UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO DE ENGENHARIA ELETRÔNICA

CURSO DE ENGENHARIA ELÉTRICA

LUIS OTÁVIO MANENTE

PEDRO MORENO CRESPO

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE ACESSO COM ARMAZENAMENTO DE DADOS EM NUVEM

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2019

LUIS OTÁVIO MANENTE

PEDRO MORENO CRESPO

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE ACESSO COM ARMAZENAMENTO DE DADOS EM NUVEM

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia Elétrica, do Departamento de Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Sérgio Luiz Stevan Jr.

PONTA GROSSA

2019

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Ponta Grossa
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Eletrônica

Engenharia Elétrica

TERMO DE APROVAÇÃO

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE ACESSO COM ARMAZENAMENTO DE DADOS EM NUVEM

por

LUIS OTÁVIO MANENTE

e

PEDRO MORENO CRESPO

Este Trabalho de Conclusão de Curso foi apresentado em 04 de julho de 2019 como requisito parcial para a obtenção do título de Bacharel(a) em Engenharia Elétrica. O(A) candidato(a) foi arguido(a) pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof(a). Dr. Sérgio Luiz Stevan Junior
Orientador(a)

Prof(a). Drª. Marcella Scoczynski Ribeiro Martins
Membro Titular

Prof(a). Dr. Murilo Oliveira Leme
Membro Titular

Prof. Dr. Josmar Ivanqui
Responsável pelos TCC

Prof. Dr. Sergio Okida
Coordenador do Curso

– O Termo de Aprovação assinado encontra-se na Coordenação do Curso –

Dedicamos este trabalho às nossas famílias e professores, que nos apoiaram para chegarmos até aqui.

AGRADECIMENTOS

Este trabalho não poderia ter sido realizado sem a ajuda de várias pessoas e o apoio prestado pela instituição ao longo desses anos de graduação.

Primeiramente, gostaríamos de agradecer aos nossos familiares, pelo apoio incondicional durante nossa trajetória acadêmica, contribuindo imensamente para que chegássemos até aqui.

À nossa instituição, pelo acolhimento, onde criamos laços de amizade e que nos forneceu toda a assistência durante nossos anos de graduação, visando nosso crescimento pessoal e sucesso profissional.

Ao professor Sérgio Luiz Stevan Jr. pela orientação durante a realização do trabalho.

À todos os professores e amigos, que direta ou indiretamente, nos apoiaram para a conclusão deste trabalho e durante nossa trajetória acadêmica.

RESUMO

MANENTE, L. O.; CRESPO, P. M. **Desenvolvimento de um sistema de controle de acesso com armazenamento de dados em nuvem.** 2019. 86 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2019.

O avanço da tecnologia é notável em diversos setores do nosso cotidiano e uma das inovações que ganha espaço de maneira muito perceptível é a dos dispositivos IoT, que são utilizados para variadas aplicações, em especial a de automatização de processos. Este trabalho tem seu escopo baseado na aplicação de um dispositivo microcontrolado com a gestão de dados em nuvem possibilitando o desenvolvimento de um sistema de controle de acesso à ambientes, isto é, com enfoque na segurança, utilizando de diferentes métodos, como senhas numéricas, leitura de *tags* RFID e autenticação de acesso via aplicativo pela tecnologia *Bluetooth*.

Palavras-chave: IoT. Automatização. Nuvem. Acesso. Segurança.

ABSTRACT

MANENTE, L. O.; CRESPO, P. M. **Development of a access control system with cloud data storage.** 2019. 86 p. Work of Conclusion Course. (Graduation in Electrical Engineering) – Federal University of Technology - Paraná. Ponta Grossa, 2019.

The technology's breakthrough is notable in many sectors in our daily routine and one of the innovations that takes an impressive place in a perceptibly way is the IoT devices, which are widely used for many end-uses, in special those related to the automation's process. This work has its scope based on the application of a microcontroller and a cloud data management system aiming to develop an access control system, that is, focused on safety by making use of variable methods, such as numerical passwords, RFID tags reading and access authentication via application using Bluetooth.

Keywords: IoT. Automation. Cloud. Access. Safety.

LISTA DE SIGLAS

ACL	Asynchronous Connection Less
BD	Base de Dados
EPC	Electronic Product Code
IDE	Integrated Development Environment
IETF	Internet Engineering Task Force
iOS	iPhone Operation System
ISM	Industrial, Scientific, Medical
MIT	Massachussetts Institute of Technology
NFC	Near Field Communication
P2P	Peer to Peer
PWM	Pulse Width Modulation
RFC	Request for Coment
RFID	Radio Frequency Identification
RSA	Rivest-Shamir-Adleman
SCO	Synchronous Connection Oriented
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
URL	Uniform Resource Locutora
UUID	Universally Unique Identifier

LISTA DE ACRÔNIMOS

CoAP	Constrained Application Protocol
IEEE	Institute of Electrical and Electronic Engineers
IoT	Internet of Things
LED	Light Emitting Diode
MANET	Mobile Ad-hoc Network
SPAN	Smartphone Ad-hoc Network
VANET	Vehicular Ad-hoc Network
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WPAN	Wireless Personal Area Network

SUMÁRIO

1 INTRODUÇÃO

O controle de acesso à ambientes é uma das preocupações que impacta à todas as pessoas e locais, em virtude de envolver um fator muito relevante, que é a segurança, seja ela de dados, pessoas, produtos, entre outras.

Na maioria destes ambientes, o uso de chaves tradicionais é o mais comum, entretanto, em termos de praticidade, agilidade e segurança, este uso torna-se questionável. Na atualidade, com o avanço da tecnologia, em especial o de interface entre sistemas físicos e sistemas *web*, além da redução de tamanho de sistemas microcontrolados e a facilidade na implementação, observa-se uma transição entre métodos de acesso padrão para algo mais avançado e confiável.

Nos dias de hoje, o controle de acesso faz uso de variadas ferramentas, e dentre elas, a utilização da tecnologia RFID (*Radio Frequency Identification*) para autenticação assume uma grande amplitude, justificada possivelmente pelo seu baixo custo, como expõe Boccucci (2010). Além desta, a inserção de conceitos de IoT (*Internet of Things*) é cada vez mais difundida, já que por meio dela torna-se possível a conectividade entre sistemas físicos e plataformas digitais, o que possibilita uma interação homem-máquina, permitindo usufruir desta para diversas aplicações.

O presente trabalho tem por enfoque o desenvolvimento de um sistema de controle acesso, por meio da aplicação das tecnologias RFID*,* senhas numéricas, interface com aplicativo via *Bluetooth* e conectividade com uma base de dados (BD) em nuvem, por meio dos conceitos de IoT, com objetivo de melhorar a segurança, eficácia e agilidade.

1.1 JUSTIFICATIVA

A escolha deste projeto como trabalho de conclusão de curso foi realizada mediante a análise atual da situação de como o acesso aos laboratórios do bloco V da Universidade Tecnológica Federal do Paraná, do campus Ponta Grossa, é realizado. Devido ao grande número de laboratórios, a questão da utilização das

chaves é problemática, visto que havia basicamente apenas uma por ambiente, o que tornava o processo de acesso demorado, caso o responsável pela administração das mesmas não estivesse no departamento.

Outro fator importante que levou a elaboração deste projeto foi a questão do monitoramento deficiente do acesso aos laboratórios, visto que não há um controle de horário de entrada e saída, e muitas vezes não era possível identificar quem adentrou a certos laboratórios.

Um outro motivo pelo qual este trabalho tem um alto valor é devido a instituição à qual ele está associado, que sempre busca trazer a inovação tecnológica para o corpo docente e discente, em prol do seu reconhecimento como propagadora de avanços científicos.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo geral é o desenvolvimento de um protótipo para um sistema de controle de acesso. Por meio da utilização de sistema microcontrolados, de múltiplas ferramentas de autenticação e estabelecimento de uma conexão sem fio com um sistema em nuvem para monitoramento dos acessos, através de uma base de dados, com a aplicação de conceitos de IoT.

1.2.2 Objetivos Específicos

- Criar um protótipo para sistema de controle de acesso para múltiplos métodos;
- Criar uma base de dados em nuvem para registro dos acessos, utilizando as planilhas da *Google*;
- Desenvolver um aplicativo para realizar o acesso por meio da tecnologia *Bluetooth*;

- Elaborar um código-fonte para o microcontrolador realizar todo o processo de autenticação dos acessos realizados, controle físico da fechadura e estabelecimento da conexão sem fio com a base de dados;

1.3 ESTRUTURA DO TRABALHO

Este trabalho pode ser dividido nos seguintes capítulos:

O capítulo 2 aborda todo o referencial teórico utilizado, como o aspecto do atual panorama das tecnologias de segurança, os diferentes modelos e protocolos de comunicação, a segurança na IoT e as principais plataformas de prototipagem rápida.

No capítulo 3 é descrito toda a metodologia, materiais utilizados e funcionamento do protótipo desenvolvido, bem como uma tabela de custos para implementação.

O capítulo 4 expõe os resultados do trabalho, com o protótipo final exibido e uma análise dos critérios referentes aos métodos de acesso utilizados, segurança, acessibilidade.

No capítulo 5 são apresentadas as conclusões, bem como as dificuldades encontradas e sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 PANORAMA ATUAL DE TECNOLOGIAS DE ACESSO

Com o constante progresso da tecnologia e a necessidade cada vez maior de proporcionar segurança às pessoas e patrimônios, o mercado de sistemas de segurança eletrônica tem apenas crescido. Uma das evidências de que este nicho alavancou, principalmente no Brasil, é a grande quantidade de empresas prestadoras de serviço de monitoramento em tempo real e de fabricantes de dispositivos de segurança, os quais fazem parte de diversos locais que frequentamos.

2.1.1 O Uso Tradicional de Chaves

Para realizar um comparativo do avanço destas tecnologias no nosso dia a dia, a melhor maneira é avaliar um dos métodos mais utilizados no passado, que ainda permanece no presente em larga escala, mas que perante o desenvolvimento técnico-científico apresenta limitações mais evidentes do que outros métodos: o uso de chaves tradicionais.

A utilização de chaves e a necessidade de segurança formam um paradoxo forte nos dias de hoje, uma vez que é de grande dificuldade criar uma atmosfera segura por meio do uso deste método. Primeiramente, em termos de agilidade e eficácia, as chaves apresentam limitações, já que a probabilidade de ser danificada é elevada e isto pode acarretar na inutilização de ambientes. Além disso, o fato de perder a chave traz à tona um segundo fator complicador, que é relacionado a vulnerabilidade, uma vez que pessoas má intencionadas podem ter acesso à locais, cujo são restritos, e assim colocar em risco vidas e bens. Vale ressaltar também que, em termos de controle de acesso utilizando chaves, não é possível coletar dados sobre quem adentrou certo recinto e em que momento, enfatizando a deficiência desta modalidade.

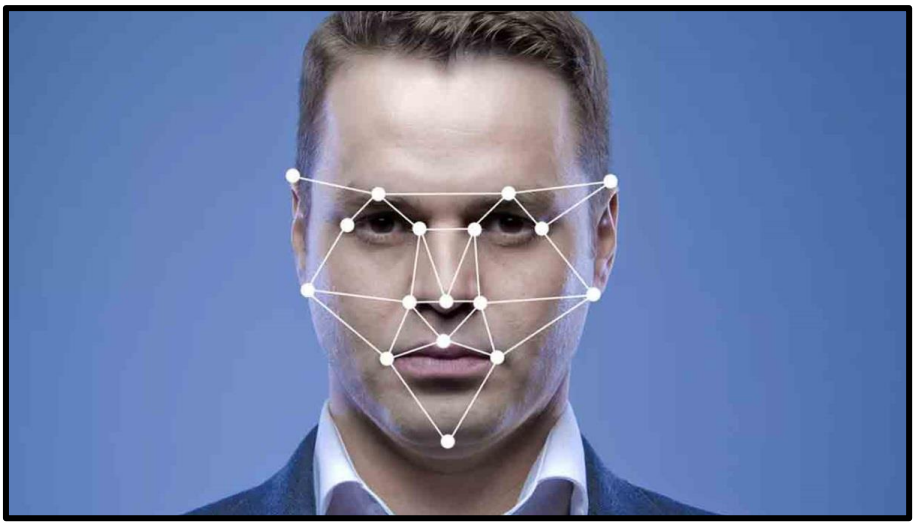
2.1.2 Autenticação Por Métodos Biométricos

Com o progresso tecnológico, o processo de identificação de um indivíduo por métodos biométricos ganhou grande destaque e atualmente, eles fazem parte de muitos sistemas de segurança, tanto no Brasil como no mundo. Segundo Magalhães et al. (2003), “O termo biometria deriva do grego bios (vida) + metron (medida) e, na autenticação, refere-se à utilização de características próprias de um indivíduo para proceder à sua autenticação e/ou identificação [...]”.

Existem diversos métodos biométricos, como o de reconhecimento facial, impressão digital, reconhecimento de voz, leitura de íris e cada um apresenta um alto grau de complexidade. Como expõe Magalhães et al. (2003), o grau de complexidade não só é em virtude da tecnologia empregada para validar os métodos, mas também pela quantidade de informações que os fabricantes fornecem.

O reconhecimento facial é um método que leva em conta para identificação pontos específicos do rosto, como nariz, boca, olhos e não o rosto por completo, em virtude de que o ser humano pode modificar sua aparência pelo simples fato de utilizar um óculos, barba, por exemplo. A figura 1 exemplifica pontos específicos do rosto que são lidos para realização do reconhecimento facial.

Figura 1 - Reconhecimento facial



Fonte: (FIORE; 2018)

Em termos de segurança, este tipo de autenticação possui muitas vantagens sobre o uso de chaves, uma vez que torna a identificação dos indivíduos que

acessam um ambiente mais precisa e de caráter único, pelo fato de os traços físicos de uma pessoa ser inerente apenas a ela. Em termos de controle de acesso, por meio deste método consegue-se integrar diversas funcionalidades, que são capazes de armazenar dados referentes ao usuário que foi ou tentou ser identificado, data e hora, o que caracteriza este sistema como sendo mais confiável e preciso.

2.1.3 Identificação por Cartões Inteligentes

Uma outra maneira de autenticação bastante presente no dia a dia, que compõe um controle de acesso é o uso de cartões inteligentes. O controle de acesso à ambientes não se restringe apenas à escolha de algo mais inovador, mas deve ser levado em conta se o método de controle a ser implantado está de acordo com as diretrizes e requisitos atuais e como usufruir da tecnologia para melhorias do perfil de segurança.

Com o intuito de assegurar maior segurança, privacidade e flexibilidade, a tecnologia embarcada nestes cartões está baseada na criptografia de dados, que são lidos por um leitor, através de uma antena. Mediante a leitura, estes dados são objetos de verificação para autenticação e assim, permitir o acesso ou não à indivíduos. Posteriormente neste trabalho, será abordada com mais detalhes este processo de autenticação pelos cartões, bem como seu funcionamento ilustrado.

O sistema de autenticação por cartões pode ter adoção de recursos adicionais, que reforçam o grau de seguridade do sistema e tornam este tipo identificação confiável para controles de acesso, adquirindo uma vasta utilização no Brasil e no mundo.

2.2 CONECTIVIDADE

O desenvolvimento tecnológico nos dias de hoje, traz cada vez mais à sociedade aplicações em diversos setores, com a inserção de funcionalidades das mais diversas, atingindo o cotidiano das pessoas, o ambiente domiciliar, do trabalho, tornando-o cada vez mais dinâmico e com maiores níveis de praticidade.

Uma das percepções mais observadas no avanço acentuado da tecnologia é o alto grau de conectividade entre os dispositivos que estão disponíveis para o uso,

permitindo elevar, com qualidade e variedade, o caráter de entretenimento. Como exemplo, por meio da expansão de aplicativos destinados à *streaming* de vídeos e músicas, jogos, mensagens instantâneas, que de maneira exponencial tornam-se cada vez mais inerentes à realidade em que vivemos. Além do entretenimento, o ambiente de trabalho experimenta uma transição acentuada nos métodos de se produzir, observada pela gama de ferramentas que progresso na era digital trouxe, sendo possível hoje monitorar por completo uma cadeia produtiva de diversas filiais de uma certa empresa sem a necessidade de estar presente fisicamente, realizar reuniões com alto grau de confiabilidade e dinamismo, como se estivessem sendo executadas pessoalmente e até mesmo substituir imensos escritórios, que exigem um elevado grau de investimento, por uma simples sala dentro da própria casa, conhecidos como *home offices*, nos quais consegue-se produzir e interagir fortemente com diversas pessoas nos mais variados locais.

Figura 2 - Horizontes da conectividade

Fonte: (RODRIGUES; 2017)

Os horizontes que a conectividade é capaz de atingir são diversos, como mostra a figura 2 e para cada um deles, a inovação segue um crescimento ascendente, de tal forma que em um curto período de tempo somos apresentados à novas tendências e assim, nos adaptamos para usufruir das melhorias, transformando nosso modo de viver. Uma das novas tendências desta relação entre a conectividade e como ela está ligada ao cotidiano das pessoas e aos ambientes

que essas estão inseridas e que está em pauta em muitos projetos atuais, envolvendo aplicações em variados espectros, é o conceito de IoT.

2.2.1 Conceito de Internet das Coisas

Nos dias de hoje, a internet está presente em grandes proporções no cotidiano das pessoas, servindo como meio para realização de tarefas e permitindo que haja uma comunicação entre as pessoas.

De acordo com Figueira (2016), a internet surgiu como um experimento financiado pelos militares norte-americanos, com o intuito de estabelecer um meio de troca de informações entre computadores de médio e grande porte para compartilhamento de processamento dos seus equipamentos. Com o avanço em paralelo da ciência e da tecnologia, a dimensão da rede aumentou, atingindo escalas globais, o que permitiu não somente a interação entre máquinas, mas também entre pessoas e atualmente, entre objetos e computadores, caracterizando a fase da IoT.

> Pode-se dividir a história da internet em 3 fases: a primeira é a fase da internet como uma rede de computadores, com o tempo ela evoluiu para sua segunda fase, onde se tornou uma rede de pessoas e comunidades. Atualmente está acontecendo o nascimento da terceira fase, que é chamada fase da Internet das Coisas (do inglês, *Internet of Things* – IoT). Nessa fase a internet não mais interliga apenas computadores, ela agora passa a interligar vários tipos de objetos e dispositivos inteligentes que irão interagir com as pessoas, tornando o dia a dia mais fácil. (FIGUEIRA, 2016).

Com o surgimento deste conceito, a interação entre dispositivos passou de um patamar de uma simples conectividade para um estágio em que a conexão entre eles ocorre e ao mesmo tempo é formado um sistema, o qual é capaz de reagir mediante a aquisição e interpretação de dados, caracterizando-o como inteligente. Segundo Rodrigues et al. (2016), este conceito de Internet das Coisas é interdisciplinar, tangenciando a área da linguagem, comunicação, tecnologia e rede e que na sociedade atual tem respondido com resultados positivos, podendo ser considerado por muitos indivíduos como o terceiro grande ápice da indústria da informação.

2.2.2 Modelos de Comunicação

A IETF (*Internet Engineering Task Force*) é uma organização destinada à designers de rede, assim como pesquisadores e operadores que se preocupam com os avanços da arquitetura da internet e por meio das RFC (*Request for Coments*)**,** que são documentos comprometidos a fornecer dados técnicos e organizacionais, contribuem para a compreensão destes avanços. No âmbito da IoT*,* em março de 2015 foi publicada a RFC 7452, entitulada *Architectural Considerations in Smart Object Networking* e com ela foi estabelecido quatro modelos básicos de comunicação possíveis: dispositivo para dispostivo, dispositivo para nuvem, dispositivo para *gateway* e compartilhamento *back-end*.

2.2.2.1 Dispositivo para dispositivo

Neste modelo de comunicação, dois dispositivos são conectados por uma rede *wireless*, além de poder integrar protocolos de comunicação variados, como o *Bluetooth, Z-Wave, Zigbee.*

2.2.2.2 Dispositivo para nuvem

Neste tipo de comunicação, a comunicação é feita diretamente entre um dispositivo alvo e o servidor em nuvem, o que torna dispensável a utilização de um segundo equipamento para estabelecer a comunicação. Uma ressalva exposta pela RFC 7452 referente ao presente modelo é de que o uso de tecnologias como WLAN (*Wireless Local Area Network*) em conjunto com tecnologias de autenticação de acesso à rede pode ser necessário, em virtude de que às redes de acesso que os dispositivos inteligentes estão conectados nem sempre estão sob controle do servidor em nuvem.

2.2.2.3 Dispositivo para *gateway*

De acordo com a RFC 7452, este modelo de comunicação mantém um servidor em nuvem, como o previamente citado, que administra um fluxo de dados, entretanto, possui como diferencial a presença de um dispositivo intermediário, denominado *gateway,* entre o sistema em nuvem e o dispositivo à quem esse está

conectado. Além dessa função de realizar o intermédio entre as duas pontas do sistema, com o *gateway* pode-se usufruir de funcionalidades que se tornam essa comunicação mais robusta, como segurança e protocolos de tradução.

Em questões estruturais, para se construir uma comunicação deste tipo, geralmente opta-se por *gateways* do mesmo fabricante que os dispositivos IoT à quais ele está ligado, com a finalidade de excluir problemas de interoperabilidade, porém, a tendência, segundo é relatado na RFC em questão, é que o uso dos *gateways* passe a ser genérico, isto é, o fabricante não irá mais ser relevante, e para isso, deve-se migrar para o uso de protocolos de internet genéricos, tornando a necessidade de protocolos de tradução facultativa, o que acarreta em uma redução da complexidade estrutural e de custos.

2.2.2.4 Compartilhamento *back – end*

Este modelo é uma expansão do modelo de dispositivo para nuvem, pois até o primeiro momento era possível apenas cada servidor em nuvem interagir com um único dispositivo da IoT. Entretanto, com a expansão, torna-se capaz um certo servidor interagir com outros servidores, permitindo, para certas aplicações, uma combinação de dados, visando uma maior acuracidade na informação.

2.2.3 Protocolos de Comunicação

Os sistemas IoT, para seu correto funcionamento, dependem diretamente de um adequado estabelecimento da comunicação entres os dispositivos, visando uma transferência de dados eficiente. Uma das entidades responsáveis pela criação de normas e padrões é a IEEE (*Institute of Electrical and Electronic Engineers*) e no quesito de redes, o comitê 802 detém da responsabilidade do desenvolvimento das mesmas.

Como a maioria das aplicações envolvem comunicação sem fio, os padrões IEEE 802.11, que está relacionado à rede WLAN, IEEE 802.15 referente à rede WPAN (*Wireless Personal Area Network*), IEEE 802.16 referente à WMAN (*Wireless Metropolitan Area Network*) e P2P (*Peer to Peer*) são os mais utilizados, em virtude de questões como flexibilidade e alcance serem relevantes, além de que o conceito IoT pretendeu não modificar a estrutura da internet utilizada em ambientes

empresariais e residenciais, por se tratar de algo já padronizado internacionalmente, como expõe Leite et al. (2017).

A seguir são apresentadas as tecnologias mais utilizadas com os padrões IEEE previamente citados, bem como suas características que as distinguem umas das outras.

2.2.3.1 Padrão IEEE 802.11

Este padrão de comunicação faz referência a tecnologia *Wi-Fi*, presente em diversas aplicações IoT e comumente utilizada para uso da internet sem fio em ambientes residenciais e comerciais. Detém de características técnicas como sendo uma tecnologia baseada na rádio frequência por difusão, alcance de médias distâncias, alta qualidade e flexibilidade. Na tabela 1, mostra-se dados técnicos referentes às versões do padrão 802.11, para efeitos de comparação.

Tabela 1 – Dados comparativos do padrão IEEE 802.11

Versão do padrão 802.11	Frequência (GHz)	Largura de Banda (MHz)	Taxa de transferência de dados (Mbps)	Alcance aproximado (m)
a	5	20	54	120
b	2,4	20	11	140
g	2,4	20	54	140
n	2,4/5	20	100	250

Fonte: (WELEKAR et al.; 2012)

No que se refere a utilização das diferentes versões do padrão apresentado, a versão 802.11b é mais amplamente difundida, presente em uso doméstico e em pequenas empresas, sendo seu alcance uma das principais vantagens. A nova tendência é de que cada vez mais fabricantes desenvolvam produtos com o padrão 802.11n, como já é feito pela *Apple*, nos modelos de *iPhone* da quarta geração e alguns modelos de *MacBooks*, permitindo que seja possível a operação em dois níveis de frequência, ter um ótimo alcance e realizar transferência de dados de 65 a 100 Mbps, o que torna essa versão diferenciada em relação às demais, com base em seus benefícios.

2.2.3.2 Padrão IEEE 802.15

Este padrão engloba três conceitos muito importantes no âmbito da IoT, que são a rede AdHoc e as tecnologias *Bluetooth* e *ZigBee*. Primeiramente, será abordado as características singulares da rede AdHoc.

Esta rede foi concebida mediante a análise de como seria realizada a comunicação entre dispositivos em um ambiente militar, onde tais dispositivos estão em constante movimento e necessitam trocar constantemente informações.

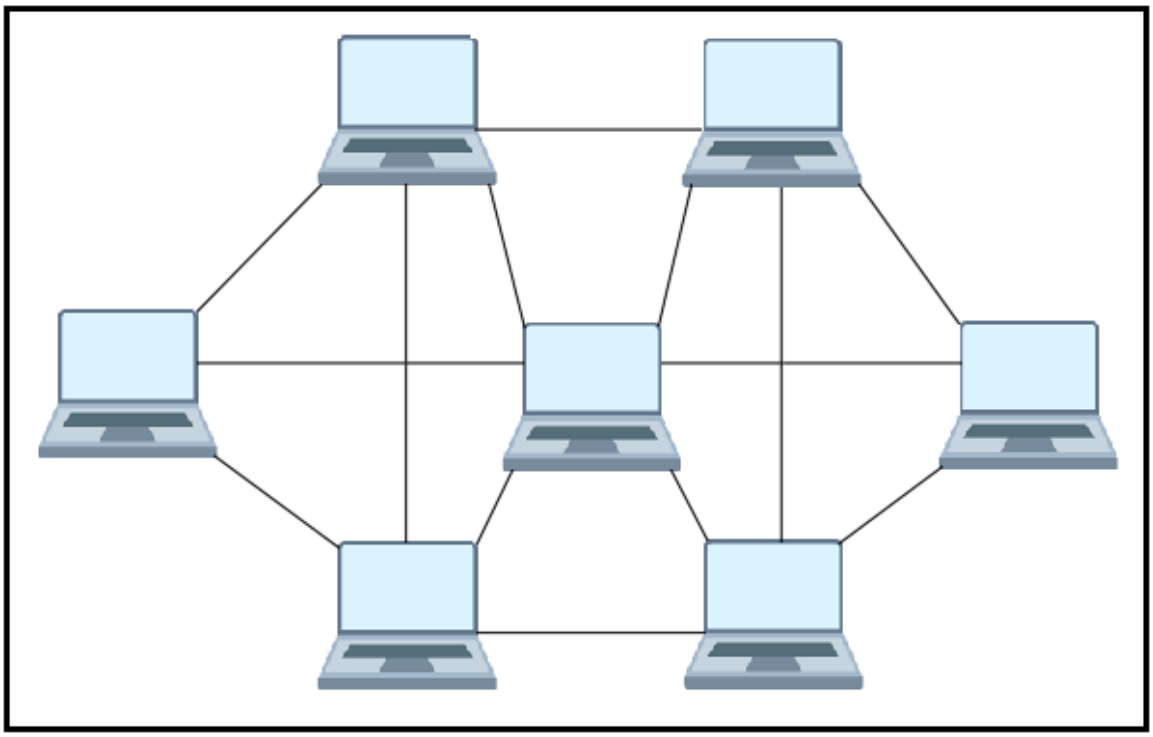
> Basicamente, a comunicação se dá ponto a ponto e esse tipo de rede traz grandes desafios devido à topologia altamente dinâmica da rede, pois todos os nodos podem estar em movimento e podem ter comunicação intermitente. (FIGUEIREDO et al., 2003).

Neste tipo de rede não há um nó específico à qual o fluxo de dados converge e posteriormente o encaminha para os respectivos destinatários, mas sim uma descentralização, em que todos os dispositivos estão interligados e estão aptos a estabelecer a comunicação entre si, como mostra na figura 3. Vale ressaltar que o termo nó não se restringe apenas à computadores, mas também se pode expandir a dispositivos periféricos, como impressoras, celulares, scanners, entre outros, desde que haja suporte para tal topologia.

Dentre as aplicações da rede AdHoc, Leite et al.(2017) elenca diversas opções como MANETs (*Mobile Ad-Hoc Network*), VANETs (*Vehicular Ad-Hoc Network*), SPANs (*Smartphone Ad-Hoc Network*) que são redes cujos nós correspondem a veículos, *smartphones*, por exemplo, que trocam informações entre si.

Assim como no padrão IEEE 802.11, no padrão IEEE 802.15 também existem diversas versões e a rede AdHo*c* pertence à versão 802.15.4. Após apresentar as tecnologias presentes na IoT que empregam o padrão 802.15, uma tabela comparativa irá destacar as características técnicas mais relevantes de cada versão.

Figura 3 - Topologia de uma rede AdHoc



Fonte: Adaptado de Moraes et al.(2009)

Iniciando as tecnologias pertencentes a este padrão, primeiramente temos a *Bluetooth* e leva esse nome em homenagem ao rei Harold Blatand, que teve a proeza de unificar os povos escandinavos, e assim pôde-se relacionar à tecnologia, já que um de seus impactos é a unificação de diferentes dispositivos. Esta tecnologia foi desenvolvida em 1994, pela *Ericsson* e tem o padrão IEEE 802.15.1 como responsável por toda a comunicação e está presente em variadas aplicações, que requerem um consumo de energia baixo e não necessitam de um grande alcance.

Dos tipos de dispositivos que podem se conectar utilizando esta tecnologia, existe uma gama deles, como computadores, fones de ouvido, eletrodomésticos, impressoras, entre outros e para tanto, uma rede denominada *Piconet* é necessária, derivada da rede AdHoc previamente discutida. Como expõe Siqueira (2006), uma rede *Piconet* é aquela formada por um mestre e os escravos interligados à ele, por meio de uma conexão *Bluetooth.* Em termos de capacidade, pode-se conectar em torno de 8 dispositivos e caso seja requisitado a ampliação para mais deles, migra-se para uma rede *Scatternet*, que nada mais é que duas ou mais *Piconet* sobrepostas, onde um ou mais dispositivos são terminais em comum.

O meio pelo qual a comunicação é realizada baseia-se no da rádio frequência e pelo fato dela ser desenvolvida para funcionar em todo o mundo, uma frequência de rádio aberta é necessária. Para tanto, a faixa ISM (*Industrial, Scientific, Medical*) foi definida como padrão, variando de 2,4 a 2,5 GHz.

O princípio básico de funcionamento do *Bluetooth* está associado à designação do dispositivo que iniciou a conexão como sendo mestre e os demais como escravos. Em virtude desta tecnologia permitir o envio e recebimento de dados, é necessária uma adequação perante a função de cada um na rede, distribuindo entre os dispositivos *slots* de envio ou de recebimento, além estabelecer o sincronismo, sendo que ambas as tarefas estão atribuídas ao mestre. Em termos do sincronismo, pode-se ter dois padrões diferentes: SCO (*Synchronous Connection Oriented*) e ACL (*Asynchronous Connection Less*).

Segundo Nakayama (2012), o padrão SCO corresponde à uma comunicação baseada no link ponto a ponto entre um mestre e um único escravo, de tal forma que o envio de dados, predominantemente áudios, ocorre em intervalos de tempo específicos. Ainda, é importante ressaltar que neste enlace, a retransmissão de dados não ocorre e caso haja a identificação problemas no pacote enviado, o sistema pode tentar a correção por meio de múltiplos envios até ser recebido a informação correta. Já no padrão ACL, o mesmo autor expõe que, de maneira diferente ao SCO, a comunicação é baseada em em um link ponto a multiponto, isto é, um mestre está conectado à todos os escravos ativos do sistema, a transmissão de dados é assíncrona e para correções de dados enviados, a retransmissão torna-se possível.

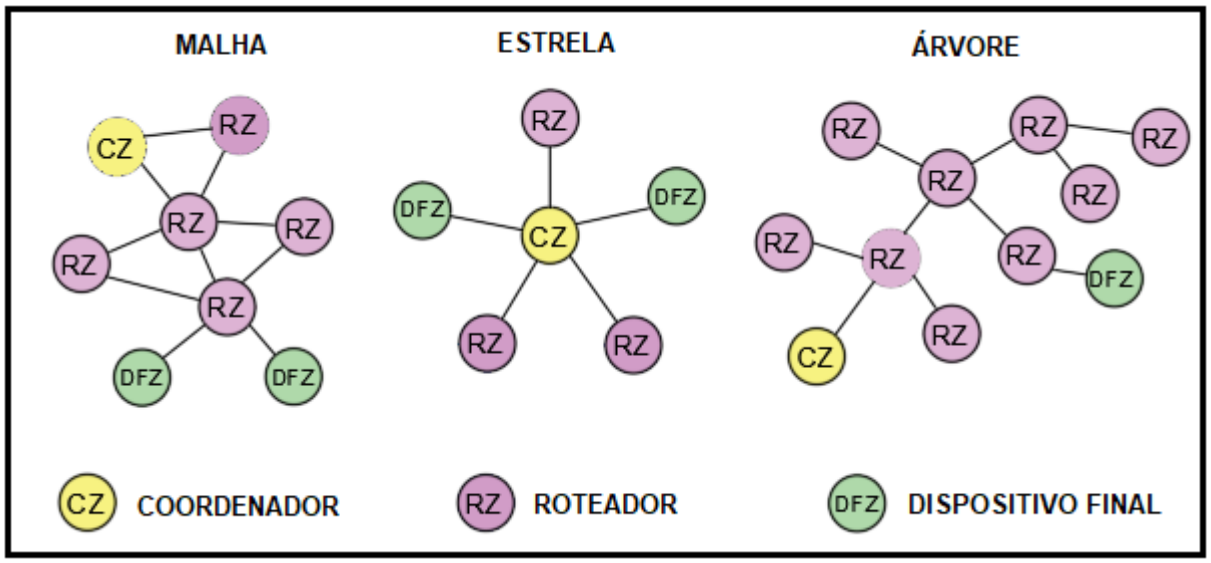
Atualmente, é comum deparar-se com diversas versões desta tecnologia e o avanço delas dá-se nas melhorias de fatores como taxa de transferência de dados, melhor controle do consumo de energia, inovações em procedimentos de segurança, entre outros. A versão *Bluetooth* 4.1 em especial e em concordância com o objeto de estudo deste trabalho trouxe à tona recursos para maior aplicação à dispositivos móveis, contribuindo para a incorporação de mais uma tecnologia à rede IoT.

Outra tecnologia que deriva deste padrão IEEE é a *ZigBee*, que é um avanço na área da comunicação entre dispositivos e que ainda está em fase de desenvolvimento. Assim como a tecnologia previamente apresentada, o padrão

IEEE 802.15.4 é o que rege a comunicação nesta tecnologia e os motivos pelos quais despertou o interesse por ela vão de encontro a necessidade de de um modelo com baixo custo, baixo consumo de energia e que possibilite um monitoramento e controle eficiente, como descreve Welekar et al. (2012).

A rede AdHoc é a base da topologia desta nova tecnologia e essa tem por características singulares o baixo alcance, ser *wireless* e baixas taxas de transferência de dados. Em termos de disposição da rede *ZigBee*, temos as estruturas em malha, estrela ou árvore como possíveis, como mostra a figura 4, e a presença de três elementos chaves: coordenador, roteador e dispositivo final. O coordenador é o responsável por criar a rede e ser o único elemento capaz de fazer a comutação de dados entre redes. Já o roteador é o elemento intermediário para realizar a troca de informações entre os dispositivos da rede. E o dispositivo final tem por única função receber comandos e executá-los, sem a possibilidade de estabelecer comunicação com outros elementos.

Figura 4 – Topologias de rede *ZigBee*



Fonte: Adaptado de Gomes et al.(2013)

Mediante a apresentação de algumas tecnologias referentes ao padrão IEEE 802.15, a tabela 2 a seguir mostra alguns dados técnicos referentes às versões.

Tabela 2 - Dados comparativos do padrão IEEE 802.15

Versões do padrão *IEEE* 802.15	Frequência (GHz)	Taxa de transferência de dados	Alcance aproximado (m)
802.15.1	2,4	3 Mbps	100
802.15.3	2,4	110 Mbits	10
802.15.4	2,4	250 kbps	75

Fonte: (WELEKAR et al; 2012)

2.2.3.3 Padrão IEEE 802.16

Por meio deste padrão, fatores como taxa de transferências de dados, bandas de frequência e alcance não se tornaram mais problemas no quesito comunicação, em virtude de que pode-se atingir cerca de 1 Gbits/s, 11 GHz e até 50 km, respectivamente, evidenciando o caráter de dimensões metropolitanas, como expõe Welekar et al.(2012).

Com os benefícios técnicos que este padrão traz ao sistema, o mesmo é considerado como o de maior custo-benefício por fornecer um seguro e confiável fluxo de dados, integrando diversas aplicações no ramo empresarial, institucional e municipal.

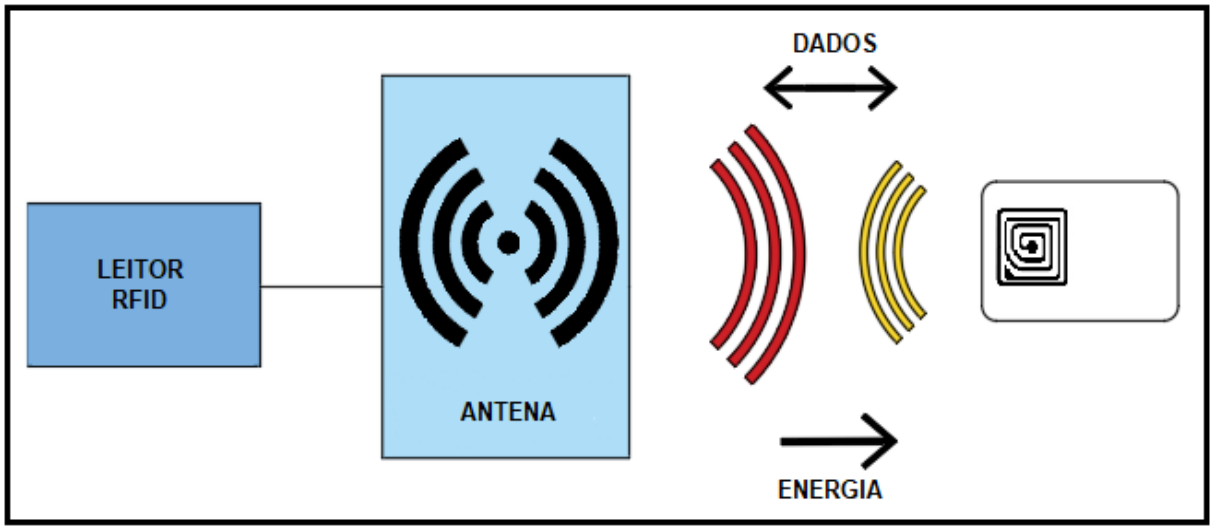
2.2.3.4 Tecnologias RFID e NFC

Estas duas tecnologias pertencem à um tipo de rede de comunicação denominada P2P, na qual cada ponto da rede é capaz de executar a função como cliente ou servidor, permitindo que não seja necessário um servidor central para realizar a troca de informações. No âmbito da IoT, estas duas tecnologias estão presentes em larga escala, como em pagamentos móveis, controles de acesso e este trabalho fará uso delas, de forma a contribuir para a comunidade científica e atender os requisitos do sistema a ser implementado.

A tecnologia RFID utiliza de uma *tag*, que encapsula uma antena e um microchip e nele está contido informações referentes à identificação, por meio do EPC (*Electronic Product Code*). Para realizar a leitura dos dados encapsulados, o sistema requer de um leitor, que por meio da emissão de sinais de rádio frequência e proximidade da *tag*, um campo magnético é estabelecido, energizando a mesma e

assim, as informações são repassadas à um mediador, cujo é responsável por filtrar e agrupar os dados. Posteriormente, estes dados são enviados à um dispositivo para manipulação dos dados, geralmente um servidor. Em questão das *tags*, essas podem ser passivas, isto é, não possuem uma fonte de energia própria e somente são energizadas quando sofrem a ação do campo magnético do leitor; podem ser semi-passivas, nas quais possuem uma fonte de energia própria, mas somente operam quando a *tag* é requisitada e podem ser ativas, nas quais possuem uma fonte própria e podem iniciar a comunicação com o leitor independente se forem requisitadas ou não (LEITE et al., 2017; JUELS, 2005). A figura 5 exemplifica o processo de leitura de uma *tag* RFID.

Figura 5 - Processo de leitura RFID



Fonte: Adaptado de Tizyi et al. (2015)

A tecnologia NFC é muito semelhante à previamente discutida e tem suas diferenças baseadas no alcance e estilo de comunicação. Em questão de alcance, na RFID esse pode atingir dezenas de metros, o que favorece a aplicação para variados cenários, ao passo que para a NFC (*Near Field Communication*), o alcance é limitado a 10 cm no máximo. Apesar deste curto alcance, benefícios na segurança emergem, em virtude de tornar o sistema menos susceptível à ataques, como relata Sab et al.(2013). Segundo os mesmos autores, no quesito de estilo de comunicação, enquanto na RFID a comunicação é realizada exclusivamente em um único sentido, isto é, o leitor interroga a *tag* e essa responde a aquele, na NFC temos a possibilidade de um mesmo dispositivo atuar como leitor ou *tag,* evidenciando uma troca verdadeira de dados.

2.2.4 Segurança na IoT

Com a expansão das redes IoT cada vez mais expressiva, o quesito segurança passou a ser questionado ainda mais. A razão pela qual a vulnerabilidade do sistema torna-se mais intensa com a presença desses novos dispositivos deve-se ao fato de que os mesmos são portas relativamente fáceis para ataques de vírus, comprometendo a segurança das informações que trafegam entre o dispositivo e um servidor.

A facilidade com que possa ocorrer um acesso indevido ao sistema tem relação com a baixa proteção que sensores, por exemplo, apresentam, em virtude da ausência na maioria dos casos, de softwares destinados à proteção contra vírus e desta forma permite que seja formada uma rota disponível para o ataque. Além disso, como a tendência é a utilização cada vez maior de smartphones para realizar a interface com os dispositivos IoT, nem sempre os níveis de confiabilidade nestes pontos de acesso atingem patamares desejáveis e torna-se assim necessário a utilização de mecanismos externos de segurança, como expõe Figueira (2016).

Dois mecanismos muito utilizados, que estão associados tanto ao gerenciamento de quem está acessando o sistema quanto à codificação da informação transmitida e ou recebida é a utilização de base de dados e a criptografia, que serão explanados nas seções adiante.

2.2.4.1 Base de dados

Com o crescimento da IoT, a maneira pela qual os dispositivos se relacionam com a rede e interagem com o nosso dia a dia muda constantemente, inundando o sistema com uma quantidade muito volumosa de dados. Estes dados necessitam ser armazenados, processados e recuperados mediante alguma solicitação e devido à essas três necessidades que surgiram as BD.

Em certas aplicações, a utilização dessas bases não só está atrelada ao gerenciamento do fluxo de dados, mas também como um meio de segurança. Por meio da visualização dos registros, é possível extrair informações úteis para um controle sobre o sistema, por exemplo, identificação de quem acessou, data e hora, verificação de movimentação de bens por meio de um pré cadastro, entre outras.

A segurança por trás desses sistemas pode ser estabelecida pela utilização de diversas regras e que, mediante a necessidade de recuperação de alguma informação, tal requisição é comparada com assertivas preestabelecidas. A seguir, são apresentados alguns tipos de métodos e definições que envolvem a segurança em base de dados, segundo Macêdo (2011):

- Controle de acesso: engloba as diretivas impostas pelo administrador do BD no quesito de concessão ou remoção de funcionalidades à certos usuários e atribuição de um nível de segurança aos mesmos;
- Controle de inferência: este tipo de controle atua sobre as bases de dados estatísticos, que detém de informações de um certo grupo. Pelo fato de este grupo conter informações sigilosas de certos indivíduos, este método assegura que o nível de dados extraídos seja limitado, visando inibir o vazamento de dados confidenciais;
- Controle de fluxo: este mecanismo de segurança está focalizado no estabelecimento de um fluxo unidirecional entre camadas que detém de informações vinculadas umas às outras, isto é, inibe que informações de certo nível crítico, proveniente de uma ou mais intermediárias, seja obtida indiretamente sem que seja possível obter diretamente o conteúdo das subcamadas;
- Usuários: por meio do cadastro de um número específico de usuários estipulado de acordo com a necessidade da aplicação, permite-se que apenas esses tenham acesso às informações contidas no BD. Este cadastro geralmente está associado à um nome e senha, elevando o grau de segurança do sistema;
- Domínio de segurança: entende-se por domínio de segurança um conjunto de propriedades que determinam o que e quanto o usuário pode usufruir do sistema. Dependendo da função desempenhada, esse domínio pode ser expandido ou reduzido;
- Autoridade: consiste no agrupamento de privilégios e níveis de acesso dos administradores e operadores do BD quando comparadas com a manutenção e operações permitidas a serem executadas;
- Privilégios: correspondem a permissões únicas para um certo usuário ou grupo e por meio delas, o nível de profundidade que pode ser

atingido no banco de dados varia, habilitando menos ou mais recursos para serem usufruídos e;

- Classes de segurança: as classes de segurança são estratificadas em altamente sigilosa, secreta, confidenciais e não classificada. Esses estratos são utilizados para a definição das regras para o controle de acesso ao BD, de acordo o valor dos dados armazenados.

2.2.4.2 Criptografia

Uma outra técnica largamente utilizada para fortalecer a segurança eletrônica de dados é a criptografia. Ela está associada à manipular o dado que está armazenado ou que trafega no sistema, de tal forma que o torna ilegível para qualquer pessoa não autorizada, a não ser o destinatário, que detém de uma chave, capaz de decifrar o conteúdo em sigilo.

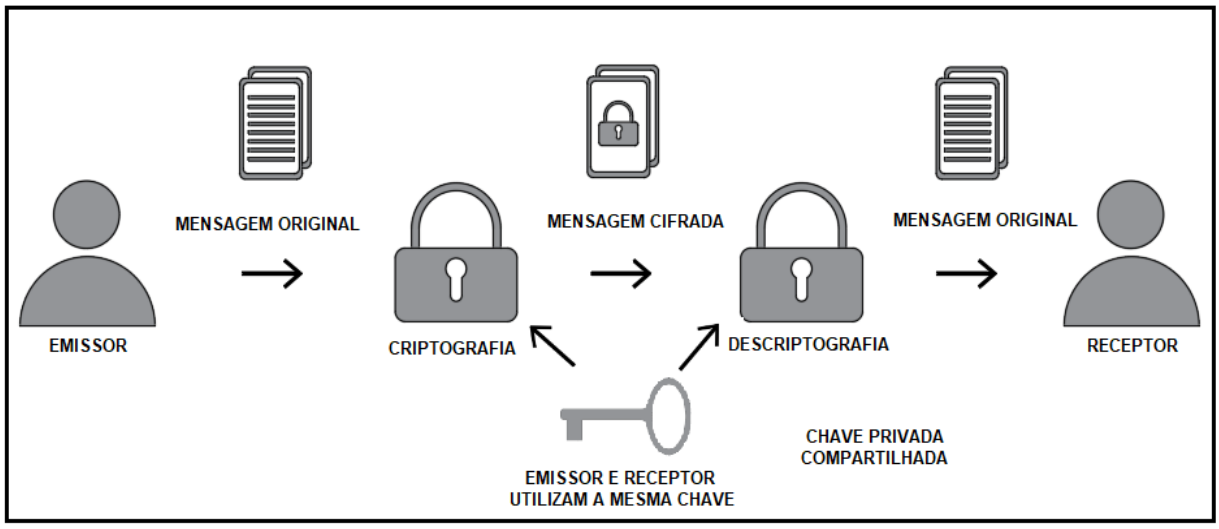
A criptografia está sustentada por quatro objetivos, segundo Macêdo (2011), que são:

- Confidencialidade da mensagem**:** apenas o destinatário deve ser capaz de obter o conteúdo da mensagem;
- Integridade da mensagem**:** o destinatário deve possuir a funcionalidade de verificar se a mensagem foi alterada ou não;
- Autenticação do remetente**:** o destinatário deve ser capaz de identificar o remetente e verificar se foi ele mesmo que enviou a mensagem;
- Irretratabilidade do emissor**:** o emissor não pode negar a autoria da mensagem enviada.

Dentre as chaves criptográficas existentes, pode-se listar duas possibilidades: chaves simétricas e assimétricas.

As chaves simétricas levam esta denominação pelo fato de serem as mesmas tanto para a criptografia quanto para decriptografia. Para realizar o uso delas, basta que os integrantes do sistema permitidos a conhecer o conteúdo das mensagens utilizem uma chave específica e um algoritmo de criptografia ou decriptografia para mascarar a mensagem a ser enviada. A figura 6 ilustra o criptossistema utilizando chaves simétricas.

Figura 6 - Chaves simétricas



Fonte: Adaptado de E-VAL Tecnologia

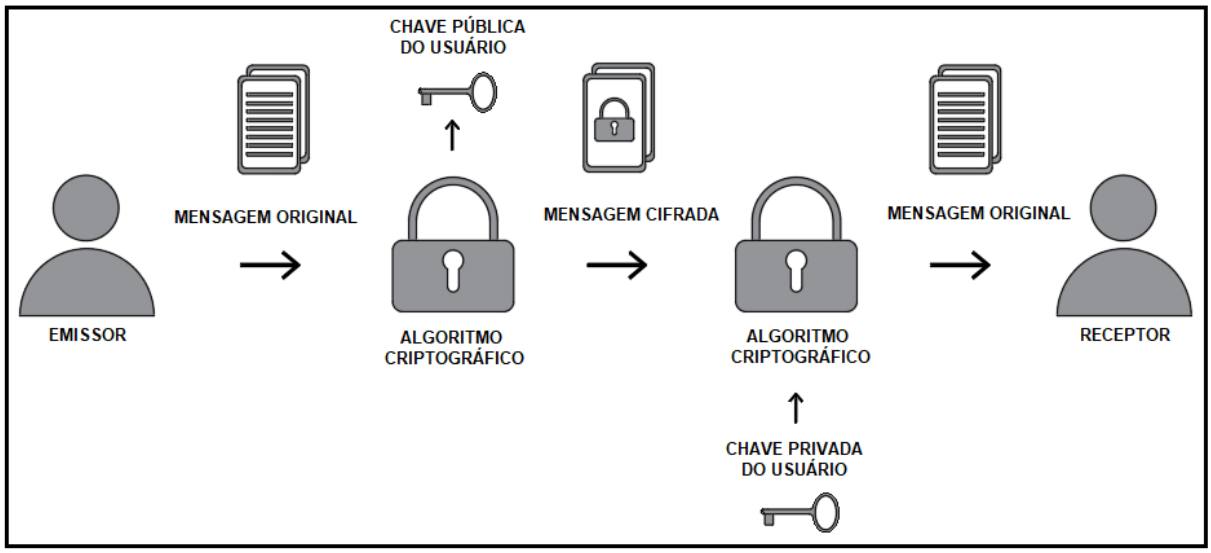
Em respeito aos algoritmos utilizados com essas chaves, utiliza-se dois principais: cifras de bloco e cifras de fluxo.

As cifras de bloco são algoritmos que realizam o processo de criptografia e decriptografia com blocos de um número específico de *bytes*. Em questão de aplicação, esses algoritmos são muito utilizados com dados que possuem um tamanho predeterminado, entretanto deve-se atentar para estabelecimento de padrões repetitivos.

As cifras de fluxo, por outro lado, são algoritmos que se dedicam à cada bit da mensagem, o que os tornam mais rápidos. Vale ressaltar que uma chave secundária é criada por meio da criação de uma sequência de bits, derivada de uma chave simétrica e toda a mensagem é verificada utilizando a operação lógica XOR em conjunto com a chave secundária.

Diferentemente do caso anterior, a criptografia por chaves assimétricas é composta por uma chave pública, que é disponibilizada para qualquer pessoa e uma chave privada, que somente é de posse do dono. Ambas as chaves são distintas e estão matematicamente relacionadas e somente o destinatário com a chave privada correspondente que é capaz de decriptografar a mensagem enviada. A figura 7 mostra a dinâmica do processo com chaves assimétricas.

Figura 7 - Chaves assimétricas



Fonte: Adaptado de E-VAL Tecnologia

Os algoritmos de criptografia utilizados em chaves assimétricas mais conhecidos são: curvas elípticas, RSA (*Rivest-Shamir-Adleman*), El Gamal, Diffie – Hellman. Dentre eles, o mais utilizado no mundo é o RSA, desenvolvido no MIT (*Massachusetts Institute of Technology*), em 1977. O princípio dele é o da multiplicação de números primos para obter um terceiro número, e o que torna este método o mais seguro é a complexidade em se obter os números primos utilizados, conhecido como fatoração, visto que são gerados produtos muito grandes e o tempo para realizar essa fatoração extrapola o limite essencial para desmantelar o algoritmo.

Enquanto as chaves simétricas aumentam à medida que a velocidade de processamento cresce, o mesmo não ocorre com chaves assimétricas, ao menos que sejam utilizados algoritmos de curvas elípticas. Esses foram desenvolvidos para aplicações em que se diminui os requisitos de velocidade de processamento e memória.

2.2.5 Interface Com o Usuário

Nos sistemas de segurança comentados, bem como nas aplicações de Internet das Coisas, existe uma interface de comunicação com o usuário. É por meio desta interface que o usuário se comunica, manipula e visualiza as informações do sistema. Uma interface de comunicação com o usuário deve ser de fácil

entendimento, apresentar informações de forma clara, métodos simples de operação e deve responder ao usuário o que está acontecendo, se as operações estão sendo realizadas ou apresentando falhas.

A interface pode ser desde modelos mais simples, como um teclado matricial para inserir dados numéricos, uma tela operada com botões, uma tela do sistema sensível ao toque ou até mesmo por meio de um *smartphone*. A utilização de aplicativos para *smartphones* como interface de sistemas vêm se tornando cada vez mais usual. Com o avanço dos protocolos de comunicação (*Wi-Fi*, *Bluetooth*), muitos sistemas já possuem aplicativos com esta função. Como exemplo de sistemas que utilizam esta interface por meio do *smartphone*, podemos citar o scanner do sistema de injeção eletrônica de um automóvel. Através de um adaptador conectado no módulo do sistema, seu *smartphone* é pareado com o adaptador e se torna a interface com o usuário.

Os aplicativos para celular, normalmente são programados em linguagens como *Python*, *Swift* e *JavaScript*, mas como são linguagens mais complexas, exigem que sejam utilizadas por pessoas com experiência nas mesmas. Para facilitar o desenvolvimento de aplicativos, podemos utilizar softwares online como o MIT *App Inventor* e o *Thunkable*.

O MIT *App Inventor* é uma plataforma online e gratuita, disponibilizada no site https://appinventor.mit.edu/, baseada em blocos que facilita a construção de aplicativos para Android e iOS, mesmo para pessoas que não tenham conhecimento na área. Foi desenvolvida pelo Instituto de Tecnologia de Massachusetts (MIT) e foi lançada em 2010. O *Thunkable* também é uma plataforma online e gratuita, disponibilizada no site https://thunkable.com/, baseada em blocos para a construção de aplicativos muito parecido com o MIT *App Inventor*. Os aplicativos criados nessa plataforma também podem ser utilizados nos sistemas Android e iOS (*iPhone Operating System*).

Ambos os softwares apresentam uma forma de programação de aplicativos simplificada, deixando mais acessível o desenvolvimento de aplicativos, mesmo para pessoas sem conhecimento profundo na área. A escolha entre os softwares apresentados se dará pelas preferências de cada usuário. O único requisito para utilizar as plataformas é uma conta do *Google* para login e armazenamento dos projetos criados.

2.3 PLATAFORMAS DE PROTIPAGEM RÁPIDA

As plataformas de prototipagem são dispositivos eletrônicos largamente utilizados para o desenvolvimento de projetos sem a necessidade de um alto investimento, seja eles com ou sem fins lucrativos, podendo conter as mais diversas aplicações, desde um simples controle de iluminação residencial até o gerenciamento de um setor de uma planta industrial, composta por diversos sensores, por exemplo.

Essas plataformas possuem diferentes especificações técnicas entre si, que são de extrema importância serem avaliadas previamente à aquisição, visto que é necessário verificar quais são os requisitos que certo projeto exige, visando a correta operação. Além disso, são compostas por certas funcionalidades em comum, mas as diferenças entre elas que atraem a atenção dos projetistas para um melhor custo-benefício, como por exemplo a presença de módulos que permitem a conectividade entre dois ou mais dispositivo, estabelecimento de link com sistemas em nuvem, entre outras.

A seguir, são apresentadas as plataformas mais conhecidas no mercado e posteriormente, um estudo comparativo dos microcontroladores que as compõem.

2.3.1 Arduino

A plataforma Arduino surgiu em 2005, na Itália, por meio de um projeto elaborado pelo professor Massimo Banzi e o engenheiro eletrônico David Cuartielles, que visava o ensino de programação e eletrônica. Como não havia na época uma placa de baixo custo que permitisse o uso para fins didáticos, este projeto foi o precursor dessa plataforma muito utilizada mundo a fora.

Para a elaboração dos mais variados projetos é necessário a utilização de um ambiente de desenvolvimento open-source criado pela própria fabricante, que tem como linguagem de programação baseada em C e C++. Os conceitos básicos de programação neste ambiente são os mesmos utilizados pelas variadas linguagens existentes e com a vasta disponibilidade de bibliotecas é possível desenvolver projetos para inúmeras aplicações.

A família de plataformas Arduino é extensa, contabilizando em torno de 20 modelos diferentes, sendo cada um com um aspecto físico e funcionalidades distintas. Essas funcionalidades estão sustentadas na quantidade disponível de portas digitais e analógicas, portas para controle PWM (*Pulse Width Modulation*), portas para comunicação serial além da USB (*Universal Serial Bus*), módulos de conectividade sem fio, entre outras. Dentre esses modelos, o Arduino Uno e Arduino Mega são os que mais se destacam.

O Arduino Uno é a plataforma mais vendida pela Arduino, e é baseada no microcontrolador ATmega328P. Este microcontrolador é da série de 8 bits fabricada pela Atmel, trabalha a 20 MHz de velocidade de clock com apenas 1 núcleo, 32 kB de memória *Flash*, 2 kB de memória SRAM, 1 kB de memória EEPROM e 23 entradas e saídas digitais programáveis. A plataforma faz parte de uma imensidão de projetos voltados para a robótica simples e automatização de processos, por meio do estabelecimento de uma interface com sensores, atuadores e *shields*, que são placas de circuito externas que podem ser facilmente acopladas às plataformas microcontroladas, expandindo as funcionalidades de fábrica, como o acoplamento de displays LCD, relés, módulos de comunicação, entre outras.

O Arduino Mega é um outro exemplar da Arduino muito difundido. É baseada no microcontrolador ATmega2560, com arquitetura de 8 bits, trabalha a 16 MHz de velocidade de clock com apenas 1 núcleo, possui 256 kB de memória *Flash*, 8 kB de memória SRAM, 4 kB de memória EEPROM e 86 entradas e saídas digitais programáveis. A plataforma é destinada para aplicações mais complexas, em virtude de que o microcontrolador que o comanda é superior se comparado ao do Arduino Uno, atendendo os pré-requisitos de projetos mais robustos. Assim como previamente exposto, com esta plataforma também é possível a conexão com outras placas externas.

2.3.2 Raspberry Pi

Esta plataforma de prototipagem rápida foi desenvolvida no Reino Unido pela Fundação Raspberry Pi, que tem por objetivo alimentar os jovens em escolas e universidades a se interessarem pela computação, por meio de um preço justo. O primeiro Raspberry Pi é baseado em um microcontrolador *Broadcom* BCM2835, com

arquitetura de 32 bits, possui 700 MHz de velocidade de clock com apenas um núcleo, 512 MB de memória RAM, 26 entradas e saídas digitais e não possui uma memória não-volátil integrada.

Diferentemente das plataformas Arduino, os modelos Raspberry, que atualmente conta com 8 exemplares, são considerados microprocessadores, uma vez que possui um poder de processamento e memória tão grande quanto de um computador tradicional. A intenção do fabricante é justamente permitir que o usuário seja capaz de executar tarefas e ter acesso aos componentes principais que um computador possui, porém, utilizando de um dispositivo pequeno e por um preço acessível.

Em termos de linguagem de programação, os dispositivos Raspberry são na maioria das vezes programados utilizando *Scratch* e *Python*, além da tradicional linguagem C, que é utilizada em muitas plataformas de prototipagem.

Em termos de aplicação, essas plataformas são utilizadas para diversas finalidades, como controlar um processo automatizado em indústrias, projetos no ramo da IoT, robótica como hobby ou aplicada, navegação na internet, criação de conteúdo multimídia, ente outras.

2.3.3 ESP

O mercado de plataformas de prototipagem rápida é muito vasto, com inúmeros fabricantes, os quais desenvolvem dispositivos com características chaves, que, para muitos projetos, são mais viáveis do que outros. E é com essas singularidades que tornou o ESP, plataforma desenvolvida pela fabricante chinesa Espressif, um dos dispositivos microcontrolados mais utilizados na atualidade.

Os modelos ESP mais conhecidos são o ESP8266 e o ESP32, sucessor do anterior. O ESP8266 possui 16 entradas e saídas digitais, processador Tensilica L106, arquitetura de 32 bits, trabalha a 80MHz de velocidade de clock com apenas 1 núcleo e possui 4 MB de memória *Flash*. Já o ESP32, possui 34 entradas e saídas digitais, processador Xtensa LX6, arquitetura de 32 bits, trabalha a 160 MHz de velocidade de clock com 2 núcleos e possui 4 MB de memória *Flash*. Apesar de pertencer a outro fabricante, os modelos ESP são muito similares aos dispositivos da Arduino, tanto em design quanto em linguagem de programação, uma vez que o

ambiente de desenvolvimento da fabricante italiana é compatível com plataformas ESP na maioria dos casos. Para aplicações específicas, é necessário desenvolver com a linguagem LUA, através do software *Eclipse*, por exemplo.

Uma característica notável destas plataformas, é a conectividade que estas proporcionam, como *Wi-Fi* e *Bluetooth*, amplamente utilizadas em projetos que envolvem os conceitos de IoT. Em relação a estes recursos, no ESP8266 apenas o módulo *Wi-Fi* é integrado, sendo que é possível realizar a conexão *Bluetooth*, porém precisa-se de um módulo externo. Já no ESP32 ambos os recursos vêm de fábrica, o que o torna diferenciado.

2.3.4 Análise Comparativa

Para um projeto eletrônico que faz uso de plataformas de prototipagem rápida, a análise das especificações técnicas da placa em si e do microcontrolador que a comandará é de extrema importância, visando o atendimento dos pré-requisitos. Nos documentos técnicos dos microcontroladores, há uma infinidade de parâmetros, entretanto alguns são cruciais para uma escolha correta, como a velocidade de processamento, capacidade de memória disponível.

A tabela 3 expõe de maneira clara e objetiva alguns parâmetros importantes para se comparar os microcontroladores associados às plataformas de prototipagem rápida Arduino UNO, Arduino MEGA, ESP8266 e ESP32. A exclusão do Raspberry da mesma deve-se ao fato desta plataforma ser considerada já um microprocessador, isto é, faz parte de uma outra hierarquia, que para este projeto foge do escopo para comparação.

Tabela 3 – Análise comparativa de plataformas de prototipagem rápida

	Arduino UNO	**Arduino MEGA**	**ESP8266**	**ESP32**
Barramento	8 bits	8bits	32 bits	32 bits
Núcleos	1	1	1	2
Clock (MHz)	20	16	80	160
SRAM	2kB	8kB	-	520kB
FLASH	32kB	256kB	4MB	4MB
EEPROM	1kB	4kB	-	-
GPIO (Microcontrolador)	23	86	-	-
GPIO (Plataforma)	23	54	16	34
Wi-Fi	-	-	Padrão 802.11 (HT20)	Padrão 802.11 (HT40)
Bluetooth	-	-	-	*Bluetooth* 4.2 *Low Energy*
				

Fonte: *Datasheets* Atmel, Microchip e Espressif Systems

As plataformas ESP levam vantagem em relação a quantidade de memória, sendo a ESP32 também a que tem maior velocidade de processamento. Ambas apresentam módulos de comunicação *Wi-Fi,* tendo ainda o módulo de comunicação Bluetooth na ESP32. Quanto ao número de portas GPIO, a plataforma Arduino MEGA leva vantagem por ter 54 portas, seguida da ESP32 com 34 portas, Arduino Uno com 23 portas e por último a plataforma ESP8266 com 16 portas.

3 MATERIAIS E MÉTODOS

Como apresentado na seção anterior, os sistemas de segurança são amplamente utilizados para garantir a proteção de ambientes com acesso restrito, como residências, prédios públicos, etc. A partir das informações sobre sistemas de segurança apresentadas até aqui, elaborou-se um projeto para construção de um protótipo de um sistema de controle de acesso, no qual o acesso pode ser realizado por múltiplos métodos e mediante ele, uma base de dados em nuvem registra os dados. Por meio deste sistema torna-se capaz garantir mais segurança e facilidade ao acesso à ambientes, dispensando a utilização de chaves convencionais para abertura. Na subseção seguinte são apresentadas as etapas para a implementação do projeto.

3.1 METODOLOGIA

Para a construção do protótipo é necessário seguir algumas etapas de elaboração do projeto, como a definição dos requisitos do sistema, as ferramentas para auxiliar o desenvolvimento e dos materiais que serão utilizados. As subseções seguintes, apresentam os passos para a elaboração do projeto, bem como a definição dos materiais utilizados e o desenvolvimento do projeto.

3.1.1 Definição dos Requisitos de Projeto

O ponto de partida para a construção da fechadura é a definição dos requisitos de projeto. Nesta etapa, devemos definir onde será instalado o sistema, qual é o nível de segurança requisitado, quais métodos de acesso serão utilizados, os protocolos de comunicação que estão disponíveis para uso no local, as fontes de energia disponíveis e se existem fatores de risco para a utilização de componentes eletrônicos, como a interferência eletromagnética.

Como forma de utilizar a metodologia apresentada, optou-se por construir um protótipo da fechadura com controle de acesso, montado em uma plataforma que simula uma porta real, controlada pelo sistema construído. Desta forma, os requisitos de projeto para a fechadura não serão tão impactantes no resultado final,

visto que modelos de fechadura, quantidade de sensores e protocolos de comunicação poderiam ser adaptados para o mesmo.

O local de instalação da fechadura é o que tem mais influência nos requisitos de projeto. Se for utilizada em um ambiente externo, como o portão de uma residência, são necessários cuidados adicionais com o dispositivo de travamento utilizado. Como é um local que exige maior segurança, devem ser empregados equipamentos mais robustos, que tenham maior resistência mecânica e que não fiquem expostos. Outro ponto de atenção é quanto ao estado do dispositivo sem alimentação. Um dispositivo para ambientes externos jamais poderia ser aberto em casos de falta de energia.

Alguns modelos de dispositivo de travamento, como o utilizado neste protótipo, possuem o sistema convencional de chaves para casos em que não se possa abrir eletronicamente. Se a fechadura for utilizada em ambientes internos, como nos laboratórios de uma universidade, um sistema com dispositivo que abre em falta de energia pode ser utilizado, garantindo ainda a segurança dos alunos e professores, que não ficarão aprisionados em seu interior.

O nível de segurança na autenticação de acesso é quem determina os métodos de acesso que podem ser utilizados. O uso de *tags* RFID é um método rápido para abertura, porém, não é totalmente seguro. Um cartão RFID pode ser utilizado por outras pessoas, que não a que realmente possui permissão de acesso. Mesmo assim, os dados de acesso serão registrados, e o proprietário do cartão pode ser responsabilizado. Acesso com senhas numéricas seguem o mesmo princípio do cartão RFID.

A utilização do smartphone para acesso pode reduzir essa fraude no sistema, porém, só sistemas como a biometria e o reconhecimento da retina são capazes de bloquear totalmente o acesso de pessoas não autorizadas. Como a tecnologia tem seu custo, o uso de recursos como a biometria e o reconhecimento da íris acabam aumentando o valor do projeto, por isso o nível de segurança exigido deve ser criteriosamente avaliado, para evitar gastos com segurança que não tragam tantos benefícios.

Os protocolos de comunicação precisam ser definidos para a escolha dos dispositivos utilizados. Se o local possui rede sem fio, a fechadura pode se conectar para enviar os dados de acesso e consultar informações de login. Na falta de rede

sem fio, a comunicação com um servidor poderia ser realizada via rede cabeada, interferindo na escolha dos dispositivos utilizados. A utilização de *smartphones* para a abertura exige um sistema com tecnologia *Bluetooth* implementado para a comunicação.

As fontes de energia para alimentação também precisam ser analisadas, não só em relação aos níveis de tensão, mas também quanto a estabilidade do fornecimento. Se o local apresenta muitos registros de falta de energia, um sistema secundário de alimentação pode ser necessário. Muitos sistemas de alarme instalados para diminuir a incidência de roubos, contam com o armazenamento de energia por meio de baterias. Desta forma, em casos de falta de energia, seja ela por falha na rede ou por desligamento proposital, o sistema seria capaz de manter a operação da fechadura.

Com os requisitos de projeto levantados, é possível iniciar a escolha de materiais que serão utilizados e o restante do desenvolvimento do sistema com maior segurança e garantia de que não ocorrerão falhas no sistema construído.

3.1.2 Fechos, Travas e Fechaduras Elétricas

Apesar do controle ser realizado por meio eletrônico, o microcontrolador precisa se comunicar com um dispositivo elétrico responsável pela abertura e travamento da porta. Este dispositivo precisa estar de acordo com os requisitos de projeto, para garantir o funcionamento do sistema sem falhas.

Em um sistema sem fonte alternativa de alimentação, como baterias, um dispositivo com seu estado normalmente aberto pode não ser recomendado. Em portas externas e portões residenciais como exemplo, no caso de falta de energia, a segurança fica comprometida. Nesta situação, as fechaduras elétricas normalmente fechadas são utilizadas. São dispositivos com maior qualidade de construção e resistência, pois operam em condições mais severas de ambientes externos (chuva, ventos, possíveis tentativas de violação). Seu funcionamento ocorre com um mecanismo operado por um eletroímã ou chave convencional, que fica normalmente travado. Quando recebe um pulso de energia no eletroímã, o mecanismo é destravado. Ao fechar a mesma, o mecanismo é ativado, sem a necessidade de chave ou energia.

Já em ambientes internos, utilizando o sistema em portas de salas e ambientes menores, pode ser recomendada a utilização de dispositivos normalmente abertos mesmo sem um sistema alternativo de alimentação. Em caso de falta de energia, esse modelo de dispositivo garante a segurança dos usuários, evitando deixar pessoas trancadas em salas, o que pode ser fatal num possível incêndio do local. Normalmente as travas e fechos apresentam esta característica, funcionando com a atuação de um eletroímã. Quando energizadas, mantém o sistema travado, quando a energia é interrompida, deixa de atuar instantaneamente.

Para um sistema com fonte alternativa de alimentação, qualquer tipo de dispositivo pode ser utilizado. Mesmo com a falta de energia, o sistema tem capacidade de continuar operando normalmente.

3.1.3 Software para Programação de Microcontroladores

Com os requisitos de projetos definidos, a escolha do microcontrolador pode ser realizada, e com ele, qual software será utilizado para a programação do mesmo. Para a programação de microcontroladores, geralmente existem softwares específicos para sua utilização, como é o caso da IDE (*Integrated Development Environment*) do Arduino.

A IDE do Arduino é um ambiente de desenvolvimento de códigos para aplicação no microcontrolador. Utiliza a linguagem C/C++ para programação e possui todas as ferramentas necessárias para desenvolver aplicações para Arduino. Apesar de ser desenvolvido para esta plataforma, pode ser utilizada para a programação de outros microcontroladores, como o ESP32. Originalmente este microcontrolador é programado em linguagem Lua, mas com a adição de algumas bibliotecas, pode ser programado de forma mais simples na IDE do Arduino.

Uma outra IDE que é amplamente utilizada para a programação de microcontroladores é a IDE *PlatformIO*. Este ambiente foi desenvolvido para a programação de diversas plataformas e também pode ser utilizada para programar o ESP32 em linguagem C/C++.

Ambas as plataformas possuem o necessário para a programação dos microcontroladores citados. A escolha de apenas uma delas para a programação

não trará prejuízos para quem não utilizar uma ou outra, pois as diferenças ficam mais simplificadas em preferências de cada programador.

3.1.4 Desenvolvimento de Interface com o Usuário

Para controlar o sistema, será necessário desenvolver uma interface de comunicação com o usuário. Dependendo dos métodos de acesso definidos no projeto, poderão ser aplicadas desde interfaces simples, como um teclado matricial, até a utilização de *smartphones*. Com a utilização de *smartphones* para acesso da fechadura, torna-se necessário a construção de um aplicativo como interface de comunicação para gerenciar este processo, juntamente com o microcontrolador da fechadura. O aplicativo será responsável por apresentar um layout de fácil compreensão para o usuário, por realizar a autenticação do usuário e para enviar comandos para a abertura da fechadura.

3.1.5 Ferramentas para Armazenamento na Nuvem

Com a necessidade de armazenamento das informações de autenticação de usuários e os registros de acessos, torna-se necessária a implementação de uma base de dados para o sistema. Com a instalação da fechadura em locais onde tenha conexão com a internet, é possível armazenar esses dados na nuvem. Desta forma, esses dados ficam acessíveis a qualquer momento, podendo ser consultados e alterados a qualquer momento.

O *Google* disponibiliza de forma gratuita para os seus usuários o *Drive*, que é uma plataforma online para armazenamento de arquivos. Além deste recurso, conta com as Planilhas e as *Fusion Tables*, que são amplamente utilizadas para o compartilhamento de informações. Ambos podem ser utilizados em paralelo com o protótipo, para o armazenamento de informações, e ambos os recursos podem ser armazenados no Drive, para acesso tanto da fechadura quanto dos gerenciadores do sistema.

3.1.6 Softwares para Simulação de Circuitos Eletrônicos

A fim de garantir o correto funcionamento de potência dos circuitos utilizados na fechadura, alguns softwares de simulação podem ser utilizados. Como exemplo, pode-se citar o OrCAD e o Proteus.

O software OrCAD foi desenvolvido para a simulação de circuitos eletrônicos, confecção de placas de circuito impresso, etc. O programa é capaz de prever inúmeros fenômenos físicos que podem prejudicar o funcionamento dos componentes, o que o torna uma ferramenta robusta na simulação de circuitos eletrônicos.

O Software Proteus também foi desenvolvido para simular circuitos eletrônicos, porém, não tem a mesma capacidade do OrCAD para prever falhas em circuitos eletrônicos. Uma vantagem do Proteus, é que, por ter uma interface mais simples, acaba sendo utilizado por um maior número de pessoas. Isto aumenta o interesse pelo software e faz com que tenha mais componentes modelados em suas bibliotecas, como é o caso de muitos microcontroladores. Desta forma, um projeto mais complexo, com a utilização de um microcontrolador, pode ter todo o seu funcionamento simulado.

3.2 MATERIAIS

A partir dos requisitos de projeto definidos na seção 3.1.1, a escolha dos materiais pôde ser realizada, respeitando os requisitos de projeto, para evitar que o sistema apresente falhas em seu funcionamento. As subseções seguintes, apresentam a definição dos materiais utilizados para a construção do protótipo.

3.2.1 ESP32

Para o controle da fechadura, foi utilizado o ESP32 NodeMCU-32S versão 1.1. O modelo foi escolhido por possuir 160 MHz de velocidade de processamento, sendo o mais rápido entre os comparados, e principalmente por possuir os módulos para comunicação *Wi-Fi* e *Bluetooth* integrados.

3.2.2 Periféricos

Além do microcontrolador, alguns periféricos foram adicionados a fechadura para poder operar com todas as formas de acesso escolhidas para o protótipo, e sinalizar corretamente o funcionamento da fechadura.

Para inserir a senha numérica de administrador cadastrada, ou novas senhas adicionadas na programação do microcontrolador, foi utilizado um teclado matricial 4x3 de membrana. É um periférico de baixo custo, possui um conector com 7 pinos, tempo de contato de 5 ms, corrente máxima de 100 mA e adesivo para fixação.

Para a leitura de *tag* RFID, foi utilizado o leitor RFID RC-522 para chips Mifare, que operam na faixa de 13,56 MHz. A comunicação com o microcontrolador é realizada com o protocolo SPI de comunicação serial, onde o leitor RFID opera como escravo e o microcontrolador como mestre. Para detectar a posição da porta (aberta/fechada), uma chave fim de curso simples foi utilizada no protótipo, enviando um sinal para uma porta Input do microcontrolador. O *buzzer* utilizado para a sinalização sonora, trabalha na tensão de 3 V, emite ondas sonoras na frequência de 2400 Hz e consome aproximadamente 30 mA, o que impossibilita alimentá-lo diretamente pelas portas digitais do ESP32. Para resolver o problema, foi utilizado um transistor BC549, alimentado na base pela porta digital, o coletor com a tensão de 3,3 V e o emissor ligado ao *buzzer*, como será apresentado no diagrama de ligações da subseção 3.3.1. Os LEDs (*Light Emitting Diode*) de sinalização luminosa, são alimentados diretamente pelas portas digitais. São ligados em série com uma carga resistiva, limitando a corrente de saída das portas.

A fechadura elétrica utilizada é uma FC90 A2225/12 V fabricada pela IPEC Eletrônica e apresenta boa relação custo benefício. Como a intenção é simular uma porta simples, de um possível ambiente externo, e sem sistema alternativo de energia, o modelo se adequa ao protótipo. Alimentada com tensão contínua de 12 V, consome uma corrente de 450 mA para abertura. Um único pulso de corrente é necessário para a abertura da fechadura, mas devido a limitação de corrente das portas do ESP32, precisamos utilizar um transistor TIP122 para aumentar a corrente fornecida. O transistor é alimentado na base pela porta digital, o coletor com a tensão de 12 V e o emissor ligado aos terminais da fechadura, como será

apresentado no diagrama de ligações da subseção 3.3.1. Este modelo de fechadura ainda apresenta a possibilidade de abertura por chave convencional, pois em casos de falta de energia, ou eventuais falhas de projeto, não seria possível abrir a fechadura.

3.2.3 Fornecimento de Energia ao Sistema

O protótipo tem a necessidade de alimentação em dois níveis de tensão, sendo 12 V para a fechadura elétrica e 3,3 V para o microcontrolador e alguns periféricos. No projeto, uma fonte chaveada tipo colmeia, com tensão de 12 V e capacidade de 5 A foi utilizada para alimentar todo o sistema. Baseado no consumo da fechadura e do microcontrolador, uma fonte 12 V com capacidade de 1 A já é capaz de suprir a demanda do projeto, mas por questão de disponibilidade, utilizamos o modelo citado.

Para regular a tensão de 3,3 V e alimentar o microcontrolador, foi utilizado um módulo regulador de tensão Mini-360 *DC-DC Buck converter*. Baseado no Circuito integrado MP2307DN, é amplamente utilizado em dispositivos IoT. O módulo é capaz de converter uma tensão de entrada entre 4,75 V e 23 V, para um valor entre 1 V e 17 V, respeitando sempre o princípio de abaixar a tensão.

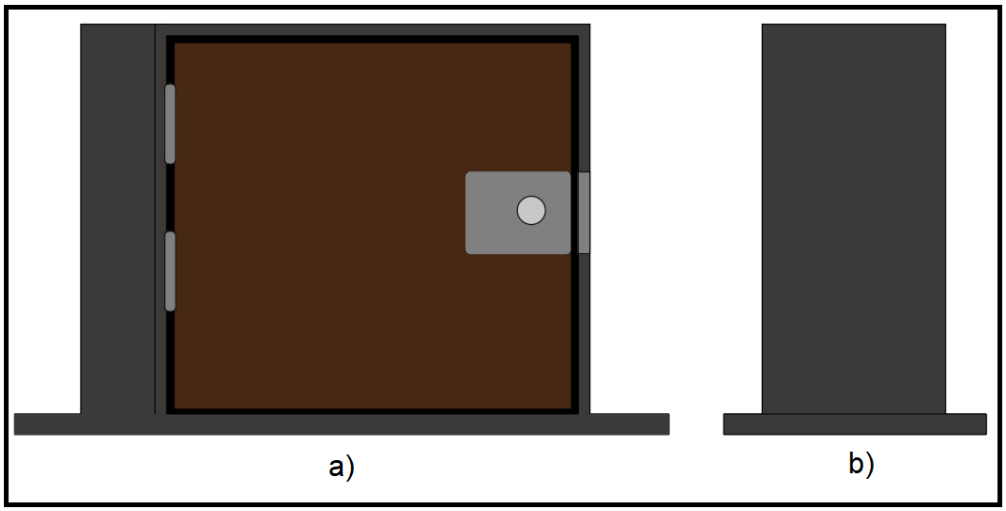
3.2.4 Armazenamento de Dados

Para o armazenamento de dados para a fechadura, optou-se pela utilização de uma base de dados na nuvem, com o uso de planilhas, formulários e *Fusion Tables*. A facilidade de utilização destes recursos na nuvem, a simplicidade de funcionamento com acesso por outros dispositivos e a disponibilização gratuita das ferramentas foram os fatores decisivos para a utilização desta forma de armazenamento.

3.2.5 Materiais para Confecção da Estrutura do Protótipo

O protótipo para a simulação do funcionamento da fechadura, foi construído de acordo com a figura 8, onde é apresentada a vista frontal (a) e a vista lateral (b) do protótipo ainda sem o sistema de controle. Consiste basicamente em simular o movimento de uma porta convencional.

Figura 8 - Ilustração do Protótipo, a) Vista frontal. b) Vista lateral.



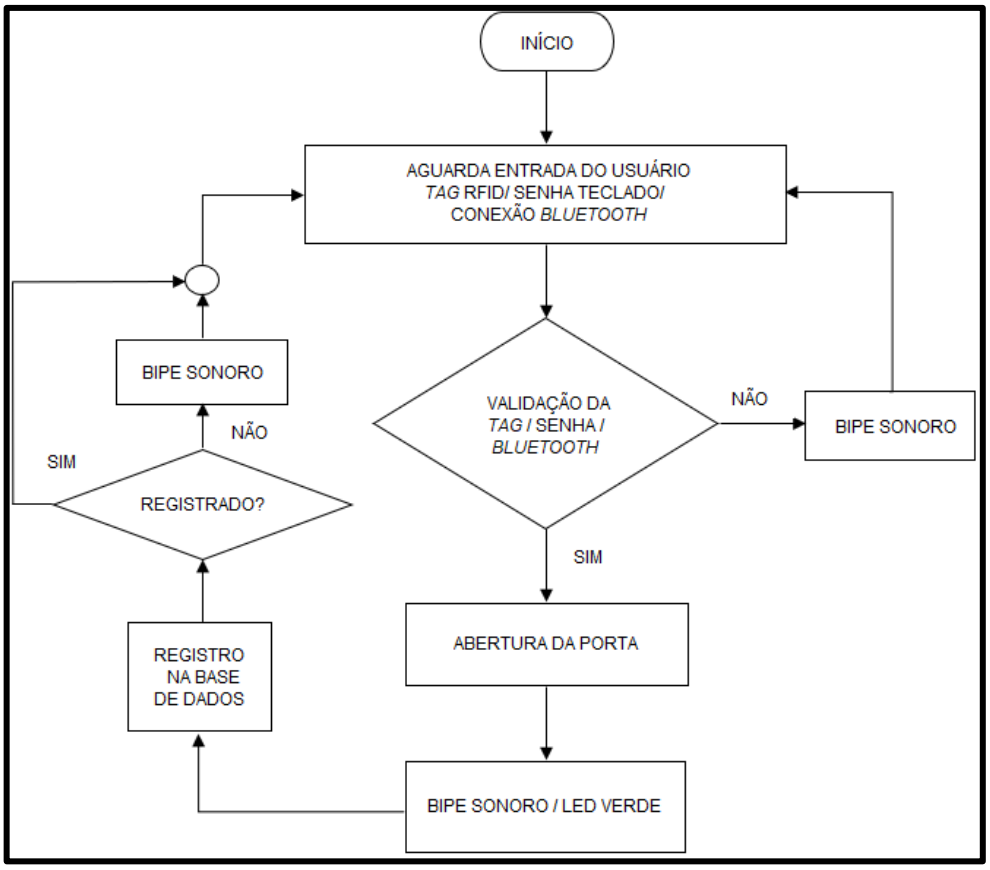
Fonte: Autoria própria

Na construção, foi empregado predominantemente o MDF, que não é o material com menor custo, mas foi utilizado devido a disponibilidade. Parafusos de 3x35mm foram utilizados nas junções, duas dobradiças de porta, para manter a fixação e movimento da porta e uma tinta spray para acabamento.

3.3 DESCRIÇÃO DO PROJETO

A partir das informações técnicas apresentadas até aqui, foi elaborado um projeto de construção de uma fechadura com controle de acesso. Seu funcionamento ocorre de acordo com o *firmware* do microcontrolador da mesma, representado no fluxograma da figura 9.

Figura 9 – Firmware do microcontrolador da fechadura



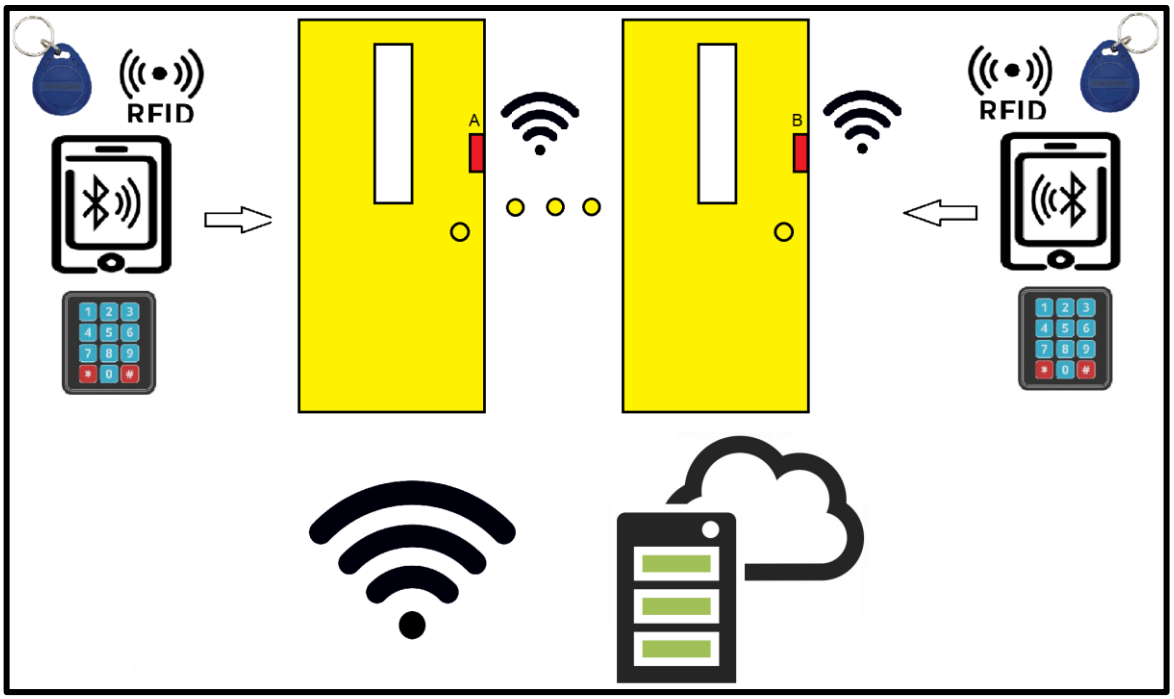
Fonte: Autoria própria

O sistema possui um microcontrolador, que é responsável por controlar os periféricos da fechadura e realizar as comunicações necessárias. A prioridade do sistema é conceder o acesso, e em segundo plano, registar o mesmo. Se por alguma falha de comunicação o registro não for realizado, a abertura da fechadura não será prejudicada. Ao iniciar, o sistema fica aguardando um método de acesso, que pode ser por uma senha numérica de administrador, uma *tag* RFID ou por uma conexão *Bluetooth*. Se o método de acesso não é validado, um bipe sonoro é emitido e a fechadura volta ao estado inicial. Se o método de acesso é validado, a fechadura é aberta, um bipe sonoro e um LED verde são acionados para sinalizar ao usuário. Após a abertura da fechadura, o microcontrolador estabelece uma comunicação via *Wi-Fi* para acessar o banco de dados e registrar o acesso. Se a comunicação falhar com algum motivo, um bipe sonoro é acionado para indicar a falha do procedimento.

Vale ressaltar que, como o sistema da fechadura controla apenas a abertura de uma porta, outras fechaduras poderiam ser implementadas no mesmo ambiente.

Cada uma opera de forma independente, liberando o acesso e registrando na base de dados. Um usuário com acesso pelo aplicativo, poderia escolher a porta que necessita acesso, ou possuir uma tag que tenha permissão para acessar todas as portas. A figura 10 representa o sistema com a possibilidade de múltiplos acessos.

Figura 10 – Controle de acessos com múltiplas fechaduras



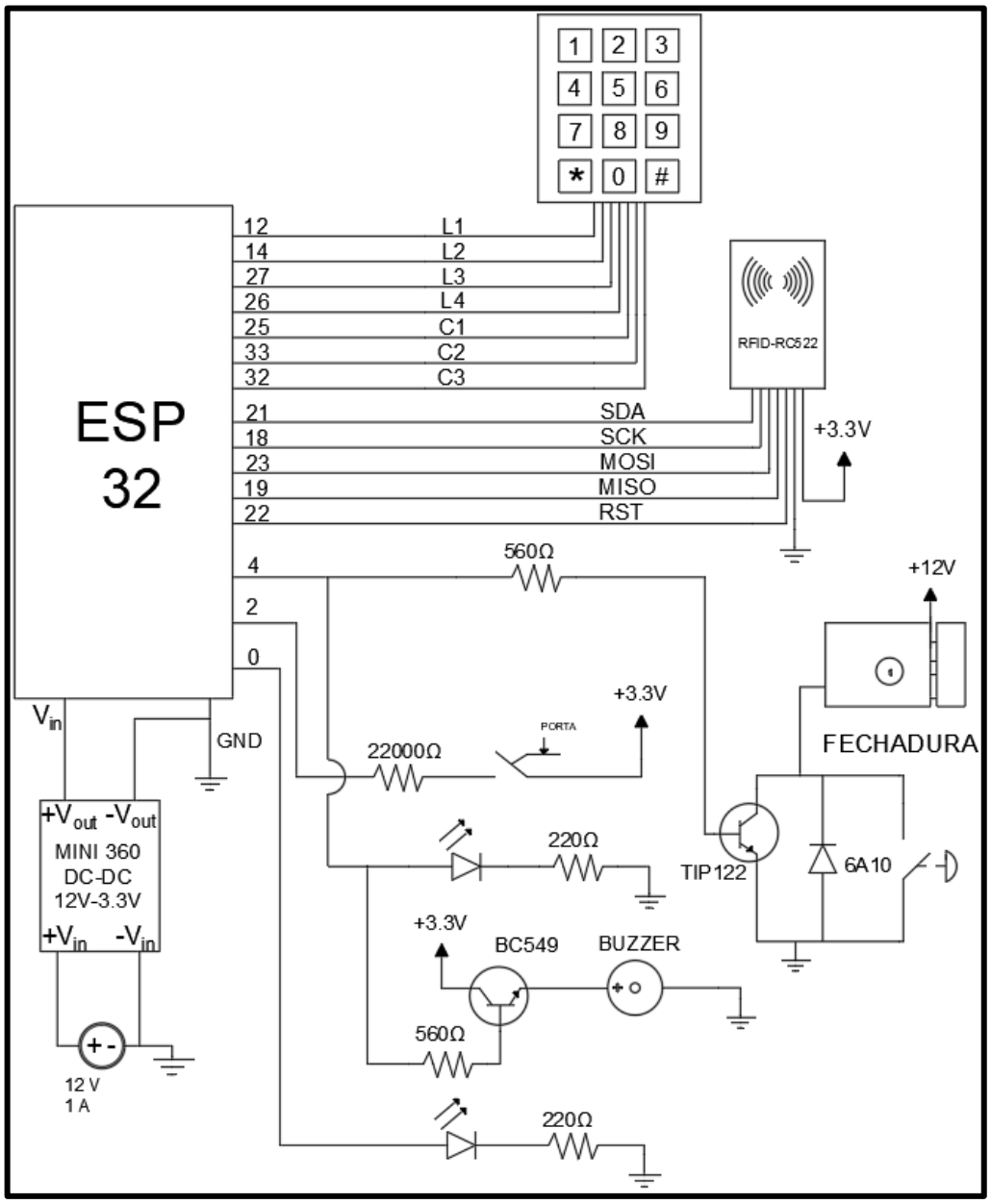
Fonte: Autoria própria

Ambas as fechaduras podem estabelecer de forma independente uma conexão com a base de dados, e os registros de acesso de cada fechadura serão armazenados separadamente. Os cartões de acesso e as senhas cadastradas podem ter acesso a ambas ou apenas a uma fechadura, e usuários cadastrados para o aplicativo, terão a possibilidade de acesso a todas as fechaduras do sistema.

3.3.1 Diagrama de Ligações

A figura 11 apresenta o diagrama de ligações entre os periféricos do sistema e a alimentação, incluídos os pinos do ESP32 utilizados na elaboração do código.

Figura 11 – Diagrama de ligações entre os componentes



Fonte: Autoria própria

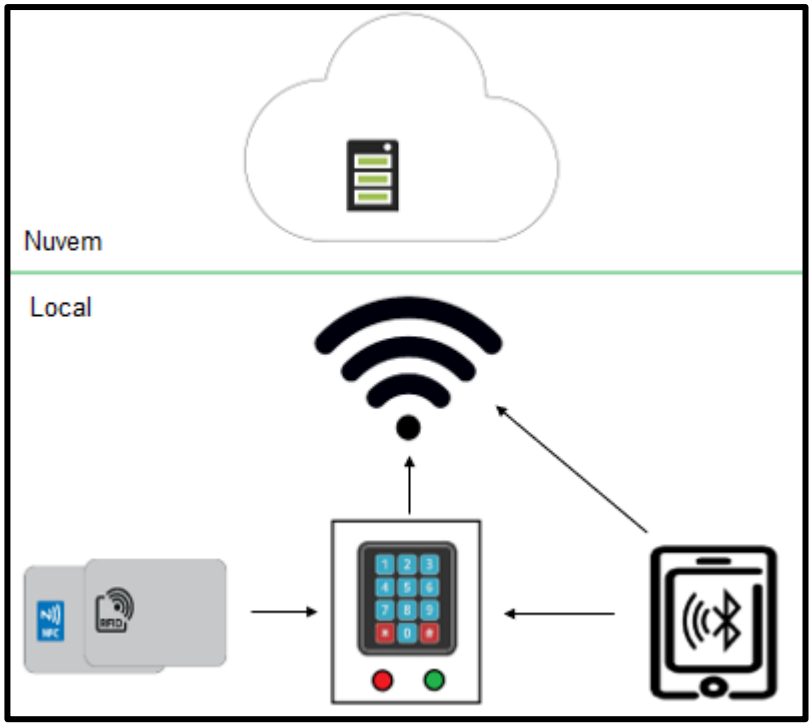
No esquemático, em especial as ligações entre o ESP32 (mestre) e o leitor das *tags* (escravo), contam com a nomenclatura utilizada para representar a comunicação serial SPI entre o microcontrolador e periféricos como segue:

- MISO: ligação que permite o escravo enviar dados ao mestre.
- MOSI: ligação que permite o mestre enviar dados ao escravo.
- SCK: estabelece o *clock* de sincronização para transmissão de dados.
- RST: *reset* de configurações.
- SDA: permite que o escravo se comunique com o mestre.

3.3.2 Segurança das Informações

As informações utilizadas pelos sistemas do protótipo, podem ser divididas em dois níveis de segurança, sendo o nível de segurança local e o nível de segurança em nuvem, como representado na figura 12.

Figura 12 – Níveis de segurança



Fonte: Autoria própria

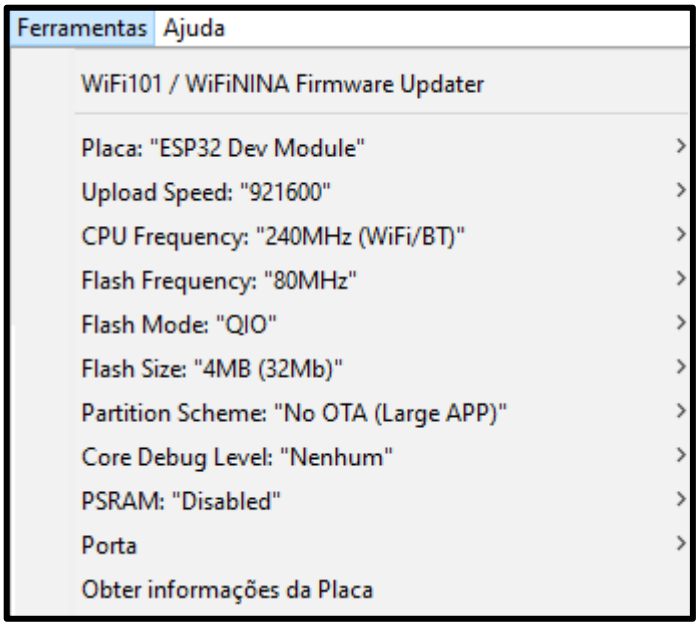
As comunicações locais, como a comunicação *Bluetooth* entre os dispositivos e a comunicação RFID ou NFC, podem ser protegidas pela distância de comunicação, limitada aos ambientes onde o sistema se encontra, e podem ser empregados métodos adicionais, como a criptografia de dados. A partir do momento que as informações são colocadas em nuvem, utilizando os recursos do Google, como as *Fusion Tables* e planilhas, a segurança fica por conta dos métodos de proteção utilizados por estes sistemas em nuvem.

Para os dados da comunicação *Bluetooth* entre o *smartphone* e a fechadura, foi implementada uma criptografia simétrica, além dos métodos de criptografia já utilizados pelo aplicativo e pelas bibliotecas de programação do microcontrolador. A chave simétrica utilizada será apresentada na seção 3.3.2 junto com o código-fonte.

3.3.3 Descrição do Código-Fonte

Para iniciar o desenvolvimento do código fonte da plataforma ESP32 foi necessário configurar alguns parâmetros, como a placa que foi utilizada, bem como a velocidade de upload do código, frequências de operação da CPU, da memória *Flash*, ilustrada na figura 13. Um ponto muito importante a ressaltar é de que, apesar do ambiente de desenvolvimento ser da Arduino, há a compatibilidade com a plataforma ESP, por meio da instalação de um módulo para ESP32.

Figura 13 - Configurações na IDE



Fonte: Autoria própria

Como na maioria dos projetos envolvendo plataformas de prototipagem para as mais variadas aplicações, é necessário fazer o uso de bibliotecas específicas, que permitem a utilização de diversos métodos destinados a facilitar o desenvolvimento. A figura 14 expõe as bibliotecas que foram utilizadas no projeto.

A biblioteca “Keypad.h” destinou-se à manipulação das variáveis que estão associadas ao teclado matricial, como mapeamento das linhas e colunas do mesmo e consequentemente atribuição à um objeto criado. Com a “SPI.h”, foi realizado a comunicação com dispositivos periféricos, como o leitor RFID no projeto, dado o ESP32 como mestre.

Por meio da biblioteca “MFRC522.h” gerenciou-se todo o controle de leitura e escrita das *tags* utilizadas como método de acesso do projeto. Com o conjunto de

bibliotecas com o prefixo “BLE” foi estabelecido todos parâmetros para a comunicação *Bluetooth*, entre o microcontrolador e o aplicativo desenvolvido, que será explanado adiante. Por fim, com o intuito de estabelecer uma conexão sem fio entre o sistema e um banco de dados em nuvem para os registros de acesso, foram utilizados pacotes para a comunicação *Wi-Fi.*

Figura 14 - Bibliotecas utilizadas no projeto

```
//----------------------Bibliotecas----------------------------------------------

#include <Keypad.h>             // Teclado Matricial
#include <SPI.h>                //biblioteca para comunicação do barramento SPI
#include <MFRC522.h>            //Leitor RFID
#include <BLEDevice.h>
#include <BLEServer.h>          //Bibliotecas para comunicação Bluetooth
#include <BLEUtils.h>
#include <BLE2902.h>
#include <WiFi.h>               //Bibliotecas para comunicação WiFi
#include <WiFiClientSecure.h>
                                ...
```

Fonte: Autoria própria

Estas bibliotecas contam já com métodos de criptografia implementados, em especial com uso da metodologia RSA, que baseia-se no uso de uma chave pública para encriptação e uma chave privada para decriptação, derivadas da multiplicação de dois números primos grandes.

Após a definição de parâmetros para um funcionamento adequado à aplicação, a primeira seção do código desenvolvido é a declaração das variáveis, como mostra a figura 15. Nela constam as variáveis globais e estas foram responsáveis por realizar processos de contagem de caracteres selecionados no teclado, de disposição dos caracteres do teclado em um *array,* de definição da senha que acessa o sistema, de identificação e numeração as *tags* que estão autorizadas, de armazenamento dos dados criptografados, de definição dos pinos utilizados do microcontrolador.

Figura 15 - Declaração de variáveis

```
//--------------------Declaração de Variáveis---------------------

int    sensorPorta = HIGH;
int    contagem = 0;
char   usuarioBluetooth[40];
char   senha[5], senhaPadrao[5]={"1234"};
String usuario, armazenaTag;
// Número das tags
String tags[4]={"D8 32 76 E9", "A2 97 A9 89",
                "B9 AD AB 56", "F0 C0 AC 47"};
// Identificação das tags
String usuarioTag[4]={"stevanTag", "pedrocrespoTag",
                      "luismanenteTag", "visitanteTag"};
const byte linhas = 4;
const byte colunas = 3;
byte  pinosLinhas[linhas] = {12,14,27,26};
byte  pinosColunas[colunas] = {25,33,32};

char Teclado [linhas][colunas] =
{
   {'1','2','3'},
   {'4','5','6'},
   {'7','8','9'},
   {'*','0','#'}
};
```

Fonte: Autoria própria

Os pinos utilizados da plataforma de prototipagem rápida para conectar ao teclado numérico não possuem numeração especial, bastando apenas ser conectado a uma porta IO disponível e programada. O vetor *string tags* e usuárioTag são responsáveis pelo armazenamento do número das *tags* e dos usuários correspondentes. Estes dados precisam ser inseridos na programação do microcontrolador e não podem ser alterados durante o funcionamento da fechadura. Para alterar as informações referentes as *tags* e usuários, o sistema precisa ser desligado e o microcontrolador conectado ao software de programação. Para armazenar 80 *tags*, é consumido aproximadamente 2 kB da memória para variáveis globais. Como a plataforma possui 327,680 kB de memória dinâmica, e as variáveis de todo o código consomem 65 kB, o armazenamento de *tags* não será afetado pela quantidade de memória da plataforma.

Posterior à definição das variáveis, foi realizado a inicialização de alguns parâmetros, como mostra a figura 16, pertinentes ao leitor das *tags*, ao mapeamento das linhas e colunas do teclado, criação de um cliente para a comunicação *Wi-Fi* e a

definição de parte do endereço associado ao formulário de armazenamento de registros do *Google* Planilhas, que será exposto no decorrer do trabalho.

Figura 16 - Inicialização geral

```
//----------------------Inicializações---------------------------------------------------------------------

MFRC522 mfrc522(21, 22); // MFRC522
Keypad teclado = Keypad(makeKeymap(Teclado),pinosLinhas, pinosColunas, linhas,colunas);
WiFiClientSecure client;//Cria um cliente seguro (para ter acesso ao HTTPS)
String textFix = "GET /forms/d/e/XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX/formResponse?ifq&entry.XXXXXXXXXX=";
...
```

Fonte: Autoria própria

Na inicialização geral, foram criados objetos (por exemplo teclado, client) que serão utilizados para efetuar o processo de leitura do teclado, estabelecimento do link com a base de dados. Em relação aos pinos 21 e 22 conectados ao leitor RFID, estes necessitam não ser arbitrários, uma vez que são responsáveis pela comunicação SPI.

Com a finalidade de tornar o desenvolvimento e a compreensão do código mais fácil, foi realizado a modularização do mesmo por meio da criação de diversas funções. A primeira função denominada registrarAcesso() pode ser ilustrada na figura 17 abaixo, bem como seu fluxograma, na figura 18:

Figura 17 - Função para registro de acessos

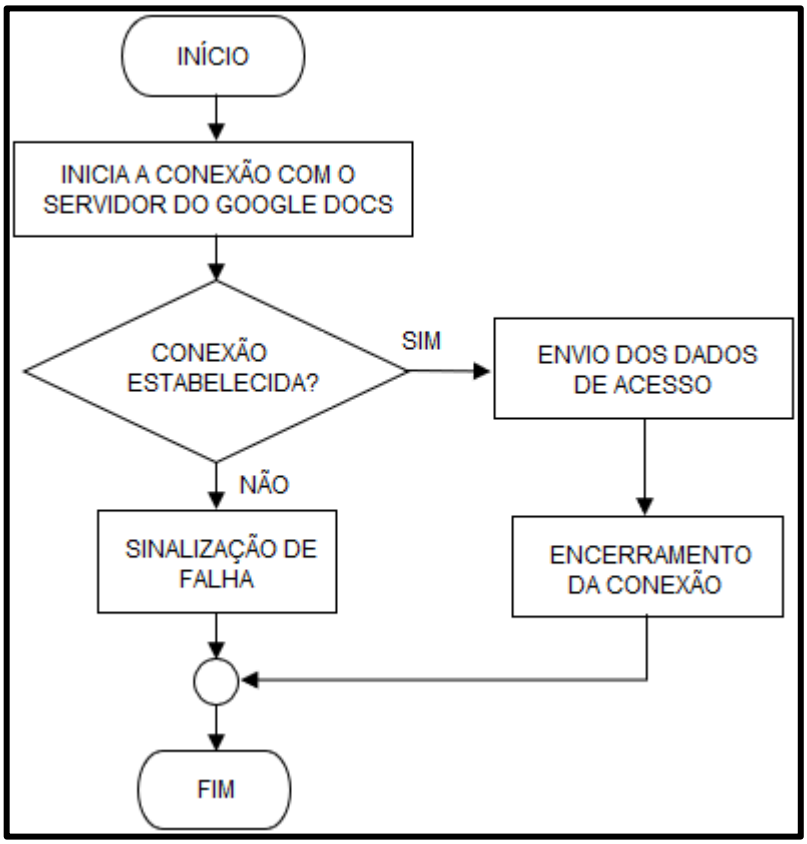
```
//----------------------Procedimentos-----------------------------------------------------------------------

void registrarAcesso (){  //Registra o acesso no banco de dados
  if (client.connect("docs.google.com", 443) == 1)//Tenta conexão ao servidor do Google docs (HTTPS)
  {
    String toSend = textFix;//Atribuimos a String auxiliar na nova String que sera enviada
    toSend += usuario;//Adiciona o usuário
    toSend += "&submit=Submit HTTP/1.1";//Completa o metodo GET para o formulario.
    client.println(toSend);//Envia o GET para o servidor
    client.println("Host: docs.google.com");
    client.println();
    client.stop();//Encerra a conexao com o servidor
  }
  else
    falhaAbertura();
}
...
```

Fonte: Autoria própria

Os métodos utilizados nesta função derivam das bibliotecas declaradas para a comunicação *Wi-Fi*, o que facilita o processo de codificação da estrutura para registro dos acessos na nuvem.

Figura 18 - Fluxograma da função registrarAcesso()



Fonte: Autoria própria

O objetivo desta função, quando requisitada, é de estabelecer o link com o servidor para registrar o acesso ao sistema na planilha criada na plataforma da *Google*. A *string* declarada contém o endereço completo do formulário em uso e caso não seja possível realizar o acesso, a função falhaAbertura() é chamada.

Para iniciar a compreensão sobre os métodos disponíveis para acesso, temos o uso do teclado matricial. Uma função denominada acessoTeclado() foi escrita (ver figura 19) e seu funcionamento é simples, como mostra o fluxograma na figura 20. Como explícito na declaração de variáveis, foi definido uma senha padrão para autenticação. Na função escrita, uma senha de 4 dígitos é armazenada em um vetor de acordo com o que o usuário digita e mediante isto, é feita uma comparação, pelo método strcmp. Caso a senha digitada coincida com a pré-definida, o sistema

reconhece o usuário como administrador e faz chamada da função abreFechadura(). Do contrário, a função falhaAbertura() é chamada, negando o acesso ao sistema.

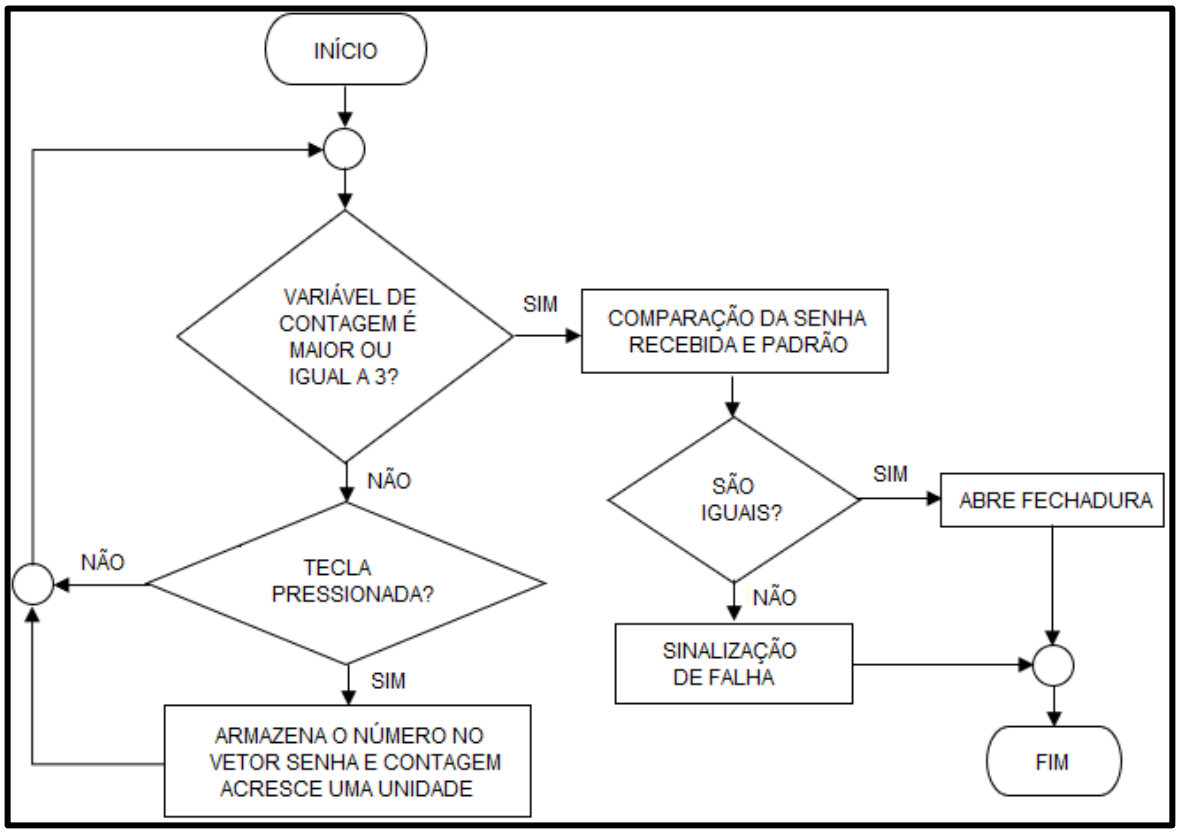
Figura 19 - Método para acesso com teclado matricial

```
void acessoTeclado()     //Realiza o acesso pelo teclado
{
  while(contagem<=3)
  {
    char teclaClicada = teclado.getKey();
    if(teclaClicada)
    {
      senha[contagem] = teclaClicada;
      contagem++;
    }
  }
  contagem = 0;
  senha[5] = '\0';
  if(strcmp(senha, senhaPadrao) == 0)
  {
    usuario = {"admin"};
    abreFechadura();
  }
  else{
    falhaAbertura();
  }
}
...
```

Fonte: Autoria própria

Esta função que controla o acesso ao teclado matricial executa a manipulação de *strings*, primeiramente em relação à inserção de um caracter nulo, representado por \0, ao fim do ao fim *array* “senha[]”, indicando o fim do mesmo, que contém a senha dada pelo usuário. Posteriormente, o método strcmp compara a senha pré-definida com a senha digitada e sinaliza, caso coincidam, será identificado o usuário como “admin” e a abertura da fechadura é efetuada. Caso as senhas não coincidam, há a sinalização da falha e fechadura mantém-se travada.

Figura 20 - Fluxograma da função acessoTeclado()



Fonte: Autoria própria

O outro método de acesso do projeto é por meio da leitura de *tags* cadastradas, por meio da tecnologia RFID. Uma função denominada leituraTag() foi escrita, conforme mostra a figura 21 e seu funcionamento, mostrado no fluxograma da figura 22, é baseado no seguinte: por meio de um método específico, consegue-se verificar a presença de uma *tag* nas proximidades do leitor e através da leitura da mesma, é armazenada a identificação da *tag*, formada por 4 blocos de 2 dígitos hexadecimais cada em uma *string*.

A identificação armazenada é comparada com a sequência previamente definida no sistema e caso coincidam, a autenticação é realizada, liberando o acesso por meio da chamada da função abreFechadura() e consequentemente, o registro é realizado, pela função registrarAcesso().

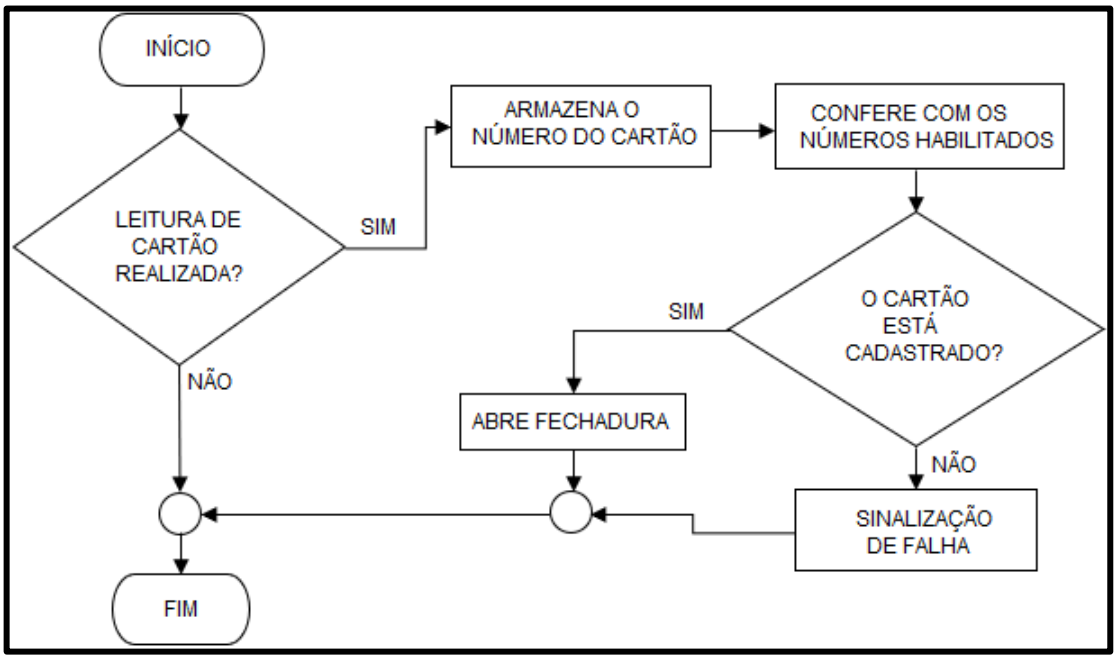
Figura 21 - Método de acesso pela leitura de *tag*

```
void leituraTag()         //Realiza o acesso pela TAG
{
  if ( ! mfrc522.PICC_IsNewCardPresent())   // Procura por cartao RFID
  {
    return;
  }
  if ( ! mfrc522.PICC_ReadCardSerial())     // Seleciona o cartao RFID
  {
    return;
  }
  String conteudo= "";
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     conteudo.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     conteudo.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  conteudo.toUpperCase();
  while(contagem<4)
  {
    if (conteudo.substring(1) == tags[contagem]) //UID 1 - Cartao
    {
      usuario = usuarioTag[contagem];
      abreFechadura();
      contagem = 4;
    }
    if(contagem == 3){
      falhaAbertura();
    }
    contagem++;
  }
  contagem = 0;
  conteudo = "";
}
```

Fonte: Autoria própria

No processo de leitura das *tags*, observa-se uma grande manipulação da variável “contagem”, que é responsável por conferir o endereçamento dos cartões cadastrados, permitindo ou não a abertura da fechadura. A string “conteudo” é responsável por armazenar a identificação do cartão utilizado. Este número de identificação é verificado um a um com os dados cadastrados na memória do microcontrolador. Após a verificação realizada pelo procedimento, a *string* “conteudo” é limpa para outras verificações.

Figura 22 - Fluxograma da função leituraTag()



Fonte: Autoria própria

Para realizar o controle da fechadura, a função abreFechadura() foi escrita e é responsável por duas tarefas: acusar a abertura por meio do acendimento de um LED verde e um bipe sonoro e enviar um sinal elétrico, que irá posteriormente ser amplificado por um transistor de potência para acionar o mecanismo da fechadura. Quando esta função é chamada, a registrarAcesso() também é disparada, armazenando os dados do acesso realizado.

Caso a abertura da fechadura ou registro do acesso não seja possível ser realizado, a função falhaAbertura() é chamada, e é responsável por gerar um bipe sonoro diferente do qual é feito quando a autenticação ocorre de maneira adequada. A figura 23 ilustra as funções explanadas.

Figura 23 - Métodos para abertura correta da fechadura ou falha

```
void abreFechadura()    //Abre a fechadura elétrica
{
  digitalWrite(0, HIGH);
  digitalWrite(4, LOW);
  digitalWrite(15, HIGH);
  delay(1000);
  digitalWrite(0, LOW);
  digitalWrite(4, HIGH);
  digitalWrite(15, LOW);
  registrarAcesso();
  delay(2000);
}

void falhaAbertura(){  //Indica falha na tentativa de abertura ou registro
  for (int i = 0; i <3; i++){
    digitalWrite(0, HIGH);
    delay(250);
    digitalWrite(0, LOW);
    delay(100);
  }
}
...
```

Fonte: Autoria própria

Para realizar a comunicação *Bluetooth*, são necessárias algumas declarações iniciais. As características da comunicação e as chamadas para conectar e desconectar de dispositivos são configuradas, como mostra a figura 24. O identificador único universal UUID (*Universal Unique Identifier*) definido nesta etapa, representa a identificação que o dispositivo *Bluetooth* e o aplicativo vão utilizar para trocar informações.

Os identificadores UUID são números de 128 bits, que são representados por 32 dígitos hexadecimais em cinco grupos. A sua representação possui um caracter dedicado que representa o modo como a UUID foi gerada. O caracter 13 na ordem, apresenta esta função, podendo ser entre 1 e 5. Para o projeto, foi utilizado um gerador de UUID de forma aleatória, no site Online UUID Generator. A geração de UUID de forma aleatória, carrega o caracter “4” na posição 13, indicando que foram gerados usando um número aleatório ou pseudoaleatório.

Figura 24 – Inicialização das configurações para comunicação *Bluetooth*

```
//--------------------Definições e abertura através do Bluetooth----------------------------

BLECharacteristic *pCharacteristic;
bool deviceConnected = false;

#include <iostream>
#include <string>
#define SERVICE_UUID           "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" // UART service UUID
#define CHARACTERISTIC_UUID_RX "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
#define CHARACTERISTIC_UUID_TX "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"

class MyServerCallbacks: public BLEServerCallbacks {
  void onConnect(BLEServer* pServer) {
    deviceConnected = true;
  };
  void onDisconnected(BLEServer* pServer) {
    deviceConnected = false;
  }
};
...
```

Fonte: Autoria própria

As diretivas do código apresentado definem a estrutura dos identificadores UUID utilizados e a classe é responsável por indicar se o dispositivo *Bluetooth* está conectado ao sistema.

A segurança das informações trocadas pela conexão *Bluetooth* e *Wi-Fi* com o microcontrolador é garantida com a utilização da criptografia de dados presente nas bibliotecas de cada função. Ambas utilizam o método de criptografia com chave pública RSA. Devido a importância da segurança das informações, e para tornar mais compreensível a questão de criptografia, foi utilizado um modelo simples para a comunicação. O modelo consiste em uma criptografia com chave simétrica, onde a mesma chave é utilizada para encriptação e decriptação de texto. A tabela 4 apresenta a chave utilizada para o modelo de criptografia.

O método de acesso por *Bluetooth* apresentado na figura 25, consistem em receber uma s*tring* do dispositivo conectado, descriptografar a mensagem, conferir se o dispositivo apresenta permissão para acessar a fechadura e se a porta está fechada, armazenar o conteúdo em um *string* para registro de acesso e por fim, liberar o acesso do usuário.

Figura 25 – Método de acesso pela comunicação *Bluetooth*

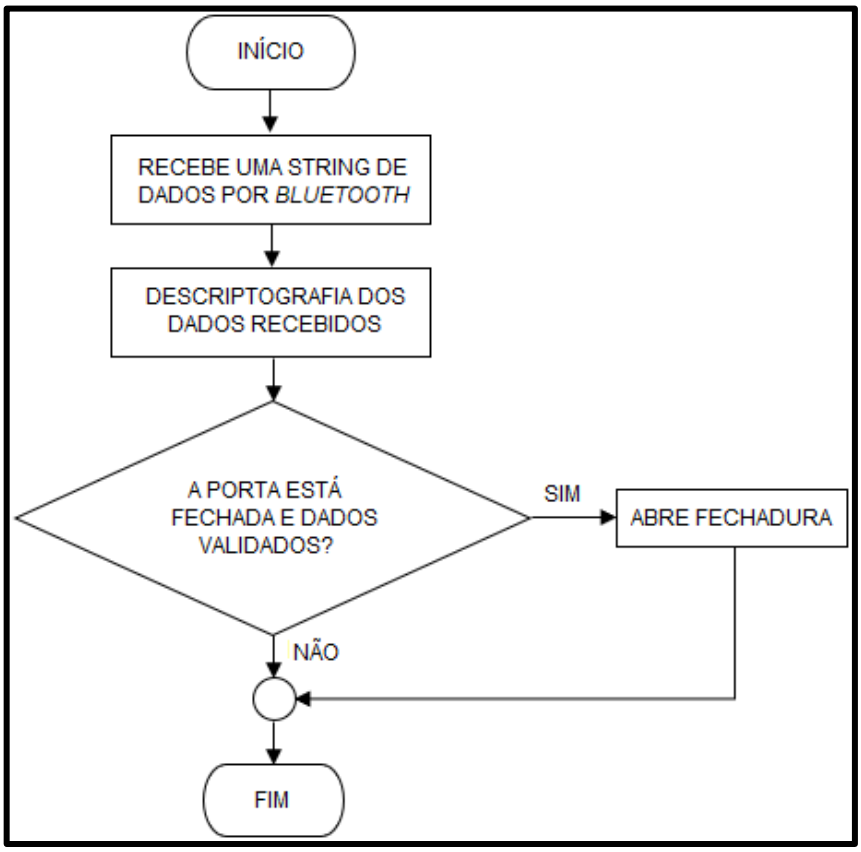
```
class MyCallbacks: public BLECharacteristicCallbacks {
  void onWrite(BLECharacteristic *pCharacteristic) {
    std::string rxValue = pCharacteristic->getValue();
      if (rxValue.length() > 0) {
        for (int i = 0; i < rxValue.length(); i++) {
          if (rxValue[i] == '0')
            rxValue[i] = 'a';
          if (rxValue[i] == '1')
            rxValue[i] = 'c';
                      .
                      .
                      .
          if (rxValue[i] == '.')
            rxValue[i] = 'j';
          usuarioBluetooth[i] = rxValue[i];
        }
        for (int i = 0; i < 40; i++){
          if (usuarioBluetooth[i] == ' ')
            usuarioBluetooth[i] = '\0';
        }
      }
      if ((rxValue.find("opendoor") != -1) && (sensorPorta == HIGH)) {
        usuario = usuarioBluetooth;
        abreFechadura();
      }
    }
};
                                    ...
```

Fonte: Autoria própria

O conteúdo da *string* recebida pela comunicação *Bluetooth* é armazenado em rxValue e seu conteúdo é verificado posição por posição. O caractere da posição é verificado por cada condicional e se for correspondente ao sinal da tabela de criptografia, será substituído na *string*. Após a decriptografia da string, o nome de usuário é armazenado, o código de abertura é procurado na *string* e a porta é aberta se a verificação for confirmada.

O fluxograma na figura 26 ilustra como o sistema opera, quando o acesso é realizado pela comunicação *Bluetooth.*

Figura 26 – Fluxograma para a comunicação Bluetooth



Fonte: Autoria própria

Tabela 4 – Dados para obter a chave simétrica

Chave para criptografia e decriptografia	
Caractere	**Chave criptográfica**
a	0
c	1
e	2
f	3
i	4
p	5
o	6
s	7
u	8
g	9
k	#
n	*
m	+
j	.

Fonte: Autoria própria

A tabela 4 apresenta como foi definida a chave de criptografia utilizada no projeto, por meio da substituição por caracteres numéricos ou especiais.

A figura 27 apresenta a inicialização do microcontrolador e de algumas funções. Nesta etapa são definidas as portas digitais como input ou output, o estado de inicialização (*LOW*, *HIGH*), a taxa de transferência de bits por segundo (*Serial.begin*) e a inicialização da comunicação via barramento SPI (*Serial Peripheral Interface*). Também são inicializadas algumas funções referentes a leitura de *tag*, comunicação *Wi-Fi* e *Bluetooth*.

Figura 27 – Inicialização do microcontrolador e funções

```
//----------------------Setup de inicialização---------------------------------

void setup() {
  Serial.begin(115200);
  SPI.begin();          // Inicia  SPI bus
  mfrc522.PCD_Init();   // Inicia MFRC522
  pinMode(0, OUTPUT);
  pinMode(2, INPUT);
  pinMode(4, OUTPUT);
  digitalWrite(0, LOW);
  digitalWrite(4, HIGH);


  BLEDevice::init("ESP32 Porta");
  BLEServer *pServer = BLEDevice::createServer();
  pServer->setCallbacks(new MyServerCallbacks());
  BLEService *pService = pServer->createService(SERVICE_UUID);
    pCharacteristic = pService->createCharacteristic(
                      DHTDATA_CHAR_UUID,
                      BLECharacteristic::PROPERTY_NOTIFY
                    );
  pCharacteristic->addDescriptor(new BLE2902());
  BLECharacteristic *pCharacteristic = pService->createCharacteristic(
                                         CHARACTERISTIC_UUID_RX,
                                         BLECharacteristic::PROPERTY_WRITE
                                       );
  pCharacteristic->setCallbacks(new MyCallbacks());
  pService->start();
  pServer->getAdvertising()->start();
  WiFi.mode(WIFI_STA);
  WiFi.begin("ANGELA 2G", "94988570");//Login na rede WiFi
  delay(2000);//Espera um tempo para se conectar a rede WiFi
}
```

Fonte: Autoria própria

No trecho de código acima, foi utilizado uma rede sem fio residencial, dado por “ANGELA 2G”, cuja senha é “94988570” e desta forma habilitou o link do sistema desenvolvido com a base de dados em nuvem.

O loop principal do código é apresentado na figura 28, bem como seu fluxograma na figura 29. É neste loop que o sistema fica a maior parte do tempo, aguardando por um método de acesso inserido pelo usuário.

Figura 28 – Loop principal do código

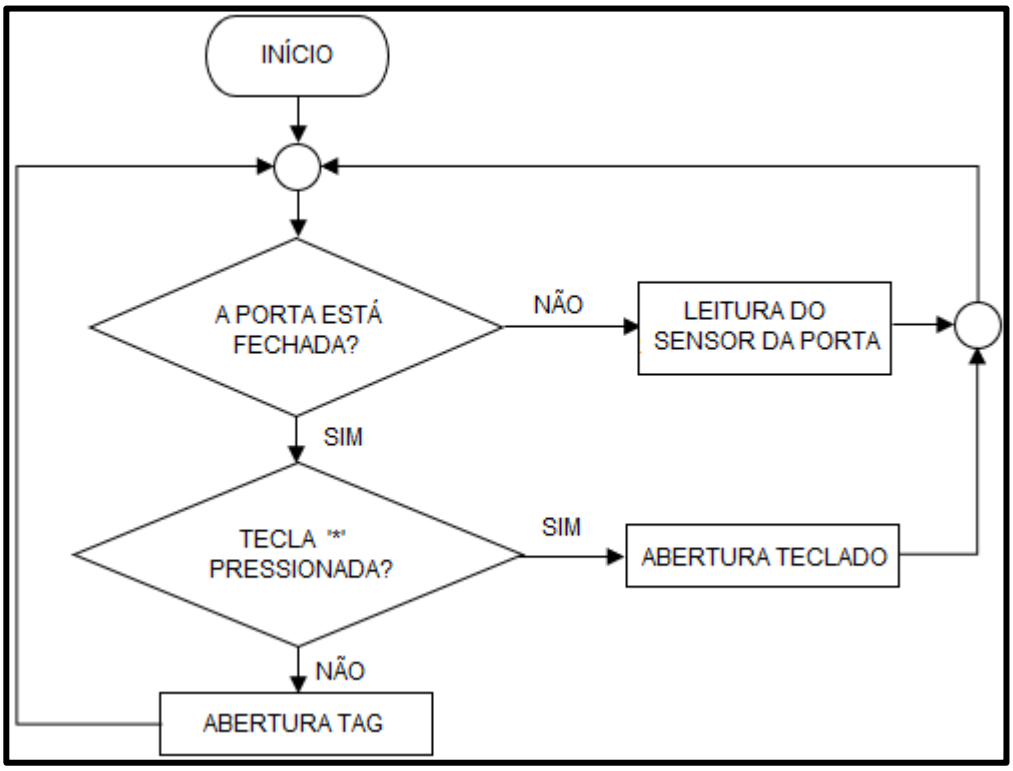
```
//----------------------Loop principal-----------------

void loop()               //Loop principal
{
  if(sensorPorta==HIGH){
    char teclaClicada = teclado.getKey();
    if(teclaClicada == '*'){
      acessoTeclado();
    }
    leituraTag();
  }
  sensorPorta = digitalRead(2);
}
```

Fonte: Autoria própria

Ao iniciar, o microcontrolador verifica se a porta está fechada e fica esperando uma entrada do usuário, fazendo a varredura dos métodos de acesso. Ao inserir algum dado via teclado numérico, realizar a comunicação *Bluetooth* ou aproximar uma *tag*, o código chama os procedimentos apresentados anteriormente, responsáveis pelos métodos de acesso, registro de dados e abertura da fechadura.

Figura 29 - Fluxograma do loop principal



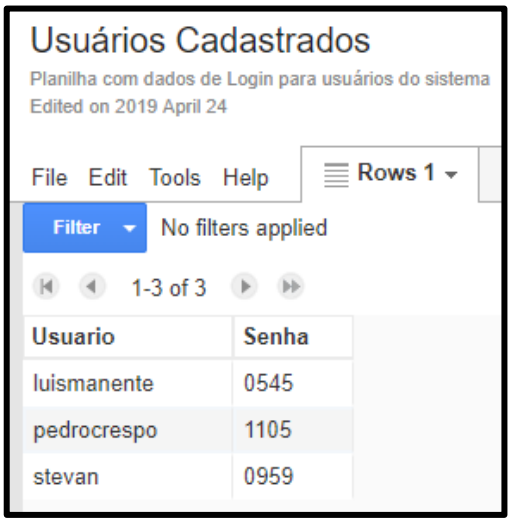
Fonte: Autoria própria

3.3.4 Armazenamento na Base de Dados

Para o armazenamento das informações necessárias ao funcionamento da fechadura, como os dados de login e usuários cadastrados, foram utilizados recursos gratuitos disponibilizados pelo *Google*, como as planilhas, formulários e *Fusion Tables.*

A *Fusion Table* utilizada serve para armazenamento dos dados de login no aplicativo. O aplicativo está configurado para acessar a planilha pela URL(*Uniform Resource Locutora*) e ter acesso as informações necessárias. A figura 30 apresenta a *Fusion Table* utilizada e configurada com os dados de login.

Figura 30 – *Fusion Table* utilizada no projeto

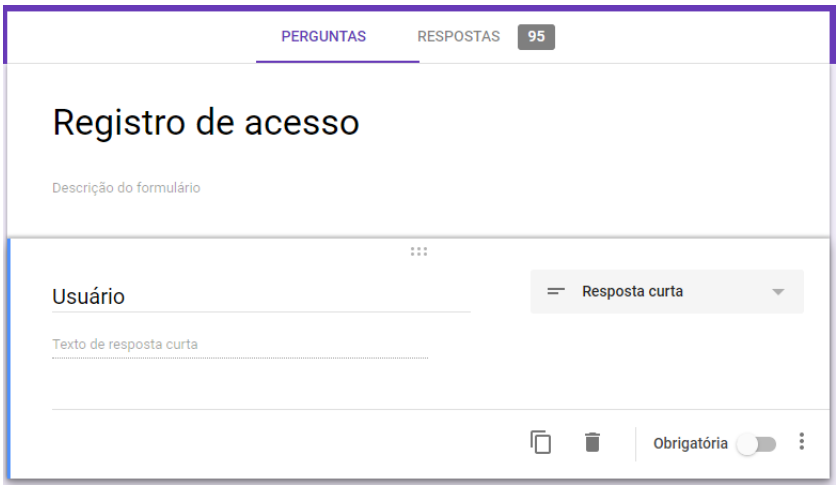


Fonte: Autoria própria

Uma das vantagens de se ter estas informações de login na nuvem é a possibilidade de conceder e bloquear acesso de forma remota à qualquer pessoa que possua um smartphone. Um recurso que pode ser muito útil em situações emergenciais, podendo conceder acesso e monitorar a utilização da fechadura.

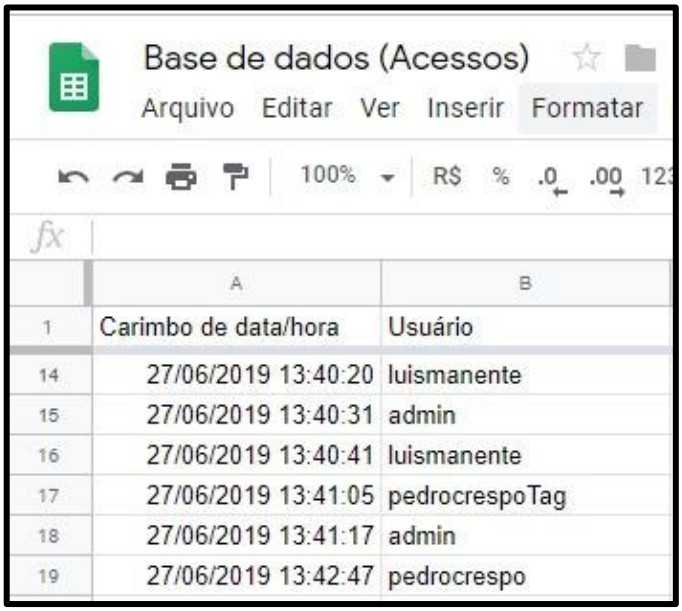
O formulário do *Google* utilizado é a forma de comunicação do ESP32 com a base de dados. Configurado para acesso à internet, o microprocessador acessa a URL do formulário e envia como resposta o nome do usuário que abre a fechadura. As respostas posteriormente são armazenadas em uma planilha do *Google*, junto com um carimbo de data e hora. Desta forma, é possível identificar o usuário que faz o acesso, bem como a data e hora do mesmo. A figura 31 apresenta o formulário para registro de acessos, e a figura 32 apresenta a planilha de registro de acessos.

Figura 31 – Formulário para registro de acessos



Fonte: Autoria própria

Figura 32 – Planilha de registro de acessos



Fonte: Autoria própria

3.3.5 Aplicativo Para Interface Com o Usuário

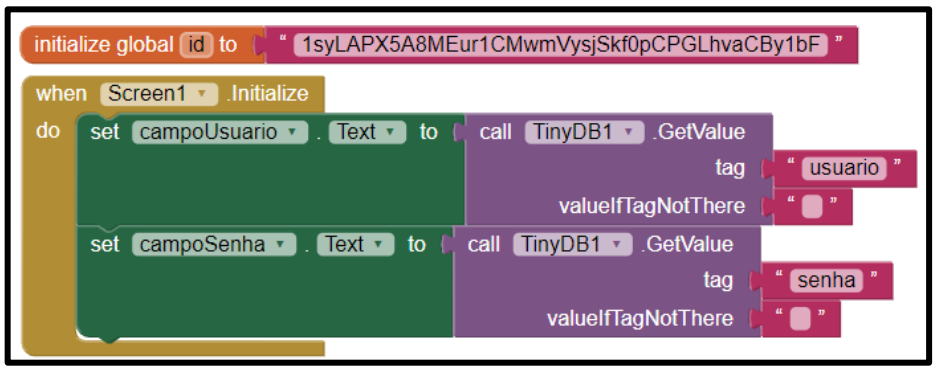
Para a construção do aplicativo, utilizamos a plataforma do MIT *App Inventor*, que é disponibilizada gratuitamente no link https://appinventor.mit.edu/. O aplicativo é composto de duas telas principais, a primeira para realizar o login no sistema e a segunda para conectar com a fechadura e realizar o acesso. A figura 33 apresenta a tela inicial do aplicativo, que possui duas caixas de texto para inserir os dados e um botão para realizar o login.

Figura 33 – Tela inicial do aplicativo

Fonte: Autoria própria

Nos blocos da figura 34, são inicializadas as configurações da tela 1. A inicialização global contém o endereço da *Fusion Table*, onde estão armazenadas as informações de login para o aplicativo.

Figura 34 – Inicialização da tela 1 do aplicativo

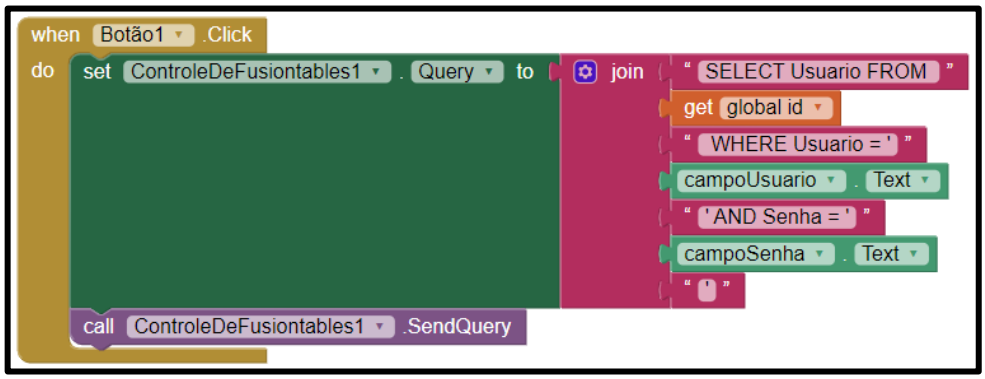


Fonte: Autoria própria

Quando a tela 1 se inicia, a função TinyDB1, que serve para armazenamento, é iniciada para que os dados do usuário usados pela última vez sejam preenchidos nos campos de login.

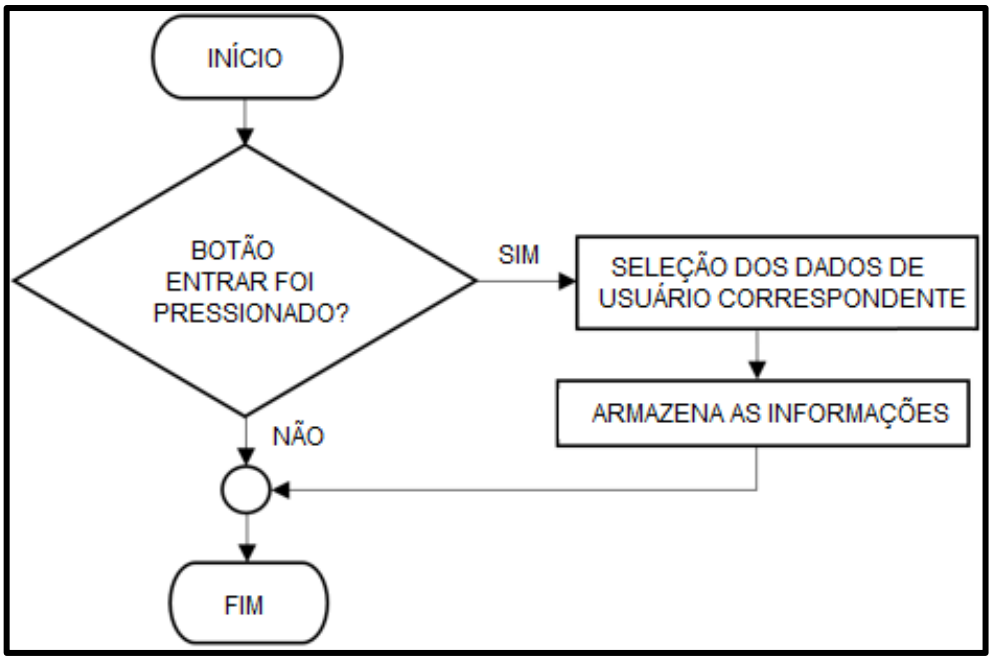
Quando o botão “Entrar” é clicado, o aplicativo realiza a comunicação com a base de dados através da rede *Wi-Fi* e faz uma varredura nas colunas de Usuário e Senha da *Fusion Table*. A figura 35 apresenta os blocos que realizam esta função, bem como o fluxograma equivalente, na figura 36.

Figura 35 – Bloco de configuração do botão 1



Fonte: Autoria própria

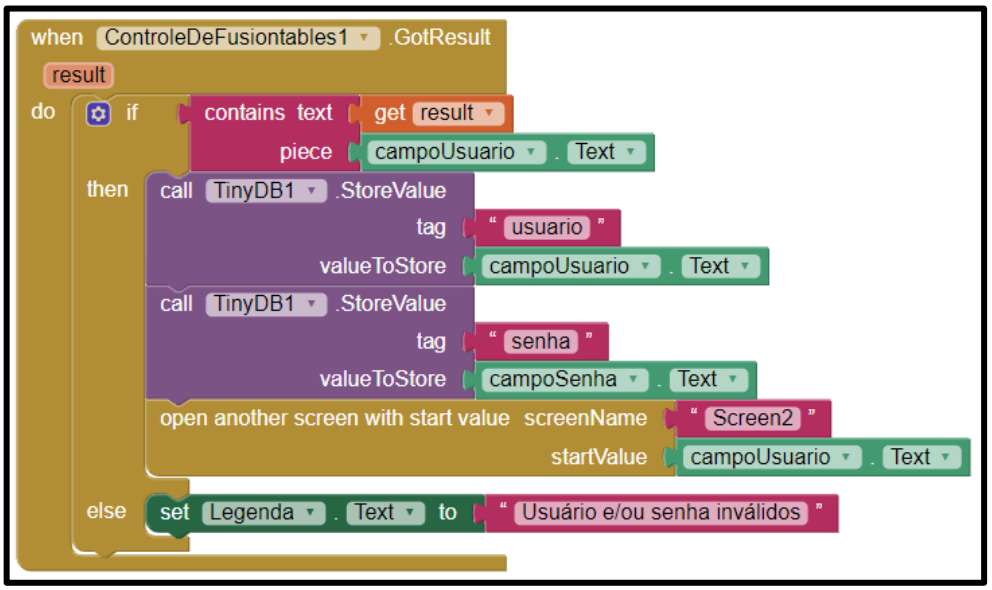
Figura 36 - Fluxograma do botão Entrar



Fonte: Autoria própria

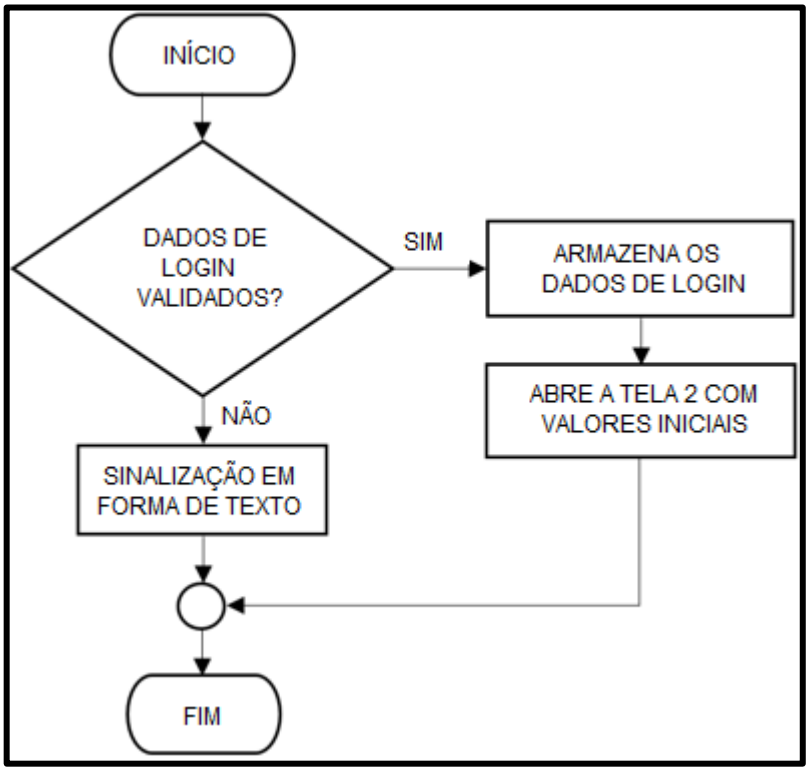
Quando a função retorna os dados da varredura, os blocos da figura 37 realizam a conferência dos dados de login. Se os valores estão cadastrados, ambos são armazenados em TinyDB1 para um próximo acesso e uma nova tela é aberta. O nome de usuário é carregado para a tela 2. Se as informações de login não estão cadastradas, a mensagem “Usuário e/ou senha inválidos” é exibida na tela na cor vermelha. O funcionamento descrito acima pode ser visto na figura 38.

Figura 37 – Bloco de consulta ao banco de dados



Fonte: Autoria própria

Figura 38 - Fluxograma para conferência de dados de login



Fonte: Autoria própria

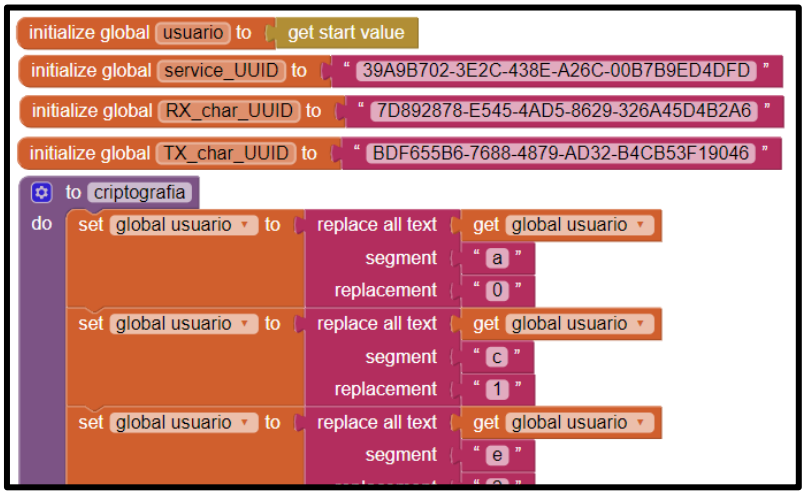
A figura 39 apresenta a tela 2 do aplicativo, onde é possível realizar a conexão *Bluetooth* com a fechadura, desconectar e abrir a fechadura conectada. Para a tela, foram adicionados o nome do usuário que está utilizando o aplicativo, um botão para conectar, um para desfazer a conexão e um para liberar o acesso.

Figura 39 –Tela após o login no aplicativo

Fonte: Autoria própria

Nos blocos da figura 40, a inicialização global resgata o nome do usuário que fez o acesso na tela 1 e os identificadores UUID para a comunicação *Bluetooth* são definidos. O procedimento denominado criptografia será responsável por criptografar a *string* que será enviada ao ESP32 para abertura da fechadura.

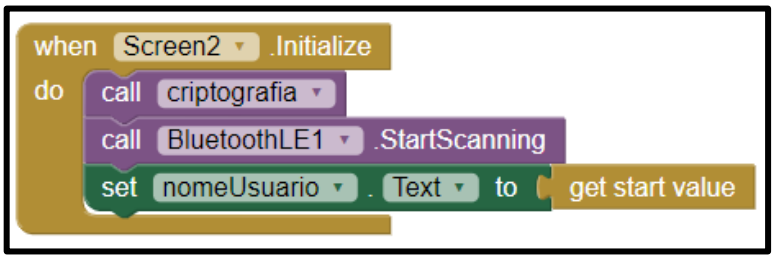
Figura 40 – Inicialização e configuração da criptografia



Fonte: Autoria própria

Quando a tela 2 é iniciada, o nome do usuário é criptografado com a chave do procedimento criptografia, a busca pelos dispositivos *Bluetooth* é iniciada e o nome do usuário é apresentado na caixa de texto no topo da tela. A figura 41 apresenta os blocos para estas funções.

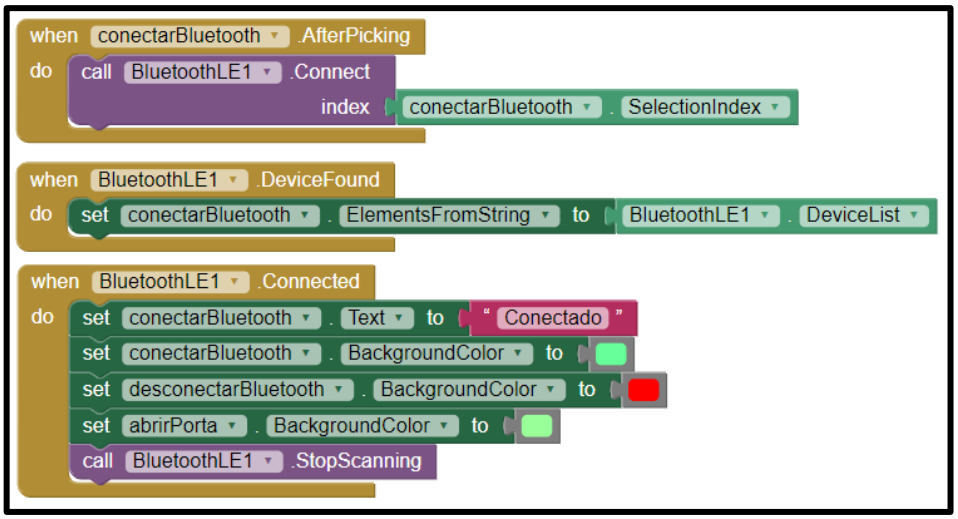
Figura 41 – Inicialização da tela 2 do aplicativo



Fonte: Autoria própria

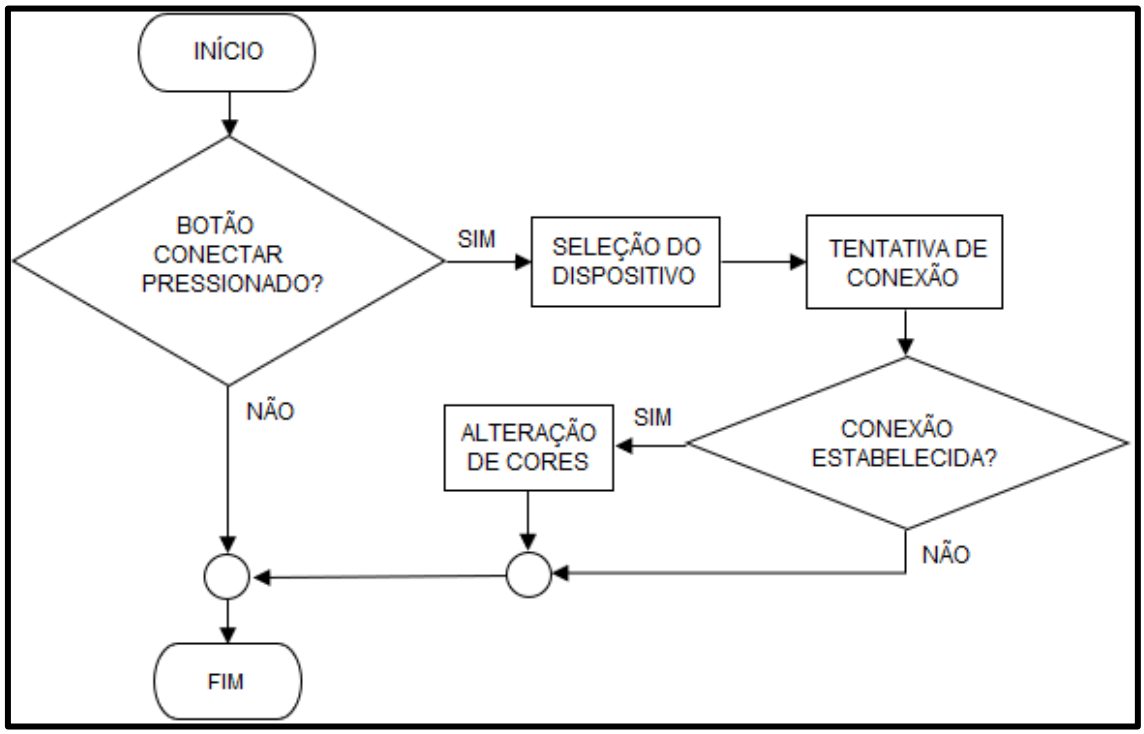
Ao pressionar o botão “Conectar”, o aplicativo apresenta uma lista com os dispositivos *Bluetooth* que podem ser conectados e no caso de mais de uma fechadura próxima com o mesmo sistema, poderia escolher nesta etapa em qual seria realizado o acesso. A figura 42 apresenta os blocos do aplicativo destinados à comunicação *Bluetooth* e alteração da tela após a conexão com o ESP32, o fluxograma na figura 43 e a figura 44 apresenta a janela de seleção de dispositivos *Bluetooth.*

Figura 42 – Conexão Bluetooth e alteração da tela



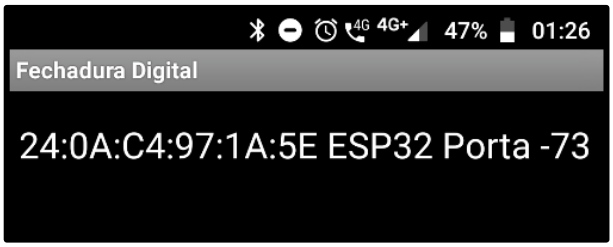
Fonte: Autoria própria

Figura 43 - Fluxograma ao estabelecer a conexão



Fonte: Autoria própria

Figura 44 – Tela de seleção do dispositivo Bluetooth para conexão



Fonte: Autoria própria

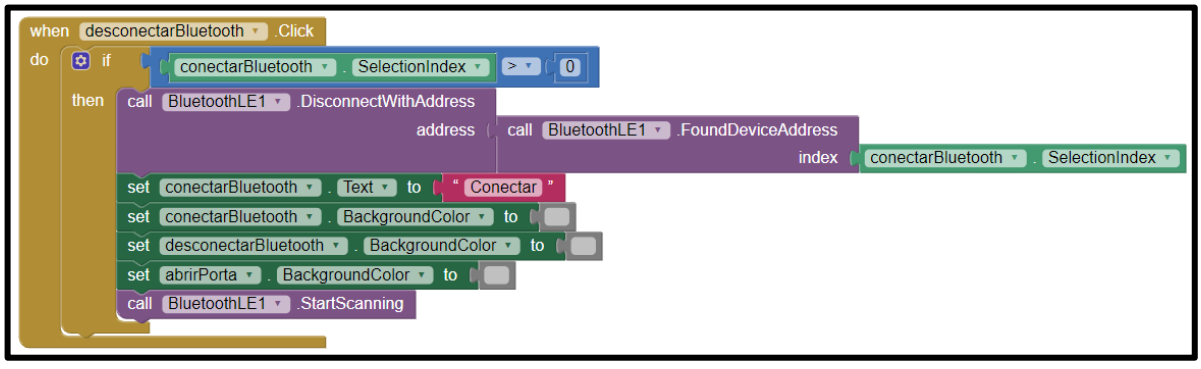
Após a conexão ser estabelecida, o botão “Conectar” é alterado para “Conectado” e as cores da página são alteradas, como mostra a figura 45.

Figura 45 –Tela após conexão com o ESP32

Fonte: Autoria própria

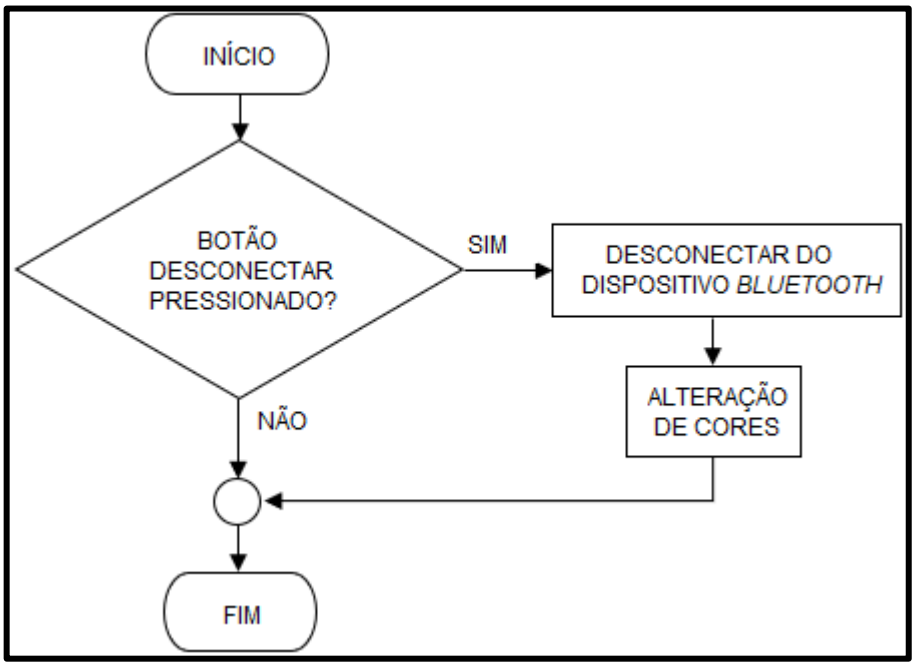
Se pressionado o botão “Desconectar”, os blocos apresentados na figura 46 realizam este procedimento, junto com a mudança das cores da tela 2 para o padrão inicial, quando não estava conectado. A figura 47, por meio do fluxograma, ilustra esta situação.

Figura 46 – Desconectar do dispositivo Bluetooth e alteração da tela



Fonte: Autoria própria

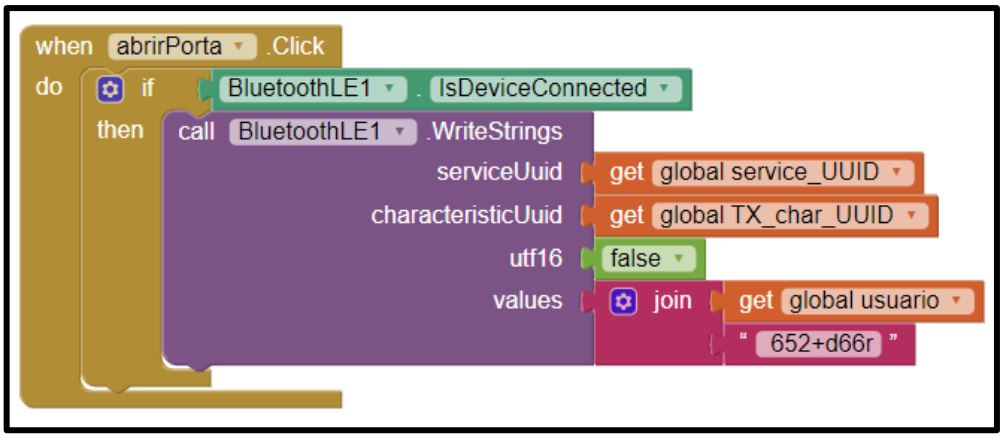
Figura 47 - Fluxograma ao desconectar o dispositivo do sistema



Fonte: Autoria própria

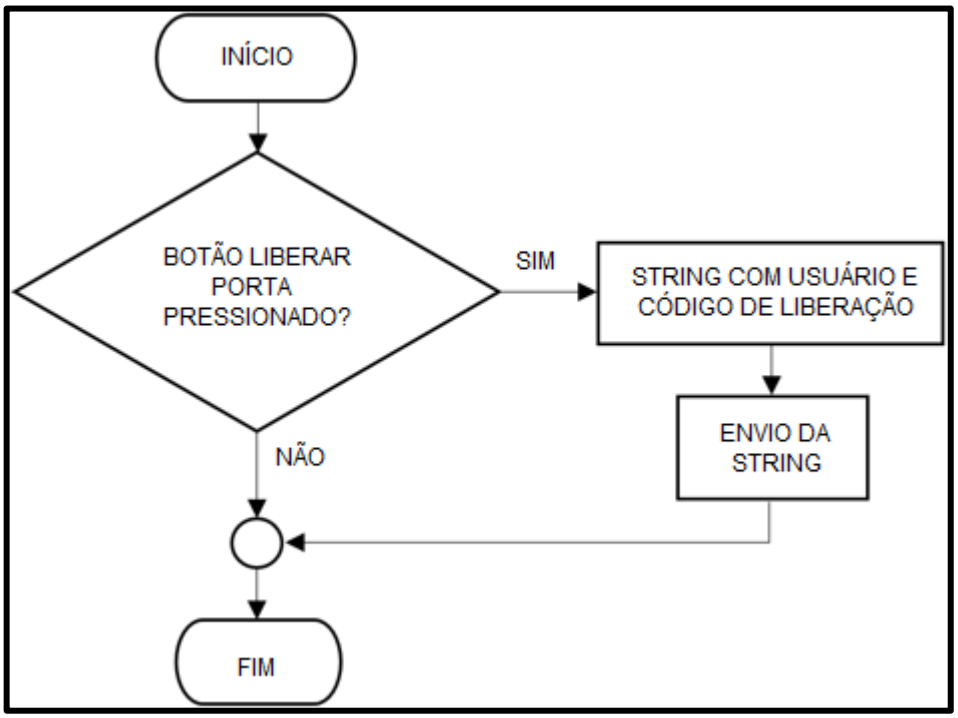
Por fim, quando o botão “Liberar Porta” é pressionado, a comunicação *Bluetooth* é estabelecida. Uma *string* contendo o nome do usuário que fez o acesso e um código criptografado “opendoor” para realizar a abertura é enviada ao ESP32. A *string* recebida pelo ESP32 é decriptografada, a fechadura é aberta e o nome de usuário que realizou o acesso é registrado na planilha do *Google* utilizada como banco de dados, como é exposto na figura 48.

Figura 48 – Comunicação para abertura da fechadura



Fonte: Autoria própria

Figura 49 - Fluxograma para abertura da fechadura



Fonte: Autoria própria

Após concluída a programação, o aplicativo pode ser exportado da plataforma App Inventor em formato .APK e instalado no dispositivo celular que irá operar a fechadura eletrônica. A publicação em plataformas como a *Google Play Store* não se torna viável devido aos custos de acesso, visto que a aplicação não tem fins lucrativos.

3.4 CUSTOS PARA IMPLEMENTAÇÃO DO PROTÓTIPO

Para o levantamento do custo total do protótipo, foram considerados os valores dos produtos junto com os custos de entrega. Os componentes mais caros foram adquiridos pela internet e os componentes pequenos (transistor, *buzzer*, LED, resistor) em loja física. A tabela 5 apresenta os custos do protótipo construído.

Tabela 5 – Custos para implementação do protótipo – junho/2019

Custos dos materiais para o protótipo – junho/2019	
Item	**Valor**
ESP32 NodeMCU-32S	R$ 44,90
Teclado matricial 4x3 membrana	R$ 11,95
Fechadura IPEC FC90	R$ 100,00
Leitor RFID RC-522	R$ 19,90
Fonte 12V 1A	R$ 8,90
Mini-360 DC-DC	R$ 5,70
Buzzer	R$ 3,00
TIP122	R$ 2,00
BC546	R$ 2,00
LED	R$ 1,00
Resistores	R$ 1,00
Cabos, Soldas, Conexões	R$ 10,00
Total	R$ 210,35

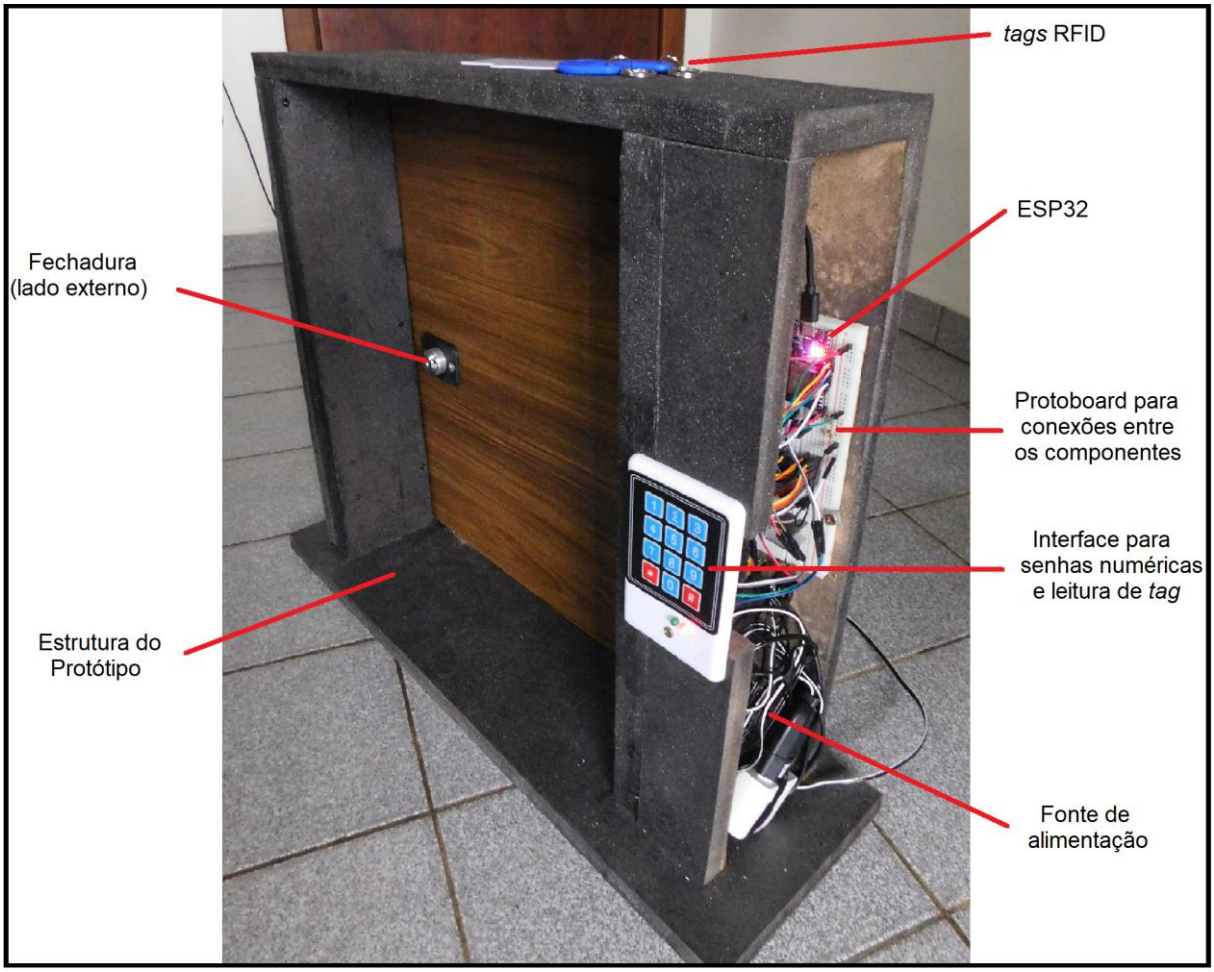
Fonte: Autoria própria

Como os custos apresentados são referentes ao valor total para a construção do protótipo, não podem ser levados em consideração para se estimar a produção do sistema em escala. Se os itens forem adquiridos em maior quantidade, os preços podem ser reduzidos, bem como os custos de entrega.

4 RESULTADOS

Após o estudo realizado, para compor o embasamento teórico sobre diferentes métodos de acesso, requisitos de segurança, fonte de energia, bem como do registro em uma base de dados em nuvem, foi construído um protótipo para o sistema proposto, como pode ser visto na figura 50.

Figura 50 - Protótipo desenvolvido



Fonte: Autoria própria

Para que os requisitos de projeto fossem atendidos, este protótipo conta com os três diferentes métodos de acesso propostos, que são por meio de uma senha numérica, leitura de cartão ou *tag* RFID e autenticação através de um aplicativo desenvolvido via comunicação *Bluetooth*. Todos esses métodos foram estudados previamente, visando o correto funcionamento no projeto e exercem as funções de trazer o caráter tecnológico ao mesmo, com o objetivo de adaptar uma

simples abertura de porta em algo mais inovador e ainda elevar a segurança para uma futura aplicação em ambientes que requerem um controle de acesso.

Os requisitos de protocolos de comunicação são atendidos, pois o ambiente de instalação da fechadura possui rede *Wi-Fi* para comunicação com a base de dados e o microcontrolador possui *Bluetooth* integrado para comunicar com a interface via *smartphone*.

Os múltiplos métodos de acesso garantem maior acessibilidade ao sistema, tornando sua utilização simplificada para qualquer pessoa. Quem não possui um *smartphone* ou tem limitações físicas, com uma deficiência visual, pode portar um cartão RFID e realizar o acesso sem dificuldades. Pessoas que preferem algo mais prático e não costumam carregar objetos, podem utilizar uma senha numérica. Como o leitor RFID utilizado trabalha na frequência de 13,56 MHz, pode se cadastrar o cartão de acesso ao transporte coletivo da cidade. Muitas empresas utilizam esta tecnologia de identificação de usuários. Desta forma, a utilização do sistema não vai exigir que a pessoa carregue mais um cartão ou tenha uma senha numérica.

Vale ressaltar que, durante os processos de codificação para utilização de cada método de acesso, a inovação foi gradual, sendo que a senha numérica é o mais simples, não evidenciando tanta complexidade ao passo que o desenvolvimento do aplicativo trouxe mais dificuldades, porém proporcionou ao projeto um caráter diferenciado.

A utilização de uma base de dados para registros enquadrou o projeto como uma aplicação dos conceitos de telemetria que, atualmente, está presente em diversas situações do nosso cotidiano. A planilha disponibilizada para o registro de acessos, não possui limitações de quantidade de dados inseridos, podendo armazenar as informações de acesso por logos períodos, sem a necessidade de descartá-las.

Por fim, na avaliação do protótipo, o mesmo atendeu os requisitos que foram propostos e mediante as sugestões de trabalhos futuros que serão expostas na seção 5, tal projeto pode ser incrementado, adaptando-se às tecnologias que surgem, servindo como um objeto de estudo de grande utilidade.

5 CONCLUSÃO E PROJEÇÕES DE TRABALHOS FUTUROS

O objetivo deste trabalho foi a construção de um protótipo para um sistema de controle de acesso por diferentes métodos, baseado na utilização na aplicação de sistemas microcontrolados e inserção dos conceitos de IoT, que pôde ser observado pela criação de um link entre o sistema físico e uma base de dados em nuvem para armazenamento dos acessos realizados, como exposto nas figuras da seção anterior.

O protótipo para teste do sistema foi criado seguindo a metodologia apresentada, para definir todos os requisitos de projeto necessários para o correto funcionamento do sistema. O protótipo funcionou conforme o esperado, não apresentando problemas de funcionamento e falhas graves. A base de dados foi criada para registar acessos e informações de usuários, utilizando as planilhas do *Google*, conforme previamente estabelecido nos objetivos.

A escolha do microcontrolador levou em conta diversos aspectos técnicos, em especial a velocidade de processamento, capacidade de memória e como a questão da conectividade com outros dispositivos e possibilidade de estabelecimento de uma conexão *wireless* estava presente. Desta forma, o ESP32 foi o dispositivo com melhor adesão aos requisitos do projeto. O código-fonte para o microcontrolador realizar todos os processos de autenticação e controle da fechadura foi elaborado, com seu funcionamento atendendo aos requisitos de projeto estabelecidos.

No quesito dos métodos de acesso escolhidos, a ideia de utilizar três meios distintos deu maior dinamismo ao projeto e evidenciou a evolução da tecnologia, partindo de uma tradicional senha numérica para autenticação por meio de leitura de *tags* e aplicativo via *Bluetooth*, que correspondem a possibilidades muito difundidas atualmente em sistemas de controle de acesso comercializados.

O aplicativo necessário para o usuário realizar o acesso pelo smartphone foi desenvolvido, funcionando com uma interface para a comunicação *Bluetooth*. Foi utilizada a plataforma MIT *App Inventor* para o desenvolvimento em linguagem de blocos, e está disponível para o sistema Android e iOS (*iPhone Operating System*).

Dentre as dificuldades encontradas neste projeto, pode-se citar o processo de definição dos equipamentos adequados para o protótipo, bem como a fase de

desenvolvimento dos códigos, tanto o que foi embarcado no microcontrolador, quanto o que se destinou ao aplicativo, em virtude da necessidade da realização constante de testes prévios. Vale ressaltar que a utilização do banco de dados em nuvem foi também um desafio encontrado, por ser algo novo e desta forma, necessitou um estudo específico para sua utilização de maneira correta.

Tendo em vista o presente trabalho, como projeção para trabalhos futuros pode-se citar o aprofundamento nos métodos de acesso ao sistema, com a implementação de leitor biométrico e reconhecimento facial, por exemplo, elevando a segurança, além de expandir o conceito de acesso para não somente movimentação de pessoas, mas também de bens materiais, que torna muito útil para ambientes que requerem um constante monitoramento. Uma outra inovação ao projeto seria criar uma base de dados mais completa, com uso de métodos de criptografia e um maior detalhamento dos acessos realizado.

REFERÊNCIAS

ATMEL. Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V. **Datasheet**. 2014. Disponível em: <https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf>. Acesso em: 18/04/19.

BOCCUCCI, Giuliano Estevam França. **Sistema de monitoração e controle de acesso para condomínios utilizando a tecnologia de identificação por rádio frequência (RFID).** 2010. 72f. TCC (Graduação) – Curso de Engenharia da Computação, Centro Universitário de Brasília, Brasília, 2010.

E-VAL TECNOLOGIA. **Criptografia de dados e gerenciamento de chaves**. Disponível em: <https://www.evaltec.com.br/criptografia-de-dados-e-gerenciamento-de-chaves/>. Acesso em: 28/05/19.

FIGUEIRA, Vitor Pinheiro. "**Internet das coisas": um estudo sobre questões de segurança, privacidade e infraestrutura.** 2016. 65f. TCC (Graduação) – Curso de Tecnologia em Sistemas de Computação, Universidade Federal Fluminense, Niterói, 2016.

FIGUEIREDO, Carlos M.S.; NAKAMURA, Eduardo. Computação Móvel: novas oportunidades e novos desafios. **Revista T&C Amazônia**, Manaus, n.2, p.16-28, jun. 2003.

FIORE, M. **Novidades no sistema de reconhecimento facial vão integrar diferentes redes sociais à partir da tecnologia**, 2018. Disponível em: <https://www.b9.com.br/95180/novo-sistema-de-reconhecimento-facial-funciona-integrando-diversas-redes-sociais/>. Acesso em : 19/03/19.

GOMES, R. O.; OLIVEIRA, L. M.; JÚNIOR, L. O. A.; OLIVEIRA, Â. R. Desenvolvimento de um software open source para controle digital remoto utilizando tecnologia Zigbee. In: CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, 61., 2013, Gramado. **Anais...**Gramado: 2013.

JUELS, Ari. **RFID Security and Privacy: a research survey.** RSA Laboratories, 2005.

LEITE, J. R.; MARTINS, P.S.; URSINI, E. L. **A internet das coisas (IoT): tecnologias e aplicações.** School of Technology, University of Campinas (UNICAMP), Limeira, 2017.

MACÊDO, D. **Conceitos sobre segurança em banco de dados.** 2011. Disponível em: < https://www.diegomacedo.com.br/conceitos-sobre-seguranca-em-banco-de-dados/>. Acesso em: 20/06/19.

MAGALHÃES, P. S.; SANTOS, H. D. Biometria e autenticação. IN: CONFERÊNCIA DA ASSOCIAÇÃO PORTUGUESA DE SISTEMAS DE INFORMAÇÃO, 04. 2003. Porto. **Atas...**Porto: 2003.

MICROCHIP. ATmega48A/PA/88A/PA/168A/PA/328/P. **megaAVR Datasheet.** 2018. Disponível em: <https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf >. Acesso em: 18/04/19.

MORAES, A.L. et al. **Redes Ad Hoc Protocolos DSR, AODV, OLSR, DSDV**, 2009. Disponível em: <https://www.gta.ufrj.br/grad/09_1/versao-final/adhoc/index.html>. Acesso em: 13/04/19.

NAKAYAMA, C. M. **Sistemas de comunicação BLUETOOTH voltada para automação residencial.** 2012. 61 f. Trabalho de Graduação (Graduação em Engenharia Elétrica) – Faculdade de Engenharia do Campus de Guaratinguetá, Universidade Estadual Paulista, Guaratinguetá, 2012.

RODRIGUES, J.C. **Internet & Conectividade: uso versus utilidade**, 2017. Disponível em: <https://www.updateordie.com/2017/04/04/internet-conectividade-uso-versus-utilidade/>. Acesso em: 07/04/19.

RODRIGUES, O. S.; ROCHA, C. S. IoT e conectividade: índices do futuro em espaços museais. In: ENCONTRO INTERNACIONAL DE ARTE E TECNOLOGIA, 15., 2016. Brasília. **Anais...**Brasília: 2016.

SAB, G. A. A.; FERREIRA, R. C.; ROZENDO, R.G. **Near field communication.** 2013. Disponível em: < https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2013_2/nfc/funcionamento.html> Acesso em: 20/05/19.

SIQUEIRA, T. S. ***Bluetooth* – Características, protocolos e funcionamento.** Instituto de Computação, Universidade Estadual de Campinas, 2006.

SYSTEMS, Espressif. Esp32 Series. **Datasheet**. 2019. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>. Acesso em: 05/06/19.

SYSTEMS, Espressif. Esp8266EX. **Datasheet**. 2018. Disponível em: <https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf >. Acesso em: 03/04/19.

TIZYI, H.; RIOUCH, F.; TRIBAK, A.; NAJID, A.; MEDIAVILLA, A. **Compact dual-band microstrip antenna for handheld RFID reader.** 2015.

UUID GENERATOR. **Online UUID generator.** Disponível em: <https://www.uuidgenerator.net/>. Acesso em: 01/06/19.

WELEKAR, A. R.; BORKAR, P.; DORLE, S. S. **Comparative study of IEEE 802.11, 802.15, 802.16, 802.20 standards for distributed VANET.** Nagpur, 2012.