



Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

Grata - Gerenciamento de Reuniões e ATAs

Autor: Victor Hugo Lopes Mota
Orientador: Dr. Wander Cleber Maria Pereira da Silva

Brasília, DF
2019



Victor Hugo Lopes Mota

Grata - Gerenciamento de Reuniões e ATAs

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Dr. Wander Cleber Maria Pereira da Silva

Brasília, DF

2019

Victor Hugo Lopes Mota

Grata - Gerenciamento de Reuniões e ATAs/ Victor Hugo Lopes Mota. –
Brasília, DF, 2019-

67 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Wander Cleber Maria Pereira da Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2019.

1. Gerenciamento de Reuniões. 2. Otimização. I. Dr. Wander Cleber Maria
Pereira da Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV.
Grata - Gerenciamento de Reuniões e ATAs

CDU 02:141:005.6

Victor Hugo Lopes Mota

Grata - Gerenciamento de Reuniões e ATAs

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Trabalho aprovado. Brasília, DF, 12 de Dezembro de 2019:

**Dr. Wander Cleber Maria Pereira da
Silva**
Orientador

Pr. Rafael Fazzolino
Convidado 1

Pr. Izaías Cabral
Convidado 2

Brasília, DF
2019

Eu dedico este trabalho à minha família, a Deus, amigos e aos professores da FGA que cada á sua maneira, sempre estiveram ao meu lado me dando forças. Em especial, dedico mais uma vez a Minha mãe Juscelina, meu pai Francisco e minha irmã Jéssica que são a base para que me esforce cada vez mais todos os dias. Á Deus, que mesmo que poucos acreditem me dá forças a maneira dele. Meus amigos, sobretudo á Karine Valença, Mateus Farias, Lucas Castro, Francisco Allysson, Pedro Henrique, Filippe Leal, Felipe Marques, Naiara Andrade, Rebecca Louise, Matheus Figueiredo, Gustavo Sabino e Halê Valente. Ao professor Wander e as professoras Milene e Carla que são educadores que sempre se preocupam com os alunos e se esforçam para melhorar como profissionais e pessoas. Dedico também a todos aqueles que mesmo que não foram citados aqui, e que estiveram diretamente ou não presentes na minha graduação.

Agradecimentos

Agradeço principalmente à minha família, minha mãe Juscelina, meu pai Francisco e minha irmã Jéssica por sempre me apoiarem, por nunca desistirem totalmente de mim e por mostrar os valores morais da vida. Além da minha família que é muito maior que apenas meus pais e minha irmã, tem meus amigos conheço que desde o ensino fundamental e aqueles que consegui através da faculdade, todos tiveram uma parte fundamental para eu conseguir chegar até aqui.

Quero agradecer ao meu orientador de TCC, Wander Cleber, por ter aceitado me orientar neste trabalho. Por último, mas não menos importante, quero agradecer à Deus, pois sem fé nada se alcança e com ele tudo é possível.

Resumo

Uma parte importante no desenvolvimentos dos projetos, sejam quais sejam, são as reuniões, pois é nelas que são debatidas os tópicos levantados pelos gerentes de projetos e as pessoas interessadas. Os líderes destes encontros podem estimativam prazos de entregas, e a partir destas, medir o nível de satisfação de seus funcionários. Conduzir uma reunião é antes de tudo, garantir que todos os envolvidos com o projeto estejam por dentro do que esta acontecendo e que possuem pontos de vistas comuns. O problema gerado com o número elevado de reuniões ineficazes, aumento na insatisfação dos envolvidos e o custo para gerir uma reunião já são pontos que levantaram estudos sobre o tema e estes são a base que impulsam este trabalho.

Palavras-chave: Gerenciamento de Reuniões, Desenvolvimento de *Software*, Satisfação, Produtividade.

Abstract

An important part in the development of projects, whatever they may be, are the meetings, because it is in them that the topics raised by the project managers and the interested people are discussed. The leaders of these meetings can estimate delivery times, and from these, measure the level of satisfaction of their employees. Conducting a meeting is first of all, ensuring that everyone involved with the project is aware of what is happening and that they have common points of view. The problem generated by the high number of ineffective meetings, the increase in the dissatisfaction of those involved and the cost to manage a meeting are already points that raised studies on the theme and these are the basis that drive this work.

Key-word: Meeting Management, Software Development, Satisfaction, Productivity.

Lista de ilustrações

Figura 1 – Grupos de processos do Gerenciamento de Projetos, Fonte: Própria. . .	17
Figura 2 – Superposição dos Grupos de Processo. Fonte: PMPDIGITAL (2009). .	18
Figura 3 – Hierarquia em projetos tradicionais. Fonte: Jhonatas (2018).	20
Figura 4 – Exemplo de Caso de Uso. Fonte: Vieira (2015).	20
Figura 5 – Ciclo de Vida SCRUM. Fonte: Fabiane (2016).	22
Figura 6 – Quadro Kanban. Fonte: Lameck (2016).	23
Figura 7 – Papéis Scrum. Fonte: Gonçalves (2016).	24
Figura 8 – Ciclo de Vida <i>Scrum</i> Solo. Fonte: PAGOTTO et al. (2016).	25
Figura 9 – Definição dos Requisitos. Fonte: Própria.	27
Figura 10 – Metodologia. Fonte: Própria	31
Figura 11 – Organograma Senado Federal. Fonte: Senado (2019).	32
Figura 12 – Processo de Desenvolvimento do Grata. Fonte: Própria	36
Figura 13 – Casos de Uso Grata. Fonte: Própria	38
Figura 14 – Diagrama de Classe. Fonte: Própria	40
Figura 15 – Banco de Dados Modelo Conceitual. Fonte: Própria	41
Figura 16 – Banco de Dados Modelo Lógico. Fonte: Própria	42
Figura 17 – Diagrama React	43
Figura 18 – Diagrama Django REST Framework	44
Figura 19 – Login Grata. Fonte: Própria	45
Figura 20 – Página Inicial Grata. Fonte: Própria	46
Figura 21 – Reuniões Que Participo. Fonte: Própria	46
Figura 22 – Ata. Fonte: Própria	47
Figura 23 – Resultados Questionário. Fonte: Própria	47
Figura 24 – Modelagem de Processo Geral 1 Parte 1. Fonte: Própria	56
Figura 25 – Modelagem de Processo Geral 1 Parte 2. Fonte: Própria	57
Figura 26 – Modelagem de Processo Geral 2 Parte 1. Fonte: Própria	58
Figura 27 – Modelagem de Processo Geral 2 Parte 2. Fonte: Própria	59
Figura 28 – Gerenciar Login. Fonte: Própria	60
Figura 29 – Gerenciar Projeto. Fonte: Própria	61
Figura 30 – Marcar Reunião. Fonte: Própria	62
Figura 31 – Pauta Reunião. Fonte: Própria	63
Figura 32 – Questionário. Fonte: Própria	64
Figura 33 – Tela Inicial. Fonte: Própria	66
Figura 34 – Login. Fonte: Própria	67

Lista de tabelas

Tabela 1 – Usuário Administrador	37
Tabela 2 – Usuário Participante	37
Tabela 3 – Requisitos Não-Funcionais	39
Tabela 4 – Ferramentas Auxiliares	44
Tabela 5 – Histórias de Usuário Administrador Parte 1	52
Tabela 6 – Histórias de Usuário Administrador Parte 2	53
Tabela 7 – Histórias de Usuário Administrador Parte 3	54
Tabela 8 – Histórias de Usuário Participante	54
Tabela 9 – Histórias Técnicas	55

Lista de abreviaturas e siglas

GRATA, *Gerenciamento de Reuniões e ATAs*

MAS, *Mindless Accept Syndrome*

NMIL, *Núcleo de Modernização da Informação Legislativa*

TCC, *Trabalho de Conclusão de Curso*

PMBOK, *Project Management Body of Knowledge*

UTFPR, *Universidade Tecnológica Federal do Paraná*

MVC, *Model-View-Controller*

SGM, *Secretária Geral da Mesa*

DGER, *Diretoria Geral*

SINFLEG, *Secretaria de Informação Legislativa*

PRODASEN, *Secretaria de Tecnologia da Informação*

TAP, *Termo de Abertura do Projeto*

PO, *Product Owner*

JAD, *Joint Application Development*

UML, *Unified Modeling Language*

GERTIQ, *Gerenciador de Tíquetes*

Sumário

1	INTRODUÇÃO	14
1.1	Contexto	14
1.2	Justificativa	14
1.3	Problema de Pesquisa	15
1.4	Objetivos	16
1.4.1	Objetivos Gerais	16
1.4.2	Objetivos Específicos	16
2	FUNDAMENTAÇÃO TEÓRICA	17
2.1	Gerenciamento de Projetos	17
2.1.1	Modelo Tradicional - PMBOK	17
2.1.1.1	Áreas de Conhecimento PMBOK	18
2.1.1.2	Casos de Uso	20
2.1.2	Modelo Ágil	21
2.1.2.1	Scrum	22
2.1.2.1.1	Product Owner	24
2.1.2.1.2	Scrum Master	24
2.1.2.1.3	Dev Team	24
2.1.2.2	Scrum Solo	25
2.2	Processo de Desenvolvimento de Software	26
2.2.1	Definição dos Requisitos	26
2.2.1.1	Elicitação dos Requisitos	27
2.2.1.1.1	Entrevista	27
2.2.1.1.2	Observação	28
2.2.1.2	Requisitos Funcionais	28
2.2.1.3	Requisitos Não-Funcionais	28
2.2.2	Linguagem de Software	28
2.2.2.1	Front-end	28
2.2.2.2	Back-end	29
2.2.2.3	Arquitetura de Software	29
2.2.2.4	Model-View-Controller	29
3	METODOLOGIA	31
3.1	A Instituição	31
3.1.1	Senado Federal	31
3.1.2	NMIL	32

3.2	Metodologia de Desenvolvimento	34
3.2.1	Scrum	34
3.2.1.1	Papéis	34
3.2.1.2	Sprints	34
4	PROCESSO DE DESENVOLVIMENTO DE SOFTWARE	36
4.1	Elicitação dos Requisitos	36
4.1.1	Requisitos Funcionais	37
4.1.1.1	Backlog do Produto	38
4.1.1.2	Histórias de Usuário	38
4.1.1.3	Histórias Técnicas	38
4.1.2	Requisitos Não-Funcionais	39
4.1.3	Diagrama de Classe	39
4.1.4	Banco de Dados	41
4.1.4.1	Modelo Conceitual	41
4.1.4.2	Modelo Lógico	42
4.1.5	Front-End	42
4.1.6	Back-End	42
4.1.7	Protótipos	42
4.1.8	Deploy	43
4.1.9	Arquitetura do Projeto	43
4.2	Ferramentas Auxiliares	44
5	RESULTADOS	45
5.1	Ferramenta	45
6	CONSIDERAÇÕES FINAIS	48
	REFERÊNCIAS	49
	APÊNDICES	51
	APÊNDICE A – HISTÓRIAS DE USUÁRIO	52
	APÊNDICE B – HISTÓRIAS TÉCNICAS	55
	APÊNDICE C – FIGURAS	56
	APÊNDICE D – PROTÓTIPOS	60
	APÊNDICE E – MANUAL DO USUÁRIO	65

E.1	Introdução	65
E.2	Instalação	65
E.2.1	Windows	65
E.2.1.1	Docker	65
E.2.2	Ubuntu	66
E.2.2.1	Docker	66
E.2.2.2	Docker-Compose	66
E.3	Como Utilizar?	66
E.3.1	Tela Inicial e Login	66
E.3.2	Perfis de Usuários	67

1 Introdução

Um dos instrumentos mais fundamentais e que são cada vez mais crescentes na vida organizacional de uma empresa são as reuniões. Segundo [Allen \(2016\)](#), uma instituição utiliza em média 15% do tempo coletivo da organização com reuniões.

1.1 Contexto

Encontros institucionais que são improdutivos e sem sentido, é o que [David \(2013\)](#) chama de *Mindless Accept Syndrome* (MAS), que é um tradução livre é a "Síndrome de Aceitação Sem Sentido". [David \(2013\)](#) define o MAS como "um reflexo involuntário em que uma pessoa aceita um convite de reunião sem sequer saber o porquê. Uma doença comum entre o escritório e trabalhadores em todo mundo". Atividades que deveriam ser simples e rápidas se tornam complicadas com várias reuniões para a execução completa delas. Estes encontros não são nenhum ponto de prazer dentro de uma instituição, contudo se os próprios funcionários não conseguem visualizar o sentido da reunião e os tópicos abordados, mostra que a empresa como um todo está fadada ao fracasso.

1.2 Justificativa

As reuniões são criadas para promover o compartilhamento de informações, melhorar a tomada de decisões, promover a resolução de problemas, construir a coesão da equipe e reforçar a cultura organizacional [Leach \(2015\)](#). Por se tratar de um encontro dentro da empresa, segundo [Leach \(2015\)](#), as reuniões podem gerar emoções tanto positivas quanto negativas dentre os participantes da reunião. É possível sair de uma reunião sentindo-se energizado e inspirado, ou afastar-se deste encontro sentindo-se esgotado e desmoralizado.

[Macleod \(2011\)](#) estimou que entre 30% a 60% do tempo gasto em uma reunião é desperdiçado. Um gerente pode passar onze horas por semanas em reuniões e metade desse tempo é improdutivo. Ao realizar uma pesquisa sobre a produtividade das reuniões, [Perlow \(2017\)](#) constatou que 65% dos gerentes entrevistados indicaram que os encontros os impedem de realizar suas próprias atividades no trabalho e 71% alegam que as reuniões são improdutivas e ineficazes. Uma reunião pode render comentários positivos e negativos, contudo segundo [Leach \(2015\)](#), os comentários negativos são os mais pertinentes e são relacionados a estrutura da reunião. Problemas como falta de planejamento, informações de baixa relevância e impacto pouco claro de participação são os fatores que mais compõem negativamente uma reunião.

Rogelberg (2005) examinou as reuniões a partir de duas teorias: capacidade atencional e teoria da ação. Ao aplicar essas práticas, foi constatado que a fadiga diária e a carga de trabalho subjetiva estão relacionadas diretamente as reuniões atendidas. O estudo sugere ainda que a frequência de reuniões é mais importante do que o tempo gasto em reuniões ao longo do dia. Ainda segundo Rogelberg (2005), a "natureza disruptiva dos resultados das reuniões na drenagem dos recursos emocionais ou mentais e fadiga subsequente". A conclusão do estudo foi que tanto a qualidade quanto a quantidade das reuniões são importantes para o bem-estar do funcionário.

As reuniões por mais importantes que sejam, não apresentam de fato o verdadeiro sentido que elas possuem. Como apresentado nos tópicos anteriores, estudos nessa área apresentam diversos problemas nesses encontros com níveis de improdutividade elevados, contudo ao analisar esses estudos é possível propor uma solução computacional que auxilie gerentes e funcionários a não gastarem tanto tempo em reuniões e atingir um maior nível de produtividade. A proposta computacional visa aumentar o engajamento dos participantes, auxiliar o planejamento das reuniões, informações com maiores níveis de relevância e por fim diminuir os altos níveis de improdutividade apresentados pelos autores nos tópicos anteriores.

Ao realizar uma pesquisa de mercado, foi encontrado quatro *softwares* semelhantes ao proposto nesse projeto, sendo eles:

- *Meeting* (gestão de projetos)
- Qualiex
- Eata
- *Google calendar*
- *Microsoft Teams*

Destes cinco *softwares*, os três primeiros além de serem pagos, não possuem o código aberto, dificultando a customização da ferramenta pelo usuário. O *Google calendar* é um *software* gratuito, contudo possui o mesmo problema dos três primeiros, pois não possui um código aberto e não permite uma maior customização por parte do usuário. O *Microsoft Teams* é uma ferramenta que possui funcionalidades gratuitas, mas com a maior parte destas de forma paga.

1.3 Problema de Pesquisa

O crescente problema com reuniões mal gerenciadas seja por gerentes não capacitados, ou por falta da especificação prévia dos tópicos a serem abordados, levam diretamente

a reuniões mal sucedidas e com isso desperdício de tempo e dinheiro. Estima-se que empresas gastam em média US \$ 37 bilhões anualmente em reuniões [Drake \(2014\)](#). O custo real desses encontros impulsionou a universidade de [Harvard \(2016\)](#) a criar uma calculadora que ajuda gerentes a estimarem o verdadeiro custo de um encontro.

Nessa viés e utilizando a justificativa desenvolvida no tópico [1.2](#), é possível se ter o problema de pesquisa. A problemática a ser resolvida neste projeto é: *Como desenvolver um sistema para auxiliar a condução de reuniões em organizações?*

1.4 Objetivos

1.4.1 Objetivos Gerais

O objetivo desse projeto é propor uma solução computacional que auxilie gerentes de qualquer empresa a realizarem seus encontros, e que permita às instituições, por meio de código aberto, a customização do software a maneira que desejarem.

1.4.2 Objetivos Específicos

- Desenvolver um *software* que auxilie gerentes e líderes a terem reuniões mais objetivas;
- permitir que o código do *software* seja aberto;
- disponibilizar a documentação do *software*;
- permitir mecanismos de controle gerencial sobre as reuniões;
- dividir os usuários em duas categorias: gerente e participante da reunião;
- permitir o controle das informações provindas das reuniões;
- guardar os dados da reunião de forma que eles não sejam perdidos;

2 Fundamentação Teórica

2.1 Gerenciamento de Projetos

2.1.1 Modelo Tradicional - PMBOK

Uma metodologia de gerenciamento de projetos no modelo tradicional, de acordo com Kerzner (2009) é o alcance da excelência no controle dos projetos e que se torna impossível sem um processo repetitivo que possa ser utilizado em cada projeto.

No modelo tradicional, um dos modelos utilizados é o definido no PMBOK (2012). Jairson (2003) define gerenciamento de projetos como uma interação de ações, em que a falta de alguma ação, pode afetar também outras áreas. Essas interações podem levar a mudanças no projeto, dependendo do seu tamanho, e mudanças quase sempre afetam o prazo de entrega do projeto, e elevam os custos. O PMBOK (*Project Management Body of Knowledge*) possui cinco grupos de processos, como pode ser na Figura 1:

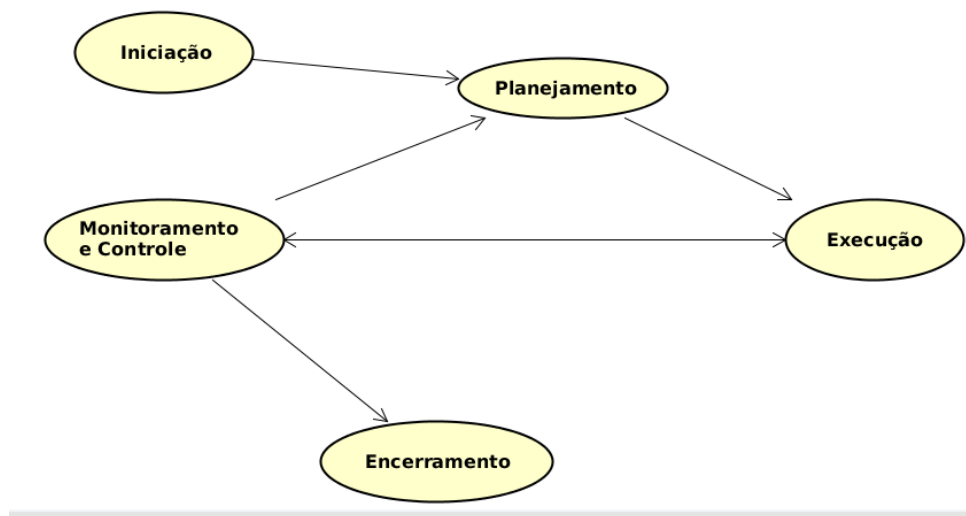


Figura 1 – Grupos de processos do Gerenciamento de Projetos, Fonte: Própria.

A definição desses grupos de processos podem ser vistos a seguir:

- **Processos de Iniciação** – reconhecer que um projeto ou fase deve começar e se comprometer para executá-lo(a);
- **Processos de Planejamento** – planejar um esquema de trabalho que seja viável para atingir os objetivos do projeto;
- **Processos de Execução** – Coordenar pessoas e recursos para implementar o plano;

- **Processos de Monitoramento e Controle** – Averiguar que os objetivos do projeto estão sendo seguidos, monitorados e avalia-los segundo seu progresso, realizando ações corretivas e replanejando sempre que necessário;
- **Processos de Encerramento** – formalizar a aceitação do projeto ou fase e encerrá-lo(a) de uma forma organizada.

Esses processos são ligados pelos resultados que são desenvolvidos neles, sendo que a saída de um processo é a entrada de outro processo. O desenvolvimento do plano de gerenciamento do projeto é uma atividade iterativa ao longo do ciclo de vida do projeto, sempre pronto para melhoria contínua e permitindo à equipes do projeto definir e trabalhar com maior nível de detalhes. De acordo com o [PMBOK \(2012\)](#) podem ser sobrepostas, ou seja, o início de uma fase é ao término de uma outra, isso leva a algumas atividades ocorrerem de forma paralela. A maneira como este tipo de projeto é definido, aumenta os riscos, retrabalhos, e exige recursos adicionais para permitir que as atividades ocorram em paralelo, como mostrado na Figura 2.

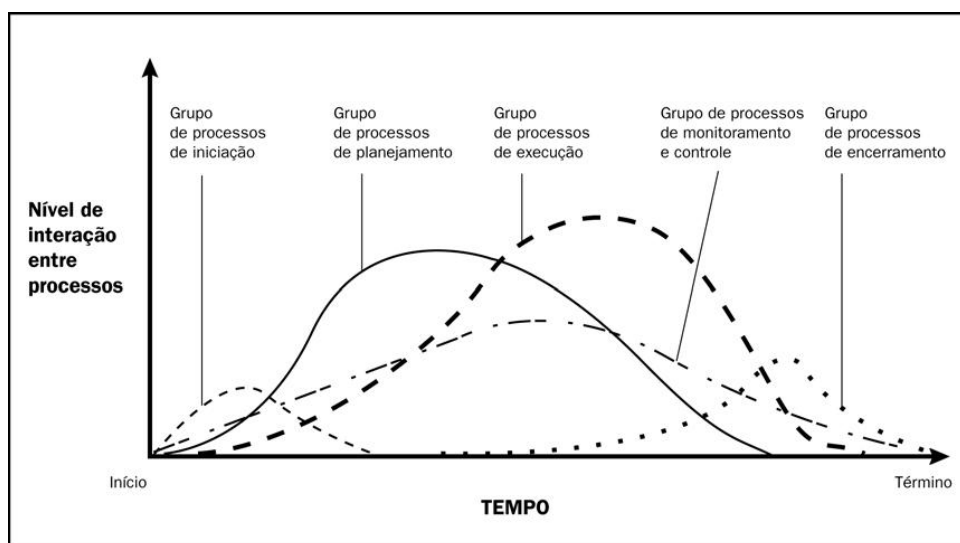


Figura 2 – Superposição dos Grupos de Processo. Fonte: [PMPDIGITAL \(2009\)](#).

2.1.1.1 Áreas de Conhecimento PMBOK

O PMBOK, na versão 5, possui nove áreas de conhecimentos, sendo elas:

- **Gerência da Integração do Projeto** - descreve os processos necessários para que os diversos elementos do projeto sejam adequadamente coordenados. Responsável pelo desenvolvimento do plano do projeto, execução do plano do projeto e controle das mudanças;
- **Gerência do Escopo do Projeto** - descreve os processos necessários para que o projeto atenda ao que foi definido no início do projeto, e nada além disso. Contempla

a fase de iniciação, planejamento, detalhamento, verificação e controle de mudanças do escopo;

- **Gerência do Tempo do Projeto** - descreve os processos necessários para que o projeto seja concluído dentro do prazo previamente estabelecido. Definição, sequenciamento e estimativa das atividades, além do cronograma fazem parte dessa área;
- **Gerência do Custo do Projeto** - descreve os processos necessários para assegurar que o projeto seja completado dentro do orçamento previsto. Supervisiona o planejamento dos recursos, estimativa, orçamento e controle dos custos;
- **Gerência da Qualidade do Projeto** - descreve os processos necessários para assegurar que as necessidades que originaram o desenvolvimento do projeto serão satisfeitas. Abrange o planejamento, garantia e controle da qualidade;
- **Gerência dos Recursos Humanos do Projeto** - descreve os processos necessários para proporcionar a melhor utilização das pessoas envolvidas no projeto. Responsável pelo planejamento organizacional, montagem e desenvolvimento da equipe;
- **Gerência das Comunicações do Projeto** - descreve os processos necessários para assegurar que a geração, captura, distribuição, armazenamento e pronta apresentação das informações do projeto sejam feitas de forma adequada e no tempo certo. Ela é composta pelo planejamento das comunicações, distribuição das informações, relato de desempenho e encerramento administrativo.
- **Gerência dos Riscos do Projeto** - descreve os processos que dizem respeito à identificação, análise e resposta a riscos do projeto. É composta pela identificação, quantificação, desenvolvimento das respostas e controle das respostas aos riscos.
- **Gerência das Aquisições do Projeto** - descreve os processos necessários para a aquisição de mercadorias e serviços fora da organização que desenvolve o projeto. É responsável pelo planejamento, preparação das aquisições, obtenção de propostas, seleção de fornecedores, administração dos contratos e encerramento do contrato.

Este modelo de gerenciamento de projeto é mais voltado em empresas já consolidadas, de ramo mais formal, que possui mais burocracia em seus projetos e por tanto maior rigor de documentação e de liderança dos gerentes. Essa hierarquia pode ser vista na Figura 3.



Figura 3 – Hierarquia em projetos tradicionais. Fonte: Jhonatas (2018).

2.1.1.2 Casos de Uso

Os casos de uso é uma das principais características presentes na linguagem de modelagem UML (*Unified Modeling Language*), ou linguagem modelada unificada. Sommerville (2011) define os objetivos dos casos de uso como identificar os atores envolvidos e suas interações com o projeto. Os casos de uso é um diagrama de auto nível, ou seja, não possui uma linguagem técnica e o usuário final consegue entender sem dificuldades.

Na Figura 4 é possível visualizar um exemplo de caso de uso:

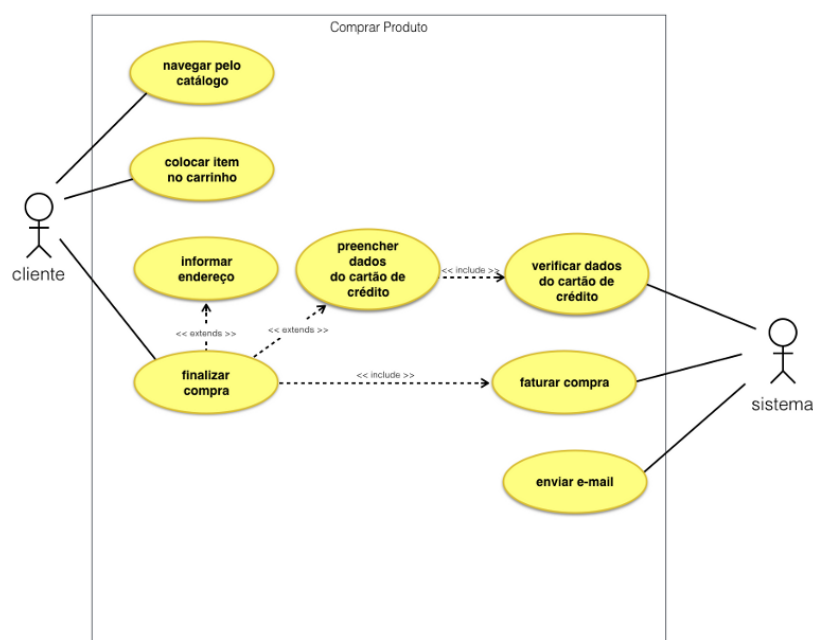


Figura 4 – Exemplo de Caso de Uso. Fonte: Vieira (2015).

2.1.2 Modelo Ágil

Em contraposição ao modelo tradicional, surge o manifesto ágil como uma reação contra o processo burocrático presente no PMBOK, que possuem por característica atividades sequências em modelo iterativo e cascata. Segundo a [Standish \(2014\)](#) apenas 16,2% dos projetos entregues por companhias americanas foram entregues respeitando prazos, custos previamente acordados e objetivos determinados. Segundo a própria [Standish \(2014\)](#), as principais causas destes problemas estavam relacionadas com o modelo sequencial tradicional.

No modelo ágil, segundo [Soares \(2009\)](#), as mudanças são bem vindas, desde que ajude a entregar um projeto melhor. Cabe a equipe de gerência agir de maneira rápida, sabendo receber, avaliar e responder as mudanças como elas devem ser atendidas e repassar suas consequências aos desenvolvedores e clientes. As principais características da metodologia ágil são:

- Desenvolvimento iterativo e incremental;
- Comunicação;
- Documentação reduzida;

Em 2001, membros da comunidade de *software* se reuniram e criaram o [Agile \(2001\)](#). O objetivo deste manifesto é utilizar as melhores práticas observadas em projetos anteriores que obtiveram sucessos.

Os principais conceitos do manifesto ágil são:

- Indivíduos e interações ao invés de processos e ferramentas;
- *software* executável ao invés de documentação;
- colaboração do cliente ao invés de negociação de contratos;
- resposta rápida a mudanças ao invés de seguir planos pré-estabelecidos.

No modelo ágil os requisitos dos clientes podem ser mudados a qualquer momento, e o time de gerência e desenvolvimento devem estar preparados para conversar com o cliente a fim de resolver as alterações de requisitos da melhor maneira possível. Este tipo de pensamento no modelo tradicional é mais difícil de acontecer, pois neste modelo é possível notar que ao iniciar uma fase, essa mesma fase não é retornada mais tarde, ou seja, no modelo tradicional uma troca de requisitos pode levar ao reinício do projeto, ou a um produto final que não atenda mais a necessidade do cliente.

Este modelo ágil é mais focado para empresas emergentes, que não são muito rigorosas em seus processos e aceita que mudanças nos requisitos ou na visão do produto são sempre bem vindas, desde que melhore o projeto final.

2.1.2.1 Scrum

Uma das boas práticas adotadas ao modelo ágil é o *Scrum*. O *Scrum*, é um *framework* que se refere ao jogo *Rugby*, em que os jogadores para avançar, devem avançar no campo e planejar de forma conjunta. Essa ação de planejamento conjunto, aumenta o engajamento dos envolvidos no projeto. Um dos principais pontos de vista do *Scrum* é planejar um projeto com pequenos ciclos e aumentar as interações entre os participantes, mas com visão a longo prazo.

O ciclo de vida de um projeto *Scrum* pode ser visto na figura 5:

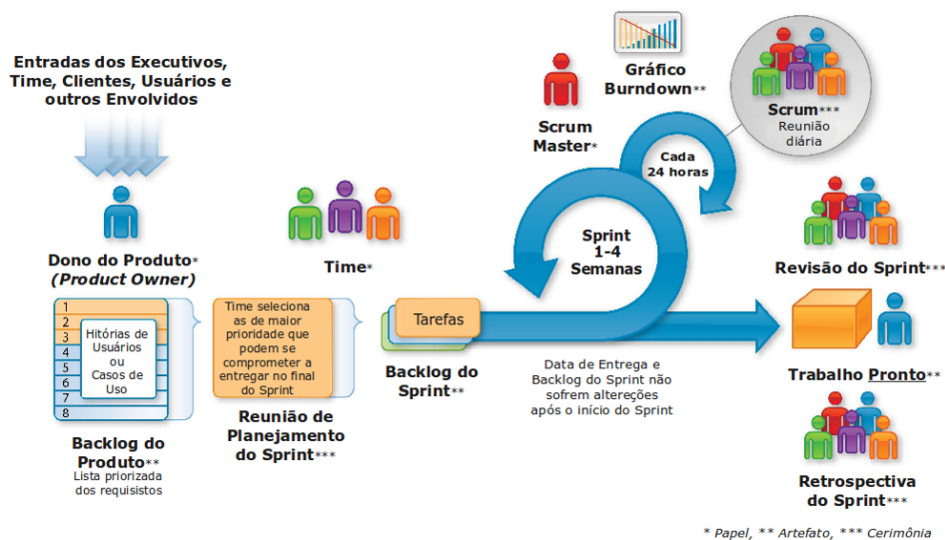


Figura 5 – Ciclo de Vida SCRUM. Fonte: Fabiane (2016).

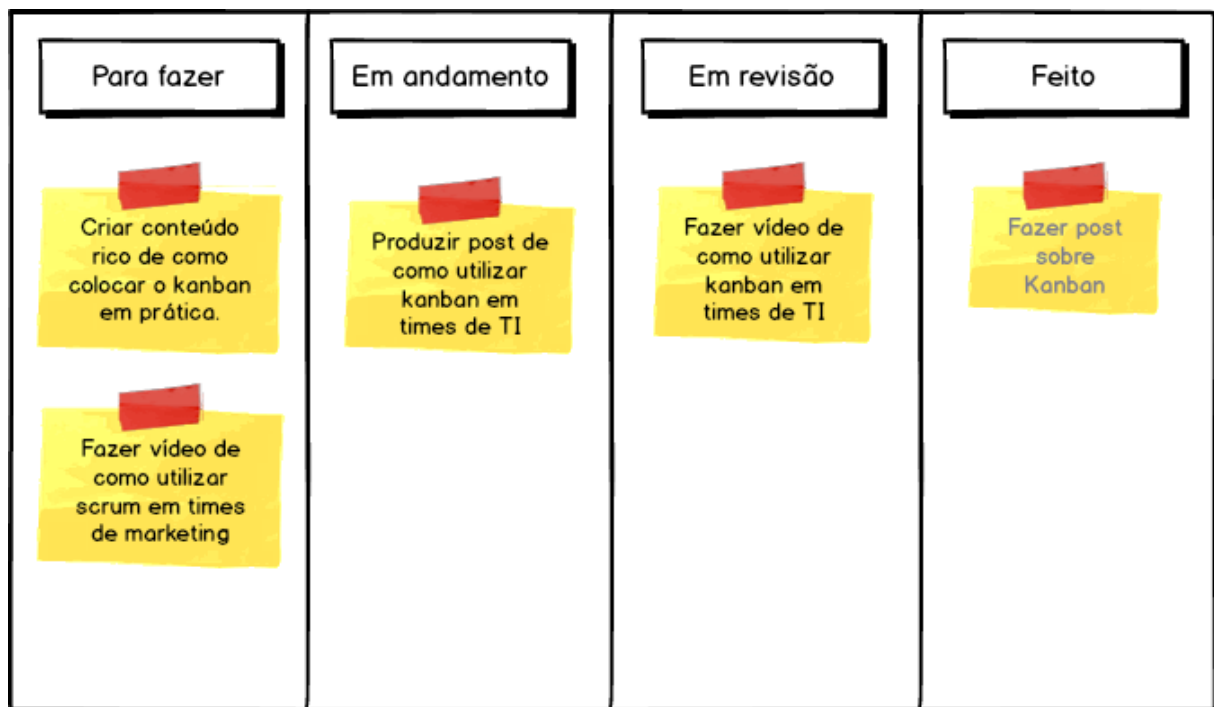
Como pode ser visto na figura 5, o *Scrum* é um ciclo progressivo de várias iterações bem definidas, denominadas *Sprints*. As *Sprints* podem ter duração de uma a quatro semanas. Antes de cada *Sprint*, deve ser realizada a reunião de planejamento da *Sprint*, chamada de *Sprint Planning Meeting*. A *Sprint Planning Meeting* é uma reunião de planejamento em que o *Product Owner* prioriza os itens do *Product Backlog* e a equipe seleciona as atividades que serão implementadas ao longo da *Sprint*. No *Product Backlog* são registradas as funcionalidades que serão implementadas pedidas pelo *Product Backlog*.

Com o objetivo de saber o progresso de cada equipe dentro da *Sprint*, ocorrem as reuniões diárias, denominadas *Daily Meetings*, que tem duração de no máximo 15 minutos e ocorrem com todos os participantes em pé, respondendo perguntas como: "O que você fez ontem?", "O que você fez hoje?" e "O que você vai fazer amanhã?".

Ao final de uma *Sprint* é feita uma análise gráfica do progresso do projeto através do *Sprint Backlog* durante a *Sprint Review*. Após a *Sprint Review* ocorre a *Sprint Retrospective* que é a análise de experiências que ocorreram durante a *Sprint* sejam boas ou não a fim de melhorá-las.

Segundo Fowler (2005), as equipes devem possuir um quadro para registro das atividades, denominado *Kanban*. O *Kanban* possui o objetivo de auxiliar as equipes em relação ao progresso da *Sprint*, esse quadro pode ser dividido em 4 fases:

- Para fazer;
- Em andamento (com o nome do responsável pela atividade);
- Em revisão;
- Feito.



Created with Balsamiq - www.balsamiq.com

Figura 6 – Quadro Kanban. Fonte: Lameck (2016).

O *Scrum* possui seus papéis bem definidos, podendo ser alterados ao longo do desenvolvimento do projeto. Esses papéis podem ser vistos na figura 7.

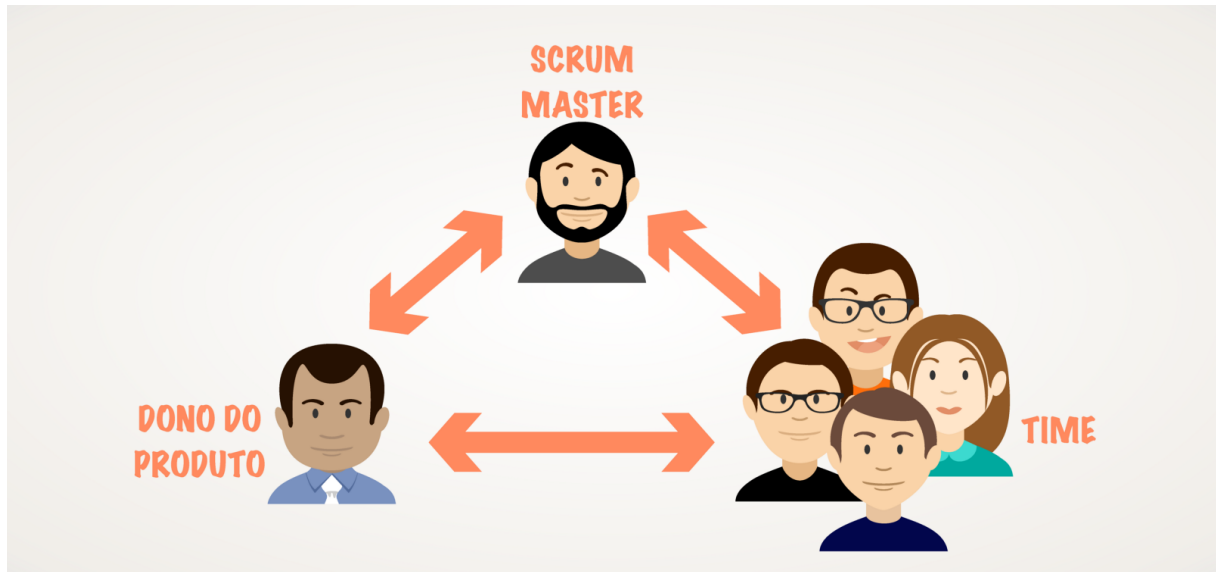


Figura 7 – Papéis Scrum. Fonte: [Gonçalves \(2016\)](#).

Como observado na figura 7, o *Scrum* possui três papéis bem definidos: o *Product Owner*, conhecido como PO, o *Scrum Master* e por fim temos o *Dev Team*.

2.1.2.1.1 Product Owner

O *Product Owner*, é o dono do produto. É ele que fornece o conhecimento do negócio em forma de requisitos para a equipe e na forma prática são os patrocinadores/clientes do produto. O PO organiza e prioriza o *Product Backlog* (que são os itens que devem ser desenvolvidos), esse papel deve ser assumido por pessoas que sejam boas em se comunicar, pois esse papel é responsável por trabalhar e tirar dúvidas da *Dev Team*.

2.1.2.1.2 Scrum Master

Ao analisar a figura 7, a figura do *Scrum Master* parece ser superior aos outros papéis, o que não é verdade. O *Scrum Master* possui o dever de ajudar a comunicação entre o PO e o *Dev Team* além de remover todos os impedimentos que estão prejudicando o desenvolvimento, tem a função de auxiliar o amadurecimento da *Dev Team* e promover as cerimônias que o *Scrum* preza, como *Daily Meetings*, *Sprint Review* e *Sprint Retrospective*.

2.1.2.1.3 Dev Team

Esse papel é voltada para as pessoas que de fato irão desenvolver o produto. O *Dev Team* é auto-organizável e decide entre si como implementar os itens priorizados pelo PO.

2.1.2.2 Scrum Solo

Uma das adaptações do *Scrum* é o *Scrum Solo*, criado por [PAGOTTO et al. \(2016\)](#) na UTFPR, Universidade Tecnológica Federal do Paraná. O *Scrum Solo* é voltado para o desenvolvimento individual de *software*, contudo sem seguir todos os rituais presentes no *Scrum*, como visto no tópico [2.1.2.1](#).

Ao final de uma *Sprint*, um incremento de *software* deve ser entregue assim como no *Scrum*. Os artefatos de *Product Backlog* e *Sprint Backlog* são feitos da mesma maneira em ambos os casos. No *Scrum Solo* como é desenvolvido individualmente, não há necessidade de reuniões diárias entre os membros como no *Scrum*, havendo reunião somente quando necessário entre o desenvolvedor e o grupo de validação. Esse ciclo de vida pode ser visualizado na Figura 8:



Figura 8 – Ciclo de Vida *Scrum Solo*. Fonte: [PAGOTTO et al. \(2016\)](#).

As atividades do *Scrum Solo*, segundo [PAGOTTO et al. \(2016\)](#) são:

- *Requeriment*: definir escopo, caracterizar o cliente e definir o *Product Backlog*;
- *Sprint*: selecionar o que será desenvolvido a partir do *Sprint Backlog* em duração máxima de uma semana;
- *Deployment*: tem como objetivo disponibilizar o produto para o cliente;
- *Management*: planejar, monitorar e controlar o desenvolvimento do produto.

Segundo [PAGOTTO et al. \(2016\)](#), os atores envolvidos são:

- *Product Owner*: proprietário do produto;
- Desenvolvedor: responsável por executar o processo e desenvolver o produto;
- Orientador: consultor que conhece a fundo o processo;
- Grupo de validação: possíveis usuários do produto.

2.2 Processo de Desenvolvimento de Software

[Sommerville \(2011\)](#) define o processo de desenvolvimento de software como "Um processo de *software* é um conjunto de atividades relacionadas que levam à produção de um produto de *software*."

Neste trabalho, foram definidas as principais atividades a serem realizadas para alcançar o objetivo final de ter um *software* gratuito, com código aberto e que auxilie os gerentes a otimizar suas reuniões por meio computacional são:

- Especificação do *software*: funcionalidades e restrições do *software*;
- Projeto e implementação do *software*: as especificações que o *software* deve atender;
- Validação de *software*: para que atenda as expectativas do cliente, o *software* deve ser validado pelo mesmo;
- Evolução do *software*: o *software* deve ser extensível a mudanças, tendo assim seu código aberto.

2.2.1 Definição dos Requisitos

Requisito não é um termo usado apenas pela Engenharia de Software . Há casos em que requisitos são apenas uma declaração abstrata em alto nível de um serviço ou restrição que um sistema deve oferecer.

[Sommerville \(2011\)](#) os define como: "Os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços que oferece e as restrições a seu funcionamento. Esses requisitos refletem as necessidades dos clientes para um sistema que serve a uma finalidade determinada, como controlar um dispositivo, colocar um pedido ou encontrar informações."

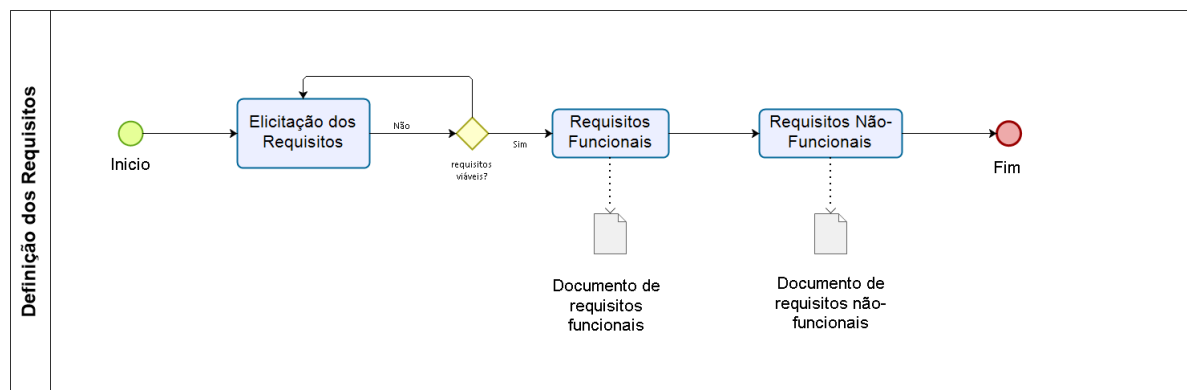


Figura 9 – Definição dos Requisitos. Fonte: Própria.

2.2.1.1 Elicitação dos Requisitos

[Sommerville \(2011\)](#) define a elicitação de requisitos como uma fase do projeto onde são extraídas informações do cliente sobre o que ele deseja que seja construído. É a fase em que o profissional de TI entende a necessidade do cliente e o orienta. É o momento de conversa com o usuário, de sentimento sobre o que este espera que seja entregue a ele. Na elicitação de requisitos são percebidas as necessidades do usuário final e as características que esse sistema deve ter. Dentre todas as técnicas dentro da literatura, foram selecionadas duas como candidatas a serem usadas no projeto:

- Entrevista
- Observação

2.2.1.1.1 Entrevista

Segundo [Sommerville \(2011\)](#), as entrevistas podem ser formais ou informais, sendo esta técnica uma das mais utilizadas nos processos de engenharia de requisitos. As entrevistas são realizadas pela equipe de engenharia de requisitos, em que são feitas perguntas aos *stakeholders* sobre o sistema em vigor e o que será desenvolvido. A partir dessas perguntas surgem os requisitos.

As entrevistas podem ser de dois tipos: fechadas e abertas. Nas entrevistas fechadas, os *stakeholders* respondem perguntas um conjunto de perguntas pré-definidas. Já nas entrevistas abertas, são realizadas perguntas abertas sobre o sistema, e é nesse tipo de entrevista em que ocorre uma melhor compreensão das necessidades dos *stakeholders*.

2.2.1.1.2 Observação

(SOMMERVILLE, 2011) define observação como uma técnica que pode ser usada para compreender os processos operacionais e ajudar a extrair os requisitos de apoio para esses processos. O trabalho do dia a dia é observado e são feitas anotações sobre as tarefas reais em que os participantes estão envolvidos. O valor da observação é que ela ajuda a descobrir requisitos implícitos do sistema que refletem as formas reais com que as pessoas trabalham, em vez de refletir processos formais definidos pela organização.

2.2.1.2 Requisitos Funcionais

Os requisitos funcionais descrevem o que o sistema deve de fato ser. Requisitos funcionais podem ser tão específicos quanto necessário, por exemplo, podem ter sistemas com requisitos funcionais gerais e outros que além de refletir os sistemas, também abrangem as formas de trabalho de uma organização. Requisitos funcionais de um sistema deve ser completo, isso quer dizer que todos os serviços requisitados pelo usuário devem ser definidos.

2.2.1.3 Requisitos Não-Funcionais

Requisitos não-funcionais são requisitos que são relacionados as propriedades do sistema como confiabilidade, tempo de espera, desempenho, segurança e até restrições do sistema. Requisitos não-funcionais podem possuir tanta relevância quanto os requisitos funcionais, pois em uma reunião de levantamento de requisitos, o cliente sonha o mundo e não está atento se os recursos próprios recursos e os recursos da empresa conseguem atender ao requisito. Um requisito não-funcional não atendido pode inclusive inutilizar um projeto. Exemplo disso é caso um sistema de uma aeronave não consiga atingir a confiabilidade necessária, não será dado o certificado de segurança para operar, sendo assim a aeronave não poderá voar.

2.2.2 Linguagem de Software

Linguagem de programação são instruções passadas de maneira que o computador entenda e apresente um retorno. Existem diversas linguagens de programação, desde a mais baixa a alto nível.

Linguagens de *software*, como também podem ser chamadas, são divididas em duas frentes: *front-end* e *back-end*. Ambas serão explicadas nos tópicos a seguir.

2.2.2.1 Front-end

A programação de um *software* pelo ponto de vista do *front-end* é a visão final do usuário com o sistema. *Front-end* é a responsável pela interação do usuário com o sistema

e essa interação é dada a partir de telas/páginas. Existem diversos tipos de *frameworks* que auxiliam os desenvolvedores a trabalhar com essa frente, como:

- *Bootstrap*
- *Materialize*
- *ReactJs*
- *Angular 4*

2.2.2.2 Back-end

A programação *back-end* possui as responsabilidades de receber os dados fornecidos pelo usuário à partir do *front-end* e transformar esses dados em informações relevantes ao usuário final. O *back-end* possui o dever de tratar os dados, validá-los e fomentá-los a visão do usuário.

Existem diversas linguagens *back-end* que auxiliam os desenvolvedores a trabalhar em uma linguagem que o computador entende, como:

- *Python Django-Rest*
- *Java*
- *Ruby on Rails*
- *PHP*

Para este projeto foram escolhidas as linguagens de *front-end* o *ReactJs* e para *back-end* o *Python Django-Rest*.

2.2.2.3 Arquitetura de Software

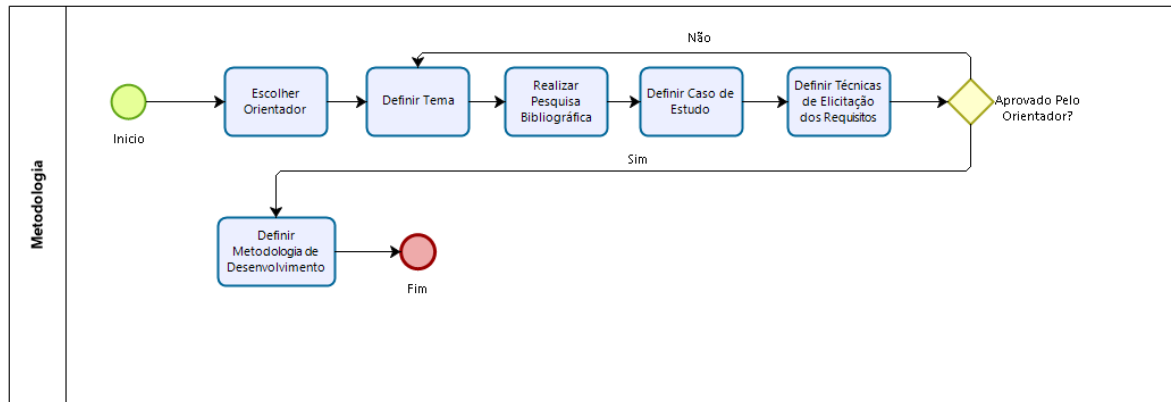
A arquitetura de *software* é como o sistema deve ser organizado com a estrutura geral do projeto. A arquitetura possui um valor alto dentro da construção de um *software*, pois nela se tem o elo entre o projeto e a engenharia de requisitos. Possui o dever identificar os principais componentes estruturais no sistema e o relacionamento entre eles.

2.2.2.4 Model-View-Controller

O padrão arquitetural MVC é responsável de responsabilidades em camadas. A primeira é *Model*(modelo), que é responsável pela manipulação de dados, ou seja, leitura, escrita de dados e também suas validações é de responsabilidade da *Model*. A segunda camada é a *View*(visão), que possui a responsabilidade de interação com o usuário. Por

último se tem a *Controller* (controladora), responsável por receber as aquisições do usuário. A controller também tem o dever de disponibilizar os dados para a *view* e assim ocorrer a interação com o usuário.

3 Metodologia



Powered by
bizagi
Modeler

Figura 10 – Metodologia. Fonte: Própria

A figura 10 mostra como foi elaborada a metodologia deste projeto. Alguns processos demonstrados já foram elaborados como "Escolher o Orientador, Definir Tema" e "Realizar Pesquisa Bibliográfica". A seguir, os demais processos serão contemplados e explicados adequadamente.

3.1 A Instituição

Tendo a justificativa para o projeto no tópico 1.2, seguida do problema de pesquisa (1.3) e os objetivos descritos no tópico 1.4, houve a necessidade de escolher alguma empresa que seria usada como caso de estudo para o projeto, no caso foi definido o NMIL (Núcleo de Modernização da Informação Legislativa), um setor localizado no Senado Federal Brasileiro.

3.1.1 Senado Federal

As funções do Senado Federal são exercidas pelos senadores da República, que são eleitos segundo o princípio majoritário para representarem os estados e o Distrito Federal. Os estado e o Distrito Federal elegem três senadores para um mandato de oito anos. A renovação da representação se dá a cada quatro anos, alternadamente, por um e dois terços. Cada senador é eleito com dois suplentes.

A Estrutura Administrativa compreende a formação das unidades do Senado, suas atribuições, responsáveis e formas de contato.

A Administração tem como ênfase os compromissos com o Parlamento; com excelência na prestação de serviços públicos; com qualidade de vida dos colaboradores; com a igualdade; com a livre disseminação de ideias; com a transparência; com a responsabilidade na utilização de recursos públicos; com a memória do Senado; e com a comunidade. Na figura 11 pode ser visualizado o organograma organizacional do Senado Federal com suas casas e secretárias.

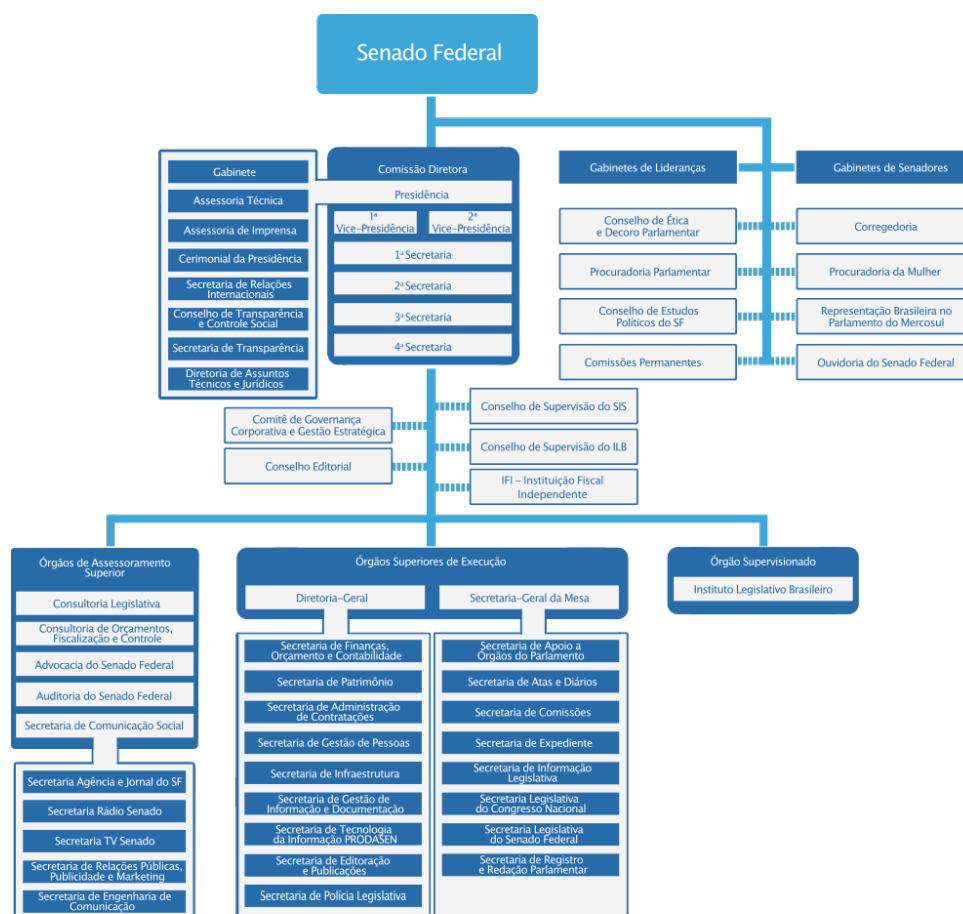


Figura 11 – Organograma Senado Federal. Fonte: Senado (2019).

3.1.2 NMIL

A Comissão Diretora é composta pelo Presidente, dois Vice-Presidentes e quatro Secretários. A composição dessa comissão muda a cada dois anos, sendo assim na primeira e terceira fase de uma legislatura, sendo esta com duração de 4 anos. É de responsabilidade da Comissão a direção da casa, designando atividades às unidades que dão suporte.

Essas unidades são: Secretaria Geral da Mesa (SGM), representante da atividade fim da casa; e Diretoria Geral (DGER), que, representa as atividades meio da casa. As duas contam com secretarias, às quais delegam atividades exigidas pelo Presidente.

Quando o Presidente da Comissão Diretora necessita de apoio tecnológico, delega esta atividade à SGM, que, ao receber o problema, começa a definir diretrizes estratégicas para a solução do problema. Após o término da definição das diretrizes, encaminha-as à Secretaria de Informação Legislativa (Sinfleg).

O diretor da Sinfleg atua como Gerente do Projeto, e conta com o apoio da equipe do NMIL na administração do projeto.

A equipe do NMIL realiza reuniões com as áreas afetadas pelo projeto até conseguir definir todos os requisitos para o produto que será gerado. Após a definição, convoca uma reunião com o Gerente para entrega dos requisitos definidos. O Gerente analisa esses requisitos para saber se são viáveis. Caso não sejam, pede ao NMIL novos requisitos e, só após receber requisitos viáveis, aprova a proposta de solução.

O próximo passo é dado pelo NMIL, convocando reunião com a Secretaria de Tecnologia da Informação (Prodasen). Nesta reunião discute-se os requisitos aprovados e prepara-se o Termo de Abertura do Projeto (TAP). O TAP, deve ser encaminhado pelo NMIL ao Gerente para que este aprove o documento; caso não aprove, pede-se um novo, até que seja aprovado.

Após o Gerente aprovar o projeto, ele o apresenta ao Secretário Geral da Mesa, que é o representante da SGM, para uma aprovação final. O Secretário Geral da Mesa também pode pedir um novo projeto, mas se não for o caso, apenas o autoriza.

Dada a aprovação do Secretário Geral da Mesa, a equipe do Prodasen, responsável pela construção do produto, dá início à construção do projeto, fazendo as entregas de ambiente de homologação (definidas no TAP) ao NMIL, a fim de que este realize testes. Se forem encontrados erros, estes são listados e repassados ao Prodasen para que sejam reparados. Quando não há mais erros, o NMIL dá sua aprovação do produto. Em seguida o Prodasen termina sua parte do projeto e entrega o produto finalizado ao NMIL. O NMIL encaminha o produto ao Gerente que autoriza o produto e apresenta-o ao Secretário Geral da Mesa.

O Secretário Geral da Mesa, após receber o produto, pode solicitar alterações ao NMIL, que em seguida encaminha esta solicitação ao Prodasen. A equipe do Prodasen responsável pelo produto faz as alterações necessárias e encaminha de volta ao NMIL, passando pelo processo de teste e aprovação novamente até que o Secretário Geral da Mesa autorize a implantação.

Quando o Secretário Geral da Mesa autorizar a implantação, cabe ao NMIL apresentar aos usuários o novo Sistema ou as atualizações em sistemas já existentes.

As figuras relacionadas ao mapeamento do processo que ocorre atualmente no NMIL, pode ser vistos no apêndice C, sendo as figura 24 e 25 como o processo de pedido da comissão diretora para um novo sistema passando pela SGM, Sinfleg, NMIL e por fim ao Prodasen. As figuras 26 e 27 se referem ao processo que o NMIL passa até conseguir atingir um sistema estável e que atenda ao pedido da comissão diretora.

3.2 Metodologia de Desenvolvimento

Por conta de possuir um maior conhecimento sobre a metodologia e pela proposta em entregas mais frequentes em períodos menores, foi escolhida a metodologia ágil *Scrum*, explicada no tópico 2.1.2 como metodologia de desenvolvimento do *software* juntamente com algumas práticas do *Scrum* Solo.

3.2.1 Scrum

Uma das principais vantagens do *Scrum* é a adaptação dele a projetos menores e que não são rigorosos a processos, e após ser feita uma análise dos tipos de gerenciamento de projetos no tópicos 2.1, foi escolhido o *Scrum* Solo como adaptação do *framework Scrum* como metodologia de desenvolvimento deste TCC.

3.2.1.1 Papéis

Este projeto será desenvolvido de forma individual, os papéis do *framework Scrum* foram distribuídos de forma com que cada ator tenha a seguinte responsabilidade: o responsável pelo desenvolvido do sistema, Victor Mota, assume os papéis de Time de Desenvolvimento e *Scrum Master*. O papel de *Product Owner* será assumido por Pedro Marques, servidor do Senado Federal e do NMIL, que é o setor de caso de estudo deste trabalho.

3.2.1.2 Sprints

Para este projeto, as *Sprints* foram definidas com duração máxima de duas semanas. Assim como define no *Scrum*, as *Sprints* devem ter as atividades de planejamento e revisão da mesma, para que seja constatado se o que foi planejado foi entregue ao longo das duas semanas.

- *Sprint Planning*: Esta atividade é realizada no primeiro dia de *Sprint* e é nela que são selecionados os itens do *Product Backlog* que serão desenvolvidos ao longo da *Sprint*;

- *Sprint Review*: Esta atividade é realizada no último dia de *Sprint* e é nela são discutidas com *Product Owner* as histórias de usuário desenvolvidas ao longo da *Sprint*.

4 Processo de Desenvolvimento de Software

Como definido o tópico 2.2, foi elaborado um processo de desenvolvimento de *software* deste projeto e que pode ser visualizado na Figura 12.

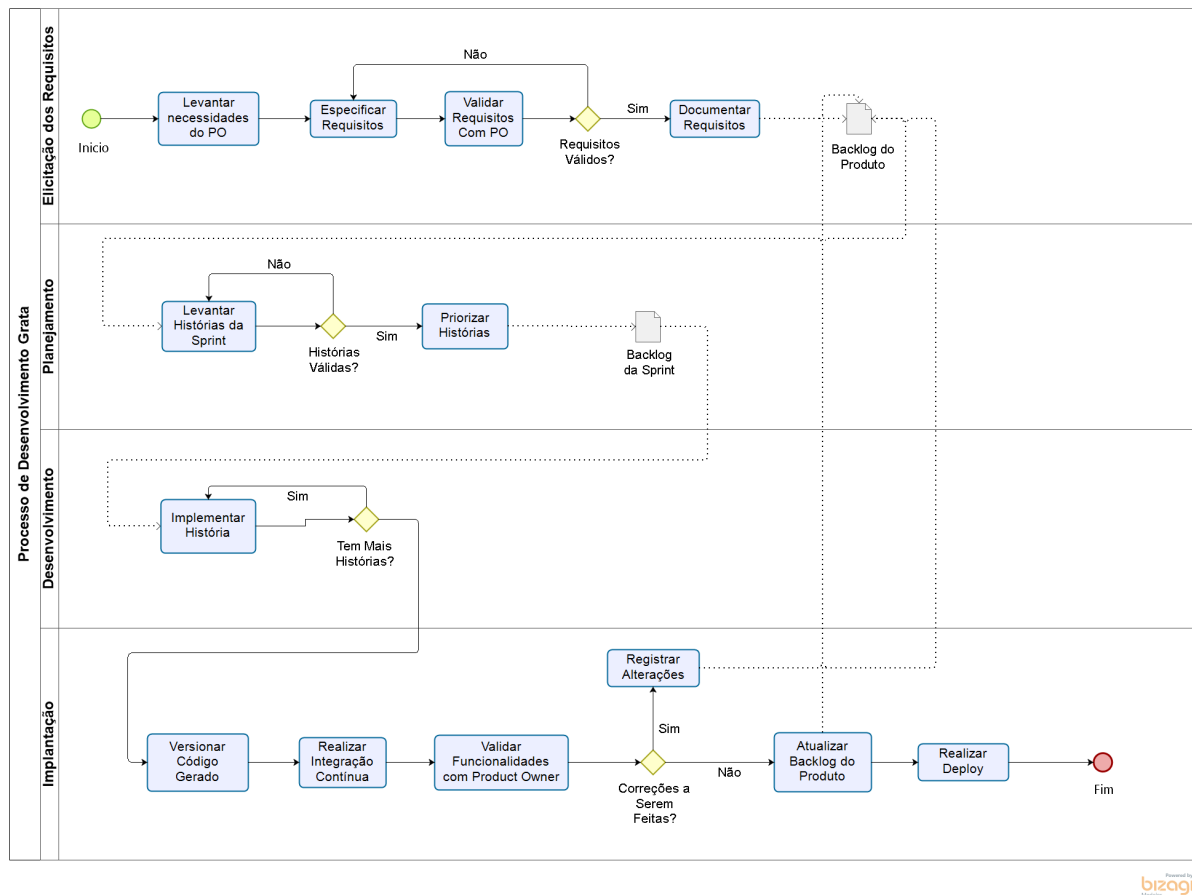


Figura 12 – Processo de Desenvolvimento do Grata. Fonte: Própria

4.1 Elicitação dos Requisitos

De acordo com a Figura 12, a primeira atividade a ser realizada é elicitação dos requisitos, definidos no tópico 2.2.1.1. As técnicas utilizadas para elicitação dos requisitos foram entrevista e observação, definidas respectivamente nos tópicos 2.2.1.1.1 e 2.2.1.1.2.

A primeira atividade do processo 12 foi facilitada pois o desenvolvedor deste projeto estagiou no setor do estudo de caso, NMIL, e com isso foram realizadas entrevistas a fim de entender como funciona todo o processo de marcar uma reunião até ela ser realizada. As entrevistas foram do tipo informal, pois assim foi possível entender melhor o contexto

inserido. Esses processos podem ser visualizados nas Figuras 24, 25, 26 e 27, que juntam englobam todo o processo do setor envolvendo reuniões, que conta com a presença de diversos participantes de diferentes setores.

Após realizar as entrevistas, foram realizadas observações sobre como os *stakeholders* interagem com o sistema em vigor, e foi constatado que o sistema atual chamado Gertiq (Gerenciador de Tíquetes), ele é usado para marcar reuniões, contudo os participantes são chamados via email pela ferramenta *Outlook*, não possuem um sistema para anotar as pautas e atas das reuniões, sendo feitas hoje a partir de folhas de papel.

4.1.1 Requisitos Funcionais

Para a elaboração dos requisitos funcionais, foi constatado a necessidade de uma identificação e descrição dos usuários do sistema, sendo apresentadas a seguir nas Tabelas 1 e 2:

Usuário	Administrador
Descrição	Usuário que irá ter controle sobre o sistema e todas as funcionalidades.
O que ele faz?	Ele é responsável pela condução das reuniões, documentar as ATAs, apresentar relatórios e pauta das reuniões, acompanhar a evolução dos projetos, e receber <i>feedbacks</i> sobre as reuniões.
O que ele precisa?	Ele precisa de usuário e senha para conseguir acessar o sistema. Visualizar um ambiente completo com todas as funcionalidades disponíveis no sistema.

Tabela 1 – Usuário Administrador

Usuário	Participante das Reunioes
Descrição	Usuário que irá acessar o sistema para visualizar a ATA das reuniões que participou.
O que ele faz?	Contribui com informações nas reuniões usadas para gerar requisitos do produto.
O que ele precisa?	Precisa de usuário e senha para conseguir acessar o sistema, visualizar as ATAs das reuniões que participou, podendo imprimir, baixar, e deixar comentários.

Tabela 2 – Usuário Participante

Após identificados e descritos os usuários do sistema, foi elaborado um diagrama de caso de uso, explicado no tópico 2.1.1.2, sendo apresentado a seguir:

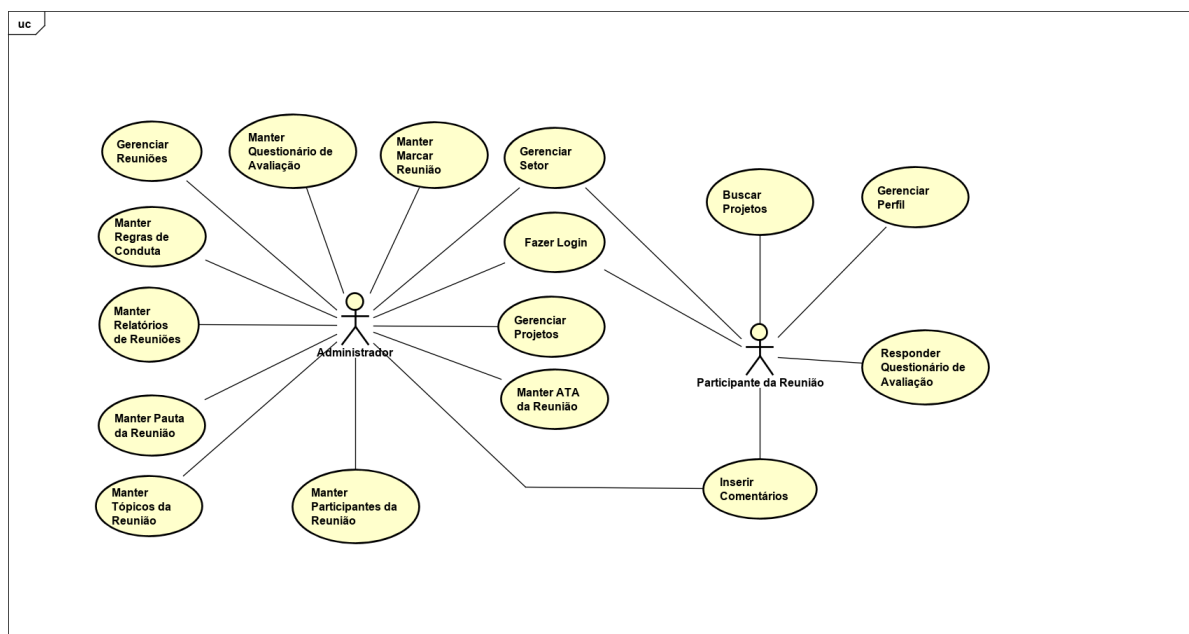


Figura 13 – Casos de Uso Grata. Fonte: Própria

O diagrama de casos de uso da Figura 13 foi elaborado não apenas para visualização das funcionalidades, bem como as interações dos atores com o sistema, mas desenvolvido também como auxílio visual das funcionalidades para montar o *backlog* do produto.

4.1.1.1 Backlog do Produto

O Backlog do Produto é composto pelas Histórias de Usuário e Histórias Técnicas elicítadas e desenvolvidas durante as Sprints. Estas representam necessidades levantadas reais para a aplicação, levantadas pelo PO e desenvolvedor.

4.1.1.2 Histórias de Usuário

As Histórias de Usuário representam as necessidades levantadas pelo usuário, que culminam em funcionalidades da aplicação. Nas Tabelas 5, 6, 7, e 8 são apresentadas as Histórias de Usuário elicítadas a partir da análise do diagrama de casos de uso da Figura 13.

4.1.1.3 Histórias Técnicas

As Histórias Técnicas são necessidades levantadas para cumprir questões pontuais que não necessariamente estão ligadas aos requisitos funcionais. Essas histórias podem agregar ou não valor ao produto diretamente. As histórias técnicas deste projeto estão na Tabela 9.

4.1.2 Requisitos Não-Funcionais

Os requisitos não-funcionais, explicitados no tópico 2.2.1.3, são explicitados a seguir na Tabela 3:

Requisito	Descrição
Usabilidade	O sistema de gerenciamento de reuniões e ATAs será construído para rodar em ambiente web. Deverá possuir um design responsivo. A interface do sistema deverá se comportar adequadamente independente do <i>front-end</i> que será utilizado para o acesso - <i>Browser</i> , <i>Smartphone</i> ou <i>Tablet</i> .
Compatibilidade	O sistema é desenvolvido para ser um sistema web, logo não deve apresentar problemas de compatibilidade entre <i>browsers</i> , sendo eles o <i>Google Chrome</i> , <i>Mozilla Firefox</i> , <i>Microsoft Edge</i> e <i>Safari</i> .
Segurança	O sistema deve garantir a segurança dos dados mais críticos tais como senha dos usuários e <i>emails</i> .
Validação	O sistema deverá validar todos os campos obrigatórios tais como, nome do usuário, senha, nome e descrição da reunião.
Manutenção e Evolução	O código deverá seguir padronização pela comunidade de <i>software</i> , seguir recomendações de comentários em partes de código mais complexas, para assim facilitar a manutenção e evolução no futuro.

Tabela 3 – Requisitos Não-Funcionais

4.1.3 Diagrama de Classe

Para o desenvolvimento do sistema e auxiliar uma manutenção e evolução do sistema, foi montado um diagrama de classe que pode ser visto na Figura 14:

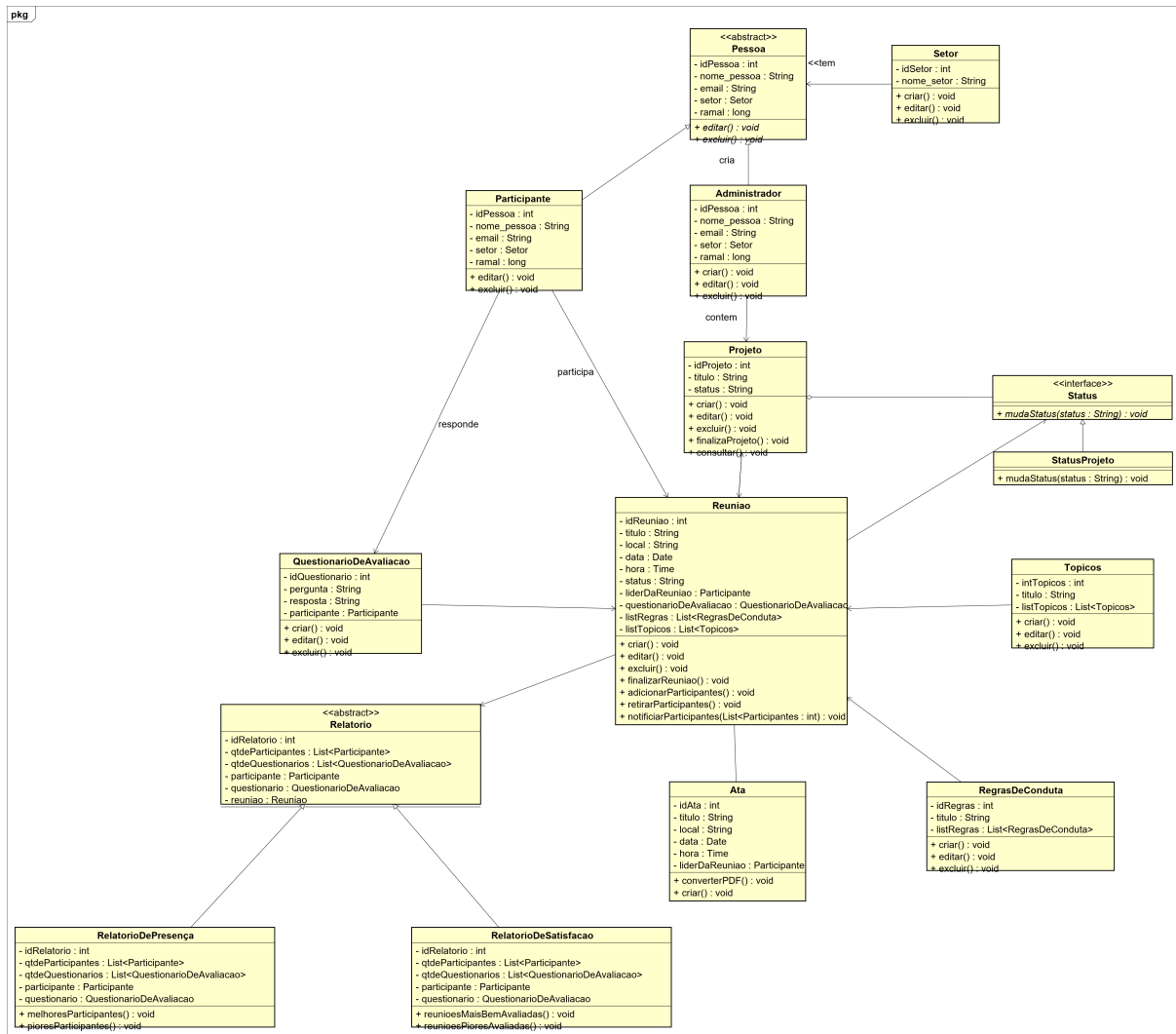


Figura 14 – Diagrama de Classe. Fonte: Própria

4.1.4 Banco de Dados

4.1.4.1 Modelo Conceitual

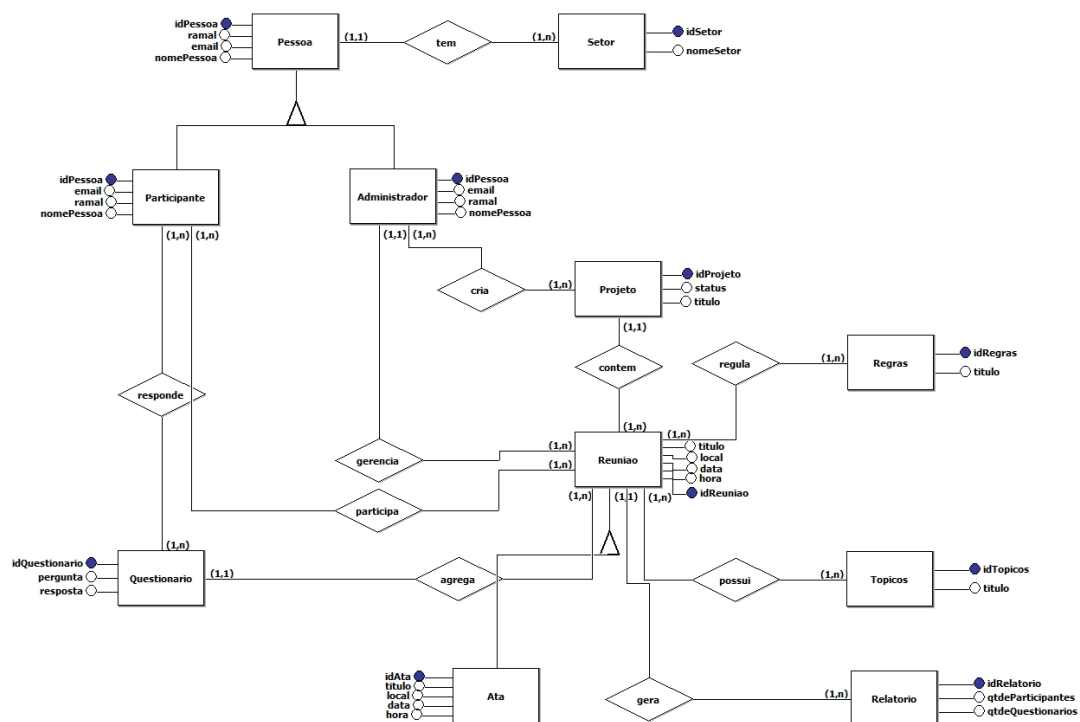


Figura 15 – Banco de Dados Modelo Conceitual. Fonte: Própria

4.1.4.2 Modelo Lógico

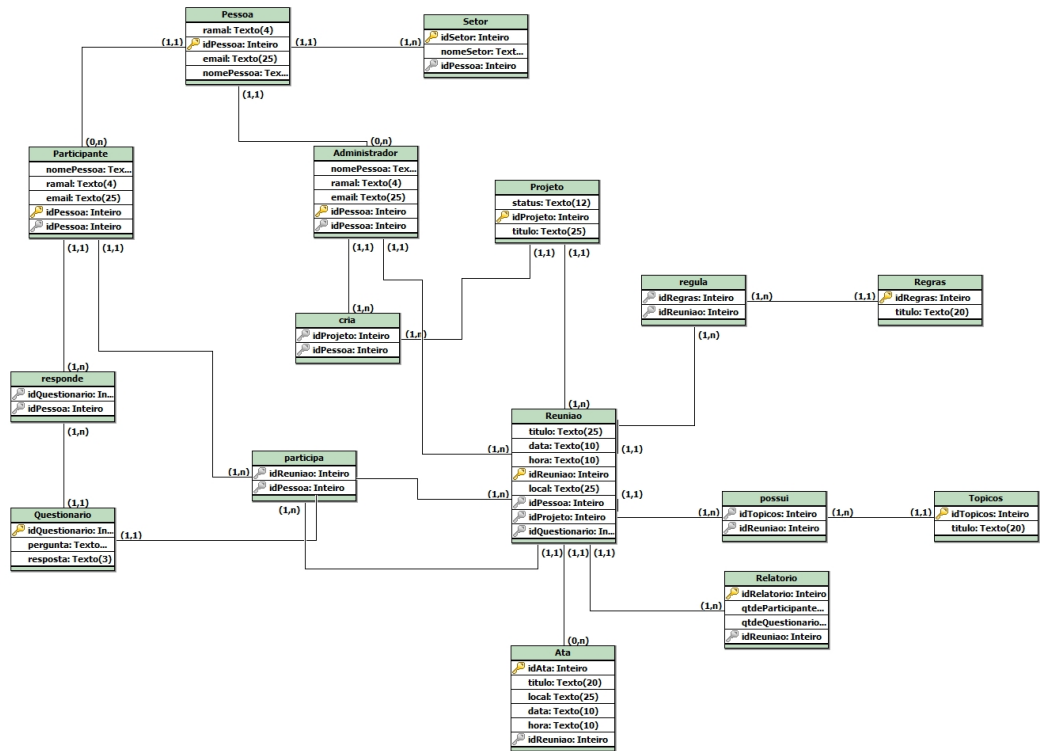


Figura 16 – Banco de Dados Modelo Lógico. Fonte: Própria

4.1.5 Front-End

No tópico 2.2.2.1, foi explicado e exemplificada algumas linguagens *front-end*. A linguagem *front-end* escolhida para este projeto, foi o *React*, pois além de facilitar o desenvolvimento e interação com usuário final, é uma das mais utilizadas ao redor do mundo, então facilita uma manutenção futura e evolução do *software*.

4.1.6 Back-End

No tópico 2.2.2.2, foi explicado e exemplificado algumas linguagens *back-end*. A linguagem *back-end* escolhida para este projeto, foi a *Python Django-Rest*, pois tem uma ótima conexão com a linguagem *front-end*, e por ser muito utilizada, possibilitando assim uma manutenção e evolução futura.

4.1.7 Protótipos

Foram elaborados alguns protótipos para auxiliar o desenvolvimento deste projeto, sendo possível visualiza-los nos apêndices D.

4.1.8 Deploy

Para o *deploy*, foi escolhido o *Heroku*, pois é uma plataforma que além de criar, monitorar, entregar e escalar produtos, também é uma ferramenta que possui integração com o *GitHub* e que para este projeto o *Heroku* será utilizado para hospedar os dados da aplicação.

4.1.9 Arquitetura do Projeto

Neste projeto é feita uma adaptação ao padrão MVC, exemplificado no tópico [2.2.2.4](#), por conta da escolha da linguagem *front-end*. O *React* possui um padrão arquitetural diferente do MVC, chamado de arquitetura de componentes. Esse padrão possui semelhanças ao MVC, e o que será utilizado dele será a parte da *View*. Enquanto a linguagem *back-end* é responsável por trabalhar os dados provindos do *front-end* e oferecer um retorno a ele.

A seguir será mostrado como funciona separadamente o *React*, relacionado ao *front-end* que engloba a *View*, enquanto o *Python Django-Rest* é responsável pelo *back-end* e que engloba a *Model* e a *Controller*:

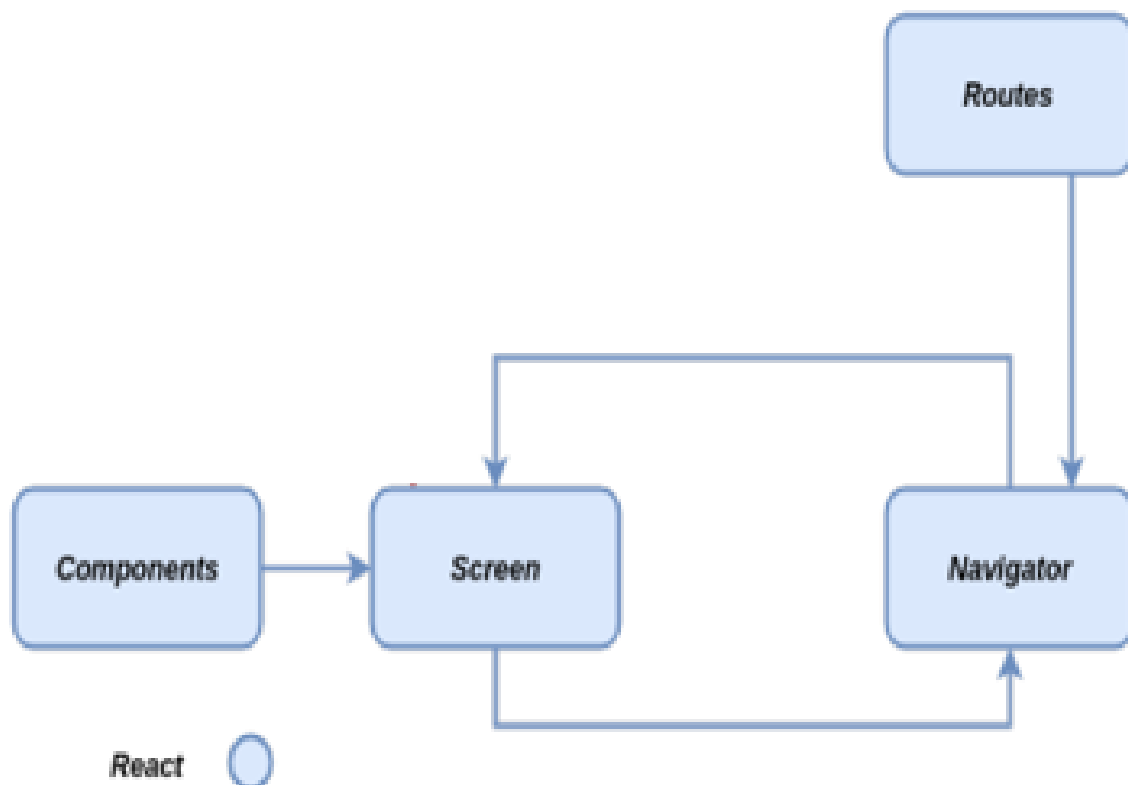


Figura 17 – Diagrama React

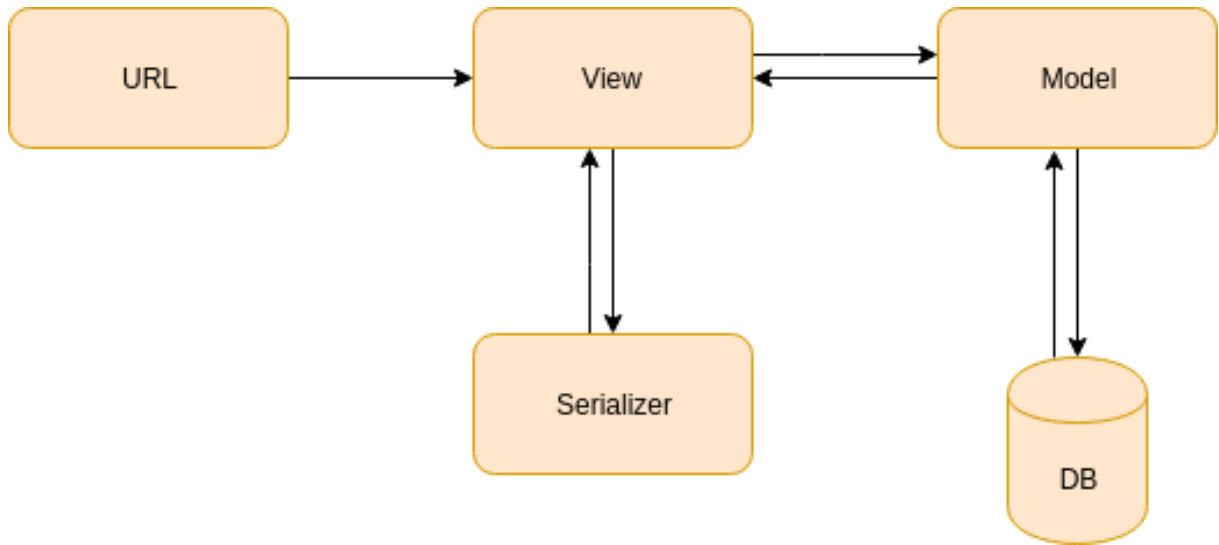


Figura 18 – Diagrama Django REST Framework

4.2 Ferramentas Auxiliares

Na Tabela 4 é mostrada as figuras que compõem este projeto.

Ferramenta	Uso
Astah UML	Diagrama de Casos de Uso.
	Diagrama de Classe
Bizagi Modeler	Modelagem dos Processos
BrModelo	Diagrama de Banco de Dados
Ganttter	Cronograma

Tabela 4 – Ferramentas Auxiliares

5 Resultados

O Grata foi desenvolvido utilizando o *Python Rest* como *Backend* e o *React* como *Frontend*. Para facilitar futuras evoluções foi utilizado o *Docker* e o *Deploy* foi realizado na plataforma *Heroku*. Tudo desenvolvido para este projeto pode ser encontrado nos repositórios:

- Documentação Grata: <<https://github.com/FGAProjects/TCC>>;
- Frontend: <<https://github.com/MrVictor42/Grata-Frontend-v2>>;
- Backend: <<https://github.com/MrVictor42/Grata-Backend-v2>>.

O links dos deploys da ferramenta:

- Backend: <<https://api-grata.herokuapp.com/>>
- Frontend: <<https://projeto-grata.herokuapp.com/>>

5.1 Ferramenta

A seguir algumas imagens da ferramenta desenvolvida:

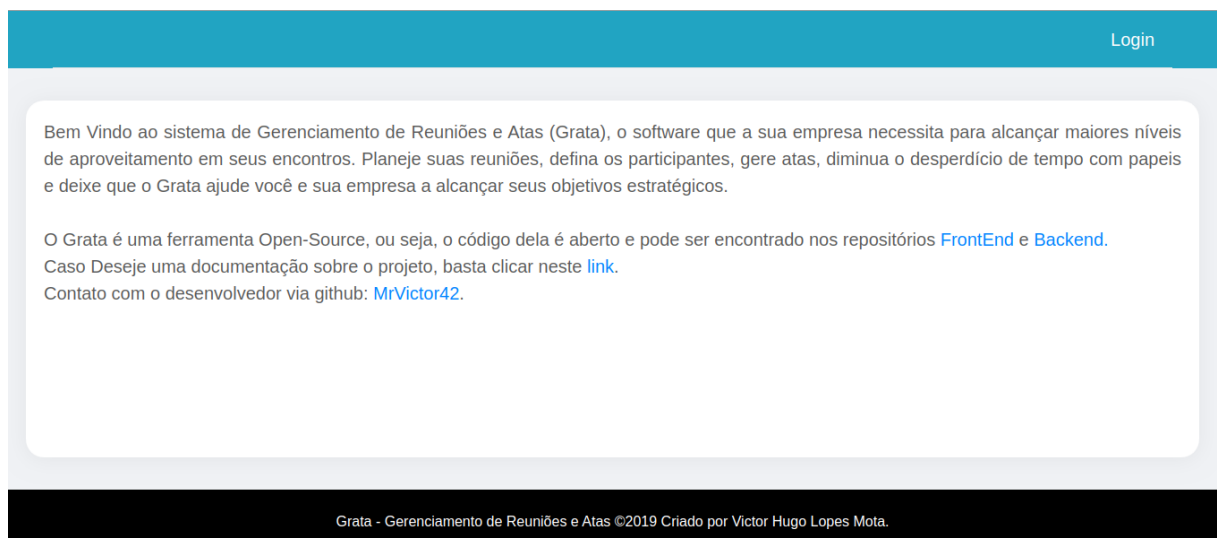


Figura 19 – Login Grata. Fonte: Própria



Figura 20 – Página Inicial Grata. Fonte: Própria

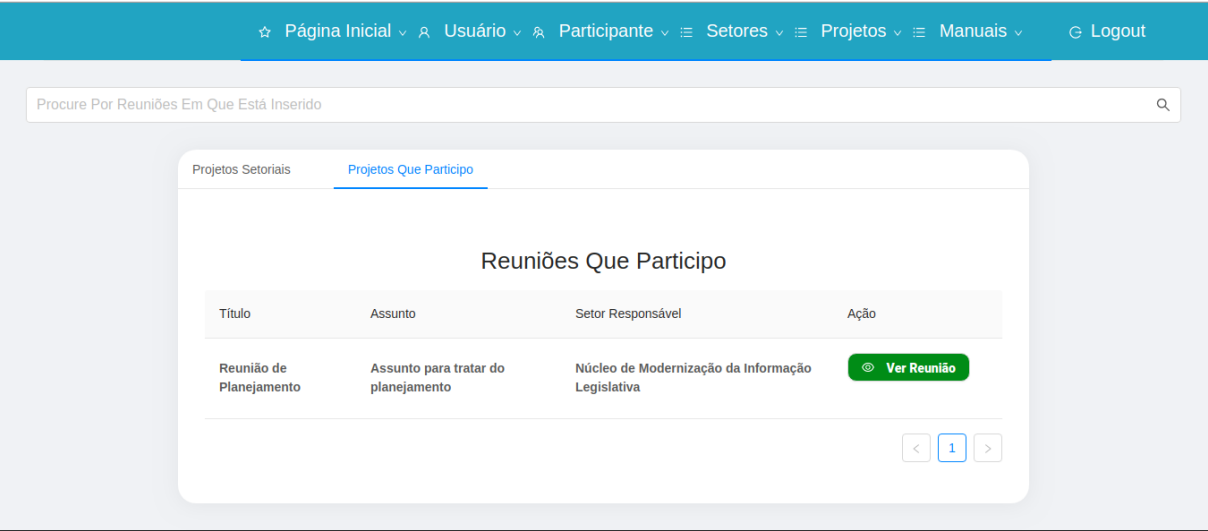


Figura 21 – Reuniões Que Participo. Fonte: Própria

Reunião de Planejamento		
Título da Reunião	Assunto da Reunião	Setor Responsável
Reunião de Planejamento	Assunto para tratar do planejamento	Núcleo de Modernização da Informação Legislativa
Data de Abertura	Data de Encerramento	Hora de Inicio
12/12/2019	20/12/2019	04:18:41
Hora de Encerramento	Status da Reunião	
07:18:41	● Confirmada	
Usuários Confirmados na Reunião	Tópicos	Regras da Reunião
Administrador Victor Hugo Lopes Mota	Tópico01	A regra é clara

Figura 22 – Ata. Fonte: Própria

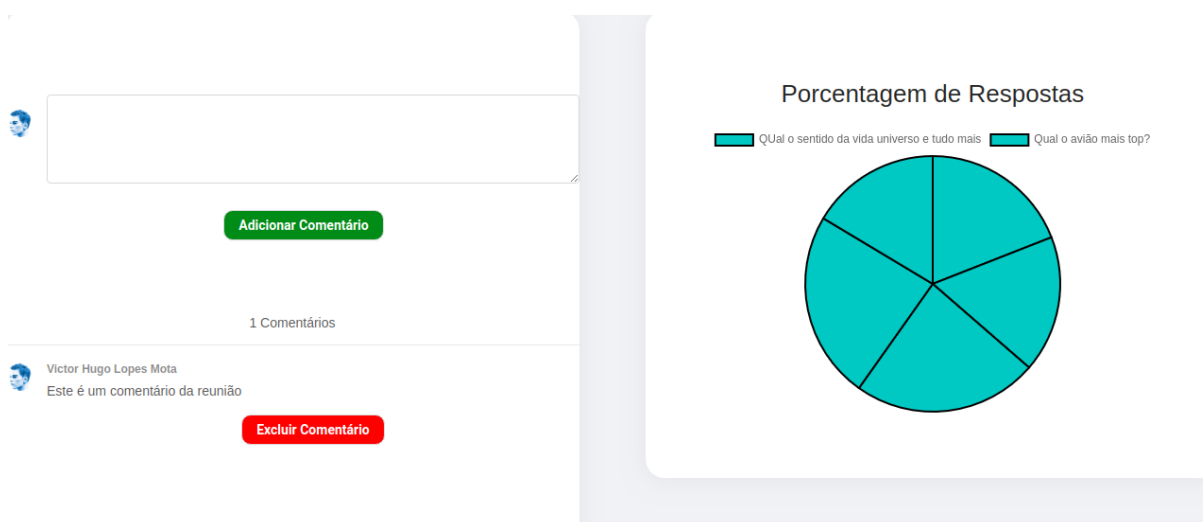


Figura 23 – Resultados Questionário. Fonte: Própria

6 Considerações Finais

Referências

- AGILE, M. *Manifesto for Agile Software Development*. 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso em: 05.05.2019. Citado na página 21.
- ALLEN, L.-W. Meetings as a positive boost? how and when meeting satisfaction impacts employee empowerment. *Journal of Business Research*, 2016. Acesso em: 25.04.2019. Citado na página 14.
- DAVID, G. *How to save the world (or at least yourself) from bad meetings*. 2013. Disponível em: https://www.ted.com/talks/david_grady_how_to_save_the_world_or_at_least_yourself_from_bad_meetings. Acesso em: 22.04.2019. Citado na página 14.
- DRAKE, B. 37 billion is lost every year on these 12 meeting mistakes. *Business Insider*, 2014. Disponível em: <https://www.businessinsider.com/37-billion-is-lost-every-year-on-these-meeting-mistakes-2014-4>. Acesso em: 29.04.2019. Citado na página 16.
- FABIANE, S. *Ciclo de Vida do Scrum*. 2016. Disponível em: <https://br.pinterest.com/pin/417357090459098830/>. Acesso em: 05.05.2019. Citado 2 vezes nas páginas 8 e 22.
- FOWLER, M. The new methodology. 2005. Disponível em: https://moodle2016-17.ua.es/moodle/pluginfile.php/69142/mod_resource/content/1/martin-fowler-the-new-methodology.pdf. Acesso em: 05.04.2019. Citado na página 23.
- GONÇALVES, G. *Papeis Scrum*. 2016. Disponível em: <https://guildadocodigo.atelie.software/como-cada-um-dos-pap%C3%A9is-do-scrum-contribui-para-o-sucesso-do-seu-projeto-b6e8b5f01e57>. Acesso em: 03.06.2019. Citado 2 vezes nas páginas 8 e 24.
- HARVARD, B. *Estimate the Cost of a Meeting with This Calculator*. 2016. Disponível em: <https://hbr.org/2016/01/estimate-the-cost-of-a-meeting-with-this-calculator>. Acesso em: 29.04.2019. Citado na página 16.
- JAIRSON, R. Pmbok e asy – proposta de um ambiente de estudo para gerentes de projetos. *UNIVERSIDADE FEDERAL DE PERNAMBUCO*, 2003. Acesso em: 08.07.2019. Citado na página 17.
- JHONATAS, T. *PCM Descomplicado – Planejamento e Controle de Manutenção*. 2018. Disponível em: <https://engeteles.com.br/pcm-descomplicado/>. Acesso em: 04.05.2019. Citado 2 vezes nas páginas 8 e 20.
- KERZNER, H. Gestão de projeto 2ª edição. *Bookman Editora*, 2009. Acesso em: 03.05.2019. Citado na página 17.
- LAMECK, O. *Uma Breve Introdução ao Kanban*. 2016. Disponível em: <https://blog.diferencialti.com.br/uma-breve-introducao-ao-kanban/>. Acesso em: 05.05.2019. Citado 2 vezes nas páginas 8 e 23.

- LEACH, G. Meetings at work: Perceived effectiveness and recommended improvements. *Journal of Business Research*, 2015. Disponível em: <<https://www.sciencedirect.com/science/article/abs/pii/S0148296315000879>>. Acesso em: 02.06.2019. Citado na página 14.
- MACLEOD, L. Conducting a well-managed meeting. *Physician Executive*, 2011. Disponível em: <<http://dev.orgwise.ca/sites/osi.ocasi.org.stage/files/Conducting%20a%20Well-Managed%20Meeting.pdf>>. Acesso em: 02.06.2019. Citado na página 14.
- PAGOTTO, T. et al. *Ciclo de Vida do Scrum Solo*. 2016. Disponível em: <<https://scrumsolo.wordpress.com/>>. Acesso em: 09.07.2019. Citado 3 vezes nas páginas 8, 25 e 26.
- PERLOW, H. Stop the meeting madness: How to free up time for meaningful work. *Harvard Business Review*, 2017. Disponível em: <<https://hbr.org/2017/07/stop-the-meeting-madness>>. Acesso em: 02.06.2019. Citado na página 14.
- PMBOK, P. *Um Guia do Conhecimento em Gerenciamento de Projetos 5ª edição*. [S.l.]: Saraiva, 2012. Acesso em: 30.04.2019. Citado 2 vezes nas páginas 17 e 18.
- PMPDIGITAL, W. *Conceitos – Fases x Grupos de Processos de Gerenciamento*. 2009. Disponível em: <<https://pmpdigital.wordpress.com/2009/05/21/conceitos-fases-x-grupos-de-processos-de-gerenciamento/>>. Acesso em: 08.07.2019. Citado 2 vezes nas páginas 8 e 18.
- ROGELBERG, S. Meetings and more meetings: The relationship between meeting load and the daily well-being of employees. *Group Dynamics: Theory, Research, and Practice*, 2005. Disponível em: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.74.2962&rep=rep1&type=pdf>>. Acesso em: 02.06.2019. Citado na página 15.
- SENADO, F. *Estrutura Senado*. 2019. Disponível em: <<https://www12.senado.leg.br/institucional/estrutura>>. Acesso em: 06.05.2019. Citado 2 vezes nas páginas 8 e 32.
- SOARES, S. Metodologias Ágeis extreme programming e scrum para o desenvolvimento de software. 2009. Acesso em: 05.04.2019. Citado na página 21.
- SOMMERVILLE, I. *Engenharia de Software 9ª edição*. [S.l.]: Pearson Education do Brasil, 2011. Acesso em: 01.04.2019. Citado 4 vezes nas páginas 20, 26, 27 e 28.
- STANDISH, G. Failure record. *Standish Group Report Chaos*, 2014. Disponível em: <<https://www.projectsmart.co.uk/white-papers/chaos-report.pdf>>. Acesso em: 05.04.2019. Citado na página 21.
- VIEIRA, R. *Casos de Uso*. 2015. Disponível em: <<https://medium.com/operacionalti/uml-diagrama-de-casos-de-uso-29f4358ce4d5>>. Acesso em: 05.06.2019. Citado 2 vezes nas páginas 8 e 20.

Apêndices

APÊNDICE A – Histórias de Usuário

Funcionalidade	História de Usuário
Fazer Login	Eu, como Administrador, desejo poder realizar login, para assim poder acessar as funcionalidades do sistema.
Gerenciar Reuniões	Eu, como Administrador, desejo poder criar, editar, excluir as reuniões, para assim conseguir melhor conduzir as reuniões.
	Eu, como Administrador, desejo que apenas eu possa gerenciar reuniões, para assim ter total controle sobre a reunião.
Gerenciar Tópicos da Reunião	Eu, como Administrador, desejo poder criar os tópicos da reunião, para assim ser mais objetivo com o desenvolvimento da reunião.
	Eu, como Administrador, desejo que os tópicos criados possam ser adicionados a reuniões.
Gerenciar Projetos	Eu, como Administrador, desejo poder criar, editar, excluir, e procurar projetos, para que assim tenha um controle sobre os mesmos e sobre as reuniões.
	Eu, como Administrador, desejo que apenas eu tenha acesso a permissões de adição, edição e exclusão sobre os projetos.
Gerenciar Regras de Conduta	Eu, como Administrador, desejo poder criar as regras de conduta, para que assim a reunião não tenha dispersões de foco.
	Eu, como Administrador, desejo que as regras de conduta criadas possam ser adicionadas às reuniões.

Tabela 5 – Histórias de Usuário Administrador Parte 1

Funcionalidade	História de Usuário
Gerenciar Relatórios de Reuniões	Eu, como Administrador, desejo que o sistema consulte a presença dos participantes.
	Eu, como Administrador, desejo que o sistema calcule o total de presença dos participantes.
	Eu, como Administrador, desejo que o sistema exiba o relatório de participantes, contendo a média de presença dos participantes, melhores participantes para se convocar, participantes duvidosos e piores participantes.
	Eu, como Administrador, desejo que seja possível visualizar os níveis de satisfação das reuniões.
	Eu, como Administrador, desejo que seja mostrada o total de horas utilizadas ao longo do projeto.
Gerenciar Usuários	Eu, como Administrador, desejo poder criar e excluir usuários do sistema, para assim conseguir ter controle sobre os usuários do sistema.
	Eu, como Administrador, desejo que apenas eu possa adicionar outros usuário.
Gerenciar Participantes da Reunião	Eu, como Administrador, desejo poder adicionar e remover os participantes das reuniões.
	Eu, como Administrador, desejo que quando a reunião seja marcada, não seja mais possível retirar um participante da reunião.
	Eu, como Administrador, desejo que ao incluir um participante a reunião, o sistema exporte as informações deste para a ATA.
Gerenciar Marcar Reunião	Eu, como Administrador, desejo que eu possa incluir, editar e excluir os dados da reunião.
	Eu, como Administrador, desejo poder visualizar os participantes confirmados ou não à reunião.
	Eu, como Administrador, desejo que ao marcar reunião, o status da reunião mude para "Agendada" e exiba data e hora.
	Eu, como Administrador, desejo que caso a reunião seja cancelada, todos os participantes devem receber por <i>email</i> a mensagem.
	Eu, como Administrador, desejo que quando faltar menos que 48 horas para acontecer a reunião, caso ainda não tenha a confirmação de pelo menos 75% dos convocados, o sistema deve cancelar a reunião.
Gerenciar Questionário de Avaliação	Eu, como Administrador, desejo poder criar, editar e excluir o questionário de avaliação da reunião, para assim ter um <i>feedback</i> sobre a mesma.

Tabela 6 – Histórias de Usuário Administrador Parte 2

Funcionalidade	História de Usuário
Gerenciar Questionário de Avaliação	Eu, como Administrador, desejo que seja possível realizar <i>download</i> do questionário.
	Eu, como Administrador, desejo que o sistema só permita a criação da ATA, após o questionário ser gerado.
	Eu, como Administrador, desejo que as perguntas fiquem disponíveis dentro de uma reunião específica, quanto seja possível utilizar-las em outro questionário.
	Eu, como Administrador, desejo que ao excluir uma pergunta do questionário, essa pergunta não seja excluída de um questionário anterior.
	Eu, como Administrador, desejo que apenas eu possa visualizar as respostas do questionário.
Gerenciar ATA da Reunião	Eu, como Administrador, desejo criar, editar, excluir a ATA da reunião.
	Eu, como Administrador, desejo que o sistema mostre os tópicos da reunião previamente adicionados.
	Eu, como Administrador, desejo que seja mostrado os dados anteriormente cadastrados na ATA.
	Eu, como Administrador, desejo que a ATA seja possível converter em PDF.
Gerenciar Setor	Eu, como Administrador, desejo poder criar, editar e excluir um setor, para que assim consiga alocar tanto outros administradores, quanto participantes da reunião em seus devidos locais de trabalho.

Tabela 7 – Histórias de Usuário Administrador Parte 3

Funcionalidade	História de Usuário
Buscar Projetos	Eu, como Participante da Reunião, desejo poder procurar qualquer projeto no sistema.
Gerenciar Perfil	Eu, como Participante da Reunião, desejo poder editar as informações do meu perfil, para assim manter minhas informações atualizadas.
	Eu, como Participante da Reunião, desejo poder excluir meu perfil.
Inserir Comentários	Eu, como Participante da Reunião, desejo poder inserir comentários nas reuniões em que eu participar, para aumentar o <i>feedback</i> da reunião para os administradores.
Responder Questionário de Avaliação	Eu, como Participante da Reunião, desejo poder responder o questionário de avaliação para dar um <i>feedback</i> da reunião para os administradores.
Gerenciar Setor	Eu, como Participante da Reunião, desejo poder alterar meu setor.

Tabela 8 – Histórias de Usuário Participante

APÊNDICE B – Histórias Técnicas

História Técnica	Descrição
Criar Ambiente Estável	Eu, como desenvolvedor, quero criar um ambiente estável de desenvolvimento, para assim facilitar a manutenção e evolução futura do sistema.
Escolher Ferramenta de Deploy	Eu, como desenvolvedor, quero escolher uma ferramenta para ao final de todo o desenvolvimento do projeto, entregar o produto final ao cliente.
Passar as Histórias de Usuário Para o Zenhub	Eu, como desenvolvedor, quero passar as histórias de usuário presentes no backlog do produto para o <i>Zenhub</i> , para assim manter um controle do que está sendo desenvolvido.
Realizar Deploy da Aplicação	Eu, como desenvolvedor, quero realizar o deploy da Aplicação, para que assim o cliente possa usufruir do mesmo

Tabela 9 – Histórias Técnicas

APÊNDICE C – Figuras

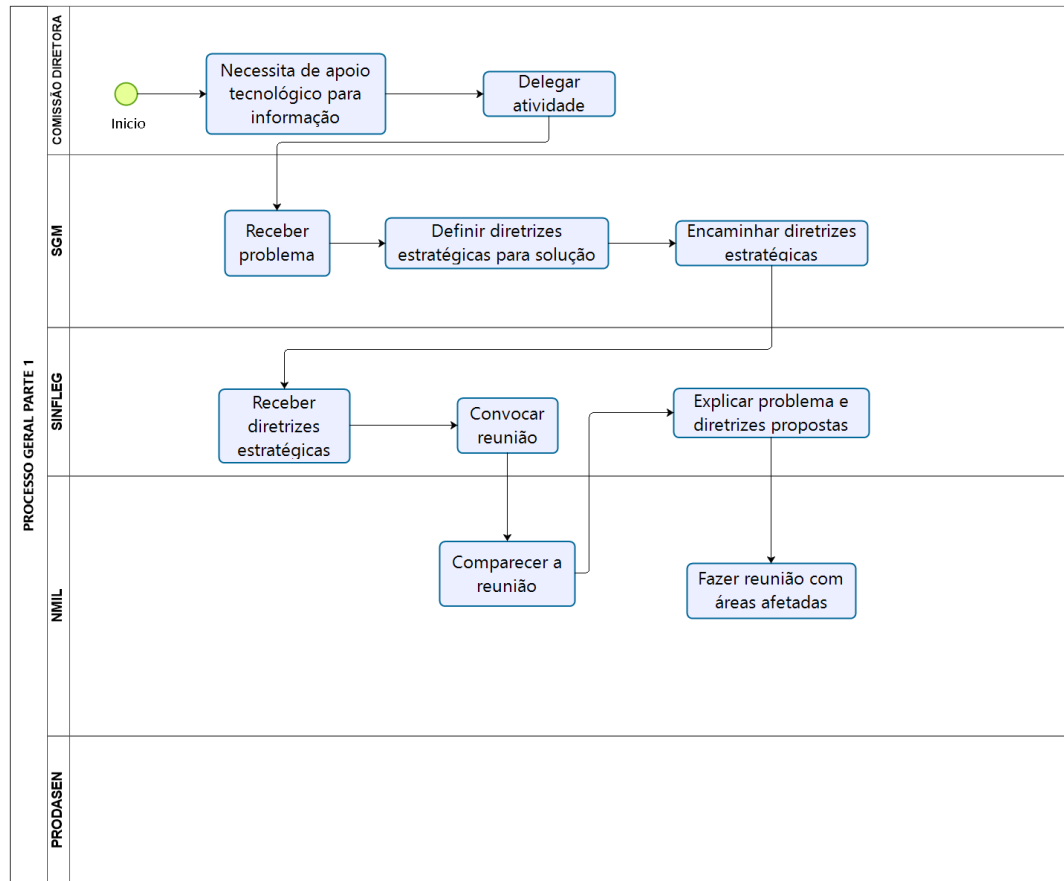


Figura 24 – Modelagem de Processo Geral 1 Parte 1. Fonte: Própria

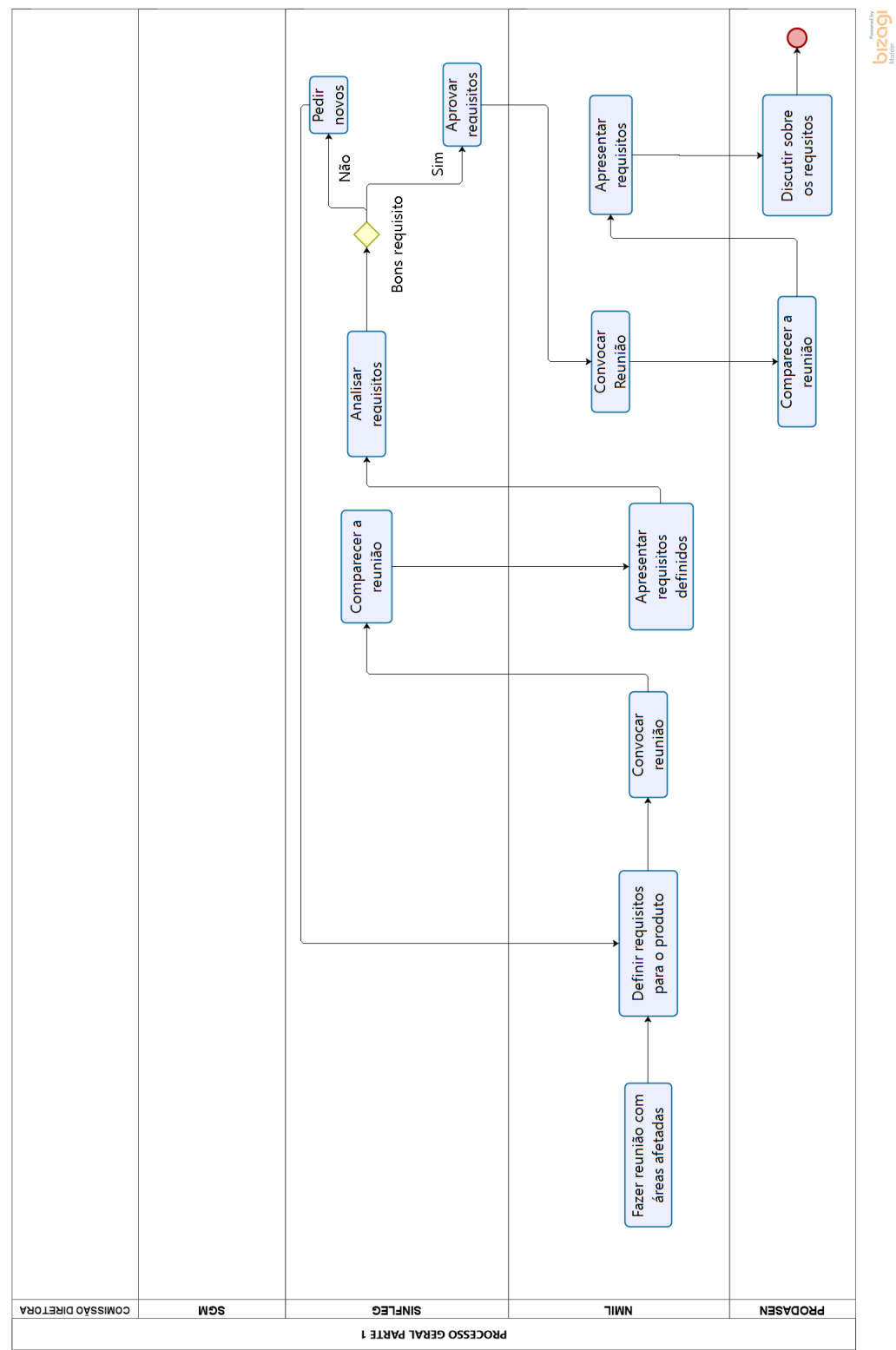


Figura 25 – Modelagem de Processo Geral 1 Parte 2. Fonte: Própria

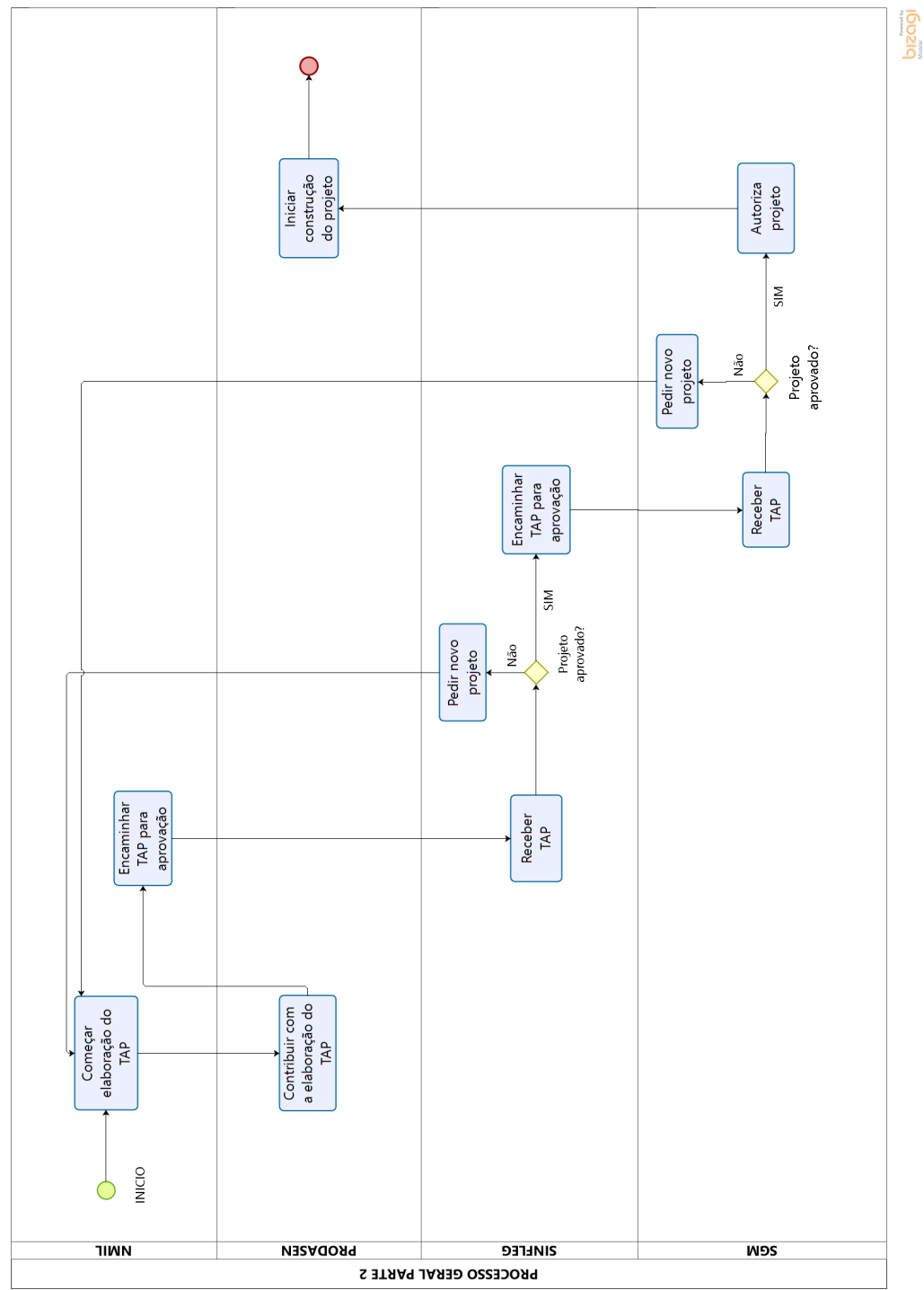


Figura 26 – Modelagem de Processo Geral 2 Parte 1. Fonte: Própria

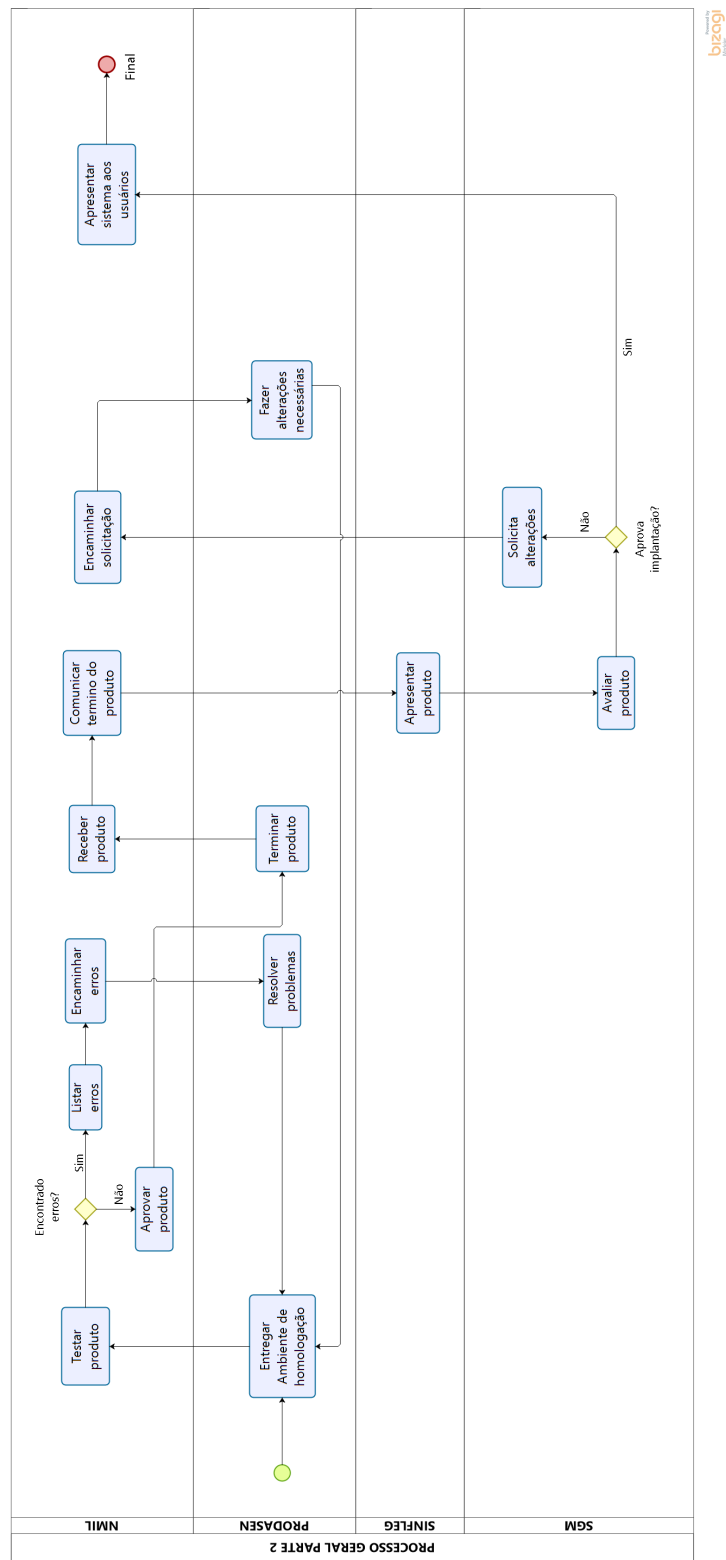


Figura 27 – Modelagem de Processo Geral 2 Parte 2. Fonte: Própria

APÊNDICE D – Protótipos

O protótipo apresenta uma janela de navegador com uma barra de endereços vazia. No topo centralizado, há um ícone de logo (um quadrado com uma 'X') seguido pelo texto "Logo". Centralizado na área principal, há um formulário retangular com o título "Acessar o GRATA". O formulário contém dois campos de entrada: "Usuário" com o texto "geovannis" e "Senha" com caracteres de máscara ("*****"). Abaixo dos campos, há um botão "Entrar".

Figura 28 – Gerenciar Login. Fonte: Própria

Projetos

Novo Projeto

Projeto... Pesquisar

Título	Patrocinador	Gerente	Status
ABC	Zagalo	Robinho	Finalizado

Novo Projeto

Título: Gerente:

Patrocinador:

Áreas envolvidas:

Criar Projeto Cancelar

<< 1 2 3 4 5 >> Editar Excluir

Figura 29 – Gerenciar Projeto. Fonte: Própria

Logo

[Sair](#)

[Projetos](#) [ABC](#) [Abertura](#) [Marcar Reunião](#)

Marcar Reunião

Participantes: [Inclu...](#)

Assunto:

Local:

Hora de início:

Hora de término:

[Perspectiva gerencial](#) [Pauta da reunião](#)

[Confirmados](#) [Cancelar Reunião](#) [Marcar Reunião](#)

Figura 30 – Marcar Reunião. Fonte: Própria

The wireframe depicts a web browser window with a standard address bar and navigation icons. The page layout includes a header with a logo placeholder and a 'Sair' (Logout) link. A breadcrumb trail shows the user's path: 'Projetos' > 'ABC' > 'Abertura' > 'Pauta da reunião'. The main heading is 'Pauta da Reunião'. The content is organized into four sections, each with a title and a text input area: 'Introdução:', 'Revisão da perspectiva gerencial;', 'Regras de conduta:', and 'Tópicos da reunião:'. The 'Tópicos da reunião:' section features a table with a header row containing 'Entrada de texto com botão' and 'Adicionar'. Below the header, there are four rows, each with a radio button and the text 'Caixa de seleção'. The first row has the radio button selected. At the bottom of this table, there are 'Editar' and 'Excluir' buttons. To the right of the table, there are three stacked buttons: 'Editar', 'Excluir', and 'Salvar'.

Logo

[Sair](#)

Projetos ABC Abertura Pauta da reunião

Pauta da Reunião

- Introdução:

Text area

- Revisão da perspectiva gerencial;

Text area

- Regras de conduta:

Text area

- Tópicos da reunião:

Entrada de texto com botão	Adicionar
<input type="radio"/> Caixa de seleção	
<input type="radio"/> Caixa de seleção	
<input type="radio"/> Caixa de seleção	
<input checked="" type="radio"/> Caixa de seleção	

Editar Excluir

Editar Excluir Salvar

Figura 31 – Pauta Reunião. Fonte: Própria



The image shows a web browser window with a questionnaire titled "Avalie a Reunião". The browser's address bar is empty, and the page has a header with a "Logo" placeholder, a "Sair" button, and navigation links for "Projetos" and "Reuniões". The questionnaire consists of a table with 10 rows. The first two rows contain specific questions about meeting timing, while the remaining eight rows are empty for additional questions. Each row has columns for a number, a question, and a response area with radio buttons for "SIM" and "NÃO". A blue rectangular box highlights the first two rows of the table. At the bottom right of the table, there is an "Imprimir" button. Below the table, there are two buttons: "Salvar Resposta" and "Cancelar".

Avalie a Reunião		
Nº	Perguntas	Resposta
1	A reunião começou na hora marcada?	<input checked="" type="radio"/> SIM <input type="radio"/> NÃO
2	A reunião terminou na hora fixada?	<input type="radio"/> SIM <input checked="" type="radio"/> NÃO
3		
4		
5		
6		
7		
8		
9		
10		

Imprimir

Salvar Resposta Cancelar

Figura 32 – Questionário. Fonte: Própria

APÊNDICE E – Manual do Usuário

E.1 Introdução

Seja muito bem vindo ao manual do usuário do Grata (Gerenciador de Reuniões e Atas), a ferramenta que irá te ajudar você, caro gerente de projetos, a melhorar suas reuniões.

Este *software* foi desenvolvida para ser uma ferramenta completamente gratuita, com código aberto, e que permite uma customização das empresas para seus problemas específicos. Tudo relacionado podem ser encontrados nos seguintes links: [TCC](#), [Backend](#), e [Frontend](#).

Em caso de qualquer dúvida, pertinentes a este projeto, por favor entrar em contato com o desenvolvedor deste projeto pelo email: mrvector042@gmail.com.

A ferramenta foi desenvolvida para ser gratuita e adaptável as necessidades das empresas, e a customização de novas funcionalidades, servidores, banco de dados e qualquer coisa além do desenvolvido neste projeto, ficará a cargo da empresa.

E.2 Instalação

O Grata é uma ferramenta online, não necessitando de instalação, contudo para que seja realizada alterações no mesmo deve ser realizado via código. Pedimos que qualquer desejo de mudança na ferramenta, seja realizada a partir de um *"fork"* nos repositórios do *Github*, linkados acima.

Para ajudar no desenvolvimento de novas funcionalidades, ou ainda alterações no formato original do Grata, todos os códigos relacionados a ferramenta foram implementados utilizando o *Docker*. A seguir, como instalar e utilizar o *Docker* do projeto:

E.2.1 Windows

E.2.1.1 Docker

Para instalar o *Docker* no sistema operacional *Windows*, basta seguir os passos do link a seguir: [Docker-Windows](#).

E.2.2 Ubuntu

E.2.2.1 Docker

Para instalar o *Docker* no sistema operacional *Ubuntu* e suas derivações, basta seguir os passos do link a seguir: [Docker-Ubuntu](#).

E.2.2.2 Docker-Compose

Para instalar o *Docker-Compose* no sistema operacional *Ubuntu* e suas derivações, basta seguir os passos do link a seguir: [Docker-Compose-Ubuntu](#).

No sistema operacional *Ubuntu* e suas derivações, em ambas os projetos (*frontend/backend*), devem ser rodado os seguintes comandos:

- `docker-compose build` (Para criar a build do projeto);
- `docker-compose up` (Para rodar o projeto).

E.3 Como Utilizar?

E.3.1 Tela Inicial e Login

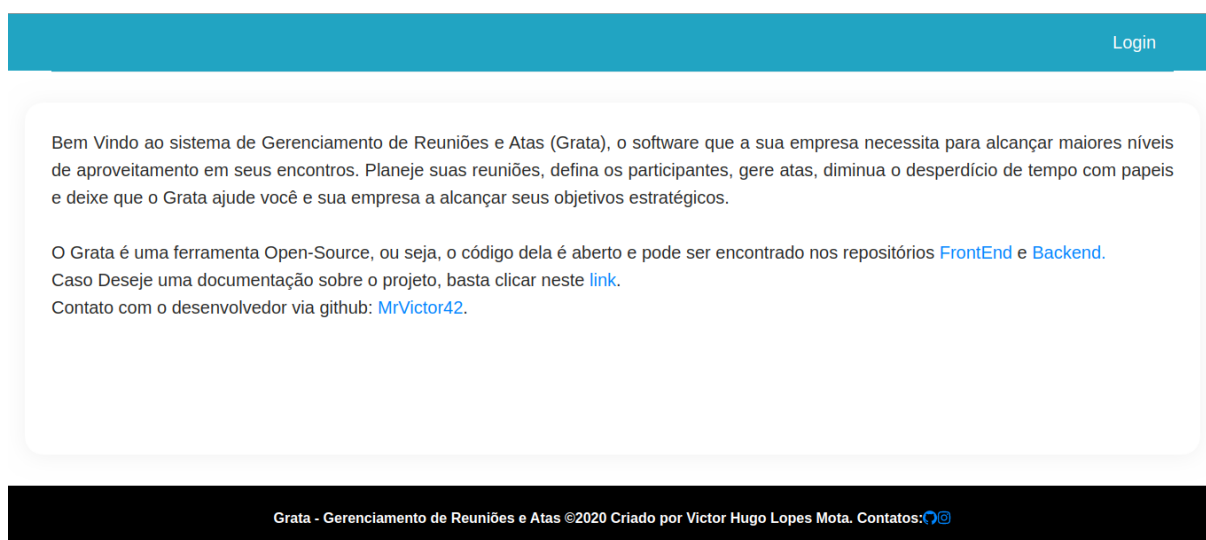
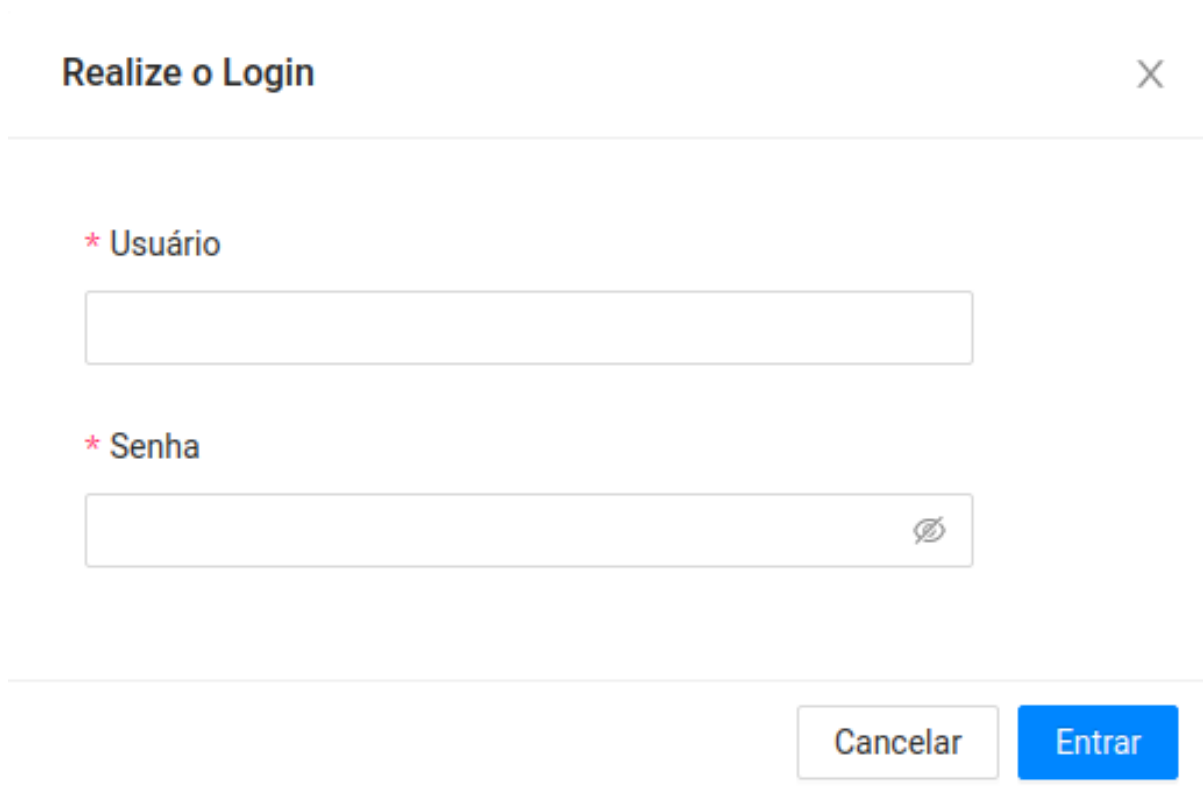


Figura 33 – Tela Inicial. Fonte: Própria

A figura 33, mostra a tela inicial da ferramenta, com uma rápida explicação sobre o que é o *software* em questão. Para começar a acessar os recursos da ferramenta, basta clicar em "*Login*".



The image shows a login window titled "Realize o Login" with a close button (X) in the top right corner. Below the title bar, there are two input fields. The first field is labeled with a red asterisk and the text "Usuário". The second field is labeled with a red asterisk and the text "Senha" and includes a toggle icon (an eye with a diagonal line) on its right side. At the bottom right of the form, there are two buttons: a light gray button labeled "Cancelar" and a blue button labeled "Entrar".

Figura 34 – Login. Fonte: Própria

Inserindo suas credenciais na ferramenta, você poderá acessar adequadamente as funções do sistema.

No primeiro acesso na ferramenta, entre em contato com o desenvolvedor.

E.3.2 Perfis de Usuários

Os usuários no Grata são divididos em dois grupos: Administrador e Participante da Reunião.

O **Administrador**, é aquele que tem as maiores liberdades dentro do sistema, podendo adicionar novos usuários, criar setores, gerencia reuniões, comentar nas reuniões, e é ele que prove os insumos para uma reunião. Este perfil também pode excluir usuários do sistema e alterar as permissões dos demais usuários, ou seja, ele pode transformar um participante da reunião em um administrador.

O **Participante**, é aquele que deve ser convidado pelo administrador a compor uma reunião, além de poder comentar sobre a mesma e estar por dentro do que rumo dos projetos em que está participando.