Technical Documentation — Customer Churn Reporting & Visualization Platform

Project Name: Customer Churn Reporting & Visualization Platform
Track: Data Engineering — DEPI

Objective:
Develop a complete data engineering project that enables understanding of customer churn behavior for a telecom company. The project includes the full data lifecycle — ingestion from MySQL, transformation and modeling using DuckDB and dbt, automation with Airflow, and visualization with Dash.

1. Setup Guide

Below are the detailed steps to set up the environment and tools.

Tool	Purpose	Installation Command
Python ≥ 3.9	Main programming language	python.org/downloads
MySQL Server	Transactional source database	dev.mysql.com/downloads/mysql
DuckDB	Local analytical data warehouse	pip install duckdb
Apache Airflow	Pipeline orchestration	pip install apache-airflow
dbt-duckdb	Transformation & modeling	pip install dbt-core dbt-duckdb
Dash & Plotly	Interactive visualization	pip install dash plotly
Pandas	Data manipulation	pip install pandas
MySQL Connector	Connect Python with MySQL	pip install mysql-connector-python

Project Structure:

```
Project_Customer_Churn_Reporting__Visualization_Platform/
│
├─ data/
│   ├─ customer_churn.csv
│   ├─ customer_churn_warehouse.duckdb
│
├─ sql/
│   └─ schema.sql
│
├─ src/
│   ├─ data_generator.py        # generate synthetic churn dataset
│   ├─ load_to_mysql.py         # load CSV data into MySQL
│   ├─ mysql_to_duckdb.py       # migrate data from MySQL → DuckDB
│   ├─ db.py                    # database helper functions (DB-API)
│   ├─ etl.py                   # ETL logic (extract-transform-load)
│   ├─ plotly_report.py         # generate static HTML visualizations
│   ├── app_dash.py              # Dash dashboard app
│
├── customer_churn_dbt/
│   ├── dbt_project.yml          # dbt project configuration
│   ├── models/
│   │   ├── staging/
│   │   │   ├── staging_customer_churn.sql
│   │   │   └── staging_customer_churn.yml
│   │   ├── intermediate/
│   │   │   └── intermediate_customer_metrics.sql
│   │   ├── marts/
│   │   │   └── fact_churn.sql
│
├── airflow_dags/
│   └── churn_pipeline_dag.py    # Airflow DAG to orchestrate the pipeline
│
├── docs/
│   ├── Technical_Documentation_Customer_Churn.pdf
│
├── outputs/
│   ├── churn_report.html
│
└── README.md                    # project description & setup instructionsEnvironment
```

Configuration:

All credentials and database connection details can be stored in a **.env** file as follows:

```
MYSQL_HOST=localhost MYSQL_USER=root MYSQL_PASSWORD=your_password
MYSQL_DB=customer_churn_db DUCKDB_PATH=./data/customer_churn_warehouse.duckdb
```

2. Architecture Overview

The architecture consists of five layers:

1■■ **Data Ingestion Layer** — CSV data is loaded into MySQL using Python & Pandas.
2■■ **Data Warehousing Layer** — Data is migrated into DuckDB for analytical processing.
3■■ **Transformation Layer** — dbt manages transformations, testing, and documentation.
4■■ **Orchestration Layer** — Airflow automates daily runs for ETL and dbt workflows.
5■■ **Visualization Layer** — Dash provides an interactive dashboard connected to DuckDB.

3. Documentation and Diagrams

• **Data Flow Diagram:** Shows data movement from CSV → MySQL → DuckDB → dbt → Dash. • **ERD:** Star schema with fact_churn and dimension tables (dim_customer, dim_service, dim_geography).
• **dbt Model Lineage:** Visualizes transformation dependencies from staging → intermediate → mart models.
• **Setup Guide:** Contains step-by-step environment and dependency instructions.
• **Architecture Overview:** Explains the purpose and logic behind each pipeline component.

4. Conclusion

This project demonstrates the end-to-end data engineering process using tools covered in the DEPI track — from ingestion (Python, MySQL) to modeling (dbt, DuckDB), automation (Airflow), and visualization (Dash). It provides a reproducible, well-documented, and scalable pipeline for churn analytics in a telecom scenario.