PRD — Plataforma Analítica de Crédito PJ (Versão para Produção)

Data: 08/09/2025 • Escopo: Sprints 2→4 (documentação técnica, protótipo navegável e MVP 100% funcional).

Este documento incorpora o regulamento do Challenge e o layout de dados fornecido, detalhando arquitetura, modelos, APIs, qualidade, segurança e plano de entrega para o time de produção.

# 1) Visão, desafios e metas

* Desafio 1 — Momento de vida da empresa: classificar estágio (início, crescimento, maturidade, declínio) com EDA, feature engineering e algoritmos supervisionados e/ou não supervisionados.
* Desafio 2 — Cadeias de relações financeiras: mapear redes, hubs, dependências e riscos sistêmicos; gerar insights acionáveis no dashboard.

KPIs de Produto:

* AUC ≥ 0,80 para score de risco (baseline ML-lite).
* Latência < 2s (p95) para consulta de score em ambiente Docker local.
* Protótipo navegável do dashboard (Sprint 2) e documentação técnica de arquitetura (Sprint 2).

# 2) Dados de entrada (Challenge FIAP - Bases.xlsx)

Total de empresas (Base 1 - ID): 10,000

Total de transações (Base 2 - Transações): 100,000

Período Base 1 (DT\_REFE): 31/01/2025 a 31/05/2025

Período Transações (DT\_REFE): 01/03/2025 a 31/05/2025

Tipos de transação e contagem:

* PIX: 70,000
* BOLETO: 15,000
* TED: 10,000
* SISTEMICO: 5,000

Dicionário de dados (resumo):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Base | Variável | Nome Lógico | Tipo | Especificações |
| nan | ID | Número do Registro do cliente | String | 10 mil IDs distintos. Completamente anonimizados. |
| nan | VL\_FATU | Valor do Faturamento | Long | Valor do faturamento anual da empresa, de 0 a 300 milhões. Valores mascarados, mantendo exclusivamente a ordenação dos faturamentos das empresas |
| nan | VL\_SLDO | Valor do Saldo em Conta corrente na data de referência | Long | Valor do saldo em conta corrente no dia de fechamento do mês. Valores mascarados, mantendo exclusivamente características anonimizadas de escala e valores positivos e negativos. |
| nan | DT\_ABRT | Data de abertura da empresa | Date | Data de abertura das empresas. Valor de dia e mês mascarados, podendo variar 10 dias para frente ou para trás e 3 meses para frente ou para trás. |
| nan | DS\_CNAE | Descrição do CNAE | String | Nome lógico do CNAE da empresa, utilizando as especificações do site público do governo: https://www.gov.br/receitafederal/pt-br/assuntos/orientacao-tributaria/cadastros/cnpj/classificacao-nacional-de-atividades-economicas-2013-cnae |
| nan | DT\_REFE | Data de referência da Base | Date | Data de referência da base. São os últimos dias de 5 meses de janeiro a maio de 2025. Valores finais com range de 5 dias para frente ou para trás. |
| Base 2 | Variável | Nome Lógico | Tipo | Especificações |
| nan | ID\_PGTO | ID do cliente que enviou o valor | String | ID do pagador, ou seja, a origem da transação, de onde sai o dinheiro. O pagador tem como atribuição realizar o pagamento. Contém somente IDs da 1º tabela. |
| nan | ID\_RCBE | ID do cliente que recebeu o valor | String | ID do recebedor, ou seja, quem receptará a transação, para onde vai o dinheiro. O recebedor tem como atribuição receber o pagamento. Contém somente IDs da 1º tabela |
| nan | VL | Valor transacionado | Long | Valor transacionado. Valores mascarados, mantendo exclusivamente características anonimizadas de escala. Contém somente valores absolutos. Todo valor na visão do pagador é visto como despesa, ou seja, negativo, em contrapartida todo valor na visão recebedor é visto como uma entrada, ou seja, positivo. |
| nan | DS\_TRAN | Descrição da transação, sendo PIX, TED, BOLETO ou SISTEMICO | String | Descrição do tipo de transação. |
| nan | DT\_REFE | Data de referência da base | Date | Data de referência da transação. Contém somente transações de 3 meses, de março a maio de 2025. As datas podem variar 5 dias para frente ou para trás, mas garantindo que estão sempre no mês de referência em que aconteceu a transação. |

# 3) Arquitetura (alto nível)

Estilo: micro-front + microserviços, com Core API (Spring Boot), ML-Service (FastAPI) e Data Plane (PostgreSQL + MinIO). Orquestração local com Docker Compose.

Camadas & fluxo:

* Ingestão/ETL (Python): carga de Base 1 e Base 2; consumo de séries públicas (BACEN/IBGE) para overlay macro.
* Data Store: PostgreSQL 17 (transacional/analítico leve) e MinIO (objetos).
* Processamento Analítico: SNA (NetworkX) e Modelagem (scikit-learn, XGBoost, statsmodels).
* Core API (Java 21 + Spring Boot 3.5.5): serviços REST, segurança, documentação.
* ML-Service (FastAPI + Uvicorn): inferência de score, ARIMA básico e métricas SNA.
* Frontend (Angular 20 + Node 22 LTS): dashboard com ng-apexcharts e vis-network.

# 4) Modelo de dados alvo (camada relacional)

-- Empresa (cadastro)  
CREATE TABLE empresa (  
 id TEXT PRIMARY KEY,  
 cnpj CHAR(14),  
 ds\_cnae TEXT,  
 dt\_abrt DATE  
);  
  
-- Financeiro por referência (Base 1)  
CREATE TABLE empresa\_financeiro (  
 id BIGSERIAL PRIMARY KEY,  
 empresa\_id TEXT REFERENCES empresa(id),  
 dt\_ref DATE NOT NULL,  
 vl\_fatu NUMERIC,  
 vl\_sldo NUMERIC,  
 UNIQUE (empresa\_id, dt\_ref)  
);  
  
-- Transações (Base 2)  
CREATE TABLE transacao (  
 id BIGSERIAL PRIMARY KEY,  
 id\_pgto TEXT REFERENCES empresa(id),  
 id\_rcbe TEXT REFERENCES empresa(id),  
 vl NUMERIC,  
 ds\_tran TEXT,  
 dt\_ref DATE NOT NULL  
);  
  
-- SNA métricas  
CREATE TABLE centralidade\_snapshot (  
 id BIGSERIAL PRIMARY KEY,  
 dt\_calc DATE NOT NULL,  
 empresa\_id TEXT REFERENCES empresa(id),  
 grau NUMERIC,  
 betweenness NUMERIC,  
 eigenvector NUMERIC,  
 cluster\_id INT  
);  
  
-- Score & decisões  
CREATE TABLE score\_risco (  
 id BIGSERIAL PRIMARY KEY,  
 empresa\_id TEXT REFERENCES empresa(id),  
 dt\_calc TIMESTAMP,  
 modelo TEXT, score NUMERIC,  
 auc\_valid NUMERIC, threshold NUMERIC, versao\_modelo TEXT  
);  
  
CREATE TABLE decisao\_credito (  
 id BIGSERIAL PRIMARY KEY,  
 empresa\_id TEXT REFERENCES empresa(id),  
 dt\_decisao TIMESTAMP, score NUMERIC, aprovacao BOOLEAN,  
 limite NUMERIC, moeda TEXT, motivo TEXT  
);

# 5) ETL e mapeamentos

Mapeamento das colunas (Base 1 → Tabelas alvo):

* ID → empresa.id (TEXT)
* DT\_ABRT → empresa.dt\_abrt
* DS\_CNAE → empresa.ds\_cnae
* VL\_FATU, VL\_SLDO, DT\_REFE → empresa\_financeiro (por mês/ID)

Mapeamento (Base 2 → Transações e SNA):

* ID\_PGTO → transacao.id\_pgto
* ID\_RCBE → transacao.id\_rcbe
* VL → transacao.vl; DS\_TRAN → transacao.ds\_tran; DT\_REFE → transacao.dt\_ref
* Grafo: arestas (ID\_PGTO → ID\_RCBE, peso=ΣVL por mês/tipo). SNA: grau, betweenness, comunidades (cluster\_id).

# 6) Feature Engineering

* Crescimento de faturamento (YoY/3m), volatilidade de saldo (rolling std), sazonalidade.
* Idade da empresa (today - DT\_ABRT), setor (one-hot de DS\_CNAE).
* Centralidades na rede (grau, betweenness, eigenvector), densidade ego-network.
* Risco sistêmico: exposição a hubs de alto risco (propagação por vizinhos).

# 7) Modelagem (Desafios 1 e 2)

Desafio 1 — Momento de vida:

* Abordagem 1 (não supervisionado): clustering (KMeans/DBSCAN) em features de crescimento/estabilidade.
* Abordagem 2 (supervisionado): se rótulos estiverem disponíveis, RandomForest/XGBoost.

Desafio 2 — Cadeias financeiras:

* Detecção de hubs e dependências via centralidades + análise de comunidade.
* Indicadores no dashboard: top hubs, empresas periféricas, score de dependência.

# 8) Design de APIs (REST)

Core API (base: /api/v1)

GET /empresas/{id}/score  
200 OK  
{  
 "empresaId": "CNPJ\_00001",  
 "score": 0.82,  
 "faixa" : "médio",  
 "explicacoes": {"faturamento\_yoy": 0.3, "betweenness": 0.2}  
}  
  
GET /empresas/{id}/rede  
200 OK  
{  
 "nodes": [{"id":"CNPJ\_00001","cluster":3,"grau":12, "betweenness":0.08}],  
 "edges": [{"from":"CNPJ\_00001","to":"CNPJ\_00099","peso": 123456}]  
}  
  
GET /macro?serie=IPCA&from=YYYY-MM  
POST /decisoes (ROLE\_ANALISTA)

ML-Service (base: /ml/v1)

POST /score  
Request: {"features": {...}}  
Response: {"score": 0.82, "modelo":"rf-baseline","versao":"1.0.0"}  
  
POST /forecast/arima  
Request: {"serie": "IPCA", "horizonte": 6}  
Response: {"forecast": [..]}  
  
POST /sna/centralidades  
Request: {"edges":[["CNPJ\_00001","CNPJ\_00099",123.45], ...]}  
Response: {"grau": {...}, "betweenness": {...}, "clusters": {...}}

# 9) Frontend (Angular 20)

* Busca por ID e por CNPJ (campo único aceita ambos; validação simples).
* Dashboard: cards para Score, Rede (grafo interativo), KPIs Macro (IPCA/Selic/PIB) e Alertas.
* Página de Empresa: abas Score/Histórico, Rede, Decisões.
* PWA opcional; testes E2E com Playwright.

# 10) Segurança, LGPD e Governança

* JWT assinado localmente; papéis: ROLE\_GERENTE, ROLE\_ANALISTA, ROLE\_COMITE, ROLE\_ADMIN.
* Minimização: evitar PII; uso de IDs anonimizados e CNPJ quando fornecido no dataset.
* Auditoria: tabelas de decisão e score; logs sem dados sensíveis.

# 11) Observabilidade & Qualidade

* Health/metrics via Spring Boot Actuator.
* Integração (Testcontainers), E2E (Playwright).
* CI local: build + testes + linters; versionamento semântico das APIs.

# 12) Infra/DevOps — Docker Compose (local)

version: "3.9"  
services:  
 postgres:  
 image: postgres:17  
 environment:  
 POSTGRES\_DB: pjcredit  
 POSTGRES\_USER: pj  
 POSTGRES\_PASSWORD: pjpass  
 ports: ["5432:5432"]  
  
 minio:  
 image: minio/minio:latest  
 command: server /data --console-address ":9001"  
 environment:  
 MINIO\_ROOT\_USER: minio  
 MINIO\_ROOT\_PASSWORD: minio12345  
 ports: ["9000:9000", "9001:9001"]  
  
 backend:  
 build: ./backend  
 environment:  
 SPRING\_PROFILES\_ACTIVE: dev  
 DB\_URL: jdbc:postgresql://postgres:5432/pjcredit  
 DB\_USER: pj  
 DB\_PASS: pjpass  
 JWT\_PUBLIC\_KEY: ${JWT\_PUBLIC\_KEY}  
 ports: ["8080:8080"]  
  
 mlservice:  
 build: ./mlservice  
 ports: ["8000:8000"]  
  
 frontend:  
 build: ./frontend  
 ports: ["4200:4200"]

# 13) Critérios de aceitação (mapeados ao Regulamento)

* Sprint 2 (Doc Técnico): diagrama + camadas + tecnologias inter-relacionadas (conforme seção 3).
* Sprint 2 (Protótipo Navegável): dashboard com telas-chave (conforme seção 9).
* Sprint 3 (Solução implementada): funcionalidades 1.2.1 (classificação do momento de vida + redes financeiras).
* Sprint 4 (Dashboard completo): monitoramento e KPIs conforme protótipo.
* Banca Final: MVP 100% funcional com apresentação e documentos entregues.

# 14) Cronograma sugerido (semanas)

* S2-P1: infra compose, schema, ingestão bases, protótipo Angular shell.
* S2-P2: SNA básico + RF baseline + dashboard navegável.
* S3: ARIMA + XGBoost; decisões + auditoria.
* S4: refino UX, métricas, segurança; preparação de banca.

# 15) Riscos & mitigação

* Qualidade de dados (mascaramento): validações e outlier handling.
* Performance do grafo: agregação mensal e clustering em vis-network.
* Mudanças de escopo: congelar versões para a banca; backlog pós-banca.

# 16) Anexos operacionais

# Dev local  
cp infra/.env.example .env  
docker compose up -d --build  
  
# Backend: Swagger  
http://localhost:8080/swagger-ui  
  
# ML-Service: docs  
http://localhost:8000/docs  
  
# Frontend  
http://localhost:4200