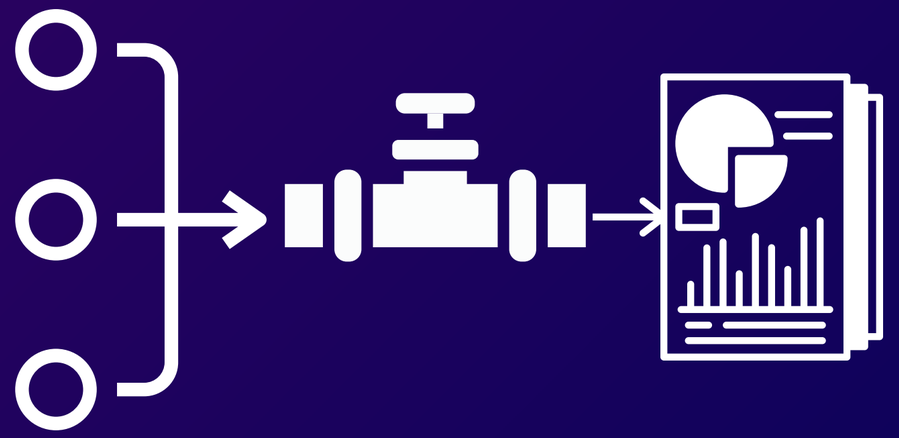
GlassFlow

No Kafka, No JVM: Shaping the future of real-time data pipeline



Bobur Umurzokov

Developer Advocate at GlassFlow | Ex-Dev Lead at Microsoft | Microsoft MVP

Connect on LinkedIn


APISIX

Bob Dreamer

Data Engineer at mid-sized DreamTogether company.

Experience

3+ years at different scale companies.

Education

BS in Computer Science.

Skills

Python, Spark, AWS.

Learning

AI, Streaming tools and technologies

***Not open to work**

ID	Model	Fuel Type	Mileage
2539	BMW	petrol	148686

ID: 123
Model: BMW
Fuel Type: petrol
Mileage: 148686
Suggested Price: 20 000 USD

I will build an AI-powered streaming data pipeline...

Okay, we are curious about the outcome...

Batch Processing

- Processes large volumes of data all at once
- May required dedicated staff to handle issues

Stream Processing

- Analyses streaming cross-device data in near real-time
- The data output rate must be just as fast as the input rate

Real-time data streaming pipeline

Data Source

Relational Databases

Data streams

Applications

Real-time data ingestion
Stream processing
Transformation
Aggregation
Filtering
Enrichment
Run ML models

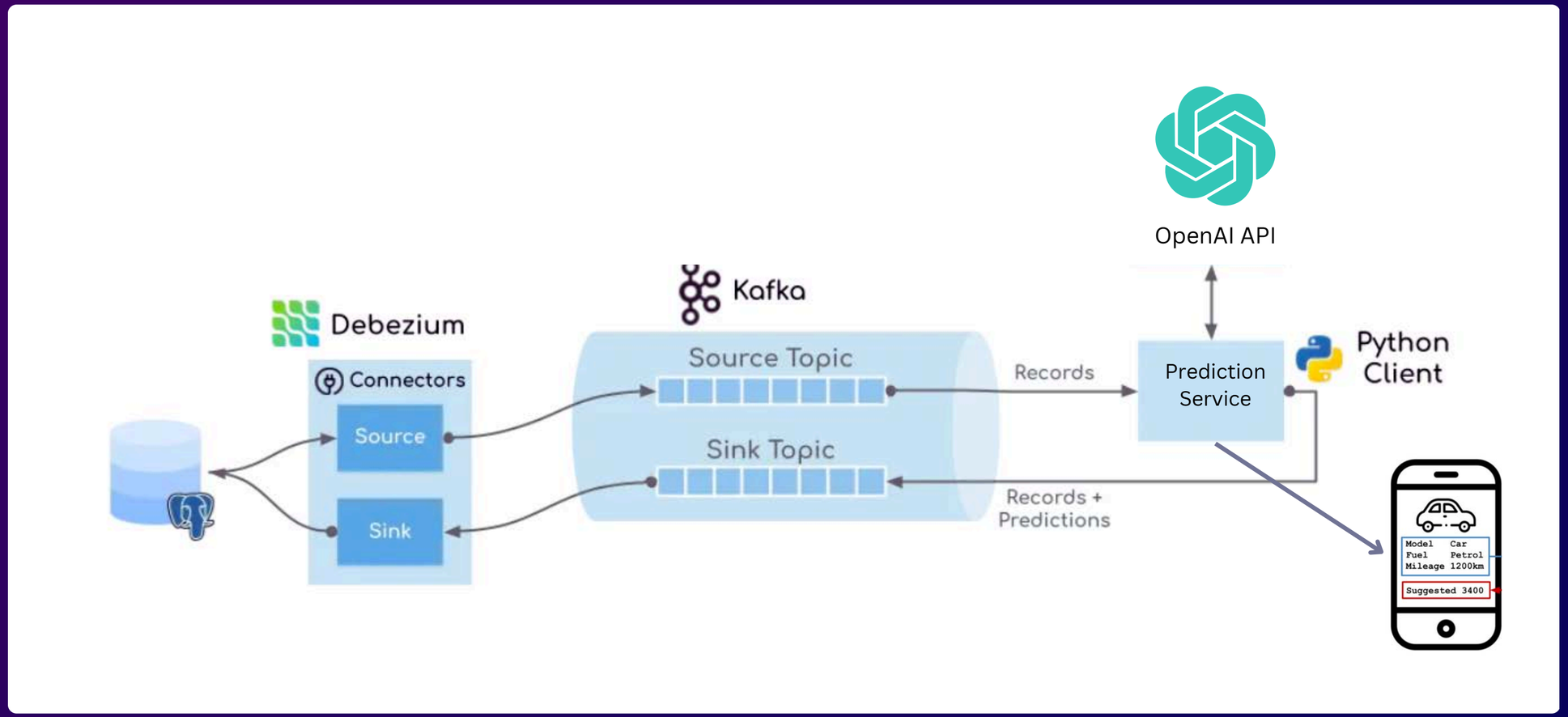
Data Sink

Analytics and BI tools

Data Driven Apps

Data Warehouse

1 Solution


OpenAI API
Kafka
Debezium
Connectors
Source
Sink
Source Topic
Sink Topic
Records
Records +
Predictions
Prediction
Service
Python
Client
Model Car
Fuel Petrol
Mileage 1200km
Suggested 3400

Challenges with the solution

- He does not have experience with Kafka.
- He does not want to deal with the infrastructure.
- He does not have enough time to learn Kubernetes.
- He wants to implement everything in pure Python.
- He wants multiple data engineers can work on one data pipeline space.
- Predict service should also notify the web/mobile app in real-time.

Google and ChatGPT can not help me

I will ask my real friends..

Kafka users were telling him stories

“It took us 9 months to implement Kafka."

“Our data engineers are dependent on our backend devs to make changes in Java.”

“I hated hiring people just to manage Kafka.”

Self-managing Kafka presents several challenges

- What team is responsible for Kafka?
- What is the correct configuration?
- How do you deploy changes to ec2/k8s/machine?
- How should you upgrade the brokers?
- How do you monitor?
- How can you train developers to manage Kafka and its configuration?
- Should you implement 1 cluster for a company or a few clusters?

Managed Kafka Providers

Amazon MSK
Managed Streaming for Kafka

CONFLUENT

Azure HDInsight

Redpanda

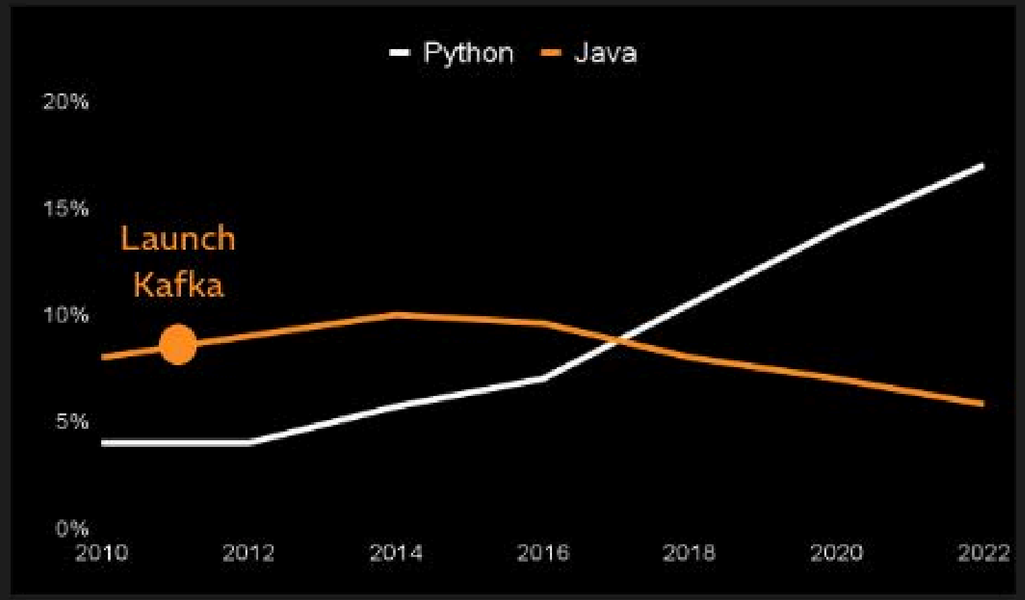
WarpStream

upstash

aiven

Data teams want self-sufficiency in Python

Python popularity overtook Java popularity on Stack Overflow



Kafka leads are becoming bottlenecks in organisations

aiven

Heikki Nousiainen
CTO & Founder @ aiven
Investor @ GlassFlow

"I see customers having upwards 50 teams relying on a single Kafka cluster managed by one person".

TOP 10 COMMON DATA ENGINEERS AND SCIENTISTS PAIN POINTS IN 2024

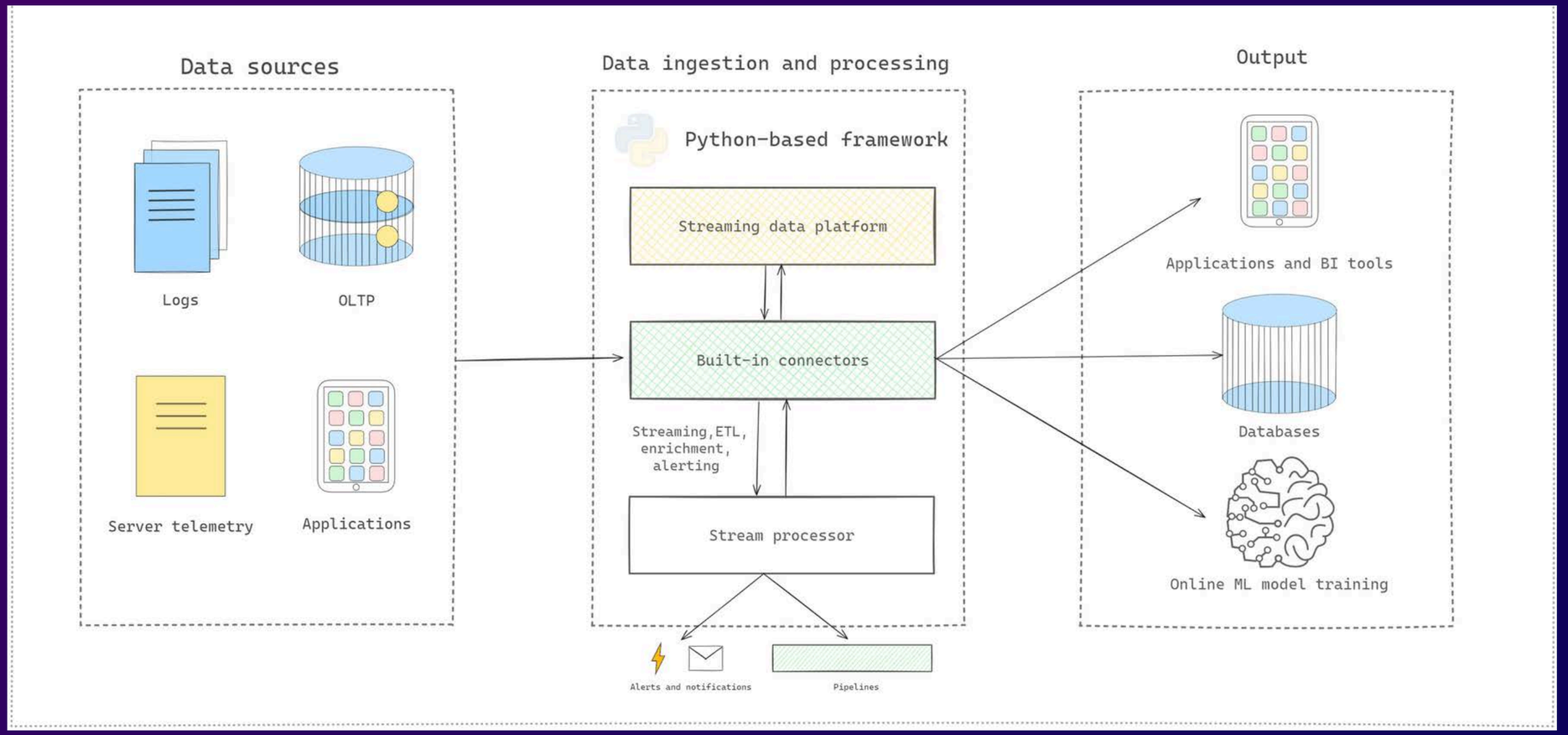
Scan Me

Kafka Alternatives

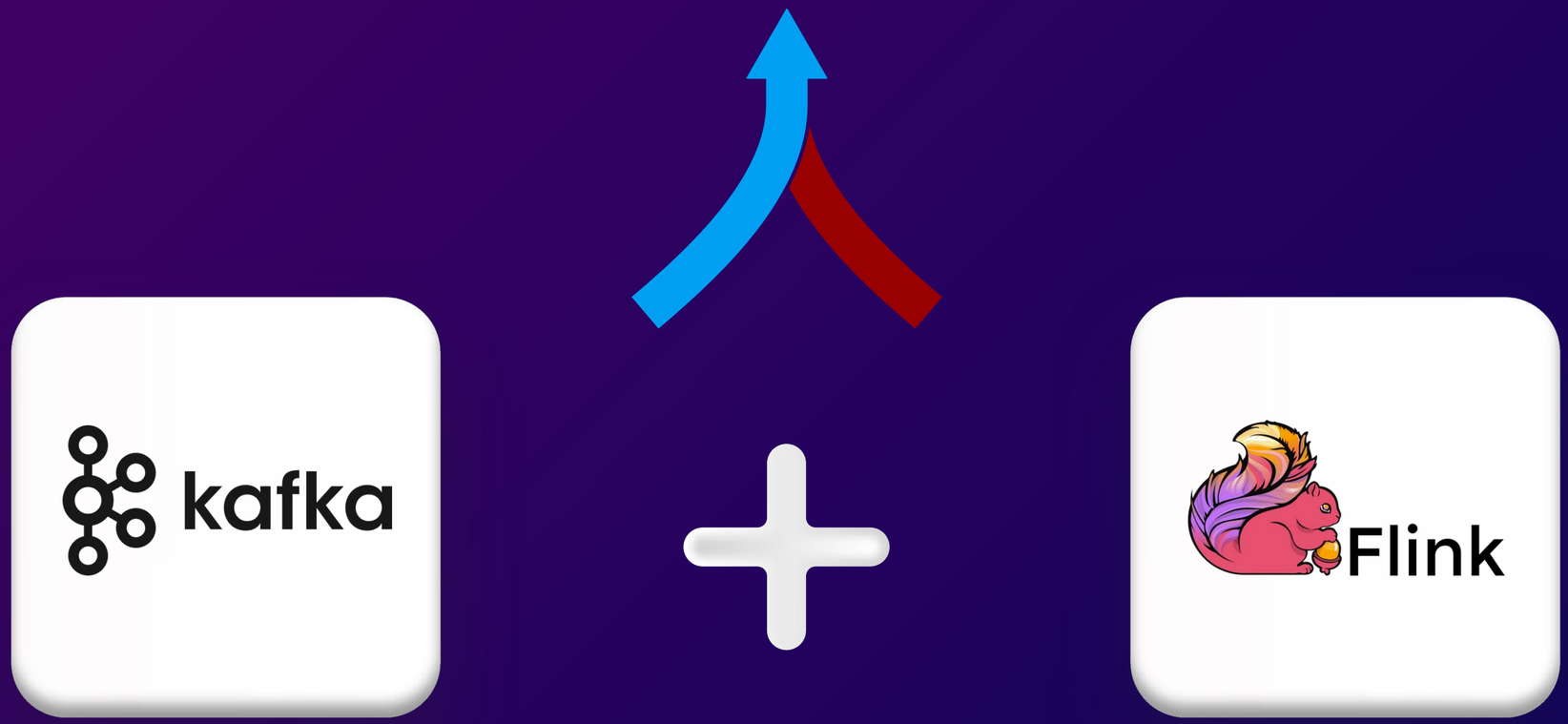
Cloud Pub/Sub

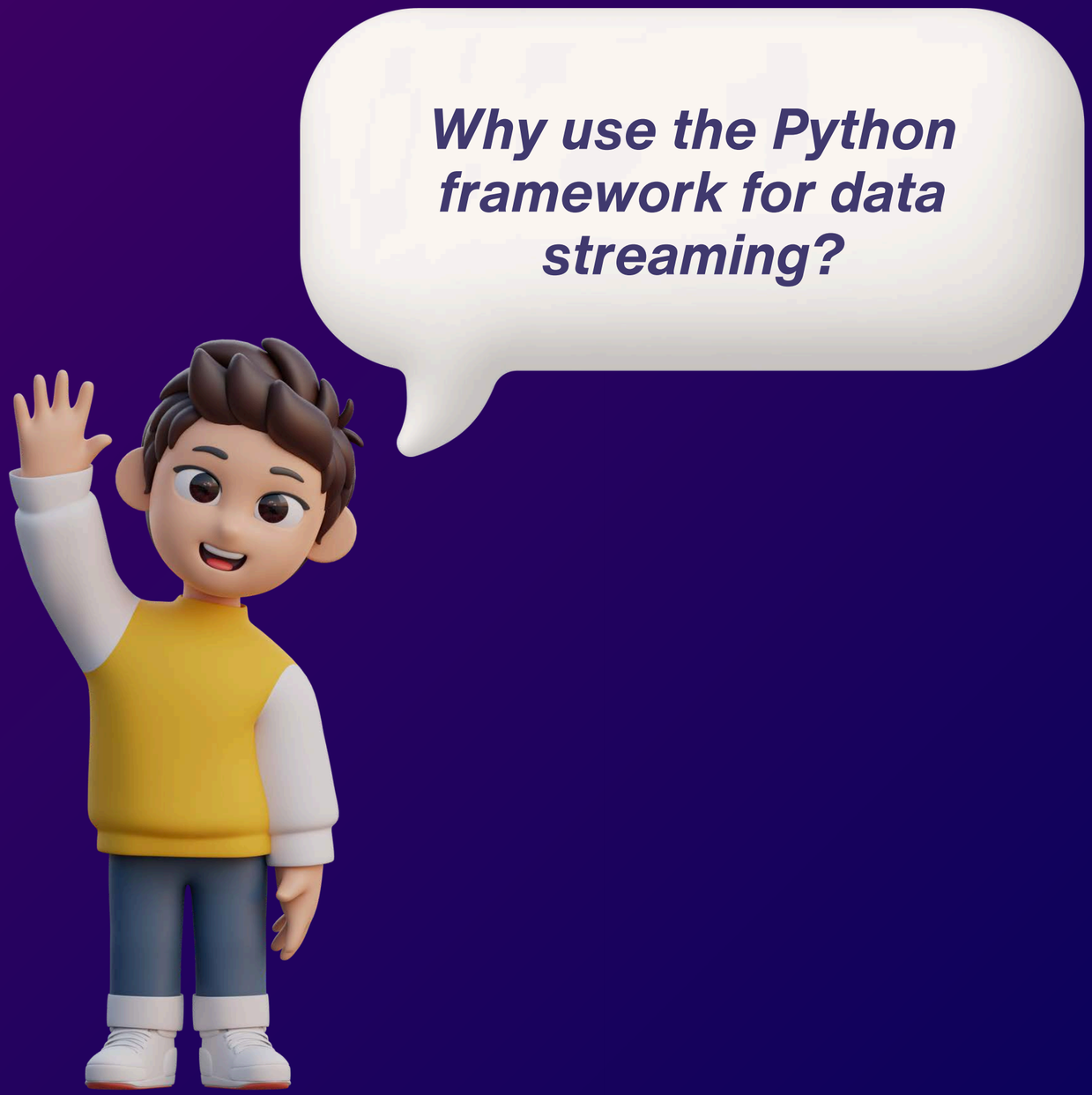
Don't give up. Please, check stream processing frameworks in Python..

Stream processing frameworks in Python



Stream processing frameworks in Python




Why use the Python framework for data streaming?

Why use Python framework for data streaming?

- No JVM, no wrappers, no orchestrator, no server-side engine.
- They can be used out-of-the-box with any existing Python library.
- Unifies the streaming data platform and stream processor components.
- You install them without a complex initial setup.
- Your original data stays where it is.
- They do real-time incremental in-memory transformation.
- You can run your local code right from Jupyter Notebook.
- They offer serverless platforms.

What is GlassFlow?

Messaging Brokers

Google Pub/Sub Amazon SQS

Operational Databases

PostgreSQL MongoDB

Event-Driven Apps

Mobile apps Backend apps

Data Streaming Services

Azure Event Hub Amazon Kinesis

Source Connectors

CLI Web App

A fully managed serverless streaming data pipeline in minutes

Zero-infrastructure transformations

SDK

Code first development in Python

Sink Connectors

Data Warehouses

Snowflake Google BigQuery

Cloud Storages

Amazon S3 Azure Blob Storage

Real-time Apps

Mobile apps Backend apps

Search & Analytical Databases

ClickHouse Chroma DB

Zero infrastructure data transformations

Real-time

Built-in message broker

Casual Python style

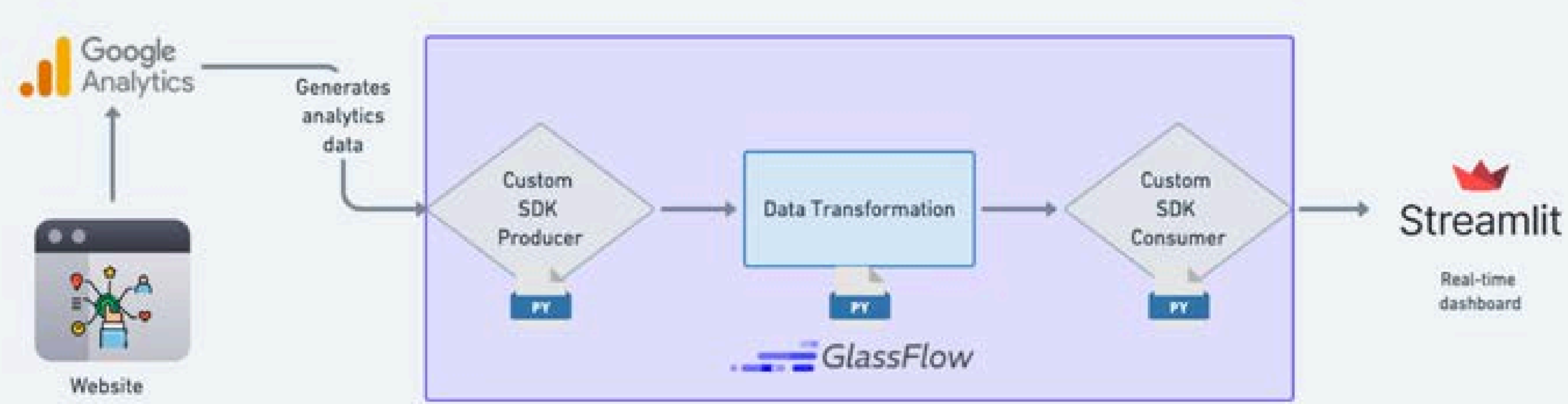
Build data streaming pipelines within minutes

1. Connect live data sources using the **GlassFlow Python SDK** or built-in **integrations**
2. Create a pipeline using **GlassFlow WebApp or CLI**
3. Implement a transformation function in **Python**
4. Transform your data in real-time in the **serverless execution engiine**
5. Consume processed data using the **GlassFlow Python SDK** or built-in **integrations**

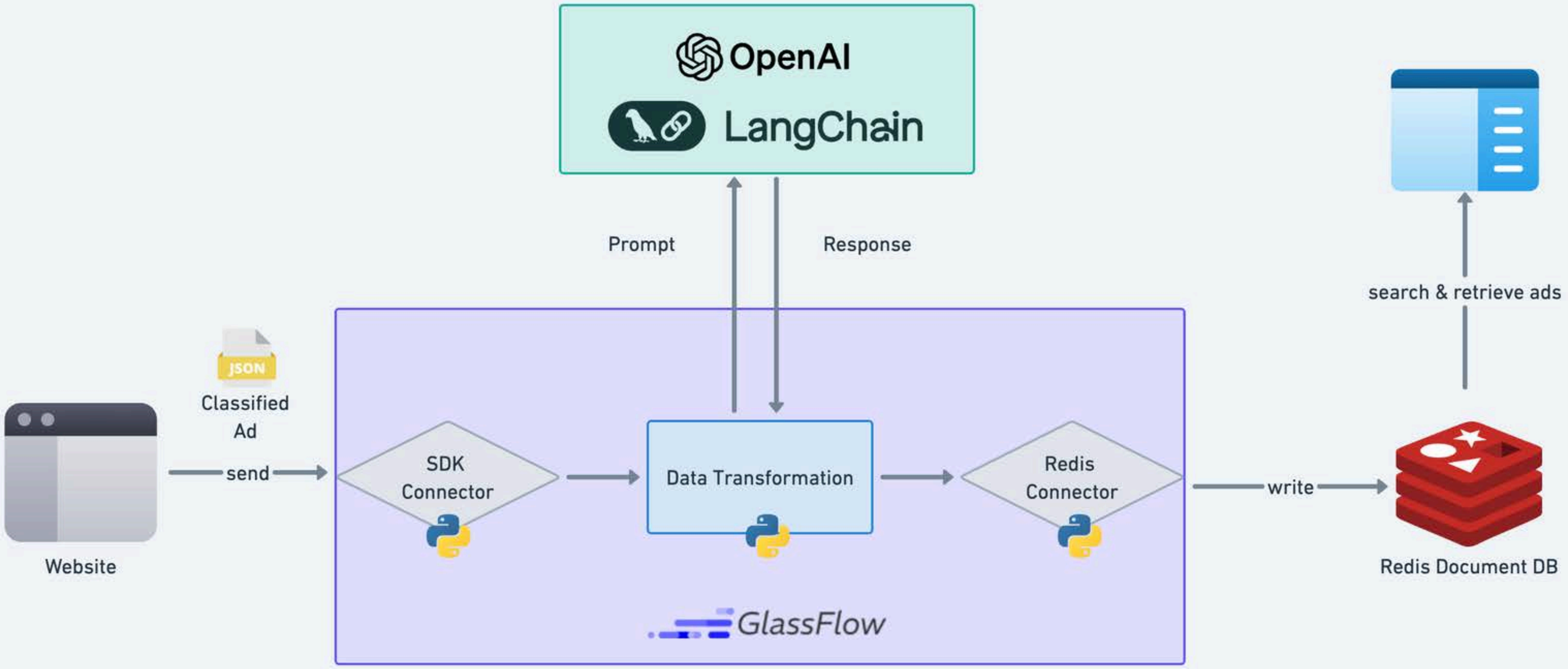
Built using robust technologies


Bring me examples for real-world scenarios.

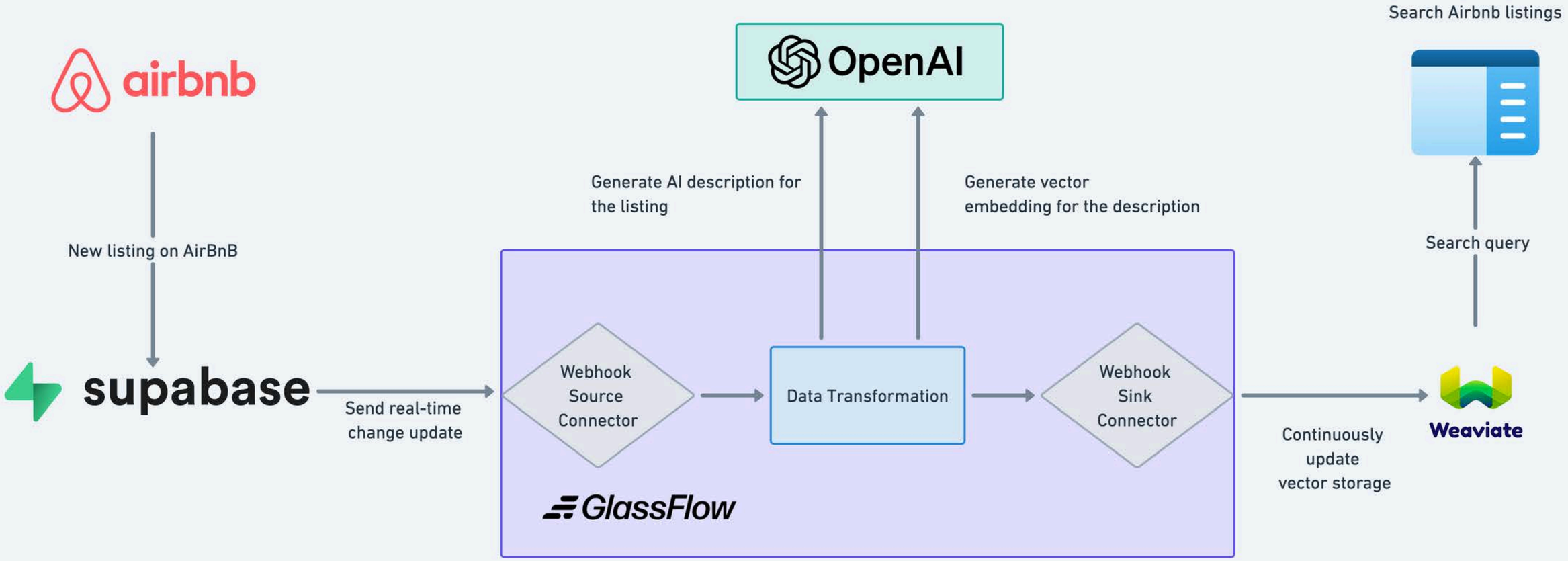
Real-time clickstream analytics



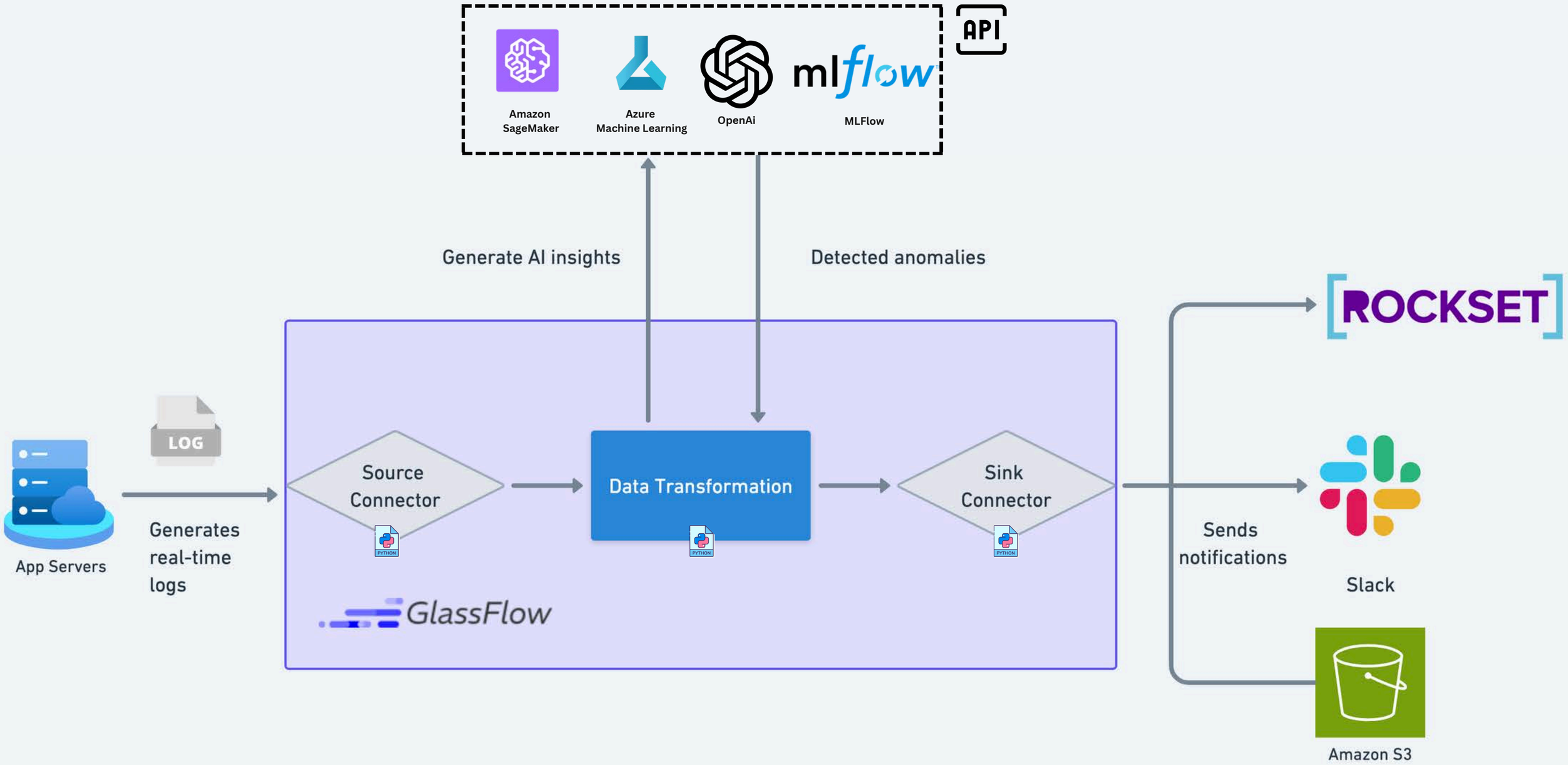
Real-time classified ads enrichment



Continuously update the vector database



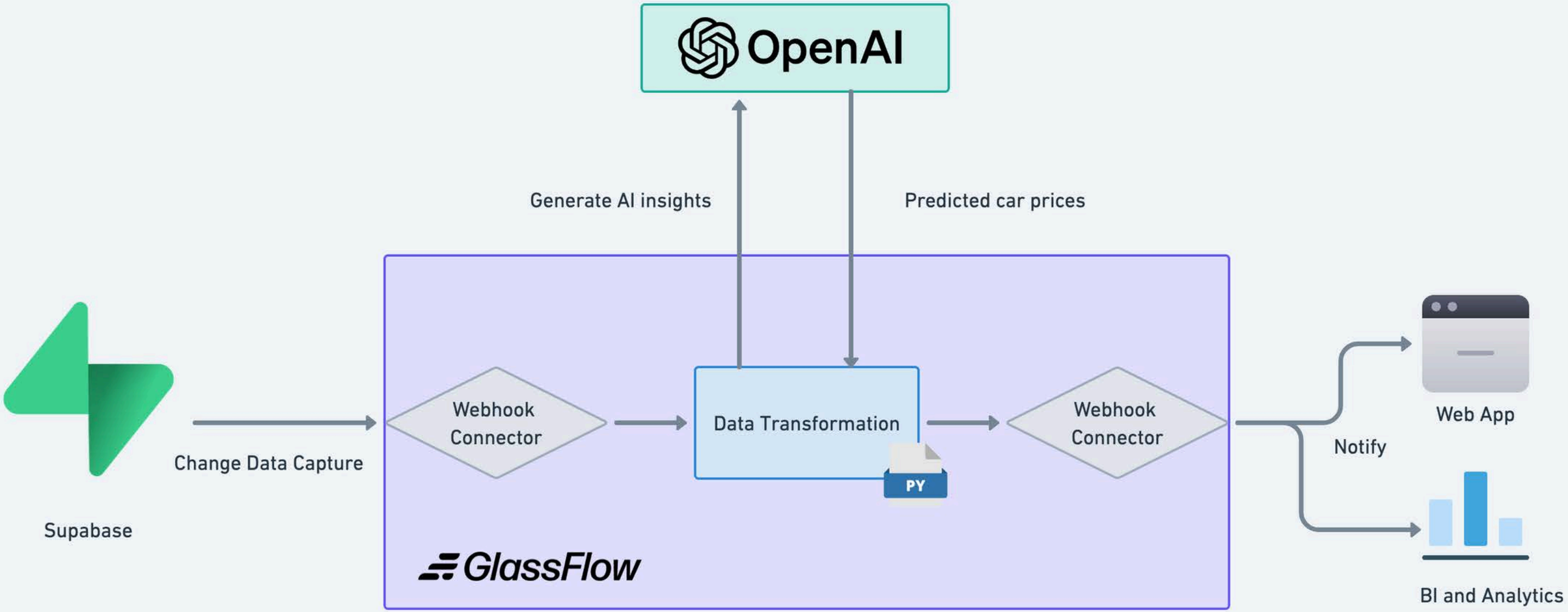
Real-time anomaly detection



Explore use cases

Scan Me

Solution 2: Real-time price recommendation



Tools we use

- GlassFlow WebApp - to create a pipeline in a low-code interface.
- OpenAI - to predict car prices.
- Supabase - to store registered cars for selling/buying.

Project code on GitHub

Scan Me

That's it, you enabled your data pipeline from day 1.

O, yeah! Thanks for help!

What do people typically use GlassFlow for?

When they want to:

- Improve latency
- Reduce cloud computing costs
- Integrate with real-time data sources
- Improve data team collaboration
- Enable event-driven architecture
- Transform billions of records efficiently

Summary

Bob's challenges

Data engineers face problems with JVM-based real-time processing tools nowadays

Bob's wants

Data engineers want self-sufficiency in Python

What Bob needed

Stream processing frameworks in Python

Bob's used

Serverless stream processing pipeline

Connect on LinkedIn

Thank You

Bobur Umurzokov

@BoburUmurzokov

@Boburmirzo