PLATAFORMAS IOT APLICADAS À AUTOMAÇÃO RESIDENCIAL

IOT PLATFORMS APPLIED TO RESIDENTIAL AUTOMATION

João Henrique Feitoza Fernandes[1]
José Jonas Dantas Freire Filho[2]
Vanderlei Luiz Daneluz Miranda[3]

RESUMO

Considerando os avanços tecnológicos, há diversos dispositivos hoje em dia no mercado que atribuem características diversas ao ambiente doméstico, desde dispositivos controlados por voz, até um software capaz de tomar decisões para melhoria da residência e conforto ao usuário. Pensando nisso, este projeto tem como objetivo aplicar o conceito de Internet das Coisas, e as respectivas plataformas, a fim de viabilizar soluções de Automação Residencial. Com isso, espera-se ainda, esclarecer tópicos relacionados à compatibilidade e integração dos sistemas que compõem uma solução de Automação Residencial. A metodologia utilizada compreendeu os seguintes métodos de pesquisa: na primeira fase, utilizou-se a revisão bibliográfica com o propósito de analisar na literatura o estado da arte da IoT na automação residencial; no segundo momento, optou-se por implementar uma prova de conceito (protótipo), utilizando-se hardware de baixo custo; os resultados obtidos indicaram que o a implementação foi um sucesso, atendendo aos requisitos básicos a primeiro momento; pode-se concluir que é viável utilizar hardware de baixo custo na automação residencial.

Palavras-chave: Internet das Coisas, Automação Residencial, Hardware de Baixo Custo.

[1] Graduado em Engenharia Elétrica no Centro Universitário UNIFAFIBE de Bebedouro, SP. E-mail: joaohenriquefeitoza@gmail.com
[2] Graduado em Engenharia Elétrica no Centro Universitário UNIFAFIBE de Bebedouro, SP. E-mail: jonasfreif@gmail.com
[3] Professora Mestre no Centro Universitário UNIFAFIBE de Bebedouro, SP. E-mail: vandmir@gmail.com

ABSTRACT

Having identified technological advances, several devices on the market today that attribute different characteristics to the home environment, from voice-controlled devices to software capable of making decisions to improve the home and user comfort. With that in mind, this project aims to apply the concept of the Internet of Things, and as platforms, in order to enable Home Automation solutions. With that, it is also expected to clarify related to the compatibility and integration of the systems that make up a Home Automation solution.

The methodology used included the following research methods: in the first phase, a literature review is used with the purpose of analyzing the state of the art of IoT in home automation in the literature; in the second moment, it was decided to implement a proof of concept, using low-cost hardware; the results obtained indicated that the implementation was a success, meeting the basic requirements at first; one can conclude that it is feasible to use low-cost hardware for home automation.

Keywords: Internet of Things, Home Automation, Low Cost Hardware.

1 INTRODUÇÃO

Nos dias atuais, a visão do mundo moderno tem evoluído consideravelmente devido à influência da tecnologia que tem alcançado freneticamente as mais diversas áreas da vida do ser humano, despertando nele a necessidade de controlar tudo no seu entorno. As necessidades de controlar e automatizar as rotinas e tarefas em residências deram início a um novo domínio de aplicação tecnológico designado por Domótica ou Automação Residencial (LUSEMBO, 2017).

Com essa porta aberta, a Domótica junto a Internet das Coisas (IoT) vem nos ajudar nas atividades rotineiras, ou até mesmo, no controle dos aparelhos eletrônicos de nossas casas, podendo diminuir o consumo de energia, ou agir em determinadas situações para melhorar a segurança.

Para que isso ocorra é necessária uma plataforma IoT, capaz de se comunicar com dispositivos interligados à Web, como no caso o microcontrolador ESP8266, que possui uma alta variedade de módulos, e o melhor, ter um custo acessível.

As aplicações para dispositivos presentes nas residências são enormes. Porém, deve-se levar em consideração o desejo do usuário e o quanto está disposto

a investir. Pensando nesses fatores, esse projeto irá focar em aplicações básicas como o controle da iluminação, climatização, persianas, portão eletrônico e sensores de alarme. Adicionalmente tem como objetivo compreender o que são as plataformas de IoT, compreender como a sua abordagem pode viabilizar soluções de Automação Residencial e implementar um protótipo do estudo realizado utilizando hardware de baixo custo.

2 REFERENCIAL TEÓRICO

A seguir são apresentados os conceitos essenciais sobre IoT e suas aplicações, elementos básicos da Automação Residencial e os Microcontroladores Arduino e ESP8266.

2.1 Internet das Coisas (IoT)

A IoT trata-se de um ecossistema que conecta objetos físicos do dia a dia à Internet, incluindo objetos domésticos comuns, como lâmpadas, dispositivos médicos e acessórios, dispositivos smart e até mesmo cidades inteligentes. são dispositivos que necessita da internet para funcionar apropriadamente, assim como outros equipamentos de grande porte como servidores de grandes empresas.

No geral, são vários dispositivos que podem ser conectados à internet. A ideia principal por trás da IoT é a de facilitar a vida dos usuários e clientes, tornando o uso de certos elementos mais simples e até permitindo a automação de tarefas. Por exemplo:

a. Aspirador de pó: robô pode ser programado para limpar a casa depois da hora de dormir;

b. Lâmpadas: podem emitir luzes em tons específicos durante vários momentos do dia, ou se apagarem quando todos saírem;

c. Ar-condicionado: pode se ativar cinco minutos antes de você chegar, deixando o ambiente na temperatura correta;

d. Fogão: seria capaz de cortar o fornecimento de gás e avisar a companhia fornecedora ao detectar um vazamento;

e. Freezers comerciais: avisaram o fabricante em caso de defeito, evitando a perda de comida, vacinas ou outros elementos perecíveis ou pouco duráveis sem refrigeração;

f. Tratores automatizados: seriam capazes de fazer o trabalho de um funcionário mesmo à noite, com dados via satélite para evitar desperdício e utilizando a rede apenas quando necessário;

g. Hospitais: podem utilizar equipamentos capazes de coletar dados armazenados em smartwatches, pulseiras inteligentes e outros dispositivos vestíveis que monitorem os dados vitais do paciente, otimizando o atendimento e facilitando o diagnóstico (GOGONI, 2019).

2.2 Automação Residencial

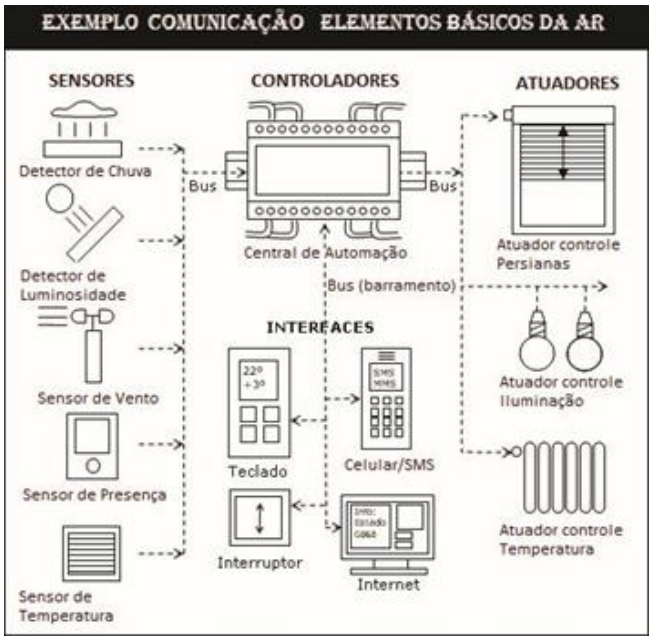
Segundo Bolzani (2004), “a automação residencial pode ser definida como um conjunto de tecnologias que ajudam na gestão e execução de tarefas domésticas cotidianas. A sua utilização tem por objetivo proporcionar um maior nível de conforto, comodidade e segurança além de um menor e mais racional consumo de energia”.

Para (WORTMEYER; FREITAS; CARDOSO, 2005) “automação residencial representa o emprego de tecnologias ao ambiente doméstico (incluindo residências, condomínios, hotéis), com o objetivo de propiciar conforto, praticidade, produtividade, economia, eficiência e rentabilidade, com valorização da imagem do empreendimento e de seus usuários”.

Elementos Básicos da Automação Residencial:

Por trás da automação residencial existem diversos elementos envolvidos, de simples sensores até complexas centrais de automação, que fornecem uma experiência ideal para as necessidades, desejos e condições de cada usuário. Nesta seção são apresentados os elementos básicos da Automação Residencial. Dificilmente se encontrará uma Residência Inteligente sem algum dos elementos: Controladores, Sensores, Atuadores, Barramentos e Interfaces, descritos a seguir.

FIGURA 1. Exemplo da comunicação dos elementos básicos na Automação Residencial



Fonte: Casadomo (2010)

A Figura 1 apresenta um exemplo de como os elementos básicos se comunicam. A esquerda dessa figura estão os sensores, que encaminham as informações sobre algum evento (chuva, vento etc.) aos controladores (ao centro) e estes por sua vez acionam os ativadores (à direita), de acordo com a tarefa programada para aquele evento, como por exemplo abrir a persiana. As interfaces (interruptores, celular etc.) se conectam diretamente aos controladores de forma a permitir que o usuário visualize as informações e interaja com o sistema de automação. Diversos barramentos podem ser utilizados na comunicação entre os elementos básicos (rede elétrica, telefônica etc.).

Os Controladores controlam os dispositivos automatizados (sensores e atuadores). Monitora as informações dos sensores, podendo enviar comandos para que um atuador ative ou desative algum equipamento. De maneira geral podem possuir interfaces independentes, na forma de um controle remoto, ou serem sofisticadas centrais de automação (ALMEIDA, 2009; ACCARDI; DOVOROV, 2012).

Os Sensores são os dispositivos que detectam estímulos, medem e monitoram grandezas físicas e eventos (temperatura, umidades etc.), convertendo-as em um

valor passível de manipulação por sistemas computacionais. São eles que encaminham as informações aos controladores sobre algum evento, para que os controladores possam enviar os comandos adequados para os atuadores (ALMEIDA, 2009; ACCARDI; DOVOROV, 2012).

Os Atuadores são dispositivos eletromecânicos, que recebem os comandos do sistema de automação e ativam os equipamentos automatizados. São os módulos de acionamento ligados entre a rede elétrica e os equipamentos (ALMEIDA, 2009; ACCARDI; DOVOROV, 2012). Existem atuadores para portas, janelas, persianas, fechadura magnética, sirene, indicadores luminosos, etc.

O Barramento é o meio físico responsável pelo transporte das informações (rede elétrica, telefônica etc.) (CASADOMO, 2010; ACCARDI; DOVOROV, 2012).

As Interfaces são os dispositivos ou mecanismos (navegador de internet, celular, painéis, controles remotos, interruptores etc.) que permitem ao usuário visualizar as informações e interagir com o sistema de automação (CASADOMO, 2010; ACCARDI; DOVOROV, 2012).

2.3 Nuvem - *Sinric Pro*

A nuvem contempla dois serviços, o servidor da Amazon que recebe a mensagem da Alexa e executa o comando e o Sinric Pro um sistema de controle de dispositivos smarts.

O *Sinric Pro* foi desenvolvido por um usuário da *Alexa* e Google Home, apesar de não ter o código aberto, seu uso é gratuito. Sua vantagem é fazer todo o tratamento da API da Amazon para o dispositivo a ser controlado e aceitar os comandos já desenvolvidos pela Amazon para dispositivos smart. Mesmo que os comandos utilizados pela Amazon sejam traduzidos para outras línguas a API não é alterada, não sendo necessário alteração no código para incluir as novas línguas. No momento somente nos idiomas alemão, francês e japonês possuem alguns comandos configurados. No momento que o usuário envia um comando de voz “Alexa, desligue o dispositivo televisivo” detecta o comando e direciona para a aplicação relevante, neste caso o Sinric Pro para completar a tarefa.

Para o servidor conseguir reconhecer o uso dessa aplicação, a mesma deve ser ativada na Alexa por meio da skill, nome dado às funções extras da Alexa (funcionam como aplicativos no caso dos celulares), e vinculada à conta do Sinric Pro

pessoal do usuário. Cada conta no Sinric Pro possui uma chave de acesso da API única a qual permite que o sistema encontre os dispositivos vinculados à conta do usuário. O servidor envia para o Sinric Pro uma API com a identificação do dispositivo, a função a ser executada e o valor caso tenha. E o Sinric Pro repassa essas informações para o dispositivo, caso consiga ou transmita a mensagem envia um feedback para o servidor da Amazon (LYRA, 2018).

2.4 Arduino

O Arduino foi criado em 2005 por um grupo de 5 pesquisadores: Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. O objetivo era elaborar um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de programar, sendo dessa forma acessível a estudantes e projetistas amadores. Além disso, foi adotado o conceito de hardware livre, o que significa que qualquer um pode montar, modificar, melhorar e personalizar o Arduino, partindo do mesmo hardware básico.

Assim, foi criada uma placa composta por um microcontrolador Atmel, circuitos de entrada/saída e que pode ser facilmente conectada à um computador e programada via IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado) utilizando uma linguagem baseada em C/C++, sem a necessidade de equipamentos extras além de um cabo USB.

Depois de programado, o microcontrolador pode ser usado de forma independente, ou seja, você pode colocá-lo para controlar um robô, uma lixeira, um ventilador, as luzes da sua casa, a temperatura do ar condicionado, pode utilizá-lo como um aparelho de medição ou qualquer outro projeto que vier à cabeça (THOMSEN, 2014).

2.5 Microcontrolador ESP8266

O Microcontrolador ESP8266 (SCHWARTZ, 2016) trata-se de um dispositivo voltado a IoT, que consiste de um microprocessador ARM 32 bits, que possui Wi-Fi integrado. O mesmo pode ser utilizado junto ao Arduino como módulo para atribuir conexões sem fio ao mesmo, e ter mais possibilidades de melhoria no projeto.

Também há diversas versões do ESP8266, sendo uma delas o ESP-01 (PRADO, 2019), que possui um tamanho reduzido, com menos portas de conexão, mas com um alto potencial e custo baixo.

O ESP8266 utiliza a IDE Arduino para sua programação, sendo assim, a programação entre o Arduino e o ESP8266, é praticamente igual baseada em C/C++, tornando mais fácil a programação do mesmo.

TABELA 1. Comparação entre ESP8266, ESP32 e Arduino UNO R3.

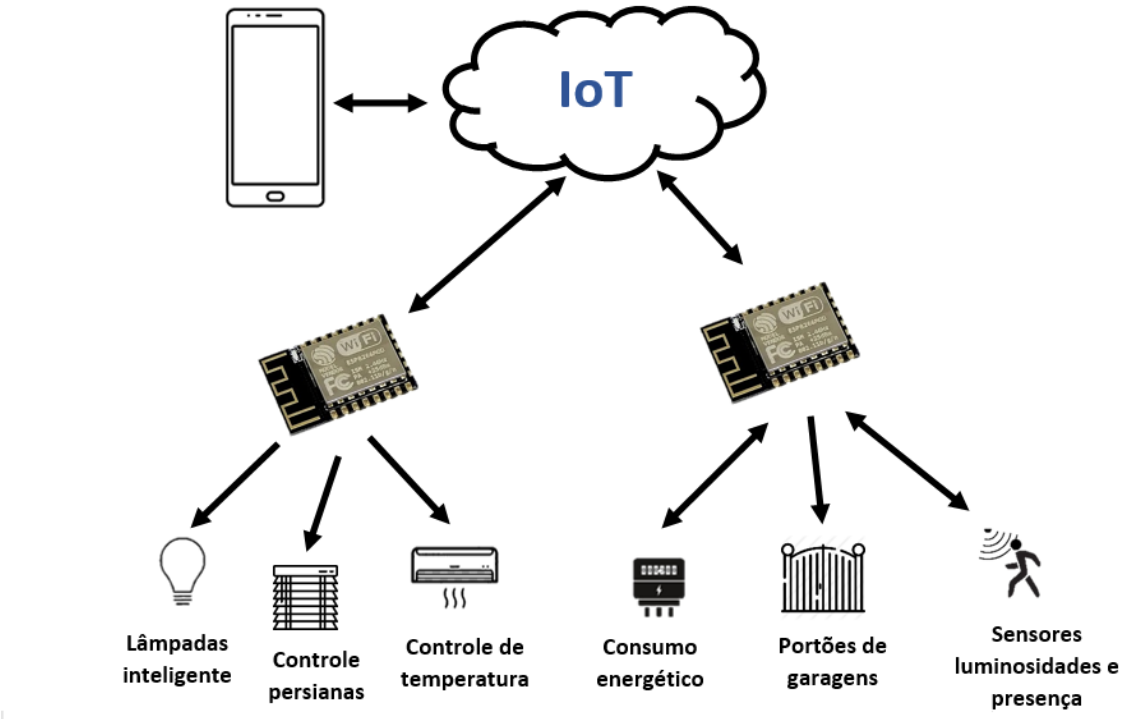
	ESP8266	**ESP32**	**Arduino Uno R3**
Corrente	197 mA	220 mA	40 mA
Núcleo	1	2	1
Arquitetura	32 bits	32 bits	8 bits
Clock	8 - 160Mhz	160 - 240 Mhz	16 Mhz
Wi-Fi	SIM	SIM	NÃO
Bluetooth	NÃO	SIM Clássico e BLE	NÃO
RAM	160KB	520KB	2KB
FLASH	16Mb	16Mb	32KB
GPIO	11	22	12
DAC	0	2	0
ADC	1	18	6
Interfaces	SPI - 12C - UART - 12S	SPI - 12C - UART - 12S - CAN	SPI - 12C - UART

Fonte: FVM Learning

3 METODOLOGIA (Materiais e Métodos??)

A fim de validar a pesquisa bibliográfica deste trabalho, foi feita a construção de um protótipo. Na figura 2, apresentamos um diagrama do funcionamento do ambiente da IoT proposto no projeto.

FIGURA 2. Diagrama do Projeto



Fonte: Elaborada pelos autores (2021)

3.1 Microcontroladores

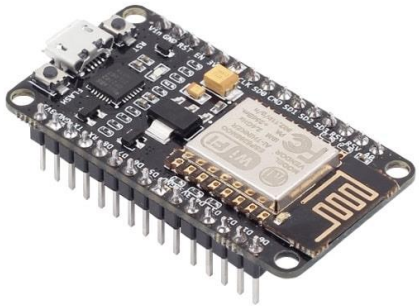
3.1.1 ESP8266

O microcontrolador é o coração do projeto, pois é nele que será ligado todos os sensores e atuadores, é nele que toda lógica irá ser implementada. Primeiramente foi utilizado um modelo mais simples para a implementação do primeiro projeto, porém deve-se escolher bem o modelo, para que não haja a falta de portas, ou memória.

O modelo utilizado no primeiro protótipo foi uma versão menor do ESP8266, o ESP-01, a fim de levantar informações de consumo e compatibilidade com o sistema ao todo.

Para o projeto final, simulando a automação de uma casa em uma maquete, foi escolhido o microcontrolador ESP8266 com o módulo NodeMCU, por possuir diversos pinos para conexão, capaz de nos entregar uma automação de até 8 dispositivos em sua configuração mais simples.

FIGURA 3. Módulo NodeMCU



Fonte: FILIPEFLOP (2021)

3.2 Sensores

A seguir apresentamos sensores de possível implementação a um projeto de automação residencial.

3.2.1 Sensor LDR

Para obter informações sobre a iluminação externa, foi utilizado um sensor LDR (Light Dependent Resistor) que é um componente cuja resistência varia de acordo com a intensidade da luz. Quanto mais luz incidir sobre o componente, menor a resistência.

FIGURA 4. Sensor LDR

Fonte: FILIPEFLOP (2021)

3.2.2 Sensor de Movimento Presença PIR

O Sensor de Movimento PIR DYP-ME003 consegue detectar o movimento de objetos que estejam em uma área de até 7 metros.

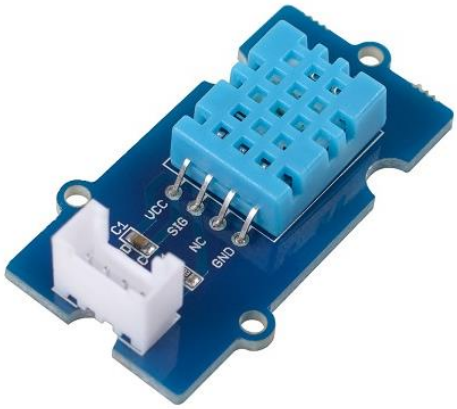
FIGURA 5. Sensor de Movimento Presença PIR

Fonte: FILIPEFLOP (2021)

3.2.3 Sensor de Temperatura e Umidade

O Sensor de Temperatura e Umidade Grove DHT11 é um módulo que contém um sensor DHT11 na placa, sendo interligado ao microcontrolador por meio de um cabo de 4 pinos padrão Grove.

FIGURA 6. Sensor de Temperatura e Umidade



Fonte: FILIPEFLOP (2021)

3.3 Atuadores

3.3.1 Iluminação LEDs

Para simular as lâmpadas, serão utilizados LEDs, o LED (Light Emitting Diodo, ou Diodo Emissor de Luz) é formado por um material semicondutor que emite luz quando uma tensão é aplicada nos terminais.

FIGURA 7. Sensor LDR

Fonte: FILIPEFLOP (2021)

3.3.2 Relé

Utilizado para o acionamento de dispositivos interligados à rede elétrica, o relé age como um interruptor a partir do sinal recebido pela controladora.

FIGURA 8. Relé

Fonte: FILIPEFLOP (2021)

3.4 Plataformas

3.4.1 Arduino IDE

A principal plataforma foi o Arduino IDE que é uma plataforma que cruza a linguagem de programação C e C++, nela usamos para escrever e fazer upload de códigos em placas compatíveis com Arduino (SHANKO, 2019).

3.4.2 Sinric Pro

Para integração entre os dispositivos e as assistentes virtuais, foi utilizada a plataforma Sinric Pro, ela recebe informações do servidor da Amazon ou Google que recebe a mensagem da Alexa ou Google Home e executa o comando e o nos dispositivos smarts.

Foi utilizado essa plataforma pois não há necessidade de dispositivos físicos como no caso o Echo Dot, apenas da assistente pessoal no smartphone.

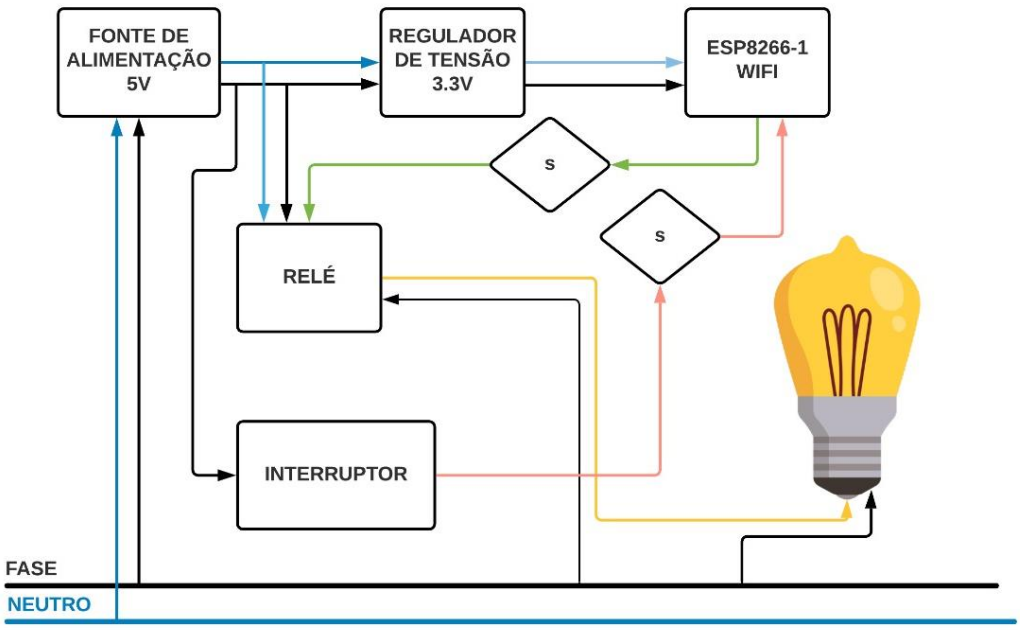
3.5 Implementação

3.5.1 Implementação do Protótipo 1

Inicialmente, foi implementado um primeiro protótipo, em uma lâmpada, a fim de verificar a estabilidade e compatibilidade do equipamento com a rede Wi-Fi e nuvem. Através do software Arduino IDE versão 1.8.16 foi desenvolvido um script que compõem a conexão com Wi-Fi e nuvem onde serão atribuídos os dispositivos (lâmpadas, sensores, etc.). O intuito é cadastrar os dispositivos na nuvem e controlá-los remotamente ou atribuir rotinas para melhor conforto.

Para melhor interpretação do protótipo foi desenvolvido um diagrama simples, composto por 4 itens e ambiente a qual foi instalado:

FIGURA 9. Diagrama automação de uma lâmpada



Fonte: Elaborado pelos autores (2021

A. ESP8266-1: Microcontrolador, sendo o coração do projeto, onde é atribuído o código para conexão com Wi-Fi e nuvem;
B. Fonte 5V: foi utilizado um carregador de celular para garantir a tensão de 5V para o circuito;

C. Regulador de tensão 1117: utilizado para baixar a tensão para 3.3V vinda da fonte 5V, pois o ESP-01 utiliza 3.3V em sua entrada;
D. Relé: utilizado como interruptor, tendo 3 entradas: 5V, GND e Entrada de dados; e 3 saídas para acoplamento a lâmpada.

O sistema funciona da seguinte forma: é alimentado o circuito com a tensão de fase, onde ele irá rebaixar essa tensão para 5V através do “carregador de celular”, essa tensão irá ser utilizada para alimentar o relé diretamente e para alimentar o ESP-01, passará primeiro por um regulador de tensão fornecendo 3.3V. O ESP-01, conectado com a rede Wi-Fi, através da plataforma Sinric Pro, podemos criar até 3 dispositivos de forma gratuita, podemos controlar esse interruptor por assistentes virtuais, disponíveis no smartphone. Com todas as configurações feitas, obtemos o pino de saída do ESP, que será utilizado para mandar um sinal booleano, para o relé acionando o mesmo ou desligando.

Para acionar a lâmpada de forma manual utilizamos um interruptor ligado a uma porta de entrada no ESP, sendo assim podendo ligar tanto via internet quanto manualmente.

FIGURA 10. Protótipo 1

Fonte: Elaborado pelos autores (2021)

3.5.2 Implementação na Maquete

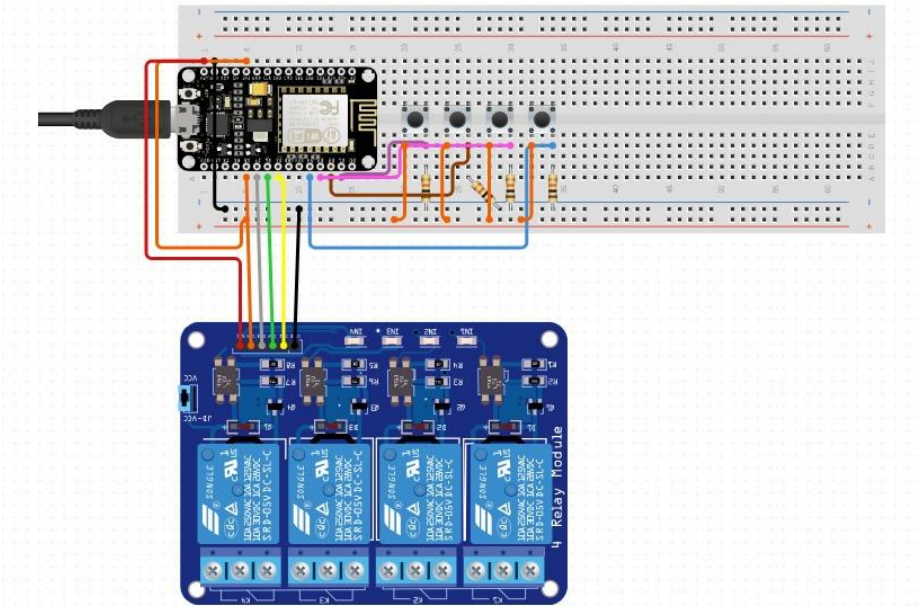
O teste final foi realizado utilizando uma maquete a fim de simular em um ambiente residencial. Para a implementação foi automatizada a luz do quarto, da sala e o controle do portão.

FIGURA 11. Maquete

Fonte: Elaborado pelos autores (2021)

O Circuito possui uma configuração simples, conforme esquemático da figura 12, onde há somente dois dispositivos principais, que são eles Microcontrolador ESP8266 e os Relés. Saindo desses é ligado os dispositivos citados na maquete acima, ou pra a implementação real, em dispositivos presentes na residência a ser automatizada.

FIGURA 12. Circuito da maquete



Fonte: Elaborado pelos autores (2021)

3.6 Testes de Validação

Com base no código desenvolvido para implementação desse projeto, dividimos em duas partes principais para implementação do projeto, sendo elas a comunicação com o Wi-Fi e comunicação com a plataforma Sinric Pro.

3.6.1 Comunicação com Wi-Fi

No código abaixo vemos as bibliotecas e passos necessários para se realizar a conexão do ESP8266 com o Wi-Fi, ele nos permite configurar a rede se a necessidade de modificar o código para cada Wi-Fi diferente, o mesmo já armazena na memória as redes já conectadas, ou seja, não é necessário configurar novamente quando a rede ficar inativa.

Código:

```
//Bibliotecas Necessárias
#include <ESP8266Wi-Fi.h> //Inclusão do microcontrolador ESP8266
#include <DNSServer.h> //// Servidor DNS local usado para redirecionar
todas as solicitações para o portal de configuração
#include <ESP8266WebServer.h> // WebServer local usado para servir o portal
de configuração
```

```
#include <Wi-FiManager.h> // Gerenciador de conexões Wi-Fi

void setup() {

    Serial.begin(115200); //chamando monitor serial
    Wi-FiManager; //Objeto de manipulação do Wi-Fi
    Wi-FiManager.autoConnect("AutoConnectAP"); //Função para se auto
conectar na rede
    Serial.println("connected...yeey :)"); //Mensagem após conectado
}

void loop() {
}
```

Após o upload do programa na placa, a mesma irá gerar um Wi-Fi com a denominação “AutoConnectAP” que pode ser configurada tanto pelo computador ou Smartphone, sendo possível escolher a rede de preferência.

FIGURA 13. Configuração do Wi-Fi

Fonte: Elaborado pelos autores (2021)

3.6.2 Comunicação com Sinric Pro

No código abaixo vemos as bibliotecas e passos necessários para se realizar a conexão do ESP8266 com o a plataforma Sinric Pro, Ele é responsável por se comunicar com a plataforma, a partir dos dados cadastrados nela, e com isso receber os sinais enviados para o controle dos dispositivos.

Código:

```
//Inicio
#ifdef ENABLE_DEBUG
       #define DEBUG_ESP_PORT Serial
       #define NODEBUG_WEBSOCKETS
       #define NDEBUG
#endif

#include <Arduino.h>
#include <ESP8266Wi-Fi.h>
#include "Sinric ProPro.h"
#include "Sinric ProProSwitch.h"

#include <map>

//Wi-Fi Offiline
#define WI-FI_SSID         "********"
#define WI-FI_PASS         "********"

#define APP_KEY  "de0bxxxx-1x3x-4x3x-ax2x-5dabxxxxxxxx" //Chave do APP
Sinric Pro
#define APP_SECRET "5f36xxxx-x3x7-4x3x-xexe-e86724a9xxxx-4c4axxxx-3x3x-
x5xe-x9x3-333d65xxxxxx"////Senha do APP Sinric Pro

//Insira os IDs do dispositivo presente no Sinric Pro aqui
#define device_ID_1   "SWITCH_ID_NO_1_HERE"
#define device_ID_2   "SWITCH_ID_NO_2_HERE"
#define device_ID_3   "SWITCH_ID_NO_3_HERE"
#define device_ID_4   "SWITCH_ID_NO_4_HERE"

// Definir o GPIO conectado com relés e interruptores
#define RelayPin1 5  //D1 // Lampada quarto
#define RelayPin2 4  //D2 // Lampada sala
#define RelayPin3 14 //D5 // Portão
#define RelayPin4 12 //D6 // Reserva

#define SwitchPin1 10  //SD3 // Botão lâmpada quarto
#define SwitchPin2 0   //D3  // Botão lâmpada sala
#define SwitchPin3 13  //D7  // Botão portão
#define SwitchPin4 3   //RX  // Reserva

#define Wi-FiLed   16   //D0

// comente a seguinte linha se você usar interruptores em vez de botões
táteis
#define TACTILE_BUTTON 1

#define BAUD_RATE   9600
```

```
#define DEBOUNCE_TIME 250

typedef struct {      // struct for the std::map below
  int relayPIN;
  int flipSwitchPIN;
} deviceConfig_t;

// esta é a configuração principal
// insira seu deviceId, o PIN para Relay e o PIN para flipSwitch
// isso pode ser até N dispositivos ... dependendo de quantos pinos está
disponível em seu dispositivo;)
// agora temos 4 devicesIds indo para 4 relés e 4 interruptores flip para
alternar o relé manualmente
std::map<String, deviceConfig_t> devices = {
    //{deviceId, {relayPIN,  flipSwitchPIN}}
    {device_ID_1, {  RelayPin1, SwitchPin1 }},
    {device_ID_2, {  RelayPin2, SwitchPin2 }},
    {device_ID_3, {  RelayPin3, SwitchPin3 }},
    {device_ID_4, {  RelayPin4, SwitchPin4 }}
};

typedef struct {  // struct for the std::map below
  String deviceId;
  bool lastFlipSwitchState;
  unsigned long lastFlipSwitchChange;
} flipSwitchConfig_t;

std::map<int, flipSwitchConfig_t> flipSwitches;  // este mapa é usado para
mapear PINs flipSwitch para deviceId e lidar com verificações de estado de
debounce e última flipSwitch, será configurado na função
"setupFlipSwitches", usando informações do mapa de dispositivos
void setupRelays() {
  for (auto &device : devices) {// para cada dispositivo (relé, combinação
flipSwitch)
    int relayPIN = device.second.relayPIN; // obtendo o pino de
retransmissão
    pinMode(relayPIN, OUTPUT);  // define o pino do relé para OUTPUT
  }
}

void setupFlipSwitches() {
  for (auto &device : devices)  { // para cada dispositivo (combinação relé
/ flipSwitch)
    flipSwitchConfig_t flipSwitchConfig; // cria-se uma nova configuração
flipSwitch

    flipSwitchConfig.deviceId = device.first; // definir o deviceId
    flipSwitchConfig.lastFlipSwitchChange = 0; // definir o tempo de
debounce
    flipSwitchConfig.lastFlipSwitchState = false; // definir
lastFlipSwitchState para false (LOW)

    int flipSwitchPIN = device.second.flipSwitchPIN; // obtenha o
flipSwitchPIN

    flipSwitches[flipSwitchPIN] = flipSwitchConfig; // salve a configuração
flipSwitch no mapa flipSwitches
    pinMode(flipSwitchPIN, INPUT_PULLUP); // defina o pino flipSwitch para
INPUT
  }
```

```
}

bool onPowerState(String deviceId, bool &state)
{
  Serial.printf("%s: %s\r\n", deviceId.c_str(), state ? "on" : "off");
  int relayPIN = devices[deviceId].relayPIN; // obtenha o pino de
retransmissão para o dispositivo correspondente
  digitalWrite(relayPIN, !state);    // define o novo estado de relé
  return true;
}

void handleFlipSwitches() {
  unsigned long actualMillis = millis(); // obter milis reais
  for (auto &flipSwitch : flipSwitches) { // para cada flipSwitch no mapa
flipSwitches
 unsigned long lastFlipSwitchChange =
flipSwitch.second.lastFlipSwitchChange;// obtém o carimbo de data / hora
quando flipSwitch foi pressionado da última vez (usado para eliminar /
limitar eventos)

    if (actualMillis - lastFlipSwitchChange > DEBOUNCE_TIME) {// se o tempo
for> tempo de debounce ...

      int flipSwitchPIN = flipSwitch.first; // obtenha o pino flipSwitch da
configuração
      bool lastFlipSwitchState = flipSwitch.second.lastFlipSwitchState;
// obtenha o lastFlipSwitchState
      bool flipSwitchState = digitalRead(flipSwitchPIN); // lê o estado
flipSwitch atual
      if (flipSwitchState != lastFlipSwitchState) { // se flipSwitchState
mudou ...
#ifdef TACTILE_BUTTON
        if (flipSwitchState) { // se o botão tátil for pressionado
#endif
          flipSwitch.second.lastFlipSwitchChange = actualMillis;// atualiza
a hora do lastFlipSwitchChange
          String deviceId = flipSwitch.second.deviceId;// obtenha o
deviceId da configuração
          int relayPIN = devices[deviceId].relayPIN; // obtém o relayPIN da
configuração
          bool newRelayState = !digitalRead(relayPIN); // define o novo
estado do relé
          digitalWrite(relayPIN, newRelayState); // define o trelay para o
novo estado

          Sinric ProProSwitch &mySwitch = Sinric ProPro[deviceId];// obter
dispositivo de switch de Sinric ProPro
          mySwitch.sendPowerStateEvent(!newRelayState);
       // envie o evento
#ifdef TACTILE_BUTTON
        }
#endif
        flipSwitch.second.lastFlipSwitchState =
flipSwitchState;                  // atualizar lastFlipSwitchState
      }
    }
  }
}

void setupWi-Fi()
{
```

```
void setupSinric ProPro()
{
  for (auto &device : devices)
  {
    const char *deviceId = device.first.c_str();
    Sinric ProProSwitch &mySwitch = Sinric ProPro[deviceId];
    mySwitch.onPowerState(onPowerState);
  }

  Sinric ProPro.begin(APP_KEY, APP_SECRET);
  Sinric ProPro.restoreDeviceStates(true);
}

void setup()
{
   Serial.begin(BAUD_RATE);

  pinMode(Wi-FiLed, OUTPUT);
  digitalWrite(Wi-FiLed, HIGH);

  setupRelays();
  setupFlipSwitches();
  setupWi-Fi();
  setupSinric ProPro();
}

void loop()
{
  Sinric ProPro.handle();
  handleFlipSwitches();
}
```

Nas figuras 14 e 15, vemos as credenciais e códigos dos dispositivos utilizados para configuração do código, esses dados são disponibilizados pela plataforma Sinric Pro, e são únicas para cada usuario ou dispositivo.

FIGURA 14. Credenciais

Fonte: Elaborada pelos autores (2021)

FIGURA 15. Dispositivos

Fonte: Elaborada pelos autores (2021)

5 RESULTADOS

O resultado deste trabalho foi a implantação de um protótipo para validação e logo após a construção de uma maquete representando em escala reduzida o funcionamento da automação em uma residência, onde foi possível testar e verificar a implementação de como seria em um ambiente real, com o controle da iluminação e do portão via botões virtuais ou comandos de voz.

Foi também possível verificar o consumo de energia de cada equipamento, a partir do acionamento do mesmo, e com uma função bastante interessante nas plataformas Google Home e Alexa, foi possível atribuir rotinas a cada equipamento, definindo os horários de ligamento e desligamento, por motivos de segurança ou por falhas humanas.

TABELA 2. Discriminação do consumo de energia em 24h – Luz do Quarto

LIGADO	DESLIGADO	WATT HORAS	DURAÇÃO (HORAS)
18/11/2021 05:00	18/11/2021 05:20	3,02	0.336
18/11/2021 05:32	18/11/2021 05:59	3,97	0.441
17/11/2021 19:31	17/11/2021 21:08	14,62	1.624
17/11/2021 23:29	17/11/2021 23:32	0,43	0.048
17/11/2021 05:00	17/11/2021 05:10	1,58	0.176
17/11/2021 05:00	17/11/2021 05:20	3,04	0.338
17/11/2021 05:40	17/11/2021 05:58	2,71	0.301
17/11/2021 19:24	17/11/2021 19:29	0,74	0.082
TOTAL		**30,11**	**1624**

Fonte: Sinric Pro

CONCLUSÃO

Conclui-se que para esta pesquisa e implementação do projeto sobre plataformas IoT, que todos os objetivos foram atingidos, utilizado todo conhecimento da pesquisa e aplicando-o na pratica. Porém, faltou a implementação dos sensores neste projeto, sendo somente necessário realizar algumas alterações no código e na plataforma Sinric Pro.

REFERÊNCIAS

ACCARDI; DOVOROV, A.; E. Automação Residencial: Elementos Básicos, Arquiteturas, Setores, Aplicações e Protocolos. Revista TIS, São Carlos, v. 1, n. 2, p. 156-166, 2012. ISSN 2316-2872.

FERNANDES, Glauco Fagundes. Automação residencial: utilizando internet das coisas e ESP-8266. 2018.

GOGONI, R. O que é Internet das Coisas? Tecnoblog. Apresenta textos e notícias voltados à tecnologia. Disponível em: <https://tecnoblog.net/263907/o-que-e-internet-das-coisas/>. Acesso em: 09 jun. 2021.

GUERRA, Filipe Henrique Moreira. Automação residencial de baixo custo com protocolo X10 e ESP8266. 2016.

LUSEMBO, PLAMEDI L. PROTÓTIPO DE SISTEMA DE AUTOMAÇÃO RESIDENCIAL INTEGRADO COM RASPBERRY PI UTILIZANDO WINDOWS.

LYRA, Gustavo Correa de. Controle de dispositivos infravermelhos usando Amazon Echo. 2018.

OLIVEIRA, G. NodeMCU – Uma plataforma com características singulares para o seu projeto ioT, Tecnoblog. Apresenta textos e notícias voltados à tecnologia. Disponível em: <https://blogmasterwalkershop.com.br/>. Acesso em: 10 nov. 2021.

SUBHAJIT – Smart Home with Google Assistant & Alexa using NodeMCU ESP8266, Tecnoblog. Apresenta projetos voltados à tecnologia. Disponível em: < https://iotcircuithub.com/smart-home-with-google-assistant-alexa/> Acesso em:26 out. 2021.

SUHANKO, D. – A “linguagem do Arduino” é C ou C++?, Tecnoblog. Apresenta textos e artigos voltados à tecnologia. Disponível em: < https://www.dobitaobyte.com.br/a-linguagem-do-arduino-e-c-ou-c/>. Acesso em: 30 out. 2021.

SCHWARTZ, Marco. Internet of Things with ESP8266. Packt Publishing Ltd, 2016.
PRADO, Paterson Anunciação. Sistema de monitoramento remoto de consumo de energia elétrica em residências. 2019

THOMSEN, A. O que é Arduino?. FilipeFlop. Apresenta conteúdos educacionais, voltados a tecnologia engajada no movimento maker. Disponível em: <https://www.filipeflop.com/blog/o-que-e-arduino/>. Acesso em: 23 jun. 2021.

WORTMEYER, Charles; FREITAS, Fernando; CARDOSO, Líuam. Automação Residencial: Busca de Tecnologias visando o Conforto, a Economia, a Praticidade e a Segurança do Usuário. II Simpósio de Excelência em Gestão e Tecnologia SEGeT2005, 2005.