UNIVERSIDADE SALVADOR - UNIFACS
MODELOS, MÉTODOS E TÉCNICAS DA ENGENHARIA DE SOFTWARE

Eraldo de Oliveira Silva Neto - 12722210973
Rilton Bispo dos Santos - 1272218589
Lucas Davi Barros dos Santos - 1272218717
João Vitor Campos Gottschalk- 12722130690
Matheus Silva de Oliveira - 1272227031
Marissa de Paula Oliveira Nascimento - 12722213256
Rafael José de Jesus Santana - 12722213062

Trabalho prático A3

SALVADOR - BA
2024

Escopo Inicial

Configuração do ambiente de desenvolvimento

Necessário ter um computador com um o node instalado na versão 20 ou superior, necessário ter o git para controle de versão, uma IDE qualquer, por exemplo (Visual Studio Code, IntelliJ, Neovim…), ferramenta de engenharia de software Trello.

Definição de requisitos funcionais e não funcionais para a plataforma.

Requisitos Funcionais:

Gerenciamento de Requisitos:

Requisitos Funcionais:

Gerenciamento de Requisitos:

- **Elicitação:**
  - Coletar requisitos de diversas fontes (entrevistas, questionários, etc.).
  - Armazenar e organizar os requisitos em um repositório centralizado.
  - Permitir a visualização e edição dos requisitos.
- **Priorização:**
  - Definir a importância e a urgência dos requisitos.
  - Visualizar a priorização dos requisitos em diferentes formatos.
- **Análise de Impacto:**
  - Avaliar o impacto de alterações nos requisitos no processo de desenvolvimento.
  - Rastrear as dependências entre os requisitos.

Modelagem de Processos:

- **Criação de Diagramas:**
  - Suporte para notações de modelagem populares (BPMN, UML, etc.).
  - Ferramentas intuitivas para criação de diagramas de processos.
- **Elementos de Modelagem:**
  - Ampla variedade de elementos para representar atividades, decisões, fluxos de dados, etc.
- **Simulação de Processos:**
  - Simular o comportamento do processo antes da implementação.
  - Identificar gargalos e pontos de melhoria no processo.

Automação de Tarefas:

- **Integração com Ferramentas:**

    - Conexão com ferramentas de desenvolvimento (IDEs, ferramentas de integração contínua, etc.).
    - Automação de tarefas repetitivas no processo de desenvolvimento.
- **Geração de Código:**
    - Geração automática de código a partir dos modelos de processo.
    - Suporte para diferentes linguagens de programação.
- **Execução de Processos:**
    - Ambiente para execução dos processos modelados na plataforma.
    - Monitoramento do andamento dos processos e acompanhamento das métricas.

Colaboração em Equipe:

- **Controle de Versões:**
    - Rastreamento de alterações nos modelos e requisitos.
    - Possibilidade de reverter para versões anteriores.
- **Gerenciamento de Tarefas:**
    - Atribuição de tarefas aos membros da equipe.
    - Acompanhamento do progresso das tarefas e comunicação entre os membros da equipe.
- **Compartilhamento de Modelos:**
    - Compartilhamento de modelos com outros membros da equipe ou stakeholders.
    - Controle de acesso aos modelos.

Requisitos Não Funcionais:

Desempenho:

- A plataforma deve ser capaz de lidar com grandes volumes de dados e usuários simultâneos.
- O tempo de resposta da plataforma deve ser rápido e eficiente.

Segurança:

- A plataforma deve ter mecanismos para garantir a segurança dos dados e dos modelos de processo.
- O acesso à plataforma deve ser controlado por meio de autenticação e autorização.

Usabilidade:

- A plataforma deve ser fácil de usar e entender, mesmo para usuários não técnicos.
- A interface da plataforma deve ser intuitiva e amigável.

Escalabilidade:

- A plataforma deve ser capaz de crescer e se adaptar às necessidades da organização.

Suporte:

- O fornecedor da plataforma deve oferecer suporte técnico de qualidade aos usuários.

Exemplos Adicionais:

Funcional:

- **Importação e Exportação de Modelos:**
  - Permitir a importação e exportação de modelos em diferentes formatos (BPMN, XML, etc.).
- **Análise de Qualidade:**
  - Ferramentas para verificar a qualidade dos modelos de processo e requisitos.
  - Identificação de possíveis falhas e inconsistências.

Não Funcional:

- **Acessibilidade:**
  - A plataforma deve ser acessível a usuários com deficiência.
- **Internacionalização:**
  - A plataforma deve ser traduzida para diferentes idiomas.
- **Monitoramento e Logging:**
  - A plataforma deve ter mecanismos para monitorar seu desempenho e registrar eventos importantes.

Modelo de processo de software que utilizaremos no nosso projeto

Modelo Cascata:

- **Processo bem definido:** O modelo Cascata oferece um processo estruturado e bem definido, com etapas claras e documentação detalhada. Isso pode ser útil para projetos com requisitos bem conhecidos e prazos e orçamentos fixos.
- **Fácil de gerenciar:** A natureza linear do modelo Cascata facilita o gerenciamento do projeto, pois cada etapa é concluída antes de passar para a próxima. Isso pode ser útil para equipes com experiência em métodos tradicionais de desenvolvimento de software.
- **Facilidade de acompanhamento:** O modelo Cascata facilita o acompanhamento do progresso do projeto, pois cada etapa tem um marco bem definido. Isso pode ser útil para stakeholders que desejam acompanhar o desenvolvimento do projeto.

Desvantagens do Modelo Cascata:

- **Falta de flexibilidade:** O modelo Cascata é um modelo rígido e inflexível, o que pode dificultar a adaptação a mudanças nos requisitos ou no ambiente do projeto. Isso pode

ser um problema para projetos com requisitos mal definidos ou que estão sujeitos a mudanças frequentes.

- **Dificuldade em voltar atrás:** Uma vez que uma etapa é concluída no modelo Cascata, é difícil voltar atrás e fazer alterações. Isso pode ser um problema se forem encontrados erros ou problemas em etapas anteriores do processo.
- **Tempo e custo:** O modelo Cascata pode ser um processo demorado e caro, especialmente para projetos grandes e complexos. Isso pode ser um problema para projetos com prazos apertados ou orçamentos limitados.

Metodologia ágil definida.

Scrum

1. Objetivos do projeto:

- **Objetivo Primário:** Primeiro objetivo aceito em grupo, será separar funções para cada componente, estabelecendo prazos e encontros diários revisando e facilitando a criação do desenvolvimento do software.

2. Backlog do produto:

Funcionalidades Essenciais:

- **Autenticação:**

1. Login com e-mail e senha
2. Login com redes sociais
3. Esqueci minha senha
4. Criação de conta

- **Perfil do Usuário:**

1. Editar perfil
2. Visualizar histórico de atividades
3. Gerenciar configurações

- **Funcionalidade principal do app:**

    1. Ajudar o desenvolvedor na hora de criar um processo de software.
    2. Incluir todos os recursos e opções disponíveis para auxiliar o desenvolvedor em um trabalho de grupo.
    3. Considerar diferentes tipos de usuários e suas necessidades

Funcionalidades Adicionais:

- **Notificações:**
    - Notificações push para eventos importantes
    - Configurações de notificações
- **Mensagens:**
    - Sistema de mensagens privadas entre usuários
    - Envio de fotos, vídeos e outros arquivos
- **Integração com redes sociais:**
    - Compartilhamento de conteúdo nas redes sociais
    - Conexão com amigos e familiares
- **Análise de dados:**
    - Rastreamento de uso do app
    - Coleta de dados de desempenho
    - Geração de relatórios

Melhorias e Correções:

- **Correção de bugs:**
    - Descrever os bugs encontrados
    - Priorizar a correção dos bugs mais críticos
- **Melhorias de desempenho:**
    - Otimizar o código para melhorar a velocidade do app
    - Reduzir o consumo de bateria
- **Melhorias na interface do usuário:**
    - Tornar a interface mais intuitiva e fácil de usar
    - Implementar um design mais atraente e moderno

Exemplos de Histórias de Usuário:

- Como usuário, quero logar no app com meu e-mail e senha.
- Como usuário, quero editar meu perfil e atualizar minhas informações.
- Como usuário, quero receber notificações push quando receber uma nova mensagem.
- Como usuário, quero enviar mensagens privadas para outros usuários.
- Como usuário, quero compartilhar conteúdo do app nas redes sociais.

Priorização do Backlog:

- Priorize as funcionalidades de acordo com o valor que elas agregam ao app e o impacto que elas têm nos usuários.
- Utilize técnicas de priorização como MoSCoW (Must have, Should have, Could have, Won't have) para definir a ordem de desenvolvimento das funcionalidades.

3. Sprint:

- O grupo irá se encontrar duas vezes na semana com etapas concluídas visando entregar um conjunto específico de funcionalidades. No início de cada sprint, a equipe realizará reuniões de planejamento para selecionar as funcionalidades do backlog durante discussões de projeto implementando para estimar o tempo necessário para cada uma.

4. Reuniões diárias:

- Ao decorrer do projeto e chegando ao tempo estipulado em grupo, iremos marcar encontros diários visando as etapas finais e entrega do projeto.

5. Sprint final:

- Nesta etapa mostraremos as atividades divididas com conclusões para analisarmos possíveis erros e ajudar a finalizar a etapa.