GoByBus: aplicativo para visualização dos custos de viagens por transporte público

Andressa Santos Araújo[1], Daniela Marques[1], Gustavo Bartz Guedes[1]

[1]Instituto Federal de São Paulo (IFSP)
Avenida Thereza Ana Cecon Breda, s/n - Vila São Pedro Hortolândia-SP - Brasil - Cep 13183-25

andressasantosaraujo@gmail.com,{marquesdaniela,gubartz}@ifsp.edu.br

Abstract. *Currently in the metropolitan region of Campinas different bus lines companies operate, municipal and intermunicipal. Travel cost information is published to the population in different ways. This paper presents the analysis of technical feasibility and the development of a web application that proposes the centralization of this information. In this way, the application can provide greater inputs and travel comparisons possibilities to the user in order to choose the trip that best suits his needs.*

Resumo. *Atualmente na região metropolitana de Campinas operam diferentes linhas de empresas de ônibus, municipais e intermunicipais. As informações referentes ao custo das viagens são divulgadas à população de diversas maneiras. Este trabalho apresenta a análise e o desenvolvimento de uma aplicação web que propõe a centralização dessas informações. Desta maneira, a aplicação poderá proporcionar maiores insumos e possibilidades de comparações de viagens para que o usuário possa escolher qual delas melhor atende suas necessidades.*

1. Introdução

De acordo com as dimensões do desenvolvimento sustentável, pode-se considerar que a mobilidade dentro da visão da sustentabilidade pode ser alcançada sob dois enfoques: um relacionado com a adequação da oferta de transporte ao contexto socioeconômico e outro relacionado com a qualidade ambiental. No primeiro se enquadram medidas que associam o transporte ao desenvolvimento urbano e à equidade social em relação aos deslocamentos e no segundo se enquadram a tecnologia e o modo de transporte a ser utilizado [Campos 2014].

Sistemas Avançados de Transporte Público (APTS) representam o uso de tecnologias avançadas para melhorar a segurança, eficiência e efetividade dos sistemas de transporte público. Os benefícios para os usuários incluem a minimização dos tempos de espera, segurança e facilidade para o pagamento da tarifa, bem como informações precisas e atualizadas sobre itinerários e horários [Silva 2000].

Campos (2014) e Silva (2000) observam a relevância do transporte público para o desenvolvimento socioeconômico da população, justificando a utilização de tecnologias como alternativa para facilitar o acesso e utilização do transporte pela população.

Esse estudo desenvolve uma aplicação, chamada GoByBus, com o intuito de oferecer a população informações referente as viagens de transporte público disponíveis, como insumo para comparação, análise e escolha da opção de viagem que melhor atenda às necessidades individuais do usuário. A aplicação sugerida contém como característica adicional as alternativas já existentes a consulta as viagens realizadas com informações do valor gasto.

Para atingir o objetivo proposto, este artigo está organizado da seguinte maneira: a Seção 2 apresenta os trabalhos correlatos, mostrando as maneiras atuais de visualização dos custos e opções de viagens, a Seção 3 apresenta uma pesquisa de opinião aplicada a população

usuária do transporte público, a Seção 4 contextualiza o leitor com o referencial teórico, a Seção 5 identifica os materiais utilizados para desenvolvimento do projeto, a Seção 6 descreve o desenvolvimento do projeto, a Seção 7 conclui o artigo e ao final são indicadas as referências bibliográficas utilizadas.

2. Trabalhos correlatos

As empresas de ônibus que compõem o transporte público disponibilizam as informações referente a itinerários e horários de diferentes maneiras. No caso da Empresa Metropolitana de Transportes Urbanos de São Paulo (EMTU) as informações referentes às linhas de ônibus intermunicipais são disponibilizadas em seu próprio aplicativo (Figura 1). Os usuários do Sistema de Transporte Metropolitano podem consultar, no aplicativo EMTU/SP (Android e IOS), o itinerário e a localização dos ônibus intermunicipais (regiões metropolitanas de São Paulo, Campinas, Vale do Paraíba/Litoral Norte, Baixada Santista e Sorocaba). As principais funcionalidades do sistema são:

- Linhas: permite conferir as linhas gerenciadas pela EMTU/SP e sua operação em tempo real.
- Endereço: permite conferir as linhas que estão próximas a você ou senão informando um endereço.
- Sua viagem: permite simular o caminho de sua origem para o seu destino com tempo previsto, estimativa de custo e itinerário. [EMTU 2019].

Figura 1. Imagem do aplicativo EMTU

Fonte: [EMTU 2019]

O aplicativo *Transurc Smart* tem como objetivo oferecer uma alternativa segura e prática de pagamento das tarifas dos ônibus que atendem a cidade de Campinas. O aplicativo permite realizar a compra de viagens, o pagamento das tarifas e o acompanhamento do saldo (Figura 2). Para sua utilização é necessário baixar o aplicativo, disponível na plataforma Android e, em seguida, efetuar a compra de créditos com cartão. Para utilização do crédito é preciso passar o QR Code na tela do celular em um dos leitores instalados dentro dos ônibus para liberação da catraca [Transurc 2019].

Figura 2. Imagem do aplicativo Transurc Smart

Fonte: [Transurc 2019]

O *Google Maps* é outro sistema que mostra aos usuários o tempo necessário para se concluir um trajeto. Para sua utilização é preciso abrir o *Google Maps App* e informar os endereços de origem e destino. Depois de selecionado a rota, é preciso selecionar a opção "Transporte Público" para que seja apresentada uma listagem de opções de trajetos com número da linha, tempo estimado para embarque, duração da viagem e valor (Figura 3). Você pode ver partidas de transporte público no app do O *Google Maps* permite a visualização de partidas de transporte públicos em tempo real ou por meio de um cronograma de partidas [Google 2019].

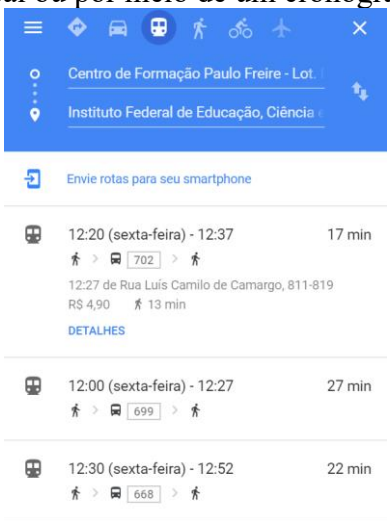

Figura 3. Imagem do aplicativo Google Maps

Fonte: [Google 2019]

O Moovit apresenta opções de rota utilizando os serviços de metrô, ônibus e trem por meio da inserção de um ponto de origem e destino. Disponível em Campinas e região, além de várias cidades do país e do exterior, trata-se de um aplicativo colaborativo que utiliza as informações reportadas pelos próprios usuários [Moovit 2019]. Pode ser utilizado nas

plataformas *mobile* e *desktop*. A Figura 4 mostra imagens da aplicação Moovit, nota-se que informações correspondentes a itinerário, linha, tempo, tamanho do percurso e transferências entre linhas são exibidas.

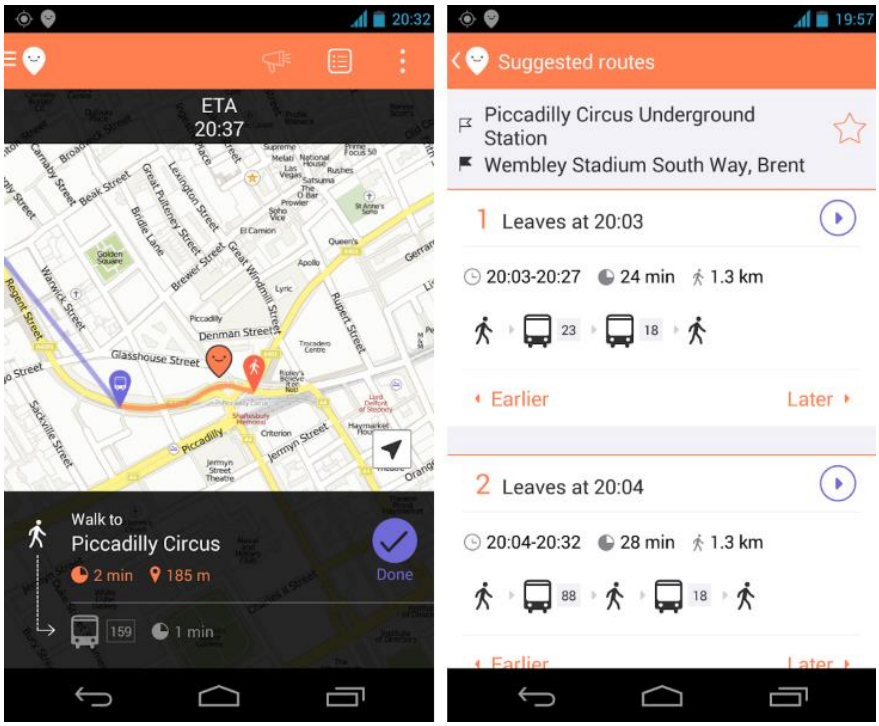

Figura 4. Imagem da aplicação Moovit

Fonte: [Moovit 2019]

3. Pesquisa

Realizou-se uma pesquisa de campo composta por 7 questões alternativas (Tabela 1) com objetivo de mensurar a aceitação do público aos meios de pagamento da tarifa do transporte público atual. A amostra da pesquisa estatística selecionada foram 50 usuários aleatórios do transporte público de Hortolândia com idade variada entre 18 e 60 anos.

Tabela 1. Questionário utilizado na Pesquisa

Perguntas
1. Utiliza o cartão de tarifação do transporte público em Hortolândia?
2. Qual a principal desvantagem em utilizar o cartão de tarifação do transporte público em Hortolândia?
3. Qual a principal desvantagem em pagar a tarifa do ônibus com dinheiro?
4. Você utiliza smartphone?
5. Acredita que o processo de tarifação seria melhor se realizado via um aplicativo para smartphone?
6. Qual a principal desvantagem em pagar a tarifa do transporte público via smartphone?
7. Qual sua idade?

As entrevistas ocorreram nas diferentes plataformas do Terminal Metropolitano de Ônibus de Hortolândia situado em Av. Santana, s/n Parque Hortolândia - Hortolândia entre os dias 10 de abril de 2017 ao dia 26 de abril de 2017 no intervalo das 07:00 às 08:00. A pesquisa foi anônima, com tempo médio de aplicação correspondente a 5 minutos com orientação geral inicial do aplicador mas sem acompanhamento.

Os dados coletados foram tabulados levando em consideração a quantidade total da amostra. A seguir são mostrados os gráficos com ilustração das porcentagens de respostas coletadas acompanhados de sua análise descritiva.

As questões 4 e 7 compõem o perfil dos respondentes, desta forma, essas perguntas são descritas primeiramente. Na Figura 5 é possível observar que 42% dos usuários do transporte público da amostra coletada estão na faixa etária entre 15 e 30 anos e apenas 8% dos entrevistados tinham mais de 60 anos.

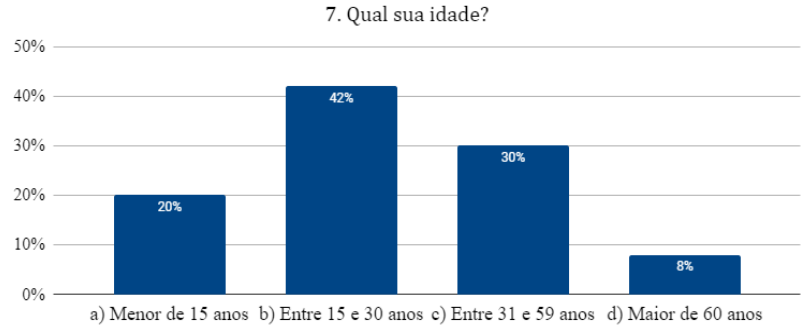

Figura 5. Gráfico de barras ilustrando as respostas da sétima pergunta do questionário

A Figura 6 mostra que o *smartphone* é utilizado por 88% das pessoas entrevistadas. Esse resultado pode estar associado ao fato do maior público respondente ter entre 15 e 30 anos de idade.

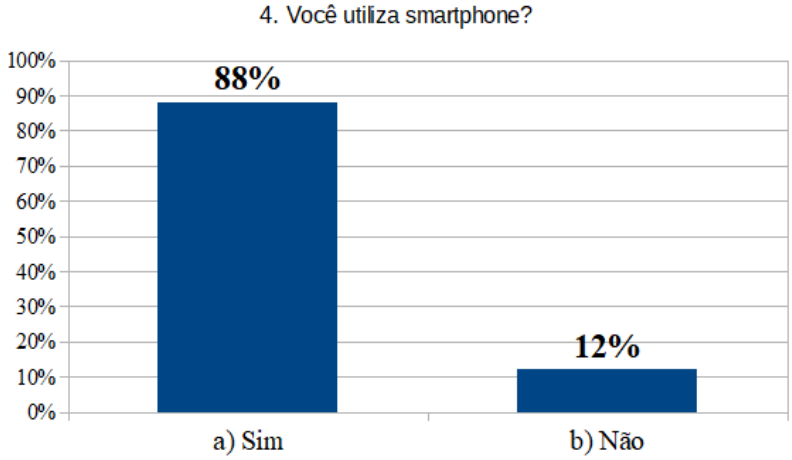

Figura 6. Gráfico de barras ilustrando as respostas da quarta pergunta do questionário

Na Figura 7 nota-se que 40% dos usuários não utilizam o cartão de tarifação e apenas 10% alegaram que sempre fazem a utilização do cartão para tarifação do transporte público em Hortolândia.

Considerando os dados coletados, representados na Figura 8, 36% do total de entrevistados responderam que a principal desvantagem em utilizar o cartão de tarifação do transporte público em Hortolândia é o difícil acesso aos pontos de cadastro. Foi reportado ainda por 26% da amostra o difícil acesso aos pontos de recarga e indisponibilidade de equipamentos de leitura, como grande desvantagem.

Nota-se na Figura 9 que 40% do total da amostra entrevistada apontou a insegurança como principal desvantagem com o pagamento da tarifa em dinheiro.

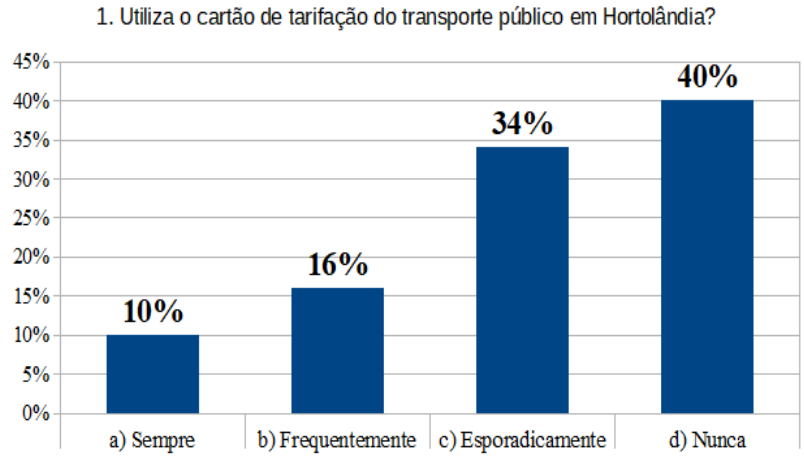

Figura 7. Gráfico de barras ilustrando as respostas da primeira pergunta do questionário

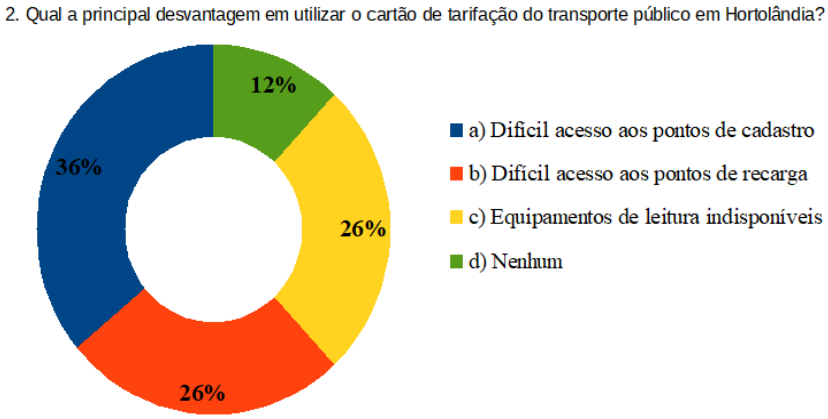

Figura 8. Gráfico de pizza ilustrando as respostas da segunda pergunta do questionário

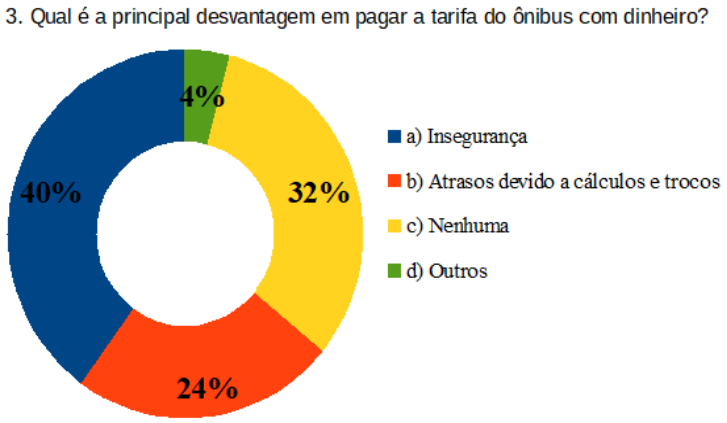

Figura 9. Gráfico de pizza ilustrando as respostas da terceira pergunta do questionário

Na Figura 10 é possível observar que 68% da amostra total entrevistada acredita ser viável utilizar o *smartphone* como alternativa auxiliando o processo de tarifação.

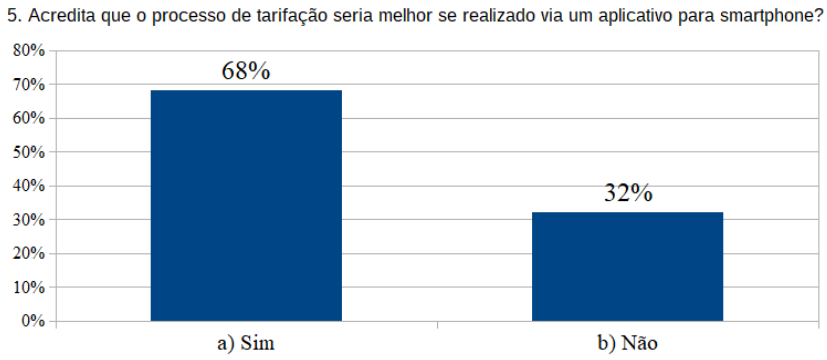

Figura 10. Gráfico de barras ilustrando as respostas da quinta pergunta do questionário

Conforme ilustrado graficamente na Figura 11, do total de entrevistados, 36% entendem que a praticidade é a maior vantagem do uso do *smartphone* como alternativa para auxílio no processo de tarifação.

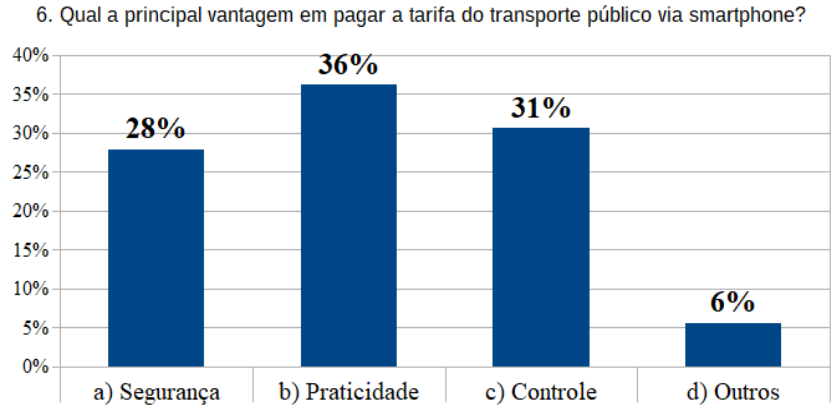

Figura 11. Gráfico de barras ilustrando as respostas da sexta pergunta do questionário

4. Referencial Teórico

4.1. Arquitetura Hexagonal

A arquitetura hexagonal também conhecida como portas e adaptadores, onde uma aplicação como um todo parece um hexágono, com o domínio comercial localizado no interior. Enquanto o domínio de negócios só se preocupa com seu próprio domínio e conceitos, usa portas para se comunicar com o exterior. As portas são as interfaces com o mundo exterior através de uma API que estabelece o que é necessário e como deve se fornecer. Isso dá muita flexibilidade e não apenas permite a troca de adaptadores, por exemplo, durante um teste ou no caso de se tentar diferentes tecnologias rapidamente para encontrar a melhor, pode-se testar com a aplicação [Lopez 2017 apud, Fehre 2015].

4.2. Arquitetura *Model View Control* (MVC)

De acordo com GAMMA (1998), MVC consiste em três tipos de objetos que compõem camadas bem definidas: Modelo (*Model*) é o objeto da aplicação, responsável por implementar a lógica das regras de negócio tal qual o armazenamento persistente. Visão (*View*) é camada que compõe a apresentação do sistema para o usuário.

RUBY (2009) define o controlador (*Controller*) como o objeto que conduz a aplicação recebendo eventos do mundo externo, interagindo com o modelo e exibe uma visão apropriada [Duarte 2011].

4.3. Requisitos de Software

Requisitos de software são sentenças que expressam as necessidades dos clientes e que condicionam a qualidade do software, ou especificações de serviços que o sistema deve prover, restrições no sistema e conhecimentos necessários para desenvolvê-lo [Lessa 2013].

Os requisitos podem ser descritos no formato de casos de uso, levando em consideração cenários e comportamentos.

4.4. Prototipação

Este método para elicitação de requisitos utiliza-se do uso de uma técnica de prototipação para a avaliação do usuário. O conjunto inicial de requisitos é usado como base para criar o Protótipo de Interface do Usuário com o sistema inicial (simplificado). Para essa criação, o projetista precisa manter o protótipo tão simples quanto possível. O ponto forte desta atividade é apresentar muitas alternativas para o usuário antes de se gastar muito esforço para qualquer protótipo em particular. Após a aceitação do protótipo pelos usuários, os desenvolvedores precisam criar um documento de especificação dos requisitos paralelo ao protótipo de interface [Castro 2017, apud McConnel 1998].

A prototipação por *wireframe* é uma técnica de validação da coleta dos requisitos em relação a expectativa do usuário e produz a ilustração de baixa ou média fidelidade das telas do sistema proposto.

4.5. *Responsive Web Design*

O *Responsive Web Design* é uma técnica que permite a navegação num site a partir de diferentes dispositivos, ajustando-se, seja ele de que tamanho for, de forma a proporcionar uma melhor experiência ao leitor [Gomes 2014].

4.6. Produto Mínimo Viável

Produto mínimo viável (em inglês, *Minimum Viable Product* - MVP) é a versão mais simples de um produto que pode ser disponibilizada para validação de um pequeno conjunto de hipóteses sobre um negócio [Caroli 2015].

4.7. MongoDB

Segundo [Vieira 2017], MongoDB é um sistema *open-source* de banco de dados orientado a documentos, focado em alta performance possui características semelhantes a um banco de dados relacional, porém oferendo suporte a complexos tipos de dados. O MongoDB foi escrito em C++ e possui poderosos recursos a consulta volumosas devido aos dados persistidos em um estado aninhado.

5. Materiais e Métodos

Para o desenvolvimento deste trabalho, os requisitos do sistema foram levantados através da técnica de elicitação de requisitos. Depois foi aplicada a prototipação por *wireframe* para produzir ilustrações de baixa fidelidade das telas do sistema proposto.

Com a definição e validação dos requisitos, obteve-se insumos para a realização da engenharia de valor, técnica que estima o *backlog* que entrega o maior valor possível no menor prazo possível.

Foi projetado um MVP (Produto Viável Mínimo) construído por meio de um processo de desenvolvimento incremental, com ciclos de 15 dias de desenvolvimento permitindo desde a primeira entrega que o usuário validasse se o trabalho realizado agregava valor ao seu negócio.

O *framework* para desenvolvimento de aplicações *web* Angular em sua versão 7 foi utilizado para a implementação de toda aplicação *web* do GoByBus. Além de promover por padrão a arquitetura componentizada de aplicações *web* ainda conta com um vasto *suite* de ferramentas utilitárias que auxiliam o processo de desenvolvimento.

No desenvolvimento do GoByBus foi utilizado o *Node Package Manager* (NPM) como repositório de bibliotecas javascript de código aberto e gerenciador de pacotes.

Foi utilizado para tráfego de dados entre as aplicações *backend* e *frontend o JavaScript Object Notation* (JSON), formato de arquivo e objetos para tráfego de informações entre aplicações [JSON 2019].

A gestão e versionamento do código fonte do GoByBus foi realizada com o Git, ferramenta de controle de versão de código fonte bastante utilizada atualmente.

O *Spring Boot* auxiliou o processo de configuração de ambiente para a aplicação *backend* do GoByBus. Trata-se de uma ferramenta do *framework Spring* com objetivo promover a preparação inicial de uma aplicação JAVA de maneira fácil e rápida.

No desenvolvimento do GoByBus armazenamos dados do usuário no MongoDB.

O ambiente de desenvolvimento do GoByBus conta com os recursos disponibilizados pelo *Docker Compose* para orquestrar *containers*. Dessa forma garantimos que o *container* servidor da aplicação *backend* sempre será iniciado após a disponibilização do *container* servidor do banco de dados.

O GoByBus utiliza como modelo de autenticação o protocolo de autorização Auth0, que promove através de seus *bearer tokens* segurança no consumo dos serviços da aplicação [Auth0 2019].

A funcionalidade principal do GoByBus baseia-se na apresentação de informações de viagem a partir do informe de dois pontos distintos, origem e destino. Para obter essas informações, utilizamos alguns serviços da *Google Maps API*, mantida pela Google, com a finalidade de oferecer vários serviços baseados em dados geográficos. Os dados são coletados pelas diferentes aplicações que a empresa suporta no mercado, o consumo comercial desses dados é monetizado [Google 2019]. A API segmenta os serviços oferecidos em três tipos de informações, visualização de mapas, rotas ou lugares.

Para consumir os dados dessa API é necessário que o desenvolvedor acesse o portal de serviços da Google em *Cloud*, selecione o produto desejado, crie um projeto e configure o faturamento. Para utilização dos serviços é necessário obter uma chave de autorização a ser utilizada em todas as requisições.

As APIs são REST, com requisições https utilizando a URI base https://maps.googleapis.com/maps/api/, seguida dos dados de identificação do serviço a ser consumido, o tipo de saída desejada e por último os parâmetros específicos para o consumo de cada serviço. Por fim, é enviada chave de acesso obtida anteriormente. A Figura 12 mostra a requisição utilizada para obter os dados referentes às viagens.

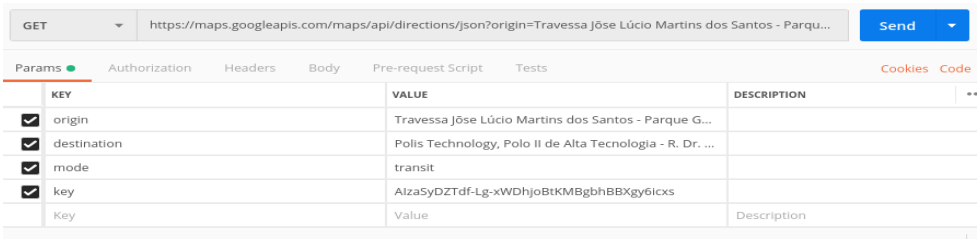

Figura 12. Requisição enviada para o Google Maps API

Os parâmetros específicos utilizados nessa requisição são *origin* representando o ponto de partida do usuário, *destination* representando o ponto final do usuário e *mode* representando o modo de transporte do usuário.

Os parâmetros *origin* e *destination* são strings inseridas pelo próprio usuário, já o parâmetro *mode* é fixo com valor *transit* como string, indicando que as rotas devem considerar o modo por transporte público disponível. Os tipos de transportes que podem ser retornados são exibidos na Figura 13. A Figura 14 mostra uma representação do objeto retornado pelo *Google Maps API*.

Value	Definition
RAIL	Rail.
METRO_RAIL	Light rail transit.
SUBWAY	Underground light rail.
TRAM	Above ground light rail.
MONORAIL	Monorail.
HEAVY_RAIL	Heavy rail.
COMMUTER_TRAIN	Commuter rail.
HIGH_SPEED_TRAIN	High speed train.
LONG_DISTANCE_TRAIN	Long distance train.
BUS	Bus.
INTERCITY_BUS	Intercity bus.
TROLLEYBUS	Trolleybus.
SHARE_TAXI	Share taxi is a kind of bus with the ability to drop off and pick up passengers anywhere on its route.
FERRY	Ferry.
CABLE_CAR	A vehicle that operates on a cable, usually on the ground. Aerial cable cars may be of the type GONDOLA_LIFT.
GONDOLA_LIFT	An aerial cable car.
FUNICULAR	A vehicle that is pulled up a steep incline by a cable. A Funicular typically consists of two cars, with each car acting as a counterweight for the other.
OTHER	All other vehicles will return this type.

Figura 13. Listagem das opções de transportes retornadas pelo Google Maps API

Fonte: [Google 2019]


```
{
   "geocoded_waypoints": [
      {
         "geocoder_status": "OK",
         "place_id": "ChIJfaDjTFi5yJQR9lXmllvoAc8",
         "types": [
            "route"
         ]
      },
      {
         "geocoder_status": "OK",
         "place_id": "ChIJ8b-3aw7EyJQRwxQnQAPPexk",
         "types": [
            "establishment",
            "point_of_interest"
         ]
      }
   ],
   "routes": [
      {
         "bounds": {
            "northeast": {
               "lat": -22.7873117,
               "lng": -47.0299683
            },
            "southwest": {
               "lat": -22.9081202,
               "lng": -47.2120197
            }
         },
         "copyrights": "Map data ©2019 Google",
         "legs": [
            {
               "arrival_time": {
                  "text": "9:44pm",
                  "time_zone": "America/Sao_Paulo",
                  "value": 1563497042
               },
               "departure_time": {
                  "text": "7:38pm",
                  "time_zone": "America/Sao_Paulo",
                  "value": 1563489493
               },
               "distance": {
                  "text": "40.0 km",
                  "value": 39958
               },
               "duration": {
                  "text": "2 hours 6 mins",
                  "value": 7549
               },
               "end_address": "Polo II de Alta Tecnologia - R. Dr. Ricardo Benetton Martins, 1000 - Bosque das Palmeiras, Campinas - SP, 13086-510, Brazil",
               "end_location": {
                  "lat": -22.8169739,
                  "lng": -47.0449446
               },
               "start_address": "Travessa Jõse Lúcio Martins dos Santos - Parque Gabriel, Hortolândia - SP, 13186-033, Brazil",
               "start_location": {
                  "lat": -22.8753843,
                  "lng": -47.2120197
               },
```


Figura 14. Representação do objeto retornado pelo Google Maps API

6. Resultados

6.1. Elicitação dos Requisitos

Levando em consideração o levantamento dos aplicativos correlatos e a aplicação da pesquisa de opinião, foi realizado o processo de levantamento e validação dos requisitos do sistema. Esse processo resultou na identificação dos casos de uso representados na Figura 15.

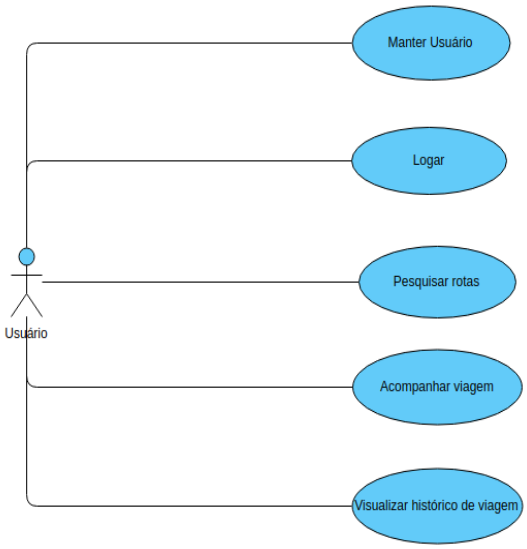

Figura 15. Diagrama de Casos de Uso do GoByBus

Como usuário sem cadastro ao acessar o *site* GoByBus, a tela principal é exibida com as opções para cadastrar ou logar no aplicativo, ilustrado na Figura 16. A partir desta tela, são acessadas as funcionalidades do sistema. A seguir são descritos os casos de uso identificados para compor a versão inicial do MVP:

Figura 16. Protótipo da tela inicial da aplicação web GoByBus

1. Manter Usuário: para utilização do aplicativo é necessário um cadastro prévio do usuário, com informações pessoais, locais e financeiras. O usuário deverá inserir seu e-mail, nome, telefone e senha. Na tela apresentada na Figura 17, ao inserir com sucesso todas as informações o usuário poderá concluir o cadastro, onde o sistema exibirá a mensagem "Cadastro realizado com sucesso, um e-mail de confirmação foi enviado a você!".

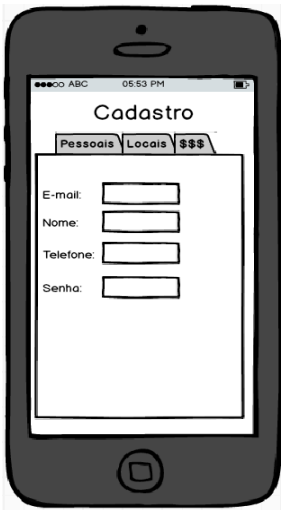

Figura 17. Protótipo da tela de cadastro da aplicação web GoByBus

2. Logar: esse caso de uso será acessado para o usuário cadastrado. Na tela inicial (Figura 16), ao clicar em "Logar" o usuário irá inserir seu e-mail e senha, conforme ilustrado na Figura 18. Esses campos devem conter validações de preenchimento e conteúdo. Deve haver o botão "Logar" novamente para validar as credenciais do usuário, liberando o acesso ao aplicativo e carregando o perfil do usuário. O sistema deverá apresentar mensagens de indicação dos erros ao usuário quando houverem, como " preenchimento obrigatório", " e-mail inválido" entre outros. Quando, por qualquer motivo, as credenciais do usuário não forem validadas, apresentar mensagem ao usuário informando o ocorrido.

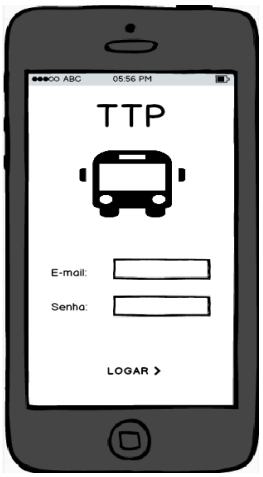

Figura 18. Protótipo da tela de login da aplicação web GoByBus

3. Pesquisar Rotas: após o login no sistema, o usuário deve preencher os endereços de partida e destino (Figura 19). O usuário visualizará as linhas de ônibus disponíveis e o detalhamento das mesmas quando selecionadas. O usuário poderá solicitar embarque nas linhas disponíveis e selecionadas. No detalhamento da linha de ônibus deve haver o botão "Embarcar" o qual apresentará o tempo de espera previsto.

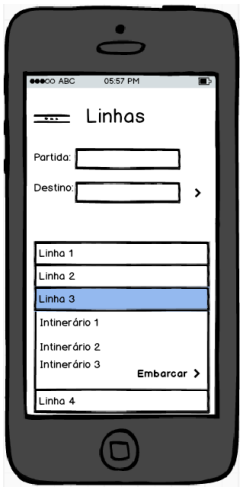

Figura 19. Protótipo da tela de pesquisa de rotas da aplicação web GoByBus

4. Acompanhar Viagem: Após a solicitação da viagem o usuário poderá visualizar em tela o trajeto que está percorrendo. Receberá notificações relacionadas ao tempo para embarque ou desembarque em seu destino. Poderá ainda, a qualquer momento, cancelar sua viagem, conforme a ilustração da Figura 20.
5. Visualizar Histórico de Viagens: o usuário logado ao clicar no menu lateral, pode selecionar a opção "Histórico de viagens" (Figura 21) para visualizar a listagem de todas as viagens realizadas. Caso o usuário selecione um registro da listagem, o mesmo deverá apresentar o detalhamento da viagem com as informações: trajeto, horários, valores e linha.

6.2. Autenticação Auth0

Auth0 é um *Open Standard* que provê autenticação e autorização como serviço. A API Auth0 oferece um *endPoint* para que o usuário possa fazer *login*, em caso de sucesso o retorno é um *token* de autorização. Este método de autenticação é o *OAuth2 Access Token*, que é requerido no *header* das requisições que precisam de autenticação [Auth0 2019].

Para implementação da API de autenticação foi usado o *Framework* Spring *security* para fazer o gerenciamento da camada de segurança da aplicação. Como dito anteriormente, ele oferece a implementação para o OAuth2, é altamente editável, podendo editar a classe de autenticação e possuí prevenções XSS e CRFS etc.

Figura 20. Protótipo da tela de acompanhamento de viagem da aplicação web GoByBus

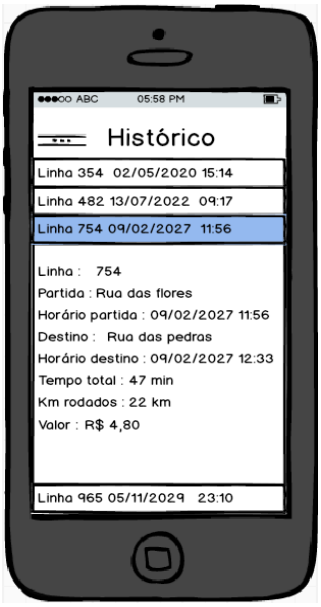

Figura 21. Protótipo da tela de histórico de viagens da aplicação web GoByBus

6.3. Arquitetura *Frontend*

A arquitetura utilizada na aplicação *web* do projeto foi a hexagonal. Esta arquitetura parte do princípio da modularização dos componentes que representam uma funcionalidade da aplicação. A Figura 22 mostra a ilustração da estruturação do aplicativo em módulos com suas funcionalidades principais.

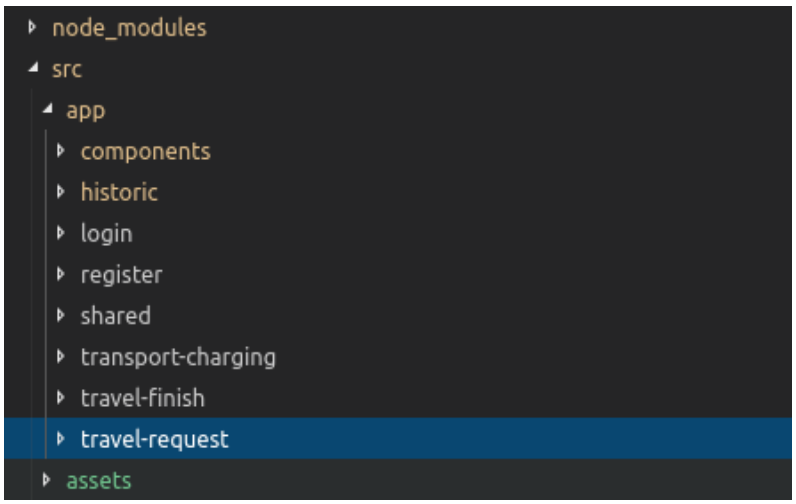

Figura 22. Imagem da estrutura do projeto *frontend* GoByBus

Dessa forma as funcionalidades segmentadas em módulos devem conter todas as dependências e mecanismos para seu funcionamento independentes e testáveis, podendo ser exportadas e importadas em diferentes partes da aplicação.

A Figura 23 apresenta a ilustração da expansão de um de seus módulos, login, e a representação dos artefatos que compõem todo o mecanismo do módulo, html, scss, spec, *component* e *module*. Essa abordagem promoveu a reutilização de componentes nas diferentes páginas da aplicação.

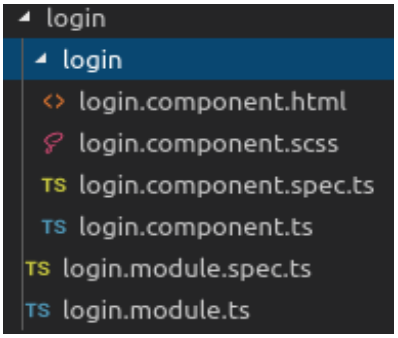

Figura 23. Imagem dos artefatos que compõe os módulos no projeto *frontend* GoByBus

6.4. Desenvolvimento do Aplicativo e *Webservice*

O desenvolvimento dos aplicativos *frontend* e *backend* foi realizado de maneira paralela no decorrer das *Sprints* que seguiam o planejamento do *backlog*. Esta seção contém a descrição da implementação incremental dos casos de uso.

A arquitetura de sistema desenvolvida para atender aos requisitos do software, ilustrada na Figura 24, é composta por duas camadas, sendo estas *client* e *server* respectivamente. A camada *client* é responsável pela interface de interação do usuário com a aplicação. Nesta camada podem coexistir interfaces para diferentes *devices* conforme a necessidade de negócio. Esta camada é totalmente independente, contendo apenas as regras de negócio referente a jornada do usuário na aplicação. A camada *server* contém recursos para prover todas as informações processadas e persistidas para a camada *client*. É a camada responsável por todas as regras de negócio da aplicação.

É possível observar na Figura 24 que a camada *server* é composta por um *gateway*, um *load balance* e *containers* autônomos. O *gateway* é responsável por receber as requisições da camada *client* e endereçar para o serviço adequado. O *load balance* realiza o balanceamento da quantidade de requisições recebidas para cada serviço, evitando qualquer sobrecarga de serviços independente da quantidade de requisições realizadas pelo *client*. Os *containers* autônomos têm responsabilidade única referente a cada funcionalidade da aplicação. Possuem uma estrutura que viabiliza seu funcionamento independente e desacoplado, contendo cada um servidor de aplicação e um banco de dados.

O principal objetivo da arquitetura sugerida é manter uma camada *server* performática, organizada e autônoma para servir as regras de negócio para qualquer aplicação *client* permitida.

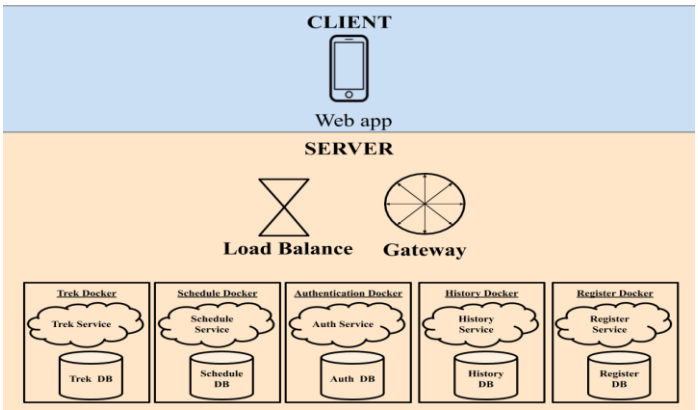

Figura 24. Imagem da arquitetura da aplicação GoByBus

6.4.1. Sprint 1: Manter Usuário e Logar

A *Sprint* 1 teve como objetivo principal o desenvolvimento dos casos de uso de cadastro e login de usuário, atendendo aos casos de uso Manter Usuário e Logar. Inicialmente considerou-se a fase de criação e configuração dos projetos *backend* e *frontend*.

Foi realizado o desenvolvimento do serviço de cadastro e o serviço de autenticação de usuário. Chamado *signin,* o serviço de cadastro é do tipo *post* e possui os parâmetros *name, key, contact, email, address, number address, credit card number e ccv*, todos do tipo *string*. Seu retorno é composto pelo status da requisição.

O desenvolvimento do módulo *frontend* de cadastro contou com formulários reativos para criação do formulário de inserção dos dados do usuário utilizando validação padrão do *framework*. A estilização foi possível através da utilização do pré processador de folhas de estilos scss com algumas animações. Houve a preocupação em manter um *design* leve e confortável, como mostra a Figura 25.

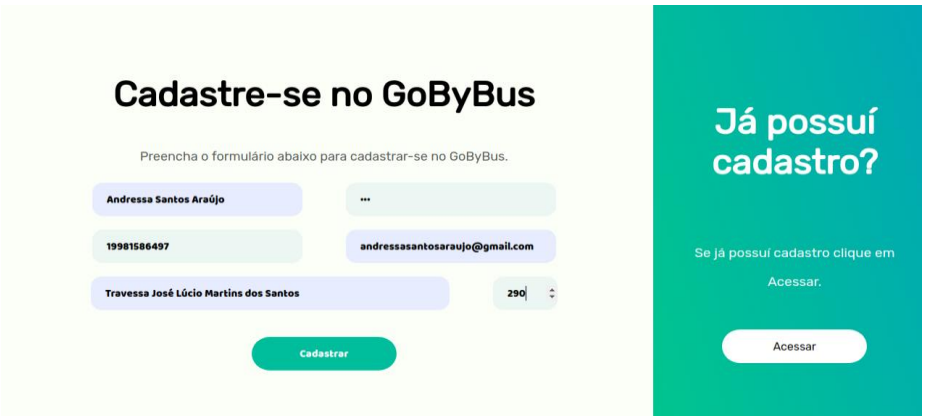

Figura 25. Imagem da tela de cadastro da aplicação

Já o serviço de autenticação chamado login do tipo *post,* tem como parâmetros o nome e a senha cadastrados pelo usuário. Compõe o retorno do serviço um *bearer token*. Como podemos observar na Figura 26, a preocupação no desenvolvimento desse módulo de login no aspecto do *frontend* foi reutilizar os componentes criados no módulo de cadastro e manter o padrão de interface.

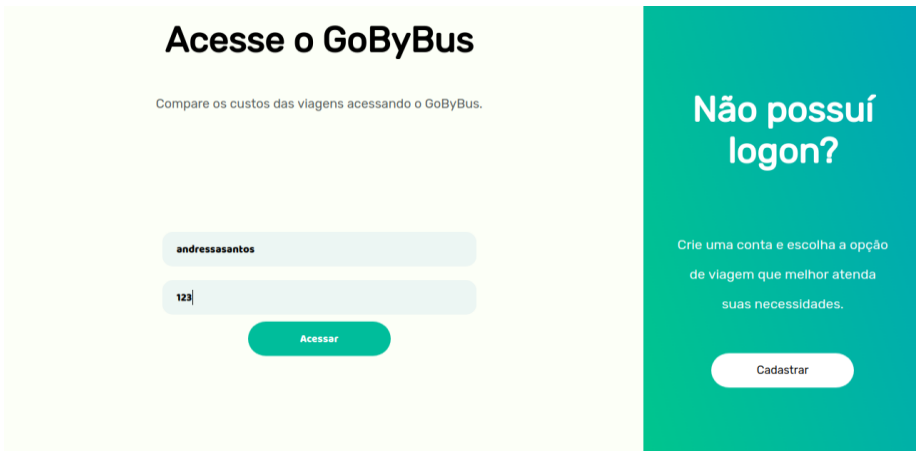

Figura 26. Imagem da tela de login

6.4.2. Sprint 2: Pesquisar Rotas

O maior valor agregado e complexidade ao negócio são concentrados no fluxo de pesquisa de rotas. Para o desenvolvimento desta funcionalidade, foi implementado todo o mecanismo de consumo das informações do serviço de *directions* do Google API. Foi utilizado também uma biblioteca utilitária do *framework* Angular para apresentação do mapa com as localizações inseridas. A Figura 27 mostra a interface desenvolvida.

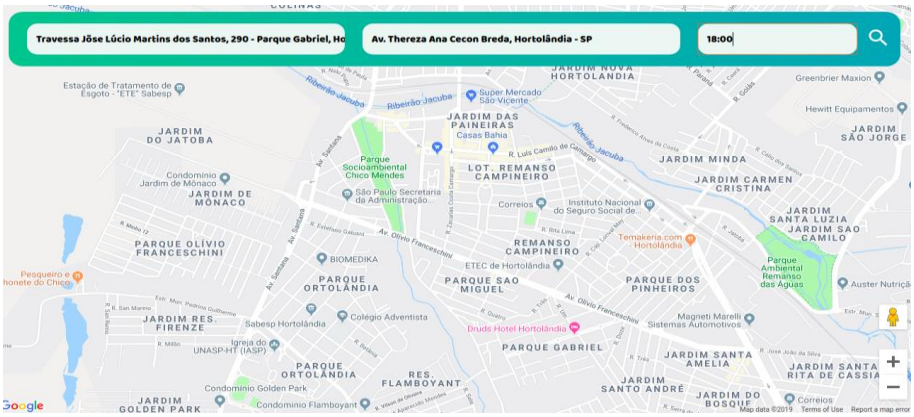

Figura 27. Imagem da tela de pesquisa de viagens

Com o retorno da requisição do serviço de *directions*, após a indicação das localizações e horários, o objeto JSON é processado e as informações são listadas (Figura 28) com eventos de seleção por linha da listagem.

Desenvolveu-se ainda uma tela para visualização e confirmação das informações da viagem a ser selecionada, com opção de cancelamento, conforme é possível visualizar na Figura 29.

6.4.3. Sprint 3: Acompanhar Viagem e Visualizar Histórico de Viagens

Na terceira *sprint* desenvolveu-se um componente baseado em um contador, para apresentar o tempo restante para início da viagem, levando-se em consideração o horário de início da viagem selecionada pelo usuário. Na Figura 30 é possível observar a utilização do serviço pelo componente de mensageria.

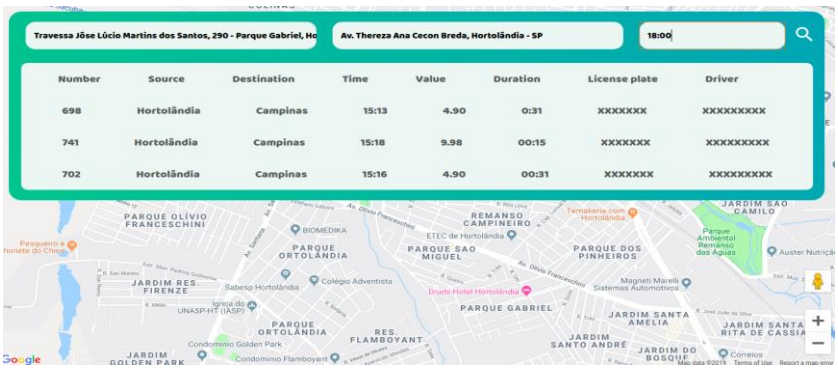

Figura 28. Imagem da listagem de viagens

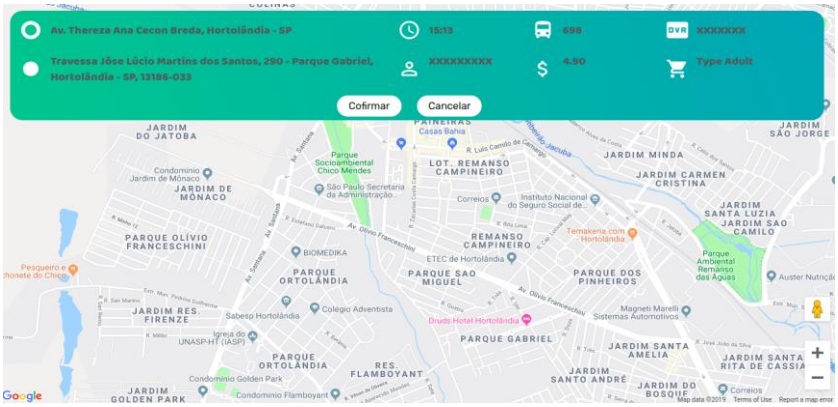

Figura 29. Imagem do detalhamento da viagem selecionada

Figura 30. Imagem da tela de acompanhamento de viagens

Um serviço para persistência das viagens é utilizado para obter os dados do histórico de viagens. Este serviço é consumido pelo componente de listagem das viagens na tela de pesquisa e seleção das viagens. Reutilizou-se o componente de listagem e seleção das viagens na tela de pesquisa de viagens, reduzindo consideravelmente a complexidade desta tela.

7. Conclusão

Este artigo abordou a análise e desenvolvimento do aplicativo GoByBus o qual apresenta informações tarifárias do transporte público.

Para o processo de elicitação de requisitos foram aplicados os conhecimentos adquiridos na disciplina Análise Orientada a Objetos. Os conhecimentos adquiridos na disciplina de Engenharia e Qualidade de Software foram utilizados para construção do *backlog* baseado na técnica de engenharia de valor. O conhecimento adquirido nas disciplinas de Programação Orientada a Objetos, Estrutura de Dados, Linguagem de Programação, Algoritmos e Programação e Banco de Dados foi aplicado no desenvolvimento dos aplicativos *frontend* e *backend*.

Considera-se viável tecnicamente, o desenvolvimento de uma aplicação *web,* para apresentação das informações, referentes ao custo de viagens do transporte público.

Os aplicativos correlatos EMTU, *Transurc Smart*, Moovit e *Google Maps* possuem características distintas que oferecem informações aos usuários. Podemos observar na Tabela 2 que o itinerário é uma informação provida ao usuário pelos aplicativos EMTU, Moovit e *Google Maps*.

Já informações tarifárias são apresentadas aos usuários pelos aplicativos EMTU, *Transurc Smart* e *Google Maps*.

Com excessão do *Transurc Smart* todos os aplicativos mencionados anteriormente são multiplataforma.

Relatórios de consumo é uma informação provida ao usuário apenas pela aplicação *Transurc Smart*.

Tabela 2. Comparativo entre o GoByBus e aplicativos correlatos

Aplicativo	Itinerário	Inf. Tarifárias	Multiplataforma	Relatórios
EMTU	x	x	x	
Transurc Smart		x		x
Moovit	x		x	
Google Maps	x	x	x	
GoByBus	x	x	x	x

A aplicação GoByBus propõe a apresentação das informações consideradas na Tabela 2, aos potenciais usuários como insumos, para escolha da opção de viagem que melhor atende suas necessidades.

Como trabalhos futuros sugerimos a disponibilização da aplicação GoByBus para um grupo de usuários, a fim de obter uma percepção do valor agregado ao processo de escolha da viagem.

Referências

Auth0, D. (2019). Auth0 Docs. https://auth0.com/docs/api/authentication#introduction. [Online; Acessado em 08 de Junho de 2019].

Campos, V. (2014). Uma Visão da Mobilidade Urbana Sustentável. (Abril).

Caroli, P. (2016) Direto ao ponto; criando produtos de forma enxuta, Casa do Código, 2a edição. (Março).

Castro, J. (2017) Engenharia de Requisitos: Uma Introdução a Engenharia de Requisitos. (Janeiro).

Duarte, A (2011). Metodologia Rails: Análise da Arquitetura Model View Controller Aplicada. http://www.bibliotecadigital.ufmg.br/dspace/bitstream/handle/1843/BUOS-94MMY9/andreyrochaduarte.pdf?sequence=1, (Maio).

EMTU (2019). EMTU.http://www.emtu.sp.gov.br/emtu/itinerarios-e-tarifas/app-emtu.fss. [Online; Acessado em 18 de Agosto de 2019].

Fiegenbaum, Jader Osvino. (2017). Estudo comparativo entre os sistemas gerenciadores de banco de dados PostgreSQL e MongoDB para o armazenamento e busca de metadados MARC. http://hdl.handle.net/10737/1251, (Janeiro).

Gamma, E et al. (1998). Design Paterns: Elements of Reusable Object-Oriented Software. (Junho).

Gomes, Cláudia Raquel Castro. (2014). Responsive web design, um novo modelo integrador para a informação de hoje. http://hdl.handle.net/10400.26/8226, Outubro.

Gomilde, A. (2003). Transporte Urbano e Inclusão Social: Elementos para Políticas Públicas. (Julho).

Google. (2019). Google Maps Plataform. https://developers.google.com/maps/documentation/directions/intro#DirectionsRequests. [Online; Acessado em 13 de Maio de 2019].

Json.org (2018). Json. http://www.json.org/. [Online; Acessado em 08 de Agosto de 2018].

Leite, A, Rocha, J e Baptista, C. (2012). Busão: um Sistema de Informações Móvel para Auxílio á Mobilidade Urbano Através do Uso de Transporte Coletivo. (Dezembro).

Len Bass, Paul Clements, R. K. (2003). Software Architecture in Practice, Second Edition. Addison Wesley.

Lessa, R. e Junior, E. (2013). Modelos de Processos de Engenharia de Software. https://ead.uepg.br/apl/sigma/assets/editais/PS0059E0080.pdf, Outubro.

López, D. (2017). Aquitectura De Software Basada En Microservicios Para Desarrollo De Aplicaciones Web De La Asamblea Nacional, Setembro.

Moovit. (2019). Moovit. https://moovitapp.com/. [Online; Acessado em 10 de Julho de 2019].

Ruby, S. Thomas, D. Hansson, D. (2009). Agile Web Development With Rails. 3 ed. Raleigh: The Pragmatic Bookshelf. (Setembro).

Silva, D. (2000). Sistemas Inteligentes no Transporte Público Coletivo por Ônibus. (Maio).

Sommerville, I. (2006). Engenharia de Software. Pearson, nona edição.

Transurc, Smart. (2019). Transurc Smart. https://www.transurc.com.br/index.php/novo-app-transurc-smart-facilita-compra-de-qr-code/. [Online; Acessado em 15 de Agosto de 2019].

Verroni, J. (2006). Tarifa do transporte público urbano por ônibus: uma contribuição para determinação de seu valor. (Maio).

Vieira, M. R. et al. (2019). Bancos de dados nosql: conceitos, ferramentas, linguagens estudos de casos no contexto de big data. Simpósio Brasileiro de Banco de Dados. (Julho).

Zhao, H. and Doshi, P. (2009). Towards automated restful web service composition. In 2009 IEEE International Conference on Web Services, pages 189–196.

Documento Digitalizado Restrito

Anexo I - TCC Andressa Santos Araujo

Assunto: Anexo I - TCC Andressa Santos Araujo
Assinado por: Daniela Marques
Tipo do Documento: Relatorio
Situação: Finalizado
Nível de Acesso: Restrito
Tipo do Conferência: Documento Original

Documento assinado eletronicamente por:

- **Daniela Marques**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 04/11/2019 09:42:28.

Este documento foi armazenado no SUAP em 04/11/2019. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 275416
Código de Autenticação: 3ef3d1f7a8