Estructura de repositorio propuesta

```
pqr-hybrid-lakehouse/
├─ README.md
├─ LICENSE
├─ CHANGELOG.md
├─ .gitignore
├─ .editorconfig
├─ Makefile
├─ justfile                          # opcional, si te gusta just
├─ .env.example
├─ .github/
│  ├─ workflows/
│  │  ├─ ci.yml                      # lint, tests, contracts, build images
│  │  └─ release.yml                 # tags + changelog
│  └─ CODEOWNERS
│
├─ docs/
│  ├─ 00-overview/
│  │  ├─ SYSTEM-OVERVIEW.md
│  │  └─ SCOPE-MVP.md
│  ├─ 01-architecture/
│  │  ├─ ARCH-PRINCIPLES.md
│  │  ├─ ARCH-PRINCIPLES-MAP.puml
│  │  ├─ ARCH-EVOLUTION-MVP-F2-F3.puml
│  │  ├─ C4-CONTEXT.puml
│  │  ├─ C4-CONTAINERS.puml
│  │  └─ diagrams/                   # svg generados (artefactos)
│  ├─ 02-adr/
│  │  ├─ ADR-0001-...md
│  │  ├─ ...
│  │  ├─ ADR-0013-no-iceberg-delta-en-mvp.md
│  │  ├─ ADR-0014-fase-2-evolucion-lakehouse.md
│  │  └─ ADR-0015-fase-3-industrializacion.md
│  ├─ 03-runbooks/
│  │  ├─ RUNBOOK-local-dev.md
│  │  ├─ RUNBOOK-operations.md
│  │  ├─ RUNBOOK-incidents.md
│  │  └─ RUNBOOK-troubleshooting.md
│  ├─ 04-guides/
│  │  ├─ GUIDE-day-by-day.md
│  │  ├─ GUIDE-dask.md
│  │  ├─ GUIDE-prefect.md
│  │  ├─ GUIDE-rustfs.md
│  │  └─ GUIDE-metabase.md
│  └─ 99-appendix/
│     ├─ glossary.md
│     └─ references.md
│
├─ data/
│  ├─ contracts/
│  │  ├─ v1/
│  │  │  ├─ raw/
│  │  │  │  └─ RAW-EMAIL.schema.json
│  │  │  ├─ bronze/
│  │  │  │  └─ BRONZE-email.md
│  │  │  ├─ silver/
│  │  │  │  └─ SILVER-email.md
```

```
│   │   │   ├─ gold/
│   │   │   │  └─ GOLD-kpis.md
│   │   │   └─ governance/
│   │   │      ├─ DATA-LAYOUT-S3.md
│   │   │      ├─ NAMING-CONVENTIONS.md
│   │   │      └─ DATA-QUALITY-CHECKS.md
│   │   ├─ README.md
│   │   └─ CHANGELOG.md                  # cambios de contratos
│   ├─ samples/                          # ejemplos pequeños para docencia
│   │  └─ raw-email-example.jsonl
│   └─ seeds/                            # opcional: seeds para demo local
│      └─ postgres/
│         └─ seed_gold_demo.sql
│
├─ infra/
│   ├─ compose/
│   │  ├─ docker-compose.yml
│   │  ├─ docker-compose.observability.yml
│   │  ├─ docker-compose.bi.yml
│   │  └─ profiles.md                    # qué levanta cada perfil
│   ├─ grafana/
│   │  ├─ dashboards/
│   │  └─ provisioning/
│   ├─ prometheus/
│   │  └─ prometheus.yml
│   ├─ metabase/
│   │  └─ init/
│   ├─ supabase/
│   │  ├─ migrations/
│   │  └─ policies/
│   └─ rustfs/
│      ├─ config/
│      └─ init/
│
├─ apps/
│   ├─ generator/                        # genera emails sintéticos (raw)
│   │  ├─ README.md
│   │  ├─ pyproject.toml
│   │  └─ src/pqr_generator/...
│   ├─ pipelines/                        # flows Prefect + tasks Dask
│   │  ├─ README.md
│   │  ├─ pyproject.toml
│   │  └─ src/pqr_pipelines/...
│   ├─ ai/                               # extracción/clasificación (silver)
│   │  ├─ README.md
│   │  ├─ pyproject.toml
│   │  └─ src/pqr_ai/...
│   ├─ dashboard-streamlit/
│   │  ├─ README.md
│   │  └─ app.py
│   └─ admin/                            # opcional: scripts de admin/ops
│      └─ ...
│
├─ db/
│   ├─ sql/
│   │  ├─ schema/
│   │  │  ├─ core.sql
│   │  │  ├─ meta.sql
│   │  │  └─ gold.sql
```

```
│   │   ├─ views/
│   │   └─ rpcs/
│   ├─ README.md
│   └─ migrations/                    # si no usas supabase migrations aquí
│
├─ scripts/
│   ├─ build-diagrams.sh              # puml -> svg
│   ├─ lint.sh
│   ├─ smoke-test.sh                  # e2e minimal
│   └─ local-reset.sh
│
├─ tests/
│   ├─ contracts/                     # validación schemas/specs
│   ├─ pipelines/
│   └─ e2e/
│
└─ notebooks/                         # para docencia (opcionales)
    ├─ 00-intro.ipynb
    ├─ 01-raw-to-bronze.ipynb
    ├─ 02-bronze-to-silver-ai.ipynb
    └─ 03-silver-to-gold-kpis.ipynb
```

Reglas de oro (para que no se degrade con el tiempo)

1)

docs/

es “autoridad documental”

- Arquitectura, ADR, runbooks y guías viven ahí.
- Nada de scripts sueltos dentro de docs.

2)

data/contracts/

es “contrato, no dataset”

- Solo specs, schemas, governance.
- Samples mínimos van en data/samples/.

3)

infra/

solo infraestructura

- compose, provisioners, config, dashboards
- Cero lógica de negocio en infra/.

4)

apps/

es código de producto

- generator, pipelines, ai, dashboard
- Cada app con pyproject.toml para packaging y tests.

5)

db/

separa SQL del runtime

- schema, views, rpcs, seeds
- reproducible, versionado y auditable.

Perfiles recomendados de Docker Compose (MVP-friendly)

- core: rustfs + postgres/supabase + dask + prefect
- bi: metabase + streamlit
- obs: prometheus + grafana
- all: todo

Esto evita que estudiantes levanten 12 contenedores cuando solo están en día 1.

Convenciones recomendadas (para el repo)

- ADR: ADR-####-slug.md

- Diagramas: .puml + export .svg en docs/.../diagrams/
- Contratos: data/contracts/v1/...
- “Run-first”: todo pipeline recibe run_id y escribe por run

Qué te sugiero hacer inmediatamente

1. Crear el árbol y commits “baseline”
2. Mover SYSTEM-OVERVIEW.md, ADRs y diagramas a la estructura
3. Crear data/contracts/v1/ y poner ahí contratos
4. Luego arrancamos Día 1 con infra/compose/docker-compose.yml