RAYDSON FERREIRA CARLOTA

RELATÓRIO DE ESTÁGIO: DESENVOLVIMENTO DE SISTEMAS WEB NO LABORATÓRIO DE ESTUDOS E PROJETOS EM MANEJO FLORESTAL

LAVRAS - MG

2019

RAYDSON FERREIRA CARLOTA

RELATÓRIO DE ESTÁGIO: DESENVOLVIMENTO DE SISTEMAS WEB NO LABORATÓRIO DE ESTUDOS E PROJETOS EM MANEJO FLORESTAL

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

Profa. Dra. Renata Teles Moreira
Orientadora

Bsc. Franciele Aparecida Ferreira
Coorientadora

LAVRAS – MG

2019

RAYDSON FERREIRA CARLOTA

RELATÓRIO DE ESTÁGIO: DESENVOLVIMENTO DE SISTEMAS WEB NO LABORATÓRIO DE ESTUDOS E PROJETOS EM MANEJO FLORESTAL

INTERNSHIP REPORT: DEVELOPMENT OF WEB SYSTEMS IN THE LABORATORY OF STUDIES AND PROJECTS IN FOREST MANAGEMENT

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

Aprovado em 22 de Novembro de 2019.
Prof. Dr. Maurício Ronny de Almeida Souza - UFLA
Prof. Dr. Ramon Gomes Costa - UFLA

Profa. Dra. Renata Teles Moreira
Orientadora

Bsc. Franciele Aparecida Ferreira
Coorientadora

LAVRAS – MG

2019

AGRADECIMENTOS

A Deus, primeiramente, pela oportunidade de realizar esse sonho.

Aos meus pais, Roméria e Ronaldo pelo incentivo e apoio em todos os momentos da minha vida, e aos meus irmãos Rildson (*in memorian*) e Robson.

Aos meus amigos, que são sempre suporte e alegria, por sempre confiarem em mim.

A orientadora Renata e a coorientadora Franciele por me guiarem durante a construção deste trabalho.

E a todos que de algum modo fizeram ou fazem parte da minha vida, deixo aqui o meu muito obrigado.

RESUMO

O presente relatório visa apresentar os desafios enfrentados pelo discente durante a realização do estágio na empresa FUNDECC (Fundação de Desenvolvimento Científico e Cultural), e sua contribuição com o desenvolvimento de sistemas Web, que foram desde a parte da gerência própria da empresa, até sistemas que são capazes de monitorar e auxiliar a fiscalização e preservação de áreas florestais, as quais demandam atenção prioritária por parte dos órgãos federais. Alocado no prédio do LEMAF (Laboratório de Estudos e Projetos em Manejo Florestal) o discente teve a oportunidade de aprender e aplicar diversas tecnologias usadas para o desenvolvimento, como Java, Vue.js, AngularJS, Play Framework, PostgreSQL, entre outras, além de vivenciar e aplicar na prática a metodologia ágil Scrum. Ao fim, pode-se perceber a importância de se investir tempo em uma empresa para a realização de estágio, pois o conhecimento adquirido no mesmo, somado ao que é aprendido nas disciplinas, foi de suma importância para que ocorresse a contratação do discente, que realizando atividades que agregam valor a empresa, pode ter o seu trabalho reconhecido.

Palavras-chave: Sistemas Web. Preservação Ambiental. Java. Scrum.

ABSTRACT

This report aims to present the challenges faced by the student during the internship at FUNDECC (Fundação de Desenvolvimento Científico e Cultural), and the development of Web systems, ranging from the company's own management, to systems that are able to monitor and assist the inspection and preservation of forest areas, which require priority attention from federal agencies. Housed in the building of LEMAF (Laboratório de Estudos e Projetos em Manejo Florestal) the student had the opportunity to learn and apply various technologies used for development, such as Java, Vue.js, AngularJS, Play Framework, PostgreSQL, among others, besides experiencing and applying the agile Scrum methodology in practice. In the end, one can realize the importance of investing time in a company for the internship, because the knowledge acquired in it, added to what is learned in the subjects, was of paramount importance for hiring the student, which Performing activities that add value to the company can have your work recognized.

Keywords: Web systems. Environmental preservation. Java. Scrum.

SUMÁRIO

1. INTRODUÇÃO

Empresas constantemente exigem experiências e qualificações para contratação de seus colaboradores, e a realização de um estágio se torna de suma importância para a carreira de alunos que se interessam pelo mercado de trabalho. Com o crescimento das áreas ligadas a tecnologia da informação, surgem novas oportunidades de vivências dentro e fora das universidades para a capacitação dos alunos, como empresas júniors, bolsas de iniciação científica, estágios, entre outras que fornecem conhecimento e experiências necessárias para se adentrar em uma empresa.

Com vista a formar profissionais nas diversas áreas da tecnologia da informação, o curso de Bacharelado em Sistemas de Informação possui diversos docentes com experiências empresariais dispostos a passar toda a experiência através da ministração das matérias, tendo como resultado profissionais capacitados para o desenvolvimento e gerenciamento de sistemas, resolvendo problemas organizacionais e tecnológicos das mais diversas áreas.

Neste contexto, o desejo do estagiário de adquirir conhecimento prático nas áreas onde obteve domínio teórico, devido às matérias vistas na graduação, fez com que este buscasse a realização de um estágio no Laboratório de Estudos em Manejo Florestal (LEMAF) na área de desenvolvimento de *software*.

1.1. Objetivos

Este relatório tem como objetivo descrever as atividades desenvolvidas durante o estágio realizado no Laboratório de Estudos em Manejo Florestal (LEMAF), no período de Novembro/2018 a Setembro/2019, tendo como foco o desenvolvimento de sistemas Web voltados para o manejo florestal. Neste contexto, o estagiário atuou como desenvolvedor em 04 projetos, sendo eles: Sistema Gerencial; Sistema de Avaliação Qualitativa (SAQ 360); Protocolo Digital; e Fiscalização.

O estagiário é aluno do curso de bacharelado em Sistemas de Informação na Universidade Federal de Lavras (UFLA). O objetivo geral para a realização deste estágio foi obter conhecimento prático no desenvolvimento de *software* e obter experiência com o

trabalho em equipe em uma empres. Para contribuir com o alcance do objetivo geral, foram definidos os seguintes objetivos específicos:

- Conhecer e trabalhar com *frameworks* e tecnologias atuais que são usadas no mercado de trabalho;
- Trabalhar com Java voltado para o desenvolvimento Web;
- Adquirir experiência com o processo de desenvolvimento de *software* em organizações;
- Ter vantagem no mercado de trabalho, pela vivência na prática de um estágio;
- Desenvolver o trabalho em equipe.

1.2. **Estrutura do Trabalho**

Além deste capítulo introdutório, o presente documento conta com a seguinte estrutura: No capítulo 2 é apresentada a empresa onde foi realizado o estágio, sua estrutura organizacional, seus clientes e parceiros, e os principais serviços prestados pela mesma; No capítulo 3 são apresentados as principais tecnologias utilizadas para a realização do estágio e os *frameworks* de auxílio; No capítulo 4 pode-se ver as atividades desenvolvidas no período de estágio juntamente com exemplos do que foi desenvolvido; Por fim, no capítulo 5 são apresentadas as considerações finais relatando os resultados obtidos.

2. A EMPRESA

Localizada dentro da Universidade Federal de Lavras, a FUNDECC (Fundação de Desenvolvimento Científico e Cultural) tem por finalidade apoiar o desenvolvimento de atividades de ensino, pesquisa e extensão bem como os desenvolvimentos institucionais, científicos e tecnológicos da Universidade Federal de Lavras, mediante assessoramento à elaboração de projetos e administração dos recursos financeiros auferidos, etc (FUNDECC, 2019).

De acordo com a FUNDECC (2019), ela é reconhecida como entidade cuja atuação serve de base para que as ideias desenvolvidas na Universidade Federal de Lavras possam se transformar em projetos com resultados imediatos, produtivos, levando a Universidade além da sua função primordial, a produção de conhecimento e inteligência.

Parte dos funcionários da Fundação estão alocados no LEMAF, de onde são desenvolvidos os projetos voltados ao meio ambiente, sendo eles variados, como licenciamento para instalações de operações, fiscalização ambiental, cadastro ambiental rural, entre outros diversos projetos.

Inserido no DCF (Departamento de Ciências Florestais) da UFLA, o LEMAF tem como objetivo efetuar pesquisa, ensino e extensão relacionado ao manejo de floresta nativa e plantada. O LEMAF conduz diversos projetos em parceria e/ou convênio com órgãos estaduais e federais, bem como com a iniciativa privada.

O LEMAF tem como alicerces três áreas de conhecimento:

- Manejo Florestal: Administração da floresta para obtenção de benefícios econômicos, sociais e ambientais, respeitando-se os mecanismos de sustentação do ecossistema objeto do manejo.
- Geoprocessamento e Sensoriamento Remoto: Tratamento de informações geográficas, ou de dados georreferenciados, por meio de *softwares* específicos e cálculos.
- Tecnologia da Informação: Soluções providas por recursos de computação que visam produção, armazenamento, transmissão, acesso, segurança e o uso de informações.

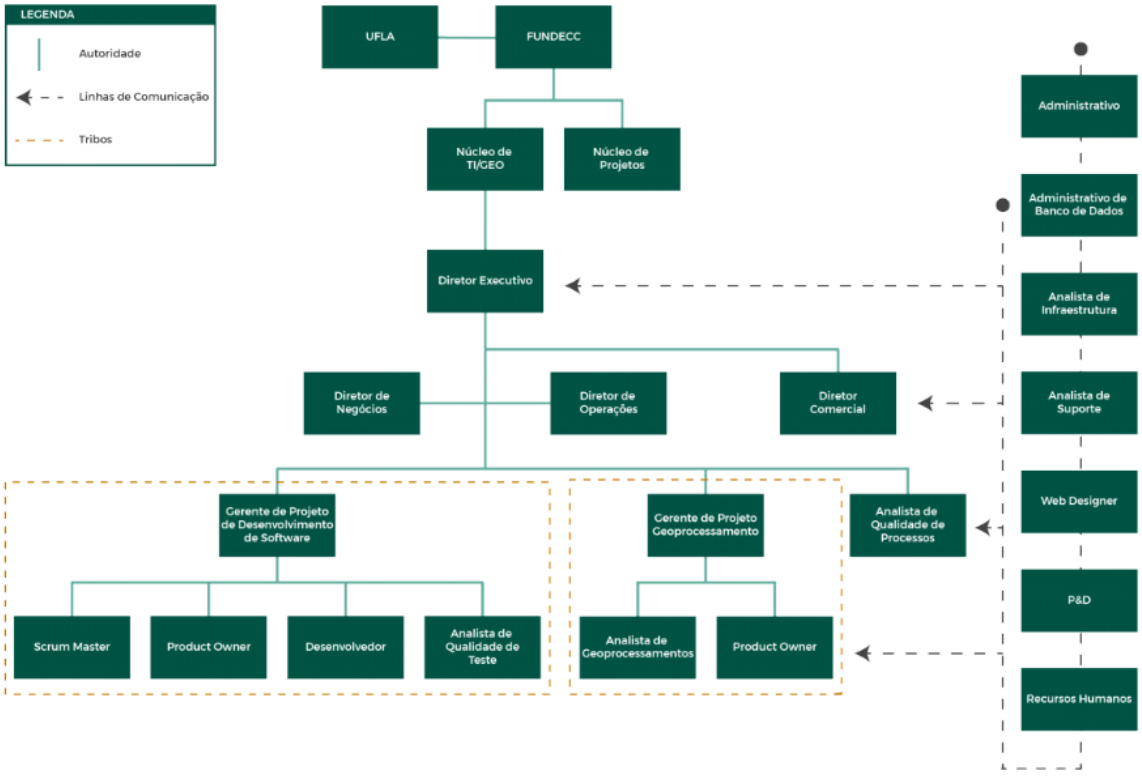
O estágio relatado neste documento foi realizado no setor de Tecnologia da Informação do LEMAF. No período de estágio, este setor do LEMAF contava com cerca de

120 funcionários, sendo estes divididos em 4 grupos chamados de tribos, somando um total de 12 equipes. As equipes são compostas geralmente por 1 Product Owner, 3 Desenvolvedores e 1 Analista de qualidade. Neste sentido, o estagiário atuou como desenvolvedor.

2.1. Processo da Organização

Seguindo um processo padronizado com base em seu organograma, conforme mostrado na Figura 1, o LEMAF é composto por quatro diretorias, sendo elas de negócios, operações, comercial e essas respondendo diretamente à diretoria executiva, a qual responde diretamente ao núcleo de TI e GEO da FUNDECC. Abaixo das diretorias, estão as Tribos, que são formadas por equipes de desenvolvimento de *software* e equipes de geoprocessamento, sendo estas compostas por pelo menos um gerente de projetos e um Product Owner.

Figura 1 – Organograma da organização - FUNDECC/LEMAF



Fonte: LEMAF (2018)

As tribos, no período do estágio, já não seguiam fielmente o organograma, pois foram inseridas pessoas que atuavam em outras áreas. Com isso a estrutura organizacional das tribos até o fim do estágio foi composta por:

- **Gerente de Projetos**: profissional responsável pela realização dos objetivos do projeto. Sendo suas funções no projeto: identificar as necessidades, estabelecer objetivos claros e alcançáveis, balancear as demandas conflitantes de qualidade, escopo, tempo e custo e adaptar as especificações, dos planos e da abordagem às diferentes preocupações e expectativas das diversas partes interessadas.
- **Analista de banco de dados**: profissional responsável por administrar os bancos de dados referentes aos projetos da tribo.
- **Designer**: profissional responsável por desenhar e criar protótipos, logos, e tudo relacionado a interface gráfica dos projetos da tribo.
- **Assistente administrativo**: profissional responsável por auxiliar o gerente de projetos a administrar os projetos, cuidando dos papéis referentes às ordens de serviços, ofícios e partes administrativas da gerência de pessoas da tribo.
- **Squads**: os squads ou equipes de desenvolvimento são responsáveis por todos os aspectos de um determinado projeto. Tendo autonomia para tomar decisões e definir prioridades que devem sempre estar alinhadas aos objetivos do negócio.

3. TECNOLOGIAS UTILIZADAS

Este Capítulo tem por objetivo elencar as tecnologias e *frameworks* utilizados no desenvolvimento das atividades pelo estagiário no período de estágio.

3.1. Tecnologias *Front-end*

São chamadas de tecnologias *front-end* aquelas que são executadas no lado do cliente das aplicações. Nesta seção serão apresentadas as principais tecnologias para desenvolvimento *front-end* utilizadas durante o estágio.

3.1.1. HTML[1]

HTML (*Hypertext Markup Language,* que significa Linguagem de Marcação de Hipertexto) é utilizada em páginas Web como uma linguagem de marcação. Criada por Tim Berners-Lee em 1991, o HTML usa "marcação" para declarar texto, imagens e outros conteúdos para exibição em um navegador da Web. A marcação HTML inclui "elementos" especiais, como <head>, <title>, <body>, <header>, <footer>, <p>, , , <img>, <nav>, e muitos outros (Mozilla, 2019).

3.1.2. CSS[2]

CSS (*Cascading Style Sheets* ou Folhas de Estilo em Cascata) é uma linguagem de estilização utilizada para descrever a apresentação de um documento escrito em HTML ou em XML[3]. As tecnologias CSS e HTML geralmente trabalham juntas para descrever a aparência e a estrutura de uma página Web (BESSER, 2001). Utilizando CSS, há a possibilidade da formatação das informações que são entregues ao navegador. Essas informações podem ser qualquer coisa, sendo um texto, imagem, vídeo, áudio ou qualquer outro elemento que a linguagem HTML aceita. Essa formatação, na maioria das vezes é visual (EIS; FERREIRA, 2012).

---

[1] https://www.w3schools.com/tags/
[2] https://www.w3schools.com/cssref/
[3] XML (Extensible Markup Language) é uma recomendação da W3C para gerar linguagens de marcação para necessidades especiais.

Segundo o Mozilla (2019), o CSS é uma das principais linguagens da *open Web* e tem sido padronizada pela Especificação da W3C[4]. Com seu desenvolvimento em níveis, o CCS1 está atualmente obsoleto, dando espaço para o CCS2.1 que é uma recomendação e o CSS3, que está dividido em pequenos módulos, e progredindo para a sua padronização.

3.1.3. **JavaScript**

JavaScript é uma linguagem de programação interpretada de alto nível, que também pode ser caracterizada como dinâmica, fracamente tipada e multi-paradigma. Dentro da *World Wide Web* é uma das três principais tecnologias, juntamente com HTML e CSS. Por garantir interatividade em páginas Web, o JavaScript é uma parte essencial dos aplicativos desenvolvidos para a plataforma (FLANAGAN, 2011).

Ainda segundo Flanagan (2011), A maioria dos sites e, todos os navegadores modernos, utilizam o JavaScript, além de *desktops*, consoles de jogos, tablets, e *smartphones* que possuem interpretadores de JavaScript, tornando-o a linguagem de programação mais onipresente da história. JavaScript faz parte do conjunto de tecnologias que todo desenvolvedor da Web deve aprender, junto do HTML e CSS.

3.1.4. **Pug**

Pug, é um *template engine* com alto desempenho e rico em recursos, tem sua base implementada com JavaScript para o ecossistema Node.js[5]. Pug funciona como um intermediário entre o Node.js e o HTML, ou seja, em tempo de execução, o Pug substitui as variáveis do projeto por valores atuais e, em seguida, envia a sequência HTML resultante para o cliente (PUG).

O Pug é um pré-processador e, como tal, ajuda a realizar tarefas como acabar com o trabalho repetitivo, fornecendo recursos não disponíveis em HTML simples. (REGADAS, 2016). Além disso, apresenta facilidade ao trabalhar com uma sintaxe indentada fazendo com

[4] O World Wide Web Consortium é a principal organização de padronização da World Wide Web. Consiste em um consórcio internacional com 450 membros, agrega empresas, órgãos governamentais e organizações independentes com a finalidade de estabelecer padrões para a criação e a interpretação de conteúdos para a Web. (W3C, 2019).

[5] Node.js é um interpretador de JavaScript assíncrono com código aberto orientado a eventos, criado por Ryan Dahl em 2009, focado em migrar a programação do JavaScript do cliente para os servidores.

que o código seja mais legível ao se trocar as *tags* (marcações) por simples declarações e hierarquias, como pode ser visto na Figura 2.

Figura 2 - Exemplo de código escrito em Pug

```
1. doctype html
2. html(lang='pt')
3.         head
4.                 title Pug
5.         body
6.                 h1 hello world em pug
7.                 div.container
8.                         p pug é muito simples!
```

Fonte: Do autor (2019)

O código demonstrado na Figura 2, ao ser compilado, gerará um código HTML que pode ser observado pela Figura 3.

Figura 3 - Exemplo de código compilado de Pug para HTML

```
1. <!DOCTYPE html>
2. <html lang="pt">
3.         <head>
4.                 <title>Pug</title>
5.         </head>
6.         <body>
7.                 <h1>hello world em pug</h1>
8.                 <div class="container">
9.                         <p>pug é muito simples!<p>
10.                </div>
11.        </body>
12. </html>
```

Fonte: Do autor (2019)

3.1.5. **Sass**

Sass ("*syntactically awesome stylesheets*", ou "folhas de estilo sintaticamente incríveis") é uma metalinguagem sobre CSS usada para descrever o estilo de um documento de maneira limpa e estrutural, com mais poder do que o CSS simples permite. O Sass fornece uma sintaxe mais simples e elegante para CSS e implementa vários recursos que são úteis para criar folhas de estilo gerenciáveis (CATLIN, 2013).

Sass consiste em duas sintaxes: (i) a sintaxe original, chamada de "sintaxe indentada"; e (ii) a sintaxe mais recente, “SCSS”. Conforme pode ser visto na Figura 4, a primeira sintaxe usa indentação para separar blocos de código e caracteres de nova linha para separar regras. A segunda sintaxe, conforme mostrado na Figura 5, usa formatação de bloco, como a de CSS. A sintaxe SCSS usa chaves para designar blocos de código e ponto-e-vírgula para separar linhas dentro de um bloco. Os arquivos com “sintaxe de indentação” e “SCSS” são tradicionalmente utilizados contendo as extensões .sass e .scss, respectivamente (CATLIN, 2013).

Figura 4 - Exemplo de código escrito em Sass com “sintaxe indentada”

```
1.  #area-CAR
2.      .labeltitle
3.          .inputCAR
4.              input
5.                  border-radius: 4px 0 0 0
6.              .el-input-group__append
7.                  border-radius: 0 4px 0 0
8.              .buttonAddCAR
9.                  background-color: #FF8C00
10.                 color: #fff
11.                 border: solid 1px #E0E0E0
12.                 border-radius: 0 4px 0 0
13.                 height: 40px
```

Fonte: Do autor (2019)

Figura 5 - Exemplo de código escrito em Sass com “sintaxe SCSS”

```
1. #area-CAR {
2.         .labeltitle {
3.                 .inputCAR {
4.                         input {
5.                                 border-radius: 4px 0 0 0
6.                         }
7.                         .el-input-group__append {
8.                                 border-radius: 0 4px 0 0
9.                         }
10.                        .buttonAddCAR {
11.                                background-color: #FF8C00
12.                                color: #fff
13.                                border: solid 1px #E0E0E0
14.                                border-radius: 0 4px 0 0
15.                                height: 40px
16.                        }
17.                }
18.        }
19. }
```

Fonte: Do autor (2019)

3.2. Tecnologias *Back-end*

São chamadas de tecnologias *back-end* que são executadas no lado do servidor das aplicações. Nesta seção serão apresentadas as principais tecnologias utilizadas para desenvolvimento *back-end* durante o estágio.

3.2.1. Java

Java é uma linguagem de programação orientada a objetos, desenvolvida na década de 90, e lançada oficialmente no ano de 1995 pela empresa Sun Microsystems em uma

conferência. O Java gerou interesse imediato na comunidade comercial por causa do interesse pela *World Wide Web* (DEITEL, 2005, p. 59).

De acordo com Deitel (2005), Java é uma poderosa linguagem de programação que pode ser utilizada tanto para diversão para os iniciantes, quanto oferece recursos apropriados para os programadores experientes construírem poderosos sistemas de informação.

Para Campione e Walrath (1996), o Java é uma linguagem de programação completa, que pode ser usada para diversos fins, como o desenvolvimento de aplicações baseadas na rede Internet, redes fechadas ou até programas stand-alone.

O uso da plataforma Java para desenvolvimento de sistemas Web tem se mostrado bastante adequado, fazendo com que empresas em todo mundo adote esta tecnologia para esse fim. (AQUINO JUNIOR, 2002, p. iii).

3.2.2. **PostgreSQL**

O PostgreSQL é um sistema de gerenciamento de banco de dados objeto-relacional (SGBDOR) baseado no POSTGRES (que era seu o nome antigo), desenvolvido na Universidade da Califórnia no Departamento de Ciência da Computação de Berkeley. O POSTGRES foi pioneiro em muitos conceitos que só se tornaram disponíveis em alguns sistemas comerciais de banco de dados muito mais tarde (POSTGRES, 2019).

O PostgreSQL é um descendente de código aberto desse código original de Berkeley. Com grande parte do padrão SQL, ele oferece muitos recursos modernos, como:

- Consultas complexas;
- Chaves estrangeiras;
- Gatilhos;
- Visualizações atualizáveis;
- Integridade transacional;
- Controle de simultaneidade multiversão.

Além disso, o PostgreSQL pode ser estendido pelo usuário de várias maneiras, por exemplo, adicionando novos tipos de dados, funções, operadores, funções agregadas, métodos de índice e linguagem processuais.

Devido à sua licença liberal, o PostgreSQL pode ser usado, modificado e distribuído gratuitamente por qualquer pessoa, para qualquer fim, seja ele privado, comercial ou acadêmico (POSTGRES, 2019).

3.3. **Frameworks**

Segundo Ré (2002), os *Frameworks* possibilitam não somente a reutilização de componentes isolados, mas também de toda a arquitetura pertencente à domínio específico. Nesta seção serão apresentadas os principais *Frameworks* utilizados para desenvolvimento tanto do *front-end* quanto do *back-end* durante o estágio.

3.3.1. **Vue.js**

O Vue.js é um *Framework* JavaScript que está popularizando e crescendo muito rapidamente no meio de desenvolvimento Web. O Vue.js foi criado em 2014 e teve sua primeira versão lançada por Evan You. (MACRAE, 2018).

De acordo com Macrae (2018), o Vue.js facilita aos desenvolvedores a criação de sites interativos e ricos, permitindo a criação de sistemas Web totalmente funcionais. Os sistemas criados através do Vue.js podem manipular de dados complexos, manipular o roteamento do lado do cliente sem depender de um servidor e, às vezes, criar páginas que acessem o servidor apenas uma vez para carregar os dados.

3.3.2. **AngularJS**

O AngularJS é um *Framework* estrutural para aplicativos Web dinâmicos. Ele permite o uso do HTML como linguagem base e estende a sua sintaxe para expressar os componentes do aplicativo de forma clara e sucinta. A ligação de dados e a injeção de dependência do AngularJS eliminam grande parte do código que teria de ser escrito. Tudo isso acontece no navegador, tornando-o um parceiro ideal com qualquer tecnologia de servidor (Google, 2018).

Seguindo o padrão MVC (*Model-View-Controller*), o AngularJS encoraja o baixo acoplamento entre apresentação, dados e componentes lógicos. Com o uso de injeção de dependência, serviços que comumente eram designados ao lado servidor da aplicação, como as *controllers* para os componentes de visualização, são trazidos para o lado cliente da aplicação. Consequentemente, o peso do *back-end* é radicalmente reduzido, levando a aplicações muito mais leves (Google, 2018).

3.3.3. **Leaflet**

Segundo Agafonkin (2019), o Leaflet, é a principal biblioteca JavaScript de código aberto para mapas interativos, compatíveis com dispositivos móveis e *desktop*. Ele possui todos os recursos de mapeamento que a maioria dos desenvolvedores precisa.

O Leaflet foi projetado para ser simples, tendo em mente desempenho e usabilidade. Ele funciona de maneira eficiente em todas as principais plataformas móveis e de *desktop*, pode ser estendido através da utilização de plugins, possui uma API bonita, fácil de usar e bem documentada e um código-fonte simples e legível, o que facilita na contribuição (AGAFONKIN, 2019).

3.3.4. **Play Framework**

O Play é um *Framework* é uma estrutura voltada para o desenvolvimento Web, baseado nas linguagens Java e Scala[6]. Visando a alta produtividade, ele integra componentes e APIs (“*Application Programming Interface*" ou "Interface de Programação de Aplicativos") para o desenvolvimento moderno de aplicativos da web (Lightbend, 2018).

O Play Framework também utiliza a arquitetura MVC, fornecendo padrões de programação concisos e funcionais. Nele é possível encontrar todos os componentes necessários para criar aplicativos Web e serviços REST[7], como um servidor HTTP integrado, tratamento de formulários, segurança na comunicação entre aplicações, um poderoso mecanismo de roteamento, entre outras funcionalidades. O Play economiza um tempo

[6] https://www.scala-lang.org/

[7] Representational State Transfer, em português Transferência Representacional de Estado, é um estilo de arquitetura de software que define um conjunto de restrições a serem usados para a criação de web services (W3C, 2004).

precioso de desenvolvimento, oferecendo suporte direto às tarefas diárias e carregamento imediato, fazendo com que o trabalho desenvolvido seja compilado e mostrado instantaneamente (Lightbend, 2018).

3.3.5. **Scrum**

O Scrum é um *Framework* utilizado para gerenciar o desenvolvimento e manutenção de produtos complexos (SCHWABER; SUTHERLAND, 2018). Usado muita das vezes em trabalhos complexos nos quais é impossível prever tudo o que ocorrerá, o Scrum oferece uma estrutura e um conjunto de práticas que garante visibilidade. Isso permite que os profissionais que utilizam o Scrum saibam exatamente o que está acontecendo e façam ajustes no local para manter o projeto em direção às metas desejadas (SCHWABER, 2004).

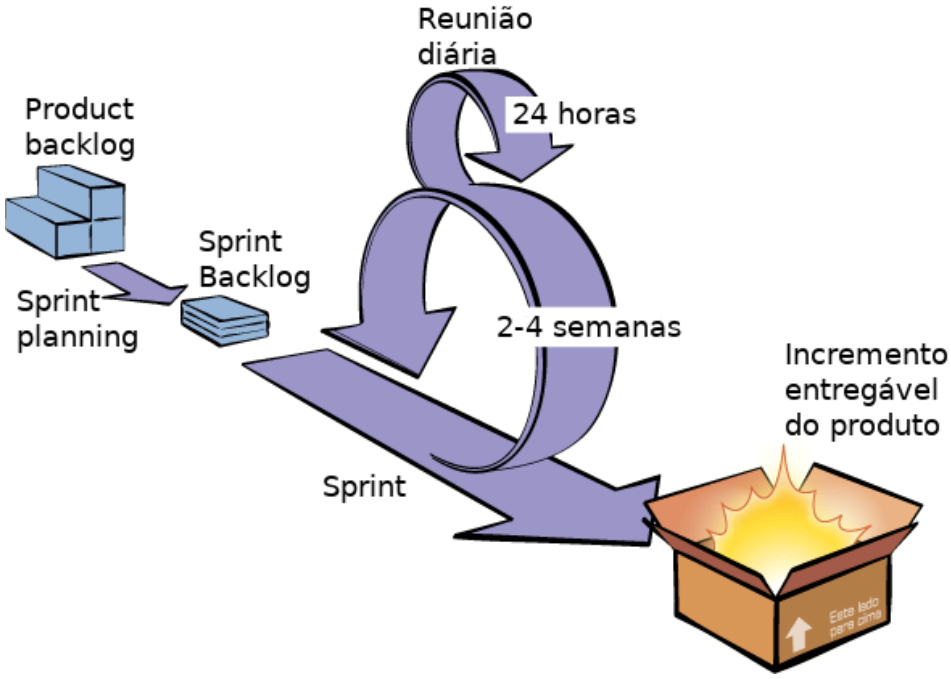
Segundo Schwaber e Sutherland (2013), o Scrum é composto por eventos, papéis e artefatos. São eventos do Srcum o Sprint, a reunião diária, e as cerimônias de planejamento, revisão e retrospectiva dos sprints. Durante estes eventos, os artefatos do Scrum são concebidos e/ou evoluídos. Os artefatos do Scrum são o Backlog do produto, o Backlog do Sprint, e os incrementos do produto. Descrições mais detalhadas destes artefatos e eventos são apresentadas a seguir, e a Figura 6 ilustra a relação entre eventos e artefatos no *framework*.

- *Sprint Planning*: Uma reunião na qual toda equipe está presente, bem como qualquer pessoa interessada que esteja representando a gerência ou o cliente. Durante a *Sprint Planning*, são descritas as funcionalidades de maior prioridade para a equipe. A equipe faz perguntas durante a reunião de modo que seja capaz de transformar as funcionalidades em tarefas técnicas, dando origem ao *Sprint Backlog*.

- *Daily* Scrum: A cada dia do Sprint, a equipe faz uma reunião diária, chamada *Daily* Scrum. Ela tem como objetivo disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho a ser realizado no dia que se *inicia.*

- *Sprint Retrospective*: O *Sprint Retrospective* ocorre ao final de um Sprint e serve para identificar o que funcionou bem, o que pode ser melhorado e que ações serão tomadas para melhorar.

Figura 6 - Ciclo de funcionamento do Scrum



Fonte: Do autor (2019)

- *Sprint Review*: Ao final de cada Sprint é feito a *Sprint Review*. Durante esta reunião, a equipe de desenvolvedores mostra o que foi alcançado durante o *Sprint* e o projeto é avaliado em relação aos objetivos determinados na *Sprint Planning*. Idealmente, a equipe completou cada um dos itens do *Product Backlog* trazidos para fazer parte do *Sprint*, mas o importante mesmo é que a equipe atinja o objetivo geral do *Sprint*.

- *Product Backlog*: O *Product Backlog* é uma lista contendo todas as funcionalidades desejadas para um produto. O conteúdo desta lista é definido pelo responsável do projeto. O *Product Backlog* não precisa estar completo no início de um projeto. Pode-se começar com tudo aquilo que é mais óbvio em um primeiro momento. Com o tempo, o *Product Backlog* cresce e muda à medida que se aprende mais sobre o produto e seus usuários.

- *Sprint Backlog*: O *Sprint Backlog* é uma lista de tarefas que o time se compromete a fazer em um *Sprint*. Os itens do *Sprint Backlog* são extraídos do *Product Backlog*, pela equipe, com base nas prioridades definidas pelo responsável do projeto e a percepção da equipe sobre o tempo que será necessário para completar as várias funcionalidades.

Para que tudo isso ocorra dentro do planejado, o Scrum conta com equipes, que segundo Schwaber e Sutherland (2017), são formadas por três papéis: *Product Owner, Scrum Master* e o Time de Desenvolvimento. O Product Owner, ou dono do produto, é o responsável por manter o *Product Backlog* atualizado e priorizado. O Time de Desenvolvimento (normalmente formado por 4 a 8 pessoas) é responsável por desenvolver os incrementos do produto. Uma característica importante dos Times de Desenvolvimento Scrum, é que estes são preferencialmente multidisciplinares e auto gerenciáveis. Finalmente, o Scrum *Master* procura assegurar que a equipe respeite e siga os valores e as práticas do Scrum.

As equipes Scrum são auto-organizadas e multifuncionais. As equipes auto-organizadas escolhem a melhor forma de realizar seu trabalho, em vez de serem direcionadas por outras pessoas fora da equipe (SCHWABER e SUTHERLAND, 2017).

4. ATIVIDADES DESENVOLVIDAS

Neste capítulo estão descritas as atividades desenvolvidas pelo estagiário para alcançar os objetivos propostos no período de estágio. Estes objetivos, visavam a aplicação dos conhecimentos adquiridos pelo estagiário durante a graduação, além da aplicação dos métodos utilizados pela empresa, os quais auxiliaram o estagiário a resolver os problemas que a empresa propõe solucionar, como o monitoramento de áreas florestais para garantir a sua preservação.

Por já ter tido contato com as metodologias utilizadas na empresa, o estagiário não passou pelo período de treinamento, que geralmente é oferecido para novos colaboradores. Este contato foi proporcionado durante algumas disciplinas ofertadas no curso, além da atuação como bolsista em outra empresa que utilizava *frameworks* e linguagens similares com a que o estagiário utilizaria durante o período de estágio.

Atuando na área de desenvolvimento, o estagiário participou do desenvolvimento de projetos internos da empresa, antes de atuar nos projetos para clientes externos. Esses projetos internos, a saber, Sistema Gerencial e Sistema de Avaliação Qualitativa (SAQ 360), são sistemas de uso próprio da organização.

Após o término do desenvolvimento dos projetos internos, o estagiário foi alocado em uma equipe para o desenvolvimento de um sistema denominado Protocolo Digital, sistema esse para o estado do Pará. Nesse sistema, o estagiário foi responsável pelo desenvolvimento do *front-end*, enquanto outros membros da equipe foram responsáveis pelo desenvolvimento *back-end*.

Após o término do contrato do sistema Protocolo Digital, o estagiário foi realocado para uma equipe de desenvolvimento de projetos para o estado do Amazonas. Nesta equipe, o estagiário participou do projeto Sistema de Fiscalização do Amazonas, onde auxiliou no desenvolvimento tanto do *front-end* quanto do *back-end*.

Para todos os projetos em que o estagiário participou, exceto os projetos internos, foi utilizada a metodologia ágil Scrum. Os colaboradores eram divididos em tribos (grupos formado por diversas equipes de desenvolvimento), e essas formadas por *squads* (equipes de desenvolvimento). Cada *squad* é composto por pelo menos 1 *Product Owner*, 1 analista de

qualidade e 1 desenvolvedor. Os squads em que o estagiário atuou eram compostos por 1 *Product Owner*, 1 analista de qualidade e de 3 a 4 desenvolvedores. Em todos os casos, o estagiário atuou como desenvolvedor.

Seguindo os ritos do Scrum, o estagiário participou de *squads* que realizavam as reuniões diárias sempre na parte da manhã, onde cada membro falava sobre o que havia feito no dia anterior e o que faria no decorrer do dia.

No início de cada *Sprint* era realizada a *Sprint planning,* onde determinava-se o que deveria ser desenvolvido durante o *Sprint*. Essa reunião durava cerca de 4 horas e nela as atividades a serem realizadas eram divididas e as estimativas eram, realizadas pela equipe. Os *Sprints* tinham duração de 10 dias úteis, ou duas semanas. Ao final de cada Sprint era realizada a *Sprint Review*, uma reunião para repassar o que foi desenvolvido e levantar pontos negativos e positivos da *Sprint*.

Durante os *Sprints*, o estagiário participava das atividades escolhendo aquelas que ficariam sob sua responsabilidade desenvolver. Essas atividades eram cadastradas e gerenciadas na ferramenta de gestão de projetos Taiga[8], utilizando um quadro kanban[9]. Na ferramenta, cada atividade poderia ter os seguintes estados:

- *To Do* (A fazer): Atividade a ser desenvolvida;
- *Doing* (Fazendo): Atividade em desenvolvimento;
- *Dev Done* (Desenvolvimento feito): Atividade que foi finalizada o seu desenvolvimento;
- *Testing* (Em teste): Atividade que está sendo testada;
- *Done* (Finalizada): Atividade que foi finalizada em todo seu desenvolvimento;
- *Cancelled* (Cancelada): Atividade que foi cancelada.

Todas as atividades desenvolvidas eram adicionadas ao repositório, a fim de manter o controle de versões dos projetos. O GitLab[10] era a ferramenta utilizada para o controle de versões. Finalizado o desenvolvimento de uma atividade, o código era adicionado em uma

[8] https://taiga.io/.
[9] Kanban é uma estratégia para otimizar o fluxo de valor para stakeholders através de um processo que utiliza um sistema visual que limita a quantidade de trabalho em andamento através de um sistema puxado. (Vacanti, 2018).
[10] https://about.gitlab.com/.

branch (do português, ramificação), que é a duplicação de um objeto sob controle de versão, para que as modificações possam ocorrer em paralelo ao longo de várias ramificações. Depois de testada a atividade, fazia-se o *merge* (união das *branches*) com o código principal do projeto, que ficava na *branch master*. Assim, o controle de versionamento era mais confiável, garantindo a qualidade do código e a prevenção contra erros.

O banco de dados utilizado em todos os projetos em que o estagiário participou era o PostgreSQL. O estagiário auxiliou na evolução do banco, conforme a necessidade da atividade que ele estava desenvolvendo. O estagiário era responsável por desenvolver as *evolutions* (arquivos SQL[11] utilizados para atualizar o banco de dados conforme a necessidade da atividade) e um profissional da área de banco de dados fazia a validação das mesmas e atualizava os bancos de teste, homologação e produção executando o arquivo SQL.

Nas subseções seguintes são descritos os sistemas que o estagiário atuou como desenvolvedor, a saber: Sistema Gerencial, Sistema de Avaliação Qualitativa**,** Protocolo Digital e Sistema de Fiscalização do Amazonas.

4.1. **Sistema Gerencial**

O Sistema Gerencial é utilizado para gerenciar colaboradores, equipes, salas e pode ser configurado quanto a cargos e funções dos colaboradores. Armazenando informações úteis, ele tem seu funcionamento baseado em um *CRUD* (*Create, Read, Update, Delete*), onde a gerência de informações importantes para a empresa mantém o sistema atualizado.

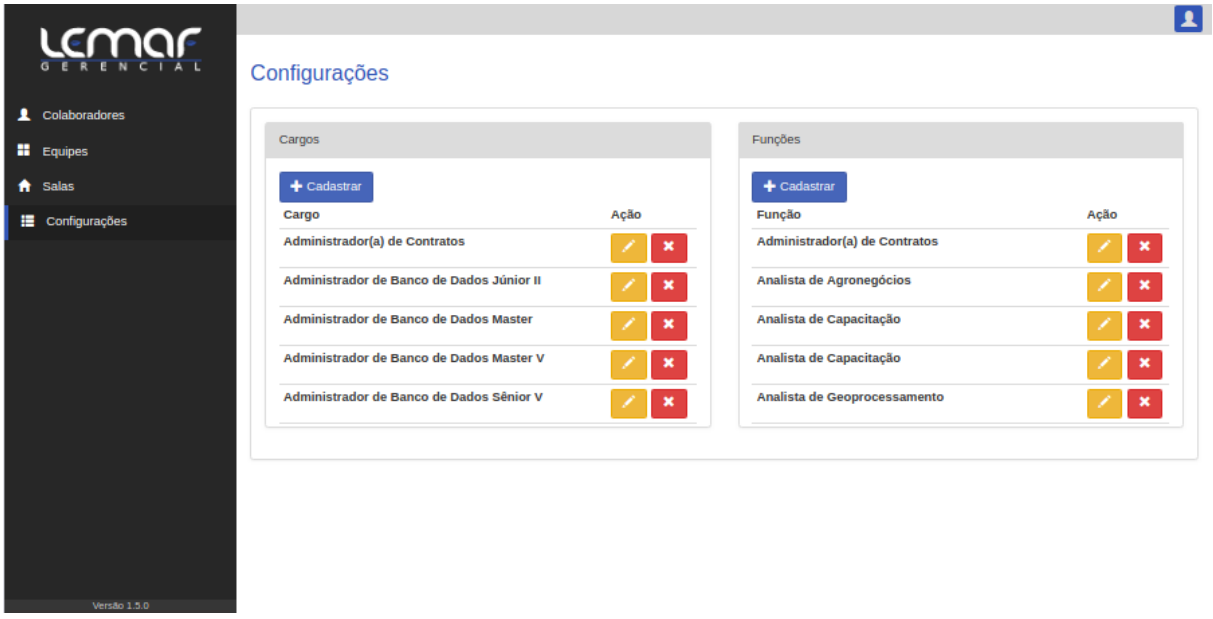
Para desenvolvimento do Sistema Gerencial não foi utilizada a metodologia Scrum. Neste projeto, as atividades eram divididas entre os membros da equipe, e a equipe se auto-organizava para desenvolvê-las. Para o desenvolvimento do *back-end* foi utilizado o Java 1.7 com o padrão de projetos MVC (*Model-View-Controller*) junto ao *Play Framework* e a versão 9.6 do PostgreSQL. Para o desenvolvimento do *front-end* foi utilizado o *framework* Angular e os pré-processadores de HTML e CSS, Pug e Sass.

Como o aluno estava no começo do estágio, e em fase de aprendizado, ele ficou responsável por corrigir erros e fazer melhorias na aplicação, conforme era solicitado pela

[11] Structured Query Language, ou Linguagem de Consulta Estruturada, é a linguagem de pesquisa declarativa padrão para banco de dados relacional.

equipe da gerência que utiliza o sistema. Posteriormente, o estagiário apoiou no desenvolvimento do menu configurações, onde implementou no front-end a listagem de cargos e funções. É possível visualizar essa listagem a partir da Figura 7, onde é mostrado os cargos e funções cadastrados bem como seus botões de cadastro, o qual está posicionado acima da listagem, edição, na cor amarela e exclusão, na cor vermelha.

Figura 7 - Página de Configurações (Sistema Gerencial)



Fonte: Do autor (2019)

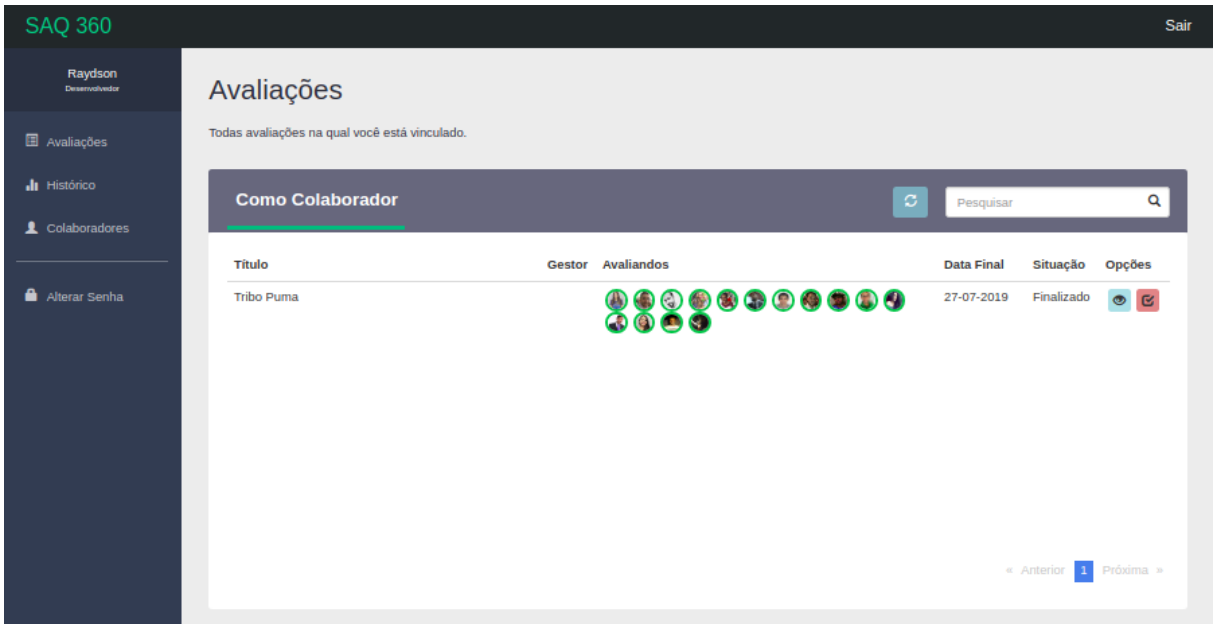
4.2. **Sistema de Avaliação Qualitativa**

O Sistema de Avaliação Qualitativa (SAQ 360) é um sistema usado no LEMAF para avaliação dos colaboradores. Através de questionários, os membros da organização se auto avaliam de acordo com a equipe em que estão alocados. A avaliação é composta por perguntas relacionadas a diversas áreas, como trabalho em equipe, qualidade do trabalho, concentração/foco, proatividade, capacidade de resolver problemas e comprometimento.

A Figura 8 mostra a tela inicial do SAQ 360, onde podemos ver uma listagem de avaliações, junto com a foto de cada colaborador que deve ser avaliado pelo usuário que está ativo no sistema. O colaborador pode visualizar o seu resultado pessoal, o histórico das últimas avaliações e seus resultados, todos os colaboradores da empresa, além de outras funcionalidades.

Para desenvolvimento do SAQ 360, também não foi utilizado o Scrum. Para o desenvolvimento desse sistema foi utilizado, para o *back-end*, Java 1.8, *Play Framework* 2 e a versão 9.6 do PostgreSQL. Para o *front-end* foi utilizado o *framework* AngularJS junto com o *TypeScript*, HTML e CSS.

Figura 8 - Página inicial (SAQ 360)



Fonte: Do autor (2019)

Assim como no Sistema Gerencial, por trabalhar em paralelo nele e no SAQ 360, o aluno foi responsável pelos erros e melhorias que eram relatados. Por estar no início do estágio, ele não chegou a desenvolver novas funcionalidades, somente deu suporte nas duas aplicações, corrigindo erros frequentes como, mudança de *labels,* alteração de lógica para habilitar ou desabilitar funções, erro ao salvar informações, entre outros.

4.3. **Protocolo Digital**

O Protocolo Digital é um sistema cujo objetivo principal é a gerência e criação de processos e protocolos possibilitando a tramitação dos mesmos pelo(s) Órgão(s) / Setor(es) do estado do Pará. O sistema faz integração com sistema Entrada Única[12] por meio do *login* do usuário e do acesso ao perfil selecionado. Todos os processos criados no sistema possuem um

[12] Sistema que faz o controle de acesso dos usuários em todos os sistemas desenvolvidos para o Pará.

identificador único, dados do processo, empreendimento vinculado, histórico de tramitação e dados dos protocolos criados nesse processo.

Por ser um projeto novo, as linguagens e *frameworks* utilizados para o desenvolvimento foram baseadas no conhecimento que a equipe possuía. Dessa forma, escolheu-se as tecnologias mais atuais no mercado, como *Vue.Js* e *play Framework* 2. Como o estagiário era responsável pelo *front-end*, exigia-se uma capacitação maior no *framework Vue.Js*, pois o mesmo havia trabalho somente com *AngularJS*, o qual possui o mesmo objetivo do *VueJs*. Para isso foram disponibilizados cursos para o estagiário, para que o mesmo se capacita-se e pudesse desempenhar seu papel desenvolvendo o *front-end*.

4.3.1. **Página Inicial**

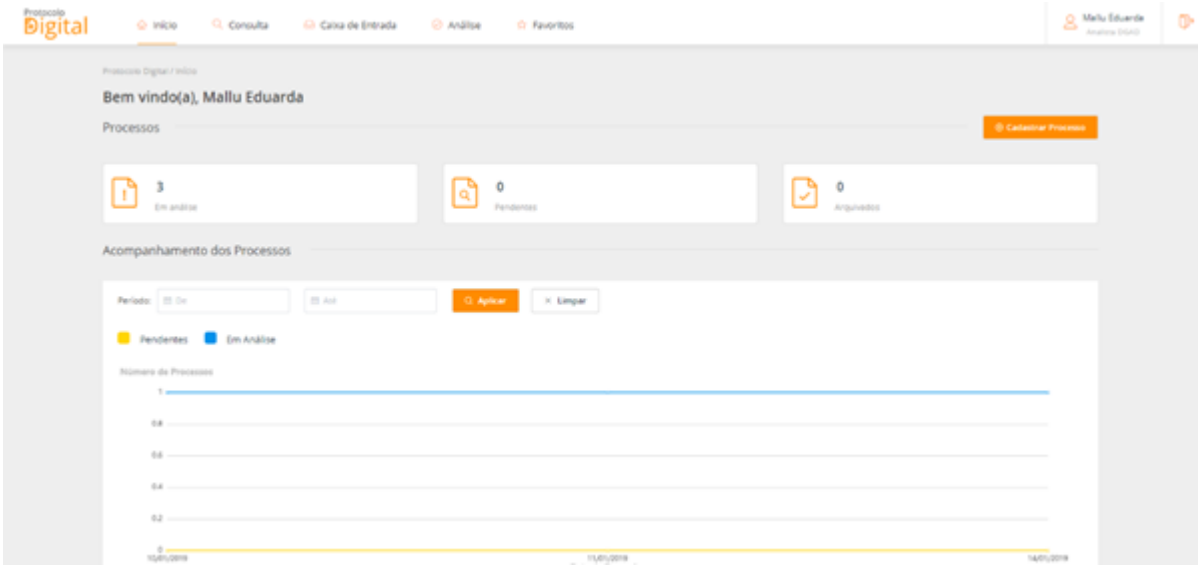
No início do projeto o estagiário foi responsável por estruturar a página inicial do sistema. Através da página inicial, como pode ser visto na Figura 9, pode-se visualizar os componentes que apresentam informações dos processos nas seguintes situações: Pendentes de análise, Em análise e Arquivados pelo usuário. Pode-se observar, também, o acompanhamento dos processos e o botão que disponibiliza o cadastro de novos processos.

Para o desenvolvimento da página inicial foi utilizado *Vue.Js,* como informado anteriormente, devido ao fato do mesmo oferecer um bom suporte a componentização, o que facilita o desenvolvimento, pois aumenta a capacidade de reaproveitamento dos componentes em diversas partes. Além disso, foi utilizada a biblioteca *Element.io*, pois a mesma oferece vários componentes pré estruturados, dando um visual harmônico ao projeto. A biblioteca teve sua criação baseada na versão 2.0 do *Vue.Js*, por isso quando usado em projetos desse *Framework* possui um suporte e retorno muito bom, agilizando o desenvolvimento.

Para a parte do gráfico apresentada na Figura 9, utilizou-se uma biblioteca que também é baseada na versão 2.0 do *VueJs*. Essa biblioteca, chamada *v-charts*[13], oferece facilidade quando se trata de gráficos, seja ele de linha, pizza, barra, etc. Para sua utilização, é preciso informar uma pequena quantidade de atributos, como cores, dados, e outras variáveis, para o gráfico aparecer na tela.

[13] https://v-charts.js.org/#/en/

Figura 9 - Tela Início (Protocolo Digital)



Fonte - Do autor (2019)

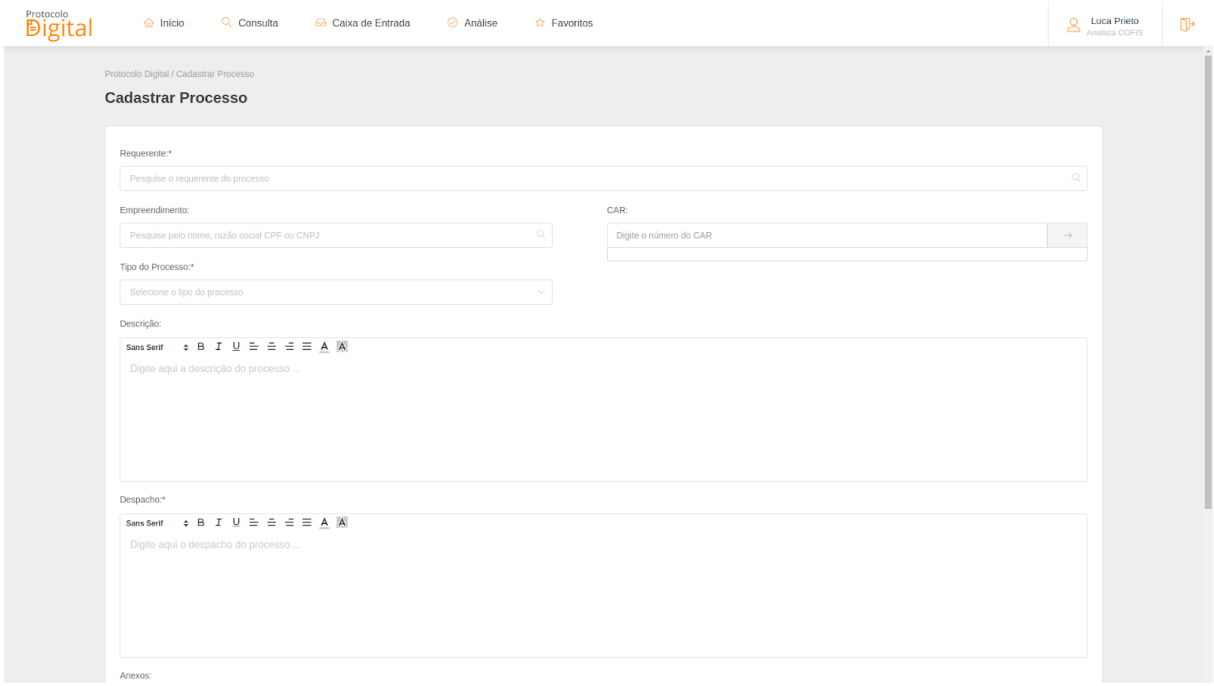
4.3.2. **Cadastrar Processo**

A tela "cadastrar processo" também foi desenvolvida pelo estagiário. A tela cadastrar processo apresentada nas figuras Figura 10 e Figura 11, permite ao usuário cadastrar um processo por vez. Para isso, foi desenvolvido um formulário que contava com elementos encontrados no HTML de texto e seleção (*input* e *select)*, componentes de adição de CAR[14] (Cadastro Ambiental Rural), campos de texto com editor e upload de arquivos.

Para o desenvolvimento do componente de adição de CAR foi necessário a utilização de diversos componentes, tanto do *Element.io* quanto elementos básicos disponibilizados pela linguagem HTML, como campo de texto *input*, botão *button*, e a área onde ficariam os cards adicionados *div*. Os cards adicionados que contém o número do CAR, foi retirado do *Element.io*, e a partir do componente lá disponibilizado foi feita uma alteração para se adequar ao que se necessitava. Além de fazer validação com o número informado, através de uma rota na API do CAR Federal[15], onde é verificado se o número é válido.

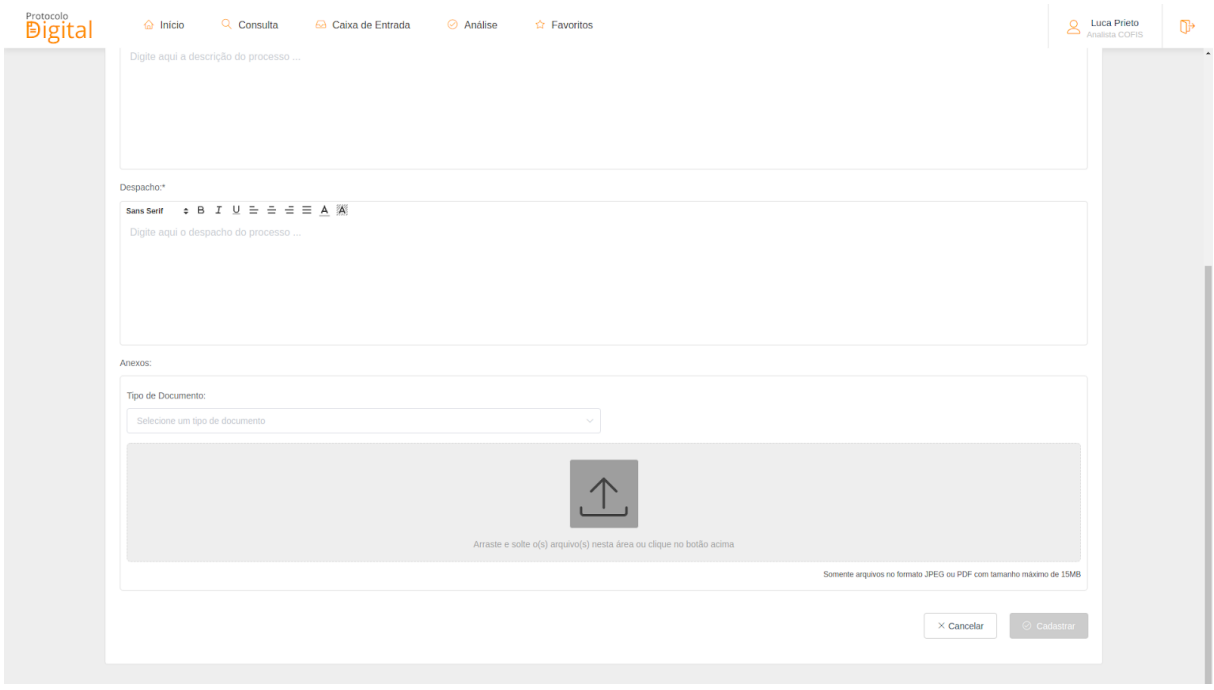
[14] Componente de preenchimento não obrigatório, utilizado no cadastro de processos, indicando se determinado processo que estava sendo cadastrado possuía alguma área rural, a qual já havia seu cadastro no sistema do CAR, vinculada.
[15] http://www.car.gov.br/#/https:/

Figura 10 - Tela Cadastrar Processo (Parte Superior)



Fonte: Do autor (2019)

Figura 11 - Tela Cadastrar Processo (Parte Inferior)



Fonte: Do autor (2019)

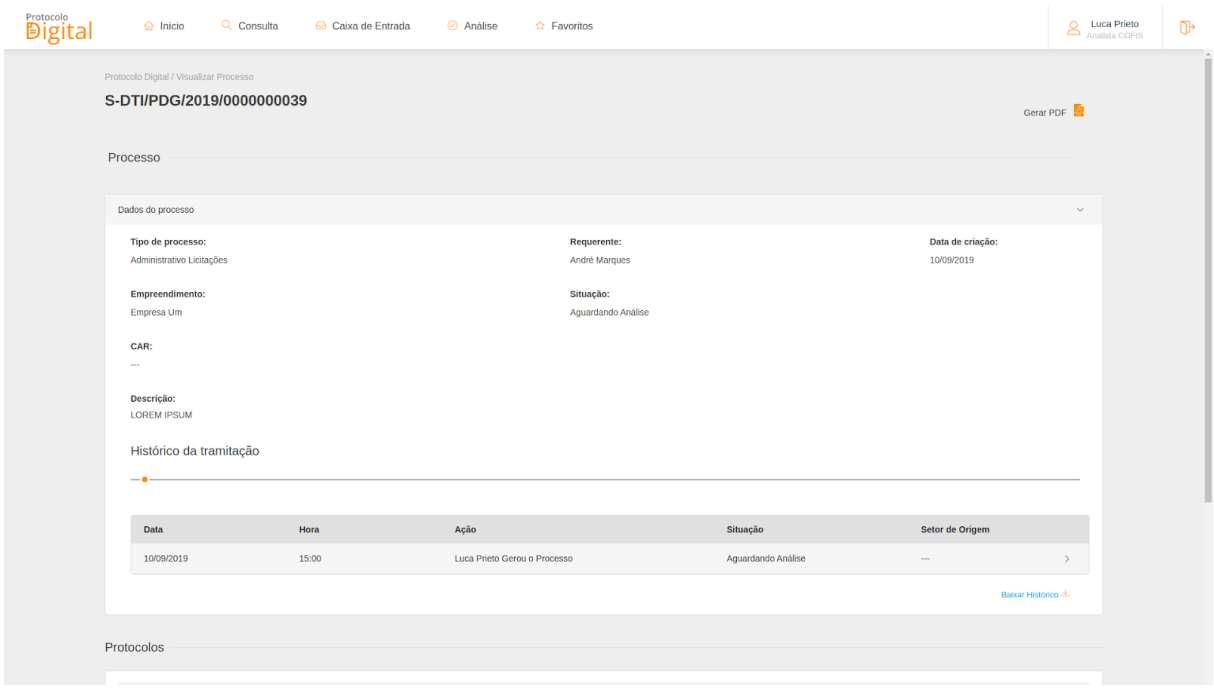
Para os campos de texto com edição, foi utilizado um componente próprio para *VueJs*, a saber *Vue2Editor*[16], que proporciona a edição de textos, podendo adicionar palavra em negrito, sublinhadas, itálico, entre outras funções para edição de textos.

Para o *upload* de arquivos na tela de cadastro de processos, utilizou-se um componente disponibilizado no *Element.io*. Esse componente permite o upload de arquivos, sendo fácil sua configuração para restringir pela extensão os arquivos que serão aceitos e o tamanho máximo dos mesmos.

4.3.3. **Visualizar Processo**

É possível visualizar na Figura 12 o processo cadastrado em algumas partes do sistema. Os processos possuem um identificador que é único, o qual é composto por: sigla do órgão/ sigla do sistema origem/ ano e um número de dez dígitos. Ao realizar o cadastro, o usuário consegue fazer o *download* do processo no formato pdf cadastrado.

Figura 12 - Visualizar processo



Fonte: Do autor (2019)

[16] https://www.vue2editor.com/

Conforme pode ser mostrado na Figura 12, a visualização do processo pode ser realizada em duas partes, sendo elas:

- Processo
    - Dados do processo: exibe os Dados do Processo inseridos no momento do cadastro, sendo eles: tipo de processo, requerente, data de criação, empreendimento, CAR, situação, descrição. O campo situação varia de acordo com a condição do processo atualmente, na tramitação do processo. Esses dados são fundamentais para a criação do processo.
    - Histórico de tramitação: Exibe o histórico de tramitação do processo, mostrando as ações realizadas no processo em visualização. Para isso, podemos visualizar uma linha do tempo das ações e o histórico de tramitação. Na linha do tempo são exibidos pontos na cor laranja e, ao posicionar o mouse em cima do mesmo, é exibida a data e a ação realizada naquela data..
    - É possível fazer *download* do histórico de tramitação do processo em visualização, em um arquivo no formato pdf.
- Protocolos
    - Os protocolos são criados ao realizar o cadastro de um processo ou na tramitação do mesmo, além da opção de ser adicionado de modo avulso. O identificador único do protocolo é composto pelos dados do tipo do protocolo, subtipo do protocolo, situação, despacho e anexos.

Na tela de visualizar processos foram utilizados componentes disponibilizados também no *Element.io*, sendo eles: o *collapse*, componente que é utilizado para mostrar e esconder conteúdos e o *table,* onde pode-se exibir vários dados com formato semelhante. No *table* há a possibilidade de classificar, filtrar e comparar seus dados. Para o histórico de tramitação foi utilizado *JavaScript* puro junto com CSS e HTML para dar a forma, onde a distância das datas de salvas no histórico e que são exibidas, é dada pelo cálculo apresentado na Figura 13.

Figura 13 - Método de cálculo do histórico de tramitação.

```
1.  montarTimeLine () {
2.      let dias = JSON.parse(JSON.stringify(this.processo.historicos));
3.      const historico = dias[0];
4.      if (historico.emEdicao) {
5.          dias.splice(0, 1);
6.      }
7.      if (dias.length === 1) {
8.          let element = document.getElementById('dot0');
9.          element.style.marginLeft = 1 + '%';
10.     } else {
11.         let numPontos = dias.length;
12.         let porcento = 100 - ((numPontos + 1) * 2);
13.         let diferencaTotal = dias[0].dataDoCadastro - dias[numPontos - 1]
            .dataDoCadastro;
14.         for (let index = dias.length - 1; index >= 0; index--) {
15.             if (index !== dias.length - 1) {
16.                 let diferenca = dias[index].dataDoCadastro - dias[index + 1]
                    .dataDoCadastro;
17.                 let x = (porcento * diferenca) / diferencaTotal;
18.                 document.getElementById('dot' + (dias.length - 1 - index))
                    .style.marginLeft = x + '%';
19.             }
20.         }
21.     }
22. }
```

Fonte: Do autor (2019)

Como exibido na Figura 13, o código pega os dias na linha 2, a partir da lista de históricos de um processo e entre as linhas 3 e 6, verifica se o histórico mais atual está em edição, se sim, ele o remove da lista. Caso exista somente um histórico do processo, entre as

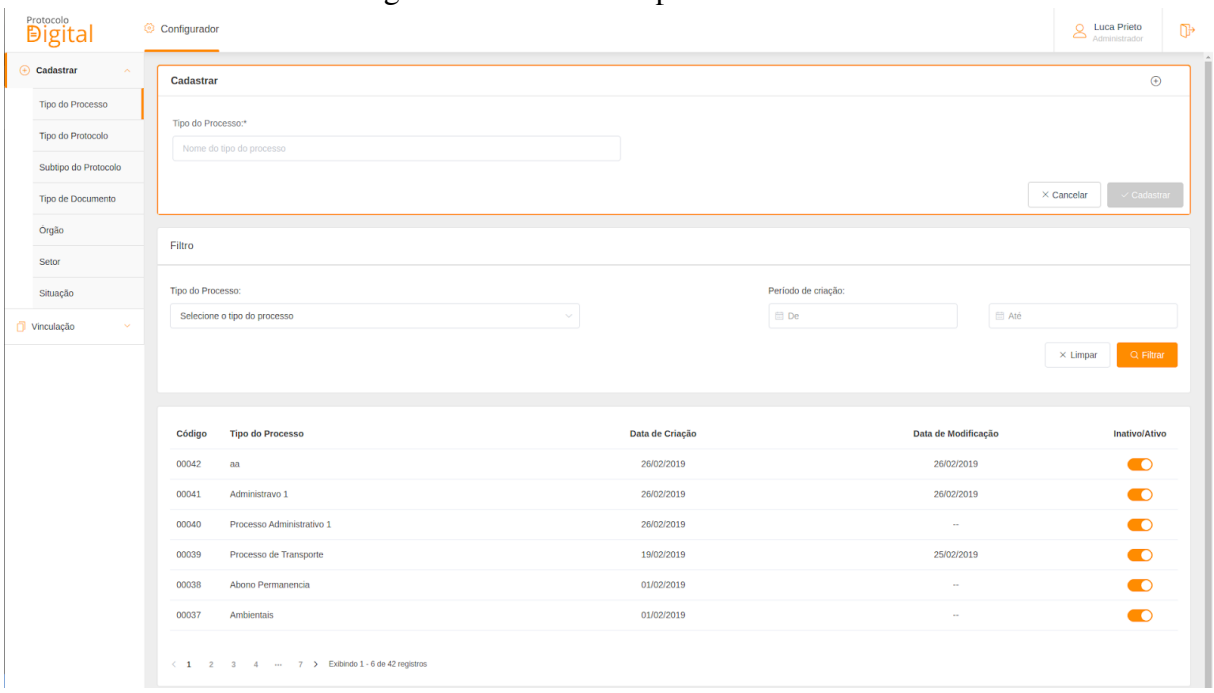
linhas 7 e 9, o código posiciona somente uma bolinha laranja na *TimeLine*. Existindo vários históricos do processo o código pega, na linha 11, a quantidade de pontos conforme a quantidade de dias existentes, na linha 12, a porcentagem de distância conforme a quantidade de pontos, e na linha 13, a diferença baseada na data entre o primeiro e último ponto. O *for* da linha 14, percorre cada ponto calculando a distância em que as bolinhas laranjas (apresentadas na Figura 12) terão uma das outras, representando uma *TimeLine* do ciclo de vida do processo o qual está sendo visualizado.

4.3.4. **Módulo Configurador**

Dentro do Protocolo Digital existe um módulo denominado “Configurador”, o qual pode ser acessado através do perfil de Administrador do sistema, dando permissão de acesso para o usuário ao menu configurador. Por meio deste módulo é possível efetuar cadastros e vinculações para o funcionamento ideal e customizado do Protocolo Digital.

O administrador pode efetuar o cadastro de vários itens (Tipo do Processo, Tipo do Protocolo, Subtipo do Protocolo, Tipo de Documento, Órgão, Setor e Situação), conforme mostrado na Figura 14. Como pode ser visualizado, no menu lateral esquerdo, existem várias opções de cadastro, além das opções de filtragem e listagem dos itens cadastrados.

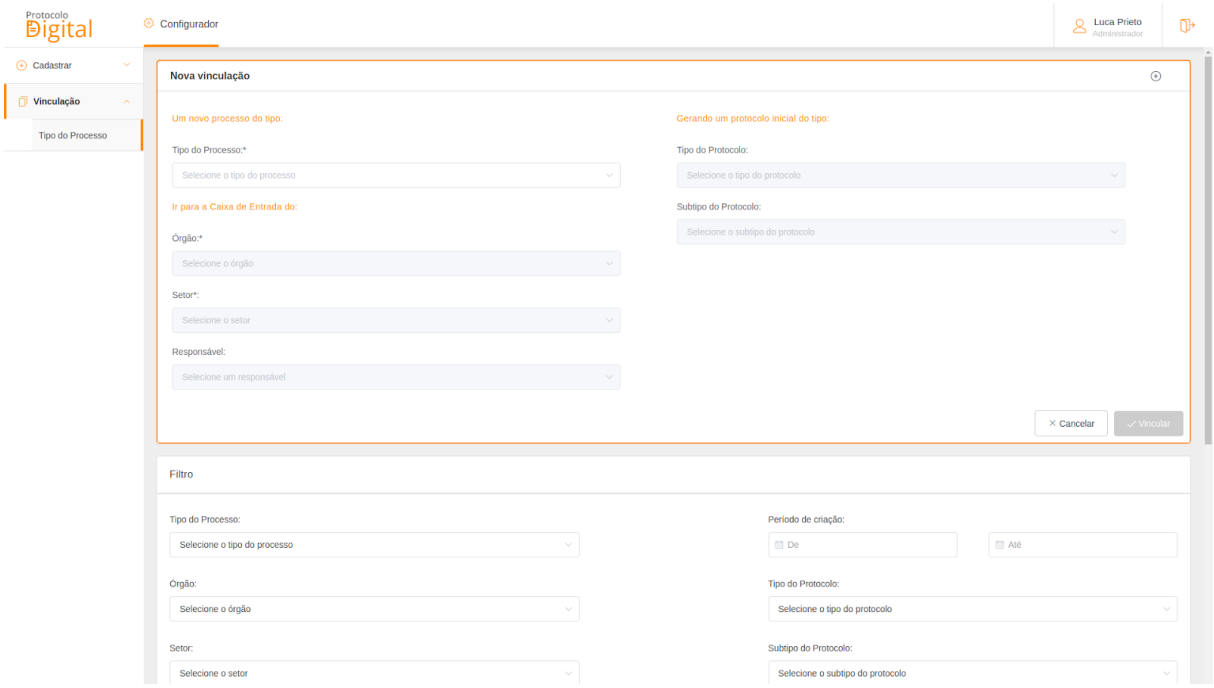
Figura 14 - Cadastrar tipo do Processo



Fonte: Do autor (2019)

Após o cadastro, o Administrador do sistema tem a opção de vincular os itens recém cadastrados, fazendo com que haja uma ligação entre esses itens e, com isso, os demais perfis consigam cadastrar os processos e protocolos selecionando esses itens. Essa vinculação acontece conforme é mostrado na Figura 15, onde são selecionados os itens desejados para uma vinculação.

Figura 15 - Vinculação de Tipo de Processo



Fonte: Do autor (2019)

Na implementação deste módulo foram usadas as mesmas tecnologias e bibliotecas das telas anteriores, citadas nessa seção (4.3). Para o menu lateral foi utilizado o componente *tab* da biblioteca *Element.io*. Para o componente de cadastro foi utilizado o componente *collapse,* com uma pequena alteração no ícone que fica a direita usando CSS. Além disso, foi utilizado o componente de data *date picker,* também do *Element.io*.

Finalizadas as atividades referentes ao Protocolo Digital, o estagiário foi alocado na equipe de desenvolvimento do Fiscalização Ambiental do Amazonas.

4.4. **Fiscalização Ambiental do Amazonas**

O sistema de Fiscalização do Amazonas tem como objetivo principal gerenciar de forma digital e integrada as fases de demanda, planejamento e operação das fiscalizações no

estado. Os dados usados no sistema são georreferenciados, possibilitando o rastreamento das informações e permitindo o controle e monitoramento de forma efetiva.

Para este projeto, foi utilizado no *front-end,* o *AngularJs* na versão 1.5 junto com Sass e Pug, além de *frameworks* e bibliotecas de auxílio. Para o *back-end* foi utilizado Java 1.8 junto com *Play Framework* 1.5.2. Diferente da atuação no Protocolo Digital, neste projeto, o estagiário desenvolveu atividades relacionadas não só ao *front-end*, mas também ao *back-end* do sistema.

O sistema Fiscalização do Amazonas foi iniciado a partir de um *fork*[17] do sistema Fiscalização do Pará. Portanto, coube ao time de desenvolvimento customizá-lo para se adequar às necessidades do Amazonas. Esse trabalho não foi trivial, , uma vez que as leis são diferentes entre os estados, sendo que foi necessário uma customização totalmente diferente entre os sistemas.

A adequação do sistema para o estado do Amazonas foi realizada em 03 meses. No início do projeto, os *sprints* tinham como objetivo correções e refatoração de código. Neste período, o estagiário realizou diversas atividades de alterações de *labels* e imagens referente ao estado, relacionadas ao *front-end.*

Após a adequação inicial do projeto Fiscalização para atender as necessidades do estado do Amazonas, foi necessário criar um novo menu para habilitar a opção de cadastro de um relatório denominado “Relatório Técnico de Operação”. Esta atividade foi alocada para o estagiário.

4.4.1. **Relatório Técnico de Operação**

O Relatório Técnico de Operação tem o objetivo de descrever as atividades que serão desenvolvidas em uma determinada operação fiscal, a qual é preciso o cadastro de informações como: objetivos, agentes de fiscalização, equipe de apoio, localização geográfica, anexo de termos, e outras informações necessárias para compor o relatório. As Figuras 16, 17, 18 e 19 apresentam o formato do Relatório Técnico de Operação.

[17] Na engenharia de software, um *fork* (bifurcação ou ramificação) acontece quando um desenvolvedor inicia um projeto independente com base no código de um projeto já existente.

O cadastro dessas informações cadastro é dividido em 5 partes listadas a seguir, a saber:

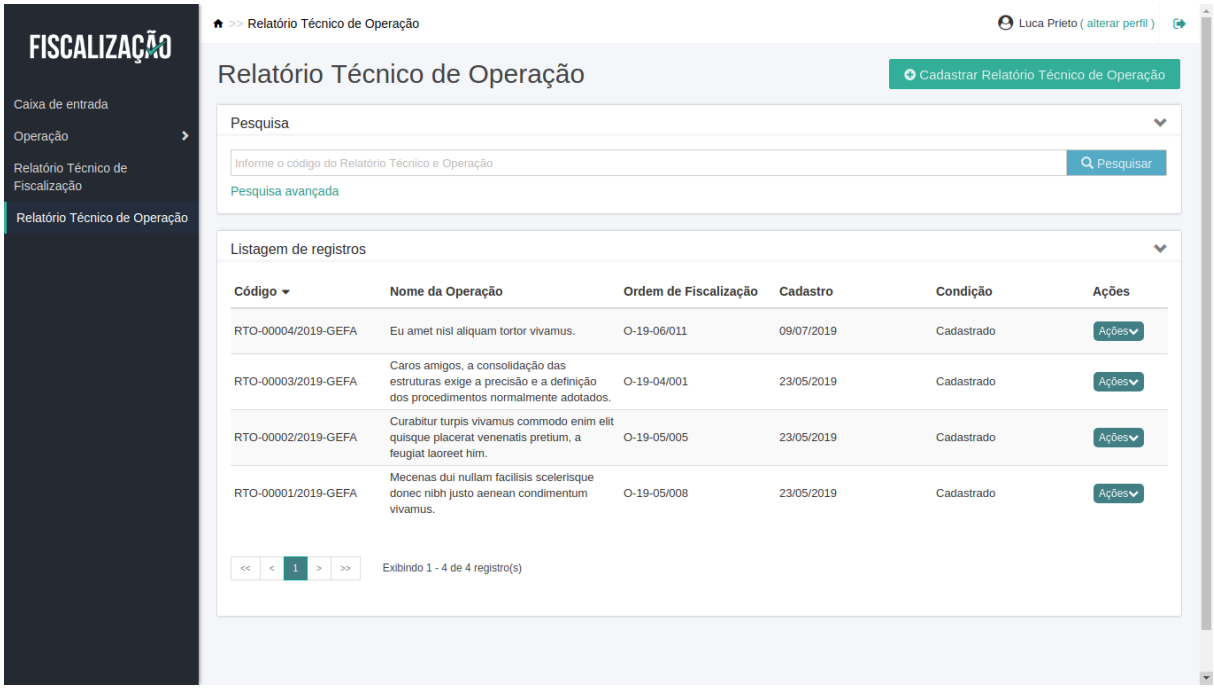
- Dados: a etapa dados é a parte onde os informações textuais do relatório são informadas;
- Localização;
- Auto de Infração e Termos;
- Arquivos;
- Resumo.

Para a implementação das telas do Relatório Técnico de Operação foram utilizadas as tecnologias citadas anteriormente na seção 4.3, trocando somente Vue.js por AngularJS, além do *framework* para mapas, *leaflet*[18], o qual é largamente utilizado na empresa em diversos projetos. Também foi utilizado componentes de *upload* de arquivos, para que os usuários do sistema pudessem anexar toda documentação necessária.

Observa-se que na tela inicial (Figura 16), do Relatório Técnico de Operação, há uma listagem dos relatórios cadastrados, mostrando algumas informações no *grid* de listagem, e acima a opção de filtro. Na Figura 17 pode-se observar uma parte do cadastro do relatório, avançando por cada etapa ao preencher os campos que são obrigatórios. Na Figura 18 é mostrado o mapa adicionado usando o framework *leaflet*. Por último, na Figura 19 observa-se a parte do cadastro onde é realizado o *upload* dos arquivos.

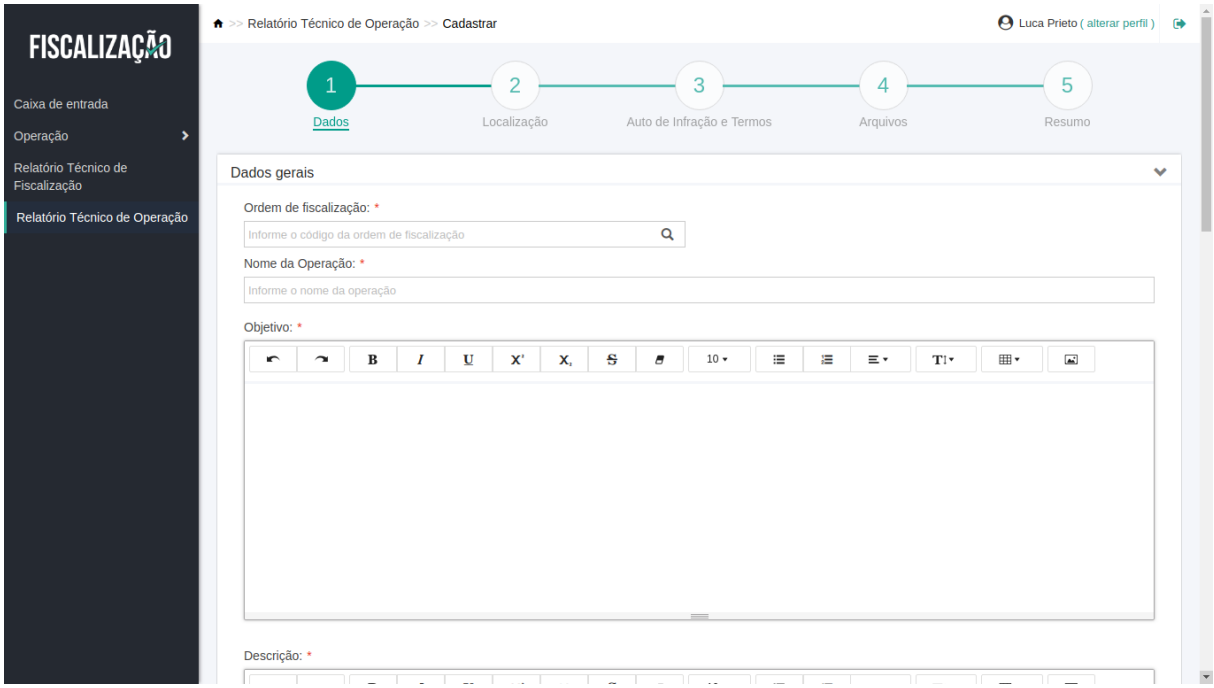
[18] https://leafletjs.com/

Figura 16 - Tela de listagem dos Relatórios Técnico de Operação



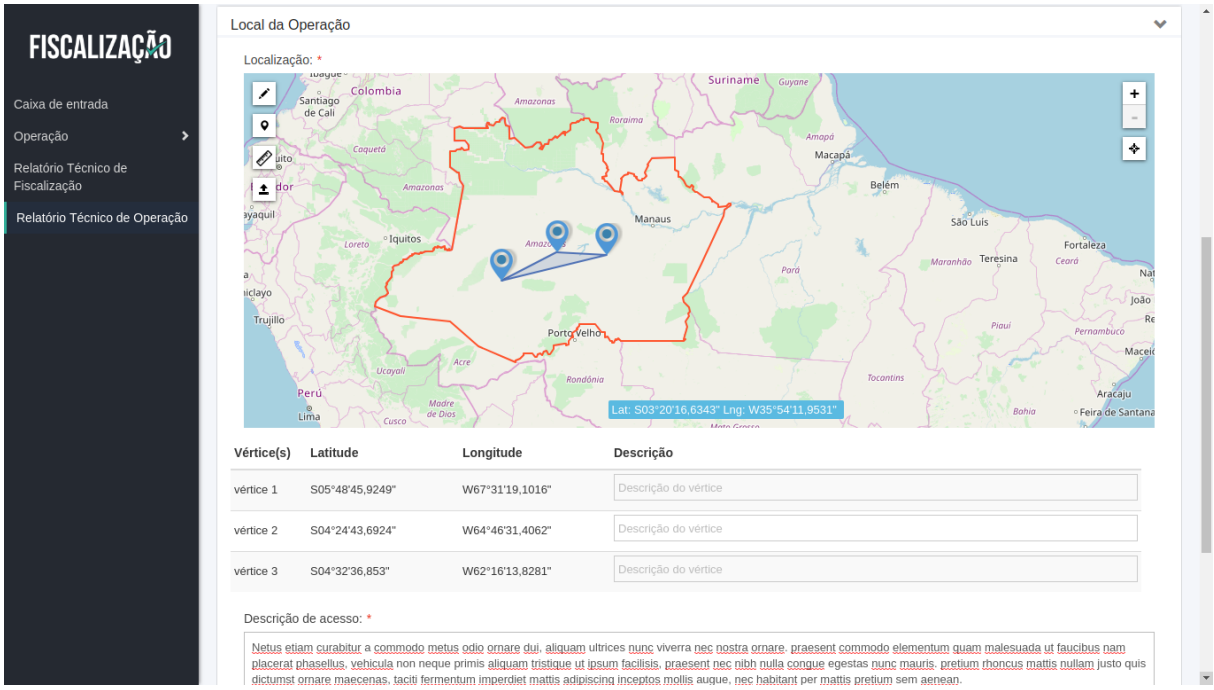
Fonte: Do autor (2019)

Figura 17 - Etapa dados no cadastro do Relatório Técnico de Operação.



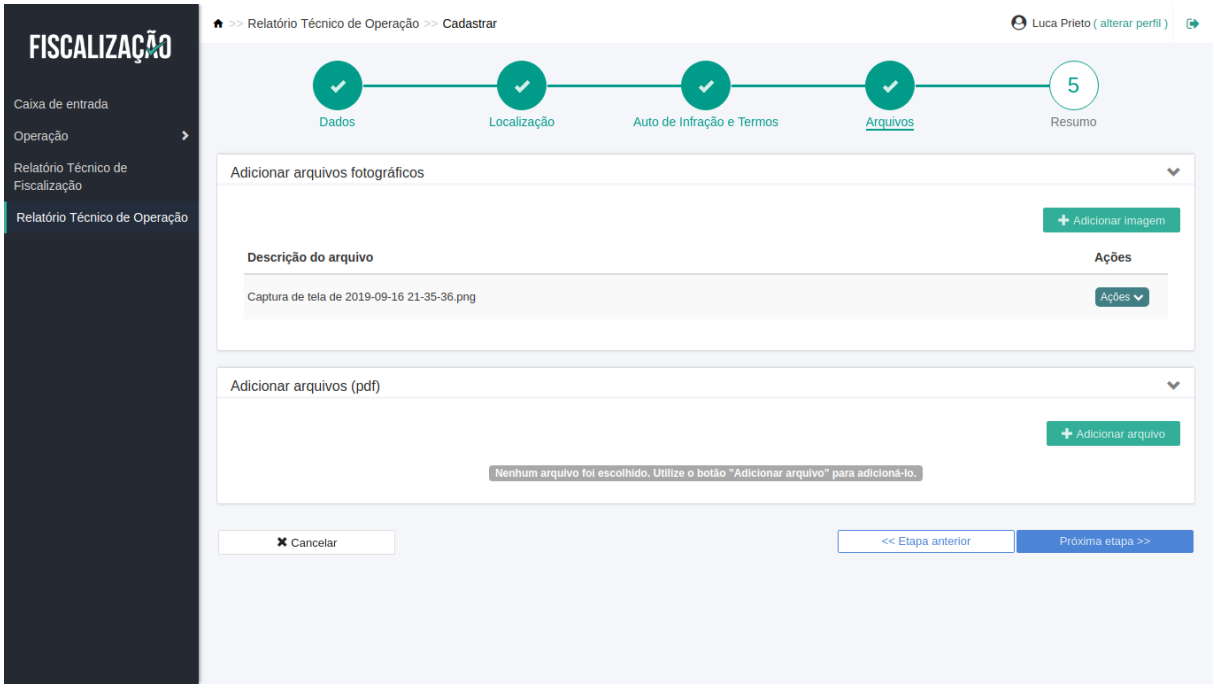
Fonte: Do autor (2019)

Figura 18 - Etapa localização no cadastro do Relatório Técnico de Operação.



Fonte: Do autor (2019)

Figura 19 - Etapa Arquivos no cadastro do Relatório Técnico de Operação.



Fonte: Do autor (2019)

5. CONSIDERAÇÕES FINAIS

Os sistemas internos, por serem de uso particular da empresa, não passaram pelo processo de homologação, que ocorre junto ao cliente. No entanto, estes sistemas se encontram hospedados no servidor interno da empresa e atualmente atende às necessidades dos que os usam, sendo solicitado eventualmente correções e ajustes.

O sistema Protocolo Digital passou pelo processo de homologação, validando o sistema junto ao cliente, porém, atualmente não se encontra em produção, pois o estado que o solicitou ainda não necessitou usá-lo. Portanto, o sistema está pronto para uso, esperando somente a solicitação para colocá-lo em produção.

Já o sistema Fiscalização, já está sendo utilizado pelo estado do Amazonas. O mesmo foi homologado junto ao cliente, e teve sua versão de produção gerada, estando disponível para que os órgãos responsáveis possam fazer a fiscalização junto a comunidade amazonense, monitorando a população e as florestas que as circundam.

O desenvolvimento de todos esses sistemas ao longo do período de estágio trouxe ao aluno experiências que ao longo de sua graduação, foram de suma importância para o aprendizado e aplicação em trabalhos. O estágio ofereceu uma vivência muito produtiva, onde pôde-se colocar em prática o que se aprendeu na graduação.

Dentre o que foi aprendido na graduação para se aplicar no estágio, pode se destacar matérias como, Estrutura de Dados, Práticas de Programação Orientada a Objetos, Engenharia de Software, Introdução a Sistemas de Banco de Dados, Sistemas Gerenciadores de Banco de Dados, Processos de Software, Sistemas Distribuídos, Gerência de Projetos de Software.

Diante disso, pode-se concluir que o estágio proporcionou um grande crescimento na carreira profissional do aluno, pois o mesmo agora carrega toda uma bagagem de vivência na prática sobre a área de desenvolvimento WEB, abrindo assim, muitas portas para que o mesmo possa decidir seu futuro. O estágio também proveu experiências que ajudou o aluno a tomar a decisão sobre qual carreira seguir dentro da área de tecnologia da informação.

REFERÊNCIAS

AGAFONKIN, V. **Leaflet**: an open-source JavaScript library for mobile-friendly interactive maps. 2019. Disponível em: <https://leafletjs.com/index.html>. Acesso em 27 de Outubro de 2019.

AQUINO JUNIOR; G. S. D. **Desenvolvimento de Sistemas Web em Java**: Frameworks, Padrões de Projeto e Diretrizes para a Camada de Apresentação. 2002. Disponível em: <https://repositorio.ufpe.br/bitstream/123456789/2573/1/arquivo5091_1.pdf>. Acesso em: 27 de Outubro de 2019.

BESSER, R. Cascading style sheets: Designing for the web. **Technical Communication**, Society for Technical Communication, v. 48, n. 3, p. 340–340, 2001.

CAMPIONE, M; WALRATH, K. **The Java Tutorial**: Object-Oriented Programming for the Internet. [S.I.]: SunSoft Press, 1996.

CATLIN, H. **sass.**: style with attitude. 2013. Disponível em: <https://web.archive.org/web/20130901145805/http://sass-lang.com/about.html>. Acesso em: 20 de Outubro de 2019.

DEITEL, H. M. **Java**: Como Programar. 6 ed. São Paulo: Pearson education do Brasil, 2005.

EIS, D.; FERREIRA, E. **HTML5 e CSS3 com farinha e pimenta.** São Paulo: Tableless, 2012. Disponível em: <https://docplayer.com.br/1431492-Html5-e-css3-com-farinha-e-pimenta-diego-eis-elcio-ferreira.html>. Acesso em: 26 Outubro de 2019.

FLANAGAN, D. **JavaScript**: The Definitive Guide. 2011. Disponível em: <https://pepa.holla.cz/wp-content/uploads/2016/08/JavaScript-The-Definitive-Guide-6th-Edition.pdf 2002>. Acesso em: 19 de Outubro de 2019.

FUNDEC, **Sobre a FUNDECC**. 2019. Disponível em: <http://www.fundecc.org.br/sobre-a-fundecc/>. Acesso em: 19 de Outubro de 2019.

GOOGLE. **What Is AngularJS?**. 2018. Disponível em: <https://docs.angularjs.org/guide/introduction>. Acesso em: 26 de Outubro de 2019.

LIGHTBEND. **Play Framework makes it easy to build web applications with Java & Scala**. 2018. Disponível em: <https://www.playframework.com/>. Acesso em: 19 de outubro de 2019.

MACRAE, C. ***Vue.js: Up and Running: Building Accessible and Performant Web Apps***. Sabastopol: O'Reilly Media, 2018. Disponível em: <http://calameo-download.tiny-tools.com/pages.php?doc_id=005635274f66716e886c0>. Acesso em: 28 de Outubro de 2019.

MOZZILA. **HTML**: Hypertext Markup Language. 2019. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTML>. Acesso em: 19 de Outubro de 2019.

POSTGRES. **O que é o PostgreSQL**. 2019. Disponível em <http://pgdocptbr.sourceforge.net/pg82/intro-whatis.html>. Acesso em: 26 de Outubro de 2019.

PUG. **Getting Started**. Disponível em: <https://pugjs.org/api/getting-started.html>. Acesso em: 19 de Outubro de 2019.

RÉ, R. **Um Processo para Construção de Frameworks a partir da Engenharia Reversa de Sistemas de Informação Baseados na Web: Aplicação ao Domínio dos Leilões Virtuais**. 2002. 143 f. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) - ICMC/USP, São Paulo, 2002.

REGADAS, A. **Getting started with Pug template engine**. 2016. Disponível em: <https://codeburst.io/getting-started-with-pug-template-engine-e49cfa291e33>. Acesso em: 20 de Outubro de 2019.

SCHWABER, K. **Agile Project Management with Scrum**. 2004. Disponível em: <https://pdfs.semanticscholar.org/43e7/ddf062e81253083e2a5b7c619f9bc0d42333.pdf>. Acesso em: 27 de Outubro de 2019.

SCHWABER, K.; SUTHERLAND, J. **Guia do Scrum**: Um guia definitivo para o Scrum: As regras do jogo. 2013. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em 27 de Outubro de 2019.

SCHWABER, K.; SUTHERLAND, J. **What is Scrum?**. 2018. Disponível em: <https://www.scrumguides.org/>. Acesso em 27 de Outubro de 2019.

VACANTI, D. **O Guia do Kanban para Times Scrum**. 2018. Disponível em: <http://andrelmgomes.com.br/artigosedocumentos/Andr%C3%A9%20Gomes%20-%20Guia%20Kanban%20para%20Times%20Scrum%202018.pdf>. Acesso em 5 de Novembro de 2019.

W3C. W3C: About W3C. 2019. Disponível em: <https://www.w3.org/Consortium/>. Acesso em: 19 Outubro de 2019.