# BOOKED, A LIVRARIA VIRTUAL QUE CONECTA LEITORES A AUTORES INDEPENDENTES:

IMPLEMENTAÇÃO DO FRONT-END MOBILE DO APLICATIVO BOOKED COM REACT NATIVE

Suelen Salvino da Silva

Orientadora: Liliane Alves do Nascimento Sales

**Resumo**

O presente trabalho tem como objetivo documentar o desenvolvimento do *front-end mobile* da aplicação Booked, uma plataforma voltada à interação entre leitores e autores independentes no ambiente digital. Diante do crescimento do consumo de conteúdos literários em dispositivos móveis e da expansão das comunidades virtuais de leitura, o projeto propôs uma solução tecnológica que integra leitura, interação e visibilidade em um único ambiente. O desenvolvimento foi realizado com React Native, TypeScript e Expo, adotando uma arquitetura em camadas com separação entre interface, lógica de estado, serviços e padronização visual. A gestão de estado global foi implementada por meio da *Context* API, associada a um fluxo de dados unidirecional, garantindo previsibilidade e organização estrutural. Também foram aplicados princípios de UX/UI, usabilidade e acessibilidade. O processo foi orientado pelo *framework* Scrum, permitindo a evolução iterativa e incremental das funcionalidades. Como resultado, foi desenvolvida uma aplicação funcional com múltiplas telas, integração com API e comunicação em tempo real, organizada em arquitetura modular e alinhada às boas práticas de desenvolvimento mobile.

**Palavras-chave:** *Front-end mobile*; React Native; Arquitetura em camadas; UX/UI; Comunidade literária digital.

**Abstract**

This study aims to document the development of the mobile front-end of the Booked application, a platform designed to foster interaction between readers and independent authors in the digital environment. Considering the growth of literary content consumption on mobile devices and the expansion of online reading communities, the project proposed a technological solution that integrates reading, interaction, and visibility within a single environment. The development was carried

out using React Native, TypeScript, and Expo, adopting a layered architecture with separation between interface, state management logic, services, and visual standardization. Global state management was implemented through the Context API, combined with a unidirectional data flow to ensure structural organization and predictability. UX/UI, usability, and accessibility principles were also applied. The process was guided by the Scrum *framework*, allowing the iterative and incremental evolution of functionalities. As a result, a functional mobile application was developed, featuring multiple screens, API integration, real-time communication, and a modular architecture aligned with best practices in mobile development.

**Keywords:** Mobile front-end; React Native; Layered architecture; UX/UI; Digital literary community.

## 1. Introdução

O crescimento do uso de dispositivos móveis impulsionou o desenvolvimento de aplicações mobile com foco em desempenho, usabilidade e integração com serviços em nuvem e isso mudou significativamente a forma como as pessoas consomem conteúdo e interagem no ambiente digital. Esse movimento também alcançou o universo da literatura, com a popularização da leitura digital e o fortalecimento de comunidades virtuais de leitores e autores independentes.

Nesse cenário de transformação, o mercado editorial brasileiro tem observado um crescimento constante do interesse pelo consumo de livros em formatos digitais. Segundo a Pesquisa Conteúdo Digital do Setor Editorial Brasileiro de 2024, realizada pelo Sindicato Nacional dos Editores de Livros (SNEL), as bibliotecas virtuais registraram um aumento de 59% na sua utilização em comparação ao ano de 2023, indicando não apenas a expansão do acesso aos conteúdos digitais, mas também uma mudança consistente nos hábitos de leitura.

Além disso, a pesquisa Retratos da Leitura no Brasil, conduzida pelo Instituto Pró-Livro, aponta que, em 2024, 75% dos leitores brasileiros utilizam o celular como principal dispositivo para leitura digital, um crescimento em relação aos 73% registrados em 2019. Esse comportamento reforça a presença do dispositivo móvel como ferramenta central na experiência de leitura contemporânea.

Como consequência, formam-se comunidades ativas de leitores nas redes sociais, enquanto autores independentes têm utilizado essas plataformas como meio

de divulgação de seus trabalhos, produzindo conteúdos, apresentando suas obras e alcançando novos públicos por meio de suas próprias redes. Esse processo favorece não apenas a ampliação da visibilidade desses autores, mas também a aproximação e a interação direta com seus leitores. Esse conjunto de fatores evidencia não apenas a evolução do comportamento do público leitor, mas também a necessidade de soluções tecnológicas que integrem leitura, interação e visibilidade.

Nesse contexto, foi proposta a aplicação Booked, com o objetivo de atender a essas demandas por meio de uma plataforma mobile voltada à interação entre leitores e autores independentes. Além disso, o front-end mobile foi desenvolvido de forma integrada a um *back-end* já existente, por meio de APIs (*Application Programming Interface*) responsáveis pela comunicação de dados, autenticação e gerenciamento das funcionalidades da aplicação.

A partir disso, este trabalho tem como objetivo documentar a implementação do *front-end mobile* do aplicativo Booked. Os objetivos específicos são:

- Descrever as tecnologias e a arquitetura adotadas no desenvolvimento do *front-end mobile*;
- Apresentar a estrutura e a organização do código da aplicação;
- Documentar as decisões de UX/UI, usabilidade e identidade visual aplicadas ao projeto.

O desenvolvimento deste relatório está estruturado da seguinte forma: inicialmente são apresentados os conceitos fundamentais relacionados ao front-end e à experiência do usuário. Em seguida, são descritas as tecnologias utilizadas, a arquitetura do projeto e a organização do código do front-end mobile do Booked. Por fim, são abordados os aspectos relacionados à usabilidade, identidade visual e funcionalidades implementadas na aplicação.

## 2. Conceito de front-end

De acordo com Souto (2019), “o *front-end* compreende a parte visual de sites e aplicações, ou seja, a área das páginas em que as pessoas podem interagir”. Dessa forma, o *front-end* trata-se de tudo aquilo com que o usuário vê, interage e utiliza diretamente em um sistema, seja ele um site, um aplicativo web ou um

aplicativo *mobile*, como é o caso deste projeto. Em essência, o *front-end* estabelece a ponte entre o *design* e a lógica de negócio, transformando *layouts*, protótipos e fluxos de navegação em telas funcionais e interativas.

No desenvolvimento *front-end*, o principal foco é criar experiências que sejam intuitivas, responsivas, acessíveis e visualmente consistentes. Para isso, o *front-end* lida com aspectos como a navegação, apresentação de conteúdos, animações, formulários, componentes interativos e a comunicação com o *back-end* por meio de APIs.

Essa camada é essencial para garantir que o usuário final tenha uma interação fluida, agradável e eficiente com a aplicação, alinhando estética, usabilidade e desempenho.

## 3. Arquitetura do Projeto

Esta seção descreve como o *front-end mobile* do Booked foi estruturado, quais tecnologias foram utilizadas, como ocorre a comunicação com o *back-end* e qual modelo arquitetural orientou a organização das camadas do aplicativo.

### 3.1 Tecnologias Utilizadas

O Booked é uma aplicação *mobile* desenvolvida com base na linguagem TypeScript. O TypeScript é um superconjunto do JavaScript que adiciona um sistema de tipos estáticos, permitindo maior segurança, previsibilidade e robustez durante o desenvolvimento (MICROSOFT, 2025). A escolha dessa linguagem se deu pela sua capacidade de reduzir erros em tempo de compilação e facilitar a manutenção do código ao longo do ciclo de vida da aplicação.

Para facilitar o processo de implementação e permitir o desenvolvimento de aplicações *cross-platform* com componentes nativos em ambos os sistemas operacionais, optou-se pela utilização do *framework* React Native. O React Native é um *framework* de código aberto criado pela Meta (antigo Facebook) em 2015, que possibilita a construção de aplicativos móveis nativos utilizando JavaScript ou TypeScript, a partir de uma única base de código (META, 2015).

No desenvolvimento do Booked, o React Native serviu como base estrutural de toda a interface *mobile*, permitindo criar telas responsivas, fluidas e alinhadas ao *design* proposto para a aplicação. Essa tecnologia também favoreceu a reutilização

de componentes, a integração com APIs e a adoção de boas práticas de engenharia de *software* voltadas para o ecossistema *mobile*.

Além disso, utilizou-se o Expo como ferramenta de apoio ao desenvolvimento, por meio de um conjunto de ferramentas e bibliotecas que aceleram o desenvolvimento com React Native, simplificando configurações nativas, testes e distribuição (EXPO, 2025). Com isso, foi possível reduzir a complexidade associada ao desenvolvimento nativo, além de facilitar a geração de *builds* para múltiplas plataformas e concentrar esforços na implementação das funcionalidades da aplicação.

### 3.2 Estrutura e Organização do Código

A estrutura do *front-end mobile* do Booked foi organizada de forma modular, priorizando a separação de responsabilidades, facilidade de manutenção e escalabilidade do projeto. O código foi distribuído em pastas específicas, cada uma com função bem definida  dentro da arquitetura da aplicação.

- *App*: concentra as principais telas e rotas do aplicativo, sendo responsável pela organização do fluxo de navegação e pela composição das interfaces apresentadas ao usuário.
- *Components*: reúne componentes visuais e funcionais reutilizáveis, compartilhados entre diferentes telas, como botões, *cards* e contadores de listas, promovendo padronização e reutilização de código.
- *Hooks*: agrupa *hooks* personalizados desenvolvidos para encapsular regras de negócio e lógicas de acesso a contextos e serviços, favorecendo a organização e a reutilização da lógica da aplicação.
- *Contexts*: responsável pela gestão de estado global da aplicação por meio da *Context* API do React, permitindo o compartilhamento de dados entre múltiplos componentes e telas de forma centralizada. A *Context* API foi escolhida por atender de forma adequada às necessidades do projeto, que exige o compartilhamento de informações como autenticação, navegação e dados de uso comum entre diferentes partes da aplicação. Essa abordagem elimina o repasse excessivo de propriedades (*prop drilling*), simplifica a estrutura

do código e reduz a dependência de bibliotecas externas de gerenciamento de estado.

- *Services*: camada destinada à comunicação com o *back-end*, concentrando as chamadas às APIs, o tratamento de respostas, a autenticação e o gerenciamento de erros.
- *Utils*: contém funções auxiliares utilizadas em tarefas recorrentes, como formatação de dados, validações, aplicação de máscaras de entrada e outras rotinas compartilhadas.
- *Theme:* centraliza as definições visuais da aplicação, como cores, tipografia, espaçamentos e temas, garantindo consistência visual e facilitando ajustes globais de estilo.

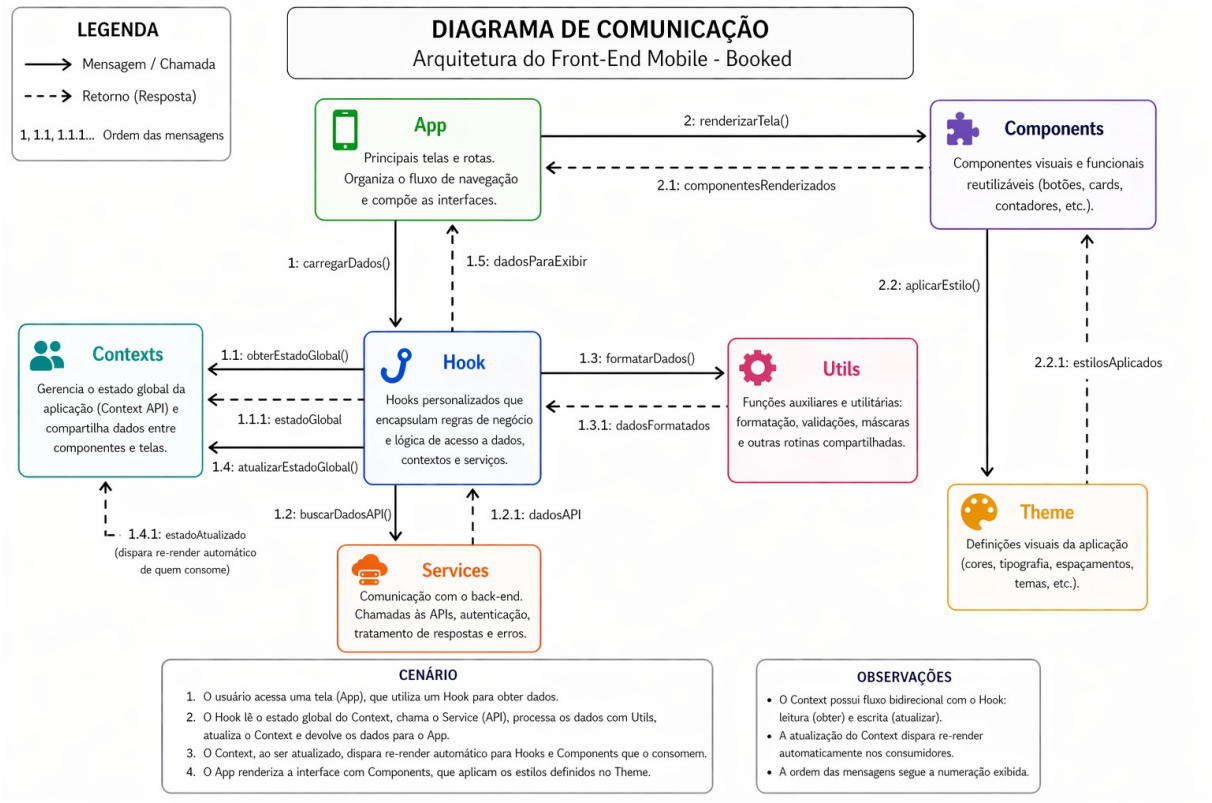


Figura A: Diagrama de Comunicação | *front-end mobile* Booked

A Figura A apresenta o diagrama de comunicação da aplicação, ilustrando a interação entre as principais camadas do *front-end*. A comunicação entre os componentes ocorre por meio do modelo de composição do React, associado ao uso de contextos globais e *hooks* personalizados.

O fluxo de dados adotado é unidirecional, iniciando-se nas camadas de serviços (*services*), responsáveis pela comunicação com a API e pelo acesso aos dados, passando pela camada de lógica de estado (*contexts e hooks*), onde as informações são processadas, armazenadas e disponibilizadas, até chegar aos componentes visuais (*app e components*), que consomem esses dados exclusivamente para renderização da interface. As interações do usuário, por sua vez, disparam eventos que retornam às camadas de estado por meio de funções controladas, sem alterar diretamente os dados, reforçando a previsibilidade e o controle do fluxo da aplicação.

O modelo arquitetural adotado no Booked segue o padrão de arquitetura em camadas, segmentando a aplicação em interface (*app e components*), lógica de estado (*contexts e hooks*), serviços (*services*), utilitários (*utils*) e padronização visual (*theme*). Essa separação de responsabilidades é evidenciada no diagrama de comunicação, que permite compreender de forma visual como cada camada se relaciona e quais são seus limites de atuação.

Além de representar a estrutura do sistema, o diagrama de comunicação desempenha um papel fundamental na manutenção e escalabilidade do projeto. Ao documentar explicitamente o fluxo de dados e as dependências entre camadas, o diagrama facilita a identificação de impactos decorrentes de alterações, reduzindo o risco de efeitos colaterais indesejados e otimizando o processo de correção e evolução do código. Em termos de escalabilidade, essa representação auxilia no planejamento da adição de novas funcionalidades, contextos ou serviços, garantindo que o crescimento da aplicação ocorra de forma organizada e alinhada às boas práticas de desenvolvimento de *front-end mobile*. O uso do diagrama de comunicação contribui para a compreensão arquitetural, a padronização das decisões técnicas e a sustentabilidade do sistema ao longo de seu ciclo de vida.

### 3.3 UX/UI, Usabilidade e Identidade Visual

O desenvolvimento da interface do Booked foi orientado por princípios de UX (*User Experience*) e UI (*User Interface*), com foco em oferecer uma experiência intuitiva, visualmente consistente e adequada ao uso em dispositivos móveis. As decisões de *design* buscaram reduzir a complexidade das interações e facilitar a navegação, considerando o perfil dos usuários e o contexto de consumo de conteúdo literário em ambiente digital.

No que se refere à usabilidade, o aplicativo adota uma estrutura de navegação simples e organizada, com fluxos bem definidos e componentes padronizados. Elementos como botões, *cards* e listas seguem um padrão visual consistente, contribuindo para a previsibilidade da interface e para a facilidade de aprendizado por parte do usuário. A hierarquia visual foi definida de forma a destacar ações principais e informações relevantes em cada tela.

Em relação à acessibilidade, foram consideradas boas práticas aplicáveis ao contexto *mobile*, como contraste adequado entre cores, tamanhos de fonte legíveis e espaçamentos consistentes entre os elementos interativos, visando facilitar a leitura e a interação do usuário. Essas diretrizes estão alinhadas às recomendações das *Web Content Accessibility Guidelines* (WCAG), que estabelecem princípios para tornar interfaces digitais mais perceptíveis, operáveis e compreensíveis em diferentes dispositivos (W3C, 2018).

A responsividade foi tratada como um requisito essencial, uma vez que o aplicativo é utilizado em dispositivos com diferentes tamanhos e resoluções de tela. O uso do React Native possibilitou a construção de *layouts* flexíveis, garantindo a adaptação da interface e a manutenção da consistência visual tanto em dispositivos Android quanto iOS.

A identidade visual do Booked foi construída com base em uma paleta de cores que remete ao universo literário e à proposta de interação social da aplicação. As cores de ênfase são compostas por variações de tons de roxo, frequentemente associadas à criatividade, sofisticação e inovação no design visual (OCTET DESIGN, 2024). Já as cores base, em tons de cinza e preto, são empregadas para equilibrar a interface e garantir legibilidade. Essa combinação permite destacar ações importantes e manter uma leitura confortável do conteúdo.

A tipografia adotada no projeto foi a Montserrat, uma fonte sem serifa amplamente utilizada em interfaces digitais. Fontes *sans-serif* são associadas a um visual moderno e apresentam melhor desempenho em telas, sendo recomendadas para aplicações digitais devido à sua clareza e legibilidade (ADOBE, 2024).

A tipografia adotada no projeto foi a Montserrat, escolhida por sua boa legibilidade em telas digitais e por seu aspecto moderno e neutro, adequado a aplicações *mobile*. O uso consistente da tipografia, aliado à padronização de cores e espaçamentos, contribui para a coerência visual e para a identidade da aplicação como um todo.

Para o planejamento e prototipação das interfaces, foi utilizada a ferramenta Figma, possibilitando a definição da identidade visual, criação dos *layouts* das telas e validação dos fluxos de navegação antes da implementação no código.

### 3.4 Funcionalidades implementadas

O desenvolvimento das funcionalidades do *front-end mobile* da aplicação Booked foi orientado por histórias de usuário registradas como *issues* no repositório do projeto no GitHub. O método de desenvolvimento adotado foi o Scrum, um *framework* ágil que visa organizar e gerenciar o desenvolvimento de produtos complexos por meio de ciclos iterativos e incrementais denominados *sprints*. Cada funcionalidade implementada está associada a uma *issue* específica, permitindo rastreabilidade entre os requisitos definidos, as *sprints* executadas e as entregas realizadas. Os detalhes das funcionalidades estão presentes no Apêndice A.

| **Funcionalidade Implementada** | **Issue (GitHub)** | **Sprint** |
|---|---|---|
| Aquisição e compra de livros | #39 – Aquisição e compra de livros | *Sprint* 6 |
| Visualização do histórico de compras | #36 – Visualização de compra do livro | *Sprint* 6 |
| Visualização do histórico de compras | #32 – Visualização de compra do livro | *Sprint* 6 |
| Arquitetura de comunicação em tempo real (*Sockets*) | #44 – [Socket.io] Arquitetura orientada a *socket* | *Sprint* 8 |
| Leitura de livros na plataforma | #92 – Leitura dos livros | *Sprint* 9 |

| | | |
|---|---|---|
| Documentação de componentes da interface | #43 – [*Storybook*] Documentação de componentes | *Sprint* 9 |
| Lista de desejos | #48 – Lista de desejos | *Sprint* 11 |
| Solicitação e pagamento de livros | #50 – Solicitação e pagamento de livros | *Sprint* 11 |
| Comunicação privada entre usuários (Chat) | #45 – [Chat] Comunicação entre usuários | *Sprint* 11 |
| Seguidores entre usuários | #51 – Seguidores entre usuários | *Sprint* 12 |
| Comunicação em grupo (Chat em grupo) | #56 – [Chat] Comunicação em grupo | *Sprint* 12 |

**Fonte:** Elaborado pela autora com base nas *issues* registradas no repositório GitHub do projeto Booked (2023).

## 4. Considerações finais

A princípio, o desenvolvimento do Booked representou um desafio significativo, por se tratar do primeiro contato com o desenvolvimento mobile e com a *stack* React Native. Embora a familiaridade prévia com o React — biblioteca JavaScript na qual o React Native é baseado — tenha contribuído para facilitar parte do processo, foi necessário compreender particularidades do ambiente *mobile*, como adaptação de *layout*, gerenciamento de estado e otimização de desempenho.

O desenvolvimento do *front-end mobile* do Booked buscou oferecer uma interface acessível, organizada e intuitiva, capaz de aproximar leitores e escritores por meio de uma experiência digital consistente. Sob o ponto de vista técnico, foram utilizadas tecnologias como React Native, TypeScript e Expo, aliadas à organização

arquitetural em camadas, ao uso da *Context* API para gestão de estado global e à definição de um fluxo de dados unidirecional. Tais decisões possibilitaram a construção de uma aplicação modular e escalável, estruturada conforme boas práticas de desenvolvimento mobile e orientada à manutenção, evolução contínua e organização eficiente do código.

Além disso, princípios de UX/UI, usabilidade, acessibilidade e identidade visual foram considerados na elaboração das interfaces, garantindo coerência estética, consistência visual e adequação ao contexto de uso em dispositivos móveis. A utilização de um diagrama de comunicação e a aplicação do *framework* Scrum também contribuíram para estruturar o desenvolvimento e facilitar a manutenção e evolução do projeto.

Como perspectivas futuras, pretende-se ampliar as funcionalidades da interface, incluindo recursos de moderação e edição de conteúdos, melhorias na experiência de leitura e otimizações de desempenho. Dessa forma, espera-se que o Booked continue evoluindo como uma solução tecnológica voltada ao fortalecimento da comunidade literária, cumprindo o propósito de apoiar autores independentes e ampliar o acesso à leitura no ambiente digital.

**Referências**


**ADOBE.** ***Serif vs Sans Serif Fonts*****. 2024.** Disponível em: https://www.adobe.com/creativecloud/design/discover/serif-vs-sans-serif.html. Acesso em: 16 abr. 2026.

**BOOKED. Protótipo de interface da aplicação Booked.** Disponível em: https://www.figma.com/design/7cF9H8JHQjQHBSNCbmkv8m/Booked?node-id=1-4&t=AysG8YI70urQQ0iU-1. Acesso em: 15 dez. 2025.

**CONTEÚDO DIGITAL DO SETOR EDITORIAL BRASILEIRO.** Disponível em: https://cbl.org.br/wp-content/uploads/2025/05/Conteudo_Digital_anobase_2024_imprensa.pdf. Acesso em: 7 nov. 2025.

**EXPO. Expo Documentation.** Disponível em: https://docs.expo.dev/. Acesso em: 27 nov. 2025.

**FIGMA. Figma – online interface design tool. 2024.** Disponível em: https://www.figma.com/. Acesso em: 15 dez. 2025.

**IPL. 75% dos brasileiros preferem ler livros pelo celular.** Disponível em: https://www.prolivro.org.br/2025/01/09/75-dos-brasileiros-preferem-ler-livros-pelo-celular/. Acesso em: 5 nov. 2025.

**META. React Native Documentation.** Disponível em: https://reactnative.dev/. Acesso em: 27 nov. 2025.

**MICROSOFT. TypeScript Documentation.** Disponível em: https://www.typescriptlang.org/docs/. Acesso em: 27 nov. 2025.

**OCTET DESIGN.** ***Purple Color Meaning*****. 2024.** Disponível em: https://octet.design/journal/purple-color-meaning/. Acesso em: 16 abr. 2026.

**SILVA, Ian. Booked: livraria virtual para autores independentes.** GitHub, 2025. Disponível em: https://github.com/i4n-v/booked. Acesso em: 15 dez. 2025.

**SOCKET.IO. Socket.IO: Bidirectional and low-latency communication for every platform. 2024.** Disponível em: https://socket.io/. Acesso em: 15 dez. 2025.

**SOUTO, M. O que é Front-End, Back-End e Full Stack? Aprenda as diferenças entre essas áreas. Alura, 24 set. 2019.** Disponível em: https://www.alura.com.br/artigos/o-que-e-front-end-e-back-end. Acesso em: 5 fev. 2026.

**WORLD WIDE WEB CONSORTIUM (W3C). Web Content Accessibility Guidelines (WCAG) 2.1. 2018.** Disponível em: https://www.w3.org/TR/WCAG21/. Acesso em: 28 jan. 2026.

# APÊNDICE A – DESCRIÇÃO DAS FUNCIONALIDADES IMPLEMENTADAS

## Épico: Aquisição e compra de livros

***Issue*** **associada: #39**

***Sprint*****: 6**

**Descrição:**

Refere-se ao conjunto de funcionalidades que permitem ao leitor visualizar, adquirir e acessar livros disponíveis na plataforma, compondo o fluxo de comercialização de conteúdos digitais do Booked.

### História de Usuário 1 – Visualização de livro

**História de Usuário:**

Como leitor, desejo visualizar os detalhes de um livro, para que eu possa decidir se desejo adquiri-lo.

**Critérios de Aceitação:**

- O usuário deve visualizar título, autor, descrição e preço;
- O sistema deve exibir a capa do livro.

### História de Usuário 2 – Início da compra

**História de Usuário:**

Como leitor, desejo iniciar a compra de um livro, para que eu possa prosseguir com a aquisição.

**Critérios de Aceitação:**

- O usuário deve conseguir iniciar o processo de compra;
- O sistema deve registrar a solicitação de compra.

### História de Usuário 3 – Confirmação da compra

**História de Usuário:**
Como leitor, desejo confirmar a compra de um livro, para que eu possa acessá-lo posteriormente.

**Critérios de Aceitação:**

- O sistema deve confirmar a compra;
- O sistema deve atualizar o status da transação.

### História de Usuário 4 – Acesso ao livro adquirido

**História de Usuário:**
Como leitor, desejo acessar os livros adquiridos, para que eu possa lê-los na plataforma.

**Critérios de Aceitação:**

- O livro deve estar disponível na biblioteca do usuário;
- O acesso deve ser liberado após a confirmação da compra.

---

**Visualização do histórico de compras**

***Issues* associadas:** #36, #32
***Sprint*:** 6

**Descrição:**
Permite ao usuário consultar todas as compras realizadas, incluindo informações relevantes das transações.

**História de Usuário:**
Como leitor, desejo visualizar meu histórico de compras, para que eu possa acompanhar minhas aquisições e transações.

**Critérios de Aceitação:**

- O sistema deve listar todas as compras realizadas;

- Cada item deve exibir informações do livro e da compra;
- O usuário deve visualizar o status da compra;
- O histórico deve estar disponível a qualquer momento.

---

**Leitura de livros na plataforma**

***Issue*** **associada:** #92

***Sprint*****:** 9

**Descrição:**

Funcionalidade que permite ao usuário acessar e ler livros adquiridos diretamente no aplicativo.

**História de Usuário:**

Como leitor, desejo ler livros diretamente na plataforma, para que eu possa acessar meus conteúdos de forma prática no dispositivo móvel.

**Critérios de Aceitação:**

- O usuário deve acessar livros previamente adquiridos;
- O conteúdo deve ser exibido de forma legível;
- A navegação entre páginas deve ser fluida;
- A interface deve ser responsiva para dispositivos móveis.

---

**Lista de desejos**

***Issue*** **associada:** #48

***Sprint*****:** 11

**Descrição:**

Permite ao usuário salvar livros de interesse para aquisição futura.

**História de Usuário:**
Como leitor, desejo adicionar livros a uma lista de desejos, para que eu possa organizá-los para compras futuras.

**Critérios de Aceitação:**

- O usuário deve adicionar livros à lista;
- O sistema deve armazenar os livros selecionados;
- O usuário deve visualizar a lista de desejos;
- O usuário deve remover livros da lista.

---

**Solicitação e pagamento de livros**

***Issue* associada:** #50
***Sprint*:** 11

**Descrição:**
Fluxo que permite ao leitor solicitar a compra de um livro e ao autor gerenciar essa solicitação.

**História de Usuário:**
Como leitor, desejo solicitar a compra de um livro, para que o autor possa aprovar ou rejeitar minha solicitação.

**Critérios de Aceitação:**

- O usuário deve enviar solicitação de compra;
- O autor deve visualizar solicitações recebidas;
- O autor deve aceitar ou rejeitar a solicitação;
- O sistema deve atualizar o status da solicitação;
- Deve existir comunicação entre leitor e autor durante o processo.

---

**Comunicação privada entre usuários (chat)**

***Issue* associada:** #45

***Sprint*:** 11

**Descrição:**

Permite troca de mensagens diretas entre usuários em tempo real.

**História de Usuário:**

Como usuário, desejo trocar mensagens privadas com outros usuários, para que eu possa me comunicar diretamente dentro da plataforma.

**Critérios de Aceitação:**

- O usuário deve enviar mensagens;
- O destinatário deve receber mensagens em tempo real;
- O sistema deve armazenar o histórico;
- O chat deve exibir mensagens em ordem cronológica.

---

**Comunicação em grupo (chat em grupo)**

***Issue* associada:** #56

***Sprint*:** 12

**Descrição:**

Permite a criação de conversas com múltiplos participantes.

**História de Usuário:**

Como usuário, desejo participar de conversas em grupo, para que eu possa interagir com múltiplos usuários simultaneamente.

**Critérios de Aceitação:**

- O usuário deve participar de grupos;
- O sistema deve permitir envio de mensagens no grupo;
- Todos os participantes devem visualizar as mensagens;
- O histórico deve ser armazenado.

---

**Seguidores entre usuários**

***Issue*** **associada:** #51

***Sprint*****:** 12

**Descrição:**

Permite que usuários sigam outros perfis na plataforma.

**História de Usuário:**

Como usuário, desejo seguir outros usuários, para que eu possa acompanhar suas atividades.

**Critérios de Aceitação:**

- O usuário deve seguir outros perfis;
- O sistema deve registrar seguidores;
- O usuário deve visualizar quem segue;
- O usuário deve deixar de seguir quando desejar.

---

**Arquitetura de comunicação em tempo real**

***Issue*** **associada:** #44

***Sprint*****:** 8

**Descrição:**

Foi adotada uma arquitetura baseada em comunicação em tempo real utilizando *sockets***.**

**História de Usuário (técnica):**

Como sistema, desejo garantir comunicação em tempo real, para que as interações ocorram instantaneamente.

**Critérios de Aceitação:**

- O sistema deve estabelecer conexão via *socket*;
- Eventos devem ser enviados e recebidos em tempo real;
- O sistema deve tratar reconexões;
- A comunicação deve suportar múltiplos usuários simultâneos.

---

**Documentação de componentes da interface**

***Issue* associada:** #43
***Sprint*:** 9

**Descrição:**
Documentação dos componentes utilizando *Storybook*.

**História de Usuário (técnica):**
Como desenvolvedor, desejo documentar os componentes da interface, para que eu possa facilitar manutenção e reutilização.

**Critérios de Aceitação:**

- Os componentes devem estar documentados no *Storybook*;
- Cada componente deve possuir variações documentadas;
- A documentação deve ser acessível;
- Deve existir padronização visual.

---

## Síntese da rastreabilidade

A associação entre funcionalidades implementadas e *issues* do GitHub evidencia a adoção de um processo de desenvolvimento organizado, incremental e rastreável. Essa abordagem fortalece a confiabilidade técnica do projeto, permitindo a verificação das entregas realizadas e sua correspondência com os requisitos inicialmente definidos.