

# **Projeto final**

## **Introdução à eletricidade e eletrônica**

### **Acionamento do circuito de relé com Internet das coisas**

**Vitor V. de Moura**

<sup>1</sup> Instituto Federal de Educação, Ciência e Tecnologia do Ceará(IFCE) – Campus Maracanaú  
61939-140 – Maracanaú, CE - Brasil

vitorverasm@gmail.com

**Abstract.** This project aims to facilitate the activation of a relay board using Internet of things through a Raspberry Pi that runs a communication client with the Slack platform.

**Resumo.** Este projeto tem como objetivo facilitar o acionamento de uma placa de relé utilizando Internet das coisas por meio de um Raspberry Pi que executa um cliente de comunicação com a plataforma Slack.

## **1. Introdução**

Com o passar dos anos, mais e mais dispositivos são criados com recursos Wi-fi e sensores incorporados, os custos de tecnologia e conexão estão baixando e as vendas de *smartphones* só aumenta. Todos esses fatores estão criando o ambiente perfeito para o mercado de Internet das coisas [Forbes 2014].

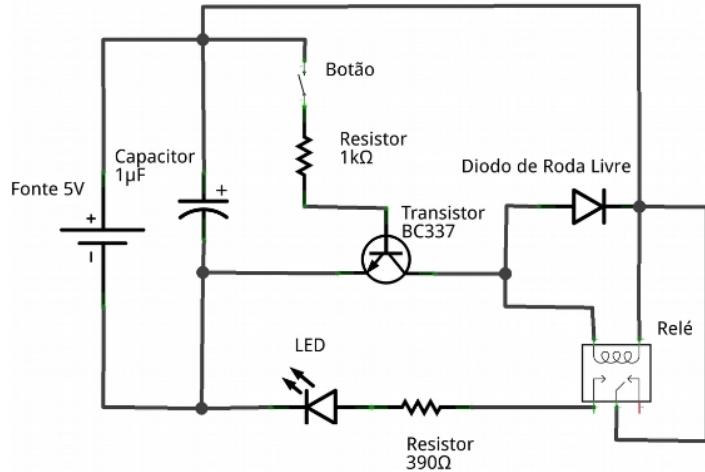
*Internet of things(Iot)*, consiste basicamente em conectar qualquer dispositivo com um interruptor de liga/desliga à internet. Estes podem ser dos mais variados como: cafeteiras, máquinas de lavar, carros ou as lâmpadas de casa [Meola 2016].

Este projeto tem como objetivo facilitar o controle deste tipo de dispositivos embarcados(*things*) por meio de um computador de baixo custo que servirá de central conectada à internet.

Todos os componentes usados para a elaboração desta configuração de automação serão citados adiante.

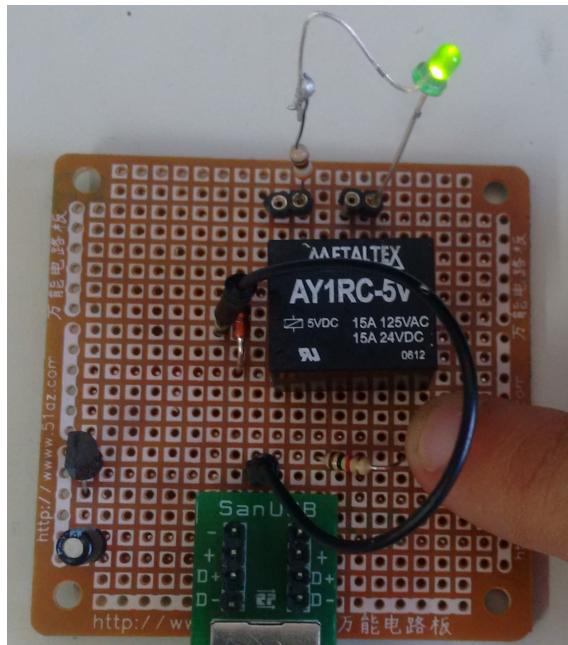
### **1.1. Circuito de acionamento de relé**

Como exemplo de circuito de acionamento *ON/OFF* deste projeto temos a placa de acionamento de relé feita previamente na disciplina de introdução à eletricidade e eletrônica. O esquema do circuito pode ser observado na Figura 1



**Figura 1. Esquema do circuito**

O resultado final da placa confeccionada pode ser visto na Figura 2 e todo o processo de elaboração está documentado em relatório contido no repositório: *GitHub* [Moura 2017a].



**Figura 2. placa de acionamento de relé**

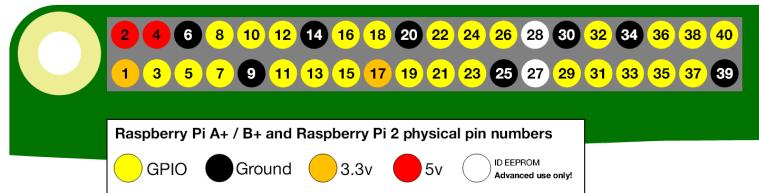
## 1.2. Raspberry Pi

Para a central que vai agregar os dispositivos a serem controlados foi utilizado um Raspberry Pi B+ modelo 1 que foi conectado à internet, como mostra a Figura 3.



**Figura 3. Raspberry Pi B+**

O Raspberry Pi(RPI) é um computador de custo e tamanho reduzidos que, neste modelo, possui 40 pinos sendo 26 deles pinos GPIO(*general purpose input/output*). O esquema com todos os pinos do RPI usado pode ser visto na Figura 4 e mais informações da documentação do modelo de Raspberry Pi utilizado podem ser encontradas no link: [RaspberryPi 2017a].



**Figura 4. Esquema de pinos do RPI usado**

### 1.3. Slack

A plataforma escolhida pra automatizar pela internet nossos dispositivos foi o Slack [Slack 2017c] onde nele vamos criar um *Bot*(robô) que irá responder à nossas requisições.

O Slack é um aplicativo multiplataforma que unifica todas as comunicações de um grupo seja ele time de desenvolvimento, setor de uma empresa, grupo de estudos entre outros. No Slack você pode gerenciar times, delegar tarefas, compartilhar arquivos, fazer conferências em vídeo, marcar datas de entrega e muito mais, tudo que torne o fluxo de trabalho de sua equipe mais dinâmico e colaborativo. Uma foto mostrando a plataforma pode ser vista na Figura 5

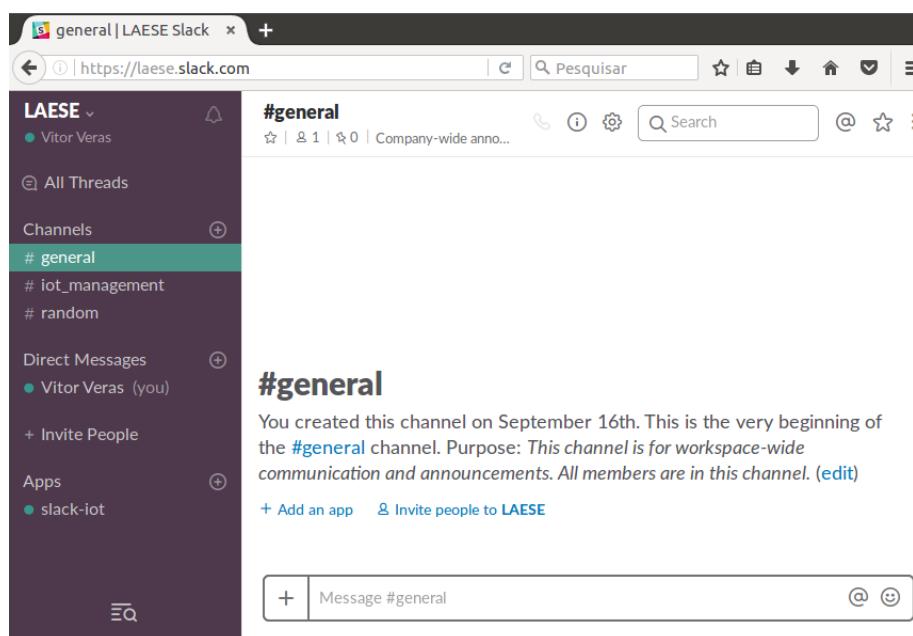


Figura 5. Plataforma web do Slack

### 1.4. Python3 e bibliotecas

Para fazer a comunicação com os servidores do Slack e manipular o RPI foi utilizado o Python3 e as bibliotecas *slackclient*, *re*, *json*, *psutil*, *time* e *rpi.gpio*.

A biblioteca *slackclient* permite que você crie aplicativos conectados com o Slack e com o *Slack Real Time Messaging (RTM) API*, toda a documentação da biblioteca pode ser encontrada em: [Slack 2017b]. Foi utilizado também a biblioteca *json* para exibir uma cópia da solicitação feita ao *Bot* no terminal por meio do *json.dump* que recebe um objeto e retorna uma *String*. Outra biblioteca utilizada foi a *re(regular expressions)* que serve para comparação de padrões de texto, assim pode-se verificar se a mensagem enviada ao *Bot* corresponde à alguma das reservadas para ativação do circuito. Por fim utilizamos a biblioteca *rpi.gpio*, *time* e *psutil* para respectivamente controlar os pinos de *GPIO*, configurar intervalos de tempo e receber informações sobre o sistema como porcentagem de memória e processamento.

## 2. Desenvolvimento

Este capítulo mostra a preparação do projeto desde as etapas mais básicas como por exemplo habilitar a conexão SSH(*Secure Shell*), conectar os dispositivos embarcados ao RPI até etapas mais complexas como a adaptação do código para o *Slack Workspace*.

### 2.1. Configurações iniciais do Raspberry Pi

O Raspberry Pi é um minicomputador e como um computador deve ter sistema operacional, neste caso, seu sistema operacional deve ser instalado em um cartão SD que será colocado no RPI. É necessário então fazer o *download* do sistema operacional, neste projeto foi utilizado o Raspbian *Stretch Lite* versão 4.9 que é uma adaptação mínima do Debian para a arquitetura ARM(processador do RPI), o link para a página do Raspbian é: [RaspberryPi 2017c].

Após feito o *download* do *Raspbian Lite* deve-se gravar o sistema no cartão SD, para isso, foi utilizado o programa Etcher, Figura 6, disponível em multiplataformas(Linux, Windows e Mac) e de código aberto, com documentação e link de *download* em: [etcher 2017]. Por meio do Etcher basta selecionar a imagem do Raspbian, o cartão SD e gravar.

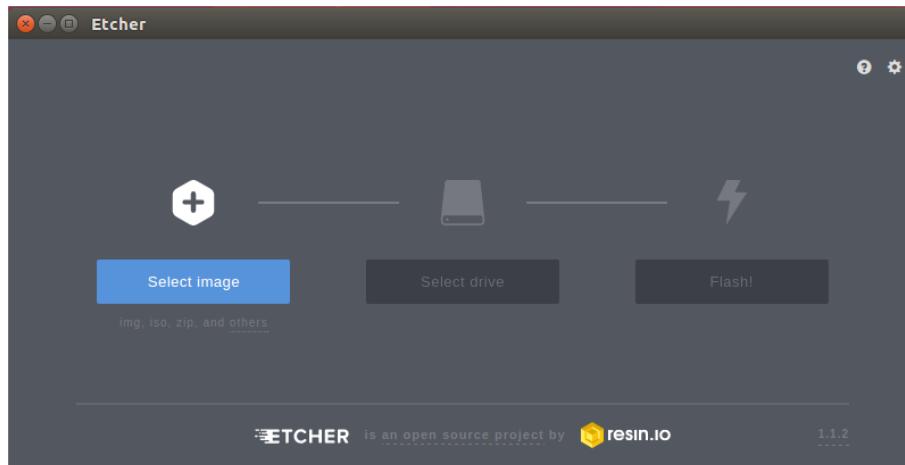


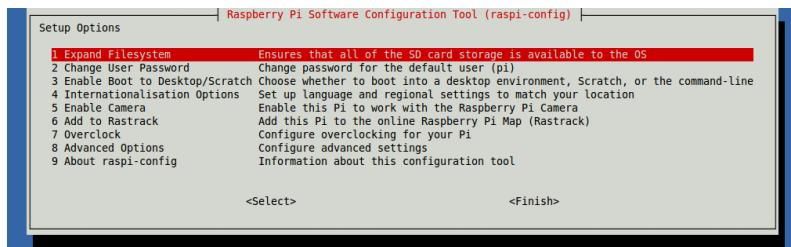
Figura 6. Etcher

Agora pode-se fazer a inicialização do sistema no RPI, utilizando o usuário e senha padrão do Raspbian, pi e raspberry respectivamente. O RPI possui uma saída de vídeo HDMI, portanto pode-se configurá-lo utilizando monitor e teclado, porém para este projeto o RPI foi configurado remotamente por meio de conexão SSH.

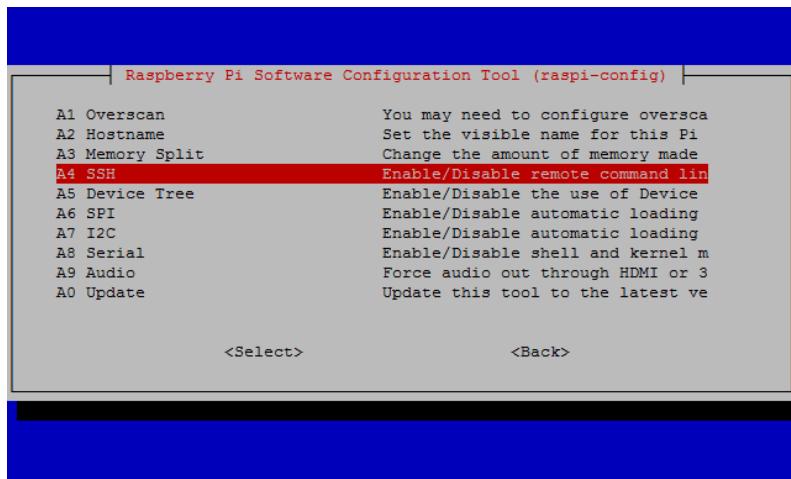
Para isso, foi necessário habilitar a conexão SSH, que neste modelo de Raspberry Pi vem desabilitada de fábrica. Deve-se ligar o RPI pela primeira vez com monitor e teclado, logar-se no sistema e digitar o comando:

```
$ sudo raspi-config
```

prosseguir para *advanced options* e depois SSH como mostrado nas Figuras 7 e 8.



**Figura 7. raspi-config**



**Figura 8. habilitando SSH**

Tendo habilitado o SSH, basta conectar-se remotamente ao RPI utizando o comando:

```
$ sudo ssh pi@[RPI-IP-ADDRESS]
```

Agora deve-se clonar o repositório do *GitHub* que contém o projeto, para isso é necessário instalar o *git* no RPI com o comando:

```
$ sudo apt-get install git
```

e depois disso clonar o repositório com:

```
$ git clone https://github.com/vitor-veras/slackbot_iot.git
```

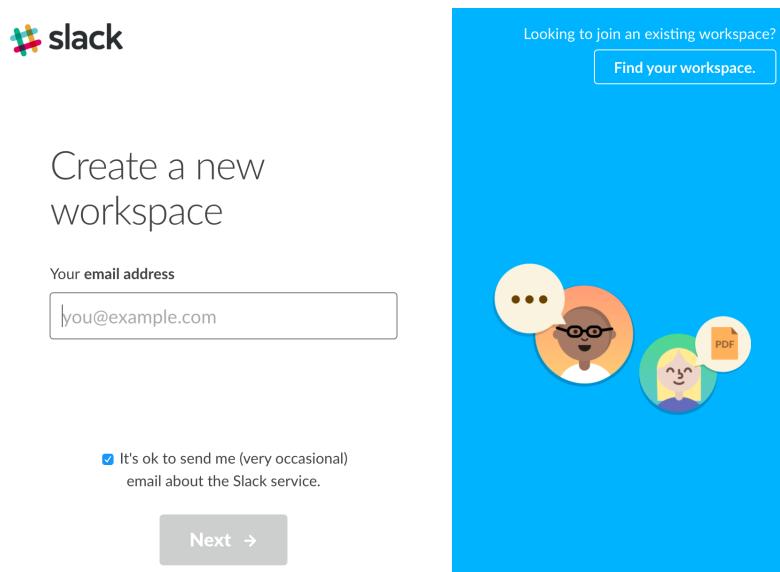
Prosseguindo, temos que instalar o Python3 e as bibliotecas necessárias(*slackclient*, *psutil* e *rpi.gpio3*), basta executar os comandos:

```
$ sudo apt-get install python3 python3-pip -y
$ sudo apt-get install python3-dev python3-rpi.gpio -y
$ pip3 install slackclient
$ pip3 install psutil
```

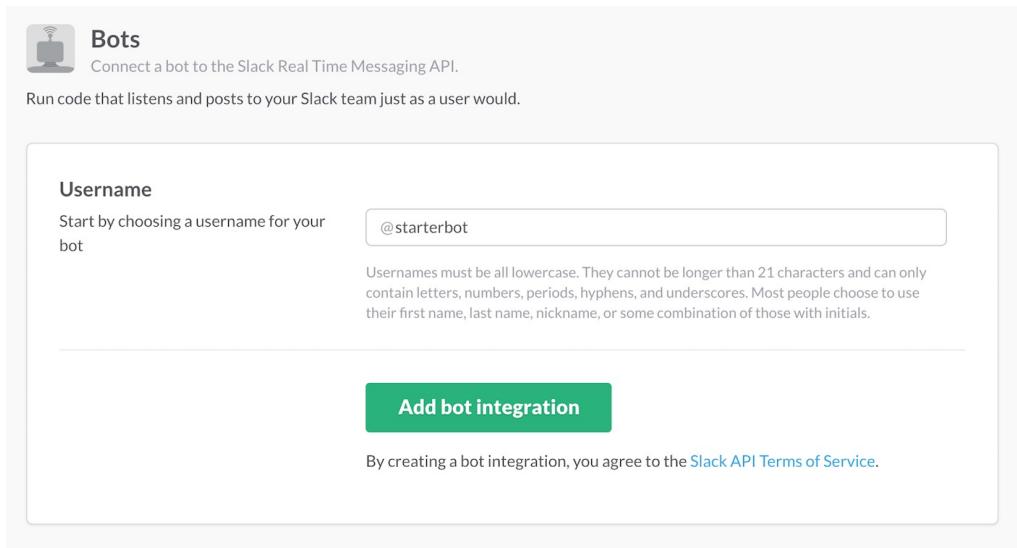
Tendo feito as configurações iniciais prosseguimos para a configuração da plataforma Slack.

## 2.2. Configuração do Slack *Workspace*

Primeiramente siga para [Slack 2017c] e crie um novo *workspace* com o nome desejado. Agora deve-se criar o *Bot* que controlará os dispositivos, para isso, vá ao link: [Slack 2017a] e escolha nome e descrição do robô, este processo pode ser visto nas Figuras 9 e 10 abaixo.

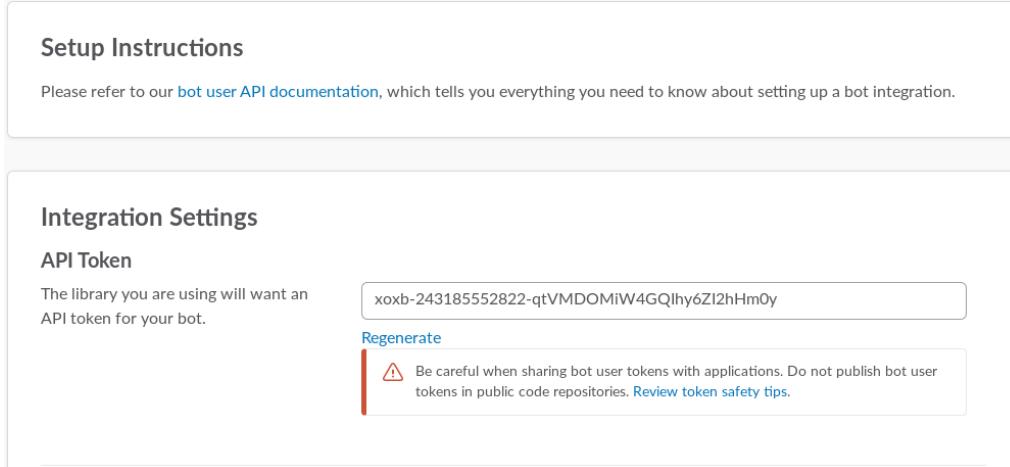


**Figura 9. Criação do espaço de trabalho**



**Figura 10. Criação do *Bot***

Feito isso o Slack irá gerar um API *token*, Figura 11, que serve de autenticação para o uso da API de conexão *slackclient*, portanto guarde essa informação para uso futuro.



**Figura 11. API token**

### 2.3. Manipulando o cliente Python

Tendo clonado o repositório [Moura 2017b] existem algumas alterações de código que devem ser feitas para a adaptação do cliente. Como pode exemplo no trecho abaixo, deve-se colocar o API *token* gerado pelo Slack(mostrado na Figura 11).

```
slack_client = SlackClient("your-api-token-here")
```

Outra modificação a ser feita é colocar o nome escolhido para o *Bot* no trecho abaixo:

```
user_list = slack_client.api_call("users.list")
for user in user_list.get('members'):
    if user.get('name') == "your-bot-name-here":
        slack_user_id = user.get('id')
        break
```

Feito isso, agora é hora de personalizar o cliente, como por exemplo modificar os pinos GPIO a serem utilizados:

```
#GPIO SETUP
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(8,GPIO.OUT)
```

E criar novos métodos utilizando estes pinos declarados. No repositório já estão contidos alguns exemplos de métodos(lightOn, LightOff, fanOn, fanOff, alarmOn, alarmOff e alguns métodos que vem com a biblioteca *psutil*), porém o administrador que irá configurar este cliente tem total liberdade na criação dos métodos para manipular os pinos.

Após a criação dos métodos personalizados, agora deve-se vinculá-los às sentenças que o *Bot* reconhecerá, para isso é feito o uso da biblioteca *re*(*regular expression*) que compara *Strings* padrão, como pode ser visto no trecho de código abaixo:

```
if re.match(r'*(light_off)*', message_text, re.IGNORECASE):
    // words you want your bot to recognize
    lightOff() // the gpio method to execute
    slack_client.api_call(
        "chat.postMessage",
        channel=message['channel'],
        text="Light_off!",
        // bot response
        as_user=True)
```

Neste código o *Bot* reconhecerá uma sentença que contém *light off* em qualquer parte e então executará *lightOff()*.

Por fim, tendo o código personalizado e o RPI configurado, resta apenas fazer com que o cliente Python seja executado assim que o RPI ligue e em segundo plano, para isso utiliza-se as instruções dadas pelo fabricante no link: [RaspberryPi 2017b]. Basta editar o arquivo ”rc.local” e adicionar o comando que executa o cliente, como é mostrado no código abaixo:

```
$ sudo nano /etc/rc.local
```

Deve-se adicionar a linha abaixo ao final do arquivo:

```
python3 /home/pi/slackbot_iot/slackbot_iot.py &
```

O diretório será diferente, dependendo de onde esteja salvo o cliente Python no RPI. Dessa forma, o cliente será executado, em segundo plano, no momento da inicialização do Raspberry Pi.

### 3. Resultados

A disposição física do RPI e dos dispositivos pode ser vista na Figura 12 abaixo.

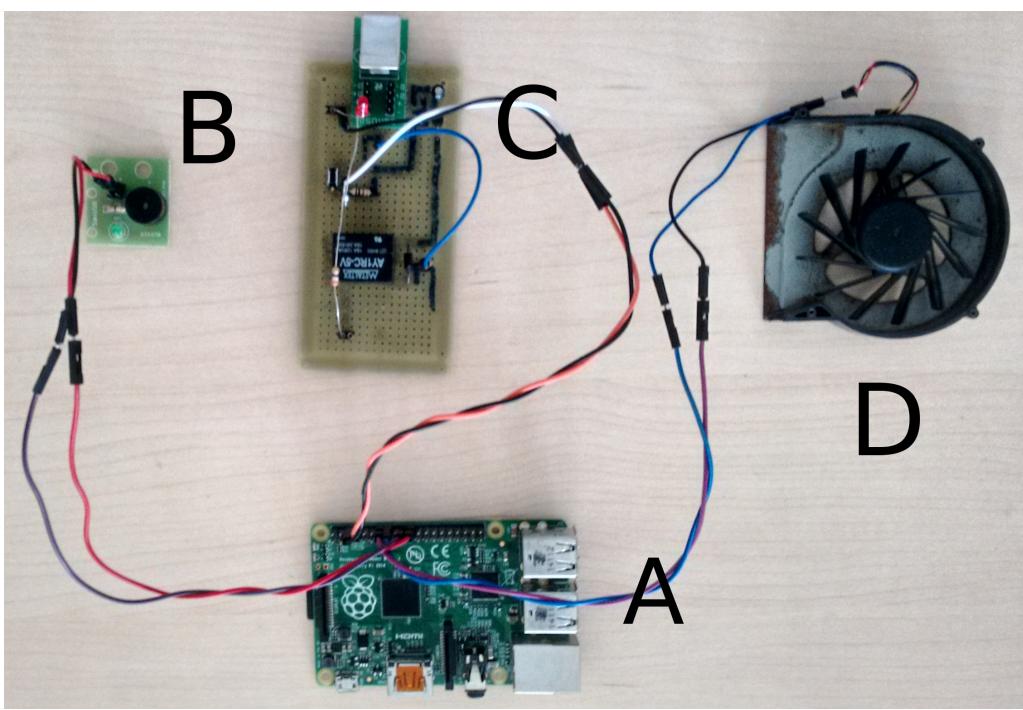
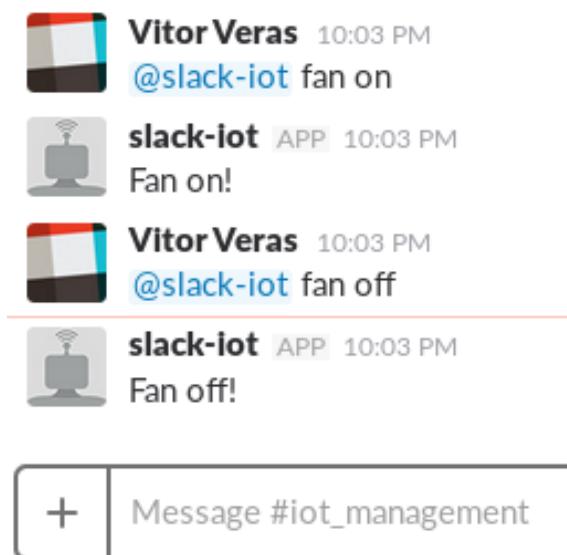


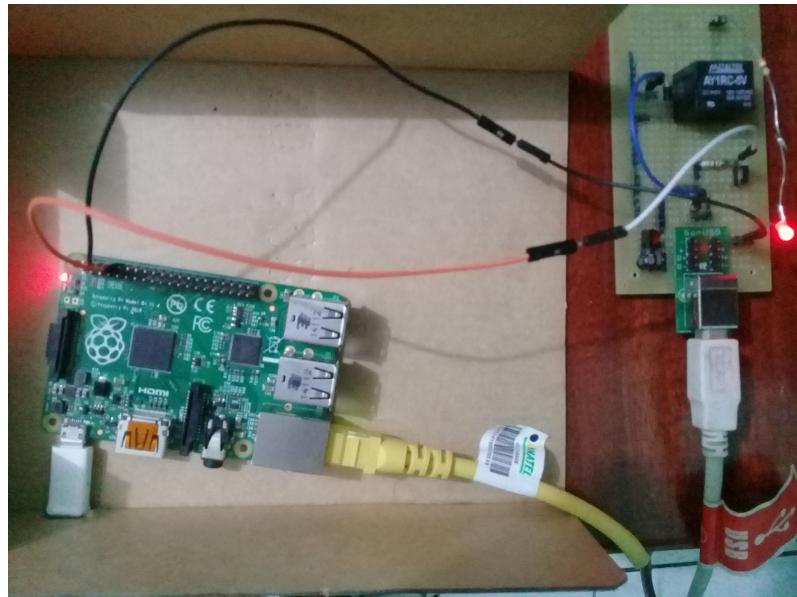
Figura 12. Esquema físico do projeto

Em (A) temos o Raspberry Pi modelo 1 B+ onde estão sendo usados os pinos 8, 16 e 22 de GPIO e 6, 14 e 20 de GROUND. Em (B) temos um circuito com uma campainha que será usado como alarme. Já (C) representa a placa de relé com circuito segundo a Figura 1 que representará uma lâmpada. E por fim em (D) temos um *cooler* reaproveitado que representa um ventilador. Todos os dispositivos necessitaram de dois *jumpers*, um para a ligação com o pino GPIO e outro para o GROUND.

Um vídeo com o completo funcionamento do projeto pode ser visto em [Moura 2017c] e fotos da resposta do *Bot* e do projeto acionado podem ser vistas nas Figuras 13 e 14 respectivamente.



**Figura 13. Comunicação com o Bot**



**Figura 14. Foto do projeto acionado**

#### 4. Considerações finais

Internet das coisas ou *Iot* é um mercado de tendência mundial já que cada vez mais temos demanda por um mundo conectado até nos mais ínfimos dispositivos. Neste projeto foram abordados os tópicos de eletrônica na placa de acionamento de relé juntamente com conceitos de *Iot* na integração do Raspberry Pi com a plataforma *Web* do Slack. Um projeto simples, escalonável e de baixo custo que pode ser uma ótima solução para automatizar o seu ambiente de trabalho em grupo.

## Referências

- etcher (2017). [GitHub] /Etcher. <https://github.com/resin-io/etcher>. [Online; acessado em 19 de setembro de 2017].
- Forbes (2014). A simple explanation of 'the internet of things. <https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#510487981d09>. [Online; acessado em 19 de setembro de 2017].
- Meola, A. (2016). What is the internet of things (iot)? <http://www.businessinsider.com/what-is-the-internet-of-things-definition-2016-8>. [Online; acessado em 19 de setembro de 2017].
- Moura, V. V. (2017a). [GitHub]vitor-veras/relay-board. <https://github.com/slackapi/python-slackclient>. [Online; acessado em 19 de setembro de 2017].
- Moura, V. V. (2017b). [GitHub]vitor-veras/slackbot-iot. <https://github.com/vitor-veras/slackbot-iot>. [Online; acessado em 17 de setembro de 2017].
- Moura, V. V. (2017c). [Youtube]Slack Bot with Iot. <https://youtu.be/oNQft1fzo-o>. [Online; acessado em 20 de setembro de 2017].
- RaspberryPi (2017a). Gpio: Models a+, b+, raspberry pi 2 b and raspberry pi 3 b. <https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/>. [Online; acessado em 19 de setembro de 2017].
- RaspberryPi (2017b). Ssh (secure shell). <https://www.raspberrypi.org/documentation/remote-access/ssh/>. [Online; acessado em 18 de setembro de 2017].
- RaspberryPi (2017c). Where work happens. <https://www.raspberrypi.org/downloads/>. [Online; acessado em 19 de setembro de 2017].
- Slack (2017a). Create a new bot. <https://my.slack.com/services/new/bot>. [Online; acessado em 19 de setembro de 2017].
- Slack (2017b). [github]slackapi/python-slackclient. <https://github.com/slackapi/python-slackclient>. [Online; acessado em 19 de setembro de 2017].
- Slack (2017c). Where work happens. <https://slack.com/>. [Online; acessado em 19 de setembro de 2017].