Centro Universitário de Brasília – CEUB

Curso: Ciência da Computação
Disciplina: Desenvolvimento de Sistemas

Documento de Evidências
Projeto Final – HOT WHEELS

Autores: Francisco Beleza Neto; Arthur Gabriel da Silva Barbosa
Data: 15/09/2025

1. Planejamento de Atividades

Semana	Atividade	Responsável
1	Levantamento de requisitos de negócio (BRD)	Francisco / Arthur
2	Definição dos requisitos de software (ERS/SRS)	Francisco / Arthur
3	Modelagem inicial (casos de uso, classes, DER)	Equipe
4	Configuração do ambiente (Docker + DB)	Arthur
5	Implementação do login e autenticação segura	Francisco
6	Implementação do CRUD de Clientes	Francisco
7	Implementação do CRUD de Produtos	Arthur

Semana	Atividade	Responsável
8	Implementação do CRUD de Vendas	Arthur
9	Integração do Swagger + Testes automatizados	Equipe
10	Documentação final (Readme + Evidências)	Francisco
11	Preparação da apresentação em sala	Equipe

2. Diagramas

Figura 1 – Diagrama de Casos de Uso do sistema Hot Wheels.

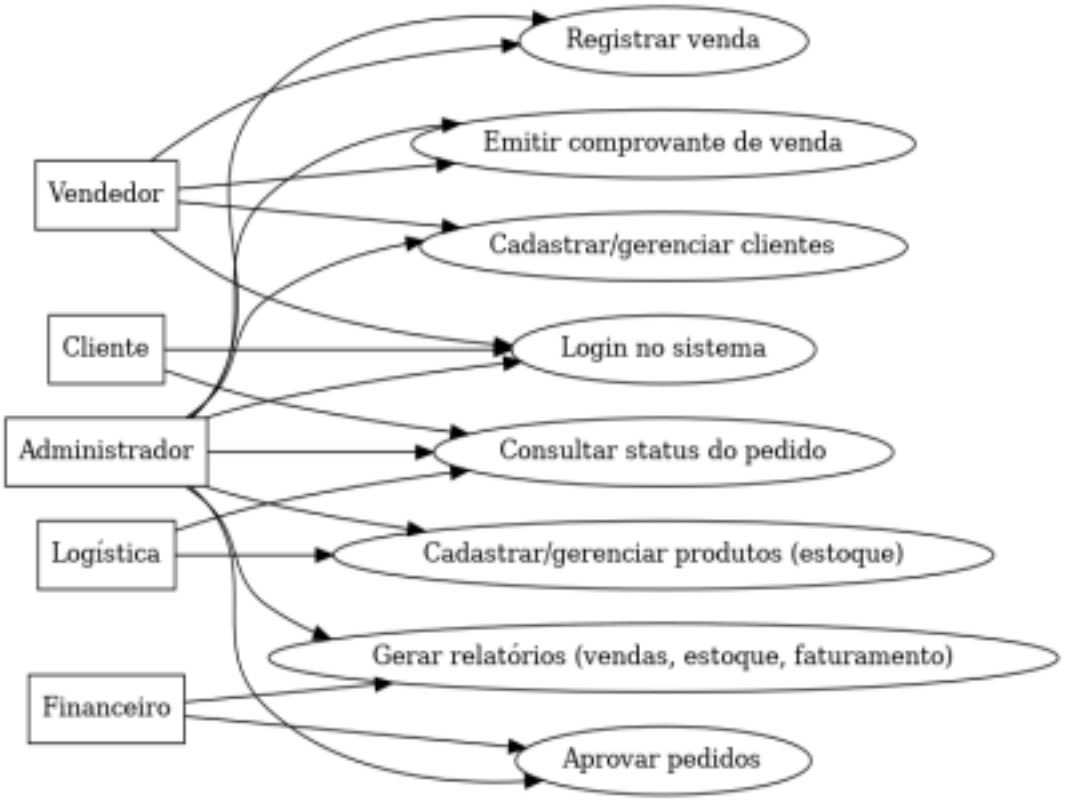


Figura 2 – Diagrama de Classes do sistema Hot Wheels.

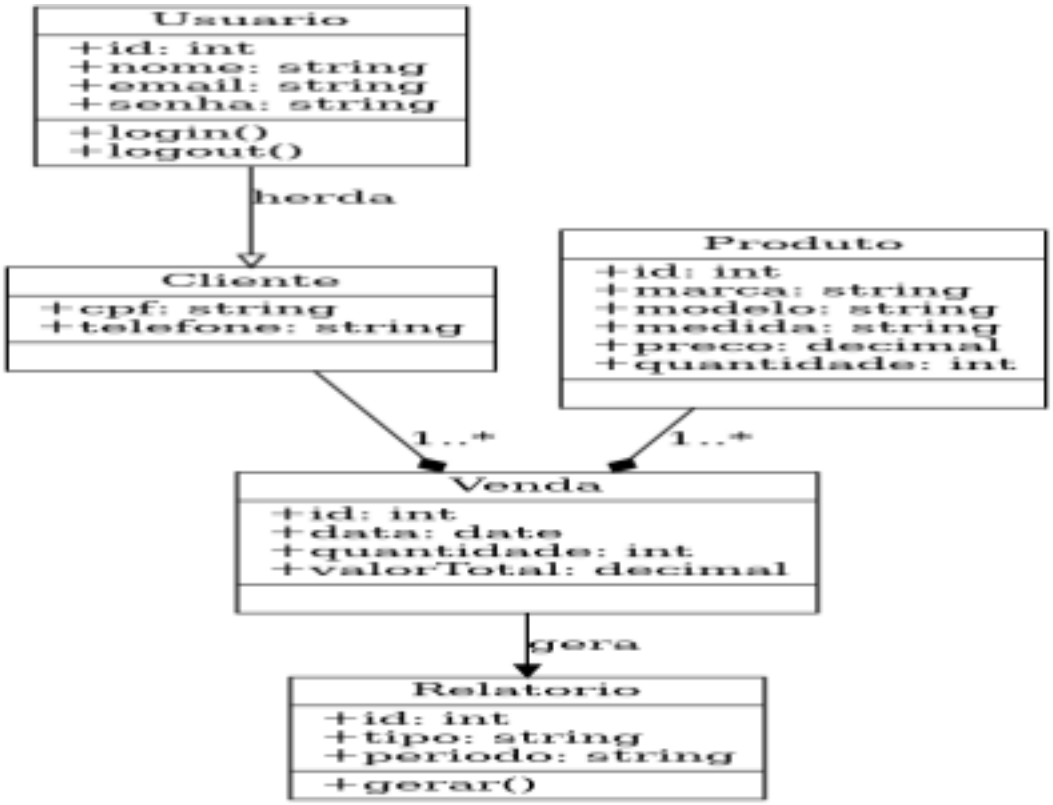


Figura 3 – Diagrama de Entidade-Relacionamento do sistema Hot Wheels.

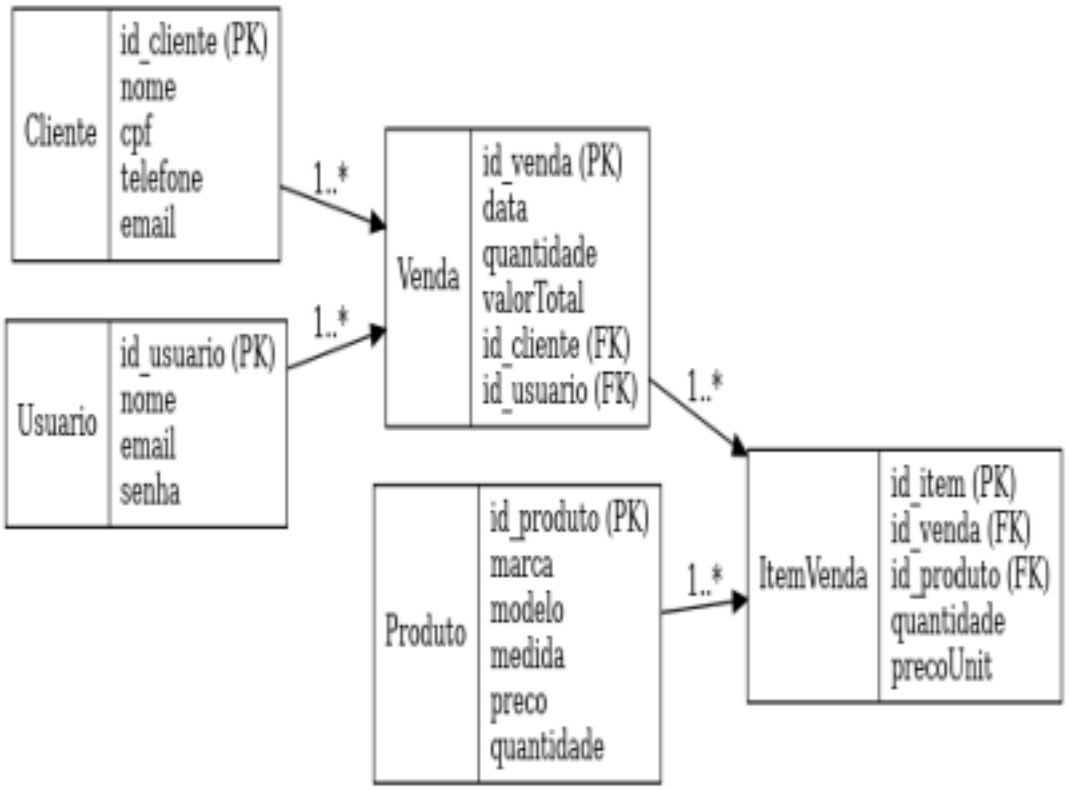
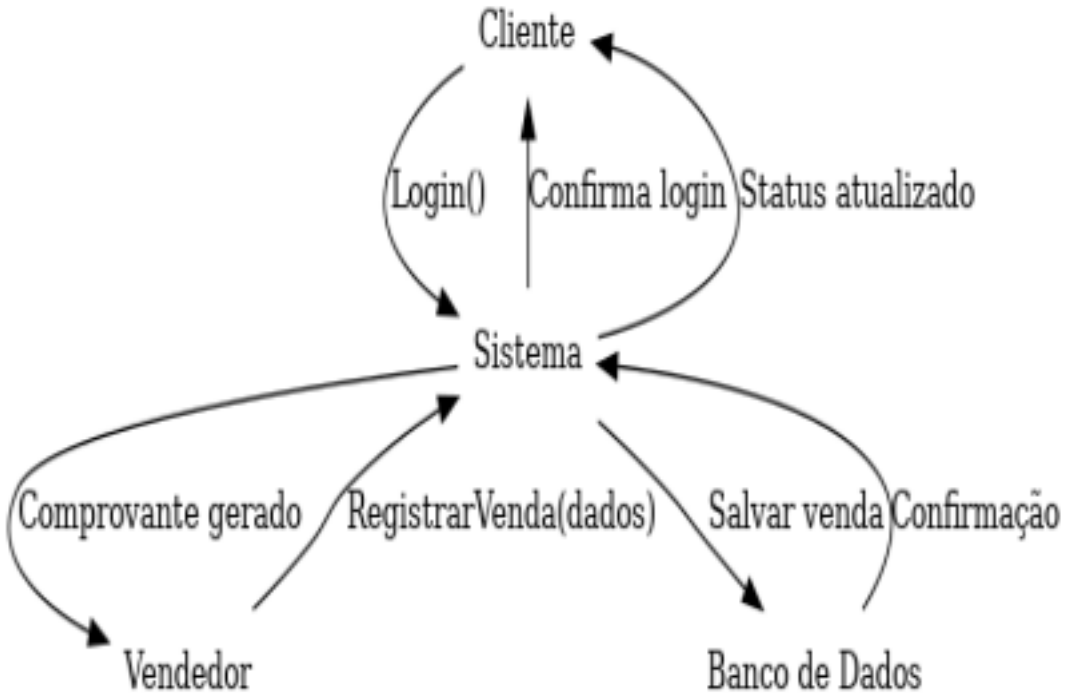


Figura 4 – Diagrama de Sequência do sistema Hot Wheels.



Documento de Evidências – Sistema Hot Wheels

1. Introdução

Este documento consolida todas as evidências necessárias sobre o backend do sistema HotWheels, incluindo arquitetura, funcionamento da API, inicialização do servidor, criação do banco de dados, funcionamento das rotas e validação por testes automatizados.

2. Swagger e API Funcionando

A API foi iniciada com sucesso utilizando o comando:

uvicorn main:app --reload --port 8001

O servidor exibiu a seguinte mensagem, confirmando sua inicialização:

• Uvicorn running on http://127.0.0.1:8001

• Application startup complete.

Isso confirma que o Swagger está acessível pelo endereço:

http://127.0.0.1:8001/docs

3. Banco de Dados

O banco de dados SQLite foi criado automaticamente na inicialização do aplicativo, através da função create_db_and_tables. As tabelas criadas foram:

• user
• cliente
• produto
• venda
• itemvenda

4. Testes Automatizados

Os testes foram executados utilizando pytest com o comando:

pytest -q

O resultado final foi:

5 passed, 0 failed

Isso comprova que todas as rotas principais — Cliente, Produto e Venda — funcionam corretamente e que o estoque é atualizado conforme esperado.

5. Estrutura da API

A API segue arquitetura modular com separação em:
• models
• schemas
• crud
• routes
• db
• core

Essa organização segue os princípios de Clean Architecture e facilita a manutenção e escalabilidade.

6. Conclusão

Com base nas evidências apresentadas — funcionamento do servidor, Swagger acessível, banco de dados configurado e 100% dos testes passando — o backend do sistema Hot Wheels está completamente funcional e pronto para integração com frontend e para entrega acadêmica.

Arquivo README

1. Tecnologias utilizadas

- Python 3.11
- FastAPI
- SQLModel / SQLAlchemy
- SQLite
- Uvicorn
- Pytest

2. Como rodar

2.1. Ativar ambiente

.\.venv\Scripts\Activate.ps1

2.2. Rodar servidor

uvicorn main:app --reload

2.3. Abrir swagger

http://127.0.0.1:8000/docs

2.4. Testes

pytest -s

2.5. Estrutura

- hotwheels/
- app/
- models/
- schemas/
- crud/
- routes/
- db/
- core/
- tests/
- main.py
- hotwheels.db
- README.md