GIOVANI FERREIRA ANDRADE BOTELHO

UTILIZAÇÃO DE METODOLOGIAS E FERRAMENTAS NA ÁREA DE ANÁLISE DE QUALIDADE DE SOFTWARE PARA REALIZAÇÃO DE TESTES EM SISTEMAS WEB NA AGÊNCIA ZETTA UFLA

LAVRAS – MG

2021

GIOVANI FERREIRA ANDRADE BOTELHO

UTILIZAÇÃO DE METODOLOGIAS E FERRAMENTAS NA ÁREA DE ANÁLISE DE QUALIDADE DE SOFTWARE PARA REALIZAÇÃO DE TESTES EM SISTEMAS WEB NA AGÊNCIA ZETTA UFLA

Relatório de estágio supervisionado apresentado ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso para obtenção do título de Bacharel em Ciência da Computação.

Prof. DSc. André Pimenta Freire
Orientador

B.Sc. Pâmela Andrade de Almeida
Coorientadora

LAVRAS – MG
2021

Ficha catalográfica elaborada pela Coordenadoria de Processos Técnicos

da Biblioteca Universitária da UFLA

Botelho, Giovani Ferreira Andrade

UTILIZAÇÃO DE METODOLOGIAS E FERRAMENTAS NA ÁREA DE ANÁLISE DE QUALIDADE DE SOFTWARE PARA REALIZAÇÃO DE TESTES EM SISTEMAS WEB NA AGÊNCIA ZETTA UFLA / Giovani Ferreira Andrade Botelho. 1^a ed. rev., atual. e ampl. – Lavras : UFLA, 2021.

52 p. : il.

Trabalho de conclusão de curso (Graduação) - Curso de Ciência da Computação–Universidade Federal de Lavras, 2021.

Orientador: Prof. DSc. André Pimenta Freire.

Bibliografia.

1. TCC. 2. Monografia. 3. Qualidade de Software. I. Universidade Federal de Lavras. II. Título.

CDD-808.066

GIOVANI FERREIRA ANDRADE BOTELHO

UTILIZAÇÃO DE METODOLOGIAS E FERRAMENTAS NA ÁREA DE ANÁLISE DE QUALIDADE DE SOFTWARE PARA REALIZAÇÃO DE TESTES EM SISTEMAS WEB NA AGÊNCIA ZETTA UFLA

Relatório de estágio supervisionado apresentado ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso para obtenção do título de Bacharel em Ciência da Computação.

APROVADA em 14 de Maio de 2021.

Prof. DSc. Paulo Afonso Parreira Junior - UFLA
Prof. DSc. Renata Teles Moreira - UFLA

Prof. DSc. André Pimenta Freire
Orientador

B.Sc. Pâmela Andrade de Almeida
Co-Orientadora

LAVRAS – MG
2021

RESUMO

Devido à vasta dimensão territorial do Brasil, a análise estratégica do setor agropecuário se torna dificultosa para quem formula questões relacionadas a políticas públicas. Embora existam vários sistemas e bases de dados geoespaciais sobre a agropecuária nacional, não há uma integração entre esses dados. Partindo desta premissa, o projeto, por parte do Ministério da Agricultura, Pecuária e Abastecimento do Brasil contempla o projeto Observatório da Agropecuária Brasileira. Em meio a isso, o presente relatório descreve a experiência de atuação pelo discente como analista de qualidade no projeto Plataforma Geoespacial do Observatório da Agropecuária Brasileira, em um estágio supervisionado na Agência Zetta UFLA. O relatório apresenta uma breve descrição do projeto, para contextualização, assim como as rotinas, metodologias e ferramentas utilizadas na realização dos testes durante o período de desenvolvimento do produto. Desta maneira, espera-se uma certificação da qualidade do software, garantindo que o produto final atenda todas as expectativas e todos os requisitos definidos inicialmente.

Palavras-chave: teste de software, Engenharia de Software, qualidade.

ABSTRACT

Due to the vast territorial dimension of Brazil, the strategic analysis of the agricultural sector becomes difficult for those who formulate questions related to public policies. Although there are several geospatial systems and databases on national agriculture, there is limited integration between these data. Based on this premise, the project of the Ministry of Agriculture. Livestock and Supply of Brazil contemplates the project Observatório da Agropecuária Brasileira. Within this context, this report describes the student's experience of acting as a quality analyst in the Geospatial Platform project of the Brazilian Agricultural Observatory, in a supervised internship at Zetta UFLA Agency. The report presents a brief description of the project, for contextualization, as well as the routines, methodologies and tools used in carrying out the tests during the product development period. In this way, a certification of the quality of the software is expected, ensuring that the final product meets all expectations and all requirements defined initially.

Keywords: software testing, Software Engineering, quality.

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

O Brasil, por se tratar de um país de território com dimensões continentais, torna extremamente dificultosa a análise global e estratégica de diversos setores. Isso acaba por trazer obstáculos a quem formula políticas públicas. O setor agropecuário, por estar presente em todo o território nacional, enfrenta essa mesma dificuldade. Contudo, atualmente, existe uma grande quantidade de sistemas e de bases de dados geoespaciais sobre a agropecuária nacional disponíveis nos diversos órgãos de governo e no setor privado. Esses sistemas são de grande importância na contextualização de cenários atuais e futuros, sendo ferramentas de amparo técnico nas tomadas de decisão.

Embora exista essa enorme quantidade de dados, não há uma integração entre eles, gerando uma fragilidade no que tange a informações completas e assertivas para o planejamento estratégico do governo. Partindo desta contextualização, observa-se que se trata de um assunto já levantado em discussões do projeto, com ações tomadas, mas de forma isolada, atingindo objetivos específicos. No entanto, essa visão macro do setor ainda não foi alcançada. Motivo pelo qual o [1] Plano Estratégico Corporativo presente na gestão do Ministério da Agricultura, Pecuária e Abastecimento do Brasil contempla o projeto [2] Observatório da Agropecuária Brasileira, para o qual a consultoria solicitada certifica a necessidade da contratação de serviços técnicos especializados para dimensionar este conjunto de dados espaciais e não espaciais. Um produto que já se encontra em desenvolvimento pela a Agência Zetta Ufla.

Segundo Sommerville (2007), praticamente todos os países dependem de sistemas complexos baseados em computadores. Esses sistemas, com softwares de controle, já automatizam a manufatura e as distribuições industriais, assim como os sistemas financeiros, além de contemplar as infraestruturas e serviços nacionais. Por este motivo, produzir e manter o software com qualidade é essencial para o funcionamento da economia nacional e internacional.

Com isso, o presente relatório se mostra como uma análise das atividades realizadas pelo discente, atuando na Agência Zetta UFLA, durante sua primeira experiência de trabalho no desenvolvimento de um sistema de uso real, sendo ele a Plataforma Geoespacial do Observatório da Agropecuária Brasileira. Neste projeto, o estagiário atuou como analista de qualidade

[1] Relatório interno obtido via Agência Zetta UFLA.

[2] O Observatório da Agropecuária Brasileira é um Projeto Estratégico Corporativo concebido e executado pelo Mapa/SDI/CGIE, criado para integrar e sistematizar informações da agropecuária brasileira.

em uma equipe de desenvolvimento. Este cargo, que possibilitou ótimas oportunidades para praticar conceitos teóricos abordados durante o período de graduação, preparou o discente para assumir novos desafios, tanto na atuação profissional quanto na acadêmica. Uma experiência de suma importância, já que o mercado de trabalho constantemente exige que seus colaboradores possuam certas qualificações e habilidades, para que possam ser de fato contratados.

1.2 Objetivos

O objetivo deste relatório é apresentar a trajetória do discente, do curso de bacharelado em Ciência da Computação na Universidade Federal de Lavras (UFLA), durante o período do dia 1 de novembro ao dia 12 de Abril de estágio supervisionado oferecido pela Agência Zetta UFLA, descrevendo a aplicação de processos e atividades para alcançar o objetivo de entrega do sistema Plataforma Geoespacial do Observatório da Agropecuária Brasileira. A experiência de estágio envolve tanto os conceitos estudados durante a graduação, quanto os adquiridos durante o próprio processo de estágio. Por ser uma vivência prático-pedagógica, o estágio se caracterizou como uma capacitação, proporcionando ao estudante crescer e amadurecer profissionalmente na área de atuação, assim como a formação de sua carreira no mercado de trabalho. Com isso, dentre a finalidade já citada e as outras que o estágio busca atingir, estão:

- Atuar em atividades de Garantia da Qualidade de Software no contexto do projeto.
- Proporcionar oportunidade para aprendizado prático com aplicação e complementação dos conhecimentos adquiridos no curso de graduação.
- Minimizar a dificuldades encontradas na transição da vida acadêmica para a profissional, no mercado de trabalho.
- Desenvolver habilidades de trabalho em grupo, considerando um maior controle sobre as atitudes e posturas ao se relacionar com os outros membros da equipe.
- Entendimento sobre o funcionamento de processos, normas e diretrizes das metodologias de desenvolvimento de software.

1.3 Estrutura do texto

Este documento está organizado da seguinte forma. Esta introdução aborda a contextualização e os objetivos do relatório de estágio. O Capítulo 2 descreve o processo organizacional

do ambiente de trabalho, assim como suas áreas de atuação. Já o Capítulo 3, tem o propósito de informar sobre as estratégias gerais de pesquisa, no qual, expõe quais métodos, ferramentas e *frameworks* foram utilizados para alcançar o nível de qualidade buscado no software.

O Capítulo 4 destaca os detalhes do projeto Plataforma Geoespacial, explicando toda a base de seu funcionamento, para que o leitor se familiarize com as regras e funcionalidades do sistema. No Capítulo 5, é relatado todo o roteiro de atividades realizadas pelo discente durante o estágio, ditando a forma como são desenvolvidas. Por fim, o Capítulo 6 trata das considerações finais sobre todo o processo de atuação, analisando os resultados obtidos através dos conhecimentos abordados e adquiridos no estágio.

2 A AGÊNCIA ZETTA UFLA

Aprovada pelo Conselho Universitário da UFLA e sendo resultado de uma iniciativa da Pró-Reitoria de Pesquisa, a Agência Zetta UFLA nasceu com o intuito de gerar valores que possam contribuir para a melhoria da sociedade através da utilização de inovação e tecnologias. Atualmente, a Agência Zetta UFLA é composta por um time pluridisciplinar da Universidade, mas teve sua origem no Laboratório de Estudos e Projetos em Manejo Florestal - Lemaf.

A Agência Zetta UFLA tem a missão de potencializar o conhecimento gerado pelos pesquisadores da UFLA, para transformá-lo em produtos inovadores que ultrapassam os limites da pesquisa e promovem o desenvolvimento sustentável do país. A Agência Zetta UFLA busca focar na integração institucional, permitindo maior cooperação entre as diferentes áreas acadêmicas, para que possam trabalhar em conjunto com geotecnologias e sistemas inteligentes vinculados ao agronegócio e a sustentabilidade ambiental. Para alcançar esse feito, projetos são desenvolvidos abrangendo diferentes regiões e estados, como o norte com Acre, Amapá, Amazonas, Pará, Rondônia e Tocantins; centro-oeste com o Mato Grosso do Sul, Goiás e Distrito Federal; sul com Paraná, Santa Catarina e Rio Grande do Sul e sudeste com Espírito Santo e Minas Gerais.

Além da atuação em gestão territorial e governamental, a Agência Zetta UFLA possui um conjunto de soluções que também atende diferentes mercados de tecnologia da informação, abrangendo serviços que promovem maior agilidade, transparência e produtividade aos processos, como:

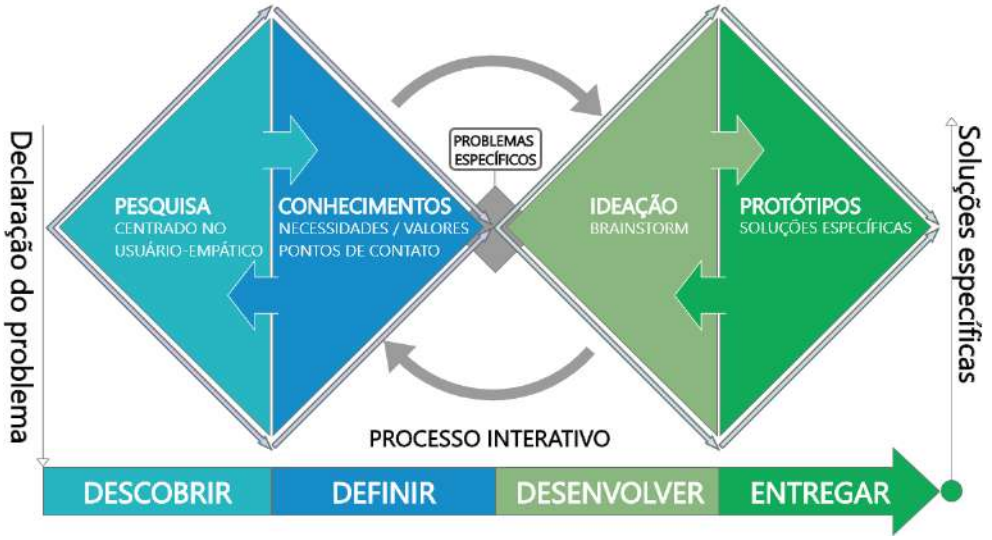
- Inovação

  Com o conceito de que não é necessária a dependência da tecnologia para agregar valor tanto para os indivíduos quanto para as organizações, a Agência Zetta UFLA reúne ferramentas e metodologias, utilizando o método *Flow*, para a execução do *Design Thinking*, através do método duplo diamante. Uma abordagem "humanizada" que ajuda na busca de alternativas para resolver problemas em qualquer área de negócio, ilustrado na Figura 2.1.

- Geotecnologia

  Sendo muito mais que uma simples coleta de dados, a Agência Zetta UFLA promove qualidade na disponibilização dos dados com referência geográfica, de forma rápida, segura e com resultados imediatos. Através da coleta, processamento e análise, concentra-se na construção de bases de dados, transformando-os em informações de grande valor para os

Figura 2.1 – *Double Diamond*



Fonte: Adaptado de Vancouver (2014)

clientes em diferentes áreas de atuação, como meio ambiente, gestão pública, agronegócio e florestal.

- Sistemas Inteligentes

  Possibilita facilitar a solicitação de determinados serviços, proporcionando técnicas mais assertivas. Com isso, são gerados os detalhes a respeito da tomada de decisão dos usuários, cuja análise de comportamento, detecção de fraudes; através da Inteligência Artificial (IA), otimizam ao máximo o sistema, auxiliando a produção de diagnósticos automatizados sobre os processos.

- Agricultura Digital

  Credenciada como Unidade de Agricultura Digital da Empresa Brasileira de Pesquisa e Inovação Industrial (Embrapii), a Agência Zetta UFLA caracteriza-se como uma organização social que atua por meio da cooperação com instituições de pesquisas científicas e tecnológicas, públicas ou privadas. A agência pode executar projetos juntamente às empresas do agronegócio brasileiro. O modelo de negócio Embrapii oferece vantagens como:

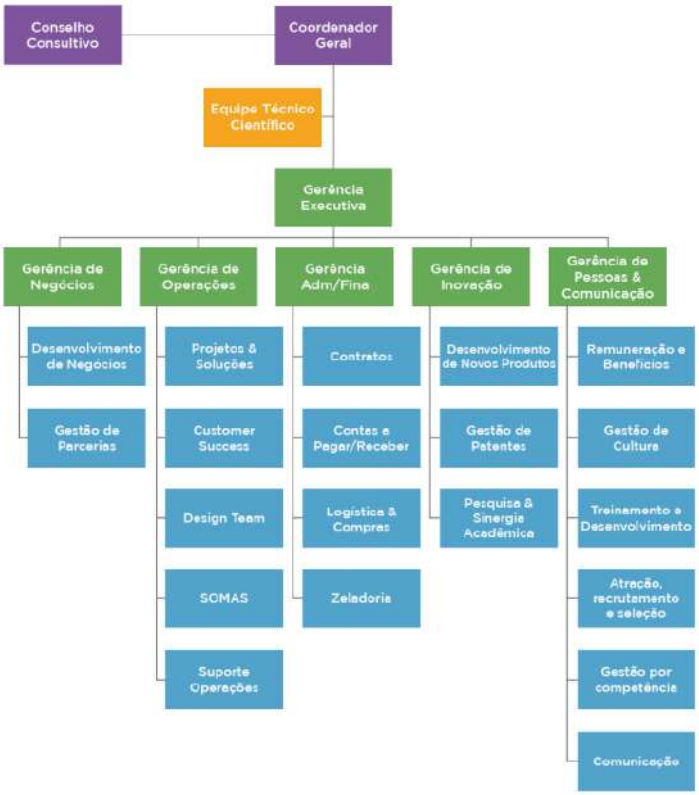
  – A empresa negocia o projeto direto com a Agência Zetta UFLA.

  – A aprovação e contratação é feita diretamente entre a empresa e a Agência Zetta UFLA.

- Os recursos já estão disponíveis para uso imediato na Agência Zetta UFLA.
- Fluxo contínuo: a qualquer momento a empresa pode realizar projetos, sem esperar um edital, diminuindo muito a burocracia para iniciar o ciclo de inovação.

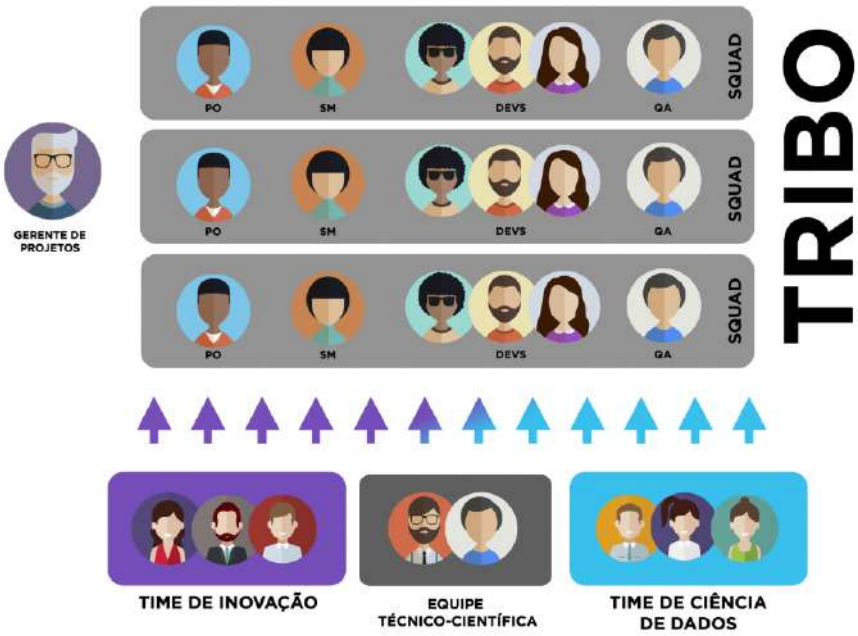
Composta por uma equipe pluridisciplinar, a Agência Zetta UFLA conta com mais de 200 colaboradores, dentre eles, os de corpo técnico científico, docentes renomados da UFLA e alunos de graduação ou pós-graduação. De acordo com seu processo organizacional, mostrado na Figura 2.2, ela compõe seis diferentes tipos de gerência, das quais cinco são coordenadas pela Gerência Executiva. Dentre os diferentes setores administrativos, a Gerência de Operações se encarrega do desenvolvimento dos produtos e soluções para a resolução de problemas reais no processo agroindustrial brasileiro. Para isso, a Agência Zetta UFLA conta com sub equipes (*Squads*), nas quais se agrupam em diferentes esferas de atuação (tribos), ilustrada na Figura 2.3, supervisionadas por um(a) Gerente de Projetos. Cada *Squad* é composta por *Product Owner*, desenvolvedores(as) e um(a) analista de qualidade, além do auxílio do time de inovação, equipe técnico-científica e do time de ciência de dados.

Figura 2.2 – Processo Organizacional- Agência Zetta UFLA



Fonte: Agência Zetta UFLA (2021)

Figura 2.3 – Esquema organizacional das tribos - Agência Zetta UFLA



Fonte: Agência Zetta UFLA (2021)

3 METODOLOGIAS E FERRAMENTAS UTILIZADAS

3.1 Framework

Framework possui uma definição que vai além do mercado de software. Tem como destaque, o papel de facilitar o entendimento e comunicação entre participantes de uma situação que possam ter diferentes perspectivas, auxiliando no processo de tomada de decisão e de resolução de problemas (ODEH; KAMM, 2003). Ele consiste em uma estrutura básica que compõe práticas, técnicas, ferramentas ou conceitos, com o objetivo de orientar ou gerenciar a solução de problemas recorrentes e tarefas repetitivas. Através de uma abordagem genérica e ágil, permite que quem o utiliza foque os esforços em sua execução, minimizando a preocupação da forma como o processo ou a atividade deve ser implementada. Isso dispensa a necessidade de terceirizar determinadas partes do projeto, justificadas pela falta de conhecimento na área de mercado. Justamente por possuir um conjunto pronto de abordagens que otimizam os resultados, no projeto, a utilização de *frameworks* foi voltada ao contexto de processos. Sendo eles, o *framework Scrum* e o *framework Kanban*.

Tendo a finalidade de auxiliar, o *framework* possui as documentações, que são voltadas para orientar os profissionais sobre a maneira que o conteúdo deve ser utilizado. *Frameworks* facilitam tanto na agilidade da execução das funcionalidades, quanto na didática. A utilização de *frameworks* tem sido cada vez mais acessível, independente dos níveis de experiência. Ele torna-se ainda melhor quando a documentação é ampla, completa e possui fácil entendimento. Dessa forma, antes mesmo de começar a utilizá-lo, é de grande valia analisar sua documentação. Um *framework* existe para facilitar o trabalho, e a documentação possui um grande papel nessa simplicidade.

Outro fator muito importante, que define a melhor condução, performance e comportamento do *framework*, é a checagem das atualizações presentes em seu conteúdo. Com isso, maiores serão as chances de executá-lo com segurança e da melhor forma possível, garantindo qualidade na resolução do problema e no alcance do objetivo.

3.2 Framework Scrum

Scrum é um *framework* de metodologia ágil, simples e flexível, com a finalidade de auxiliar a resolução de produtos complexos e adaptativos, de forma produtiva e criativa (SCRUM-GUIDES.ORG, 2021). O ciclo do *Scrum* é ilustrado na Figura 3.1, apresentando o refinamento

do produto com o *Backlog* da *Sprint*, além dos ciclos de desenvolvimento e ritos que devem ser realizados durante todo o processo, até alcançar o incremento de produto planejado. A metodologia ágil não está diretamente relacionada somente ao produto, mas sim ao processo, levando em consideração o valor de negócio que ele representa para os clientes envolvidos. Para isso, o *Scrum* maximiza a entrega de software de modo eficaz, adaptando-se às mudanças, onde as funcionalidades de maior valor são desenvolvidas antecipadamente, enquanto é discutida a necessidade, ou não, das que possuem menos prioridade (PRIKLADNICKI; WILLI; MILANI, 2014).

Figura 3.1 – Ciclo do Scrum

Fonte: Adaptado de Pressman (2011)

Utilizando uma abordagem interativa e incremental, o *Scrum* garante, com frequência, valor nas entregas das partes, potencialmente funcionais, do produto, auxiliando a redução de possíveis riscos sobre o projeto. Sendo assim, o *Scrum* divide os projetos em diferentes ciclos, com a duração necessária para todo o esforço de desenvolvimento, chamados de *Sprints*. Cada nova *Sprint* começa após a conclusão da anterior, de forma imediata. Além da execução das atividades de desenvolvimento, as *Sprints* são compostas por diferentes reuniões, a de planejamento das atividades, a diária, de revisão e de retrospectiva; que auxiliam todo o *Scrum Team* com o controle do projeto. A definição do trabalho realizado dentro da *Sprint* é direcionada

por uma adaptação do problema debatido, de forma mutável, onde cada consideração provinda da equipe de desenvolvimento pode alterar diretamente o andamento da implementação do produto. (PRESSMAN, 2011).

Para a execução das *Sprints*, o *framework Scrum* consiste de equipes, onde são associados papéis, artefatos, eventos e regras. Existem três papéis, o *Product Owner*, o *Scrum Team* e o *Scrum Master*. Cada profissional possui as competências necessárias, para não necessitar do aguardo de ordens e realizar suas funções sem dependências do próprio time.

- *Product Owner*

  O *Product Owner*, considerado como a ligação entre o cliente e a empresa, tem a responsabilidade de possuir todo o conhecimento do projeto, fornecendo a visão do produto através dos requisitos documentados nas regras de negócio e no *product backlog*. Ele coordena diretamente quais necessidades dos clientes devem ser implementadas a cada entrega.

- *Scrum Team*

  O Time de Desenvolvimento é composto por profissionais multifuncionais que atuam diretamente na implementação dos incrementos do produto. Sua eficiência e eficácia é resultante da responsabilidade compartilhada e da flexibilidade de organização.

- *Scrum Master*

  O *Scrum Master* é um servo-líder responsável por garantir que todos os ritos e práticas do *framework*, aderidos à teoria, sejam devidamente executados pelo *Scrum Team*. Durante todo o processo de desenvolvimento, ele também possui a responsabilidade de retirar quaisquer impedimentos/obstáculos que atrasem ou prejudiquem o andamento do projeto, maximizando o valor no trabalho criado pelos membros do *Scrum Team*.

Ao iniciar uma *Sprint*, é preparada uma reunião de planejamento, no qual o *Product Owner* prioriza as funcionalidades esperadas para o produto, listadas por ele no Product Backlog. Com isso, o *Scrum Team* seleciona as funcionalidades que se comprometem a implementar durante a *Sprint*, alocando-as do *Product Backlog* para uma lista de tarefas, o *Sprint Backlog*.

Em cada dia da *Sprint* uma reunião diária é realizada. A reunião dura 15 minutos e é mantida no mesmo horário e local todos os dias, para que o *Scrum Team* possa alinhar sobre as

atividades e criar um plano para as próximas 24 horas. Chegando ao fim da *Sprint*, as funcionalidades implementadas são apresentadas em uma reunião de revisão, através da colaboração das partes interessadas com o *Scrum Team*. Além de ser realizada uma reunião de retrospectiva, reunião que fornece a oportunidade para o *Scrum Team* se inspecionar e criar um plano de melhorias a serem aplicadas na próxima *Sprint*, assim reiniciando o ciclo.

Além do *Product Backlog* e o *Sprint Backlog*, o *Scrum* possui o *Definition of Done*, que documenta as definições para as histórias de usuário, e o *Burndown*, um gráfico usado para monitorar o andamento do *Product Backlog*, quanto do *Sprint Baklog*. Estes artefatos são importantes e promovem a transparência dos itens que precisam ser elaborados para entrega de um produto.

3.3 Framework Kanban

Com origem japonesa, o *framework* surgiu como uma parte da Linha Toyota de Produção, futuramente implantado no desenvolvimento de software por David J. Anderson. O *Kanban* se caracteriza por ser um sistema que funciona por meio de um quadro com colunas e cartões. As colunas representam a evolução das funcionalidades produzidas para o produto, cujo status das tarefas são definidos de forma correspondente ao seu posicionamento nas colunas, também definindo qual momento do processo elas se encontram (BORTOLUCI et al., 2015). Já os cartões são as menores partes do *Kanban*. São as tarefas necessárias para que o resultado final seja alcançado. Ou seja, são as atividades que devem ser realizadas para implementar as funcionalidades do produto. Segundo Anderson (2010):

> "O *Kanban* rapidamente elimina as questões que prejudicam o desempenho, e desafia uma equipe para se concentrar em resolver essas questões a fim de manter um fluxo constante de trabalho".

O *Kanban* não é um sistema prescritivo, que impõe regras para que o trabalho seja feito corretamente. Ele proporciona uma colaboração entre todos os envolvidos, para que a equipe de desenvolvimento execute as tarefas de forma clara. Como auxílio, o quadro de atividades, considerando a indicação de limite de capacidade da equipe e do software em desenvolvimento, permite gerenciar a quantidade de esforço adicionado às atividades presentes no ciclo de desenvolvimento (ARRUDA, 2012). Com isso, permite visualizar o fluxo de trabalho, de forma transparente, sem preocupações com as iterações e estimativas, possibilitando acompanhar o ritmo de trabalho de toda a equipe, além de facilitar a agilidade das entregas.

Através dos cartões, também é possível indicar a produtividade individual de cada membro, já que podem possuir um responsável definido; assim como os riscos/obstáculos do projeto e os gargalos que ocasionam o atraso das entregas. Como complemento, no gerenciamento das atividades do projeto, a Agência Zetta UFLA utiliza marcadores: P1, P2, P3, P4. Esses marcadores auxiliam no nível de priorização ou definição do tipo de tarefa. Eles representam o nível de urgência que cada uma das atividades deve ser realizada.

- P1 - Prioridade Urgente, realizar o tão logo seja possível, na *release* atual e com menos de 2 dias
- P2 - Prioridade alta, realizar em até 5 dias
- P3 - Prioridade média, realizar em até 10 dias
- P4 - Prioridade baixa, realizar em até 30 dias

Um *kanban* geralmente possui três colunas, *To Do*, *Running* e *Done*. Mas assim como os marcadores, o quadro foi customizado pela equipe de desenvolvimento segundo o processo institucionalizado de desenvolvimento interno da Agência Zetta UFLA, com treze colunas, para auxiliar na padronização e gerência de todo o desenvolvimento das atividades. Sendo elas:

- *Open* – atividades que foram criadas para concluir o projeto. São inseridas pelo *Product Owner*.
- *Workflow: Planning* – Utilizada apenas quando o *Kanban* é agregado ao *framework Scrum*. Atividades analisadas e separadas pelo *Product Owner* durante a reunião de planejamento das funcionalidades do projeto. São inseridas pelo *Product Owner*.
- *Workflow: Blocked* – atividades que não podem ser executadas por falta de definição de regras de negócio ou dependências. São inseridas pelo *Product Owner*.
- *Workflow: QA analysis (Layout)* – atividades, a respeito de layout do sistema, de considerações que os analistas de qualidade julgam ser necessárias corrigir. São inseridas pelos analistas de qualidade.
- *Workflow: QA analysis* - atividades, a respeito de *bugs* e inconsistências que os analistas de qualidade julgam ser necessárias corrigir. São inseridas pelos analistas de qualidade.

- *Workflow: Ready for development* – atividades que estão com suas regras bem definidas e estão prontas para inicializar o desenvolvimento. São inseridas pelo *Product Owner*.
- *Workflow: In Dev* – atividades que estão sendo implementadas, no momento, por responsabilidade dos desenvolvedores. São inseridas pelos desenvolvedores.
- *Workflow: Dev Done* – atividades que o desenvolvimento foi realizado. São inseridas pelos desenvolvedores.
- *Workflow: Ready for testing* – atividades que estão prontas para que os analistas de qualidade possam testar. São inseridas pelos analistas de qualidade.
- *Workflow: In Test* – atividades que estão sendo testadas, no momento, pelos analistas de qualidade. São inseridas pelos analistas de qualidade.
- *Workflow: Test Done* – atividades já testadas pelos analistas de qualidade. São inseridas pelos analistas de qualidade.
- *Workflow: Ready for deploy* – atividades já finalizadas, que estão prontas para serem incluídas nos servidores de homologação e produção. São inseridas pelos desenvolvedores.
- *Closed* – atividades que passaram por todos os passos de desenvolvimento e que estejam inseridas no produto final. São inseridas pelos analistas de qualidade.

3.4 Miro

Miro (MIRO, 2021) é uma lousa digital *on-line*, disponível em qualquer dispositivo, que conta com diferentes formas de utilização para representar conceitos idealizados de forma visual e objetiva. A plataforma contribui com foco na construção e edição de mapas mentais, diagramas, sessões de *brainstorming*, cerimônias ágeis, entre outras coisas em tempo real e com possibilidade de colaboração em equipe, de maneira simples e eficaz. Independentemente da quantidade de membros presentes.

Além de possibilitar a criação de conceitos do zero, a plataforma também fornece diversos templates, como base para possíveis projetos. Dessa maneira, o início das edições realizadas no espaço de trabalho possui um direcionamento predefinido e ágil. Ela permite a adição de comentários, modificações dos itens apresentados em tela e monitoramento das atividades realizadas por todos os membros participantes.

Para questões de segurança a nível empresarial e controle administrativo avançado, o Miro inclui a opção para adicionar privacidade através de controles de conformidade. A ferramenta permite que apenas membros devidamente selecionados, possam participar do projeto. Impedindo que ideias, como por exemplo: de conceitos, prototipagem, entre outras documentações sejam visualizadas por usuários não autorizados.

3.5 Adobe XD

A criatividade é um elemento chave para criação de telas e *mockups* para projetos do gênero de aplicativos e sites responsivos, e o *Adobe Experience Design* veio para ajudar designers nessa tarefa (GAMEIRO, 2021). Pertencente à *Adobe Systems*, o Adobe XD (ADOBE, 2021), é uma ferramenta que integra design e prototipagem em um sistema *desktop*, rápido e fluido, que auxilia na organização do fluxo de trabalho para o desenvolvimento de aplicações.

A ferramenta possui duas funcionalidades, que através de uma interface simples e de usabilidade intuitiva, permite abordar os passos de concepção, visualização e compartilhamento do projeto de prototipação de telas com a equipe de desenvolvimento. A primeira, chamada protótipo, permite a edição de telas e elementos visuais que compõem o projeto. Já a segunda funcionalidade, chamada *preview*, complementa a primeira, simulando os testes dos protótipos em tempo real. Além de fornecer opções de compartilhamento, para que os *stakeholders* e outros envolvidos no projeto possam visualizar o protótipo em qualquer local e em qualquer dispositivo.

Dentre algumas das funcionalidades de edição visual presentes na ferramenta estão, acesso a modelos, sejam computadores, celulares, *Tablets* ou um tamanho personalizado; vetores básicos como triângulos, círculos, linhas e outros; caixas de texto para adição de palavras; precisão matemática, para modificações nas extremidades e no deslocamento dos elementos em tela; agrupamento dos elementos em pastas, com possibilidades de organização por parte do designer; dentre outras. Elementos estes, que auxiliam na revisão e ajustes, se necessários, de inconsistências encontradas no projeto, antes que cheguem ao estágio de desenvolvimento e programação.

3.6 GitLab

O *GitLab* (GITLAB, 2021), é uma plataforma *DevOps web*, para hospedagem de códigos, baseada em banco de dados e *open source*. Surgiu em meados de 2011, fundada por Dmitriy Zaporozhets e Valery Sizov, contando com a ajuda de vários colaboradores, que segundo seu site, conta com mais de 100 mil organizações e milhões de usuários utilizando a plataforma. *Gitlab* se apresenta como uma ferramenta de auxílio para todo o ciclo de *DevOps*, cujo termo é definido como a prática exercida por engenheiros de operações e desenvolvimento em todos os processos do ciclo de vida do serviço, desde o princípio do projeto até o suporte em produção (MUELLER, 2019). Unifica gerenciamento de tarefas, organização da documentação do projeto, revisão de código, CI - Integração Contínua e CD - Entrega Contínua em uma única interface de usuário.

O pipeline CI/CD, ou seja, integração contínua e entrega contínua, permite que, quando o código é enviado para o repositório, inicia-se um processo que o compila, faz o download dos pacotes necessários e executa os testes de unidade, de integração, entre outros. Com isso, é implementada uma abordagem, de engenharia de software, que permite que o projeto seja desenvolvido e implementado de forma rápida, confiável e repetitiva, necessitando de uma mínima participação do usuário.

Utilizando as ferramentas de CI e CD presentes na plataforma, é possível realizar a automatização de diversas tarefas no processo de *deploy*, agilizando as entregas de atividades pendentes e, apresentando muito mais segurança entre a integração das partes do projeto. Mostra-se uma abordagem muito útil na geração de *builds* e sobre a análise das falhas de compilação. Sendo esta, uma abordagem utilizada pela equipe durante o processo de desenvolvimento da aplicação apresentada neste relatório.

A organização de um repositório presente no *GitLab* funciona através dos grupos e projetos. Um grupo *GitLab*, se caracteriza por um conjunto de projetos. Cada grupo possui um nome como identificador, e sua composição à um repositório é opcional. Um projeto *GitLab* corresponde grosso modo a um repositório *Git*. Cada projeto possui, dentre todas suas funcionalidades, a *Wiki*, um sistema separado e integrado, que armazena e gerencia todo o conteúdo informativo sobre o projeto em desenvolvimento, para que todos os envolvidos compreendam as funcionalidades e regras de negócio. No caso da aplicação apresentada neste relatório, possui o grupo com nome “Observatório”, um projeto nomeado como “mapa-geo-interativo” e apresenta

toda documentação necessária. Sendo elas, documentação técnica, de negócios, sobre o *deploy*, de versões do sistema e documentos entregáveis.

Assim como o espaço reservado para a organização das documentações do projeto, o *GitLab* também fornece um gerenciador para as atividades realizadas durante o desenvolvimento da aplicação. Através das *issues*, é possível registrar as tarefas relacionadas ao desenvolvimento de novas funcionalidades, problemas e falhas da aplicação. O processo de produção se estrutura por meio de uma interface, o *board*, sendo que sua disposição facilita a movimentação das *issues* entre as colunas. Quando uma nova *issue* é inserida no *board*, ela é alocada mais à esquerda e, ao longo do processo, avança até chegar à coluna mais à direita, indicando que a tarefa foi concluída com sucesso. Para a organização das *issues*, a respeito de prioridades, finalidades, entre outros fatores, são utilizadas as *labels*, que ajudam na classificação das tarefas, facilitando que sejam encontradas entre todas envolvidas no projeto. Estas ferramentas contribuem muito durante o ciclo de desenvolvimento de uma aplicação.

3.7 DBeaver

O DBeaver – Software cliente para administração de banco de dados (DBEAVER, 2021), é um software *open-source*, distribuído pela licença do Apache, e multiplataforma para a gestão de bancos de dados, com suporte a vários tipos de extensões. Ele também é encontrado na forma corporativa e de código fechado, por meio da versão *Enterprise Edition* (EE), com suporte a bancos de dados (NoSQL / BigData), que é distribuído sob uma licença comercial. Para realizar as interações com banco de dados relacionais, ele utiliza a API do JDBC para suportar qualquer base de dados. Já para os bancos de dados NoSQL, faz o uso de *drivers* proprietários. A ferramenta também consegue tratar conexões que envolvem túneis SSH - *Secure Shell*, traduzido no sentido literal como “cápsula segura”, e possui a vantagem de armazenar os dados adquiridos através das conexões.

Desenvolvido por meio da linguagem Java e baseado na IDE Eclipse, com uma interface cuidadosamente projetada e implementada, trata-se de uma ferramenta bem versátil e completa. Ele fornece um editor com funcionalidades parecidas com as de uma planilha comum, no qual, ele permite manipular e editar os dados na própria tabela. Possui também a funcionalidade de ordenação dos dados através de suas colunas, filtrar elementos e exportar as informações em diferentes formatos. Outras funcionalidades, como gerar diagramas automaticamente, gerar

queries SQL de acordo com os dados selecionados da tabela e muitos outros recursos, facilitam o trabalho para usuários avançados em atividades relacionadas a bancos de dados.

3.8 Postman

O Postman é uma ferramenta que possibilita a realização de testes e depurações de serviços do modelo REST e de requisições em geral, a partir de uma interface gráfica. A ferramenta permite simular e salvar solicitações HTTP e HTTPS, para que possam ser utilizadas posteriormente. Adicionalmente, a ferramenta permite a desenvolvedores analisarem o funcionamento de serviços externos, com o intuito de gerenciar a sua utilização (TOMLINSON, 2017). Ela fornece meios para consumir, documentar e analisar o retorno das requisições feitas em serviços locais e na internet. Além disso, ela também concede a criação de servidores *mock*, que possuem a característica de simular requisições sem necessitar de uma API, retornando respostas específicas para as diferentes solicitações.

Com a funcionalidade que possibilita analisar as respostas das requisições, o período de desenvolvimento e o de realização dos testes, que validam o funcionamento da aplicação em diferentes situações, é reduzido drasticamente. Considerando um cenário sem a utilização da ferramenta, o serviço poderia ser testado somente após a implementação de uma aplicação composta por interface. Esta situação demandaria um esforço adicional e possivelmente, causaria maior gasto de recursos. O tempo gasto para que todo esse processo seja finalizado poderia ser melhor aproveitado durante a execução do ciclo de desenvolvimento do produto final. A ferramenta está disponível como uma extensão do *Chrome* e como um aplicativo de *desktop* para sistemas operacionais *MAC, Windows e Linux* (POSTMAN, 2021).

3.9 GeoServer

O Geoserver é um software livre mantido pelo *Open Planning Project*, que permite o desenvolvimento de soluções *web mapping* e integra diferentes repositórios de dados geográficos, mantendo a simplicidade e alta performance em seus serviços (OLIVEIRA, 2008). Além disso, integra diversos repositórios de dados geográficos com simplicidade e alta performance, através de padrões abertos que seguem especificações e implementam protocolos definidos pela *Open Geospatial Consortium* (OGC). Também permite publicar dados através de representações visuais por meio do WMS (*Web Map Service*), dados reais com o WCS (*Web Coverage Service*)

e editar (atualizar, deletar ou inserir) novos elementos por meio do serviço WFS (*Web Feature Service*), em sua versão transacional. Esta interoperabilidade torna as informações geográficas mais acessíveis, pois permite que a aplicação se comunique, sem impedimento algum, com outros sistemas. Por ser um projeto voltado para a comunidade, o GeoServer é desenvolvido, testado e apoiado por diferentes colaboradores e organizações de todo o mundo (GEOSERVER, 2021).

Os cadastros de dados no GeoServer são realizados por meio da integração dos principais formatos vetoriais e *raster*. Já a forma para disponibilizá-los, abrange diversas maneiras, adequando-se à necessidade de quem utiliza a ferramenta, sendo que possui suporte ao KML, que possibilita a visualização dos dados diretamente no *Google Earth e Google Maps; OpenLayers*, que permite solicitações para visualizar o mapa no próprio navegador; *SVG*, uma linguagem *XML* que descreve desenhos gráficos bidimensionais de forma vetorial; além de outros como *PDF, Excel, CSV, GeoRSS, GeoJSON e GML.*

3.10 QGIS

QGIS (QGIS, 2021) é uma abreviação do antigo nome *Quantum GIS*, passando a ser chamado assim oficialmente desde setembro de 2013, é um projeto oficial da *Open Source Geospatial Foundation* (OSGeo), multiplataforma e licenciado pela Licença Pública Geral GNU, que funciona como um Sistema de Informação Geográfica (SIG) com suporte à diversos formatos. Dentre as funcionalidades, o QGIS permite realizar explorações interativas de dados, por meio da visualização e seleção de atributos; consultas espaciais, criação de simbologias *raster* e vetoriais (ALMEIDA, 2011). Ele também fornece um conjunto amplo de funcionalidades de Geoprocessamento, para a manipulação dos dados, no qual permite visualizar, gerir, editar e analisar as informações, além de possibilitar a composição de mapas imprimíveis.

A ferramenta possui fatores que contribuem para que suas funcionalidades continuem em constante crescimento. Um deles é a capacidade de integrar diversos outros programas, permitindo adaptar e ampliar seus recursos através da instalação de extensões. Algumas das extensões QGIS como *GDAL, GRASS GIS, PostGIS* e *PostgreSQL* são consideradas importantes, visto que aumentaram significativamente sua usabilidade (PEJOVIĆ et al., 2014). Com esses complementos, projetos que envolvem diversas áreas do conhecimento como, meio ambiente, arquitetura/urbanismo, engenharia, agricultura, e outros que demandam análises espaciais com-

plexas, são planejados e executados de forma mais otimizada. Dentre outros pacotes, o QGIS pode integrar *plugins* como, o Sistema Estatístico R, *GRASS*, *PostgreSQL*/*PostGIS* e *Saga GIS*.

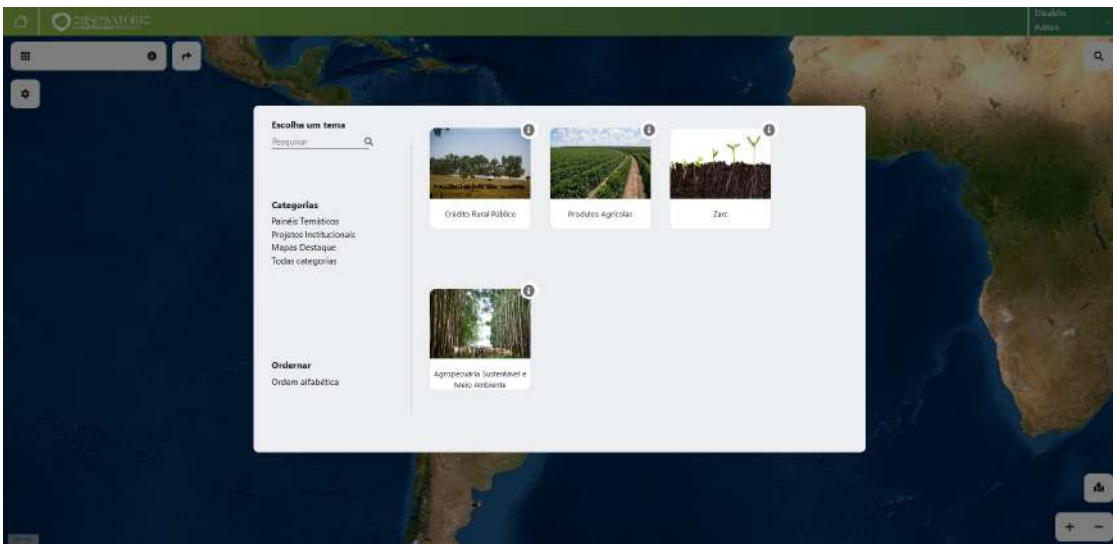
O QGIS por ser de domínio público sob licença livre possibilita aos usuários, por meio da comunidade Quantum GIS, utilizar, autenticar, produzir material didático, codificar e corrigir o software, gerar novas ferramentas ou mesmo melhorar os já existentes (SILVA, 2010). Com isso, tornou-se uma das principais ferramentas para usuários que atuam na área de Geotecnologia.

4 O PROJETO

A Plataforma Geoespacial, na qual o estagiário atuou no projeto, é um ambiente do Observatório da Agropecuária Brasileira dedicado à disposição e organização de conteúdos, dados e informações, através dos quais podem ser visualizados e construídos de acordo com a necessidade e interpretação dos usuários, maximizando sua experiência. Ilustrada na Figura 4.1, a Plataforma Geoespacial vai além de um ambiente SIG, disponibilizando camadas e ferramentas inteligentes, que auxiliam na realização de pesquisas. Trata-se de um sistema que visa a integração de bases de dados georreferenciados, abertos ao público, do setor agropecuário brasileiro, por meio de métodos de modelagem e de ferramentas de ciência de dados, gerando temas dinâmicos onde é possível ter a visualização em conjunto com outras bases de informação dispostas por diversas fontes oficiais.

O sistema também possui ferramentas de interatividade acopladas, imagens de satélite de resoluções variáveis, dados não espaciais que complementam o entendimento das cadeias produtivas do setor, geradas por modelos espaciais complexos. Além disso, o sistema possibilita a simulação de cenários considerando variáveis do meio físico, sendo elas socioeconômicas ou ambientais em temas relevantes para o setor agropecuário.

Figura 4.1 – Plataforma GeoEspacial



Fonte: Agência Zetta UFLA (2021)

4.1 Plataforma Geoespacial do ZARC

A Plataforma Geoespacial do ZARC, ilustrada na Figura 4.2, tem como propósito apresentar as informações da EMBRAPA/MAPA consolidadas de todas as portarias do zoneamento para cada cultura, grupo, período do ano e tipo de solo. Esses dados estão disponíveis no [1] portal de indicadores da agricultura, gerenciado pela Coordenação de Risco Climático – Departamento de Gestão de Riscos da SPA/MAPA. Além disso, a plataforma oferece ferramentas integradas, com o intuito de promover maior usabilidade e interatividade na interpretação da informação, apresentando a tábua de risco, a relação de cultivares e também mapas de visualização dos dados por município, estado e região.

Figura 4.2 – Plataforma GeoEspacial ZARC (Soja)

Fonte: Agência Zetta UFLA (2021)

4.2 Plataforma Geoespacial de Produtos Agrícolas

A Plataforma Geoespacial de Produtos Agrícolas, ilustrada na Figura 4.3, traz dados do cultivo brasileiro, apresentando importantes informações de produção e mercado, com representações a nível estadual e municipal, além de um amplo período de dados (1974 até 2020). As informações estão dispostas por meio de mapas temáticos e relatórios, podendo o usuário agregar nas pesquisas com dados numéricos e gráficos disponíveis para interação. As informações consolidadas são oriundas da Companhia Nacional de Abastecimento (Conab) e do Insti-

[1] O portal oferece ferramentas integradas, com o intuito de promover maior usabilidade e intuitividade na interpretação da informação, centralizando a tábua de risco, a relação de cultivares e também mapas de visualização dos dados por município, estado e região.

tuto Brasileiro de Geografia e Estatística (IBGE). Estão dispostos dados das seguintes culturas agrícolas: arroz, cacau, café, feijão, milho, soja e trigo.

Figura 4.3 – Plataforma GeoEspacial Produtos Agrícolas (Área Plantada - Café)

Fonte: Agência Zetta UFLA (2021)

4.3 Plataforma Geoespacial do Crédito Rural Público

A Plataforma Geoespacial do Crédito Rural Público, ilustrada na Figura 4.4, disponibiliza dados de contratos do Programa ABC originadas do Banco Central do Brasil - Bacen, de maneira consolidada e interativa, avaliando a quantidade, valor e área de contratos. Os filtros aplicados na plataforma oferecem a visualização por período dos dados, estratificação a nível estadual e municipal e por tecnologias: Recuperação de Pastagem Degradada (RPD), Integração, Lavoura, Pecuária, Floresta / Sistemas Agroflorestais (ILPF / SAF), Sistema de Plantio Direto (SPD), Fixação Biológica de Nitrogênio (FBN), Floresta Plantada (FP), Tratamento de Dejetos Animais (TDA). As ferramentas também permitem ao usuário uma análise do desempenho do Crédito Rural Público, assim como uma visão detalhada e personalizada dos dados e das informações estruturadas.

4.4 Plataforma Geoespacial da Agropecuária Sustentável e Meio Ambiente

A Plataforma Geoespacial da Agropecuária Sustentável e Meio Ambiente, ilustrada na Figura 4.5 e na Figura 4.6, disponibiliza uma variedade de dados da agropecuária sustentável, regularização ambiental, regularização fundiária e uso do solo e monitoramento, apresentando

Figura 4.4 – Plataforma GeoEspacial Crédito Rural Público (Recuperação Pastagens)

Fonte: Agência Zetta UFLA (2021)

informações por meio de mapas temáticos, pontuais e áreas localizadas a nível nacional, estadual e municipal. Com as ferramentas é possível obter informações consolidadas, as quais são extraídas por meio de cruzamentos espaciais e tabulares entre as bases de dado, apresentadas tanto espacialmente quanto no formato de relatórios com números e gráficos. Os dados são disponibilizados por meio das fontes: Agência Nacional de Águas e Saneamento Básico - ANA, Banco Central do Brasil - Bacen, Instituto Brasileiro do Meio Ambiente e Recursos Naturais - Ibama, Serviço Florestal Brasileiro - SFB, Instituto Nacional de Colonização e Reforma Agrária - INCRA, Laboratório de Processamento e Imagens de Geoprocessamento - LAPIG, Instituto Nacional de Pesquisas Espaciais - INPE, Empresa Brasileira de Pesquisa Agropecuária - Embrapa e Fundação Nacional do Índio - Funai.

Figura 4.5 – Plataforma GeoEspacial Agropecuária Sustentável e Meio Ambiente (Irrigação)

Fonte: Agência Zetta UFLA (2021)

Figura 4.6 – Plataforma GeoEspacial Agropecuária Sustentável e Meio Ambiente (Imóveis Rurais (CAR) x UC‘s)

Fonte: Agência Zetta UFLA (2021)

5 ATIVIDADES REALIZADAS NO ESTÁGIO

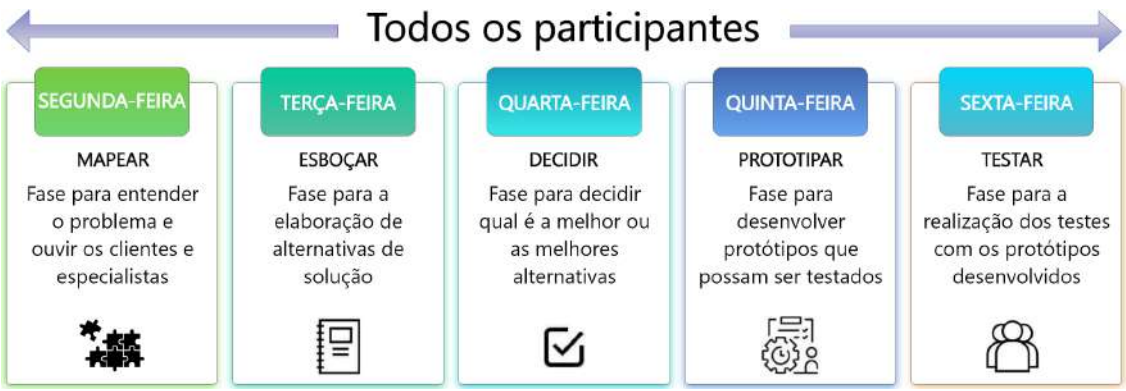
5.1 Participação no *DesignSprint*

No início de desenvolvimento do projeto, o período de estágio realizado pelo discente o proporcionou oportunidades de exercer funções como designer e facilitador de um *Design Sprint*. Este *Design Sprint* consistiu na execução de um processo estruturado com elementos do *design thinking* e também de metodologias ágeis. Segundo Pinheiro e Alt (2011), o *design thinking* não é uma técnica específica, e sim um modelo mental pautado na empatia, colaboração e experimentação. Também pontuam que, a partir do momento que uma empresa adota tal abordagem, ela se aproxima mais de seus consumidores, tornando-a mais suscetível às mudanças de mercado.

Com um tempo limitado de 5 dias e uma equipe dedicada para esclarecer todo o detalhamento de negócios, envolveu tanto a elaboração de ideias, de forma colaborativa com a prototipação, quanto a medição de sua eficácia, por meio de testes com os clientes, que, através dos *feedbacks*, apresentaram novas considerações.

Além do discente, a equipe envolvida foi composta por *Stakeholders*, clientes donos do projeto; um *Product Manager*, que possui conhecimento sobre os usuários; a *Product Owner* (PO) da equipe de desenvolvimento, responsável por maximizar o valor do produto; um desenvolvedor, voltado à assuntos técnicos e um *Sprint Master*, que teve o papel de administrar o progresso e direcionar a operação das atividades das sessões. A execução dessas atividades foi realizada com o auxílio da ferramenta Miro e dividida em diferentes fases, dependentes umas das outras e com duração de no máximo 8 horas diárias. De acordo com a Figura 5.1, são elas: mapear, esboçar, decidir, prototipar e testar.

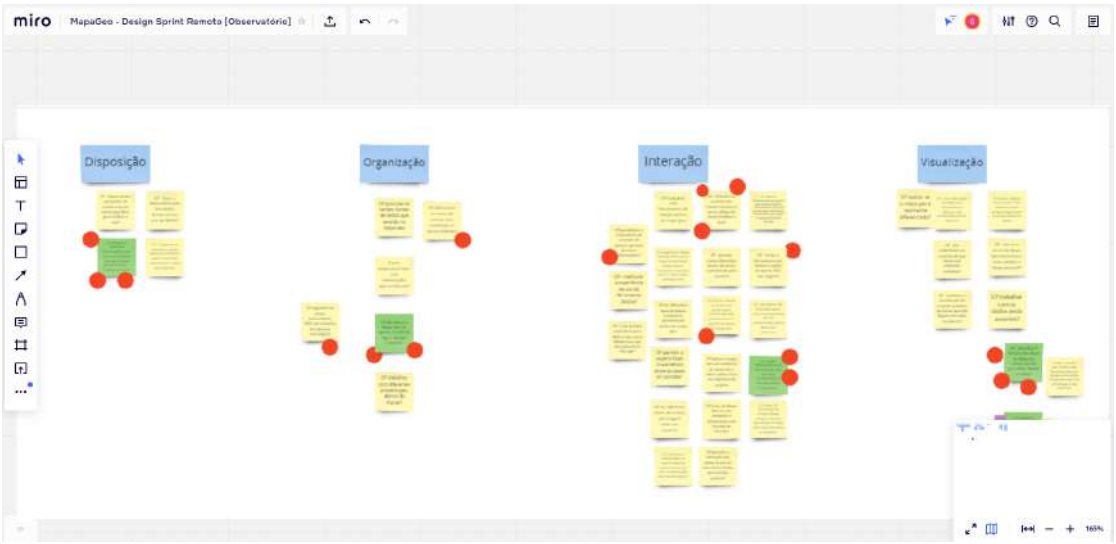
Figura 5.1 – Modelo do design sprint



Fonte: Adaptado de ProValore (2021)

A fase de mapeamento, no primeiro dia, baseou-se no entendimento das necessidades abordadas pelas partes interessadas, exteriorizando todos os conceitos e questões que envolviam a ideia do produto. Informações estas, que inicialmente se encontram dispersas e nebulosas por todos os participantes. Para esta etapa, é importante definir um ponto de partida e um posicionamento sobre o objetivo do projeto. Dessa forma, foram realizadas atividades que abordavam partes do conhecimento de todos os envolvidos, através da desconstrução do produto, com um *Storyboard*; definição das métricas abordadas, além de expressar a voz do consumidor, descobrindo quais seriam as possíveis funcionalidades realmente utilizadas por um suposto usuário, como ilustrado na Figura 5.2 e na Figura 5.3. Esta é a base do *Sprint Master*, para que ele consiga guiar todo o cronograma dos próximos dias do *Design Sprint*.

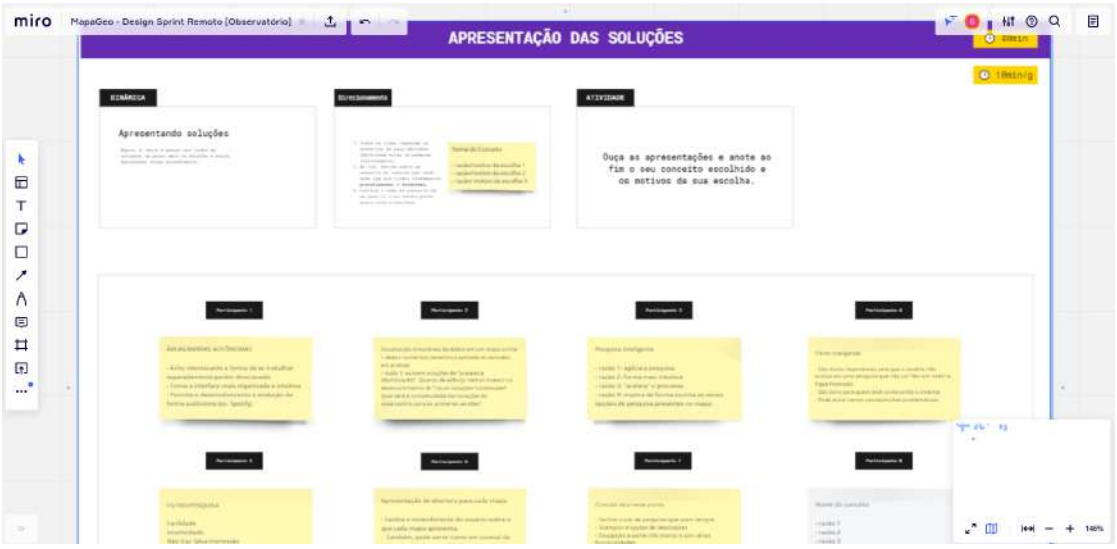
Figura 5.2 – Levantamento de considerações



Fonte: Arquivo da Agência Zetta UFLA (2021)

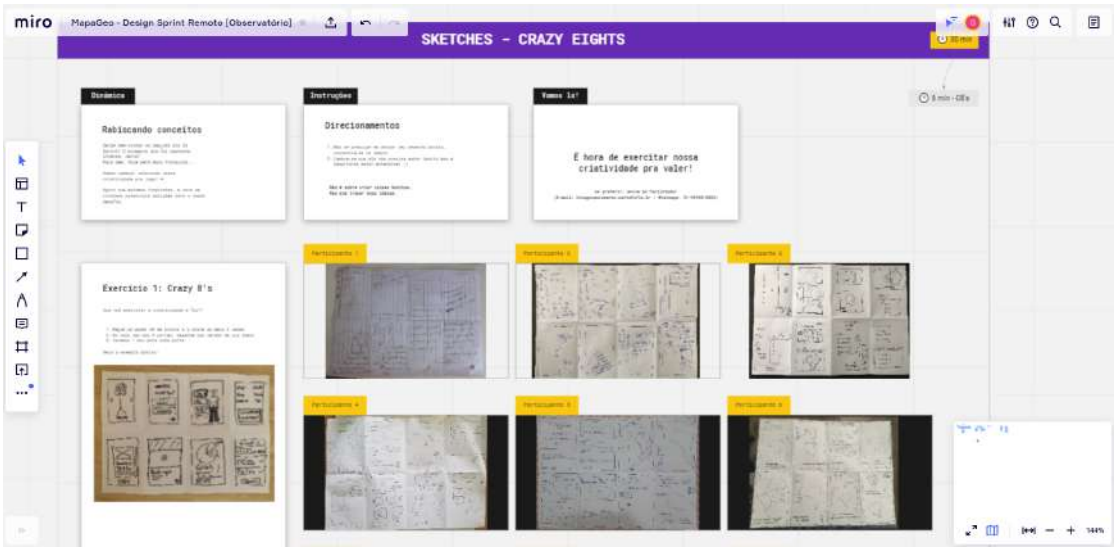
O segundo dia, na fase de esboço, iniciou por meio do *Agile Stories*, com o intuito de fixar ainda mais os conceitos e elaborar fluxos variados para a utilização do sistema, considerando diferentes personas e cenários. Com isso, as experiências de usuário foram coletadas, organizadas e analisadas como oportunidades para futuras funcionalidades. Após definir os objetivos e direcionamentos, foi chegado o momento de ideação das diferentes soluções possíveis, que, com o auxílio da técnica *Crazy* 8s, mostrada na Figura 5.4, resultou em uma ampla variedade de esboços. Em meio às soluções, toda a equipe foi subdividida em pequenas formações. Cada uma possuía no máximo três *Stakeholders* e um facilitador, com o propósito de iniciar o desenho de diferentes *sketches* de forma ágil e democrática, funcionando verdadeiramente como um *brainstorming* sem critérios de julgamento.

Figura 5.3 – Apresentação das Soluções



Fonte: Arquivo da Agência Zetta UFLA (2021)

Figura 5.4 – Crazy Eights

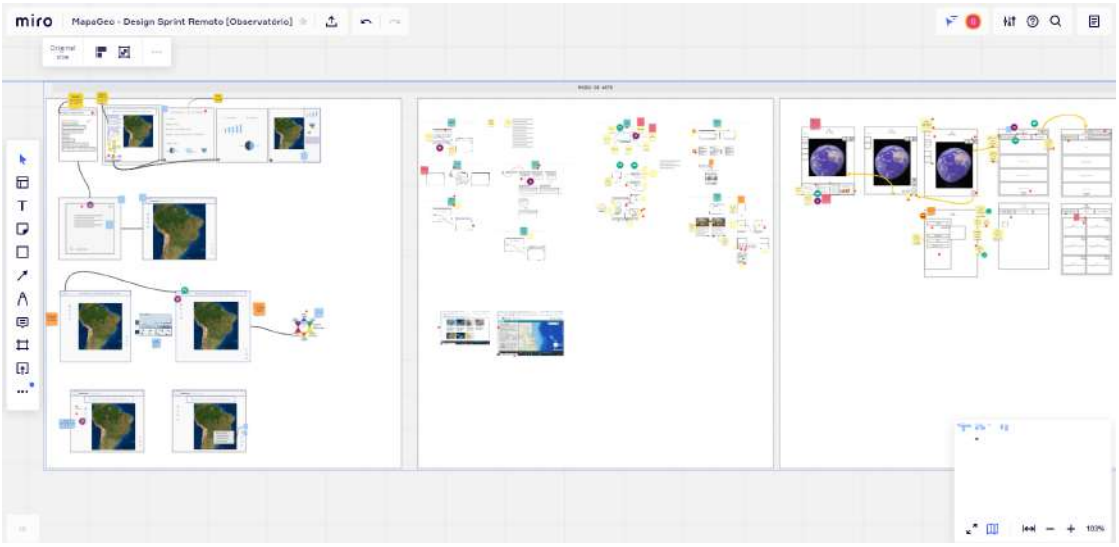


Fonte: Arquivo da Agência Zetta UFLA (2021)

No terceiro dia, a fase de decisão, teve como objetivo a escolha e a priorização das melhores ideias construídas na fase de esboço, caracterizando-se tanto como um design de interface quanto à uma definição de funcionalidade, ilustrada na Figura 5.5. Para isso, os *sketches* foram alocados em um único espaço, para que os participantes pudessem julgá-los através da votação por pontos, selecionando as ideias que seriam realmente consideradas na etapa de prototipação. As pontuações foram distribuídas em diferentes quantidades para cada um, levando em consideração o destaque dos *Stakeholders* como os principais decisores do projeto, possuindo assim a maior quantidade. As pontuações foram definidas em diferentes tipos e pesos:

- Vermelhas: Votos livres e comuns para todos os participantes, onde as funcionalidades ou interfaces que recebessem a maior quantidade, seriam debatidas para participar da fase de prototipação.
- Com as iniciais do nome: Comuns para todos os participantes, com a quantidade limitada em duas escolhas, para cada, de funcionalidades ou interfaces que seriam debatidas para participar da fase de prototipação.
- Estrelas: Limitadas apenas aos *Stakeholders*, com a quantidade limitada em quatro escolhas de funcionalidades ou interfaces que iriam participar da fase de prototipação.

Figura 5.5 – Modelo do design sprint



Fonte: Arquivo da Agência Zetta UFLA (2021)

Já no quarto dia, foi realizada a fase de prototipação, que consistiu na criação de um protótipo rápido, de média fidelidade, para representar a solução escolhida na etapa anterior. Esse procedimento precisou ser executado da maneira mais simples possível, para garantir uma alta produtividade, levando em consideração o tempo fornecido para finalizá-lo. Dessa maneira, foi necessária a escolha de uma ferramenta de prototipagem mais familiar à equipe do *Design Sprint*. O Adobe XD permitiu realizar a etapa de forma simples e facilitada, o que também garantiu a eficácia na identificação de oportunidades de melhorias, antecipação de erros, além de reduzir possíveis riscos sobre os investimentos que não agregassem o projeto.

Por fim, a fase de teste permitiu a realização de entrevistas para que os potenciais usuários e *Stakeholders* pudessem avaliar o protótipo criado, através de sua utilização em sessões

individuais. Com o protótipo sendo apresentado e monitorado em tempo real, foi possível controlar os *feedbacks*, no qual foram documentados e corrigidos posteriormente.

Devido a algumas ocorrências de não conformidade, o primeiro dia da segunda semana serviu para tratar os problemas encontrados. Ao fim do mesmo dia, ocorreram as novas entrevistas para que as alterações fossem novamente avaliadas, finalizando a etapa. O protótipo resultante de todo o processo do *Design Sprint*, serviu de base para que o *Scrum Team* iniciasse o ciclo de desenvolvimento da Plataforma Geoespacial.

5.2 Participação na elaboração do protótipo do sistema

O processo de prototipação foi realizado através da ferramenta Adobe XD e com participação do discente, no qual ele auxiliou na transferência das ideias do âmbito conceitual, definidas nas primeiras fases do Design *Sprint*, para um objeto virtual e tangível, como mostrado na Figura 5.6. Segundo Back et al. (2008), o protótipo se caracteriza como a representação do produto a ser projetado, incluindo suas características funcionais e dimensionais.

A atividade de prototipação consistiu em produzir uma versão inicial, reduzida e de média fidelidade do sistema, atentando-se a problemas de design, usabilidade e adequação. Esses conceitos foram vistos durante a disciplina Interação Humano-Computador, que se mostraram de grande valia para que o resultado final gerasse satisfação, na experiência do usuário, durante a utilização do produto.

Figura 5.6 – Montagem do protótipo

Fonte: Arquivo da Agência Zetta UFLA (2021)

Considerando o processo de definição de ideias para o projeto, a prototipação é apresentada como uma das últimas fases. No entanto, foi necessário executá-la paralelamente com as outras atividades presentes no ciclo de desenvolvimento. Mesmo já possuindo diversos requisitos definidos, novos alinhamentos e conceitos, levantados pelos clientes, surgiram e foram julgados como indispensáveis. Dessa forma, a partir do protótipo inicial, algumas funcionalidades tiveram que ser reajustadas para adequar às novas demandas apontadas pela *Product Owner* da equipe. Essa atividade possuiu a duração necessária para que as expectativas alcançassem os objetivos, garantindo facilidade na comunicação entre a equipe, fornecedores e clientes, melhorando o entendimento sobre os componentes e funcionamento do produto (ROMEIRO et al., 2010).

Além da prototipação evolucionária ou incremental, na qual o protótipo é adaptado durante o avanço do desenvolvimento, também foi utilizada a prototipação descartável. Essa atividade, assim como a evolucionária, também é realizada em diversas etapas do desenvolvimento, mas os diferentes detalhes de cada momento do projeto são prototipados separadamente e depois descartados. Esta abordagem maximizou a eficiência da equipe.

5.3 Participação no Workshop de demonstração

Antes mesmo de iniciar o processo de desenvolvimento da aplicação, foi necessário realizar um *workshop*, conduzido pela *Product Owner* (PO) da equipe com auxílio do estagiário, cujo objetivo foi apresentar/alinhar todas as regras, objetivos, funcionalidades e informações importantes sobre o produto final. Por meio da utilização de uma série de documentações amplamente visuais, foi possível esclarecer todo o conteúdo para os membros da equipe, de forma simples e facilitada, envolvendo as representações das ideias coletadas durante o Design *Sprint*, como protótipos desenvolvidos, *product backlog* com as histórias de usuário, entre outras documentações.

Como a equipe era composta por alguns membros menos experientes, os conceitos do projeto precisaram ser explicados por meio da navegação do protótipo, demonstrando os possíveis fluxos que a aplicação permitia executar, assim como suas funcionalidades. O protótipo contava com navegação do mapa, geração de relatórios, visualização de camadas, entre outras ferramentas. Essa forma de execução ofereceu bons resultados, pois proporcionou uma experiência conjunta e dinâmica, no qual permitiu uma participação mais direta dos colaboradores, por meio de debates e troca de conhecimentos.

5.4 Participação na elaboração do documento de regras do sistema

Por se tratar de um sistema que integra muitas informações geoespaciais, diferentes funcionalidades precisaram ser implementadas para exibir os dados da melhor forma possível. Dessa maneira, por meio de uma série de validações e alinhamentos, foram consolidadas as regras, que definem o modo de utilização das informações disponibilizadas, assim como o funcionamento da aplicação.

Em meio a isso, durante o período de estágio realizado pelo discente, foi possível atuar diretamente na criação e atualização do documento de regras de negócio do sistema. Isto permitiu ao discente observar, na prática, conceitos abordados na disciplina de Engenharia de Software, no qual aborda que um software consiste na união do código desenvolvido e sua documentação. A atividade acabou se caracterizando como um complemento às responsabilidades da área de atuação do estagiário.

O documento de regras tratou de reunir todas as instruções e particularidades que o sistema necessitou contemplar, envolvendo restrições, condições e exceções simples ou mais complexas. Informações estas, que possuem o papel de orientar o comportamento e definir o que, onde, quando, como e o porquê; algo deve ser implementado na aplicação. Dessa forma, algumas regras não foram necessariamente designadas para tratar uma funcionalidade propriamente dita, mas sim, para definir o modo como determinadas operações deveriam ser realizadas, envolvendo uma ou mais funcionalidades.

Para que o produto alcançasse às expectativas esperadas pelos clientes, foi necessário descrever as regras de forma mais objetiva, abordando a sequência lógica de sua execução, assim como os possíveis fluxos que as envolviam. Assim, foi possível assegurar o entendimento, por parte da equipe de desenvolvimento, de todo o funcionamento da aplicação. Para garantir ainda mais resultados positivos durante o processo de desenvolvimento, as regras foram submetidas a revisões constantes, com o objetivo de identificar os gargalos, duplicidades e incidências. Em certas circunstâncias, regras inicialmente identificadas como fundamentais, precisaram ser reescritas ou até mesmo descartadas. Este é o caso da funcionalidade de download das camadas, que precisou ser retirada da aplicação por possuir um desempenho abaixo do esperado.

5.5 Participação na organização dos testes

Com os requisitos do sistema definidos nas etapas iniciais do ciclo de vida do software, o discente, por meio do estágio, teve a oportunidade de planejar os procedimentos de teste necessários para validá-lo. Os procedimentos abrangem diversas atividades, envolvendo desde o aprimoramento da qualidade, até a verificação e validação apropriada das expectativas buscadas. Os métodos aplicados influenciam diretamente na garantia da capacidade da equipe de desenvolvimento, buscando assegurar a satisfação, por parte dos clientes, sobre os resultados encontrados no produto final.

Levando em consideração os tipos de exigências feitas pelas partes interessadas, foi possível realizar as validações de diferentes maneiras. Os requisitos funcionais foram validados por meio de descrições que simulam o uso do sistema, e os requisitos voltados às normas de utilização, através de descrições de teste mais aprofundados.

A primeira atividade realizada foi a construção dos casos de uso, um diagrama que representa a sequência de possíveis caminhos que o usuário pode percorrer dentro do sistema, cujo conceito foi abordado na disciplina Engenharia de Software, e foi responsável por dar início à organização dos testes. Para os autores Lima (2011) e Booch, Rumbaugh e Jacobson (2005), o diagrama de casos de uso possui um papel central durante a modelagem do comportamento de um sistema, onde busca representar toda sua funcionalidade, evidenciando interações externas com outras entidades. De acordo com Melo (2010):

> "Um caso de uso descreve uma sequência de ações que representam um cenário principal (perfeito) e cenários alternativos, com o objetivo de demonstrar o comportamento de um sistema (ou parte dele), através de interações com autores".

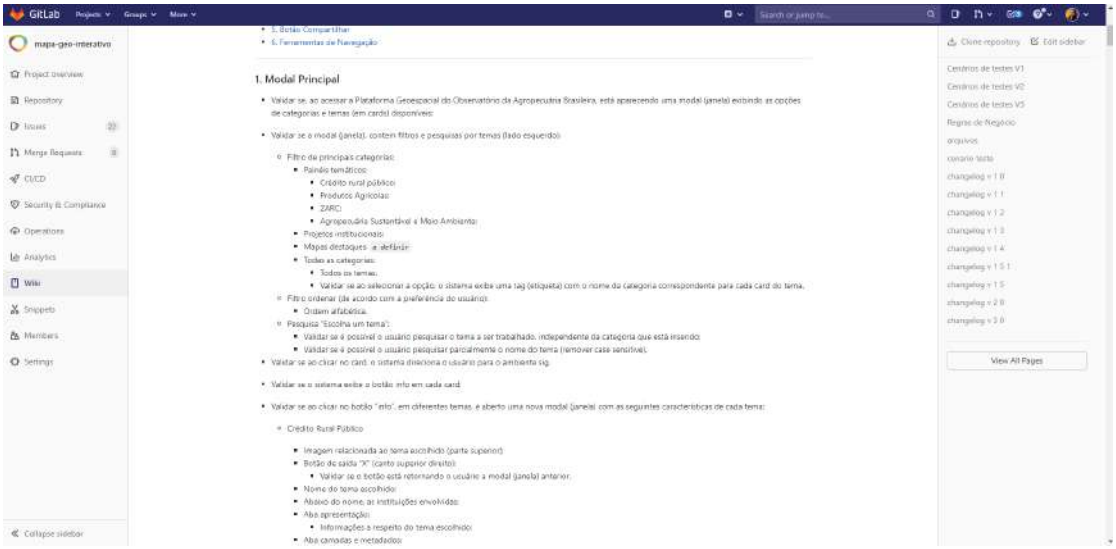
Por se tratar de derivações dos casos de uso, os casos de teste se caracterizaram como a primeira documentação utilizada para certificar a conformidade dos requisitos. Neles são relatados os fluxos específicos a serem validados. Para isso, foi preciso realizar uma análise sobre cada requisito definido no projeto, envolvendo a maior parte dos caminhos presentes na aplicação e percorrendo todos os fluxos básicos, assim como os alternativos.

Adequando-se a complementar as descrições dos casos de uso, os *scripts* de teste foram utilizados para propor um maior detalhamento nos passos necessários para a realização dos testes. Este processo foi realizado informando as regras envolvidas nas funcionalidades, os botões a serem pressionados, além dos resultados adquiridos em cada etapa. No entanto, devido

a constantes refatorações das regras e adição de novas funcionalidades, o estagiário percebeu que o tempo gasto para a atualização das descrições dos *scripts*, poderia ser melhor investido na execução dos testes. Dessa forma, pensando em situações mais abrangentes, onde a equipe de teste também precisaria se conciliar ainda mais com as definições e regras do projeto, foi necessário buscar outras formas de documentação dos testes.

Como solução, os cenários de testes, Figura 5.7, proporcionaram descrições menos detalhadas e mais diretas, com a proposta de informar o objetivo a ser encontrado, e não descrever como realizar as etapas para alcançar o resultado desejado. Os cenários de teste obrigam a escolha de abordagens mais lógicas. Dessa maneira, incentivou o estagiário a ser mais criativo e habilidoso para encontrar as possíveis inconsistências na aplicação.

Figura 5.7 – Cenários de teste



Fonte: Arquivo da Agência Zetta UFLA (2021)

5.6 Participação na realização dos testes

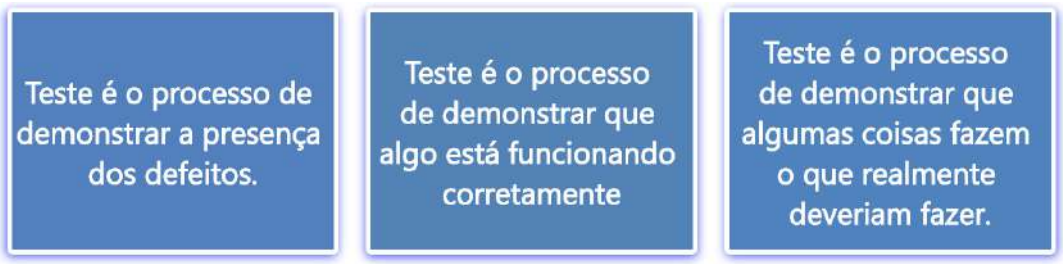
Mesmo planejando toda a construção do software utilizando diversas ferramentas e métodos, seus resultados, ao longo do processo, foram suscetíveis a falhas. Para que então fosse possível garantir o controle sobre sua qualidade, correspondendo às demandas durante o desenvolvimento, de forma geral, foram realizados os testes de software, desde o momento de criação até sua implantação. O objetivo dos testes consiste em verificar o funcionamento do sistema e encontrar, previamente, os possíveis defeitos que ele pode oferecer, antes mesmo que sejam detectados no ambiente de produção. Também é preciso verificar se os mesmos foram correta-

mente corrigidos e implementados, sempre buscando atender aos requisitos especificados pelas partes interessadas.

Dessa forma, durante o estágio, o discente precisou participar ativamente na certificação da qualidade do sistema, realizando os testes durante todas as fases do processo de desenvolvimento. No qual, buscou alcançar a melhoria da qualidade por meio da aplicação de heurísticas, métricas e métodos.

Segundo Pressman (2011), "teste é um elemento crítico para a garantia da qualidade de sistemas". Os testes de software podem possuir diferentes definições, dependendo da interpretação dos membros da equipe de desenvolvimento, conforme é mostrado na Figura 5.8.

Figura 5.8 – Simplificação das definições de teste



Fonte: Adaptado de Bartié (2002)

Sendo assim, o teste de software se apresentou muito além de um processo de checagem, onde apenas seria preciso executar a aplicação e verificar os resultados obtidos. Envolveu tanto a etapa de modelagem de técnicas para auxiliar durante a validação das condições de uso, quanto a simulação real do comportamento do software, conjunto ao desenvolvimento de documentações para relatar as falhas encontradas. Isso permitiu otimizar a gestão dos recursos da Agência Zetta UFLA sobre o projeto, através da descoberta dos defeitos durante o ciclo de desenvolvimento do sistema. Considerando que o quanto antes são detectados, menor é o impacto de implementação e de custo para corrigi-los.

Levando as metodologias ágeis em consideração, que priorizam as entregas de acordo com as necessidades dos clientes, o *framework Scrum* e o *Kanban* influenciaram diretamente na implantação dos testes como uma tarefa de gestão contínua ao longo do processo, evitando ser comparado a uma mera etapa durante o desenvolvimento. Para Pressman (2011), as metodologias ágeis se desenvolveram com o objetivo de sanar as fraquezas perceptíveis na Engenharia de Software convencional. Para se adequar a esta abordagem, o processo de execução dos testes foi dividido em planejamento, preparação, especificação, execução dos testes e relato do progresso/resultados, podendo alternar entre as etapas, dependendo da necessidade.

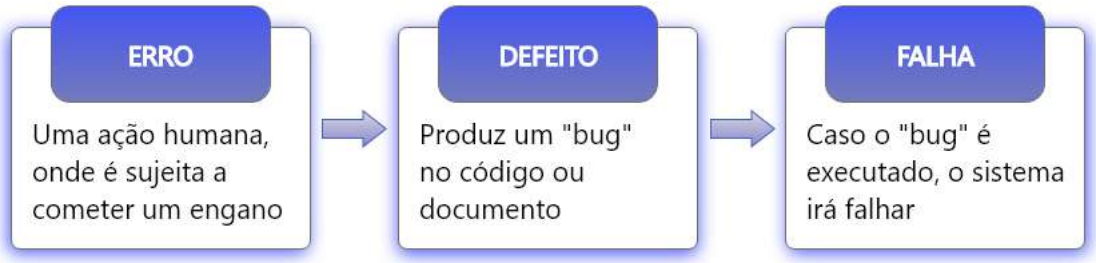
- Planejamento: As técnicas e métodos de teste são definidos para compor um plano de teste.
- Preparação: Etapa reservada para assegurar que o ambiente de teste está bem estruturado e apresenta todas as modificações realizadas pela equipe de desenvolvimento, para que possam ser realizados os testes da maneira planejada.
- Especificação: Fase onde os roteiros de teste são revisados, assegurando que estão em conformidade com o que é apresentado no ambiente de teste.
- Execução dos testes: Momento em que os roteiros de teste são colocados a prova, executando-os em forma de simulações dos possíveis cenários que um usuário final pode realizar no sistema, podendo considerar a atividade, responsável pela implementação de uma funcionalidade ou regra do sistema, como finalizada ou não finalizada.
- Relato do progresso/resultados: Se baseia na documentação de todos os resultados obtidos no fim do processo.

Dessa forma, o estagiário, participando como um dos analistas de qualidade, habituou-se em estar sempre atualizado sobre as regras do projeto e a executar os diferentes tipos de testes necessários, através de um perfil pessimista, para que pudesse validar as diferentes combinações do sistema e relatar as inconsistências encontradas. As inconsistências, foram registradas de forma detalhada através da listagem dos *bugs*, assim como sugestões para melhoria do sistema, exercendo a base para que os outros membros do *Scrum Team* se familiarizassem com as considerações e as corrigissem posteriormente.

Para isso, primeiro, o estagiário precisou entender a diferença entre os tipos de inconformidade, sendo eles, erro, defeito e falha, mostrados na Figura 5.9. O defeito se caracteriza como a manifestação do erro, também conhecido como *bug*, uma ação humana que resulta em algo incorreto; além de também poder ocasionar a falha. Nem todos os testes resultam em falhas, podem gerar falsos positivos, que não são defeitos; ou também falsos negativos, como a não detecção de defeitos que deveriam ser realmente encontrados.

Portando todos esses conhecimentos, o estagiário precisou elaborar o máximo de cenários possíveis sobre o sistema. Sendo alguns deles, inimagináveis pelos outros membros do *Scrum Team*, para que conseguisse detectar os mínimos detalhes existentes que poderiam fugir dos requisitos definidos pelos clientes. Para este propósito, através dos conceitos adquiridos na

Figura 5.9 – Erro, Defeito e Falha



Fonte: Autor (2021)

disciplina Engenharia de Software, foram utilizadas diferentes perspectivas de teste de software. Sendo que a escolha de quais abordagens utilizar, dependeu de vários aspectos, como tempo, documentação disponível, necessidade, ciclo de vida do sistema, complexidade, além da ponderação do que, como e quando testar; para então construir o plano de teste da melhor forma possível, dentro das expectativas, para conseguir aproximar ao máximo da qualidade exigida para o produto.

Com isso, em relação ao projeto abordado neste documento, foram utilizados os testes em função do estágio do ciclo de vida do produto, envolvendo testes de integração, sistema, aceitação e de manutenção; além da utilização de testes em função do objetivo do teste, que envolveram os testes funcionais, não funcionais e de regressão. Os testes funcionais buscam analisar o comportamento do sistema, através de seu uso, com a orientação das documentações criadas para a analise de qualidade. Já os testes não funcionais, são utilizados para verificar a operação do sistema em casos inesperados.

Portanto, através do teste caixa preta, uma técnica funcional, foi possível focar exclusivamente nos requisitos funcionais do sistema, desconsiderando o código fonte e verificando aspectos da parte externa, sempre ponderando as especificações do programa. Dessa forma, permitiu analisar se um conjunto de cenários simulados alcançavam as respostas esperadas, focando no que o software realmente deveria fazer.

Para complementar, os testes de sistema, baseados em especificações de alto nível, também serviram para tratar o comportamento da aplicação, analisando requisitos não funcionais completos, incompletos ou não documentados. Esta abordagem maximizou a identificação dos erros, atentando-se às diferentes maneiras que os usuários poderiam acessar a aplicação.

Para então certificar ainda mais a qualidade do produto, foram executados os testes de configuração, que tiveram o objetivo de assegurar o funcionamento da aplicação em diferentes

ambientes, tanto hardware quanto software, verificando a responsividade dos itens exibidos em tela. Para isso, foram utilizados dispositivos de diferentes tamanhos, como tablets, celulares e computadores; assim como os sistemas operacionais *Windows* e *Linux*.

Além dos testes de configuração, também foram utilizados os testes de usabilidade, com a finalidade de compreender o quão intuitivo é o programa para o usuário final, permitindo realizar testes com os clientes para verificar se o produto satisfazia suas necessidades, assim como impressões; dando liberdade para que opinassem e definissem novas considerações sobre a organização do layout ou as funcionalidades.

Passando para uma parte mais técnica do sistema, foram utilizados os testes de integração. Realizados após a garantia do funcionamento individual de cada funcionalidade ou regra. Os testes de integração serviram para testar se as mesmas atuavam em conjunto, integradas umas às outras, averiguando aspectos sobre as dependências de suas componentizações, considerando tanto o *frontend*, parte dinâmica com a qual o usuário consegue interagir, quanto o *backend*, parte estrutural que apoia as ações do usuário. Então, através da utilização do DBeaver, Postman, Geoserver e QGIS foi possível assegurar que todos os relacionamentos entre as funcionalidades e os dados que seriam exibidos em tela estivesse em conformidade. Atividades realizadas baseando-se nos conhecimentos adquiridos nas disciplinas, Introdução a Sistemas de Banco de Dados e Sistemas Gerenciadores de Banco de Dados, permitindo com que o estagiário conseguisse realizar consultas e analisasse os dados da aplicação.

Obtendo todos os resultados dos diferentes tipos testes citados anteriormente, os testes de regressão juntamente com os de manutenção serviram para evitar a recorrência de problemas já tratados, assim como a possibilidade do surgimento de novas inconsistências, retirando todas as instabilidades do software e impedindo que ele fique defasado ou inoperante. Essas técnicas consistiram na repetição dos testes sobre as atualizações implementadas, sendo que em determinados casos, diferentes mudanças comprometeram a lógica do programa, invalidando alguns testes já realizados no processo de produção. Casos como esses ocorreram durante as correções emergenciais ou planejadas, mediante à alterações no ambiente de desenvolvimento, além das atualizações dos bancos de dados e versões do sistema. Dessa forma, a cada incremento, o estagiário precisava reexecutar todo o processo de teste de software sobre as atividades finalizadas pelo time de desenvolvimento.

Por fim, chegando a data de entrega final do produto, o sistema foi submetido aos testes de estresse, com a finalidade de mensurar o comportamento do software e avaliar seus limites de

tensão em relação ao suporte e tráfego de informações, levando em consideração que a aplicação envolvia uma grande quantidade de dados sensíveis. Com isso, consistindo na junção de todo o *Scrum Team* com as partes interessadas do projeto, foi testado o nível de performance que o produto estava alcançando, sendo que todos, de forma simultânea, consumiram um número considerável de requisições às funcionalidades de navegação do mapa, além da ativação de camadas e relatórios que exigiam grande custo computacional.

Aproveitando então a presença dos *Stakeholders* no teste de estresse, também foi realizado o teste de aceitação, onde foi validado se a aplicação estava em conformidade com todas as definições e se poderia ser considerada, consequentemente, como finalizada e pronta para o ambiente de produção. Caso surgisse considerações ou inconformidades, elas seriam retornadas para o ciclo de desenvolvimento do sistema, para que pudessem ser corrigidas e retestadas pelo *Scrum Team*, depois sendo então reapresentadas para os clientes até que fosse alcançado um acordo entre ambas as partes.

5.7 Participação na Planning

No início de cada *sprint* durante o processo de desenvolvimento do produto, foi necessária a realização de várias reuniões para o planejamento das atividades. Um plano criado através do trabalho colaborativo de todo o *Scrum Team*, por meio de duas etapas, o refinamento das atividades e a estimativa do tempo de execução das mesmas. Caracteriza-se como a primeira reunião dentre outras presente no *Scrum*, frente a uma *sprint*. Cada uma possuiu a duração definida em um time-box de no máximo dois dias, com a garantia, por parte do *Scrum Master*, que esse tempo fosse respeitado por todos os participantes. Segundo Schwaber K.; Sutherland (2013):

> "Os itens do Backlog do Produto selecionados entregam uma função coerente, que pode ser o objetivo da Sprint".

O *Produckt Backlog* é a entrada para a reunião de planejamento. Com ele o estagiário pode trabalhar diretamente na definição da meta da *Sprint* e o porquê da construção dos incrementos no sistema, onde as atividades foram difundidas em escopos menores, de acordo com os impactos para suas entregas, e documentadas no *Sprint Backlog*. O que ocasionou facilidade no entendimento sobre as funcionalidades do produto, por parte dos desenvolvedores, ao iniciar a implementação. Com isso, cada história de usuário definida pela *Product Owner* foi debatida,

analisando o método de execução, as possíveis dificuldades, dúvidas, assim como as formas de teste realizadas pelos analistas de qualidade.

Com as subdivisões das atividades, a equipe analisou e estimou a quantidade de esforço necessário para concluir as mesmas. Definindo o esforço em horas, essa etapa foi realizada considerando todas as atividades como partes independentes umas das outras, garantindo uma investigação individual e mais precisa. Para estimá-las, utilizando o *Scrum Poker*, as horas definidas por cada participante em momentos de divergência, eram debatidas sobre o motivo de suas escolhas até que todos chegassem a um acordo. Caso o time de desenvolvimento acusasse excesso ou falta de trabalho, as atividades presentes no *Sprint Backlog* eram renegociadas (SCHWABER K.; SUTHERLAND, 2013).

Ao final do planejamento, para completar o objetivo da *Sprint* e criar o incremento previsto no sistema de forma organizada, o *GitLab* foi utilizado para armazenar todas as atividades, por meio das *issues*. Cada *issue* é composta pelas definições do que é preciso ser feito, imagens do protótipo, dependências e observações, assim como o tempo gasto para finalizá-las, garantindo acesso para todos os membros do *Scrum Team* envolvidos no projeto.

5.8 Participação nas reuniões diárias

As reuniões diárias fizeram o papel de um evento chave para inspeção e adaptação dos integrantes do *Scrum Team*, em meio aos processos e ao avanço das tarefas realizadas durante o período de desenvolvimento do projeto. Com duração em um curto espaço de tempo e mantida no mesmo horário e local todos os dias, proporcionou esclarecer o que foi feito, o que era necessário colocar em prática no dia e identificar os obstáculos que impediam atender a meta definida na *Sprint*. Sendo essas, considerações apontadas por cada um dos participantes.

Mesmo sendo uma prática simples, focando sempre em oferecer uma autonomia ao time, onde os próprios integrantes poderiam conduzi-la, foi desejável a participação tanto da *Product Owner* (PO), que buscava sanar as dúvidas sobre as regras de negócio, quanto do *Scrum Master*, que mantinha o respeito e a exigência sobre as limitações de tempo definidas na cultura ágil da organização.

Segundo Jeff Sutherland, a ideia da reunião diária é ajudar a identificar a menor mudança, que é possível realizar no dia, para que a *Sprint* torne o time ainda mais rápido e a ajude entender melhor toda a organização imposta durante o desenvolvimento (GARRIDO, 2017). Dessa forma, através dos alinhamentos por parte da equipe, permitiu esclarecer muitas ques-

tões como, as responsabilidades direcionadas a cada membro, os níveis de conhecimento sobre as regras, a urgência das tarefas e, além disso, também permitia saber as dificuldades de desenvolvimento de cada um, sendo por falta de capacitação ou até mesmo referente à detalhes sobre o equipamento utilizado, entre muitos outros detalhes. Posicionamentos, que facilitaram a busca imediata de soluções plausíveis para os problemas encontrados, impedindo o processo de ser paralisado ou prejudicado, além de manter a Agência Zetta UFLA e as partes interessadas, cientes de todas as necessidades e andamentos provindos da execução das *Sprints*.

Além de alinhar todas as questões sobre a *Sprint*, as reuniões diárias proporcionaram uma maior aproximação e comunicação entre os integrantes do *Scrum Team*, mantendo todos informados do rendimento das etapas do trabalho. O que eliminou a necessidade de diferentes reuniões para incentivar o diálogo entre todos (SCHWABER K.; SUTHERLAND, 2013). Apesar de o *Scrum Master* reforçar que apenas os integrantes do *Scrum Team* podem participar das reuniões diárias, em algumas situações, membros de outras equipes precisaram alinhar questões importantes sobre o tratamento de dados do projeto, sendo estas, tarefas direcionadas apenas para a equipe de ciência do dados, pertencente à Agência Zetta UFLA.

5.9 Participação na revisão

Se caracterizando por ser uma reunião informal, ela foi executada como uma das últimas atividades de cada *Sprint*, cujo objetivo era inspecionar o incremento do produto, definido juntamente com a meta da *Sprint* na fase de planejamento das atividades. Com a participação da *Product Owner*, que também atuou no papel originalmente definido para os *Stakeholders*, e da Gerente de Projetos da tribo, foram apresentados e esclarecidos todos os itens do *Sprint Backlog*, executados e dados como "finalizados", assim como os quais não foram, entrando na *Sprint* seguinte.

Com isso, a *Product Owner*, juntamente com o *Scrum Team*, puderam colaborar na definição dos artefatos que futuramente seriam abordados para otimizar o valor do produto. Atividade realizada até a penúltima *Sprint* de desenvolvimento. Na última reunião de revisão, foram discutidas as questões apontadas sobre a apresentação do produto, comportando-se como um *feedback* para o time, onde o produto executou perfeitamente, restando apenas atividades referentes à realização de *deploy* para o ambiente de produção e relacionadas à atualização dos dados presentes na aplicação, assim como as documentações necessárias. Segundo Lerche-Jensen (2019), a revisão do *Sprint* inclui diferentes elementos, como:

- Os envolvidos na reunião são a equipe de desenvolvimento e os convidados pelo Dono do Produto.
- O responsável por ser a ligação entre o cliente e a empresa apresenta:
  - atividades que foram feitas durante a *Sprint*
  - atividades que foram feitas durante a *Sprint*
  - esforço adicionado
  - atividades removidas da *Sprint*
- A equipe de desenvolvimento apresenta:
  - o que ocorreu de certo durante a execução da *Sprint*
  - quais impedimentos tiveram
  - como os impedimentos foram solucionados
- O time de desenvolvimento define o trabalho como concluído e possivelmente utilizável para implementação.
- A situação dos requisitos do produto é apresentado e explicado pelo dono do produto.
- Todos debatem sobre o que é preciso ser feito, a fim de garantir valor para a próxima reunião de planejamento.
- Será realizado um levantamento sobre as possíveis mudança que o produto pode possuir.
- Também podem ser discutidas questões sobre orçamentos e cronogramas dos próximos incrementos do produto.

Além da análise e revisão do *Sprint Backlog*, para então atualizar os itens presentes no *Product Backlog*, também foi apresentado, em certos momentos, o documento responsável para informar o progresso a caminho do objetivo da *Sprint*, o *Burndown*. Um gráfico, que possibilitou verificar se as atividades estavam dentro do esperado, no que se refere ao cronograma definido no início de cada *Sprint*. Dessa forma, a partir dele, o *Scrum Team* consegue reorganizar sua atuação em uma próxima *Sprint*, caso identifique certas inconsistências na ferramenta visual.

5.10 Participação na retrospectiva

Através de uma *issue* já cadastrada, juntamente com o término do planejamento das atividades no início da *Sprint*, os integrantes do *Scrum Team* tinham a possibilidade de apontar como a *Sprint* estava se comportando durante sua execução. Com a definição do que poderia continuar ou parar, possibilitou traçar planos e possíveis ações que buscariam melhorar continuamente o processo de desenvolvimento, levando em consideração a *Sprint* seguinte.

Sendo uma oportunidade de auto-inspeção do time, a retrospectiva é tida como a última cerimônia realizada em cada *Sprint*, onde permitiu analisar e debater todas as considerações apontadas por cada colaborador, identificando as relações pessoais, os processos, ferramentas, aspectos positivos e negativos, assim como as potenciais melhorias para tornar o ambiente de trabalho mais agradável e produtivo.

A existência de várias Técnicas de Retrospectiva *Scrum* permite que o *Scrum Master* defina a melhor técnica para o momento, no entanto, a utilizada durante o estágio foi o modelo "*start-stop-continue*". Nesse modelo de retrospectiva da *Sprint*, cada membro da equipe de desenvolvimento deve responder a três questões (MÉTODOÁGIL, 2021):

- O que devemos começar a fazer? (*start*)
- O que devemos parar de fazer? (*stop*)
- O que devemos continuar a fazer? (*continue*)

Possuindo um tempo curto de duração e sendo executada de forma totalmente colaborativa, permitiu a participação de todos os integrantes, onde cada membro teve a oportunidade de apontar suas ideias. Para que isso fosse possível, o *Scrum Master* chamou um por vez, com o intuito de que todos debatessem sobre os temas abordados, informando as concordâncias e discordâncias, além de possíveis sugestões que complementassem os argumentos listados. Não importando o quão bom seja o *Scrum Team*, é sempre necessário, por parte dos membros que o compõem, buscar oportunidades para melhorar o próximo ciclo de desenvolvimento.

6 CONCLUSÃO

Este relatório de estágio apresentou o processo de atuação com atividades de Qualidade de Software em sistemas Web na Agência Zetta UFLA. Ao acompanhar o desenvolvimento e implantação do sistema em uso, o discente teve a oportunidade de participar de todas as etapas do ciclo de desenvolvimento, desde a definição dos requisitos, regras e funcionalidades, até sua entrega. Com isso, o estágio permitiu correlacionar diversos conceitos abordados em múltiplas disciplinas ministradas no curso de Ciência da Computação, como Engenharia de Software, Interação Humano-Computador, Introdução a Sistemas de Banco de Dados, Sistemas Gerenciadores de Banco de Dados, Processos de Software, Gerência de Projetos de Software, entre outras. Isso facilitou uma melhor adaptação do estagiário para com o ambiente de trabalho e seu papel nas diferentes áreas de atuação da organização.

A prática no processo de estágio também permitiu ao discente compreender sobre as diferentes funções e responsabilidades em uma equipe de desenvolvimento de software. A Agência Zetta UFLA proporcionou, em certos momentos, cooperação na troca de conhecimento por meio de comunicações efetivas com colaboradores mais experientes. Este aspecto foi resultado de sua cultura organizacional, onde define que as equipes necessitam de união e colaboratividade entre seus membros, facilitando a busca de soluções para diferentes empecilhos e aprimorando a agilidade no processo de desenvolvimento, para juntos alcançar o objetivo em comum, a entrega do projeto. Diante disto, o discente, ao longo do tempo, desenvolveu uma maior autonomia na realização de suas obrigações, além de possuir a capacidade de auxiliar diferentes colaboradores em frentes de trabalho com questões fora de seu escopo, sempre garantindo competência e valor na entrega de suas atividades. Estas ações resultaram reconhecimento, por parte da agência, como consequência dos serviços prestados.

Dentre as diferentes frentes de trabalho que o estagiário auxiliou, não contando suas obrigações como analista de qualidade, destacam-se a elaboração da documentação das regras de negócio e o desenvolvimento dos protótipos do sistema. O documento de regras possibilitou maior imersão no processo de desenvolvimento do projeto. Com ele, foi possível para o estagiário compreender as definições dos requisitos e melhorar o controle sobre os processos. Ao ter um maior conhecimento das regras, o estagiário possuiu maior facilidade nos processos de tomada de decisão e percepção de erros sobre as atividades destoantes do que era proposto pelos *Stakeholders*. Já a elaboração dos protótipos permitiu o estagiário compreender melhor

sobre a interação das funcionalidades existentes no sistema, validando os conceitos e possíveis melhorias que poderiam ser incorporadas durante o seu desenvolvimento.

O discente, em meio ao desempenho do estágio, também conseguiu compreender, de fato, a importância da implantação de metodologias ágeis no desenvolvimento de software. Embora já tenha tido contato com os conceitos, através de trabalhos práticos em diferentes disciplinas, essas experiências caracterizaram-se apenas como simples simulações, resultando em implantações falhas das metodologias. Justamente por não abordar todos os ritos, as diferentes composições do *Scrum Team* e, até mesmo, a não utilização do quadro de cartões para o *framework Kanban*. Com isso, o estágio na Agência Zetta UFLA proporcionou ao discente suprir todas estas lacunas encontradas durante a graduação, preenchendo-as com conceitos antes não colocados em prática. No estágio, o discente teve possibilidade de executar as metodologias ágeis seguindo todas as cerimônias existentes nelas, além da possibilidade de produzir os artefatos envolvidos nos processos, realmente exercendo suas funções de maneira correta.

A realização dos testes de software, a principal atividade como responsabilidade do estagiário, foi essencial para que o projeto tivesse garantia na qualidade de suas funcionalidades, assim como em sua navegação e layout. Com isso, o discente pode compreender a importância de se localizar possíveis fragilidades ao longo do ciclo de vida de desenvolvimento do sistema, cuja prática ofereceu métodos e ferramentas antes não vistas e praticadas, como por exemplo, as de gerenciamento de banco de dados. Também foi possível adquirir conhecimento sobre os diferentes tipos de inconsistências, sendo elas erro, defeito e falha; que o ajudaram no processo de testes, auxiliando diretamente na priorização das atividades de reparo dos erros encontrados. Isso permitiu gerenciar para que as tarefas mais relevantes fossem priorizadas durante a implementação, garantindo que a entrega de um produto de valor, para os clientes, fosse respeitada.

Ainda a respeito dos testes, foi possível observar que, além da garantia de qualidade do sistema, eles também são utilizados como utensílios pela agência, para minimizar os custos. Com isso, foi possível levar em consideração que defeitos e erros podem causar muitos prejuízos no futuro. Tendo isso em mente, o estagiário pode observar, na prática, a notabilidade de se realizar os testes desde a primeira fase do desenvolvimento do sistema até sua entrega. Ao detectar os problemas no início, o tempo e esforço para corrigi-los foi reduzido drasticamente, comparado com os descobertos, em outros projetos, no ambiente de produção.

Sendo caracterizada como a primeira experiência, o estágio influenciou diretamente nas tomadas de decisão do discente, onde pôde conhecer diferentes áreas de atuação que antes não

eram de seu conhecimento. Outro fator que o influenciou foi a busca por autonomia de suas atividades, o que proporcionou maior segurança e crescimento em sua carreira profissional no mercado de trabalho.

Diante de todos os pontos levantados anteriormente, conclui-se que o estágio é uma fase necessária no fim da graduação, pois possibilita associar o conhecimento científico, adquiridos por meio dos conceitos e teorias oferecidas pelas diferentes disciplinas cursadas, com o cotidiano de um ambiente de trabalho, promovendo futuros profissionais da área de formação, e facilitando seu entendimento sobre as diferentes práticas que direcionam o desenvolvimento profissional do discente.

REFERENCES

ADOBE. Adobe Xd, 2021. Disponível em: <https://www.adobe.com/br/products/xd.html>. Acesso em: 21 fev. 2021.

ALMEIDA, L. C. de. Análise espacial de dados com o quantum gis: exercícios realizados durante tópico especial ofertado pelo programa de pós-graduação em geografia da ufsc. **Observatorium: Revista Eletrônica de Geografia**, v. 3, n. 8, 2011.

ANDERSON, D. J. **Kanban: successful evolutionary change for your technology business**. Washington: Blue Hole Press, 2010. 261 p.

ARRUDA, L. V. **Desenvolvimento Ágil de Software:**: uma análise sintética a partir da metodologia kanban. Palmas, Tocantins: Anais do VII CONNEPI – Congresso Norte Nordeste de Pesquisa e Inovação, 2012. Disponível em: <http://propi.ifto.edu.br/ocs/index.php/connepi/vii/paper/viewFile/3644/961>. Acesso em: 18 fev. 2021.

BACK, N. et al. Projeto integrado de produtos: planejamento, concepção e modelagem. Barueri: Manole, 2008.

BARTIÉ, A. Garantia da qualidade de software: Adquirindo maturidade organizacional. **Rio de Janeiro**, editora Campos, p. 26, 2002.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML: guia do usuário.** Rio de Janeiro: Elsevier Brasil, 2005.

BORTOLUCI, R. et al. Análise das características do kanban e de possíveis procedimentos aplicáveis a métodos de produção de software. **X WORKSHOP DE PÓS-GRADUAÇÃO E PESQUISA DO CENTRO PAULA SOUZA**, São Paulo, 2015. ISSN 2175-1897.

DBEAVER. **DBeaver, universal database client**. 2021. Disponível em: <https://dbeaver.com/>. Acesso em: 23 mar. 2021.

GAMEIRO, L. **Adobe Xd:**: O que é e como usar. POST DIGITAL, 2021. Disponível em: <https://www.postdigital.cc/blog/artigo/adobe-xd-o-que-e-e-como-usar>. Acesso em: 21 fev. 2021.

GARRIDO, M. **Mitos e Fatos sobre a Daily Scrum**. K21, 2017. Disponível em: <https://k21.global/blog/mitos-fatos-daily-scrum>. Acesso em: 07 mar. 2021.

GEOSERVER. **DBeaver, universal database client**. 2021. Disponível em: <https://learning.getpostman.com/docs>. Acesso em: 03 mar. 2021.

GITLAB. 2021. Disponível em: <https://about.gitlab.com/>. Acesso em: 20 fev. 2021.

LERCHE-JENSEN, S. **Fundamentos Internacionais do Scrum Master**. SCRUM.AS, 2019. Disponível em: <https://www.scrum.as/academy.php?show=5&chapter=26&name=6.4>. Acesso em: 07 mar. 2021.

LIMA, A. da S. **UML 2.3-do Requisito à Solução**. 1. ed. São Paulo: Érica, 2011.

MELO, A. C. **Desenvolvimento de aplicações com UML 2.2: Do conceitual a implementação. 3ª edição**. Rio de Janeiro: Brasport Livros e Multimídia Ltda, 2010. 340 p.

MIRO. 2021. Disponível em: <https://miro.com/about/>. Acesso em: 19 fev. 2021.

MUELLER, E. **What is DevOps**. 2019. Disponível em: <https://theagileadmin.com/what-is-devops/>. Acesso em: 20 fev. 2021.

MÉTODOÁGIL. **Sprint Retrospectiva:**: Em busca da melhoria continua. 2021. Disponível em: <https://www.metodoagil.com/sprint-retrospectiva/>. Acesso em: 07 mar. 2021.

ODEH, M.; KAMM, R. Bridging the gap between business models and system models. **Information and Software Technology**, Elsevier, v. 45, n. 15, p. 1053–1060, 2003.

OLIVEIRA, E. Geo na web: As novidades da internet geográfica. **Periódico, edição 55 – InfoGeo**, Curitiba, 2008.

PEJOVIĆ, M. et al. Solving a surveying problem by using r and qgis: Setting out of a land expropriation zone. **Geonauka**, Savez geodeta Srbije, Beograd, v. 2, n. 2, p. 12–18, 2014.

PINHEIRO, T.; ALT, L. **Design Thinking Brasil: empatia, colaboração e experimentação para pessoas, negócios e sociedade**. Rio de Janeiro: Alta Books Editora, 2011.

POSTMAN. Postman Docs, 2021. Disponível em: <https://learning.getpostman.com/docs>. Acesso em: 25 fev. 2021.

PRESSMAN, R. S. **Engenharia de Software**. 7. ed. Porto Alegre: Pearson Makron Books, 2011.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. **Métodos Ágeis para Desenvolvimento de Software**. 1. ed. Porto Alegre: Bookman Editora, 2014. 312 p.

PROVALORE. **Diferenças entre Design Thinking e Design Sprint**. 2021. Disponível em: <https://andersonmedeiros.files.wordpress.com/2010/04/tutorial-qgis-0-8-1.pdf>. Acesso em: 23 mar. 2021.

QGIS. 2021. Disponível em: <https://qgis.org/pt_BR/site/>. Acesso em: 19 fev. 2021.

ROMEIRO, E. et al. **Projeto do produto**. São Paulo: Elsevier Brasil, 2010.

SCHWABER K.; SUTHERLAND, J. **Guia do Scrum-Um guia definitivo para o Scrum:**: As regras do jogo. 2013. Disponível em: <https://scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em: 23 fev. 2021.

SCRUMGUIDES.ORG. **What is Scrum?** 2021. Disponível em: <http://www.scrumguides.org>. Acesso em: 19 fev. 2021.

SILVA, J. **Tutorial em sistema de informação geográfica para o Quantum Gis**. 2010. Disponível em: <https://andersonmedeiros.files.wordpress.com/2010/04/tutorial-qgis-0-8-1.pdf>. Acesso em: 05 mar. 2021.

SOMMERVILLE, I. **Engenharia de Software**. 8. ed. São Paulo: Pearson Addison Wesley, 2007.

TOMLINSON, T. Integrating drupal 8 development: For advanced projects and large development teams. In: . 1. ed. Tigard, Oregon, USA: Springer, 2017. p. 309.

VANCOUVER, S. D. **Service Design:**: The design process. 2014. Disponível em: <https://servicedesignvancouver.ca/wp-content/uploads/2014/11/SDV-DoubleDiamond.pdf>.