posterior do projeto, caberia adicionar uma câmera de vigilância. Para, caso haja um movimento detectado ele automaticamente abra o aplicativo mostrando o local, e assim com mais dados agregar mais funções para o monitoramento do ambiente.

AGRADECIMENTOS

Agradeço a Deus por minha vida, meus pais e professores. Agradeço ao meu orientador ter conduzido esse trabalho. Agradeço a esta banca pelo seu tempo, a amigos que me ajudaram em tempos difíceis quando imaginei não conseguir. Agradeço acima de tudo a mim por ter persistido e finalizado esse trabalho de pesquisa.

REFERÊNCIAS

AVED, Adeel. **Criando Projetos com Arduino para a Internet das Coisas**: experimentos com aplicações do mundo real. São Paulo: Novatec, 2017. 275p.

Agência Brasil - **Internet das coisas: saiba como essa tecnologia pode afetar sua vida**. Disponível em: <https://agenciabrasil.ebc.com.br/geral/noticia/2019-09/internet-das-coisas-saiba-como-essa-tecnologia-pode-afetar-sua-vida > Acesso em: 15 nov. 2019

Arduinoecia - **Como usar um sensor PIR**. Disponível em: <https://www.arduinoecia.com.br/sensor-presenca-arduino-modulo-pir-dyp-me003/> Acesso em: 15 nov. 2019

Baú da eletrônica. Disponível em: <https://www.baudaeletronica.com.br> Acesso em: 28 mai. 2020

Blynk - Blynk. Disponível em: < https://blynk.io/en/getting-started/> Acesso em: 18 abr. 2020

CloudMQTT - **Websocket UI**. Disponível em: < https://api.cloudmqtt.com/console/82643868/websocket/> Acesso em: 20 jun. 2020

DE MELLO, Danilo Augusto. **Solução para monitoramento ambiente utilizando arduino**, Guarapuava, 2016.

Eletrogate. Disponível em: <https://www.eletrogate.com/?gclid=EAIaIQobChMIqZaK56La6QIVhBCRCh0ugAC7EAAYASAAEgK71_D_BwE> Acesso em: 28 mai. 2020

Flipeflop - **Sensor de Movimento Presença PIR** . Disponível em: <https://www.filipeflop.com/produto/sensor-de-movimento-presenca-pir/> Acesso em: 15 nov. 2019

IBM. **Conhecendo o MQTT**. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html> Acesso em: 17 nov. 2019

Microcontrolades-c. Disponível em: <https://microcontroladores-c.blogspot.com/2018/02/nodemcu-mqtt-android-iot.html> Acesso em: 28 mai. 2020

Mundo mais tech - **Internet das Coisas: conheça 4 casos de uso da tecnologia na agricultura**. Disponível em: <https://mundomaistech.com.br/iot/internet-das-coisas-conheca-4-casos-de-uso-da-tecnologia-na-agricultura/
Acesso em: 20 abril. 2020

NodeMCU – **Uma plataforma com características singulares para seu projeto IoT** - MasterWalker. Disponível em < https://blogmasterwalkershop.com.br/embarcados/nodemcu/nodemcu-uma-plataforma-com-caracteristicas-singulares-para-o-seu-projeto-iot/> Acesso em: 28 maio. 2020

Estadão - **Sistema de alarme para sua casa.** Disponível em <http://patrocinados.estadao.com.br/verisure/saiba-como-escolher-o-melhor-sistema-de-alarme-para-a-sua-casa/> Acesso em: 15 nov. 2019

TUNG, Liam, ZDnet - **IoT devices will outnumber the world's population this year for the first time**. Disponível em: <https://www.zdnet.com/article/iot-devices-will-outnumber-the-worlds-population-this-year-for-the-first-time/ > Acesso em: 05 abril. 2020

Vida de Sílicio. Disponível em: <https://www.vidadesilicio.com.br> Acesso em: 28 mai. 2020.

ZAMBARDA, Pedro. **'Internet das Coisas': entenda o conceito e o que muda com a tecnologia**. 2014. Disponível em: https://www.techtudo.com.br/noticias/noticia/2014/08/internet-das-coisas-entenda-o-conceito-e-o-que-muda-com-tecnologia.html. Acesso em: 08 abril. 2020