WS Lock: Fechadura Tecnológica Automatizada Aplicada a Segurança Residencial

Wilian Hendler Borges[1], Matheus Leandro Ferreira[1]

[1]Curso de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) - Criciúma – SC – Brasil

Wilianx7@hotmail.com, mlf@unesc.net

Abstract. *As technologies aimed at home automation evolve, there is a scenario in which they can be widely applied: the maximization of home security and management. This research aimed to develop a technological door lock prototype that can be activated through a mobile application with the aid of a microcontroller and with the intermediation of communication with an API. Differentiating itself from other related works mainly because it allows interaction with the prototype from anywhere in the world. The results were satisfactory, allowing the activation of the lock by smartphone from any distance as long as it was connected to the internet. In addition, through the history available in the application, it became possible to verify all actions performed on the lock.*

Resumo. *A medida em que as tecnologias voltadas a automação residencial evoluem, observa-se um cenário em que elas podem ser amplamente aplicadas: a maximização da segurança e gerenciamento das residências. Esta pesquisa teve como objetivo desenvolver um protótipo de fechadura tecnológica que possa ser ativada por meio de um aplicativo móvel com auxílio de um microcontrolador e com a intermediação da comunicação com uma API. Diferenciando-se de outros trabalhos correlatos principalmente pelo fato de permitir a interação com o protótipo de qualquer lugar o mundo. Os resultados foram satisfatórios, permitindo a ativação da fechadura pelo smartphone de qualquer distância desde que conectado à internet. Além disso, por meio do histórico disponibilizado no aplicativo, tornou-se possível a verificação de todas as ações realizadas no dispositivo.*

1. Introdução

Aperfeiçoar a proteção das residências e de seus familiares sempre foi e continua sendo motivo de preocupação entre a população. Segundo Mandarini (2005), a falta de segurança é considerada um fenômeno mundial que se espalhou por todos os níveis sociais, principalmente nas grandes concentrações urbanas, onde a requisição da sociedade por esse quesito não vem sendo adequadamente atendida pelo Estado.

A automação residencial e predial encontra-se em constante evolução e apresenta cada vez mais alternativas que visam utilizar a tecnologia em favor da comodidade e segurança da população. Segundo artigo publicado no site ARCHDAILY pelo autor Eduardo Souza, a automação residencial está se mostrando uma adição cada vez mais fundamental e acessível a projetos arquitetônicos, seja para novos edifícios ou reformas (SOUZA, 2019, tradução nossa).

Atualmente a interação dos usuários com dispositivos autônomos, presencialmente ou à distância, acontece de maneira mais facilitada, tendo em vista a grande melhoria das redes de internet e dos equipamentos. Com a interação remota e estando em um local que disponibiliza conexão de internet, o usuário pode controlar vários aspectos de sua residência, trazendo inúmeros benefícios, tais como: gerenciar suas fechaduras; ligar/desligar seus eletrodomésticos como, por exemplo, ar-condicionado; abrir/fechar cortinas e janelas; acionar portões de garagem.

Conforme pesquisa realizada em 2017 pela empresa GARTNER pode-se observar a evolução e a quantidade dos dispositivos conectados, onde 8,4 bilhões de dispositivos conectados estarão em uso em todo o mundo em 2017, um aumento de 31% em relação a 2016 e atingirá 20,4 bilhões em 2020 (GARTNER, 2017, tradução nossa).

Diante do número de furtos e arrombamentos, em ambientes prediais ou residenciais, observa-se a necessidade de oferecer à população uma quantidade cada vez maior de meios para evitar tais acontecimentos.

Segundo matéria publicada pelo jornal Gazeta do Povo, em 2017 a taxa de roubos registrados no Brasil batia em 820 casos para cada 100 mil habitantes, um número que permanece entre os mais altos do continente (GAZETA DO POVO, 2019).

Frente a esses problemas e com a evolução tecnológica que vem ocorrendo nas últimas décadas, as soluções computacionais se mostram cada vez mais atrativas no auxílio e resolução de tais impasses.

Os circuitos embarcados, utilizados em projetos de sistemas eletrônicos juntamente com os microcontroladores, consistem na combinação de *hardwares* e *softwares* acoplados em um único chip eletrônico. Por sua vez, tem como principal função a execução de uma aplicação, ou conjunto de aplicações, relacionadas como um único sistema (PATTERSON; HENNESSY, 2013). A utilização desses recursos é de extrema importância no que tange a automação predial e residencial, pois abrem diversas portas para a IOT (*Internet Of Things*).

No entanto, esses dispositivos precisam de uma interface amigável para que os usuários possam interagir e extrair informações dos mesmos. Nesse sentido, a utilização de smartphones pela população cresce a cada dia, tornando-se uma ferramenta indispensável no cotidiano e que consequentemente pode ser utilizada para esse tipo de integração.

De acordo com o levantamento bibliográfico realizado com foco na utilização de dispositivos autônomos em benefício da população, em sua maioria, foram encontrados trabalhos que utilizaram como principais recursos arduinos, smartphones, serviços web, teclados matriciais, dispositivos de autenticação biométrica e NFC.

Oliveira (2013) desenvolveu um sistema capaz de controlar o acionamento de uma fechadura eletrônica por meio da tecnologia NFC e com a utilização do microcontrolador Arduino. Menezes (2018) construiu um sistema embarcado para realização de autenticação biométrica ou senha numérica para controlar o acesso ao Laboratório de Automação Predial da UFOP. Foram utilizados os dispositivos: Microcontrolador Arduino UNO; Tela LCD 20 x 4; Sensor óptico de impressão digital Adafruit; Teclado matricial 4 x 3. Declerque (2014) implementou uma solução de controle e monitoramento para sua residência com o objetivo de realizar as funções: controle do acendimento de lâmpadas externas; acionamento do portão de veículos e das travas presentes na porta e

no portão da residência. Para a implementação do *hardware*, todos os dispositivos foram conectados a módulos de rede da tecnologia *ZigBee*. No que se refere ao *software*, foi desenvolvida uma interface capaz de gerenciar o funcionamento de todo o sistema bem como um servidor de dados acessado através de uma página web certificada.

Portanto, esta pesquisa tem por objetivo desenvolver um protótipo de fechadura tecnológica que possa ser ativada por meio de um aplicativo móvel com auxílio de um microcontrolador e com a intermediação da comunicação com uma API. Os objetivos específicos consistem em: conhecer o funcionamento e desenvolvimento de sistemas embarcados; reconhecer os diversos aspectos sobre a segurança e automação residencial; entender as melhores técnicas para o desenvolvimento de *softwares mobile*; criar aplicativo *mobile* para comunicação com a fechadura; desenvolver *Application Programming Interface;* integrar aplicativo com o protótipo de fechadura por intermédio da API.

2. Materiais e Métodos

A presente pesquisa caracteriza-se por ser aplicada e de base tecnológica. Desenvolveu-se o protótipo de um sistema para controlar o acionamento de uma fechadura eletrônica por meio do smartphone. O protótipo constitui-se em três módulos que foram denominados: **módulo executor, módulo móvel e módulo integrador** (Figura 1).

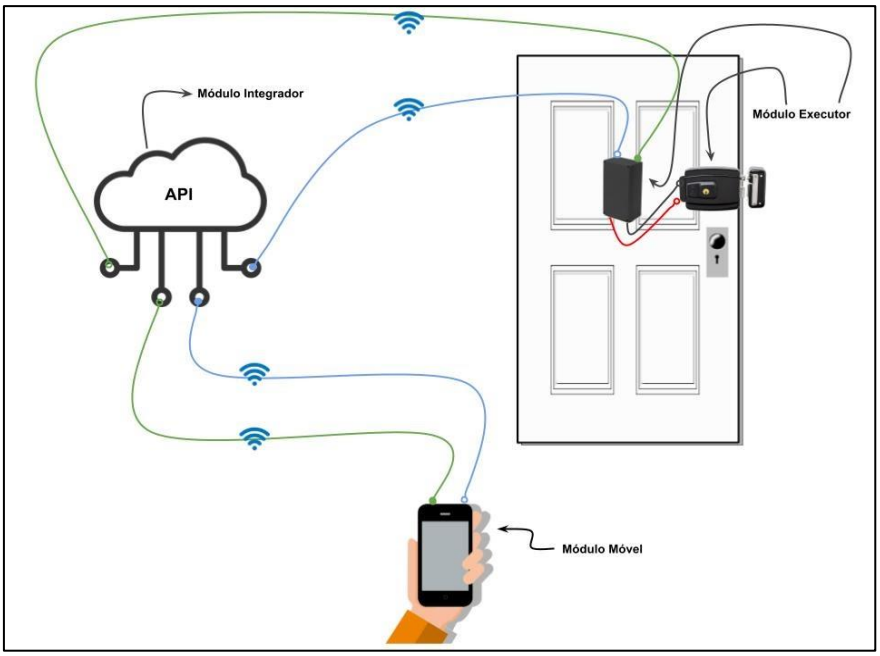

Figura 1. Funcionamento do protótipo do sistema

O módulo executor empregou conceitos da computação embarcada, sendo fixado próximo à fechadura eletrônica instalada na porta modelo para controlar o acionamento da tranca. O módulo móvel consiste no desenvolvimento de um aplicativo que será utilizado pelos usuários para acionar e gerenciar suas fechaduras. O módulo integrador utilizou-se de conceitos da computação em nuvem e é responsável por intermediar todas as comunicações realizadas entre os módulos móvel e executor. Além disso, também gerencia a persistência das informações no banco de dados, garantindo segurança e confiança na transmissão e armazenamento dessas informações.

O protótipo do sistema integrou recursos de *hardware* e *software* para realizar o acionamento remoto da fechadura. A seguir, na figura 2, tem-se uma visão geral de sua arquitetura.

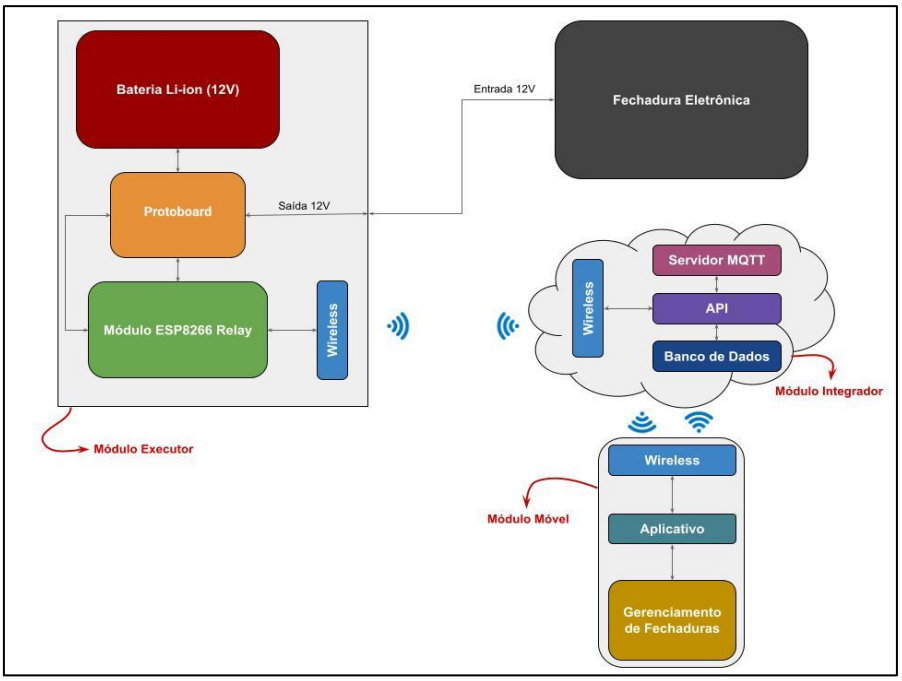

Figura 2. Arquitetura do protótipo

O módulo executor necessitou de componentes eletrônicos como: uma bateria de Li-ion (12V), uma placa constituída de dois relés e conexão para um ESP8266 e uma mini protoboard. O módulo móvel necessitou de um smartphone e do *software* de um aplicativo. Para o módulo integrador foi necessária uma máquina hospedeira para executar a API, o servidor MQTT e o banco de dados.

2.1 Módulo Executor

O módulo executor foi dividido em três elementos: a fechadura eletrônica, a porta modelo e uma caixa plástica contendo todos os componentes eletrônicos responsáveis pela comunicação *wireless*, alimentação e acionamento da fechadura.

Devido à complexidade na implementação de um protótipo de fechadura próprio, optou-se neste trabalho a utilização e adaptação de um modelo de fechadura eletrônica já oferecido pelo mercado. O modelo escolhido foi a fechadura elétrica reversível modelo LR100 da marca AGL (figura 3). Sua instalação é feita de maneira sobreposta a porta ou portão desejado e possui uma tensão de alimentação de 12V.

Figura 3. Fechadura

A caixa plástica escolhida para o armazenamento dos componentes é da marca Fortek e possui as seguintes dimensões: 200x120x55mm (Figura 4). Em seu interior existem três elementos: uma bateria de Li-ion, um módulo relay e uma mini protoboard.

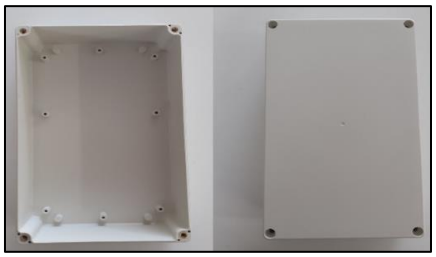

Figura 4. Caixa plástica Fortek

Com relação a porta modelo, optou-se pela aquisição de uma porta sob medida onde o módulo executor foi instalado para uma demonstração de uso mais precisa (figura 5). As dimensões da porta são: 600x400x80mm.

Figura 5. Porta modelo com módulo executor

2.1.1 Alimentação do Circuito

Para a alimentação do circuito associado a fechadura foi utilizada a bateria recarregável de Li-ion modelo UR18650A da marca Sanyo (figura 6) que possui uma tensão nominal de 12V e capacidade de 2200Mah integrada a uma placa de proteção de carga PCB BMS.

Figura 6. Bateria Li-ion Sanyo UR18650A

A escolha se deu devido ao fato de o circuito necessitar dessa tensão de alimentação específica e as dimensões da bateria serem favoráveis a caixa plástica onde ela foi fixada.

2.1.2 Módulo Relay

A fechadura eletrônica escolhida possui um mecanismo de dois estados, um quando está alimentada e outro quando não está. No momento em que o mecanismo não está sendo provido de energia, sua tranca permanece bloqueada. A partir do instante em que uma tensão de 12V é aplicada, o mecanismo realiza a liberação da sua tranca e a mantém liberada até que a alimentação seja interrompida.

Portanto, para controlar tais estados tornou-se necessária a utilização de um relé. Foi escolhido o módulo ESP8266 Relay x2 da fabricante LC Technology que possui dois relés com conexões NO (Normal Aberto) e NC (Normal Fechado), com capacidade máxima de corrente de 10A. A alimentação do módulo é de 12V e possui comunicação serial por meio dos pinos RX e TX (figura 7). Além disso, essa placa possui um conector especial onde um ESP8266 pode ser acoplado para controlar o acionamento dos relés.

Os principais motivos para escolha deste módulo foram o conector para o ESP8266 e sua tensão de alimentação de 12V que, por consequência, é a mesma tensão necessária para o acionamento da fechadura eletrônica.

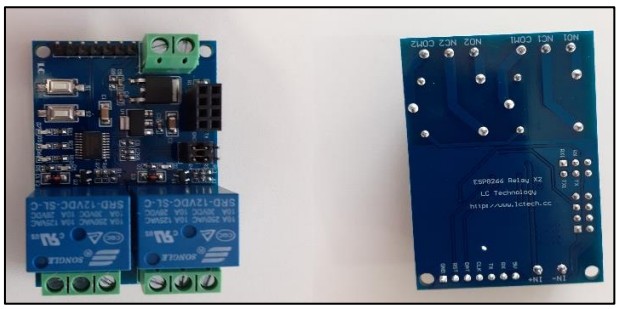

Figura 7. Módulo ESP8266 Relay x2

Conforme mencionado, o módulo de relés necessita de um microcontrolador ESP8266 que será responsável por comandar seu funcionamento. Sendo assim, o modelo designado foi o Módulo WiFi ESP8266 ESP-01 fabricado pela LC Technology (figura 8).

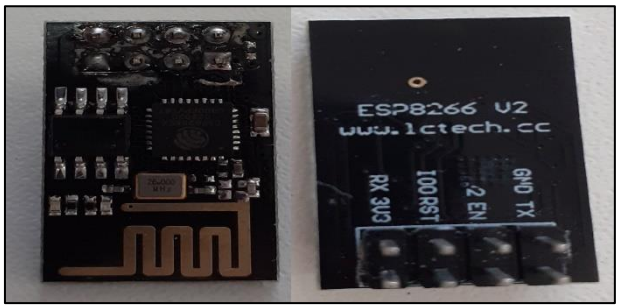

Figura 8. Módulo WiFi ESP8266

A escolha se deu pelo fato de que este módulo possui conexão WiFi e um dos propósitos do protótipo desenvolvido era de permitir ao usuário a interação com sua fechadura de qualquer lugar desde que conectado à internet. Além disso, suas dimensões são pequenas (25x14x1mm) e suas conexões encaixam perfeitamente no módulo de relés.

2.1.3 Protoboard

Para realizar a conexão entre os circuitos, tornou-se necessária a utilização de uma protoboard (figura 9).

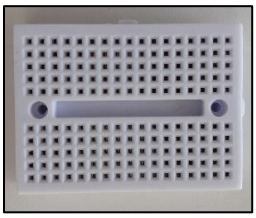

Figura 9. Mini protoboard 170 pontos

Como o número de conexões necessárias era baixo, optou-se por uma mini protoboard de 170 pontos devido ao seu tamanho reduzido de 45x34x8.5mm.

2.1.4 Montagem do Circuito

Para montagem do circuito, todas as conexões foram intermediadas pela protoboard sendo que a alimentação foi provida pela bateria de Li-ion de 12V conforme esquema demonstrado abaixo na figura 10.

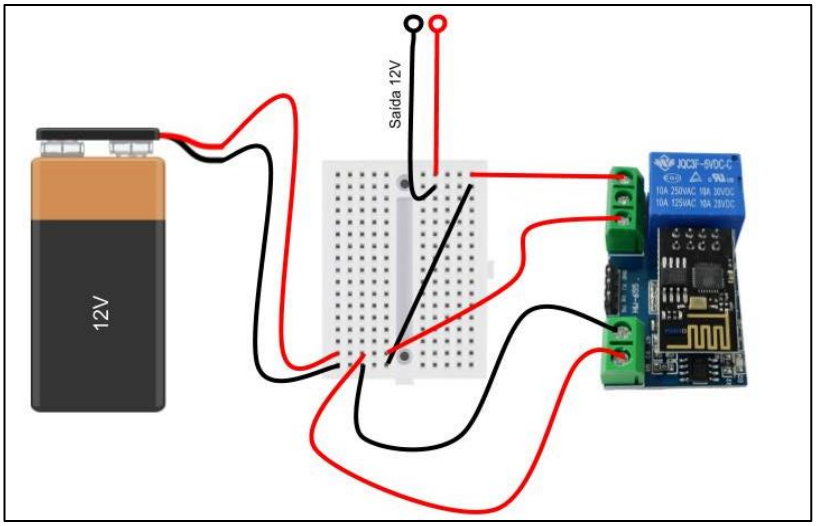

Figura 10. Esquema do circuito

2.1.5 Desenvolvimento do Software para o ESP8266

A conexão do ESP8266 de forma nativa com a rede WiFi da residência normalmente é realizada via código e pode ser complexa para os usuários finais. Dessa forma, no *software* implementado foi utilizada a biblioteca WiFiManager.h desenvolvida por Tzapu em sua versão 0.16.0 com o objetivo de facilitar essa conexão.

A biblioteca WiFiManager.h permite ao usuário conectar o ESP8266 a sua rede de internet por meio de uma interface mais amigável, informando o nome (SSID) e senha da rede desejada (figura 11). Para acessar a página de configuração, após todos os módulos estarem ativos, o usuário deverá conectar seu smartphone ou notebook a rede criada pelo ESP8266, abrir um navegador e acessar o endereço “192.168.4.1”.

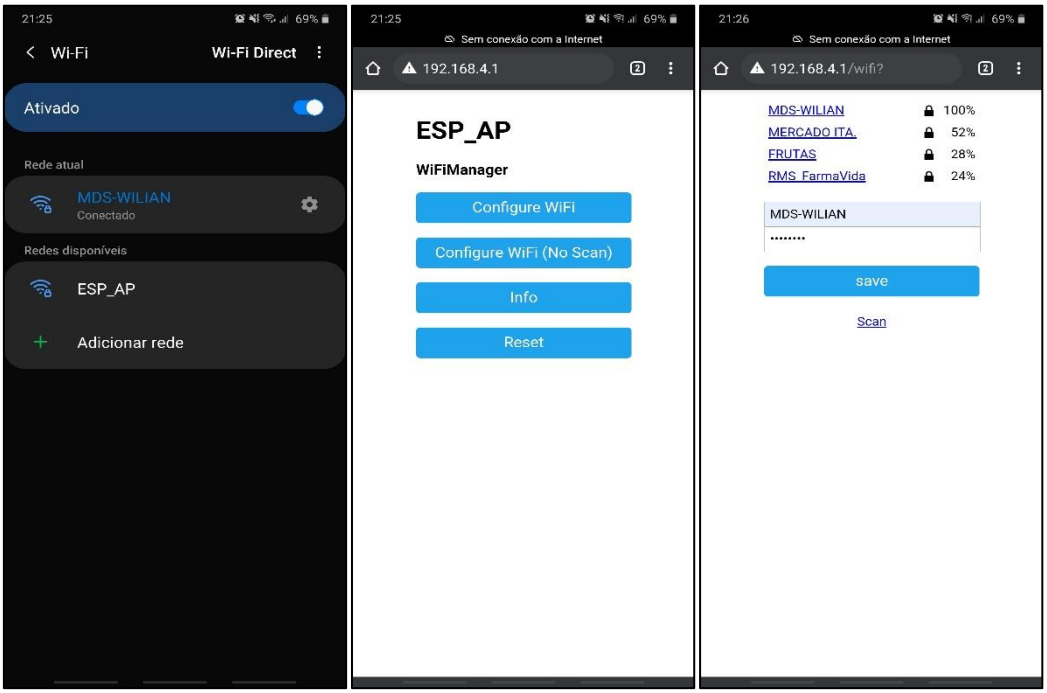

Figura 11. WiFiManager

Para evitar a realização contínua de requisições HTTP diretas à API por parte do ESP8266, foi utilizado o protocolo de comunicação MQTT. Esse protocolo permite a subscrição e publicação de mensagens em tempo real agrupadas por tópicos. Sendo assim, para empregar este recurso foi utilizada a biblioteca PubSubClient.h desenvolvida por Nick O'Leary em sua versão 2.8.0.

Em sua etapa de configuração inicial a taxa de transferência Serial é definida para 115200 bits por segundo e o WiFiManager é iniciado, verificando a existência de uma configuração já salva anteriormente para conexão de rede. Caso essa configuração exista e a conexão seja bem sucedida, o *software* segue para a próxima etapa. Caso contrário, o ESP8266 entra em seu modo de *Access Point,* criando uma rede WiFi chamada "ESP_AP" com a senha "12345678". A partir disso, o usuário deverá conectar a essa rede, abrir a página "192.168.4.1" e realizar a configuração para conectar o ESP8266 a sua rede WiFi.

Após a conexão ser bem sucedida, o microcontrolador entra no modo *Station* e o MQTT é iniciado definindo-se o endereço IP, a porta do servidor e uma função *callback* que será executada sempre que chegar alguma mensagem nos tópicos aos quais o ESP8266 está inscrito.

Passando por essa configuração inicial, o próximo passo é conectar ao servidor MQTT. Para isso, dentro da função *loop*, que é executada continuamente, é verificado se a conexão com o servidor já foi realizada. Em caso negativo, é definido um *delay* de 5 segundos para a próxima tentativa de conexão. Em caso positivo, são realizadas as subscrições nos tópicos "OPEN_DOOR" e "CLOSE_DOOR". Será a partir da publicação nesses tópicos que a ativação ou desativação dos relés será disparada.

É importante destacar que a subscrição nesses tópicos é protegida por configurações de usuário e senha definidas no servidor. Portanto, para realizar a conexão o código empregado no microcontrolador possui as chaves de acesso.

A partir deste momento, sempre que houverem mensagens nestes tópicos a função de processamento será executada. No entanto, supondo-se que existam 10 microcontroladores em operação ao mesmo tempo, todos eles irão receber todas as mensagens publicadas mesmo que essa mensagem seja destinada a apenas um deles.

Para contornar essa situação, a primeira ação da função de processamento, após tratar a mensagem recebida, é verificar se o conteúdo da mensagem corresponde ao endereço MAC do microcontrolador que está executando o código. Em caso negativo, a mensagem é ignorada. Do contrário, de acordo com o tópico de origem da mensagem um dos relés é ativado ou desativado por meio de escrita serial. Para ativar o relé, é escrito, via serial, o seguinte comando: "{0xA0, 0x01, 0x01, 0xA2}". Para desativar: "{0xA0, 0x01, 0x00, 0xA1}".

Para realizar o desenvolvimento, gerenciamento e carregamento do *software* no ESP8266, foi utilizado a Arduino IDE em sua versão 1.8.13. Esta IDE utiliza uma linguagem muito semelhante ao C++. Além disso, foi utilizado um adaptador USB com chip controlador CH340G (figura 12).

Figura 12. Adaptador USB

Por meio dele foi possível conectar o ESP8266 ao computador em que o código foi escrito, facilitando sua transferência.

2.2 Módulo Integrador

O módulo integrador pode ser dividido em três partes: o servidor MQTT, a API e o banco de dados. Seus principais objetivos são de gerenciar o armazenamento de todas as informações do sistema no banco de dados e viabilizar a comunicação entre o módulo móvel e executor por meio de requisições HTTP e do servidor MQTT.

Como citado anteriormente, esse módulo precisa estar em execução a todo momento para que possa cumprir com seu propósito. Portanto, foi utilizada uma instância computacional em nuvem provida pela Amazon Web Services. A instância escolhida foi a denominada *t2.micro* com um núcleo de processamento e 1GB de memória RAM.

2.2.1 Servidor MQTT

O servidor MQTT consiste em um serviço de recebimento e entrega de mensagens agrupadas em tópicos utilizando o protocolo MQTT. Para isso, foi utilizado o Mosquitto MQTT, um *broker* que faz o gerenciamento deste protocolo. Este *software* é desenvolvido pela Eclipse Foundation e distribuído de forma gratuita.

Por meio do Mosquitto torna-se possível a criação de usuários com login e senha e o controle, por usuário, da escrita e leitura de cada tópico. Dessa forma, o serviço atua de maneira segura não permitindo o acesso de usuários não autorizados nem a inscrição ou publicação por parte de usuários legítimos, mas que não possuem a permissão necessária.

Portanto, foram criados dois usuários:

- **ws-lock-api:** este usuário é utilizado pela API para a publicação e subscrição de tópicos no serviço MQTT. Por sua vez, possui permissão de escrita aos tópicos “OPEN_DOOR” e “CLOSE_DOOR”;
- **esp-8266-client:** usuário utilizado por todos os microcontroladores ESP8266. Possui permissão de leitura dos tópicos “OPEN_DOOR” e “CLOSE_DOOR”.

Conforme citado acima, foram criados dois tópicos, cada um com a sua responsabilidade. O tópico “OPEN_DOOR” tem como objetivo comunicar o microcontrolador correspondente ao endereço MAC informado na mensagem deste tópico que a tranca da fechadura deve ser liberada. Por consequência, o tópico “CLOSE_DOOR” faz justamente o oposto.

2.2.2 API

Para o desenvolvimento deste componente, optou-se pela linguagem de programação PHP juntamente ao *Framework* de desenvolvimento Laravel. O motivo se deu pelo fato de serem tecnologias consolidadas no mercado e pela afinidade do autor com as mesmas.

Por meio da API são realizadas ações como: autenticação de usuários; cadastro e edição de usuários; cadastro, edição, deleção e controle de acesso de fechaduras; consulta do histórico de ações realizadas em cada fechadura; acionamento de fechaduras.

2.2.2.1 Autenticação e segurança

Com relação a segurança e autenticação de usuários, foi utilizado um método baseado em tokens provido pelo *plugin* jwt-auth do desenvolvedor Sean Tymon na versão 1.0.2 e que está disponível gratuitamente no GitHub. Seu objetivo consiste em fornecer um token aleatório para cada usuário, assinado com outro token fixo que fica na API, sempre que ele é autenticado. Além disso, sempre que o usuário fecha o módulo móvel e abre novamente, mesmo permanecendo logado, um novo token é gerado para ele.

Com base neste token a API poderá distinguir requisições de usuários autenticados e não autenticados ou com autenticações fajutas. Dessa forma, torna-se possível a criação de rotas públicas e privadas (figura 13).

```php
/**
 * Public routes
 */
Route::group(['middleware' => ['api', 'throttle:180,1']], function () {
    Route::post( uri: 'login', [AuthController::class, 'login']);

    Route::put( uri: 'users/create-or-update', [UserController::class, 'createOrUpdate']);

    Route::put( uri: 'validations/unique', [ValidationController::class, 'unique']);
});

/**
 * Private routes
 */
Route::group(['middleware' => ['api', 'auth:api', 'throttle:60,1']], function () {
    Route::post( uri: 'logout', [AuthController::class, 'logout']);
    Route::post( uri: 'refresh', [AuthController::class, 'refresh']);
    Route::get( uri: 'validate', [AuthController::class, 'validateJwt']);

    Route::apiResource( name: 'users', controller: UserController::class)->except(['store', 'update']);

    Route::put( uri: 'locks/create-or-update', [LockController::class, 'createOrUpdate']);
    Route::apiResource( name: 'locks', controller: LockController::class)->except(['store', 'update']);

    Route::apiResource( name: 'lock-histories', controller: LockHistoryController::class)->only(['index']);

    Route::put( uri: 'mqtt/open-door', [MQTTController::class, 'openDoor']);
    Route::put( uri: 'mqtt/close-door', [MQTTController::class, 'closeDoor']);
});
```

Figura 13. Rotas púbicas e privadas da API

Como públicas, foram definidas somente as rotas de login, criação de usuários e validação de registros únicos. Todas as demais foram definidas como privadas e só podem ser acessadas por um usuário devidamente autenticado.

2.2.2.2 Testes unitários

Com o objetivo de garantir que todas as funcionalidades da API estejam funcionando corretamente, foram implementados testes unitários utilizando-se do *plugin* phpunit de Sebastian Bergmann em sua versão 9.4.0.

Estes testes consistem em aferir que todas as funções estão funcionando corretamente para as mais diversas situações. Portanto, foram implementados testes para os métodos de cada *controller* disponível na API, simulando diversos cenários (figura 14).

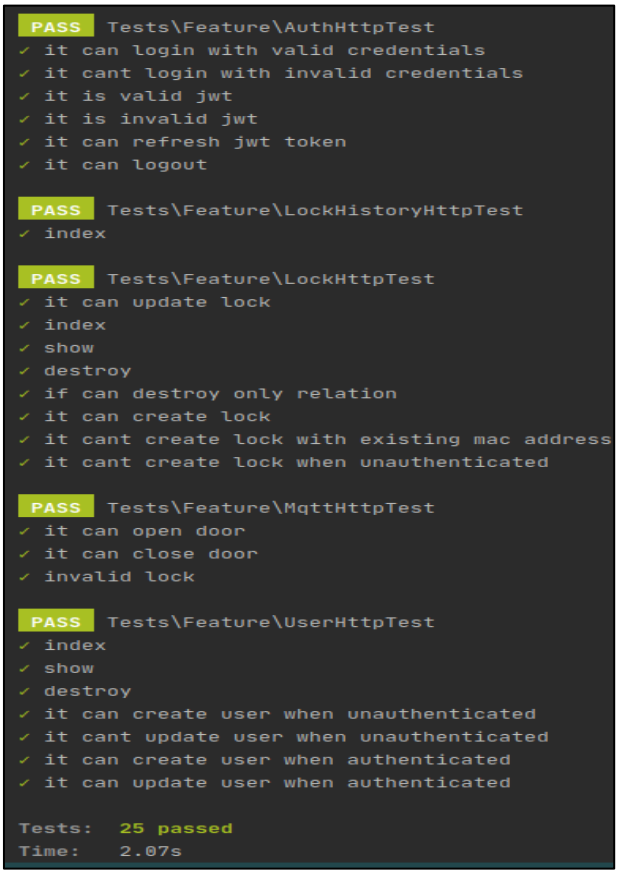

Figura 14. Execução dos testes unitários

Na figura acima estão todas as situações simuladas em cada *controller* e seus resultados.

2.2.2.3 *Migrations* e *seeders*

Foram utilizadas também as funcionalidades de *migrations* e *seeders* disponíveis no *Framework* Laravel para criação das tabelas no banco de dados e inserção de dados padrão (figura 15).

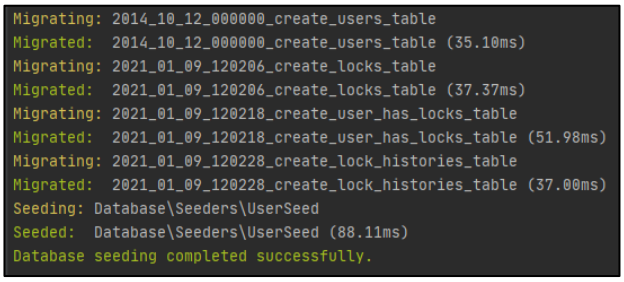

Figura 15. Execução das *migrations* e *seeders*

Dessa forma, foram criadas as tabelas: *users*; *locks*; *user_has_locks*; *lock_histories*. Com relação aos dados padrões, foi inserido um usuário chamado de "admin".

2.2.3 Banco de Dados

O banco de dados utilizado para a persistência das informações do sistema foi o MySQL em sua versão 8.0. Conforme citado anteriormente, as tabelas criadas e suas respectivas funções foram:

- ***users:*** armazenamento de todos os usuários do sistema;
- ***locks:*** armazenamento de todas as fechaduras do sistema;
- ***user_has_locks:*** tabela intermediária com o objetivo de interligar cada usuário do sistema com suas respectivas fechaduras;
- ***lock_histories:*** armazenamento de todas as ações proferidas em cada fechadura juntamente ao usuário responsável.

Foi criada também uma regra lógica chamada de *constraint* com o objetivo de garantir que não haja duas fechaduras com o mesmo endereço MAC cadastradas simultaneamente, bloqueando o cadastro no caso desta tentativa.

2.3 Módulo Móvel

O módulo móvel consiste em um aplicativo voltado para a plataforma Android. O *software* desenvolvido foi responsável por fornecer uma interface amigável ao usuário para a interação com suas fechaduras. No seu desenvolvimento foram empregados a IDE Android Studio na versão 4.1.2 e a linguagem JAVA na versão 8.

Para comunicação do módulo móvel com o módulo integrador foi utilizada a biblioteca Retrofit distribuída pela Square Open Source em sua versão 2.9.0. O principal objetivo dessa biblioteca é facilitar a integração do aplicativo com a API e a realização de requisições HTTP utilizando os métodos *GET*, *POST*, *PUT*, *PATCH* e *DELETE.*

Ao abrir o aplicativo, é apresentado ao usuário a tela de autenticação (figura 16) e logo abaixo do botão "Entrar" existe um texto clicável "Não tem conta? crie uma!" onde é possível abrir o formulário para cadastro de um novo usuário (figura 16). Após realizar a autenticação, o usuário será direcionado a tela inicial (figura 16).

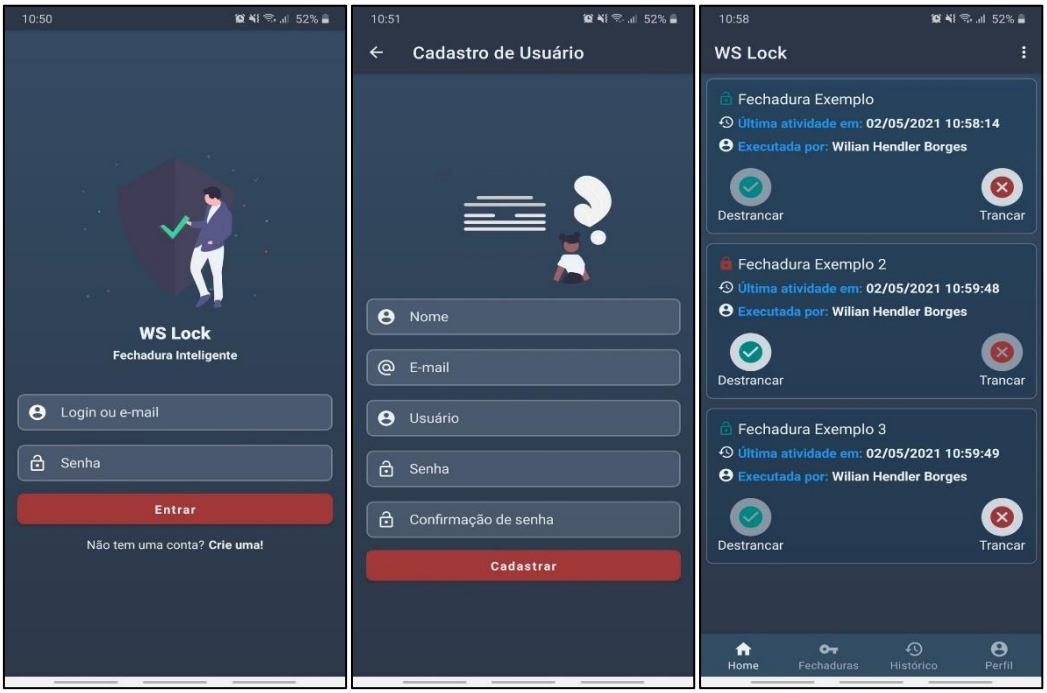

Figura 16. Telas de autenticação, cadastro de usuário e home

As principais funcionalidades foram divididas em cinco abas que podem ser vistas na extremidade inferior, são elas: Home, Fechaduras, Histórico e Perfil.

A Home é a aba principal e nela são listadas todas as fechaduras do usuário autenticado. Cada fechadura listada possui um botão para trancar e outro para destrancar. A partir da ação nestes botões, é feita uma requisição para o módulo intermediador que por sua vez comunica-se com o módulo executor e devolve o resultado para o módulo móvel.

Na aba Fechaduras são listadas todas as que o usuário autenticado possui acesso, permitindo editá-las, excluí-las ou cadastrar novas (figura 17).

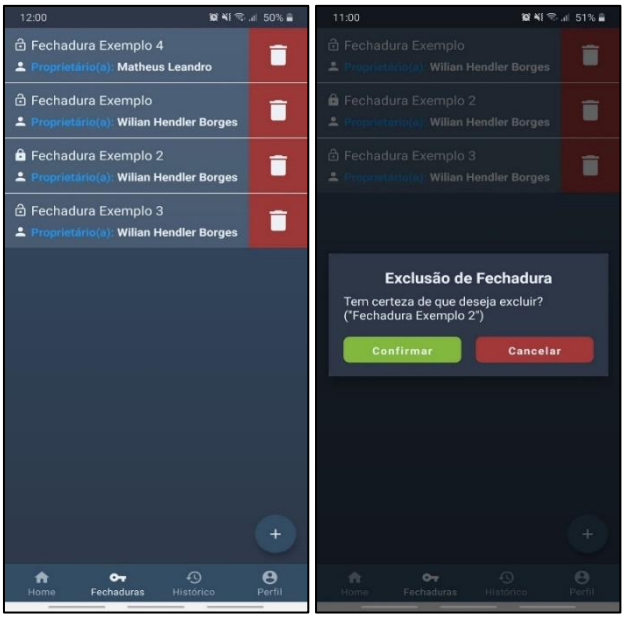

Figura 17. Listagem e exclusão de fechaduras

No formulário de cadastro/edição da fechadura (figura 18) é possível determinar quais usuários têm acesso a ela. No entanto, caso o usuário autenticado não seja o que realizou o cadastro da fechadura sendo editada, ele somente poderá alterar a sua descrição, pois essa informação é vinculada a cada usuário.

Figura 18. Cadastro e edição de fechaduras

Na aba Histórico é possível visualizar todas as ações realizadas em cada fechadura e qual foi o usuário responsável, sendo possível filtrar por uma em específico (figura 19). São exibidas as informações: Fechadura (descrição), Usuário (que realizou a ação), Descrição (Fechadura aberta/trancada) e data/hora.

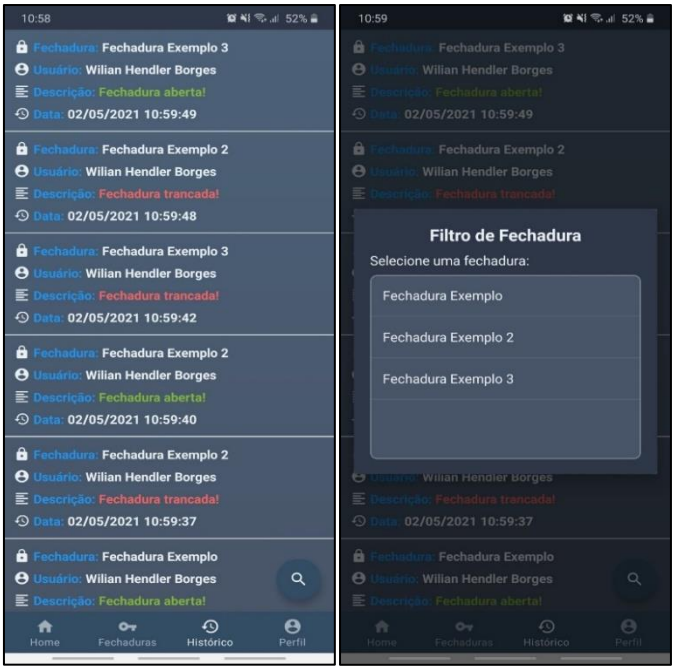

Figura 19. Histórico de ações e filtro por fechadura

Por fim, na aba Perfil, é possível editar as informações do usuário autenticado (figura 20).

Figura 20. Perfil do usuário autenticado

Dessa forma, por meio do aplicativo torna-se possível acompanhar e gerenciar tudo o que acontece no sistema de maneira facilitada e de qualquer lugar desde que conectado a uma rede de internet.

3. Resultados e Discussões

O aplicativo desenvolvido para o módulo móvel mostrou-se eficaz. A biblioteca Retrofit funcionou como esperado e conseguiu realizar a comunicação com o módulo integrador adequadamente. A principal dificuldade foi a comunicação com o módulo integrador para testes na rede local pois, como a API ainda não estava na nuvem, somente era possível realizar a conexão utilizando um emulador na mesma máquina que executava a API. O emulador utilizado é nativo da Android Studio IDE e não tem um desempenho muito eficiente. Para contornar a situação, descobriu-se que era possível executar a API no endereço IP referente a máquina hospedeira e abrir a conexão em uma porta de rede específica. Dessa forma, foi possível executar os testes com um smartphone real conectado a mesma rede Wi-Fi.

No desenvolvimento do módulo executor a ideia era construir um equipamento mais perto do comercial possível, portanto, os componentes utilizados foram escolhidos pensando em ocupar o menor espaço e ter um baixo consumo de energia. No entanto, mesmo assim o circuito final acabou consumindo uma corrente relativamente alta, ficando na casa de 91,6mAh. A bateria escolhida possui capacidade de 2200mAh, sendo assim, consegue manter o módulo alimentado por cerca de 24 horas. Para demonstração do protótipo desenvolvido, não houve nenhum problema com relação a isso, no entanto, comercialmente este resultado não é satisfatório.

Outra dificuldade encontrada deu-se pela escolha da biblioteca e sua respectiva versão para comunicação do ESP8266 com o servidor MQTT. Na maioria das versões disponibilizadas pela biblioteca PubSubClient.h ocorreram erros de compilação ou relacionados a conexão com o servidor. Porém, ao utilizar a versão 2.8.0 tudo funcionou perfeitamente.

Além disso, a conexão do ESP8266 com a rede Wi-Fi também se mostrou como um ponto de preocupação, pois a alteração do código fonte pelo usuário final a fim de informar as credenciais da sua rede era inviável. Felizmente, a biblioteca WiFiManager.h conseguiu quebrar grande parte desta barreira, permitindo ao usuário uma interação mais facilitada. No entanto, o ideal seria que por meio do aplicativo desenvolvido no módulo móvel fosse possível realizar essa conexão sem a necessidade de ficar trocando de rede e abrindo páginas no navegador com endereços IP.

Outro ponto a ser destacado no módulo executor refere-se ao módulo relé utilizado. No site em que ele foi adquirido, não havia nenhuma documentação sobre como realizar a operação dos relés por meio do código presente no ESP8266 e foi extremamente difícil encontrá-la na internet. Por fim, descobriu-se que era necessário realizar a escrita serial de códigos específicos para cada comando em cada relé.

O módulo integrador demonstrou-se extremamente eficiente para o gerenciamento das comunicações entre os módulos executor e móvel, para o tratamento e persistência dos dados e pela autenticação dos usuários. Todas as suas funcionalidades foram devidamente comprovadas utilizando-se dos testes unitários automatizados desenvolvidos.

A instância escolhida para a hospedagem do módulo na nuvem foi capaz de suportar todas as funcionalidades necessárias, apesar de sua configuração limitada, demonstrando que a API, o servidor MQTT e o banco de dados foram também eficientes no que diz respeito ao consumo de recursos computacionais.

O protocolo MQTT disponibiliza um recurso chamado QoS (*Quality of Service*) para verificar se a mensagem publicada no tópico foi recebida corretamente em cada dispositivo inscrito. Este recurso possui três estados representados pelos valores: 0, 1 e 2. No estado 0, o protocolo recebe e propaga a mensagem no tópico somente uma vez sem nenhuma verificação, sendo o método mais rápido. No estado 1, é garantido que a mensagem será recebida por cada assinante pelo menos uma vez, no entanto, podem ocorrer duplicações. Já o estado 2, garante que todas as mensagens sejam entregues e que não ocorra duplicações, porém, é o mais lento devido ao seu *overhead.*

Tanto para o protótipo desenvolvido quanto para a API foi definido o nível de QoS 1, garantindo que a mensagem para trancar ou destrancar a fechadura seja recebida pelo ESP8266. Como este é o nível intermediário, foi possível manter um bom desempenho na execução do serviço e evitar falhas na comunicação.

O maior obstáculo encontrado, neste caso, foi a criação de métodos base utilizados para a construção dos dados encapsulados na resposta das requisições. Esses métodos deveriam ser genéricos para todos os *models* definidos, incluir somente os relacionamentos solicitados pelo cliente e utilizar a estrutura de dados JSON. Para isso, foram sobrescritos, em classes abstratas, alguns métodos das classes *JsonResource* e *ResourceCollection* do *framework* Laravel.

Comparando-se esta pesquisa com as demais já desenvolvidas na mesma área, observa-se que a principal diferença está na possibilidade de comunicação com a fechadura de qualquer lugar do mundo desde que conectado à internet. Outro diferencial observado sob as pesquisas realizadas por Oliveira (2013), Menezes (2018) e Declerque (2014), é a utilização do microcontrolador ESP8266 invés do Arduino. Além disso, o aplicativo desenvolvido para esta pesquisa possui mais funcionalidades como, por exemplo: autenticação de usuários; histórico de ações em cada fechadura; cadastro e acionamento, por usuário, de suas fechaduras; gerenciamento de acesso, por usuário e fechadura, a terceiros que venham a utilizar a fechadura.

4. Conclusão

Nesta pesquisa foi desenvolvido um protótipo de fechadura tecnológica, ativada por meio de um aplicativo móvel com auxílio de um microcontrolador e com a intermediação da comunicação com uma API para maximizar a segurança residencial. Para tanto, foram aplicados conceitos baseados em computação embarcada, computação em nuvem e Internet das Coisas.

Os resultados obtidos foram satisfatórios, tendo em vista que por meio do protótipo desenvolvido tornou-se possível a ativação de uma fechadura de qualquer lugar, bastando estar conectado à internet. Além disso, é possível visualizar um histórico completo de todas as ações que ocorreram na fechadura bem como gerenciar quais usuários tem acesso a ela.

Dessa forma, os cidadãos que optarem pela utilização do sistema poderão ter um melhor gerenciamento de sua residência no que tange a sua permissão e de terceiros as portas de acesso. Sendo assim, considera-se que o objetivo da pesquisa foi atingido.

Embora os resultados tenham sido positivos, faz-se necessário um estudo mais aprofundado principalmente no quesito *hardware*, buscando aprimorar a durabilidade da bateria, diminuir o consumo de energia e construir uma fechadura própria com todos os componentes necessários ao funcionamento do sistema já embarcados.

Com base nos conhecimentos adquiridos, bem como nos resultados obtidos, propõe-se para futuros trabalhos: detectar tentativas forçadas de abrir a fechadura, alertando seus proprietários por meio de notificações em seu smartphone; desenvolver modelo de fechadura próprio com o microcontrolador e demais dispositivos já embarcados; permitir a definição de um intervalo de tempo padrão no aplicativo que servirá para trancar a fechadura automaticamente e, caso a porta não esteja no marco, notificar o usuário; aumentar a autonomia energética do sistema; aplicar testes de utilização em cenários reais.

Referências

ABBOTT, Jonatas; SANDS, Caroline. **Tecnologia Mobile:** Não é vício, é necessidade. 2018. Disponível em: https://www.dinamize.com.br/blog/tecnologia-mobile/. Acesso em: 25 out. 2020.

ANDRADE, Ana Paula de. **O que é uma IDE (Ambiente de Desenvolvimento Integrado)?** 2020. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-uma-ide-ambiente-de-desenvolvimento-integrado/. Acesso em: 12 out. 2020.

ANDRADE, Ana Paula de. **O que é Laravel?** 2019. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-laravel/. Acesso em: 02 nov. 2020.

ARDUINO. **What is Arduino?** 2018. Disponível em: https://www.arduino.cc/en/Guide/Introduction. Acesso em: 26 set. 2020.

BAUERMEISTER, Giovanni. **Guia do Usuário do ESP8266**. 2018. Disponível em: https://www.filipeflop.com/blog/guia-do-usuario-do-esp8266/. Acesso em: 21 nov. 2020.

BECKER, Michael. **Marketing Móvel:** mantendo o iDireto na Palma da mão do seu cliente. In: RAPP, Stan (Org.). Redefinindo o marketing direto interativo na era digital – como aplicar com sucesso conceitos de marketing iDireto e iBranding em seu plano de marketing. São Paulo: M.Books, 2011.

BEGHINI, Lucas Bragazza. **AUTOMAÇÃO RESIDENCIAL DE BAIXO CUSTO POR MEIO DE DISPOSITIVOS MÓVEIS COM SISTEMA OPERACIONAL ANDROID**. 2013. 78 f. TCC (Graduação) - Curso de Engenharia Elétrica, Universidade de São Paulo, São Carlos, 2013.

BRASIL. **Lei nº7.102, de 20 de junho de 1983**. Dispõe sobre segurança para estabelecimentos financeiros, estabelece normas para constituição e funcionamento das empresas particulares que exploram serviços de vigilância [...]. Brasília - DF, 1983. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/l7102.htm. Acesso em: 12 out. 2020.

CARVALHO, Filipe. **Tecnologia mobile é a cultura do novo século**. 2019. Disponível em: https://itforum365.com.br/tecnologia-mobile-e-a-cultura-do-novo-seculo/. Acesso em: 25 out. 2020.

DEITEL, Paul; DEITEL, Harvey; DEITEL, Abbey. **Android**: como programar. 2. ed. Porto Alegre: Bookman, 2015. 728 p. Tradução: João Eduardo Nóbrega Tortello.

DECLERQUE, Maurício Souza. **AUTOMAÇÃO E MONITORAMENTO REMOTO RESIDENCIAL**. 2014. 70 f. TCC (Graduação) - Curso de Tecnologia em Redes de Computadores, Universidade Federal de Santa Maria, Santa Maria, 2014.

FEZARI, Mohamed; DAHOUD, Ali Al. **Integrated Development Environment "IDE" For Arduino**. 2018. Disponível em: https://www.researchgate.net/publication/328615543_Integrated_Development_Environment_IDE_For_Arduino. Acesso em: 03 out. 2020.

G1. **Automação residencial auxilia moradores desde tarefas simples até aumentar a segurança do imóvel**. Disponível em: https://g1.globo.com/sp/sao-jose-do-rio-preto-aracatuba/mercado-imobiliario-do-interior/noticia/2020/04/01/automacao-residencial-auxilia-moradores-desde-tarefas-simples-ate-aumentar-a-seguranca-do-imovel.ghtml. 2020. Acesso em: 26 Set. 2020.

G1. **Em plena quarentena, Brasil tem alta de 8% no número de assassinatos em abril**. Disponível em: https://g1.globo.com/monitor-da-violencia/noticia/2020/06/17/em-plena-quarentena-brasil-tem-alta-de-8percent-no-numero-de-assassinatos-em-abril.ghtml. 2020. Acesso em: 26 Set. 2020.

GARCIA, Fernando. **Introdução aos sistemas embarcados e microcontroladores**. Disponível em: https://www.embarcados.com.br/sistemas-embarcados-e-microcontroladores/. 2018. Acesso em: 27 Set. 2020.

GARTNER. **Gartner Says 8.4 Billion Connected Things Will Be in Use in 2017, Up 31 Percent From 2016**. 2017. Disponível em: https://www.gartner.com/newsroom/id/3598917. Acesso em: 20 Mar. 2020.

GAZETA DO POVO. **Depois de bater 2 milhões de furtos e roubos por ano, Brasil começa a reduzir índices**. 2019. Disponível em: https://www.gazetadopovo.com.br/ideias/epidemia-do-crime-furtos-e-roubos-passam-da-casa-do-milhao-todo-ano-no-brasil/. Acesso em: 28Mar. 2020.

GOOGLE. **O que é o Android**. 2020. Disponível em: https://www.android.com/intl/pt-BR_br/what-is-android/. Acesso em: 26 out. 2020.

GUEDES, Marylene. **Para que serve um framework?** 2019. Disponível em: https://www.treinaweb.com.br/blog/para-que-serve-um-framework/. Acesso em: 02 nov. 2020.

HALFACREE, Gareth. **THE OFFICIAL Raspberry Pi Beginner's Guide**: how to use your new computer. 2018. Disponível em: https://www.raspberrypi.org/magpi-issues/Beginners_Guide_v1.pdf. Acesso em: 03 out. 2020.

HILL, W.; HOROWITZ, P. **A Arte da Eletrônica**. 3ª ed. 2018.

INTERNATIONAL DATA CORPORATION. **Smartphone Market Share**. 2020. Disponível em: https://www.idc.com/promo/smartphone-market-share/os. Acesso em: 27 out. 2020.

JIMBLOM; DUDERINO, El. **Arduino Shields v2**. [ca 2013]. Disponível em: https://learn.sparkfun.com/tutorials/arduino-shields-v2/all. Acesso em: 29 set. 2020.

KAUR, Parmjit; SHARMA, Sumit. **Google Android a mobile platform**: a review. A review. 2014. Disponível em: https://www.researchgate.net/publication/269310487_Google_Android_a_mobile_platform_A_review. Acesso em: 31 out. 2020.

KOLBAN, Neil. **Kolban's Book on the ESP8266**. Texas: Neil Kolban, 2015. 309 p.

Li, Q. **Real-time Concepts for Embedded Systems**. 2003.

MANDARINI, Marcos. **Segurança corporativa estratégica: Fundamentos**. São Paulo: Manoele, 2005. 336 p.

MCROBERTS, Michael. **Arduino básico**. São Paulo: Novatec Editora, 2011. 456 p.

MENEZES, Árly Assis Martins Cordeiro. **Implementação de um sistema para acesso pessoal ao Laboratório de Automação Predial do DECAT**. 2018. 51 f. TCC (Graduação) - Curso de Engenharia de Controle e Automação, Universidade Federal de Ouro Preto, Ouro Preto, 2018.

MONK, Simon. **Programming Arduino:** getting started with sketches. Estados Unidos: McGraw-Hill, 2012. 177 p. Disponível em: http://index-of.es/Varios-2/Programming%20Arduino.pdf. Acesso em: 12 out. 2020.

OLIVEIRA, Plínio Cardoso de. **SISTEMA DE CONTROLE DE ACESSO A AMBIENTES FÍSICOS COM DESTRAVAMENTO DE FECHADURAS ELETRÔNICAS POR MEIO DE NFC (NEAR FIELD COMMUNICATION)**. 2013. 54 f. TCC (Graduação) - Curso de Sistemas de Informação, Centro Universitário Luterano de Palmas, Palmas, 2013.

PAIXÃO, João Roberto da. **O que é Laravel? Porque usá-lo?** 2017. Disponível em: https://medium.com/desenvolvendo-com-paixao/o-que-%C3%A9-laravel-porque-us%C3%A1-lo-955c95d2453d. Acesso em: 02 nov. 2020.

PATTERSON, D. A.; HENNESSY, J. L. **Computer Organization and Design: The Hardware/Software Interface 5th Edition**. Morgan Kaufmann-Elsevier, 2013. 680 p.

PRUDENTE, Francesco. **Automação Predial e Residencial - Uma Introdução**. Rio de Janeiro; Livros Técnicos e Científicos Editora Ltda, 2011. 228 p. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/978-85-216-2024-2/. Acesso em: 20 mar 2020.

QUERO AUTOMAÇÃO. **Como a automação residencial contribui para a segurança da casa? Entenda**. Disponível em: https://www.queroautomacao.com.br/seguranca-da-casa/. 2020. Acesso em: 26 set 2020.

REDHAT. **O que é API?** [ca 2018]. Disponível em: https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces. Acesso em: 02 nov. 2020.

REIS, Fábio dos. **O que são Sensores**. 2018. Disponível em: http://www.bosontreinamentos.com.br/eletronica/curso-de-eletronica/o-que-sao-sensores/. Acesso em: 04 out. 2020.

Robillard, M. P., & DeLine, R. (2011). **A field study of API learning obstacles:** Empirical Software Engineering, 16(6), 703–732.

SINICKI, Adam. **I want to develop Android apps**: what languages should i learn? 2019. Disponível em: https://www.androidauthority.com/develop-android-apps-languages-learn-391008/. Acesso em: 01 nov. 2020.

SOUSA, Alex Luiz de et al. **Proposta de um Sistema de Apoio a Tomada de Decisão para o Monitoramento Remoto de Centrais de Alarme Patrimoniais**. 2011. Disponível em: http://seer.upf.br/index.php/rbca/article/view/1311/1306. Acesso em: 12 out. 2020.

SOUZA, Eduardo. **How Will Home Automation Affect our Future?**. 2019. Disponível em: https://www.archdaily.com/923478/how-will-home-automation-affect-our-future. Acesso em: 25 Mar. 2020.

STALLINGS, W. **Arquitetura e organização de computadores**. Pearson Prentice-Hall, 10ª ed., São Paulo. 2017.

Stylos, J., Faulring, A., Yang, Z., & Myers, B. A. (2009). **Improving API documentation using API usage information:** In Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing (pp. 119–126). Washington, DC: IEEE.

TANENBAUM, A. **Organização Estruturada de Computadores**. 6ª Edição, 2013.

TOCCI, R. J; WIDMER, N.S; MOSS, G. L. **Sistemas Digitais - Princípios e Aplicações**. Prentice-Hall, 11ª ed. São Paulo. 2011.

VALENTE, Jonas. **Brasil é 5ª país em ranking de uso diário de celulares no mundo**. Disponível em: https://agenciabrasil.ebc.com.br/geral/noticia/2019-01/brasil-foi-5o-pais-em-ranking-de-uso-diario-de-celulares-no-mundo. 2019. Acesso em: 20 Mar. 2020.