# Revisão do Pré-projeto

**Disciplina: Trabalho de Conclusão de Curso I – SIS**

Caro, orientando,

segue abaixo o Termo de Compromisso, as DUAS revisões do seu pré-projeto contendo a avaliação do professor “avaliador” e professor “TCC1”. É muito importante que revise com cuidado e discuta possíveis dúvidas decorrente das revisões com o seu professor orientador, e com o professor de TCC1. Sempre procure fazer todos os ajustes solicitados, até mesmo os menores detalhes, pois todos são importantes e irão refletir na sua nota nesta disciplina. Lembre de abrir localmente em um visualizador PDF para poder ver as anotações que foram feitas. E, aparecendo uma anotação feita por mim (prof. De TCC1) que inicie com “TF-...” (ex. “TF-ALÍNEA”) se refere a ajustes de formatação indicando que deve usar o estilo do Word correto do modelo do projeto.
Mas, caso o professor orientador julgue que algumas anotações das revisões não devam ser feitas, ou mesmo que sejam feitas de forma diferente a solicitada pelo revisor, anexe ao final do seu projeto a ficha “Projeto: Observações – Professor Orientador” disponível no material da disciplina, e justifique o motivo.
Lembrem que agora o limite de páginas do projeto é no máximo 16 (dezesseis) páginas.

Atenciosamente,

Dalton Solano dos Reis:66568803987

Digitally signed by Dalton Solano dos Reis:66568803987
Date: 2022.11.06 23:42:23 -03'00'

UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
CURSO DE SISTEMAS DE INFORMAÇÃO

# TERMO DE COMPROMISSO

| I – IDENTIFICAÇÃO DO ALUNO | |
|---|---|
| Nome: | João Manoel Sanches Lima |
| CV Lattes: | http://lattes.cnpq.br/8570536659191368 |
| E-mail: | jmslima@furb.br |
| Telefone: | (47) 997213459 |

| II – IDENTIFICAÇÃO DO TRABALHO | |
|---|---|
| Título provisório: | TEAM MANAGEMENT: SISTEMA PARA GESTÃO DE EQUIPES DE QUALIDADE DE SOFTWARE. |
| Orientador: | Simone Erbs da Costa |
| Coorientador (se houver): | |
| Linha de Pesquisa: | ☐ Tecnologias aplicadas à informática na educação<br>☒ Tecnologias aplicadas ao desenvolvimento de sistemas |

## III – COMPROMISSO DE REALIZAÇÃO DO TCC

Eu (aluno), João Manoel Sanches Lima

comprometo-me a realizar o trabalho proposto no semestre 1/2023 , de acordo com as normas e os prazos determinados pela FURB, conforme previsto na resolução nº.20/2016.

Assinatura: NÃO É NECESSÁRIO – Encaminhar por mail ao orientador

## IV – COMPROMISSO DE ORIENTAÇÃO

Eu (orientador), Simone Erbs da Costa

comprometo-me a orientar o trabalho proposto no semestre 1/2023 , de acordo com as normas e os prazos determinados pela FURB, conforme previsto na resolução nº.20/2016.

Assinatura: NÃO É NECESSÁRIO – Encaminhar por mail ao professor de TCC I

Blumenau, 05 de Agosto de 22

| CURSO DE SISTEMAS DE INFORMAÇÃO – TCC ACADÊMICO | |
|---|---|
| ( x) PRÉ-PROJETO ( ) PROJETO | ANO/SEMESTRE: 2/2022 |

# TEAM MANAGEMENT: SISTEMA PARA GESTÃO DE EQUIPES DE QUALIDADE DE SOFTWARE

Aluno: João Manoel Sanches Lima

Prof. Simone Erbs da Costa – Orientadora

## 1 INTRODUÇÃO

A busca por melhor qualidade tem orientado trabalhos e pesquisas desde sempre. Para Salviano (2020), desde que o ser humano começou a desenvolver e utilizar ferramentas, o conhecimento sobre como fazer, e, ainda melhor aquilo que é feito, foi aumentando. Esse conceito de "fazer melhor" está fortemente relacionado com a qualidade, independente da definição específica utilizada (SALVIANO, 2020). Santos e Oliveira (2017) observam que a qualidade de software é um ponto de suma importância no desenvolvimento de um projeto, ou seja, uma tarefa gerencial de natureza contínua que requer execução durante o ciclo de vida do software.

Wazlawick (2019) coloca que a qualidade de software está relacionada a disciplina de teste. A disciplina de teste passou a ser considerada extremamente importante, fazendo parte do processo de desenvolvimento de software (WAZLAWICK, 2019). Os testes de software também foram incorporados nos métodos ágeis como uma atividade crítica, assumindo inclusive que os casos de teste deveriam passar a ser escritos antes das unidades de software que pretendem testar (PRESSMAN; MAXIM, 2021). A finalidade desse processo de testes de software é proporcionar informações que garantam a qualidade do produto e dos processos de teste (PEREIRA; VARGAS, 2019). Dessa forma, Xavier *et al.* (2019 *apud* SOMMERVILLE, 2011) colocam que a realidade no mercado de software propõe inúmeras alterações tanto na forma de utilização quanto na forma de desenvolver ou adquirir um software com a devida qualidade.

Para Fraga e Barbosa (2017), os métodos ágeis vêm ganhando mais espaço nesse mercado por serem práticas mais focadas na adaptação dos processos de uma forma incremental e iterativa. Isso possibilita aumentar a produção da equipe e melhorar a qualidade, bem como evitar erros com entregas mais contínuas (FRAGA; BARBOSA, 2017). Devatz e Polido (2017 *apud* BINATO, 2017) afirmam que um dos maiores desafios do mundo corporativo é o trabalho em equipe. Diante desse cenário, este trabalho propõe o desenvolvimento de um sistema para auxiliar no gerenciamento de times de teste. Conjectura-se que a construção deste sistema auxilie o gestor a aprimorar a eficiência da equipe, gerando resultados mais eficientes e aumentando o desempenho da equipe.

## 1.1 OBJETIVOS

O Objetivo geral deste trabalho proposto é desenvolver um sistema para auxiliar na gestão dos testes nas equipes de qualidade de software, auxiliando no desempenho da equipe e buscando mais eficiência nos resultados. Os objetivos específicos são:

a) gerenciar e criar planos de testes que devem ser executados pela equipe por meio de interfaces disponibilizadas;
b) controlar o trabalho da equipe de testes utilizando os métodos ágeis como o Scrum e o Kanban no desenvolvimento das funcionalidades;
c) analisar e avaliar as funcionalidades e a usabilidade e a experiência do usuário das interfaces desenvolvidas, por meio do Método Relationship of M3C with User Requirements and Usability and Communicability Assessment in groupware (RURUCAg).

# 2 TRABALHOS CORRELATOS

Nesta seção serão descritos os trabalhos correlatos que possuem semelhanças com o trabalho proposto. A seção 0 traz o sistema Jira focado em gestão de trabalho que utiliza soluções ágeis como Kankan (ATLASSIAN, 2019). A seção 2.2 apresenta o sistema Qtest que objetiva o gerenciamento e a análise de teste (TRICENTIS, 2019). Por fim, a seção 2.3 descreve o sistema de gerenciamento de teste TestLink que tem como foco principal o gerenciamento de casos testes de software (REIS, 2018).

## 2.1 JIRA SOFTWARE: GESTÃO DE PROJETOS E MONITORAMENTO DE TAREFAS.

A Atlassian (2019) traz o Jira que faz parte de um conjunto de soluções ágeis, proporcionado que seja realizado o gerenciamento e potencializado o trabalho de equipes no desenvolvimento ágil de sistemas. O Jira é amplamente utilizado dentro das empresas para o gerenciamento de equipes, criação e controle de sprint. Ele pode ser utilizado por desenvolvedores, analistas de qualidade, gerentes de projeto, scrum master, analistas de negócio e designers de produto/User eXperience (UX). O Jira foi desenvolvido usando as linguagens Node.JS, Java, Vue.js e está disponível para as plataformas Web, Android e iOS. Algumas das funções do Jira são: gestão de times, controle de sprints, fluxo de atividades, gestão de casos de teste, histórico de atividades, estatísticas e permissões de usuário (ATLASSIAN, 2019).

O Jira permite a criação de uma listagem de demandas a serem feitas pela equipe na visão do gestor. Nessa listagem é possível ver os tipos de demandas que foram abertas, quem criou a demanda, o responsável por ela no momento, o seus status e o tempo estimado para sua

execução. Essa listagem permite que o usuário tenha uma visão geral do que acontece com o projeto, mostrando as principais informações sobre as demandas além de permitir a criação, exclusão e alteração delas. Outra função que ela permite é o acesso as estatísticas da equipe e a base de informação do projeto (ATLASSIAN, 2019).

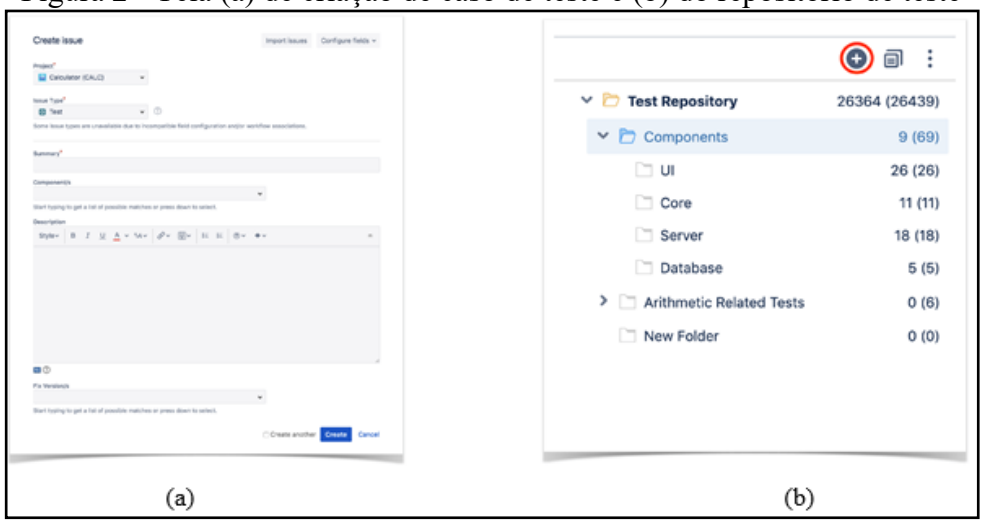
A Figura 1 traz a tela do quadro Kanban. Nessa tela é possível visualizar as demandas separadas em colunas e por status, criando assim o fluxo das atividades. Esse quadro permite ao usuário movimentar os cartões (demandas) para qualquer status que seja necessário, assim como incluir ou excluir as etapas no fluxo de atividades da sprint. Pela Figura 1 também é possível ver as informações das demandas nos cartões, em cada uma delas é mostrado o `título`, o `tipo de demanda`, a sua `prioridade` e o `responsável` por ela. Na parte superior (de cima para baixo, destacado pela letra A) consta os controles para gerenciar a sprint. Ela permite que o usuário controle o tempo dessa sprint, quando ela vai ser terminada usando o botão `Complete sprint` e as suas configurações (ATLASSIAN, 2019).

Figura 1 - Tela do Quadro Kanban


Fonte: Atlassian (2019).

A Figura 2 (a) traz a tela para criar o caso de teste. Nela o usuário pode inserir o projeto a qual o caso de teste pertence, a sua descrição, a versão em qual o caso foi ou deve ser executado e os seus componentes. Esta tela permite ao usuário adicionar ou excluir os campos do formulário usado na criação dos casos de teste e fazer o link com outro caso de teste ou demanda já criada. Já Figura 2 (b) mostra o repositório de testes no qual é possível ver todos os testes criados separados por pastas e o gerenciamento deles. O usuário também pode fazer alterações nos casos de testes e ao clicar no botão + ele pode adicionar novas pastas ao repositório (ATLASSIAN, 2019).

Figura 2 - Tela (a) de criação de caso de teste e (b) do repositório de teste




Fonte: Atlassian (2019).

Em cada demanda criada no Jira é gerado um histórico de atividades que salva todas as alterações feitas nela. Na demanda criada tem duas áreas para o registro de atividades. A primeira é para o registro de tempo gasto na atividade, nela o usuário registra o que ele fez e quanto tempo ele gastou na demanda. A segunda área é para o registro das alterações feitas na demanda durante o seu trâmite, com a inserção de um anexo, mudança de status, responsável pela issue e a data e hora da alteração (ATLASSIAN, 2019).

O Jira fornece uma série de gráficos que auxiliam na gestão e no monitoramento do time. Com os relatórios de análise de demandadas a equipe ou o gestor consegue saber quanto tempo é gasto na resolução das demandas, quantas demandas foram abertas ou resolvidas em um determinado período e é possível realizar a comparação dessas informações. Ele também possui relatórios e gráficos para o controle do tempo e gestão da equipe, assim como gera relatórios para controle das sprints para o gestor (ATLASSIAN, 2019).

## 2.2 QTEST: UMA FERRAMENTA PARA GERENCIAMENTO E ANÁLISE DE TESTE.

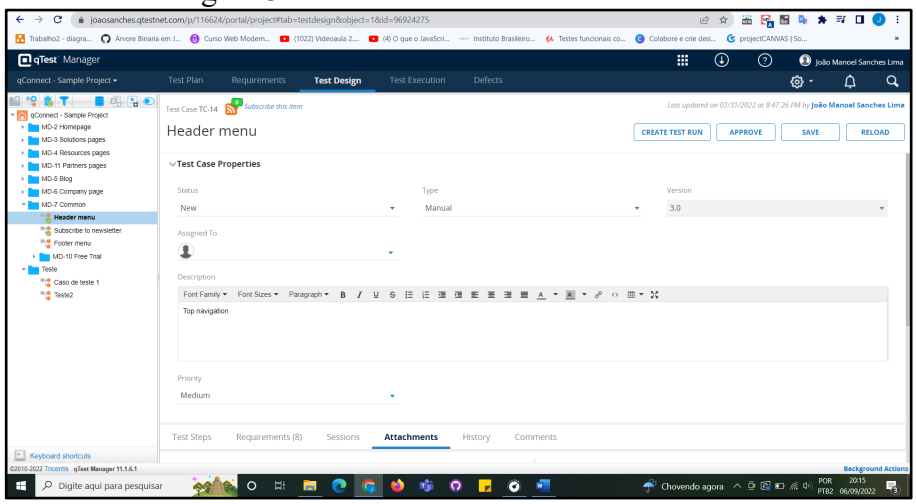
A Tricentis (2019) traz o Qtest, que é um sistema Web para o gerenciamento e análise de casos de teste nas equipes de qualidade de software. O sistema permite que os gestores façam um gerenciamento completo dos testes, podendo criar um projeto de testes com todos os planos e casos de teste da equipe. Além disso o sistema possibilita que se faça uma análise completa dos testes executados. As funções mais destacadas do Qtest são: gestão de time, gestão de casos de teste com a criação de planos e repositórios de teste, historio de atividades, estatísticas por meio de geração de relatórios e permissões de usuário (TRICENTIS, 2019).

O Qtest permite ao usuário gerenciar a equipe de qualidade, captando dados e estatísticas durante a execução dos testes. O sistema possibilita a criação de um projeto de testes que pode ser composto por planos de teste, suítes de casos testes, versionamento de versões e listagem

de requisitos. O sistema também possui um controle sobre os usuários, permitindo que o gestor adicione contas de usuários e o seu nível de permissão (TRICENTIS, 2019).

A Tricentis (2019) traz um gerenciamento de casos de teste, permitindo que o usuário tenha controle sobre os casos que foram criados. Nesse sentido, a Figura 3 traz a tela para a criação do caso de teste, na parte de cima (de cima para baixo) é possível fazer a inserção das informações básicas como o título, responsável, tipo de teste e a versão. Já na parte inferior da referida figura consta as informações complementares como os passos do teste para a sua execução, os anexos, os requisitos necessários do teste, o seu histórico e os comentários relacionados a ele. O Qtest permite ao usuário organizar os casos de teste por meio de um repositório, separando-os por projetos e pastas (TRICENTIS, 2019).

Figura 3 – Tela de cadastro de casos de teste




Fonte: Tricentis (2019).

Segundo Tricentis (2019), o sistema possui um histórico de versões para cada função gravando todos os dados que foram alterados e os listando posteriormente. Em cada aba o sistema guarda as alterações que foram feitas por meio de um histórico detalhado e os exibe por meio de uma lista. Tricentis (2019) ainda coloca que o sistema possui um fácil acesso aos históricos, no qual em cada função existe uma aba com o histórico e os usuários conseguem acessá-lo livremente podendo ver quem fez a alteração, quando e o que foi alterado.

De acordo com Tricentis (2019), o Qtest traz um painel com menus e métricas do sistema, permitindo que o usuário tenha um controle sobre o andamento do projeto. Nesse sentido, o sistema traz um menu com os dados dos testes executados, mostrando a velocidade em que os casos foram executados, a cobertura que eles atingiram e qualidade dos testes. Além disso, o sistema possui um dashboard que mostra em tempo real quantos casos foram executados, bloqueados, falharam e ele também permite a geração de relatórios sobre eles. Ele também exibe gráficos, deixando mais visual a demonstração dos dados e possibilitando que o

usuário altere os gráficos por meio de filtros e exibe as informações de um determinado período ou versão do sistema (TRICENTIS, 2019).

Cabe destacar, que o Qtest possibilita ao gestor adicionar usuário, gerando um *login* e senha e colocando um nível de permissão de acesso ao sistema. O gestor pode administrar todos os usuários, assim como pode os separar em grupos de projetos e de permissão. Além disso, o sistema possibilita que o gestor designe tarefas para o usuário o colocando como responsável por ela. A ferramenta também permite que o gestor possa rastrear as atividades que um usuário executou durante os testes, assim como ver quais casos foram designados ou executados por um determinado usuário (TRICENTIS, 2019).

## 2.3 TESTLINK: UMA FERRAMENTA DE GERENCIAMENTO DE TESTES DE SOFTWARE

De acordo com Reis (2018), o TestLink é um sistema open source para o gerenciamento de testes. Ele permite que as equipes de testes trabalhem de forma sincronizada no mesmo espaço de trabalho seja presencialmente ou remotamente. O Test Link foi disponibilizado na plataforma Web e tanto o seu *back-end* como o *front-end* foram desenvolvidos utilizando a linguagem Hypertext PreProcessor (PHP). Ele também possui uma integração com os bancos PostgreSQL e MySQL, permitindo que os dados do projeto de teste sejam salvos externamente. Algumas das funções do TestLink são: gestão de times, a criação de fluxo de atividades, gestão de casos de teste, histórico de atividades, estatísticas e permissões de usuários (REIS, 2018).

Segundo Reis (2018), o TestLink permite que o usuário faça a gestão do seu time adicionando colaboradores nos projetos de teste. O usuário administrador pode gerenciar o desempenho de cada membro do time e controlar cada conta que foi adicionada dando um nível de permissão. Além disso, o sistema permite que cada usuário seja colocado em um projeto de teste específico. Já sobre o fluxo de atividades, Reis (2018) coloca que ele pode ser configurado pelo usuário. Nele é possível cadastrar um projeto de teste, uma baseline de teste, um plano de teste, uma suíte de teste e casos de teste. Nos fluxos de execução de um plano ou caso de teste ~~a~~ três ações podem ser tomadas: quando o teste é executado com sucesso o caso é aprovado; caso o teste não possa ser feito o caso é bloqueado e o motivo do bloqueio do caso de teste é descrito; quando é encontrado um erro ou uma falha no teste o caso é reprovado e o problema é descrito no caso de teste para documentação e análise do problema (REIS, 2018).

Referente a gestão de casos de teste, o Test Link permite ao usuário criar um projeto de testes específico para cada produto da empresa, dentro do projeto é permitido o cadastro da plataforma e versão do software em que os testes serão executados. Ele também possibilita a criação de planos de testes para cada plataforma que precisa ser testada, para a execução de

testes regressivos em versões do sistema e permite a organização dos casos de teste em suítes os separando por função a ser testada (REIS, 2018).

Reis (2018) destaca um histórico de versões e execuções em cada caso, suíte, plano ou projeto de teste. A cada modificação do caso de teste o sistema grava quem o criou e quem o alterou por último e ele também possibilita a criação de várias versões do mesmo caso de teste. Ao executar o caso de teste o sistema guarda a informações de cada caso, qual foi o status dado a ele, a data, hora e o usuário que executou o caso de teste. Além disso, segundo Reis (2018), o sistema possui um painel de métricas que gera uma série de informações sobre os projetos de teste que foram criados e a forma que são executados. Durante a execução de um plano de teste o Test Link consegue gerar um relatório com o número de casos de teste que foram executados no dia e no total, quais foram aprovados, reprovados e bloqueados. Dessa forma, o sistema consegue gerar gráficos prevento o tempo de execução dos planos de teste se baseando no número de casos de testes executados por dia (REIS, 2018).

Reis (2018) ainda destaca que o controle de permissão e de usuários possibilita um trabalho em equipe sincronizado e de vários lugares. Nesse sentido, o sistema permite a adição de usuários a um projeto com diferentes níveis de acesso como administrador, líder de testes, projetista de teste, testador e convidado. Na criação de um plano de teste o sistema tem a opção que se faça a designação de um testador para casos de teste específicos (REIS, 2018).

# 3 PROPOSTA DO SISTEMA

Nesta seção serão descritas as justificativas para o desenvolvimento do trabalho proposto na subseção **Erro! Fonte de referência não encontrada.**; os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF) na subseção **Erro! Fonte de referência não encontrada.**; e para finalizar será descrito as metodologias e planejamento do cronograma para o desenvolvimento desta proposta na subseção 3.3.

## 3.1 JUSTIFICATIVA

A relevância do projeto proposto foi evidenciada nas seções 1 e 2. Segundo Iteris (2022), a gestão da qualidade de sistema é indispensável para qualquer empresa que deseja garantir a entrega do produto certo de forma correta. Pois, por meio dela a empresa consegue otimizar o desenvolvimento, garantindo um produto funcional e que agregue valor ao negócio (ITERIS, 2022). Já para Baumgartner (2022), o controle da qualidade de sistema é um conjunto de atividades técnicas que visam monitorar o processo de desenvolvimento, fazendo com que teste possua funcionalidades e características que atendam as expectativas de todos os envolvidos no projeto. Nesse contexto, Atlassian (2019), Tricentis (2019) e Reis (2018) identificaram a possibilidade de criar um sistema para a gestão de qualidade de sistema, possibilitando tanto

que o gerenciamento dos testes como da equipe de qualidade. No Quadro 1 é mostrado um comparativo dos trabalhos correlatos descritos na seção 2, as linhas representam as características e as colunas os trabalhos relacionados.

Quadro 1 - Comparativo dos trabalhos correlatos

| Trabalhos Correlatos / Características | Jira (ATLASSIAN, 2019) | Qtest (TRICENTIS, 2019) | TestLink (REIS, 2022) |
|---|---|---|---|
| Gestão de times | ✓ | ✓ | ✓ |
| Controle de Sprints | ✓ | X | X |
| Fluxo de atividades | ✓ | X | ✓ |
| Gestão de casos de teste | ✓ | ✓ | ✓ |
| Histórico de atividades | ✓ | ✓ | ✓ |
| Estatísticas | ✓ | ✓ | ✓ |
| Permissões de usuário | ✓ | ✓ | ✓ |
| Plataforma | Web, iOS e Android | Web | Web |
| Linguagem/Banco de dados | Node.JS, Vue.js e Java | Não informada | PHP/Postgres e MySQL |

Fonte: elaborado pelo autor.

Ao analisar o Quadro 1 é possível constatar que as soluções de Atlassian (2019), Tricentis (2019) e Reis (2018) possibilitam a gestão de times. Essa característica é importante porque permite que o gestor controle as atividades que cada membro do time está executando, assim como que ele designe tarefas para a equipe. Somente ~~a solução~~ a solução da Atlassian (2019) permite a criação de sprints com as atividades que a equipe vai executar, possibilitando que o gestor organize as atividades que vão ser executadas por meio de um quadro Kankan e permitindo que ele saiba em etapa está cada tarefa e quem a está fazendo.

Atlassian (2019) e Reis (2018) possuem a característica de criação de um fluxo de atividades, possibilitando que se crie um passo a passo para a execução das tarefas de cada usuário. Já a característica de gestão de casos de testes está presente em Atlassian (2022), Tricentis (2019) e Reis (2018). A gestão de casos de teste permite a documentação de todos os cenários, casos e ambientes de teste que a equipe utiliza durante o seu trabalho.

Atlassian (2019), Tricentis (2019) e Reis (2018) possuem tanto um histórico de atividades como estatísticas. Esse histórico permite que se documente cada alteração feita nas tarefas e documentos de cada projeto da equipe. Já a geração de estatísticas possibilita o controle do desempenho da equipe e fornece um panorama do que foi testado e o que precisa ser abordado. Atlassian (2019), Tricentis (2019) e Reis (2018) ainda se destacam por possuírem permissão de usuários, possibilitando o controle de quais projetos, funções e arquivos os usuários poderão acessar.

O trabalho aqui proposto tem paridade com os três trabalhos corretos. Ele busca facilitar a gestão dos testes, realizar o controle dos bugs e a cobertura de testes. Além disso, ele visa criar uma gestão da equipe de testes mais prática ao gestor, possibilitando que ele tenha um controle de quais testes estão sendo executados e o que cada membro da equipe está fazendo.

O sistema também vai possibilitar que o gestor possa gerar uma serie de relatórios para a medição do desempenho da equipe, ver quais pontos precisam ser melhorados, vai poder medir a cobertura que os testes estão alcançando e em quais rotinas ocorrem mais bugs.

Com base nessas características e tal como apresentadas no Quadro 1, é perceptível que o trabalho possuí relevância para a sociedade. O sistema traz valor ao contribuir com a qualidade de software nas empresas, incentivando a criação de uma gestão focada na área de testes, melhorando a qualidade do produto da empresa, o desempenho das equipes de testes e sendo possível acompanhá-lo por meio de gráficos e relatórios. Como contribuição acadêmica, a proposta traz o uso, do método Relationship of M3C with User Requirements and Usability and Communicability Assessment in groupware (RURUCAg) na área da qualidade de software, que poderá ser utilizada em outros trabalhos. Já como contribuição tecnológica, destaca-se o desenvolvimento de um sistema Web, no qual no *front-end* será utilizado a biblioteca JavaScript Angular e o *back-end* será implementado com python juntamente com banco de dados PostgreSQL. Além disso, traz a contribuição da solução proposta ser disponibilizada a partir de recursos de computação em nuvem.

## 3.2 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Nessa subseção serão descritos os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF), conforme o Quadro 2.

Quadro 2 - Principais Requisitos Funcionais e Não Funcionais

| **O sistema deve:** | **Tipo** |
|---|---|
| permitir ao usuário manter usuários dos tipos líderes de testes e testador (Create, Read, Update and delete - CRUD) | RF |
| permitir ao usuário manter o cadastro dos casos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de planos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de projetos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de uma Sprint (CRUD) | RF |
| permitir ao usuário sinalizar o final da execução de um plano, sprint ou projeto de testes | RF |
| permitir ao usuário marcar um caso de teste como concluído, bloqueado ou com falha | RF |
| permitir ao usuário visualizar um relatório com o histórico das atividades executadas | RF |
| permitir ao usuário gerar um relatório com as métricas dos testes | RF |
| permitir ao usuário receber notificação por meio de uma caixa de notificações | RF |
| ser construído usando a biblioteca JavaScript Angular juntamente com a linguagem TypeScript | RNF |
| ser responsiva | RNF |
| ser construído usando a linguagem Python para a Application Interface Program (API) | RNF |
| utilizar JavaScript Object Notation (JSON) escritos como API | RNF |
| utilizar o método RURUCAg para modelar a relação dos requisitos com as heurísticas de Nielsen | RNF |
| utilizar o método RURUCAg para avaliar a usabilidade e a experiência de uso | RNF |
| ser construído utilizando o banco de dados PostgreSQL | RNF |
| ser disponibilizado na nuvem | RNF |
| utilizar os métodos ágeis como o Kanban e o Scrum no desenvolvimento | RNF |

Fonte: elaborado pelo autor.

## 3.3 METODOLOGIA

A proposta dessa metodologia será constituída pelos seguintes instrumentos metodológicos e será desenvolvido nas etapas relacionadas ao Quadro 3:

a) aprofundamento bibliográfico: realizar estudos mais aprofundados na qualidade de software, testes de software e gerenciamento de equipes com métodos ágeis como Scrum e Kanban;
b) levantamento dos requisitos: analisar os requisitos funcionais e não-funcionais já definidos e, se necessário, especificar outros a partir da etapa do aprofundamento realizado;
c) especificação e análise: formalizar as funcionalidades do sistema por meio da construção de casos de uso e diagramas da Unified Modeling Language (UML), utilizando a ferramenta Lucidchart;
d) implementação: desenvolver o sistema utilizando a biblioteca JavaScript Angular juntamente com a linguagem TypeScript utilizando a IDE Visual Studio Code e desenvolver a API que será publicada utilizando recursos de computação em nuvem em conjunto com um banco de dados PostgreSQL;
e) verificação, validação e análise: realizar os testes do sistema e validar com o usuário a usabilidade e a experiência do usuário das interfaces desenvolvidas, assim como de suas funcionalidades pelo método RURUCAg.

Quadro 3 - Cronograma

| Quinzenas / Etapas | 2023 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | fev. | | mar. | | abr. | | maio | | jun. | |
| | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| Aprofundamento bibliográfico | | | | | | | | | | |
| Levantamento dos requisitos | | | | | | | | | | |
| Especificação e análise | | | | | | | | | | |
| Implementação | | | | | | | | | | |
| Verificação, validação e análise | | | | | | | | | | |

Fonte: elaborado pelo autor.

## 4 REVISÃO BIBLIOGRÁFICA

Nesta seção serão descritos os conceitos de maior relevância para o trabalho, sendo eles: qualidade de software; testes de software e gerenciamento de equipes com métodos ágeis como o Scrum e o Kanban.

Segundo a ABNT (2015, p. 6), "[...] qualidade é o grau no qual um conjunto de características inerentes satisfaz aos requisitos.". A ABNT (2015) ainda coloca que quando um produto ou sistema atinge os requisitos solicitados pelos clientes, ele possuiu a qualidade desejada pelo usuário. Pressman e Maxim (2021) definem a qualidade de software como uma gestão de qualidade efetiva aplicada de modo a criar um produto útil que forneça valor mensurável para aqueles que produzem e para aqueles que utilizam. Sommerville (2019) ainda afirma que a qualidade de software tem três preocupações principais, sendo: do nível organizacional, do nível do projeto e do nível de planejamento No nível organizacional a

atenção é o estabelecimento de um *framework* de processos organizacionais e padrões que levem a softwares de alta qualidade. Já no nível do projeto a qualidade de software envolve a aplicação de processos específicos de qualidade, verificando se os processos planejados foram seguidos, e garantindo que as saídas de projeto estejam em conformidade com os padrões aplicáveis ao projeto. Por fim, no nível de planejamento a qualidade está preocupada com estabelecer um plano de qualidade e ele deve definir as metas de qualidade tanto para o projeto quanto para quais processos e padrões devem ser usados (SOMMERVILLE, 2019).

O teste de software é destinado a mostrar que um programa faz o que é proposto a fazer e para descobrir os seus defeitos antes dele ser utilizado (SOMMERVILLE, 2019). Segundo Sommerville (2019), ao se testar o software, os resultados do teste são verificados à procura de erros, anomalias ou informações sobre os mesmo referente aos atributos não funcionais do programa. Para Wazlawick (2019), um dos objetivos da área de testes é definir conjuntos finitos e exequíveis de testes, que mesmo não garantindo que o software esteja livre de defeitos, consigam localizar os mais prováveis, permitindo assim sua eliminação. Já segundo Pressman e Maxim (2021), os testes são um conjunto de atividades que podem ser planejadas com antecedência e executadas sistematicamente. Pressman e Maxim (2021) ainda afirmam que por causa destas atividades deverá ser definido, para o processo de software, um modelo para o teste e um conjunto de etapas no qual podem ser empregadas técnicas específicas de projeto de caso de teste e métodos de teste.

Devatz e Polido (2017 *apud* AYRES, 2009) mencionam que um dos desafios da gestão de equipes é quando determinados grupos de trabalhadores organizados em equipes agem mais parecendo “eu-equipes”, deixando prevalecer os interesses pessoais quando comparados aos interesses do grupo. Ao aplicar os métodos ágeis, como Scrum e o Kanban, no gerenciamento é possível amenizar esses problemas e aumentar a eficiência da equipe (DEVATZ; POLIDO, 2017). Segundo Rubin (2017), o Scrum é um método ágil para o desenvolvimento de serviços e produtos inovadores. O método começa com a criação de um *backlog* do produto, no qual primeiro são feitos os itens de maior importância ou de maior prioridade e quando o tempo ou os recursos acabarem qualquer trabalho que não foi terminado será de menor preferência que o trabalho concluído (RUBIN, 2017). Já o Kanban segundo Silva e Anastácio (2019), é um método que auxilia no gerenciamento das atividades, assegurando a sua produção em uma base horária ou diária. Para Wazlawick (2019), Kanban é um método de gerenciamento mais fácil que o Scrum, pois utiliza o quadro Kanban para representar as tarefas ao invés de papéis e reuniões. O método também propõe que as atividades sejam priorizadas, sendo que as mais importantes devem ocupar o topo das colunas. Além disso, pode-se trabalhar com cartões

coloridos para indicar a prioridade, sendo a cor vermelho, alta; amarelo, média; e verde, baixa (WAZLAWICK, 2019).

## REFERÊNCIAS


ABNT- Associação Brasileira de Normas Técnicas. **NBR ISO 9000/2015 - Sistema de Gestão da Qualidade – Requisitos**, ABNT, 2015. Disponível em: https://files.comunidades.net/lodineimarchini/iso2015_versao_completa.pdf . Acesso em: 27 set. 2022.

ATLASSIAN. **Página inicial**. Jira, 2019. Disponível em: https://www.atlassian.com/software/jira. Acesso em: 27 set. 2022.

BAUMGARTNER, Cristiano. **Descubra a importância e os benefícios que a qualidade de software pode gerar para a sua empresa**, Testing Company, 2021. Disponível em: https://testingcompany.com.br**.** Acesso em: 24 set. 2022.

DEVAZT, Weverton Isaque; POLIDO, Ariela. Gestão de equipes: Uma ferramenta impulsionadora de resultados na logística empresarial contemporânea. **Jornacitec Botucatu**, Botucatu, v. 6, n. 6, 2017.

FRAGA, Bárbara; BARBOSA, Marcelo. A Engenharia de Requisitos nos métodos ágeis: uma revisão sistemática da literatura. *In*: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAÇÃO (SBSI), 13., 2017, Lavras. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2017. p. 309-315.

ITERIS, Software. **Gestão da qualidade do software**. Iteris, 2022. Disponível em: https://www.iteris.com.br/pt-br/o-que-fazemos/estrategia/gestao-qualidade-de-software. Acesso em: 24 set. 2022.

PRESSMAN, Roger S; MAXIM, Bruce R. **Engenharia de Software:** Uma abordagem profissional. 9. ed. AMGH Editora: Porta Alegre, 2021.

PEREIRA, João Vitor dos Santos; VARGAS, Alessandra Alves Fonseca. **Método testes de software:** uma ampla visão conceitual. Revista pesquisa & educação a distância, rio de janeiro, v. 1 n. 19, 2020.

REIS, Luana. **Testlink:** uma ferramenta de gerenciamento de testes de software, Medium, 2018. Disponível em: https://medium.com/@luanasreis/testlink-uma-ferramenta-de-gerenciamento-de-testes-de-software-44001b816f64. Acesso em: 27 set. 2022.

RUBIN, K. S. **Scrum Essencial**: Um guia prático para o mais popular processo ágil. Rio de Janeiro: Alta Books Editora, 2017.

SALVIANO, Clenio Figueiredo. **Qualidade de Software**. 1. ed. Editora Senac: São Paulo, 2020.

SANTOS, Luiz Diego Vidal; OLIVEIRA, Catuxe Vargão de Santana. **Introdução à garantia de qualidade de software**. 1. ed. Editora: Cia do Ebook, Timburi, 2017.

SILVA, Jessica Belém da; ANASTÁCIO, Francisca Alexandra de Macedo. Método Kanban como Ferramenta de Controle de Gestão. **Revista Multidisciplinar e de Psicologia**, Jaboatão dos Guararapes, V. 13, N. 43, p1018-1027, 2019.

SOMMERVILLE, I. **Software Engineering**. 10. ed. Editora Pearson: Boston, 2019.

TRICENTIS. **Página inicial**. qTest, 2019. Disponível em: https://www.tricentis.com/products/unified-test-management-qtest. Acesso em: 29 set. 2022.

WAZLAWICK, Raul Sidnei. **Engenharia de Software -** Conceitos e Práticas. 2. ed. Editora Elsevier: Rio de Janeiro, 2019.

XAVIER, Alan *et al*. Aplicação da UML no contexto das metodologias ágeis. *In*: ENCONTRO NACIONAL DE COMPUTAÇÃO DOS INSTITUTOS FEDERAIS (ENCOMPIF), 6., 2019, Belém. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2019.

# FORMULÁRIO DE AVALIAÇÃO SIS ACADÊMICO
# PROFESSOR AVALIADOR – PRÉ-PROJETO

Avaliador(a): Marcel Hugo

Atenção: quando o avaliador marcar algum item como atende parcialmente ou não atende, deve obrigatoriamente indicar os motivos no texto, para que o aluno saiba o porquê da avaliação.

| | ASPECTOS AVALIADOS | atende | atende parcialmente | não atende |
|---|---|---|---|---|
| ASPECTOS TÉCNICOS | 1. INTRODUÇÃO<br>O tema de pesquisa está devidamente contextualizado/delimitado? | | X | |
| | O problema está claramente formulado? | | X | |
| | 2. OBJETIVOS<br>O objetivo principal está claramente definido e é passível de ser alcançado? | | X | |
| | Os objetivos específicos são coerentes com o objetivo principal? | | X | |
| | 3. TRABALHOS CORRELATOS<br>São apresentados trabalhos correlatos, bem como descritas as principais funcionalidades e os pontos fortes e fracos? | | X | |
| | 4. JUSTIFICATIVA<br>Foi apresentado e discutido um quadro relacionando os trabalhos correlatos e suas principais funcionalidades com a proposta apresentada? | | X | |
| | São apresentados argumentos científicos, técnicos ou metodológicos que justificam a proposta? | | X | |
| | São apresentadas as contribuições teóricas, práticas ou sociais que justificam a proposta? | | X | |
| | 5. REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO<br>Os requisitos funcionais e não funcionais foram claramente descritos? | X | | |
| | 6. METODOLOGIA<br>Foram relacionadas todas as etapas necessárias para o desenvolvimento do TCC? | X | | |
| | Os métodos, recursos e o cronograma estão devidamente apresentados e são compatíveis com a metodologia proposta? | X | | |
| | 7. REVISÃO BIBLIOGRÁFICA<br>Os assuntos apresentados são suficientes e têm relação com o tema do TCC? | | X | |
| | As referências contemplam adequadamente os assuntos abordados (são indicadas obras atualizadas e as mais importantes da área)? | X | | |
| ASPECTOS METODOLÓGICOS | 8. LINGUAGEM USADA (redação)<br>O texto completo é coerente e redigido corretamente em língua portuguesa, usando linguagem formal/científica? | X | | |
| | A exposição do assunto é ordenada (as ideias estão bem encadeadas e a linguagem utilizada é clara)? | | X | |

| CURSO DE SISTEMAS DE INFORMAÇÃO – TCC ACADÊMICO | |
|---|---|
| ( x) PRÉ-PROJETO ( ) PROJETO | ANO/SEMESTRE: 2/2022 |

**TEAM MANAGEMENT: SISTEMA PARA GESTÃO DE EQUIPES DE QUALIDADE DE SOFTWARE**

Aluno: João Manoel Sanches Lima

Prof. Simone Erbs da Costa – Orientadora

## 1 INTRODUÇÃO

A busca por melhor qualidade tem orientado trabalhos e pesquisas desde sempre. Para Salviano (2020), desde que o ser humano começou a desenvolver e utilizar ferramentas, o conhecimento sobre como fazer, e, ainda melhor aquilo que é feito, foi aumentando. Esse conceito de "fazer melhor" está fortemente relacionado com a qualidade, independente da definição específica utilizada (SALVIANO, 2020). Santos e Oliveira (2017) observam que a qualidade de software é um ponto de suma importância no desenvolvimento de um projeto, ou seja, uma tarefa gerencial de natureza continua que requer execução durante o ciclo de vida do software.

Wazlawick (2019) coloca que a qualidade de software está relacionada a disciplina de teste. A disciplina de teste passou a ser considerada extremamente importante, fazendo parte do processo de desenvolvimento de software (WAZLAWICK, 2019). Os testes de software também foram incorporados nos métodos ágeis como uma atividade crítica, assumindo inclusive que os casos de teste deveriam passar a ser escritos antes das unidades de software que pretendem testar (PRESSMAN; MAXIM, 2021). A finalidade desse processo de testes de software é proporcionar informações que garantam a qualidade do produto e dos processos de teste (PEREIRA; VARGAS, 2019). Dessa forma, Xavier *et al.* (2019 *apud* SOMMERVILLE, 2011) colocam que a realidade no mercado de software propõe inúmeras alterações tanto na forma de utilização quanto na forma de desenvolver ou adquirir um software com a devida qualidade.

Para Fraga e Barbosa (2017), os métodos ágeis vêm ganhando mais espaço nesse mercado por serem práticas mais focadas na adaptação dos processos de uma forma incremental e iterativa. Isso possibilita aumentar a produção da equipe e melhorar a qualidade, bem como evitar erros com entregas mais contínuas (FRAGA; BARBOSA, 2017). Devatz e Polido (2017 *apud* BINATO, 2017) afirmam que um dos maiores desafios do mundo corporativo é o trabalho em equipe. Diante desse cenário, este trabalho propõe o desenvolvimento de um sistema para auxiliar no gerenciamento de times de teste. Conjectura-se que a construção deste sistema auxilie o gestor a aprimorar a eficiência da equipe, gerando resultados mais eficientes e aumentando o desempenho da equipe.

## 1.1 OBJETIVOS

O Objetivo geral deste trabalho proposto é desenvolver um sistema para auxiliar na gestão dos testes nas equipes de qualidade de software, auxiliando no desempenho da equipe e buscando mais eficiência nos resultados. Os objetivos específicos são:

a) gerenciar e criar planos de testes que devem ser executados pela equipe por meio de interfaces disponibilizadas;
b) controlar o trabalho da equipe de testes utilizando os métodos ágeis como o Scrum e o Kanban no desenvolvimento das funcionalidades;
c) analisar e avaliar as funcionalidades e a usabilidade e a experiência do usuário das interfaces desenvolvidas, por meio do Método Relationship of M3C with User Requirements and Usability and Communicability Assessment in groupware (RURUCAg).

# 2 TRABALHOS CORRELATOS

Nesta seção serão descritos os trabalhos correlatos que possuem semelhanças com o trabalho proposto. A subseção 0 traz o sistema Jira focado em gestão de trabalho que utiliza soluções ágeis como Kankan (ATLASSIAN, 2019). A subseção 2.2 apresenta o sistema Qtest que objetiva o gerenciamento e a análise de teste (TRICENTIS, 2019). Por fim, a seção 2.3 descreve o sistema de gerenciamento de teste TestLink que tem como foco principal o gerenciamento de casos testes de software (REIS, 2018).

## 2.1 JIRA SOFTWARE: GESTÃO DE PROJETOS E MONITORAMENTO DE TAREFAS

Excluído: .

A Atlassian (2019) traz o Jira que faz parte de um conjunto de soluções ágeis, proporcionado que seja realizado o gerenciamento e potencializado o trabalho de equipes no desenvolvimento ágil de sistemas. O Jira é amplamente utilizado dentro das empresas para o gerenciamento de equipes, criação e controle de sprint. Ele pode ser utilizado por desenvolvedores, analistas de qualidade, gerentes de projeto, scrum master, analistas de negócio e designers de produto/User eXperience (UX). O Jira foi desenvolvido usando as linguagens Node.JS, Java, Vue.js e está disponível para as plataformas Web, Android e iOS. Algumas das funções do Jira são: gestão de times, controle de sprints, fluxo de atividades, gestão de casos de teste, histórico de atividades, estatísticas e permissões de usuário (ATLASSIAN, 2019).

O Jira permite a criação de uma listagem de demandas a serem feitas pela equipe na visão do gestor. Nessa listagem é possível ver os tipos de demandas que foram abertas, quem criou a demanda, o responsável por ela no momento, o seus status e o tempo estimado para sua

execução. Essa listagem permite que o usuário tenha uma visão geral do que acontece com o projeto, mostrando as principais informações sobre as demandas além de permitir a criação, exclusão e alteração delas. Outra função que ela permite é o acesso as estatísticas da equipe e a base de informação do projeto (ATLASSIAN, 2019).

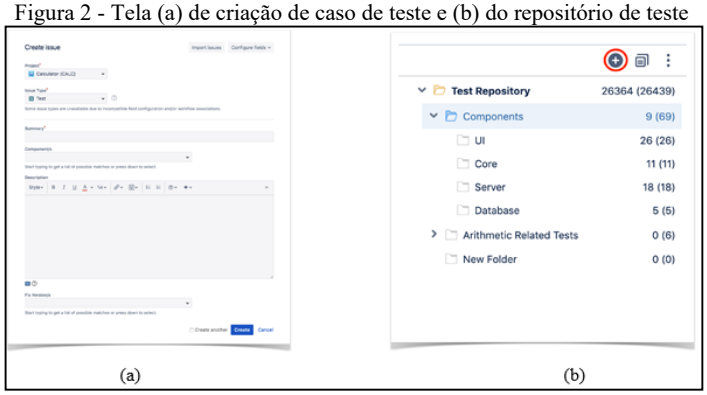
A Figura 1 traz a tela do quadro Kankan. Nessa tela é possível visualizar as demandas separadas em colunas e por status, criando assim o fluxo das atividades. Esse quadro permite ao usuário movimentar os cartões (demandas) para qualquer status que seja necessário, assim como incluir ou excluir as etapas no fluxo de atividades da sprint. Pela Figura 1 também é possível ver as informações das demandas nos cartões, em cada uma delas é mostrado o `título`, o `tipo de demanda`, a sua `prioridade` e o `responsável` por ela. Na parte superior (de cima para baixo, destacado pela letra A) consta os controles para gerenciar a sprint. Ela permite que o usuário controle o tempo dessa sprint, quando ela vai ser terminada usando o botão `Complete sprint` e as suas configurações (ATLASSIAN, 2019).

Figura 1 - Tela do Quadro Kanban


Fonte: Atlassian (2019).

A Figura 2 (a) traz a tela para criar o caso de teste. Nela o usuário pode inserir o projeto a qual o caso de teste pertence, a sua descrição, a versão em qual o caso foi ou deve ser executado e os seus componentes. Esta tela permite ao usuário adicionar ou excluir os campos do formulário usado na criação dos casos de teste e fazer o link com outro caso de teste ou demanda já criada. Já Figura 2 (b) mostra o repositório de testes no qual é possível ver todos os testes criados separados por pastas e o gerenciamento deles. O usuário também pode fazer alterações nos casos de testes e ao clicar no botão + ele pode adicionar novas pastas ao repositório (ATLASSIAN, 2019).

Figura 2 - Tela (a) de criação de caso de teste e (b) do repositório de teste




Fonte: Atlassian (2019).

Em cada demanda criada no Jira é gerado um histórico de atividades que salva todas as alterações feitas nela. Na demanda criada tem duas áreas para o registro de atividades. A primeira é para o registro de tempo gasto na atividade, nela o usuário registra o que ele fez e quanto tempo ele gastou na demanda. A segunda área é para o registro das alterações feitas na demanda durante o seu trâmite, com a inserção de um anexo, mudança de status, responsável pela *issue* e a data e hora da alteração (ATLASSIAN, 2019).

Formatado: Fonte: Itálico

O Jira fornece uma série de gráficos que auxiliam na gestão e no monitoramento do time. Com os relatórios de análise de demandadas a equipe ou o gestor consegue saber quanto tempo é gasto na resolução das demandas, quantas demandas foram abertas ou resolvidas em um determinado período e é possível realizar a comparação dessas informações. Ele também possui relatórios e gráficos para o controle do tempo e gestão da equipe, assim como gera relatórios para controle das sprints para o gestor (ATLASSIAN, 2019).

## 2.2 QTEST: UMA FERRAMENTA PARA GERENCIAMENTO E ANÁLISE DE TESTE

Excluído: .

A Tricentis (2019) traz o Qtest, que é um sistema Web para o gerenciamento e análise de casos de teste nas equipes de qualidade de software. O sistema permite que os gestores façam um gerenciamento completo dos testes, podendo criar um projeto de testes com todos os planos e casos de teste da equipe. Além disso o sistema possibilita que se faça uma análise completa dos testes executados. As funções mais destacadas do Qtest são: gestão de time, gestão de casos de teste com a criação de planos e repositórios de teste, historio de atividades, estatísticas por meio de geração de relatórios e permissões de usuário (TRICENTIS, 2019).

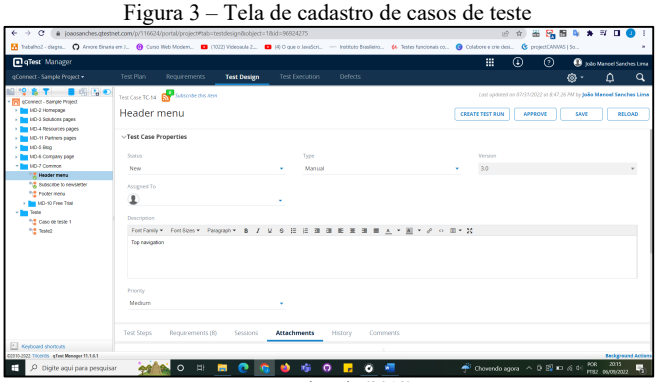
O Qtest permite ao usuário gerenciar a equipe de qualidade, captando dados e estatísticas durante a execução dos testes. O sistema possibilita a criação de um projeto de testes que pode ser composto por planos de teste, *suítes* de casos testes, versionamento de versões e listagem

Formatado: Fonte: Itálico

de requisitos. O sistema também possui um controle sobre os usuários, permitindo que o gestor adicione contas de usuários e o seu nível de permissão (TRICENTIS, 2019).

A Tricentis (2019) traz um gerenciamento de casos de teste, permitindo que o usuário tenha controle sobre os casos que foram criados. Nesse sentido, a Figura 3 traz a tela para a criação do caso de teste, na parte de cima (de cima para baixo) é possível fazer a inserção das informações básicas como o título, responsável, tipo de teste e a versão. Já na parte inferior da referida figura consta as informações complementares como os passos do teste para a sua execução, os anexos, os requisitos necessários do teste, o seu histórico e os comentários relacionados a ele. O Qtest permite ao usuário organizar os casos de teste por meio de um repositório, separando-os por projetos e pastas (TRICENTIS, 2019).

Figura 3 – Tela de cadastro de casos de teste




Fonte: Tricentis (2019).

Segundo Tricentis (2019), o sistema possui um histórico de versões para cada função gravando todos os dados que foram alterados e os listando posteriormente. Em cada aba o sistema guarda as alterações que foram feitas por meio de um histórico detalhado e os exibe por meio de uma lista. Tricentis (2019) ainda coloca que o sistema possui um fácil acesso aos históricos, no qual em cada função existe uma aba com o histórico e os usuários conseguem acessá-lo livremente podendo ver quem fez a alteração, quando e o que foi alterado.

De acordo com Tricentis (2019), o Qtest traz um painel com menus e métricas do sistema, permitindo que o usuário tenha um controle sobre o andamento do projeto. Nesse sentido, o sistema traz um menu com os dados dos testes executados, mostrando a velocidade em que os casos foram executados, a cobertura que eles atingiram e qualidade dos testes. Além disso, o sistema possui um *dashboard* que mostra em tempo real quantos casos foram executados, bloqueados, falharam e ele também permite a geração de relatórios sobre eles. Ele também exibe gráficos, deixando mais visual a demonstração dos dados e possibilitando que o

**Formatado:** Fonte: Itálico

usuário altere os gráficos por meio de filtros e exibe as informações de um determinado período ou versão do sistema (TRICENTIS, 2019).

Cabe destacar, que o Qtest possibilita ao gestor adicionar usuário, gerando um *login* e senha e colocando um nível de permissão de acesso ao sistema. O gestor pode administrar todos os usuários, assim como pode os separar em grupos de projetos e de permissão. Além disso, o sistema possibilita que o gestor designe tarefas para o usuário o colocando como responsável por ela. A ferramenta também permite que o gestor possa rastrear as atividades que um usuário executou durante os testes, assim como ver quais casos foram designados ou executados por um determinado usuário (TRICENTIS, 2019).

## 2.3 TESTLINK: UMA FERRAMENTA DE GERENCIAMENTO DE TESTES DE SOFTWARE

De acordo com Reis (2018), o TestLink é um sistema open source para o gerenciamento de testes. Ele permite que as equipes de testes trabalhem de forma sincronizada no mesmo espaço de trabalho seja presencialmente ou remotamente. O Test Link foi disponibilizado na plataforma Web e tanto o seu *back-end* como o *front-end* foram desenvolvidos utilizando a linguagem Hypertext PreProcessor (PHP). Ele também possui uma integração com os bancos PostgreSQL e MySQL, permitindo que os dados do projeto de teste sejam salvos externamente. Algumas das funções do TestLink são: gestão de times, a criação de fluxo de atividades, gestão de casos de teste, histórico de atividades, estatísticas e permissões de usuários (REIS, 2018).

Segundo Reis (2018), o TestLink permite que o usuário faça a gestão do seu time adicionando colaboradores nos projetos de teste. O usuário administrador pode gerenciar o desempenho de cada membro do time e controlar cada conta que foi adicionada dando um nível de permissão. Além disso, o sistema permite que cada usuário seja colocado em um projeto de teste específico. Já sobre o fluxo de atividades, Reis (2018) coloca que ele pode ser configurado pelo usuário. Nele é possível cadastrar um projeto de teste, uma *baseline* de teste, um plano de teste, uma *suíte* de teste e casos de teste. Nos fluxos de execução de um plano ou caso de teste a três ações podem ser tomadas: quando o teste é executado com sucesso o caso é aprovado; caso o teste não possa ser feito o caso é bloqueado e o motivo do bloqueio do caso de teste é descrito; quando é encontrado um erro ou uma falha no teste o caso é reprovado e o problema é descrito no caso de teste para documentação e análise do problema (REIS, 2018).

Formatado: Fonte: Itálico

Formatado: Fonte: Itálico

Referente a gestão de casos de teste, o Test Link permite ao usuário criar um projeto de testes específico para cada produto da empresa, dentro do projeto é permitido o cadastro da plataforma e versão do software em que os testes serão executados. Ele também possibilita a criação de planos de testes para cada plataforma que precisa ser testada, para a execução de

testes regressivos em versões do sistema e permite a organização dos casos de teste em *suítes* os separando por função a ser testada (REIS, 2018).

Reis (2018) destaca um histórico de versões e execuções em cada caso, *suíte*, plano ou projeto de teste. A cada modificação do caso de teste o sistema grava quem o criou e quem o alterou por último e ele também possibilita a criação de várias versões do mesmo caso de teste. Ao executar o caso de teste o sistema guarda a informações de cada caso, qual foi o status dado a ele, a data, hora e o usuário que executou o caso de teste. Além disso, segundo Reis (2018), o sistema possui um painel de métricas que gera uma série de informações sobre os projetos de teste que foram criados e a forma que são executados. Durante a execução de um plano de teste o Test Link consegue gerar um relatório com o número de casos de teste que foram executados no dia e no total, quais foram aprovados, reprovados e bloqueados. Dessa forma, o sistema consegue gerar gráficos prevento o tempo de execução dos planos de teste se baseando no número de casos de testes executados por dia (REIS, 2018).

Reis (2018) ainda destaca que o controle de permissão e de usuários possibilita um trabalho em equipe sincronizado e de vários lugares. Nesse sentido, o sistema permite a adição de usuários a um projeto com diferentes níveis de acesso como administrador, líder de testes, projetista de teste, testador e convidado. Na criação de um plano de teste o sistema tem a opção que se faça a designação de um testador para casos de teste específicos (REIS, 2018).

## 3 PROPOSTA DO SISTEMA

Nesta seção serão descritas as justificativas para o desenvolvimento do trabalho proposto na subseção **Erro! Fonte de referência não encontrada.**; os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF) na subseção **Erro! Fonte de referência não encontrada.**; e para finalizar será descrito as metodologias e planejamento do cronograma para o desenvolvimento desta proposta na subseção 3.3.

### 3.1 JUSTIFICATIVA

A relevância do projeto proposto foi evidenciada nas seções 1 e 2. Segundo Iteris (2022), a gestão da qualidade de sistema é indispensável para qualquer empresa que deseja garantir a entrega do produto certo de forma correta. Pois, por meio dela a empresa consegue otimizar o desenvolvimento, garantindo um produto funcional e que agregue valor ao negócio (ITERIS, 2022). Já para Baumgartner (2021), o controle da qualidade de sistema é um conjunto de atividades técnicas que visam monitorar o processo de desenvolvimento, fazendo com que teste possua funcionalidades e características que atendam as expectativas de todos os envolvidos no projeto. Nesse contexto, Atlassian (2019), Tricentis (2019) e Reis (2018) identificaram a possibilidade de criar um sistema para a gestão de qualidade de sistema, possibilitando tanto

Formatado: Fonte: Itálico

Formatado: Fonte: Itálico

Excluído: 2

que o gerenciamento dos testes como da equipe de qualidade. No Quadro 1 é mostrado um comparativo dos trabalhos correlatos descritos na seção 2, as linhas representam as características e as colunas os trabalhos relacionados.

Quadro 1 - Comparativo dos trabalhos correlatos

| Trabalhos Correlatos / Características | Jira (ATLASSIAN, 2019) | Qtest (TRICENTIS, 2019) | TestLink (REIS, 2022) |
|---|---|---|---|
| Gestão de times | ✓ | ✓ | ✓ |
| Controle de Sprints | ✓ | X | X |
| Fluxo de atividades | ✓ | X | ✓ |
| Gestão de casos de teste | ✓ | ✓ | ✓ |
| Histórico de atividades | ✓ | ✓ | ✓ |
| Estatísticas | ✓ | ✓ | ✓ |
| Permissões de usuário | ✓ | ✓ | ✓ |
| Plataforma | Web, iOS e Android | Web | Web |
| Linguagem/Banco de dados | Node.JS, Vue.js e Java | Não informada | PHP/Postgres e MySQL |

Fonte: elaborado pelo autor.

Ao analisar o Quadro 1 é possível constatar que as soluções de Atlassian (2019), Tricentis (2019) e Reis (2018) possibilitam a gestão de times. Essa característica é importante porque permite que o gestor controle as atividades que cada membro do time está executando, assim como que ele designe tarefas para a equipe. Somente a solução da Atlassian (2019) permite a criação de *sprints* com as atividades que a equipe vai executar, possibilitando que o gestor organize as atividades que vão ser executadas por meio de um quadro Kankan e permitindo que ele saiba em etapa está cada tarefa e quem a está fazendo.

Excluído: a solução

Formatado: Fonte: Itálico

Atlassian (2019) e Reis (2018) possuem a característica de criação de um fluxo de atividades, possibilitando que se crie um passo a passo para a execução das tarefas de cada usuário. Já a característica de gestão de casos de testes está presente em Atlassian (2022), Tricentis (2019) e Reis (2018). A gestão de casos de teste permite a documentação de todos os cenários, casos e ambientes de teste que a equipe utiliza durante o seu trabalho.

Atlassian (2019), Tricentis (2019) e Reis (2018) possuem tanto um histórico de atividades como estatísticas. Esse histórico permite que se documente cada alteração feita nas tarefas e documentos de cada projeto da equipe. Já a geração de estatísticas possibilita o controle do desempenho da equipe e fornece um panorama do que foi testado e o que precisa ser abordado. Atlassian (2019), Tricentis (2019) e Reis (2018) ainda se destacam por possuírem permissão de usuários, possibilitando o controle de quais projetos, funções e arquivos os usuários poderão acessar.

O trabalho aqui proposto tem paridade com os três trabalhos corretos. Ele busca facilitar a gestão dos testes, realizar o controle dos *bugs* e a cobertura de testes. Além disso, ele visa criar uma gestão da equipe de testes mais prática ao gestor, possibilitando que ele tenha um controle de quais testes estão sendo executados e o que cada membro da equipe está fazendo.

Formatado: Fonte: Itálico

O sistema também vai possibilitar que o gestor possa gerar uma serie de relatórios para a medição do desempenho da equipe, ver quais pontos precisam ser melhorados, vai poder medir a cobertura que os testes estão alcançando e em quais rotinas ocorrem mais *bugs*.

Formatado: Fonte: Itálico

Com base nessas características e tal como apresentadas no Quadro 1, é perceptível que o trabalho possuí relevância para a sociedade. O sistema traz valor ao contribuir com a qualidade de software nas empresas, incentivando a criação de uma gestão focada na área de testes, melhorando a qualidade do produto da empresa, o desempenho das equipes de testes e sendo possível acompanhá-lo por meio de gráficos e relatórios. Como contribuição acadêmica, a proposta traz o uso, do método Relationship of M3C with User Requirements and Usability and Communicability Assessment in groupware (RURUCAg) na área da qualidade de software, que poderá ser utilizada em outros trabalhos. Já como contribuição tecnológica, destaca-se o desenvolvimento de um sistema Web, no qual no *front-end* será utilizado a biblioteca JavaScript Angular e o *back-end* será implementado com Python juntamente com banco de dados PostgreSQL. Além disso, traz a contribuição da solução proposta ser disponibilizada a partir de recursos de computação em nuvem.

Excluído: p

## 3.2 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Nessa subseção serão descritos os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF), conforme o Quadro 2.

Quadro 2 - Principais Requisitos Funcionais e Não Funcionais

| **O sistema deve:** | **Tipo** |
|---|---|
| permitir ao usuário manter usuários dos tipos líderes de testes e testador (Create, Read, Update and delete - CRUD) | RF |
| permitir ao usuário manter o cadastro dos casos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de planos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de projetos de testes (CRUD) | RF |
| permitir ao usuário manter o cadastro de uma Sprint (CRUD) | RF |
| permitir ao usuário sinalizar o final da execução de um plano, sprint ou projeto de testes | RF |
| permitir ao usuário marcar um caso de teste como concluído, bloqueado ou com falha | RF |
| permitir ao usuário visualizar um relatório com o histórico das atividades executadas | RF |
| permitir ao usuário gerar um relatório com as métricas dos testes | RF |
| permitir ao usuário receber notificação por meio de uma caixa de notificações | RF |
| ser construído usando a biblioteca JavaScript Angular juntamente com a linguagem TypeScript | RNF |
| ser responsiva | RNF |
| ser construído usando a linguagem Python para a Application Interface Program (API) | RNF |
| utilizar JavaScript Object Notation (JSON) escritos como API | RNF |
| utilizar o método RURUCAg para modelar a relação dos requisitos com as heurísticas de Nielsen | RNF |
| utilizar o método RURUCAg para avaliar a usabilidade e a experiência de uso | RNF |
| ser construído utilizando o banco de dados PostgreSQL | RNF |
| ser disponibilizado na nuvem | RNF |
| utilizar os métodos ágeis como o Kanban e o Scrum no desenvolvimento | RNF |

Fonte: elaborado pelo autor.

## 3.3 METODOLOGIA

A proposta dessa metodologia será constituída pelos seguintes instrumentos metodológicos e será desenvolvido nas etapas relacionadas ao Quadro 3:

a) aprofundamento bibliográfico: realizar estudos mais aprofundados na qualidade de software, testes de software e gerenciamento de equipes com métodos ágeis como Scrum e Kanban;
b) levantamento dos requisitos: analisar os requisitos funcionais e não-funcionais já definidos e, se necessário, especificar outros a partir da etapa do aprofundamento realizado;
c) especificação e análise: formalizar as funcionalidades do sistema por meio da construção de casos de uso e diagramas da Unified Modeling Language (UML), utilizando a ferramenta Lucidchart;
d) implementação: desenvolver o sistema utilizando a biblioteca JavaScript Angular juntamente com a linguagem TypeScript utilizando a IDE Visual Studio Code e desenvolver a API que será publicada utilizando recursos de computação em nuvem em conjunto com um banco de dados PostgreSQL;
e) verificação, validação e análise: realizar os testes do sistema e validar com o usuário a usabilidade e a experiência do usuário das interfaces desenvolvidas, assim como de suas funcionalidades pelo método RURUCAg.

Quadro 3 - Cronograma

| Quinzenas / Etapas | 2023 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | fev. | | mar. | | abr. | | maio | | jun. | |
| | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| Aprofundamento bibliográfico | | | | | | | | | | |
| Levantamento dos requisitos | | | | | | | | | | |
| Especificação e análise | | | | | | | | | | |
| Implementação | | | | | | | | | | |
| Verificação, validação e análise | | | | | | | | | | |

Fonte: elaborado pelo autor.

## 4 REVISÃO BIBLIOGRÁFICA

Nesta seção serão descritos os conceitos de maior relevância para o trabalho, sendo eles: qualidade de software; testes de software e gerenciamento de equipes com métodos ágeis como o Scrum e o Kanban.

Segundo a ABNT (2015, p. 6), "[...] qualidade é o grau no qual um conjunto de características inerentes satisfaz aos requisitos.". A ABNT (2015) ainda coloca que quando um produto ou sistema atinge os requisitos solicitados pelos clientes, ele possuiu a qualidade desejada pelo usuário. Pressman e Maxim (2021) definem a qualidade de software como uma gestão de qualidade efetiva aplicada de modo a criar um produto útil que forneça valor mensurável para aqueles que produzem e para aqueles que utilizam. Sommerville (2019) ainda afirma que a qualidade de software tem três preocupações principais, sendo: do nível organizacional, do nível do projeto e do nível de planejamento No nível organizacional a

atenção é o estabelecimento de um *framework* de processos organizacionais e padrões que levem a softwares de alta qualidade. Já no nível do projeto a qualidade de software envolve a aplicação de processos específicos de qualidade, verificando se os processos planejados foram seguidos, e garantindo que as saídas de projeto estejam em conformidade com os padrões aplicáveis ao projeto. Por fim, no nível de planejamento a qualidade está preocupada com estabelecer um plano de qualidade e ele deve definir as metas de qualidade tanto para o projeto quanto para quais processos e padrões devem ser usados (SOMMERVILLE, 2019).

O teste de software é destinado a mostrar que um programa faz o que é proposto a fazer e para descobrir os seus defeitos antes dele ser utilizado (SOMMERVILLE, 2019). Segundo Sommerville (2019), ao se testar o software, os resultados do teste são verificados à procura de erros, anomalias ou informações sobre os mesmo referente aos atributos não funcionais do programa. Para Wazlawick (2019), um dos objetivos da área de testes é definir conjuntos finitos e exequíveis de testes, que mesmo não garantindo que o software esteja livre de defeitos, consigam localizar os mais prováveis, permitindo assim sua eliminação. Já segundo Pressman e Maxim (2021), os testes são um conjunto de atividades que podem ser planejadas com antecedência e executadas sistematicamente. Pressman e Maxim (2021) ainda afirmam que por causa destas atividades deverá ser definido, para o processo de software, um modelo para o teste e um conjunto de etapas no qual podem ser empregadas técnicas específicas de projeto de caso de teste e métodos de teste.

Devatz e Polido (2017 *apud* AYRES, 2009) mencionam que um dos desafios da gestão de equipes é quando determinados grupos de trabalhadores organizados em equipes agem mais parecendo “eu-equipes”, deixando prevalecer os interesses pessoais quando comparados aos interesses do grupo. Ao aplicar os métodos ágeis, como Scrum e o Kanban, no gerenciamento é possível amenizar esses problemas e aumentar a eficiência da equipe (DEVATZ; POLIDO, 2017). Segundo Rubin (2017), o Scrum é um método ágil para o desenvolvimento de serviços e produtos inovadores. O método começa com a criação de um *backlog* do produto, no qual primeiro são feitos os itens de maior importância ou de maior prioridade e quando o tempo ou os recursos acabarem qualquer trabalho que não foi terminado será de menor preferência que o trabalho concluído (RUBIN, 2017). Já o Kanban segundo Silva e Anastácio (2019), é um método que auxilia no gerenciamento das atividades, assegurando a sua produção em uma base horária ou diária. Para Wazlawick (2019), Kanban é um método de gerenciamento mais fácil que o Scrum, pois utiliza o quadro Kanban para representar as tarefas ao invés de papéis e reuniões. O método também propõe que as atividades sejam priorizadas, sendo que as mais importantes devem ocupar o topo das colunas. Além disso, pode-se trabalhar com cartões

coloridos para indicar a prioridade, sendo a cor vermelho, alta; amarelo, média; e verde, baixa (WAZLAWICK, 2019).

## REFERÊNCIAS


ABNT- Associação Brasileira de Normas Técnicas. **NBR ISO 9000/2015 - Sistema de Gestão da Qualidade – Requisitos**, ABNT, 2015. Disponível em: https://files.comunidades.net/lodineimarchini/iso2015_versao_completa.pdf . Acesso em: 27 set. 2022.

ATLASSIAN. **Página inicial**. Jira, 2019. Disponível em: https://www.atlassian.com/software/jira. Acesso em: 27 set. 2022.

BAUMGARTNER, Cristiano. **Descubra a importância e os benefícios que a qualidade de software pode gerar para a sua empresa**, Testing Company, 2021. Disponível em: https://testingcompany.com.br**.** Acesso em: 24 set. 2022.

DEVAZT, Weverton Isaque; POLIDO, Ariela. Gestão de equipes: Uma ferramenta impulsionadora de resultados na logística empresarial contemporânea. **Jornacitec Botucatu**, Botucatu, v. 6, n. 6, 2017.

FRAGA, Bárbara; BARBOSA, Marcelo. A Engenharia de Requisitos nos métodos ágeis: uma revisão sistemática da literatura. *In*: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAÇÃO (SBSI), 13., 2017, Lavras. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2017. p. 309-315.

ITERIS, Software. **Gestão da qualidade do software**. Iteris, 2022. Disponível em: https://www.iteris.com.br/pt-br/o-que-fazemos/estrategia/gestao-qualidade-de-software. Acesso em: 24 set. 2022.

PRESSMAN, Roger S; MAXIM, Bruce R. **Engenharia de Software:** Uma abordagem profissional. 9. ed. AMGH Editora: Porta Alegre, 2021.

PEREIRA, João Vitor dos Santos; VARGAS, Alessandra Alves Fonseca. **Método testes de software:** uma ampla visão conceitual. Revista pesquisa & educação a distância, rio de janeiro, v. 1 n. 19, 2020.

REIS, Luana. **Testlink:** uma ferramenta de gerenciamento de testes de software, Medium, 2018. Disponível em: https://medium.com/@luanasreis/testlink-uma-ferramenta-de-gerenciamento-de-testes-de-software-44001b816f64. Acesso em: 27 set. 2022.

RUBIN, K. S. **Scrum Essencial**: Um guia prático para o mais popular processo ágil. Rio de Janeiro: Alta Books Editora, 2017.

SALVIANO, Clenio Figueiredo. **Qualidade de Software**. 1. ed. Editora Senac: São Paulo, 2020.

SANTOS, Luiz Diego Vidal; OLIVEIRA, Catuxe Vargão de Santana. **Introdução à garantia de qualidade de software**. 1. ed. Editora: Cia do Ebook, Timburi, 2017.

SILVA, Jessica Belém da; ANASTÁCIO, Francisca Alexandra de Macedo. Método Kanban como Ferramenta de Controle de Gestão. **Revista Multidisciplinar e de Psicologia**, Jaboatão dos Guararapes, V. 13, N. 43, p1018-1027, 2019.

SOMMERVILLE, I. **Software Engineering**. 10. ed. Editora Pearson: Boston, 2019.

TRICENTIS. **Página inicial**. qTest, 2019. Disponível em: https://www.tricentis.com/products/unified-test-management-qtest. Acesso em: 29 set. 2022.

WAZLAWICK, Raul Sidnei. **Engenharia de Software -** Conceitos e Práticas. 2. ed. Editora Elsevier: Rio de Janeiro, 2019.


**Comentado [DSdR1]:** Referências bibliográficas: ordem alfabética

XAVIER, Alan *et al*. Aplicação da UML no contexto das metodologias ágeis. *In*: ENCONTRO NACIONAL DE COMPUTAÇÃO DOS INSTITUTOS FEDERAIS (ENCOMPIF), 6., 2019, Belém. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2019.

FORMULÁRIO DE AVALIAÇÃO SIS ACADÊMICO
PROFESSOR TCC I - PRÉ-PROJETO

Avaliador(a): Dalton Solano dos Reis

| | ASPECTOS AVALIADOS | atende | atende parcialmente | não atende |
|---|---|---|---|---|
| ASPECTOS TÉCNICOS | 1. INTRODUÇÃO<br>O tema de pesquisa está devidamente contextualizado/delimitado? | X | | |
| | O problema está claramente formulado? | X | | |
| | 2. OBJETIVOS<br>O objetivo principal está claramente definido e é passível de ser alcançado? | X | | |
| | Os objetivos específicos são coerentes com o objetivo principal? | X | | |
| | 3. TRABALHOS CORRELATOS<br>São apresentados trabalhos correlatos, bem como descritas as principais funcionalidades e os pontos fortes e fracos? | X | | |
| | 4. JUSTIFICATIVA<br>Foi apresentado e discutido um quadro relacionando os trabalhos correlatos e suas principais funcionalidades com a proposta apresentada? | X | | |
| | São apresentados argumentos científicos, técnicos ou metodológicos que justificam a proposta? | X | | |
| | São apresentadas as contribuições teóricas, práticas ou sociais que justificam a proposta? | X | | |
| | 5. REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO<br>Os requisitos funcionais e não funcionais foram claramente descritos? | X | | |
| | 6. METODOLOGIA<br>Foram relacionadas todas as etapas necessárias para o desenvolvimento do TCC? | X | | |
| | Os métodos, recursos e o cronograma estão devidamente apresentados e são compatíveis com a metodologia proposta? | X | | |
| | 7. REVISÃO BIBLIOGRÁFICA<br>Os assuntos apresentados são suficientes e têm relação com o tema do TCC? | X | | |
| | As referências contemplam adequadamente os assuntos abordados (são indicadas obras atualizadas e as mais importantes da área)? | X | | |
| ASPECTOS METODOLÓGICOS | 8. LINGUAGEM USADA (redação)<br>O texto completo é coerente e redigido corretamente em língua portuguesa, usando linguagem formal/científica? | X | | |
| | A exposição do assunto é ordenada (as ideias estão bem encadeadas e a linguagem utilizada é clara)? | X | | |
| | 9. ORGANIZAÇÃO E APRESENTAÇÃO GRÁFICA DO TEXTO<br>A organização e apresentação dos capítulos, seções, subseções e parágrafos estão de acordo com o modelo estabelecido? | X | | |
| | 10. ILUSTRAÇÕES (figuras, quadros, tabelas)<br>As ilustrações são legíveis e obedecem às normas da ABNT? | X | | |
| | 11. REFERÊNCIAS E CITAÇÕES<br>As referências obedecem às normas da ABNT? | | X | |
| | As citações obedecem às normas da ABNT? | X | | |
| | Todos os documentos citados foram referenciados e vice-versa, isto é, as citações e referências são consistentes? | X | | |