Trabalho de Conclusão de Curso

Métodos de integração entre ESP8266 e Assistente virtual Alexa

de **Leonardo Alexandre Ferreira da Silva**

orientado por
Prof. Dr. João Raphael Souza Martins

Universidade Federal de Alagoas
Instituto de Computação
Maceió, Alagoas
01 de junho de 2023

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

MÉTODOS DE INTEGRAÇÃO ENTRE ESP8266 E ASSISTENTE VIRTUAL ALEXA

Trabalho de Conclusão de Curso submetido ao Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do grau de Engenheiro de Computação.

Leonardo Alexandre Ferreira da Silva

Orientador: Prof. Dr. João Raphael Souza Martins

Banca Avaliadora:

Glauber Rodrigues Leite Prof. Me, UFAL
Tiago Alves de Almeida Prof. Dr, UFAL

Maceió, Alagoas
01 de junho de 2023

.

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecária: Helena Cristina Pimentel do Vale CRB4 - 661

S586m Silva, Leonardo Alexandre Ferreira da.
Metodos de integração ao entre ESP8266 e assistente virtual Alexa / Leonardo Alexandre Ferreira da Silva. – 2023.
48 f. : il.

Orientador: João Raphael Souza Martins .
Monografia (Trabalho de Conclusão de Curso em Engenharia de Computação) – Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2023.

Bibliografia: f. 43-44.
Inclui apêndices.

1. Automação residencial. 2. ESP8266 (Microcontrolador). 3. Alexa (Assistente virtual) . 4. Inteligência artificial. I. Título.

CDU: 004.89:681.3.068

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

MÉTODOS DE INTEGRAÇÃO ENTRE ESP8266 E ASSISTENTE VIRTUAL ALEXA

Trabalho de Conclusão de Curso submetido ao Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do grau de Engenheiro de Computação.

Aprovado em 01 de junho de 2023:

gov.br
Documento assinado digitalmente
JOAO RAPHAEL SOUZA MARTINS
Data: 21/06/2023 10:53:15-0300
Verifique em https://validar.iti.gov.br

João Raphael Souza Martins,
Prof. Dr., Orientador

gov.br
Documento assinado digitalmente
GLAUBER RODRIGUES LEITE
Data: 21/06/2023 17:35:30-0300
Verifique em https://validar.iti.gov.br

Glauber Rodrigues Leite,
Prof. Me, UFAL

gov.br
Documento assinado digitalmente
TIAGO ALVES DE ALMEIDA
Data: 21/06/2023 12:35:07-0300
Verifique em https://validar.iti.gov.br

Tiago Alves de Almeida,
Prof. Dr, UFAL

Dedicatória

Dedico este trabalho a minha família que com apoio incondicional, me muniu com incentivo e coragem para a conclusão de mais esta etapa.

Leonardo Alexandre Ferreira da Silva

Agradecimentos

Agradeço primeiramente a Deus, por ter me dado condições para concluir mais essa etapa da minha vida.

Minha esposa Silmara Alexandre por todo apoio e meu filho Leônidas Alexandre por me motivar nesses últimos momentos com seu nascimento.

Meus pais que me educaram para ser o homem que sou hoje, que trabalharam para que eu pudesse estudar.

Aos meus colegas de turma, que fizeram com que esses anos na Universidade Federal de Alagoas, UFAL, fossem menos longos e mais alegres. Sem eles o trajeto seria muito mais difícil.

A UFAL e o Instituto de Computação, IC, por me abrir as portas, possibilitando um aprendizado de qualidade, com professores capacitados para o ensino.

Por fim, agradeço ao meu orientador, Dr.João Raphael Souza Martins, por se disponibilizar e me auxiliar na construção deste trabalho.

01 de junho de 2023, Maceió, Alagoas *Leonardo Alexandre Ferreira da Silva*

A sabedoria é a coisa principal; adquire pois a sabedoria, emprega tudo o que possuis na aquisição de entendimento. Exalta-a, e ela te exaltará; e, abraçando-a tu, ela te honrará

Provérbios 4:7,8

Resumo

O avanço da automação no âmbito residencial tem proporcionado um crescimento em novas tecnologias e estudos acerca da popularização e acessibilidade do tema. A plataforma NodeMCU ESP8266 vem ganhando espaço no cenário de prototipagem e projetos, se mostrando uma excelente concorrente da já conhecida plataforma Arduino.

Neste trabalho, é proposto um estudo de caso entre dois métodos de integração para ESP8266 e a assistente virtual da Amazon, Alexa, utilizando um dispositivo Echo Dot como *hub* de comunicação. Estes elementos estarão conectados à internet via WIFI, possibilitando assim um controle de uma carga conectada ao ESP8266 através de um comando de voz.

Para o estudo de caso são abordadas as aplicações Blynk e Voiceflow, no primeiro método de integração e a biblioteca espalexa no segundo. Como cargas a serem acionadas são utilizados dois leds que representarão dispositivos de iluminação e climatização no ambiente doméstico. Um sensor de temperatura é utilizado para controle e acionamento secundário de uma das cargas.

Tendo como principal objetivo o estudo de possibilidades de acionamento por comando de voz de uma carga qualquer conectada ao ESP8266, visando um contexto de acessibilidade, uma vez que, o controle por comando de voz oferece maior comodidade e facilidade a pessoas com condições impeditivas de locomoção ou visuais, por exemplo. Por fim é apresentado um comparativo entre os métodos aplicados.

***Palavras-chave*: *Automação Residencial*; *ESP8266*; *Alexa*.**

Abstract

The advancement of automation in the residential field has provided a growth of new technologies and studies on the popularization and accessibility of the subject. The NodeMCU ESP8266 platform has been gaining space in the prototyping and project scenario, proving to be an excellent competitor to the already known Arduino platform.

This article proposes a case study between two methods of integrating the ESP8266 and Amazon's virtual assistant, Alexa, using an Echo Dot device as a communication hub. These elements are connected to the Internet via WIFI, allowing control of a load connected to the ESP8266 by voice command. by voice command.

The case study uses the Blynk and Voiceflow applications in the first integration method and the espalexa library in the second. The loads to be controlled are two LEDs representing lighting and air conditioning devices for the home environment. A temperature sensor is used for control and secondary activation of one of the loads.

The main goal is to study the possibilities of voice command activation of any load connected to the ESP8266, aiming at an accessibility context, since voice command offers greater convenience and ease for people with mobility or visual impairments, for example.

keywords*: *Home Automation*; *ESP8266*; *Alexa.

Lista de Figuras

Sumário

Capítulo 1

Introdução

A tecnologia tem alcançado um espaço cada vez maior no cotidiano da sociedade moderna, a sua presença na vida do ser humano representa, mais que facilidades, um fator cultural.

O ser humano sempre procurou automatizar e aprimorar seus processos, desde os primórdios da humanidade, embora essas tentativas de aprimoramento não sejam vistas como automação, a invenção da roda, das armas e das ferramentas agrícolas mostram como a evolução social, aparentemente, acontece em paralelo com a tecnologia. Com a automação não ocorre diferente, começando na indústria por volta do século XVIII, na Inglaterra, a automação ganha espaço fora dela e alcança variados setores da sociedade.

A palavra Domótica vem de "Domus" que significa casa, fazendo fusão com a palavra "Robótica", no sentido de realizações de ações automáticas. Tem como objetivo controlar e automatizar a função dos equipamentos elétricos e eletrônicos de um lar, interligando o sistema central, podendo ser controlado de forma local ou remota.

Serão abordados dois métodos de integração entre o ESP8266 e a Assistente virtual Alexa, afim de automatizar uma residência, ou parte dela. No primeito método serão alvos do estudo deste trabalho as aplicações Blynk e Voiceflow e no segundo a utilização da biblioteca espalexa.

Este trabalho tem a finalidade aplicar conhecimentos adquiridos ao longo do curso de Engenharia de computação, da Universidade Federal de Alagoas, em um estudo de caso sobre métodos utilizados, a fim de demonstrar como já é possível alcançar certo nível de automação residencial sem altos custos e com bastante facilidade, como pode-se ver a seguir.

- Precificação
  - Echo dot 4º geração, R$ 400.
  - ESP 8266 Nodemcu, R$ 30.
  - Blynk, versão *free*.
  - Voiceflow, versão *free*.

1.1 Motivação

A grande parte dos usuários nem imagina que já possuem vários aparelhos computacionais em suas residências. Televisores, geladeiras, climatizadores e aparelhos de som possuem, em suas versões mais modernas, microcontroladores embutidos em seus circuitos, realizando inúmeras funções, embora a maioria desses dispositivos possuam inteligência limitada e pouca ou nenhuma interação com o ambiente em que estão inseridas. Por isso, várias tecnologias estão surgindo com o intuito de integrar dispositivos domésticos e desenvolver aplicações que solucionem problemas existentes.

Fica fácil assumir que em um futuro próximo o que ainda é restrito a uma classe mais abastada, passará a ser difundido em mais residências, tendo em vista isso, mais tecnologias precisam ser estudadas a fim de popularizar a automação residencial. Com o grande número de produtos que estão sendo desenvolvidos para este fim, um problema que surge é o da compatibilidade entre os fabricantes, além disso, existe a questão da segurança dos dados, conectividade e custo."A tendência é que, no momento seguinte, a segurança, economia, conforto e praticidade proporcionados passem para o primeiro plano na valorização desta tecnologia."(WORTMEYER,2005,p.1066).

Como outras tecnologias no passado, a automação residencial ainda é percebida como algo caro, de difícil acesso onde os que possuem algo do tipo, possuem mais por status do que propriamente pelo conforto e praticidade que pode ser alcançado.

Existem várias vantagens no uso da domótica, algumas delas como a comodidade e o conforto, uma vez que ações simples como apagar uma lâmpada, controlar a temperatura, podem ser programadas pelo usuário; eficiência no consumo elétrico, uma vez que é possível definir momentos exatos onde um dispositivo estará em funcionamento; aumento na segurança, uma vez que pode-se substituir fechaduras convencionais por fechaduras elétricas que podem ser acionadas à distância, ou sistemas de câmera com sensores que podem se comunicar com outros dispositivos. Essas aplicações estão dentre as várias possibilidades que a automação residencial proporciona.

Nesse contexto, o presente trabalho busca explorar possibilidades de utilizar dispositivos de baixo custo e amplamente disponíveis, como o microcontrolador ESP8266 e o dispositivo Echo Dot, para desenvolver um sistema de automação residencial por comando de voz. Uma vez que, o controle por comando de voz oferece maior comodidade e acessibilidade a pessoas com dificuldades de locomoção ou visuais, por exemplo.

1.2 Objetivo geral

O objetivo deste trabalho é desenvolver um estudo de caso para métodos de integração entre o ESP 8266 NodeMCU e a assistente virtual Alexa, abordando uma aplicação para automação residencial, que seja capaz de com um comando de voz acionar cargas, gerando

assim uma demonstração do potencial do ESP no contexto da domótica.

O meio de comunicação do ESP8266 é via WIFI, logo esse projeto visa fazer com que o usuário se comunique com o ESP8266 via comando de voz por meio da assistente Alexa através de um dispositivo Echo Dot, criando um protótipo de automação residencial, com controle de iluminação e climatização através de comando de voz.

Dois métodos de integração serão abordados neste trabalho, e assim um estudo de caso comparativo entre os métodos será realizado. A integração será realizada entre ESP e Alexa através da aplicação Blynk e Voiceflow no primeiro método e a biblioteca espalexa no segundo. Por fim, será realizada uma análise dos resultados, apontando os pontos mais relevantes de cada método.

1.3 Objetivos específicos

Os seguintes pontos serão abordados ao longo deste trabalho, como objetivos específicos, com intuito de alcançar os objetivo geral:

Método 1

- Implementar a comunicação do ESP8266 com o aplicativo Blynk.
- Estabelecer a comunicação da assistente Alexa com o aplicativo Blynk através de uma *skill* desenvolvida com o voiceflow.
- Comunicação da Alexa com o ESP8266.

Método 2

- Implementar a comunicação do ESP8266 com a Alexa através da biblioteca ESPALEXA.

Comparação entre os métodos

- Vantagens e desvantagens
- Análise dos resultados.

Capítulo 2

Revisão bibliográfica

2.1 Automação residencial

Domótica deriva do latim, Domus, significando casa, e da palavra robótica, no sentido de executar uma ação de forma automática. De acordo com Sgarbi (2007, p. 48) pode-se imaginar que uma residência inteligente é algo como uma casa interativa, dinâmica, ou seja, os sistemas de Domótica devem interagir com os habitantes da residência.

Um ambiente inteligente é então aquele que consegue otimizar algumas funções do ambiente residencial, funções operacionais e administrativas. A residência passa a ter uma autonomia podendo conter unidades de processamento, sensores e atuadores. É possível então que se "controle as luzes, realize-se o agendamento de tarefas, assista-se a programas de vídeo em qualquer cômodo, visualize-se as crianças brincando enquanto se assiste à TV ou navegue-se pela Internet"(WORTMEYER; FREITA; CARDOSO, 2005, p. 1066), garantindo assim mais conforto, segurança.

A domótica é a área que trabalha com a ideia de controlar e automatizar equipamentos elétricos, eletrônicos, inteligentes, computadorizados ou não, de uma residência. Todavia, o custo para se manter um sistema automatizado em uma residência ainda é um desafio para a propagação da automação residencial. "A exemplo de outras tecnologias como a telefonia celular ou o aparelho de DVD, que quando foram colocados no mercado eram considerados produtos para classes mais favorecidas" (WORTMEYER; FREITA; CARDOSO, 2005, p. 1066). Com automação residencial não é diferente, todavia novas tecnologias estão derrubando esse conceito com o rápido barateamento e a popularização dos equipamentos. Produtos nacionais estão sendo desenvolvidos e estão se destacando no cenário.

2.2 ESP8266

É praticamente impossível falar em automação sem mencionar a plataforma arduino. O Arduino é uma plataforma de hardware livre, projetado em uma placa com entradas e saídas, que é utilizada em projetos e protótipos na automação, possui um microcontrolador Atmel, programável, com uma linguagem de programação baseada em C/C++.

Principalmente em automação residencial, domótica, o arduino já se tornou um meio familiar para se alcançar resultados de variados projetos. Segundo McRoberts (2011), acredita-se que mais de 500 mil placas de desenvolvimento arduino, originais e cópias, tenham sido vendidas desde o início do projeto em 2005.

Agora podemos dizer que o Arduino possui um concorrente à altura, o ESP8266 NodeMCU. Este, que será um dos objetos de estudo deste trabalho, é uma plataforma de desenvolvimento que ganha cada vez mais espaço no meio dos desenvolvedores e hobbistas.

O ESP8266 é um chip microcontrolador que chegou ao mercado no ano de 2014, desenvolvido pela empresa chinesa Espressif Systems. "Os módulos baseados no microcontrolador ESP8266 representam um grande avanço na relação de preço-recursos e podem ser um componente muito interessante para soluções de IoT."(OLIVEIRA, 2017, p. 8). Possui um circuito totalmente integrado com interfaces I/O digitais e analógicas, interface WIFI integrada (que é um grande diferencial em relação ao chip do Arduino), processador de 32 bit, capaz de atuar a 160Hz.

O ESP8266 NodeMCU foi escolhido para este trabalho por seu baixo custo e sua facilidade de desenvolvimento, que pode ser realizado no ambiente de desenvolvimento do Arduino. Praticamente todos os módulos que são utilizados na plataforma Arduino podem ser utilizados na plataforma do ESP8266. Existem diversos módulos, ou versões, que podem ser encontradas comercialmente:

- ESP 01: Versão mais simples e compacta, sendo por esse segundo aspecto a mais conhecida e utilizada.

- ESP 05: É basicamente um módulo WIFI sem portas para trabalhar, porém com um conector de antena externa para ampliar consideravelmente o alcance do sinal, sendo uma boa solução para trabalhar com o Arduino.

- ESP 07: É um módulo bastante compacto sem pinos de conexão, com uma antena cerâmica interna e um conector externo para ampliação de sinal, bastante interessante para uso em projetos compactos de automação residencial.

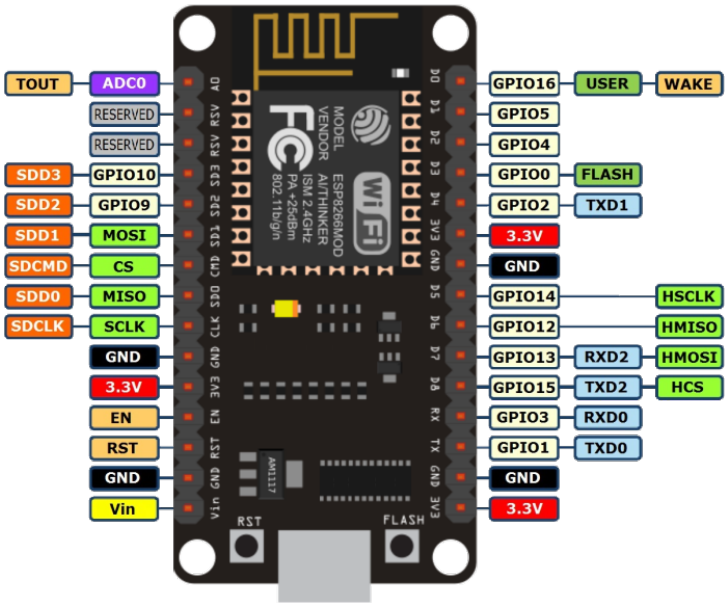
- ESP 12E: Semelhante ao ESP 07 porém sem conector para antena externa, possui 11 pinos GPIO e é utilizado para elaboração de outros módulos como NodeMCU.

- NodeMCU ESP 12E: É uma plataforma de desenvolvimento completa, semelhante ao Arduino, podendo ser conectado diretamente em um protoboard. Possui 11

pinos GPIO, conector micro USB para programação e alimentaçã., podendo ser programado facilmente pela IDE Arduino.

Neste trabalho será utilizado NodeMCU ESP12E, cujo esquemático é apresentado na figura 2.1, pois além das vantagens já descritas, pode-se citar as seguintes características:

Figura 2.1: Esquema de pinos da Node MCU ESP8266



Fonte: Holdentechnology(2018)

- Processador ESP8266-12E
- Arquitetura RISC de 32 bits
- Processador pode operar em 80 MHz / 160 MHz
- 4 Mb de memória flash
- 64 Kb para instruções
- 96 Kb para dados
- WIFI nativo padrão 802.11b/g/n
- Opera em modo AP, Station ou AP + Station
- Alimentação - 5VDC através do conector micro USB
- Possui 11 pinos digitais
- Possui 1 pino analógico com resolução de 10 bits

- Pinos digitais, exceto o D0 possuem interrupção, PWM, I2C e one wire
- Pinos operam em nível lógico de 3.3V
- Pinos não tolerantes a 5V
- Possui conversor USB Serial integrado
- Programável via USB ou WIFI (OTA)
- Compatível com a IDE do Arduino
- Compatível com módulos e sensores utilizados no Arduino

2.3 Echo Dot e Alexa

De acordo com a thinkwithgoogle (2019), mais de 60% de pessoas no Brasil já fizeram uso de comandos de voz em seus dispositivos para acessar algo, ou alguma funcionalidade, seja no trânsito, trabalho ou em casa, as assistentes virtuais estão presentes em smartphones, TVs ou nos famosos *smart speakers*.

No mercado existem várias assistentes virtuais, como a Siri, assistente da Apple; a Cortana, assistente da Microsoft; Google assistente da Google; Alexa da Amazon. Existem ainda outras assistentes que não são tão conhecidas e utilizadas, pode-se dizer que o mercado tem se expandido e cada vez mais essas tecnologias estão sendo incorporadas na vida e no cotidiano das pessoas.

O Echo Dot, figura 2.2, é o dispositivo inteligente da Amazon, com a assistente virtual Alexa, dispositivo este que tem como principal função auxiliar em tarefas simples, desde tocar uma música até acionar uma lâmpada.

A Alexa por sua vez, é assistente virtual da Amazon. Essa inteligência artificial funciona em diversos idiomas, dentre eles o português, e possui a capacidade de executar rotinas e aprender novas habilidades (*skills*), sendo possível integrá-la a outras plataformas como Uber, Spotify, Deezer, e várias outras. Segundo a agência Newvoice (2022) a Alexa tem a maior popularidade entre os assistentes de voz em vários países.

Iremos utilizar Echo Dot em sua 4° geração, pois além de possuir a inteligência artificial Alexa, com suas *skills* e funcionalidades, possui uma boa comunicação, boa qualidade de hardware e baixo custo. Funcionará como um um hub que conectará o NodeMCU com a Alexa através da rede WIFI. Com isso é importante definir:

- Dispositivos - São componentes externos ao hardware em que a Alexa está instalada, ou seja, são componentes inteligentes que podem se comunicar com a Alexa, através da rede WIFI, podendo ser uma lâmpada, um ar condicionado, uma câmera, uma cafeteira etc.

Figura 2.2: Echo Dot 4a geração

Fonte: tecnoeasyinformatica (2021)

- *Skills* - São habilidades que permitem a Alexa executar ações específicas, seja tocar uma música, agendar uma reunião dentre outras. As *skills* da Alexa permitem que ela se comunique com dispositivos e plataformas, e que aprenda novas respostas e ações. Novas *skills* são criadas a todo momento e as funcionalidades dos dispositivos Echo estão cada vez mais amplas.

- Rotinas - Rotinas são uma forma da Alexa executar uma série de habilidades em cascata, ou seja, com um simples "Alexa, Boa noite" pode-se configurar a assistente para responder, apagar as luzes da casa e ouvir música suave, tudo isso com apenas uma pequena saudação.

2.4 Blynk

O Blynk é uma plataforma IoT completa para conectar dispositivos programáveis à nuvem, com ele é possível prototipar, implantar e gerenciar remotamente dispositivos eletrônicos diversos, servindo como uma solução para projetos pessoais até empresarial, é composto por uma página WEB, um aplicativo *mobile*, um servidor online e suas bibliotecas.

O aplicativo está disponível para Android e iOS, e através do aplicativo é possível projetar aplicações para controlar dispositivo de hardware através de *widgets* que implementam funções como botões e chaves que por sua vez armazenam seus estados no servidor online, esses dados serão acessados pelo dispositivo para executar determinada função.

O servidor Blynk é um servidor online responsável por armazenar dados brutos ou processados, dependendo do plano assinado, de sensores ou atuadores conectados à placa MCU, na nuvem.

A comunicação entre o dispositivo, que neste caso será NodeMCU ESP12E, e o apli-

cativo é feito através das bibliotecas, “Há bibliotecas para todas as plataformas mais populares e compatíveis com o Blynk, permitindo a comunicação com o servidor na nuvem (cloud) ou local, processando todos os comandos de entrada e saída.”(MASTERWALKER SHOP,2017).

Em resumo o Blynk é uma plataforma IoT para realizar a conexão de hardware à nuvem, projetar aplicações para controlá-los e fazer gerenciamento dos produtos implantados em escala. A seguir tem-se algumas vantagens e desvantagens da solução:

- Vantagens
  - Suporte a mais de 400 placas, incluindo NodeMcu com ESP 8266 e ESP 32 e Arduino(Vários modelos);
  - Com os aplicativos Blynk para aplicativos iOS e Android, é possível facilmente criar *dashboards* apenas arrastando itens sem precisar programar as aplicações nos dispositivos de controle, smartphones e computadores.
- Desvantagens
  - Produto gratuito até certo ponto, possuindo algumas funcionalidades apenas em um plano de assinante;
  - Servidor em nuvem o que gera dependência exclusiva da internet, não possui integração direta com assistente de voz.

2.5 Voiceflow

O Voiceflow é uma ferramenta que permite a criação de *skills* para assistentes virtuais como a Alexa e Google Assistente, através desta ferramenta online é possível criar *skills* de forma simples, sem conhecimento em programação ou uma linguagem específica. “Hospeda mais de 200 milhões de interações por ano, com cerca de sete mil aplicativos de voz desenvolvidos."(NEWVOICE, 2021).

As ferramentas do Voiceflow permitem a criação de aplicativos de voz para qualquer sistema compatível. “Os designs podem ser prototipados e preparados para publicação em várias plataformas, atuando como assistentes de voz autônomos, capazes de se adaptar ao ambiente"(SCHWARTZ, 2021).

O Voiceflow possui funções como falar e escolher, onde através de blocos de funções é possível projetar como será realizada a comunicação com a assistente virtual e qual ação ou resposta ela oferecerá. É possível estabelecer as opções de condições que podem ser selecionadas pelo usuário, também é possível adicionar blocos lógicos, dentre outras funções.

Com isso é possível criar, simular e por fim associar essas *skills* na assistente virtual de forma simples, sem uma linha de código explícito.

2.6 Biblioteca Espalexa

Espalexa é uma biblioteca padrão do arduino com a ideia de fácil utilização. Essa biblioteca emula partes do protocolo SSDP e da API Philips hue para fazer o dispositivo ser encontrado.

- Vantagens
  - Espalexa permite definir uma gama de valores, por exemplo, brilho, temperatura e, opcionalmente, uma cor, além do controle liga / desliga padrão;
  - Por padrão permite que até 10 dispositivos sejam controlados podendo ser expandido ou diminuído;
  - Se utiliza de um WEB *server* interno, sendo assim não depende de servidor de terceiros para funcionamento, diminuindo então a limitação de uso.
- Desvantagens
  - Não é conhecida por ser muito estável;
  - Nem sempre detecta os dispositivos facilmente;
  - Não tem tanta segurança de continuidade.

A biblioteca visa trabalhar com todos os Echos do mercado, mas há alguns pontos que precisam ser observados. Espalexa só funciona com um dispositivo Echo genuíno, não é garantido o funcionamento com emuladores e com o aplicativo Alexa para smartphones.

Capítulo 3

Metodologia

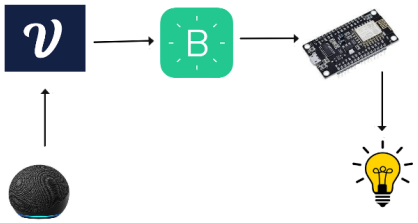
Neste trabalho é aprsentadp uma proposta de criar a integração entre o dispositivo Echo Dot Alexa e o módulo ESP8266 NodeMCU, via comando de voz, principalmente, de forma a criar um dispositivo capaz de automatizar uma residência.

O objetivo é que com um comando de voz direcionado ao dispositivo Alexa, consiga-se acionar o NodeMCU e este por sua vez controle a iluminação do ambiente da casa, além disso será utilizado um sensor de temperatura, a fim de obter dados da temperatura do ambiente e assim poder climatizar a residência.

Para a integração serão adotados dois métodos, o primeiro utilizará as ferramentas Blynk e Voiceflow e o segundo a biblioteca Espalexa.

No primeiro método será realizado, inicialmente, a integração entre o ESP e o Blynk que funcionará como um servidor intermediário entre a Alexa e o ESP. Após, cria-se uma *skill* para a Alexa, através do Voiceflow, que fará uma requisição, via API, ao Blynk controlando assim o ESP. Como demonstra a figura 3.1, onde pode-se ver os logotipos do voiceflow e Blynk.

Figura 3.1: Integração ESP8266 e Alexa



Fonte: elaborado pelo autor.

No segundo método, será utilizado a biblioteca espalexa, que emula partes do protocolo SSDP e da API Hue Philips, apenas para possibilitar que o dispositivo seja descoberto e

controlado pela Alexa. Com isso nesse método toda a integração será via código.

Para o desenvolvimento da programação será utilizada a IDE do Arduino, que é um ambiente integrado de desenvolvimento, assim será possível instalar as bibliotecas necessárias para o desenvolvimento de forma simples e intuitiva para o ESP8266 NodeMCU utilizando a linguagem de programação C/C++ por meio de seus comandos e instruções.

3.1 Método 1

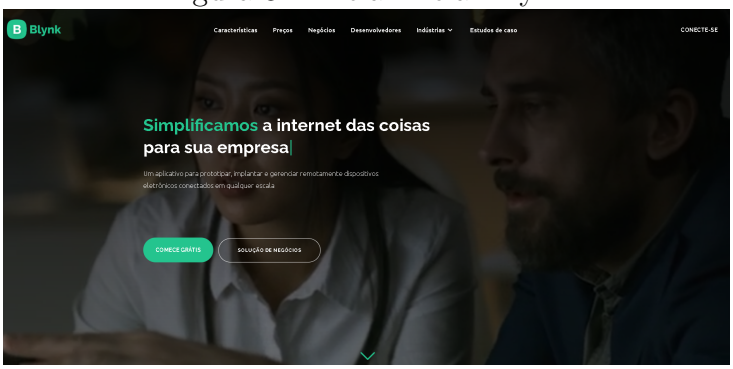
3.1.1 Configuração Blynk

O aplicativo Blynk possibilita a criação de um painel de comandos personalizável com várias funções que permitirão o controle da entrada e saída de dados, são botões, medidores, displays, blocos de instruções, dentre outras possibilidades. Para o projeto, serão utilizados botões virtuais para acionar um protótipo de iluminação residencial, essa ferramenta porém só consegue se comunicar com o NodeMCU mas não com a Alexa.

A ferramenta voiceflow possibilitará a criação de uma *skill* para o dispositivo Alexa, que através de um bloco de API, obterá e enviará dados para o aplicativo Blynk, se comunicando indiretamente com o ESP.

No primeiro passo será preciso configurar o aplicativo blynk, para isso é preciso acessar o site `https://blynk.io/`, e criar uma conta, a figura 3.2 mostra a tela inicial da página da aplicação.

Figura 3.2: Tela Inicial Blynk

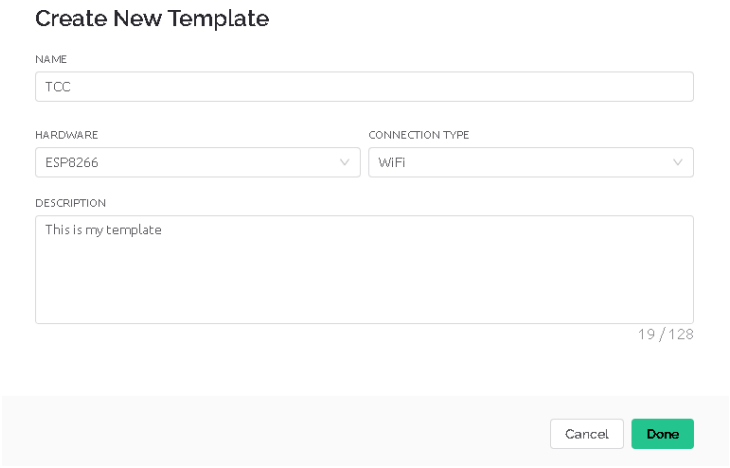


Fonte: *print screen* da tela inicial da aplicação Blynk

Após fazer o login, é necessário criar um template, como mostram as figuras 3.3 e 3.4 para isso, clica-se em templates, new template e adiciona as informações do dispositivo a ser programado, neste caso em particular seleciona o hardware como ESP8266 e o tipo de comunicação via WIFI.

Depois de criado o *template*, é necessário adicionar os dispositivos, as chaves que serão utilizadas e os pinos necessários para o acionamento. Para isso é necessário selecionar

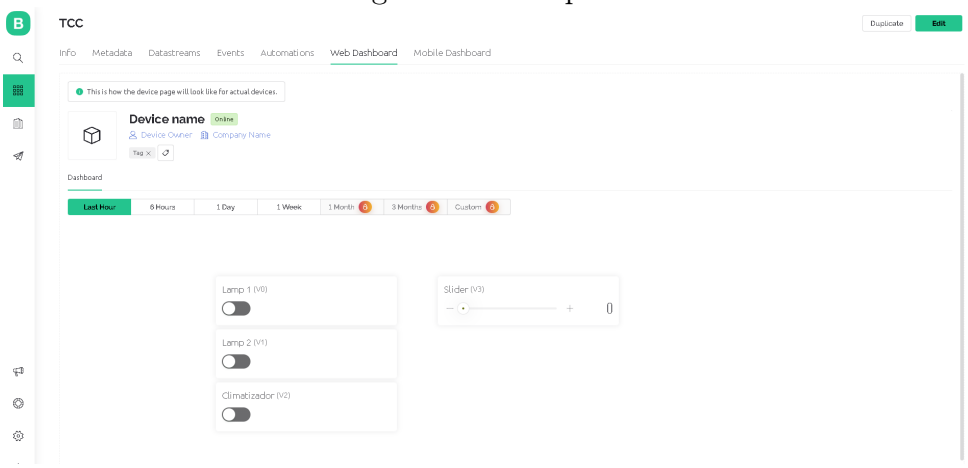
Figura 3.3: Criar template no Blynk



Fonte: elaborado pelo autor.

WEB *dashboard*, selecionar o componente para controle e configurar no pino do hardware selecionado anteriormente.

Figura 3.4: Template



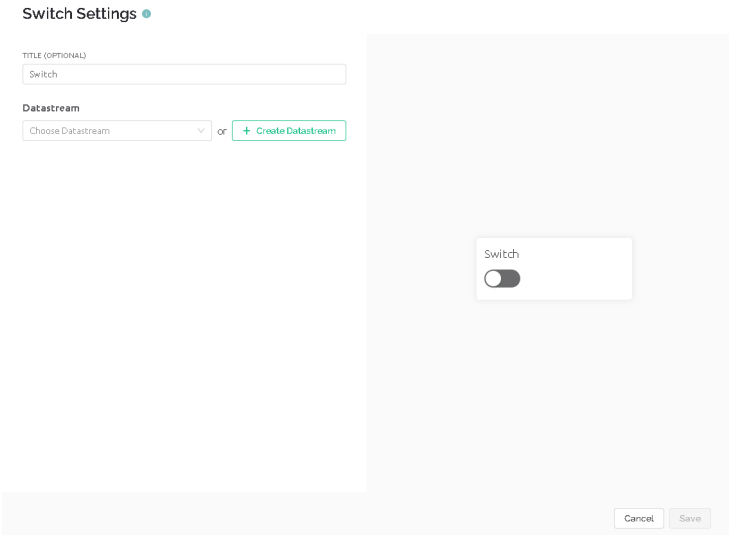
Fonte: elaborado pelo autor.

Para configurar o componente selecionado, neste caso um switch, clica-se em *create*, *datastream*, selecionando a opção *virtual pin*, como mostra a figura 3.5.

Assim pode-se configurar o componente, figura 3.6, adicionando, nome, escolhendo o pino a ser utilizado no ESP8266, tipo e dados como inteiro e valor padrão de inicialização.

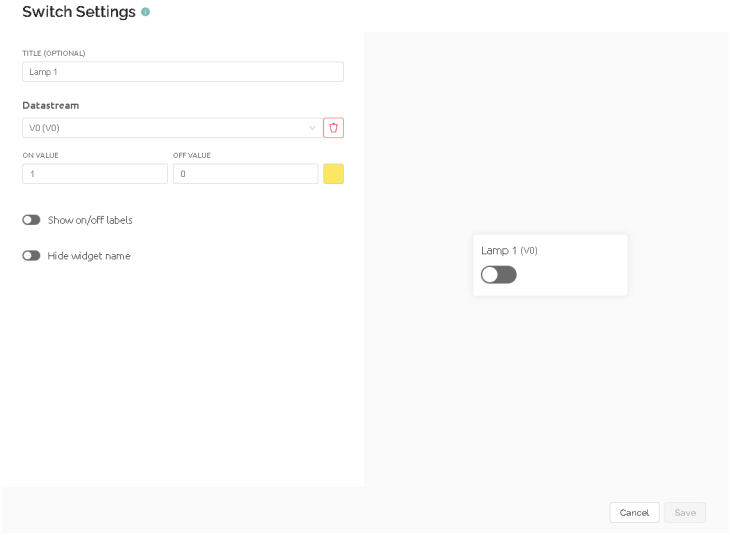
Depois de clicar em create é possível verificar o id do template e algumas informações do dispositivo a ser programado, como demonstrado na figura 3.7, essas informações serão utilizadas na programação do ESP8266.

Figura 3.5: Configurando Componentes



Fonte: elaborado pelo autor.

Figura 3.6: Configurando Componentes



Fonte: elaborado pelo autor.

Figura 3.7: Informações da aplicação criada

FIRMWARE CONFIGURATION

```
#define BLYNK_TEMPLATE_ID "TMPLcQ9vn2GD"
#define BLYNK_TEMPLATE_NAME "TCC"
#define BLYNK_AUTH_TOKEN "-q1R68xVofT_AHeqFVTPD6vXOYZ_s6vM"
```

Fonte: elaborado pelo autor.

3.1.2 Programando o ESP8266

A figura 3.8 mostra o resultado final do projeto no Blynk. Após realizar a configuração na aplicação Blynk, será necessário programar o dispositivo ESP para que o *dashboard*

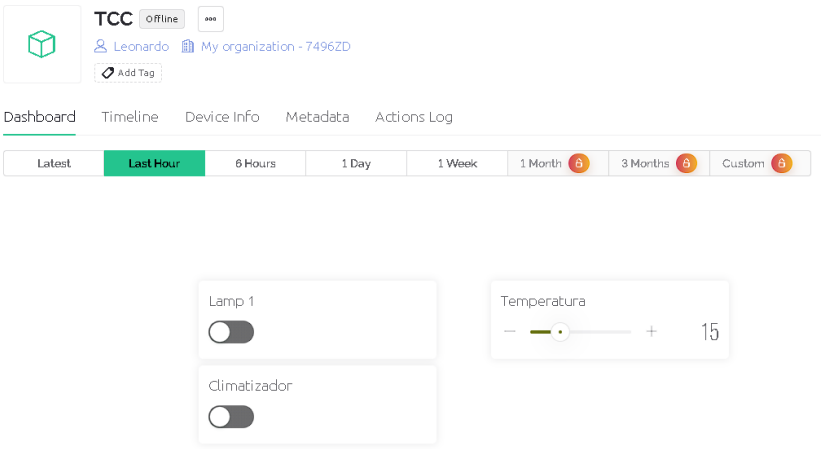


Figura 3.8: Projeto Final Blynk

criado se comunique com o mesmo. Para isso a IDE Arduino será utilizada.

Inicialmente, é necessário definir algumas variáveis que serão úteis no decorrer do código, como demonstra a figura 3.9.

Figura 3.9: definindo variáveis

```
1  #define BLYNK_PRINT Serial
2  #define BLYNK_TEMPLATE_ID "TMPLcQ9vn2GD"
3  #define BLYNK_DEVICE_NAME "TCC"
4  #define BLYNK_AUTH_TOKEN "-q1R68xVofT_AHeqFVTPD6vXOYZ_s6vM"
```

Fonte: elaborado pelo autor.

Essas variáveis receberão exatamente os dados obtidos na criação do template, elas servirão para permitir a comunicação com o servidor Blynk. A variável BLYNK_PRINT será utilizada para chamar a função Serial e apresentar o status da conexão na saída serial do Arduino. Será possível apresentar no monitor serial as informações de conexão, se bem sucedida, como o IP resultante dessa conexão, o status do servidor Blynk, e sua versão, e a rede conectada, como demonstra a figura 3.10.

As demais variáveis armazenam, respectivamente, a ID do template, o nome do dispositivo criado na plataforma e o token de acesso a aplicação criada;

Após, deve-se instalar as bibliotecas a serem utilizadas, figura 3.11.

Primeiramente tem-se a biblioteca ESP8266Wifi.h, já que o trabalho se baseia no módulo ESP 12 pois se fosse utilizado o módulo com o microcontrolador ESP32 seria utilizado a biblioteca WiFi.h. Essas bibliotecas são responsáveis por permitir a configuração dos módulos como estação ou ponto de acesso, conectando assim a rede WIFI do usuário para que seja possível a comunicação com a internet.

Já a biblioteca BlynkSimpleEsp8266.h é responsável por permitir a conexão entre o hardware do ESP8266 e o servidor Blynk, fazendo com que seja possível trabalhar os dados enviados e recebidos pela a aplicação Blynk, que foi configurada anteriormente. E

Figura 3.10: saía serial a conexão com o servior Blynk

```
[277] Connecting to brisa-2734614
[4505] Connected to WiFi
[4505] IP: 192.168.0.14
[4505]
    ___  __          __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v1.1.0 on ESP8266

 #StandWithUkraine    https://bit.ly/swua


[4516] Connecting to blynk.cloud:80
[6389] Ready (ping: 1640ms).
```

Fonte: elaborado pelo autor.

Figura 3.11: declarando bibliotecas

```
5   #include <ESP8266WiFi.h>
6   #include <BlynkSimpleEsp8266.h>
7   #include <DHT.h>
```

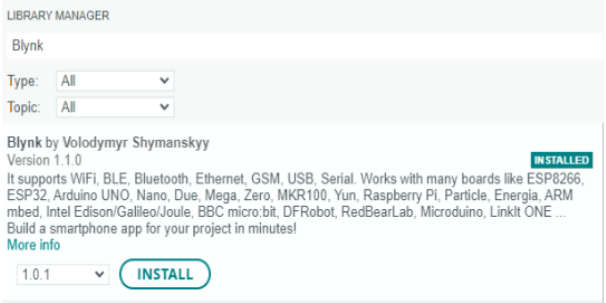
Fonte: elaborado pelo autor.

por fim a biblioteca DHT.h que é a biblioteca necessária para trabalhar com o sensor Dht11.

É importante ressaltar que existia uma outra versão do Blynk, abaixo da versão 1.0, que ainda possui o servidor em atividade, por isso caso a biblioteca Blynk esteja desatualizada pode-se esperar que o dispositivo ESP conecte-se a este servidor e a aplicação criada não funcione, já que para este projeto a versão utilizada do Blynk é a 1.1.0.

Pode-se verificar a versão da biblioteca, instalar e atualizar, caso necessário , na aba *Sketch*, da IDE, *Include Library,Manage Libraries*, e digitar a palavra Blynk no campo disponível para pesquisa, como mostra a figura 3.12

Figura 3.12: Verificanddo versão da biblioteca Blynk



Fonte: *print screen da IDE arduino*

A versão de Vlodymyr é a biblioteca que deve ser instalada, na sua versão 1.0.1, para

que a integração funcione.

Para estabelecer a conexão com a rede WIFI, as seguintes variáveis, representadas na figura 3.13, serão utilizadas.

Figura 3.13: Declarando variáveis

```
9   char auth[] = BLYNK_AUTH_TOKEN ;
10  char ssid[] = "NOME DA REDEE";
11  char pass[] = "SENHA DA REDE";
```

Fonte: elaborado pelo autor.

Na variável auth, será atribuído o *token* de acesso gerado na criação do template, somente com esse *token* é possível permitir que o hardware do ESP se comunique com a aplicação no servidor do Blynk.

A variável char ssid recebe uma string com o nome da rede WIFI na qual irá se conectar o hardware, assim como pass recebe uma *string* com a senha de conexão da rede. Com isso é muito importante garantir que todos os dados foram digitados de forma correta a fim de evitar erros de conexão e mau funcionamento, uma vez que a integração entre o Blynk e o ESP8266 depende totalmente de uma conexão com a internet.

Figura 3.14: Configurando sensor

```
12  #define DHTTYPE DHT11
13  DHT dht(D5, DHTTYPE);
14  float temp;
15
```

Fonte: elaborado pelo autor.

Para utilizar o sensor é preciso definir na função dht o pino e o tipo de sensor a ser utilizado, neste caso, como mostra a figura 3.14, dht recebe dois parâmetros, D5 que é o pino do ESP que receberá a leitura do sensor e DHT11 que é o sensor que será utilizado no projeto.

Após a declaração das variáveis e das bibliotecas, será necessário declarar as funções que irão fazer uso dessas.

A função void setup, demonstrada na figura 3.15, é a função principal do código, nela serão chamadas algumas funções e feitas algumas configurações no ESP. Ela inicia imediatamente após o Nodemcu ligar.

Primeiramente BLYNK_PRINT.begin vai ser responsável por iniciar a saída serial e mostrar o resultado final da programação do dispositivo.

Já Blynk.begin, vai receber as 3 variáveis declaradas no início do código, auth,ssid e pass, essas variáveis armazenam o *token* de acesso, o ssid da rede WIFI e *password*, respectivamente, como dito anteriormente, então é nessa linha que a conexão com a rede WIFI e a integração com o servidor Blynk é feita.

Figura 3.15: Função setup

```
17  void setup()
18  {
19    // Debug console
20    Blynk.begin(auth, ssid, pass);
21
22    BLYNK_PRINT.begin(115200);
23    delay(50);
24    pinMode( D1,OUTPUT);
25    pinMode( D2,OUTPUT);
26    dht.begin();
27
28  }
```

Fonte: elaborado pelo autor.

O método pinMode é utilizado para configurar um determinado pino do dispositivo que no caso deste trabalho é o ESP8266. Esta função recebe dois parâmetros, que são o pino e o modo de operação. Os pinos utilizados serão D1 e D2, todos configurados como saída. Os modos podem ser *OUTPUT*, *INPUT* ou INPUT_PULLUP, que correspondem respectivamente a entrada, saída e entrada com *pull-up* ativado. Por último, dht.begin permitirá a utilização dos métodos do sensor Dht11.

Figura 3.16: Função BLYNK_WRITE

```
29  BLYNK_WRITE(V1)
30  {
31  int valor=param.asInt();
32  if(valor)digitalWrite (D1,HIGH);
33  else digitalWrite (D1,LOW);
34  }
```

(a) BLYNK_WRITE(V1)

```
36  BLYNK_WRITE(V2)
37  {
38  int valor=param.asInt();
39  if(valor==0){valor=0;;}
40  else valor=1;
41  digitalWrite (D2,valor);
42  }
```

(b) BLYNK_WRITE(V2)

Fonte: elaborado pelo autor.

A função BLYNK_WRITE, figura 3.16, será responsável por obter o estado atual do pino virtual associado ao componente escolhido para uso na aplicação e assim poder repassar esse estado ou utilizá-lo para acionar o pino configurado do ESP.

A função recebe como parâmetro o pino virtual do Blynk que será tratado, neste caso o pino V1, dentro da função é declarado uma variável, valor, do tipo int, que guardará o valor de V1 através de param.asInt.

Verifica-se o dado da variável, através de um IF e repassamos para o pino físico do ESP através da função digitalwrite, que receberá o pino físico e o valor a ser transmitido por ele, neste caso D0 e a variável valor.

Na plataforma Blynk foram configurados os pinos virtuais V1 e V2, e .atribuídos a *switchs*. Como mostra a figura 3.1. Com isso repete-se a função BLYNK_WRITE(), para os dois pinos virtuais, passando como parâmetro cada um deles e substituindo os pinos do ESP a serem controlados, a saber D1 ou D2. Após finalizada essa etapa, o programa deve ficar rodando no ESP, desde que este esteja energizado, sem interrupções. Para isso utiliza-se a função void loop, figura 3.19.

Figura 3.17: Pinos virtuais

Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max
Lamp 1	Lamp 1		V1	Integer		false	0	1
Climatizador	Climatizador		V2	Integer		false	0	1
Slider	Slider		V3	Double		false	0	50

Fonte: *print screen da aplicação Blynk*

Figura 3.18: Função BLYNK_Write(V3)

```
44  BLYNK_WRITE(V3)
45  {
46  int temperatura=param.asInt();
47  if(temp>=temperatura)  digitalWrite (D2,HIGH);
48  Serial.print("Temperatura: ");
49  Serial.print(temp);
50  Serial.println(" ºC");
51  }
```

Fonte: elaborado pelo autor.

A função BLYNK_WRITE(V3) é responsável por configurar em qual temperatura a saída D2 será acionada, dessa forma é possivel que por comando de voz uma temperatura seja adotada para que D2 só seja acionada pelo sensor caso a temperatura ultrapasse essa condição.

Figura 3.19: Função loop

```
53  void loop()
54  {
55    Blynk.run();
56   temp = dht.readTemperature();
57  }
58
```

Fonte: elaborado pelo autor.

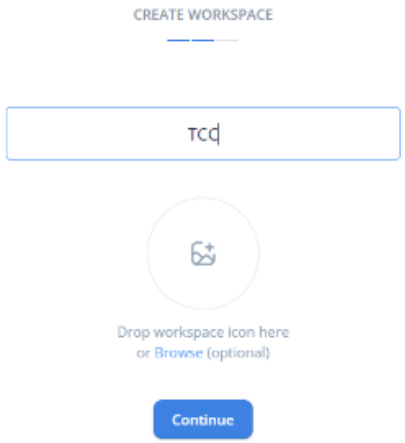
Por fim a função loop conterá um método da biblioteca Blynk, Blynk.run, que fará com que o dispositivo mantenha o código rodando e em comunicação com o servidor Blynk e uma variável chamada temp, que armazenará o valor da temperatura obtido por dht.readTemperature e ficará em execução.

3.1.3 Criando uma skill com Voiceflow

Feito a integração entre o dispositivo ESP e o Blynk, pode-se agora alcançar o objetivo da integração do ESP com a assistente virtual Alexa, através de uma *skill*. Para criar a *skill* será utilizado a aplicação Voiceflow, para isso primeiramente deve-se criar uma conta na plataforma através do link `https://creator.voiceflow.com/signup`.

Após criar a conta será necessário adicionar algumas informações e criar um *workspace*,

Figura 3.20: Criando uma área de trabalho

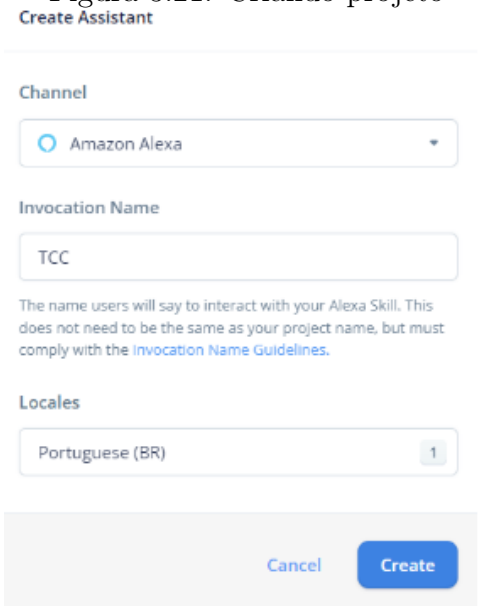


Fonte: elaborado pelo autor.

figura 3.30, antes de começar a desenvolver, lembrando que na versão gratuita só é possível criar um.

Feito isso é possível iniciar um novo projeto, com as especificações necessárias para o desenvolvimento, no caso deste trabalho serão selecionados Amazon Alexa, o idioma Português e a palavra que será utilizada para chamar a *skill*, esta é de livre escolha. As informações podem ser vistas na figura 3.21.

Figura 3.21: Criando projeto



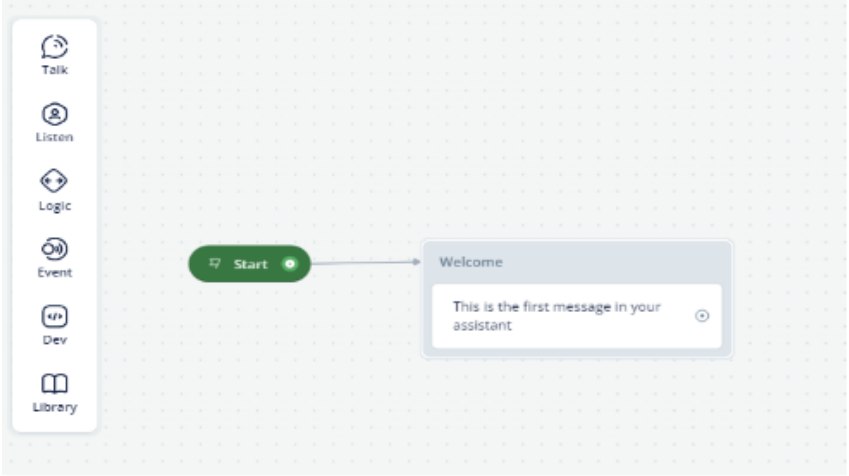
Fonte: elaborado pelo autor.

Criar uma *skill* com o Voiceflow é relativamente fácil, porém pode exigir um pouco de prática. A ferramenta tem sua estrutura baseada em blocos com funções distintas, com isso o primeiro bloco de um projeto sempre será o de invocação, que será acionado pela

palavra ou frase definida na criação do projeto.

Após finalizar o projeto será possível iniciar a *skill* com o seguinte comando “Alexa, abrir TCC”. O dispositivo Echo irá responder e executar aquilo que a *skill* for programada para fazer.

Figura 3.22: Exemplo inicial



Fonte: *print screen do bloco inicial da aplicação* voiceflow.

Após esse bloco inicial, figura 3.22, que todo projeto no voiceflow precisa possuir, deve-se incluir os demais blocos que irão configurar ou definir o que a *skill*, que será carregada na Alexa, será capaz de realizar.

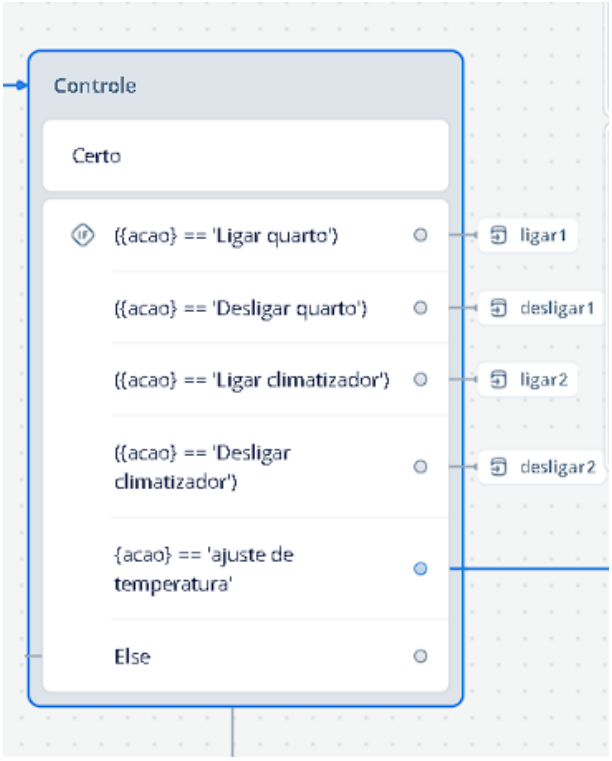
A ideia é fazer uma *skill* que consiga se comunicar com os pinos virtuais do Blynk e assim controlar os dispositivos ligados ao ESP realizando o acionamento e o desligamento das saídas do NodeMCU.

Para iniciar, será necessário criar um bloco de controle que seja capaz de realizar todas as ações solicitadas pelo usuário, como mostra a figura 3.23. Este será um bloco condicional, que basicamente será um if e else, sendo assim, se o comando dado for “ligar Quarto", o bloco ligar1 será invocado, caso o comando seja “desligar quarto"o bloco desligar2 será, selecionado. Assim também funcionará para ligar e desligar o climatizador.

Uma terceira condição será responsável pelo ajuste de temperatura, podendo o usuário dizer por comando de vóz um valor de temperatura que quando atingida pelo ambiente, acione o climatizador. Caso esta opção seja selecionada os blocos da figura 3.24 serão acionados. O bloco “Definir temperatura"armazena o valor dito pelo usuário em uma variável chamada “temperatura"e o envia para o pino virtual V3 do Blynk através do bloco “Temperatura definida", que é um bloco API, com uma solicitação GET, como mostra a figura 3.25.

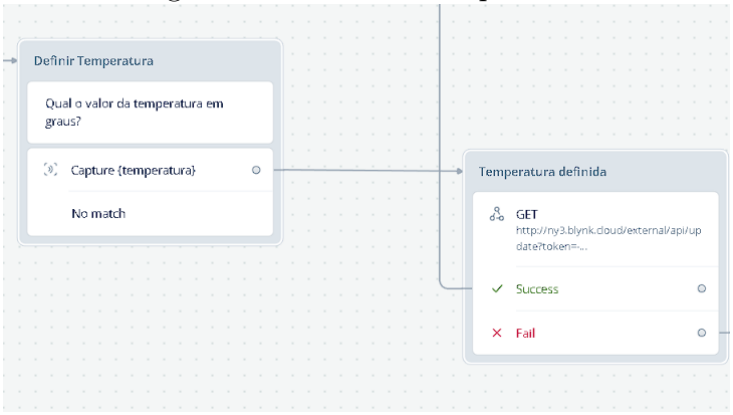
Os blocos “Ligar1", “desligar1", “ligar2"e “desligar2", seguem a mesma ideia do bloco “Temperatura definida". É possivel ver na figura 3.25, que a comunicação com a aplicação Blynk é feita através do token disponibilizado na criação do projeto no Blynk. Na figura

Figura 3.23: Blocos iniciais



Fonte: elaborado pelo autor.

Figura 3.24: Definir Temperatura



Fonte: elaborado pelo autor.

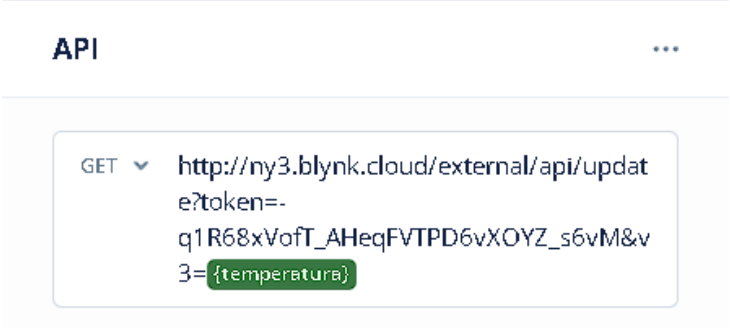
3.26 é possivel ver todos os blocos criados para o funcionamento da *skill* projetada no voiceflow.

3.2 Método 2

Primeiramente é necessário definir as bibliotecas que serão utilizadas, para a integração entre o ESP e a Alexa, como mosta a fugura 3.27.

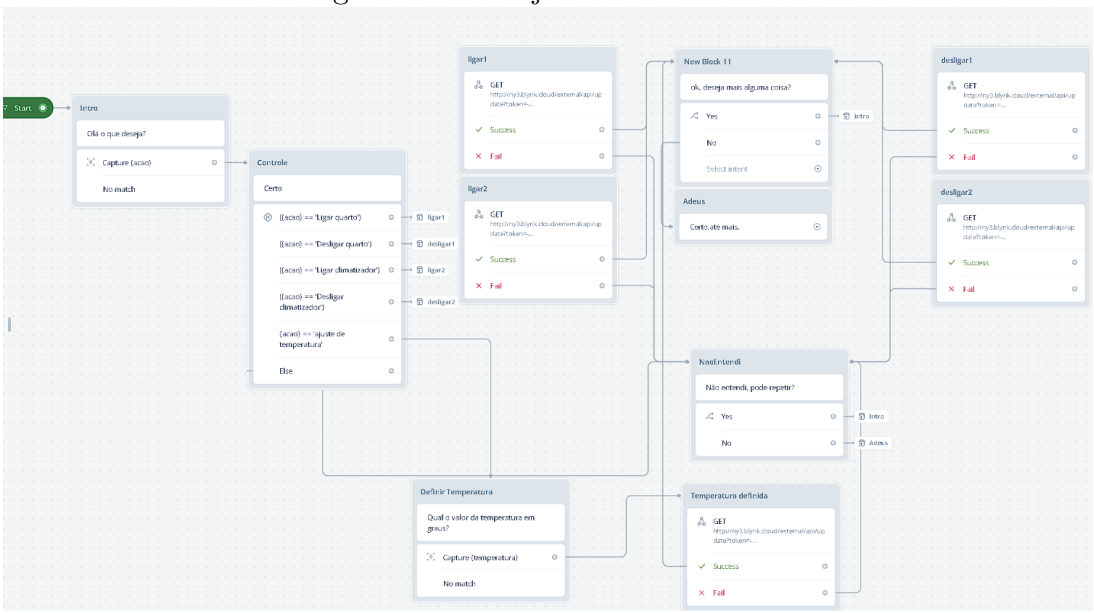
Novamente, será necessário utilizar a biblioteca ESP8266WiFi, que é responsável pela

Figura 3.25: Armazenando valor no pino V3



Fonte: elaborado pelo autor.

Figura 3.26: Projeto final Voiceflow



Fonte: elaborado pelo autor.

Figura 3.27: Instalação de bibliotecas

```
1  #include <ESP8266WiFi.h>
2  #include <Espalexa.h>
3  #include <DHT.h>
```

Fonte: elaborado pelo autor.

conexão à rede sem fio. A biblioteca Espalexa que será a responsável direta pelo controle do microcontrolador via comando de voz através da assistente virtual e por fim a biblioteca DHT, responsável pelos métodos do sensor Dht11, sensor de temperatura.

Como mostra a figura 3.28, será necessário definir algumas variáveis, primeiro o tipo do sensor, neste caso Dht11, após declarar-se o método dht que recebe a porta do ESP, que receberá os dados, e o tipo de sensor como parâmetros, a variável "temp"que receberá o valor da temperatura captado pelo sensor e ssid e *password* que armazena as credenciais para se conectar a rede WIFI.

Figura 3.28: Declarando variaveis

```
5   #define DHTTYPE DHT11
6   DHT dht(D5, DHTTYPE);
7   float temp;
8   const char* ssid = "brisa-2734614";
9   const char* password = "usvjjkmd";
10  Espalexa espAlexa;
```

Fonte: elaborado pelo autor.

Figura 3.29: Função setup

```
14  void setup() {
15    Serial.begin(115200);
16    delay(50);
17    pinMode(D1, OUTPUT);
18    pinMode(D2, OUTPUT);
19    digitalWrite(D1, LOW);
20    digitalWrite(D2, LOW);
21    dht.begin();
22    ConectarWifi();
23    espAlexa.addDevice("Led1", acionar_D1);
24    espAlexa.addDevice("Led2", acionar_D2);
25    espAlexa.begin();
26  }
```

Fonte: elaborado pelo autor.

Como dito anteriormente, essa função será responsável por iniciar algumas funções e configurar o dispositivo. Como mostra a figura 3.29, serão definidos os pinos D1 e D2 como saídas e serão iniciados em nível lógico baixo. Para adicionar um dispositivo na Alexa, é necessário o método espAlexa.addDevice, ele receberá como parâmetros o nome do dispositivo a ser controlado e a função que será responsável por controlá-lo.

Figura 3.30: Função ConectarWifi

```
42  void ConectarWifi() {
43    if (WiFi.status() != WL_CONNECTED) {
44      WiFi.mode(WIFI_STA);
45      WiFi.begin(ssid, password);
46      Serial.println("");
47      Serial.println("Connecting to WiFi");
48      while (WiFi.status() != WL_CONNECTED) {
49        delay(500);
50        Serial.print(".");
51      }
52      Serial.print("Connected to ");
53      Serial.println(ssid);
54      Serial.print("IP address: ");
55      Serial.println(WiFi.localIP());
56    }
57  }
```

Fonte: elaborado pelo autor.

ConectarWifi, ver figura 3.30, é responsável por conectar o ESP a rede sem fio, de acordo com o ssdi e *password* declarados anteriormente. Caso a conexão tenha êxito será mostrado no serial o nome da rede e o IP conectado.

Ainda é necessário configurar como as portas irão funcionar no decorrer da execu-

Figura 3.31: Função acionar_D0

```
45  void acionar_D1(uint8_t luz) {
46    if (luz) {
47      digitalWrite(D1, HIGH);
48    }
49
50    else {
51      digitalWrite(D1, LOW);
52    }
53  }
```

Fonte: elaborado pelo autor.

ção do código, para isso tem-se as funções acionar_D1, como mostra a figura 3.31 e a lógica se repete para acionar_D2 adicionando apenas uma outra condição caso a variável “temp"seja maior ou igual ao valor predefinido. A função acionar_D1 recebe como parâmetro uma variável do tipo uint8_t por padrão da biblioteca, de nome “luz".

Figura 3.32: Função loop

```
62  void loop() {
63    ConectarWifi();
64    espAlexa.loop();
65    temp = dht.readTemperature();
66
67  }
```

Fonte: elaborado pelo autor.

Essas funções serão passadas na função *setup* no método espAlexa.addDevice, e a depender do comando falado para a Alexa, a condição será atendida ou não, ligando ou desligando a porta referida do ESP.

E por fim a função *loop*, figura 3.32, que será responsável por manter o código em execução, invocando as funções ConectarWifi e espAlexa.loop, além de armazenar em "temp"o valor de temperatura lido pelo sensor.

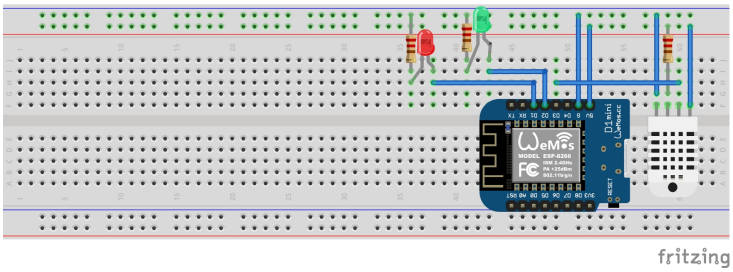
Capítulo 4

Resultados

4.1 Experimento

A seguir será descrito a execução do experimento, com ambos os métodos utilizados, será simulado um ambiente residencial e uma interação entre usuário e a assistente virtual Alexa. O circuito representado na figura 4.1 representa essa simulação.

Figura 4.1: Circuito montado



Fonte: elaborado pelo autor.

4.1.1 Experimento Método 1

Acender uma lâmpada

Usuário: “Alexa abrir projeto."

Alexa: “Ola o que deseja?"

Usuário: “Ligar quarto.’

Alexa: “Ok, deseja mais alguma coisa?"

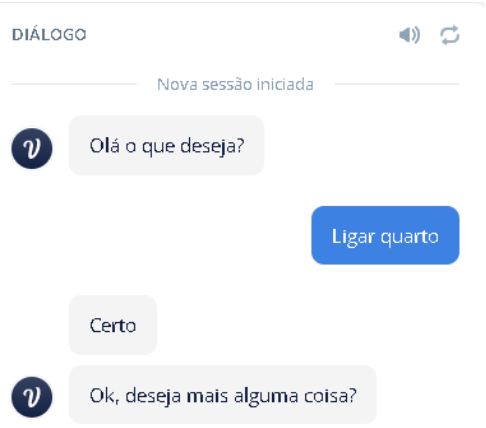
Definir temperatura

Usuário: “Alexa abrir projeto."

Alexa: “Ola o que deseja?"

Usuário: “Ajuste de temperatura."

Figura 4.2: Tela do voiceflow simulando o funcionamento da Alexa



Fonte: *print screen* da aplicação voiceflow sendo testada.

Alexa: “Qual o valor da temperatura em graus?"

Usuário: “30"

Alexa: “Ok, deseja mais alguma coisa?"

Em caso de erro ou comando invalido a alexa responderá com a seguinte frase "Não entendi, pode repetir?".

Apagar uma lâmpada

Usuário: Alexa abrir projeto.

Alexa: “Ola o que deseja?"

Usuário: “apagar quarto."

Alexa: “Ok, deseja mais alguma coisa?"

4.1.2 Experimento Método 2

Acender uma lâmpada

Usuário: “Alexa, ligar quarto."

Alexa: “Certo."

Apagar uma lâmpada

Usuário: “Alexa, apagar quarto."

Alexa: “Certo."

4.2 Dificuldades encontradas

Ao contrário do Arduino, o ESP não possui uma documentação tão extensa e detalhada, além disso, poucos são os trabalhos que o referenciam. Os métodos utilizados são pouco abordados em outros trabalhos, e alguns mal documentados, o que faz com que alguns erros sejam complicados de solucionar, tendo em vista que é necessário buscar em várias fontes distintas.

Foram encontradas algumas dificuldades ao longo do desenvolvimento e as quais serão abordadas neste tópico. No entanto pode-se afirmar que os resultados obtidos foram satisfatórios, ambos os métodos funcionaram dentro do que era esperado, a seguir cada método será exposto e analisado para no fim serem comparados e finalizar o estudo de caso.

4.2.1 Método 1

No método 1 trabalhamos com a aplicação Blynk e a Voiceflow, para realizar a integração da Alexa com o ESP, as aplicações são gratuitas e possuem muitas funcionalidades. Entretanto, apesar de serem ferramentas gratuitas, existem restrições que só podem ser desbloqueadas em uma assinatura, a aplicação Blynk possui assinaturas a partir e US$ 6,99 ao mês. Um dos problemas encontrados, no decorrer do projeto, foi o fato de existir dois servidores ativos para o Blynk, um mais antigo que não está mais em uso, porém pode ser associado ao projeto, e o mais atual, caso a versão da biblioteca instalada na IDE escolhida, IDE Arduino, esteja desatualizada o ESP se conectará ao servidor antigo e com isso o projeto não funcionará.

O Voiceflow também está limitado a dois projetos, com isso, se o usuário quiser criar mais que duas *skills* será necessário assinar um plano com mais recursos e vantagens, como neste projeto o intuito é realizar um estudo de caso entre os dois métodos propostos, não foi necessário. O Voiceflow é uma aplicação de desenvolvimento de habilidades para a Alexa, de fácil desenvolvimento, com configurações através de blocos, o que permite um ambiente interativo e amigável ao desenvolvedor.

Um dos erros encontrados foi na comunicação via API entre a aplicação Voiceflow e o servidor do Blynk, quando um erro acontece, devido a pouca documentação da aplicação, se torna complicado solucionar, além disso, como a comunicação entre o Voiceflow e o ESP depende dos servidores do Blynk, qualquer problema que influencie em uma má conexão com internet, como uma rede congestionada ou até mesmo uma falha na rede WIFI local, vai ocasionar em uma comunicação defeituosa.

Como o Voiceflow cria *skills* para a Alexa, é necessário deixar bem definido quais comandos a assistente receberá e qual será sua ação, todavia como os comandos definidos são textuais e na execução serão falados e não digitados, foi percebido que algumas frases não foram reconhecidas ou foram entendidas de forma errônea pela assistente.

4.2.2 Método 2

Neste método a biblioteca espalexa foi utilizada. Por padrão, é possível adicionar até um total de 10 dispositivos com a biblioteca, enquanto no método 1, foi trabalhado a ideia de *skills* para controlar o ESP. Com a espalexa cada componente adicionado ao código se torna um dispositivo que pode ser encontrado pelo aplicativo Alexa e aí que foram encontrados os principais problemas.

Como os próprios desenvolvedores afirmam, a biblioteca não é totalmente estável, com isso os erros mais comuns foram dos dispositivos conectados ao ESP não serem encontrados, mesmo o Noemcu estando na mesma rede, para isso foi necessário que ao longo do projeto o dispositivo Echo tivesse que ser reiniciado para que assim conseguisse encontrar os dispositivos conectados ao ESP, porém após o dispositivo ser encontrado não ocorreram problemas do tipo.

4.3 Análise dos resultados

Os resultados obtidos com ambos os métodos foram satisfatórios, ao testar o funcionamento pode-se verificar que o método 1, possui um pouco mais de lentidão no processo de execução inicial até o estágio final. Isso se deve porque o método 1 trabalha com servidores externos e necessita de mais comandos para realizar uma execução.

A aplicação Blynk é uma opção válida para projetos de automação com o ESP, envolvendo a internet, pois possui um bom tempo de resposta e confiabilidade nos valores obtidos do sensor de temperatura utilizado, não apresentou grandes variações ou erros de leitura, além de ser relativamente simples de implementar, configurar e desenvolver aplicações por meio de sua plataforma. Asssim também a plataforma Voiceflow, possui uma interface simples e de fácil desenvolvimento, com bom tempo de resposta e boa integração com a Blynk.

Desta forma com o primeiro método, a integração ESP e Alexa, foi bem sucedida e mostra-se uma ótima opção de desenvolvimento e estudos no uso do ESP8266 para a automação residencial.

No segundo método, a biblioteca espalexa permite ligar e desligar uma porta configurada do ESP, a biblioteca foi criada para controlar dispositivos de iluminação, com isso existe métodos para alterar a intensidade da luminosidade e a temperatura da cor, porém esses métodos não funcionaram tão bem nos testes, com isso o objetivo foi focado em ligar e desligar um dispositivo, que pode ser um led ou um relé, escalando assim a funcionalidade do projeto para qualquer equipamento elétrico ou eletrônico.

Ao contrário do primeiro método, a espalexa faz com que os pinos configurados do ESP sejam reconhecidos como um dispositivo inteligente, sendo assim, a única dificuldade é fazer o dispositivo echo reconhecer o dispositivo na rede.

Sendo assim, os dois métodos funcionaram de forma satisfatória e a seguir serão listadas as vantagens de cada um, assim também como as desvantagens.

- Vantagens método 1
  - Ferramentas gratuitas, até certo ponto.
  - Facilidade na implementação.
  - Bom tempo de resposta.
  - Compatibilidade entre as aplicações utilizadas.
- Vantagens método 2
  - Biblioteca funcional.
  - Totalmente compativel com o dispositivo Echo D ot.
  - Integração entre ESP e Alexa de forma direta, sem necessidade de aplicações de terceiros.
  - Biblioteca gratuita.
  - Possibilidade de configurar até 10 dispositivos por padrão.
- Desvantagens método 1
  - As aplicações não fazem integração entre a Alexa e o ESP de forma direta.
  - Aplicações particulares que podem ser restringidas.
  - Pouca referência bibliográfica para aplicar ao trabalho.
  - Servidores externos que fazem com que a segurança seja terceirizada.
  - Comandos precisam passar por um bloco inicial no voice flow.
  - Dois servidores Blynk ativos, o que pode gerar conflito caso a biblioteca esteja desatualizada.
- Desvantagens método 2
  - Biblioteca não atualizada.
  - Não funciona corretamente quando utilizada no aplicativo para smartphone.
  - Instabilidade ao identificar o dispositivo pela primeira vez.
  - Só é totalmente funcional nos métodos de ligar e desligar.
  - Dificuldade em configurar e obter os dados de temperatura do sensor por comando de voz.

Capítulo 5

Conclusão

Neste trabalho , foram apresentados dois métodos de integração entre a plataforma NodeMCU ESP8266 e a assistente virtual Alexa. O projeto foi feito de forma simples, a fim de demonstrar a facilidade de desenvolvimento de uma solução básica para automação residencial, e de avaliar o potencial do microcontrolador ESP nesse contexto.

A associação de um dispositivo comercial, Echo Dot, com o microcontrolador ESP fazendo com que o usuário se comunique, através de uma assistente virtual, por comando de voz, com elementos não inteligentes ligados ao ESP, é um exemplo de como projetos do tipo podem beneficiar públicos com necessidades especiais.

A plataforma do ESP mostra-se bastante promissora e eficiente no âmbito da domótica, uma vez que se utiliza de bibliotecas e funções do Arduino, associados com sua interface de rede sem fio que possibilitam inúmeras soluções para automação residencial.

Ambos os métodos foram eficientes no quesito tempo, com poucas falhas detectadas, nota-se que a biblioteca espalexa torna-se mais eficiente em relação aos comandos, já que são pré definidos, enquanto com o Voiceflow, a depender da frase de acionamento escolhida, pode acontecer uma má compreensão da assistente virtual. Vale salientar também que, no segundo método para que o dispositivo seja encontrado, é necessário que ambos os dispositivos Echo e ESP estejam conectados na mesma rede, todavia é possível acionar o ESP pelo aplicativo Alexa no smartphone em qualquer rede em ambos os métodos.

Em resumo, como o primeiro método trabalha com o conceito de *skill*, os principais problemas encontrados estão relacionados com mau entendimento da assistente sobre o comando dado pelo usuário e falha na comunicação com os servidores Blynk e Voiceflow, já que a *skill* é carregada diretamente na conta do usuário da Alexa. Enquanto que no segundo método, a ideia é simular dispositivos que serão associados às portas do ESP, sendo assim, os principais problemas encontrados são no reconhecimento destes dispositivos, o que fez com que fosse necessário reiniciar o dispositivo Echo em alguns momentos.

Para trabalhos futuros, pode-se abordar outras tecnologias associadas ao ESP8266, como outros sensores, câmeras, controles de persianas com motores analógicos e princi-

palmente objetos no contexto de segurança. Explorar outras assistentes virtuais, tentar desenvolver aplicações próprias, como uma assistente virtual e sistemas de controle offline e assim se livrar da utilização de servidores externos. Ainda é possível aplicar outros modelos do ESP, como o ESP 32, e associar outras plataformas ESP para que se comuniquem dentro de um mesmo projeto. Enfim, são várias as possibilidades que podem ser abordadas.

Referências

1. Alexa tem a maior popularidade entre os assistentes de voz. Newvoice, Rio de Janeiro, 28, jun. 2022, Brasil. Disponivel em: https://newvoice.ai/2022/06/28/alexa-tem-a-maior-popularidade-entre-os-assistentes-de-voz/. Acesso em: 15, fev. 2023.

2. Conhecendo o Blynk. Masterwalkershop, Mina Gerais, 14, nov. 2017, Brasil. Disponivel em: https://blogmasterwalkershop.com.br/blynk/conhecendo-o-blynk. Acesso em: 10, fev. 2023.

3. Com o Google Assistente, os brasileiros usam a voz e transformam seu dia a dia. thinkwithgoogle, Brasil, 03, jul. 2019. Disponivel em: https://www.thinkwithgoogle.com/intl/pt-br/futuro-do-marketing/novas-tecnologias/com-o-google-assistente-os-brasileiros-usam-voz-e-transformam-seu-dia-dia/#:~:text=Uma%20pesquisa%20recente%20mostra%20que,de%20voz%20usando%20o%20smartphone. Acesso em: 05, jan. 2023.

4. Echo Dot 4° Geração com Alexa. Tecnoeasyinformatica, Paraíba, 04, jun. 2021, Brasil. Disponivel em: https://tecnoeasyinformatica.com.br/produto/echo-dot-4-geracao/. Acesso em: 12, fev. 2023.

5. ESP8266 NODEMCU (ARDUINO COMPILER) PINS. NOW WITH WEMOS D1 MINI!, California, 21, set. 2018, Estados Unidos. Disponivel em: https://www.holdentechnology.com/2018/09/21/esp8266-nodemcu-arduino-compiler-pins/. Acesso em: 05, jan. 2023

6. MCROBERTS, Michael. Arduino Básico, São Paulo: Novatec, 2011.

7. OLIVEIRA, Sergio. Internet das coisas com ESP8266, Arduino e Raspberry PI. São Paulo: Novatec, 2017.

8. SCHWARTZ, ERIC Hal. Voiceflow fecha rodada de financiamento de $ 20 milhões para estender a plataforma de AI de conversação. Voicebot.ai, 29, agosto,2021. Disponível em: https://voicebot.ai/2021/07/29/

voiceflow-closes-20m-funding-round-to-extendconversational-ai-platform/ >. Acesso em: 11,jan.2023.

9. SGARBI, Julio André. Domótica inteligente: Automação residencial baseada em comportamento, São Bernardo do Campo, 2007.

10. Voiceflow agora permite criar apps de voz em um só lugar. Newvoice,Rio de Janeiro, 23, fev. 2021, Brasil. Disponivel em: https://newvoice.ai/2021/02/23/voiceflow-agora-permite-criar-apps-de-voz-em-um-so-lugar/. Acesso em: 20, fev. 2023.

11. WORTMEYER, C; FREITAS, F; CARDOSO, L. Automação Residencial: Busca de Tecnologias visando o Conforto, a Economia, a Praticidade e a Segurança do Usuário, Rio de Janeiro, 2005.

Apêndice A - Código fonte Método 1

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPLcQ9vn2GD"
#define BLYNK_DEVICE_NAME "TCC"
#define BLYNK_AUTH_TOKEN "-q1R68xVofT_AHeqFVTPD6vXOYZ_s6vM"
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "*******";
char pass[] = "*******";

#define DHTTYPE DHT11
DHT dht(D5, DHTTYPE);
float temp;


void setup()
{
  // Debug console
  Blynk.begin(auth, ssid, pass);

  BLYNK_PRINT.begin(115200);
  delay(50);
  pinMode( D1,OUTPUT);
  pinMode( D2,OUTPUT);
  dht.begin();

}
BLYNK_WRITE(V1)
{
int valor=param.asInt();
if(valor)digitalWrite (D1,HIGH);
else digitalWrite (D1,LOW);
}

BLYNK_WRITE(V2)
{
int valor=param.asInt();
if(valor==0){valor=0;;}
else valor=1;
digitalWrite (D2,valor);
}
```

```
BLYNK_WRITE(V3)
{
int temperatura=param.asInt();
if(temp>=temperatura)  digitalWrite (D2,HIGH);
Serial.print("Temperatura: ");
Serial.print(temp);
Serial.println(" °C");
}

void loop()
{
  Blynk.run();
 temp = dht.readTemperature();
}
```

Apêndice B- Código fonte Método 2

```
#include <ESP8266WiFi.h>
#include <Espalexa.h>
#include <DHT.h>

#define DHTTYPE DHT11
DHT dht(D5, DHTTYPE);
float temp;
const char* ssid = "******";
const char* password = "******;
Espalexa espAlexa;


void setup() {
  Serial.begin(115200);
  delay(50);
  pinMode(D1, OUTPUT);
  pinMode(D2, OUTPUT);
  digitalWrite(D1, LOW);
  digitalWrite(D2, LOW);
  dht.begin();
  ConectarWifi();
  espAlexa.addDevice("Quarto", acionar_D1);
  espAlexa.addDevice("Climatizador", acionar_D2);
  espAlexa.begin();
}
// Função para conectar com WIFI
void ConectarWifi() {
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);
    Serial.println("");
    Serial.println("Connecting to WiFi");
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
    Serial.print("Connected to ");
    Serial.println(ssid);
    Serial.print("IP : ");
    Serial.println(WiFi.localIP());
  }
}
```

```
// Função para o pin D1,D2
void acionar_D1(uint8_t luz) {
  if (luz) {
    digitalWrite(D1, HIGH);
  }

  else {
    digitalWrite(D1, LOW);
  }
}
void acionar_D2(uint8_t luz) {
  if (luz || temp>=30) {
    digitalWrite(D2, HIGH);
  }
  else {
    digitalWrite(D2, LOW);
  }
}
void loop() {
  ConectarWifi();
  espAlexa.loop();
  temp = dht.readTemperature();

}
```