UNIVERSIDADE ESTADUAL DO MARANHÃO
CENTRO DE CIÊNCIAS TECNOLÓGICAS
CURSO DE ENGENHARIA DE COMPUTAÇÃO

DESENVOLVIMENTO DE UM APLICATIVO MOBILE MICROCONTROLADO PARA BUSCA DE VAGAS DE ESTACIONAMENTO

HUMBERTO DE JESUS FERREIRA JUNIOR

SÃO LUÍS
2019

HUMBERTO DE JESUS FERREIRA JUNIOR

DESENVOLVIMENTO DE UM APLICATIVO MOBILE MICROCONTROLADO PARA BUSCA DE VAGAS DE ESTACIONAMENTO

Trabalho de Conclusão de Curso apresentado à Banca Examinadora do Curso de Engenharia de Computação, da Universidade Estadual do Maranhão, como pré-requisito para a obtenção de título de Bacharel em Engenharia de Computação.

Orientador: Wesley Batista Dominices de Araujo

SÃO LUÍS
2019

Junior, Humberto de Jesus Ferreira.

Desenvolvimento de um aplicativo mobile microcontrolado para busca de vagas / Humberto de Jesus Ferreira Junior. – São Luís, 2019.

63 f.

Monografia (Graduação) – Curso de Engenharia de Computação, Universidade Estadual do Maranhão, 2019.

Orientador: Prof. Me. Wesley Batista Dominices de Araujo.

1. Rede WI-FI. 2. Microcontrolador. 3. Aplicação mobile. I. Título.

HUMBERTO DE JESUS FERREIRA JUNIOR

DESENVOLVIMENTO DE UM APLICATIVO MOBILE MICROCONTROLADO PARA BUSCA DE VAGAS DE ESTACIONAMENTO

Trabalho de Conclusão de Curso apresentado à Banca Examinadora do Curso de Engenharia de Computação, da Universidade Estadual do Maranhão, como pré-requisito para a obtenção de título de Bacharel em Engenharia de Computação.

Aprovado em: _____/_____/_____

BANCA EXAMINADORA:

Prof. Me. Wesley Batista Dominices de Araujo (Orientador)
Departamento de Engenharia da Computação
Universidade Estadual do Maranhão

Prof. Dr. Leonardo Henrique Gonsioroski Furtado da Silva (1° membro)
Departamento de Engenharia da Computação
Universidade Estadual do Maranhão

Prof. Me. Pedro Brandão Neto (2° membro)
Departamento de Engenharia da Computação
Universidade Estadual do Maranhão

SÃO LUÍS
2019

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus, pois, apesar de todos os reveses da vida, me manteve focado no que realmente importa, me concedeu saúde, discernimento e maturidade para terminar esse curso. E, por mesmo com diversas aulas aos sábados ter me mantido em seus caminhos e fiel em Seu propósito.

Aos meus pais Ellen Christine Bastos Ferreira e Humberto de Jesus Ferreira por me guiarem no caminho correto de Deus, me incentivando sempre a estudar e a correr atrás dos meus objetivos, sempre me dando o necessário para tal, com educação, boa alimentação e diversão. E irmãos por me ajudarem em diversos caminhos na universidade principalmente minha irmã Héllia Karoline Bastos Ferreira.

Aos meus professores e amigos, que me ajudaram muito nesse caminho me incentivando e ajudando cada um à sua maneira a terminar o curso da melhor forma possível.

Agradecer ao meu orientador Prof. Me. Wesley Batista Dominices de Araujo e amigo que me ajudou muito nesse projeto, identificando erros e soluções que fizeram com que esse projeto tivesse êxito e boas soluções para o problema que ele visa resolver.

Finalmente agradecer a Universidade Estadual do Maranhão, que apesar das limitações me proporcionou uma graduação de qualidade, com ótimos professores, salas confortáveis e almoços na maioria das vezes saborosos.

“Saiba que são suas decisões, e não suas condições, que determinam seu destino”

(Anthony Robbins)

LISTA DE FIGURAS

LISTA DE SIGLAS

API	Application Programming Interface
CPF	Cadastro de Pessoa Física
CPU	Central Processing Unit
GPS	Global Positioning System
HTML	Hyper Text Markup Language
IBGE	Instituto Brasileiro de Geografia e Estatística
ICR	Infrared Cut Removable
IDE	Integrated Development Environment
IEEE	Institute of Eletrical and Eletronic Engineers
IoT	Internet of Things
IP	Protocolo de Internet
MDF	Medium Density Fiberboard
MIT	Massachusetts Institute of Technology
OCR	Optical Character Recognition
OFDM	Orthogonal Frequency Division Multiplexing
OTA	Over The Air
PIR	Passive Infrared
PWM	Pulse Width Modulation
QR	Quick Response
RFID	Radio-Frequency IDentification
SMS	Short Message Service
SOC	System on chip
TCP	Transmission Control Protocol
UTP	Unshielded Twisted Pair

LISTA DE TABELAS

RESUMO

Com o crescimento substancial da população e facilidade de compra de carros através de créditos facilitados, houve um crescimento na polução de carros nas cidades, gerando por conseguintes grandes dificuldades para estacionar nas mesmas. Tendo em vista que, a tecnologia veio para facilitar as tarefas humanas e poupar tempo para o que realmente é importante, esse trabalho visa facilitar a vida dos motoristas e poupar seu tempo para encontrar vagas no estacionamento. Usando para isso a rede WI-FI local para comunicação com a internet, um microcontrolador conectado à rede que tem como função controlar o motor, ler os dados que vem do sensor e os alocar em uma página HTML, que se localiza em um endereço da rede, sendo utilizada por uma aplicação mobile para plataforma Android que permitirá ao cliente localizar as vagas e saber a quantidade de vagas livres, possibilitando também o cliente se cadastrar e ter acesso a todas as utilidades do aplicativo, tendo como foco a facilidade do acesso remoto, conforto e economia.

Palavras-chave: Rede WI-FI. Microcontrolador. Aplicação mobile.

ABSTRACT

With the substantial growth of the population and the ease of buying cars through easy credits, there was a growth in the pollution of cars in the cities, resulting in great difficulties to park in them. Given that technology has come to ease human tasks and save time for what's really important, this work is meant to make drivers' lives easier and save their time finding parking spaces. Using the local WI-FI network to communicate with the internet, a network-connected microcontroller that has the function of controlling the motor, reading the data coming from the sensor and allocating it to an HTML page, which is located at a web address. network, being used by a mobile application for Android platform that will allow the customer to locate the vacancies and to know the amount of vacancies, also allowing the customer to register and have access to all the utilities of the application, focusing on the ease of remote access. , comfort and economy.

Keywords: *WIFI network. Microcontroller. Mobile App*

SUMÁRIO

1 INTRODUÇÃO

Na década de 1920, devido ao aumento considerável de carros nas cidades e a falta de lugar para colocá-los, começaram a ser ocupados lugares que antes eram destinados a passagem de pessoas, para solucionar esse problema se fez necessário a criação de estacionamentos. Na década atual com aumento da população e os créditos fáceis para financiamentos de carros esse problema se repete, com o aumento exponencial.

No Brasil, entre 2001 a 2011, o aumento do número de carros nas capitais foi de 11,5 milhões para 20,5 milhões, conforme o Observatório das Metrópole e o Departamento Nacional de Trânsito (Denatran, 2012). Isso tem dificultado a vida dos motoristas para estacionar, custando tempo e muito trabalho para achar vagas perto do seu destino.

Na atualidade esse problema está longe de ter uma solução, os estacionamentos vêm sendo um forte vilão à mobilidade urbana por ocuparem espaços importantes, os quais poderiam ser utilizados como calçadas e ciclovias. Mas, ao mesmo tempo podem ser um importante aliado, basta imaginar uma cidade como São Luís, com 1 milhão de habitantes, e cerca de 420 mil veículos (IBGE, 2018), o que resulta em uma média de 0,42 veículos por habitante, sem vagas de estacionamento suficientes a circulação de veículos e pessoas ficaria amplamente comprometida.

Os problemas de estacionamento só vêm aumentando, o que vem dificultando achar uma vaga para estacionar os veículos, seja perto da loja ou perto do seu local de trabalho. O objetivo desse projeto é, portanto, contornar esses obstáculos criando maneiras mais fáceis e rápidas de encontrar vagas, através de uma solução inteligente, utilizando o conceito que chamamos Internet das Coisas.

Internet of Things (Internet das Coisas), esse conceito onde tudo está conectado não é tão novo, ela surgiu no MIT (*Massachusetts Institute of Technology*) em 1999, contudo na época não existiam conexões rápidas o suficiente para a utilização dessa tecnologia. Na atualidade, entretanto, com o uso de meios de comunicação mais rápidos (banda larga, 3g e 4g) ela se faz muito presente em nossas vidas, em celulares, televisões, geladeiras e até mesmo em estacionamentos.

Segundo escreveu Ashton (Jornal de RFID,1999): “Se tivéssemos computadores que soubessem tudo sobre as coisas em geral - usando dados que coletassem sem a nossa ajuda - seríamos capazes de rastrear e contar tudo, e reduzir bastante o desperdício, a perda e os custos” (Cisco,2013, p.2).

Hoje em dia existem uma série de facilidades que diminuem as perdas e os custos nos estacionamentos, como luzes mostrando quais vagas estão ocupadas, painéis sinalizando o número de vagas por setor, sistemas de reservas de vagas e até mesmo pessoas que indicam onde há vagas o que reduz significativamente a perda de tempo.

Contudo, a solução que esse trabalho se propõem a dá é a utilização de um aplicativo, um microcontrolador e a rede interna do local, para conectar o cliente ao estacionamento de casa, ou, até mesmo no caminho para lá, fazendo com que ele tenha uma noção de quantas vagas existem no local e quais estão livres, gerando uma locomoção mais rápida até as vagas livres.

1.1 OBJETIVOS

Tendo em vista os problemas enfrentados pelos motoristas e com o intuito de amenizá-los, este trabalho estabelece o objetivo geral e os específicos, a seguir.

1.1.1 Objetivo Geral

Desenvolver um protótipo computacional que auxilie o processo de busca por vagas em estacionamentos.

1.1.2 Objetivos Específicos

- Desenvolver um aplicativo didático e fácil de manusear, que conecte o motorista aos dados coletados e transmitidos pelo microcontrolador, dessa forma permitindo que o mesmo tenha uma visão das vagas e sua localização, agilizando assim o processo de localização e poupando tempo ao usuário;
- Conectar o sensor ao ESP8266 e configurá-lo;
- Executar testes para analisar as funcionalidades de todas as partes do sistema e assim confirmar o seu funcionamento da forma esperada;
- Verificação dos custos da aplicação do projeto para um estacionamento;
- Demonstrar de forma funcional o projeto através da utilização do estacionamento da Universidade Estadual do Maranhão e a rede interna da mesma;

1.2 ESTRUTURA DO TRABALHO

Este trabalho está divido em cinco capítulos. No primeiro capítulo foi feita uma Introdução dissertando sobre como se fez necessário à criação dos estacionamentos, de sua importância para mobilidade urbana e de como hoje em dia o motorista vem tendo problemas para realizar essa tarefa, ressalta-se também a importância de tudo estar conectado (Internet das Coisas) e de como isso pode ajudar quanto a problemática de estacionar, concluindo com nossa solução para resolução do problema. Os componentes que se agregam a esse capítulo são os objetivos gerais e específicos, que deixam claro os pontos trabalhados nesse trabalho de conclusão de curso.

O segundo capítulo aborda a fundamentação teórica na qual foi baseada o trabalho, se tratando de conceituar teoricamente cada etapa do mesmo.

O terceiro capítulo, é composto pela metodologia que dá a ideia de como foi desenvolvido esse projeto.

O quarto capítulo, visa tratar das informações detalhadas dos componentes físicos e os métodos utilizados no projeto, ou seja, o Hardware, tendo a presença de todos os *datasheets* para que se possa ter um conhecimento mais aprofundado.

O quinto capítulo, tem como objetivo explanar o desenvolvimento do projeto perpassando pela montagem, nesse descrevendo como foi feito e de que maneira foram realizados os encaixes; a programação na plataforma da placa, com a descrição do programa desenvolvido e apresentação de imagens comentadas sobre o seu funcionamento; o desenvolvimento do aplicativo e do banco de dados, usando imagens do aplicativo e da programação para auxiliar o entendimento, analisando por último a sua aplicação e viabilidade comercial.

O sexto capítulo serão mostrados os testes realizados neste projeto, como teste das conexões, do software do WeMos D1 R1 e do aplicativo.

O sétimo e último capítulo, irá tratar da conclusão do trabalho, incluindo sucessos e os percalços que aconteceram ao longo do projeto, terminando com propostas para melhorá-lo, tornando o trabalho ainda mais atrativo.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados conceitos teóricos que ajudaram na elaboração do projeto, incluindo Figuras, Tabelas e pesquisas, que contribuirão para o entendimento.

2.1 AUTOMAÇÃO DOS ESTACIONAMENTOS

Com a população crescendo rapidamente, principalmente nas grandes cidades, e, tendo em vista, o desejo da maioria dos habitantes em ter seu carro próprio, interligando ainda a esse fator a facilidade de financiamentos de carros oferecidos por bancos com parcelas baixas, atrelado a disponibilidade de um transporte público precário, a tendência é o aumento da quantidade de carros. Esse aumento, portanto, gera grandes congestionamentos em cidades resultando em uma busca muito grande por vagas de estacionamento, até mesmo porque um veículo se dirige de um estacionamento para outro. Em São Paulo, por exemplo, existe a Lei nº 15.150 de 2010, que regulamenta que as construções devem ter um mínimo de vagas de estacionamentos para serem construídas levando em conta o seu tamanho, esta Lei já sofreu algumas alterações pela Lei de 16.642 de 2017.

Uma pesquisa realizada em São Paulo pelo EY Consultoria, verificou que em 15 distritos na capital paulista se tem um número de 384 mil vagas, tendo 509 mil carros que se locomovem para região, ou seja, existem 125 mil motoristas que não encontram vagas e ficam circulando pela cidade a procura delas, o que gera um desperdício de tempo, gasto de gasolina e uma poluição acentuada pela emissão de CO2.

Sabendo desses problemas, o avanço na tecnologia tornou possível alcançar um conforto com a automação dos estacionamentos, tornando possível os chamados “estacionamento inteligentes” que se tornaram uma necessidade nas grandes cidades, com o auxílio deles há uma diminuição do tempo de procura, economia de gasolina e diminuição da poluição causada pelos carros na hora de procurar vagas, facilitando, dessa forma, a vida dos motoristas.

Essa tecnologia de automação dos estacionamentos, não é somente uma mercadoria a ser vendida para estabelecimentos grandes, mas também em pequenos, condomínios, por exemplo, posto algumas soluções não serem muito caras para implementar, tendo em vista a facilidades que geram e a qualidade de vida melhor para os motoristas e meio ambiente.

Algo importante de ressaltar são os tipos de soluções e que tecnologia elas utilizam para chegar a objetivos semelhantes. Começando por soluções que usam a tecnologia de reconhecimento de imagens através de câmeras ICR/OCR (*Infrared Cut Removable /Optical*

Character Recognition) para averiguar presença de carros e placas, podendo ser utilizadas para levantar a cancela na presença de carros que têm determinadas placas, com a utilidade também de seu reconhecimento fazer a verificação de vagas no estacionamento.

Outra tecnologia utilizada para acesso autorizado a vagas de estacionamento são os leitores RFID (*Radio-Frequency IDentification*) que funcionam utilizando rádio frequência, que se dá através de uma tag RFID presente no carro a qual se comunica com a cancela do estabelecimento permitindo acesso, podendo facilitar muito em alguns tipos de estacionamento como os de condomínios.

Há também tecnologias que utilizam sensores (ópticos, ultrassonoros e mecânicos) para informar a presença de carros em determinadas vagas para assim ter noção de quais vagas estão livres e a quantidade de vagas no estacionamento.

2.2 INTERNET DAS COISAS (IOT)

A Internet das Coisas (no inglês *Internet of Things* (IoT)), é uma conexão de objetos a uma rede de comunicação a qual proporciona que ele envie ou receba informações, interagindo com pessoas ou outros dispositivos. Apesar de acharmos que esse conceito é novo, por ser algo tão interligado com as novas tecnologias, ele foi criado em 1999 por Kevin Ashton em seu trabalho com a tecnologia RFID (*Radio-Frequency IDentification*), onde até então se pensava que o conceito (IoT) estava ligado somente tecnologia (RFID).

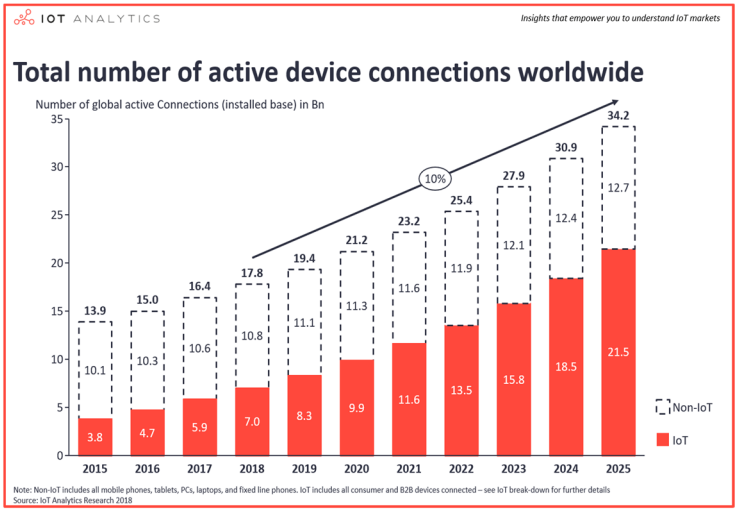
Na atualidade IoT se desenvolveu muito, devido aos avanços da tecnologia, podendo hoje ser encontrado em várias áreas, industrial (máquinas que funcionam com auxílio de sensores e se conectando à nuvem para tomar decisões, e sistema de monitoramento industrial agilizando as tomadas de decisão), empresarial (programas de gerenciamento online, que através de dados depositados auxiliam na tomada de decisão da empresa), agrícola (Equipamentos como colheitadeiras autônomas que funcionam por GPS (*Global Positioning System*) e drones que auxiliam no monitoramento da colheita), residencial (Automação de iluminação, portas, cortinas e equipamentos (Tv, geladeira, micro-ondas, *webcams* e consoles de vídeo game), urbana (estacionamentos automatizados, câmeras e até sinais inteligentes) e hospitalar (aplicativos de saúde onde se tem prontuários médicos e banco de dados de doenças, marca-passos inteligentes).

A Internet das Coisas expandiu o alcance da internet, ano após ano existem mais dispositivos conectados à rede chegando, atualmente, a 19,4 bilhões de dispositivos conectados, segundo a IoT Analytics. Salienta-se que, empresas especializadas em disponibilizar essa tecnologia através de serviços IoT, *software* e nuvem têm crescido muito, como é o caso da

Amazon AWS e Microsoft Azure que expandiram 49% e 93% as suas receitas, segundo a IoT Analytics (2018).

Apesar do número de dispositivos conectados ser exorbitante, esse número continua crescendo podendo chegar a 34,2 bilhões em 2025, como mostra a pesquisa na Figura 1.

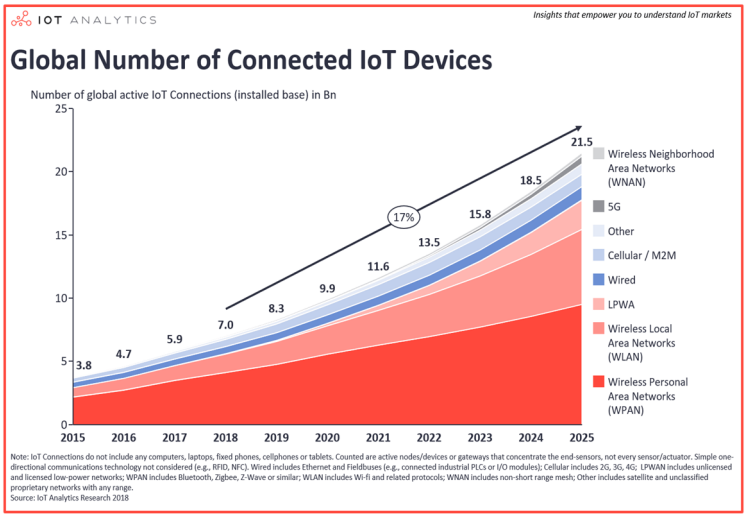
Figura 1 – Números de dispositivos conectados de 2015 a 2025



Fonte: (IOT ANALYTICS, 2018)

Segundo a IoT Analytics (2018), o número de dispositivos IoT conectados terá um salto de 10 bilhões de dispositivos até 2020 e 22 bilhões até 2025 como demonstrado na Figura 2. Sendo que, esses números referem-se apenas aos dispositivos novos, não levando em consideração os que já foram comprados antes.

Figura 2 – Números de novos dispositivos IoT conectados de 2015 a 2025



Fonte: (IOT ANALYTICS, 2018)

Nesse sentido, é possível averiguar que esse mercado é uma tendência e está em pleno crescimento, contribuindo com isso para evolução das tecnologias e principalmente da sociedade que tem uma melhora significativa no estilo de vida quando esses dispositivos são empregados.

2.3 MICROCONTROLADOR

Com o advento dos microcontroladores, a construção de máquinas muito mais complexas e compactas se tornou possível, visto que necessitavam de um conjunto muito maior de dispositivos e circuitos que exerciam funções diferentes, o que tornava muito complexo e robusto o controle de uma máquina, dificultando a programação do sistema.

Um microcontrolador é um pequeno chip formado por circuitos que têm a capacidade de controlar um conjunto de circuitos, funcionando assim como o "cérebro", tendo também a capacidade de guardar informações, logo, tendo memória (voláteis e não voláteis) a qual permite que guardem comandos para serem programados de forma que exerçam determinadas funções de controle. Dessa forma se constituem como um circuito integrado, tal qual os reguladores de tensão e amplificadores, mas tem características bem diferentes que o tornam únicos, as quais serão expostas a seguir.

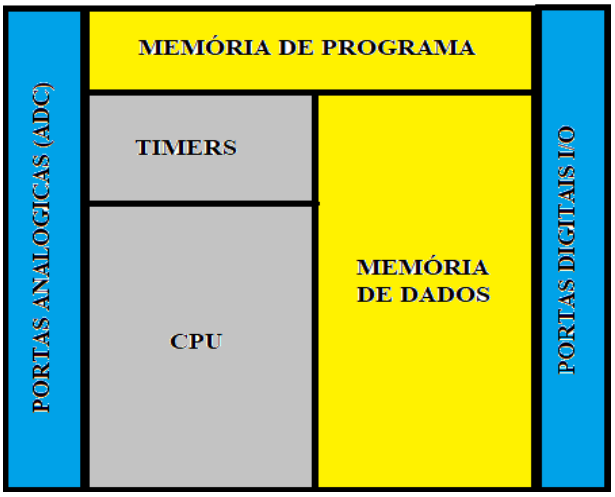
Os microcontroladores tem acoplado a ele uma unidade central de processamento ou CPU (*Central Processing Unit*), que é um processador capaz de executar operações lógicas e aritméticas executando cálculos (por exemplo: a distância de um objeto através de um sensor ultrassonoro), ele é a parte inteligente do microcontrolador, mas tem a limitação de entender somente a linguagem que foi utilizada para sua programação.

Outra característica é a memória, que se divide em dois tipos: a memória dos dados, que armazena as informações capturadas pelo microcontrolador através de leituras de sensores, e a memória *Flash*, que é a memória do programa a qual armazena os dados de maneira "permanente", ou seja, mesmo que o microcontrolador seja desligado o dado estará salvo, nela contém o que o controlador deve fazer, também podendo ser reprogramada.

Outro elemento importante no controlador são os *Timers*, que administram o tempo de processamento do controlador. O tempo de processamento é muito rápido, então através dos *timers* e com auxílio do *clock* esse tempo pode ser mais lento.

Para finalizar, as portas de analógicas que possuem conversores digitais (ADC) e as portas de entrada e saída digitais (I/O). Os componentes apresentados se encontram a seguir, demonstrados na Figura 3.

Figura 3 – Esquema de componentes do microcontrolador



Fonte: Autoria própria

Algo que vale destacar é a aplicação desses microcontroladores, os quais têm uma versatilidade muito grande, sendo usado em diversos eletrodomésticos, controles, relógios, impressoras, em máquinas industriais e também presentes na famosa placa Arduino, que é muito usada em projetos de robótica e automação industrial.

Alguns projetos podem ser citados, tais como, o de controle de iluminação do ambiente, que utiliza um Arduino e um Ethernet Shield, possibilitando, através de um celular, tablet ou computador alterar o estado de iluminação do local. Outro projeto bastante interessante, é o de controle de acesso com RFID, que visa substituir as chaves de um local por *tags* RFID, possibilitando assim acesso aos locais se utilizando das mesmas, de forma que ao serem aproximadas de um leitor que estar conectado a um Arduino, este verifica se o código está cadastrado, e, em caso positivo, efetua a abertura da porta. O último projeto vem a ser algo interessante para pessoas que têm jardins e as vezes se esquecem do processo de irrigação, ele funciona com um Arduino, sensores de umidade, relé, uma bomba de água e *jumpers*. Operando da seguinte maneira, ele faz a verificação da umidade do solo se utilizando dos sensores para isso, depois se for averiguado que o solo está seco o Arduino aciona a bomba de água, regando-as na medida certa.

2.4 WI-FI (IEEE 802.11)

Nos primórdios das Redes de computadores, todos os computadores eram conectados por cabos, podendo ser ponto a ponto ou multipontos, mas isso gerava uma série de problemas em termo de mobilidade, por não poder mover os aparelhos além dos limites dos cabos, tendo também o problema estético por esses terem que passar por diversos lugares para conectar os

computadores e por último o problema de construções civis terem de fazer planejamentos adequados segundo a norma da NBR 14565.

As limitações das redes de computadores com fio, motivou a procura de novas tecnologias que proporcionassem mais mobilidade. Essa tecnologia segundo ADAM ENGST E GLENN FLEISHMAN (2005) começou no Havaí em 1971, quando conseguiram conectar computadores de 4 ilhas. Mas, essa tecnologia só veio a se tornar comercial em 2002 quando IEEE (*Institute of Eletrical and Eletronic Engineers*) lançou o IEEE 802.11a (54 Mbps e 5 GHz) tendo no mesmo ano o lançamento do IEEE 802.11g (54 Mbps e 2,4 GHz), tendo a taxa de transferência igual à do 802.11a e a frequência semelhante ao IEEE 802.11b. (ENGST & FLEISHMAN, 2005).

Vale destacar a criação do consórcio Wireless Ethernet Compatibility Alliance (WECA), que foi uma fundação criada por empresas (3Com, Nokia, Alcatel-Lucent e Symbol Technologies) para padronizar as redes sem fio, onde mais tarde novas empresas se uniram ao consórcio, e em 2003 mudou-se o nome para Wi-Fi Alliance.

A tecnologia Wi-Fi (IEEE 802.11) funciona através de ondas de rádio que se propagam através do ambiente podendo chegar a muitos metros, dependendo do Roteador wireless, não necessitando de cabos para propagação do sinal e comunicação das máquinas. Sendo o IEEE 802.11 uma tecnologia que necessitava de padronização, o IEEE definiu que o intervalo de frequência fosse 2,4 GHz a 2,4835 GHz. Essa tecnologia, entretanto, dispõe de vários padrões desenvolvidos ao longo do tempo sendo um deles o IEEE 802.11a, que opera em taxas de transmissão de 6 Mb/s, 9 Mb/s, 12 Mb/s, 18 Mb/s, 24 Mb/s, 36 Mb/s, 48 Mb/s e 54 Mb/s usando a técnica de transmissão OFDM - *Orthogonal Frequency Division Multiplexing*.

O IEEE 802.11b (utiliza a frequência de 2,4 GHz a 2,4835 GHz, tendo a taxa de transmissão de 1 Mb/s, 2 Mb/s, 5,5 Mb/s e 11 Mb/s usando a técnica de transmissão DSSS - *Direct Sequence Spread Spectrum*, O IEEE 802.11g (mesma frequência que 802.11b, e trabalha com taxas de transmissão iguais 802.11a utiliza normalmente a técnica de transmissão OFDM), o IEEE 802.11n (Frequência de 2,4 GHz a 5 GHz podendo se comunicar com todas as tecnologias anteriores, tendo a taxa de transmissão de 300 Mb/s podendo chegar até 600 Mb/s utiliza a técnica de transmissão OFDM em conjunto com a MIMO - *Multiple-Input Multiple-Output,* que é o destaque da tecnologia, o qual expande sua área de cobertura, podendo chegar até 400 metros), por último um padrão e mais atual, o IEEE 802.11ac (funciona na frequência 5 GHz, trabalhando com a taxa de transmissão de até 433 Mb/s de maneira simples, mas podendo chegar até 6 Gb/s e faz uso da técnica MU-MUMO (Multi-User MIMO).

2.5 APLICATIVOS PARA *SMARTPHONES*

Com o advento dos *Smartphones*, novas funcionalidades foram implementadas através de aplicativos que possibilitam a realização de várias tarefas como, GPS, acesso à internet, previsão do tempo, comunicação por *chats*, assistir filmes e séries, pedir uma locomoção, permitir o controle e acessibilidade de funções da casa (ligar luz, abrir janelas e portas) e as funções básicas de um celular, ligações, envio de SMS e tirar fotos em melhor qualidade. A evolução dos *Smartphones* também possibilita a melhoria dos aplicativos por dispor de maiores espaços nos discos, melhor capacidade de processamento e diversas outras melhorias no *hardware* e no sistema operacional do aparelho.

A tecnologia desses aparelhos é bem diversificada, devido aos diversos fabricantes, destacando-se diversas plataformas como, Google Android, Apple IOS e Windows Mobile, cada uma com a sua linguagem própria, sendo elas, Java utilizada para Android, Objective C usada pelo IOS e C# para Windows. Sendo assim, fazer com que um determinado aplicativo alcance uma série de plataformas exige um conhecimento específico daquela linguagem, um custo alto de desenvolvimento e um tempo excessivo para realização do projeto, sem contar a manutenção do mesmo. Mas existem também vantagens de se desenvolver aplicativos nativos, quais sejam: ter acesso APIs (*Application Programming Interface*) de cada plataforma e a funcionalidade do sistema operacional (e-mail, calendário, data e hora, GPS, SMS e outros) além da possibilidade de ser comercializado nas lojas das plataformas.

Ademais, existe outra forma menos trabalhosa, mais econômica e que exige menos tempo do programador para alcançar diversas plataformas: a programação multiplataforma, ela se baseia em ferramentas de desenvolvimento as quais tornam possível a geração de aplicativos multiplataformas que são empacotados como aplicativos nativos tendo acesso a muitos recursos do sistema operacional e permitindo serem disponibilizados em várias plataformas. Mas, existem também desvantagens que cercam essa solução, ela tem o acesso limitado aos recursos nativos do aparelho e depende muito das atualizações realizadas na ferramenta para manutenção e atualização do aplicativo.

Vale destacar que, com a evolução dos *smartphones* um novo mercado nasceu, o mercado dos aplicativos, os quais podem ser comercializados em lojas de diversas plataformas sendo referidas como App Store, o que incentivou programadores a se especializarem nessa área, buscando meios de tornar seu produto (aplicativo) mais atrativo, usual e até mesmo mais divertido (jogos).

3 METODOLOGIA

Esse projeto foi organizado de maneira a ser dividido em 5 etapas, seguem a seguir as etapas realizadas:

- 1ª etapa: pesquisou-se o melhor microcontrolador e o sensor para se utilizar no projeto, considerando o custo x benefício;
- 2ª etapa: conexão fisica todos os componentes (servo motor, sensor reflexivo e ESP8266);
- 3ª etapa: programação do microcontrolador (WeMos D1 R1), para realizar as funções de mover o servo motor, capturar os dados do sensor e criar uma página HTML;
- 4 ª etapa: criou-se um aplicativo para *Android* que será manuseado pelo usuário através do celular;
- 5ª etapa: testar os resultados finais;

Logo no início do projeto foi feito um estudo sobre qual microcontrolador seria utilizado, a priori, pretendia-se utilizar o microcontrolador mais comum o Arduino, o Uno Rev 3 por ter ele à disposição. No entanto, teria que se usar acoplado a ele um módulo de Ethernet Shield W5100 para Arduino que possibilitaria a comunicação com a rede. Mas, isso significava uma limitação física, já que implicaria na conexão de um cabo UTP (*Unshielded Twisted Pair*) bem longo para que o Arduino Uno Rey 3 ficasse no estacionamento. Por essa razão, optou-se pela utilização de uma maneira mais prática e com uma limitação física bem menor, a placa WeMos D1 R1 WiFi ESP8266, que por sua vez pode se conectar à rede pelo Wi-Fi, dando mais mobilidade e facilitando a implementação.

Tendo escolhido a placa passou-se para pesquisa de que sensor seria utilizado, pensou-se primeiro no módulo sensor de movimento PIR, mas depois verificou-se que ele ativaria logo após o carro chegar e desativaria em seguida, o que seria um problema, pois ainda haveria um carro na vaga. Então pensou-se no sensor ultrassônico HC-SR04, por ser mais rentável e tê-lo a disposição, porém após alguns testes verificou-se que o sensor possuía uma limitação de distância, funcionando somente com uma distância muito curta e uma interferência considerável, o que poderia ser um problema caso o carro estacionasse longe da vaga. Após mais algumas pesquisas, encontrou-se um sensor que se encaixava nos requisitos, o sensor Óptico Reflexivo de Proximidade Infravermelho E18-D80NK, ele oferece uma excelente

precisão e alcança uma grande distância, de até 11 metros, tendo como ponto negativo o custo, pois é relativamente alto.

Escolhidos o microcontrolador e o sensor, resolveu-se por ampliar o trabalho e torná-lo mais completo e econômico. Utilizou-se para isso um servo motor capaz de fazer com que um sensor que cobre uma distância considerável, conseguisse verificar 2 vagas de estacionamento ao mesmo tempo. Dessa forma, o projeto se tornou mais econômico e comercialmente aplicável, visto que a soma desses materiais chegou a um montante de apenas R$ 161 reais.

Para realização da segunda etapa, foram feitos estudos nos *datasheets* dos materiais para que se pudesse trabalhar com eles e através do conhecimento adquirido verificou-se, primeiramente, uma diferença na pinagem da placa WeMos D1 R1 WiFi ESP8266 para o Arduino Uno Rey 3, sendo que o primeiro tem uma pinagem diferente da que está escrita na sua placa. Passada essa etapa, foram realizadas conexões, através de fios de cobre, conectando todos os dispositivos envolvidos no processo, feito isso, foi realizada a organização dos fios para se ter um controle de suas funções utilizando para isso *protoboard*. Por último, foi utilizada uma caixa de MDF para fixar o motor e aderiu-se a esse o sensor, através de uma fita Hellerman, fazendo com que o mesmo acompanhe o movimento do motor.

Tendo concluída a segunda etapa se passou para terceira, que é a programação da placa WeMos D1 R1 WiFi ESP8266, utilizando para isso a mesma IDE do Arduino, que utiliza como base a linguagem C++, mas tendo que incluir bibliotecas personalizadas para placa (ESP8266WiFi.h, WiFiClient.h, ESP8266WebServer.h, ESP8266mDNS.h) e também para o servo motor (Servo.h), sem deixar de ser uma linguagem bem amigável e fácil de utilizar, onde o programador escreve o código e a IDE do Arduino verifica a sintaxe, enviando depois para a placa, que executa os comandos.

A penúltima etapa, demandou mais tempo, pois foi necessário o uso de uma ferramenta nova (App Inventor 2) para o idealizador deste projeto, necessitando, portanto, de tempo para adaptação e manuseio, entretanto, por se tratar de um instrumento bastante intuitivo para programar, usando blocos, e baseado em linguagem Java script o aprendizado se tornou mais fácil. Em conjunto com essa ferramenta, utilizou-se ainda o Firebase, que é um banco de dados seguro utilizado para armazenar dados dos clientes e permitir o *login* dos mesmos.

A etapa derradeira se dividiu em fases: primeira, verificar se os dispositivos estavam bem conectados e se alimentação estava correta, para que não ocorresse a queima de nenhum componente; segunda, confirmar se o microcontrolador estava realizando os comandos de maneira correta e se os dados impressos por ele estavam de acordo com a realidade; e a terceira

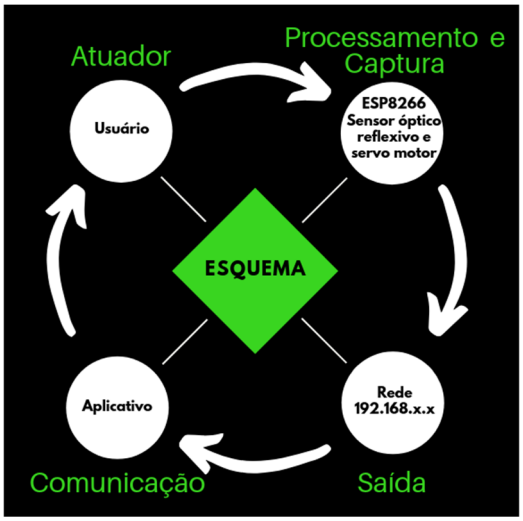
e última, que se deu ao redor do aplicativo, visando verificar, se os dados do cliente estavam sendo armazenados corretamente, se a autenticação e o login funcionavam de maneira adequada e se os dados vindo do microcontrolador através da rede Wi-Fi estavam sendo lidos corretamente pelo aplicativo.

3.1 HARDWARE

O projeto se utiliza de um ESP8266 no estacionamento de uma localidade e por meio desse visa facilitar a vida dos motoristas a encontrar de maneira mais rápida as vagas para estacionar. Seu funcionamento é pautado no controle de um servo motor, o girando no sentido horário e anti-horário de 25° a 145° de maneira intermitente, com intervalos de 3 segundos aproximadamente, esses movimentos são realizados usando uma biblioteca Servo.h.

A utilização do servo, no entanto, para além disso tem o objetivo de auxiliar o sensor óptico reflexivo na captura de informações de objetos (automóveis) em mais de uma vaga de estacionamento, onde este, através do envio e recepção de raios infravermelhos, consegue identificar a presença de um objeto naquele local, e, com auxílio dele, o microcontrolador captura a informação e envia para localização na rede sendo um IPv4 pré-estabelecido no ESP8266 e assim permitindo que o aplicativo possa capturar esses dados e tornar o cliente ciente das informações. Os métodos descritos neste capítulo se encontram representados em um esquema na Figura 4, a seguir.

Figura 4 – Esquema do projeto



Fonte: Autoria própria

Foi utilizado nesse trabalho a IDE do Arduino para inserir os comandos no ESP8266 utilizando para isso bibliotecas (ESP8266WiFi.h, WiFiClient.h, ESP8266WebServer.h,

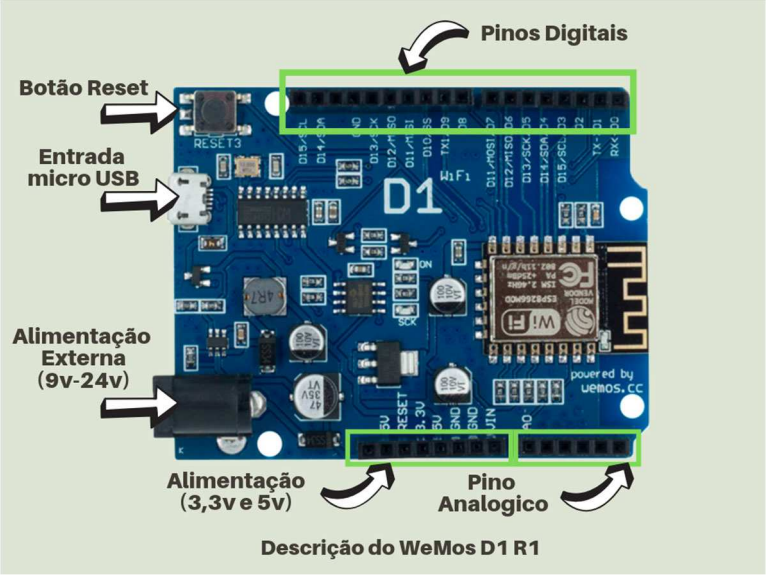
ESP8266mDNS.h) que podem ser baixadas através da IDE, essa, por sua vez, se utiliza da linguagem C++ para desenvolvimento da lógica do microcontrolador. C++ é uma linguagem um pouco antiga, mas ainda sim muito utilizada na atualidade por ser eficiente, ela também é uma linguagem orientada a objeto e tem múltiplas utilidades.

3.1.1 Módulo ESP8266- WeMos D1 R1

O WeMos D1 R1 é um microcontrolador com um SOC (System on chip) com um módulo ESP8266 embutido o qual tem em sua programação protocolos da arquitetura TCP/IP, e também um módulo WiFi em seu hardware, e, se utilizando desse conjunto, o microcontrolador pode ter acesso ao Wi-Fi.

O microcontrolador WeMos D1 R1 é muito utilizado em projetos de IoT, justamente por proporcionar uma conectividade utilizando Wi-Fi dando assim, uma mobilidade maior aos objetos. O WeMos, inclusive, tem uma arquitetura semelhante ao Arduino UNO, podendo usar alguns Shields do Arduino. Ele também tem um nível de programação simples e se utiliza da mesma IDE do Arduino. Na Figura 5 é mostrada a placa WeMos D1 R1. (filipeflop, 2019)

Figura 5 – Descrição da WeMos D1 R1



Fonte: MASTER WALKER(Adaptado), 2019

As características da placa WeMos D1 R1 podem ser encontradas na Tabela 1, a seguir:

Tabela 1 – Características da WeMos D1 R1

Componentes	Características
Processador	ESP8266-12E
Arquitetura	RISC de 32 Bits
Frequência do processador	80 Mhz/160 Mhz
Memória Flash	4 Mb
Flash para instrução	64 Kb
Flash para dados	96 Kb
WiFi nativo padrão	802.11B/G/N
Opera em modo ap	Station ou Ap + Station
Alimentação	5V Dc através do conector micro USB
Alimentação pino Jack	Tesão de 9 a 24V Dc
Conversor USB	Serial CH340G
Pino Analógico	Resolução de 10 Bits
Programação	Via USB ou WiFi(OTA)
Pinos digitais	11 pinos digitais, PWM,12 C e one wire, nível lógico de 3.3V

Fonte: MASTER WALKER (Adaptado), 2019

Algo interessante que vale destacar é que a Placa WeMos D1 R1 proporciona fazer programações se utilizando da OTA (Over The Air), sendo assim, os códigos podem ser enviados por meio do Wi-Fi. (MasterWalker, 2019)

Algo diferente no WeMos D1 R1 do Arduino é a pinagem e sua numeração. Na tabela 2 serão mostradas suas funções e a enumeração da pinagem segundo o ESP8266-12E:

Tabela 2 – Especificações Função e relação dos pinos

PINO WEMOS D1	FUNÇÃO DO PINO	PINO ESP8266-12E
TX	TXD	TXD
RX	RXD	RXD
A0	Entrada analógica	A0
D0	Entrada / saída	GPIO16
D1	(Entrada / saída) / SCL	GPIO5
D2	(Entrada / saída) / SDA	GPIO4
D3	(Entrada / saída) / 10K PULL-UP	GPIO0
D4	(Entrada / saída) / 10K PULL-UP / BUILTIN_LED	GPIO2
D5	(Entrada / saída) / SCK	GPIO14
D6	(Entrada / saída) / MISO	GPIO12
D7	(Entrada / saída) / MOSI	GPIO13
D8	(Entrada / saída) / 10K PULL-DOWN / SS	GPIO15
GND	Ground	GND
5V	VCC 5V	-
3V3	VCC 3.3V	3V3
RST	Reset	RST

Fonte: (MASTER WALKER, 2019)

Tendo em vista que, o Arduino é uma placa mais popular, visando o melhor entendimento, far-se-á uma comparação disposta na Tabela 3.

Tabela 3 – Comparativo entre Arduino UNO x WeMos D1 R1

Fonte: MASTER WALKER (Adaptado), 2019

Logo, pode-se concluir que o WeMos é um microcontrolador potente com uma performance ótima, e, em alguns pontos, superior ao Arduino.

3.1.2 Sensor Óptico Reflexivo de Proximidade Infravermelho E18-D80NK

Um sensor funciona basicamente reagindo a sinais externos e transformando esses sinais em dados que possam ser lidos por pessoas ou por dispositivos eletrônicos, com o sensor óptico reflexivo isso não é diferente, ele emite uma luz infravermelha (transmissor infravermelho) que se for refletida (receptor infravermelho) em certa distância (3cm a 11m) indicará a presença de um objeto naquele local, transmitindo essa informação para o microcontrolador. Essa distância pode ser modificada e ajustada por um potenciômetro que está localizado atrás do sensor (rosca preta), a imagem do sensor pode ser vista na Figura 6 e a descrição dele na Tabela 4.

Figura 6 – Visão do Sensor Óptico Reflexivo de Proximidade Infravermelho

Fonte: (MASTERWALKER, 2019)

Tabela 4 – Descrição do Sensor Óptico Reflexivo de Proximidade Infravermelho

Componentes	Características
Modelo	E18-D80NK
Tensão de Operação	5V Dc
Corrente de operação	25MA
Tempo de resposta	2MS
Ângulo de Detecção	15°
Distancia de Detecção	3 centímetros a 11 metros
Temperatura de operação	-25 a 55° Celsius

Fonte: MASTERWALKER (Adaptado), 2019

3.1.3 Micro Servo 9g SG90

Ele é um motor que se movimenta de 0 a 180 graus em ambos os sentidos, com velocidade máxima 0,12 segundos/60Graus (sem peso), e ângulo programável, tem uma potência alta de saída, porém possui limitações devido ao seu tamanho. Para ser programável utilizando a IDE do Arduino necessita da biblioteca servo.h. Conforme a Figura 7 ele apresenta 3 fios, o fio preto é o fio terra, o vermelho a alimentação e o amarelo onde passa os sinais. Pode-se conferir outras descrições sobre o servo na Tabela 5.

Figura 7 – Visão superior e de perfil do micro servo 9g SG90

Fonte: (FILIPEFLOP, 2019)

Tabela 5 – Descrição técnicas do Miro Servo 9g SG90 Tower Pro

Componentes	Características
Voltagem de operação	3,0 -7,2 V
Ângulo de rotação	180 °
Velocidade	0,12 segundos /60°(4,8V)
Torque	1,2 Kg.CM(4,8V) e 1,6 Kg.Cm (6,0V)
Temperatura de Operação	-30C~+60C
Tipo de engrenagem	Nylon
Temperatura de operação	-25 a 55° Celsius
Tamanho do cabo	245 mm
Dimensões	32 x 30 x 12 mm
Peso	9G

Fonte: FILIPEFLOP (Adaptado), 2018

3.2 SOFTWARE

A ferramenta utilizada para desenvolver o aplicativo é o MIT App inventor, uma ferramenta de código aberto que foi criada pela Google, mas, na atualidade está sendo gerenciada pelo MIT (Instituto de Tecnologia de Massachusetts). Ela tem como característica ser amigável, visto ser uma programação em blocos onde o programador pode mover um bloco com uma função previamente estabelecida e ir unindo a outros blocos, dando forma ao aplicativo, o que, facilita muito a programação.

Ela também funciona na "nuvem", utilizando para isso um navegador, assim permitindo que você crie ou modifique seu aplicativo em qualquer lugar que tenha internet e um computador. MIT App inventor conta, inclusive, com um aplicativo de celular o qual permite que o usuário instale o aplicativo ou teste em tempo real através de um QR code.

Outro fator que vale destacar é a possibilidade de empacotamento do aplicativo criado, dessa maneira permitindo que seja compartilhado com outras pessoas. Vale ressaltar, que é uma ferramenta a qual desenvolve aplicativos nativos somente para Android, trazendo com isso algumas vantagens permitindo que o aplicativo inicialize mais rapidamente e possa utilizar ferramenta da plataforma como (GPS, câmera, sensores e etc).

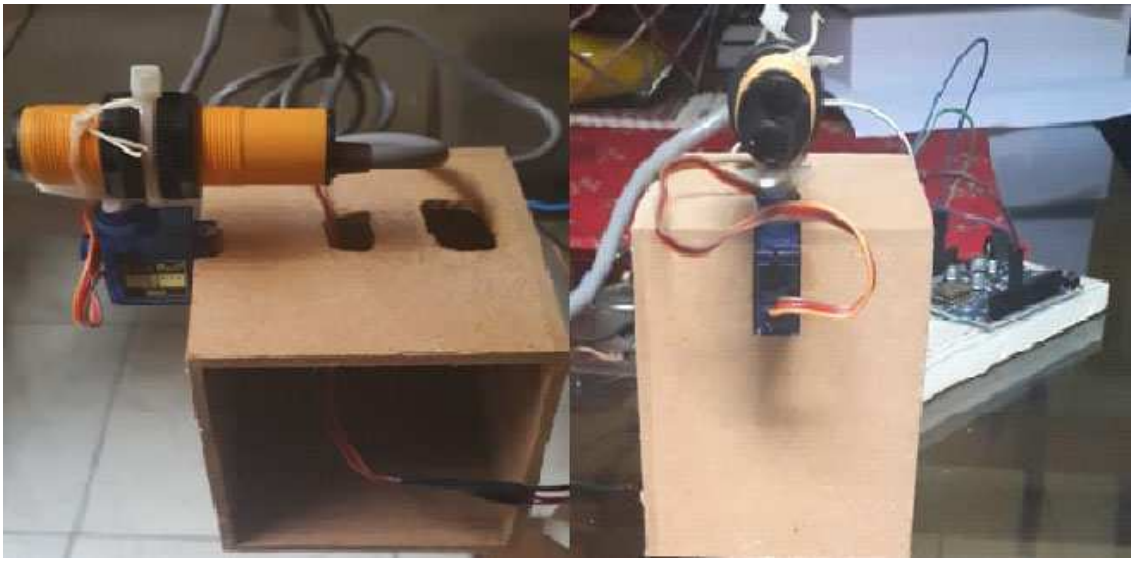
4. DESENVOLVIMENTO DO MODELO PROPOSTO

Este capítulo, apresentará a forma como o projeto foi elaborado, e, as etapas que foram desenvolvidas na realização do projeto, sendo elas, montagem, criação do aplicativo e programação do microcontrolador.

4.1 MONTAGEM

A construção física do projeto se deu com a interligação dos componentes, e a busca por uma base que tivesse um pouco de altura e pudesse proporcionar um equilíbrio para a rotação do motor, nesse sentido, se foi pensado em uma caixa de madeira que pudesse servir como base e também como local para armazenar o microcontrolador, mas, por não haver uma caixa de madeira disponível no momento, utilizou-se uma caixa feita de MDF com medidas de 8 cm por 8 cm. Nessa caixa foram realizados 2 furos, 1 para passagem dos 3 fios do motor e outro para o cabo do sensor. Após a feitura dos furos, foi fixado na caixa, no centro de uma das suas partes laterais, o servo motor, utilizando-se para isso um parafuso, logo após isso plugou-se o sensor óptico reflexivo ao servo com auxílio de um fio de seda e uma fita hellerman para que pudesse acompanhar o giro do motor e assim conseguisse ter em seu raio de atuação duas vagas de carro. A finalização dessa etapa pode ser acompanhada na Figura 8.

Figura 8 – Visão lateral e frontal da montagem finalizada



Fonte: Autoria própria

As implementações que serão realizadas no estacionamento serão, a fixação da caixa em conjunto com o sensor e o motor em um objeto que tenha a altura de 20 cm, de forma a permitir que o sensor possa estar na mesma altura do para-choque de um carro, ele deve se

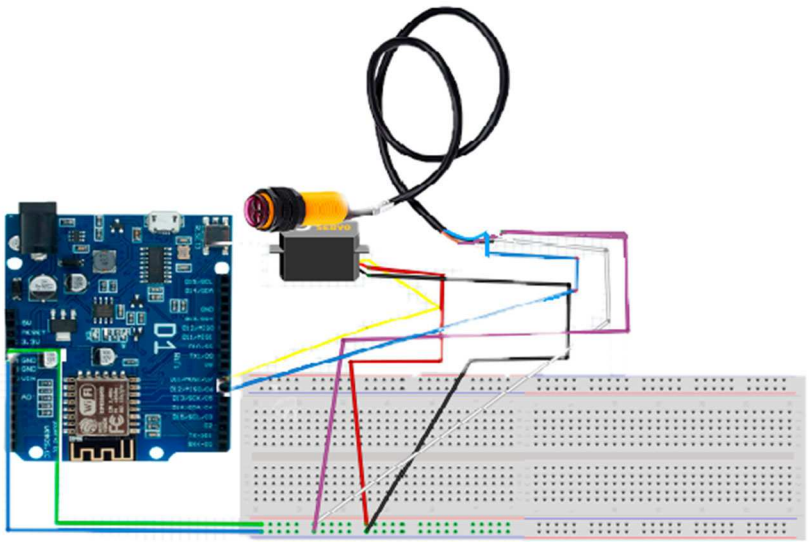
encontrar em uma região intermediária entre duas vagas e uma distância de aproximadamente 1 m, para que, assim, ele possa ter uma boa linha de visão dos automóveis.

4.2 LIGAÇÕES FÍSICAS

As conexões foram feitas por meio de fios de cobre, conectando a placa WeMos D1 R1 ao servo motor e ao sensor óptico reflexivo, levando em consideração a pinagem, a voltagem e a corrente elétrica que os dispositivos suportam. Para conexão se utilizou um protoboard que facilitou a conexão, já que, tanto o servo motor como o sensor reflexivo podem ser alimentados com 5v, então foram colocados em série sendo alimentados pelo WeMos D1 R1.

As conexões realizadas nesse projeto podem ser conferidas na figura 9. Através delas pode se verificar como os fios foram conectados (as cores não são exatamente iguais a do projeto) e a suas pinagens. Destaque-se que, essa figura se trata de um esquema utilizando o programa fritzing. Outro fator importante a ressaltar é que foi utilizado a porta USB mini para alimentação de 5 volts, através de um carregador de celular.

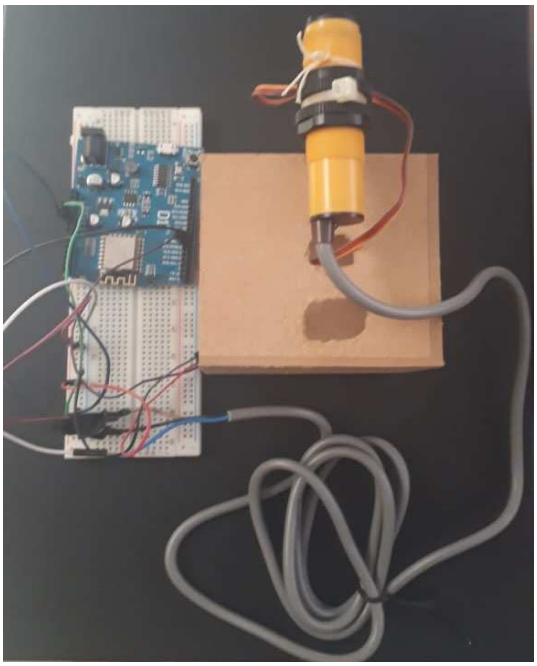
Figura 9 – Ligações físicas do projeto



Fonte: Autoria própria

A Figura 10 que se segue vem ser a foto do projeto das ligações realizadas, com as reais cores e com a montagem finalizada.

Figura 10 – Visão de cima das ligações físicas do projeto



Fonte: Autoria própria

Pode se conferir na Tabela 6, a seguir, a pinagem e a qual dispositivo ela pertence, podendo assim fazer uma correlação.

Tabela 6 – Visão de cima das ligações físicas do projeto

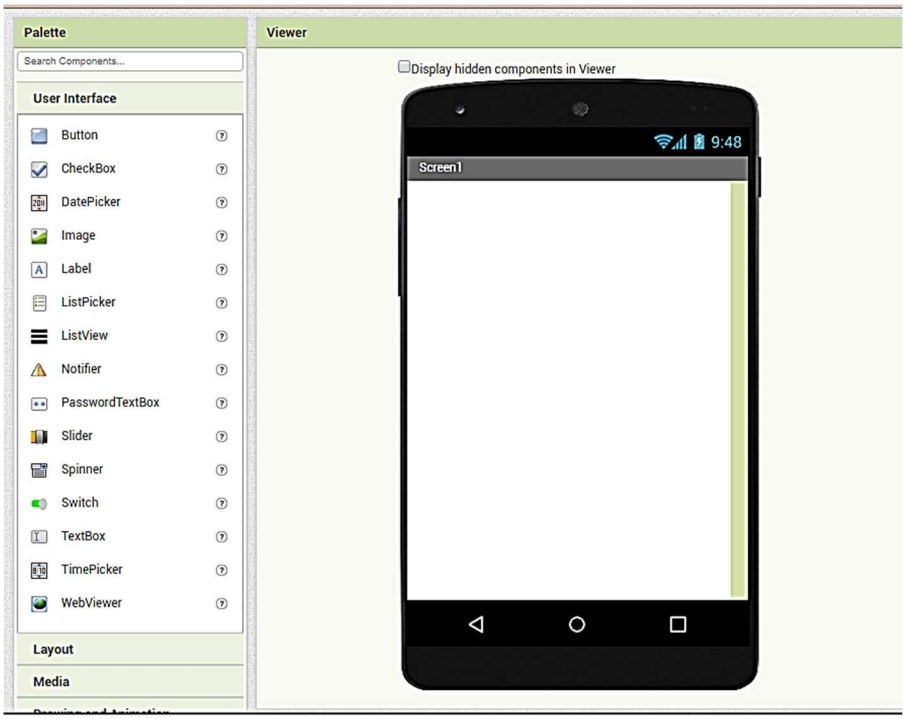
Dispositivos	Pino/WeMos
Sensor óptico reflexivo	13
Servo Motor	12

Fonte: Autoria própria

4.3 DESENVOLVIMENTO DO APLICATIVO CELULAR

Após a familiarização com a ferramenta, como pode ser vista na Figura 11, onde cada uma das telas foi montada de forma a pegar um item e adicionando conforme a necessidade.

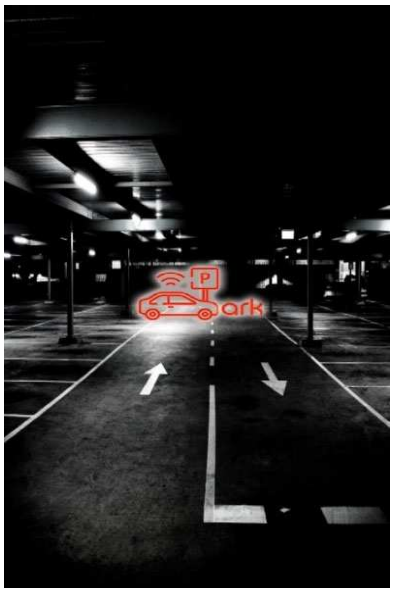
Figura 11 – Interface do MIT App inventor



Fonte: Autoria própria

Logo após isso se iniciou o trabalho com a primeira tela do aplicativo que tinha por objetivo ser algo chamativo, bonito e onde se localizaria a identificação do app através da logo, essa feita com auxílio do Adobe Photoshop CC 2018. Essa tela dispõe de um botão contendo a logo que, ao clicar, te conduz a tela de login. A imagem da tela e da programação dela podem ser vistas na Figura 12.

Figura 12 – Tela inicial do aplicativo



Fonte: Autoria própria

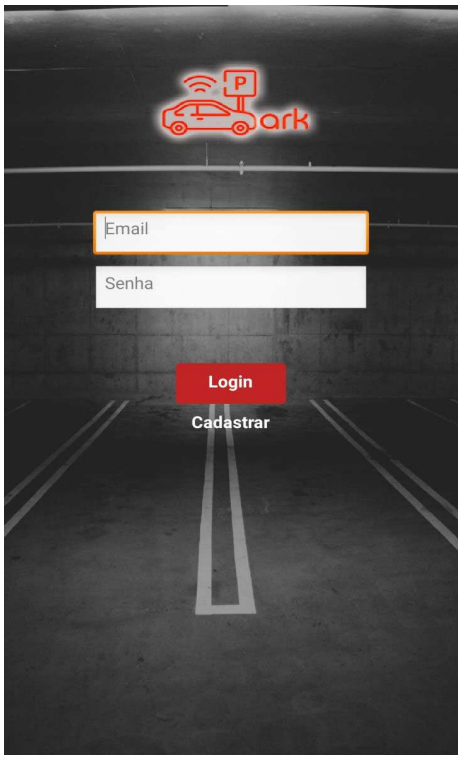
A programação dessa tela se deu por um comando, *open another screenName "Login"*, que realiza a função de abrir a tela de login. E pode ser vista na Figura 13.

Figura 13 – Programação da tela inicial

Fonte: Autoria própria

A tela de login é composta por um botão de cadastro e pelo login, onde o usuário pode se conectar através de e-mail e senha às reais funcionalidades do aplicativo, essa tela pode ser vista na Figura 14. Essa tela funciona da seguinte forma o cliente já previamente cadastrado pode logar ao aplicativo, através de uma base de dados em tempo real, Firebase, que é um banco de dados criado pelo Google, através desse banco que está conectado ao aplicativo é feita autenticação do cliente (cada cliente tendo um ID) e assim podendo logar ou, caso o usuário tenha esquecido a senha, pode ser redefinida através de um e-mail, o usuário digitará o e-mail cadastrado e pedirá a redefinição da senha, logo após ele receberá uma mensagem em seu correio eletrônico que lhe permitirá a troca da senha. Já o botão de cadastro, permite acesso a tela do cadastro a qual será melhor descrita nas linhas que se seguem.

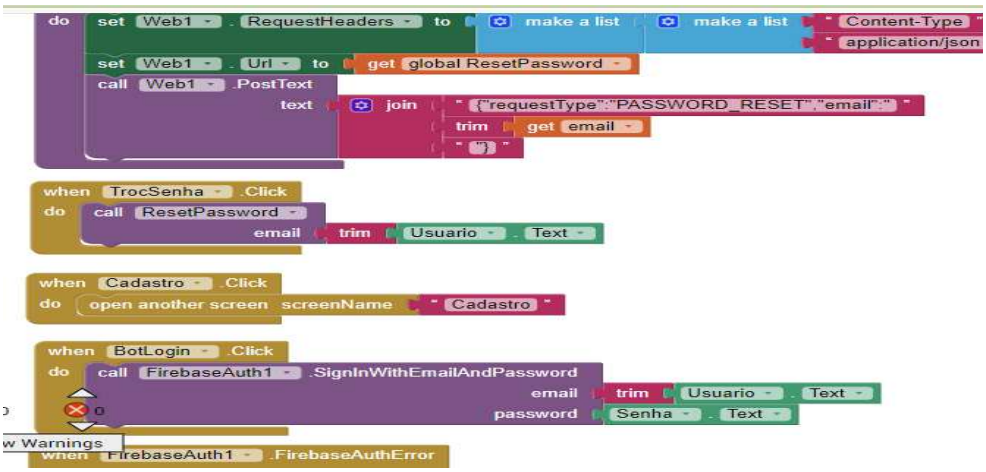
Figura 14 – Tela de login do aplicativo



Fonte: Autoria própria

A programação para se efetuar o login se dar pela busca de dados do banco através de IdToken fornecido pelo banco para o aplicativo poder acessar e modificar seu conteúdo, logo, após isso, é feita uma comparação desses dados com os digitados pelo cliente para fazer login, caso os dados estejam errados aparecerá uma mensagem de erro (esse e-mail não estar cadastrado ou um botão solicitando a troca de senha), caso estejam corretos ele será direcionado para tela de Menu. Pode-se verificar essa programação na Figura 15.

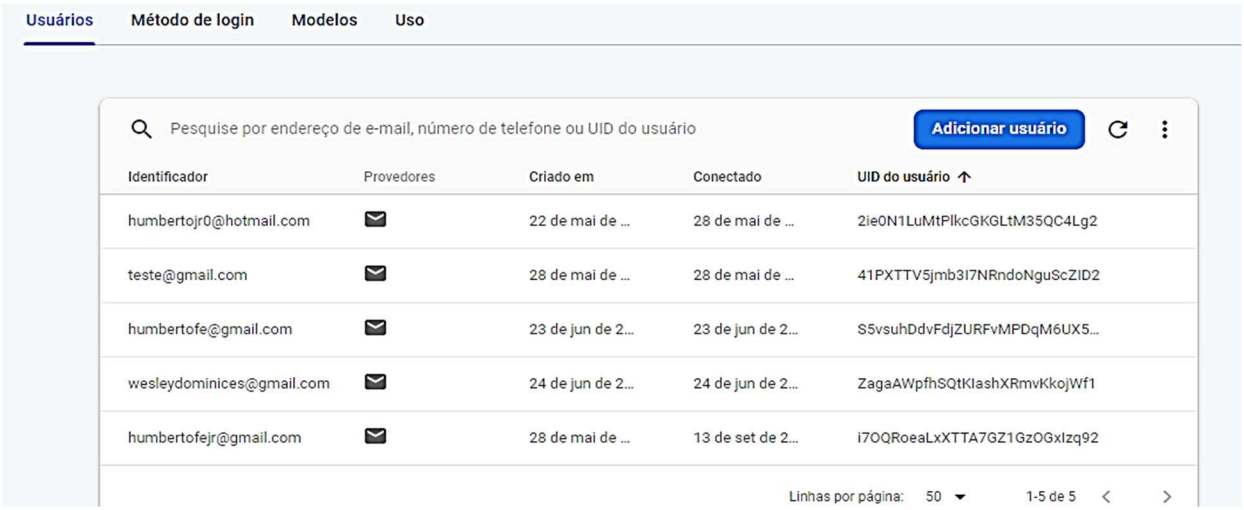
Figura 15 – Programação da tela login



Fonte: Autoria própria

Para evidenciar o que foi descrito anteriormente segue-se a Figuras 16. Que ilustra os dados cadastrado por clientes no app, servindo como meio para autenticação o e-mail, que é verificado pelo banco quando efetua o cadastro e a senha, após isso os dados são armazenados no banco para eventuais logins, sendo anexados a eles um id para a identificação do usuário.

Figura 16 – Autenticação do cliente através do banco



Fonte: Autoria própria

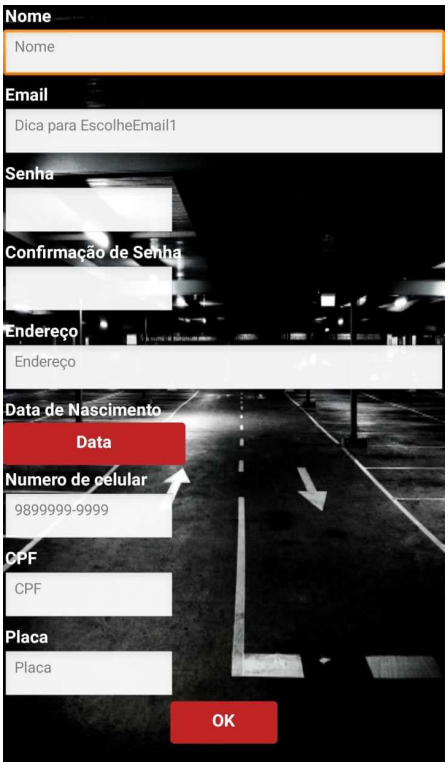
Como já foi falado, um e-mail é enviado pelo Firebase para o usuário do aplicativo previamente cadastrado, mas que veio a se esquecer da senha, esse e-mail contém uma mensagem, perguntando se foi ele que solicitou a mudança de senha e um link para efetuar a ação de troca em caso positivo. Para se ilustrar o que foi falado se tem a Figura 17.

Figura 17 – E-mail de redefinição de senha

Fonte: Autoria própria

A tela de cadastro é composta por dados que o cliente tem obrigatoriedade de preencher para que possa se cadastrar e logar, esses dados incluem nome, e-mail, senha, endereço, data de nascimento, número de celular, CPF e placa. Os dados cadastrados são armazenados no Firebase em conjunto com o ID da conta, depois disso, se não houver nenhum erro de autenticação, aparecerá uma mensagem que apontará o registro com sucesso da sua conta permitindo, portanto, que o usuário efetue o login com e-mail e senha cadastrados. Na Figuras 18 pode-se encontrar a tela de cadastro.

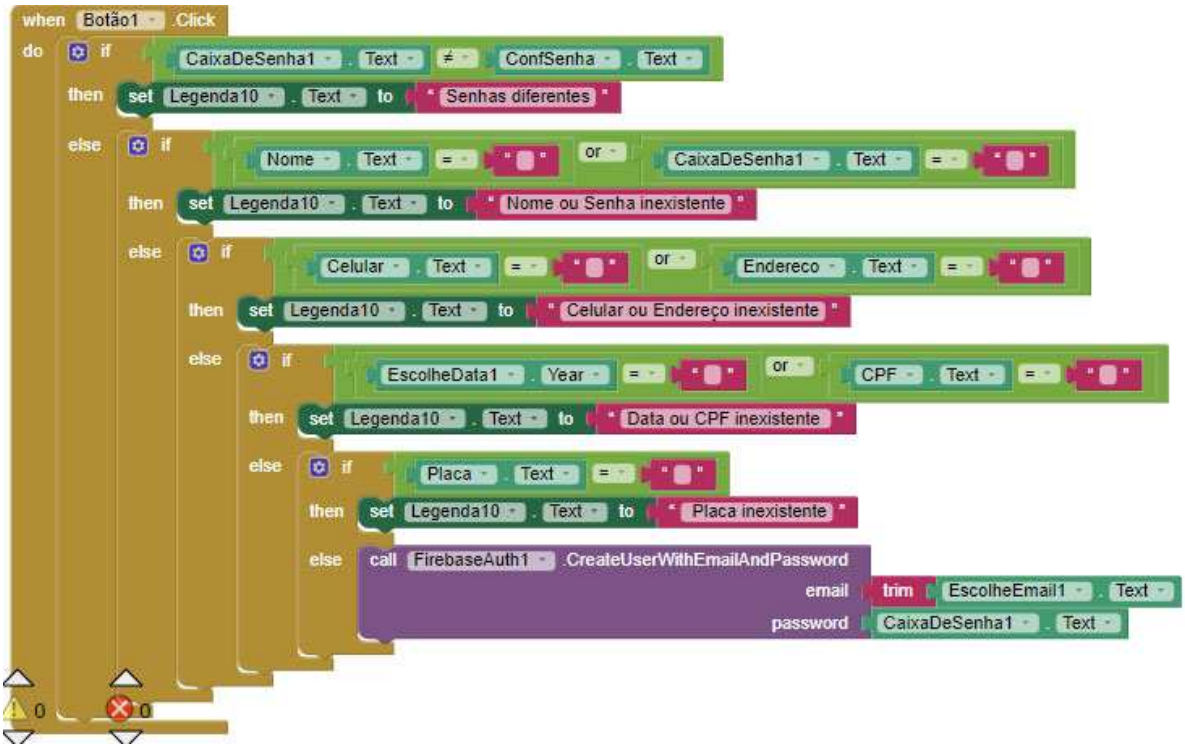
Figura 18 – Tela de cadastro do aplicativo



Fonte: Autoria própria

A programação dessa tela de cadastro se dar pela verificação da senha cadastrada se é igual a registrada em confirmação de senha, se forem diferentes aparecerá uma mensagem de erro, se nenhum dado foi deixado em branco, caso contrário aparecerá uma mensagem indicando qual componente deixou de ser preenchido. Tendo também uma verificação se o e-mail é valido (conter @ e .com) e, senão, nenhum correio eletrônico igual cadastrado no banco. Logo após essas verificações os dados se tornam aptos a serem cadastrados no banco de dados, com um ID para identificação do usuário. Essa programação se encontra na Figura 19.

Figura 19 – Programação de cadastro



Fonte: Autoria própria

Esses dados ficam organizados da seguinte forma no banco de dados, cliente sendo a classe principal e as subclasses sendo os Ids únicos de cada cliente cadastrado, e seus respectivos dados fichados, havendo também um dado “Singln” para verificar se aquela conta já efetuou o login. Essa organização se encontra na Figura 20.

Figura 20 – Dados do Cliente

Fonte: Autoria própria

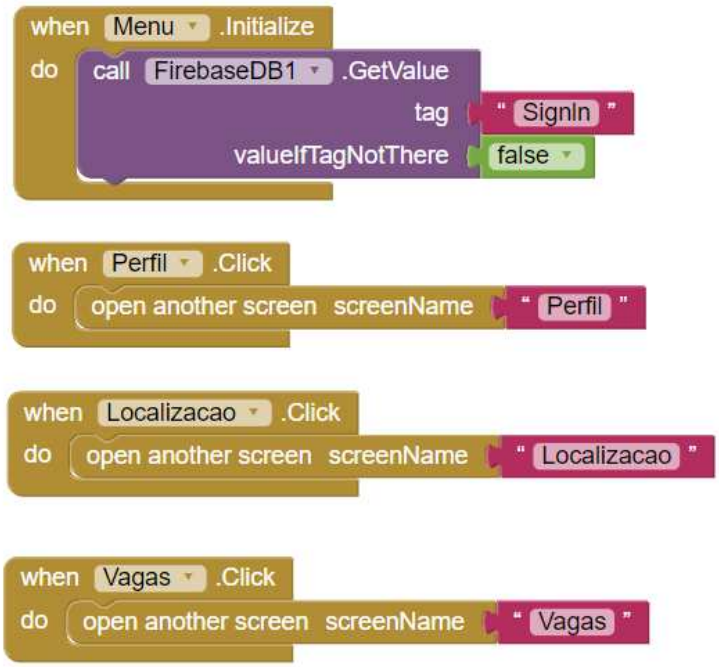
Efetuado o login, o cliente terá acesso a tela do Menu, nela se localizam as três opções de acesso, o perfil do cliente, localização e as vagas, ele pode ter acesso a essas opções clicando em seus botões correspondentes. Esse painel serve basicamente como ponte, interligando o cliente as opções que ele desejar do aplicativo. A Figura 21 apresenta a imagem do Menu.

Figura 21 – Tela de menu do aplicativo

Fonte: Autoria própria

A montagem do programa dessa tela foi realizada da seguinte forma, com a confirmação de login do cliente, armazenando isso em formato booleano no banco, e com botões que levam a tela de perfil, localização e vagas.

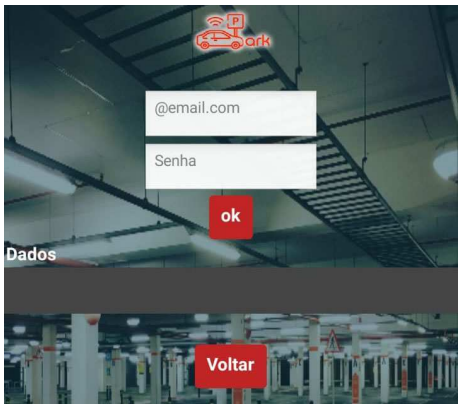
Figura 22 – Programação do menu



Fonte: Autoria própria

Quando o usuário escolhe a opção perfil abre-se uma tela que garante o acesso aos dados, através da digitação de e-mail e senha novamente, por questões de segurança. Logo após isso, o usuário terá acesso aos seus dados cadastrados, através de um Get do banco de dados (Firebase) e assim poderá verificar se estão corretos. Realizada a verificação o cliente poderá retornar ao menu através de um botão localizado abaixo dos dados. As Figuras 23 mostra a tela de perfil.

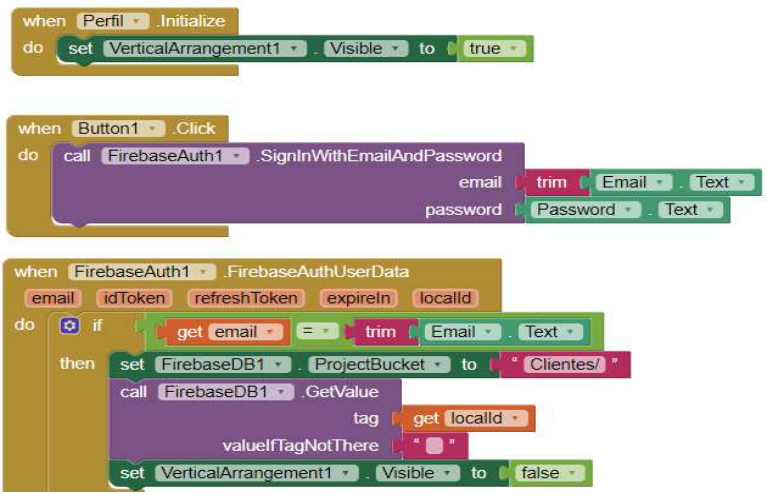
Figura 23 – Tela de perfil do aplicativo



Fonte: Autoria própria

A programação da tela de perfil se dá incialmente pelo login que é realizado novamente pelo usuário que possibilita a seleção dos dados que ele busca, através de um Get realizado no banco utilizando o e-mail digitado para isso, depois disso esses dados são organizados de uma maneira que possibilite o cliente compreender, retirando para isso “;{[“ e os organizando em parágrafos, depois disso os dados são impressos na caixa de texto dados.

Figura 24 – Programação do perfil



Fonte: Autoria própria

Clicando no botão de localização, abrir-se-á uma tela com a localização do estacionamento, que está demarcado como sendo o estacionamento da UEMA. Utilizando-se para isso de uma WebView com endereço do Google Maps e a localização do estacionamento do Centro de Ciências Tecnológicas (CCT), permitindo que o cliente se locomova até as vagas. Vale ressaltar, que se utilizou esse método, posto a inserção da ferramenta do Google Maps no aplicativo demandar o pagamento de uma taxa. Pode-se encontrar na Figura 25 a imagem da tela.

Figura 25 – Tela de localização do aplicativo

Fonte: Autoria própria

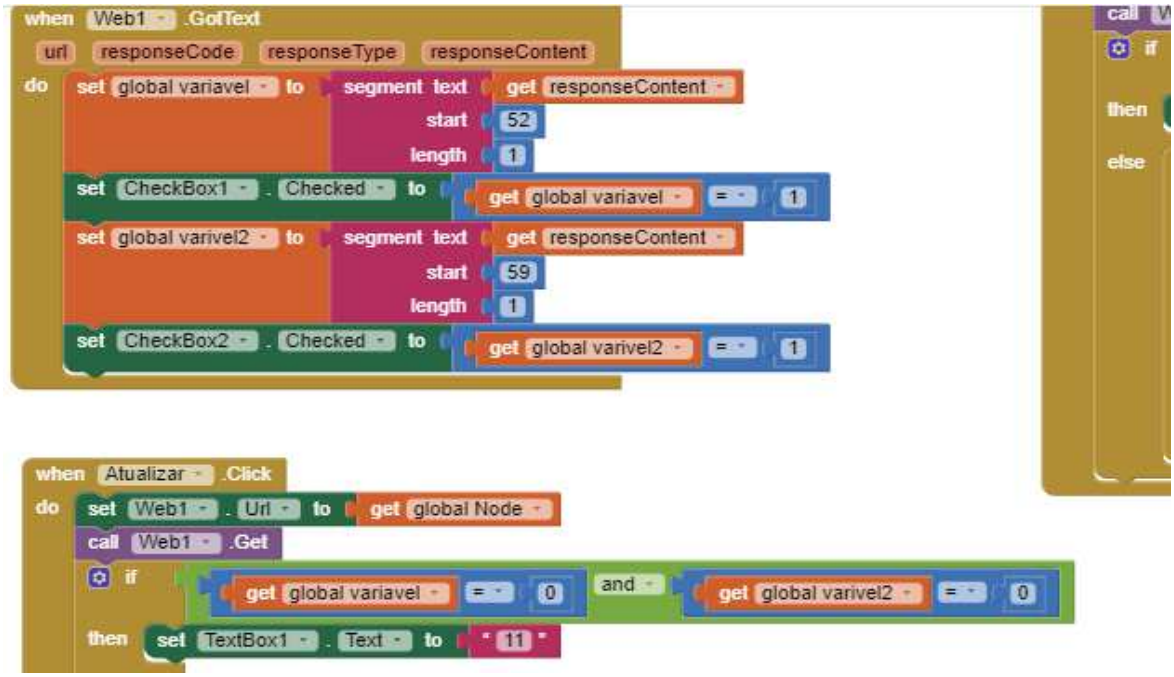
A última tela vem tratar do objetivo do projeto, nela o usuário poderá saber quantas vagas existem no estacionamento e a localização de cada vaga, para que assim ele possa se dirigir de maneira mais rápida ao local. Isso é feito através de informações retiradas de um endereço IPV4 (192.168.0.13), onde o microcontrolador envia as informações verificadas pelo sensor, a partir dessas informações se seleciona o caractere 1 ou 0 que identifica a presença ou a ausência de veículos nas vagas, dispondo de tais dados o aplicativo os converte para um check, caso o dado seja igual a 1, e, se for igual 0, ele permanece vazio. Como sensor só tem autonomia para duas vagas o aplicativo só possui dois espaços para check, sendo os outros meramente ilustrativos. A Figura 26 irá demonstrar o que foi tratado nesse parágrafo.

Figura 26 – Tela de vagas do aplicativo

Fonte: Autoria própria

A programação da última tela se deu da captura de informações de uma página HTML localizada no endereço IPV4, depois dessa captura é ajustado o segmento do texto que interessa ao aplicativo, sendo selecionado através de uma contagem de caracteres, depois disso, como já foi mencionado, é feito uma comparação com esse caractere. Já a contagem de vagas é realizada utilizando *if* e *else*, comparando valores das variáveis globais capturadas, assim definindo o número que será impresso na caixa de texto.

Figura 27 – Programação das vagas

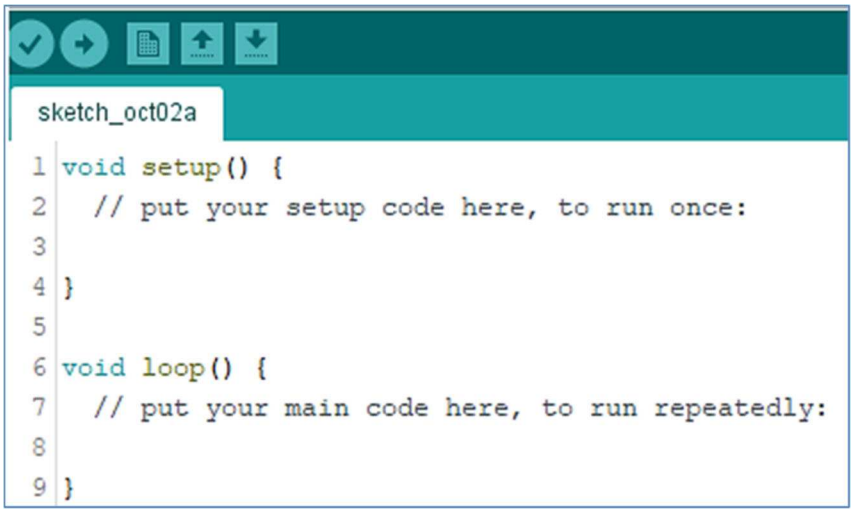


Fonte: Autoria própria

4.4 DESENVOLVIMENTO DO PROGRAMA PARA WEMOS D1 R1

Utilizou-se para elaboração do programa um ambiente de desenvolvimento integrado (IDE) do Arduino, esse software conta com uma facilidade manuseio que possibilita, até mesmo pessoas com pouco conhecimento de programação, a programar seu microcontrolador. Apesar de ser uma IDE do Arduino, ele conta com um suporte muito grande para outros microcontroladores como WeMos D1 R1, e, para facilitar ainda mais ele conta com uma série de exemplos que podem ser modificados para atender a necessidade do usuário, permitindo assim que ele não comece do zero. O ambiente da IDE se utiliza também da linguagem C e C++ para programar microcontrolador e de bibliotecas exclusivas. Na Figura 28 é exibida uma visão da IDE do Arduino.

Figura 28 – Visão da IDE do Arduino



Fonte: Autoria própria

A programação se utilizando da IDE e um cabo USB para comunicação entre o computador e o WeMos D1 R1 só é possível graças a placa que envolve o microcontrolador e tem anexada um bootloader o qual possibilita que o programa seja gravado sem auxílio de um gravador externo.

Depois de instalado o IDE do Arduino, disponibilizado no site do Arduino, se fez necessário ainda adicionar o gerenciador de placa para o ESP8266, e, através dele inicializou-se a programação WeMos D1 R1, utilizando para isso um código exemplo ESP8266WiFiWeb server que auxiliou nas configurações iniciais e nas bibliotecas que deveriam ser utilizadas, tendo suas funções modificadas de acordo com objetivo do projeto.

Nas configurações iniciais se ajustou a rede que foi utilizada e sua senha, para que o ESP8266 tenha acesso a rede e possa utilizá-la para receber e enviar informações. Foi definido também o IP, Gateway e a máscara, para que assim ele não seja dinâmico tendo seu endereço mudado pelo roteador o que dificultaria o acesso a ele pelo aplicativo, já que teria que ser feito sempre alterações do seu endereço conforme a mudança e dessa forma também auxiliando o acesso externo a rede. Por último se definiu a porta do servidor que se localiza na 80, e a utilização do servo. Isso pode ser visto na Figura 29.

Figura 29 – Configurações iniciais do ESP8266

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <Servo.h>
#include <String.h>
#define pinSensor 13
#define IP "192.168.0.13"
#define GATEWAY "192.168.0.1"
#define SUBNET "255.255.255.0"
const char* ssid = "Net Virtua casa 5";
const char* password = "hjfcent071509eejk";

ESP8266WebServer server(80);
Servo myservo;

```

Fonte: Autoria própria

Já criação de variáveis, foram definidas as variáveis: “pos”, que é do tipo inteiro e define o ângulo inicial do servo, “V1”, sendo de tipo inteiro e servindo para identificação do teste, e a “m” de tipo inteiro que serve para ser alterada pela função test e impressa pela função test1. Após a definição das variáveis se encontra a inicialização da página 404, caso nenhuma página possa ser encontrada o servidor irá chamar essa página que deixará claro ao cliente que ocorreu um erro no servidor da página. A figura 30 irá demonstrar o que foi explanado no texto.

Figura 30 – Configurações das variáveis e da página 404

```
int pos = 25;
int v1=1;
int m;

// variable to store the servo position

void handleNotFound(){

  String message = "File Not Found\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET)?"GET":"POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i=0; i<server.args(); i++){
    message += " " + server.argName(i) + ": " + server.arg(i) + "\n";
  }
  server.send(404, "text/plain", message);
```

Fonte: Autoria própria

A próxima etapa foi entrar na função “setup” para definir o pino 13 utilizado pelo sensor e que serve de entrada de informações, ou seja, input. Se definindo também o pino 12 para o servo. Designando em seguida a velocidade da comunicação serial inicial que é 115200. Logo após, no “wifi.begin” inicia-se um processo de login com nome e senha da rede e na sequência instaura-se um processo de tentativas de conexão, com intervalos de cinco segundos, imprimindo no serial o nome da rede e o endereço IP definido (192.168.0.13). Na figura 31 é mostrado o que foi explicado no texto.

Figura 31 – Configuração da inicialização do server

```
void setup(void){
  pinMode(pinSensor, INPUT);
  myservo.attach(12);


  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.println("");

  // Wait for connection
 while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Conectando a rede");
  Serial.println(ssid);
```

Fonte: Autoria própria

A função void loop, como o nome já identifica, é uma função que se repete, nela existe a função “For”, que garante a movimentação constante do servo motor, permitindo com isso a realização dos testes. Através dessa função o servo motor iniciará exatamente no ângulo 25º, que foi o valor definido para a variável “pos”, movendo-se em seguida no sentido anti-horário para 145º. No intervalo para o próximo “For” chama-se outra função, a função test 1, responsável por inicializar a página web e a variável do sensor booleano para verificação da presença ou ausência de objetos, imprimindo o nome da vaga 2 e o valor da variável “m”, que está contido na função test, a qual na presença de um objeto, assume o valor de 1, caso não, assumirá o valor de 0. Impressa a página, a variável retornará para o valor 0 até a próxima verificação. O código das funções pode ser visto na Figuras 32.

Figura 32 – Código do void loop

```
void loop(){
  yield();
server.handleClient();
   for (pos = 25; pos <= 145; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);
    delay(10);    // waits 15ms for the servo to reach the position
  }
    test1;
  delay (3000);

   for (pos = 145; pos >= 25; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
  delay(10);
   }
   test(v1);
    delay(3000);
}
void test(int p){
  delay(1000);
    bool sensor = digitalRead(pinSensor);
    if(!sensor){
      m=1;
      }
  else{
    m=0;
    }
  }
```

Fonte: Autoria própria

Depois da verificação da função test1, se inicializa o próximo “For” que executará o sentido inverso (de 145º para 25º). Finalizada essa etapa executa-se a verificação de test da mesma forma que foi feita no test1 sendo esse para vaga 2, e o test para vaga 1. Depois realizada a verificação, utilizando o sensor e imprimido os resultados na página HTML. O código das funções pode ser visto na Figura 33.

Figura 33 – Código das funções test e test1

```
98  void test(int p){
99    delay(1000);
100     bool sensor = digitalRead(pinSensor);
101     if(!sensor){
102       m=1;
103       }
104   else{
105     m=0;
106     }
107   }
108  void testl() {
109   String textoHTML;
110   bool sensor = digitalRead(pinSensor);
111   textoHTML +=("Vaga2=");
112   textoHTML +=(m);
113   m*0;
114   textoHTML += ",";
115    textoHTML += "vagal=";
116   if(!sensor){
117        textoHTML += "1 ";
118   Serial.println("1");}
119   else{
120   Serial.println("0");
121   textoHTML += " 0";}
122   textoHTML += ",";
123
124    server.send(200, "text/html", textoHTML);
125       }
```

Fonte: Autoria própria

5. TESTE DAS CONEXÕES, DO SOFTWARE DO WEMOS D1 R1 E DO APLICATIVO

É uma etapa fundamental para garantir que não ocorra nenhuma queima de aparelhos e que o funcionamento ocorrerá da maneira que foi planejado, sem imprevistos. Com esse objetivo, foram realizados testes das conexões na energização da placa e nas conexões do sensor e do servo motor, sendo elas encontradas na tabela 7.

Os testes feitos com software, foram a criação da página HTML, a conexão com a rede local, a movimentação angular do servo, para saber se supriria as necessidades com o ângulo configurado e se os dados captados e enviados pelo sensor estavam corretos bem como se a ordem estava correta, sendo essa da direita para esquerda. Esses testes podem ser encontrados na tabela 8.

Por último foram realizados testes no aplicativo sendo esses: o cadastro, se estavam sendo feitos com algumas autenticações e de que forma estavam sendo registrados no banco, finalizado esse teste passou-se para o teste de login, onde foram testados se o cliente poderia logar com e-mail e senha cadastrados e se o espaçamento, geralmente concedido pelo corretor ao término da digitação, atrapalharia no login. Em seguida, realizou-se teste quanto a mudança de senha, na hipótese de o usuário ter esquecido, realizando-se testes para constatar se o usuário receberia o e-mail e se através dele poderia ser realizada a modificação.

Após o login fez-se o teste no perfil para verificar se os dados que o cliente solicitava eram de fato os que ele havia cadastrado. Voltando para o menu, fomos para localização para examinar se o local indicado estava correto. Por último passamos a testar registro de vagas, com intuito de confirmar se os dados conseguidos no endereço 192.168.0.13 estavam sendo registrados corretamente no aplicativo e se a contagem de vagas estava sendo feito da maneira correta. Conforme a tabela 9.

Tabela 7 – Teste somente para uma entrada

LOCAL	ENTRADA
Placa(energização)	OK
Servo motor	OK
Sensor óptico reflexivo	OK

Fonte: Autoria própria

Tabela 8 – Teste software do microcontrolador

LOCAL	SAÍDA
Página HTML	OK
Conexão com a rede	OK
Movimento angular	OK
Dados do Sensor	OK
Ordem	OK

Fonte: Autoria própria

Tabela 9 – Teste do aplicativo

Aplicação	SAÍDA
Cadastro	OK
Banco de Dados	OK
Login	OK
Trocar senha	OK
Perfil	OK
Localização	OK
Verificação de Vagas	OK

Fonte: Autoria própria

A Tabela 10 mostra os custos do projeto

Tabela 10 – Custos do Projeto

Descrição do Item	Valor Unitário (R$)	Unidade(s)	Valor Total (R$)
WeMos D1 R1	**56,00**	**1**	**56,00**
Sensor óptico reflexivo	**94,00**	**1**	**94,00**
Servo Motor	**6,00**	**1**	**6,00**
Protoboard	**5,00**	**1**	**5,00**
Valor Total do Projeto (R$)			**161,00**

Fonte: Autoria própria

6. CONCLUSÕES

O projeto foi de suma importância para aquisição de conhecimentos, pois abrange diversas áreas como, automação, telecomunicação e tecnologia da informação, áreas de especialização do curso da engenharia da computação. Além do conhecimento nessas áreas ele proporcionou um conhecimento prático e teórico, no que diz respeito ao teórico, consegue-se citar os conhecimentos adquiridos sobre a internet das coisas (IoT) e microcontroladores, e do prático consegue-se mencionar a criação de aplicativos para smartphones uma vez que nunca havia criado algo semelhante e nem utilizado a ferramenta MIT App Innventor, a programação de um microcontrolador do tipo ESP8266 também gerou um grande aprendizado até por haver um conhecimento que limitava apenas ao Arduino e por último a configuração de uma rede.

Tendo em vista os objetivos traçados no início do projeto pode-se concluir que ele conseguiu atingir as metas e até ultrapassar certas limitações. Também comprovou sua funcionalidade, apesar de ser só um protótipo. Se mostrou ser um projeto rentável economicamente, como pode ser conferido na tabela 10 que os preços de cada componente utilizado, ou seja, ele pode ser usado em estacionamentos dessa maneira com poucas modificações.

Um fator importante a ressaltar é que o projeto não se limitou somente em fazer algo que funcionasse em uma página html o que, de certo ponto de vista, seria algo mais dispendioso para o usuário, todavia, utilizável. Visando facilitar esse processo criamos um aplicativo que é fácil de usar e tendo usabilidade comprovada, fazendo assim essa interconexão entre a página html gerada pelo microcontrolador e o aplicativo, dessa maneira tornando o projeto mais completo.

Apesar do sucesso do projeto houveram algumas dificuldades no seu decorrer que tornaram o desenvolvimento mais longo e dispendioso, como a dificuldade de selecionar um sensor adequado, a programação do microcontrolador que foi cercada de tentativas e erros e por último o aplicativo com e-mails que não eram enviados, mas com os testes realizados puderam ser consertadas.

Logo conclui-se que o projeto se demonstrou funcional e alcançou os objetivos traçados e se for aplicado de forma usual irá facilitar a vida dos motoristas e diminuir o tempo na procura de vagas, fazendo com que haja mais comodidade nesse processo.

6.1 SUGESTÕES PARA TRABALHOS FUTUROS

Para o desenvolvimento de trabalhos futuros são sugeridos os seguintes:

- Listar as melhores vagas e quais vagas são para deficientes;
- Fazer com que o sensor possa ler mais vagas por vez para economizar na compra de sensores;
- Fornecer a possibilidade de pagamentos pelo aplicativo;
- Adesão de mais sensores para cobrir mais vagas;
- Criar uma função no aplicativo para reservas de vagas;
- Disponibilizar o aplicativo na Play Store;
- Disponibilizar o aplicativo para mais plataforma (IOS);
- Investir na segurança do aplicativo;
- Disponibilizar propagandas para ambientes comerciais.

REFERÊNCIAS

AVANTIA. **Como funciona um sistema inteligente de monitoramento para indústria**. Disponível em: http://www.grupoavantia.com.br/sistema-inteligente-monitoramento-industria/. Acesso em: 28 mai. 2019.

BITRIX24. **Programa de gerenciamento de empresas**. Disponível em: https:// www.bitrix24.com.br/uses/programa-de-gerenciamento-de-empresas-gratis.php?gclid= CjwKCAjwmZbpBRAGEiwADrmVXqbJtUdPd1NjmGNiBdThWjIu-NOwsLVIFK-6izHEja4_FfktcR8exBoChUwQAvD_BwE. Acesso em: 30 ago. 2019.

BLUM, Jeremy. **Explorando o Arduino:** Técnicas e Ferramentas para Mágicas de Engenharia. 1. ed. Rio de Janeiro: Editora Alta Books, 2016. p. 19-70.

ENGST, Adam; FLEISHMAN, Glenn. **Kit do Iniciante em Redes Sem Fio:**: O guia prático sobre redes Wi-Fi para Windows e Macintosh. 2. ed. São Paulo: Editora Pearson Makron Books, 2005.

EXAME. **Estacionamentos, os novos vilões da mobilidade urbana**. Disponível em: https://exame.abril.com.br/brasil/estacionamentos-os-novos-viloes-da-mobilidade-urbana/. Acesso em: 3 set. 2019.

EXAME. **Trânsito: número de automóveis dobrou nos últimos 10 anos**. Disponível em: https://exame.abril.com.br/brasil/transito-numero-de-automoveis-dobrou-nos-ultimos-10-anos/. Acesso em: 3 set. 2019.

FILIPEFLOP. **Placa WeMos D1 R1 Wifi ESP8266**. Disponível em: https://www.filipeflop.com/produto/placa-wemos-d1-r1-wifi-esp8266/. Acesso em: 14 ago. 2019.

G4S. **O que você precisa saber sobre estacionamento inteligentes**. Disponível em: https://blog.g4s.com.br/estacionamentos-inteligentes/. Acesso em: 22 ago. 2019.

IBGE. **Frota de veículos**. Disponível em: https://cidades.ibge.gov.br/brasil/ma/sao-luis/pesquisa/22/28120. Acesso em: 30 ago. 2019.

INDIGO. **MOBILIDADE URBANA E ESTACIONAMENTO: ENTENDA A RELAÇÃO**. Disponível em: https://www.parkindigo.com.br/empresas/estacionamento-e-mobilidade-urbana/. Acesso em: 28 ago. 2019.

INFO WESTER. **O que é Wi-Fi (IEEE 802.11)?** Disponível em: https://www.infowester.com/wifi.php. Acesso em: 4 set. 2019.

IOT ANALYTICS. **State of the IoT 2018: Number of IoT devices now at 7B – Market accelerating**. Disponível em: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/. Acesso em: 4 set. 2019.

KARVINEN, Kimmo; KARVINE, Tero. **Primeiros Passos com Sensores**: Perceba o mundo usando eletrônica, Arduino e Raspberry Pi. 1. ed. São Paulo: Editora Novatec, 2014.

MASTERWALKER. **Como usar com Arduino – Sensor de Proximidade Infravermelho E18-D80NK**. Disponível em: https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-sensor-de-proximidade-infravermelho-e18-d80nk/. Acesso em: 5 set. 2019.

MASTERWALKER. **Conhecendo a Wemos D1**. Disponível em: http://blogmasterwalkershop.com.br/embarcados/wemos/conhecendo-a-wemos-d1/. Acesso em: 4 set. 2019.

NEWTONCBRAGA. **O básico sobre os Microcontroladores**. Disponível em: http://www.newtoncbraga.com.br/index.php/eletronica/52-artigos-diversos/13263-obasico-sobre-os-microcontroladores-parte-1-mic139. Acesso em: 16 jul. 2019.

OLIVEIRA, Sérgio De. **Internet das Coisas:** com ESP8266, Arduino e Raspberry Pi. 1. ed. Brasil: Novatec, 2017.

RESEARCH, Lopez. Uma introdução à Internet da Coisas (IoT): Série d e IoT. **O que é Internet das Coisas (IoT)?,** San Francisco, v. 1, n. 1, p. 2-5, nov./2013. Disponível em: <https://www.cisco.com/c/dam/global/pt_br/assets/brand/iot/iot/pdfs/lopez_research_an_introduction_to_iot_102413_final_portuguese.pdf>. Acesso em: 3 set. 2019.

SANTOS, B. P. *et al*. Internet das Coisas: da Teoria à Prática. **Internet das Coisas**, Mato Grosso, v. 1, n. 1, p. 2-4, out./2016. Disponível em: <https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>. Acesso em: 4 set. 2019.

SILVA, Marcelo Moro; SANTOS, M. T. P. Os paradigmas de desenvolvimento de aplicativos para aparelhos celulares. **Os paradigmas de desenvolvimento de aplicativos para aparelhos celulares.**, São Carlos, v. 3, n. 2, p. 162-166, ago./2014.

SILVA, Rodrigo Adamshuk; STEVAN, Sergio Luiz. **Automação e Instrumentação Industrial com Arduino**: Teoria e Projetos. 1. ed. Brasil: Editora Érica Saraiva, 2015.

USINAINFO. **AUTOMAÇÃO RESIDENCIAL ARDUINO: IDEIAS PARA DEIXAR SUA CASA IGUAL A DO HOMEM DE FERRO!** Disponível em: https://www.usinainfo.com.br/blog/automacao-residencial-arduino-ideias-para-deixar-sua-casa-igual-homem-de-ferro/. Acesso em: 27 set. 2019.

APÊNDICE A – Código Fonte Principal do Projeto

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <Servo.h>
#include <String.h>
#define pinSensor 13
#define IP "192.168.0.13"
#define GATEWAY "192.168.0.1"
#define SUBNET "255.255.255.0"
const char* ssid = "Net Virtua casa 5";
const char* password = "hjfcent071509eejk";

ESP8266WebServer server(80);
Servo myservo;

int pos = 25;
int v1=1;
int v2=2;
int m;

// variable to store the servo position

void handleNotFound(){

  String message = "File Not Found\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET)?"GET":"POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i=0; i<server.args(); i++){
    message += " " + server.argName(i) + ": " + server.arg(i) + "\n";
  }
  server.send(404, "text/plain", message);

}

void setup(void){
  pinMode(pinSensor, INPUT);
  myservo.attach(12);

  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.println("");
```

```
  // Wait for connection
 while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  if (MDNS.begin("esp8266")) {
    Serial.println("MDNS responder started");
  }
  server.on("/",test1);
  server.on("/inline", [](){
    server.send(200, "text/plain", "this works as well");
  });

  server.onNotFound(handleNotFound);

  server.begin();
  Serial.println("HTTP server started");
}

void loop(){
  yield();
server.handleClient();
   for (pos = 25; pos <= 145; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);
    delay(10);    // waits 15ms for the servo to reach the position
  }
    test1;
  delay (3000);

   for (pos = 145; pos >= 25; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
  delay(10);
   }
   test(v1);
    delay(3000);
}
void test(int p){
  delay(1000);
    bool sensor = digitalRead(pinSensor);
    if(!sensor){
      m=1;
      }
  else{
```

```
    m=0;
    }
  }

void test1() {
  String textoHTML;
  bool sensor = digitalRead(pinSensor);
  textoHTML +=("Vaga2=");
  textoHTML +=(m);
  m*0;
  textoHTML += ",";
   textoHTML += "vaga1=";
  if(!sensor){
      textoHTML += "1 ";
  Serial.println("1");}
  else{
  Serial.println("0");
  textoHTML += " 0";}
  textoHTML += ",";

   server.send(200, "text/html", textoHTML);
     }
```