

<p>Elementos do Relatório do Projeto2024-2</p> <p>Disciplina: Tópico em Engenharia de Software</p> <p>Projetando Linhas de Produto de Software</p>

1 - Enunciado do Projeto.

Tendo em mente o desenvolvimento de microsserviços para a web, o projeto proposto foi o e-commerce de livros, onde o usuário pode acessar o acervo de livros, mesmo estando deslogado, registrar-se, entrar, adicionar ao seu carrinho de compra, fazer seu pedido e constatar os pedidos feitos pela plataforma. Esse sistema, contou ainda com um sistema de recomendações cujo funcionamento se baseia nos livros adicionados ao carrinho como também os pedidos que foram realizados.

2 - Metodologia Ágil escolhida:

A Metodologia Ágil abordada foi a Extreme Programming(XP) e princípios de DevOps, partindo pelos fatores de desenvolvimento incremental, iterativo(voltado a testes(TDD) e modular, focada na escalabilidade e manutenção.

4 - Descrever o desenvolvimento do projeto segundo a Metodologia Ágil as Técnicas de Reuso de Software escolhidas.

O desenvolvimento do projeto foi conduzido segundo os princípios de um fluxo semelhante ao praticado em **Extreme Programming (XP)**, com foco em **processos incrementais e iterativos**, priorizando a qualidade do software, a modularidade da arquitetura e a validação contínua de funcionalidades, além de ser marcado por ter sido adotado ciclos curtos e intensivos de desenvolvimento. Cada componente do sistema foi implementado como um **microsserviço independente**, permitindo o desenvolvimento isolado, testabilidade individual e reusabilidade em diferentes contextos da aplicação.

5 - Colocar o Product Backlog, o Sprint Backlog para cada Sprint.

Uso do Super Productivity usando Kanban e Matriz de Eisenhower, entretanto, não foi feita a captura da tela. Logo, o único meio de comprovação é a exportação de log que segue abaixo. As atividades foram feitas das seguintes formas:

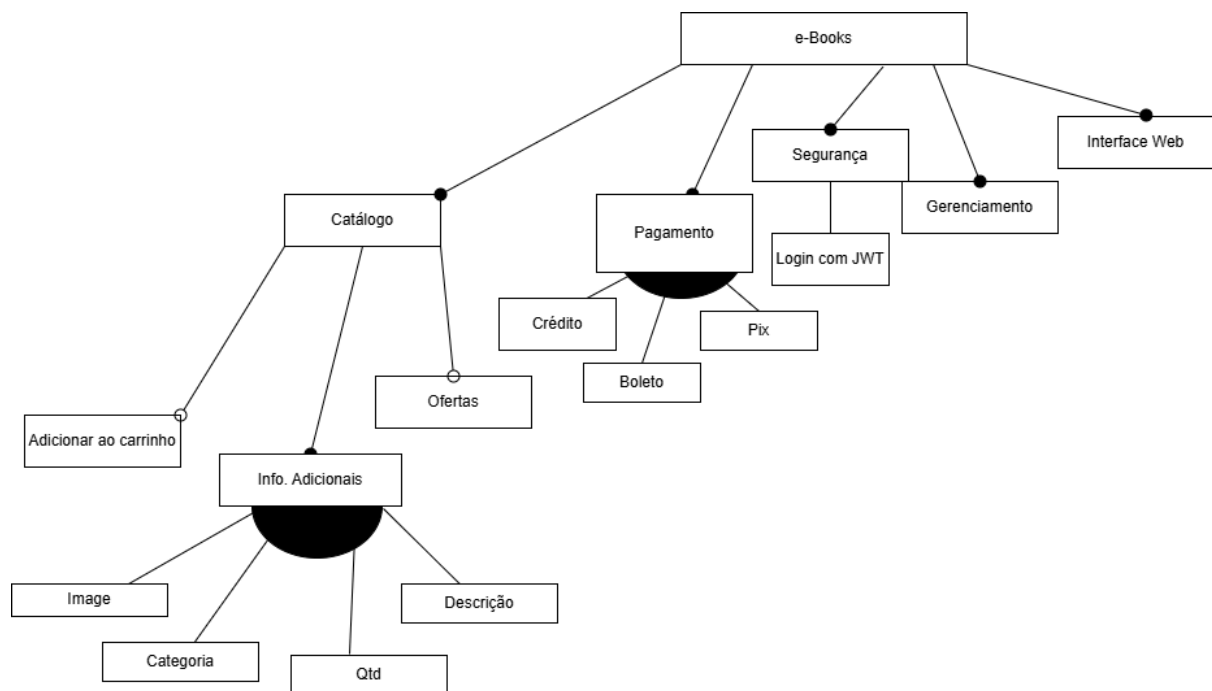
- 1 parte: Exploração de estrutura de projetos e linguagens(Django, Python, Flask, FastAPI).
- 2 parte: Back-end com a criação de cada microsserviços de Autenticação, Livros, etc.
- 3 parte: Front-end com a implementação de comunicação entre os microsserviços.

Exportação de log de trabalho 5/5/2025-5/11/2025				
Data	Começou a trabalhar	Trabalho encerrado	Hora como relógio (ex.: 5:23)	Títulos e Títulos de subtarefa
Date	Start	End	Worked	Titles
-	-	-	-	[REDACTED]
2025-05-08	-	-	1:00	Entrega do projeto funcional Eng. de Softw.
2025-05-05	-	-	1:40	[REDACTED]
2025-05-05	-	-	1:00	Olhar repositórios
-	-	-	-	Rever o repositório
2025-05-09	-	-	1:00	[REDACTED]
-	-	-	-	[REDACTED]
2025-05-05	-	-	1:00	Leitura do roteiro apresentação de Microserviços
-	-	-	-	[REDACTED]
2025-05-06 - 2025-05-08	-	-	4:00	[REDACTED]
2025-05-06	-	-	1:00	[REDACTED]
2025-05-06	-	-	1:53	Microserviços(Projeto)
-	-	-	-	[REDACTED]
-	-	-	-	Fazer relatório do Arturo

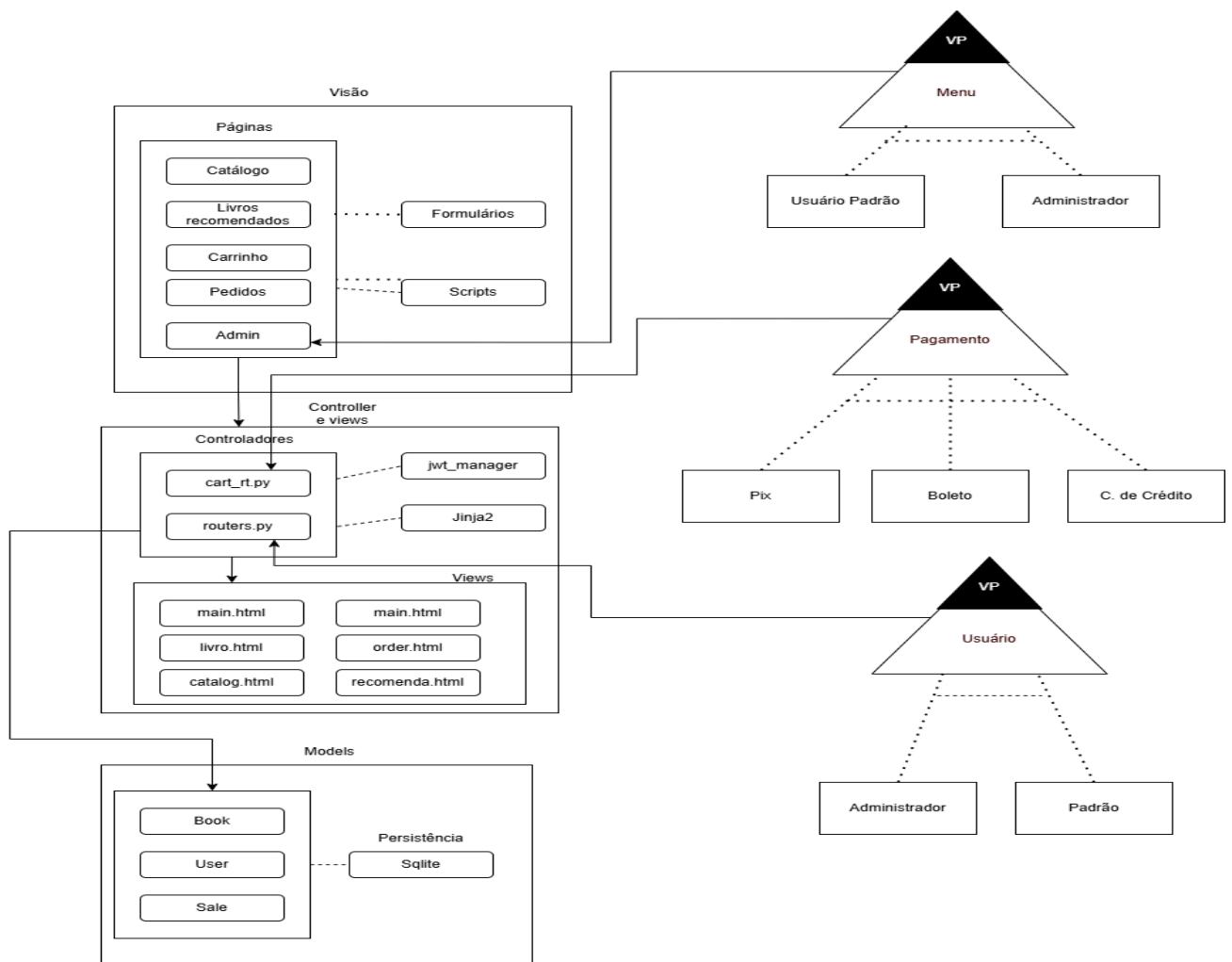
2025-05-02	-	-	1:06	Microserviços(Apresentação)
2025-05-02	-	-	0:44	Avaliar outras ferramentas que possam ser ágeis
2025-05-01	-	-	2:52	Microserviços(FastAPI, Flask)
-	-	-	-	Estudar para reav(22/05)

2025-04-07	-	-	1:50	Microservices - Atv
2025-04-07	-	-	1:25	Django

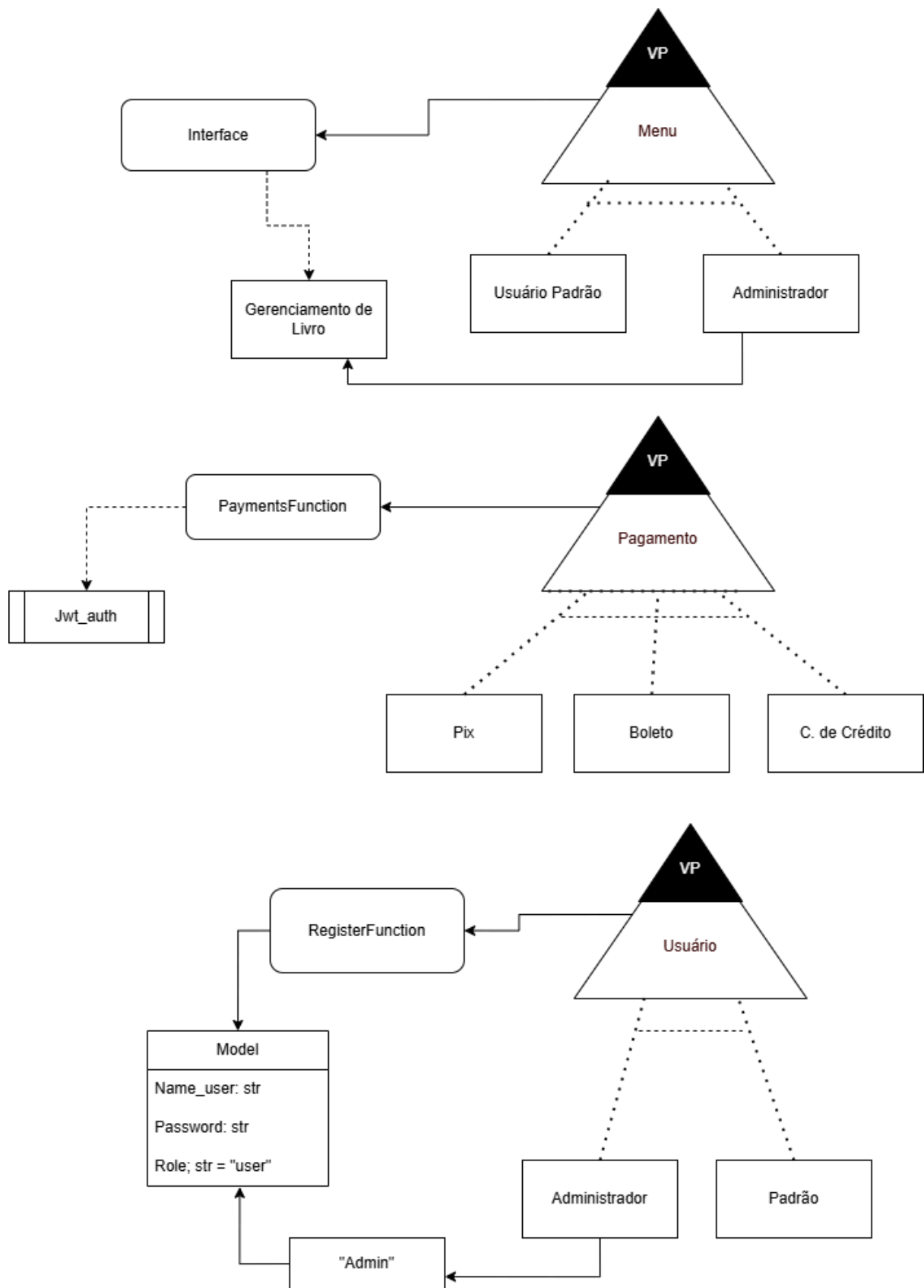
5.1-Colocar o Diagrama de Features.



5.2 - Colocar a Arquitetura em Camadas e o Diagrama de Variabilidade (pontos de variação e variantes).

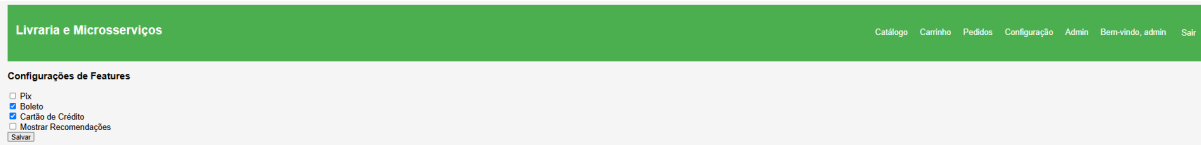


5.3 - Colocar o Diagrama de Componentes de Software e o Diagrama de Variabilidade (pontos de variação e variantes).

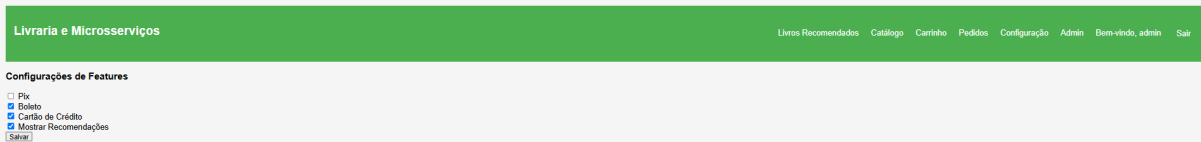


5.4 - Colocar a Tela de Configuração (seletor de features) e realizar a construção de um novo Produto.

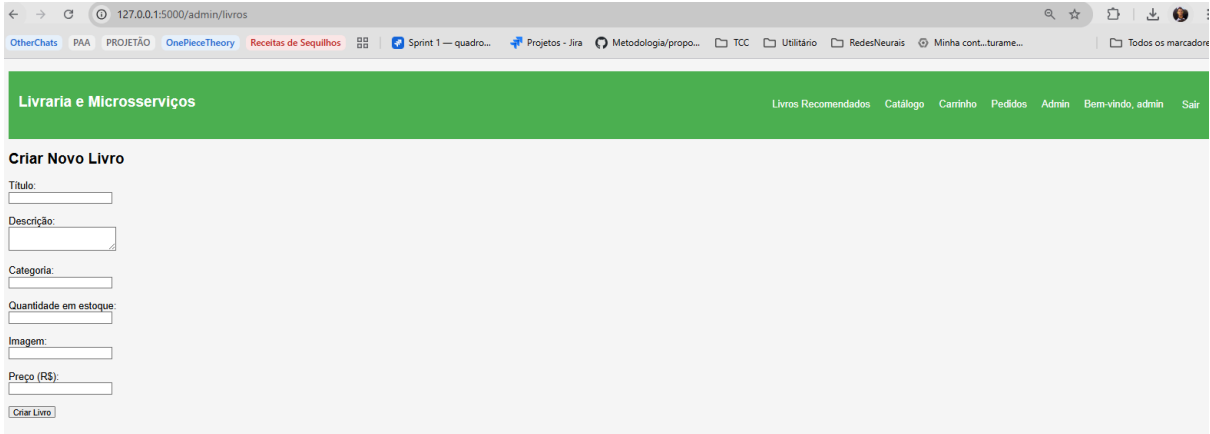
1. Tela de Seletor de Feature, onde pode ser selecionado as opções de pagamento e até o sistema de recomendação:
 - a - Sem o sistema de recomendação(Notar o navbar):



- b - com o sistema de recomendação:



2. Tela de Criação de livros:



6- Colocar a Arquitetura do sistema baseada em Microservices

- No caso do projeto utilizando e Microcroserviços:

6.1-Colocar a Arquitetura do Sistema usando Microserviços. Nesse sistema deve ser considerado o processo de vendas online, isto é:

- 1.Apresentação do catálogo dos produtos,
- 2.Escolha dos produtos e colocar no carrinho,
- 3.Calculo do Frete,
4. Forma de Pagamento, e
5. Finalizar compra.

O sistema também deve fazer recomendação de produtos considerando o perfil de compras do cliente (histórico de compras). Devem ser considerados microservices para o processo de vendas online e um microservice para fazer a recomendação de produtos.

Sistema de Livros

└─ Front-end (porta 5000)

- └─ Interface com o usuário (Jinja2 + Flask)
- └─ Renderiza templates (catálogo, carrinho, pedidos, login)
- └─ Comunicação com os microserviços via HTTP

└─ Autenticação - user_auth (porta 3000)

- Registro de Usuário
- Login com geração de JWT
- Gerenciador de permissionamento
- Validação de tokens em endpoints protegidos

- Catálogo de Livros - catalog (porta 8002)
 - Listagem de livros disponíveis
 - Consulta por ID ou filtros
 - Banco de dados: PostgreSQL (livros)

- Carrinho e Vendas - sale (porta 8003)
 - Adicionar livros no carrinho
 - Visualizar carrinho do usuário
 - Criar e listar pedidos (orders)
 - Integração com serviço de pagamento

- Pagamento - payment (porta 8001)
 - Processamento de pagamento
 - Confirmação de compra

6.2-Colocar a especificação (código documentado) da API gateway

A API Gateway usada foi uma simples baseada em Flask, responsável por:

- Repassar requisições do frontend aos microserviços de catalog e sale.
- Formatar e exibir os dados em templates Jinja2.
- Fornecer um ponto único de comunicação com cliente(front-end)

```
6 # Endereços dos microserviços de catálogo e vendas
7 API_URL_BOOKS = "http://127.0.0.1:8002"
8 API_URL_SALES = "http://127.0.0.1:8003"
```

```
18 # Página principal (catálogo de livros)
19 @router.route('/')
20 def index():
21     """
22     Rota principal que lista os livros do catálogo.
23     Consulta o microserviço de catálogo e renderiza os livros formatados.
24     """
```

```
34 # Página de registro de novo usuário
35 @router.route("/registrar")
36 def pagina_registro():
37     return render_template("registro.html")
38
39 # Página com detalhes de um livro
40 @router.route('/livros/<int:id>')
41 def livro(id):
42     """
43     Rota que exibe detalhes de um livro específico com base no ID.
44     Consulta o microserviço de catálogo.
45     """
```

```

54 # Página de recomendação
55 @router.route('/recomenda')
56 def recomenda():
57     return render_template("recomenda.html")
58
59 # Página de administração de livros
60 @router.route("/admin/livros")
61 def pagina_admin():
62     return render_template("admin/livros.html")

```

6.3-Colocar a especificação (código documentado) de cada microservice

```

└─ user_path(port: 3000):
    └─ POST /update_role
    └─ POST /register
    └─ POST /login
    └─ GET /userinfo
    └─ GET /protected

└─ payment(8001)
    └─ POST /payment

└─ payment(8002)
    └─ GET /livros
    └─ GET /livros/{id}
    └─ POST /livros
    └─ GET /livros_recomendacao/{user_id}

└─ sale(8003)
    └─ POST /sales
    └─ POST /order_pay/{user_id}
    └─ GET /get_cat/{user_id}
    └─ GET /order_pay/{user_id}
    └─ GET /cart/{useer_id}

└─ front-end(5000)
    └─ routers:
        └─ (Home) /
        └─ /registrar
        └─ /recomenda
        └─ /carrinho
        └─ /orders
        └─ /livros/<int:id>

```

6.4-Colocar a tela mostrando todos os microservices em execução

```
> python catalog
> python frontend
> python payment
> python sale
> python user_auth
```

Front-end:

```
* Running on http://192.168.0.12:5000
Press CTRL+C to quit
* Restarting with stat
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.0.12:5000
* Running on http://127.0.0.1:5000
* Running on http://192.168.0.12:5000
* Running on http://192.168.0.12:5000
```

Catálogo:

```
de\Flask_Micro\catalog> python .\main.py
INFO: Will watch for changes in these directories: [
INFO: Uvicorn running on http://127.0.0.1:8002 (Pres
INFO: Started reloader process [22000] using StatRel
INFO: Started server process [14748]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

Payment:

```
INFO: Will watch for changes in these directories: ['C:\V
INFO: Uvicorn running on http://127.0.0.1:8001 (Press CTR
INFO: Started reloader process [16116] using StatReload
INFO: Started server process [21216]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

Sale:

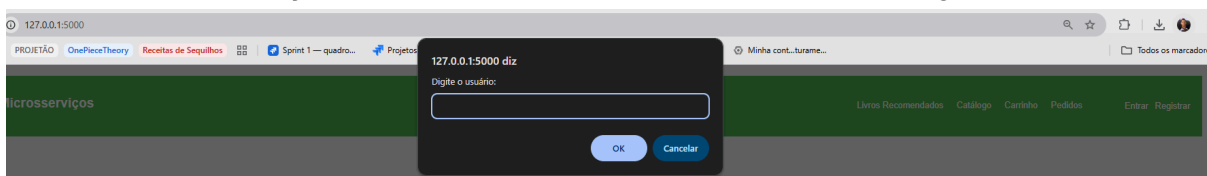
```
INFO: Will watch for changes in these directories:
INFO: Uvicorn running on http://127.0.0.1:8003 (Pre
INFO: Started reloader process [15800] using StatRe
INFO: Started server process [22436]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

User auth:

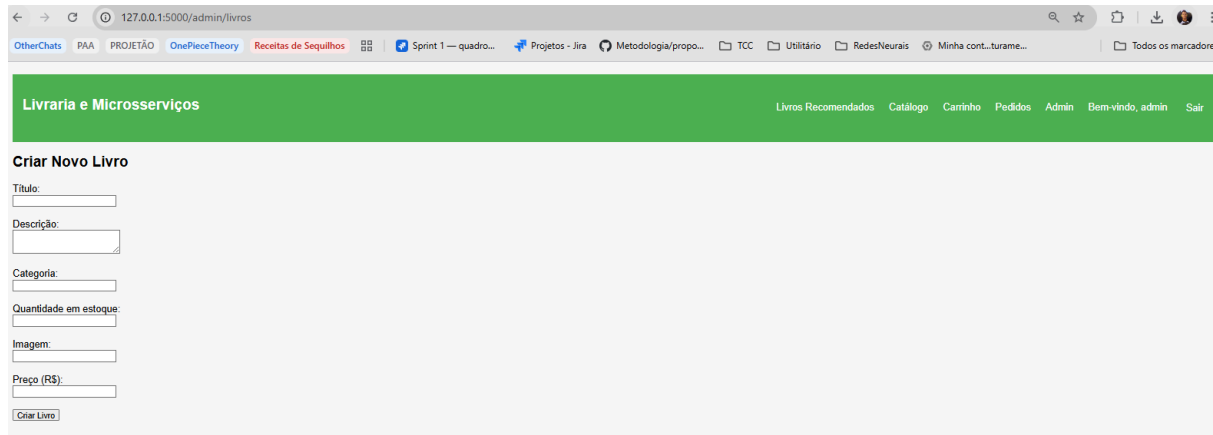
```
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:3000
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:3000
Press CTRL+C to quit
```

6.5 - Descrever, passo a passo, como construir uma Aplicação (Produto) a partir da Linha de Produto de Software construída.

Para a construção de um produto pelo front-end, é necessário logar:



Depois de logado, se o usuário tiver a regra “role”: “admin”, é possível acessar a aba Admin e criar o produto:



127.0.0.1:5000/admin/livros

OtherChats PAA PROJETO OnePieceTheory Receitas de Sequilhos Sprint 1 — quadro... Projetos - Jira Metodologia/propo... TCC Utilitário RedesNeurais Minha cont...turame... Todos os marcadores

Livraria e Microserviços Livros Recomendados Catálogo Carrinho Pedidos Admin Bem-vindo, admin Sair

Criar Novo Livro

Título:

Descrição:

Categoria:

Quantidade em estoque:

Imagem:

Preço (R\$):

Parte 2 - Implementação

7-Colocar as Tecnologias utilizadas na implementação do Sistema usando Microservices. Para cada Microserviços qual a Linguagem de Programação utilizada e qual o Banco de Dados utilizado se houver.

Foi utilizadas as seguintes ferramentas:

1. Python

- Linguagem principal usada em todos os microserviços.
- Fácil de aprender e altamente produtiva.
- Grande ecossistema de bibliotecas para web, dados e segurança

2. Flask

- Microframework usado no front.
- Simples, leve e flexível para criar rotas e lógica de apresentação.
- Usa Jinja2 para renderizar páginas HTML dinamicamente.

3. Jinja2

- Template engine utilizada com Flask.
- Permite inserir dados dinâmicos nos HTMLs (`{{ user.name }}`).
- Suporta filtros, laços, condicionais etc.

4. FastAPI

- Framework moderno, rápido e leve para APIs em Python.
- Baseado em tipagem estática (usando pydantic) para validações automáticas.
- Documentação interativa gerada automaticamente (Swagger).

5. JWT (JSON Web Token)

- Utilizado no `user_auth` para autenticação e autorização.
- Gera tokens criptografados contendo os dados do usuário.
- Permite que o sistema se mantenha estável (sem sessões armazenadas no backend).

6. SQLite

- Banco de dados relacional leve, usado localmente durante o desenvolvimento.

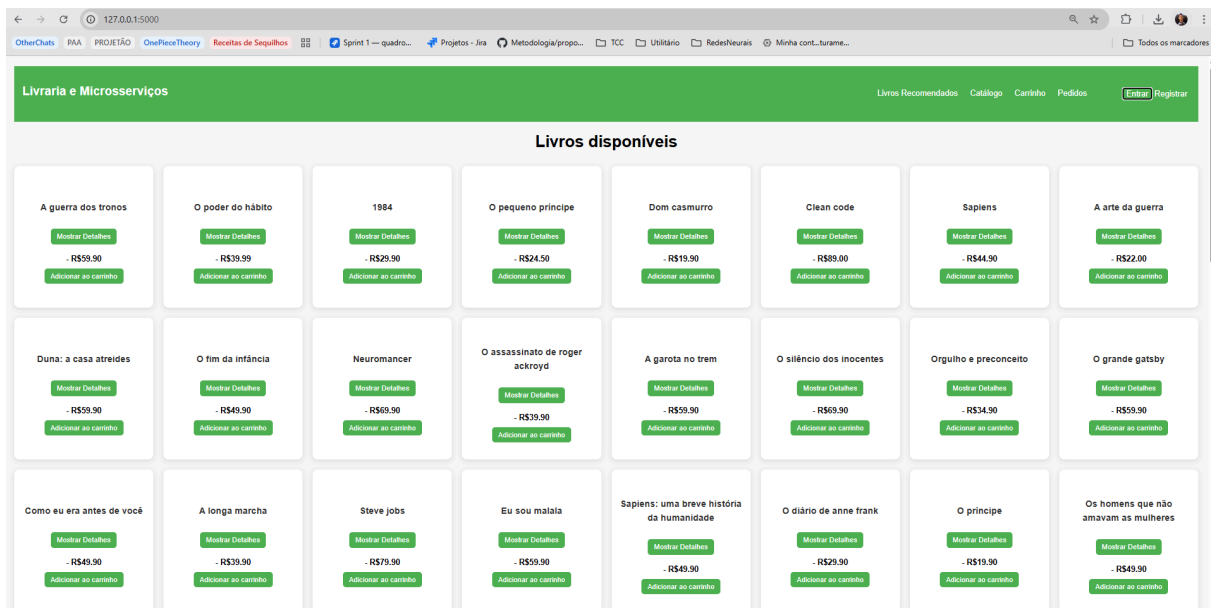
- Armazena os dados em arquivos .db.
- Ideal para testes e ambientes simples.

7. SuperProductivity e Flocos

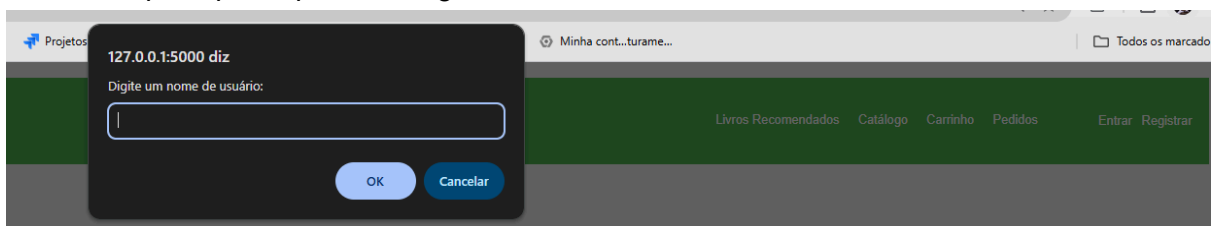
- Gerenciamento de tarefas e tempo.
- Podendo atribuir tarefas como importantes, urgentes, usando Matriz de Eisenhower e trabalhar também na forma de Kanban.

8-Mostrar e explicar passo a passo as telas do funcionamento do Produto (aplicação de e-commerce) construído. Destacar na Interface principal do seu sistema os Microserviços utilizados.

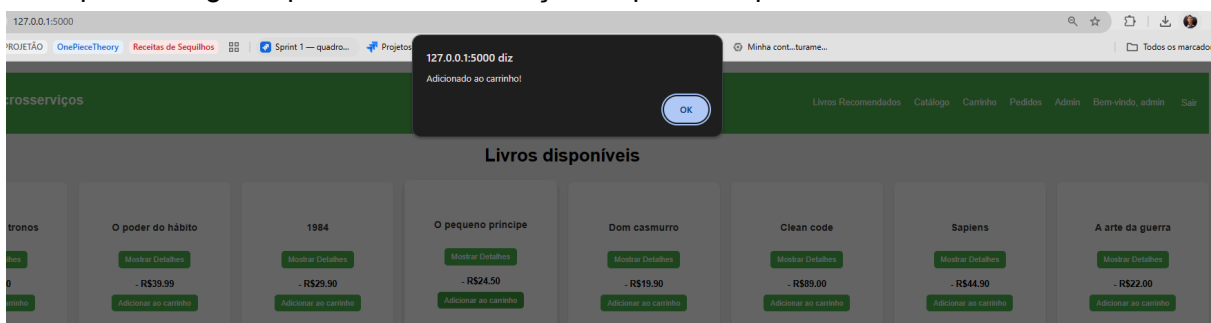
a - Tela principal onde o usuário acessa, podendo visualizar informações dos livros:



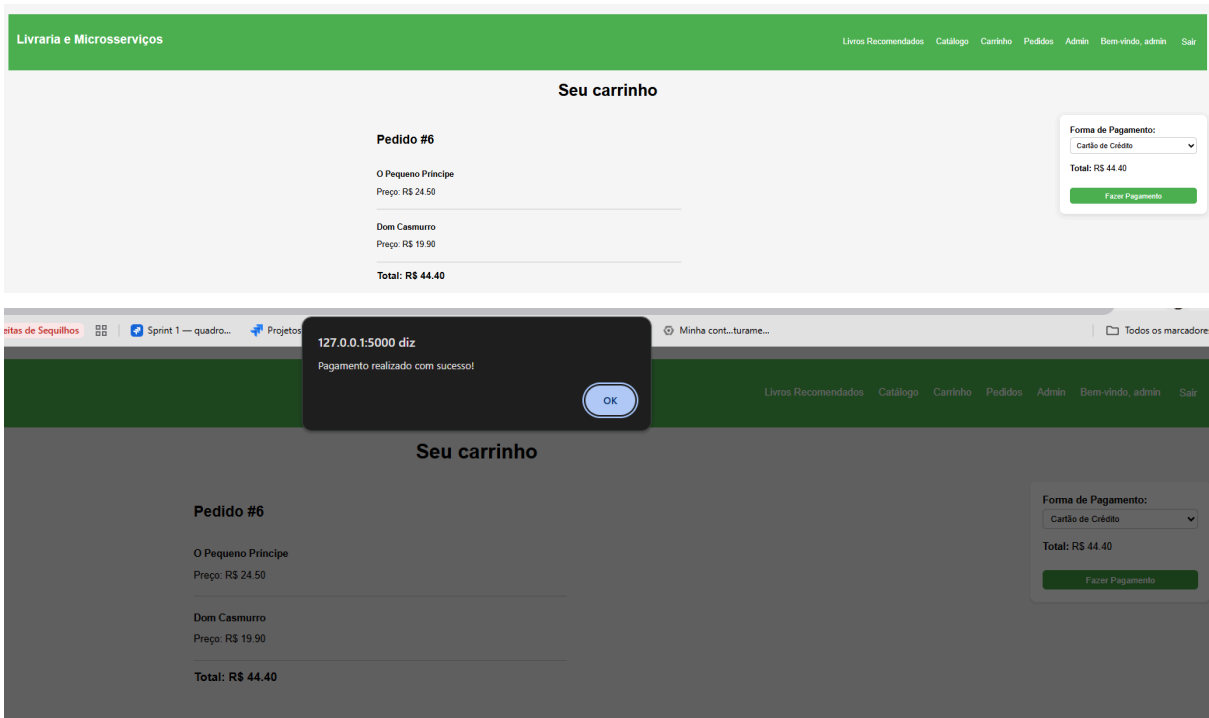
b - Na tela principal, é possível registrar-se ou entrar:



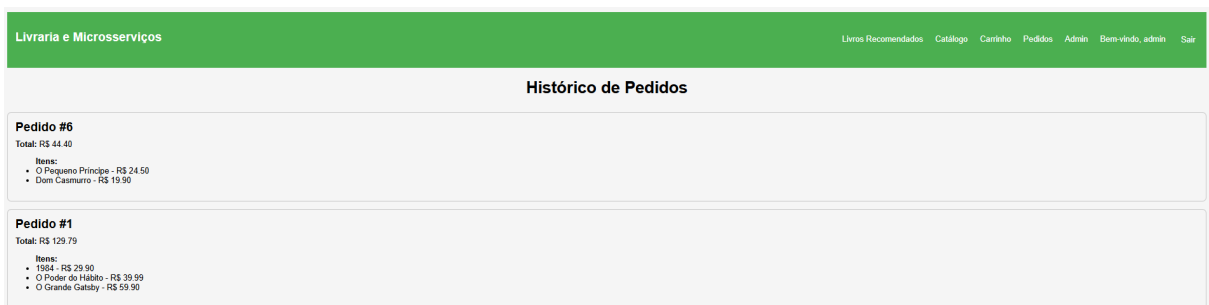
c - Depois de logado, pode ser feita a adição de produtos para o carrinho



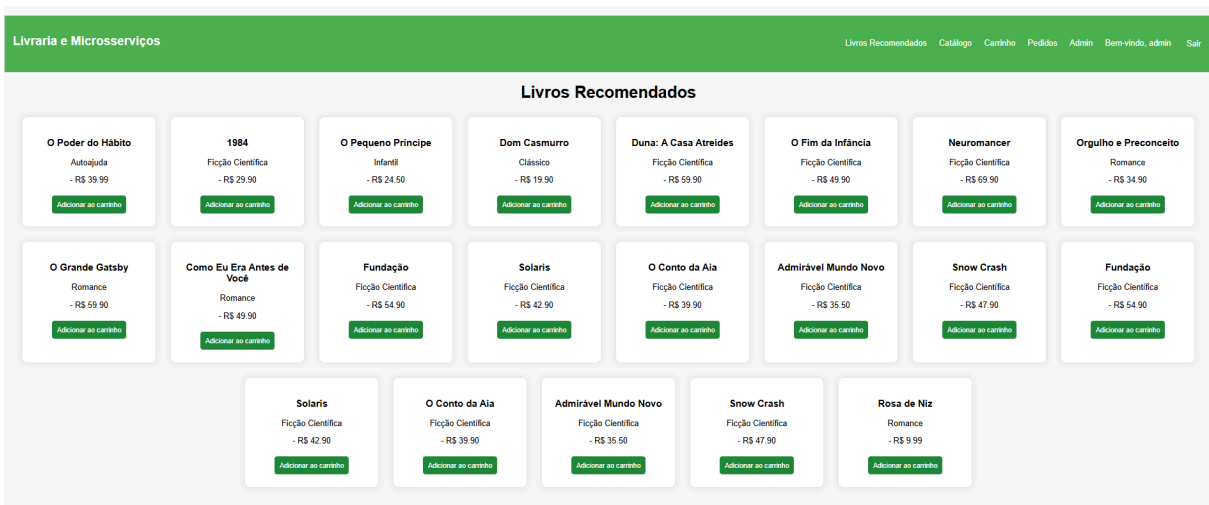
d - Acessando o carrinho, pode ver o total do carrinho e escolher a forma de pagamento:



e - Pode acessar os pedidos e visualizá-los:



f - Após acrescentar e/ou finalizar as compras, é possível acessar os Livros Recomendados com base no histórico. Esses livros são sugeridos com base nas categorias:



9-Colocar todo o código do seu projeto funcionando ok no github e colocar as explicações necessárias para a boa execução do seu sistema, e também colocar no Relatório nesta seção o link do seu projeto no github.

<https://github.com/CarlosALPessoa/E-commerce>

10-Colocar as referências bibliográficas utilizadas no desenvolvimento do seu projeto.

Flask Documentation - <https://flask.palletsprojects.com/>

FastAPI Documentation - <https://fastapi.tiangolo.com/>

PyJWT Documentation - <https://pyjwt.readthedocs.io/>

SQLite Documentation - <https://www.sqlite.org/docs.html>

Requests: HTTP for Humans - <https://requests.readthedocs.io/>

OpenAPI Specification (Swagger) - <https://swagger.io/specification/>

RESTful API Design — Microsoft -

<https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design>

TDVLE: The Dank Virtual Learning Environment - Lucas de Oliveira Amorim

Uma Linha de Produto de Software Para Sistema de Gerenciamento de Refeições - Wilamis

Micael de Araujo Aviz