



**UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO - UFERSA  
DEPARTAMENTO DE ENGENHARIAS - DE  
CENTRO MULTIDISCIPLINAR DE CARAÚBAS - CMC  
BACHARELADO EM ENGENHARIA ELÉTRICA**

**JOEL DO AMARAL GURGEL NETO**

**SISTEMA PARA MONITORAMENTO REMOTO DE  
RESERVATÓRIO DE ÁGUA PARA RESIDÊNCIA UNIFAMILIAR**

**CARAÚBAS - RN**

**2021**



**UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO - UFERSA**  
**DEPARTAMENTO DE ENGENHARIAS - DE**  
**CENTRO MULTIDISCIPLINAR DE CARAÚBAS - CMC**  
**BACHARELADO EM ENGENHARIA ELÉTRICA**

**JOEL DO AMARAL GURGEL NETO**

**SISTEMA PARA MONITORAMENTO REMOTO DE**  
**RESERVATÓRIO DE ÁGUA PARA RESIDÊNCIA UNIFAMILIAR**

Trabalho de Conclusão de Curso submetido  
à Universidade Federal Rural do Semi-Árido  
como parte dos requisitos necessários para a ob-  
tenção do Grau de Engenheiro Eletricista.

Orientador: Prof. Dr. Walber Medeiros Lima

**CARAÚBAS - RN**  
**2021**

© Todos os direitos estão reservados a Universidade Federal Rural do Semi-Árido. O conteúdo desta obra é de inteira responsabilidade do (a) autor (a), sendo o mesmo, passível de sanções administrativas ou penais, caso sejam infringidas as leis que regulamentam a Propriedade Intelectual, respectivamente, Patentes: Lei nº 9.279/1996 e Direitos Autorais: Lei nº 9.610/1998. O conteúdo desta obra tomar-se-á de domínio público após a data de defesa e homologação da sua respectiva ata. A mesma poderá servir de base literária para novas pesquisas, desde que a obra e seu (a) respectivo (a) autor (a) sejam devidamente citados e mencionados os seus créditos bibliográficos.

N469s    Neto, Joel do Amaral Gurgel.  
              SISTEMA PARA MONITORAMENTO REMOTO DE  
RESERVATÓRIO DE ÁGUA PARA RESIDÊNCIA UNIFAMILIAR /  
Joel do Amaral Gurgel Neto. - 2021.  
              77 f. : il.

              Orientador: Walber Medeiros Lima.  
              Monografia (graduação) - Universidade Federal  
Rural do Semi-árido, Curso de Engenharia  
Elétrica, 2021.

              1. Microprocessador. 2. Python. 3. HTML. 4.  
Monitoramento. I. Medeiros Lima, Walber, orient.  
II. Título.

O serviço de Geração Automática de Ficha Catalográfica para Trabalhos de Conclusão de Curso (TCC's) foi desenvolvido pelo Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (USP) e gentilmente cedido para o Sistema de Bibliotecas da Universidade Federal Rural do Semi-Árido (SISBI-UFERSA), sendo customizado pela Superintendência de Tecnologia da Informação e Comunicação (SUTIC) sob orientação dos bibliotecários da instituição para ser adaptado às necessidades dos alunos dos Cursos de Graduação e Programas de Pós-Graduação da Universidade.

**JOEL DO AMARAL GURGEL NETO**

**SISTEMA PARA MONITORAMENTO REMOTO DE  
RESERVATÓRIO DE ÁGUA PARA RESIDÊNCIA  
UNIFAMILIAR**

Trabalho de Conclusão de Curso submetido  
à Universidade Federal Rural do Semi-Árido  
como parte dos requisitos necessários para a ob-  
tenção do Grau de Engenheiro Eletricista.

Defendida em: 31 de maio de 2021.

**Banca Examinadora**

**WALBER MEDEIROS**  
**LIMA:02843981441**  
Assinado de forma digital por  
WALBER MEDEIROS  
LIMA:02843981441  
Dados: 2021.06.08 10:33:07 -03'00'

---

**Prof. Dr. Walber Medeiros Lima**  
Prof. Dr. (UFERSA)  
Orientador

**FRANCISCO CARLOS**  
**GURGEL DA SILVA**  
**SEGUNDO:06062106444**  
Assinado de forma digital por  
FRANCISCO CARLOS GURGEL DA  
SILVA SEGUNDO:06062106444  
Dados: 2021.06.08 13:35:04 -03'00'

---

**Francisco Carlos Gurgel da Silva Segundo**  
Prof. Dr. (UFERSA)  
Membro da Banca

**CECILIO MARTINS DE**  
**SOUSA**  
**NETO:06286737448**  
Assinado de forma digital por  
CECILIO MARTINS DE SOUSA  
NETO:06286737448  
Dados: 2021.06.08 13:56:15 -03'00'

---

**Cecílio Martins de Sousa Neto**  
Prof. Dr. (UFERSA)  
Membro da Banca

**CARAÚBAS - RN**  
**2021**



*Este trabalho é dedicado às crianças adultas  
que, quando pequenas, sonharam  
em se tornar Engenheiros.*

# AGRADECIMENTOS

*Primeiramente agradeço à Deus, que em sua infinita sabedoria colocou força em meu coração para vencer essa grande etapa de minha vida com coragem e garra. A fé no Senhor, sem dúvidas, foi quem me fez chegar até aqui.*

*Gostaria de agradecer a toda minha família em ter contribuído na formação, a minha mãe Iris Martins, ao meu pai Ed Gurgel, por sempre estar do meu lado em minhas escolhas. Agradeço ao meu irmão mais novo Igor Gurgel por propulsionar muitas felicidades e risos diariamente e também aos meus outros irmãos Ed Junior, Eduardo e Elder Gurgel, por me ajudar nessa conquista.*

*Agradeço a minha companheira Daiara que me ajudou bastante durante todo o curso, sempre estando ao meu lado, fazendo não desistir das batalhas, me deixando mais forte a cada dia.*

*Em especial à minha vovó Salete que sempre me ajudou e que lutou junto comigo, sempre foi o meu maior exemplo de luta e determinação nessa vida.*

*Agradeço a professor e orientador Prof. Dr. Walber Medeiros, grande orientador que me ajudou muito durante todo curso, tendo uma enorme paciência na elaboração desse trabalho, agradeço por sua confiança e incansável dedicação.*

*Agradeço aos professores participantes da banca examinadora que dividiram comigo este momento tão importante e esperado: Prof. Dr. Walber Medeiros, Prof. Dr. Francisco Carlos Gurgel da Silva Segundo e Prof. Dr. Cecílio Martins de Sousa Neto.*

*Agradeço também aos grandes amigos que a universidade me deu para vida, Lorano, Esdras, Paulo Neto, Mateus Praxedes, José Hugo, Gessica, Alefjohn, Lucas Hanibal, Luan, Jairo, Anderson, Francisco Jonatham, Yago e Moises, que nunca negaram um apoio durante minha trajetória acadêmica.*

*Agradeço todos os membros da equipe Acceptor Aerodesign, e aos orientadores que passaram pelo projeto Wendell Albano, Rafael Luz e Filipe Marques e aos Professores Jackson de Brito, Daniel Freitas e Guymmann Clay por me orientar em outros projetos durante a graduação.*

*“A persistência é muito importante.  
Você não deve desistir, a menos  
que seja forçado a desistir.”  
(Elon Musk)*

## RESUMO

Este trabalho tem como objeto a construção de um sistema de automação residencial e o monitoramento do consumo de água de uma residência em tempo real, a fim de gerar um maior conforto para o usuário e economia no consumo e desperdício de água. Nesse sistema de automação residencial, o cliente não precisará se preocupar em ligar a válvula para encher a caixa d'água ou desligá-la quando ela estiver transbordando, além disso, será possível realizar em tempo real o monitoramento do nível e consumo da caixa d'água fazendo com que os moradores da residência reduzam o consumo quando estiverem com um nível mais baixo, prevenindo contra uma possível falta de água. Para esse sistema foi utilizado o computador *Raspberry Pi model B+* responsável pelo controle do nível da caixa d'água conforme o consumo da residência, além disso, todos os dados serão enviados de forma automática para um banco de dados na nuvem, que através de página *web* desenvolvido em HTML o usuário poderá visualizar todo o monitoramento pelo computador ou celular. Para melhor notificar o usuário, teremos que toda vez quê uma das válvulas de controle mudar o seu estado lógico, o proprietário da residência receberá uma mensagem no seu *e-mail* avisando o acontecimento. Os resultados do trabalho propõem uma alternativa para automação do sistema hidráulico de uma residência, visando uma economia no consumo de água, através da implementação de um sistema de monitoramento e controle d'água de uma residência.

**Palavras-chave:** Microprocessador. *Python*. HTML. Monitoriamento.

## ABSTRACT

This work aims at building a home automation system and monitoring the water consumption of a residence in real time, in order to generate greater comfort for the user and savings in water consumption and waste. In this home automation system, the customer will not have to worry about turning on the valve to fill the water tank or turning it off when it is overflowing, in addition, it will be possible to perform real-time monitoring of the level and consumption of the water tank. 'water causing the residents of the residence to reduce consumption when they are at a lower level, preventing a possible lack of water. For this system, the computer *Raspberry Pi model B* + responsible for controlling the water tank level according to household consumption was used, in addition, all data will be sent automatically to a database in the cloud, that through a web page developed in HTML, the user will be able to view all monitoring by computer or cell phone. In order to better notify the user, every time one of the control valves changes its logical state, the owner of the residence will receive a message in his *e-mail* advising the event. The results of the work propose an alternative for automation of the hydraulic system of a residence, aiming at an economy in the consumption of water, through the implementation of a system of monitoring and control of water in a residence.

**Keywords:** Microprocessor. Python. HTML. Monitoring.

## LISTA DE ILUSTRAÇÕES

|                                                                                                      |    |
|------------------------------------------------------------------------------------------------------|----|
| Figura 1 – Distribuição do consumo de água residencial . . . . .                                     | 16 |
| Figura 2 – Comunicação <i>Internet-of-Things</i> . . . . .                                           | 20 |
| Figura 3 – Sensor ultrassônico HC-SR04. . . . .                                                      | 22 |
| Figura 4 – Sensor ultrassônico JSN-SR04T. . . . .                                                    | 23 |
| Figura 5 – Diagrama de tempo do HC-SR04. . . . .                                                     | 24 |
| Figura 6 – Efeito <i>Hall</i> . . . . .                                                              | 25 |
| Figura 7 – Sensor de vazão modelo YF S201. . . . .                                                   | 26 |
| Figura 8 – Exemplos de Atuadores do mercado. . . . .                                                 | 27 |
| Figura 9 – Princípio de funcionamento do relé eletromecânico. . . . .                                | 28 |
| Figura 10 – Módulo relé de 2 Canais. . . . .                                                         | 28 |
| Figura 11 – Válvula solenoide. . . . .                                                               | 29 |
| Figura 12 – Primeira versão do <i>Raspberry Pi</i> , 1 modelo A. . . . .                             | 31 |
| Figura 13 – <i>Raspberry Pi 3 model B+</i> . . . . .                                                 | 32 |
| Figura 14 – Diagrama de pinagem do <i>Raspberry Pi 3 model B+</i> . . . . .                          | 33 |
| Figura 15 – Área de trabalho do Sistema Operacional <i>Raspbian</i> . . . . .                        | 34 |
| Figura 16 – Modelo do <i>ThingSpeak</i> . . . . .                                                    | 36 |
| Figura 17 – Representação do sistema de automação. . . . .                                           | 38 |
| Figura 18 – Planta da caixa d'água da residência. . . . .                                            | 39 |
| Figura 19 – <i>Raspberry Pi</i> com seu sistema operacional instalado . . . . .                      | 41 |
| Figura 20 – <i>Raspberry Pi</i> com alguns periféricos conectados. . . . .                           | 42 |
| Figura 21 – Circuito para acender um LED desenvolvido no <i>Software Fritzing</i> . . . . .          | 42 |
| Figura 22 – Código em <i>python</i> para piscar um LED compilado no IDE <i>Thonny</i> . . . . .      | 43 |
| Figura 23 – Diagrama elétrico de ligação para o sensor ultrassônico HC-SR04. . . . .                 | 43 |
| Figura 24 – Código em <i>Python</i> para medir distância com um sensor ultrassônico HC-SR04. . . . . | 44 |
| Figura 25 – Diagrama elétrico do sensor de fluxo. . . . .                                            | 45 |
| Figura 26 – Código para calibrar o sensor de vazão YF S201. . . . .                                  | 46 |
| Figura 27 – Diagrama elétrico para o teste da válvula solenoide. . . . .                             | 47 |
| Figura 28 – Código para o teste da válvula solenoide. . . . .                                        | 47 |
| Figura 29 – Notificações recebidas por <i>e-mail</i> do usuário. . . . .                             | 48 |
| Figura 30 – Criando um canal para o envio dos dados para o <i>ThingSpeak</i> . . . . .               | 49 |
| Figura 31 – Configurando o canal do <i>ThingSpeak</i> . . . . .                                      | 49 |
| Figura 32 – Dados enviado do <i>Raspberry Pi</i> ao <i>ThingSpeak</i> . . . . .                      | 50 |
| Figura 33 – Página <i>Web</i> acessada pelo computador. . . . .                                      | 51 |
| Figura 34 – Página <i>Web</i> acessada pelo o <i>smartphone</i> . . . . .                            | 51 |
| Figura 35 – Diagrama elétrico do projeto. . . . .                                                    | 53 |
| Figura 36 – Prototipo montado. . . . .                                                               | 53 |
| Figura 37 – Projeto completo. . . . .                                                                | 54 |

## LISTA DE TABELAS

|                                                                                      |    |
|--------------------------------------------------------------------------------------|----|
| Tabela 1 – Parâmetros de funcionamento do sensor ultrassônico HC- SR04. . . . .      | 23 |
| Tabela 2 – Parâmetros de funcionamento do sensor de vazão YF S201. . . . .           | 27 |
| Tabela 3 – Especificações do Relé JQC-3FF-S-Z. . . . .                               | 29 |
| Tabela 4 – Parâmetros de funcionamento da válvula solenoide. . . . .                 | 30 |
| Tabela 5 – Teste com sensor ultrassônico com velocidade do som de $343m/s$ . . . . . | 44 |
| Tabela 6 – Resultado dos testes de calibração do sensor de fluxo. . . . .            | 46 |
| Tabela 7 – Custo do projeto em mercado nacional. . . . .                             | 52 |

## LISTA DE ABREVIATURAS E SIGLAS

|        |                                                       |
|--------|-------------------------------------------------------|
| API    | <i>Application Programming Interface</i>              |
| CERN   | <i>European Council for Nuclear Research</i>          |
| CLP    | Controladores Lógicos Programáveis                    |
| CPU    | <i>Central Process Unit</i>                           |
| DSP    | <i>Digital Signal Processor</i>                       |
| GB     | <i>Gigabyte</i>                                       |
| GPIO   | <i>General Purpose Input</i>                          |
| HD     | <i>Hard Disk</i>                                      |
| HTML   | <i>Hyper Text Markup Language</i>                     |
| HTTP   | <i>Hyper Text Transfer Protocol</i>                   |
| IDE    | <i>Integrated Development Environment</i>             |
| IoT    | <i>Internet of Things</i>                             |
| LDR    | <i>Light Dependent Resistor</i>                       |
| LED    | <i>Light Emitting Diode</i>                           |
| MMC    | <i>Multi Media Card</i>                               |
| MQTT   | <i>Message Queuing Telemetry Transport</i>            |
| ONU    | <i>Organização das Nações Unidas</i>                  |
| PC     | <i>Personal Computer</i>                              |
| PoE    | <i>Power over Ethernet</i>                            |
| PWM    | <i>Pulse-Width Modulation</i>                         |
| RAM    | <i>Random Access Memory</i>                           |
| REST   | <i>Representational State Transfer</i>                |
| SABESP | Companhia de Saneamento Básico do Estado de São Paulo |



|        |                                                 |
|--------|-------------------------------------------------|
| SANASA | Sociedade de Abastecimento de Água e Saneamento |
| SD     | <i>Secure Digital Card</i>                      |
| SDIO   | <i>Secure Digital Input Output</i>              |
| SGBD   | Sistemas de Gerenciamento de Bancos de Dados    |
| SMTP   | <i>Simple Mail Transfer Protocol</i>            |
| SO     | Sistema Operacional                             |
| SoC    | <i>System-on-a-Chip</i>                         |
| USB    | <i>Universal Serial Bus</i>                     |
| WWW    | <i>World Wide Web</i>                           |

## LISTA DE SÍMBOLOS

|             |                                       |
|-------------|---------------------------------------|
| $A$         | Ampére                                |
| $cm$        | Centímetro                            |
| $B$         | Campo Magnético                       |
| $F$         | Força de Lorentz                      |
| $Fe$        | Frequência de pulsos                  |
| $g$         | Grama                                 |
| $L$         | Litro                                 |
| $m$         | Metro                                 |
| $min$       | Minutos                               |
| $m^3$       | Metro cúbico                          |
| $q$         | Carga de Elétron                      |
| $R\$$       | Reais                                 |
| $v$         | Velocidade da Carga                   |
| $Vac$       | Tensão alternada                      |
| $Vdc$       | Tensão contínua                       |
| $Vz$        | Vazão do fluido em litros por minutos |
| $s$         | Segundo                               |
| $^{\circ}C$ | Grau Celsius                          |

# SUMÁRIO

|             |                                                                     |           |
|-------------|---------------------------------------------------------------------|-----------|
| <b>1</b>    | <b>INTRODUÇÃO . . . . .</b>                                         | <b>16</b> |
| <b>1.1</b>  | <b>Objetivos . . . . .</b>                                          | <b>18</b> |
| 1.1.1       | Objetivo Geral . . . . .                                            | 18        |
| 1.1.2       | Objetivos Específicos . . . . .                                     | 18        |
| <b>2</b>    | <b>TRABALHOS RELACIONADOS . . . . .</b>                             | <b>19</b> |
| <b>3</b>    | <b>REFERENCIAIS TEÓRICOS . . . . .</b>                              | <b>20</b> |
| <b>3.1</b>  | <b>Internet das Coisas . . . . .</b>                                | <b>20</b> |
| <b>3.2</b>  | <b>Eletrônica Digital . . . . .</b>                                 | <b>20</b> |
| <b>3.3</b>  | <b>Sensores . . . . .</b>                                           | <b>21</b> |
| 3.3.1       | Sensores Digitais: . . . . .                                        | 21        |
| <b>3.4</b>  | <b>Sensor Ultrassônico . . . . .</b>                                | <b>22</b> |
| <b>3.5</b>  | <b>Efeito Hall . . . . .</b>                                        | <b>24</b> |
| <b>3.6</b>  | <b>Sensor de Vazão de Fluido . . . . .</b>                          | <b>26</b> |
| <b>3.7</b>  | <b>Atuadores . . . . .</b>                                          | <b>27</b> |
| <b>3.8</b>  | <b>Módulo Relé . . . . .</b>                                        | <b>27</b> |
| <b>3.9</b>  | <b>Válvula Solenoide . . . . .</b>                                  | <b>29</b> |
| <b>3.10</b> | <b>Raspberry Pi . . . . .</b>                                       | <b>31</b> |
| 3.10.1      | Especificações do <i>Raspberry Pi 3 model B+</i> . . . . .          | 32        |
| 3.10.2      | Portas GPIO . . . . .                                               | 33        |
| <b>3.11</b> | <b>Sistema Operacional . . . . .</b>                                | <b>33</b> |
| <b>3.12</b> | <b>Software e Linguagem de Programação . . . . .</b>                | <b>34</b> |
| <b>3.13</b> | <b>Linguagem HTML . . . . .</b>                                     | <b>35</b> |
| <b>3.14</b> | <b>Banco de Dados . . . . .</b>                                     | <b>36</b> |
| <b>4</b>    | <b>METODOLOGIA . . . . .</b>                                        | <b>37</b> |
| 4.0.1       | Descrição Detalhada do Sistema . . . . .                            | 38        |
| <b>5</b>    | <b>TESTES, CALIBRAÇÃO E RESULTADOS . . . . .</b>                    | <b>41</b> |
| 5.0.1       | Escolha e Instalação do Sistema Operacional . . . . .               | 41        |
| 5.0.2       | Teste para o funcionamento da GPIO do <i>Raspberry Pi</i> . . . . . | 42        |
| 5.0.3       | Teste do Sensor Ultrassônico . . . . .                              | 43        |
| 5.0.4       | Teste do Sensor de Vazão . . . . .                                  | 45        |
| 5.0.5       | Teste da Válvula Solenoide . . . . .                                | 47        |
| 5.0.6       | Envio de notificação por e-mail . . . . .                           | 48        |
| 5.0.7       | Configuração do <i>ThingSpeak</i> e da Página Web . . . . .         | 48        |
| 5.0.8       | Análise de Custos para o Sistema de Automação Residencial . . . . . | 52        |
| 5.0.9       | Resumo dos resultados . . . . .                                     | 53        |
| <b>6</b>    | <b>CONCLUSÃO . . . . .</b>                                          | <b>56</b> |
| <b>7</b>    | <b>BIBLIOGRAFIA . . . . .</b>                                       | <b>57</b> |

|                                                                                                       |               |
|-------------------------------------------------------------------------------------------------------|---------------|
| <b>APÊNDICES</b>                                                                                      | <b>63</b>     |
| <b>APÊNDICE A – CÓDIGO DO SISTEMA PARA O CONTROLE DE<br/>UMA CAIXA D'ÁGUA DE UMA RESIDÊNCIA . . .</b> | <b>64</b>     |
| <b>APÊNDICE B – CÓDIGO EM HTML DA PÁGINA WEB PARA O<br/>SISTEMA DE MONITORAMENTO . . . . .</b>        | <b>70</b>     |
| <br><b>ANEXOS</b>                                                                                     | <br><b>72</b> |
| <b>ANEXO A – <i>DATASHEET</i> DO SENSOR DE VAZÃO MODELO YF -<br/>201 . . . . .</b>                    | <b>73</b>     |
| <b>ANEXO B – <i>DATASHEET</i> DO SENSOR ULTRASSÔNICO MODELO<br/>HC-SR04 . . . . .</b>                 | <b>75</b>     |

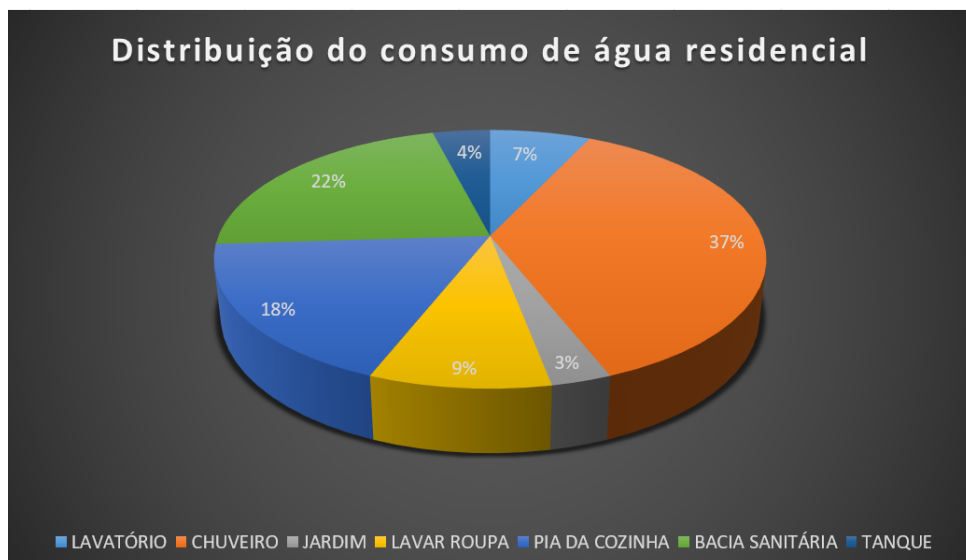
# 1 INTRODUÇÃO

Atualmente os recursos hídricos e sua conservação são um dos maiores desafios do desenvolvimento sustentável. Nesse século, tem-se como questão “A Crise da Água” identificada pela redução da oferta e aumento da demanda de água no planeta. Temos que, apenas no último século, a demanda de água aumentou seis vezes enquanto a população aumentou três vezes, tendo como consequência a deterioração dos mananciais e também a diminuição de água para o consumo das pessoas (TUCCI, 2017).

A grande escassez de água doce em nosso planeta pode ser ilustrada de diversas maneiras, entre elas, destacam-se os 22% do volume de água não desperdiçados, de um total de 80 milhões de metros cúbicos fornecidos pela SANASA (Sociedade de Abastecimento de Água e Saneamento) - empresa responsável pela rede hídrica de cidades do estado de São Paulo (SANASA, 2006). Diante desse quadro, temos o âmbito residencial como um dos responsáveis por grande parcela desse desperdício, seja através de consumo exagerado ou por pequenas vazões em reservatórios, torneiras, canos, entre outros (SANTAELLA *et al.*, 2007).

Segundo a ONU (Organização das Nações Unidas) uma pessoa necessita de cerca de 3,3 mil litros de água por mês, ou seja, em média 110 litros de água para fornecer às necessidades de consumo e higiene durante um dia (SABESP, 2015).

Figura 1 – Distribuição do consumo de água residencial .



Fonte: Adaptado de HAFNER, 2007.

No Brasil, esse consumo diário pode atingir mais de 200 litros/dia, sendo que cerca de 50% desse consumo, é utilizada no banheiro, em banhos, descargas, entre outras utilizações (SABESP, 2015). De acordo com a SABESP (Companhia de Saneamento Básico do Estado de São Paulo), um banho de 15 minutos com a válvula aberta, consome em média 135 litros de água, entretanto ao fechar o registro ao se ensaboar, a quantidade de água pode cair para até 45 litros. Na Figura 1 ilustrada a média do consumo de água em residências no Brasil, em geral, o chuveiro e a bacia sanitária são os maiores responsáveis no gasto de água em uma casa.

Visando a extinção de vazamentos relacionados aos reservatórios de água residenciais e uma redução da parcela de desperdício de água doce, oriundas desses vazamentos, a instalação de aparelhos capazes de monitorar o consumo e os níveis de água para a detecção de possíveis vazamentos chegaram como um recurso essencial, tanto para redução dos níveis de desperdício, quanto para preservação desse recurso que é essencial para a vida do planeta.

Automação residencial corresponde a um conjunto de tecnologias que consegue controlar dispositivos eletroeletrônicos na residência por meio de controladores, podendo ser de forma remota se estiverem conectados na rede (BOLZANI, 2010). Com o uso dessa tecnologia é possível realizar várias aplicações e facilitar o dia a dia dos integrantes da residência. O funcionamento da automação residencial pode ser realizado através de vários dispositivos, utilizando protocolos de comunicação como *ZigBee*, *Bluetooth*, *Z-Wave*, *Wi-Fi*, entre outros tipos (GURGEL, 2016).

De acordo com Bolzani (2010), a maioria dos sistemas de automação tem sensores que possibilitam o monitoramento de vários dados como: temperatura, incidência de luz, som, presença infravermelho, umidade, pressão, dentre outros. Com a criação do CLP (Controlador Lógico Programável) na década de 70, as indústrias começaram a se automatizarem, facilitando assim seus processos de fabricações, havendo um crescimento dos sistemas de automação industrial, fazendo com que tempos depois a automação chegasse até as residências (ROGGIA; FUENTES, 2016).

## 1.1 Objetivos

### 1.1.1 Objetivo Geral

O objetivo desse trabalho é desenvolver um sistema de automação e monitoramento do consumo de água, nível de água de uma caixa d'água e possíveis desperdícios de água por vazamento e/ou torneiras abertas, através de um sistema de automação residencial, onde o usuário terá acesso de forma remota aos dados do seu consumo d'água.

### 1.1.2 Objetivos Específicos

- Elaborar um projeto de sistema de automação e monitoramento d'água, através de um sistema de automação residencial utilizando o *Raspberry Pi*;
- Analisar os sensores e atuadores mais adequados para o sistema de monitoramento proposto;
- Instalar o sistema operacional *Raspbian* no *Raspberry Pi* e, em seguida, estudar a linguagem de programação *python* para implementação do sistema proposto;
- Desenvolver um sistema de monitoramento remoto para visualizar o controle hídrico de uma residência;
- Simular e verificar o funcionamento do sistema de monitoramento proposto.

## 2 TRABALHOS RELACIONADOS

A motivação que cerca a elaboração desse trabalho, tem o intuito de incentivar a preservação do meio ambiente por meio da conscientização acerca do uso racional da água através de recursos tecnológicos, como a automação residencial. Neste cenário, a tecnologia entra como uma aliada, que além de gerar um maior conforto ao usuário em seu dia a dia, auxilia na preservação do meio ambiente através do uso adequado da água no ambiente residenacial. Como embasamento teórico utilizou-se os trabalhos realizados por SOUZA, NUNES e BIANCHINI (2016); TAVARES ZANUZZO (2017) e SILVA (2014) que desenvolveram um sistema supervisor para monitoramento de consumo de água em uma residência utilizando o comparador de baixo custo, o *Raspberry Pi*.





ao invés de um sinal analógico e esse tipo de eletrônica faz-se o uso da álgebra booleana, a qual é base dos diversos circuitos digitais que conhecemos, como os *smartphone*, *desktop*, *tablet* e outros dispositivos eletrônicos presentes no mercado. Temos que na eletrônica digital, todas as saídas digitais são criadas a partir das entradas digitais (pulsos de onda quadrada) (SILVA FILHO, 1998).

### 3.3 Sensores

Existem diversas definições sobre o que é um sensor, mas, conforme Roggia e Fuentes (2016) um sensor é um dispositivo eletrônico que é sensível à energia de um ambiente, podendo ser: energia sonora, térmica, cinética, entre outras, e que relaciona essas informações sobre uma grandeza que será medida, como: pressão, posição, temperatura, vazão e outras grandezas físicas, fornecendo um sinal de saída de acordo com um sinal recebido em sua entrada. Conforme o sinal de saída dos sensores, eles podem ser classificados de duas formas, sendo elas: os sensores analógicos e os sensores digitais.

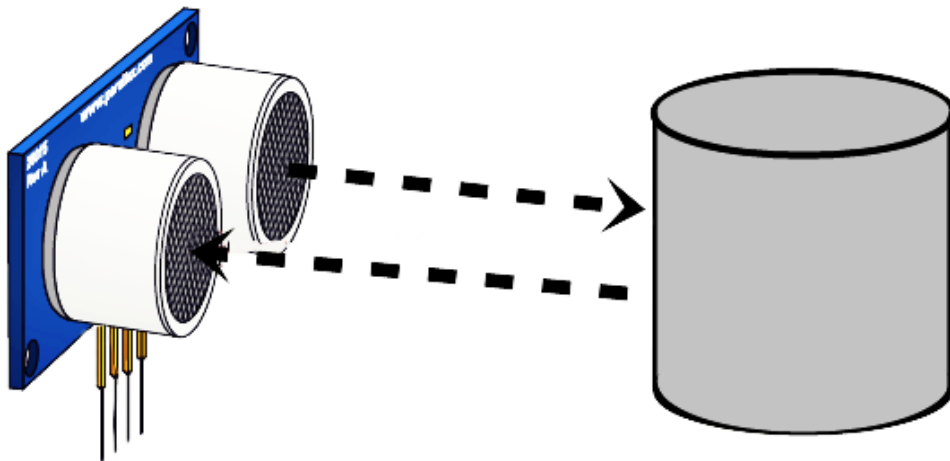
#### 3.3.1 Sensores Digitais:

Os sensores digitais são utilizados para verificar uma ocorrência ou não de um determinado evento, apresentando na sua saída apenas dois estados distintos, gerando um nível descrito como alto (*High*) ou baixo (*Low*) (“1” e “0”, respectivamente) ou ligado (*on*) e desligado (*off*), determinando a ausência ou presença da grandeza de entrada (PATSKO, 2006). Sendo o oposto dos sensores analógicos, nos quais seus valores são praticamente infinitos, temos que um sensor digital pode variar apenas entre certos estados já definidos, não havendo assim um valor intermediário entre eles (WENDLING, 2010).

### 3.4 Sensor Ultrassônico

O sensor ultrassônico HC-SR04 e o JSN-SR04T são instrumentos utilizados para medições de distâncias através das ondas ultrassônicas. Os dois modelos tem consigo um sistema de controle no qual pode transmitir e receber sinal ultrassônico (ALDOULAH, 2017). A Figura 3 ilustra o dispositivo enviando e recebendo o sinal refletido pelo objeto. O sensor JSN-SR04t, demonstrado na Figura 4, tem uma grande vantagem quando comparado com o HC-SR04, pois ele utiliza de um piezoelétrico para efetuar a transmissão e a captação do sinal, sendo uma peça única que contém um cabo com 2,5 m de comprimento. Com isso, o transdutor pode entrar em contato com água, enquanto sua placa de circuito fica sem o contato com a água, tendo um alcance de 4,5 m (PRUDÊNCIO, 2020).

Figura 3 – Sensor ultrassônico HC-SR04.



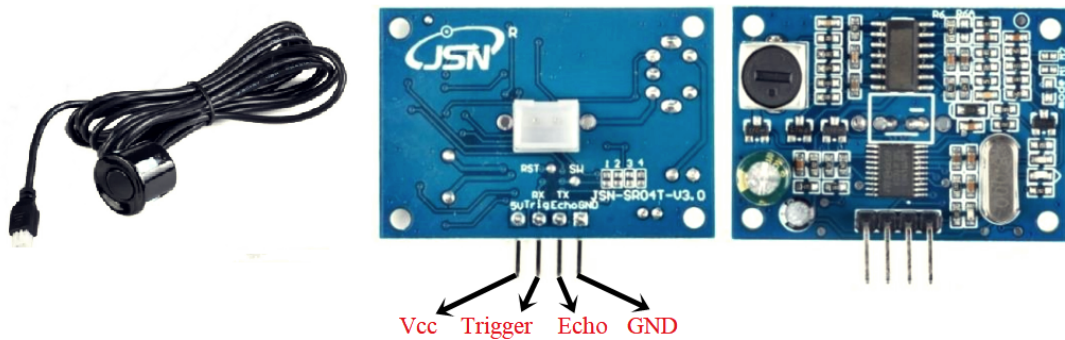
Fonte: Adaptado de ALDOULAH, 2017.

Principais aplicações dos sensores ultrassônicos são elas:

- Medição de distância;
- Verificação de presença;
- Detecção de obstáculos;
- Rompimento de fio;
- Posicionamento de sistemas robóticos.

Tanto o sensor HC-SR04, quanto o JSN-SR04T tem em sua placa quatro pinos, sendo eles, respectivamente, Vcc, Trigger, Echo e GND, conforme ilustrado na Figura 4 (PRUDÊNCIO, 2020).

Figura 4 – Sensor ultrassônico JSN-SR04T.



Fonte: Adaptada de SILVA, 2020.

- Vcc: Pino para alimentação do sensor em que se utiliza uma tensão contínua de 5 Volts para seu funcionamento;
- Trigger: Esse pino é utilizado para ativar o envio da onda ultrassônica pelo transmissor do sensor, em que o sinal do trigger deve ser realizado com um pulso com 10  $\mu$ s a nível lógico *high*. Assim que o sensor receber um sinal de trigger em *high*, o mesmo emitirá 8 pulsos de uma onda ultrassônica na frequência de 40 kHz do transmissor;
- Echo: Pino responsável para receber o sinal ultrassônico refletido pelo objeto;
- GND: Pino para conectar o módulo com a referência de tensão utilizada, ou seja, o terra do circuito.

Para aplicações em projetos, os módulos ultrassônicos podem ser utilizados de forma bastante eficaz, visto que o dispositivo é silencioso e seguro, pois os seres humanos não podem ouvir o som a essa frequência. Devido ao seu baixo custo de aquisição, os projetos podem ser desenvolvidos com um bom custo benefício contendo uma boa precisão. Conforme descrito pelo *datasheet* do fabricante disponível em anexo, o módulo HC-SR04 pode medir de 2 cm a 400 cm de distância, com uma precisão de até 3 mm. Na Tabela 1 mostra alguns parâmetros de funcionamento para o sensor segundo o fabricante.

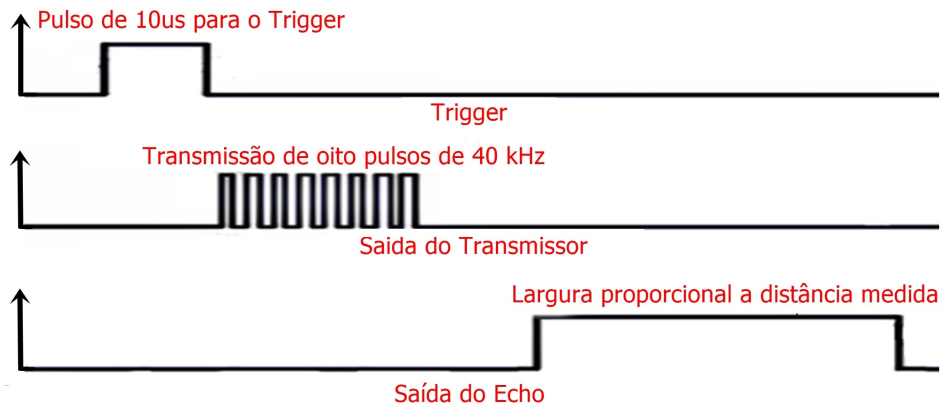
Tabela 1 – Parâmetros de funcionamento do sensor ultrassônico HC- SR04.

| Parâmetros              | Valores            |
|-------------------------|--------------------|
| Tensão de Funcionamento | 5 Volts            |
| Frequência operacional  | 40 kHz             |
| Intervalo de detecção   | 2 cm a 400 cm      |
| Resolução               | 3 mm               |
| Corrente operacional    | $\leq 15mA$        |
| Dimensões do sensor     | 45mm x 20mm x 15mm |

Fonte: Adaptado do *datasheet* do fabricante, disponível em anexo.

O funcionamento do módulo é realizado quando ocorre um envio de um pulso de 10  $\mu$ s, indicando o início da transmissão do sinal. Como mostra na Figura 5 são enviados oito pulsos de 40 kHz, assim o dispositivo espera o retorno do sinal refletido ao nível lógico *high*, sendo possível determinar a distância entre o sensor e o objeto através do tempo de percurso e a velocidade do som no ambiente (JAHANKIT, 2020).

Figura 5 – Diagrama de tempo do HC-SR04.



Fonte: Adaptado de JAHANKIT, 2020.

Os dois módulos apresentados funcionam através do envio de um sinal ultrassônico emitido pelo sensor e quando esse sinal é refletido pelo objeto e captado pelo receptor, pode-se calcular o tempo de percurso da onda (THOMAZINI, 2005). Através do tempo total entre o envio e a recepção deste pulso, pode-se calcular a distância do objeto, conforme é mostrado na Equação 3.1

$$Distancia = \frac{(TempoTotal) * (VelocidadeDoSom)}{2} \quad (3.1)$$

Onde:

- Distancia: Distância do sensor ultrassônico até o objeto (*m*);
- TempoTotal: Tempo de resposta captada pelo o sensor (*s*);
- VelocidadeDoSom: Velocidade do som (*m/s*).

### 3.5 Efeito Hall

Descoberto em 1879 pelo físico estadunidense Edwin Hall, o efeito Hall é dado pela força de Lorentz com a movimentação de elétrons envolvidos em um campo magnético (LIMA, 2017). Na existência de um fluxo de corrente em um material que não esteja submetido a um campo magnético, as linhas equipotenciais que cruzam perpendicularmente o fluxo são linhas retas (LIMA, 2017). Na Equação 3.2, pode ser vista a força de Lorentz no deslocamento de elétrons em um material.

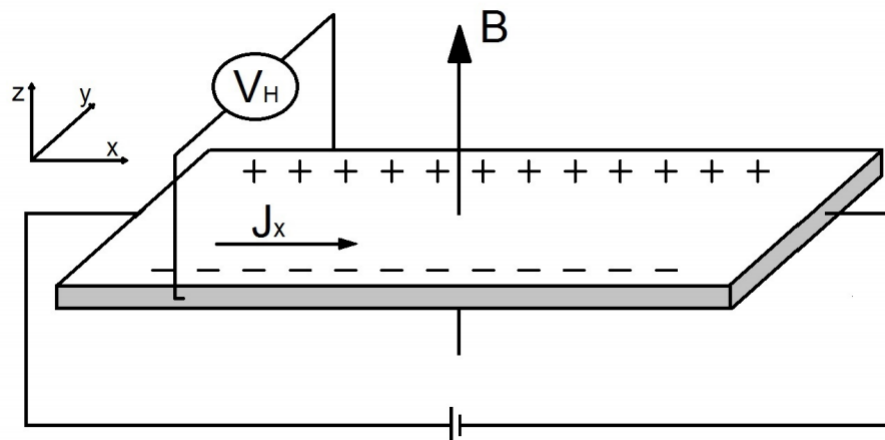
$$F = q(vB) \quad (3.2)$$

Onde:

- $q$ : Carga do Elétron;
- $v$ : Velocidade da Carga;
- $B$ : Campo Magnético.

O resultado da equação demonstra a força com uma direção mutuamente perpendicular ao fluxo de corrente e ao campo magnético. Segundo Cassiolato (2003), na existência de um fluxo de corrente em um material suscetível a um campo magnético perpendicular, o ângulo do qual o fluxo de corrente é mudado pelo campo magnético é chamado de ângulo *Hall* e é um parâmetro dado conforme o material, dado que é determinado através da mobilidade do elétron  $\mu$ , determinando o coeficiente de *Hall*  $R_H$ . Isso faz com que as suas linhas equipotenciais ao longo do comprimento do material sejam inclinadas, criando uma tensão medida ao longo do material, sendo conhecida com tensão *Hall*, assim existindo um sinal com proporção ao campo magnético aplicado (PIMENTEL JUNIOR, 2007).

Figura 6 – Efeito *Hall*.



Fonte: LIMA, 2017.

De acordo com Minucci e Lima (2013), os portadores de carga quando deslocados sofrem a ação de uma força magnética perpendicular na direção de sua velocidade, fazendo com que suas cargas se desviem de seu caminho retilíneo, se juntando a um dos lados do condutor. Com essa diferença na posição das cargas nos vários pontos do condutor, surge uma diferença de potencial elétrica entre os dois lados da placa, podendo ser verificada através de voltímetro, ou por controlador, visto que é emitido um sinal de tensão conforme a Figura 6.

### 3.6 Sensor de Vazão de Fluido

O sensor de fluxo de vazão é um dispositivo que consegue medir a quantidade de fluido que percorre o seu interior. O sensor de vazão YF S201 ilustrado na Figura 7 é compatível com diversos controladores. É formado pelo seu corpo em plástico, um rotor para girar através de uma hélice na passagem do fluido e um sensor de efeito *hall* que emite um sinal elétrico com uma determinada frequência sendo proporcional à quantidade do fluido que passou pelo dispositivo (MATOS; MACIEL, 2019).

Figura 7 – Sensor de vazão modelo YF S201.



Fonte: Adaptado de SANTOS, 2021.

Conforme descrito no *datasheet* disponibilizado pelo fabricante, exibido em anexo, quando o sensor representar o valor equivalente de 1 L/min, sua frequência será de 7,5 pulsos/s, logo podemos calcular a quantidade de litro por minuto que atravessa o sensor através da Equação 3.3.

$$V_z = \frac{F_e}{7,5} \quad (3.3)$$

Onde:

- $V_z$  = Vazão do fluido em litros por minutos;
- $F_e$  = Frequência de pulsos.

Conforme demonstrado na Figura 7 esse sensor tem três conexões para a ligação com o controlador. O condutor preto representa o GND (terra) do dispositivo, o vermelho é o Vcc (tensão de alimentação) e o amarelo é utilizado para a saída de pulso gerada pelo efeito *Hall* (DORNELAS, 2017). Na Tabela 2 são mostrados alguns dos parâmetros de funcionamento disponibilizados pelo fabricante do sensor.

Tabela 2 – Parâmetros de funcionamento do sensor de vazão YF S201.

| Parâmetros                   | Valores             |
|------------------------------|---------------------|
| Tipo do sensor               | Efeito <i>Hall</i>  |
| Tensão de operação           | 5 a 24 Vdc          |
| Corrente máxima              | 15 mA               |
| Faixa de fluxo               | 1 a 30 (L/min)      |
| Pulso por litro              | 450                 |
| Frequência                   | 7,5 * Fluxo (L/min) |
| Pressão maxima               | 2 MPa               |
| Temperatura de funcionamento | -28 a 80 °C         |
| Bitola das conexões          | 1/2 de polegadas    |

Fonte: Adaptado do *datasheet* do fabricante, disponível em anexo.

### 3.7 Atuadores

Segundo Botto, Neves e Camargo (2016) atuadores são dispositivos que desempenham movimentos físicos, realizando a conversão de energia, sendo a mais comum a elétrica, em energia mecânica, hidráulica e pneumática. Temos que o atuador necessita de um sinal de controle e uma fonte de energia que ao receber um sinal de controle, o dispositivo utiliza da energia da fonte para produzir um movimento mecânico, podendo ser uma válvula utilizada para controlar a vazão de um fluido, uma bomba para regular o fluxo, entre outros (FERNANDES, 2018). Existem vários tipos de atuadores no mercado, alguns deles são apresentados na Figura 8.

Figura 8 – Exemplos de Atuadores do mercado.



Fonte: Adaptado de SANTOS, 2021.

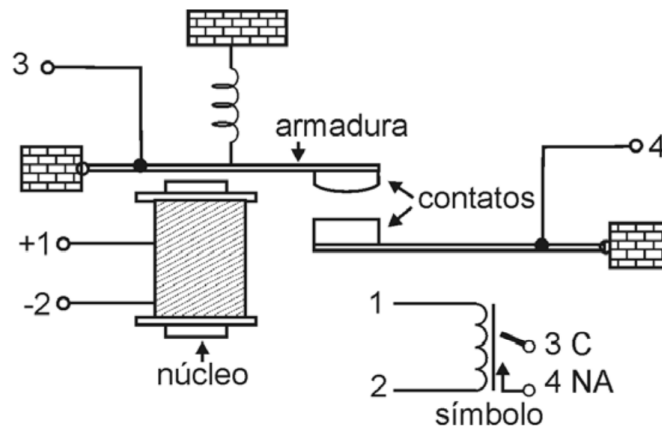
### 3.8 Módulo Relé

Relé é um interruptor eletromecânico que tem a capacidade realizar o controle da passagem de cargas elétricas fazendo o uso de um sistema de controle de baixa potência. O módulo relé funciona quando uma corrente elétrica ao passar por uma bobina origina um campo eletromagnético e altera o estado dos contatos elétricos do dispositivo (SAN, 2019). Existem dois funcionamentos nos módulos relés, tem o normalmente fechado, onde a saída para carga fica ativado quando o relé não é acionado e o normalmente aberto, quando o relé não é acionado e a



carga fica desativada. Nos dois modelos quando o relé é submetido a uma corrente de ativação ambos mudam seu estado (BRAGA, 2017). A Figura 9 mostra o princípio de funcionamento do relé eletromecânico.

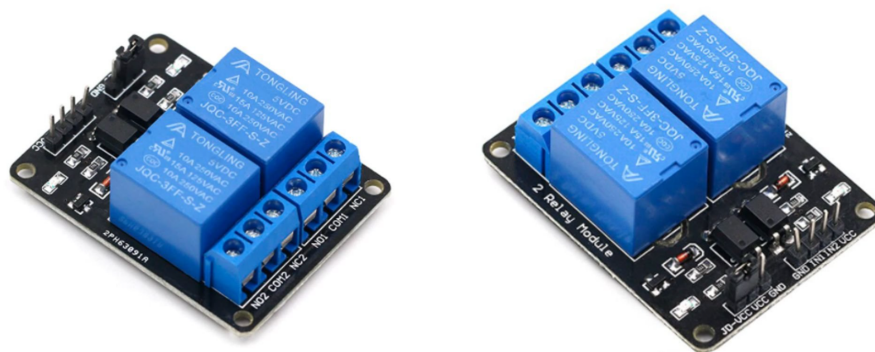
Figura 9 – Princípio de funcionamento do relé eletromecânico.



Fonte: BRAGA, 2017.

A utilização de relés em microcontroladores e microprocessadores tem uma grande vantagem, pois o acionamento de cargas elétricas não dispõe de conexão entre o circuito dos controladores e a carga de alta potência, tendo a proteção do controlador contra elevação de corrente e tensão. A Figura 10 ilustra um módulo relé compatível com controladores como *Raspberry PI*, ESP 32, Arduino, entre outros.

Figura 10 – Módulo relé de 2 Canais.



Fonte: Adaptado de SANTOS, 2021.

Os módulos relés podem ser encontrados no mercado e podem conter apenas um relé ou vários relés integrados no circuito. Cada um deles dispõe de um LED (*Light Emitting Diode*) indicador para demonstrar o estado lógico do relé. Conforme especificado pelo fabricante desse módulo, cada uma de suas saídas pode suportar cargas de até 10 A em 250 Vac, 15 A em 125 Vac ou 30 Vdc. Algumas de suas principais características de funcionamento estão descritas na Tabela 3.

Tabela 3 – Especificações do Relé JQC-3FF-S-Z.

| Parâmetros                      | Valores      |
|---------------------------------|--------------|
| Modelo                          | JQC-3FF-S-Z  |
| Tempo de acionamento de contato | 10 ms        |
| Temperatura de funcionamento    | -45 a 85 °C  |
| Peso unitário                   | 10 g         |
| Proteção do relé                | IP67         |
| Humidade de operação            | - 40 a 80 %  |
| Tensão de operação              | 5 V          |
| Max. tensão de comutação        | 277Vac/30Vdc |
| Max. corrente de comutação      | 15 A         |

Fonte: Adaptado do *datasheet* do fabricante, disponível em anexo.

### 3.9 Válvula Solenoide

Segundo Silva (2002) solenoides são dispositivos eletromecânicos que funcionam através do deslocamento criado pela ação de um campo magnético gerado por meio de uma bobina. Solenoide é um termo utilizado para se referir a uma bobina de fio utilizada como um eletroímã e são usadas na construção de diversos dispositivos, como é o caso do uso em alguns tipos de válvulas para controle de fluidos.

A válvula solenoide, diferentemente da válvula manual, elimina a necessidade do contato humano para ligar ou desligar. Ao invés do uso de um registro temos o próprio solenoide realizando a movimentação do êmbolo abrindo ou fechado a passagem de fluido (OLIVEIRA, 2017).

Figura 11 – Válvula solenoide.



Fonte: Adaptado de Robocore, 2021.

As válvulas solenoide são um atuador de muita importância na utilização de projetos hidráulicos, principalmente nas automatizações residenciais, visto que, além do seu baixo custo de aquisição, ela tem um simples funcionamento e fácil instalação, podendo até substituir a válvula manual convencional (OLIVEIRA, 2017). A Figura 11 mostra uma válvula solenoide NF

(Normalmente Fechada) acionada por 220 V, que quando acionada libera a passagem do fluido em seu interior, e no instante que é desenergizada, bloqueia a passagem do fluido (OLIVEIRA; CAVALCANTI, 2017). Na Tabela 4 mostra as especificações do fabricante de uma válvula solenoide.

Tabela 4 – Parâmetros de funcionamento da válvula solenoide.

| Parâmetros                   | Valores                  |
|------------------------------|--------------------------|
| Tipo de válvula              | NF (Normalmente Fechada) |
| Vazão máxima                 | 40 Litros/min            |
| Tensão da bobina             | 220 VAC                  |
| Consumo de corrente          | 74 mA                    |
| Diâmetro de conexão          | 3/4"                     |
| Vida útil                    | 50000 operações          |
| Temperatura máxima do fluido | 60 °C                    |

Fonte: Adaptado do *datasheet* do fabricante, em anexo.

Nota-se que a válvula solenoide possui boas especificações, conforme a Tabela 4, quanto as suas características para um sistema de automação residencial simples.

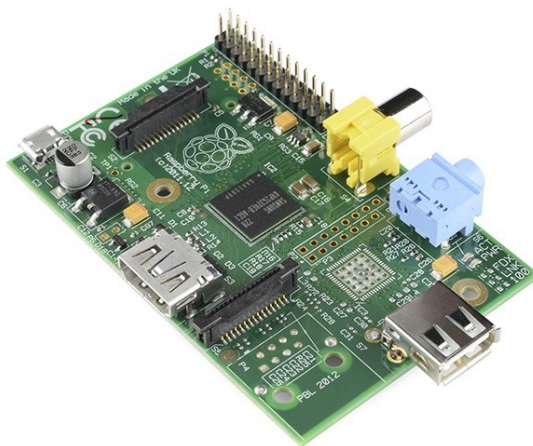
### 3.10 *Raspberry Pi*

Com início em 2006 no Reino Unido, uma fundação sem fins lucrativos, formada por um grupo de cientistas do Laboratório de Computação da Universidade de Cambridge, na Inglaterra, começou a trabalhar em um microcomputador baseado no Atmel ATmega 644, que posteriormente serviria de base para o *Raspberry Pi* (ATZINGEN, 2017). Tempos depois esse grupo desenvolveu um computador com o tamanho de aproximadamente o mesmo de um cartão de crédito, sendo capaz de ter as mesmas funções de um *desktop* ou notebook comum, com o foco de facilitar o acesso a computação básica para jovens em escolas e universidades da Europa, com um computador de baixíssimo custo (Raspberry Foundation, 2006).

A origem do seu nome se deu por ser bastante comum, sendo uma tradição já de que nomes de empresas voltadas a área de informática e tecnologia, recebem nomes de frutas, assim surgiu o nome *Raspberry*, que em português se refere a fruta Framboesa (SANTOS, 2018). Já o nome “*Pi*” surgiu a partir da palavra ‘*Python*’ dado que seus desenvolvedores tinham nos objetivos da plataforma a construção de um computador que poderia se utilizar da linguagem de programação *Python* e que a plataforma da linguagem já vinha instalada no sistema (SANTOS, 2018).

Seus criadores no começo pensavam que *Python* seria a única linguagem de programação que seria disponibilizada na plataforma, logo depois foram acrescentadas diversas outras linguagem no *Raspberry Pi* possibilitando mais aplicações com o computador (Raspberry Foundation, 2006).

Figura 12 – Primeira versão do *Raspberry Pi*, 1 modelo A.



Fonte: Adaptado de *Electrofun*, 2018.

A principal diferença do *Raspberry Pi*, para um computador comum, é que esse dispositivo é muito compacto e mesmo assim ele possui todos os principais componentes de um computador numa pequena placa. Em 2012, veio seu primeiro lançamento oficial, chamado de *Raspberry Pi 1 Model B+*, podendo ser visto na Figura 12, na época custava cerca de 35 dólares e pouquíssimo tempo depois foi lançada outra versão com o valor de 25 dólares, um valor bem

acessível quando comparado com outros tipos de PCs. Ele vinha com um *System-on-a-Chip* (SoC) da Broadcom, o BCM2835, sendo assim, em uma pequena placa já estava integrado GPU, DSP, SD, RAM, CPU e uma porta USB (MERCES, 2013).

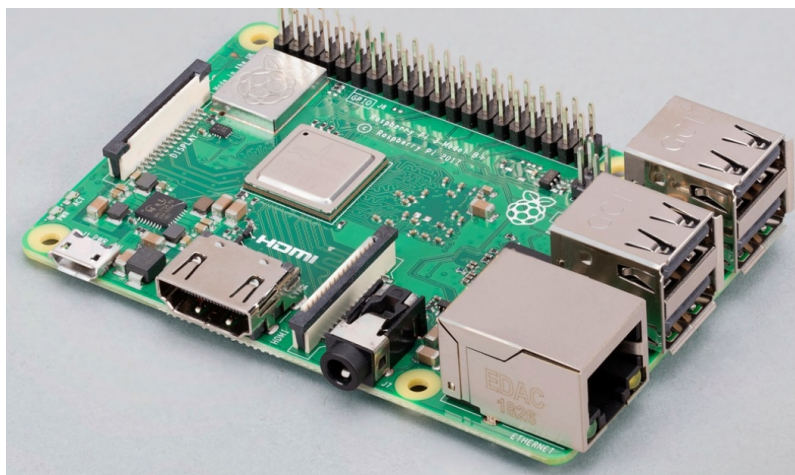
O primeiro lançamento do *Raspberry Pi*, vinha com cerca 256 MB de RAM e com um processador ARM *single-core* de 700 MHz e seu armazenamento era dado através de uma entrada para cartão de memória, compatível com os formatos SD, SDIO, MMC, que para o seu tempo já tinha um bom poder de processamento visando o custo de aquisição, (MERCES, 2013).

### 3.10.1 Especificações do *Raspberry Pi 3 model B+*

Atualmente existem diversos modelos e versões do *Raspberry Pi*, mas para o nosso projeto trabalhou-se com o modelo *Raspberry Pi 3 model B+*, visto que ele tem um ótimo poder de processamento e um bom custo benefício, além disso, dispõe de *Wi-Fi* e *Bluetooth* integrados à placa, dispensando a necessidade de adaptadores para esses tipos de comunicação (LOBO, 2019). O *Raspberry Pi* utilizado nesse projeto pode ser encontrado no mercado internacional com um preço a partir de 30 dólares, sendo bem acessível diante seu poder de processamento.

Esse novo modelo mantê-se com mesmo processador Broadcom BCM2837, porém com um encapsulamento diferente, fazendo com que seja eficiente na questão de dissipar calor, gerando uma melhoria no seu processador, tornando-se capaz de operar em uma velocidade de 1,4GHz. Essa versão conta com 1 GB de memória RAM, além de que, já vem com a conexão Wi-Fi, tendo suporte até 5 GHz e também com as conexões *Bluetooth 4.2/BLE* (LOBO, 2019). Nesse modelo conta também com o sistema PoE (*Power over Ethernet*) que através de um HAT específico, fazendo com que seja capaz a alimentação da placa, utilizando um cabo de rede sem precisar recorrer a outras fontes externas, logo a baixo pode-se visualizar na Figura 13 o *Raspberry Pi 3 model B+* (LOBO, 2019).

Figura 13 – *Raspberry Pi 3 model B+*.



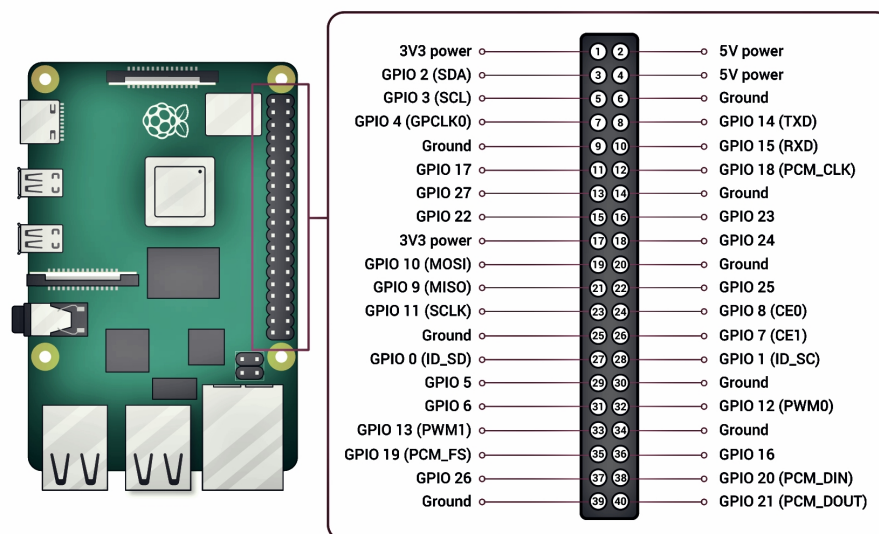
Fonte: Adaptado de *Raspberrypi.org*, 2018.

### 3.10.2 Portas GPIO

De acordo com Rakshit e Panse (2017), GPIO (*General Purpose Input/Output*) são portas genéricas em um circuito integrado que podem ser programadas como portas de entrada ou saída de dados, provendo uma interface entre os periféricos e os microcontroladores/microprocessadores. O *Raspberry Pi* tem um poderoso recurso, que é a sua linha de pinos GPIO integrados à placa, totalizando um conjunto com 40 pinos, conforme ilustrado na Figura 14, sendo encontrado em todas as placas *Raspberry Pi* a partir da versão 1 Modelo B+ (2014) (BACHRUDIN, 2017).

O *Raspberry Pi 3 model B+* tem dois pinos de 5V e mais um 3,3V, além de vários GPIO de aterramento GND (0V) na placa, os demais pinos da placa são de uso geral com uma tensão de 3,3 V, podendo ser configurado para saída ou entrada (BACHRUDIN, 2017). Uma porta GPIO designada como um pino de saída ou entrada pode ser definida como nível lógico alto 3,3V ou nível lógico baixo 0V, sendo configurados pelo seus resistores internos *pull-down* ou *pull-up* via *software*.

Figura 14 – Diagrama de pinagem do *Raspberry Pi 3 model B+*.



Fonte: Adaptado de *Raspberrypi.org*, 2018.

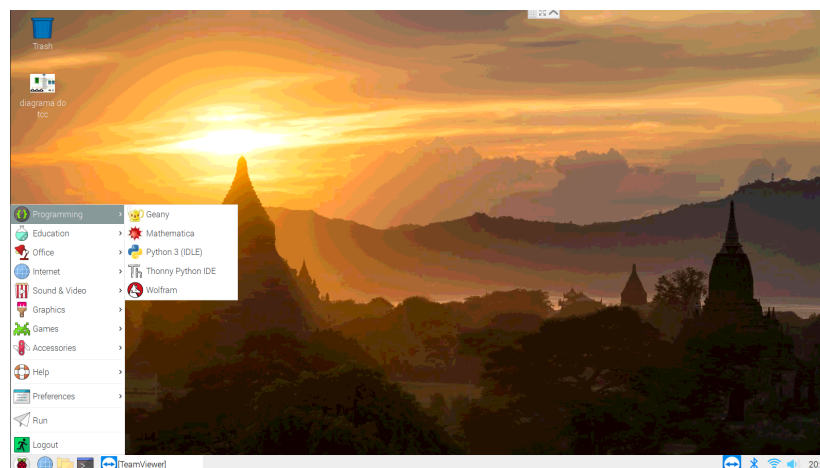
## 3.11 Sistema Operacional

Um Sistema Operacional (SO) é um conjunto de *Softwares* cuja função é fornecer uma interface permitindo que os seus usuários possam interagir com o computador (SILVA, 2019). Atualmente no mercado existem várias opções de SO, por exemplo, o *MacOS*, *Windows*, *Linux*, que podem ser utilizados tanto em *desktops*, quanto em notebooks (SILVA, 2017). A maioria dos computadores do mercado recorre a um desses sistemas operacionais como: *Microsoft Windows* ou *MacOS*. Entretanto, esses SO foram criados a partir de um código fonte fechado, desenvolvidos em um ambiente privado, havendo um custo para aquisição do sistema (UPTON; HALFACREE, 2013).



O *Raspberry Pi* faz uso do sistema operacional *Raspbian*, um sistema operacional de código aberto, derivado do *Debian* do *Linux*, sendo disponibilizado gratuitamente para *download* no site do *Raspberry Pi* (<http://www.raspberrypi.org/downloads>) (EBERMAM, *et al.*, 2017), permitindo que a placa tenha um custo reduzido, já que não é necessário pagar pelo sistema operacional. Entretanto, o *Raspberry Pi* pode utilizar várias outras distribuições do *Linux*, além de outros sistemas operacionais (EBERMAM *et al.*, 2017). Na Figura 15 pode ser vista a área de trabalho do SO *Raspbian*.

Figura 15 – Área de trabalho do Sistema Operacional *Raspbian*.



Fonte: Arquivo Próprio, 2021.

### 3.12 Software e Linguagem de Programação

Faz-se necessário o uso de uma linguagem de programação para realizar a comunicação entre o usuário e o computador, sendo ela responsável em converter as instruções dos seres humanos em um código que a máquina possa compreender e executá-la (EBERMAM *et al.*, 2017). Utilizando de uma linguagem de programação, é possível escrever um código com diversas instruções e tarefas, e, através de um compilador, é possível converter esse código em uma linguagem binária que é compreendida pela máquina, obtendo o valor 0, quando está sem a passagem de corrente elétrica e o valor 1, quando existe passagem de corrente elétrica (EBERMAM *et al.*, 2017).

Como o *Raspberry Pi* usa um sistema operacional completo, o *Raspbian*, isso possibilita o uso de diversas linguagens de programação disponíveis para as criações de projetos na plataforma, sendo mais comum a utilização da linguagem de programação C e *Python*, visto que essa duas linguagem tem bibliotecas para realizar o controle das portas GPIO do controlador (SOUZA; NUNES; BIANCHINI, 2016).

Desenvolvida em 1989 e divulgada em 1991, por Guido Van Rossum, *Python* é uma linguagem de programação *open-source* (código livre) interpretada como linguagem de alto nível e gerenciada pela *Python Software Foundation*, um modelo de desenvolvimento comunitário aberto (WOJCIECHOWSKI *et al.*, 2018). Uma das características mais importante da linguagem

*Python* é que ela possui um nível de abstração ao qual fica mais fácil de ser compreendido, podendo ser desenvolvido um código de forma clara e direta facilitando assim o projeto.

O *Raspberry Pi* foi desenvolvido para trabalhar com esse tipo de linguagem. O sistema operacional do *Raspbian* vem com a IDE (*Integrated Development Environment*) do *Python* já instalada. Além do mais, existe a biblioteca *RPi.GPIO*, da *Python Software Foundation*, sendo capaz de realizar todo o controle das portas GPIO da placa, possibilitando a criação de programas através de um *script* em um editor de texto e depois executá-lo no IDE da plataforma (CAMPANHA, 2017).

### 3.13 Linguagem HTML

A internet é uma rede formada por diversas redes que estão interligadas entre si, construindo uma imensa comunidade usuários virtuais (COSTA, 2007). WWW (*World Wide Web*) é um sistema de informação que conecta o usuário ao servidor distribuído na Internet. Na *Web* devido seu grande crescimento, surgiu a necessidade de incorporar as informações (dados, documentos, menus) (COSTA, 2007).

Por volta do ano 1990, surgiu a criação do protocolo HTTP (*Hyper Text Transfer Protocol*), pouco tempo após o seu desenvolvimento, foi criada no CERN (*European Council for Nuclear Research*) na Suíça, a linguagem HTML (do inglês, *Hyper Text Markup Language*) no português conhecida como Linguagem de Marcação de Hipertexto (TORRES, 2018). Sendo desenvolvida inicialmente apenas para o próprio uso dos desenvolvedores, dado que o objetivo foi no desenvolvimento de uma linguagem onde simplificasse a transmissão de dados e informações entre os diversos cientistas de diferentes universidades (SILVA, 2019).

Tempos depois a linguagem HTML foi se popularizando bastante e recebendo atualizações, se tornando padrão no uso e exibição de informações para o acesso das páginas na *Web* (SILVA, 2019). Essa linguagem é composta de textos e códigos especiais chamados de marcas ou *tags*. Através dessa linguagem as suas linhas de códigos são entendidas pelo *browser* e exibe a pagina final no navegador ao usuário, sem necessidade de realizar a compilação do código (COSTA, 2007).

*Web Browser* é um *software* que possibilita a visualização de uma página Web localizada em outro servidor, que pode ser escrita em HTML, por exemplo. Para tal transferência de dados ser possível é necessário a utilização do protocolo de transferência hipertexto ou HTTP (*Hypertext Transfer Protocol*) (COSTA, 2007).

Importante ressaltar que o HTML não é considerado uma linguagem de programação, como *Python*, *C++* e *Assembly*, por exemplo, mas sim uma linguagem de marcação, permitindo aos desenvolvedores a possibilidade de definir onde e como os elementos da página deverão aparecer no navegador, por exemplo: cor da fonte, tamanho da letra, modelo de fonte, tamanho de imagem, posicionamento, etc (TORRES, 2018).

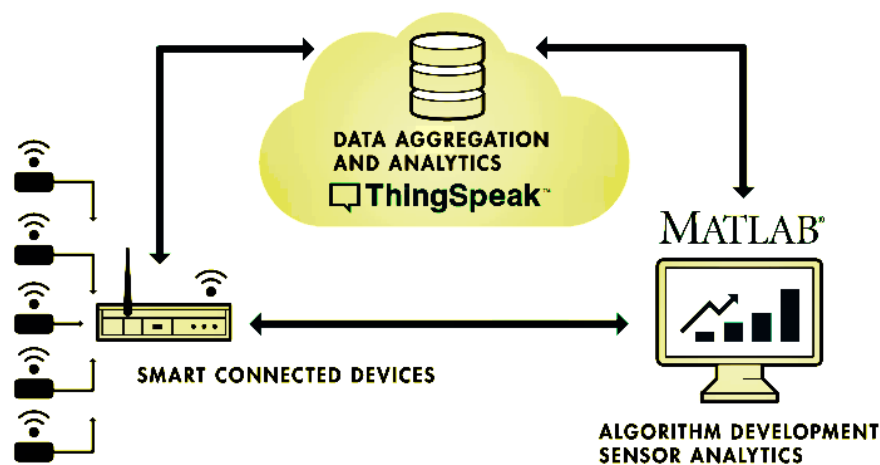


### 3.14 Banco de Dados

Segundo Ramez e Navathe (2005), um banco de dados é um conjunto de dados relacionados em forma de fatos que podem ser gravados e que possuem um significado implícito, exemplo: nomes de pessoas, lista telefônica, endereço, entre outros dados. SGBD (Sistemas de Gerenciamento de Bancos de Dados) são *softwares* que realizam o armazenamento de dados. Esses dados são normalmente organizados em tabelas e mantêm relacionamentos de acordo com a lógica de organização dos dados (RAMEZ e NAVATHE, 2005).

O *ThingSpeak* é uma plataforma de serviço e API (*Application Programming Interface*) para internet das coisas capaz de realizar o armazenamento de dados e também fazer o envio de dados (comando) para diferentes dispositivos que estejam conectados à internet em tempo real. Para receber e enviar dados no *ThingSpeak*, o usuário pode escolher dois métodos, sendo eles o REST (*Representational State Transfer*) e o MQTT (*Message Queuing Telemetry Transport*) (OLIVEIRA, 2019). A plataforma possui também a integração com o *software* *Twilio*, *MATLAB* e *Twitter*, permitindo o desenvolvimento de protótipos e sistemas de internet das coisas sem necessitar de desenvolver um servidor ou realizar a criação de *software web* (THINGSPEAK, 2021).

Figura 16 – Modelo do *ThingSpeak*.



Fonte: Adaptado de *ThingSpeak*, 2021.

Na Figura 16 está descrito o funcionamento do *ThingSpeak*, onde à esquerda temos os dispositivos inteligentes capazes de coletar dados através de diferentes sensores, como exemplo, sensores de temperatura, umidade, distância, pressão e entre outras. No centro, temos um banco de dados na nuvem onde esses dados são armazenados e analisados em tempo real e por fim, ao lado direito da Figura encontra-se a criação do algoritmo associado à aplicação IoT onde o usuário pode visualizar todas as análises históricas dos dados coletados pelos sensores diante de qualquer dispositivo que tenham acesso à internet (ThingSpeak, 2021).

## 4 METODOLOGIA

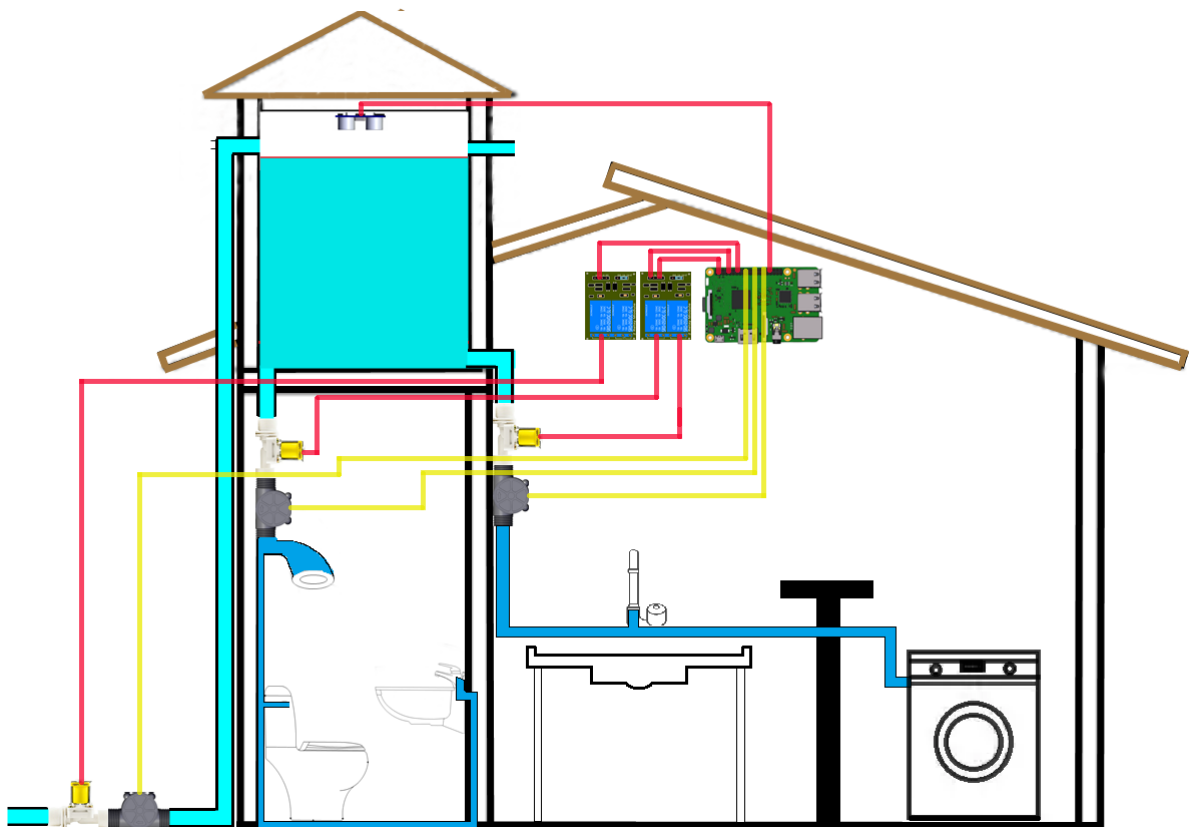
O presente trabalho será desenvolvido de acordo com a seguinte metodologia:

- Instalação do Sistema Operacional no *Raspberry Pi*;
- Estudo da linguagem de programação *python* e linguagem de marcação HTML;
- Estudo teórico e prático sobre o funcionamento das GPIO do *Raspberry Pi* utilizando sensores e atuadores;
- Realização de teste e calibração dos sensores e atuadores;
- Integração da automação com um banco de dados na nuvem para o monitoramento do sistema, através de uma página *Web*;
- Análise de custos do sistema;
- Simulação e análise teórica e prática acerca da montagem do sistema de automação residencial;
- Avaliação dos resultados obtidos.

#### 4.0.1 Descrição Detalhada do Sistema

O sistema consiste em realizar uma automação residencial para controlar o sistema hidráulico de uma casa. O projeto foi construído para uma residência simples, dispondo de dois quartos, uma sala, uma área, um banheiro e uma cozinha com lavanderia. O abastecimento de água localiza-se no banheiro e na cozinha/lavanderia. A planta do projeto é apresentado na Figura 17.

Figura 17 – Representação do sistema de automação.



Fonte: Arquivo Próprio, 2021.

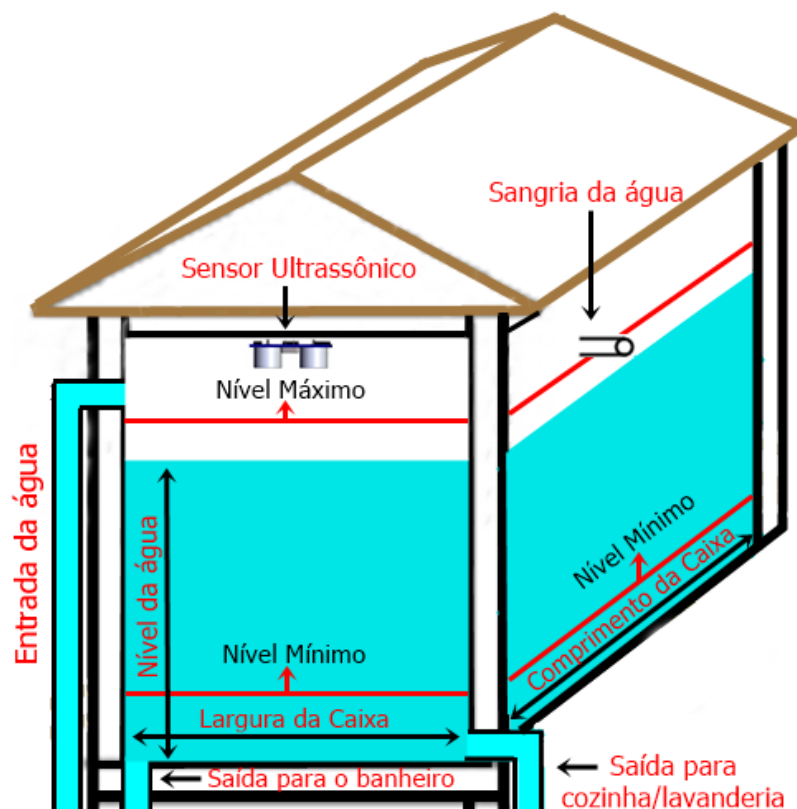
Para realizar esse trabalho será necessário o uso dos seguintes materiais:

- 1 *Raspberry Pi model 3 B+*;
- 1 Cartão SD de 64 Gb;
- 1 Mouse;
- 1 Teclado;
- 1 Sensor ultrassônico HC-SR04;
- 3 Sensores de fluxo YF-S201;

- 2 Módulos relés de dois canais;
- 3 Válvula solenoide;
- 1 Fonte reguladora de tensão;
- 1 Conversor de nível lógico;
- 2 *Protoboards*.

A residência possui uma caixa d'água construída de alvenaria com um volume de  $3m^3$ , contendo 1 metro de largura, 2 metros de comprimento e 1,5 metros de altura como ilustrado na Figura 18. Utilizou-se um *Raspberry Pi* para o desenvolvimento de um programa utilizando a linguagem em *python* para realizar a automação do sistema. A entrada da caixa d'água dispõe de um sensor de fluxo configurado para medir todo o consumo de água. Os valores lidos por esse sensor serão transferidos para um banco de dados e seu usuário, através da página *Web*, terá acesso a esses dados em tempo real.

Figura 18 – Planta da caixa d'água da residência.



Fonte: Arquivo Próprio, 2021.

A automação principal do projeto consiste em controlar o volume de água da caixa d'água por meio de uma válvula solenoide normalmente fechada na entrada e com o auxílio de um relé.

Será utilizado um sensor ultrassônico para calcular a distância entre o teto da caixa d'água e o nível de água, para posteriormente calcular o volume de água e enviar para o *ThingSpeak*. Esse dado pode ser visualizado também por meio da página *Web*.

Quando a caixa d'água atingir o nível mínimo de 20% da capacidade do reservatório, o relé acionará a válvula solenoide e o usuário receberá uma notificação no seu *e-mail* avisando que a válvula foi acionada para encher a caixa, e, quando o nível de água chegar ao valor máximo de 90% de sua capacidade, o relé desativará a válvula de entrada e novamente de forma automática o cliente será notificado instantaneamente em seu *e-mail* comunicando que a caixa atingiu seu volume máximo e a válvula foi desativada.

Existem duas saídas da caixa d'água para o abastecimento do imóvel, sendo uma das saídas para o banheiro e a outra para área da cozinha em conjunto com a lavanderia, sendo que ambas as saídas também tem válvula solenoide normalmente abertas, que podem ser controladas através do relé e monitorar o consumo de água usando um sensor de vazão de fluxo por saída.

Será realizada uma programação para que caso alguma torneira/registro permaneça ligada por um tempo superior a 15 minutos, a válvula daquele ambiente automaticamente feche e notifique pelo *e-mail* o proprietário da residência, evitando o desperdício de água da propriedade. A válvula poderá ser aberta novamente para passagem de água por meio de um botão, que ao ser pressionado deixará a válvula normalmente aberta novamente. Foi utilizado o tempo de 15 minutos, visto que segundo a SABESP, 2015, é um tempo suficiente para as principais atividades de uso de água para uma residência simples, como lavar louça, tomar banho e encher o reservatório de descarga.

## 5 TESTES, CALIBRAÇÃO E RESULTADOS

Nesse tópico será apresentado como foi desenvolvido o sistema de automação residencial que tem como objetivo a automação e monitoramento do controle de água de uma residência simples, além de descrever todos os testes e escolhas realizados no projeto. Será também demonstrada a descrição do projeto e um resumo dos resultados.

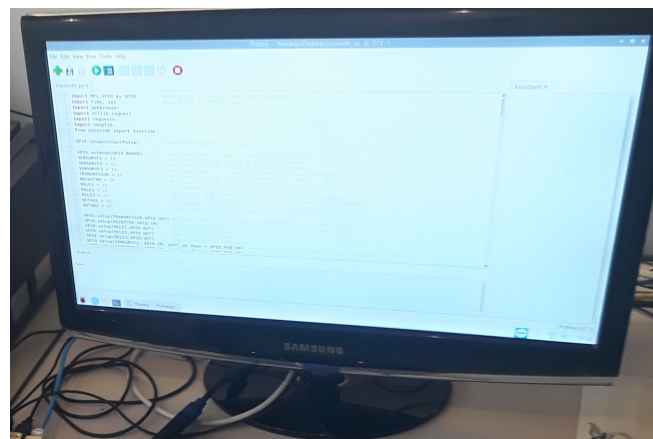
### 5.0.1 Escolha e Instalação do Sistema Operacional

No mercado existem diversos controladores capazes de realizar esse tipo de automação como é o exemplo do Arduino, ESP 32, microcontroladores da família PIC, Banana e entre outros. Todavia, optou-se por utilizar o *Raspberry Pi model 3 B+*, pois quando comparado com os outros, apresenta um grande poder de processamento e é capaz de operar na velocidade de até 1.4GHz, além disso, conta com 1 GB de memória RAM e também uma grande capacidade de armazenamento de informações, podendo utilizar um Cartão SD para armazenamento dos dados.

Primeiramente foi necessária a instalação do Sistema Operacional no *Raspberry Pi* e foi escolhido o SO *Raspbian* que é uma variante do Debian baseada no ARM *hard-float*, utilizando da arquitetura *Wheezy*. Por não possuir HD (*Hard Disk*), foi necessária a utilização de um cartão micro SD para gravação do sistema operacional. O cartão utilizado foi um micro SD de 64GB da classe 10 devido a sua maior velocidade na leitura e escrita, aumentando a velocidade de execução do SO.

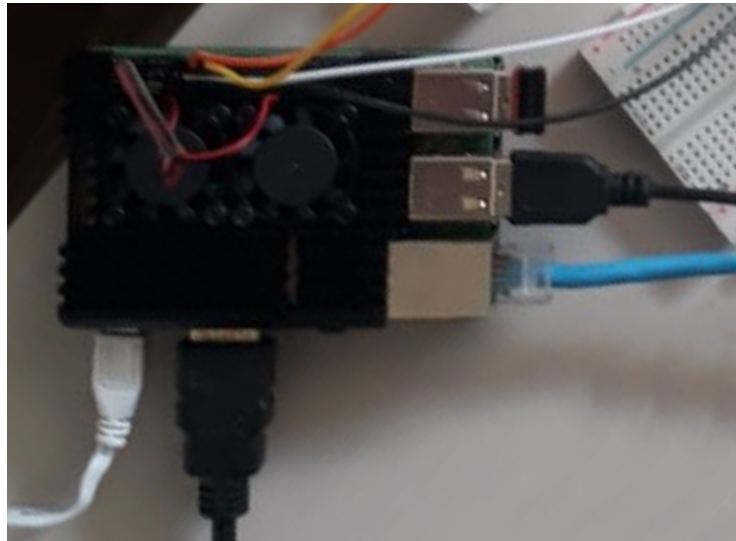
Foi necessário acessar o site oficial da fundação *Raspberry Pi*, e realizar o *download* da imagem do sistema operacional para realizar o processo de gravação no cartão micro SD, conforme descrito o passo a passo disponível pelo fabricante na página de *download*. Feito isso, conectou-se ao controlador o cartão, um mouse, um teclado, um monitor e uma fonte de tensão de 5 V e 2 A para fornecer potência a placa. Na Figura 19 ilustra o *Raspberry Pi* com seu sistema operacional *Raspbian* instalado.

Figura 19 – *Raspberry Pi* com seu sistema operacional instalado



Fonte: Arquivo Próprio, 2021.

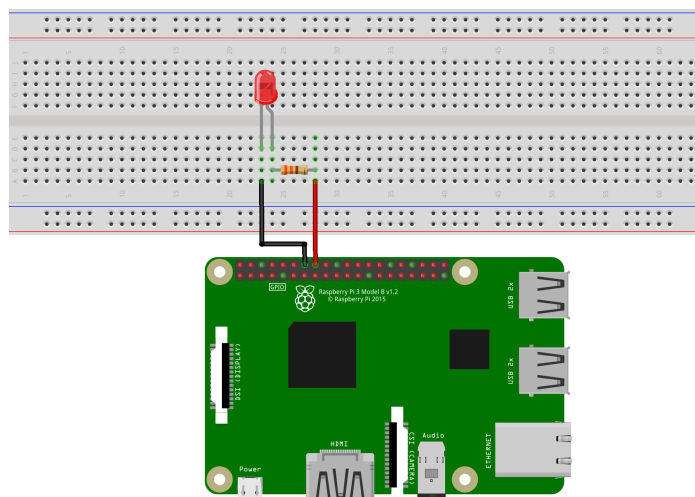
Na Figura 20 apresenta o *Raspberry Pi* com seus periféricos, após a instalação do SO, pronto para programar.

Figura 20 – *Raspberry Pi* com alguns periféricos conectados.

Fonte: Arquivo Próprio, 2021.

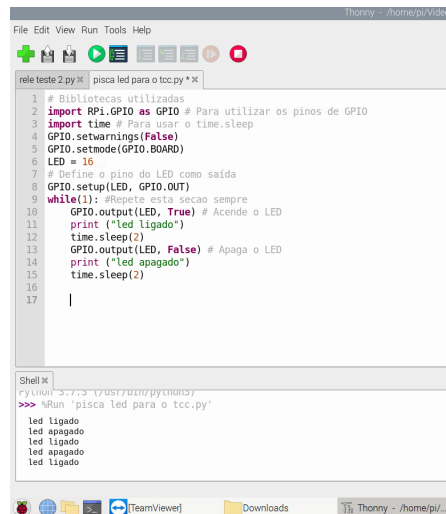
### 5.0.2 Teste para o funcionamento da GPIO do *Raspberry Pi*

Com o interesse de verificar os sensores e atuadores em funcionamento, montou-se um protótipo para cada sistema de medição. Para saber se a GPIO do *Raspberry Pi* estaria funcionando normalmente o primeiro teste foi montar um circuito simples apenas para visualizar o seu funcionamento, o sistema consiste em um circuito com um resistor de  $330\ \Omega$  em série para limitar a corrente de um LED, o ânodo foi ligado na porta GPIO 16 no formato BOARD enquanto o cátodo foi conectado na porta GPIO 14 e o *ground* como ilustrado a Figura 21.

Figura 21 – Circuito para acender um LED desenvolvido no *Software Fritzing*.

Fonte: Arquivo Próprio, 2021.

Em seguida foi desenvolvido um código em *python* para piscar o LED com um intervalo de tempo de 2 segundos como ilustra a Figura 22.

Figura 22 – Código em *python* para piscar um LED compilado no IDE *Thonny*.

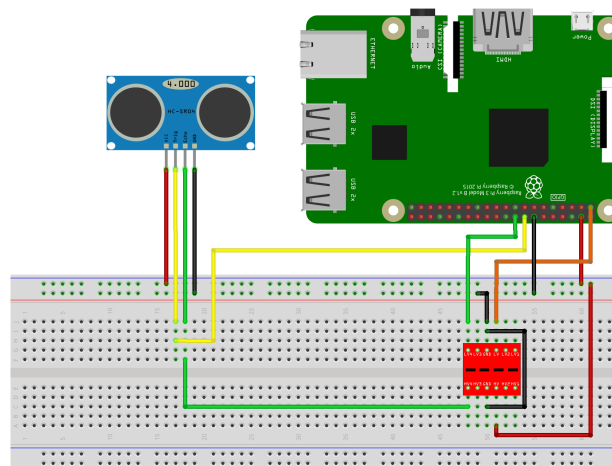
Fonte: Arquivo Próprio, 2021.

Como apresentado na Figura 24, utilizou-se a biblioteca *RPi.GPIO* para o controle das portas de entrada e saída e a biblioteca *time* para o controle de tempo. Após esse teste conclui-se um ótimo funcionamento no *Raspberry Pi*, conforme foi programado, enviando a informação de forma muito rápida e precisa de acordo com o teste.

### 5.0.3 Teste do Sensor Ultrassônico

Para verificar o funcionamento e a calibração do sensor ultrassônico HC-SR04, primeiramente foi realizada a montagem do circuito com o módulo conforme é apresentado no diagrama elétrico na Figura 23, utilizando o pino Trigger para o envio de onda ultrassônica e o Echo para enviar o dado ao *Raspberry Pi*. Foi necessária a utilização de conversor de nível lógico para converter a tensão do sinal de 5V para 3,3V para não danificar a GPIO do controlador.

Figura 23 – Diagrama elétrico de ligação para o sensor ultrassônico HC-SR04.

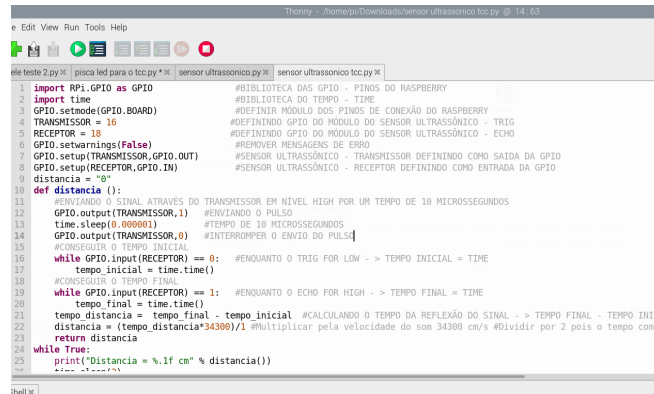


Fonte: Arquivo Próprio, 2021.



Em seguida desenvolveu-se um código em *Python* para utilização do sensor sendo capaz de medir a distância por meio da Equação 3.1. O código para o teste do sensor ultrassônico é apresentado na Figura 24, foi utilizada a função `time.time()` para calcular a diferença de tempo entre o envio e a detecção da onda pelo sensor.

Figura 24 – Código em *Python* para medir distância com um sensor ultrassônico HC-SR04.



```

1 import RPi.GPIO as GPIO #BIBLIOTECA DAS GPIO - PINOS DO RASPBERRY
2 import time #BIBLIOTECA DO TEMPO - TIME
3 GPIO.setmode(GPIO.BOARD) #DEFINIR MODO DOS PINOS DE CONEXÃO DO RASPBERRY
4 TRANSMISSOR = 16 #DEFININDO GPIO DO MÓDULO DO SENSOR ULTRASSÔNICO - TRIG
5 RECEPTOR = 18 #DEFININDO GPIO DO MÓDULO DO SENSOR ULTRASSÔNICO - ECHO
6 GPIO.setwarnings(False) #REMOVER MENSAGENS DE ERRO
7 GPIO.setup(TRANSMISSOR, GPIO.OUT) #SENSOR ULTRASSÔNICO - TRANSMISSOR DEFININDO COMO SAÍDA DA GPIO
8 GPIO.setup(RECEPTOR, GPIO.IN) #SENSOR ULTRASSÔNICO - RECEPTOR DEFININDO COMO ENTRADA DA GPIO
9 distancia = 0
10 def distancia():
11     #ENVIANDO O SINAL ATRAVÉS DO TRANSMISSOR EM NÍVEL HIGH POR UM TEMPO DE 10 MICROSSEGUNDOS
12     GPIO.output(TRANSMISSOR, 1) #ENVIANDO O PULSO
13     time.sleep(0.00001) #TEMPO DE 10 MICROSSEGUNDOS
14     GPIO.output(TRANSMISSOR, 0) #INTERROMPER O ENVIO DO PULSO
15     #RECEBER O TEMPO INICIAL
16     while GPIO.input(RECEPTOR) == 0: #ENQUANTO O TRIG FOR LOW -> TEMPO INICIAL = TIME
17         tempo_inicial = time.time()
18     #RECEBER O TEMPO FINAL
19     while GPIO.input(RECEPTOR) == 1: #ENQUANTO O ECHO FOR HIGH -> TEMPO FINAL = TIME
20         tempo_final = time.time()
21     tempo_distancia = tempo_final - tempo_inicial #CALCULANDO O TEMPO DA REFLEXÃO DO SINAL -> TEMPO FINAL - TEMPO INICIAL
22     distancia = (tempo_distancia*34300)/2 #Multiplicar pela velocidade do som 34300 cm/s #Dividir por 2 pois o tempo com
23     return distancia
24 while True:
25     print("Distancia = %.1f cm" % distancia())

```

Fonte: Arquivo Próprio, 2021.

Para realizar a calibração do sensor ultrassônico foi realizado um experimento em que o sensor foi colocado em um local fixo, e com auxílio de uma caixa e uma trena foram coletadas varias medições. Foram realizadas 10 medições de distância com a trena começando a partir de 10 *cm* variando a cada 15 *cm*, até chegar a 150 *cm*, utilizando a velocidade do som com o valor de 343 *m/s*. Os resultados do teste é ilustrado na Tabela. 5.

Tabela 5 – Teste com sensor ultrassônico com velocidade do som de 343*m/s*.

| Distância medida<br>com a trena | Distância calculada pelo o sensor<br>com a velocidade do som a 343 <i>m/s</i> |
|---------------------------------|-------------------------------------------------------------------------------|
| 10 cm                           | 9,98 cm                                                                       |
| 25 cm                           | 24,96 cm                                                                      |
| 40 cm                           | 39,91 cm                                                                      |
| 55 cm                           | 54,93 cm                                                                      |
| 70 cm                           | 69,89 cm                                                                      |
| 85 cm                           | 84,90 cm                                                                      |
| 90 cm                           | 89,84 cm                                                                      |
| 105 cm                          | 104,79 cm                                                                     |
| 120 cm                          | 119,67 cm                                                                     |
| 135 cm                          | 134,65 cm                                                                     |
| 150 cm                          | 149,61 cm                                                                     |

Fonte: Arquivo Próprio, 2021.

Após os resultados apresentado na Tabela 5 notou-se uma pequena variação nas distâncias medidas pelo sensor, sendo sempre o valor um pouco menor com o medido pela trena, por meio da Equação 3.1 os valores que variam são o tempo e a velocidade do som, então a calibragem do sensor foi realizado variando um pouco a velocidade e após vários testes conclui-se que a melhor

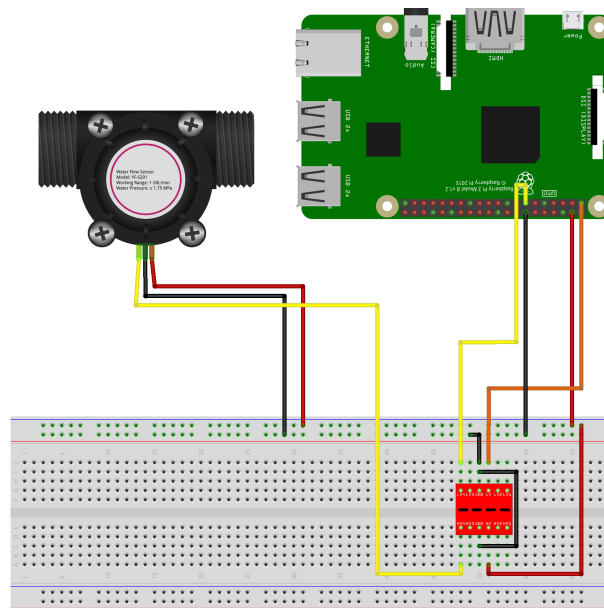
precisão do módulo se deu quando utilizou-se uma velocidade no valor de  $350\text{ m/s}$  mostrando um resultado muito confiável diante os testes realizados à uma temperatura média de  $30^{\circ}\text{C}$ .

#### 5.0.4 Teste do Sensor de Vazão

Também foram realizados testes utilizando o sensor de vazão YF S201 para entender melhor o seu funcionamento e também realizar a sua calibração. Esse sensor consiste em uma mini turbina onde o fluido gira uma hélice com um ímã em uma de suas partes, visto que quando a hélice gira, esse ímã interagem magneticamente com um sensor de efeito *Hall* criando um pulso de tensão que pode ser captada pelo controlador.

Primeiramente foi realizada a montagem do circuito utilizando sensor de vazão conforme apresentado a Figura 25. O sensor de efeito *Hall* gera o pulso de tensão de saída, tomando como referência a sua tensão de alimentação, logo não é possível conectar diretamente na GPIO do *Raspberry Pi*, dado que segundo o fabricante o nível de tensão para suas portas de *input* e *output* é  $3,3\text{V}$ . Para solucionar esse problema utilizou-se um conversor de nível lógico bidirecional que foi capaz de converter o sinal de  $5\text{V}$  para  $3,3\text{V}$  possibilitando que o *Raspberry Pi* realizasse a leitura de forma segura.

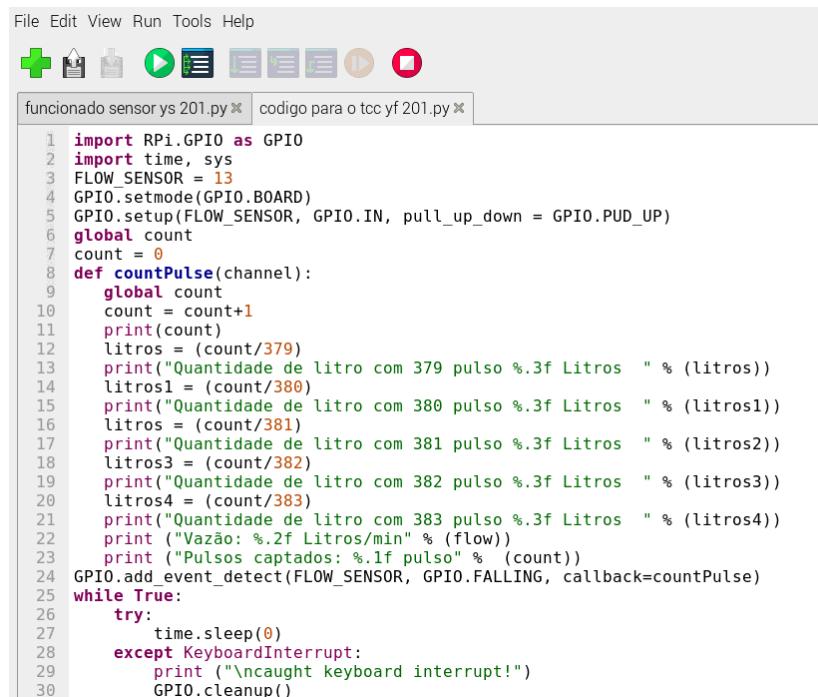
Figura 25 – Diagrama elétrico do sensor de fluxo.



Fonte: Arquivo Próprio, 2021.

Em seguida foi desenvolvido um código em *python* como ilustra a Figura 26 para testar o funcionamento do sensor, onde a programação consiste em realizar a contagem dos pulsos de sinais gerados pelo sensor de vazão quando acontece a passagem do fluido internamente. Diante disso, foi possível criar uma constante que relacionasse a quantidade de pulsos captados pela quantidade de água que atravessa o sensor.

Figura 26 – Código para calibrar o sensor de vazão YF S201.



```

1 import RPi.GPIO as GPIO
2 import time, sys
3 FLOW_SENSOR = 13
4 GPIO.setmode(GPIO.BOARD)
5 GPIO.setup(FLOW_SENSOR, GPIO.IN, pull_up_down = GPIO.PUD_UP)
6 global count
7 count = 0
8 def countPulse(channel):
9     global count
10    count = count+1
11    print(count)
12    litros = (count/379)
13    print("Quantidade de litro com 379 pulso %.3f Litros " % (litros))
14    litros1 = (count/380)
15    print("Quantidade de litro com 380 pulso %.3f Litros " % (litros1))
16    litros = (count/381)
17    print("Quantidade de litro com 381 pulso %.3f Litros " % (litros2))
18    litros3 = (count/382)
19    print("Quantidade de litro com 382 pulso %.3f Litros " % (litros3))
20    litros4 = (count/383)
21    print("Quantidade de litro com 383 pulso %.3f Litros " % (litros4))
22    print ("Vazão: %.2f Litros/min" % (flow))
23    print ("Pulsos captados: %.1f pulso" % (count))
24 GPIO.add_event_detect(FLOW_SENSOR, GPIO.FALLING, callback=countPulse)
25 while True:
26     try:
27         time.sleep(0)
28     except KeyboardInterrupt:
29         print ("\ncaught keyboard interrupt!")
30         GPIO.cleanup()

```

Fonte: Arquivo Próprio, 2021.

Para o desenvolvimento do código exibido na Figura 26, foi criada uma função para detectar a mudança de borba na GPIO do *Raspberry Pi*, em outras palavras, detectar o sinal PWM (*Pulse-Width Modulation*) gerado pelo sensor de vazão, sendo assim foi necessário utilizar os resistores internos do controlador deixando em *pull up*. Os resultados do teste estão ilustrados na Tabela 6.

Tabela 6 – Resultado dos testes de calibração do sensor de fluxo.

| Quantidade de litros (L) | Quantidade de pulsos |
|--------------------------|----------------------|
| 1                        | 378                  |
| 1                        | 379                  |
| 1                        | 379                  |
| 1                        | 381                  |
| 1                        | 383                  |
| 1                        | 376                  |
| 1                        | 377                  |

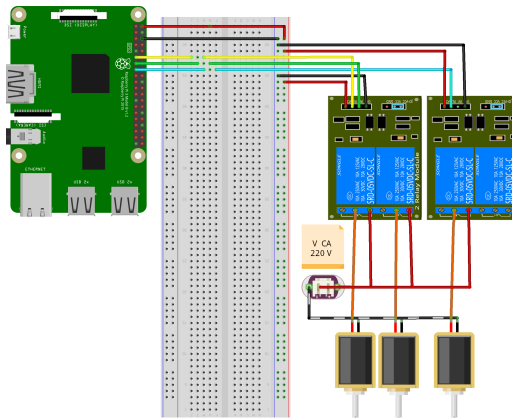
Fonte: Arquivo Próprio, 2021.

Através dos dados da Tabela 6 pode-se obter a relação entre a quantidade de pulsos gerada por litros, fazendo uma média aritmética da relação de cada teste e obtendo o valor de 379 pulsos gerados por litro.

### 5.0.5 Teste da Válvula Solenoide

Um teste foi feito para analisar e entender o funcionamento da válvula solenoide para realizar o controle de fluxo de água tanto da entrada quanto das saídas de uma caixa d'água. Para isso foi montado o esquema elétrico como apresentado na Figura 27, as válvulas solenoides serão ativadas e desativadas com o auxílio de dois módulos relés com dois canais cada. A tensão de operação da válvula solenoide é 220 V.

Figura 27 – Diagrama elétrico para o teste da válvula solenoide.



Fonte: Arquivo Próprio, 2021.

Utilizando a linguagem *python* desenvolveu-se um código para verificar o funcionamento da válvula solenoide como ilustra a Figura 28. Foi configurado para que os relés acionem, abrindo uma válvula e depois de aguardar 20 segundos o relé será desativado, fechando a válvula. O procedimento repetiu com as demais válvulas.

Figura 28 – Código para o teste da válvula solenoide.

```
Thonny - C:\Users\Gurgel Engenharia\Desktop\TCC ENGENHARIA ELETRICA\codigo da valvula solenoide.py ©
Arquivo  Editar  Visualizar  Executar  Ferramentas  Ajuda

iniciando (TI).py  código da valvula solenoide.py * 1X

1 import RPi.GPIO as GPIO #BIBLIOTECA DAS GPIO DO RASPBERRY PI
2 import time, sys        #BIBLIOTECA DO TEMPO - TIME
3
4 GPIO.setwarnings(False)
5 GPIO.setmode(GPIO.BOARD) #DEFINIR MODO DE OPERAÇÃO DAS GPIO
6 RELE1 = 11
7 RELE2 = 13
8 RELE3 = 15
9 GPIO.setup(RELE1,GPIO.OUT)
10 GPIO.setup(RELE2,GPIO.OUT)
11 GPIO.setup(RELE3,GPIO.OUT)
12 while (True):
13
14     GPIO.output(RELE1, True) #ATIVAR O RELÉ
15     print ("Relé 1 ativado, válvula energizada")
16     time.sleep(20)
17     GPIO.output(RELE1, False) #DESATIVAR O RELÉ
18     print ("Relé 1 desativado, válvula desenergizada")
19     time.sleep(20)
20     GPIO.output(RELE2, True) #ATIVAR O RELÉ
21     print ("Relé 2 ativado, válvula energizada")
22     time.sleep(20)
23     GPIO.output(RELE2, False) #DESATIVAR O RELÉ
24     print ("Relé 2 desativado, válvula desenergizada")
25     time.sleep(20)
26     GPIO.output(RELE3, True) #ATIVAR O RELÉ
27     print ("Relé 3 ativado, válvula energizada")
28     time.sleep(20)
29     GPIO.output(RELE3, False) #DESATIVAR O RELÉ
30     print ("Relé 3 desativado, válvula desenergizada")
31     time.sleep(20)
```

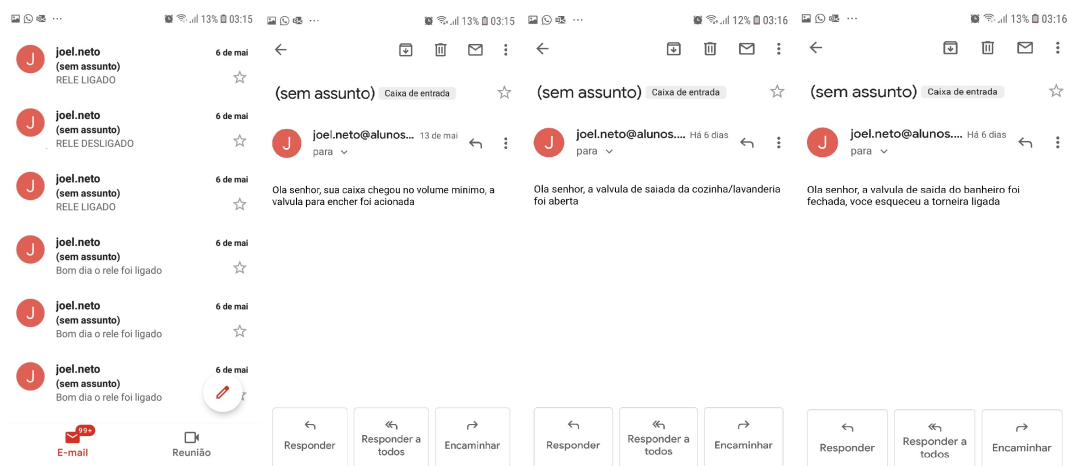
Fonte: Arquivo Próprio, 2021.

Por meio de um *loop* no código da Figura 28 foi possível realizar esse teste diversas vezes, demonstrando um ótimo funcionamento tanto dos módulos relés, quanto das válvulas solenoide. Na Figura 28 pode-se visualizar o código desse teste para os atuadores.

### 5.0.6 Envio de notificação por e-mail

Com o objetivo de notificar ao usuário quando algo importante acontece no sistema, utilizou-se de um protocolo familiar ao HTTP para realizar o envio dessas informações. Foi utilizado o protocolo SMTP (*Simple Mail Transfer Protocol*) para envio de *e-mail* que dita como o *e-mail* será formatado, criptografado e transmitido por meio dos servidores online acessado através do *Raspberry Pi*. Foi criado um algoritmo usando a biblioteca “smtplib” do *python3*, para realizar o envio de notificações da automação do imóvel ao usuário. Esse protocolo se mostrou muito eficaz, visto que nos testes realizados os envios foram feitos de forma instantânea do remetente ao destinatário, além disso, possibilita a utilização de uma ferramenta gratuita de envio. Na Figura 29 ilustra alguns dos *e-mails* recebidos do remetente de forma instantânea nos testes realizados.

Figura 29 – Notificações recebidas por *e-mail* do usuário.



Fonte: Arquivo Próprio, 2021.

Na Figura 29 apresenta algumas das notificações enviada do *Raspberry Pi* ao usuário, onde foi testado várias vezes enviando vários e-mail de forma simultânea e sendo sempre processadas e recebidas pelo destinatário.

### 5.0.7 Configuração do *ThingSpeak* e da Página *Web*

Primeiramente para utilizar a plataforma do *ThingSpeak* é necessário a criação de uma conta de forma gratuita com *e-mail* e *login* no site do desenvolvedor, e, após a verificação do *e-mail*, o usuário já recebe a licença gratuita de forma vitalícia com as limitações já apresentadas no referencial teórico, podendo utilizar das funcionalidades da plataforma imediatamente. Iniciando com a configuração da plataforma foi necessária a criação de um canal como mostrado na Figura 30 para o armazenamento e o monitoramento dos dados enviados pelo *Raspberry Pi* dos sensores e atuadores.

Figura 30 – Criando um canal para o envio dos dados para o *ThingSpeak*.

The screenshot shows the 'Meus Canais' (My Channels) page on the ThingSpeak website. At the top, there is a navigation bar with the ThingSpeak logo and links for 'Canais', 'Apps', and 'apoio'. Below the navigation bar, the title 'Meus Canais' is displayed. A green button labeled 'Novo Canal' is on the left, and a search bar labeled 'Pesquisar por tag' is on the right. The main content area shows a table of channels. The first channel is 'Sistema de Monitoramento de Água', created on 2021-02-21 and last updated on 2021-04-26 13:06. Below the channel name, there are tags: 'consumo, água, automação, framboesa'. There are also buttons for 'privado', 'público', 'Configurações', 'compartilhamento', and 'Chaves de API'. At the bottom of the channel entry, there is a button for 'Importação / Exportação de Dados'.

| nome                                                                                                                                                                        | Criado     | atualizado       |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|------------------|
| Sistema de Monitoramento de Água<br>consumo, água, automação, framboesa<br>privado público Configurações compartilhamento Chaves de API<br>Importação / Exportação de Dados | 2021-02-21 | 2021-04-26 13:06 |

Fonte: Arquivo Próprio, 2021.

Após a criação do canal Sistema de Monitoramento de Água, apresentada na Figura 30, foi realizada a configuração do canal com oito campos como ilustrado na Figura 31, representando respectivamente: consumo mensal de água da residência; nível da caixa d'água; consumo mensal do banheiro; consumo mensal da cozinha/lavanderia; volume da caixa d'água; estado da válvula de entrada; estado da válvula do banheiro e o estado da válvula da cozinha/lavanderia. Na sequência foram criados oito *widget* para melhor visualizar o *status* do nível lógico das três válvulas, nível da caixa d'água, volume da caixa d'água em litros e consumo geral em litros. Além disso, foi adicionado ao canal a localização do controlador que está enviando os dados para o servidor do banco de dados.

Figura 31 – Configurando o canal do *ThingSpeak*.

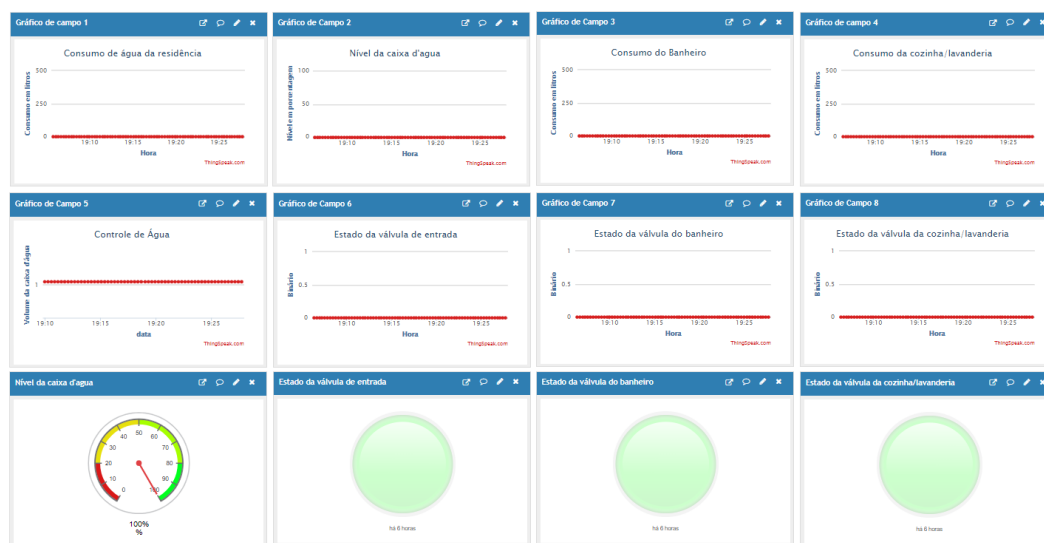
The screenshot shows the 'Configurações do canal' (Channel Settings) page for the 'Sistema de Monitoramento de Água' channel. The page has a navigation bar with the ThingSpeak logo and links for 'Canais', 'Apps', and 'apoio'. Below the navigation bar, there are tabs for 'Vista privada', 'Vista pública', 'Configurações do canal', and 'compartilhamento'. The 'Configurações do canal' tab is selected. The page title is 'Configurações do canal'. Below the title, there is a progress bar labeled 'Porcentagem completa' with a value of 70%. The 'ID do canal' is 1369346. The 'nome' field is 'Controle de Água'. The 'descrição' field is 'Projeto de automação do consumo de água de uma residência'. There are eight 'Campo' (Field) settings, each with a name and a checkbox: 'Campo 1' (Consumo de água da r), 'Campo 2' (Nível da caixa d'água), 'Campo 3' (Consumo do Consum), 'Campo 4' (Consumo da cozinha), 'Campo 5' (Volume da caixa d'água), 'Campo 6' (Estado da válvula de e), 'Campo 7' (Estado da válvula do b), and 'Campo 8' (Estado da válvula da c). All checkboxes are checked. There is a 'metadados' (Metadata) field and a 'Tags' field with the value 'consumo, água, automação, raspberry'. A note at the bottom says '(As tags são separadas em vírgula)'.

Fonte: Arquivo Próprio, 2021.

Após finalizar a configuração do canal, foi desenvolvido um código em *python* com as bibliotecas necessárias para realizar a comunicação com *ThingSpeak*, onde serão enviados oito dados, um para cada campo de forma simultânea, o envio é realizado a através da chave de escrita fornecida pela plataforma na aba Chave de API.

Em sequência foram realizados vários testes, enviando os dados do *Raspberry Pi* ao banco de dados, demonstrando uma ótima confiabilidade nos resultados, podendo ser visualizados nos campos e *widget*, assim como foi configurado e programado no projeto. Na Figura 32 é ilustrado uma prévia dos gráficos com os valores recebidos pelo controlador.

Figura 32 – Dados enviado do *Raspberry Pi* ao *ThingSpeak*.



Fonte: Arquivo Próprio, 2021.

Para melhorar o monitoramento por parte do usuário, foi desenvolvida uma página *Web* através da linguagem de marcação HTML, a qual o código é apresentado no Apêndice 2. Foi feita a transferência dos dados do *ThingSpeak* que foi recebido através do *Raspberry Pi*, dados esses que foram captados pelos sensores e atuadores do sistema. Também foi configurada para visualizar a localização do controlador através dessa página *Web*. Com isso foi possível visualizar de forma bastante intuitiva e prática o monitoramento hidráulico da residência. Entretanto, essa página web só pode ser acessada através do *desktop* do qual foi desenvolvido o código, visto que a página *Web* não está hospedada em nenhum servidor online.

Para isso foi necessário realizar a hospedagem dessa página *Web* em um servidor na nuvem através do uso da plataforma do *InfinityFree*, que consiste em uma iniciativa independente dedicada a fornecer serviços de hospedagem gratuitos e confiáveis. Após a inscrição na plataforma foram realizados vários passos até ser gerado um domínio para o código HTML, criando assim a página *Web* sob o seguinte link de domínio: [monitoriamentoraspberryiot.epizy.com](http://monitoriamentoraspberryiot.epizy.com).



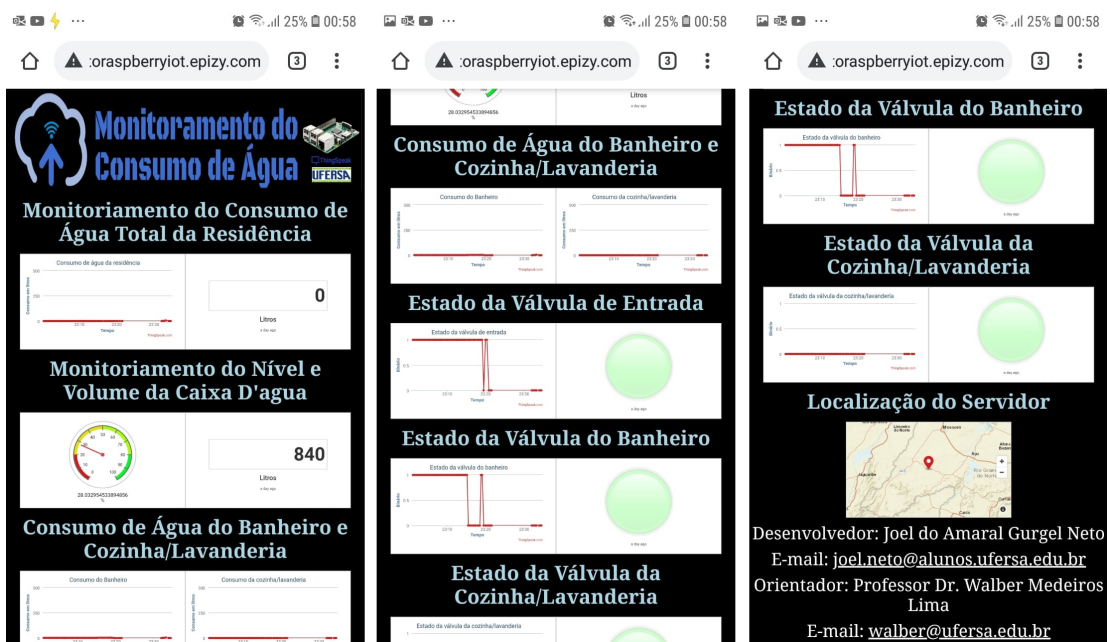
Foi possível obter o acesso ao monitoramento da automação residencial de forma bastante simples, utilizando apenas o link do domínio e acessar de qualquer dispositivo com acesso à internet de forma instantânea e gratuita. A Figura 33 ilustra o site sendo acessado através de um computador e a Figura 34 mostra o acesso realizado utilizando um *smartphone*.

Figura 33 – Página Web acessada pelo computador.



Fonte: Arquivo Próprio, 2021.

Figura 34 – Página Web acessada pelo o *smartphone*.



Fonte: Arquivo Próprio, 2021.

Tem-se que tanto a Figura 33 quanto a Figura 34, demonstram de forma bastante clara os dados gerados pelo os sensores e o *status* dos atuadores, esses dados exibidos nas duas Figuras servem para mostrar o funcionamento do sistema, enviados durante os testes realizados na elaboração do projeto.



### 5.0.8 Análise de Custos para o Sistema de Automação Residencial

Foi realizada uma análise de custo total para instalação do sistema de automação conforme uma pesquisa realizada na internet com as principais lojas online no mercado nacional. Um preço médio por cada componente utilizado no projeto totalizou um valor equivalente à 716,48 reais, valor esse que diante os benefícios ofertados pela automação apresentou um bom custo benefício para o proprietário da residência. Na Tabela 7 são exibidos detalhadamente os materiais necessários para o sistema e os valores. Com base em pesquisa em sites internacionais, foi visto também que esse custo pode reduzir cerca de 40% do custo quando comparado com mercado nacional.

Tabela 7 – Custo do projeto em mercado nacional.

| Produtos                    | Quantidade | Preço unitário | Preço total |
|-----------------------------|------------|----------------|-------------|
| Raspberry Pi model 3+       | 1          | R\$ 329,90     | R\$ 329,90  |
| Cartão SD de 64 Gb          | 1          | R\$ 64,90      | R\$ 64,90   |
| Sensor ultrassônico HC-SR04 | 1          | R\$ 11,29      | R\$ 11,29   |
| Sensor de fluxo YF S201     | 3          | R\$ 27,90      | R\$ 83,70   |
| Módulo relé de dois canais  | 2          | R\$ 14,90      | R\$ 29,80   |
| Válvula solenoide NA        | 2          | 59,90          | R\$ 119,80  |
| Válvula solenoide NF        | 1          | 49,90          | R\$ 49,90   |
| Fonte reguladora de tensão  | 1          | R\$ 9,90       | R\$ 9,90    |
| Conversor de nível lógico   | 1          | R\$ 5,59       | R\$ 5,59    |
| Chave tátil                 | 2          | R\$ 0,90       | R\$ 1,80    |
| Protoboards                 | 2          | R\$ 9,90       | R\$ 9,90    |
| Custo total                 | -          | -              | R\$ 716,48  |

Fonte: Arquivo Próprio, 2021.

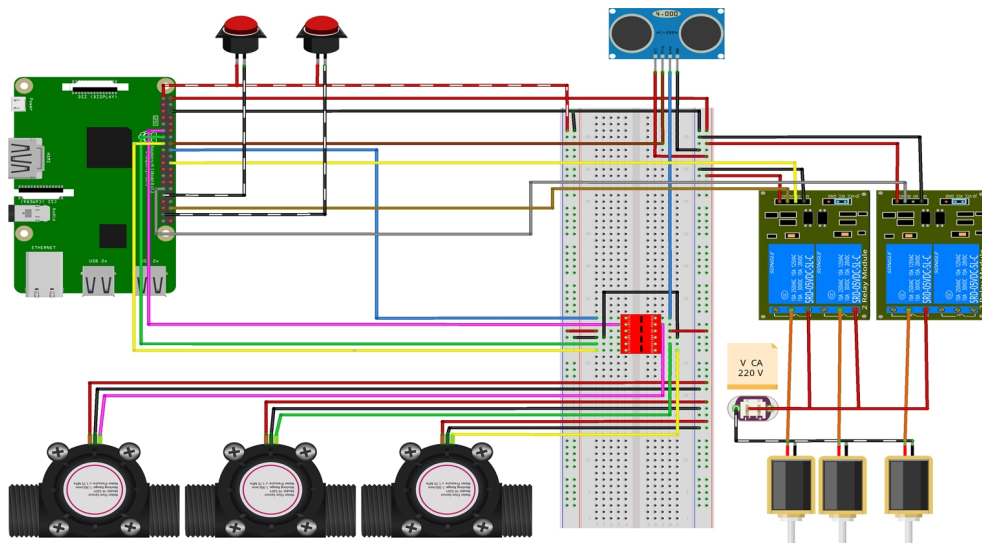
Como ilustrado na Tabela 7, o dispositivo com o maior valor é *Raspberry Pi*, contudo esse computador tem um poder computacional suficiente para um projeto mais complexo, visto que, além de bom processador, ele dispõe de 40 GPIO e para um projeto cuja residência tenha mais cômodos, com a compra de mais alguns sensores e atuadores e uma simples modificação na programação, pode-se aumentar o sistema de automação sem aumentar de forma significativa seu valor, gerando um melhor custo benefício ao usuário.

### 5.0.9 Resumo dos resultados

Após todos os testes e configurações, foi desenvolvido um código em Python apresentado na Apêndice 1, contendo todos os parâmetros e objetivos do projeto para automação e monitoramento do sistema hidráulico da residência, ao qual o sistema foi exposto a vários testes que obtiveram resultados bem sucedidos.

O diagrama elétrico completo para conexão dos sensores e atuadores do projeto da automação é exibida na Figura 35.

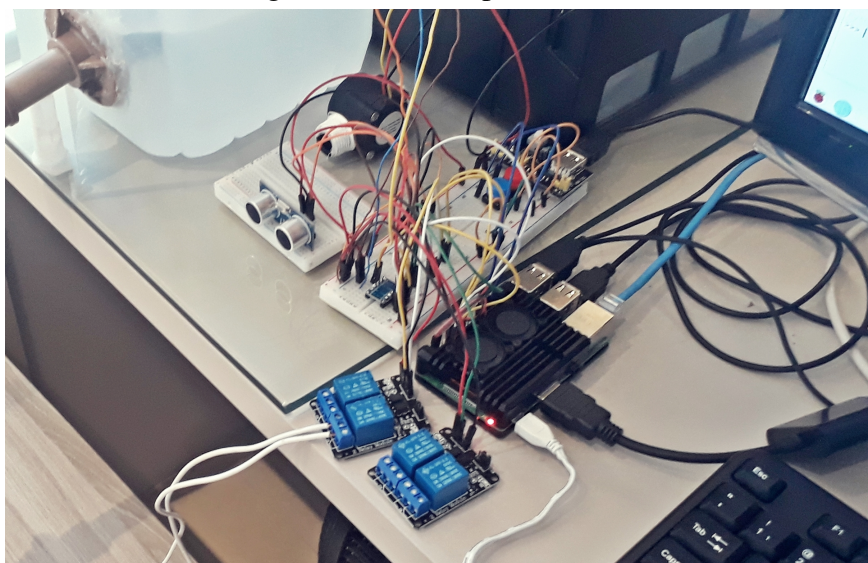
Figura 35 – Diagrama elétrico do projeto.



Fonte: Arquivo Próprio, 2021.

Na Figura 35 mostra a alimentação dos sensores e atuadores conectado ao *Raspberry Pi*, entretanto devido a grande quantidade de dispositivos conectado, utilizou-se de uma fonte reguladora de tensão de 5V e 3,3V, com uma capacidade de corrente de 700 mA para alimentação desses dispositivos, protegendo as GPIO do controlador contra sobrecarga, como ilustra a Figura 36, apresenta o circuito do projeto montado.

Figura 36 – Protótipo montado.



Fonte: Arquivo Próprio, 2021.

Na Figura 36 é apresentado o circuito do projeto concluído, com seus sensores e atuadores, onde foi analisado o funcionamento para a obtenção dos resultados deste trabalho na prática, nota-se também o uso de um *case* com dois *coolers* para um melhor resfriamento dos componentes eletrônicos da placa.

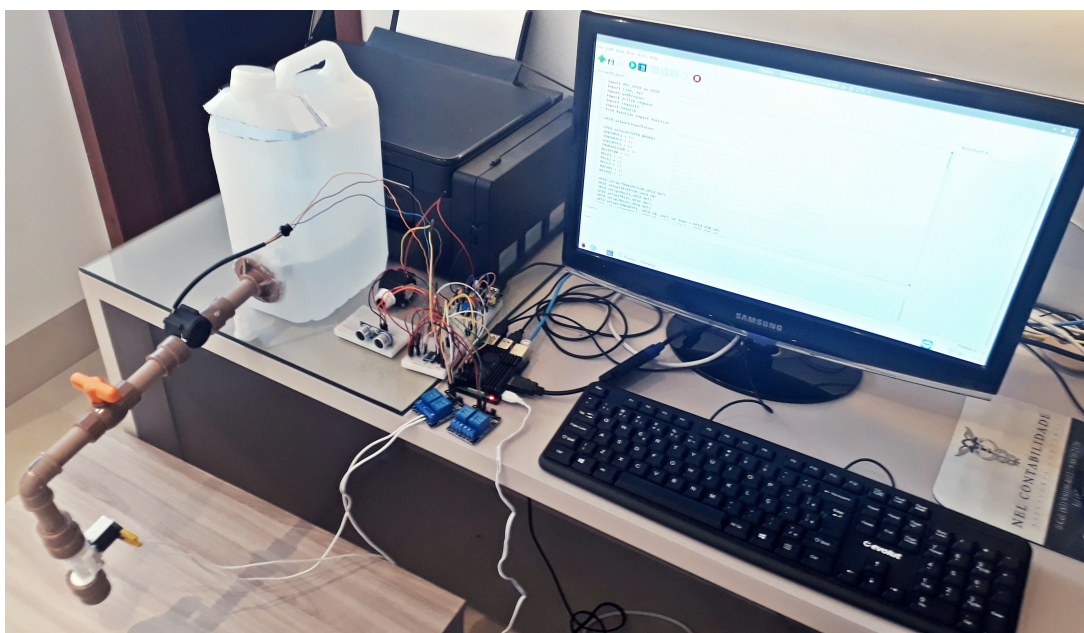
Diante a finalização do projeto de automação de residencial, o sistema mostrou-se muito eficaz e confiável, melhorando o conforto do usuário diariamente com a automação. O sensor ultrassônico HC-SR04 se mostrou bastante eficaz para essa aplicação, entretanto sugere-se o uso do sensor ultrassônico modelo JSN-SR04T, visto que é aprova d'água e tem uma maior vida útil devido ao ambiente exposto. O sensor vazão de fluido se mostrou também um ótimo funcionamento na leitura de água, estando conforme as especificações estipuladas pelo fabricante, assim como o acionamento dos relés e das válvulas solenoide.

A biblioteca utilizada em *Python* para enviar as notificações ao usuário via *e-mail* demonstrou-se como sendo uma ótima ferramenta, visto que em todos os testes com automação residencial completa, conseguiu atender a necessidade do projeto fazendo o envio de forma instantânea ao destinatário.

Quanto a aplicação do banco de dados em nuvem, temos que o *ThingSpeak* demonstrou ser uma ótima plataforma gratuita tanto para armazenar quanto exibir os dados fornecidos pelos sensores e atuadores. O serviço de hospedagem *Infinity Free* mostrou ser uma ferramenta bastante útil e satisfatória que, além de ser gratuita, tem um rápido processamento no carregamento da página *Web*.

A Figura 37 apresenta o circuito do sistema de automação residencial montado em uma *protoboard* sendo utilizado para comprovar o trabalho.

Figura 37 – Projeto completo.



Fonte: Arquivo Próprio, 2021.

A Figura 37 além de apresentar todo o sistema, também ilustra seus periféricos, como: mouse, teclado, monitor e outros. Entretanto, muito desde testes foram realizados de forma remota, utilizando o *software TeamViewer* para obter o acesso ao *Raspberry Pi*, sem a necessidade do uso desses periféricos.

Os resultados do trabalho foram muito satisfatórios, dado que foi possível realizar a simulação de um projeto para o monitoramento d'água através de um sistema de automação residencial, utilizando o computador *Raspberry Pi 3 model b+* conforme os objetivos específicos do projeto, desenvolvendo uma automação que pode facilitar a vida do usuário dando um maior conforto, além de fazer uma redução no desperdício de água da residência reduzindo assim a tarifa mensal do imóvel.

## 6 CONCLUSÃO

O presente trabalho teve como objetivo desenvolver um sistema de automação residencial capaz de medir o consumo de água de uma residência e o controle da entrada e saída de uma caixa d'água com o seu monitoramento de forma remota utilizando um computador de baixo custo. O *Raspberry Pi* mostrou ser muito eficiente na utilização de sistema de automação e IoT devido ao seu sistema operacional, que apresenta vantagens na utilização quando comparado com outros controlados, além disso, tem um grande benefício, pois vem com o *hardware* de *Wi-Fi* e *Bluetooth* integrados em sua placa facilitando a conexão com dispositivos IoT. Através da linguagem de programação *python* foi possível o desenvolvimento de um algoritmo capaz de automatizar uma residência com auxílio de dispositivos como sensores e atuadores.

Com o uso da plataforma *ThingSpeak* e a linguagem *python* foi possível realizar a configuração para visualização dos dados coletados pelo *Raspberry Pi* em um banco de dados. Com a criação da página *Web* utilizando a linguagem HTML, foi possível o acesso remoto de maneira bastante simples do monitoramento desses dados coletados pelo sistema. Para os testes realizados para medição dos sensores foram obtidos resultados muito satisfatórios, tendo em vista a precisão quando comparado com seu valor real, demonstrando uma grande confiabilidade dos resultados do trabalho. Além disso, o envio das notificações por *e-mail* utilizando a biblioteca “*smtplib*” mostrou uma grande confiabilidade no envio, apresentando uma grande velocidade no tempo de transferência das notificações do remetente ao destinatário.

Portanto, ao longo prazo, além de uma redução no consumo de água da residência devido aos possíveis vazamentos ou esquecimento de torneiras ligadas, será gerado um maior conforto ao usuário, visto que o mesmo pode monitorar o volume da caixa d'água em tempo real, além disso, é notificado sempre quando há alguma mudança de estado das válvulas. Logo, conclui-se que o trabalho foi bastante satisfatório, dado que pode-se aplicar os conhecimentos adquiridos durante o curso de Engenharia Elétrica conseguindo desenvolver todos os objetivos apresentados no trabalho.

## 7 BIBLIOGRAFIA

AGUIAR, Juscelino. **Curso de manutenção eletrônica analógica**. biblioteca24horas, 2012.

ALDOULAH, Zaid. **Aplicação de radares ultrassônicos em um sistema de detecção de distância em tempo real para indivíduos com deficiência visual**. Tese de Doutorado. Universidade de Toledo. 2017.

ANTONACCI, Meire Juliana et al. NCL. **Uma Linguagem Declarativa para Especificação de Documentos Hipermídia na Web**. VI Simpósio Brasileiro de Sistemas Multimídia e Hipermídia-SBMídia2000, Natal, Rio Grande do Norte, 2000.

ATZINGEN, Gustavo Voltani von. **Simulação, controle e automação de um forno tipo túnel utilizando tecnologia embarcada**. 2017. Tese de Doutorado. Universidade de São Paulo.

BACHRUDIN, Zainal; WIDODO, Catur Edi; ADI, Kusworo. - **Simulator input-output sistem kontrol menggunakan Raspberry Pi**. *Youngster Physics Journal*. v. 6, n. 3, p. 272-279, 2017.

BOLZANI, Caio. **Desmistificando a domótica** – Escola Politécnica da Universidade de São Paulo, 2010.

BRAGA, Newton C. **Relés: Circuitos e aplicações**. Editora Newton C. Braga, 2017.

CAMPANHA, Caroline de Souza. **Aplicação web para programação visual usando Raspberry Pi**. 2017. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná. 2017.

CARVALHO, Osvaldo. Departamento de ciência da computação. **Introdução à Organização e à Programação de Computadores - Usando Logisim e Scilab**. UFMG. 2014.

CASSIOLATO, César. **Sensor Hall – A Tecnologia dos Posicionadores Inteligentes de Última Geração**. Revista Controle Instrumentação, Curitiba (PR), Edição nº 81, Junho 2003.

COSTA, Carlos J. **Desenvolvimento para web**. HTML press/Lusocredito, 2007.

DOHR, A. et al. G. Schreier (2010). **The internet of things for ambient assisted living**. In: Seventh International Conference on Information Technology.

DORNELAS, Everton; CAMPELLO, Sérgio. **Monitoramento de consumo doméstico de água utilizando uma meta-plataforma de IoT.** Revista de Engenharia e Pesquisa Aplicada, v. 2, n. 2, 2017.

EBERMAM, Elivelto et al. **Programação para leigos com Raspberry Pi.** 2017.

Electrofun. **Curso- Raspberry Pi** (2018). Acesso em 20 de Fev de 2021, disponível em <https://www.electrofun.pt/blog/curso-raspberry-pi-2-o-que-e-o-raspberry-pi/>.

FERNANDES, Pedro de Carvalho. **Projeto, Construção E Implementação de Uma Válvula Proporcional Para Controle de Nível em um Tanque.** Rio de Janeiro. 2018.

GOLDENBERG. Figura iot. **The Internet of Thing's Customer Service Challenges and Big Data impact (ismguide.com).** 2017. Disponível em: < <https://ismguide.com/the-internet-of-things-customer-service-challenges/> > Acesso em: 07, março e 2021.

GURGEL, Murilo Almeida do Amaral - **Avaliação das tecnologias de Automação Residencial e Implementação de um sistema usando Z-Wave.** 2016.

HAFNER, Ana Vreni. **"Conservação e Reuso de Água em Edificações"** – experiências nacionais e internacionais [Rio de Janeiro] 2007 XVI, 161 p. (COPPE/UFRJ, M.Sc., Engenharia Civil). Dissertação – Universidade Federal do Rio de Janeiro. 2017.

JAHANKIT, Electronic. JSN-SR04T-2.0. **Datasheet.** 2019. Disponível em: < <https://www.maker-guides.com/wp-content/uploads/2019/02/JSNSR04T-Datasheet.pdf> > Acesso em: 10, março e 2021.

LOBO, Ricardo Jorge Gonçalves da Fonseca. **PenTest4All–Sistema automatizado de análise à segurança informática de uma rede.** 2019. Tese de Doutorado.

LIMA, Anderson Alves de. **Efeitos da torção em matéria condensada.** 2017. 65 f. Tese (Doutorado em Física) - Universidade Federal da Paraíba, João Pessoa, 2017.

MAGRANI, Eduardo. **A internet das coisas.** Editora FGV, 2018.

MATOS, Gustavo Batista de Castro; MACIEL, Melquíades Rodrigues. **Telemetria para sistemas hídricos.** 2019.

MERCES, Ricardo. **Raspberry Pi: Conceito e Prática**. Ricardo Mercês editora. 2013.. 2ª edição revista e ampliada, 2013.

MINUCCI, Amanda da Silva; LIMA, Fernando Aparecido dos Santos. **Automatização no processo de leitura de hidrômetros**. 2013. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná.

OLIVEIRA, Adailton Medeiros Rodrigues DE; CAVALCANTI, Gustavo Oliveira. **Sistema Automatizado de Controle de Abasteci-mento de Água Proveniente de Poços Artesianos com Monitoramento Remoto**. Revista de Engenharia e Pesquisa Aplicada, v. 2, n. 2, 2017.

OLIVEIRA, André Barros de Mello Oliveira. "**Diagnóstico De Falhas De Curto-Circuito Nos Interruptores De Um Vsi Monofásico Em Ponte Baseado Na Teoria De Conjuntos Aproximados**". Tese de Mestrado. Universidade Federal de Itajubá – UFI/PPGEE, Itajubá – MG, 2017.

OLIVEIRA, Botto DE, Caio Vinicius; NEVES, Filipe Eduardo; DE CAMARGO, Rafael Franco. **Projeto de uma Prensa Hidráulica: Dimensionamento e Seleção dos Componentes**. 2016.

OLIVEIRA, Isabella Ferreira de. **Desenvolvimento de um sistema de automação residencial baseado em IoT para controle e monitoramento de dispositivos elétricos**. 2019.

OLIVEIRA, Sérgio de. **Internet das coisas com ESP8266, Arduino e Raspberry PI**. Novatec Editora, 2017.

PATSKO, Luís Fernando. Tutorial: **Aplicações, Funcionamento e utilização de sensores**. 2006.

PINTO, Fábio da Costa. SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL (SENAI). **Sistemas de Automação e Controle**. Apostila elaborada pelo Professor de Instrumentação/Automação Industrial. 2005.

PIMENTEL JUNIOR, Jorge Luiz. **Efeito Hall extraordinário no supercondutor magnético . RuSr/sub 2/GdCu/sub 2/O/sub 8**. 2007.

PRUDÊNCIO FILHO, Clovis José. **Hardware simulador de detecção a elementos radioativos para uso em simulação de resposta à emergência radiológica**. 2020.



RAKSHIT, Bhandare, Dr; PANSE, M. S **System Development for Verification of General Purpose Input Output**. 2017.

RAMEZ, E.; NAVATHE, S. B. **Sistemas de banco de dados**. São Paulo: Pearson Addison Wesley, 2005.

ROGGIA, Leandro; FUENTES, Rodrigo Cardozo. **Automação industrial**. Santa Maria: E-tec Brasil, 2016.

Raspberry Foundation. (2006). **Raspberry Pi**. Acesso em 20 de Fev de 2021, Disponível em Raspberry Pi: [www.raspberrypi.org](http://www.raspberrypi.org).

Raspberry Foundation. (2018). **Raspberry Pi**. Acesso em 22 de Fev de 2021, Disponível em Raspberry Pi: [www.raspberrypi.org](http://www.raspberrypi.org).

ROBOCORE. **Atuador e Relé Válvula Solenoide**. 2021. Disponível em: < <https://www.robocore.net/atuador-rele/valvula-solenoide> > Acesso em: 04 abril de 2021.

SABESP. Meio Ambiente. 2015. Disponível em: <<http://site.sabesp.com.br/site/interna/Default.aspx?secaoId=595>> Acesso em: 1, março e 2021.

SAN, Kaung Nyunt. **Implementation of IoT-based Home Control and Monitoring System using Raspberry Pi and NodeMCUs** Tese de Doutorado. University of Computer Studies, Yangon. 2019.

SANASA. SOCIEDADE DE ABASTECIMENTO DE ÁGUA E SANEAMENTO S/A-SANASA. **Controle de Perdas de Água na Distribuição e Faturamento**. 2006. Disponível em:. Acesso em: 10 de Janeiro 2021.

SANTAELLA, Mauricio L. Santaella; PAULA, Fábio Nonato de Paula; TEIXEIRA, Maria Carolina Ferreira B. Teixeira; ALMEIDA, Pedro Henrique A. de Almeida. **Quantização do Desperdício de Água em Vazamentos Representados por Gotejamentos e Filetes de Água**. Revista Ciências do Ambiente On-Line Fevereiro, 2007 Volume 3, Número 1.

SILVA, Anderson Ferreira. **Sistema Operacional Linux** – 1. ed. – Brasília: NT Editora, 2017.

SANTOS, JOSE J. X. DOS. Eletrônicos. **Eletronline Shop**. 2021. Disponível em: < <https://www.eletronlineshop.com.br/> > Acesso em: 20, abril e 2021.

SANTOS, Wilka Kally Tavares dos. **Exemplos de utilização do raspberry pi para auxílio no ensino fundamental**. 2018.

SILVA, Cássio Moreira et al. **Sistema de controle de apresentação por meio de Raspberry Pi**. 2019.

SILVA FILHO, Bernardo Severo da. **Curso de Controladores Lógicos Programáveis**. LEEUERJ, Rio de Janeiro, RJ, 1998.

SILVA, J. A. F. da, LAGO, C. L. do. **Módulo eletrônico de controle para válvulas solenóides**. Química Nova, 25(5), 842–843, 2002.

SILVA, Luana Maria Menezes da. **Desenvolvimento de sistema de monitoramento IoT para rios urbanos de fácil implantação e baixo custo**. 2020.

SILVA, Matheus Hoffmann. **Sistema de Automação Residencial baseado em Raspberry Pi e Open ZWave**. 2014.

SILVA, Mauricio Samy. **HTML5: a linguagem de marcação que revolucionou a web**. Novatec Editora, 2019.

SOUZA, Andrey Fürst de; NUNES, Gláuber Muzyka Oyarzabal; BIANCHINI, Thiago Hoppen. **Sistema supervisorio para monitoramento de consumo de água**. 2016. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná.

TAVARES ZANUZZO, Lucas. **Sistema de monitoramento do consumo de água controlado por um aplicativo Android**. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) Universidade de São Paulo -Escola de Engenharia de São Carlos, São Carlos, 2017.

THINGSPEAK. **ThingSpeak for IoT**. 2021. Disponível em: < <https://thingspeak.com/pages/commerciallearnmore> > Acesso em: 02 abril de 2021.

THOMAZINI, Daniel, e ALBUQUERQUE, Pedro Urbano. **Sensores Industriais: Fundamentos e Aplicações**. Editora Érica, SãoPaulo, 2005.

TORRES, Victor Monteiro. **HTML e seus Componentes**. Revista Ada Lovelace, v. 2, p. 99-101, 2018.

TUCCI, Carlos E. M. **Existe crise da Água no Brasil?**. Instituto de Pesquisas Hidráulicas Universidade Federal do Rio Grande do Sul, 2017.

UPTON, Eben; HALFACREE. **Gareth. Raspberry Pi Manual do Usuário**. 1. ed. São Paulo: Novatec Editora Ltda, 2013.

WENDLING, Marcelo. Sensores. **Universidade Estadual Paulista. São Paulo**, v. 2010, p. 20, 2010.

WOJCIECHOWSKI, J., FLAUSINO, A. D. A., RODRIGUES, J. L. F., Luz, M. J. D., Hey, M. T. **WM–water monitoring**. 2018.

## **Apêndices**

## APÊNDICE A – CÓDIGO DO SISTEMA PARA O CONTROLE DE UMA CAIXA D'ÁGUA DE UMA RESIDÊNCIA

```

import RPi.GPIO as GPIO #BIBLIOTECA DAS GPIO - PINOS DO RASPBERRY
import time, sys        #BIBLIOTECA DO TEMPO - TIME
import webbrowser
import urllib.request
import requests
import smtplib
from datetime import datetime

GPIO.setwarnings(False) #REMOVER MENSAGENS DE ERROS
GPIO.setmode(GPIO.BOARD) #MODO DE OPERAÇÃO DA GPIO

SENSORYF1 = 11          #DEF. GPIO DO SENSOR DE VAZAO DE ENTRADA
SENSORYF2 = 13          #DEF. GPIO DO SENSOR DE VAZAO DO BANHEIRO
SENSORYF3 = 15          #DEF. GPIO DO SENSOR DE VAZAO DA COZ/LAVANDERIA
TRANSMISSOR = 16        #DEF. GPIO DO SENSOR ULTRASSÔNICO - TRIG
RECEPTOR = 18         #DEF. GPIO DO SENSOR ULTRASSÔNICO - ECHO
RELE1 = 36              #DEF. GPIO DO RELÉ DA VALVULA DE ENTRADA
RELE2 = 22              #DEF. GPIO DO RELÉ DA VALVULA DE SAIDA P/ O BANHEIRO
RELE3 = 29              #DEF. GPIO DO RELÉ DA VALVULA DE SAIDA P/ COZ/LAVANDERIA
BOTA01 = 31             #DEF. GPIO DO BOTAO P/ ABRIR A VALVULA DO BANHEIRO
BOTA02 = 37             #DEF. GPIO DO BOTAO P/ ABRIR A VALVULA DA COZ/LAVANDERIA

GPIO.setup(TRANSMISSOR,GPIO.OUT) #TRANSMISSOR DEFININDO COMO SAIDA DA GPIO
GPIO.setup(RECEPTOR,GPIO.IN)   #RECEPTOR DEFININDO COMO ENTRADA DA GPIO
GPIO.setup(RELE1,GPIO.OUT)       #RELE1 DEFININDO COMO SAIDA
GPIO.setup(RELE2,GPIO.OUT)       #RELE2 DEFININDO COMO SAIDA
GPIO.setup(RELE3,GPIO.OUT)       #RELE3 DEFININDO COMO SAIDA
GPIO.setup(SENSORYF1, GPIO.IN, pull_up_down = GPIO.PUD_UP) #SENSORYF1 DEF./ C./ SAIDA
GPIO.setup(SENSORYF2, GPIO.IN, pull_up_down = GPIO.PUD_UP) #SENSORYF2 DEF./ C./ SAIDA
GPIO.setup(SENSORYF3, GPIO.IN, pull_up_down = GPIO.PUD_UP) #SENSORYF3 DEF./ C./ SAIDA
GPIO.setup(BOTA01, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)  #BOTA01 DEF./ C./ ENTRADA
GPIO.setup(BOTA02, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)  #BOTA02 DEF./ C./ ENTRADA

#INICIO DO PROGRAMA
tempo = 0                #VARIABEL TEMPO
inicio = time.time()     #INCIO DO TEMPO
print("Programa Iniciado" )

distancia = "0" #VARIABEL DA DISTANCIA DO SENSOR ULTRASSONICO
volume = "0"     #VARIABEL DO VOLUME DA CAIXA D'AGUA
porce = "0"      #VARIABEL DA PORCENTAGEM DE VOLUME DE AGUA

```

```

senal_1 = "0"    #VARIÁVEL DO SINAL DE ESTADO DA VÁLVULA DE ENTRADA
senal_2 = "0"    #VARIÁVEL DO SINAL DE ESTADO DA VÁLVULA DE SAÍDA DO BANHEIRO
senal_3 = "0"    #VARIÁVEL DO SINAL DE ESTADO DA VÁLVULA DE SAÍDA DA COZ./LAVANDERIA
tempomes = 0     #VARIÁVEL TEMPO DE UM MÊS

count1 = 0       #VARIÁVEL DO CONTADOR DE PULSO DO SENSOR DE VAZÃO 1
litros1 = 0      #VARIÁVEL DO VOLUME EM LITROS DO SENSOR DE VAZÃO 1
def countPulse(channel):    #FUNÇÃO PARA O SENSOR YS S201 1
    global count1
    global litros
    count1 = count1+1       #CONTADOR DE PULSO
    litros1 = (count1/379)   #VOLUME DE LITROS
    print("Quantidade de litro sensor 1%.3f Litros  " % (litros1))
# print ("Pulsos captados sensor 1: %.1f pulso" % (count1))
#DETECTAR BORDA DO PWM PARA O SENSOR VAZÃO 1
GPIO.add_event_detect(SENSORYF1, GPIO.FALLING, callback=countPulse)

count2 = 0       #VARIÁVEL DO CONTADOR DE PULSO DO SENSOR DE VAZÃO 2
litros2 = 0      #VARIÁVEL DO VOLUME EM LITROS DO SENSOR DE VAZÃO 2
count_2 = 0      #VARIÁVEL AUX. DO CONTADOR DE PULSO DO SENSOR DE VAZÃO 2
litros_2 = 0     #VARIÁVEL AUX. DO VOLUME EM LITROS DO SENSOR DE VAZÃO 2
def countPulse(channel):    #FUNÇÃO PARA O SENSOR YS S201 2
    global count2
    global litros2
    global count_2
    global litros_2
    count2 = count2+1       #CONTADOR DE PULSO
    count_2 = count_2+1     #CONTADOR DE PULSO AUXILIAR
    litros2 = (count2/379)   #VOLUME DE LITROS
    litros_2 = (count_2/379) #VOLUME DE LITROS AUXILIAR
    print("Quantidade de litro sensor 2 %.3f Litros  " % (litros2))
    print ("Pulsos captados sensor 2: %.1f pulso " % (count2))
    print("Quantidade de litro sensor 2 auxiliar %.3f Litros  " % (litros_2))
    print ("Pulsos captados sensor 2 auxiliar %.1f pulso" % (count_2))
    if count_2 > 68300:
        count_2 = 0         #ZERAR O CONTADOR AUXILIAR
#DETECTAR BORDA DO PWM PARA O SENSOR DE VAZÃO 2
GPIO.add_event_detect(SENSORYF2, GPIO.FALLING, callback=countPulse)

count3 = 0       #VARIÁVEL DO CONTADOR DE PULSO DO SENSOR DE VAZÃO 2
litros3 = 0      #VARIÁVEL DO VOLUME EM LITROS DO SENSOR DE VAZÃO 2
count_3 = 0      #VARIÁVEL AUXILIAR DO CONTADOR DE PULSO DO SENSOR DE VAZÃO 2
litros_3 = 0     #VARIÁVEL AUXILIAR DO VOLUME EM LITROS DO SENSOR DE VAZÃO 2
def countPulse(channel):    #FUNÇÃO PARA O SENSOR YS S201 2
    global count3
    global litros3
    global count_3

```

```

global litros_3
count3 = count3+1          #CONTADOR DE PULSO
litros3 = (count3/379)     #CONTADOR DE PULSO AUXILIAR
count_3 = count_3+1       #VOLUME DE LITROS
litros_3 = (count_3/379)   #VOLUME DE LITROS AUXILIAR
print("Quantidade de litro sensor 2 %.3f Litros  " % (litros3))
print ("Pulsos captados sensor 2: %.1f pulso " % (count3))
print("Quantidade de litro sensor 2 auxiliar %.3f Litros  " % (litros3aux))
print ("Pulsos captados sensor 2 auxiliar %.1f pulso" % (count3aux))
if count_3 > 68300:
    count_3 = 0            #ZERAR O CONTADOR AUXILIAR
#DETECTAR BORDA DO PWM PARA O SENSOR DE VAZÃO 3
GPIO.add_event_detect(SENSORYF3, GPIO.FALLING, callback=countPulse)

while True:

    def distancia ():
        #ENVIANDO O SINAL DO TRANSMISSOR EM NÍVEL HIGH POR UM TEMPO DE 10 MICROSSEGUNDOS
        GPIO.output(TRANSMISSOR,1)          #ENVIANDO O PULSO
        time.sleep(0.000001)                #TEMPO DE 10 MICROSSEGUNDOS
        GPIO.output(TRANSMISSOR,0)          #INTERROMPER O ENVIO DO PULSO
        while GPIO.input(RECEPTOR) == 0:  #ENQUANTO O TRIG FOR LOW
            tempo_inicial = time.time()      #TEMPO INICIAL
        while GPIO.input(RECEPTOR) == 1:  #ENQUANTO O ECHO FOR HIGH
            tempo_final = time.time()        #CONSEGUIR O TEMPO FINAL
        #CALCULANDO O TEMPO DA REFLEXÃO DO SINAL
        tempo_distancia = tempo_final - tempo_inicial
        distancia = (tempo_distancia*35000)/2 #EQUAÇÃO 2.1
        return distancia

    print("Distancia = %.1f cm" % distancia())
    def volume(): #FUNÇÃO PARA CALCULAR O VOLUME
        largura = 2          #LARGURA DA CAIXA D'ÁGUA
        comprimento = 1      #COMPRIMENTO DA CAIXA D'ÁGUA
        volume = 10*(largura*comprimento*(150-distancia())) #VOLUME EM LITROS
        if (volume < 0):      #ELIMINAR O VALOR NEGATIVO
            volume = 0
        if (volume > 3000):    #ELIMINAR ERRO DE VOLUME MÁXIMO
            volume = 3000
        return volume
    print("Volume da caixa d'agua em Litros = %.1f Litros" % volume())

    def porce(): #FUNÇÃO PARA NÍVEL DO VOLUME EM PORCENTAGEM
        volume_maximo = 3000
        porce = (100*volume()/volume_maximo)
        if (porce < 0):       #ELIMINAR PORCENTAGEM NEGATIVA
            porce = 0

```

```
if (porce > 100):          #ELIMINAR PORCENTAGEM MAIOR QUE 100%
    porce = 100

global sinal_1
nivel_minimo = 20        #NÍVEL MÍNIMO DA CAIXA D'ÁGUA
nivel_maximo = 90        #NÍVEL MÁXIMO DA CAIXA D'ÁGUA
#FUNÇÃO PARA ATIVAR VÁLVULA DE ENTRADA PARA ENCHER A CAIXA D'ÁGUA
if (porce <= nivel_minimo):
    GPIO.output(RELE1, True) #ATIVAR O RELÉ
    print ("Relé ativado")
    sinal_1 = 1              #SINAL PARA ENVIAR AO THINGSPEAK
    print("sinal = %.1f nivel " % sinal_1)
    #FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
    filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
    msg = "OLÁ SENHOR, SUA CAIXA D'ÁGUA ATINGIU O NÍVEL MÍNIMO,
    A VÁLVULA FOI ACIONADA"
    server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
    server.quit()
#FUNÇÃO PARA DESATIVAR VÁLVULA DE ENTRADA PARA ENCHER A CAIXA D'ÁGUA
GPIO.output(RELE1, False) #DESATIVAR O RELÉ
sinal_1 = 0              #SINAL PARA ENVIAR AO THINGSPEAK
print ("Relé desativado")
print("sinal = %.1f nivel " % sinal_1)
#FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
msg = "OLÁ SENHOR, SUA CAIXA D'ÁGUA ATINGIU O NÍVEL MÁXIMO,
A VÁLVULA FOI DESATIVADA"
server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
server.quit()

global sinal_2
#FUNÇÃO PARA DETECTAR TORNEIRA OU CHUVEIRO ESQUECIDO LIGADO
if (litros_2 > 180):
    GPIO.output(RELE2, True) #FECHAR A VÁLVULA DE SAÍDA DO BANHEIRO
    sinal_2 = 0              #SINAL PARA ENVIAR AO THINGSPEAK
    print ("sinal 2 ", sinal_2)
    print ("A VÁLVULA DO BANHEIRO FOI FECHADA")
    #FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
    filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
```



```
server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
msg = "Ola senhor, a valvula de saida do banheiro foi fechada,
voce esqueceu a torneira ou chuveiro ligado"
server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
server.quit()

if (GPIO.input(31) == True): #ATIVAR VÁLVULA QUANDO PRESSIONAR O BOTÃO
    GPIO.output(RELE2, False) #ATIVAR A VÁLVULA
    sinal_2 = 1 #SINAL PARA ENVIAR AO THINGSPEAK
    print ("sinal 2 ", sinal_2)
    print ("A VÁLVULA DO BANHEIRO FOI ABERTA.")
    #FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
    filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
    msg = "Ola senhor, a valvula de saida do banheiro foi aberta"
    server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
    server.quit()
    count_2 = 0

global sinal_3
if (litros_3 > 180): #FUNÇÃO PARA DETECTAR TORNEIRA ESQUECIDA LIGADA
    GPIO.output(RELE3, True) #DESATIVAR A VÁLVULA
    sinal_3 = 0 #SINAL PARA ENVIAR AO THINGSPEAK
    print ("sinal 3 baixo", sinal_3)
    print ("A VÁLVULA DA COZINHA/LAVANDERIA FOI FECHADA")
    #FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
    filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
    msg = "Ola senhor, a valvula de saida da cozinha/lavanderia foi fechada"
    server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
    server.quit()

if (GPIO.input(37) == True): #ATIVAR VÁLVULA QUANDO PRESSIONAR O BOTÃO
    GPIO.output(RELE3, False) #ATIVAR A VÁLVULA
    sinal_3 = 1 #SINAL PARA ENVIAR AO THINGSPEAK
    print ("sinal 3 alto", sinal_3)
    print ("A VÁLVULA DA COZINHA/LAVANDERIA FOI ABERTA")
    #FUNÇÃO PARA ENVIAR A NOTIFICAÇÃO PARA O E-MAIL
    filename = datetime.now().strftime("%H. %M, %S_%Y-%m_%d.txt")
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login("E-MAIL DA AUTOMAÇÃO", "SENHA")
    msg = "Ola senhor, a valvula de saida da cozinha/lavanderia foi aberta"
```

```

        server.sendmail("E-MAIL DA AUTOMAÇÃO", "E-MAIL DO USUÁRIO", msg)
        server.quit()
        count_3 = 0
    return porce
print("Porcentagem de agua = %.1f por cento" % porce())

try:
    #FUNÇÃO PARA ENVIAR PARA O BANCO DE DADOS
    queries = {"api_key": "XX CHAVE DE ESCRITA XX", #CHAVE DE ESCRITA
    #AS VÁRIAVEIS QIUE SERÃO ENVIADA AO THINGSPEAK
        "field1": str(litros1),
        "field2": str(porce()),
        "field3": str(litros2),
        "field4": str(litros3),
        "field5": str(volume()),
        "field6": str(sinal_1),
        "field7": str(sinal_2),
        "field8": str(sinal_3)}
    r = requests.get('https://api.thingspeak.com/update', params=queries)
    if r.status_code == requests.codes.ok:
        print("ENVIADO PARA O BANCO DE DADOS COM SUCESSO")

    if tempo >= 900:          #FUNÇÃO PARA ZERAR O TEMPO A CADA 15 MINUTOS
        count_2 = 0
        count_3 = 0
        inicio = time.time()
        tempomes = tempomes + 20
    if tempomes > 2592000: #FUNÇÃO PARA ZERAR DO CONSUMO TEMPO A CADA 1 MÊS
        count1 = 0
        count2 = 0
        count3 = 0
    print("tempo mes", tempomes)
    fim = time.time()
    tempo = fim - inicio    #TEMPO DE EXECUÇÃO DO CÓDIGO
    print("tempo de programa", tempo)

#FINALIZAR O PROGRAMA

except KeyboardInterrupt:
    print ("\ncaught keyboard interrupt!")
    GPIO.cleanup()
    sys.exit()

```

## APÊNDICE B – CÓDIGO EM HTML DA PÁGINA WEB PARA O SISTEMA DE MONITORAMENTO

```

<html>
<meta charset="utf-8">
  <head>
    </>
    <h1 style="color:lightblue;" align="center">Monitoriamento do Consumo
      de Água Total da Residência </h1>
  </head>

  <body style="background-color:black;">
    <p align="center"><iframe width="450" height="260" style="border: 1px solid
      #cccccc;"src="https://thingspeak.com/channels/1369346/charts/1?bgcolor=
      %23ffffff&color=%23d62020&dynamic=true&results=74&round=1&title=Consumo
      +de+C3%A1gua+da+resid%C3%A2ncia&type=line&xaxis=Tempo&yaxis=Consumo+
      em+litros&yaxismax=500&yaxismin=0"></iframe><iframe width="450" height="260"
      style="border: 1px solid#cccccc;"src="https://thingspeak.com/channels/1369346/
      widgets/302691"></iframe></p>

    <h1 style="color:lightblue;" align="center"> Monitoriamento do Nivel
      e Volume Mensal da Caixa D'agua </h1>

    <p align="center"><iframe width="450" height="260" style="border: 1px
      solid #cccccc;" src="https://thingspeak.com/channels/1369346/widgets/
      302649"></iframe><iframe width="450"height="260" style="border: 1px
      solid #cccccc;"src="https://thingspeak.com/channels/1369346/widgets
      /302653"></iframe></p>

    <h1 style="color:lightblue;" align="center"> Consumo de Água do Banheiro
      e Cozinha/Lavanderia </h1>

    <p align="center"><iframe width="450" height="260" style="border: 1px solid
      #cccccc;"src="https://thingspeak.com/channels/1369346/charts/3?bgcolor=
      %23ffffff&color=%23d62020&dynamic=true&results=72&round=1&title=Consumo
      +do+Banheiro&type=line&xaxis=Tempo&yaxis=Consumo+em+litros&yaxismax=500&
      yaxismin=0"></iframe><iframe width="450"height="260" style="border: 1px solid
      #cccccc;"src="https://thingspeak.com/channels/1369346/charts/4?bgcolor=%23
      ffffff&color=%23d62020&dynamic=true&results=72&round=1&title=Consumo+da+
      cozinha%2FLavanderia&type=line&xaxis=Tempo&yaxis=Consumo+em+litros&yaxismax=
      500&yaxismin=0"></iframe></p>

    <h1 style="color:lightblue;" align="center"> Estado da Válvula de Entrada </h1>

    <p align="center"><iframe width="450" height="260" style="border: 1px

```

```

solid#cccccc;"src="https://thingspeak.com/channels/1369346/charts/6
?bgcolor=%23ffffff&color=%23d62020&dynamic=true&results=72&round=1&
title=Estado+da+v%C3%A1lvula+de+entrada&type=line&xaxis=Tempo&yaxis=
Bin%C3%A1rio&yaxismax=1&yaxismin=0"></iframe><iframe width="450" height
="260" style="border: 1px solid#cccccc;"src="https://thingspeak.com/
channels/1369346/widgets/302654"></iframe></p>

<h1 style="color:lightblue;" align="center"> Estado da Válvula do Banheiro </h1>

<p align="center"><iframe width="450" height="260" style="border: 1px solid #
cccccc;"src="https://thingspeak.com/channels/1369346/charts/7?bgcolor=%23ffffff
&color=%23d62020&dynamic=true&results=72&round=1&title=Estado+da+v%C3%A1lvula+
do+banheiro&type=line&xaxis=Tempo&yaxis=Bin%C3%A1rio&yaxismax=1&yaxismin=0">
</iframe><iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/1369346/widgets/302656"></iframe></p>

<h1 style="color:lightblue;" align="center"> Estado da Válvula da Cozinha/
Lavanderia </h1>

<p align="center"><iframe width="450" height="260" style="border: 1px solid
#cccccc;"src="https://thingspeak.com/channels/1369346/charts/8?bgcolor=%23
ffffff&color=%23d62020&dynamic=true&results=72&round=1&title=Estado+da+v%C3
%A1lvula+da+cozinha%2Flavanderia&type=line&xaxis=Tempo&yaxis=Bin%C3%A1rio&
yaxismax=1&yaxismin=0"></iframe></iframe><iframe width="450" height="260"
style="border: 1px solid #cccccc;"src="https://thingspeak.com/channels/1369346
/widgets/302658"></iframe></p>

<h1 style="color:lightblue;" align="center"> Localização do Servidor </h1>

<p align="center"><iframe width="450" height="260" style="border: 1px
solid #cccccc;"src="https://thingspeak.com/channels/1369346/maps/channel
_show"></iframe></p>
</body>
<footer>
  <p style="color:white;" align="center"> Desenvolvedor: Joel do Amaral Gurgel
  Neto</p>
  <p style="color:white;" align="center"> E-mail: <a style="color:white;"
  href="joel.neto@alunos.ufersa.edu.br"> joel.neto@alunos.ufersa.edu.br
  </a></p>
  <p style="color:white;" align="center"> Orientador: Professor Dr. Walber
  Medeiros Lima</p>
  <p style="color:white;" align="center"> E-mail: <a style="color:white;"
  href="walber@ufersa.edu.br"> walber@ufersa.edu.br </a> </p>
</footer>
</html>

```

## **Anexos**

## ANEXO A – DATASHEET DO SENSOR DE VAZÃO MODELO YF - 201

### YIFA the plastics Ltd Prodcut Introduction

- 1.Modle:YF-21
- 2.Product Name:Hall sensor
- 3.Flow Range: 1-30L/MIN
- 4.(1)Connection Method



(2)Voltage Range 3.5-24VDC, Pulse Characteristic:  $F=7Q(L/MIN)$ .

(3)Extent of error:  $\pm 5\%$ .

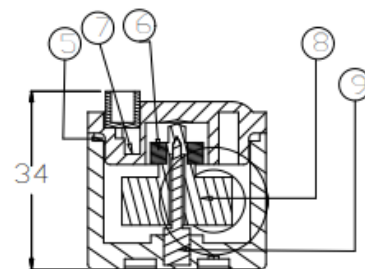
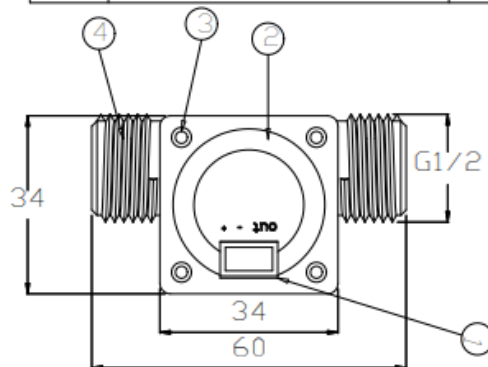
(4)Flow-Pulse

2L/MIN=16HZ 4L/MIN=32.5HZ 6L/MIN=49.3HZ

8L/MIN=65.5HZ 10L/MIN=82HZ

5.Bom

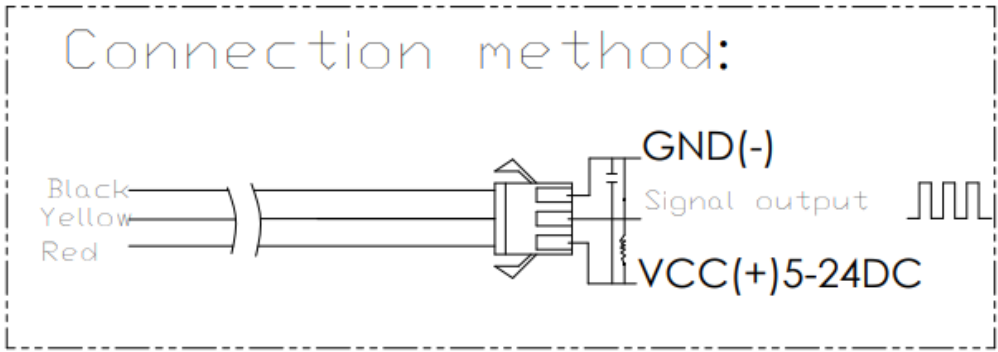
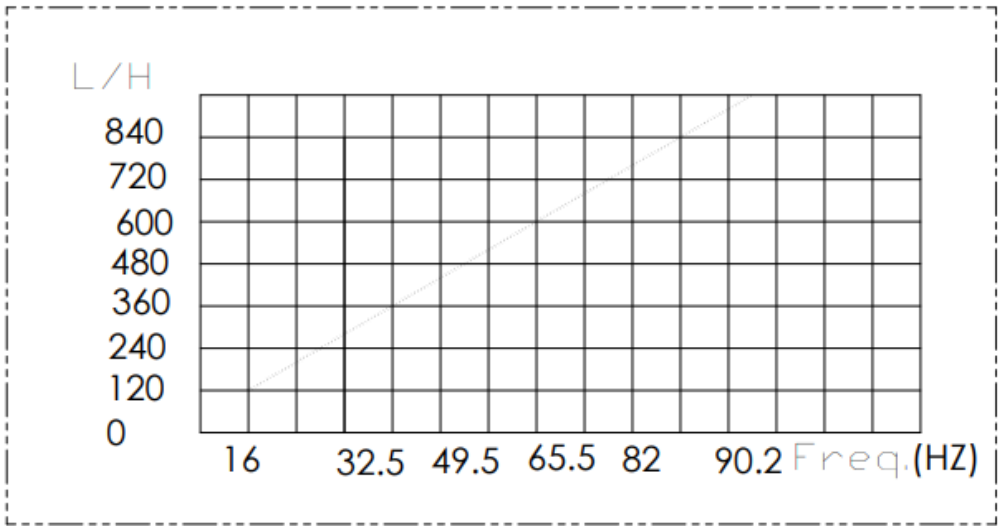
| No. | Item                | Material | Qty. |
|-----|---------------------|----------|------|
| 1   | Connection wire     |          | 1    |
| 2   | Bonnet              | PA       | 1    |
| 3   | Screw               |          | 4    |
| 4   | Valve body          | PA       | 1    |
| 5   | Leak press valve    |          | 1    |
| 6   | Magnet              |          | 1    |
| 7   | Hall                |          | 1    |
| 8   | Impeller            | POM      | 1    |
| 9   | Rustless steel axis | SUS304   | 1    |
| 10  |                     |          |      |
| 11  |                     |          |      |



| Flow-Pulse | Flow-Curve | Connection method |
|------------|------------|-------------------|
|------------|------------|-------------------|

一.Flow Range:100L/H-/1800H-L/H

| Flow (L/H) | Frezq.(HZ) | Erro range |
|------------|------------|------------|
| 120L/H     | 16         | ±10        |
| 240L/H     | 32.5       |            |
| 360L/H     | 49.3       |            |
| 480L/H     | 65.5       |            |
| 600L/H     | 82         |            |
| 720L/H     | 90.2       |            |



## ANEXO B – DATASHEET DO SENSOR ULTRASSÔNICO

### MODELO HC-SR04



Tech Support: [services@elecfreaks.com](mailto:services@elecfreaks.com)

### Ultrasonic Ranging Module HC - SR04

#### Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:

- (1) Using IO trigger for at least 10us high level signal,
- (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
- (3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.

Test distance = (high level time×velocity of sound (340M/S) / 2,

#### Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

#### Electric Parameter

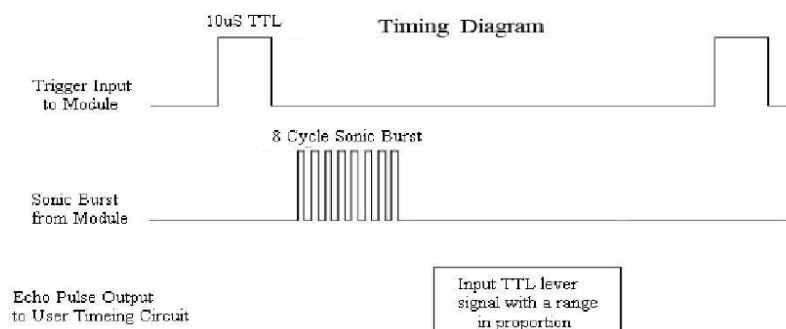
|                      |                                                    |
|----------------------|----------------------------------------------------|
| Working Voltage      | DC 5 V                                             |
| Working Current      | 15mA                                               |
| Working Frequency    | 40Hz                                               |
| Max Range            | 4m                                                 |
| Min Range            | 2cm                                                |
| MeasuringAngle       | 15 degree                                          |
| Trigger Input Signal | 10uS TTL pulse                                     |
| Echo Output Signal   | Input TTL lever signal and the range in proportion |
| Dimension            | 45*20*15mm                                         |





### Timing diagram

The Timing diagram is shown below. You only need to supply a short 10 $\mu$ s pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion. You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula:  $\mu\text{s} / 58 = \text{centimeters}$  or  $\mu\text{s} / 148 = \text{inch}$ ; or: the range = high level time \* velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



---

**Attention:**

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

[www.Electronics.com](http://www.Electronics.com)

