FELIPE FERREIRA CARVALHO SILVA

DESENVOLVIMENTO DE UM SISTEMA *WEB* PARA LICENCIAMENTO AMBIENTAL

LAVRAS - MG

2021

FELIPE FERREIRA CARVALHO SILVA

DESENVOLVIMENTO DE UM SISTEMA *WEB* PARA LICENCIAMENTO AMBIENTAL

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

Prof. Dr. Ramon Gomes Costa
Orientador

LAVRAS - MG

2021

FELIPE FERREIRA CARVALHO SILVA

DESENVOLVIMENTO DE UM SISTEMA *WEB* PARA LICENCIAMENTO AMBIENTAL

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

APROVADA em 30 de abril de 2021.

Prof. Dr. Ramon Gomes Costa	UFLA
Prof. Dr. Paulo Afonso Parreira Júnior	UFLA
Gustavo Soares Melo	LEMAF/UFLA

Prof. Dr. Ramon Gomes Costa
Orientador

LAVRAS - MG
2021

Dedico este trabalho a todos os que me ajudaram ao longo desta caminhada.

AGRADECIMENTOS

Agradeço aos meus avós, pais e irmãs por me oferecerem todo apoio necessário para concluir esta importante fase de minha vida. Agradeço aos meus amigos que estiveram comigo todo esse tempo, em especial a Lorena Tavares, William Coelho e Rodrigo Herculano, que nunca mediram esforços para me auxiliar e me fizeram companhia em todos os momentos. Ao meu orientador, Prof. Dr Ramon Gomes Costa (DCC - UFLA), agradeço por toda a atenção oferecida durante a realização deste trabalho. A todos os docentes da Universidade Federal de Lavras (UFLA), por todo conhecimento passado através das disciplinas lecionadas pelos mesmos. Agradeço também a todas as pessoas que conheci durante o estágio, pela oportunidade, pelo companheirismo e pela força que sempre me deram em todos os momentos.

RESUMO

Com o crescimento da informatização, muitos trabalhos manuais começaram a ser integrados à sistemas Web, e com isso, a demanda por desenvolvimento de *software* cresceu. O presente relatório visa apresentar as tarefas realizadas pelo discente durante o estágio na empresa Fundação de Desenvolvimento Científico e Cultural (FUNDECC). As atividades têm como enfoque o desenvolvimento de um sistema Web voltado à preservação e fiscalização ambiental, e foram desenvolvidas no espaço do Laboratório de Estudos e Projetos em Manejo Florestal (LEMAF). O trabalho busca apresentar o processo de desenvolvimento em que o discente teve participação, desde a apresentação das tecnologias utilizadas, como exemplo Play *Framework*, AngularJS e PostgreSQL, descrição da rotina de trabalho e o processo de desenvolvimento dos sistemas.

Palavras-chave: Desenvolvimento; *Software*; Web; Licenciamento; Ambiental;

ABSTRACT

With the growth of computerization, many manual works started to be integrated with Web systems, and with that, the demand for the development of *software* grew. This report aims to present the tasks performed by the student during the internship at the company Fundação de Desenvolvimento Científico e Cultural (FUNDECC). The activities focus on the development of a Web system aimed at environmental preservation and inspection, and were developed in the space of the Laboratório de Estudos e Projetos em Manejo Florestal (LEMAF). The work seeks to present the development process in which the student participated, from the presentation of the technologies used, such as Play *Framework*, AngularJS and PostgreSQL, description of the work routine and the systems development process.

Keywords: Development; *Software*; Web; Licensing; Environmental;

SUMÁRIO

1	**Introdução**	9
1.1	**Motivação**	9
1.2	**Objetivos**	9
1.3	**Organização do texto**	10
2	**O LEMAF**	11
3	**Tecnologias Utilizadas**	13
3.1	**HTML e CSS**	13
3.2	**Javascript**	13
3.3	**Pug**	14
3.4	**AngularJS**	16
3.5	**Java**	16
3.6	**Play framework**	17
3.7	**PostgreSQL**	19
3.8	**Git e GitLab**	19
3.9	**Scrum**	20
3.9.1	**Scrum Team**	21
3.9.2	**Scrum Events**	22
3.9.3	**Scrum Artifacts**	23
3.10	**Taiga**	24
4	**Atividades desenvolvidas**	26
4.1	**Desenvolvimento do Estágio**	26
4.2	**Ambientes de Desenvolvimento**	28
4.3	**Desenvolvimento das Atividades**	29
4.3.1	**Licenciamento Ambiental do Amazonas**	30
4.3.2	**Análise do Licenciamento Ambiental do Amazonas**	31
4.3.2.1	**Verificação de Status de Comunicados**	33
4.3.2.2	**Parecer Jurídico**	35

4.3.3 Refatoração do Empreendimento 39
5 Considerações Finais 41

LISTA DE FIGURAS

Figura 3.1 – Página exemplo em Pug 15
Figura 3.2 – Página exemplo em HTML 15
Figura 3.3 – Estrutura do padrão MVC 18
Figura 3.4 – Representação do ciclo do Scrum 21
Figura 3.5 – Representação do Quadro Kanban no Taiga 25
Figura 4.1 – Diagrama de Dependências 27
Figura 4.2 – Fluxo de repositórios dos projetos 29
Figura 4.3 – Representação da função de verificação de status dos comunicados 34
Figura 4.4 – Representação da *model* do parecer jurídico 36
Figura 4.5 – Exemplo de e-mail enviado ao Diretor Jurídico 37
Figura 4.6 – Representação da *service* do parecer jurídico 37
Figura 4.7 – Tela de resposta do parecer jurídico 38

1 INTRODUÇÃO

Em uma graduação, o estudante adquire o conhecimento através das disciplinas e da informação passada pelos professores. Sendo assim, atividades como iniciação científica, a participação em grupos de estudo e a prática realizada em cursos e oficinas auxiliam o discente a desenvolver suas capacidades e seus conhecimentos. O estágio é uma dessas atividades, que além de ajudar o estudante a melhorar suas habilidades, permite ao aluno obter experiência no mercado de trabalho, vivenciando o dia a dia de uma empresa.

Neste relatório serão apresentadas as atividades realizadas pelo discente no Laboratório de Estudos e Projetos em Manejo Florestal (LEMAF), onde o foco principal das atividades foi o desenvolvimento de sistemas Web voltados à fiscalização ambiental. Os projetos em questão eram o Licenciamento Ambiental do Amazonas e o Análise do Licenciamento Ambiental do Amazonas, que serão descritos posteriormente neste trabalho.

1.1 Motivação

Com o crescimento da profissionalização da área de tecnologia da informação, a necessidade de experiência no mercado de trabalho aumentou. Com isso, a realização do estágio tem papel fundamental na formação do discente.

A grande carga teórica do curso de Ciência da Computação também é um motivo para o ingresso no estágio, pois assim o aluno consegue colocar em prática os conhecimentos adquiridos durante a graduação.

1.2 Objetivos

Em vista das necessidades do discente, o objetivo do estágio é promover o crescimento pessoal e profissional do aluno, aplicando as tecnologias estudadas em um cenário real de desenvolvimento.

Durante esse período, foram mantidas duas aplicações Web. A primeira é o Licenciamento Ambiental do Amazonas, cuja finalidade é realizar o cadastro de solicitações de licenças ambientais no estado do Amazonas. Seu objetivo é facilitar todo processo, que anteriormente era feito de forma manual, diminuindo assim o tempo de espera dos usuários que desejam solicitar uma licença. A segunda aplicação é a Análise do Licenciamento Ambiental do Amazonas. Esta é um módulo do sistema citado anteriormente, e utilizado pelos funcionários do Instituto de Proteção Ambiental do Amazonas[1] (IPAAM), que verificam os dados das licenças solicitadas e fazem a fiscalização, de forma presencial, na área informada.

1.3 Organização do texto

O trabalho se divide da forma a seguir: o presente Capítulo contém a parte introdutória, as motivações e o objetivo da realização do estágio. O Capítulo 2 trata da descrição do laboratório, como ele se divide e como é o processo de trabalho. Já no Capítulo 3 são descritas as tecnologias utilizadas, desde ferramentas usadas para auxiliar no desempenho da equipe, até *frameworks* para o desenvolvimento das aplicações. Os projetos executados pelo discente estão descritos no Capítulo 4, desde a fase de planejamento até o desenvolvimento. Por fim, no Capítulo 5, é apresentada uma análise dos resultados obtidos durante o estágio.

[1] http://www.ipaam.am.gov.br/

2 O LEMAF

O LEMAF, fundado em 2004, possui como foco essencial o apoio à atividades acadêmicas, como pesquisa, ensino e extensão. Sediado no campus da Universidade Federal de Lavras (UFLA), os principais projetos do laboratório são desenvolvidos em apoio e convênio com órgãos do governo, estaduais e federais. Existem também projetos realizados com o setor privado. (LEMAF, 2019).

Ainda segundo LEMAF (2019), o LEMAF tem como base as seguintes áreas:

- **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."

Durante o estágio, o discente trabalhou na área de tecnologia da informação, atuando no desenvolvimento de sistemas. O laboratório é dividido por setores. O primeiro setor que o discente teve contato foi o de Recursos Humanos (RH). Este tem um papel fundamental na gestão de pessoas, realiza as contratações e auxilia os funcionários com a parte burocrática. A Gerência é o setor que cuida principalmente da parte administrativa e de gestão de novos projetos. O restante dos membros está incluso no setor das Tribos, que é a denominação das pessoas que atuam diretamente nos projetos. Estão inclusos nas Tribos os seguintes profissionais:

- **Times de desenvolvimento:** São compostos pelas equipes que atuam diretamente no desenvolvimento dos projetos, essas equipes seguem os ritos do Scrum[1] e possuem, no geral, três desenvolvedores, sendo que um deles é o *scrum master*, um analista de qualidade e um *product owner*.

- **Gerente de Projetos (GP):** Tem como principal papel o planejamento de novos projetos, coordenar as atividades em desenvolvimento e servir como ligação dos times de desenvolvimento com a Gerência.

- **Administrador de Banco de Dados:** Tem o papel de gerenciar toda parte de banco de dados da tribo, cuida da criação das bases de dados, manutenção e evolução dos bancos de dados de cada projeto.

[1] https://www.scrum.org/

3 TECNOLOGIAS UTILIZADAS

O objetivo deste capítulo é descrever e apresentar as tecnologias, *frameworks* e ferramentas utilizadas durante o desenvolvimento dos projetos.

3.1 HTML e CSS

Na área de programação *web*, a linguagem de marcação de hipertexto[1] (HTML) é uma das formas mais simples de construção de *sites web*. O HTML define a estrutura da página *web* e seu conteúdo, e, para isso, é utilizado juntamente com CSS[2], para estilizar a página, e Javascript[3], para manipular seu comportamento (MOZILLA, 2021b).

Ainda de acordo com Mozilla (2021b), o HTML utiliza de marcações, que são denotadas através de símbolos especiais, como exemplo, <html>, <head>, <body> e <footer>, para poder representar e estruturar o conteúdo de uma página Web.

O CSS, citado anteriormente, é uma linguagem utilizada para estilizar documentos HTML, ou seja, ela é responsável por definir como um conteúdo de uma página Web vai ser apresentado na tela. (CSS, 2021).

O estagiário teve o primeiro contato com ambas as linguagens durante a realização da matéria eletiva de Programação Web, e durante o estágio aprofundou o conhecimento por meio de cursos e no próprio desenvolvimento dos projetos.

3.2 Javascript

Javascript (JS) é utilizado para manipular elementos e controlar o comportamento dos dados em uma página Web por meio da ocorrência de eventos. Estes são ações de usuários decorrentes da interação com elementos da página Web. JS

[1] https://www.w3schools.com/html/
[2] https://www.w3schools.com/css/
[3] https://www.javascript.com/

é uma linguagem interpretada que contém grande capacidade multi-paradigma e dinâmica, o que permite a ela suportar os estilos orientado a objetos, imperativo e funcional (MOZILLA, 2021a).

No âmbito dos projetos desenvolvidos pelo discente, o Javascript foi utilizado para controlar e manipular dados nas páginas Web e assim ser possível fazer o sistema responder às ações do usuário. O primeiro contato prático do discente com o Javascript foi durante a realização de cursos no início do estágio, e durante todo o período no LEMAF foi se aprimorando e evoluindo na construção de aplicações que utilizavam a tecnologia.

3.3 Pug

Pug[4] é um pré-processador utilizado para facilitar e aprimorar a escrita de códigos HTML. Por conta de permitir a utilização de instrumentos de programação, como condicionantes e iterações, e ter suporte nativo ao Javascript, o Pug facilita a implementação de páginas que precisam ter interação com dados (MEDIUM, 2018).

O Pug utiliza a identação do código para organizar a ordem de hierarquia das marcações. A Figura 3.1 apresenta um exemplo de código em Pug e a Figura 3.2 apresenta o código processado e transformado em HTML.

[4] https://pugjs.org/

Figura 3.1 – Página exemplo em Pug

```
doctype html
html(lang='pt-br')
  head
    title Este e um teste de titulo!
  body
    h1 Ola, mundo!
    div.test
      p Teste deu certo!
```

Fonte: Do autor (2021)

Figura 3.2 – Página exemplo em HTML

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Este e um teste de titulo!</title>
  </head>
  <body>
    <h1>Ola, mundo!</h1>
    <div class="test">
      <p>Teste deu certo!</p>
    </div>
  </body>
</html>
```

Fonte: Do autor (2021)

Durante o estágio, o Pug foi utilizado no *front-end*, pois facilitava o entendimento e a construção dos códigos HTML. Por ter suporte nativo ao Javascript, a manipulação de dados na tela era feita de forma simplificada.

3.4 AngularJS

AngularJS é um *framework* estrutural utilizado no desenvolvimento *front-end* de sistemas *web*. O *framework* promove a construção de uma nova estrutura HTML. Este permite que o HTML seja usado como a linguagem base do projeto, com sua sintaxe estendida, para que assim seja possível manipular os dados de uma página Web de forma clara e simplificada (GOOGLE, 2020).

Ainda segundo Google (2020), o AngularJS oferece uma série de ferramentas para a construção de uma aplicação *front-end* completa. Como exemplo, o *framework* tem suporte a *data-binding*, roteamento, injeção de dependências e validação de formulários. Sendo assim, a construção de uma aplicação que contém as 4 operações básicas, *Create* (Criar), *Read* (Consultar), *Update* (Atualizar) e *Delete* (Deletar) (*CRUD*), é feita de forma rápida e simplificada pelo framework.

O AngularJS, na versão 1.6, foi o *framework* usado na estruturação do *front-end* dos projetos em que o discente participou. As tecnologias citadas anteriormente foram utilizadas juntamente com o *framework* para que fosse possível ser feita a construção das telas dos sistemas.

3.5 Java

Segundo Deitel e Deitel (2015), o Java[5] é uma linguagem de programação utilizada em diversos ambientes computacionais. Isso se deve ao fato de que ela foi construida para ser capaz de produzir programas que podem ser executados em diferentes sistemas computacionais. Com o crescimento da *web*, o Java também cresceu, por conta de sua alta flexibilidade e capacidade de adaptação. Ele é utilizado para o desenvolvimento de sistemas de grande porte, na construção de aplicativos de uso geral e outros propósitos.

[5] https://www.java.com/pt-BR/

O Java foi a linguagem utilizada para construir o *back-end* dos projetos em que o discente trabalhou, sendo utilizado com o auxílio do *framework* Play. Durante a graduação, o aluno aprendeu e fez trabalhos práticos utilizando a linguagem, o que auxiliou no desenvolvimento dos projetos durante o estágio.

3.6 Play framework

O Play[6] é um *framework* que tem o foco de simplificar o desenvolvimento de sistemas Web Java. Ele conta com as ferramentas necessárias para a construção de um sistema Web robusto, como o suporte nativo a bancos de dados relacionais através do JDBC[7], suporte a *Hibernate*[8] para realizar o mapeamento das *models* com o banco de dados através do *Java Persistence API* (JPA), consumo direto de *web services* em JSON ou XML, dentre outras ferramentas. (PLAY, 2021).

Em Play (2021) é informado que o Play economiza um tempo precioso de desenvolvimento, principalmente por oferecer *'hot reloading'*, o que significa que é possível visualizar imediatamente as alterações feitas no *back-end*, sem precisar recarregar todo o sistema.

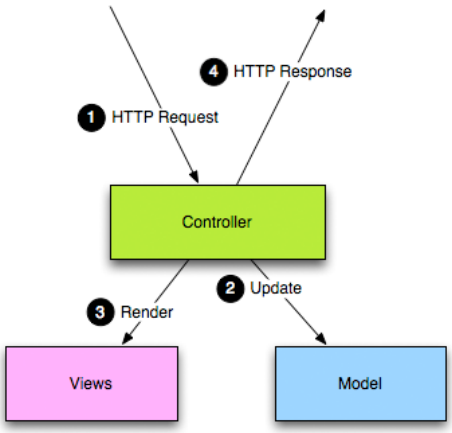
Um conceito importante que o *framework* implementa é o padrão Model/-View/Controller (MVC) (Figura 3.3). De acordo com Play (2021), esse padrão divide a aplicação em camadas, sendo elas:

- **Model:** é a representação das informações de uma entidade presente no banco de dados do sistema. A lógica da entidade, presente na *model*, trás significado aos dados do sistema. No caso do LEMAF, os dados brutos são armazenados em um banco de dados relacional e mapeados através do JPA nas *models*.

[6] https://www.playframework.com/
[7] https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/
[8] https://hibernate.org/

- **View:** tem o papel de renderizar a *model* em um formato que seja adequado para interações com o usuário final. Podem existir diversas *views* para uma única *model*, de acordo com o propósito necessário. Nos projetos, como eram aplicações Web, a *view* era renderizada por meio das tecnologias *front-end* citadas anteriormente nesta seção.
- **Controller:** responsável por escutar as requisições HTPP vindas do *front-end*, processar e extrair as informações importantes, invocar as alterações na *model* e no fim responder à requisição.

Figura 3.3 – Estrutura do padrão MVC



Fonte: Play (2021)

O Play é utilizado na versão 1.5 em ambas as aplicações citadas neste trabalho. Por ser uma versão antiga, é difícil encontrar uma documentação clara e funcional dos métodos presentes nesta versão, o que causou alguns transtornos durante o desenvolvimento.

O primeiro contato do discente com o *framework* foi no início do estágio, nas primeiras semanas foi realizado um treinamento utilizando o mesmo. Após entrar na Tribo, os membros do time de desenvolvimento ajudaram o discente

a criar familiaridade com o *framework*. Ao longo do tempo, com a utilização diária do Play, o discente foi capaz de aprender muito sobre o *framework* e com isso começou a conduzir *workshops* para novos membros da equipe, explicando o funcionamento e como o mesmo deveria ser configurado.

3.7 PostgreSQL

O PostgreSQL[9] é um Sistema Gerenciador de Banco de Dados Objeto Relacional (SGBDOR) de código aberto, que contém diversas ferramentas necessárias para a construção de grandes aplicações. O SGBDOR é compatível com ACID[10] - Atomicidade, Consistência, Isolamento e Durabilidade. (POSTGRESQL, 2021)

Ainda de acordo com PostgreSQL (2021), o PostgreSQL contém muitos recursos destinados a ajudar tanto os desenvolvedores quanto os administradores de banco de dados. Uma importante e poderosa extensão presente no PostgreSQL é o PostGIS[11], que oferece suporte ao armazenamento de objetos geoespaciais.

Na tribo, todos os sistemas utilizam o PostgreSQL como Sistema Gerenciador de Banco de Dados (SGBD), bem como utilizam o PostGIS para poder armazenar e utilizar os dados geográficos necessários para a construção dos sistemas.

3.8 Git e GitLab

”Git[12] é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com todos os tipos de projetos, desde pequenas aplicações até grandes sistemas, com velocidade e eficiência.” (GIT, 2021)

[9] https://www.postgresql.org/
[10] https://database.guide/what-is-acid-in-databases/
[11] https://postgis.net/
[12] https://git-scm.com/

Nos projetos, o Git era utilizado para fazer todo controle de *branches* (ramificações) locais, já o armazenamento remoto do código fonte era feito através da plataforma GitLab[13].

Segundo GitLab (2021), o GitLab é um sistema de controle de repositórios baseado em Git. Sua principal vantagem é que ele fornece uma plataforma completa, com segurança no armazenamento dos dados, interface gráfica clara e intuitiva e a possibilidade de armazenamento do serviço nos servidores locais da empresa.

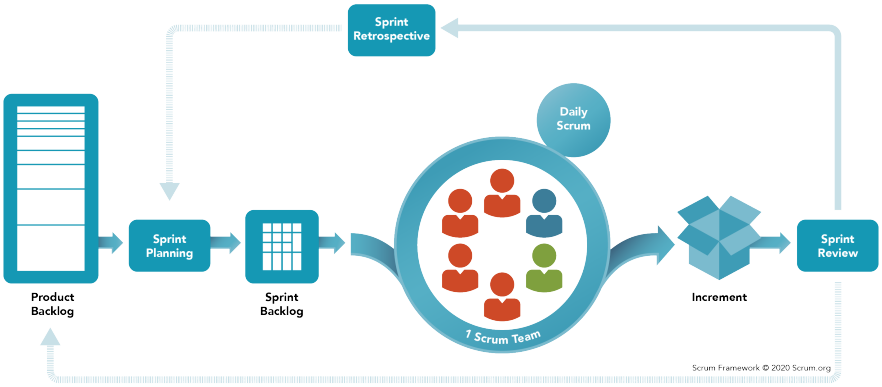
O LEMAF tinha seu próprio servidor do GitLab, o que trazia vantagens, pois os problemas eram resolvidos rapidamente no servidor local e o acesso ao sistema era muito mais rápido. Por outro lado, haviam problemas decorrentes da disponibilidade de energia elétrica, pois quando haviam problemas na rede, os servidores ficavam indisponíveis.

3.9 Scrum

”Scrum é um *framework* leve que ajuda as pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos.” (SCRUMGUIDES.ORG, 2020)

[13] https://about.gitlab.com/

Figura 3.4 – Representação do ciclo do Scrum



Fonte: Scrum (2020)

A Figura 3.4 apresenta todo o ciclo do Scrum. Durante o tempo em que o discente esteve no LEMAF, o Scrum foi implementado e três tópicos relevantes foram abordados e serão descritos a seguir.

3.9.1 Scrum Team

Segundo ScrumGuides.org (2020), o *Scrum Team* é composto por:

- ***Developers*:** são as pessoas no *Scrum Team* responsáveis pelo desenvolvimento das atividades. No LEMAF, o time de desenvolvimento é composto por 3 pessoas, onde cada uma tem a responsabilidade de realizar todas as tarefas propostas, bem como resolver os problemas de código que surgirem durante o desenvolvimento.

- ***Product Owner* (PO):** Dentro dos projetos, o PO tem o papel de gerenciar todas as tarefas que devem ser realizadas, entender as necessidades dos clientes e conseguir passar de forma clara e detalhada para o time de desenvolvimento.

- ***Scrum Master*:** é responsável por verificar se o Scrum está sendo praticado de maneira correta por todo *Scrum Team*. Na equipe em que o discente participou, o *Scrum Master* era um desenvolvedor. Tal decisão trazia a vantagem de que ele conseguia ver as necessidades de todo *Scrum Team*, bem como fiscalizar de perto se o Scrum estava sendo executado na prática.

3.9.2 Scrum Events

Segundo ScrumGuides.org (2020), eventos são instrumentos utilizados no Scrum para criar transparência e regularidade em todo processo. Alguns desses eventos são utilizados no dia a dia da Tribo, são eles:

- ***Sprint*:** pode ser considerada a principal parte do Scrum, pois é onde as atividades são realmente executadas. As *Sprints* são eventos de duração fixa. No LEMAF, uma *Sprint* geralmente durava 15 dias, se o escopo das atividades planejadas fosse pequeno, esse período caia para 7 dias. Elas permitem previsibilidade, proporcionando a possibilidade de previsão e adaptação do escopo das atividades.

- ***Planning*:** é a marca que inicia a *Sprint*, nela são definidas as atividades (histórias) que serão executadas durante esse período. Para definir as histórias, o *Product Owner* expõe o que deve ser feito na *Sprint*, e com isso todo o *Scrum Team* discute sobre quais são as atividades prioritárias que devem entrar no escopo. Para não sobrecarregar os desenvolvedores, a quantidade de atividades aceitas é de acordo com a capacidade de execução do time. Para auxiliar nisso, o *Scrum Team* do discente aplicava o *Scrum Poker* [14] que permitia os integrantes darem uma nota para a dificuldade de execução da tarefa. Os pontos de cada história seguiam a Sequência de Fibonacci[15].

[14] https://scrumpoker.online/
[15] http://www.im.ufrj.br/ pacifico/calculo2/1-Fibonacci.pdf

Sendo assim, aquelas que recebiam 1, 2 ou 3 pontos eram consideradas fáceis de serem desenvolvidas e testadas. Tarefas que recebiam nota 5 tinham dificuldade média. As que recebiam o valor de 8 pontos eram consideradas difíceis, mas que eram possíveis de serem desenvolvidas dentro do escopo da *Sprint*. Atividades que recebiam notas 13 ou superior eram divididas em histórias menores. Após avaliar todas as tarefas, era discutido quais delas entrariam no escopo da *Sprint* atual.

- ***Daily Scrum*:** é uma reunião de 10 a 15 minutos que tem o objetivo de inspecionar o progesso da *Sprint*. Todo *Scrum Team* participa da reunião, algumas vezes, pessoas de fora do time - mas que estão envolvidas na *Sprint* - também participam. Cada pessoa fala brevemente sobre as atividades que estão desenvolvendo, os problemas e as conquistas do dia. A *Daily* auxilia muito no entendimento de todo time a respeito do desenvolvimento do projeto.

- ***Sprint Retrospective*:** é uma reunião feita ao final de cada *Sprint*. O *Scrum Master* reúne toda a equipe para discutir como foi a *Sprint*, os pontos positivos e negativos, e possíveis melhorias para as próximas. É tão importante quanto as *Dailys*, pois por ter uma duração maior, permite que todos os pontos do *Scrum Team* possam ser expostos e discutidos.

3.9.3 Scrum Artifacts

Segundo ScrumGuides.org (2020), os *Scrum Artifacts* representam as informações de tudo que deve ser feito em determinado projeto, sendo projetados para tentar maximizar a transparência das informações principais. Fazendo com que toda equipe tenha uma base confiável de onde consultar o que deve ser feito. No LEMAF, eram utilizados dois tipos de artefatos:

- ***Product Backlog*:** é uma lista construída pelo PO que define atividades necessárias para a melhoria e evolução do produto. Durante a Planning, as atividades que entrarão no planejamento estão no *Product Backlog*, e muitas vezes, antes delas entrarem de fato para o *Sprint*, elas são quebradas em atividades menores por meio de um Refinamento. Essa atividade serve para adicionar mais detalhes ao item.

- ***Sprint Backlog*:** é uma forma de representar a evolução da *Sprint* em tempo real, tornando possível visualizar no *Sprint Backlog* o status de todas as atividades, e com isso analisar o progesso da *Sprint*. No LEMAF, esse progresso era visualizado através do quadro Kanban[16] presente no Taiga, que será descrito na próxima seção.

3.10 Taiga

"O Taiga[17] é uma ferramenta de gerenciamento de projetos para equipes ágeis. Possui um rico conjunto de recursos e, ao mesmo tempo, é muito simples de se utilizar por meio de sua interface de usuário intuitiva." (TAIGA, 2021)

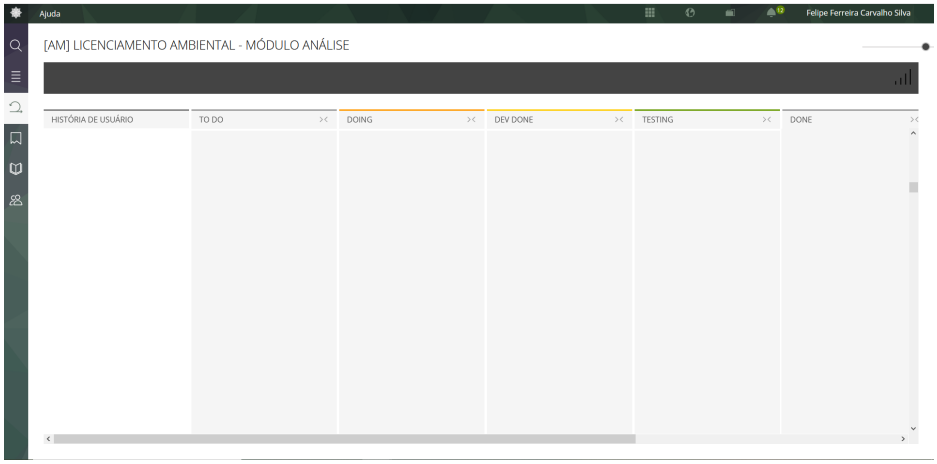
Essa ferramenta tem um papel fundamental na comunicação entre os membros do time de desenvolvimento. Ela permite com que os projetos a serem desenvolvidos e em desenvolvimento sejam cadastrados para haver um maior controle.

No LEMAF, o Taiga era utilizado em todos os projetos. Todas as atividades do *Product Backlog* eram cadastradas na ferramenta para que pudessem ser discutidas e divididas em *Sprints* quando necessário. A Figura 3.5 apresenta o quadro Kanban de uma *Sprint*. Durante a *Sprint* era possível verificar as atividades necessárias de serem realizadas na *Sprint* (*TO-DO*) e as que estavam em desenvolvimento (*DOING*). Na coluna *DEV DONE* ficavam as atividades finalizadas, mas que ainda necessitavam de testes. A coluna *TESTING* continha as atividades que

[16] https://www.atlassian.com/agile/kanban/boards
[17] https://www.taiga.io/

estavam sendo testadas. Caso algum erro fosse encontrado, a atividade era bloqueada até que o desenvolvedor corrigisse. Por fim, a coluna *DONE* apresentava as atividades finalizadas e testadas. Durante o desenvolvimento, poderia existir o surgimento de *bugs* no código e problemas no sistema, que eram registrados na aba de Problemas pelo analista de qualidade.

Figura 3.5 – Representação do Quadro Kanban no Taiga



Fonte: Taiga - LEMAF (2021)

4 ATIVIDADES DESENVOLVIDAS

Nesta seção é apresentado como foi o desenvolvimento do estágio, como era o dia a dia, os projetos que o discente participou, bem como as atividades que o mesmo desenvolveu.

4.1 Desenvolvimento do Estágio

O presente capítulo tem o propósito de mostrar como foi realizado o estágio, como eram divididas as tarefas e como era o dia a dia no laboratório.

O primeiro mês de estágio teve o foco em apresentar as tecnologias e *frameworks* utilizados dentro do LEMAF através de treinamentos. Primeiramente, o discente teve contato com o *framework* Scrum através de um curso ministrado pela Gerente de Projetos da Tribo PUMA. Nele foram apresentados os conceitos descritos no Capítulo 3 deste trabalho. Após isso, foram realizados diversos cursos focados em programação *Web*, como cursos de HTML e CSS básicos, Javascript, dentre outros.

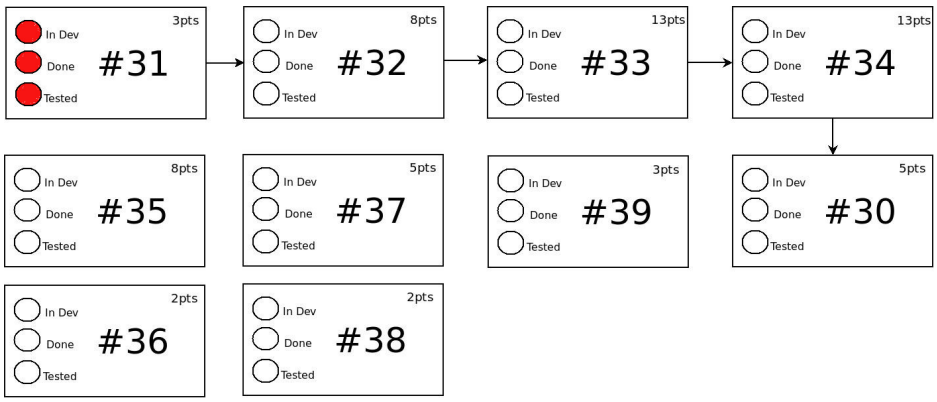
Também houve um curso focado nos conceitos de banco de dados, onde foi possível relembrar vários conceitos já vistos através da disciplina de Introdução a Sistemas de Banco de Dados, e aprender mais sobre como funcionava o processo de criação dos bancos de dados nos projetos do LEMAF. Durante o curso, também foi apresentada pela primeira vez a extensão PostGIS, do PostgreSQL, que era responsável por armazenar os dados geográficos dos sistemas.

Ao final dos treinamentos, o discente foi alocado na tribo PUMA, para fazer parte do *Squad* (*Scrum Team*) responsável pelos sistemas de Licenciamento Ambiental e Análise do Licenciamento Ambiental, ambos do estado do Amazonas. Toda tribo utilizava a metodologia ágil Scrum para organizar e desenvolver os projetos. O *Squad* era formado por um *Product Owner*, três desenvolvedores e um analista de qualidade de *software*. Em ambos os projetos o discente atuou como desenvolvedor.

O dia a dia do *Squad* seguia de acordo com os ritos do Scrum. As histórias em pendência de desenvolvimento estavam registradas no *Product Backlog*, e a partir disso era realizada a reunião de planejamento da *Sprint* (*Planning*).

Um problema encontrado durante as Sprints foi a concorrência entre as atividades, ou seja, algumas tarefas dependiam de outras que estavam na mesma *Sprint*, e enquanto um desenvolvedor não finalizasse a mesma, não era possível continuar as demais atividades. Para diminuir essa concorrência, foi feito um diagrama de fluxo de dependências (Figura 4.1) para visualizar quais atividades dependiam uma da outra. A direção das setas representa a direção de dependência. Atividades sem seta não possuem dependência. O uso do diagrama facilitava bastante na hora de começar novas atividades, pois era possível alocar atividades prioritárias e que tinham muitas dependências a desenvolvedores mais experientes e atividades que não tinham dependências a desenvolvedores inexperientes.

Figura 4.1 – Diagrama de Dependências



Fonte: Do autor (2021)

A reunião diária (*Daily Scrum*) também era um aspecto fundamental da *Sprint*. Todos os dias o *Squad* se reunia para apresentar o que havia sido feito no dia e avaliar o andamento das atividades.

Ao final da *Sprint*, era realizada a reunião de retrospectiva (*Sprint Retrospective*) para avaliar o resultado da mesma. A retrospectiva era um espaço onde

todo o *Squad* tinha a liberdade de apresentar o que tinha sido positivo na *Sprint*. Tal reunião tinha a finalidade de trazer *melhorias* para todo o projeto.

4.2 Ambientes de Desenvolvimento

Uma parte importante de ambos os projetos em que o discente participou era a divisão dos ambientes de desenvolvimento.

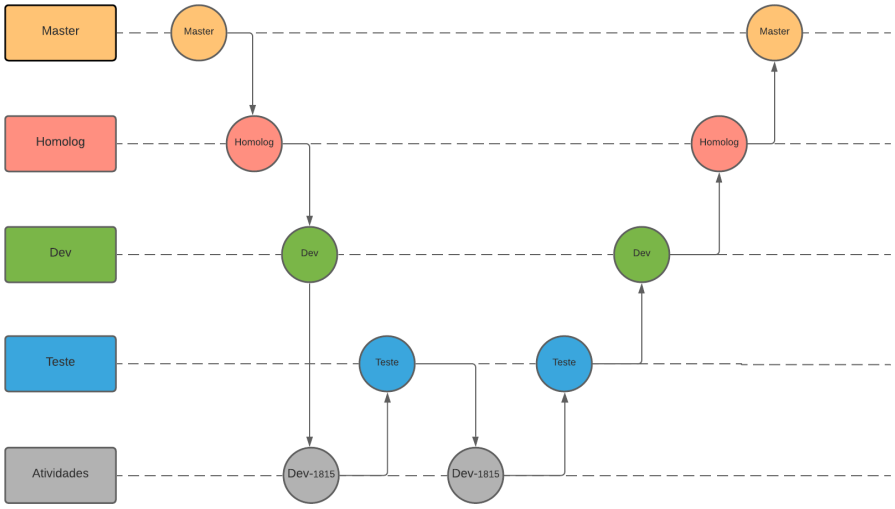
- ***Local*:** corresponde a máquina local do desenvolvedor, é onde as atividades da *Sprint* são realizadas.
- ***Teste*:** esse ambiente está disponível somente para o *Squad* responsável pelo desenvolvimento do sistema. Nele, os desenvolvedores aplicam as atividades realizadas durante a *Sprint* e o analista de qualidade avalia se as atividades foram realizadas de forma correta.
- ***Homologação*:** o ambiente de homologação é acessado pelo cliente e pelo *Squad* responsável pelo desenvolvimento da aplicação. Nele são apresentadas as alterações testadas nos ambientes anteriores e que estão passíveis de serem aplicadas em produção, caso ohaja aprovação do cliente.
- ***Produção*:** o ambiente de produção consiste na aplicação que está disponível para o público em geral, ou seja, após as melhorias serem aprovadas em todos os ambientes, elas são aplicadas no ambiente de produção.

Essa divisão permite que erros sejam encontrados e corrigidos antes das melhorias chegarem ao público final.

Todo código do projeto era armazenado e manejado através do GitLab. O projeto possuía quatro *branches* principais, a Master representava o código que estava presente no ambiente de produção, a *branch* Homolog possuia o código presente no ambiente de homologação, a Dev possuía o código estável presente no ambiente de teste e a branch Teste continha o código ainda não testado. Para a

realização das atividades da *Sprint*, era criada uma ramificação a partir da *branch* Dev com o código da atividade, exemplo: Dev-1815. Após realizada a atividade, o desenvolvedor fazia o *merge* da *branch* da atividade com a *branch* Teste. Após os testes, caso a atividade fosse aceita, o desenvolvedor fazia o *merge* da *branch* da atividade com a *branch* de Dev, isso garantia a integridade do código e minimizava a presença de erros. A Figura 4.2 representa o fluxo descrito.

Figura 4.2 – Fluxo de repositórios dos projetos



Fonte: Do autor (2021)

4.3 Desenvolvimento das Atividades

O presente capítulo visa apresentar os projetos e atividades específicas que o discente trabalhou. Durante esse tempo, o mesmo atuou como desenvolvedor em dois grandes projetos. O Licenciamento Ambiental do Amazonas (Licenciamento) e o Análise do Licenciamento Ambiental do Amazonas (Análise). Em ambos os projetos, o desenvolvedor tinha a escolha de desenvolver qualquer história disponível no escopo das *Sprints*, entretanto, havia um planejamento estratégico de acordo com a experiência de cada desenvolvedor para assumir uma tarefa.

No início do estágio, o discente foi instruído a resolver problemas mais simples, como resolver *bugs* relacionados a *labels* no *front-end* e promover melhorias nas páginas do sistema. Entretanto, após algumas semanas, o discente começou a participar ativamente no desenvolvimento do *back-end* do sistema, bem como executar atividades complexas e de alta importância.

Por conta do sigílo do código-fonte das aplicações, todas as representações de código nessa seção são somente inspiradas no código-fonte original.

4.3.1 Licenciamento Ambiental do Amazonas

Segundo Licenciamento-AM (2020), ”o Licenciamento Ambiental do Amazonas é um sistema eletrônico onde devem ser realizadas as solicitações de licenças ambientais para atividades que utilizem de recursos ambientais e que sejam consideradas efetiva ou potencialmente poluidoras, previstas na Lei nº 3.785/2012[1].”

O Licenciamento é uma aplicação que permite ao usuário cadastrar seus dados, de empreendedor e empreendimento, através de um formulário, que foi implementado utilizando o *framework* AngularJS, Pug, Javascript e CSS. Após finalizar o preenchimento, os dados são salvos por meio do *framework* Play e os mesmos são persistidos no banco de dados do sistema. Em seguida, com os dados cadastrados, o usuário pode solicitar sua licença de acordo com a atividade desenvolvida no empreendimento. Para cada atividade existe uma regra de cálculo da taxa que o solicitante deve pagar, a qual depende do tamanho do empreendimento, do nível de poluição que a atividade causa, do período de vigência, do tipo de licença que o usuário deseja obter, dentre outros fatores. Todas as regras de negócio são sigilosas e estão centradas no banco de dados, sendo assim, o sistema foi desenvolvido pensando em consumir as regras diretamente do banco de dados para fazer os cálculos.

[1] https://www.legisweb.com.br/legislacao/?id=243659

Ao ser inserido no Licenciamento, a primeira atividade do discente foi realizar a troca de diversas *labels* no sistema. Apesar de ser uma tarefa simples, foi importante para que o discente começasse a se familiarizar com o código em si e com o fluxo dos sistema. Após essa atividade, como o sistema estava em desenvolvimento e o aluno não tinha muito conhecimento do mesmo, ele foi responsável por resolver *bugs* no código, que estavam registrados na aba de Problemas do Taiga.

4.3.2 Análise do Licenciamento Ambiental do Amazonas

O sistema Análise do Licenciamento Ambiental do Amazonas é um módulo de acesso restrito aos funcionários do IPAAM. Ele é utilizado para fazer a validação de todas as solicitações de licenças ambientais feitas no sistema do Licenciamento.

O Análise é dividido em 5 perfis, que são eles:

- **Analista Geo:** é o primeiro perfil onde as solicitações chegam. O Analista Geo é responsável por validar qualquer tipo de inconsistência referente aos dados geográficos informados pelo solicitante, ou seja, ele deve verificar se o empreendimento sobrepõe alguma área de restrição, que pode ser uma área de preservação de Saium-de-coleira[2], um sítio arqueológico, aréas protegidas, zonas de amortecimento de terras indígenas, dentre outras. Se necessário, o Analista Geo vai a campo para verificar as informações. Caso haja inconsistências, o pedido é indeferido e a solicitação volta para o Licenciamento para o solicitante corrigir as informações.

- **Gerente:** o perfil de gerente é responsável apenas por validar as análises geo e técnicas. Em ambas as análises, o gerente somente pode visualizar o que foi descrito pelos analistas e tem o poder de aprovar, não aprovar ou solicitar

[2] https://www.nationalgeographicbrasil.com/animais/mamiferos/saium-de-coleira

ajustes. Se aprovado, o processo segue o fluxo, caso não seja, o processo vai para um novo analista para uma nova análise. Por fim, se o gerente solicitar ajustes, o processo volta para o analista que já estava cuidando do processo, para algum tipo de reajuste informado pelo gerente.

- **Analista Técnico:** após o gerente aprovar a análise geo, o analista técnico é responsável por validar todos os documentos informados pelo solicitante. Caso haja alguma inconsistência, assim como na análise geo, o analista indefere o pedido e o solicitante pode corrigir as informações incorretas. Caso esteja tudo de acordo, o processo continua.

- **Diretor:** visualiza as informações dos demais perfis e aprova ou não as solicitações. Independente da escolha do diretor, o pedido continua para o presidente.

- **Presidente:** visualiza o resultado da análise dos outros perfis, e tem o poder de aceitar que uma licença seja retirada. Caso seja aprovado, o processo volta para o licenciamento e o solicitante pode pagar a taxa para retirar a licença. Caso não seja, o processo é finalizado como não aprovado e o solicitante deve começar uma nova consulta.

Assim como o sistema de Licenciamento Ambiental, o Análise foi construído utilizando o *play Framework* na versão 1.5 para o desenvolvimento do *back-end*, bem como o *PostgreSQL* na versão 9.6. Para o *front-end* foi utilizado o *framework AngularJS* na versão 1.6, juntamente com HTML, Pug e CSS.

O módulo de Análise foi desenvolvido em paralelo com o sistema de Licenciamento Ambiental. Sendo assim, após resolver os *bugs* encontrados no Licenciamento, o discente começou a desenvolver atividades para o Análise.

4.3.2.1 Verificação de Status de Comunicados

Uma das primeiras atividades complexas que o discente teve contato foi a realização de um *cron job* (*job*) para a verificação de comunicados.*Cron jobs* são tarefas programadas para serem executadas em horários específicos ou em intervalos regulares (CLOUD, 2021). No código do Análise, ele tem a função de executar periodicamente uma função de verificação de *status* dos comunicados.

Os comunicados são e-mails enviados aos órgãos responsáveis informando uma inconsistência na solicitação da licença, por sobrepor uma área de restrição. Durante a análise geo, algumas restrições geram automaticamente comunicados aos órgão responsáveis. Um exemplo é quando o empreendimento está localizado em uma zona de amortecimento de terras indigenas. Nesse caso, um comunicado é enviado para a Fundação Nacional do Índio[3] (FUNAI), informando sobre as inconsistências encontradas.

Quando isso ocorre, o processo não pode continuar antes de uma resposta do órgão responsável. Para que isso fosse possível, o discente implementou um *job* que fazia verificações a cada 10 minutos para verificar se algum órgão já havia respondido à algum processo específico.

[3] www.funai.gov.br/index.php

Figura 4.3 – Representação da função de verificação de status dos comunicados

```
1   public void verificarComunicado() {
2
3       List<Comunicado> comunicados = Comunicado.findAll();
4       List<Comunicado> analisesGeo = comunicados.stream()
5                                       .filter(comunicado -> comunicado.ativo)
6                                       .map(comunicado -> comunicado.analisesGeo)
7                                       .filter(distintcByKey(analiseGeo -> analiseGeo.id))
8                                       .collect(Collectors.toList());
9
10      // Verifica para todas as análises Geo que possuem comunicados
11      for (AnalisesGeo analiseGeo:analisesGeo) {
12
13          Boolean podeTramitar = true;
14          // Função que busca os comunicados vinculados à análise Geo
15          List<Comunicado> comunicadosAnalise = Comunicado.findByAnaliseGeo(analiseGeo.id);
16
17          for (Comunicado comunicado: comunicadosAnalise) {
18
19              if (comunicado.ativo) {
20
21                  if (vencimentoPrazo(comunicado) && !comunicado.resolvido) {
22                      podeTramitar = false;
23                  } else {
24                      comunicado.ativo = false;
25                      comunicado.validateAndSave();
26                  }
27              }
28          }
29
30          if (podeTramitar) {
31              Tramita(analiseGeo);
32          }
33      }
34  }
```

Fonte: Análise do Licenciamento Ambiental do Amazonas (Adaptado) - LEMAF (2021)

A dificuldade do problema estava no fato de que o custo para verificar todas as análises geo, de 10 em 10 minutos, em busca de comunicados respondidos, era alto. Para resolver o problema, o discente pensou em uma forma de filtrar a quantidade de objetos analisados. A Figura 4.3 apresenta um pseudocódigo da atividade desenvolvida pelo discente. Na linha 3, são armazenadss em uma lista todos os comunicados disponíveis no sistema, em seguida, para descobrir quais análises geo possuem comunicados ativos, foi realizado um filtro na lista de comunicados, afim de selecionar essas análises. Esse filtro diminuía drasticamente a quantidade de análises que o *job* iria analisar. A partir disso, para cada análise, é verificado se existe um comunicado que foi respondido pelos órgãos responsáveis, ou se algum

deles já venceu o prazo de resposta, que é de 30 dias corridos a partir da data de envio do comunicado (linha 21). O processo pode tramitar para o Gerente apenas se o prazo tenha vencido ou o órgão tenha respondido.

4.3.2.2 Parecer Jurídico

Outra atividade desenvolvida pelo discente foi o parecer jurídico. Antes do processo chegar para a análise técnica, era necessária a validação jurídica de alguns documentos. Por esse motivo, após o Gerente aprovar a análise geo, um e-mail é disparado para o Diretor Jurídico do IPAAM, informando da necessidade de análise jurídica para os documentos apresentados.

O discente foi responsável por implementar o código do *back-end*. Portanto, foi necessário primeiramente criar o mapeamento da entidade na *model*, chamada ParecerJuridico. A Figura 4.4 apresenta um pseudocódigo de como é representado o parecer. É possível perceber o uso do *JPA Hibernate* para mapear a entidade com o banco de dados. Na *model* também foram criados os métodos necessários para a criação, atualização, leitura e exclusão dos pareceres. Também foi implementada a *controller*, denominada PareceresJuridicos, que é responsável por receber as requisições vindas do *front-end*.

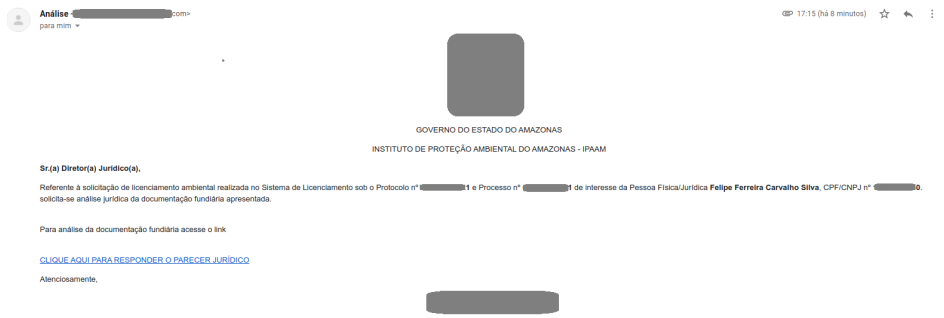
Figura 4.4 – Representação da *model* do parecer jurídico

```java
@Entity
@Table(schema="analise", name="parecer_juridico")
public class ParecerJuridico extends GenericModel {

    @Id
    public Long id;

    @ManyToOne
    @JoinColumn(name="id_analise_geo")
    public AnaliseGeo analiseGeo;

    @ManyToOne
    @JoinColumn(name="id_analise_tecnica")
    public AnaliseTecnica analiseTecnica;

    @OneToOne
    @JoinColumn(name = "id_documento_fundiario", referencedColumnName = "id")
    public Documento documentoFundiario;

    @Column(name="data_cadastro")
    @Temporal(TemporalType.TIMESTAMP)
    public Date dataCadastro;

    @Column(name="data_resposta")
    @Temporal(TemporalType.TIMESTAMP)
    public Date dataResposta;

    @Column(name="parecer")
    public String parecer;

    @Column(name="resolvido")
    public Boolean resolvido;

    @Column(name="ativo")
    public Boolean ativo;

```

Fonte: Análise do Licenciamento Ambiental do Amazonas (Adaptado) - LEMAF (2021)

Após a estruturação do *back-end*, o discente foi responsável por desenvolver um método que dispara um e-mail para o Diretor Jurídico contendo o *link* para responder ao parecer jurídico. O e-mail foi construído utilizando HTML para estruturar o mesmo e CSS para estilizá-lo. A Figura 4.5 representa um exemplo de e-mail enviado.

Figura 4.5 – Exemplo de e-mail enviado ao Diretor Jurídico



Fonte: Análise do Licenciamento Ambiental do Amazonas - LEMAF (2021)

Para que fosse possível responder ao parecer jurídico, foi necessária a construção de uma página Web com as informações necessárias. Para realizar a tarefa, foram utilizadas as tecnologias de *front-end* descritas neste trabalho. Para desenvolver as telas, três arquivos foram vitais nessa construção. O primeiro contém os serviços necessários para se comunicar com a *controller* no *back-end*. Nele são definidas as funções possíveis para o serviço, os métodos HTTP, juntamente com as rotas para o *back-end*. A Figura 4.6 apresenta um exemplo de construção da *service*, nela é possível perceber a definição do método salvarParecer, que contém uma requisiçao POST, a rota e os parâmetros esperados pelo método na *controller*.

Figura 4.6 – Representação da *service* do parecer jurídico

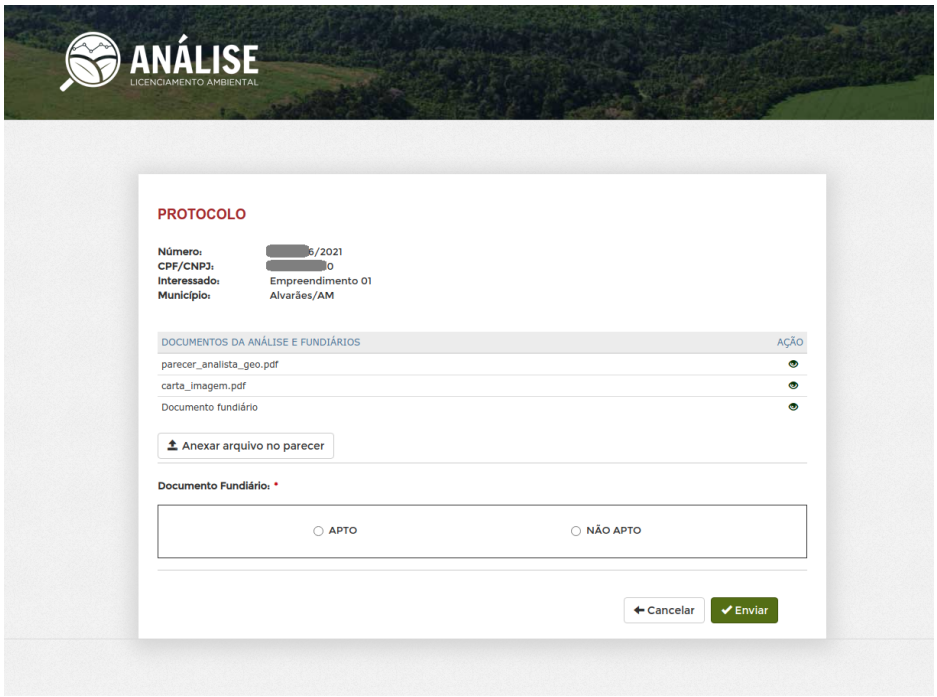
```
var ParecerJuridicoService = function(request,config) {

    this.salvarParecer = function(params){
        return request
            .post(config.BASE_URL() + 'parecerJuridico/salvarParecerJuridico' , params);
    };
};

exports.services.ParecerJuridicoService = ParecerJuridicoService;
```

Fonte: Análise do Licenciamento Ambiental do Amazonas (Adaptado) - LEMAF (2021)

O próximo arquivo criado foi a *controller*. Esta é responsável por definir as variáveis e métodos responsáveis por armazenar e manipular os dados vindos do *back-end* através da *service*.

Por fim, foi criado o arquivo Pug para estruturar, estilizar e apresentar os dados na tela. Por meio da service e da controller, foi possível buscar e armazenar os dados necessários. Na tela de resposta, o Diretor consegue visualizar o número do processo e os dados pessoais do empreendedor. Também é possível visualizar os documentos necessários para a análise, ao clicar no botão de visualizar, uma nova aba é aberta com o documento em questão. Se necessário, o Diretor pode anexar um arquivo ao parecer. Após a análise, o Diretor informa se os documentos estão aptos ou não para continuar o processo.

Figura 4.7 – Tela de resposta do parecer jurídico



Fonte: Análise do Licenciamento Ambiental do Amazonas - LEMAF (2021)

4.3.3 Refatoração do Empreendimento

Um passo importante no processo de solicitação de licenças é o cadastro do empreendedor e de seu respectivo empreendimento. Para solicitar a licença, é necessário que o empreendedor e o empreendimento estejam cadastrados no Cadastro Unificado do Amazonas (Entrada Única).

O Cadastro Unificado é um portal que reúne informações de empreendimentos e empreendedores. Os dados presentes no portal vêm através de uma conexão com a API da Junta Comercial do Estado do Amazonas[4] (JUCEA) e são salvos no banco de dados do sistema.

Os dados utilizados no Licenciamento eram inicialmente carregados de acordo com o portal do Entrada Única. Entretanto, ao cadastrar um novo empreendimento, os dados de empreendedor e empreendimento eram salvos novamente no banco de dados do Licenciamento. Por conta disso, inconsistências começaram a acontecer, pois não existia uma sincronia dos dados que pudesse garantir a integridade dos mesmos. Sendo assim, diversos chamados de produção eram abertos por existir divergência de dados, gerando transtorno para o administrador de banco de dados da Tribo e para o *Squad*.

Como o custo para mapear todos os dados duplicados e resolver os problemas pontuais dos clientes estava ficando muito alto, a solução mais correta era a de centralizar os dados pessoais e de empreendimento no Cadastro Unificado e o Licenciamento apenas consumir esses dados. Porém, a complexidade dessa refatoração era alta, pois gerava impactos em diversos sistemas que se comunicavam com o Licenciamento Ambiental, ou seja, era necessário alterar os sistemas que consumiam os dados do banco de dados do Licenciamento Ambiental para começarem a consumir os dados do banco de dados do Cadastro Unificado. O módulo de Análise do Licenciamento era um dos mais afetados.

[4] http://www.jucea.am.gov.br/

Para a solução do problema, algumas tabelas tiveram que ser deletadas do banco de dados do Licenciamento e todo mapeamento precisou ser refeito no *back-end*. Foi necessário inserir uma nova tabela denominada ”empreendedor” no banco de dados do Cadastro Unificado. Todas as tabelas referentes aos dados pessoais do empreendedor foram removidas do Licenciamento. A entidade Empreendimento, pertencente ao Licenciamento se tornou apenas uma referência do Empreendimento por completo, que está salvo no Entrada Única, garantindo assim que qualquer dado de empreendimento e pessoa que o representa, ou que esteja vinculada ao empreendimento, seja salvo somente no Cadastro Unificado. Assim, o Licenciamento e todos os outros sistemas que dependem desses dados apenas o consumem.

Após remodelar os bancos de dados, foi necessário mapear as entidades de forma correta em todo o *back-end*, o que causou problemas em diversas partes do código. Foi realizada, então, uma análise dos problemas decorrentes da refatoração e os mesmos foram divididos em algumas *Sprints*. O discente participou do processo de refatoração e da resolução de alguns desses problemas.

Para realizar a atividade, as matérias de Estruturas de Dados e Paradigmas de Linguagens de Programação permitiram que o aluno tivesse conhecimento teórico de conceitos como encapsulamento, classes, herança, sobrecarga e sobrescrita, dentre outros, tornando possível a adaptação do *back-end*. A disciplina Introdução a Sistemas de Bancos de Dados também foi de grande importância, pois o discente teve conhecimento de diversos conceitos, como entidades e relacionamentos, aprendeu sobre o modelo relacional e suas características e também a como realizar consultas SQL.

5 CONSIDERAÇÕES FINAIS

Durante o estágio, o aluno participou ativamente no desenvolvimento de diversas atividades, desde a solução de problemas, como correções de *bugs* e *labels* incorretas, até atividades complexas que eram vitais para a continuação do projeto. Essa evolução das atividades veio por conta do aumento da confiança do desenvolvedor e o auxílio dos membros do *Squad*.

A utilização do *framework* Scrum nos projetos permitiu que o estagiário tivesse um papel importante no desenvolvimento dos *softwares*, participando da criação das atividades, do planejamento das *Sprints* e na participação ativa nas reuniões diárias.

A graduação teve um papel fundamental durante o tempo de estágio. O conhecimento teórico e prático adquirido pela realização das disciplinas do curso de Ciência da Computação auxiliaram no processo de desenvolvimento das atividades. As disciplinas do início do curso serviram como base teórica para que os códigos desenvolvidos tivessem a melhor estrutura possível. As disciplinas relacionadas a bancos de dados também foram importantes, pois como o sistema estava sendo construído e as regras de negócio ficavam armazenadas no banco de dados, houve a necessidade de utilizar os conhecimentos adquiridos nas disciplinas.

A experiência obtida no estágio permitiu que o discente crescesse na área de desenvolvimento de *software*. Permitindo que o mesmo subisse de cargo no LEMAF e assumisse mais responsabilidades nos projetos.

Em meio a pontos positivos, algumas questões observadas pelo discente podem ser melhoradas, com o intuito de melhorar os processos internos do LEMAF:

- a falta de descrição detalhada nas atividades presentes no *Product Backlog* era recorrente e causava transtornos durante as reuniões de planejamento, pois além de ser difícil definir o que era pra ser feito, acontecia da tarefa não ser feita de acordo com o que o cliente queria, causando retrabalho e es-

tresse no time de desenvolvimento. Portanto, uma documentação detalhada é importante para que não haja atrasos nas reuniões e os retrabalhos sejam minimizados;

- a alocação de membros sem experiência em momentos críticos do projeto era outro problema. Por falta de pessoas disponíveis, eram alocados novos membros para realizar tarefas complexas. Isso causava um estresse não somente no projeto, mas também no novo membro, que tinha de aprender todo o fluxo do sistema, entender o que era necessário a ser desenvolvido e as tecnologias utilizadas. Sendo assim, um planejamento melhor deveria ser feito, com um treinamento maior dos novos membros antes de ingressarem nos projetos.

Ao final do estágio, foi possível perceber o crescimento do aluno:

- antes de ingressar no estágio, o discente não tinha conhecimento em desenvolvimento de aplicações web. Entretanto, após as experiências vividas neste período, o aluno adquiriu a habilidade de desenvolver sistemas, desde a construção de um *back-end* robusto, até o desenvolvimento de telas funcionais e responsivas no *front-end*;

- a convivência em um ambiente que implementa o *framework* Scrum permitiu que o aluno criasse uma familiaridade com o framework e aprendesse na prática os conceitos citados neste trabalho. Com isso, foi possível visualizar os pontos positivos de se aplicar o *framework* em um contexto de desenvolvimento de *softwares*;

- além do crescimento técnico, o estágio trouxe uma experiência de vida para o aluno. O dia a dia no laboratório permitiu que o discente desenvolvesse habilidades de comunicação e capacidade de resolver problemas que serão levados para a vida toda.

REFERÊNCIAS BIBLIOGRÁFICAS

CLOUD, G. *Como programar tarefas com o Cron para Java 8*. 2021. Disponível em: <https://cloud.google.com/appengine/docs/standard/java/config/cron?hl=pt-br>. Acesso em: 08/04/21.

CSS, M. *CSS*. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/CSS>. Acesso em: 12/04/21.

DEITEL, P.; DEITEL, H. *Java - Como Programar*. 10. ed. Londres: Pearson, 2015. 13–13 p. ISBN 9788543004792.

GIT. *About Git*. 2021. Disponível em: <https://git-scm.com/>. Acesso em: 12/04/21.

GITLAB. *About GitLab*. 2021. Disponível em: <https://about.gitlab.com/>. Acesso em: 12/04/21.

GOOGLE. *What Is AngularJS?* 2020. Disponível em: <https://docs.angularjs.org/guide/introduction>. Acesso em: 01/04/21.

LEMAF. *LEMAF - Laboratório de Estudos em Manejo Florestal*. 2019. Disponível em: <https://web.archive.org/web/20191002001755/http://www.lemaf.ufla.br/>. Acesso em: 20/04/21.

LICENCIAMENTO-AM. *Licenciamento Ambiental - AM*. 2020. Disponível em: <http://sistemas.ipaam.am.gov.br/portal-ipaam/manual_licenciamento_am.pdf>. Acesso em: 30/03/21.

MEDIUM, U. H. *Pug.js to make your life easier with HTML templates*. 2018. Disponível em: <https://medium.com/jspoint/pug-js-to-make-your-life-easier-with-html-templates-9c62273626e0>. Acesso em: 31/03/21.

MOZILLA. *About Javascript*. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/About_JavaScript>. Acesso em: 31/03/21.

MOZILLA. *HTML*. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML>. Acesso em: 31/03/21.

PLAY. *Play 1.5.x documentation*. 2021. Disponível em: <https://www.playframework.com/documentation/1.5.x/home>. Acesso em: 31/03/21.

POSTGRESQL. *About PostgreSQL*. 2021. Disponível em: <https://www.postgresql.org/about/>. Acesso em: 01/04/21.

SCRUM. *What is Scrum*. 2020. Disponível em: <https://www.scrum.org/resources/what-is-scrum>. Acesso em: 04/04/21.

SCRUMGUIDES.ORG. *Scrum Guide*. 2020. Disponível em: <https://scrumguides.org/scrum-guide.html>. Acesso em: 04/04/21.

TAIGA. *About Taiga*. 2021. Disponível em: <https://www.taiga.io/>. Acesso em: 01/04/21.