

Projeto Detran-like

Projeto A3: Programação de
Soluções Computacionais



Integrantes do Projeto

- **Erick Vinícius Ferreira da Silva / RA: 12925114010**
- **Fabrício Jardim Zietlow / RA: 12925114421**
- **Pedro Henrique Silva de Oliveira / RA: 12925115152**
- **Adriano Junior de Oliveira / RA: 12925114205**





Objetivo do Projeto

- Desenvolver um sistema de gestão de veículos completo, inspirado nas funcionalidades do DETRAN.
- Aplicar na prática os conceitos fundamentais do semestre:
- Programação Orientada a Objetos (POO).
- Persistência de dados
- Boas práticas de desenvolvimento e organização de código.



Nossa Solução

Uma Aplicação Desktop Funcional e Intuitiva

Tecnologias:

- **Java:** A base do nosso sistema.
- **Java Swing:** Para construir uma interface gráfica simples e funcional.
- **MySQL:** Para guardar os dados de forma permanente e segura.

Interface Principal:

- Um menu centralizado com acesso a todas as funcionalidades, pensado para ser fácil de usar.

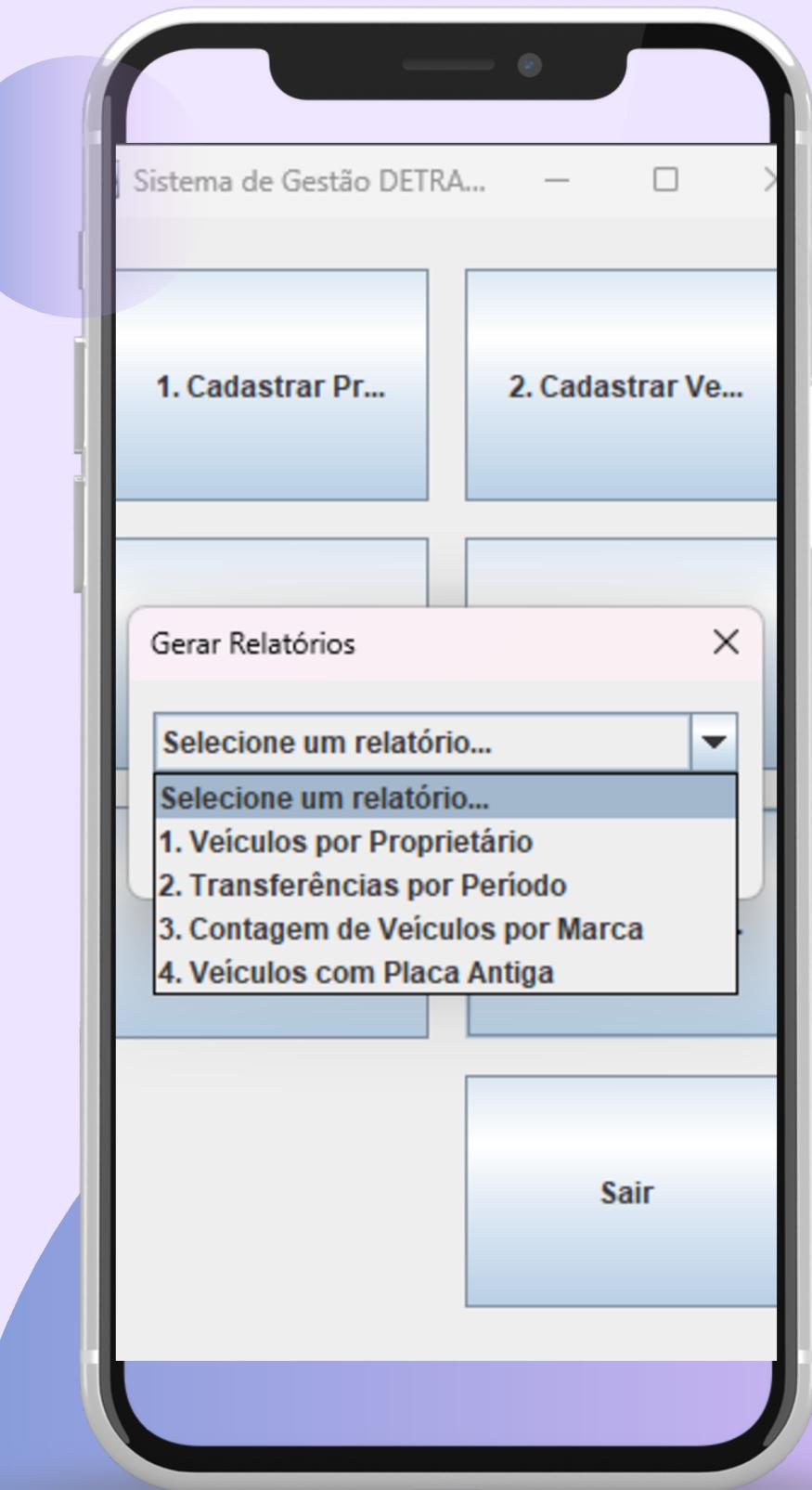


Arquitetura do Código

● Um Projeto Organizado em Camadas

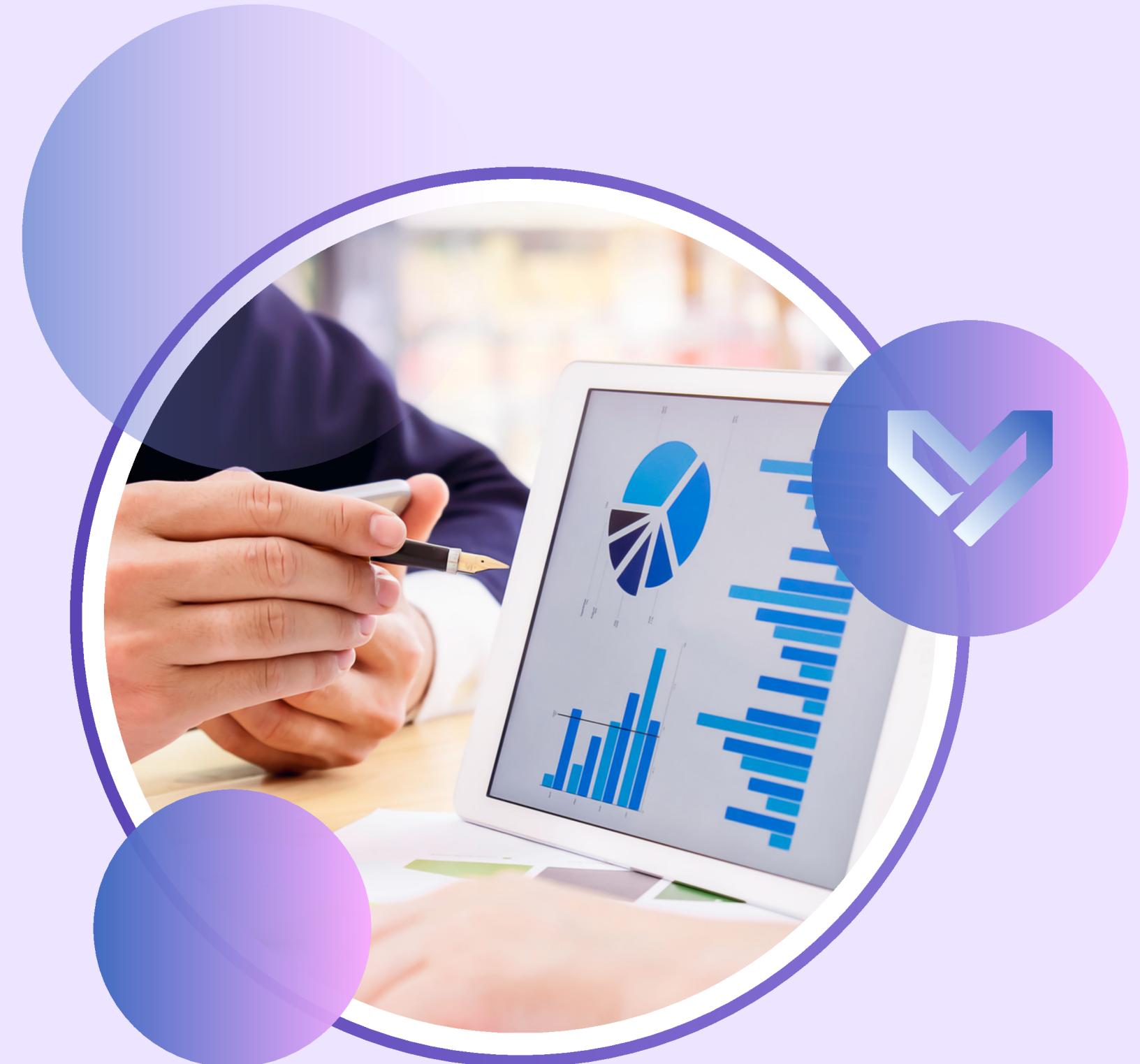
Para manter o código limpo e fácil de dar manutenção, dividimos o projeto em pacotes com responsabilidades bem definidas:

- **main**: A nossa interface gráfica, a "cara" do projeto.
- **model**: Os "moldes" dos nossos dados (Veiculo, Proprietario, etc.).
- **dao**: O nosso "tradutor", que é o único que fala com a base de dados.
- **service**: O especialista que cuida das regras de negócio, como a conversão de placas.
- **util**: Onde fica a nossa classe de conexão com a base de dados.



Como os Dados se Conectam

- A nossa base de dados tem 3 tabelas principais: **proprietarios**, **veiculos** e **transferencias**.
- Usamos chaves primárias (**PRIMARY KEY**) para garantir que cada registro é único (**CPF** e **Placa**).
- A relação entre um veículo e o seu dono é feita com uma chave estrangeira (**FOREIGN KEY**), ligando a tabela veiculos à proprietarios.



Regras de Negócio na PlacaService

- **Isolamento:** Toda a lógica de placas está na classe PlacaService para tornar o código mais flexível.
- **Geração de Placas:** O método gerarPlacaMercosul() cria automaticamente uma placa no padrão novo para cada veículo cadastrado.
- **Conversão Segura:** O método converterParaMercosul() tem uma "guarda de segurança" (**.matches()**) que só converte placas se elas estiverem no formato antigo, evitando erros.



Persistência de Dados no SistemaDAO

- **Segurança Primeiro:** Usamos PreparedStatement para proteger o sistema contra ataques de SQL Injection.
- **Transferência Atómica:** A operação mais crítica, a transferência, é tratada como uma transação bancária.
- **conn.setAutoCommit(false);**: "Não grave nada ainda."
- **conn.commit();**: "Deu tudo certo, pode gravar."
- **conn.rollback();**: "Deu erro, desfaz tudo!" Isto garante que a nossa base de dados nunca ficará com dados inconsistentes.



Desafios e Aprendizados

- **Maior Desafio:** Entender e implementar corretamente as transações atómicas no JDBC. Foi um passo crucial para garantir a integridade dos dados e o aspetto mais profissional do nosso backend.
- **Principal Aprendizado:** A importância da separação de camadas. Manter a interface, as regras de negócio e o acesso a dados em sítios diferentes tornou o projeto muito mais fácil de desenvolver e depurar em equipe



Por hoje é só,
pessoal.