UNIVERSIDADE DE SÃO PAULO

ARTHUR YANG TUNG - 14559819

FILIPE VALERIANO DE OLIVEIRA – 14570701

RENAN MOURA NASCIMENTO - 14748921

THEO DJRDJRJAN BRITO – 13688367

KEVIN TAMAYOSE - 14669711

Relatório do EP da disciplina Banco de Dados 2

SÃO PAULO

2025

1.Introdução

O **PetMatch** é uma plataforma web desenvolvida para facilitar a **adoção de animais** e oferecer **produtos e serviços** voltados ao público pet.
A principal motivação do projeto é contribuir com o combate ao abandono de animais, promovendo a adoção consciente e conectando tutores a serviços confiáveis. O projeto une **responsabilidade social** e **tecnologia**, oferecendo uma solução completa para o bem-estar animal.

2. Arquitetura do Sistema

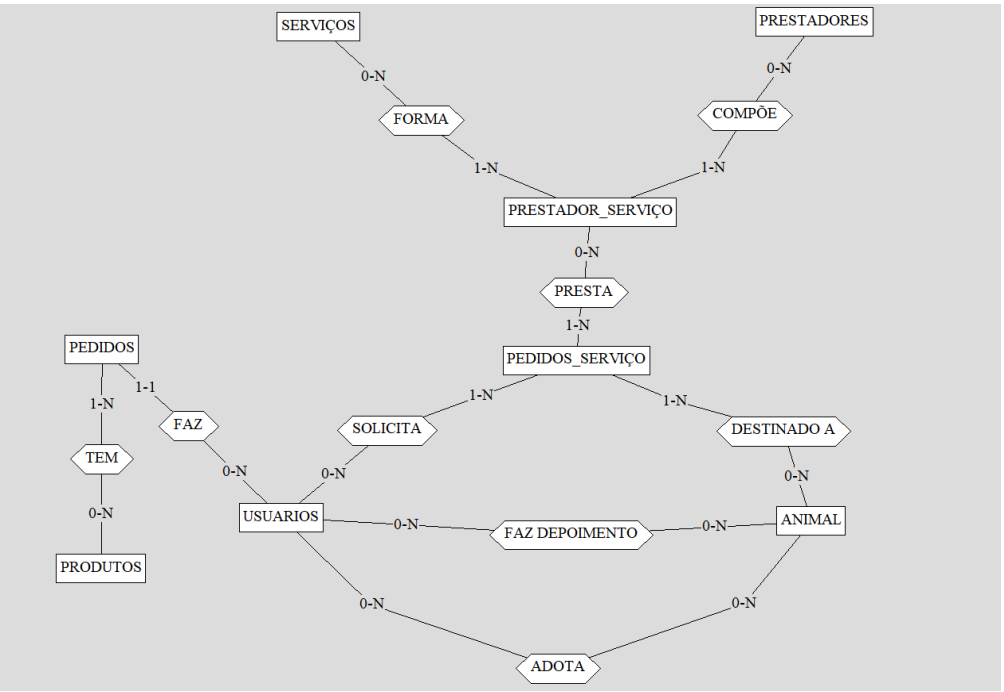
O sistema foi estruturado em camadas, com a seguinte arquitetura:

- **Frontend:** HTML, CSS, com templates Jinja2 integrados ao Flask.
- **Backend:** Python com Fastapi, responsável pelo roteamento, renderização de páginas, integração com banco de dados e API.
- **Banco de Dados:** PostgreSQL, modelado com foco em integridade referencial, normalização e escalabilidade.
- **Ferramentas de apoio:** WSL para ambiente Linux, pgAdmin para gerenciamento de dados, e pg_restore para restauração de backups.

3. Modelagem de Dados

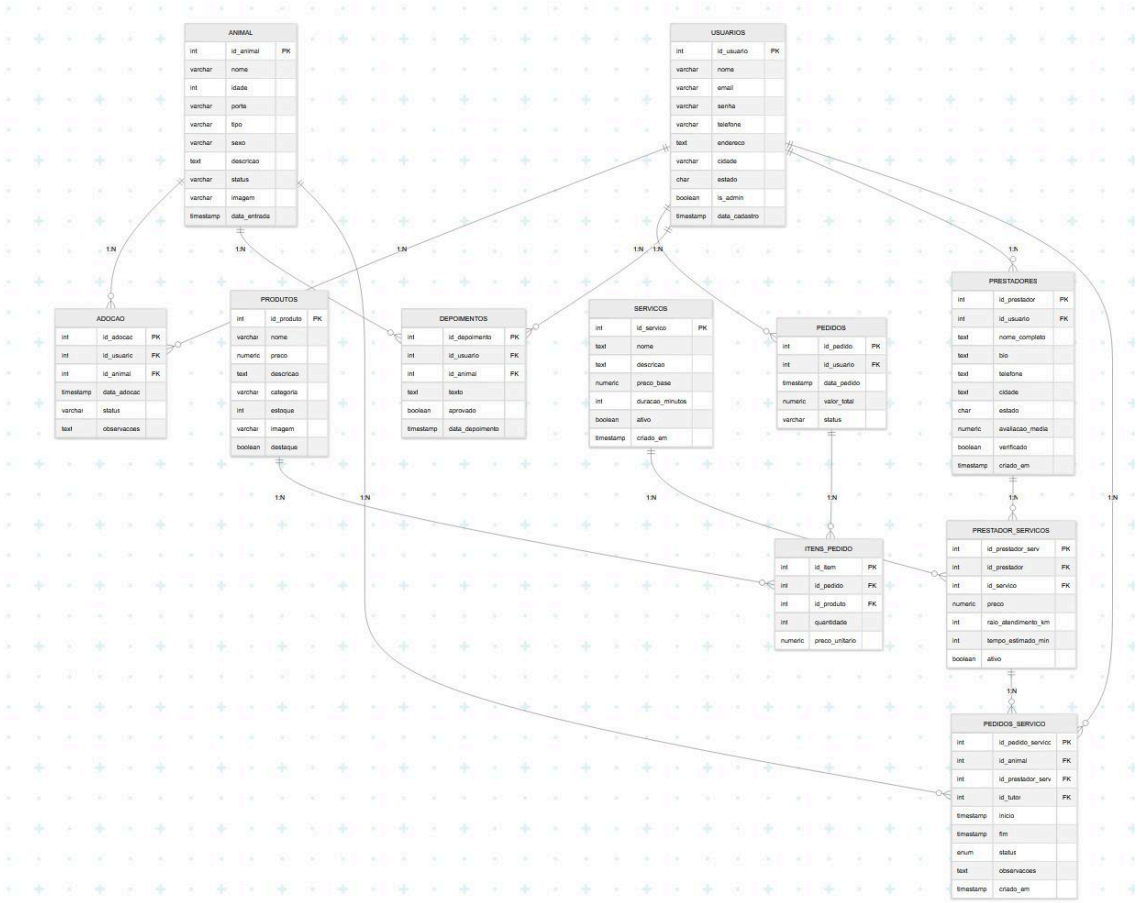
3.1 Modelo Conceitual

O modelo conceitual foi estruturado com as principais entidades: **usuários, animais, produtos, serviços, adoções** e **pedidos**.



3.2 Modelo Lógico

Utilizando a ferramenta **Mermaid**, foi possível modelar visualmente as entidades, atributos e relacionamentos, garantindo **normalização** e **integridade referencial**.



3.3 Modelo Físico

O banco foi implementado em **PostgreSQL**, incluindo:

- Scripts completos para criação das tabelas.

- API com classes padronizadas e CRUD completo para todas as entidades (com restrições para administradores).

- **Índices** em colunas-chave de busca e filtros..

- População de dados com biblioteca **Faker**.

- Código de benchmark comparando desempenho com e sem índices, para inserção e leitura.

4. Funcionalidades Implementadas

4.1 Adoção de Animais

- Cadastro de animais com status, tipo, porte e imagem.
- Listagem dos animais em destaque na página inicial.
- Página individual para cada pet com botão "Quero Adotar".

4.2 Loja Virtual

- Exibição de produtos com imagem, descrição e preço.
- Carrinho funcional implementado com JavaScript.
- Estrutura para pedidos e itens de pedido já presente no banco.

4.3 Serviços

- Cadastro de serviços com preço base e duração.
- Relacionamento com prestadores.
- Tabela de pedidos de serviços com status de confirmação.

5. Uso de Inteligência Artificial

Diversas IAs foram utilizadas para acelerar e otimizar o desenvolvimento:

Pontos Positivos:

- **ChatGPT:** modelagem de banco, explicações, correções e queries.
- **Claude:** código Fastapi e lógica de backend.
- **Copilot:** sugestões de código no VSCode.
- **Mermaid:** geração do modelo ERD visual.

Pontos Negativos:

- Dependência em trechos complexos.
- Geração de código genérica, nem sempre adaptada.
- Modelos com relacionamentos mal definidos, exigindo uso complementar do **CASE DBMain**.

6. Backup e Restauração

Para garantir a persistência e recuperação de dados, foram utilizados comandos PostgreSQL como:

- pg_dump para gerar o backup completo do banco.
- pg_restore para restaurar o banco com estrutura e dados a partir do .dump.

7. Otimizações

Durante o desenvolvimento do sistema, **foram implementados índices no banco de dados PostgreSQL** com o objetivo de melhorar o desempenho das consultas mais frequentes. As colunas escolhidas são amplamente utilizadas em filtros, joins e ordenações.

7.1 Exemplos de Índices criados:

- idx_usuarios_cidade_estado — Acelera filtros por localização (cidade e estado).
- idx_animal_status — Otimiza buscas por status de adoção.
- idx_animal_tipo — Otimiza filtragem por tipo de animal.
- idx_usuarios_is_admin — Facilita a identificação de usuários administradores.
- idx_adocao_status, idx_adocao_id_usuario, idx_adocao_id_animal — Melhoram a performance de joins e consultas relacionadas à tabela de adoções.
- idx_animal_data_entrada — Acelera buscas por data de entrada do animal.

7.2 Resultados:

Esses índices proporcionaram **ganhos de desempenho significativos** em consultas específicas, principalmente aquelas com condições altamente seletivas ou que envolvem múltiplas tabelas.

8. Considerações finais

O projeto PetMatch consolidou os aprendizados da disciplina, unindo modelagem, implementação e otimização de banco de dados. A aplicação foi construída com backend em Fastapi e integração com PostgreSQL, incluindo funcionalidades completas e uso de índices para melhorar o desempenho.

O uso de ferramentas de IA auxiliou no desenvolvimento, embora exigisse revisão manual. O trabalho em equipe e a boa divisão de tarefas foram essenciais para a entrega de um sistema funcional e organizado.

9. Participação Individual

- **Renan Moura Nascimento - 14748921**

Desenvolveu o backend com Fastapi, incluindo rotas públicas e administrativas. Implementou o CRUD de animais e produtos com integração ao PostgreSQL. Criou o painel administrativo, tratamento de erros e mensagens de feedback ao usuário.

- **Theo Djrdjrjan Brito - 13688367**

Fez os modelos conceitual e lógico, organizou todo o relatório e slides das apresentações, assim como corrigiu erros e deu apoio a revisões, além de participar da apresentação final.

- **Kevin Tamayose - 14669711**

Modelou o banco no PostgreSQL, criou scripts SQL e aplicou otimizações com índices e views materializadas.

- **Arthur Yang Tung - 14559819**

Atuou no desenvolvimento do backend, criando e manipulando endpoints e classes. Ajudou na construção do frontend com páginas e visuais do app. Trabalhou na integração entre o sistema e o banco de dados, além de participar ativamente na resolução de bugs e inconsistências.

- **Filipe Valeriano Batista de Oliveira - 14570701**

Responsável pelos testes finais, revisão de dados e apoio na documentação e roteiro da apresentação.

Link do Repositório

https://github.com/Renan-MouraN/EP-BD2