

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
AMAZONAS
CAMPUS MANAUS CENTRO
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA E COMUNICAÇÃO
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS**

ALESSANDRO BARBOSA DE OLIVEIRA

**PROTÓTIPO DE FERRAMENTA PARA GERENCIAMENTO AUTOMÁTICO
SCRUM EM PEQUENOS PROJETOS ÁGEIS**

**Manaus - AM
2023**

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
AMAZONAS
CAMPUS MANAUS CENTRO**

ALESSANDRO BARBOSA DE OLIVEIRA

**PROTÓTIPO DE FERRAMENTA PARA GERENCIAMENTO AUTOMÁTICO
SCRUM EM PEQUENOS PROJETOS ÁGEIS**

Trabalho de conclusão de curso apresentado à banca examinadora do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Amazonas - Campus Manaus Centro, como requisito para o cumprimento da disciplina TCC2 - Desenvolvimento de Software.

Orientador (a): Jorge Abílio Abinader Neto

**Manaus - AM
2023**

Biblioteca do IFAM – Campus Manaus Centro

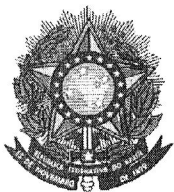
O48p Oliveira, Alessandro Barbosa.
Protótipo de ferramenta para gerenciamento automático SCRUM em
pequenos projetos ágeis / Alessandro Barbosa Oliveira. – Manaus, 2023.
55 p. : il. color.

Trabalho de Conclusão de Curso (Tecnologia em Análise e
Desenvolvimento de Sistema) – Instituto Federal de Educação, Ciência e
Tecnologia do Amazonas, *Campus* Manaus Centro, 2023.

Orientador: Prof. Me. Jorge Abílio Abinader Neto.

1. Desenvolvimento de sistema. 2. Framework Scrum. 3. Ferramentas de
software. I. Abinader Neto, Jorge Abílio. (Orient.) II. Instituto Federal de
Educação, Ciência e Tecnologia do Amazonas III. Título.

CDD 005.3



MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO MÉDIA E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA - AM
DEPARTAMENTO ACADÊMICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
AMAZONAS

TERMO DE APROVAÇÃO

A monografia, que tem como título: **PROTÓTIPO DE FERRAMENTA PARA GERENCIAMENTO AUTOMÁTICO SCRUM EM PEQUENOS PROJETOS ÁGEIS** foi submetida à defesa pública, sob a avaliação de banca examinadora, como parte dos requisitos necessários para a obtenção do título de graduação do curso superior de Tecnologia em Análise e Desenvolvimento de Sistemas.

AUTOR (A): ALESSANDRO BARBOSA DE OLIVEIRA

Monografia aprovada em: 26 / 6 / 2023

Alessandro

Orientador (a):

Rogério Lomini

Primeiro (a) examinador (a):

[Assinatura]

Segundo (a) examinador (a):

AGRADECIMENTOS

Agradeço primeiramente a Deus que me permitiu chegar a esse momento, com saúde, dedicação e foco no desenvolvimento deste trabalho.

Por conseguinte, agradeço aos meus familiares por me apoiarem no transcorrer da graduação e principalmente no foco até essa reta final da conclusão do curso. Por fim agradeço, e muito ao meu orientador, pela confiança neste trabalho por não me deixar desistir em nenhum momento, pelas devidas orientações, cobranças, permitindo assim a conclusão deste trabalho no seu devido tempo necessário.

"Seja você quem for, seja qual for a posição social que você tenha na vida, a mais alta ou a mais baixa, tenha sempre como meta muita força, muita determinação e sempre faça tudo com muito amor e com muita fé em Deus, que um dia você chega lá. De alguma maneira você chega lá."

(Ayrton Senna)

RESUMO

Atualmente o framework Scrum é adaptável em quase todos os contextos na construção de projetos, em particular no desenvolvimento de projetos de software. Para colaborar com os envolvidos no framework Scrum houve a necessidade de ferramentas para o gerenciamento de projetos, o mais conhecido é o quadro Kanban, cuja finalidade é controlar e gerenciar as tarefas diárias da equipe por meio de cartões. Com base nesse contexto este trabalho tem como objetivo apresentar um sistema web colaborativo para gerenciamento de tarefas, permitindo o seu controle pelos membros da equipe Scrum, com o diferencial de substituir os movimentos manuais dos cartões de tarefas, (o que é característica das ferramentas Kanban) por movimentos automatizados dos mesmos. Para este cenário a ferramenta foi desenvolvida em HTML, CSS, JavaScript, PHP em contribuição com os frameworks Laravel, Materialize CSS e Tailwind CSS, além da linguagem de banco de dados MySQL. Os resultados esperados foram obtidos conforme o diferencial da ferramenta perante o tradicional utilizado pelas inúmeras ferramentas do mercado.

Palavras-chave: Framework Scrum, quadro de tarefas Kanban.

ABSTRACT

Currently, the Scrum framework is adaptable in almost all contexts in the construction of projects, in particular in the development of software projects. To collaborate with those involved in the Scrum framework, there was a need for tools for project management, the best known is the Kanban board, whose purpose is to control and manage the team's daily tasks through cards. Based on this context, this work aims to present a collaborative web system for task management, allowing its control by the members of the Scrum team, with the differential of replacing the manual movements of the task cards, (which is characteristic of the Kanban tools) by automated movements thereof. For this scenario, the tool was developed in HTML, CSS, JavaScript, PHP in contribution to the frameworks Laravel, Materialize CSS and Tailwind CSS, in addition to the MySQL database language. The expected results were obtained according to the differential of the tool compared to the traditional one used by the numerous tools on the market.

Keywords: Scrum framework, Kanban task board.

LISTA DE FIGURAS

Figura 1: o Ciclo do Scrum.....	18
Figura 2: Exemplo de quadro de tarefas representando a Sprint Backlog.	20
Figura 3: Código CSS com o atributo draggable “true”.	24
Figura 4: Scrum Board (Quadro Scrum).....	27
Figura 5: Quadro Scrumwise (Quadro Kanban)	27
Figura 6: Quadro de Tarefas IceScrum	28
Figura 7: Diagrama de caso de uso do sistema	33
Figura 8: Diagrama de classe do sistema	34
Figura 9: Modelagem Conceitual de Dados do sistema	37
Figura 10: Modelagem Lógica de Dados do sistema	38
Figura 11: Trecho de código para adicionar um novo item ao backlog	39
Figura 12: Trecho de código que demonstra como adicionar um novo item a coluna backlog.....	40
Figura 13: trecho de código correspondente a adicionar um cartão de tarefas ao dashboard.	41
Figura 14: trecho de código que adiciona um cartão de tarefas ao dashboard.	41
Figura 15: Trecho de código referente aos avanços dos cartões no dashboard.	42
Figura 16: Tela de Cadastro de Itens	43
Figura 17: Tela de Dashboard para acesso à tela de Adicionar Tarefas.....	44
Figura 18: Tela de Cadastro de Tarefas.....	45
Figura 19: Tela de Dashboard do Sistema	45
Figura 20: Tela de apresentação do sistema	52
Figura 21: Tela de Registro de Usuário do sistema	52
Figura 22: Tela de Login de Usuário do Sistema	53

LISTA DE TABELAS

Tabela 1: Lista de requisitos funcionais	30
Tabela 2: Lista de requisitos não funcionais	31
Tabela 3: Caso de uso Gerar Lista de Itens	34
Tabela 4: Caso de Uso Manter cartões do quadro Scrum	35
Tabela 5: Caso de uso Gerenciar quadro Scrum	36
Tabela 6: Comparativa das Ferramentas	47

LISTA DE QUADROS

Quadro 1: Trecho de código para adicionar um atributo draggable	25
Quadro 2: Exemplo de propriedade dataTransfer	25
Quadro 3: Exemplo de código com o método para imagem dragImage	26

LISTA DE ABREVIATURAS E SIGLAS

API (Application Programming Interface) Interface de Programação de Aplicação.

CSS (Cascading Style Sheet) Folha de Estilo em Cascatas.

DnD (Drag and Drop) Arraste e Solte.

HTML (Hypertext Markup Language) Linguagem de Marcação de Hipertexto.

MVC (Model View Controller) Modelo Visão e Controle.

PHP (Hypertext Preprocessor) Pré Processador de Hipertexto.

RF Requisitos Funcionais.

RNF Requisitos não Funcionais.

SMS (Short Message Service) Serviço de Mensagens Curtas.

SQL (Structured Query Language) Linguagem de Consulta Estruturada.

UML (Unified Modeling Language) Linguagem de Modelagem Unificada.

SUMÁRIO

1. INTRODUÇÃO	12
1.1 Problematização	13
1.2 Justificativa	13
1.3 Objetivos	14
1.3.1 Objetivo geral	14
1.3.2 Objetivos específicos	14
1.4 Estrutura do Trabalho	15
2. REFERÊNCIAL TEÓRICO	16
2.1 Scrum.....	16
2.1.1 Time Scrum.....	16
2.1.2 Eventos Scrum.....	17
2.1.3 Artefatos do Scrum	19
2.2 Laravel	20
2.2.1 Autenticação	21
2.2.2 Laravel MVC	22
2.3 HTML5 Drag & Drop	23
2.3.1 Eventos Drag	24
2.4 Ferramentas Correlatas	26
2.4.1 A Ferramenta Jira Software	26
2.4.2 A Ferramenta Scrumwise.....	27
2.4.3 A Ferramenta IceScrum	28
3. DESENVOLVIMENTO	29
3.1 Introdução	29
3.2 Introdução a Ferramenta.....	29
3.2.1 Requisitos de Software	30
3.2.2 Características da Ferramenta.....	31
3.3 Diagramas UML da Ferramenta.....	32
3.3.1 Diagrama de Caso de Uso.....	32
3.3.2 Diagrama de Classe	33
3.4 Descrição dos principais Casos de Uso da Ferramenta	34
3.4.1 Caso de uso Gerar Lista de Itens	34

3.4.2 Caso de Uso Manter cartões do quadro Scrum	35
3.4.3 Caso de uso Gerenciar Quadro Scrum	36
3.5 Modelagem Conceitual de dados	37
3.6 Modelagem Lógica de dados	37
3.7 Principais trechos de codificação	39
3.7.1 Funcionalidade Gerar Lista de Itens	39
3.7.2 Funcionalidade Gerenciar Cartões do Quadro Scrum	41
3.7.3 Funcionalidade Gerenciar Quadro Scrum	42
3.8 Principais telas do Sistema	43
3.8.1 Tela de cadastro de Item da Sprint Backlog	43
3.8.2 Tela de Cadastro de cartão de tarefas	44
3.8.3 Tela de Dashboard (Quadro Scrum) do Sistema	45
3.8.4 Resultados obtidos	46
4. CONSIDERAÇÕES FINAIS	48
4.1 Riscos e Dificuldades	48
4.2 Lições Aprendidas	48
4.3 Trabalhos Futuros	49
4.4 Conclusão	49
REFERÊNCIAS	50
APÊNDICES	52

1. INTRODUÇÃO

O framework Scrum existe desde a década de 90 e se popularizou nas décadas seguintes, ganhando o mundo, e inclusive substituindo métodos tradicionais no desenvolvimento de projetos de software (SABBAGH, 2022). Os idealizadores desse framework são Jeff Sutherland e Ken Schwaber, que em 1995 formalizaram esse método e o chamaram de Scrum que desde então vem recebendo atualizações em seu documento, Scrum Guide - Guia do Scrum (AUDY, 2015).

A metodologia ágil Scrum envolve um conjunto de diretrizes para construção em projetos de software que tem como principal característica os incrementos que são pequenas funcionalidades a serem entregues ao cliente, geralmente ocorrem de duas a três semanas, conforme as especificações (SOMMERVILLE, 2019). Os clientes são envolvidos durante a concepção do software para poderem dar um feedback rápido de acordo com a evolução dos requisitos (SOMMERVILLE, 2019).

O Scrum é pequeno e simples podendo ser aplicado em diferentes contextos de projetos de software, que vão desde os mais simples até críticos. Projetos esses que podem ser sites da internet, softwares comerciais, softwares embarcados, sistemas financeiros, dispositivos móveis, jogos e etc. (SABBAGH, 2022).

Diversos conceitos e características são aprendidos e aplicados dentro do Scrum como, facilidade, trabalho em equipe, auto-organização, metas de negócio, motivação, relacionamento com clientes, e muitos outros. Conceitos novos são relativamente poucos, o que o Scrum idealiza é unir práticas de mercado já utilizadas e organizá-las de maneira que funcione (SABBAGH, 2022).

Por meio destes conceitos este trabalho terá como foco o framework Scrum, que com sua metodologia traz muitos benefícios e vantagens em relação a outras maneiras de se dirigir projetos de software (MARTINS, 2017). Como todo processo de desenvolvimento de software é necessárias ferramentas para gerenciar todo um processo metodológico que auxilia e organiza o trabalho da equipe Scrum e considerando este cenário, o trabalho em questão desenvolveu um protótipo de ferramenta para organizar e gerenciar as tarefas que os desenvolvedores realizam na metodologia ágil Scrum.

1.1 Problematização

A metodologia ágil Scrum tem em seus conceitos característicos que o fazem atualmente no framework mais usado no mercado de desenvolvimento de software, visto que pequenas equipes (delimitadas de três a cinco membros) são as que mais a adaptam para os seus projetos permitindo que todos tenham visão do que precisa ser feito durante todo o processo de desenvolvimento até atingirem a meta do projeto (SABBAGH, 2022).

O Scrum Team é consideravelmente pequeno, consiste dos seguintes participantes: um Scrum Master, um Product Owner e os Developers (SCHWABER, SUTHERLAND, 2020). O foco das equipes em questão será o Scrum Master e a sua equipe de desenvolvedores (de três a cinco membros).

Assim como em todo projeto em construção há a necessidade de gerenciá-lo e organizá-lo de forma que as equipes envolvidas possam conceber seus projetos sem perda da qualidade e principalmente entregues dentro dos prazos definidos. Esse gerenciamento pode ser feito por meio de ferramentas de uso organizacional ou pessoal (a depender da familiaridade de cada equipe), com o apoio delas é possível organizar as tarefas e distribuí-las aos membros da equipe (MARTINS, 2017).

Tendo em vista que o uso dessas ferramentas auxilia as equipes ágeis Scrum, este estudo tem como finalidade a responder a pergunta: Como o desenvolvimento de uma ferramenta de gestão ampararia os processos dentro da metodologia ágil em pequenas equipes Scrum e qual seria a contribuição para o aumento da produtividade com a construção dessa ferramenta para automatizar o gerenciamento de tarefas dessas equipes ágeis?

1.2 Justificativa

A rotina das equipes ágeis é definida na fase inicial de desenvolvimento do projeto até ser finalizado, portanto é importante que toda a equipe de desenvolvedores seja multidisciplinar e motivada para que possa ter um bom controle, gerenciamento do grupo e organização das tarefas (SABBAGH, 2022).

As tarefas são divididas de acordo com as habilidades de cada membro da equipe que podem ser visualizadas por todos (SUTHERLAND e SCHWABER, 2020),

e conforme a forma de trabalho (presencialmente ou remotamente) a ferramenta de gerenciamento precisa operar de maneira a atender a todas as necessidades da equipe (MARTINS, 2017).

Quando se trata de pequenas e delimitadas as equipes ágeis o Scrum é bastante flexível no que diz respeito à organização, compartilhamento de informações e principalmente a comunicação dos membros da equipe com o projeto, no caso das tarefas elas são postas em um quadro (quadro Scrum) que são movidos de maneira manual por cada membro, permitindo maior integração de toda a equipe (SEGUNDO, 2019).

Com a utilização de uma ferramenta de gestão automatizada o quadro Scrum poderá eliminar o controle manual, o que é uma vantagem visto que algum membro da equipe poderá esquecer-se de movê-lo, mesmo que a tarefa tenha avançado ao próximo nível ou concluída. Esse fato consequentemente poderia atrasar a entrega de alguma funcionalidade, além do problema de comunicação quanto ao responsável pela tarefa em questão e em qual fase se encontra. O que é o principal propósito da ferramenta que desenvolvemos e relatamos nesse estudo.

1.3 Objetivos

1.3.1 Objetivo geral

Este trabalho tem como objetivo descrever o desenvolvimento de uma ferramenta para pequenas equipes ágeis Scrum a fim de customizar os movimentos dos cartões, além organizar e compartilhar as tarefas do quadro Scrum em seus projetos.

1.3.2 Objetivos específicos

- Atender as pequenas equipes Scrum integrando um processo customizado por meio de uma ferramenta de gestão de projetos;
- Proporcionar o gerenciamento, controle e compartilhamento de informações no quadro de tarefas Scrum de toda a equipe;
- Permitir o avanço com um clique das tarefas executadas para o próximo nível até concluí-las, eliminando os movimentos manuais realizados pelos membros da equipe.

1.4 Estrutura do Trabalho

Este trabalho está organizado conforme se segue:

O capítulo dois trata-se dos principais conceitos e características a respeito do framework Scrum por meio dos referenciais teóricos abordados, além disso, apontam-se os principais recursos e tecnologias que serão utilizados para a construção da ferramenta com o também framework Laravel.

O capítulo três descreve todos os detalhes para a construção do sistema, suas características e funcionalidades com ênfase nas principais, tecnologias empregadas para a construção.

O capítulo quatro apresenta os riscos, as dificuldades ocorridas, conceituações aprendidas com o desenvolvimento deste trabalho, e os futuros projetos além das devidas considerações finais.

2. REFERÊNCIAL TEÓRICO

Este capítulo apresenta a fundamentação teórica base que foi obtida por meio de pesquisas bibliográficas e demais trabalhos correlatos sobre o framework Scrum e suas principais características abordadas neste trabalho. Além destas os demais materiais técnicos, das obras literárias apresentadas que colaboraram para a construção do sistema.

2.1 Scrum

O Scrum será abordado nesse trabalho com caráter superficial, serão destacados os eventos, os seus integrantes e principais características adotadas pelo time no cotidiano de desenvolvimento do seu trabalho, a fim de esclarecer o conhecimento necessário para a compreensão da ferramenta desenvolvida.

2.1.1 Time Scrum

O Scrum Team consiste de um time relativamente pequeno de pessoas. Um Scrum Master, um Product Owner e os Developers (no contexto deste projeto o time será constituído de no máximo cinco integrantes). Dentro do Scrum eles são conhecidos como papéis e são os profissionais que focam no trabalho e como objetivo final alcançar a meta do produto (SCHWABER e SUTTERLAND, 2020).

2.1.1.1 Developers

É a equipe de desenvolvimento e demais profissionais que participam, compartilham do mesmo objetivo e comprometem-se na criação dos incrementos utilizáveis (SCHWABER e SUTTERLAND, 2020).

Os developers são multidisciplinares, ou seja, os membros da equipe detêm todos os conhecimentos necessários para a realização dos trabalhos. São caracteristicamente organizados, possuem a capacidade de gerenciar a si mesmos, além de definir como irão realizar os seus trabalhos, a fim de alcançar a meta do produto (SABBAGH, 2022).

2.1.1.2 Product Owner

O Product Owner é o grande responsável por tomadas de decisões perante o time, ele representa o cliente dentro da equipe (AUDY, 2015). Tem um importante

papel, por determinar a comunicação e a constante visibilidade do Produto conforme ele é desenvolvido pelo time (SABBAGH, 2022).

A visibilidade do Product Owner permite que o mesmo crie planos mais apropriados na espera do desenvolvimento do produto e o seu progresso ao longo do tempo (termo conhecido como Roadmap do produto). Em linhas gerais é um plano expresso por linha do tempo com as metas almejadas no decorrer de sua evolução e a sua devida concordância no futuro (SABBAGH, 2022).

2.1.1.3 Scrum Master

O Scrum Master faz o papel de líder e que servem ao Scrum Team assim como a toda organização. Ele é responsável por promover o Scrum dentro do time conforme o guia do Scrum (SCHWABER e SUTTERLAND, 2020).

O Scrum Master tem como principais atribuições e responsabilidades, treinar, orientar e gerenciar o time Scrum, tem conhecimento da empresa e seus colegas, proporcionando e incentivando aos envolvidos para o crescimento de todos do time, e que possam alcançar os seus objetivos (AUDY, 2015).

2.1.2 Eventos Scrum

Dentro do Scrum os eventos são uma oportunidade de inspecionar, analisar e adaptar os artefatos do Scrum. O que auxilia na identificação da transparência destas adaptações. Os eventos no Scrum são utilizados com a intenção de criar regularidade, evitando assim reuniões desnecessárias do time (SCHWABER e SUTTERLAND, 2020).

2.1.2.1 A Sprint

A construção do produto é dividida em ciclos do projeto, dentro destes ciclos todo o trabalho é desenvolvido, ciclos esses chamados de Sprints, que são relativamente à essência do Scrum (SABBAGH, 2022), como pode ser visto o ciclo completo da Sprint na figura 1.

Esses eventos duram fixamente a cada um mês ou menos, conforme o contexto do projeto, a fim de criar uma padronização. E a cada Sprint finalizada inicia-se imediatamente a outra (SCHWABER e SUTTERLAND, 2020).

Dentro das Sprints ocorrem todos os trabalhos que são executados para atingir a meta do produto, que são: SPRINT PLANNING, DAILY SCRUM, SPRINT REVIEW E SPRINT RETROSPECTIVE (SCHWABER E SUTTERLAND, 2020).

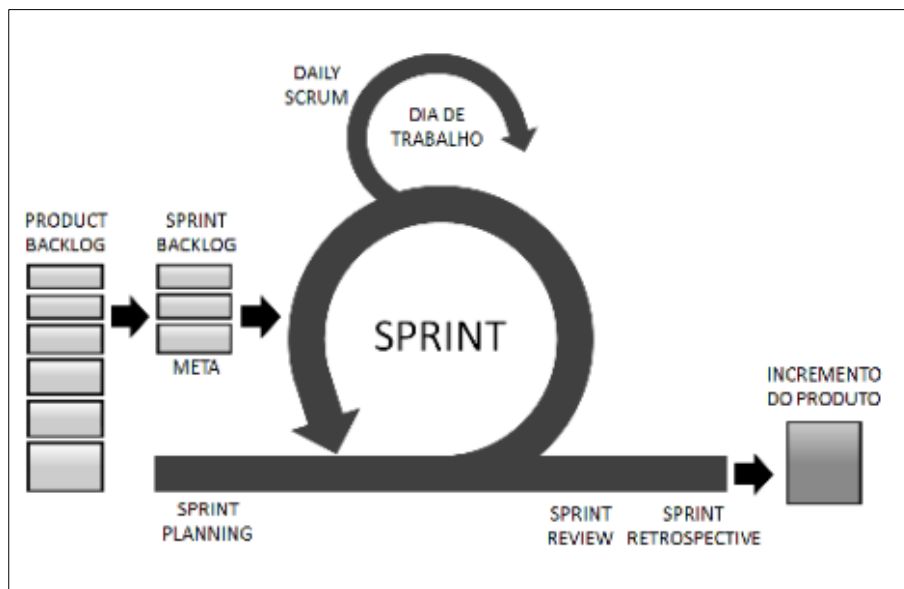


Figura 1: o Ciclo do Scrum Fonte: (SABBAGH, 2022).

2.1.2.2 Sprint Planning

Como em todo trabalho bem realizado há a necessidade de um bom planejamento, no caso a Sprint de início com a reunião da Sprint Planning. Essa reunião todo o time de desenvolvimento e o Product Owner discutem e negociam os itens que compõem o Product Backlog, que será desenvolvido (SABBAGH, 2022).

Dentro da Sprint Planning são abordados os seguintes tópicos: A avaliação da Sprint e a sua utilidade, com as definições de uma meta a ser alcançada. O que pode ser feito nesta Sprint, já que o Time Scrum define os itens dessa Sprint. Por fim como o trabalho será realizado, com o devido planejamento necessário dos itens a serem desenvolvidos e incrementados na Sprint (Sabbagh, 2022).

2.1.2.3 Daily Scrum

Em cada dia de trabalho da equipe de desenvolvimento, há encontros que duram no máximo quinze minutos, preferencialmente no mesmo horário e local, Daily Scrum. O seu objetivo é para tornar visível e de conhecimento de todos da

equipe o andamento dos trabalhos, e a fim de planejar o próximo dia de trabalho, afirma SABBAGH (2022).

2.1.2.4 Sprint Review

Ao encerramento do *Sprint*, o Time de Desenvolvimento e o Product Owner realizam uma ou duas reuniões importantes para a finalização da Sprint atual, essa reunião é a Sprint Review, que tem por objetivo realizar a inspeção e adaptação do produto (SABBAGH, 2022).

O Scrum Team apresenta os resultados de seu trabalho para os principais interessados do produto, assim a fim de debaterem o progresso para a próxima Meta do Produto (SCHWABER e SUTTERLAND, 2020).

2.1.2.5 Sprint Retrospective

A última reunião realizada é a Sprint Retrospective, que tem por visão o time, faz-se a inspeção e adaptações a serem tratadas na forma de trabalhar do Time Scrum durante o desenvolvimento do produto (SABBAGH, 2022).

O Scrum Team analisa a Sprint anterior em relação aos processos adotados, os indivíduos, interações do time, ferramentas e sua Definition of Done (definição de pronto). O Scrum Team discute e avalia como foram as dificuldades encontradas e o que ocorreu de certo durante a Sprint e como foram ou não resolvidos (SCHWABER e SUTTERLAND, 2020).

2.1.3 Artefatos do Scrum

Os artefatos do Scrum representam o trabalho ou valor, e são projetados para estimar a eficiência das principais informações. Cada artefato é composto de um compromisso a fim de garantir que as informações aumentam a transparência e foco a fim de reforçar o empirismo e o esforço do Scrum para o Scrum Team (SCHWABER e SUTTERLAND, 2020).

2.1.3.1 Product Backlog

O Product Backlog contém todos os requisitos necessários e já identificados pelos envolvidos, sendo uma lista ordenada do que é sucinto para aprimorar o produto (AUDY, 2015). O Product Owner cria essa lista ordenada, geralmente incompleta e dinâmica de itens que representam o que será produzido ao longo da

construção do projeto. Os itens do Product Backlog são os mais importantes e podem conter detalhes que os priorizam na construção, sendo os menos importantes compostos na lista mais abaixo (SABBAGH, 2022).

2.1.3.2 Sprint Backlog

O Sprint Backlog é composto pela Meta da Sprint e corresponde aos itens que foram escolhidos para a Sprint, assim como um plano de ação para a devida entrega do incremento ao seu término (SCHWABER e SUTTERLAND, 2020). O conjunto desses itens da Sprint Backlog é geralmente expressado em tarefas a serem realizadas e é comumente demonstrado em um plano de quadro Kanban ou Quadro de tarefas (SABBAGH, 2022) a figura 2 demonstra um exemplo de quadro Kanban.

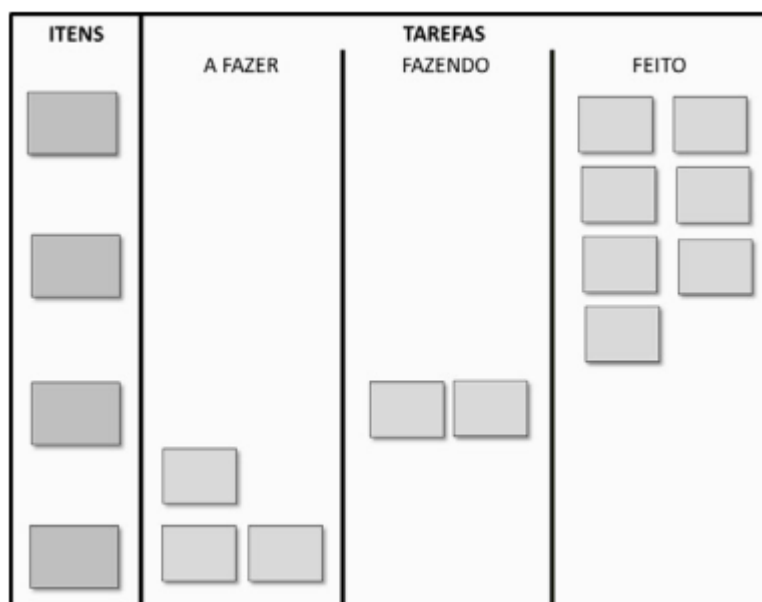


Figura 2: Exemplo de quadro de tarefas representando a Sprint Backlog Fonte: (SABBAGH, 2022).

2.2 Laravel

O Laravel é um framework fullstack criado em junho de 2011 por Taylor Otwell, e até a produção deste trabalho estava na versão 10.0. Desde então vem sendo bastante utilizado por meio de suas features e contribuições quanto ao desenvolvimento de aplicações e diversos serviços para a web (CASTRO, 2019).

Essa seção apresentará alguns recursos utilizados e importantes para a construção da ferramenta, conceitos e principais detalhes de suas aplicações que podem ser utilizados com o framework Laravel.

2.2.1 Autenticação

O Laravel permite criar um serviço de autenticação de acesso aos usuários (e-mail e password), o que facilita muito a construção em poucos passos esse serviço de forma rápida e produtiva. Todo esse recurso pronto é ágil e dinâmico na aplicação e o acesso do usuário ao painel do sistema (Dashboard), de acordo com CASTRO (2019).

O Laravel possui um grande pacote de kits iniciais para seus aplicativos. Esses kits ajudam automaticamente com as rotas, controllers e views a fim de registrar e autenticar os usuários de sua aplicação.

Um destes kits é o Laravel Breeze que corresponde a uma implementação simples e mínima para trabalhar com recursos de autenticação da Laravel, como login, registro, redefinição de senha, verificação de email e confirmação de senha. O kit inclui também uma implementação de página simples de perfil onde dados podem ser atualizados pelo usuário (LARAVEL, 2023 - tradução nossa).

Após a criação de um novo projeto Laravel e configurado o seu banco de dados, executa-se o comando de instalação do Laravel Breeze usando o composer require do Laravel (LARAVEL, 2003 - tradução nossa).

Com essa instalação do pacote Laravel Breeze, você pode utilizar um pacote de pilhas do Breeze que permite a publicação e exibição de autenticação, rotas, controladores e outros recursos em seu aplicativo (LARAVEL, 2003 - tradução nossa).

A pilha padrão do Breeze é o Blade, que é um template simples para renderizar o front-end de seu aplicativo. O Blade já possui diversas estruturas que facilitam a construção e principalmente a criação das suas views, estruturas que contém condicionais, laços, assim como diversas diretivas para auxiliar a criação dos controles de maneira rápida e simples (CASTRO, 2019), o Blade será abordado mais adiante.

2.2.2 Laravel MVC

O modelo padrão de mapeamento base utilizado pelo Laravel é o padrão MVC, ou, Model-View-Controller. A grande maioria dos frameworks da atualidade utiliza este padrão para melhor organização dos arquivos e demais componentes dos projetos Laravel (CASTRO, 2019).

2.2.2.1 Model

A camada de Model ou Modelo dentro do Laravel serão as classes que representam as tabelas do banco de dados da sua aplicação, que posteriormente podem ser manipulados conforme as requisições ao banco de dados. Por padrão, as classes do model podem ser encontradas dentro da pasta app na raiz do projeto (CASTRO, 2019).

Os Models são a representação conforme os objetos das tabelas do banco de dados, ou em outros termos representam as entidades correspondentes a todos os dados da tabela em questão (CASTRO, 2019).

Por convenção o Laravel tentará representar a tabela em questão de acordo com o nome requisitado, por exemplo, caso você tenha uma tabela que atende por nome de posts na base, a representação de seu model na tabela será uma classe chamada de Post. Se eu tenho uma tabela users sua representação via model será uma classe atendida por nome de User (CASTRO, 2019).

2.2.2.2 Controller

A camada de Controller ou Controle tem por ideia ser o mais simples possível, afim de que se evite utilizar regras excessivamente complexas em seus Controllers. Dentro do projeto Laravel estes controllers encontram-se na pasta app/Http/Controllers (CASTRO, 2019).

O Controller é importantíssimo na arquitetura do framework Laravel, como já mencionados os controllers são o ponto de delegação entre as camadas de Model e View. No Laravel por meio do comando artisan pode-se automatizar a geração de controllers, com o comando make (CASTRO, 2019).

2.2.2.3 Rotas

O mapeamento de nossas URLs no Laravel permite melhor visibilidade e controle das mesmas. O mapeamento das urls sucede dentro dos arquivos de rotas, o que facilita ter mais controle sobre o que será exposto, além de customização as rotas que queremos realizar (CASTRO, 2019).

No Laravel os arquivos de rotas são bem distribuídos, o que consequentemente ajuda na organização das rotas dos arquivos a serem trabalhados, auxiliando também conforme o contexto da aplicação, tornando inclusive as rotas maiores nesse quesito (CASTRO, 2019).

2.2.2.4 View

A camada que permite a interação do usuário é a View ou Visualização. Esta camada é exclusivamente para exibição de resultados e dentro do Laravel nossas views encontram-se na pasta resources/views (CASTRO, 2019).

2.2.2.5 Blade

O Laravel possui diversos templates para auxiliar na interação com o usuário, o template padrão é a engine Blade, para a criação de layouts para as views da aplicação. O Blade ao longo do tempo vem recebendo diversos incrementos e melhorias, por padrão o Blade cria um arquivo chamado app.blade.php, que fica localizado na pasta layout dentro da view do projeto (CASTRO, 2019).

Como padrão de organização do template Blade no Laravel é necessário à criação de uma pasta layout que, por conseguinte está dentro da pasta view. Com esse padrão devidamente estruturado, é criado o arquivo como no exemplo acima, app.blade.php e com esse layout definido, o Blade permite herdar os templates estendendo o layout Blade, conforme o exemplo `@extends('layouts.app')` (CASTRO, 2019):

2.3 HTML5 Drag & Drop

No HTML há um recurso (API) que permite ao usuário arrastar qualquer elemento de uma página web, recurso esse chamado Drag and Drop (arraste e solte) o que é uma característica comum dos websites (ANDREW E BIDEELLMAN,

2021 - tradução nossa). As interfaces Drag and Drop permitem que as aplicações possam utilizar as funcionalidades de arrastar e soltar através do navegador (MOZILLA, 2022 - tradução nossa).

Para tornar os elementos arrastáveis, você necessitará utilizar as APIs do DnD (Drag and Drop) HTML5, o que pode ser realizado com o seguinte elemento na tag do HTML5: `draggable="true"`, permitindo assim que seja habilitado para arrastar, imagens, arquivos, links, ou marcação da página (ANDREW E BIDEELLMAN, 2021 - tradução nossa).

No exemplo a seguir é criada uma interface para reorganizar colunas, que foram dispostas com um Grid em CSS. Uma marcação básica como no exemplo da figura 3 para as colunas aonde cada elemento, no caso "div" em que o atributo "draggable" é definido com "true", o que permite ao elemento draggable ser controlado e arrastável pela Grid do CSS neste exemplo (ANDREW E BIDEELLMAN, 2021 - tradução nossa).

```
<div class="container">
  <div draggable="true" class="box">A</div>
  <div draggable="true" class="box">B</div>
  <div draggable="true" class="box">C</div>
</div>
```

Figura 3: Código CSS com o atributo draggable "true"
Fonte: (WEBDEV, 2021).

2.3.1 Eventos Drag

Os eventos de Drag que o HTML utiliza faz uso dos eventos conhecido como modelo de eventos DOM que são herdados dos eventos de mouse. Um exemplo típico de evento de mouse é o usuário selecionar um elemento com a propriedade draggable permitindo que o arraste até a um elemento definido como "droppable" (Drop) (MOZILLA, 2022).

As interfaces no HTML o drag and drop são constituídos dos eventos: DragEvent, DataTransfer, DataTransferItem e DataTransferItemList. DragEvent consiste de um construtor e uma propriedade, DataTransfer é um objeto, para cada

objeto `DataTransfer` é necessária uma propriedade `items` que é uma lista. Por fim cada objeto `DataTransferItem` representa um `Drag Item` (MOZILLA, 2022 - tradução nossa).

Para que um elemento permita ser arrastado, será necessário adicionar o atributo `draggable`, aos eventos globais `ondragstart`, conforme mostrado no quadro 1 (MOZILLA, 2022 - tradução nossa).

Quadro 1: Trecho de código para adicionar um atributo `draggable`

```
1 function dragstart_handler(ev) {
2     console.log("dragStart");
3     // Adiciona o id do elemento em questão ao objeto de transferência de
      dados (dataTransfer)
4     ev.dataTransfer.setData("text/plain", ev.target.id);
5 }
6 <body>
7 <p id="p1" draggable="true" ondragstart="dragstart_handler(event);">Este
      elemento é arrastável.</p>
8 </body>
```

Fonte: (MOZILLA, 2022)

Em cada evento `drag` a mesma possui uma propriedade `dataTransfer` (linha 4) que segura os dados do evento utilizado. Essa propriedade, que por sinal é um objeto `dataTransfer` também inclui um método para gerenciar os dados do arraste, no caso o `drag`. Por fim o método `setData()` (linha 4) é usado para acrescentar um item aos dados do arraste (MOZILLA, 2022 - tradução nossa), confira o quadro 10 a seguir.

Quadro 2: Exemplo de propriedade `dataTransfer`

```
1 function dragstart_handler(ev) {
2     // Adiciona os dados do arraste (drag)
3     ev.dataTransfer.setData("text/plain", ev.target.id);
4     ev.dataTransfer.setData("text/html", "<p>Parágrafo de exemplo</p>");
5     ev.dataTransfer.setData("text/uri-list", "http://developer.mozilla.org");
6 }
```

Fonte: (MOZILLA, 2022)

Por fim você define uma imagem como padrão para a sua aplicação, contudo, uma aplicação pode definir uma imagem customizada utilizando o método `setDragImage()` (linha 7) como demonstrado no exemplo do quadro a seguir.

Quadro 3: Exemplo de código com o método para imagem `dragImage`

```
1    function dragstart_handler(ev) {
2        // Cria uma imagem e então a utiliza como a "drag image".
3        // NOTA: mude "example.gif" como uma imagem existente, caso
contrário
4        // ela não será criada e a imagem padrão será utilizada como padrão.
5        var img = new Image();
6        img.src = 'example.gif';
7        ev.dataTransfer.setDragImage(img, 10);
8    }
```

Fonte: (MOZILLA, 2022)

2.4 Ferramentas Correlatas

Neste tópico serão abordadas três ferramentas que foram escolhidas (dentre diversas no mercado) para melhor compreensão do software a ser desenvolvido. Será feito uma breve descrição destas ferramentas, apontando as principais características de cada uma.

2.4.1 A Ferramenta Jira Software

Distribuído pela empresa Atlassian, Jira é uma ferramenta, considerada a número um por muitos em desenvolvimento de software por equipes ágeis (ATLASSIAN, 2022). A ferramenta possui como principais características, um Painel do Scrum que, auxilia as equipes ágeis a dividir os projetos grandes tal qual o seu grau de complexidade e reduzi-las em partes menores de gerenciamento.

Além do Painel Scrum a ferramenta possui roteiros para que as equipes tenham uma visão total do contexto do projeto a ser desenvolvido, conforme o Dashboard do Jira na figura 4, relatório de dados por meio de painéis com contextos específicos, flexibilidade do projeto, repositório e integração com outras ferramentas de projeto e gerenciamento, como Github, Git, Figma (ATLASSIAN, 2022).

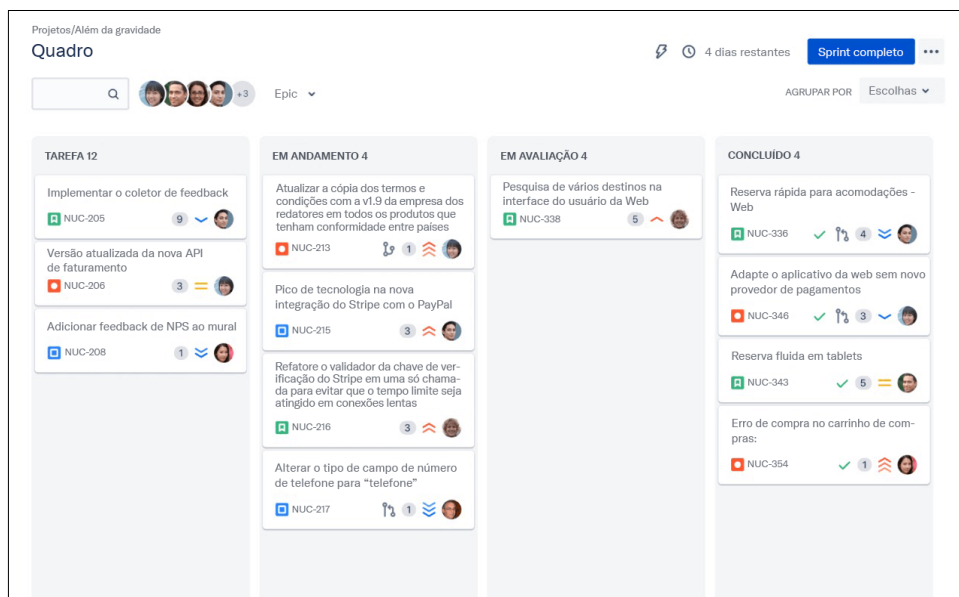


Figura 4: Scrum Board (Quadro Scrum)
Fonte: (ATLASSIAN, 2022).

2.4.2 A Ferramenta Scrumwise

O Scrumwise é uma ferramenta online e desktop para criar e gerenciar equipes, atribuir funções, compartilhar informações e tarefas, além controlar o trabalho de toda sua equipe Scrum (SCRUMWISE, 2022).

O software trabalha com o quadro de tarefas Kanban que corresponde aos itens da Sprint Backlog, figura 5, o que permite todo acompanhamento de informações e tarefas da equipe Scrum em tempo real, além disso, podem-se criar os gráficos de Burndown (SCRUMWISE, 2021).

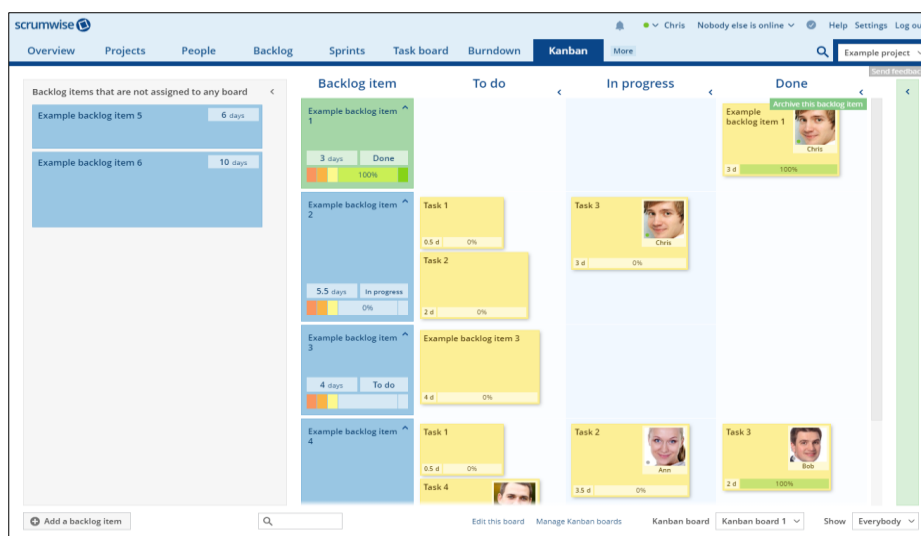


Figura 5: Quadro Scrumwise (Quadro Kanban)
Fonte: (SCRUMWISE, 2023)

2.4.3 A Ferramenta IceScrum

O IceScrum é um software web para gestão de projetos em equipes ágeis, o software permite gerenciar o Backlog da Sprint por meio de um quadro Kanban de tarefas, figura 6, o que facilita a coordenação e o trabalho de toda a equipe Scrum (ICESCRUM, 2022).

O IceScrum possibilita o planejamento de metas de forma segura, permitindo o seu posterior cumprimento, além disso, mantém uma boa iteração em seu progresso até a sua finalização. O que permite uma visão geral de seu projeto com identificadores ágeis e personalizados (ICESCRUM, 2022).

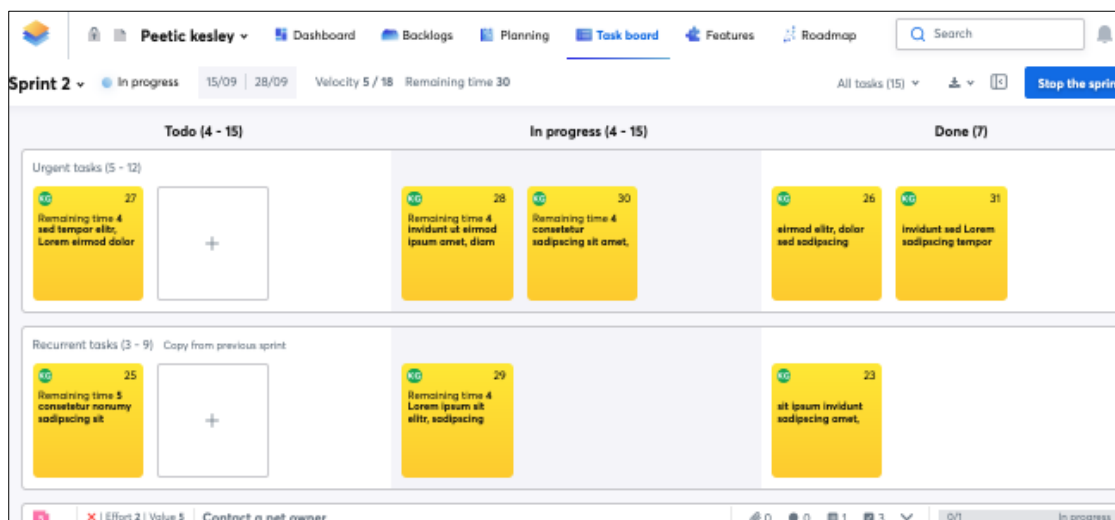


Figura 6: Quadro de Tarefas IceScrum
Fonte: (ICESCRUM, 2019)

3. DESENVOLVIMENTO

Neste capítulo será apresentada toda a estrutura base que foi utilizada para construir a ferramenta proposta (Scrum Conts – Scrum Controle de Tarefas) como as linguagens, codificação, aplicativos e recursos adicionais para a construção da aplicação. Serão detalhados os requisitos do sistema, diagramas da UML, modelo de banco de dados relacional, além do seu mapeamento lógico, descrição dos principais casos de uso e as funcionalidades mais importantes do sistema.

3.1 Introdução

Foi realizada uma breve investigação quanto aos requisitos necessários para o desenvolvimento da ferramenta (Scrum Conts), a fim de obter a descrição com as devidas funcionalidades do sistema e concluindo com as modelagens iniciais para uma breve compreensão da ferramenta antes de sua implementação.

Desta maneira o estudo realizado assim como a busca por tecnologias adequadas optou-se pelo desenvolvimento de um sistema web com o objetivo de agregar valor e que atenda as necessidades encontradas. O que será destacado a seguir com a história da ferramenta (Scrum Conts), seus requisitos e características principais.

3.2 Introdução a Ferramenta

Assim como os sistemas do mercado, o sistema permitirá que os membros da equipe possam organizar gerenciar e principalmente compartilhar as tarefas que o time realiza por meio do quadro de tarefas Scrum (uso semelhante ao quadro Kanban), com o diferencial de movimentar os cartões de tarefas com apenas um clique ou mudança de status.

O sistema atendeu as necessidades e requisitos que foram analisados, conforme descrito na seção 3.2.1 deste capítulo. A ferramenta (Scrum Conts) permite que o usuário compartilhe a lista de Backlog da Sprint, tal quais as suas tarefas realizadas e seu status, por meio de cartões que descrevem as tarefas dos membros da equipe.

O sistema permite o gerenciamento de projetos a serem devidamente cadastrados e em seguida inicializados, e conforme as Sprints avançam e o desenvolvimento do software ocorre, os cartões do quadro de tarefas a cada Sprint

também avança, até que seja finalizada a Sprint com o status de todos os cartões da lista da Sprint Backlog esteja "pronto" (done) e uma nova Sprint possa ser inicializada.

Estes são os principais elementos que a ferramenta desenvolvida (Scrum Conts) permite que o usuário as realize. A descrição dos requisitos funcionais e não funcionais, assim como as características de desenvolvimento da ferramenta serão abordadas nas seções seguintes, 3.2.1 e 3.2.2.

3.2.1 Requisitos de Software

Conforme definido na introdução da ferramenta (Scrum Conts) com os principais elementos (funcionalidades) que o sistema irá exercer exige-se a necessidade de atribuir os requisitos que serão transformados em funcionalidades no sistema a fim de atender as necessidades dos usuários. Esses requisitos da ferramenta (Scrum Conts) foram divididos em requisitos funcionais e requisitos não funcionais e são descritos nas tabelas 1 e 2 a seguir:

3.2.1.1 Requisitos funcionais

Tabela 1: Lista de requisitos funcionais

Referência	Requisito funcional
RF01	Fazer login e autenticação: responsável por permitir o login no sistema dos usuários cadastrados previamente.
RF02	Manter usuário: responsável por manter os usuários cadastrados no sistema.
RF03	Manter projetos: responsável por manter os projetos cadastrados no sistema.
RF04	Manter times: responsável por manter os times previamente cadastrados no sistema.
RF05	Manter Sprint Backlog: responsável por manter as sprints cadastradas em um projeto no sistema.
RF06	Criar dashboard do quadro Scrum: responsável por manter os itens da lista da Sprint e mostrá-los no dashboard do sistema.
RF07	Gerenciar cartões automáticos das tarefas do quadro: responsável por criar e manter os cartões de tarefas referentes a um item no dashboard do sistema.
RF08	Gerenciar Quadro Scrum: responsável por controlar e gerenciar o quadro Scrum do sistema, permitir seus avanços até o seu status de pronto da Sprint.

Fonte: autoria própria

3.2.1.2 Requisitos não funcionais

Tabela 2: Lista de requisitos não funcionais

Referência	Requisito não funcional
RNF01	O sistema foi desenvolvido para web.
RNF02	O limite máximo de integrantes da equipe Scrum está definido em até cinco integrantes.
RNF03	O sistema foi desenvolvido com a linguagem PHP

Fonte: autoria própria

Valem destacar as funcionalidades **RF06**, **RF07** e **RF08** que serão abordados e devidamente explicados na seção 3.3.1.

3.2.2 Características da Ferramenta

O sistema é totalmente livre ou free, e pode ser acessado em qualquer navegador web, e seu objetivo é de auxiliar os usuários do framework Scrum em controlar e gerenciar as tarefas do dia a dia de desenvolvimento do time de forma automatizada.

Como uma característica importante dos aplicativos da atualidade o sistema foi desenvolvido com o padrão de projeto (arquitetura), MVC ou Model-View-Controller, que por características particulares distribui o sistema em diversas camadas, separando a parte da lógica do negócio da interface do usuário.

Para a criação do sistema observou-se as suas características e optou-se pelas seguintes tecnologias para o seu desenvolvimento, linguagem PHP, versão 8 com o uso do framework Laravel (já descrito na seção 2.2), a linguagem HTML5, CSS3, a linguagem JavaScript.

Foi utilizado o também framework Tailwind CSS e o Materialize CSS, que permitem trabalhar com a estilização das páginas, já a linguagem SQL foi utilizada para a base de dados, cuja versão optou-se pela 8.0 do MySQL. Por fim o Github foi o repositório para que o projeto possa ser mantido para futuras melhorias e implementações.

A partir do cadastro do dono do projeto e em seguida o seu login o sistema permite realizar as seguintes tarefas e trabalhar com as seguintes funcionalidades definidas nos requisitos de software:

- Cadastro de usuários;

- Cadastro de projetos;
- Cadastro de time Scrum;
- Cadastro de Sprint Backlog;
- Controle de tarefas do quadro Scrum (normalmente conhecido como quadro Kanban)
- Gerenciamento e controle de cartões (itens do Sprint Backlog);
- Gerenciamento e controle dos projetos;

A seguir serão detalhados os diagramas da UML (Caso de Uso e de Classe), diagrama de modelagem conceitual de dados, além do modelo lógico e funcionalidades principais do sistema implementadas.

3.3 Diagramas UML da Ferramenta

Conforme a descrição do sistema e os requisitos definidos foram desenvolvidos os seguintes diagramas da UML, Caso de Uso (figura 7) e o diagrama de Classe (figura 8).

3.3.1 Diagrama de Caso de Uso

A figura 7 destaca o diagrama de caso de uso do sistema, e o usuário como o centro de controle de todas as operações a serem realizadas no sistema, descrevendo de maneira breve o usuário necessita realizar login (perante realização de um cadastro se usuário novo) para ter acesso ao Dashboard do sistema ou Quadro Scrum caso já tenha um projeto em andamento.

O usuário ainda terá a responsabilidade de manter as tarefas da equipe por meio do caso de uso manter cartões de tarefas, assim como manter times cadastrados (caso de uso manter times), controlar a Sprint Backlog a cada Sprint dos projetos da equipe e gerenciar a lista da Sprint Backlog. Por fim o usuário tem o controle e gerenciamento total do Quadro Scrum da equipe que é o principal caso de uso do sistema.

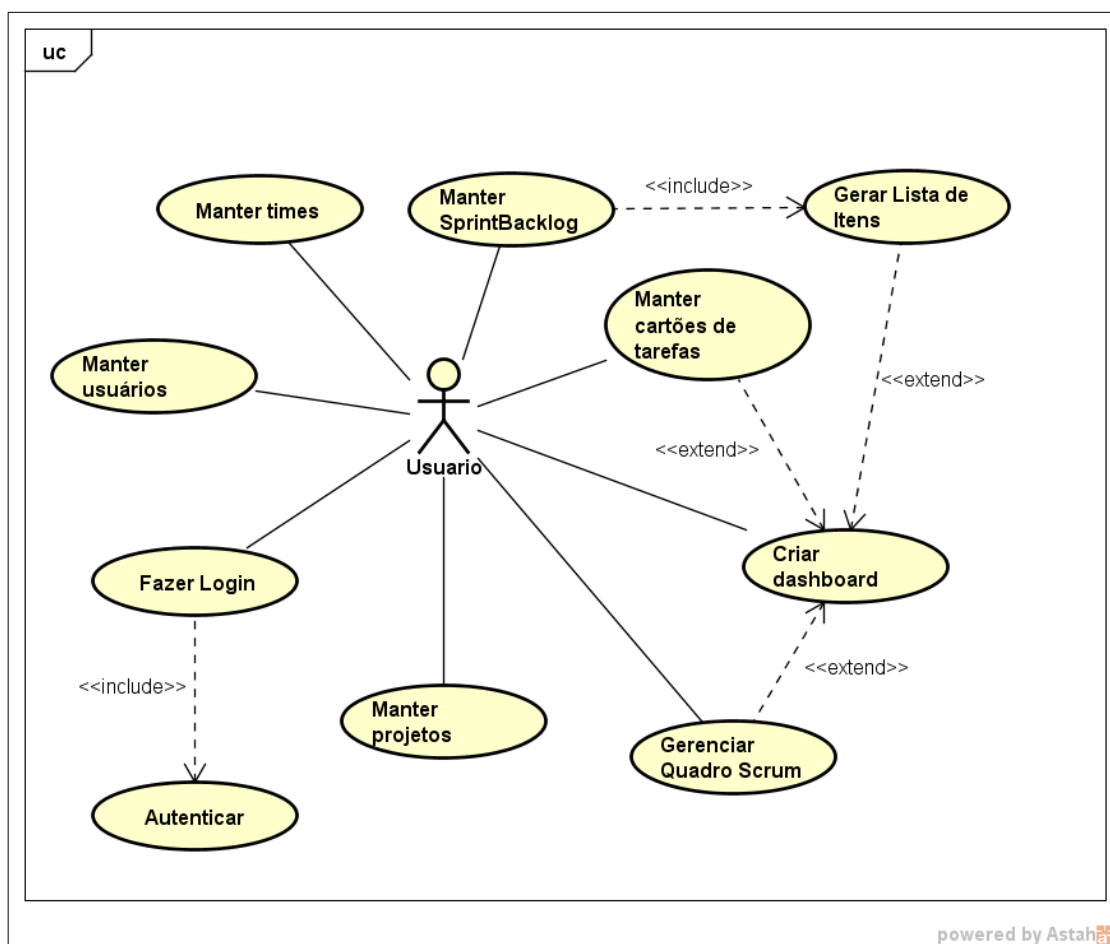


Figura 7: Diagrama de caso de uso do sistema
Fonte: autoria própria

3.3.2 Diagrama de Classe

O diagrama de classe revela as classes que serão utilizadas para compor a base de dados do sistema e seus atributos o que pode ser observado na figura 8, a seguir.

Destacam-se as classes e os atributos de usuário, time e projeto, classes que são essenciais para o controle do sistema. A classe Sprint Backlog é responsável pelo controle de informações da Sprint dos projetos do time e por meio da Sprint Backlog controla-se a classe de itens que por fim está ligada a classe de tarefas que são controladas pelos usuários.

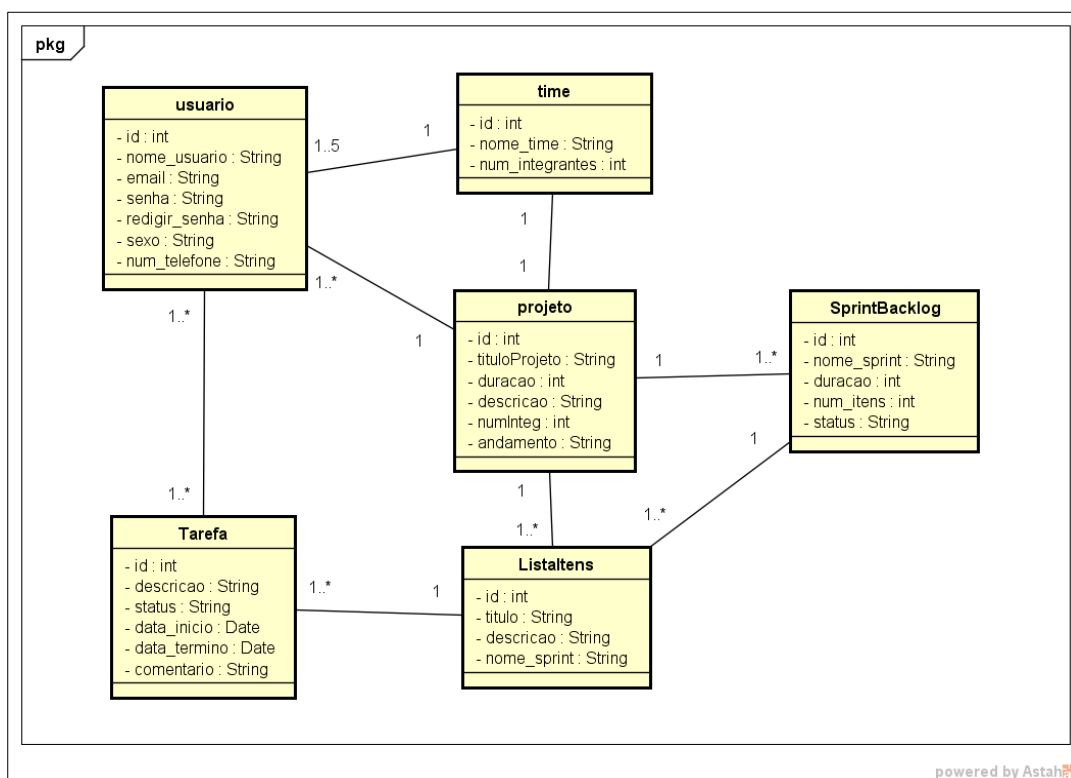


Figura 8: Diagrama de classe do sistema
 Fonte: autoria própria

3.4 Descrição dos principais Casos de Uso da Ferramenta

Neste subtópico serão apresentadas as principais funcionalidades com a utilização dos casos de uso do sistema.

3.4.1 Caso de uso Gerar Lista de Itens

Tabela 3: Caso de uso Gerar Lista de Itens

Nome do caso de uso	Gerar Lista de Itens
Caso de Uso de Geral	Manter Sprint Backlog
Ator Principal	Usuário
Atores Secundários	
Resumo	Este caso de uso descreve as etapas necessárias para adicionar itens que compõem a Sprint Backlog.
Pré-condições	Necessário cadastro realizado na base da Sprint Backlog atual.
Pós-condições	
Ações do Ator	Ações do Sistema
1. Usuário irá clicar no botão "Adicionar Item" na coluna Backlog	
	2. Sistema apresenta a tela de Adicionar Itens.
3. O usuário deve preencher os campos	

do formulário "adicionar item".	
7. Usuário deve clicar no botão "Adicionar"	
	8. O Item é adicionado à coluna "backlog" em forma de cartão.
Restrições/Validações	1. A Sprint selecionada precisa estar com o status de "iniciada".
Fluxo de exceção	

Fonte: autoria própria

3.4.2 Caso de Uso Manter cartões do quadro Scrum

Tabela 4: Caso de Uso Manter cartões do quadro Scrum

Nome do Caso de Uso	Manter Cartões do Quadro Scrum
Caso de Uso Geral	Criar Dashboard, gerar lista de itens.
Ator Principal	Usuário
Atores Secundários	
Resumo	Este caso de uso retrata os passos necessários para adicionar os cartões de tarefas do quadro Scrum.
Pré-condições	A lista da Sprint atual já deverá estar disponível no quadro de tarefas.
Pós-condições	
Ações do Ator	Ações do Sistema
1. O usuário deverá clicar no botão "Adicionar Cartão".	
	2. O sistema exibe uma tela de adicionar cartão.
2. O usuário deverá preencher as informações necessárias.	
3. Depois de preenchidas as informações e em seguida clicar no botão "Adicionar".	
	4. O sistema irá adicionar o cartão na coluna "A fazer" do Dashboard.
Restrições/Validações	
Fluxo de exceção	

Fonte: autoria própria

3.4.3 Caso de uso Gerenciar Quadro Scrum

Tabela 5: Caso de uso Gerenciar quadro Scrum

Nome do Caso de Uso	Manter Gerenciar o Quadro Scrum
Caso de Uso Geral	Criar Dashboard, Manter Sprint Backlog, Gerar Lista de Itens.
Ator Principal	Usuário
Atores Secundários	
Resumo	Este caso de uso descreve os passos necessários para o gerenciamento do quadro de tarefas Scrum até a sua finalização da Sprint.
Pré-condições	O Cartão previamente escolhido deverá estar na coluna "A Fazer".
Pós-condições	
Ações do Ator	Ações do Sistema
1. O usuário irá descrever as tarefas no cartão escolhido, depois clicar no botão "avançar" no cartão.	
	2. O sistema irá avançar o cartão para a etapa "Fazendo".
3. O usuário irá clicar novamente no botão "Avançar".	
	4. O sistema irá avançar o cartão para a etapa "Pronto".
5. O usuário irá certificar que os cartões estão todos na coluna, "Pronto".	
6. O usuário clica no botão "finalizar" para que o sistema fique sem nenhuma tarefa no dashboard.	
Restrições/Validações	
Fluxo de Exceção -	

Fonte: autoria própria

3.5 Modelagem Conceitual de dados

A seção a seguir demonstra a base de dados com a utilização da modelagem conceitual do sistema (figura 9).

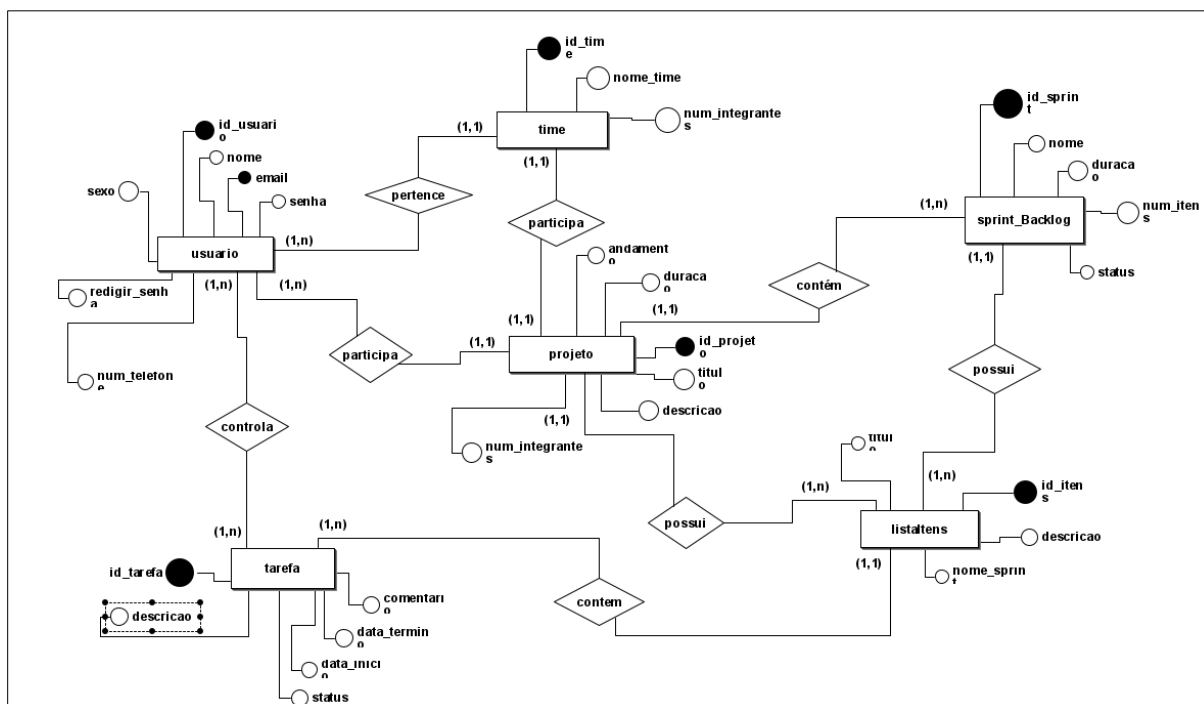


Figura 9: Modelagem Conceitual de Dados do sistema
Fonte: autoria própria

O modelo conceitual ilustra as entidades correspondentes as tuplas (“tabelas”) pertencentes ao banco de dados, destacamos as entidades, usuário, projeto e time. O usuário pertence a um time e um time participa de um projeto. As demais entidades são a Sprint Backlog que está ligado a um projeto. O projeto possui uma lista de itens, da mesma maneira uma Sprint possui uma lista de itens. Cada item pode ter uma ou mais tarefas controladas pelos usuários.

3.6 Modelagem Lógica de dados

Para melhor análise e visualização quanto à construção do banco de dados do sistema, a figura 10 representa a base de dados com todas as tabelas por meio da utilização da modelagem lógica de dados do sistema.

O modelo lógico de dados ilustra o contexto mais fiel ao banco de dados do sistema como, por exemplo, as chaves primárias e chaves estrangeiras dos seus relacionamentos.

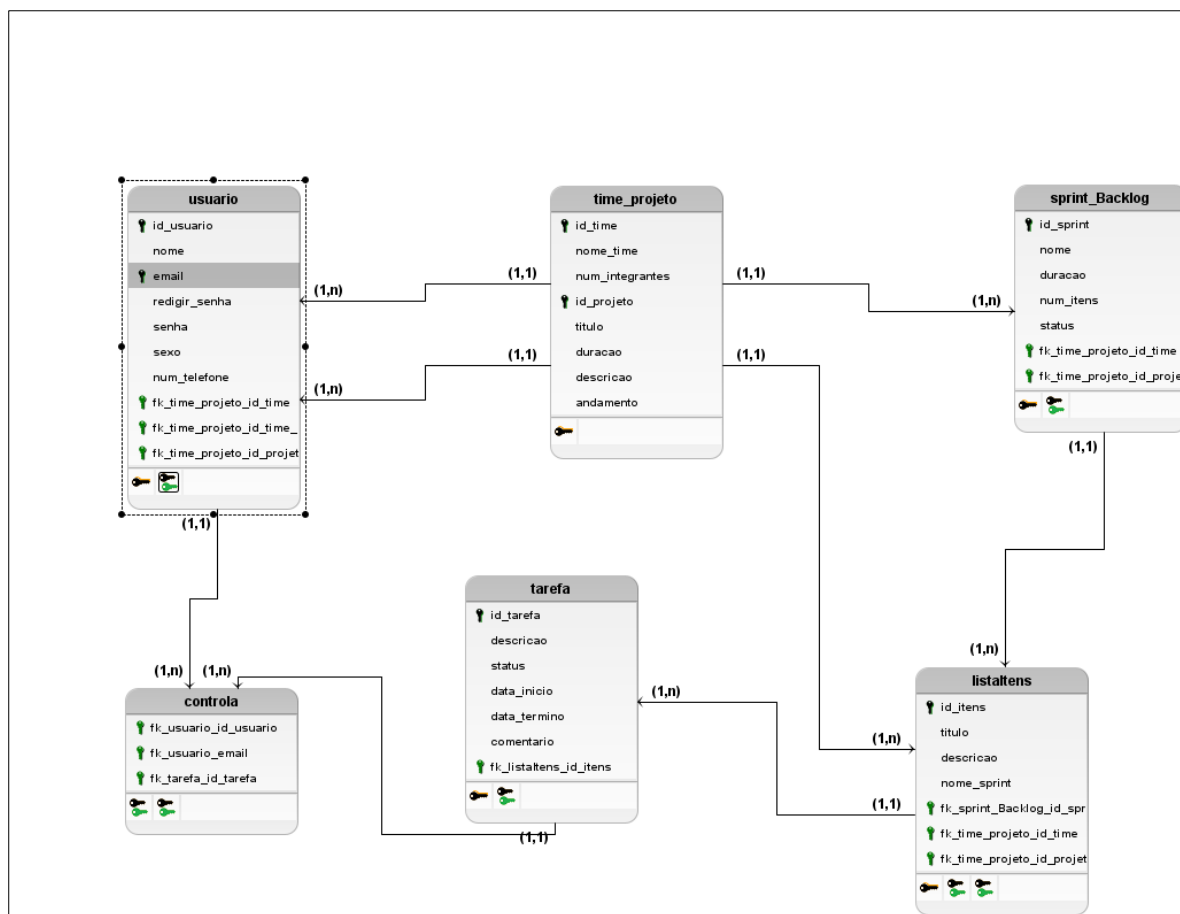


Figura 10: Modelagem Lógica de Dados do sistema
Fonte: autoria própria

3.7 Principais trechos de codificação

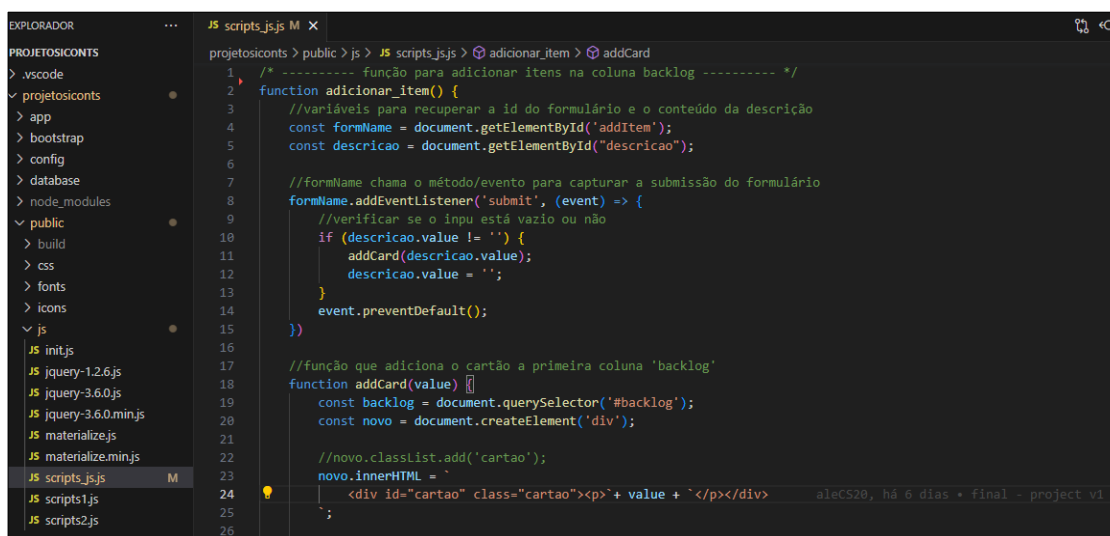
Nesta seção são apresentados alguns dos trechos mais importantes dos casos de usos citados na seção 3.4 deste capítulo.

Os tópicos seguintes demonstram por meio de imagens com alguns dos trechos codificados e devidamente comentados com as suas devidas aplicações no sistema.

3.7.1 Funcionalidade Gerar Lista de Itens

Conforme as especificações do Caso de Uso “Gerar Lista de Itens”, discutido na sessão 3.4.1, a partir desta funcionalidade o sistema permite criar e adicionar um novo Item da Lista do Backlog da Sprint.

Por meio do botão "adicionar item" da tela de dashboard o usuário adiciona um novo item que corresponde a um cartão de informação com a descrição do item a ser trabalhado pela equipe.



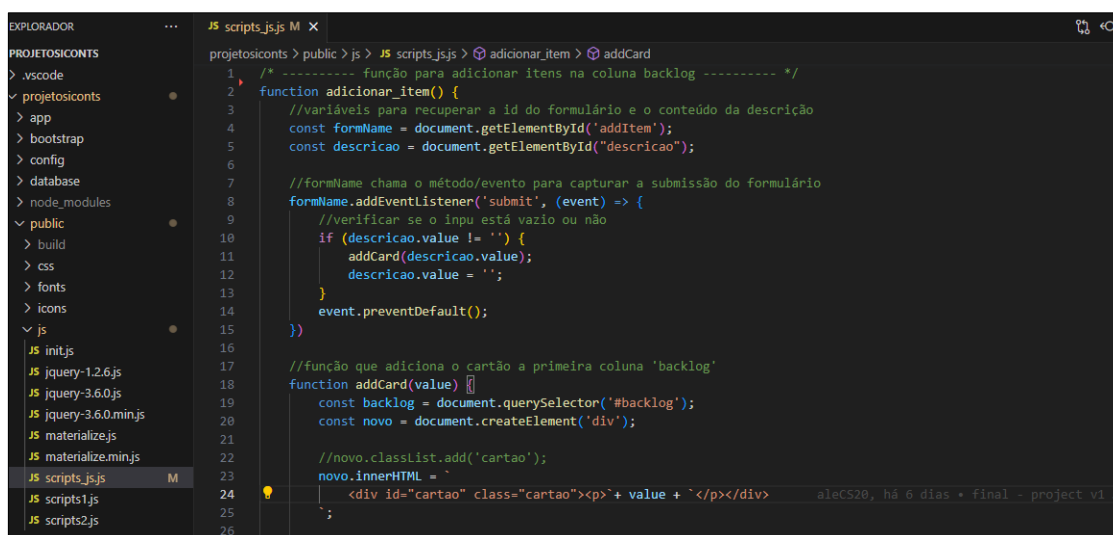
```
EXPLORADOR ... JS scripts.js.js M X
PROJETOSICONTS
> .vscode
> projetosicons
  > app
  > bootstrap
  > config
  > database
  > node_modules
  > public
    > build
    > css
    > fonts
    > icons
    > js
      JS init.js
      JS jquery-1.2.6.js
      JS jquery-3.6.0.js
      JS jquery-3.6.0.min.js
      JS materialize.js
      JS materialize.min.js
      JS scripts.js.js M
      JS scripts1.js
      JS scripts2.js
  > logs

projetosicons > public > js > JS scripts.js.js > adicionar_item > addCard
1 /* ----- função para adicionar itens na coluna backlog ----- */
2 function adicionar_item() {
3   //variáveis para recuperar a id do formulário e o conteúdo da descrição
4   const formName = document.getElementById('addItem');
5   const descricao = document.getElementById("descricao");
6
7   //formName chama o método/evento para capturar a submissão do formulário
8   formName.addEventListener('submit', (event) => {
9     //verificar se o input está vazio ou não
10    if (descricao.value != '') {
11      addCard(descricao.value);
12      descricao.value = '';
13    }
14    event.preventDefault();
15  })
16
17  //função que adiciona o cartão a primeira coluna 'backlog'
18  function addCard(value) {
19    const backlog = document.querySelector("#backlog");
20    const novo = document.createElement('div');
21
22    //novo.classList.add('cartao');
23    novo.innerHTML = `
24      <div id="cartao" class="cartao"><p>+ value + `</p></div>
25    `;
26  }
```

Figura 11: Trecho de código para adicionar um novo item ao backlog
Fonte: próprio autor

A figura 11 ilustra o trecho de código (JavaScript) que corresponde a adicionar um cartão na coluna de "Backlog" do dashboard. A função "adicionar_item()" (linha 2) permite captar o nome do formulário "addItem" e o campo de input "descricao", (linhas 4 e 5) que serão devidamente adicionados as variáveis constantes, "formName" e "descricao".

A linha 8 corresponde ao evento de clique "addEventListener" quando o usuário clica no botão "adicionar" no formulário de adicionar item, o evento verifica se input não está vazio a fim de captar a descrição informada pelo usuário para que o conteúdo seja adicionado ao cartão.



```
1  /* ----- função para adicionar itens na coluna backlog ----- */
2  function adicionar_item() {
3      //variáveis para recuperar a id do formulário e o conteúdo da descrição
4      const formName = document.getElementById("addItem");
5      const descricao = document.getElementById("descricao");
6
7      //formName chama o método/evento para capturar a submissão do formulário
8      formName.addEventListener('submit', (event) => {
9          //verificar se o input está vazio ou não
10         if (descricao.value != '') {
11             addCard(descricao.value);
12             descricao.value = '';
13         }
14         event.preventDefault();
15     });
16
17     //função que adiciona o cartão a primeira coluna 'backlog'
18     function addCard(value) {
19         const backlog = document.querySelector("#backlog");
20         const novo = document.createElement('div');
21
22         //novo.classList.add('cartao');
23         novo.innerHTML = `
24         <div id="cartao" class="cartao"><p>+ value + `</p></div>
25         `;
26     }
```

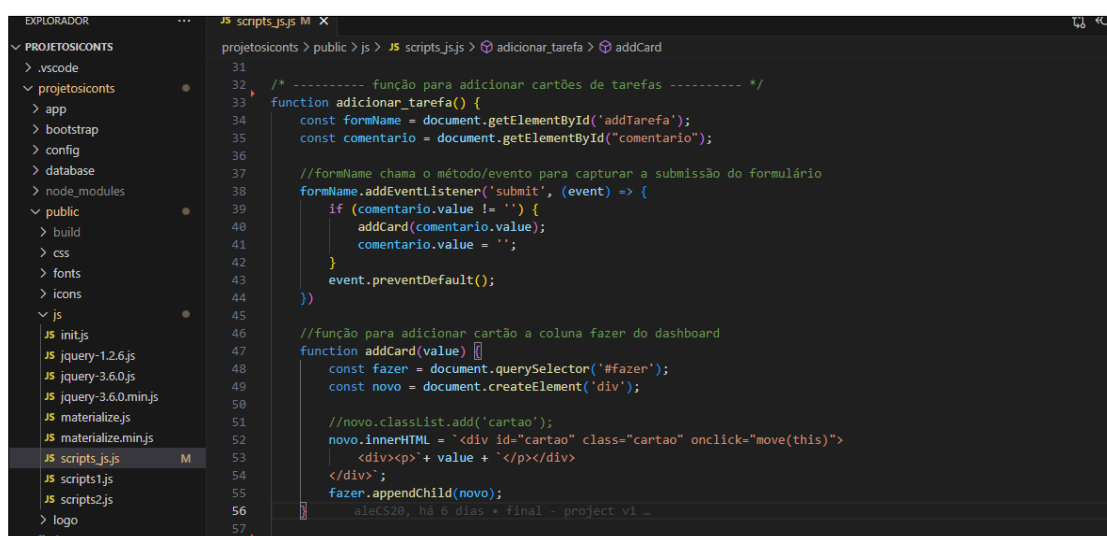
Figura 12: Trecho de código que demonstra como adicionar um novo item a coluna backlog.
Fonte: próprio autor

figura 12 corresponde ao trecho que insere o cartão na coluna backlog, a função "addCard(value)" (linha 18), possui o parâmetro "value" cuja descrição informada no input do formulário. O trecho que inicia em "newCard.innerHTML" (linha 23 até a linha 25) permite que na tela de dashboard, coluna backlog seja adicionada o cartão de item com a descrição digitada no formulário.

3.7.2 Funcionalidade Gerenciar Cartões do Quadro Scrum

Após as especificações do Caso de Uso “Gerenciar Cartões do Quadro Scrum”, discutido na sessão 3.4.2, a funcionalidade a seguir permitirá ao usuário do sistema criar e adicionar cartões de tarefas no Quadro Scrum.

Através do botão "adicionar tarefa" o usuário será levado a um formulário modal que após o seu preenchimento, adiciona um novo cartão de tarefa. Conforme a figura 13 o código correspondente a adicionar um novo cartão a ser devidamente adicionado e gerenciado pelo usuário a partir da coluna "A Fazer" do Quadro Scrum.



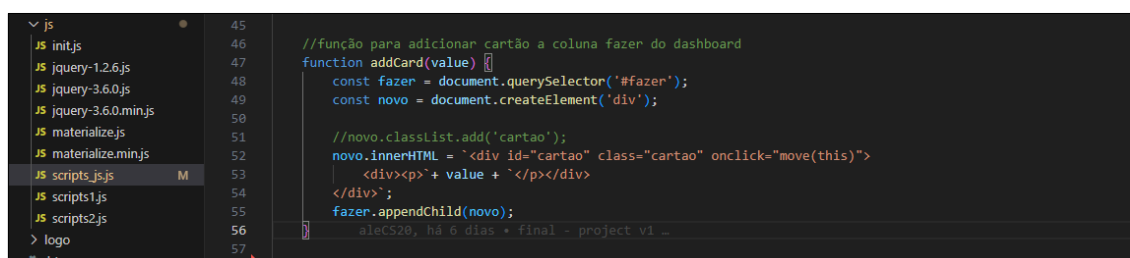
```

31
32
33 /* ----- função para adicionar cartões de tarefas ----- */
34 function adicionar_tarefa() {
35     const formName = document.getElementById('addTarefa');
36     const comentario = document.getElementById("comentario");
37
38     //formName chama o método/evento para capturar a submissão do formulário
39     formName.addEventListener('submit', (event) => {
40         if (comentario.value != '') {
41             addCard(comentario.value);
42             comentario.value = '';
43         }
44         event.preventDefault();
45     });
46
47     //função para adicionar cartão a coluna fazer do dashboard
48     function addCard(value) {
49         const fazer = document.querySelector("#fazer");
50         const novo = document.createElement('div');
51
52         //novo.classList.add('cartao');
53         novo.innerHTML = `<div id="cartao" class="cartao" onclick="move(this)">
54             <div><p> + value + `</p></div>
55         </div>`;
56         fazer.appendChild(novo);
57         aleCS20, há 6 dias + final - project v1
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Figura 13: trecho de código correspondente a adicionar um cartão de tarefas ao dashboard.
Fonte: próprio autor

Semelhante a adicionar um item à função "adicionar_tarefa()" (linha 33) permite captar o nome do formulário e o comentário digitado pelo usuário (linhas 34 e 35). O evento "addEventListener" permite verificar se o input captado não está vazio a fim de ser adicionado a variável "comentario" (linhas 39 a 41).



```

45
46 //função para adicionar cartão a coluna fazer do dashboard
47 function addCard(value) {
48     const fazer = document.querySelector("#fazer");
49     const novo = document.createElement('div');
50
51     //novo.classList.add('cartao');
52     novo.innerHTML = `<div id="cartao" class="cartao" onclick="move(this)">
53         <div><p> + value + `</p></div>
54     </div>`;
55     fazer.appendChild(novo);
56     aleCS20, há 6 dias + final - project v1
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

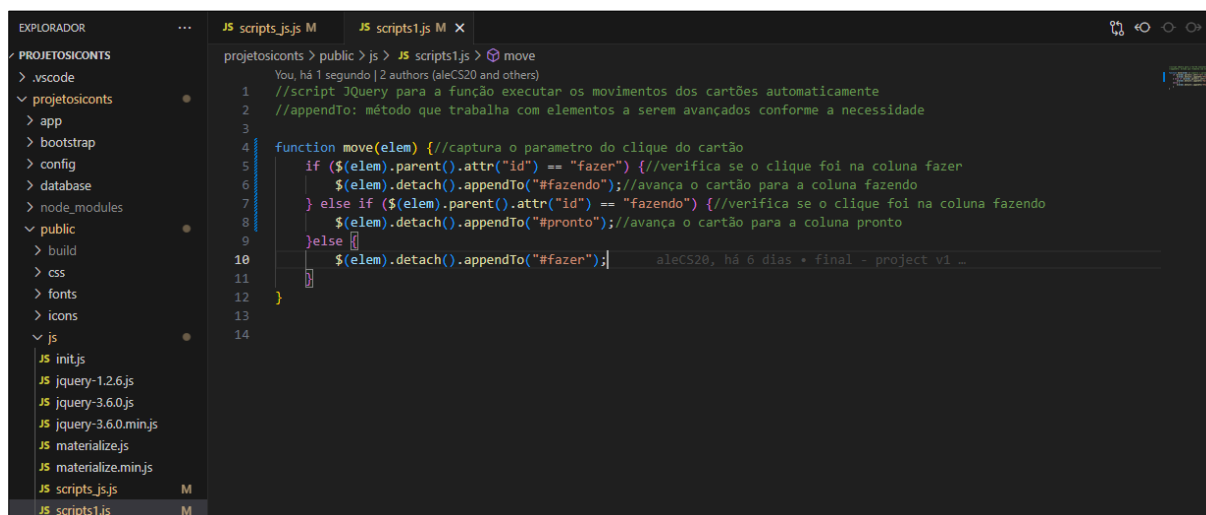
Figura 14: trecho de código que adiciona um cartão de tarefas ao dashboard.
Fonte: próprio autor

A figura 14 indica o trecho do código, e a função "addCard(value)" (linha 47) permite adicionar o cartão de tarefa a partir da coluna a fazer (afazer). A partir do comando "newCard.innerHTML" irá construir o cartão e adicionará na div "status todo" (linhas 52 a 54).

3.7.3 Funcionalidade Gerenciar Quadro Scrum

O Caso de Uso "Gerenciar Quadro Scrum", que foi definido na sessão 3.4.3, a partir desta funcionalidade o sistema poderá ser devidamente controlado e gerenciado pelo usuário, assim como controlar as demais funcionalidades tanto dos cartões de tarefas como também dos itens da lista da Backlog da Sprint.

A figura 15 indica o controle a ser realizado pelos cartões, ou seja, permite o seu avanço pelo dashboard nas colunas "fazer", "fazendo" e "pronto". A função "move" ao ser acionado por um clique no cartão captura o seu elemento por meio do parâmetro "elem" (linha 4). O método "parent" da função verifica se o elemento clicado por meio de sua id a está na coluna "fazer" (linha 5), e permite que haja troca de local e avança o cartão para a div ou coluna seguinte "fazendo" (linha 6), até que o próximo passo seja a coluna "pronto" (linha 8).



```
EXPLORADOR    ...    JS scripts_js.js M    JS scripts1.js M X
PROJETOSICONS
> .vscode
> projetosicons
  > app
  > bootstrap
  > config
  > database
  > node_modules
  > public
    > build
    > css
    > fonts
    > icons
    > js
      JS init.js
      JS jquery-1.2.6.js
      JS jquery-3.6.0.js
      JS jquery-3.6.0.min.js
      JS materialize.js
      JS materialize.min.js
      JS scripts_js.js M
      JS scripts1.js M

projetosicons > public > js > JS scripts1.js > move
You, há 1 segundo | 2 authors (aleCS20 and others)
1 //script JQuery para a função executar os movimentos dos cartões automaticamente
2 //appendTo: método que trabalha com elementos a serem avançados conforme a necessidade
3
4 function move(elem) { //captura o parametro do clique do cartão
5   if ($(elem).parent().attr("id") == "fazer") { //verifica se o clique foi na coluna fazer
6     $(elem).detach().appendTo("#fazendo"); //avança o cartão para a coluna fazendo
7   } else if ($(elem).parent().attr("id") == "fazendo") { //verifica se o clique foi na coluna fazendo
8     $(elem).detach().appendTo("#pronto"); //avança o cartão para a coluna pronto
9   } else {
10    $(elem).detach().appendTo("#fazer");
11  }
12 }
13
14
```

Figura 15: Trecho de código referente aos avanços dos cartões no dashboard.
Fonte: próprio autor

3.8 Principais telas do Sistema

Nas seções seguintes são apresentadas as principais telas do sistema e algumas funcionalidades importantes, assim como a descrição de cada uma dentro do sistema.

3.8.1 Tela de cadastro de Item da Sprint Backlog

A tela de cadastro dos itens da Sprint Backlog, figura 16, apresenta as informações a serem trabalhadas na Sprint atual da equipe. Nesta tela o usuário pode escolher qual Sprint está sendo trabalhada no momento pela equipe.

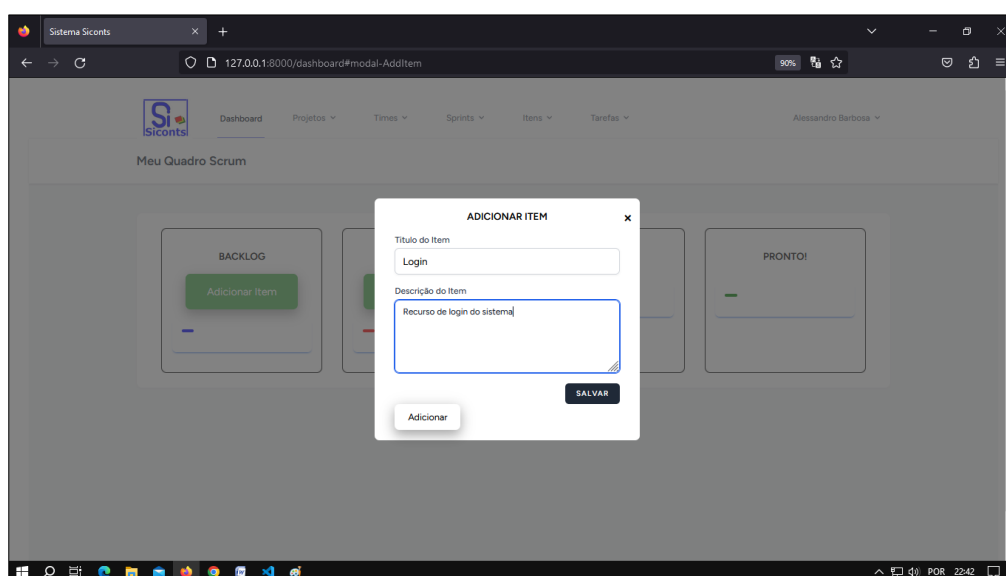


Figura 16: Tela de Cadastro de Itens
Fonte: autoria própria

Essa escolha definirá para quais itens no momento serão agregados a Sprint Backlog no sistema. Após escolher a Sprint o usuário irá definir um título para o item, campo “Título do Item”, feito isso o usuário informará uma breve descrição sobre o item que está sendo cadastrado.

Finalizando o usuário irá clicar no botão “Adicionar”, que irá inserir o item na primeira coluna (Itens) do Dashboard do Quadro Scrum, em seguida clicará no botão “Salvar” para que as informações sejam salvas na base de dados.

3.8.2 Tela de Cadastro de cartão de tarefas

Semelhante ao cadastro de Itens da Lista do Backlog os cartões de tarefas dos usuários são adicionados bastando um clique no botão de “adicionar tarefa”, conforme a figura 17 a seguir.

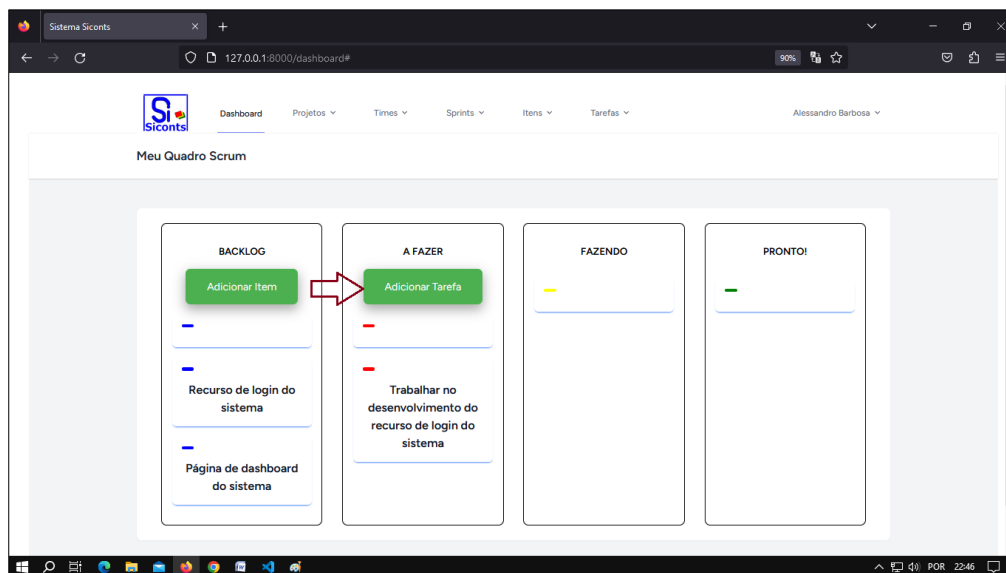


Figura 17: Tela de Dashboard para acesso à tela de Adicionar Tarefas
Fonte: autoria própria

O usuário terá acesso à tela de “cadastro de tarefas”, figura 16 para que sejam preenchidas informações a serem armazenadas no banco de dados do cartão de tarefa a ser controlado pelo usuário.

Na tela de Adicionar Cartão figura 18, o usuário escolhe pelo título o item a ser relacionado com o cartão, uma vez escolhido o título o usuário preenche as informações subsequentes iniciando pelo campo “comentário”.

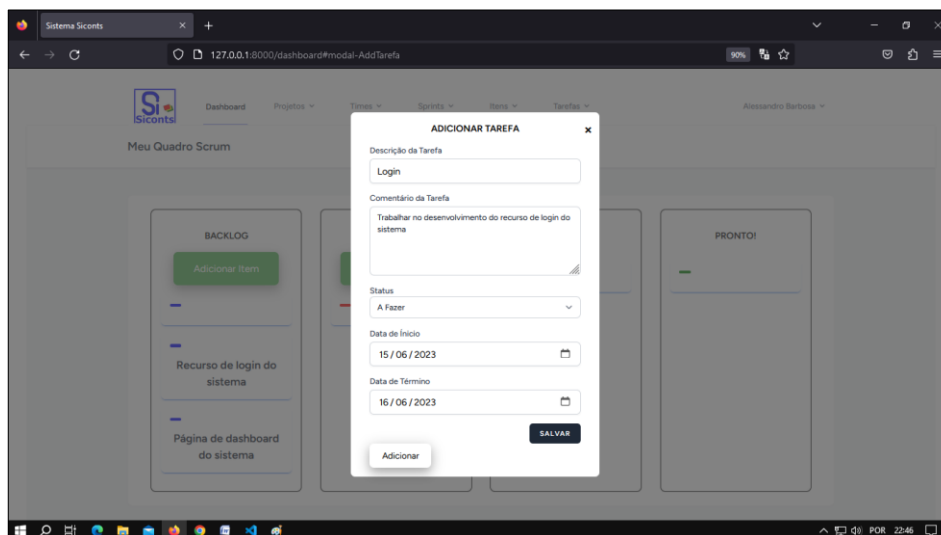


Figura 18: Tela de Cadastro de Tarefas
Fonte: autoria própria

No menu dropdown “status” o usuário escolherá entre as opções “a fazer”, “fazendo”, “concluído”, esses status definem a qual coluna ou etapa o cartão de tarefa se encontra, inicialmente como “A Fazer”.

Por fim os campos “data início” e “data término” são definidos pelo usuário a fim de ser controlada a duração de determinada tarefa. Concluindo o usuário irá adicionar o cartão no dashboard. O cartão será inserido na coluna “A Fazer” do Quadro Scrum.

3.8.3 Tela de Dashboard (Quadro Scrum) do Sistema

Após a realização de login no sistema o usuário será encaminhado até a tela de Dashboard do sistema, conforme figura 19. Nesta tela o usuário tem acesso a todos os recursos do sistema, a principal delas é o Quadro Scrum que permitirá ao usuário total controle das tarefas a serem realizadas pelos membros da equipe Scrum.

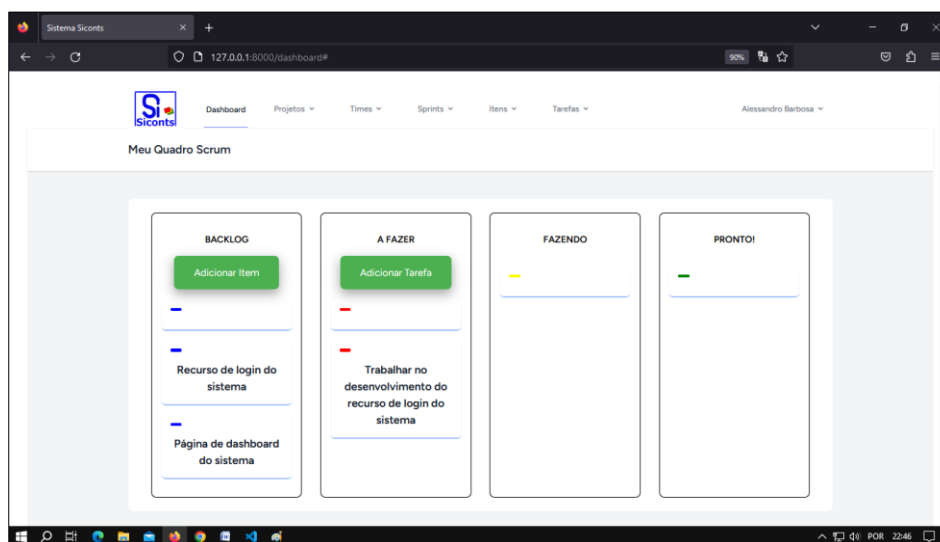


Figura 19: Tela de Dashboard do Sistema
Fonte: autoria própria

Após a criação e a inicialização da Sprint atual, uma lista de itens da Sprint é devidamente mostrada na primeira coluna do Dashboard “Itens”, mencionado no item 3.8.1, que por meio deste os usuários observam essa lista e criam os seus cartões de controle de tarefas.

O controle de tarefas no Quadro Scrum ocorre conforme a definição dos dias ou de acordo com a necessidade do usuário. Para os avanços dos cartões automatizados, basta um clique no botão de controle “avançar”, conforme imagem a seguir, o cartão irá avançar para a coluna subsequente, uma vez que todos os cartões criados na coluna “A fazer” estiverem sido movimentados até a coluna definida como, “Pronto”, a Sprint atual poderá ser finalizada e uma nova Sprint pode ser inicializada.

3.8.4 Resultados obtidos

A ferramenta (Scrum Conts) desenvolvida neste projeto teve como finalidade apresentar um sistema mais simples e enxuto para o gerenciamento e compartilhamento de tarefas para pequenas equipes Scrum. Os softwares tradicionais que contribuíram para a idealização deste trabalho foram o IceScrum, Jira Software e Scrumwise e fazem uso do quadro de tarefas estilo Kanban, os que possuem uma característica em comum, os movimentos manuais dos cartões de tarefas para o trabalho da equipe.

Uma breve comparação entre as ferramentas utilizadas como referência e a ferramenta (Scrum Conts) pode ser conferida na tabela 6 a seguir, nesta pode-se observar que dentre as principais características abordadas destaca-se o modo como é realizado e conferido o status dos cartões ou simplesmente os movimentos dos cartões de tarefas das ferramentas referenciadas. Essa característica foi o principal objeto de estudo para o desenvolvimento da ferramenta (Scrum Conts), destacando como podem ser realizados os status dos cartões apenas com um clique.

Tabela 6: Comparativa das Ferramentas

COMPARATIVO DAS FERRAMENTAS				
	FERRAMENTAS			
Características	Protótipo	Jira Software	IceScrum	Scrumwise
Quadro Scrum/Kanban	Sim	Sim	Sim	Sim
Versão Gratuita	Sim	Até 10 usuários	Teste 14 dias	Versão de teste
Gerenciamento de Projetos	Sim	Sim	Sim	Sim
Idioma Português	Sim	Sim	Não	Não
Movimentos Automáticos dos cartões	Sim	Não	Não	Não
Itens do Backlog da Sprint	Sim	Sim	Sim	Sim

Fonte: autoria própria

Em um primeiro contato com as ferramentas citadas neste trabalho observou-se certa dificuldade em encontrar ou mesmo realizar operações básicas quanto ao gerenciamento das tarefas e o quadro Kanban. Isso acaba sendo comum em ferramentas mais completas e com mais recursos oferecidos aos usuários em geral.

O Sistema (Scrum Conts) permite maior flexibilidade de acesso aos recursos essenciais de um sistema de gerenciamento de tarefas, sua interface simples auxilia aos usuários terem melhor controle e compartilhamento de seu quadro de tarefas com toda a equipe Scrum, além do recurso de movimentação automático dos cartões e seus status.

Por fim este trabalho pode ser utilizado como material de apoio, além da disponibilização de seu código fonte para futuras melhorias e demais contribuições que se encontram disponíveis no Github do projeto, conforme o enlace de endereço a seguir.

<https://github.com/aleCS20/ProjetoSiconts.git>

4. CONSIDERAÇÕES FINAIS

O presente capítulo apresenta os tópicos correspondentes aos riscos e dificuldades encontradas, lições aprendidas, assim como os trabalhos futuros e a conclusão final do trabalho.

4.1 Riscos e Dificuldades

Nesta seção destacam-se as dificuldades e riscos que foram detectados durante o desenvolvimento deste trabalho.

1. Durante o desenvolvimento da ferramenta (Scrum Conts) ocorreu à necessidade de aprendizagem da tecnologia com o framework Laravel, e que em alguns momentos ocorreram fatores de maior dificuldade na sua aplicabilidade em relação à ferramenta desenvolvida;
2. Após a conclusão da ferramenta houve a necessidade de hospedagem e disponibilização do software na nuvem, dificuldade ocorrida em encontrar um serviço devidamente adequado para realização de hospedagem do sistema na internet e disponibilizá-lo para os usuários e empresas;
3. O principal risco identificado foi o grau de complexidade no desenvolvimento do sistema, assim como suas funcionalidades para diversos usuários e o compartilhamento de informações.

4.2 Lições Aprendidas

Com o desenvolvimento deste trabalho pode-se destacar as lições aprendidas principalmente no conhecimento em novas tecnologias e ferramentas empregadas no trabalho, conforme a seguir.

O conhecimento de alguns frameworks novos, em particular o Laravel, seus principais recursos e aplicações para diversos contextos de aplicações de sistema, em particular a ferramenta desenvolvida. A posterior dificuldade encontrada quanto ao grau de conhecimento mais profundo que foi aplicado durante o desenvolvimento da ferramenta.

Outros frameworks aprendidos e devidamente explorados no transcorrer do desenvolvimento da ferramenta como o Tailwind e o Materialize que trabalham com a linguagem CSS e renderizam de forma sucinta o Front-end do sistema.

A experiência adquirida em como disponibilizar a ferramenta na internet que de acordo com as características que a aplicação necessitava para o que os usuários da web possam acessá-lo sem dificuldades.

4.3 Trabalhos Futuros

Como trabalhos futuros tem-se a intenção de realizar melhorias, acrescentar funcionalidades e principalmente o desenvolvimento de uma versão para uso em dispositivos móveis, ampliando assim o uso da ferramenta no mercado, concluindo a sua disponibilização em um serviço web para acesso aos usuários. Além destas características a serem exploradas para trabalhos futuros destacam-se também:

- Funcionalidade de relatórios gerais de projetos, usuários, sprints e itens do sistema;
- Permitir a utilização de gráficos de Burndown do sistema;
- Adição de outros recursos nos cartões de tarefas como imagem do usuário, gráfico de progressão das tarefas, anexos de arquivos;
- Disponibilizar a ferramenta na nuvem.

4.4 Conclusão

O problema descrito na seção 1.1 foi demonstrado na prática nas seções 3.1 a 3.8 com o desenvolvimento de um sistema para tratar os itens mencionados, bem como os objetivos principais abordados na seção 1.3, 1.3.1 e 1.3.2 que foram devidamente atendidos.

A aplicação desenvolvida executa as funcionalidades principais descritas durante o processo de análise e em seguida predeterminadas de maneira satisfatória a fim de demonstrar todos os recursos da ferramenta, além das funcionalidades de compartilhamento e controle das informações das tarefas da equipe Scrum.

A ferramenta (Scrum Conts) oferece uma maneira customizada de movimento dos cartões de uma fase para a outra, característica não encontrada nas pesquisas em trabalhos correlatos. Esperamos que todos tirem proveito máximo da mesma.

REFERÊNCIAS

ANDREW, Rachel; BIDEELLMAN, Eric. **Usando a API HTML5 Drag and Drop**. Web.Dev. atualizado em 30 ago 2021. Disponível em: < <https://web.dev/drag-and-drop/> > acesso em 10 mar. 2023.

ATLASSIAN. **Recursos do Jira Software**. Disponível em: <<https://www.atlassian.com/br/software/jira/features>>. Acesso em 02 jul 2022.

AUDY, Jorge. **Scrum 360: um guia completo e prático de agilidade**. Casa do Código, 2015.

FLOWBITE. Introduction. Disponível em <<https://flowbite.com/docs/getting-started/introduction/>>. Acesso em 04 abr 2023.

GOULART, Gabriel. **Fazendo um formulário modal com Javascript, HTML e CSS**. Portal GSTI, atualizado em 06 abr 2018. Disponível em. <<https://www.portalgsti.com.br/2018/04/fazendo-um-formulario-modal-com-javascripthtml-e-css.html>>. Acesso em 04 abr 2023.

HTML Drag and Drop API. W3 Schools. Disponível em < https://www.w3schools.com/html/html5_draganddrop.asp>. Acesso em 02 fev 2023.

ICESCRUM. **Tudo o que você precisa para o seu gerenciamento de projetos Agile**. Disponível em: <<https://www.icescrum.com/features/>>. Acesso em 02 jul 2022.

LARAVEL. **Documentação**. Disponível em: <<https://laravel.com/docs/10.x>>. Acesso em 06 mar 2023.

LUCIANO. **Blade Engine: Utilizando templates no Laravel**. Devmedia, publicado em 2016. Disponível em. <<https://www.devmedia.com.br/blade-engine-utilizando-templates-no-laravel/36749>>. Acesso em 13 mar 2023.

MARTINS, L. de S. **Scrum framework e sua usabilidade com a ferramenta de princípios ágeis, Trello**. Universidade de Araraquara, 2017.

MATERIALIZE. **Uma estrutura de front-end responsiva moderna baseada no Design de Material**. Disponível em: < <https://materializecss.com/>>. Acesso em 01 abr 2023.

MOZILLA. **Drag and Drop**. Disponível em < https://developer.mozilla.org/pt-BR/docs/Web/API/HTML_Drag_and_Drop_API >. Acesso em 01 abr 2022.

SABBAGH, Rafael. **Scrum: gestão ágil para projetos de sucesso**. Casa do Código, 2022.

SCRUMWISE. **Features Scrumwise**. Disponível em: <<https://www.scrumwise.com/features.html>>. Acesso em 02 jul 2022.

SEGUNDO, Emir. **Quadro Scrum: o que você deve saber para atuar em um projeto de software com este framework**. Blog Impulso Network. 2019. Disponível em: <<https://blog.impulso.network/quadro-scrum-o-que-voce-deve-saber/>>. Acesso em 20 abr 2022.

SOMMERVILLE, Ian. **Engenharia de Software**. 10 Edição, cap 3, p59-60, São Paulo: Pearson, 2019.

STRACK, Everton. **Drag and Drop API: Introdução**. EvertonStrack, em 13 jun 2020. Disponível em < <https://evertonstrack.com.br/drag-and-drop-api/>>. Acesso em 10 abr 2023.

STRACK, Everton. **Drag and Drop API: Aprendendo na prática**. EvertonStrack, em 18 jul 2020. Disponível em < <https://evertonstrack.com.br/drag-and-drop-api-na-pratica/>>. Acesso em 10 abr 2023.

SUTHERLAND, Jeff; SCHWABER, Ken. **O guia do Scrum: o guia definitivo para o Scrum: as regras do jogo**. Creative Commons, 2020.

TAILWINDCSS. **Biblioteca de componentes Flowbite - Tailwind CSS**. Disponível em: <<https://flowbite.com/docs/getting-started/introduction/>>. Acesso em 02 abr 2023.

APÊNDICES

Tela de Apresentação do Sistema

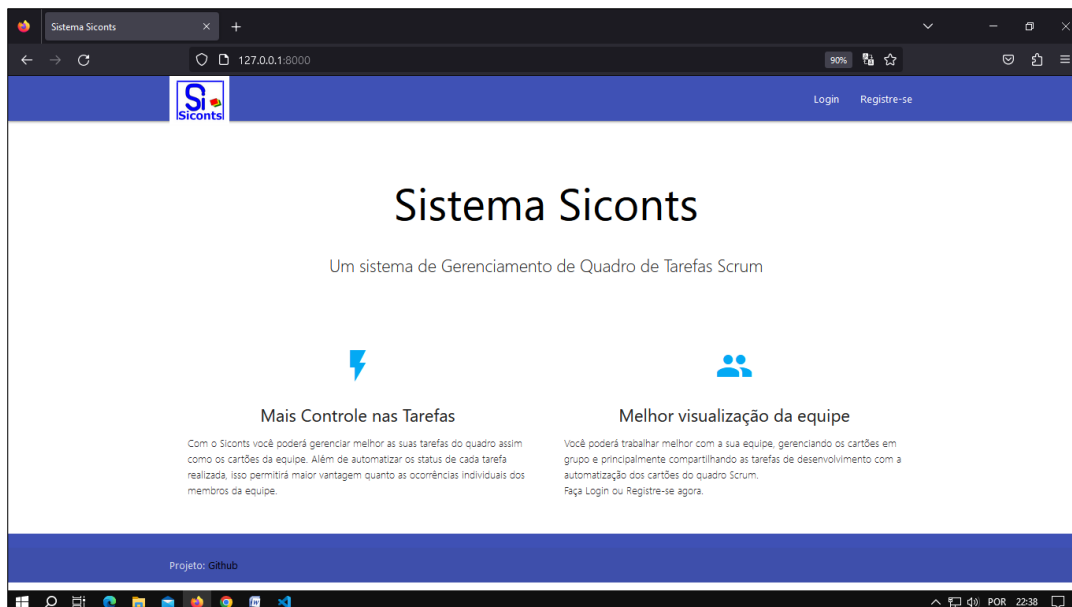


Figura 20: Tela de apresentação do sistema
Fonte: autoria própria

Tela de Registro de Usuário do Sistema

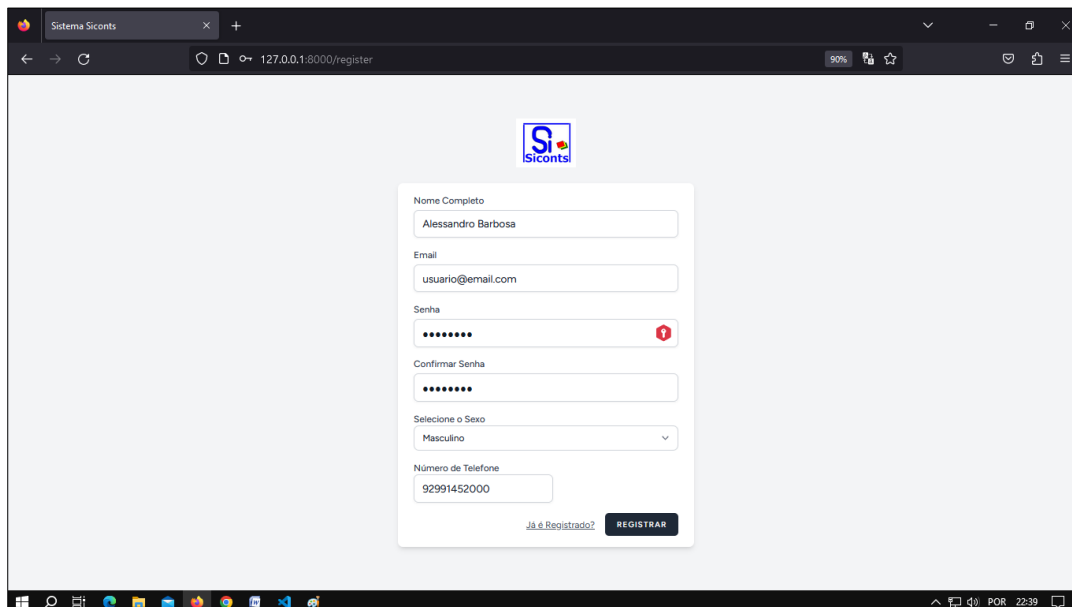


Figura 21: Tela de Registro de Usuário do sistema
Fonte: autoria própria

Tela de Login de Usuário do Sistema

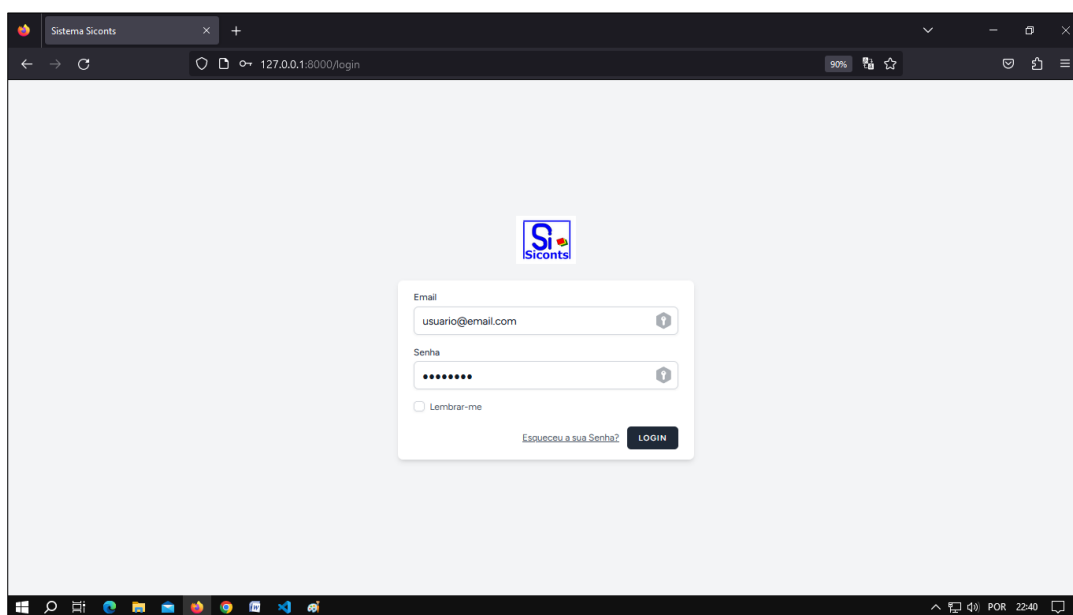


Figura 22: Tela de Login de Usuário do Sistema
Fonte: autoria própria