



UNIVERSIDADE AUTÓNOMA DE LISBOA
LUÍS DE CAMÕES

DEPARTAMENTO DE CIÊNCIAS E TECNOLOGIAS

**(PROTÓTIPO DE UM SISTEMA INTELIGENTE PARA SEGURANÇA
RESIDENCIAL)**

Relatório de Projeto para a obtenção do grau de Licenciatura em Engenharia
Informática

Autor/a: Artur Victor Meireles Costa, Enzo Bernardo Vieira de Sousa, Francisco Duarte dos
Santos Alves, Tomás Simões Dias

Orientador/a: Professor Doutor Héctor Orrillo Ascama

Número do/a candidato/a: 30007662, 30006693, 30007068, 30007620

junho de 2023

Lisboa

(Página em Branco)

Dedicatória

Dedicamos este relatório de projeto final a todas as pessoas que nos acompanharam e apoiaram ao longo desta jornada.

Em especial, agradecemos aos nossos familiares, amigos e entes queridos que estiveram ao nosso lado, incentivando-nos e compreendendo as demandas e dedicação necessárias para a realização deste projeto.

Expressamos também a nossa gratidão aos professores e orientadores que nos guiaram ao longo do processo, compartilhando os seus conhecimentos e contribuindo para o nosso crescimento acadêmico e profissional.

Aos nossos colegas de equipa, agradecemos pela colaboração, trabalho em conjunto e superação de desafios. Foi com união e determinação que alcançamos os nossos objetivos e entregamos este projeto em qualidade.

Que este relatório possa ser um reflexo do nosso esforço, dedicação e aprendizado ao longo desta experiência académica.

Dedicatória feita com gratidão e reconhecimento,

Artur Costa, Enzo Sousa, Francisco Alves e Tomás Dias

Epígrafe

“Education is the most powerful weapon which you can use to change the world.” - Nelson Mandela [1]

"Knowledge itself is power." - Sir Francis Bacon [2]

"Persistence is the path to success." - Charles Chaplin [3]

Resumo

O relatório apresenta o desenvolvimento de um protótipo de um Sistema Inteligente para segurança residencial de baixo custo, no contexto da Domótica. O objetivo principal do projeto é implementar um sistema que ofereça segurança, conforto e comunicação total num espaço residencial.

O sistema é composto por diferentes módulos, incluindo um módulo sensor para detecção de possíveis invasões, uma central de processamento para comunicação entre os módulos, processamento de dados e armazenamento numa base de dados, um módulo de alarme que emite um sinal sonoro em caso de invasão e um sistema de notificações para informar o utilizador sobre a ocorrência.

Além disso, o projeto também prevê o desenvolvimento de uma aplicação de controle, que permite gerir utilizadores, controlar o módulo de alarme e receber notificações de invasão.

O protótipo busca oferecer uma solução eficiente e acessível para a segurança residencial, utilizando recursos da Domótica e explorando a integração de tecnologias como sensores de movimento, central de processamento e comunicação via aplicação para o telemóvel.

Ao longo deste relatório, serão abordados os principais conceitos teóricos relacionados à Domótica e à segurança residencial, a descrição detalhada dos componentes do sistema, a metodologia utilizada para o desenvolvimento do modelo proposto, as funcionalidades do sistema e os casos de uso, além dos resultados e discussões obtidos durante a implementação e teste do protótipo.

Palavras-chave: Domótica, segurança residencial, protótipo, sistema inteligente, aplicação para o telemóvel.

Abstract

The document presents the development of a prototype of an Intelligent System for low-cost residential security, in the context of Domotics. The main goal of the project is to implement a system that offers security, comfort, and total communication in a residential space.

The system is composed by different modules, including a sensor module for detecting possible invasions, a processing central for communication between the modules, data processing and storage in a database, an alarm module that emits a sound signal in case of invasion and a notification system to inform the user about the occurrence.

Besides, the project also foresees the development of a control application, which allows managing users, controlling the alarm module, and receiving intrusion notifications.

The prototype seeks to offer an efficient and accessible solution for residential security, using Domotics resources and exploring the integration of technologies such as motion sensors, central processing, and communication via mobile phone application.

Throughout this document, the main theoretical concepts related to Domotics and residential security, the detailed description of the system components, the methodology used for the development of the proposed model, the system functionalities and the use cases will be addressed, besides the results and discussions obtained during the prototype implementation and testing.

Keywords: Domotics, residential security, prototype, intelligent system, mobile phone application.

Índice

Dedicatória	3
Epígrafe	3
Resumo	4
Abstract	5
Índice	6
Lista de Fotografias/Ilustrações	10
Lista de Siglas e Acrónimos	11
1 Introdução.....	12
1.1 Descrição do Problema	12
1.2 Metodologia do relatório.....	13
1.3 Descrição geral do projeto	14
1.4 Objetivos	15
1.4.1 Objetivo do relatório.....	15
1.4.2 Objetivo gerais	15
1.4.3 Objetivos específicos	16
1.5 Justificativa	16
1.6 Estrutura do trabalho.....	17
2 Fundamentação Teórica	19
2.1 Domótica.....	19
2.2 Sistemas de Segurança Residencial	20
2.3 IoT.....	20
3 Descrição dos componentes e das suas funções.....	23
3.1 Arduino UNO-WIFI-REV2	23
3.1.1 Arduino	23
3.1.2 C++	24
3.1.3 Funções e recursos	25

3.1.4	Especificações técnicas.....	25
3.2	Sensor HC-SR04	26
3.2.1	Princípio de funcionamento.....	26
3.2.2	Utilização no projeto de segurança residencial.....	26
3.3	Câmara OV-7670.....	27
3.3.1	Características e capacidades	27
3.3.2	Integração com o projeto de segurança residencial.....	27
3.4	Alarme Buzzer.....	28
3.4.1	Propósito e funcionamento	28
3.4.2	Utilização no contexto do projeto	28
3.5	Telefone e aplicação móvel.....	29
3.5.1	Aplicação móvel – Flutter	29
3.5.2	Utilização na aplicação de segurança residencial	30
3.5.3	Integração com outros componentes	30
3.6	Jumper Wires (Fios de Ligação).....	31
3.7	Breadboard	32
3.8	USB-B.....	33
3.9	LEDs	35
3.10	Computador	36
3.11	Base de Dados com Docker e PostgreSQL	38
3.11.1	Docker	38
3.11.2	PostgreSQL	39
3.11.3	Configuração do ambiente de base de dados	40
3.11.4	Funções e importância da base de dados.....	40
3.12	API em Node.js.....	41
3.12.1	API.....	41
3.12.2	Node.js.....	42

3.12.3	Papel da API no projeto.....	43
3.12.4	Funcionalidades e endpoints.....	43
3.13	Ambiente de Desenvolvimento (IDE).....	44
3.13.1	Utilização do IntelliJ para desenvolvimento da API e aplicação:	44
3.13.2	Utilização do VSCode para desenvolvimento do script para o	
Arduino:	45	
3.14	Mecanismo de Comunicação do Sistema:	46
3.14.1	Wifi entre Arduino e API:	46
3.14.2	API entre Aplicação - Sockets:	46
3.14.3	API entre Base de Dados - Sockets:.....	46
4	Metodologia do modelo proposto	48
4.1	Apresentação geral do modelo proposto.....	48
4.2	Desenvolvimento do Modelo Proposto.....	49
4.2.1	Construção da Maquete.....	49
4.2.2	Implementação física do Protótipo.....	49
4.2.3	Implementação da Aplicação.....	50
4.2.4	Integração com a Aplicação.....	51
4.3	Histórico.....	52
4.4	Configuração do Ambiente de Desenvolvimento.....	53
4.5	Integração dos Componentes e Comunicação entre Eles	54
4.6	Desenvolvimento da API em Node.js.....	55
4.7	Integração com a Base de Dados PostgreSQL	56
5	Funcionalidades e Casos de Uso	57
5.1	5.1 Detecção de Intrusão usando o Sensor HC-SR04	57
5.2	Captura de imagens com a câmara OV-7670.....	58
5.3	Acionamento do alarme buzzer	58
5.4	Notificação no telefone	59

5.5	Armazenamento de dados na base de dados	59
5.6	Controle Remoto do Sistema de Segurança	60
6	Resultados e Discussões	61
6.1	Testes e Validação das Funcionalidades.....	61
6.2	Desempenho do Sistema	62
6.3	Limitações e Possíveis Melhorias	62
7	Considerações Finais	64
7.1	Recapitulação dos Objetivos Alcançados	64
7.2	Lições Aprendidas	64
7.3	Recomendações para Projetos Futuros	65
8	Conclusões.....	67
9	Bibliografia	68
10	Anexos.....	73
10.1	Manual de Instalação.....	73
10.2	Configuração e Utilização da Aplicação	73
10.2.1	Configuração da Aplicação.....	73
10.2.2	Utilização da Aplicação.....	75
10.3	Código fonte	81

Lista de Fotografias/Ilustrações

Figura 1 - Arduino UNO WIFI REV2.....	23
Figura 2 - Logotipo C++	25
Figura 3 - Sensor de distância HC-SR04.....	26
Figura 4 - Câmara OV-7670.....	27
Figura 5 - Alarme Buzzer	28
Figura 6 - Telemóvel.....	29
Figura 7 - Logotipo Flutter.....	30
Figura 8 - Fios de Ligação.....	31
Figura 9 - Breadboard	33
Figura 10 - Cabo de ligação USB-B	34
Figura 11 - LED.....	36
Figura 12 - Computador.....	38
Figura 13 - Logotipo Docker	39
Figura 14 - Logotipo PostgreSQL.....	40
Figura 15 - Funcionamento de uma API	41
Figura 16 - Logotipo node.js	43
Figura 17 - Logotipo IntelliJ	45
Figura 18 - Logotipo VS Code	46
Figura 19 - Arquitetura do Projeto.....	49
Figura 20 - API.....	73
Figura 21 - Flutter.....	74
Figura 22 - Base de Dados	74
Figura 23 - Criar Utilizador.....	75
Figura 24 - Fazer login.....	76
Figura 25 - Home da aplicação.....	77
Figura 26 - Historico do Alarme.....	78
Figura 27 - Pagina de gestão do admin	79
Figura 28 - Página de perfil de cada utilizador	80
Figura 29 - Validação do login	81
Figura 30 - Buscar alarmes.....	81
Figura 31 - Adicionar alarmes	82
Figura 32 - Desligar alarmes	82
Figura 33 - Buscar todos os alarmes	82
Figura 34 - Login.....	83
Figura 35 - logout	83
Figura 36 - criar user.....	83
Figura 37 - apagar user.....	84
Figura 38 - atualizar user.....	84
Figura 39 - Ficheiro SQL (1).....	85
Figura 40 - Ficheiro SQL (2).....	85
Figura 41 - Ficheiro SQL (3).....	85
Figura 42 - Pedido para autenticação do user	86
Figura 43 - Código Arduino	86
Figura 44 - Função main do Java.....	86
Figura 45 - Pedido para troca de password feita pelo admin	86
Figura 46 - Exemplo do widget	86

Lista de Siglas e Acrónimos

API	Application Programming Interface
IoT	Internet of Things
LPWAN	Low Power Wide Area Networks
IDE	Integrated Development Environment
LED	Light-Emitting Diode
UART	Universal Asynchronous Receiver/Transmitter
SPI	Serial Peripheral Interface
I2C	Inter-Integrated Circuit
VGA	Video Graphics Array
RGB	Red, Green, and Blue
YUV	Luminance and Chrominance
CMOS	Complementary Metal-Oxide Semiconductor
UI	User Interface
MM	Macho-Macho
FF	Fêmea-Fêmea
MF	Macho-Fêmea
CI	Circuito Integrado
PCB	Printed Circuit Board
USB	Universal Serial Bus
SMD	Surface-Mounted Device
IR	Infrared
PWM	Pulse With Modulation
CPU	Central Processing Unit
HDD	Hard Disk Drive
SSD	Solid-State Drive
GPU	Graphics Processing Unit
SQL	Structed Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol
HTTP	Hypertext Transfer Protocol

1 Introdução

Apresenta-se uma visão geral sobre os sistemas inteligentes para segurança residencial, abordando o seu propósito e a sua relevância tanto em nível global quanto em Portugal. Esses sistemas têm se tornado uma solução cada vez mais procurada para proteger residências contra ameaças e proporcionar maior tranquilidade aos moradores.

Os sistemas inteligentes para segurança residencial têm como principal objetivo garantir a proteção do lar, oferecendo recursos avançados de detecção e supervisão. Por meio da integração de tecnologias como sensores, câmaras, alarmes e sistemas de comunicação, esses sistemas são capazes de identificar atividades suspeitas e acionar medidas de segurança adequadas.

Em Portugal, assim como em muitos outros países, a adoção desses sistemas tem crescido significativamente. Os proprietários de residências reconhecem a importância de garantir a segurança dos seus lares e buscam soluções eficientes para esse fim. Os sistemas inteligentes oferecem a vantagem de proporcionar proteção contínua e supervisão em tempo real, contribuindo para a tranquilidade dos moradores.

Em âmbito global, os sistemas inteligentes para segurança residencial também têm sido amplamente adotados. Países ao redor do mundo reconhecem a necessidade de investir em medidas de segurança avançadas para proteger as suas comunidades. Esses sistemas desempenham um papel fundamental na prevenção de intrusões, incêndios e outros incidentes indesejados, ajudando a preservar a integridade das residências e garantindo a segurança dos moradores.

Ao considerar os custos, é importante observar que os sistemas inteligentes para segurança residencial oferecem uma variedade de opções, desde soluções mais acessíveis até sistemas mais sofisticados e abrangentes. Os custos podem variar conforme a complexidade dos recursos e das tecnologias utilizadas. No entanto, é importante destacar que os benefícios proporcionados por esses sistemas superam os investimentos realizados, garantindo a proteção do patrimônio e a segurança dos moradores.

Além da proteção contra intrusões, os sistemas inteligentes para segurança residencial oferecem uma série de benefícios adicionais. Eles possibilitam o supervisão remoto da residência, permitindo que os moradores tenham controle e acesso às informações de segurança em tempo real. Além disso, a integração desses sistemas com dispositivos inteligentes e a possibilidade de receber notificações instantâneas em caso de atividades suspeitas proporcionam maior comodidade e tranquilidade.

1.1 Descrição do Problema

A segurança residencial é uma preocupação crescente na sociedade atual, devido ao aumento dos índices de criminalidade. As residências são frequentemente alvos de invasões e roubos, causando danos materiais e emocionais aos moradores. Além disso, a sensação de insegurança pode afetar negativamente a qualidade de vida das pessoas.

Diante desse cenário, é fundamental encontrar soluções eficientes e tecnologicamente avançadas para garantir a segurança das residências. Um sistema de segurança residencial automatizado e inteligente pode ser a resposta para esse problema. Esse tipo de sistema,

conhecido como domótica utiliza dispositivos eletrônicos e de comunicação para monitorizar e controlar diversos aspetos da residência, incluindo a segurança.

O sistema proposto oferecerá maior tranquilidade aos moradores, permitindo que eles supervisionem a sua residência de forma remota e recebam alertas em tempo real em caso de atividade suspeita. Além disso, o sistema poderá armazenar dados sobre eventos de segurança, possibilitando análises posteriores e auxiliando nas investigações em caso de incidentes.

Com a implementação desse sistema de segurança residencial baseado em domótica, espera-se reduzir os índices de invasões e roubos em residências, proporcionando maior proteção e conforto aos moradores. [4] [5]

1.2 Metodologia do relatório

A metodologia utilizada neste projeto segue uma abordagem iterativa e incremental, permitindo a evolução progressiva do sistema de segurança residencial. De seguida enumeramos as principais etapas da metodologia utilizada:

- **Planeamento:** O planeamento do projeto consiste numa abordagem estruturada, que visa estabelecer as bases para o desenvolvimento eficiente e eficaz do projeto. Por meio do Project, foi possível criar um cronograma detalhado, definindo as atividades a serem realizadas, as suas dependências, as estimativas de duração e alocação de recursos.
- **Levantamento de requisitos:** Nesta etapa, foram realizadas pesquisas e análises para compreender as necessidades e expectativas do sistema de segurança residencial. Foram identificados os requisitos funcionais e não funcionais, bem como os casos de uso do sistema.
- **Seleção dos componentes:** Com base nos requisitos levantados, foram selecionados os componentes eletrônicos adequados para o projeto, levando em consideração a compatibilidade, a disponibilidade e as funcionalidades oferecidas. Os componentes escolhidos incluem o Arduino UNO-WIFI-REV2, o sensor HC-SR04, a câmara OV-7670, o alarme buzzer e o telefone.
- **Configuração do ambiente de desenvolvimento:** Nesta etapa, o ambiente de desenvolvimento foi configurado para permitir a integração e a comunicação entre os componentes eletrônicos. Foram instaladas as bibliotecas necessárias para o Arduino e para a câmara, além de serem configuradas as conexões de hardware e software adequadas.
- **Implementação da API em Node.js:** foi desenvolvida uma API em Node.js para gerir a comunicação entre os componentes eletrônicos. A API foi responsável por receber os dados do sensor HC-SR04, acionar o alarme buzzer, capturar imagens com a câmara OV-7670, enviar notificações para o telefone e armazenar os dados relevantes na base de dados PostgreSQL.
- **Integração dos componentes:** Os componentes eletrônicos foram integrados de acordo com o circuito e as conexões definidas. A comunicação entre o Arduino, o sensor HC-SR04, a câmara OV-7670, o alarme buzzer, o telefone e a API em

Node.js foi estabelecida, permitindo o fluxo de informações e interações entre os componentes.

- Testes e validação: Foram realizados testes para validar as funcionalidades do sistema de segurança residencial. Os testes incluíram a detecção de intrusão pelo sensor HC-SR04, a captura de imagens pela câmara OV-7670, o acionamento do alarme buzzer, a notificação no telefone e o armazenamento de dados na base de dados PostgreSQL. Os resultados dos testes foram analisados e quaisquer problemas encontrados foram corrigidos.

A metodologia utilizada permitiu o desenvolvimento progressivo do sistema de segurança residencial, garantindo a qualidade e a funcionalidade do sistema final. As iterações e a abordagem incremental possibilitaram ajustes e aprimoramentos ao longo do processo, resultando num sistema mais eficiente. [6] [7]

1.3 Descrição geral do projeto

O projeto de segurança residencial consiste em desenvolver um sistema que tem como objetivo garantir a proteção e a supervisão de uma residência por meio da utilização de componentes eletrônicos e tecnologias específicas. O objetivo principal é detetar intrusões e acionar um alarme, além de fornecer notificações ao proprietário sobre eventos de segurança por meio de um telefone ou outro dispositivo móvel.

O sistema é composto por diversos componentes interligados, incluindo o Arduino UNO-WIFI-REV2, o sensor HC-SR04, a câmara OV-7670, o alarme buzzer, o telefone e uma base de dados PostgreSQL. Cada componente desempenha uma função específica no projeto e está integrado ao sistema por meio de uma API em Node.js.

Estes componentes abrangem as seguintes funcionalidades e aspetos:

- Cérebro do sistema: O arduino UNO-WIFI-REV2 é responsável por receber e processar os dados dos sensores e controlar as ações dos outros componentes.
- Detecção de intrusão: O sistema deve ser capaz de detetar a presença de intrusos na residência por meio do sensor HC-SR04. O sensor mede a distância entre ele e um objeto próximo, permitindo identificar a aproximação não autorizada de pessoas.
- Captura de imagens: A câmara OV-7670 é utilizada para captar imagens quando uma intrusão é detetada. As imagens são registadas e podem ser visualizadas posteriormente para fins de investigação ou monitoramento.
- Acionamento do alarme: Quando uma intrusão é identificada, o alarme buzzer é acionado para emitir um som audível, alertando os residentes e afastando possíveis invasores.
- Notificação no telefone: O sistema deve ser capaz de enviar notificações para o telefone do proprietário, informando sobre eventos de segurança, como a detecção de intrusão ou acionamento do alarme. Isso permite que o proprietário tome medidas imediatas.
- Armazenamento de dados: O sistema deve armazenar dados relevantes, como registos de intrusões e informações do sistema, numa base de dados PostgreSQL. Isso permite a consulta e análise posterior dos dados coletados.

- Integração dos componentes: Os componentes eletrônicos, incluindo o Arduino, o sensor, a câmara, o alarme, o telefone e a base de dados, devem ser integrados para permitir a comunicação e a interação entre eles, garantindo o correto funcionamento do sistema.

A integração entre os componentes é realizada por meio de uma API em Node.js, que gere a comunicação entre o Arduino, os sensores, a câmara, o alarme, o telefone e a base de dados. Esta API permite o fluxo de informações entre os componentes, bem como o controlo das ações e a troca de dados relevantes para o funcionamento do sistema.

O projeto tem como objetivo fornecer um sistema de segurança residencial eficiente, confiável e personalizável, permitindo ao proprietário monitorar e proteger a sua residência de forma mais segura e conveniente. [8] [9] [10] [11]

1.4 Objetivos

É essencial definir claramente os objetivos que queremos alcançar. Os objetivos direcionam o trabalho, estabelecendo metas específicas que orientam as ações e a tomada de decisões ao longo do processo.

1.4.1 Objetivo do relatório

Este relatório tem como objetivo fornecer uma descrição detalhada de um projeto de segurança residencial que utiliza componentes eletrônicos e tecnologias específicas. Serão apresentados o escopo do projeto, a metodologia utilizada, a descrição dos componentes e as suas funcionalidades, a arquitetura do projeto, a implementação, as funcionalidades e casos de uso, os resultados obtidos e uma conclusão.

O relatório servirá como um guia para entender o funcionamento do sistema, as suas funcionalidades e o contexto em que foi desenvolvido.

1.4.2 Objetivo gerais

O objetivo geral deste trabalho é criar um sistema de segurança residencial completo e funcional, utilizando uma maquete como representação física de um ambiente residencial. O sistema será desenvolvido por meio da programação e implementação de dispositivos IoT, para monitorar e proteger a residência contra possíveis invasões e eventos indesejados.

Para alcançar esse objetivo, serão utilizados diversos componentes e tecnologias, como sensores de movimento, câmaras de segurança, alarmes sonoros e visuais, além de ‘interfaces’ de controlo.

A programação dos dispositivos IoT será essencial para que o sistema de segurança funcione de maneira eficiente. Serão utilizadas linguagens de programação adequadas, como Arduino ou Python, para desenvolver a lógica de deteção de intrusos, o controlo das câmaras, a ativação do alarme e a comunicação entre os diferentes componentes do sistema.

Além disso, a criação de uma ‘interface’ de controlo intuitiva e acessível será um aspeto importante desse objetivo geral. Essa ‘interface’ permitirá aos utilizadores controlar o sistema

de segurança, ativar ou desativar os dispositivos, receber notificações e configurar as opções de segurança de forma simples e prática.

No final, o objetivo geral é alcançar um sistema de segurança residencial que proporcione proteção efetiva para a maquete, simulando as funcionalidades e benefícios que um sistema real poderia oferecer. Esse objetivo geral visa proporcionar uma experiência de aprendizado e demonstração dos conceitos de segurança residencial e IoT, destacando a importância desses sistemas na proteção de residências contra possíveis riscos e ameaças. [12]

1.4.3 Objetivos específicos

Os objetivos específicos são responsáveis pelos resultados concretos que o projeto pretende atingir e contribuem para a realização do objetivo geral. Logo, vamos enumerar os seguintes objetivos específicos que achamos necessários para alcançarmos o nosso objetivo geral:

- Desenvolver um sistema de segurança residencial inteligente que seja capaz de se comunicar com uma aplicação para smartphone, permitindo aos utilizadores supervisionar e controlar o seu sistema de segurança de forma remota.
- Implementar o sistema de segurança residencial inteligente e a aplicação para smartphone num ambiente de teste, de modo a avaliar o desempenho, a eficiência e a facilidade de uso do sistema.
- Realizar testes de campo para verificar a efetividade do sistema de segurança residencial inteligente com comunicação com a aplicação para smartphone, avaliando a sua capacidade de proteger a residência e fornecer informações úteis aos utilizadores. [12]

1.5 Justificativa

A justificativa para a realização desse projeto reside na importância crescente da segurança residencial na nossa sociedade. Com o aumento dos índices de criminalidade e invasões domiciliares, é fundamental que os moradores tenham acesso a sistemas de segurança eficientes e confiáveis para proteger as suas residências e garantir a segurança dos seus entes queridos e dos seus bens.

A criação de um sistema de segurança residencial utilizando uma maquete e dispositivos IoT oferece diversas vantagens e oportunidades. Em primeiro lugar, permite uma abordagem prática e *hands-on* para a compreensão e aplicação dos conceitos de segurança residencial e IoT. A maquete oferece uma representação física tangível de uma residência, facilitando a visualização e compreensão do funcionamento do sistema de segurança.

Além disso, a utilização de dispositivos IoT proporciona uma série de benefícios, como a capacidade de monitorar e controlar o sistema de segurança remotamente, por meio de dispositivos móveis. Isso traz conveniência e flexibilidade aos utilizadores, que podem estar sempre cientes do status de segurança da sua residência, mesmo quando estão longe de casa.

Outra justificativa para a realização desse trabalho é a oportunidade de explorar e aplicar conhecimentos relacionados à programação de dispositivos IoT. A programação dos sensores de movimento, câmaras e alarmes que nos permitira desenvolver habilidades em linguagens de

programação, como Arduino ou Python, além de compreenderem os princípios de integração e comunicação entre os dispositivos.

Além disso, este trabalho oferece uma oportunidade de aprendizagem interdisciplinar, combinando conceitos de engenharia, tecnologia da informação e segurança.

1.6 Estrutura do trabalho

Este trabalho está organizado da seguinte maneira:

- **Introdução:** Neste tópico, é feita uma apresentação inicial do trabalho, destacando a descrição do problema a ser abordado, os objetivos do projeto, tanto o objetivo geral quanto os objetivos específicos. Além disso, é fornecida uma justificativa para a realização do trabalho e uma visão geral da estrutura do documento.
- **Fundamentação Teórica:** Nessa seção, é abordada a base teórica que sustenta o projeto. É apresentado o conceito de domótica, enfatizando a importância desse campo para a automação residencial e a segurança. Também são mencionados termos como sistemas de segurança residencial, IoT e são apresentados alguns casos de uso.
- **Descrição dos Componentes e as suas Funções:** Aqui, são detalhados os componentes físicos utilizados no projeto, as suas características e as suas respectivas funções dentro do sistema.
- **Metodologia do Modelo Proposto:** Nessa seção, é apresentada uma visão geral do modelo proposto para o projeto. É descrita a arquitetura do sistema, incluindo os componentes físicos e as suas interações.
- **Funcionalidades e Casos de Uso:** Aqui, são destacadas as principais funcionalidades do sistema desenvolvido. Também são apresentados casos de uso específicos, demonstrando como o sistema pode ser aplicado em diferentes situações.
- **Resultados e Discussões:** Nessa seção, são apresentados os resultados obtidos a partir da implementação do projeto e dos testes realizados. São discutidos o desempenho do sistema, eventuais limitações encontradas e possíveis melhorias que podem ser realizadas no futuro. Essa análise crítica dos resultados permite uma reflexão sobre a eficácia e a aplicabilidade do projeto.
- **Conclusões:** Nesse último tópico, são apresentadas as conclusões finais do trabalho. São recapitulados os objetivos alcançados, destacando as principais contribuições do projeto. Além disso, são mencionadas as lições aprendidas durante a realização do trabalho e são feitas recomendações para projetos futuros, visando aprimorar e expandir o sistema desenvolvido.

- Bibliografia: Aqui são listadas as referências bibliográficas utilizadas no trabalho.
- Apêndices: Essa secção contém informações complementares e detalhes adicionais que não foram incluídos no corpo principal do trabalho, mas que podem ser relevantes para o entendimento completo do projeto.
- Anexos: Nessa parte, são incluídos materiais adicionais, como imagens, diagramas, tabelas, códigos-fonte ou outros elementos que complementem o trabalho e forneçam informações extras para os leitores.

2 Fundamentação Teórica

Por meio dessa fundamentação teórica, buscamos fornecer uma base sólida de conhecimento e compreensão, a fim de fundamentar as etapas seguintes do nosso trabalho, como a descrição dos componentes, a metodologia adotada, as funcionalidades desenvolvidas, os resultados obtidos e as conclusões tiradas.

2.1 Domótica

A domótica refere-se à automação residencial, que é o uso de tecnologia para controlar e automatizar as funções e dispositivos de uma casa. A palavra "domótica" vem da junção das palavras "domus" (casa, em latim) e "robótica", indicando o uso de sistemas automatizados para tornar uma casa mais eficiente, confortável e segura.

Através da domótica, é possível integrar diversos sistemas e dispositivos eletrônicos numa rede centralizada, permitindo o controle e a monitorização remota de várias funções da casa.

Existem várias tecnologias utilizadas na domótica, incluindo:

- **Redes de comunicação:** São usadas redes de comunicação como Wi-Fi e Bluetooth para conectar dispositivos numa rede doméstica.
- **Sensores:** Sensores de movimento e outros são usados para detetar as condições do ambiente e fornecer informações para o sistema de automação.
- **Atuadores:** São dispositivos responsáveis por executar as ações desejadas, como acionar luzes, abrir cortinas, ligar aparelhos eletrônicos, entre outros.
- **Controladores e interfaces:** Podem ser painéis de controle fixos, smartphones, tablets ou assistentes de voz como Amazon Echo ou Google Home, que permitem aos utilizadores controlar e monitorizar os dispositivos e sistemas automatizados.

Alguns exemplos de aplicação da domótica incluem:

- **Controle de iluminação:** É possível programar a iluminação da casa para se ajustar automaticamente às necessidades, ligando e desligando luzes em determinados horários ou em resposta a sensores de movimento.
- **Segurança:** Sistemas de alarme e câmaras de segurança podem ser integrados ao sistema domótico, permitindo monitorização e controle remoto da segurança da casa.

- Gestão de energia: Com a domótica, é possível monitorizar e controlar o consumo de energia em tempo real, permitindo otimizar o seu uso e reduzir os custos de eletricidade.

A domótica oferece uma série de benefícios, como maior conforto, segurança e eficiência energética. No entanto, é importante considerar aspetos de segurança cibernética ao implementar sistemas domóticos, garantindo que a rede doméstica esteja protegida contra possíveis ataques. [5] [13] [14]

2.2 Sistemas de Segurança Residencial

Um sistema de segurança residencial é um conjunto de dispositivos e tecnologias projetados para proteger residências contra intrusões e outras ameaças à segurança. Esse tipo de sistema utiliza diversos componentes, como sensores, câmaras, alarmes e sistemas de comunicação, para detetar e responder a eventos de segurança.

A principal função de um sistema de segurança residencial é garantir a proteção da residência e a segurança dos moradores. Isso é feito por meio da deteção de intrusões, monitoramento de áreas externas e internas, notificação de eventos de segurança, acionamento de alarmes e, em alguns casos, integração com serviços de segurança externos.

Os sistemas de segurança residencial podem ser projetados para atender a diferentes necessidades e níveis de segurança. Eles podem incluir componentes como sensores de movimento, sensores de abertura de portas e janelas, câmaras de vigilância, alarmes sonoros e visuais, controle de acesso, sistemas de monitoramento remoto e integração com dispositivos móveis e sistemas de automação residencial.

A implementação de um sistema de segurança residencial pode variar de acordo com as necessidades e o orçamento de cada residência. Desde soluções mais simples e autónomas até sistemas mais avançados e integrados, é importante escolher os componentes e tecnologias adequados para atender às necessidades específicas de segurança de cada residência. [15]

2.3 IoT

IoT, ou Internet das Coisas (do inglês *Internet of Things*), é um conceito que se refere à conexão e comunicação de objetos físicos através da Internet. Esses objetos podem ser praticamente qualquer coisa, desde dispositivos eletrónicos, eletrodomésticos, veículos, sensores e até mesmo roupas ou acessórios pessoais.

A ideia central do IoT é capacitar os objetos do quotidiano a coletar e trocar dados entre si, além de interagir com os utilizadores e com o ambiente ao seu redor. Essa interconexão permite a criação de um ecossistema inteligente, no qual os dispositivos podem ser monitorados, controlados e automatizados remotamente.

A tecnologia IoT é impulsionada por três elementos principais:

- Dispositivos IoT: São os objetos físicos que possuem sensores, atuadores e conectividade para interagir com o ambiente e transmitir dados. Esses dispositivos

podem variar desde pequenos sensores de temperatura até veículos autônomos complexos.

- **Conectividade:** A conectividade é essencial para permitir a comunicação entre os dispositivos IoT e a Internet. Isso pode ser alcançado por meio de diferentes tecnologias, como Wi-Fi, Bluetooth, redes móveis, redes de área ampla de baixa potência (LPWAN) e até mesmo a próxima geração de redes 5G.
- **Plataforma e Aplicações:** As plataformas IoT são responsáveis pela gestão dos dispositivos, coleta de dados, processamento, armazenamento e análise dos dados gerados pelos dispositivos conectados.

O IoT tem a capacidade de trazer uma série de benefícios e oportunidades. Por exemplo:

- **Eficiência e automação:** Os dispositivos IoT podem automatizar tarefas e processos, melhorando a eficiência operacional e reduzindo custos. Por exemplo, os sensores podem monitorar o consumo de energia numa casa e ajustar automaticamente a iluminação e a temperatura para economizar energia.
- **Melhoria na qualidade de vida:** O IoT pode trazer benefícios para a saúde, segurança e bem-estar das pessoas. Por exemplo, os dispositivos médicos conectados podem monitorar os sinais vitais dos pacientes e enviar alertas em caso de emergência.
- **Tomada de decisões baseada em dados:** O IoT gera grandes quantidades de dados que podem ser utilizados para obter perspectivas valiosas. As análises desses dados podem levar a decisões mais informadas e a melhorias em produtos e serviços.
- **Oportunidades de negócio:** O IoT cria um mercado para desenvolvedores, empresas e empreendedores que podem criar produtos e serviços inovadores para atender às necessidades desse ecossistema conectado.

No entanto, o IoT também apresenta desafios, como a segurança e privacidade dos dados, a interoperabilidade entre diferentes dispositivos e plataformas, e a escalabilidade da infraestrutura necessária para suportar o crescimento de dispositivos conectados.

No nosso caso, falando do nosso projeto, a IoT desempenha um papel fundamental nos sistemas de segurança residencial modernos. Ela permite a interconexão e comunicação entre os dispositivos utilizados no sistema, possibilitando a troca de informações em tempo real e o controle remoto dos componentes.

No contexto de um sistema de segurança residencial, a IoT permite a integração de sensores, câmaras, alarmes e outros dispositivos para monitorar e proteger a residência de maneira mais eficiente. Os dispositivos conectados podem coletar e transmitir dados para uma central de controle ou para os dispositivos móveis dos proprietários, permitindo que eles monitorem e controlem a segurança das suas residências a qualquer momento e de qualquer lugar.

Com a IoT, é possível obter informações precisas e atualizadas sobre eventos de segurança, como detecção de movimento. Esses dados podem ser analisados em tempo real para acionar alarmes, notificar os proprietários e acionar medidas de segurança apropriadas.

A IoT também possibilita a atualização e o aprimoramento contínuo dos sistemas de segurança residencial por meio de atualizações de software e firmware, garantindo que o

sistema esteja sempre atualizado e preparado para enfrentar as ameaças de segurança mais recentes. [16] [17] [18] [19] [20]

3 Descrição dos componentes e das suas funções

Este tópico tem como objetivo apresentar e descrever os principais componentes utilizados no projeto de segurança residencial. Cada componente desempenha um papel crucial no sistema, contribuindo para a detecção de intrusões, a captura de imagens, o acionamento do alarme e a notificação no telemóvel. A seguir, serão detalhados o Arduino UNO-WIFI-REV2, o sensor HC-SR04, a câmara OV-7670, o alarme *buzzer*, o telemóvel, *jumper wires*, *breadboard*, USB-B, LEDs, computador e a base de dados com Docker e PostgreSQL.

3.1 Arduino UNO-WIFI-REV2

O Arduino UNO-WIFI-REV2 é uma placa de desenvolvimento baseada no microcontrolador ATmega4809, com recursos adicionais de conectividade Wi-Fi. Essa placa é uma versão aprimorada do Arduino UNO, oferecendo a flexibilidade e facilidade de uso do Arduino combinadas com a capacidade de comunicação sem fio. [21]

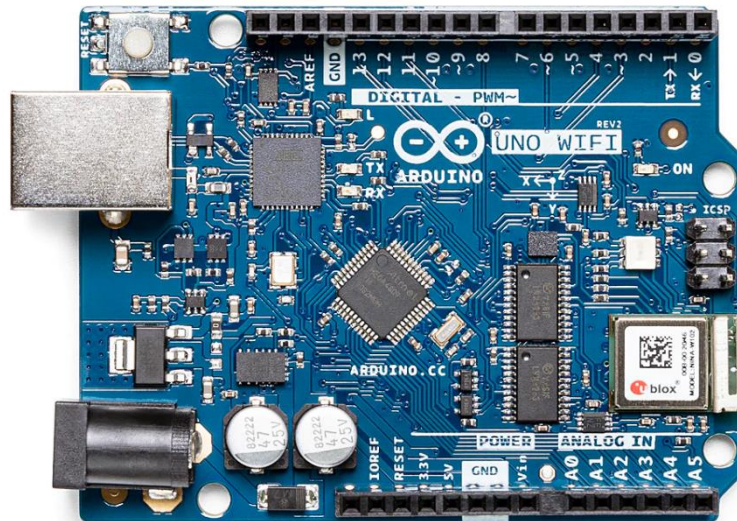


Figura 1 - Arduino UNO WIFI REV2

Fonte: <https://shorturl.at/bgsF6>

3.1.1 Arduino

O Arduino é uma plataforma de prototipagem eletrônica de código aberto, desenvolvida por Massimo Banzi e David Cuartielles em 2005, que permite a criação de projetos interativos. Ele consiste numa placa de circuito impresso com um microcontrolador embutido e uma interface de programação fácil de usar. O objetivo do Arduino é fornecer uma maneira acessível e simplificada para que iniciantes e entusiastas possam criar projetos eletrônicos interativos.

A placa Arduino é alimentada por um microcontrolador, que é basicamente um pequeno computador num único chip.

O Arduino utiliza uma linguagem de programação baseada em C/C++, que é bastante amigável para iniciantes. No nosso projeto, utilizámos a linguagem de programação C++. O

ambiente de desenvolvimento integrado (IDE) do Arduino oferece uma interface gráfica simples para escrever e carregar o código no microcontrolador da placa Arduino. Ele também possui uma biblioteca extensa que simplifica a interação com os componentes eletrônicos, como sensores, atuadores e displays.

Uma das principais características do Arduino é a facilidade de conexão com componentes eletrônicos externos. Ele possui pinos de entrada/saída que podem ser usados para conectar uma variedade de sensores, botões, LEDs, motores e outros dispositivos. Essa flexibilidade permite que os usuários criem projetos interativos e personalizados segundo as suas necessidades.

Além disso, o Arduino tem uma comunidade ativa de desenvolvedores e entusiastas que compartilham projetos, tutoriais e bibliotecas. Isso facilita o aprendizado e a colaboração, além de possibilitar a expansão das funcionalidades do Arduino com base no trabalho de outras pessoas.

O Arduino é amplamente utilizado em várias áreas, como automação residencial, robótica, arte interativa, IoT (Internet das Coisas), educação e muitos outros domínios. É uma plataforma acessível para aprender eletrônica e programação, permitindo que as pessoas transformem as suas ideias criativas em realidade. [22] [23]

3.1.2 C++

C++ é uma linguagem de programação de propósito geral, conhecida por ser uma extensão da linguagem C.

A linguagem C++ combina recursos de alto nível, como abstração de dados, encapsulamento, herança e polimorfismo, com recursos de baixo nível, como acesso direto à memória e manipulação de ponteiros (um ponteiro é uma variável capaz de armazenar um endereço de memória ou o endereço de outra variável).

Uma das principais características do C++ é o suporte à programação orientada a objetos (POO). Ela permite que os desenvolvedores organizem o seu código em classes, que encapsulam dados e comportamentos relacionados. A herança é outro recurso importante, permitindo que classes derivadas herdem características de classes base, promovendo a reutilização de código.

O C++ também suporta polimorfismo, que permite que objetos de diferentes classes sejam tratados de forma polimórfica, ou seja, usando a mesma interface, mas com comportamentos específicos de cada classe. Isso é alcançado por meio de funções virtuais e classes abstratas.

C++ é amplamente utilizado em várias áreas, como desenvolvimento de jogos, sistemas operativos, software de alto desempenho, aplicativos de desktop, sistemas embarcados e muito mais. É uma linguagem poderosa e versátil, mas também pode ser complexa de aprender e usar corretamente. [24] [25]



Figura 2 - Logotipo C++

Fonte: <https://shorturl.at/krDR1>

3.1.3 Funções e recursos

O Arduino UNO-WIFI-REV2 oferece uma variedade de funções e recursos que o tornam adequado para projetos de segurança residencial. Ele possui 6 portas analógicas e 14 portas digitais, permitindo a conexão de sensores, câmaras e outros dispositivos eletrônicos. Além disso, possui uma interface UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface) e I2C (Inter-Integrated Circuit) para a comunicação com outros dispositivos.

A principal característica do Arduino UNO-WIFI-REV2 é a conectividade Wi-Fi embutida. Isso permite que o microcontrolador se conecte a redes sem fio, como redes domésticas, facilitando a comunicação e o envio de dados pela Internet. Com essa capacidade, é possível enviar notificações, transmitir informações de segurança e interagir com outros dispositivos conectados na rede. [26]

3.1.4 Especificações técnicas

As especificações técnicas do Arduino UNO-WIFI-REV2 incluem:

- Microcontrolador: ATmega4809
- Clock: 16 MHz
- Memória Flash: 48 KB (dos quais 6 KB são usados pelo bootloader)
- SRAM: 6,144 KB
- EEPROM: 256 bytes
- Voltagem de operação: 5V
- Portas analógicas: 6
- Portas digitais: 14
- Comunicação: Wi-Fi 802.11b/g/n, UART, SPI, I2C

Essas especificações tornam o Arduino UNO-WIFI-REV2 uma plataforma versátil e poderosa para o desenvolvimento de projetos de segurança residencial, permitindo a integração de sensores, câmaras e outros dispositivos, além de fornecer conectividade sem fio para a comunicação com outros dispositivos e serviços online. [21] [26]

3.2 Sensor HC-SR04

O sensor HC-SR04 é um dispositivo amplamente utilizado para medir a distância entre o sensor e um objeto na sua proximidade. Baseado no princípio do ultrassom, ele emite sinais sonoros e mede o tempo que esses sinais levam para retornar após refletir num objeto. Essa medida de tempo é usada para calcular a distância entre o sensor e o objeto. [27]



Figura 3 - Sensor de distância HC-SR04

Fonte: <https://shorturl.at/cMZ49>

3.2.1 Princípio de funcionamento

O sensor HC-SR04 consiste num emissor de ultrassom (transdutor) e um recetor. O emissor emite pulsos ultrassônicos que se propagam no ambiente até encontrar um objeto sólido. Quando esses pulsos encontram o objeto, eles são refletidos e retornam ao sensor. O recetor recebe os pulsos refletidos e, com base no tempo de retorno, é possível determinar a distância do objeto.

O sensor HC-SR04 utiliza o tempo de ida e volta dos pulsos ultrassônicos para calcular a distância. Para isso, são necessárias duas operações: a primeira envolve o envio de um pulso de ativação para o emissor, que emite uma sequência de pulsos ultrassônicos. A segunda operação consiste em medir o tempo que o pulso leva para retornar ao recetor. A partir desse tempo, é possível calcular a distância usando a velocidade do som como referência. [27]

3.2.2 Utilização no projeto de segurança residencial

No contexto do projeto de segurança residencial, o sensor HC-SR04 desempenha um papel fundamental na deteção de intrusões. Ele é colocado numa posição estratégica para monitorar uma determinada área. Quando um objeto ou pessoa se aproxima dessa área, o sensor é capaz de detetar a presença e medir a distância relativamente ao objeto. Essa informação é usada para identificar uma possível intrusão.

Ao detetar uma distância abaixo de um limite pré-estabelecido, o sensor HC-SR04 aciona o sistema de segurança, como o acionamento de alarmes ou a captura de imagens. Essa

deteção de intrusão é essencial para a proteção da residência, pois permite identificar atividades suspeitas e acionar as medidas apropriadas para garantir a segurança dos moradores.

O sensor HC-SR04 é amplamente utilizado devido à sua precisão, baixo custo e facilidade de integração com microcontroladores como o Arduino UNO-WIFI-REV2. A sua utilização no projeto de segurança residencial contribui para a deteção eficaz de intrusões, fortalecendo a proteção e monitoramento da residência.

3.3 Câmara OV-7670

A câmara OV-7670 é uma câmara de baixo custo e tamanho compacto, amplamente utilizada em projetos que envolvem captura de imagens em tempo real. Ela oferece recursos básicos de captura de imagem e é facilmente integrada a microcontroladores. A sua integração eficaz com o sistema permite obter informações visuais adicionais, fortalecendo a deteção de intrusões e melhorando a capacidade de resposta diante de eventos suspeitos. [28] [29]



Figura 4 - Câmara OV-7670

Fonte: <https://shorturl.at/fsNU5>

3.3.1 Características e capacidades

A câmara OV-7670 possui várias características e capacidades que a tornam adequada para o projeto de segurança residencial. Ela possui uma resolução VGA (640x480 pixels) e é capaz de capturar imagens coloridas em tempo real. Além disso, ela suporta diferentes formatos de saída de imagem, como RGB (Red Green Blue) e YUV (Luminance and Chrominance).

A câmara é equipada com um sensor de imagem CMOS (Complementary Metal-Oxide-Semiconductor) que permite a captura de imagens com boa qualidade em diferentes condições de iluminação. Ela também possui uma interface de comunicação serial que facilita a integração com microcontroladores, como o Arduino UNO-WIFI-REV2. [28] [29]

3.3.2 Integração com o projeto de segurança residencial

No projeto de segurança residencial, a câmara OV-7670 desempenha um papel importante na obtenção de informações visuais sobre eventos suspeitos. Quando uma intrusão é detetada pelo sensor HC-SR04, a câmara é acionada para capturar imagens da área monitorada. Essas imagens podem ser usadas posteriormente para identificação de pessoas, análise de comportamento e registo de eventos.

A integração da câmara OV-7670 com o sistema de segurança residencial envolve a configuração da interface de comunicação com o microcontrolador. O microcontrolador envia comandos para a câmara, como iniciar a captura de imagens e transmitir os dados para processamento ou armazenamento. A partir das imagens capturadas, é possível tomar ações apropriadas, como notificar os utilizadores.

3.4 Alarme Buzzer

O alarme buzzer é um dispositivo sonoro utilizado em diversos projetos, incluindo o de segurança residencial. Ele é responsável por emitir um som audível quando uma intrusão é detetada ou quando uma situação de perigo é identificada. O alarme buzzer desempenha um papel crucial ao alertar os moradores sobre possíveis ameaças e contribui para dissuadir invasores. [30] [31]



Figura 5 - Alarme Buzzer

Fonte: <https://shorturl.at/arT29>

3.4.1 Propósito e funcionamento

O propósito do alarme buzzer é fornecer um aviso sonoro instantâneo em resposta a eventos específicos. Ele é ativado pelo sistema de segurança quando uma intrusão é detetada por meio do sensor HC-SR04 ou quando uma situação perigosa é identificada. O som emitido pelo alarme buzzer chama a atenção dos moradores e alerta sobre a presença de uma possível ameaça.

O funcionamento do alarme buzzer é relativamente simples. Ele consiste num dispositivo piezoelétrico que vibra quando uma corrente elétrica é-lhe aplicada. Essas vibrações geram ondas sonoras que resultam no som característico do buzzer. O controlo do buzzer é feito por meio do microcontrolador, que envia um sinal elétrico para acionar o dispositivo e produzir o som. [30] [31]

3.4.2 Utilização no contexto do projeto

No projeto de segurança residencial, o alarme buzzer é acionado quando uma intrusão é detetada pelo sensor HC-SR04 ou quando uma situação perigosa é identificada. Por exemplo, quando alguém se aproxima da área monitorada de forma suspeita ou quando é detetado um movimento não autorizado dentro da residência. O som emitido pelo buzzer alerta imediatamente os moradores sobre a ocorrência e permite que eles tomem as medidas apropriadas.

O buzzer pode ser integrado ao sistema de segurança de forma a trabalhar em conjunto com outros dispositivos, como o envio de notificações para o telefone dos moradores ou a gravação de imagens pela câmara.

A presença do alarme buzzer no projeto de segurança residencial contribui para a proteção efetiva do ambiente, fornecendo um aviso sonoro imediato em caso de intrusões ou perigos detetados. Ele aumenta a consciencialização dos moradores sobre eventos de segurança e ajuda a dissuadir possíveis invasores, proporcionando uma camada adicional de proteção para a residência.

3.5 Telefone e aplicação móvel

O telefone desempenha um papel essencial na aplicação de segurança residencial, proporcionando uma forma de comunicação direta e rápida entre o sistema de segurança e os moradores. Ele permite receber notificações instantâneas sobre eventos de segurança, bem como realizar ações remotas para garantir a proteção da residência.



Figura 6 - Telemóvel

Fonte: <https://shorturl.at/qHMW3>

3.5.1 Aplicação móvel – Flutter

O Flutter é um framework de código aberto desenvolvido pela Google, lançado em 2017, que permite a criação de aplicativos nativos para dispositivos móveis, web e desktop a partir de um único código-base. Ele utiliza a linguagem de programação Dart, também desenvolvida pela Google.

O objetivo principal do Flutter é facilitar o desenvolvimento de interfaces de utilizador (UI) bonitas, rápidas e fluidas. Ele fornece um conjunto abrangente de widgets personalizáveis que permitem a criação de interfaces ricas e responsivas. Além disso, o Flutter utiliza a técnica de renderização direta, o que significa que a interface é desenhada pixel a pixel, resultando numa aparência nativa e um desempenho de alta qualidade em várias plataformas.

O Flutter tem sido amplamente adotado por desenvolvedores e empresas para criar aplicativos móveis e web de alta qualidade.

No nosso caso, a aplicação móvel é desenvolvida utilizando o Flutter, garantindo compatibilidade multiplataforma e um desempenho otimizado. Isso permite que os moradores utilizem o aplicativo em dispositivos iOS e Android, proporcionando uma ampla cobertura de utilizadores. [32]



Figura 7 - Logotipo Flutter

Fonte: <https://shorturl.at/jBD08>

3.5.2 Utilização na aplicação de segurança residencial

No contexto do projeto de segurança residencial, o telefone é utilizado como um meio de comunicação entre o sistema de segurança e os moradores. Através de um aplicativo ou plataforma dedicada, o telefone recebe notificações em tempo real sobre eventos importantes, como detecção de intrusões, emergências ou alertas críticos.

Essas notificações podem incluir informações relevantes, como a área específica da residência onde ocorreu a detecção, imagens capturadas pela câmara, status de sensores ou outras informações importantes. Os moradores podem ser alertados por meio de notificações push no aplicativo, permitindo que eles tomem ações imediatas em resposta aos eventos de segurança.

Além disso, o telefone também pode ser utilizado para realizar ações remotas, como ativar ou desativar o sistema de segurança, acionar dispositivos de controle, visualizar câmaras ao vivo ou realizar o acesso remoto a funcionalidades do sistema. Isso oferece maior flexibilidade e controle aos moradores, permitindo que eles supervisionem e protejam a residência mesmo quando estão ausentes.

3.5.3 Integração com outros componentes

O telefone é integrado ao sistema de segurança residencial por meio de uma conexão com a rede local ou a Internet. Ele comunica-se com os outros componentes do sistema, como o Arduino UNO-WIFI-REV2, a câmara e o banco de dados, para receber e enviar informações relevantes. Essa integração permite que o telefone receba notificações em tempo real e interaja com o sistema de segurança de maneira eficaz.

Por exemplo, quando uma detecção de intrusão é feita pelo sensor HC-SR04 e confirmada pela câmara, o sistema envia uma notificação para o telefone dos moradores, informando sobre a intrusão e fornecendo evidências visuais. Os moradores podem então tomar medidas imediatas, como contactar a polícia, verificar o status do sistema ou tomar outras medidas de segurança necessárias.

A utilização do telefone na aplicação de segurança residencial fortalece a capacidade de monitoramento e resposta dos moradores, permitindo uma comunicação rápida e eficaz com o sistema de segurança. Ele desempenha um papel vital na obtenção de informações em tempo real, tomada de decisões e garantia da segurança da residência, oferecendo maior tranquilidade aos moradores.

3.6 Jumper Wires (Fios de Ligação)

Os fios de ligação, também conhecidos como cabos jumper wires, são componentes eletrônicos geralmente utilizados para criar conexões entre vários componentes numa breadboard. Geralmente, são feitos de fios isolados flexíveis com conectores em ambas as extremidades, permitindo fácil inserção e remoção.

Aqui estão alguns pontos importantes sobre os fios jumper:

- **Propósito:** Os jumper wires são utilizados principalmente para estabelecer conexões elétricas entre diferentes pontos num circuito. Eles permitem a transferência de sinais, energia ou dados entre componentes como microcontroladores, sensores, LEDs, resistores e outros módulos eletrônicos.
- **Tipos:** Os jumper wires estão disponíveis em vários tipos e configurações. Os tipos mais comuns incluem macho-macho (MM), macho-fêmea (MF) e fêmea-fêmea (FF). Os fios MM possuem pinos em ambas as extremidades, adequados para conectar dois conectores machos. Os fios MF possuem um pino numa extremidade e uma entrada na outra, permitindo a conexão de um conector macho a um conector fêmea. Os fios FF possuem entradas em ambas as extremidades e são utilizados para conectar dois conectores fêmea. [33]



Figura 8 - Fios de Ligação

Fonte: <https://shorturl.at/girEL>

3.7 Breadboard

Uma breadboard, também conhecida como placa de ensaio ou placa de prototipagem, é uma ferramenta fundamental na eletrônica e na prototipagem de circuitos. Ela é projetada para permitir a montagem rápida e temporária de circuitos eletrônicos sem a necessidade de soldagem.

Aqui estão alguns pontos importantes sobre as breadboards:

- **Estrutura:** Uma breadboard é composta por uma base de plástico com uma matriz de orifícios pequenos e condutores metálicos internos. Esses orifícios são organizados em linhas e colunas, seguindo um padrão pré-estabelecido. A maioria das breadboards possui uma faixa central dividida, chamado “canal”, que permite a conexão de componentes integrados, como circuitos integrados (CIs).
- **Conexões:** Os orifícios de uma breadboard estão interligados em grupos de linhas e colunas. Cada linha possui uma conexão comum, enquanto as colunas são separadas. Isso permite que os componentes eletrônicos sejam inseridos nos orifícios e conectados facilmente. Os componentes podem ser fixados nos orifícios por meio de pinos ou pernas, e as conexões são feitas inserindo-se os jumper wires nos orifícios adjacentes, estabelecendo assim as conexões elétricas.
- **Prototipagem:** A breadboard é amplamente utilizada para prototipar e testar circuitos eletrônicos. Ela permite que os componentes sejam conectados e desconectados rapidamente, facilitando a montagem e a modificação de circuitos sem a necessidade de soldagem. Isso torna as breadboards uma ferramenta valiosa para experimentar diferentes configurações de circuito, testar a funcionalidade dos componentes e realizar projetos eletrônicos iniciais.
- **Alimentação:** As breadboards geralmente possuem trilhos de alimentação na parte superior e inferior, permitindo a conexão fácil dos fios de alimentação. Esses trilhos estão conectados a várias linhas da matriz, fornecendo uma fonte de alimentação comum para os componentes conectados. Isso simplifica a distribuição da energia elétrica no circuito.

As breadboards são amplamente utilizadas por estudantes e profissionais de eletrônica devido à sua facilidade de uso e capacidade de prototipagem rápida. Elas permitem testar ideias, criar circuitos temporários e validar o funcionamento dos componentes antes de fazer a montagem definitiva em uma placa de circuito impresso (PCB). [34] [35]

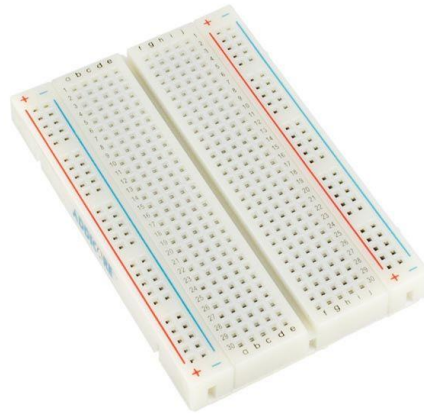


Figura 9 - Breadboard

Fonte: <https://shorturl.at/eyAX5>

3.8 USB-B

O USB-B é um dos conectores utilizados no padrão Universal Serial Bus (USB) para a conexão de dispositivos eletrônicos. Ele é um dos diversos tipos de conectores USB disponíveis, cada um com características específicas.

Aqui estão algumas informações importantes sobre o USB-B:

- **Propósito:** O conector USB-B é geralmente utilizado para a conexão de dispositivos periféricos, como impressoras, scanners, hubs USB, e outros dispositivos de comunicação de dados. Ele é projetado para ser conectado a uma porta USB-A ou USB-C em um computador, adaptador de energia ou outro dispositivo host.
- **Formato:** O USB-B possui um formato retangular com uma das extremidades ligeiramente mais larga do que a outra, criando uma forma trapezoidal. Isso garante que o conector seja inserido corretamente na porta USB, evitando danos aos pinos internos.
- **Variantes:** Existem diferentes variantes do USB-B. As duas mais comuns são:
 - **USB-B padrão:** É o conector mais comumente encontrado em dispositivos periféricos. Possui uma forma quadrada com bisel nos

cantos superiores para garantir uma inserção correta. Esse conector tem uma configuração de quatro ou cinco pinos.

- Mini USB-B: É uma versão menor do USB-B padrão. Foi muito utilizado em dispositivos portáteis, como câmeras digitais, telefones celulares e tocadores de música. O conector Mini USB-B é retangular com um chanfro em uma das extremidades. Ele possui uma configuração de cinco pinos.
- Velocidade de transferência: O USB-B suporta diferentes velocidades de transferência de dados, dependendo da versão do USB que está sendo utilizada. As versões mais recentes, como USB 3.0, USB 3.1 e USB 3.2, oferecem velocidades de transferência muito mais rápidas em comparação com as versões mais antigas, como USB 2.0 e USB 1.1.

É importante ressaltar que o USB-B está sendo gradualmente substituído por conectores mais modernos, como o USB-C, que oferece vantagens adicionais, como reversibilidade e suporte a recursos avançados, como transferência de energia bidirecional e suporte a protocolos de dados mais rápidos. [36]



Figura 10 - Cabo de ligação USB-B

Fonte: <https://shorturl.at/iKMY4>

3.9 LEDs

Os LEDs (Light-Emitting Diodes) são dispositivos semicondutores que emitem luz quando uma corrente elétrica passa por eles. Eles são amplamente utilizados em aplicações de iluminação, sinalização, displays, eletrônica e muito mais. Aqui estão algumas informações importantes sobre os LEDs:

- **Funcionamento:** Os LEDs são baseados no princípio da emissão de luz emissor de luz. Eles são construídos com camadas de materiais semicondutores que, quando energizadas pela corrente elétrica, liberam energia na forma de luz visível. A cor da luz emitida depende do material semicondutor usado no LED.
- **Eficiência energética:** Os LEDs são conhecidos por sua alta eficiência energética. Eles convertem a maioria da energia elétrica em luz, em vez de desperdiçá-la como calor, como acontece com outras fontes de luz, como lâmpadas incandescentes. Isso os torna muito mais eficientes em termos de consumo de energia.
- **Vida útil:** Os LEDs têm uma vida útil geralmente mais longa em comparação com outras tecnologias de iluminação. Eles são projetados para durar dezenas de milhares de horas, o que significa que podem durar muitos anos, dependendo do uso e das condições de operação.
- **Cores e aplicações:** Os LEDs estão disponíveis em uma ampla variedade de cores, incluindo vermelho, verde, azul, amarelo, branco e até mesmo cores multicoloridas ou RGB (Red-Green-Blue). Isso os torna versáteis para diversas aplicações, como iluminação residencial e comercial, sinalização, indicadores de painel, telas de informação, dispositivos eletrônicos, entre outros.
- **Tipos de LEDs:** Além dos LEDs convencionais, existem outros tipos especializados, como LEDs SMD (Surface Mount Device), LEDs de alto brilho, LEDs de potência e LEDs infravermelhos (IR), que não são visíveis ao olho humano, mas têm diversas aplicações em comunicação remota, detecção de proximidade, controles remotos, entre outros.
- **Controle de brilho:** É possível controlar o brilho dos LEDs ajustando a corrente que passa por eles ou utilizando técnicas de modulação de largura de pulso (PWM), que alteram a proporção entre o tempo em que o LED está ligado e desligado para criar uma ilusão de brilho variável.

Os LEDs são componentes populares e amplamente utilizados na eletrônica moderna devido à sua eficiência, durabilidade, tamanho compacto e flexibilidade de aplicações. A sua versatilidade e baixo consumo de energia os tornam uma escolha comum em uma variedade de projetos e produtos. [37] [38]



Figura 11 - LED

Fonte: https://ae01.alicdn.com/kf/Sc8402e73478843f8b9f10f4c9fc4a1f1N/250-p-s-lote-10mm-led-vermelho-azul-branco-amarelo-verde-emissor-rosa-roxo-laranja.jpg_.webp

3.10 Computador

Um computador é uma máquina eletrônica que processa informações de acordo com um conjunto de instruções pré-programadas. Ele consiste em hardware físico, como processador, memória, armazenamento, periféricos, e software, como sistema operativo e aplicativos.

Componentes principais:

- Processador (CPU): Responsável por executar as instruções e realizar cálculos.
- Memória: Armazena dados e instruções temporariamente para acesso rápido pelo processador.
- Armazenamento: Dispositivos como discos rígidos (HDDs) e unidades de estado sólido (SSDs) para armazenar dados permanentemente.
- Placa-mãe: Conecta e permite a comunicação entre os componentes principais.
- Placa de vídeo (GPU): Responsável pelo processamento de gráficos e exibição de imagens em monitores.

- Fonte de alimentação: Fornecimento de energia para todos os componentes do computador.
- Periféricos: Dispositivos externos, como teclado, rato, monitor, impressora, etc.

Tipos de computadores:

- Desktop: Computadores de mesa projetados para uso em um local fixo.
- Laptop: Computadores portáteis que oferecem mobilidade e podem ser usados em diferentes lugares.
- Servidores: Computadores projetados para fornecer serviços, armazenamento e recursos para outros dispositivos conectados em uma rede.

Software:

- Sistema operativo: Software que gere os recursos do computador e permite a execução de outros programas.
- Aplicativos: Software projetado para realizar tarefas específicas, como processamento de texto, edição de fotos, navegação na web, entre outros.

Evolução:

- Os computadores evoluíram ao longo do tempo, tornando-se cada vez mais poderosos, compactos e acessíveis. As gerações de computadores incluem válvulas eletrônicas, transístores, circuitos integrados e microprocessadores.

Conectividade:

- Os computadores podem ser conectados em redes locais ou na internet, permitindo a comunicação e compartilhamento de recursos entre vários dispositivos.

Usos comuns:

- Os computadores são amplamente utilizados em várias áreas, como educação, negócios, entretenimento, pesquisa científica, design gráfico, desenvolvimento de software e muito mais.

Essas são apenas algumas informações básicas sobre computadores. O campo da computação é vasto e em constante evolução, abrangendo muitos tópicos e subáreas. [39]



Figura 12 - Computador

Fonte: <https://shorturl.at/cgMR4>

3.11 Base de Dados com Docker e PostgreSQL

A base de dados é um componente essencial no projeto de segurança residencial, sendo responsável por armazenar e gerir as informações coletadas pelo sistema. No contexto deste projeto, optou-se por utilizar o Docker como ambiente de *contêiner* e o PostgreSQL como um sistema de gestão de base de dados.

3.11.1 Docker

O Docker é uma plataforma de código aberto que permite automatizar a implantação, o dimensionamento e a gestão de aplicativos dentro de contêineres. Um *contêiner* é uma unidade isolada que empacota todos os elementos necessários para executar um software, incluindo o código, as bibliotecas, as dependências e as configurações do sistema.

A principal ideia por trás do Docker é a de fornecer uma maneira consistente de empacotar e distribuir aplicativos, garantindo que eles sejam executados de forma confiável em qualquer ambiente. Ao usar contêineres, podemos eliminar a necessidade de configurações complexas do sistema e problemas de compatibilidade, permitindo que os aplicativos sejam executados de maneira consistente em diferentes máquinas.

Aqui estão algumas vantagens do uso do Docker:

- **Portabilidade:** Os contêineres Docker são executados de maneira consistente em qualquer ambiente, desde máquinas locais até servidores em nuvem, independentemente do sistema operativo subjacente.
- **Isolamento:** Cada *contêiner* é isolado dos outros, o que significa que podemos executar vários aplicativos num único host sem que eles interfiram uns nos outros.
- **Eficiência:** Os contêineres compartilham o núcleo do sistema operativo hospedeiro, o que significa que eles usam menos recursos em comparação com a execução de máquinas virtuais separadas para cada aplicativo.

- Escalabilidade: O Docker facilita a implementação de aplicativos em escala, permitindo que seja possível aumentar ou diminuir o número de contêineres em execução conforme as demandas de tráfego.

O Docker usa um arquivo chamado Dockerfile para definir a configuração do *contêiner*, especificando quais dependências e comandos são necessários para executar o aplicativo. [40]

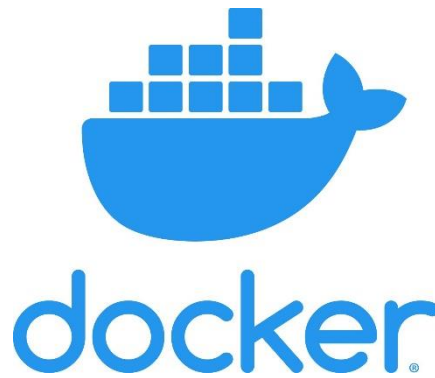


Figura 13 - Logotipo Docker

Fonte: <https://shorturl.at/moX45>

3.11.2 PostgreSQL

O PostgreSQL é um sistema de gestão de base de dados relacional de código aberto. Ele fornece um ambiente confiável e escalável para armazenar, organizar e recuperar dados de forma eficiente. Aqui estão alguns pontos-chave sobre o PostgreSQL:

- Estrutura de Base de Dados: O PostgreSQL organiza os dados em tabelas com colunas e linhas. Ele suporta uma ampla gama de tipos de dados, incluindo inteiros, *strings*, datas, *arrays*, entre outros. Além disso, oferece suporte a recursos avançados, como chaves primárias, chaves estrangeiras, índices e visões.
- Linguagem SQL: O PostgreSQL suporta a linguagem SQL (*Structured Query Language*) padrão, permitindo que seja possível fazer consultas complexas, atualizações e manipulações de dados. Ele também suporta extensões SQL avançadas e recursos específicos do PostgreSQL, como funções definidas pelo utilizador e *triggers*.
- Recursos Avançados: O PostgreSQL oferece recursos avançados, como transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade) para garantir a integridade dos dados e a recuperação em caso de falhas. Além disso, possui suporte a replicação, particionamento de tabelas, indexação avançada e consultas geoespaciais.
- Extensibilidade: O PostgreSQL permite que seja possível estender as suas funcionalidades por meio de extensões e plug-ins. Isso significa que podemos adicionar recursos personalizados à base de dados, como tipos de dados personalizados, funções definidas pelo utilizador e novos métodos de indexação.

- Comunidade Ativa: O PostgreSQL tem uma comunidade de desenvolvedores ativa e uma base de utilizadores sólida. Isso resulta em atualizações regulares, correções de segurança e melhorias contínuas no sistema.
- Suporte a Plataformas: O PostgreSQL é multiplataforma e pode ser executado em várias plataformas, incluindo Linux, Windows e macOS. Ele também é compatível com uma variedade de linguagens de programação, como Python, Java, PHP e muitas outras.

Ao usar o PostgreSQL, podemos criar, modificar e consultar bases de dados de maneira eficiente, garantindo a integridade e a segurança dos dados. Ele é amplamente utilizado em diferentes tipos de aplicativos, desde pequenos projetos até grandes sistemas corporativos. [41] [42] [43]



Figura 14 - Logotipo PostgreSQL

Fonte: <https://shorturl.at/xIU08>

3.11.3 Configuração do ambiente de base de dados

No projeto de segurança residencial, o Docker é utilizado para configurar e implantar o ambiente de base de dados de forma rápida e simplificada. Ele oferece vantagens como portabilidade, escalabilidade e facilidade de configuração. O PostgreSQL permite o armazenamento seguro e eficiente das informações coletadas pelo sistema de segurança residencial.

3.11.4 Funções e importância da base de dados

A base de dados desempenha diversas funções no projeto de segurança residencial. Ela é responsável por armazenar informações como registros de eventos de detecção de intrusões, capturas de imagens, status do sistema e outras informações relevantes. Esses dados podem ser consultados posteriormente para análise, investigação ou referência.

Além disso, a base de dados permite a integração com outros componentes do sistema, como a API em Node.js, possibilitando o acesso e a manipulação dos dados armazenados. Através da base de dados, é possível realizar consultas, atualizações, exclusões e inserções de informações de forma eficiente e organizada.

A importância da base de dados reside na sua capacidade de armazenar dados de forma segura e estruturada, garantindo a disponibilidade e integridade das informações. Ela permite o registo e o histórico de eventos de segurança, facilitando a análise e a tomada de decisões. Além

disso, a base de dados possibilita a geração de relatórios, a realização de estatísticas e a criação de mecanismos de backup e recuperação de dados.

A utilização do Docker em conjunto com o PostgreSQL oferece uma solução eficiente e escalável para a gestão de dados no projeto de segurança residencial. A base de dados desempenha um papel fundamental na coleta, armazenamento e recuperação de informações relevantes, contribuindo para a segurança e o monitoramento da residência.

3.12 API em Node.js

A API em Node.js desempenha um papel crucial no projeto de segurança residencial, atuando como intermediária entre os dispositivos físicos e os aplicativos de software. Ela permite a comunicação e a interação entre os diferentes componentes do sistema, bem como a exposição de funcionalidades e serviços para os utilizadores.

3.12.1 API

Uma API (*Application Programming Interface*, ou Interface de Programação de Aplicativos) é um conjunto de regras e protocolos que permite que diferentes softwares se comuniquem e interajam entre si. Ela define os métodos de comunicação e os formatos de dados que os aplicativos podem usar para enviar e receber informações.

Basicamente, uma API atua como uma ponte entre diferentes componentes de software, permitindo que eles troquem informações e solicitem a execução de determinadas tarefas. Ao utilizar uma API, um aplicativo pode aceder às funcionalidades ou dados de outro aplicativo, sem precisar conhecer detalhes internos de como essas funcionalidades são implementadas.

Com o crescimento da interconectividade e da integração de sistemas, as APIs tornaram-se fundamentais para o desenvolvimento de software moderno. Elas são amplamente usadas em aplicações web, aplicativos móveis, serviços em nuvem e muitas outras soluções de software para permitir a comunicação entre diferentes componentes e sistemas. [44]

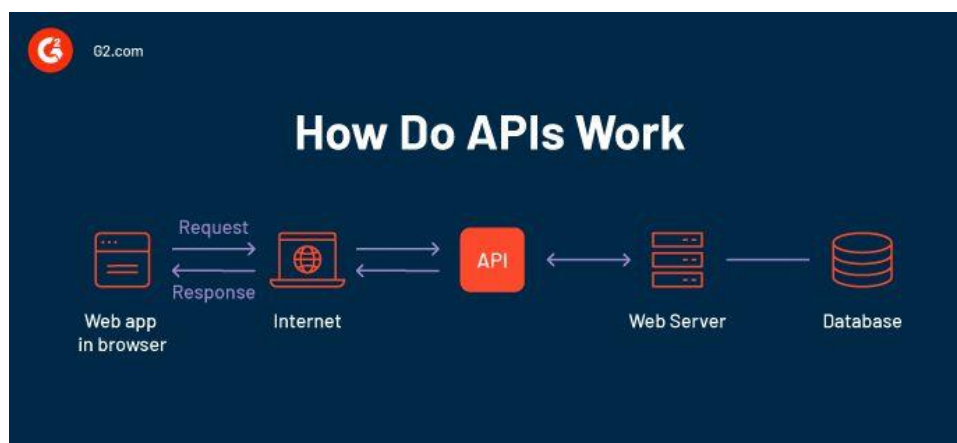


Figura 15 - Funcionamento de uma API

Fonte: <https://shorturl.at/bgnB5>

3.12.2 Node.js

O Node.js é um ambiente de tempo de execução (*runtime*) de código aberto, construído em cima do motor JavaScript V8 do Google Chrome. Ele permite que os desenvolvedores executem código JavaScript no lado do servidor, em vez de apenas no navegador.

Uma das principais características do Node.js é sua arquitetura orientada a eventos e baseada em chamadas de retorno (*callbacks*). Isso significa que ele é projetado para lidar com muitas solicitações assíncronas simultaneamente, tornando-o eficiente para aplicações em tempo real e de alta escalabilidade. Em vez de bloquear a execução enquanto espera por uma operação de entrada/saída, o Node.js usa chamadas de retorno para continuar a execução e manipular a resposta quando a operação for concluída.

O Node.js também possui um sistema de gestão de pacotes chamado npm (Node Package Manager). O npm permite que os desenvolvedores compartilhem, instalem e façam a gestão dos pacotes e bibliotecas de terceiros de maneira fácil e eficiente. Isso proporciona uma vasta quantidade de módulos e bibliotecas disponíveis para uso no desenvolvimento de aplicativos com o Node.js.

Por ser baseado em JavaScript, o Node.js permite que os desenvolvedores usem a mesma linguagem tanto no lado do servidor quanto no lado do cliente, facilitando a criação de aplicativos completos com código compartilhado entre o *front-end* e o *back-end*. Isso ajuda a reduzir a complexidade e aumentar a produtividade no desenvolvimento de aplicações web. [45] [46] [47]



Figura 16 - Logotipo node.js

Fonte: <https://shorturl.at/dfjBX>

3.12.3 Papel da API no projeto

A API em Node.js é responsável por fornecer uma interface de comunicação padronizada e segura para os componentes do sistema de segurança residencial. Ela permite que os dispositivos, como o Arduino UNO-WIFI-REV2, a câmera e o sensor HC-SR04, enviem e recebam dados e comandos, facilitando a integração e o funcionamento do sistema como um todo.

Além disso, a API é responsável por receber as requisições dos aplicativos de software, como os aplicativos móveis ou os painéis de controle web, e fornecer as respostas correspondentes. Isso permite aos utilizadores interagir com o sistema de segurança, receber notificações, visualizar dados e realizar ações remotas, como ativar ou desativar o sistema, monitorizar câmaras ao vivo ou verificar o status do sistema.

3.12.4 Funcionalidades e endpoints

A API em Node.js implementa uma série de funcionalidades e endpoints que permitem a interação com o sistema de segurança residencial. Alguns exemplos de funcionalidades e endpoints incluem:

- Autenticação e autorização: a API gere a autenticação dos utilizadores e garante que apenas utilizadores autorizados possam aceder às funcionalidades do sistema.
- Notificações em tempo real: a API envia notificações em tempo real para os aplicativos de software quando ocorrem eventos de segurança, como deteção de intrusões ou emergências.
- Controle do sistema: a API permite que os utilizadores ativem ou desativem o sistema de segurança, ajustem configurações, monitorizem o status dos dispositivos e realizem ações remotas.

- Acesso a dados: a API fornece endpoints para aceder e consultar informações armazenadas na base de dados, como registos de eventos, captura de imagens e outros dados relevantes.
- Integração com outros serviços: a API pode ser integrada a outros serviços, como provedores de armazenamento em nuvem, para permitir o armazenamento seguro de dados e imagens capturadas.

A API em Node.js desempenha um papel central na comunicação e interação do sistema de segurança residencial. Ela oferece uma interface padronizada e segura para os dispositivos e aplicativos de software, permitindo uma experiência completa e integrada para os usuários.

3.13 Ambiente de Desenvolvimento (IDE)

Neste trabalho utilizamos o IntelliJ para desenvolvimento da API e da aplicação, uma vez que é uma IDE amplamente utilizada por desenvolvedores de Java e outras linguagens, conhecida por sua rica gama de recursos e suporte a várias tecnologias. Para o desenvolvimento do “script” do arduino usamos o VSCode uma vez que ao utilizar o VSCode para desenvolver ‘scripts’ para o Arduino, os desenvolvedores podem aproveitar recursos como realce de sintaxe, formatação automática, depuração, controle de versão e a vasta biblioteca de extensões disponíveis.

3.13.1 Utilização do IntelliJ para desenvolvimento da API e aplicação:

O IntelliJ é uma poderosa e popular IDE (*Integrated Development Environment*) amplamente utilizada para o desenvolvimento de aplicativos e APIs. A escolha de utilizar o IntelliJ para o desenvolvimento da API e aplicação do sistema de segurança residencial traz diversos benefícios.

Primeiramente, o IntelliJ oferece um ambiente de desenvolvimento altamente produtivo, com recursos avançados de autocompletar código, depuração e refratoração. Isso permite que os desenvolvedores escrevam código de forma mais eficiente e com menos erros, agilizando o processo de desenvolvimento.

Além disso, o IntelliJ oferece suporte a várias linguagens de programação, como Java, Kotlin, Python, entre outras. Isso proporciona flexibilidade na escolha da linguagem de programação mais adequada para o desenvolvimento da API e aplicação do sistema de segurança residencial, dependendo dos requisitos do projeto e das preferências da equipa de desenvolvimento.

Outro ponto importante é a disponibilidade de uma ampla gama de plugins e integrações com ferramentas de desenvolvimento populares. Isso facilita a integração de *frameworks* e bibliotecas relevantes para a implementação do sistema de segurança residencial, proporcionando maior agilidade e eficiência no desenvolvimento. [48]

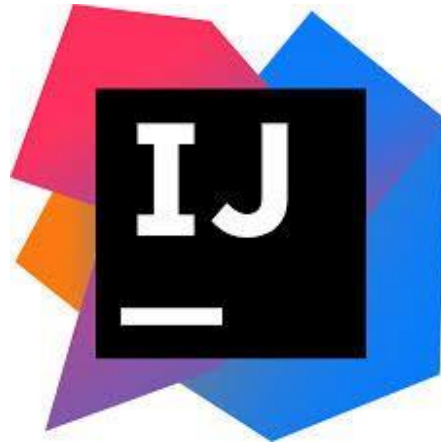


Figura 17 - Logotipo IntelliJ

Fonte:

https://upload.wikimedia.org/wikipedia/commons/thumb/9/9c/IntelliJ_IDEA_Icon.svg/1024px-IntelliJ_IDEA_Icon.svg.png

3.13.2 Utilização do VSCode para desenvolvimento do script para o Arduino:

O VSCode é um editor de código-fonte leve e altamente configurável, desenvolvido pela Microsoft. Embora seja conhecido por sua ampla aplicação no desenvolvimento web, também é uma escolha popular para programação de sistemas embarcados, como o Arduino.

A utilização do VSCode para o desenvolvimento do script para o Arduino traz diversas vantagens. Primeiramente, o VSCode oferece suporte a várias extensões relacionadas ao desenvolvimento para Arduino, como a plataforma oficial do Arduino e outras extensões de terceiros. Essas extensões fornecem recursos avançados, como verificação de código, autocompletar, depuração e upload direto para a placa Arduino.

Além disso, o VSCode possui uma interface intuitiva e personalizável, permitindo que os desenvolvedores ajustem o ambiente de desenvolvimento conforme as suas preferências e necessidades. Isso inclui a seleção de temas, atalhos de teclado personalizados e a configuração de extensões específicas para otimizar a produtividade.

Outro aspecto importante é a facilidade de integração com controle de versão, como o Git. O VSCode possui recursos nativos de controle de versão que permitem aos desenvolvedores gerir e visionar o código-fonte do script do Arduino eficientemente, facilitando a colaboração e o rastreamento de alterações.

Para conseguirmos fazer a comunicação entre o Arduino com o VSCode, tivemos de usar uma extensão chamada Arduino extension (uma espécie de plug-in). [49] [50]

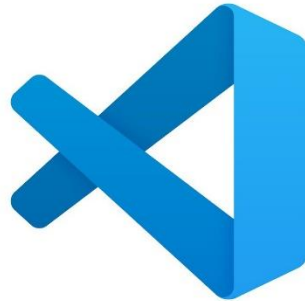


Figura 18 - Logotipo VS Code

Fonte: <https://shorturl.at/aQR37>

3.14 Mecanismo de Comunicação do Sistema:

O sistema de segurança residencial utilizará diferentes mecanismos de comunicação para permitir a interação entre os diferentes componentes. Os principais mecanismos de comunicação são os seguintes:

3.14.1 Wifi entre Arduino e API:

O Arduino se comunicará com a API por meio de uma conexão Wi-Fi. O Arduino estará equipado com um módulo Wi-Fi, que permitirá que ele se conecte à rede Wi-Fi local. Através dessa conexão, o Arduino poderá enviar dados para a API, como leituras dos sensores de movimento ou solicitações de ativação de dispositivos de segurança. A API receberá essas informações e processará as ações correspondentes com base nas regras de segurança definidas.

3.14.2 API entre Aplicação - Sockets:

A comunicação entre a aplicação móvel e a API será estabelecida por meio de uma conexão de *socket*. Os *sockets* permitem a comunicação bidirecional em tempo real entre a aplicação e a API. A aplicação estabelecerá uma conexão com a API usando um protocolo de comunicação baseado em *sockets*, como o TCP/IP ou *WebSocket*. Isso permitirá a transmissão de dados em tempo real, como a solicitação de status de segurança, configurações personalizadas e o recebimento de notificações instantâneas da API. [51]

3.14.3 API entre Base de Dados - Sockets:

A API também estabelecerá uma conexão de *socket* com o banco de dados para acessar e manipular os dados armazenados. Isso permite que a API envie consultas e comandos diretamente para o banco de dados e receba as respostas correspondentes. Essa comunicação bidirecional permite que a API execute operações de leitura e gravação no banco de dados, garantindo a consistência dos dados e a atualização em tempo real.

Esses mecanismos de comunicação garantem a interação eficiente e em tempo real entre os diferentes componentes do sistema de segurança residencial. A conexão Wi-Fi entre o Arduino e a API permite que o Arduino envie informações de sensoriamento e receba comandos de ativação de dispositivos de segurança. A comunicação por *sockets* entre a aplicação e a API permite que os utilizadores interajam em tempo real com o sistema, recebam notificações instantâneas e atualizem as configurações. A conexão de *socket* entre a API e o banco de dados permite que a API acesse e manipule os dados armazenados eficientemente.

Esses mecanismos de comunicação combinados fornecem um sistema de segurança residencial altamente responsivo e confiável, garantindo uma comunicação efetiva e atualização constante dos dados entre os diferentes componentes do sistema.

4 Metodologia do modelo proposto

Nesta secção, será apresentada a metodologia adotada para o desenvolvimento do modelo proposto de sistema de segurança residencial. Serão abordados os processos de construção da maquete, implementação física do protótipo, desenvolvimento da aplicação e integração entre os componentes.

4.1 Apresentação geral do modelo proposto

O modelo proposto consiste num sistema de segurança residencial que integra uma maquete física, um protótipo implementado com componentes eletrónicos, uma API desenvolvida em Node.js e uma aplicação móvel criada em Flutter. O objetivo do sistema é simular e oferecer funcionalidades de segurança para uma residência, permitindo aos utilizadores monitorar e controlar remotamente o ambiente residencial.

A maquete física é uma representação em escala de um ambiente residencial, onde são colocados os diferentes componentes, como sensores de movimento, câmara, LEDs e buzzer, em posições estratégicas para simular a deteção de eventos e a ativação dos dispositivos de segurança. Ela permite visualizar e compreender concretamente o funcionamento do sistema de segurança.

O protótipo implementado com componentes eletrónicos é responsável por capturar e processar os dados dos sensores, realizar o controlo dos dispositivos de segurança e estabelecer a comunicação com a API e a aplicação móvel. O Arduino REV2 WIFI é utilizado como o microcontrolador principal do protótipo, conectado aos sensores e dispositivos por meio de fios e breadboard. Ele é programado para executar as lógicas de segurança, acionando os dispositivos apropriados com base nos dados recebidos dos sensores.

A API desenvolvida em Node.js fornece os serviços e funcionalidades necessários para a comunicação entre o protótipo e a aplicação móvel. Ela atua como um intermediário, recebendo solicitações da aplicação móvel por meio de endpoints definidos, processando essas solicitações e enviando comandos ao Arduino. Além disso, a API também recebe dados dos sensores e os disponibiliza para a aplicação móvel, permitindo a visualização em tempo real do status de segurança.

A aplicação móvel desenvolvida em Flutter é a interface de utilizador do sistema de segurança residencial. Ela permite aos utilizadores acedam ao sistema por meio de um dispositivo móvel, como smartphones ou tablets. A aplicação oferece funcionalidades como visualização do status de segurança, configurações personalizadas, notificações em tempo real e interação com o sistema de segurança residencial, como ativar ou desativar dispositivos de segurança.

Essa abordagem integrada do modelo proposto, combinando a maquete física, o protótipo eletrônico, a API e a aplicação móvel, permite uma compreensão completa do funcionamento do sistema de segurança residencial. Ele oferece aos utilizadores uma experiência imersiva e interativa, fornecendo controle e monitoramento do ambiente residencial de forma eficiente e conveniente.

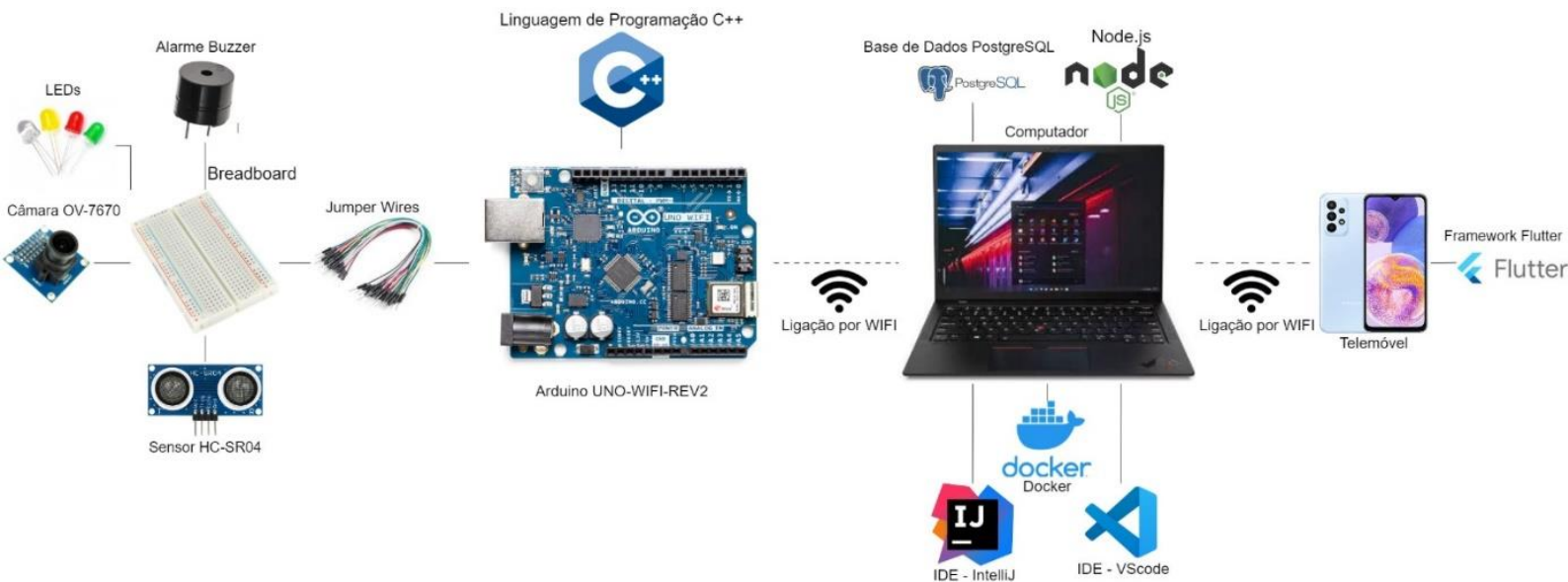


Figura 19 - Arquitetura do Projeto

4.2 Desenvolvimento do Modelo Proposto

O desenvolvimento do modelo proposto começa com a construção da maquete física. A maquete é uma representação em escala de um ambiente residencial, incluindo diferentes áreas, como sala de estar, quarto, cozinha e áreas externas. Durante a construção da maquete, são instalados os componentes físicos, como sensores de movimento, câmera, LEDs e buzzer, em locais estratégicos da maquete para simular a detecção de eventos e a ativação dos dispositivos de segurança.

4.2.1 Construção da Maquete

O desenvolvimento do modelo proposto começa com a construção da maquete física. A maquete é uma representação de um ambiente residencial. Durante a construção da maquete, são instalados os componentes físicos, como sensores de movimento, câmera, LEDs e buzzer, em locais estratégicos da maquete para simular a detecção de eventos e a ativação dos dispositivos de segurança.

4.2.2 Implementação física do Protótipo

A implementação física do protótipo é uma etapa crucial no desenvolvimento do modelo proposto de sistema de segurança residencial. Nessa etapa, os componentes eletrônicos, como

sensores, câmara, LEDs, buzzer e Arduino REV2 WIFI, são conectados e configurados para o sistema funcionar corretamente.

A primeira etapa é conectar os componentes eletrônicos ao Arduino REV2 WIFI. Utilizando fios e uma breadboard, os sensores de movimento, câmara, LEDs e buzzer são conectados aos pinos correspondentes no Arduino. É importante seguir o esquema de conexão correto, utilizando os pinos adequados para cada componente e garantindo uma conexão segura e estável.

Após a conexão física dos componentes, a próxima etapa envolve a programação do Arduino REV2 WIFI. É necessário escrever o código em linguagem C++ para definir o comportamento do sistema de segurança residencial. O código deve incluir a lógica de leitura dos sensores, acionamento dos LEDs e buzzer, e comunicação com a API.

A lógica de leitura dos sensores é responsável por monitorar as alterações de movimento detetadas pelos sensores de movimento e acionar as ações correspondentes. Por exemplo, se um sensor de movimento deteta uma presença, o Arduino pode enviar um sinal para acender um LED indicando o evento e também ativar o buzzer para emitir um alarme sonoro.

A comunicação com a API é estabelecida por meio de conexão Wi-Fi. O Arduino se conecta à rede local e envia os dados dos sensores para a API, que por sua vez os disponibiliza para a aplicação móvel. Além disso, a API pode enviar comandos de ativação ou desativação de dispositivos de segurança para o Arduino, permitindo que a aplicação móvel controle o sistema de segurança residencial remotamente.

Durante a implementação física do protótipo, é importante realizar testes e ajustes para garantir o correto funcionamento do sistema. É necessário verificar se os sensores detetam corretamente os eventos, se os LEDs e buzzer são acionados conforme as condições definidas, e se a comunicação com a API é estabelecida adequadamente.

Essa etapa de implementação física do protótipo é essencial para transformar o modelo conceitual num sistema funcional. A correta conexão e programação dos componentes eletrônicos, com a comunicação com a API, permitem que o sistema de segurança residencial opere eficientemente e interaja com os utilizadores por meio da aplicação móvel.

4.2.3 Implementação da Aplicação

A implementação da aplicação móvel é uma parte fundamental do modelo proposto de sistema de segurança residencial. Nessa etapa, a interface de utilizador é desenvolvida usando o framework Flutter e são criadas as funcionalidades necessárias para interagir com o sistema de segurança.

A aplicação móvel é responsável por fornecer uma interface intuitiva e amigável para os utilizadores acederem e controlarem o sistema de segurança residencial. Ela permite visualizar o status de segurança, configurar as preferências do sistema e receber notificações em tempo real sobre eventos relevantes.

Para iniciar a implementação da aplicação, é necessário definir a estrutura de navegação e os principais elementos da interface, como telas, botões e campos de entrada. O Flutter oferece uma ampla gama de widgets personalizáveis que podem ser utilizados para criar uma experiência visual agradável e coerente.

A comunicação entre a aplicação móvel, a API e o sistema de segurança são estabelecidas por meio de solicitações e respostas HTTP. A aplicação móvel envia solicitações para a API, solicitando informações do sistema ou enviando comandos para ativar ou desativar dispositivos de segurança. A API processa essas solicitações e retorna as respostas apropriadas, permitindo a interação em tempo real entre a aplicação e o sistema.

Além disso, a aplicação móvel pode implementar recursos adicionais, como autenticação de utilizador, histórico de eventos, configurações personalizadas e notificações *push*. A autenticação de utilizador garante que apenas utilizadores autorizados possam aceder e controlar o sistema de segurança residencial. O histórico de eventos permite aos utilizadores visualizar eventos passados, como deteção de movimento ou ativação de dispositivos. As configurações personalizadas permitem que os utilizadores personalizem as preferências do sistema, como tempo de atraso para ativação do alarme. As notificações *push* mantêm os utilizadores atualizados sobre eventos importantes, mesmo quando a aplicação não está em uso.

Durante a implementação da aplicação, é importante realizar testes para garantir o seu correto funcionamento e usabilidade. É necessário verificar se as telas são renderizadas corretamente, se as solicitações HTTP são enviadas e processadas corretamente pela API e se as notificações são recebidas adequadamente.

Essa etapa de implementação da aplicação móvel é fundamental para permitir que os utilizadores interajam com o sistema de segurança residencial de maneira conveniente e intuitiva. Através da aplicação, os utilizadores podem monitorar o status de segurança, controlar os dispositivos de segurança e receber notificações em tempo real, proporcionando uma experiência completa e funcional do sistema.

4.2.4 Integração com a Aplicação

A integração entre o sistema de segurança residencial e a aplicação móvel é uma etapa crucial para garantir que os utilizadores possam interagir e controlar o sistema eficientemente. Nessa etapa, é estabelecida a comunicação entre a aplicação e a API, permitindo o envio e recebimento de dados relevantes.

A integração começa com a configuração adequada da API para receber solicitações da aplicação móvel. A API deve estar preparada para receber dados de sensores, comandos de ativação ou desativação de dispositivos, e também para fornecer informações sobre o status do sistema e eventos passados.

A aplicação móvel utiliza solicitações HTTP para se comunicar com a API. Por exemplo, quando o utilizador abre a aplicação, ela envia uma solicitação para a API para obter informações atualizadas sobre o status do sistema, como se os sensores estão ativos ou inativos. Essas informações são então exibidas na interface da aplicação, permitindo que o utilizador tenha uma visão clara da segurança da residência.

Além disso, a aplicação permite que o utilizador interaja com o sistema de segurança residencial. Por exemplo, o utilizador pode ativar ou desativar os sensores de movimento, configurar o tempo de atraso para ativação do alarme ou visualizar o histórico de eventos registados pelo sistema. Essas interações são realizadas por meio de solicitações enviadas à API, que executa as ações correspondentes no sistema físico.

A integração também pode envolver a implementação de notificações *push* na aplicação móvel. Essas notificações são enviadas pela API para informar o utilizador sobre eventos

relevantes, como a detecção de movimento ou a ativação do alarme. As notificações *push* são exibidas no dispositivo móvel, mesmo quando a aplicação não está em uso, mantendo o utilizador informado sobre a segurança da residência em tempo real.

Durante a integração, é importante realizar testes para garantir a correta comunicação entre a aplicação e a API. Os desenvolvedores devem verificar se as solicitações são enviadas e recebidas corretamente, se os dados são processados adequadamente e se as notificações *push* são exibidas corretamente no dispositivo móvel.

A integração eficiente entre a aplicação móvel e a API do sistema de segurança residencial é essencial para garantir uma experiência completa e funcional aos utilizadores. Ela possibilita que os utilizadores controlem e recebam notificações em tempo real sobre a segurança das suas residências, proporcionando tranquilidade e controle sobre o sistema.

4.3 Histórico

O histórico é uma parte importante do modelo proposto de sistema de segurança residencial, pois permite aos utilizadores aceder e rever os eventos e atividades registados pelo sistema ao longo do tempo. Essa funcionalidade fornece um registo detalhado das ocorrências passadas, como deteções de movimento, ativações de dispositivos de segurança e quaisquer outros eventos relevantes.

No histórico, os eventos são armazenados numa base de dados, que registar as informações essenciais de cada evento, como data, hora, tipo de evento e descrição. Isso permite que os utilizadores revejam e analisem o histórico, compreendendo o comportamento do sistema e identificando padrões ou anomalias.

Através da aplicação móvel, os utilizadores podem aceder ao histórico e navegar pelos eventos registados. Eles podem visualizar os detalhes de cada evento, como a data e a hora em que ocorreu, o tipo de evento (por exemplo, deteção de movimento) e uma breve descrição do evento em si. Além disso, é possível utilizar filtros de pesquisa para facilitar a localização de eventos específicos ou aplicar filtros de data e hora para limitar o período de busca.

A função do histórico é fornecer aos utilizadores um panorama completo das atividades do sistema de segurança residencial, permitindo que eles identifiquem eventuais problemas, analisem padrões de atividade e tomem medidas adequadas com base nas informações disponíveis. Por exemplo, se houver uma série de deteções de movimento num determinado horário todas as noites, o utilizador pode decidir ajustar as configurações de sensibilidade do sensor ou implementar medidas adicionais de segurança.

É importante ressaltar que o histórico deve ser de fácil acesso e compreensão para os utilizadores. A interface da aplicação móvel deve ser projetada de maneira intuitiva, permitindo que os utilizadores naveguem facilmente pelos eventos registados e obtenham informações relevantes de concisamente.

Neste trabalho, foi apresentado um modelo proposto para um sistema de segurança residencial, abordando desde a construção da maquete até a implementação física do protótipo e a criação da aplicação móvel. O objetivo principal foi desenvolver um ambiente de desenvolvimento completo para o sistema, utilizando o IntelliJ para a API e aplicação, e o VSCode para o script do Arduino.

4.4 Configuração do Ambiente de Desenvolvimento

A configuração do ambiente de desenvolvimento consiste na preparação do ambiente de software e hardware necessário para o desenvolvimento do projeto de segurança residencial. As seguintes etapas foram realizadas:

- **Instalação do Arduino IDE:** O Arduino IDE é a plataforma de desenvolvimento utilizada para programar o microcontrolador Arduino UNO-WIFI-REV2. Ele fornece uma interface de programação amigável, recursos de edição de código, compilação e carregamento do código no microcontrolador. O Arduino IDE foi baixado do site oficial e instalado no computador.
- **Configuração do Arduino UNO-WIFI-REV2:** O Arduino UNO-WIFI-REV2 é o componente central do sistema de segurança residencial. Antes de iniciar o desenvolvimento, foi necessário configurar o Arduino UNO-WIFI-REV2. Isso incluiu a conexão física do Arduino UNO-WIFI-REV2 ao computador usando um cabo USB, bem como a seleção da placa correta e da porta serial no Arduino IDE.
- **Instalação de bibliotecas:** Para facilitar o desenvolvimento do projeto, foram instaladas bibliotecas adicionais no Arduino IDE. Essas bibliotecas são conjuntos de código pré-desenvolvido que fornecem funções e recursos úteis para trabalhar com componentes específicos. No caso do projeto de segurança residencial, foram instaladas bibliotecas relevantes para os componentes utilizados, como o sensor HC-SR04 e a câmara OV-7670. As bibliotecas foram baixadas dos repositórios oficiais e adicionadas ao Arduino IDE.
- **Configuração dos pinos e periféricos:** O Arduino UNO-WIFI-REV2 possui vários pinos digitais e analógicos que podem ser configurados para diferentes finalidades. Foi feita uma análise dos requisitos do projeto e a configuração adequada dos pinos e periféricos do Arduino UNO-WIFI-REV2. Isso incluiu a atribuição de pinos para a conexão dos componentes, como o sensor HC-SR04 e a câmara OV-7670, e a configuração dos modos de operação adequados para cada pino.
- **Teste da configuração:** Após a configuração do ambiente de desenvolvimento, foram realizados testes preliminares para verificar se o Arduino UNO-WIFI-REV2 estava corretamente conectado e se o ambiente de desenvolvimento estava configurado corretamente. Isso incluiu a execução de um código de teste simples para acender um LED conectado a um dos pinos do Arduino UNO-WIFI-REV2 e verificar se ele funcionava corretamente.

A configuração do ambiente de desenvolvimento é uma etapa crucial para o sucesso do projeto de segurança residencial. A correta configuração do ambiente, incluindo a instalação do Arduino IDE, a preparação do Arduino UNO-WIFI-REV2 e a instalação de bibliotecas relevantes, proporcionou as condições adequadas para o desenvolvimento do sistema. Os testes realizados após a configuração foram essenciais para verificar a correta conexão e funcionamento dos componentes. Com o ambiente de desenvolvimento configurado, foi possível avançar para a integração dos componentes e o desenvolvimento das funcionalidades do sistema de segurança residencial.

4.5 Integração dos Componentes e Comunicação entre Eles

A integração dos componentes é necessária para que eles possam funcionar de forma coordenada e realizar as funcionalidades desejadas no projeto de segurança residencial. As seguintes etapas foram seguidas para realizar essa integração:

- **Conexão física dos componentes:** Inicialmente, os componentes foram devidamente conectados ao Arduino UNO-WIFI-REV2. Por exemplo, o sensor HC-SR04 foi conectado a pinos específicos do Arduino para receber e enviar sinais ultrassônicos. Da mesma forma, a câmara OV-7670 foi conectada a pinos dedicados para captar imagens.
- **Programação do Arduino UNO-WIFI-REV2:** em seguida, o código foi desenvolvido no Arduino IDE para controlar os componentes conectados. Foram implementadas funções e lógicas específicas para interagir com cada componente, como a leitura de dados do sensor HC-SR04, a ativação da câmara OV-7670 para captura de imagens e o acionamento do alarme buzzer.
- **Comunicação entre os componentes:** Para permitir a comunicação entre os componentes, foram utilizados protocolos de comunicação adequados. Por exemplo, o sensor HC-SR04 pode fornecer leituras de distância ao Arduino UNO-WIFI-REV2 por meio de sinais ultrassônicos, que são interpretados e processados pelo código do Arduino. Da mesma forma, a câmara OV-7670 pode ser controlada pelo Arduino para captar imagens e enviá-las para processamento posterior.
- **Testes de integração:** Após a implementação do código de integração, foram realizados testes para verificar se os componentes estavam se comunicando corretamente e se as funcionalidades desejadas eram executadas adequadamente. Isso envolveu a verificação da leitura correta dos dados do sensor HC-SR04, a captura de imagens pela câmara OV-7670 e a ativação do alarme buzzer quando necessário.
- **Verificação da comunicação com o telefone:** um aspeto importante da integração foi a verificação da comunicação entre o Arduino UNO-WIFI-REV2 e o telefone. Isso permitiu a notificação em tempo real de eventos de segurança residencial para o utilizador. Foi realizada a configuração adequada de uma conexão sem fio entre os dispositivos e a implementação de um protocolo de comunicação para o envio das notificações.

A integração dos componentes e a comunicação entre eles são etapas essenciais para o funcionamento adequado do projeto de segurança residencial. A correta conexão física dos componentes, juntamente com a programação adequada do Arduino UNO-WIFI-REV2, permitiu a coordenação das funcionalidades de deteção de intrusão, captura de imagens, acionamento do alarme buzzer e notificação no telefone. Os testes de integração foram fundamentais para verificar a correta comunicação entre os componentes e garantir o bom funcionamento do sistema. Com a integração dos componentes concluída, foi possível avançar para o desenvolvimento da API em Node.js e a integração com a base de dados PostgreSQL.

4.6 Desenvolvimento da API em Node.js

A API em Node.js desempenha um papel fundamental no projeto de segurança residencial, fornecendo serviços e funcionalidades adicionais. Neste relatório, será discutido o desenvolvimento da API, abordando as etapas envolvidas e as principais funcionalidades implementadas.

- Configuração do ambiente de desenvolvimento: Antes de iniciar o desenvolvimento da API em Node.js, o ambiente de desenvolvimento foi configurado. Isso envolveu a instalação do Node.js e do gestor de pacotes npm. Além disso, foram instalados os pacotes necessários, como o Express.js, para facilitar o desenvolvimento da API.
- Criação do projeto e instalação de pacotes: Foi criado um projeto Node.js utilizando o comando “*npm init*”, fornecendo as informações necessárias, como nome, versão e descrição do projeto. Em seguida, foram instalados os pacotes necessários para o desenvolvimento da API, como o Express.js e o pacote pg para a integração com o PostgreSQL.
- Configuração do servidor Express.js: Foi criado um arquivo *server.js* para configurar o servidor Express.js. Nesse arquivo, foram definidas as configurações básicas do servidor, como a porta em que ele irá escutar as solicitações HTTP.
- Definição de rotas e controladores: Foram definidas as rotas da API, que correspondem às diferentes funcionalidades que serão disponibilizadas. Por exemplo, foram definidas rotas para autenticação de utilizadores, acesso aos dados do sistema de segurança, controle do sistema de segurança, entre outras. Para cada rota, foram criados controladores correspondentes que lidam com as solicitações recebidas e executam as operações necessárias.
- Integração com o banco de dados: Utilizando o pacote pg, foi estabelecida a conexão com o PostgreSQL. Foram implementadas funções para interagir com o banco de dados, como consultas, inserções, atualizações e exclusões de dados. Essas funções foram chamadas pelos controladores da API para realizar as operações desejadas no banco de dados.
- Implementação das funcionalidades da API: Com base nos requisitos do projeto, foram implementadas as funcionalidades necessárias na API. Isso incluiu o acesso aos dados dos sensores, o controle do sistema de segurança, a autenticação de utilizadores e a integração com outros sistemas ou dispositivos. As funcionalidades foram implementadas nos controladores correspondentes, que se comunicaram com o banco de dados, executaram lógicas de negócio e forneceram as respostas adequadas.
- Teste e depuração: Após a implementação da API, foram realizados testes para verificar o correto funcionamento das funcionalidades. Utilizaram-se ferramentas de teste de API, como o *Postman*, para testar cada rota e verificar as respostas retornadas. Eventuais erros e problemas identificados durante os testes foram depurados e corrigidos.

- Documentação: A API foi devidamente documentada, incluindo as rotas disponíveis, os parâmetros esperados, as respostas retornadas e outras informações relevantes. Essa documentação é importante para facilitar o entendimento e a utilização da API por outros desenvolvedores, além de auxiliar na manutenção e evolução do sistema.

A API em Node.js desenvolvida para o projeto de segurança residencial desempenha um papel crucial na integração dos componentes e na disponibilização de funcionalidades adicionais. O desenvolvimento da API envolveu a configuração do ambiente de desenvolvimento, a definição de rotas e controladores, a integração com o banco de dados e a implementação das funcionalidades necessárias. A realização de testes e a documentação adequada foram etapas essenciais para garantir o correto funcionamento e facilitar a utilização da API.

4.7 Integração com a Base de Dados PostgreSQL

A integração com a base de dados é essencial para armazenar e recuperar informações relevantes para o projeto de segurança residencial. As seguintes etapas foram realizadas para integrar a aplicação com a base de dados PostgreSQL:

- Configuração do ambiente de banco de dados: primeiramente, foi necessário configurar o ambiente de banco de dados. Isso incluiu a instalação do PostgreSQL no servidor adequado e a configuração das credenciais de acesso, como nome de utilizador e senha. Também foi definido o esquema de banco de dados a ser utilizado, incluindo as tabelas necessárias para armazenar os dados do projeto.
- Conexão com a base de dados: em seguida, a aplicação foi configurada para estabelecer uma conexão com a base de dados PostgreSQL. Foi utilizado um driver adequado para a linguagem de programação em que a aplicação foi desenvolvida (por exemplo, Node.js). A conexão foi estabelecida fornecendo as informações de conexão, como host, porta, nome do banco de dados e credenciais de acesso.
- Criação de tabelas e estrutura do banco de dados: com a conexão estabelecida, foram criadas as tabelas necessárias para armazenar os dados relevantes para o projeto de segurança residencial. A estrutura do banco de dados foi definida conforme as necessidades do projeto, incluindo os campos e tipos de dados adequados para cada tabela.
- Manipulação de dados: Após a criação da estrutura do banco de dados, a aplicação foi programada para manipular os dados. Isso incluiu a inserção de novos registos na base de dados quando eventos de segurança ocorriam (por exemplo, deteção de intrusão), a atualização de registos existentes quando necessário e a recuperação de dados para exibição ou processamento posterior.
- Testes e validação: Foram realizados testes para verificar a correta integração da aplicação com a base de dados. Isso envolveu a execução de operações de

inserção, atualização e recuperação de dados para verificar se os dados foram corretamente armazenados e recuperados da base de dados PostgreSQL.

A integração com a base de dados PostgreSQL é fundamental para armazenar e recuperar informações relevantes para o projeto de segurança residencial. A correta configuração do ambiente de banco de dados, a conexão adequada com a base de dados, a criação das tabelas e a manipulação dos dados foram etapas essenciais para a integração bem-sucedida da aplicação com a base de dados. Com a integração concluída, foi possível armazenar os dados relevantes do projeto e utilizá-los para análises, relatórios e outras funcionalidades.

5 Funcionalidades e Casos de Uso

Relativamente ao nosso projeto, os casos de uso referem-se a situações ou cenários específicos nos quais o sistema de segurança residencial é aplicado para atender a necessidades de segurança. Cada caso de uso descreve uma situação em que o sistema de segurança é utilizado para detetar, monitorar ou responder a eventos de segurança numa residência.

5.1 5.1 Detecção de Intrusão usando o Sensor HC-SR04

A detecção de intrusão é uma funcionalidade fundamental em sistemas de segurança residencial. O sensor HC-SR04 é amplamente utilizado para esse propósito devido à sua capacidade de medir a distância entre o sensor e um objeto usando ondas ultrassónicas.

O sensor HC-SR04 consiste num emissor de ultrassom que emite pulsos ultrassónicos e um recetor que captura os pulsos refletidos pelo objeto. Com base no tempo de ida e volta desses pulsos, é possível calcular a distância entre o sensor e o objeto.

Ao integrar o sensor HC-SR04 num projeto de segurança residencial, é possível posicionar o sensor em pontos estratégicos, como portas, janelas ou áreas externas, para monitorar a aproximação de pessoas ou objetos. Quando uma intrusão é detetada, o sistema pode acionar dispositivos de segurança, como alarmes sonoros, câmaras de vigilância ou sistemas de notificação.

O sensor HC-SR04 oferece precisão na medição da distância, o que permite uma detecção confiável de intrusões. No entanto, é importante considerar possíveis limitações, como obstáculos físicos que possam interferir no alcance das ondas ultrassónicas e interferências externas que possam prejudicar a interpretação dos sinais capturados pelo sensor.

Para otimizar a detecção de intrusão usando o sensor HC-SR04, é recomendável realizar testes e ajustes adequados. Isso envolve encontrar a posição ideal do sensor, considerar obstáculos físicos e ajustar os limites de distância para distinguir intrusões reais de falsos alarmes. Além disso, a integração do sensor com outros componentes de segurança, como câmaras e sistemas de notificação, permite uma resposta mais abrangente e rápida diante de uma intrusão.

5.2 Captura de imagens com a câmara OV-7670

A câmara OV-7670 é um componente utilizado para captura de imagens em projetos de segurança residencial. Com as suas características e capacidades, ela desempenha um papel importante na obtenção de informações visuais para auxiliar na deteção de intrusões e monitoramento do ambiente.

A câmara OV-7670 é uma câmara de baixo custo que possui uma matriz de pixéis e é capaz de captar imagens em formato VGA (640x480 pixéis). Ela é especialmente projetada para aplicações em sistemas embarcados e oferece uma interface fácil de usar com placas como o Arduino.

No contexto de um projeto de segurança residencial, a câmara OV-7670 pode ser integrada ao sistema para captar imagens em momentos específicos, como quando uma intrusão é detetada ou em intervalos regulares para monitorar o ambiente. Essas imagens podem ser usadas para análise posterior, identificação de intrusos ou até mesmo para fornecer evidências em caso de ocorrências indesejadas.

A integração da câmara OV-7670 com outros componentes, como sensores de deteção de intrusão e sistemas de notificação, permite uma abordagem mais completa para a segurança residencial. Por exemplo, quando uma intrusão é detetada pelo sensor HC-SR04, a câmara OV-7670 pode ser acionada para captar imagens da área afetada. Essas imagens podem ser armazenadas, exibidas em tempo real ou enviadas para dispositivos remotos para análise e tomada de decisões.

É importante mencionar que a câmara OV-7670 possui algumas limitações em termos de qualidade de imagem e capacidade de processamento. Ela é mais adequada para aplicações simples de captura de imagens e pode não oferecer recursos avançados, como reconhecimento facial ou deteção de movimento. No entanto, para a maioria dos cenários de segurança residencial, a sua resolução e capacidade de captura são suficientes.

5.3 Acionamento do alarme buzzer

O alarme buzzer é um componente importante num sistema de segurança residencial, pois desempenha o papel de alertar os residentes e possíveis intrusos sobre a deteção de uma intrusão ou evento de segurança. O alarme buzzer é um dispositivo sonoro que emite um som alto e audível para chamar a atenção das pessoas próximas.

No contexto do projeto de segurança residencial, o acionamento do alarme buzzer ocorre quando uma intrusão é detetada por meio de sensores, como o sensor HC-SR04. Quando o sistema identifica a presença de um intruso, um sinal é enviado para o alarme buzzer, que emite um som alto e estridente para alertar os residentes sobre a situação.

O alarme buzzer pode ser instalado em locais estratégicos, como dentro da residência ou em áreas externas, para garantir que o som seja audível em todos os ambientes. Ele pode ser conectado a um circuito de controle, como uma placa Arduino, que recebe o sinal de deteção de intrusão e aciona o buzzer.

Além de alertar os residentes, o alarme buzzer também pode ter a função de dissuadir os intrusos. O som alto e estridente pode assustar o intruso, levando-o a abandonar a tentativa de invasão ou a atrair a atenção de vizinhos e pessoas próximas, aumentando as hipóteses de intervenção rápida e a captura do intruso.

É importante considerar a intensidade e o padrão do som emitido pelo alarme buzzer para garantir a sua eficácia. O som deve ser alto o suficiente para ser ouvido em toda a residência e nas proximidades, mas também deve ser diferenciado o bastante para ser reconhecido como um alarme de segurança.

Além disso, é necessário levar em conta a possibilidade de acionamento acidental do alarme buzzer, seja por falsos alarmes ou por erros no sistema. Medidas de segurança devem ser implementadas para evitar acionamentos indevidos, como a implementação de mecanismos de confirmação de intrusão ou a utilização de códigos de acesso para desativar o alarme.

5.4 Notificação no telefone

A notificação no telefone é uma funcionalidade essencial num sistema de segurança residencial, pois permite que os residentes recebam alertas e informações sobre eventos de segurança em tempo real, mesmo quando estão fora de casa. Com o uso de tecnologias de comunicação, como redes de celular ou Wi-Fi, é possível enviar notificações instantâneas para dispositivos móveis, como smartphones.

Quando um evento é detetado, o sistema de segurança pode enviar uma notificação para o telefone cadastrado, informando sobre a ocorrência. Essas notificações podem ser enviadas por meio de aplicativos móveis dedicados como mensagens de notificação *push*.

As notificações no telefone permitem que os residentes tomem medidas imediatas, como acionar autoridades de segurança, verificar câmaras de vigilância remotamente, entrar em contacto com vizinhos ou até mesmo retornar à residência para lidar com a situação. Essa comunicação em tempo real é essencial para garantir a eficácia do sistema de segurança e a proteção da residência.

É importante ressaltar que a notificação no telefone requer uma conexão estável com a internet, seja por meio de uma rede celular (3G/4G/5G) ou Wi-Fi. Além disso, é necessário configurar adequadamente o sistema de segurança para enviar as notificações corretamente, considerando aspeto como a configuração do aplicativo móvel, a integração com a plataforma de comunicação e a gestão das permissões de notificação.

5.5 Armazenamento de dados na base de dados

O armazenamento de dados numa base de dados é uma funcionalidade importante num sistema de segurança residencial, pois permite registrar e manter um histórico das atividades e eventos ocorridos na residência. Isso inclui informações sobre detecções de intrusão, registos de acesso, ocorrências de falhas ou incidentes, entre outros dados relevantes para a segurança.

No contexto do projeto de segurança residencial, uma base de dados, como o PostgreSQL, é normalmente utilizada para armazenar os dados coletados pelos diferentes componentes do sistema, como sensores, câmaras e dispositivos de controle. Esses dados podem incluir registos de eventos, imagens capturadas, informações sobre utilizadores autorizados, configurações do sistema, entre outros.

Ao armazenar os dados numa base de dados, é possível aceder aos mesmos posteriormente para análise, geração de relatórios e tomada de decisões. Por exemplo, é possível identificar padrões de atividade, verificar históricos de acessos, analisar estatísticas de detecção de intrusões ou investigar incidentes passados. Essas informações podem ser valiosas para

aprimorar o sistema de segurança, ajustar configurações, identificar vulnerabilidades ou até mesmo fornecer evidências em caso de ocorrências indesejadas.

Além disso, a base de dados pode ser utilizada para realizar integrações com outros sistemas e aplicativos. Por exemplo, é possível integrar a base de dados com um aplicativo móvel ou uma plataforma de monitoramento remoto, permitindo que os utilizadores acedam informações em tempo real, visualizem imagens das câmaras de vigilância ou recebam notificações de eventos diretamente nos seus dispositivos móveis.

Para garantir a segurança e integridade dos dados, é importante implementar medidas de proteção adequadas na base de dados, como autenticação de utilizadores, criptografia de dados, controle de acesso e backups regulares. Isso assegura que as informações armazenadas estejam protegidas contra acessos não autorizados, perdas acidentais ou corrupção dos dados.

5.6 Controle Remoto do Sistema de Segurança

O controle remoto do sistema de segurança é uma funcionalidade que permite aos utilizadores gerir e controlar o sistema de segurança residencial de forma conveniente e eficiente, mesmo quando estão distantes da residência. Por meio de um dispositivo remoto, como um smartphone, tablet ou computador, os utilizadores podem aceder e controlar diferentes aspetos do sistema de segurança.

Com o controle remoto, os utilizadores têm a capacidade de ativar ou desativar o sistema de segurança de maneira rápida e conveniente. Isso permite que eles habilitem a proteção quando saem de casa ou desativem o sistema temporariamente, por exemplo, quando estão a esperar a chegada de um familiar ou de um prestador de serviços.

Além disso, o controle remoto possibilita a gestão de outros componentes do sistema, como câmaras de vigilância, sensores de movimento e alarmes. Os utilizadores podem visualizar imagens em tempo real das câmaras instaladas na residência, verificar o status dos sensores e receber notificações de eventos de segurança diretamente nos seus dispositivos remotos.

O controle remoto também pode oferecer recursos avançados, como a programação de cenários ou modos de segurança personalizados. Os utilizadores podem definir configurações específicas para diferentes situações, como o modo de segurança ativado durante a noite, que aciona sensores de movimento em áreas específicas da residência e mantém as câmaras em modo de gravação contínua.

Além disso, a integração do controle remoto com outros dispositivos inteligentes na residência é uma possibilidade. Por exemplo, os utilizadores podem conectar o sistema de segurança a dispositivos de automação residencial, como lâmpadas, fechaduras e termostatos, permitindo o controle desses dispositivos por meio do mesmo aplicativo ou interface remota.

Para garantir a segurança do controle remoto, é fundamental implementar medidas de segurança adequadas, como autenticação de utilizadores, criptografia de dados e acesso seguro aos dispositivos remotos. Isso protege contra acessos não autorizados e garante que apenas os utilizadores autorizados possam controlar o sistema de segurança.

6 Resultados e Discussões

Neste tópico, serão apresentados os resultados obtidos e as discussões realizadas durante a implementação e teste do protótipo do Sistema Inteligente para segurança residencial. O objetivo é analisar e avaliar o desempenho do sistema, verificando se ele cumpriu com os objetivos propostos e discutindo os principais aspectos observados. Os resultados são essenciais para compreender o funcionamento do protótipo, verificar a eficácia das funcionalidades implementadas e identificar possíveis melhorias. Além disso, as discussões proporcionam um espaço para refletir sobre as limitações encontradas, as dificuldades enfrentadas durante o desenvolvimento e as soluções adotadas.

6.1 Testes e Validação das Funcionalidades

Os testes e a validação das funcionalidades são etapas essenciais no desenvolvimento de um sistema de segurança residencial para garantir o seu funcionamento adequado, confiabilidade e eficácia. Essas atividades visam identificar e corrigir possíveis falhas, validar o desempenho do sistema e garantir que todas as funcionalidades estejam a operar conforme o esperado.

Durante os testes, cada funcionalidade do sistema é avaliada individualmente para verificar se ela atende aos requisitos estabelecidos. Por exemplo, no caso da detecção de intrusão usando o sensor HC-SR04, é realizado um teste específico para verificar se o sensor é capaz de detectar a presença de objetos ou pessoas na área monitorada com precisão e confiabilidade. Os testes também são realizados em diferentes cenários e condições, a fim de garantir a robustez do sistema.

Além dos testes individuais, é importante realizar testes de integração, nos quais as diferentes funcionalidades do sistema são combinadas e testadas em conjunto para verificar a sua interação e interoperabilidade. Por exemplo, testes que verificam se a detecção de intrusão aciona corretamente o acionamento do alarme buzzer e o envio de notificações para o telefone cadastrado.

A validação das funcionalidades envolve a avaliação do sistema em condições reais de uso, simulando situações e cenários reais. Por exemplo, é possível simular uma tentativa de intrusão para verificar se o sistema de segurança responde adequadamente, acionando o alarme buzzer, captando imagens da câmara e enviando notificações para o telefone. Esses testes de validação permitem verificar a eficácia do sistema em proteger a residência e fornecer a segurança desejada.

Além disso, é importante envolver os utilizadores reais do sistema durante os testes e a validação, solicitando o seu feedback e avaliação. Isso permite obter intuições valiosas sobre a usabilidade, facilidade de uso e satisfação geral do sistema. Os comentários dos utilizadores podem ser usados para aprimorar o sistema, corrigir possíveis falhas e atender às necessidades específicas dos utilizadores.

6.2 Desempenho do Sistema

A avaliação do desempenho do sistema de segurança residencial é uma etapa importante para garantir que o sistema atenda às necessidades e expectativas dos utilizadores. Durante essa avaliação, são analisados diferentes aspetos do desempenho do sistema, como tempo de resposta, capacidade de processamento, consumo de recursos, escalabilidade e confiabilidade.

Um dos principais aspetos a serem considerados é o tempo de resposta do sistema, ou seja, o tempo que leva para o sistema reagir a eventos, como deteção de intrusão. É importante que o sistema responda de forma rápida e eficiente, acionando alarmes, captando imagens e notificando os utilizadores em tempo hábil. A latência do sistema deve ser minimizada para garantir uma resposta adequada em emergências.

Outro aspeto relevante é a capacidade de processamento do sistema. Isso envolve a análise da capacidade do hardware e dos componentes utilizados no sistema para lidar com as demandas de processamento de dados. Por exemplo, a capacidade de processamento da câmara OV-7670 para captar imagens de forma eficiente e transmiti-las para o sistema central. É importante garantir que o sistema seja dimensionado adequadamente para lidar com as tarefas exigidas.

O consumo de recursos, como energia, memória e largura de banda, também deve ser avaliado. É importante que o sistema seja projetado de forma eficiente para minimizar o consumo de recursos, garantindo o uso adequado dos dispositivos e a economia de energia. Isso é especialmente relevante para sistemas alimentados por bateria ou que possuam restrições de recursos.

A escalabilidade é outro aspeto a ser considerado no desempenho do sistema. O sistema deve ser capaz de lidar com um aumento na quantidade de dispositivos e utilizadores conectados, sem comprometer a sua funcionalidade e desempenho. É importante que o sistema possa ser facilmente expandido para acomodar um crescimento futuro.

Por fim, a confiabilidade do sistema é fundamental. O sistema de segurança residencial deve ser confiável e operar de forma estável, sem falhas ou interrupções frequentes. É importante realizar testes de estabilidade e confiabilidade para identificar e corrigir possíveis problemas.

Durante a avaliação do desempenho do sistema, é necessário utilizar métricas e indicadores relevantes para medir o desempenho em cada um dos aspetos mencionados. Isso pode ser feito por meio de testes de carga, simulações de cenários reais e monitoramento contínuo do sistema em operação.

6.3 Limitações e Possíveis Melhorias

Ao desenvolver um sistema de segurança residencial, é importante reconhecer as suas limitações e identificar possíveis melhorias para aprimorar o desempenho e a funcionalidade do sistema. Aqui estão algumas limitações comuns que podem ser encontradas, juntamente com sugestões de melhorias:

- Alcance dos sensores: Dependendo do tipo de sensor utilizado, pode haver limitações no seu alcance. Por exemplo, o sensor HC-SR04 usado para deteção de intrusão tem um alcance máximo. Uma melhoria possível seria a

implementação de uma rede de sensores sem fio para estender o alcance e cobertura do sistema.

- Precisão dos sensores: Os sensores podem ter uma margem de erro ou falsos positivos/negativos, o que pode comprometer a eficácia do sistema. É recomendável realizar testes e ajustes nos sensores para melhorar a sua precisão e minimizar falsos alarmes ou omissões.
- Capacidade de armazenamento: Dependendo da quantidade de câmaras e da frequência de captura de imagens, o sistema pode enfrentar limitações de capacidade de armazenamento. Uma solução seria utilizar sistemas de armazenamento em nuvem ou implementar técnicas de compactação de dados para otimizar o uso do armazenamento.
- Segurança da rede: A segurança da rede é fundamental para proteger o sistema contra ataques cibernéticos e acessos não autorizados. Melhorias podem ser feitas implementando protocolos de criptografia, autenticação de dispositivos e monitoramento contínuo da rede para identificar possíveis vulnerabilidades.
- Integração com outros sistemas: Para fornecer uma solução abrangente de segurança residencial, é importante que o sistema possa integrar-se com outros dispositivos e sistemas, como sistemas de automação residencial, assistentes virtuais e dispositivos móveis. Melhorias podem ser feitas para aprimorar a interoperabilidade e a integração com outros sistemas.
- Interface do utilizador: A usabilidade e a interface do utilizador podem ser melhoradas para garantir uma experiência intuitiva e fácil de usar. Isso pode envolver o design de uma interface amigável, recursos de personalização e a inclusão de tutoriais ou ajuda contextual para orientar os utilizadores.
- Automação avançada: Uma melhoria adicional seria a implementação de recursos de automação avançada, como aprendizado de máquina e inteligência artificial. Isso permitiria que o sistema identificasse padrões de comportamento suspeitos, adaptasse-se às preferências dos utilizadores e fornecesse recomendações de segurança personalizadas.

Para além destas limitações e possíveis melhorias no futuro, não conseguimos implementar a câmara. Mas até à data da apresentação iremos fazer todos os possíveis para que tenhamos isso implementado e a funcionar da melhor maneira possível.

7 Considerações Finais

É nesta secção que se faz uma reflexão final sobre o trabalho realizado, abordando tanto os aspetos positivos quanto os desafios encontrados. As considerações finais fornecem uma visão geral do projeto, resumindo os resultados e conclusões mais relevantes, refletindo sobre os desafios enfrentados e oferecendo sugestões para estudos futuros.

7.1 Recapitulação dos Objetivos Alcançados

Durante o desenvolvimento deste projeto de segurança residencial, foram estabelecidos objetivos claros e definidos, os quais foram alcançados com sucesso. Esses objetivos incluíram:

Projeto e implementação de um sistema de segurança residencial: O objetivo principal foi desenvolver um sistema eficiente e confiável que garantisse a segurança de uma residência. Isso envolveu a escolha dos componentes adequados, a integração entre eles e a implementação das funcionalidades necessárias para detetar intrusões, captar imagens, acionar alarmes e notificar os utilizadores.

Integração dos componentes do sistema: Um dos desafios era garantir que todos os componentes do sistema, como Arduino UNO-WIFI-REV2, sensor HC-SR04, câmara OV-7670, alarme buzzer, telefone, base de dados com Docker e PostgreSQL, e a API em Node.js, pudessem se comunicar e funcionar em conjunto. Foi necessário estabelecer as conexões apropriadas, configurar os dispositivos e desenvolver a lógica de integração.

Testes e validação das funcionalidades: Para garantir a eficácia e o desempenho do sistema, foram realizados testes e validação das funcionalidades implementadas. Isso incluiu testes de deteção de intrusão, captura de imagens, acionamento do alarme buzzer e notificação no telefone. Os resultados desses testes foram analisados para garantir que o sistema estivesse a funcionar conforme o esperado.

Configuração do ambiente de desenvolvimento: Foi necessário configurar o ambiente de desenvolvimento adequado para o projeto, incluindo a instalação e configuração dos componentes e software necessários. Isso envolveu a configuração do Arduino IDE, a instalação dos drivers necessários para os componentes e a configuração do ambiente de desenvolvimento Node.js.

Ao final do projeto, todos esses objetivos foram alcançados com sucesso. O sistema de segurança residencial foi implementado e as funcionalidades essenciais foram desenvolvidas e integradas corretamente. Os testes e validação realizados demonstraram que o sistema estava a funcionar conforme o esperado, cumprindo o seu propósito de garantir a segurança de uma residência.

7.2 Lições Aprendidas

Durante o desenvolvimento deste projeto de segurança residencial, uma série de lições valiosas foram aprendidas, contribuindo para o conhecimento e aprimoramento dos envolvidos. Algumas das lições aprendidas incluem:

- **Importância da análise e compreensão dos requisitos:** foi essencial realizar uma análise completa e clara dos requisitos do sistema, considerando as necessidades

dos utilizadores e as restrições do ambiente residencial. Uma compreensão sólida dos requisitos ajudou a orientar as decisões de projeto.

- **Planeamento adequado:** O planeamento detalhado foi fundamental para o sucesso do projeto. Isso incluiu a definição de marcos, o estabelecimento de prazos realistas, a alocação adequada de recursos e a identificação de dependências. Um planeamento adequado ajudou a garantir que as etapas do projeto fossem executadas de forma eficiente e dentro do cronograma.
- **Necessidade de flexibilidade e adaptação:** Durante o desenvolvimento, surgiram desafios e obstáculos inesperados. Foi importante ser flexível e capaz de se adaptar às mudanças de requisitos, às limitações técnicas e a outras circunstâncias imprevistas. A capacidade de encontrar soluções alternativas e contornar problemas foi essencial para o progresso contínuo do projeto.
- **Importância dos testes e validação:** A realização de testes abrangentes e validação das funcionalidades foi crucial para garantir o funcionamento adequado do sistema. Os testes permitiram identificar e corrigir problemas antes da implantação, garantindo a qualidade e a confiabilidade do sistema.
- **Documentação adequada:** A documentação clara e abrangente desempenhou um papel fundamental no projeto. Isso incluiu a documentação dos requisitos, a criação de diagramas de arquitetura, a documentação do código-fonte e a elaboração de manuais de utilizador. Uma documentação adequada facilitou a compreensão do projeto e a manutenção futura do sistema.
- **Trabalho em equipa e colaboração:** O projeto exigiu uma colaboração efetiva entre os membros da equipa. A comunicação clara e o compartilhamento de conhecimentos foram essenciais para superar desafios e alcançar os objetivos do projeto.

Essas lições aprendidas podem ser aplicadas em projetos futuros, ajudando a evitar erros comuns, melhorar a eficiência e aumentar as hipóteses de sucesso. A reflexão sobre essas lições é fundamental para o desenvolvimento contínuo das habilidades e conhecimentos dos envolvidos no projeto de segurança residencial.

7.3 Recomendações para Projetos Futuros

Com base nas experiências adquiridas durante o desenvolvimento deste projeto de segurança residencial, algumas recomendações podem ser feitas para projetos futuros semelhantes:

- Realizar uma análise aprofundada dos requisitos do projeto, considerando as necessidades específicas dos utilizadores e as limitações do ambiente residencial.
- Explorar tecnologias emergentes e tendências no campo da segurança residencial, como inteligência artificial, aprendizado de máquina e reconhecimento facial, para aprimorar as funcionalidades do sistema.

- Garantir a segurança do sistema, implementando práticas robustas de segurança da informação, como criptografia, autenticação forte e monitoramento contínuo da rede.
- Investir na usabilidade e na experiência do utilizador, projetando interfaces intuitivas e fornecendo recursos de personalização para atender às preferências individuais dos utilizadores.
- Considerar a escalabilidade do sistema desde o início, permitindo a adição de novos componentes e a integração com outros sistemas no futuro.
- Promover a modularidade e a reutilização de componentes, facilitando a manutenção e a expansão do sistema.
- Realizar testes abrangentes e validação do sistema em diferentes cenários e condições para garantir o seu desempenho e confiabilidade.
- Manter uma documentação atualizada do projeto, incluindo diagramas de arquitetura, manuais de utilizador e código-fonte comentado, para facilitar a compreensão e a manutenção do sistema ao longo do tempo.

Essas recomendações podem contribuir para o sucesso de projetos futuros de segurança residencial, permitindo o desenvolvimento de sistemas mais avançados, seguros e eficientes.

8 Conclusões

Ao longo do trabalho, foram discutidos diversos tópicos relevantes. Na fundamentação teórica, abordamos os sistemas de segurança residencial, destacando as vantagens da IoT e explorando casos de uso. Em seguida, detalhamos a estrutura do código, com o uso do Arduino, API em Node.js, aplicação em Flutter e banco de dados Docker (SQL).

Além disso, apresentamos o mecanismo de comunicação do sistema, envolvendo a comunicação via Wi-Fi entre o Arduino e a API, bem como o uso de *sockets* para a comunicação entre a API, a aplicação móvel e o banco de dados.

Discutimos também os componentes físicos do sistema, como sensores HC-SR04, câmara OV7670, breadboard, fios, LEDs, Arduino REV2 WIFI e buzzer, que desempenham papéis fundamentais na detecção de movimento, capta de imagens e ativação de dispositivos de segurança.

No que diz respeito à metodologia do modelo proposto, detalhamos a construção da maquete, a implementação física do protótipo e a implementação da aplicação móvel. Cada etapa envolveu a definição de requisitos, desenvolvimento do código, testes e integração com outros componentes do sistema.

Por fim, abordamos o histórico do sistema, que permite aos utilizadores aceder e rever os eventos registados ao longo do tempo, fornecendo informações valiosas para análise e melhoria contínua da segurança residencial.

Em conclusão, o modelo proposto apresenta uma solução completa e integrada para um sistema de segurança residencial, abrangendo desde a construção física até a interface de utilizador da aplicação móvel. O ambiente de desenvolvimento estabelecido utilizando o IntelliJ e o VSCode permitiu uma implementação eficiente e organizada do sistema.

O trabalho proporcionou uma visão abrangente dos aspetos técnicos e funcionais do sistema, destacando a importância da integração dos componentes físicos, da comunicação entre os diferentes elementos e do histórico para a análise e monitoramento da segurança residencial.

9 Bibliografia

- [1] P. Ellis, “How Nelson Mandela created the conditions for success,” Cambridge Assessment International Education, 28 março 2019. [Online]. Available: <https://blog.cambridgeinternational.org/nelson-mandela/>. [Acedido em 23 junho 2023].
- [2] L. Azamfirei, “Knowledge is Power,” The Journal of Critical Care Medicine, 9 maio 2016. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5939128/>. [Acedido em 23 junho 2023].
- [3] A. Bento, “NÓS SOMOS CAPAZES DE FAZER O NOSSO MELHOR! WE ARE ABLE TO DO OUR BEST!,” 2020. [Online]. Available: <http://nossomoscapazes.blogspot.com/2020/06/persistence-is-path-to-success-charles.html>. [Acedido em 23 junho 2023].
- [4] “Blog iad,” 25 fevereiro 2022. [Online]. Available: <https://blog.iadportugal.pt/noticias-imobiliario/mercado-imobiliario/nacional/o-que-e-a-domotica-e-como-torna-as-casas-mais-inteligentes/>. [Acedido em 5 junho 2023].
- [5] “www.sislite.pt,” [Online]. Available: <https://www.sislite.pt/domus.htm>. [Acedido em 6 junho 2023].
- [6] R. Cronapp, “Cronapp,” 25 novembro 2019. [Online]. Available: https://blog.cronapp.io/desenvolvimento-iterativo-e-incremental/#O_que_e_desenvolvimento_iterativo_e_desenvolvimento_incremental. [Acedido em 6 junho 2023].
- [7] “imsi03.blogs.sapo.pt,” [Online]. Available: <https://imsi03.blogs.sapo.pt/8314.html>. [Acedido em 6 junho 2023].
- [8] 6 janeiro 2021. [Online]. Available: <https://blog.kalatec.com.br/arduino-o-que-e/>. [Acedido em 7 junho 2023].
- [9] “Curto Circuito,” [Online]. Available: <https://curtocircuito.com.br/sensor-ultrassonico-hc-sr04.html>. [Acedido em 7 junho 2023].

<https://www.desterroeletricidade.com.br/blog/sistema-de-seguranca/casa-inteligente-a-iot-no-futuro-da-automacao-residencial/>. [Acedido em 9 junho 2023].

[21 “Arduino Official Store,” [Online]. Available:
] <https://store.arduino.cc/products/arduino-uno-wifi-rev2>. [Acedido em 10 junho 2023].

[22 ArduinoPortugal.pt, “Arduino Portugal,” 22 março 2017. [Online].
] Available: <https://www.arduinoportugal.pt/o-que-e-arduino/>. [Acedido em 10 junho 2023].

[23 “Blog da Trybe,” 1 novembro 2021. [Online]. Available:
] <https://blog.betrybe.com/tecnologia/arduino-tudo-sobre/>. [Acedido em 10 junho 2023].

[24 “Blog da Trybe,” 13 outubro 2021. [Online]. Available:
] <https://blog.betrybe.com/linguagem-de-programacao/cpp/>. [Acedido em 10 junho 2023].

[25 “acervolima.com,” [Online]. Available: <https://acervolima.com/recursos-do-c/>. [Acedido em 10 junho 2023].

[26 Eletrogate, “Blog Eletrogate,” 24 maio 2022. [Online]. Available:
] <https://blog.eletrogate.com/placa-de-desenvolvimento-arduino-uno-wifi/>. [Acedido em 10 junho 2023].

[27 “www.sta-eletronica.com.br,” [Online]. Available: <https://www.sta-eletronica.com.br/artigos/arduinos/sensor-ultrassonico-hc-sr04>. [Acedido em 11 junho 2023].

[28 A. Thomsen, “MakerHero,” 19 novembro 2013. [Online]. Available:
] <https://www.makerhero.com/blog/modulo-camera-vga-ov7670/>. [Acedido em 11 junho 2023].

[29 F. D. Garcia, “Embarcados - Sua fonte de informações sobre Sistemas Embarcados,” 12 julho 2018. [Online]. Available:
] <https://embarcados.com.br/modulo-ov7670-fifo-interface-sccb/>. [Acedido em 11 junho 2023].

- [30 [Online]. Available:
] <https://www.blogdarobotica.com/2020/10/05/utilizando-o-buzzer-ativo-no-arduino/>. [Acedido em 12 junho 2023].
- [31 R. Nogueira, “BC END - Equipamentos para Ensaios Não Destrutivos,” 25
] outubro 2011. [Online]. Available: <https://bcend.com.br/o-que-e-efeito-piezoeletrico/>. [Acedido em 12 junho 2023].
- [32 “Alura,” [Online]. Available: <https://www.alura.com.br/artigos/flutter>.
] [Acedido em 12 junho 2023].
- [33 Isaac, “Hardware libre,” 1 fevereiro 2022. [Online]. Available:
] <https://www.hwlibre.com/pt/cable-jumper/#Tipos>. [Acedido em 17 junho 2023].
- [34 “Opencircuit,” 14 abril 2023. [Online]. Available:
] <https://opencircuit.pt/blog/breadboards-beginners-tips-tricks>. [Acedido em 17 junho 2023].
- [35 “www.sciencedirect.com,” [Online]. Available:
] <https://www.sciencedirect.com/topics/engineering/breadboard>. [Acedido em 17 junho 2023].
- [36 “Oficina da Net,” 6 julho 2020. [Online]. Available:
] <https://www.oficinadanet.com.br/hardware/31696-o-que-e-e-para-que-serve-o-cabo-usb-tipo-b>. [Acedido em 17 junho 2023].
- [37 “curtocircuito.com.br,” [Online]. Available:
] <https://curtocircuito.com.br/blog/eletronica-basica/o-que-e-um-led>. [Acedido em 18 junho 2023].
- [38 “Sala da Elétrica,” 21 agosto 2017. [Online]. Available:
] <https://www.saladaeletrica.com.br/como-funcionam-as-lampadas-leds/>. [Acedido em 18 junho 2023].
- [39 “www.ic.uff.br,” [Online]. Available:
] <http://www.ic.uff.br/~aconci/componentes.html>. [Acedido em 18 junho 2023].
- [40 D. C, “Hostinger Tutoriais,” 25 outubro 2022. [Online]. Available:
] <https://www.hostinger.com.br/tutoriais/o-que-e-docker>. [Acedido em 13 junho 2023].
- [41 “Rock Content - BR,” 4 agosto 2020. [Online]. Available:
] <https://rockcontent.com/br/blog/postgresql/>. [Acedido em 13 junho 2023].

- [42 F. Jânio, “Gitbooks.io,” 2018. [Online]. Available:
] <https://fabiojaniolima.gitbooks.io/banco-de-dados-modelagem-de-dados/content/capitulo-1/1.6-ACID-atomicidade-consistencia-isolamento-e-durabilidade.html>. [Acedido em 13 junho 2023].
- [43 R. X. Educação, “XP Educação,” 4 outubro 2022. [Online]. Available:
] <https://blog.xpeducacao.com.br/banco-de-dados-postgresql/>. [Acedido em 13 junho 2023].
- [44 “www.redhat.com,” [Online]. Available: <https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces>. [Acedido em 14 junho 2023].
- [45 lenon, “Opus Software,” 5 setembro 2018. [Online]. Available:
] <https://www.opus-software.com.br/index.html%3Fp=7232.html#>. [Acedido em 14 junho 2023].
- [46 “Rock Content - BR,” 21 julho 2020. [Online]. Available:
] <https://rockcontent.com/br/blog/node-js/>. [Acedido em 14 junho 2023].
- [47 “Tecnoblog,” [Online]. Available: <https://tecnoblog.net/responde/o-que-e-node-js-guia-para-iniciantes/>. [Acedido em 15 junho 2023].
- [48 “Awari,” 24 janeiro 2023. [Online]. Available: <https://awari.com.br/intelli-j/>. [Acedido em 15 junho 2023].
- [49 “www.treinaweb.com.br,” [Online]. Available:
] <https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar>. [Acedido em 16 junho 2023].
- [50 R. Online, “Remessa News - Notícias Sobre Transferências Internacionais
] e Câmbio,” 26 outubro 2021. [Online]. Available: <https://www.remessaonline.com.br/blog/visual-studio-code-confira-as-principais-funcoes-da-ferramenta/>. [Acedido em 16 junho 2023].
- [51 “wiki.inf.ufpr.br,” [Online]. Available:
] https://wiki.inf.ufpr.br/maziero/doku.php?id=pua:comunicacao_em_rede. [Acedido em 17 junho 2023].

10 Anexos

Nesta secção será apresentado o manual de instalação do projeto, a configuração e utilização da aplicação e todo o código fonte utilizado. Decidimos por bem colocá-las no relatório ao invés de num ficheiro à parte.

10.1 Manual de Instalação

Aqui apresentamos os sites onde vamos buscar os executáveis (as tecnologias necessárias para executar a API, a aplicação e a base de dados):

- API - <https://radixweb.com/blog/installing-npm-and-nodejs-on-windows-and-mac>
- Aplicação - <https://docs.flutter.dev/get-started/install>
- Base de Dados - <https://docs.docker.com/get-docker/>

10.2 Configuração e Utilização da Aplicação

10.2.1 Configuração da Aplicação

Nesta secção demonstramos como configurar o sistema:

- API
 1. Abrir o cmd;
 2. Navegar até ao diretório “**Software-Engineering-Final-Project/Api**”;
 3. Correr “**npm install**”;
 4. Correr “**npm start**”;
 5. Após a realização dos passos anteriores, é suposto aparecer o seguinte:

```
[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node ./bin/www`
```

Figura 20 - API

- Aplicação
 1. Habilitar o developer mode;
 - a. <http://samsung.com/us/support/answer/ANS00087642/> link de como se ativa o developer mode.
 2. Conectar o telemóvel com o computador e ativar o modo transferência de ficheiros;
 3. Abrir o cmd;
 4. Navegar até ao diretório **“Software-Engineering-Final-Project/arduino_security_system”**;
 5. Correr **“flutter devices”** e copiar o ID do dispositivo móvel;
 6. Correr **“flutter run -d [ID]”**;
 7. Desconectar o cabo USB.
 8. Após a realização dos passos anteriores, é suposto aparecer o seguinte:

```

> flutter run -d chrome
Launching lib/main.dart on Chrome in debug mode...
Waiting for connection from debug service on Chrome... 40.0s
This app is linked to the debug service: ws://127.0.0.1:38117/sS0gVxCxKWE=/ws
Debug service listening on ws://127.0.0.1:38117/sS0gVxCxKWE=/ws

🔥 To hot restart changes while running, press "r" or "R".
For a more detailed help message, press "h". To quit, press "q".

A Dart VM Service on Chrome is available at: http://127.0.0.1:38117/sS0gVxCxKWE=
The Flutter DevTools debugger and profiler on Chrome is available at: http://127.0.0.1:9100?uri=http://127.0.0.1:38117/sS0gVxCxKWE=

```

Figura 21 - Flutter

- Base de Dados
 1. Abrir o cmd;
 2. Navegar até ao diretório **“Software-Engineering-Final-Project/PostgresSQL”**;
 3. Correr **“docker-compose build --no-cache”**;
 4. Correr **“docker-compose up -d”**;
 5. Após a realização dos passos anteriores, é suposto aparecer o seguinte:

```

> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
74160134d562  arduino-security-system-postgres-db  "docker-entrypoint.s..."  13 seconds ago  Up 12 seconds  0.0.0.0:5432->5432/tcp, :::5432->5432/tcp
ty-system-postgres-db

```

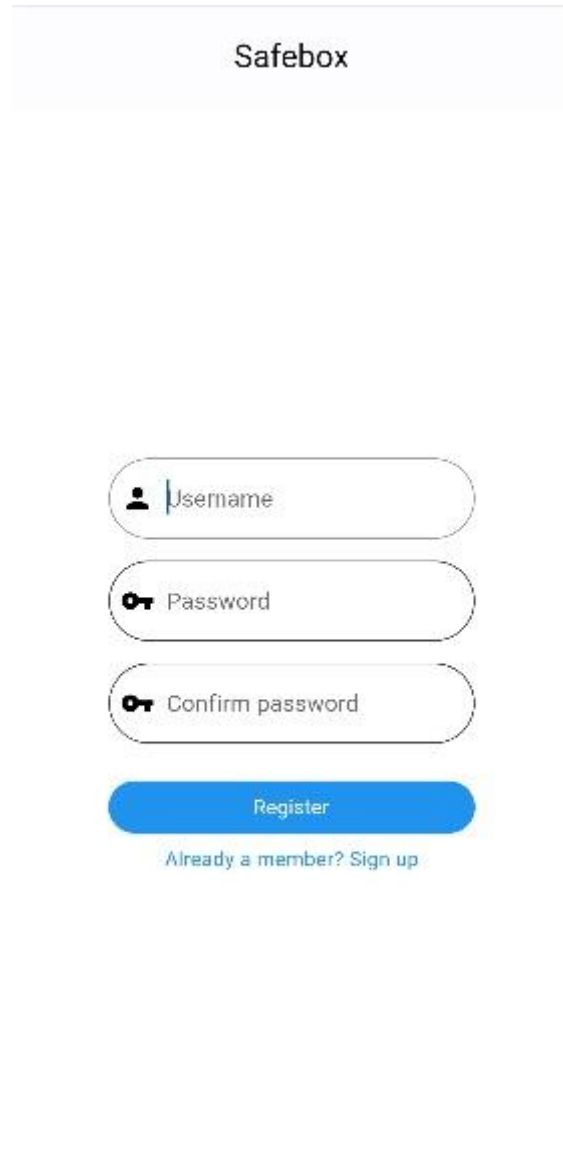
Figura 22 - Base de Dados

- Arduino
 1. Conectar o arduino à fonte.

10.2.2 Utilização da Aplicação

Nesta secção vamos mostrar como é a nossa aplicação e o que se pode fazer nela.

1. Criar o utilizador

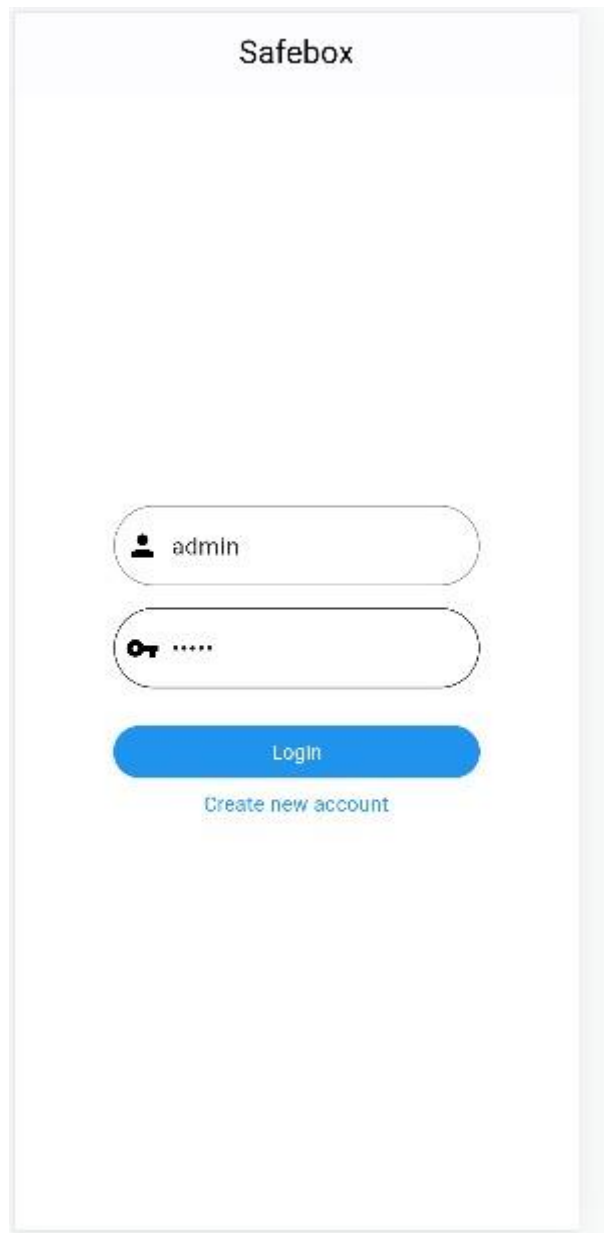


The image shows a mobile application interface for creating a user. At the top, there is a header with the text "Safebox". Below the header, there are three input fields for registration: "Username" (with a person icon), "Password" (with a key icon), and "Confirm password" (with a key icon). Below these fields is a blue "Register" button. At the bottom, there is a link that says "Already a member? Sign up".

Figura 23 - Criar Utilizador

Aqui podemos ver como é a interface quando se vai criar um utilizador.

2. Fazer o login



The image shows a mobile application interface for 'Safebox'. At the top, the word 'Safebox' is displayed in a light grey header. Below this, there are two input fields: the first contains the text 'admin' next to a person icon, and the second contains a password represented by six dots next to a key icon. Below the password field is a prominent blue button labeled 'Login'. Underneath the 'Login' button is a smaller, light blue link that says 'Create new account'.

Figura 24 - Fazer login

Após a criação de um utilizador, esta é a interface do login.

3. Home da aplicação

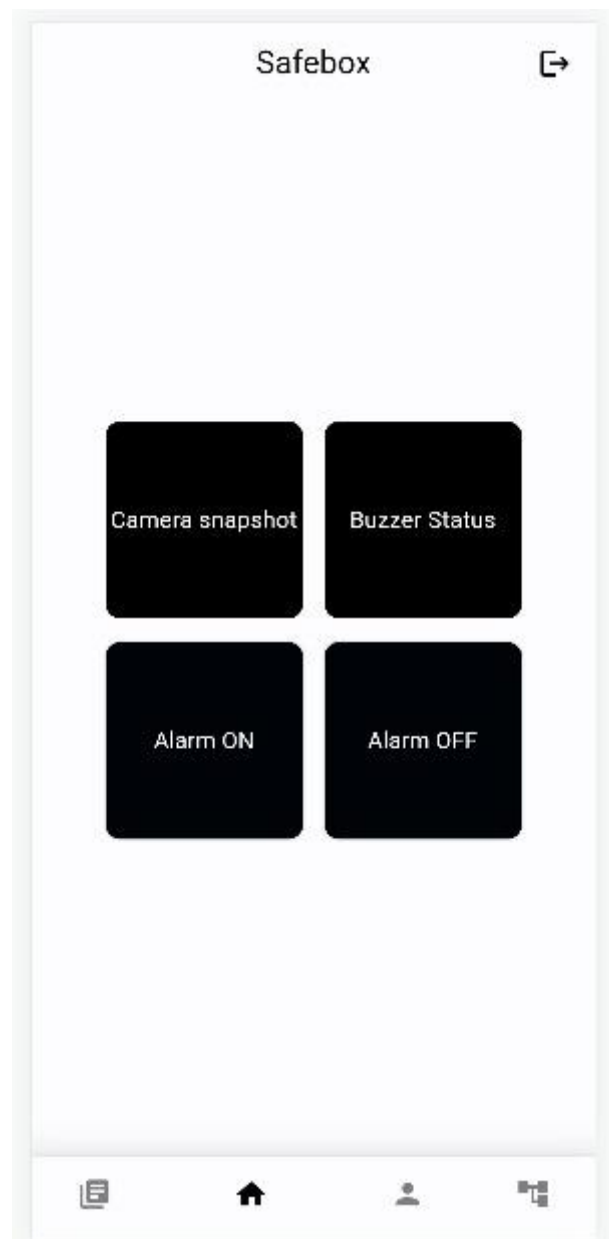


Figura 25 - Home da aplicação

Aqui, podemos visualizar que existem 4 quadrados. Cada quadrado tem a sua própria função no sistema de segurança residencial. O quadrado “Camera Snapshot”, mostra uma imagem em tempo real que é disponibilizada pela Câmara OV-7670. O quadrado “Buzzer Status”, mostra o estado do alarme (se este está “disparado” ou não). Este só se “dispara” quando o utilizador liga o alarme no quadrado “Alarm On” e existe uma deteção de movimento por parte do sensor HC-SR04. Como já referimos anteriormente, o quadrado “Alarm On” é um botão onde o

utilizador ativa o alarme e o quadrado “*Alarm Off*” é um botão também onde o utilizador desativa o alarme.

4. Histórico do alarme

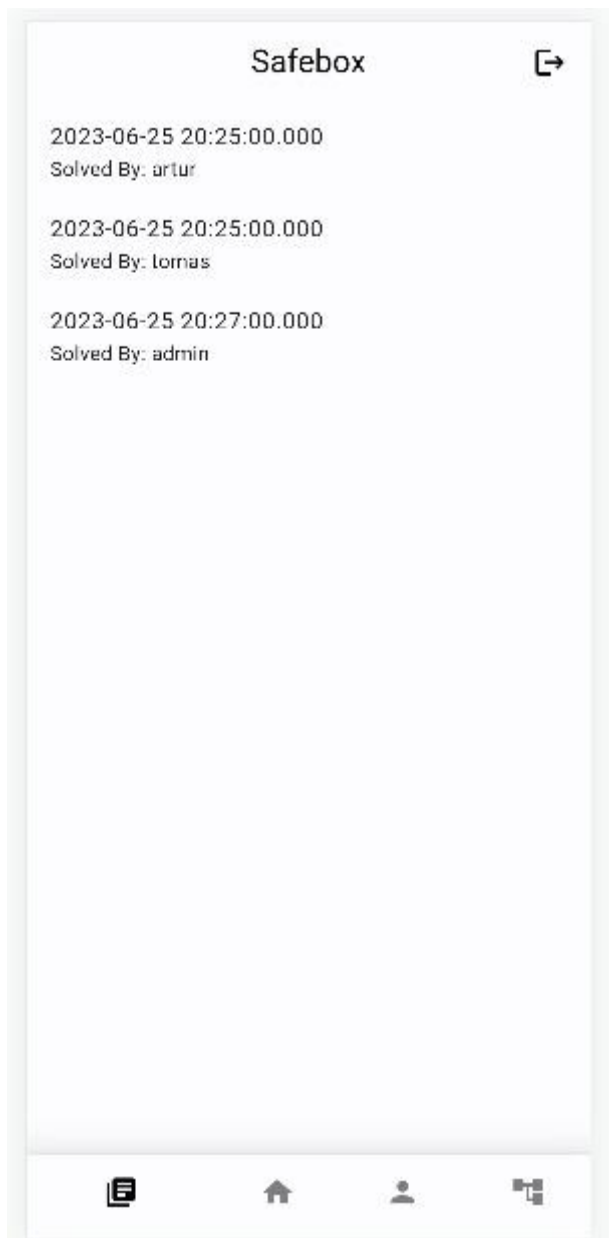


Figura 26 - Historico do Alarme

Nesta página, guardamos o histórico. Quando o alarme é ativado, é nesta página que se regista quando é que o alarme foi ativado. Aparecem informações sobre qual foi o utilizador que ativou o alarme e a que horas é que o ativou.

5. Página de gestão do admin

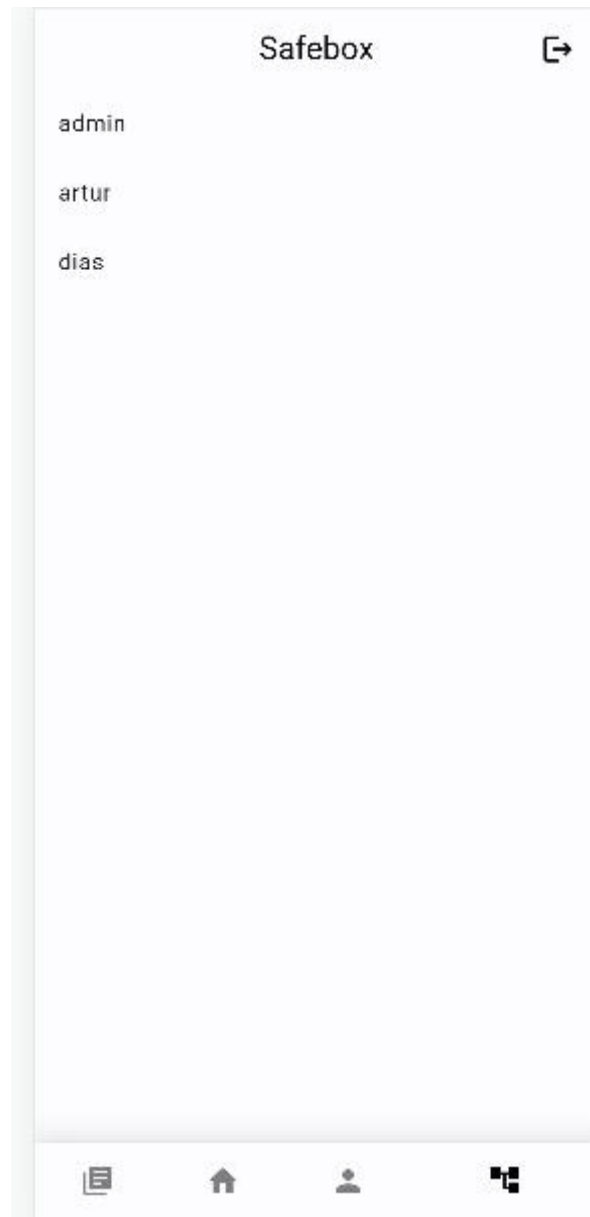


Figura 27 - Pagina de gestão do admin

É só o admin que tem acesso a esta página, podendo assim mudar as palavras-passes de todos os utilizadores.

6. Página de perfil de cada utilizador

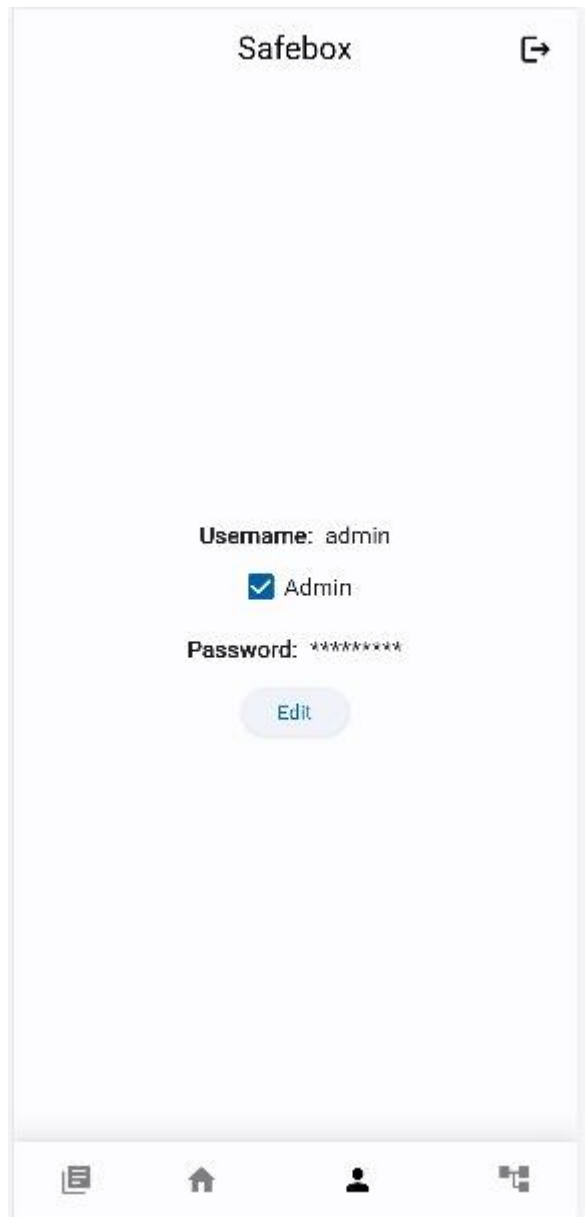


Figura 28 - Página de perfil de cada utilizador

Aqui, o utilizador consegue ver as informações sobre a sua conta. Se este utilizador for o admin, este irá ter um o “checkbox” preenchido. Para além disso, é aqui que o utilizador pode mudar a sua própria palavra-passe e o seu nome de utilizador, caso queira. Para isso basta carregar no botão “edit” e fazer as devidas alterações.

10.3 Código fonte

```
async function validateBearerToken(req, res, next) {
  try {
    const authHeader = String(value: req.headers['authorization'] || '');
    if (!authHeader.startsWith('Bearer ')) {
      throw {code: 401, message: "Invalid bearer token"};
    }

    const token = authHeader.substring(7);
    let session = (await req.db.query(`SELECT *,
                                     CASE
                                       WHEN NOW() >= date_expires THEN true
                                       ELSE false
                                       END AS expired
                                     FROM "arduino-security-system-postgres-db"."user_session"
                                     WHERE token = '${token}'`)).rows[0];

    /*await req.db.knexOne(knex("user_session")
      .select("user_session.*", "username", "auth_admin", knex.raw("NOW() >= date_expire as expired"))
      .join("user", "user.id", "user_session.user")
      .where({token}));*/

    if (session && session.expired) {
      await req.db.query(`DELETE
                          FROM "arduino-security-system-postgres-db"."user_session"
                          WHERE token = '${token}'`);
      await req.db.commit();
      session = null;
    }
  }
}
```

Figura 29 - Validação do login

```
router.get( path: '/', routeWrapper( route: async (req) => {
  let teste = null
  if (req.query.userID)
    return (await req.db.query(`select *
                                from "arduino-security-system-postgres-db".registro_alarme
                                inner join "arduino-security-system-postgres-db"."User"
                                on "arduino-security-system-postgres-db"."User".userid =
                                   "arduino-security-system-postgres-db".registro_alarme.userid`)).rows;
  await req.db.query(`select *
                      from "arduino-security-system-postgres-db".registro_alarme`)
  return teste.rows;
}));
```

Figura 30 - Buscar alarmes

```

router.post( path: '/add',
  routeWrapper( route: async (req) => {
    let existentUser = await req.db.query(`select *
                                          from "arduino-security-system-postgres-db"."User"
                                          where userid = '${req.body.userid}'`)
    if (existentUser.rows.length === 0)
      throw "USER doesn't exists"
    await req.db.query(`INSERT INTO "arduino-security-system-postgres-db".registro_alarma(distancia, userid)
                        VALUES ($1, $2)`, [req.body.distancia.toFixed( fractionDigits: 3),
      req.body.userid]);
    return "Create Alarm";
  }));

```

Figura 31 - Adicionar alarmes

```

router.post( path: '/:alarmID/turnOff',
  body( fields: "userID").trim().matches( pattern: /\d+\/),
  routeWrapper( route: async (req) => {
    let existentAlarm = await req.db.query(`select *
                                          from "existentAlarm"
                                          where registroid = '${req.params.alarmID}'`)
    if (existentAlarm.rows.length === 0)
      throw "ALARM doesn't exists"
    await req.db.query(`update "existentAlarm"
                        SET userID = req.body.userID
                        where registroid = '${req.params.alarmID}'`);
    return "Delete Alarm";
  }));

module.exports = router;

```

Figura 32 - Desligar alarmes

```

router.get( path: '/', validateBearerToken, routeWrapper( route: async (req) => {
  let teste = await req.db.query(`select *
                                  from "arduino-security-system-postgres-db"."User"`)
  return teste.rows;
}));

```

Figura 33 - Buscar todos os alarmes

```

router.post( path: '/authenticate',
  routeWrapper( route: async (req) => {
    //TODO
    let password = sha512.crypt(req.body.password, salt: "$6$rounds=1000$ueCGNzfSS9DT")
    let userLogin = await req.db.query(`select *
                                        from "arduino-security-system-postgres-db"."User"
                                        WHERE password = '${password}'
                                        and username = '${req.body.username}'`)

    if (userLogin.rows.length === 0)
      throw "USER DOESN'T EXISTS"
    let token = crypto.randomUUID();

    let teste = await req.db.query(`INSERT INTO "arduino-security-system-postgres-db"."user_session" (userid, token, date_expires)
                                    VALUES ($1, $2, CURRENT_TIMESTAMP + INTERVAL '5 minutes')`,
    [parseInt(userLogin.rows[0].userid), token]);
    return {...userLogin.rows[0], token};
  }));

```

Figura 34 - Login

```

router.post( path: '/logout_everywhere',
  validateBearerToken,
  routeWrapper( route: async (req) => {
    await req.db.query(`DELETE
                        FROM "arduino-security-system-postgres-db"."user_session"
                        WHERE "userid" = '${req.session.userid}'`);
    return {message: "Success"};
  }));

```

Figura 35 - logout

```

router.post( path: '/create_user',
  body( fields: "username", message: "Invalid username").trim().matches( pattern: /^[a-z]+$),
  body( fields: "password", message: "Invalid password").trim().isLength( options: {min: 1}),
  body( fields: "nif", message: "Invalid nif").trim().isLength( options: {min: 9}),
  body( fields: "morada", message: "Invalid morada"),
  routeWrapper( route: async (req) => {
    let existentUser = await req.db.query(`select *
                                        from "arduino-security-system-postgres-db"."User"
                                        where username = '${req.body.username}'`)

    (function crypt(input: string, salt: string): string (index.d.ts)
    let password = sha512.crypt(req.body.password, salt: "$6$rounds=1000$ueCGNzfSS9DT")
    await req.db.query(`INSERT INTO "arduino-security-system-postgres-db"."User" (nif, "morada", admin, "username", "password")
                        VALUES ($1, $2, $3, $4, $5)`,
    [req.body.nif, req.body.morada, req.body.admin || false, req.body.username, password]);
    let avatar = req.files?.avatar;
    if (avatar) {
      let image = await sharp(avatar.data).resize( widthOrOptions: 150, height: 150,
        options: {
          fit: 'cover',
          position: 'center',
        })
        .toBuffer()
      await createStorage(image, path: `avatar/${req.body.username}`);
    }

    return "create_user Module";
  }));

```

Figura 36 - criar user

```

router.post( path: '/delete_user',
  body( fields: "username", message: "Invalid username").trim().matches( pattern: /^[a-z]+$/),
  validateBearerToken,
  routeWrapper( route: async (req) => {
    let existentUser = await req.db.query(`select *
                                          from "arduino-security-system-postgres-db"."User"
                                          where username = '${req.body.username}'`)
    if (existentUser.rows.length === 0 || existentUser.rows[0]?.active === false)
      throw "USER DOESN'T EXISTS"
    if (req.session.admin === false)
      throw "You dont have permission"
    await req.db.query(`update "arduino-security-system-postgres-db"."User"
                        SET active= false
                        where username = '${req.body.username}'`);
    return "create_user Module";
  }));

```

Figura 37 - apagar user

```

router.post( path: '/update',
  body( fields: "username", message: "Invalid username"),
  body( fields: "password", message: "Invalid password"),
  validateBearerToken,
  routeWrapper( route: async (req) => {
    if (req.session.admin === false && req.session.username !== req.body.username)
      throw "You dont have permission"

    let password = sha512.crypt(req.body.password, {salt: "$6$rounds=1000$ueCGNzfSS9DT"})

    await req.db.query(`update "arduino-security-system-postgres-db"."User"
                        SET "password"= '${password}'
                        where username = '${req.body.username}'`);
    let avatar = req.files?.avatar;
    if (avatar) {
      let image = await sharp(avatar.data).resize( widthOrOptions: 150, height: 150,
        options: {
          fit: 'cover',
          position: 'center',
        })
        .toBuffer()
      await createStorage(image, path: `avatar/${req.body.username}`);
    }
    return "Password updated";
  }));

```

Figura 38 - atualizar user

```

CREATE SCHEMA IF NOT EXISTS "arduino-security-system-postgres-db";

DROP TABLE IF EXISTS "arduino-security-system-postgres-db"."User";

CREATE TABLE IF NOT EXISTS "arduino-security-system-postgres-db"."User"
(
  userid      bigint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 9223372036854775807 CACHE 1 ),
  nif         character varying(9) COLLATE pg_catalog."default",
  morada     character varying(50) COLLATE pg_catalog."default",
  admin      boolean,
  username   character varying(20) COLLATE pg_catalog."default" UNIQUE,
  password   character varying(300) COLLATE pg_catalog."default",
  active     boolean DEFAULT true,
  CONSTRAINT pk_user PRIMARY KEY (userid)
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS "User"
  OWNER to postgres;

```

Figura 39 - Ficheiro SQL (1)

```

DO
$$
BEGIN
  IF NOT EXISTS(SELECT 1 FROM "arduino-security-system-postgres-db"."User" WHERE username = 'admin') THEN
    -- Criar o usuário "admin" com senha criptografada e propriedade "admin" como true
    INSERT INTO "arduino-security-system-postgres-db"."User" (nif, morada, admin, username, password)
    VALUES (241144949, NULL, true, 'admin',
            '$6$rounds=1000$ueCGNzfSS9DT$QZut03yqivh6uJ.PLN2TtBaL/piLVbJ0Te5ghh.47U3b516eVEUeClaweF5yExNIWUqX/m...gt290xD8KLFK/|');
  END IF;
END
$$;

```

Figura 40 - Ficheiro SQL (2)

```

DROP TABLE IF EXISTS "arduino-security-system-postgres-db"."User_session";

CREATE TABLE IF NOT EXISTS "arduino-security-system-postgres-db"."User_session"
(
  id          bigint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 9223372036854775807 CACHE 1 ),
  userid     bigint,
  token      character varying(50) COLLATE pg_catalog."default",
  date_add   TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  date_expires TIMESTAMP,
  CONSTRAINT id PRIMARY KEY (id),
  CONSTRAINT fk_tabela_filho_tabela_pai FOREIGN KEY (userid)
    REFERENCES "arduino-security-system-postgres-db"."User" (userid) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS "User_session"
  OWNER to postgres;

```

Figura 41 - Ficheiro SQL (3)

```

void loop() {
    digitalWrite(greenLedPin, HIGH);
    long duration, distance;

    // Generate a pulse to the trigger pin20
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Measure the duration of the pulse on the echo pin
    duration = pulseIn(echoPin, HIGH);

    // Calculate the distance in centimeters
    distance = duration * 0.034 / 2;

    if (distance < 20) {
        tone(buzzerPin, 3500); // Change the frequency to adjust
        digitalWrite(redLedPin, HIGH);
    } else {
        noTone(buzzerPin);
        digitalWrite(redLedPin, LOW);
    }

    delay(50);
}

```

Figura 43 - Código Arduino

```

static Future<String> login(String username, String password) async {
    final response = await http.post(
        Uri.parse(loginUrl),
        headers: <String, String>{
            'Content-Type': 'application/json; charset=UTF-8',
        },
        body: jsonEncode(<String, String>{
            "username": username,
            "password": password
        })),
    );
    if (response.statusCode == 200) {
        User.fromJson(jsonDecode(response.body));
        return "done";
    }
    return ApiResponse.fromJson(jsonDecode(response.body)).message;
}

```

Figura 42 - Pedido para autenticação do user

```
static Future<String> updateUserPassword(String? username, String password, String? token) async {
  final response = await http.post(
    Uri.parse(updatePasswordUrl),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
      'authorization': 'Bearer $token',
    },
    body: jsonEncode(<String, String>{
      "username": username!,
      "password": password
    })),
  );
  if (response.statusCode == 200) {
    return "done";
  }
  return ApiResponse.fromJson(jsonDecode(response.body)).message;
}
```

Figura 45 - Pedido para troca de password feita pelo admin

```
import ...

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'SafeBox',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.blue),
        useMaterial3: true,
      ), // ThemeData
      home: Login(),
      initialRoute: "/",
      onGenerateRoute: RouteGenerator.generateRoute,
    ); // MaterialApp
  }
}
```

Figura 44 - Função main do Java

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      automaticallyImplyLeading: false,
      title: const Text('Safebox', style: TextStyle(
        color: Colors.black
      )), // TextStyle, Text
      centerTitle: true,
    ), // AppBar
    body: Container(
      width: MediaQuery.of(context).size.width,
      decoration: const BoxDecoration(
        color: Colors.white,
      ), // BoxDecoration
      padding: const EdgeInsets.all(30),
      child: Center(
        child: SizedBox(
          width: MediaQuery.of(context).size.width * 0.65, // MediaQuery.of(context)
          child: SingleChildScrollView(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.stretch,
              children: [
                TextField(
```

Figura 46 - Exemplo do widget