Documento de Visão do Produto – LearnLoop

# 1. Nome do Projeto

LearnLoop

# 2. Objetivo Geral

Desenvolver uma plataforma digital baseada na metodologia Scrum, adaptada para contextos educacionais, com o intuito de auxiliar alunos, professores e coordenadores a organizar e acompanhar projetos acadêmicos de forma colaborativa, prática e eficiente.

# 3. Público-Alvo

- Alunos (ensino fundamental, médio e superior)  
- Professores  
- Coordenadores pedagógicos e profissionais da educação

# 4. Justificativa do Projeto

No cenário educacional atual, há uma demanda crescente por metodologias ativas e ferramentas que promovam o trabalho em equipe, a autogestão e o acompanhamento de projetos. O LearnLoop se propõe a preencher essa lacuna, oferecendo uma ferramenta simples, interativa e pedagógica baseada nos princípios do Scrum, adaptada para o ambiente acadêmico.

# 5. Funcionalidades Principais (Escopo do MVP)

Projetos:  
- Criar e remover projetos  
- Adicionar e remover colaboradores  
- Gerenciar permissões dos participantes  
  
Tarefas:  
- Criar, editar e excluir tarefas  
- Atribuir e desatribuir tarefas  
- Definir prioridade  
- Definir estado (ex: “a fazer”, “em andamento”, “concluído”)  
- Enfileirar tarefas por ordem de execução  
  
Usuário:  
- Cadastrar conta / login  
- Editar e remover perfil  
- Visualizar permissões dentro de cada projeto  
  
Telas:  
- Login e cadastro  
- Painel de projetos do usuário  
- Tela de visualização e gerenciamento de projeto  
- Tela de visualização de tarefas  
- Tela de criação/edição de tarefa  
- Tela de criação/edição de projeto  
- Tela de gerenciamento de permissões  
- Área administrativa do sistema (SysAdmin)

# 6. Fora do Escopo Inicial (MVP)

- Funções de acessibilidade  
- Integrações externas com plataformas de ensino (ex: Moodle) ou outros softwares  
- Suporte a múltiplos idiomas  
- Gamificação ou conquistas  
- Relatórios analíticos e estatísticas avançadas

# 7. Arquitetura e Tecnologia

- Framework principal: Django (Python)  
- Motivação técnica:  
 - Reduz complexidade com banco de dados  
 - Escalabilidade horizontal de desenvolvimento  
 - Ótima estrutura para projetos em equipe

# 8.Template de Labels

8.1. Template para a Label [Épico]:  
[Épico] Nome do Épico

**🎯 Objetivo do Épico**

Descreva de forma clara e resumida **o que esse épico visa resolver** ou entregar. Pense em uma visão geral do recurso ou funcionalidade.

Exemplo: Criar toda a estrutura de gerenciamento de tarefas, permitindo que usuários possam criar, editar, priorizar e acompanhar tarefas dentro de um projeto.

🧩 **Histórias de Usuário Relacionadas**

Liste aqui as issues (histórias) que fazem parte desse épico. Você pode marcar com - [ ] para acompanhar o progresso diretamente.

Exemplo:

- [ ] #12 Criar tarefa

- [ ] #13 Editar tarefa

- [ ] #14 Definir prioridade

- [ ] #15 Atribuir usuário

Quando essas issues forem criadas, basta colar o número delas com #, e o GitHub cria o link automático.

**✅ Critérios de Aceitação**

Liste os critérios que **precisam ser atendidos para o épico ser considerado concluído**:

* Todas as histórias relacionadas foram concluídas
* Testes manuais ou automatizados executados
* Validação com usuários ou stakeholders
* Documentação criada ou atualizada

#### 🛠️ ****Notas Técnicas (opcional)****

Espaço para anotações de arquitetura, bibliotecas envolvidas, considerações técnicas ou possíveis obstáculos.

- Usar Django com ORM para tarefas e relacionamentos

- Verificar integração futura com módulo de notificações

**🔗 Relacionamentos**

Você pode usar o sistema de **Linked Issues** do GitHub para conectar este épico com outras issues que ele **depende ou está bloqueando**.

8.2. Template para a Label [História de Usuário]:

[História de Usuário] Nome resumido da funcionalidade

**📖 Como um(a) *[tipo de usuário]* quero *[ação ou funcionalidade]* para *[benefício ou motivo]***

Exemplo:  
**Como um(a)** aluno(a) **quero** criar tarefas dentro de um projeto **para** organizar minhas entregas de trabalho acadêmico.

**🎯 Objetivo**

Descrever brevemente o que essa história se propõe a resolver. Seja direto e prático.

**✅ Critérios de Aceitação**

Esses são os requisitos mínimos que precisam estar presentes para que a história seja considerada "pronta":

* Deve permitir ao usuário criar uma tarefa com nome, descrição e prazo
* A tarefa deve estar vinculada a um projeto existente
* Após criada, deve aparecer na tela de tarefas
* A tarefa deve ser editável

#### 🔍 ****Cenários de Teste (opcional)****

Descreva exemplos práticos de como a funcionalidade pode ser testada ou validada:

Exemplo:

Dado que estou logado e em um projeto

Quando clico em "Criar tarefa" e preencho os campos obrigatórios

Então a nova tarefa deve aparecer na lista de tarefas daquele projeto

**🛠️ Notas Técnicas (opcional)**

Se necessário, use esse espaço para observações técnicas:

Exemplo:

- Campos obrigatórios: nome, projeto

- Usar form Django ModelForm com validação client-side

**🔗 Relacionamentos**

Você pode usar o sistema de **Linked Issues** do GitHub para conectar este épico com outras issues que ele **depende ou está bloqueando**. E também usar textos caso queira deixar mais claro.

Exemplo:

Parte do épico: #11

Bloqueia: #16 Integração com notificações

8.2. Template para a Label [Tarefa Técnica]:

[Tarefa Técnica] Nome resumido da atividade

**🧠 Descrição**

Descreva o que precisa ser feito de forma objetiva e clara. Aqui entram coisas como:

Exemplo:  
**Implementar autenticação com Django AllAuth para permitir login e registro de usuários.**

**🎯 Objetivo**

Explique por que essa tarefa é necessária. Pode ser por questões de performance, segurança, organização do código, integração, etc.

Exemplo:  
**Facilitar o controle de login social e validação de email com um pacote confiável e bem documentado.**

**✅ Critérios de Aceitação  
  
Exemplo:**

* Funcionalidade implementada e testada
* Não quebrou outras partes do sistema
* Cobertura de testes mínima mantida (se aplicável)
* Documentação (interna ou externa) atualizada

🧪 **Cenários ou Estratégia de Teste**

**Exemplo:**

- Testar autenticação com email e senha

- Testar fluxo de redefinição de senha

- Validar criação de usuário via formulário

**🛠️ Notas Técnicas**

Espaço para você indicar dependências, pacotes a serem usados, decisões de arquitetura etc.

Exemplo:

- Instalar o pacote django-allauth

- Configurar `INSTALLED\_APPS` e rotas no settings

- Adicionar suporte ao template de email

**🔗 Relacionamentos**

Você pode usar o sistema de **Linked Issues** do GitHub para conectar este épico com outras issues que ele **depende ou está bloqueando**. E usar textos caso queira deixar mais claro.

Exemplo:

Bloqueia: #22 Registro de novos usuários

Relacionada a: #5 Setup inicial do projeto

# 9. Regras do repositório

Com o objetivo de garantir a integridade, organização e rastreabilidade do desenvolvimento colaborativo deste projeto, foram estabelecidas as seguintes diretrizes de contribuição, acordadas entre os membros da equipe:

**1. Controle de Acesso à Branch main**

* Apenas o responsável pelo repositório, na condição de **administrador**, possui autorização para realizar *push* diretamente na branch main.
* Os demais membros da equipe **não estão autorizados a realizar *push* diretamente na branch main** sob nenhuma circunstância.

**2. Política de *Merge* para a Branch main**

* Toda e qualquer alteração a ser incorporada na branch main deverá ser realizada por meio de um **Pull Request (PR)**.
* A aprovação de um PR para a branch main está condicionada ao cumprimento dos seguintes requisitos:
  + A branch de origem do PR deve estar **atualizada em relação à branch main**, sem conflitos de merge.
  + O PR deverá ser **aprovado por pelo menos três (3) dos cinco (5) membros da equipe**, por meio da funcionalidade de revisão do GitHub.
  + As alterações devem estar em conformidade com as **tarefas previstas no planejamento do projeto** e devidamente relacionadas às respectivas *issues* registradas.

**3. Boas Práticas de Submissão**

* Recomenda-se que cada PR inclua, sempre que possível:
  + Uma **descrição clara e objetiva** das alterações implementadas;
  + **Capturas de tela ou demonstrações visuais**, em casos de alterações na interface do usuário ou em fluxos de interação;
  + Referência direta às *issues* correspondentes, utilizando a sintaxe Closes #n.