

# Projeto tabela FIPE-consultas

Alexsandro Pereira

11/02/2026 — Entrega final

## Sumário

- Planning
- Modelo de Dados
- Modelo de Componentes (TDD resumido)

Eu sou o time inteiro e preciso agir como empresa: **planejar → documentar → prototipar → codar → testar → apresentar.**

## Escopo mínimo implementável

- Interface de consulta (usuário final), estilo FIPE:
  - Dropdown 1: **Marca**
  - Dropdown 2: **Modelo**
  - Dropdown 3: **Ano-modelo / versão**
  - Botão **Consultar**
  - Resultado: **preço + mês de referência**
- Log simples de consulta (registrar que houve consulta).
- Regra de fallback: **se não existir cotação no mês escolhido, retornar o último mês disponível.**

**Fora do escopo de código (apenas projetado/documentado):** Admin / Gerente / Coordenador / Pesquisador / Lojista / Batch mensal (fechamento).

# 1 1) Planning da semana

## 1.1 1.1 Board

Figura abaixo: visão do Jira (*SEM3 board*) com tarefas de planejamento concluídas e rastreabilidade do trabalho.

The screenshot shows a Jira board titled "SEM3 board". It has four columns: "BACKLOG" (30/31), "SELECTED FOR DEVELOPMENT", "IN PROGRESS", and "DONE". The "BACKLOG" column contains several items under "STORY 1" and "STORY 2". The "SELECTED FOR DEVELOPMENT" column has items like "EPIC 2 — Consulta Tabela FIPE (Usuário Final)" and "Criar schema SQLite (marca, modelo, versão, cotação)". The "IN PROGRESS" and "DONE" columns are currently empty. The interface includes various filters and search bars at the top.

Figura 1: Jira SEM3 — organização da planning.

## 1.2 1.2 Board da semana

A semana é planejada para maximizar **clareza de arquitetura + entrega mínima funcional** dentro do tempo.

This screenshot shows a broader view of the Jira board, likely a macro plan. It includes a sidebar with navigation links like "For you", "Recent", "Starred", "Apps", "Plans", "Spaces", and "Goals". The main area has four columns: "BACKLOG" (20/20), "SELECTED FOR DEVELOPMENT" (6/6), "IN PROGRESS" (1/1), and "DONE" (4/4). The tasks listed are similar to Figure 1, including EPICs and specific stories. The interface includes a sidebar with navigation links like "Summary", "Timeline", "Kanban board", and "More".

Figura 2: Plano semanal (macro)

## 1.3 6. Visão do Sistema Completo (Projeto Global)

Embora apenas o módulo de **consulta pública** tenha sido implementado, o sistema foi projetado considerando o fluxo completo de geração da Tabela FIPE, incluindo papéis operacionais e processamento mensal (batch). O objetivo é separar claramente **entidades do domínio, papéis (roles), processos e responsabilidades**.

### 1.3.1 6.1 Papéis do Sistema (Role-Based)

Os papéis são **atributos comportamentais do usuário** (roles), e **não entidades independentes**. Assim, o sistema utiliza uma única entidade `usuario` e diferencia ações por `role`.

- **Usuário Final:** consulta valores consolidados.
- **Usuário Administrador (role=admin):** gerencia usuários e permissões.
- **Usuário Gerente de Catálogo:** cadastra marca/modelo/versão.
- **Usuário Coordenador Regional:** organiza regiões, lojas e atribuições.
- **Usuário Pesquisador:** registra preços coletados em lojas.

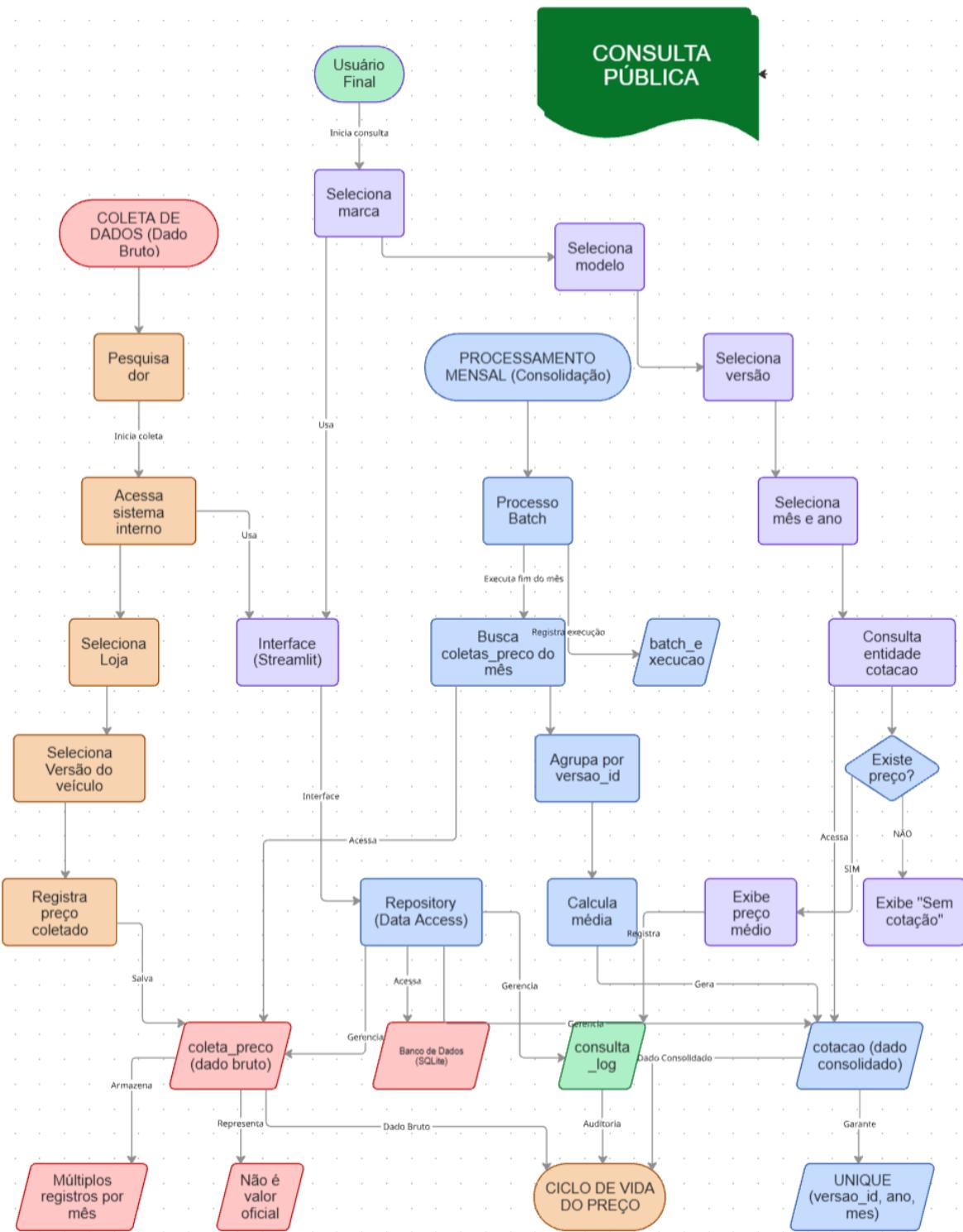


Figura 3: Enter Caption

### 1.3.2 6.2 Fluxo Completo de Geração da Cotação

(Batch)

1. Gerente cadastra **marca/modelo/versão**.
2. Coordenador define **lojas por região** e atribui pesquisadores.
3. Pesquisadores coletam preços e registram em **coleta\_preco** ( dado bruto).
4. Processo **batch mensal** agrupa e calcula **preço médio**.
5. Resultado consolidado é persistido em **cotacao**.
6. Usuário final consulta o valor consolidado (módulo implementado).

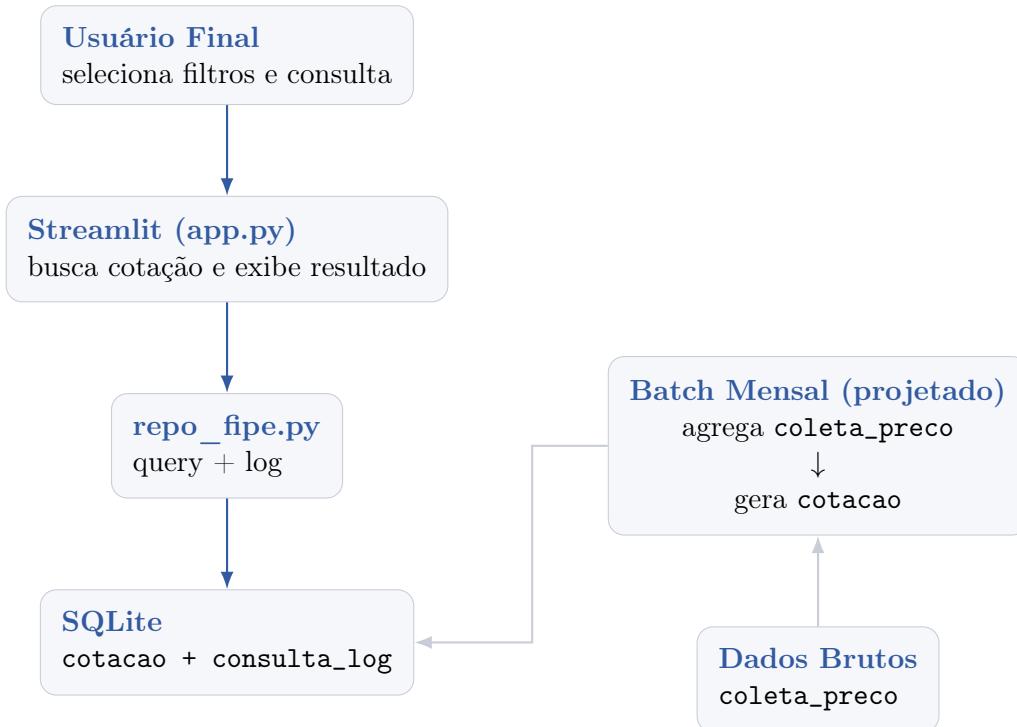


Figura 4: Fluxo síncrono de consulta (implementado) e processamento batch mensal (projetoado).

### 1.3.3 6.3 Entidades Adicionais Projetadas (Não Implementadas)

As entidades abaixo fazem parte do desenho global, mas não foram implementadas por restrição de escopo:

- `usuario` (com `role` para admin/gerente/coordenador/pesquisador)
- `regiao, loja`
- `coleta_preco` (registro bruto)
- `batch_execucao` (processamento mensal)

## 1.4 7. Modelo Conceitual Completo (ER – Texto Estruturado)

### 1.4.1 7.1 Entidades e Relacionamentos (Resumo)

O modelo conceitual completo foi projetado considerando o ciclo integral de geração da Tabela FIPE, desde a coleta bruta de preços em campo até a consolidação mensal das cotações.

As entidades marca, modelo e versao estruturam o domínio do veículo, enquanto cotação representa o valor médio consolidado por mês e ano.

A entidade consulta\_log registra auditoria das consultas públicas realizadas pelos usuários. No desenho arquitetural completo (não implementado integralmente nesta etapa), o sistema inclui ainda usuário, região, loja e coleta\_preco, permitindo modelar o fluxo real de pesquisa de campo. A consolidação desses dados ocorre por meio de uma execução mensal representada pela entidade batch\_execucao, responsável por transformar dados brutos em cotações oficiais.

Nesta entrega, foi implementado apenas o fluxo síncrono de consulta pública, mantendo o restante como projeção arquitetural formalmente documentada.

#### 1.4.2 Modelo de Dados do Sistema

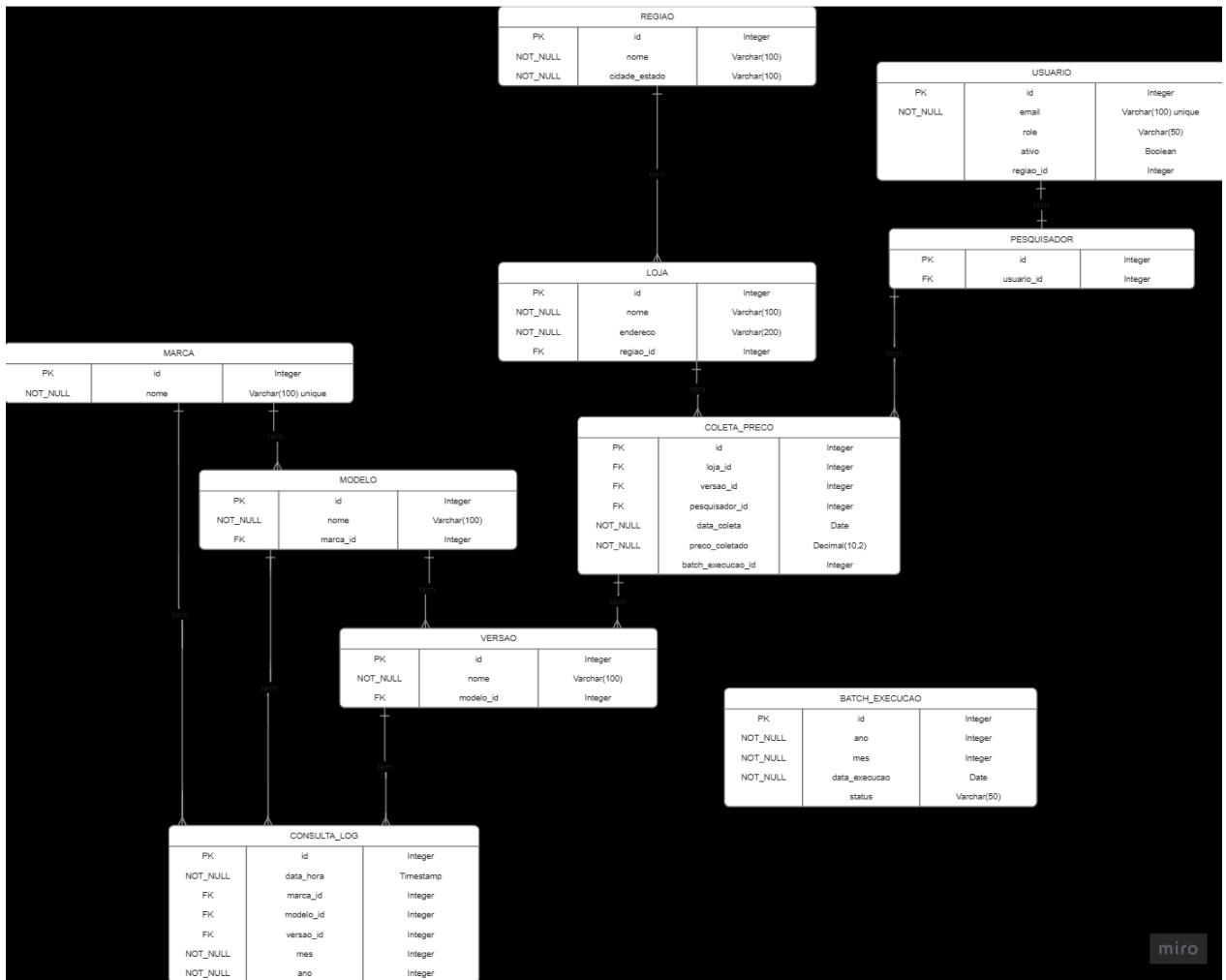


Figura 5: Enter Caption

Figura 6: Modelo conceitual do sistema FIPE-like (implementado + projetado).

### 1.5 8. Diagrama de Componentes (Arquitetura de Software)

#### 1.5.1 8.1 Camadas e Componentes

A arquitetura adotada é um **monolito organizado em camadas**, adequada ao escopo e ao tempo de implementação, preservando separação de responsabilidades.



Figura 7: Arquitetura em camadas do módulo implementado (consulta pública).

## 1.6 1.3 Entregas e DoD (Definition of Done)

**Entrega 2:**

- Documento com: Planning + Modelo de Dados + Modelo de Componentes (TDD resumido).

**DoD do MVP :**

- Tela Streamlit com 3 dropdowns em cascata + botão Consultar.
- Retorna preço + mês de referência.
- Aplica fallback do último mês disponível quando faltar cotação no mês selecionado.
- Registra log simples da consulta.
- Código organizado em camadas (UI / Service / Repository).

## 2 Decisões explícitas (Mini ADRs)

### ADR-001 — Persistência

**Decisão:** usar **SQLite** com **seed local**.

**Motivo:** velocidade de implementação + simplicidade + rastreabilidade do dado.

**Consequência:** dados reais de operação não existem; simulação via seed.

### ADR-002 — Arquitetura

**Decisão:** monólito em camadas (**UI → Service → Repository**).

**Motivo:** fácil de avaliar e evoluir (troca regra sem mexer na UI; troca BD sem quebrar).

regra).

**Consequência:** sem microserviços; foco na clareza.

ADR-003 — Regra de consulta com fallback

**Decisão:** se mês selecionado não tiver cotação, retornar **último mês disponível** para a mesma versão/ano.

**Motivo:** comportamento realista e robusto (dados podem estar incompletos).

**Consequência:** precisa de query por “máximo mês disponível”.

### 3 Rastreabilidade mínima (Requisito → Entidade → Componente)

Requisito	Entidades (dados)	Componentes (código)
Consulta FIPE (MVP)	Marca, Modelo, VersaoAno, CotacaoMensal	UI(Streamlit), QueryService, SQLiteRepo
Fallback último mês	CotacaoMensal	QueryService (regra) + SQLiteRepo (query ordenada por mês)
Log simples de consulta	ConsultaLog	QueryService (orquestra) + SQLiteRepo (insert log)

Tabela 1: Rastreabilidade do MVP (evidência de pensamento de liderança técnica).

## 4 Modelo de Dados (ERD )

#### 4.1 Diagrama ER (com destaque do MVP)

**MVP** Entidades do MVP: Marca, Modelo, VersaoAno, CotacaoMensal, ConsultaLog.

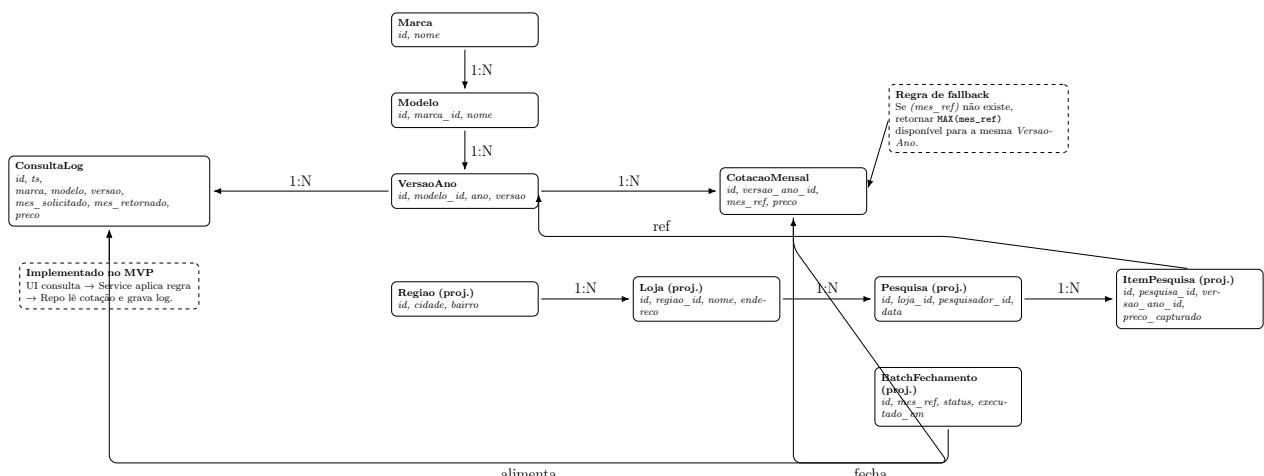


Figura 8: ERD completo (projeto) com foco no **MVP** e regra de fallback.

## 5 Modelo de Componentes (TDD resumido)

### 5.1 Camadas e responsabilidades

- **UI (Streamlit)**: captura entradas e renderiza saída. Sem regra de negócio.
- **Service (Domínio)**: valida seleção, aplica fallback do mês, orquestra log.
- **Repository (SQLite)**: queries e persistência (cotação e log).

### 5.2 5.2 Contratos (interfaces) — versão mínima

- **QueryService**
  - `list_marcas()`
  - `list_modelos(marca_id)`
  - `list_versoes(modelo_id)`
  - `consultar(versao_id, mes_solicitado) → (preco, mes_retornado)`
- **SQLiteRepo**
  - `get_cotacao(versao_id, mes)`
  - `get_ultima_cotacao(versao_id, mes_limite)`
  - `insert_log(...)`

### 5.3 5.3 Sequência do fluxo (consulta)

1. Usuário escolhe Marca → Modelo → Versão/Ano e um mês.
2. UI chama `QueryService.consultar()`.
3. Service tenta `Repo.get_cotacao()`.
4. Se não existir: Service chama `Repo.get_ultima_cotacao()` e retorna o último mês disponível.
5. Service registra `ConsultaLog`.
6. UI exibe preço + mês retornado.

## 6 Critérios de aceite do MVP (3 bullets)

- Dropdowns em cascata (marca → modelo → versão/ano) com estado consistente.
- Consulta retorna **preço + mês de referência**.
- Na ausência do mês solicitado, retorna **último mês disponível** e registra log.

## 7 Plano de teste mínimo

### Testes essenciais (manual + 1 automatizado)

- **Happy path**: existe cotação no mês escolhido → retorna preço correto e log gravado.
- **Fallback**: mês solicitado sem cotação → retorna último mês disponível + log com `mes_retornado`.
- **Dados incompletos**: marca sem modelos (ou modelo sem versões) → UI não quebra, mostra mensagem clara.

- **Automatizado (unit)**: teste do Service para fallback (mock do repo).
- 

*Entrega 1 foca em: clareza + organização + MVP consultável e robusto.*