Itens de um Plano de Projeto

Histórico da Revisão

Data	Versão	Descrição	Autor(es)
11/04/2025	1.0	Criação do documento inicial do plano de projeto.	Felipe da Costa Coqueiro

1. Identificação

a. Nome do Projeto: RCEI - Repositório Cientifico e Educacional Integrado
b. Equipe: Felipe da Costa Coqueiro e Gustavo Poffo
c. Data de criação do documento: 11/04/2025

2. Introdução

Este documento descreve o plano de projeto para o desenvolvimento de um produto que visa aprimorar a experiência do usuário (UX) e a gestão de informações na plataforma ORCID. O público-alvo deste plano de projeto são os professores, pesquisadores e alunos interessados na área acadêmica.

O objetivo principal do projeto é criar uma aplicação que, utilizando a API pública do ORCID, ofereça uma interface mais intuitiva e funcionalidades adicionais para pesquisadores e usuários comuns interessados em informações acadêmicas.

O escopo do projeto é dinâmico e evoluirá ao longo do ciclo de vida do projeto, refletindo as mudanças e ajustes necessários para garantir o sucesso. Será utilizado um método de desenvolvimento ágil Scrum para permitir flexibilidade e adaptação às mudanças nos requisitos e prioridades.

3. Escopo do projeto

O projeto consiste no desenvolvimento de uma aplicação [Web/Mobile/Desktop - escolher a opção adequada] que se integra com a API do ORCID para fornecer as seguintes funcionalidades:

(i) Para Pesquisadores:

- Visualização de estatísticas de publicações.
- Gerenciamento visual e interativo de publicações.
- Análise da rede de colaboração acadêmica.
- Alertas e notificações (ex: novas citações, publicações).

- Gerenciamento de projetos de pesquisa (em andamento e finalizados).
- Visualização do histórico acadêmico.

(ii) Para Usuários Comuns:

- Busca de pesquisadores, artigos, projetos de pesquisa e áreas de pesquisa.
- Visualização de perfis de pesquisadores.
- Visualização de detalhes de projetos de pesquisa.
- Sistema de avaliação/comentários.

4. Equipe e infraestrutura

(i) Equipe:

- Gerente de Projeto: Felipe da Costa Coqueiro
- Desenvolvedor(es) Front-End: Gustavo Poffo
- Desenvolvedor(es) Back-End: Felipe da Costa Coqueiro
- Designer UX/UI: Gustavo Poffo
- Testador(es): Felipe da Costa Coqueiro e Gustavo Poffo

(ii) Infraestrutura:

Ferramentas de Desenvolvimento: VStudio Code
Controle de Versão: GitHub
Gerenciamento de Projetos: Trello
Comunicação: Whatsapp
Softwares de Apoio: Figma
Equipamentos: Computadores, acesso à internet.

5. Acompanhamento do projeto

(i) Reuniões:

Daily: Reuniões curtas (15 minutos) para alinhamento da equipe e identificação de impedimentos.

Planning: No início de cada sprint, planejar as atividades a serem realizadas.

Review: Ao final de cada sprint, apresentar o trabalho realizado e obter feedback.

Retrospective: Ao final de cada sprint, identificar oportunidades de melhoria no processo.

- Acompanhamento Assíncrono: Utilização de ferramentas de gerenciamento de projetos para acompanhar o progresso das tarefas.
- Comunicação com o Cliente: Reuniões periódicas (quinzenais/mensais) com o cliente para apresentar o progresso do projeto e obter feedback. Relatórios de progresso semanais.

6. Cronograma e Marcos do projeto

Atividade	Início	Término	Duração	Entregas (Artefatos)
1. Planejamento e Design	13/04	27/04	15 dias	Documento de Requisitos (ER) finalizado, Protótipo UX/UI
2. Desenvolvimento Front-End	27/04	04/05	15 dias	Interface do usuário funcional
3. Desenvolvimento Back-End	04/05	23/05	20 dias	API integrada, Banco de dados configurado
4. Testes e Correções	23/05	06/06	15 dias	Relatório de testes, Versão corrigida do produto
5. Implantação e Entrega	06/06]	20/06	15 dias	Produto final implantado e documentado

(i) Marcos Importantes (Milestones):

MVP (Minimum Viable Product): 23/05/2025

Release 1.0: 20/06/2025

7. **Gerência de Riscos**

Risco	Probabilidade	Impacto	Mitigação
Atualização da API do ORCID	Médio	Alto	Monitoramento constante da API, implementação de versionamento, testes automatizados.
Problemas com sistemas de IA open source	Baixo	Alto	Avaliação cuidadosa dos sistemas, implementação de medidas de segurança, backups regulares, consideração de sistemas privados.
Falta de comunicação entre a equipe	Médio	Médio	Reuniões diárias, utilização de ferramentas de comunicação eficientes.
Falhas de segurança	Baixo	Alto	Implementação de medidas de segurança robustas, testes de segurança periódicos.
Mudanças nos requisitos durante o projeto	Médio	Médio	Utilização de metodologia ágil para permitir adaptação às mudanças, comunicação constante com o cliente.

8. **Testes do produto**

(i) Tipos de Teste:

- Testes Unitários: Testar componentes individuais do código.
- Testes de Integração: Testar a interação entre diferentes componentes.
- Testes de Sistema: Testar o sistema como um todo.
- Testes de Aceitação: Testes realizados pelo cliente para verificar se o produto atende aos requisitos.
- Testes de Usabilidade: Avaliar a facilidade de uso da interface.
- Testes de Segurança: Avaliar a segurança do sistema (vulnerabilidades).

(ii) Estratégias:

- Testes automatizados para garantir a qualidade do código.
- Testes manuais para avaliar a usabilidade e a experiência do usuário.
- Participação do cliente nos testes de aceitação.

(iii) Requisitos de Aceitação:

- O produto deve atender a todos os requisitos funcionais e não funcionais descritos no Documento de Requisitos (ER).
- O produto deve passar em todos os testes de segurança.
- O cliente deve aprovar os testes de aceitação.

9. Gerenciamento de configuração de software

- *Padronização da Nomenclatura*: Definição de um padrão para nomear arquivos, branches e commits.
- *Versionamento*: Utilização do GitHub para controle de versão do código.
- *Git Flow*: Adoção do Git Flow como modelo de ramificação para gerenciar o ciclo de vida do software (branches develop, release, hotfix).
- *Gerenciamento de Dependências*: Utilização de ferramentas de gerenciamento de dependências (npm, pip) para controlar as dependências do projeto, criação de requeriments.txt.

10. Plano de Manutenção de software

- *Identificação*: O usuário identifica um problema ou sugere uma melhoria.
- *Registro*: O problema/sugestão é registrado em uma ferramenta de gerenciamento de tarefas.
- *Análise*: A equipe de desenvolvimento analisa o problema/sugestão para determinar sua prioridade e complexidade.
- *Planejamento*: O problema/sugestão é incluído em um sprint.
- *Implementação*: A equipe de desenvolvimento implementa a correção/melhoria.
- *Teste*: A correção/melhoria é testada pela equipe de teste.
- *Implantação*: A correção/melhoria é implantada no ambiente de produção.
- *Monitoramento*: A equipe de desenvolvimento monitora o sistema para garantir que a correção/melhoria tenha sido implementada com sucesso.

Referências

SOMMERVILLE, I. Engenharia de Software. 9. ed. São Paulo: Pearson Prentice Hall, 2011. (Manual geral de engenharia de software, útil para contexto).

KNIBERG, M.; SKARIN, M. Kanban and Scrum: making the most of both. InfoQ, 2009. Disponível em: https://www.infoq.com/articles/kanban-scrum/. Acesso em: [Data de Acesso]. (Se utilizar Kanban ou uma combinação Kanban-Scrum).