**Plano de Projeto - CarrerFlow**

**1. Identificação do Projeto**

**Nome do Projeto:** CarrerFlow

**Versão**: 1.1

**Categoria**: Aplicação web de produtividade pessoal

**Equipe: Evellyn, Italo Alves, Juarez, Victor, Eleonora, Carlos Eduardo, Anderson e Erick**

**Data de início:** Setembro 2025

**Prazo Final:** Novembro 2025

**1.1 Stakeholders**

**- Product Owner: Evellyn**

**- Scrum Master: Italo**

**- Desenvolvedores:Carlos Eduardo, Eleonora, Juarez, Stefany e Victor Vahid**

**- Quality Assurance (QA):Erick Vitor**

**- DBA: Anderson Leite**

**- Usuários:** Profissionais em busca de emprego, recém-formados, pessoas em transição de carreira

### **1.2. Persona do Público-Alvo**

Para garantir que o desenvolvimento do **CarrerFlow** seja guiado por uma compreensão clara e empática das necessidades do usuário final, o nosso projeto adotará a seguinte persona. A documentação e as decisões de design devem sempre considerar o público para o qual o produto se destina. A definição desta persona servirá como um filtro para todas as decisões de funcionalidades, assegurando que a equipe construa um produto que resolva problemas reais.

**Perfil da Persona: "O Graduado Desempregado"**

* **Nome:** Lorenzo Pereira da Silva
* **Idade:** 22 anos
* **Ocupação:** Recém-graduado em Engenharia de Software, em busca ativa de seu primeiro emprego em tempo integral na área de desenvolvimento de software.

**Comportamentos e Ferramentas Atuais:**

* **Proficiência Técnica:** Confortável com tecnologia web, mas prefere ferramentas que funcionem "sempre" e não exijam configuração complexa.
* **Processo Atual:** Usa o LinkedIn para 80% de sua busca por vagas. Ele salva vagas no próprio LinkedIn, mas para rastrear o que foi enviado, ele usa uma "planilha desorganizada" (provavelmente Google Sheets ou Excel).
* **Gerenciamento de Notas:** Tenta anotar detalhes de triagens e nomes de recrutadores em um bloco de notas digital (ex: Google Keep ou Notion), mas essas notas ficam desconectadas da sua planilha, causando confusão

**Objetivos:** Garantir uma posição como desenvolvedor de software júnior em até três meses. Manter-se organizado e sentir que tem o controle de seu processo de busca por emprego. Aprender com cada candidatura e entrevista para aprimorar sua estratégia.

**Frustrações:** Gerenciar mais de 20 candidaturas ativas em uma planilha desorganizada. Esquecer detalhes importantes de uma triagem por e-mail realizada na semana anterior. Não saber quais fontes de vagas (ex: LinkedIn, Grupos de Vagas, indicação) são mais eficazes. Sentir-se sobrecarregado e reativo, em vez de proativo e estratégico.

**2. Objetivo do Projeto**

**Problema:** Profissionais em busca de novas oportunidades de carreira carecem de uma ferramenta dedicada e de fácil utilização para rastrear suas candidaturas de forma eficaz. Essa lacuna leva à perda de oportunidades, dificuldade de preparação para entrevistas (devido à falta de anotações consolidadas) e à incapacidade de aprender com o histórico de candidaturas (por exemplo, identificar quais tipos de vagas ou fontes geram mais entrevistas). O **CarrerFlow** busca ser uma plataforma de resolução para esse problema ao oferecer um CRM (Customer Relationship Management) pessoal, projetado especificamente para o ciclo de vida da busca por emprego.

**2.1 Objetivo Geral**

Desenvolver uma plataforma web intuitiva que permita aos usuários cadastrar, visualizar e gerenciar todas as suas candidaturas de emprego em um único lugar. A plataforma fornecerá uma interface clara e organizada para acompanhar o status de cada aplicação, registrar anotações detalhadas sobre entrevistas e empresas, e configurar lembretes para atividades importantes, otimizando o processo de busca por emprego e aumentando as chances de sucesso do candidato.

**2.2 Objetivos Específicos**

**Centralização de Informações**

* Centralizar todas as candidaturas em um local único e organizado

**Organização e Acompanhamento**

* Implementar sistema de categorização por status de candidatura
* Desenvolver dashboard com visão geral do pipeline de candidaturas
* Criar histórico detalhado de interações por vaga

**Produtividade e Eficiência**

* Desenvolver sistema de lembretes e notificações
* Implementar funcionalidade de notas e observações por candidatura

**Análise e Insights**

* Implementar métricas de acompanhamento do processo

**3. Escopo do Projeto**

A clareza no escopo é crítica para gerenciar as expectativas das partes interessadas e prevenir o aumento descontrolado do escopo, um dos riscos mais comuns em projetos de software. O projeto do CarrerFlow preza por uma definição rigorosa dos requisitos, entendendo que esse é um pilar para o sucesso de qualquer projeto.

### **3.1 Inclusões: Requisitos Funcionais e Não Funcionais**

#### Este é um conjunto concreto de funcionalidades e características do CarrerFlow, formando a base para o Product Backlog inicial.

**Requisitos Funcionais (O que o sistema fará):**

**Gerenciamento de Usuário:** Registro seguro, login (com e-mail e senha) e logout do usuário.

**Dashboard:** Uma visão centralizada que exibe um resumo das candidaturas por status (ex: 5 Aplicadas, 2 em Entrevista, 1 Oferta).

**Gerenciamento de Candidaturas (CRUD - Create, Read, Update, Delete):**

* **Criar (Create):** Um formulário para adicionar uma nova candidatura com campos para Nome da Empresa, Título da Vaga, Data da Candidatura, Fonte (ex: LinkedIn, Indeed) e Status.
* **Ler (Read):** Uma visualização em lista ou quadro (Kanban) de todas as candidaturas, com capacidade de filtrar e ordenar por status e data.
* **Atualizar (Update):** Capacidade de editar todos os detalhes da candidatura, com destaque para a alteração de status.
* **Excluir (Delete):** Capacidade de remover um registro de candidatura.

**Visão Detalhada:** Uma página dedicada para cada candidatura, permitindo visualizar todos os detalhes e anotações associadas.

**Anotações:** Uma área de texto dentro da visão detalhada para adicionar anotações com data e hora sobre entrevistas, contatos ou outras informações relevantes.

**Lembretes/Notificações:** Uma funcionalidade simples para definir uma data futura para um lembrete de acompanhamento (follow-up) para uma candidatura específica.

**Requisitos Não Funcionais (Como o sistema irá se comportar):**

**Usabilidade:** A aplicação deve possuir uma interface de usuário limpa, intuitiva e responsiva, que funcione de maneira fluida nos navegadores web modernos (Chrome, Firefox, Safari) em dispositivos desktop.

**Desempenho:** Todas as páginas devem carregar em menos de 3 segundos. As consultas ao banco de dados devem ser executadas em menos de 500ms.

**Segurança:** As senhas dos usuários devem ser armazenadas utilizando técnicas de hashing e salting. Toda a comunicação entre o cliente e o servidor deve ocorrer via HTTPS. Devem ser implementadas proteções básicas contra vulnerabilidades web comuns (ex: XSS, SQL Injection).

**Manutenção:** O código deve ser bem documentado e seguir padrões consistentes para facilitar a colaboração e o desenvolvimento futuro.

### **3.2 Exclusões**

Deixar de forma clara o que não será feito é uma poderosa estratégia de mitigação de riscos contra o aumento descontrolado do escopo. A lista a seguir representa funcionalidades que foram consideradas no escopo do **CarrerFlow**, mas estrategicamente adiadas para garantir a entrega de um MVP de alta qualidade dentro do prazo.

**Lista de Exclusões:**

**Aplicação Mobile:** Este projeto será exclusivamente uma aplicação web. Nenhuma aplicação nativa para iOS ou Android será desenvolvida.

**Integração Direta com Portais de Emprego:** O sistema não irá importar automaticamente vagas do LinkedIn, Indeed, etc. Toda a entrada de dados será manual.

**Funcionalidades de IA/Machine Learning:** Não haverá análise de currículo, correspondência com descrições de vagas ou análises preditivas baseadas em inteligência artificial.

**Análises Avançadas:** O dashboard será simples. Não serão implementados gráficos complexos ou análises de tendências históricas.

**Funcionalidades Sociais:** Não haverá interação entre usuários, compartilhamento de candidaturas ou recursos de comunidade.

**Integração com Calendários de Terceiros:** Os lembretes serão internos à aplicação; não haverá integração com Google Calendar, Outlook, etc.

**Upload de Documentos:** Os usuários não poderão fazer upload de currículos, cartas de apresentação ou outros documentos.

**4. Cronograma**

Um cronograma bem definido ajuda a estabelecer um caminho e um calendário para um projeto. A tabela a seguir fornece uma visão geral do ciclo de vida completo do projeto do CarrerFlow, mapeando as fases principais para o cronograma Servindo como uma ferramenta de comunicação estratégica para a equipe, definindo expectativas claras sobre o foco do time em diferentes momentos.

| **Fase** | **Sprint(s)** | **Período Estimado** | **Principais Entregas** |
| --- | --- | --- | --- |
| **Fase 1: Planejamento e Setup** | Sprint 0 | 08/09 a 14/09 (1 semana) | Definição da arquitetura, setup dos ambientes de desenvolvimento, criação do Product Backlog detalhado, design inicial da UI/UX no Figma. |
| **Fase 2: MVP - Principais Features** | Sprints 1 e 2 | 15/09 a 12/10 (4 semanas) | Módulo de autenticação funcional, CRUD completo de candidaturas, dashboard principal com listagem de vagas. |
| **Fase 3: Features Adicionais** | Sprints 3 e 4 | 13/10 a 09/11 (4 semanas) | Implementação dos sistemas de notas e lembretes, refinamento da interface, início dos testes de responsividade. |
| **Fase 4: Estabilização e QA** | Sprint 5 | 10/11 a 23/11 (2 semanas) | Testes de ponta a ponta, correção intensiva de bugs, preparação da documentação final do projeto e do material de apresentação. |
| **Fase 5: Encerramento** | - | 24/11 a 30/11 (1 semana) | Ensaios da apresentação, deploy da versão final, entrega do projeto e apresentação final. |

**5. Metodologias**

### **5.1 Metodologia de Desenvolvimento: Agile com Scrum**

O Scrum é um framework ágil para gerenciar projetos complexos, que divide o trabalho em ciclos curtos chamados Sprints. Ele se baseia nos pilares de transparência, inspeção e adaptação para otimizar a previsibilidade e controlar os riscos.

**Papéis e Responsabilidades da Equipe:**

* **Product Owner (PO) (1 membro):** É a única voz do stakeholder, representando a persona "O Graduado Desempregado" e os objetivos acadêmicos do projeto. É responsável por definir e priorizar o Product Backlog para maximizar o valor entregue. Toma a decisão final sobre a implementação de funcionalidades.
* **Scrum Master (SM) (1 membro):** Atua como um líder-servidor e coach para a equipe Scrum. É responsável por garantir que o framework Scrum seja compreendido e seguido. Facilita os eventos do Scrum, remove impedimentos e protege a equipe de distrações externas.
* **Time de Desenvolvimento (Development Team) (6 membros):** Um grupo multifuncional e autônomo de profissionais que realiza o trabalho de entregar um Incremento de produto "Pronto" e potencialmente utilizável ao final de cada Sprint. Este time inclui todas as habilidades necessárias: desenvolvimento front-end, back-end, garantia de qualidade (QA) e administração de banco de dados (DBA). Os papéis de QA e DBA não são entidades separadas, mas sim especialidades integradas ao Time de Desenvolvimento. Esta estrutura promove a responsabilidade coletiva da qualidade e da integridade dos dados, evitando a criação de ruídos e gargalos.

**Eventos do Scrum (Cerimônias):**

* **Duração do Sprint:** 2 semanas. Este intervalo oferece um bom equilíbrio entre planejamento e feedback para um projeto de 12 semanas, permitindo um Sprint de configuração e quatro Sprints completos de desenvolvimento.
* **Planejamento do Sprint (Sprint Planning) (4 horas, no início do Sprint):** Toda a equipe Scrum colabora para definir uma Meta do Sprint e selecionar itens do Product Backlog para o Sprint Backlog.
* **Daily Scrum (15 minutos, diariamente):** Uma reunião rápida para o Time de Desenvolvimento sincronizar as atividades e criar um plano para as próximas 24 horas.
* **Revisão do Sprint (Sprint Review) (2 horas, no final do Sprint):** A equipe Scrum apresenta o trabalho concluído ("Incremento") às partes interessadas (ex: professor, outros grupos de alunos) para coletar feedback. O Product Backlog é atualizado com base nesse feedback.
* **Retrospectiva do Sprint (Sprint Retrospective) (1,5 horas, no final do Sprint):** A equipe Scrum inspeciona seu próprio processo e identifica melhorias para o próximo Sprint.

**Artefatos do Scrum:**

* **Product Backlog:** Uma lista única e ordenada de tudo o que é conhecido como necessário no produto. É gerenciado pelo Product Owner.
* **Sprint Backlog:** O conjunto de itens do Product Backlog selecionados para o Sprint, mais um plano para entregar o Incremento do produto. É gerenciado pelo Time de Desenvolvimento.
* **Incremento e Definição de "Pronto":** A soma de todos os itens do Product Backlog concluídos durante um Sprint. A definição de “Pronto” é um entendimento compartilhado do que significa o trabalho estar completo (ex: "o código foi revisado por pares, os testes unitários foram aprovados, a funcionalidade foi verificada pelo QA e o código foi integrado à branch principal").

### **5.2. Metodologia: Estratégia de Garantia de Qualidade e Testes**

O especialista em QA no Time de Desenvolvimento lidera esse esforço, mas a qualidade é uma responsabilidade de toda a equipe.

**Camadas de Teste:**

* **Testes Unitários:** Os desenvolvedores escrevem testes unitários para seu código (especialmente a lógica do back-end) para garantir que os componentes individuais funcionem como esperado.
* **Testes de Integração:** A equipe testará a interação entre diferentes partes da aplicação (ex: garantir que o front-end chame corretamente a API do back-end).
* **Testes de Ponta a Ponta (End-to-End - E2E):** O especialista em QA será responsável por criar e executar testes automatizados que simulam os fluxos de trabalho do usuário do início ao fim (ex: registrar, fazer login, adicionar uma candidatura e fazer logout).
* **Testes de Aceitação do Usuário (UAT):** Conforme delineado no roteiro, uma fase formal de UAT será conduzida na Semana 10 para coletar feedback de usuários fora da equipe principal.

### **5.3. Metodologia: Análise de Riscos e Plano de Mitigação**

O gerenciamento de riscos é uma área crítica em projetos de TI. Envolve a identificação de riscos potenciais, a avaliação de seu impacto e o desenvolvimento de planos de mitigação.Embora métodos ágeis como o Scrum lidem com riscos de forma implícita através de ciclos curtos e feedback constante, um registro de riscos formal ainda é uma prática recomendada..

Um registro de riscos eficaz deve ser adaptado ao contexto único de um projeto como o do CarrerFlow liderado por estudantes com prazo fixo. Os riscos mais significativos muitas vezes não são puramente técnicos, mas estão relacionados à dinâmica da equipe, lacunas de habilidades e ao próprio ambiente acadêmico. O quadro abaixo fornece uma maneira estruturada de identificar, analisar e planejar respostas a ameaças potenciais ao sucesso do projeto. Este é um documento vivo que deve ser revisado em cada retrospectiva de Sprint.

| **ID** | **Descrição do Risco** | **Categoria** | **Probabilidade (1-5)** | **Impacto (1-5)** | **Pontuação (P\*I)** | **Estratégia de Mitigação (Proativa)** | **Plano de Contingência (Reativo)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| R01 | **Aumento do Escopo (Scope Creep):** Membros da equipe adicionam funcionalidades não planejadas, comprometendo o cronograma do MVP. | Escopo | 4 | 5 | 20 | - O Product Owner tem autoridade final sobre o backlog. - Aderir estritamente à lista de "Exclusões". - Todas as novas ideias são adicionadas ao Product Backlog para consideração futura, não ao Sprint atual. | Durante a Revisão do Sprint, o PO irá remover funcionalidades de menor prioridade do próximo Sprint para acomodar qualquer trabalho crítico e imprevisto. |
| R02 | **Lacuna de Habilidade Técnica:** A equipe não possui expertise em uma área chave da pilha de tecnologia escolhida (ex: hooks do React, agregação no MongoDB). | Técnico / Recurso | 3 | 4 | 12 | - Incentivar a programação em par. - Alocar tempo para "spikes" (tarefas de pesquisa com tempo limitado) para investigar desafios técnicos. - Priorizar implementações mais simples primeiro. | Buscar orientação do supervisor/professor acadêmico. Reavaliar o escopo da funcionalidade afetada para encontrar uma solução técnica mais simples. |
| R03 | **Participação Desigual dos Membros da Equipe:** Alguns membros se tornam desengajados ou indisponíveis devido a outros compromissos acadêmicos. | Recurso / Equipe | 3 | 5 | 15 | - O Scrum Master monitorará o engajamento nas Daily Scrums. - Fomentar uma cultura de transparência e propriedade coletiva. - Tornar os compromissos do Sprint realistas e baseados na capacidade conhecida. | O Scrum Master facilitará uma discussão com o membro da equipe. Se não for resolvido, escalar para o supervisor acadêmico. A equipe replanejará o Sprint para absorver o trabalho. |
| R04 | **Problemas de Integração:** Componentes de front-end e back-end não funcionam juntos como esperado no final de um Sprint. | Técnico | 4 | 3 | 12 | - Definir contratos de API claros no início do Sprint. - Incentivar integrações frequentes e pequenas em vez de uma única "grande integração" no final. - Usar ferramentas como Postman para testes de API. | Alocar um "dia de integração" de emergência dentro do Sprint. A equipe se concentrará no problema para resolvê-lo. |
| R05 | **Testes Inadequados:** Bugs são descobertos tardiamente no projeto (durante o UAT ou a demonstração final), exigindo retrabalho significativo. | Qualidade | 3 | 4 | 12 | - Fazer cumprir a Definição de "Pronto", que inclui testes. - O especialista em QA irá defender as práticas de teste e criar testes E2E automatizados desde o início. - Realizar revisões de código por pares. | Dedicar a primeira metade do último Sprint (Sprint 4) para correção focada de bugs e estabilização antes da fase final de UAT. |

**6. Recursos Utilizados**

#### **6.1. Equipe**

A equipe do **CarrerFlow** é composta por 8 membros, tamanho ideal para uma única equipe Scrum, que recomenda de 3 a 9 membros no Time de Desenvolvimento, além do PO e do SM. A equipe trabalhará como uma unidade única e localizada (ou virtualmente localizada por meio de ferramentas de comunicação) para maximizar a eficiência da colaboração e da comunicação, um pilar do Scrum.

* **1x Product Owner:** Responsável por gerenciar o backlog do produto e garantir que o desenvolvimento agregue valor.
* **1x Scrum Master:** Responsável por facilitar as cerimônias Scrum e remover impedimentos.
* **3x Desenvolvedores (Full-stack/Frontend/Backend):** Responsáveis pela codificação e implementação das funcionalidades.
* **2x Quality Assurance (QA):** Responsável pela criação e execução de planos de teste.
* **1x Database Administrator (DBA):** Responsável pela modelagem, criação e manutenção do banco de dados.

#### **6.2. Stack de Tecnologias**

Uma stack tecnológica é a combinação de linguagens de programação, frameworks e bancos de dados usados para construir uma aplicação. A escolha deve ser baseada nos requisitos do projeto, na experiência da equipe e na escalabilidade.

| **Categoria** | **Tecnologia** | **Justificativa** |
| --- | --- | --- |
| **Frontend** | React.js ou Vue.js | Frameworks modernos, componentizáveis e com vasta documentação, ideais para criar interfaces ricas e intuitivas. |
| **Backend** | Node.js (com Express.js) ou  Python(FastAPI ou Flask) | Permite o uso de JavaScript em todo o stack, unificando a linguagem e facilitando a colaboração. É performático para aplicações web. |
| **Banco de Dados** | PostgreSQL, SQLite ou  MongoDB | Um banco de dados relacional robusto, de código aberto e confiável, adequado para a estrutura de dados do projeto. |
| **Hospedagem** | Vercel ou Netlify (Frontend), Heroku ou Render (Backend), Supabase (BD) | Opções com planos gratuitos generosos, ideais para projetos acadêmicos, com processos de deploy simplificados. |

#### **6.3. Ferramentas**

Projetos eficazes dependem de um conjunto de ferramentas para gerenciamento, comunicação e controle de versão.

* **Controle de Versão com Git & GitHub**. Git é o padrão da indústria para controle de versão, essencial para o desenvolvimento colaborativo de software. O GitHub oferece uma plataforma centralizada na nuvem para hospedar repositórios Git e inclui funcionalidades para revisão de código (pull requests) e gerenciamento de projetos. É amigável para iniciantes e amplamente utilizado em ambientes profissionais e de código aberto.
* **Gerenciamento de Projetos:** Para um projeto acadêmico, a abordagem visual e simples de quadros Kanban do Trello ou Jira é altamente eficaz para gerenciar o Product Backlog e os Sprint Backlogs.
* **Comunicação da Equipe:** **Discord ou Slack**. Uma ferramenta de comunicação dedicada é vital. Tanto o Slack quanto o Discord oferecem mensagens em tempo real, canais para tópicos específicos (ex: #geral, #frontend, #backend, #random) e se integram bem com outras ferramentas como o GitHub, apoiando o valor Scrum de transparência.

**7. Conclusão**

Este plano de projeto fornece uma estrutura abrangente e detalhada para a concepção, desenvolvimento e entrega bem-sucedida da aplicação **CarrerFlow**. Ao adotar o framework Scrum, uma stack tecnológica moderna e práticas robustas de gerenciamento de escopo, qualidade e risco, a equipe está posicionada não apenas para entregar um produto funcional que atenda a uma necessidade real do mercado, mas também para demonstrar um alto nível de competência em engenharia de software e gerenciamento de projetos.

O sucesso dependerá da adesão disciplinada aos processos aqui definidos, da comunicação transparente e do compromisso coletivo com os objetivos do produto e do processo. A execução deste plano permitirá que a equipe encare os desafios inerentes ao processo de desenvolvimento de software, transformando a ideia inicial do **CarrerFlow** em uma aplicação de alta qualidade, entregue dentro do prazo e do escopo definidos.

## **Referências**

1. ATLASSIAN. **Funções do Scrum**. Atlassian. Disponível em:<https://www.atlassian.com/br/agile/scrum/roles>. Acesso em: 12 set. 2025.
2. ATLASSIAN. **O que é Scrum?**. Atlassian. Disponível em:<https://www.atlassian.com/br/agile/scrum>. Acesso em: 10 set. 2025.
3. CHANTY. **As 10 melhores ferramentas de comunicação em equipe para experimentar em 2025**. Chanty Blog, 2025. Disponível em:<https://www.chanty.com/blog/pt/team-communication-tools-pt/>. Acesso em: 09 set. 2025.
4. FULL SCALE. **Top 5 Tech Stacks for Web Development in 2025**. Full Scale Blog, 2025. Disponível em:<https://fullscale.io/blog/top-5-tech-stacks/>. Acesso em: 14 set. 2025.
5. GURU. **Documentação de software: o que é, tipos e práticas recomendadas**. Guru, 2024. Disponível em:<https://www.getguru.com/pt/reference/software-documentation>. Acesso em: 14 set. 2025.
6. INSTAGANTT. **Modelo de plano de projeto de desenvolvimento de software**. Instagantt. Disponível em:<https://www.instagantt.com/br/software-development-project-plan-template>. Acesso em: 13 set. 2025.
7. LYNCAS. **Boas práticas para escrever requisitos de software**. Lyncas Blog. Disponível em:<https://lyncas.net/blog/boas-praticas-para-escrever-requisitos-de-software/>. Acesso em: 11 set. 2025.
8. MADEFY. **As 10 melhores práticas para desenvolvimento de software**. Madefy Blog. Disponível em:<https://madefy.com.br/as-10-melhores-praticas-para-desenvolvimento-de-software/>. Acesso em: 11 set. 2025.
9. MILARE, Bruna Naira; LARIEIRA, Claudio Luis Carvalho. Gestão de riscos em projetos de desenvolvimento de software com Scrum: um estudo de caso. **GEP**,, v. 10, n. 2, p. 165-183, ago. 2019.
10. PROJECT MANAGEMENT INSTITUTE. **Um Guia do Conhecimento em Gerenciamento de Projetos (Guia PMBOK®)**. 6. ed. Newtown Square, PA: Project Management Institute, 2017.
11. TABLEAU. **O que são metas SMART?**. Tableau. Disponível em:<https://www.tableau.com/pt-br/learn/articles/smart-goals-criteria>. Acesso em: 15 set. 2025.
12. TARGET SO. **Projetos de TI: como gerir riscos e desafios inesperados?**. Target SO Blog. Disponível em:<https://www.targetso.com/artigos/projetos-de-ti-como-gerir-riscos/>. Acesso em: 12 set. 2025.
13. TERLIZZI, Marco Alexandre; BIANCOLINO, César Augusto. Estrutura analítica de riscos (EAR) como ferramenta de apoio à gestão de riscos em projetos. **Revista Gestão & Tecnologia**, Pedro Leopoldo, v. 14, n. 2, p. 49-76, mai./ago. 2014.
14. TRELLO. **Trello**. Disponível em:<https://trello.com/pt-BR>. Acesso em: 15 set. 2025.
15. ZUP. **Git, GitHub e GitLab: o que são e quais as diferenças?**. Zup Blog. Disponível em:<https://zup.com.br/blog/git-github-e-gitlab/>. Acesso em: 10 set. 2025.