VPBank powered by aws

VPBank Technology Hackathon 2024

General Brief

Please fill up this table and use this document as a template to write your proposal.

Challenge Statement	Customer 360
Team Name	Cerebro

Team Members

Full Name	Role	Email Address	School Name (if applicable)	Faculty / Area of Study	LinkedIn Profile URL
Phạm Đình Gia Dũng	Cloud Engineer	dungpham.020901@gmail.com	Hanoi University of Science and Technology	Data Science and Artificial Intelligence	https://www.linkedin.com/in/giadungpham/
Hà Thị Phương Hoa	Data Analyst	htphoa95@gmail.com	National Economics University	Corporate Finance	https://www.linkedin.com/in/hannah-ha-a44984188/
Đỗ Tuấn Anh	Data Analyst	tuananhdoo2505@gmail.com	Hanoi School of Business and Management	Management of Enterprise and Technology	https://www.linkedin.com/in/aarondo2505/
Phạm Bá Hiếu	Full Stack Developer	phamhieutb.dev@gmail.com	Academy of Cryptography Techniques	Information technology	https://www.linkedin.com/in/phamhieus

Content Outline

Solutions Introduction

What is your solution?

Our solution is to design and build a customer data infrastructure for VPBank's ecosystem, enabling real-time, unified, and accessible data across various systems.

Discuss main features clearly.

1. Ingest and process data from diverse sources, including structured, semi-structured, and unstructured data and implement a centralized data repository for unified data storage and management.
2. Leverage a combination of SQL, NoSQL, and Big Data technologies to handle design data models aligned with business requirements and supporting various analytics use cases; Optimize data models for efficient data retrieval and analysis by departments and microservices.
3. Implement robust data governance to ensure data quality, integrity, and security: Establish data access controls and role-based permissions to safeguard sensitive customer information.
4. Provide comprehensive data documentation clearly defining data elements, sources, and usage.
5. Develop a robust API layer for seamless data access by various internal systems and applications.
6. Build a unified Customer 360 platform to provide a holistic view of each customer's profile and interactions.

Impact of Solution

How does your solution benefit the society / the target audience?

1. Business Intelligence and Data Analyst:

- Provides clean and standardized data for report generation and enhances query efficiency using data visualization tools (BI Tools).
- Save time processing data to exploit and propose business use cases.

2. Sales, Marketing and Customer Experience:

- Delivers a comprehensive customer profile and supports business decision-making.
- Bridges the gap between Sales, Marketing, and Customer Experience activities.

3. Data Scientist and AI Engineer:

- Customer 360 provides structured and unstructured data through an API interface, enabling users to easily build machine learning models, predictions, and customer segmentation.

Why is your solution a good solution? How is it better than existing solutions in the market / competitors? What is your solution's competitive advantage / unique selling point?

1. The solution design is a combination of AWS and open-source technologies, resulting in optimized infrastructure deployment time and operating costs.
2. Integrated transformation flows and data transformation are designed to minimize development time with go-live mechanisms for new features tailored to the specific needs of each department.
3. Robust governance and security protocols are in place to ensure the organization maintains strict adherence to all relevant banking industry data regulations and standards, even with the majority of its infrastructure operating within a cloud environment.
4. The selected big data technologies offer high scalability and excellent connectivity between system components.

Deep Dive into Solution

1. **Sourcing simulation:** Leveraging AWS products and third-party technologies to simulate realistic source data, which are:
    a. Utilize Amazon S3 to store CSV and Excel report files from the accounting department and audio files from customer service (call recordings).
    b. Employ an FTP server to store and provide XML files, simulating data from the T24 Core Banking system (customer data).
    c. Use AWS Lambda and API Gateway to simulate third-party data APIs.
    d. Establish a database to replicate data from ERP systems like SAP and other products within the ecosystem.
    e. Use Kafka (May be managed by MSK) to simulate real-time customer behavior data with real-time updates.

2. **Continuous Development:** Leverage Git and integrated AWS version control features to streamline data transformation model development, quickly apply changes to AWS infrastructure, and differentiate between Production and Dev/UAT environments.
    a. GitHub Actions/Gitlab Pipelines enable rapid deployment of dbt data transformation models to UAT environments for ad-hoc purposes and model performance evaluation.
    b. AWS Glue Version Control: Allow local development and seamless application of local changes to infrastructure.
    c. Implement Airflow with GitSync to directly synchronize newly developed DAGs.

3. **Scalability:** Employ AWS Auto Scaling to automatically scale the compute capacity of the cluster hosting the web application, ensuring optimal resource utilization based on real-time user demand. Achieve cost efficiency by dynamically adjusting resource allocation, minimizing idle resources, and maximizing resource utilization during peak usage periods.

Architecture of Solution

1. **Data Ingestion and Integration pipeline for Relational and Graph Database**
    a. Technologies: AWS Glue, AWS Lambda
    b. Data Extraction, Transformation, and Loading (ETL) Process:
    - AWS Glue executes and orchestrates spark jobs to extract data from various sources, leveraging its extensive range of connectors to ensure seamless data retrieval.
    - AWS Lambda establishes a Kafka consumer. This consumer receives data from Kafka and persists it into the Database

2. **Database**

3.1 Relational Database

a. Technologies: PostgreSQL, RDS

b. Deployment:

- RDS automates hardware provisioning, backup, and database setup, minimizing time-consuming tasks.
- PostgreSQL for a balance between maintenance costs, ability to handle large amounts of Data, stability, and great community support.

c. Database data zone:

- Source: Data from various sources is ingested into the Data Warehouse.
- Staging: Data is cleansed and undergoes basic transformations.
- Intermediate: Data undergoes further middle transformations.
- Mart: Customer 360 data is prepared for end-user consumption.
- Curated: Data is transformed for specific business use cases (analysis, dashboard/report creation).

` **3.2 Graph Database**

a. Technologies: Neo4j, EC2

b. Deployment:

- Neo4j on EC2: Since RDS does not support Neo4j, we deploy Neo4j on an EC2 instance using an Ubuntu environment to streamline the installation process.

- Data Integration: Use AWS Glue to process data from the relational database, transform it into graph-structured data, and integrate it into Neo4j.

3. Data Warehouse and Data Transformation

a. Technologies: dbt, Airflow, ECR, EKS

b. Deployment:

- dbt: Leverages dbt's model-driven approach for data transformation using SQL statements. Models are stored in a GitHub or GitLab repository, enabling direct execution via GitHub Actions or GitLab CI/CD (primarily for Dev/UAT environments and rapid ad-hoc data table creation).
- Airflow: Employs Airflow to schedule and manage data transformation processes. Airflow can retrieve the dbt repository from git, store and directly execute commands locally. Enable GitSync to sync changes from specific branches.
- Deployment: Deploys Airflow and dbt on a Kubernetes cluster managed by Amazon EKS, utilizing EC2 instances as compute nodes.

4. API

a. Technologies: AWS Lambda, API Gateway

b. Deployment:

- API Gateway: Leverages API Gateway to establish endpoints and receive requests. Employs Resource Policy to restrict API access to authorized IP addresses.
- AWS Lambda: Requests are processed by Lambda Functions, utilizing Python scripts and executing on Python Runtime.

5. Web Application

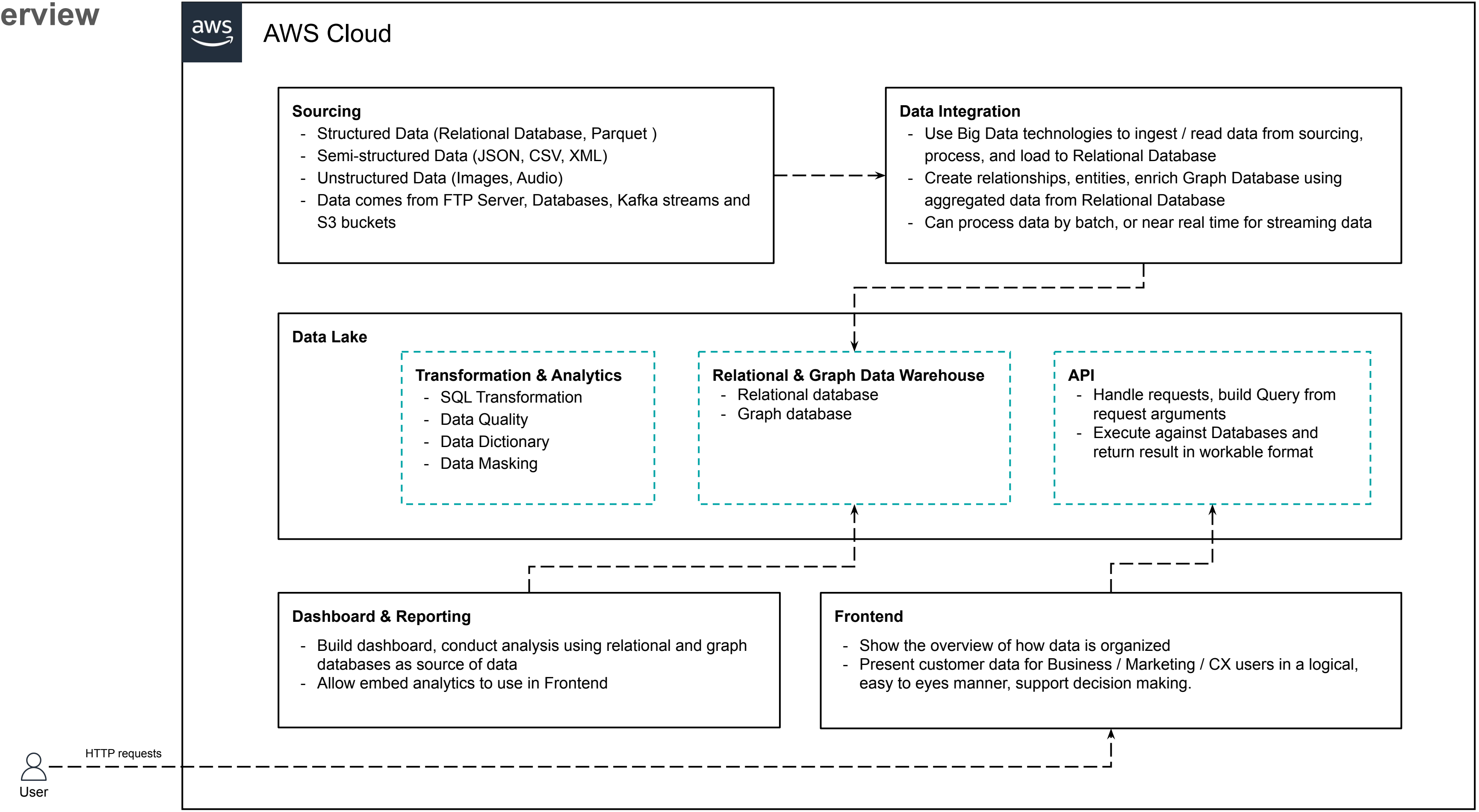
a. Technologies: Django, React JS, EC2

b. Deployment:

- User Interface (UI): Construct a UI to display customer profiles and a search function to filter potential customers. For DS and AI engineer users, show available data, previews, quick filtering and compose API calls corresponding to applied filters for quick data retrieval.

- Deploy on EC2 instances as a web application and API cluster, managed by Auto Scaling groups to balance traffic across instances based on incoming traffic.

Overview



Sourcing

- Structured Data (Relational Database, Parquet)
- Semi-structured Data (JSON, CSV, XML)
- Unstructured Data (Images, Audio)
- Data comes from FTP Server, Databases, Kafka streams and S3 buckets

Data Integration

- Use Big Data technologies to ingest / read data from sourcing, process, and load to Relational Database
- Create relationships, entities, enrich Graph Database using aggregated data from Relational Database
- Can process data by batch, or near real time for streaming data

Data Lake

Transformation & Analytics

- SQL Transformation
- Data Quality
- Data Dictionary
- Data Masking

Relational & Graph Data Warehouse

- Relational database
- Graph database

API

- Handle requests, build Query from request arguments
- Execute against Databases and return result in workable format

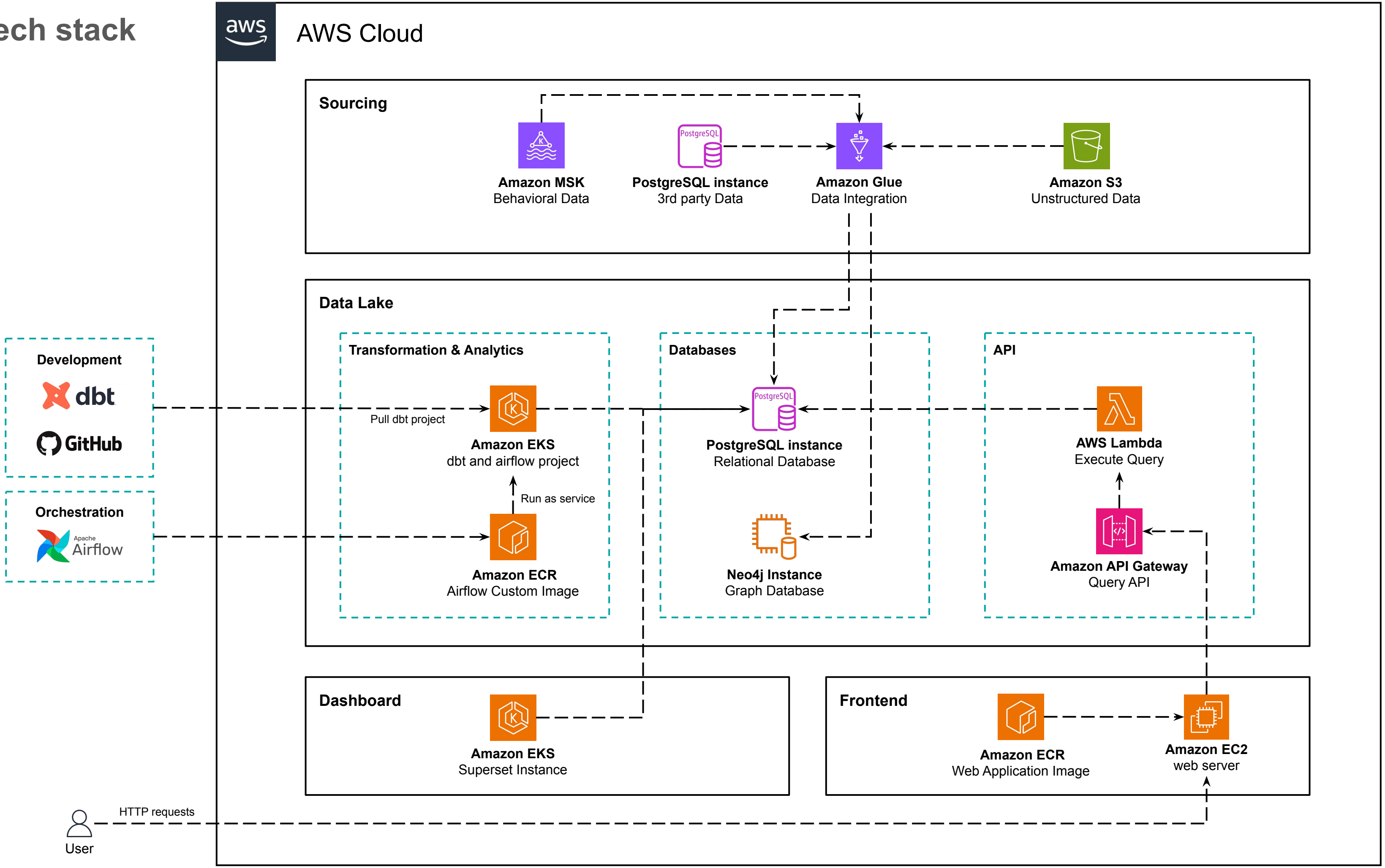
Dashboard & Reporting

- Build dashboard, conduct analysis using relational and graph databases as source of data
- Allow embed analytics to use in Frontend

Frontend

- Show the overview of how data is organized
- Present customer data for Business / Marketing / CX users in a logical, easy to eyes manner, support decision making.

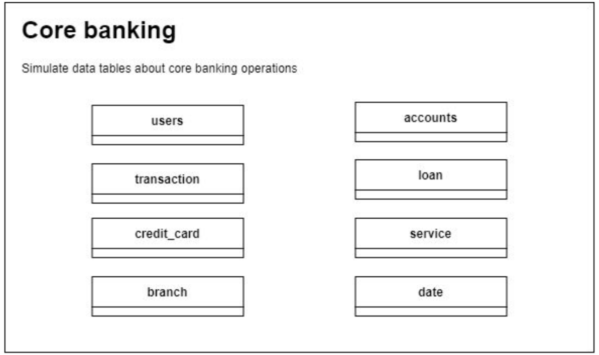
Tech stack



AWS Cloud
Sourcing
Amazon MSK
Behavioral Data
PostgreSQL instance
3rd party Data
Amazon Glue
Data Integration
Amazon S3
Unstructured Data
Data Lake
Transformation & Analytics
Pull dbt project
Amazon EKS
dbt and airflow project
Run as service
Amazon ECR
Airflow Custom Image
Databases
PostgreSQL instance
Relational Database
Neo4j Instance
Graph Database
API
AWS Lambda
Execute Query
Amazon API Gateway
Query API
Development
dbt
GitHub
Orchestration
Apache Airflow
Dashboard
Amazon EKS
Superset Instance
Frontend
Amazon ECR
Web Application Image
Amazon EC2
web server
HTTP requests
User

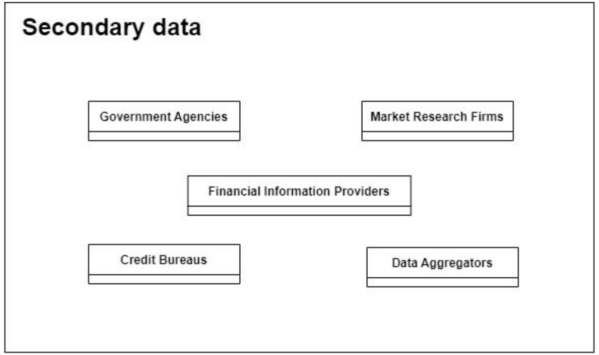
Sourcing

This section visualizes the source data tables that will be centrally stored in the Data Warehouse AFTER being extracted and processed from various data sources across VPBank entire ecosystem. Serving as the foundation for data centralization, data transformation (transaction), and outcome implementation.

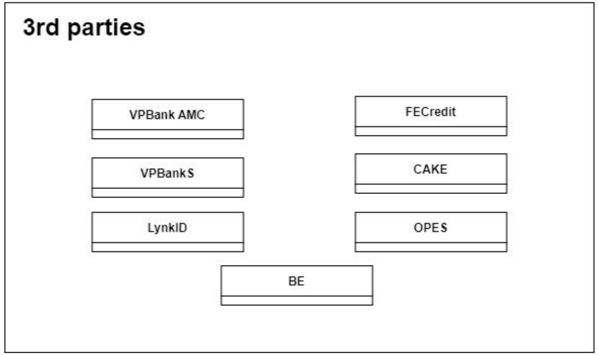


Core banking

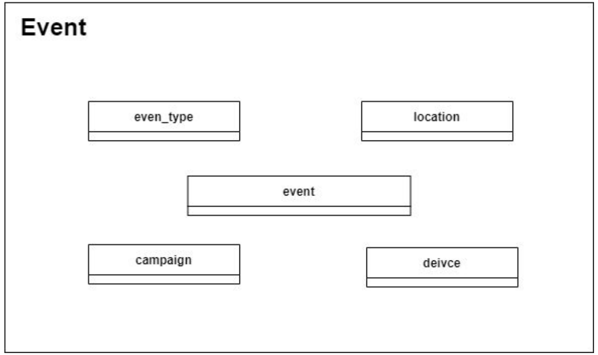
Simulate data tables about core banking operations



Secondary data



3rd parties

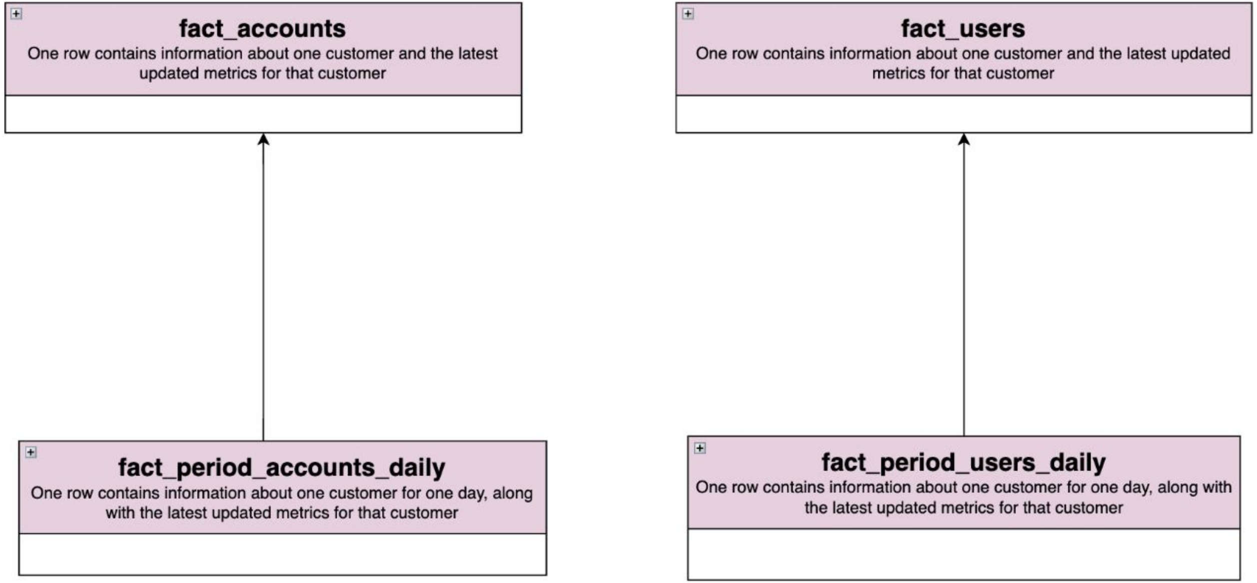


Event

Data Mart

After selecting the appropriate dataset, our team will generate and transform the data tables in the following order:

1/ Source: Data from various sources is transferred to the Data Warehouse.
2/ Staging: Data cleaning and basic transformations are performed.
3/ Intermediate: Intermediate transformations are applied.
4/ Mart: Customer 360 data is prepared for end-user consumption.
These are the mart models our team plans to implement.



fact_accounts
One row contains information about one customer and the latest updated metrics for that customer
fact_users
One row contains information about one customer and the latest updated metrics for that customer
fact_period_accounts_daily
One row contains information about one customer for one day, along with the latest updated metrics for that customer
fact_period_users_daily
One row contains information about one customer for one day, along with the latest updated metrics for that customer