UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
COINT - TECNOLOGIA EM SISTEMAS PARA INTERNET
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

MARCUS VENICIUS PINA CAVALCANTI

SISTEMA PARA GERENCIAMENTO DE ACESSO DE AUTOMÓVEIS POR RECONHECIMENTO DE IMAGENS

PROJETO DE TRABALHO DE CONCLUSÃO DE CURSO

GUARAPUAVA
2019

MARCUS VENICIUS PINA CAVALCANTI

SISTEMA PARA GERENCIAMENTO DE ACESSO DE AUTOMÓVEIS POR RECONHECIMENTO DE IMAGENS

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de Tecnólogo.

Orientador: Prof. Dr. Roni Fabio Banaszewski
Universidade Federal do Paraná - Campus Guarapuava

Coorientador: Prof. Me. Paulo André Filipak
Universidade Federal do Paraná - Campus Guarapuava

GUARAPUAVA
2019

Dedico a todos os envolvidos, professores, alunos, dentre muitos outros da qual fizeram um esforço para este trabalho seja executado.

É dito que um programa de computador aprende a partir de uma experiência E com respeito a uma classe de tarefas T e medida de desempenho P, se seu desempenho nas tarefas em T, segundo a medida P, melhora com a experiência E. (Mitchell, 1997).

RESUMO

Cavalcanti, Marcus. SISTEMA PARA GERENCIAMENTO DE ACESSO DE AUTOMÓVEIS POR RECONHECIMENTO DE IMAGENS. 2019. 29 f. Projeto de Trabalho de Conclusão de Curso – Curso de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná. Guarapuava, 2019.

O surgimento de algoritmos de aprendizagem de máquina proporcionou novos campos de estudos na computação e diversas aplicabilidades no mundo real. Muitas destas aplicabilidades estão relacionadas à visão computacional, que estuda o processamento de imagens para interpretação e avaliação das informações contidas nela. Como exemplo de tais aplicabilidades estão Sistemas de Reconhecimento Automatizado de Placas (ALPR), os quais tem a finalidade de reconhecer os caracteres que integram uma placa veicular. A tecnologia ALPR é largamente utilizada em sistema de tráfego e monitoramento de estacionamento. Neste sentido, a corrente proposta consiste na implementação de uma sistema de controle de acesso ao estacionamento por meio de ALPR, e possíveis contribuições.

Palavras-chave: ALPR. OCR. ALPR em Raspberry.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

DERAC-GP	Departamento de Registros Acadêmicos
DIRGEP	Diretoria de Gestão de Pessoas
ALPR	Automatically Recognize License Plates
OCR	Optical Caracters Recognition
IoT	Internet of Things
WoT	Web of Things
REST	Representational State Transferer
IA	Inteligência Artificial
XML	Extensible Markup Language
URI	Uniform Resource Identifiers
GPIO	General-purpose input/output

SUMÁRIO

1 INTRODUÇÃO

No campus Guarapuava da Universidade Tecnológica Federal do Paraná (UTFPR) há um fluxo diário elevado de pessoas e veículos, principalmente em horários que antecedem os períodos das aulas. O acesso de pessoas e veículos nas dependências do campus é controlado por funcionários de uma empresa de segurança terceirizada. Estes funcionários são responsáveis por permitir ou não o acesso de veículos ao estacionamento, além de recepcionar os visitantes.

A tomada de decisão para a autorização de acesso ao campus se dá através da verificação da existência de um adesivo com características particulares colado nos para-brisas dos veículos. Este adesivo contém o logo da UTFPR e é disponibilizado a alunos, técnico administrativos e professores com o propósito de facilitar a identificação. Para cada uma das três categorias de motoristas é definida uma cor para os adesivos: o azul é usado para identificar terceirizados, o verde para alunos e o amarelo para professores e técnicos administrativos. Portanto, por meio das cores, o funcionário consegue extrair uma informação a mais para tomada de decisão. Porém, ele não tem dados suficientes para identificar o motorista.

Além disso, este processo de autorização de acesso traz consigo alguns problemas, tais como: o desgaste prematuro do adesivo se colado em local externo, problema muito comum em motocicletas; posicionamento do adesivo em um local de difícil visibilidade ao funcionário de segurança, que ocorre quando o adesivo é apenas deixado no painel do carro e mesmo o não uso do adesivo, quando o funcionário já conhece o veículo do servidor. Ademais, um outro problema ocorre quando o motorista vende o veículo e esquece de remover o adesivo. Com isso, o novo dono do carro poderia ter livre acesso ao campus sem necessariamente fazer parte da comunidade universitária. Neste sentido, este controle baseado em adesivos também se apresenta falho por causa da facilidade de falsificação de tais adesivos, levando ao acesso ao estacionamento de qualquer indivíduo que pactue com tal fraude.

Uma solução seria criar uma base de dados dos motoristas com permissão de acesso ao campus e também de seus veículos identificados por suas placas. A equipe de segurança tendo acesso a esta base de dados de uma forma simples e rápida, poderia ter maior poder para a tomada de decisão. Esta rapidez demanda a implementação de um sistema constituído por uma câmera de imagem e da tecnologia de Reconhecimento Automatizado de Placas (ALPR) de veículos. Com isso, as informações dos motoristas poderiam ser pesquisadas na base de dados e apresentadas automaticamente ao funcionário imediatamente a chegada de um veículo na portaria do campus. A identificação da placa por uma solução inteligente de tratamento de imagens evita esforços do funcionário para ler a placa do veículo e digitar no sistema, atividade que pode prejudicar o bom fluxo de veículos por demandar tempo e ser tendente a erro de digitação.

Com esta solução, não haveria mais a necessidade de depender de adesivos, uma vez que a placa serviria como credencial de acesso. Também, caso o motorista venda o veículo

ou termine o vínculo com a UTFPR por algum motivo, bastaria atualizar esta informação na base de dados. Neste sentido, quando um palestrante externo for convidado ao campus, bastaria cadastrá-lo nesta base de dados juntamente com os dados de seu veículo para este ter livre acesso ao estacionamento do campus. Também, o funcionário de segurança poderia gerenciar o número de vagas do estacionamento, uma vez que daria para controlar quantos veículos entraram e saíram no dia. Ainda, havendo mais informações sobre o motorista, tal como email ou mesmo telefone, o funcionário pode entrar em contato com o motorista sobre alguma infração cometida(ex, som alto, velocidade alta) ou mesmo para aviso (ex. vidro aberto, farol ligado).

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo deste trabalho é desenvolver um sistema com reconhecimento de placas de automóveis para controlar o acesso ao campus da UTFPR de Guarapuava.

1.1.2 Objetivos Específicos

- Implantação de um ALPR em uma plataforma *Raspberry*.;
- Construir uma aplicação web para receber os dados do ALPR;
- Construir uma aplicação para integração com OpenALPR;
- Treinamento algorítimo ALPR para as placas de carro do Brasil;

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado da seguinte maneira: O capítulo 2 apresenta o embasamento teórico, onde serão descritos os sistemas correlatos e a base teórica para composição do projeto. Em consonância, o capítulo 3 apresenta a metodologia aplicada no projeto, contendo conceitos teóricos sobre metodologias ágeis.

A seguir, o capítulo 4 apresenta os artefatos de projeto do sistema a ser desenvolvido. Por fim, o capítulo 5 apresenta a conclusão sobre o trabalho, dando uma perspectiva sobre trabalhos futuros.

2 REVISÃO DE LITERATURA

Neste capítulo é apresentada a base teórica que orienta o projeto. Basicamente, é apresentado um estudo sobre aplicações de mercado que diretamente ou indiretamente servem ao mesmo propósito e as tecnologias a serem utilizadas. Mais precisamente, na seção 2.1 são apresentadas as aplicações similares à proposta, na seção 2.2 será apresentado uma explanação dos conceitos teóricos mais gerais e por fim, na 2.3 serão apresentadas as tecnologias a serem utilizadas para o desenvolvimento do sistema.

2.1 SISTEMAS CORRELATOS

2.1.1 Sistema de Identificação da empresa de tecnologia DBA

O sistema trabalha com *Optical Character Recognition* (OCR) em conjunto com ALPR para fazer o controle de acesso e rastreabilidade dos veículos nas dependências do estacionamento (DBA, 2019). Este sistema tem integração com alguns outros softwares de gerenciamento de estacionamento desprovidos das tecnologias de OCR ou ALPR. Entretanto, o ALPR da DBA é um software proprietário não dando a possibilidade de um cadastro aberto, deixando apenas a equipe de segurança adicionar os dados dos usuário e deixando inviável a prática de dar a responsabilidade para o usuário. Outrossim, por ser um software proprietário, sua customização torna-se inviável para as necessidade do campus.

2.1.2 Detecção, Reconhecimento de Placas da empresa Meerkat

A Meerkat é uma empresa que trabalha com tecnologias da Visão Computacional. Mais precisamente, a empresa trabalha com as seguintes tecnologias: reconhecimento facial, reconhecimento de logomarcas e reconhecimento de placas (MEERKAT, 2019). O ALPR da empresa funciona com respostas em formato JSON advindas do reconhecimentos OCR. Contudo, o sistema é proprietário. A aquisição é através de licença de uso com tempo determinado, tornado impraticável sua aquisição.

2.1.3 AutoVu ALPR

O AutoVu ALPR foi desenvolvido pela empresa Genetec. Ele consiste em um módulo que compõe o sistema Security Center da empresa, uma solução completa com videomonitoramento, rastreamento em tempo real, com aplicabilidade em diversos situações como controle de trafego de cidades, controle de proteção a furtos de veículos (GENETEC, 2019). Por fim, sendo o AutoVu parte de um sistema robusto e complexo, não sendo viável para o problema do acesso ao estacionamento do campus.

2.1.4 Estudo Comparativo

Os sistemas apresentados são todos proprietários. O sistema da Genect é um sistema completo, sendo que o ALPR apenas compõe um módulo, ao passo que os sistemas da DBA e Meerkat são soluções da qual necessita de uma estrutura pronta, tal como câmera e servidor para instalar o sistema. Por fim, todos não são customizáveis e relativamente caros, inviabilizando a aquisição pelo campus.

2.2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção é apresentada uma breve explicação que norteia alguns conceitos teóricos para desenvolver o projeto do sistema. Mais precisamente, na subseção 2.2.1 são apresentados alguns conceitos gerais sobre Inteligência Artificial, agentes e aprendizagem de máquina, na subseção 2.2.2 é explanado sobre visão computacional. Por fim, na subseção 2.2.3 é dada uma visão geral sobre algoritmos ALPR e na subseção 2.2.4 são introduzidos alguns conceitos sobre Internet das Coisas(IoT) e Web das Coisas(WoT).

2.2.1 Aprendizagem de Máquina

A Inteligência Artificial(IA) tem como propósito o aprimoramento cognitivo dos agentes, além de processamento de linguagem natural, representação do conhecimento, raciocínio automatizado, visão computacional e robótica. Como explica RUSSEL Stuart; NORVIG (2013):

> Processamento de linguagem natural, para permitir que eles se comuniquem com sucesso em um idioma natural, representação de conhecimento para armazenar o que sabe ou ouve; raciocínio automatizado para usar as informações armazenadas com finalidade de responder a perguntas e tirar novas conclusões; aprendizado de máquina para se adaptar a novas circunstâncias e para detectar e extrapolar padrões; [...] visão computacional para receber objetos e robótica para manipular objetos e movimentar-se.

Segundo RUSSEL Stuart; NORVIG (2013) um agente pode através da aprendizagem melhor seu comportamento baseando em alguns fatores: o conhecimento prévio do agente, feedback de aprendizagem. Algumas abordagens de aprendizagem são: aprendizagem indutiva, analítica e dedutiva. Agentes é o termo dado para um componente computacional que percebe seu ambiente através de sensores, e interagir com este mesmo ambiente com atuadores. (RUSSEL STUART; NORVIG, 2013). Como base nisto pode-se constar que um feedback na aprendizagem é dedutivo, haja vista, que o agente pode coletar informações, ao passo que explora seu ambiente. Diante disto temos os conceitos de feedback de aprendizagem, temos três tipos destes, aprendizagem não supervisionada, por reforço e por fim aprendizagem supervisionada.

Aprendizagem não supervisionada define que o agente aprende através da detecção de grupos de exemplos, de entradas de dados úteis. Logo temos, que o agente analisa um

conjunto de dados e realiza através de probabilística[1] afim de agrupar dados similares. O agente aprender baseando no agrupamento resultados positivos e negativos sem nunca ter rotulado exemplos dos mesmo (RUSSEL STUART; NORVIG, 2013).

Aprendizagem supervisionada o agente recebe um conjunto de dados conhecido, que observa os exemplos dos pares de dados, conjunto de entrada e saída, entradas são as percepções do agente, a saída por sua vez, e representado por um valor explícito pelo instrutor do agente. Takatuzi (2017) explica que neste tipo de aprendizado "é fornecido ao algoritmo de aprendizagem um conjunto de dados de treinamento nos quais a classe associada a cada dado é conhecida por um supervisor externo."

Na técnica por reforço, a abordagem de aprendizagem o agente aprende a medida que recebe recompensa ou punições, esta tipo de aprendizagem o agente decide quais das ações anteriores baseando-se no reforço que recebeu (RUSSEL STUART; NORVIG, 2013).

2.2.2 Visão computacional

Aprendizagem de máquina proporcionou novos campos de estudo como a Visão computacional, da qual estuda o processamento de imagens na busca por interpretar e avaliar as informações contidas nela. Utilizando-se de técnicas de aprendizagem de máquina supervisionada, proporcionou a descoberta de tecnologias como Reconhecimento ótico de caracteres (OCR da sigla em inglês *Optical Character Recognition*), que por sua vezes deu espaço para mais descobertas, tais como, Sistema de Reconhecimento Automatizado de Placas (ALPR), a qual tem a finalidade de reconhecer os caracteres que integram uma placa veicular. Segundo ALVARENGA (2017) os autores explicam que em um sistema ALPR, a etapa de reconhecimento de caracteres óticos (OCR) consiste em analisar um fragmento da imagem contendo a representação de um único caractere contida na imagem.

O processamento de imagens muito se assemelha a visão computacional, um paralelo entre esses dois campos é mais que natural, contudo, diferenças sutis podem ser aferir estudando os conceitos estabelecidos na literatura como explica Marengoni e Stringhini (2009) "processamento de imagens é um processo onde a entrada do sistema é uma imagem e a saída é um conjunto de valores numéricos, que podem ou não compor uma outra imagem", ao passo que visão computacional utiliza-se de conceitos de aprendizagem de máquina para processar uma imagem afim de emular a visão humana Marengoni e Stringhini (2009) explica que a visão computacional emprega um processo similar, possui como entrada uma imagem, com uma diferença que a saída é a interpretação da imagem. Este paralelo, entre visão computacional, pode compreendido pelo fato de o processo de visão computacional em muitos casos, inicia-se através do processamento da imagem, para facilitar o agente identificar, por exemplo, o processamento de imagem, redução de ruído, aumentando ou diminuindo o contraste da imagem,

[1] Técnicas de amostragem em que a seleção é aleatória de tal forma que cada elemento tem igual probabilidade de ser sorteado para a amostra

para identificar um objeto especifico na imagem.

Uma imagem após adquirida digitalmente, ela pode ser vista como uma matriz de números inteiros, logo pode atribuída operações lógicas e aritméticas Filho e Neto (1999), denominamos o processo de digitalização de imagem em aquisição de imagem o processo de converte-la de uma cena real, imagem tridimensional, para uma imagem analógica, cena bidimensional, assumindo que uma imagem pode ser adquirida, sem se preocupar com o processo empregado neste processo. Um processamento de imagem pode ser classificada como Scuri (1999) em duas classes, por escopo ou por resultado, cada um com técnicas e processos distintos das quais, norteiam um fator em comum a qualidade, Scuri (1999) explica que existem duas divisões no contexto de qualidade em processamento de imagem são eles: fidelidade e inteligibilidade, na fidelidade o processo e voltado para aproximar a imagem processada da imagem original ou de um padrão estipulado que a melhor represente. No segundo caso, nos preocupamos com a informação que conseguimos extrair da imagem, seja pelo olho humano, seja por algum processamento Scuri (1999). Podemos perceber que o processamento de imagem, e voltado para alcançar uma das duas subdivisões de qualidade a qual Scuri (1999) explica.

No paragrafo anterior o uso do processamento de imagem tem como finalidade a qualidade da imagem, a visão computacional utiliza-se inteligibilidade afim de diminuir o ruim da imagem, para então processar a imagem para reconhecimento de objetos ou rastreamento, ambos os processos possuem uma interação intima na área de inteligência artificial. Processo de reconhecer um objeto é uma das principais área de visão computacional, da qual esta correlata ao estudo de padrões. Como explica. Marengoni e Stringhini (2009)

> As técnicas de reconhecimento de padrões podem ser divididas em dois grandes grupos: estruturais, onde os padrões são descritos de forma simbólica e a estrutura é a forma como estes padrões se relacionam; o outro grupo é baseado em técnicas que utilizam teoria de decisão, neste grupo os padrões são descritos por propriedades quantitativas e deve-se decidir se o objeto possui ou não estas propriedades.

Ao passo que o processo de rastreamento é o processo de reconhecer um padrão em uma finita sequência de imagens, Marengoni e Stringhini (2009) diz que o processo de rastreamento está atrelado ao conhecimento sobre o movimento do objeto que está sendo rastreado no intuito de minimizar a busca dentre as imagens em uma sequencia. Já o processo de segmentação é o processo de dividir a imagem por região ou objetos distintos, o algoritmo busca características distintas do objeto, cor, tonalidade, pixels das quais tem similaridades e podem ser conectados algumas técnicas de segmentação são:

1. Segmentação por borda;
2. Segmentação por corte; e
3. Segmentação por crescimento de região.

Com base nos parágrafos anteriores podemos perceber similaridades entre processamento de imagem e visão computacional, os processos de reconhecimento de objetos e rastreamentos de objetos em sequencia de imagens, são processamentos de imagens onde um

agente utiliza-se das técnicas de aprendizagem de máquina supervisionada para reconhecer padrões nas imagens, assim como o rastreamento de imagem. Em detrimento do processamento de imagem, tem como finalidade a qualidade holística da imagem. Ao passo que a visão computacional a prega o processamento de imagem como um ante processo para sua finalidade.

2.2.3 Algoritmos ALPR

O principal objetivo de um sistema ALPR (*automatic licence plate recognition*) consistem em um processo em 3 etapas com uma entrada complexa, imagem ou stream de vídeo, e ter como saída simples, os caracteres da placa, texto.

São necessárias ao menos 3 etapas para um reconhecimento de placas de veículos, localização, segmentação dos caracteres e reconhecimentos dos caracteres. A etapa de localização consistem em fazer um rastreamento na imagem afim de detectar uma região, onde a placa será encontrada, o agente conhece previamente o objeto buscado, Anagnostopoulos et al. (2008) os autores demonstram uma técnica de analise estática para identificar irregularidades locais na imagem, denominada de *Sliding Concentric Windows*, uma vez identificada a localização da imagem o processo agora é segmentar cada caractere da placa em objetos distinto, para a etapa de reconhecimento e identificação dos objetos segmentados e associa-los ao um caractere.

2.2.4 Internet das Coisas e Web das Coisas

Inúmeras possibilidades podem surgir quando dispositivos são conectados a internet, como mencionado no parágrafo anterior. Estes são conceitualmente denominados objetos inteligentes ou *Smart Objects* em inglês, em suma um objeto inteligente deve necessariamente conter as seguintes características:

1. Unidade de processamento;
2. Unidade de memória;
3. Capacidade para comunicação entre outros dispositivos;
4. Sensores ou atuadores;

Estes dispositivos podem ter os mais variados e diferentes propósitos, mas, como explica França et al. (2011) "a maioria dos objetos são atualmente conectados a internet (e algumas vezes a Web) utilizando softwares e interfaces proprietárias, o que torna onerosa a criação de aplicações", a complexidade pode aumentar com a utilização de protocolos e interfaces especificas como continua o autor França et al. (2011) "pois é necessário que o desenvolvedor possua conhecimento especializado para cada dispositivo utilizado no projeto" o autor conclui citando Guinard e Trifa, "é necessário utilizar uma linguagem comum a diferentes dispositivos".

Na literatura diversos trabalhos tratam de temas relacionados a IoT, este paradigma está impactando no social e no dia a dia, Entretanto um desafio que deve ser alcançado um deles e a padronização da comunicação entre esses dispositivos, que em muitos casos utilizam

interfaces ou protocolos proprietários, dificultando o desenvolvimento. WoT propõe utilizar-se de protocolos Web entre o meio físico e o digital, na tentativa de criar uma linguagem comum para comunicação destes dispositivos, dando a possibilidade desses, dispositivos, gerir um serviço que será utilizado em diversas aplicações diferentes, em outras palavras reutilizável, além de proporcionar um serviço Web que será dentre outros casos, comunica-se com protocolo HTTP trafegando JSON entre as partes, seguindo a arquitetura REST *Representation State Transfer* proposta por Roy Fielding.

Na literatura trabalhos e pesquisas referente a Internet das Coisas tem diferentes definições, como aponta Atzori, Iera e Morabito (2010), os autores faz um paralelo entre as diferentes definições e apontam que as pesquisas encontradas focam nos termos: internet, coisas e visão orientada a internet. Cada termo destina-se a um aspecto da IoT, apesar das pesquisas envolvidas alavancarem o paradigma da IoT muitos desafios, ainda devem ser alcançados, um deles é a padronização da comunicação, embora proposta de 6LowPAN, um protocolo IP que tem como proposito para trabalhar com dispositivos, da qual tem pouca disponibilidade de recursos computacionais, aquele problema a Web das Coisas propõe solucionar.

Utilizar os protocolos Web como meio de comunicação, como explica França et al. (2011) "A WoT propõe que os protocolos Web sejam utilizados como linguagem comum para integração de dispositivos físicos no meio digital", esta comunicação ocorre sobre a camada de aplicação do modelo de camadas TCP/IP França et al. (2011). Isto possibilita o uso de recursos web sejam disponibilizados utilizando-se de estilo arquiteturas REST, possibilitando duas abordagens, como explica:

> A Web das Coisas emprega os princípios REST para disponibilizar as funcionalidades dos dispositivos inteligentes na Web utilizando duas abordagens. Na primeira abordagem, são implantados servidores Web embarcados em dispositivos inteligentes e as funcionalidades desses dispositivos são disponibilizadas na forma de recursos RESTful. Na segunda abordagem, quando um objeto inteligente não possui recursos de hardware suficientes para executar um servidor embarcado, é possível utilizar outro dispositivo como ponte para disponibilizar as funcionalidades do dispositivo inteligente na Web através de uma interface RESTful.

(FRANÇA et al., 2011) Estas abordagens permitem que os dispositivos exponham através dos recursos Web interfaces reutilizáveis e bem definidas o serviço pode ser consumido por diferentes aplicações.

2.2.5 RESTFul

Fielding e Taylor (2000) em sua dissertação de doutorado propôs uma estilo de arquitetura, denominada REST, este estilo propõe que Cliente e Servidor fiquem separados e independes, alcançando a separação de responsabilidade, o estilo foi baseado e um estilo arquitetura denominado Cache-Servidor sem Estado, da qual um serviço não dependa de estados de sessão para controlar a resposta da requisição, assim uma requisição necessariamente deve

ter as propriedades necessárias para ser completada o seu processo, assim o servidor não se responsabiliza pelas informações necessária deixando a cargo do cliente conhecer todas as informações necessárias. Outra característica evidente ao estilo REST é a interface uniforme, padronizando as chamadas através do uso do protocolo HTTP e seus métodos, e códigos de resposta.

O padrão REST determina como deve ser realizada a transferência de dados, que representa o corresponde ao conjunto de valores que representa uma determinada entidade em um dado momento. Essa transmissão de estados se dá a partir da especificação de parâmetros, em alguns casos chamados de restrições. Quando isso ocorre, o que se obtém são serviços escaláveis, de fácil modificação e manutenção, e que apresentam boa performance, tornando esses serviços adequados a serem utilizados através da internet.

Nos serviços RESTful, tanto os dados quanto as funcionalidades são considerados recursos e ficam acessíveis aos clientes através da utilização de URIs (Uniform Resource Identifiers), que normalmente são endereços na web que identificam tanto o servidor no qual a aplicação está hospedada quanto a própria aplicação e qual dos recursos oferecidos pela mesma está sendo solicitado.

Os recursos disponíveis em um aplicação RESTful podem ser acessados ou manipulados a partir de um conjunto de operações predefinido pelo padrão. como explica Vernon (2016) as operações possibilitam alterar (PUT), ler (GET), criar (POST) e apagar (DELETE) recursos, e estão disponíveis a partir de mensagens utilizando o protocolo HTTP. Já para os usuários que utilizam sistemas baseados em RESTful, são serviços que além de as solicitações aos serviços apresentarem boa responsividade, de acordo com a infraestrutura disponível, os usuários não fazem muita distinção entre um sistema executando localmente ou em um servidor de aplicação. Além dessas qualidades.

2.3 TECNOLOGIAS

Nesta seção será demonstrando as tecnologias utilizadas no projeto, desde os equipamentos eletrônicos, como linguagem de programação, módulos e frameworks para construção da aplicação Web.

2.3.1 Raspberry

Fazendo-se necessário a integração dos Objetos Inteligentes a utilização de protocolos Web, isto possibilita que muitos outros dispositivos se conectem através de uma linguagem comum, possibilitando uso de arquiteturas REST do inglês *Representational State Transferer* – Transferencia de Estado Representacional. A utilização do Raspberry PI encaixa-se no conceito de objeto Inteligente, pois detém as características de um objeto inteligente, demonstra nos paragrafo anteriores, desenvolvido pela fundação Inglesa de mesmo nome. No campo da computação a plataformas deu novos caminhos para revolução digital, ad hoc. Afim de promover

soluções de WoT, pois a plataforma proporciona um protótipo de baixo para novas soluções.

Um objeto inteligente ou *Smart Object* é composta por quatro unidades: processamento ou memoria, sensores ou atuadores, comunicação energia Santos et al. (2016). Nesta definição o Raspberry Pi está caracterizado pois, em uma descrição simples é um computador de pequeno porte que utiliza o processador multimídia Broadcom BCM2835, do tipo SoC (*system-on-chip* – em português, sistema em um chip). Isso significa que a grande maioria dos componentes do sistema, incluindo sua unidade central e as de processamento gráfico, assim como o hardware de áudio e de comunicações, está montada em um único componente. O Raspberry Pi não tem nenhum disco rígido na sua composição de hardware, em vez disso, utiliza um cartão de memória. Desenvolvido no Reino Unido pela Fundação Raspberry Pi com principal objetivo de promover o ensino em Ciência da Computação básica em escolas. A plataforma Raspberry ainda provem de porta GPIO (*general-purpose input/output*), entra de comunicação genérica, digital da qual pode ser usada para comunicação entre outros dispositivos, esta porta tem dois estados, 0 e 1, onde um e para ligada e 0 desligada, quando ligada a tensão no pino da GPIO é de 3.3V, ao passo que quando desligada e 0v.(FOUNDATION,)

2.3.2 OpenALPR

OpenALPR é uma biblioteca de reconhecimento de placa de licença automática de código aberto escrita em C ++ esta analisa imagens e fluxos de vídeo para identificar placas de carros. A saída é a representação de texto de quaisquer caracteres da placa de licença. Utiliza-se das técnicas de visão computacional para o processamento de imagem e aprendizagem de máquina para o trabalho. A biblioteca tem integração com diversas linguagens. seu funcionamento consiste da seguinte forma, como consta na sua documentação:

1. The image stream will constantly be pulled from the IP camera via MJPEG over HTTP.
2. Alprd will process the stream as quickly as possible while looking for plate images. The daemon will automatically skip frames to stay in sync with clock time.
3. When one or more plates are detected, the information will be written to a local beanstalkd queue (tube name “alprd”) as JSON data.
4. Optionally, alprd will also save the image to a configurable location as a jpeg.
5. Optionally, alprd will also run a separate process that will drain the beanstalkd queue and upload data to a remote HTTP server via POST.

(TECHNOLOGY, 2017)

Desta maneira a integração com um servidor Web com integração da biblioteca de código fonte aberto como o Wiki:

> *OpenALPR is an open source Automatic License Plate Recognition library written in C++ with bindings in C#, Java, Node.js, and Python. The library analyzes images and video streams to identify license plates. The output is the text representation of any license plate characters.*
>
> (TECHNOLOGY, 2017)

Aquela prove também o uso de utilização de *Stream* de vídeo, sendo seu uso mais efetivo. Esta biblioteca facilitará o desenvolvimento do sistema, embora toda explicação teórica por trás do algoritmo de ALPR referenciado 2.3.2, além de informações pertinentes ao tema.

2.3.3 Java

Foi desenvolvida no inicio da década de 90 nos laboratórios da Sun Microsystems com o objetivo ser executa em eletrônicos de consumo, inicialmente o projeto era chamado de Green Project (LUCKOW; MELO, 2010). Um dos requisitos para esse tipo de software é ter código compacto e de arquitetura neutra Java é uma linguagem simples, de fácil migração, possui uma plataforma denomina JVM (Sigla para expressão em inglês - *Java Virtual Machine* a qual o código fonte e interpretado, sendo assim portátil, além de, fácil aprendizado, pois não a necessidade de, fazer tratamentos de ponteiros e gerenciamento de memória via código de programação também faz com que a programação em Java seja mais eficiente. Contém um conjunto de bibliotecas que fornecem grande parte da funcionalidade básica da linguagem, incluindo rotinas de acesso à rede e criação de interface gráfica. Baseada no paradigma da Orientação a Objetos - encapsulamento em um bloco de software dos dados e métodos de manipulação desses dados - a linguagem permite a modularização das aplicações, reuso e manutenção simples do código já implementado.

2.3.4 Spring Boot

O Spring é um projeto de código fonte aberto para plataforma java, criado por Rod Johson, hoje e mantido pela Pivotal (JOHNSON et al., 2004), O projeto Spring engloba um grande ecossistema, com diversos outros frameworks ou interfaces de configurações iniciais, utiliza-se como base para essas interfaces a inversão de dependência, responsábiliza o container do projeto instanciar e destruir os objetos da aplicação, assim como a injeção de dependência, o Spring gerencia o contêiner da aplicação entregando os objetos necessários para lógica do negocio com o mínimo de acoplamento possível, dando ao cargo da infraestrutura do Projeto Spring este controle. O Spring Boot é um projeto da Spring, patrocinado e mantido pela pela mesma empresa, Pivotal que veio para facilitar o processo de configuração e publicação de nossas aplicações. A intenção é ter o seu projeto rodando o mais rápido possível e sem complicação.

Com convenção sobre a configuração. Basta declarar quais módulos deseja utilizar (Web, Template, Persistência, Segurança, são exemplos.) e a infraestrutura do Spring Boot se encarrega de configurar através dos starters que inclui no pom.xml, arquivo XML(*Extensible Markup Language*) utilizado pelo Apache Maven, do seu projeto. Eles, basicamente, são dependências que agrupam outras dependências. Inclusive, como temos esse grupo de dependências representadas pelo starter, o pom.xml, do projeto, acaba por ficar mais organizado. Apesar do Spring Boot, através da convenção, já deixar tudo configurado, é possível crie as configurações

customizações caso sejam necessárias. O maior benefício do Spring Boot é que ele deixa os desenvolvedores mais livres para focar nas regras de negócio da aplicação.

2.3.5 Thymeleaf

Thymeleaf e um mecanismo de modelo server-side para ambientes Web (TEAM, 2013), como JSP(Singla para expressão em ingles *Java Server Page*), JSTL (Singla para o termo em ingles - *Java Server Pages Standard Template Library*), de código fonte aberto desenvolvido pelo espanhol Daniel Fernandez, tem boa integração com o projeto Spring. Este processa um modelos de arquivo html5, a qual detém um dialeto próprio para processar a lógica de negocia e envie para o cliente o arquivo html pronto para ser renderizado pelo navegador. Thymeleaf é ideal para o desenvolvimento web de JVM de HTML5 moderno, logo será um componente importante para estrutura da aplicação Web do projeto, da qual será utilizado para compor a interface do usuário.

2.3.6 Metodologia Scrum

A metodologia Scrum será utilizada para orientar o desenvolvimento, metodologias ágeis como Scrum, uma ferramenta conceitual de fácil implementação tem como objetivo melhor o processo de desenvolvimento como explica

> O Manifesto Ágil reconhece que a utilização de processos, ferramentas, documentação, contratos e planos pode ser importante para o sucesso do projeto, mas são ainda mais importantes os chamados valores Ágeis: os indivíduos e interações entre eles, software (ou produto) em funcionamento, colaboração com o cliente e responder a mudanças" (SABBAGH, 2013)

O Scrum segue as diretrizes do Manifesto Ágil cunhado em 2001 na reunião de fevereiro no estado de Utah nos Estados Unidos da América. Dentro do desenvolvimento o Scrum define alguns papeis, Product Owner, Scrum Master e o Time de desenvolvimento, cada um com responsabilidades distintas.

O Product Owner tem como responsabilidade descrever o produto a ser construído, ele também tem o encargo de aumentar o valor do produto, pois ele deve traduzir o problema do cliente e suas reais necessidade em funcionalidades. Por sua vez o Scrum Master tem como responsabilidade em auxiliar o time de desenvolvimento, liderando, treinando para manter as metas alcançáveis, por fim temos o time de desenvolvimento, responsável por criar as funcionalidades do sistema ou produto, em tempo hábil. SUTHERLAND (2013) Explica que o intervalo de tempo denominada Sprint tem como objetivo criar um espaço de tempo chamado de time-boxed que em geral ocorrem com duração de 7 dias ou até 30, marca a adição de nova funcionalidade ao sistema, até sua conclusão.

Para compor o conjunto de funcionalidades o *Product Owner* tem a sua mão a ferramenta *Product Backlog*, das quais descreve as funcionalidades que devem ser implementadas no sistema, que por sua vez e divididas em tarefas a serem executadas pelo time de

desenvolvimento. O *Product Backlog* é uma lista inicial de funcionalidades desejadas pelo cliente. O backlog não necessariamente consiste em todas as funcionalidades do projeto. Em síntese, são as funcionalidades desejadas do produto. Geralmente o *Product Backlog* cresce à medida que se aprende mais sobre o produto. Este será a peça que circular por todo as fases do projeto, adicionado novas itens, alterando outros, ou até mesmo removendo alguns, durante o *Spring Planning Meeting*, reunião onde os o *Product Owner*, *Srcum Master* vai descrever as prioridades para o desenvolvimento, quais Product Backlog devem ser iniciados, para definerem as Sprints.

3 METODOLOGIA

Neste capítulo será apresentado a metodologia que norteará o desenvolvimento do projeto. Também será apresentada uma descritiva das etapas que serão vencidas até o final do projeto. A metodologia ágil Scrum será utilizada para orientar o desenvolvimento. Scrum é uma ferramenta conceitual de fácil implementação e tem como objetivo melhorar o processo de desenvolvimento. Dentro do desenvolvimento, o Scrum define alguns papéis: Product Owner, Scrum Master e o Time de desenvolvimento ou *Scrum Team*.

O Scrum sugere que o desenvolvimento de um projeto de software ocorra de maneira incremental e interativa. Neste projeto, a cada nova interação um conjunto de funcionalidades será apresentado para o orientador, sendo que este fará o papel de Product Owner. Neste sentido, o coorientador fará o papel de Scrum Master. Ademais, será necessário uma interação com os membros da comissão do estacionamento do campus, usuários do sistema, para entrega de um artefato funcional. Após esta etapa, dependendo do feedback do usuário poderá ser adicionado ou removido funcionalidades. Este processo se dará até o último ciclo do projeto, Sprint.

Cada Sprint ocorrerá em um intervalo de 15 dias corridos, sendo que a cada final de Sprint o orientador e coorientador poderão dar sugestões, tecer observações e solicitar alterações. Caso haja tais reivindicações, estas serão adicionadas à próxima Sprint.

De início, foi coletado os critérios com a comissão do estacionamento do campus, o que poderia melhorar no processo de autorização do acesso ao estacionamento. Ao identificar as necessidades, foi construído o Backlog do produto que contém as principais histórias do usuário, estas levantadas inicialmente em reuniões com o papel de *Product Owner*, as quais estão descritas na seção 4.1. Certamente, à medida que as sprints do projeto avançam, novas histórias podem ser escritas e adicionadas ao backlog.

O desenvolvimento das histórias pode ser organizado em três grandes etapas: a primeira se refere ao desenvolvimento dos serviços do objeto inteligente com a utilização da biblioteca OpenALPR. A segunda consiste no treinamento do algoritmo OCR do OpenALPR e por fim, a terceira consiste na aplicação web para consumir os dados do objeto inteligente. Com a definição deste escopo e entendimento das principais histórias que compõem o backlog, é possível planejar o tempo da execução das atividades por meio de um gráfico Burndown. Esta ferramenta permitirá melhor visibilidade do ritmo de desenvolvimento do projeto e dará melhor auxílio caso novas funcionalidades e Sprint tenham que ser criadas.

As sprints iniciais planejadas para o projeto serão descritas na seção 4.1.1, uma vez que novas histórias podem ser levantadas pelo cliente e para realizá-las, podem ser necessárias o planejamento e execução de novas sprints.

Ao final de cada sprint será feita a entrega do artefato e apresentação para o Product Owner e Scrum Master para homologação e devidas anotações e observação. Logo em seguida

também será demostrada a funcionalidade para comissão do estacionamento do campus para feedback.

O desenvolvimento das histórias de usuário em cada Sprint será subdivida em tarefas, que compõem a funcionalidade, seguindo os passo da seguinte maneira - criação de testes para a tarefa, implementação da tarefa, seguido para testes de campo e por fim refatoração da implementação. Cada história concluída será anotado seu tempo, os impedimentos caso ocorram descrevendo os motivos e o tempo para solução. Estes dados serão de ajuda para compor o gráfico Burndown, da qual será utilizado para medir o ritmo do projeto.

4 DESENVOLVIMENTO

Neste capítulo são apresentados os artefatos de projeto elaborados para o desenvolvimento do sistema proposto. Mais precisamente, na seção 4.2.3 é apresentada a arquitetura do sistema, uma vez que ela é composta também por uma infraestrutura de hardware. A seção 4.2 as aplicações que compõe o sistema, assim como será apresentando a comunicação entre o agente e a aplicação Web. Na seção 4.4 é apresentada a modelagem do banco de dados da aplicação. Por fim, na seção 4.5 é apresentado os protótipos das páginas a serem desenvolvidas.

4.1 HISTÓRIAS DE USUÁRIOS

Esta seção apresenta o backlog do produto. O backlog contém as principais histórias do usuário levantadas inicialmente em reuniões com o papel de *Product Owner*, demonstrada no capitulo anterior 3. Para este projeto, o papel de *Product Owner* será representado pelo professor orientador e coorientador do projeto, e também por um representante da comissão do estacionamento do campus. O *Srcum Master* será o orientador do projeto e por fim, o time de desenvolvimento ou *Scrum Team*, será composto pelo autor do projeto.

O *Product Backlog* inicial é composto com base nos seguintes critérios que forma o escopo da aplicação:

1. O sistema deverá reconhecer uma placa de carro com a utilização do *stream* de vídeo.
2. O sistema deverá ler os dados de um motorista assim como, cadastrar, atualizar, e remover um motorista e a placa do seu carro.
3. O sistema deverá cadastrar usuários com perfil de operador e administrador, bem como remover e alterar tais usuários.
4. O sistema deverá notificar quando uma placa for reconhecida e se a placa está cadastrada ou não.

Com base nestes critérios mínimos, o *Product Backlog* é formado pelas histórias do usuário descritas nas Tabelas, 1, 2, 3. As histórias estão ordenadas por grau de importância conforme definição do *Product Owner*. Estas histórias serão subdividas em tarefas em cada Sprints, cada tarefa é uma divisão das partes da funcionalidade da qual a história descreve, após feita esta divisão, o desenvolvimento será iniciado, após concluído esta tarefa os testes de campo será executado, testes este que será executado em ambiente controlado para verificação que as outras funcionalidades não foram impactadas, além do auxilio de testes de unidades e testes de funcionalidades que serão feitos ao longo do desenvolvimento da tarefa.

Tabela 1 – Histórias dos Administradores.

ID	Prioridade	História
1	1	Como administrador gostaria de atualizar meus dados.
2	2	Como administrador gostaria de cadastrar novos operadores no sistema, assim como atualizar e remover um operador existente
3	3	Como administrador gostaria de adicionar autorização de administrador a um operador já cadastrado, como também remover a autorização
4	4	Como administrador gostaria de adicionar um novo motorista ao sistema, como atualizar seus dados ou removê-lo do cadastro os dados seria nome, R.A, uma foto para identificação.
5	5	Como administrador gostaria de adicionar uma única placa de carro a um motorista já cadastrado.
6	6	Como administrador gostaria de bloquear o acesso de um motorista já cadastrado, temporariamente, assim como desbloqueá-lo, assim como ver a lista dos bloqueados.
7	7	Como administrador do sistema gostaria de ver a lista de motorista que teve acesso no dia da minha consulta.

Tabela 2 – Histórias dos Operadores.

ID	Prioridade	História
8	1	Como operador gostaria de liberar o acesso a um motorista mediante a sinalização do sistema que ele tem autorização.
9	8	Como Operador gostaria de atualizar meus dados.

Tabela 3 – Histórias dos Motoristas.

ID	Prioridade	História
10	1	Como motorista gostaria de atualizar ou remover uma placa do meu carro no sistema, assim como os meus dados cadastrados no sistema.

4.1.1 Sprints

Nesta sessão será descrita o desenvolvimento das histórias, dividas em Sprints como dito anteriormente, com prazo de 15 dias corridos, neste intervalo será desenvolvido a funcionalidade dos sistemas descritos 4.2. Cada nova funcionalidade criada seguira o seguinte passos:

- Desenvolvimento;
- testes de campo;
- testes funcional; e
- refatoração.

O planejamento das Sprints será inicialmente será:

- **Sprint 1**:
  1. Será executado as tarefas para configuração do ambiente do, instalação do S.O no Raspberry as configurações para comunicação na rede.

2. Instalação da câmera e configuração da comunicação entre o Raspberry e a câmera.
3. Implementação das histórias de ID 1 à 3

- **Sprint 2**:
  1. Será feito a configuração instalação a biblioteca OpenALPR e treinamento do Agente.
  2. Implementação das histórias de ID 4 à 5 e 8.
- **Sprint 3**:
  1. Teste de campo de desempenho do Agente ALPR, neste teste será executado em ambiente controlado o funcionamento do reconhecimento das placas.
  2. analise de desempenho, incluindo taxa de erro e de acerto.
  3. Implementação das histórias de ID 4 à 5 e 8.
- **Sprint 4**:
  1. Otimização do desempenho do Agente ALPR, otimização da captura do Stream de vídeo.
- **Sprint 5**:
  1. Implementação das histórias de ID 9 à 10, teste de campo para avaliação do desempenho do Agente.

4.2 APLICAÇÕES

Nesta subseção será demonstrado as aplicações que corresponde ao sistema. O desenvolvimento do sistema será divido em duas aplicações, a primeira a aplicação que contempla o serviço do objeto inteligente com a utilização da biblioteca OpenALPR, é servidor Web cliente que consumira os dados o objeto inteligente.

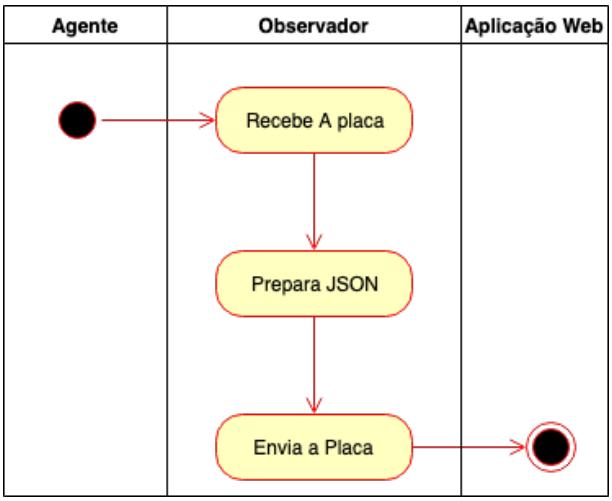
4.2.1 Aplicação de Integração

O Raspberry será o objeto inteligente da qual detém a responsabilidade de comunicar com a aplicação web fora do objeto inteligente instalada em um servidor web dentro da rede. A biblioteca OpenALPR servirá como componente auxiliar a execução do algoritmo de reconhecimento, processando o stream de vídeo. O OpenALPR estará conectada a câmera via MJPEG. Uma vez encontrada uma placa de carro, dando as devidas atenções ao processamento da imagem. A aplicação de integração, a qual, será escrita em Java e tem a responsabilidade de receber os reconhecimentos das placas e enviar para aplicação Web a placa reconhecida para encontrar o motorista cadastrado, além de fazer a comunicação com a porta GPIO.

Aplicação de integração aqui denominada observador, consistem receber o reconhecimento do agente, a placa do carro, logo em seguida que receber, processara a informação convertendo o dado no formato JSON para o envio, assim notificando a aplicação Web através de uma requisição HTTP com método POST, contendo as informações do reconhecimento, placa reconhecida, como ilustra Figura 1 demostra a atividade do observador. A aplicação de

integração, observador, ainda tem poderá receber a informação da aplicação Web para enviar um sinal para porta ativar GPIO, entretanto esta função estará presente no sistema mas, o escopo do projeto não contempla a instalação de outro periférico, como por exemplo, para uso ativar a cancela abrindo-a.

Figura 1 – Diagrama de atividade aplicação de integração



Fonte: autor

4.2.2 Aplicação Web

A aplicação Web será construída com a utilização das tecnologias Java, para auxiliar o desenvolvimento o framework Spring Boot, além de usar como engine Thymeleaf para telas no lado do servidor. Aplicação web uma vez que receber a requisição do observador, aplicativo que executa no Raspberry. como citado anteriormente receberá os dados do processamento em formato JSON. Então, fará uma busca na sua base de dados pela placa encontrada. O resultado será enviado para tela do cliente, como ilustra a Figura 2.

O URL que a aplicação web recebe as requisições de placa exemplificada na quatro 1. Esta aplicação tera a responsabilidades de alterar, remover e adicionar novos administradores e operadores, assim como disponibilizar uma interface gráfica para os motorista poderem alterar os dados do seu carro. Esta aplicação tem como objetivo a interação com o usuário.

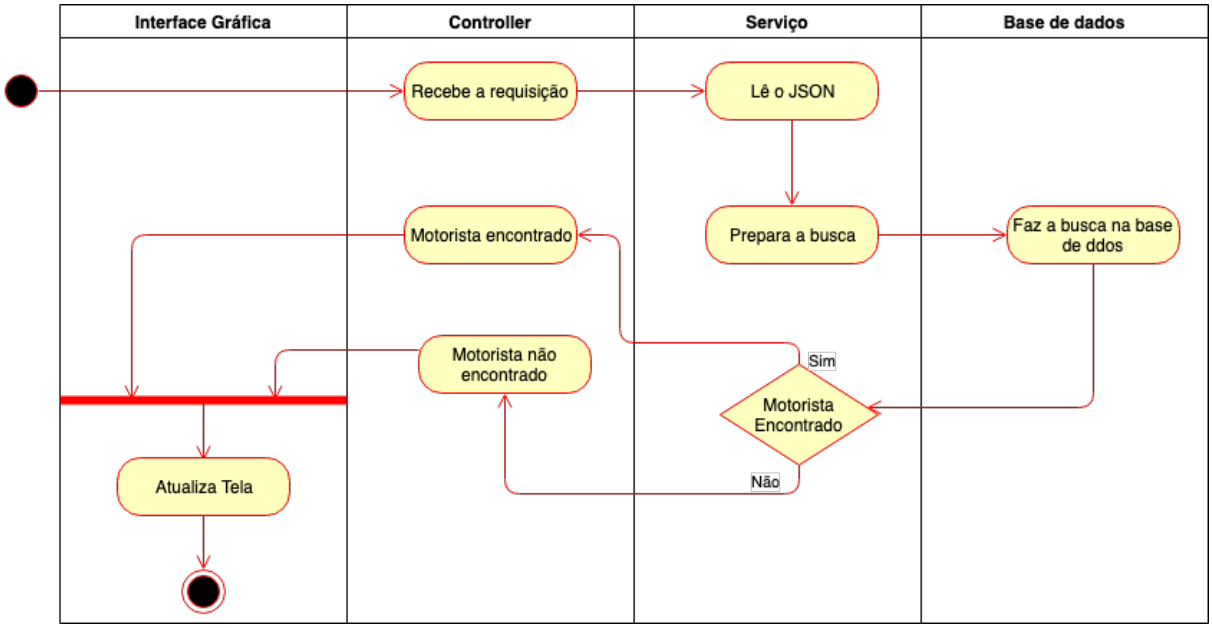
A na seção 4.4 será demonstrado o projeto do bando de dados do projeto, logo fica apenas necessário o funcionamento da aplicação Web.

Quadro 1 – Rota de Envio da Placa reconhecida - URL.

Método HTTP	**URI**	**Dados**
POST	/recognized-plate	AAA-9999

Fonte: autor

Figura 2 – Diagrama de atividade aplicação web



Fonte: autor

Como citado no subseção 4.2.1 o observador tem em sua programação comunicação com uma porta GPIO para uso, por exemplo de cancela, esta ação será ativada pela execução de uma ação provida na interface do operador, quando uma motorista for identificado, esta ação enviaram uma requisição HTTP com método GET para ativar a porta, contudo, como citado o projeto não contempla a instalação de perifericos para este fim.

Será utilizado a tecnologia WebSocket para atualizar a tela do usuário acada reconhecimento que a aplicação Web receberá do observador.

4.2.3 Arquitetura do Sistema

O fluxo de atividade do sistema é melhor exemplificado com auxílio da figura 3. Esta ilustra uma visão global do fluxo de atividade das quais o sistema executará quando uma motorista entra no campo de visão do sistema. Este campo será fixado para obter um melhor ângulo.

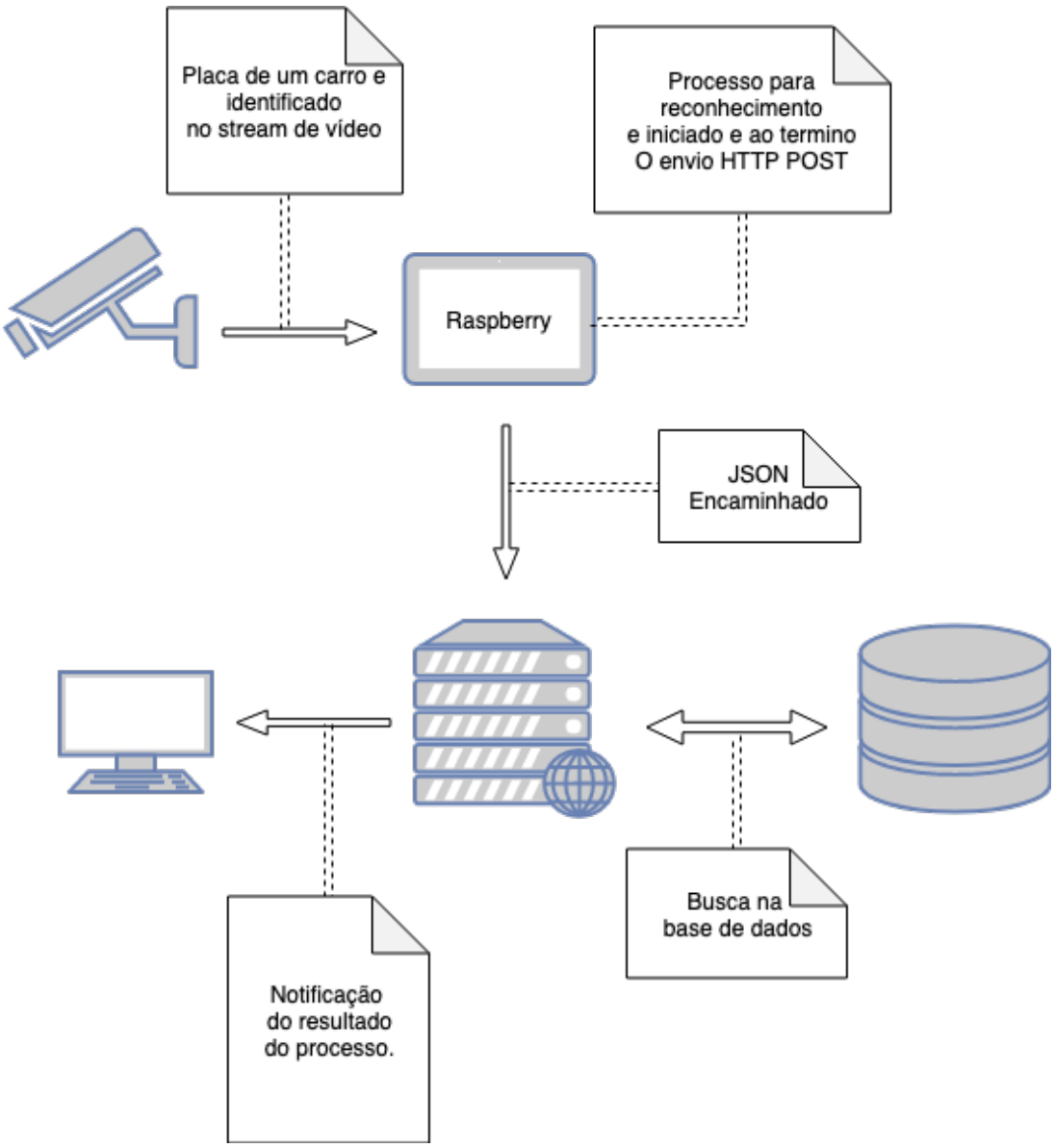
4.3 TREINAMENTO DO AGENTE

O reconhecimento da placa, como explicada nas seções anteriores será executado dentro do objeto inteligente do Raspberry PI. Para esta tarefa, o algoritmo ALPR utilizado será o da biblioteca OpenALPR.

O treinamento que será em feito em duas fases:

- Instalação e configuração
- Aprendizagem;

Figura 3 – Ilustração fluxo de processo de reconhecimento da placa



Fonte: autor

Na fase de instalação, a biblioteca OpenALPR será integrada a uma aplicação Java que terá como responsabilidade atuar como um observador do OpenALPR, como explicado na 4.2. Para obter melhor desempenho e qualidade, algumas configurações e ajustes finos poderão ser necessárias com base nas informações da documentação.

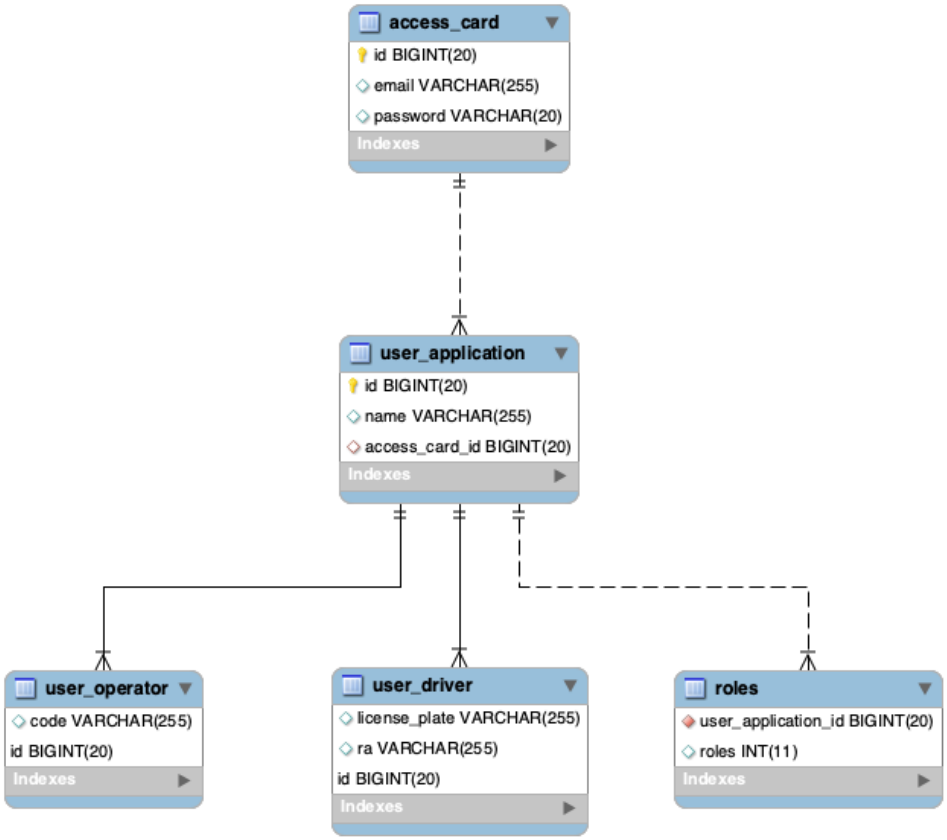
O treinamento consiste em ensinar o agente, o algoritmo ALPR, a reconhecer os padrões de placas de carros no modelo brasileiro tradicional e do Mercosul. Para poder segmentar os caracteres, o agente será treinado com a técnica de aprendizagem de máquina supervisionada em duas etapas. Este treinamento consistirá de aproximadamente 200 a 300 imagens de placas de carros, de ambos os modelos, e tradicional e Mercosul, com um tempo aproximado de 20 horas de processo de aprendizagem. Este processamento é importante para conhecer os padrões de localização da placa do carro no *Stream* de video. Assim, o algoritmo conseguirá separar

a placa do carro através da segmentação por borda, seguindo para reconhecer os caracteres correspondente da placa segmentando-os um a um através da técnica de segmentação por corte. Por fim, serão realizados os testes de comportamento e desempenho do agente para melhorar a segunda fase de treinamento, onde o agente será submetido a avaliação de reconhecimento desta vez com *Stream* de vídeo para comprovar se o agente consegue identificar com uma boa performance as placas de carro.

4.4 PROJETO DO BANCO DE DADOS

A Figura 4 apresenta a modelagem da base de dados do sistema Web, que fará a interação com os usuários. Por sua vez, a aplicação no objeto inteligente não necessitarão de banco de dados. Para o processo de aprendizagem será necessário apenas de um arquivo de texto, contendo os *Dataset*, dados de informação das entradas e saídas da aprendizagem do OpenALPR, a qual foi demostrada como aprendizagem supervisionada.

Figura 4 – Modelagem do banco de dados



Fonte: autor

Na Figura 4 pode-se aferir que o sistema Web terá uma entidade chamada *AccessCard*. Esta entidade será responsável por salvaguardar as credenciais dos usuários, seja ele qualquer um dos tipos de usuário que o sistema Web terá, que são eles:

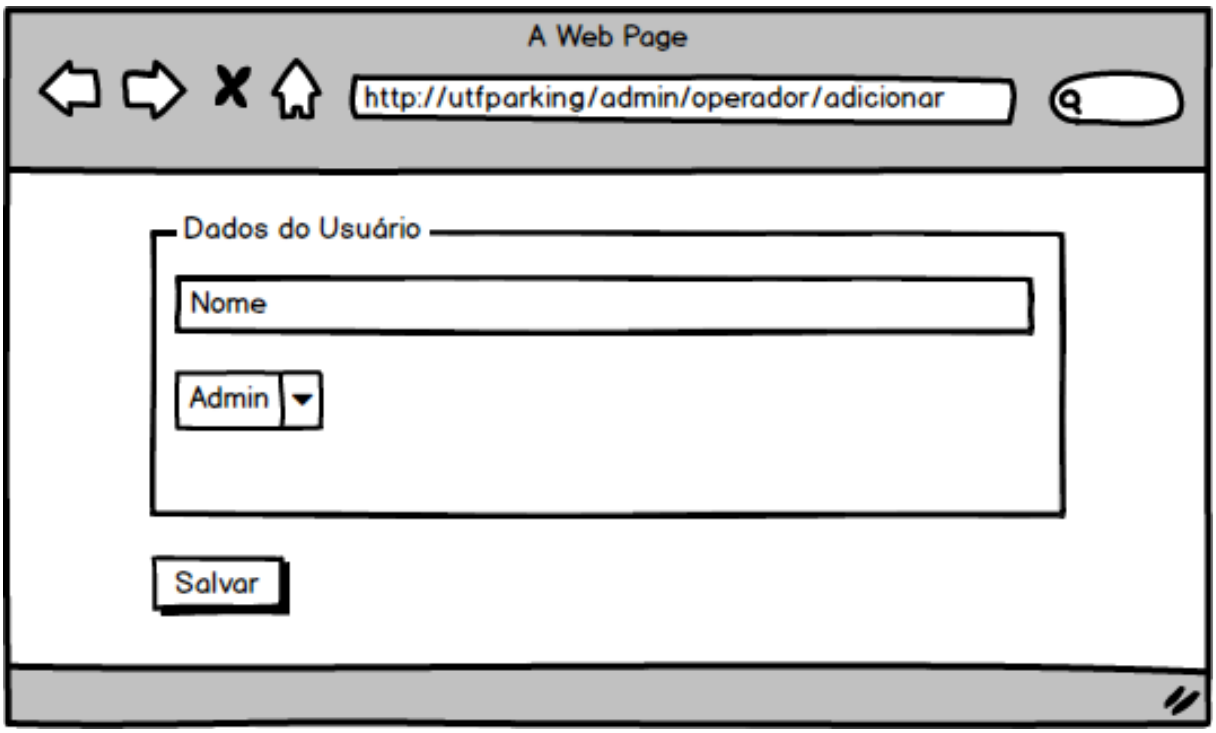
1. **UserOperador**: usuário, administrador ou operador, variantes que depende do perfil, atribuído a ele, este usuário será a representação do usuário operador do sistema, funcionários da empresa de segurança, componentes da comissão do estacionado, são exemplos deste tipo de usuário no mundo real.
2. **UserDriver**: como nome sugere, esta entidade será a representação do o usuário do motorista, o usuário que pede acesso ao estacionamento do campus.

4.5 PROTOTIPAGEM DE TELAS

Nesta seção será demonstrado o projeto das interfaces do usuário e algumas funcionalidades iniciais que o sistema terá.

Uma funcionalidade que é aferida do *Product Backlog* do administrador, Tabela 1, consistem nas histórias 1 e 2 referente ao cadastro e atualização de dados de operadores. O protótipo de tela é ilustrado na Figura 5. Também representa a história de ID 3 da Tabela 2, assim como a história para alterar os dados do administrador, como também representada a história de ID 10, para esta história o campo de perfil, ilustrado na Figura 5 o campo de seleção será bloqueado, haja vista que um operador não poderá alterar seu perfil, para administrador.

Figura 5 – Adicionar Usuário operador ou Administrador

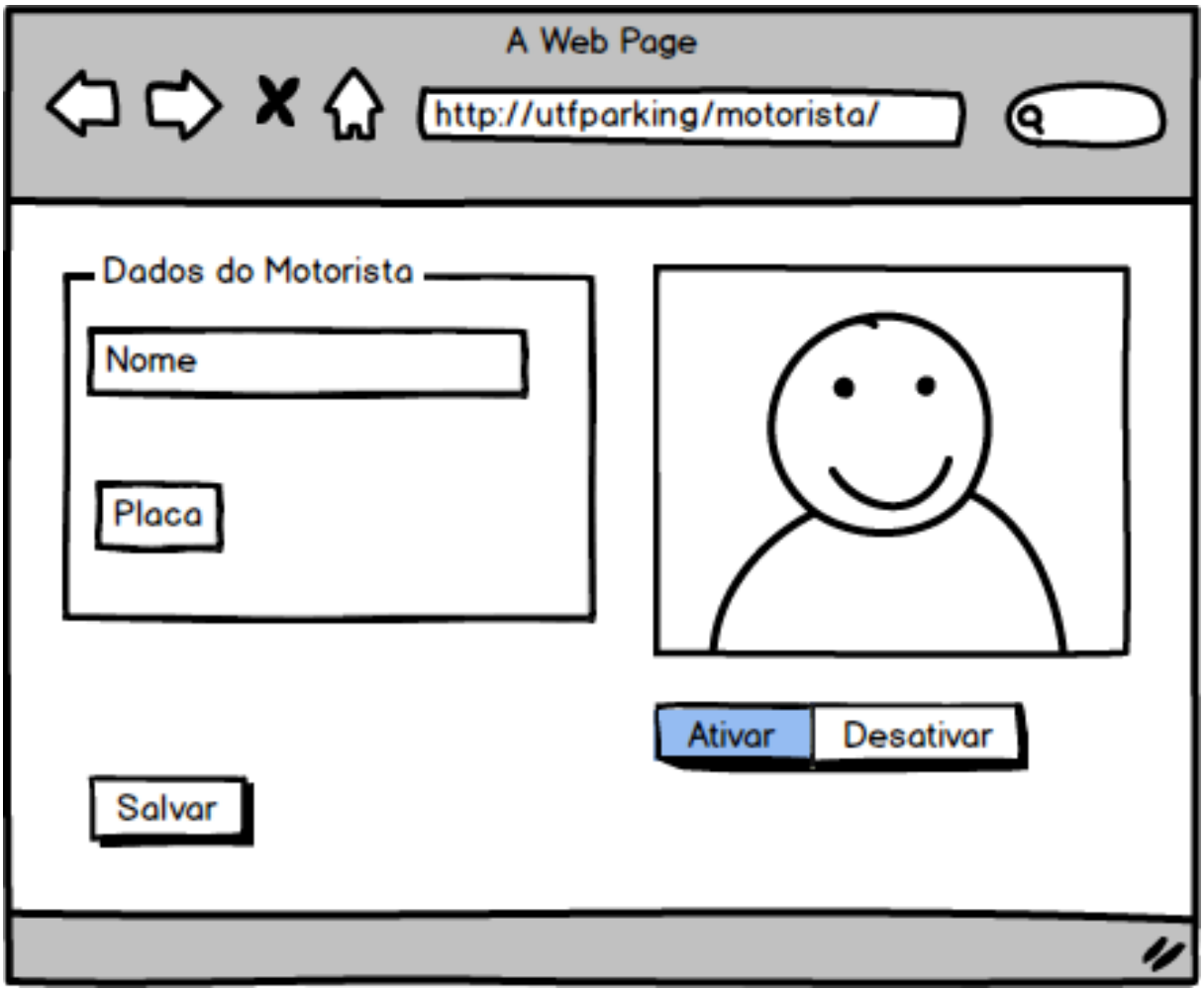


Fonte: autor

Conforme o *Product Backlog* como referenciado na Tabela 1 com ID 5, a interface do usuário para esta funcionalidade é ilustrada na Figura 6. Esta ilustração também está representando a história do administrador, também referenciada na Tabela 1, com identificador

7 no sistema. Esta funcionalidade é apresentado pelo botões: Ativar e Desativar na ilustração a Figura 6.

Figura 6 – Adicionar Motorista

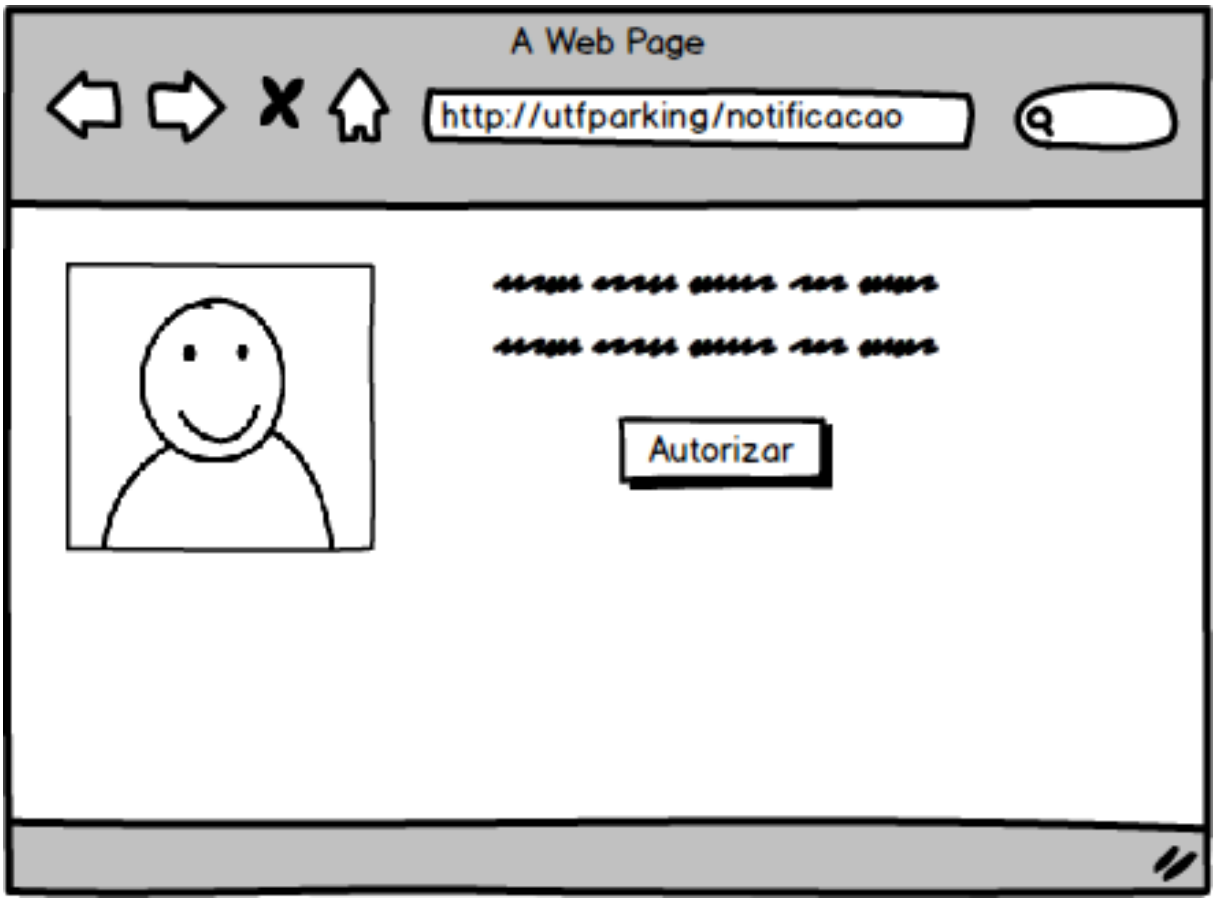


Fonte: autor

A Figura 7 ilustra a interface do usuário para a funcionalidade do sistema, da história 9, do *Product Backlog* inicial referenciada na tabela 2, uma vez identificado a placa de seu carro, e encontrado os dados necessários, a aplicação web exibirá, como mostra a Figura 7, uma foto de identificação, os dados do usuário, e um botão de ação para autorizar o acesso, como solicita a história 9. É necessário elucidar novamente que esta funcionalidade ativada pelo botão autorizar ilustrado na Figura 7está ativa, mas depende de instalação de outro periférico, tal como cancela, para total funcionamento desta funcionalidade da qual ative a porta GPIO do Raspberry.

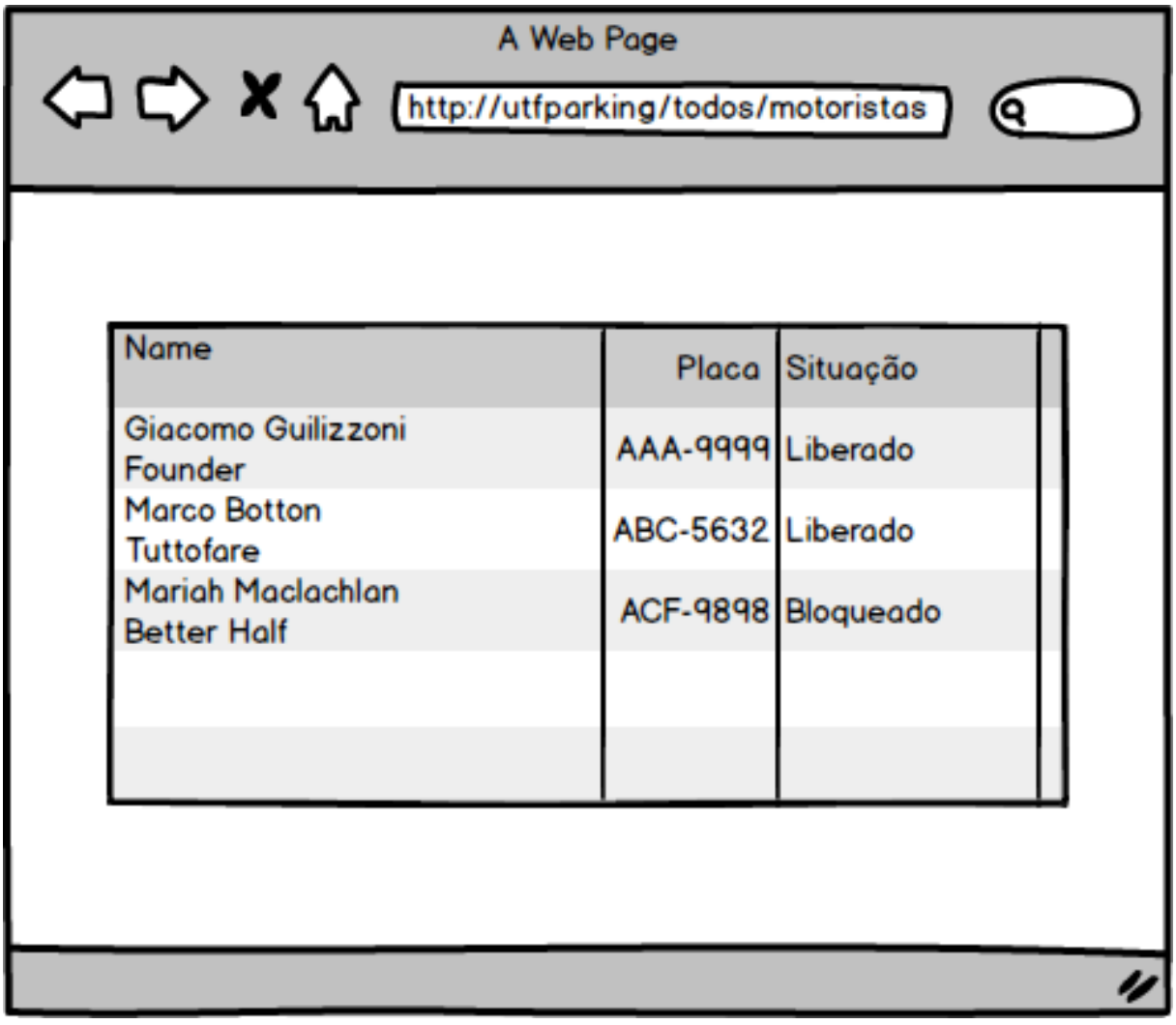
Por fim a Figura 8. ilustra a funcionalidade da aplicação Web, respondendo a história 8, referencia na tabela 1.

Figura 7 – Autorizar Motorista



Fonte: autor

Figura 8 – Lista de Motorista



Fonte: autor

5 CONCLUSÃO

5.1 TRABALHOS FUTUROS

O projeto, não abrange coisas comuns em sistema de gerenciamento de estacionamento, haja vista que seu escopo é apenas do controle de acesso, quem pode ou não pode entrar, como base neste escopo o projeto se concentro na automatização desta tarefa, desenvolver um agente racional, utilizando-se de visão computacional para o reconhecimento das placas de carros como base para conceber o acesso, mas este projeto pode ser incrementando para receber funcionalidades descrita anteriormente, outras mais. como:

1. Adicionar funcionalidade para controle de vagas;
2. Adicionar funcionalidade controle de entrada e saída dos usuários;

São exemplos de novas funcionalidades iniciais para aprimorar o projeto, além dele poder ser integrado ao sistema oficial da UTFPR, deixa automatizado desde o pedido de acesso ao acesso do estacionamento da UTFPR do campus Guarapuava.

5.2 CONSIDERAÇÕES FINAIS

O sistema tem como objetivo tentar solucionar o problema atual de controle de acesso ao estacionamento, automatizando o processo e não dependendo mais de adesivos que levam a tantos problemas. Portanto, esta proposta define o escopo de um sistema que satisfaz as necessidades do campus no quesito de entrada e saída de veículos utilizando ALPR e dá plataforma Raspberry. Como demonstrado nos trabalhos futuros, o sistema ainda poderá, fazer o controle de vagas, adicionar logs de informação de entrada e saída do campus, de cada motorista.

Referências

ALVARENGA, E. P. D. Optical character recognition for automated license plate recognition systems. MAXWELL, 2017. Citado na página 5.

ANAGNOSTOPOULOS, C.-N. E. et al. License plate recognition from still images and video sequences: A survey. **IEEE Transactions on intelligent transportation systems**, IEEE, v. 9, n. 3, p. 377–391, 2008. Citado na página 7.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Computer Networks**, v. 54, n. 15, p. 2787 – 2805, 2010. ISSN 1389-1286. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1389128610001568>. Citado na página 8.

DBA. **dba.eng.br**. 2019. Disponível em: <https://dba.eng.br/br/solucoes/ocr/ocr-alpr>. Acesso em: 21 de abril de 2019. Citado na página 3.

FIELDING, R. T.; TAYLOR, R. N. **Architectural styles and the design of network-based software architectures**. [S.l.]: University of California, Irvine Doctoral dissertation, 2000. v. 7. Citado na página 8.

FILHO, O. M.; NETO, H. V. **Processamento digital de imagens**. [S.l.]: Brasport, 1999. Citado na página 6.

FOUNDATION, R. P. **Documentação Raspberry**. Disponível em: <https://www.raspberrypi.org/documentation/usage/gpio/>. Acesso em: 21 de abril de 2019. Citado na página 10.

FRANÇA, T. C. de et al. **Web das coisas: conectando dispositivos físicos ao mundo digital**. 2011. Citado 2 vezes nas páginas 7 e 8.

GENETEC. **genetec.com**. 2019. Disponível em: <https://www.genetec.com/br/solucoes/todos-os-produtos/autovu>. Acesso em: 21 de abril de 2019. Citado na página 3.

JOHNSON, R. et al. The spring framework–reference documentation. **interface**, v. 21, p. 27, 2004. Citado na página 11.

LUCKOW, D. H.; MELO, A. A. de. **Programação Java para a WEB**. [S.l.]: Novatec Editora, 2010. Citado na página 11.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à visão computacional usando opencv. **Revista de Informática Teórica e Aplicada**, v. 16, n. 1, p. 125–160, 2009. Citado 2 vezes nas páginas 5 e 6.

MEERKAT. **Meerkat.com/alpr**. 2019. Disponível em: <https://www.meerkat.com.br/solution_automatic_license_plate_recognition.html>. Acesso em: 21 de abril de 2019. Citado na página 3.

RUSSEL STUART; NORVIG, P. **Inteligência Artificial**. 2. ed. [S.l.]: Campos, 2013. Citado 2 vezes nas páginas 4 e 5.

SABBAGH, R. **Gestão Ágil para projetos de sucesso**. São Paulo: Casa do Código, 2013. Citado na página 12.

SANTOS, B. P. et al. Internet das coisas: da teoria à prática. **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos**, 2016. Citado na página 10.

SCURI, A. E. Fundamentos da imagem digital. **Pontifícia Universidade Católica do Rio de Janeiro**, 1999. Citado na página 6.

SUTHERLAND, K. S. e. J. **Guia do Scrum**. 2013. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em: 21 de abril de 2019. Citado na página 12.

TAKATUZI, F. K. O. **Adaptive-IDT: algoritmo incremental para aprendizagem de árvores de decisão adaptativas**. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2017. Citado na página 5.

TEAM, T. T. **Thymeleaf**. 2013. Disponível em: <https://www.thymeleaf.org/documentation.html>. Acesso em: 21 de abril de 2019. Citado na página 12.

TECHNOLOGY, I. O. **OpenALPR Documentation**: Getting started. [S.l.], 2017. Disponível em: <http://http://doc.openalpr.com/>. Acesso em: 12 de maio de 2019. Citado na página 10.

VERNON, V. **Implementando o Domain-Driven Design**. [S.l.]: Alta Book, 2016. v. 1. Citado na página 9.