BookFlix

Lendo to the next level

Integrantes

Documentação

Clever Raimundo

Jonathan Magalhães

Matheus Costa

Design e Prototipação

Geovanna Lessa

Jezreel Anderson

Vitor Lima de Carvalho

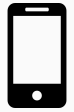
Programação

Daniel Borges

Geovanna Lessa

Marcela Donata

Sobre



Aplicação mobile

Roteiros de leitura

Roteiros de leitura

BookFlix é uma aplicação mobile que incentiva o hábito da leitura através do conceito ‘gameficado’ de recompensas.

O usuário constrói um roteiro de livro, com base em sugestões de obras, e define um prazo para seu consumo, ganhando pontos ao cumpri-los.

Problema

66% dos jovens brasileiros não leem textos com mais de 10 páginas

Programa Internacional de Avaliação de Estudantes - 2018

Objetivo

Este projeto vai ao encontro do Objetivo Global de Desenvolvimento Sustentável 4:

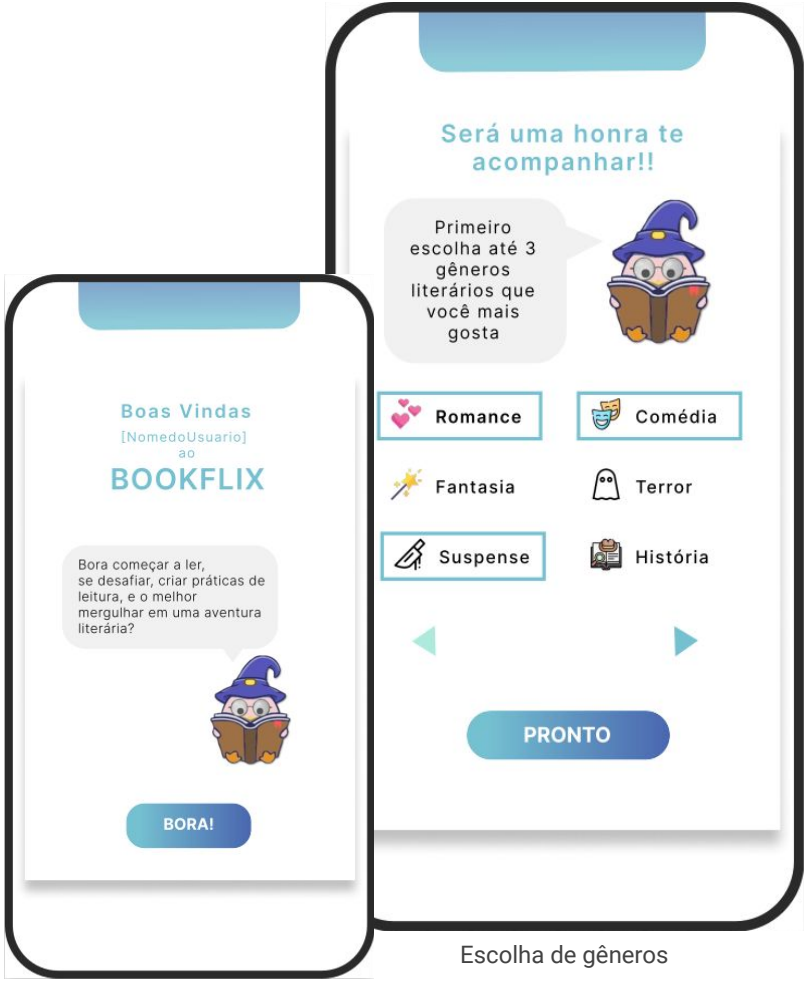
Promoção de oportunidades de aprendizagem ao longo da vida para todos

> *O hábito de leitura tem relação comprovada com a de saúde mental. A leitura, por envolver imaginação, mentalização, antecipação e aprendizagem (sempre aprendemos, ao menos, palavras novas), funciona como um 'exercício' para o cérebro humano.*
>
> Augusto Buchweitz

O app

Prototipação das telas

BookFlix apresenta ao usuário roteiros de leitura, sugerindo livros com base nas preferências por gêneros literários.



Escolha de gêneros

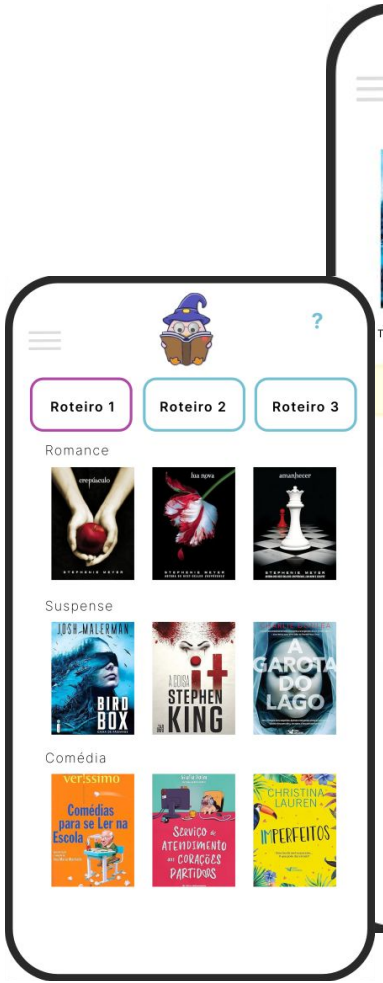
Tela de boas vindas

O app

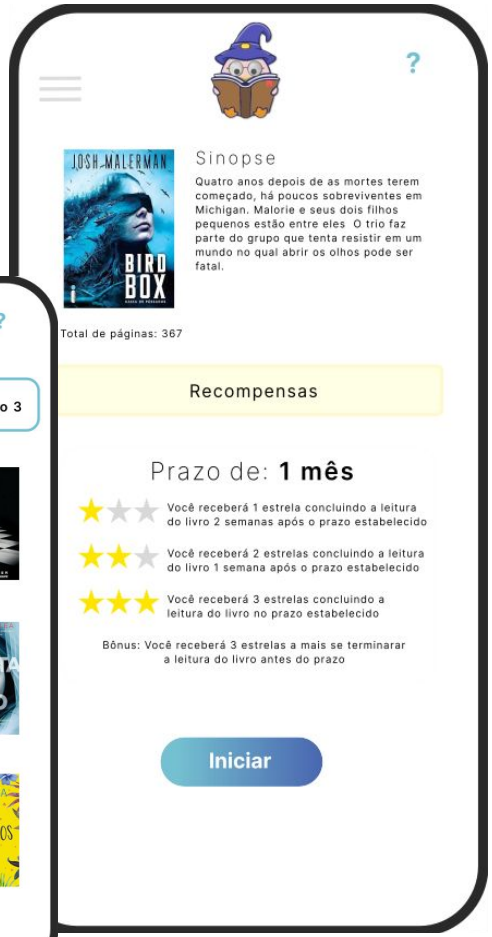
Prototipação das telas

Para o sistema de pontos, o usuário precisa concluir a leitura da obra em um determinado prazo.

Pontos são concedidos conforme o usuário cumpre o roteiro.



Escolha dos roteiros



Apresentação do prazo

O app

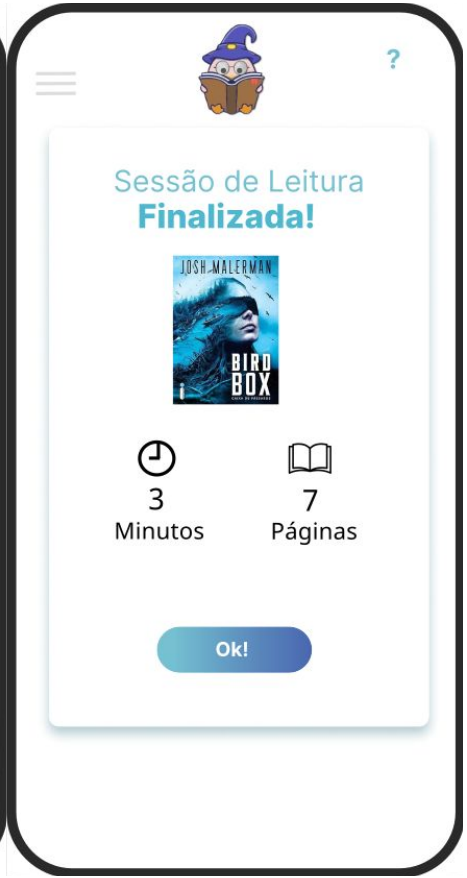
Prototipação das telas

Para que o usuário registre seu progresso, a aplicação oferece um controle temporizador (cronômetro).

Ao fim da sessão, o usuário informa a última página que leu.

No futuro pretende-se implementar um leitor de livros em formato digital, onde o controle será automatizado.

Controle de sessão de leitura



Fim da sessão de leitura

O app

Prototipação das telas

Conceitos de ‘gameficação’ serão utilizados no sistema de recompensas, onde o usuário recebe pontos por suas conquistas e avança de nível em determinados marcos.

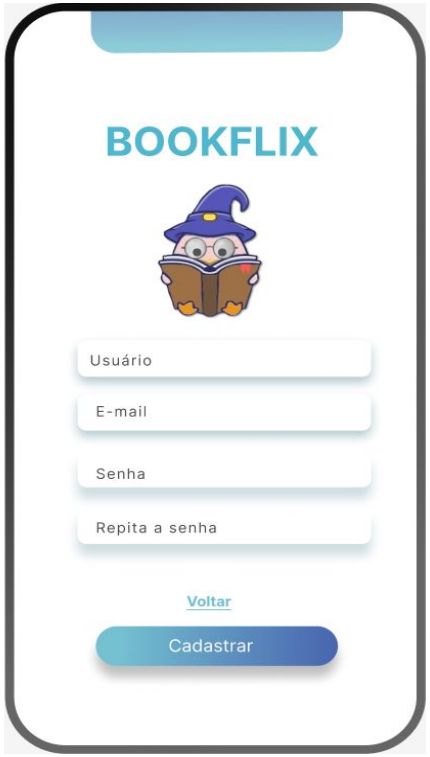
Página principal do usuário

O app

Prototipação das telas

Telas essenciais

Página de login



Página de cadastro

O app

Prototipação das telas

Telas essenciais

Menu lateral da Barra

Página para troca de senha

Exemplo de notificações

Arquitetura

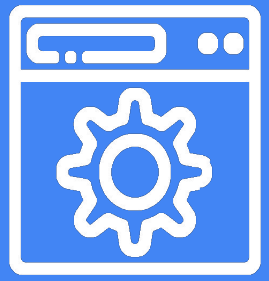
Banco de Dados

PostgreeSQL
Relacional

Front End

Flutter - Dart
Componentização

Back End



Node JS
Express

API

OpenLibrary
RESTful

Arquitetura

Banco de Dados

Postgree é um sistema de banco de dados relacional a objetos, padronizado ao SQL

Gratuito

Seguro

Extensível

```
TABLE usuario {
  id interger notnull nextval
  nome char varying(120) notnull
  email char varying(120) notnull
  senha char varying(120) notnull
  nivel interger default 0
  pontos interger default 0
  livro_atual char varying(13)
  data_criacao timestamp current
  generos text[ ]
}
Indexes:
  usuario_pkey (id)
  email_unico constraint (email)
```

Arquitetura

BackEnd

NodeJS é um runtime interpretador assíncrono para JavaScript

Gratuito

Performance

Microsserviços

Criar usuário

```
app.post('/usuario', async (req, res) => {
    const { nome, email, senha, nivel, pontos, livro_atual, data_criacao } = req.body;
  try {
   const usuarioExiste = await pool.query('SELECT * FROM usuario WHERE email = $1', [email]);
   if (usuarioExiste.rows.length > 0) {
   return res.status(400).json({ error: 'E-mail já está em uso' });
   }
  const senhaComHash = await hashSenha(senha);
   const result = await pool.query('INSERT INTO usuario (nome, email, senha, nivel, pontos, livro_atual, data_criacao) VALUES ($1, $2, $3, $4, $5, $6, $7) RETURNING *', [nome, email, senhaComHash, nivel, pontos, livro_atual, data_criacao]);
   res.json(result.rows[0]);
  } catch (err) {
   console.error('Erro ao criar usuário:', err);
   res.status(500).json({ error: 'Erro interno do servidor' });
  }});
```

Arquitetura

FrontEnd

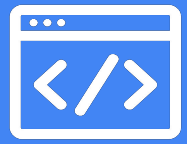
Flutter é uma framework multiplataforma para interfaces

Personalizável

Abrangente

Open Source

Renderizar atributos



```
@override
Widget build(BuildContext context) {
 final double largura = MediaQuery.of(context).size.width * 0.9;
final double altura = largura * 0.4; // proporcao da largura

    return Center(
      child: Container(
        width: largura,
        height: altura,
        padding: const EdgeInsets.all(16),
        decoration: BoxDecoration(
          color: Colors.white,
          borderRadius: BorderRadius.circular(10),
        ),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            buildResumoItem(Icons.emoji_events, nivel, 'Nível'),
            buildResumoItem(Icons.star, pontos, 'Pontos'),
            buildResumoItem(Icons.book, livros, 'Livros Lidos'),
], ), ), ); }
```

Arquitetura

API

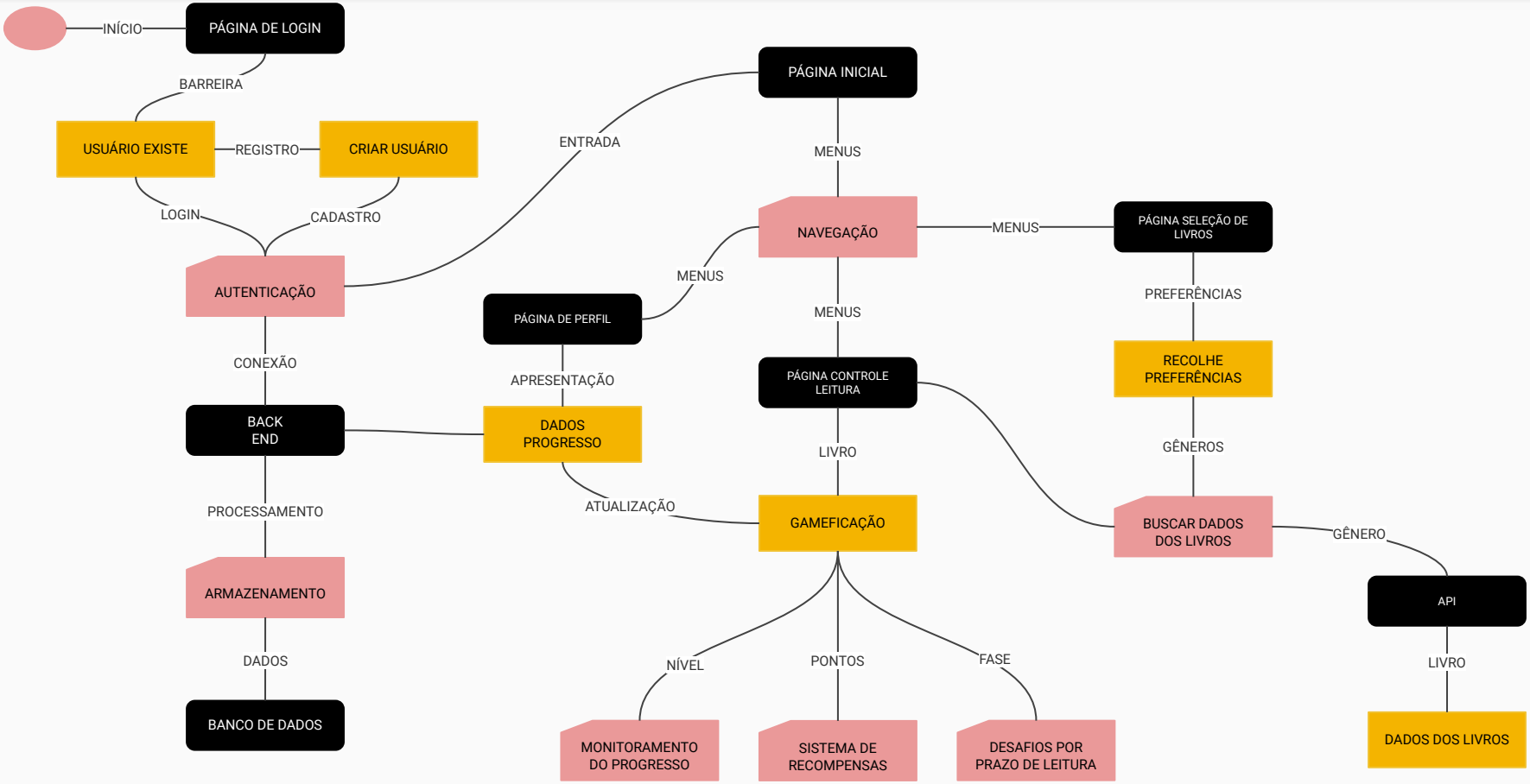
Open Library é um projeto de catalogação de ‘todos’ os livros através da internet.

São oferecidas API’s para recolher diversos dados de livros através de requisições RESTful

Requisição de dados

```
@Future<Livro> buscarPorGenero(String genero,
    List<String> livrosJaExibidos) async {

bool pesquisa = false;
  //livro não ter os dados essenciais? > próximo
  //limite requisicao em 7 para validacao
while (!pesquisa) {

   final response = await http.get(Uri.parse(
'https://openlibrary.org/search.json?subject=$genero&sort=rating&limit=7&fields=title,isbn,cover_i,author_name,publish_date'
 ));

   if (response.statusCode == 200) {
   final booksDataArray = json.decode(response.body)['docs'];

(...)
```

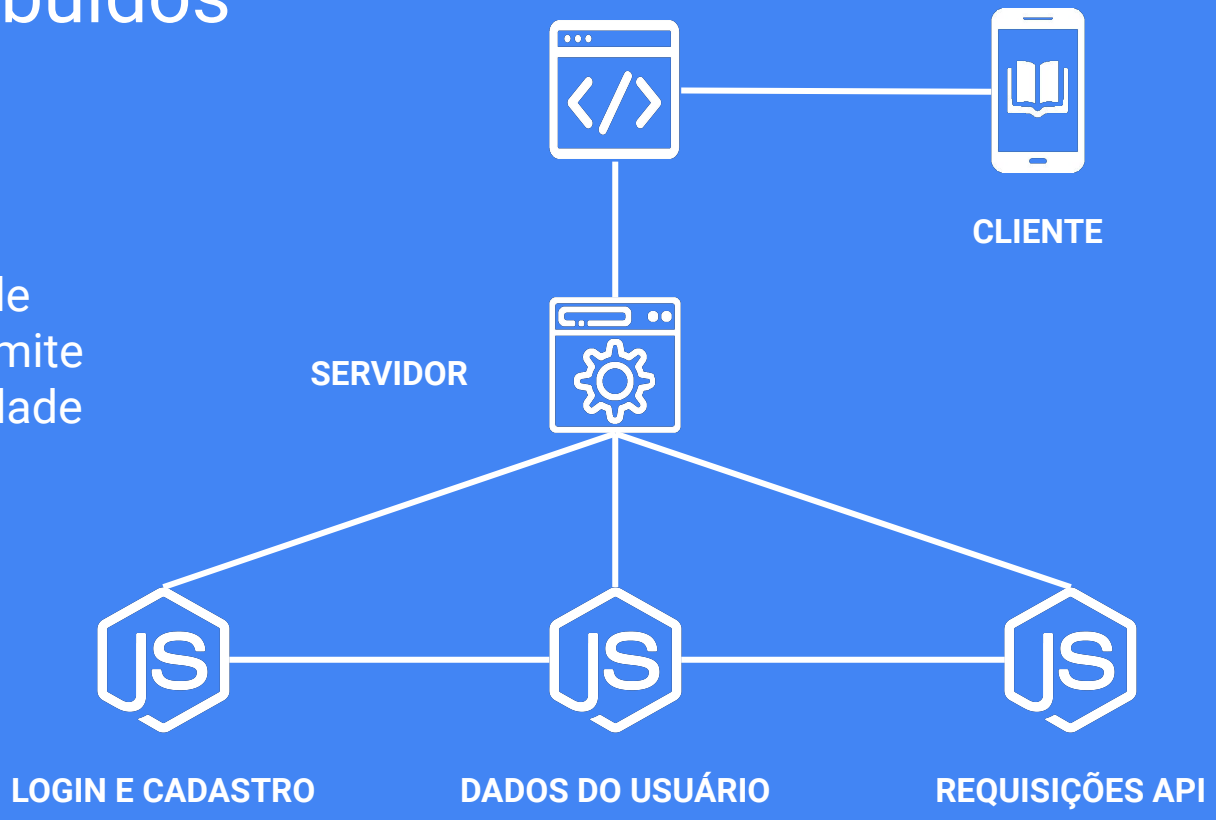
Fluxo da aplicação



Sistemas Distribuídos

Microsserviços

Dividir os processos de forma independente permite também uma escalabilidade e manutenibilidade independentes

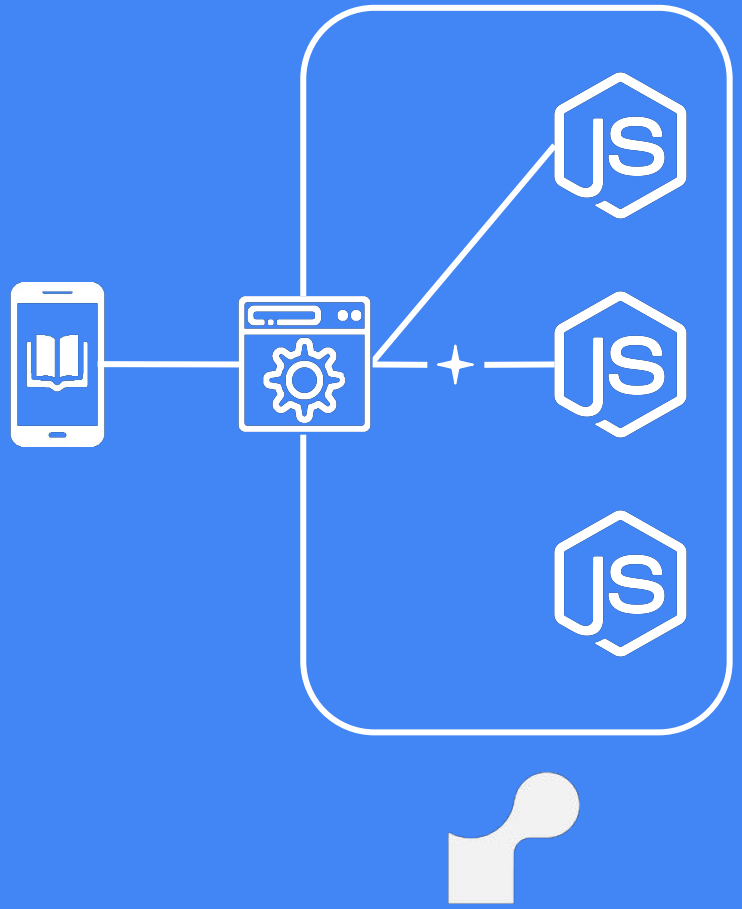


Sistemas Distribuídos

Balanceamento

Atualmente, para o desenvolvimento do protótipo, está sendo usada hospedagem de web service na plataforma Render.

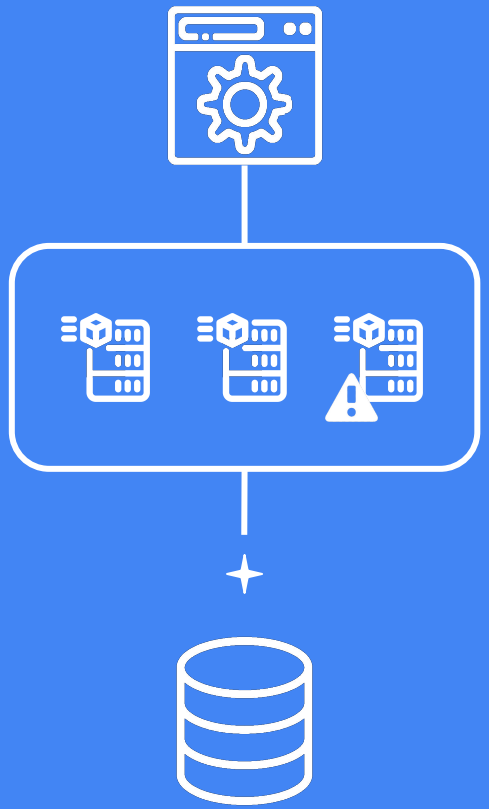
A plataforma já consta com processos de balanceamento de cargas entre instâncias, permitindo inicialmente uma escalabilidade horizontal



Sistemas Distribuídos

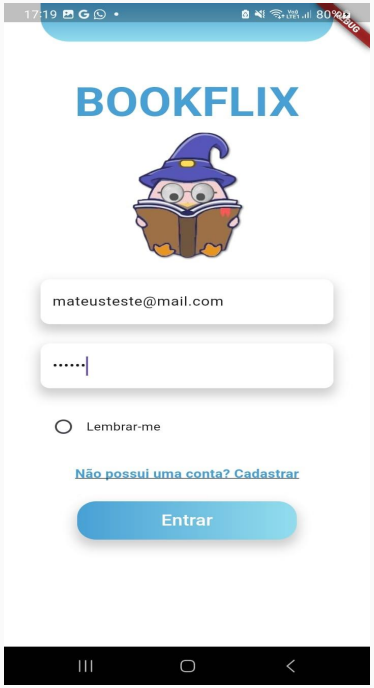
Transações

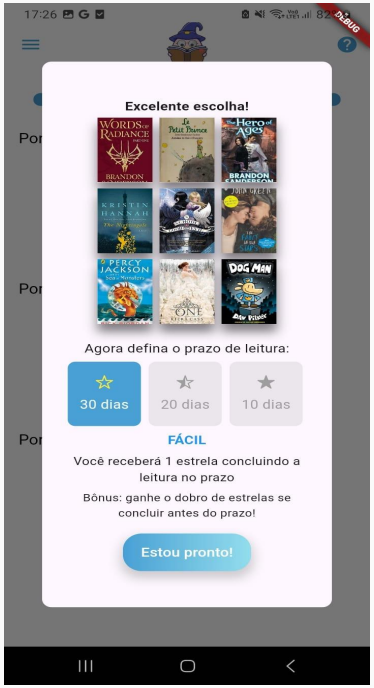
Serão usadas também transações atômicas no BackEnd, no contexto de operações no banco de dados, para garantir a integridade dos dados e evitar discrepâncias.



Protótipo

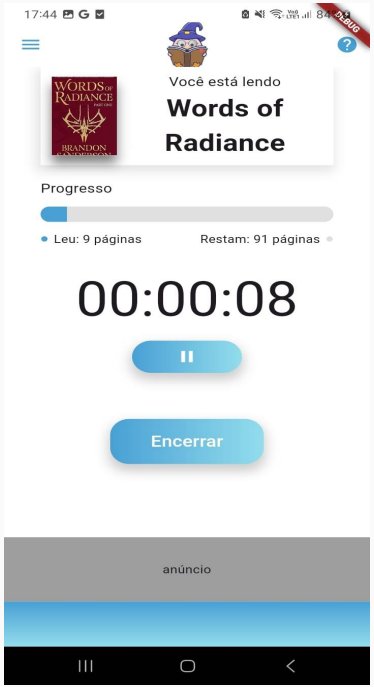
Telas renderizadas no aplicativo

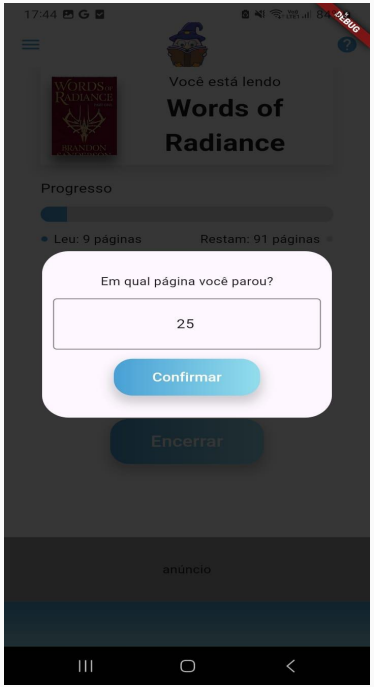


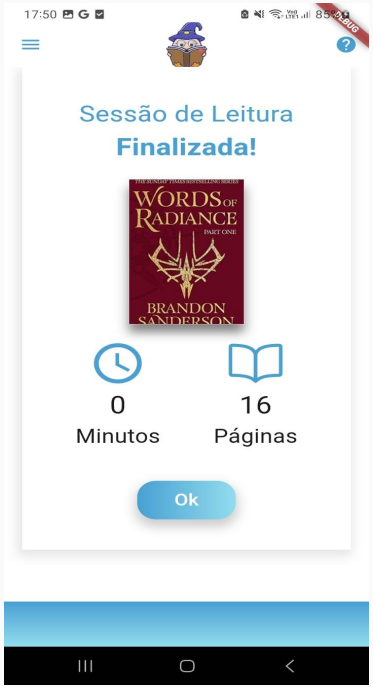


Protótipo

Telas renderizadas no aplicativo





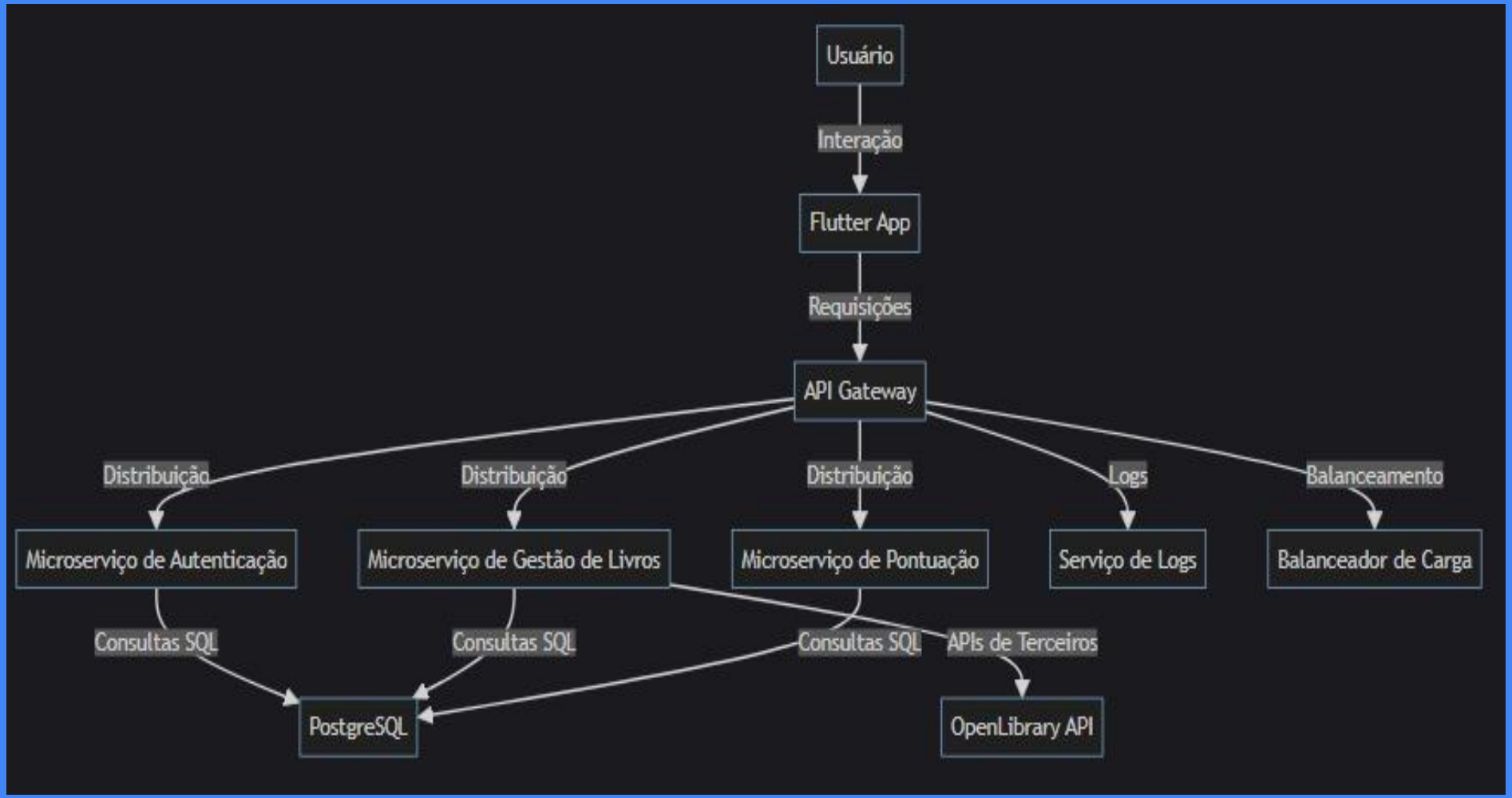


BookFlix

Lendo to the next level

Obrigado

Boa leitura!


Usuário
Interação
Flutter App
Requisições
API Gateway
Distribuição
Distribuição
Distribuição
Logs
Balanceamento
Microserviço de Autenticação
Microserviço de Gestão de Livros
Microserviço de Pontuação
Serviço de Logs
Balanceador de Carga
Consultas SQL
Consultas SQL
Consultas SQL
APIs de Terceiros
PostgreSQL
OpenLibrary API