

Aquisição de Dados de GPS e Raspberry PI 3B+ via Wi-Fi e Servidor Labrador V2

Autor(es) Professor(es)

Luís Henrique do Nascimento Silva, luiz.silva1340@etec.sp.gov.br

Matheus de Souza Melo da Silva, matheus.silva2073@etec.sp.gov.br

Nicholas Eiti Dan, nicholas.dan@etec.sp.gov.br

Bruno Medina Pedroso, brunomedina@etec.sp.gov.br

1. Introdução

Este roteiro consiste no envio de dados de posicionamento via internet a partir de uma Raspberry Pi para uma base de dados em Mariadb em um Labrador V2 da Caninos Loucos, utilizando assim, as técnicas de IoT. A partir disto, um computador ou celular conectado a rede local, deve ser capaz de observar as informações enviadas por meio de uma página em PHP, que irá imprimir os dados enviados para a base de dados.

Visto o conceito de IoT (*Internet of Things* ou Internet das Coisas), ele se define como uma tecnologia emergente na qual diversos dispositivos, sensores, softwares e afins são todos conectados a partir da internet, assim possibilitando um controle e aquisição de dados mais inteligentes nas aplicações que a utiliza. Tanto é o interesse por este setor que estima-se que o mercado de IoT seja tão grande quanto 11 trilhões de dólares em 2025 [1].

Para a aquisição e envio de coordenadas, utiliza-se no projeto uma Raspberry Pi 3B+, como mencionado anteriormente. A Raspberry Pi é uma série de computadores de placa única, que como o nome sugere, é formado por somente uma placa. A empresa tem por finalidade criar computadores baratos e pequenos, assim democratizando o uso de computadores para a população [2]. Visto que esta aplicação será utilizada em um drone, foi escolhido este produto devido a sua acessibilidade, recursos e tamanho. Como a placa não possui um GPS próprio, utiliza-se no projeto um módulo de GPS NEO-6M para a geolocalização.

Para guardar o webserver e a base de dados, utiliza-se o computador de placa única Labrador V2, inspirado na linha de computadores Raspberry Pi. A Labrador é uma placa nacional feita pelo projeto Caninos Loucos da USP com o objetivo de trazer ao

Brasil computadores acessíveis e incentivar a aprendizagem de eletrônica nacional [3].

2. Recursos

Abaixo estão descritos todos os recursos a serem utilizados pelo projeto

2.1. Hardware

Qtda	Descrição
01	Raspberry Pi 3B+
01	Labrador V2
02	Cartão microSD de pelo menos 8GB (usados para baixar os sistemas na Raspberry e no Labrador; podem não ser necessários caso já instalados)
01	Mouse
01	Teclado
01	Monitor
01	GPS NEO-6M
03	Jumpers – Coloridos

2.2. Software

Estarão detalhados abaixo um breve resumo de cada um dos principais serviços, sistemas operacionais e bibliotecas utilizadas ao longo do projeto. Este segmento tem como objetivo apenas listar estes componentes, sendo eles naturalmente retratados ao longo deste documento.

Sistemas operacionais	Descrição
Raspbian	Sistema utilizado e desenvolvido pela Raspberry baseado em Debian
Debian	Distribuição Linux utilizada na Labrador; é uma das principais distribuições em Linux

Linguagens de programação	
Python	Linguagem de programação popular e de fácil aprendizagem; a Raspberry foi feita especialmente para utilizar Python
PHP	Linguagem de script embutida em HTML utilizada em <i>backends</i> de webserver (o script é processado no próprio server e não no cliente)
HTML	Linguagem de estrutura de sites muito utilizada no desenvolvimento web
Serviços	
Apache2	Serviço em Debian responsável por oferecer serviços de um servidor HTTP, assim abrindo uma página web na rede
MariaDB	Database baseada em MySQL
Adminer	Serviço de gerenciamento de base de dados
Bibliotecas	
pynmea2	Biblioteca que interpreta sentenças com o protocolo nmea2 (para geoposicionamento)
requests	Biblioteca que trata o envio e recebimento de dados por meio do protocolo HTTP

3. Desenvolvimento

Abaixo estão descritas e separadas as etapas de desenvolvimento do projeto, adicionalmente, serão apresentados com mais detalhes as linguagens e os sistemas que cada parte do projeto utiliza.

3.1. Sensor(es)

Abaixo estão descritos os sensores utilizados no projeto.

3.1.1. GPS NEO-6M



Fig. 1. Foto do módulo de GPS NEO-6M

Fonte: Electrogate. <https://www.electrogate.com/modulo-gps-neo-6m-com-antena>

O módulo de GPS NEO-6M, obtém sua localização junto com outras informações relevantes como a aceleração, horário e satélites acessíveis. O protocolo utilizado para enviar essas informações é o NMEA-0183 em uma taxa de transmissão (*baud rate*) de 9600Hz. Este sensor possui 4 pinos, 2 para alimentação (VCC e GND) e 2 para comunicação serial (RX e TX) [4].

O módulo se utiliza de diversos tipos de sentenças diferentes para enviar suas informações, e para o desenvolvimento do projeto, utiliza-se da sentença RMC (*Recommended minimum specific GPS/Transit data*), a qual envia, dentre outras informações, a latitude e longitude [5].

3.2. Plataforma de Desenvolvimento

Abaixo estão descritas as em detalhes as etapas na conexão entre o módulo de GPS e a Raspberry.

3.2.1. Prototipagem

Com a Raspberry desligada, realize a seguinte ligação com o módulo de GPS:

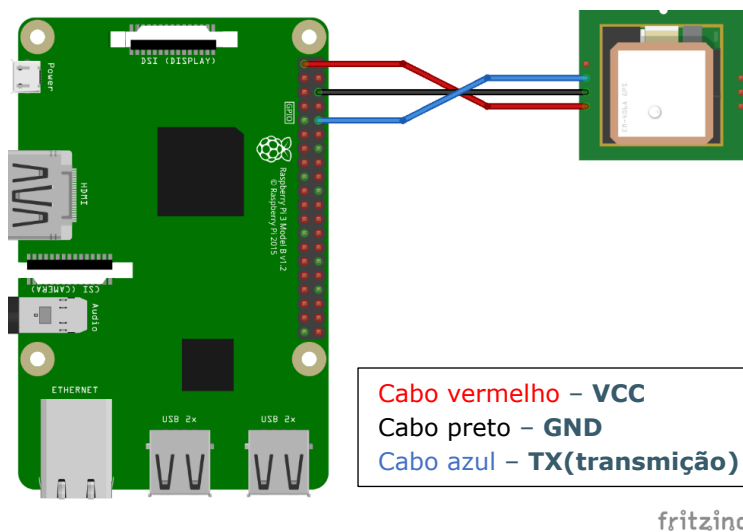


Fig. 2. Esquema gráfico da conexão entre o módulo de GPS e a Raspberry Pi
Diagrama realizado pelo software fritzing

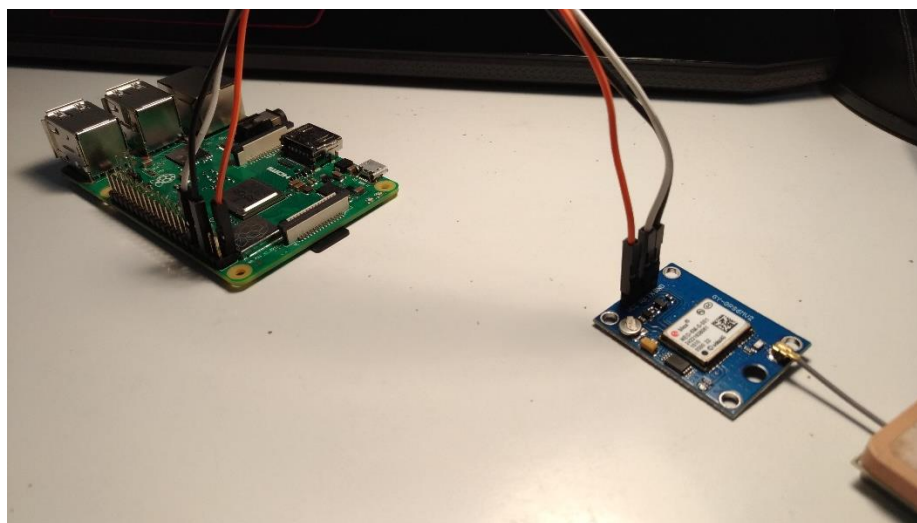


Fig. 3. Foto da conexão entre o módulo de GPS e a Raspberry Pi

Antes de prosseguir com a prototipagem da Raspberry, caso o sistema operacional ainda não esteja instalado, vá até o item 4.1. e siga os passos de instalação. Como recomendação padrão após a instalação da OS (Sistema Operacional ou *Operating System*) em um computador Linux, é visado atualizar a versão da OS caso ainda não realizado antes de começar a instalar novos pacotes. Considerando isso, insira os seguintes comandos no terminal do Raspberry:

```
$ sudo apt update -y
```

```
$ sudo apt upgrade -y
```

Após o término da atualização, reinicie sua máquina para que a instalação seja efetivamente realizada. Em seguida, será necessário editar algumas configurações no arquivo `/boot/config.txt` para o uso do módulo de GPS, digite no terminal:

```
$ sudo nano /boot/config.txt
```

No final do arquivo insira as linhas:

```
# ativa a comunicação SPI (Serial Peripheral Interface)
dtparam=spi=on
# desabilita o bluetooth
dtoverlay=disable-bt
# define a frequência da GPU
core_freq=250
# ativa a comunicação UART (Universal Asynchronous Receiver Transmitter)
enable_uart=1
# força modo turbo mesmo com pouca atividade
force_turbo=1
```

Para salvar as alterações tecle `Ctrl+O` e tecle `Ctrl+X` para sair. Após esta configuração, é necessário a desativação da comunicação UART como console serial. Para tanto, insira no terminal:

```
$ sudo nano /boot/cmdline.txt
```

Substitua o conteúdo no arquivo pelo conteúdo abaixo:

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4
elevator=deadline fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-
consoles
```

Para salvar as alterações tecle `Ctrl+O` e tecle `Ctrl+X` para sair. Após esta etapa, reinicialize a Raspberry novamente.

Por fim, conecte a Raspberry em uma rede forte, visto que tanto a Raspberry

tanto o Labrador terão que estar conectados na mesma rede para se comunicarem.

3.2.2. Codificação

Com a Raspberry ligada e o GPS conectado, coloque o módulo em um espaço a céu aberto ou perto de uma janela e espere até o módulo se conectar a rede GPS, indicada por uma led azul piscante. Quando o módulo estiver piscando, insira no terminal:

```
$ sudo cat /dev/ttyAMA0
```

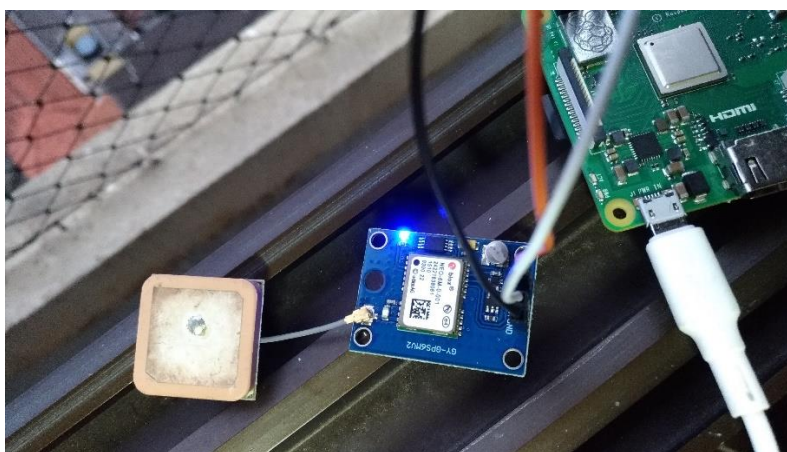


Fig. 4. Foto do LED piscante indicando o recebimento do GPS

Este comando irá imprimir todas a data enviada no porte serial selecionado. Este comando deve imprimir valores como:

<colocar print do terminal>

Para o desenvolvimento do projeto, será necessário a instalação da biblioteca de python pynmea2, que irá transcrever as sentenças recebidas no protocolo nmea2 pela entrada serial da Raspberry, já que, como mencionado anteriormente, este é protocolo utilizado pelo módulo de GPS para comunicação. Também será necessário a instalação da biblioteca requests, que será responsável por enviar as informações adquiridas para o webserver. Detalhes desta biblioteca serão discutidos posteriormente no item 3.4.1. Para tanto, insira no terminal este comando:

```
$ sudo pip install pynmea2
```



```
$ sudo pip install requests
```

Após concluído a instalação das bibliotecas, crie este código em python na Raspberry: (tema de cor quiet light)

gps_test.py

```
"""
Programa Python para enviar coordenadas para um arquivo de PHP

Adaptado de: https://sparklers-the-makers.github.io/blog/robotics/use-neo-6m-
module-with-raspberry-pi/
"""

import serial # biblioteca que cuida de comunicacoes seriais
import time
import string
import pynmea2 # biblioteca que transcreve sentencas em nmea2
import requests # biblioteca que cuida do envio e recebimento de informacoes por
HTTP

# O URL da pagina em que as informacoes serao enviadas
SITE_URL = "http://192.168.15.51/gps_test/pos_sql_send.php"

# funcao que enviara as informacoes para a pagina em PHP
def post_data(data, url):
    hd = {'Content-type': "application/x-www-form-urlencoded"} # define as
caracteristicas da mensagem a ser enviada
    r = requests.post(url, data, headers=hd) # funcao que envia as informacoes
para a pagina

    if r.status_code == requests.codes.ok:
        print("Data posted with success!")
```



```
return True
```

```
print(r.raise_for_status())
```

```
return False
```

```
while True:
```

```
    port="/dev/ttyAMA0"
```

```
    ser=serial.Serial(port, baudrate=9600, timeout=0.5) # inicializa o port a ser lido
```

```
    dataout = pynmea2.NMEAStreamReader()
```

```
    newdata=ser.readline() # le a sentenca serial enviada
```

```
    # verifica a sentenca para se utilizar (é a escolhida?)
```

```
    if newdata[:6] == b"$GPRMC":
```

```
        sdata = newdata.decode('UTF-8') # cria uma variavel com a sentenca em texto
```

```
        newmsg=pynmea2.parse(sdata) # transforma a sentenca (em texto) em um objeto com as variaveis apropriadamente definidas
```

```
        # definicao dos valores das coordenadas
```

```
        lat=str(newmsg.latitude)
```

```
        lng=str(newmsg.longitude)
```

```
        gps_msg = "Latitude=" + lat + ", Longitude=" + lng
```

```
        # objeto a ser transmitido
```

```
        coord = {
```

```
            'latitude':lat,
```

```
            'longitude':lng
```

```
        }
```

```
        post_data(coord, SITE_URL)
```

```
print(gps_msg)
time.sleep(5) # intervalo entre leituras de 5s
```

A parte em azul indica a parte dedicada à conexão entre o GPS e o Raspberry.

3.3. Conectividade

Abaixo estão descritas os detalhes sobre a conexão da Raspberry para o Labrador (webserver).

3.3.1. Prototipagem

Como a Raspberry possui um módulo de conexão wireless LAN, nenhuma mudança no experimento na Raspberry será necessária. Certifique-se que, assim como mencionado anteriormente, a placa esteja em um local com boa conectividade.

3.3.2. Codificação

Abaixo destaca-se desta vez a parte responsável pelo envio das coordenadas para o webserver do código já criado na Raspberry `gps_test.py`.

gps_test.py

```
"""
```

```
Programa Python para enviar coordenadas para um arquivo de PHP
```

```
Adaptado de: https://sparklers-the-makers.github.io/blog/robotics/use-neo-6m-  
module-with-raspberry-pi/
```

```
"""
```

```
import serial # biblioteca que cuida de comunicacoes seriais
```

```
import time
```

```
import string
```

```
import pynmea2 # biblioteca que transcreve sentencas em nmea2
```

```
import requests # biblioteca que cuida do envio e recebimento de informacoes por  
HTTP
```

```
# O URL da pagina em que as informacoes serao enviadas
SITE_URL = "http://192.168.15.51/gps_test/pos_sql_send.php"

# funcao que enviara as informacoes para a pagina em PHP
def post_data(data, url):
    hd = {'Content-type': "application/x-www-form-urlencoded"} # define as
caracteristicas da mensagem a ser enviada
    r = requests.post(url, data, headers=hd) # funcao que envia as informacoes
para a pagina

    if r.status_code == requests.codes.ok:
        print("Data posted with success!")
        return True

    print(r.raise_for_status())
    return False

while True:
    port="/dev/ttyAMA0"
    ser=serial.Serial(port, baudrate=9600, timeout=0.5) # inicializa o port a ser
lido
    dataout = pynmea2.NMEAStreamReader()
    newdata=ser.readline() # le a sentenca serial enviada

    # verifica a sentenca para se utilizar (é a escolhida?)
    if newdata[:6] == b"$GPRMC":
        sdata = newdata.decode('UTF-8') # cria uma variavel com a sentenca em
texto
        newmsg=pynmea2.parse(sdata) # transforma a sentenca (em texto) em um
objeto com as variaveis apropriadamente definidas
```

```
# definicao dos valores das coordenadas
lat=str(newmsg.latitude)
lng=str(newmsg.longitude)

gps_msg = "Latitude=" + lat + ", Longitude=" + lng

# objeto a ser transmitido
coord = {
    'latitude':lat,
    'longitude':lng
}

post_data(coord, SITE_URL)

print(gps_msg)
time.sleep(5) # intervalo entre leituras de 5s
```

As partes em azul indicam as partes responsáveis pela transmissão de dados da Raspberry para o Labrador. A função que efetivamente enviará as informações para o webserver neste código é a função `requests.post()`, que se utiliza o protocolo de comunicação HTTP (muito comum em aplicações web) pelo método POST, a ser detalhado no item 3.4.1.

3.4. Transferência de Dados

Abaixo está o esquema geral do funcionamento do webserver a ser aplicado neste projeto.

Diagrama do Sistema Webserver

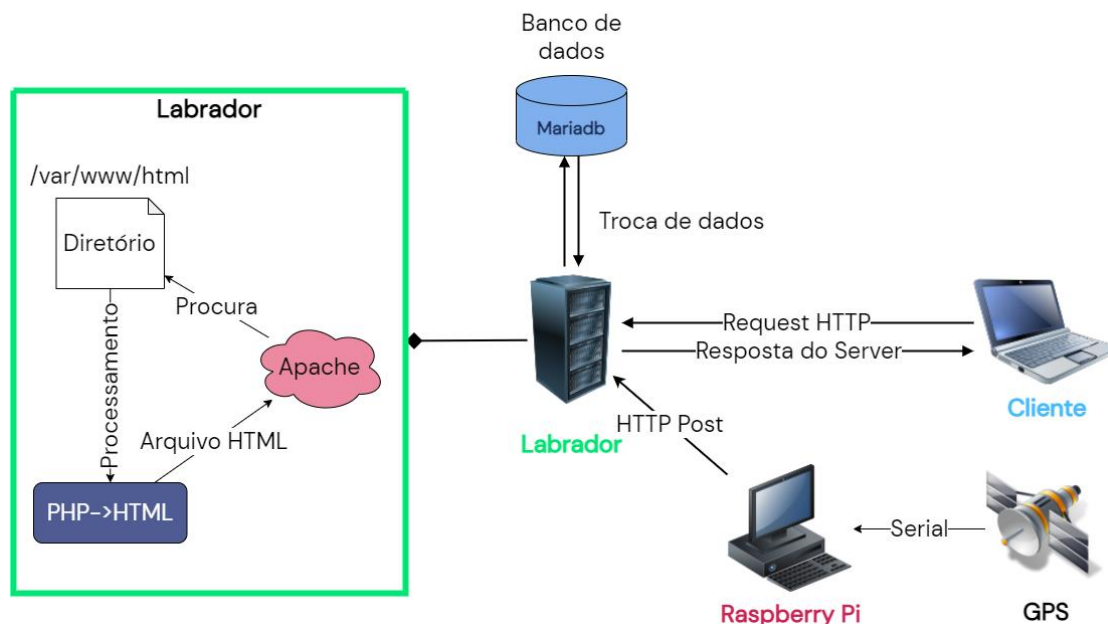


Fig. 6. Diagrama do funcionamento do Webserver

Apache, o serviço utilizado para transformar uma máquina Debian em um webserver, utiliza-se do protocolo HTTP (*Hypertext Transfer Protocol*) para tal. HTTP é um protocolo de nível de aplicação muito comum em aplicações web para realizar interações (requisição e resposta) com o servidor e o cliente. Neste protocolo, quando um cliente realiza uma requisição (*HTTP Request*), o cliente enviará a ele uma resposta (*HTTP Response*) como um arquivo em hipertexto [6] – um tipo de texto não linear com referências a outros textos e com elementos não verbais como gráficos ou imagens, ou seja, uma página web [7].

3.4.1. Transmissão (*Transmitter*)

O envio das coordenadas adquiridas pela Raspberry são enviadas à um arquivo PHP por meio do método de requisição HTTP POST. Este método envia dados para um webserver, sendo normalmente utilizado em envios de formulários online e requiere a localização de envio (link), conteúdo e cabeçalho, que definirá que tipo de conteúdo será transmitido, como tuplas ou JSON [7,8]. As coordenadas são enviadas por meio deste método através da biblioteca requests, que simplifica este processo.

3.4.2. Recepção (Receiver)

A página que receberá as informações enviadas pelo método POST serão recolhidas por meio do array `$_POST` – um array padrão do PHP que guarda todas os dados enviados por HTTP POST [9]. Posteriormente estes dados serão enviados para uma base de dados, abordado com mais detalhes no item 3.5.2.

3.5. Servidor

Caso nunca tenha utilizado o Labrador, vá até o item 4.1. e siga os passos iniciais do Labrador. Caso um sistema operacional não esteja instalado, vá novamente ao item 4.1. e siga os passos de instalação do sistema.

Após iniciar a máquina, atualize seu sistema antes de começar a instalar os pacotes a serem utilizados, para tanto, insira no terminal:

```
$ sudo apt update -y  
$ sudo apt upgrade -y
```

Após a instalação da atualização, reinicie sua máquina.

3.5.1. Serviço

Assim como explicado no item 3.4, o principal serviço a ser utilizado pelo projeto é o Apache2, que retomando, utiliza HTTP (Transferência de Hipertexto ou *Hypertext Transfer Protocol*) para transmitir arquivos de hipertexto locais na rede, ou seja, mostrar uma página web. Apache é um software livre e *open-source* visado na criação de web servers , para saber mais acesse [10].

Para instalar o serviço no Labrador, insira no terminal:

```
$ sudo apt install -y apache2
```

O protocolo HTTP utiliza-se do port 80 para sua comunicação na rede, portanto, para habilitar este porte e permitir o envio de hipertextos e o funcionamento correto do Apache, insira no terminal:

```
$ sudo iptables -A INPUT -p tcp --dport 80 -m conntrack --ctstate NEW,ESTABLISHED  
-j ACCEPT
```

```
$ sudo iptables -A OUTPUT -p tcp --sport 80 -m conntrack --ctstate ESTABLISHED -  
j ACCEPT
```

Este comando se utiliza do serviço de firewall básico iptables, que é uma coletânea de regras para ports, permitindo ou não o envio e recebimento de dados por eles. Após a habilitação do firewall reinicie o serviço de Apache inserindo o comando:

```
$ sudo systemctl restart apache2
```

Após esta etapa, digite em um browser – em um dispositivo conectado na mesma rede conectada no Labrador – o IP do Labrador. Para descobrir o IP, insira o comando:

```
$ ip addr
```

caninos@debian-armhf: ~

```
caninos@debian-armhf:~$ ip addr  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: bond0: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN group default qlen 1000  
    link/ether a6:8b:6b:b6:9f:5f brd ff:ff:ff:ff:ff:ff  
3: eql: <MASTER> mtu 576 qdisc noop state DOWN group default qlen 5  
    link/slip  
4: ifb0: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN group default qlen 32  
    link/ether 46:56:e9:ef:71:96 brd ff:ff:ff:ff:ff:ff  
5: ifb1: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN group default qlen 32  
    link/ether d6:51:ef:74:5e:17 brd ff:ff:ff:ff:ff:ff  
6: tegl0: <NOARP> mtu 1500 qdisc noop state DOWN group default qlen 100  
    link/void  
7: tunl0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000  
    link/ipip 0.0.0.0 brd 0.0.0.0  
8: gre0@NONE: <NOARP> mtu 1476 qdisc noop state DOWN group default qlen 1000  
    link/gre 0.0.0.0 brd 0.0.0.0  
9: gretap0@NONE: <BROADCAST,MULTICAST> mtu 1462 qdisc noop state DOWN group default qlen 1000  
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff  
10: erspan0@NONE: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default qlen 1000  
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff  
11: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group default qlen 1000  
    link/ether 02:e5:fc:fa:77:4b brd ff:ff:ff:ff:ff:ff  
12: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000  
    link/ether 14:6b:9c:7c:06:40 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.15.51/24 brd 192.168.15.255 scope global dynamic wlan0  
        valid_lft 14238sec preferred_lft 14238sec  
    inet6 2804:1b3:a180:23e6:166b:9c:fe7c:640/64 scope global dynamic mngtmpaddr  
        valid_lft 43169sec preferred_lft 43169sec  
    inet6 fe80::166b:9c:fe7c:640/64 scope link  
        valid_lft forever preferred_lft forever  
caninos@debian-armhf:~$
```

Fig. 7. Captura do comando ip addr no terminal do Labrador

Inserindo o IP do Labrador em um browser, deverá ser visto a seguinte página:

<http://ocubo.cpsctec.com.br/>

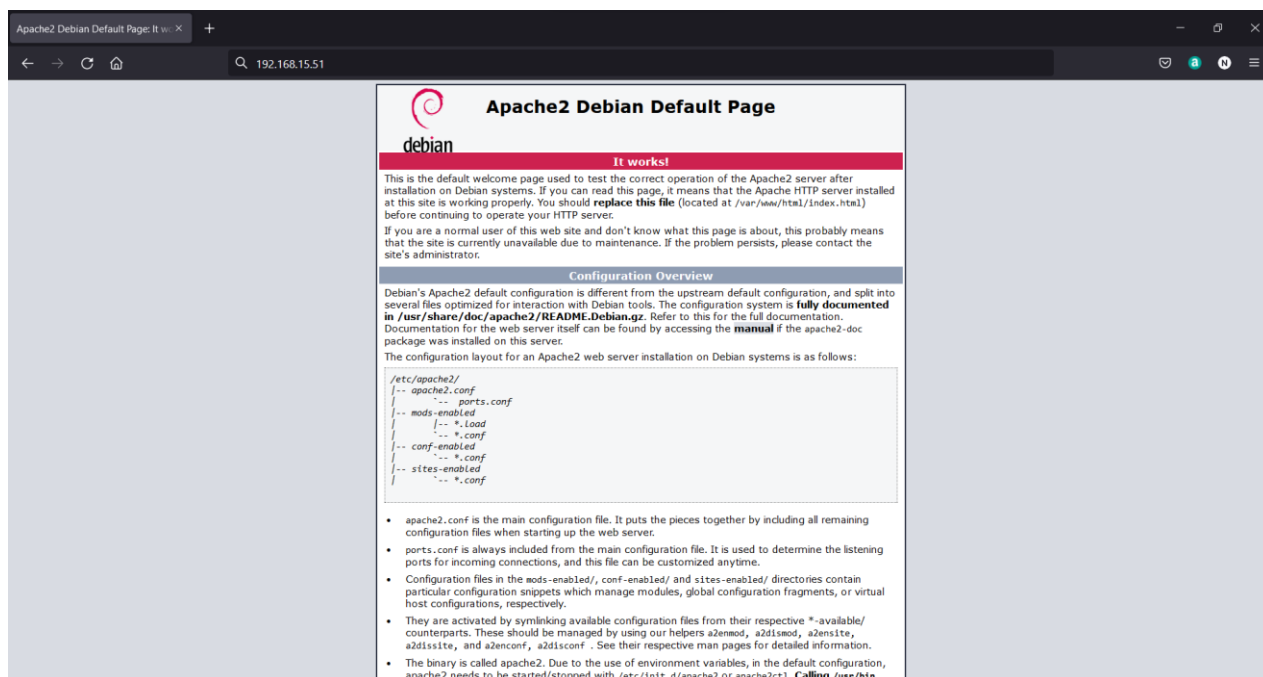


Fig. 8. Página padrão do Apache (index.html)

Após a testagem inicial do Apache, instale os pacotes referentes ao PHP que se utilizará para formatar as páginas do projeto. Visto isso, insira os seguintes comandos:

```
$ sudo apt install -y php libapache2-mod-php php-mysql
```

Com a instalação terminada, adicione ao diretório padrão do Apache (/var/www/html) uma nova pasta para guardar as páginas em PHP deste projeto, para tanto, insira no terminal:

```
$ sudo mkdir /var/www/html/*nome_do_seu_projeto*
```

Ademais, para testar o funcionamento do PHP no Apache, crie nesta pasta um arquivo em PHP inserindo no terminal:

```
$ sudo nano /var/www/html/*nome_do_seu_projeto*/teste.php
```

Insira neste arquivo as seguintes linhas:

teste.php

```
<?php  
echo "Hello World!";
```

<http://ocubo.cpscetec.com.br/>

```
?>
```

Clique Ctrl+O e Ctrl+X em seguida, e depois reinicie o Apache inserindo novamente no terminal:

```
$ sudo systemctl restart apache2
```

Após isso abra a página do webserver em um browser, porém, ao invés de somente inserir o IP, adicione logo depois `/*nome_do_seu_projeto*/teste.php`. Deverá, assim ser visto a seguinte página:

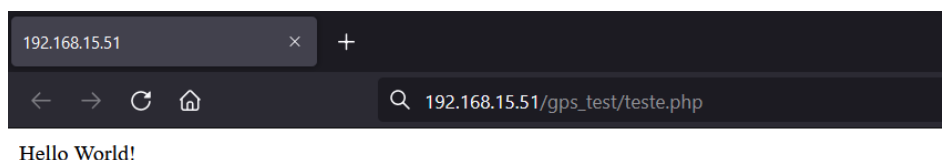


Fig. 9. Exibição de teste.php em um cliente

Posteriormente, para remover este arquivo, insira no terminal:

```
$ sudo rm /var/www/html/*nome_do_seu_projeto*/teste.php
```

O Apache não inicializa junto com o sistema de forma padrão, portanto é recomendado colocar o serviço na lista de inicialização, visto que inicializá-lo cada vez a máquina for reinicializada vira moroso ao passar do tempo. Para tanto, insira no terminal:

```
$ sudo update-rc.d apache2 defaults
```

Após de terminada a instalação do Apache, será instalada os pacotes relacionados a base de dados. Em concordância, insira no terminal:

```
$ sudo apt install -y mariadb-server
```

Com a instalação concluída, ative o seguinte script imbutido ao mariadb:

```
$ sudo mysql_secure_installation
```

Este script garantirá maior segurança ao seu banco de dados. Ele realizará diversos prompts (ações) sobre segurança o qual oferecerá opções para serem escolhidas. Abaixo estão listadas as respostas sugeridas:

1. *Enter current password for root:* está pedindo sua senha root, mas como é sua primeira instalação, apenas pressione **ENTER**
2. *Set root password?:* está perguntando para cadastrar uma nova senha root, porém visto que root já possui outros métodos de autenticação e que tem autoridade o suficiente para quebrar o sistema, digite **N**
3. *Remove anonymous user?:* está pedindo permissão para bloquear o acesso de usuários sem nome e senha, é mais recomendado digitar **Y**
4. *Disallow root login remotely?:* está pedindo permissão para bloquear o acesso remoto do root, como visto acima, root tem muitas habilitações, então digite **Y**
5. *Remote test database and access to it?:* está pedindo permissão para remover a database de teste, visto que isso não afeta a segurança da base, responda à vontade

Após essas configurações, é necessário criar um novo usuário com privilégios de root para poder editar as databases, sendo assim, abra o prompt do mariadb:

```
$ sudo mariadb
```

Com o prompt aberto, insira:

```
MariaDB [(none)]> CREATE USER 'admin'@'localhost' IDENTIFIED BY 'alguma_senha';  
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost' WITH GRANT  
OPTION;  
MariaDB [(none)]> FLUSH PRIVILEGES;
```

Em seguida saia do prompt:

```
MariaDB [(none)]> exit
```

Com a base de dados instalada e configurada, deve-se instalar adminer, um serviço de gerenciamento de base de dados:

```
$ sudo apt install -y adminer
```

Com o adminer instalado, abra no browser o seguinte URL: http://IP_do_server/adminer/, deve ser visto a seguinte página:

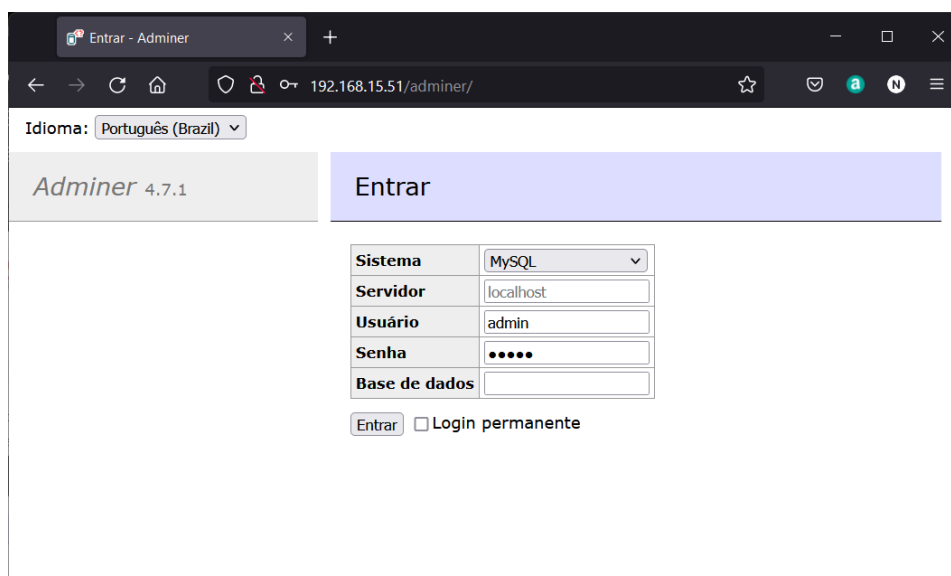


Fig. 10. Interface padrão do adminer

Inserindo seu usuário e senha entre no painel de controle principal do programa. Para criar uma nova base de dados clique no link no canto superior esquerdo. Apenas coloque o nome da sua base de dados e ignore a formatação da base, clique salvar.

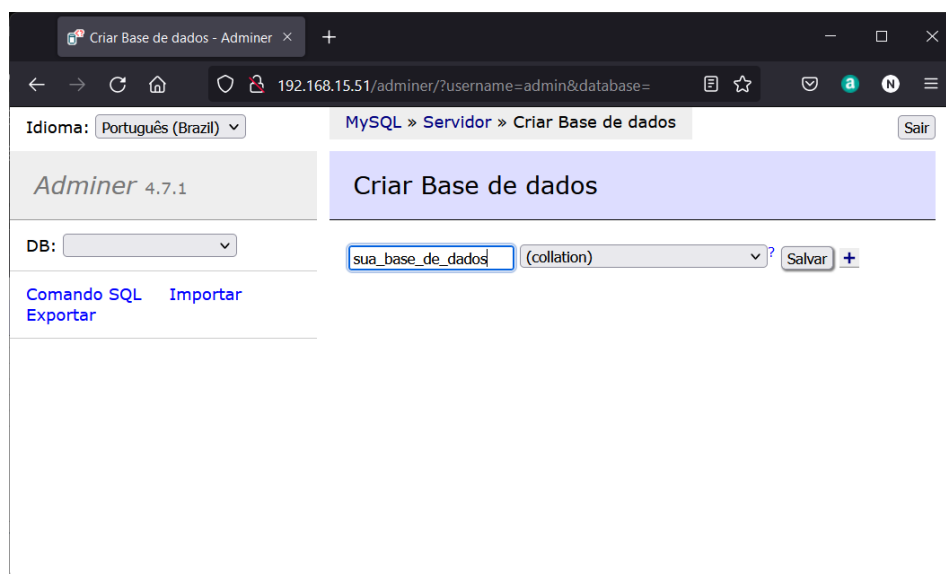


Fig. 11. Interface de criação de uma base de dados

Com a tabela criada, será necessário criar os valores (as colunas da tabela ou *columns*) dentro de uma instância de data, para tal abra o link Comando SQL e insira:

```
CREATE TABLE sua_tabela (  
    `id` int(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,  
    `lat` varchar(20),  
    `long` varchar(20),  
    `time` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP  
)
```

Entre na tabela criada e verifique se todas as colunas foram inseridas corretamente. Sua estrutura deve ficar:

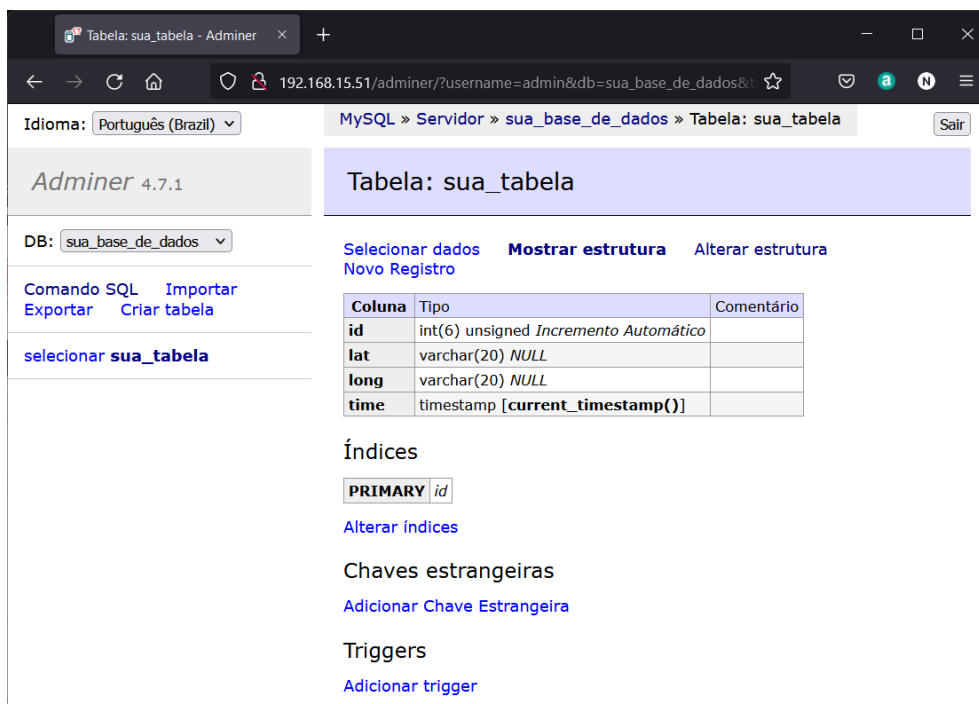


Fig. 13. Estrutura das colunas da tabela criada

3.5.2. Armazenamento de Dados

Assim como mencionado no item 3.4.2., o array `$_POST` será utilizado para recolher os dados enviados. Assim adquiridos, uma conexão com uma base de dados em mariadb é feita e esses dados serão enviados para ele.

Uma base de dados é uma coleção organizada de data armazenadas digitalmente em um computador, ela é composta de diversas tabelas, as quais conterão linhas (*rows*) – uma única instância de data – e colunas (*columns*) – valores que compõem as linhas de data. SQL (Structured Query Language), um termo altamente relacionado à base de dados, é utilizado na aquisição, insesão e manipulação de dados.

Crie este código em PHP no Labrador na pasta do seu projeto para realizar a transferência de dados para a base de dados:

```
pos_sql_send.php

<?php

/*
```

Código em PHP que enviará os dados de recebidos de coordenada para uma tabela em SQL

Fonte: Random Nerd Tutorial (<https://randomnerdtutorials.com/esp32-esp8266-raspberry-pi-lamp-server/>)

*/

```
echo "<p>Esta página enviará os dados do coordenadas para uma tabela em SQL, esta página não deve ser vista pelo usuário comum</p>";
```

```
# Informações sobre a conexão da tabela no SQL
```

```
$servername = "localhost";
```

```
$dbname = "lab_iot";
```

```
$username = "admin";
```

```
$passwd = "admin";
```

```
$lat = $lng = $deglat = $deglng = "";
```

```
# Verificar se chegou informações vindo de outro cliente HTTP (utilizando o método POST)
```

```
if ($_SERVER['REQUEST_METHOD'] == "POST"){
```

```
    # Tratamento dos dados dentro de $_POST
```

```
    $lat = htmlspecialchars($_POST['latitude']);
```

```
    $lng = htmlspecialchars($_POST['longitude']);
```

```
# Cria conexão com a tabela na base de dados
```

```
$conn = new mysqli($servername, $username, $passwd, $dbname);
```

```
# Verifica a integridade da conexão
```

```
if ($conn->connect_error){
```

```
    die("Connection failed: " . $conn->connect_error);
```

```
}
```

```
# Sentença de SQL a ser processada
```



```
$sql = "INSERT INTO gps_info (`lat`,`long`) VALUES ('" . $lat . "', '" .  
$lng . "')";  
  
# Verifica se o processo da sentença foi feita com sucesso  
if ($conn->query($sql) === TRUE){  
    echo "Nova linha inserida com sucesso!";  
}  
else{  
    echo "Error: " . $sql . "<br>" . $conn->error;  
}  
$conn->close();  
  
}  
  
else{  
    echo "No data posted with HTTP POST."  
}  
  
?>
```

Este código se utiliza de sentenças em SQL para enviar os dados para uma tabela. Para saber mais sobre SQL vá até o item 4.1 e veja um dos links sobre o assunto.

3.5.3. Visualização de Dados

Os dados, já enviados para a base de dados, são lidos por um outro arquivo em PHP, interpretados com SQL e assim colocados na página. Pela natureza do PHP (back-end), a página é incapaz de atualizar conforme dados entram na tabela, porém visto que a natureza dos dados é de posicionamento, tal precisão não é necessária. Sendo assim, crie este código em PHP na página do seu projeto:

pos_sql_recieve.php

```
<!DOCTYPE html>  
<html>
```

```
<header>
  <title>Recebimento de dados de GPS em PHP</title>
</header>

<body>

<h1>Lab IoT - ETEC Presidente Vargas</h1>

<h2>Equipe VAR - Aquisição de dados de GPS e Raspberry PI 3B+ via Wi-Fi e
Servidor Labrador V2</h2>

<p>Demo prática do Rodeiro Experimental.</p>

<h3>Introdução</h3>

<p>Neste exemplo, uma Raspberry Pi (mini computador do tamanho de uma placa),
receberá a sua localização por meio de um módulo de GPS (NEO-6M) conectado por
um port serial. Então, o
raspberry enviará, via wifi pelo protocolo HTTP POST, essas informações à um
servidor web sitiado em uma placa Labrador V2 - parte do projeto Caninos Loucos
da USP.</p>

<p>Este site está, sitiado dentro de uma Labrador.</p>

<h3>Função desta página</h3>

<p>Em vista disso, esta página irá ler e imprimir esses valores imprimidos</p>

<?php
/*
```

Código em PHP que irá capturar as informações de uma tabela de SQL e imprimi-las

Fonte: Random Nerd Tutorials (<https://randomnerdtutorials.com/esp32-esp8266-raspberry-pi-lamp-server/>)

```
*/

# Informações sobre a conexão da tabela no SQL
$servername = "localhost";
$dbname = "lab_iot";
$username = "admin";
$password = "admin";

# Cria conexão com a tabela na base de dados
$conn = new mysqli($servername, $username, $password, $dbname);

# Verifica a integridade da conexão
if($conn->connect_error){ die("Connection failed: "
    . $conn->error);
}

# Sentença em SQL
$sql_cmd = "SELECT `id`, `lat`, `long`, `TIME` FROM gps_info";

# Realiza a operação da sentença SQL
if($result = $conn->query($sql_cmd)) {
    while($row = $result->fetch_assoc()){ # Itera por cada item recolhido
        echo "id: " . $row['id'] . "; lat: " . $row['lat'] . "; lng: " .
        $row['long'];
        echo "<br>";
    }
    # Limpa a variável
    $result->free();
}
```

```
}  
  
# Fecha a conexão  
$conn->close();  
  
?>  
</body>  
</html>
```

4. Informações Adicionais

Repositório em Github do projeto (ainda em desenvolvimento):
<https://github.com/NicholasEiti/server-labrador>

4.1. Links de tutoriais

Guia de configuração do sistema do Raspberry Pi

Caso a Raspberry Pi não tenha um cartão SD já configurado para uso, siga um desses links:

- Página oficial do Raspberry Foundation em Português sobre o setup inicial do Raspberry: <https://projects.raspberrypi.org/pt-BR/projects/raspberry-pi-setting-up>
- Vídeo-tutorial do canal Curso em Vídeo sobre a instalação do sistema Raspbian: <https://youtu.be/ANfrmttyKO2w> (parte da playlist sobre primeiros passos: https://youtube.com/playlist?list=PLHz_AreHm4dnGZ_nudmN4rvyLk2fHFRzy)

Guia de configuração do sistema do Labrador V2

Siga o guia abaixo antes de ligar a Labrador:

- Wiki oficial do Caninos Loucos sobre a configuração inicial do sistema: https://wiki.caninosloucos.org/index.php/Começando_com_a_Labrador

Caso o Labrador não tenha um sistema operacional instalado (normalmente o sistema vem de fábrica), siga esse guia:

- Wiki oficial do Caninos Loucos sobre a instalação do sistema: https://wiki.caninosloucos.org/index.php/Instalando_um_sistema_32_bits (nota: se

perceber que o comando `$ sudo ./install-32bits.sh` não funciona, tente executar o comando `$ ls` e verifique o nome do arquivo)

Introdução aos comandos básicos em Linux

Ao longo do desenvolvimento do projeto, será necessário certa manipulação no terminal linux (em Debian OS), portanto, caso não esteja familiarizado com a plataforma, recomenda-se ver um dos conteúdos abaixo:

- Página da Devmedia sobre alguns comandos básicos do Linux: <https://www.devmedia.com.br/comandos-importantes-linux/23893>
- Vídeos-tutorial do canal Curso em Vídeo sobre o terminal do Linux: <https://youtu.be/mgs92GtkQCE> (parte 1) e <https://youtu.be/kK6eBHSxAgA> (parte 2)

Introdução ao HTML

Para a estruturação das páginas se utiliza HTML, portanto, caso queira uma introdução à linguagem, veja um dos links abaixo:

- Página do MDN Web Docs sobre HTML: https://developer.mozilla.org/pt-BR/docs/Learn/HTML/Introduction_to_HTML/Getting_started
- Playlist do canal Curso em Vídeo de introdução ao HTML: https://youtube.com/playlist?list=PLHz_AreHm4dlAnJ_jItV29RFxnPHDuk9o

Introdução ao PHP

Utiliza-se PHP intensamente ao longo do projeto, para tanto, caso queira ver algum conteúdo sobre o assunto, veja um dos links;

- Página do Devmedia sobre PHP: <https://www.devmedia.com.br/php-tutorial/32540>
- Playlist do canal Curso em vídeo de introdução ao PHP: https://youtube.com/playlist?list=PLHz_AreHm4dm4beCCcmW4xwpmLf6EHY9k

Introdução à SQL

Visto que neste guia há a utilização de base de dados, é recomendável que o leitor tenha pelo menos um conhecimento básico sobre SQL. Para tanto, veja um dos links:

- Página do Khan Academy: <https://pt.khanacademy.org/computing/computer-programming/sql>
- Playlist do canal curso em vídeo sobre bancos de dados e MySQL: https://www.youtube.com/playlist?list=PLHz_AreHm4dkBs-795Dsgvau_ekxg8g1r

4.2. Acesso remoto via Windows

Ao longo do projeto, é recomendado possuir o acesso remoto em ambas as plataformas para que não seja necessário a troca dos periféricos constantes para a manipulação de uma delas.

4.2.1. Raspberry Pi

O acesso remoto visual ao Raspberry é recomendado, visto que isto permite a uma experiência completa da placa. Caso seu computador não utilize Windows, realize a conexão via VNC. Para os usuários de Windows, será necessário a instalação dos seguintes pacotes:

```
$ sudo apt install -y xrdp  
$ sudo apt install -y tightvncserver
```

Com a instalação feita, adquira o IP do Raspberry carregando o comando `ip addr` e o insira na aplicação Controlo de Área de Trabalho Remota nativa do Windows.

4.2.2. Labrador

Visto que o Labrador é somente um webserver, é recomendado o acesso remoto via SSH (*Secure Shell*), que lhe dá acesso ao terminal da máquina alvo. Isto não irá limitar seu aproveitamento na máquina, visto que neste guia, utiliza-se somente ele. Instale o pacote OpenSSH-server, que o configurará como um computador capaz de ser acessado via SSH:

```
$ sudo apt install -y openssh-server
```

Depois de instalado o pacote, será necessário abrir o porte 22, já que este é o porte utilizado pelo OpenSSH:

```
$ sudo iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED
```

```
-j ACCEPT  
$ sudo iptables -A OUTPUT -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -  
j ACCEPT
```

Agora com o SSH configurado no Labrador é necessário baixar algum software de cliente de SSH em sua máquina. Caso use um computador Windows recomenda-se o software PuTTY. Link de instalação: <https://www.putty.org/>

4.3. Escopo Futuro

Este guia se trata da simples comunicação entre um arquivo em Python e uma página em PHP em um webserver, não estando preso à qualquer tipo de hardware, assim, qualquer server que aceite PHP e um computador que possa carregar um arquivo .py pode aplicar este projeto, além disso, ele não está preso a um tipo de valor específico, sendo possível o envio de qualquer tipo de informação para o webserver, como: temperatura, dados de questionários, entre outros...

Por se tratar de uma simples aplicação, o aspecto visual das páginas pode ser melhoradas, como melhor GUI e até um mapa interativo mostrando o espaço percorrido pela Raspberry.

Referências

[1] K. Rose, S. Eldridge e L. Chapin, "THE INTERNET OF THINGS: AN OVERVIEW Understanding the Issues and Challenges of a More Connected World" em *Internet Society*, Out. 2015. [Online]. Disponível em: <https://www.internetsociety.org/wp-content/uploads/2017/08/ISOC-IoT-Overview-20151221-en.pdf>

[2] Raspberry Pi Foundation, "What is a Raspberry Pi?" www.raspberrypi.org. <https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/> (acesso em Set. 25, 2022)

[3] Caninos Loucos, "Programa – Caninos Loucos" caninosloucos.org. <https://caninosloucos.org/pt/program-pt/> (acesso em Set. 25, 2022)

[4] u-blox, "NEO-6 u-blox 6 GPS Modules Data Sheet", GPS.G6-HW-09005-E datasheet, Ago. 31, 2009 [Revisado Dez. 05, 2011]

[5] E. S. Raymond, "NMEA Revealed" [gpsd.gitlab.io.
https://gpsd.gitlab.io/gpsd/NMEA.html](https://gpsd.gitlab.io/gpsd/NMEA.html) (acesso em Set. 25, 2022)

[6] *HTTP Semantics*, RFC 9110, Internet Engineering Task Force (IETF), Jun. 2022. [Online]. Disponível em: <https://httpwg.org/specs/rfc9110.html>

[7] World Wide Web Consortium (W3C), "What is HyperText", [www.w3.org
https://www.w3.org/WhatIs.html](https://www.w3.org/WhatIs.html) (acesso em Out. 8, 2022)

[8] MDN Web Docs, "POST - HTTP" [developer.mozilla.org
https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST) (acesso em Out. 4, 2022)

[9] The PHP Documentation Group. PHP Manual. (2022). Acesso em: Out. 9, 2022. [Online]. Disponível em: <https://www.php.net/manual/en/>

[10] The Apache Software Foundation, "Welcome!" [httpd.apache.org
https://httpd.apache.org/](https://httpd.apache.org/) (acesso em Set. 25, 2022)

[11] Oracle, "What Is a Database?" [www.oracle.com
https://www.oracle.com/database/what-is-database/](https://www.oracle.com/database/what-is-database/) (acesso em Out. 11, 2022)