UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Controle Inteligente de Vagas para Estacionamentos Utilizando o Conceito de Internet das Coisas

Autor: Thiago Pires Romanelli Vicente
Orientador: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos
2016

Thiago Pires Romanelli Vicente

Controle Inteligente de Vagas para Estacionamento Utilizando o Conceito de Internet das Coisas

Trabalho de Conclusão de Curso apresentado
À Escola de Engenharia de São Carlos, da
Universidade de São Paulo

Curso de Engenharia Elétrica

ORIENTADOR: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos
2016

Vicente, Thiago Pires Romanelli
V632c Controle Inteligente de Vagas para Estacionamento Utilizando o Conceito de Internet das Coisas / Thiago Pires Romanelli Vicente; orientador Evandro Luis Linhari Rodrigues. São Carlos, 2016.

Monografia (Graduação em Engenharia Elétrica com ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2016.

1. Controle Inteligente de Vagas. 2. Internet das Coisas. 3. Camada de Sensoriamento. 4. Camada de Aplicação. 5. Aplicação Web. I. Título.

FOLHA DE APROVAÇÃO

Nome: Thiago Pires Romanelli Vicente

Título: "Controle inteligente de vagas para estacionamento utilizando o conceito de Internet das Coisas"

Trabalho de Conclusão de Curso defendido e aprovado em 25/11/2016,

com NOTA 10,0 (DEZ, ZERO), **pela Comissão Julgadora:**

Prof. Associado Evandro Luis Linhari Rodrigues - Orientador - SEL/EESC/USP

Mestre Alex Antonio Affonso - Doutorando-SEL/EESC/USP

Mestre Leonardo Mariano Gomes - Doutorando-SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado José Carlos de Melo Vieira Júnior

Dedicatória

Aos meus pais, Marcelo e Denise, e às minhas avós, Francisca e Maria do Carmo, por seus ensinamentos, torcida, apoio e suporte ao longo desses anos.

Thiago Pires Romanelli Vicente.

Agradecimentos

- Primeiramente à Deus por todas as realizações em minha vida.
- Aos meus pais, Marcelo e Denise, por estarem ao meu lado dia-a-dia, apoiando, incentivando e torcendo em todas minhas decisões.
- Aos meus outros familiares por contribuírem cada um à sua maneira para meu desenvolvimento como pessoa.
- Aos meus amigos e colegas de turma, por compartilharem das mesmas angústias, anseios, conquistas e realizações durante todos esses anos em que convivemos juntos na Escola de Engenharia de São Carlos.
- Aos meus amigos e colegas de trabalho na PromonLogicalis pelos ensinamentos e conselhos durante o período de estágio.
- Ao meu orientador, Prof. Dr. Evandro Luis Linhari Rodrigues, pelos conselhos e orientação dados à mim ao longo deste trabalho.
- À todos os professores da Escola de Engenharia de São Carlos que contribuíram de alguma forma para minha formação acadêmica.

Thiago Pires Romanelli Vicente.

" É graça divina começar bem. Graça maior persistir na caminhada certa. Mas a graça das graças é não desistir nunca."

Dom Hélder Câmara.

Resumo

Considerando que a Internet das Coisas é vista por muitos como a nova revolução da Internet e observando o crescimento deste conceito nos últimos anos, este projeto tem como objetivo demonstrar o conceito e reproduzir uma arquitetura completa de Internet das Coisas por meio do desenvolvimento de uma solução de controle inteligente de vagas para estacionamento. Para melhor construção do trabalho ele foi dividido em duas etapas. A primeira foi a criação da camada de sensoriamento, onde foi simulada por um sensor ultrassônico associado à um microcontrolador, os quais tinham a tarefa de fazer a verificação das vagas e enviar os dados para a nuvem. A segunda parte consistiu no desenvolvimento da camada de aplicação onde uma aplicação web foi criada para fazer a interface entre o usuário e a lógica de controle de vagas, proporcionando conforto ao usuário e permitindo ao mesmo fazer a reserva de sua vaga remotamente. Ao final do trabalho um protótipo objetivo e funcional foi criado oferecendo ainda condições de ser escalado e melhorado para ser implementado em um ambiente comercial.

Palavras-Chave: Controle Inteligente de Vagas, Internet das Coisas, Camada de sensoriamento, Camada de Aplicação, Aplicação Web.

Abstract

Whereas the Internet of Things is seen by many as the new Internet revolution and observing the growth of this concept in recent years, this project aims to demonstrate the concept and reproduce a complete architecture of the Internet of Things through the development of an intelligent control of parking spaces. To further construction, this work was divided into two stages. The first was the creation of the sensing layer, which was simulated by an ultrasonic sensor associated with a microcontroller, which had the task of checking the vacancies and send the data to the cloud. The second part consisted in the development of the application layer where a web application was created to interface between the user and the logic behind the control of parking spaces, providing comfort and even allowing the user to book its place remotely. At the end of the work an objective and functional prototype was created still offering conditions to be improved for an implementation in a commercial environment.

Keywords: Intelligent Control of Parking Spaces, Internet of Things, Sensing Layer, Application Layer, Web Application.

Lista de Figuras

Lista de Tabelas

Siglas

IoT	*Internet of Things* – Internet das Coisas
MIT	*Massachussets Institute of Technology* – Instituto de Tecnologia de Massachussets
RFID	*Radio Frequency Identification* – Identificação por Rádio Frequência
IBSG	*Internet Business Solutions Group*
Denatran	Departamento nacional de trânsito
WSN	*Wireless sensor network* – Rede de Sensores Sem Fio
M2M	*Machine to Machine* – Máquina para Máquina
RF	*Radio Frequency* - Rádio Frequência
TCP	*Transmission Control Protocol* - Protocolo de Controle de Transmissão
RS232	*Recommended Standard 232*
RS485	*Recommended Standard 485*
I2C	*Inter-Integrated Circuit*
SPI	*Serial Peripheral Interface*
Wi-Fi	*Wireless Fidelity* - Fidelidade Sem Fio
GSM	*Global System for Mobile*
GPRS	*General Packet Radio Services*
CDMA	*Code Division Multiple Access*
3G	*Third Generation* – Terceira Geração
SaaS	*Software as a Service* – Software como um Serviço
PaaS	*Platform as a Service* – Plataforma como um Serviço
IaaS	*Infrastructure as a Service* – Infraestrutura como um Serviço
SimpleDB	*Simple Data Base*
API	*Application Programming Interface* – Interface de Programação de Aplicações
SQL	*Structured Query Language*
CoAP	*Constrained Application Protocol*
MQTT	*Message Queuing Telemetry Transport*
IETF	*Internet Engineering Task Force*
HTTP	*HyperText Transfer Protocol*

UDP	*User Datagram Protocol*
IP	*Internet Protocol* – Protocolo de Internet
DTLS	*Datagram Transport Layer Security*
IBM	*International Business Machines*
QoS	*Quality of Service* – Qualidade de Serviço
LWT	*Last Will and Testment* – Último pedido e Testamento
HTTPS	*Hyper Text Transfer Protocol Secure*
XMPP	*Extensible Messaging and Presence Protocol*
AWS	*Amazon Web Services*
PHP	*Personal Home Page*
LED	*Light emitter diode* – Diodo Emissor de Luz
SoC	*System on Chip* – Sistema em um Chip
SoM	*System on Module* – Sistema em um Módulo
SBC	*Single Board Computer* – Computador em Placa Única
IDE	*Integrated Development Environment*
GPIO	*General Purpose Input/Output* – Entrada e Saída de Propósito Geral
PWM	*Pulse Width Modulation* – Modulação por Largura de Pulso
ADC	*Analogic Digital Converter* – Conversor Analógico Digital
VCC	*Voltage Continuous Current* – Tensão em Corrente Contínua
GND	*Ground* - Terra
SSH	*Secure Shell*
HTML5	*Hypertext Markup Language 5*
URL	*Uniform Resource Locator*
QR code	*Quick Response code* – Código de Resposta Rápida
CPU	*Central Processing Unit*

Sumário

Capítulo 1

Introdução

Muito se debate, hoje em dia, acerca do conceito de *Internet of Things* (Internet das Coisas, em português), considerado por muitos a próxima revolução da Internet e que vem sendo amplamente estudado e desenvolvido mundo afora. Este conceito trata da conexão de diversos dispositivos inteligentes à rede com o intuito de coletar, analisar e distribuir dados transformando-os em informações relevantes para serem utilizadas pelo usuário ou até mesmo serem manipuladas de modo autônomo.

Suas raízes tiveram seu desenvolvimento no MIT (Massachusetts Institute of Technology) em 1999, por um grupo que fazia pesquisas no campo de identificação por rádio frequência (RFID) e tecnologias de sensores emergentes, o Auto-ID Center. Aliados com mais sete universidades em quatro continentes, eles foram responsáveis pelo desenho da primeira arquitetura de IoT [1].

Com o passar dos anos novas plataformas, hardwares e softwares foram surgindo e o entusiasmo da comunidade de desenvolvedores aumentou. Todos esses fatores contribuíram para o rápido crescimento dessa tecnologia que atingiu números antes inimagináveis. De acordo com o Cisco Internet Business Solutions Group (IBSG), o nascimento da "Internet das Coisas" foi o momento onde o número de "coisas ou objetos" conectados à Internet superou o número de pessoas, e deu-se entre os anos de 2008 e 2009 [1]. Segundo previsão feita pela empresa Gartner Inc., cerca de 6,4 bilhões de dispositivos conectados à rede estarão em uso em 2016, número 30% maior que 2015, e esse número alcançará a marca de 20,8 bilhões de dispositivos no ano de 2020 [2].

A seguir a Tabela 1 apresenta a relação dos números levantados pela Gartner Inc.:

Tabela 1 - Dispositivos conectados (IoT) divididos em categorias (Milhões de unidades) [2]

Categoria	2014	2015	2016	2020
Consumidor	2277	2023	4024	13509
Negócios: Organizações inter-profissionais	632	815	1092	4408
Negócios: Verticais específicas	898	1065	1276	2880
Total	3807	4902	6392	20797

A própria Gartner realiza todo ano um estudo de todas as tecnologias emergentes naquele ano, a chamada *Gartner's Hype Cycle*, a qual lista as tecnologias de acordo com sua maturidade e adoção pelas pessoas e como elas são potencialmente relevantes para resolver problemas reais e explorar novas oportunidades. No estudo de 2016 as plataformas de Internet das Coisas estão classificadas no limite entre *innovation trigger* (gatilho de inovação) e *peak of inflated expectations* (pico de expectativas elevadas), caracterizando-as como um assunto em plena discussão e desenvolvimento em toda parte do globo. A seguir na Figura 1.1 está representada a *Gartner's Hype Cycle 2016* [3].

Figura 1.1 - Gartner's Hype Cycle 2016 [3]

Com tantos dispositivos conectados, um enorme número de dados e informações é gerado, exigindo toda uma infraestrutura para dar suporte, armazenar estes dados e manter esta rede que a cada dia cresce mais. É aí que entra outro conceito importante e que dá suporte para a Internet das Coisas, o chamado *Cloud Computing*. Este consiste em fornecer ferramentas computacionais tais como espaço de armazenagem, segurança, poder de processamento, largura de banda, etc., para o desenvolvimento de diversas aplicações [4].

Soluções de *Cloud Computing* são flexíveis e customizadas de acordo com a necessidade do usuário e o mesmo paga apenas pelo serviço contratado, além de ter acesso aos seus dados de qualquer lugar do globo à qualquer hora [5]. É também nesta camada que os dados adquiridos são armazenados e tratados para que deixem de ser apenas dados brutos e se tornem dados efetivos para tomada de decisões, complementando assim o propósito do conceito de Internet das Coisas [6].

A fim de provar os conceitos citados acima, este trabalho procurou contextualizá-los em um cenário comum em nosso dia-a-dia, que é o caos da mobilidade urbana. Este sempre esteve presente na vida dos brasileiros, especialmente aqueles que vivem em metrópoles ou megalópoles.

Historicamente o Brasil sempre adotou o modelo rodoviarista como principal modelo de transporte. Essa política teve início com o ex-presidente Washington Luis na década de 20 e teve sua implementação mais contundente com o ex-presidente Juscelino Kubitschek [7]. Suas principais justificativas eram que com mais rodovias, mais interligado e integrado seria o país e que o incentivo da ampliação da malha ferroviária brasileira viria a atrair empresas internacionais do ramo automobilístico, junto com as indústrias que a amparam (indústrias de autopeças, componentes elétricos, lubrificantes, etc.) [7]. Tendo em vista essa estratégia de atração de capitais e geração de empregos, as ferrovias, que tanto foram utilizadas no período do ciclo do café, foram ficando de lado.

A justificativa para o aumento da frota brasileira a cada ano é justamente esse modelo rodoviarista. Dados disponibilizados pelo Denatran apontam que em um período de 10 anos o crescimento do número de veículos automotores no Brasil foi de 138,6% enquanto que o aumento de sua população foi de apenas 12,2%, ou seja, em uma década o crescimento do número de veículos automotores foi dez vezes maior que o crescimento da população [8].

Em virtude disso, o brasileiro vive o mesmo enredo diariamente: trânsito intenso, engarrafamentos intermináveis, estacionamentos lotados, etc.

Olhando para este contexto, mais especificamente o último ponto citado acima, que se vê uma oportunidade de facilitar a vida de nós motoristas. Todos sabem o quão difícil é para encontrarmos estacionamento apropriado e com pouco tempo de procura, assim, este trabalho tem a finalidade de criar uma alternativa inteligente para solucionar este problema.

Atualmente alguns serviços de estacionamentos inteligentes são oferecidos, tais como disponibilização de luzes indicativas em cada uma das vagas (luzes verdes indicam que as vagas estão livres e vermelhas indicam que estão ocupadas) e aplicativos que possibilitam a reserva de vagas, mas que exigem pagamento antecipado e não se utilizam de sensores para verificação das vagas.

A ideia deste trabalho é criar uma alternativa à estas soluções já existentes, ou seja, um protótipo de um controle inteligente de vagas para estacionamento que alia os conceitos de Internet das Coisas e *Cloud Computing* e que será discutido em detalhes nos tópicos que se seguem.

1.1. Motivação

A motivação deste trabalho se deu após a realização da disciplina SEL0373 – Projetos de Sistemas Digitais com o Prof. Dr. Evandro Luis Linhari Rodrigues, na qual foi possível ter um contato mais aprofundado com microcontroladores e microprocessadores, desenvolvendo um projeto escolhido pelos alunos do início ao fim.

Na ocasião, desenvolveu-se um projeto de automação residencial utilizando uma placa Intel Galileo Gen1 e alguns sensores para simular os atuadores de uma casa, tais como lâmpadas, portão de garagem, sistema de aquecimento, etc.

Uma vez feito o primeiro contato com este mundo de IoT, foi a experiência de estágio profissional que alavancou o interesse em dar sequência nos estudos dessa área, dado que esse conceito foi trabalhado diariamente durante este período. Desta forma, com uma visão de dentro do mercado de trabalho sobre a Internet das Coisas, foi possível notar o quanto o mundo fala sobre e o quanto estamos motivados para não ficar para trás nessa corrida pela tecnologia.

1.2. Objetivo

O objetivo deste trabalho é demonstrar os conceitos e reproduzir uma arquitetura completa de Internet das Coisas por meio do desenvolvimento de uma solução de controle inteligente de vagas para estacionamento, onde o usuário tem a possibilidade de reservar uma vaga antes de chegar ao estabelecimento através de uma aplicação web acessada por seu smartphone.

1.3. Justificativa

A principal justificativa deste trabalho é criar uma solução que facilite a vida dos motoristas utilizando componentes de baixo custo, apresentando escalabilidade e que está alinhado com o que há de mais desenvolvido no quesito Internet das Coisas, gerando conhecimento necessário para trabalhos futuros, continuação e evolução deste e também para o trabalho de terceiros.

1.4. Organização do Trabalho

Este trabalho está distribuído em cinco capítulos, incluindo esta introdução, e estão organizados da seguinte maneira:

Capítulo 2: Descreve a teoria e as principais ferramentas utilizadas para o desenvolvimento deste trabalho.

Capítulo 3: Descreve e discorre sobre os materiais e métodos utilizados para a realização deste projeto.

Capítulo 4: Apresenta os resultados obtidos e discute a atuação das escolhas feitas no projeto.

Capítulo 5: Apresenta a conclusão do trabalho bem como os êxitos e dificuldades encontradas ao longo do desenvolvimento além de propor melhorias para trabalhos futuros derivados deste.

Capítulo 2

Embasamento Teórico

O projeto está baseado em quatro pilares principais: IoT, soluções em *Cloud Computing*, protocolos de comunicação e sistemas embarcados; e em um tópico de apoio que é o QR Code, o qual é peça importante para a implementação da solução proposta. A seguir será discutida a relevância destes tópicos e como serviram de fundamentação para este trabalho.

2.1. IoT – Internet das Coisas

Conforme já mencionado na introdução deste texto, o conceito de Internet das Coisas descreve a inserção de dispositivos inteligentes à rede mundial de computadores, a famigerada Internet, para aquisição de dados e utilização dos mesmos para tomada de decisões, sejam elas autônomas ou automáticas. Estes dispositivos englobam qualquer tipo de aparelho eletrônico, que vão desde *smartphones* e *tablets* até eletrodomésticos, automóveis, etc.

Por mais que o conceito pareça simples, ele possui uma arquitetura complexa para que possa funcionar corretamente e entregar aos seus usuários o que necessitam. Essa arquitetura é composta de uma camada onde acontece a aquisição dos dados, chamada de camada de sensoriamento, passa por uma camada de rede de acesso, depois por uma camada de gerenciamento e operações até chegar a camada de aplicações e serviços. A configuração da arquitetura da Internet das Coisas está descrita na Figura 2.1.

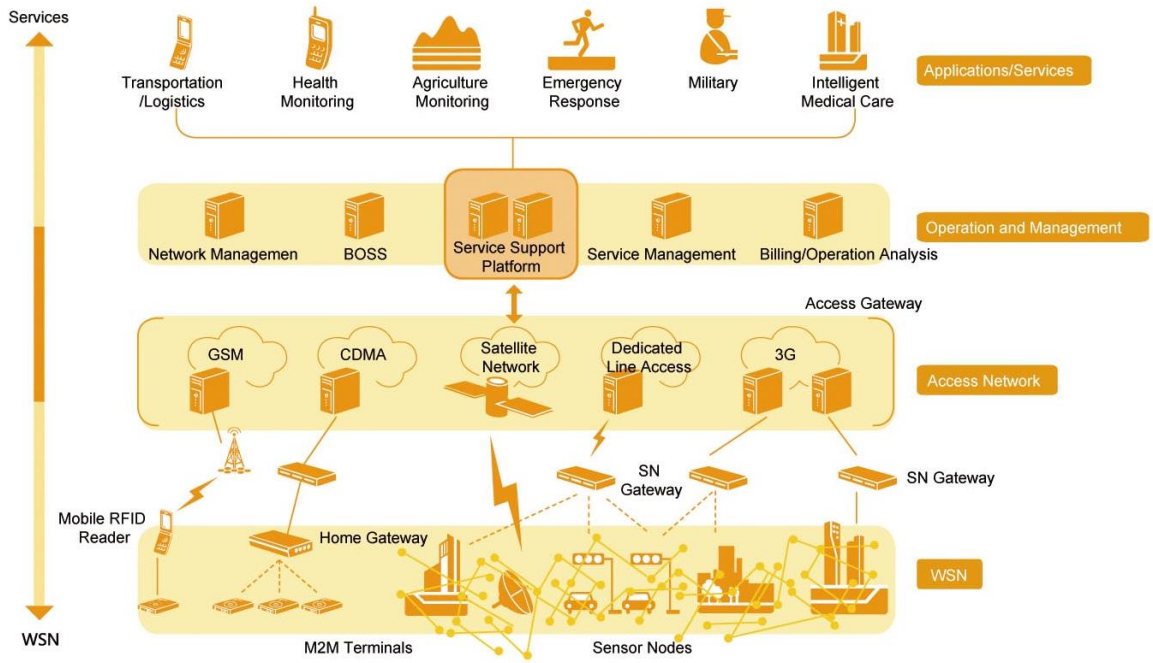

Figura 2.1 - Arquitetura da Internet das Coisas [9]

A camada de sensoriamento é composta por uma *Wired Sensor Network* (WSN), onde dispositivos autônomos utilizam-se de sensores para monitorar condições físicas ou ambientais ao seu redor, e por terminais M2M (*Machine-to-machine*), conhecidos também como *gateways*, que fazem a integração da leitura dos dados feita por esses sensores com Internet [9]. A comunicação entre a WSN e os *gateways* é feita através de diversos protocolos bem conhecidos e utilizados no meio industrial, e variam de acordo com a aplicação a ser monitorada. Os protocolos mais comuns utilizados são: RF, TCP/IP, Modbus, RS232, RS485, I2C, SPI, etc.

Acima da camada de sensoriamento tem-se a camada de redes de acesso que consiste no modo que o *gateway* se conectará com a Internet para o envio dos dados adquiridos. Existe a possibilidade de rede cabeada (cabo ethernet) ou via rede Wi-Fi, onde estão os protocolos de rede celulares GSM, GPRS, CDMA e 3G [9].

Na camada de gerenciamento e operações, os dados recebidos das camadas anteriores são armazenados em bancos de dados localizados em *storages* e recebem um tratamento de *analytics* onde são transformados de sua forma bruta para informações significativas para o usuário. Essas informações obtidas na camada de gerenciamento e operações são utilizadas na camada de aplicações e serviços, as quais facilitam tomadas de decisão para negócios, melhoram serviços e tornam aplicações antes “analógicas” (sem inteligência aplicada) em aplicações inteligentes.

2.2. Cloud Computing

Cloud computing, mais conhecido como “Nuvem”, nada mais é que o oferecimento de recursos computacionais, que vão desde aplicações até *data centers*, sob demanda através da Internet e baseado em um sistema *pay-per-use*, ou seja, pague pelo que for usar [10].

Está localizado na camada de gerenciamento e serviços da arquitetura da Internet das Coisas e é considerado uma ferramenta chave deste conceito, uma vez que é responsável por concentrar todos os dados aquisitados na camada de sensoriamento e por proporcionar ferramentas para a manipulação desses dados.

Este conceito traz consigo grandes benefícios não antes imaginados do ponto de vista de *hardware* tais como a ilusão de que há infinitos recursos computacionais, eliminando a necessidade de qualquer tipo de planejamento a longo prazo, a escalabilidade da demanda por *hardware* quando necessário, ou seja, começar com uma demanda pequena e aumentá-la conforme sua aplicação ou negócio cresça, e também formas de pagamento facilitadas baseadas no tipo de uso dos recursos oferecidos que o usuário utiliza, como por exemplo processador por hora e armazenamento por dia [11].

O serviço de *Cloud Computing* está dividido em três categorias principais, são elas:

- SaaS (*Software as a Service*): Aplicações que estão hospedadas na nuvem e que podem ser acessadas de qualquer lugar via Internet através de um *web browser* [10]. Como exemplo de aplicação SaaS pode-se citar o salesforce.com [12].
- PaaS (*Platform as a Service*): Ambientes baseados na nuvem com toda a infraestrutura para a criação de aplicações na web [10]. Têm-se como exemplo de PaaS a Microsoft Azure [13] e o Google App Engine [14].
- IaaS (*Infrastructure as a Service*): Proporciona aos usuários recursos tais como servidores, recursos de rede, armazenamento e espaço nos data centers utilizando o sistema de *pay-per-use* [10].

As principais características desse conceito são [11]:

- Ultra larga escala: As empresas que prestam esse serviço possuem milhares de servidores espalhados pelo globo.
- Virtualização: O usuário consegue acessar sua aplicação ou servidor de qualquer lugar e a qualquer hora utilizando um notebook ou smartphone.
- Alta confiabilidade: Os provedores de *cloud computing* utilizam diversos processos para garantir a confiabilidade de seu serviço.
- Versatilidade: Não possui um único padrão de aplicação. O serviço de *cloud computing* pode suportar diversos tipos de aplicações ao mesmo tempo em uma única nuvem.
- Alta extensibilidade: A escala da nuvem pode se extender dinamicamente para atender aos requerimentos do usuário.
- Serviço sob demanda: Os provedores de *cloud computing* oferecem uma gama de serviços e o usuário escolhe qual irá adquirir conforme sua necessidade, sendo cobrado somente por aquilo que utilizar.

Hoje no mercado de serviços de *Cloud* existem três grandes empresas concorrendo, são elas: Amazon com seu Amazon Web Services [15], Microsoft com seu Microsoft Azure [13] e Google com seu Google App Engine [14]. A seguir, na Tabela 2, encontra-se uma comparação entre estes três *vendors* levando em conta seus modelos computacionais, modelos de armazenamento e modelos de rede.

Tabela 2-Comparação entre os principais vendors de Cloud Computing [11]

	Amazon Web Services	Microsoft Azure	Google App Engine
Modelo computacional	• Conjunto de instruções da arquitetura x86 (ISA) via Xen VM. • Permite escalabilidade, porém o usuário deve construir o mecanismo.	• Microsoft CLR VM, executado em ambiente gerenciado. • Máquinas são provisionadas baseadas em descrições declarativas; balanceamento automático de cargas.	• Estrutura de aplicação e frameworks pré-definidos; ferramentas escritas em Python. • Balanceamento computacional e de armazenamento automático; rede e servidores a prova de falha.
Modelo de armazenamento	• Gama de modelos desde EBS até SimpleDB. • Escalamento automático varia desde nenhum até completamente automático, dependendo do modelo utilizado. • Garantias de consistência variam dependendo do modelo utilizado. • API's variam de padronizado até proprietário.	• Serviços de dados SQL. • Serviço de armazenamento Azure.	• MegaStore/BigTable
Modelo de rede	• Especificações declarativas da topologia de IP. • Grupos de segurança habilitam quais os nós devem se comunicar. • Disponibilidade de zonas proporcionam abstração de falha de rede independente. • Endereços de IP elásticos proporcionam nomes de rede persistentemente roteáveis.	• Modelo de rede automático baseado nas descrições declarativas dos componentes da aplicação.	• Topologia de rede fixa para acomodar a estrutura 3-tier web. • Escalamento é automático e invisível ao programador.

2.3. Protocolos de Comunicação

Quando olha-se para a definição mais simples de Internet das Coisas, que é a conexão de diversos dispositivos à Internet, percebe-se que o termo conectividade é palavra chave para seu sucesso. Conectividade essa que precisa ser dinâmica para poder atender à grande quantidade de "coisas" conectadas.

Para que esse dinamismo obtenha êxito, algumas características precisam estar presentes nos protocolos utilizados, tais como [16]:

- Emissão de informações de um nó para muitos nós;
- Estar preparado para atender a eventos a qualquer hora que acontecerem;
- Distribuir pequenos pacotes de dados em volumes altos;
- Alta sensibilidade: ao volume de dados sendo transmitido (custo), ao consumo de energia (dispositivos à bateria) e à responsividade (perto de uma resposta em tempo real);
- Segurança e privacidade;
- Escalabilidade;

Com o surgimento dessa nova demanda, novos protocolos direcionados para a Internet das Coisas foram criados. Neste contexto, destacam-se dois protocolos: CoAP e MQTT.

2.3.1. CoAP

Criado pelo órgão IETF (*Internet Engineering Task Force*), o protocolo CoAP, que é a sigla para *Constrained Application Protocol*, é definido como um protocolo de software direcionado para dispositivos simples e que possuem baixa potência, permitindo a eles se comunicarem com a Internet [17].

O CoAP é um protocolo do modelo cliente-servidor que provê uma interação baseada em *request/report* (pedidos/respostas) com acomodação para multicast e que foi desenvolvido para

interoperar com HTTP e RESTful através de proxy's simples, tornando-o compatível para a Internet [18].

Tem como vantagens ser ligado ao protocolo de transporte UDP, que é intencionalmente desenhado para ser menos confiável que o TCP facilitando a transmissão de pequenos pacotes de dados; ter suporte à multicast, que habilita requisições de outros dispositivos ao mesmo tempo; ser seguro, uma vez que utiliza DTLS na camada de transporte UDP; possuir comunicação assíncrona, possibilitando os pacotes de dados de serem recebido/enviados conforme são gerados.

Por ser um protocolo novo e em fase de desenvolvimento, ainda possui instabilidades e não é perfeitamente maduro para certas aplicações [18].

2.3.2. MQTT

Criado pela IBM, o MQTT (sigla para *Message Queuing Telemetry Transport*) foi criado com o objetivo de fazer a comunicação entre equipamentos petrolíferos e satélites.

É um protocolo destinado à comunicação entre dispositivos simples e redes de baixa largura de banda e alta latência [19].

Utiliza um modelo de *publish/subscribe* (publicador/subscritor) para troca de mensagens com um *broker* central, o qual é responsável por gerenciar e direcionar as mensagens entre os múltiplos "emissores" e os "receptores". As mensagens são publicadas em tópicos pré-definidos e ficam concentradas no *broker*. Para se ter acesso a essas mensagens, faz-se necessária a subscrição no mesmo tópico, completando o ciclo da mensagem. Abaixo na Figura 2.2 está uma ilustração do funcionamento do protocolo MQTT.

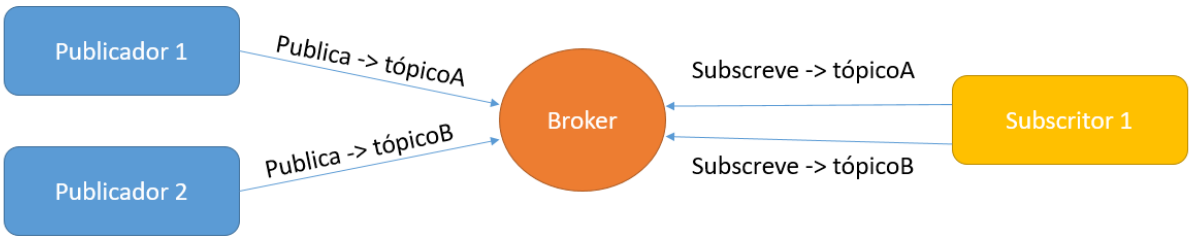

Figura 2.2 - Exemplo de funcionamento MQTT

Este modelo de publicador/subscritor gera algumas vantagens tais como [18]:

- Operação independente dos nós: uma vez que todas as informações são concentradas no *broker*, os nós podem publicar mensagens no broker e subscrever às mensagens de outros nós sem saberem da existência do outro, permitindo uma operação sem inferferência externa.
- Dissociação de tempo e sincronização: Os nós podem publicar uma informação e subscrever-se à um tópico a qualquer momento em que estiverem ativos, permitindo a possibilidade de permanecer em modo *sleepy* (“dormindo”) economizando assim energia.

Além disso, o MQTT possui outras duas funcionalidades interessantes. A primeira é a chamada QoS (*Quality of Service,* qualidade de serviço em português) que são níveis de garantia da entrega de mensagens. Quando o QoS é 0, conhecido como *fire and forget* (“atirar” e “esquecer” em português) os nós mandam os pacotes de dados sem saber se os mesmos chegaram ao seu destino. Para o Qos 1, os nós mandam os pacotes de dados e obtêm pelo menos uma resposta do receptor avisando que a mensagem chegou corretamente. Até que a confirmação de entrega chegue de volta ao nó publicador, a mensagem é armazenada e retransmitida pelo receptor periodicamente. Já em QoS 2 o nó primeiramente avisa ao receptor que está em modo QoS 2. O receptor responde que está pronto para receber a mensagem. Em seguida o nó envia seu pacote de dados e ao receber a mensagem o receptor responde com uma confirmação de entrega para o nó publicador. É o modo de envio mais seguro dos três apresentados [18].

A segunda funcionalidade se chama *Last Will and Testament* (LWT) onde o MQTT disponibiliza uma mensagem que será armazenada pelo *broker* para caso o nó se desconecte da rede. Essa mensagem contém todas as informações do nó previamente à sua desconexão da rede. Além disso, caso o nó não consiga reestabelecer sua conexão, o *broker* avisa à todos os seus subscritores que ele está offline. Caso o nó consiga se reestabelecer, o *broker* o notifica sua situação prévia à desconexão [18].

Com as funcionalidades e características apresentadas, vê-se que o MQTT é o protocolo mais indicado para ser utilizado com o conceito de Internet das Coisas. Além disso possui uma grande comunidade na Internet que se fortalece a cada dia provendo todo o suporte necessário para o desenvolvimento de novas aplicações utilizando o MQTT.

2.3.3. Comparação com outros protocolos

No subtópico anterior foram discutidos os protocolos mais apropriados para a Internet das Coisas. Entretanto existe uma série de outros protocolos que também podem ser utilizados em IoT. Protocolos já conhecidos e difundidos como o HTTP, base do modelo cliente-servidor utilizado para a web, conseguem desempenhar esse papel em IoT, porém não possuem grande eficiência.

Isso é comprovado pelo estudo feito pelo antigo especialista em tecnologias emergentes da IBM Stephen Nicholas, e seus resultados são mostrados na Tabela 3 abaixo.

Tabela 3 - Comparação entre HTTP e MQTT [20]

Characteristics		3G		WiFi	
		HTTPS	***MQTT***	***HTTPS***	***MQTT***
Receive Messages	**Messages / Hour**	1,708	160,278	3,628	263,314
	Percent Battery / Hour	18.43%	16.13%	3.45%	4.23%
	Percent Battery / Message	0.01709	0.00010	0.00095	0.00002
	Messages Received (Note the losses)	240 / 1024	1024 / 1024	524 / 1024	1024 / 1024
Send Messages	**Messages / Hour**	1,926	21,685	5,229	23,184
	Percent Battery / Hour	18.79%	17.80%	5.44%	3.66%
	Percent Battery / Message	0.00975	0.00082	0.00104	0.00016

Analisando a tabela acima é possível notar a robustez do MQTT, que consegue ter uma maior eficiência no envio e recebimento de mensagens, tanto em 3G quanto em WiFi, e também é mais econômico na utilização de bateria quando comparado ao HTTP.

Outros protocolos como XMPP e RESTful HTTP também conseguem suprir as necessidades de IoT, entretanto cada um deles é direcionado para uma aplicação específica. A seguir, na Tabela 4, faz-se uma comparação entre MQTT, XMPP, CoAP e RESTful HTTP em termos de funcionalidades e casos de aplicações de sucesso.

Tabela 4 - Comparativo entre protocolos IoT [19]

Protocol	CoAP	XMPP	RESTful HTTP	MQTT
Transport	UDP	TCP	TCP	TCP
Messaging	Request/Response	Publish/Subscribe Request/Response	Request/Response	Publish/Subscribe Request/Response
2G, 3G, 4G Suitability (1000s nodes)	Excellent	Excellent	Excellent	Excellent
LLN Suitability (1000s nodes)	Excellent	Fair	Fair	Fair
Compute Resources	10Ks RAM/Flash	10Ks RAM/Flash	10Ks RAM/Flash	10Ks RAM/Flash
Success Stories	Utility Field Area Networks	Remote management of consumer white goods	Smart Energy Profile 2 (premise energy management, home services)	Extending enterprise messaging into IoT applications

2.4. Sistemas Embarcados

Sistemas embarcados são sistemas com propósitos especiais nos quais o computador está completamente encapsulado pelo dispositivo que ele controla e sua complexidade não transparece para o usuário. Além disso são programados para performar apenas tarefas previamente definidas e específicas para a aplicação em que estão empregados [21].

Em termos de design os sistemas embarcados possuem algumas premissas à serem atendidas pelos fabricantes para se tornarem competitivos no mercado, tais como custo de manufatura, custo de design, performance e consumo de energia, assim a cada dia surgem novos dispositivos cada vez mais completos, independentes e robustos [21]. Como exemplo disso pode-

se citar os dispotitivos *Single Board Computer* ou SBC, os *System on Chip* ou SoC e os *System on Module* ou SoM.

Os SBC's são capazes de oferecer tudo o que um computador funcional oferece em uma única placa e em tamanho reduzido, ou seja, possui um microprocessador, memória, I/O's, etc [22]. Pode-se tomar como exemplo de SBC placas bem conhecidas e difundidas na comunidade de desenvolvedores, tais como o Arduino [23], Raspberry Pi [24], Beaglebone Black [25], etc.

Já os SoC e SoM combinam os recursos eletrônicos de vários componentes computacionais em um único chip integrado ou módulo integrado. A diferença entre eles é que os SoM necessitam de uma *base board* (placa base) para que possam funcionar, entretanto ganham a possibilidade da modularidade, uma vez que é possível utilizar diferentes módulos para diferentes aplicações com uma mesma *base board* [22] [26]. Exemplos de SoC e SoM são o NodeMCU (ESP8266) [27], Toradex Colibri [28] e a Intel Edison [29].

Os sistemas embarcados se utilizam de interfaces de comunicação comuns a muitos dispositivos tais como RS232, RS485, SPI, I2C, I/O's digitais e analógicas, placas de rede, entre outros. Além disso, alguns deles já possuem módulos Wi-Fi e Bluetooth, o que facilita ainda mais a troca de informações entre dispositivos.

Essa adaptabilidade dos sitemas embarcados em relação à comunicação com outros dispositivos, seu tamanho físico e seu método de trabalho, os tornam peças importantes no desenvolvimento do mundo da Internet das Coisas, sendo utilizados em praticamente todas as aplicações desse ramo.

2.5. QR Code

Quick response codes, ou códigos de resposta rápida são códigos de barras bidimensionais capazes de armazenar diversos tipos de informações em quatro formatos diferentes: numérico, alfanumérico, byte/binário e kanji (caracteres da língua japonesa). Essas informações armazenadas nos QR codes variam desde URL's de sites até números telefônicos e são empregadas em diferentes aplicações, sejam elas para publicidade ou controle de estoque [30].

São facilmente interpretados pelos computadores e podem ser escaneados a partir de qualquer dispositivo celular que tenha um aplicativo de leitura de QR codes instalado, o qual utilizará a câmera do dispositivo para realizar tal processo.

Para este trabalho o QR code é considerado uma peça importante no desenvolvimento do protótipo de controle de vagas, sendo responsável pela principal verificação que indica se o usuário que reservou a vaga é o mesmo que estacionou seu veículo na vaga física correspondente à reserva.

Capítulo 3

Materiais e Métodos

Neste capítulo serão apresentados os materiais e os métodos implementados na construção da prova do conceito discutido no capítulo anterior. Trata-se de um protótipo de um controle inteligente de vagas para estacionamento de estabelecimentos privados que implementa todas as etapas de uma arquitetura de Internet das Coisas e cujos detalhes de sua construção serão descritos nos tópicos que seguem.

3.1. Materiais

Para a construção deste protótipo foram necessários os seguintes materiais:

- Servidor hospedado na Amazon AWS [15] com um sistema operacional Ubuntu [31] instalado;
- Softwares instalados no servidor hospedado na nuvem: Apache, MySQL e PHP;
- ESP8266 – NodeMCU Devkit v1.0 [27];
- Sensor ultrassônico HC-SR04;
- Protoboard
- Conectores
- LED
- Software PuTTy [32];
- Software Filezilla [33];

A escolha por um servidor hospedado em um serviço de *cloud computing* foi para exemplificar a camada de gerenciamento e operação da arquitetura de IoT, uma vez que todos os dados e aplicações ficam concentrados nesse servidor. O motivo da hospedagem nos servidores da Amazon deu-se devido a facilidade que este provedor oferece, tanto na disponibilização de suas ferramentas e facilidade de acesso quanto na facilidade de pagamento, além de um atrativo extra que foi o oferecimento de um ano grátis para a utilização de seus serviços.

O ESP8266 foi escolhido devido à sua versatilidade e sua facilidade de implementação. Visto que o aluno já possuía experiência com o uso do mesmo, uma vez que em seu estágio se faz uso corriqueiro, este dispositivo se sobressaiu na comparação com outros no mercado.

O Sensor HC-SR204 foi escolhido devido à necessidade de reprodução de uma vaga no protótipo do controle inteligente de vagas para estacionamento e seu papel no projeto será descrito na seção “Métodos”.

Os softwares PuTTy e Filezilla foram escolhidos pela familiaridade do aluno com os mesmos e pela facilidade na implementação das tarefas que estes proporcionam.

3.1.1. Módulo ESP8266 -– NodeMCU Devkit v1.0

O módulo ESP8266 é um SOC (*System on chip*) que possui um módulo Wi-Fi integrado além de uma pilha de protocolos TCP/IP dando a qualquer microcontrolador a capacidade de se conectar à ele por meio de sua rede Wi-Fi. Possui diversos modelos que variam entre si por alguns recursos, tais como número de portas de entrada e saída, disponibilidade de entrada analógica, etc.

Possui capacidade de processamento e memória suficientes para ser integrado com sensores ou outras aplicações através de suas GPIOs, além de ser facilmente programado através da IDE do Arduino [34]. Um exemplar do módulo NodeMCU Lua ESP-12E está representado na Figura 3.1 a seguir.

.

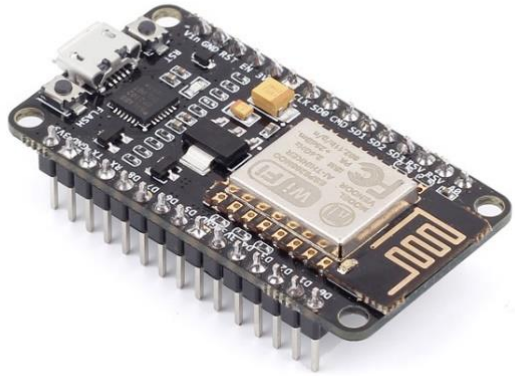

Figura 3.1 - Módulo NodeMCU Lua ESP-12E [35]

Suas especificações são as seguintes [36]:

- Módulo NodeMcu Lua ESP-12E
- Wireless padrão 802.11 b/g/n
- Antena embutida
- Conector micro-usb
- Modos de operação: STA/AP/STA+AP
- Suporta 5 conexões TCP/IP
- Portas GPIO: 11
- GPIO com funções de PWM, I2C, SPI, etc
- Tensão de operação: 4,5 ~ 9V
- Taxa de transferência: 110-460800bps
- Suporta Upgrade remoto de firmware
- Conversor analógico digital (ADC)
- Distância entre pinos: 2,54mm
- Dimensões: 49 x 25,5 x 7 mm

É uma interessante ferramenta para IoT pelo fato de sua comunicação sem fio, baixo preço, facil programação e integração com sensores.

3.1.2. Sensor ultrassônico HC-SR04

É um sensor capaz de medir distâncias que variam em uma faixa que vai de 2cm até 4m com boa precisão. É um módulo de baixo custo que possui circuito com emissor e receptor acoplados e quatro pinos para medição: Vcc, trigger, echo e GND. Um exemplar do sensor Ultrassônico HC-SR04 está representado na Figura 3.2 a seguir.

Figura 3.2 - Sensor Ultrassônico HC-SR04 [36]

Seu funcionamento se dá da seguinte forma: o sensor emite ondas ultrassônicas em uma frequência de 40 kHz que viaja pelo ar até encontrar um obstáculo à sua frente. Caso isso ocorra essas ondas rebatem no objeto e retornam para o módulo. Considerando o tempo que a onda demorou para voltar ao módulo e a velocidade do som, é possível assim calcular a distância que o obstáculo está do módulo [37].

O pino trigger tem que ficar 10µs em HIGH para que depois sejam liberados oito ciclos de ondas ultrassônicas, que viajarão pelo ar na velocidade do som e serão recebidas pelo pino echo. Este por sua vez retornará para o microcontrolador o tempo em microsegundos que a onda viajou. Fazendo-se cálculos simples é possível se obter a distância medida pelo sensor [37].

3.2. Métodos

Este protótipo simulará uma arquitetura completa de Internet das Coisas, desde a camada de sensoriamento até a camada de aplicação. Sua visão geral está exemplificada na Figura 3.3 a seguir.

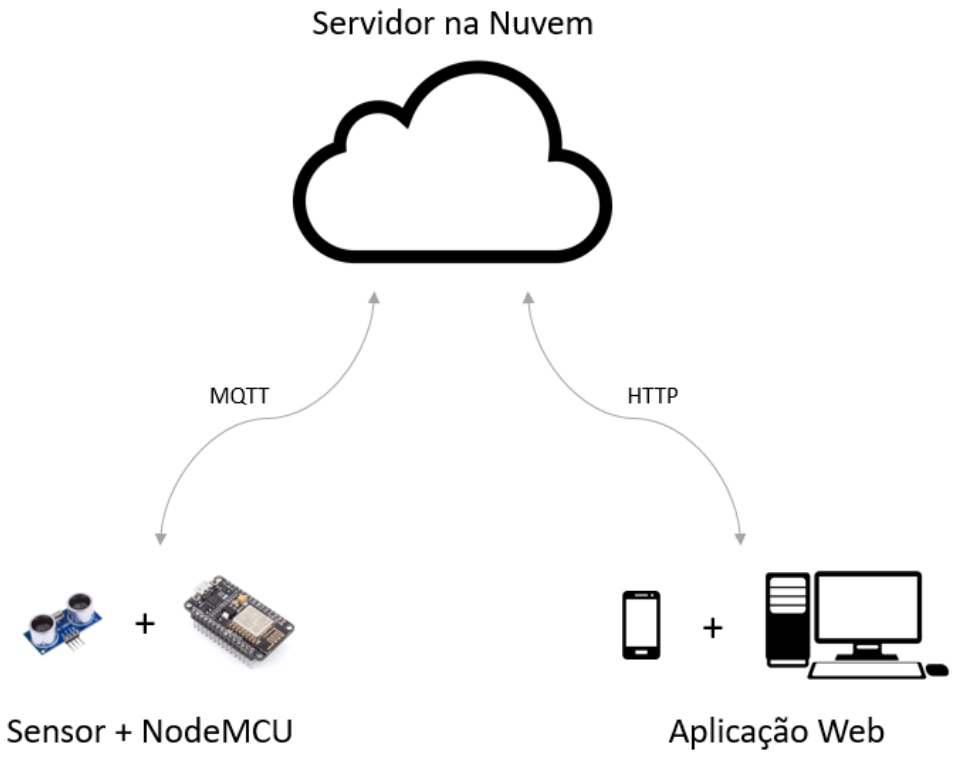

Figura 3.3 - Visão geral do protótipo de Controle Inteligente de Vagas Para Estacionamento

O seu desenvolvimento foi dividido em duas fases: implementação da camada de sensoriamento (microcontrolador + sensor) e criação da camada de aplicação (software da aplicação web).

A primeira fase teve como objetivo a integração do sensor HC-SR04 com o microcontrolador NodeMCU para atuar na verificação da presença de carros na vaga do estacionamento. O sensor foi ligado ao NodeMCU conforme esquemático apresentado na Figura 3.4.

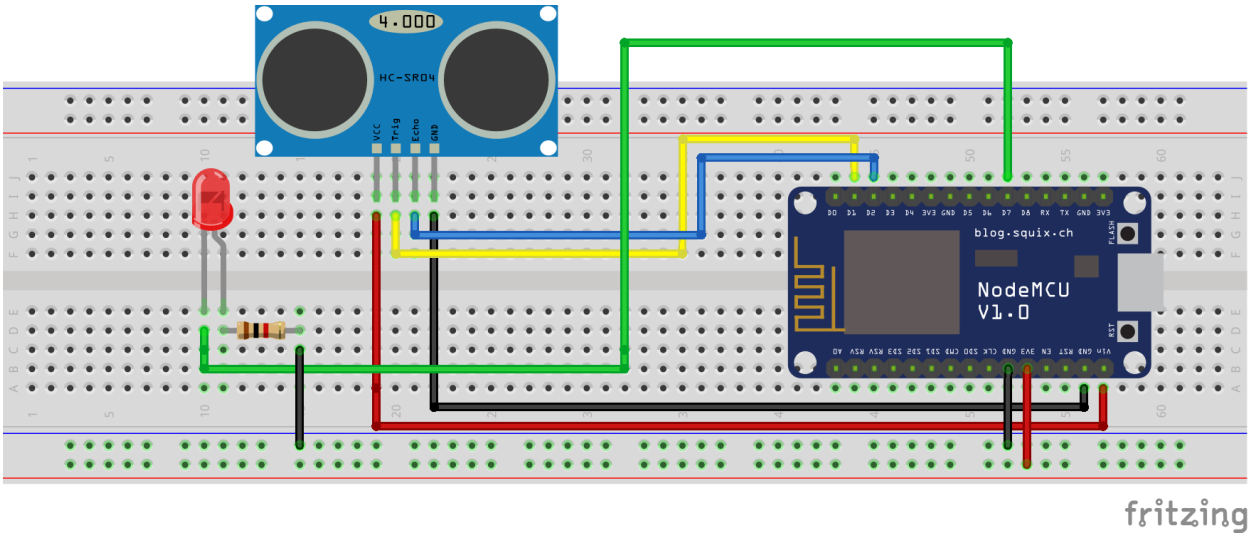

Figura 3.4 - Esquemático da ligação do sensor HC-SR04 com NodeMCU

Em seu código, o NodeMCU se conecta à internet via WiFi, se conecta ao *broker* MQTT onde serão publicados os pacotes de dados gerados pelo sensor e faz a leitura da distância medida pelo sensor. Se a distância lida for menor que um *threshold* definido previamente, uma variável recebe valor igual a 1, acusando que há presença de carro na vaga. Caso a distância for maior que o *threshold* a variável recebe valor 0, indicando que a vaga está desocupada. O valor desta variável é então publicado no *broker* via MQTT para que possa ser utilizado pela aplicação web e entrar na lógica de verificação das vagas. O *loop* leva 5 segundos para realizar todo o processo, ou seja, as mensagens são publicadas no *broker* a cada 5 segundos.

Este mesmo código possui uma função de *callback* que será utilizada para acender um LED caso ocorra alguma inconsistência na lógica da aplicação web. Ela funciona da seguinte forma: o NodeMCU ao se conectar à Internet se subscreve em um tópico específico para essa função. A função de *callback* fica então sempre “escutando”, ou seja, fica esperando que alguma mensagem chegue ao tópico preparado para ela. Caso isso ocorra, a função é executada. Nesse caso ela tem a finalidade de acender um LED que serve para simular um alerta à central que toma conta do estacionamento caso ocorra alguma inconsistência na ocupação das vagas.

Com a primeira fase do projeto concluída deu-se início à segunda fase, que teve como objetivo a criação da aplicação web onde o usuário consegue fazer suas reservas de vaga e acompanhar todos os detalhes da sua estadia no estacionamento.

Para a realização dessa fase fez-se necessário também a criação da camada de gerenciamento na nuvem, ou seja, o servidor onde estão hospedadas as aplicações necessárias para o gerenciamento dos dados e implementação das lógicas.

O primeiro passo foi a criação de uma conta na Amazon AWS, provedor de serviços de *cloud computing*. Feito isso criou-se uma instância já com Ubuntu instalado e definiu-se os grupos de segurança responsáveis por gerenciar as portas de comunicação da máquina virtual. Foram liberadas as portas para comunicação via SSH, com o servidor MySQL, HTTP, HTTPS, e com o servidor MQTT. Em seguida utilizando-se do software PuTTy estabeleceu-se uma conexão via SSH para acessar a máquina virtual e fazer a instalação de todos os pacotes necessários. Foram instalados então os servidores Apache e MySQL e o interpretador da linguagem PHP.

O servidor web Apache é responsável por toda a comunicação cliente-servidor em HTTP, sendo necessário para a construção da aplicação web do controle inteligente de vagas para estacionamento. O servidor de banco de dados MySQL é responsável por todo o armazenamento de dados que a aplicação web utilizará, sendo eles dados criados por ela mesma e os dados recebidos via MQTT do sensor ultrassônico. Já o interpretador de PHP é importante para traduzir os códigos em linguagem PHP utilizados na programação da aplicação web.

Após a instalação destes pacotes, fez-se necessária a instalação do servidor Mosquitto MQTT, que age como *broker* e concentra todas as publicações e subscrições feitas tanto pelo lado do microcontrolador quanto pelo lado da aplicação web.

Terminada as instalações dos componentes da camada de gerenciamento, deu-se início ao desenvolvimento do site do controle inteligente de vagas para estacionamento. As linguagens de programação utilizadas foram: HTML5, PHP e JavaScript. A linguagem HTML5 é responsável por toda interface visual da aplicação, o JavaScript é uma linguagem mais dinâmica utilizada para alguns blocos de código específicos como por exemplo a interpretação do QR Code (o qual será detalhado a diante), e o PHP foi a linguagem escolhida para ficar no lado do servidor realizando todas as lógicas principais da aplicação.

O site é dividido em três páginas principais conforme a Figura 3.5:

- Página Inicial;
- Página para reserva de vagas;
- Página “Minha Conta”;

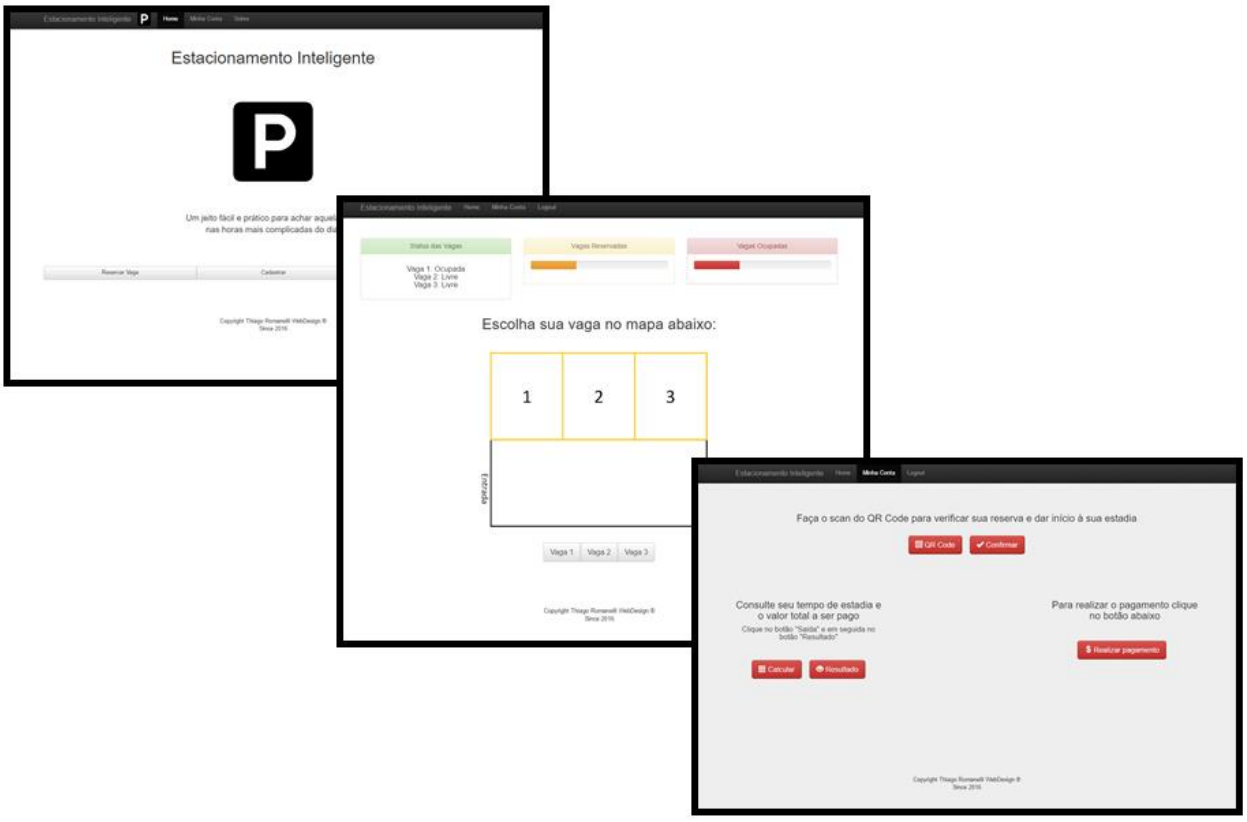

Figura 3.5 - Layouts das páginas da aplicação web

A Página Inicial é aquela que abre para o usuário quando acessa a URL do site. Esta apresenta três opções de navegação no site: “Reservar vaga”, “Cadastrar” e “Contactar Central”. Ao clicar na opção “Reservar Vaga”, a página pede para o usuário inserir o Login e a Senha cadastrados previamente para ter acesso à página de reserva de vagas. Ao clicar na opção “Cadastrar”, a página mostra dois campos para o usuário cadastrar um Login e uma Senha de acesso. Por fim, ao clicar em “Contactar Central”, a página abre o aplicativo de telefone do celular do usuário já com o telefone da central na tela para caso algum problema ocorra com o mesmo. O fluxograma da Página Inicial se encontra na Figura 3.6.

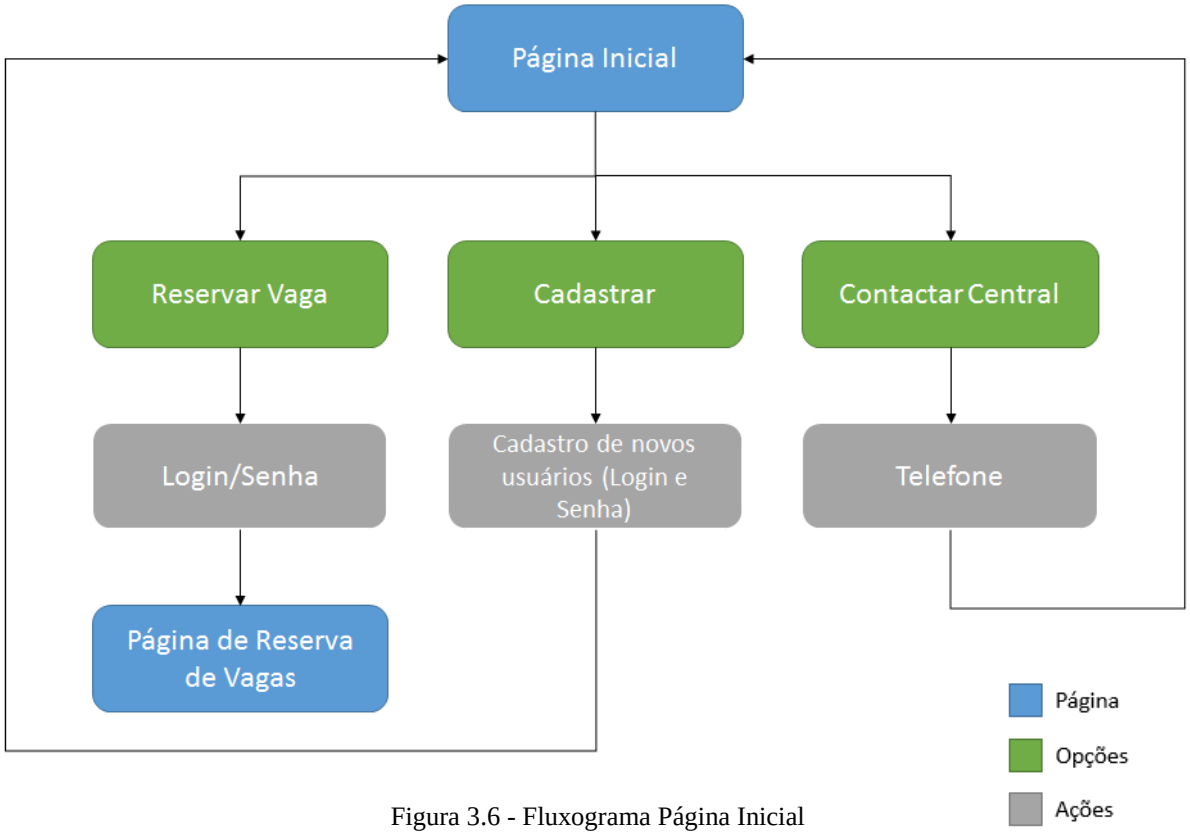

Figura 3.6 - Fluxograma Página Inicial

Ao acessar a Página de Reserva de Vagas o usuário se depara com três quadros informativos. O primeiro informa quais vagas estão livres/ocupadas/reservadas, o segundo mostra o número de vagas reservadas e o terceiro mostra o número de vagas ocupadas. Com isso o usuário consegue saber o status do estacionamento no momento da reserva e pode escolher a melhor vaga disponível para ele. Após isso seleciona a vaga desejada, o dado da reserva é enviado para o banco de dados no servidor e o usuário é redirecionado para a página "Minha Conta". Por trás dessa página, do lado do servidor, existe a lógica de verificação de vagas onde o servidor busca no banco de dados MySQL o status da vaga enviado pelo sensor via MQTT e as vagas já reservadas por outros usuários. Faz-se o cruzamento das informações e tem-se o resultado de vagas disponíveis, reservadas e ocupadas (disponíveis = nenhuma reserva e a *flag* do sensor é 0, reservada = reservado por algum usuário e a *flag* do sensor é 0, ocupada = reservada por algum usuário e a *flag* do sensor é 1, erro = *flag* do sensor é 1 e não há nenhuma reserva). Na Figura 3.7 pode-se ver o fluxograma do funcionamento da Página de Reserva de Vagas.

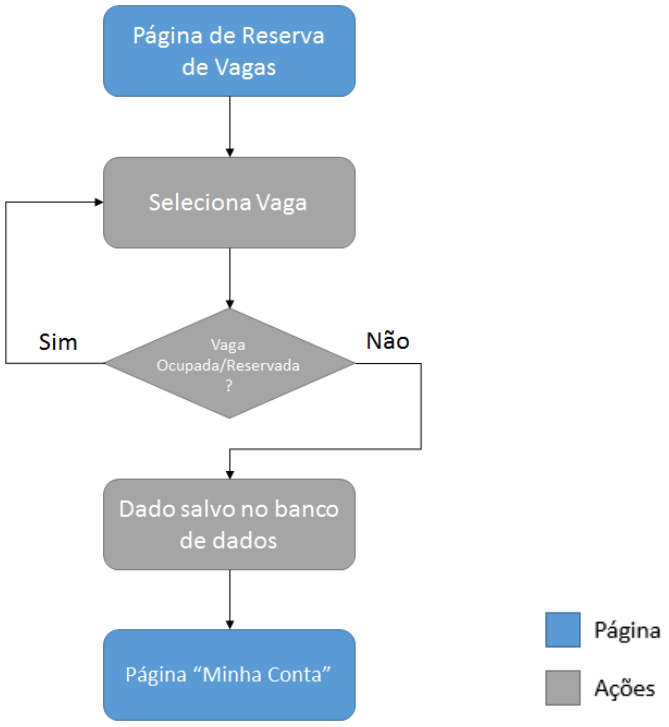

Figura 3.7 - Fluxograma funcionamento da Página de Reserva de Vagas

Para melhor entendimento, a Tabela 5 contém a demonstração da lógica de verificação de disponibilidade de vagas.

Tabela 5 - Lógica de verificação de disponibilidade de vagas

Dados do sensor	Há reserva?	Resultado
0	Não	Disponível
0	Sim	Reservada
1	Sim	Ocupada
1	Não	Erro

Nota-se na Tabela 5 que quando o sensor indica que há presença de carro mas não encontra uma reserva atrelada àquela vaga, ocorre uma inconsistência. Assim uma mensagem de "Erro" é mostrada no primeiro painel da Página de Reserva de Vagas, indicando à central e ao usuário que alguém parou naquela vaga sem ter feito sua devida reserva.

A página “Minha Conta” é onde o usuário tem as funcionalidades de confirmar a vaga que reservou ao chegar no estabelecimento, ver o tempo de estadia e o valor a ser pago, e realizar o pagamento online.

A confirmação de vaga precisa acontecer para evitar que haja trocas de vagas indevidas, ou seja, por exemplo, o usuário A parar na vaga do usuário B. Dessa forma, assim que o usuário chegar em sua vaga, o mesmo deve scanear um QR Code que estará visível naquela vaga (cada vaga terá um QR Code exclusivo) para confirmar sua chegada ao estabelecimento.

A lógica funciona da seguinte maneira: Ao reservar uma vaga, o número dela fica salvo no banco de dados atrelado ao Login do usuário. Quando este usuário chega ao estabelecimento e estaciona em sua vaga, o sensor detecta a presença do carro. Ao fazer a leitura do QR Code, o código retornará um número correspondente à vaga em que o QR Code está fixado. A contagem do tempo de estadia só se inicia quando há presença de carro (detectado pelo sensor) e a vaga reservada coincide com a leitura do QR Code (por exemplo, vaga 1 foi reservada e o QR Code retornou vaga 1 em sua leitura). Qualquer outra configuração além desta indica situação de erro. Quando isso ocorre o usuário é avisado que houve alguma inconsistência e a central que cuida do estacionamento também é alertada para tomar qualquer providência necessária. Neste protótipo a simulação dessa inconsistência se dá através do acendimento de um LED (que é acionado via MQTT, ou seja, quando a inconsistência é detectada o código publica em um tópico pré-determinado. O microcontrolador está preparado para ficar escutando à este tópico e caso alguma mensagem chegue ele aciona o LED). A seguir a Tabela 6 mostra uma exemplificação do funcionamento da lógica para a vaga 1.

Tabela 6 - Exemplificação funcionamento da lógica de vagas

Exemplo	Sensor Vaga 1	Reserva	Leitura QR Code	Resultado	LED
#1	Há carro	Vaga 1	Vaga 1	Inicia contagem do tempo de estadia	Apagado
#2	Não há carro	Vaga 1	Vaga 2	Erro! Central acionada	Aceso

No exemplo #1 da Tabela 6, o usuário reservou a vaga 1, estacionou corretamente na vaga reservada por ele e a leitura do QR Code confirmou que ele estava na vaga correta. Com isso seu tempo de estadia foi iniciado (tempo inicial é salvo no banco de dados) e não ocorreu nenhuma inconsistência.

No exemplo #2 da Tabela 6, o usuário reservou a vaga 1, mas estacionou seu carro na vaga 2. Assim o sensor da vaga 1 não acusou a presença de carro e a leitura do QR Code retornou um valor diferente daquele que ele havia reservado. Dadas as inconsistências, a central foi acionada e o LED foi aceso.

Outra funcionalidade da Página “Minha Conta” é a opção de ver o tempo de estadia e o valor devido ao estabelecimento. Ao clicar no botão “Calcular” o código realiza a seguinte lógica: Para calcular o tempo total da estadia até aquele momento, o código pega o tempo atual e subtrai do tempo inicial salvo no banco de dados após a confirmação da vaga feito pelo usuário. Para calcular o valor a ser pago o código faz uma simples multiplicação do número de horas por um valor pré-determinado. Caso a estadia total passe de seis horas, o valor cobrado será de R$25,00, valor de uma diária. Estes dois dados são apresentados para o usuário ao clicar-se no botão “Resultado”.

A última opção desta página é a de “Realizar Pagamento” que apenas faz a simulação da saída do usuário do estabelecimento. Ao clicar neste botão o código calcula o tempo de estadia total e o valor a ser pago da mesma forma que a função anterior e mostra para o usuário em uma janela de alerta. Além disso o código apaga no banco de dados os valores de “tempo inicial” e “vaga reservada”, habilitando o usuário para novo uso do estacionamento.

A seguir, a Figura 3.8 contém o fluxograma da página “Minha Conta”.

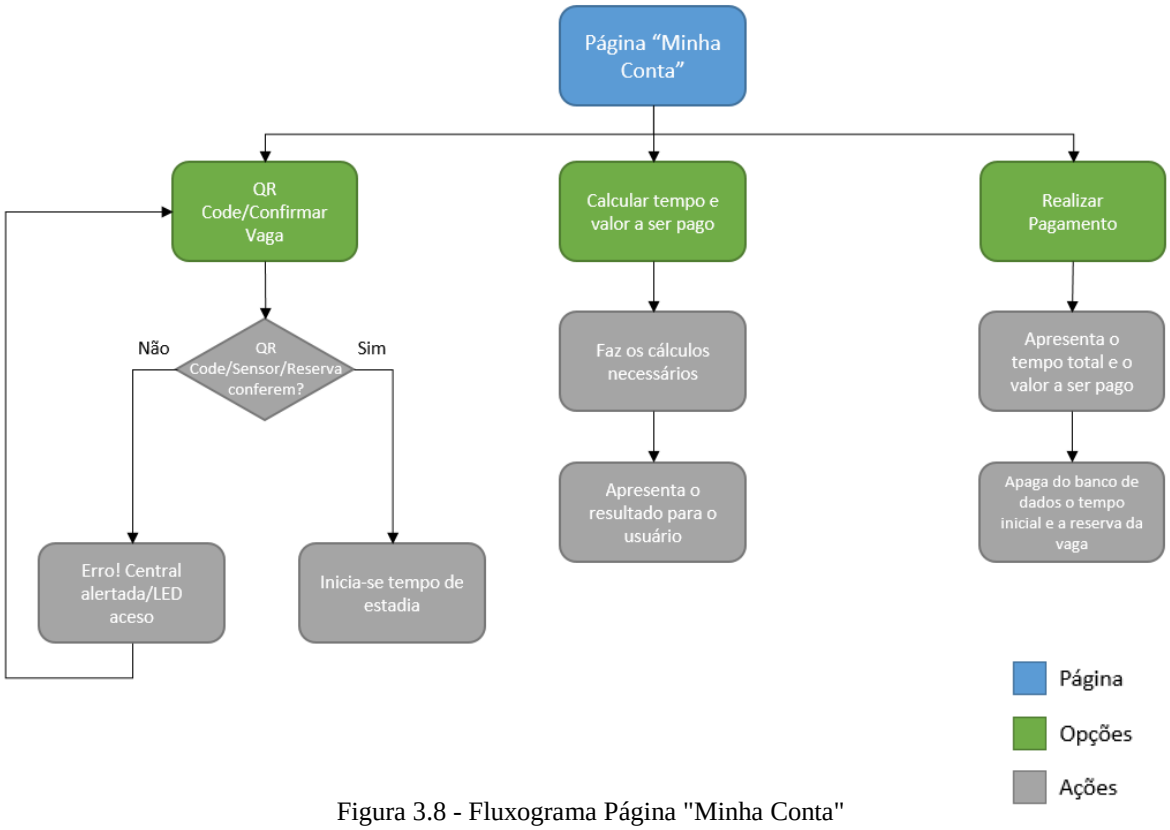

Figura 3.8 - Fluxograma Página "Minha Conta"

Todo o gerenciamento dos dados armazenados no banco de dados MySQL é feito pelo software phpMyAdmin, o qual também foi instalado no servidor que está hospedado na nuvem. Com ele é possível gerenciar todas as tabelas de dados, monitorando as vagas, reservas, usuários, etc. de uma forma visual e simples através de uma interface gráfica. Com ele criou-se uma tabela onde se faz os logs de todos os usuários que passaram pelo estacionamento para controle interno do administrador do estabelecimento, registrando o tempo de estadia total, o valor pago, a data da visita e a identificação do usuário.

Ao final, tem-se na Figura 3.9 um fluxograma que mostra um passo-a-passo geral da aplicação completa do controle inteligente de vagas para estacionamento.

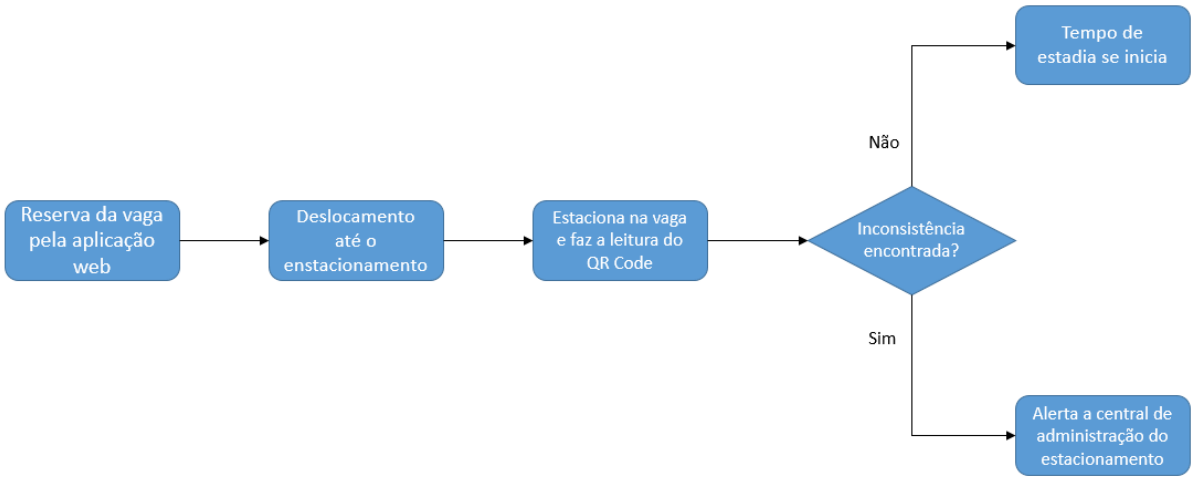

Figura 3.9- Passo-a-passo geral da aplicação completa do controle inteligente de vagas para estacionamento

Capítulo 4

Resultados e Discussões

Conforme citado na introdução deste trabalho, algumas soluções de estacionamentos inteligentes já existem no mercado. Todas elas trazem uma certa praticidade ao usuário do estacionamento mas possuem algumas limitações. Analisando a opção por luzes indicativas nas vagas, o usuário ganha uma facilidade visual, onde ao ver a luz verde indica que aquela vaga está livre e vermelha indica que a vaga está ocupada. É uma opção simples mas não tão prática, uma vez que esta solução não tira do usuário a necessidade de ficar procurando pelas vagas disponíveis. Analisando o aplicativo disponibilizado por uma rede de estacionamentos, este possibilita a reserva de vaga e o pagamento da estadia online, porém é necessário o pagamento antecipado da reserva e segundo *reviews* de consumidores é necessário ter o comprovante de reserva impresso para comprovação da mesma, sendo pouco prático e abrindo margem à erros. O protótipo construído neste trabalho apresenta uma alternativa ainda mais automatizada que as listadas acima, onde se utiliza de um sensor para verificação de presença de carro na vaga e possibilita a comprovação da reserva pelo próprio usuário sem necessidade de pagamento adiantado, uma vez que o tempo de sua estadia bem como o total a ser pago começam a ser contados a partir do momento que o usuário faz a verificação da vaga no próprio estabelecimento. Além disso é possível fazer uma integração com a solução de luzes indicativas, para também auxiliar o cliente a encontrar sua vaga ao adentrar o estabelecimento.

O desenvolvimento da primeira fase, que foi a montagem da camada de sensoriamento, foi simples, uma vez que tanto o sensor ultrassônico HC-SR04 como o microcontrolador NodeMCU são bem conhecidos pela comunidade Maker (comunidade de entusiastas pela tecnologia com

paixão por criar e construir os próprios projetos) e comunidade do Arduino, havendo diversos tutoriais explicando como integrá-los e seu funcionamento completo. Um ponto que se teve cuidado foi que o sensor é alimentado com 5V e o microcontrolador oferecer apenas 3,3V para alimentação. Entretanto se o NodeMCU for alimentado via cabo USB, um de seus pinos, Vin, fornece 5V, assim conseguiu-se alimentar corretamente o sensor e obteve-se o funcionamento correto da solução.

A comunicação via MQTT mostrou-se eficiente e simples de ser implantada tal qual seu objetivo, bastando apenas que todos os subscritores conseguissem se conectar ao *broker* do servidor hospedado na nuvem para haver a troca de informações. Ao passo que as mensagens foram publicadas nos tópicos pré-determinados as respostas tanto da aplicação web quanto do microcontrolador foram instantâneas. Somente o acendimento do LED não é instantâneo pois depende do *delay* configurado no *loop* do código do NodeMCU, então leva cerca de 5 segundos para ser aceso. Em termos de robustez e velocidade, o servidor Mosquitto MQTT instalado se encaixa muito bem para a aplicação do estacionamento inteligente. Isto pode ser comprovado observando os testes feitos pelo órgão mqtt.org em conjunto com a empresa ScalAgent, onde o uso de CPU para um número de publicadores entre 1 e 10000 fica na faixa de 0% à 50% enquanto sua latência para o mesmo número de publicadores é na faixa de 4 à 60 ms [38].

Estes dados podem ser observados nos gráficos da Figura 4.1 e 4.2 a seguir.

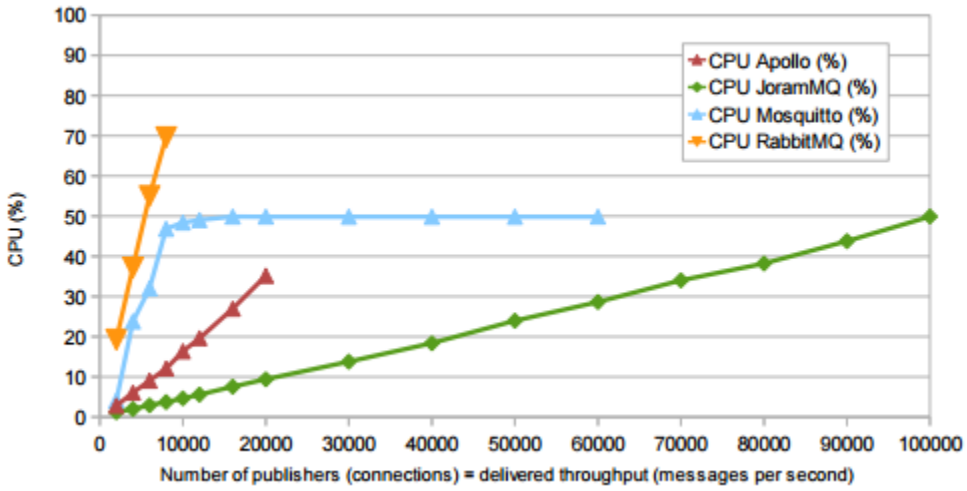

Figura 4.1 - Comparação entre servidores MQTT: Porcentagem de uso da CPU [38]

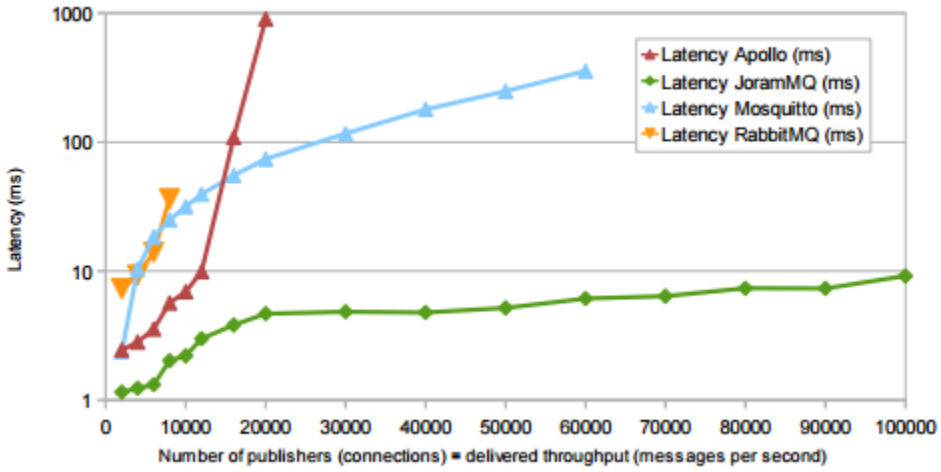

Figura 4.2 - Comparação entre servidores MQTT: Latência [38]

Como neste protótipo foi-se utilizado apenas um nó publicador, o servidor escolhido atende muito bem às necessidades do projeto e mostra que é possível escalar a aplicação para até 60000 nós publicadores com um tempo abaixo de 1 segundo de latência e uma eficiência quanto ao uso da CPU de 50%.

O desenvolvimento da aplicação web foi a etapa de maior dificuldade, dado que o conhecimento prévio das linguagens de programação PHP, SQL, JavaScript e HTML5 era limitado. Entretanto este obstáculo foi superado com o acesso à diversas aulas online, tutoriais e fóruns, os quais proporcionaram um aprendizado rápido e efetivo para a realização das tarefas que esta etapa demandava.

A escolha do QR Code para a confirmação das vagas se deu pelo fato de ser uma abordagem mais simples, de fácil implementação e custo baixo, uma vez de que não foi necessário a compra de nenhum equipamento extra para realizar a leitura do mesmo. Existem outros métodos onde seria possível fazer o mesmo tipo de abordagem do QR Code, como por exemplo os *Beacons* (dispositivos que utilizam Bluetooth *Low Energy* que se comunicam com smartphones e servem como GPS *indoor*), porém seria necessário um investimento razoavelmente alto para realização da solução completa (custam em torno de 12 dólares cada peça).

A escolha da linguagem PHP para a programação do lado do servidor se deu pela simplicidade da linguagem e pela semelhança que ela possui com a linguagem C, a qual foi aprendida durante a graduação. Por ser uma linguagem de *backend* (atua do lado do servidor apenas) no decorrer do desenvolvimento do projeto surgiram dificuldades para a execução de

alguns scripts no *frontend* (atua do lado do cliente) e por isso alguns blocos de código, como o leitor e interpretador de QR Code, demandaram a uso do JavaScript, linguagem que executa scripts do lado do cliente sem passar pela interpretação do servidor.

O conceito de *Cloud* não foi explorado em sua totalidade neste trabalho, sendo utilizado apenas a ferramenta de *storage* e criação de máquinas virtuais. Porém foi importante para se ter uma ideia de como o conceito funciona em uma aplicação real de Internet das Coisas.

O banco de dados MySQL também se mostrou eficiente para o projeto. O tempo de resposta das *queries* (consultas em português) está na faixa de 100 a 200 µs de acordo com a tabela acessada. Este tempo foi medido acessando o gerenciador phpMyAdmin, que mostra o resultado das *queries* em sua interface e está exemplificado na Figura 4.3 a seguir.

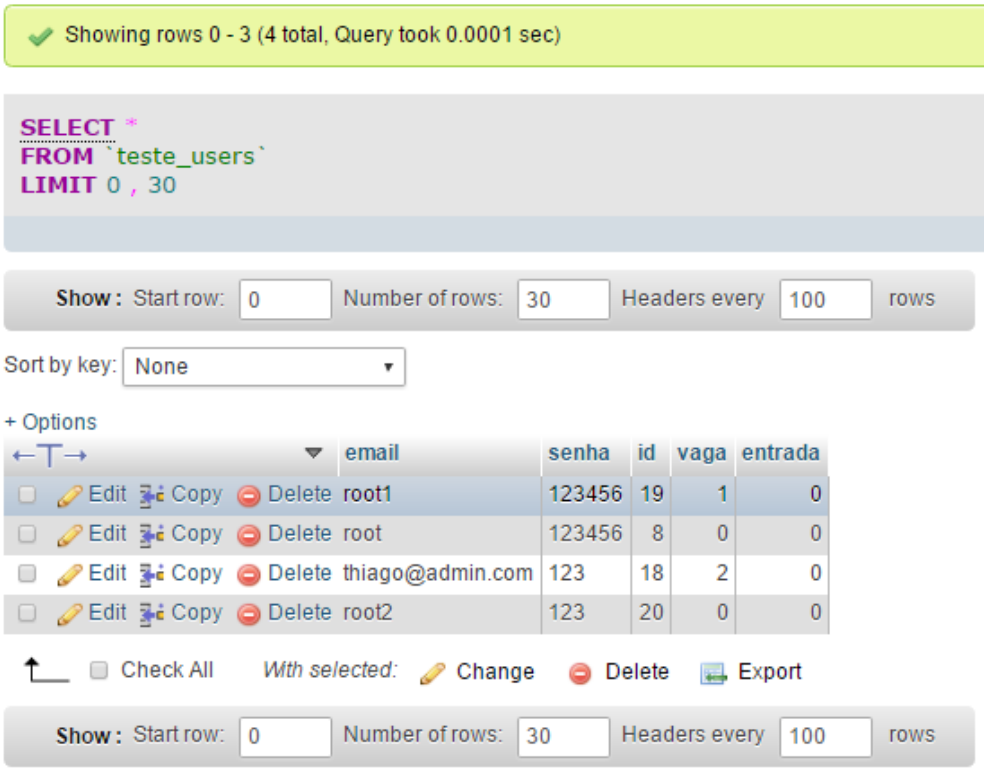

Figura 4.3 - Tempo de consulta ao MySQL mostrado no phpMyAdmin

O MySQL também fornece grande espaço de armazenamento. Segundo seu próprio manual o tamanho máximo de uma linha da tabela é de 65535 bytes e o tamanho máximo de uma tabela em um sistema operacional Linux 2.4+ é de até 4 TB [39]. Este protótipo não exigiu um grande

volume de armazenamento uma vez que em três de suas quatro tabelas utilizadas, todas as informações salvas, ao invés de se criar um novo registro, foram atualizadas conforme ocorriam as alterações.

Foram feitos testes para saber o tempo de resposta da aplicação web em diferentes ambientes de rede: rede WiFi com Internet de 120 MB e 4G. Em ambos os casos foram medidos os tempos das três páginas principais da aplicação, que podem ser verificados na Tabela 7 a seguir.

Tabela 7 - Comparação do tempo de resposta em diferentes ambientes de rede

WiFi 120MB	
Página Inicial	7 requests | 4.4 KB transferred | Finish: 1.60 s | DOMContentLoaded: 1.17 s | Load: 1.61 s
Reserva de vagas	9 requests | 39.5 KB transferred | Finish: 1.43 s | DOMContentLoaded: 761 ms | Load: 1.44 s
Minha conta	8 requests | 19.0 KB transferred | Finish: 1.63 s | DOMContentLoaded: 1.73 s | Load: 1.73 s

4G	
Página Inicial	7 requests | 3.5 KB transferred | Finish: 1.50 s | DOMContentLoaded: 1.52 s | Load: 1.52 s
Reserva de vagas	9 requests | 4.5 KB transferred | Finish: 2.83 s | DOMContentLoaded: 2.34 s | Load: 2.83 s
Minha conta	8 requests | 20.1 KB transferred | Finish: 1.22 s | DOMContentLoaded: 1.10 s | Load: 1.44 s

Em termos de hardware o NodeMCU se mostrou uma boa ferramenta para aplicações para Internet das Coisas. Primeiramente pela sua versatilidade, podendo ser empregado em diversas aplicações. Seu tamanho reduzido também é um diferencial se for comparado com os tamanhos de outros microcontroladores, tais como o próprio Arduino, Intel Galileo e Raspberry Pi. Quanto ao seu consumo de energia o NodeMCU carrega um módulo Wi-Fi ESP12-E que consome corrente em torno de 170mA quando está em modo de transmissão contínua e em torno de 56mA quando está em modo de recepção contínua [40]. Para habilitar o módulo para aplicações de IoT utilizando bateria é necessário uma redução dessa corrente e para isso pode-se empregar os modos *Modem-Sleep*, onde a CPU continua trabalhando mas corta a conexão do circuito do modem Wi-Fi quando não há transmissão, *Light-Sleep*, onde CPU e conexão do circuito do modem Wi-Fi são suspensas, e *Deep-Sleep*, que estabelece conexão apenas quando há necessidade de transmissão de dados

(programável). [40] A seguir na Tabela 8 são apresentados os dados relativos ao consumo do módulo Wi-Fi.

Tabela 8 - Consumo de energia do módulo Wi-Fi ESP12-E [39]

Parameters	Min	Typical	Max	Unit
Tx802.11b, CCK 11Mbps, P OUT=+17dBm		170		mA
Tx 802.11g, OFDM 54Mbps, P OUT =+15dBm		140		mA
Tx 802.11n, MCS7, P OUT =+13dBm		120		mA
Rx 802.11b, 1024 bytes packet length , -80dBm		50		mA
Rx 802.11g, 1024 bytes packet length, -70dBm		56		mA
Rx 802.11n, 1024 bytes packet length, -65dBm		56		mA
Modem-Sleep①		15		mA
Light-Sleep②		0.9		mA
Deep-Sleep③		10		uA

Analisando a Tabela 8 pode-se dizer que o modo *Deep-Sleep* é o mais indicado para as aplicações em IoT com alimentação à bateria, uma vez que consome pouca corrente.

Se fizermos uma análise hipotética onde alimentamos o sistema com uma bateria alcalina *long-life,* que possui capacidade energética de 2,122 Ah, e realizarmos as contas necessárias para saber o tempo de duração dessa pilha considerando o modo *Deep-Sleep,* verifica-se que o tempo total seria de aproximadamente 24 anos.

Capítulo 5

Conclusões

Ao se analisar os objetivos propostos no início deste trabalho, seu desenvolvimento e os resultados obtidos ao término do mesmo, pode-se concluir que um protótipo funcional foi criado e que este pode ser escalado e melhorado tanto para trabalhos futuros quanto para a implantação de um produto comercial.

O sucesso na recriação de uma arquitetura completa de IoT desde sua camada de sensoriamento, passando pela camada de gerenciamento, até chegar a uma camada de aplicação através do desenvolvimento do projeto do Controle Inteligente de Vagas para Estacionamento trouxe um entendimento mais aprofundado sobre como funciona o conceito de Internet das Coisas na prática, além de expor as ferramentas mais importantes para tornar este conceito funcional.

O ponto alto deste projeto foi a integração da camada de sensoriamento com a aplicação web, onde ficou evidente a facilidade que o protocolo MQTT traz para as aplicações em IoT com seu modelo de publicação e subscrição na troca de dados. Além disso a utilização, mesmo que parcial, do conceito de *Cloud computing* mostrou a infinita gama de ferramentas que este serviço proporciona e o quanto fortalece e complementa o conceito de Internet das Coisas.

A criação da aplicação web desde sua concepção até seu funcionamento pleno foi de grande aprendizado, uma vez que foi preciso o desenvolvimento de habilidades em três diferentes linguagens de programação, HTML5, PHP e JavaScript, das quais havia conhecimento limitado. Além disso, este trabalho proporcionou a interação com uma plataforma de hardware bem difundida no mundo de IoT que é o NodeMCU, plataforma essa que proporciona uma arquitetura

que atende à leitura de sensores e permite a criação de diversas aplicações, sejam elas para automação ou monitoramento.

Dificuldades surgiram ao longo do trabalho, principalmente na hora de integrar as duas fases do protótipo, mas foram superadas realizando diversos testes, *troubleshooting*, consultando fontes de informação e sem dúvida com persistência e determinação.

O conhecimento gerado a partir do desenvolvimento deste trabalho é de grande valia pois não cairá no esquecimento visto que é utilizado no dia-a-dia profissional, e será aperfeiçoado daqui para frente.

Ao final pode-se concluir que este trabalho conseguiu demonstrar um pouco da visão do funcionamento da Internet das Coisas e como ele pode ser empregado em nosso dia-a-dia por meio de diversas aplicações.

5.1. Sequência do Trabalho

Como este trabalho é apenas um protótipo de um Controle Inteligente de Vagas para Estacionamento, isto abre margem para uma sequência de evoluções que o mesmo pode sofrer caso venha a ser implementado comercialmente.

Dentre essas evoluções pode-se destacar o aumento da rede de sensores e o gerenciamento de seus dados; a preocupação com segurança, visto que uma série de dispositivos conectados à Internet abre espaço para *hackers* (pessoas de má índole que se utilizam da Internet para cometer crimes) se utilizarem dos mesmos como *bots* para a realização de ataques; desenvolvimento de um modelo de negócio caso o serviço de Controle Inteligente de Vagas para Estacionamento venha a se tornar um produto comercial.

Referências Bibliográficas

[1] D. Evans, “A Internet das Coisas. Como a próxima evolução da internet está mudando tudo. Cisco.,” Abril 2011. [Online]. Available: http://www.cisco.com/web/BR/assets/executives/pdf/internet_of_things_iot_ibsg_0411final.pdf. [Acesso em 12 Junho 2016].

[2] R. v. d. Meulen, "Gartner.com," Gartner Inc., 10 Novembro 2015. [Online]. Available: http://www.gartner.com/newsroom/id/3165317. [Accessed 3 Outubro 2016].

[3] R. V. d. M. Amy Ann Forni, "Gartner Inc," [Online]. Available: http://www.gartner.com/newsroom/id/3412017. [Accessed 27 Outubro 2016].

[4] W. Na, "Internet of Things based on the Cloud Computing Architecture," *2015 Seventh International Conference on Measuring Technology and Mechatronics Automation*, p. 585, 2015.

[5] C. Dores, L. P. Reis and N. Vasco Lopes, "Internet of Things and Cloud Computing," *Information Systems and Technologies (CISTI), 2014 9th Iberian Conference on*, p. 2, 2014.

[6] D. Canellos, "CSA - Cloud Security Alliance," 5 Junho 2013. [Online]. Available: https://blog.cloudsecurityalliance.org/2013/06/05/how-the-internet-of-things-will-feed-cloud-computings-next-evolution/. [Accessed 3 Outubro 2016].

[7] J. C. L. d. Silva, "Brasil Escola," [Online]. Available: http://brasilescola.uol.com.br/geografia/por-que-brasil-adotou-utilizacao-das-rodovias-ao-inves-.htm. [Accessed 11 Junho 2016].

[8] J. M. Rodrigues, "Observatório das Metrópoles, Instituto Nacional de Ciência e Tecnologia," [Online]. Available: http://www.observatoriodasmetropoles.net/index.php?option=com_content&view=article&id=1772%3Acrise-de-mobilidade-urbana-brasil-atinge-marca-de-50-milhoes-de-automoveis&catid=34%3Aartigos&Itemid=124&lang=pt. [Accessed 11 Junho 2016].

[9] F. Wanpeng and L. Yu, "Opportunities, Challenges and Practices of the Internet of Things," ZTE, 10 Maio 2010. [Online]. Available: http://wwwen.zte.com.cn/endata/magazine/ztetechnologies/2010/no5/articles/201005/t20100510_184418.html. [Accessed 05 Outubro 2016].

[10] IBM, "IBM Cloud," [Online]. Available: https://www.ibm.com/cloud-computing/what-is-cloud-computing. [Accessed 08 Outubro 2016].

[11] X. Wang, B. Wang and J. Huang, "Cloud computing and its key techniques," *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on*, pp. 404-410, 2011.

[12] "salesforce.com," [Online]. Available: https://www.salesforce.com. [Accessed 08 Outubro 2016].

[13] "Microsoft Azure," [Online]. Available: https://azure.microsoft.com/. [Accessed 08 Outubro 2016].

[14] "Google App Engine," [Online]. Available: https://appengine.google.com/. [Accessed 08 Outubro 2016].

[15] "Amazon Web Services," [Online]. Available: https://aws.amazon.com/console/. [Accessed 27 Outubro 2016].

[16] C. Karasiewicz, "IBM developerWorks," 09 Setembro 2013. [Online]. Available: https://www.ibm.com/developerworks/community/blogs/mobileblog/entry/why_http_is_not_enough_for_the_internet_of_things?lang=en. [Accessed 09 Outubro 2016].

[17] A. Foster, "Messaging Technologies for the Industrial Internet and the Internet of Things," 25 Novembro 2013. [Online]. Available: http://www.prismtech.com/sites/default/files/documents/MessagingComparsionNov2013USROW_vfinal.pdf. [Accessed 10 Outubro 2016].

[18] J. Stansberry, "The IoT communication protocols," 16 Outubro 2015. [Online]. Available: https://www.linkedin.com/pulse/iot-communication-protocols-james-stansberry. [Accessed 10 Outubro 2016].

[19] M. E. Software, "Micrium.com - Designing the Internet of Things," [Online]. Available: https://www.micrium.com/iot/internet-protocols/. [Accessed 10 Outubro 2016].

[20] S. Nicholas, "stephendnicholas.com," 31 Maio 2012. [Online]. Available: http://stephendnicholas.com/posts/power-profiling-mqtt-vs-https. [Accessed 12 Outubro 2016].

[21] I. Harris, "Coursera - Introduction to the Internet of Things and Embedded Systems - UC Irvine," [Online]. Available: https://www.coursera.org/learn/iot/lecture/RnvF2/lecture-1-2-more-on-embedded-systems. [Accessed 25 Outubro 2016].

[22] Techopedia.com, "Techopedia.com," [Online]. Available: https://www.techopedia.com/definition. [Accessed 27 Outubro 2016].

[23] Arduino, "Arduino Genuino," [Online]. Available: www.arduino.cc. [Accessed 27 Outubro 2016].

[24] "Raspberry Pi," [Online]. Available: https://www.raspberrypi.org/. [Accessed 27 Outubro 2016].

[25] "BeagleBoard.org," [Online]. Available: https://beagleboard.org/black. [Accessed 27 Outubro 2016].

[26] CriticalLink.com, "CriticalLink.com," [Online]. Available: http://www.criticallink.com/system-on-module/. [Accessed 27 Outubro 2016].

[27] "NodeMCU Documentation," [Online]. Available: https://nodemcu.readthedocs.io/en/master/. [Accessed 27 Outubro 2016].

[28] "Toradex.com," [Online]. Available: https://www.toradex.com/computer-on-modules/colibri-arm-family. [Accessed 27 Outubro 2016].

[29] "Intel Developer Zone," [Online]. Available: https://software.intel.com/en-us/iot/hardware/edison. [Accessed 27 Outubro 2016].

[30] I. Pankiewicz, "Tecmundo," 23 Abril 2009. [Online]. Available: https://www.tecmundo.com.br/imagem/1995-o-que-sao-os-qr-codes-.htm. [Accessed 30 Novembro 2016].

[31] "Ubuntu," [Online]. Available: https://www.ubuntu.com/. [Accessed 31 Outubro 2016].

[32] "PuTTy," [Online]. Available: http://www.putty.org/. [Accessed 27 Outubro 2016].

[33] "FileZilla," [Online]. Available: https://filezilla-project.org/. [Accessed 27 Outubro 2016].

[34] "Spark Fun Electronics," [Online]. Available: https://www.sparkfun.com/products/13678. [Accessed 12 Junho 2016].

[35] "Seeed Studio," [Online]. Available: https://www.seeedstudio.com/NodeMCU-v2---Lua-based-ESP8266-development-kit-p-2415.html. [Accessed 12 Outubro 2016].

[36] "FilipeFlop Componentes Eletrônicos," [Online]. Available: http://www.filipeflop.com/. [Accessed 12 Junho 2016].

[37] "How to Mechatronics," [Online]. Available: http://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/. [Accessed 12 Outubro 2016].

[38] S. Mqtt.org, "Benchmark of MQTT servers," 2015.

[39] Oracle, "MySQL Documentation," [Online]. Available: http://dev.mysql.com/doc/. [Accessed 17 Outubro 2016].

[40] A.-T. Team, "ESP-12E Wi-fi Module," AI-Thinker, 2015. [Online]. Available: http://www.kloppenborg.net/images/blog/esp8266/esp8266-esp12e-specs.pdf. [Accessed 25 Outubro 2016].

Apêndice A

Códigos criados para o projeto

A.1. Código NodeMCU

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Ultrasonic.h>

//Define os pinos para o trigger e echo
#define pino_trigger D1
#define pino_echo D2

//Inicializa o sensor nos pinos definidos acima
Ultrasonic ultrasonic(pino_trigger, pino_echo);

int maximumRange = 200;
int minimumRange = 0;
int threshold = 10;
long duration, distance;
int presenca;

char buffer[25];
char message_buff[100];

const char* ssid = "ssid"; //defina o ssid da rede
const char* password = "password"; //defina a senha de acesso à rede
const char* mqtt_server = "mqtt.server.address"; //defina o ip do mqtt server

WiFiClient espClient;
PubSubClient client(espClient);


long lastMsg = 0;
char msg1[50];
char msg2[50];
int value = 0;
```

```
//-----------------------------------------------------------------------------------------------------------------

void setup() {
Serial.begin(9600);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  pinMode(D7, OUTPUT);

  digitalWrite(D7,HIGH);
}

//-----------------------------------------------------------------------------------------------------------------

void setup_wifi() {

  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

   if (client.connect("tcc1")) {
    client.publish("outTopic","Conectado!");
    client.subscribe("vaga1/erro");
  }
}


//-----------------------------------------------------------------------------------------------------------------


void callback(char* topic, byte* payload, unsigned int length) {

    int i = 0;

  for(i=0; i<length; i++) {
    message_buff[i] = payload[i];
  }
  message_buff[i] = '\0';

  String msgString = String(message_buff);
  //Serial.println("Payload: " + msgString);

  float erro = msgString.toFloat();
  //Serial.println(erro);
  if(erro==1){
```

```
    digitalWrite(D7,LOW);
    delay(5000);
    digitalWrite(D7,HIGH);
    //int porta = digitalRead(D5);
    Serial.println("deu certo");
  }
  }

//-----------------------------------------------------------------------------------------------------------------

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");

    if (client.connect("ESP8266Client")) {
      Serial.println("connected");

      client.publish("outTopic", "hello world");

      client.subscribe("vaga1/erro");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");

      delay(5000);
    }
  }
}

//-----------------------------------------------------------------------------------------------------------------


void loop() {

if (!client.connected()) {
   reconnect();
}
client.loop();

 //Le as informacoes do sensor, em cm e pol
  float cmMsec, inMsec;
  long microsec = ultrasonic.timing();
  cmMsec = ultrasonic.convert(microsec, Ultrasonic::CM);
  inMsec = ultrasonic.convert(microsec, Ultrasonic::IN);

  if (cmMsec < 10){
    presenca = 1;
    //Serial.println(presenca);
  }

  if (cmMsec > 10){
    presenca = 0;
    //Serial.println(presenca);
  }

//Exibe informacoes no serial monitor
```

```
  Serial.print("Distancia em cm: ");
  Serial.print(cmMsec);
  Serial.print(" - Distancia em polegadas: ");
  Serial.println(inMsec);
  Serial.println(presenca);
 delay(1000);

 //String aux1 = "{\"presenca\":[{\"value\": \"" + String(floatToString(buffer, presenca, 1, 0) ) + "\"}]}";
String aux1=String(floatToString(buffer, presenca, 1, 0) );
 aux1.toCharArray(msg1, aux1.length()+1);

 String aux2 = "{\"distancia\":[{\"value\": \"" + String(floatToString(buffer, cmMsec, 1, 0) ) + "\"}]}";
 aux2.toCharArray(msg2, aux2.length()+1);




 delay(50);

  long now = millis();
  if (now - lastMsg > 2000) {
    lastMsg = now;
    ++value;

    Serial.print("Publish message: ");
    Serial.println(msg1);
    Serial.print("Publish message: ");
    Serial.println(msg2);

    client.publish("vaga1/presenca", msg1);
    client.publish("vaga1/distancia", msg2);
  }

  delay(5000);
}

//--------------------------------------------------------------------------------------------------------------

// Coverter FLOAT para String
char * floatToString(char * outstr, double val, byte precision, byte widthp){
 char temp[16];
 byte i;


 double roundingFactor = 0.5;
 unsigned long mult = 1;
 for (i = 0; i < precision; i++)
 {
   roundingFactor /= 10.0;
   mult *= 10;
 }

 temp[0]='\0';
 outstr[0]='\0';

 if(val < 0.0){
   strcpy(outstr,"-\0");
   val = -val;
```

```
}

val += roundingFactor;

strcat(outstr, itoa(int(val),temp,10));  //prints the int part
if( precision > 0) {
  strcat(outstr, ".\0"); // print the decimal point
  unsigned long frac;
  unsigned long mult = 1;
  byte padding = precision -1;
  while(precision--)
    mult *=10;

  if(val >= 0)
    frac = (val - int(val)) * mult;
  else
    frac = (int(val)- val ) * mult;
  unsigned long frac1 = frac;

  while(frac1 /= 10)
    padding--;

  while(padding--)
    strcat(outstr,"0\0");

  strcat(outstr,itoa(frac,temp,10));
}


if ((widthp != 0)&&(widthp >= strlen(outstr))){
  byte J=0;
  J = widthp - strlen(outstr);

  for (i=0; i< J; i++) {
    temp[i] = ' ';
  }

  temp[i++] = '\0';
  strcat(temp,outstr);
  strcpy(outstr,temp);
}

return outstr;
}
```

A.2. Código para página inicial da aplicação web

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
    <meta name="description" content="">
    <meta name="author" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
    img {
       max-width: 70%;
       height: auto;
    }
    </style>

    <link rel="icon" href="img/img.png">

    <title>Estacionamento Inteligente</title>

    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">

    <!-- Optional theme -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-theme.min.css" integrity="sha384-fLW2N01lMqjakBkx3l/M9EahuwpSfeNvV63J5ezn3uZzapT0u7EYsXMjQV+0En5r" crossorigin="anonymous">

  </head>

  <body>

    <nav class="navbar navbar-inverse navbar-fixed-top">
      <div class="container">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" href="qr.html">Estacionamento Inteligente</a>
          <a href="#" class="navbar-left"><img src="img/img.png" width="50px" height="50px" style="margin-top:8px" ></a>

        </div>
        <div id="navbar" class="collapse navbar-collapse">
          <ul class="nav navbar-nav">
            <li class="active"><a href="index.html">Home</a></li>
            <li><a href="loginMA.html">Minha Conta</a></li>
            <li><a href="about.html">Sobre</a></li>
```

```
        </ul>
      </div><!--/.nav-collapse -->
    </div>
  </nav>

  <div class="container" style="padding-top: 80px">

      <div class="starter-template text-center">
        <h1 style="font-size:300%;">Estacionamento Inteligente</h1><br><br><br><br>
        <img src="img/img.png" width="200px" height="200px"><br><br><br><br><br><br>
        <p class="lead">Um jeito fácil e prático para achar aquela vaga<br> nas horas mais complicadas do dia.</p>
      </div><br><br><br>

    <div class="btn-group btn-group-justified" role="group" aria-label="...">
      <div class="btn-group" role="group">
        <button id="btn-reservar" type="button" class="btn btn-default" onclick="document.getElementById('registro').style.display='block';
document.getElementById('cadastro').style.display='none';                    document.getElementById('central').style.display='none';
document.getElementById('success').style.display='none';">Reservar Vaga</button>
      </div>

      <div class="btn-group" role="group">
        <button id="btn-cadastrar" type="button" class="btn btn-default" onclick="document.getElementById('cadastro').style.display='block';
document.getElementById('registro').style.display='none';                    document.getElementById('central').style.display='none';
document.getElementById('success').style.display='none';">Cadastrar</button>
      </div>

      <div class="btn-group" role="group">
        <button id="btn-central" type="button" class="btn btn-default" onclick="document.getElementById('central').style.display='block';
document.getElementById('cadastro').style.display='none';                    document.getElementById('registro').style.display='none';
document.getElementById('success').style.display='none';">Contactar Central</button>
      </div>
    </div>

    <!-- Cadastro -->
    <div class="input-group" id="cadastro" style="display:none">
      <br><br>
      <h3>Cadastro: </h3>
      <form name="signup" method="POST" action="cadastro1.php">
      <p style="font-size:100%;">Insira um usuário, uma senha e clique no botão "Salvar" </p>
      <input type="text" class="form-control" placeholder="Usuário" aria-describedby="basic-addon1" name="login">
      <br><br>
      <input type="password" class="form-control" placeholder="Senha" aria-describedby="basic-addon2" name="senha">
      <br><br><br>
      <!-- Botao checar se o usuario foi salvo no banco de dados - chama funcao para mostrar ok-->
      <button value="cadastrar" type="submit" class="btn btn-default btn-lg" onclick="document.getElementById('success').style.display='block';
document.getElementById('central').style.display='none';                    document.getElementById('cadastro').style.display='none';
document.getElementById('registro').style.display='none';">
        <span class="glyphicon glyphicon-thumbs-up" aria-hidden="true"></span> Cadastrar
      </button>
      </form>
    </div>

    <div class="lead text-center" id="success" style="display:none">
      <br><br><p>Seu cadastro foi efetuado com sucesso!</p>
      <br>
      <img src="img/check.png" width="100px"><br>
    </div>

    <!-- Login -->
```

```
    <div class="input-group" id="registro" style="display:none">
      <br><br>
      <h3>Login: </h3>
            <form name="userlogin" method="POST" action="login1.php">
            <p style="font-size:100%;">Insira seu login, senha e clique no botão "Próximo" </p>
      <input type="text" class="form-control" placeholder="Login" aria-describedby="basic-addon1" name="login">
      <br><br>
      <input type="password" class="form-control" placeholder="Senha" aria-describedby="basic-addon2" name="senha">
      <br><br><br>
      <!-- Botao checar se o usuario consta no banco de dados e se os dados batem - chama funcao para proxima pagina-->
      <button value="entrar" type="submit" class="btn btn-default btn-lg">
        <span class="glyphicon glyphicon-triangle-right" aria-hidden="true"></span> Próximo
      </button>
    </div>

    <!-- Ligar Central -->
    <div class="lead text-center" id="central" style="display:none">
      <br><br><p>Telefone: (11) 9999-9999</p>
      <br>
      <button id="btn-ligarCentral" type="button" class="btn btn-default btn-lg">
        <span class="glyphicon glyphicon-phone-alt" aria-hidden="true"></span><a href="tel:11-9999-9999"> Ligar</a>
            </button>
    </div>

    <br><br><br><br><br><p class="lead text-center" style="font-size:100%;">Copyright Thiago Romanelli WebDesign ®<br> Since 2016</p>
  </div><!-- /.container -->


  <!-- Bootstrap core JavaScript
  ================================================== -->
  <!-- Placed at the end of the document so the pages load faster -->
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
  <script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery.min.js"><\/script>')</script>
  <script src="js/jquery.rwdImageMaps.min.js"></script>
  <script>
     $(document).ready(function(e) {
  $('img[usemap]').rwdImageMaps();
  });
  </script>
  <!--<script src="../../dist/js/bootstrap.min.js"></script>-->
 </body>
</html>
```

A.3. Código para cadastro de usuários

```
<html>
<head>
<title>Cadastrando...</title>
<script type="text/javascript">
function subscribe_success(){
		alert("Cadastro efetuado com sucesso!")
		setTimeout("window.location='index.html'",3000);
}

</script>
</head>

<body>

<?php
$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());
?>

<?php
$login=$_POST['login'];
$senha=$_POST['senha'];
$sql=mysql_query("INSERT INTO teste_users(email, senha) VALUES('$login','$senha')");
echo "<script>subscribe_success()</script>";
?>
<br><br><br><center><img src="loading.gif" width="100px"></center><br>
</body>
</html>
```

A.4. Código para login de usuários

```
<html>
<head>
<title>Logando...</title>
<script type="text/javascript">
function login_success(){
		alert("Voce foi autenticado com sucesso! Aguarde um instante enquanto redirecionamos voce para pagina destino...");
		setTimeout("window.location='painel.php'",2000);
}

function login_failed(){
		alert("Usuario ou senha incorretos. Aguarde enquanto redirecionamos voce para a pagina de login...");
		setTimeout("window.location='index.html'",2000);
}
</script>
</head>
```

```
<body>

<?php
$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());
?>

<?php
$login=$_POST['login'];
$senha=$_POST['senha'];
$sql=mysql_query("SELECT * FROM teste_users WHERE email = '$login' and senha = '$senha'") or die(mysql_error());
$row = mysql_num_rows($sql);
if($row > 0){
        session_start();
        $_SESSION['login']=$_POST['login'];
        $_SESSION['senha']=$_POST['senha'];
        echo "<script>login_success()</script>";
        }

        else{
                echo "<script>login_failed()</script>";
        }
?>

<br><br><br><center><img src="img/loading.gif" width="100px"><br></center>
</body>
</html>
```

A.5. Código para logout de usuários

```
<?php
        session_start();
        session_destroy();
        header("Location: index.html");
?>
```

A.6. Código para página de reserva de vagas

```
<?php
   $host = "localhost";
   $user = "root";
   $pass = "123456";
   $banco = "usuarios";
   $conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
   mysql_select_db($banco) or die(mysql_error());
   ?>
```

```
  <?php
    session_start();
    if(!isset($_SESSION["login"]) || !isset($_SESSION["senha"])){
      header("Location: loginMA.html");
      exit;
    } else {
      echo "voce esta logado!";
    }
  ?>

<?php

function parkingCapacity(){

 //vagas reservadas
 $sql11 = mysql_query("SELECT * FROM teste_users WHERE vaga = 1") or die(mysql_error());
 $status11 = mysql_num_rows($sql11);
 if($status11 >0){
  $um = 1;
  } else {
    $um = 0;
 }
  $sql22 = mysql_query("SELECT * FROM teste_users WHERE vaga = 2") or die(mysql_error());
  $status22 = mysql_num_rows($sql22);
  if($status22 >0){
  $dois = 1;
  } else {
    $dois = 0;
 }
  $sql33 = mysql_query("SELECT * FROM teste_users WHERE vaga = 3") or die(mysql_error());
  $status33 = mysql_num_rows($sql33);
  if($status33 >0){
  $tres = 1;
  } else {
    $tres = 0;
 }

 //vagas ocupadas
 $sql1 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 1") or die(mysql_error());
 $status1 = mysql_fetch_row($sql1);
 if($status1[1] == 1){
  $umOcup = 1;
 } else if ($status1[1] == 1){
  $umOcup = 0;
 }

 $sql2 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 2") or die(mysql_error());
 $status2 = mysql_fetch_row($sql2);
 if($status2[1] == 1){
  $doisOcup = 1;
 } else if ($status2[1] == 1){
  $doisOcup = 0;
 }

 $sql3 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 3") or die(mysql_error());
 $status3 = mysql_fetch_row($sql3);
 if($status3[1] == 1){
  $tresOcup = 1;
```

```
 } else if ($status3[1] == 1){
  $tresOcup = 0;
 }

 if($um == 0 and $umOcup == 0){
  $statusVaga1 = "Livre";
 } else if ($um == 1 and $umOcup == 0){
  $statusVaga1 = "Reservada";
 } else if ($um == 1 and $umOcup == 1){
  $statusVaga1 = "Ocupada";
 } else if ($um == 0 and $umOcup == 1){
  $statusVaga1 = "Erro!";
 }

 if($dois == 0 and $doisOcup == 0){
  $statusVaga2 = "Livre";
 } else if ($dois == 1 and $doisOcup == 0){
  $statusVaga2 = "Reservada";
 } else if ($dois == 1 and $doisOcup == 1){
  $statusVaga2 = "Ocupada";
 } else if ($dois == 0 and $doisOcup == 1){
  $statusVaga2 = "Erro!";
 }

 if($tres == 0 and $tresOcup == 0){
  $statusVaga3 = "Livre";
 } else if ($tres == 1 and $tresOcup == 0){
  $statusVaga3 = "Reservada";
 } else if ($tres == 1 and $tresOcup == 1){
  $statusVaga3 = "Ocupada";
 } else if ($tres == 0 and $tresOcup == 1){
  $statusVaga3 = "Erro!";
 }

 $total = 3;
 $somaReserva = $um + $dois + $tres;
 $somaOcup = $umOcup + $doisOcup + $tresOcup;
 $percentReserva = $somaReserva / $total;
 $percentOcup = $somaOcup / $total;

 return array($somaReserva, $somaOcup, $percentReserva, $percentOcup, $statusVaga1, $statusVaga2, $statusVaga3);

}

 function spot1(){
  $login = $_SESSION["login"];
  $sql1 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 1") or die(mysql_error());
  $sql11 = mysql_query("SELECT * FROM teste_users WHERE vaga = 1") or die(mysql_error());
  $status1 = mysql_fetch_row($sql1);
  $status11 = mysql_num_rows($sql11);
  if($status1[1] == 1 or $status11 > 0){
   //$_SESSION['erro1'] = "1";
   $msg1 = "Vaga ja esta reservada! Por favor escolha outra ou volte mais tarde :)";
   echo '<script type="text/javascript">alert("' . $msg1 . '")</script>';
  }
  if($status1[1] == 0 and mysql_num_rows($sql11) == 0){
   //$_SESSION['erro1'] = "0";
   $sql=mysql_query("UPDATE teste_users SET vaga='1' WHERE email='$login'") or die(mysql_error());
```

```
    $msg1 = "Vaga reservada com sucesso!";
    echo '<script type="text/javascript">alert("' . $msg1 . '")</script>';
    header( "refresh:1; url=myaccount.php" );
   }
}

function spot2(){
  $login = $_SESSION["login"];
  $sql2 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 2") or die(mysql_error());
  $sql22 = mysql_query("SELECT * FROM teste_users WHERE vaga = 2") or die(mysql_error());
   $status2 = mysql_fetch_row($sql2);
   $status22 = mysql_num_rows($sql22);
   if($status2[1] == 1 or $status22 > 0){
    // $_SESSION['erro2'] = "1";
     $msg2 = "Vaga ja esta reservada! Por favor escolha outra ou volte mais tarde :)";
     echo '<script type="text/javascript">alert("' . $msg2 . '")</script>';
   }
   if($status2[1] == 0 and mysql_num_rows($sql22) == 0){
    // $_SESSION['erro2'] = "0";
     $sql=mysql_query("UPDATE teste_users SET vaga='2' WHERE email='$login'") or die(mysql_error());
     $msg2 = "Vaga reservada com sucesso!";
     echo '<script type="text/javascript">alert("' . $msg2 . '")</script>';
     header( "refresh:1; url=myaccount.php" );
   }
}

function spot3(){
   $login = $_SESSION["login"];
   $sql3 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 3") or die(mysql_error());
   $sql33 = mysql_query("SELECT * FROM teste_users WHERE vaga = 3") or die(mysql_error());
   $status3 = mysql_fetch_row($sql3);
   $status33 = mysql_num_rows($sql33);
   if($status3[1] == 1 or $status33 > 0){
    // $_SESSION['erro2'] = "1";
     $msg3 = "Vaga ja esta reservada! Por favor escolha outra ou volte mais tarde :)";
     echo '<script type="text/javascript">alert("' . $msg3 . '")</script>';
   }
   if($status3[1] == 0 and mysql_num_rows($sql33) == 0){
     //$_SESSION['erro3'] = "0";
     $sql=mysql_query("UPDATE teste_users SET vaga='3' WHERE email='$login'") or die(mysql_error());
     $msg3 = "Vaga reservada com sucesso!";
     echo '<script type="text/javascript">alert("' . $msg3 . '")</script>';
     header( "refresh:1; url=myaccount.php" );
   }
}

if(isset($_POST['post1'])){
       spot1();
}
if(isset($_POST['post2'])){
       spot2();
}
if(isset($_POST['post3'])){
       spot3();
}

$park = parkingCapacity();
$reservadas = $park[2]*100;
```

```
$ocupadas = $park[3]*100;

echo '<script type="text/javascript">console.log("' . $park[0] . ',' . $park[1] . ',' . $park[2] . ',' . $park[4] . ',' . $park[5] . ',' . $park[6] . '")</script>'
?>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
    <meta name="description" content="">
    <meta name="author" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
    img {
        width: 50%;
        height: auto;
    }
    </style>
    <link rel="icon" href="img/img.png">
    <title>Escolha de vagas</title>
    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-
1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">

    <!-- Optional theme -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-theme.min.css" integrity="sha384-
fLW2N01lMqjakBkx3l/M9EahuwpSfeNvV63J5ezn3uZzapT0u7EYsXMjQV+0En5r" crossorigin="anonymous">

<script type="text/javascript">
</script>
  </head>
  <body>

    <nav class="navbar navbar-inverse navbar-fixed-top">
      <div class="container">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-
controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" href="#">Estacionamento Inteligente</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
          <ul class="nav navbar-nav">
            <li><a href="index.html">Home</a></li>
            <li><a href="myaccount.php">Minha Conta</a></li>
            <li><a href="logout.php">Logout</a></li>
          </ul>
        </div><!--/.nav-collapse -->
      </div>
    </nav>
<div class="container" style="padding-top: 80px">
  <div class="starter-template text-center">
      <!-- Example row of columns -->
```

```
    <div class="row">
      <div class="col-md-4">
        <div class="panel panel-success">
                    <div class="panel-heading">
                      <h3 class="panel-title">Status das Vagas</h3>
                    </div>
                    <div class="panel-body">
                      <h4><?php echo "Vaga 1: $park[4]<br>Vaga 2: $park[5]<br>Vaga 3: $park[6]"?></h4>
                    </div>
                  </div>
      </div>
      <div class="col-md-4">
        <div class="panel panel-warning">
                    <div class="panel-heading">
                      <h3 class="panel-title">Vagas Reservadas</h3>
                    </div>
                    <div class="panel-body">
                      <div class="progress">
                        <div class="progress-bar progress-bar-warning" role="progressbar" aria-valuenow="<?php echo
htmlspecialchars($reservadas);?>" aria-valuemin="0" aria-valuemax="100" style="width: <?php echo
htmlspecialchars($reservadas);?>%"><span class="sr-only">80% Complete (danger)</span>
                        </div>
                      </div>
                    </div>
                  </div>
      </div>
      <div class="col-md-4">
        <div class="panel panel-danger">
                    <div class="panel-heading">
                      <h3 class="panel-title">Vagas Ocupadas</h3>
                    </div>
                    <div class="panel-body">
                      <div class="progress">
                        <div class="progress-bar progress-bar-danger" role="progressbar" aria-valuenow="<?php echo
htmlspecialchars($ocupadas);?>" aria-valuemin="0" aria-valuemax="100" style="width: <?php echo htmlspecialchars($ocupadas);?>%"><span
class="sr-only">80% Complete (danger)</span>
                        </div>
                      </div>
                    </div>
                  </div>
      </div>
    </div>
        <h1>Escolha sua vaga no mapa abaixo:</h1><br><br>
    <img id="parking" src="img/parkinglot.png" alt="parkinglot" usemap="#parkinglot"><br><br><br>
    <map name="parkinglot">
      <area name="post1" type="submit" shape="rect" coords="0,34,276,294"
onclick="document.getElementById('parking').style.display='none'; document.getElementById('spot1').style.display='block';"></area>
      <area name="post2" type="submit" shape="rect" coords="279,0,520,294"
onclick="document.getElementById('parking').style.display='none'; document.getElementById('spot2').style.display='block';"></area>
      <area name="post3" type="submit" shape="rect" coords="521,0,761,294"
onclick="document.getElementById('parking').style.display='none'; document.getElementById('spot3').style.display='block';"></area>
    </map>

    <form action="" method="post">
      <div id="vagaBtn" class="btn-group btn-group-lg" role="group" aria-label="...">
        <input id="campoVaga" type="text" value="1" hidden></input>
          <button name="post1" type="submit" class="btn btn-default" onclick="post(1);
document.getElementById('vagaBtn').style.display='none'; document.getElementById('parking').style.display='none';
document.getElementById('spot1').style.display='block';">Vaga 1</button>
```

```
        <button name="post2" type="submit" class="btn btn-default" onclick="document.getElementById('vagaBtn').style.display='none';
document.getElementById('parking').style.display='none'; document.getElementById('spot2').style.display='block';">Vaga 2</button>
        <button name="post3" type="submit" class="btn btn-default" onclick="document.getElementById('vagaBtn').style.display='none';
document.getElementById('parking').style.display='none'; document.getElementById('spot3').style.display='block';">Vaga 3</button>
      </div>
    </form>

    <div class="lead text-center" id="spot1" style="display:none">
      <img src="img/spot1.png"><br>
    </div>

    <div class="lead text-center" id="spot2" style="display:none">
      <img src="img/spot2.png"><br>
    </div>

    <div class="lead text-center" id="spot3" style="display:none">
      <img src="img/spot3.png"><br>
    </div>
</div>
</div>
    <footer class=bs-docs-footer>
      <br><br><br><br><br><p class="lead text-center" style="font-size:100%;">Copyright Thiago Romanelli WebDesign ®<br> Since
2016</p>
    </footer>

  <!-- Bootstrap core JavaScript
  ================================================== -->
  <!-- Placed at the end of the document so the pages load faster -->
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
  <script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery.min.js"><\/script>')</script>
  <script src="js/jquery.rwdImageMaps.min.js"></script>
  <script>
    $(document).ready(function(e) {
  $('img[usemap]').rwdImageMaps();
  });
  function post(vaga){
    console.log("teste")
    console.log($('#campoVaga').val())
  }
  </script>
  </body>
</html>
```

A.7. Código para página Minha Conta

```
<?php
$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());
?>
```

```
<?php
    session_start();
    $sucesso = $_SESSION['sucesso'];

    if(!isset($_SESSION["login"]) || !isset($_SESSION["senha"])){
      header("Location: login_myaccount.php");
      exit;
    } else {
      echo "voce esta logado!";
    }
?>

<?php
require("phpMQTT.php");

function publish(){
$mqtt = new phpMQTT("localhost", 1883, "phpMQTT Pub Example"); //Change client name to something unique
 if ($mqtt->connect()) {
   $mqtt->publish("vaga1/erro","1",0);
   $mqtt->close();
 }
}

function success($sucesso){
 switch($sucesso){
   case 0:
      $resultado = "time1";
      publish();
      $msg1 = "O usuario esta em uma vaga incompativel com sua reserva ou nao ha presenca de veiculo no local. Central esta sendo avisada.";
      echo '<script type="text/javascript">alert("' . $msg1 . '")</script>';

      break;
   case 1:
      $resultado = "time";
      $msg2 = "Verificacao realizada com sucesso!";
      echo '<script type="text/javascript">alert("' . $msg2 . '")</script>';
      break;
   default:
      echo "bla";
   }
 return $resultado;
}

function sec2hms ($sec, $padHours = true)
{
 $hours = intval(intval($sec) / 3600);
 if ($padHours) {
   $hms = str_pad($hours, 2, "0", STR_PAD_LEFT). ":";
 } else {
   $hms = $hours. ":";
 }
 $minutes = intval(($sec / 60) % 60);
 $hms .= str_pad($minutes, 2, "0", STR_PAD_LEFT). ":";
 $seconds = intval($sec % 60);
 $hms .= str_pad($seconds, 2, "0", STR_PAD_LEFT);

 return array($hms,$hours,$minutes,$seconds);
}
```

```
function display_exit(){

 $login = $_SESSION["login"];
 $sql=mysql_query("SELECT email,senha,id,vaga,entrada FROM teste_users WHERE email='$login'") or die (mysql_error());
 $row=mysql_fetch_row($sql);
 $nowtime1 = time();
 $exit = $nowtime1 - $row[4];
 $hms=sec2hms($exit);

 if($hms[2] > 6){
  $valor = 25;
 }

 else{
 $valor = $hms[2] * 3;
 }
 return array($hms[0],$valor,$exit,$row[4]);
}

function payparking(){
 $login = $_SESSION["login"];
 $payment=display_exit();
 $sql1=mysql_query("UPDATE teste_users SET vaga='0' WHERE email='$login'") or die(mysql_error());
 $sql2=mysql_query("UPDATE teste_users SET entrada='0' WHERE email='$login'") or die(mysql_error());
 $sql3=mysql_query("INSERT INTO teste_log(id, data, user, tempo, valor) VALUES (NULL, CURRENT_DATE(), '$login', '$payment[0]',
'$payment[1]')") or die(mysql_error());

    echo '<script type="text/javascript">alert("Seu tempo de estadia foi de ' . $payment[0] . ' e o valor a ser pago é de ' . $payment[1] . '. Clique em
OK para confirmar pagamento. ")</script>';
    header( "refresh:5; url=index.html" );
}
if(isset($_POST['saida'])){
      $valor1= display_exit();
      //$result=success($sucesso);
}
if(isset($_POST['confirma'])){
      $result=success($sucesso);
}
if(isset($_POST['pagar'])){
      payparking();
}
?>

<html>
<head>
<meta charset="utf-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
   <meta name="description" content="">
   <meta name="author" content="">
   <link rel="icon" href="img/img.png">

   <title>Minha Conta</title>

      <!-- Latest compiled and minified CSS -->
```

```
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">

  <!-- Optional theme -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-theme.min.css" integrity="sha384-fLW2N01lMqjakBkx3l/M9EahuwpSfeNvV63J5ezn3uZzapT0u7EYsXMjQV+0En5r" crossorigin="anonymous">

  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/1.9.1/jquery.js"></script>
  <!-- <script src="http://code.jquery.com/jquery-1.9.1.js"></script>  -->
  <script src="js/jsqrcode-combined.min.js"></script>
  <script src="js/html5-qrcode.js"></script>

</head>

<body>

<nav class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
      <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar">
          <span class="sr-only">Toggle navigation</span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
        </button>
        <a class="navbar-brand" href="index.html">Estacionamento Inteligente</a>
      </div>
      <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
          <li><a href="index.html">Home</a></li>
          <li class="active"><a href="myaccount.php">Minha Conta</a></li>
          <li><a href="logout.php">Logout</a></li>
        </ul>
      </div><!--/.nav-collapse -->
    </div>
  </nav>

  <div class="jumbotron">
  <div class="container">

<br><br>

  <div class="starter-template text-center">
   <h1 style="font-size:175%;">Faça o scan do QR Code para verificar sua reserva e dar início à sua estadia</h1><br>
  <center><div id="reader" align="center" style="width:300px; height:250px; display:none">
  </div></center>

  <form action="" method="post">
  <button id="btn-ligarCentral" type="button" class="btn-danger btn-lg" onclick="document.getElementById('reader').style.display='block';">
      <span class="glyphicon glyphicon-qrcode" aria-hidden="true"></span> QR Code
  </button> 
  <button name="confirma" type="submit" class="btn-danger btn-lg"><span class="glyphicon glyphicon-ok" aria-hidden="true"></span> Confirmar</button>
  </form>

  <br><h4 id="qrResult" name="qrResult"><h4><br>
  </div>
```

```
   <script type="text/javascript">
     $('#reader').html5_qrcode(function(data){
       // do something when code is read
         console.log(data)
                 document.getElementById("qrResult").innerHTML = data;

     $.ajax({
                 type: "POST",
                 url: 'qrcode.php',
                 data: {data : data},
                 //success: function(data)
                 //{
                     //alert("QR code cadastrado com sucesso!");
                 //}
               });
         },
         function(error){
           //show read errors
           console.log(error)
         }, function(videoError){
           //the video stream could be opened
           console.log(videoError)
         }
      );
   </script>
     </div>
<div class="container text-center">
     <!-- Example row of columns -->
     <div class="row">
       <div class="col-md-4">
         <div class="starter-template" style="padding-top: 20px">
          <div class="starter-template text-center">
             <h1 style="font-size:175%;">Consulte seu tempo de estadia e o valor total a ser pago </h1>
             <h4>Clique no botão "Saída" e em seguida no botão "Resultado"</h4><br><br>
             <form action="" method="post">
             <div class="btn-group" role="group" aria-label="...">
                <button name="saida" type="submit" class="btn-danger btn-lg"><span class="glyphicon glyphicon-th" aria-hidden="true"></span>
Calcular</button> 
                <button name="displayhour" type="button" class="btn-danger btn-lg"
onclick="document.getElementById('time').style.display='block';"><span class="glyphicon glyphicon-eye-open" aria-hidden="true"></span>
Resultado</button>
             </div>
             </form>
             <br><br>
             <div class="lead" id="time" style="display:none">
               <h3><?php echo "<br>Tempo total: $valor1[0]<br><br>Valor total: R$ $valor1[1],00"; ?></h3>
             </div><br><br>
         </div><br><br><br>
       </div><!-- /.container -->
     </div>
     <div class="col-md-4">
     </div>
     <div class="col-md-4">
       <div class="starter-template" style="padding-top: 20px">
       <div class="starter-template text-center">
       <h1 style="font-size:175%;">Para realizar o pagamento clique no botão abaixo </h1><br><br>
       <form action="" method="post">
```

```
          <div class="btn-group" role="group" aria-label="...">
          <button name="pagar" type="submit" class="btn-danger btn-lg"><span class="glyphicon glyphicon-usd" aria-hidden="true"></span> Realizar pagamento</button> 
          </div>
        </form>
        </div>
        </div>
      </div>
  </div>
 </div>

   <footer class=bs-docs-footer>
     <br><br><br><br><br><p class="lead text-center" style="font-size:100%;">Copyright Thiago Romanelli WebDesign ®<br> Since 2016</p>
   </footer>

   <!-- Bootstrap core JavaScript
   ================================================== -->
   <!-- Placed at the end of the document so the pages load faster -->
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
   <script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery.min.js"><\/script>')</script>
   </body>
</html>
```

A.8. Código para servidor MQTT

```
<?php

$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());


$client = new Mosquitto\Client('teste_mqtt');

$client->onConnect(function($code, $message) use ($client) {
   /* Subscribe to the broker's $SYS namespace, which shows debugging info */
   $client->subscribe('vaga1/presenca', 0);
   $client->subscribe('vaga2/presenca', 0);
   $client->subscribe('vaga3/presenca', 0);
});

$client->onMessage(function($message) {
            switch ($message->topic) {
                        case 'vaga1/presenca':
                                    $sql=mysql_query("UPDATE teste_vagas SET vaga1 = '$message->payload' WHERE id = 1") or die(mysql_error());
                           echo $message->topic, "\n", $message->payload, "\n\n";
                                    break;

                        case 'vaga2/presenca':
```

```
                        $sql=mysql_query("UPDATE teste_vagas SET vaga1 = '$message->payload' WHERE id = 2") or
die(mysql_error());
                    echo $message->topic, "\n", $message->payload, "\n\n";
                        break;

                case 'vaga3/presenca':
                        $sql=mysql_query("UPDATE teste_vagas SET vaga1 = '$message->payload' WHERE id = 3") or
die(mysql_error());
                    echo $message->topic, "\n", $message->payload, "\n\n";
                        break;

                default:
                        echo 'bla';
                        break;
        }
});

$client->connect('localhost', 1883);
$client->loopForever();
?>
```

A.9. Código para verificação de vagas

```
<html>
<head>
<title>Cadastrando...</title>
</head>

<body>
<?php
//include 'login1.php';
$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());
?>

<?php
session_start();
$login = $_SESSION['login'];

if(isset($_POST['data'])){

        $vaga = $_POST['data'];
        switch($vaga){
        case 1:
                $sql1 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 1") or die(mysql_error());
                $status1 = mysql_fetch_row($sql1);
                if($status1[1] == 1){
                        $_SESSION['erro1'] = "1";
                        $msg = "Vaga ja esta reservada";
                        echo '<script type="text/javascript">alert("' . $msg . '")</script>';
```

```
                }
                if($status1[1] == 0){
                        $_SESSION['erro1'] = "0";
                        $sql=mysql_query("UPDATE teste_users SET vaga='$vaga' WHERE email='$login'") or die(mysql_error());
                }
                break;

        case 2:
                $sql2 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 2") or die(mysql_error());
                $status2 = mysql_fetch_row($sql2);
                if($status2[1] == 1){
                        $_SESSION['erro2'] = "1";
                }
                if($status2[1] == 0){
                        $_SESSION['erro2'] = "0";
                        $sql=mysql_query("UPDATE teste_users SET vaga='$vaga' WHERE email='$login'") or die(mysql_error());
                }
                break;
        case 3:
                $sql3 = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = 3") or die(mysql_error());
                $status3 = mysql_fetch_row($sql3);
                if($status3[1] == 1){
                        $_SESSION['erro3'] = "1";
                }
                if($status3[1] == 0){
                        $_SESSION['erro3'] = "0";
                        $sql=mysql_query("UPDATE teste_users SET vaga='$vaga' WHERE email='$login'") or die(mysql_error());
                }
        break;
        default:
                echo "bla";
        }
}
?>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
   <script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery.min.js"><\/script>')</script>
</body>
</html>
```

A.10. Código para verificação de QR Code

```
<html>
<head>
<title>Cadastrando...</title>
<script type="text/javascript">
function login_failed(){
        alert("Vaga reservada não condiz com a vaga ocupada, ou não há a presença de veículo no local");
        //setTimeout("window.location='index.html'",2000);
}
</script>
</head>
```

```
<body>
<?php
$host = "localhost";
$user = "root";
$pass = "123456";
$banco = "usuarios";
$conexao = mysql_connect($host, $user, $pass) or die(mysql_error());
mysql_select_db($banco) or die(mysql_error());
?>

<?php

session_start();
$login = $_SESSION["login"];

if(isset($_POST['data']))
{
$qrcode = $_POST['data'];
$sql=mysql_query("INSERT INTO teste_qr(qrcode) VALUES('$qrcode')") or die(mysql_error());
}
$sql = mysql_query("SELECT id,vaga1 FROM teste_vagas WHERE id = '$qrcode'") or die(mysql_error());
$vaga = mysql_fetch_row($sql);
$sql1=mysql_query("SELECT email,senha,id,vaga,entrada FROM teste_users WHERE email='$login'") or die (mysql_error());
$reserva = mysql_fetch_row($sql1);

        if($qrcode == $reserva[3] and $vaga[1] == 1){
                $nowtime = time();
    $sql=mysql_query("UPDATE teste_users SET entrada='$nowtime' WHERE email='$login'") or die(mysql_error());
    $sucesso=1;
        }
        else {
                $falha = "Vaga reserva nao condiz com a vaga estacionada";
                $sucesso=0;
                $_SESSION['falha'] = $falha;
        }
        $_SESSION['sucesso']=$sucesso;
?>
</body>
</html>
```