GUSTAVO COSTA NEVES

ESTÁGIO EM DESENVOLVIMENTO DE SISTEMAS WEB NO LEMAF/UFLA

LAVRAS – MG

2019

GUSTAVO COSTA NEVES

ESTÁGIO EM DESENVOLVIMENTO DE SISTEMAS WEB NO LEMAF/UFLA

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

Profa. Marluce Rodrigues Pereira

Orientadora

LAVRAS – MG

2019

GUSTAVO COSTA NEVES

ESTÁGIO EM DESENVOLVIMENTO DE SISTEMAS WEB NO LEMAF/UFLA

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

APROVADA em 02 de Dezembro de 2019.

Profa. Marluce Rodrigues Pereira	UFLA
Prof. Rafael Serapilha Durelli	UFLA
Prof. Matheus de Andrade Fiausino	LEMAF

Marluce Rodrigues Pereira
Profa. Marluce Rodrigues Pereira
Orientadora

LAVRAS – MG
2019

Dedico esse trabalho de conclusão de curso aos meus pais que sempre me apoiaram em toda essa trajetória.

AGRADECIMENTOS

Agradeço a todos que estiveram contribuindo comigo durante esse período de UFLA.

Agradecimentos especiais para:

Bruna Gomes.

Danilo Guarizzo.

Ivan Carvalho.

Bruno Custódio.

Highlander Silva.

Matheus Flausino.

Amanda Lima.

Todos funcionários do LEMAF com quem trabalhei.

Aos professores que me tutoriaram durante a graduação.

RESUMO

Este relatório de estágio descreve as atividades desenvolvidas no Laboratório de Projetos e Estudos em Manejo Florestal –LEMAF, com duração de 1 ano e com atuação em vários projetos: SICAR, PRA, Consulta Pública, SEIRH-CMS, SIGERH, entre outros. O objetivo principal foi atuar no desenvolvimento de sistemas e ferramentas de software para controle e manejo ambiental. O relatório visa descrever os processos de desenvolvimento dentro da empresa, como metodologias ágeis, estágios de desenvolvimento (prototipação, design system, desenvolvimento, homologação) e organização no geral. As atividades desenvolvidas permitiram que o estagiário aplicasse os conhecimentos adquiridos no curso de graduação, e aprendesse a utilizar novas ferramentas e processos, bem como trabalhar em equipe para gerar um produto.

Palavras-chave: Processos agéis, Desenvolvimento *web*, *Frameworks*.

ABSTRACT

This internship report describes how activities are carried out in the Laboratory of Forest Management Studies and Projects - LEMAF, lasting one year and working in various projects: SICAR, PRA, Consulta Pública, SEIRH-CMS, SIGERH, among others. The main objective was to perform in the development of systems and software tools for environmental control and management. The report aims to describe the development processes within the company, such as methodologies, development steps (prototyping, design system, development, approval) and overall organization. As the activities performed allow the trainee to apply the knowledge acquired in the undergraduate course, learn to use new tools and processes, and work as a team to generate a product.

Keywords: Agile Processes, *Web* Development, *Frameworks*.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Este trabalho visa apresentar as experiências vividas durante o período como estagiário do LEMAF (Laboratório de Projetos e Estudos em Manejo Florestal), laboratório situado dentro da Universidade Federal de Lavras (UFLA) onde são desenvolvidas soluções tecnológicas relacionadas a manejo florestal. O laboratório foi fundado em 2004 e, desde lá, desenvolve diversos projetos para órgãos governamentais e empresas privadas, contando em 2018 com mais de 160 funcionários.

Os projetos do LEMAF geralmente possuem contextos relacionados à preservação ambiental, como monitoramento de áreas desmatadas e uso indevido de recursos hídricos. Porém sua carteira de projetos inclui diversos temas, como portais para povos indígenas, monitoramento de número de animais atropelados em determinada região e até mesmo gerenciadores de conteúdos. Durante o estágio era exigido certas responsabilidades, tais como: seguimento de metodologias ágeis, organização, trabalho em equipe e, principalmente, o desenvolvimento de projetos *web*.

Por conta de uma grande rotação de projetos e times, foi necessário o estudo de diversos *frameworks* e tecnologias, onde os principais relacionados com *frontend* foram Angular, Vuejs e ReactJS, enquanto com *backend* foram SpringBoot, DotNet *Framework* e Play*Framework*. Para que o desenvolvimento dos projetos fluíssem efetivamente, eram utilizadas diversas metodologias ágeis, como principais *Scrum* e *Kanban*.

Com toda essa carga de conhecimento e responsabilidades, ser estagiário no LEMAF se tornou uma experiência prazerosa. Estava sempre evoluindo, conhecendo novas tecnologias do mercado e tendo oportunidade de obter conhecimento de pessoas extremamente experientes no ramo.

Os demais capítulos deste relatório estão organizados da seguinte forma: No Capítulo 2 são abordadas as ferramentas e processos utilizados durante o estágio, no Capítulo 3 são apresentadas as atividades desenvolvidas durante o estágio e no Capítulo 4 são apresentadas conclusões.

2 METODOLOGIAS E FERRAMENTAS

Para o desenvolvimento dos projetos foram utilizadas metodologias ágeis de desenvolvimento e ferramentas para desenvolvimento *web* (*backend* e *frontend*). Para entender melhor essas tecnologias as seções seguintes abordam cada uma delas.

2.1 Metodologias Ágeis

O "Manifesto ágil"surgiu em 2001 quando 17 conhecedores de metodologias ágeis se reuniram e discutiram como deveria ser o processo de desenvolvimento de software com metodologias ágeis, elencando 12 princípios que priorizam principalmente a satisfação do cliente, o trabalho em equipe, entregas de software mais rápidas e colaboração com o cliente. - (MANIFESTO. . . ,)

As metodologias ágeis são estratégias e comportamentos que devem ser seguidos afim de obter um resultado otimizado e mensurável. As principais metodologias ou *frameworks* ágeis existentes são: Scaled Agile *Framework* (SAFe) (HAYES et al., 2016); Feature Driven-Development (FDD)(PALMER; FELSING, 2001); eXtreme Programming (XP) (WILDT et al., 2015); Dynamic Systems Development Method (DSDM) (STAPLETON, 1997); *Kanban* (BOEG, 2010); entre outros.

No contexto do estágio descrito, *Scrum* e *Kanban* foram utilizadas e são descritas nas seções a seguir.

2.1.1 *Scrum*

Scrum é uma metodologia ágil que visa ter escopo fechado (número de tarefas predefinidas), calculadas através de métricas da equipe, como pontos de esforço.

Foi desenvolvida baseada nas metodologias de empresas do japão, como Toyota e Honda. É um *framework* de trabalho que pode ser adaptado para diversas áreas.

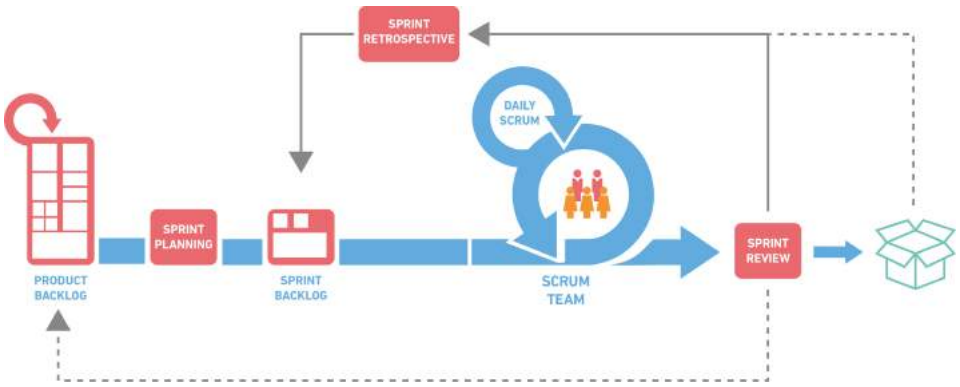
As tarefas a serem desenvolvidas ficam armazenadas no *product backlog*[1], sendo retiradas aquelas com maior prioridade para serem inseridas na *Sprint*.

> O *Sprint* é o ciclo de desenvolvimento, onde o incremento do produto pronto é gerado pelo Time de Desenvolvimento a partir dos itens mais importantes do Product Backlog. - (SABBAGH, 2014)

[1] Lista priorizada contendo as tarefas com uma breve descrição.

Também existem alguns ritos que devem ser seguidos para o bom funcionamento do *Scrum*, como as *Daily Meetings, Planning, Review Meeting e Retrospective*, que ocorrem em um fluxo como ilustrado na Figura 2.1.

Figura 2.1 – Fluxo *Scrum*



Fonte:https://blog.mjv.com.br/frameworks-ágeis-saiba-como-funcionam-na-prática

Daily Meetings são reuniões diárias, rápidas e práticas para atualização da equipe sobre em que cada um está trabalhando, para um melhor entendimento da equipe.

Review Meetings são reuniões feitas após o fim de cada *Sprint*, para entender e mostrar os resultados da equipe durante a *Sprint* para o PO.

Retrospective é uma reunião que ocorre todo fim de *Sprint* para identificar o que funcionou e o que pode ser melhorado na *Sprint*.

Planning é a reunião que deve ser feita antes de cada *Sprint* para definir quais atividades devem ser desenvolvidas, entender melhor sobre elas, descrevendo-as minuciosamente e prevendo possíveis problemas. Quem gerencia esse *product backlog* é o *Product Owner (PO)*, que cuida de organizar, definir e priorizar as tarefas.

Muitas vezes separadas em *Sprints* (tempo pré-definido pela equipe, geralmente de 10 dias úteis), as tarefas são incluídas em um quadro e devem ser finalizadas no prazo previsto.

> *Scrum* é um *framework* simples e pequeno e, assim, funciona bem em cada contexto se for utilizado em conjunto com outras técnicas e praticas a serem experimentadas e adaptadas. - (SABBAGH, 2014)

2.1.1.1 *Scrum Master*

Cargo que tem como função cuidar das obrigações impostas sobre a metodologia *Scrum*, como lembrar dos ritos, marcar reuniões e garantir o bom desenvolvimento das atividades estabelecidas para a *Sprint*.

> Garante que o time esteja totalmente funcional e produtivo.
>
> Facilita a colaboração entre as funções e áreas e elimina os impedimentos do time.
>
> Protege o time de interferências externas.
>
> Garante que o processo está sendo seguindo. Participando das reuniões diárias, revisão da *Sprint*, e planejamento. (SABBAGH, 2014)

2.1.1.2 Analista de Qualidade (Tester)

Responsável por encontrar problemas e possíveis melhorias durante o desenvolvimento, garantido o funcionamento total do sistema para que seja validado com o cliente.

2.1.1.3 Gerente de projetos (GP)

Cargo que tem como função gerenciar os projetos e times da sua tribo[2], cuidando para que sejam entregues os requisitos no prazo combinado.

> O gerente de projetos é a pessoa destacada e designada como principal responsável por atingir os objetivos do projeto. - (CRUZ, 2013)

2.1.1.4 *Product Owner (PO)*

Cargo que tem como função cuidar do relacionamento do time com o produto, definindo e priorizando requisitos.

> Define os requisitos do produto, decide a data de release e o que deve conter nela.
>
> É responsável pelo retorno financeiro (ROI) do produto.
>
> Prioriza os requisitos de acordo com o seu valor de mercado.
>
> Pode mudar os requisitos e prioridades a cada *Sprint*.
>
> o Aceita ou rejeita o resultado de cada *Sprint*. - (SABBAGH, 2014)

2.1.2 *Kanban*

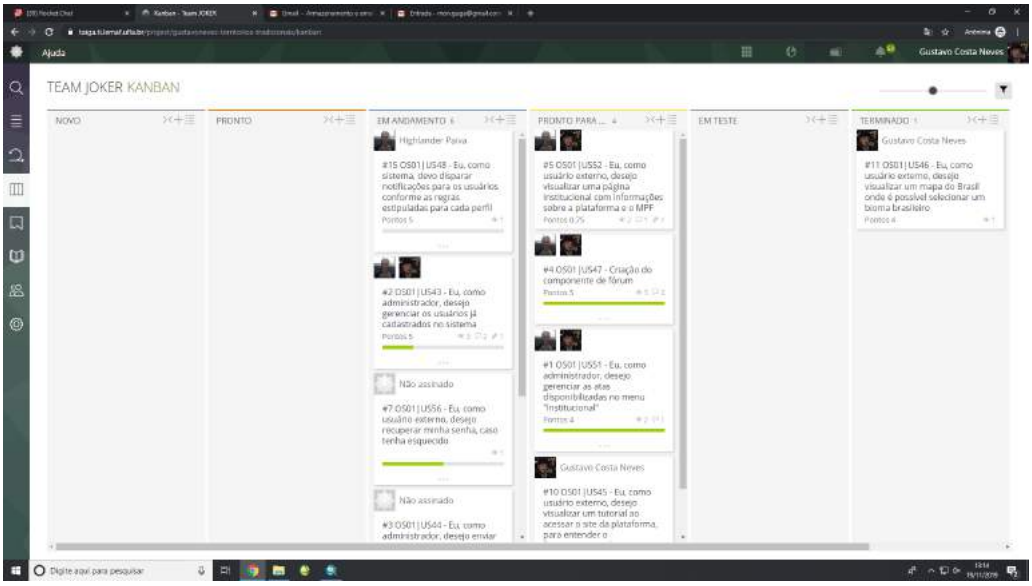
Kanban é uma opção de metodologia ágil mais adaptativa, tendo escopo aberto torna-se possível inserir atividades durante o tempo. Muitas vezes é comparado a uma tubulação de água, onde existe uma determinada quantidade de água a ser transportada, porém é necessário definir o tamanho do tubo ou vazão, sendo essa, a quantidade de esforço que a equipe consegue trabalhar. Assim como o *Scrum*, possui um backlog controlado por um Product Owner. As

[2] Conjunto de desenvolvedores e outros cargos que cuidam de um ou vários projetos.

atividades são inseridas assim que libera a vazão, como em uma tubulação. Diferentemente do *Scrum*, não há necessidade de *Reviews* e *Retrospective* uma vez que não possui *Sprints*, porém algumas reuniões para analisar o desempenho da equipe são um boa prática. Geralmente é determinado um máximo de esforço e, ao final de uma atividade, é inserida uma nova com prioridade.

O *Kanban*, como ilustra a Figura 2.2, é um quadro onde as tarefas são organizadas em filas de acordo com seu estágio de execução: Novo, Pronto, Em Andamento, Pronto para testar, Em teste, Terminado. Cada tarefa é definida com um identificador, descrição da atividade, pessoa responsável e prioridade. As tarefas podem ser movimentadas de uma fila para outra de acordo com seu estado de execução.

Figura 2.2 – Quadro *Kanban*



Fonte: https://taiga.ti.LEMAF.ufla.br/

2.2 *Frameworks*

Para facilitar o desenvolvimento, diversas ferramentas foram criadas e durante o desenvolvimento de novos projetos foi necessária a utilização das mesmas.

As aplicações que utilizam esses *frameworks* geralmente são separadas em *frontend* e *backend* devido à arquitetura cliente-servidor que é utilizada para o desenvolvimento das aplicações. O cliente é o agente responsável por enviar mensagens requisitando algum serviço ao servidor e é conhecido como front-end, pois é a parte que interage diretamente com o usuário.

O servidor é o agente responsável por realizar as operações necessárias para responder as requisições do lado cliente da aplicação, sendo também conhecido como *backend*. Essa definição facilita o desenvolvimento modularizado da aplicação, uma vez que o *backend* pode ser utilizado por diversos *frontend*.

2.2.1 Frontend

Nessa seção são apresentados frameworks para front-end utilizados pelo LEMAF.

2.2.1.1 AngularJS

AngularJS é um *framework open-source*, de desenvolvimento front-end, que possibilita o desenvolvimento de aplicações *web*.

Teve sua primeira versão lançada em 2010 pela Google.

> o AngularJS foi criado por Miško Hevery e Adam Abron em 2009 e é um *framework* JavaScript open source(código aberto), client-side(do lado do cliente) que promove uma alta produtividade na experiência do desenvolvimento *Web* - (FERREIRA; ZUCHI, 2018)

2.2.1.2 VueJS

VueJS é um *framework* JavaScript de *open-source*, focado no desenvolvimento de interfaces de usuário e aplicativos de página única.

Teve sua primeira versão lançada em 2014.

> Vue é uma lib/*framework* JavaScript reativo, para o desenvolvimento de componentes que, por sua vez, são códigos que podem ser reaproveitados em sua aplicação - (FERREIRA; ZUCHI, 2018)

2.2.1.3 ReactJS

O React é uma biblioteca JavaScript, *open-source*, com foco em criar interfaces de usuário em páginas *web*. É mantido por empresas como Facebook e Instagram e uma comunidade de desenvolvedores individuais.

Teve sua primeira versão lançada em 2013.

> Segundo a sua documentação (2018), ele é, na verdade, uma biblioteca de UI (User Interface), representando apenas a camada view do Model View Controler(MVC). - (FERREIRA; ZUCHI, 2018)

2.2.2 Backend

É a parte da aplicação que é inacessível ao usuário, onde é controlado todo o sistema (autenticação, regras de negócio, jobs e etc).

2.2.2.1 *Play Framework*

O Play *Framework* é uma alternativa "limpa"de esticar as stacks do Java Enterprise. Ele se concentra na produtividade do desenvolvedor e tem como objetivo arquiteturas RESTful.

O objetivo da estrutura do Play é facilitar o desenvolvimento de aplicativos da *web*, mantendo o Java.

Teve sua primeira versão lançada em 2009.

> *Play! Framework* 2 é planejado para ser "full stack"e completamente integrada, a boa notícia é que não há requisitos específicos para você ou meu ambiente começar a criar novos aplicativos da *web*. - (PETRELLA, 2013)

2.2.2.2 *Spring Boot*

O Spring é um *framework open-source* para a plataforma Java criado por Rod Johnson e descrito em seu livro "Expert One-on-One: JEE Design e Development". Trata-se de um *framework* baseado nos padrões de projeto inversão de controle e injeção de dependência.

Teve sua primeira versão em 2002.

> Spring *Framework* é um *framework* voltado para desenvolvimento de aplicações corporativas para a plataforma Java, baseado nos conceitos de inversão de controle e injeção de dependências. - (WEISSMANN, 2014)

2.2.2.3 *DotNet Framework*

O *.NET Framework* é uma iniciativa da empresa Microsoft, que visa uma plataforma única para desenvolvimento e execução de sistemas e aplicações. Todo e qualquer código gerado para .NET pode ser executado em qualquer dispositivo que possua um *framework* de tal plataforma.

Teve sua primeira versão lançada em 2002.

> O .Net *Framework* é um *framework* de desenvolvimento que fornece uma nova interface de programação para serviços e APIs do Windows e integra várias tecnologias que surgiram da Microsoft no final dos anos 90. (THAI; LAM, 2003)

2.2.3 Banco de Dados

É aonde ficam armazenadas informações necessárias para o funcionamento das aplicações.

2.2.3.1 Postgresql

É um sistema gerenciador de banco de dados desenvolvido na linguagem de programação C. Gerencia banco de dados relacionais e possui ótimo desempenho.

Possui extensões que contribuem com sua eficacia, como por exemplo o POSTGIS, uma extensão que possibilita o Postgresql a utilizar dados Geoespaciais. Um das vantagens é que sua licença é gratuita desde fins estudantis a empresariais. Teve sua primeira versão lançada em 2009.

> O PostgreSQL é uma das opções de banco de dados, pois se trata de um servidor SGBD de grande potencial e confiabilidade, contendo todas as características dos principais bancos de dados utilizados no mercado. Uma das suas características são suas licenças para uso gratuito, seja para fins estudantis seja para a realização de negócios, possibilitando que empresas o utilizem livremente. (MILANI, 2008)

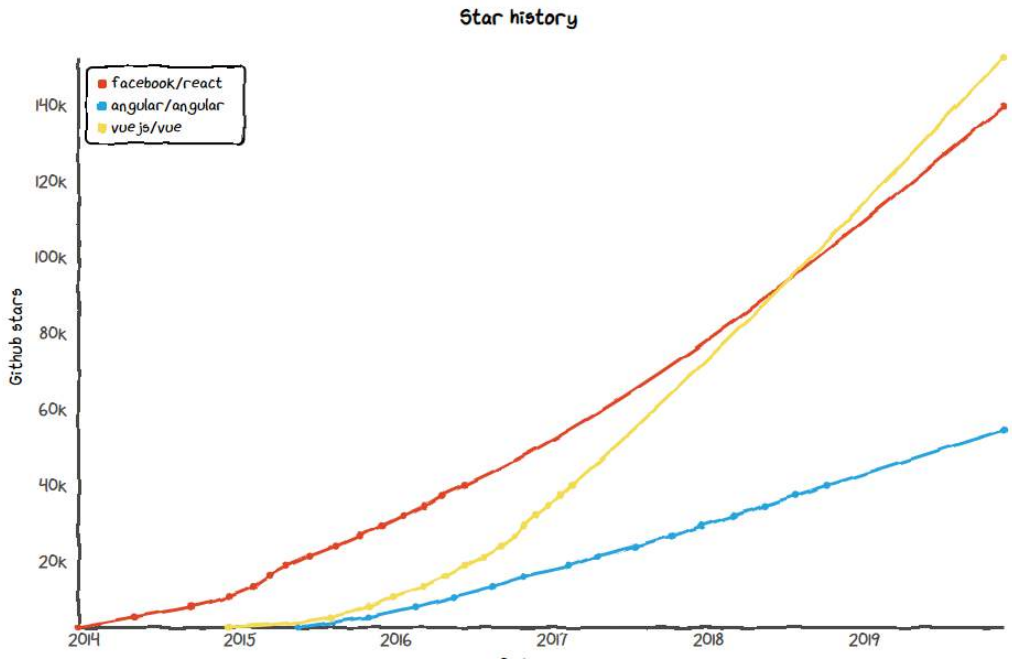
2.3 TECNOLOGIAS

Durante o estágio, tive a possibilidade de testar e utilizar diversas tecnologias e *frameworks*, com foco especial em tecnologias *frontend*. Para melhor entendimento do trabalho, é necessário um dissertação sobre as mesmas.

Os *Frameworks* principais de *frontend* são ReactJS, Angular e Vuejs. Sendo ReactJS e VueJS os principais em favoritos dentro da comunidade do GitHub[3] como demonstrado na figura 2.3

[3] Comunidade de desenvolvedores onde se é possível compartilhar e discutir sobre projetos e tecnologias.

Figura 2.3 – Comparação frameworks front-end pelo github



Fonte: https://star-history.t9t.io

Com seu lançamento tardio em relação ao ReactJS, o VueJS vem crescendo e neste ano de 2019 ultrapassou o numero de estrelas no github, enquanto o *framework* Angular cresce de forma lenta e contínua.

Os *frameworks* VueJS e ReactJS tinham como grande diferencial o uso da Virtual DOM[4] em vez da DOM(*Document Object Model*)[5]. Uma vez que a DOM era uma abstração do codigo HTML, a DOM Virtual é uma abstração da DOM.

Enquanto a DOM é uma árvore de objetos interligados, com diversas conexões e ramificações, onde modificações simples nas mesmas muito frequentes, a DOM virtual já possui configurações e implementações do *browser*, livrando o desenvolvedor de ter que configurá-las manualmente. Além disto a DOM virtual é mais leve e simples, sendo mais fácil compreendê-la e desenvolvê-la.

Outro grande diferencial graças a esta mudança, é que a abstração de componentes se tornou mais inteligente, uma vez que era possível desenvolver componentes desacoplados e acoplados somente onde eram necessários.

Em diversos projetos que tive contato, era utilizado Angular, por conta de ter sido lançado em 2010 pela Google. Foi uma grande evolução, uma vez que projetos mais velhos foram

[4] É um framework para manipulação do DOM
[5] É a representação dos componentes na página

desenvolvidos em Flex, que era uma tecnologia para interfaces em flash com integração com Java.

O *framework* Angular facilitava absurdamente o trabalho como desenvolvedor, uma vez que em diversos casos, que seriam necessárias diversas linhas de Jquery e Javascript, era somente necessário o uso de uma, como exemplificam os códigos da Figura 2.4 e a Figura 2.5 que adicionam um evento a elementos DOM pelo JQuery e VirtualDOM pelo Angular.

Figura 2.4 – Adicionando evento a elementos DOM pelo JQuery

```
var box = $( "#box" );
$( "#botao" ).on( "click", function( event ) {
    box.show();
});
```

Figura 2.5 – Adicionando evento a elementos VirtualDOM pelo Angular

```
<div ng-show={show} ng-click={() => {show = true}}/>
```

Como a base de desenvolvedores no LEMAF já possuia muita experiência com Angular, foi difícil fazer com que adotassem outras tecnologias em outros projetos.

Porém logo após iniciativa e insistência de um desenvolvedor, apresentamos o VueJS para a organização do LEMAF e conseguimos que os projetos fossem desenvolvidos com ele.

O VueJS não possuia documentação vasta como o Angular, porém possuia atualizações recorrentes, porém com sua ascensão, foi fácil demonstrar que em alguns anos a documentação dele seria melhor que a do Angular.

Porém era complicado mesmo assim, pois a credibilidade de um *framework* da Google contra a de um ex-desenvolvedor da Google(Criador do VueJS) era incomparável, era realmente um risco a mudança. Mas ao mostrar que o projeto recebia uma grande verba de patrocínio, a curva de crescimento e que o mesmo possuia uma equipe fixa de pelo menos 50 contribuidores, entenderam que o risco não era tão grande.

A adesão ao VueJS foi muito bem aceita em diversas equipes, pois a flexibilidade que ele proporciona ao desenvolvedor comparado ao Angular era gigantesca, com a possibilidade de configuração quase infinita, diversas bibliotecas e varios modos de arquitetura, o VueJS se tornou foco de estudo na empresa. Uma das iniciativas vindas dos organizadores da empresa foi fornecer alguns cursos da plataforma Alura para desenvolvimento *web* com VueJS.

Seu tamanho mesmo com sendo maior que o ReactJS, demonstrava mais fluidez, desde a ferramenta de *hot-reload*[6], até a compilação para modo produção. Sua arquitetura era simples e direta, centralizada em poucos arquivos e ao mesmo tempo separava bem os comportamentos. Sua documentação era bem escrita, descrevendo os ciclos de vida dos componentes até suas diretivas.

Muitos aprenderam por conta própria facilmente, uma vez que a complexidade abaixava, tendo toda a estrutura de um componente do Angular com 5 arquivos, em 1 arquivo .vue somente. Como vários módulos do LEMAF já utilizavam de Pug/Jade em vez do HTML puro, a transição foi muito tranquila, pois VueJS já suportava Pug e diversas outras linguagens.

Outro grande benefício encontrado durante a transição foram as bibliotecas de UI[7], pois com Angular o mais adequado era o uso do Material Design que já era da própria empresa ou Bootstrap[8], porém com o VueJS vieram varias outras bibliotecas como Vuetify e ElementUI, que tornaram as interfaces bem mais leves, modernas e bonitas, saindo daquele estilo antigo e estático.

Também foi apresentado pela equipe de design, eu e outro desenvolvedor, a ideia de Design System para a organização, que era algo que seria fácil de ser aplicado graças ao VueJS, que tinha um ótimo suporte a componentização e que se usado de forma correta(Separando componentes de visualização e lógica), poderia melhorar significantemente o tempo para desenvolvimento dos projetos.

Em 2019, vendo o advento das tecnologias de desenvolvimento hibrido para mobile, foi questionado a equipe se seria interessante o uso de *frameworks* mais próximos, para possíveis futuros projetos. Daí a adesão ao ReactJS, já que o principal utilizado para desenvolvimento mobile era o ReactJS-native, que se assemelhava bastante.

Ao estudarmos sobre, entendemos que a curva de aprendizado para quem desenvolvia VueJS iria ser bem pequena, já que ambos eram bem flexíveis.

Ele era mais leve que o VueJS que já era muito leve e ainda possuia uma documentação muito boa. porém um grande diferencial que existia era o uso do JSX, uma versão do Javascript

[6] Recarregamento quente, possibilita que a DOM seja recarregada somente nos locais de modificação, sem ser necessário o recarregamento da tela total.

[7] User Interface Design - é uma área de design que planeja e analisa os modos de iteração do usuario com o sistema.

[8] Framework para desenvolvimento de componentes e front-end para sites e aplicações *web* usando HTML, CSS e JavaScript.

que combina elementos do HTML dentro do Javascript. Era bem legível e bem similar ao HTML, mas não tinha um bom suporte para Pug.

Mesmo com este problema, resolvemos testar um projeto pequeno com ele.

As primeiras semanas foram difíceis, tivemos que estudar muito sobre o *framework* pois ele possuia diversas bibliotecas que melhoravam significantemente meu uso.

Após as duas primeiras semanas o desenvolvimento fluiu, as duvidas eram pouco recorrentes e as respostas eram fáceis de encontrar. porém voltamos ao problema inicial do Angular, a falta de um biblioteca de UI que combinasse com a identidade da empresa.

Foi colocado então em prática a construção do Design System, um processo de construção de identidade visual para empresa, com componentes próprios desenvolvidos com excelência para poderem ser utilizados em varios projetos, facilitando o desenvolvimento de todos e garantindo qualidade.

A criação dessa biblioteca tinha como objetivo fazer com que os desenvolvedores centralizassem os componentes, contribuindo publicamente com a biblioteca, sendo mais fácil corrigir um erro e depois atualizar a biblioteca que resolvê-lo em todos projetos.

E isto tornou-se prazeroso com ReactJS, pois o suporte para componentização por ele é ótimo, desde o desenvolvimento até a criação dos testes automatizados.

3 ATIVIDADES DESENVOLVIDAS

Nos primeiros dias como estagiário no LEMAF, foi necessário realizar treinamentos online para que entendesse como funcionava as principais tecnologias utilizadas lá. Forneceram uma licença na plataforma Alura para que aprendesse a utilizar o *framework* VueJS e Spring e diversos livros sobre os processos *Scrum* e banco de dados.

Um tutor foi designado para criar um quadro de tarefas que deveriam ser cumpridas para que tivesse o conhecimento básico para desenvolver os projetos. Após terminar as tarefas, foi passado um projeto para que fosse possível praticar e verificar os conhecimentos obtidos.

Na sua grande maioria, os projetos do LEMAF são sistemas *web*[1], constituídos de uma aplicação *backend* e uma *frontend*.

Durante o desenvolvimento, o *backend* é, geralmente, compilado no computador do próprio desenvolvedor, porém quando há necessidade de serem feitos testes, é criado um servidor dentro da rede local para que possam ser efetuados os mesmos, sendo mais precisos por conta de estarem em um servidor.

A infraestrutura de servidores[2] do LEMAF são servidores com diversas máquinas, todas rodando NGNIX[3] para prover as aplicações.

O *frontend* seguia o mesmo processo do *backend*, porém quando era enviado para alguma máquina externa, era criada alguma configuração para que o *backend* servisse o *frontend*, evitando assim problemas de CORS[4].

Já o banco de dados[5] era controlado por um DBA[6], que criava e gerenciava toda a estrutura de banco, sendo somente necessário aos desenvolvedores discutir melhores soluções e criar demandas para os mesmos realizarem e contemplarem suas necessidades.

Para a maioria dos bancos era utilizada o SGBD[7] Postgres[8], por conta de sua extensão POSTGIS que dava suporte a diversos tipos de dados relacionados a geometrias.

---

[1] Sistemas *web* são soluções de software que podem ser acessados através de um navegador em uma rede de computadores interna de uma empresa ou pela Internet.

[2] Computadores físicos ou virtuais que executam e servem a aplicação

[3] Disponivel em: https://www.nginx.com/

[4] CORS (Cross-Origin Resource Sharing), é uma regra criada para segurança dos usuários ao usar navegadores (<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Controle_Acesso_CORS>)

[5] Estrutura de armazenamento de dados

[6] Database Administrator, gerencia a estrutura do banco de dados, desde a escolha do gerenciador de banco de dados até a estrutura interna do mesmo.

[7] Sistemas de Gestão de Base de Dados

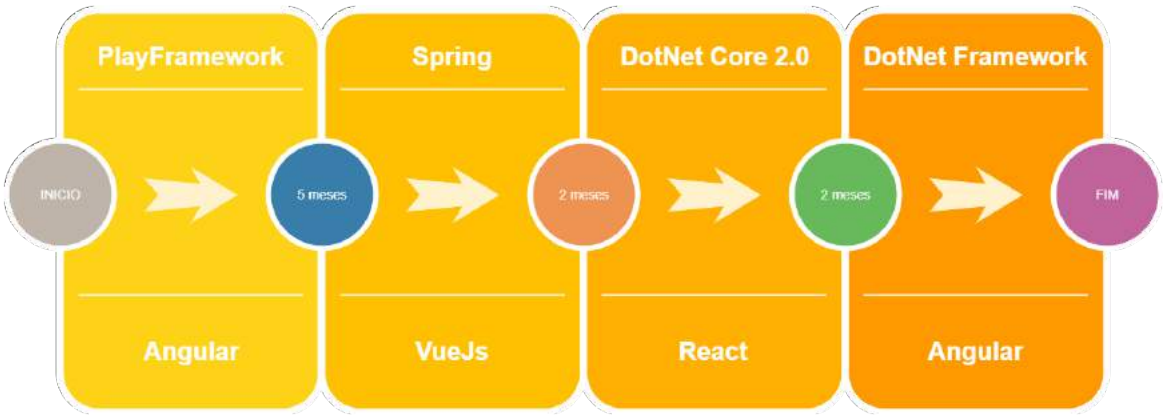
[8] Disponivel em: https://www.postgresql.org/

Como o foco do LEMAF é meio ambiente, o uso de mapas era recorrente na empresa, sendo necessário então obter um conhecimento básico sobre termos ambientais como APP e AUR.

> Área de preservação permanente - "área protegida, coberta ou não por vegetação nativa, com a função ambiental de preservar os recursos hídricos, a paisagem, a estabilidade geológica e a biodiversidade, facilitar o fluxo gênico de fauna e flora, proteger o solo e assegurar o bem-estar das populações humanas"(LEI. . . ,)
>
> Áreas de Uso Restrito - área localizada no interior de uma propriedade ou posse rural, delimitada nos termos do art. 12, com a função de assegurar o uso econômico de modo sustentável dos recursos naturais do imóvel rural, auxiliar a conservação e a reabilitação dos processos ecológicos e promover a conservação da biodiversidade, bem como o abrigo e a proteção de fauna silvestre e da flora nativa; (LEI. . . ,)

Durante o período de estagiário participei de todos squads, trabalhando com diversos times[9], projetos e tecnologias. E com isso, foi possível obter experiências com diversas tecnologias diferentes.

Figura 3.1 – Fluxo de aprendizado



3.1 SICAR

Em minha primeira equipe (Squad 1 - Carreta Furacão), tive como tecnologias necessárias JAVA (*backend*) e Angular (Frontend) para evoluir os projetos do Cadastro Ambiental Rural, incluindo os projetos SICAR, Central do Responsável Técnico, Central do Proprietário, PRA-OFF.

[9] Grupo de funcionários, geralmente formado por desenvolvedores, analistas de qualidade e Product Owners

Meu trabalho nesta equipe era desenvolver novas funcionalidades no sistema, como inserir novos dados no arquivo .PRA gerado pelo PRA-OFF, criptografá-lo e descriptografá-lo nas centrais.

Foi onde também tive meu primeiro contato com o Banco de Dados, uma vez que ainda não havia feito a disciplina deste assunto e não possuia domínio ou conhecimento necessário, tive que recorrera minha equipe que contava com desenvolvedores extremamente experientes e muito dispostos a ensinar. Houve diversas atividades onde fez-se necessário a criação de comandos SQL e como todos passavam por validação do DBA, tinha recorrentemente que corrigir os mesmos, entendendo e aprendendo com meus erros.

Também tive como obstaculo nunca ter utilizado alguma IDE, vinha programando em editores de texto desde o começo da graduação, então ao ter a necessidade de programar com eficiência, adotei o uso da IDE Eclipse, que já era utilizada pela equipe para desenvolvimento *backend*.

O *backend* era uma parte que conseguia entender, uma vez que já tinha experiências com JAVA graças a disciplina de Programação Orientada a Objetos e desenvolvimento Mobile, onde tive que desenvolver nativo para Android utilizando JAVA.

Para desenvolver e reproduzir o fluxo do sistema, foi necessário aprendizado sobre tecnologias relacionadas a georeferenciamento e mapas, como ARCGIS[10], leaflet[11] e GEOSERVER[12].

ARCGIS era uma ferramenta para desenho e exibição de geometrias, sendo possível reproduzir desde fazendas pequenas (menos de 4 módulos fiscais[13]) a áreas enormes.

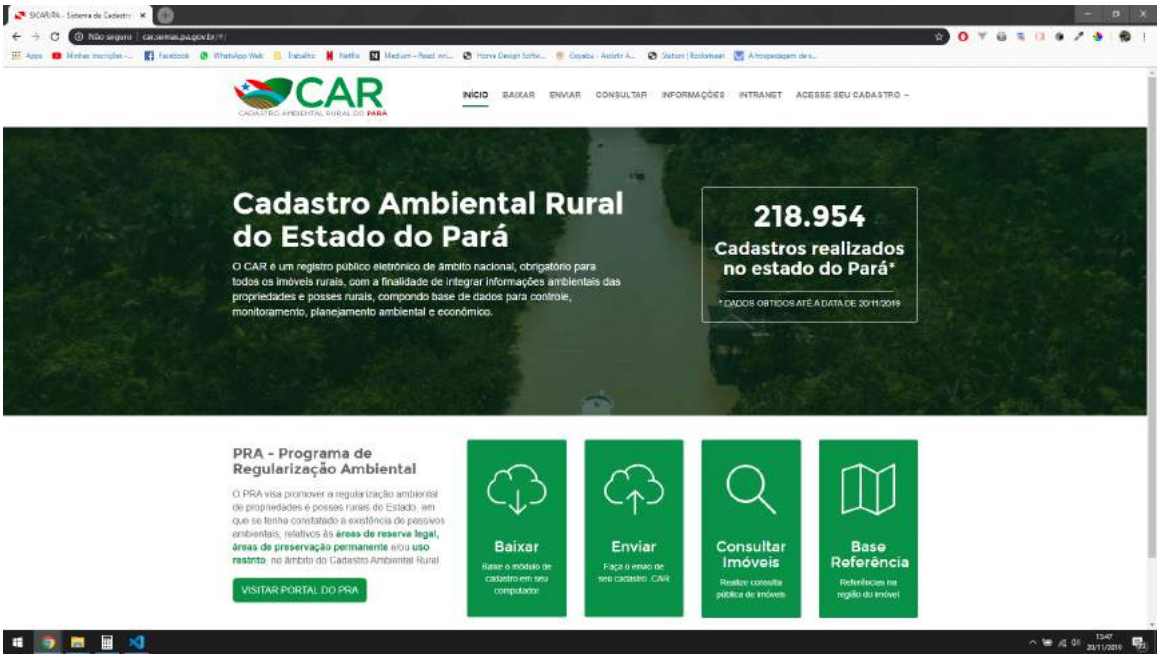
A leaflet era a principal biblioteca utilizada para mapas dentro do LEMAF, uma vez que era grátis e eficiente, com ampla documentação e compatibilidade. A ferramenta era de extrema importância para os projetos.

E para que ficasse centralizado diversos recursos de mapa, como as geometrias de APP do estado do PArá, hidrografia e outros, era utilizado o GEOSERVER, uma ferramenta que armazenava, gerenciava e provia geometrias.

[10] Disponivel em: <https://www.arcgis.com/index.html>
[11] Disponivel em: <https://leafletjs.com/>
[12] Disponivel em: <http://geoserver.org/>
[13] II - Pequena Propriedade - o imóvel rural:

a) de área compreendida entre 1 (um) e 4 (quatro) módulos fiscais;(LEI. . . ,)

Figura 3.2 – SICAR-PA



Fonte: <http://car.semas.pa.gov.br/>

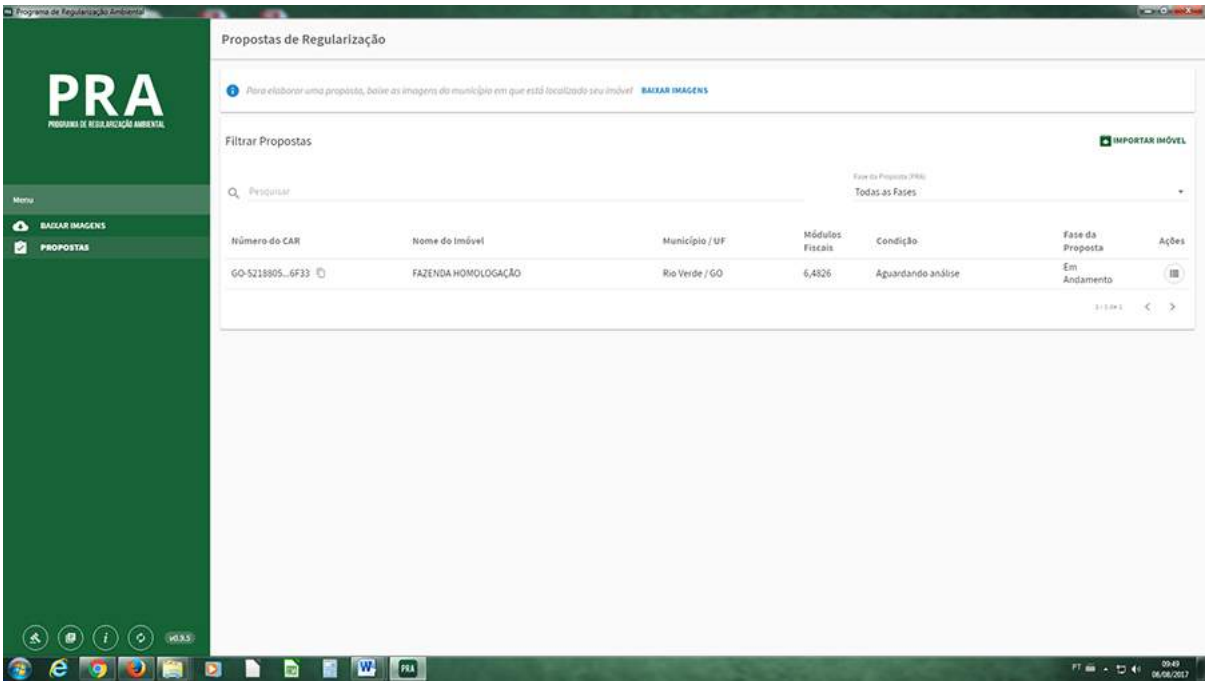
O projeto do SICAR (Figura 3.2)[14] tinha como objetivo informar e controlar os cadastros ambientais rurais feitos no estado do Pará, sendo necessárias algumas integrações com os módulos do SICAR federal e as Centrais ligadas ao PRA.

> Para adequação dos imóveis rurais à nova legislação, foi criado o Cadastramento Ambiental Rural (CAR) como registro público eletrônico de âmbito nacional, obrigatório para todos os imóveis rurais, com a finalidade de integrar as informações ambientais das propriedades e posses rurais, compondo base de dados para controle, monitoramento, planejamento ambiental e econômico e combate ao desmatamento. - (FILHO, 2015)

[14] Sistema de Cadastro Ambiental Rural, uma plataforma relacionada ao governo e que gerenciava os cadastros.

3.2 PRA

Figura 3.3 – PRA-OFF



Fonte: <http://www.cprh.pe.gov.br/Controle_Ambiental/Sistema%20Nacional%20de%20Cadastro%20Ambiental%20Rural%20-%20SICAR/PRA/43052%3B53356%3B480802%3B0%3B0.asp>

Já no projeto do PRA (Figura 3.3)[15], não eram possíveis tais integrações, pois ele era um módulo offline, sendo possível a utilização sem internet. Para o desenvolvimento do mesmo, foi necessário estudar sobre electron[16], uma ferramenta que possibilita criar módulos *web* em aplicações offline.

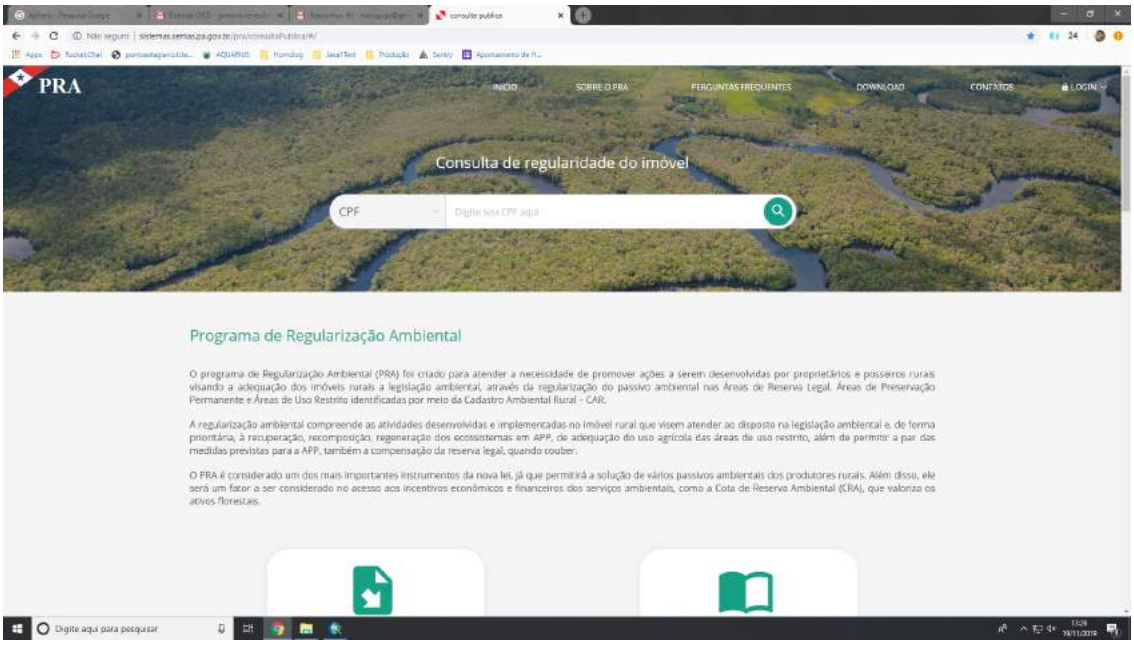
3.3 Consulta Pública

Após 5 meses a equipe foi dividida em duas e então foi feita uma redistribuição de projetos, quando fui alocado a tribo[17] Runners. Os projetos principais que contribui durante esse período foram o Consulta Pública - PARÁ e relatórios do PRA.

> O objetivo principal é promover as boas práticas de manejo florestal, por meio do enriquecimento das florestas secundárias remanescentes, localizadas fora das APPs, geralmente compondo a RL. - (FILHO, 2015)

[15] Programa de regularização ambiental, responsável por regularizar situações de desmatamentos e outras inflações ambientais
[16] Disponivel em: https://electronjs.org/
[17] Grupo de funcionários divido por equipes, gerenciado por um GP

Figura 3.4 – Consulta publica



Fonte: <http://sistemas.semas.pa.gov.br/pra/consultaPublica/#/>

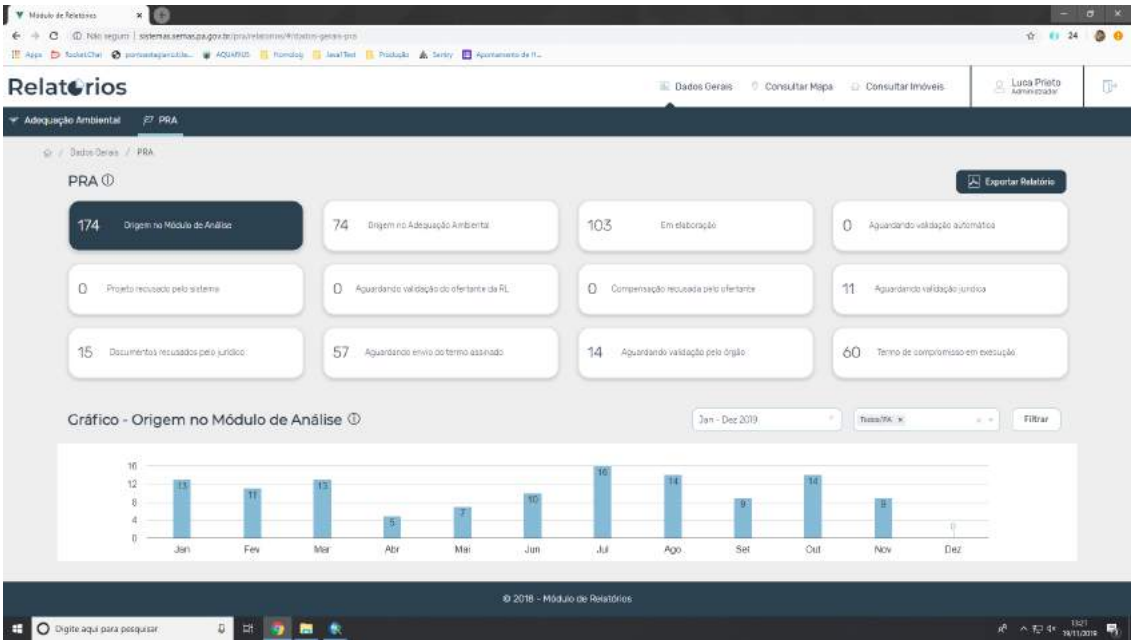
O projeto Consulta Pública (Figura 3.4) foi o primeiro projeto que tive como atividade a refatoração, pois se tratava de um projeto legado, utilizava uma das primeiras versões de VueJS no *frontend* e tinha o layout bem ruim. Foi a primeira obrigação que tive total responsabilidade, pois tinha que migrar todo o sistema para VueJS 2.0 e refatorar o *frontend*.

Como era minha primeira experiência com o *framework* VueJS, foi demandado mais tempo do que o usual, mas tive suporte do meu time que me auxiliava sobre arquitetura, padrões de projetos e boas práticas de programação. Para melhor aprendizado e controle sobre códigos ruins, todas modificações feitas por mim eram submetidas ao Code Review[18] enviadas para o projeto apenas perante aprovação.

O projeto Consulta Pública tinha como objetivo ilustrar aos proprietários de imóveis rurais suas áreas desmatadas, se seus imóveis estavam ou não de acordo com as regularizações ambientais e informações gerais sobre a geometria e hidrografia do terreno.

[18] Revisão do código, consiste em algum ou alguns membros da equipe analisando meu código e fazendo comentários de melhorias

Figura 3.5 – Relatórios do PRA



Fonte: <http://sistemas.semas.pa.gov.br/pra/relatorios/#/dados-gerais-adequacao-ambiental>

3.4 Relatórios do PRA

O projeto de relatórios (Figura 3.5) foi o primeiro projeto que tive participação desde o início, com uma equipe formada de 4 pessoas (2 desenvolvedores, uma tester e uma PO). O projeto consistia em uma plataforma de relatórios sobre o Programa de Regularização Ambiental e Adequação Ambiental.

Esse foi o primeiro momento onde tive que realmente aprender e utilizar dos conhecimentos relacionados a Banco de dados, e como havia feito a disciplina de Banco de Dados 1 e estava cursando Banco de Dados 2, foi de extrema importância a aprendizagem dessas disciplinas, já que me foi melhor explicado o funcionamento desses bancos de dados e daqueles relacionados a geometrias espaciais.

O time teve autonomia de escolha da tecnologia, então pela ótima experiência com VueJS e o alto desempenho de tal, este foi o escolhido para o *frontend*. Porém, pelo baixo conhecimento sobre *backend*, a escolha da tecnologia foi feita pelo outro desenvolvedor da equipe, que escolheu Spring Boot. Como sua experiência se limitava apenas a evolução de software, houve diversos gargalos no desenvolvimento, como criação de ambientes, scripts para automatização de deploy entre outros.

Durante o desenvolvimento deste projeto, descobriram um problema gigantesco no primeiro sistema que tive contato no PRA-OFF. Era ele quem empacotava e compactava as geo-

metrias para o arquivo .PRA, porém, quando existiam muitas geometrias, o arquivo PRA podia ter tamanhos acima de 1GB. O outro módulo, central do Responsável Técnico, que recebia os arquivos PRA, não aguentava processar o arquivo, travando no envio do mesmo.

Foi então minha responsabilidade desenvolver uma solução prática para o mesmo. Ná época estava cursando a matéria "Programação Paralela e Concorrente", daí tive a ideia de otimizar este processo através de threads e processos, porém isso não resolveu o problema. Após liberar bastante poder de processamento para o servidor e criar diversas otimizações no processo de criação do PRA, foi possível diminuir o arquivo para aproximadamente 1/4 de seu tamanho original.

Isso se tornou possível através de consultas a pessoas mais experientes com dados geográficos, que me sugeriram converter as geometrias de GEOMETRY para KML.

Mas após 2 meses, por necessitarem de um desenvolvimento com maior domínio de *frontend*, fui transferido para uma equipe especial, pois a tal não possuía tribo determinada. O projeto era uma POC (Prova de conceito) direcionada a uma empresa do ramo de agronomia. Com prazos curtíssimos e complexidade alta, o projeto foi um dos mais difíceis que já trabalhei. Ele foi construído utilizando componentização, com o *backend* feito com uma arquitetura bem definida e documentada para que fosse possível evoluir com o menor nível de dificuldade possível. Devido ao começo bem estruturado e documentado, o projeto foi bem sucedido.

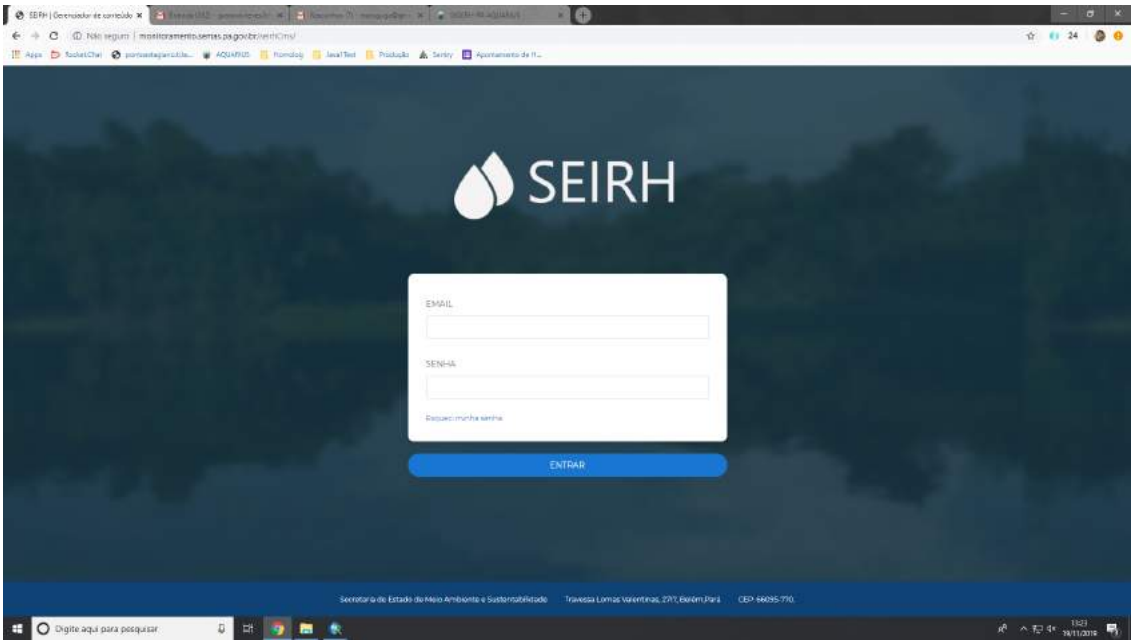
Após esse projeto, fui alocado na tribo Atlântida, onde trabalhei juntamente com mais um desenvolvedor no projeto SEIRH-CMS.

3.5 SEIRH-CMS

O projeto do SEIRH-CMS (Figura 3.6) era um gerenciador de conteúdo da aplicação SEIRH[19].

[19] Sistema Estadual de Informações Sobre Recursos Hídricos do Pará, responsável sobre oferecer informações sobre os projetos e programas relacionados a recursos hídricos do Pará

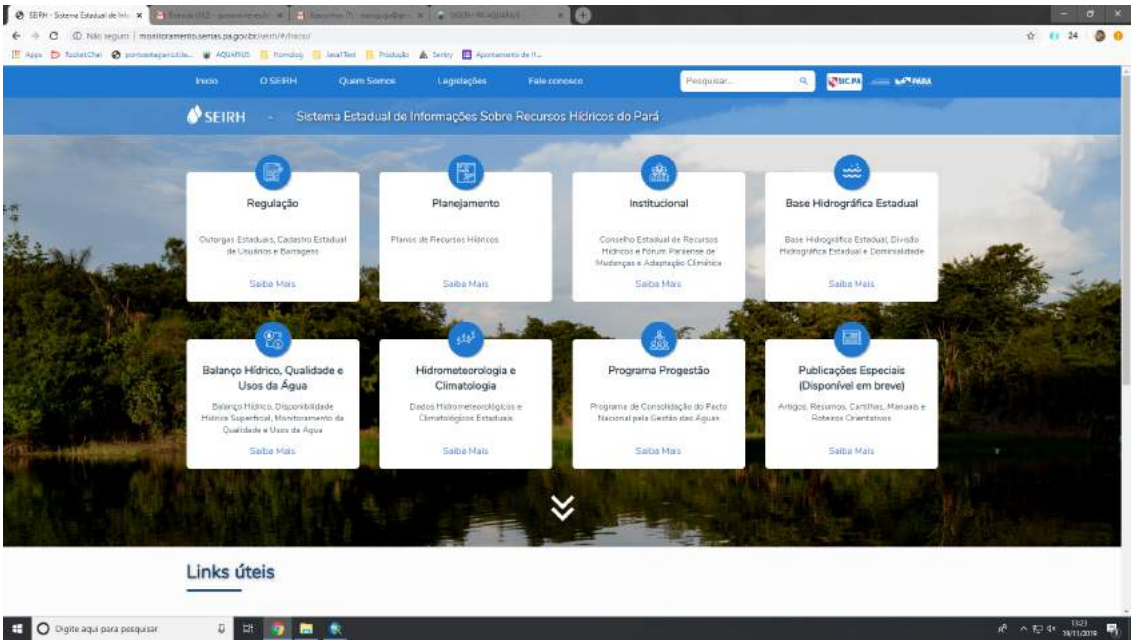
Figura 3.6 – SEIRH-CMS



Fonte: <http://monitoramento.semas.pa.gov.br/SEIRHCMS>

A plataforma *web* do SEIRH (Figura 3.7) já existia, porém não havia comunicação com um *backend*, então seu conteúdo não era gerenciável. Logo, havia a necessidade de refatoração total do sistema.

Figura 3.7 – SEIRH



Fonte: <http://monitoramento.semas.pa.gov.br/SEIRH>

Então, a refatoração do sistema foi atribuída como tarefa designada ao nosso time. Como meu conhecimento sobre *backend* já era mais amplo e devido a tribo Atlântida ser constituída de

desenvolvedores DotNet, resolvemos utilizar o *framework* DotNet Core 2.0, que provia diversas funcionalidades e atendia as nossas expectativas e necessidades. Já o *frontend* continuou sendo feito com VueJS, uma vez que nossas experiências com tal *framework* eram recentes.

O prazo para entrega do projeto era curtíssimo de duas *Sprints* (1 mês), então optamos por utilizar um quadro *Kanban* para gerenciar as atividades. Esta foi minha primeira experiência exercendo liderança, estabelecendo as prioridades e definir as atividades, definição das atividades e organização no geral.

O projeto foi finalizado com excelência e no prazo estipulado, o que me rendeu nova oportunidade de alocação em um time que já trabalhava com DotNet e necessitava de mais um desenvolvedor.

O projeto desta vez fazia parte de um grande complexo de plataformas que havia sido encomendado por uma empresa de agronomia e tinha muitos componentes de *frontend* similares entre si.

A partir dai, por iniciativa minha e de outro desenvolvedor de iniciarmos a construção do Design System em um contexto organizacional.

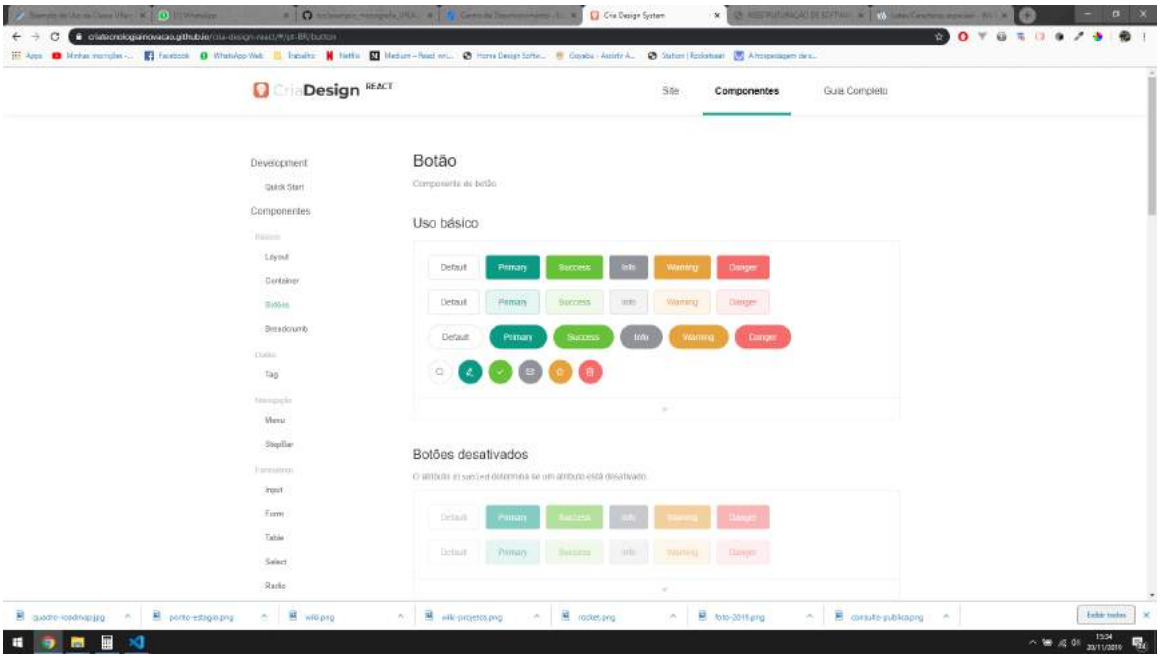
3.6 Cria Design System

Surgiu então o Cria Design System (Figura 3.8)[20], que é uma biblioteca de componentes UI[21] para ReactJS.

Todos os componentes eram criados pelo design do projeto, definindo comportamentos e animações, desenvolvidos depois. Para a visualização individual dos componentes, utilizamos o StoryBook[22], que cria e organiza o catálogo de componentes e então, para compartilhar tal biblioteca para toda a empresa e para que não surgissem bugs, foram implementados testes automatizados. Uma vez que já havia tido contato com testes automatizados, na disciplina de "Engenharia de Software", e havia sido encorajado a desenvolver os mesmo em "Desenvolvimento *Web*", resolvi que seria interessante a criação de testes desses componentes, testando-os visualmente e comportamentalmente, todos componentes e com necessidade mínima de cobertura de 80%.

[20] Disponível em <https://criatecnologiainovacao.github.io/cria-design-ReactJS/>
[21] User Interface Design - é uma área de design que planeja e analisa os modos de iteração do usuario com o sistema.
[22] Disponivel em: <https://storybook.js.org/>

Figura 3.8 – Cria Design System



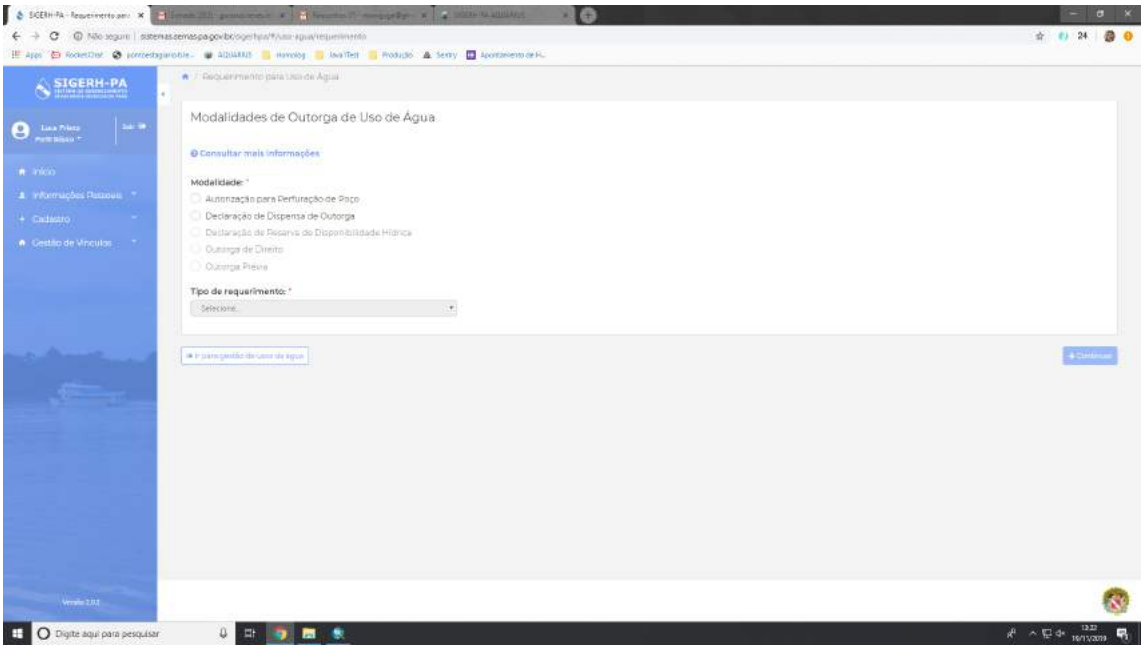
Fonte: <https://criatecnologiainovacao.github.io/cria-design-ReactJS/>

Após auxiliar com o início do projeto e facilitar o desenvolvimento do mesmo, fui incorporado a uma equipe da mesma tribo, que cuidava de alguns projetos como o SIOUT e SIGERH-PA.

3.7 SIGERH-PA

O projeto do SIGERH-PA (Figura 3.9) (Sistema de Gerenciamento de Recursos Hídricos do Pará) é uma plataforma de cadastro e regularização de recursos hídricos (poços artesianos, nascentes e outros). Nele usávamos como *backend* o *framework* DotNet *Framework* 4.0 e como *frontend* AngularJS.

Figura 3.9 – SIGERHPA



Fonte: <http://sistemas.semas.pa.gov.br/sigerhpa/>

O sistema tinha complexidade gigantesca, uma vez que para cada tipo de recurso hídrico, era necessário um fluxo especifico de cadastro.

Uma das atividades mais importantes pela qual fiquei responsável foi do desenvolvimento de um novo fluxo, totalmente diferente dos outros. Então o reaproveitamento de código foi baixíssimo, somente pude reaproveitar alguns componentes. Ao final do meu estágio, tive como tarefas principais: aprimorar e corrigir tal sistema, que tinha vários problemas, por se tratar de um sistema bastante amplo.

4 CONCLUSÃO

Toda minha experiência no LEMAF foi excepcional e de grande valia pro meu desenvolvimento profissional e pessoal, aprendendo a lidar com diversos problemas e pessoas.

A UFLA teve grande influência na minha iniciativa de estagio, uma vez que a vaga me foi oferecida num evento da SETI (Semana de Tecnologia da Informação) oferecido pela CompJunior[1] da UFLA. Consegui a vaga após falar na entrevista sobre alguns projetos que já havia desenvolvido pra disciplinas, sendo que chamou a atenção do entrevistador foi o projeto desenvolvido graças a matéria de Desenvolvimento Mobile, um aplicativo de chat com integração com Firebase[2]. Quando iniciei o estágio, meu conhecimento era somente relacionado ao contexto acadêmico lecionado pela universidade até o quinto período de graduação, o que em sua grande maioria eram somente conhecimentos e teorias sobre otimizações e códigos de baixo nível. Após alguns meses como estagiário, já possuía propriedade intelectual e autoridade para opinar em decisões estruturais de projetos, tecnologias e até mesmo estabelecer prazos, obviamente após inúmeros erros e muito estudo.

O estágio complementou a minha formação na UFLA, assim como os ensinamentos da UFLA complementaram meu estágio. Vários problemas foram solucionados graças a matérias lecionadas na universidade e, por outro lado. várias matérias que cursei se tornaram simples pois já havia experiência com aquilo.

Posso citar como exemplo as disciplinas de Banco de Dados, Sistemas Gerenciadores de Banco de Dados, Desenvolvimento *Web*, Modelagem e Implementação de Software, POO e muitas outras.

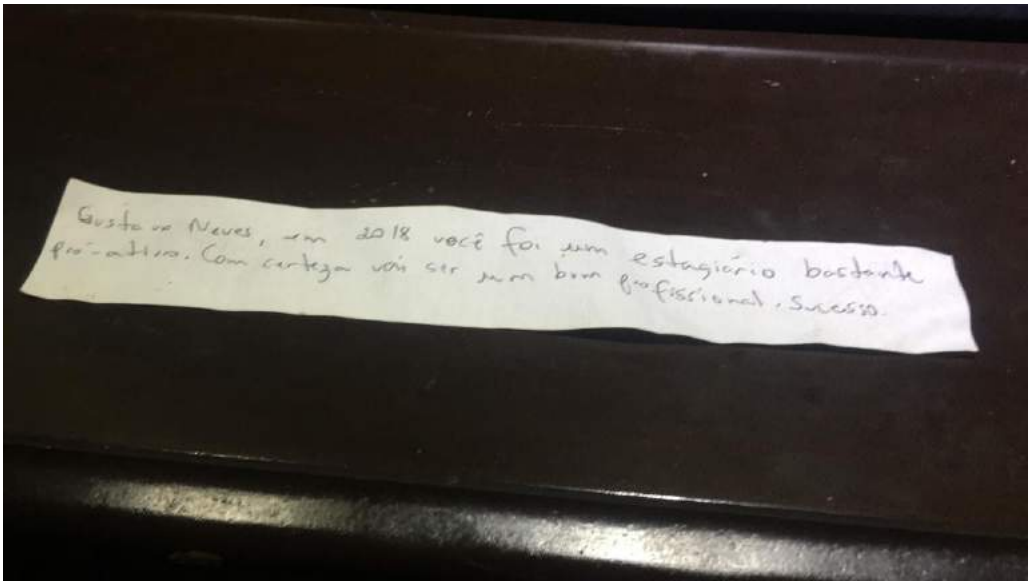
Os profissionais com os quais tive o prazer de trabalhar sempre foram muito agradáveis e solícitos, agregando muito valor ao meu desenvolvimento.

Devido aos problemas superados e projetos desenvolvidos no LEMAF, fui capaz de aprimorar minhas habilidades como desenvolvedor *web* e criar um bom portfólio, que me abriu diversas portas para o mercado de trabalho. O estágio também me proporcionou uma visão abrangente de cargos e caminhos que poderei trilhar dentro da minha área de estudo, o que fez aumentar as oportunidades de escolhas e afirmar o que realmente tinha como desejo e vocação: ser desenvolvedor.

[1] Empresa Júnior do departamento de Ciência da computação
[2] Um Baas (Backend as a Service) para aplicações *Web* e Mobile do Google

Posso afirmar com segurança que, os principais motivos da minha contratação na empresa Equals, meu emprego atual, foram minhas experiências com as tecnologias ReactJS, Spring e principalmente o desenvolvimento do Cria Design.

Figura 4.1 – Bilhete recebido na confraternização no fim de 2018



Toda minha trajetória no laboratório LEMAF foi muito prazerosa e produtiva. No ultimo dia como estagiário, fui convidado a participar do podcast da empresa, onde compartilhei um pouco da minha experiência com os outros funcionários [3].

[3] <http://blog.ti.lemaf.ufla.br/2019/08/22/podcast-5/>

REFERÊNCIAS

BOEG, J. Kanban em 10 passos. **Tradução de Leonardo Campos, Marcelo Costa, Lúcio Camilo, Rafael Buzon, Paulo Rebelo, Eric Fer, Ivo La Puma, Leonardo Galvão, Thiago Vespa, Manoel Pimentel e Daniel Wildt. C4Media**, 2010.

CRUZ, F. **Scrum e PMBOK unidos no Gerenciamento de Projetos**. [S.l.]: Brasport, 2013.

FERREIRA, H. K.; ZUCHI, J. D. Análise comparativa entre frameworks frontend baseados em javascript para aplicações web. **Revista Interface Tecnológica**, v. 15, n. 2, p. 111–123, 2018.

FILHO, C. F. M. de S. Cadastro ambiental rural (car) e povos tradicionais. **Revista da Faculdade de Direito da UFG**, v. 39, n. 1, p. 77–91, 2015.

HAYES, W. et al. Scaling agile methods for department of defense programs. figshare, 2016.

LEI nº 12.651, de 25 de maio de 2012. Acesso: 23 nov. 2019. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2012/lei/l12651.htm>. Acesso em: 23 nov. 2019.

LEI Nº 8.629, DE 25 DE FEVEREIRO DE 1993. Acesso: 23 nov. 2019. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/l8629.htm>. Acesso em: 23 nov. 2019.

MANIFESTO para Desenvolvimento Ágil de Software. [s.n.]. Acesso: dezembro/2019. Disponível em: <https://agilemanifesto.org/iso/ptbr/manifesto.html>. Acesso em: dezembro/2019.

MILANI, A. **PostgreSQL-Guia do Programador**. [S.l.]: Novatec Editora, 2008.

PALMER, S. R.; FELSING, M. **A practical guide to feature-driven development**. [S.l.]: Pearson Education, 2001.

PETRELLA, A. **Learning Play! Framework 2**. [S.l.]: Packt Publishing Ltd, 2013.

SABBAGH, R. **Scrum: Gestão ágil para projetos de sucesso**. [S.l.]: Editora Casa do Código, 2014.

STAPLETON, J. **DSDM, dynamic systems development method: the method in practice**. [S.l.]: Cambridge University Press, 1997.

THAI, T.; LAM, H. **. NET framework essentials**. [S.l.]: "O'Reilly Media, Inc.", 2003.

WEISSMANN, H. L. **Vire o jogo com Spring Framework**. [S.l.]: Editora Casa do Código, 2014.

WILDT, D. et al. **eXtreme Programming: Práticas para o dia a dia no desenvolvimento ágil de software**. [S.l.]: Editora Casa do Código, 2015.