ML @ Timeplus

How to grow under AGI waves with Timeplus

Timeplus

July 2025

Three main use cases

Driver of -
Data engine for
real-time ML feature-platform

Builder of - ML App
Providing real-time context and long term storage to LLM

Safeguard of ML infrastructure
Monitor the usage and security of LLM

Enable Real-time ML Pipeline and Access

ML Feature Platform

What is real-time ML?

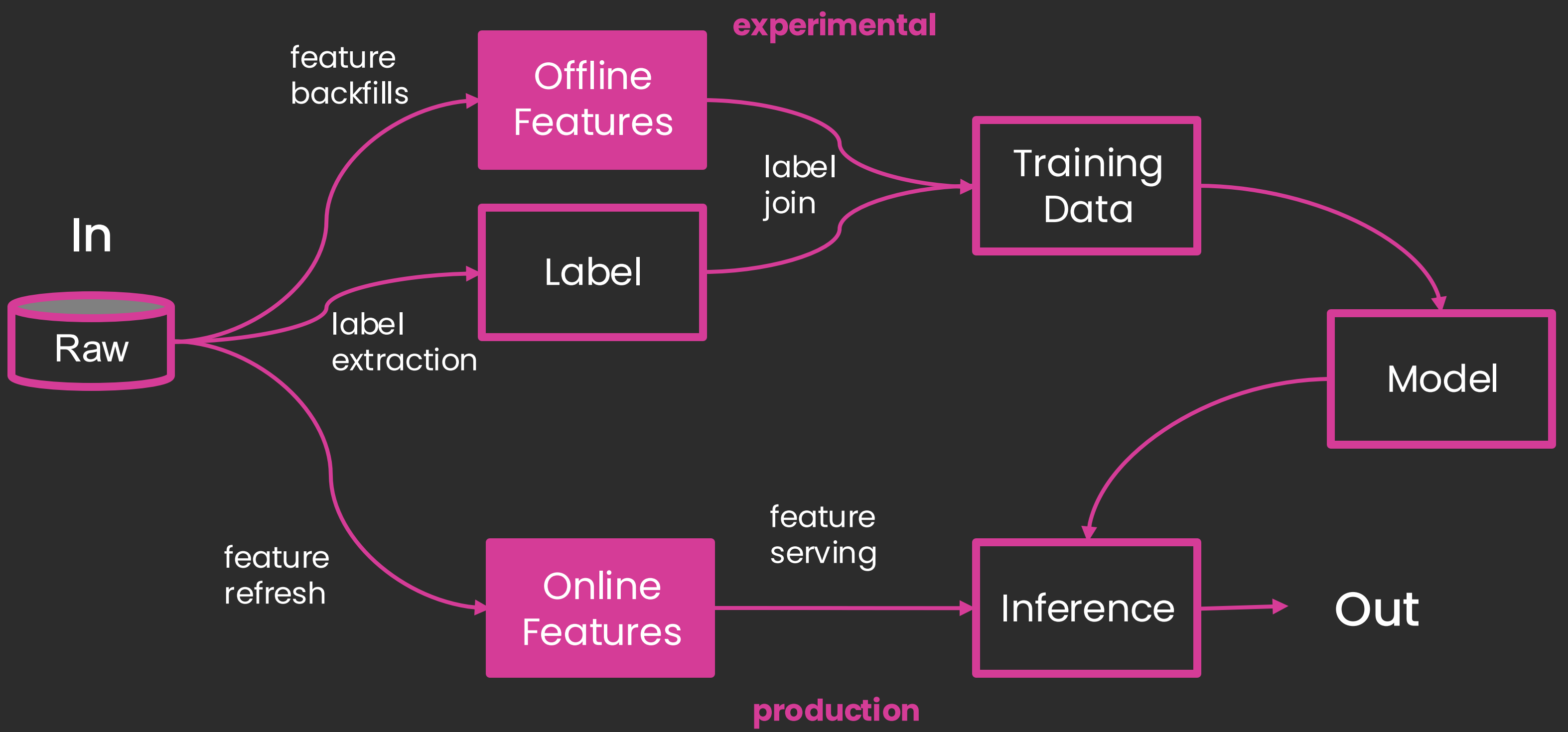
Real-time means

- Real-time prediction - low latency inference
- Online Learning - continuously model updating

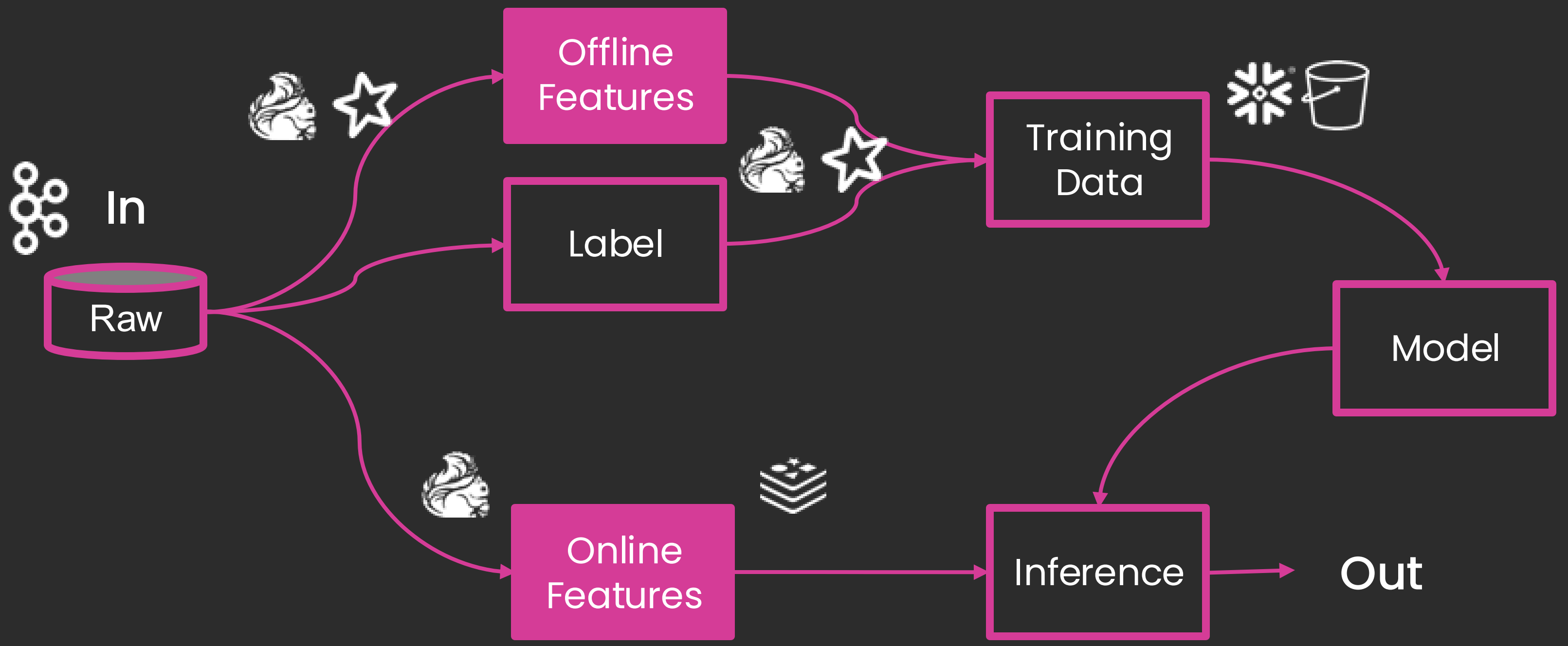
Use Cases:

- Fraud detection
- Recommendations
- Real-time pricing

Real-time ML Basic Flow



ML Basic Flow - Tools



What are features?

"**features**" refer to the individual, measurable properties or characteristics of data that are used as input for a machine learning model.

- **Batch**
  - Daily transaction volumn
  - Average Rate for 90 days
  - 5 Hours view count
- **Near Real-time**
  - Avg transaction value over last 30 mins
- **Real-time**
  - Block trade - If transaction value > $1000

Histroical

Real-time

How to create features?

- Transformation
  - Scale (log, min/max), Binning, Ranking
- Aggregation
  - Min/Max, Average, Mean, STD, Sum, Count
- Time series
  - Moving Average, Lags and Shifts
- Image/Text
  - Embeddings, frequency
- Encoding
  - one-hot, label

Most of these methods can be easily applied to **SQL**

Technology choice of feature platforms

	Feature store	Feature API (transformation - feature)	Stream comput. engine
LinkedIn	Venice, Fedex	Python - Python	Samza, Flink
Airbnb	HBase-based	Python - Python	Spark Streaming
Instacart	Scylla, Redis	? - YAML	Flink
DoorDash	Redis, CockroachDB	SQL - YAML	Flink
Snap	KeyDB (multithreaded fork of Redis)	SQL - YAML	Spark Streaming
Stripe	In-house, Redis	Scala - ?	Spark Streaming
Meta (FB)		Scala-like - ?	XStream, Velox
Spotify	Bigtable	Flink SQL - ?	Flink
Uber	Cassandra, DynamoDB	DSL - ?	Flink
Lyft	Redis, DynamoDB	SQL - YAML	Flink
Pinterest	In-house, memcached	R	Flink
Criteo	Couchbase	SQL - JSON	Flink
Binance		Flink SQL - Python	Flink
Twitter	Manhattan, CockroachDB	Scala	Heron
Gojek	DynamoDB	SQL - JSON	Flink
Etsy	Bigtable	Scala - ?	Dataflow

Most of the feature platforms today leverage **KV** as feature storage and **streaming processor** as feature computation engines

Scala, **SQL**, **Python** are the most popular transformation tools used for features

https://huyenchip.com/2023/01/08/self-serve-feature-platforms.html

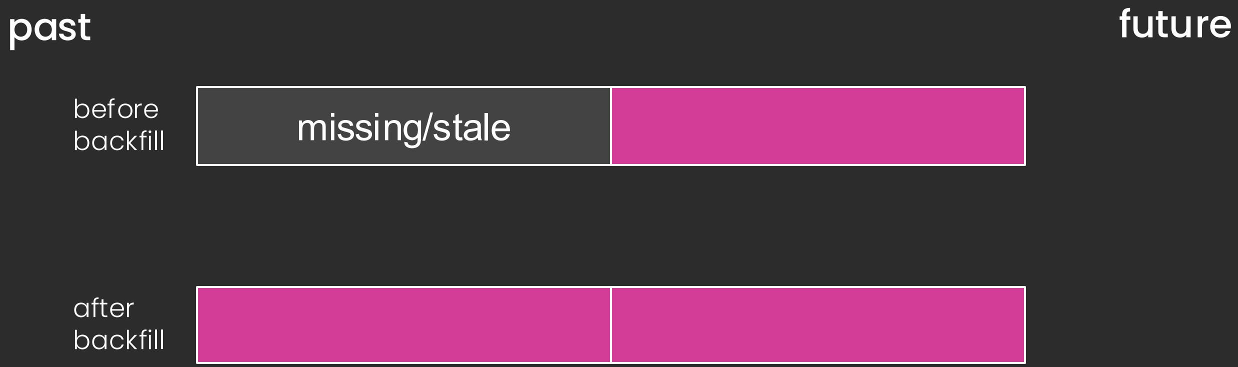
Real-time ML challeges

- Feature freshness - real-time/low latency feature
- Feature consistency - training/inference
  - Backfill historical data - regenerating features / replay
  - Point-in-time correctness - time travel
- Manage complexity of streaming system - hard to manage streaming system like Flink , hard to learn and more like to use Python

Feature Backfill

Backfilling is the process of recomputing datasets from raw, historical data.

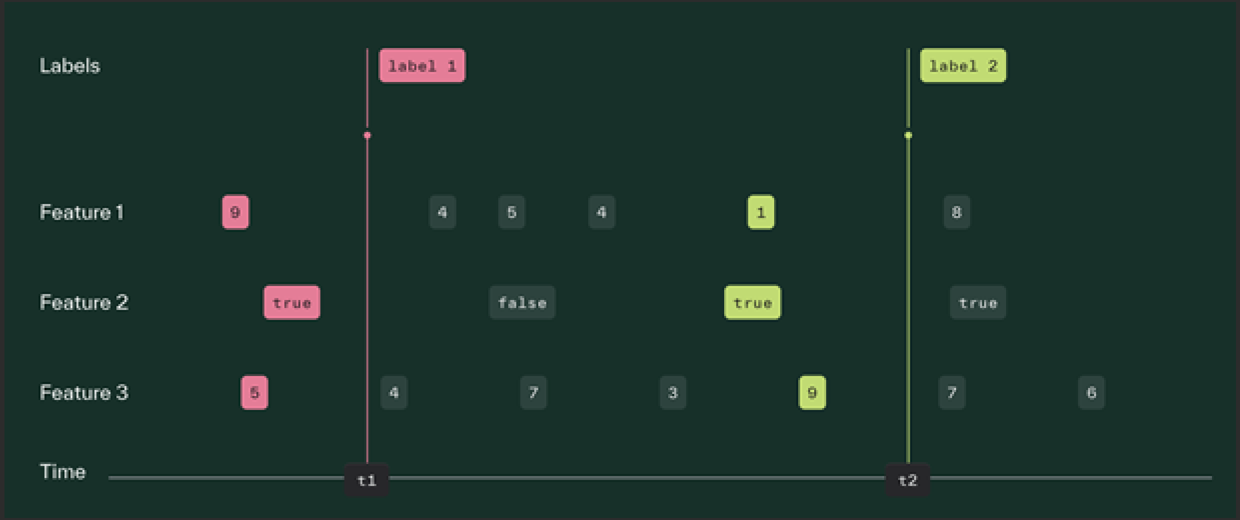
- data was incomplete or missing
- experimental with new features



Point-in-time Correctness

"Point-in-time correctness" refers to the principle that the features used for machine learning models should accurately represent the **state of the world at the point in time** for which a prediction or analysis is being made.

Prevent data leakage

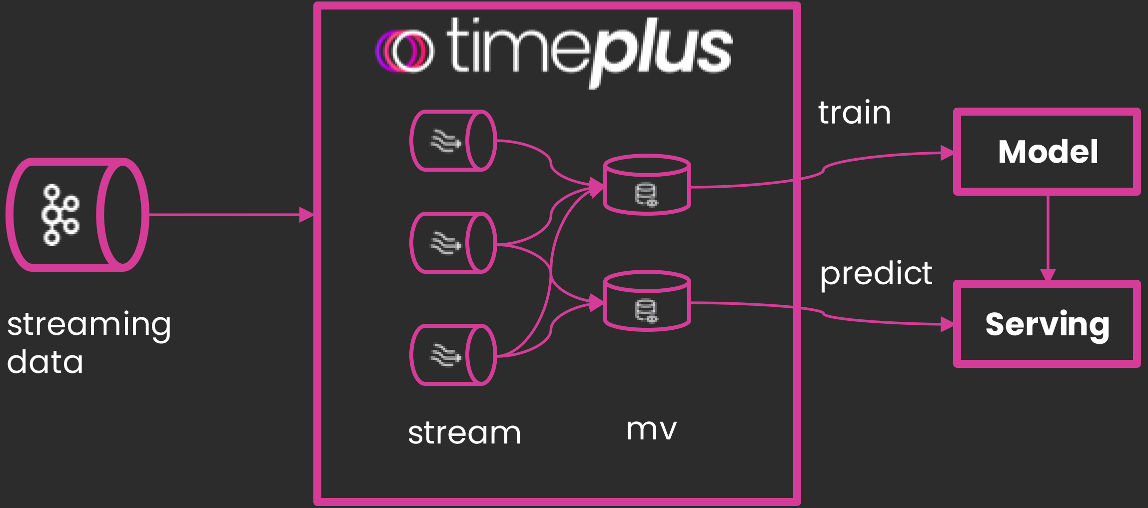


https://docs.chalk.ai/docs/temporal-consistency

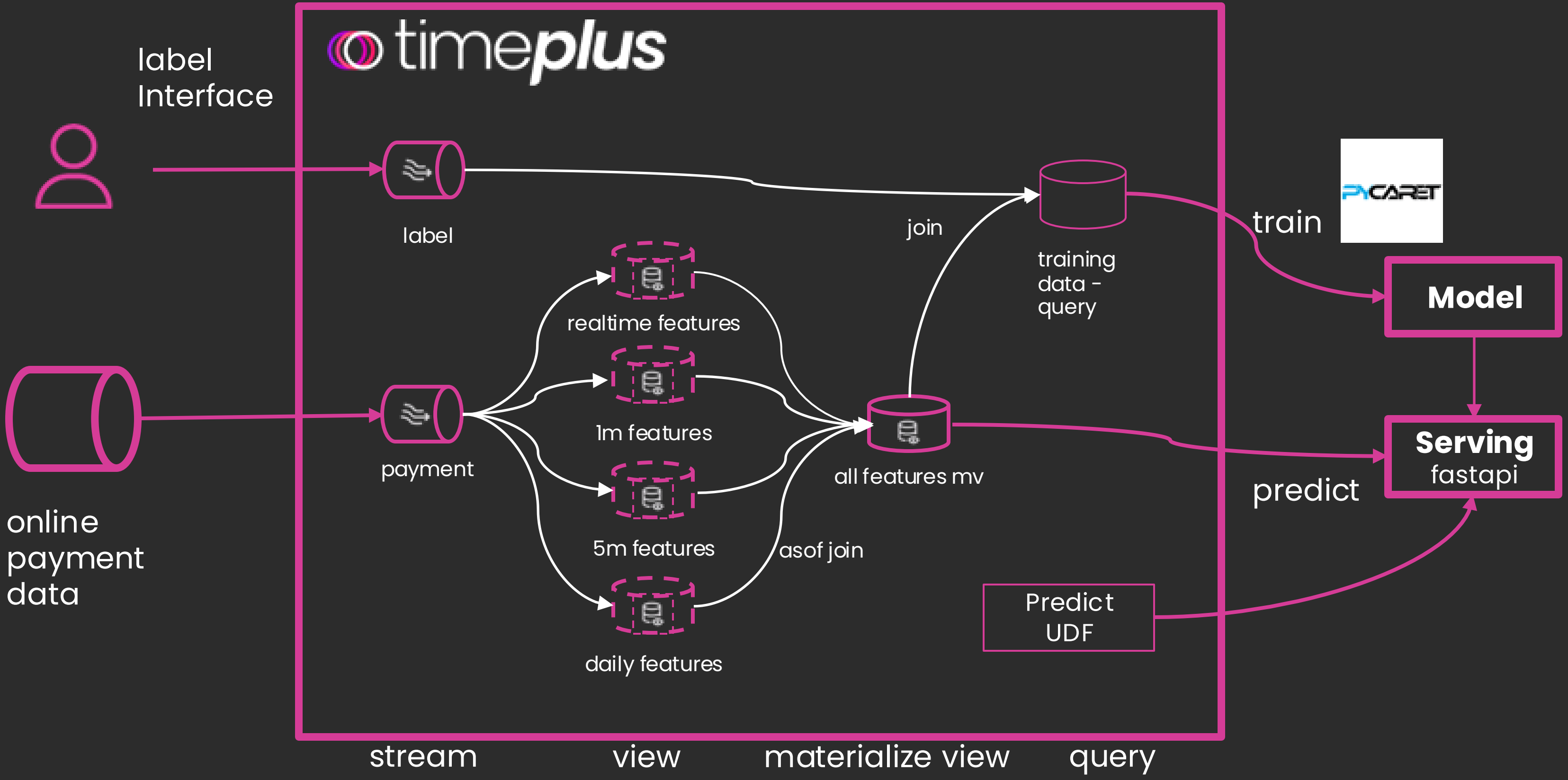
Why T+ is a good choice to build feature platform

- Feature freshness
  - Low latency streaming processing
  - Support those time range related, statful features processing with TVF
- Feature consistency
  - Unified streaming/historical processing
  - Time travel/ASOF Join for PITC
- Complexity
  - Solving the online and offline requirements for a unifying data layer, simplified overall ML application deployment
  - Provide most of functionalities for the feature generation in SQL
- Other
  - Enrichment of real-time features with historical data
  - Providing view or mv based version control

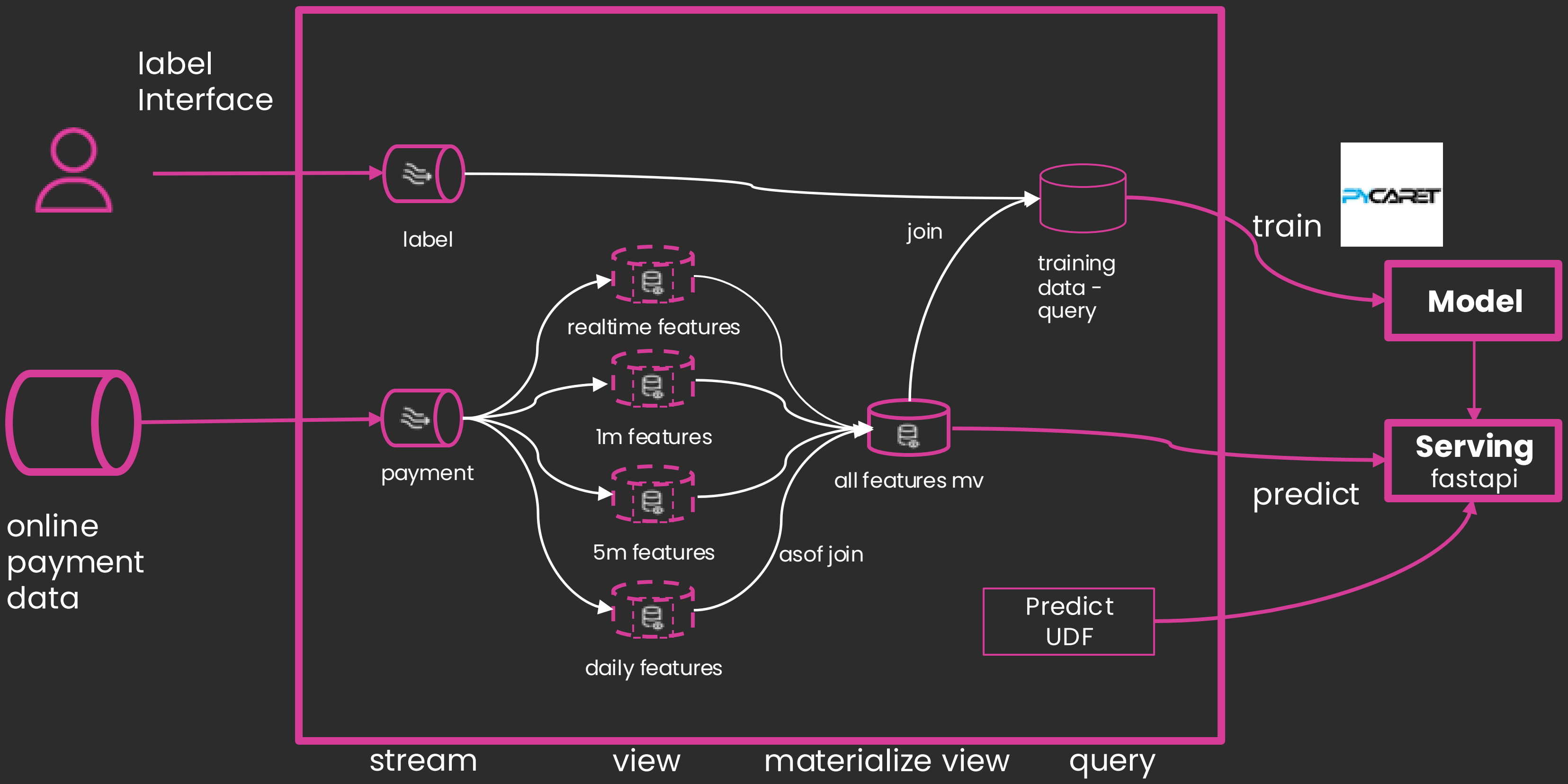
Timeplus as Real-time ML Feature Platform



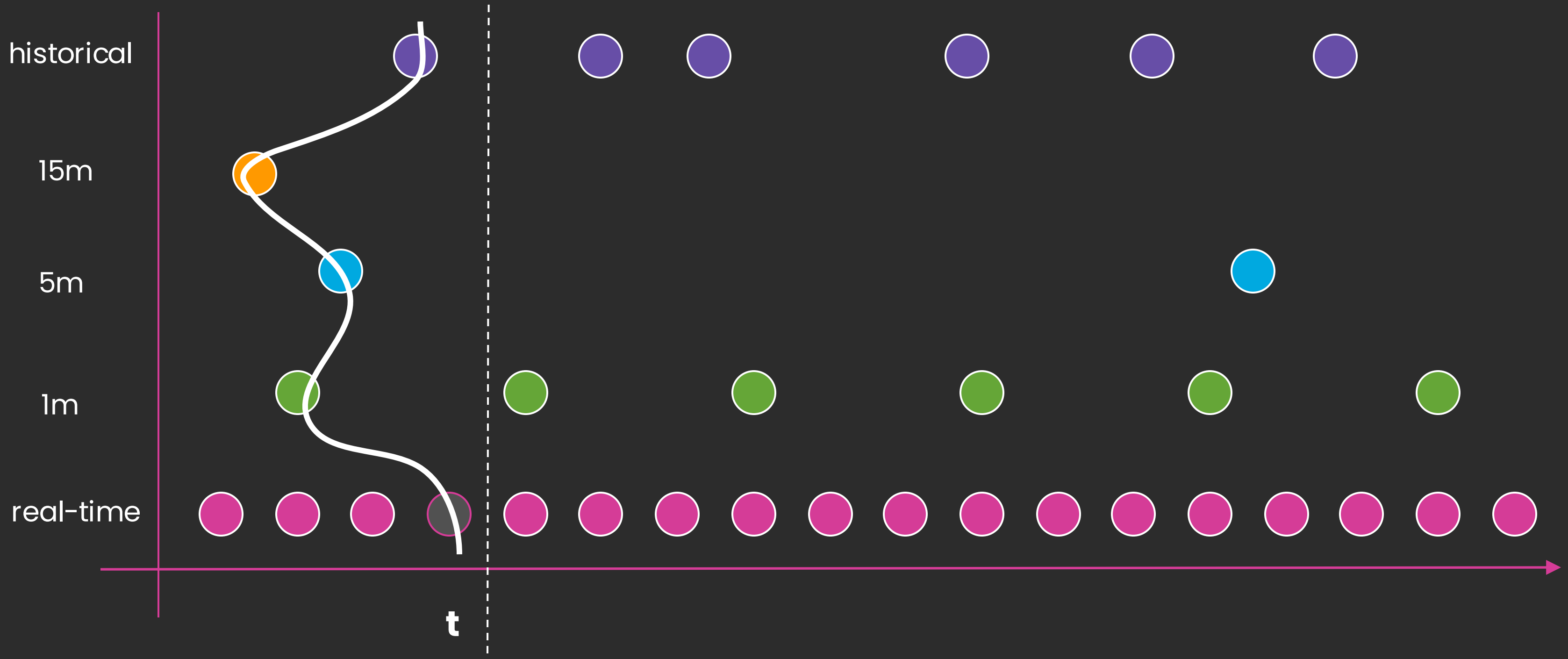
Fraud Detection Show Case - Feature Pipeline



Fraud Detection Show Case - Feature Pipeline



Point-in-time Correctness - ASOF Join



Fraud Detection Show Case - Model Performance Monitor

Model Performance Mointor 5 minutes

Last updated: Just Now

window_start	accuracy	precision	recall
2023-12-02T01:30:00Z	1	0.81	0.96
2023-12-02T01:35:00Z	0.99	0.76	0.89
2023-12-02T01:40:00Z	1	0.7	1
2023-12-02T01:45:00Z	1	0.71	1
2023-12-02T01:50:00Z	0.99	0.75	0.97

Related Startups/Project

- Feast
  Open source feature store
- Tecton
  Build, automate, and centralize production-ready batch, **streaming**, and **real-time** data pipelines to power any ML application with fresh ML features on demand
- Hopworks
  The collaborative ML platform for batch and **real-time** data
- Fennel AI
  Real-time feature platform. beautifully built
- Claypot
  Unify data for **real-time** AI
- Chalk AI
  Tired of Spark? So are we. **Just-in-time** data + Hot-reload + Rust compute

Missing Parts

1. Time travel for non-time related features
   In case a slow changing feature is managed as a versioned kv, we may need provide query that can find the specific version of time to build the feature at that time.

2. Deep python Integration
   Python are the most popular language for Data scientists/engineers, some deep integration such as Pandas will be a good UX for them

LLM Real-time Context Provider

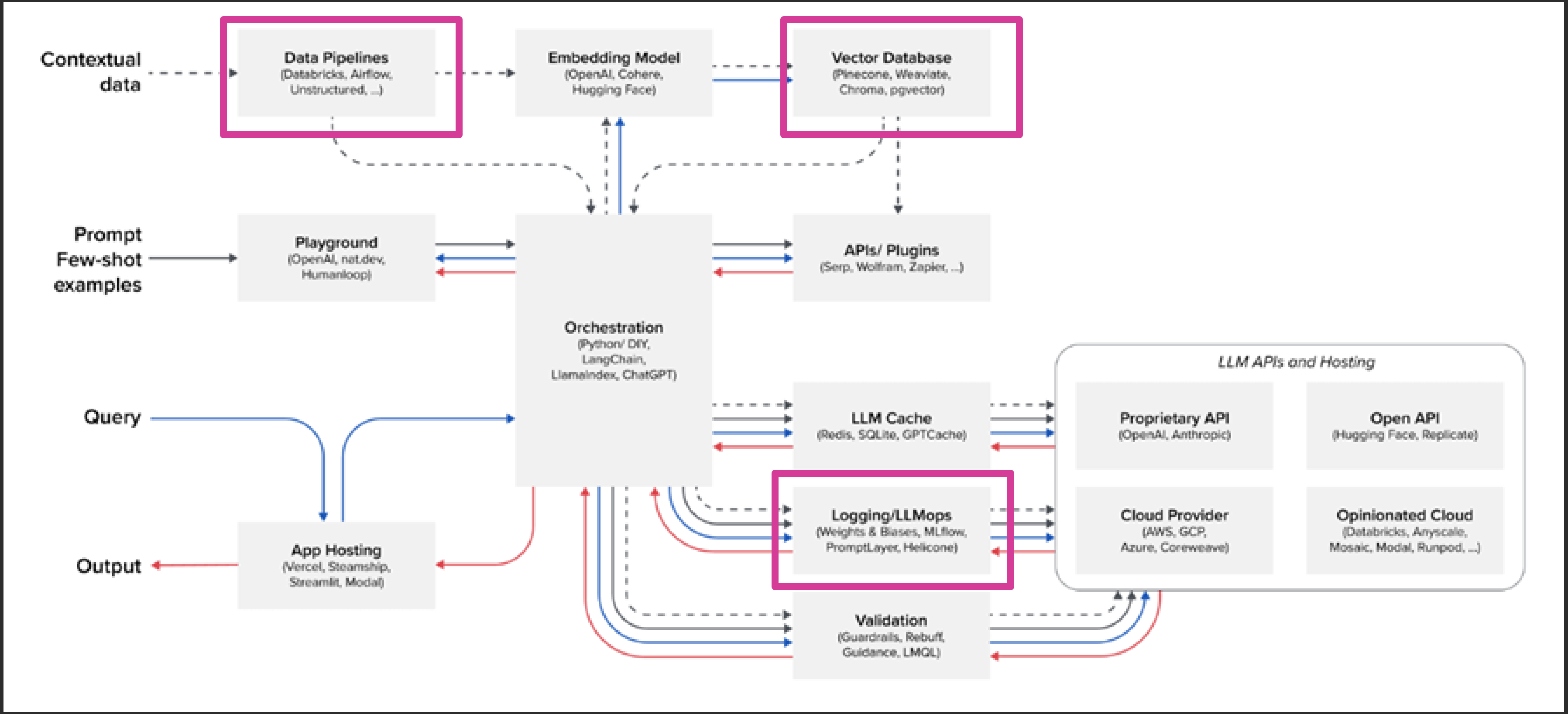
LLM App builder

Large language models are a powerful new primitive for building software. But since they are so new—and behave so differently from normal computing resources—it's not always obvious how to use them.

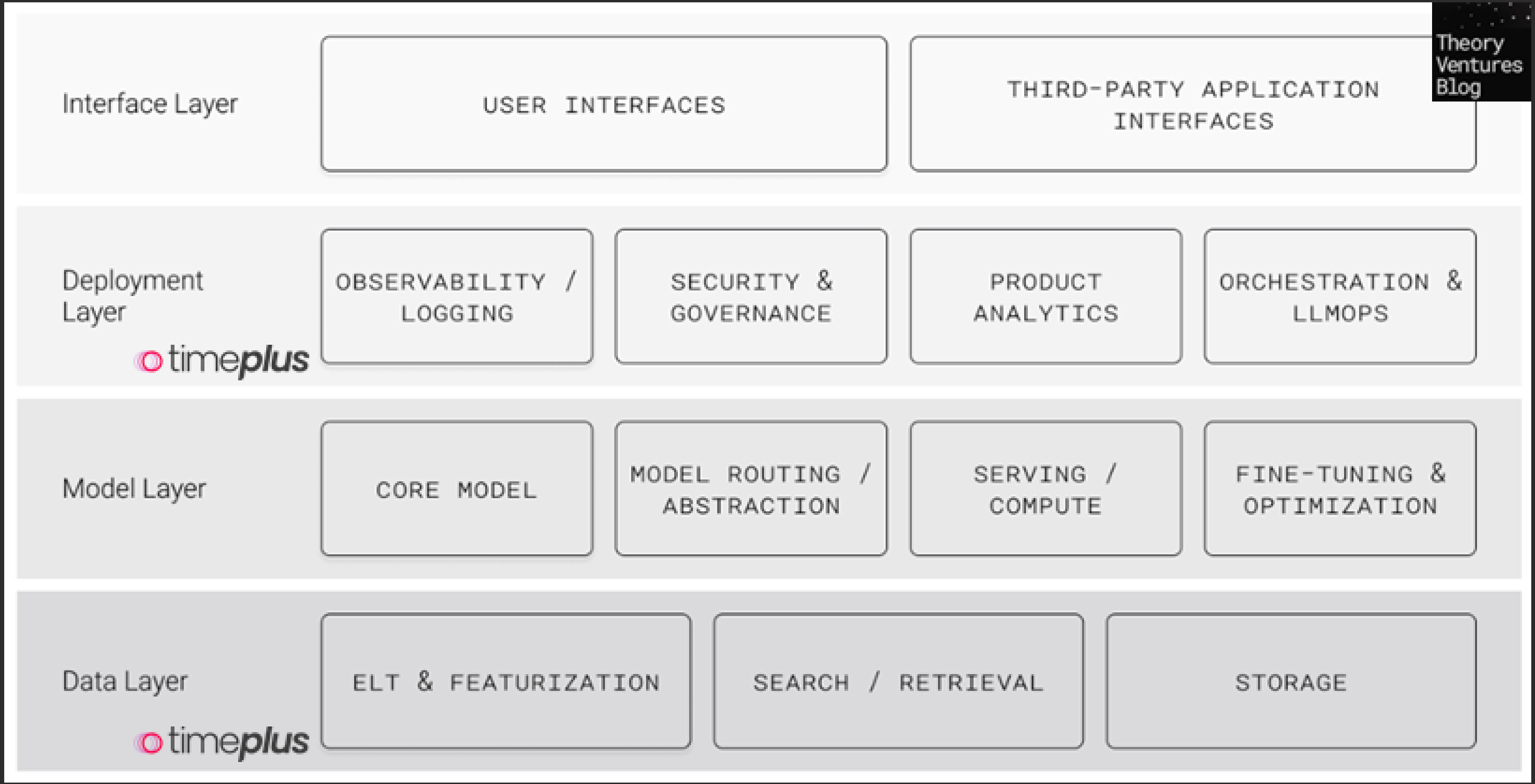
A16Z

CONFIDENTIAL

Emerging LLM App Architecture

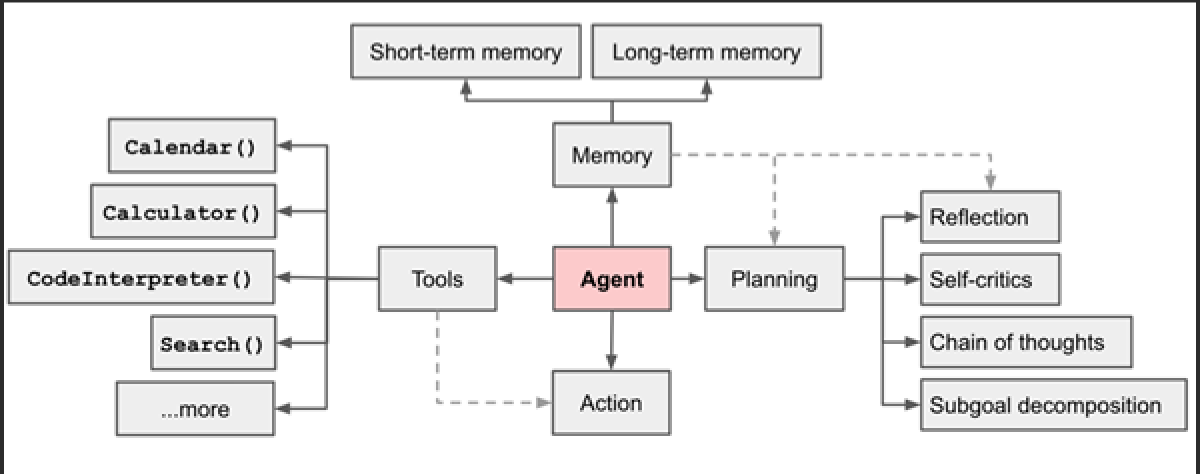


Four Layers of LLM Applications



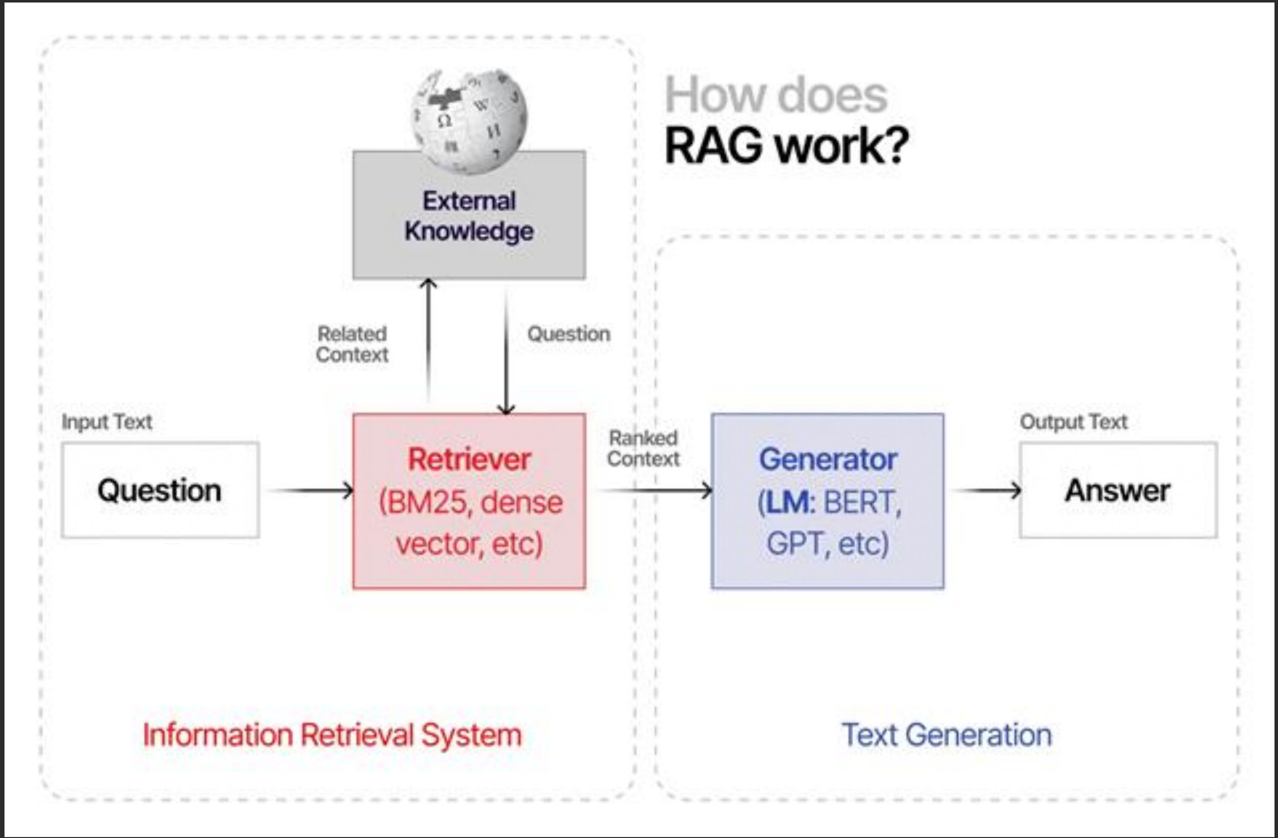
LLM Agent

New paradigms of Application building

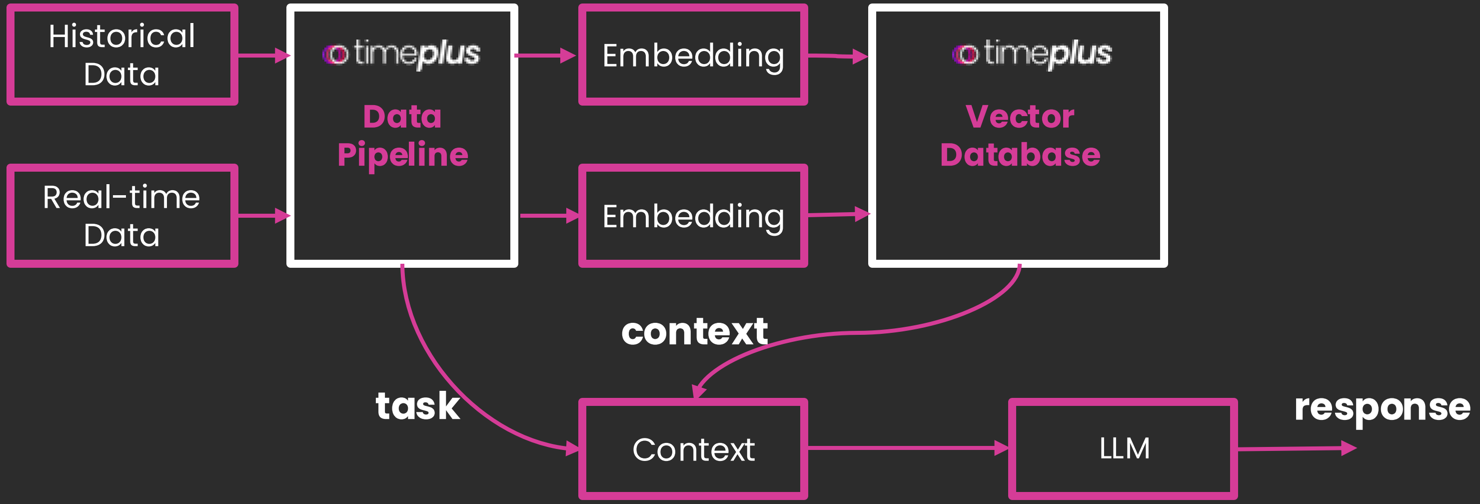


- LLM - compiler/intepreter
- prompt - code
- agent - coding framework
- services - libs/api
- vector db - storage/memory
- COT/refections - design patterns

RAG - better context based on historical/real-time data

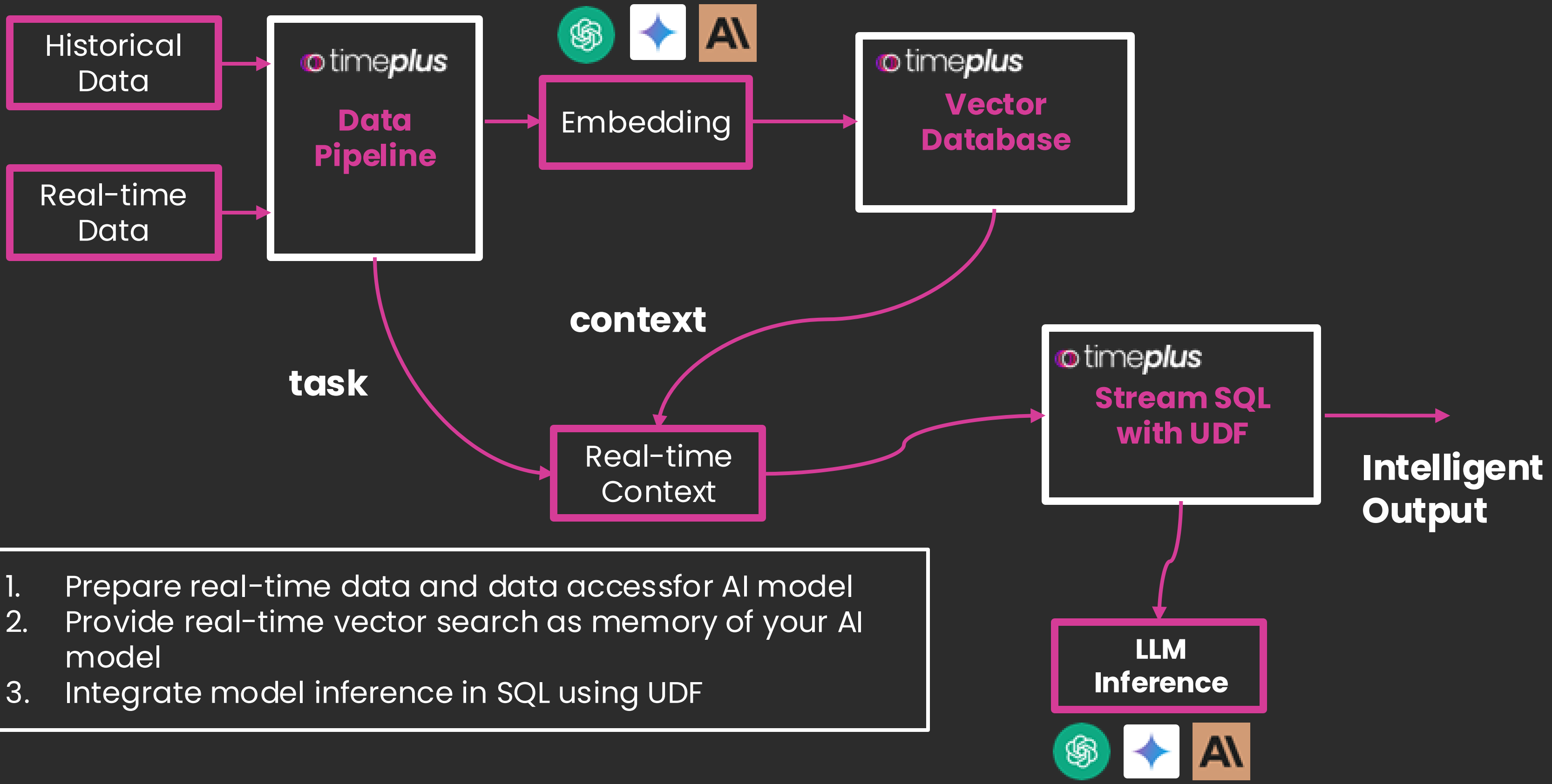


In Context Learning (ICL) App Flows



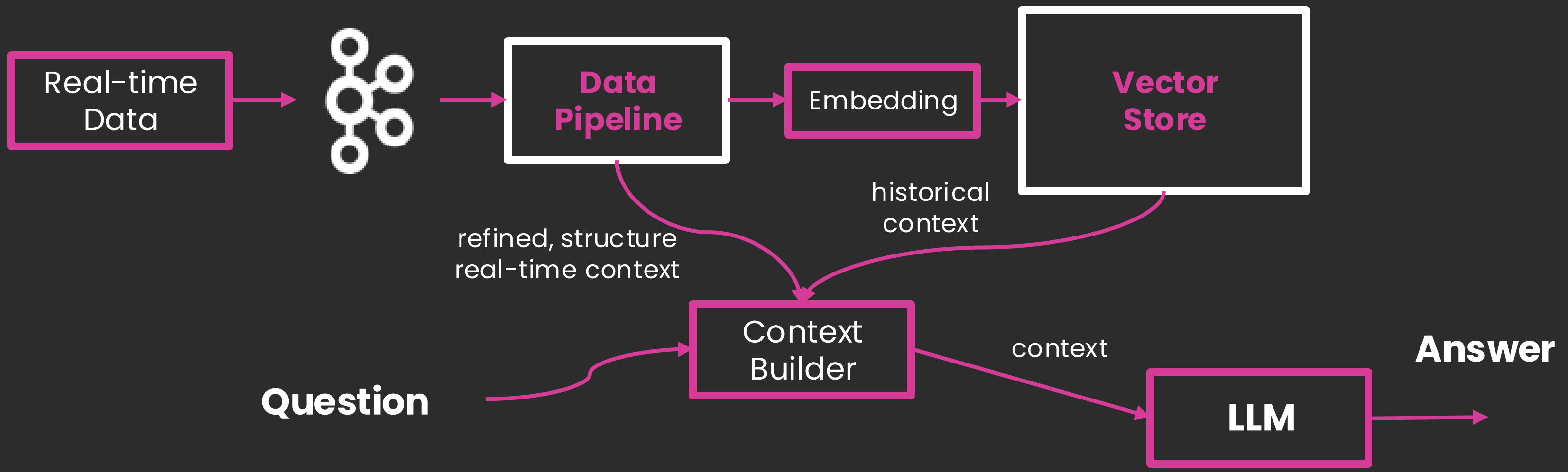
Building LLM Application is more of a data engineering problem today!

Timeplus AI Flow



1. Prepare real-time data and data accessfor AI model
2. Provide real-time vector search as memory of your AI model
3. Integrate model inference in SQL using UDF

Real Time LLM Application



Building LLM Application is more of a data engineering problem today!

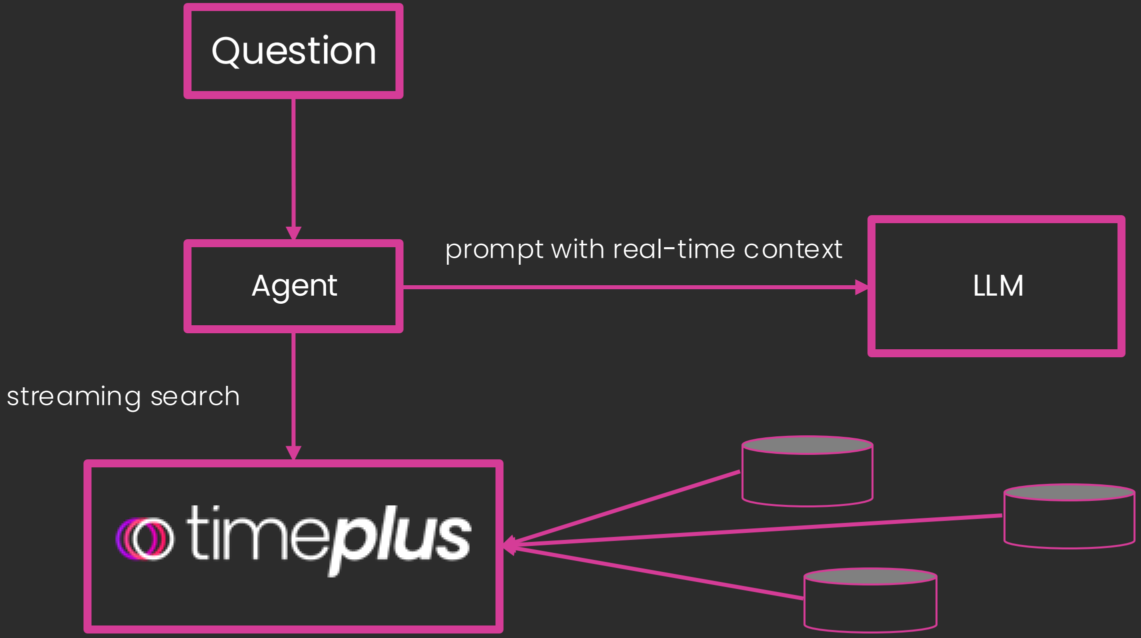
Challenges

1. Data freshness
   LLM were trained on static datasets which are now outdated
2. Data source
   Distributed, heterogeneous data sources
3. Short memory
   Context length is limited (new models are supporting bigger context)
4. High Cost
   long context cost more
5. Latency
   long output need more inference time (sequential inference, 20 token/s)
6. Data Risk
   Quality/Privacy/IP/Bias
7. Hallucination

Challenges - Solutions

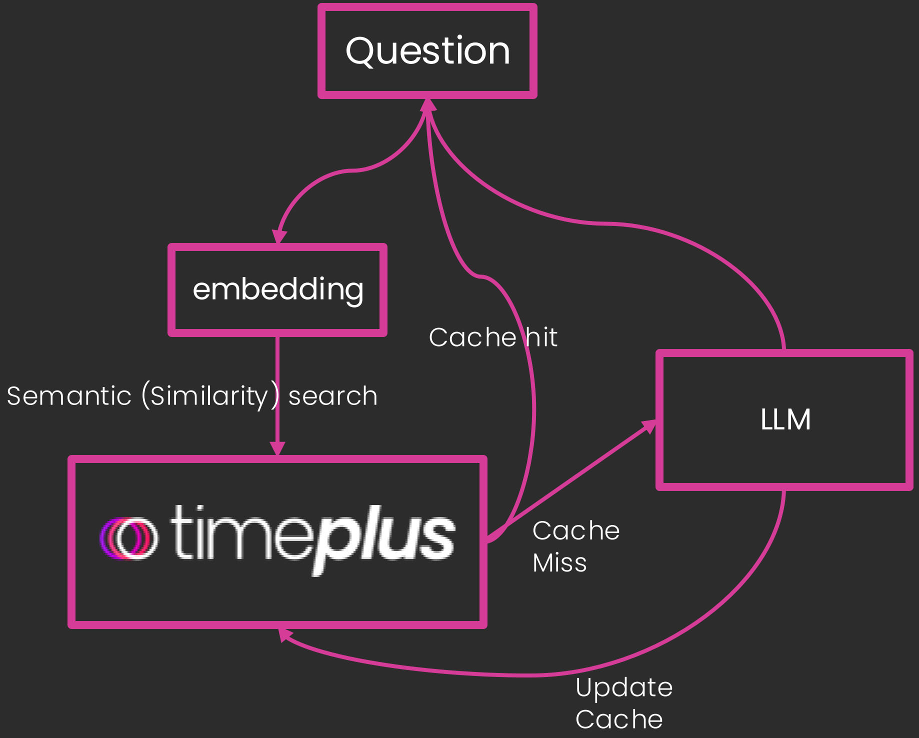
1. Data freshness - **Real-time context**
   LLM were trained on static datasets which are now outdated
2. Data source - **Real-time ETL pipline**
   Distributed, heterogeneous data sources
3. Short memory - **Vector DB as long term memory**
   Context length is limited (new models are supporting bigger context)
4. Cost - **Vector DB as cache**
   long context cost more
5. Latency - **Vector DB as cache**
   long output need more inference time (sequential inference, 20 token/s)

Real-time context data provider for LLM



streaming ETL pipline to provide real-time context

Long term storage/cache for LLM

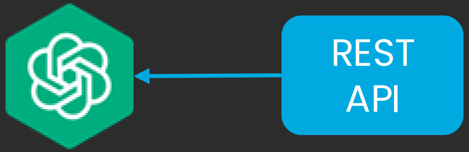


Lower the cost and latency by provide cached result leveraging vector search

LLM usage and security monitoring
LLMOps

timeplus

LLM application is complex



What you think

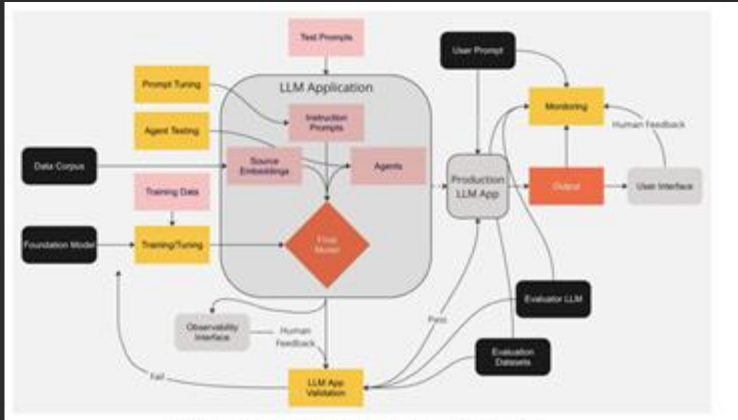


Photo created by Josh Poduska — Link to Interactive View

What actually is

LLM Challenges

1. Data freshness
   LLM were trained on static datasets which are now outdated
2. Data source
   Distributed, heterogeneous data sources
3. Short memory
   Context length is limited (new models are supporting bigger context)
4. High Cost
   long context cost more
5. Latency
   long output need more inference time (sequential inference, 20 token/s)
6. Data Risk
   Quality/Privacy/IP/Bias
7. Hallucination

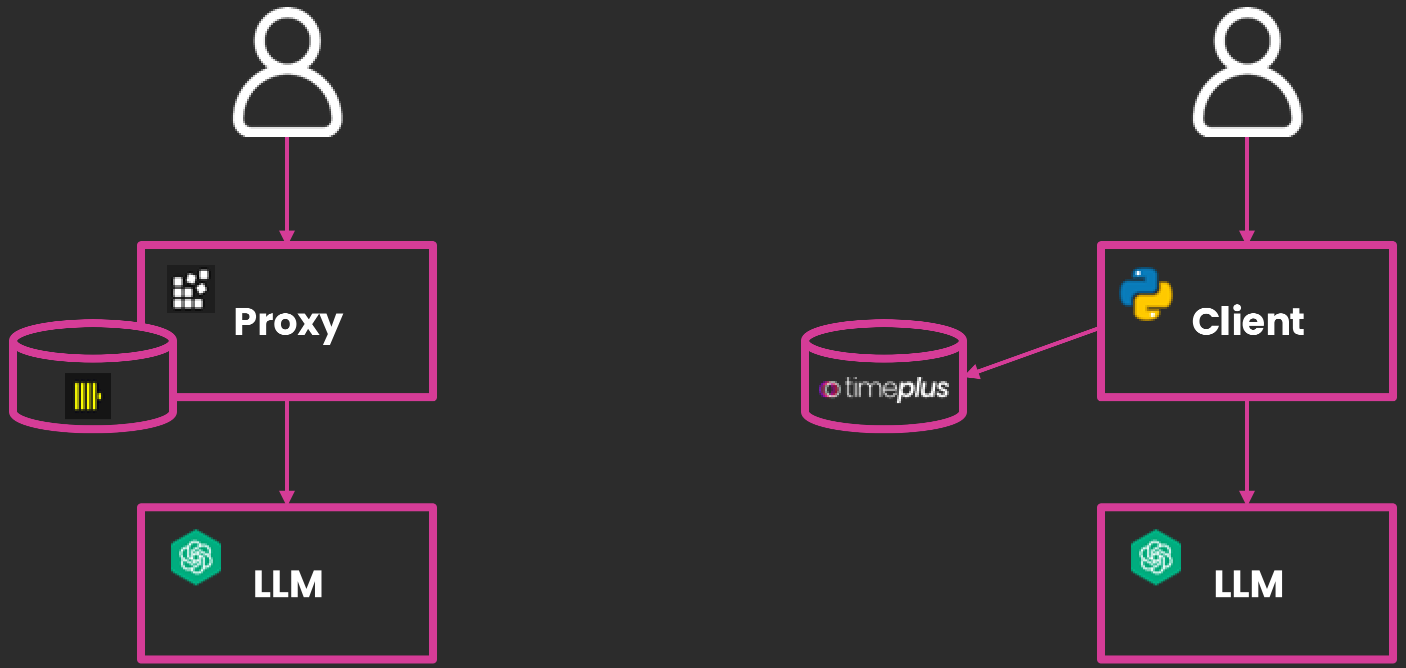
Why Observability is important to LLM applications

- **Transparency**
  LLMs are complex black box systems. Observability opens the black box to build trust by understanding how they work.
- **Interpretability**
  Observing internal representations helps identify if and why certain outputs are generated. This is key for accountability
- **Debugging**
  Monitoring metrics and artifacts during training and inference can help catch issues early. This is critical for reliability.
- **Auditability**
  Recording activities and data flows provides visibility into the model's operations. This enables audits for ethics and compliance.
- **Performance**
  Tracking metrics like loss, accuracy, etc. gives insight into how well the model is learning and performing. This aids optimization.
- **Safety**
  Detecting signs of harmful intent or behavior early is essential for safe deployment of capable LLMs.
- **Security**
  Monitoring for anomalies and malicious activities improves security against misuse or attacks.

LLM Challenges - Solutions

1. Data freshness
   LLM were trained on static datasets which are now outdated
2. Data source
   Distributed, heterogeneous data sources
3. Short memory
   Context length is limited (new models are supporting bigger context)
4. High Cost - **Monitor usage and cost by tracking API calls**
   long context cost more
5. Latency - **Monitor performance by tracking API calls**
   long output need more inference time (sequential inference, 20 token/s)
6. Data Risk - **Monitor input, output by tracking API calls**
   Quality/Privacy/IP/Bias
7. Hallucination - **Monitor input, output by tracking API calls**

LLM API Mointoring



timeplus

Real-time Streaming Analytics Made Powerful and Accessible!

Thank you!