cerebro.

members

Tuấn Anh
Data Analyst

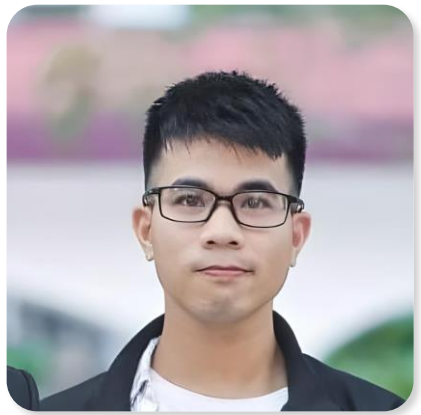
1. Define data models
2. Build dbt models
3. Build dashboards

Phương Hoa
Data Analyst

1. Define data models
2. Build dbt models
3. Build dashboards

jazz Dũng
Cloud Engineer

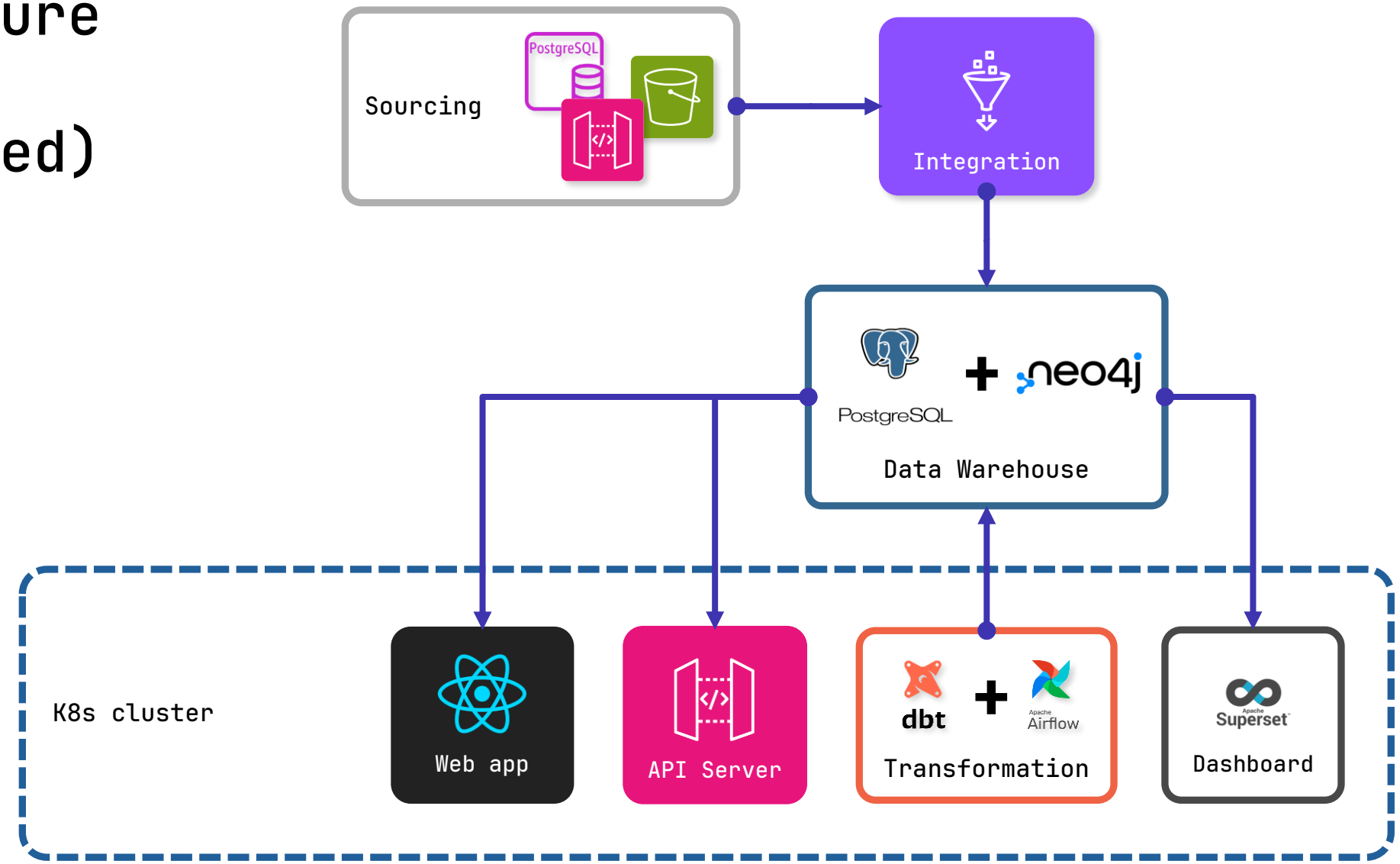
1. Simulate sources
2. Everything AWS
3. Kubernetes



Bá Hiếu
Fullstack Developer

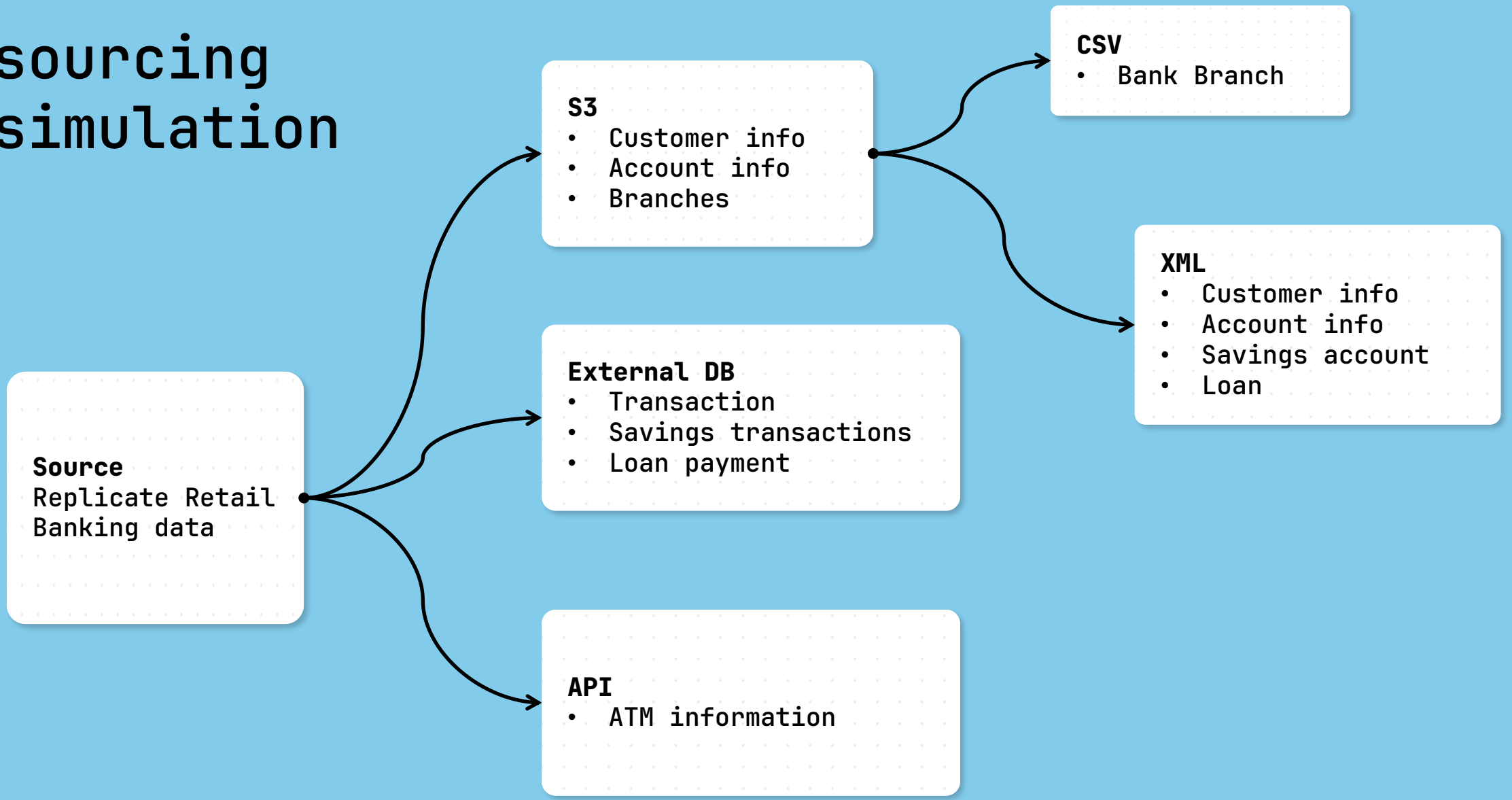
1. Build API server
2. Build web application

architecture diagram (abstracted)



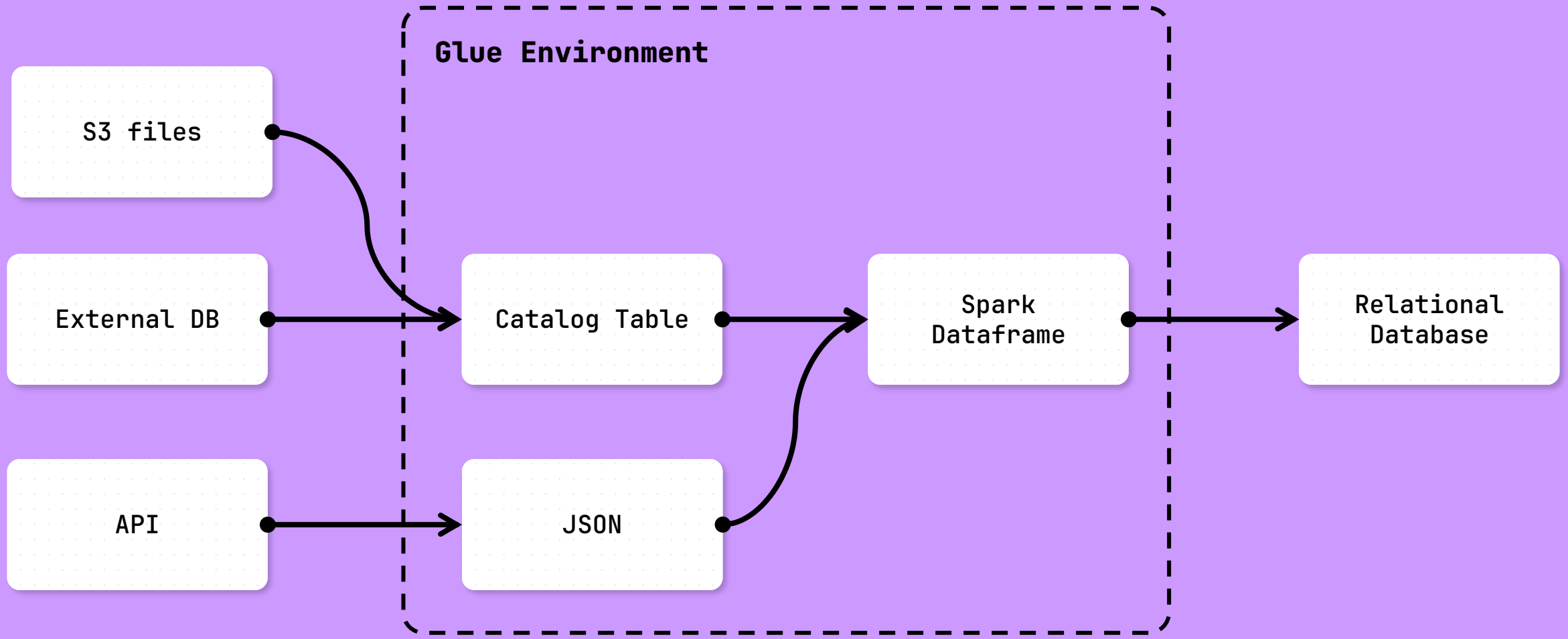
sourcing simulation

sourcing simulation

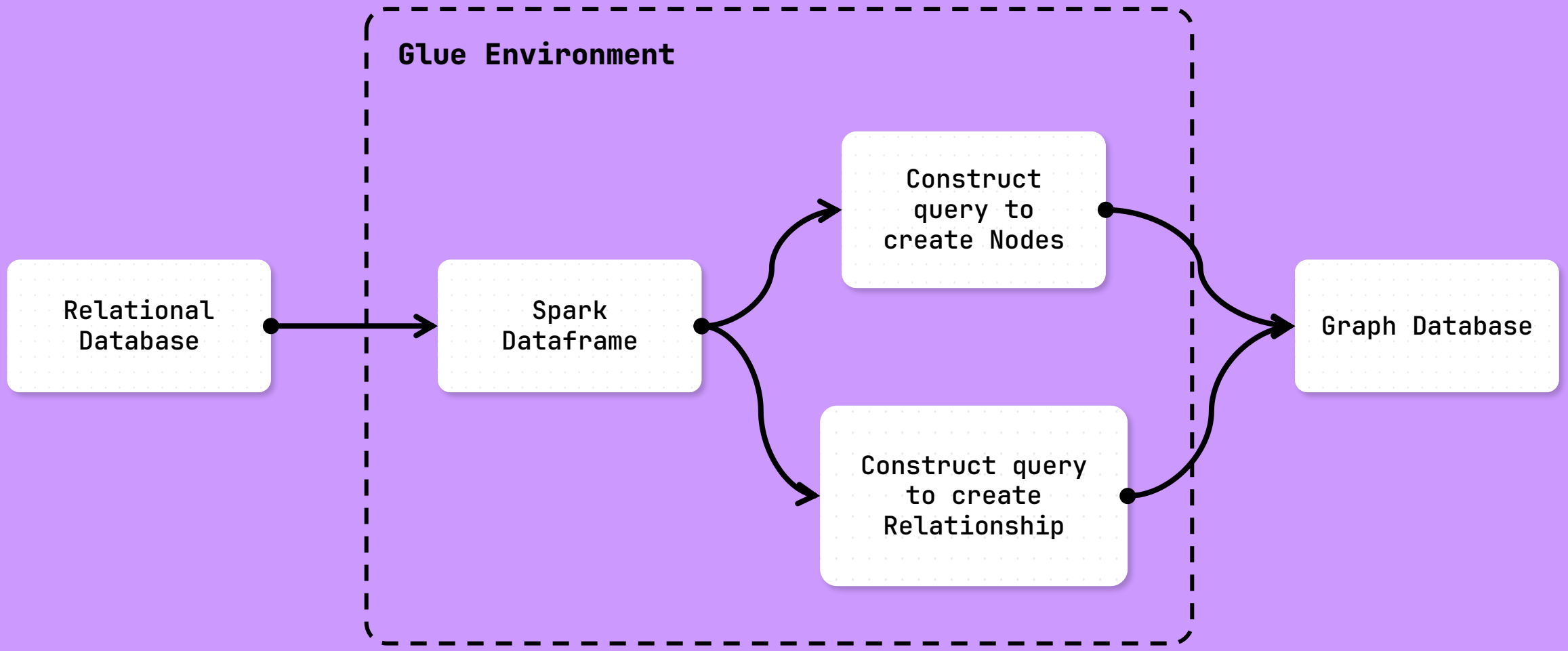


data integration

data integration (source -> relational db)



data integration (relational db -> graph db)



data transformation

data transformation (transformation layer)

Staging

- First transform tables of the imported data
- Cleaning, restructuring, cast data type
- Masking PII data
- Often stored as views

Intermediate

- Intermediate calculation tables
- Often stored as views or ephemeral

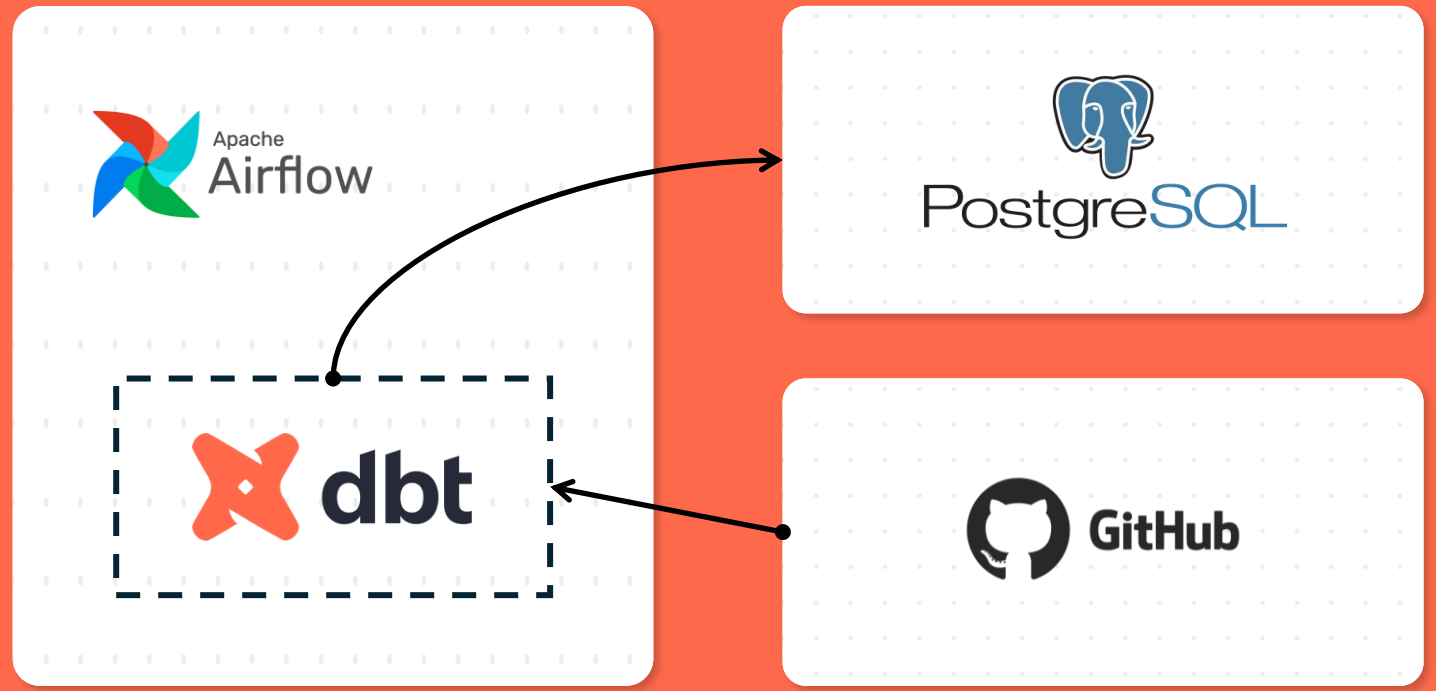
Mart

- Tables designed according to dimensional modeling, oriented for end-user use
- Convenient for users to easily search tables by functions and task
- Store as physical tables

data transformation (orchestration)

Airflow

- Pull dbt project into local repository
- Design DAGs to execute dbt commands



API

api

Frameworks and features

- Built on the Django framework
- Following a three-tier architecture
- Allow filters, sorting, paginate

```
http://customer360/api/customers/?
    query={
        "gender":{"exact": "M"},
        "age":{“lessThan":45}
    }
    &order_by={
        "columns":["age"],
        "type":["asc"]
    }
```

Sample API call

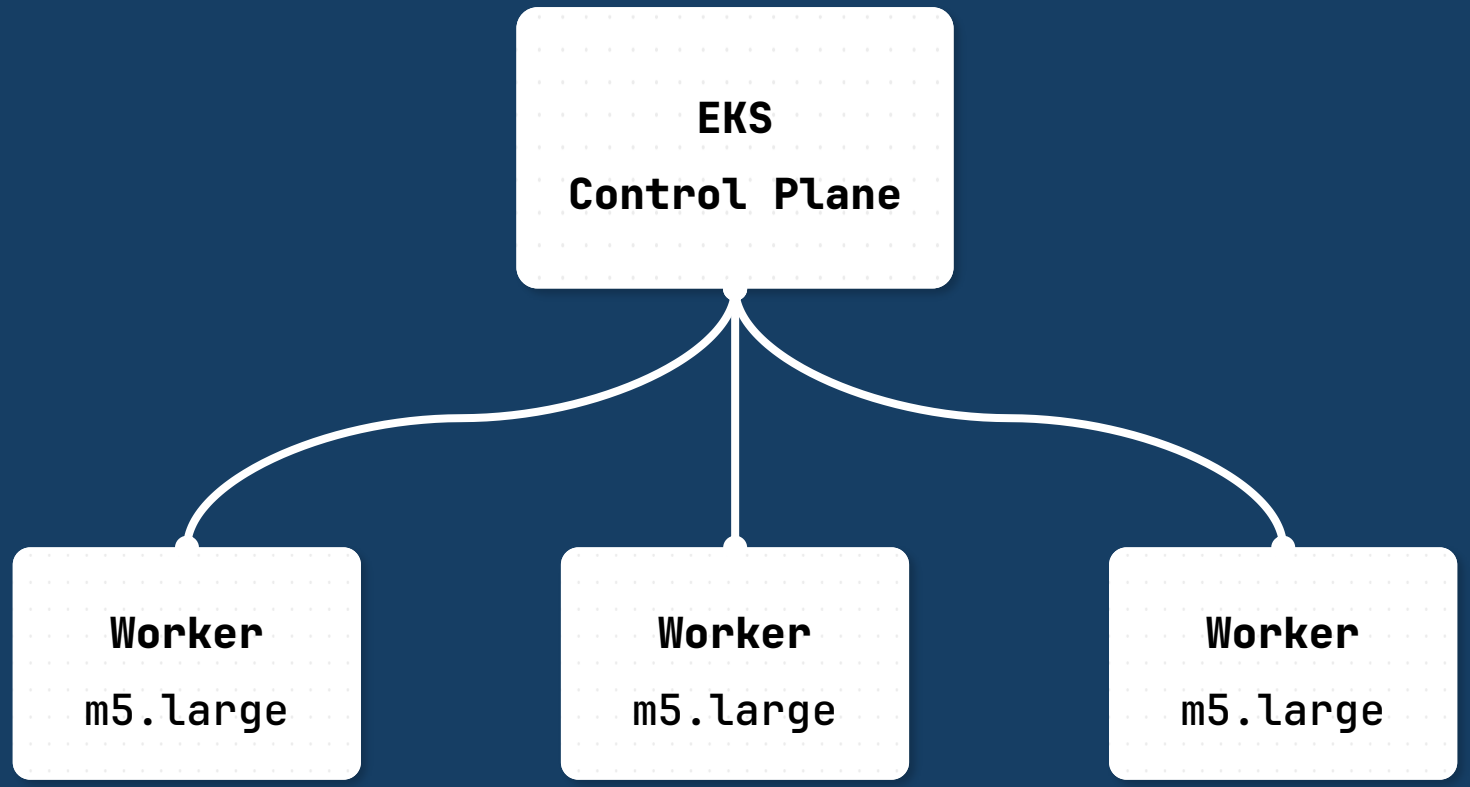

```
{
    "customer_id": "C183",
    "identity_id": ”9a5a54a1b5d2165f207c8d3”,
    "last_name": “Dũng”,
    "first_name": “jazz”,
    "date_of_birth": "2001-09-02",
    "gender": “M”,
    "address": -761,
    "phone": "Thành phố Lạng Sơn, Lạng Sơn"
},
{
    "customer_id": "C183",
    "identity_id": ”9a5a54a1b5d2165f207c8d3”,
    "last_name": “Dũng”,
    "first_name": “jazz”,
    "date_of_birth": "2001-09-02",
    "gender": “M”,
    "address": -761,
    "phone": "Thành phố Hà Nội"
},
```

data visualization
+
web application

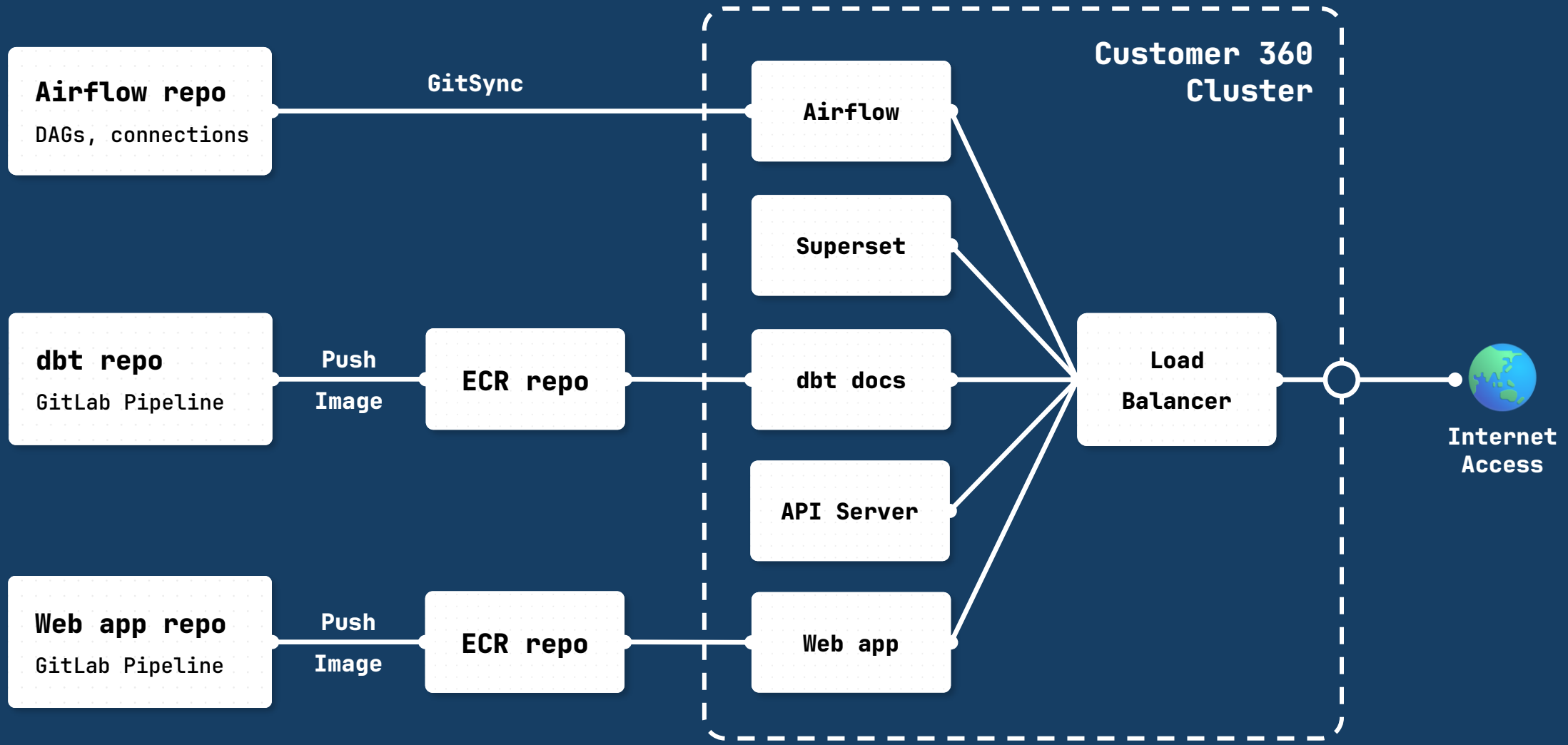
I think that it is best that I show you guys the real things

deployment

deployment (k8s cluster)

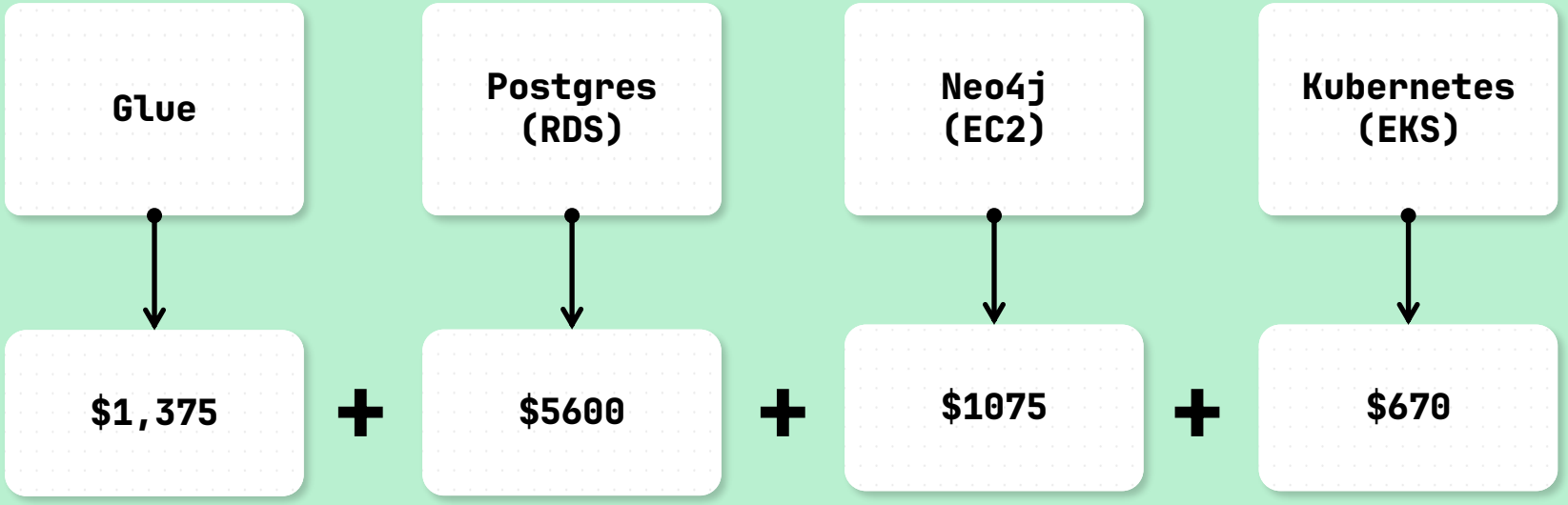


deployment (k8s services)



cost estimation

cost estimation (eks)



= **$8,795** (every month)

future improvements

future improvements (infrastructure)

Setup infrastructure with Terraform

- Infrastructure as code
- Parameterized deployment
- Disaster recovery

Use Lambda functions to automatically:

- Scale down / turn off hardwares on non-running periods
- Scale up / turn on hardwares on spike periods

future improvements (data integration)

Handle streaming data with Table formats and AWS Glue

- Use table format to create a schema definition for streaming records
- Then create glue jobs to load them into database (batch or near realtime)

future improvements (database)

Move to a columnar database (Redshift)

- Better performance
- Better scalability
- Better integration with AWS services
- I know you would love to try RA3 (SSD for hot data, S3 for cold data)

that’s all

```
QnA.start(
    time_limit=300s
)
```

future improvements (lakehouse: better)

- Data Integration
- Transformation
- Build big tables

AWS Glue

- Compute by spark instead of relational DB (Serverless, faster)
- Use SparkSQL

- Data Lake
- Data Warehouse
- Unstructured Data

AWS S3

- Cheaper storage
- Excellent for backup and historical data
- All data in 1 place

- Table format
- Metadata

Apache Iceberg

- Compute by spark instead of relational DB

- Query engine
- Analytics

AWS Athena

- Support queries on file-format and table-format
- Sort-of dbt support (Not production ready)

future improvements (lakehouse: better)

If you dare to step out of the AWS ecosystem

databricks

- Do most of the things above within a single coherent platform
- Plus support for dbt!

but for the sake of a hackathon sponsored by AWS, let’s just pretend I did not said that