LICENCIATURA EM ENGENHARIA INFORMÁTICA

STORYTAIL

ENGLISH BOOKS FOR YOUNG LEARNERS

LUÍS PINTO
MIGUEL PINHEIRO
RICARDO FREIXO
VALÉRIO PINHEIRO

LABORATÓRIO DE PROGRAMAÇÃO

VILA NOVA DE GAIA
JAN DE 2025

Resumo

O projeto **StoryTail** teve como objetivo principal o desenvolvimento de uma plataforma web interativa para a promoção da leitura e aprendizagem do inglês entre crianças dos 3 aos 9 anos e seus familiares. Esta plataforma foi concebida para oferecer uma experiência intuitiva e lúdica, permitindo o acesso a histórias interativas, funcionalidades educativas e opções de personalização. O design UI/UX focou-se na criação de um ambiente visualmente apelativo e responsivo, alinhado ao público-alvo e às diretrizes do manual de identidade visual, com destaque para a mascote central – uma raposa.

No domínio do desenvolvimento, foi implementada uma arquitetura robusta com recurso ao framework Laravel, estruturada no modelo MVC (Model-View-Controller). No front-end, foram integradas bibliotecas modernas de JavaScript e Blade para assegurar uma interface dinâmica e interativa, garantindo também a responsividade em diferentes dispositivos. O back-end incorporou funcionalidades como autenticação segura, gestão de catálogos, e mecanismos de armazenamento para ficheiros multimédia. Adicionalmente, a estrutura da base de dados foi desenhada para suportar as principais funcionalidades da plataforma, incluindo a gestão de utilizadores, livros e atividades interativas.

Entre os resultados mais relevantes, destacam-se a implementação de protótipos funcionais, a criação de stored procedures para personalização de conteúdos e sugestões de livros, e a configuração de triggers automáticas para atualização do rating médio dos livros. Estas funcionalidades asseguram uma experiência otimizada tanto para utilizadores finais (crianças e pais) quanto para os administradores da plataforma.

Em conclusão, o projeto **StoryTail** consolidou uma base sólida para o seu crescimento futuro, ao integrar soluções técnicas avançadas e estratégias de design centradas no utilizador. Como perspetiva futura, prevê-se a expansão do catálogo de conteúdos e a inclusão de novas funcionalidades educativas e interativas, reforçando o compromisso com a educação infantil e a aprendizagem do inglês.

Índice

Índice de figuras

1. Introdução

Este projeto visa desenvolver uma plataforma inovadora chamada StoryTail – English Books for Young Learners, que tem como objetivo proporcionar uma experiência educativa para crianças de 3 a 9 anos, promovendo o contato com a língua inglesa por meio de histórias infantis. Este projeto é influenciado por práticas amplamente discutidas na literatura, como a abordagem modular e organizada mencionada por Ahmed et al. (2002), que enfatiza a importância da estrutura de camadas na conceção de sistemas web. A plataforma StoryTail é um exemplo dessa aplicação, utilizando o padrão **Model-View-Controller (MVC)** para garantir uma separação clara de responsabilidades, promovendo a escalabilidade e manutenção futura. Além disso, a relevância da utilização de recursos da Web e de ferramentas tecnológicas modernas, destacada por Moreira (2003), sustenta a base tecnológica da plataforma, aproveitando Web Services para a integração de funcionalidades e serviços adicionais. Este relatório documenta o processo de conceção, desenvolvimento e implementação do sistema, abordando os requisitos funcionais da plataforma. A plataforma StoryTail oferece uma biblioteca digital de histórias em inglês, permitindo às crianças, o acesso a uma variedade de conteúdos interativos, como vídeos e atividades lúdicas associadas aos temas de cada história.

1.1 Enquadramento

O projeto StoryTail está inserido no âmbito da Unidade Curricular (UC) de Laboratório de Programação (LP) da Licenciatura em Engenharia Informática (LEI), no Instituto Politécnico de Gestão e Tecnologia (ISLA), durante o ano letivo de 2024-2025. A plataforma representa um exemplo prático da integração de conhecimentos adquiridos ao longo do curso**.** Esta abordagem reforça a aplicação de competências teóricas e práticas no desenvolvimento de um projeto concreto, comum a todas as turmas do semestre. O desenvolvimento desta plataforma segue o modelo de separação de responsabilidades em camadas (Model-View-Controller), permitindo uma estrutura organizada e escalável para o sistema. Além disso, a interface do utilizador é projetada

para ser visualmente atrativa e acessível ao público infantil, garantindo uma experiência imersiva e divertida no aprendizado do inglês.

1.2 Objetivos

Os objetivos principais deste projeto são:

- Desenvolver uma plataforma web intuitiva que permita às crianças explorar histórias infantis em inglês, promovendo o desenvolvimento da leitura na língua estrangeira.
- Oferecer funcionalidades educativas, tais como atividades relacionadas com as histórias, que incentivem a aprendizagem de forma divertida.
- Fornecer recursos para pais para que possam acompanhar o progresso das crianças, permitindo a marcação de livros favoritos, registros de leitura e acesso a conteúdos adaptados à faixa etária.
- Implementar uma área administrativa que permita a gestão de conteúdos, como a adição e organização de livros, autores, e atividades de exploração.
- Facilitar o acesso ao conteúdo premium através de um sistema de planos de assinatura, onde os utilizadores logados podem ter acesso a funcionalidades adicionais.
- Garantir uma arquitetura robusta que permita a evolução da plataforma ao longo do tempo, com a adição de novos recursos.

2. Engenharia de Software

O design do StoryTail reflete as práticas discutidas em Ahmed et al. (2002), que promovem a construção de aplicações web robustas com uma arquitetura modular. A organização de requisitos funcionais e não funcionais segue uma abordagem sistemática, semelhante à defendida por Varajão (1998), assegurando que cada componente do sistema é desenhado e implementado com clareza e foco nos objetivos do projeto. Tendo como base as *user stories* especificadas para o Sprint A foram elaborados os capítulos abaixo mencionados. Após o levantamento dos requisitos

foram catalogados os requisitos funcionais e não funcionais, dando cumprimento a US_A001.

Requisitos não funcionais

• O público-alvo principal são crianças dos 3 aos 9 anos;

• A interface da plataforma web deve ser pautada por cores leves, seguindo o manual "brand Guidelines" disponibilizado e tendo a raposa como mascote de referência;

• A plataforma web deve ser apresentada em inglês ou bilingue em inglês e português;

Requisitos Funcionais

• Um utilizador visitante deve poder fazer login para assim aceder a mais conteúdo (nomeadamente premium);

• O utilizador administrador deve poder gerir conteúdos, nomeadamente adicionar livros ao sistema;

• O sistema deverá permitir guardar os seguintes dados de cada livro: título, fotografia da capa, breve descrição, autor(es), faixa etária adequada, tags, rating, páginas do texto integral e/ou áudio e/ou vídeo, atividades de exploração, outros elementos;

• Um utilizador deve poder ler um livro por três formas: i) página-a-página folheadas digitalmente; ii) narração em áudio; ou iii) vídeo;

• As atividades de exploração têm como objetivo imergir na mensagem do livro de forma lúdica, podendo ser i) sugestões para desenhar; ii) sugestões para explorar temas relacionados com o contexto; iii) jogos quiz; ou iv) outros;

• Um utilizador logado deve ter acesso a uma página de perfil pessoal com os seus dados; livros marcados como favoritos, livros lidos, e livros em exploração (com percentagem de visualização ou progresso bar); bem como a possibilidade de pedir acesso a conteúdo premium para o seu perfil;

• Os autores de livros devem ter uma página individual com informação relevante, tal como: nome, fotografia, nacionalidade, breve descrição e livros associados;

• Quando um utilizador logado termina a leitura de um livro, deve poder atribuir um rating.

Para cumprir a US_A002 foi elaborado um diagrama de Use Cases (Figura 1) para perceber as funcionalidades do sistema. Sendo também executado um modelo relacional de dados (Figura 2) para cumprimento da US_A003. O domínio do sistema foi representado através da Figura 3 exigida na US_A004.

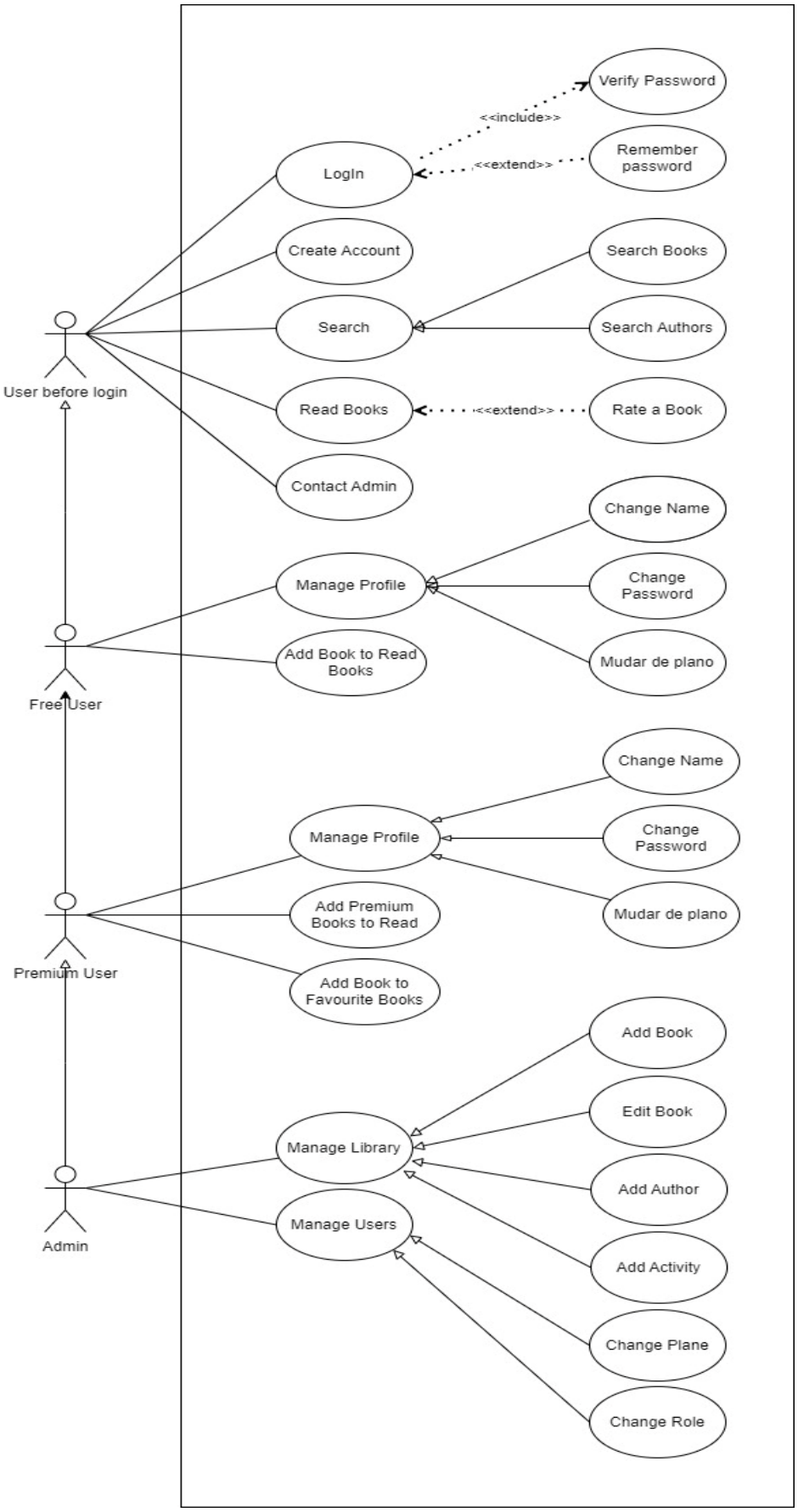

Figura 1 Diagrama de Use Cases do sistema StoryTail

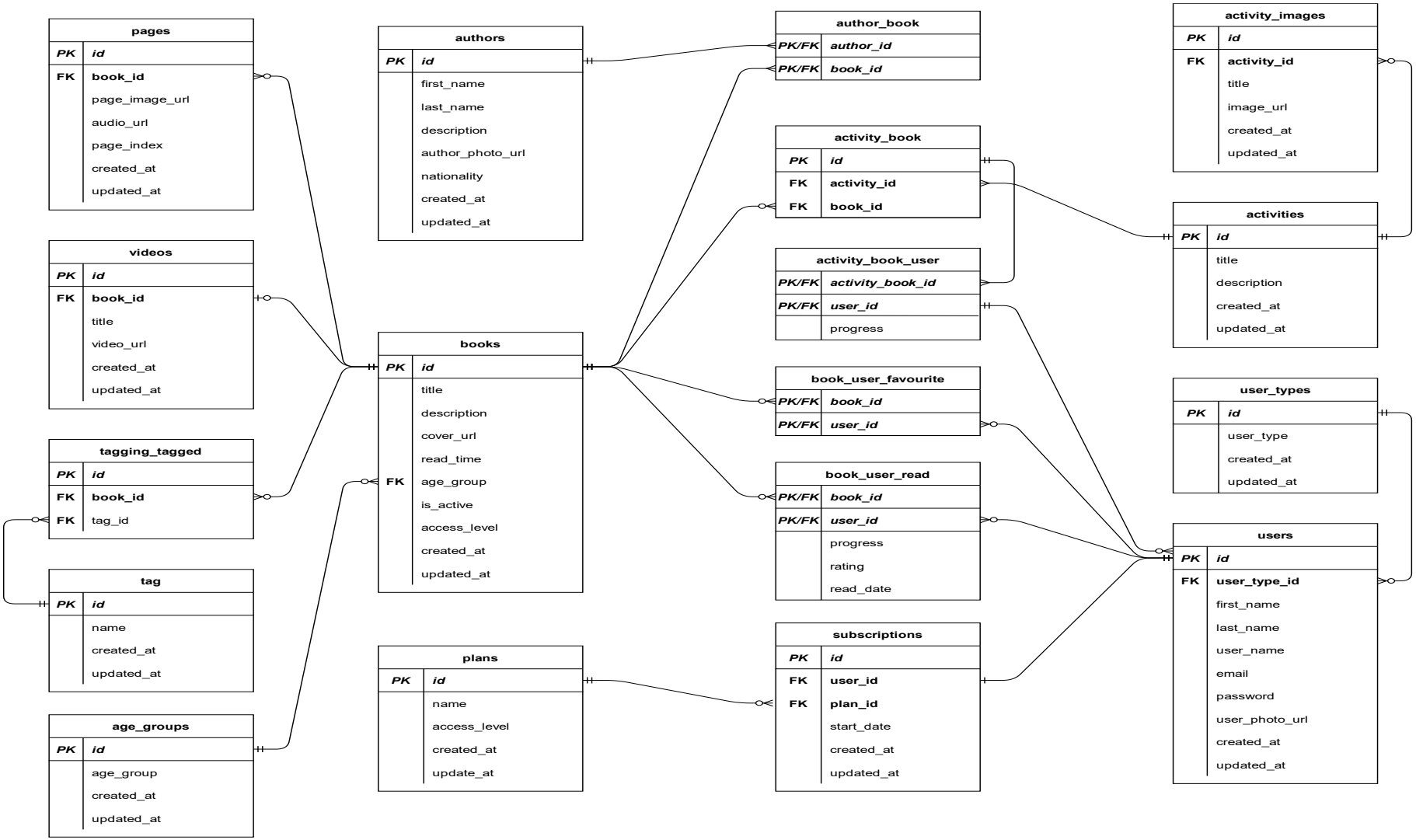

Figura 2 Modelo Relacional de Dados

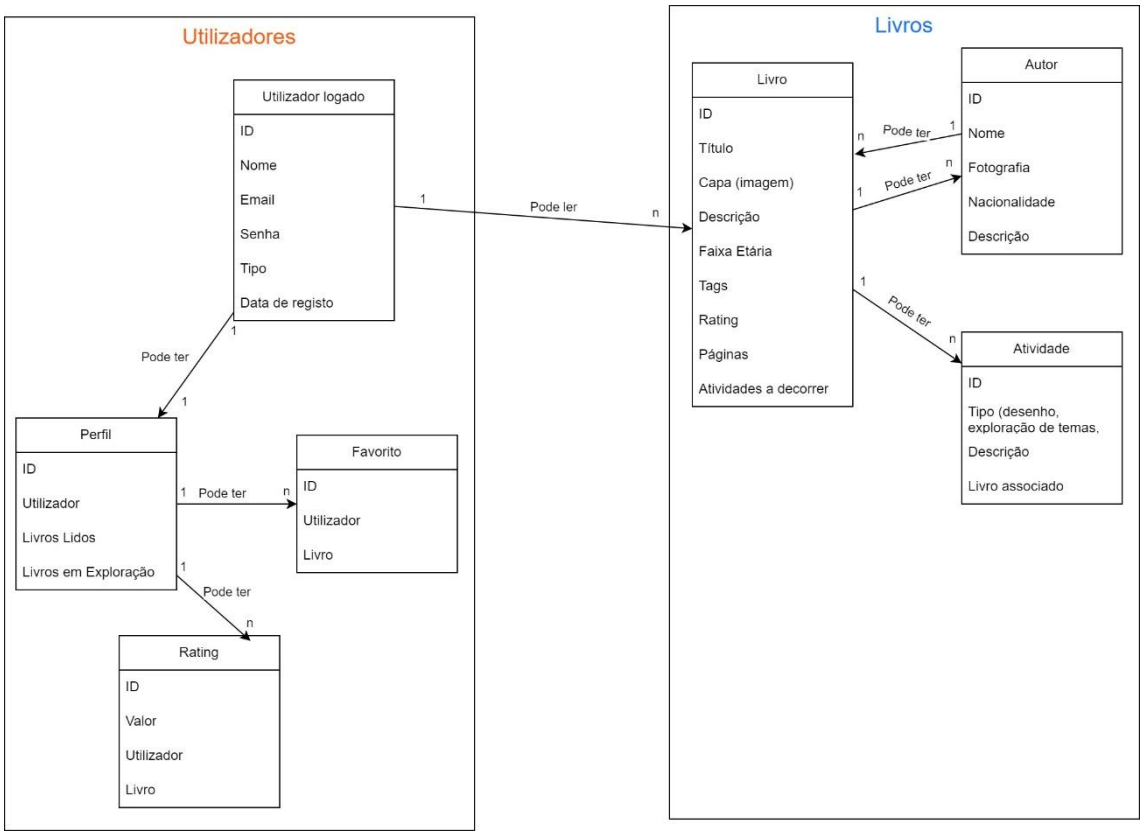

Figura 3 Modelo de Domínio

Para perceber a interação dos componentes do sistema, para o UC “Ler um Livro” foi elaborado Diagrama de Sequência (Figura 4) US_A005. Bem com a ação “Adicionar um livro” (Figura 5) US_A006.

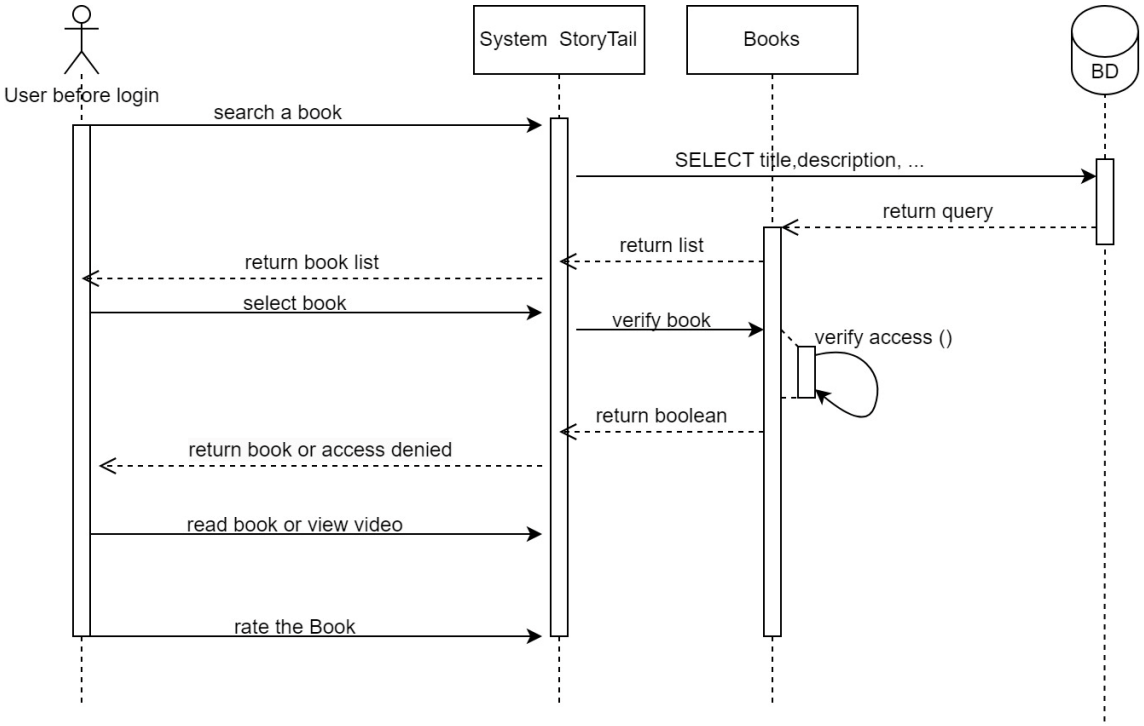

Figura 4 Diagrama de Sequência "Ler um Livro"

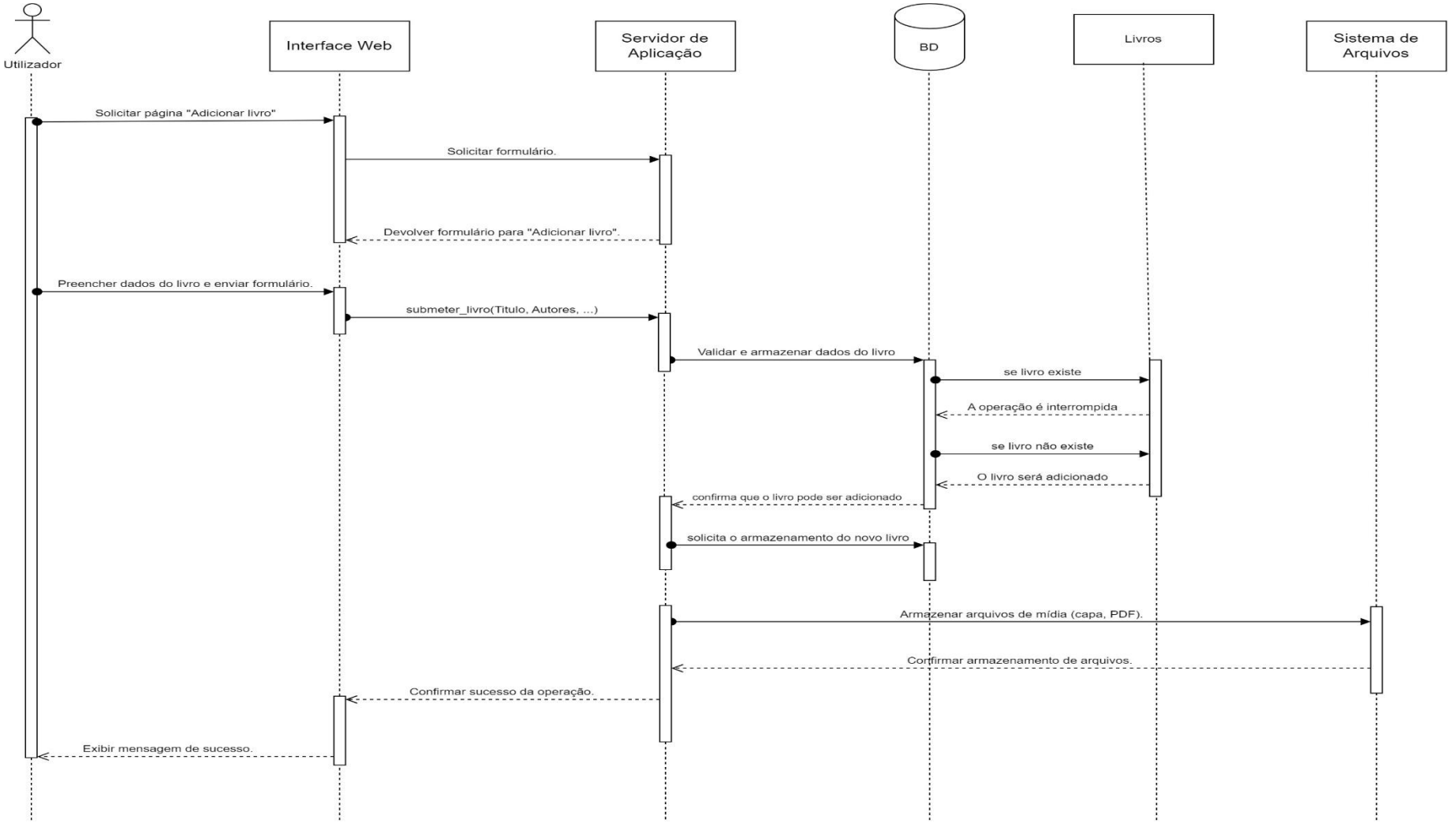

Figura 5 Diagrama de Sequência "Adicionar um Livro"

Para dar cumprimento a US_A007 foi detalhado o cenário de sucesso do UC “Ler um Livro” através da descrição estruturada a seguir relatada. O utilizador não logado procura primeiramente um livro no sistema storytail, em que este requisita na base de dados e devolve um objecto lista de livros, depois o utilizador seleciona um livro dentro da lista devolvida e é feita uma verificação se o utilizador possui acesso ao livro ou não. Tendo depois como opção ler o livro ou ver o video bem como avaliar o livro.

De igual forma foi detalhada a UC “Adicionar um Livro”. No cenário principal do caso de uso "Adicionar um Livro", o utilizador, autenticado no sistema e com as permissões adequadas, solicita a página "Adicionar Livro" através da interface web. O sistema devolve um formulário, que o utilizador preenche com os dados do livro, como o título e os autores. Após submeter o formulário, o sistema valida as informações e verifica se o livro já existe na base de dados. Caso o livro já exista, a operação é interrompida; caso contrário, o sistema continua e armazena os dados do novo livro. Além disso, o sistema solicita o armazenamento dos ficheiros associados, como a capa e o PDF, no sistema de ficheiros. Depois de armazenar os dados e os ficheiros com sucesso, o sistema exibe uma mensagem de confirmação ao utilizador, indicando que o livro foi adicionado corretamente ao catálogo.

O detalhe da informação do modelo de dados foi obtido através de um dicionário de dados (Tabela 1) para cumprimento da US_A009. Elaborado também um fluxograma relativo ao processo “Ler um Livro” (Figura 6) para cumprimento da US_A0010. O contexto da informação deste sistema está representado na Figura 7, e o diagrama de Gantt para cada uma das tarefas com os respetivos elementos na Figura 8.

Tabela 1 Dicionário de dados relativo ao Modelo de dados

Pages	**Videos**
Descrição: Armazena informações sobre páginas de um livro.	**Descrição:** Armazena vídeos relacionados aos livros.
id (PK): Identificador único da página. Int	**id (PK)**: Identificador único do vídeo **int**
book_id (FK): Referência para o livro. String	**book_id (FK**): Referência para o livro. **Int**
page_image_url: URL da imagem da página. String	**title: Título do vídeo. String**
audio_url: URL do áudio da página (se houver). string	**video_url**: URL do vídeo. **String**
page_index: Número da página. Int	**created_at**: Data de criação. Date
created_at: Data de criação. Date	**updated_at**: Data de última atualização. Date
updated_at: Data de última atualização. Date	

author_book	**activities**
Descrição: Relaciona autores e livros.	**Descrição**: Armazena atividades relacionadas aos livros.
author_id (FK): Referência para o autor. **Int**	id (PK): Identificador único da atividade. Int
book_id (FK): Referência para o livro. Int	title: Título da atividade. String
	description: Descrição da atividade. Big text
	created_at: Data de criação. Date
	updated_at: Data de última atualização. Date
users	**user_types**
Descrição: Armazena informações sobre os utilizadores.	**Descrição: Armazena os diferentes tipos de utilizadores.**
id (PK): Identificador único do utilizadores.	id (PK): Identificador único do tipo de utilizadores. Int

user_type_id (FK): Referência para o tipo de utilizadores. Int	user_type: Descrição do tipo de utilizadores.string
first_name: Primeiro nome. String	created_at: Data de criação. date
last_name: Sobrenome. String	updated_at: Data de última atualização. Date
user_name: Nome de utilizadores. String	
email: E-mail do utilizadores. String	
password: Senha do utilizadores. String	
user_photo_uri: URL da foto do utilizadores. String	
created_at: Data de criação. Date	
updated_at: Data de última atualização. Date	

subscriptions
Descrição: Armazena as assinaturas dos utilizadores aos planos.
id (PK): Identificador único. Int
user_id (FK): Referência para o utilizadores. Int
plan_id (FK): Referência para o plano. Int
start_date: Data de início da assinatura. Date
created_at: Data de criação. Date
updated_at: Data de última atualização. Date

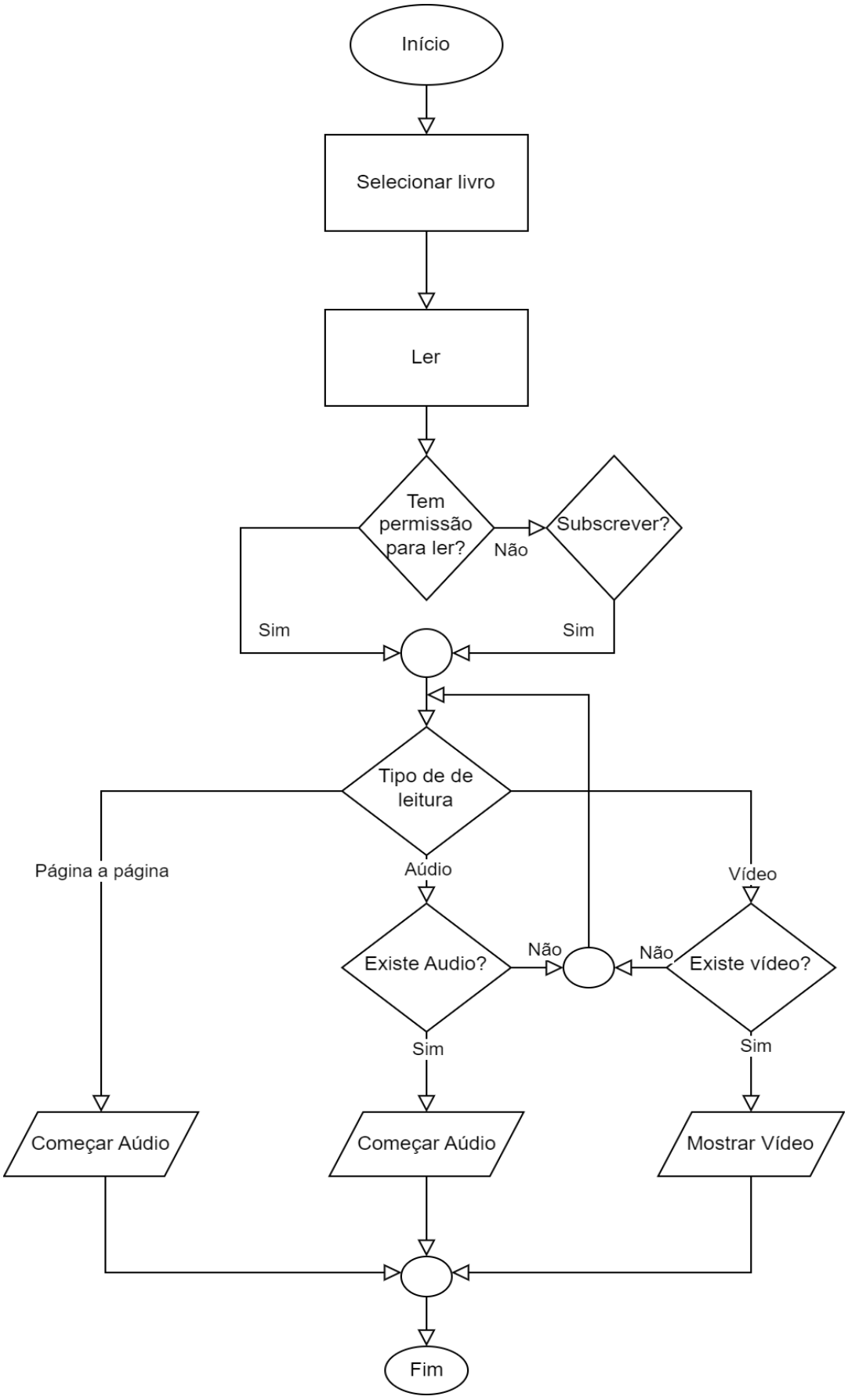

Figura 6 Fluxograma processo "Ler um Livro"

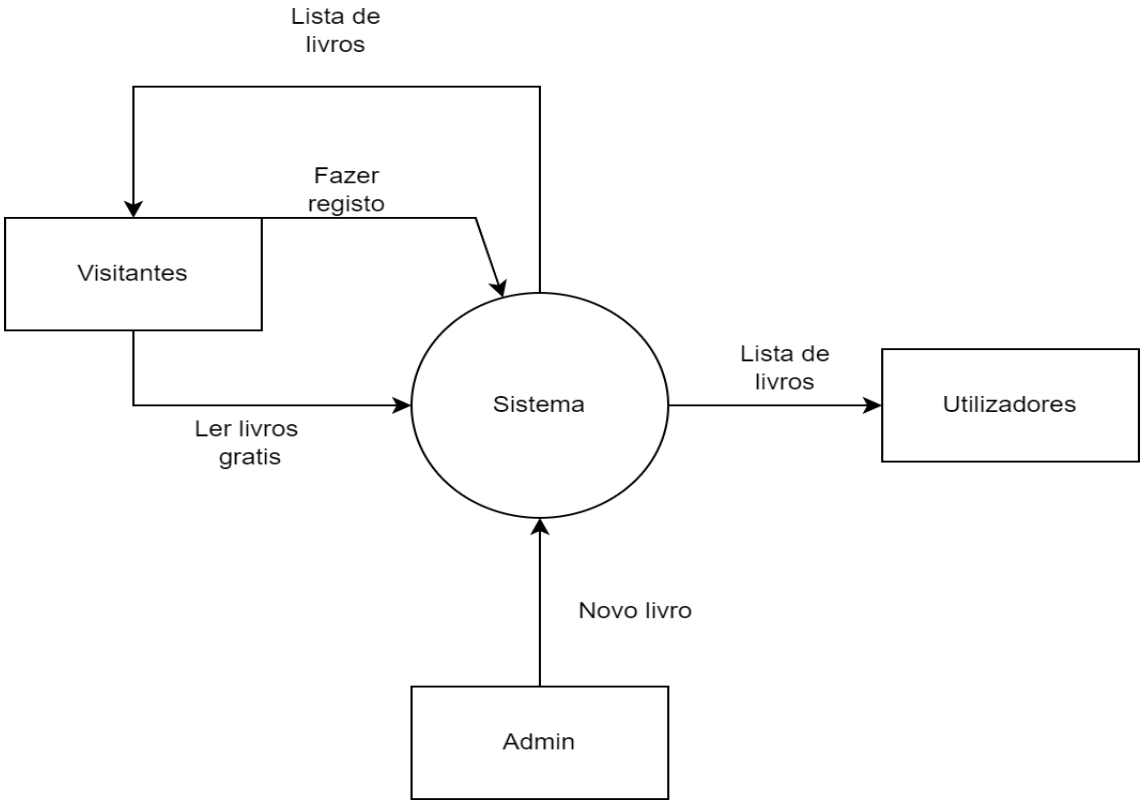

Figura 7 Diagrama de Contexto

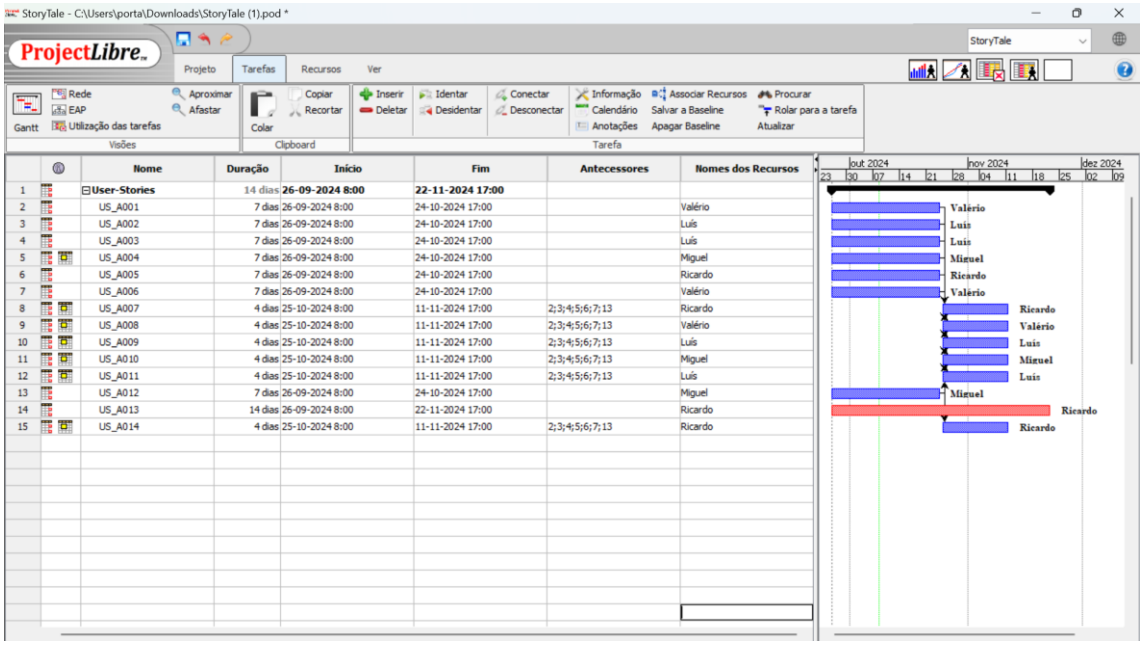

Figura 8 Diagrama de Gantt

3. UI/UX

O projeto StoryTail visa desenvolver uma plataforma web interativa de histórias infantis em inglês, voltada para crianças de 3 a 9 anos e seus familiares. O desafio da interface e experiência do utilizador (UI/UX) neste contexto envolve criar um ambiente intuitivo, e lúdico, que promova o interesse pela leitura e aprendizado do idioma de forma agradável e envolvente.

Para atingir esses objetivos, o design deve alinhar-se ao público-alvo, utilizando cores leves e elementos visuais atrativos, em consonância com o manual de identidade visual e tendo a mascote (uma raposa) como figura central de referência. A usabilidade é fundamental, pois a plataforma deve permitir uma navegação intuitiva para crianças, que poderão explorar livros, realizar atividades, marcar livros favoritos e acompanhar o progresso de leitura.

Além disso, a interface precisa ser inclusiva e responsiva, adaptando-se a diferentes dispositivos e integrando funcionalidades de fácil acesso tanto para utilizadores não logados quanto logados (com conteúdo adicional para utilizadores premium). Neste sentido, o trabalho em UI/UX envolverá o desenvolvimento de protótipos que ofereçam uma experiência simplificada, eficiente e visualmente agradável, promovendo uma interação intuitiva e segura para os jovens leitores. Foi desenvolvido um protótipo da plataforma web (Figura 8) conforme o manual de identidade visual 'Brand Guidelines' fornecido (US_B009).

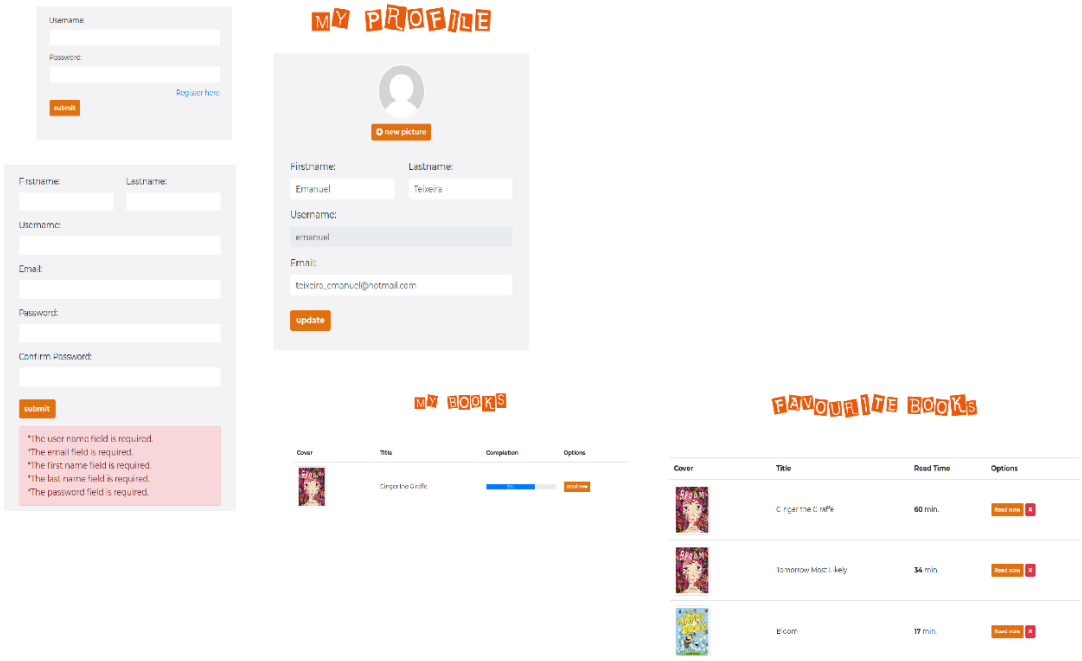

Figura 8 Protótipo da plataforma web com os vários ecrãs chave

Foram detalhadas três personas relativas aos utilizadores (US_B010), bem como especificada uma User Journey para cada persona definida (US_B011).

Personas

Criança (6 anos) - Tomás

Perfil: Tomás é um menino curioso e gosta de explorar novas histórias.

Necessidades: Acesso fácil a histórias com opções de áudio e vídeo.

Objetivos: Encontrar histórias interativas e acompanhar seu progresso.

Frustrações: Dificuldade em encontrar histórias sem suporte visual e de áudio. Design excessivamente complexo ou menus escondidos que dificultam a exploração livre.

Mãe Utilizadora (Laura, 35 anos)

Perfil: Laura é mãe do Tomás e procura ferramentas educativas para introduzir o inglês ao filho. Ela quer acompanhar o progresso dele, preferindo plataformas com conteúdos apropriados para a aprendizagem.

Necessidades: Um layout intuitivo, onde possa aceder rapidamente o progresso do filho e gerir conteúdos. A plataforma deve oferecer conteúdo adaptado para a faixa etária do filho.

Objetivos: Ajudar o filho a ler mais histórias, incentivando-o a aprender inglês.

Frustrações: Falta de controlo sobre o conteúdo acessível para o filho, dificuldades para encontrar conteúdos de acordo com o nível dele.

Administrador (Carlos, 40 anos)

Perfil: Carlos é responsável por inserir novos conteúdos e atividades na plataforma. Ele gere a base de dados e mantém a experiência interessante e relevante para o público-alvo.

Necessidades: Interface de backoffice eficiente e organizado que permita rápida inserção, edição e exclusão de conteúdos.

Objetivos: Manter a biblioteca atualizada, garantir que as atividades estejam adequadas e incentivar o uso educacional da plataforma, de acordo com o feedback.

Frustrações: Processos manuais ou repetitivos que tornem a gestão de conteúdo demorada.

User Journey

Persona 1: Tomás

Objetivo: Encontrar e explorar uma história em inglês com opções de narração em áudio e atividades.

Entrada na Plataforma

Motivação: Tomás está entusiasmado e quer explorar uma história nova.

Ação: Ele acede à plataforma no tablet, atraído por uma interface com ícones grandes e coloridos.

Exploração do Catálogo de Histórias

Motivação: Ele quer uma história visual e fácil de escolher.

Ação: Tomás rola pela lista de histórias usando o "infinite scroll" e vê as capas dos livros, estando algumas recomendadas para sua faixa etária.

Seleção da História

Motivação: Encontrar uma história com imagens coloridas e opções de narração.

Ação: Tomás toca na capa do livro, abrindo a página de detalhes. Ele escolhe a opção de narração em áudio.

Exploração e Interação

Motivação: Quer se divertir enquanto ouve a história.

Ação: Ele acompanha a narração em áudio, vendo as imagens e, ao fim de cada seção, interage com pequenas atividades, como quizzes e desafios de desenho, que aparecem de forma visual e com instruções simples.

Finalização

Motivação: Completar a história e ganhar algum tipo de reconhecimento.

Ação: Ao finalizar, ele vê uma barra de progresso aumentar e recebe uma “estrela” virtual por ter completado a leitura.

Persona 2: Laura

Objetivo: Acompanhar o progresso de leitura e aprendizagem do Tomás, acedendo a sugestões de histórias.

Acesso à Plataforma e Login

Motivação: Laura quer ver o que Tomás tem lido recentemente.

Ação: Ela entra na plataforma e faz login no perfil dela.

Visão Geral do Progresso do Tomás

Motivação: Verificar quais livros o filho leu e se interessou com as atividades.

Ação: Ela navega até o painel de progresso do Tomás, onde visualiza uma lista dos livros concluídos, os favoritos e o progresso nas atividades interativas.

Exploração de Novos Livros

Motivação: Procurar novas histórias que possam interessar ao filho.

Ação: Laura navega pelo catálogo e usa filtros (faixa etária, temas educativos) para encontrar novas sugestões de histórias.

Sugestão de Leitura

Motivação: Ajudar Tomás a escolher novas histórias.

Ação: Ela seleciona uma nova história e a marca como "favorita" para que ele a encontre facilmente. Também adiciona uma atividade como "quiz" para que ele explore ao terminar.

Verificação de Configurações

Motivação: Manter o perfil seguro e personalizado.

Ação: Laura reve as configurações para garantir que o conteúdo adequado à faixa etária e as preferências de Tomás estão bem definidas, ajustando se necessário.

Persona 3: Carlos

Objetivo: Inserir um novo livro na plataforma e configurar atividades no Acesso ao Backoffice

Motivação: Atualizar a biblioteca com um novo livro que acabou de ser adquirido.

Ação: Carlos faz login na área administrativa e acede o módulo de gestão de conteúdo.

Inserção do Novo Livro

Motivação: Adicionar o novo livro de forma rápida e intuitiva.

Ação: Ele insere as informações básicas (título, faixa etária, autor, tags, imagem da capa) e carrega o texto e áudio para as diferentes modalidades de leitura (texto, narração e vídeo).

Configuração de Atividades Interativas

Motivação: Criar atividades para reforçar a mensagem educativa do livro.

Ação: Carlos configura um quiz com perguntas relacionadas ao enredo e uma atividade de desenho. Ele define o nível de acesso como “restrito” (somente para usuários premium).

Revisão e Publicação

Motivação: Garantir que todas as informações estão corretas.

Ação: Antes de publicar, Carlos revê todos os dados do livro, as atividades e salva as alterações, tornando o conteúdo ativo na plataforma.

Foram distribuídas as US para cada elemento representadas no diagrama de Gantt da figura 9(US_B013).

User-Stories - Sprint B	14 dias	17-10-20...	22-10-202...		
US_B001	7 dias	17-10-202...	19-10-2024 ...		Miguel
US_B002	7 dias	19-10-202...	22-10-2024 ...	17	Miguel
US_B003	7 dias	19-10-202...	22-10-2024 ...	17	Miguel
US_B004	7 dias	19-10-202...	22-10-2024 ...	17	Miguel
US_B005	7 dias	19-10-202...	22-10-2024 ...	17	Miguel
US_B006	7 dias	19-10-202...	22-10-2024 ...	17	Miguel
US_B007	7 dias	19-10-202...	22-10-2024 ...	17	Valério
US_B008	7 dias	17-10-202...	19-10-2024 ...		
US_B009	7 dias	17-10-202...	19-10-2024 ...		Luís
US_B010	7 dias	17-10-202...	19-10-2024 ...		Ricardo
US_B011	7 dias	17-10-202...	19-10-2024 ...		Ricardo
US_B012	7 dias	17-10-202...	19-10-2024 ...		
US_B013	7 dias	17-10-202...	19-10-2024 ...		Valério
US_B014	14 dias	17-10-202...	22-10-2024 ...		Ricardo
US_B015	7 dias	17-10-202...	19-10-2024 ...		Luís
US_B016	14 dias	17-10-202...	22-10-2024 ...		Ricardo

Figura 9 Diagrama de Gant com distribuição das US para cada elemento

Foi também desenvolvido um poster apresentação do projeto representado na figura 10.

Figura 10 Poster apresentação do projeto

4. Desenvolvimento

4.1. Arquitetura de Software

A arquitetura do projeto foi estruturada utilizando o framework Laravel, que segue o padrão de design MVC (Model-View-Controller). Essa abordagem divide a aplicação em três camadas principais:

- **Model:** Responsável pela lógica de negócios e pela interação com a base de dados. No Laravel, utilizamos o Eloquent ORM para manipulação de dados de forma eficiente, além de consultas SQL otimizadas com stored procedures e views criadas previamente.

- **View:** Implementada utilizando o **Blade**, o motor de templates nativo do Laravel, que permitiu criar interfaces dinâmicas e reutilizáveis.

- **Controller:** Intermediário entre as views e os models, processando requisições HTTP e executando ações de acordo com a lógica de negócios.

Adicionalmente, o Laravel foi escolhido por suas funcionalidades nativas que garantem segurança e escalabilidade. A modularidade deste framework também possibilita a rápida adição de novas funcionalidades sem comprometer a estrutura existente.

4.2 Base de Dados

A estrutura da Base de Dados foi implementada com sucesso, permitindo armazenar as informações de livros, atividades e utilizadores da plataforma StoryTail. Esse passo inicial permitiu uma base robusta para o desenvolvimento das funcionalidades subsequentes (US_B001) (Anexo A). Três registos foram inseridos em cada tabela para simular dados reais. Essa inserção inicial possibilitou a validação da estrutura e das consultas, garantindo que as interações e testes funcionassem conforme esperado com

informações simuladas. (US_B002) (Anexo B). As stored procedures foram desenvolvidas para listar todos os livros de acordo com filtros recebidos por parâmetros. Com isso, a plataforma agora pode oferecer opções de busca personalizada, essencial para atender a diversas necessidades e preferências dos utilizadores (US_B003) (Anexo C). A criação de stored procedures para listar atividades associadas a cada livro permite que a plataforma exiba conteúdos adicionais específicos de cada história (US_B004) (Anexo D). Outra stored procedure foi implementada para listar livros favoritos e lidos de cada utilizador, com o progresso de leitura incluído. (US_B005) (Anexo E). A plataforma agora conta com uma stored procedure que sugere livros com base no perfil do utilizador, utilizando seu histórico e preferências. Essa funcionalidade melhora a personalização, ajudando na descoberta de novos conteúdos relevantes (US_B006) (Anexo F). Uma view foi criada para identificar e exibir os livros mais populares nos últimos três meses. Com essa funcionalidade, a plataforma oferece informação sobre os conteúdos mais consumidos entre os utilizadores, ajudando a destacar conteúdos que têm maior apelo (US_B007) (Anexo G). Como atividade extra foi desenvolvido o rating médio de um livro automaticamente.

4.3 Front-end

Durante o desenvolvimento do front-end, foi utilizada a framework Laravel em combinação com o motor de templates Blade, assegurando a criação de interfaces dinâmicas e intuitivas. Foram integrados frameworks de CSS, como o Bootstrap, para garantir responsividade e uniformidade no design em diferentes dispositivos. As principais funcionalidades implementadas incluem:

- Página inicial com destaque para livros recomendados, utilizando dados fornecidos pelos controladores;
- Filtros no catálogo de livros para facilitar a pesquisa por idade, tema ou popularidade;
- Perfis de utilizador com acesso a progresso, favoritos e configurações personalizadas;

- Adaptação responsiva, permitindo uma experiência consistente em dispositivos móveis e desktops.

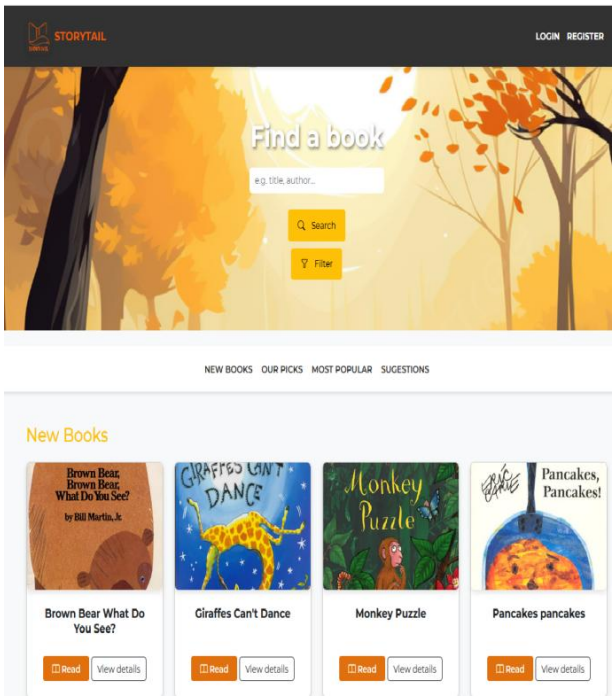

Figura 9 Front-end

4.4 Back-end

O back-end foi construído com o objetivo de suportar as funcionalidades do front-end de forma eficiente e segura. As principais características incluem:

- Implementação de APIs para gestão de livros, autenticação e integração de ficheiros multimédia (imagens e áudios);
- Gestão da base de dados com Eloquent ORM, permitindo consultas otimizadas e manipulação dinâmica de dados;
- Desenvolvimento de triggers e stored procedures para automatização de processos, como o cálculo do rating médio de livros e a sugestão de conteúdos personalizados;
- Suporte à autenticação via JWT, garantindo segurança na interação entre utilizadores e a plataforma.

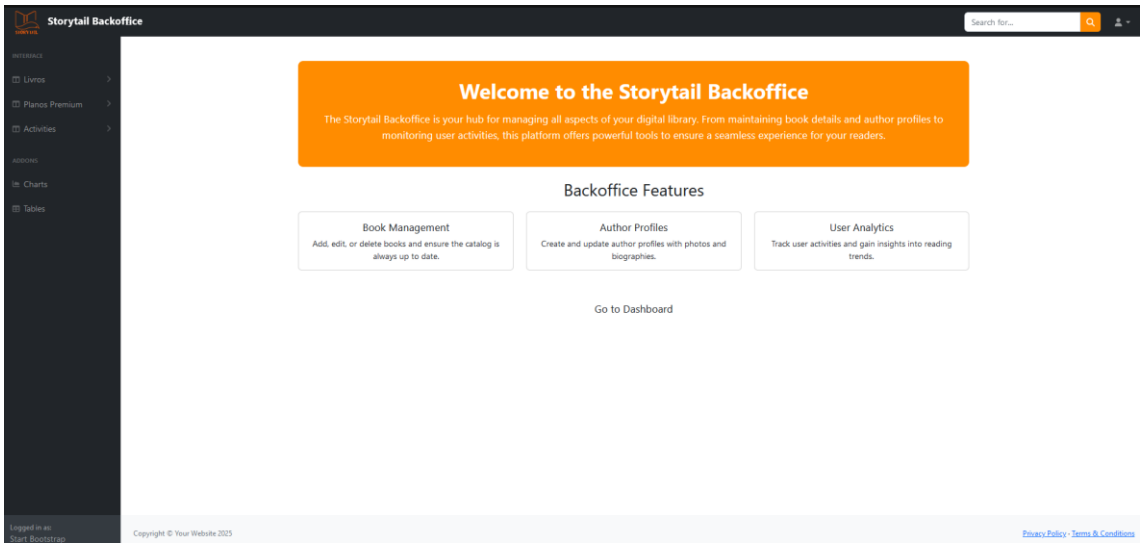

Figura 10 Backoffice

4.5 Cenário de Sucesso Principal

O cenário de sucesso principal desenvolvido consiste na jornada de um utilizador infantil (Tomás) ao explorar a plataforma. O Tomás consegue:

- Navegar facilmente pelo catálogo de livros graças a uma interface adaptada para crianças;
- Escolher um livro;
- Interagir com atividades educativas, como por exemplo desafios de desenho;
- Ler o livro;
- Adiciona-lo aos favoritos;
- Atribuir um rating;

5 Conclusões e perspetivas de trabalho futuro

O projeto **StoryTail** demonstrou ser uma solução viável para promover a leitura e a aprendizagem do inglês entre crianças, combinando tecnologia avançada com um design centrado no utilizador. A integração de ferramentas modernas, como o Laravel, e a atenção às necessidades do público-alvo permitiram criar uma base sólida para o crescimento e expansão da plataforma.

Como trabalho futuro, destacam-se:

- Expansão do catálogo de histórias com novos conteúdos multimédia;
- Introdução de funcionalidades adicionais, como jogos educativos e integração com ferramentas de aprendizagem online;
- Melhoria contínua da interface administrativa para facilitar a gestão de conteúdos;
- Otimização da personalização com algoritmos mais avançados de recomendação de livros.

Desta forma, a plataforma **StoryTail** tem o potencial de se tornar uma referência no ensino de inglês para crianças, contribuindo para a educação e desenvolvimento infantil.

6 Referências e Bibliografia

AHMED, M., GARRETT, C., FAIRCLOTH, J., PAYNE, C. [2002], *ASP.NET Web Developer's Guide*, Syngress, 2002.

MOREIRA, José J. [2003], *Plataforma .NET e Web Services*, FEUP, 2003.

VARAJÃO, J. E. Q. [1998], *A Arquitetura da Gestão de Sistemas de Informação*, FCA, 1998.

ANEXO A: Implementa a estrutura da Base de Dados (US_B001)

```
CREATE DATABASE bdteste;
USE bdteste;

CREATE TABLE authors (
   id INT AUTO_INCREMENT PRIMARY KEY,
   first_name VARCHAR(100),
   last_name VARCHAR(100),
   description TEXT,
   author_photo_url VARCHAR(255),
   nationality VARCHAR(100),
   created_at DATETIME,
   updated_at DATETIME
);

CREATE TABLE books (
   id INT AUTO_INCREMENT PRIMARY KEY,
   title VARCHAR(255),
   description TEXT,
   cover_url VARCHAR(255),
   read_time INT,
   rating_medio FLOAT,
   age_group VARCHAR(50),
   is_active BOOLEAN,
   access_level INT,
   created_at DATETIME,
   updated_at DATETIME
);
```

```
CREATE TABLE author_book (
    id INT AUTO_INCREMENT PRIMARY KEY,
    author_id INT,
    book_id INT,
    FOREIGN KEY (author_id) REFERENCES authors(id),
    FOREIGN KEY (book_id) REFERENCES books(id)
);

CREATE TABLE pages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    book_id INT,
    page_image_url VARCHAR(255),
    audio_url VARCHAR(255),
    page_index INT,
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (book_id) REFERENCES books(id)
);

CREATE TABLE videos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    book_id INT,
    title VARCHAR(255),
    video_url VARCHAR(255),
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (book_id) REFERENCES books(id)
);
```

```
CREATE TABLE tags (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    created_at DATETIME,
    updated_at DATETIME
);

CREATE TABLE tagging_tagged (
    id INT AUTO_INCREMENT PRIMARY KEY,
    book_id INT,
    tag_id INT,
    FOREIGN KEY (book_id) REFERENCES books(id),
    FOREIGN KEY (tag_id) REFERENCES tags(id)
);

CREATE TABLE age_groups (
    age_group VARCHAR(50) PRIMARY KEY,
    created_at DATETIME,
    updated_at DATETIME
);

CREATE TABLE activities (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255),
    description TEXT,
    created_at DATETIME,
    updated_at DATETIME
);
```

```
CREATE TABLE activity_images (
    id INT AUTO_INCREMENT PRIMARY KEY,
    activity_id INT,
    title VARCHAR(255),
    image_url VARCHAR(255),
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (activity_id) REFERENCES activities(id)
);

CREATE TABLE activity_book (
    id INT AUTO_INCREMENT PRIMARY KEY,
    activity_id INT,
    book_id INT,
    FOREIGN KEY (activity_id) REFERENCES activities(id),
    FOREIGN KEY (book_id) REFERENCES books(id)
);

CREATE TABLE user_types (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_type VARCHAR(100),
    created_at DATETIME,
    updated_at DATETIME
);

CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_type_id INT,
```

```
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    user_name VARCHAR(100),
    email VARCHAR(255),
    password VARCHAR(255),
    user_photo_url VARCHAR(255),
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (user_type_id) REFERENCES user_types(id)
);

CREATE TABLE activity_book_user (
    id INT AUTO_INCREMENT PRIMARY KEY,
    activity_book_id INT,
    user_id INT,
    progress INT,
    FOREIGN KEY (activity_book_id) REFERENCES activity_book(id),
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE book_user_favourite (
    id INT AUTO_INCREMENT PRIMARY KEY,
    book_id INT,
    user_id INT,
    FOREIGN KEY (book_id) REFERENCES books(id),
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE book_user_read (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
    book_id INT,
    user_id INT,
    progress INT,
    rating INT,
    read_date DATETIME,
    FOREIGN KEY (book_id) REFERENCES books(id),
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE plans (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    access_level INT,
    created_at DATETIME,
    updated_at DATETIME
);

CREATE TABLE subscriptions (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    plan_id INT,
    start_date DATETIME,
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES users(id),
    FOREIGN KEY (plan_id) REFERENCES plans(id)
);
```

ANEXO B: Inserção de 3 registos em cada tabela (US_B002)

```
INSERT INTO authors (first_name, last_name, description, nationality, created_at, updated_at)

VALUES

('John', 'Doe', 'Author of fictional books', 'American', NOW(), NOW()),

('Jane', 'Smith', 'Renowned for science fiction', 'British', NOW(), NOW()),

('Emily', 'Johnson', 'Specializes in historical novels', 'Canadian', NOW(), NOW());


INSERT INTO books (title, description, read_time, age_group, is_active, access_level, created_at, updated_at)

VALUES

('The Mystery Book', 'A thrilling mystery novel', 120, 'Adult', TRUE, 1, NOW(), NOW()),

('The Sci-Fi Adventure', 'Exploring galaxies and technology', 150, 'Young Adult', TRUE, 1, NOW(), NOW()),

('The History Chronicles', 'A dive into ancient civilizations', 180, 'Adult', TRUE, 1, NOW(), NOW());


INSERT INTO author_book (author_id, book_id)

VALUES

(1, 1),

(2, 2),

(3, 3);


INSERT INTO tags (name, created_at, updated_at)

VALUES

('Mystery', NOW(), NOW()),

('Science Fiction', NOW(), NOW()),

('History', NOW(), NOW());
```

```
INSERT INTO tagging_tagged (book_id, tag_id)
VALUES
(1, 1),
(2, 2),
(3, 3);

INSERT INTO age_groups (age_group, created_at, updated_at)
VALUES
  ('Infantil', NOW(), NOW()),
  ('Jovem', NOW(), NOW()),
  ('Adulto', NOW(), NOW());

INSERT INTO user_types (user_type, created_at, updated_at)
VALUES
('Admin', NOW(), NOW()),
('Editor', NOW(), NOW()),
('Viewer', NOW(), NOW());

INSERT INTO users (user_type_id, first_name, last_name, user_name, email, password,
user_photo_url, created_at, updated_at)
VALUES
(1, 'Alice', 'Smith', 'alice_s', 'alice@example.com', 'password123', NULL, NOW(),
NOW()),
(2, 'Bob', 'Brown', 'bob_b', 'bob@example.com', 'password456', NULL, NOW(), NOW()),
(3, 'Charlie', 'Davis', 'charlie_d', 'charlie@example.com', 'password789', NULL, NOW(),
NOW());

INSERT INTO plans (name, access_level, created_at, updated_at)
VALUES
('Premium Plan', 2, NOW(), NOW()),
```

```
('Standard Plan', 1, NOW(), NOW()),
('Basic Plan', 0, NOW(), NOW());

INSERT INTO subscriptions (user_id, plan_id, start_date, created_at)
VALUES
(1, 1, NOW(), NOW()),
(2, 2, NOW(), NOW()),
(3, 3, NOW(), NOW());

INSERT INTO book_user_favourite(book_id, user_id)
VALUES
(1, 1),
(2, 2),
(3, 3);

INSERT INTO activities (title, description, created_at, updated_at)
VALUES
('Quiz on Mystery Book', 'A quiz activity based on the mystery book', NOW(), NOW()),
('Puzzle Game', 'Solve the puzzle related to the storyline of the book', NOW(), NOW()),
('Trivia Challenge', 'General trivia based on various books', NOW(), NOW());

INSERT INTO activity_book (activity_id, book_id)
VALUES
(1, 1),
(2, 1),
(3, 2);

INSERT INTO activity_images (activity_id, title, image_url, created_at, updated_at)
VALUES
```

```
(1, 'Quiz Image', 'http://example.com/quiz_image.png', NOW(), NOW()),
(2, 'Puzzle Image', 'http://example.com/puzzle_image.png', NOW(), NOW()),
(3, 'Trivia Image', 'http://example.com/trivia_image.png', NOW(), NOW());

INSERT INTO activity_book_user (activity_book_id, user_id, progress)
VALUES
(1, 1, 50),
(2, 2, 30),
(3, 3, 70);

INSERT INTO book_user_read (book_id, user_id, progress, rating, read_date)
VALUES
(1, 1, 100, 5, NOW()),
(2, 2, 60, 4, NOW()),
(3, 3, 20, 3, NOW());

DELIMITER $$
```

ANEXO C: Cria stored procedures com queries para listar todos os livros com regras de filtros, recebidos por parâmetros. (US_B003)

```
CREATE PROCEDURE ListBooksByTags(
   IN p_tags VARCHAR(255),      -- Parâmetro para lista de tags, separadas por vírgula
   IN p_is_active BOOLEAN,      -- Parâmetro opcional para filtrar por status do livro (ativo/inativo)
   IN p_age_group VARCHAR(50)   -- Parâmetro opcional para filtrar pelo grupo de idade
)
BEGIN
   -- Variáveis temporárias para manipulação das tags
   DECLARE tag_count INT DEFAULT 0;
   DECLARE tag_condition TEXT DEFAULT '';

   -- Contar a quantidade de tags recebidas no parâmetro p_tags
   SET tag_count = (LENGTH(p_tags) - LENGTH(REPLACE(p_tags, ',', '')) + 1);

   -- Gerar a condição de tags dinamicamente
   SET tag_condition = CONCAT(
      ' AND (SELECT COUNT(DISTINCT t.id) FROM tags t ',
      'JOIN tagging_tagged tt ON t.id = tt.tag_id ',
      'WHERE tt.book_id = b.id AND FIND_IN_SET(t.name, ?)) = ?'
   );

   -- Construção da query principal para seleção dos livros
   SET @query = CONCAT(
      'SELECT b.id, b.title, b.description, b.cover_url, b.read_time, b.age_group ',
```

```
        'FROM books b ',
        'JOIN tagging_tagged tt ON b.id = tt.book_id ',
        'JOIN tags t ON t.id = tt.tag_id ',
        'WHERE FIND_IN_SET(t.name, ?) > 0'
    );

    -- Adiciona condição para status do livro, se for passado o parâmetro p_is_active
    IF p_is_active IS NOT NULL THEN
        SET @query = CONCAT(@query, ' AND b.is_active = ', p_is_active);
    END IF;

    -- Adiciona condição para o grupo de idade, se for passado o parâmetro
p_age_group
    IF p_age_group IS NOT NULL THEN
        SET @query = CONCAT(@query, ' AND b.age_group = "', p_age_group, '"');
    END IF;

    -- Executa a query com os parâmetros dinâmicos
    SET @query = CONCAT(@query, ' GROUP BY b.id');

    PREPARE stmt FROM @query;
    SET @p_tags = p_tags;
    EXECUTE stmt USING @p_tags;
    DEALLOCATE PREPARE stmt;
END$$

DELIMITER ;

-- CALL ListBooksByTags('Mystery,History', NULL, NULL);
```

ANEXO D: Cria stored procedures com queries para listar as atividades de um dado livro, recebido por parâmetro. (US_B004)

```
CREATE PROCEDURE ListBookActivities(IN bookId INT)
BEGIN
   SELECT
      a.id AS activity_id,
      a.title AS activity_title,
      a.description AS activity_description,
      a.created_at AS activity_created_at,
      a.updated_at AS activity_updated_at
   FROM activity_book ab
   JOIN activities a ON ab.activity_id = a.id
   WHERE ab.book_id = bookId;
END$$

DELIMITER ;

-- CALL ListBookActivities(1);
```

ANEXO E: Cria stored procedures com queries para listar os livros favoritos e os livros lidos de um dado utilizador, com o respetivo progresso. (US_B005)

```
CREATE PROCEDURE ListUserFavouriteBooks(IN userId INT)
BEGIN
   SELECT
      b.id AS book_id,
      b.title,
      b.description,
      b.read_time,
      b.age_group,
      b.is_active,
      b.cover_url,
      u.first_name,
      u.last_name,
      u.email
   FROM book_user_favourite buf
   JOIN books b ON buf.book_id = b.id
   JOIN users u ON buf.user_id = u.id
   WHERE buf.user_id = userId;
END$$

DELIMITER ;



DELIMITER $$
CREATE PROCEDURE ListUserReadBooks(IN userId INT)
BEGIN
```

```
    SELECT
        b.id AS book_id,
        b.title,
        b.description,
        b.read_time,
        b.age_group,
        b.is_active,
        b.cover_url,
        bur.progress,
        bur.rating,
        bur.read_date,
        u.first_name,
        u.last_name,
        u.email
    FROM book_user_read bur
    JOIN books b ON bur.book_id = b.id
    JOIN users u ON bur.user_id = u.id
    WHERE bur.user_id = userId;
END$$

DELIMITER ;

-- CALL ListUserReadBooks(1);

DELIMITER $$
```

ANEXO F: Stored procedures com queries para mostrar livros sugeridos a um dado utilizador (US_B006)

```
CREATE PROCEDURE SuggestedBooksForUser(
   IN p_user_id INT
)
BEGIN
   -- Sugere livros que partilham tags com os livros favoritos do utilizador
   SELECT DISTINCT b.id AS book_id,
         b.title,
         b.description,
         b.cover_url,
         b.age_group,
         AVG(bur.rating) AS avg_rating,
         COUNT(bur.id) AS total_reads
   FROM books b
   JOIN tagging_tagged tt ON b.id = tt.book_id
   JOIN tags t ON tt.tag_id = t.id
   JOIN book_user_favourite buf ON buf.book_id = b.id
   LEFT JOIN book_user_read bur ON b.id = bur.book_id
   WHERE buf.user_id = p_user_id              -- Compara com favoritos do utilizador
    AND b.id NOT IN (                    -- Exclui livros já lidos pelo utilizador
       SELECT book_id
       FROM book_user_read
       WHERE user_id = p_user_id
    )
   GROUP BY b.id
```

```
   ORDER BY avg_rating DESC, total_reads DESC   -- Ordena pela média de avaliação e
leituras
   LIMIT 10;
         -- Limitar a 10 livros


   -- Sugere livros populares que o utilizador ainda não leu
   SELECT DISTINCT b.id AS book_id,
        b.title,
        b.description,
        b.cover_url,
        b.age_group,
        AVG(bur.rating) AS avg_rating,
        COUNT(bur.id) AS total_reads
   FROM books b
   LEFT JOIN book_user_read bur ON b.id = bur.book_id
   WHERE b.id NOT IN (                    -- Exclui livros já lidos pelo utilizador
       SELECT book_id
       FROM book_user_read
       WHERE user_id = p_user_id
    )
   GROUP BY b.id
   ORDER BY total_reads DESC, avg_rating DESC   -- Ordena por popularidade e média
de avaliação
   LIMIT 10;
         -- Limitar a 10 livros


   -- Sugere livros do mesmo grupo de idade favoritos por outros utilizadores
   SELECT DISTINCT b.id AS book_id,
        b.title,
        b.description,
        b.cover_url,
```

```
        b.age_group,
        AVG(bur.rating) AS avg_rating,
        COUNT(bur.id) AS total_reads
    FROM books b
    JOIN book_user_favourite buf ON buf.book_id = b.id
    LEFT JOIN book_user_read bur ON b.id = bur.book_id
    WHERE b.age_group = (                    -- Filtra pelo grupo de idade
        SELECT age_group
        FROM books
        WHERE id IN (SELECT book_id FROM book_user_favourite WHERE user_id =
p_user_id)
        LIMIT 1
    )
    AND b.id NOT IN (                    -- Exclui livros já lidos pelo utilizador
        SELECT book_id
        FROM book_user_read
        WHERE user_id = p_user_id
    )
    GROUP BY b.id
    ORDER BY avg_rating DESC, total_reads DESC    -- Ordena pela média de avaliação e
leituras totais
    LIMIT 10;
            -- Limitar a 10 livros


END$$


DELIMITER ;
-- CALL SuggestedBooksForUser(1);
```

ANEXO G: Cria a view para listar os livros mais populares nos últimos 3 meses (US_B007)

```
CREATE VIEW PopularBooksLast3Months AS
SELECT
   b.id AS book_id,
   b.title,
   b.description,
   b.cover_url,
   COUNT(bur.id) AS total_reads,                -- Total de vezes que o livro foi lido
   AVG(bur.rating) AS average_rating,           -- Média das avaliações
   SUM(bur.progress) / COUNT(bur.id) AS avg_progress -- Progresso médio das leituras pelos utilizadores
FROM
   books b
JOIN
   book_user_read bur ON b.id = bur.book_id
WHERE
   bur.read_date >= DATE_SUB(CURDATE(), INTERVAL 3 MONTH)  -- Últimos 3 meses
GROUP BY
   b.id, b.title, b.description, b.cover_url
ORDER BY
   total_reads DESC,         -- Ordena por total de leituras (popularidade)
   average_rating DESC,      -- Ordena por média de avaliação
   avg_progress DESC;        -- Ordena por progresso médio

-- SELECT * FROM PopularBooksLast3Months;
```

ANEXO H: Efetua o rating médio de um livro automaticamente (EXTRA)

```
CREATE TRIGGER update_book_rating
AFTER INSERT ON book_user_read
FOR EACH ROW
BEGIN
   DECLARE avg_rating DECIMAL(3, 2);

   -- Calcula a média de rating para o livro específico
   SELECT AVG(rating) INTO avg_rating
   FROM book_user_read
   WHERE book_id = NEW.book_id;

   -- Atualiza a coluna rating_medio na tabela books
   UPDATE books
   SET rating_medio = avg_rating
   WHERE id = NEW.book_id;
END;
$$

DELIMITER ;



DELIMITER $$

CREATE TRIGGER update_book_rating_on_update
AFTER UPDATE ON book_user_read
```

```
FOR EACH ROW
BEGIN
   DECLARE avg_rating DECIMAL(3, 2);

   -- Calcula a média de rating para o livro específico
   SELECT AVG(rating) INTO avg_rating
   FROM book_user_read
   WHERE book_id = NEW.book_id;

   -- Atualiza a coluna rating_medio na tabela books
   UPDATE books
   SET rating_medio = avg_rating
   WHERE id = NEW.book_id;
END;

DELIMITER ;
```