**AUTOMAÇÃO RESIDENCIAL:
CONTROLANDO DISPOSITIVOS
REMOTAMENTE COM O MICRO CONTROLADOR
NODEMCU**

Matheus Machado de Mello Pereira Caldas

**AUTOMAÇÃO RESIDENCIAL:
CONTROLANDO DISPOSITIVOS
REMOTAMENTE COM O MICRO CONTROLADOR
NODEMCU**

Aluno: Matheus Machado de Mello Pereira Caldas

Orientador: Antonio Candea Leite

Coorientador: William de Souza Barbosa

Trabalho apresentado com requisito parcial à conclusão do curso de Engenharia Elétrica na Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

Agradecimentos

Agradeço este trabalho em especial aos meus pais e avós que me ajudaram na minha formação com recursos e uma base familiar e, sem o apoio deles, com toda certeza eu não teria chegado onde estou hoje. Agradeço também a todos meus amigos que fizeram parte dessa loucura chamada engenharia. Não podia deixar de citar também minha namorada, Daniela Monteiro, que me apoiou e esteve sempre ao meu lado em todos os momentos. A todos vocês o meu muito obrigado.

Tenho que agradecer também a todos os colegas, funcionários e professores que passaram pela minha vida durante esses cinco anos de PUC RIO. Aos que se formaram, estão se formando e que se formarão nos próximos anos compartilho a certeza que podemos e devemos mudar o nosso Brasil.

Resumo

O objetivo deste projeto é estudar e implementar um sistema de controle residencial onde é possível através de uma aplicação para smartphone controlar dispositivos como servo motores, leds, lâmpadas assim como ter em mãos ao usuário final dados relevantes da sua residência como a temperatura, umidade, alerta de presença, de chuva, vibração entre outros.

O projeto basicamente se divide em três, sendo uma parte da aplicação mobile, uma parte do servidor responsável por validar as requisições e interagir com serviços externos como, por exemplo, o Firebase Cloud Messaging e o micro controlador em si que se conecta com os diversos dispositivos e sensores.

Palavras-chave: Automação residencial; NodeMCU; Arduino; Web server; Aplicação mobile, Angular, Ionic;

Automation home system: Controlling remote devices using nodeMCU microcontroller

Abstract

This work aims to study and implement a home automation system where it's possible to control remote devices like servo motors, leds, lights and have in hands important data about your house, like temperature, humidity, rain and vibration providing alerts through a smartphone.

This project is divided in three parts: the first one is a mobile app with a friendly interface, the second one is a server responsible to validate web requests and interact with external services, eg: the Firebase Cloud Messaging. The last part is the nodeMCU microcontroller itself connected to multiple devices and sensors.

Keywords: Automation system; NodeMCU; Arduino; Web server; Mobile application; Ionic; Angular.

Sumário

Introdução	1
1. Arquitetura do projeto	2
2. Aplicação mobile	4
a) Arquitetura geral	5
b) Configuração	5
c) Dispositivos	6
d) Sensores e alertas	7
3. Servidor Web	9
a) API Rest	9
b) Arquivo de configuração	11
4. Firmware do NodeMCU	12
a) Arquitetura geral	12
b) Configuração	14
c) Controle e leitura de dispositivos	14
d) Bibliotecas	16
5. Eletrônica	17
a) Sensores	19
b) Dispositivos de saída	21
c) Ligações elétricas	23
6. Conclusões e Melhorias Futuras	31
7. Referências	32

Lista de Figuras

Figura 1 - Tela principal da aplicação mobile.....	5
Figura 2 - Configuração do aplicativo.....	6
Figura 3 Utilização do código para abrir a porta	7
Figura 4 Alerta recebido na tela inicial do smartphone	8
Figura 5 Micro controlador nodeMCU	19
Figura 6 Arquitetura do nodeMCU.....	23
Figura 7 NodeMCU conectado a todos dispositivos e sensores	24
Figura 8 Sensor de umidade e temperatura DH11.	25
Figura 9 Sensor de chuva.....	26
Figura 10 Sensor de vibração SW420	26
Figura 11 LED.....	27
Figura 12 Relé SRD-05VDC-SL-C.....	28
Figura 13 Ligação lâmpada com relé.....	28
Figura 14 Ligação servomotor.....	29
Figura 15 Servo motor.....	30
Figura 16 Buzzer.....	30

Introdução

No mundo de hoje, cada vez mais dispositivos mantêm uma conexão com a internet (ou pelo menos a uma rede local) e, através dessa conexão, é possível trocar informações efetuando a leitura de comandos e enviando de dados. Esse conceito é conhecido hoje como IOT (internet das coisas).

A ideia é que com o passar dos tempos todos os dispositivos terão algum tipo de conexão e será possível controlar qualquer coisa remotamente. E mais que isso, os dispositivos serão capazes de interagir entre si, disparando eventos e realizando tarefas de forma autônoma.

O objeto deste trabalho é mostrar que é possível construir um projeto de baixo custo onde com apenas uma aplicação, um servidor web e um micro controlador se consegue controlar dispositivos remotos em uma residência assim como conseguir dados a partir da leitura de diversos sensores.

A ideia aqui foi automatizar uma residência, sendo possível que o controle de diversos equipamentos seja efetuado de forma remota, eliminando a necessidade de se deslocar até um interruptor por exemplo.

Além disso, foi pensado em sistemas de alertas que avisam ao morador quando algum evento climático ocorre em sua residência como o início de uma chuva ou quando há um alerta de vibração, podendo ser calibrado para ser utilizado como alerta de um terremoto ou até mesmo da abertura de um cofre.

1. Arquitetura do projeto

De início, é importante explicar o funcionamento geral do sistema. Ele é composto basicamente de um servidor web, uma aplicação mobile rodando em um smartphone e o micro controlador nodeMCU conectado a sensores e dispositivos. Nessa seção será explicada a conceituação básica inicial, e nas seguintes cada parte específica do projeto será detalhada.

Servidor web: funciona de maneira central, gerenciando requisições tanto do aplicativo quanto do nodeMCU. Ele também é responsável por estabelecer contato com serviços externos como, por exemplo, o Firebase Cloud Messaging, serviço de notificação de aplicações. Tanto o aplicativo quanto o micro controlador conhecem apenas a camada de interface da API REST do servidor que faz, portanto, uma camada de abstração. Isso é interessante para o caso de se querer modificar apenas

a aplicação ou o micro controlador, bastando que os novos componentes saibam se comunicar pela API REST providenciada pelo servidor.

NodeMCU: se comunica por códigos de baixo nível, lendo e alterando valores de portas digitais e analógicas dos seus sensores e dispositivos. Em caso de algum alerta, a comunicação é sempre feita primeiro com o servidor que por sua vez, faz um processamento para validar ou não o alerta e aí sim enviar ao serviço do Firebse Cloud Messaging

Aplicação mobile: se comunica apenas através da API REST fornecida pelo servidor que responde se conseguiu ler/alterar os valores dos dispositivos e/ou sensores.

Firebase Cloud Messaging: Serviço externo responsável por enviar notificações push para o dispositivo. Quando há um alerta o servidor envia um request avisando e o serviço por sua vez comunica direto o smartphone do usuário final.

2. Aplicação mobile

A aplicação consiste em uma interface amigável para o usuário final, onde é possível controlar luzes, acionar fechaduras e ler a informação de sensores como temperatura, chuva, presença entre outros. A aplicação em si é simples, ela é responsável apenas por enviar requisições REST para o servidor quando os botões são clicados.

Inicialmente, foram comparados as vantagens e desvantagens de desenvolver a aplicação de forma nativa para a plataforma mobile Android e para o sistema iOS. Como grande vantagem do desenvolvimento nativo tem-se que desempenho é elevado para aplicações que exigem muito processamento ou que tenham uma interface gráfica complexa.

Como este não era o caso optou-se por utilizar uma plataforma híbrida, onde fosse possível escrever o código apenas uma vez e ter uma aplicação que rodasse no maior número de dispositivos possíveis. Assim, o custo atrelado do desenvolvimento do aplicativo caiu consideravelmente.

Dentre as diversas opções possíveis de sistemas híbridos no mercado foi escolhido o framework Javascript Ionic que se define como "O melhor framework open source para construir aplicações mobiles". É um projeto open source que tem como licença a MIT. Ele se manterá sempre "free" e possui uma imensa comunidade de suporte online.

Além disso, ele se define como "fully cross-plataform" o que significa que ele roda muito bem em qualquer tipo de plataforma com o mesmo código. Também possui uma ótima integração com plug-ins que fazem com que seja possível utilizar diversos recursos nativos como Bluetooth, impressões digitais entre outras.

Ele tem como pilar o uso do já consagrado framework Angular que nos últimos anos vem crescendo muito no desenvolvimento web em geral. Como tecnologias atreladas a este framework temos a linguagem Javascript/Typescript, o html para formatar a interface e o css para estilizar a aplicação. Também é possível a utilização de sass e less para modificar o estilo.

Figura 1 - Tela principal da aplicação mobile

a) Arquitetura geral

O aplicativo de forma geral é um sistema bem simples, mas ainda assim vale uma breve explicação de como ele funciona de forma geral. Toda a parte da interface com o usuário fica no componente Home. Esse componente injeta o AutomationProvider que é o serviço responsável por fazer a comunicação com o servidor.

Dessa forma, o componente apenas conhece o serviço do *frontend* e ele por sua vez se comunica pela API REST definida no servidor. Com isso, em uma mudança da API, por exemplo, é necessário apenas mudar o serviço, assim como se for interessante mudar a interface gráfica do aplicativo, é necessário modificar apenas o componente Home. Temos portando uma aplicação com suas partes bem independentes, mas que funcionam de forma harmoniosa.

b) Configuração

A configuração que deve ser feita na aplicação é bem simples. É preciso apenas definir no menu de configuração quem é o endereço do servidor web da automação.

Ex: <http://192.168.0.199:9082/automation>

Figura 2 - Configuração do aplicativo

c) Dispositivos

Como interface básica para o usuário tem-se:

Luzes: Controle de cômodos da residência, onde é possível através de um botão de On/Off acender e apagar as luzes do cômodo selecionado.

Portas: Controle da fechadura da porta da residência. Ao clicar para abrir, por medida de segurança, é solicitado um código PIN que é validado pelo servidor, permitindo a abertura da porta da residência.

Figura 3 Utilização do código para abrir a porta

d) Sensores e alertas

Sensores: Parte que indica a leitura de sensores e seus valores lidos.

Temperatura: Informa ao usuário a medição de temperatura realizada em graus Celsius.

Umidade: Informa ao usuário a umidade relativa do ar.

Alertas: Apresenta uma lista com os possíveis alertas disparados. Os alarmes disponíveis são os de chuva e alerta de vibração. Quando um alarme é disparado para confirmar que houve a leitura por parte do usuário ele precisa marcar como lido, mandando o alarme para a lixeira. Dessa maneira uma requisição é feita ao servidor avisando que de fato o alarme foi lido. O servidor por sua vez apaga a informação e o APP é atualizado sendo retirado da interface o alarme excluído.

Os alertas chegam na forma de notificações push e, dessa forma, são recebidas mesmo com o aplicativo fechado e tela do smartphone bloqueado. A imagem a seguir exemplifica o recebimento de um alerta de vibração.

Figura 4 Alerta recebido na tela inicial do smartphone

3. Servidor Web

a) API Rest

O termo API refere-se à sigla em inglês Application Programming Interface que significa em tradução livre Interface de programação da aplicação. Uma API é normalmente criada quando se quer tirar dependência do seu projeto a um tipo de aplicação. Então, cria-se a chamada API que nada mais é um nível de abstração, fornecendo uma interface conhecida por ambas as partes.

A API REST foi desenvolvida em Java utilizando-se o framework Spring. O servidor tem como responsabilidade principal receber requisições REST e enviar comandos para o micro controlador. A opção de se utilizar um servidor fazendo a ligação entre o nodeMCU e o aplicação foi que com um servidor o poder de processamento é muito maior, além da facilidade e versatilidade de desenvolver na plataforma Java se comparado ao desenvolvimento utilizando código em Arduino com o NodeMCU.

Além disso, era necessário um local seguro e robusto para validar as requisições da aplicação, onde algumas vezes é necessário ter a validação das requisições por senha e fica claro que é muito mais seguro que essas verificações estejam no lado do servidor. Como o servidor tem uma API REST bem definida ele fica independente no sentido que se algum outro projeto resolver utilizar outra aplicação para controle e leitura de sensores ele já está totalmente preparado, afinal, basta que a aplicação final realize as requisições para a API já conhecida.

Os métodos expostos pela API são:

Ler valor da temperatura atual

Exemplo de URL da requisição:

GET ***http://ipDoServidor:porta/automation/temperature***

Ler valor da humidade atual

Exemplo de URL da requisição:

GET ***http://ipDoServidor: porta/automation/humidity***

Ler valor do sensor de chuva

Exemplo de URL da requisição:

GET ***http://ipDoServidor: porta/automation/rain***

Pegar todos os alertas atuais

Exemplo de URL da requisição:

GET ***http://ipDoServidor: porta/automation/alerts***

Mudar estado de uma luz específica recebendo como parâmetro qual luz (1,2 ou 3 que será mapeada no micro controlador de acordo com a configuração explicada na próxima seção) deve ser modificada e se deve ir para o estado on ou off.

Exemplo de URL da requisição:

POST ***http://ipDoServidor: porta/automation/light?id=1&value=on***

Mudar estado de todas as luzes ao mesmo tempo recebendo se deve ir para o estado on ou off

Exemplo de URL da requisição:

POST ***http://ipDoServidor: porta/automation/allLights?value=on***

Mudar estado na porta, passando a porta (mapeada no arquivo de configuração que será explicado na próxima seção qual a porta a ser mudada) e a senha que o servidor irá validar ou não.

Exemplo de URL da requisição:

POST
http://ipDoServidor:porta/automation/door?id=1&password=1234

Inserir valor do token que será utilizado pelo serviço Firebase Cloud Messaging

POST
http://ipDoServidor:porta/automation/setToken?token=Ahadsa31h%232aNBcDEf34

Mandar uma nova notificação de alerta. O servidor primeiro verifica se o alerta já existe, se de fato for novo ele encaminha para o serviço do Firebase Cloud Messaging.

POST ***http://ipDoServidor:porta/automation/notification***

Body: JSON com a mensagem desejada do alerta que será enviado ao Firebase Cloud Messaging e de lá enviada ao dispositivo.

Excluir um alarme recebido como parâmetro da lista de alarmes.

POST
http://ipDoServidor:porta/automation/deleteAlert?alertMessage=AlertaDeVibacao

b) Arquivo de configuração

No servidor web é necessário algumas configurações que devem ser todas feitas no arquivo *application.properties* dentro da pasta config no diretório raiz do projeto.

automation.nodePath – Endereço onde o nodeMCU está rodando o servidor local para escutar requisições. Por padrão o nodeMCU é configurado para estar na porta 8080.

Ex: `http://192.198.0.181:8080`

automation.masterPass – Senha “master” para ser utilizada na aberta do portão.

Ex: 4321

automation.light1, automation.light2, automation.light3 – Pinos digitais a serem utilizados para ligação da luz número um, dois e três.

Ex: `automation.light1 = 14, automation.light2 = 12, automation.light3 = 13`

automation.door1 – Pino a ser utilizado para ligação do servo motor simulando a tranca da porta da residência.

Ex: `automation.door1 = 13`

automation.serverKey – Chave gerada pelo aplicativo que permite que ele receba notificações. Dessa forma, ao se enviar uma mensagem ao Firebase Cloud Messaging com essa chave, o serviço sabe para qual dispositivo enviar o alerta.

Ex: `automation.serveyKey=AAAASWXNnyQ:APA91bEq8fIVvQ9B6FyAoX_xM6BjE8GITnFejTNyVhFBK6BxdfVUXjCU_cxHb8ruEyDIrpC7nKa34_tXYNBNxxh1W5tQ_Vs4CncQVdc70Ps3bEegw848PcjAbzCvf73z8eLXfPJsm1aI`

4. Firmware do NodeMCU

O firmware de um sistema pode ser definido como um conjunto de instruções programadas diretamente no hardware de um dispositivo. O nodeMCU, e praticamente qualquer outro micro controlador no mercado, não vem com um sistema embarcado para operar e necessita que seu projetista desenvolva um software para executar certas tarefas.

Essa seção tem por objetivo explicar os detalhes da implementação do firmware pra este projeto. Um ponto que vale destacar é que assim como o código do servidor e do aplicativo este projeto é aberto para consulta e está disponível na plataforma github. Os links de cada projeto podem ser encontrados na seção de referência deste trabalho.

Primeiramente foram esquematizadas quais seriam as tarefas básicas necessárias que o micro controlador teria de desempenhar neste projeto. A primeira delas é que teria que lidar com a comunicação com entrada e saídas digitais e analógicas para controle de dispositivos e obtenção de dados de sensores externos. A segunda é que seria necessária a conexão com a internet, ou pelo menos com uma rede local que tivesse o servidor com acesso visível.

Pelos motivos supracitados e pelos demais já citados nas seções anteriores o nodeMCU foi escolhido como micro controlador para o projeto. Uma característica interessante e também já mencionada é que é possível desenvolver suas aplicações utilizando a IDE padrão do Arduino, facilitando e muito com o uso de depuração através das ferramentas já consolidadas de monitor serial e também de diversas bibliotecas já conhecidas.

Não há muito sentido na replicação do código fonte neste momento – o mesmo está disponível online – mas é importante mencionar as bibliotecas usadas, problemas encontrados, soluções achadas e a ideia básica de operação do software. Desenvolver para hardware sempre é mais complicado, seja por motivos de depuração e testes, seja também pelos custos e pelo seu limitado poder de processamento.

Por esses motivos optou-se por ter no firmware do nodeMCU apenas o código básico e mais simples possível das operações necessárias, sendo ele apenas um meio de campo entre a aplicação e o servidor. O firmware praticamente não processa nada, ele apenas recebe, chama e escuta requisições, verificando e/ou modificando suas saídas analógicas e digitais.

a) Arquitetura geral

Nessa seção serão explicitadas como o firmware funciona, suas características e metodologia utilizada. Ao plugar o nodeMCU na fonte de alimentação a primeira

etapa que ocorre é a execução do método `setup` que é responsável por fazer um `set` inicial das configurações necessárias para a execução do software. Inicialmente ela define a porta e o *data rate* em bits por segundo para transmissão serial. A transmissão aqui foi usada apenas para fins de depuração e se mostrou uma importante ferramenta para descobrir erros precocemente.

Além disso, o software faz um teste inicial ligando e apagando Leds e beepando o sensor de alarme para indiciar ao usuário que o hardware está funcionando da forma esperada. Em seguida é realizada a inicialização da biblioteca DHT (responsável pelo sensor de temperatura e pressão). E só então é iniciado o processo de conexão com a rede Wifi. Nesse momento utiliza-se o usuário e senha da rede definidos como constantes. O Wifi é conectado no modo *station* para conseguir ouvir requisições HTTP. A porta em que ele fica ouvindo as requisições foi definida como a de número 8080. Dessa maneira, para acessar o nodeMCU é necessário saber o seu IP e utilizar esta porta específica.

O código foi feito de forma que o micro controlador fica em loop tentando conectar a rede Wifi, não iniciando outros procedimentos antes de receber a mensagem que a conexão foi estabelecida com sucesso. Isso facilita e muito a depuração e fase de testes, sendo possível ver no monitor serial quando há um problema de conexão e quando a conexão é bem sucedida. Um ponto que vale destacar é que ao conseguir a conexão o IP definido para o micro controlador é exibido na tela. É de vital importância o conhecimento do IP do nodeMCU pois o servidor precisa saber onde ele se encontra na rede.

Isso se torna ainda mais importante em redes que utilizam DHCP onde o IP é quase que randomicamente gerado, podendo até ser modificado com o tempo. Dessa forma, caso não houvesse nenhuma forma de se apurar qual o IP do micro controlador no momento que ele recebesse outro IP na rede teríamos o problema do servidor nunca conseguir encontrar o nodeMCU e aplicação como um todo pararia de funcionar.

Continuando a explicação da lógica do firmware temos o fato de que em seguida caímos no método `loop` do Arduino que fica repetindo enquanto o micro controlador está com alimentação. Aqui são feitos checks simples para evitar a continuidade dos procedimentos em caso de falha na conexão com a rede Wifi.

Com o sucesso da requisição é chamado o método *checkNotifications* responsável por avaliar se existe algum novo alerta a ser exibido. O que ele faz basicamente são dois checks: um relativo ao sensor de chuva e outro ao sensor de vibração. Ele faz a leitura das entradas digitais (para o caso do sensor de vibração) e analógica (para o caso do sensor de chuva). Esse método possui ainda um algoritmo que 'limpa' o status de alarme.

Dessa maneira se o sensor de chuva detecta chuva ele lança um alerta. E ele só lançará outro alerta de chuva quando notar que o sensor ficou seco e novamente foi molhado. O mesmo ocorre para o sensor de vibração: ele só lança outro alerta quando percebe que o cessou a vibração e em seguida vibrou mais uma vez.

Em caso de novo alerta, tanto de chuva quanto de vibração é chamado o método *sendNotification* que tem como responsabilidade fazer uma requisição ao servidor alertando que um novo alerta foi recebido. Como parâmetro ele recebe apenas a mensagem a ser exibida que varia pro tipo de alerta(vibração ou chuva).

Continuando o entendimento do método loop, após checar se existe algum alerta/notificação ele entra em um modo de escuta de requisições HTTP. Dessa

forma, há estruturas condicionais para avaliar as seguintes requisições feitas do servidor para o nodeMCU:

- Temperatura
- Umidade
- Chuva
- Parar beep de alerta
- Escrita sensor digital
- Mudar servo
- Requisição inválida

b) Configuração

No software do node é preciso apenas configurar três variáveis.

networkName – Nome da rede Wi-fi a qual se deseja conectar.

Ex: Rede minha casa

networkPass – Senha da rede Wi-fi a qual se deseja conectar.

Ex: minhaSenha1234

serverIpNotification – Path com Ip, porta path para o endpoint de notificação do servidor. É aconselhável que o servidor seja levantado com o path /automation, ficando algo como

Ex: <http://192.168.199:9082/automation/notification>

c) Controle e leitura de dispositivos

Temperatura

Requisição feita quando se quer saber o valor da temperatura. Um método específico de temperatura é chamado que lê os sensores digitais atualizando o valor da temperatura lida, em seguida retorna seu valor.

Url da requisição: [**http://ipDoNodeMCU:8080/temperatura**](http://ipDoNodeMCU:8080/temperatura)

Umidade

Requisição feita quando se quer saber o valor da umidade. O mesmo método é chamado, pois ele atualiza tanto a temperatura quanto a umidade. Após a leitura o valor lido é retornado.

Url da requisição: [**http://ipDoNodeMCU:8080/humidity**](http://ipDoNodeMCU:8080/humidity)

Chuva

Requisição feita pelo servidor para se avaliar o valor do sensor de chuva. Como foi utilizado um sensor digital de 8 bits os valores podem variar de 0 a 1023. No firmware do nodeMCU foi considerado que abaixo de 1000 já se pode considerar como chuva.

Url da requisição: [**http://ipDoNodeMCU:8080/rain**](http://ipDoNodeMCU:8080/rain)

Parar beep de alerta

Solicitação do servidor para que o beep pare de emitir alerta sonoro. Isso é feito no nível de hardware apenas desligando a saída digital do beep. Essa solicitação é comum quando o usuário informa pela aplicação que deseja excluir o alerta. O servidor recebe o pedido e repassa ao micro controlador para desligar o beep.

Url da requisição: [**http://ipDoNodeMCU:8080/stopSound**](http://ipDoNodeMCU:8080/stopSound)

Escrita sensor digital

Requisição feita quando se deseja mudar o estado de um pino digital do micro controlador. Essa requisição exige também dois parâmetros que são recebidos na própria URL. O primeiro deles é a porta do nodeMCU que se deseja mudar o valor e

o segundo é o valor (0 para nível lógico baixo ou 1 para nível lógico alto). Segue um exemplo a seguir:

Url da requisição [**http://ipDoNodeMCU:8080/digital/write/12/0**](http://ipDoNodeMCU:8080/digital/write/12/0)

Nesse caso foi requisitado que o pino digital 12 (D6) recebesse o nível lógico 0.

Mudar servo

Requisição recebida quando se deseja mudar o estado do servo. Aqui não é recebido parâmetros para a angulação do servo, pois para esta aplicação o servo encontra-se ou na posição 0° (fechadura da porta aberta) ou posição 90° (fechadura da porta fechada). Vale lembrar que essa requisição só é feita pelo servidor caso a senha tenha sido validada corretamente pelo servidor. O micro controlador aqui é um mero cliente que não conhece a implementação da senha.

Url da requisição <http://ipDoNodeMCU:8080/changeservo>

Requisição inválida

Por último temos o caso de quando se tenta fazer algum tipo de requisição ao nodeMCU mas ela chega mal formatada ou não é nenhuma das citadas acima. Dessa maneira é apenas enviado um log ao monitor serial indiciando que a requisição é inválida e se retorna ao servidor a string 'Invalido' para o solicitante saber que algo errado ocorreu.

d) Bibliotecas

Bibliotecas no âmbito de desenvolvimento de software podem ser entendidas como um código que provem métodos/funções de forma modular. Dessa maneira, consultando apenas a documentação é possível realizar tarefas que já foram desenvolvidas anteriormente por outra equipe de desenvolvimento. É com uma caixa preta que o usuário não precisa entender seu funcionamento, mas apenas suas entradas e saídas.

Para este projeto em especial no firmware foram utilizadas algumas bibliotecas que serão explicadas a seguir:

Esp8266Wifi

Biblioteca do esp8266 para realizar diversas funções com Wifi. De acordo com sua documentação ela é baseada na biblioteca de Wifi Arduino, já bastante conhecida e debatida no mundo dos projetos com micro controladores. A Esp8266Wifi foi escolhida por ser a biblioteca oficial de Wifi para o micro controlador utilizado além de possuir uma excelente documentação.

ESP8266HTTPClient

Biblioteca do esp8266 para realizar requisições Wifi. Aqui vale salientar as diferenças entre esta biblioteca e a citadas a cima: enquanto a Esp8266Wifi tem objetivo de criar a comunicação Wifi, assim como fazer do micro controlador um *endpoint* para receber requisições a ESP8266HTTPClient serve para realizar requisições HTTP, sejam elas GET,POST,PUT ou DELETE.

Dessa forma, de maneira resumida a Esp8266Wifi cria a conexão a rede Wifi e coloca o micro controlador para ouvir comandos do servidor enquanto a ESP8266HTTPClient consegue enviar comandos para o servidor alertando por exemplo quando um novo alerta foi disparado.

Servo

Biblioteca do Arduino para controle de servo motores (neste projeto como já explicado foi utilizado um servo para simular abertura e fechamento de uma tralha de porta). É também uma biblioteca consolidada, simples e de boa documentação. Basicamente a única necessidade é definir qual pino estará *attached* ao servo motor e passar como parâmetro o valor do ângulo de rotação requerido. Dessa forma, o controle de servos fica bastante simplificado.

DHT

Foi utilizada a biblioteca de Arduino DHT responsável pela leitura dos sensores de temperatura e umidade. Sua documentação é bastante precisa e se mostrou suficiente para bom funcionamento com uma leitura precisa.

5. Eletrônica

NodeMCU

Foram cogitados alguns micro controladores para escolha deste projeto. A escolha final foi pelo NodeMCU. Ele foi nativamente desenvolvido para ter seu firmware na linguagem LUA. Entretanto, hoje já existem bibliotecas onde é possível utilizar a linguagem Arduino (basicamente C) e a própria IDE Arduino. Dessa maneira, devido a uma grande comunidade e ao já conhecimento técnico da linguagem optou-se por utilizar as bibliotecas em Arduino.

O grande concorrente de mercado hoje do nodeMCU é o Arduino Uno que tem como base o ATmega328P. Ele possui 14 pinos digitais de entrada e saída, sendo que 6 deles podendo ser controlado como saídas PWM (modulação de largura de pulso), 6 entradas analógicas e um cristal quartz de 16 mHz.

Em resumo, o Arduino tem como grande vantagem o grande número de GPIOs e uma plataforma já bastante consolidada, sendo indicada para iniciação em projetos de automação. A grande desvantagem do Arduino é que ele nativamente não possui um módulo Wifi. Como o projeto é totalmente voltado para IOT, precisar acoplar um dispositivo a mais para ter um requisito básico deste projeto parece ser algo não muito bom.

Já o nodeMCU é um módulo Wifi ESP8266 NodeMCU que é também uma placa para desenvolvimento que possui o chip ESP8266, um regulador de tensão 3.3V e uma

interface usb-serial utilizada para enviar os novos firmwares e para monitorar as portas seriais. Tem como padrão Wireless 802.11 b/g/n e uma antena já embutida. Além disso, tem suporte para até 5 conexões TCP/IP, possui 11 portas GPIO, sendo apenas uma analógica. Também possui também um conversor analógico digital (ADC).

A grande desvantagem do nodeMCU para este tipo de projeto é que ele possui nativamente apenas uma GPIO analógica que pode ser, dependendo do caso, um fator limitante. Todavia, uma simples solução para este problema poderia ser dada pela utilização de um multiplexador analógico onde é possível ter vários sensores utilizando apenas uma entrada analógica.

Para uma comparação final vale ser levantado alguns outros pontos ainda não mencionados: embora baixo para ambos, o custo do nodeMCU é um pouco menor que o do Arduino Uno. O node possui uma UART (universal asynchronous receiver/transmitir) que tem como tamanho de buffer 128 chars enquanto no ARduino esse tamanho é de apenas 62 chars.

O tamanho físico do NodeMCU é quase 3 vezes menor que o da placa Arduino Uno. O node vem com como padrão com 128kB de RAM enquanto o Uno tem apenas 2Kb. Além disso, o Node tem uma porta micro USB enquanto o UNO possui uma porta do tipo USB tipo B.

Por todos os motivos mencionados acima, em especial pelo fato de ser necessário o uso de um módulo esp8266 apenas para ter conexão Wifi com Arduino, optou-se por utilizar o nodeMCU que já possui esse módulo acoplado nativamente além das diversas outras vantagens citadas.

Os dispositivos conectados ao nodeMCU podem ser resumidos em dispositivos de saída e sensores. Na próxima seção serão apresentados os sensores e em seguida os dispositivos de saída e uma breve explicação deles. Em seguida será mostrado como eles se conectam com o micro controlador.

Figura 5 Micro controlador nodeMCU

a) Sensores

Sensores de uma forma geral transformam grandezas físicas da natureza, como temperatura, pressão, ondas sonoras, concentração de certa substância em corrente e/ou tensão para que seja possível uma medição do valor.

Os sinais elétricos podem ser digitais ou analógicos. Sinais digitais são aqueles que possuem dois valores bases: ligado e desligado, podendo ser, por exemplo, 0V e 5V ou 0V e 3.3V, dependendo do dispositivo.

Os sinais analógicos possuem uma faixa de operação e podem assumir qualquer valor dentro daquela faixa. Apenas para exemplificar, um sinal analógico operando entre 0V e 5V pode tanto assumir 0V e 5V como 2,33V ou 4,85V por exemplo.

Nesse projeto foram utilizados tanto sensores analógicos quanto digitais. Alguns deles inclusive oferecem a possibilidade de se utilizar os dois, podendo ser possível ler o estado "LIGADO/DESLIGADO" através da entrada digital ou se necessário uma leitura mais precisa através da leitura analógica.

Sensor de umidade e temperatura DH11.

É um sensor bem simples, de baixo custo que consegue ler temperaturas entre 0 e 50 graus Celsius. Além disso, é possível a leitura de umidade quando está entre 20% e 90%. Sua alimentação pode ser de 3 a 5V, tendo como limite temporário o possível valor de 5,5V.

A precisão de temperatura é garantida com um erro de 2 graus Celsius para mais ou para menos com tempo de resposta de 2 segundos. Nos extensivos testes realizados, o sensor se comportou de forma extraordinariamente bem considerando o seu custo irrisório.

Embora possua 4 pinos, utiliza-se apenas 3 (VCC, GND e DATA). O sinal recebido é digital e é interessante que o próprio fabricante disponibiliza uma biblioteca para ser usada, sendo necessário apenas passar o pino de DATA. A própria biblioteca é responsável por pegar o sinal digital lido e converter em valores reais de umidade e temperatura. Mais uma vez, o DH11 se mostrou uma excelente opção.

Havia outros sensores mais robustos e com maior precisão no mercado, todavia os eles tinham um valor que era na média 3 vezes maior. Como o uso da medição de temperatura seria em torno da temperatura ambiente e erros de medição não são cruciais para aplicação, optou-se pelo DH11 pela sua relação de custo benefício extremamente elevada.

Sensor de chuva

Foi pensado o sensor de chuva como uma opção interessante de monitoramento de uma residência. Através dele é possível que o usuário seja notificado quando começa a chover. Este dado pode ser interessante para que o morador tome alguma providência em caso de chuva forte por exemplo.

O sensor utilizado foi o "Rain Sensor Module" da openHacks que possui como características uma tensão de operação entre 3,3V e 5v, a possibilidade de se utilizar a leitura tanto digital quanto analógica, sensibilidade sendo ajustada através de um simples potenciômetro além de LEDs que indicam a tensão e saída digital.

O sensor possui um custo muito baixo e se comportou de forma muito boa nos testes realizados. Segundo o fabricante o sensor ainda apresenta um revestimento em ambos os lados que atua contra a oxidação e melhora também seu desempenho, sua condutividade e sua vida útil.

Nesse projeto em específico foi utilizada apenas a leitura do sinal analógico. Com o sinal digital é possível dizer apenas se está chovendo (nível lógico baixo) ou se não está (nível lógico alto). O uso da saída analógica dele é interessante, pois o micro controlador recebe uma tensão que varia de 0 a 3,3V e dessa forma é possível identificar se está chovendo pouco ou muito.

No código do micro controlador é utilizada a conhecida função *analogRead(pinNumber)* onde o sinal variando de 0 a 3,3V é 'mapeado' entre valores inteiros de 0 a 1023, sendo 1023 o sensor completamente seco. Para deixar margem a pequenas falhas de leitura, foi considerado 'chuva' quando o sensor efetuava uma leitura inferior a 1000. Acima disso, considerou-se que o sensor estava totalmente seco.

Apesar desta precaução, nos testes realizados o sensor se mostrou bastante preciso, indicando que havia algo molhado ao se colocar apenas uma gota de água. Mais

uma vez, assim como o sensor de temperatura e pressão, o sensor de chuva excedeu as expectativas e se mostrou uma excelente relação de custo benefício.

Sensor de vibração

O sensor utilizado para detectar vibração foi o SW-420. Como características técnicas o fabricante fornece a informação de que ele pode operar entre 3,3 e 5V, possui uma saída para leitura digital e leds indicadoras de tensão e do estado da saída digital. Além disso, possui um potenciômetro onde pode ser facilmente ajustado sua sensibilidade, sendo possível controlar quando a saída ficara em nível lógico alto (sem detecção) ou em nível lógico baixo (detecção de vibração).

Nos testes realizados ele se comportou muito bem. O sensor mudava sua saída de forma bem rápida quando era estimulado e o micro controlador não teve problemas na sua comunicação. Mais uma vez, um ótimo sensor que faz o que o fabricante tem como proposta e oferece isso a um baixíssimo custo.

b) Dispositivos de saída

Dispositivos de saída podem ser definidos de uma forma geral como dispositivos que transmitem informações ou dados através de uma mudança de estado, emitindo algum tipo de luz, som ou afetando de alguma forma o meio que o rodeia.

Os dispositivos de saída utilizados foram: 2 leds para simular uma luz ou qualquer outro dispositivo do tipo ON/OFF, um relé em ligação real com uma lâmpada, um servo motor para simular a fechadura automática de uma porta e um buzzer para efetuar o alerta sonoro para avisar sobre novos alertas.

Leds

O dispositivo de saída mais conhecido e utilizado para testes iniciais é sem dúvida o LED, ou diodo emissor de luz. Nesse projeto foram utilizados dois LEDS de cores diferentes para indicar que é bem simples o controle de saídas digitais. Claro que os LEDS poderiam ser facilmente substituídos por relés que por sua vez seriam capazes de ligar cargas consideradas altas. A seguir será explicado como foi utilizado um relé para este fim.

Relé

Um dos objetivos deste trabalho é mostrar que é possível controlar dispositivos reais de uma residência utilizando um micro controlador. Dessa maneira, embora os LEDs sejam didaticamente excelentes para mostrar que é possível controlar dispositivos remotamente eles não são suficientes para de fato mostrar a operação em dispositivos reais.

Cargas maiores precisam de correntes maiores e a ligação direta no micro controlador certamente iria causar um problema podendo inclusive queimar o nodeMCU. Por isso são utilizados dispositivos que conseguem fazer o 'meio de campo' entre o micro controlador que opera em 3,3V com cargas elevadas que podem operar em até 220V.

Apresentado esse problema, a solução encontrada para acionar uma lâmpada foi a utilização de um relé. O modelo escolhido foi o módulo relé SRD-05VDC-SL-C que tem como características básicas o controle de cargas de até 200V AC, com uma tensão de operação de 5V DC.

Vale ressaltar que embora as saídas digitais do nodeMCU consigam entregar no máximo uma diferença de potencial de 3,3V, o módulo funciona perfeitamente se a sua entrada VCC for acionada com 5V (utilizando a porta de nome Vin do microcontrolador, que por ser alimentado pelo USB consegue entregar os 5V nessa porta) e sua entrada de controle for de 3,3V.

O módulo possui na verdade 2 relés independentes e consegue operar com um tempo de resposta especificado pelo fabricante de 5 a 10ms, tempo esse mais que suficiente para a aplicação em questão. A saída para cada relé possui os contatos C(Comum), NA(Normalmente aberto) e NC(Normalmente fechado). Sua ligação com a lâmpada e o micro controlador será explicado em tópico específico.

Servo motor

O servo motor utilizado tem como objetivo simular a fechadura de uma porta. Dessa maneira ele poderia ser utilizado para melhorar a segurança do usuário do sistema sendo possível abrir e fechar fechaduras remotamente.

Por motivos de segurança para efetuar tal operação é necessário uma validação de senha por se tratar de dados sensíveis; modificar o estado de uma lâmpada de forma indevida não é tão problemático quanto abrir e fechar portas de uma residência. Essa validação é feita na comunicação entre o servidor e a aplicação móvel. O micro controlador em si não sabe da existência da senha em si. Esse detalhe será melhor explicado na parte de detalhamento de comunicação do aplicativo e servidor.

O servo motor usado foi o Micro Servo Motor 9g SG90. Ele possui como características básicas a tensão de operação podendo variar entre 3V e 7,2V, uma velocidade de 0,12seg/60graus (operando a 4,8V) sem carga, podendo operar entre -30 graus celsius e +60 graus celsius e possui o peso de apenas 9 gramas.

Buzzer

O buzzer utilizado foi o buzzer ativo que opera de modo muito simples e oferece um baixíssimo custo. Ele consegue operar entre -27 graus celsius e 70 graus celsius, tem uma corrente de operação de 30mA e tem uma dimensão de menos de 12mm. Sua saída de som impressiona bastante com capacidade de emitir a 85db a 10cm. Possui apenas duas pinagens, positiva e negativa (GND). Suas ligações com o nodeMCU serão melhor explicadas na seção que segue.

c) Ligações elétricas

Nesse tópico serão mostradas as ligações feitas entre o micro controlador, seus sensores e seus dispositivos de saída. Iremos começar apresentando o esquema básico do nodeMCU com suas respectivas portas.

Figura 6 Arquitetura do nodeMCU

A respeito da alimentação do node, optou-se por utilizar sua entrada microUSB pela facilidade que ela proporciona. Nos dias de hoje é trivial encontrar um cabo microUSB e conectar em uma porta de um notebook ou desktop. Também é

possível conectá-lo diretamente a uma fonte e esta por sua vez ligada na rede elétrica. Vale mencionar que todos os testes feitos com o micro controlador, especialmente pela facilidade de depuração, foram conectados a um desktop.

O node foi conectado em uma protoboard para que fosse possível aproveitar as conexões da mesma e se ter uma organização um pouco melhor das entradas e saídas. A primeira coisa feita foi utilizar uma trilha como GND e outra como VCC. Aqui vale mencionar que o VCC padrão neste projeto foi utilizado com uma voltagem de 3,3V.

Isso porque a maioria absoluta dos dispositivos tinham como tensão de operação 3,3V e as saídas analógicas e digitais no nodeMCU entregam exatamente essa diferença de potencial. A trilha do VCC foi conectada na entrada 3,3V (última da direita no esquema) e a do GND no GND (penúltima da direita no esquema).

Vale mencionar também que outras duas mini protoboards foram usadas, uma para isolar os leds e outra para o sensor de umidade e pressão. Isso foi feito única e exclusivamente por uma questão de organização, se fosse necessário redução de espaço poderia ser facilmente usado a mesma proboard e/ou solda das ligações.

A partir de agora quando for mencionado VCC será referente a tensão de 3,3V a menos que seja dito explicitamente uma outra voltagem (como ocorre com o caso do relé por exemplo).

Figura 7 NodeMCU conectado a todos dispositivos e sensores

Tendo feito a ligação básica das trilhas de VCC e GND foram ligados os seguintes dispositivos:

Sensores

Sensor de umidade e temperatura DH11.

O sensor de umidade e temperatura possui uma ligação bem simples, sendo necessária apenas a ligação do VCC e GND em suas respectivas trilhas. Além disso, é necessário também conectar o pino 2 em uma entrada digital que no caso foi escolhida a D3 (GPIO0). O pino 3 como especificado pelo fabricante não deve ser utilizado e possui a denominação de NC (*not connected*).

Figura 8 Sensor de umidade e temperatura DH11.

Sensor de chuva

Como já mencionado o sensor de chuva utilizado foi o "Rain Sensor Module" da openHacks. Suas conexões são como seguem:

PINO	Função
Nº 1	GND
Nº 2	VCC
Nº 3	Leitura Digital
Nº 4	Leitura analógica

Os pinos VCC e GND foram ligados nas respectivas trilhas. O pino de número 3 ou sinal digital não foi utilizado neste caso. Optou-se pelo uso exclusivo do sinal analógico para poder propiciar uma possível leitura da intensidade da chuva e não apenas uma variável dizendo se está chovendo ou não. Neste sensor, a entrada analógica foi ligada na porta ADC0 (A0 - analógica) do micro controlador nodeMCU.

Figura 9 Sensor de chuva

Sensor de vibração

O sensor utilizado foi o SW 420 e sua pinagem segue abaixo:

Este sensor também possui ligações bem fáceis, sendo apenas necessária a conexão de alimentação e de uma pinagem para leitura de dados. O VCC e o GND foram ligados em suas respectivas trilhas. O sinal digital para leitura se há um evento de vibração ou não foi conectado na porta D8(GPIO 15) do micro controlador.

Figura 10 Sensor de vibração SW420

Dispositivos de saída

LED

Os LEDs só precisam de uma alimentação DC para funcionar, sendo assim a única requisição foi conectar um resistor em série para obter uma limitação de corrente e assim não queimá-lo.

Leds são na verdade diodos emissores de luz e, portanto, possuem como característica intrínseca a passagem de corrente em apenas um sentido. Portanto, o sentido da ligação é de extrema importância para o correto funcionamento do LED. A

perna do anodo (parte positiva) foi conectada nas saídas digitais. A perna do catodo (parte negativa) foi conectada no GND.

Figura 11 LED

No nodeMCU foram utilizadas duas portas para exibição por LEDs, sendo uma para o LED vermelho e outra para o LED verde. As cores foram apenas para representar o que seriam a representação de cômodos diferentes de uma residência. Eles foram conectados nas portas D5 (GPIO 14) e D7 (GPIO 13)

Relé

O relé utilizado como já mencionado foi o SRD-05VDC-SL-C, sendo usado em um módulo com duas unidades. O seu esquema de ligação é especificado pelo fabricante como:

Figura 12 Relé SRD-05VDC-SL-C

A entrada GND foi ligada na trilha GND. A entrada VCC para este módulo em específico precisa de 5V e, portanto, não foi utilizada a trilha VCC 3,3V padrão mas sim a porta Vin 5V (última a esquerda no esquema do nodeMCU). Como foi utilizado apenas um relé (o de número 2 escolhido de forma aleatória) as entradas e saídas referentes ao relé de número um ficaram em aberto. O sinal de dados que indica ao relé se ele deve fechar ou não a chave foi conectado na saída digital D6(GPIO 12) do nodeMCU e é por ele que todo controle é feito.

Para a ligação da lâmpada com o relé e com o sistema elétrico a ligação foi a seguinte: O NO2, ou normalmente aberto foi conectado no fio comum da lâmpada que ao passar pela sua resistência vai até uma pinagem da tomada. O terminal COM2, ou comum, foi conectado na outra extremidade da pinagem da tomada.

Figura 13 Ligação lâmpada com relé

Com essa simples ligação, apenas variando o sinal de dados da entrada entre 0 e 3,3V é possível de forma remota controlar as luzes, podendo apaga-las ou acendê-las sem problemas de uma distância considerável, bastando para tal apenas conexão com a rede local.

Servomotor

A ligação do servo motor modelo 9g SG90 foi bastante trivial. O seu esquema é o que segue:

Figura 14 Ligação servomotor

Como sua tensão de operação varia de 3V a 7,2V sua entrada positiva foi conectada diretamente no VCC padrão de 3,3V e sua entrada negativa na trilha GND. A entrada de sinal por sua vez foi conectada na saída digital do nodeMCU D1 (GPIO5). Dessa forma, utilizando a biblioteca de servo é possível controlar os movimentos do servo apenas enviando dados pelo cabo de dados do servo.

Figura 15 Servo motor

Buzzer

A ligação do buzzer é, sem dúvidas, a ligação mais simples de todos os sensores e dispositivos de saídas. Isso porque ele precisa apenas de dois pinos. Um positivo e outro negativo. Dessa forma, o fio negativo ficou conectado diretamente na trilha padrão do GND. Já a alimentação positiva foi conectada na saída D2 (GPIO 04) do nodeMCU. Assim, quando se quer o disparo do buzzer basta apenas mandar um sinal digital de nível lógico alto na porta D2 que o buzzer começará a emitir um sinal sonoro bastante alto.

Figura 16 Buzzer

Vale ressaltar que, embora muitas vezes seja utilizado o termo “saída do micro controlador” ou “entrada do micro controlador”, isso foi uma escolha feita pelo projetista. As saídas podem virar entradas com uma simples modificação do firmware do micro controlador. O contrário também é verdade.

6. Conclusões e Melhorias Futuras

O objetivo deste trabalho foi atingido com sucesso, ou seja, foi possível controlar dispositivos remotamente assim como ler sensores à distância e, tudo isso, feito através de uma aplicação mobile. Além disso, foi provado que é possível desenvolver um sistema totalmente funcional de baixo custo. O sistema foi feito para rodar em diversas condições bastando apenas algumas poucas configurações exemplificadas nos capítulos de cada software.

A base central de toda a operação é o servidor web que providencia uma API REST conhecida e, portanto, caso no futuro seja necessário utilizar um micro controlador diferente até mesmo com mais recursos basta que o novo modelo use a mesma API que tudo irá funcionar como o esperado.

O mesmo ocorre para o caso da aplicação; se no futuro houver necessidade da aplicação ser desenvolvida em outra linguagem/plataforma basta apenas que a nova aplicação se comunique pela conhecida a API REST e, dessa forma, haverá uma comunicação com o servidor como esperado.

Como melhoria futura pode-se implementar outros tipos de sensores e dispositivos. A ideia básica permanece a mesma bastando apenas algumas poucas adaptações. A maior delas seria no caso de usar outro sensor analógico devido ao fato que o micro controlador utilizado possui apenas uma porta analógica. Entretanto, isso poderia ser facilmente solucionado com o uso de um multiplexador analógico e, com isso, o número de possibilidades de se utilizar sensores ou dispositivos analógicos aumentaria bastante.

7. Referências

[1] C. DORF, Richard; A. SVOBODA, James. Introdução aos circuitos elétricos. 8a. ed. [S.l.]: LTC, 2014.

[2] SEDRA, S.; SMITH, K.. Microeletrônica. 4ª. Edição, Pearson Makron Books, São Paulo, Brasil, 2005.

[3] David E. Johnson; John L. Hilburn; Johnny R. Johnson " Fundamentos de Análises de Circuitos Elétricos", 4ª Edição, Ed. LTC.

[4] BANZI, M.. Getting Started with Arduino. O'Reilly Books, California, USA, 2009.

Biblioteca de servo motor - Obtida do site <https://www.arduino.cc/en/Reference/Servo> acessado em 16 de julho (2018)

Biblioteca do sensor DHT - Obtida do site <https://github.com/adafruit/DHT-sensor-library>

Bibliotecas padrão do esp8266 DHT - Obtida do site <https://github.com/esp8266/Arduino>

Código fonte do servidor web - Obtido no repositório do github disponível em <https://github.com/matheuscaldasrj/automationSystemNode> e acessado no dia 23 de junho (2018)

Código fonte da aplicação mobile - Obtido no repositório do github disponível em <https://github.com/matheuscaldasrj/automationSystemApp> e acessado no dia 23 de junho (2018)

Código fonte do firmware do nodeMCU - Obtido no repositório do github disponível em <https://github.com/matheuscaldasrj/automationSystemNode> e acessado no dia 23 de junho (2018)

Figura 5 - Obtida do site da "Seeed Studio", disponível em <https://www.seeedstudio.com/NodeMCU-v2-Lua-based-ESP8266-development-kit-p-2415.html> acessado no dia 16 de junho (2018)

Figura 6 - Obtida do site da "Ibm", disponível em <https://www.ibm.com/developerworks/library/iot-nodemcu-open-why-use/Picture2.png> acessado no dia 16 de junho (2018)

Figura 8 - Obtida do site do "Filipe Flop", disponível em <https://www.filipeflop.com/wp-content/uploads/2017/07/450xN-9.jpg> acessado no dia 16 de junho (2018)

Figura 9 - Obtida do site do "Filipe Flop", disponível em <https://www.filipeflop.com/produto/sensor-de-chuva/> acessado no dia 16 de junho (2018)

Figura 10 - Obtida do site da "Eletrônica para todos", disponível em <https://i1.wp.com/electronica-paratodos.com/wp-content/uploads/2018/02/sw-420-vibration-sensor.png?resize=780%2C355&ssl=1> acessado no dia 16 de junho (2018)

Figura 11 - Obtida do site do "Filipe Flop", disponível em <https://www.filipeflop.com/wp-content/uploads/2017/07/Polaridade-LED-1.jpg> acessado no dia 16 de junho (2018)

Figura 12 - Obtida do site do "Techydy", disponível em <http://www.techydiy.org/wp-content/uploads/2017/02/opto-isolated-2-relay-module-connections-crop.jpg> acessado no dia 16 de junho (2018)

Figura 13 - Obtida do site do "Tec Dicas", disponível em <https://www.tecdicas.com/46/como-acender-e-apagar-uma-lampada-no-arduino-uno> acessado no dia 16 de junho (2018)

Figura 14 - Obtida do site do "Fatlion", disponível em <http://www.fatlion.com/sailplanes/images/jrconnector.png> acessado no dia 16 de junho (2018)

Figura 15 - Obtida do site do "Baú da Eletrônica", disponível em <http://www.baudaeletronica.com.br/micro-servo-9g-sg90-towerpro.html> acessado no dia 16 de junho (2018)

Figura 16 - Obtida do site do "Filipe Flop", disponível em <https://www.filipeflop.com/produto/buzzer-ativo-5v/> acessado no dia 16 de junho (2018)