UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE
CAMPUS DE NATAL
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RANIÉLISON SOARES DE OLIVEIRA

SISTEMA DE CONTROLE E GERENCIAMENTO DE ACESSO A AMBIENTES RESTRITOS UTILIZANDO RECONHECIMENTO FACIAL

NATAL
2019

RANIÉLISON SOARES DE OLIVEIRA

SISTEMA DE CONTROLE E GERENCIAMENTO DE ACESSO A AMBIENTES RESTRITOS UTILIZANDO RECONHECIMENTO FACIAL

Trabalho de Conclusão de Curso apresentado à Universidade do Estado do Rio Grande do Norte - UERN - como requisito obrigatório para obtenção do título de Bacharel em Ciência da Computação.

ORIENTADOR**:**
Prof. Dr. Anderson Abner de Santana Souza

NATAL
2019

Catalogação da Publicação na Fonte.
Universidade do Estado do Rio Grande do Norte.

O48s Oliveira, Raniélison Soares de
Sistema de controle e gerenciamento de acesso a ambientes restritos utilizando reconhecimento facial. / Raniélison Soares de Oliveira. - Natal, 2019.
63p.

Orientador(a): Prof. Dr. Anderson Abner de Santana Souza.
Monografia (Graduação em Ciência de Computação). Universidade do Estado do Rio Grande do Norte.

1. Automação. 2. Gerenciamento de acesso. 3. Controle de acesso. 4. ESP. 5. Reconhecimento facial. I. Souza, Anderson Abner de Santana. II. Universidade do Estado do Rio Grande do Norte. III. Título.

RANIÉLISON SOARES DE OLIVEIRA

SISTEMA DE CONTROLE E GERENCIAMENTO DE ACESSO A AMBIENTES RESTRITOS UTILIZANDO RECONHECIMENTO FACIAL

Trabalho de Conclusão de Curso apresentado à Universidade do Estado do Rio Grande do Norte - UERN - como requisito obrigatório para obtenção do título de Bacharel em Ciência da Computação.

Aprovado em:__/__/__.

Banca Examinadora

_____

Prof. Dr. Anderson Abner de Santana Souza
Universidade do Estado do Rio Grande do Norte

_____

Prof. Me. Bruno Cruz de Oliveira
Universidade do Estado do Rio Grande do Norte

_____

Prof. Dra. Rosiery da Silva Maia
Universidade do Estado do Rio Grande do Norte

_____

Prof. Dr. Wilfredo Blanco Figuerola
Universidade do Estado do Rio Grande do Norte

Aos meus pais.

AGRADECIMENTOS

A Deus por ter cuidado de mim ao longo de todos esses anos, colocando pessoas em minha vida e proporcionando situações que colaboraram com o meu sucesso e crescimento.

Aos meus pais Sebastião e Edileuza que nunca me deixaram faltar nada, que me criaram e me educaram ensinando o que é certo e que mesmo de longe, sempre cuidaram de mim e me deram todo apoio e incentivo que me motivaram a seguir em frente.

A minha tia Erilene por ter me acolhido em sua residência tornando possível a minha mudança para mais perto da Universidade.

Ao meu grande amigo de infância Deyvisson Dantas por todos os conselhos, apoio, incentivo e confiança durante todos esses anos.

Ao meu grande amigo e colega de estudos por longos anos Jean Carlos, por toda motivação e inspiração desde o ensino fundamental até o nível superior.

Aos meus amigos e colegas de curso Alberto Silva e Manoel Felipe, por toda ajuda, apoio, parceria e pelos bons momentos e experiências que nos foram proporcionados no decorrer do curso.

Ao meu professor e orientador Anderson Abner de Santana Souza que me ajudou no desenvolvimento deste trabalho, contribuindo com seu conhecimento e com o seu incentivo para que fosse possível a conclusão do mesmo.

A todos os meus professores da UERN que durante esses anos transferiram a mim parte de seus conhecimentos e experiências as quais vou levar pro resto da vida. Em especial aos professores Alberto Signoretti, Bruno Cruz, Camila Araújo, Carlos André, Isaac Filho e Karla Darlene, por todo o conhecimento e inspiração que me passaram.

A toda a minha família, amigos, colegas e todos os que de alguma maneira me apoiaram, incentivaram ou confiaram no meu sucesso.

A toda a UERN por ter proporcionado minha formação acadêmica e a oportunidade de adquirir conhecimento nessa área que sempre me fascinou.

RESUMO

Atualmente, a automação do gerenciamento e controle de acesso de ambientes restritos é algo difícil de se lidar devido ao alto custo de equipamentos capazes de realizar este serviço, impossibilitando a muitas empresas e instituições sua aquisição. Sem o controle automatizado de acesso, os usuários precisam normalmente ter sua própria chave, ou a empresa precisa dispor de alguém para abrir o local controlado sempre que necessário e anotar informações referentes à identificação e horários de acesso desses usuários. Sendo assim, este trabalho expõe o desenvolvimento de uma alternativa para a solução deste problema, através da implementação de um sistema de gerenciamento e controle de acesso, utilizando o módulo de câmera do ESP32 e um aplicativo de gerenciamento. O sistema proposto conta com a liberação de acesso através de reconhecimento facial que por sua vez dispara um fecho eletromagnético da porta, permitindo a abertura do local aos usuários previamente cadastrados e com acesso permitido. O sistema de controle de acesso proposto é gerenciado por meio de um aplicativo em um smartphone que se comunica com um servidor onde estão implementadas as funcionalidades de cadastrar, descadastrar e gerar relatórios. Os dados contendo as informações dos acessos realizados são armazenados localmente em um banco de dados e também enviados para a nuvem. Essas informações podem ser visualizadas no aplicativo mencionado, que as disponibiliza para o administrador do sistema.

Palavras chave: Automação. Gerenciamento de acesso. Controle de acesso. ESP. Reconhecimento facial.

ABSTRACT

Nowadays, the automation of management and access control of restricted environments are difficult to deal with due to the high cost of equipment capable of performing this service, making it impracticable for many companies and institutions to adopt it. Without automated access control, users typically have to have their own key, or the company needs to have someone to open the controlled location as needed and collect information regarding their identification and access time. In this context, this work introduces the development of an alternative to solve this problem by implementing a management and access control system using the ESP32 camera module and a management application. The proposed system allows access through facial recognition which in turn triggers the electromagnetic door closing, allowing the opening of the site, to previously registered users and with allowed access. The proposed access control system is managed through an application on a smartphone that communicates with the server where the functionalities to register, unsubscribe and generate reports are implemented. The data containing the information about the accesses allowed are stored locally in a database and also sent to the cloud. This information can be viewed in the mentioned application, which shows the information to the system administrator.

Keywords: Automation. Access Management. Access Control. ESP. Face Recognition.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Interface de programação de aplicações*
BSON	*Binary Structured Object Notation*
CLI	*Command Line Interface*
CPU	*Central Processing Unit*
FTDI	*Future Technology Devices International*
GPIO	*General Purpose Input/Output*
IDE	*Integrated Development Environment*
IOT	*Internet of Things*
JSON	*Javascript Object Notation*
LAR	*Laboratório de Aprendizagem Robótica*
LCD	*Liquid Crystal Display*
MVC	*Model View Controller*
NFC	*Near Field Communication*
REST	*Representacional State Transfer*
RFID	*Identificação por radiofrequência ou RFID*
SRAM	*Static Random Access Memory*
UERN	*Universidade do Estado do Rio Grande do Norte*
USB	*Universal Serial Bus*
VG-RAM	*The Virtual Generalizing Random Access Memory*
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A tecnologia se faz cada vez mais presente na vida das pessoas, mudando de vez a maneira como vivem, fazem suas atividades diárias e planejam seus momentos de lazer. A automação residencial e predial é um exemplo de inserção da tecnologia na vida diária. Elas trazem consigo o desenvolvimento e implantação de sistemas capazes de executar vários tipos de tarefas domésticas, que antes só eram feitas manualmente como, gerenciamento do acendimento de lâmpadas de forma inteligente e controle de eletrodomésticos por meio de smartphones. De forma geral, a automação residencial busca prover situações que proporcionem maior comodidade e segurança para as pessoas.

No que se refere a sistemas de automação para controle de acesso à ambientes restritos, há alguns sistemas disponíveis no mercado, que variam de sistemas de alarme a fechaduras eletrônicas. Porém, o alto preço dificulta a aquisição desses sistemas por usuários que não dispõem de muitos recursos financeiros.

Aliada com a Internet, a automação surge em um novo conceito que tem ganhado destaque, a Internet das Coisas, do inglês *Internet of Things* (IoT), que é composta por uma rede de "coisas" interligadas, que são capazes de coletar dados do ambiente nos quais estão, processar estes dados e atuar a partir da informação obtida depois do processamento (LA CRUZ & LA CRUZ, 2019). A Internet das Coisas não é utilizada apenas para ligar lâmpadas pelo celular, ou ativar os dispositivos pela Internet, mas busca, também, torná-los inteligentes, capazes de coletar e processar informações do ambiente ou das redes nas quais estão conectados (OLIVEIRA, 2017).

Este trabalho apresenta o desenvolvimento de um sistema de gerenciamento e controle de acesso a ambientes restritos, através de uma fechadura automática que limita o acesso a um ambiente apenas para pessoas autorizadas. O acesso é feito via autenticação por reconhecimento facial, que verifica se um usuário está habilitado ou não a entrar no ambiente restrito. Os usuários habilitados são pessoas que foram previamente cadastradas por um administrador do sistema, o qual se utiliza de um aplicativo para armazenar e gerenciar as informações de cadastro em um banco de dados localmente

acessível com cópia na nuvem (Internet). Além disso, com o aplicativo é possível ter acesso à relatórios gerados com os registros de entradas liberadas para o ambiente.

O foco do projeto é desenvolver um sistema que atenda às necessidades de limitar e gerenciar o acesso a ambientes, com um custo mais baixo do que o custo das soluções disponíveis no mercado. Nesse sentido as tecnologias escolhidas para o desenvolvimento do sistema foram: o React Native (biblioteca gratuita e *Open Source*) para o desenvolvimento do aplicativo que administra o sistema; a plataforma microcontrolada ESP que é usada para a construção do hardware; a biblioteca *face recognition* (*Open Source*) que é empregada na implementação da autenticação por reconhecimento facial; MongoDB (*Open Source*) para o banco de dados local; Firebase (ferramenta gratuita para pequenas implementações) que implementa o banco de dados na nuvem; e o Node.js (*Open Source*) para implementar as funcionalidades do servidor.

1.1 MOTIVAÇÃO E JUSTIFICATIVA

Para a realização da tarefa de controle e gerenciamento do acesso de ambientes restritos sem o auxílio da tecnologia, o método mais comum utiliza fechaduras manuais, onde só quem possui as chaves são os funcionários responsáveis pela segurança do local. Desta forma, qualquer pessoa que queira acessar o ambiente precisa solicitar a entrada para estes funcionários. Já o gerenciamento destes acessos é feito por meio do preenchimento de uma planilha contendo informações referentes à identidade do usuário e horários de entrada e saída do local.

Com a intenção de automatizar esse processo de controle e gerenciamento de acesso a ambientes restritos, este trabalho desenvolveu um sistema de controle, que utiliza como meio de autenticação o reconhecimento facial, eliminando assim, a necessidade de o usuário portar qualquer tipo de chave ou token de acesso, gerando maior comodidade ou reduzindo risco de perda. Além disso, todo o gerenciamento do sistema fica por conta de um aplicativo para smartphone que se comunica com o sistema físico e obtém os dados referentes à identidade e horários de entrada de cada usuário no local.

1.2 OBJETIVO GERAL DA MONOGRAFIA E METODOLOGIA

Objetivo deste documento é descrever o desenvolvimento de um sistema de baixo custo que limita o acesso à ambientes restritos, às pessoas autorizadas, utilizando reconhecimento facial e fazendo o gerenciamento das informações de acesso através de um aplicativo. Para se alcançar esse objetivo foram cumpridas algumas etapas de investigação, projeto e implementação.

A primeira etapa deste trabalho foi destinada a pesquisar, separar e estudar materiais e projetos relacionados ao contexto geral do sistema proposto. Assim, foram investigados, artigos, livros e outros meios com conteúdo relacionado à automação de tarefas através da tecnologia utilizada.

A segunda etapa teve como meta estudar e entender o funcionamento de maneira individual de cada componente que compõe o sistema, para que fosse possível fazer a integração destes componentes da melhor maneira possível. Para isto, foi feita uma busca e análise de materiais e trabalhos relacionados que utilizam estes componentes de maneira isolada ou em conjunto com algum outro. Além disso, foram definidas e configuradas as ferramentas utilizadas para o desenvolvimento do sistema.

Em seguida, foi feito o desenvolvimento do *backend* da aplicação, um servidor local implementado com Node.js, que se comunica tanto com o hardware de onde recebe os dados e faz o devido processamento enviando uma resposta de volta, quanto com o aplicativo de onde receberá dados para o cadastro dos usuários. Também nesta etapa foi configurada e estudado o funcionamento da biblioteca *face recognition*, utilizada para o reconhecimento facial.

Na etapa seguinte, foi realizado o desenvolvimento do aplicativo para smartphone, que implementa algumas funcionalidades necessárias para a administração do sistema, são elas: cadastrar um novo usuário, remover cadastros de usuários e mostrar relatórios contendo informações referentes aos acessos efetuados. O desenvolvimento foi feito utilizando a biblioteca React Native.

Posteriormente, se deu o desenvolvimento do sistema físico, que começou com a aquisição dos componentes e recursos necessários para a implementação do protótipo inicial. Boa parte dos componentes necessários estavam disponíveis para uso no

Laboratório de Aprendizagem Robótica (LAR) do campus de Natal da Universidade do Estado do Rio Grande do Norte (UERN), onde também dispõe do espaço que foi usado para montagem e testes. Ainda nesta etapa, foi vista e definida a comunicação do servidor com os módulos de hardware e aplicativo, por meio de uma rede Wifi local.

Por fim, foram realizados testes do funcionamento do sistema, avaliando cada uma de suas partes (hardware, servidor e aplicativo) separadamente. Posteriormente, foi verificada e a integração desses componentes, executando o processo de autenticação de entrada, com a apresentação de faces cadastradas no sistema e verificação do seu grau de acertos, até a obtenção de um nível satisfatório de acertos e de rapidez na resposta do sistema, que permita sua validação.

1.3 ESTRUTURA DO DOCUMENTO

No próximo capítulo serão analisados alguns trabalhos relacionados ao tema, destacando as semelhanças com o aqui desenvolvido e apresentando pontos positivos e negativos de cada um.

No terceiro capítulo serão descritos em detalhes todos os conceitos e tecnologias que serviram como base para este projeto.

No quarto capítulo será explicado detalhadamente o funcionamento do Sistema como um todo, e será detalhado como cada parte funciona separadamente.

Em seguida serão apresentados todos os resultados obtidos ao final do projeto, os mesmos foram obtidos através de testes realizados no LAR.

Por fim, no Capítulo 6 são colocadas as considerações finais, como também as expectativas de trabalhos futuros para a evolução deste projeto.

2 TRABALHOS RELACIONADOS

Este capítulo tem como objetivo apresentar trabalhos de propósito semelhante ao aqui desenvolvido, explicar o funcionamento dos mesmos, apontar suas diferenças e destacar os pontos positivos e negativos de cada um.

No trabalho de Peixoto (2013) é desenvolvido um sistema de controle de acesso a salas de aula, tendo como objetivo automatizar esse processo, que até então, vinha sendo feito manualmente através do uso de claviculário manual e porteiro eletrônico. O projeto é dividido em duas partes. A primeira trata-se de um middleware que foi construído utilizando a plataforma *mbed* e é responsável por receber os dados do usuário que acontece através da leitura de tags de identificação por radiofrequência (RFID). A segunda parte é um servidor que recebe os dados enviados pelo middleware via rede ethernet e envia uma resposta de volta, autorizando ou não o acesso.

O sistema conta com um meio de autenticação simples e prático, que apresenta bons tempos de resposta, porém a utilização de tags de RFID possibilita a entrada de pessoas não autorizadas caso estejam em posse (de maneira indevida) de tags de outros usuários autorizados a entrar no ambiente. Além disso, as tags podem ser perdidas, dificultando o gerenciamento dos acessos. Ademais, o sistema faz o armazenamento dos dados de entrada em um banco de dados, mas ainda não dispõe de uma interface de visualização e gerenciamento dos mesmos.

Outro trabalho semelhante a esse é o proposto por Silva (2014), onde é apresentado um sistema de controle de acesso ao Laboratório de Aprendizagem Robótica da UERN. O sistema foi desenvolvido utilizando a plataforma Arduino e fazendo o processo de autenticação através da leitura de tags RFID. O mesmo apresenta a vantagem de fazer o processo de autenticação em poucos segundos e sem a necessidade de um servidor externo, visto que o microcontrolador utilizado é capaz de fazer o processamento necessário.

Algumas limitações são encontradas no que se refere ao gerenciamento dos dados registrados, uma vez que todo registro de acessos fica em um cartão de memória que fica conectado ao sistema. O cadastro de novas tags precisa ser feito manualmente inserindo o ID das mesmas em um arquivo de texto que precisa estar presente no cartão

de memória do sistema. Há também o risco de alguém não autorizado conseguir o acesso através da posse indevida de uma tag de outro usuário autorizado.

O trabalho proposto por Marques et. al. (2017) apresenta um sistema de gerenciamento de acesso por biometria digital, o mesmo foi construindo utilizando a plataforma Arduino e um leitor de impressões digitais. O sistema possui um meio de autenticação bastante seguro e obteve uma boa precisão no reconhecimento dos usuários, também teve um custo relativamente baixo se comparado com os equipamentos de mesmo propósito existentes no mercado.

O trabalho possui algumas desvantagens e limitações, uma delas é que todos os dados de cadastro e registro de acesso são armazenados na placa Arduino, que possui um armazenamento limitado a 256 KB. Outra limitação é que o sistema não dispõe de meios de gerenciamento dos dados de acesso e toda a administração do sistema, no que se refere ao cadastro, remoção de usuários é feita através de um menu exibido em um display lcd e controlado por botões no próprio dispositivo.

O trabalho proposto por Moraes (2010) apresenta um software de controle de acesso por biometria facial, o mesmo foi desenvolvido com a linguagem de programação C# e para fazer o reconhecimento facial utilizando uma rede neural já existente chamada VG-RAM. A execução do mesmo acontece em um notebook e utiliza a webcam do próprio dispositivo para a captura das imagens, o processo acontece da seguinte forma: a câmera fica continuamente buscando por uma mão em determinada posição (previamente definida) e ao encontrar o padrão buscado, passa a procurar por rostos para realizar o processo de reconhecimento, liberando ou não o acesso.

O sistema apresentou resultados muito bons, com uma alta taxa de acertos, porém sua execução precisa ser feita a partir de um computador, o que impossibilita ou dificulta o acoplamento junto a porta e aumenta consideravelmente o custo final do equipamento.

3 FUNDAMENTAÇÃO TEÓRICA

Há diversas tecnologias para controlar algo de forma centralizada, remota ou mesmo de forma autônoma. O avanço dessas tecnologias ao longo dos anos vem proporcionando um aumento na acessibilidade e comodidade na vida das pessoas. Contudo, é importante constatar que para isso é preciso ter equipamentos desenvolvidos com a finalidade de exercer esse controle e que proporcionem conectividade e intercomunicação (STEVAN JUNIOR & FARINELLI, 2019).

No âmbito da automação, não se pode deixar de falar de Internet das Coisas. Esse termo vem ganhando cada vez mais destaque ao longo dos anos. Seu conceito é bem amplo e consiste na interação entre quaisquer dispositivos que possam se comunicar via redes com e sem fio, coletar dados contextuais do ambiente monitorado, criar uma rede de informações e fornecer novas funcionalidades e possibilidades de atuação (CHAUDHURI, 2018).

Dentre as aplicações que podem ser implementadas com a Internet das Coisas, algumas estão relacionadas à automação residencial. Estas variam desde o gerenciamento da iluminação da casa até o controle de eletrodomésticos pelo smartphone.

Outra área relacionada a essa tecnologia é a segurança, seja das pessoas que vivem em um ambiente, seja do próprio ambiente. Neste último caso podem ser empregados equipamentos como alarmes, cerca elétrica ou fechaduras eletrônicas (STEVAN JUNIOR, 2018).

Fechaduras eletrônicas podem ser desenvolvidas com o auxílio de sensores e um microcontrolador, para que além da limitação do acesso, consigam fazer o gerenciamento dos acessos ao ambiente.

A utilização de sensores como leitor de RFID, ou leitor biométrico para fazer o controle e gerenciamento de acesso a ambientes restritos é uma aplicação típica da Internet das Coisas. Dessa forma, são automatizados os processos de controlar o acesso e de registrar informações dos acessos ocorridos a estes ambientes (OLIVEIRA, 2017).

3.1 BIOMETRIA FACIAL

Para que se tenha um nível ainda mais alto de segurança em ambientes de acesso restrito, se faz necessário o uso de meios de autenticação mais avançados. Entre os métodos mais atuais, o que se mostra mais eficaz é a biometria, que usa características físicas para fazer o processo de reconhecimento e autenticação de pessoas. Dessa forma, em vez do uso de crachás, tokens ou chaves, a autenticação é feita através da impressão digital, voz, reconhecimento facial, entre outras (SANTOS A. L., 2007).

Para este projeto a biometria facial foi escolhida, por: oferecer comodidade e facilitar o processo de autenticação ao dispensar o uso de meios físicos; apresentar bons resultados em relação a taxa de acertos no reconhecimento facial; meio bastante seguro devido a unicidade da face humana; possibilidade da sua implementação poder ser feita com uma simples câmera e um algoritmo de visão computacional.

3.2 MICROCONTROLADORES

Para que as aplicações que usam o conceito de Internet das Coisas sejam implementadas, é preciso se atentar a alguns fatores necessários para que aconteça a troca de informações e execução de ações entre os dispositivos conectados. Segundo Hostgator (2018) esses fatores são:

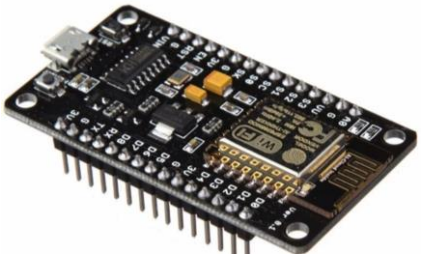
- Os dispositivos, que são todas as coisas que são o sujeito principal da automação, por exemplo, geladeiras, aparelhos de Tv, cafeteiras etc.
- A rede, que é o meio de comunicação entre os dispositivos e o microcontrolador, que pode ser Wi-Fi, Bluetooth, dados móveis, entre outras.
- O sistema de controle, que é o responsável por receber os dados coletados, processar e executar alguma ação sobre o ambiente ou dispositivo.

Para escolher um microcontrolador para a aplicação que se propõe, é preciso levar em conta aspectos como: funcionalidades exigidas, sensores e/ou atuadores suportados e também o seu custo. Muitas aplicações têm sido desenvolvidas utilizando os microcontroladores da família ESP que se destacam por terem um baixo custo e dispor de componentes como comunicação Wi-fi e Bluetooth integrados.

3.2.1 Plataforma NodeMCU

O NodeMCU é uma plataforma de código e hardware aberto que é amplamente utilizada para aplicações de Internet das Coisas. Tem como base um microcontrolador chamado ESP8266 da fabricante Espressif Systems, que dispõe de um módulo de comunicação via Wifi. A plataforma conta com pinos de *General Purpose Input/Output* (GPIO), comunicação USB-serial e pinos de alimentação de 3,3 V (STEVAN JUNIOR, 2018). Suas principais vantagens em relação aos seus similares são o seu baixo custo, tamanho reduzido e o módulo de Wi-fi integrado. A plataforma possui uma SDK própria para o desenvolvimento do código utilizando linguagem LUA, suporta também programação em linguagem C/C++ através da Arduino IDE (*Integrated Development Environment*) (BERTOLETI, 2019). A Figura 1 mostra a NodeMCU.

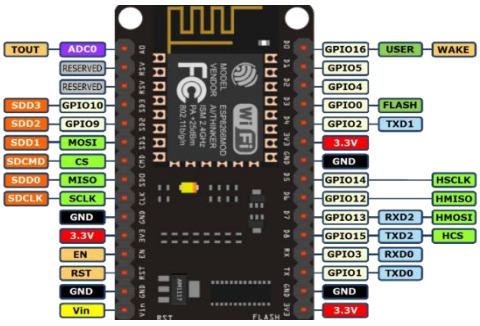
Figura 1: Plataforma NodeMCU



Fonte: Espressif Systems

Sua pinagem é disposta em duas fileiras com 15 pinos cada, e pode ser vista na Figura 2, além dos pinos de alimentação, a placa conta com vários pinos de entrada e saída de dados, a Tabela 1 apresenta e detalha alguns dos principais deles.

Figura 2: Pinagem da plataforma NodeMCU



Fonte: Arduining

Tabela 1: Pinos da plataforma NodeMCU

Pino	Função	Função opcional
VIN	Pino de alimentação externa	-
GND	Pino terra da placa	-
RST	Reset do módulo	-
3.3V	Saída de 3.3 V para alimentar outro dispositivo	-
D0	Entrada e saída de dados	Pode ser usado para acordar o ESP8266 do modo Deep sleep
D1	Entrada e saída de dados	-
D2	Entrada e saída de dados	-
D3	Entrada e saída de dados	Pode ser usado para controlar o upload do programa na memória Flash.
D4	Entrada e saída de dados	UART_TXD1 quando carregando o programa na memória FLASH
D5	Entrada e saída de dados	pode ser usado em SPI de alta velocidade (HSPI-SCLK)
D6	Entrada e saída de dados	pode ser usado em SPI de alta velocidade (HSPI-MISO)
D7	Entrada e saída de dados	pode ser usado em SPI de alta velocidade (HSPI-MOSI) ou UART0_CTS.
D8	Entrada e saída de dados	pode ser usado em SPI de alta velocidade (HSPI-CS) ou UART0_RTS.
TX	Pino serial	-
RX	Pino serial	-

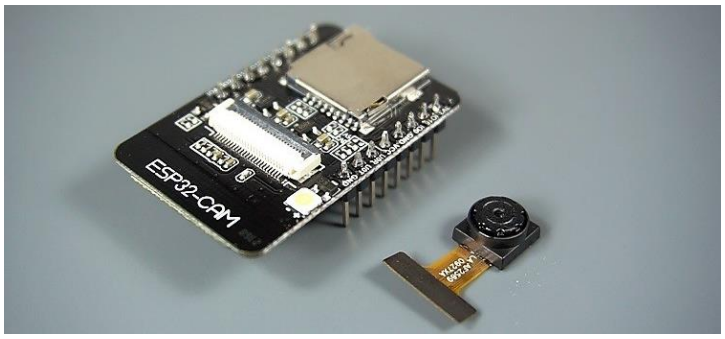
Fonte: Brain Eletrogate

No desenvolvimento deste projeto, foram utilizados apenas 4 pinos: D1 e D2 para a conexão com o display LCD, D5 para fazer a leitura do estado do botão que faz o acionamento do processo de autenticação e D7 para a conexão com o módulo relé, a alimentação da placa foi feita via usb.

3.2.2 ESP32-CAM

Também da família ESP, o ESP32-CAM (Figura 3) é uma placa específica para aplicações que necessitam de uma câmera. Ela suporta dois modelos de câmera, OV2640 e OV7670, além disso vem com um módulo Wi-fi integrado, slot para cartão de memória, CPU de 32 bits de baixa potência e SRAM de 520 KB embutida.

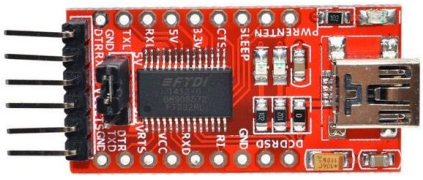
Figura 3: ESP32-CAM



Fonte: Random Nerd Tutorials

Como esta placa não possui entrada USB, o carregamento do código precisa ser feito através dos pinos seriais, para isso pode ser utilizado um conversor usb-serial, como por exemplo o módulo FTDI (*Future Technology Devices International*) mostrado na Figura 4.

Figura 4: Módulo FTDI



Fonte: Auto core robótica

A placa ESP32-CAM possui pinos para entrada e saída de dados, alguns deles estão conectados internamente no slot para sdcard. Também conta com pinos de alimentação de 3.3 V e 5V além do GND. Sua pinagem completa pode ser vista na Figura 5.

Figura 5: Pinagem da ESP32-CAM

Fonte: Seeed Studio

Semelhante as outras placas da família ESP, a ESP32-CAM também pode ser programada em C/C++, utilizando a Arduino IDE, porém é necessário que suas configurações sejam previamente instaladas na IDE.

3.2.2.1 Câmera OV2640

Para este projeto, foi utilizada a câmera OV2640, ela possui 2 mega pixels de resolução, produz cores verdadeiras de 24 bits, é capaz de produzir vários tamanhos de imagem, incluindo 320 x 240, 640 x 480, 1600 x 1200, e também vários formatos de imagem incluindo o JPEG (ZHENG, 2014).

A escolha dela foi motivada pelo fato do seu funcionamento atender as necessidades do sistema desenvolvido, capturando imagens em qualidade suficiente para que o processo de reconhecimento funcione.

3.3 NODE.JS

O Node.js é um interpretador JavaScript baseado no motor V8 desenvolvido pela Google para ser usado no seu navegador, o Google Chrome. Com a chegada do Node.js, o JavaScript passou a ser executado também no lado do servidor. O Node.js usa um loop de eventos e executa tudo em uma única *thread*, onde todas as requisições são assíncronas e não permitem bloqueios.

O loop de eventos, que é ilustrado pela Figura 6, é basicamente um loop infinito que está sempre verificando se chegou alguma requisição. Ao receber requisições que exijam algum tempo de resposta, como uma leitura no banco de dados ou uma consulta a uma Interface de programação de aplicações (API) externa, a ação é executada sem ficar detida, esperando a resposta, passando assim, para a ação seguinte. Ao receber uma resposta da ação anterior, a função de *callback*, que foi definida como resposta da ação é executada assim que possível (SANTOS G. , 2016).

Figura 6: Event-Loop do Node.js

Fonte: Pereira (2016)

Segundo Künzel (2018), o uso do Node.js possui algumas vantagens e desvantagens.

Vantagens:

- O uso do JavaScript como linguagem, visto que JavaScript é uma das linguagens mais utilizadas e conta com uma das maiores comunidades;
- A praticidade em utilizar a mesma linguagem tanto no *Frontend* quanto no *Backend*;
- Tempo de resposta rápido.

Desvantagens:

- O loop de eventos torna o tratamento de erros mais difícil.
- Relativamente novo no mercado se comparado a tecnologias concorrentes que já estão mais consolidadas.

3.3.1 Bibliotecas

Para a execução de algumas tarefas dentro do Node.js se faz necessário o uso de algumas bibliotecas de terceiros. A Tabela 3 lista algumas bibliotecas que foram utilizadas no desenvolvimento deste projeto.

Tabela 2: Bibliotecas utilizadas no Node JS

FS	Utilizado para ler, escrever, atualizar, deletar e criar arquivos.
Sharp	Usado para a manipulação de imagens fazendo conversões entre formatos de imagem ou fazendo redimensionamentos.
Multer	Utilizado para o upload de arquivos pelo Express.
Mongoose	Utilizado para lidar com a modelação do banco de dados não relacional Mongo DB baseando-se no modelo de esquemas.
Child Process	Utilizado para executar processos ou comandos nativos do sistema operacional.
Nodemon	Automatiza o processo de rodar o servidor novamente a cada atualização no código JavaScript

Fonte: própria

3.4 RECONHECIMENTO FACIAL

O reconhecimento facial é algo natural para os seres humanos desde a mais tenra idade. Alguns experimentos expostos no trabalho de Turati et al. (2006) mostram que bebês com três dias de vida já estão aptos a fazer distinção entre faces conhecidas. Porém, até o momento não se sabe exatamente quais as informações que nosso cérebro utiliza e como essas informações são tratadas para fazer o reconhecimento de padrões (como faces). Supõe-se que algumas características específicas locais (olhos, nariz, boca, forma da cabeça, entre outras) são detectadas e processadas pelo córtex cerebral para se diferenciar faces (ZHAO, CHELLAPPA, PHILLIPS, & ROSENFELD, 2003).

Para a execução deste processo os sistemas computacionais para reconhecimento facial automático se baseiam no princípio de extração de características (pontos notáveis) de uma imagem e na formatação delas em uma representação útil (OPENCV, 2019). Esses sistemas seguem um *pipeline* geral para solucionar o reconhecimento facial que segue os passos mostrados no diagrama da Figura 7, adaptada do trabalho de Zhao (2003). Cada etapa recebe o resultado da etapa anterior para proceder com suas operações.

Neste trabalho, foi adotada a biblioteca *Open Source face recognition*[1] para realizar as etapas de reconhecimento facial. A mesma se baseia na biblioteca de Visão Computacional OpenCV e outras ferramentas de Processamento de Imagens e Inteligência Artificial, como as bibliotecas, OpenFace (2019) e Dlib (2019), para a utilização de algoritmos de aprendizagem de máquina.

A seguir, será feita uma explicação das etapas expostas na Figura 7, onde ao lado de cada uma delas são apresentadas outras aplicações e abordagens onde podem ser empregadas.

[1] GEITGEY, **Face Recognition**, 2018. Disponível em: <https://github.com/ageitgey/face_recognition> Acesso em: 16 set. 2019.

Figura 7: Etapas do processo de reconhecimento facial

Fonte: Zhao (2003)

- Detecção de Face

O objetivo desta etapa é detectar faces em uma dada imagem, a fim de separá-las das demais regiões da imagem e, posteriormente, fazer o processamento adequado para extração de informações a serem utilizadas em uma tentativa de reconhecimento. Como resultado desta etapa, o algoritmo deve fornecer regiões que definem os limites das faces encontradas. A Figura 8 ilustra o resultado desta etapa.

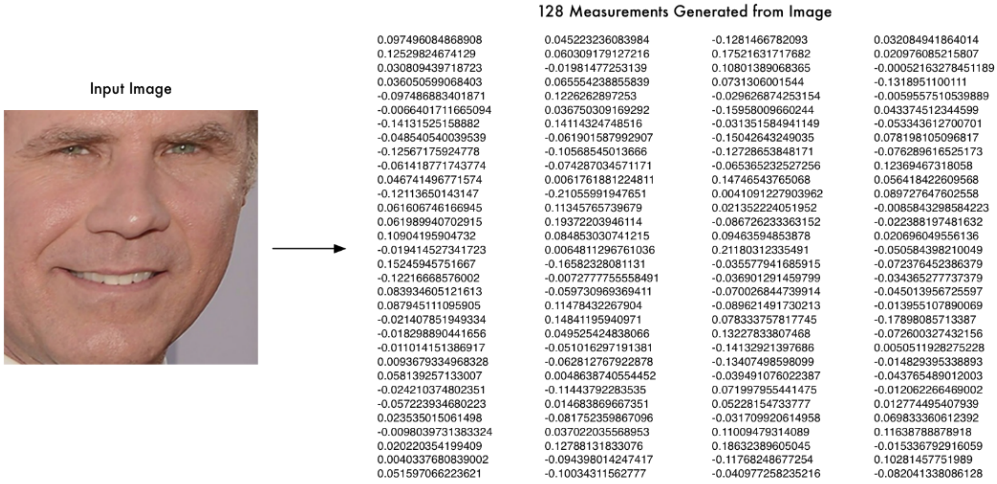
Figura 8: Delimitações da face encontrada

Fonte: Geitgey, 2018

- Extração de características

Neste ponto, tem-se as faces identificadas e isoladas. O próximo passo é a identificação ou extração de características nas faces. Essas características são medidas necessárias para se fazer as comparações entre a face analisada e faces cadastradas, já reconhecidas. A Figura 9 mostra a extração das características de uma face analisada.

Figura 9: Medidas extraídas da imagem



Fonte: Geitgey, 2018

- Reconhecimento de face

Esta etapa faz a comparação entre as características encontradas na face analisada com as características das faces armazenadas em um banco de dados. Quando as medidas de ambas as faces resultam em uma diferença considerada limiar, as faces são tidas como coincidentes, o que resulta no reconhecimento da face analisada. A biblioteca *face recognition* faz a marcação das faces reconhecidas, dando aos desenvolvedores a possibilidade de rotulá-las, como mostra a Figura 10.

Figura 10: Marcação das faces conhecidas

Fonte: Geitgey, 2018

3.5 MONGO DB

O MongoDB é um banco de dados *Open Source* orientado a documentos. Foi escrito em C++, o que o torna compatível com diferentes sistemas operacionais. É utilizado por grandes e pequenas empresas e apresenta bons resultados no que se refere a baixa latência, alta vazão e disponibilidade. Diferente dos bancos relacionais que armazenam os dados em tabelas, no MongoDB os dados são armazenados em uma estrutura BSON, que é algo semelhante a notação de objetos do Javascript (JSON) que é bastante utilizada na web. Os documentos possuem dados simples dos tipos *String*, *Number* ou *Boolean*, e juntos formam coleções, um conjunto dessas coleções formam o banco de dados (DUARTE, 2017).

3.6 FIREBASE

O Firebase é uma plataforma da Google que fornece uma gama de serviços para o desenvolvimento e gerenciamento de aplicações mobile ou web. Para usufruir desses serviços basta apenas ter uma conta no Google e criar um projeto dentro do console da plataforma, a mesma oferece alguns planos que vão desde o grátis até o pagamento de acordo com o consumo (FIREBASE, 2019).

Alguns dos serviços oferecidos pelo Firebase:

- **Firebase Authentication:** Permite a implementação de mecanismos de autenticação na aplicação, os usuários podem fazer o cadastro e login com e-mail e senha ou também com alguma rede social.
- **Firebase Hosting:** é um recurso que permite a hospedagem de aplicações web estáticas ou dinâmicas em nível de produção através de comandos que podem ser executados em apenas alguns segundos a partir de uma interface de linha de comando (CLI).
- **Firebase Realtime Database:** é um banco de dados NoSQL que fica hospedado na nuvem. Seus dados são armazenados em arquivos no formato JSON e são sincronizados com os usuários em tempo real. Todos os apps conectados a ele recebem atualizações de conteúdo sempre que há uma mudança ou entrada de novos dados.

3.7 INTERNET DAS COISAS NOS SMARTPHONES

Além dos sensores e atuadores que compõe as aplicações de IoT, é muito comum a interface usada para controlar e obter informações referentes ao sistema ser executada em smartphones. A evolução e a popularização dos aparelhos fizeram com que o seu uso se expandisse para cada vez mais funcionalidades. A integração dos smartphones com os sistemas IoT já acontece com bastante frequência, visto que os mesmos são acessíveis e dispõem de sensores que podem integrar facilmente com o hardware construído, como Bluetooth, Wi-Fi, NFC (*Near Field Communication*), giroscópio, câmera e etc. (OLIVEIRA, 2017).

Para que haja a integração de um sistema físico com os smartphones de forma que se possa controlar e ver informações do sistema, é preciso que seja desenvolvido um aplicativo específico para a aplicação. O desenvolvimento desse aplicativo pode ser feito através de diversas tecnologias, podendo gerar aplicativos nativos, onde a implementação é feita utilizando linguagens que são definidas pelos desenvolvedores do sistema operacional do smartphone; ou aplicativos híbridos, onde o desenvolvimento é feito em outra linguagem ou framework, que faz uma ponte com os componentes nativos do sistema operacional. Nesse trabalho foi desenvolvido um aplicativo utilizando o React Native que cria um aplicativo nativo, como será detalhado na seção a seguir (GOIS, 2017).

3.8 REACT NATIVE

O React Native é uma biblioteca JavaScript desenvolvida pelo Facebook para viabilizar o desenvolvimento de aplicações nativas tanto para sistemas Android quanto para sistemas iOS, com a mesma base de código, utilizando a sintaxe JSX, que é similar ao XML (*Extensible Markup Language*).

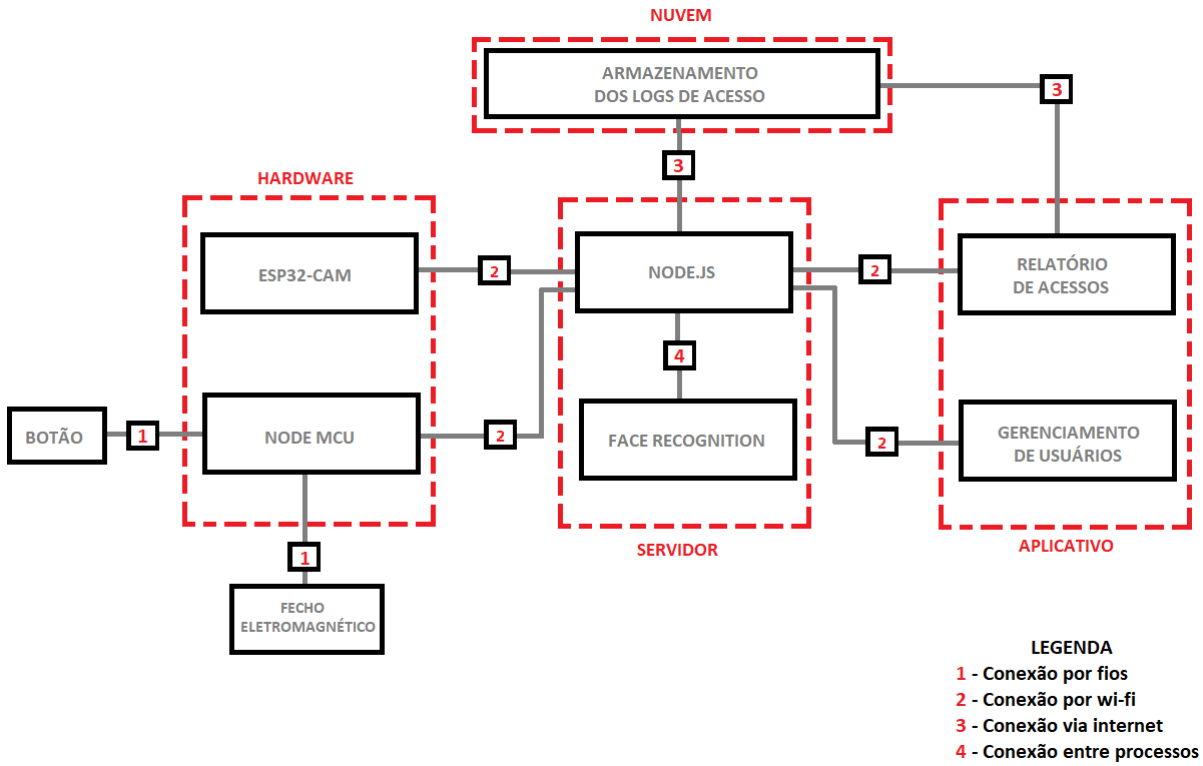
Com esta biblioteca é possível criar componentes no formato de tags que serão convertidos para componentes nativos do sistema em que for executado. Esta biblioteca é gratuita, *Open Source* e muito poderosa para a construção de aplicativos.

4 SISTEMA IMPLEMENTADO

O sistema desenvolvido e aqui explicado é dividido em três partes, são elas: o servidor que é responsável pelo processamento das imagens, armazenamento do registro de usuários e armazenamento do registro de acessos; o aplicativo que tem o papel de fazer o gerenciamento dos usuários, como também realizar o cadastro das imagens a serem utilizadas no processo de reconhecimento e visualização do registro de acessos; hardware que dispõe da câmera utilizada para a captura das imagens e o microcontrolador responsável por estabelecer a comunicação com o servidor. A alimentação do hardware é feita através de uma fonte de 5V.

A Figura 11 mostra todos os componentes do sistema e os meios de comunicação entre cada um deles.

Figura 11: Funcionamento geral



Fonte: própria

O hardware do sistema fica acoplado junto a porta, aguardando pela ação que inicia o processo de autenticação, o seu fluxo de funcionamento segue a seguinte sequência:

- O sistema possui um botão, que ao ser pressionado faz uma requisição ao servidor para que o mesmo faça a captura de uma imagem.
- Logo após a captura da imagem, o servidor inicia o processo de reconhecimento facial e envia uma resposta de volta ao hardware.
- Se o usuário estiver cadastrado no banco de dados, o fecho eletromagnético é disparado, abrindo a porta e registrando o acesso, senão, o acesso é negado.

Para o desenvolvimento de cada parte do sistema foram utilizadas tecnologias diferentes, a seguir, serão detalhados o desenvolvimento e o funcionamento de cada uma dessas partes.

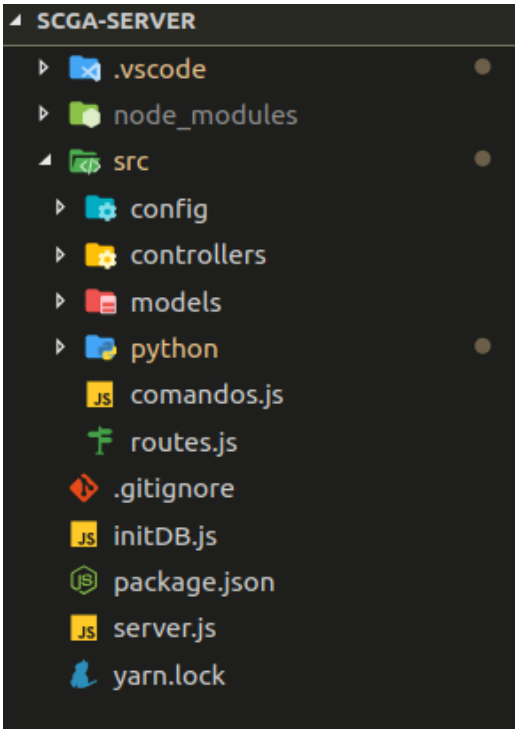
4.1 FUNCIONAMENTO DO SERVIDOR

Como o hardware utilizado neste projeto não tem poder de processamento suficiente para fazer o processamento das imagens, foi necessário o uso de um servidor para executar os algoritmos de processamento de imagem. Este servidor foi implementado utilizando Node.js e está rodando em uma rede local.

4.1.1 Estrutura de pastas

Para uma melhor organização dos arquivos foi adotada uma adaptação do modelo de padrão de projeto de software chamado MVC (*Model View Controller*), sendo que nesse caso não possui as *Views*, pois toda a parte visual é feita através do aplicativo móvel. Permanece então, os *Models* e os *Controllers*. A Figura 12 mostra a estrutura de pastas do projeto.

Figura 12: Estrutura de pastas



Fonte: própria

Algumas outras pastas e arquivos precisaram ser incluídos no decorrer do projeto, os mesmos serão citados logo a seguir:

- **. vscode:** Contém arquivos de configuração do editor do código
- **node_modules:** Contém todos os arquivos das dependências que foram instaladas no projeto
- **config:** Contém arquivos de configurações básicas do sistema
- **controllers:** Contém arquivos com todas as funções que serão chamadas durante a execução do sistema
- **models:** Contém os *schemas* de todos os objetos que precisam ser guardados no banco de dados
- **python:** contém os arquivos necessários para fazer a manipulação e execução dos algoritmos de reconhecimento facial

- **initDB.js:** arquivo que é executado uma única vez na primeira execução do servidor, serve para inicializar o banco de dados com algumas configurações padrão.
- **package.json:** arquivo responsável por armazenar uma lista com todas as dependências instaladas no projeto, também armazena scripts definidos pelo desenvolvedor.
- **comandos.js:** arquivo que possui um objeto contendo todos os comandos do shell que precisam ser executados no decorrer do projeto.
- **routes.js:** arquivo contendo todas as rotas definidas para a API
- **server.js:** arquivo principal do projeto, de onde são feitas as configurações iniciais do mesmo para só então, ser iniciado.

4.1.2 API REST

Do lado do servidor foi implementada uma *Application Programming Interface* (API) *Representational State Transfer* (REST), que é um estilo de arquitetura de software que impõe um conjunto de regras para a criação de web services. A mesma possui algumas rotas configuradas para cada etapa do processo, onde cada uma dessas rotas ao serem requisitadas executam alguma ação e enviam uma mensagem de volta para quem a requisitou.

Para cada função executada tanto pelo hardware quanto pelo aplicativo, que se comunicam com o servidor, foi definida uma rota, que é composta pelo IP e porta da máquina em que o servidor está sendo executado, seguido de uma terminação que indica para que cada uma delas serve. Ao receber uma requisição em uma dessas rotas, o servidor executa uma ação e envia uma resposta de volta a quem a requisitou. A Tabela 5 lista todas as terminações das rotas definidas e suas respectivas funções.

Tabela 3: Rotas configuradas

Rota	Função
/users	Retorna um objeto com a lista de todos os usuários cadastrados
/addUser	Recebe um novo usuário e faz o registro no banco de dados
/editUser	Edita e salva informações pessoais de um usuário
/deleteUser	Deleta um usuário
/img	Captura uma imagem com o ESP32-CAM
/upimg	Faz uploads das imagens no cadastro de um novo usuário
/rec	Para iniciar o processo de reconhecimento facial
/clearFolder	Apaga as imagens da pasta temporária após extrair os dados necessários das mesmas.

Fonte: própria

4.1.3 Execução de comandos shell

A implementação do servidor foi feita com Node.js que, por sua vez, utiliza a linguagem JavaScript, já os algoritmos de manipulação de imagens e reconhecimento facial são implementados na linguagem Python. Dessa forma, se fez necessário o uso de funções do JavaScript que implementam comandos do shell, para que fosse possível efetuar os comandos necessários para a execução dos algoritmos.

4.1.4 Reconhecimento facial

O processo de reconhecimento facial é feito através da biblioteca *face recognition*. Seu funcionamento foi explicado no Capítulo 3, a mesma foi adaptada para que pudesse ler e salvar os dados dentro da estrutura de pastas do projeto e para considerar apenas a primeira face detectada caso apareçam mais de uma face na imagem.

A ferramenta utilizada para o reconhecimento facial vale-se de uma rede neural convolucional pré-treinada para conseguir identificar os pontos de referência das faces

encontradas na imagem. Isso equivale à etapa de extração de características explicada na seção 3.4 (OPENFACE, 2019).

Antes de passar pela etapa de extração das características, as imagens que são capturadas pela câmera do smartphone e enviadas ao servidor, são manipuladas a fim de reduzir o tamanho do arquivo e agilizar o processamento das mesmas. Para isso, foi utilizada uma biblioteca JavaScript chamada Sharp. A imagem é convertida para o formato jpeg e redimensionada de modo a ter a mesma resolução das imagens capturadas pelo módulo de câmera.

Visto que o processo de reconhecimento é feito em cima de imagens estáticas onde o usuário precisa estar olhando para a câmera no momento da captura, de forma a não haver variação na posição do rosto, para cada usuário cadastrado no sistema, são utilizadas 3 imagens da face. Em tese, uma imagem já seria o suficiente, porém, para aumentar a precisão do processo de reconhecimento, são usadas 3 imagens. Esse número foi escolhido depois de alguns testes, com os quais verificou-se que com essa quantidade de imagens teríamos uma boa taxa de acerto, sem comprometer o tempo de processamento na busca do reconhecimento. Quanto mais imagens cadastradas maior o tempo necessário para o processo de reconhecimento.

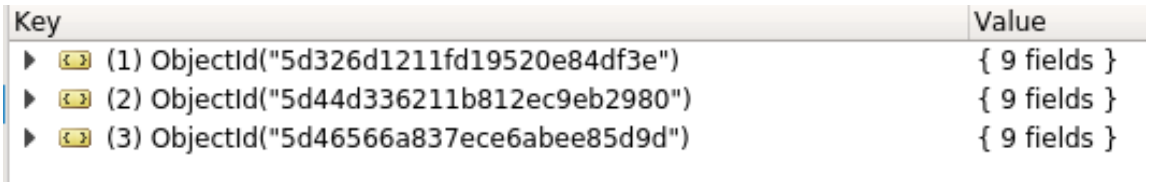
De cada uma das imagens que precisam ser cadastradas é extraído um vetor de 128 posições (operação executada pela rede neural) contendo as medições da face. Essas medições são usadas na etapa de reconhecimento, que consiste na comparação entre vetores das faces analisadas e vetores das faces cadastradas. Esses dados são armazenados em um arquivo do formato pickle, que é uma classe do Python que permite o armazenamento de estruturas de dados em um arquivo.

O arquivo contendo os dados da imagem é único e está localizado na pasta raiz do projeto. Quando uma nova imagem nova precisa ser cadastrada, a estrutura de dados contida no arquivo é montada, os dados da nova imagem são inseridos e os dados são salvos novamente no arquivo.

4.1.5 Armazenamento dos dados no banco

O armazenamento dos dados dos usuários foi feito usando o banco de dados não relacional MongoDB, que é formado por coleções, onde cada coleção é composta por um ou mais documentos. Esses documentos são identificados por um ID gerado automaticamente pelo banco como mostrado na Figura 13.

Figura 13: Coleção de usuários

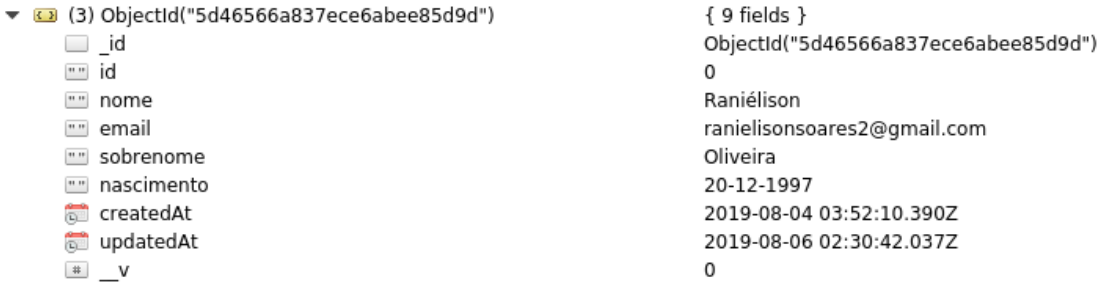


Fonte: própria

4.1.5.1 Registro de usuários

A entrada das informações acontece através do aplicativo, durante o processo de cadastro de novo usuário. Nessa etapa, é preenchido um formulário com algumas informações pessoais. Além disso, algumas informações são adicionadas automaticamente pelo próprio sistema, como ID e data de criação. A Figura 14 mostra a estrutura dos dados que são armazenados.

Figura 14: Dados do usuário



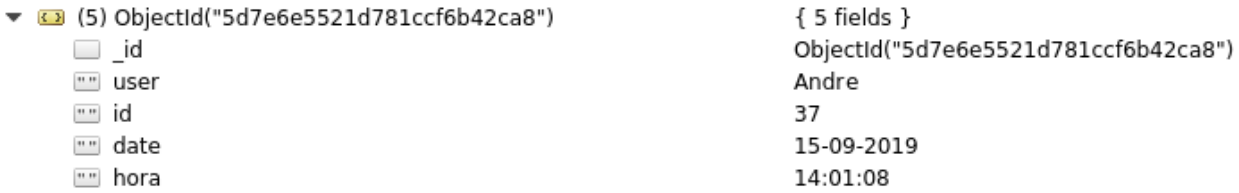
Fonte: própria

4.1.5.2 Registro de acessos

O registro dos acessos ao ambiente é feito em duas bases de dados. A primeira é similar ao armazenamento dos usuários no banco Mongo DB, localmente. A segunda é na nuvem, utilizando o banco de dados do Firebase, o Realtime Database.

No banco de dados local, os dados são armazenados na coleção acessos. Cada acesso é representado por um documento contendo as informações referentes ao acesso realizado ao ambiente, são elas: o nome do usuário, o id do usuário, data e horário do acesso, como mostrado na Figura 15.

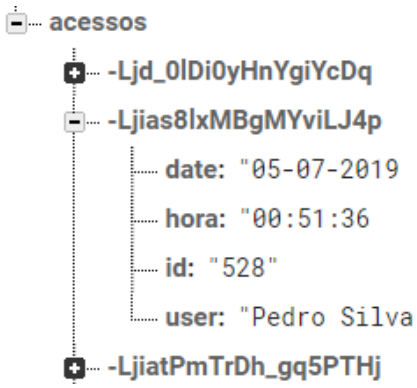
Figura 15: Registro de acesso Mongo DB



Fonte: própria

Quanto ao armazenamento na nuvem através do Firebase, os dados também são organizados no formato de coleções e documentos, como pode ser visto na Figura 16. Dentro da coleção acessos, tem vários documentos cada um representado por uma chave única gerada automaticamente pelo Firebase. Dentro de cada documento se tem os dados referente aos acessos.

Figura 16: Acessos no Firebase

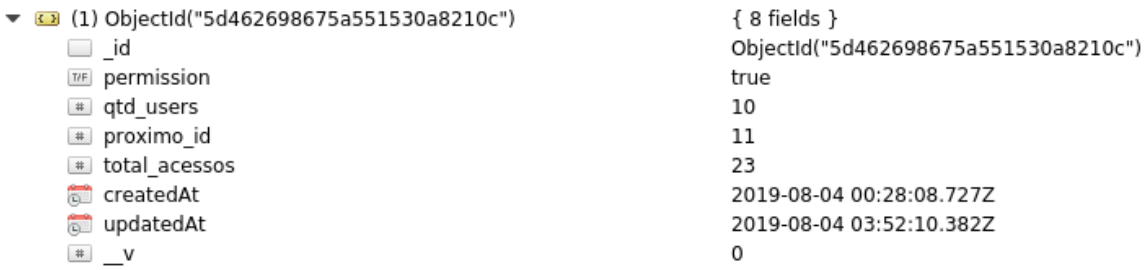


Fonte: própria

4.1.5.3 Informações adicionais

Além dos registros de acesso e dos dados dos usuários, algumas outras informações precisam ser guardadas no banco de dados, com a finalidade de armazenar alguns dados necessários durante a execução dos algoritmos ou para fins estatísticos. A Figura 17 mostra a coleção “stats” que contém um único documento contendo todos esses dados.

Figura 17: Estatísticas do sistema



Fonte: própria

4.2 FUNCIONAMENTO DO HARDWARE

Além dos componentes básicos presente em qualquer aplicação de IoT, como jumpers, resistores e LEDs, o hardware do projeto é composto por alguns componentes distintos que desempenham papéis diferentes. A seguir, os principais deles serão citados e detalhados.

4.2.1 Servidor de imagens do ESP32-CAM

Pode-se dizer que o módulo de câmera ESP32-CAM é o componente principal deste projeto, pois ele foi o responsável pela captura das imagens utilizadas para a autenticação dos usuários. O módulo conta com Wi-fi integrado e assim pôde se conectar

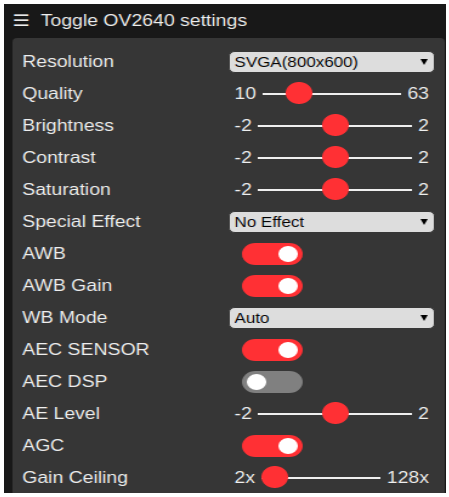
a uma rede local onde funcionou como um servidor Web, disponibilizando imagens em tempo real sempre que solicitado.

Visto que o módulo não possui nenhuma entrada USB, o carregamento do Sketch (programa) para placa teve que ser feito através dos pinos seriais, os quais foram conectados a uma placa Arduino Uno, que teve unicamente a função de alimentar e transferir o código para o módulo. O código utilizado para executar o servidor web foi o que vem por padrão nos exemplos da Arduino IDE, apenas algumas alterações foram feitas para inserir as credenciais necessárias para a conexão com a rede Wi-fi.

Ao ser iniciado, a primeira ação do módulo é se conectar à rede Wi-fi que foi configurada em seu código fonte. Em seguida, o servidor web é iniciado e é possível acessar as imagens através de qualquer navegador que esteja sendo executado em um dispositivo conectado a mesma rede que o módulo. O IP necessário para o acesso é obtido automaticamente pelo módulo e exibido no monitor serial que pode ser acessado também através da Arduino IDE.

Acessando o IP fornecido pelo módulo, pode-se visualizar uma tela com uma barra lateral como mostrado na Figura 18, onde é possível estar realizando vários ajustes na imagem da câmera, como resolução, qualidade, brilho, contraste, entre outros. Oferece ainda a opção de *stream* ou de capturar uma imagem.

Figura 18: Configurações da câmera do ESP32-CAM



Fonte: própria

4.2.2 Requisições feitas pelo ESP8266

A plataforma NodeMCU, a qual possui o microcontrolador ESP8266 e que também conta com um módulo de Wi-fi integrado é a responsável pela comunicação com o servidor principal, onde acontece o processamento das imagens. Ao ser ligada, sua primeira ação é se conectar à rede local via Wi-fi, as credenciais para a conexão são inseridas no código que foi carregado na placa.

A plataforma é a responsável por aguardar a ação do usuário, que por sua vez faz o acionamento do processo através do pressionamento de um botão, que ao ser pressionado, faz uma requisição ao servidor solicitando para que uma imagem seja capturada, nesse momento uma imagem é capturada e salva em uma pasta temporária.

Após a captura da imagem, uma outra solicitação é feita ao servidor, indicando que a imagem já foi salva e que agora aguarda pelo processo de reconhecimento. Então, o algoritmo Python responsável pelo processo de reconhecimento é executado e faz o processamento da imagem que foi capturada na etapa anterior. Ao final do processo, é feita uma consulta ao banco de dados afim de verificar se o usuário está cadastrado, e em seguida uma resposta é enviada de volta para o ESP.

Caso a resposta retornada seja positiva, a plataforma envia um sinal para o relé que está conectado ao fecho eletromagnético, o qual é disparado e a entrada é liberada. Caso seja negativa, uma mensagem é exibida no display de cristal líquido (LCD) que está conectado a placa, informando que o acesso foi negado.

A Tabela 4 mostra todos os componentes que estão conectados a plataforma e o respectivo papel de cada um.

Tabela 4: Componentes conectados a plataforma NodeMCU

Componente	**Função**
Display LCD	Interface de comunicação visual com o usuário.
Push Button	Acionamento do processo de autenticação.
Módulo relé	Acionamento do fecho eletromagnético.
Resistor de 1k omh	Resistência para ligação do *Push Button*.
Jumpers	Conexões

Fonte: própria

4.2.3 Display LCD

O Display LCD está conectado a plataforma NodeMCU e serve como interface visual para orientar e informar o usuário a respeito da tentativa de acesso. A Tabela 5 mostra todas as mensagens que são exibidas no mesmo e em quais ocasiões são exibidas.

Tabela 5: Mensagens do display LCD

Mensagem	Ocasião
Aguardando	Quando o sistema está ocioso esperando pelo acionamento do botão.
Capturando	Quando o sistema está capturando a imagem.
Processando	Quando o sistema está fazendo o processo de reconhecimento facial.
Acesso Liberado	Quando o usuário é encontrado na base de dados.
Acesso Negado	Quando o usuário não é encontrado na base de dados.

Fonte: própria

4.3 FUNCIONAMENTO DO APLICATIVO

O aplicativo que faz o gerenciamento do sistema, foi desenvolvido em React Native e está disponível tanto para Android quanto para iOS. Tem uma interface simples e direta, sua responsabilidade vai desde a parte de gerenciar usuários até a geração de relatório com informações referentes aos acessos ao ambiente. A seguir, serão mostradas as principais funcionalidades de cada tela.

4.3.1 Tela inicial

A tela principal do aplicativo é bem simples, contendo apenas a logo do sistema e os botões que redirecionam para as telas de acessos, usuários, configurações e estatísticas, como pode ser visto na Figura 19.

Figura 19: Tela inicial do aplicativo

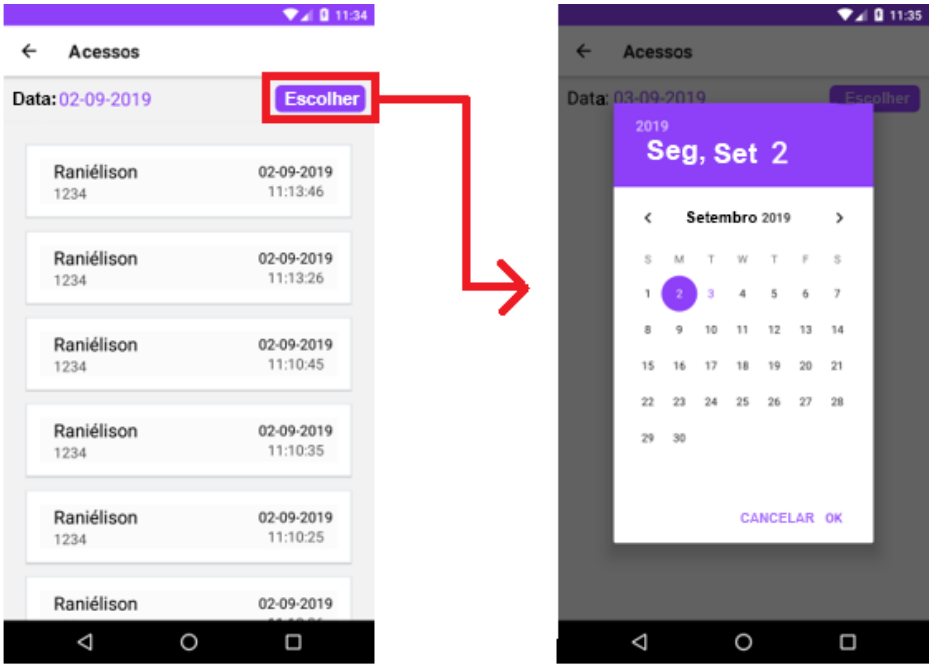
Fonte: própria

4.3.2 Ver acessos

Ao acessar o menu de acessos, a partir do menu principal, o administrador do sistema tem acesso a uma lista contendo todos os acessos que aconteceram em uma data escolhida por ele. Os dados exibidos nessa tela, são lidos do banco de dados do

Firebase, o que possibilita que se tenha acesso a esses dados de qualquer lugar que tenha conexão com a Internet. A tela é mostrada na Figura 20.

Figura 20: Visualização dos acessos

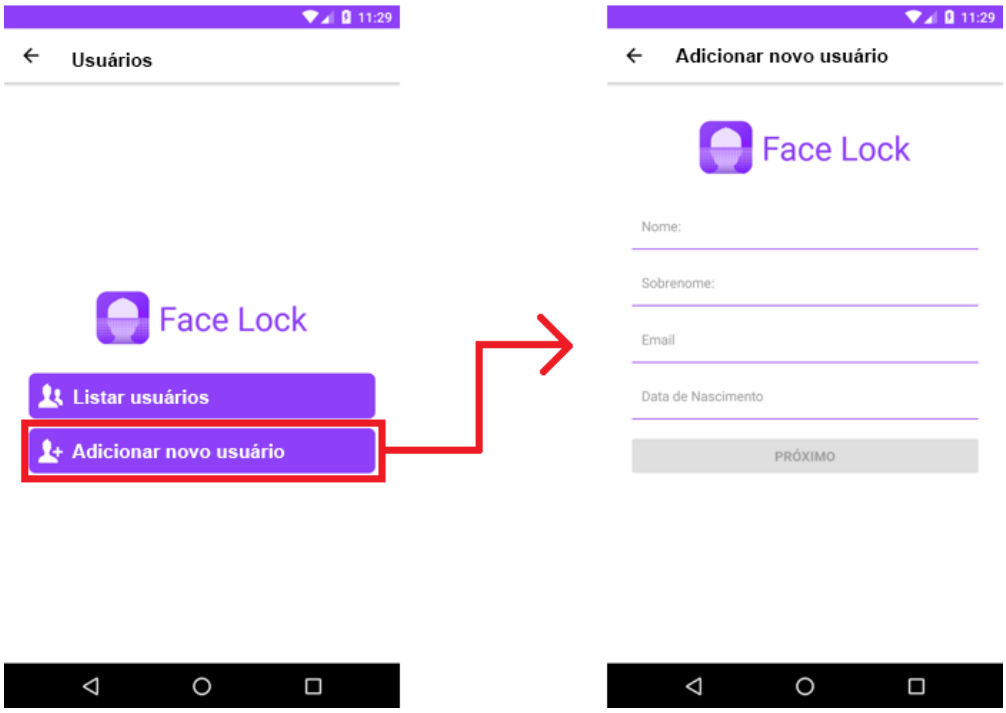


Fonte: própria

4.3.3 Cadastrar usuário

O cadastro de novos usuários pode ser feito a partir do menu de usuários, na opção “Adicionar novo usuário”. Ao ser selecionada é feito um redirecionamento para um formulário, o qual será preenchido algumas informações pessoais como mostrado na Figura 21.

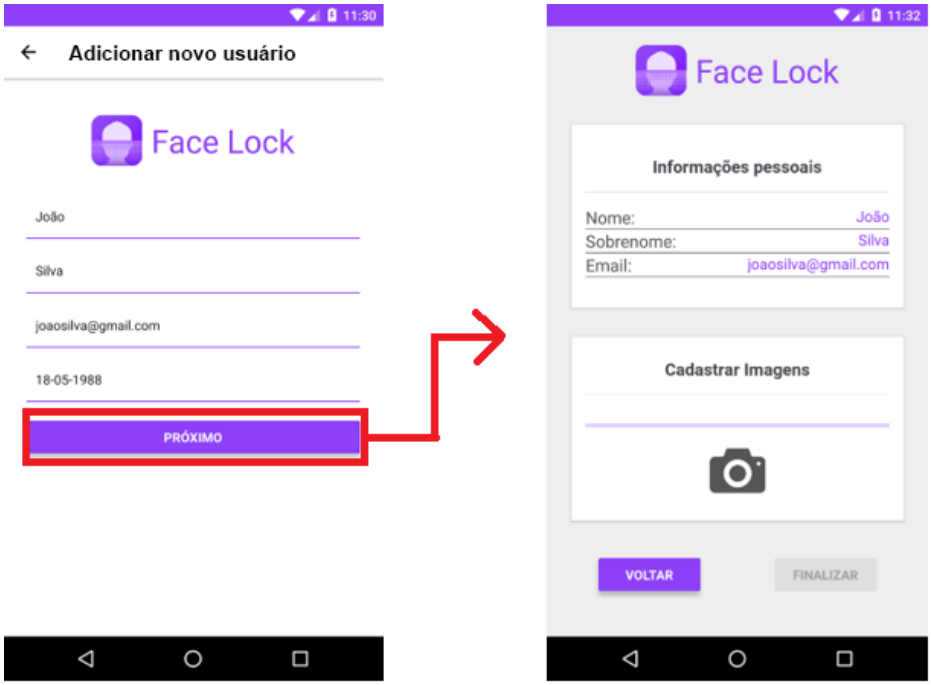
Figura 21: Cadastro de usuário



Fonte: própria

Em seguida, ao preencher todas as informações solicitadas, é possível passar para próxima etapa, que é o cadastro das imagens do usuário. Nessa tela (Figura 22), podem ser vistas as informações preenchidas na etapa anterior. É possível voltar, caso seja preciso fazer alguma alteração, ou prosseguir clicando no ícone da câmera para capturar as imagens. Ao capturar as 3 imagens solicitadas, a barra de progresso é preenchida e o botão de concluir é habilitado.

Figura 22: Cadastro de usuário etapa 2



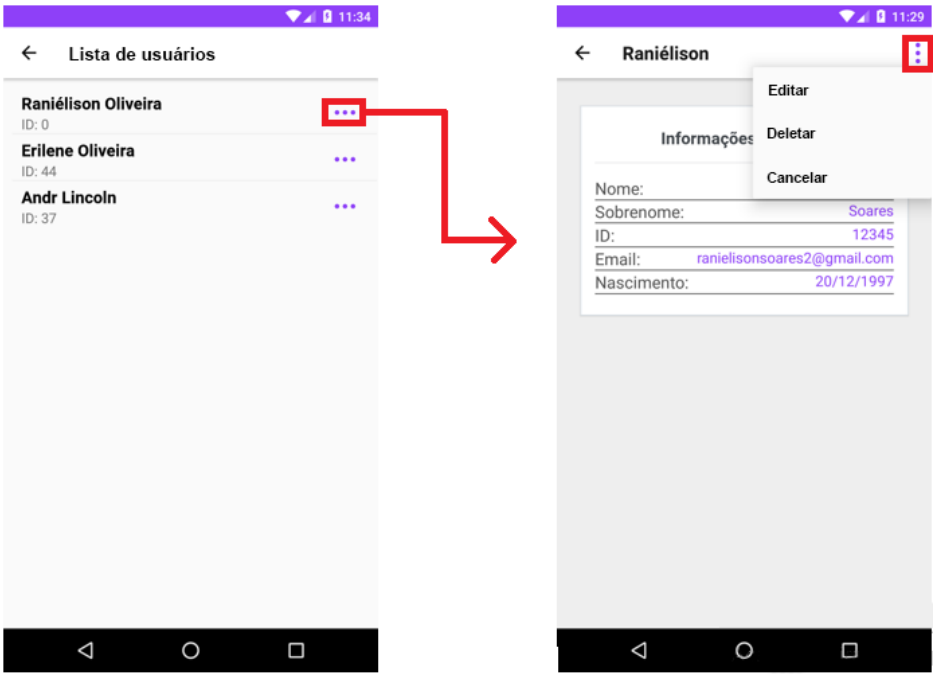
Fonte: própria

Ao clicar em concluir, os dados são enviados para o servidor, que armazena os dados do usuário no banco de dados e executa os algoritmos de extração de características, passando como entrada as imagens da face do usuário em questão. Como os dados armazenados são em forma de objeto (chave – valor), a chave que identifica os dados de cada usuário é o ID que é gerado automaticamente para cada novo usuário, concatenado com o número da imagem cadastrada.

4.3.4 Listar Usuários

Nessa tela é feita uma listagem de todos os usuários cadastrados no sistema. É uma lista simples que mostra apenas o nome e o ID do usuário como mostrado na Figura 23. Em cada usuário se tem uma opção para ver mais detalhes, onde nessa tela será possível ver mais informações do usuário e ter acesso a um menu com as opções de editar, ou excluir o usuário.

Figura 23: Listagem e visualização de usuário

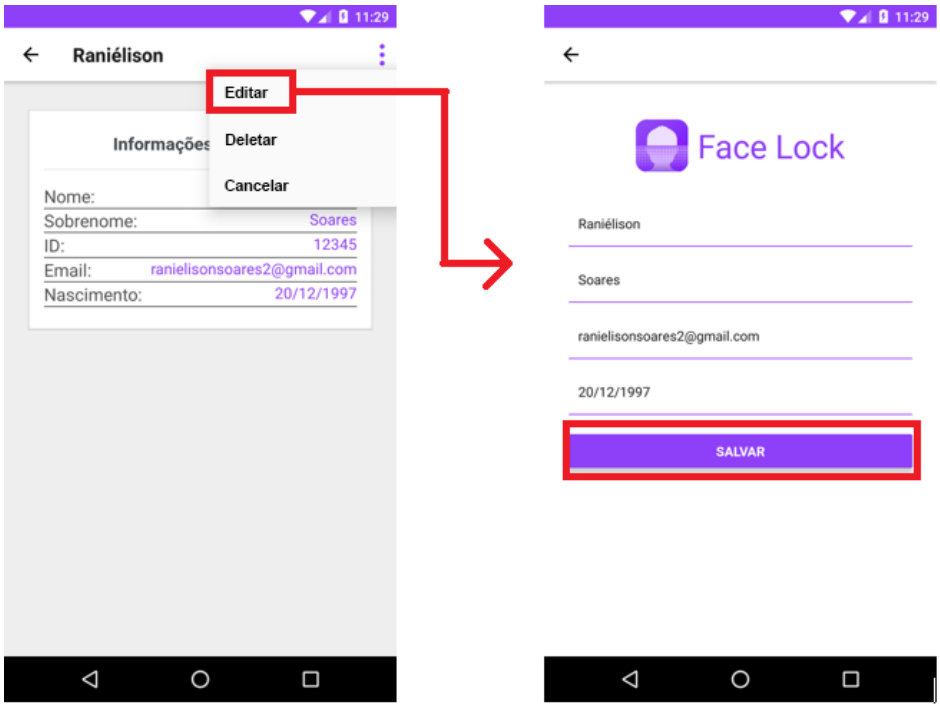


Fonte: própria

4.3.5 Editar Usuário

Ao clicar na opção de editar usuário, é feito um direcionamento para um formulário semelhante ao do cadastro onde serão carregadas as informações atuais, porém tendo a opção de editar as mesmas e em seguida salvar as alterações, como mostra a Figura 24.

Figura 24: Tela de edição de usuário

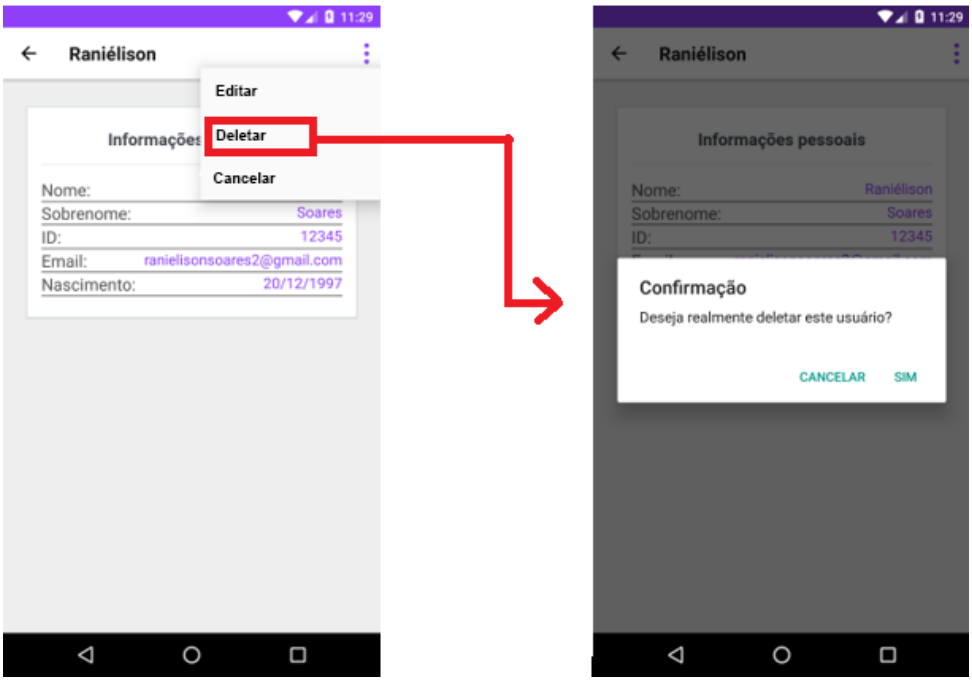


Fonte: própria

4.3.6 Deletar usuário

Para fazer a exclusão de um usuário cadastrado do banco de dados, o administrador do sistema pode acessar o menu no canto superior direito da tela de visualização de usuário, como mostra a Figura 25 e escolher a opção de deletar. Ao fazer isso e confirmar a ação, o servidor se encarrega de tirar os dados do usuário em questão do banco de dados e também do arquivo contendo os dados das imagens cadastradas, impedindo assim, que este usuário tenha acesso ao local. Porém, os dados referentes aos acessos feito pelo mesmo antes da remoção, permanecem salvos.

Figura 25: Deletar usuário



Fonte: própria

4.3.7 Configurações

A Figura 26 mostra a tela de configurações, onde administrador pode inserir as informações referente a IP e porta onde o servidor local está sendo executado. O armazenamento dessas informações é feito no próprio dispositivo através da biblioteca AsyncStorage, as mesmas permanecem salvas mesmo após o fechamento do aplicativo e podem ser atualizadas a qualquer momento.

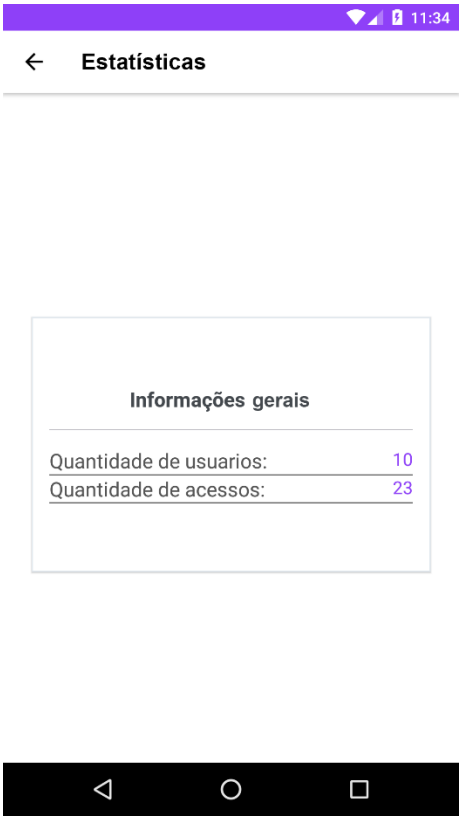
Figura 26: Configurações

Fonte: própria

4.3.8 Ver estatísticas

Nesta tela, que pode ser acessada através do menu principal, o administrador do sistema tem acesso a informações gerais do sistema. Informações como quantidade de usuários cadastrados e total geral de acessos ao ambiente desde a instalação do sistema.

Figura 27: Tela de estatísticas



Fonte: própria

4.3.9 Comunicação com o servidor

Toda a comunicação do aplicativo é feita com o Firebase para acessar o registro de acessos e com o servidor local para fazer o gerenciamento de usuários. Vale ressaltar que a visualização do registro de acessos pode ser feita de qualquer lugar, desde que o aparelho tenha conexão com a Internet. Quanto ao gerenciamento de usuários, o aparelho precisa estar conectado na mesma rede que o servidor local, dispensando assim o uso da Internet.

4.4 SOFTWARES UTILIZADOS

Para o desenvolvimento do código, foi utilizado o Visual Studio Code que é uma IDE gratuita da Microsoft e oferece uma série de recursos para o desenvolvimento de aplicações de médio e grande porte. Além da sua boa organização da estrutura de pastas do projeto, o mesmo permite a instalação de diversos *plugins* que auxiliam a escrita do código e automatiza alguns procedimentos corriqueiros do dia a dia do desenvolvedor

Para os testes na API foi utilizado o Insomnia que é uma interface que auxilia no teste de API's. Por ele é possível fazer requisições a APIS utilizando todos os métodos, GET, POST, PUT, DELETE, permite ainda o envio de dados no corpo e nos cabeçalhos das requisições além de mostrar de forma formatada e organizada o retorno das mesmas.

Para a visualização dos dados do banco de dados foi utilizado o Robo3T que é um software específico para a visualização de bases de dados do MongoDB. Oferece uma interface visual para o gerenciamento dos esquemas do banco, dando a possibilidade de adicionar, editar ou excluir documentos do banco.

5 RESULTADOS OBTIDOS

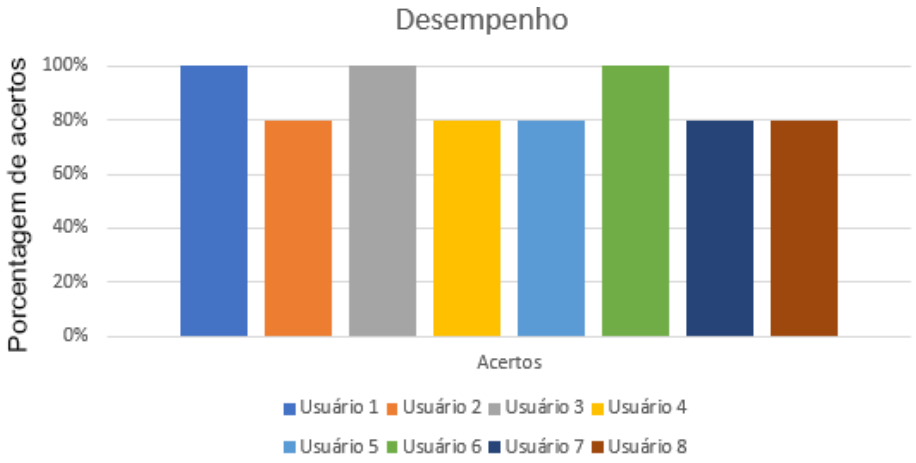
Ao final do desenvolvimento, o projeto foi submetido à etapa de validação. Neste capítulo serão apresentados os resultados obtidos em alguns quesitos importantes na avaliação do sistema, são eles: testes de desempenho, comparativo entre similares e custo de implementação. Os mesmos serão detalhados a seguir.

5.1 DESEMPENHO

Os testes de desempenho do reconhecimento se deram através de simulações de acesso ao ambiente, aconteceram no laboratório de aprendizagem robótica da UERN e as simulações foram feitas com alunos que acessam o laboratório frequentemente. O tempo de resposta é de aproximadamente 7 segundos entre o acionamento do processo de autenticação, que inicia com o pressionamento do botão, até a liberação ou negação do acesso. A Figura 28 mostra os resultados obtidos.

Para esses testes foi estabelecido um limiar de 0.52 para a *face recognition* assumir que o usuário foi reconhecido. Esse valor se baseia em testes anteriores feitos, que buscaram o ajuste desse limiar, procurando adaptá-lo à características das imagens capturadas pelo sistema no ambiente que será instalado (iluminação e localização do sistema).

Figura 28: Resultados obtidos



Fonte: própria

5.2 COMPARATIVO ENTRE SIMILARES

Como o foco deste projeto foi o desenvolvimento de um sistema de controle de acesso, visando atender as necessidades apresentadas e ao mesmo tempo oferecendo um bom custo benefício. Foi realizada uma pesquisa no mercado a fim de encontrar sistemas similares ao aqui desenvolvido e fazer uma comparação de custo benefício entre eles. A Tabela 6 mostra cada modelo e suas respectivas especificações e preço de cada um.

O objetivo não é comparar produtos, mas sim, mostrar que é possível construir um sistema de autenticação por biometria facial, com funcionalidades similares à sistemas encontrados no mercado, com custo mais reduzidos. Além disso, o sistema aqui proposto apresenta pontos vantajosos como, possibilidade de gerenciamento e acesso a relatórios pela Internet.

Tabela 6: Comparativo entre similares

	Inttelix Face Lock	**Desenvolvido**
Autenticação	Biometria Facial, senha, tag	Biometria Facial
Capacidade	100 faces, 100 senha, 100 tags	Varia de acordo com a capacidade de armazenamento do servidor
Tamanho	33,5 x 76 x 86 mm	90 x 100 x 25mm
Alimentação	rede elétrica	Fonte 5V
Servidor externo	Não utiliza	Utiliza
Gerenciamento	Possui	Possui
Valor	R$ 3.590,00	R$ 526,15

Fonte: própria

O sistema desenvolvido atendeu ao objetivo do trabalho ao ter apresentado o melhor custo benefício.

5.3 COTAÇÃO DE VALORES

Afim de calcular o custo final do protótipo desenvolvido foi feita uma cotação dos valores dos componentes utilizados no desenvolvimento do projeto. Durante o desenvolvimento e testes do protótipo, foi utilizado um computador desktop com ambiente Linux para a execução do servidor, porém, essa função pode ser desempenhada por um computador de baixo custo como o Raspberry.

A Tabela 7 mostra os valores de cada componente necessário para a execução do projeto, vale ressaltar que esses valores foram obtidos em uma pesquisa de valores feita em lojas brasileiras. Há uma possibilidade de reduzir ainda mais o custo final ao fazer a importação dos componentes.

Tabela 7: Custo dos componentes

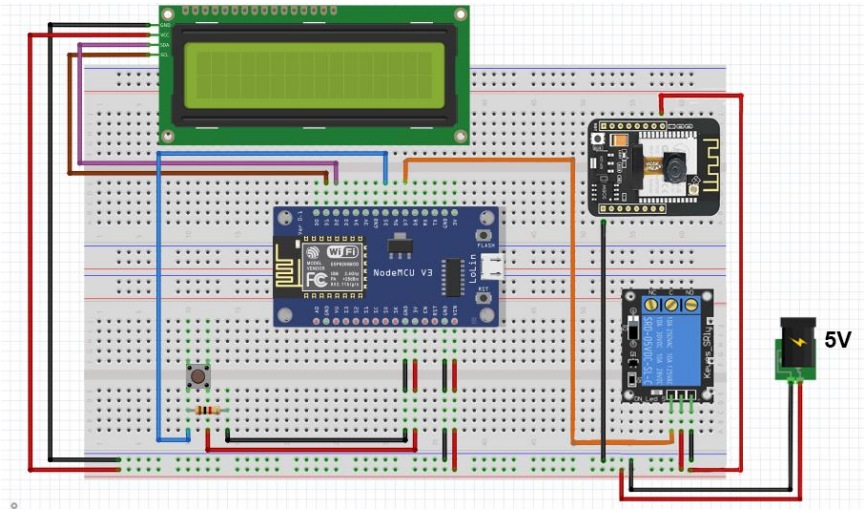
Componente	**Valor (R$)**
ESP32-CAM	97,90
ESP8266	29,90
Display LCD	29,90
Módulo relé	9,90
Push button	0,34
Servidor externo: Raspberry Pi 3 Model B+ Anatel	269,90
Fecho eletromagnético	71,42
Fonte de alimentação	15,90
Resistor 1k omh	0,99
Total	**526,15**

Fonte: própria

5.4 SISTEMA CONCLUÍDO

Para a montagem do sistema, além dos componentes mencionados no Capítulo 4 foram utilizadas duas protoboards que possibilitaram a conexão entre todos os componentes, posteriormente será desenvolvido um *case* para organizar todas as partes do sistema. Na Figura 29 podem ser vistas todas as conexões em um modelo desenhado.

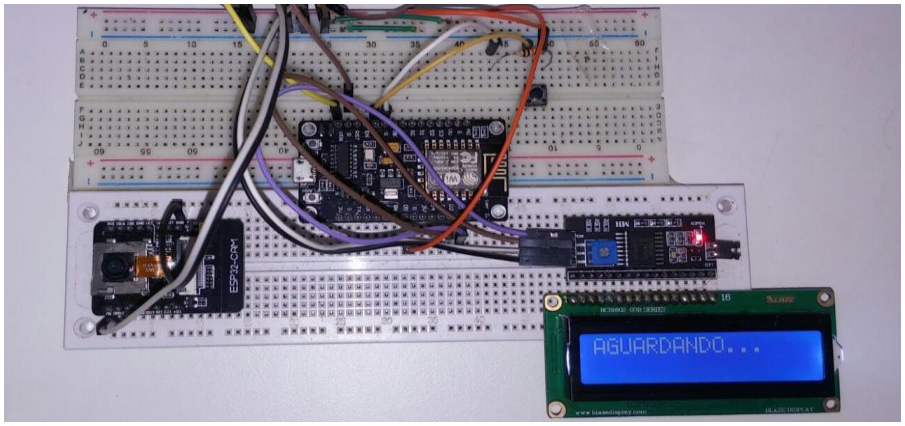
Figura 29: Modelo desenhado do sistema



Fonte: própria

A Figura 30 mostra o equipamento real montado.

Figura 30: Sistema real



Fonte: própria

6 CONCLUSÃO

Este trabalho possibilitou entender como a tecnologia pode automatizar as tarefas diárias das pessoas. Com isso pôde-se apresentar a aplicação de tais tecnologias para automatizar o processo de controle e gerenciamento de acesso a ambientes restritos, bem como apresentar todas as vantagens que se tem ao se fazer o uso das mesmas.

Diante disso, a pesquisa teve como objetivo geral desenvolver um sistema de controle e gerenciamento de acessos a ambientes restritos, visando não só atender as necessidades apresentadas como também apresentar um baixo custo se comparado aos sistemas de propósito similar que existem no mercado.

Algumas dificuldades foram encontradas no decorrer do projeto, algumas se deram devido a atrasos na comunicação entre o equipamento e o servidor, onde foi necessário usar de artifícios de programação para contornar os erros. Também relacionado ao processo de reconhecimento facial, que no geral apresenta algumas falhas de segurança que ainda não foram totalmente sanadas, falhas ocorridas devido a incapacidade da câmera em detectar se a imagem apresentada é de fato uma pessoa ou uma imagem da pessoa.

Durante o desenvolvimento da aplicação pôde se evidenciar a eficiência e a capacidade de desenvolver soluções de automação fazendo o uso de tecnologias de baixo custo. Exemplo disto é plataforma ESP que foi a base deste projeto, e conta com uma grande quantidade de funcionalidades, podendo ser adquirida a um baixo custo.

O sistema cumpriu seus objetivos ao atender os requisitos de projeto e conseguir bons resultados nos testes realizados na etapa de validação, conseguindo fazer a autenticação dos usuários de forma correta em quase 100% dos casos. Além disso, o custo final do sistema ficou abaixo dos sistemas similares existentes no mercado.

6.1 TRABALHOS FUTUROS

Para dar continuidade e aprimorar o projeto desenvolvido, foi pensado em algumas melhorias e adições de novas funcionalidades ao sistema que podem ser implementadas em trabalhos futuros. Algumas delas serão descritas a seguir:

- Desenvolver uma interface web para um melhor gerenciamento dos dados do sistema;
- Adicionar funcionalidades de limitar os acessos ao ambiente de acordo com a data e o horário;
- Notificações no aplicativo móvel sempre que alguém não autorizado tente fazer o acesso ao local;
- Implementar um módulo de configurações do sistema que possibilite a alteração das credenciais de conexão Wi-fi e credenciais do servidor sem precisar recarregar o código nas placas;
- Implementar um meio de autenticação alternativo, porém seguro, para casos em que autenticação por reconhecimento facial não funcione por algum motivo não previsto;
- Estudar meios de sanar as falhas relacionadas a segurança no processo de reconhecimento facial.

REFERÊNCIAS BIBLIOGRÁFICAS

BERTOLETI, P. **Projetos com ESP32 e LoRa**. São Paulo: Editora NCB, 2019.

CHAUDHURI, A. **Internet of Things, for Things, and by Things.** Boca Raton, Florida: CRC Press, 2018

CMAKE. Disponível em <https://cmake.org/>. Acesso em 18 set. 2019.

DLIB. Disponível em < http://dlib.net />. Acesso em 18 set. 2019

DUARTE, L. Boas práticas com MongoDB. **Umbler Blog**. Disponível em: <https://blog.umbler.com/br/boas-praticas-com-mongodb/>. Acesso em 18 set. 2019.

FIREBASE. Disponível em <https://firebase.google.com/>. Acesso em 18 set. 2019

GOIS, Adrian. **Ionic Framework: Construa aplicativos para todas as plataformas mobile.** São Paulo: Casa do Código, 2017.

HOSTGATOR. O que é Internet das Coisas e como funciona. **HostGator**. Disponível em: <https://www.hostgator.com.br/blog/internet-das-coisas/>. Acesso em 18 set. 2019.

KÜNZEL, P. Vantagens e Desvantagens de usar o Node.JS. **Mundo JS**. Disponível em: <https://www.mundojs.com.br/2018/10/16/vantagens-e-desvantagens-de-usar-o-node-js/#page-content>. Acesso em: 18 set. 2019.

LA CRUZ, J. D., & LA CRUZ, E. D. **Automação predial 4.0: a automação predial na quarta revolução.** Rio de Janeiro: Brasport, 2019.

LANA, H. C. **Projetos Maker.** São Paulo: Novatec Editora Ltda, 2018.

MARENGONI, M., & STRINGHINI, D. Tutorial: Introdução à Visão Computacional usando OpenCV. **RITA**, 16, p. 125-160, 2009.

MARQUES, C., PIFFER, E., MIORANZA, I. C., LIMA, L. C de., PIETCHAKI, A., ROCHA, C. E. D. S., FERRO, E. F. S., ANTONIASSI, G. S., SILVA, J. E. M. da. Desenvolvimento de uma aplicação de controle de presenças de acadêmicos com uso de reconhecimento através de biometria, **Akrópolis**, v. 25, n. 1, p. 81-88, 2017.

MORAES, J. L. **Controle de acesso baseado em biometria facial**, 2010. Monografia (Programa de pós-graduação em informática) Universidade Federal do Espírito Santo.

NUMPY. Disponível em < https://www.numpy.org />. Acesso em 18 set. 2019.

OLIVEIRA, S. D. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi.** São Paulo: Novatec Editora Ltda, 2017.

OPENCV. Disponível em <https://docs.opencv.org/4.1.1/da/d60/tutorial_face_main.html />. Acesso em 18 set. 2019.

OPENFACE. Disponível em < https://cmusatyalab.github.io/openface />. Acesso em 18 set. 2019.

PEIXOTO, T. M. **Sistema de controle de acesso utilizando dispositivos embarcados**, 2013. (Monografia) Universidade Federal de Juiz de Fora.

PEREIRA, C. R. **Construindo APIs REST com Node.js, 2016**, São Paulo: Casa do código, 2016.

SANTOS, A. L. **Gerenciamento de Identidades.** Rio de Janeiro: Brasport, 2007.

SANTOS, G. Node.js — O que é, por que usar e primeiros passos. **Medium**. Disponível em: <https://medium.com/thdesenvolvedores/node-js-o-que-%C3%A9-por-que-usar-e-primeiros-passos-1118f771b889>. Acesso em: 18 set. 2019.

SILVA, G. P. de. **Sistema de controle de acesso utilizando Arduino**, 2014. (Monografia) Universidade do Estado do Rio Grande do Norte.

STEVAN JUNIOR, S. L. **Internet das Coisas - Fundamentos e Aplicações em Arduino e NodeMCU.** São Paulo: Saraiva, 2018.

STEVAN JUNIOR, S. L., & FARINELLI, F. A. **DOMÓTICA - Automação Residencial e Casas Inteligentes com Arduino e ESP8266.** São Paulo: Érica Ltda, 2019.

TURATI, C., CASSIA, V. M., SIMION, F., & LEO, I. Newborns' face recognition: Role of inner and outer facial features. **Child development**, p. 297–311, 2006.

ZHAO, W., CHELLAPPA, R., PHILLIPS, P. J., & ROSENFELD, A. Face recognition: A literature survey. **Acm Computing Surveys (CSUR)**, p. 399–458, 2003.

ZHENG, F. **International Conference on Materials, Architecture and Engineering.** Lancaster, Pensilvânia: Destech Publications Inc, 2014.