Sistema de Gestão para Alfaiataria

Universidade Católica de Brasília
Laboratório de Banco de Dados – 2025/2

Autores

- Gustavo Henrique Lemos Xavier (Banco de Dados e NoSQL)
- Artur Coelho (Frontend)
- David Beckhan (Backend e Frontend)

Brasília DF

Resumo

Este artigo apresenta o desenvolvimento de um sistema completo de gestão para uma loja de alfaiataria, integrando soluções de frontend, backend, banco de dados relacional (MySQL) e banco de dados NoSQL (MongoDB). O objetivo do sistema é otimizar o controle de usuários, produtos, vendas e processos internos, garantindo segurança, escalabilidade e facilidade de manutenção. São descritas as etapas metodológicas, modelagem do banco de dados, tecnologias utilizadas e justificativas para cada decisão arquitetural.

Palavras-chave: Alfaiataria, Sistema Web, MySQL, MongoDB, Backend, Frontend, Controle de Acesso.

1. Introdução

A modernização de pequenos comércios exige a adoção de sistemas eficientes para gerenciamento de dados e processos. No contexto de uma alfaiataria, o controle de pedidos, produtos, clientes e usuários internos demanda ferramentas que garantam organização e confiabilidade. O presente artigo descreve o desenvolvimento de um sistema web completo, abordando sua modelagem, estrutura técnica, funcionalidades e integrações.

2. Objetivos

- Desenvolver um sistema web funcional para gerenciar uma alfaiataria.
- Implementar controle de acesso baseado em grupos e permissões.
- Criar um banco de dados relacional estruturado para garantir a integridade dos dados.
- Integrar um banco de dados NoSQL para funcionalidades complementares.
- Disponibilizar uma interface amigável para uso administrativo.

3. Metodologia

A divisão de responsabilidades entre os integrantes do projeto ocorreu da seguinte forma:

- **Gustavo Henrique Lemos Xavier:** responsável pela modelagem, criação e implementação do banco de dados relacional (MySQL) e pelo banco NoSQL (MongoDB).
- **Artur Coelho:** responsável pelo desenvolvimento do frontend do sistema.
- **David Beckhan:** responsável pela implementação do backend e integração com o banco de dados e as APIs.

A construção do sistema seguiu as etapas:

1. Requisitos.
2. Modelagem conceitual do banco (DER).
3. Implementação do MySQL contendo entidades principais do negócio.
4. Construção do backend (API) para comunicação com o frontend.

5. Implementação do frontend em ambiente web.
6. Integração com banco NoSQL para funcionalidades específicas.
7. Testes unitários e validação de consistência.

4. Descrição do Sistema

O sistema de alfaiataria foi projetado para centralizar informações e agilizar processos internos. Ele permite o cadastro e gerenciamento de usuários, produtos, vendas e registros adicionais, e informações auxiliares no banco NoSQL.

5. Funcionalidades

- Cadastro, edição e exclusão de usuários e grupos.
- Gerenciamento de produtos.
- Registro de vendas.
- Controle de permissões por grupo.
- Armazenamento de logs e dados não estruturados via NoSQL.

6. Tecnologias Utilizadas

Frontend

- HTML5, CSS3 (arquivo style.css), JavaScript.

Backend

- Python 3.x, FastAPI (framework web assíncrono e moderno).
- SQLAlchemy (ORM) para mapeamento objeto-relacional.
- MySQL (acesso via SQLAlchemy / pymysql).
- MongoDB (pymongo) para coleções complementares: logs, avaliações, histórico flexível.

Banco Relacional (SGBD)

- MySQL Workbench 8.0 CE

Banco NoSQL

- MongoDB

7. Justificativa das Tecnologias

- **MySQL:** Foi o que aprendemos, logo o mais intuitivo e comum até o momento.

- **Python**: foi escolhido por ser uma linguagem simples, rápida de desenvolver e com grande suporte da comunidade. Além disso, possui bibliotecas modernas, como FastAPI e SQLAlchemy, que facilitam a criação de APIs eficientes, seguras e de fácil manutenção.

- **FastAPI**: escolhido por sua simplicidade, desempenho e suporte nativo a documentação

- **OpenAPI**: Facilita criação de APIs REST, validação de entrada com Pydantic e integração com ASGI servers (uvicorn), resultando em boa produtividade e performance.

- **SQLAlchemy + MySQL**: o domínio de produtos, usuários, vendas e relacionamentos é fortemente relacional (FKs, integridade referencial).

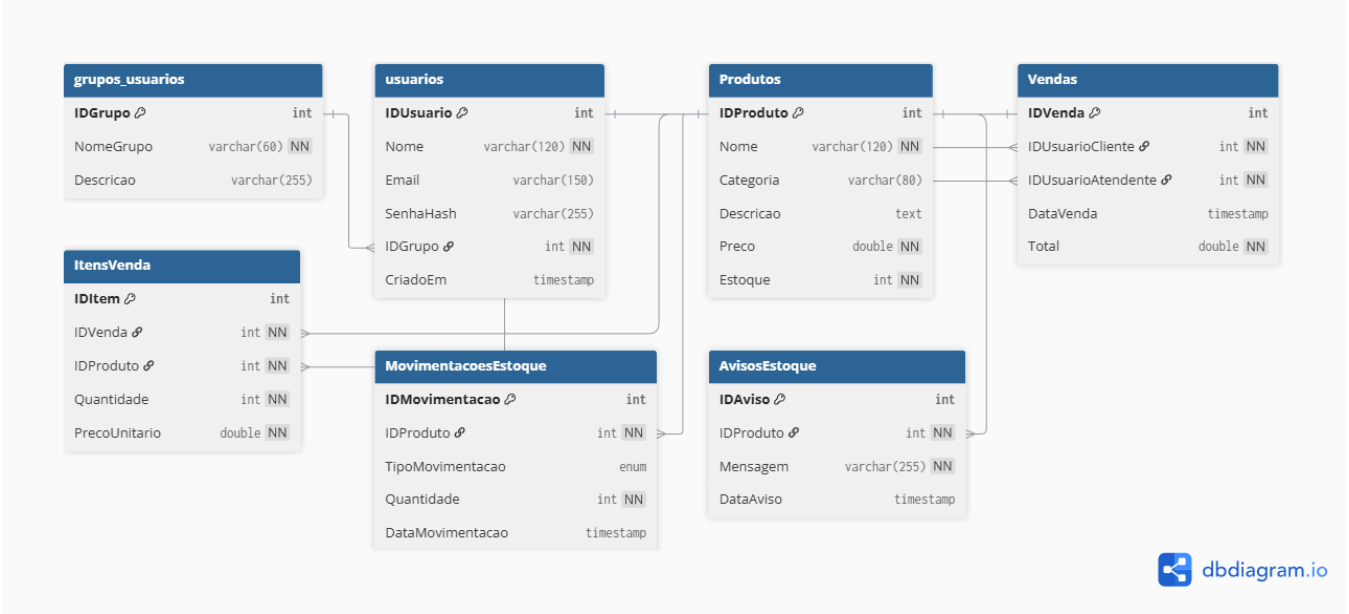
- **SQLAlchemy:** permite escrever código Python limpo e manter camadas de acesso com abstração.

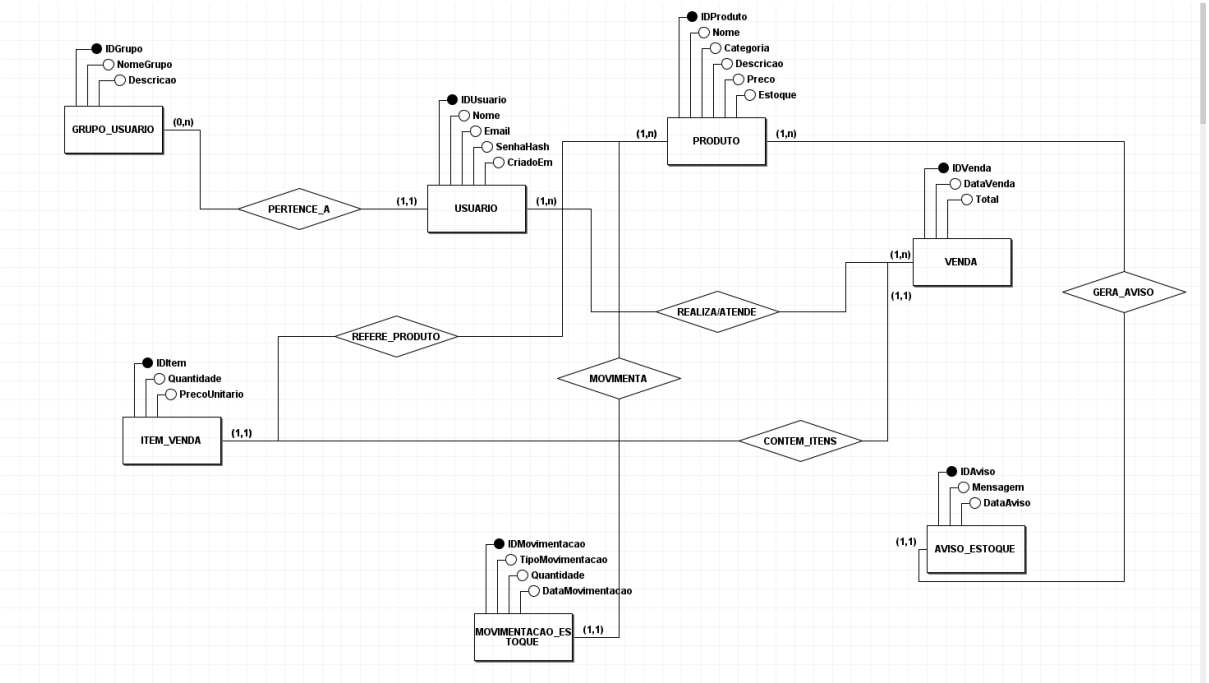
- **MongoDB**: utilizado para armazenar coleções onde o esquema pode variar, como avaliações, logs de eventos, histórico de interações, ou análises que não exigem joins fortes. MongoDB permite agilidade para dados semi-estruturados.

- **Frontend (HTML/CSS/JS):** desenvolvido em tecnologia leve sem frameworks para simplicidade, fácil hospedagem e menor complexidade para o escopo acadêmico.

8. Modelagem do Banco de Dados

8.1 Diagrama Entidade-Relacionamento (DER)





8.2 Entidades e Relacionamentos

- **grupos_usuarios:** define papéis e permissões gerais.
- **usuarios:** armazena informações essenciais dos usuários.
- **produtos:** registra itens da alfaiataria.
- **vendas:** representa transações comerciais.
- Relacionamentos utilizam chaves estrangeiras para garantir integridade.

8.3 Uso de Índices, Triggers, Views e Procedures

- **Índices:** aceleram buscas por ID e nomes.
- **Triggers:** podem validar dados e automatizar ajustes após operações.
- **Views:** facilitam consultas específicas para relatórios.
- **Procedures/Functions:** centralizam regras de negócio no banco.

9. Controle de Acesso

9.1 Usuários e Grupos

- Grupo "Gerência": Acesso completo.
- Grupo "Funcionário": Acesso restrito, apenas altera, remove e adiciona produtos no estoque.
- Grupo “Cliente”: Apenas vê os produtos para realizar o pedido

9.2 Regras de Acesso

- CRUD completo apenas para gerência.

- Funcionários podem consultar e registrar vendas.

10. Uso do Banco NoSQL

10.1 Explicação Técnica do MongoDB

MongoDB é um banco orientado a documentos (formato JSON), flexível e ideal para dados não estruturados.

10.2 Justificativa no Sistema

No sistema da alfaiataria, o MongoDB foi utilizado para:

- Armazenamento de logs de acesso.
- Registro de ações do sistema.
- Salvamento de configurações dinâmicas.

11. Conclusão

O sistema Loja Alfaiataria implementa as funcionalidades essenciais de um pequeno sistema de vendas com separação de responsabilidades. As escolhas tecnológicas (FastAPI, SQLAlchemy, MySQL e MongoDB) são coerentes com os requisitos: desempenho, consistência relacional para dados transacionais e flexibilidade para dados semi-estruturados. Recomenda-se endurecer a segurança (uso de tokens JWT, HTTPS, validação no backend), adicionar testes automatizados e documentar endpoints via OpenAPI/Swagger (já disponível via FastAPI).

12. Referências

- Repositório GitHub: https://github.com/Coelhoartur19/Alfaiataria
- Documentação do MySQL.
- Documentação do MongoDB.
- Materiais acadêmicos utilizados no curso.