VALDECI SOARES DA SILVA JUNIOR

RELATÓRIO DE ESTÁGIO - DESENVOLVIMENTO E MANUTENÇÃO DE SOFTWARE NA EMPRESA TECHNOLOG

LAVRAS – MG

2019

VALDECI SOARES DA SILVA JUNIOR

RELATÓRIO DE ESTÁGIO - DESENVOLVIMENTO E MANUTENÇÃO DE SOFTWARE NA EMPRESA TECHNOLOG

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Sistemas de Informação, para obtenção do título de Bacharel.

Prof. Paulo Afonso Parreira Júnior
Orientador

LAVRAS - MG
2019

VALDECI SOARES DA SILVA JUNIOR

RELATÓRIO DE ESTÁGIO - DESENVOLVIMENTO E MANUTENÇÃO DE SOFTWARE NA EMPRESA TECHNOLOG

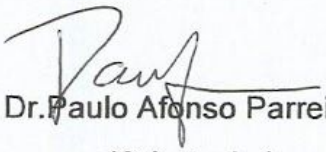
TRAINING REPORT - SOFTWARE DEVELOPMENT AND MAINTENANCE AT TECHNOLOG COMPANY

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Sistemas de Informação, para obtenção do título de Bacharel.

APROVADA em 12 de Novembro de 2019.

Profa. Dra. Renata Teles Moreira UFLA

Me. Sebastião Giovani dos Reis Technolog



Prof. Dr. Paulo Afonso Parreira Júnior
(Orientador)

LAVRAS – MG

2019

À minha mãe Elenice Aparecida Elias, pelo apoio e carinho, principal e essencial ajuda, da qual, na falta teria tornado este caminho muito mais árduo.

Ao meu pai Valdeci Soares da Silva, minha família e amigos pelo apoio e incentivo.

“A teoria sem a prática vira 'verbalismo', assim como a prática sem teoria, vira ativismo. No entanto, quando se une a prática com a teoria tem-se a práxis, a ação criadora e modificadora da realidade.” (Paulo Freire)

Dedico

AGRADECIMENTOS

À Universidade Federal de Lavras, especialmente ao Departamento de Ciência da Computação, pela oportunidade.

À TECHNOLOG, pela concessão do estágio.

Ao professor Dr. Paulo Afonso Parreira Júnior, pela orientação, paciência e disposição para ajudar.

[...] A todos funcionários do DCC/UFLA.

A todos os colegas de departamento, pois muitos se tornaram amigos.

Aos meus pais, em especial minha mãe Elenice Aparecida Elias pelo amor e apoio incondicional, em todas as minhas decisões nas diferentes etapas da minha vida e aos meus amigos e colegas.

À Jéssica Aparecida Carvalho Pereira, pelo companheirismo e apoio em todos os momentos e singular torcida.

MUITO OBRIGADO!

“A alegria não chega apenas no encontro do achado, mas faz parte do processo da busca. E ensinar e aprender não pode dar-se fora da procura, fora da boniteza e da alegria.” (Paulo Freire)

RESUMO

Este relatório de estágio descreve as atividades de desenvolvimento e manutenção de software para gestão de transportadoras, realizado na empresa Technolog. Os sistemas de software que mantidos/aprimorados são *Darwin* (responsável pela gestão de multas, pedágios, dentre outras funcionalidades) e *Manager* (responsável pela gestão dos abastecimentos, recuperação de alguns dados de posição dos veículos, entre outros). Neste relatório, são apresentadas as descrições das atividades desenvolvidas, juntamente com as tecnologias e metodologias empregadas durante o período de estágio. Como conclusão do estágio, destaca-se que o mesmo pode aprimorar a qualificação técnica do futuro profissional, permitindo que ele possa, dentre outras coisas, praticar a habilidade de trabalho em equipe, aplicar os conceitos aprendidos em sala de aula, dentre outras coisas.

Palavras-chave: Desenvolvimento ágil de software e Desenvolvimento *Web.*

ABSTRACT

This training report describes the activities of development and maintenance of software at Technolog. The software systems that were maintained are Darwin (responsible for managing fines, tolls, among others) and Manager (responsible for the management of supplies and positions of the vehicles, among others). This report presents descriptions of the activities carried out, along with the technologies and methodologies used during the internship period. As a conclusion of this training period, it is possible to highlight that a training experience may improve the technical qualification of the future professional, allowing him/her to, among other things, practice the teamwork skill and applying the concepts learned in the classroom.

Keywords: Agile software development and Web development.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

JS	JavaScript
front	Front-End
back	Back-End
cod	código
id	identificador

LISTA DE SIGLAS

ANATEL	Agência Nacional de Telecomunicações
API	*Application Programming Interface*
CNPJ	Cadastro Nacional de Pessoa Jurídica
CPF	Cadastro de Pessoa Física
CRUD	*Create, Read, Update and Delete*
CSS	*Cascading Style Sheets*
DCC	Departamento de Ciência da Computação
ERP	*Enterprise Resource Planning*
FTP	*File Transfer Protocol*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
JSON	*JavaScript Object Notation*
PHP	*Hypertext Preprocessor*
PO	*Product Owner*
POI	*Point Of Interest*
SCP	*Secure Copy Protocol*
SFTP	*SSH File Transfer Protocol*
SQL	*Structured Query Language*
SSH	*Secure SHell*
SSMS	*SQL Server Management Studio*
TI	Tecnologia da Informação
UFLA	Universidade Federal de Lavras
VPN	*Virtual Private Network*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo, é apresentada uma breve descrição da empresa Technolog, bem como de alguns dos problemas que foram tratados neste estágio supervisionado.

1.1 Contextualização

Nos últimos anos, um grande número de empresas de desenvolvimento de software tem-se estabelecido na cidade de Lavras/MG. Segundo o Empresômetro[1], um *website* especializado em pesquisas sobre instalação de empresas em todo país, atualmente existem 38 (trinta e oito) empresas nesta cidade. Trata-se de um número significativo para uma cidade com uma população de aproximadamente 92 mil habitantes, de acordo o último censo (2010)[2].

Esse aumento expressivo na quantidade de empresas pode ter sido ocasionado pela procura por mão-de-obra especializada, uma vez que a UFLA (Universidade Federal de Lavras), que oferece os cursos de Ciência da Computação e Sistemas de Informação, é reconhecidamente uma das melhores instituições públicas de ensino superior do Brasil[3].

Devido a isso, houve um aumento na oferta de vagas de estágios supervisionados aos alunos do Departamento de Ciência da Computação (DCC) da UFLA, uma oportunidade ímpar para que estes possam se preparar melhor para o mercado de trabalho. O estágio supervisionado permite ao aluno colocar em prática os conhecimentos teóricos adquiridos nas disciplinas de graduação, além de promover a motivação pessoal, bem como expandir seu conhecimento a respeito do conteúdo ministrado pelos professores em sala de aula.

Neste contexto, o presente relatório de estágio descreve as atividades de desenvolvimento e manutenção de software, desenvolvidas durante o estágio supervisionado na empresa Technolog, no período de Setembro/2018 à Agosto/2019. Por meio desse relatório, procura-se ainda, ressaltar a importância da

[1] www.empresometro.com.br
[2] https://cidades.ibge.gov.br/brasil/mg/lavras/panorama
[3] https://guiadoestudante.abril.com.br/

experiência prática, aliada aos conhecimentos teóricos adquiridos pelo acadêmico, autor do presente relatório.

1.2 Sobre a Technolog e seus produtos de software

O estágio supervisionado relatado neste trabalho foi realizado na empresa Technolog, que possui sede na cidade de Lavras/MG. A empresa conta com 50 (cinquenta) colaboradores, sendo que 8 (oito) funcionários contratados e 3 (três) estagiários estão diretamente vinculados à área de Tecnologia de Informação (TI). Os demais funcionários compõem o corpo de analistas de logística, responsáveis pela gestão de empresas clientes da Technolog, e a equipe de manutenção dos equipamentos instalados nos clientes.

O "carro-chefe" da empresa é o controle de combustível de veículos de transportadoras (empresas clientes da Technolog). Esse controle é realizado pelo software *Darwin*, que atua como um sistema CRM (*Customer Relationship Management*)[4]. O *Darwin* oferece diversos painéis para auxiliar neste controle, como por exemplo o *Volume Atual*, que mostra a quantidade de combustível nos tanques dos veículos. Essa informação é importante, pois esses veículos são de carga pesada, tais como carretas e caminhões, que possuem, em geral, quatro tanques de combustível com capacidade de duzentos e cinquenta litros cada um, em média. Assim sendo, o abastecimento desses veículos representa um dos maiores custos de uma viagem e, consequentemente, de uma transportadora.

O software *Darwin* é composto por vários módulos, tais como custos, controle de frota, identificação dos motoristas, dentre outros. Para fazer a gestão de quais módulos do *Darwin* estão disponibilizados para cada cliente, utiliza-se o software *Manager*, que também é de autoria da Technolog. Outra responsabilidade do *Manager* é a gestão dos clientes primários e secundários da empresa. Nele, é possível realizar o cadastro, alteração e exclusão dos clientes da empresa, assim como os clientes secundários, tais como postos de combustíveis, praças e concessionárias de pedágios, entre outros.

[4] Sistema CRM são softwares desenvolvidos com o objetivo de auxiliar na gestão do relacionamento com o cliente.

1.3 Objetivos

A partir da oferta dos serviços realizados pela empresa, por meio dos seus produtos de software, surge a necessidade de implementar novas funcionalidades nestes produtos, bem como realizar modificações nas que já existem. Assim sendo, este estágio supervisionado teve como objetivo o desenvolvimento de novas funcionalidades, bem como a manutenção de algumas funcionalidades já existentes em produtos de software da Technolog, conforme apresentado a seguir:

Darwin. Como exemplos de novas funcionalidades desenvolvidas pelo autor deste relatório neste software, estão: (i) a criação do módulo de lançamento de multas; (ii) a realização de traduções do sistema para língua Inglesa e Hispânica; e (iii) o desenvolvimento de APIs (*Application Programming Interface*[5]) para integração da funcionalidade do *Darwin* com aplicativos móveis.

Manager. Neste software, não foram desenvolvidas novas funcionalidades. Contudo, manutenções foram realizadas, tais como: (i) permitir alteração de período no filtro do módulo “forçar carga”; (ii) adição da opção "SimplePM", um *chip* multi-operadoras, na interface do sistema; e (iii) adição do volume computado pelos postos de combustíveis.

Mais detalhes sobre as atividades comentadas anteriormente podem ser encontrados na Seção 3 deste relatório.

1.4 Estrutura do trabalho

Este trabalho está organizado da seguinte forma: no Capítulo 2, são apresentados alguns conceitos básicos a respeito das técnicas e ferramentas utilizadas ao longo do estágio; no Capítulos 3, são apresentados as atividades desenvolvidas durante o período de estágio, bem como os resultados obtidos a partir delas; por fim, no Capítulo 4 são apresentadas as considerações finais.

[5] Uma API é um conjunto de rotinas e padrões estabelecidos por um software para a utilização de sua funcionalidade por outros produtos de software, os quais não pretendem envolver-se em detalhes da implementação do software, mas apenas em usar seus serviços.

2 REFERENCIAL TEÓRICO

Visto que este estágio teve como base o desenvolvimento e a manutenção de sistemas de software, é necessário que alguns conceitos sejam definidos. Neste capítulo, são apresentados os conceitos de Scrum e quadro de tarefas Kanban, assim como API e algumas tecnologias utilizadas para desenvolvimento web, tais como HTML (*Hypertext Markup Language*), JavaScript, CSS (*Cascading Style Sheets*), dentre outras.

2.1 Scrum

Scrum[6] é uma metodologia ágil para gestão e planejamento de projetos de software. Com Scrum, os projetos são divididos em ciclos chamados de *Sprints*. Um *Sprint* representa um período de tempo dentro do qual um conjunto de atividades deve ser executado. A funcionalidade a ser implementada em um projeto é mantida em uma lista, a qual é conhecida como *Product Backlog*. No início de cada *Sprint*, faz-se uma *Sprint Planning Meeting*, ou seja, uma reunião de planejamento na qual o PO (*Product Owner)* define os itens que compõem o *Product Backlog*.

Na *Sprint Planning Meeting,* a equipe seleciona as atividades que ela será capaz de implementar durante o *Sprint*. As tarefas alocadas em um *Sprint* são transferidas do *Product Backlog* para o *Sprint Backlog*.

A cada dia de uma *Sprint*, a equipe faz uma breve reunião, chamada *Daily Scrum*. O objetivo é disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do dia que se inicia. Ao final de um *Sprint*, a equipe apresenta as funcionalidades implementadas em uma *Sprint Review Meeting*.

Finalmente o *Scrum Master*, responsável por garantir que os valores e práticas do Scrum estejam vivos dentro do dia-a-dia da equipe, faz uma *Sprint Retrospective,* reunião que ocorre ao final de um *Sprint* com o objetivo de identificar o que funcionou bem, o que pode ser melhorado e que ações serão tomadas para

[6] A descrição das atividades e papéis desta metodologia foi realizada com base no trabalho de Sutherland (2014).

melhorar. Após o fim da *Sprint Retrospective* a equipe parte para o planejamento do próximo *Sprint*; assim reinicia-se o ciclo.

Na Technolog, o *Scrum* é utilizado em conjunto com o *Kanban*, que é apresentado na Seção 2.2, e sua aplicação é apoiada pelo apoio computacional Plan.io, apresentado na Seção 2.6.

2.2 Kanban

A tradução literal da palavra kanban é “anotação visível, ou sinal” (JUNIOR, M.L; FILHO, M. G, 2007). De modo geral, vem-se empregando na literatura esta palavra com o significado de “cartão”, pois o sistema Kanban é conhecido por empregar cartões para informar a necessidade de entregar e/ou produzir certa quantidade de peças, matérias-primas ou resultados. Em muitos trabalhos, entretanto, nota-se a utilização indiscriminada da palavra Kanban, significando tanto “cartão”, como uma metodologia para gerenciar determinado trabalho.

Neste relatório de estágio, utiliza-se a seguinte distinção de termos (LAGE, 2008): os cartões ou sinais empregados são tratados por “post-it”, reservando-se, consequentemente, a palavra Kanban ao quadro de tarefas para gerenciar o trabalho como um todo.

Na Figura 1, é apresentado o exemplo de um quadro Kanban utilizado na Technolog. Este por sua vez contém quatro colunas: a primeira coluna (*TEAM*) descreve a equipe de desenvolvimento; a segunda coluna (*DEVELOPMENT*) representa o *status* das atividades de desenvolvimento; a terceira coluna (*TEST*) apresenta o *status* das atividades de testes; por fim, a quarta coluna apresenta o *backlog* e o *sprint backlog* (mais detalhes são apresentados a seguir).

Na primeira coluna, tem-se várias linhas com *avatares*, que são representações gráficas de cada membro do time de desenvolvimento da Technolog. Dentre eles tem desenvolvedores *front-end*, *back-end*, analista de requisitos, dentre outros. Cada linha na segunda coluna, que por vez tem duas sub-colunas, representa as atividades que estão aguardando para serem desenvolvidas (*TO DO*), assim como as que estão sendo desenvolvidas (*DOING*) por determinado membro da equipe.

Figura 1 - Quadro Kanban - Technolog

Fonte: Technolog (2019)

Na terceira coluna, tem-se três sub-colunas, que representam as atividades que estão aguardando para serem testadas (*TO DO*). As atividades que dependem de um fator externo, por exemplo “aprovação do cliente para mudança no escopo da solicitação”, vão para a sub-coluna *NOT OK* e permanecem nesta coluna até que os impedimentos sejam resolvidos. Por fim, as atividades que já foram testadas e que foram liberadas para serem replicadas no ambiente de produção ficam na sub-coluna *UPLOAD*.

Na quarta e última coluna, são apresentados duas sub-colunas, sendo elas: *SPRINT* (uma abreviação de *SPRINT BACKLOG*) e *BACKLOG* (uma abreviação de *PRODUCT BACKLOG*). Na coluna de *Backlog*, é apresentada uma lista de todos os requisitos necessários para se chegar ao produto final. Já na coluna *Sprint*, estão todos requisitos que devem ser desenvolvidos em determinado *Sprint*. O principal responsável pela gestão do quadro é o *Scrum Master*.

O quadro kanban foi utilizado para orientar a equipe sobre quais requisitos precisavam ser implementados, quais estavam em teste ou em desenvolvimento. As cores dos cartões *post-it* são significativas, conforme descrito abaixo:

- **Amarelo** - Portal Posto (sistema desenvolvido pela empresa Technolog para realização de integrações com os postos de combustíveis);
- **Azul** - *Darwin* (sistema desenvolvido com foco no controle de combustível);
- **Laranja** - representa um *bug* que foi encontrado e precisa ser corrigido;
- **Rosa** - *Manager* e *Reports* (o sistema Manager é o responsável pelo gerenciamento dos clientes da Technolog. O *Reports* é responsável pela geração de relatórios customizados para os clientes da empresa);
- **Verde** - representa um requisito do projeto de desenvolvimento de aplicativos móveis.

Além das cores, os *post-it* contam com duas informações de grande importância. Uma é o *número da demanda*, que permite localizar a mesma na ferramenta de gestão de projetos *Plan.io* (Seção 2.6). Nesta ferramenta, há uma descrição mais detalhada do requisito, que é descrito em forma de história do usuário, assim como dos prazos a serem cumpridos e qual cliente aquela atividade está relacionada.

Outra informação relevante é o *nível de prioridade* que o requisito possui, podendo ser classificada como: (i) *alta*: demandas urgentes, com curto prazo de execução, geralmente 2 dias; (ii) *média*: demandas não tão urgentes, mas tem que ser resolvidas o mais rápido possível, geralmente uma semana; e (iii) *baixa*: demandas que podem esperar, pois provavelmente não causam um impacto negativo, ou algum tipo de prejuízo para o cliente.

2.3 API (*Application Programming Interface*)

Com o surgimento da ideia de *web services*, as aplicações computacionais passaram a trocar informações utilizando um formato de troca de informações intermediário, tal como o XML (*eXtensible Markup Language*), JSON (*JavaScript Object Notation*), entre outros. Isso elevou o conceito de reutilização de software a um novo patamar (MOBILE MAGAZINE, 2019).

De acordo com a *World Wide Web Consortium* (W3C)[7], organização internacional que rege os padrões de tecnologia utilizados na Internet, *web service* é

[7] https://www.w3.org/TR/ws-arch/

um sistema de software projetado para oferecer suporte à interação interoperável máquina-a-máquina através de uma rede.

Na prática, um *web service* permite a comunicação entre produtos de software que sejam desenvolvimentos sobre a mesma plataforma ou não, por meio de uma interface descrita em um formato processável por máquina. Um exemplo seria um *web service* para atualização de senha de usuário. Uma vez que este serviço tenha sido definido, duas aplicações diferentes, uma escrita em PHP (*Hypertext Preprocessor*) e outra em Java, por exemplo, podem fazer uso do serviço disponibilizado, sem se preocupar com os detalhes de implementação desse serviço.

Alguns benefícios da utilização de *web services* são (MOBILE MAGAZINE, 2019):

- Utilizar protocolos baseados em texto, o que permite tráfego de dados mais suave;
- Distribuir de código de forma modularizada, facilitando a manutenção e a correção de erros; e
- Possibilitar que dispositivos de diferentes arquiteturas, tais como computadores, *smartphones*, *tablets*, entre outros, consigam interagir e reutilizar serviços e porções de código, possibilitando um uso mais inteligente e eficiente dos recursos computacionais .

Atualmente, uma das arquiteturas mais utilizadas para implementação *web service* é a REST (*Representational State Transfer*). Sucintamente, um *web service* REST é um conjunto de recursos oferecidos por um servidor por meio de chamadas HTTP (*Hypertext Transfer Protocol*), que podem ser realizadas através dos métodos GET, POST, PUT e DELETE. O Quadro 1 mostra uma requisição de *login* do tipo GET, com dados enviados no formato JSON, especificados entre chaves.

Quadro 1 - Requisição para um *web service* REST

```
http://endereco.servidor.com.br/diretorio/subdiretorio/index.php?acao=login/logar&usuario={"login":"valdeci","senha":"abc123"}
```

Fonte: Technolog (2019)

Após a requisição ser efetuada, o servidor irá verificar se os dados passados (usuário e senha) são válidos. Caso sejam válidos, ele irá retornar um *token* de acesso para que o usuário não precise informar usuário e senha a cada nova requisição ao servidor. Contudo se os dados forem inválidos, será retornado um objeto JSON contendo mensagens que orientem o usuário a se recuperar de falha no processo de autenticação.

Umas das funções desempenhadas durante este estágio foi desenvolver novos *web services* no sistema *Darwin*, que são consumidos pelos aplicativos móveis oferecidos pela empresa Technolog aos seus clientes.

2.4 Tecnologias utilizadas

Nesta seção, são apresentadas e descritas algumas das linguagens e *frameworks* utilizados no desenvolvimento do sistema *Web Darwin*, durante o período de estágio na empresa Technolog.

2.4.1 HTML (*Hypertext Markup Language*)

HTML é uma linguagem com a qual se definem páginas web. Basicamente, trata-se de um conjunto de etiquetas (*tags*) que servem para definir a forma na qual se apresentará o texto e outros elementos da página. O Quadro 2 apresenta a estrutura básica para criação de uma página web, utilizando a linguagem de marcação de hipertexto em sua versão 5 (HTML 5) (ALVAREZ, 2018).

A primeira linha do Quadro 2 apresenta uma *tag* denominada DOCTYPE, a qual comunica aos navegadores (*browsers*), e leitores de tela, qual o tipo de documento que eles estão prestes a carregar e exibir. Após, tem-se a *tag* HTML. Isso quer dizer que tudo o que estiver entre essa *tag* e sua respectiva *tag* de fechamento (</html>) é escrito em linguagem HTML. Ao lado da palavra HTML, tem-se o atributo denominado *lang*, por meio do qual é indicado qual o idioma do texto contido na página web. Logo após a *tag* HTML, é apresentada a *tag head*, dentro da qual é indicado o título do documento (*tag title*) e também a tabela de caracteres que o *browser* deve usar para renderizar o texto (*tag meta* com a

propriedade *charset*).

Quadro 2 - Estrutura básica de uma página HTML

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <head>
4          <meta charset="utf-8">
5          <title>Título da página</title>
5      </head>
7      <body>
8       ... aqui vai todo o código HTML ...
9      </body>
10 </html>
```

Fonte: Autor

A linguagem HTML foi utilizada no estágio para inserção dos elementos que interagem com os usuários dos sistemas mantidos pela Technolog. Tais como parágrafos, imagens, caixas para entrada e saída de dados, dentre outros. Assim como, para organizá-los, de tal forma a informar o que é um título principal, título secundário, título da página, disposição de linhas e colunas das tabelas apresentadas nas visões, dentre outras *tags* disponíveis.

2.4.2 JavaScript

JavaScript, frequentemente abreviada como JS, é uma linguagem leve e portável, sendo suportada por praticamente todos os navegadores. JS é uma linguagem de programação interpretada de alto nível, caracterizada também como dinâmica, fracamente tipificada, que permite injetar lógica de programação em páginas escritas em HTML (LIMA, 2006).

O Quadro 3 mostra a inserção de uma lógica para exibição da *string* “Hello World!” em uma página HTML, página esta utilizada para que o usuário interaja com sistema.

O trecho de código capaz de executar tal lógica é descrito entre a *tag* de abertura <*script*> e a *tag* de fechamento </*script*>.

Quadro 3 - Exemplo de uso da linguagem JS

```
1 <!DOCTYPE html>
2 <html>
3      <head>
4            <title>Hello Word</title>
5      </head>
6      <body>
7            <script type="text/javascript">
8                  alert("Hello World!");
9            </script>
10     </body>
11 </html>
```

Fonte: Autor

Outro exemplo de utilização da linguagem JS é apresentado no Quadro 4. Assim como o trecho de código apresentado no Quadro 3, este código também exibe a mensagem “Hello World!”, porém, não diretamente após o carregamento da página. Neste caso, a mensagem aparecerá após o usuário clicar sobre o botão *“Exibir Alert”*, apresentado na linha 13 do mesmo código.

Quadro 4 - Função JS acionada por interação com o usuário

```
1 <!DOCTYPE html>
2 <html>
3      <head>
4            <title>Hello Word</title>
5            <script type="text/javascript">
6            function helloWord()
7            {
8                  alert("Hello Word!");
9            }
10           </script>
11     </head>
12     <body>
13           <input type="button" onclick="helloWord()" value="Exibir Alert" />
14     </body>
15 </html>
```

Fonte: Autor

A linguagem JS foi utilizada no estágio para a validação de formulários de páginas *web*, a fim de garantir que os dados informados pelos usuários estejam de acordo o esperado. A validação de dados com JS é feita no *browser* (ou seja, na

máquina do usuário) e garante a consistência dos dados, antes de eles serem enviados para o servidor e, posteriormente, para a base de dados.

2.4.3 CSS (*Cascading Style Sheets*)

CSS (*Cascading Style Sheets*) é uma "folha de estilo" composta por "camadas" e utilizada para definir a apresentação (aparência) de páginas web (CSS - Curso W3C Escritório Brasil, 2019). Uma das grandes vantagens do CSS é efetuar a separação entre o formato e o conteúdo de um documento (PEREIRA, 2009). O Quadro 5 apresenta uma página web, a qual foi formatada com os estilos definidos entre as linhas 4 e 16.

Quadro 5 - Formatação da página utilizando CSS

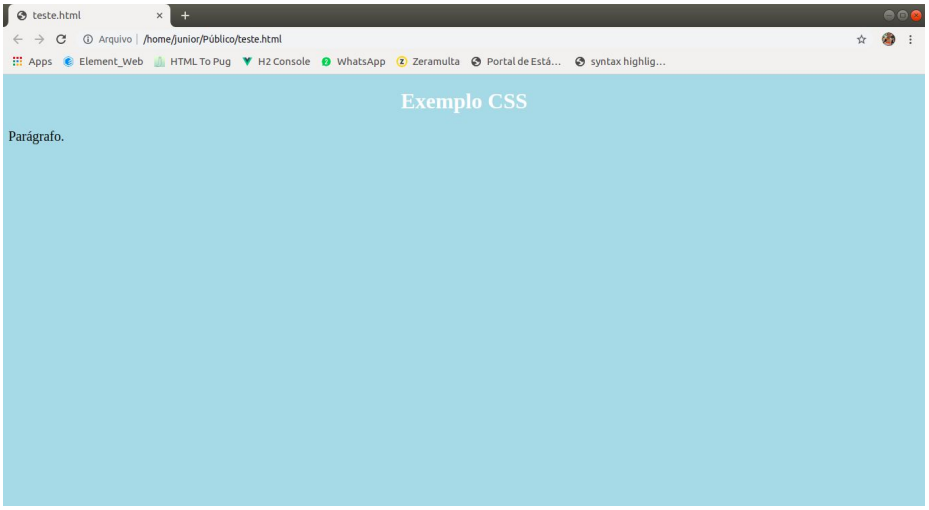
```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <style>
5             body {
6               background-color: lightblue;
7             }
8             h1 {
9               color: white;
10              text-align: center;
11            }
12            p {
13              font-family: verdana;
14              font-size: 20px;
15            }
16        </style>
17    </head>
18    <body>
19        <h1>Exemplo CSS</h1>
20        <p>Parágrafo.</p>
21    </body>
22 </html>
```

Fonte: Autor

Na linha 19, tem-se o cabeçalho da página (*tag h1*) e na linha 20, um parágrafo (*tag p*). Ambos os elementos são afetados pelo estilo descrito nas linhas 5 a 7, pois ele é aplicado sobre todos os elementos que estão entre as *tags* <*body*> e </*body*>, do arquivo HTML. Entre as linhas 8 e 11, encontram-se os estilos a serem

aplicados apenas sobre os dados que são restringidos pelas *tags* de abertura e fechamento de um cabeçalho, respectivamente, *<h1>* e *</h1>*. No exemplo em questão, conforme apresentado na Figura 2, todo cabeçalho aparece na cor branca e centralizado. Nas linhas 12 a 15, estão as formatações referentes aos conteúdos localizado entre as *tags* de parágrafo * * e * *, que são o tamanho da fonte 20 e tipo da fonte “verdana”.

Figura 2 - Exemplo de página web com formatação



Fonte: Autor

CSS foi utilizado neste estágio para realizar a estilização de algumas páginas web dos sistemas da empresa. O objetivo era fazer com que essas páginas apresentassem a identidade visual da empresa Technolog. Mais detalhes de como o CSS foi utilizado podem ser vistos na Seção 4.

2.4.4 PHP (*Hypertext Preprocessor*)

PHP é uma linguagem de programação desenvolvida nos anos 1994 por Rasmus Lerdorf, que inicialmente a utilizava em sua *home page* pessoal. Em meados de 1995, ela passou a ser utilizada por outras pessoas e foi reescrita com novos recursos, sendo renomeada para *Personal Home Page Tools/FI (FI = Form Interpreter)*. Dois anos mais tarde, PHP deixou de ser um projeto pessoal de Rasmus Lerdorf e passou a ser desenvolvida por uma equipe de colaboradores e,

neste período, foi lançada a versão 3 da linguagem (ARRAYO, 2003).

PHP (atualmente é considerado um acrônimo recursivo para “PHP: *Hypertext Preprocessor*”) é uma linguagem de programação interpretada, amplamente utilizada para o desenvolvimento de aplicações web dinâmicas, isto é, capazes de gerar conteúdo para a web em tempo de execução. Para isso, o código PHP é interpretado no lado do servidor e é gerado um código HTML, a ser visualizado no lado do cliente, por meio do *browser*. Algumas vantagens da utilização de PHP são (ARRAYO, 2003):

- É uma linguagem de fácil aprendizado;
- Seu código é aberto;
- Tem suporte nos principais servidores web do mercado, tais como Apache, NGINX;
- Suporta conexão com os sistemas gerenciadores de bancos de dados mais utilizados do mercado, como por exemplo, MySQL, PostgreSQL e Oracle;
- É multiplataforma, sendo suportada nos sistemas operacionais mais utilizados no mercado; entre outros.

O Quadro 6 apresenta estrutura básica de um código PHP, o qual fica compreendido entre tags *“<?php”* e *“?>”* (linhas 6 até 8). O exemplo em questão imprime a frase “Hello World” na tela do usuário.

Quadro 6 - Exemplos de código escrito em PHP

```
1 <html>
2 <head>
3  <title>Teste PHP</title>
4 </head>
5 <body>
6      <?php
7             echo "<p>Hello World</p>";
8      ?>
9 </body>
10</html>
```

Fonte: Autor

No estágio realizado na empresa Technolog, a linguagem PHP foi utilizada para o desenvolvimento da funcionalidade dos sistemas *Darwin* e *Manager*, bem como dos *web services* a serem consumidos pelos aplicativos móveis

disponibilizados pela empresa.

2.4.5 JQuery

JQuery é uma biblioteca JS criada para simplificar a criação de efeitos visuais e de interatividade em páginas web (SILVA, 2008). JQuery é utilizada por cerca de 55% dos 10 mil sites mais visitados do mundo, de acordo com o site “Linha de código”[8]. Algumas das vantagens da biblioteca JQuery são (Murphey, 2010):

- Resolução de incompatibilidades entre os navegadores;
- Redução da quantidade de linhas de código para especificação de um comportamento em JS;
- Reutilização de código, por meio de *plugins* e funções utilitárias; entre outros.

O Quadro 7 apresenta um exemplo de código capaz de imprimir a mensagem “Hello World” na tela do usuário.

Quadro 7 - Exemplos de uso da biblioteca JQuery

```
1 <html>
2       <head>
3             <title>jQuery Hello World</title>
4             <script
5 src="//ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"></script>
6       </head>
8 <body>
9             <script type="text/javascript">
10                  $(document).ready(function(){
11                        $("#msgid").html("Hello World");
12                  });
13            </script>
14            <div id="msgid">
15            </div>
16      </body>
17 </html>
```

Fonte: Autor

As linhas que merecem destaque são: 4 a 6, pois realizam a importação da biblioteca JQuery; 9 a 13, que criam a estrutura JS dentro do HTML. Nas linhas 10 a

[8] http://www.linhadecodigo.com.br/

12, ocorre o uso, propriamente dito, do JQuery. Na linha 10 é apresentada a sintaxe de uso do JQuery no pré-carregamento da página, criando assim a função que irá configurar a mensagem “Hello World” no elemento HTML apresentado na linha 14, que é uma *tag div* com *id ‘msgid*’.

Na linha 11, a mensagem é configurada no elemento HTML, de tal forma que quando a página for renderizada a tag *div* terá contido como seu valor a mensagem “Hello Word”. As demais linhas são compostas por elementos HTML que já foram apresentados nas seções anteriores.

Outro exemplo de uso desta biblioteca é para a criação de máscaras de entrada de dados. Estas máscaras correspondem a formatações que o usuário precisa respeitar para que os dados informados sejam aceitos. Esse tipo de recurso é bastante utilizado, por exemplo, em campos de placa veicular, CPF (Cadastro de Pessoa Física), CNPJ (Cadastro Nacional de Pessoa Jurídica), dentre outros.

2.4.6 AJAX (*Asynchronous JavaScript and XML*)

AJAX é um acrônimo de *Asynchronous JavaScript and XML* e refere-se a uma técnica de desenvolvimento web, que permite a criação de aplicações mais interativas. Um dos principais objetivos deste modelo é tornar as respostas das páginas web mais rápidas pela troca de pequenas quantidades de dados com o servidor web, de forma assíncrona. Em uma chamada assíncrona, o código que a chamou determinada função continua sua execução, sem esperar pela resposta. Quando o servidor responde, uma função JS especificada pelo programador (conhecida como função *callback*) trata os dados retornados, fazendo a atualização da da tela[9].

No Quadro 8, é apresentada a forma na qual AJAX foi utilizado durante o período do estágio. O exemplo em questão é utilizado para o envio de formulários HTML para o sistema *Darwin*. A linha 1 do Quadro 8 é responsável por selecionar o formulário, via JQuery. A partir da linha 6 até a linha 39, são apresentados os recursos que o Ajax oferece. Da linha 7 até a linha 12 está compreendido o cabeçalho da requisição, na qual tem-se informações sobre a URL para a qual será

[9] https://www.devmedia.com.br/ajax-tutorial/24797

feita a requisição, o método que será utilizado (POST, GET, entre outros), o tipo de dados (JSON, XML, entre outros), entre outras informações.

Quadro 8 - Envio de formulário HTML com AJAX

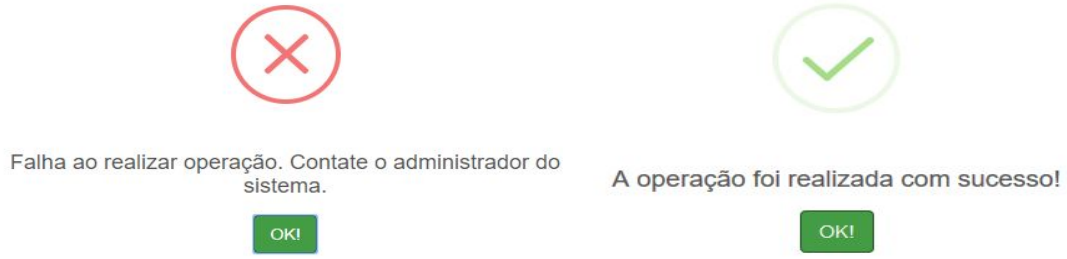
```
1 var formulario = $("#formAbastecimentoSemDarwin");
2 var url = formulario.attr("action").trim();
3 var dados = new FormData(formulario[0]);
4 var tipo = formulario.attr("method").trim();
5 var tipoDados = ‘json’;
6 $.ajax({
7     url: url,
8     type: tipo,
9     data: dados,
10    dataType: tipoDados,
11    processData:false,
12    contentType:false,
13    success:function (res) {
14          swal({
15                text: res.retorno,
16                type: res.status,
17                confirmButtonColor: '#3085d6',
18                confirmButtonText: 'OK!',
19                confirmButtonClass: 'btn btn-success',
20                buttonsStyling: false,
21                reverseButtons: true
22          }).then((result) => {
23                if (res.status) {
24                      window.location.reload()
25                }
26          });
27    },
28    error:function () {
29          swal({
30                text: 'Erro, contate o administrador do sistema.',
31                type: 'error',
32                confirmButtonColor: '#3085d6',
33                confirmButtonText: 'OK!',
34                confirmButtonClass: 'btn btn-success',
35                buttonsStyling: false,
36                reverseButtons: true
37          })
38    } //Erro no envio da requisição.
39 });
```

Fonte: Technolog (2019)

As linhas 13 e 28 descrevem funções que irão tratar, respectivamente, os casos em que o envio dos dados ao servidor foi efetuado corretamente ou não. Caso os dados sejam enviados sem falhas, a linha 14 exibe uma mensagem, informando o

status do servidor com relação à operação (Figura 3 - lado esquerdo). Caso os dados não sejam enviados corretamente para o servidor, é exibido um alerta (linha 29), informando que houve uma falha (Figura 3 - lado direito).

Figura 3 - Retorno de requisição feito com AJAX



Fonte: Technolog (2019).

2.5 Banco de dados

Banco de dados é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico (NAVATHE, 2011). Bancos de dados podem ser classificados, principalmente, em duas categorias, sendo elas: relacionais e não relacionais. Tendo em vista que o modelo de banco de dados utilizados no estágio foi o relacional será apresentado somente este modelo.

Em um banco de dados relacional, todos os dados são modelados de uma forma que eles sejam percebidos pelo usuário como tabelas (relações). SQL (*Structured Query Language*) é a linguagem de pesquisa declarativa padrão para banco de dados relacional. Muitas das características originais do SQL foram inspiradas na álgebra relacional. Esta linguagem se tornou um padrão para criação, acesso e manutenção de base de dados. Isto decorre, em partes, devido a sua simplicidade e facilidade de uso. Ela se diferencia de outras linguagens de consulta a banco de dados, no sentido em que uma consulta SQL especifica-se a forma do resultado e não o caminho para chegar a ele (NAVATHE, 2011).

No Quadro 9, é exemplificado o uso da linguagem SQL. Esta consulta por vez, retorna o nome, sobrenome e endereço dos funcionários do departamento de “Pesquisa”.

Quadro 9 - Exemplo de consulta SQL

```
1   SELECT
2           F.Pnome, F.Unome, F.Endereço
3   FROM
4           FUNCIONARIO F, DEPARTAMENTO D
5   WHERE
6           D.DNome='Pesquisa'
7           AND D.Dnumero=F.Dnr;
```

Fonte: Página 67, NAVATHE, 2011

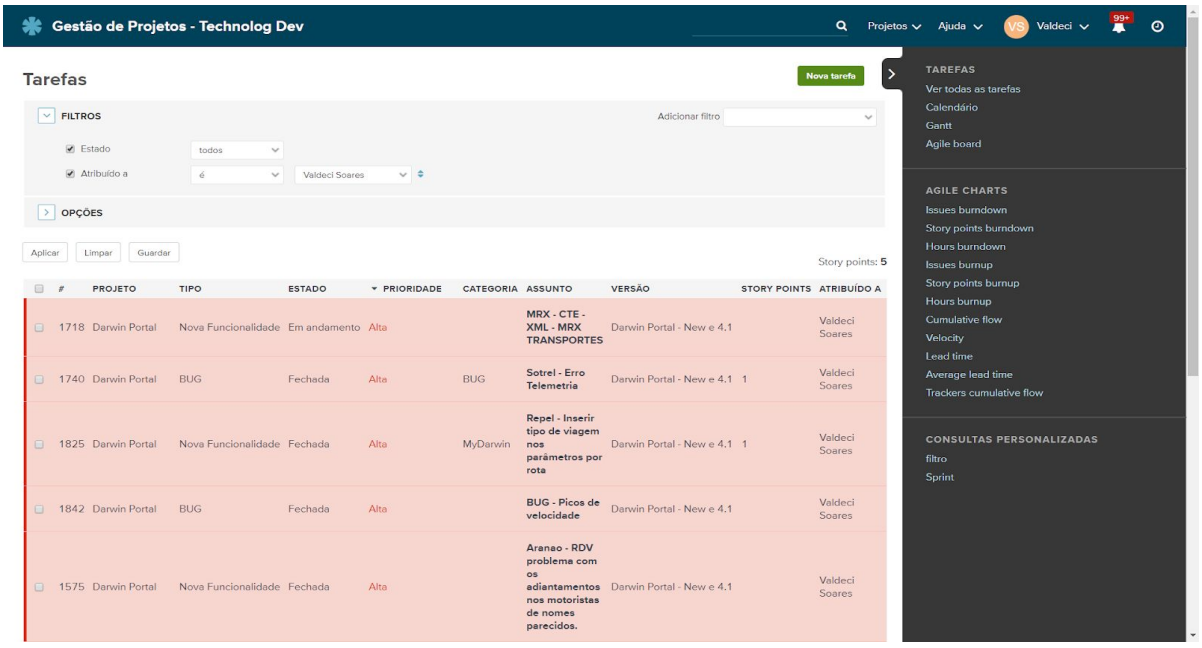
A linguagem SQL foi utilizada neste estágio para realização das operações de inserção, atualização, leitura e exclusão dos dados armazenados nos bancos de dados de todos os sistemas mantidos neste estágio. Nas atividades de manutenção dos sistemas *Manager* e *Darwin* as consultas existentes são aproveitadas e é apenas realizado os ajustes necessários. Contudo, no desenvolvimento de novas funcionalidades, novas consultas foram criadas.

2.6 Gestão de Projetos Plan.io

Para implementar as solicitações de demandas que surgem no suporte ao usuário da empresa Technolog, é preciso acessar o sistema de gestão de projetos Plan.io (Figura 4). O mesmo é utilizado para o gerenciamento de tarefas, integração de ferramentas de planejamento e manipulação intuitiva do fluxo de trabalho. Nele, estão as tarefas a serem realizadas, juntamente com suas propriedades, como *status*, prioridade, responsável pela tarefa, categoria, datas de início e conclusão e entre outros.

Este aplicativo serve de apoio à documentação dos sistemas mantidos pela empresa, possibilitando rastrear as atividades desenvolvidas e as alterações efetuadas e quais os responsáveis por elas. Um exemplo de uso do Plan.io na Technolog é quanto às alterações efetuadas no banco de dados de algum sistema. Ao descrever tais alterações no Plan.io, os demais desenvolvedores podem saber o motivo pelo qual o banco de dados sofreu determinada mudança, quais impactos as alterações provocarão em funcionalidade já desenvolvida, entre outras coisas.

Figura 4 - Gestão de Projetos Plan.lo



Fonte: Technolog (2019)

3 ATIVIDADES REALIZADAS

Neste capítulo, são descritas algumas das atividades realizadas durante o período de estágio. A maior parte das atividades desenvolvidas foi para os sistema *Darwin*, uma vez que, dentre os softwares desenvolvidos pela Technolog, ele é o mais utilizado. Contudo, também foram desenvolvidas atividades no sistema *Manager*, assim como em algumas APIs que integram o *Darwin* com aplicativos móveis.

3.1 Descrição das atividades desenvolvidas no *Darwin*

Nesta seção, são descritas as atividades que foram desenvolvidas no sistema *Darwin*, assim como as tecnologias utilizadas, as dificuldades encontradas e as lições aprendidas. No *Darwin*, é possível realizar as diversas atividades que um software CRM se propõe a fazer, porém de uma forma customizada para cada transportadora (empresas clientes da Technolog).

3.1.1 Lançamento de multas

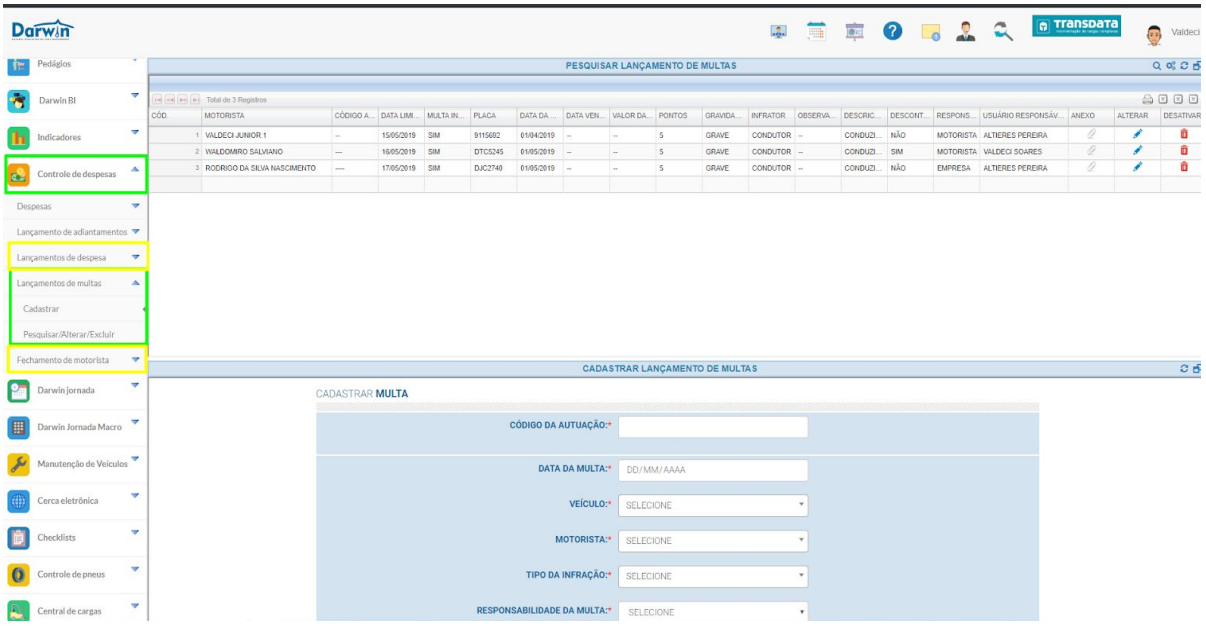
Dentre os diversos problemas que foram resolvidos neste estágio, a **gestão de multas dos motoristas** é um deles. A funcionalidade de gestão de multas é importante, pois a transportadora não pode arcar com os custos que são de responsabilidade dos motoristas. Assim, as multas devem ser descontadas do salário dos motoristas, para que, em momento oportuno, a empresa realize a quitação dos débitos dos veículos junto ao departamento de trânsito.

Neste estágio, foi criado um submenu, denominado *Lançamento de multas*, dentro do menu controle de despesas, já existente (Figura 5). Esta funcionalidade oferece as opções de cadastro, pesquisa, edição e exclusão. A tabela criada para exibição das multas consta com um filtro, que auxilia o usuário a buscar as multas lançadas no sistema *Darwin*. Este filtro possui opções de busca por motorista, pelo código da autuação, por veículo, por período de datas, entre outras.

A exclusão de uma multa é apenas uma desabilitação do registro na base de dados, pois, caso o usuário venha excluir uma multa por engano, é possível que o

setor de TI possa recuperar o registro da multa.

Figura 5 - lançamento de multas no *Darwin*



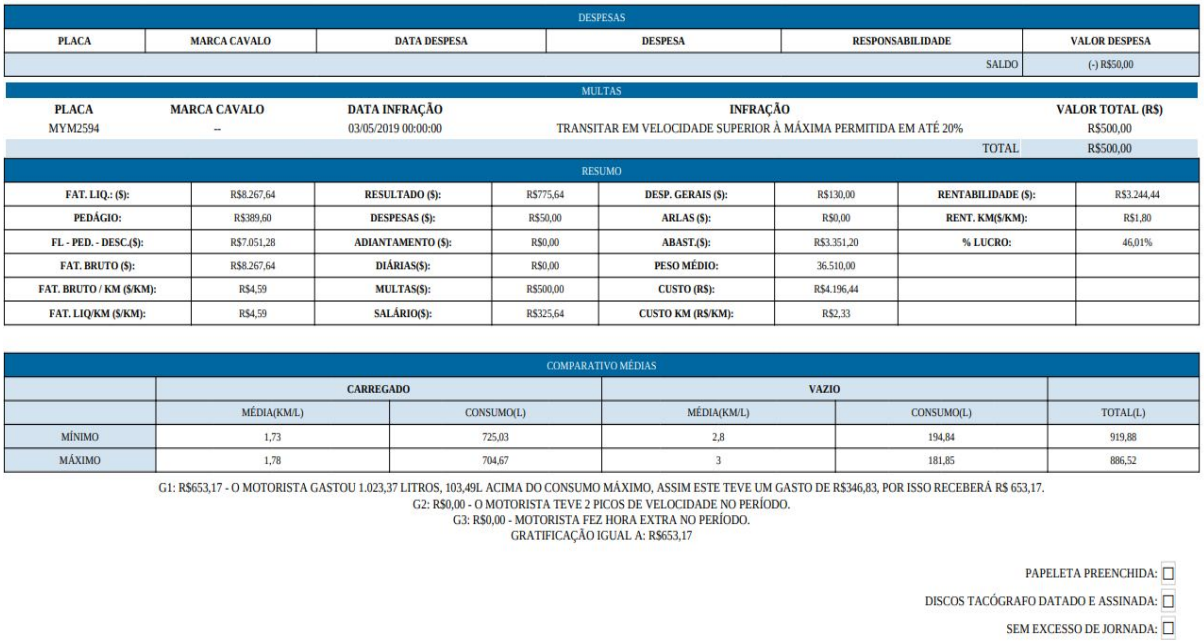
Fonte: Technolog (2019)

Os dados necessários para cadastro de uma multa são: nome do motorista e a placa do veículo autuado, data na qual a infração foi cometida, o tipo e a gravidade da infração, a pontuação gerada, o valor e a validade do boleto e, por fim, o arquivo digital (escaneado) do documento da infração enviado pelo departamento de trânsito.

Esta nova funcionalidade gerou, por consequência, uma alteração no relatório de fechamento do motorista, que é uma espécie de “folha de pagamento” o qual pode ser visualizado na Figura 6. Caso o motorista tenha cometido alguma infração de trânsito, a mesma deverá ser apresentada no campo “Multas”, deste relatório. Além disso, todo o cálculo necessário deve ser realizado para que o valor da infração seja abatido do valor total ao qual o motorista tenha para receber.

Contudo a empresa permite que o pagamento seja parcelado, logo deve ser descontado naquele fechamento, somente o valor proporcional a quantidade de parcelas que o motorista acordou com a empresa.

Figura 6 - Relatório de fechamento de motorista



DESPESAS

PLACA	MARCA CAVALO	DATA DESPESA	DESPESA	RESPONSABILIDADE	VALOR DESPESA
				SALDO	(-) R$50,00

MULTAS

PLACA	MARCA CAVALO	DATA INFRAÇÃO	INFRAÇÃO	VALOR TOTAL (R$)
MYM2594	--	03/05/2019 00:00:00	TRANSITAR EM VELOCIDADE SUPERIOR À MÁXIMA PERMITIDA EM ATÉ 20%	R$500,00
			TOTAL	R$500,00

RESUMO

FAT. LIQ.: ($):	R$8.267,64	RESULTADO ($):	R$775,64	DESP. GERAIS ($):	R$130,00	RENTABILIDADE ($):	R$3.244,44
PEDÁGIO:	R$389,60	DESPESAS ($):	R$50,00	ARLAS ($):	R$0,00	RENT. KM($/KM):	R$1,80
FL - PED. - DESC.($):	R$7.051,28	ADIANTAMENTO ($):	R$0,00	ABAST.($):	R$3.351,20	% LUCRO:	46,01%
FAT. BRUTO ($):	R$8.267,64	DIÁRIAS($):	R$0,00	PESO MÉDIO:	36.510,00		
FAT. BRUTO / KM ($/KM):	R$4,59	MULTAS($):	R$500,00	CUSTO (R$):	R$4.196,44		
FAT. LIQ/KM ($/KM):	R$4,59	SALÁRIO($):	R$325,64	CUSTO KM (R$/KM):	R$2,33		

COMPARATIVO MÉDIAS

	CARREGADO		VAZIO		
	MÉDIA(KM/L)	CONSUMO(L)	MÉDIA(KM/L)	CONSUMO(L)	TOTAL(L)
MÍNIMO	1,73	725,03	2,8	194,84	919,88
MÁXIMO	1,78	704,67	3	181,85	886,52

G1: R$653,17 - O MOTORISTA GASTOU 1.023,37 LITROS, 103,49L ACIMA DO CONSUMO MÁXIMO, ASSIM ESTE TEVE UM GASTO DE R$346,83, POR ISSO RECEBERÁ R$ 653,17.
G2: R$0,00 - O MOTORISTA TEVE 2 PICOS DE VELOCIDADE NO PERÍODO.
G3: R$0,00 - MOTORISTA FEZ HORA EXTRA NO PERÍODO.
GRATIFICAÇÃO IGUAL A: R$653,17

PAPELETA PREENCHIDA: ☐
DISCOS TACÓGRAFO DATADO E ASSINADA: ☐
SEM EXCESSO DE JORNADA: ☐

Fonte: Technolog (2019)

Esta foi uma das primeiras atividades desenvolvidas durante o tempo de estágio e, assim sendo, a principal dificuldade consistiu na ausência de conhecimento das tecnologias utilizadas. O maior desafio foi aprender a utilizar as técnicas de depuração de código nas linguagens PHP e JS. Um dos conhecimentos novos adquirido foi sobre a função *var_dump*, da linguagem PHP, que mostra informações sobre determinada variável passada como parâmetro para a função, incluindo o tipo e o valor dos dados. Outro exemplo de conhecimento adquirido, foi a respeito da função *console.log*, semelhante a *var_dump*, porém para ser utilizada na linguagem JS.

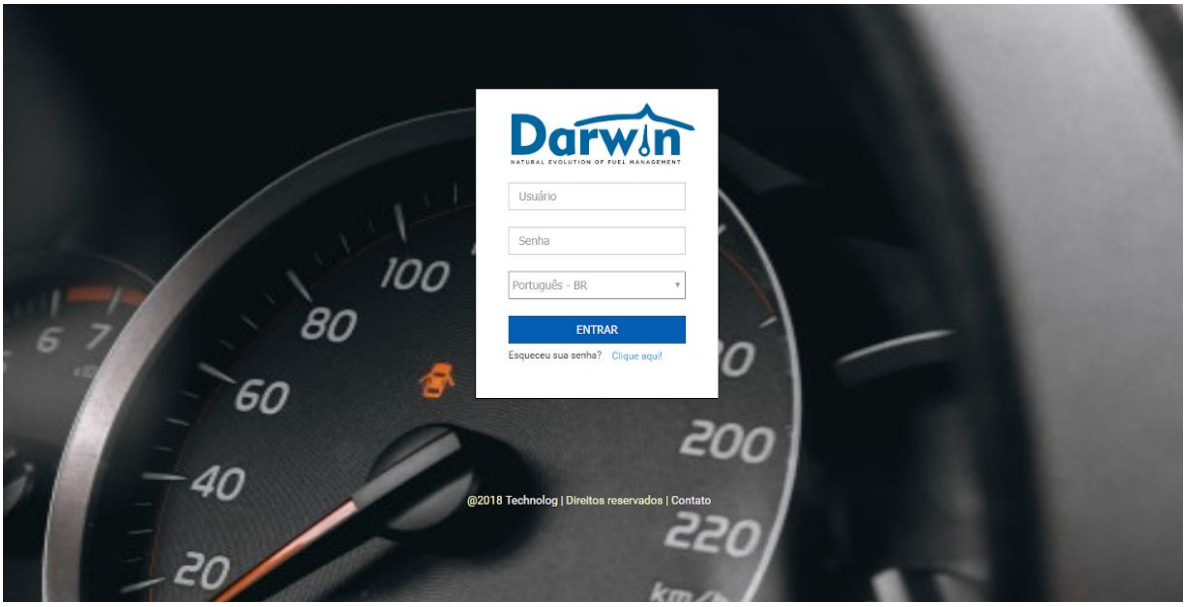
3.1.2 Traduções para as línguas Inglesa e Hispânica

A Figura 7 apresenta a tela de *login* do sistema *Darwin*. Além de permitir a autenticação, essa tela permite que o usuário escolha uma entre as linguagens disponíveis para o sistema, que são: Português, Inglês e Espanhol.

Por padrão, o idioma selecionado é o Português, contudo, ao clicar sobre a caixa de seleção, é aberta uma lista contendo todos os idiomas disponíveis. Após selecionado o idioma, o mesmo é salvo em uma variável de sessão da linguagem

PHP. Este é um recurso que permite salvar valores a serem usados ao longo da interação do usuário com o sistema. Valores salvos na sessão podem ser usados em qualquer parte do programa. Assim, cada módulo do sistema pode requisitar o valor contido nessa variável e interpretá-lo. Vale lembrar que estas variáveis permanecem configuradas até o visitante fechar o *browser* ou a sessão seja destruída.

Figura 7 - Tela login portal *Darwin*



Fonte: Technolog (2019)

No Quadro 10 é apresentado um trecho do código fonte utilizado para realização da tradução do sistema.

Na linha 2, é iniciada a sessão do usuário. Logo após, a variável de sessão é passada para uma variável local (linha 3), para que então, essa variável de sessão possa ser fechada e não cause nenhum tipo de bloqueio. Entre as linhas 6 e 16, tem-se um comando *switch-case*, no qual é realizado a leitura do valor contido na variável *$langId*. Os valores esperados por essa variável são 1, 2 e 3. Caso nesta variável esteja contido o valor 1, será adotado para a variável *$msnRet* o título em português, para o valor 2 será adotado o título em inglês e para o valor 3, em espanhol.

A partir da linha 17 é apresentado o código HTML que exibirá a mensagem,

de acordo com o idioma escolhido.

Quadro 10 - Tradução do Sistema

```
<?php
    session_start();
    $langId = $_SESSION["idioma"];
    session_write_close();

    switch ($langId) {
          case 1: //PORTUGUÊS
                $msnRet = "Processando, por favor aguarde";
                break;
          case 2: //Inglês
                $msnRet = "Processing, please wait";
                break;
          case 3: //ESPANHOL
                $msnRet = "Procesamiento, por favor espere";
                break;
    }
?>
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="utf-8">
        <title>Título da página</title>
    </head>
    <body>
     <?php echo $msnRet; ?>
    </body>
</html>
```

Fonte: Autor

Para esta tarefa, utilizou-se apenas as tecnologias PHP e HTML. A principal dificuldade consistiu na quantidade de elementos visuais que as interfaces gráficas possuíam, aumentando assim, o número de elementos que precisavam ser ajustados para contemplar as traduções.

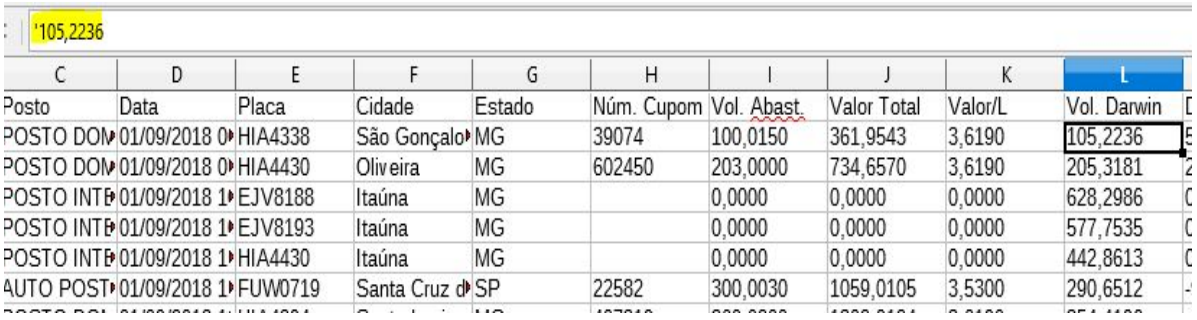
3.1.3 Correção na abertura de planilhas eletrônicas

Outro problema tratado neste estágio, consistia em resultados inadequados das células das planilhas eletrônicas que contém valores numéricos, extraídas a partir do sistema *Darwin*.

Como pode ser visto na Figura 8, essas células apresentam o caractere "apóstrofo" ('), o que interferia na utilização de funções automáticas da planilha, uma

vez que a mesma não era interpretada como valor numeral.

Figura 8 - Exemplo de problema com planilhas eletrônicas extraídas do Darwin



'105,2236

C	D	E	F	G	H	I	J	K	L
Posto	Data	Placa	Cidade	Estado	Núm. Cupom	Vol. Abast.	Valor Total	Valor/L	Vol. Darwin
POSTO DON	01/09/2018 0	HIA4338	São Gonçalo	MG	39074	100,0150	361,9543	3,6190	105,2236
POSTO DON	01/09/2018 0	HIA4430	Oliveira	MG	602450	203,0000	734,6570	3,6190	205,3181
POSTO INTE	01/09/2018 1	EJV8188	Itaúna	MG		0,0000	0,0000	0,0000	628,2986
POSTO INTE	01/09/2018 1	EJV8193	Itaúna	MG		0,0000	0,0000	0,0000	577,7535
POSTO INTE	01/09/2018 1	HIA4430	Itaúna	MG		0,0000	0,0000	0,0000	442,8613
AUTO POST	01/09/2018 1	FUW0719	Santa Cruz d	SP	22582	300,0030	1059,0105	3,5300	290,6512

Fonte: Technolog 2019

Contudo, este problema se tornou um grande problema, pois foram gastos quatro dias úteis de trabalho para resolvê-lo. Esta demora ocorreu, pois toda exportação de arquivos era feita por meio de um código JS interno de cada página. Tratava-se, assim, de um problema de duplicação de código, pois havia uma cópia da função para cada local onde era possível realizar essa exportação.

Logo, além do problema com os arquivos de planilha eletrônica, também foi resolvido o problema de código duplicado. Para isso, foi criado um arquivo, denominado *pager_gerar_arquivos.js*, que continha a lógica para exportação de arquivos. Assim, foi possível substituir os códigos duplicados em todos os arquivos, pela chamada da função function *output(type)*, pertencente a esse arquivo. Nesta função, o parâmetro *type* refere-se ao formato do arquivo a ser baixado (por exemplo, 'ods' para arquivos do LibreOffice e Google Sheets, 'xls' para arquivos do Excel 2007 ou anterior e 'xlsx', para arquivos do Excel superior ao 2007). Para realizar a chamada da função, é necessário apenas realizar a importação do arquivo JS, como é mostrado no Quadro 11.

Quadro 11 - Importação do arquivo pager_gerar_arquivo.js

```
1 <script
2     language="JavaScript" src="js/pager_gerar_arquivos.js">
3 </script>
```

Fonte: Autor

Esta solução impactou positivamente também em outra demanda, a qual consistia em não utilizar mais o software Excel, uma vez que ele pertencente ao à empresa Microsoft, que cobra pela licença de uso. A solução encontrada foi a utilização de software gratuito, tal como o *Google Sheets*, porém as planilhas geradas pelo Darwin não podiam ser carregadas para plataforma do Google A solução para esse problema consistiu em adicionar em todo o sistema *Darwin*, as opções 'xlsx' e 'odt', formatos que são lidos pelo software do Google.

Teoricamente, a previsão era que a resolução desta demanda também durasse cerca de quatro dias, como ocorreu com a manutenção citada anteriormente (solução para o problema do apóstrofo nas planilhas). Porém, com a resolução do problema de código duplicado, essa demanda teve um tempo de duração de aproximadamente de três horas. Essa redução no tempo de desenvolvimento da atividade aconteceu porque que todas as alterações foram realizadas apenas no arquivo *pager_gerar_arquivos.js*, ao invés de visitar todos os arquivos de todas as páginas em que havia exportação de planilhas eletrônicas.

Apesar dessa demanda só ter exigido conhecimentos a respeito das tecnologias HTML, JS e PHP, ela proporcionou colocar em prática os conhecimentos de boas práticas de programação, adquiridos em sala de aula.

3.1.4 *Darwin* Jornada Macro

A Lei no 13.103/2015, de Março de 2015, conhecida como Lei do Motorista[10], veio para garantir os direitos do exercício da profissão de motorista. A lei, em questão, estabelece novas normas para regulamentar a rotina de trabalho dos motoristas profissionais de passageiros e de transporte de cargas. As principais mudanças apresentadas na nova legislação abrangem desde jornada de trabalho, período de descanso e espera, até questões como a exigência dos profissionais se submeterem a exames toxicológicos e a responsabilização do contratante do frete em caso de discordância entre o conteúdo transportado e a nota fiscal.

Com base nesse novo cenário é que a funcionalidade "*Darwin* Jornada

[10] http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13103.htm

Macro” foi desenvolvida, com a finalidade de controlar, de forma preventiva e corretiva, a jornada de trabalho dos motoristas, atendendo às exigências da lei.

Com essa nova funcionalidade, os clientes da Technolog podem configurar, com flexibilidade, os parâmetros de controle, aferição dos tempos e *status* da jornada de trabalho, controle de tempos de parada (refeição, repouso, descanso e espera), direção contínua, hora extra, entre outros. A Figura 9 apresenta a tela do submenu “*Darwin* Contexto de Macro”.

Figura 9 - Contexto de Macro

Fonte: Technolog (2019)

O submenu “*Darwin* Contexto de Macro” é pertencente ao menu “Jornada Macro” e contém quatro abas de navegação, a saber:

- **Dados Gerais** - apresentada configurações básicas da macro (macros são as atividades que o motorista pode desempenhar durante uma viagem, como por exemplo um abastecimento, uma parada, carregar e descarregar, dentre outras).

- **Parâmetros de Macro** - permite ao usuário escolher quais as macros serão utilizadas, com suas respectivas configurações, tais como tempo mínimo, tempo máximo, mensagens de alerta, se irá ou não emitir um alerta no aplicativo móvel e se ela depende de outra macro.

- **Motoristas** - permite a inserção de motoristas em uma macro. É permitida a criação de contextos diferentes, com macros diferentes para cada motorista, ou grupo de motoristas.

- **Alertas** - oferece a opção de enviar o alerta para um email previamente cadastrado no sistema.

Essa funcionalidade foi desenvolvida em parceria com a equipe de desenvolvimento do aplicativo móvel. Para que o sistema funcionasse conforme o esperado, os desenvolvedores deveriam estar em sinergia, para que alterações efetuadas em um sistema não impactassem de maneira negativa no outro. Para isso, foi estabelecida a seguinte convenção entre os desenvolvedores dos dois projetos: tudo relacionado a tempo deveria ser salvo em minutos na base de dados, de forma que cada sistema trabalharia no cálculo de conversão de tempo para exibição correta, seja em segundos, minutos ou horas. Sendo assim um mesmo valor no banco poderia ser mostrado em minutos no aplicativo, enquanto no *Darwin* seria mostrado em horas.

É possível realizar todas as operações básicas em uma macro, tais como criação, leitura, alteração e exclusão. A parte mais trabalhosa desta funcionalidade foi a construção do *front-end*, que demandou a utilização de muitos recursos do JS e do CSS. Por exemplo, a tela de parâmetros de macro é toda dinâmica, pois os clientes podem querer parâmetros de macro diferentes.

Um conhecimento adquirido ao trabalhar nesta demanda foi a utilização de elementos do HTML em formato de *array*. Conforme pode ser visto no Quadro 12, os colchetes colocados após o nome do componente *input*, torna ele elemento passível de iteração, assim como é feito em um vetor (*array*).

Assim, para cada macro cadastrada no banco, itera-se sobre o *array*, construindo a tela dinamicamente, sem a necessidade de manutenção a cada que

vez uma macro for adicionada ou removida.

Quadro 12 - Array no HTML

```
<input type='tipo' id='contador' name='vetor[".contador."]'>
```

Fonte: Autor

Para o desenvolvimento desta atividade foram trabalhadas todas tecnologias mencionadas na Capítulo 2, sendo ela SQL, PHP, HTML, JavaScript e CSS, além da biblioteca JQuery.

3.1.5 Gestão dos vales pedágio

Dentre as empresas que a Technolog realiza a gestão de transportes, algumas delas fornecem vale-pedágio. Assim, foi solicitado que, quando uma carreta saísse carregada desses clientes, os pedágios em que o veículo passasse deveriam vir com o valor zerado, informando que aquela passagem foi paga pelo vale-pedágio que o cliente possui.

Para atender às especificações do cliente, foi criado um campo na tabela de “Ponto de Interesse”, para ser informado se o cliente pode usar vale pedágio ou não. Um ponto de interesse é uma localização específica, demarcada através das informações geográficas latitude e longitude de interesse do cliente. São exemplos de pontos de interesse: locais de descarregamento, locais de carregamento, postos de combustíveis, praças de pedágios, oficinas, borracharias dentre outros.

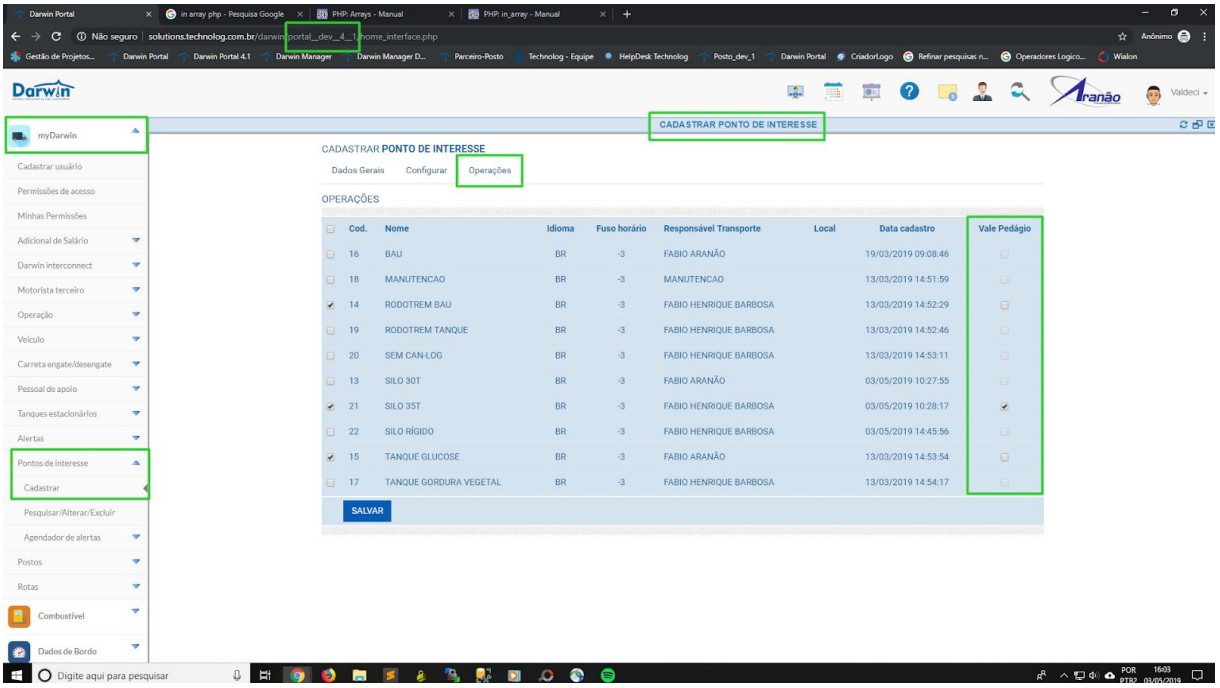
O Figura 10 apresenta a alteração realizada na interface do Darwin para contemplar esse novo requisito do cliente.

Posteriormente, foi adicionada uma função no sistema para quando o analista de logística fechar uma viagem carregada com a origem em um dos pontos de interesse, com a opção de vale pedágio, os valores do pedágio sejam indicados como pagos com o vale pedágio.

O desenvolvimento desta atividade não foi trivial, pois a lógica implementada requer a interligação de três funcionalidades diferentes, sendo elas: pontos de interesse, fechamento de viagens e relatórios. Antes de iniciar o desenvolvimento, foi

preciso uma análise aprofundada para entender com se dá o relacionamento de tais funcionalidades.

Figura 10 - Cadastro de *POI* com a opção de vale pedágio



Fonte: Technolog (2019)

Esta atividade proporcionou trabalhar com algumas das tecnologias apresentadas na Seção 2, sendo elas: HTML, PHP, SQL e CSS. O HTML foi utilizado para adição dos elementos visuais, tais como, a caixa de seleção apresentada na Figura 10. Com o CSS, realizou-se ajustes visuais do elementos inseridos no HTML, como largura das colunas adicionadas, tamanho das caixas de seleção, entre outros. A linguagem de programação PHP foi responsável pela inserção e recuperação de dados na base de dados.

3.2 Descrição das atividades desenvolvidas no Manager

O sistema *Manager* não gera tantas demandas quanto o *Darwin*, pois seu foco não é tão alinhado ao dia-a-dia das transportadoras, possuindo assim, um fluxo de visitação menor. Contudo, algumas atividades com um grau de relevância maior foram desenvolvidas nele.

3.2.1 Ajuste no filtro do módulo de forçar carga

Ao acessar o sistema *Manager*, o mesmo possui um módulo denominado “forçar carga”, o qual é utilizado caso a integração com o sistema *Wialon* (sistema não desenvolvido pela technolog, mas que armazena os dados dos hardwares instalados nos caminhões) apresente algum problema ao transpor os dados para a base de dados do *Darwin*.

Por exemplo, se a base de dados do Darwin no período de 01/10/2019 até 10/11/2019 fosse deletada por qualquer razão, seria possível recuperar estes dados forçando a carga de dados referente ao mesmo período no qual ocorreu o incidente, sendo estes do dia 01/10/2019 até 10/11/2019. Logo os dados seriam novamente carregados para a base de dados do Darwin.

Contudo, surgiu o problema que consiste na recuperação de dados obtidos em longos períodos, pois o volume de dados gerado pelos veículos é muito grande e, consequentemente, demandam muito tempo para serem transferidos. Logo, a empresa teve a necessidade de restringir esse período para não permitir que dados sejam obtidos em um intervalo maior do que trinta dias.

No exemplo supracitado acima, a recuperação dos dados teria quer ser fracionada em dois períodos, por exemplo, o primeiro período começando no dia 01/10/2019 e indo até o dia 20/10/2019, e o segundo começando no dia 21/10/2019 e indo até o dia 10/11/2019.

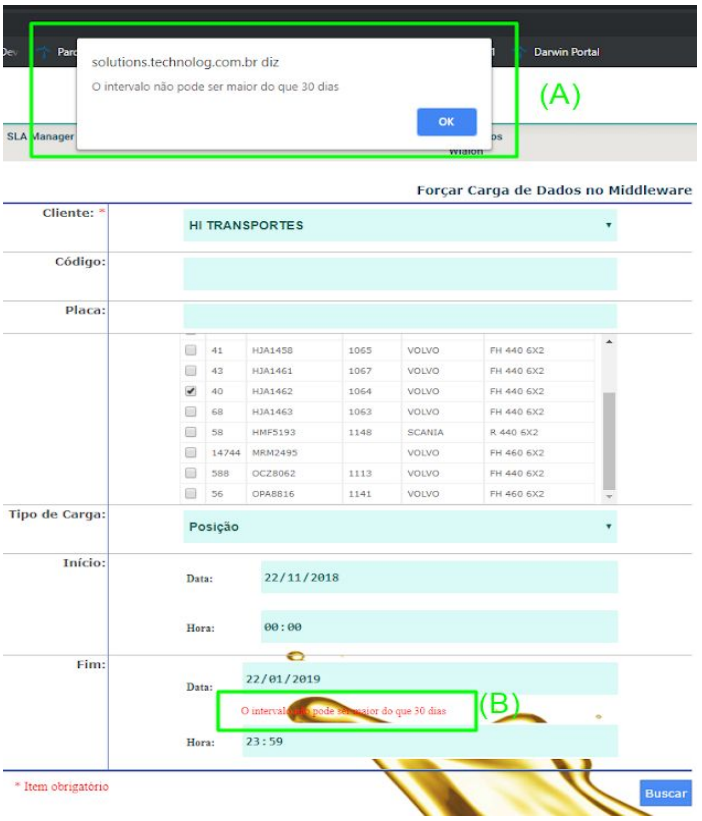
A maior parte do desenvolvimento dessa atividade foi realizada com a linguagem JS e a biblioteca *JQuery*. Na Figura 11 é apresentada a tela correspondente a essa funcionalidade. As áreas demarcadas pelos quadros na cor verde representam a implementação da resolução do problema, assim como, as atividades descritas acima.

A Figura 11 (A) refere-se ao *alert*, criado via JS contendo a mensagem de orientação do usuário, orientando o mesmo do período permitido para que seja possível realizar o procedimento de “forçar carga”.

A Figura 11 (B) refere-se à mensagem exibida ao usuário, utilizando funções da biblioteca JQuery. Por meio dessa função, busca-se o elemento HTML pelo seu

ID (*identity*) e atualiza o *status* de exibição para *show* (exibir) ou *hide* (esconder).

Figura 11 - Módulo forçar carga do sistema Manager



Fonte: Technolog (2019)

3.2.2 SimplePM

Foi adicionado ao sistema *Manager* a opção "SimplePM", um *chip* multi-operadoras. As linhas 1 e 2 do Quadro 13 apresentam a inserção deste registro na base de dados do sistema.

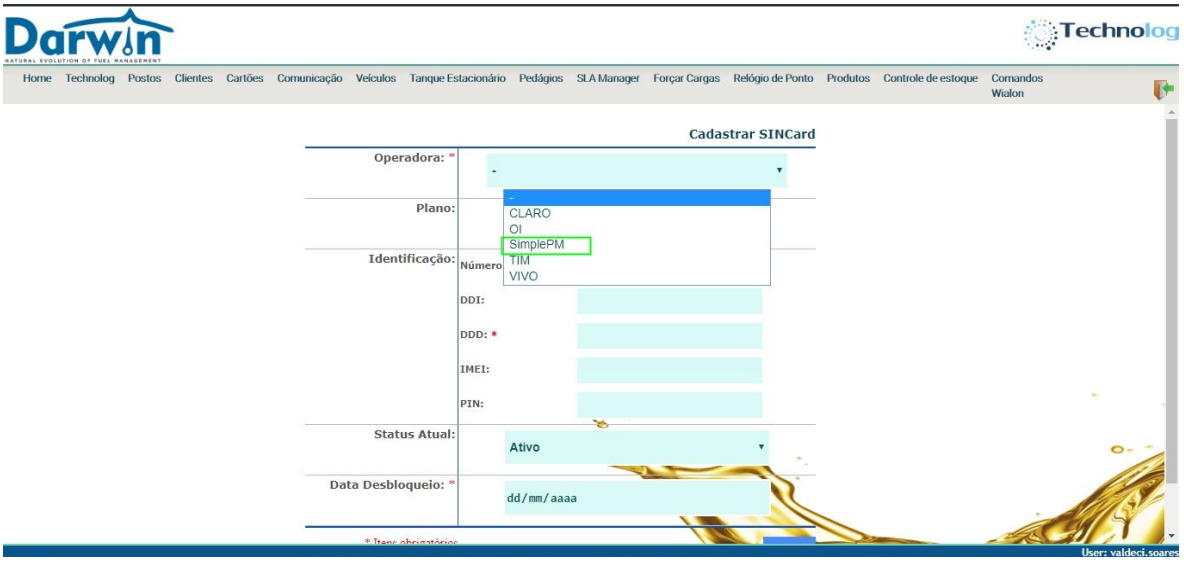
Após a cláusula de inserção, é realizado um comando "*select" linhas três e quatro* para exibir os dados que foram cadastrados, confirmando sua corretude. Este registro foi inserido na base de dados para que essa opção pudesse ser apresentada no *front-end* do sistema (Figura 12).

Quadro 13 - Inserção e visualização de valores no banco de dados

```
1 INSERT INTO t_operadoras_telefonia (nome, descricao, flg_ativo)
2 VALUES ('SimplePM', 'MULTI OPERADORAS', 1);
3 SELECT * FROM t_operadoras_telefonia WHERE flg_ativo = 1
4 ORDER BY nome;
```

Fonte: Autor

Figura 12 - Tela do sistema que representa a alteração efetuada no banco



Fonte: Technolog (2019)

A Figura 13 mostra o resultado obtido ao executar as linhas três e quatro do Quadro 13. A coluna “cod” representa a chave primária do registro, que permite ser usada como um índice de referência para criar relacionamentos com as demais tabelas do banco de dados.

A coluna “nome” representa o nome da operadora. A coluna “descrição” não é um campo de preenchimento obrigatório, trata-se de observações que o usuário pode fazer ou não.

A coluna “flg_ativo” assume o valor 1 quando o registro está ativo; ao se excluir o registro, o valor desta *flag* é setada para 0.

Figura 13 - Exibição de valores cadastrados no banco

	cod	nome	descricao	flg_ativo
1	2	CLARO		1
2	3	OI		1
3	5	SimplePM	MULTI OPERADORAS	1
4	4	TIM		1
5	1	VIVO	TELEFONICA	1

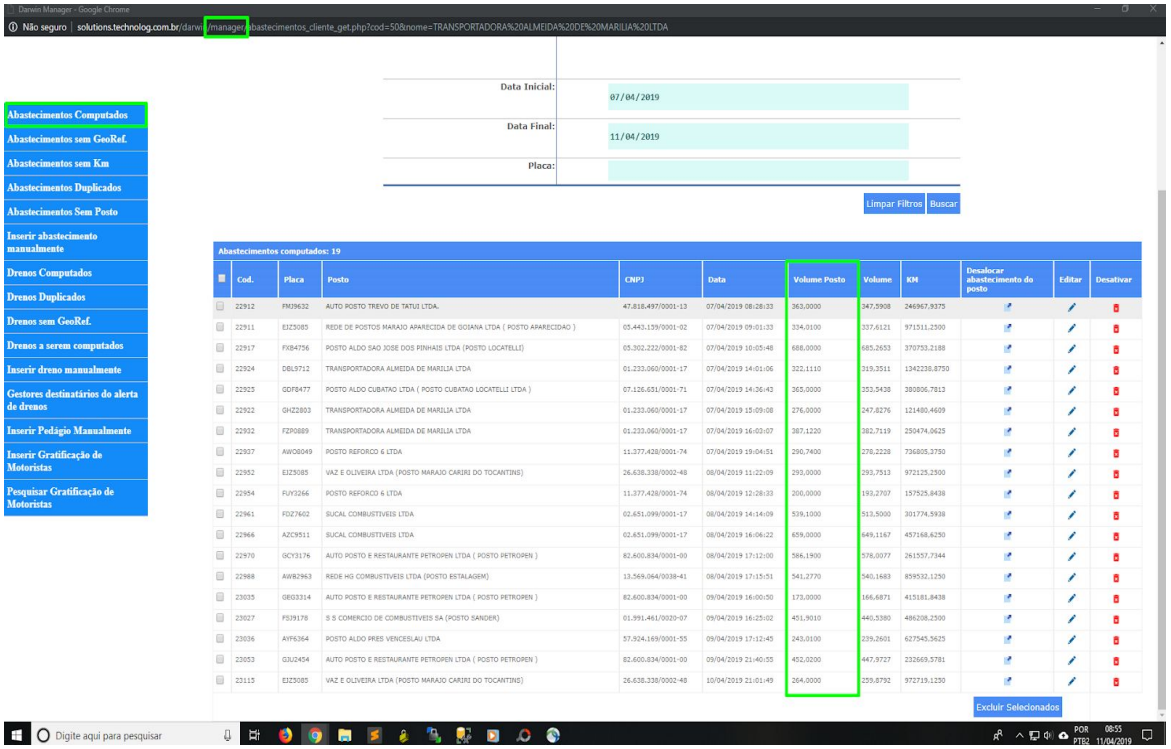
Fonte: Technolog (2019)

O desenvolvimento desta atividade apresentou poucas dificuldades, uma vez que os conhecimentos necessários para desenvolvê-la foram apenas da linguagem SQL, com uma cláusula simples de inserção de registros, bem como, da linguagem PHP para a busca das informações dos chips de telefonia móvel, na base dados.

3.2.3 Comparativo de volume do *Manager* com o dos postos

O sistema *Manager* exibe os volumes de abastecimentos computados com base nas informações geradas pelos hardwares instalados nos caminhões. Assim, os analistas realizam verificações, a fim de conferir se o volume informado pelos postos coincide com os computados nos veículos. A fim de prover uma verificação de forma mais rápida e eficiente, essas informações foram inseridas na tela do sistema em colunas lado a lado, como pode ser visualizado na Figura 14. Após o desenvolvimento desta atividade o sistema passou apresentar duas colunas, uma com os dados lançados pelos postos de combustíveis e a outra que é computada pelo *Manager*.

Figura 14 - Comparativo de abastecimento do sistema manager



Fonte: Technolog (2019)

A maior dificuldade para implementação desta demanda foi encontrar em qual tabela do sistema estava localizada a informação referente ao abastecimento efetuado pelo posto. Após ter encontrado esta informação (para isso foi necessário consultar a coordenadora de desenvolvimento e o supervisor do estágio), foram utilizadas quatro tecnologias para o desenvolvimento da funcionalidade, sendo elas: SQL, PHP, HTML e CSS

O SQL foi utilizado para recuperação dos dados no banco. A *query* é executada via código PHP, que por sua vez passa os dados para serem exibidos pelo *browser*, no formato HTML. O CSS foi utilizado para manter a identidade visual do sistema, tais como: cores, largura da coluna, justificação a esquerda, dentre outras.

3.3 Descrição das atividades desenvolvidas na API do sistema Darwin

Para a realização das atividades relacionadas à API do sistema *Darwin* foram utilizadas as tecnologias PHP e SQL. O PHP recebe as informações via requisição do tipo POST, trata os dados e realiza a manipulação necessária na base de dados, via SQL. A maior dificuldade no desenvolvimento da API foi, primeiramente, entender como funciona um *web service*. Depois, foi relembrar os conceitos de redes de computadores, para retornar os códigos HTTP corretos (por exemplo, 202 para requisição aceita, 404 para recurso não encontrado, dentre outros).

3.3.1 Alteração salarial

Foi desenvolvido um *web service* para atualizar os dados do salário do motorista. Para isso, é necessário que o sistema cliente envie uma requisição do tipo POST, passando como parâmetro no corpo (*body*) da mensagem um objeto JSON, conforme representado no Quadro 13.

A complexidade para implementação deste serviço está relacionado ao fato que, para atualizar o salário de um motorista, não basta apenas realizar uma operação de atualização (*update*) no banco de dados. Para isso, é preciso gerar um

histórico de atualização de salário, pois estas alterações ficam registradas na carteira de trabalho do motorista.

Quadro 14 - Requisição para alteração salarial dos motoristas

```
{
   "cod": 785,
   "carga_horaria_mensal": 220,
   "salario": 50000,
   "km_premiacao": 155,
   "adicionais_salario": [{
      "cod": 1,
      "tag_adicional": "VA",
      "cod_rel_adicional": 9,
      "valor_adicional": 1000
   }, {
      "cod": 2,
      "tag_adicional": "abc",
      "cod_rel_adicional": 10,
      "valor_adicional": 200
   }]
}
```

Fonte: Technolog (2019)

Como apresentado na Figura 15, além do salário base, também pode ser adicionado, removido ou atualizado adicionais de salário. Estes adicionais são bônus que a empresa oferece para os motoristas, podendo ser oferecidos por diversos motivos, tais como, cuidado com veículo, consumo de combustível dentro das metas, dentre outros.

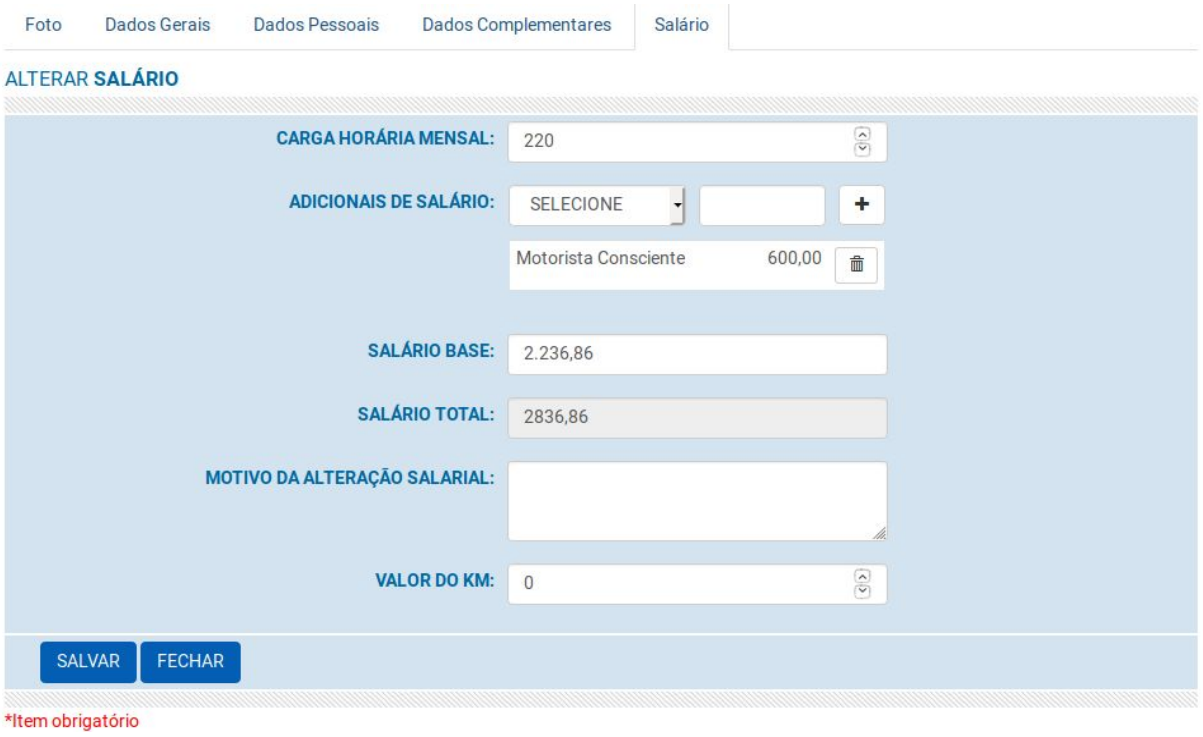
As operações de atualização de salário são iniciadas realizando uma consulta no banco de dados para verificar quais os adicionais já estão cadastrados e qual o valor do salário base cadastrado no sistema. Essa verificação é necessária, pois, caso a requisição traga os mesmos valores que estão cadastrados no banco de dados, nenhuma alteração será feita.

Este *web service* manipula dados em três tabelas, sendo elas: uma tabela para armazenar os registros referentes ao salário base, outra tabela para armazenar os registros referentes aos adicionais e uma terceira tabela que faz o relacionamento entre as duas tabelas mencionadas.

Caso o salário base seja alterado, o salário antigo é desativado e é

adicionado o novo salário base. Caso o salário antigo tenha algum adicional que o referenciava, esses adicionais são atualizados na tabela de relacionamento para que possam referenciar o novo salário base.

Figura 15 - Tela de alteração salarial do sistema *Darwin*



Fonte: Technolog (2019)

Na existência de algum adicional a ser removido, o registro do mesmo é desativado, e não excluído, possibilitando a realização de auditoria no sistema, caso seja necessário.

Os novos adicionais são inseridos em última etapa, já referenciando o salário base atual, que é o salário base verificado e com suas operações já concluídas. Não sendo preciso realizar novas verificações para essa operação, melhorando a performance da mesma, através da redução do custo de processamento que será efetuada apenas uma verificação.

A aplicação que faz uso do *web service,* após ter efetuada uma requisição, espera como retorno um objeto JSON. Esse objeto JSON contém uma variável denominada *status* que recebe o valor "*true*", caso todas as operações sejam

realizadas com sucesso, e “*false*” caso ao contrário.

3.3.2 Alterar os dados gerais dos motoristas

Foi desenvolvido também um *web service* para alteração dos dados gerais do motorista. O Quadro 14 mostra um exemplo da requisição que é feita ao servidor para atualização destes dados. Após o recebimento da requisição, o servidor verifica alguns campos obrigatórios tais como “código da empresa”, no qual o motorista está cadastrado, “código de cadastro do motorista”, “nome do motorista” e “CPF”.

Quadro 14 - Requisição para alterar dados gerais dos motoristas

http://solutions.technolog.com.br/darwin/mobile/darwin_colaboradores_dev/index.php?acao=pesquisarAlterarExcluir/atualizarDadosGerais&codCliente=43&motorista={"cod":785,"nome":”nome”,”cpf”:”000.000.000-00”,”rg”:”aa11111111”,”tel_empesa”:”00000000”,”tel_pessoal”:”00000000”,”tel_residencial”:”00000000”,”observacoes”:”teste”}

Fonte: Technolog (2019)

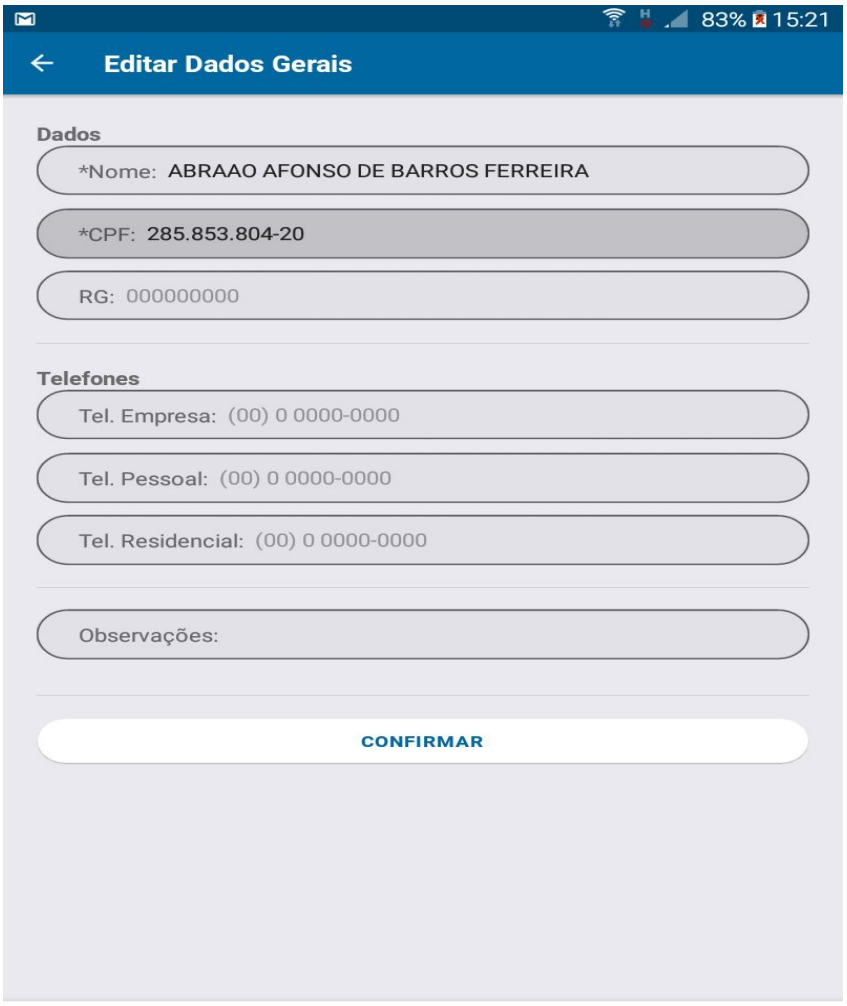
Caso alguns dos campos obrigatórios não passe na verificação, o servidor lança uma exceção com *status* de resposta contendo um código de erro 400 (código HTTP que significa que uma “solicitação inválida” foi realizada ao servidor), informando que o servidor não pode ou não irá processar a requisição devido ao campo obrigatório que está faltando.

Uma vez validados os dados, as alterações são efetuadas no banco de dados. Caso todas as operações sejam realizadas com sucesso, é retornado um objeto JSON contendo o *status* da operação “*true*”, caso ao contrário, é retornado um objeto JSON com *status* da operação “*false*”.

A Figura 16 apresenta a tela para interação com o usuário nos dispositivos móveis, a qual foi desenvolvida pelo funcionário responsável pelo desenvolvimento das aplicações móveis.

Similarmente a essa função, foram desenvolvidos *web services* para alterar os dados complementares e os dados pessoais dos motoristas, as quais não são apresentadas neste relatório, pois possuem bastante similaridade com as demais apresentadas nesta seção.

Figura 16 - Tela para editar dados Gerais nos dispositivos móveis



Fonte: Technolog (2019)

3.4 Resumo das atividades realizadas

A Tabela 1 apresenta, de forma resumida, as atividades realizadas ao longo do estágio, relacionando-as com as respectivas tecnologias utilizadas em seu desenvolvimento.

Tendo em vista que o tempo do desenvolvedor é um dos principais recursos no desenvolvimento de software, é apresentado uma coluna que relaciona as atividades desenvolvidas com o tempo aproximado em horas trabalhadas no desenvolvimento de cada atividade.

Tabela 1 - Resumo das atividades desenvolvidas e tecnologias utilizadas

Atividades Desenvolvidas	Tecnologias utilizadas	Tempo Gasto em Horas (aprox.)
Lançamento de multas	HTML, Javascript (com AJAX e Jquery), CSS, PHP e SQL.	176
Tradução do sistema	HTML e PHP	88
Correção na abertura de planilhas	HTML, Javascript e PHP	132
Compatibilidade de arquivos	Javascript e PHP	3
Darwin jornada macro	HTML, Javascript (com AJAX e Jquery), CSS, PHP e SQL.	352
Gestão de vale pedágio	HTML, Javascript (com AJAX e Jquery), CSS, PHP e SQL.	132
Ajuste no filtro de módulo de forçar carga	HTML e javascript	4
SimplePM	PHP e SQL	1
Comparativo volumétrico de combustível	HTML, CSS, PHP e SQL	8
API - Alteração salarial	PHP e SQL	88
API - Alterar dados gerais	PHP e SQL	44
API - Alterar dados complementares	PHP e SQL	44
API - Alterar dados pessoais	PHP e SQL	44

Fonte: Autor

4 CONSIDERAÇÕES FINAIS

O valor do estágio está muito além do financeiro. É uma experiência enriquecedora em diversos sentidos, principalmente para o crescimento na carreira e amadurecimento das convicções profissionais. Além de agregar valor aos currículos dos recém-formados, o estágio abre novas portas e aprimora as habilidades iniciadas e desenvolvidas na faculdade, como habilidades técnicas e interpessoais, mostrando para os futuros empregadores o grau de comprometimento desses futuros profissionais.

Ao final do estágio, conclui-se que a utilização das boas práticas adquiridas durante o período de graduação é uma forma de reduzir o retrabalho, pois as más práticas aumentam de forma considerável os custos nas etapas de manutenção e evolução do sistema.

Muitas disciplinas foram importantes para que o desenvolvimento das atividades do estágio fossem realizadas de maneira eficiente e eficaz. Dentre elas, estão as disciplinas de programação, como Introdução aos Algoritmos, Estrutura de Dados e Programação Orientada a Objetos. Tais disciplinas permitem ao aluno desenvolver o raciocínio lógico, para que o mesmo possa, então, desenvolver suas habilidades na linguagem mais apropriada para a resolução do problema. Outras disciplinas importantes de se mencionar são Engenharia de Software e Processos de Software, que fornecem um embasamento teórico adequado sobre o desenvolvimento de sistemas orientado pelas melhores práticas dessas disciplinas.

A maioria dos sistemas computacionais exige um meio de armazenamento dos dados. Por este motivo, a disciplina de Introdução a Sistemas de Banco de Dados e a disciplina de Sistemas Gerenciadores de Banco de Dados tiveram grande importância para concretização das atividades do estágio. Em geral, as bases de dados dos sistemas reais possuem grandes volumes de dados e, muitas vezes, consultas complexas devem ser realizadas nestas bases. Assim, a falta dos conhecimentos adquiridos em tais disciplinas poderiam impactar negativamente na conclusão de atividades relacionadas aos dados do sistema.

Participar de um estágio, antes de entrar no mercado de trabalho, pode gerar impactos positivos quanto ao preparo técnico e intelectual do futuro profissional, pois

permite que o conteúdo teórico aprendido em sala de aula, tenha uma maior absorção quando colocado em prática. O estágio também proporciona o conhecimento de uma ampla variedade de tecnologias, favorecendo a qualificação técnica, emocional e social. Ensina também o trabalho em equipe, no qual é possível vislumbrar novos horizontes e perspectivas diferentes.

5 REFERÊNCIAS

ALVAREZ, M. **O que é HTML**. 2004. Disponível em: <http://www.criarweb.com/artigos/7.php>. Acesso em: 12 de Novembro de 2018.

ANDERSON, David J. **Kanban**. 6 abr 2010.

ARRAYO, A.; SANTOS F. **Programação para a Web utilizando PHP**. Divisão de Serviços à Comunidade - Centro de Computação - Unicamp. P.1, 12 de maio de 2003.

CHAUBEY, R.; SURESH, J.K. **Integration vs. development: an engineering approach to building web applications**. IEEE Software, feb. 2001, p. 171-181.

CONALLEM, J.: **Building Web Applications with UML**. Addison Wesley, 2000.

Curso W3C Escritório Brasil. **CSS**. Disponível em: <http://www.w3c.br/pub/Cursos/CursoCSS3/css-web.pdf>. Acesso em: 29 de Julho de 2019.

MOBILE MAGAZINE. **DOMINE O USO DE WEB SERVICE - Tire aproveito deste recurso no Android e no iOS**; Disponível em: <https://www.devmedia.com.br/revista-mobile-magazine-51/28924>. Acesso em: 14 de Agosto de 2019

FENTON, N. E; PFLEEGER, S. L. **Software metrics: a rigorous approach**, 2ª ed. International Thomson Computer Press, 1997.

JUNIOR, M.L; FILHO, M. G. **Utilização do sistema kanban em empresas do estado de são paulo: estudo por meio de um survey**. XXVII Encontro nacional de engenharia de produção. Foz do Iguaçu, PR, Brasil, 09 a 11 de outubro de 2007, P.3.

JUNIOR, O. **Programação - Apostila Java Script**. s. d. Disponível em: <http://www.ebah.com.br/content/ABAAAfiNYAD/programacao-apostila-javascript>. Acesso em: 15 de Novembro de 2018.

JUNG, C. F. **Metodologia para pesquisa & desenvolvimento: aplicada a novas tecnologias, produtos e processos**. Rio de Janeiro/RJ: Axcel Books do Brasil Editora, 2004.

LAGE, M. J.; GODINHO, M. F. Gest. Prod., São Carlos, v. 15, n. 1, p. 173-188, jan.-abr. 2008.

LIMA, A. G.; **JavaScript - Aplicações Interativas para a WEb**. Belo Horizonte 2006. Disponível em: <http://www.elivros-gratis.net/scripts/download.asp?SEC=5&FL=Programacao-Manual-Javascript-wikibooks.zip&NOME=Javascript:%20Refer%EAncias%20%E0%20Lin

guagem&AUTOR=Adriano%20Gomes>. Acesso em: 29 de Julho de 2019.
LÓSCIO, B. F.; BURLE, C.; CALEGARI, N. **Data on the Web best practices**. W3C Working Draft, World Wide Web Consortium (W3C), May 2016. Disponível em: <https://www.w3.org/TR/dwbp/>. Acesso em: 17 de Novembro de 2018.

MCCALL, J. A.; RICHARDS, P. K.; WALTERS, G. F. **Factors in Software Quality**. [S.l.:s.n.], 1977.

NAVATHE, S. B. and ElMASRI, R. - **Sistemas de Banco de Dados**; tradução Daniel Vieira; revisão técnica Enzo Seraphim e Thatyana de Faria Piola Seraphim. -- 6.ed. -- São Paulo : Pearson Addison Eesley, ano 2011.

PEREIRA, Ana Paula. **O que é CSS?**. 2009. Disponível em: <http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm>. Acesso em: 01 de Dezembro de 2014.

PRESSMAN, R. S. **Engenharia de Software**. São Paulo: Makron Books, 1995.

Murphey, R. **Fundamentos de jQuery (jQuery Fundamentals)**. [http://www.rebeccamurphey.com]. Herberth Amaral (tradução e revisão), Elvis Luciano Guimarães (tradução e revisão), Diego Guimarães (tradução), Diego Caxito (tradução), Luciana Balieiro (revisão), Washington Botelho (revisão). Copyright © 2010.

REVISTA ENGENHARIA DE SOFTWARE MAGAZINE 39. **Reutilização de software**. Disponível em: <https://www.devmedia.com.br/reutilizacao-de-software-revista-engenharia-de-software-magazine-39/21956>. Acesso em: 13 agost. 2019.

SAMY, S. M.; Ajax com Jquery - **Requisições ajax com a simplicidade de jquery**. 1ª ed. Novatec, 2009.

SUTHERLAND, JEFF. **Scrum : a arte de fazer o dobro do trabalho na metade do tempo**. tradução de Natalie Gerhardt. - São Paulo : LeYa, 2014. Título original: Scrum : The art of doing twice the work in half the time

SILVA, M. JQuery: **A Biblioteca do Programador JavaScript**. 2008. Disponível em:<http://livrojquery.com.br/index-1ed.php>. Acesso em: 01 de Dezembro de 2018.

STRONG, D. M.; LEE, Y. W.; WANG, R. Y. **Data quality in context. Communications of the ACM**, v. 40, n. 5, p. 103-110, 1997.

WIKIPÉDIA: a enciclopédia livre. **Teste de software.** Flórida: Wikimedia Foundation, 2019. Disponível em: <https://pt.wikipedia.org/wiki/Teste_de_software>. Acesso em: 06 Agosto. 2019.

WIKIPÉDIA, a enciclopédia livre. **PHP**. Flórida: Wikimedia Foundation, 2018. Disponível em: <https://pt.wikipedia.org/w/index.php?title=PHP&oldid=53746661>.

Acesso em: 11 dez. 2018.