UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS

JÉSSICA DA SILVA PAVAN

PROTÓTIPO DE UM SISTEMA DE GERENCIAMENTO DE ACESSO

CAXIAS DO SUL

2022

JÉSSICA DA SILVA PAVAN

PROTÓTIPO DE UM SISTEMA DE GERENCIAMENTO DE ACESSO

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia de Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Orientador: Prof. Dra. Helena Graziottin Ribeiro

CAXIAS DO SUL

2022

JÉSSICA DA SILVA PAVAN

PROTÓTIPO DE UM SISTEMA DE GERENCIAMENTO DE ACESSO

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia de Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Aprovado em 01/07/2022

BANCA EXAMINADORA

_____

Prof. Dra. Helena Graziottin Ribeiro
Universidade de Caxias do Sul - UCS

_____

Prof. Msc. Marcos Eduardo Casa
Universidade de Caxias do Sul - UCS

_____

Prof. Msc. Ricardo Leal Costi
Universidade de Caxias do Sul - UCS

Este trabalho é dedicado às mulheres nas ciências exatas.

AGRADECIMENTOS

Agradeço especialmente a minha tia Grasi que me apoiou na minha adolescência, na fase de escolha do curso, me auxiliou com toda a papelada para conseguir a bolsa na faculdade, e me deu muito suporte em todo esse período. Sou muito grata por tudo.

Agradeço também ao meus pais pelo incetivo e amparo na minha criação, grande parte da importância que dei aos estudos foi um valor que a Rosi me passou. Também juntamente com a Grasi, minha vó Marlene e Gabriel que dividiram o pagamento do meu primeiro semestre até saber que eu ganharia a bolsa e por todo o suporte e convivência nessa jornada acadêmica.

A meu namorado Vinicius que nunca mediu esforços para tentar me ajudar, vivenciou comigo todas as dificuldades e ansiedades que o curso me trouxe desde que está comigo, procurou ter paciência e fez o seu melhor para me deixar bem. Foi a pessoa que eu sentia que mais entendia como era difícil e fez eu sentir que estava junto comigo para enfrentar as dificuldades. Também sou imensamente grata.

Agradeço a minha orientadora, professora Helena pelo suporte nessa etapa final da graduação, e aos professores que de alguma forma impactaram minha visão de mundo, tanto na faculdade quanto no SENAI, os quais admiro e se dedicam a passar conhecimento.

Por fim, a todos os meus amigos e pessoas que passaram pela minha vida nesse período, cada um foi importante de uma forma diferente em períodos distintos. Em especial aos meus amigos Venicius, Bruno, Adriano, China, Ingrid, Amanda, Rafael, Togni e Burile.

RESUMO

Para se garantir a segurança em ambientes controlados se faz necessário supervisionar quem tem acesso ao local. Em ambientes em que há um fluxo maior de pessoas, esse controle precisa ser efetuado de forma dinâmica, possibilitando a fluidez de pessoas mas ainda mantendo certo monitoramento sob quem está tendo este acesso. A eletrônica em conjunto com as tecnologias de rede sem fio, oferecem uma possibilidade de automação, capaz de fornecer um recurso para validação e liberação de acesso de forma automática. Visando ter um meio de gerenciar este controle de acesso, este trabalho se propôs a desenvolver um protótipo de um sistema web de gerenciamento, a partir de uma rotina de detecção de dispositivos com a tecnologia de rede sem fio Bluetooth ativada. Com o auxílio de cadastros de pessoas, de seus dispositivos e de perfis, foi possível fazer uma validação de acesso com os dados armazenados em um banco de dados. Como proposta de solução foi desenvolvido um protótipo de software que além dos cadastros, conta com uma interface de bloqueio ou liberação de acesso, atendendo o objetivo do trabalho. O mesmo teve como resultado um sistema que permite através do navegador web gerenciar quem tem acesso ao ambiente, com o intuito de oferecer um recurso que auxilie na manutenção da segurança de bens e indivíduos contidos no local.

Palavras-chave: Sistema Web. Bluetooth. Segurança.

ABSTRACT

To ensure security in controlled environments, it is necessary to supervise who has access to the site. In environments where there is a greater flow of people, this control needs to be carried out dynamically, allowing for the flow of people but still maintaining a certain monitoring of who is having this access. Electronics in conjunction with wireless network technologies offer a possibility of automation, capable of providing a feature for automatic validation and release of access. In order to have a way to manage this access control, this work aimed to develop a prototype of a web management system, based on a routine for detecting devices with Bluetooth wireless network technology enabled. With the help of registers of people, their devices and profiles, it was possible to validate access with the data stored in a database. As a solution proposal, a software prototype was developed that, in addition to the registrations, has an interface to block or release access, meeting the objective of the work. The same resulted in a system that allows, through the web browser, to manage who has access to the environment, in order to offer a resource that helps in maintaining the security of goods and individuals contained in the place.

Keywords: Web System. Bluetooth. Security.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

TCC Trabalho de Conclusão de Curso

IoT *Internet of Things*

HTML *Hyper Text Markup Language*

CSS *Cascading Style Sheets*

SIG *Special Interest Group*

URL *Uniform Resource Locator*

URI *Uniform Resource Identifier*

GSMA *Global System for Mobile Communication Association*

HTTP *HyperText Transfer Protocol*

W3C *World Wide Web Consortium*

RFID *Radio-Frequency Identification*

ECMA *European Computer Manufactorers Association*

SQL *Structured Query Language*

ANSI *American National Standards Institute*

SGBD Sistema Gerenciador de Banco de Dados

SGBDs Sistemas Gerenciadores de Banco de Dados

CWI *Centrum Wiskunde Informatica*

MVC Padrão *Model–View–Controller*

MVP Padrão *Model–View–Presenter*

MVVM Padrão *Model–View–ViewModel*

IDE *Integrated Development Environment*

API *Application Programming Interface*

ER *Entidade-Relacionamento*

BFF *Back-end for Front-end*

PPI *Python Package Index*

JSON *JavaScript Object Notation*

REST *Representational State Transfer*

IP *Internet Protocol*

ASCII *American Standard Code for Information Interchange*

SUMÁRIO

1 INTRODUÇÃO

Com os altos índices de violência, furto e invasão, se tornou imprescindível algum tipo de ferramenta de segurança visando a proteção de bens e indivíduos. Em ambientes residenciais se utilizam recursos mais convencionais e rígidos para garantir a segurança, já em ambientes empresariais e comerciais se faz necessário o uso de recursos mais dinâmicos devido ao grande fluxo de pessoas. Mesmo se tendo recursos menos restritivos do que no ambiente residencial, ainda se tenta prever algum tipo de controle de acesso na medida do possível.

Na tecnologia da informação, o desenvolvimento de programas embarcados e a microeletrônica estão em constante evolução e obtiveram grandes avanços no últimos anos, corroborando para o surgimento do conceito de *Internet of Things* (IoT). Estes avanços possibilitam uma série de automações através da troca de dados entre dispositivos. Podem também oferecer maior eficiência e possibilidades se tratando do campo de segurança eletrônica, que visa prover segurança através de sistemas com aplicações de diferentes tecnologias.

Há diferentes propostas de sistemas de controle de acesso utilizando tecnologias sem fio, como por exemplo alguns trabalhos correlacionados: Protótipo de Sistema para Monitoramento de Presença (MELO, 2018) que utiliza um módulo Bluetooth embutido em uma placa eletrônica para trabalhar com a tecnologia, Desenvolvimento de um Sistema de Controle de Acesso com Armazenamento em Nuvem (MANENTE; CRESPO, 2019) que utiliza várias tecnologias como *Radio-Frequency Identification* (RFID) para controle de acesso, e um Sistema de Controle Baseado em Localização Interna Utilizando Dispositivos Bluetooth para Sistemas IoT (HUH; SEO, 2017) que utiliza um servidor de localização para controle. Trabalhos que contém similaridades mas que apresentam diferentes propostas para sua implementação e utilização.

O Bluetooth é uma tecnologia de rede sem fio de curto alcance, e está presente em todos os dispositivos móveis fabricados atualmente, assim, surgiu a proposta de utilizar os próprios *smartphones* que possuem a tecnologia como recurso para controle de acesso a ambientes controlados.

1.1 OBJETIVOS

Este trabalho tem o objetivo de desenvolver um protótipo de um software de gerenciamento de acesso de pessoas a ambientes controlados, via liberação de acesso através de dispositivos que utilizam a tecnologia Bluetooth. A partir do objetivo geral, identificou-se os objetivos específicos que se seguem:

1. Detectar os dispositivos com o sinal Bluetooth ativado próximos do local de acesso ao ambiente controlado.

2. Construir uma base de dados a partir de uma interface de cadastros. Base com dispositivos vinculados a pessoas, que possuem acesso liberado ao ambiente.

3. Desenvolver uma interface de autenticação, e de liberação ou bloqueio do acesso.

4. Desenvolver um protótipo de um sistema de gerenciamento de acesso de pessoas a ambientes controlados.

1.2 ESTRUTURA DO TRABALHO

Inicialmente no Capítulo 2 é apresentada a importância da segurança em ambientes controlados, evidenciando recursos que podem ser utilizados como meio de controle de acesso. O Capítulo 3 expõe o conceito de desenvolvimento web, e algumas das principais tecnologias envolvidas em sua implementação. A metodologia para o desenvolvimento do protótipo do software proposto é contemplada no Capítulo 4. O fluxo de utilização da aplicação é apresentado e exemplificado no Capítulo 5, dando suporte a explicação técnica em detalhes do desenvolvimento do trabalho, que foi abordada no Capítulo 6. E por fim, o Capítulo 7 apresenta as conclusões finais do trabalho.

2 SEGURANÇA E RECURSOS DE ACESSO A AMBIENTES CONTROLADOS

Em ambientes controlados busca-se manter a segurança, tanto da integridade física das pessoas que se encontram no local, quanto de bens materiais e intelectuais. Existe uma gama de tecnologias e procedimentos que visam ter maior controle sob quem tem acesso a estes locais. A tecnologia em constante evolução na área de segurança vem automatizando processos através da troca de recursos puramente mecânicos, por recursos eletrônicos e/ou magnéticos (PORTELLA, 2011). A segurança eletrônica tem o propósito de garantir a integridade de pessoas e bens através da utilização de equipamentos eletrônicos, alguns como câmeras, cercas elétricas, e autenticação digital são exemplificados na Figura 1, utilizados em sistemas e automações.

Figura 1 – Equipamentos de segurança eletrônica

Fonte: Advocacia Maciel [1]

2.1 AMBIENTE RESIDENCIAL

Em residências existe uma maior preocupação com furtos na ausência dos habitantes, onde são utilizadas soluções mais convencionais como paredes, portas, travas, trincos, cadeados, sendo praticamente unânime o uso de fechaduras (GUEDES; SANTOS, 2016).As fechaduras contam com uma infinidade de tipos, mecanismos e tecnologias. Também é comum o uso de portões eletrônicos para se ter acesso por meio de um veículo ao perímetro residencial, fazendo o controle de acesso convencionalmente de forma manual, já que existe um fluxo pequeno de pessoas.

[1] Disponível em: <https://advocaciamaciel.adv.br/sectech-engrossa-a-seguranca-eletronica/>. Acesso em 12 out. 2021

Em várias residências tem-se também cães como uma forma de proteção, além de animal de estimação é um cão de guarda, muitas vezes ficando soltos no pátio, podendo alarmar os moradores da presença de desconhecidos e também causar medo em possíveis invasores. Por este fator é comum se ver placas de alerta de cães de guarda.

2.2 AMBIENTE EMPRESARIAL

No ambiente empresarial o controle de acesso também é rígido, podendo se fazer uso de barreiras físicas menos convencionais que no perímetro residencial, como catracas, torniquetes e cancelas para os veículos, possibilitando uma maior automatização do acesso (ALMEIDA, 2018). Fechaduras eletromagnéticas como a exemplificada na Figura 2 são amplamente utilizadas, tanto em ambientes empresariais quanto em ambientes comerciais mais restritos, devido a necessidade de uma fácil abertura, e a um fluxo maior de pessoas (GUEDES; SANTOS, 2016).

Figura 2 – Exemplo de fechadura eletromagnética

Fonte: Upper Seg [2]

Uma fechadura deste tipo funciona basicamente se mantendo fechada ao receber uma tensão, e abrindo ao interromper a mesma. Para se ter controle sob a liberação do acesso dessas barreiras físicas, é possível se fazer comprovações de que se tem permissão de acesso ao local. Algumas dessas formas de validações são senhas, características físicas e únicas humanas, módulo RFID, troca de dados por *wi-fi*, e por Bluetooth (MELO, 2018). Também muitas vezes se faz uso de um porteiro, recurso humano que complementa o controle de acesso.

Um sistema utilizando segurança eletrônica, envolve várias áreas e profissionais, desde o planejamento até a implementação, que resulta em um sistema de segurança onde podem ser aplicadas diferentes tecnologias (ALMEIDA, 2018).

[2] Disponível em: <https://www.upperseg.com.br/interfonia/fechaduras/fechadura-eletromagnetica/fechadura-eletroima-magnetica-universal-s–sensor-fe-20150-intelbras/>. Acesso em 29 set. 2021

2.3 AUTOMATIZAÇÃO DO ACESSO

Pode-se automatizar a liberação do dispositivo e/ou periférico utilizado no bloqueio físico ao ambiente, sendo provável a liberação da passagem após uma validação de permissões.

Geralmente se faz uso de um microcontrolador para a automação do sistema, que contém recursos como memória e processador. Ele é contido em um circuito integrado conforme ilustrado na Figura 3, com periféricos de entrada e saída que são programáveis, podendo ser responsável por várias funções como liberar tensão para um dispositivo. Assim, o microcontrolador exerce o controle sob o funcionamento do dispositivo. É capaz de liberar a passagem física ao ambiente, após programação do mesmo, utilizando um código que valida o acesso (GUEDES; SANTOS, 2016).

Figura 3 – Exemplo de microcontrolador

Fonte: Achei componentes eletrônicos [3]

Visando o conceito de IoT, é possível fazer esta validação através da colocação de módulos de conexão sem fio e de baixo custo em espaços internos, com o propósito de troca de dados entre dispositivos (COLLOTTA *et al.*, 2018). As tecnologias de conexão sem fio, também estão presentes em dispositivos móveis, oferecendo mais uma opção de ferramenta para tal validação.

2.3.1 Internet das Coisas

A "Internet das Coisas" em tradução livre de *Internet Of Things* – IoT, é uma rede formada de objetos físicos, que são dispositivos, instrumentos, veículos, construções, demais itens embutidos com eletrônicos, circuitos, softwares, e sensores. Ou seja, itens que possuem conectividade de rede, conectividade que possibilita a esses objetos receberem e enviarem dados (GOKHALE; BHAT; BHAT, 2018). Portanto o conceito se propõe a conectar "coisas" e possibilitar a comunicação entre elas conforme ilustra a Figura 4.

[3] Disponível em: <https://www.acheicomponentes.com.br/loja/noticia.php?loja=648216id=11

[4] Disponível em: <https://www.grupomult.com.br/iot-comunicacao-de-dados-e-microservicos-uma-visao-integrada-para-suportar-industria-4-0/

Figura 4 – Conceito de internet das coisas

Fonte: Grupo Mult [4]

2.4 DISPOSITIVOS MÓVEIS

Dispositivos móveis são caracterizados pela autonomia, facilidade de locomoção e manuseio dos mesmos. Para ser um dispositivo que se pode levar a qualquer lugar, não é viável se ter fios pendurados para se fazer uma conexão a rede, por isso dispositivos móveis fazem uso de redes sem fio (SILVA, 2017).

O conceito IoT tem um grande aliado nos dispositivos móveis, que contém uma série de sensores e emissores de sinais. Dispositivos móveis, como *smartphones* são usados pela maioria da população. No mundo, mais de 5,2 bilhões de pessoas usam algum tipo de dispositivo móvel (GLOBAL SYSTEM FOR MOBILE COMMUNICATION ASSOCIATION, 2021), segundo dados do relatório global de 2020 da *Global System for Mobile Communication Association* (GSMA), empresa que disponibiliza um relatório anual com dados econômicos referentes a dispositivos móveis em todo o mundo.

Estes dispositivos vem por padrão com tecnologias de redes sem fio, como *wi-fi* e o Bluetooth. Assim, se apresenta a possibilidade de usar o próprio dispositivo de uso pessoal como ferramenta para se conectar a uma grande variedade de dispositivos eletrônicos. O uso de *smatphones* como ferramenta de auxílio para controle de automações usando o conceito de IoT, é uma vantagem pelo fator financeiro, poupando o investimento de ter que adquirir um dispositivo para exercer esta função, já que o usuário já o possui.

2.5 TECNOLOGIA BLUETOOTH

As empresas L. M. Ericsson, IBM, Intel, Nokia e Toshiba tinham o interesse em possibilitar a conexão de aparelhos celulares com outros dispositivos, sem a necessidade de cabos para tal conexão. Então, em 1994 formaram a *Special Interest Group* (SIG) a fim de desenvolver esta tecnologia que possibilitaria a interconexão de dispositivos de comunicação e de computação, denominando-a de Bluetooth [5]. O Bluetooth 1.0 teve sua primeira versão lançada em 1999, e desde então o grupo permanece.

Os avanços na área da microeletrônica possibilitaram integrar em um único chip diversos blocos de microssistemas de comunicação de dados sem fio. Então a partir da década de 1990 transceptores de baixo custo passaram a fazer parte de dispositivos eletrônicos, possibilitando a interconexão entre eles, ou com um módulo avulso da tecnologia sem a necessidade de fios (ROCHOL, 2018). A popularidade da tecnologia foi se ampliando com o passar dos anos, refletindo na atualidade, onde todos os dispositivos eletrônicos de consumo usam a tecnologia (TANENBAUM; WETHERALL, 2011). A logomarca da tecnologia Bluetooth mostrado na Figura 5 é bastante difundida e ao ser vista em dispositivos, associa e indica a presença da tecnologia no mesmo (GUERRA, 2020).

Figura 5 – Logomarca da tecnologia Bluetooth

Fonte: Bluetooth [6]

A tecnologia Bluetooth possibilita a transferência de dados através de ondas de rádio, interconectando um ou mais dispositivos. É possível encontrar a tecnologia numa ampla gama de dispositivos eletrônicos, como *notebooks*, *smartphones*, rádios, computadores, relógios, *tablets*, impressoras, microfones, fones de ouvido, e muitos outros dispositivos eletrônicos conforme ilustra a Figura 6, que mostra a interconexão de dispositivos através da rede sem fio Bluetooth. A tecnologia permite que esses dispositivos que possuem um módulo Bluetooth no se hardware se comuniquem e através da troca de informações entre si.

[5] Disponível em: <https://www.bluetooth.com/>. Acesso em 29 set. 2021
[6] Disponível em: <https://www.bluetooth.com/>. Acesso em 29 set. 2021
[7] Disponível em: <https://miracomosefaz.com/o-que-e-bluetooth-e-para-que-serve-como-funciona-e-seu-uso-em-dispositivos-moveis/>. Acesso em 29 set. 2021

Figura 6 – Dispositivos interconectados por Bluetooth

Fonte: Mira Como Se Faz [7]

A tecnologia Bluetooth é utilizada para propósitos de menores distâncias. Assim como o *wi-fi*, é uma tecnologia de rede sem fio, mas possuem diferenças, sendo a mais importante o alcance do sinal e o consumo de energia. Enquanto a tecnologia *wi-fi* consome mais energia e possui um grande alcance, chegando a centenas de metros, a tecnologia Bluetooth se limitava a 1, 10 e 100 metros (GUERRA, 2020).

2.5.1 Bluetooth 5

Vinte e dois anos após o primeiro padrão do Bluetooth, o 1.0 e seis anos após a liberação do Bluetooth 4.0, a (SIG) lançou o Bluetooth 5 no final de 2016, trazendo significantes inovações na área de tecnologia de conexão sem fio de curtas distâncias, aumentando seus recursos, conforme ilustra a Figura 7. Mantém algumas especificidades do padrão anterior, como a frequência de trabalho de 2,4GHz, e recursos importantes como a criptografia e baixo consumo de energia, consumindo ainda menos que seu antecessor. Mas o novo padrão traz o diferencial de uma maior velocidade, maior capacidade na transmissão dos dados, maior alcance e redução de interferências de outros módulos de rede sem fio (COLLOTTA *et al.*, 2018).

[8] Disponível em: <https://mundoconectado.com.br/artigos/v/13913/bluetooth-5-conheca-as-melhorias-da-conexao-sobre-versoes-anteriores/>. Acesso em 29 set. 2021

Figura 7 – Recursos Bluetooth 5

Fonte: Adaptado e traduzido de Mundo Conectado [8]

O padrão anterior 4.X tinha um alcance de 50 a 100 metros em linha reta, em um ambiente externo, sem obstruções, e um alcance limitado de 10 a 20 metros em ambientes internos. O Bluetooth 5 tem a meta de quadruplicar este alcance, tendo na pior das hipóteses o alcance de 200 metros no exterior, e cerca de 40 metros em ambientes internos. Também comparando ao padrão anterior 4.x, que tinha uma velocidade de 1 Mbps, o novo padrão apresentou uma nova conexão de alta velocidade, dobrando seu valor, assim, passando para 2 Mbps .

Com o Bluetooh 4.x era possível enviar mensagens de 31 bytes, um tamanho muito pequeno se levar em consideração que nesses bytes além da mensagem, vão qualquer protocolo adicional que indique o tipo de dado do pacote. O Bluetooth 5 resolve esse problema, aumentando mais de oito vezes o tamanho das mensagens, passado de 31 bytes para 255 bytes (COLLOTTA *et al.*, 2018).

Na sequência o Quadro 1 traz uma breve comparação de recursos entre as versões do Bluetooth Clássico, o padrão anterior 4.x e a nova versão, o Bluetooth 5.

Quadro 1 – Comparação entre versões do Bluetooth

Recurso	**Bluetooth Clássico**	**Bluetooth 4.x**	**Bluetooth 5**
Radiofrequência (MHz)	2400 até 2483.5	2400 até 2483.5	2400 até 2483.5
Alcance (metros)	100	100	200
Taxa de dados (Mbps)	1-3	1	2
Latência (ms)	<100	<6	<3
Mensagem (bytes)	358	31	255

Fonte: A autora (2021).

2.6 DETECÇÃO DE DISPOSITIVOS MÓVEIS

A detecção de dispositivos móveis pode ocorrer por qualquer dispositivo que tenha um módulo *wireless*. Além da possibilidade de se ter o módulo avulso e projetar uma placa eletrônica para o mesmo, já existe a tecnologia na maioria dos dispositivos eletrônicos atuais, como computadores *desktop*, *notebooks*, *smartphones*, *tablets*, *headphones*, fones de ouvido, *smartwatches*, impressoras, computadores de placa única e uma infinidade de outros. É possível obter informações dos outros dispositivos que estão ao alcance do sinal de módulo *wireless*, através de softwares e aplicativos já nativos do dispositivo que listam os demais dispositivos próximos. Pode-se desenvolver uma aplicação para este fim, utilizando bibliotecas de linguagens de programação como Python, Java, C# e C++.

2.7 PROPOSTAS DE SISTEMAS DE CONTROLE DE ACESSO UTILIZANDO BLUETOOTH

Existe uma variedade de trabalhos correlacionados com este, com algumas similaridades ao que se propõem. Assim, é possível encontrar diferentes propostas utilizando a tecnologia Bluetooth e outras tecnologias sem fio. O primeiro trabalho correlacionado contempla testes com diferentes tecnologias e sensores, o segundo foca a proposta no *hardware* de placas com módulo Bluetooh HC-05, e o terceiro foca mais no desenvolvimento *mobile* de um aplicativo. Tendo todos em comum o controle de acesso, mas que diferem nos seus desenvolvimentos, na forma de armazenamento de dados e na sua metodologia. Esta sessão apresenta uma breve apresentação desses trabalhos.

2.7.1 Desenvolvimento de um Sistema de Controle de Acesso com Armazenamento em Nuvem

No Trabalho de Conclusão de Curso (TCC) intitulado Desenvolvimento de um Sistema de Controle de Acesso com Armazenamento em Nuvem, os autores propõe o desenvolvimento de um sistema de controle de acesso, utilizando diferentes recursos de acesso por meio da aplicação das tecnologias RFID, senhas numéricas, interface com aplicativo via Bluetooth e conectividade com uma base de dados em nuvem, por meio dos conceitos de IoT, com objetivo de melhorar a segurança, eficácia e a agilidade do acesso. O trabalho foi idealizado devido ao deficiente meio de acesso aos laboratórios do bloco V da Universidade Tecnológica Federal do Paraná, onde os autores cursaram sua graduação (MANENTE; CRESPO, 2019).

2.7.2 Protótipo de Sistema para Monitoramento de Presença

No TCC intitulado Protótipo de Sistema para Monitoramento de Presença, o autor propõe um sistema automático para detecção de dispositivos visando identificar seu portador e

registrar o acesso, objetivando o monitoramento de presença dentro de locais controlado em ambiente empresarial. O protótipo utiliza o módulo Bluetooth HC-05 e apresenta toda a arquitetura estrutural montada no local de trabalho do autor. O trabalho também apresenta indicadores como tempo de presença em ambientes insalubres, possibilitando o desenvolvimento de uma ferramenta de análise comportamental em futuros trabalhos (MELO, 2018).

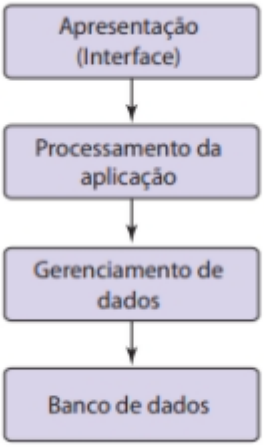
2.7.3 Sistema de Controle Baseado em Localização Interna Utilizando Dispositivos Bluetooth para Sistemas IoT

No artigo intitulado *An Indoor Location-Based Control System Using Bluetooth Beacons for IoT Systems* os autores propões um sistema que estima a localização de um dispositivo dentro de um ambiente. O sistema utiliza um servidor de localização, um cliente para prestação de serviços e um aplicativo com tecnologia de posicionamento utilizando o Bluetooth 4.0. O servidor de localização controla o acesso de dispositivos como *smartphones* com a tecnologia Bluetooth para determinar a sua localização dentro de um espaço especificado. Para o cliente foi implementado um aplicativo desenvolvido com Java JRE 1.8 utilizando Java Swing (HUH; SEO, 2017).

3 DESENVOLVIMENTO WEB

O desenvolvimento web possibilita o acesso a um sistema de forma segura e remota através de um navegador web. Para ser considerada uma aplicação consistente, levando em conta as boas práticas de desenvolvimento de software, o desenvolvimento do sistema é dividido em quatro partes. Assim, alterações, atualizações e a manutenção do sistema podem ocorrer sem prejudicar as demais partes, sendo subdividido em camadas conforme mostra a Figura 8 (MILETTO; BERTAGNOLLI, 2014).

Figura 8 – Organização da aplicação em camadas



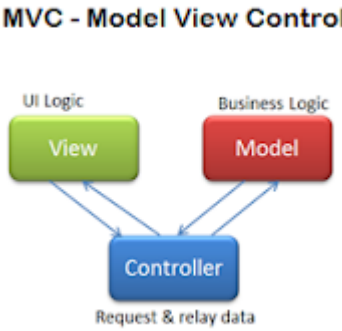
Fonte: Retirado de Evandro Manara Miletto (MILETTO; BERTAGNOLLI, 2014)

Em uma explicação breve sobre a organização da aplicação, na camada de apresentação está a parte responsável pela interação com o usuário, chamada de interface. A primeira camada tem interação direta com a camada de processamento, como o próprio nome já indica, faz os processos devidos conforme as ações do usuário. A camada de processamento não interage com o banco dados, logo, ela é responsável pela comunicação com a camada de gerenciamento de dados caso o usuário faça qualquer ação vinculada ao banco. Já a camada de gerenciamento de dados é a responsável por qualquer interação direta com o banco de dados, mediando a interação entre o processamento e o banco. E por fim, na última camada, o próprio banco, onde todos os dados da aplicação ficam armazenados.

Algumas arquiteturas de software consideram a camada de apresentação e processamento como uma só, sendo a camada de visão, como é o caso de uma das arquiteturas mais conhecidas, o Padrão *Model–View–Controller* (MVC), é um padrão de projeto de software e tem a comunicação entre as camadas como mostra a Figura 9. Estes padrões de arquitetura tem o intuito de separar partes distintas do projeto, evitando assim a dependência entre elas e deixando as regras de negócio isoladas. Existem diversos padrões de arquitetura e é necessário

avaliar a capacidade de manutenção e desempenho de cada padrão para decidir qual o mais indicado para a proposta da aplicação.

Figura 9 – Padrão *Model–View–Controller*

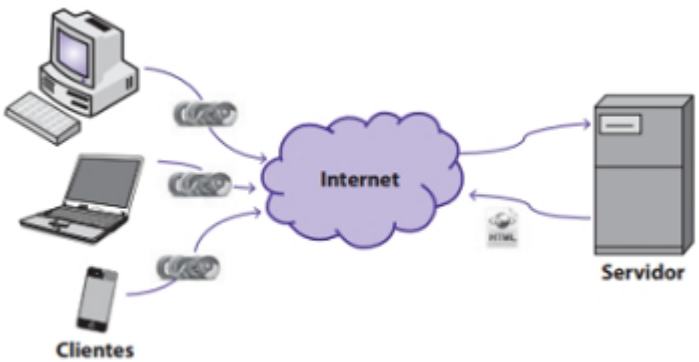


Fonte: Macoratti.net [1]

Alguns dos padrões de arquitetura mais utilizados além do MVC, são o Padrão *Model–View–Presenter* (MVP) e o Padrão *Model–View–ViewModel* (MVVM). Sendo o "M" e o "V" em comum, indicando modelo e visão e a terceira letra é de controlador, apresentação e visão-modelo respectivamente. O MVC é mais indicado para aplicações simples de poucas telas, o MVP apresenta maior testabilidade e o MVVM permite uma resposta mais rápida às alterações de projeto e facilita a utilização de testes unitários (ZENKER *et al.*, 2019).

A internet utiliza o modelo cliente-servidor conforme mostra a Figura 10. Neste modelo qualquer cliente utilizando qualquer tipo de computador de qualquer parte do mundo, consegue fazer a requisição de uma página através de um "endereço virtual", a URL, que vai na barra de endereço do navegador. O servidor por sua vez, recebe a requisição, procura o documento conforme o caminho informado e envia a resposta com o documento para o cliente (MILETTO; BERTAGNOLLI, 2014).

Figura 10 – Estrutura cliente-servidor

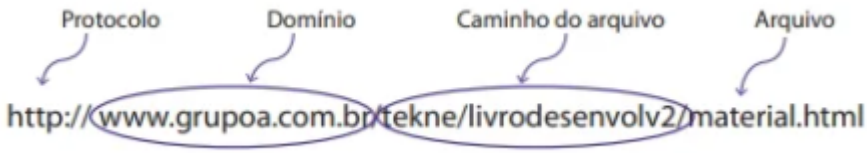


Fonte: Retirado de Evandro Manara Miletto (MILETTO, 2014)

[1] Disponível em: <http://www.macoratti.net/15/12/net$_m vvm1.htm/ > .Acessoem29set.$2021

O *HyperText Transfer Protocol* (HTTP) é o protocolo utilizado para a transferência de documentos na internet. Após receber o arquivo, o navegador é responsável por interpretá-lo e exibi-lo ao usuário. A URL é composta pelo protocolo HTTP seguido do domínio do site, após o caminho do arquivo e por último o nome do mesmo, conforme é mostrado na Figura 11 (MILETTO; BERTAGNOLLI, 2014).

Figura 11 – Estrutura da URL



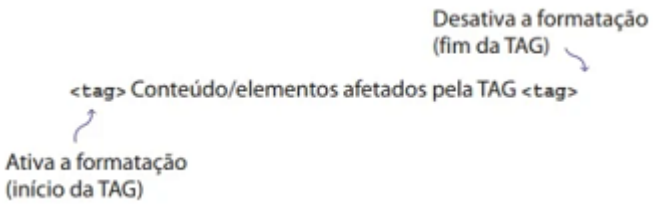
Fonte: Retirado de Evandro Manara Miletto (MILETTO, 2014)

A página HTML retornada pelo servidor será processada pelo navegador. Neste processamento podem ser identificados links para outros recursos que a página utiliza, como por exemplo para um arquivo JavaScritpt e para um arquivo CSS, e então o navegador enviará novas requisições HTTP para pegar esses arquivos. As tecnologias que rodam do lado do cliente são carregadas e executadas apenas utilizando o navegador Web, logo, não é necessário fazer mais requisições ao servidor além da requisição inicial pedindo as páginas e arquivos utilizados no *front-end* (MILETTO; BERTAGNOLLI, 2014).

3.1 HTML

O HTML [2]é uma estrutura formada por um conjunto de TAGs, que serve de base para a criação de uma página que é exibida em um navegador web, sendo ele parte da camada de apresentação dentro da estrutura da aplicação. A TAG é uma estrutura criada pelo HTML, que afeta e posiciona os elementos que estão entre sua abertura e fechamento conforme ilustra a Figura 12, uma TAG sempre é envolta por um sinal de “menor que” (<) e “maior que” (>) como também pode ser observado na Figura 12 (MILETTO; BERTAGNOLLI, 2014).

Figura 12 – Regra de utilização de TAGs



Fonte: Retirado de Evandro Manara Miletto (MILETTO, 2014)

As TAGs são utilizadas como delimitadoras de estilo e/ou conteúdo. A TAG tem efeito somente sobre o conteúdo que está entre sua abertura e fechamento. A linguagem possui deze-

[2] Disponível em: <https://en.wikipedia.org/wiki/HTML>. Acesso em 27 nov. 2021

nas de TAGs, algumas das utilizações das principais TAGs são indicar um conteúdo de documento HTML, fazer configurações sem que apareçam para o usuário, indicar o título que vai na aba do navegador e uma série de outras. Na sequência o Quadro 2 traz algumas dessas TAGs mais utilizadas e uma breve explicação da sua função.

Quadro 2 – Principais TAGs HTML

TAG	**Descrição**
<!DOCTYPE>	Marca que tipo de documento está sendo escrito
<html> </html>	Envolve todo o documento html
<head> </head>	É a "cabeça" do documento. O que estiver aqui, não será visível para os visitantes da página
<title> </title>	É o título do documento, que ficará visível na aba do navegador. Ele é diferente do título da página
<body> </body>	É o corpo do documento. Aqui é onde ficará toda a estrutura da página, como títulos, subtítulos, imagens, links, etc
<h1> </h1> até </h6>	São marcações de título de uma página, por ordem de relevância. H1 é o título principal da página
	É um paragrafo, onde fica o conteúdo da página.
 	Uma quebra de linha, que não precisa de tag de fechamento
<!– –> <!– –>	É a sintaxe de um comentário no código, que pode ser utilizado para explicar o código. O texto dentro dessa tag não será visível para o usuário

Fonte: Adaptado de reprograma[3]

A linguagem possibilita a exibição de uma série de recursos visuais de hipermídia como imagens, links, e vídeos, além de textos (MILETTO; BERTAGNOLLI, 2014). Atualmente a linguagem HTML está na sua quinta versão, chamada de HTML5, especificação mantida em https://whatwg.org/ e possui o logomarca mostrada na Figura 13.

Figura 13 – Logomarca do HTML5

Fonte: dev por aí [4]

O HTML é uma espécie de esqueleto da página, onde consta toda a estrutura e elementos que a página terá. Anteriormente se usava alguma estilização da página com as próprias

[3] Disponível em: <https://medium.com/reprogramabr/primeira-página-na-web-hello-word-6913a405f50c>. Acesso em 13 oct. 2021

[4] Disponível em: <https://devporai.com.br/10-recursos-do-html5-que-voce-pode-nao-estar-usando/>. Acesso em 12 oct. 2021

TAGs do HTML, mas atualmente tem apenas a função de delimitação e estruturação dos elementos. Para a estilização da página é utilizado o CSS e para programar os comportamentos ao se interagir com a página, são utilizadas linguagens de programação sendo a mais popular JavaScript (MILETTO; BERTAGNOLLI, 2014).

Existem as tecnologias utilizadas do lado do servidor, e as do lado do cliente. O combo convencional do lado do cliente é o HTML, CSS e JavaScript, cada uma com a sua função. Existem outras linguagens de programação que podem ser utilizadas para gerenciar o comportamento da página, e também *frameworks* que disponibilizam uma série de bibliotecas e funções que auxiliam na programação e interação com o HTML, visando simplificar e otimizar o desenvolvimento.

A área da tecnologia é muito fluída e constantemente surgem novos *frameworks*, linguagens caem em desuso e outras ganham o protagonismo, então se torna delicada a listagem das melhores ou mais utilizadas tecnologias no mercado. Mas dentre as opções, o JavaScript atualmente é sem dúvida uma das mais populares, sendo utilizada por páginas web de grandes empresas mundiais como no site da Google, Youtube, Facebook, Amazon e uma série de outras.

3.2 CSS

O CSS (*Cascading Style Sheets*), ou folhas de estilo em cascata em português, desenvolvido em 1996 pela *World Wide Web Consortium* (W3C) passa a ser responsável pela aparência da página, deixando assim o HTML apenas com a função de estruturar a mesma. Alguns navegadores web como o Google Chrome e o Firefox, permitem visualizar e inspecionar os arquivos HTML e CSS da página web que se está, pressionando a tecla F12, ou clicando com o botão direito na página e selecionando a opção “Inspecionar” e será aberta uma nova janela com as ferramentas de desenvolvimento, constando a esquerda o HTML e a direita a estilização do elemento selecionado (MILETTO; BERTAGNOLLI, 2014).

Com o CSS é possível criar classes com determinada estilização e aplicar esta classe a inúmeros elementos HTML distintos. Também oferece a opção de atrelar uma formatação a um determinado elemento HTML, como por exemplo ao elemento “<p>” que marca um parágrafo, conforme ilustra a Figura 14, assim toda vez que a TAG “<p>” for utilizada na página, sempre terá aquela formatação que foi aderida ao elemento. E uma terceira opção é dar um identificador ao elemento no HTML, e atrelar uma formatação a esse identificador. O CSS também simplifica a manutenção da aparência da página, pela possibilidade de utilizar as mesmas características aplicadas a diversos documentos HTML. Pode-se colocar uma cor padrão para o título de todas as páginas web de um site, e ao querer mudar esta cor futuramente, a alteração será aplicada a todos os documentos HTML (ALVES, 2014b).

Atualmente a linguagem está na sua terceira versão, que possui uma série de inovações comparada a sua versão anterior, se destacando por suas animações 2D e 3D, com diversos

Figura 14 – Exemplo de formatação por elemento HTML

```
# styles.css
Frontend > css > # styles.css > ...
1   p{
2       color: darkorange;
3       font-size: 50px;
4       margin-left: 50px;
5       font-family: Georgia, 'Times New Roman', Times, serif;
6       align-content: center;
7   }
8
```

Fonte: A autora

efeitos de transição, rotação e movimento. O CSS possui basicamente a mesma identidade visual que o HTML, assim, fixando a ligação complementar entre as duas tecnologias. Seguindo o mesmo padrão do HTML, a logomarca fixa a sua versão junto a sua logomarca, conforme mostra a Figura 15.

Figura 15 – Logomarca do CSS

Fonte: Tutoriart [5]

O CSS está sempre atrelado a uma página HTML, e pode ser utilizado de três formas. Pode-se aplicar a estilização *Inline*, que aplica as regras de estilo dentro da própria TAG HTML, dessa maneira é necessário colocar ou repetir o estilo em cada uma das TAGs, tornando a manutenção extremamente improdutiva e deixando espaço para falhas. O CSS também pode ser utilizado inserido no arquivo HTML, fazendo as formatações dentro da TAG "<style>" como na Figura 16, assim, toda a estilização fica centralizada em um só local. E a terceira maneira é a aplicação possuir o CSS num arquivo externo e apenas referenciar que o HTML utiliza deter-

[5] Disponível em: <https://www.tutoriart.com.br/css-o-segredo-da-beleza-das-paginas-web/>. Acesso em 12 nov. 2021

minado arquivo CSS para formatação, desta forma o mesmo arquivo CSS pode ser usado por várias páginas HTML.

Figura 16 – Exemplo de CSS no corpo do HTML

```
<!DOCTYPE html>
<html>
    <head>
        <title></title>
        <style type="text/css">
            .titulo{
                font-size: 3rem; /*tamanho da fonte do titulo*/
                color: #333; /*esta é a cor do texto*/
                text-align: center; /*o alinhamento do texto que carregar esta classe*/
            }
            #botao-bonito{
                color:white;/*cor da fonte*/
                padding-top: 15px;/*espaço superior entre o texto e o elemento*/
                padding-bottom: 15px;/*espaço inferior entre o texto e o elemento*/
                padding-right: 35px;/*espaço a direita entre o texto e o elemento*/
                padding-left: 35px;/*espaço a esquerda entre o texto e o elemento*/
                background-color:green;/*cor verde para o fundo do elemento, nesse caso o botão*/
            }
        </style>
    </head>
    <body>

        <h1 class="titulo fonte-grande sublinhado">Melhor site</h1>
        <a id="botao-bonito" href="link-para-alguma-pagina">clique nesse botão</a>
```

Fonte: HOSTINGER [6]

O CSS possibilita fazer formatações como de fonte e cor do texto, cor do fundo, tamanho dos elementos, margem, deslocamento, bordas, alinhamentos, sombras, animações, e muitos outros recursos vinculados a aparência da página Web (MILETTO; BERTAGNOLLI, 2014). Pode-se adicionar também *frameworks* que disponibilizam bibliotecas com uma variedade de componentes prontos e já estilizados. O CSS também auxilia na questão de responsividade das páginas, podendo fazer alinhamentos conforme porcentagens e tamanhos de tela.

A seguir a Figura 17 mostra uma comparação de página somente com o HTML à esquerda e de uma página com o CSS atrelado ao HTML à direita. Onde é possível perceber que ambas tem os mesmos elementos, mas na página da esquerda os elementos apenas estão dispostos, sem nenhum estilização, já na página da direita os elementos estão posicionados, dimensionados e estilizados.

3.3 JAVASCRIPT

O desenvolvimento Web é realizado por um conjunto de linguagens de programação, sendo o JavaScript uma delas. o JavaScript surgiu em 1995, é uma linguagem interpretada que é executada diretamente no navegador Web, do lado do cliente. Ele é responsável pelo comportamento e processamento da página. Podendo através dele submeter formulários, preencher

[6] Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-css-guia-basico-de-css>. Acesso em 13 oct. 2021

Figura 17 – Comparação de página sem e com CSS

Fonte: A autora

e habilitar campos e controlar uma série de comportamentos e processos (MILETTO; BERTAGNOLLI, 2014).

Assim como o CSS, é possível vincular o JavaScript a página HTML de forma interna e externa. De forma interna é vinculado através da inclusão da TAG “<script>” dentro do próprio arquivo HTML, assim, dentro da TAG é feita a programação em JavaScript conforme ilustra o exemplo da Figura 18 a seguir.

Figura 18 – Exemplo de utilização de programação em JavaScript dentro do arquivo HTML

```
<html>
  <head>
     <title>Primeira página com JS</title>
  </head>
  <body>
   Texto no HTML.<br/>
   <script type="text/javascript">
     document.write("Parte gerada via JavaScript");
   </script>
  </body>
</html>
```

Fonte: Retirado de Evandro Manara Miletto (MILETTO, 2014)

Porém, a maneira mais utilizada para inclusão do JavaScript no HTML é de forma externa, para isto também se utiliza a TAG “<script>” mas ao invés de colocar diretamente a implementação dentro da TAG, é apenas indicado o caminho onde o arquivo JavaScript vinculado a página se encontra. Essa forma é mais utilizada pela clareza e organização, deixando no HTML apenas a estrutura da página. O arquivo JavaScript é indicado pela extensão “js”, mas é um arquivo de texto comum, podendo ser editado em qualquer editor de texto. A Figura 19

mostra o mesmo exemplo anterior mas agora com a indicação de um arquivo JavaScript externo na TAG “<script>” gerando a página mostrada à direita.

Figura 19 – Exemplo de utilização de programação em JavaScript em arquivo externo ao HTML

Fonte: Retirado de Evandro Manara Miletto (MILETTO, 2014)

O JavaScript foi desenvolvida por Brendam Eich da Nestcape e é uma das linguagens mais populares no desenvolvimento Web, dentre suas principais características se tem a tipagem dinâmica, é orientada a objetos, assim como o Java e o C#, mas também é orientada a eventos. Eventos como a movimentação do mouse, o clique, arrastar e soltar elementos e diversas outras. Também realiza validações em tempo de execução, é baseada em *European Computer Manufactorers Association (ECMA) Script* [7] e faz parte da camada de processamento dentro da estrutura da aplicação (ALVES, 2015).

3.4 PYTHON

Python [8]é uma linguagem de programação que foi criada no final dos anos 1980, idealizada por Guido Van Rossum, que trabalhava, no *Centrum Wiskunde Informatica* (CWI). É uma linguagem de alto nível, sendo que quanto mais próxima da linguagem natural é mais alto nível, e quanto mais próximo da linguagem de máquina, mais baixo nível. Também é uma linguagem fortemente tipada e que possibilita escrever *scripts* com a mesma. Sendo muito popular por sua facilidade e eficiência, é possível realizar muitas funções com poucas linhas de código e também possui uma curva de aprendizado menor que outras linguagens mais convencionais como Java e C#. Por essa facilidade se tornou uma boa escolha como primeira linguagem de programação a se aprender o quê também contribui para sua popularidade (MACIEL, 2020). Python tem uma identidade visual forte, sua logomarca presente na Figura 20 é rapidamente reconhecido, sendo composta por duas cobras que remetem à antigos desenhos maias e à direita, sua nomenclatura.

[7] Disponível em: <https://www.ecma-international.org/publications-and-standards/standards/?order=last-change>. Acesso em 12 nov. 2021

[8] Disponível em: <https://www.python.org/>. Acesso em 12 nov. 2021

[9] Disponível em: <https://olhardigital.com.br/2020/08/18/noticias/grupos-usuarios-de-python-criam-consorcio-para-padronizar-linguagem/>. Acesso em 12 nov. 2021

Figura 20 – Logotipo da linguagem de programação Python

Fonte: Olhar Digital [9]

Python possui hoje duas versões que coexistem a 2.X e a 3.X até o momento. As duas versões são tão diferentes que apresentam questões de incompatibilidade entre elas. A versão 2.X ainda se dá pela manutenção de projetos legados, enquanto a 3.X contém as inovações da linguagem (BANIN, 2018).

O Python é uma linguagem para desenvolvimento geralmente de *back-end*, ou seja, do lado do servidor. Lidando com os serviços e camadas relacionadas aos dados, sendo ele parte da camada de gerenciamento de dados dentro da estrutura da aplicação. Mas é possível utilizar também a linguagem do lado do cliente, utilizando auxílio de *frameworks* web como Django e Flask.

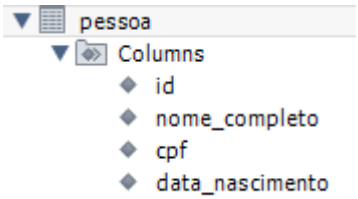
3.5 BANCO DE DADOS

Desde os primórdios o homem tem a necessidade de armazenar informações para serem preservadas e utilizadas futuramente. Informação por definição é qualquer fato ou conhecimento do mundo real que pode (ou não) ser armazenado, o que leva ao dado, que é a representação da informação e pode ser armazenado. Esse armazenamento pode ser feito desde em papel até ao disco rígido do computador (ALVES, 2014a). O armazenamento desses dados gera um banco de dados, que nada mais é que um conjunto de dados relacionados. A partir de um armazenamento de dados surge a necessidade de seu gerenciamento, então são criados gerenciadores, sendo um Sistema Gerenciador de Banco de Dados (SGBD) uma coleção de ferramentas e programas que permitem criar e manipular bancos de dados (ALVES, 2014a).

Os bancos de dados são divididos entre relacionais e não relacionais. O banco de dados relacional armazena seus dados em tabelas, enquanto o banco não relacional tem formas diversas de armazenamento podendo ser mais complexas e distintas do que em tabelas. Os bancos relacionais são os mais utilizados, e a tabela pode ser entendida como um objeto e seus atributos como colunas, assim, uma tabela nomeada "Pessoa", pode ter atributos como nome, CPF, e Data de Nascimento, além de um atributo “id” com o propósito de ter um código único para

identificação do registro da tabela, como exemplifica a Figura 21.

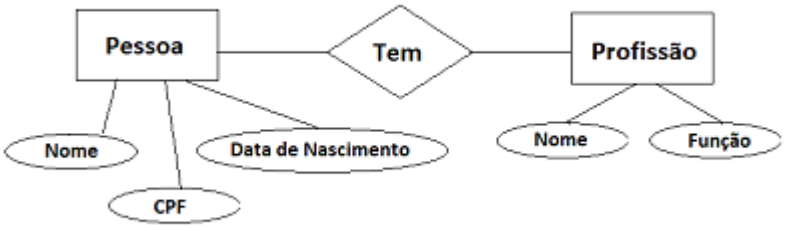
Figura 21 – Exemplo de tabela "Pessoa"



Fonte: A autora

Uma tabela pode ter relacionamento com outras, e esse relacionamento se dá a partir do atributo “id” da tabela relacionada. Uma pessoa pode ter uma profissão, então pode ser adicionada a coluna “profissão_id” a tabela “pessoa” com o seu respectivo “id”. A estrutura de um banco de dados pode ser apresentada através de outras ferramentas para ajudar na sua visualização num todo. Uma dessas ferramentas é um modelo conceitual, que é mais abstrato do que um modelo de tabelas. Um modelo conceitual muito utilizado é o modelo *Entidade-Relacionamento* (ER), representando entidades como retângulos, seus atributos como elipses e relacionamentos como losangos, conforme a Figura 22 ilustra (MACHADO, 2020).

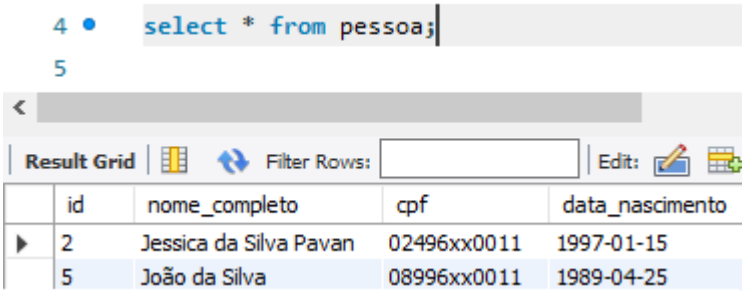
Figura 22 – Exemplo de diagrama de entidades



Fonte: A autora

Para fazer manipulações com os dados nos bancos de dados foi criada a *Structured Query Language* (SQL), permitindo manipulações como consultas, inserções, deleções, criação de estruturas e uma série de outras. A SQL, Linguagem de Consulta Estruturada em português, foi padronizada em 1986 pela *American National Standards Institute* (ANSI), com o objetivo de ser utilizadas nas interações com diferentes Sistemas Gerenciadores de Banco de Dados (SGBDs) relacionais (MILETTO; BERTAGNOLLI, 2014). Existe um grande número de SGBDs sendo os mais populares: Oracle, DB2, MySQL, SQL Server, e PostgreSQL. Eles possuem pequenas diferenças de sintaxe entre si, sendo necessárias pequenas adaptações de *script* de um SGBD para outro, mas seguem um mesmo padrão. Assim, em qualquer um desses SGBDs é possível por exemplo consultar os dados de uma tabela, utilizando o mesmo *script* “select * from tabela” conforme mostra a Figura 23, que traz dados inseridos na tabela pessoa Figura 23 já mostrada anteriormente.

Figura 23 – Exemplo de consulta com a tabela "Pessoa"



Fonte: A autora

A SQL permite criar *scripts* dos mais simples aos mais complexos, possibilitando qualquer interação necessária com o banco de dados. Todos o SGBDs oferecem uma interface onde são executados *scripts* escritos em SQL para tal interação.

4 PROPOSTA DE SISTEMA DE GERENCIAMENTO DE ACESSO

Devido a necessidade de controlar quem adentra num ambiente, por exemplo por causa de questões de segurança de indivíduos e bens que se encontram no local, foi definido neste trabalho um protótipo de software que implementa um método de controle através de dispositivos que possuem a tecnologia Bluetooth. Futuramente essa proposta poderá ser combinada com uma automação para a liberação de uma barreira física de forma automática, tendo em vista a demanda de agilidade na liberação do acesso em ambientes com um grande fluxo de pessoas.

A proposta desse trabalho é definir e implementar um protótipo de software que tem o objetivo de liberar a passagem de indivíduos que se aproximam do local de acesso ao ambiente, e oferecer suporte ao gerenciamento deste acesso. O método para a execução da proposta consiste em realizar cadastros e validações de dispositivos vinculados a pessoas com perfis de acesso, através dos seus respectivos endereços Bluetooth e informações complementares armazenadas em banco de dados.

4.1 BACK-END DA APLICAÇÃO

A aplicação será simulada em um notebook com sistema operacional Windows. O *back-end* da aplicação, onde consta a camada de gerenciamento de dados, será feito com a linguagem Python na sua versão 3.x.

4.1.1 Configuração do ambiente para a linguagem Python

Para ser possível executar os códigos em Python é preciso fazer a sua instalação previamente. Para isto, é necessário entrar no site oficial da linguagem https://www.python.org/downloads/, onde no topo da página já sugere a última versão disponibilizada para Windows para o download, conforme mostra a Figura 24.

Figura 24 – Download da linguagem Python

Fonte: Python [1]

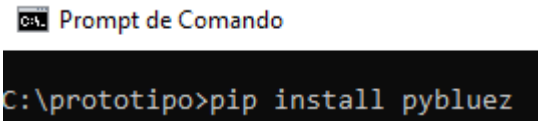
[1] Disponível em: https://www.python.org/downloads/>. Acesso em 17 nov. 2021

Após executar a instalação padrão, já será possível rodar os programas com a linguagem. Para auxiliar no desenvolvimento será utilizada uma *Integrated Development Environment* (IDE), foi escolhido o Visual Studio Code [2] como ambiente de programação tanto para o *back-end* quanto para o *fron-end*, para utilizá-lo é necessário fazer sua instalação.

4.1.2 Detecção de Dispositivos

A detecção dos dispositivos com o Bluetooth ativo será feita com a linguagem Python através da utilização da *Application Programming Interface* (API) PyBluez. Esta API é um módulo de extensão Bluetooth, e para utilizá-la é necessário fazer sua instalação, que se dá pelo comando "pip install pyblues" após já ter o Python instalado na máquina, conforme exemplifica a Figura 25.

Figura 25 – Instalação API PyBluez



Fonte: A autora

Esta API disponibiliza um conjunto de classes e funções que permitem a utilização de recursos da tecnologia Bluetooth nativos do sistema. Após importar a API, será utilizada a função "discovery_devices" a qual retorna uma lista dos dispositivos em formato *string*. Nesta lista constam os dispositivos com Bluetooth ativo ao alcance da máquina utilizada no protótipo, um notebook Aspire 5 A515-54-55L0. A máquina possui a tecnologia Bluetooth 5, que possui um alcance de até 40 metros em ambientes internos, reduzindo esse alcance conforme a existência de barreiras físicas, como paredes internas (COLLOTTA *et al.*, 2018).

A validação de acesso se dará após consultar se o dispositivo detectado já está armazenado no banco de dados e também verificando o perfil da pessoa vinculada ao mesmo, permitindo a passagem caso o perfil tenha permissão de acesso ao local. No caso do primeiro acesso com um determinado dispositivo é necessário ser feito o seu cadastro. Essa validação sucederá por meio de uma rotina que detecta dispositivos com o Bluetooth ativo, que será executada continuamente de forma assíncrona.

4.1.3 Gerenciamento de Dados

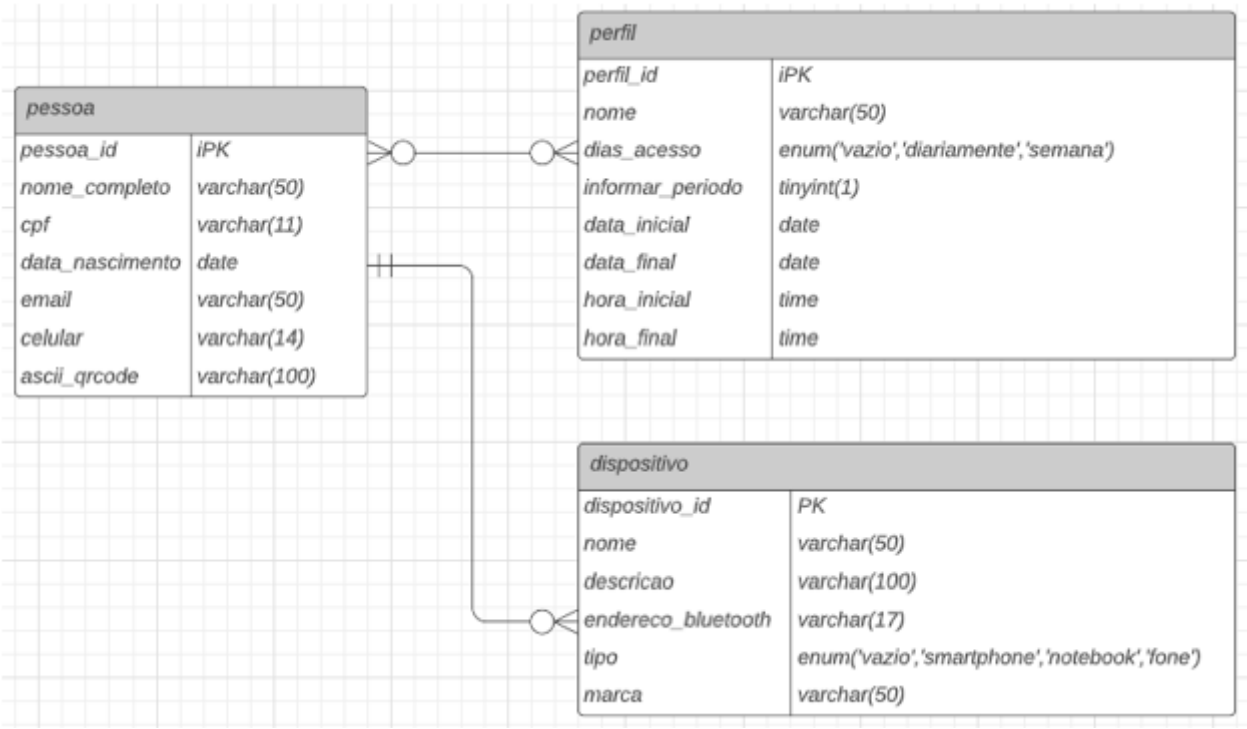
Para o funcionamento essencial do protótipo se identificou a necessidade de 3 cadastros, um histórico de acesso e uma tabela temporária para controle de dispositivos detectados. Para o armazenamento dos dados cadastrados serão criadas quatro tabelas, e mais 2 tabelas utilizadas para controle, resultando em 6 no total, as quais serão detalhadas na Seção 4.1.3.1. O

[2] Disponível em: https://code.visualstudio.com/?wt.mc$_id = DX_8 41432.Acessoem22mai.2022$

armazenamento será feito em um banco de dados relacional. Para o gerenciamento do banco foi escolhido o SGBD MySQL por ser um gerenciador gratuito e com vasto conteúdo dobre ele na internet. As interações com o banco de dados na camada de gerenciamento de dados se dará por intermédio da API Flask, sendo necessária sua instalação prévia.

O diagrama do modelo lógico do banco de dados mostrado na Figura 26 demonstra a relação e o vinculo entre os 3 cadastros, onde o tipo de relação entre eles se dá pela linha que os liga, símbolos que tem sua notação presente na Figura 27.

Figura 26 – Diagrama do modelo lógico do banco de dados

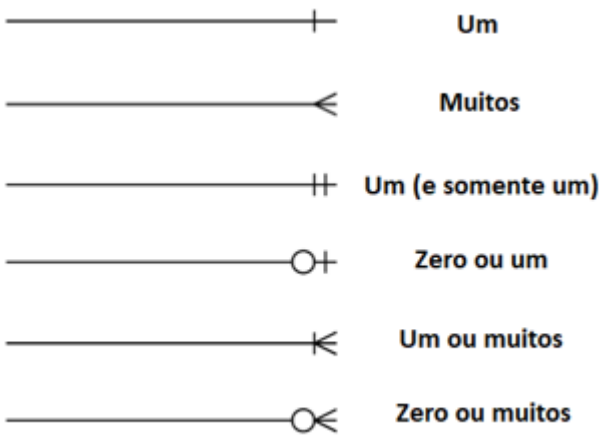


Fonte: A autora

Afim de fazer uma representação mais visual de como os cadastros estão relacionados, a Figura 28 mostra o Modelo ER desses cadastros, que também utiliza as notações presentes na Figura 27. O modelo representa a entidade pessoa que pode possuir diversos dispositivos ou nenhum, já o dispositivo obrigatoriamente pertence a uma pessoa. Para uma pessoa conseguir adentrar no ambiente controlado ela precisará de ao menos um perfil, pois o perfil é responsável por delimitar o período de acesso permitido. A pessoa pode possuir mais de um perfil atrelado a ela, ou até nenhum, apenas não conseguirá acessar o local se não o possuir. Já o mesmo perfil pode ser atrelado a quantas pessoas quanto necessário.

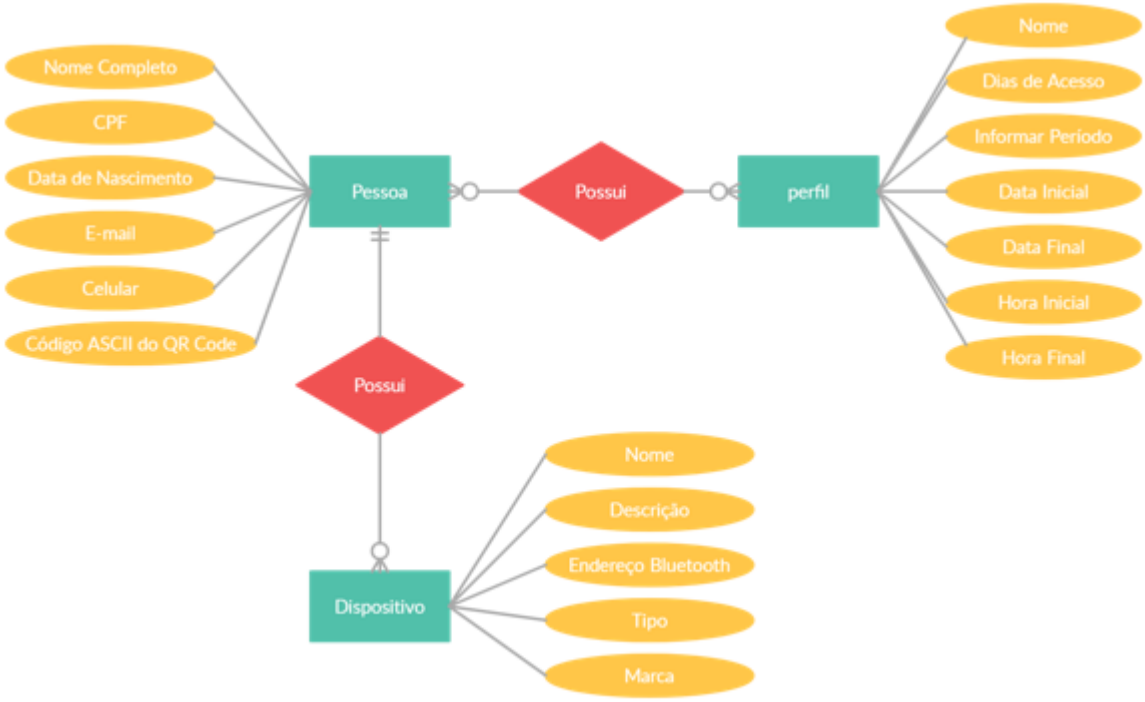
[3] Disponível em: https://www.lucidchart.com/pages/pt/simbolos-de-diagramas-entidade-relacionamento/>. Acesso em 27 abr. 2021

Figura 27 – Símbolos e notação



Fonte: Traduzido de Lucidchart [3]

Figura 28 – Modelo ER do Banco de Dados



Fonte: A autora

4.1.3.1 Estrutura de Dados

A primeira tabela se refere ao perfil de acesso, com dia/horário inicial e dia/horário final. A partir do cadastro de perfil será feita a delimitação de quando a pessoa vinculada a esse perfil tem acesso permitido. Para isto se identificou a necessidade de criação da tabela com dados que auxiliam a limitar o intervalo ou periodicidade do acesso do respectivo perfil, os campos da tabela podem ser visualizado no Quadro 3.

Quadro 3 – Tabela *perfil*

Campos	Tipo
id	int(11)
nome	varchar(50)
dias	enum('vazio','diariamente','semana')
informar_período	tinyint(1)
data_inicial	date
data_final	date
hora_inicial	time
hora_final	time

Fonte: A autora (2021).

O sistema também contará com um cadastro de pessoa, para seu armazenamento se propõe a tabela descrita a seguir, que consiste em informações para a identificação do individuo. Pode-se armazenar um grande números de dados vinculados à uma pessoa, conforme se tem interesse em tais informações. Para o propósito da aplicação, se identificou a necessidade de campos básicos para controle de acesso, e um campo que guarda o código *American Standard Code for Information Interchange* (ASCII) que será utilizada por um verificação de dois fatores, conforme detalhado nos campos da sua respectiva tabela no Quadro 4.

Quadro 4 – Tabela *pessoa*

Campos	Tipo
id	int(11)
nome_completo	varchar(50)
cpf	varchar(11)
data_nascimento	date
email	varchar(50)
celular	varchar(14)
ascii_qrcode	varchar(100)

Fonte: A autora (2021).

É permitido que uma pessoa tenha mais do que um perfil de acesso. Para isso, será criada uma tabela de relacionamento N:N, denominada *pessoa_perfil* vinculando a pessoa ao seu(s) perfil(s) através do campo id, conforme mostra o Quadro 5.

Quadro 5 – Tabela *pessoa_perfil*

Campos	Tipo
id	int(11)
pessoa_id	int(11)
perfil_id	int(11)

Fonte: A autora (2021).

Por fim o cadastro de dispositivos, que será o meio utilizado para identificar a presença dos indivíduos nos ambientes. Estes dispositivos precisam estar vinculados a uma pessoa, e além das informações descritivas do dispositivo é necessário que seja cadastrado o Endereço Blueto-oth do mesmo. O Endereço Bluetooth é um dos dados de identificação retornado pela função de

detecção de dispositivos. Os dispositivos que serão usados para simulação serão *smartphones*, notebooks e fones de ouvido. A estrutura da tabela de *dispositivo* é detalhada no Quadro 6.

Quadro 6 – Tabela *dispositivo*

Campos	**Tipo**
id	int(11)
nome	varchar(50)
descricao	varchar(100)
endereço$_b luetooth$	varchar(17)
tipo	enum('vazio','smartphone','notebook','fone')
marca	varchar(50)
pessoa_id	int(11)

Fonte: A autora (2021).

Também será mantida uma tabela temporária para manter dispositivos detectados por um determinado período de tempo, tempo que assegura a validação do código de autenticação utilizado na validação de dois fatores que será melhor detalhada na proposta de solução. Esta tabela, contida no Quadro 7, conta com dados essenciais que identificam o dispositivo que está tentando acessar o local.

Quadro 7 – Tabela *dispositivo_detectado*

Campos	**Tipo**
id	int(11)
nome	varchar(50)
endereço$_b luetooth$	varchar(17)
pessoa$_i d$	int(11)
data	datetime

Fonte: A autora (2021).

Para se consultar o histórico de quem acessou o ambiente, caso haja a necessidade de uma verificação posterior, se propôs o armazenamento dos dados de acesso. Para este armazenamento de histórico se propõe uma tabela contendo os dados mais relevantes como a pessoa, o dispositivo, a data e o horário de entrada, conforme mostrado no Quadro 8.

Quadro 8 – Tabela *historico*

Campos	**Tipo**
id	int(11)
pessoa$_i d$	int(11)
dispositivo$_i d$	int(11)
dia	date
hora	time

Fonte: A autora (2021).

4.2 FRONT-END DA APLICAÇÃO

Para o protótipo da interface da aplicação serão utilizadas as tecnologias HTML5 com CSS3 puros, sem o uso de *framewoks*, devido a proposta consistir apenas em telas para simulação. A camada de processamento da aplicação será feita utilizando JavaScript. Tanto o arquivo CSS quanto o arquivo JavaScript serão externos ao HTML. A passagem de dados entre o *front-end* e *back-end* da aplicação será em formato *JavaScript Object Notation* (JSON), através do objeto *XMLHttpRequest* em JavaScript, o qual interage com o servidor. O *front-end* também usará a IDE Visual Studio Code [4] como ferramenta de auxílio ao desenvolvimento.

4.2.1 Telas de Cadastros

Foram idealizadas 3 telas de cadastros para serem acessadas através de um navegador web, e desenvolvidos seus respectivos *wireframes* com a identidade visual proposta. Os *wireframes* também sugerem os tipos de campos para melhor usabilidade no preenchimento destes cadastros.

A primeira tela trata do cadastro de Perfil, conforme *wireframe* propõe na Figura 29. Contendo o nome do Perfil, podendo selecionar os dias de acesso padrão, permitindo ter acesso diariamente ou de segunda a sexta-feira. Caso se deseje um período arbitrário, o usuário pode optar por informar um período. E por fim os horários que este perfil terá acesso.

A segunda tela se trata do cadastro de Pessoa, com os campos básicos para identificação, podendo adicionar um ou mais perfis de acesso, e o campo de autenticação onde é necessário inserir o *token* após ler o código QR do cadastro, conforme propõe o *wireframe* da Figura 30.

[4] Disponível em: https://code.visualstudio.com/?wt.mc$_id = DX_8 41432.Acessoem22mai.2022$

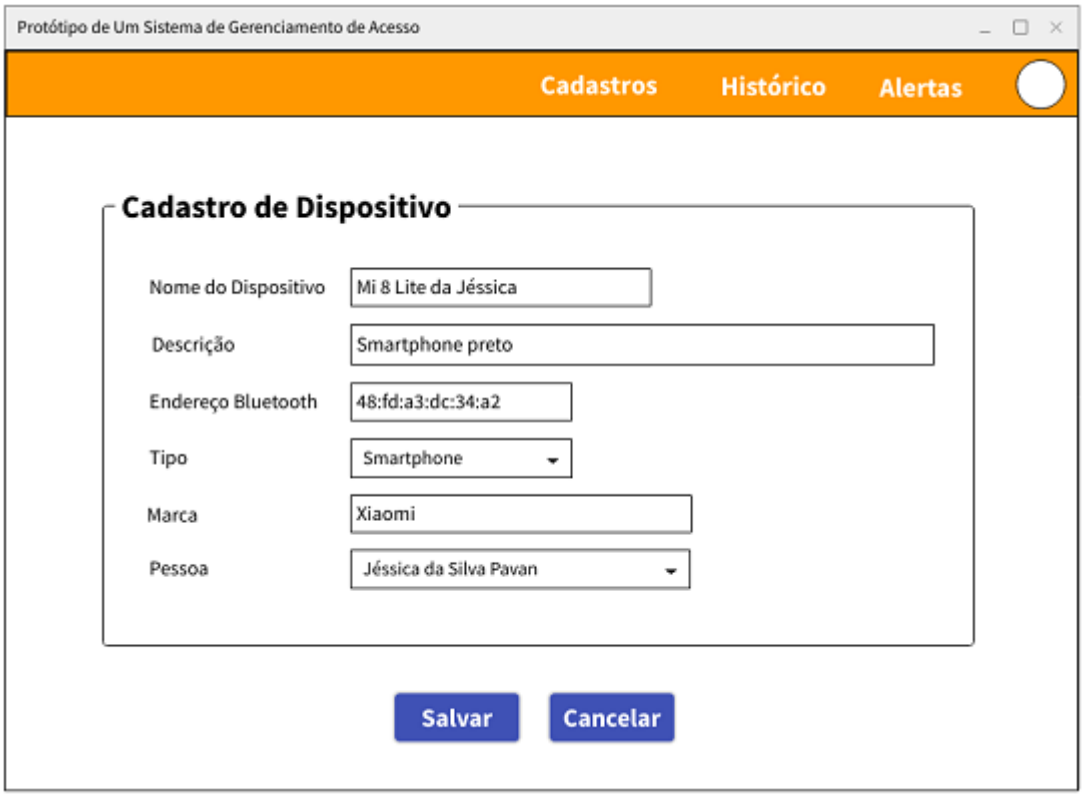
Figura 29 – Protótipo de tela de Cadastro de Perfil

Fonte: A autora

Figura 30 – Protótipo de tela de Cadastro de Pessoa

Por último o cadastro de dispositivos, que conta com as informações descritivas e relevantes sobre o mesmo, com o campo Endereço Bluetooth necessário para validação de acesso posterior>. Também é preciso vincular o dispositivo a uma pessoa já previamente cadastrada no sistema, conforme propõe o *wireframe* da Figura 31.

Figura 31 – Protótipo de tela de Cadastro de Dispositivo

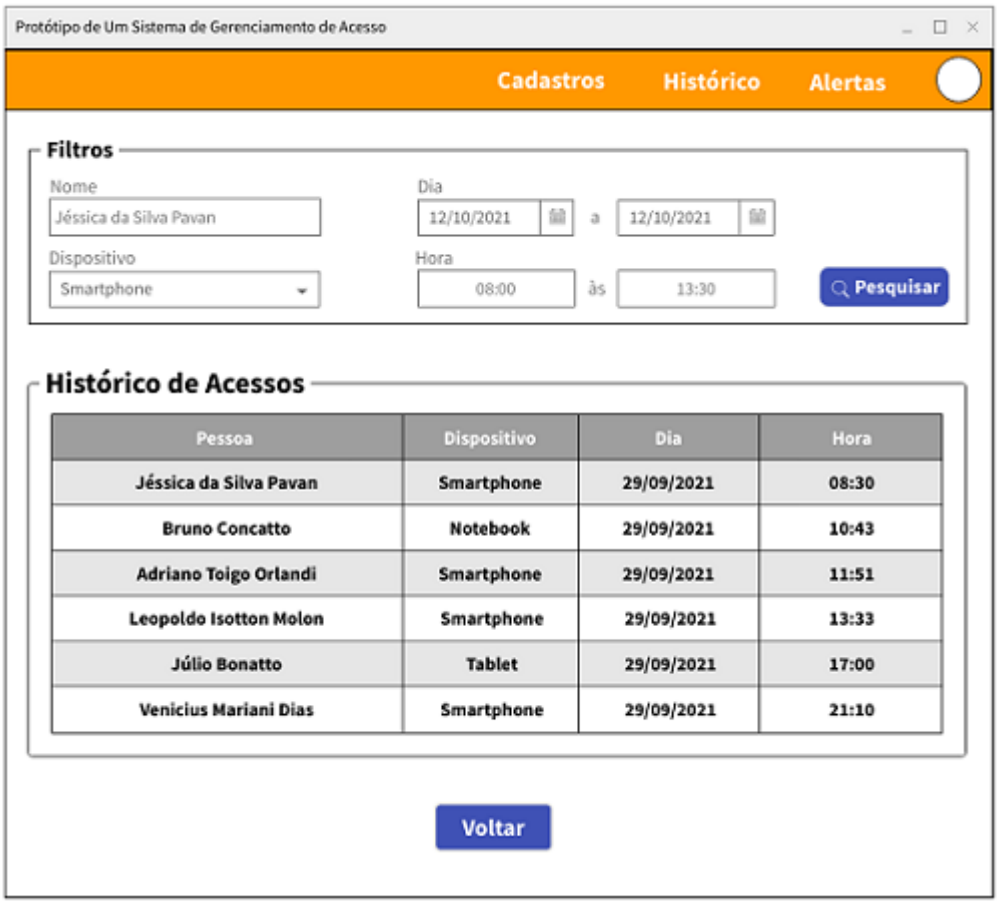


Fonte: A autora

4.2.2 Tela de Histórico de Acesso

A proposta inclui uma tela de histórico para possíveis consultas futuras em caso de haver a necessidade de identificar as pessoas que acessaram o local. A tela contará com filtros para melhor usabilidade, podendo filtrar os acessos por pessoa, período, dispositivo e horário, conforme propõe o *wireframe* da Figura 32.

Figura 32 – Protótipo de tela de Histórico

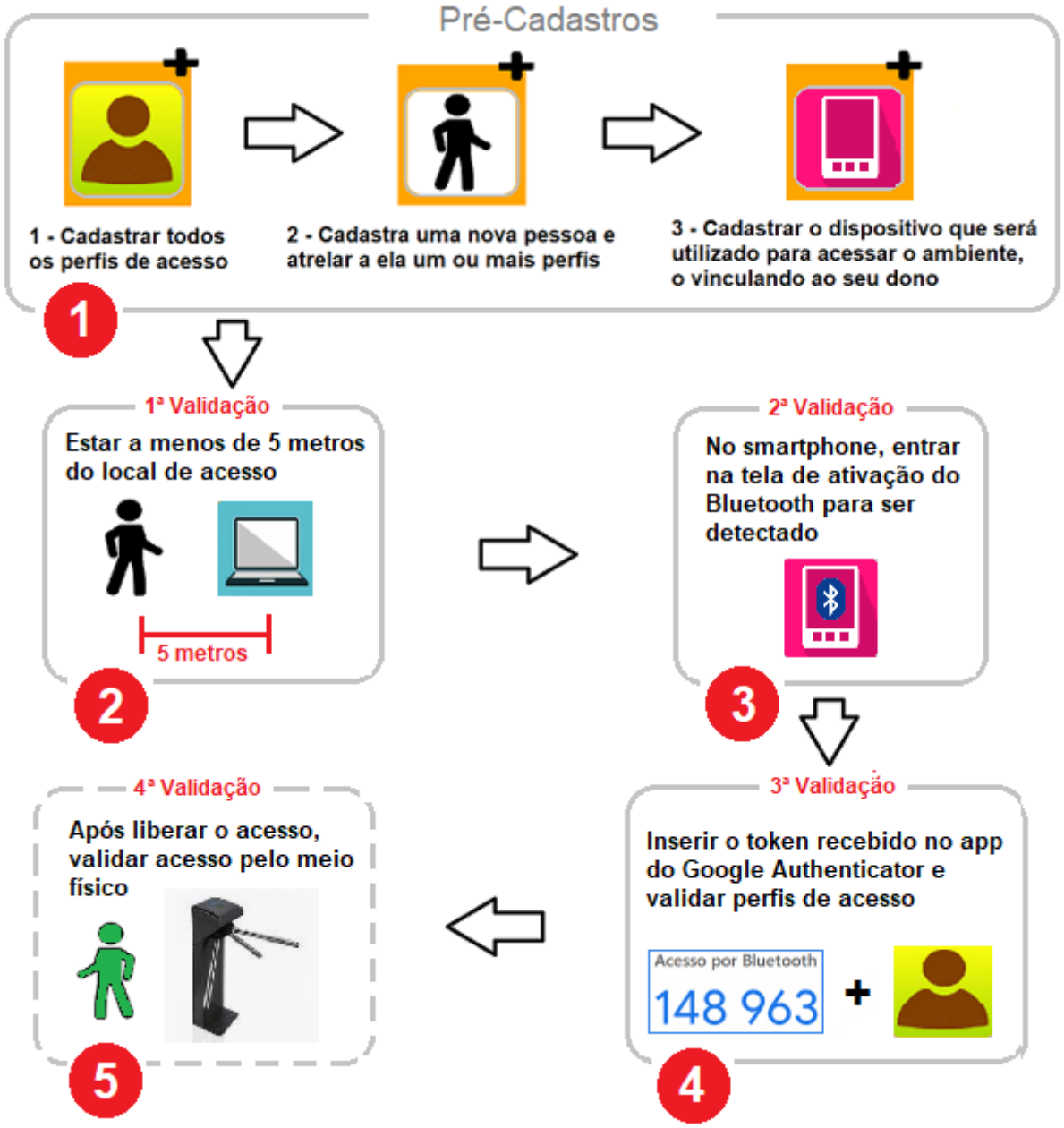


Fonte: A autora

5 FLUXO DA APLICAÇÃO

Este capítulo contempla todas as etapas necessárias para se adentrar em um ambiente controlado utilizando a proposta do trabalho. Para o funcionamento do sistema é necessário que sejam feitos os devidos cadastros antes de acessar o local, e passar por todas as validações do fluxo exibido na Figura 33, sendo todas as etapas detalhadas na sequência.

Figura 33 – Fluxo Geral do Sistema

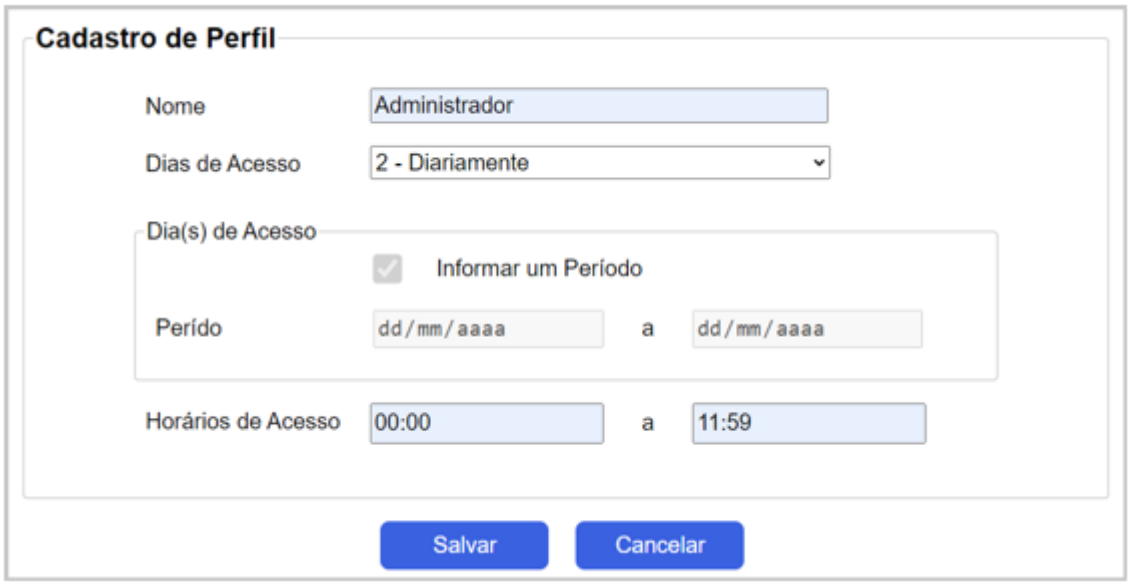


Fonte: A autora

5.1 PRÉ-CADASTROS

O primeiro passo é cadastrar todos os perfis necessários. Para exemplificar a função do cadastro, podem ser cadastrados perfis como de administrador mostrado na Figura 34, que tem acesso em qualquer horário e dia, um perfil de gerente que tenha acesso em qualquer horário mas somente de segunda a sexta-feira, e também um perfil de visitante que tenha acesso somente em um dia e intervalo de horários específicos.

Figura 34 – Exemplo de perfil Administrador com acesso total

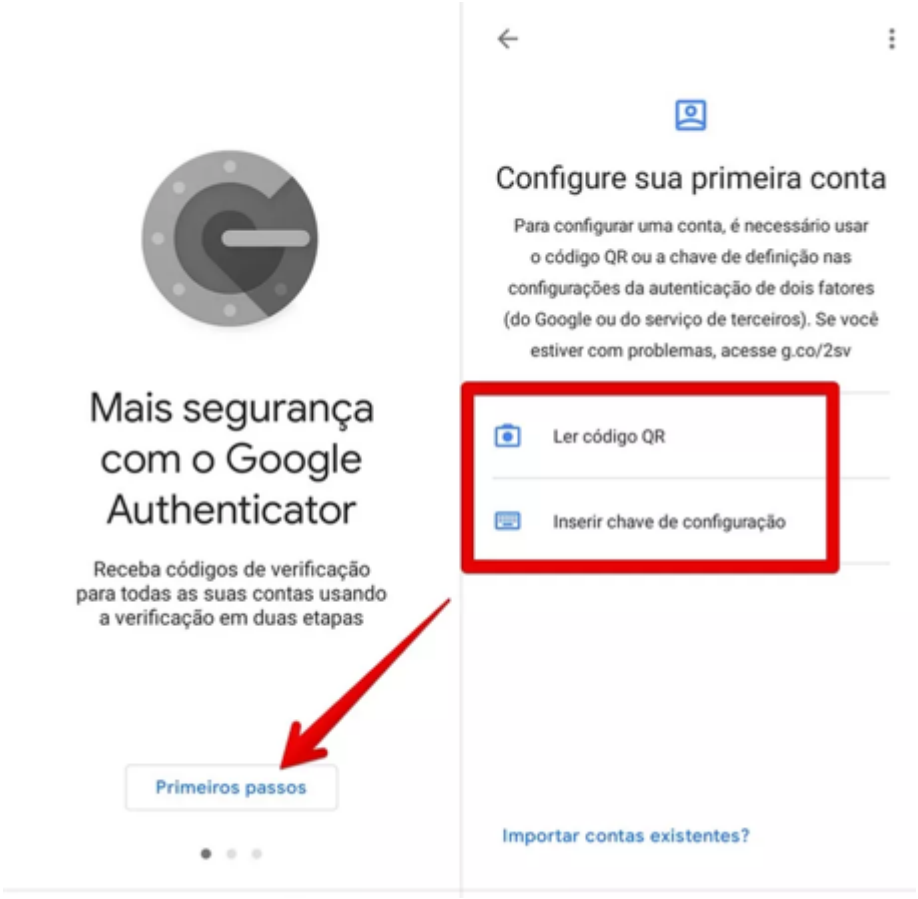


Fonte: A autora

O segundo passo é realizar o cadastro de pessoas, assim, quando for desejado conceder acesso a um novo individuo será feito seu cadastro o vinculando a um ou mais perfis de acesso. O sistema conta com a autenticação de dois fatores do Google Authenticator, *app* que deve ser instalado no *smartphone* do individuo cadastrado, o *app* é disponibilizado gratuitamente tanto para Android como para iOS. Portanto, no ato do cadastro de Pessoa, é necessário estar com o *app* instalado, e ler o código QR presente no cadastro de pessoa, através do acesso do *app* e clicando na opção "Ler código QR"conforme detalha a Figura 35.

[1] Disponível em: https://www.techtudo.com.br/dicas-e-tutoriais/2020/08/como-usar-o-google-authenticator-no-celular.ghtml>. Acesso em 03 mai. 2022

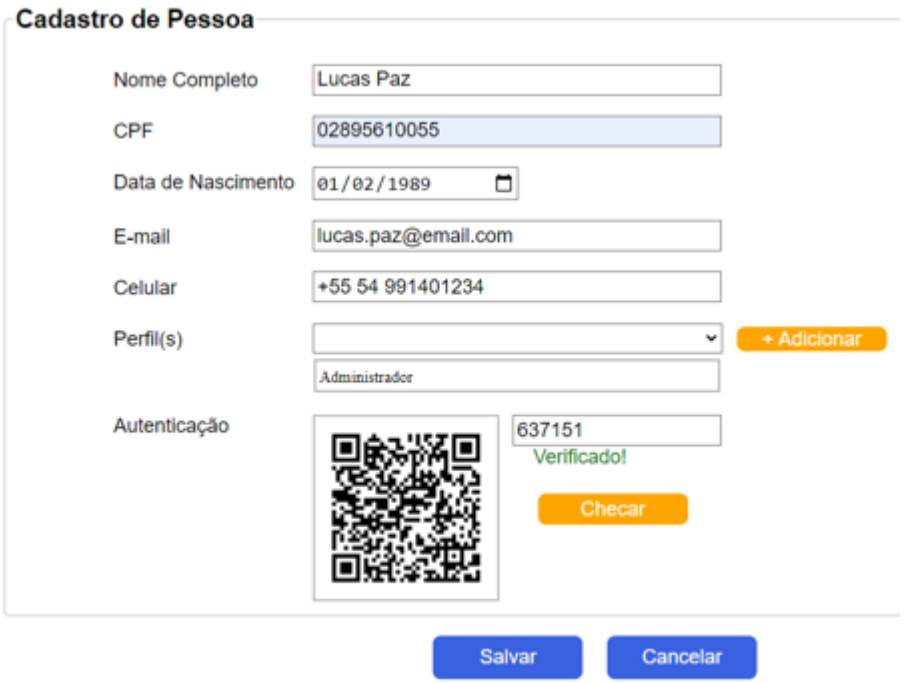
Figura 35 – Ler código QR no Google Authenticator



Fonte: techtudo [1]

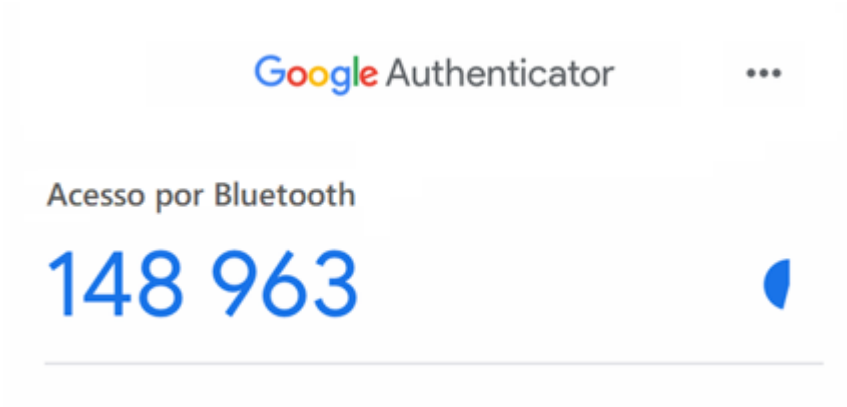
A Figura 36 exemplifica um cadastro de Pessoa, onde o código QR foi validado após inserir o *token* gerado no *app*, como ilustra a Figura 37. Após ler o código QR no ato do cadastro, uma conta nomeada "Acesso por Bluetooth" vai aparecer no *app*, gerando um novo *token* a cada 10 segundos. Importante salientar que cada cadastro de Pessoa tem um código QR aleatório e único, logo, a geração dos *tokens* no *app* que está vinculada ao código QR da pessoa cadastrada também é única e aleatória, assim cada pessoa terá um *token* válido diferente naquele instante de tempo.

Figura 36 – Exemplo de pessoa cadastrada



Fonte: A autora

Figura 37 – Exemplo de token após ler o código QR do cadastro



Fonte: A autora

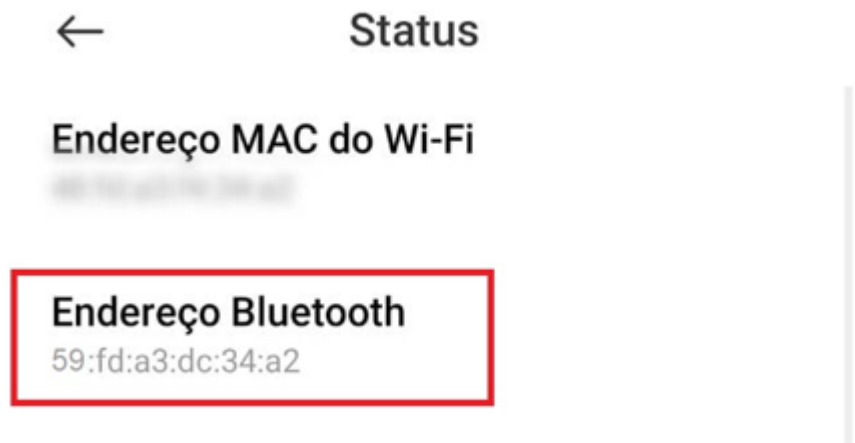
Por último é necessário fazer o cadastro do dispositivo. Embora outros dispositivos tenham comunicação por Bluetooth como fones de ouvido, *notebooks*, *tablets* e muitos outros, o protótipo se propõe a validar apenas *smartphones*, já que os mesmos possibilitam a autenticação de dois fatores. Assim, o sistema conta com mais esta validação, já que o curto alcance da detecção de dispositivos por Bluetooth obriga a pessoa a estar com seu *smartphone* a poucos metros de distância do local de acesso. O cadastro de dispositivo exemplificado na Figura 38 conta com o campo principal "Endereço Bluetooth" que é obrigatório. A informação deste campo pode ser encontrada nos próprios *smartphones* como mostra a Figura 39 através do caminho "Configurações > Sobre o telefone > Todas as especificações > Status > Endereço Bluetooth"

em *smartphones* com o sistema Android utilizados nas simulações.

Figura 38 – Exemplo de cadastro de dispositivo

Fonte: A autora

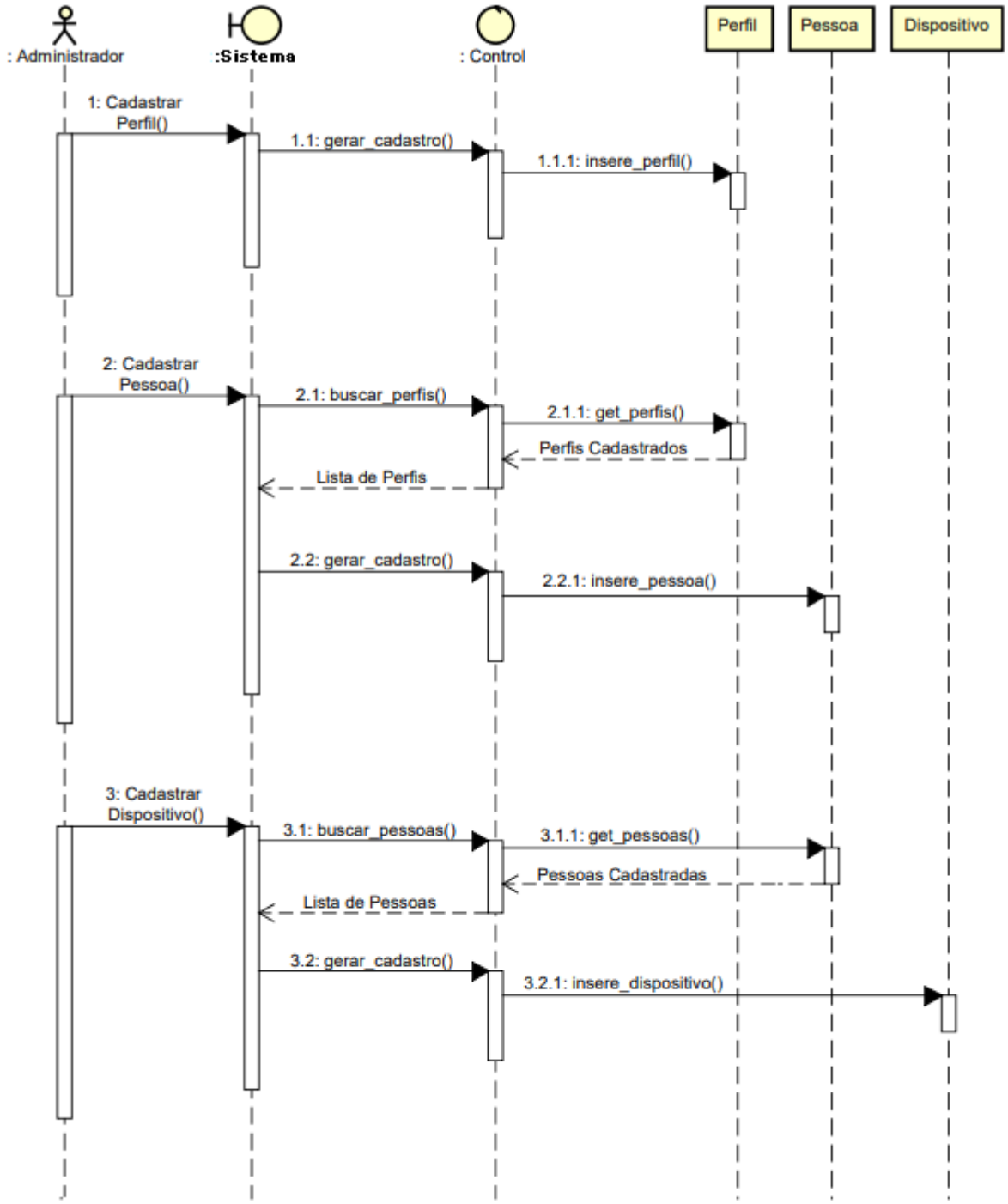
Figura 39 – Exemplo de Endereço Bluetooth encontrado nos smartphones



Fonte: A autora

Para uma demonstração mais técnica do fluxo de pré-cadastros do sistema, a Figura 40 contém o diagrama de sequência do mesmo.

Figura 40 – Diagrama de Sequência Pré-Cadastros



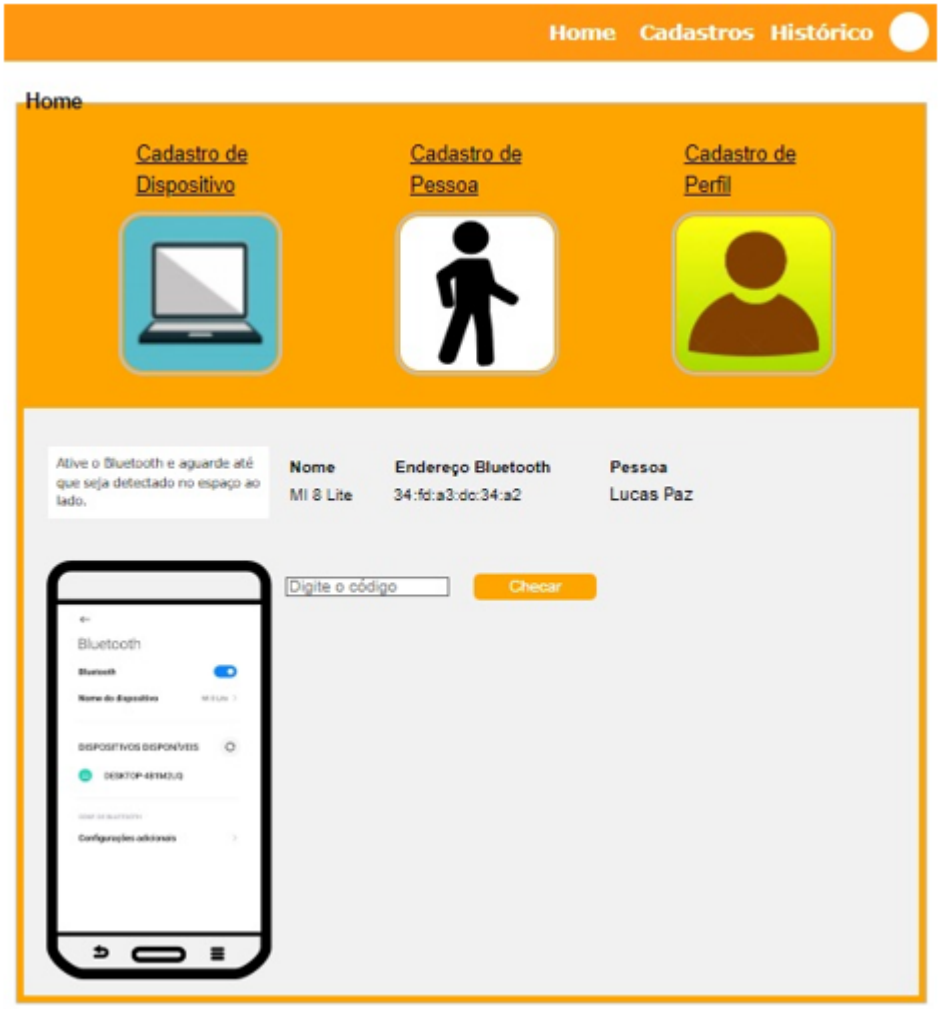
Fonte: A autora

5.2 DETECÇÃO DOS DISPOSITIVOS

Após os devidos cadastros já previamente realizados é possível acessar o ambiente controlado portando seu respectivo *smartphone*. Para isso é necessário acessar a tela inicial do sistema apresentada na Figura 41, que conta com ícones para facilitar o acesso aos cadastros e

também com uma área onde são mostrados todos os dispositivos ao alcance do *notebook* utilizado no protótipo. Nesta mesma área em cinza, possui um campo para se colocar o *token* recebido no *app* do Google Authenticator.

Figura 41 – Visão geral da Tela Inicial do Sistema

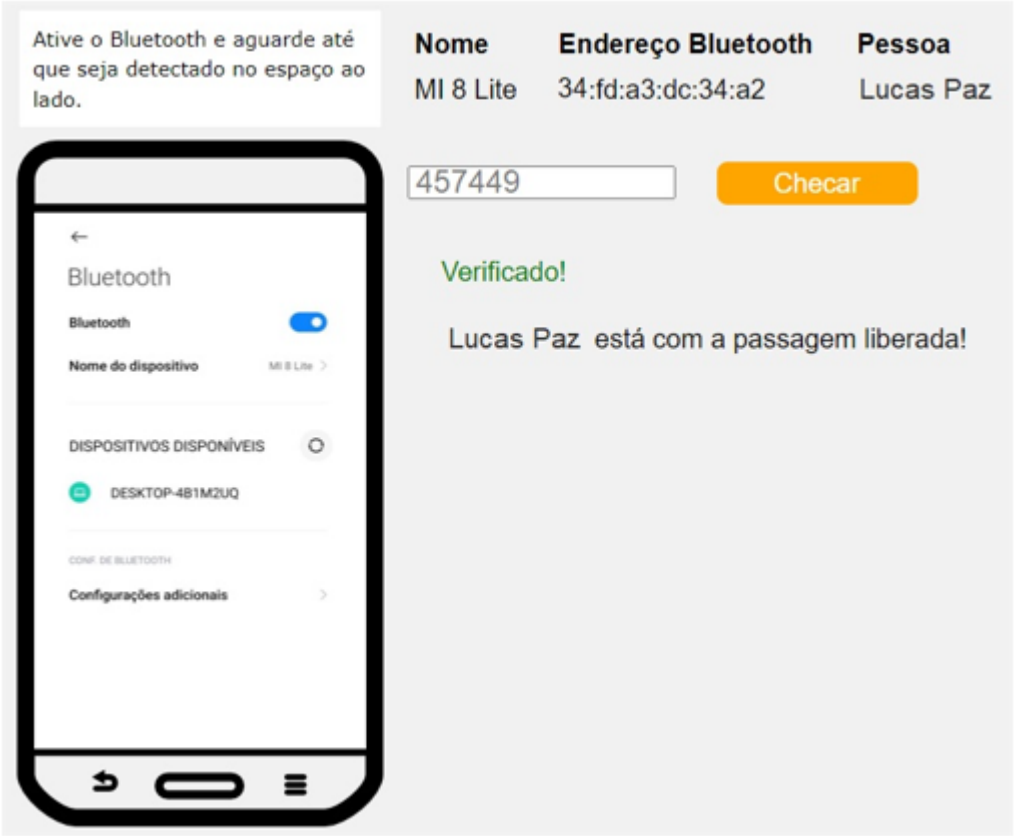


Fonte: A autora

Para que o *smartphone* seja detectado, é preciso que se entre na tela de Bluetooth do celular, a mesma que é ilustrada no lado esquerdo da listagem como mostra a Figura 42, ative o Bluetooth e aguarde cerca de 2 segundos até que o dispositivo seja listado no sistema. Após o dispositivo ser detectado e mostrado, o mesmo é armazenado na tabela temporária "dispositivo_detectado"e ficará armazenado nela por 2 minutos antes de ser excluído. Estes 2 minutos são um período mais que suficiente para abrir o *app* do Google Authenticator e inserir o *token* no sistema. Tempo definido com base em testes onde se consegue fazer o processo calmamente, errar o código do *token* 3 vezes e retentar. Após inserir o código válido será identificada a pessoa vinculada ao dispositivo. E então é feita a validação de perfis de acesso que consiste em passar por todos os perfis vinculados a pessoa e verificar se a data e horário atual está dentro do período de acesso liberado para algum dos perfis, estando liberado para ao menos um perfil

será liberada a passagem, imprimindo uma mensagem em tela conforme se pode ver também na Figura 42.

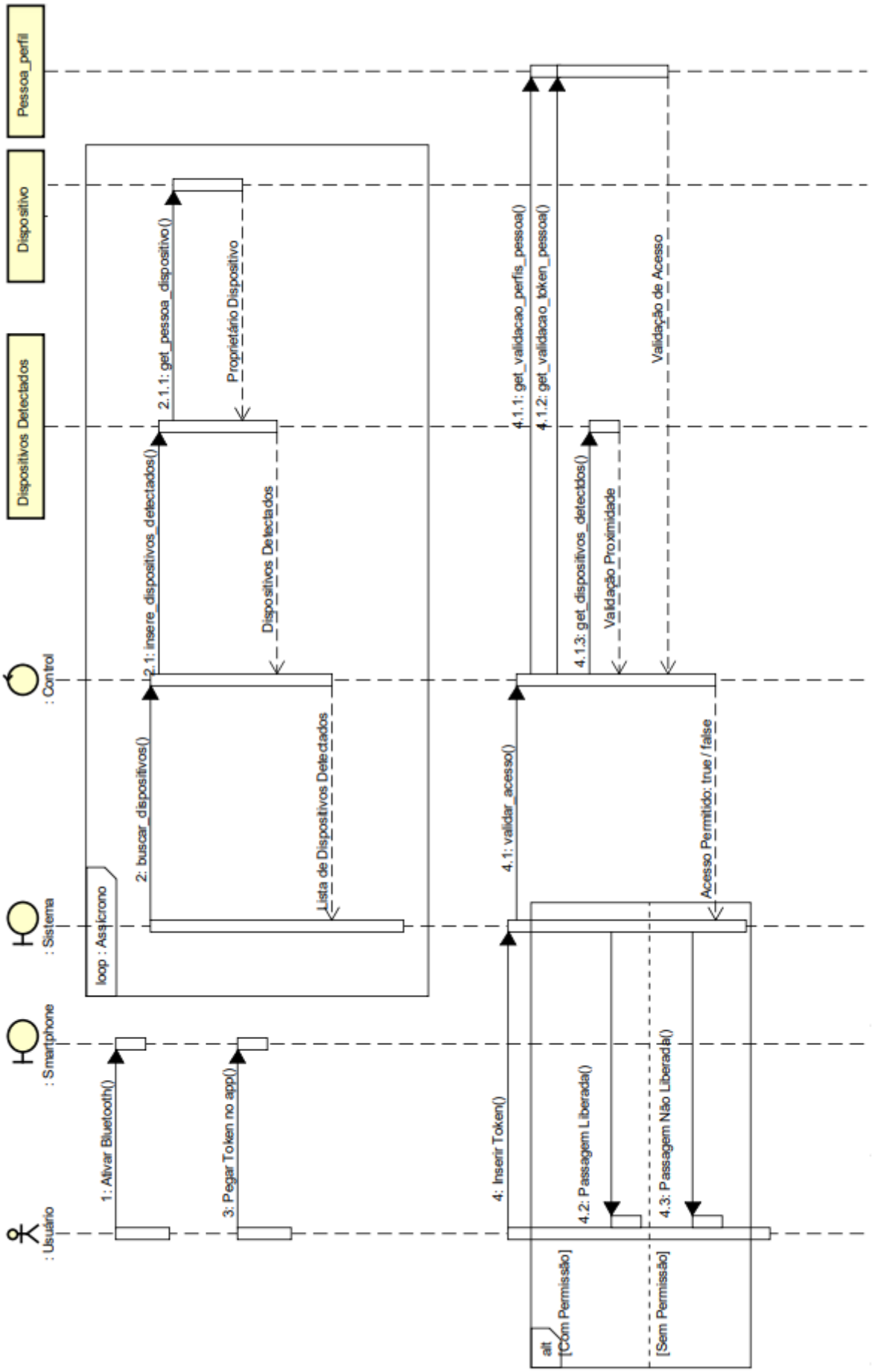
Figura 42 – Tela Bluetooth e listagem



Fonte: A autora

A Figura 43 demonstra os processos do fluxo de acesso do sistema através do diagrama de sequência do mesmo. Caso houvesse alguma automação com uma barreira física como uma catraca por exemplo, a passagem seria liberada e só validaria a entrada da pessoa após girar a mesma. Teria assim, mais uma validação, não sendo ela obrigatória e nem desenvolvida neste trabalho.

Figura 43 – Diagrama de Sequência do Fluxo de Acesso



Fonte: A autora

6 IMPLEMENTAÇÃO DO PROTÓTIPO DA PROPOSTA DE SOLUÇÃO

Neste capítulo será detalhado como foi o desenvolvimento da proposta de solução. Será apresentada a arquitetura, tecnologias, programação, *frameworks*, e todos os recursos utilizados no seu desenvolvimento.

6.1 ARQUITETURA DA APLICAÇÃO

A arquitetura da aplicação escolhida foi a *Back-end for Front-end* (BFF), onde há uma divisão entre o lado do cliente (*client-side*) e o lado do servidor (*server-side*). No lado do servidor, o *back-end* é responsável por lidar com toda parte de gerenciamento dos dados. Podendo, assim, disponibilizar e manipular esses dados conforme a demanda de um ou mais *front-ends*, a proposta do trabalho conta com apenas um *front*, que permite acessar a aplicação web através de um *browser*. Com a disponibilização de *endpoints* [1], acessados por requisições HTTP ocorre a comunicação entre as aplicações, permitindo que o *front* requisite e altere dados.

Todas as requisições feitas ao *back-end* retornam um JSON com os respectivos dados. Como é no *back* que fica o gerenciamento de dados, a rotina de detecção de dispositivos também ocorre no *back-end* através da função "discover_devices" disponibilizada pelo pacote "bluetooth". Onde para cada dispositivo detectado é fornecido o endereço Bluetooth e o nome do mesmo, esses dados são salvos em uma tabela temporária juntamente com a pessoa, a data e horário da detecção, como pode ser visto na Figura 44.

Figura 44 – Método de Detecção de Dispositivos

```
nearby_devices = bluetooth.discover_devices(duration=4, lookup_names=True)
for addr, name in nearby_devices:
    pessoa_id_get = seleciona_pessoa_por_endereco_bluetooth(addr.lower())
    dispositivo_detectado = DispositivoDetectado(nome=name,
    endereco_bluetooth=addr.lower(), pessoa_id=pessoa_id_get,
    data = datetime.today())
    dispositivos_objetos.append(dispositivo_detectado)
    db.session.add(dispositivo_detectado)
    db.session.commit()
return dispositivos_objetos
```

Fonte: A autora

O *back-end* possui um *endpoint* responsável pelo retorno dos dispositivos detectados. Já o *front-end* da aplicação possui um *listener*[2], onde é feita constantemente a requisição dos

[1] Endereços utilizados para comunicação entre uma API e um sistema externo

dispositivos detectados para o *back*, e quando o status da requisição é alterado indicando que retornou dados, o *listener* é acionado. No *front-end* ficam todas as interfaces do protótipo.

6.2 DESENVOLVIMENTO DO *BACK-END*

Esta seção aborda o desenvolvimento do *back-end* da aplicação, onde constam as classes que representam as entidades do banco de dados, a rotina de detecção de dispositivos com Bluetooth ativo já mencionada, todo o gerenciamento de dados e os *endpoints* que são acessados pelo *front-end*.

6.2.1 Gerenciamento do Banco de dados

O desenvolvimento se iniciou com a construção da base de dados, para isso, no *back-end* foi utilizada a API Flask, o pacote é instalável através do *Python Package Index* (PPI). Também é necessário instalar o flask-sqlalchemy, que é o pacote que possibilita a interação com o banco de dados e o código em Python. Para se fazer a conexão com o banco de dados foi necessário realizar a configuração mostrada na Figura 45, onde basicamente após importar os pacotes descritos é preciso informar a *Uniform Resource Identifier* (URI) do banco de dados.

Figura 45 – Configuração de conexão com o banco de dados

```
from flask import Flask, Response, request
from flask_sqlalchemy import SQLAlchemy
from flask_cors import CORS

app = Flask(__name__)
CORS(app)
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = True
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://root:@localhost/tcc'
db = SQLAlchemy(app)
```

Fonte: A autora

Foi utilizado o banco de dados relacional MySQL, com o usuário padrão "root", executando localmente para fins de teste no *localhost*, porta 3306, conforme conexão ilustrada na Figura 46. Para instalação e configuração do MySQL e do servidor web Apache foi utilizado o software WampServer que dá suporte a tais instalações.

Foi criada uma classe em Python para cada tabela do banco de dados, utilizando a variável "db", que é uma instância da classe SQLAlchemy responsável pela integração da aplicação com o banco. Cada atributo da classe corresponde a uma coluna da tabela do banco de dados, assim, foi criada a classe Perfil correspondente a tabela perfil, conforme é mostrado na figura

[2] O método JavaScript addEventListener () permite configurar funções a serem chamadas quando um evento especificado acontece, como quando um usuário clica em um botão.

Figura 46 – Conexão com o banco de dados

Fonte: A autora

Figura 47. Também foram criadas classes para as demais tabelas, sendo elas: Pessoa, PessoaPerfil, Dispositivo, e DispositivoDetectado, seguindo a mesma lógica de correlação entre a classe e a tabela.

Figura 47 – Classe Perfil

```
class Perfil(db.Model): # Perfil estende db

    __tablename__ = 'perfil'
    id = db.Column(db.Integer, primary_key= True)
    nome = db.Column(db.String(50));
    dias = db.Column(db.Enum(DiasEnum));
    informar_periodo = db.Column(db.Boolean());
    data_inicial = db.Column(db.Date());
    data_final = db.Column(db.Date());
    hora_inicial = db.Column(db.Time(50));
    hora_final = db.Column(db.Time(50));

    def to_json(self):
        return {
            "id": self.id,
            "nome": self.nome,
            "dias": self.dias,
            "informar_periodo": self.informar_periodo,
            "data_inicial": self.data_inicial,
            "data_final": self.data_final,
            "hora_inicial": self.hora_inicial,
            "hora_final": self.hora_final
        }
```

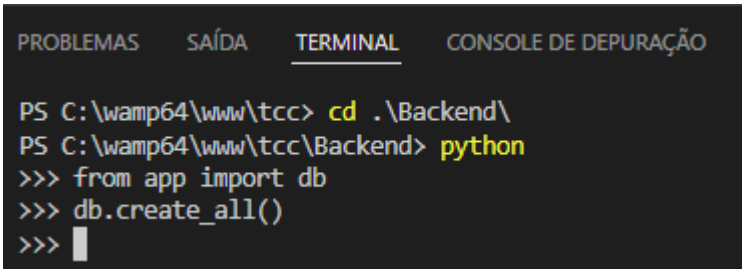
Fonte: A autora

Cada classe conta com um único método, nomeado "to_json" também exibido na Figura 47, método utilizado quando o *back-end* retorna a requisição HTTP feita pelo *front-end* no formato JSON.

Para o funcionamento local do protótipo se iniciam os serviços do MySQL e do servidor web Apache através do WampServer. Antes de começar a usar o sistema é preciso criar o banco

de dados, para isso foi criado um esquema nomeado "tcc". Com o esquema já criado é preciso navegar até o arquivo "app.py" pelo terminal, onde constam todas as classes correspondentes as tabelas do banco, e executar os comandos responsáveis pela criação das mesmas no esquema, conforme mostra a Figura 48. Ao rodar o comando de criação "db.create_all()", será criada uma tabela para cada respectiva classe presente no arquivo. Após está configuração inicial o banco de dados da aplicação está criado e pronto para o uso do protótipo do software.

Figura 48 – Comandos para criação das tabelas



Fonte: A autora

6.2.2 Endpoints

O protótipo do sistema utiliza a API *Representational State Transfer* (REST) para comunicação entre as aplicações e conta com 21 *endpoints* que podem ser acessados pelo *front-end*, suprindo a necessidade de qualquer interação com dados que a aplicação cliente precise. As URLs dos *enpoints* utilizando o método *GET* estão descritas no Quadro 9, já utilizando o método *POST* no Quadro 10, e com o método *DELETE* no Quadro 11 respectivamente. O *Internet Protocol* (IP) do servidor é 127.0.0.1, utilizando a porta padrão 5000.

Quadro 9 – *Endpoints* utilizando método *GET*

URLs	**Descrição**
http://127.0.0.1:5000/perfis	Retorna todos os perfis cadastrados
http://127.0.0.1:5000/perfil/<id>	Retorna o perfil do respectivo ID
http://127.0.0.1:5000/pessoas	Retorna todas as pessoas cadastradas
http://127.0.0.1:5000/pessoa/<id>	Retorna a pessoa do respectivo ID
http://127.0.0.1:5000/dispositivos	Retorna todos os dispositivos cadastrados
http://127.0.0.1:5000/dispositivo/<id>	Retorna o dispositivo do respectivo ID
http://127.0.0.1:5000/pessoa/<endereco_bluetooth>	Retorna o ID da pessoa vinculada ao dispositivo
http://127.0.0.1:5000/deteccao	Retorna os dispositivos detectados por Bluetooth
http://127.0.0.1:5000/dispositivos_detectados	Retorna todos dispositivos salvos na tabela temporária

Fonte: A autora (2021).

6.3 DESENVOLVIMENTO DO *FRONT-END*

No *front-end* do protótipo da aplicação constam todas as telas de interface com o usuário, arquivos contidos dentro de uma pasta nomeada "html". Sendo elas as telas de cadastro e

Quadro 10 – *Endpoints* utilizando método *POST*

URLs	Descrição
http://127.0.0.1:5000/perfil	Insere um novo perfil no bd
http://127.0.0.1:5000/perfil/<id>	Atualiza o perfil do respectivo ID
http://127.0.0.1:5000/pessoa	Insere uma nova pessoa no bd
http://127.0.0.1:5000/pessoa_perfil	Insere um novo vinculo entre pessoa e perfil no bd
http://127.0.0.1:5000/pessoa/<id>	Atualiza a pessoa do respectivo ID
http://127.0.0.1:5000/dispositivo	Insere um novo dispositivo no bd
http://127.0.0.1:5000/dispositivo/<id>	Atualiza o dispositivo do respectivo ID
http://127.0.0.1:5000/deteccao	Insere os dispositivos detectados na tabela temporária

Fonte: A autora (2021).

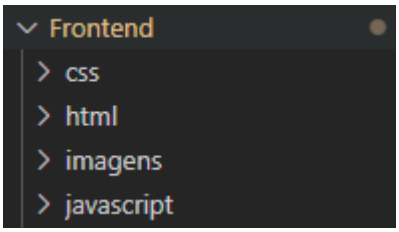
Quadro 11 – *Endpoints* utilizando método *DELETE*

URLs	Descrição
http://127.0.0.1:5000/perfil/<id>	Deleta o perfil do respectivo ID
http://127.0.0.1:5000/pessoa/<id>	Deleta a pessoa do respectivo ID
http://127.0.0.1:5000/dispositivo/<id>	Deleta o dispositivo do respectivo ID
http://127.0.0.1:5000/dispositivo_detectado/<id>	Deleta o dispositivo detectado do respectivo ID

Fonte: A autora (2021).

a tela inicial já exploradas em detalhe no Capítulo 5. Além dos arquivos HTML que contém a estrutura das páginas web, também dentro da pasta nomeada "css"constam os arquivos de estilização da página. O front-end também conta com uma pasta nomeada "imagens"contendo todas as imagens utilizadas no projeto e por fim uma pasta nomeada "javascript"contendo todos os arquivos JavaScript responsáveis por funções mais complexas atreladas as funcionalidades de cada página. A estrutura de pastas pode ser vista na Figura 49.

Figura 49 – Estrutura de pastas do *front-end*



Fonte: A autora

6.3.1 Desenvolvimento dos Cadastros

Todos os cadastros foram feitos seguindo o mesmo leiaute e estilização, apresentando, assim, a mesma identidade visual. O desenvolvimento dos mesmos será explorado nesta sessão.

6.3.1.1 Cadastro de Perfil

O cadastro de Perfil é o mais conciso, contando basicamente com duas funções. A função de cadastrar o perfil que é chamada quando se clica no botão "Salvar", a mesma atribui os

valores dos campos preenchidos em tela a um objeto nomeado "body". Ao final da função de cadastro é chamada a função "FazPost", passando por parâmetro o corpo da mensagem com o perfil que será salvo no banco e a URL do método *POST*: "http://127.0.0.1:5000/perfil". *End-point* o qual está presente no *back-end*, conforme já mencionado no Quadro 10. A função citada responsável por fazer a requisição HTTP ao *back-end* pode ser visualizada na Figura 50.

Figura 50 – Função responsável por fazer requisição POST

```
function fazPost(url, body) {

    let request = new XMLHttpRequest()
    request.open("POST", url, true)
    request.setRequestHeader("Content-type", "application/json")
    request.send(JSON.stringify(body))

    return request.responseText
}
```

Fonte: A autora

6.3.1.2 Cadastro de Pessoa

O cadastro que Pessoa é o mais complexo, pois contém a integração com a API do Google Authenticator responsável pela a autenticação de dois fatores já mencionada no fluxo do Capítulo 5, ademais possibilita vincular a pessoa a um ou mais perfis.

Além das mesmas funções já citadas no cadastro de perfil, possui uma função que é chamada ao carregar a página, a mesma chama outra função que faz uma requisição *GET* ao *back-end*, afim de buscar os perfis já cadastrados no sistema e após inserir os perfis de forma dinâmica no *ComboBox* do HTML presente na página. A função que monta o *ComboBox* de forma dinâmica está na Figura 51.

Figura 51 – Função que constrói *ComboBox* dinamicamente

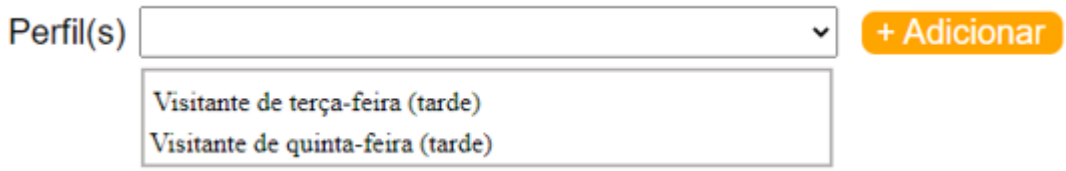
```
window.onload = function() {
    url = "http://127.0.0.1:5000/perfis"
    selectPerfis = fazGet(url)

    $('#perfil_add').append(`<option>`+`</option>`)
    for (i=0; i<selectPerfis.length; i++){
        perfis.push(selectPerfis[i].nome)
        $('#perfil_add').append(`<option value=`+selectPerfis[i].id+`>`
        +selectPerfis[i].nome+`</option>`)
    }
};
```

Fonte: A autora

Como a pessoa pode ter um ou mais perfis, foi implementada uma funcionalidade onde o usuário seleciona o perfil que deseja e clica no botão "Adicionar" ao lado do *ComboBox*, permitindo, adicionar quantos perfis seja necessário. Os perfis já selecionados vão sendo adicionados de forma dinâmica, assim como na montagem do *ComboBox* dinâmico só que agora acrescentado linhas ao elemento de tabela do HTML, esta tabela situa-se abaixo do campo de seleção como pode ser visto na Figura 52.

Figura 52 – Perfis vinculados a pessoa



Fonte: A autora

Para implementar a integração com o Google Authenticator é necessário instalar dois pacotes, o Speakeasy que é um gerador de senha de uso único, ideal para uso na autenticação de dois fatores, compatível com o Google Authenticator e com outros dispositivos de dois fatores. E o Qrcode que auxilia na geração de códigos QR. A configuração inicial consiste em atribuir um nome a conta que aparecerá no *app* de autenticação e coletar a string do código QR gerado em tela, assim como seu código ASCII vinculado a ele que será salvo no banco junto a pessoa posteriormente. Esta configuração pode ser vista na Figura 53. Para a finalização do cadastro de Pessoa é preciso escanear o código QR gerado em tela e inserir o *token* temporário gerado no *app*. Assim, após validação que é feita na função presente na Figura 54 retornará "true" caso o *token* inserido confira com o gerado no *app*, podendo finalizar o cadastro.

Figura 53 – Implementação com os pacotes Speakeasy e Qrcode

```
const speakeasy = require('speakeasy')
const qrcode = require('qrcode')

var secret = speakeasy.generateSecret({
    name: "Acesso por Bluetooth"
})

qrcode.toDataURL(secret.otpauth_url, function(data){
    document.getElementById("novo_qrcode").src = [data]
    ascii = secret.ascii
})
```

Fonte: A autora

Figura 54 – Validação do *token*

```
verify = speakeasy.totp.verify({
    secret: secret.ascii,
    encoding: 'ascii',
    token: token_campo.value
})
```

Fonte: A autora

6.3.1.3 Cadastro de Dispositivo

O cadastro de dispositivo não conta com nenhuma função complexa, contém a função de cadastro e de requisição *POST*, assim como o cadastro de perfil, com o único adicional de que é necessário vinculá-lo a uma pessoa. O *ComboBox* com as pessoas já cadastradas no sistema é montado de forma dinâmica, assim como o *ComboBox* de perfis já detalhado no cadastro de Pessoa.

6.3.2 Desenvolvimento da Tela Inicial

A tela inicial do protótipo é a página mais importante do sistema, onde é de fato liberado o acesso para fins de simulação, além de conter instruções e um menu para os cadastros como pode ser visto no Capítulo 5. O mecanismo principal da tela consiste em fazer a verificação de novos dispositivos detectados constantemente de maneira assíncrona. O fluxo lógico da tela consiste em carregar o HTML com a estilização da página e assim que a mesma é inicializada já começa o processo de detecção de dispositivos com o Bluetooth ativado ao alcance da máquina utilizada na simulação.

A detecção contínua de dispositivos acontece através da chamada da função "fasGetDeteccao". Para que esta função funcione de forma assíncrona, é preciso que ao fazer a requisição HTTP ao *back-end* acessando o *endpoint* "http://127.0.0.1:5000/deteccao" passe o terceiro parâmetro do método responsável pela abertura da requisição como "true", indicando que essa requisição ocorrerá de forma assíncrona, como pode ser visto na linha 19 da Figura 55. Assim, enquanto não chega o retorno da requisição, o front-end não ficará parado aguardando, e dará continuidade aos demais processos. Foi adicionado um evento de *listener* à requisição que pode ser visto na linha 23 da Figura 55. Assim, quando o estado dela muda, indicando o retorno da requisição, é dado seguimento ao processo de listagem de dispositivos em tela. Todo o comportamento da função descrito pode ser visto na Figura 55.

Após o retorno da requisição de detecção de dispositivos é chamada uma função, a mesma é responsável por deletar os dispositivos que estão salvos na tabela temporária do banco por mais de dois minutos. Em paralelo já é chamada novamente a função de detecção de dispositivos, para que seja executada assincronamente enquanto segue o processo de deletar os dispositivos com o tempo já expirado. Após a exclusão dos dispositivos vencidos são adicio-

Figura 55 – Função responsável pela Detecção de Dispositivos

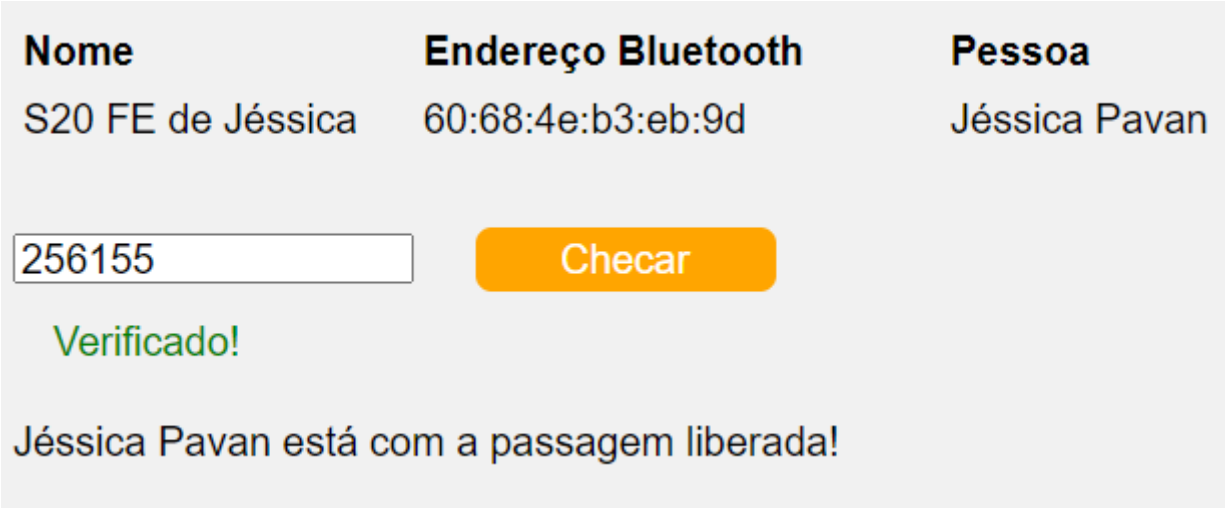
```
16 function fazGetDeteccao(url) {
17   let selectNovosDispositivos = []
18   let request = new XMLHttpRequest()
19   request.open("GET", url, true)
20   request.responseType="text";
21   request.send()
22
23   request.addEventListener('readystatechange', function() {
24     selectNovosDispositivos = JSON.parse(request.responseText)
25                               .dispositivos detectados
26     deletaDispositivosDetectadosMaisDeDoisMinutos(selectNovosDispositivos)
27   });
28 }
```

Fonte: A autora

nados a lista os demais que serão exibidos em tela, os novos dispositivos detectados, os quais permanecerão sendo exibidos por dois minutos até que também sejam apagados. Tempo este que é o suficiente para que o usuário acesse o ambiente.

Antes de listar os dispositivos em tela é verificado se o dispositivo já está cadastrado no sistema, após passar por tal verificação também é examinado o perfil de acesso da pessoa vinculada ao dispositivo. Após averiguar que o dispositivo está vinculado a um individuo com acesso permitido naquele período e perímetro é feita então a validação do *token*, semelhante a já citada na Seção 6.3.1.2 na Figura 54. Assim, caso alguma validação seja negada, é exibida uma mensagem em tela dizendo que o individuo não possui acesso ao ambiente, e caso passe em todas as validações é exibida a mensagem validando o acesso, como o mostra a Figura 56.

Figura 56 – Liberação de Acesso



Fonte: A autora

6.4 TESTES DA APLICAÇÃO

A aplicação foi testada utilizando um notebook Acer Aspire 5. Para isso foi posto o *back-end* da aplicação para rodar, o qual fica em execução no IP http://127.0.0.1:5000/. Após, toda a comunicação com o banco de dados já foi feita, e a aplicação está pronta para manipular quaisquer dados que o *front-end* requisite. O *front* é executado localmente na mesma máquina e é acessado através do endereço http://localhost/tcc/Frontend/tela_inicial.html. A partir da tela inicial é possível acessar todos os cadastros e funcionalidades do protótipo.

Os testes foram realizados através do cadastro de dois perfis fictícios com permissões em dias distintos. Esse perfis foram vinculados a duas pessoas também fictícias respectivamente. Ademais foram cadastrados dois celulares reais de marcas diferentes, os quais foram utilizados para testes.

Foi testado a que distância os celulares são detectados pelo notebook, sendo constatado uma média de 5 metros. Foram também executados testes de acesso com ambos os celulares, com perfis com e sem acesso, e após com a autenticação de dois fatores, concluindo que o sistema estava alcançando o objetivo proposto.

7 CONCLUSÕES

Visando a agilidade de acesso e a segurança de locais com grande fluxo de pessoas, uma possível alternativa a ser adotada é automatização do mesmo. A partir de um sistema web é possível gerenciar acessos, podendo ser integrado futuramente com a liberação de uma barreira física após uma validação de permissão. Para permitir o acesso é preciso fornecer uma interface que possibilite gerenciar e parametrizar como essa verificação de acesso será feita. Para executar a autenticação é preciso armazenar dados que identificam o indivíduo e seus dispositivos, e perfis associados que indicam quando o mesmo tem acesso permitido. Esta parametrização permite que, ao individuo tentar fazer o acesso, sejam verificados os dados do mesmo, dados que já estão pré-armazenados, e a partir deles liberar ou não o acesso.

Foi realizado o desenvolvimento de um protótipo de sistema web para fim de gerenciar um controle de acesso. Foram desenvolvidos cadastros de pessoa, perfil e dispositivo, sendo os dados destes cadastrados e armazenados em um banco de dados. Além de ter sido desenvolvida também a tela inicial, onde ocorre a validação do acesso. Foi desenvolvida uma rotina em Pyhton utilizando a API PyBluez que fornece uma função de detecção de dispositivos a partir do seu Bluetooth ativado, utilizando o módulo Bluetooth 5 do notebook Aspire 5 A515-54-55L0 utilizado no protótipo. Após testes com 3 *smartphones* diferentes foi identificado o alcance médio de 5 metros entre e o notebook e os *smatphones* em ambiente interno. Foi acrescentada ao protótipo uma autenticação de dois fatores para aumento de segurança. O acesso só é validado após entrar na tela de ativação do Bluetooth do *smartphone* e inserir o *token* no local de acesso em um intervalo de dois minutos.

Foram encontrados alguns obstáculos para o desenvolvimento desejado. O primeiro foi a questão da autenticação de dois fatores só poder ser utilizada por *smartphones*, então modificou-se os dispositivos utilizados na validação de acesso, se restringindo apenas a celulares. O segundo ponto se trata da necessidade de estar na tela de ativação do Bluetooth para o dispositivo em questão se tornar visível para os demais. Para contornar está situação foi salvo o dispositivo no banco de dados por um período temporário de 2 minutos, para que o individuo consiga sair da tela de ativação do Bluetooth, pegar o código no Google Authenticator e ter tempo o suficiente para acessar o local após sair da tela de Bluetooth do *smartphone*.

O presente trabalho tem como limitação o desenvolvimento de um um protótipo de software para gerenciamento de acesso e se mostrou eficiente para seu objetivo. Para garantir uma liberação física do acesso, assim, aumentando a segurança da validação surge a possibilidade de implementar em trabalhos futuros uma integração com alguma automação de liberação física, como por exemplo uma catraca eletrônica ou uma porta eletromagnética. O protótipo foi simulado localmente tanto no lado do cliente quanto do servidor, o que sugere outra possibilidade de trabalho futuro, relacionado a colocar o *back-end* da aplicação num servidor externo, podendo

também desenvolver um *front-end mobile* para a plicação, assim, possuindo mais uma aplicação acessando o mesmo *back-end*. E também o desenvolvimento futuro da tela de histórico para consulta posterior.

REFERÊNCIAS

ALMEIDA, C. A. B. de. **Tecnologias Aplicadas à Segurança**: um guia prático. 1th. ed. Curitiba: InterSaberes, 2018. ISBN 978-85-5972-649-7.

ALVES, W. P. **Banco de Dados**. 1th. ed. São Paulo: Editora Saraiva, 2014. ISBN 978-85-365-1896-1.

____. **Desenvolvimento e Design de Sites**. 1th. ed. São Paulo: Saraiva, Érica, 2014. ISBN 978-85-365-1901-2.

____. **Projetos de Sistemas Web**: Conceitos, estruturas, criação de banco de dados e ferramentas de desenvolvimento. 1th. ed. São Paulo: Saraiva Educação, Érica, 2015. ISBN 978-85-365-3642-2.

BANIN, S. L. **Python3**: Conceitos e aplicações: uma abordagem didática. São Paulo: Saraiva, Érica, 2018. ISBN 978-85-365-3025-3.

COLLOTTA, M. *et al.* Bluetooth 5: A concrete step forward toward the iot. **IEEE Communications Magazine**, v. 56, p. 125–131, 2018. ISSN 1558-1896.

GLOBAL SYSTEM FOR MOBILE COMMUNICATION ASSOCIATION. **The Mobile Economy 2021**. [S.l.], 2021.

GOKHALE, P.; BHAT, O.; BHAT, S. Introduction to iot. **International Advanced Research Journal in Science, Engineering and Technology**, v. 5, p. 41–44, 2018. ISSN 2393-8021.

GUEDES, Y. M.; SANTOS, M. C. P. **Fechadura Microcontrolada com Gerência Centralizada**. Juiz de Fora: Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais, 2016.

GUERRA, A. R. **Desenvolvimento para dispositivos móveis**. 1th. ed. Curitiba: Contentus, 2020. ISBN 978-65-5745-472-5.

HUH, J.-H.; SEO, K. **An Indoor Location-Based Control System Using Bluetooth Beacons for IoT Systems**. Geumjeong-gu: Catholic University of Pusan, 2017.

MACHADO, F. N. R. **Banco de Dados**: Projeto e implementação. 4th. ed. São Paulo: Saraiva Educação, Érica, 2020. ISBN 978-85-365-3270-7.

MACIEL, F. M. de B. **Python e Django**: Desenvolvimento web moderno e ágil. Rio de Janeiro: Alta Books, 2020. ISBN 978-65-5520-097-3.

MANENTE, L. O.; CRESPO, M. C. **Desenvolvimento de um Sistema de Controle de Acesso com Armazenamento de Dados em Nuvem**. 86 f. Dissertação (Trabalho de Conclusão de Curso) — Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2019.

MELO, R. de O. **Protótipo de um Sistema para Monitoramento de Presença**. 64 f. Dissertação (Trabalho de Conclusão de Curso) — Universidade de Caxias do Sul, Caxias do Sul, 2018.

MILETTO, E. M.; BERTAGNOLLI, S. d. C. **Desenvolvimento de Software II**: Introdução ao desenvolvimento web com html, css, javascript e php. São Paulo: Bookman, 2014.

PORTELLA, P. R. A. **Gestão de Segurança**: Segurança física. sistemas de proteção. história, metodologia e doutrina. 2th. ed. Rio de Janeiro: Rio, 2011. ISBN 85-7579-027-7.

ROCHOL, J. **Sistemas de Comunicação sem Fio**: Conceitos e aplicações. São Paulo: Bookman, 2018. ISBN 978-85-8260-456-4.

SILVA, D. **Desenvolvimento para dispositivos móveis**. São Paulo: Pearson, 2017. ISBN 978-85-430-2025-9.

TANENBAUM, A. S.; WETHERALL, D. **Desenvolvimento para dispositivos móveis**. 5th. ed. São Paulo: Pearson, 2011. ISBN 978-85-7605-924-0.

ZENKER, A. M. *et al.* **Arquitetura de Sistemas**. Porto Alegre: SAGAH EDUCAÇÃO S.A., 2019. ISBN 978-85-9502-976-7.