UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS FLORIANÓPOLIS
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Lucas Pereira Zarbato

Implementação de uma Biblioteca Virtual com Progressive Web Application

Florianópolis, Santa Catarina - Brasil
2023

Lucas Pereira Zarbato

Implementação de uma Biblioteca Virtual com Progressive Web Application

Trabalho de Conclusão de Curso do Curso de Graduação em Ciência da Computação do Campus Florianópolis da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Ciências da Computação

Orientador: Prof. Ricardo Pereira e Silva, Dr.

Florianópolis, Santa Catarina -Brasil

2023

Lucas Pereira Zarbato

Implementação de uma Biblioteca Virtual com Progressive Web Application

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de “Bacharel em Ciências da Computação” e aprovado em sua forma final pelo Curso de Graduação em Ciência da Computação.

Florianópolis, Santa Catarina - Brasil.

_____

Profa. Lúcia Helena Martins Pacheco Dr(a)
Coordenação do Curso

Banca examinadora

_____

Prof.(a) Ricardo Pereira e Silva, Dr.(a)
Orientador(a)

Prof.(a) José Eduardo De Lucca, Dr.(a)
Instituição Universidade Federal de Santa Catarina

Prof.(a) Jean Carlo Rossa Hauck, Dr.(a)
Instituição Universidade Federal de Santa Catarina

RESUMO

Este trabalho propõe a implementação de uma Biblioteca Virtual utilizando Progressive Web App (PWA) como solução tecnológica. A pesquisa abrange o desenvolvimento de uma plataforma web responsiva, acessível através de dispositivos móveis e desktops, incorporando funcionalidades de PWA para oferecer uma experiência de usuário próxima à de aplicativos nativos, como o acesso offline, notificações push e capacidade de instalação na tela inicial. Dentre os benefícios dessa abordagem, destacam-se a acessibilidade, responsividade e recursos avançados proporcionados pela combinação de uma biblioteca virtual com tecnologia PWA.

Palavras-chave: Biblioteca Virtual, Progressive Web App, PWA, Desenvolvimento Web.

ABSTRACT

This work proposes the implementation of a Virtual Library using Progressive Web App (PWA) as a technological solution. The research covers the development of a responsive web platform, accessible through mobile devices and desktops, incorporating PWA functionalities to offer a user experience close to that of native applications, such as offline access, push notifications and the ability to install on the home screen. Among the benefits of this approach, the accessibility, responsiveness and advanced features provided by the combination of a virtual library with PWA technology stand out.

Keywords: Virtual Library, Progressive Web App, PWA, Web Development.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

PWA	Progressive Web Application
MVC	Model View Controller
MVT	Model View Template
ORM	Object-Relational Mapping

SUMÁRIO

1 INTRODUÇÃO

A gestão eficiente de uma biblioteca pessoal pode ser um desafio, especialmente para entusiastas da leitura que acumulam uma coleção considerável de livros ao longo do tempo. A ausência de um sistema organizado pode resultar em dificuldades no rastreamento de livros [1], controle de empréstimos e na manutenção de um acervo atualizado. Diante desse cenário, surge a necessidade de uma solução prática e acessível que atenda às demandas específicas de uma biblioteca particular.

O tradicional método de catalogação manual torna-se cada vez mais impraticável diante do crescente número de títulos e da diversidade de formatos, incluindo livros físicos e digitais [13]. Neste contexto, a implementação de um Sistema Web Progressivo (PWA) surge como uma resposta eficaz para proporcionar uma gestão intuitiva e móvel dessa biblioteca pessoal.

Ao contrário das soluções existentes, a proposta deste projeto não apenas busca automatizar o processo de organização, mas também visa oferecer uma experiência de usuário otimizada em dispositivos móveis. A escolha da abordagem PWA não só permite o acesso offline ao acervo, mas também promove a acessibilidade, permitindo que os usuários desfrutem de funcionalidades essenciais em qualquer lugar, a qualquer momento.

Em suma, o desenvolvimento deste software busca oferecer uma solução prática e inovadora para entusiastas da leitura, simplificando a gestão de suas bibliotecas pessoais e proporcionando uma experiência aprimorada de interação com sua coleção de livros.

1.1 OBJETIVOS

1.1.1 Objetivo geral

Desenvolver um sistema web para o controle de livros de uma biblioteca pessoal, utilizando Progressive Web App (PWA) para facilitar o uso do sistema no ambiente mobile**.**

1.1.2 Objetivos específicos

- Realizar um levantamento de requisitos, identificando as funcionalidades essenciais do sistema;
- Projetar e implementar a arquitetura do sistema web, levando em consideração a utilização de tecnologias adequadas, como HTML, CSS e JavaScript, e a adoção de conceitos de PWA para garantir a experiência de uso no ambiente mobile;
- Desenvolver um banco de dados para armazenar as informações dos livros, usuários e empréstimos;
- Garantir a compatibilidade com dispositivos móveis com a tecnologia PWA.

1.2 JUSTIFICATIVA

A implementação de um sistema de gerenciamento de uma biblioteca permite a melhor gestão de livros, particularmente se o conjunto de livros for volumoso [14]. A utilização da tecnologia PWA é estrategicamente escolhida para atender à demanda crescente de acessibilidade móvel. Hoje, a maioria dos usuários acessa a internet por meio de dispositivos móveis [15], e a criação de um sistema que se adapte a esses dispositivos não apenas facilita o acesso, mas também oferece uma experiência mais amigável e intuitiva. Além de proporcionar o acesso offline do sistema, atualizando o banco de dados quando o sistema voltar a ficar online.

1.3 MÉTODO DE PESQUISA

1.3.1 Natureza: Aplicada

Visando criar conhecimento a fim de minimizar um problema existente. No caso desse trabalho, facilitar o gerenciamento de livros em uma biblioteca pessoal, utilizando como meio de implementação um sistema web PWA.

1.3.2 Abordagem: Qualitativo

Não necessita de cálculos específicos para capturar informações referentes à pesquisa. Outras soluções serão pesquisadas e serão fontes de informações para o problema abordado

1.4 ORGANIZAÇÃO DO TEXTO

Neste capítulo, foram apresentados a introdução ao tema, objetivos e motivação deste projeto, algumas características do método de pesquisa. No capítulo 2, serão apresentados conceitos importantes referentes ao tema do projeto. Esses conceitos são necessários para o entendimento do desenvolvimento e execução de projeto.

No capítulo 3 serão apresentados trabalhos com ideias similares a este. No capítulo 4, é mostrada a solução. Aqui são descritos requisitos e funcionalidades da solução, assim como maiores detalhes da solução.

E por último, o capítulo 5 terá a conclusão com a síntese dos resultados obtidos no trabalho, suas limitações e propostas de trabalhos futuros

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão descritos as tecnologias e conceitos necessários para a compreensão da implementação que foi feita no decorrer do semestre.

2.1 Gestão de Bibliotecas e Sistemas de Catalogação

A gestão eficaz de bibliotecas é um desafio que envolve a organização, catalogação e acesso eficiente a uma vasta gama de recursos literários. Segundo [1] a maximização do uso dos recursos disponíveis é um dos princípios fundamentais, enfatizando a importância da organização sistemática para facilitar o acesso do usuário.

Ao considerar a evolução para bibliotecas pessoais, a obra de [2] destaca a importância da personalização na organização de coleções individuais, permitindo uma abordagem mais flexível e adaptada às preferências do usuário. A transição para ambientes digitais também é abordada por [3], que discute os desafios da organização de bibliotecas pessoais em meio à proliferação de recursos digitais e a necessidade de sistemas inovadores para lidar com essa crescente diversidade.

A abordagem desses teóricos e estudiosos ressalta a importância de sistemas de catalogação e gestão eficazes não apenas em bibliotecas tradicionais, mas também em contextos pessoais. A aplicação desses princípios à tecnologia contemporânea, especialmente através de Sistema Web Progressivo (PWA), oferece a oportunidade de unir a eficiência da gestão bibliotecária tradicional com a flexibilidade necessária para atender às necessidades dinâmicas de bibliotecas pessoais.

2.2 Python

Python, uma linguagem de programação de alto nível, tem se destacado como uma escolha popular e versátil em diversos domínios de desenvolvimento de software. Van Rossum, o criador de Python, concebeu-a com princípios como legibilidade e simplicidade, resultando em um código que é fácil de aprender e manter. A comunidade de desenvolvedores se beneficiou da abordagem "batteries

included" de Python, fornecendo uma ampla biblioteca padrão que abrange desde manipulação de dados até interfaces gráficas.

No contexto do desenvolvimento web, o framework Django [4] se destaca como uma escolha robusta para construir aplicativos web escaláveis e seguros. Sua arquitetura seguindo o padrão Model-View-Controller (MVC) simplifica o desenvolvimento, permitindo a criação rápida de aplicativos eficientes. Flask [5], uma estrutura mais leve, oferece flexibilidade para projetos menores, mantendo o poder e a simplicidade característicos de Python.

A adoção crescente de Python em ciência de dados e aprendizado de máquina é evidenciada por bibliotecas como NumPy [6] e TensorFlow. Essas ferramentas têm impulsionado avanços significativos em análise de dados e inteligência artificial, destacando a capacidade de Python em suportar uma ampla gama de aplicações.

A linguagem python tem uma linguagem simples de se entender, o escopo de cada trecho de código é definido por sua indentação em espaços ou tabs. Um exemplo de código python é mostrado na figura 1.

```python
# Exemplo de Classe em Python para Representar um Livro na Biblioteca
class Livro:
    def __init__(self, titulo, autor, ano_publicacao):
        self.__titulo = titulo
        self.__autor = autor
        self.__ano_publicacao = ano_publicacao

    def get_titulo(self):
        return self.__titulo

    def set_titulo(self, nova_titulo):
        self.__titulo = nova_titulo

    def get_autor(self):
        return self.__autor

    def set_autor(self, novo_autor):
        self.__autor = novo_autor

    def get_ano_publicacao(self):
        return self.__ano_publicacao

    def set_ano_publicacao(self, novo_ano_publicacao):
        self.__ano_publicacao = novo_ano_publicacao

    def exibir_informacoes(self):
        print(f'Título: {self.get_titulo()}')
        print(f'Autor: {self.get_autor()}')
        print(f'Ano de Publicação: {self.get_ano_publicacao()}')

livro1 = Livro('Dom Casmurro', 'Machado de Assis', 1899)
livro2 = Livro('1984', 'George Orwell', 1949)

livro1.exibir_informacoes()
print('---')
livro2.exibir_informacoes()
```

Figura 1 - Exemplo de código python

Fonte: O autor (2023)

2.3 DJANGO

Django, um framework web em Python, foi desenvolvido com a missão de tornar o desenvolvimento web eficiente e pragmático. O "Django Philosophy" [7] destaca princípios fundamentais como "Don't Repeat Yourself" (DRY) e "Convention Over Configuration", promovendo uma abordagem estruturada e sem redundâncias no desenvolvimento de aplicativos web.

O padrão de arquitetura Model-View-Controller (MVC) adotado pelo Django, e particularmente o seu equivalente Model-View-Template (MVT), facilita a organização lógica e a manutenção de código em projetos web [8]. O ORM (Mapeamento Objeto Relacional) integrado ao Django simplifica a interação com bancos de dados, permitindo que desenvolvedores realizem operações com o banco de dados utilizando Python puro. A base de dados padrão utilizada em projetos django é sqlite3. O modelo pode ser visualizado na figura 2.

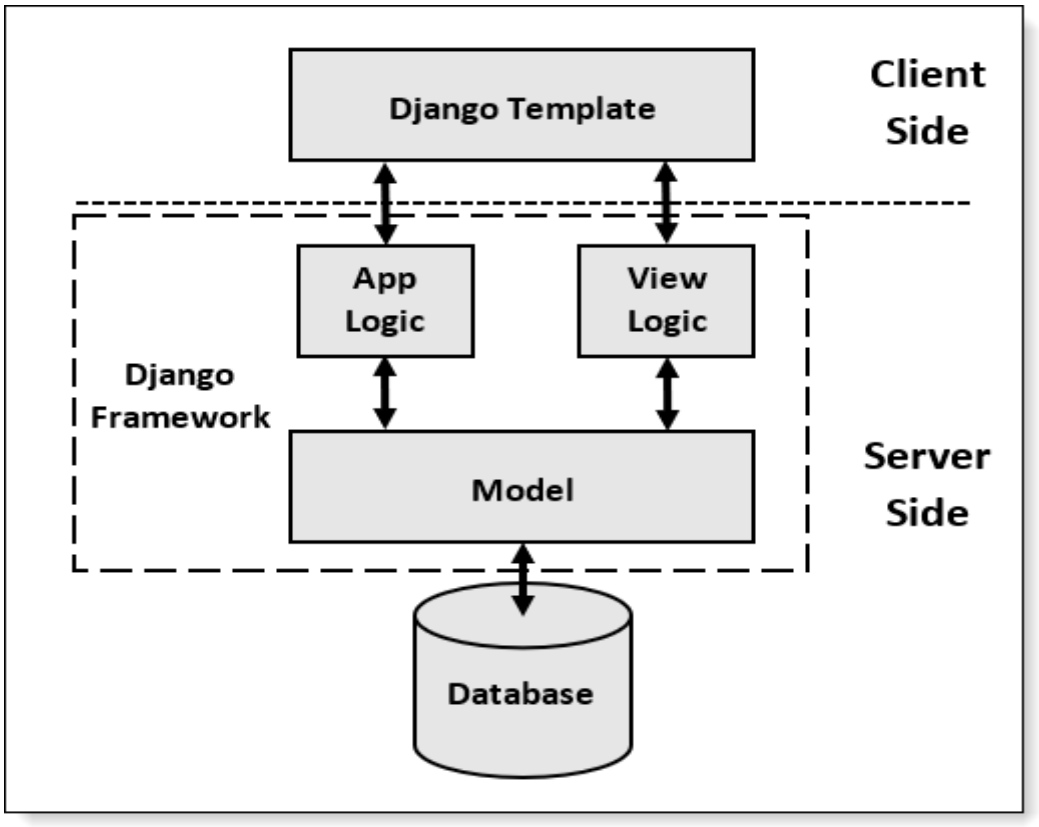


Figura 2 - Uma forma simplificada da arquitetura django

Fonte: https://masteringdjango.com/django-tutorials/mastering-django-structure/

Para a construção de interfaces de usuário, o Django utiliza o sistema de templates, facilitando a criação de páginas dinâmicas e a integração com dados provenientes do back-end. Além disso, a estrutura modular do Django, com seu sistema de aplicativos reutilizáveis, incentiva a criação de componentes independentes e extensíveis [9].

A segurança é uma prioridade no desenvolvimento web, e o Django inclui várias camadas de proteção contra vulnerabilidades comuns. O framework segue práticas recomendadas para evitar ataques como SQL injection, cross-site scripting

(XSS) e cross-site request forgery (CSRF), proporcionando uma base segura para o desenvolvimento de aplicações web [10].

2.4 Aplicações Web

Aplicações web são softwares ou programas que são executados em um navegador da web e acessados pela internet. Elas têm a vantagem de serem acessíveis em uma variedade de dispositivos (computadores, smartphones, tablets) sem exigir instalação prévia, bastando ter um navegador e conexão com a internet.

Existem diferentes tipos de aplicações web:

- **Estáticas**: São páginas ou sites estáticos, compostos principalmente por HTML, CSS e possivelmente JavaScript. São usados principalmente para apresentar informações e não possuem funcionalidades interativas complexas;
- **Dinâmicas**: Estas são mais avançadas e interativas. Utilizam bancos de dados e linguagens de programação do lado do servidor (como PHP, Ruby, Python, etc.) para fornecer conteúdo personalizado e interações mais complexas;
- **Progressive Web Apps (PWAs)**: São aplicações web que utilizam tecnologias modernas para oferecer uma experiência próxima à de aplicativos nativos em dispositivos móveis e desktops. Elas podem funcionar offline, oferecer notificações push e serem instaladas na tela inicial dos dispositivos.

As aplicações web têm a vantagem de serem facilmente atualizadas, pois uma vez atualizada no servidor, todas as instâncias acessam a versão mais recente. Além disso, eliminam a necessidade de instalação manual, são multiplataforma e podem ser acessadas de qualquer lugar, desde que haja acesso à internet.

2.4.1 PWA

O conceito de Progressive Web Apps (PWAs) surgiu em 2015, proposto pelo engenheiro do Google, Alex Russell, e pelo designer Frances Berriman. O termo foi cunhado em um artigo detalhado que descreveu a ideia de aplicativos web que ofereciam uma experiência próxima à de aplicativos nativos em dispositivos móveis, por meio da utilização de tecnologias web progressivas.

Russell e Berriman [11] destacaram a necessidade de criar aplicativos web que fossem confiáveis, rápidos e engajadores, mesmo em condições de conectividade limitada. A proposta foi impulsionada pela evolução das capacidades dos navegadores e pela demanda crescente por experiências de aplicativos móveis sem a necessidade de instalação a partir de lojas de aplicativos tradicionais.

Essa abordagem inovadora visava combinar o melhor dos aplicativos nativos, como acesso offline, notificações e experiência de usuário fluida, com a acessibilidade e facilidade de uso dos aplicativos da web. O conceito rapidamente ganhou destaque, resultando em iniciativas e padrões da indústria para impulsionar o desenvolvimento e a adoção de PWAs.

Algumas características de pwa de acordo em [11] e [12]:

- **Responsivo**: poder se adequar a qualquer formato;
- **Independente de conectividade**: aprimorado progressivamente com Service Workers para permitir que trabalhem off-line;
- **Recente**: sempre atualizado de forma transparente graças ao processo de atualização do Service Worker;
- **Seguro**: servido via TLS (um requisito do Service Worker) para evitar snooping;
- **Detectáveis**: são identificáveis como "aplicativos" graças aos manifestos W3C e ao escopo de registro do Service Worker, permitindo que os mecanismos de pesquisa os encontrem;
- **Reengajável**: pode acessar as UIs de reengajamento do sistema operacional; por exemplo. Notificações via push;
- **Instalável**: na tela inicial por meio de prompts fornecidos pelo navegador, permitindo que os usuários "mantenham" os aplicativos que consideram mais úteis sem o incômodo de uma loja de aplicativos

2.4.1.1 Service Worker

O Service Worker é uma tecnologia fundamental para o funcionamento de Progressive Web Apps (PWA), atuando como um script JavaScript executado pelo navegador em segundo plano, independentemente da página web estar aberta. Ele permite recursos [16] como:

- Cacheamento de conteúdo para acesso offline;
- Melhoria no desempenho através do pré-cache de recursos;
- Notificações push;
- Atualizações automáticas.

2.4.1.2 Manifest.json

O arquivo manifest.json é um arquivo de metadados essencial para PWAs, fornecendo informações sobre a aplicação para o navegador. Ele contém detalhes [17] como:

- Nome da aplicação;
- Ícones para diferentes dispositivos;
- Tema da aplicação (cor de fundo, cor do tema);
- Configurações de exibição (modo de exibição, orientação, tela cheia).

3 TRABALHOS RELACIONADOS

3.1 GOODREADS

O Goodreads [18] é uma plataforma social voltada para leitores que oferece recursos para catalogar, avaliar e recomendar livros. Em 2013, a Amazon, uma das maiores varejistas online do mundo, adquiriu o Goodreads, mantendo-o como uma entidade separada.

Desde a aquisição, a Amazon manteve o Goodreads como uma plataforma independente, permitindo que os usuários do Goodreads descubram, revisem e compartilhem recomendações de livros sem a necessidade de fazer compras na Amazon. No entanto, a integração entre as duas plataformas ocorreu em certos aspectos, como a possibilidade de sincronizar listas de livros entre as contas do Goodreads e da Amazon, facilitando a compra de livros sugeridos.

Esta plataforma oferece aos usuários a oportunidade de catalogar, avaliar e revisar livros, além de interagir com uma comunidade de leitores. Uma característica central do Goodreads é a capacidade dos usuários de criar estantes virtuais com livros que já leram, estão lendo no momento ou desejam ler no futuro, permitindo um gerenciamento eficaz de sua biblioteca pessoal. Além disso, a plataforma fornece recursos para descobrir novos livros por meio de recomendações personalizadas, listas de leitura, grupos de discussão e revisões detalhadas, facilitando a interação entre os membros da comunidade. Sua abordagem focada no usuário e sua ênfase na comunidade de leitores são aspectos relevantes a serem considerados na concepção de uma biblioteca virtual, visando proporcionar uma experiência enriquecedora para os usuários e incentivar o engajamento com a leitura.

As figuras 3 e 4 mostram exemplos de personalização do usuário e da navegação nas listas de livros.

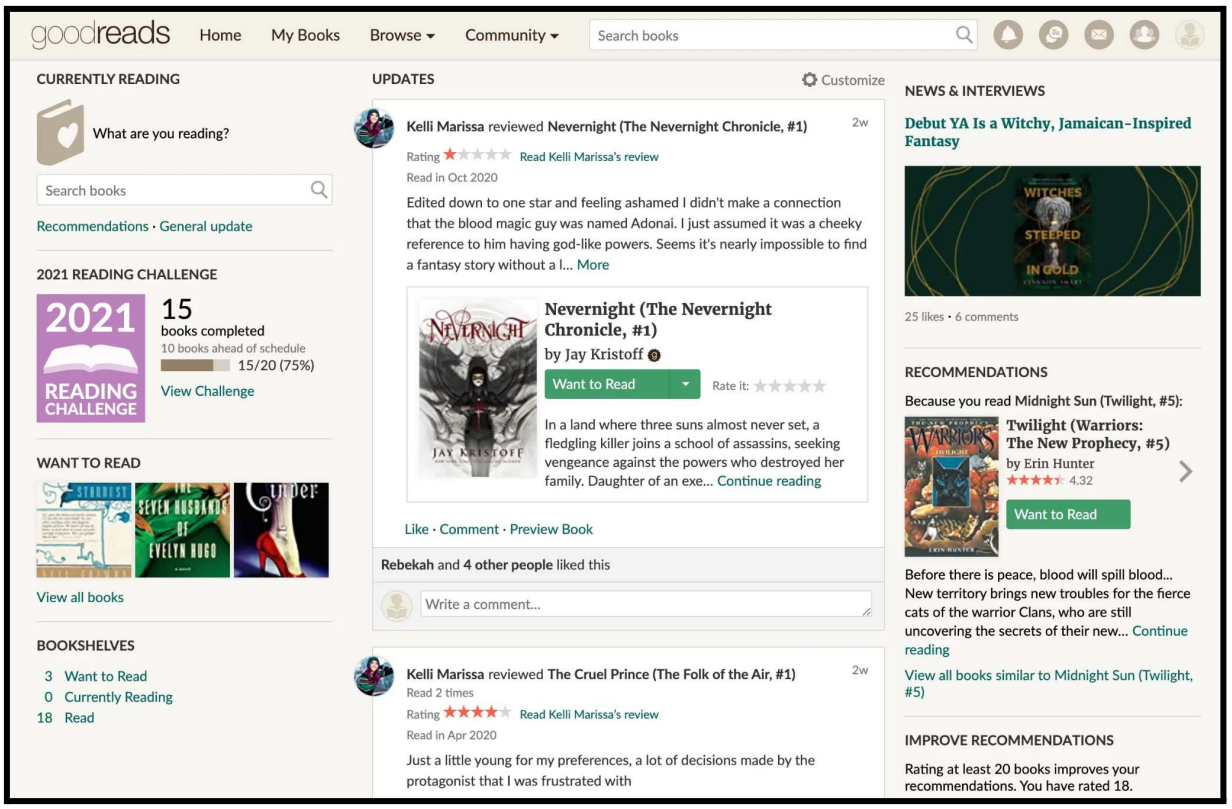


Figura 3 - Um exemplo de página inicial pós-login no goodreads

Fonte: [21]

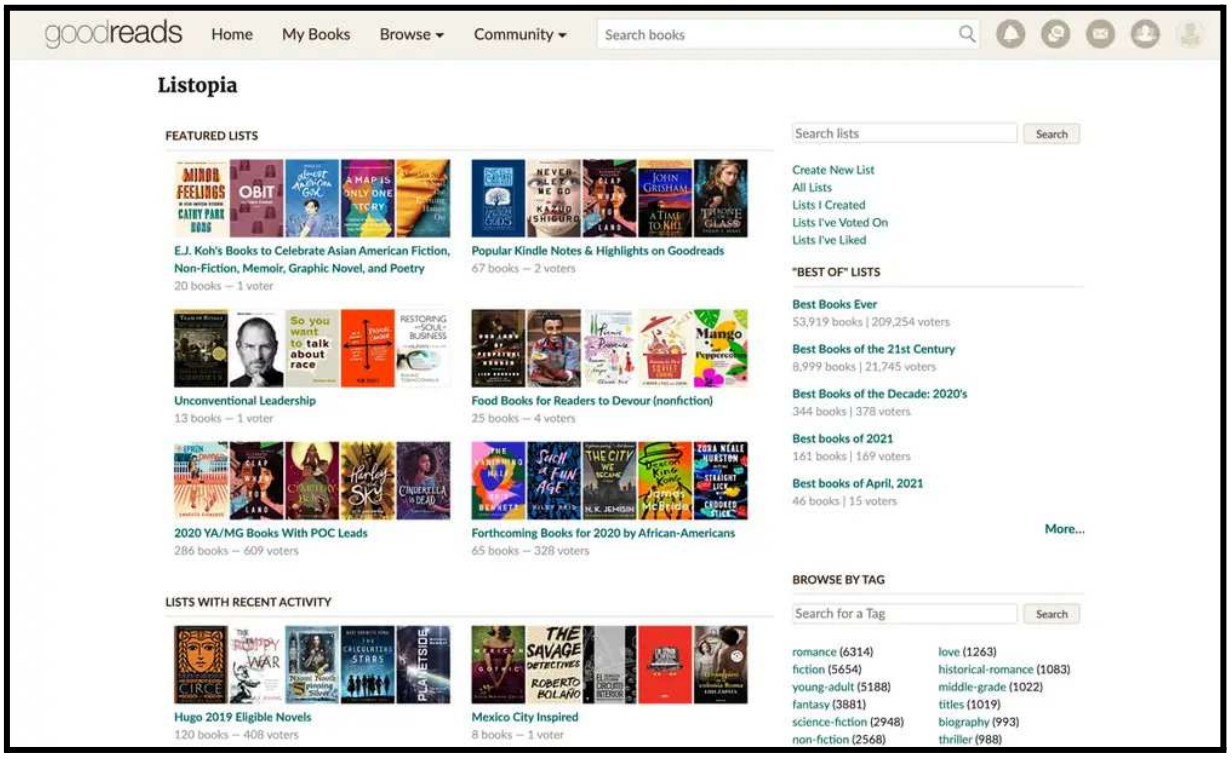


Figura 4 - Um exemplo de navegação nas listas de leitura do goodreads

Fonte: [21]

3.2 LIBRARYTHING

LibraryThing [19] é uma plataforma online dedicada à catalogação de livros e interações sociais relacionadas a eles. Criado em 2005, oferece aos usuários a capacidade de catalogar suas bibliotecas pessoais, fornecendo informações detalhadas sobre livros, incluindo títulos, autores, resumos, classificações e resenhas. A plataforma também possui recursos de interação social, permitindo que os usuários discutam livros, participem de grupos de leitura e compartilhem recomendações.LibraryThing destaca-se por sua comunidade ativa e por oferecer ferramentas avançadas de organização para amantes de livros. Embora seja uma plataforma valiosa para gestão de coleções pessoais, ela tende a focar mais na interação social do que na experiência de criação de uma biblioteca virtual personalizada e expansível, diferenciando-se, assim, do escopo proposto para este trabalho. A figura 5 mostra o exemplo de uma lista de recomendações na LibraryThing.

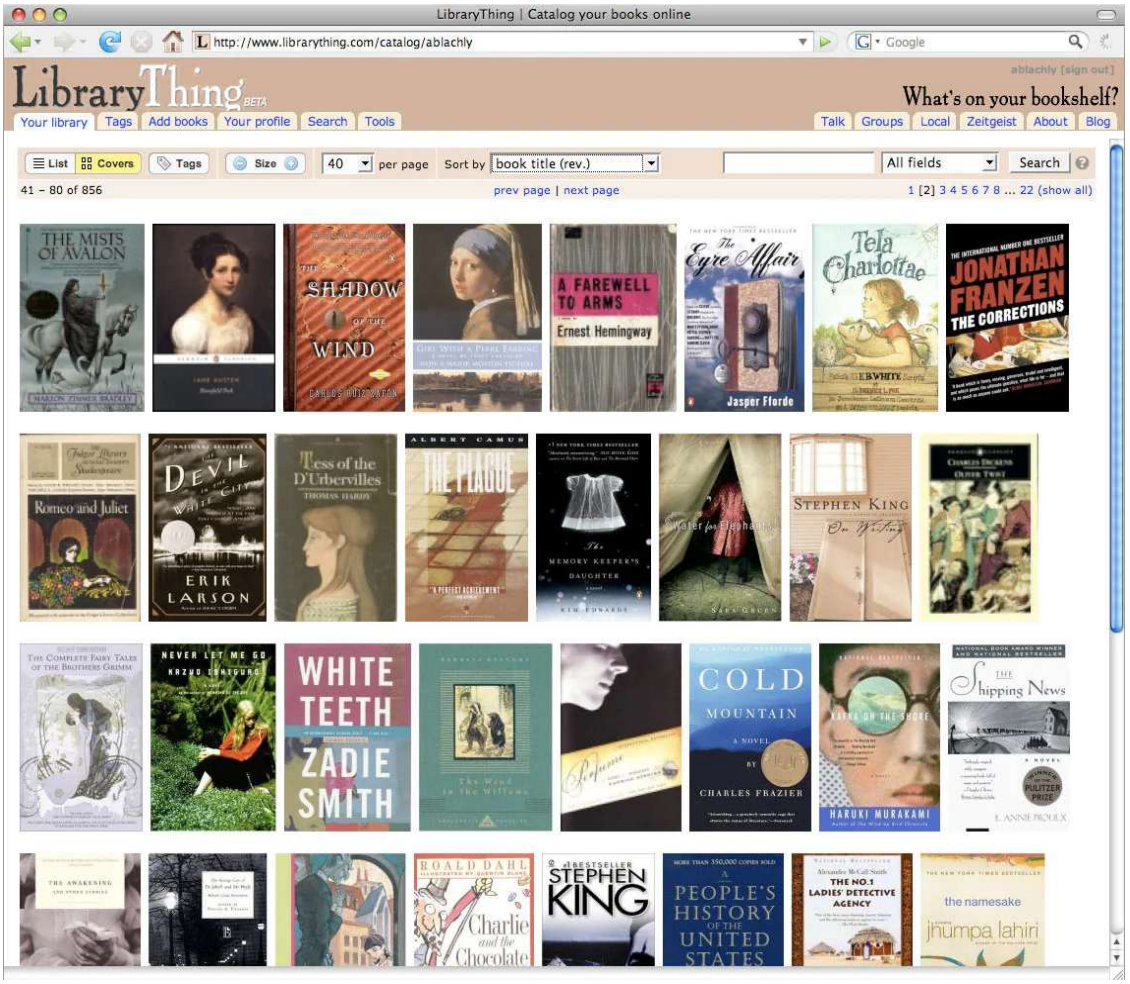


Figura 5 - Um exemplo de recomendações em LibraryThing

Fonte: [22]

3.3 ANÁLISE

O Goodreads destaca-se como uma plataforma social popular entre os leitores, oferecendo a catalogação de livros, resenhas, recomendações e interações entre usuários. Sua grande comunidade e recomendações personalizadas baseadas no histórico de leitura são pontos fortes. No entanto, por vezes, pode priorizar a interação social em detrimento da experiência de construção de uma biblioteca personalizada.

Por outro lado, o LibraryThing é reconhecido por suas ferramentas avançadas de catalogação e organização, além de uma comunidade ativa. Foca na precisão da catalogação, mas pode oferecer menos personalização na construção de uma biblioteca virtual expansível.

O software proposto por este trabalho visa criar uma plataforma que permita o gerenciamento de coleções pessoais, acesso fácil a informações sobre livros e uma experiência otimizada para dispositivos móveis. A solução visa oferecer uma experiência mais personalizada, ao mesmo tempo em que incorpora recursos de catalogação e acessibilidade offline proporcionados por um PWA.

4 DESENVOLVIMENTO

Neste capítulo será descrito o sistema proposto para o gerenciamento de livros em uma biblioteca virtual.

4.1 SOLUÇÃO PROPOSTA

O sistema proposto será uma aplicação web PWA desenvolvido em python utilizando django como base do sistema web e por conseguinte sqlite3 como base de dados, uma vez que é o banco de dados padrão de projetos django.

4.1.1 Diagrama de classes

É mostrado na figura 6 o diagrama de classes referente às entidades do sistema, *Usuario* se refere às informações do usuário logado e *Livro* se refere as informações do livro e empréstimos gerenciados pelo sistema.

Figura 6 - Diagrama de classes referente às entidades do projeto

Fonte: O Autor

4.1.2 Requisitos do sistema

Identificação	Descrição
RF-01	O programa deve permitir o início de sessão para cada usuário (Log-in e log-out)
RF-02	O programa deve manter informações sobre os usuários do sistema (criar, pesquisar, editar, atualizar e listar os dados referentes).
RF-03	O programa deve manter informações sobre livros, inseridos pelos usuários (criar, pesquisar, editar, atualizar e listar os dados referentes).
RF-04	O programa deve permitir um usuário compartilhar sua biblioteca/ lista de livros com outro usuário
RF-05	O programa deve possibilitar a extração de informações do livro pela imagem ou número do código de barras
RNF-01	O programa será construído para rodar em ambiente web, compatível com os principais navegadores disponíveis no mercado: Microsoft Edge, Google Chrome, Firefox.
RNF-02	Será utilizado práticas de PWA para que o programa funcione de forma responsiva tanto por dispositivos mobile quanto por notebook e desktop

Tabela 1 - Requisitos funcionais e não-funcionais do sistema

Fonte: O Autor (2023)

4.1.3 Dicionário de dados

Entidade	Atributo	Descrição	Obrigatoriedade	Tipo de dado
Livro	id	Identificador único do livro	Sim	Inteiro
Livro	titulo	Título do livro	Sim	Texto
Livro	autor	Autor do livro	Sim	Texto
Livro	ano_publicacao	Ano de publicação do livro	Sim	Inteiro

Livro	descricao	Uma breve descrição fazendo um breve resumo do livro	Sim	Texto
Livro	isbn	Código ISBN do livro	Sim	Texto
Livro	emprestado	Flag indicando se o livro foi emprestado ou não	Não, mas desejável	Booleano
Livro	doado	Flag indicando se o livro foi doado ou não	Não, mas desejável	Booleano
Livro	avaliacao	Nota dada ao livro pelo usuário	Sim	Ponto Flutuante
Livro	nome_pessoa_empr	Nome da pessoa para a qual o livro foi emprestado ou doado, se for o caso	Não, mas desejável	Texto
Livro	data_emprest_doa	Data que o livro foi emprestado ou doado, se for o caso	Não, mas desejável	Data
Usuario	id	Identificador único do usuário	Sim	Inteiro
Usuario	login	Login do usuário	Sim	Texto
Usuario	senha	Senha do usuário	Sim	Texto

Tabela 2 - Dicionário de dados das entidades do projeto

Fonte: O Autor (2023)

4.1.4 Desenvolvimento do sistema

4.1.4.1 Venv

Venv é um módulo python que permite criar ambientes virtuais isolados para projetos python. Ele é importante para isolar um projeto python do restante do sistema, evitando conflitos e auxilia a manter a consistência nos ambientes de desenvolvimento.

A primeira coisa que foi feita no projeto foi a criação desse ambiente virtual com o comando python -m venv venv e depois é feita a habilitação desse ambiente sempre que entrar no ambiente de desenvolvimento com o comando source Scripts/activate.

É mostrado na figura 7 a estrutura de arquivos do venv.

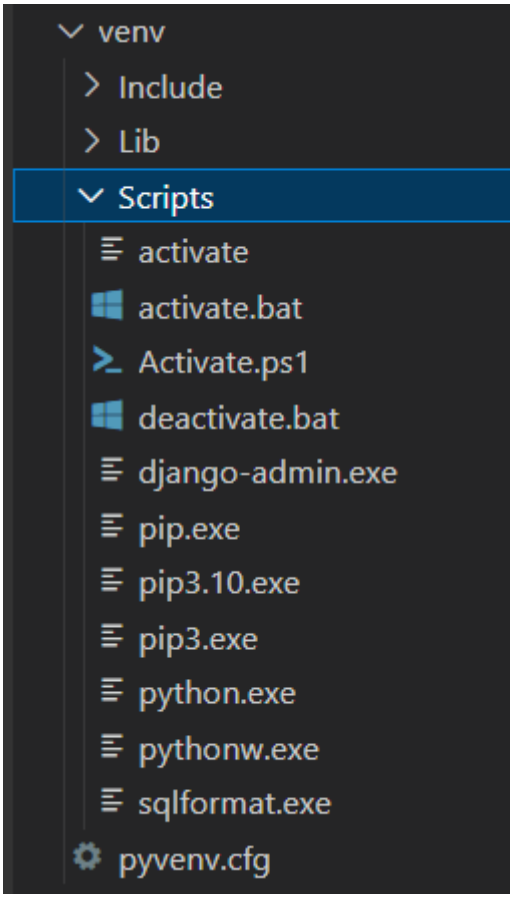


Figura 7 - Estrutura do venv

Fonte: O Autor(2023)

4.1.4.2 Requirements.txt

Assim como o arquivo package.json para o Node.js e o arquivo pom.xml para o java, foi criado um arquivo neste projeto chamado “requirements.txt” que contém dependências a serem instaladas e atualizadas. Isso é feito ao abrir o projeto com o comando pip install -r requirements.txt .

Neste arquivo é curto e contém as linhas: “django” e “django-pwa”. A maior parte das bibliotecas e scripts necessários para o projeto vem com a dependência ”django”.

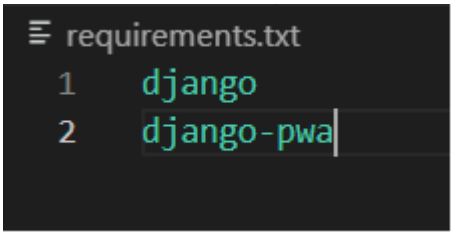


Figura 8 - Arquivo requirements.txt

Fonte: O Autor(2023)

4.1.4.3 Django-admin e manage.py

O projeto django é iniciado com o comando django-admin, `django-admin startproject config .` e após isso todas as operações feitas no servidor django são feitas pelo script manage.py. Seja inicializar o servidor, sinalizar uma nova entidade no banco de dados ou entrar em linha de comando python para acompanhar os dados criados e alterados. Ressaltam-se os comandos *runserver, makemigrations, migrate* e *shell* referentes às ações mencionadas anteriormente. A figura 9 mostra os arquivos referente ao projeto base criado via django-admin.

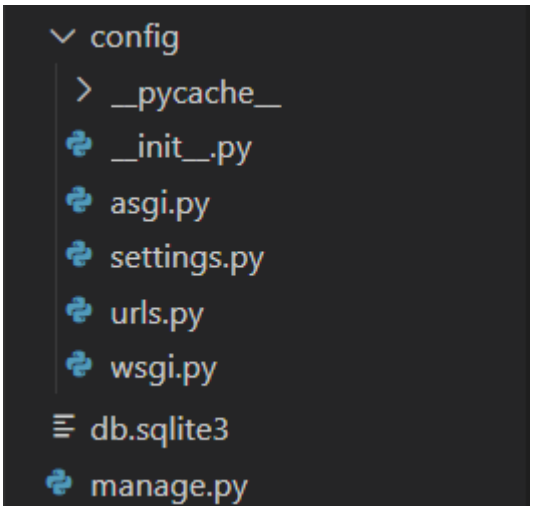


Figura 9 - Arquivos referentes a django-admin

Fonte: O Autor(2023)

4.1.4.4 App livro e entidade usuario

O termo “app livro” se refere a aplicação e entidade criada pelo comando *startapp* com o script manage.py, é aqui que está a definição do modelo da entidade do banco de dados referente aos livros e seus empréstimos, além de configurações da respectiva view do front-end. A figura 10 mostra a estrutura de arquivos desse app.

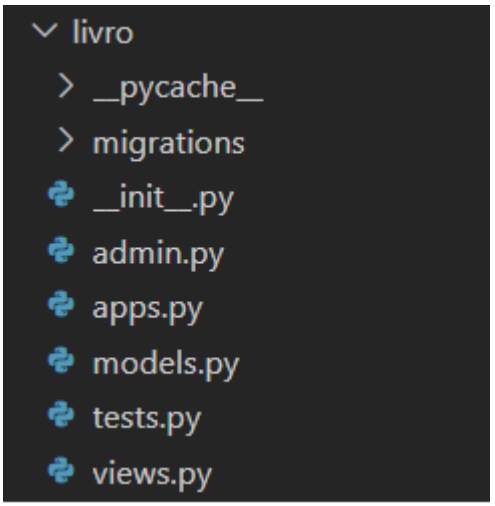


Figura 10 - Arquivos referentes ao app livro

Fonte: O Autor(2023)

A entidade livro é definida com os campos: titulo, autor, ano_publicacao, descricao, isbn, emprestado, doado, avaliacao, nome_pessoa_empr e data_emprest_doa conforme a figura 11. E a entidade usuario é definida pelos atributos login e senha como visto na figura 12. Além desses campos, também tem o campo de *pk* referente a chave primária que está definida na superclasse models.Model.

```python
class Livro(models.Model):
    titulo           = models.CharField(max_length=200)
    autor            = models.CharField(max_length=100)
    ano_publicacao   = models.IntegerField()
    descricao        = models.TextField(blank=True)
    isbn             = models.CharField(max_length=30)

    emprestado       = models.BooleanField(default=False)
    doado            = models.BooleanField(default=False)
    avaliacao        = models.FloatField()

    nome_pessoa_empr  = models.CharField(max_length=250, null=True, blank=True)
    data_emprest_doa  = models.DateField(null=True, blank=True)
```

Figura 11 - Arquivo model.py referentes ao app livro

Fonte: O Autor(2023)

```python
class Usuario(models.Model):
    login = models.CharField(max_length=120, null=False, unique=True)
    senha = models.CharField(max_length=120, null=False)
```

Figura 12 - Arquivo model.py referentes ao app livro

Fonte: O Autor(2023)

4.1.4.5 Autenticação

Os scripts e classes referentes a autenticação no projeto django estão na biblioteca `django.contrib.auth'. Tal biblioteca precisa estar declarada na flag INSTALLED_APPS do arquivo settings.py, ela estará junto de outros apps padrões e meus apps do sistema, como visto da figura 13

Além disso, é necessário configurar o roteamento e configurações de página no arquivo urls.py, onde pode é usado o caminho padrão do django “accounts/” ou “login/”. Isso é visto na figura 14 e por fim é necessária a anotação **@login_required** nas funções referentes às views

```python
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',    # responsavel pela autenticacao do usuario
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    # meus apps
    'livro',
]
```

Figura 13 - Arquivo settings.py referentes a INSTALLED_APPS

Fonte: O Autor(2023)

```python
urlpatterns = [
    path('', home_view, name="home"),
    path('admin/', admin.site.urls),

    path('login/', login_view, name="login"),

    path('dashboard/', dashboard_view, name="dashboard"),
    path('novoLivro/', novo_livro_view, name="novo_livro"),
    path('pesquisaLivro/', pesquisa_livro_view, name="pesquisa_usuario"),
    path('novoUsuario/', novo_usuario_view, name="novo_usuario"),
    path('pesquisaUsuario/', pesquisa_usuario_view, name="pesquisa_livro"),

    path('editarUsuario/<int:id_usuario>', editar_usuario_view, name="editar_usuario"),
    path('editarLivro/<int:id_livro>', editar_livro_view, name="editar_livro"),

]
```

Figura 14 - Arquivo urls.py referentes a autenticação

Fonte: O Autor(2023)

4.1.4.6 PWA

Neste projeto é utilizado a biblioteca django-pwa que dá suporte a implementação da tecnologia pwa no django e como visto anteriormente, foi adicionado em requirements.txt. Para configurá-lo é necessário adicionar ‘pwa’ em INSTALLED_APPS de settings.py, e adicioná-la no roteamento como visto na figura 15.

```
INSTALLED_APPS = [
    ...

    'pwa',
]

urlpatterns = [
    ...
    path('', include('pwa.urls'))
```

Figura 15 - Variáveis em settings.py e urls.py modificadas

Fonte: O Autor(2023)

O serviceworker, script javascript, que é responsável pela cache e outros serviços no front end é representado na figura 16

```
1
2   var staticCacheName = 'djangopwa-v1';
3
4   self.addEventListener('install', function(event) {
5     event.waitUntil(
6       caches.open(staticCacheName).then(function(cache) {
7         return cache.addAll([
8           '',
9         ]);
10      })
11    );
12  });
13
14  self.addEventListener('fetch', function(event) {
15    var requestUrl = new URL(event.request.url);
16      if (requestUrl.origin === location.origin) {
17        if ((requestUrl.pathname === '/')) {
18          event.respondWith(caches.match(''));
19          return;
20        }
21      }
22      event.respondWith(
23        caches.match(event.request).then(function(response) {
24          return response || fetch(event.request);
25        })
26      );
27  });
```

Figura 16 - serviceworker.js

Fonte: O Autor(2023)

O django cria um arquivo manifest.json de forma automática, para depois configurações específicas sejam alteradas.

E por último, anotações {% load pwa %} e{% progressive_web_app_meta %} são adicionadas a página inicial do projeto para efetuar a carga das configurações.

4.1.4.7 Telas

4.1.4.7.1 Exemplos Código

Uma das telas principais do projeto são referente a tela de login, onde é feita a autenticação do usuário e é passado um token csrf, padrão do django para garantir segurança quando for executado o form. Referente na figura 17. A outra tela diz respeito visualização dos dados de livros, com código da figura 18, esse arquivo html não foi utilizado no projeto, mas esse código foi utilizado posteriormente na tela de pesquisa e filtro de livros.

```
<form method="post" action="{% url 'login' %}">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Login</button>
</form>
```

Figura 17 - parte de login.html

Fonte: O Autor(2023)

```
<h1>Lista de Livros</h1>
<table border="1">
    <thead>
        <tr>
            <th>Título</th>
            <th>Autor</th>
            <th>Ano de Publicação</th>
            <th>Descrição</th>
            <th>Disponível</th>
            <th>ISBN</th>
        </tr>
    </thead>
    <tbody>
        {% for livro in livros %}
            <tr>
                <td>{{ livro.titulo }}</td>
                <td>{{ livro.autor }}</td>
                <td>{{ livro.ano_publicacao }}</td>
                <td>{{ livro.descricao }}</td>
                <td>{{ livro.disponivel }}</td>
                <td>{{ livro.isbn }}</td>
            </tr>
        {% endfor %}
    </tbody>
</table>
<p>logout</p>
<a href="{% url 'logout' %}">Logout</a>
```

Figura 18 - parte de lista_livros.html

Fonte: O Autor(2023)

4.1.4.7.2 Tela Home

Esta tela se refere a página raiz do projeto, ela é a página inicial. O único propósito dela é redirecionar o usuário para a tela de login

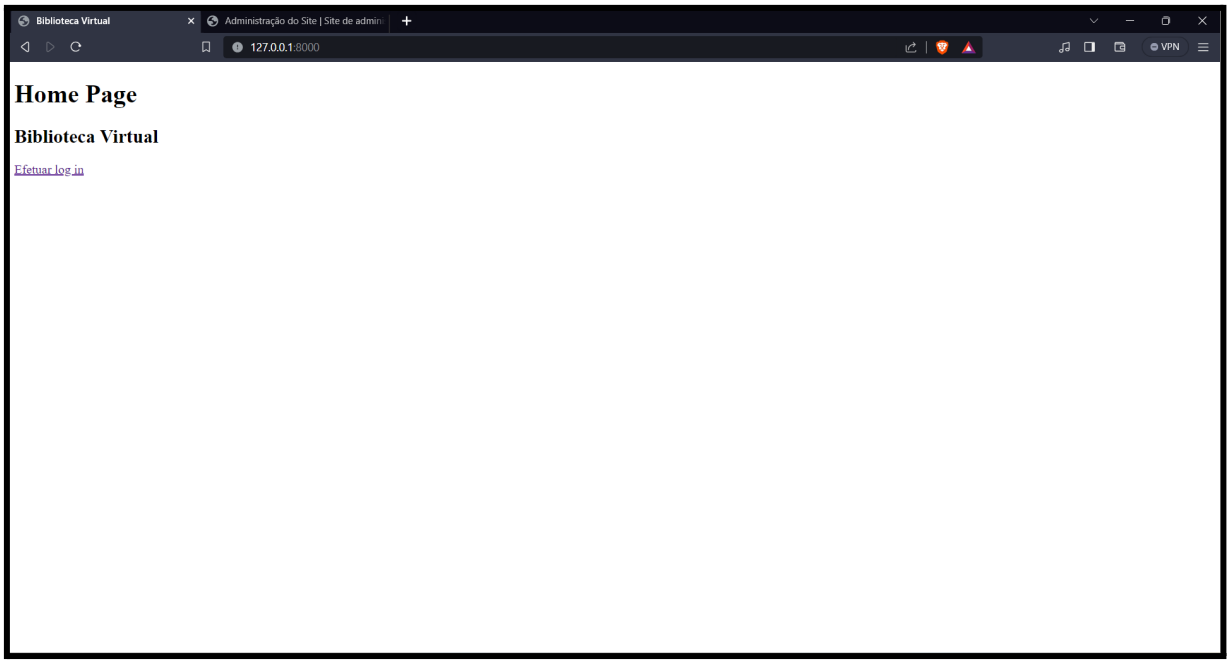


Figura 19 - Página home

Fonte: O Autor

4.1.4.7.3 Tela login

Esta tela é a tela referente a autenticação do usuário. Esperando o login e senha em seus respectivos campos de entrada. Se o usuário existir e a senha for correta, o usuário será redirecionado para a página dashboard.

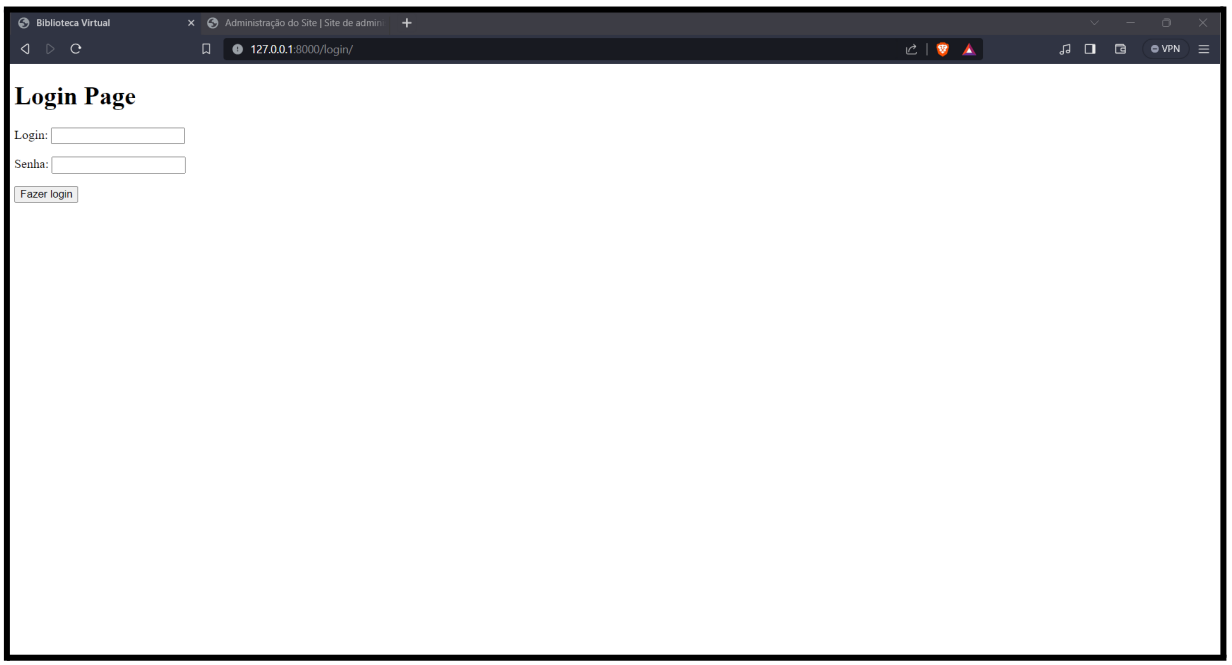


Figura 20 - Página login

Fonte: O Autor

4.1.4.7.4 Tela dashboard

Esta página é a página inicial pós-login, nela tem um menu de navegação para outras telas do sistema e para fazer o logout. Há também uma mensagem de bem-vindo(a) com o login do usuário logado.

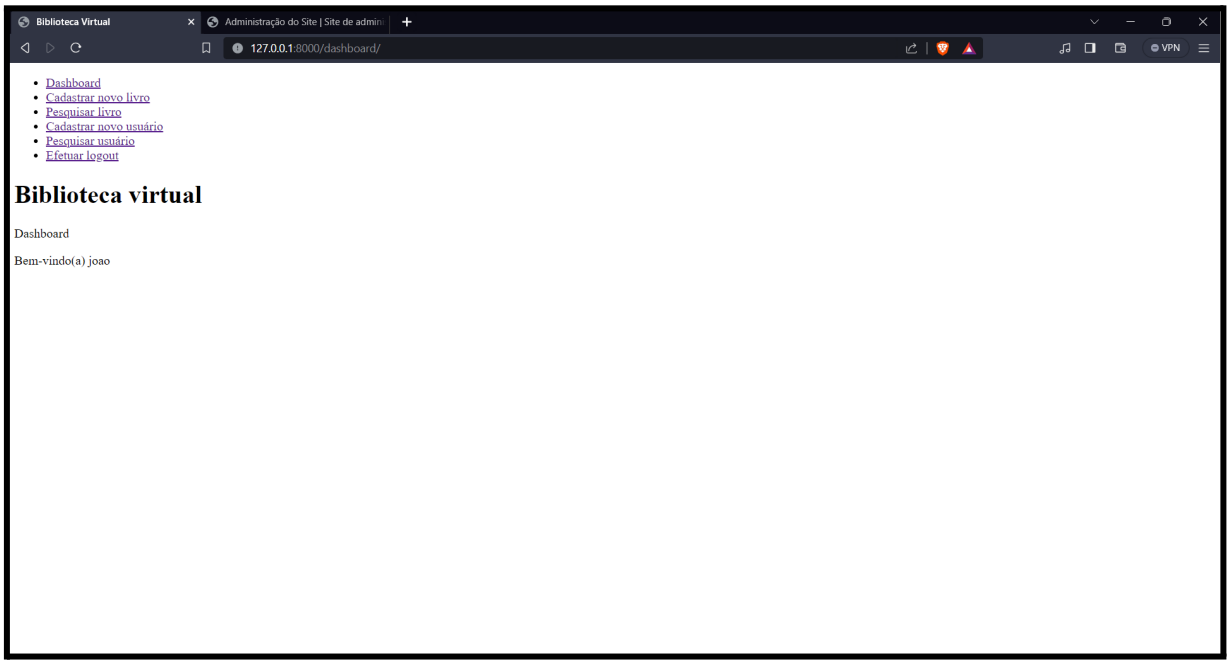


Figura 21 - Página dashboard

Fonte: O Autor

4.1.4.7.5 Tela pesquisa usuário

Nesta tela, é possível visualizar os usuários cadastrados no sistema, filtrar o usuário pelo login e ir para uma página de edição do registro ao clicar no link Editar na linha referente ao registro.

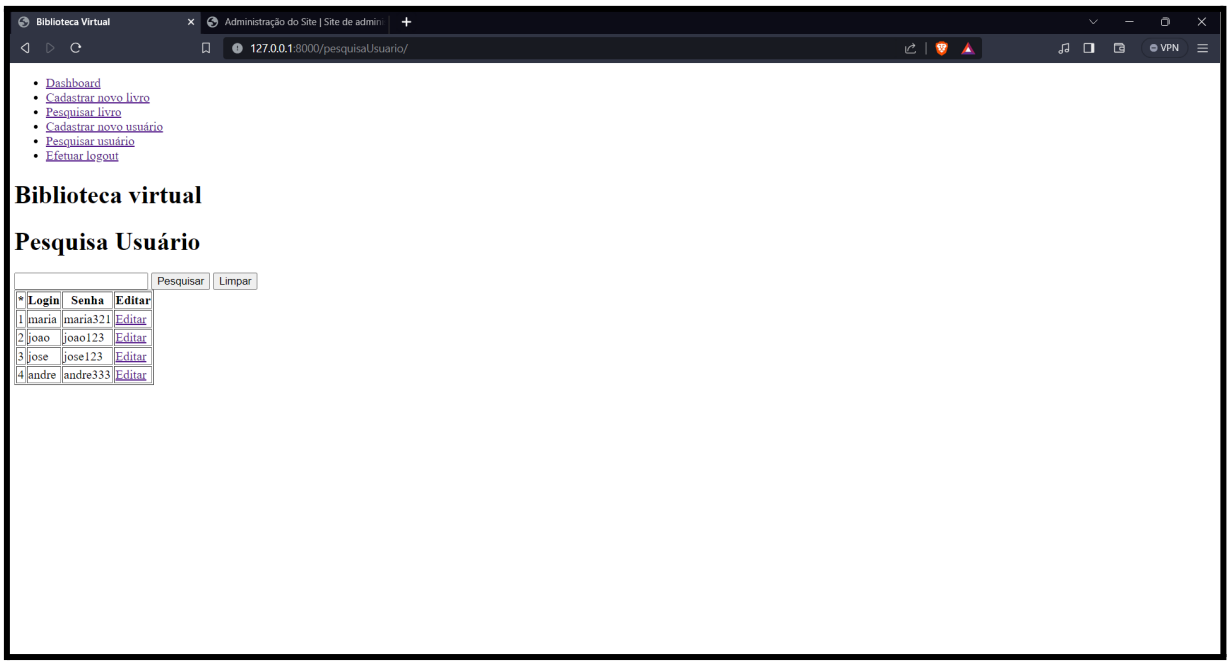


Figura 22 - Página pesquisa usuário

Fonte: O Autor

4.1.4.7.6 Tela editar usuário

Chega-se nesta dela ao clicar no link “Editar” no registro selecionado na página de pesquisa de usuários. Não é possível alterar o login, mas pode-se editar a senha.

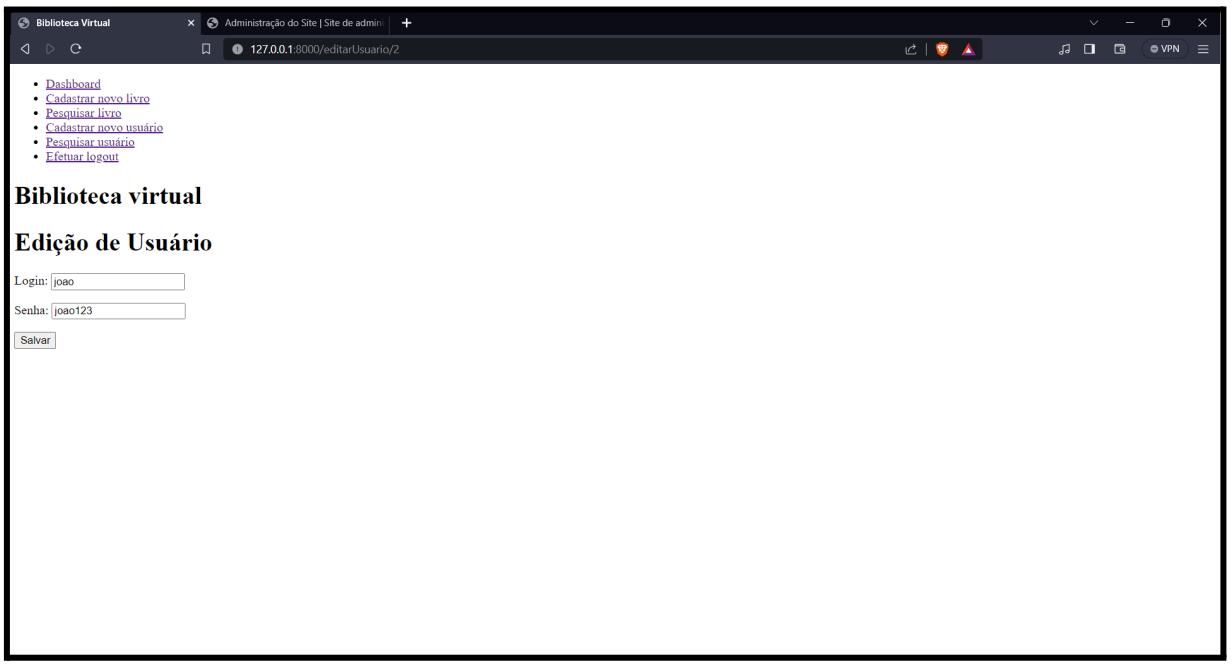


Figura 23 - Página editar usuário

Fonte: O Autor

4.1.4.7.7 Tela novo usuário

Esta tela é similar à de edição. Mas ao invés de salvar sobre um registro já existente, é criado um novo registro ao salvar.

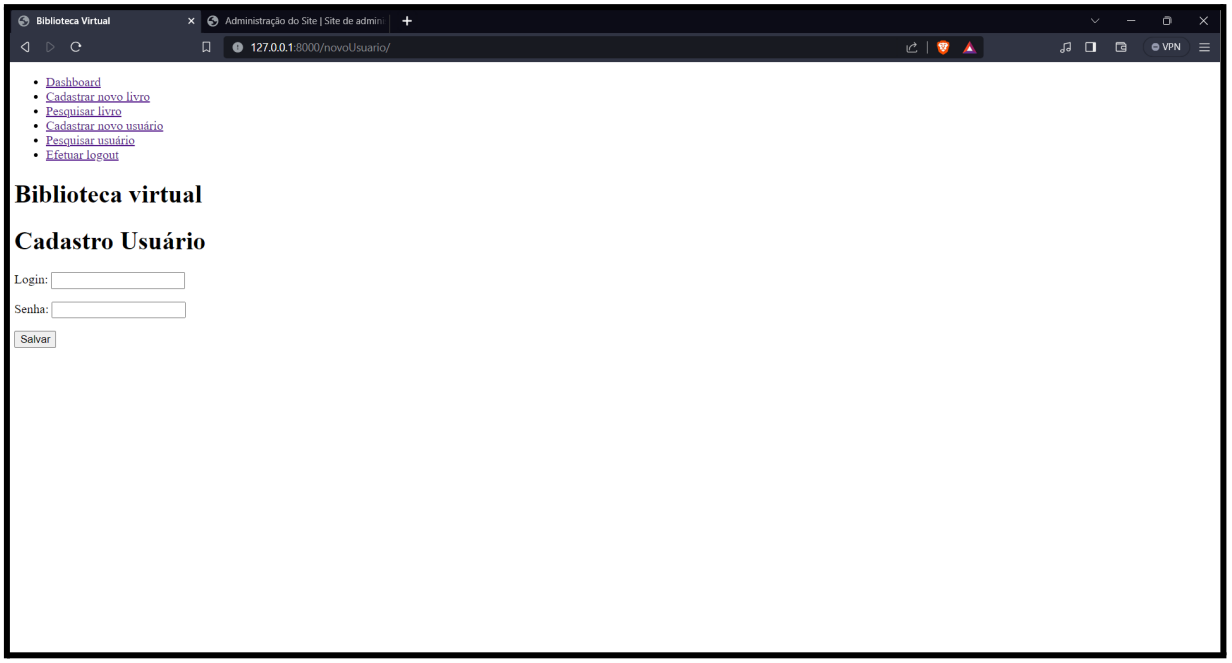
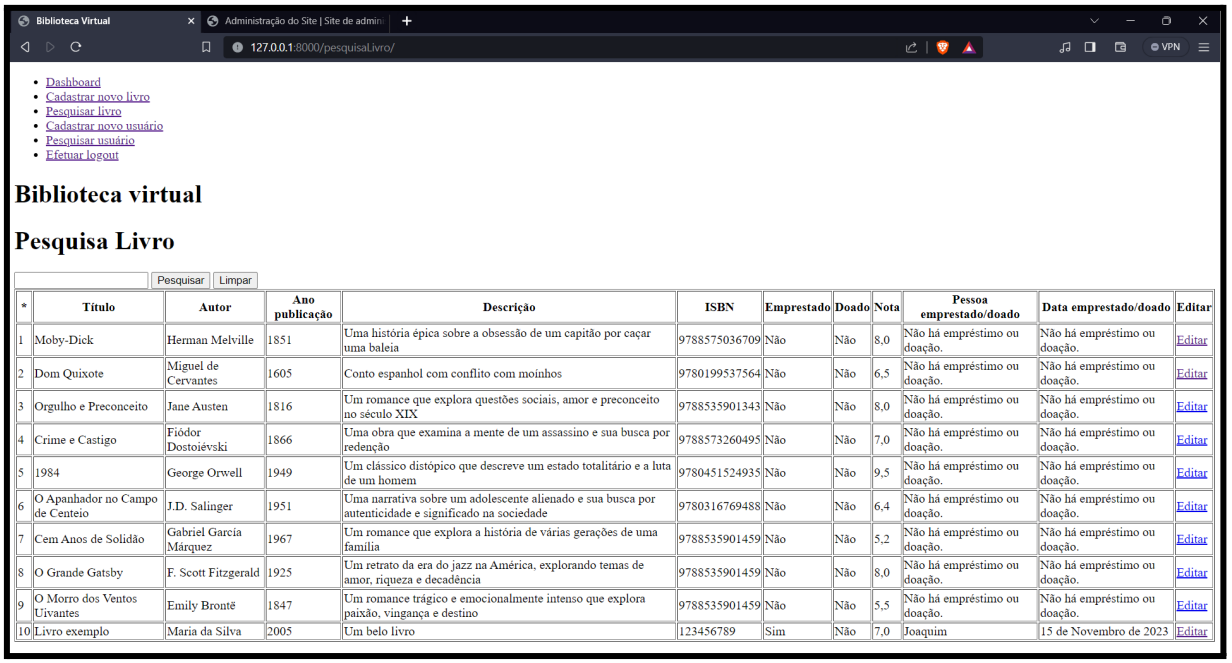


Figura 24 - Página de cadastro do usuário

Fonte: O Autor

4.1.4.7.8 Tela pesquisa livro

Nesta tela, é possível visualizar os livros cadastrados no sistema, filtrar o livro pelo título e ir para uma página de edição do registro ao clicar no link Editar na linha referente ao registro.



*	Título	Autor	Ano publicação	Descrição	ISBN	Emprestado	Doado	Nota	Pessoa emprestado/doado	Data emprestado/doado	Editar
1	Moby-Dick	Herman Melville	1851	Uma história épica sobre a obsessão de um capitão por caçar uma baleia	9788575036709	Não	Não	8,0	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
2	Dom Quixote	Miguel de Cervantes	1605	Conto espanhol com conflito com moinhos	9780199537564	Não	Não	6,5	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
3	Orgulho e Preconceito	Jane Austen	1816	Um romance que explora questões sociais, amor e preconceito no século XIX	9788535901343	Não	Não	8,0	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
4	Crime e Castigo	Fiódor Dostoiévski	1866	Uma obra que examina a mente de um assassino e sua busca por redenção	9788573260495	Não	Não	7,0	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
5	1984	George Orwell	1949	Um clássico distópico que descreve um estado totalitário e a luta de um homem	9780451524935	Não	Não	9,5	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
6	O Apanhador no Campo de Centeio	J.D. Salinger	1951	Uma narrativa sobre um adolescente alienado e sua busca por autenticidade e significado na sociedade	9780316769488	Não	Não	6,4	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
7	Cem Anos de Solidão	Gabriel García Márquez	1967	Um romance que explora a história de várias gerações de uma família	9788535901459	Não	Não	5,2	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
8	O Grande Gatsby	F. Scott Fitzgerald	1925	Um retrato da era do jazz na América, explorando temas de amor, riqueza e decadência	9788535901459	Não	Não	8,0	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
9	O Morro dos Ventos Uivantes	Emily Brontë	1847	Um romance trágico e emocionalmente intenso que explora paixão, vingança e destino	9788535901459	Não	Não	5,5	Não há empréstimo ou doação.	Não há empréstimo ou doação.	Editar
10	Livro exemplo	Maria da Silva	2005	Um belo livro	123456789	Sim	Não	7,0	Joaquim	15 de Novembro de 2023	Editar

Figura 25 - Página de pesquisa de livros

Fonte: O Autor

4.1.4.7.9 Tela editar livro

Chega-se nesta dela ao clicar no link “Editar” no registro selecionado na página de pesquisa de livros. Se a edição for executada com sucesso, a página é redirecionada para a página de pesquisa de livros. Aqui é possível também redefinir as flags que indicam se o livro foi emprestado ou doado.

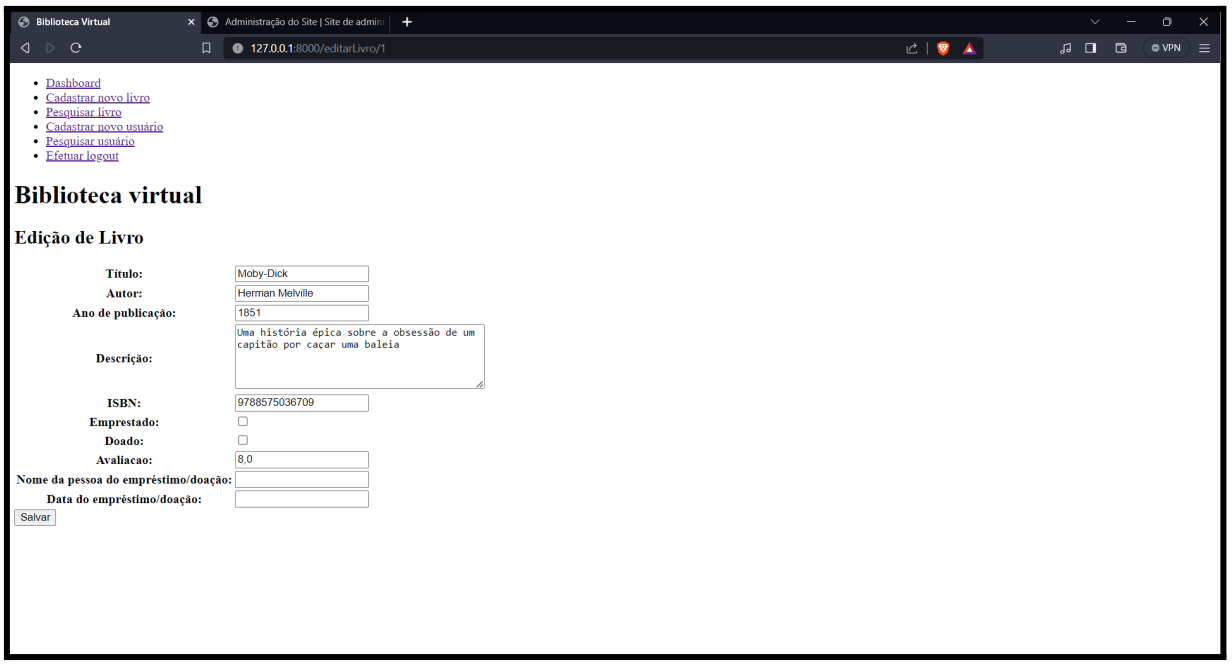


Figura 26 - Página de edição de livros

Fonte: O Autor

4.1.4.7.10 Tela novo livro

Esta é a tela de cadastro de livros no sistema.

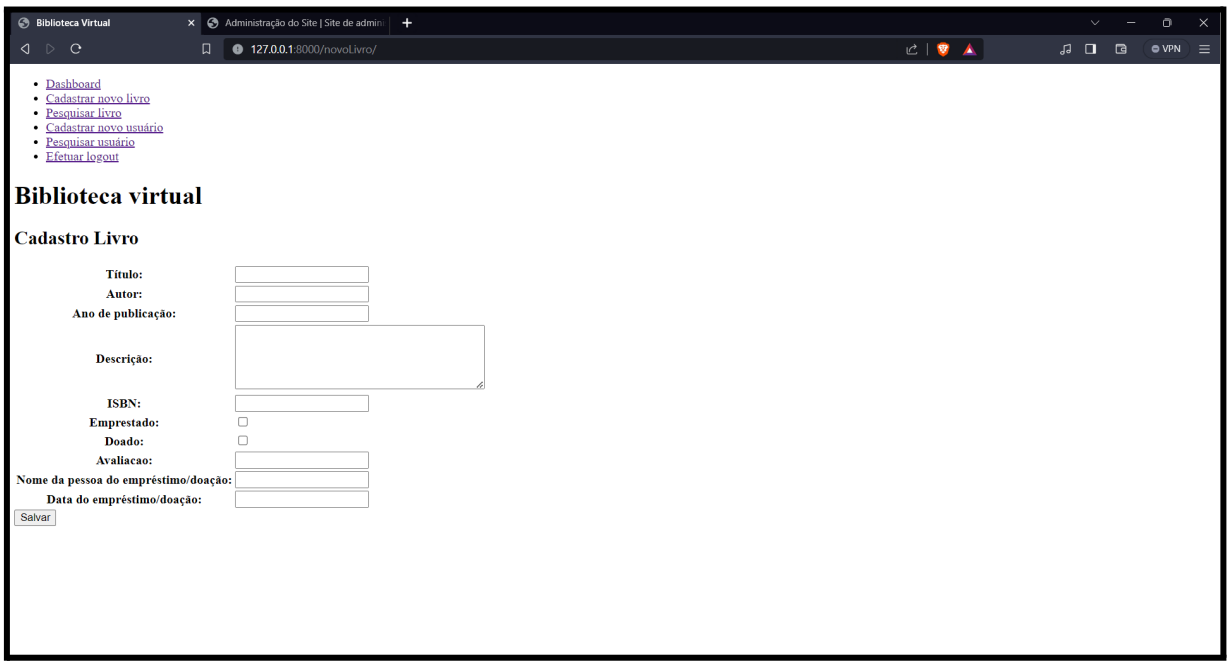


Figura 27 - Página de cadastro de livros

Fonte: O Autor

4.1.4.8 Hospedagem

Por causa de restrições de tempo, o projeto roda somente em localhost e não em nenhum servidor externo. Por isso, maiores testes referentes ao comportamento do sistema como aplicativo nativo Android não são possíveis.

5 CONCLUSÃO E TRABALHOS FUTUROS

Em relação aos objetivos de realizar um levantamento de requisitos, projetar e implementar a arquitetura de um sistema web e desenvolver um banco de dados para armazenar informações dos livros, usuários e empréstimos, pode-se dizer que foram concluídos com êxito. Mas deve-se ressaltar que a garantia de compatibilidade com dispositivos móveis com a tecnologia pwa não pode ser avaliada, uma vez que a hospedagem do projeto em um servidor externo e conexão https são elementos necessários para tal.

Com relação aos requisitos mencionados no capítulo 4, a maioria deles foi atendida pelo projeto. Porém há dois funcionais e um não funcional em que dificuldades foram encontradas ou não foram concluídos. E esses são tratados a seguir. Primeiro, os requisitos funcionais, o 4, sobre o compartilhamento das coleções de livros entre usuários e o 5, sobre a possibilidade de extração de informações de livros a partir do código de barras, a limitação do tempo de desenvolvimento disponível impediu seus tratamentos. Segundo, o requisito não funcional 2, sobre a adoção de PWA para obter a responsividade do sistema em ambos ambientes, desktop e mobile. Neste caso, também por restrições do tempo de desenvolvimento disponível, optou-se por executar a aplicação apenas em localhost que, como já mencionado, impede a verificação das principais características da abordagem PWA, a saber, responsividade e instalação fazendo parecer aplicação nativa.

Como trabalhos futuros para sanar limitações deste trabalho e prover melhoria deste trabalho, é proposta a utilização de alguma API de uso aberto para captar informações de livros a partir do ISBN, como por exemplo o google books API [23], com respeito a possibilidade de compartilhamento de bibliotecas entre usuários sugere-se a criação de um sistema de permissões para os usuários. Com relação à abordagem PWA, para se obter responsividade, propõe-se a utilização do framework bootstrap do css [24], e para a validação e teste do comportamento nativo num celular, como instalação e notificações, é proposta a hospedagem num servidor externo, possibilitando o uso do protocolo TLS/SSL sobre o HTTP. Outra vantagem da hospedagem num servidor externo seria a possibilidade de testar o programa em produção com usuários, o que também é uma necessidade a ser tratada em esforços futuros.

REFERÊNCIAS

[1] Ranganathan, S. R. (1931). **The Five Laws of Library Science**. Madras Library Association.

[2] Van House, N. A., Davis, M., Ames, M., Finn, M., & Viswanathan, V. (2006). **The Uses of Personal Networked Digital Imaging**: An Empirical Study of Cameraphone Photos and Sharing. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06).

[3] Borgman, C. L. (2000). **From Gutenberg to the Global Information Infrastructure**: Access to Information in the Networked World. MIT Press.

[4] Forshaw, A. (2008). **Django Web Development with Python**. Packt Publishing.

[5] Ronacher, A. (2010). **Flask Web Development**. O'Reilly Media.

[6] Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). **The NumPy Array**: A Structure for Efficient Numerical Computation. Computing in Science & Engineering.

[7] Holovaty, A., & Kaplan-Moss, J. (2005). **The Django Philosophy.** https://docs.djangoproject.com/en/stable/misc/design-philosophies/

[8] Flanagan, D., & Flask, A. (2018). **Web Development with Django Cookbook**. Packt Publishing.

[9] Roy, A. (2014). **Django by Example**. Packt Publishing.

[10] Vanderkam, D. (2018). **Django Security Releases**: How and Why. https://adamj.eu/tech/2020/07/22/django-security-releases-how-and-why/

[11] Russell, Alex; Berriman, Frances. "**Progressive Web Apps: Escaping Tabs Without Losing Our Soul**." , https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/

[12] TANDEL, Sayali Sunil; JAMADAR, Abhishek. **Impact of Progressive Web Apps on Web App Development**. International Journal of Innovative Research in Science,

Engineering and Technology, Maharashtra, India, v. 7, n. 9, p. 9439-9444, set. 2018. DOI 10.15680/IJIRSET.2018.0709021. Disponível em: https://www.researchgate.net/profile/Sayali-Tandel-2/publication/330834334_Impact_of_Progressive_Web_Apps_on_Web_App_Development/links/5c5605d3a6fdccd6b5dde018/Impact-of-Progressive-Web-Apps-on-Web-App-Development.pdf. Acesso em: 02 dez. 2023.

[13] Bastos da Cunha, Murilo. **Desafios na construção de uma biblioteca digital**. Ciência da Informação, Brasília, v. 28, n. 3, p. 309-318, set./dez. 1999. DOI: 10.1590/S0100-19651999000300003.

[14] John Tonding, Fabiana. Andréa de Souza Vanz,Samile. **Plataformas de Serviços de Bibliotecas: a evolução dos sistemas para gerenciamento de bibliotecas**. Perspectivas em Ciência da Informação, v. 23, n. 04, out./dez. 2018. DOI: 10.1590/1981-5344/3302

[15] EXPLORING TOPICS. **Internet Traffic from Mobile Devices**. Disponível em: https://explodingtopics.com/blog/mobile-internet-traffic. Acesso em: 12 dez 2023.

[16] Google Developers. **Service Workers: an Introduction**. Disponível em: https://developers.google.com/web/fundamentals/primers/service-workers. Acesso em: 12 dez 2023.

[17] Google Developers. **The Web App Manifest** Disponível em: https://developers.google.com/web/fundamentals/web-app-manifest. Acesso em 12 dez 2023

[18] Goodreads. Disponível em: https://www.goodreads.com/. Acesso em: 13 dez 2023.

[19] Librarything. Disponível em: https://www.librarything.com/. Acesso em: 13 dez 2023

[21] Business Insider, **What is goodreads** https://www.businessinsider.com/guides/tech/what-is-goodreads. Acesso em: 13 dez 2023

[22] Washingtonpost. **Beyond Goodreads: Four tools that help readers track their books** https://www.washingtonpost.com/entertainment/books/reading-tools-book-recommendations/2021/07/09/cd64ca1e-e0d5-11eb-b507-697762d090dd_story.html. Acesso em : 13 dez 2023

[23] Google developers https://developers.google.com/books/docs/v1/using?hl=pt-br Acesso em: 14 dez 2023

[24] Getootstrap. https://getbootstrap.com/ Acesso em: 15 dez 2023

APÊNDICE A – CÓDIGO DO PROJETO

O código do projeto está disponível no repositório no link: https://github.com/Zarbatoa/SistemaBibliotecaVirtualTCC

APÊNDICE B - ARTIGO NO FORMATO SBC

Implementação de uma Biblioteca Virtual com Progressive Web Application

Lucas Pereira Zarbato[1], Ricardo Pereira e Silva[1]

[1]Departamento de Informática e Estatística (INE) - Universidade Federal de Santa Catarina (UFSC) Caixa postal: 476 - CEP: 88040-370 - Campus Universitário Trindade, Sala 102 Florianópolis, SC - Brasil

lpzarbato@gmail.com, ricardo.silva@ufsc.br

Abstract. This work proposes the implementation of a Virtual Library using Progressive Web App (PWA) as a technological solution. The research covers the development of a responsive web platform, accessible through mobile devices and desktops, incorporating PWA functionalities to offer a user experience close to that of native applications, such as offline access, push notifications and the ability to install on the home screen. Among the benefits of this approach, the accessibility, responsiveness and advanced features provided by the combination of a virtual library with PWA technology stand out.

Resumo. Este trabalho propõe a implementação de uma Biblioteca Virtual utilizando Progressive Web App (PWA) como solução tecnológica. A pesquisa abrange o desenvolvimento de uma plataforma web responsiva, acessível através de dispositivos móveis e desktops, incorporando funcionalidades de PWA para oferecer uma experiência de usuário próxima à de aplicativos nativos, como o acesso offline, notificações push e capacidade de instalação na tela inicial. Dentre os benefícios dessa abordagem, destacam-se a acessibilidade, responsividade e recursos avançados proporcionados pela combinação de uma biblioteca virtual com tecnologia PWA.

1. Introdução

A gestão eficiente de uma biblioteca pessoal pode ser um desafio, especialmente para entusiastas da leitura que acumulam uma coleção considerável de livros ao longo do tempo. A ausência de um sistema organizado pode resultar em dificuldades no rastreamento de livros [Ranganathan 1931], controle de empréstimos e na manutenção de um acervo atualizado. Diante desse cenário, surge a necessidade de

uma solução prática e acessível que atenda às demandas específicas de uma biblioteca particular.

O tradicional método de catalogação manual torna-se cada vez mais impraticável diante do crescente número de títulos e da diversidade de formatos, incluindo livros físicos e digitais [Bastos da Cunha 1999]. Neste contexto, a implementação de um Sistema Web Progressivo (PWA) surge como uma resposta eficaz para proporcionar uma gestão intuitiva e móvel dessa biblioteca pessoal.

Ao contrário das soluções existentes, a proposta deste projeto não apenas busca automatizar o processo de organização, mas também visa oferecer uma experiência de usuário otimizada em dispositivos móveis. A escolha da abordagem PWA não só permite o acesso offline ao acervo, mas também promove a acessibilidade, permitindo que os usuários desfrutem de funcionalidades essenciais em qualquer lugar, a qualquer momento.

Em suma, o desenvolvimento deste software busca oferecer uma solução prática e inovadora para entusiastas da leitura, simplificando a gestão de suas bibliotecas pessoais e proporcionando uma experiência aprimorada de interação com sua coleção de livros.

2. Fundamentação

O projeto foi desenvolvido em Python, uma linguagem de programação de alto nível, que tem se destacado como uma escolha popular e versátil em diversos domínios de desenvolvimento de software. Van Rossum, o criador de Python, concebeu-a com princípios como legibilidade e simplicidade, resultando em um código que é fácil de aprender e manter. A comunidade de desenvolvedores se beneficiou da abordagem "batteries included" de Python, fornecendo uma ampla biblioteca padrão que abrange desde manipulação de dados até interfaces gráficas.

Django, um framework web em Python, foi desenvolvido com a missão de tornar o desenvolvimento web eficiente e pragmático. O "Django Philosophy" [Holovaty and Kaplan-Moss 2005] destaca princípios fundamentais como "Don't Repeat Yourself" (DRY) e "Convention Over Configuration", promovendo uma abordagem estruturada e sem redundâncias no desenvolvimento de aplicativos web.

Para a construção de interfaces de usuário, o Django utiliza o sistema de templates, facilitando a criação de páginas dinâmicas e a integração com dados provenientes do back-end. Além disso, a estrutura modular do Django, com seu sistema

de aplicativos reutilizáveis, incentiva a criação de componentes independentes e extensíveis [Roy 2014].

2.1. Progressive Web Applications

O conceito de Progressive Web Apps (PWAs) surgiu em 2015, proposto pelo engenheiro do Google, Alex Russell, e pelo designer Frances Berriman. O termo foi cunhado em um artigo detalhado que descreveu a ideia de aplicativos web que ofereciam uma experiência próxima à de aplicativos nativos em dispositivos móveis, por meio da utilização de tecnologias web progressivas.

Russell e Berriman [Russel and Berriman] destacaram a necessidade de criar aplicativos web que fossem confiáveis, rápidos e engajadores, mesmo em condições de conectividade limitada. A proposta foi impulsionada pela evolução das capacidades dos navegadores e pela demanda crescente por experiências de aplicativos móveis sem a necessidade de instalação a partir de lojas de aplicativos tradicionais.

Essa abordagem inovadora visava combinar o melhor dos aplicativos nativos, como acesso offline, notificações e experiência de usuário fluida, com a acessibilidade e facilidade de uso dos aplicativos da web. O conceito rapidamente ganhou destaque, resultando em iniciativas e padrões da indústria para impulsionar o desenvolvimento e a adoção de PWAs.

3. Estado da Arte

O Goodreads [Goodreads] é uma plataforma social voltada para leitores que oferece recursos para catalogar, avaliar e recomendar livros. Em 2013, a Amazon, uma das maiores varejistas online do mundo, adquiriu o Goodreads, mantendo-o como uma entidade separada.

LibraryThing [Librarything] é uma plataforma online dedicada à catalogação de livros e interações sociais relacionadas a eles. Criado em 2005, oferece aos usuários a capacidade de catalogar suas bibliotecas pessoais, fornecendo informações detalhadas sobre livros, incluindo títulos, autores, resumos, classificações e resenhas.

O Goodreads destaca-se como uma plataforma social popular entre os leitores, oferecendo a catalogação de livros, resenhas, recomendações e interações entre usuários. Sua grande comunidade e recomendações personalizadas baseadas no histórico de leitura são pontos fortes. No entanto, por vezes, pode priorizar a interação social em detrimento da experiência de construção de uma biblioteca personalizada.

Por outro lado, o LibraryThing é reconhecido por suas ferramentas avançadas de catalogação e organização, além de uma comunidade ativa. Foca na precisão da

catalogação, mas pode oferecer menos personalização na construção de uma biblioteca virtual expansível.

O software proposto por este trabalho visa criar uma plataforma que permita o gerenciamento de coleções pessoais, acesso fácil a informações sobre livros e uma experiência otimizada para dispositivos móveis. A solução visa oferecer uma experiência mais personalizada, ao mesmo tempo em que incorpora recursos de catalogação e acessibilidade offline proporcionados por um PWA.

4. Desenvolvimento

4.1. Requisitos do sistema

O passo inicial para o desenvolvimento de um programa é o levantamento dos requisitos funcionais e não funcionais. Com os requisitos coletados há maior facilidade em definir as funcionalidades esperadas do programa. A Tabela 1 define os requisitos funcionais e não funcionais levantados

Identificação	**Descrição**
RF-01	O programa deve permitir o início de sessão para cada usuário (Log-in e log-out)
RF-02	O programa deve manter informações sobre os usuários do sistema (criar, pesquisar, editar, atualizar e listar os dados referentes).
RF-03	O programa deve manter informações sobre livros, inseridos pelos usuários (criar, pesquisar, editar, atualizar e listar os dados referentes).
RF-04	O programa deve permitir um usuário compartilhar sua biblioteca/ lista de livros com outro usuário
RF-05	O programa deve possibilitar a extração de informações do livro pela imagem ou número do código de barras
RNF-01	O programa será construído para rodar em ambiente web, compatível com os principais navegadores disponíveis no mercado: Microsoft Edge, Google Chrome, Firefox.
RNF-02	Será utilizado práticas de PWA para que o programa funcione de forma responsiva tanto por dispositivos mobile quanto por notebook e desktop

Tabela 1 - Requisitos funcionais e não-funcionais do sistema

4.2. Dicionário de dados

Foi definido também, no projeto, um dicionário de dados. Ou seja, uma breve descrição exploratória dos dados, se tal dado é obrigatório ou somente desejável e o tipo de dado. Essas informações foram definidas para cada atributo de cada entidade do projeto como visto na Tabela 2.

Entidade	Atributo	Descrição	Obrigatoriedade	Tipo de dado
Livro	id	Identificador único do livro	Sim	Inteiro
Livro	titulo	Título do livro	Sim	Texto
Livro	autor	Autor do livro	Sim	Texto
Livro	ano_publicacao	Ano de publicação do livro	Sim	Inteiro
Livro	descricao	Uma breve descrição fazendo um breve resumo do livro	Sim	Texto
Livro	isbn	Código ISBN do livro	Sim	Texto
Livro	emprestado	Flag indicando se o livro foi emprestado ou não	Não, mas desejável	Booleano
Livro	doado	Flag indicando se o livro foi doado ou não	Não, mas desejável	Booleano
Livro	avaliacao	Nota dada ao livro pelo usuário	Sim	Ponto Flutuante
Livro	nome_pessoa_empr	Nome da pessoa para a qual o livro foi emprestado ou doado, se for o caso	Não, mas desejável	Texto
Livro	data_emprest_doa	Data que o livro foi emprestado ou doado, se for o caso	Não, mas desejável	Data
Usuario	id	Identificador único do usuário	Sim	Inteiro
Usuario	login	Login do usuário	Sim	Texto
Usuario	senha	Senha do usuário	Sim	Texto

Tabela 2 - Dicionário de dados das entidades do projeto

4.3. Desenvolvimento do sistema

A primeira coisa que foi feita no projeto foi a criação desse ambiente virtual com o comando “python -m venv venv” e depois é feita a habilitação desse ambiente

sempre que entrar no ambiente de desenvolvimento com o comando “source Scripts/activate”.

Assim como o arquivo package.json para o Node.js e o arquivo pom.xml para o java, foi criado um arquivo neste projeto chamado “requirements.txt” que contém dependências a serem instaladas e atualizadas. Isso é feito ao abrir o projeto com o comando pip install -r requirements.txt .Neste arquivo é curto e contém as linhas: “django” e “django-pwa”. A maior parte das bibliotecas e scripts necessários para o projeto vem com a dependência ”django”.

O projeto django é iniciado com o comando django-admin, “django-admin startproject config .” e após isso todas as operações feitas no servidor django são feitas pelo script manage.py.

O termo “app livro” se refere a aplicação e entidade criada pelo comando startapp com o script manage.py, é aqui que está a definição do modelo da entidade do banco de dados referente aos livros e seus empréstimos, além de configurações da respectiva view do front-end.

A entidade livro é definida com os campos: titulo, autor, ano_publicacao, descricao, isbn, emprestado, doado, avaliacao, nome_pessoa_empr e data_emprest_doa conforme a figura 1. E a entidade usuario é definida pelos atributos login e senha como visto na figura 2. Além desses campos, também tem o campo de *pk* referente a chave primária que está definida na superclasse models.Model.

```
class Livro(models.Model):
    titulo           = models.CharField(max_length=200)
    autor            = models.CharField(max_length=100)
    ano_publicacao   = models.IntegerField()
    descricao        = models.TextField(blank=True)
    isbn             = models.CharField(max_length=30)

    emprestado       = models.BooleanField(default=False)
    doado            = models.BooleanField(default=False)
    avaliacao        = models.FloatField()

    nome_pessoa_empr  = models.CharField(max_length=250, null=True, blank=True)
    data_emprest_doa  = models.DateField(null=True, blank=True)
```

Figura 1 - Arquivo model.py referentes ao app livro

```
class Usuario(models.Model):
    login = models.CharField(max_length=120, null=False, unique=True)
    senha = models.CharField(max_length=120, null=False)
```

Figura 2 - Arquivo model.py referentes ao app livro

4.4. PWA e hospedagem

Neste projeto é utilizado a biblioteca django-pwa que dá suporte a implementação da tecnologia pwa no django e como visto anteriormente, foi adicionado em requirements.txt. Para configurá-lo é necessário adicionar 'pwa' em INSTALLED_APPS de settings.py, e adicioná-la no roteamento.

O Service Worker é uma tecnologia fundamental para o funcionamento de Progressive Web Apps (PWA), atuando como um script JavaScript executado pelo navegador em segundo plano, independentemente da página web estar aberta.O arquivo manifest.json é um arquivo de metadados essencial para PWAs, fornecendo informações sobre a aplicação para o navegador.

O django por causa da biblioteca django-pwa, gera um service worker e um manifest.json padrões e embora seja desejável personalizá-los, isso não foi feito neste projeto

E por último, anotações "{% load pwa %}" e "{% progressive_web_app_meta %}" são adicionadas a página inicial do projeto para efetuar a carga das configurações.

Com relação a hospedagem, por causa de restrições de tempo, o projeto roda somente em localhost e não em nenhum servidor externo. Por isso, maiores testes referentes ao comportamento do sistema como aplicativo nativo Android não são possíveis.

5. Conclusão

Em relação aos objetivos de realizar um levantamento de requisitos, projetar e implementar a arquitetura de um sistema web e desenvolver um banco de dados para armazenar informações dos livros, usuários e empréstimos, pode-se dizer que foram concluídos com êxito. Mas deve-se ressaltar que a garantia de compatibilidade com dispositivos móveis com a tecnologia pwa não pode ser avaliada, uma vez que a hospedagem do projeto em um servidor externo e conexão https são elementos necessários para tal.

Com relação aos requisitos mencionados no capítulo 4, a maioria deles foi atendida pelo projeto. Porém há dois funcionais e um não funcional em que dificuldades foram encontradas ou não foram concluídos. E esses são tratados a seguir. Primeiro, os requisitos funcionais, o 4, sobre o compartilhamento das coleções de livros entre usuários e o 5, sobre a possibilidade de extração de informações de livros a partir do código de barras, a limitação do tempo de desenvolvimento disponível impediu seus tratamentos. Segundo, o requisito não funcional 2, sobre a adoção de PWA para obter a responsividade do sistema em ambos ambientes, desktop e mobile. Neste caso, também por restrições do tempo de desenvolvimento disponível, optou-se por executar a aplicação apenas em localhost que, como já mencionado, impede a verificação das principais características da abordagem PWA, a saber, responsividade e instalação fazendo parecer aplicação nativa.

Como trabalhos futuros para sanar limitações deste trabalho e prover melhoria deste trabalho, é proposta a utilização de alguma API de uso aberto para captar informações de livros a partir do ISBN, como por exemplo o google books API [Google developers], com respeito a possibilidade de compartilhamento de bibliotecas entre usuários sugere-se a criação de um sistema de permissões para os usuários. Com relação à abordagem PWA, para se obter responsividade, propõe-se a utilização do framework bootstrap do css [Getbootstrap], e para a validação e teste do comportamento nativo num celular, como instalação e notificações, é proposta a hospedagem num servidor externo, possibilitando o uso do protocolo TLS/SSL sobre o HTTP. Outra vantagem da hospedagem num servidor externo seria a possibilidade de testar o programa em produção com usuários, o que também é uma necessidade a ser tratada em esforços futuros.

Referências

[Ranganathan 1931] Ranganathan, S. R. (1931). The Five Laws of Library Science. Madras Library Association.

[Bastos da Cunha 1999] Bastos da Cunha, Murilo. Desafios na construção de uma biblioteca digital. Ciência da Informação, Brasília, v. 28, n. 3, p. 309-318, set./dez. 1999. DOI: 10.1590/S0100-19651999000300003.

[Holovaty and Kaplan-Moss 2005] Holovaty, A., & Kaplan-Moss, J. (2005). The Django Philosophy.

[Roy 2014] Roy, A. (2014). Django by Example. Packt Publishing.

[Russel and Berriman] Russell, Alex; Berriman, Frances. "Progressive Web Apps: Escaping Tabs Without Losing Our Soul."

[Goodreads] Goodreads.

[Librarything] Librarything.

[Google developers] Google developers. Google books API

[Getbootstrap] Getootstrap.