UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

BACHARELADO EM ENGENHARIA DE SOFTWARE

MARCELO HENRIQUE PEREIRA

COPLAV: UMA PROPOSTA DE APLICAÇÃO MÓVEL PARA CONSULTA DE SITUAÇÃO VEICULAR EM BASE DE DADOS CENTRALIZADA

QUIXADÁ

2017

MARCELO HENRIQUE PEREIRA

COPLAV: UMA PROPOSTA DE APLICAÇÃO MÓVEL PARA CONSULTA DE SITUAÇÃO VEICULAR EM BASE DE DADOS CENTRALIZADA

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Orientador: Prof. Dr. Cristiano Bacelar de Oliveira

QUIXADÁ

2017

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

P493c Pereira, Marcelo Henrique.
Coplav: uma proposta de aplicação móvel para consulta de situação veicular em base de dados centralizada / Marcelo Henrique Pereira. – 2017.
47 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2017.
Orientação: Prof. Dr. Cristiano Bacelar de Oliveira.

1. Android (Recurso eletrônico). 2. Veículos a motor. I. Título.

CDD 005.1

MARCELO HENRIQUE PEREIRA

COPLAV: UMA PROPOSTA DE APLICAÇÃO MÓVEL PARA CONSULTA DE SITUAÇÃO VEICULAR EM BASE DE DADOS CENTRALIZADA

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Aprovada em: ___/___/___

BANCA EXAMINADORA

Prof. Dr. Cristiano Bacelar de Oliveira (Orientador)
Universidade Federal do Ceará – UFC

Prof. Me. Enyo José Tavares Gonçalves
Universidade Federal do Ceará - UFC

Prof.ª Dra. Ingrid Teixeira Monteiro
Universidade Federal do Ceará - UFC

AGRADECIMENTOS

Às minhas mães Marleide e Lindalva que sempre fizeram o possível e o impossível para que eu tivesse uma boa educação, e por sempre acreditarem no meu potencial. A todos aqueles que contribuíram de forma direta ou indireta para o meu trabalho. Ao meu orientador Professor Cristiano Bacelar que aceitou me guiar no projeto de pesquisa e na realização do trabalho, pela paciência e dedicação ao me orientar.

“Todos os dias, faça alguma coisa que você tem medo.”

(Eleanor Roosevelt)

RESUMO

O índice de recuperação de veículos roubados ou furtados no Brasil ainda pode ser considerado baixo. Em 2014, apenas 17,54% desses veículos foram recuperados. Pensando nisso, foi feita uma proposta de aplicação móvel para o sistema operacional *Android* para obtenção de situação legal de veículos de todo o território brasileiro a partir da imagem ou inserção manual da placa, como uma forma de aumentar esse índice de recuperação. Para uma consulta por meio de imagem, foi utilizada a *OpenALPR* para identificar os caracteres presentes nas placas. Após essa identificação, é feita uma busca em uma base de dados centralizada abastecida pelos próprios usuários e também na base de dados do Sinesp, unidade governamental responsável por fornecer informações sobre a situação legal de um veículo. A aplicação proposta foi desenvolvida com a utilização de testes unitários e posteriormente foram realizados testes de cobertura para as principais partes do código da aplicação móvel. Ao fim do desenvolvimento, alguns usuários utilizaram a aplicação para fins de validação.

Palavras-chave: *Android*. Reconhecimento. Placa. Veículos.

ABSTRACT

The recovery rate of stolen vehicles in Brazil can still be considered low. In 2014, only 17.54% of these vehicles were recovered. With this in mind, a proposal was made for a mobile application for the *Android* operating system to obtain legal status of vehicles from all over the Brazilian territory from the image or manual insertion of the plate, as a way to increase this rate of recovery. For an image query, *OpenALPR* was used to identify the characters present on the plates. After this identification, a search is done in a centralized database supplied by the users themselves and also in the database of Sinesp, a governmental unit responsible for providing information on the legal status of a vehicle. The proposed application was developed with the use of unit tests and subsequent coverage tests for the main parts of the mobile application code. At the end of the development, some users used the application for validation purposes.

Keywords: *Android*. Recognition. Plate. Vehicles.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

Sinesp	Sistema nacional de informações de segurança pública, prisionais e sobre drogas
Serpro	Serviço federal de processamento de dados
Senasp	Agência nacional de segurança pública do ministério da justiça
Denatran	Departamento nacional de trânsito
API	*Application Programming Interface*
SOAP	*Simple Object Access Protocol*
XML	*Extensible Markup Language*
AWS	*Amazon Web Services*
SAAS	*Software as a Service*
PAAS	*Platform as a Service*
IAAS	*Infrastructure as a Service*
AWS	*Amazon Web Services*

SUMÁRIO

1 INTRODUÇÃO

A segurança pública, em âmbito estatal, deve garantir aos habitantes bom convívio, trabalho e diversão, protegendo-os dos riscos aos quais estão expostos. Um aspecto, ainda deficiente, é o baixo índice de veículos que são devolvidos aos donos após roubos ou furtos. Dos 513.023 veículos que foram roubados ou furtados no ano de 2014, apenas 90.000 foram recuperados, um índice de apenas 17.54% de recuperação (SNSP, 2015). Somente no ano de 2015, esse número foi de 509.978 veículos. Em números, absolutos houve uma redução de apenas 0,6% em relação ao ano de 2014 (FBSP, 2016). Índice ainda baixo levando em consideração o número total de ocorrências.

Outro problema de segurança, é a clonagem de veículos. Ela consiste em trocar a placa de um carro, por uma outra com os mesmos caracteres da placa de um veículo que contenha as mesmas características, resultando em dois veículos com as mesmas características e placas. A fim de solucionar esse problema, foi adicionado um QR-Code nas placas e documentos de todos os veículos fabricados a partir de 2017. As informações da placa do veículo podem ser obtidas por qualquer cidadão, a partir da leitura do QR-Code, e se referem à fabricação da placa, tais como: código do fabricante, lote e data de fabricação (BRASIL, 2017). Porém, apenas o QR-Code não é suficiente para evitar que alguém adquira um carro clonado, pois este não fornece informações mais detalhadas, como o número do chassi ou situação legal do veículo ao qual ela pertence. Isto faz com que o cidadão não tenha a garantia da legitimidade de um veículo apenas consultando as informações do QR-Code da placa.

Com o grande aumento do uso de dispositivos móveis e com a ascensão do sistema operacional *Android* como o mais usado do mundo que se conecta à internet (HAMANN, 2017), surge a oportunidade do desenvolvimento de uma aplicação que possa informar de maneira rápida e eficaz a situação de um veículo. Por se tratar de uma informação sensível e disponibilizada apenas para órgãos públicos, não havia aplicações que oferecessem esse tipo de serviço. A aplicação pioneira nesse quesito foi a Sinesp Cidadão, desenvolvida em 2014 pelo Serpro e gerenciada pela Senasp, que qualquer pessoa pode ter acesso. O SERPRO possui acesso aos dados do Denatran, o que permitiu o desenvolvimento da aplicação. Somente no mês de janeiro de 2015, foram realizadas 930.501 pesquisas de restrição a partir da placa do veículo e, nesse mesmo período, foram encontrados 7.900 veículos que tiveram buscas realizadas na aplicação (SNSP, 2015).

Com isso, o objetivo desse trabalho foi desenvolver o Coplav. Uma aplicação móvel

para o sistema operacional *Android* para consulta da situação legal de veículos de todo o território brasileiro, verificando se um veículo possui denúncia de roubo, furto ou clonagem, a partir da imagem ou inserção manual da placa do veículo, contribuindo com o aumento do índice de recuperação de veículos. A aplicação contará também com um mapa que irá exibir locais de incidências de roubos e furtos e com a opção de denunciar uma ocorrência.

Para isso, serão analisadas técnicas de reconhecimento de caracteres contidos em placas brasileiras. Identificar de maneira correta os caracteres da placa é desejável para uma busca rápida e obtenção dos resultados correspondentes ao veículo.

A aplicação contará com os métodos já existentes de captura dos dados e tem como público alvo a população brasileira, mais especificamente pessoas que desejem adquirir veículos, bem como autoridades policiais que poderão realizar a consulta durante abordagens. O Capítulo 2 contará com trabalhos relacionados, os quais apresentam aspectos relacionados com a aplicação proposta.

2 TRABALHOS RELACIONADOS

2.1 Protótipo para informatização de controle de estacionamento em área azul no município de Blumenau-SC

No trabalho de GOLL (2013) é apresentado um protótipo de uma aplicação móvel para a plataforma *Android* para controle de estacionamentos da Zona Azul da cidade de Blumenau-SC. A aplicação conta com um módulo web para consulta dos dados a ser feita pelo aplicativo *Android* e está dividida em dois níveis de acesso:

a) Ao nível de usuário, são obtidas informações a respeito dos locais de estacionamento de Zona Azul, aquisição de créditos para o estacionamento e acesso às notificações recebidas através da aplicação.
b) Ao nível de monitor, é possível emitir notificações e consultar veículos a partir da placa.

As consultas dos veículos são realizadas a partir da inserção manual dos caracteres das placas e do número que identifica o estacionamento. A busca é realizada em uma base de dados centralizada e as informações referentes ao veículo são exibidas (GOLL, 2013). As informações não fazem referência à situação legal do veículo, logo não informam se é roubado ou furtado. Com isso, não possui aplicação prática para consulta de situação veicular.

Nesse projeto de pesquisa, também será utilizado um módulo web para verificação dos dados da placa, assim como o que foi utilizado por GOLL (2013). Porém contará apenas com um nível de acesso, que será utilizado pelos usuários para a realização da consulta. Além disso, as informações que serão recebidas pela aplicação *Android* não são de caráter informativo sobre o tempo de estacionamento do veículo buscado, mas sobre sua situação legal.

2.2 Sinesp Cidadão

Com o Sinesp Cidadão, é possível verificar a situação de um veículo a partir de sua placa. Ele acessa diretamente a base de dados do SERPRO, que por sua vez é abastecida a partir de dados do DENATRAN. Para verificar se um veículo é roubado ou furtado, o mecanismo de busca recebe dados da placa preenchidos manualmente pelo usuário.

Além de informações a respeito de situação veicular, o aplicativo também possui funcionalidades de consulta de mandado de prisão em aberto e de pessoas desaparecidas.

O Sinesp cidadão possui apenas um método para obtenção dos dados da placa de um veículo, que será utilizado neste projeto. Adicionalmente será inserida a funcionalidade na aplicação para captura de imagens das placas para que seja feita a verificação da situação do veículo. Com isso, o projeto proposto contará com dois modos de verificação veicular, por texto e por imagem.

2.3 Localização e validação de regiões candidatas de placas a partir da análise de imagens digitais

No trabalho de SALES (2010) é apresentada uma solução para identificação de uma possível área de placa em imagens digitais, utilizando técnicas de binarização por limiares globais e detecção de bordas.

A principal etapa de reconhecimento de placas, é a detecção da área de uma placa em imagens digitais. Para se conseguir uma área candidata de placa, o sistema foi dividido nas seguintes fases:

a) Aplicar filtros para realce das imagens e opcionalmente converte para a escala de cinza;
b) Localizar áreas de interesse, que consiste em letras escuras sob fundo claro ou letras claras sob fundo escuro e formato retangular;
c) Validação da possível área de placa, com a exclusão de áreas que não contém a placa. Para isso, é utilizado um algoritmo classificador, levando em consideração as mesmas condições do item anterior.

Como validação do sistema, foram utilizadas 906 imagens obtidas a partir de equipamentos em operação real. Regiões que continham parcialmente a placa, foram classificadas como regiões que não continham placa. Para determinar a sua eficiência, foram submetidas 23.287 imagens com possíveis regiões de placas, das quais 5.941 continham a placa e 17.896 que não continham a placa veicular. Foram aplicadas as 906 imagens ao algoritmo proposto e como resultado global obteve-se uma taxa de acerto de 98,90% (SALES, 2010).

3 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados os principais conceitos para a fundamentação desse projeto.

3.1 Reconhecimento de caracteres alfanuméricos

Para o obter os caracteres das placas de veículos, são necessárias três etapas básicas, que consistem na localização da imagem correspondente à área da placa do veículo, extração dos caracteres e reconhecimento dos caracteres. A principal fase é a de localização da placa na imagem do veículo, pois sem uma área que possa corresponder a uma placa, não seria possível executar as outras duas etapas (SALES, 2010).

Após feita a localização da área da placa dentro da imagem de todo o veículo, podemos usar bibliotecas de reconhecimento de caracteres ópticos (OCR) para a extração e, por fim, o reconhecimento dos caracteres. Para o reconhecimento dos caracteres da placa será utilizada a *OpenALPR* (BUHUS; TIMIS; APATEAN, 2016), que consegue identificar uma possível área de placa. A *OpenALPR* é um biblioteca para reconhecimento automático de placas de código aberto escrita em C++. A biblioteca tem suporte a imagens digitais e fluxos de vídeos para identificação dos caracteres da placa. Como saída do algoritmo, obtém-se a representação textual dos caracteres da placa. Os padrões reconhecidos pelo algoritmo são para placas veiculares da América do Norte, Europa, Austrália, Grã Bretanha, Singapura, Coreia do Sul e Brasil (BUHUS; TIMIS; APATEAN, 2016). Nesse projeto ela será utilizada para reconhecimento da placa a partir de imagens digitais obtidas por um dispositivo móvel. Essa biblioteca utiliza ferramentas da *OpenCV* para identificação dos caracteres na imagem e utiliza a biblioteca *Tesseract* para reconhecimento dos caracteres extraídos da imagem, ambos descritos a seguir.

A biblioteca *OpenCV* tem como objetivo fornecer ferramentas que possam ser usadas para a resolução de problemas que envolvem visão computacional. Possui desde processamento de imagens de baixo nível até funções de processamento de alto nível, tal como detecção de rostos. O desenvolvimento da biblioteca foi iniciado a partir de um projeto de pesquisa da Intel em 1998 e disponibilizada em 2000 sob a licença BSD de código aberto (QUEUE, 2012).

A *Tesseract*, por sua vez, é uma biblioteca de rede neural para reconhecimento de texto. Foi originalmente desenvolvida para o reconhecimento de textos apenas em inglês, porém

a partir da versão 2.0, foi possível reconhecer qualquer texto, desde que estivesse no formato UTF-8. Atualmente, a biblioteca se encontra na versão 4.0, trazendo melhorias de precisão no reconhecimento de textos (TESSERACT, 2017). Com ela, é possível reconhecer quais os caracteres de uma placa após ter sido feita a extração. Para isso, basta realizar o treinamento da biblioteca para reconhecer o padrão de placas brasileiras.

A *OpenALPR* foi escolhida como biblioteca de OCR por integrar todas as ferramentas necessárias para obtenção dos caracteres das placas veiculares. Com ela, não se faz necessário um tratamento das imagens enviadas ao *WebService*, como aplicações de filtros para um melhor resultado, o que traz facilidade para o desenvolvimento da aplicação.

3.2 *Android*

O *Android* é um sistema operacional para dispositivos móveis lançado em 2007, que foi desenvolvido tendo em vista os vários conjuntos de usuários e seus diferentes níveis de capacidade. O sistema consiste em uma pilha de softwares completos, que são baseados na linguagem de programação Java (GANDHEWAR; SHEIKH, 2010).

O *Android* é o sistema operacional para dispositivos móveis mais utilizado no mundo, alcançando 85% dos *smartphones* presentes no mercado internacional, como pode ser visto na Tabela 1. Por esse motivo, esse sistema operacional foi o escolhido para o desenvolvimento da aplicação, que contou com o uso da linguagem nativa para execução dessa atividade.

Tabela 1 – Presença dos sistemas operacionais no mercado internacional de *smartphones*

Período	Android	iOS	Windows Phone	Outros
2016Q1	83,4%	15,4%	0,8%	0,4%
2016Q2	87,6%	11,7%	0,4%	0,3%
2016Q3	86,8%	12,5%	0,3%	0,4%
2016Q4	81,4%	18,2%	0,2%	0,2%
2017Q1	85,0%	14,7%	0,1%	0,1%

Fonte – IDC (2017)

No *Android* é comum a concorrência por dados entre as diferentes funcionalidades de uma aplicação, tendo em vista que as aplicações se utilizam de diferentes eventos do sistema operacional. Com isso, eventos assíncronos são gerados a partir de diversas fontes, que vão desde a interface de usuário até os sensores do dispositivo. Essa assincronia dos eventos pode

fazer com que a aplicação seja corrompida durante a sua utilização, impactando na sua utilização (PAVOL; VESELIN; MARTIN, 2015). Para que isso seja evitado, é desejável que as *threads* da aplicação sejam bem gerenciadas.

3.3 Usabilidade

Para o desenvolvimento da aplicação, serão utilizados artefatos que viabilizem a utilização da aplicação. Como a usabilidade está diretamente relacionada com a interface (NIELSEN, 1990), existem vários aspectos que melhoram a usabilidade de um software. Nielsen (1995) define cinco aspectos para que seja medida e avaliada a usabilidade de um software. Esses aspectos são: *fácil de aprender*, *eficiente para usar*, *fácil de lembrar*, *pouco sujeita a erros* e *agradável de usar*.

Afim de proporcionar o maior nível de satisfação do usuário ao utilizar a aplicação proposta nesse projeto, esses aspectos serão usados para definir o *design* da aplicação, além de ajudarem na definição dos fluxos de interação a serem realizados pelos usuários.

3.4 *WebService*

O conceito de *WebService* surgiu a partir da necessidade de comunicação entre sistemas, não importando a linguagem na qual cada sistema foi desenvolvido, plataforma de hardware ou software. Para o funcionamento do *WebService*, presume-se dois sistemas trocando informações, onde um provê serviços e o outro solicita os serviços. Quando um serviço é solicitado, o provedor busca as informações requeridas em uma base de dados e posteriormente envia as informações ao solicitante (ZAVALIK, 2004).

Base64 é um método de codificação e proteção de dados. Esse método consiste em encontrar todos os caracteres ASCII, converter em números binários e em seguida dividir os números binários em textos, para posterior conversão do texto em valores correspondentes da Base64 (AHMAD; AHMAD; FAKHRI, 2012). Como resultado da codificação, é obtido um texto com que pode conter até 64 tipos de diferentes caracteres. Por exemplo: a palavra "Placa", ao ser codificada se torna "UGxhY2E=". Com ele é possível converter desde textos até arquivos, onde poderá ser feito a decodificação e posterior transformação do texto em arquivo novamente.

Nesse projeto será desenvolvido um *WebService* para recebimento das imagens digitais convertidas em Base64 e retorno de consultas sobre a situação legal do veículo que serão

realizadas pelo dispositivo móvel. Os dados sobre a situação do veículo serão obtidos a partir de uma base de dados abastecida pelos próprios usuários da aplicação e também a partir do Sinesp. A consulta ao Sinesp será realizada a partir de um micro serviço usando uma *API* fornecida por desenvolvedores.

3.5 *Web 2.0*

O conceito da *Web 2.0* surgiu a partir das novas formas de se interagir ao se utilizar aplicações web. Ela trouxe consigo maior repercussão social em sites de notícias, blogs e aplicações do gênero. Usuários passaram a se conectar mais a essas aplicações, dando suas opiniões e até contribuindo para os conteúdos que estavam sendo acessados.

Com essa nova fase da web, os processos de trabalho coletivo, como troca efetiva, circulação e produção de informações foram potencializados (PRIMO, 2006). A partir desse conceito, foi inserida na aplicação a opção de denunciar roubos, furtos e clonagens de veículos.

Todas as denúncias inseridas na aplicação serão persistidas em uma base de dados centralizada. Afim de evitar denúncias falsas, a aplicação conta com dois dados obrigatórios para se poder realizar a denúncia: placa e chassi. Não há uma forma de verificar se a placa inserida é condizente com o chassi inserido, porém, é possível validar o número do chassi. Esse número possui um padrão numérico, onde é verificado se o número inserido possui esse padrão ao se realizar a denúncia.

3.6 Micro serviços

O Princípio da Responsabilidade Única descrito por Robert C. Martin serviu de inspiração para que Newman (2015) propusesse uma arquitetura baseada em micro serviços.

Nessa arquitetura, serviços pequenos trabalham de forma autônoma e cooperativa. Respeitando o princípio citado acima, cada serviço possui uma única responsabilidade, evitando assim o aumento de código fonte, fazendo com que a manutenibilidade dos serviços se torne menos complicada (CHIARADIA; MACEDO; DUTRA, 2017).

Utilizando-se esse conceito, foram criados alguns micro serviços para a aplicação. Nem todos eles são acessados diretamente pela aplicação, um dos micro serviços é acessado por outros micro serviços que requisitam informações sobre o veículo, passando a placa a ser verificada. Esse micro serviço que não é acessado pela aplicação, foi desenvolvido utilizando

NodeJS, pois utilizamos uma *API* dessa linguagem para fazer acesso ao Sinesp. Os outros micro serviços foram desenvolvidos em Java.

É importante salientar que iremos chamar de *Webservice* o serviço responsável por receber as imagens, armazenar, executar o comando da biblioteca para realizar o reconhecimento dos caracteres da imagem, excluir a foto depois do processo de reconhecimento e enviar o resultado ao micro serviço responsável por realizar a busca no Sinesp. Além disso, ele monitora os outros micro serviços afim de descobrir possíveis falhas. Como ele não possui responsabilidade única, não pode ser inserido na categoria de micro serviço.

3.7 *SOAP*

O *SOAP* é um protocolo usado para trocar informações estruturadas. Com esse protocolo, é possível haver comunicação entre serviços distribuídos e descentralizados. Ele se utiliza de *XML* para definir a troca de mensagens (SCHEPKE; SOUZA; VIANA, 2012).

Esse protocolo foi utilizado para se estabelecer uma comunicação com o servidor da Sinesp, que por sua vez realiza uma consulta em sua base de dados centralizada. Essa comunicação foi realizada a partir de um micro serviço NodeJS, como citado anteriormente.

Desenvolvedores que disponibilizaram a *API* necessária para a comunicação, fizeram engenharia reversa no protocolo de comunicação realizado entre a aplicação do Sinesp e sua base de dados. A partir disso, foi obtido um *XML* padronizado contendo os dados esperados pelo servidor. Com isso, foi possível realizar as modificações necessárias no arquivo a ser enviado na requisição para se obter uma resposta válida.

O arquivo é estruturado contendo informações obtidas do próprio dispositivo em que foi feita a requisição. Como a requisição estava sendo realizada de uma máquina virtual ao invés de um dispositivo móvel, foi necessário inserir dados manualmente no arquivo. Esses dados foram definidos a partir da análise do dispositivo utilizado no ambiente de testes. Por fim, foi possível realizar as requisições e obter os resultados.

3.8 Computação em nuvem

Baseada no conceito de recursos sob demanda, onde o usuário paga apenas pelos recursos que ele utilizar, a computação em nuvem surgiu para contornar dificuldades encontradas no desenvolvimento de aplicações que utilizam a *Web*. Ela consegue provê de fácil acesso a

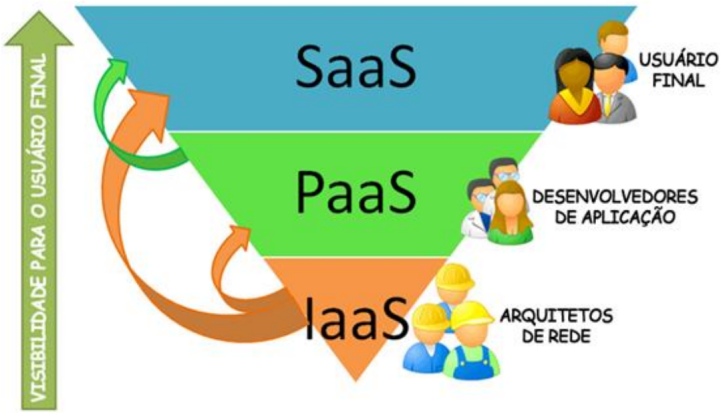
recursos, baixo custo e com garantias de disponibilidade e escalabilidade (SOUSA; MOREIRA; MACHADO, 2009). Com isso, os desenvolvedores deixam de lado a preocupação com a infraestrutura dos equipamentos e conseguem focar exclusivamente nos interesses do seu negócio.

Pensando nisso, a *AWS* foi escolhida para ser a plataforma a ser utilizada para o uso da computação em nuvem. Ela será responsável pelo armazenamento das imagens enviadas pelo dispositivo e por hospedar o *Webservice* e os micro serviços.

3.8.1 Modelos de serviço

Para se obter uma melhor estruturação, a arquitetura da computação em nuvem foi dividida em três tipos de camadas. Cada camada é capaz de provar um tipo diferente de serviço. Essas camadas estão divididas da seguinte forma: *SAAS*, *PAAS* e *IAAS*, como podem ser vistas na Figura 1.

Figura 1 – Representação das três camadas da computação em nuvem



Fonte: Vitor Meriat (2011)

Para o desenvolvimento da aplicação, foram utilizados serviços do tipo *PAAS* e *IAAS*. O *PAAS*, camada central na arquitetura, provê um ambiente completo de desenvolvimento onde o desenvolvedor pode testar as aplicações na nuvem. Nesse contexto, foi utilizado um serviço de armazenamento para o registro das ocorrências denunciadas pela aplicação. O *IAAS*, camada inferior na arquitetura, provê hardware sob demanda, como poder de processamento, infraestrutura de redes, entre outros. Com esse serviço, foi possível obter acesso virtual a esses recursos, onde foi possível criar uma máquina virtual para hospedar o *Webservice* e micro serviços.

4 PROCEDIMENTOS METODOLÓGICOS

O desenvolvimento da aplicação se deu com as seguintes etapas:

a) Análise dos requisitos
b) Definição do fluxo de atividades para obtenção da situação veicular a partir da imagem do veículo;
c) Desenvolvimento do *WebService* e micro serviços;
d) Desenvolvimento da aplicação;
e) Testes e validação da aplicação.

Detalhes sobre cada uma dessas etapas são apresentados ao longo desta seção.

4.1 Análise dos requisitos

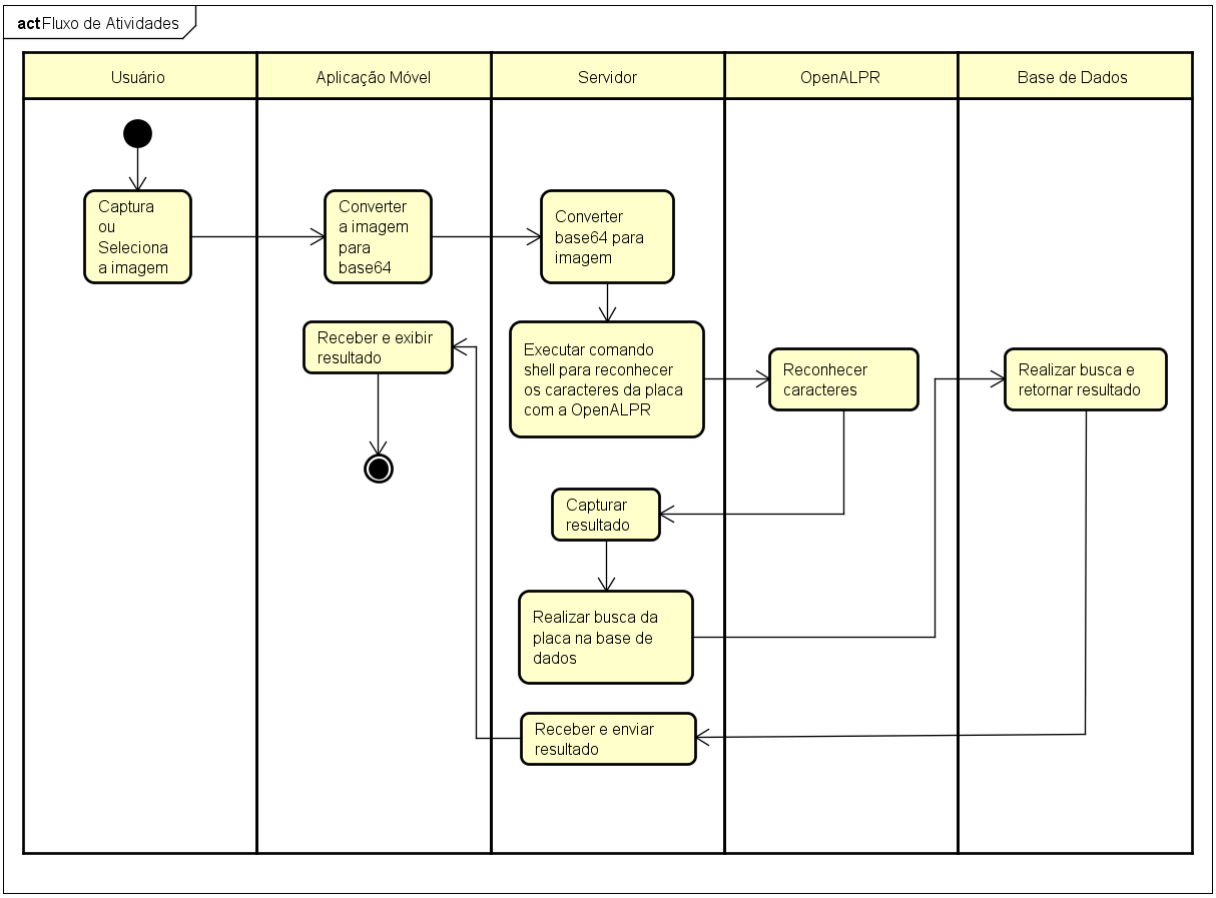
Nessa primeira fase foram levantados os requisitos da aplicação. Os requisitos foram definidos a partir da análise de aplicações que possuem objetivos parecidos com o desse projeto, além de conversas informais com dois especialistas no desenvolvimento de aplicações que auxiliam a consulta de situação veicular. As conversas foram a partir da troca de *e-mails*, nessas conversas foram feitas perguntas sobre o funcionamento para obtenção dessas informações, além de discutir sobre novas funcionalidades. A partir disso foram levantados requisitos que poderão ser vistos na seção que apresentamos a nossa aplicação.

4.2 Definição do fluxo de atividades para o usuário obter a situação de um veículo a partir dos dados enviados pela aplicação

Para uma melhor visualização do fluxo de consulta da situação veicular a partir da imagem do veículo, foi definido o fluxo de atividades para que sejam obtidos os caracteres da placa e posterior exibição do resultado da busca para o usuário. Como a consulta por imagem é uma funcionalidade com uma alta dificuldade de implementação, essa definição do fluxo se deu para uma melhor visualização das atividades a serem feitas para que fosse realizada a consulta. A partir dessa definição, as outras funcionalidades foram implementadas respeitando esse fluxo.

Está descrito o fluxo para obter a situação veicular a partir da foto do veículo na Figura 2.

Figura 2 – Diagrama de atividades para verificação da situação veicular



Fonte: O Autor

4.3 Desenvolvimento do *WebService* e micro serviços

Nesta fase foi desenvolvido o *WebService* e os micro serviços para recebimento dos dados a serem buscados. Como a *OpenALPR*, biblioteca para reconhecimento dos caracteres contidos na imagem funciona por linha de comando, será executado um comando Linux a cada imagem recebida. Após feito o reconhecimento da imagem, serão obtidos os caracteres da placa, que serão utilizados para pesquisa na base de dados.

Caso a busca seja realizada por inserção manual dos caracteres, é feita a requisição ao micro serviço responsável por essa funcionalidade.

4.4 Desenvolvimento do Coplav

Nesta fase, foi desenvolvido o protótipo de aplicação *Android* para captura das imagens digitais e dados para posterior envio ao *WebService* e micro serviços. Foi utilizado o *Android Studio*[1] para desenvolvimento da aplicação e usado o *Git*[2] juntamente com o *Bitbucket*[3] para controle de versão do código. Além disso, foi realizada a definição da alocação dos recursos necessários para a aplicação e foi definida também a arquitetura da solução proposta.

Para a funcionalidade de verificação de situação veicular a partir da imagem do veículo, após realizado o envio da imagem e finalizado os processos realizados pelo *Webservice* e micro serviços, o dispositivo tem como resposta a situação do veículo, caso o reconhecimento tenha sido feito, caso contrário, é informado que não foi possível obter os dados da placa. A única diferença da busca manual para a busca por imagem, é a não necessidade de conexão com o *Webservice* para a consulta.

4.5 Testes e validação da aplicação

Nesta fase foram realizados os testes da aplicação móvel. A metodologia escolhida foi a de testes unitários, que posteriormente foram utilizados para a realização do teste de cobertura para as principais funcionalidades da aplicação móvel. Além disso, foi realizado um *survey* de validação da aplicação desenvolvida.

Foram definidos casos de testes unitários e escolhidas as principais partes do código da aplicação móvel para a realização dos testes. Também foram definidas atividades para a realização do *survey*. Mais detalhes podem ser encontrados no Capítulo 6.

[1] https://developer.android.com/studio/index.html
[2] https://git-scm.com/
[3] https://bitbucket.org/

5 A APLICAÇÃO COPLAV

O Coplav é uma aplicação *Android* para verificação de situação veicular a partir de imagens das placas dos veículos e da inserção manual dos caracteres da placa. Ela foi desenvolvida com o propósito de disponibilizar ao usuário uma forma diferente das existentes para consulta dos dados de um veículo.

Este capítulo contém os requisitos da aplicação, suas telas e funcionalidades.

5.1 Requisitos

Esta seção tem o objetivo de apresentar os requisitos levantados para a aplicação. Foram definidos os seguintes requisitos para a aplicação:

- Requisitos funcionais
  - Verificar situação via inserção manual dos dados: O usuário pode inserir via teclado os caracteres da placa e obter a situação do veículo buscado;
  - Verificar a situação via imagem do veículo: O usuário pode selecionar uma imagem do veículo que contenha uma área da placa para obter a situação do veículo buscado;
  - Recortar imagem a ser buscada: Como a biblioteca possui melhor precisão quando a imagem da área da placa está em destaque, o usuário poderá recortar a imagem original a área que contém a placa do veículo;
  - Denunciar ocorrência: O usuário poderá denunciar uma ocorrência. Ela poderá ser do tipo roubo, furto e clonagem. Ele deverá informar placa e chassi válido;
  - Visualizar mapa de ocorrências: O usuário poderá visualizar um mapa contendo marcadores onde ocorreram as ocorrências denunciadas pela aplicação.
- Requisitos não funcionais de portabilidade
  - O dispositivo requer sistema operacional *Android* com versão igual ou superior a *4.0.3*
- Restrições
  - O dispositivo requer conexão à internet para realizar as buscas, denúncias e atualizações da base de dados interna

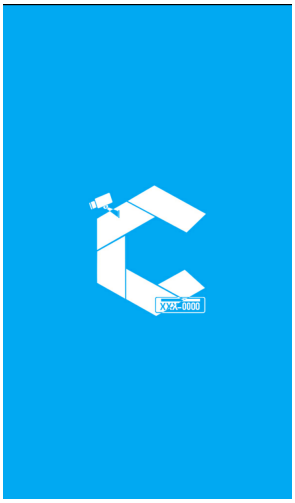
5.2 O desenvolvimento do Coplav

Esta seção tem o objetivo de apresentar as telas da aplicação Coplav e explicar o funcionamento de cada uma.

5.2.1 Tela de início

Na primeira tela ilustrada pela Figura 3, a base de dados das ocorrências interna da aplicação é populada, caso ela ainda esteja vazia. Caso contrário são buscadas atualizações na base de dados centralizada. Essa busca ocorre sendo enviada a data da última atualização registrada na aplicação e são retornados apenas os dados que foram inseridos posteriormente a essa data. Com isso evitamos que consumo de banda seja utilizado sem necessidade.

Figura 3 – Tela de carregamento da aplicação



Fonte: O Autor

5.2.2 Tela de busca manual

Na tela ilustrada pela Figura 4, o usuário insere via teclado os caracteres da placa a ser realizada a busca. Foi inserida uma máscara no campo de entrada dos dados, fazendo com que o padrão de placa brasileiro, que corresponde a três letras seguidas de quatro números, seja respeitado.

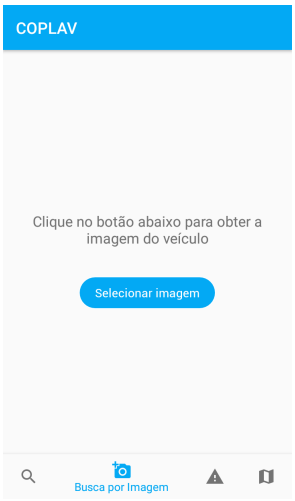
Figura 4 – Tela para buscar manualmente a situação veicular

Fonte: O Autor

5.2.3 Tela de busca por imagem

Na tela ilustrada pela Figura 5, o usuário seleciona a foto de um veículo a ser realizada a busca. A seleção da foto pode ser feita buscando fotos já salvas no aparelho, como também captura via câmera do dispositivo.

Figura 5 – Tela para buscar por imagem a situação veicular

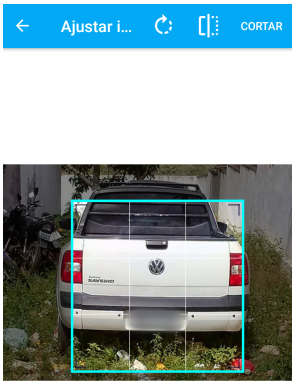


Fonte: O Autor

5.3 Tela de recorte da imagem selecionada

Na tela ilustrada pela Figura 6, o usuário seleciona a área da foto que contém a placa do veículo. Com isso, a precisão de identificação dos caracteres aumenta, visto que durante os testes da aplicação, o não recorte da área da placa acarretava em um significativo número de falhas na identificação dos caracteres. A *OpenALPR* é a responsável por realizar a identificação dos caracteres presentes na placa. Quando a placa não é identificada, uma mensagem é exibida mostrando uma dica para melhor seleção da área da placa.

Figura 6 – Tela para recortar a imagem selecionada para a verificação da situação veicular



Fonte: O Autor

5.4 Tela de denúncia de ocorrência

Na tela ilustrada pela Figura 7, o usuário insere os dados referentes ao veículo e ao tipo da ocorrência para que seja disponibilizada a outros usuários após inserida na base de dados da aplicação. O endereço e desejável para que seja obtidas informações de latitude e longitude da ocorrência para que ela possa ser exibida no mapa.

Figura 7 – Tela para realizar a denúncia de ocorrência

Fonte: O Autor

5.5 Tela do mapa das ocorrências

Na tela ilustrada pela Figura 8, o usuário pode obter os locais onde aconteceram as ocorrências. Ao selecionar um marcador no mapa, obtêm-se informações sobre a ocorrência registrada. Esses dados consistem na placa do veículo, cor, modelo, fabricante, ano, ano modelo, número do chassi com alguns caracteres omitidos e o endereço da ocorrência.

Figura 8 – Tela para visualizar os locais das ocorrências

Fonte: O Autor

6 TESTES E VALIDAÇÃO DA APLICAÇÃO

Nesta seção são apresentados os resultados do *survey* de validação, testes unitários e de cobertura das principais partes do código. Mais detalhes são apresentados no decorrer deste capítulo.

6.1 Testes unitários

Para a realização dos testes unitários, foram utilizados o *JUnit*[1] e o *Mockito*[2]. Ambos são *frameworks* para esse tipo de teste. A escolha deles se deu pelo suporte nativo oferecido pelo *Android Studio*, onde foi desenvolvida a aplicação.

O *JUnit* é utilizado para definir as regras dos testes, como os retornos esperados de cada teste. Já o *Mockito* auxilia na execução desses testes, fornecendo anotações que podem ser usadas para instanciar os objetos que são necessários como parâmetros para as funcionalidades que estão sendo testadas.

6.2 Teste de cobertura

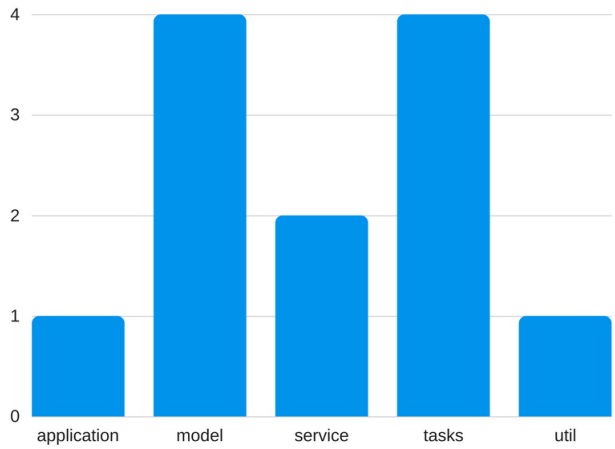
Após a criação dos testes unitários, foi realizado o teste de cobertura. Esse teste, como o próprio nome diz, visa obter o percentual de cobertura do código do projeto.

A classes da aplicação foram divididas em pacotes. Ao todo foram criados sete pacotes onde foram divididas as dezoito classes do projeto. Dois desses pacotes eram exclusivos para as classes que representavam as telas da aplicação. Esses foram desconsiderados para o teste de cobertura.

Na Figura 9, temos como estão divididas as classes utilizadas para esse teste.

[1] http://junit.org/junit4/
[2] http://site.mockito.org/

Figura 9 – Divisão das classes do projeto utilizadas para o teste de cobertura



Fonte: O Autor

Cada barra do gráfico representa um pacote contido no projeto da aplicação. A altura de cada barra representa o número de classes presentes no pacote.

6.3 *Survey* de validação

A fim de se avaliar a experiência do usuário no uso da aplicação desenvolvida, foi realizado um *survey* de validação. Um *survey* é um método de pesquisa quantitativo no qual é possível obter dados ou informações sobre características, ações ou opiniões de determinado grupo de pessoas, indicado como representante de uma população (FREITAS et al., 2000). Mais detalhes sobre o planejamento e execução desse *survey* serão descritos nas próximas subseções.

6.3.1 Preparação

Nesta fase foram definidos os perfis dos usuários participantes do *survey*. Como a aplicação não possui um público alvo muito específico, foram definidos possíveis perfis de pessoas que poderiam vir a utilizar a aplicação proposta. No total foram escolhidos cinco participantes. Por mais que seja um número pequeno, conseguimos diversificar os perfis, tornando a validação mais abrangente. O Quadro 1 contém uma breve descrição de cada um dos participantes. Afim de manter a privacidades dos usuários, eles foram nomeados de P1, P2, P3,

P4 e P5.

O *smartphone* utilizado foi um Samsung Galaxy J5 Prime, com o sistema operacional *Android Marshmallow* na versão 6.0.1. Cada um dos participantes assinou um termo de consentimento presente no Apêndice A. O cenário definido para a realização do teste pode ser conferido no Apêndice B.

Quadro 1 – Perfil dos participantes do *survey* de validação.

Participante	Descrição
P1	O participante P1 é do sexo masculino, 24 anos, é mecânico em uma oficina automotiva que fazia uso de aplicações para consulta de situação veicular como o Sinesp.
P2	O participante P2 é do sexo masculino, 22 anos, é policial militar com experiência em uso de dispositivos móveis.
P3	O participante P3 é do sexo masculino, 28 anos, é servidor técnico-administrativo da UFC – Campus Quixadá que pretendia trocar de carro.
P4	O participante P4 é do sexo feminino, 27 anos, é dentista e planejava comprar seu primeiro carro.
P5	O participante P5 é do sexo masculino, 36 anos, é gerente de uma concessionária automotiva que fazia uso de aplicações para consulta de situação veicular como o Sinesp.

Fonte – O Autor

6.3.2 Coleta de dados

O *survey* foi realizado nas casas de cada um dos participantes entre os meses de novembro e dezembro de 2017, mas precisamente nos fins de semanas. No primeiro dia de testes foi realizado o teste piloto, onde foi encontrado um erro no micro serviço que fazia a busca na base de dados do Sinesp, o qual não havia sido detectado no ambiente de testes. Com o erro devidamente corrigido, foi dado início aos outros dias de testes.

Para a coleta dos dados resultantes da execução dos testes, foram feitas as atividades descritas a seguir:

- Os participantes realizaram as tarefas propostas no cenário de teste que foi apresentado anteriormente;
- Foi aplicado um questionário ao final dos testes. Nesse questionário presente no Apêndice C, haviam questões a respeito de aspectos da aplicação e outras para livre manifestação da opinião dos participantes, onde poderiam sugerir melhorias para o sistema.

7 RESULTADO DOS TESTES E DA VALIDAÇÃO DA APLICAÇÃO

7.1 Testes unitários

Ao todo foram criados 35 casos de testes, sendo trinta dedicados aos testes a nível de programação e os outros cinco foram testes automatizados usando a interface do dispositivo em que a aplicação estava instalada.

Os testes serviram para a validação das regras de negócios da aplicação. Boa parte desses testes consistiram na verificação de integridade dos dados inseridos pelo usuário, tais como:

- Inserção da placa para a realização da busca manual;
- Inserção da placa e chassi na realização de uma denúncia;
- Inserção do endereço da ocorrência;
- Seleção de uma imagem válida na busca por imagem.

Salientamos que para os cenários acima, foram criados diversos casos de testes. Outros casos de testes criados consistiram na verificação de partes do código que consideramos importantes para a aplicação, tais como:

- Instanciação de *threads* e classes responsáveis pela conexão com o servidor;
- Adição de máscaras em entradas de texto
- Persistência dos dados recebidos pelo servidor;
- Busca de atualizações para a base de dados interna;
- Instanciação da classe responsável por iniciar a aplicação.

Em todos os casos criados, houve sucesso na realização dos testes. Com isso foi possível realizar o teste de cobertura, pois se um teste falhasse, o teste de cobertura seria interrompido.

7.2 Teste de cobertura

As principais funcionalidades da aplicação dependiam das classes do pacote *service*. As classes contidas nele eram as responsáveis por realizar a conexão remota com o servidor a partir da criação de novas *threads*. No *Android* não é possível realizar uma conexão a partir da *thread* principal. Além disso, como mencionado no Seção 3.2, o mal funcionamento das *threads* pode fazer com que a aplicação trave. Pensando nisso, foi realizado o teste de cobertura desse

pacote.

No pacote *service*, onde foi realizado o teste de cobertura, a taxa cobertura das classes foi de 100%, dos métodos foi de 100% e das linhas foi de 97%. Foi obtido um alto índice de cobertura no pacote responsável pelo bom funcionamento da aplicação.

7.3 Interpretação dos resultados do *survey*

Os critérios analisados durante o *survey* estão listados a seguir:

- Facilidade de utilização
- Design das telas
- Nomenclatura das telas
- Assimilação das informações
- Nível de satisfação no uso da aplicação

Para a avaliação dos critérios foi dada uma nota que variava de um a cinco. Onde a nota um seria a pior avaliação possível e consequentemente a nota cinco seria a melhor. Após aplicados todos os testes e questionários, foram contabilizadas as notas dadas pelos participantes. Com base nos dados, foram gerados gráficos que podem ser vistos nas próximas subseções desse capítulo.

7.4 Relato dos resultados do *survey*

Para uma melhor estruturação do relato dos resultados, foi seguida a mesma ordem dos critérios avaliados. O primeiro critério foi o de facilidade de utilização. Os resultados são apresentados na Tabela 2

Tabela 2 – Resultado do questionário aplicado no *survey*

Critério	Nota 1	Nota 2	Nota 3	Nota 4	Nota 5
Facilidade de utilização	1 participante		3 participantes	1 participante	
Design das telas			2 participantes	1 participante	2 participantes
Nomenclatura das telas		1 participante	1 participante	2 participantes	1 participante
Assimilação das informações		1 participante	2 participantes	2 participantes	
Nível de satisfação		1 participante	2 participantes	2 participantes	

Fonte – O autor

No critério de facilidade de utilização, três participantes consideraram normal a facilidade de utilização, um considerou fácil e um considerou muito difícil. No critério de

design das telas, dois participantes consideraram normal o *design*, um participante considerou bom e os outros dois consideraram muito bom. No critério de nomenclatura das telas, um participante achou normal a nomenclatura, dois acharam clara, um achou muito clara e o último achou confusa. No critério de assimilação das informações, dois participantes acharam normal a assimilação, dois acharam fácil e um considerou difícil. No último critério, o de nível de satisfação, dois participantes consideraram normal a satisfação, dois consideraram alto e um considerou baixo.

Como foi apresentado na Tabela 2, a Coplav obteve um bom desempenho em todos os critérios avaliados no *survey* de validação.

O questionário aplicado após a realização do teste de usabilidade continha questões onde o usuário poderia sugerir melhorias para a aplicação conforme foi explicado na subseção 6.3.2. Abaixo estão algumas sugestões consideradas importantes para a aplicação:

- Realizar a busca manual sem a necessidade de pressionar o botão *Buscar*, podendo ser feita a partir do botão *Enter* do teclado virtual do dispositivo;
- Exibir como ficou a foto após realizado do recorte;
- Diminuir a quantidade de campos para se realizar uma denúncia;
- Exibir um tutorial quando a aplicação for aberta pela primeira vez.

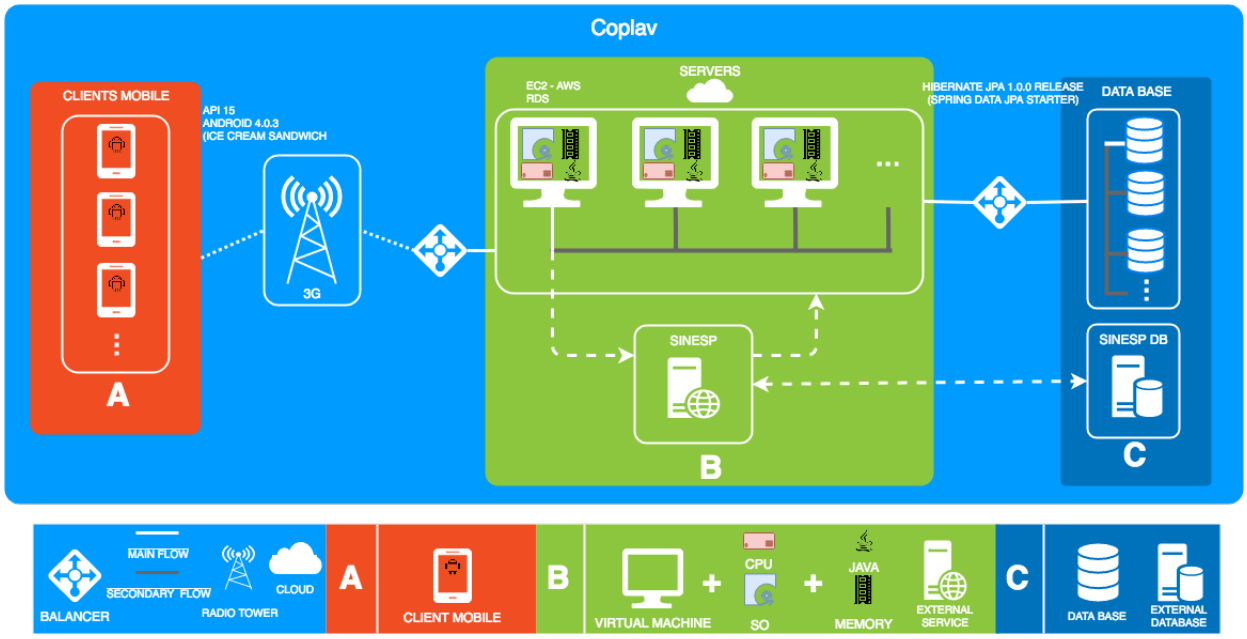
8 RESULTADOS

Neste capítulo são apresentados os resultados do desenvolvimento do Coplav. Os detalhes serão apresentados nas próximas seções.

8.1 Alocação de recursos e visão arquitetural da proposta

Na Figura 10 é possível observar como ficou a alocação dos recursos necessários para o funcionamento da aplicação desenvolvida e como funciona a sua arquitetura.

Figura 10 – Alocação de recursos e visão arquitetural



Fonte: O Autor

A arquitetura contém três tipos de recursos básicos, clientes da aplicação móvel, máquinas virtuais que funcionam como servidores alocadas na nuvem da *AWS* e dois tipos de bases de dados, uma desenvolvida para a aplicação e abastecida pelos próprios usuários e uma outra base de dados do Sinesp que é administrada pelo próprio órgão, não sendo possível saber qual o seu tipo e como funciona a sua comunicação.

Os clientes utilizam a rede para se comunicar com o servidor na nuvem. Essa comunicação passa primeiro por um balanceador de carga que visa redirecionar o cliente a máquina que possui mais disponibilidade. A priori, apenas uma máquina funciona, mas pode haver a criação de novas máquinas utilizando o conceito de escalabilidade da computação em

nuvem. Para que haja a criação de novas máquinas, é necessário definir critérios. O critério utilizado foi a quantidade de memória *ram* que estava sendo utilizada pela máquina. Se o consumo de memória chegasse a 80% do total, seria criada uma nova máquina.

Para a comunicação com a base de dados da aplicação, também existe um balanceador de carga e podem existir múltiplas instâncias da base de dados. Como foi utilizada uma plataforma como serviço da *AWS*, todas as ações necessárias de balanceamento de carga e integridade dos dados fica por conta da própria plataforma.

Além do *Webservice* e micro serviços desenvolvidos para o servidor, há também a presença do servidor do Sinesp. A comunicação com esse servidor é feita utilizando o protocolo *SOAP* a partir de um micro serviço. O micro serviço realiza uma requisição a esse servidor enviando o *XML* empacotando as informações necessárias. Esse servidor então recebe esse arquivo, extrai as informações, valida e posteriormente realizar uma busca na base de dados própria que é abastecida pelo DENATRAN. Após essa consulta é dada uma resposta ao micro serviço, que então realiza os procedimentos necessários para enviar fornecer uma resposta ao cliente.

A solução proposta se mostrou bastante eficaz, não sendo detectado nenhum gargalo no seu funcionamento.

9 DISCUSSÃO

Foi desenvolvida uma solução arquitetural e um protótipo de aplicação móvel afim de aumentar o índice de recuperação de veículos. Tanto a solução como o protótipo se mostraram eficazes para que sejam amplamente utilizados. Os testes desenvolvidos foram bem sucedidos e o *survey* de validação obteve um bom resultado. O bom resultado da validação se deu a partir da utilização dos conceitos de usabilidade apresentados na Seção 3.3.

Após uma análise de técnicas para reconhecimento de caracteres a *OpenALPR* foi utiliza para identificar os caracteres das placas. Foram explorados os conceitos de computação em nuvem, onde foi alocado os recursos necessários para o funcionamento da aplicação.A partir do conceito da *Web 2.0* foi criada uma base de dados para ser colaborativamente abastecida pelos usuários. A solução contou com o desenvolvimento de *Webservice* e micro serviços, onde o protocolo *SOAP* foi utilizado para realizar a comunicação entre serviços distribuídos.

10 CONSIDERAÇÕES FINAIS

Este trabalho de conclusão de curso apresentou uma proposta de aplicação móvel que auxilia a identificação de veículos com queixa de roubo e furto, podendo ser usada como uma ferramenta para aumento do índice de recuperação desses veículos.

Foi desenvolvido um protótipo completamente funcional para o sistema operacional *Android*. A aplicação contou com as funcionalidades presentes nos requisitos funcionais apresentados na Seção 5.1 e demonstrou eficácia na realização de suas atribuições. Serão feitas melhorias e adição de novas funcionalidades para que então seja disponibilizada na loja de aplicativos do *Android*.

Após desenvolvida o proposta de aplicação móvel, foram realizados testes unitários, de cobertura e feita uma validação a partir de um *survey*, onde ao final foi obtida uma aplicação funcional e que apresentou os métodos de verificação já presentes nas aplicações atuais, com a adição da busca por imagem, opção de denúncia e mapa com as ocorrências presentes na base de dados centralizada.

O trabalho de Sales (2010), o (SNSP, 2015) e a discussão realizada com os desenvolvedores de aplicações que utilizavam a *OpenALPR* e que possuíam funcionalidades de consulta veicular contribuíram bastante para o desenvolvimento da aplicação proposta. A partir deles foi possível escolher a biblioteca mais adequada para a identificação dos caracteres da placa e definir as funcionalidades da aplicação, obtendo-se assim a aplicação proposta.

Um ponto negativo da aplicação é a obrigatoriedade na conexão de internet para realizar as buscas. Por outro lado, a funcionalidade de mapa com os locais de roubos ainda pode ser acessada mesmo sem conexão, uma vez que os dados são acessados a partir da base de dados presente no próprio dispositivo.

Como trabalho futuro, é indicada a inserção da consulta por vídeo, para que seja possível realizar uma transmissão em tempo real e auto-detecção de placas veiculares a serem pesquisadas, pois a *OpenALPR* possui suporte a essa funcionalidade, para assim se tornar uma opção de destaque e que possa ser usada amplamente também por autoridades policiais. Além disso, pode ser introduzida a funcionalidade de avisar ao dono do veículo roubado uma localização onde o veículo foi visto. Esse aviso deverá ser feito por *e-mail*, sendo enviado uma notificação ao contato disponibilizado no momento da realização da denúncia. Com isso, os avisos ficam restritos as denúncias realizadas dentro da própria aplicação.

REFERÊNCIAS

AHMAD, M.; AHMAD, H.; FAKHRI, I. Protection of the texts using base64 and md5. **Journal of Advanced Computer Science and Technology**, p. 22–34, 2012. Disponível em: <http://s3.amazonaws.com/academia.edu.documents/31542035/212-1036-1-PB-m.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1495685581&Signature=dsRmil54s2gC2iyyTus60OdCvVY%3D&response-content-disposition=inline%3B%20filename%3D212_1036_1_PB_m.pdf>. Acesso em: 14 mai. 2017.

BRASIL, T. **Placas de carro agora vem com QR Code e tornam mais difícil clonagem de veículos**. [S.l.], 2017. Disponível em: <http://tvbrasil.ebc.com.br/reporter-df/episodio/placas-de-carro-agora-vem-com-qr-code-e-tornam-mais-dificil-clonagem-de>. Acesso em: 17 abr. 2017.

BUHUS, E. R.; TIMIS, D.; APATEAN, A. Automatic parking access using openalpr on raspberry pi3. **ACTA TECHNICA NAPOCENSIS**, v. 57, n. 3, p. 10–15, 2016. Disponível em: <http://users.utcluj.ro/~atn/papers/ATN_3_2016_2.pdf>. Acesso em: 04 jul. 2017.

CHIARADIA, L. F.; MACEDO, D. D. J.; DUTRA, M. L. Proposta de arquitetura de microsserviÇos para um sistema de crm social. **I Workshop de Informação, Dados e Tecnologia, UFSC**, 2017. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/180285/ST1.6.pdf?sequence=1&isAllowed=y>. Acesso em: 30 nov. 2017.

FBSP, F. B. de S. P. **Anuário Brasileiro de Segurança Pública 2016**. [S.l.], 2016. Disponível em: <http://www.forumseguranca.org.br/storage/10_anuario_site_18-11-2016-retificado.pdf>. Acesso em: 13 abr. 2017.

FREITAS, H.; OLIVEIRA, M.; ZANELA, A.; MASCAROLA, J. O método de pesquisa *survey*. **Revista de Administração**, v. 35, n. 3, p. 105–112, 2000.

GANDHEWAR, N.; SHEIKH, R. Google android: An emerging software platform for mobile devices. **International Journal on Computer Science and Engineering (IJCSE)**, p. 12–17, 2010. Disponível em: <http://www.enggjournals.com/ijcse/doc/003-IJCSESP24.pdf>. Acesso em: 19 mai. 2017.

GOLL, G. C. **Protótipo para informatização de controle de estacionamento em área azul no município de Blumenau-SC**. [S.l.], 2013. Disponível em: <http://dsc.inf.furb.br/arquivos/tccs/monografias/2013_2_guilherme-c-goll_monografia.pdf>. Acesso em: 17 abr. 2017.

HAMANN, R. **Android passa Windows e é o SO mais usado no mundo; pelo menos na internet**. [S.l.], 2017. Disponível em: <https://www.tecmundo.com.br/android/115494-android-passa-windows-so-usado-mundo-internet.htm>. Acesso em: 15 abr. 2017.

NIELSEN, J. Evaluating hypertext usability. In: JONASSEN, D. H.; MANDL, H. (Ed.). **Designing Hypermedia for Learning**. 67. ed. Springer Berlin Heidelberg, 1990. p. 147–168. Disponível em: <https://link.springer.com/chapter/10.1007/978-3-642-75945-1_9>. Acesso em: 12.06.2017.

PAVOL, B.; VESELIN, R.; MARTIN, V. Scalable race detection for android applications. **International Conference on Object-Oriented Programming, Systems, Languages, and Applications**, p. 332–348, 2015. Disponível em: <http://www.srl.inf.ethz.ch/papers/oopsla15-eventracer-android.pdf>. Acesso em: 21 mai. 2017.

PRIMO, A. O aspecto relacional das interações na web 2.0. **XXIX Congresso Brasileiro de Ciências da Comunicação**, 2006. Disponível em: <http://www.lume.ufrgs.br/bitstream/handle/10183/1264/000548498.pdf?sequence=1>. Acesso em: 28 nov. 2017.

QUEUE, A. Real-time computer vision with opencv. **Communications of The ACM**, v. 55, n. 6, p. 61–69, jun. 2012.

SALES, R. B. **Localização e validação de regiões candidatas de placas a partir da análise de imagens digitais**. Dissertação (Mestrado) — Universidade Estadual do Ceará, Fortaleza, Dezembro 2010.

SCHEPKE, C.; SOUZA, S.; VIANA, V. **Avaliação de Desempenho de SOAP sobre HTTP, SMTP e BEEP**. 2012. Disponível em: <https://s3.amazonaws.com/academia.edu.documents/34435360/errc07.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1512515611&Signature=6m5AfIMnkjmU7mmi6UTo1l4JR4g%3D&response-content-disposition=inline%3B%20filename%3DAvaliacao_de_Desempenho_de_SOAP_sobre_HT.pdf>. Acesso em: 1 dez. 2017.

SNSP, S. N. D. S. P. **Sinesp Cidadão**. p. 143–157, 2015. Disponível em: <http://repositorio.enap.gov.br/bitstream/handle/1/2724/Sinesp%20Cidad%C3%A3o.pdf?sequence=1&isAllowed=y>. Acesso em: 04 mai. 2017.

SOUSA, F. R. C.; MOREIRA, L. O.; MACHADO, J. C. Computação em nuvem: Conceitos, tecnologias, aplicações e desafios. **RCEMAPI 2009**, 2009. Disponível em: <https://www.researchgate.net/profile/Javam_Machado/publication/237644729_Computacao_em_Nuvem_Conceitos_Tecnologias_Aplicacoes_e_Desafios/links/56044f4308aea25fce3121f3.pdf>. Acesso em: 5 dez. 2017.

TESSERACT. **TrainingTesseract 4.00**. [S.l.], 2017. Disponível em: <https://github.com/tesseract-ocr/tesseract/wiki/TrainingTesseract-4.00>. Acesso em: 12 mai. 2017.

ZAVALIK, C. **Integração de Sistemas de Informação através de Web Services**. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Sul, Porto Alegre, Novembro 2004.

APÊNDICE A – TERMO DE CONSENTIMENTO

Termo de consentimento

Este termo de consentimento é referente ao trabalho de conclusão de curso do aluno Marcelo Henrique Pereira, com o título: "Coplav: Aplicação móvel para consulta de situação veicular em base de dados centralizada". Os dados coletados durante os testes serão estritamente utilizados para análise e melhorias na aplicação.

Posteriormente, os resultados dos testes serão divulgados no trabalho de conclusão de curso do aluno citado anteriormente, aonde o mesmo garante preservar o anonimato dos participantes.

- ( ) Aceito participar dos testes.
- ( ) Não aceito participar dos testes.

Assinatura do participante

Assinatura do aplicador

APÊNDICE B – CENÁRIO DO TESTE

Cenário: Você no papel de usuário começará a realização dos testes.

- Tarefa 1: Você realiza uma consulta inserindo manualmente os caracteres da placa
- Tarefa 2: Você realiza uma consulta selecionando uma imagem de um veículo
- Tarefa 3: Você realiza uma denúncia de ocorrência de um veículo
- Tarefa 4: Você visualiza informações de uma ocorrência selecionando um marcador no mapa

APÊNDICE C – INSTRUMENTO DE COLETA DE DADOS

Questionário

O questionário a seguir funcionará como instrumento de coleta de dados para o trabalho de conclusão de curso do aluno Marcelo Henrique Pereira, estudante da Universidade Federal do Ceará – Campus Quixadá. O título do trabalho é: "Coplav: Aplicação móvel para consulta de situação veicular em base de dados centralizada".

1. Marcar o número correspondente ao grau que você mais concorda.

1.	Facilidade de utilização	Muito difícil 1	Difícil 2	Normal 3	Fácil 4	Muito fácil 5
2.	Design das telas	Muito ruim 1	Ruim 2	Normal 3	Bom 4	Muito bom 5
3.	Nomenclatura das telas (Nomes de campos, botões e etc.)	Muito confusa 1	Confusa 2	Normal 3	Clara 4	Muito clara 5
4.	Assimilação das informações	Muito difícil 1	Difícil 2	Normal 3	Fácil 4	Muito fácil 5
5.	Nível de satisfação no uso da aplicação	Muito baixo 1	Baixo 2	Normal 3	Alto 4	Muito alto 5

2. Aponte situações em que você sentiu dificuldades ao utilizar a aplicação (Opcional).

_____

_____

_____

_____

3. O espaço abaixo é reservado para que você exponha sua opinião e sugira melhorias na aplicação (Opcional).

_____

_____

_____

_____

_____