DUBIUM

VISÃO DO PRODUTO E PROJETO

Versão 1.2

Histórico de Revisão

Data	Versão	Descrição	Autor
26/10	1.0	Criação do documento	Gian, Giulia, Helder, Júlio e Silas
02/11	1.1	Atualização do documento	Gian, Giulia, Helder, Júlio e Silas
09/11	1.1	Atualização do documento	Giulia, Helder e Silas
10/11	1.0	Criação do GitPages	Júlio
14/11	1.2	Atualização do documento	Eduarda, Gian, Giulia, Helder, Júlio e Silas
15/11	1.1	Atualização do GitPages	Júlio
15/11	1.0	Criação do Read.Me	Giulia
16/11	1.2	Gravação do Vídeo da Unidade 1	Gian
16/11	1.3	Atualização do Documento	Gian, Giulia, Helder, Júlio e Silas

Sumário

VISÃO DO PRODUTO E PROJETO

1 VISÃO GERAL DO PRODUTO

Declaração de Posição do Produto

Para	Alunos e Professores da FGA
Quem	Deseja postar suas dúvidas, sabendo que serão vistas por outros alunos e respondidas sequencialmente, caso saibam as soluções.
O Dubium	É um sistema web de auxílio acadêmico
Que	Auxílio no acesso do esclarecimento de dúvidas dentro de múltiplos conteúdos e no agendamento de prestações de ajuda presencial
Ao contrário	Stack Overflow, Brainly
Nosso produto	Possuirá um acesso a aplicação por meio de cadastro dos alunos da UnB, sendo gratuito e mais direcionado a resolução de dúvidas acadêmicas específicas. Também, através de uma gamificação, irá incentivar os alunos a auxiliarem a usarem-no e se sentirem mais motivados para ajudarem os outros.

Objetivos do Produto

O objetivo do Dubium é ajudar / dar suporte para a resolução de dúvidas específicas de alunos, provendo agendamentos de reuniões presenciais, se for do desejo dos solicitantes, ou simplesmente solucioná-las via fórum. Além de ajudar professores a perceberem o maior foco de dificuldade em suas matérias, fazendo-os melhorar o planejamento de conteúdos complexos.

Tecnologias a Serem Utilizadas

JavaScript

Typescript

MySQL

NodeJS

NestJS

Material UI

React

Jest

2 VISÃO GERAL DO PROJETO

Organização do Projeto

Papel	***Atribuições***	***Responsável***	***Participantes***
Desenvolvedor	*Codificar o produto, codificar testes unitários, realizar refatoração*	*Helder*	*Gian, Giulia,, Julio, Silas*
Dono do Produto	*Atualizar o escopo do produto, organizar o escopo das sprints, validar as entregas*	*Gian*	*Giulia, Helder, Julio, Silas*
Analista de Qualidade	*Garantir a qualidade do produto, garantir o cumprimento do conceito de pronto, realizar inspeções de código*	*Silas*	*Gian, Giulia, Helder, Julio*

Mestre do Scrum	*Delegar tarefas, duração da sprint e sprint review*	*Giulia*	*Gian, Helder, Julio, Silas*

Planejamento das Fases e/ou Iterações do Projeto

Sprint	***Produto (Entrega)***	***Data Início***	***Data Fim***
Sprint 1	*Definição do Produto*	*25/10/22*	*01/10/22*
Sprint 2	*Alinhamento do Produto*	*01/11/22*	*08/11/22*
Sprint 3	*Análise Visão de Produto e Projeto;*	*08/11/22*	*15/11/22*
Sprint 4	*Visão de Produto e Projeto*	*15/1122*	*22/11/22*
Sprint 5	*Definição do Backlog do Produto;*	*22/11/22*	*29/11/22*
Sprint 6	*Definição de MVP*	*29/11/22*	*06/12/22*
Sprint 7	*Refinamento de backlog da sprint;*	*06/12/22*	*13/12/22*
Sprint 8	*Validação do MVP-1*	*27/12/22*	*03/01/22*
Sprint 9	*MVP-1*	*03/01/22*	*10/01/22*
Sprint 10	*Validação do MVP-2*	*24/01/22*	*31/01/22*
Sprint 11	*MVP-2*	*31/01/22*	*07/02/22*

Matriz de Comunicação

A comunicação do grupo se dará principalmente por meio de reuniões semanais via Google Meets, diálogos via Whatsapp e Dailys feitas via Telegram.

Descrição	***Área/ Envolvidos***	***Periodicidade***	***Produtos Gerados***
Daily	*Equipe do Projeto*	*- Diário*	*- Relatório da daily* *- Relatório de situação do projeto* *- Atualização sobre o andamento do projeto*
Sprint Planning	*Equipe do Projeto e Cliente*	*- Semanal*	*- Relatório Sprint Planning* *- Planejamento do que será feito no ciclo da Sprint*
Sprint Review	*Equipe do Projeto e Cliente*	*- Junto ao Sprint Planning*	*- Relatório Sprint Review* *- Validação do Produto*
Retrospectiva	*Equipe do Projeto*	*- Quinzenalmente*	*- Verificação do Produto*
Comunicar a situação do projeto	*Equipe do Projeto e Professor*	*Mensalmente*	*Apresentações da Entrega de cada Unidade*

Gerenciamento de Riscos

A análise e gerenciamento de riscos referem-se à identificação dos possíveis pontos que podem representar riscos para o projeto. Precisam ser acompanhados, a cada sprint, se referindo assim, ao projeto como um todo e não apenas ao produto.

Segundo Charette, existem três tipos de riscos de Software (PRESSMAN,2006):

Riscos de projeto mostram problemas potenciais de orçamento, cronograma, organizacionais que impactam o projeto. (PRESSMAN,2006)

Riscos técnicos perturbam a qualidade e a entrega do software. Também mostram problemas potenciais de projeto, implementação, interface, verificação e manutenção. (PRESSMAN,2006)

Riscos de negócio ameaçam a viabilidade do software e do produto. Existem cinco principais riscos de negócios que são: (1) criar um excelente produto ou sistema que ninguém realmente quer (risco de mercado), (2) criar um produto que não se encaixe mais na estratégia geral de negócios da empresa (risco estratégico), (3) criar um produto que a equipe de vendas não sabe como vender (risco de vendas), (4) perda de suporte da alta gerência devido à mudança no foco ou mudança de profissionais (risco gerencial), e (5)perda do orçamento ou do comprometimento dos profissionais (riscos de orçamento). (PRESSMAN,2006)

FIGURA 28.1
Avaliação de impacto
***Fonte:* (Boe89)**

Componentes / Categoria		Desempenho	Suporte	Custo	Cronograma
Catastrófico	1	Falha em satisfazer o requisito resultaria em falha da missão		A falha resulta em aumento de custo e atrasos no cronograma com valores previstos que excedem $ 500 mil	
	2	Degradação significativa até não cumprimento do desempenho técnico	Software que não responde com agilidade ou que é difícil de dar suporte	Dificuldades financeiras significativas, provável estouro no orçamento	Data de entrega não exequível
Crítico	1	Falha em atender o requisito degradará o desempenho do sistema até um ponto no qual o sucesso da missão é questionável		Falha resulta em atrasos operacionais e/ou aumento de custos com valores estimados entre $ 100 mil e $ 500mil	
	2	Alguma redução no desempenho técnico	Pequenos atrasos nas modificações de software	Alguma falta de recursos financeiros, possíveis estouros de orçamento	Possível atraso na data de entrega
Marginal	1	Falha em atender o requisito resultaria na degradação de missão secundária		Custos, impactos e/ou atrasos de cronograma recuperáveis com valores estimados de $ 1 mil a $ 100 mil	
	2	De mínima a pequena redução no desempenho técnico	Suporte responsivo de software	Recursos financeiros suficientes	Cronograma realístico e possível
Negligenciável	1	Falha em atingir o requisito criaria inconveniência ou impacto não operacional		Erro resulta em pequeno impacto no custo e/ou cronograma com valor esperado de menos de $ 1 mil	
	2	Nenhuma redução do desempenho técnico	Software facilmente suportável	Possível sobra no orçamento	Data de entrega pode ser antecipada

Notas: (1) Potencial consequência de erros ou falhas de software não detectadas.
(2) Potencial consequência se o resultado desejado não é obtido.

Figura 1, Tabela de Previsão de Riscos (PRESSMAN,2006)

Os riscos do projeto devem ser acompanhados e atualizados a cada ciclo.

Histórico de Riscos:

Sprint	***Risco Encontrado***	***Nível***
3	*Adição de um novo membro na equipe*	*Negligenciável*

Critérios de Replanejamento

Os critérios de replanejamento referem-se à identificação dos pontos que, se ocorrerem, necessariamente, vão causar um replanejamento do projeto. Precisam ser acompanhados a cada sprint, se referindo assim, ao projeto como um todo e não apenas ao produto.

- *Entrada de um novo membro na equipe*

Sprint	***Solução Encontrada***	***Resultado Esperado***
3	*Explicações sobre o projeto e nova divisão de tarefas*	*Capacitação da Equipe*

Os critérios de replanejamento do projeto devem ser acompanhados e atualizados a cada ciclo. E, aplicados, conforme necessidade.

3 PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

3.1 Metodologia

Baseado na proposta do Sommerville (2018), foi respondido um conjunto de questões distintas para definir a abordagem que melhor se encaixa ao projeto e time.

Questões técnicas:

- *Qual é o tamanho do sistema que está sendo desenvolvido? É um software de pequeno porte.*
- *Que tipo de sistema está sendo desenvolvido? Aplicação Web.*
- *O sistema está sujeito a controle externo? Sim, o sistema será consumido pela instituição de ensino.*

Questões organizacionais:

- *É importante ter uma especificação e um projeto (design) bem detalhados antes de passar para a implementação - talvez por motivos contratuais? Como será utilizada uma metodologia de design orientado à função, não tem necessidade de ser bem definida antes da implementação.*
- *É realista uma estratégia de entrega incremental, na qual o software é entregue aos clientes ou outros stakeholders e um rápido feedback é obtido? Por conta da proximidade do Product Owner(PO) com o cliente e a equipe de desenvolvimento, é realista.*
- *Os representantes do cliente estarão disponíveis e dispostos a participar do time de desenvolvimento? Sim, uma vez que o representante faz parte da equipe.*
- *Existem questões culturais que possam afetar o desenvolvimento do sistema? Não.*

Considerando os resultados obtidos e as necessidades da equipe em priorizar a flexibilidade e o desenvolvimento iterativo, escolheu-se a metodologia ágil. Dessa forma, a abordagem SCRUM foi adotada como ciclo de vida e processo de desenvolvimento baseado no XP - Extreme programming - assim como pelos seguintes motivos:

- *Feedback contínuo do cliente*

- *Equipe pequena*
- *Construção dos requisitos que, junto com o software, permitem qualquer mudança necessária a ser feita sem prejudicar o desenvolvimento da aplicação*
- *Realização do controle das atividades da sprint*
- *Atua de maneira evolutiva, com refinamentos sucessivos de requisitos e solução*
- *Método de Pair Programming*

3.2 Ferramentas

Para a execução dessa metodologia, escolheu-se as seguintes Ferramentas de organização e controle da equipe:

- *Trello - para controle e gerenciamento dos resultados das sprints*
- *Figma - para protótipos do front-end*
- *Miro - para gerenciamento de ideias*

3.3 Processos e Procedimentos

De acordo com a abordagem SCRUM, definiu-se:

- *Sprint:* com duração de 1 semana;
- *Dailys:* ao fim do dia;
- *Backlog do Produto e da Sprint:* definidos em cada Planning;
- *Planning:* realizada no primeiro dia de cada sprint;
- *Implementação:* codificação - modelagem de banco de dados, programação back-end e front-end, criação de cenários de testes e documentação dos requisitos;
- *Review:* ao final de cada ciclo para validar os requisitos com o Product Owner e com o cliente.
- *Retrospectiva:* ao final de cada ciclo, para a verificar a qualidade interna do produto e da equipe;
- *Entrega:* finalização de cada parte do produto em releases do github.

3.3.1 ELICITAÇÃO DE REQUISITOS

Atividade	***Método***	***Ferramenta***	***Entrega***
Definição do Produto e do Projeto	*Reunião em grupo e estudo/análise individual*	*Google Meets*	*Documento de Visão do Produto e do Projeto no Pages*

3.3.2 ANÁLISE DE REQUISITOS

Atividade	***Método***	***Ferramenta***	***Entrega***

3.3.3 DOCUMENTAÇÃO DE REQUISITOS

Atividade	***Método***	***Ferramenta***	***Entrega***

3.3.4 VERIFICAÇÃO E VALIDAÇÃO DE REQUISITOS

Atividade	*Método*	*Ferramenta*	*Entrega*

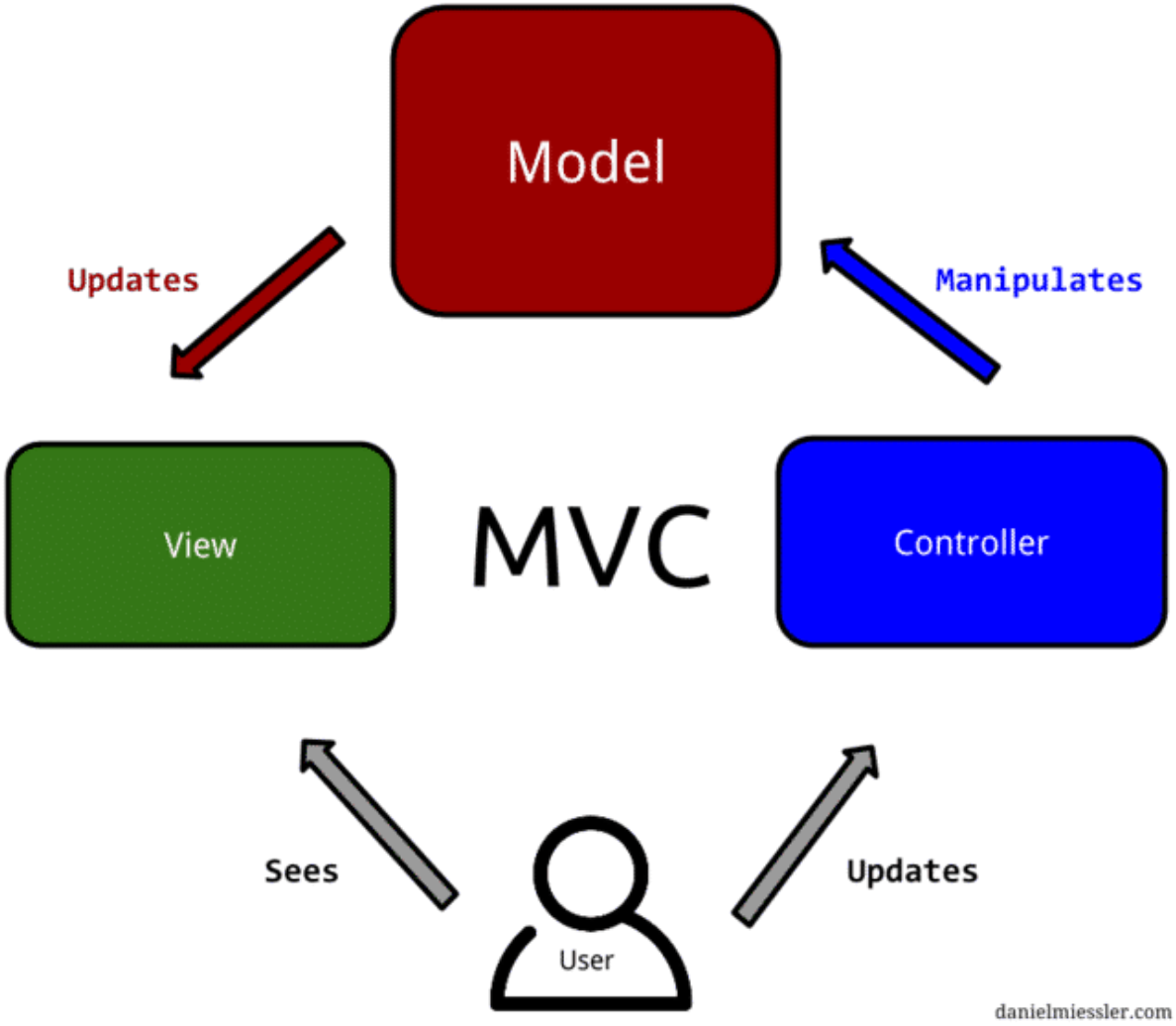
3.3.5 GERENCIAMENTO DE REQUISITOS

Atividade	*Método*	*Ferramenta*	*Entrega*

3.4 Arquitetura do Projeto

Com base na escolha de frameworks e foca na visualização e organização do projeto, definiu-se a escolha da aplicação MVC (Model-View-Controller), que é um padrão de arquitetura dividido logicamente em três partes:

1. Model: Gerenciamento e controle de dados por meio de funções lógicas e regras de negócios;
2. View: Apresentação de informações de forma visual ao usuário;
3. Controller: Intermédio entre as requisições feitas pela camada View e as respostas oferecidas pela camada Model;


Model
Updates
Manipulates
View
MVC
Controller
Sees
User
Updates
danielmiessler.com

4 LIÇÕES APRENDIDAS

Unidade 1

Fundamentos de Engenharia de Software

Disciplinas de Engenharia de Software

Metodologias de Desenvolvimento de Software

Ciclos de Vida

Atividades da Engenharia de Requisitos:

- *Elicitação*
- *Documentação*
- *Validação e Verificação*
- *Análise*
- *Gerência*

5 A MELHORAR

Unidade 1

Demora para solucionar dificuldades

Organização para trabalhar em grupo

6 REFERÊNCIAS BIBLIOGRÁFICAS

1. *https://www.atlassian.com/br/agile/scrum/roles#:~: texto%20 Scrum%20 tem%20 tr%C3%AAs%20 pap%C3%A9is,membros%20da%20 equipe%20de%20desenvolvimento.*
2. *https://www.lewagon.com/pt-BR/blog/o-que-e-padrao-mvc*
3. *Sommerville, I. and Sawyer, P. (1997) Requirements Engineering: A Good Practice Guide. John Wiley & Sons, Inc., Hoboken*
4. *PRESSMAN, R S. Engenharia de Software. 6.ed. São Paulo: Mc Graw Hill Internacional, 2006*