

Batch# 1949 PT LeWagon Data Engineering , Aug 2025

# Modern Banking DWH

*Turning fragmented banking data into a reliable,  
analytics-ready warehouse*

Deniz , Emma, Lina

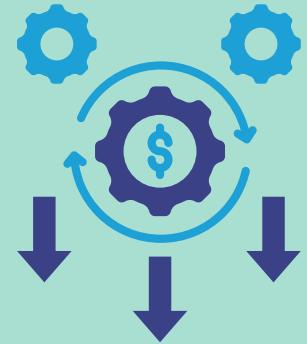


# Why this matters: Problems & Impact



## Siloed, Slow, and Unmanaged Data

- Data scattered across systems makes analysis slow and unreliable
- Slow batch ingestion delays insights, weakens fraud detection, and limits timely market response



## Higher Costs, Lower Trust

- Operational costs rise due to duplicate ETL pipelines and inefficient storage.
- Without centralized governance and lineage, confidence in analytics drops.



## Fraud Loss Multiplied

- When banks absorb €4.12 for every €1 of fraud, customers feel it too — through higher fees, tighter controls, and lower trust
- In EMEA, 52% of fraud losses now occur via digital channels, surpassing physical fraud for first the time .

# Our Solution

*A Scalable, Modular Data Platform*

01.

## Data Vault + 6NF Modeling

Replaces siloed, inconsistent data with a unified structure optimized for auditability, flexibility, and analytics.

02.

## Modular Cloud Architecture

Separates ingestion, staging, modeling, and visualization — supports future scale and ML use.

03.

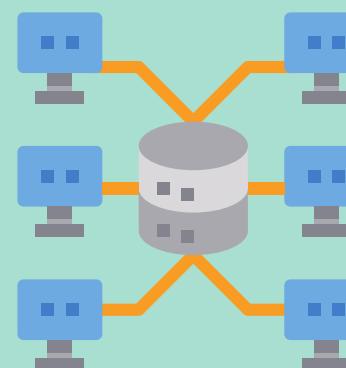
## CI/CD + Orchestration

Enables automated updates, reliable testing, and faster delivery through GitHub Actions + Airflow.



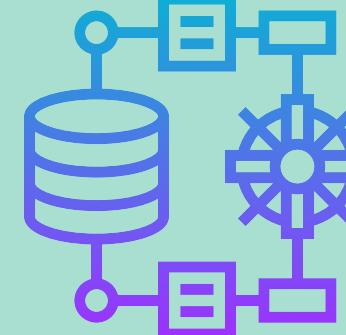
# Data Vault + 6NF

*A robust and flexible foundation for scalable analytics*



## Data Vault

- Separates raw data from business logic
- Flexible & scalable — schema changes don't break the model
- Built for auditability and long-term growth



## 6NF Modeling

- Clean separation of every changing attribute
- Enables versioned, historical analysis (e.g. setting/plan changes)



## Impact on Analytics

- Time-based insights — enables history-aware analytics
- Full lineage — every metric can be traced back to source & timestamp

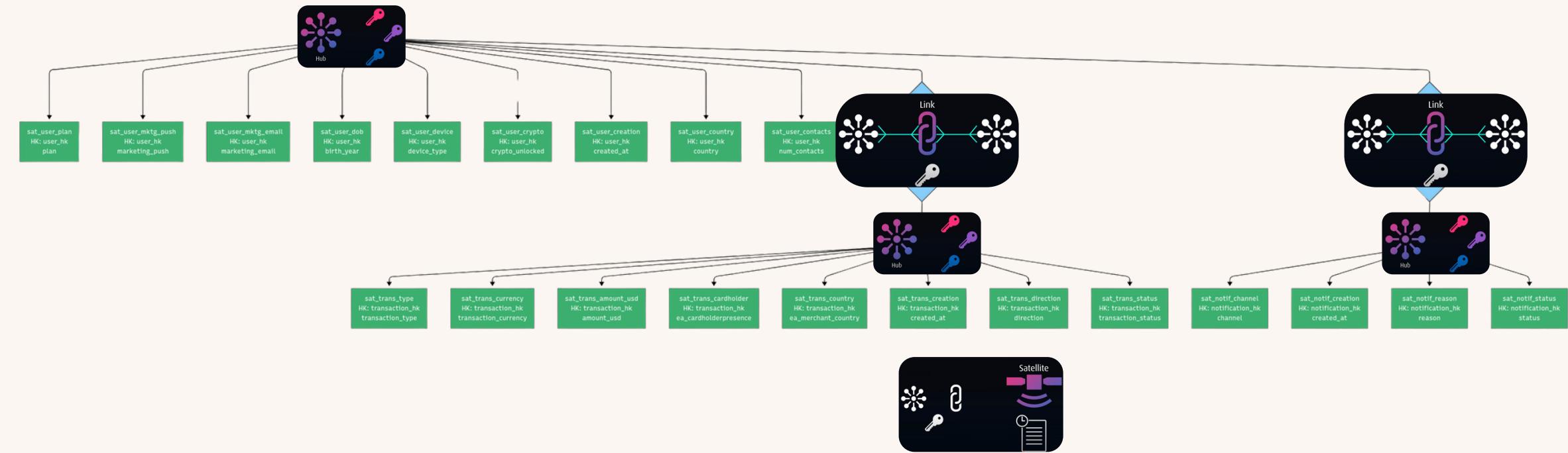
# Our vault architecture

*Modelling data using Hub, Links and Satellites: Present all the data, all the time*

## Why vault?



- **Incremental:** add sources & changes with minimal rework
- **Raw first:** store unchanged history, apply rules later
- **Traceable:** full lineage & auditability built in
- **Automatable:** insert-only, parallel loads



# Our vault architecture

*Incremental, insert-only, parallel; BI on marts*

## How it works?

*Philosophy: Raw first, rules later; keep all data + lineage*

Layers: Staging → Raw Vault →  
(Business Vault) → Marts



Hub = business key



Link = relationship



Satellite = descriptive history

ETL



- Load hubs: new business keys → HK,
- Load Links - new relations → LK+ HKs
- Load Satellites on Hubs/Links

Incremental, insert-only, parallelizable

# Continuous Integration & Development (CI/CD)

## *Safe & Automated Development*

### Continuous Integration



GitHub Actions

- CI runs on pull requests to dbt models only
- Builds on ubuntu-latest
- Uses Poetry to install dependencies
- Injects GCP credentials securely
- Automatically rewrites profiles.yml for dataset selection
- Runs dbt deps, dbt build, and SQLFluff for linting
- Fails the build and blocks merge if anything breaks

A screenshot of a GitHub pull request page titled "Testing CI for the presentation. #23". It shows a green "CI ready" status bar at the top. Below it, there's a comment from "mitchell" and a message from "dbt CI". A summary section indicates "Some checks haven't completed yet" and lists a failed "dbt CI/dbd-ci (pull\_request)" check. A note says "No conflicts with base branch". At the bottom, there's a "Merge pull request" button.

A screenshot of a GitHub CI pipeline summary for the same pull request. It shows a green "Status: abc-ml" bar. The pipeline steps listed are: Get up job, Run unit/integration tests, Set up Python 3.11, Install dependencies, Setup GCP service account key, Setup dbt profile sync, Run dbt checks, Lint dbt models, Open Source Python 3.10, Post-Run actions/theadout/ind, and Complete job.

```
17:32:06 Failure in model test (models/test.sql)
17:32:06 Database Error in model test (models/test.sql)
CREATE VIEW columns must be named, but column 1 has no name at [line]
compiled code at target/run/dbt_neobank/models/test.sql
17:32:06
17:32:06 compiled code at target/compiled/dbt_neobank/models/test.sql
17:32:06
17:32:06 Finished running 8 table models, 231 data tests, 31 view models in 8 hours 8 minutes and 12.22 seconds (12.22s).
17:32:06
17:32:06 Completed with 268 errors, 0 partial successes, and 0 warnings!
17:32:06
17:32:06 Failure in model test (models/test.sql)
17:32:06 Database Error in model test (models/test.sql)
CREATE VIEW columns must be named, but column 1 has no name at [line]
compiled code at target/run/dbt_neobank/models/test.sql
```

### Continuous Development

- Streamlit cloud provides CD

# Orchestration

*Triggering ingestion, transformation, and model runs on schedule*

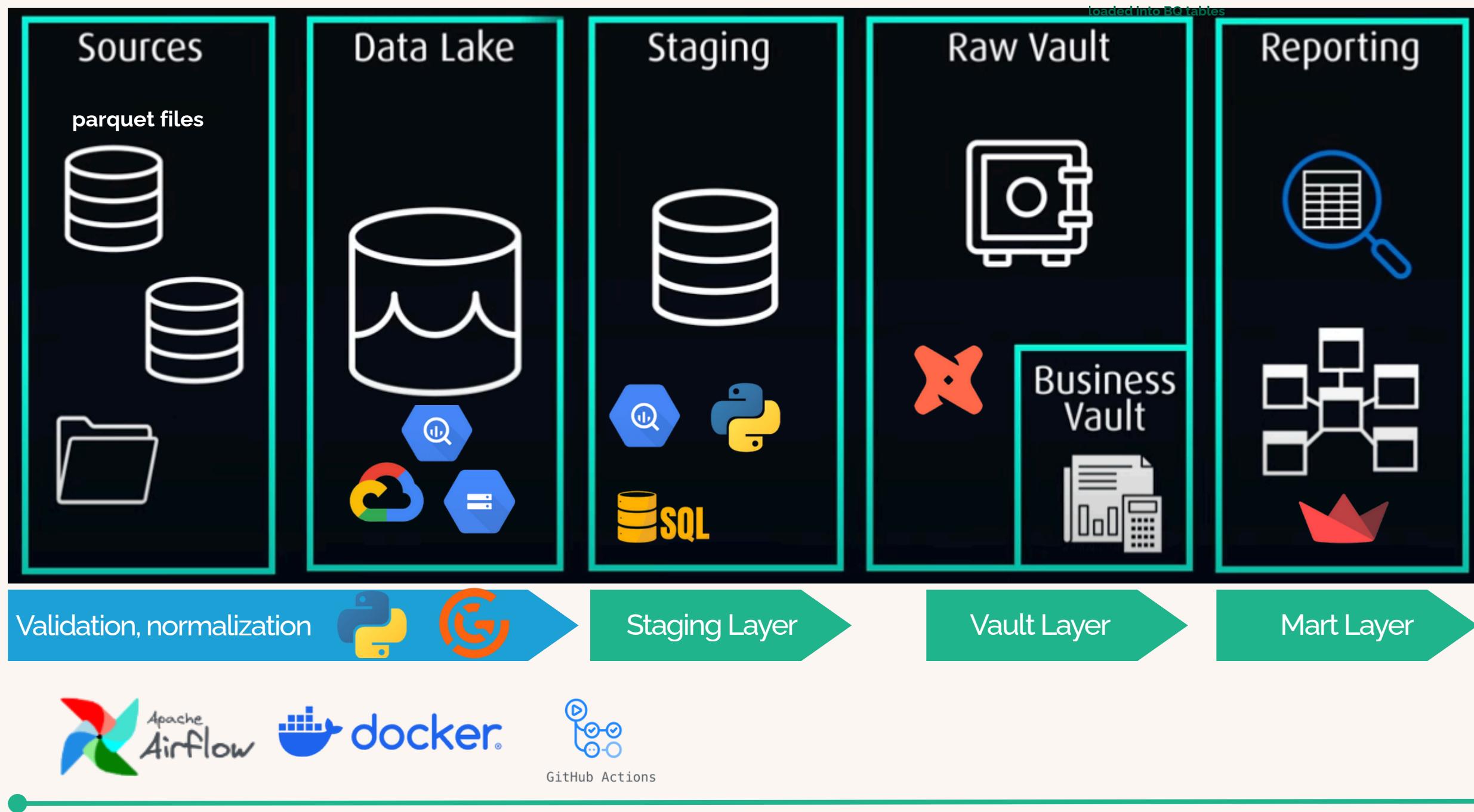


- DAGs split by stage: ingest, staging, and vault
- Each DAG handles a distinct layer of the pipeline
- Can be triggered manually or scheduled
- Airflow ensures dependency order and scalability

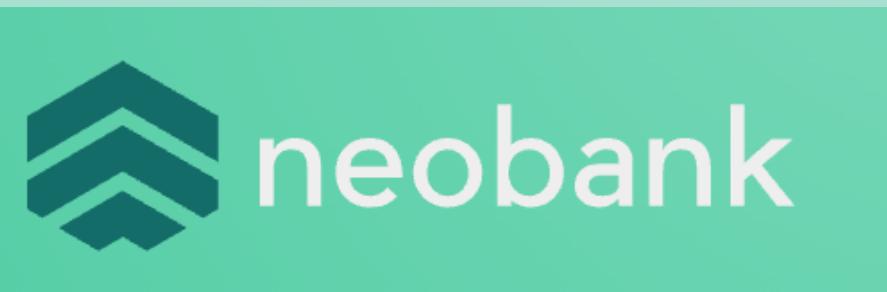
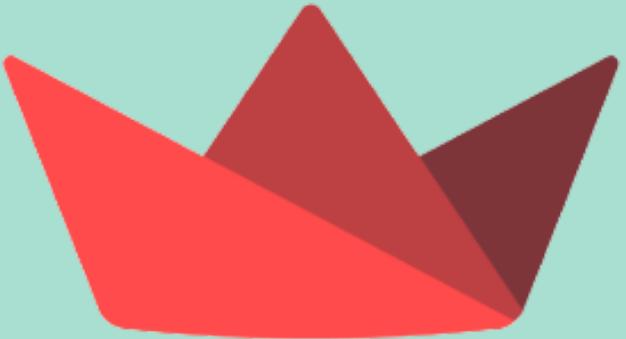
DAG	Owner	Runs	Schedule	Last Run	Next Run
dbt_ingest_dag_test	airflow	1	None	2025-08-07, 00:39:58	
dbt_staging_dag_test	airflow	3	None	2025-08-07, 00:26:28	
dbt_vault_dag_test	airflow	0	None		
generate_synthetic_data_dag	airflow	0	@daily	2025-08-06, 00:00	

# Our Pipeline in Action

*How we ingest, transform, model & visualize the data*



# Visualization



# Success Metrics

*Turning complete data ingestion into actionable insights –  
with automation built in.*

- 100% ingested data sources
- Tested all models
- Created visualizations per each mart
- Provided MVP + CICD integration
- Fully orchestrated by airflow!

# Dev Team



**Deniz**  
**SOFTWARE ENG**



**Emma**  
**DATA ANALYST**



**Lina**  
**MATHEMATICIAN**

# Architecture as a Team

## *External and Internal Challenges*

- 3 developer VMs on GCP for local experimentation
- Central GitHub repository for collaboration & version control
- 1 production VM for deployment & dashboard hosting

### Workflow:

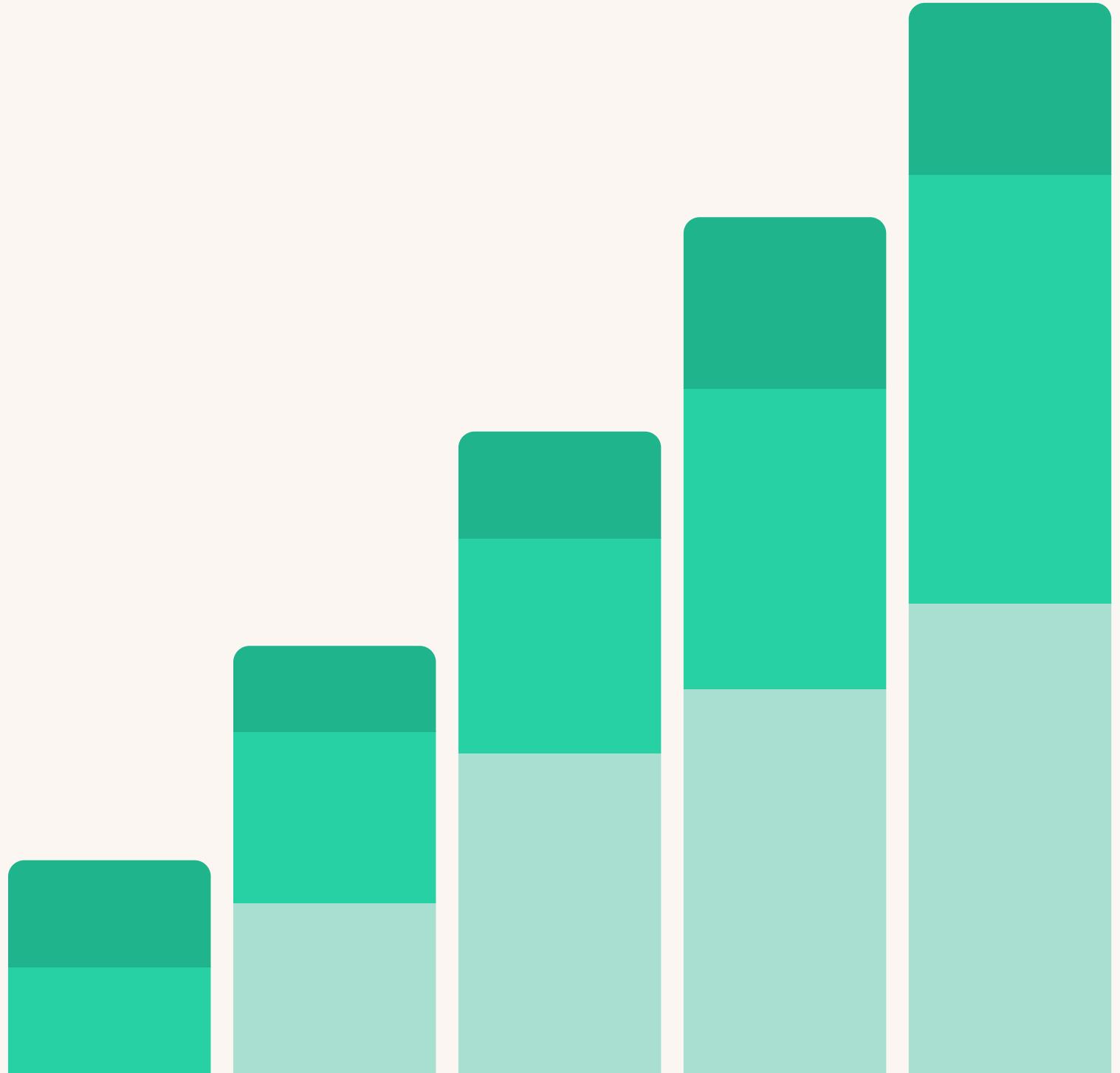
- Developers work in branches
- Submit Pull Requests to dev
- After validation, merge to main



# Keep up

How we plan to scale and improve our data platform in the short term:

-  Enable API access to our dashboard
- → Use LLMs to assist with natural language queries  
(e.g., "Show churn by plan last month")
-  Introduce ML models
- → Predict fraud patterns and analyze churn risk
-  Add Airbyte for real-time ingestion
- → Automate ingestion from multiple sources to reduce latency and manual work



# Thank you!

