

**Plataforma de Gerenciamento e Apresentação de Principais Ferramentas de Engenharia de Software**

**Aluno:** Alice Gomes Nascimento

**RA:** 1272328052

**Aluno:** Anselmo Raimundo dos S C Nascimento

**RA:** 12723116545

**Aluno:** Carlos Henrique Santos Alves

**RA:** 1272328055

**Aluno:** Celso Magalhães Santana

**RA:** 12723112786

**Aluno:** Dinaldo Ribeiro de Carvalho

**RA:** 1272329374

**Aluno:** Douglas Eloi Conceição Lima

**RA:** 1272310473

Jonathan Cortppassi

**RA:** 12723213891

**Aluno:** Lucas Batista de Oliveira Araujo

**RA:** 1272327463

**Aluno:** Pedro Sousa Alves de Jesus

**RA:** 12723112339

**Aluno:** Rogério Carmo Rodrigues

**RA:** 1272316474

**Aluno:** Samuel Pereira de Souza

**RA:** 132723112947

Salvador – Bahia

2024

**Folha de Aprovação**

Aprovamos o documento aqui apresentado para o desenvolvimento do projeto Você Vereador - Sistema de Gestão de Demanda dos Bairros da Prefeitura de Salvador.

|  |  |  |
| --- | --- | --- |
| Prof. Felipe | Product Owner |  |
| Dinaldo Ribeiro | Scrum Master |  |
| Pedro Souza | Scrum Team |  |
| Douglas Eloi | Scrum Team |  |
| Alice Nascimento | Scrum Team |  |
| Anselmo Raimundo | Scrum Team |  |
| Rogério Carmo | Scrum Team |  |
| Jonathan Cortopassi | Developer |  |
| Samuel Pereira | Developer |  |
| Lucas Batista | Developer |  |
| Carlos Henrique Alves | Developer |  |
| Celso Santana | Developer |  |

**Revisões Anteriores**

|  |  |  |
| --- | --- | --- |
| **Revisor** | **Descrição** | **Data** |
| Equipe | Reunião para definição do Elaboração | 03/04/2024 |
| Equipe | Elaboração deste documento | 04/04/2024 |
| Equipe | 1ª Atualização deste Documento | 08/04/2024 |
| Equipe | Draft das interfaces | 15/04/2024 |
| Equipe | 2ª Atualização para Segunda entrega | 22/0/2024 |
|  |  |  |

**Requisitos Funcionais**

**1 - Gerenciamento de Ferramentas**

* Oferecer um catálogo abrangente de ferramentas para engenharia de software.
* Incluir modelagem de processos, análise de requisitos, testes de software, entre outros.
* O sistema permite buscar e filtrar por categoria, funcionalidade e critérios relevantes.
* Disponibilizar avaliações e comentários de usuários para insights na escolha.

**2 - Manutenção e Segurança**

* A plataforma é continuamente atualizada para garantir relevância e eficácia.
* Recursos de segurança robustos incluem criptografia de dados e proteção contra ataques.
* Medidas como injeção SQL e XSS são consideradas para segurança eficaz.

**3 - Ambiente Colaborativo**

* Espaço dedicado ao compartilhamento de conhecimentos e boas práticas.
* Fóruns de discussão e grupos específicos para cada ferramenta promover colaboração.
* Suporte à colaboração em tempo real para impulsionar produtividade e eficiência.

**4 - Documentação Automatizada**

* Capacidade de gerar documentação automaticamente a partir do código fonte.
* Incluir geração detalhada de documentação de API.
* Comentários de código serão extraídos e formatados para insights sobre lógica e estrutura.

**5 - Internacionalização e Localização**

* Suporte à adaptação para diferentes idiomas e regiões.
* Tradução eficiente de interfaces de usuário e outros elementos de texto.

**6 - Painel de Controle Personalizado**

* Visualização abrangente do progresso dos projetos.
* Acompanhamento detalhado de tarefas e status em tempo real.
* Recursos como quadros de tarefas, versionamento de código e relatórios personalizáveis.

**7 - Suporte Offline**

* O sistema deve oferecer a opção do usuário trabalhar efetivamente sem conexão com a internet.
* Funcionalidade de sincronização inteligente para acesso e edição de dados offline.

**8 - Favoritas**

* Favoritar as ferramentas mais utilizadas para acesso rápido.
* Funcionalidade de "favoritos" para marcar ferramentas mais úteis.

**9 - Suporte a Plugins**

* Suporte nativo para adição de plugins.
* Arquitetura modular para personalização da plataforma.

**10 - Notificações**

* Sistema de notificações para informar sobre atualizações e novidades.
* O usuário poderá escolher receber notificação proativas para se manter atualizados.

**Requisitos não funcionais**

**1 - Documentação**

* Elaboração de documentação técnica abrangente para desenvolvedores.
* Manuais do usuário detalhados.
* Guias de administração abrangentes.

**2 - Desempenho**

* Garantia de tempo de resposta ágil.
* Suporte a um elevado número de usuários simultâneos.
* Escalabilidade para acompanhar o crescimento da demanda.

**3 - Requisito de segurança**

* Implementação de sistemas robustos de autenticação e autorização.
* Emprego de criptografia de ponta a ponta para proteger dados.

**4 - Usabilidade**

* Interface amigável e intuitiva.
* Design centrado no usuário.
* Documentação clara e completa.

**5 - Portabilidade**

* Implantação em diferentes ambientes de hospedagem sem modificações significativas.

**6 - Tempo de resposta**

* Especificação de tempos de resposta esperados para diferentes operações.
* Definição clara de metas de desempenho.

**7 - Customização e configuração**

* Permitir personalização e configuração da plataforma pelos usuários.

**8 - Monitoramento**

* Implementação de um sistema robusto de monitoramento de desempenho e segurança.

**9 - Suporte multicanal**

* Oferecer suporte para múltiplos canais de acesso à plataforma.
* Integração com serviços de e-mail e mensagens instantâneas.

**10 - Manutenibilidade**

* Código modular e bem documentado.
* Divisão do código em componentes independentes e reutilizáveis

**Modelo de Processo de Software – Incremental**

O modelo incremental de desenvolvimento de software é uma abordagem que divide o processo de desenvolvimento em pequenas partes funcionais chamadas de incrementos. Cada incremento é desenvolvido, testado e entregue de forma independente, adicionando novas funcionalidades ao produto em cada ciclo. Isso permite que os usuários tenham acesso a partes do software em estágios iniciais, possibilitando feedback contínuo e ajustes ao longo do tempo. Essa metodologia oferece flexibilidade para lidar com mudanças nos requisitos e reduz gradualmente os riscos do projeto, ao mesmo tempo que permite a integração gradual de novas ferramentas e funcionalidades.

**1. Entregas Iterativas**

* A plataforma será implementada em partes distintas, permitindo aos usuários experimentarem e fornecerem feedback enquanto o desenvolvimento prossegue. Esse método possibilita o ajuste do software de acordo com as necessidades reais dos usuários.

**2. Flexibilidade para Mudanças**

* Em projetos de alta complexidade como este, é comum que os requisitos sofram alterações. O modelo incremental confere a capacidade de ajustar as funcionalidades ao longo do tempo, garantindo a liberdade para responder prontamente às demandas em evolução.

**3. Feedback Rápido dos Usuários**

* A cada entrega de uma parte da plataforma, os usuários terão a oportunidade de expressar suas preferências, apontar aspectos a serem melhorados e indicar suas necessidades. Esse retorno é fundamental para garantir que o desenvolvimento esteja alinhado com as expectativas dos usuários.

**4. Redução Gradual de Riscos**

* Com a entrega de incrementos, identificar e solucionar problemas ocorrerá de maneira gradual. Isso implica que a gestão dos riscos do projeto aconteça de forma eficiente ao longo do tempo, evitando a necessidade de enfrentá-los integralmente no final.

**5. Integração Gradual de Ferramentas**

* Considerando a complexidade das diversas ferramentas envolvidas, é essencial integrá-las cuidadosamente. O desenvolvimento incremental permite realizar essa integração de forma progressiva, garantindo a harmonia entre todas as ferramentas.

**6. Justificativa**

* Esses são os principais fundamentos que levaram à escolha do modelo incremental para o desenvolvimento da plataforma de gerenciamento de ferramentas de engenharia de software. Tal abordagem proporciona a flexibilidade e adaptabilidade necessárias para enfrentar os desafios e mudanças ao longo do projeto, ao mesmo tempo que garante a entrega de um produto que atenda às demandas dos usuários.

**Aplicação Detalhada da Metodologia Scrum ao Projeto de Plataforma de Engenharia de Software**

**1. Visão Geral do Projeto**

O projeto visa desenvolver uma plataforma que agregue as principais ferramentas de engenharia de software para auxiliar desenvolvedores em todo o ciclo de vida do desenvolvimento. A plataforma deve ser amigável, eficiente e personalizável, permitindo que os usuários encontrem as ferramentas que precisam de forma rápida e fácil.

**1.2 Definição da Metodologia Scrum**

Scrum é um framework leve que ajuda pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos.

**2. Detalhamento dos Papéis e Reuniões do Scrum**

**2.1 Scrum Team - Pedro Sousa, Douglas Eloi, Alice Nascimento, Anselmo Raimundo, Rogério Carmo**

A unidade fundamental do Scrum é um pequeno time de pessoas, um Scrum Team. O Scrum Team consiste em um Scrum Master, um Product Owner e Developers. Dentro de um Scrum Team, não há sub-times ou hierarquias. É uma unidade coesa de profissionais focados em um objetivo de cada vez, a Meta do Produto.

**2.2 Scrum Master - Dinaldo Ribeiro**

Responsável por estabelecer o Scrum conforme definido, e responsável pela eficácia do Scrum Team, são verdadeiros líderes que servem ao Scrum Team e à organização como um todo.

* **Responsabilidades**
  + Facilitar as reuniões Scrum e garantir o cumprimento dos princípios e valores;
  + Treinar os membros do time em autogerenciamento e cross-funcionalidade;
  + Monitorar o progresso do projeto e garantir a entrega das funcionalidades no prazo.

**2.3 Product Owner (PO) – Prof. Felipe**

Responsável por maximizar o valor do produto resultante do trabalho do Scrum Team.

* **Responsabilidades**
  + Definir a visão geral e os objetivos da plataforma;
  + Criar e comunicar claramente os itens do Product Backlog;
  + Colaborar com a equipe de desenvolvimento para garantir que a plataforma atenda às necessidades dos usuários;
  + Obter feedback dos stakeholders e incorporá-lo ao backlog do produto.

**2.4 Equipe de Developers – Jonathan Cortopassi, Samuel Pereira, Lucas Batista**

Developers são as pessoas do Scrum Team que estão comprometidas em criar qualquer aspecto de um Incremento utilizável a cada Sprint.

**Responsabilidades**

* + Desenvolver, testar e entregar as funcionalidades da plataforma de acordo com o backlog do produto;
  + Colaborar com o PO e o Scrum Master para garantir a qualidade da plataforma;
  + Seguir as melhores práticas de desenvolvimento de software.

**2.5 Reuniões Scrum e Sprints**

A Sprint é um contêiner para todos os outros eventos. Sprints são o coração do Scrum, onde ideias são transformadas em valor. São eventos de duração fixa de um mês ou menos para criar consistência. Uma nova Sprint começa imediatamente após a conclusão da Sprint anterior. Todo o trabalho necessário para atingir a meta do Produto, incluindo Sprint Planning, Daily Scrums, Sprint Review e Sprint Retrospective, acontece dentro de Sprints.

* **Sprint Planning**

A Sprint Planning inicia a Sprint ao definir o trabalho a ser realizado na Sprint. Este plano resultante é criado pelo trabalho colaborativo de todo o Scrum Team.

**Processo**

1. Product Owner (Prof. Felipe) apresenta as funcionalidades priorizadas do backlog do produto.
2. Equipe de Desenvolvimento (Jonathan, Samuel e Lucas) estima o tempo necessário para cada funcionalidade.
3. Scrum Master (Prof. Felipe) facilita a seleção das funcionalidades para o próximo sprint.
   * Duração: 2-4 horas;
   * Participantes: PO, Scrum Master, Equipe de Desenvolvimento;

* **Daily Scrum**

O propósito da Daily Scrum é inspecionar o progresso em direção a Meta da Sprint e adaptar o Sprint Backlog conforme necessário, ajustando o próximo trabalho planejado.

**Processo**

1. Reuniões realizadas diariamente com cada membro da equipe compartilhando o que realizou desde o último Daily Scrum, o que planeja fazer até o próximo e se enfrentou ou prevê enfrentar algum impedimento.
2. Impedimentos são identificados e discutidos pela equipe.
3. Qualquer ajuste necessário ao Sprint Backlog é discutido brevemente.
   * Duração: 15 minutos;
   * Participantes: Equipe de Desenvolvimento;

* **Sprint Review**
* O propósito da Sprint Review é inspecionar o resultado da Sprint e determinar as adaptações futuras.

**Processo**

1. A equipe demonstra as funcionalidades desenvolvidas durante a Sprint.
2. O Product Owner (Prof. Felipe) fornece o feedback sobre as funcionalidades apresentadas.
3. Todos discutem o que foi bem, o que pode ser melhorado e se alguma adaptação é necessária ao Product Backlog.
4. O Product Owner (Prof. Felipe) revisa e prioriza o Product Backlog com base no feedback recebido e nas novas informações obtidas durante a Sprint Review.
   * Duração: 1-2 horas;
   * Participantes: PO, Scrum Master, Equipe de Desenvolvimento.

* **Sprint Retrospective**
* O propósito da Sprint Retrospective é planejar maneiras de aumentar a qualidade e a eficácia.

**Processo**

1. O Scrum Master (Dinaldo Ribeiro) inicia a retrospectiva reforçando o propósito e as regras básicas.
2. A equipe discute o que funcionou bem durante a Sprint e o que poderia ser melhorado.
3. A equipe cria um plano de ação para implementar as melhorias na próxima Sprint.
   * Duração: 1-2 horas;
   * Participantes: Equipe de Desenvolvimento;

**3. Artefatos de Scrum**

**3.1 Product Backlog**

O Product Backlog é uma lista ordenada e emergente do que é necessário para melhorar o produto. É a única fonte de trabalho realizado pelo Scrum Team.

* **Compromisso**

Meta do Produto: A Meta do Produto descreve um estado futuro do produto que pode servir como um alvo para o Scrum Team planejar. A Meta do produto está no Product Backlog. A Meta do Produto é o objetivo de longo prazo para o Scrum Team. Eles devem cumprir (ou abandonar) um objetivo antes de assumir o próximo.

**3.2 Sprint Backlog**

O Sprint Backlog é um plano feito por e para os Developers. É uma imagem altamente visível, em tempo real do trabalho que os Developers planejam realizar durante a Sprint para atingir a Meta da Sprint.

* **Compromisso**

Meta da Sprint**:** A Meta da Sprint é o único objetivo da Sprint. Embora a Meta da Sprint seja um compromisso dos Developers, esta fornece flexibilidade em termos do trabalho exato necessário para alcançá-la. A Meta da Sprint é criada durante o evento Sprint Planning e então adicionada ao Sprint Backlog. Conforme os Developers trabalham durante a Sprint, eles mantêm a Meta da Sprint em mente.

**4. Mapeamento das Ferramentas de Engenharia de Software para as Funcionalidades da Plataforma**

* **Gerenciamento de Requisitos**
  + Ferramenta: Jira, Trello.
  + Funcionalidade: Criar, priorizar e acompanhar requisitos.
* **Análise e Especificação de Requisitos**
  + Ferramenta: Visual Paradigm, Lucidchart.
  + Funcionalidade: Modelar e documentar requisitos.
* **Validação de Requisitos**
  + Ferramenta: Protractor, Selenium.
  + Funcionalidade: Testar e validar os requisitos.
* **Gestão de Modelos de Processos de Software**
  + Ferramenta: Bizagi Modeler, ARIS.
  + Funcionalidade: Criar e Organizar.

**Documento de Definição do Ambiente de Desenvolvimento**

**Controle de Versão**

* **Sistema de Controle de Versão:** Git
* **Repositório Principal:** GitHub
* **Estratégia de Branch:** Git Flow
* **Acesso ao Repositório:** A equipe de desenvolvimento terá acesso de escrita/leitura ao repositório.

**IDE (Ambiente de Desenvolvimento Integrado**

* **IDE Utilizada:** Visual Studio Code
* **Extensões:** As extensões serão escolhidas de acordo com as demandas do projeto.
* **Configuração Inicial:** A configuração inicial seguirá o escopo das ferramentas que serão utilizadas.

**Ferramentas de Engenharia de Software**

* **Gerenciamento de Projetos:** Jira
* **Gestão de Tarefas:** Através das features do Jira.
* **Automação de Testes:** Jest
* **Integração Contínua:** GitLab CI

**Ambiente de Desenvolvimento**

* **Sistema Operacional:** Windows
* **Configuração Inicial do Ambiente:** De acordo com o escopo do projeto, a configuração inicial do ambiente será definida.
* **Virtualização/Containerização:** Não será utilizada.

**Documentação e Colaboração**

* **Plataforma de Documentação:** Google Docs
* **Colaboração:** Discord
* **Diretrizes de Documentação:** Não foram especificadas.