UNIVERSIDADE PAULISTA - UNIP

Felipe Guedes

ESTACIONAMENTO INTELIGENTE COM IoT

Limeira

2016

UNIVERSIDADE PAULISTA - UNIP

Felipe Guedes

ESTACIONAMENTO INTELIGENTE COM IoT

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade de Ciências da Computação da UNIP, como requisito parcial à obtenção do grau de Bacharel em Ciências da Computação.

Orientadores: Marcos Gialdi, Antônio Mateus Locci e Fernando Bryan Frizzarin

Limeira

2016

Felipe Guedes

ESTACIONAMENTO INTELIGENTE COM IoT

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade de Ciências da Computação da UNIP, como requisito parcial à obtenção do grau de Bacharel em Ciências da Computação.

Orientadores: Marcos Gialdi, Antônio Mateus Locci e Fernando Bryan Frizzarin

Aprovado em __ de ____ de 201__.

BANCA EXAMINADORA

_____

Prof. Dr. Nome completo

_____

Prof. Me. Nome completo

_____

Prof. Esp. Nome completo

DEDICATÓRIA

Ao professor Antônio Mateus Locci. Companheiro de caminhada ao longo do Curso de Ciências da Computação. Eu posso dizer que a minha formação, inclusive pessoal, não teria sido a mesma sem sua pessoa. Aos meus familiares e amigos que não mediram esforços para que eu chegasse até esta etapa de minha vida.

AGRADECIMENTOS

Agradeço a todos os professores por me proporcionar o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a mim, não somente por terem me ensinado, mas por terem me feito aprender. A palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão os meus eternos agradecimentos.

Life Begins at the end of your comfort zone.

(Neale Donald Walsch)

RESUMO

O Estacionamento Inteligente foi pensado no conforto dos seus usuários e no meio ambiente, pois com este serviço você encontrará uma vaga muito mais fácil com a economia de combustível e consequentemente diminuindo a emissão de poluentes em nosso planeta.

O Estacionamento Inteligente tem o propósito de diminuir o tempo perdido e de muito estresse que passamos ao procurar uma vaga no estacionamento de um Shopping, por exemplo, pois com o auxílio de nossos serviços o usuário poderá previamente reservar uma vaga pagando uma taxa pela reserva em qualquer dispositivo conectado a Internet.

Palavra-Chave: SmartParking. IoT. Estacionamento Inteligente.

ABSTRACT

The…

Key words:

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

IoT	Internet of Things “Internet das Coisas”
DIY	Do It Yourself “Faça Você Mesmo”
OSI	Open Source Initiative
UI	User Interface
UX	User Experience
GPL	General Public Licence
IDE	Integrated Development Enviroment

SUMÁRIO

1. INTRODUÇÃO

Depois de passar muito estresse procurando uma vaga no estacionamento de um shopping em um dia de estreia de um filme, veio em mente que isso era um problema que enfrentamos frequentemente no meio em que vivemos e não nos damos conta de quanto tempo perdemos procurando uma vaga.

Após refletir muito sobre este assunto, resolvi propor uma solução para este problema onde, não mais necessitamos de ficar procurando uma vaga toda vez que formos assistir a um filme por exemplo, pois se podemos reservar o assento que desejamos ao comprar o bilhete do filme pela internet, por que não reservar a vaga do estacionamento junto?

Foi ai que surgiu a ideia do Estacionamento Inteligente, onde podemos previamente escolher e reservar uma vaga no estacionamento de um shopping.

Após achar uma solução que resolveria este problema fui buscar mais informações no âmbito profissional para verificar se esta solução seria possível com as tecnologias que temos disponíveis no mercado. E claramente era possível resolver este problema com a tecnologia disponível no mercado, onde se encaixou perfeitamente a Internet das Coisas.

A Internet das coisas nada mais é que conectar objetos entre si, com usuários e com a rede, com fim de gerar informações a serem usadas nas mais variadas áreas.

Segundo a Cisco “A Internet das Coisas (IoT) está aumentando a conexão entre pessoas e coisas em proporções jamais imaginadas. Os dispositivos conectados superam a população mundial em de 1.5 para 1. O ritmo de adoção da IoT pelo mercado está aumentando cada vez mais” (CISCO, 2016, http://www.cisco.com/c/pt_br/solutions/internet-of-things/overview.html)

A idéia de conectar objetos é discutida desde 1991, quando a conexão TCP/IP e a Internet que conhecemos hoje começou a se popularizar. Bill Joy,

co-fundador da Sun Microsystems, foi quem pensou sobre a conexão de Device para Device (D2D).

E foi em 1999 que Kevin Ashton do MIT propôs o termo “Internet das Coisas” e dez anos depois escreveu o artigo “A Coisa da Internet das Coisas” para o RFID Journal, e a partir dai o termo se popularizou.

Ashton diz que com a falta de tempo em nossas rotinas, precisamos conectar a internet cada vez mais para resolvermos problemas do dia a dia como, por exemplo, extratos bancários e etc.

E com a tecnologia avançando cada dia mais, estamos vivenciando uma nova era da computação, a computação ubíqua.

1.1 COMPUTAÇÃO UBÍQUA

O Pai da computação Ubíqua ou Era da tecnologia calma, é Mark Weiser, que foi diretor da XEROX em 1996 e em um de seus artigos ele diz:

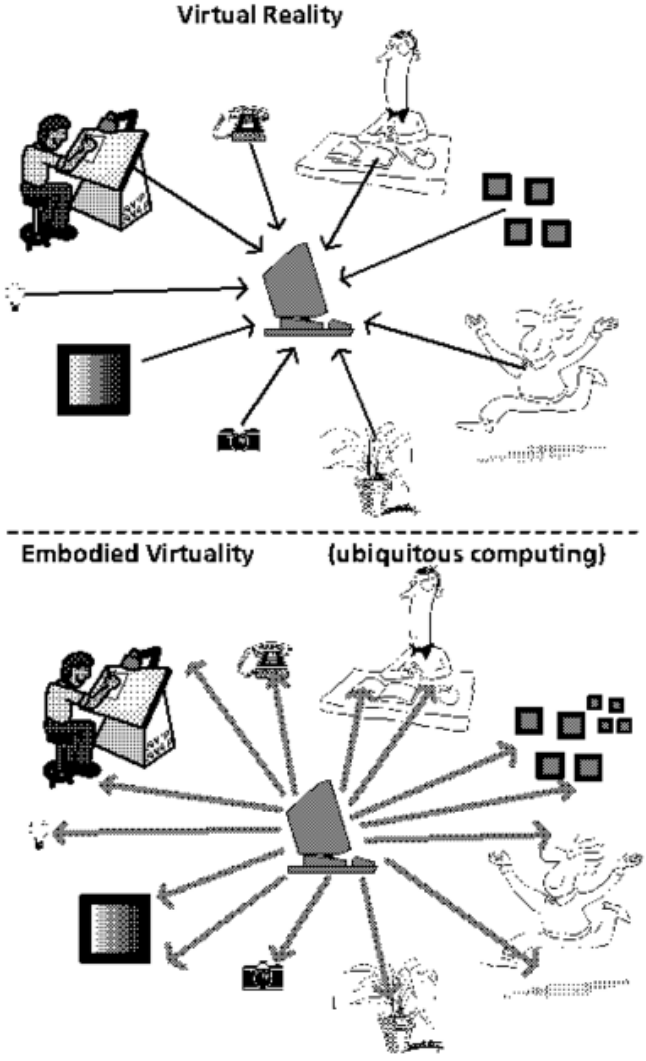
> “A computação ubíqua é a terceira onda da computação, que está apenas começando.
>
> Primeiro tivemos os mainframes, compartilhados por várias pessoas. Estamos na era da computação pessoal, com pessoas e máquinas estranhando umas às outras. A seguir vem a computação ubíqua, a era da tecnologia ‘calma’, quando a tecnologia recua para o plano de fundo de nossas vidas.”

Mark completa:

“As tecnologias mais importantes são aquelas que desaparecem. Elas se integram à vida do dia a dia, ao nosso cotidiano, até serem indistinguíveis dele”.

A computação ubíqua é praticamente o oposto da realidade virtual, pois a realidade virtual nos coloca em um mundo gerado pelo computador, já a computação ubíqua força o computador a viver em nosso mundo, de modo a se tornar muito mais difícil sua integração pois necessitamos de fatores humanos, engenharia, ciências sociais entre outros.

Figura 1 – Realidade Virtual x Computação Ubíqua



Fonte: http://www.ubiq.com/hypertext/weiser/VrvsUbi.gif (2016)

1.2 A INTERNET DAS COISAS APLICADA NO ESTACIONAMENTO

Como ainda estamos na fase de transição da era da computação pessoal para a computação ubíqua, ainda estamos gastando muito tempo em tarefas simples, sendo que com a ajuda da tecnologia poderíamos poupar muito tempo ou até mesmo deixar algumas tarefas para o computador fazer.

O estacionamento inteligente foi pensado no conforto dos seus usuários e no meio ambiente, pois com este serviço você encontrará uma vaga muito mais fácil com a economia de combustível e consequentemente diminuindo a emissão de poluentes em nosso planeta.

Segundo o estudo “Cruising for parking por Donald Shoup”, 30% do trânsito da cidade que experimentamos é causado por pessoas procurando por um lugar para estacionar, com o estacionamento inteligente podemos diminuir este tempo “perdido” e de muito estresse, pois com nossos serviços os usuários poderão previamente reservar uma vaga pagando uma taxa pela reserva em qualquer dispositivo conectado a Internet.

No aplicativo o usuário poderá verificar quais vagas estão livres e efetuar a reserva pelo seu smatrphone, após fazer o check-in de entrada na vaga o sistema começará a contar o tempo que ficou estacionado, para que na hora de fazer o check-out seja gerado o valar a ser pago.

No aplicativo o usuário também conseguirá verificar o tempo permanecido estacionado e o valor a se pagar em tempo real. Também poderá localizar onde seu veículo se encontra estacionado.

2. METODOLOGIA

2.1 METODOLOGIA DE PESQUISA

- Este projeto segue uma linha de pesquisa bibliográfica OpenSource baseada em consultas a materiais publicados em livros e sites de DIY. Com bases nestas pesquisas o projeto teve sua finalidade aplicada em um protótipo.
- Foi realizado uma análise de requisitos que foi realizado através de um estudo de caso sobre a abordagem do tema.

2.2 SOFTWARES UTILIZADOS

Todos os softwares utilizados neste trabalho são de origem OSI.

A OSI foi formada em 1998 como uma organização, de advocacia e de administração em um momento tão importante na história do desenvolvimento colaborativo. [OPENSOURCE 2016]

O movimento começou quando a empresa do navegador Netscape decidiu tornar público seu código fonte para que qualquer pessoa pudesse baixar e contribuir de forma gratuita com o projeto.

Os participantes acreditaram que os argumentos pragmáticos de negócios que motivaram a Netscape a liberar seu código ilustravam uma maneira valiosa de se envolver com potenciais usuários de software e desenvolvedores e convencê-los a criar e melhorar o código-fonte participando de uma comunidade envolvida.[OPENSOURCE 2016]

Eles são:

- Linux Ubuntu 16.04 LTS
- Chromium 49.0.2623.108
- Firefox 46
- Python 2.7.11+

- Mosquitto 1.4.8
- Paho mqtt client
- Apache 2.4.18
- Mysql 5.7.12
- PHP 7
- Bootstrap 3

2.3 HARDWARES UTILIZADOS

2.3.1 ARDUINO

De acordo com Bryan Frizzarin (2016, p.4)

> Arduino é uma plataforma formada por um equipamento, eletrônico e um ambiente de programação integrado (Integrated Development Enviroment - IDE) para prototipagem eletrônica e de software. O equipamento eletrônico da plataforma Arduino consiste em uma placa de circuitos integrados, devidamente equipada com seus componentes eletrônicos e cujo componente central é um microprocessador do tipo AVR da Atmel®.

Escolhi o Arduino para ser a plataforma de prototipagem para este projeto pois o hardware Arduino é uma plataforma de prototipagem de baixo custo e pequena curva de aprendizagem (BRYAN FRIZZARIN, 2016, p.12).

Outra característica importante do Arduino de acordo com Bryan Frizzarin (2016, p.12)

> Também é possível pensar em Arduino pela ótica da Internet das Coisas (ou Internet of Things - IoT) que visa criar equipamentos com capacidade e objetivo de transmitir dados e interagir diretamente com a internet sem a efetiva intervenção humana. Exemplos para esse uso são vastos e passam por estações meteorológicas automáticas, sensores de presença e liberação de acesso até rastreadores veiculares.

2.3.2 SENSORES

De um modo geral os sensores são periféricos que capturam informações do mundo real e são capazes de transformar estas informações em pulsos elétricos.

A utilização de sensores neste projeto é de suma importância, pois são eles que identificam se existe um carro estacionado em uma vaga e envia estas informações para o sistema. O sensoreamento escolhido para este projeto foi o baseado em infravermelho que, comparado aos ultrassônicos, apresenta algumas vantagens, como ser invisível ao olho humano e ter um custo praticamente irrisório (JONES & FLYNN, 1993). No entanto, sua precisão não é garantida (GIBILISCO, 1994), mas como neste projeto nós não precisamos da precisão de distância e sim da informação de se existe algum carro estacionado na vaga, este foi o sensor escolhido para o projeto.

2.3.3 OUTROS HARDWARES

Também foi utilizado um shield, para o Arduino Uno se conectar com a internet chamado Ethernet Shield. E para fazer as ligações foi necessário uma protoboard e alguns jumpers para realizar as devidas conexões dos sensores com a placa.

2.4 LINGUAGENS UTILIZADAS

2.4.1 ARDUINO

A linguagem de programação utilizada para a programação no Arduino é uma variação da Linguagem C, baseada na linguagem Wiring.

2.4.2 PYTHON

Esta linguagem foi utilizada no client paho que funciona como um Middleware para o projeto pois segundo o livro Python para desenvolvedores “O Python tem uma sintaxe clara e concisa que favorece a legibilidade do código-fonte, tornando a linguagem mais produtiva”.

A história do python segundo o livro Python para desenvolvedores (p.15)

> A linguagem foi criada em 1990 por Guido van Rossum, no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda (CWI), e tinha originalmente foco em usuários como físicos e engenheiros. O Python foi concebido a partir de outra linguagem existente na época, chamada ABC.

Escolhi esta linguagem para desenvolver o middleware em nosso protótipo pois:

> A linguagem inclui diversas estruturas de alto nível (listas, dicionários, data/hora, complexos e outras) e uma vasta coleção de módulos prontos para uso, além de frameworks de terceiros que podem ser adicionados. Também inclui recursos encontrados em outras linguagens modernas, tais como geradores, introspecção, persistência, metaclasses e unidades de teste. Multiparadigma, a linguagem suporta programação modular e funcional, além da orientação a objetos. Mesmo os tipos básicos no Python são objetos. A linguagem é interpretada através de bytecode pela máquina virtual Python, tornando o código portável. Com isso é possível compilar aplicações em uma plataforma e rodar em outros sistemas ou executar direto o código-fonte.

E também pelo fato de que o Python é um software de código aberto com licença compatível com a GPL, porém menos restritiva, permitindo que o Python seja inclusive incorporado em produtos proprietários.

2.4.3 PHP

Utilizado junto ao Bootstrap para deixar o FrontEnd dinâmico de nossa aplicação web.

> O PHP (um acrônimo recursivo para PHP: Hypertext Preprocessor) é uma linguagem de script open source de uso geral, muito utilizada, e especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do HTML.
>
> Em vez de muitos comandos para mostrar HTML (como acontece com C ou Perl), as páginas PHP contém HTML em código mesclado que faz "alguma coisa" (neste caso, mostra "Olá, eu sou um script PHP!"). O código PHP é delimitado pelas instruções de processamento (tags) de início e fim <?php e ?> que permitem que você pule para dentro e para fora do "modo PHP".

> O que distingue o PHP de algo como o Javascript no lado do cliente é que o código é executado no servidor, gerando o HTML que é então enviado para o navegador. O navegador recebe os resultados da execução desse script, mas não sabe qual era o código fonte. Você pode inclusive configurar seu servidor web para processar todos os seus arquivos HTML com o PHP, e então não haverá realmente nenhum modo dos usuários descobrirem se você usa essa linguagem ou não.
>
> A melhor coisa em usar o PHP é que ele é extremamente simples para um iniciante, mas oferece muitos recursos para um programador profissional.

2.4.4 BOOTSTRAP

O Bootstrap é um framework que trabalha com HTML, CSS e JS para o desenvolvimento de projetos responsivo e focado para dispositivos móveis na web. O Bootstrap foi utilizado no projeto para aumentar a produtividade no desenvolvimento da interface (UI) de nosso FrontEnd e para dar uma ótima experiência para os usuários (UX) ao utilizar o site.

2.4.4.1 HTML 5

HTML significa Linguagem de Marcação de Hiper Texto. É uma linguagem universal usada na criação de documentos para a Internet. Criada por Tim Berners-Lee nos anos 90. Criada para descrever a estrutura de um documento, esta linguagem permite a formatação de texto, através de marcações (tags) que possuem funções específicas.

2.4.4.2 JAVASCRIPT

O JavaScript é uma linguagem interpretada e foi originalmente implementada como parte dos navegadores web para que scripts pudessem ser executados do lado cliente e interagissem com o usuário sem a necessidade deste script passar pelo servidor, controlando o navegador, realizando comunicação assíncrona e alterando o conteúdo do documento exibido.

É atualmente a principal linguagem para programação do lado do cliente em navegadores web.

2.4.4.3 CSS 3

O CSS surgiu por volta de 1996 possibilitando a formatação de estilos dos sites, deixando mais amigáveis para os usuários.

2.5 BANCO DE DADOS

O banco de dados utilizado foi o Mysql por ser OpenSource e ter uma curva de aprendizado muito interessante, pois possui uma comunidade muito ativa.

2.6 PROTOCOLOS DE COMUNICAÇÃO

2.6.1 HTTP

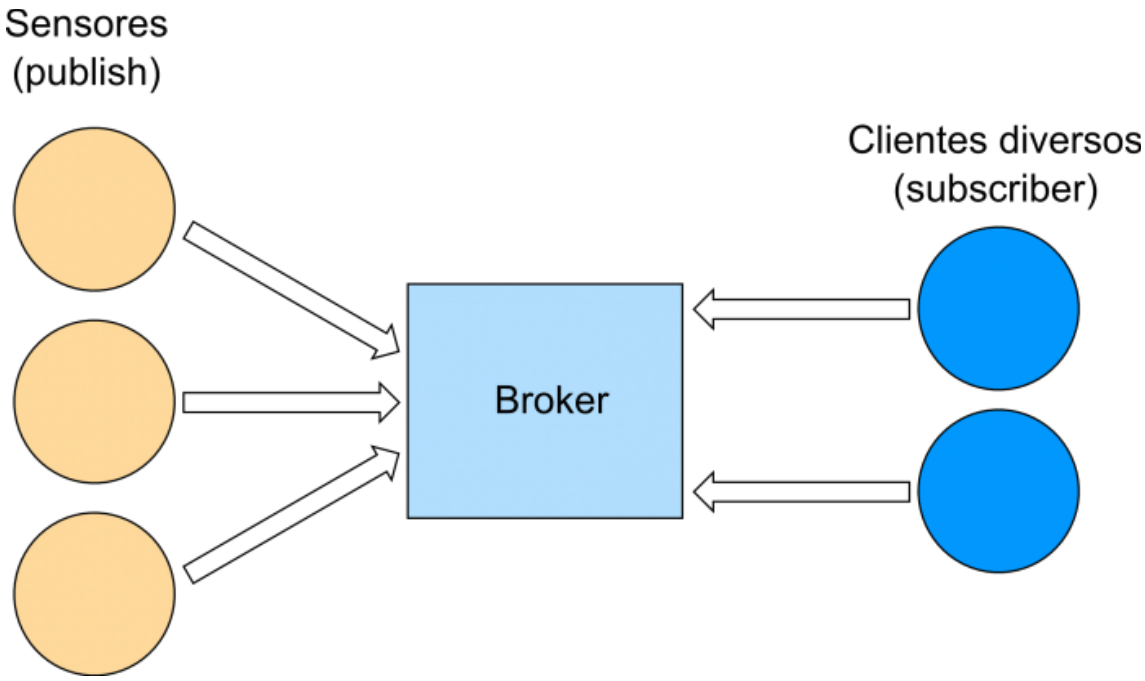
O HTTP é um acrônimo para Hyper Text Transfer Protocol que é o protocolo utilizado no internet para a distribuição e recuperação de informação. A troca de informações entre um browser e um servidor Web é toda feita através deste protocolo, que foi criado especificamente para a Internet.

O HTTP define uma forma de conversação no estilo pedido-resposta entre um cliente e um servidor, e toda a conversação se dá no formato ASCII (texto puro).

2.6.2 MQTT

Um dos protocolos de troca de mensagens para IoT em uso recente é o MQTT (Message Queue Telemetry Transport). Criado pela IBM no final da década de 90, ele é muito leve seguro, pois possui a possibilidade de implementarmos uma camada de segurança além das qualidades de serviço (QOS) e de fácil implementação. Estas características fazem do MQTT um bom candidato para implementações e usos embarcados.

Figura – 2 Diagrama de funcionamento do MQTT



Fonte: https://www.embarcados.com.br/wp-content/uploads/2015/06/mqtt-overview-696x409.png (2016)

2.8 MIDDLEWARE

Como nosso projeto é um sistema distribuído, precisamos da presença de um middleware para abstrair toda a complexidade da comunicação do hardware com o software. Desta forma com a camada do middleware feita em Python, isto permitiu a construção de aplicações independentes do hardware e dos sistemas operacionais.

O Middleware construído fica “escutando” as publicações do hardware que são enviadas através do protocolo MQTT e estas informações são tratadas pelo middleware e enviadas ao banco de dados.

2.7 SERVIDORES

2.7.1 APACHE

O Apache é um servidor web que é encarregado de processar as solicitações HTTP que é o padrão da web atualmente. Quando usamos um navegador de internet para acessar um site, por exemplo, este faz as

solicitações devidas ao servidor Web através do HTTP e então recebe o conteúdo do site.

De acordo com uma pesquisa realizada em 2015 pelo site netcraft o Apache é foi o servidor mais utilizado em 2014, pois ele é OpenSource, robusto, tem uma excelente performance, seguro e tem compatibilidade com diversas plataformas.

3. DESENVOLVIMENTO

3.1 INTRODUÇÃO

Neste capítulo será realizado uma análise da solução proposta para a construção da prototipagem do estacionamento inteligente.

3.2 DESENVOLVIMENTO DA SOLUÇÃO

3.2.1 HARDWARE

O hardware é a base de nosso projeto, pois ele será o responsável por nos enviar as informações das vagas inteligentes de nosso estacionamento.

Para o desenvolvimento de nosso protótipo utilizaremos um Arduino uno para ser o microcontrolador responsável por receber os dados vindos dos sensores infravermelhos da vaga que consiste em nos informar se existe um veículo estacionado ou não em uma vaga, para publicar estas informações para o broker MQTT na rede.

Em nosso protótipo o Arduino está conectado a uma rede através de um shield ethernet para que consiga publicar as informações corretamente para o broker.

3.2.1.1 ANÁLISE DE REQUISITOS

O requisito deste projeto é de desenvolver um hardware que detecte a presença de um veículo estacionado em uma vaga e publique estas informações para o broker MQTT. Com este requisito em mãos, optamos por utilizar um Arduino como controlador e os sensores infravermelhos pois

como já mencionado no item **2.3.2** não precisamos de precisão de distância e o custo beneficio deste tipo de sensor para este projeto é melhor comparado ao um sensor ultrassônico.

3.2.1.2 PROJETO

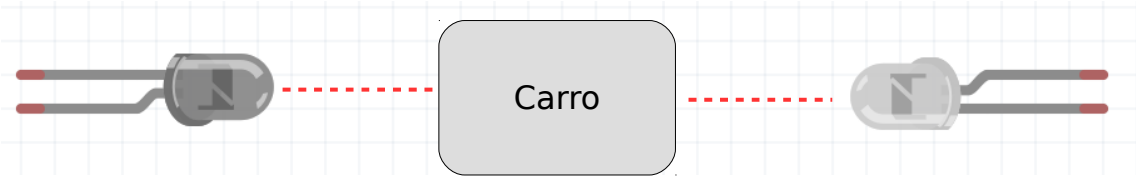
O Projeto do hardware consiste em alinhar o sensor infravermelho emissor junto ao receptor com o intuito de formar uma “linha” invisível para a vaga.

Figura 3 – Esquema dos sensores (Vaga livre)

Fonte: Próprio Autor (2016)

Pois para o sistema identificar que existe um veículo estacionado na vaga esta linha tem que ser “quebrada”, ou seja caso o led receptor infravermelho pare de receber as informações enviadas pelo emissor existe um veículo ocupando aquela vaga, caso contrário a vaga está livre.

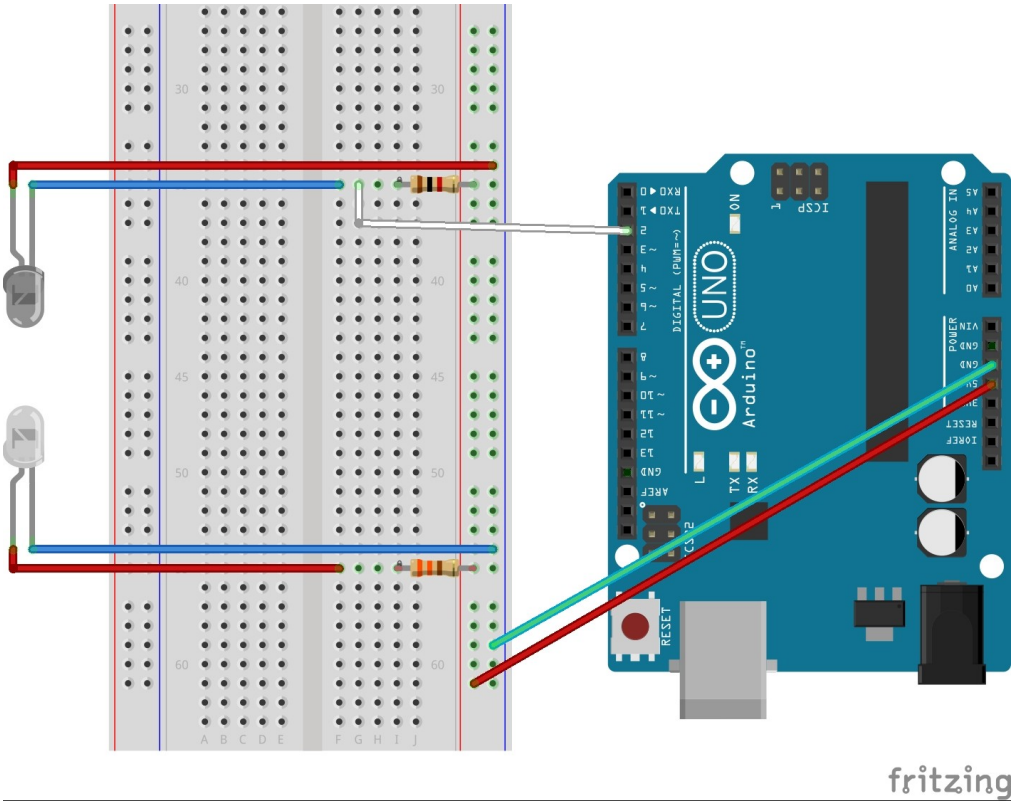
Figura 4 – Esquema dos sensores (Vaga ocupada)



Fonte: Próprio Autor (2016)

Sabendo disto precisamos ligar os sensores da vaga na placa controladora (que será um Arduino para este protótipo) desta maneira.

Figura 5 – Layout de montagem dos sensores com o Arduino



Fonte: Próprio Autor (2016)

Feito isto, o hardware do protótipo já está pronto, faltando somente a programação da placa Arduino para tratar os dados vindo dos sensores instalados.

3.2.1.3 IMPLEMENTAÇÃO

Com nosso hardware devidamente montado, agora precisamos tratar os dados vindos dos sensores infravermelhos, e para isto é necessário abrir a IDE de programação do Arduino e tratar a entrada do pino 2 (dois) como no exemplo abaixo:

Figura 6 – Trecho do código de configuração do Arduino

```
void setup()
{
  pinMode(sensor, INPUT); //Configura o pino 2 como entrada
  Serial.begin(9600);

  mqttClient.setServer(server, 1883);
  mqttClient.setCallback(callback);

  Ethernet.begin(mac, ip);

  //Seta configuraço da vaga ao iniciar o dispositivo
  if(digitalRead(sensor)){
    status_vagal = "1";
  }else{
    status_vagal = "0";
  }

  // Allow the hardware to sort itself out
  delay(1500);
}
```

Fonte: Próprio Autor (2016)

Após isto nosso hardware já é capaz de identificar se exite ou não um veículo estacionado em nossa vaga, agora precisamos publicar esta informação para o broker MQTT, e para isto utilizaremos uma biblioteca chamada PubSubClient para Arduino disponível em: https://github.com/knolleary/pubsubclient.

Com esta biblioteca instalada em sua IDE, na parte do loop do programa do Arduino devemos configurar de quanto em quanto tempo devemos publicar os dados recebidos pelos sensores para nosso broker MQTT, como mostra a figura abaixo:

Figura 7 – Trecho de código de execução do Arduino

```
void loop()
{
  if (!mqttClient.connected()) {
    reconnect();
  }

  // Publish sensor reading every X milliseconds
  if (millis() > (time + 6000)) {
    time = millis();
    b_envia = false;
    if(digitalRead(sensor)){
      if(status_vagal != "1"){
        b_envia = true;
        Serial.println("Mudou o status de 0 para 1");
      }
      status_vagal = "1";
      Serial.println("Vaga Ocupada");
    }
    else{
      if(status_vagal != "0"){
        b_envia = true;
        Serial.println("Mudou o status de 1 para 0");
      }
      status_vagal = "0";
      Serial.println("Vaga Livre");
    }
    if(b_envia){
      mqttClient.publish("estacionamento/vaga/1",status_vagal);
    }
  }
  mqttClient.loop();
}
```

Fonte: Próprio Autor (2016)

Com esta configuração somente vamos publicar no tópico “estacionamento/vaga/1” quando o status da vaga for alterado de “Ocupada” para “Livre” ou de “Livre” para “Ocupada”. Porém o Arduino vai chegar a cada 6 (seis) segundos o status da vaga, pois este é um tempo coerente para eventuais passagens de objetos não identificados na frente do sensor, como uma pessoal por exemplo.

3.2.2 SOFTWARE

O software também é uma parte importante para nosso protótipo, pois sem ele a utilização deste serviço se torna inviável, pois não teremos uma UX agradável ao checar o status de alguma vaga via linha de código.

3.2.2.1 ANÁLISE DE REQUISITOS

Os requisitos para este protótipo é um sistema que possa ser aberto em qualquer dispositivo que contenha um navegador para a internet, e que nos possibilita a reserva e o gerenciamento das vagas de forma simples e fácil.

Para uma experiência mais agradável aos usuários e aos administradores dos estabelecimentos precisamos de uma interface na qual o usuário tenha somente acesso à sua área que são elas as configurações de sua conta, reserva de vagas disponíveis, tempo estacionado, valor atual referente ao tempo estacionado e o histórico de suas reservas e para os administradores são elas o acesso ao dashboard das vagas do estabelecimento e manutenção das vagas além dos acessos de um usuário comum.

3.2.2.2 PROJETO

Sabendo disto utilizaremos HTML 5 para fazer nosso FrontEnd pois ele é interpretado por qualquer navegador e para deixar a experiência do usuário em nosso site mais agradável utilizaremos o Bootstrap 3 com sua padronização e seus estilos montados em CSS 3 e JavaScript. Precisamos também montar o website responsivo com a ajuda do Bootstrap, pois para este projeto nosso website será acessado de qualquer dispositivo com navegador, incluindo tablets, celulares e desktops. Para projetos futuros o acesso via smartphones será via aplicativo.

3.2.2.3 IMPLEMENTAÇÃO

Para começar o projeto precisamos de uma classe de conexão com nosso banco de dados e para isto foi criado esta unit chamada conexao.php que será chamada nas telas para fazer a conexão com o banco para ser gravado os dados.

Figura 8 – Trecho de código de conexão do Front-End

```php
<?php
  /**
   *
   */
  class conexao {

    public static $instance;

    private function __construct() {
      //
    }

    public static function getInstance() {
      if (!isset(self::$instance)) {
        self::$instance = new PDO('mysql:host=localhost;dbname=projectTCC', 'usuario', 'senha',
                                  array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"));
        self::$instance->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        self::$instance->setAttribute(PDO::ATTR_ORACLE_NULLS, PDO::NULL_EMPTY_STRING);
      }
      return self::$instance;
    }
  }
?>
```

Fonte: Próprio Autor (2016)

Com este arquivo pronto agora temos que fazer todas as partes analisadas no item **3.2.2.1**, por exemplo na tela de login será chamado este arquivo em PHP para tratar a tela de login:

Figura 9 – Trecho de código para login no sistema

```php
<?php
  require_once('conexao.php');
  session_start();

  $user = trim($_POST['user_login']);
  $senha = trim($_POST['senha_login']);

  $sql = "SELECT COUNT(*) AS CONT FROM A_USERS WHERE USU_USUARI = :user AND USU_SENHA = :senha";
  $result = conexao::getInstance()->prepare($sql);

  $result->bindValue(":user", $user);
  $result->bindValue(":senha", $senha);
  $result->execute();

  $row = $result->fetch(PDO::FETCH_ASSOC);

  if ($row["CONT"] > 0) {
    $sql1 = "SELECT USU_ADMIN FROM A_USERS WHERE USU_USUARI = :user AND USU_SENHA = :senha";
    $result1 = conexao::getInstance()->prepare($sql1);

    $result1->bindValue(":user", $user);
    $result1->bindValue(":senha", $senha);
    $result1->execute();

    $row1 = $result1->fetch(PDO::FETCH_ASSOC);

    $_SESSION['user'] = $user;
    $_SESSION['senha'] = $senha;
    $_SESSION['admin'] = $row1["USU_ADMIN"];

    header('location:../index.php');
  } else {
    $_SESSION['login_erro'] = 1;

    unset($_SESSION['user']);
    unset($_SESSION['senha']);
    unset($_SESSION['admin']);

    header('location:../login.php');
  }

?>
```

Fonte: Próprio Autor (2016)

Note que estamos fazendo as tratativas do protótipo dentro das variáveis $_SESSION, tanto para saber o usuário logado quanto para retornar os tipos dos possíveis erros que possam ser retornados para o usuário.

3.2.3 MIDDLEWARE

3.2.3.1 PROJETO

Precisamos fazer um client em python que estaja inscrito em todos os tópicos das vagas de nosso estacionamento para que ele trate as publicações feitas pelos Arduino e envie para nosso banco de dados.

3.2.3.2 IMPLEMENTAÇÃO

Utilizamos como base o client feito em python de código aberto chamado Paho e fizemos algumas modificações nele para atender a necessidade deste projeto.

O padrão de publicação das vagas para este protótipo será neste formato: estacionamento/vaga/número_da_vaga

O Middleware será inscrito no tópico “#” que analisa tudo o que está sendo postado nesta rede e ele analisará somente os tópicos que contenha “estacionamento/vaga/” para este protótipo, como nos exemplos abaixo:

Figura 10 – Trecho de código de configuração de conexão do Middleware

```
#MQTT Config
broker = '192.168.1.101'
#broker = '192.168.0.171'
broker_port = 1883
broker_topic = '#'
#mysql config
mysql_server = 'localhost'
mysql_username = 'usuario'
mysql_passwd = 'senha'
mysql_db = 'projectTCC'
```

Fonte: Próprio Autor (2016)

Figura 11 – Trecho de código de execução do middleware

```
45 # Gera a string de conexao ex.: seu host, seu usuario, sua senha e seu db
46 db = MySQLdb.connect(host=mysql_server, user=mysql_username, passwd=mysql_passwd, db=mysql_db)
47 # Posiciona o cursor
48 cursor = db.cursor()
49
50 client = mqtt.Client()
51 client.on_connect = on_connect
52 client.on_message = on_message
53
54 client.connect(broker, broker_port, 60)
55
56 # Blocking call that processes network traffic, dispatches callbacks and
57 # handles reconnecting.
58 # Other loop*() functions are available that give a threaded interface and a
59 # manual interface.
60 client.loop_forever()
61
```

Fonte: Próprio Autor (2016)

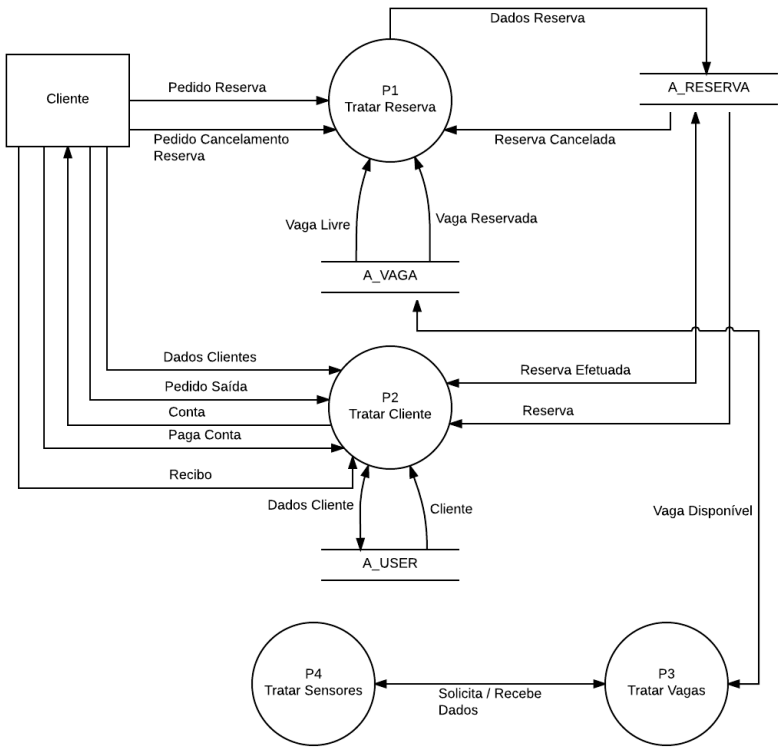
3.2.4 BANCO DE DADOS

3.2.4.1 DFD

O diagrama de fluxo de dados (DFD) é uma representação gráfica do “fluxo” de dados através de um sistema de informação, modelando seus aspectos de processo. Ele fornece apenas uma visão do sistema, a visão estruturada das funções, ou seja, o fluxo dos dados.

O DFD de nível 0 (zero) abaixo mostra quais são os itpos de informações entrará e sairá de nosso sistema:

Figura 12 – Diagrama de fluxo de dados nível 0

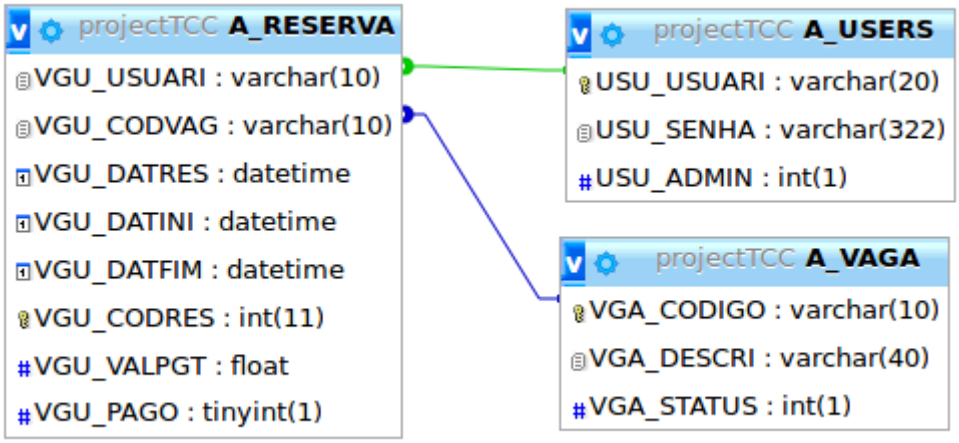


Fonte: Próprio Autor (2016)

3.2.4.3 Modelagem do Banco

Este modelo foi criado com o objetivo de representar a semântica associada aos dados, onde em cima deste diagrama foi montado o banco de dados.

Figura 13 – Modelagem do banco de dados



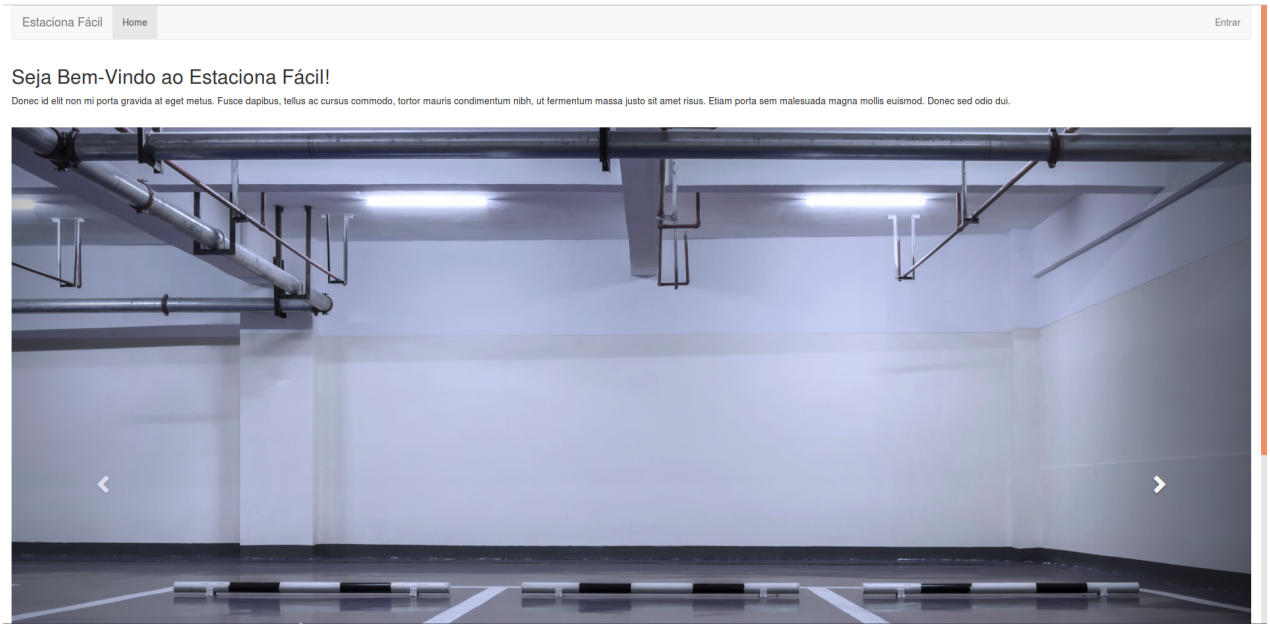
Fonte: Próprio Autor (2016)

4. INTERFACE

4.1 TELA INICIAL

Esta é a tela inicial, onde posteriormente irá ser apresentadas as informações mais uteis para os clientes.

Figura 13 – Tela inicial vista em um desktop



Fonte: Próprio Autor (2016)

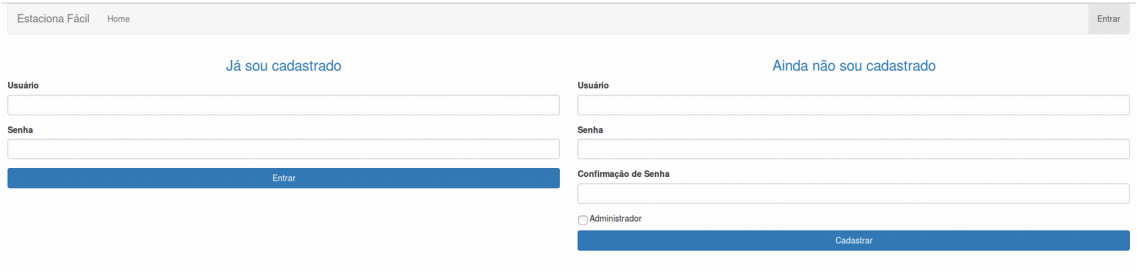
Figura 14 – Tela inicial vista em um celular

Fonte: Próprio Autor (2016)

4.2 TELA DE LOGIN

Esta é a tela de login e de cadastro para novos usuários, o sistema não libera nenhuma função caso o usuário não esteja cadastrado e logado no sistema.

Figura 15 – Tela de login vista em um desktop



Fonte: Próprio Autor (2016)

Figura 16 – Tela de login vista em um celular

Fonte: Próprio Autor (2016)

4.3 TELA MANUTENÇÃO DE VAGAS

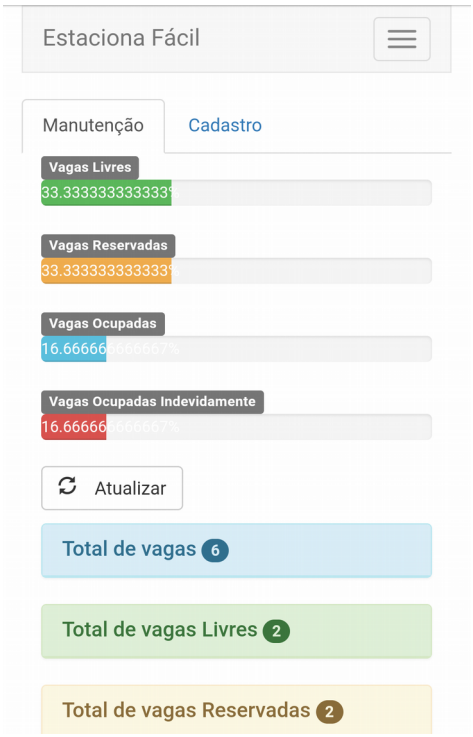
4.3.1 DASHBOARD

Quem tem acesso a esta tela são somente os administradores do sistema, pois nela conseguimos visualizar o estacionamento como um todo, facilitando o gerenciamento das vagas do estacionamento.

Figura 17 – Dashboard visto em um desktop

Fonte: Próprio Autor (2016)

Figura 18 – Dashboard visto em um celular



Fonte: Próprio Autor (2016)

4.3.2 CADASTRO E PESQUISA DAS VAGAS

Nesta aba é onde os administradores dão manutenção nas vagas dos estacionamentos.

Figura 19 – Tela de cadastro das vagas vista em um desktop

Fonte: Próprio Autor (2016)

Figura 20 – Tela de cadastro de vagas vista em um celular

Fonte: Próprio Autor (2016)

4.4 TELA DE MANUTENÇÃO DE RESERVAS

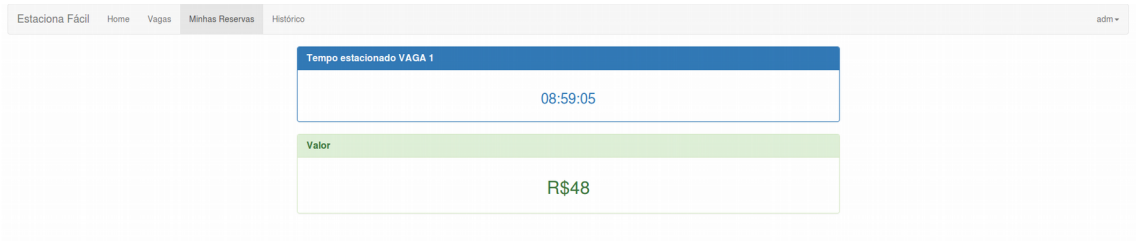
Esta é uma tela dinâmica, onde se o usuário não tiver nenhuma reserva em andamento o sistema nos mostra esta tela, onde o usuário poderá fazer a reserva de alguma vaga que está disponível no momento.

Figura 21 – Tela de manutenção das reservas (vagas livres para serem reservadas) vista em um desktop

Fonte: Próprio Autor (2016)

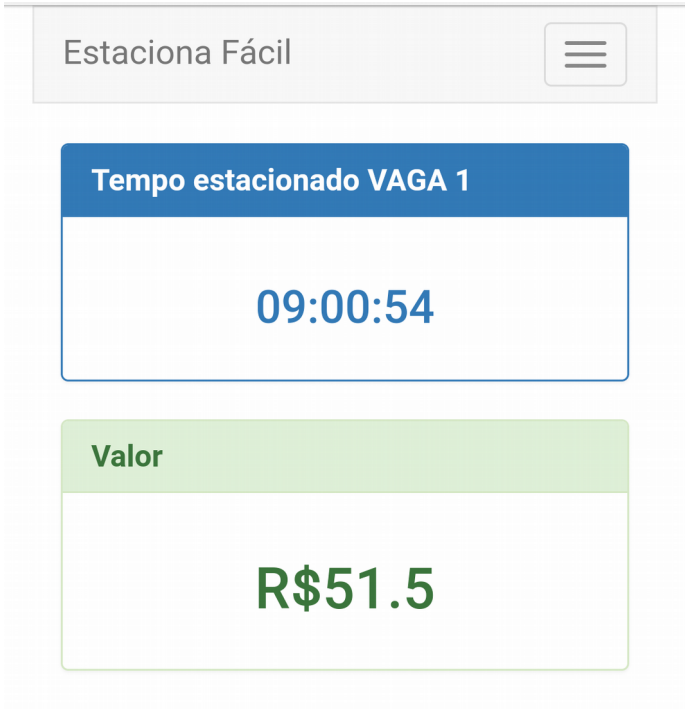
Caso o usuário já tenha uma vaga reservada, é nesta tela que aparecerá o tempo estacionado e o valor ao ser pago ao sair da vaga reservada.

Figura 22 – Tela de manutenção das reservas (vaga reservada) vista em um desktop



Fonte: Próprio Autor (2016)

Figura 23 – Tela de manutenção das reservas (vaga reservada) vista em um celular



Fonte: Próprio Autor (2016)

4.5 TELA DE HISTÓRICO DE RESERVAS

Esta é a tela responsável por mostrar todas as informações das reservas efetuadas pelo usuário.

Figura 24 – Tela de histórico das reservas vista em um desktop

Fonte: Próprio Autor (2016)

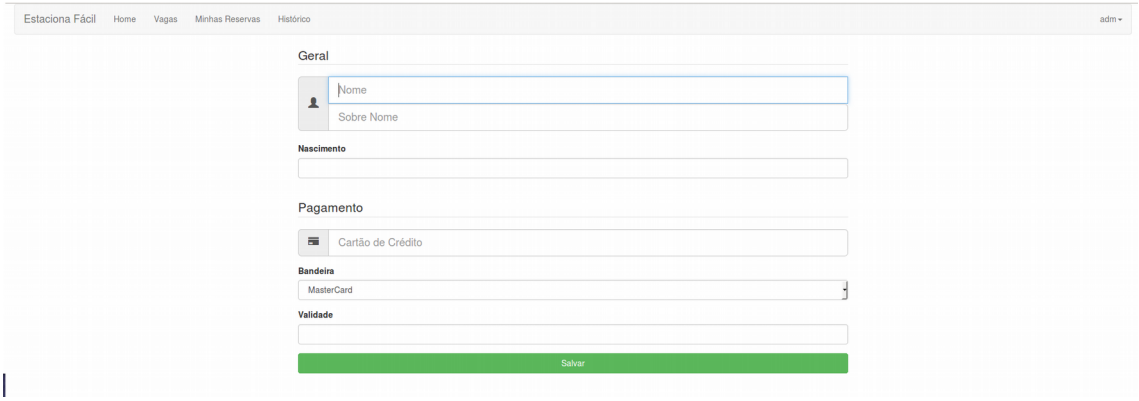
Figura 25 – Tela de histórico das reservas vista em um celular

Fonte: Próprio Autor (2016)

4.6 TELA DE CONFIGURAÇÃO

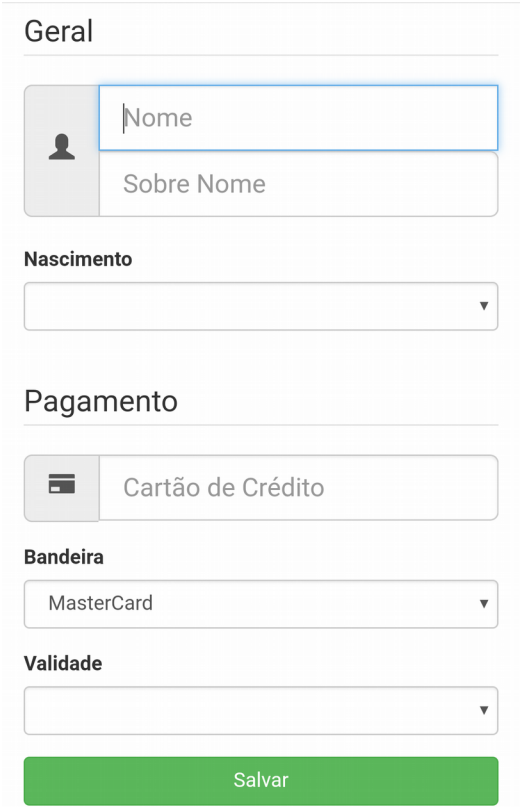
Aqui temos a tela de configuração, onde o usuário irá preencher suas informações e o sistema cobrará pelo cartão de crédito aqui configurado.

Figura 26 – Tela de configuração da conta vista em um desktop



Fonte: Próprio Autor (2016)

Figura 27 – Tela de configuração da conta vista em um celular



Fonte: Próprio Autor (2016)

5. CONCLUSÃO

Com o estacionamento inteligente podemos diminuir a procura de vagas de estacionamento drasticamente, pois não vamos mais precisar ficar procurando qual vaga está disponível quando formos em um estabelecimento que possua esta tecnologia implantada. Com este tipo de tecnologia também vamos diminuir a emissão de poluentes no meio ambiente, pois se vamos economizar tempo evitando voltas e mais voltas sem achar uma vaga, acabamos economizando combustível e consequentemente ajudando o meio ambiente.

E não vamos mais ficar preocupados onde vamos parar o nosso veículo e se terá uma vaga disponível, pois podemos escolher a vaga especifica que gostaríamos de reservar, e com isto evitamos também de nós perder naqueles imensos estacionamentos onde dificilmente conseguimos lembrar onde paramos o veículo, pois com esta tecnologia temos em nosso telefone celular a vaga que foi reservada. Outra funcionalidade de nosso aplicativo para a localização de seu veículo será um rastreador via GPS onde podemos demarcar o local onde estacionamos e posteriormente podemos mandar nosso celular nos guiar pelo estacionamento até a localização de nosso veículo deixando todo o trabalho estressante que passamos ao tentar achar uma vaga e até mesmo posteriormente chegar de volta em nosso veículo sem nos perder para o aplicativo, onde você usuário deve somente se preocupar em se divertir e não se preocupar mais com estes pequenos problemas do dia a dia.

5.1 PROJETOS FUTUROS

Para projetos futuros temos muitas coisas a se melhorar no protótipo onde algumas delas são:

Avisos de permanência de tempo e de valor, onde o usuário configura um teto a que se quer pagar pela vaga, e o aplicativo informará o usuário assim que estiver chegando próximo ao tempo e valor pré-configurado evitando sustos ao pagar pela sua permanência.

Abrir para o gerenciamento de filiais e diversos estabelecimentos.

Possibilidade de colocar este projeto em estacionamentos públicos, o protótipo foi desenvolvido para funcionar em ambientes fechados como

shoppings centers e posteriormente será feito um estudo para aplicar esta tecnologia em estacionamentos públicos.

Integração com outros aplicativos do mercado, como, por exemplo, uma integração com sites de vendas de ingressos online, onde o usuário poderá optar por também já garantir uma vaga para aquele show que ele vai assistir.

REFERÊNCIAS BIBLIOGRÁFICAS

https://arduinohistory.github.io/

http://www.ubiq.com/weiser/weiser.html

http://nic.br/videos/ver/a-internet-das-coisas-explicada-pelo-nic-br/

https://opensource.org/history

Livro: Arduino Guia para colocar suas ideias em prática

https://repositorio.ufsc.br/bitstream/handle/123456789/81296/141801.pdf?sequence=1&isAllowed=y

https://secure.php.net/manual/pt_BR/intro-whatis.php

http://www.abusar.org.br/ftp/pitanga/Intranet/http.PDF

http://www.rfidjournal.com/articles/view?4986

http://www.itrco.jp/libraries/RFIDjournal-That%20Internet%20of%20Things%20Thing.pdf

https://books.google.com.br/books?hl=pt-BR&lr=&id=aL2QThQPuxgC&oi=fnd&pg=PA17&dq=Bill+Joy+internet+of+things&ots=lPnL1pYfV1&sig=bkDKNRkAbPFrTlESV6P1O-o-21c#v=onepage&q&f=false

http://www.informatik.hs-furtwangen.de/~hanne/Pervasive/ACM-ACS-SS08/Ubiq-MarkWeiser.pdf

https://secure.php.net/manual/pt_BR/intro-whatis.php

https://www.w3.org/html/wg/wiki/History

https://www.w3.org/standards/webdesign/htmlcss

http://w3c.github.io/html/introduction.html#history-1

https://pt.wikipedia.org/wiki/JavaScript

https://developer.mozilla.org/en-US/docs/Web/JavaScript

https://news.netcraft.com/archives/2015/01/15/january-2015-web-server-survey.html

https://www.apache.org/foundation/

https://www.ime.usp.br/~jef/bd02

https://www.embarcados.com.br/mqtt-protocolos-para-iot/