Building a **Real-Time** ML System

Together

2

Session 1

2024-09-16

Plan for today

1. **Problem framing** → From a business problem to an ML system

Plan for today

1. Problem framing → From a business problem to an ML system

2. **ML System design**

Plan for today

1. Problem framing → From a business problem to an ML system

2. ML System design
   - **Feature-Training-Inference Pipelines**

Plan for today

1. Problem framing → From a business problem to an ML system

2. ML System design
    - Feature-Training-Inference Pipelines
    - **Infrastructure**

Plan for today

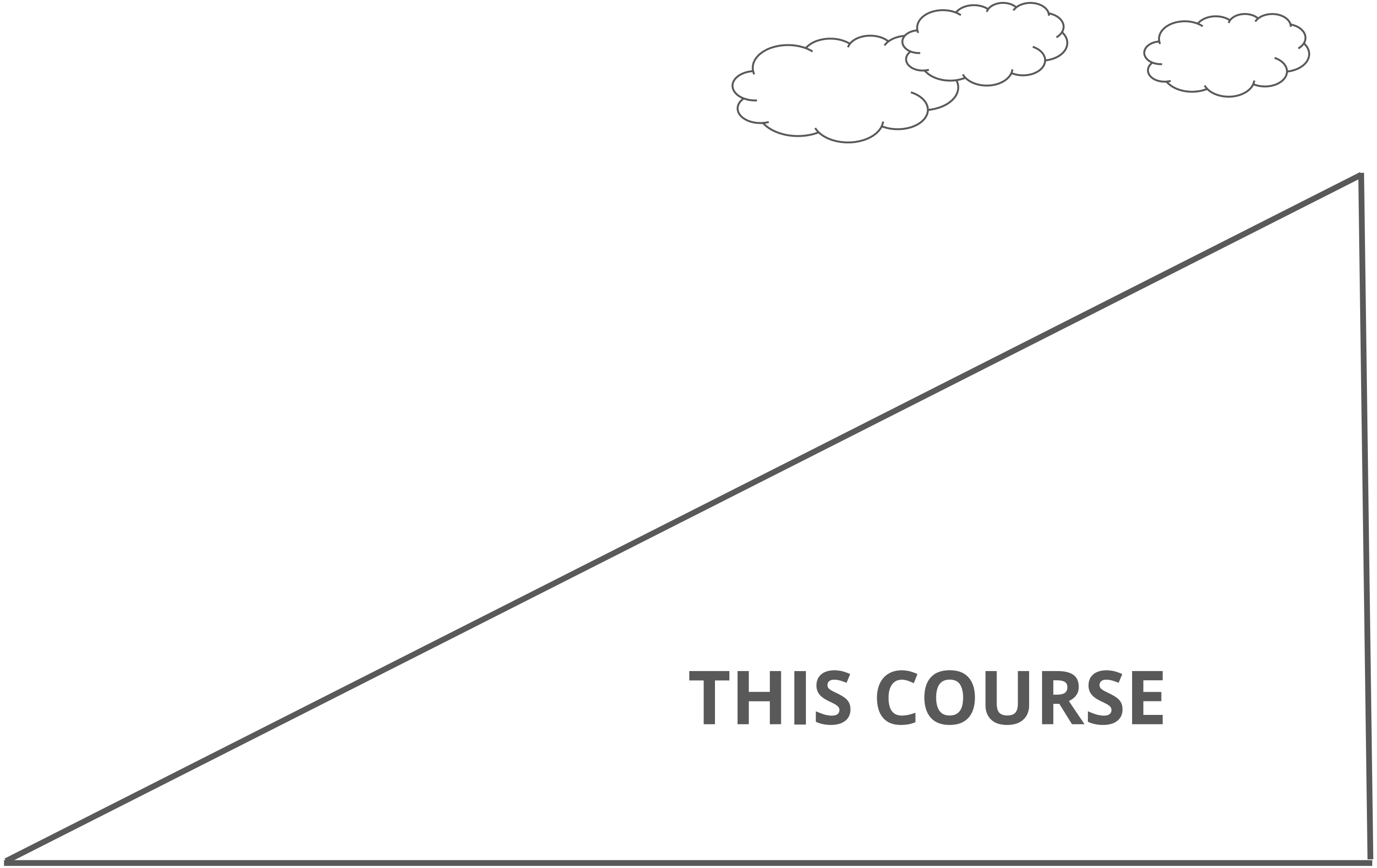
1. Problem framing → From a business problem to an ML system

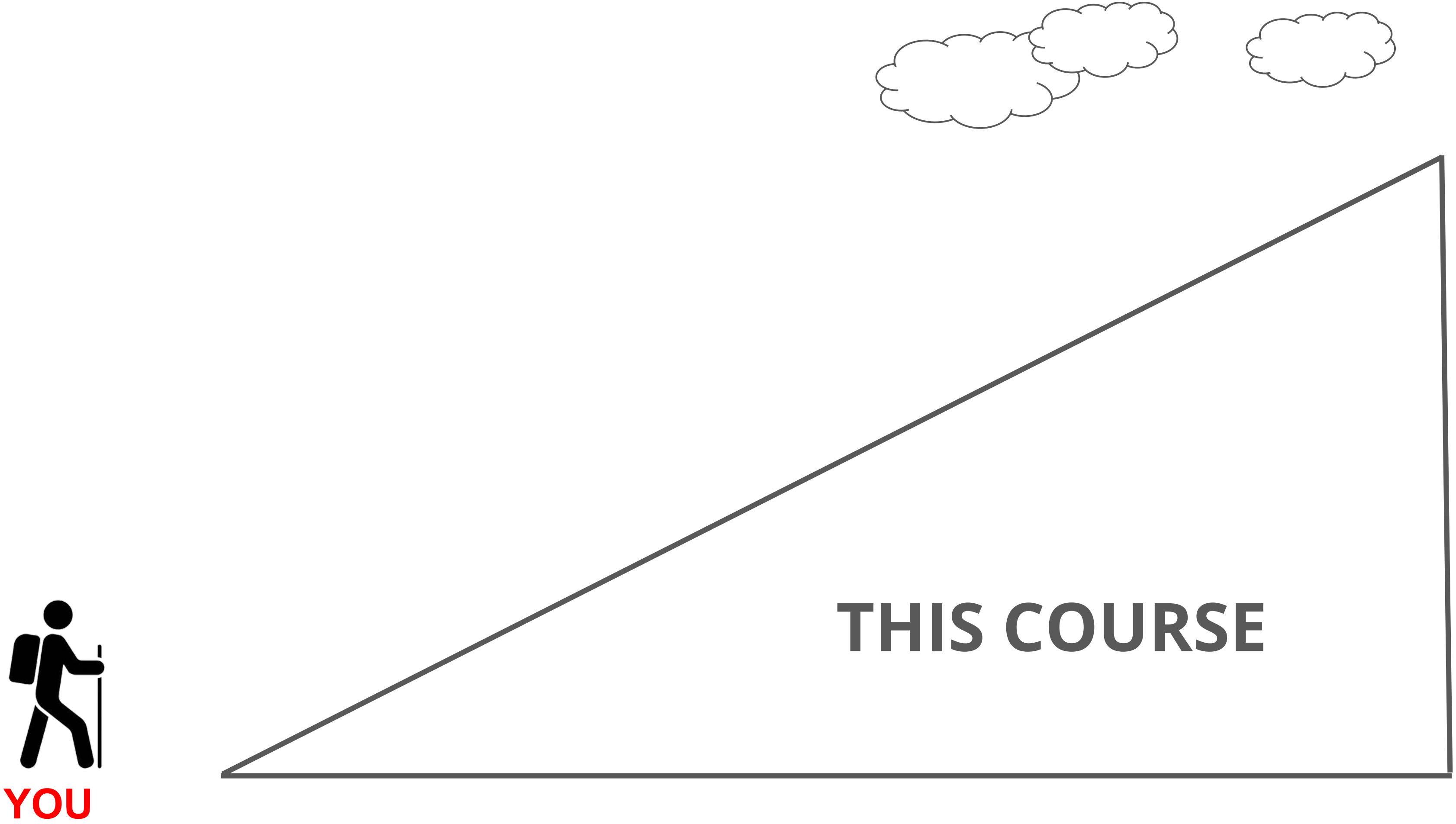
2. ML System design
   - Feature-Training-Inference Pipelines
   - Infrastructure

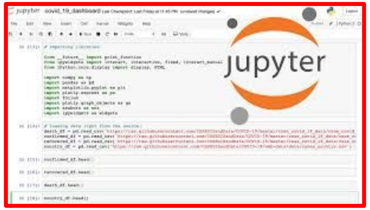
3. **Start building Feature Pipeline**

Are you ready to work hard❓

What is this course about ❓

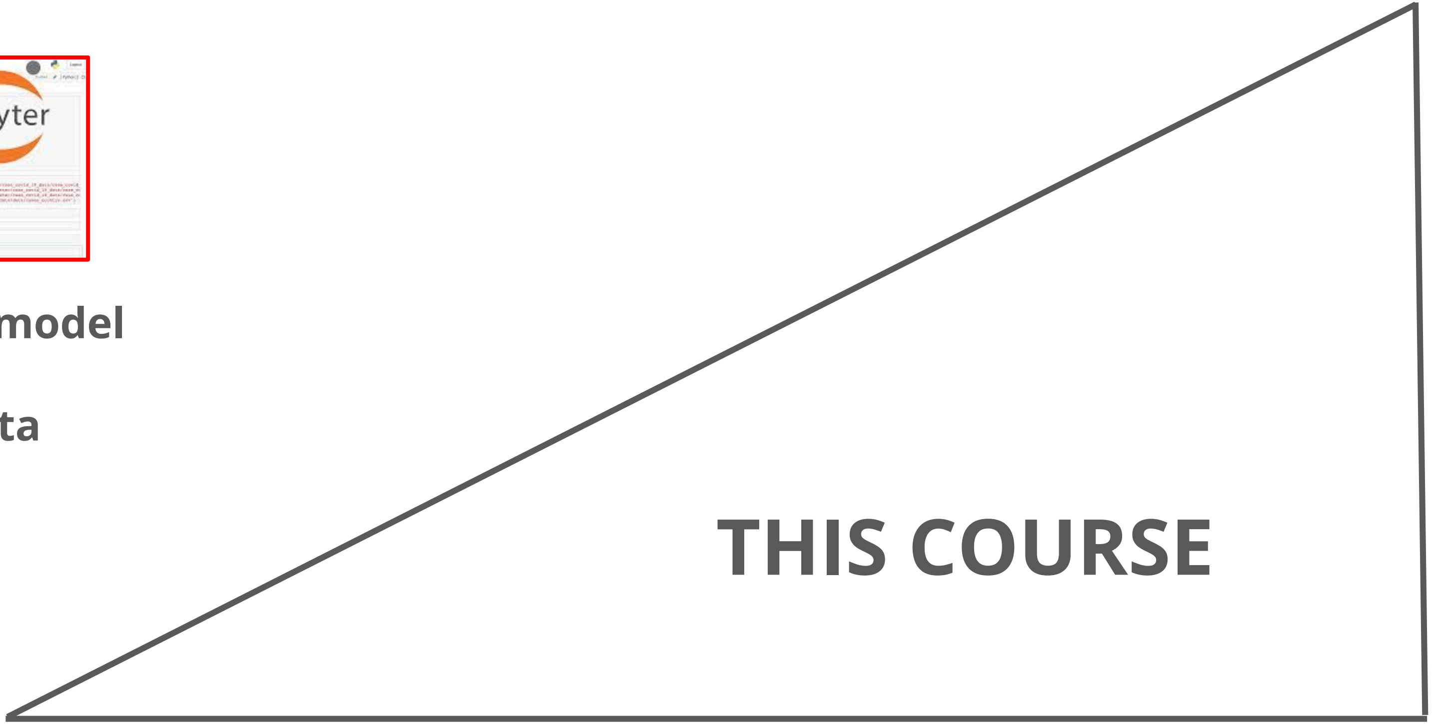

THIS COURSE


THIS COURSE
YOU



Training ML model
from
static data

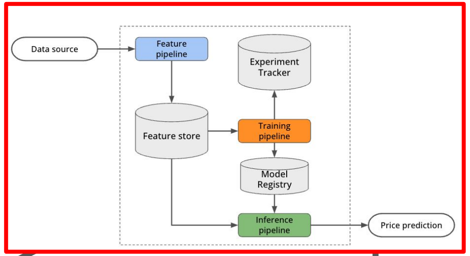
YOU



Training ML model
from
static data

YOU

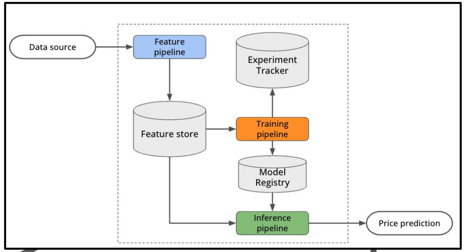
Building ML system
from
real-time data



THIS COURSE

Training ML model
from
static data

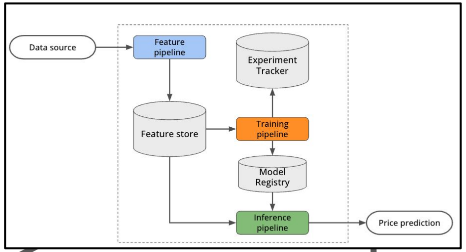
Building ML system
from
real-time data



THIS COURSE

YOU

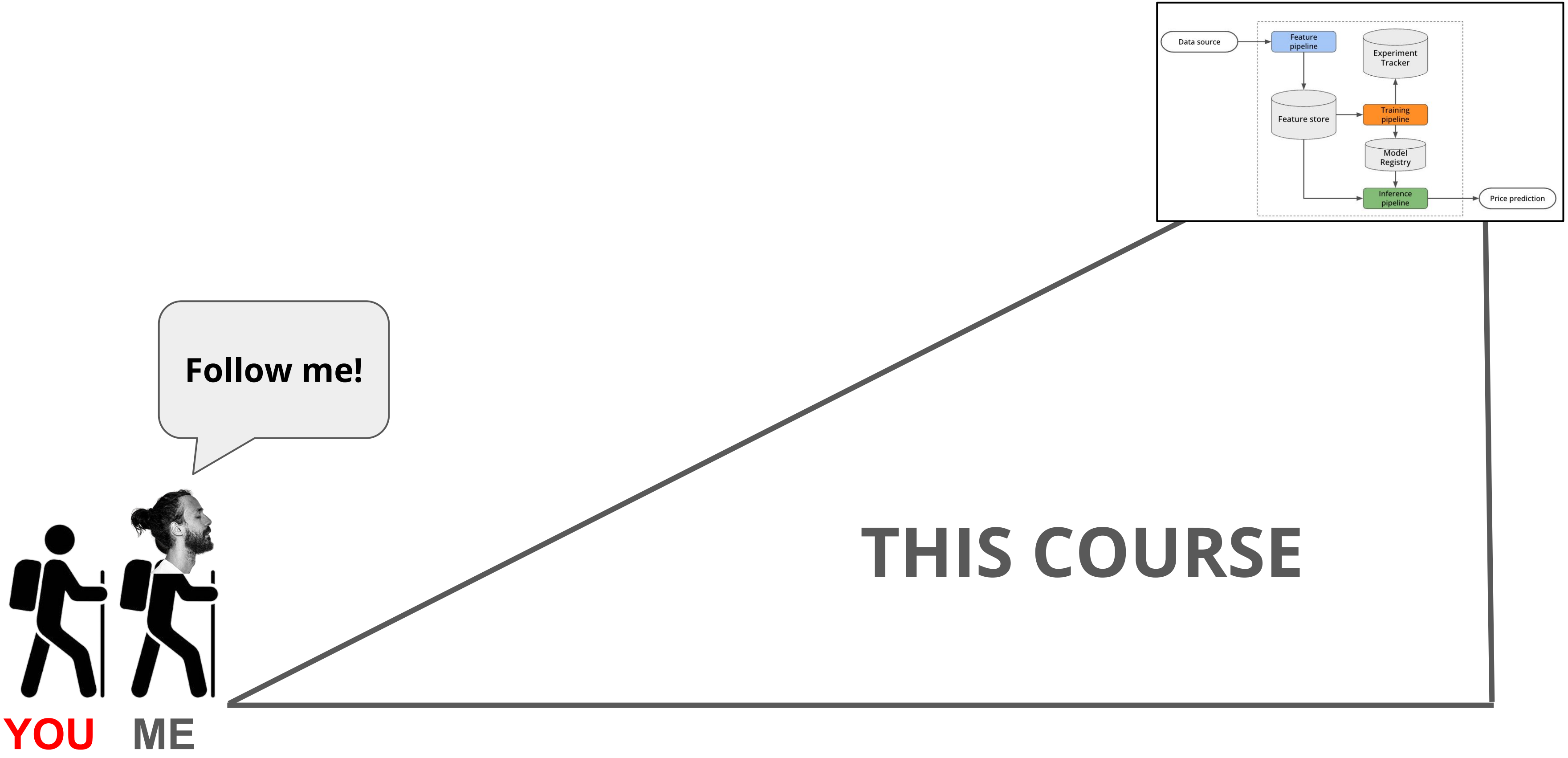
Training ML model
from
static data

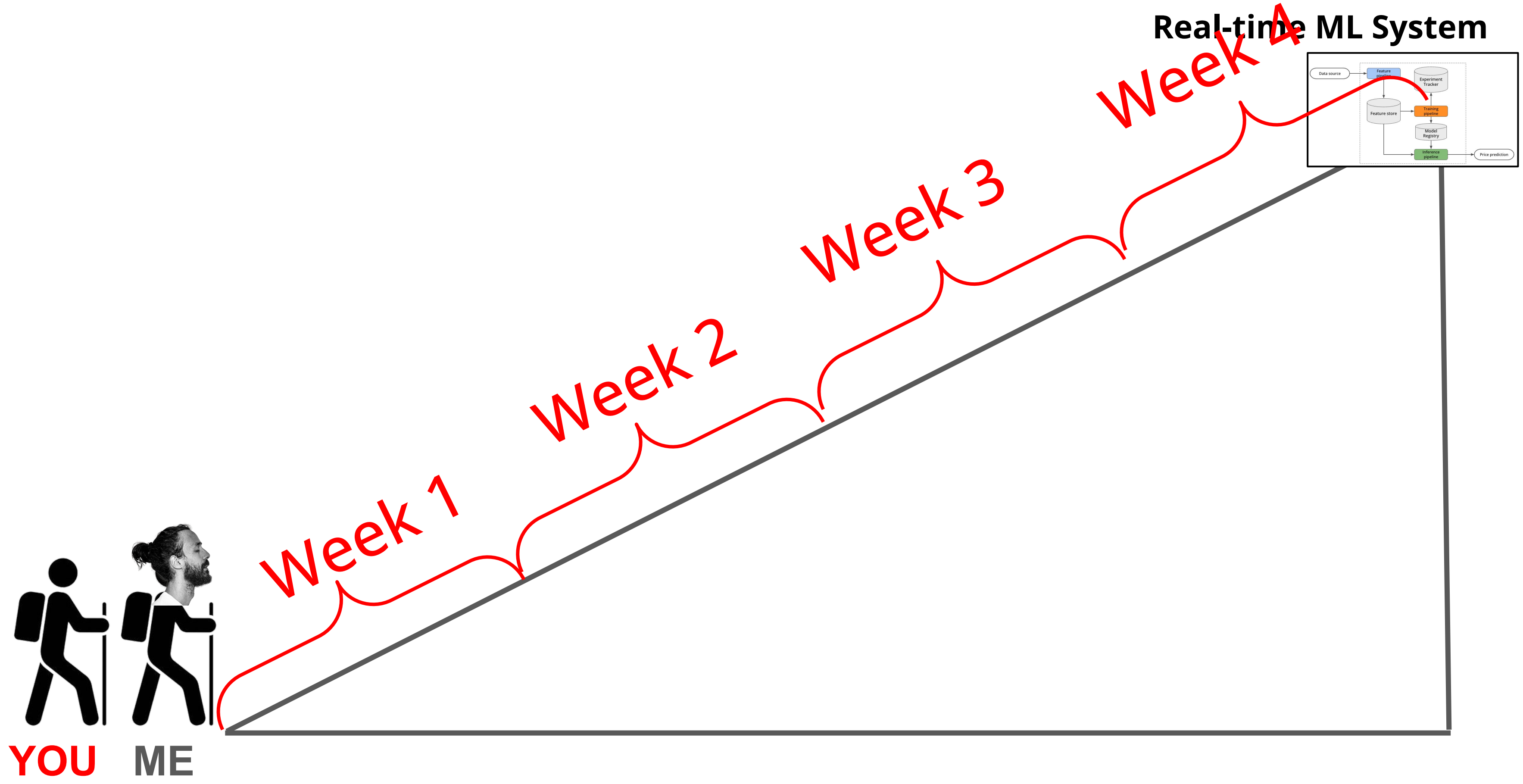


Building ML system
from
real-time data

THIS COURSE

YOU


Data source
Feature pipeline
Experiment Tracker
Feature store
Training pipeline
Model Registry
Inference pipeline
Price prediction
Follow me!
THIS COURSE
YOU
ME



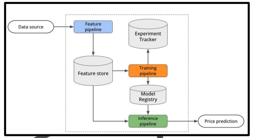
Real-time ML System
Data source
Feature pipeline
Experiment Tracker
Feature store
Training pipeline
Model Registry
Inference pipeline
Price prediction
Week 1
Week 2
Week 3
Week 4
YOU
ME

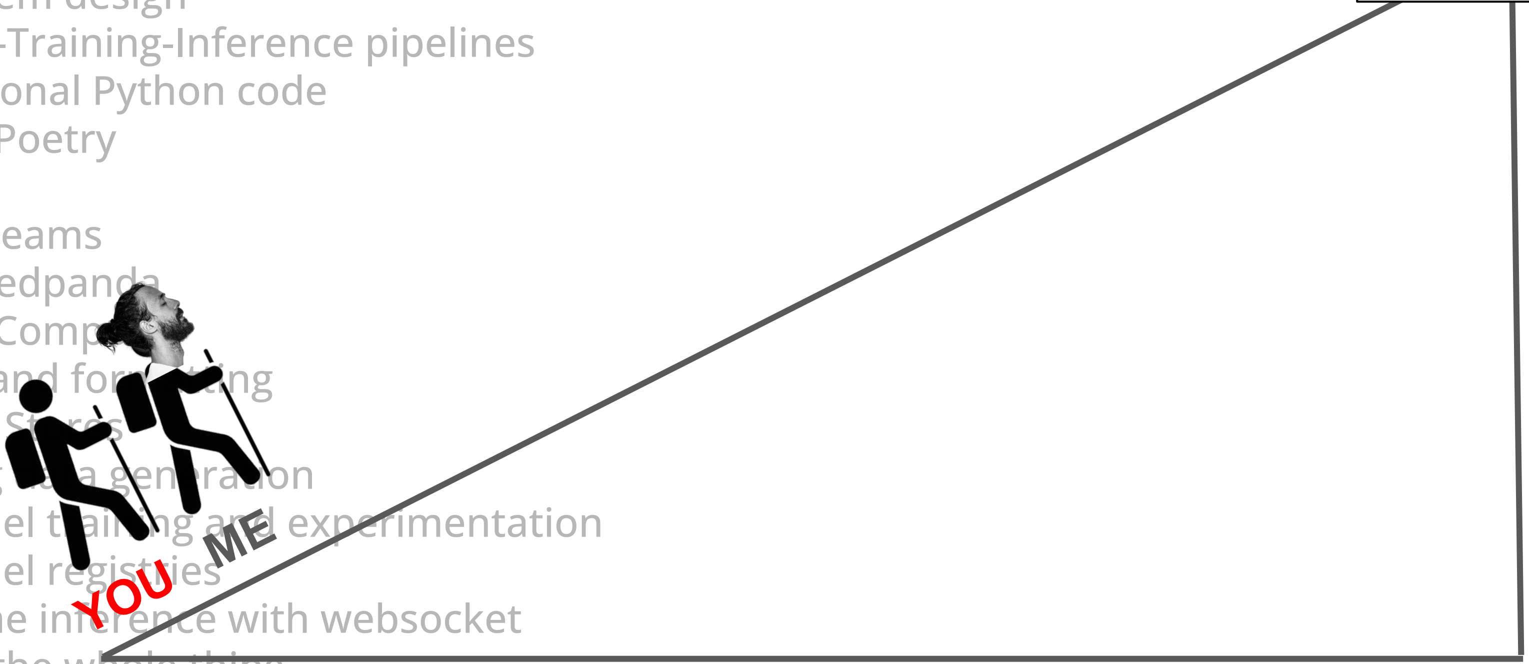
This is what you will learn

> From business problem to ML system

> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

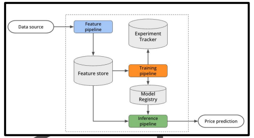
Real-time ML System





This is what you will learn

Real-time ML System

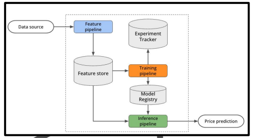


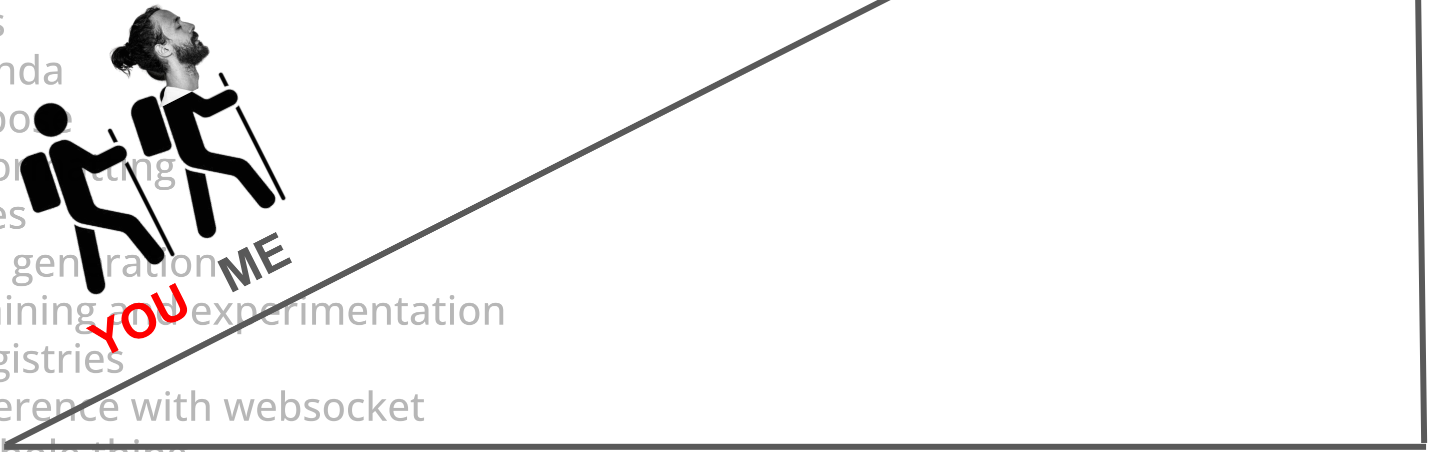
> From business problem to ML system
> **ML System design**
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

This is what you will learn

> From business problem to ML system
> ML System design
> **Feature-Training-Inference pipelines**
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System

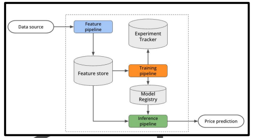




This is what you will learn

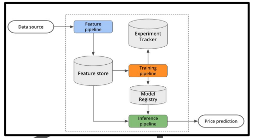
> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> **Professional Python code**
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System



This is what you will learn

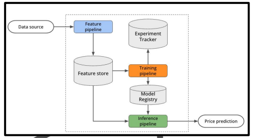
Real-time ML System



> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> **Python Poetry**
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

This is what you will learn

Real-time ML System

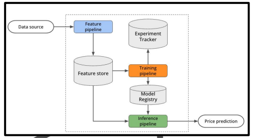


> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> **Docker**
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

This is what you will learn

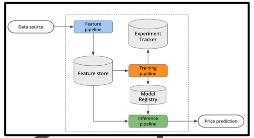
> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> **Quix Streams**
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System



This is what you will learn

Real-time ML System

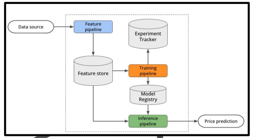


> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> **Kafka/Redpanda**
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> **Docker Compose**
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

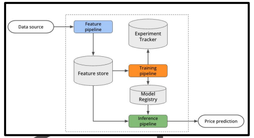
Real-time ML System



This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> **Linting and formatting**
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

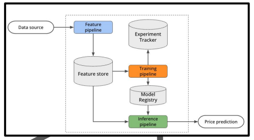
Real-time ML System



This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> **Feature Stores**
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

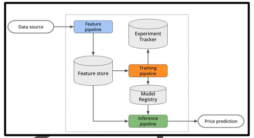
Real-time ML System



This is what you will learn

> From business problem to ML syst
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

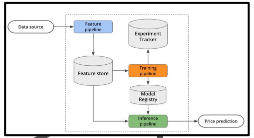
Real-time ML System



This is what you will learn

> From business problem to ML syst
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> **Feature Stores**
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

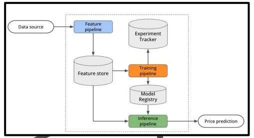
Real-time ML System



This is what you will learn

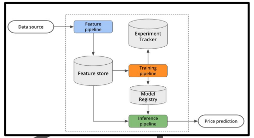
> From business problem to ML syst
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> **Feature Stores**
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System



This is what you will learn

Real-time ML System



> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> **Training data generation**
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

This is what you will learn

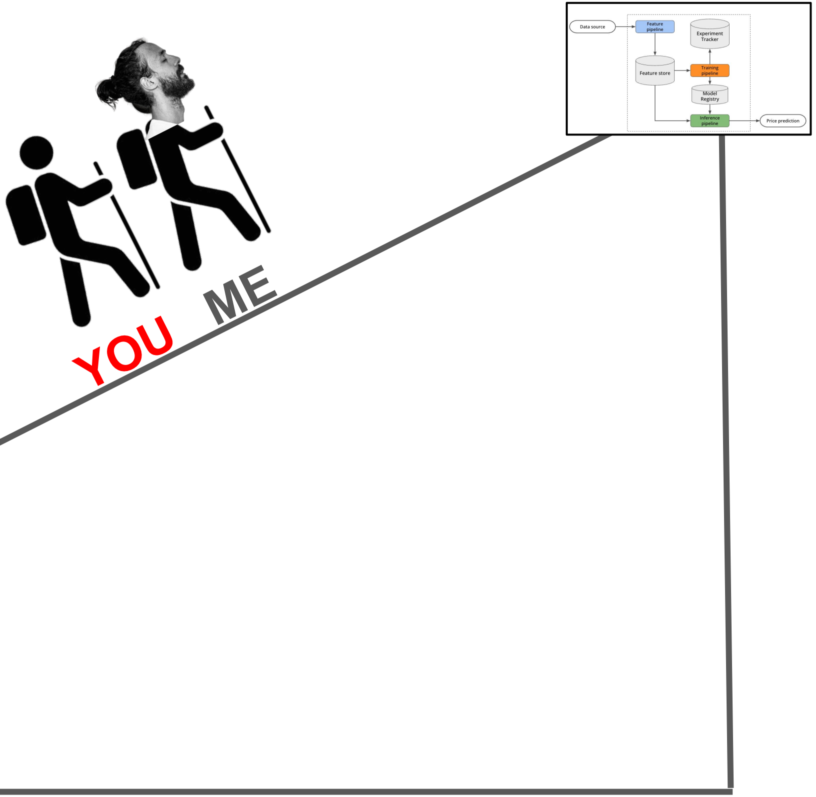
> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> **Training data generation**
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System

This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

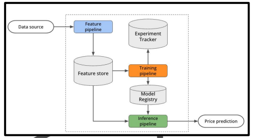
Real-time ML System



This is what you will learn

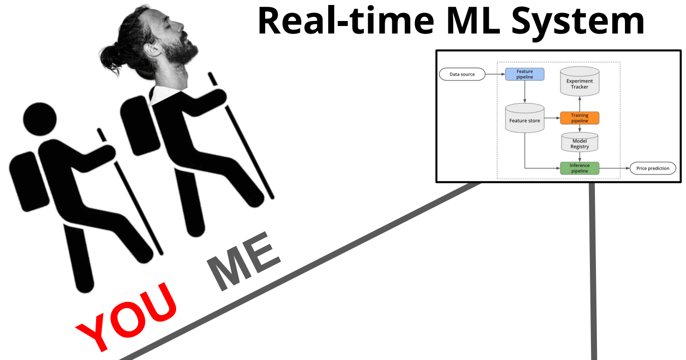
> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> **ML model training and experimentation**
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

Real-time ML System



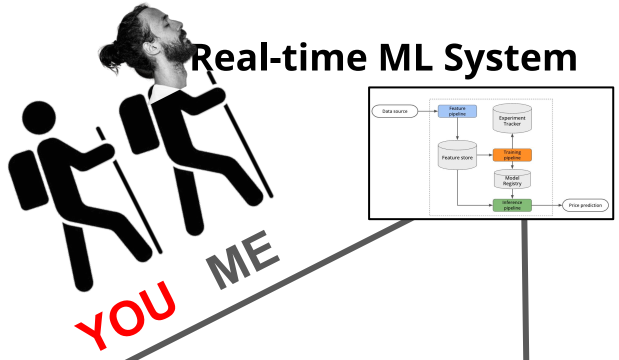
This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> **ML model registries**
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency



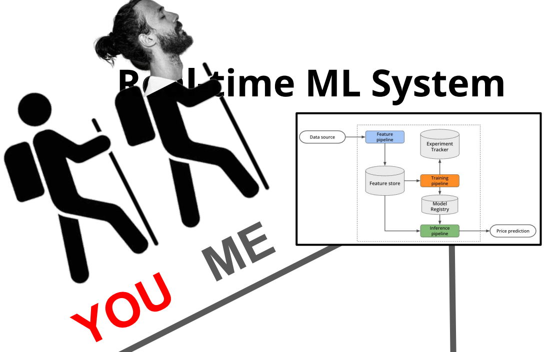
This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency



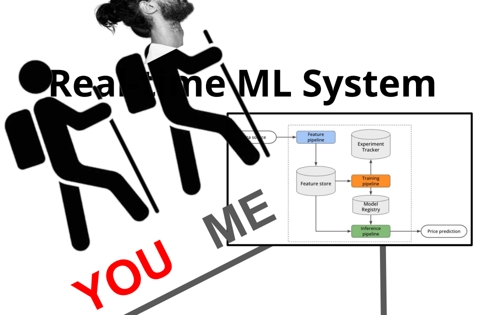
This is what you will learn

> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency



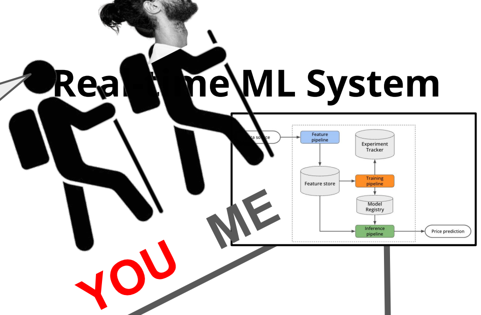
This is what you will learn

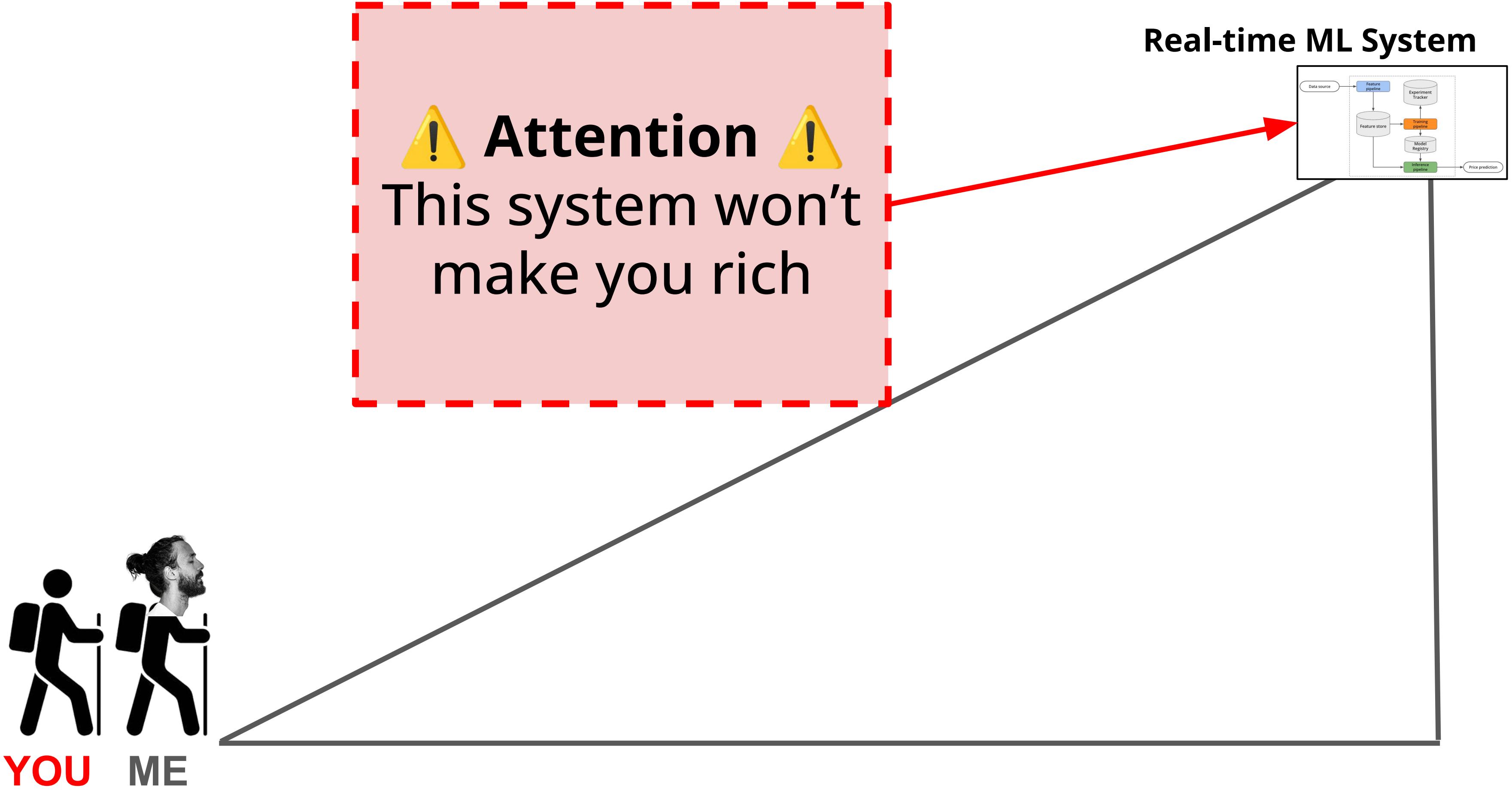
> From business problem to ML system
> ML System design
> Feature-Training-Inference pipelines
> Professional Python code
> Python Poetry
> Docker
> Quix Streams
> Kafka/Redpanda
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency

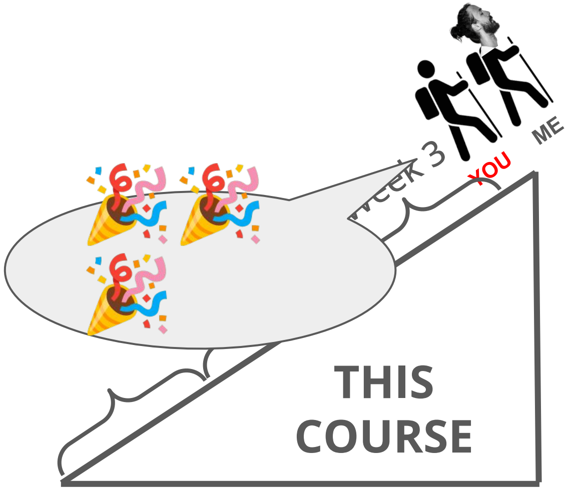


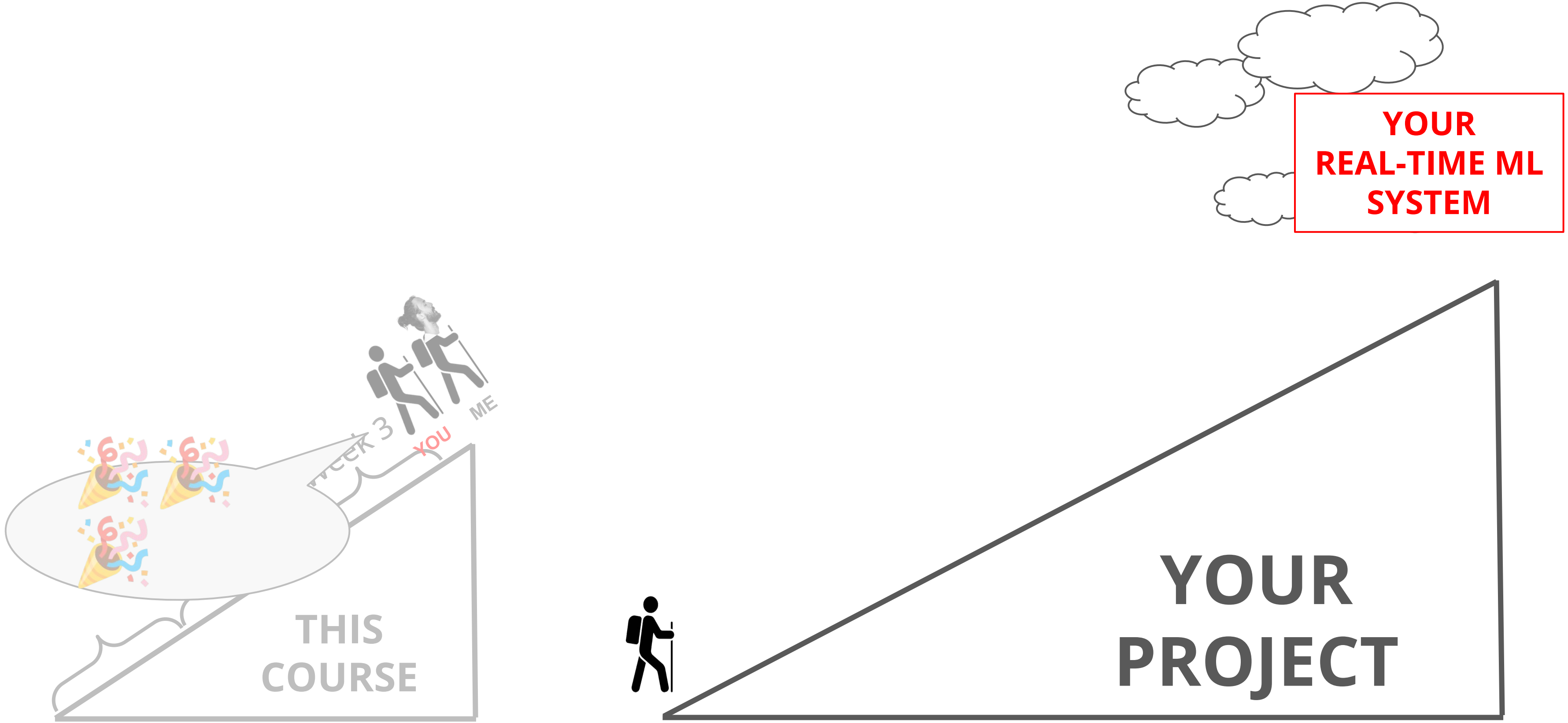
This is what you will learn

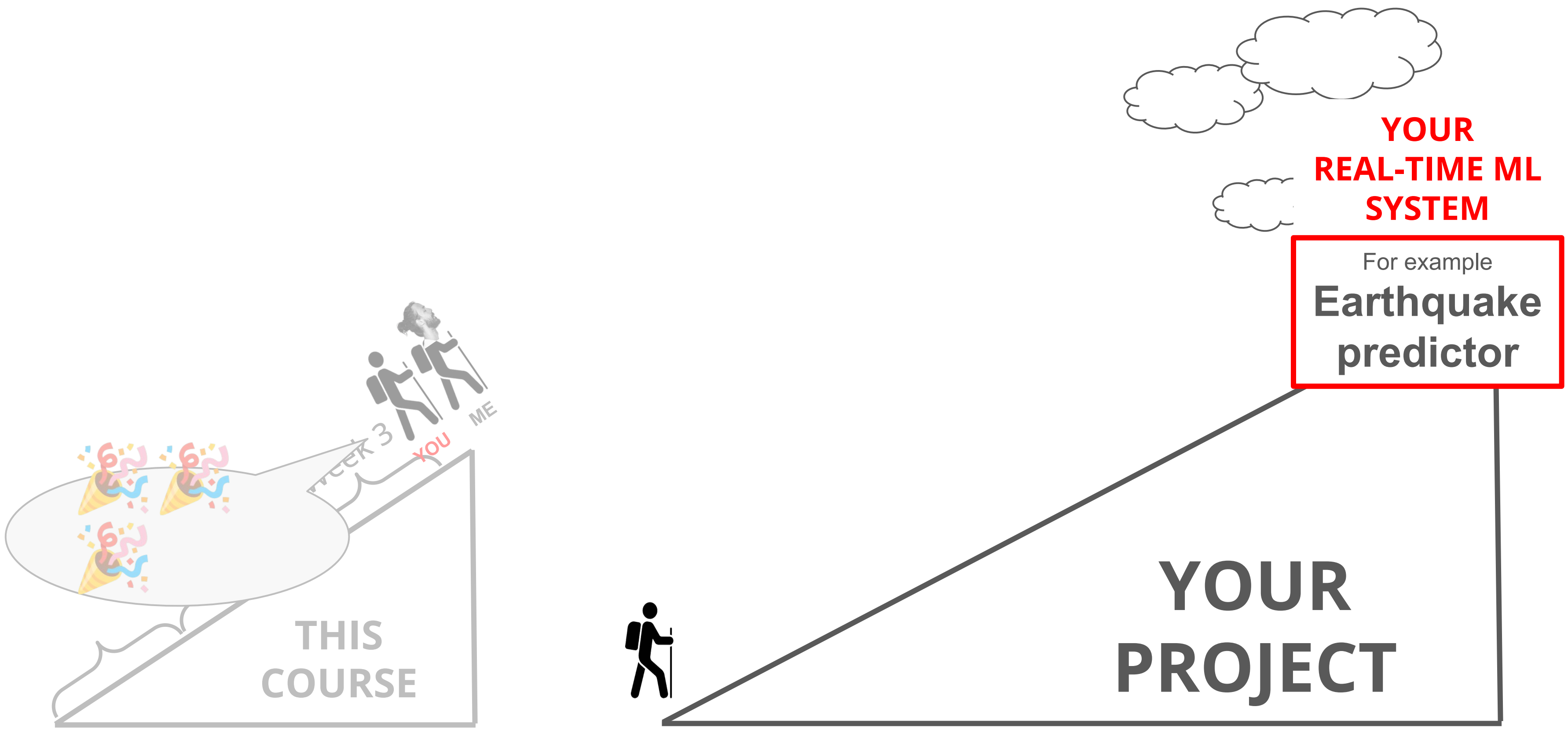
> From business problem
> ML System de
> Feature-T
> Profes
> Pytho
> Docke
> Quix Str
> Kafka/Redpa
> Docker Compose
> Linting and formatting
> Feature Stores
> Training data generation
> ML model training and experimentation
> ML model registries
> Real-time inference with websocket
> Deploy the whole thing
> Monitoring of model and latency



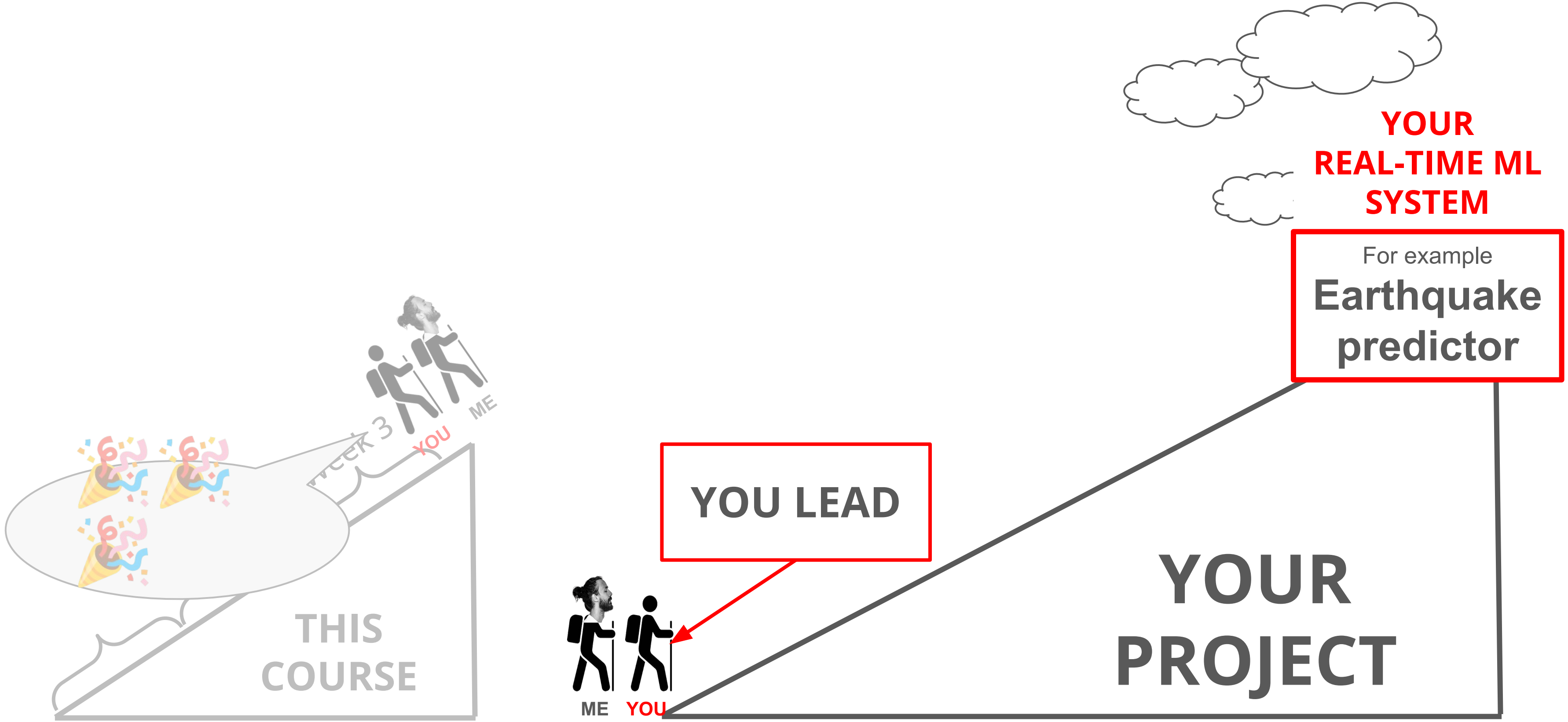

⚠️ Attention ⚠️
This system won't make you rich
Real-time ML System
Data source
Feature pipeline
Experiment Tracker
Feature store
Training pipeline
Model Registry
Inference pipeline
Price prediction
YOU
ME


YOU
ME
THIS
COURSE

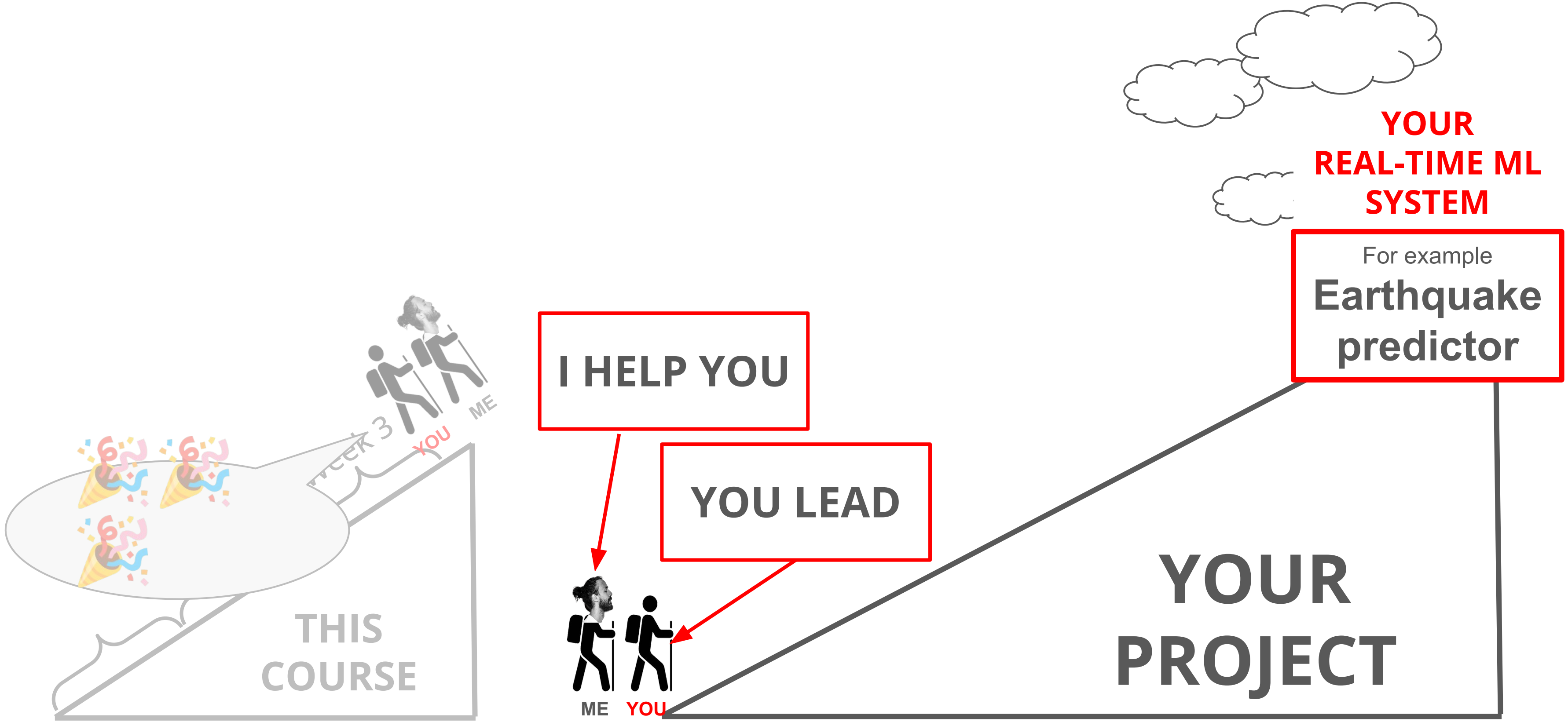

YOUR REAL-TIME ML SYSTEM
ME
YOU
THIS COURSE
YOUR PROJECT



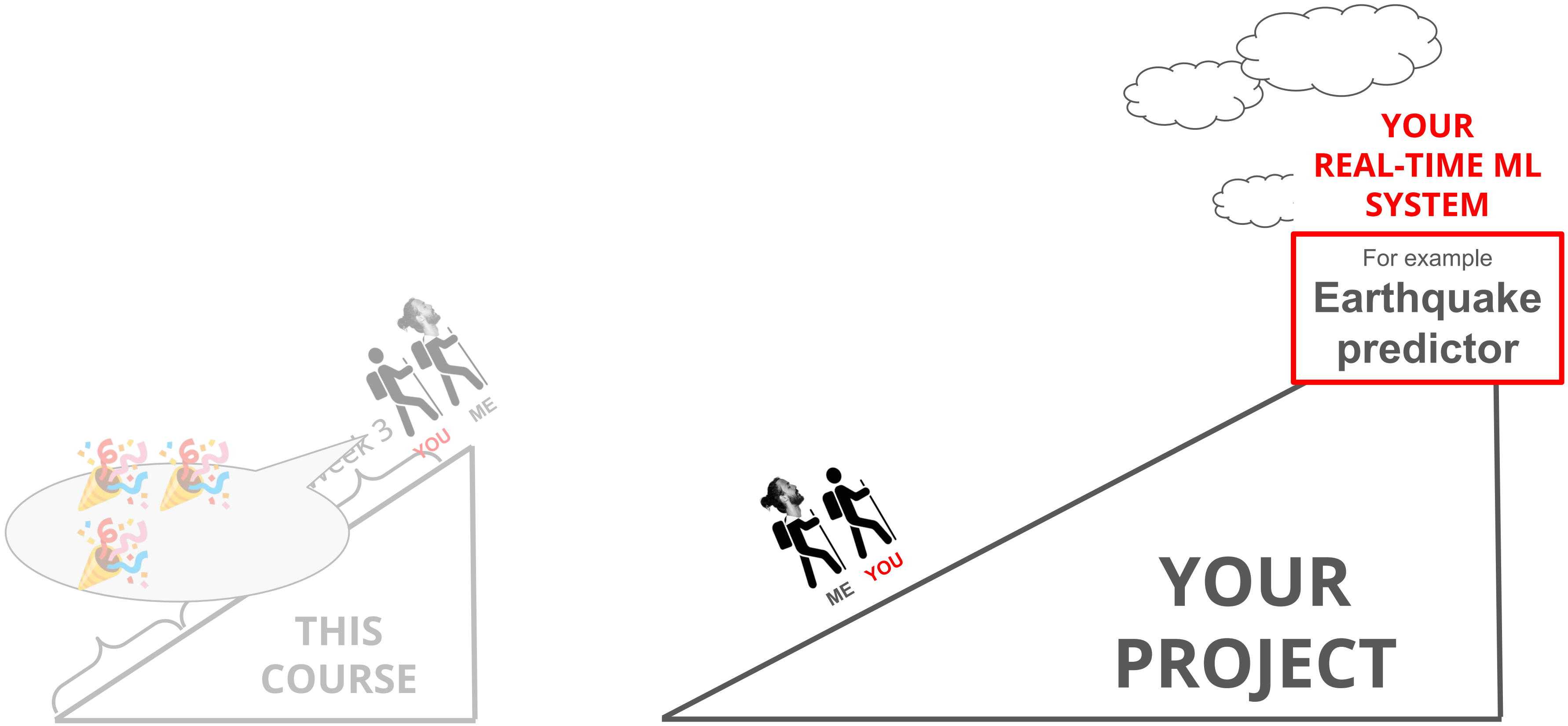
YOUR
REAL-TIME ML
SYSTEM
For example
Earthquake
predictor
YOU
ME
YOUR
PROJECT
THIS
COURSE



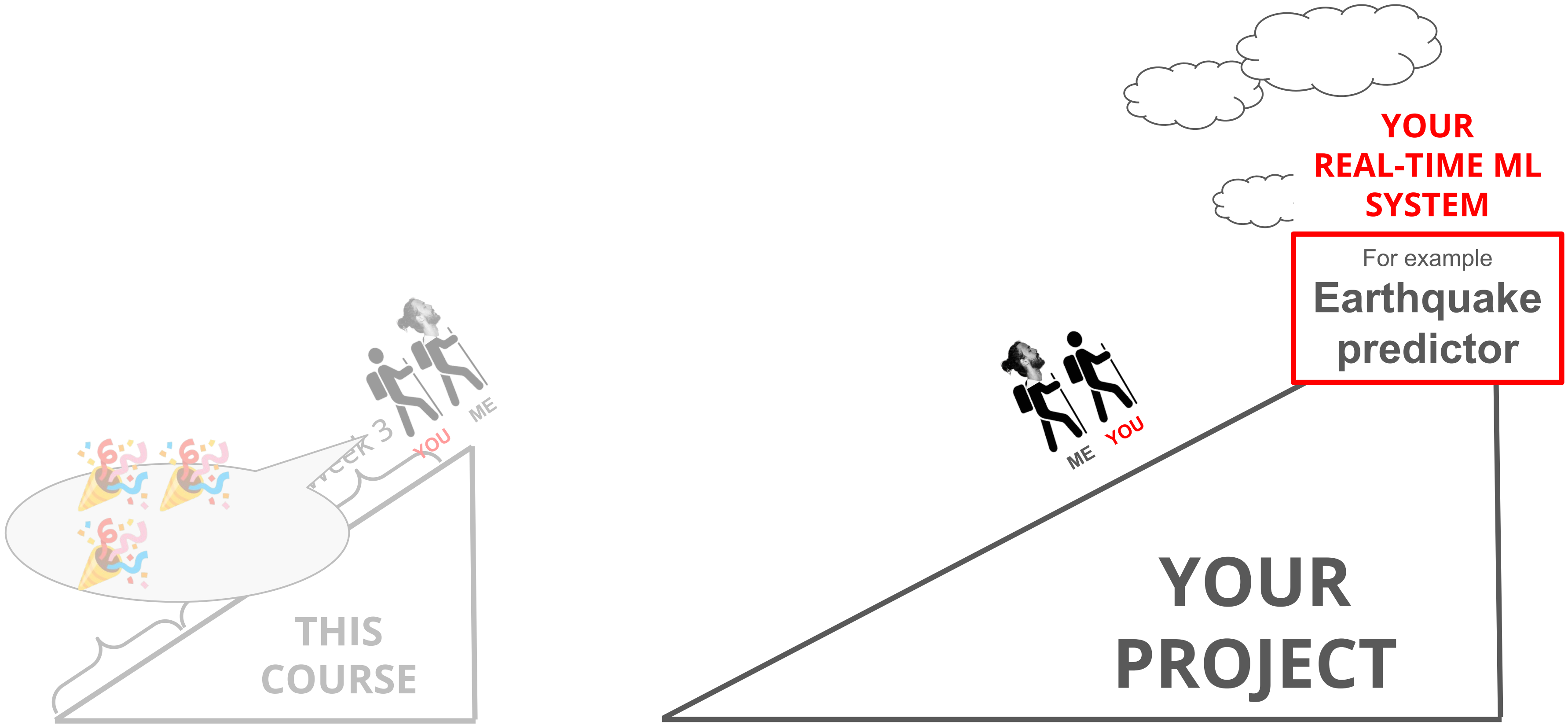
YOUR
REAL-TIME ML
SYSTEM
For example
Earthquake predictor
YOU LEAD
YOUR PROJECT
THIS COURSE
ME
YOU



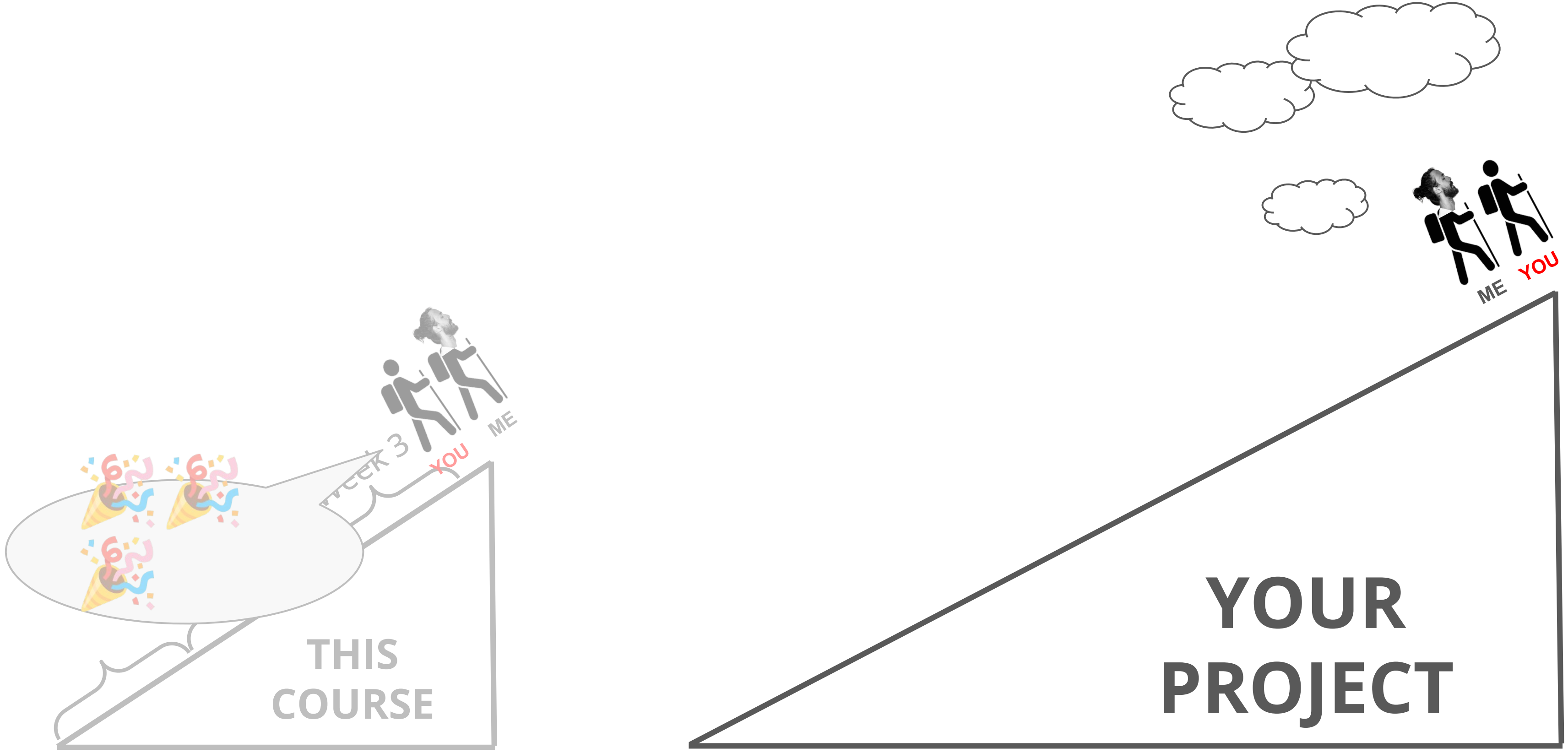
YOUR
REAL-TIME ML
SYSTEM
For example
Earthquake
predictor
I HELP YOU
YOU LEAD
YOUR
PROJECT
THIS
COURSE
Week 3
YOU
ME
ME
YOU



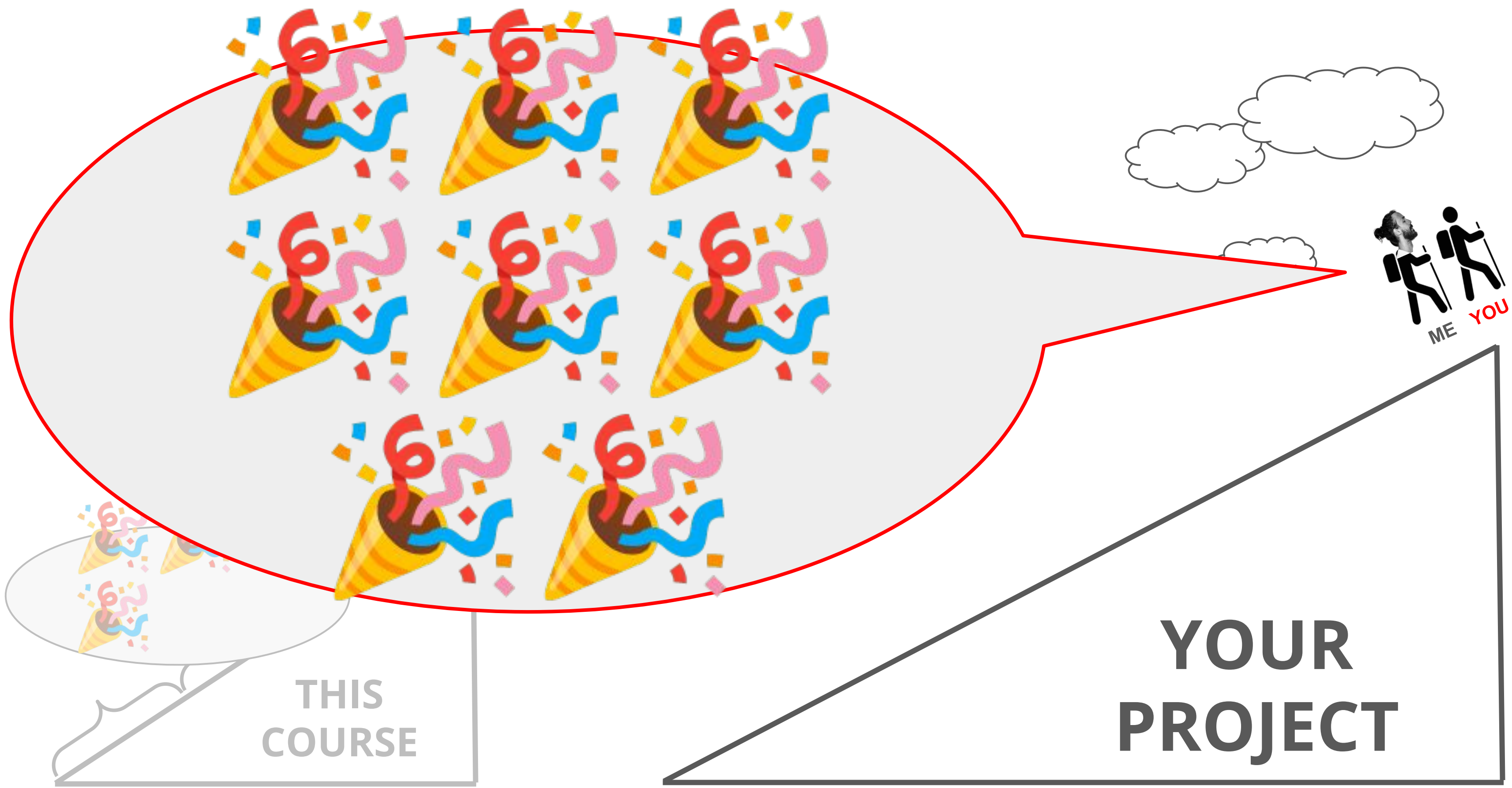
YOUR
REAL-TIME ML
SYSTEM
For example
Earthquake predictor
ME
YOU
YOUR PROJECT
YOU
ME
THIS COURSE

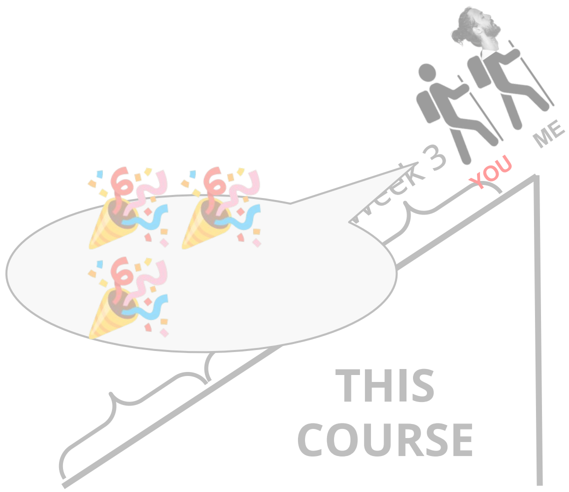


Week 3
YOU
ME
THIS
COURSE
ME
YOU
YOUR
REAL-TIME ML
SYSTEM
For example
Earthquake
predictor
YOUR
PROJECT

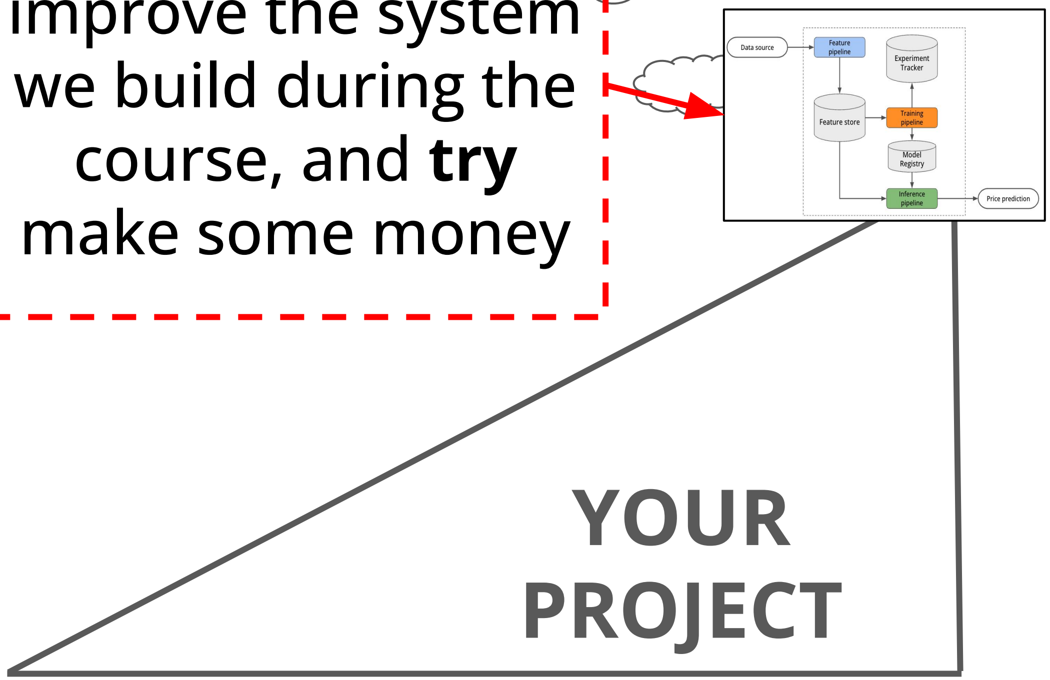


YOU
ME
THIS
COURSE
ME
YOU
YOUR
PROJECT


ME
YOU
THIS COURSE
YOUR PROJECT



You can also improve the system we build during the course, and **try** make some money



Problem framing

From a Business Problem to an ML System

The Business Problem

➔ We work in a trading firm, and want to increase the daily profit of our trading bot.

1. The Business metric we want to impact

➔ We work in a trading firm, and want to increase the daily profit of our trading bot.

1. Business metric (KPI)

2. The End User of our solution

➔ We work in a trading firm, and want to increase the daily profit of our trading bot.

1. Business metric (KPI)

2. End user

3. The ML Proxy metric

- ➔ We work in a trading firm, and want to increase the daily profit of our trading bot.
- ➔ Let's build a system to predict short-term crypto prices

1. Business metric (KPI)

2. End user

3. The ML Proxy metric we will predict

- ➔ We work in a trading firm, and want to increase the daily profit of our trading bot.
- ➔ Let's build a system to predict short-term crypto prices

1. Business metric (KPI)
2. End user
3. The ML Proxy metric

3. The ML Proxy metric we will predict

- ➔ We work in a trading firm, and want to increase the daily profit of our trading bot.
- ➔ Let's build a system to predict short-term crypto prices

1. Business metric (KPI)

2. End user

3. The ML Proxy metric

For example 💁
A system that predicts the price of Ethereum in the **next 10 seconds**.

4. Data Sources

- We work in a trading firm, and want to increase the daily profit of our trading bot.

- Let's build a system to predict short-term crypto prices using real-time market data.

1. Business metric (KPI)

2. End user

3. The ML Proxy metric

4. Data Sources

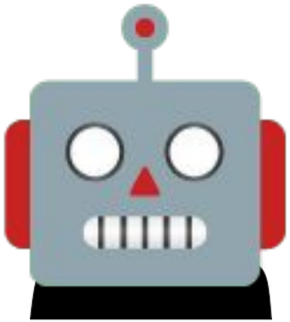
- ➔ We work in a trading firm, and want to increase the daily profit of our trading bot.

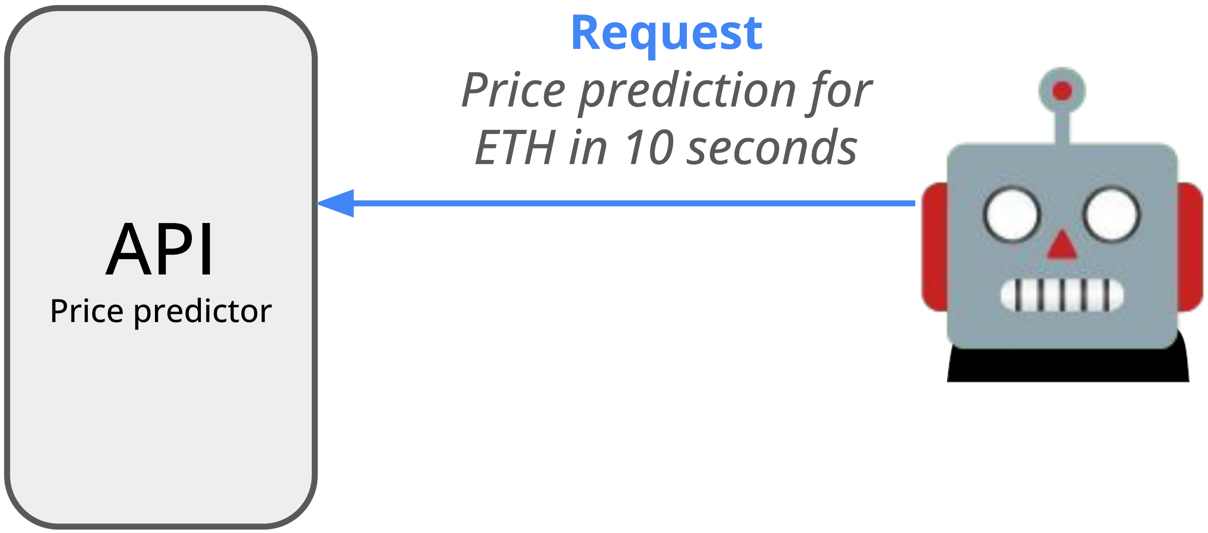
- ➔ Let's build a system to predict short-term crypto prices using real-time market data.

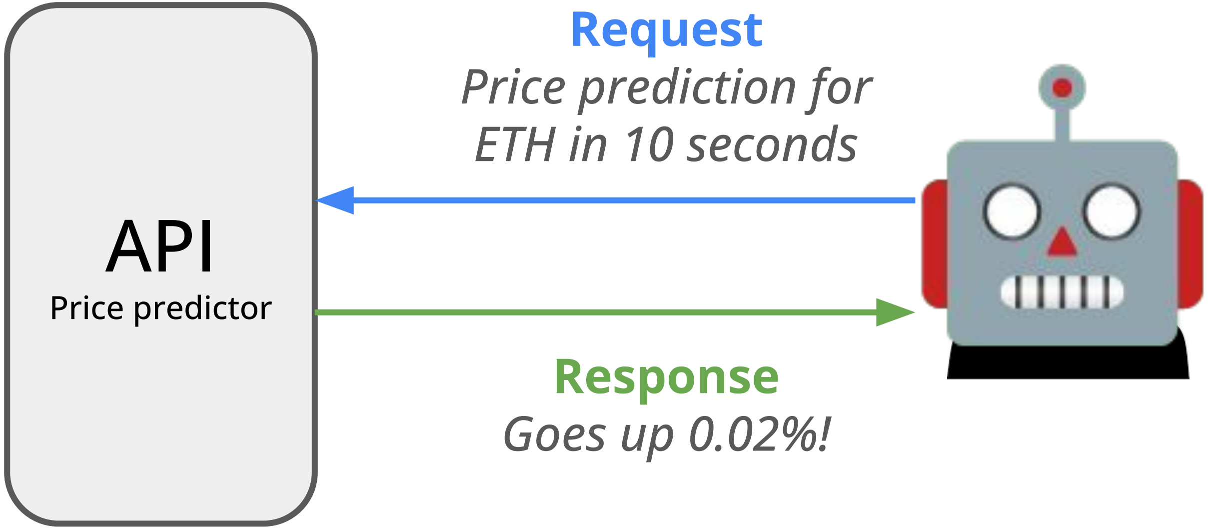
1. Business metric (KPI)
2. End user
3. The ML Proxy metric
4. Data sources

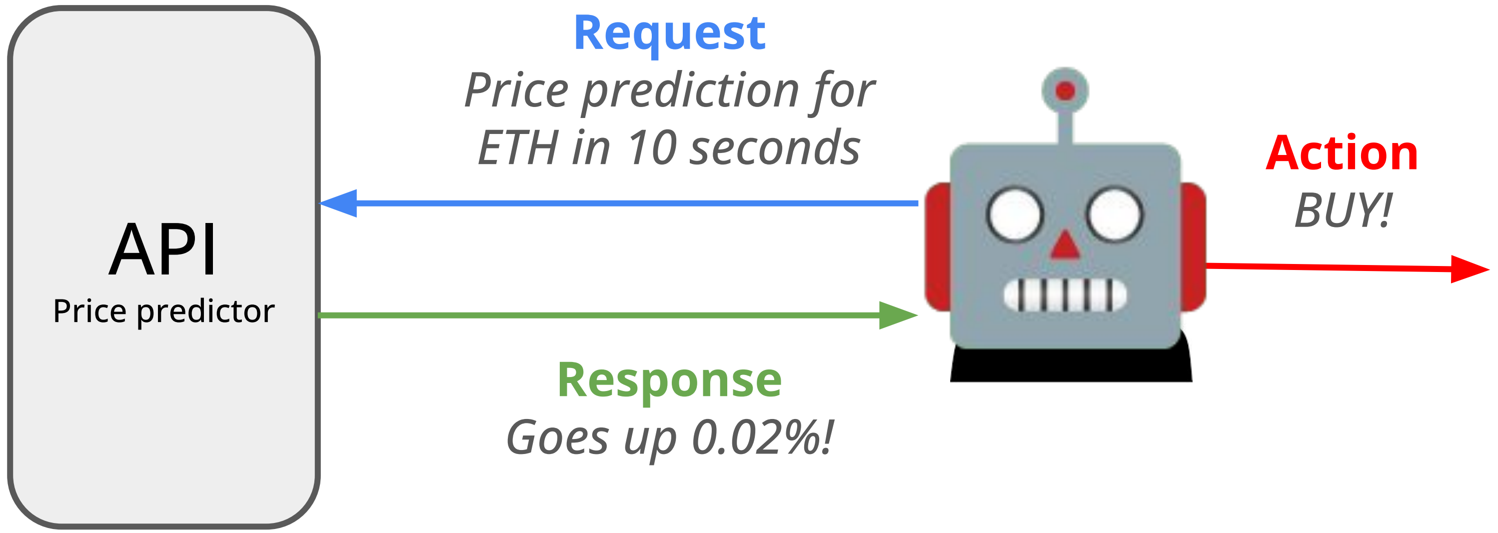
API

Price predictor




API
Price predictor
Request
Price prediction for
ETH in 10 seconds


Request
Price prediction for
ETH in 10 seconds
API
Price predictor
Response
Goes up 0.02%!

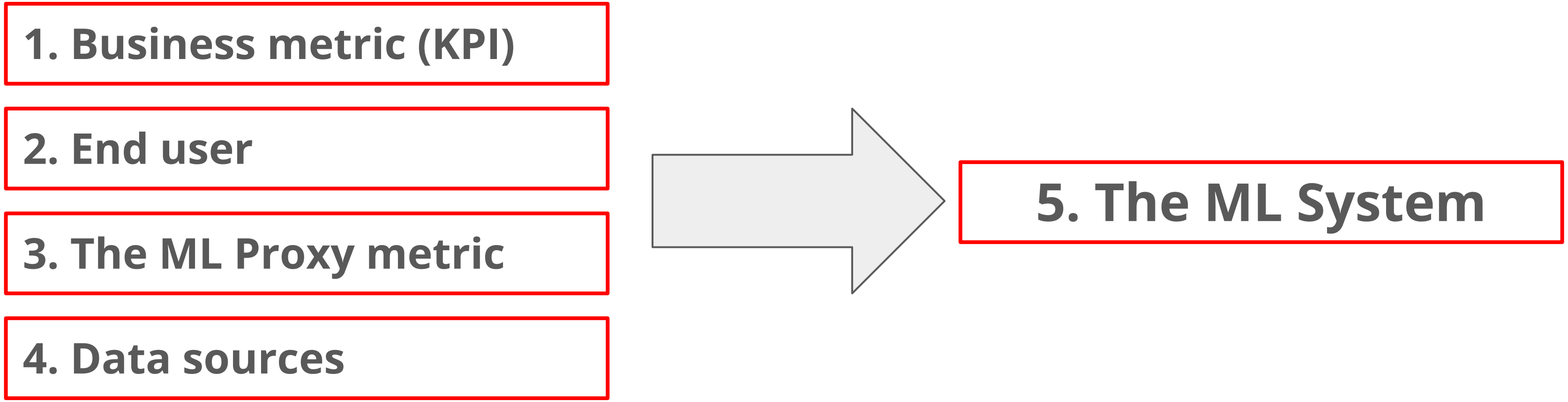

API
Price predictor
Request
Price prediction for ETH in 10 seconds
Response
Goes up 0.02%!
Action
BUY!

1. Business metric (KPI)

2. End user

3. The ML Proxy metric

4. Data sources


1. Business metric (KPI)
2. End user
3. The ML Proxy metric
4. Data sources
5. The ML System

1. Business metric (KPI)

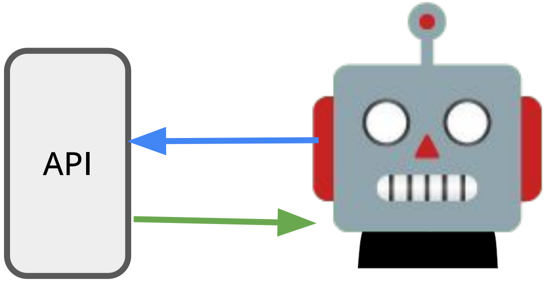
2. End user

3. The ML Proxy metric

4. Data sources

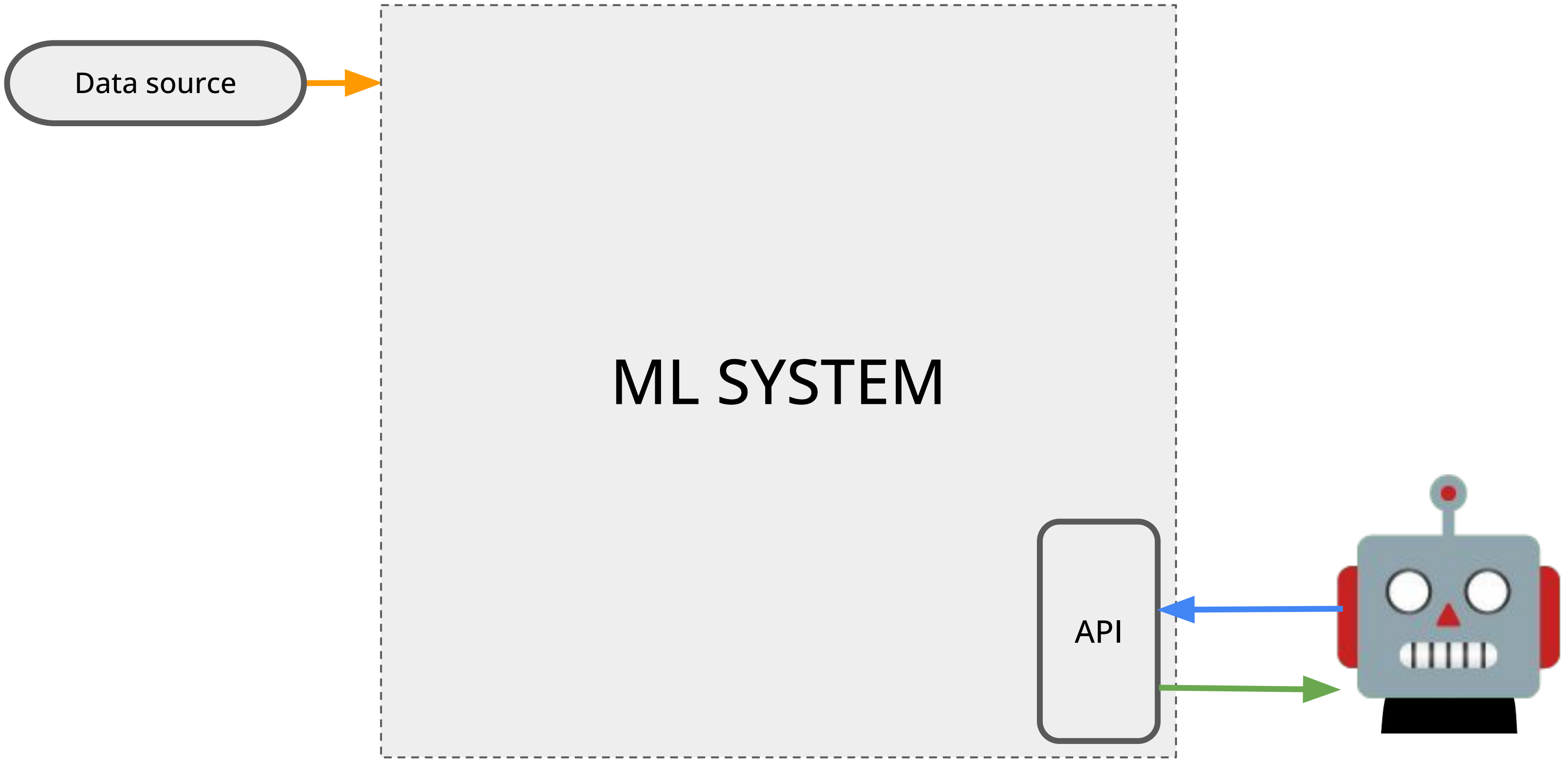
HOW?

5. The ML System


API

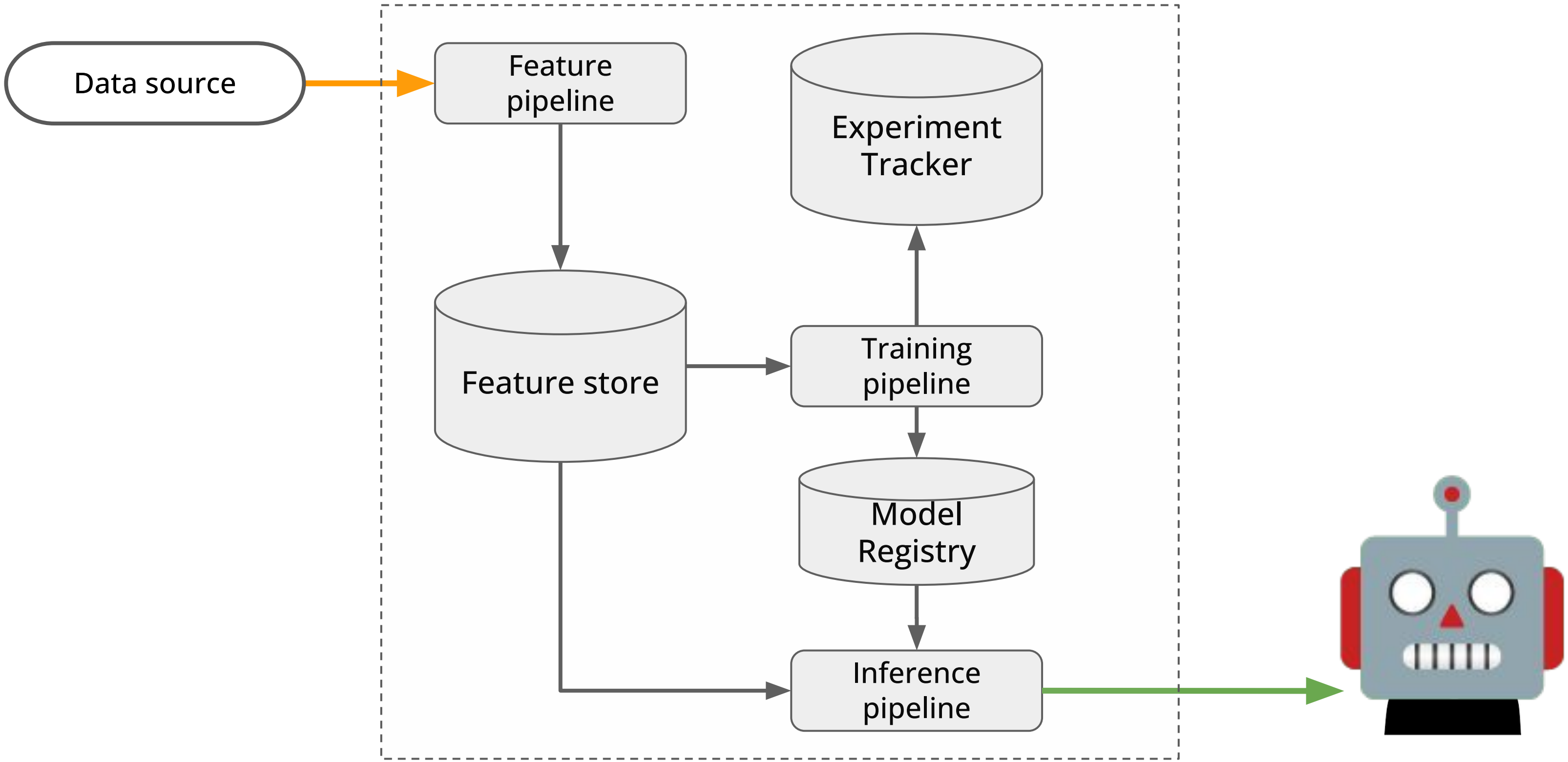
The **ML system** behind our API

From **raw market data** to **real-time predictions**

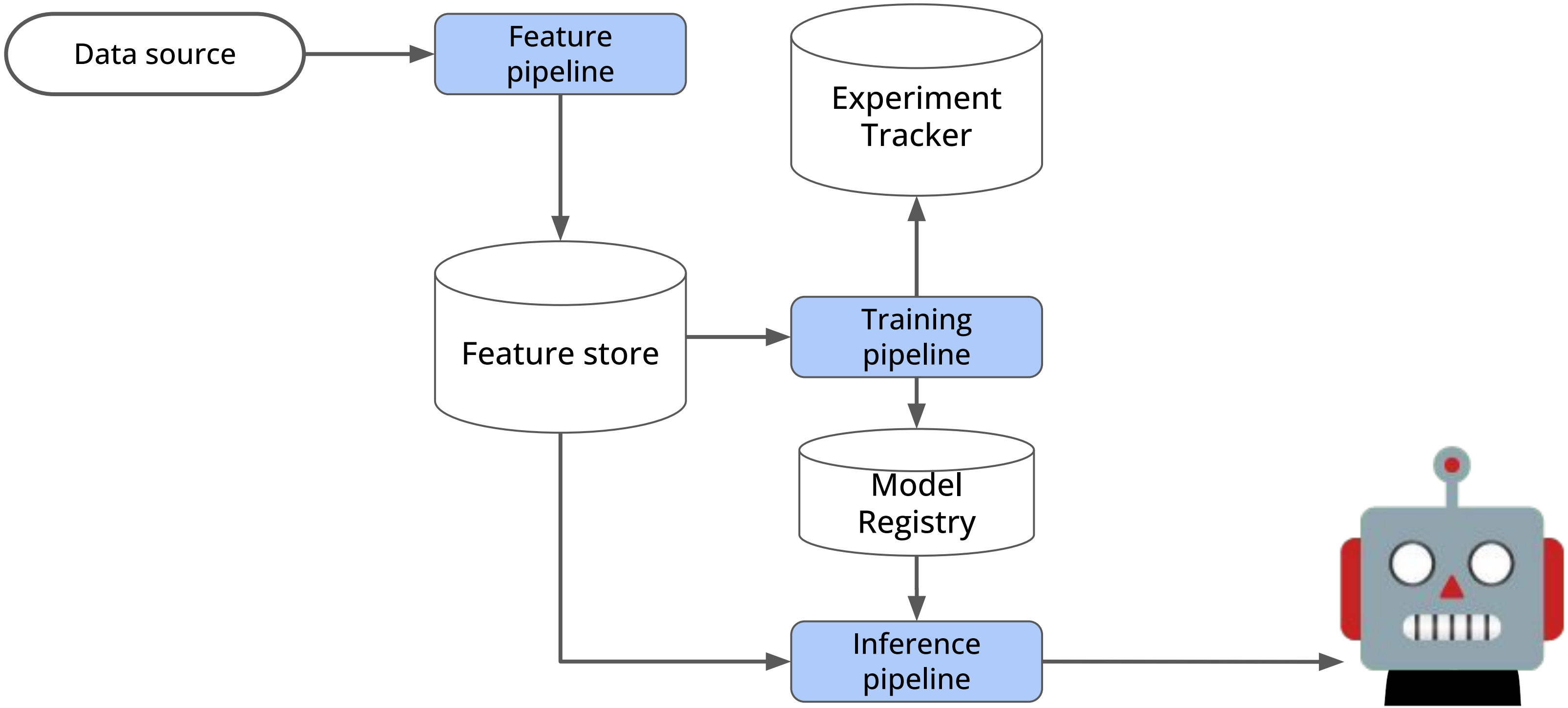


The **ML system** behind our API

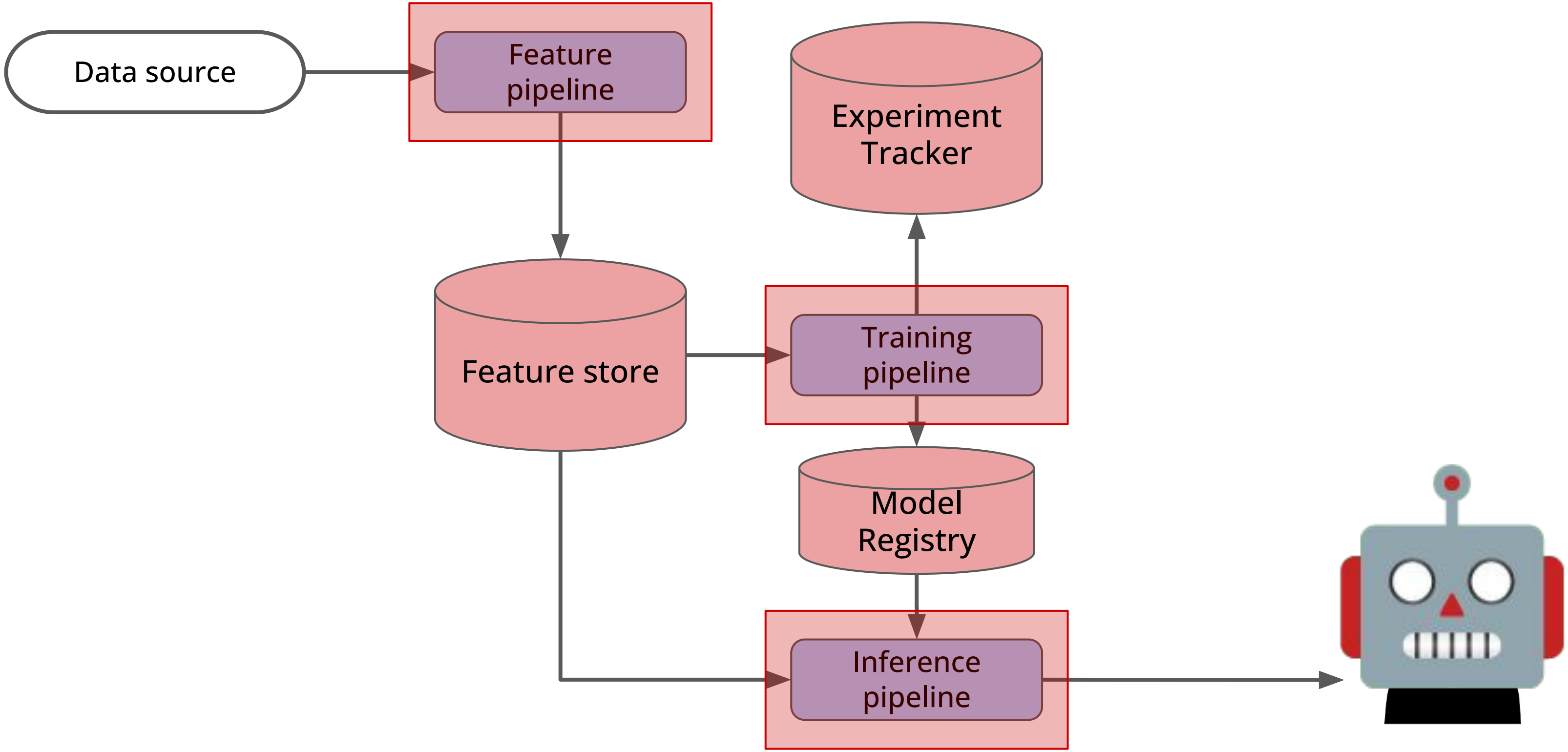
From **raw market data** to **real-time predictions**



ML system = Pipelines



ML system = Pipelines + Infrastructure



What is a Pipeline?

A Pipeline

➔ is a **computer program (e.g Python script)**

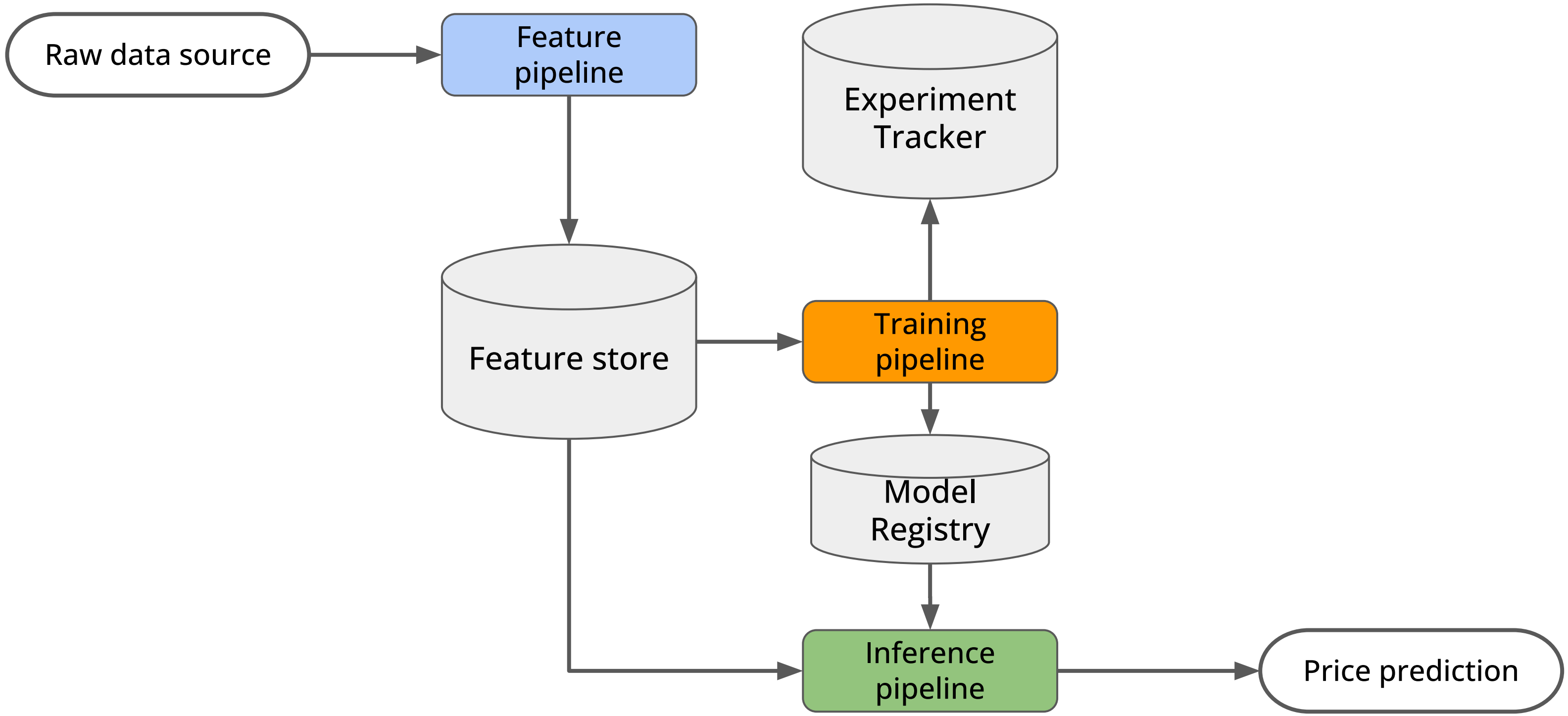
A Pipeline

- ➔ is a computer program **(e.g Python script)**
- ➔ with clearly defined **inputs** and **outputs**

A Pipeline

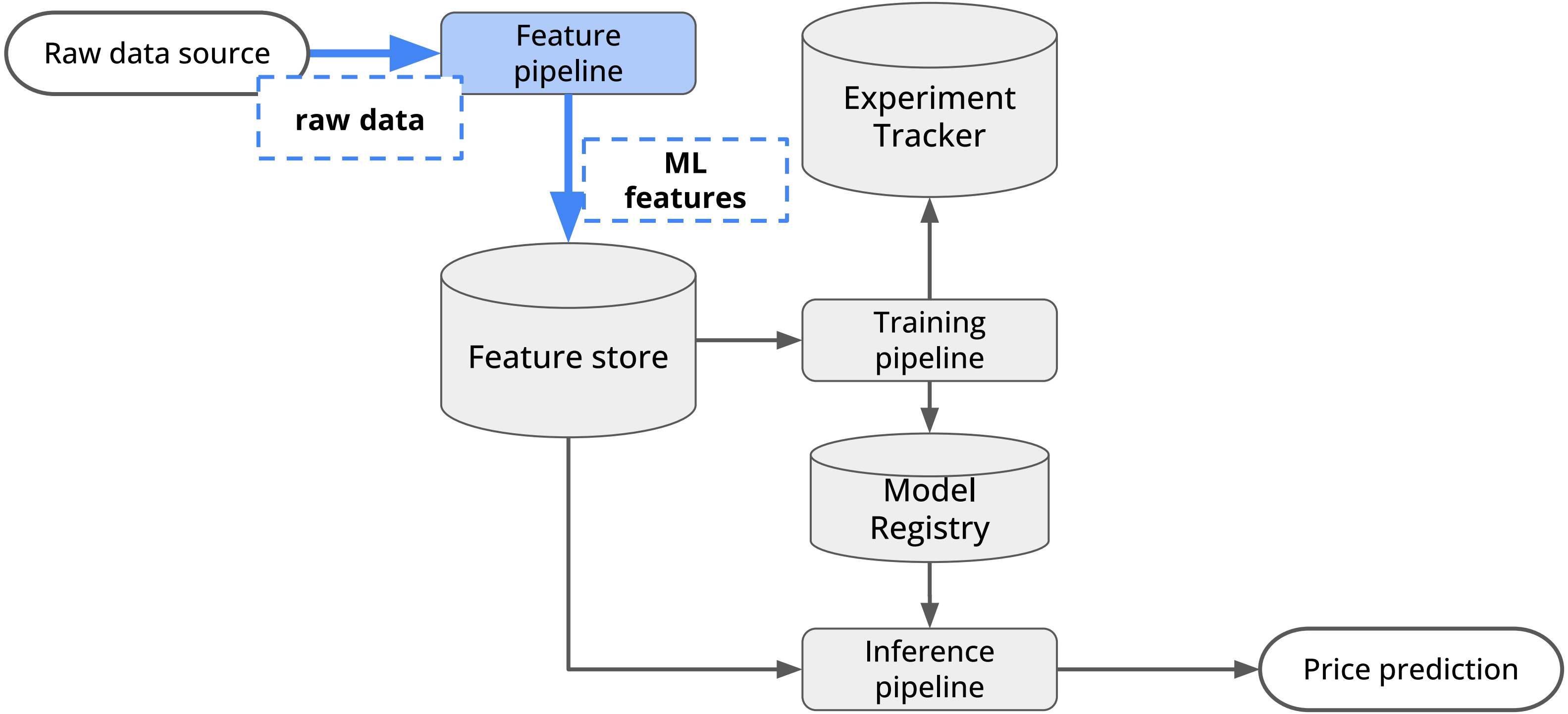
- ➔ is a computer program
- ➔ with clearly defined inputs and outputs
- ➔ that runs on a **schedule** or **continuously**

Feature-Training-Inference pipelines



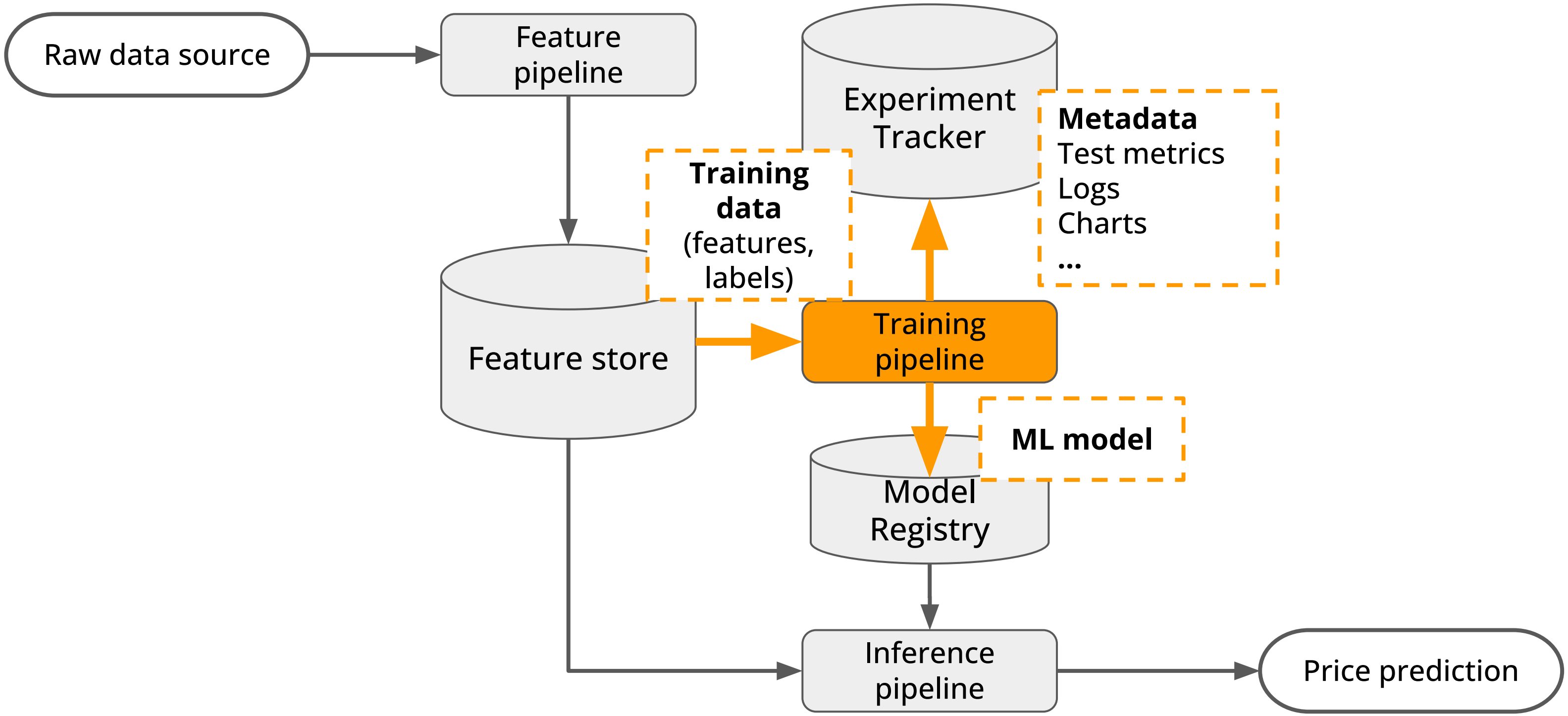
1. Feature pipeline

From **raw data** to ***reusable*** **features** for ML models



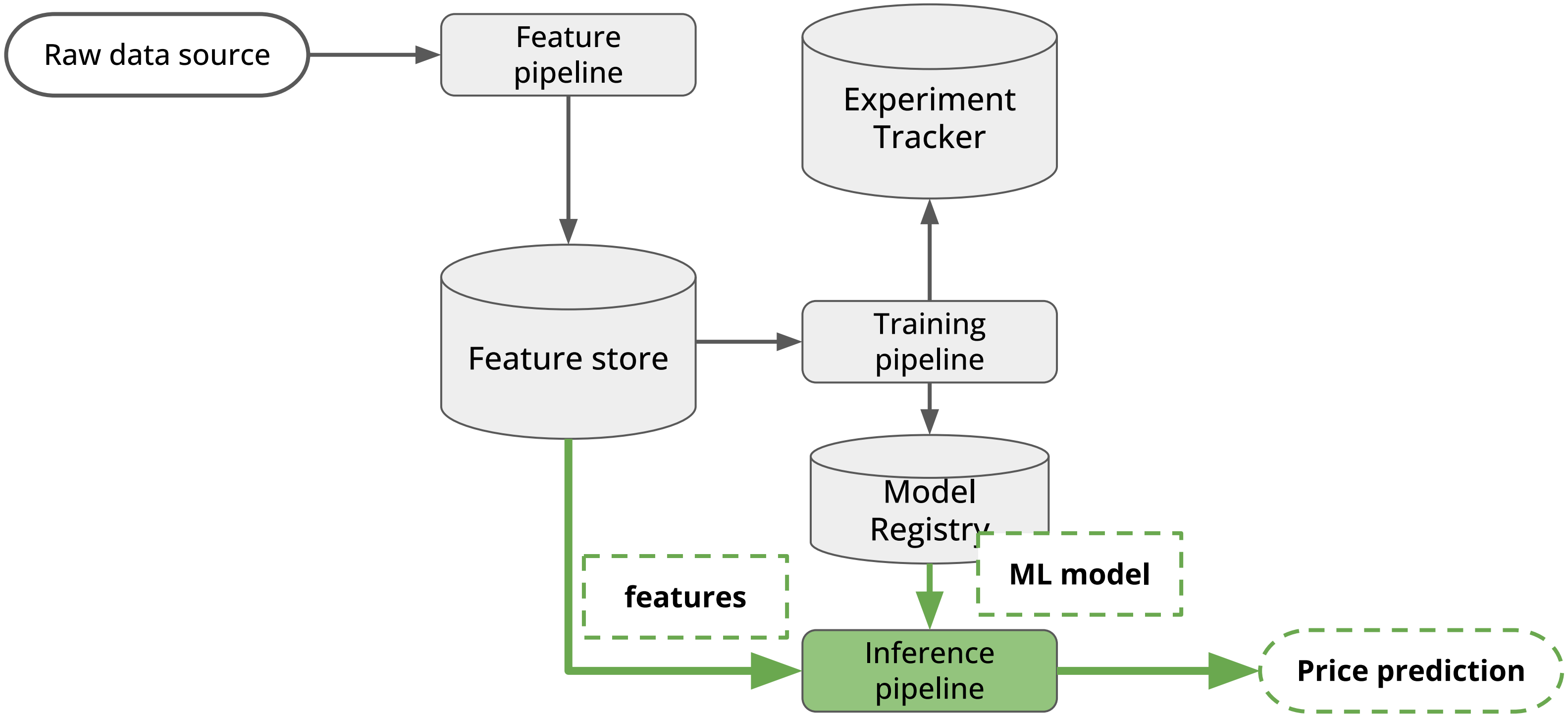
2. Training pipeline

From **training data** to **an ML model + metadata**

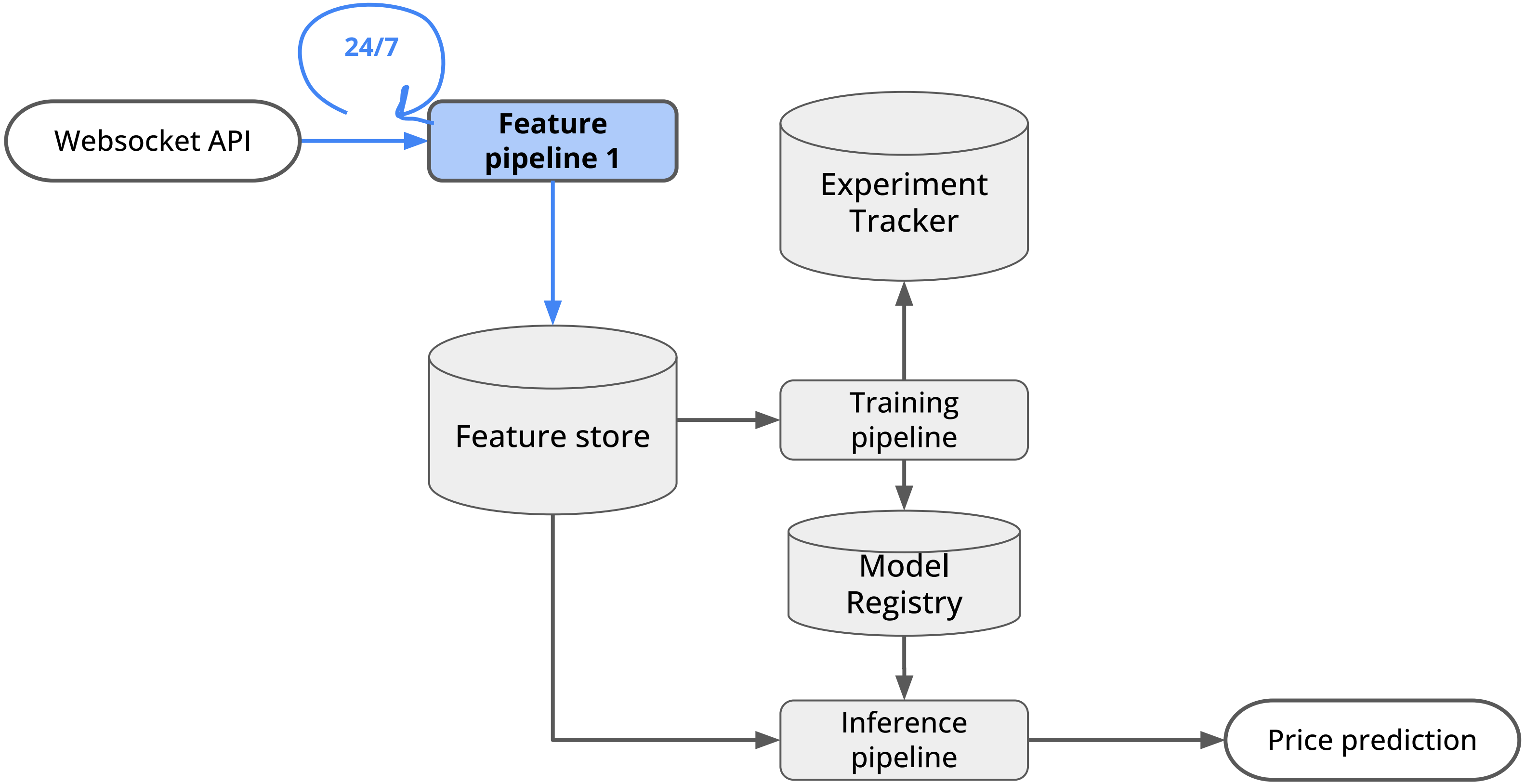


3. Inference pipeline

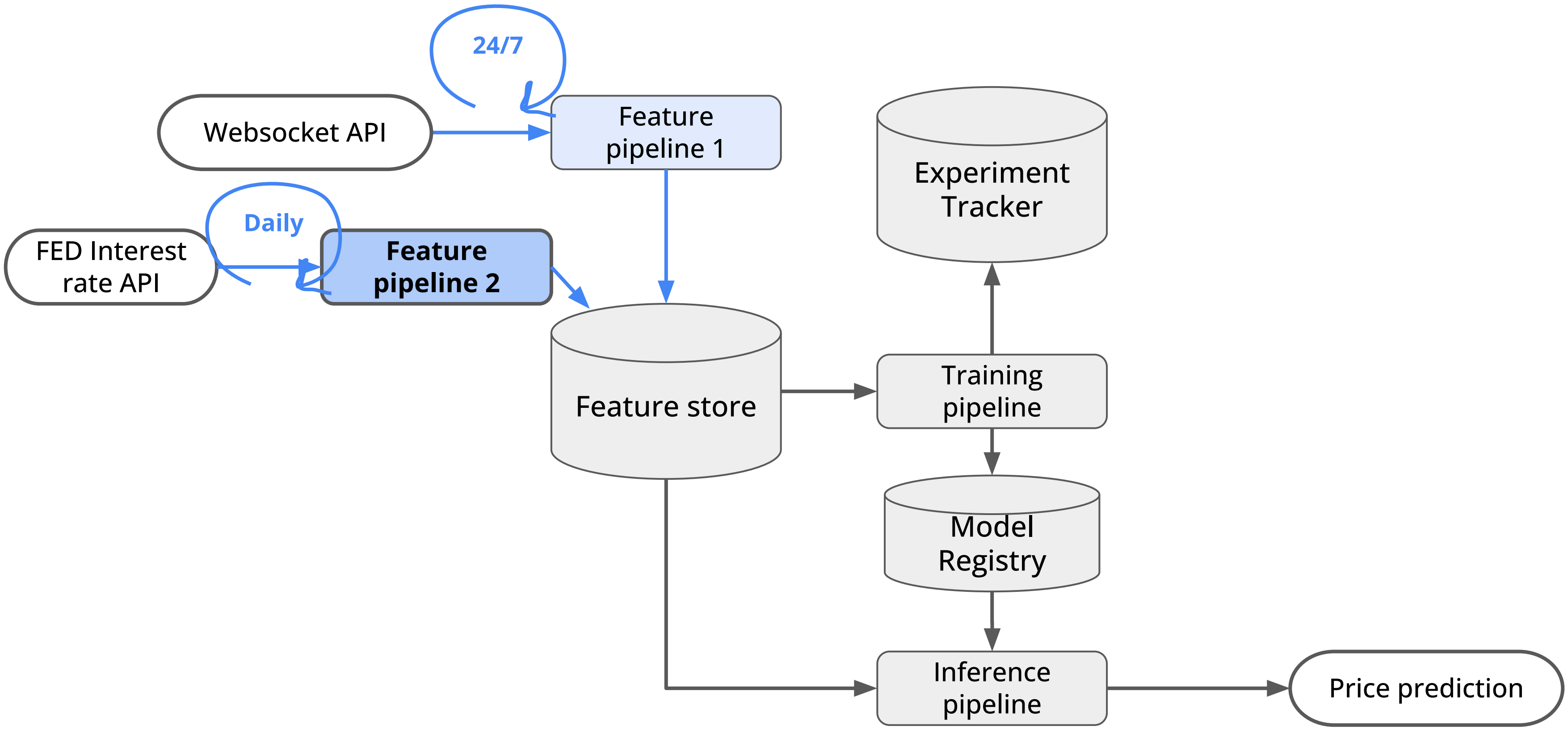
From an **ML model** and fresh **features** to **predictions**



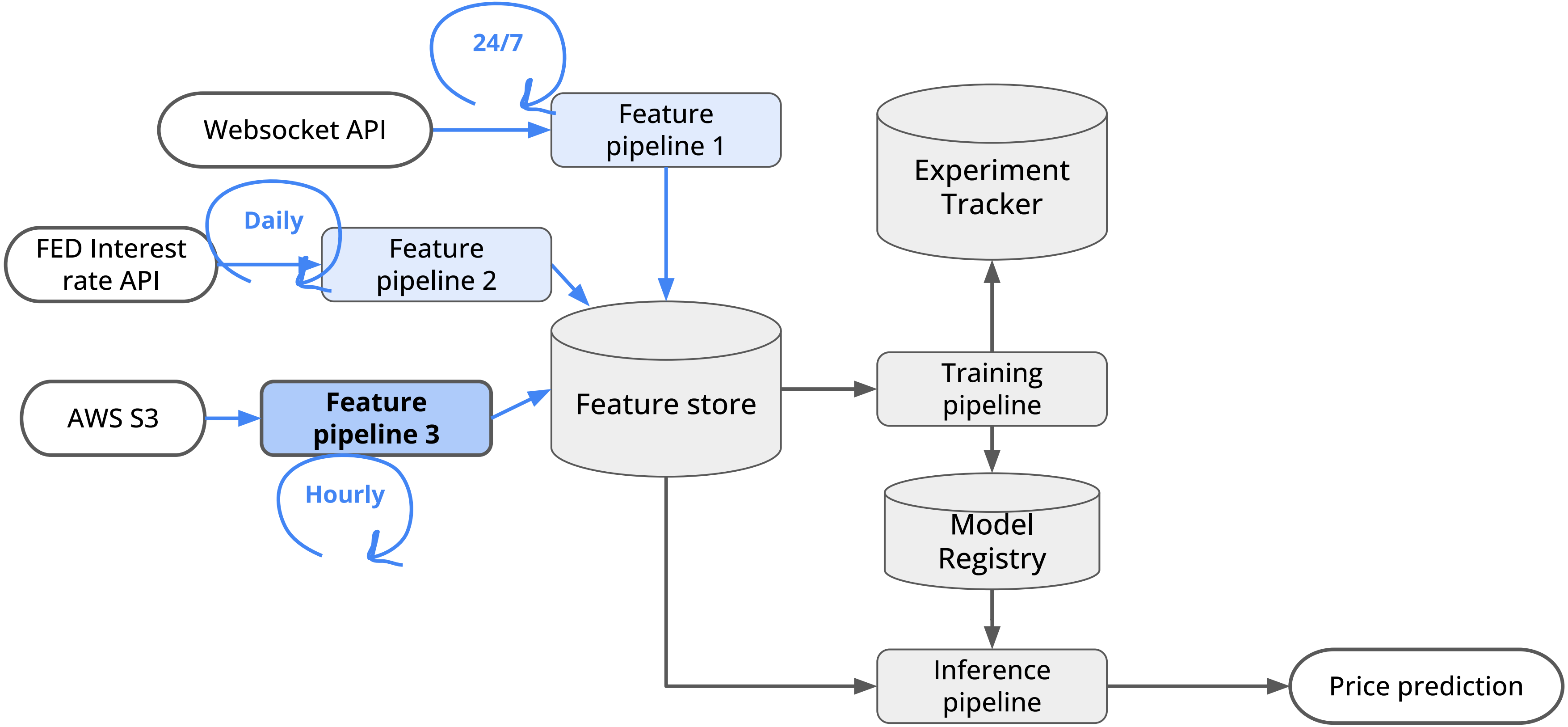
1 real-time feature pipeline



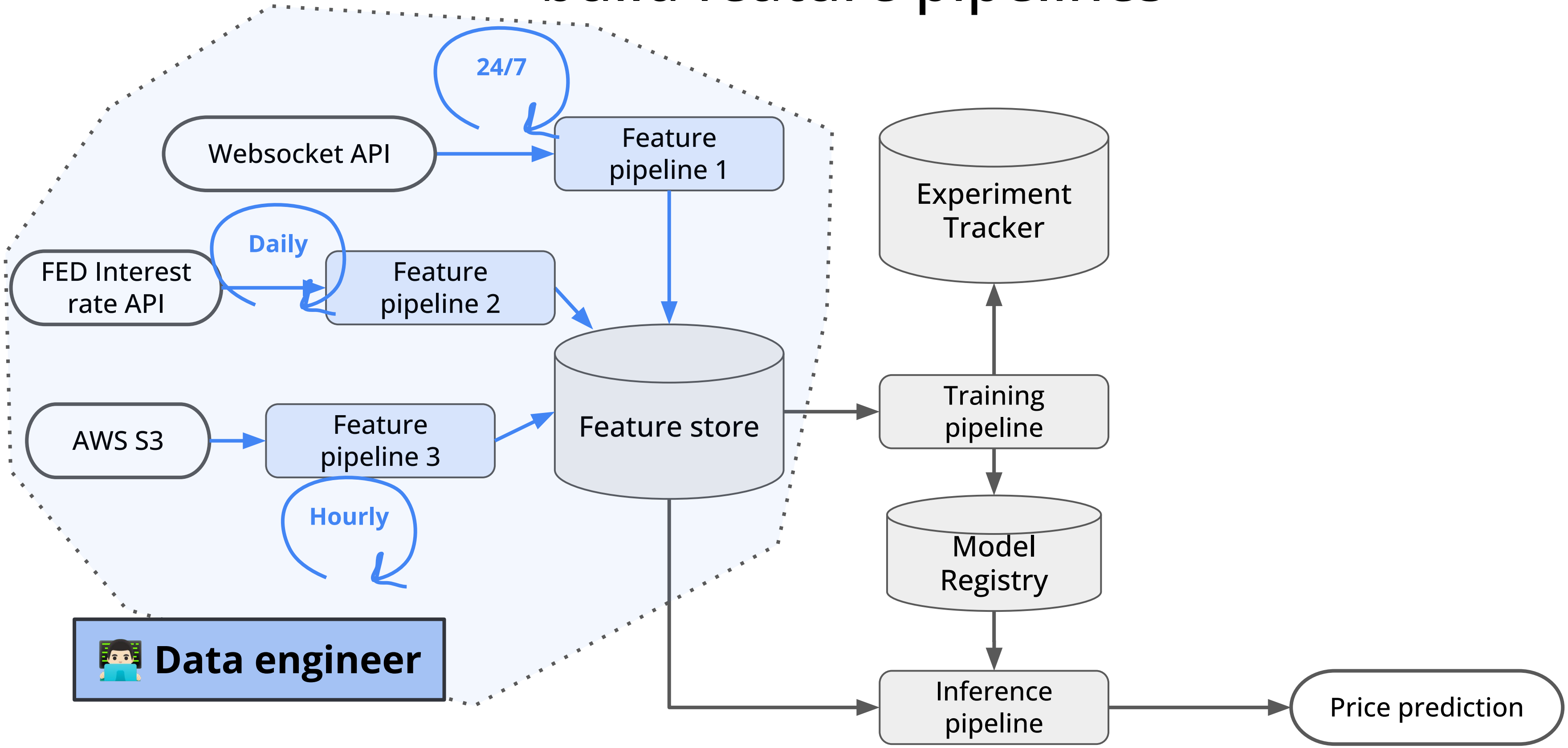
1 real-time + **1 daily** feature pipeline



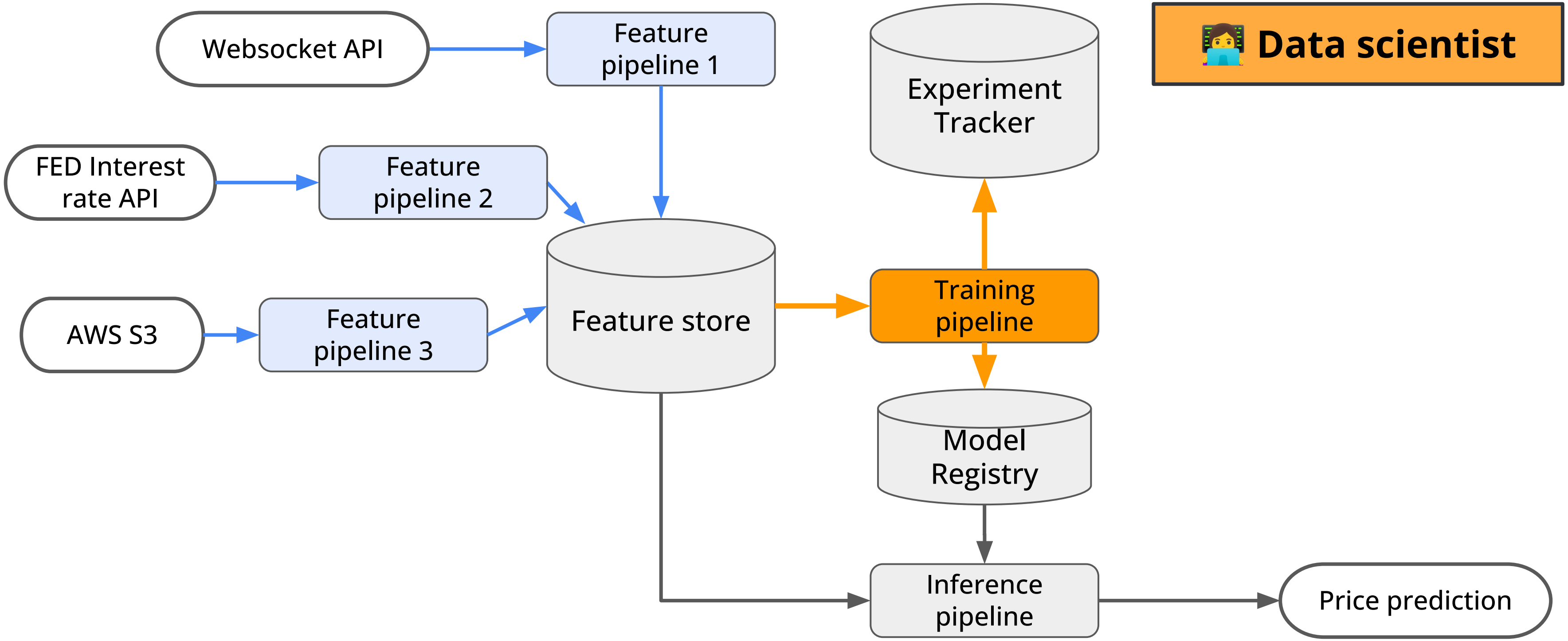
1 real-time + 1 daily + 1 hourly feature pipeline



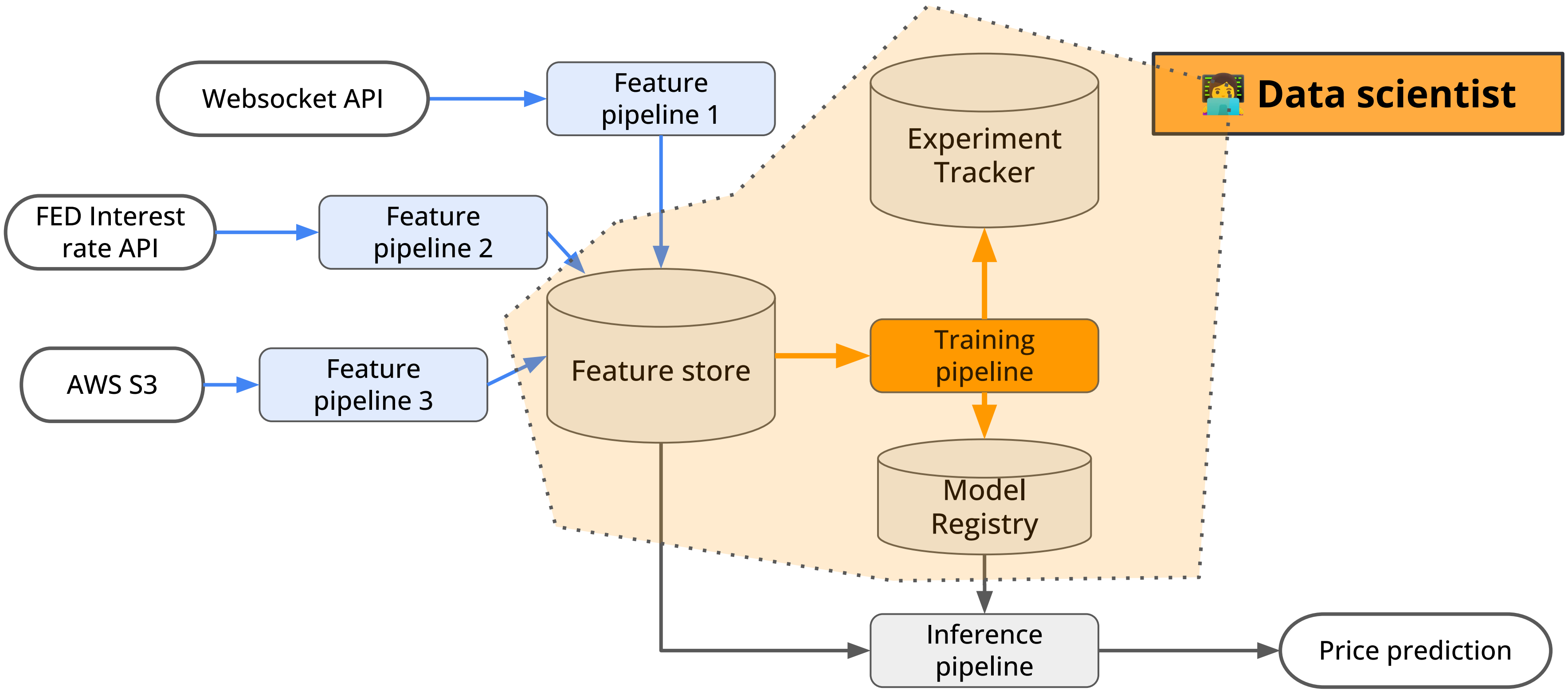
Data engineers
build feature pipelines



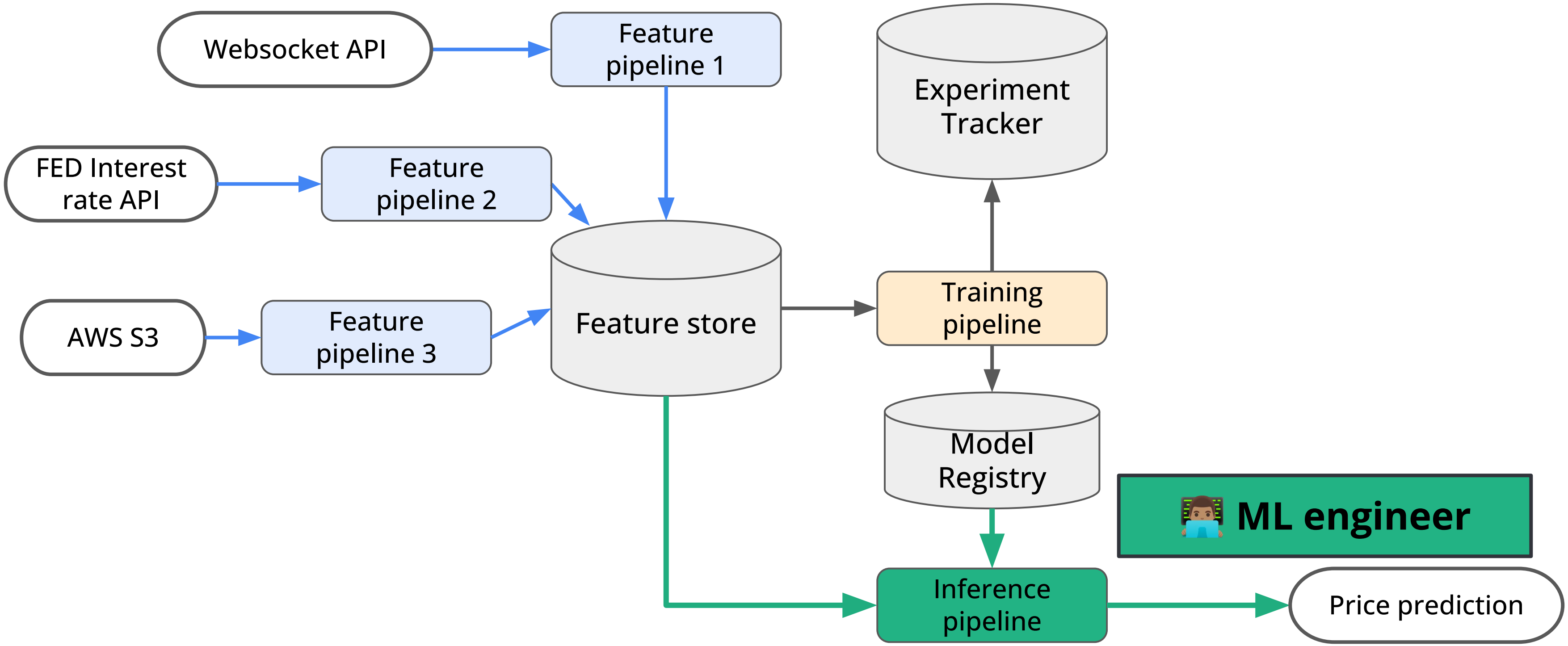
Data scientists
train ML models



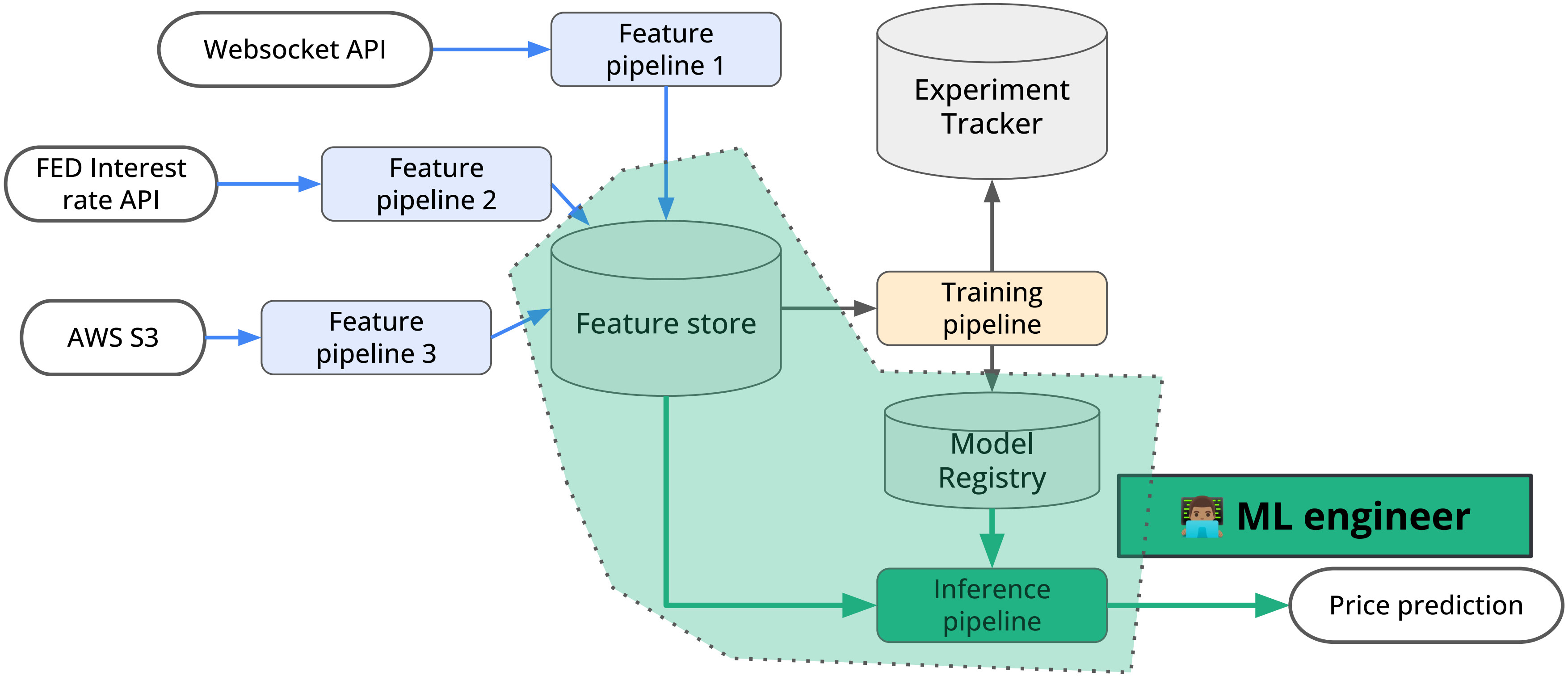
Data scientists train ML models



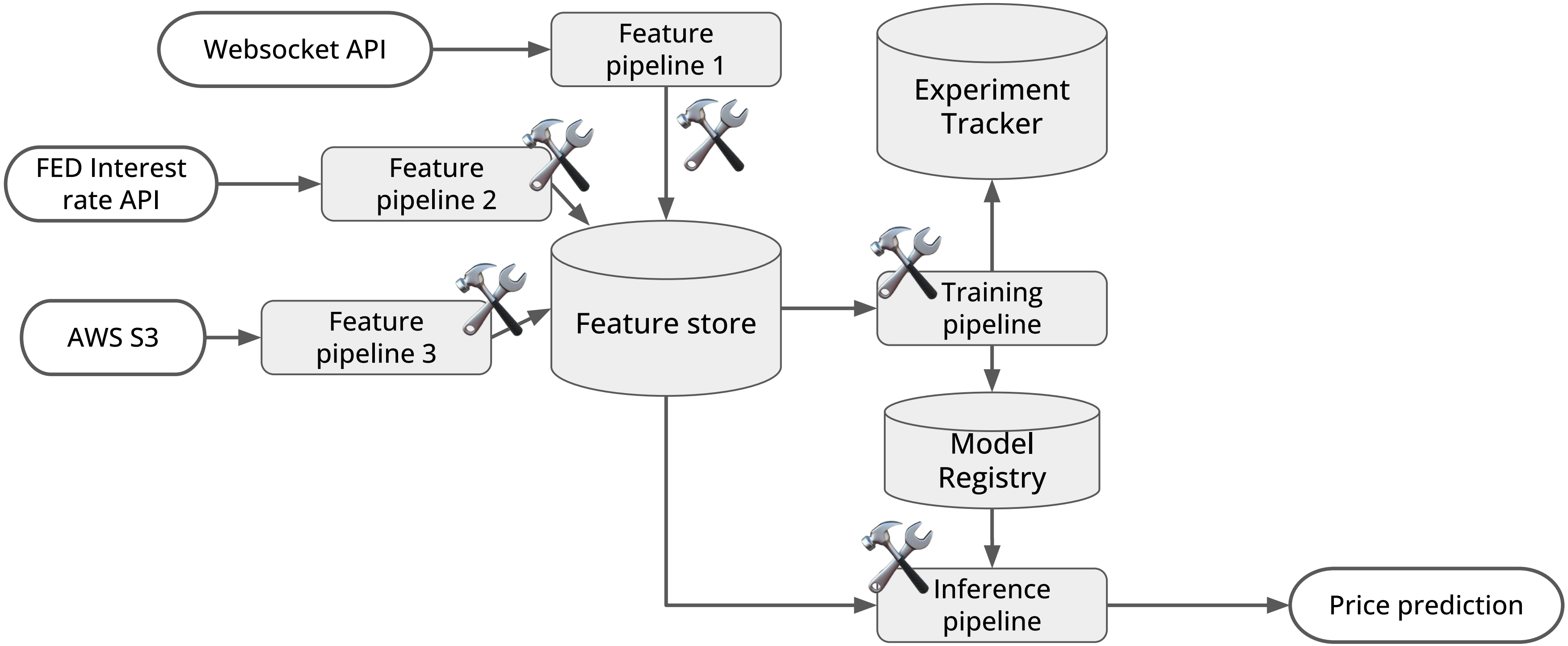
ML engineers
deploy these models



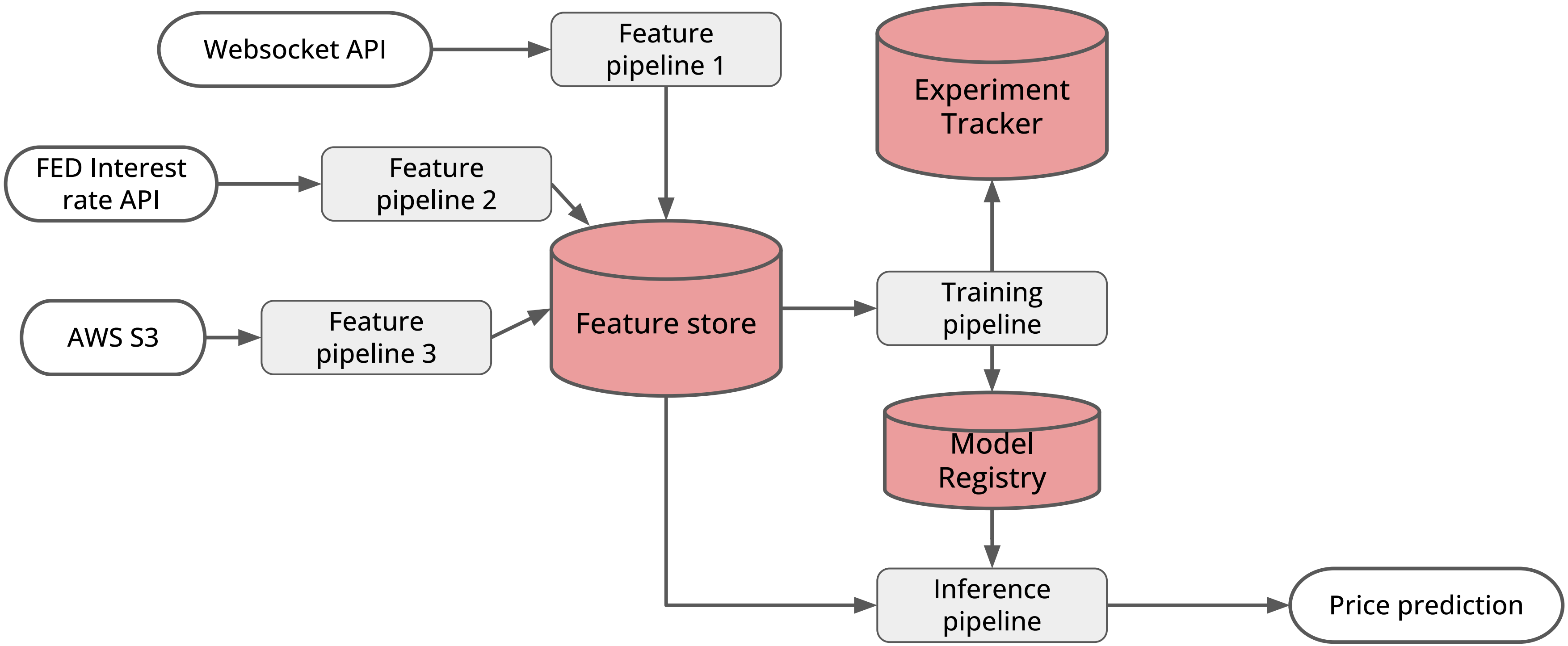
ML engineers
deploy these models



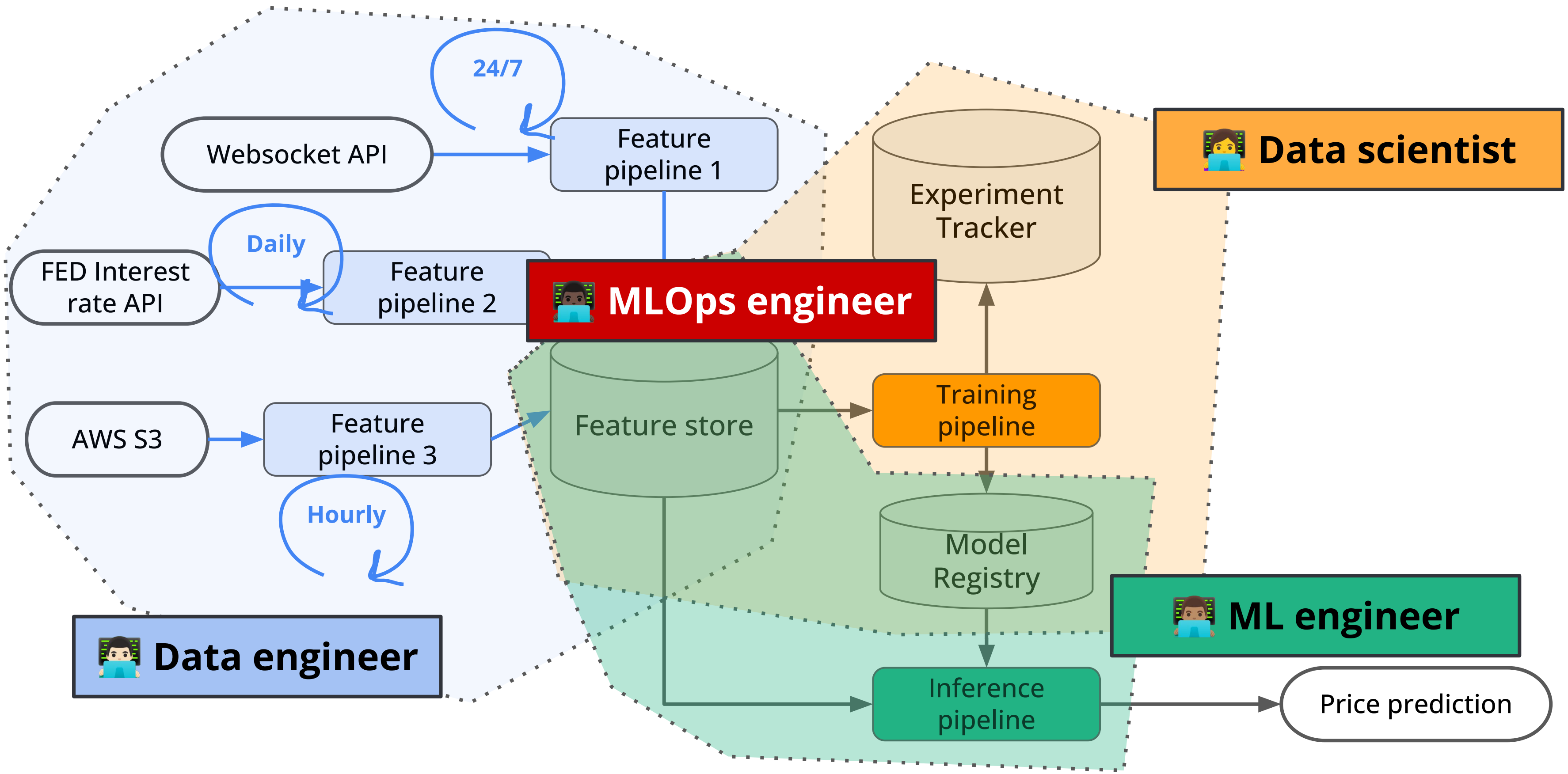
ML Ops engineers
Build tools for the rest of the team



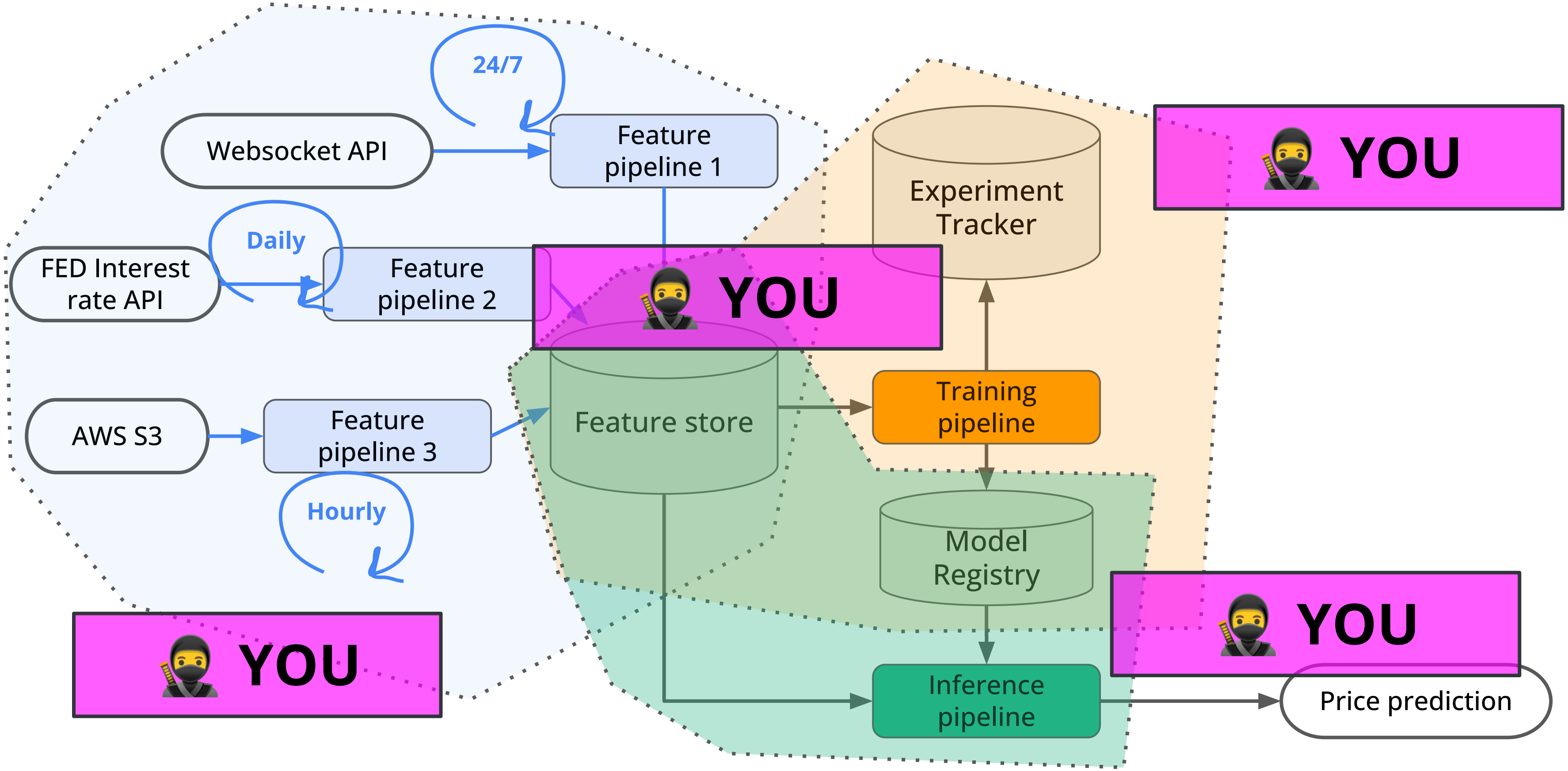
ML Ops engineers
Take care of infrastructure



The whole team



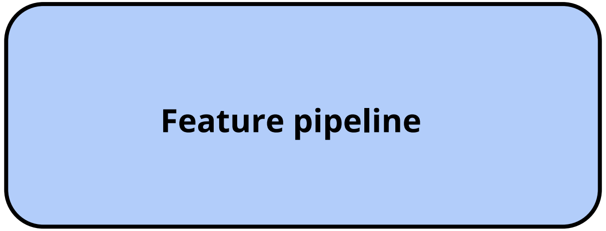
ML ninjas do everything 🥷



But HOW?

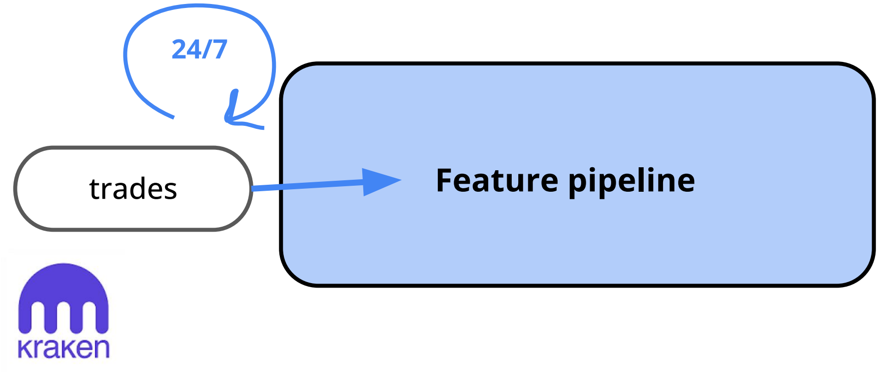
System Design

Feature pipeline

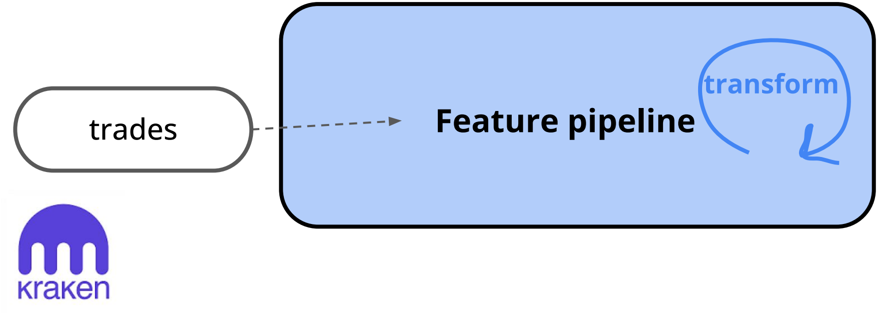


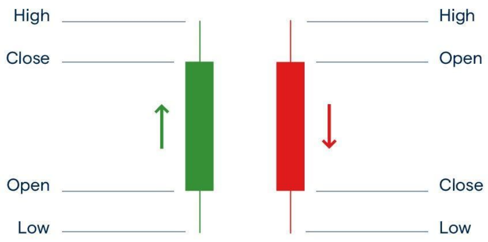
Real-time Feature pipeline

Reads stream of trades from Kraken Websocket

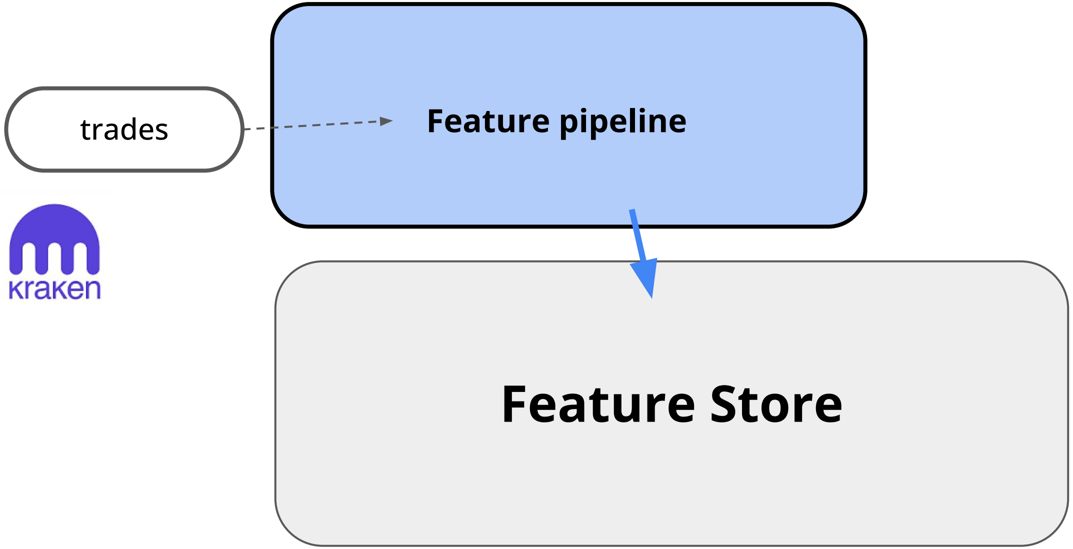


Real-time Feature pipeline
Transforms them into **1-minute OHLC candles**



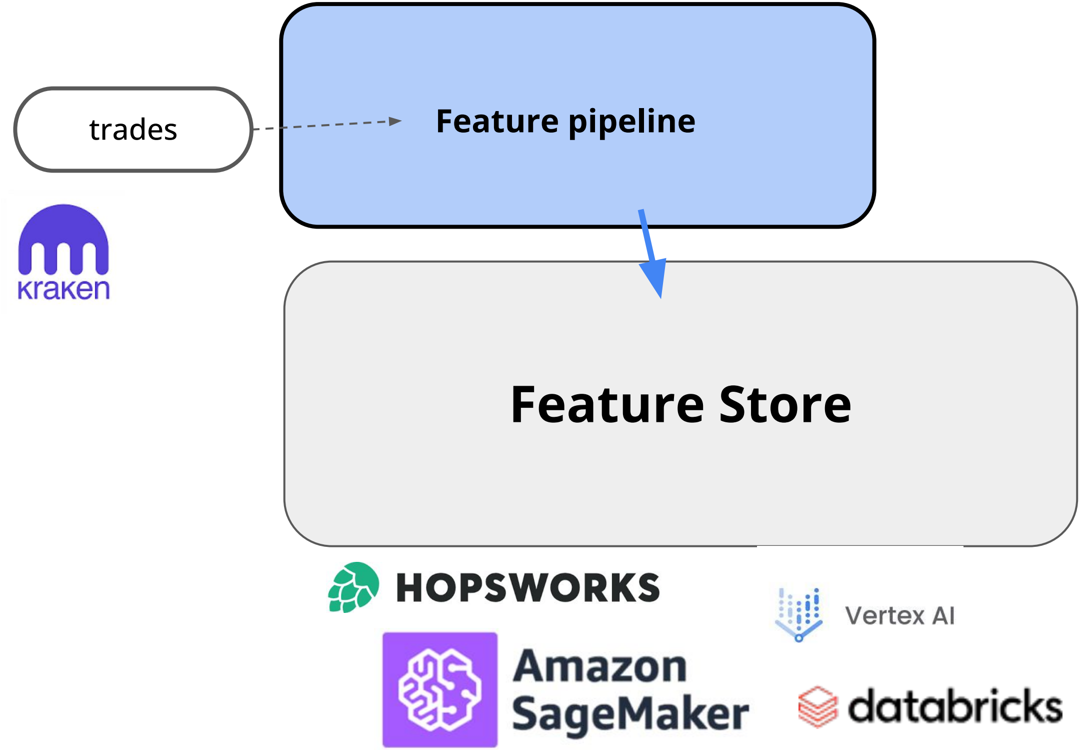


Real-time Feature pipeline
Save 1-minute OHLC candles to **Feature Store**



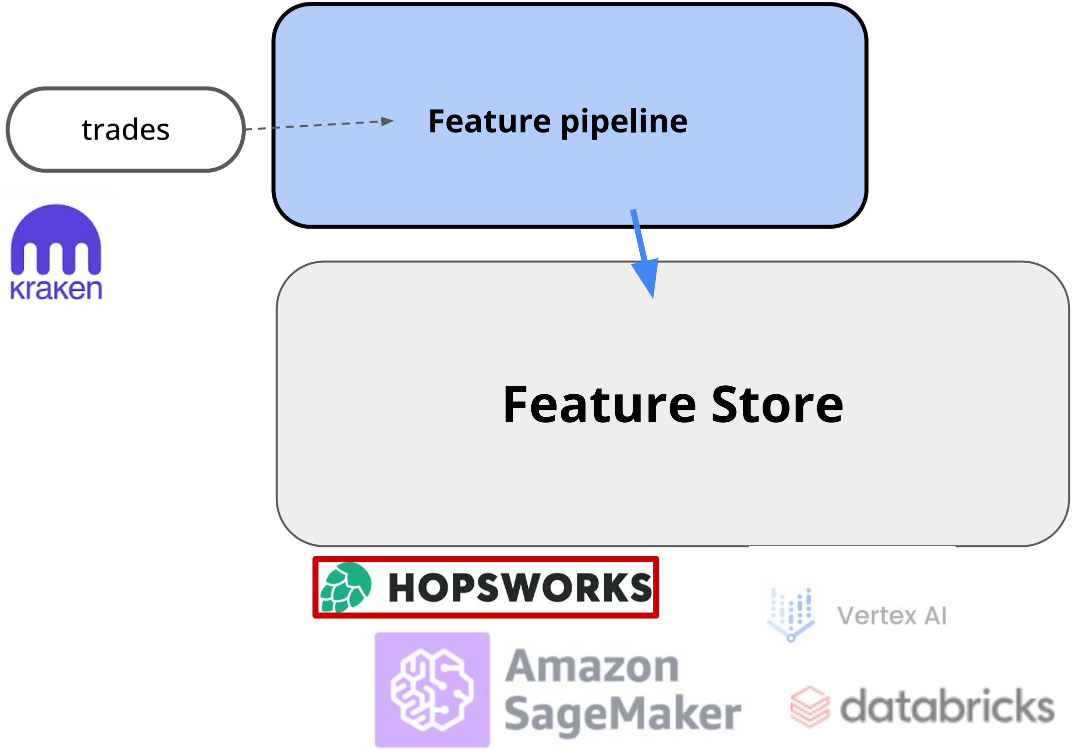
Real-time Feature pipeline

Save 1-minute OHLC candles to **Feature Store**

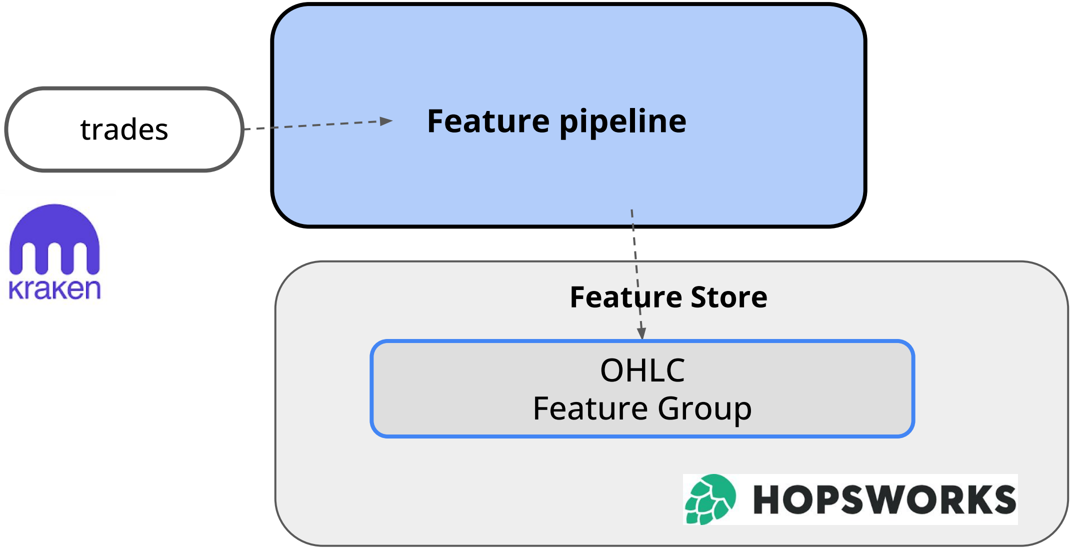


Real-time Feature pipeline

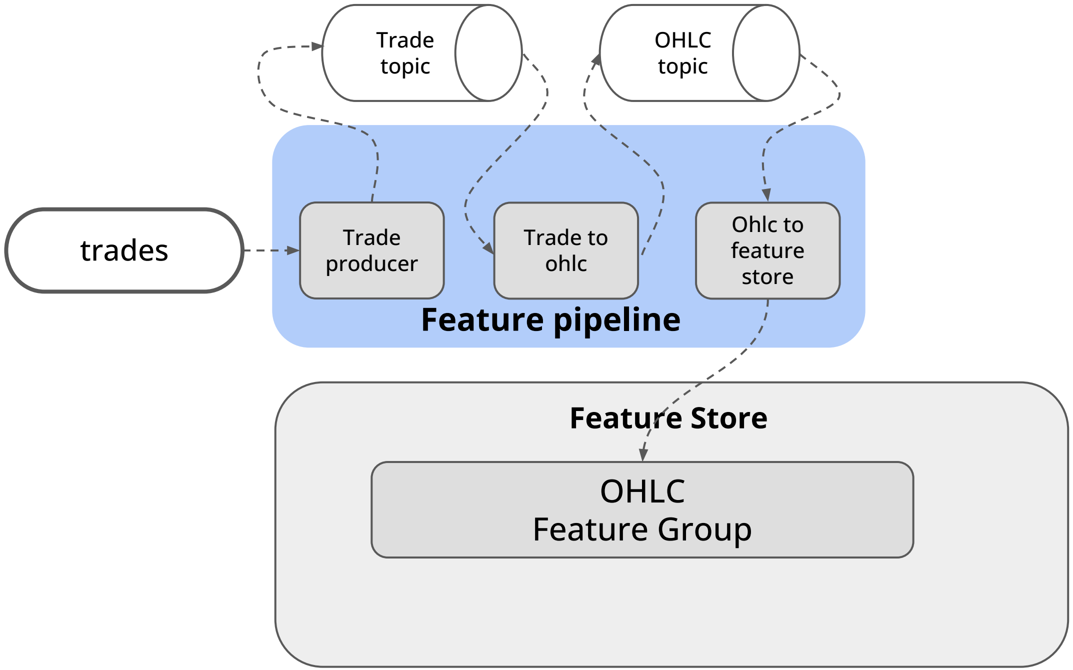
Save 1-minute OHLC candles to **Feature Store**



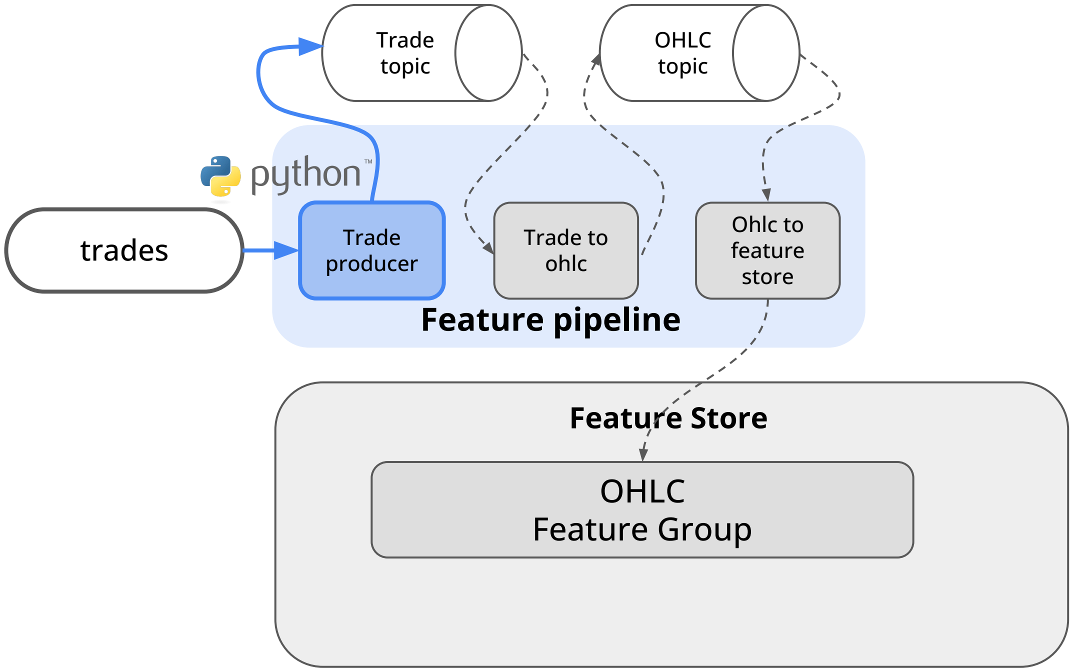
Real-time Feature pipeline
More precisely, into a Feature Group



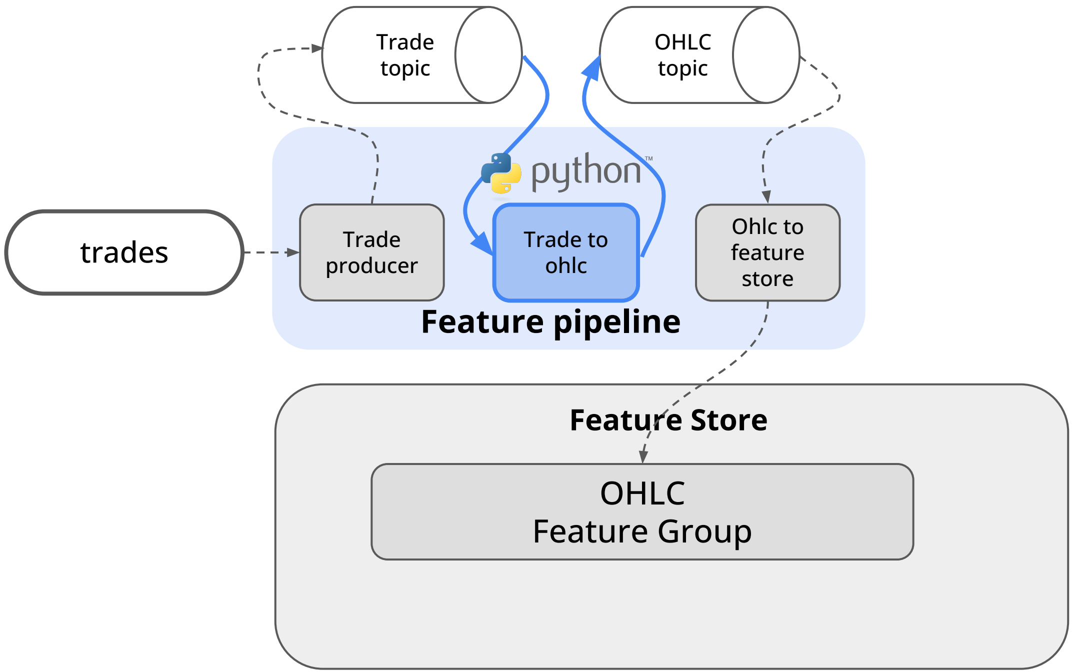
Modular Real-time Feature pipeline



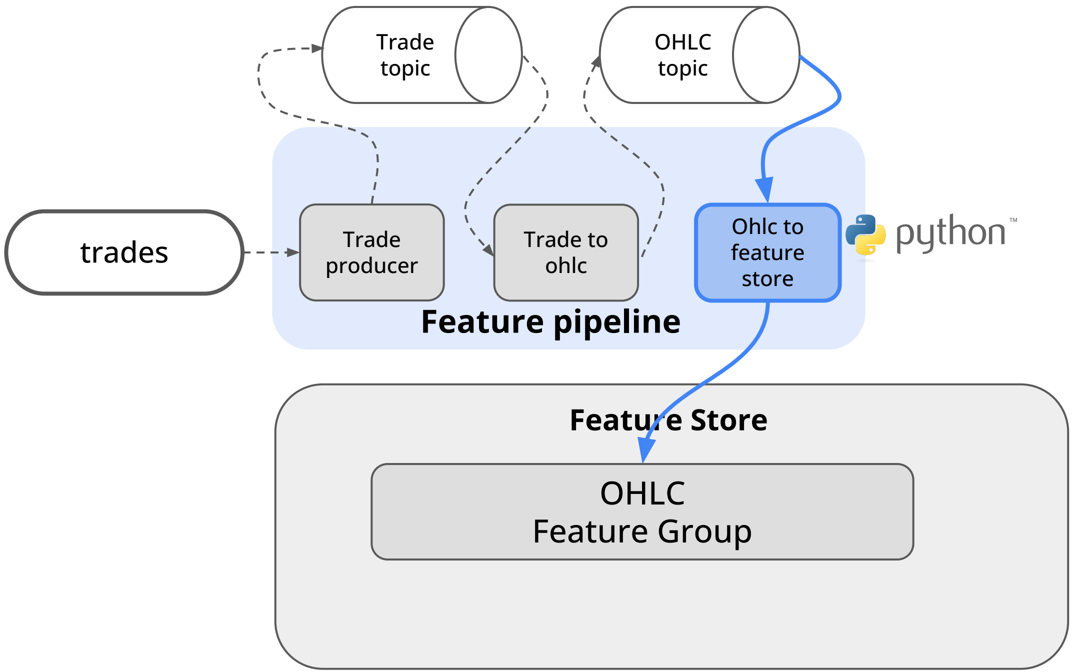
Ingests raw trades



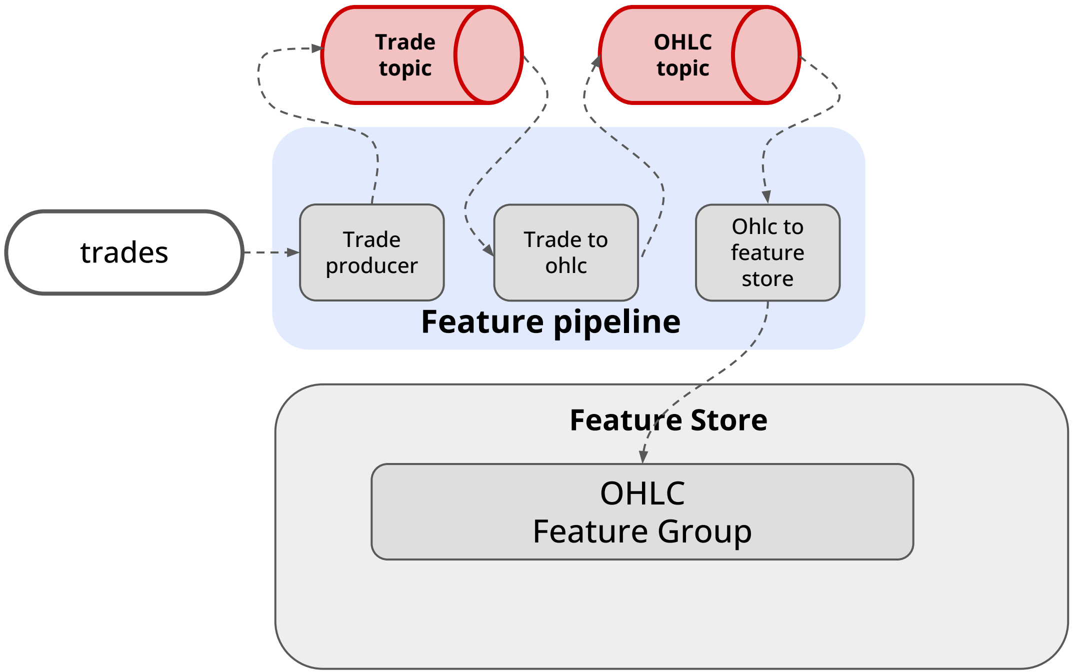
Transform raw trades into OHLC candles



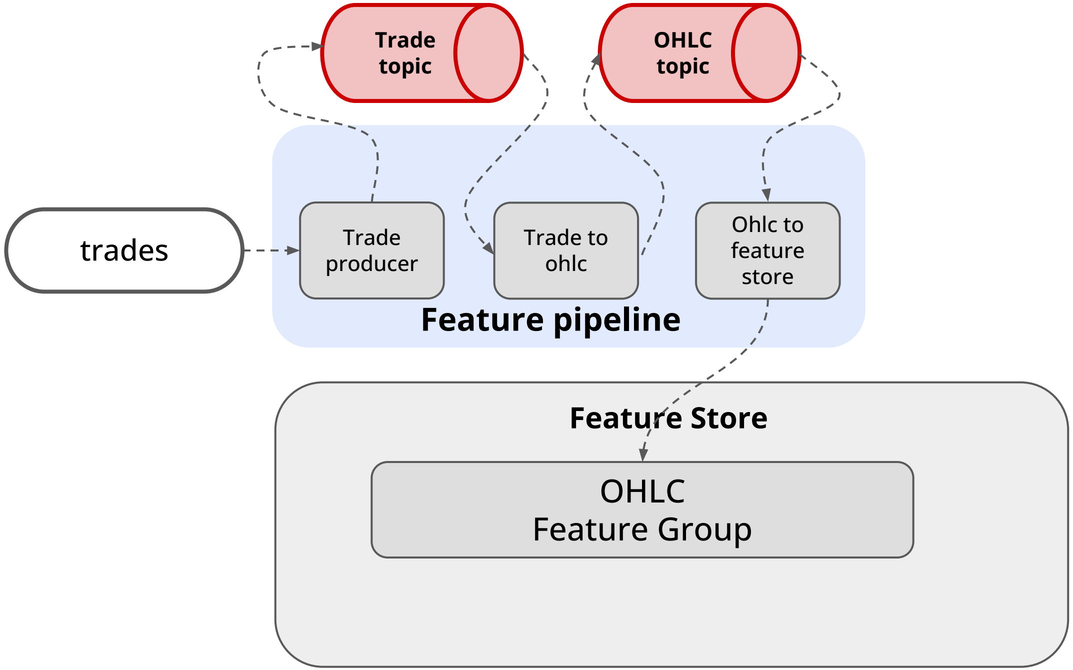
Pushed OHLC candles to the Feature Store



Streaming data platform for communication

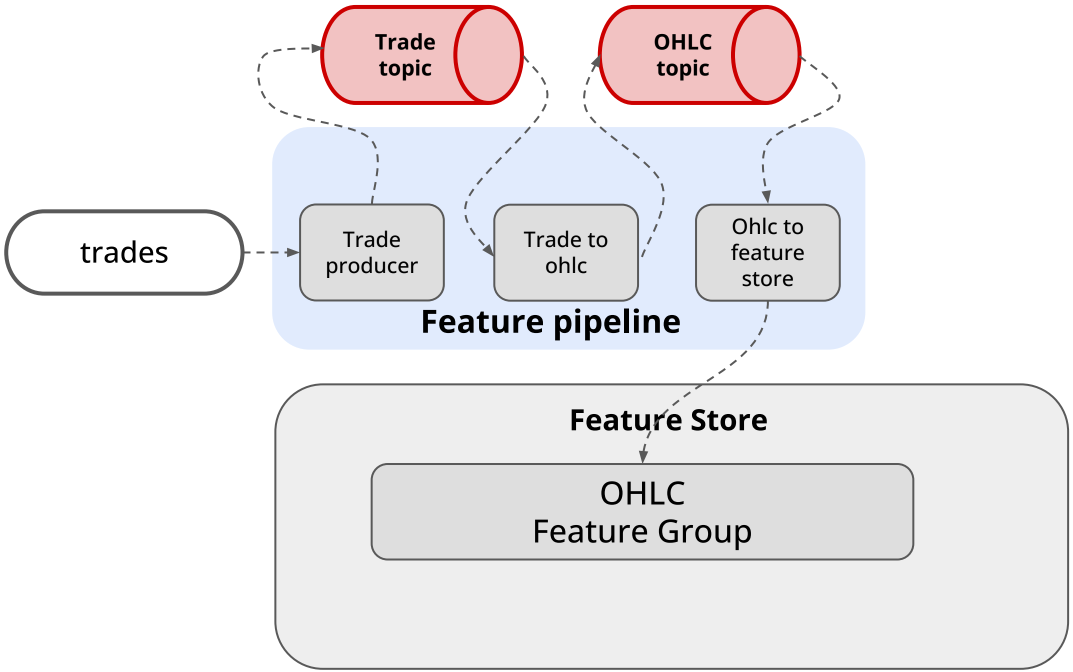


Streaming data platform for communication

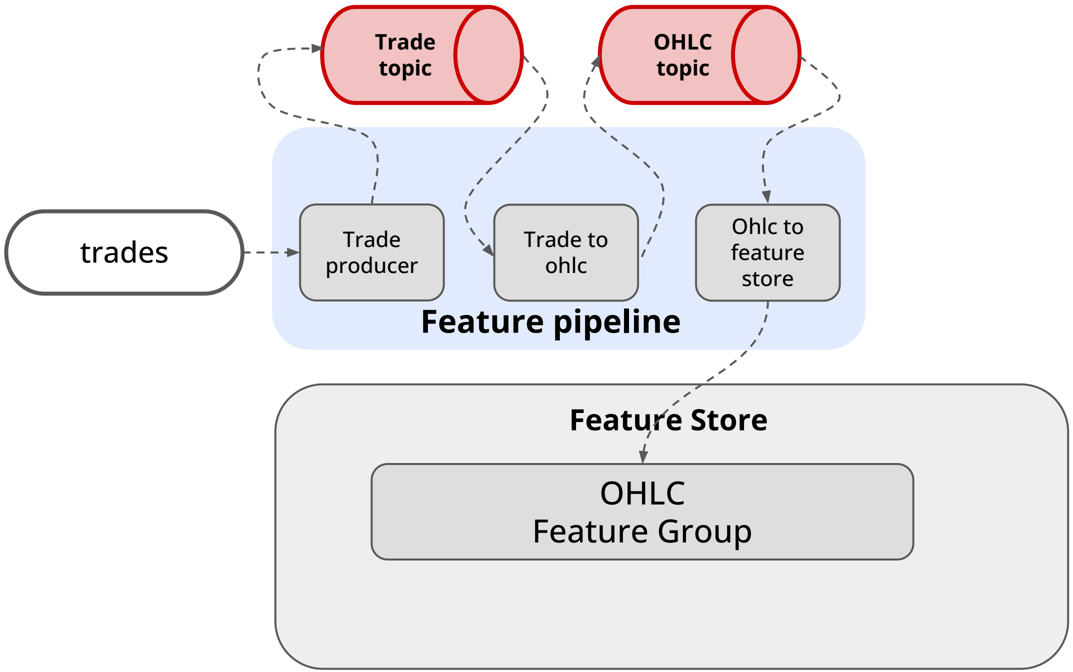


kafka

Streaming data platform for communication



Streaming data platform for communication

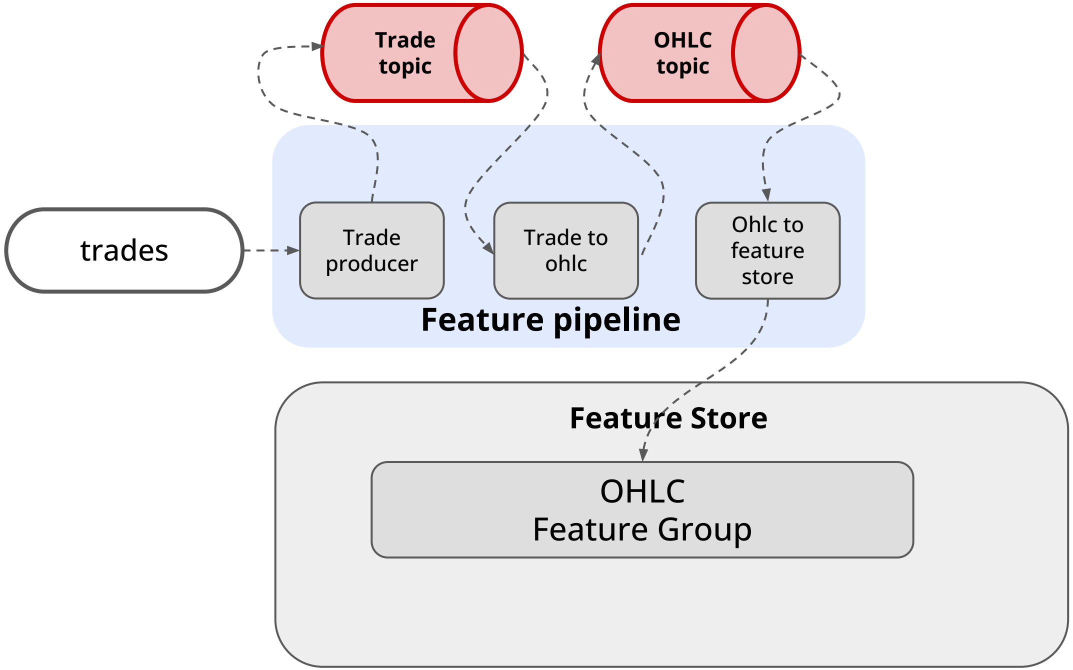


kafka

Redpanda

Google Cloud Pub/Sub

Streaming data platform for communication

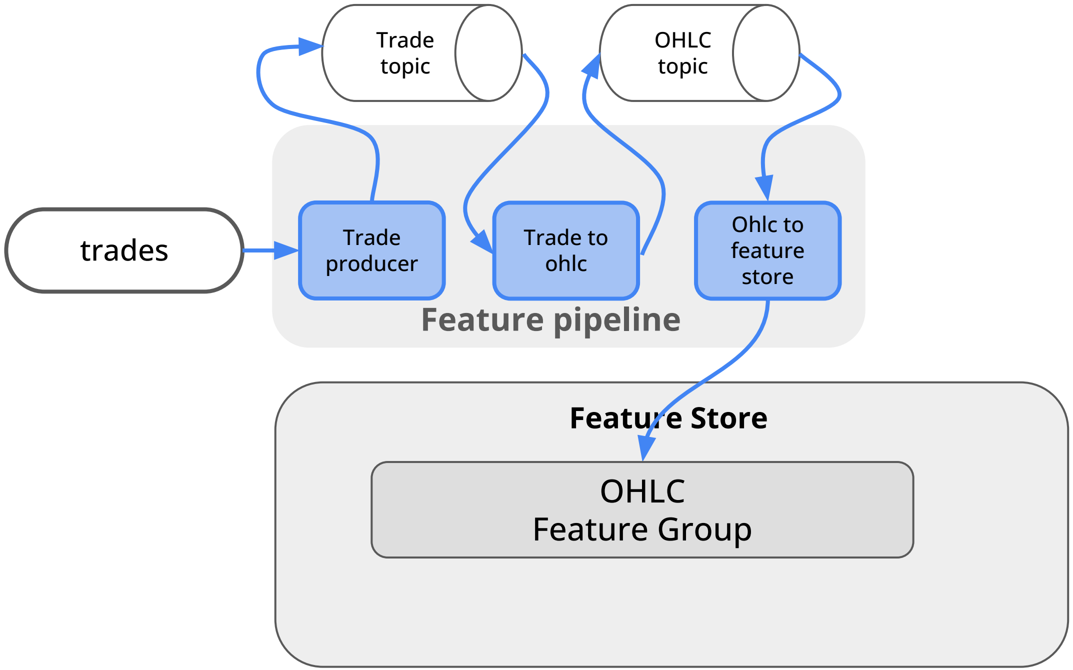


kafka

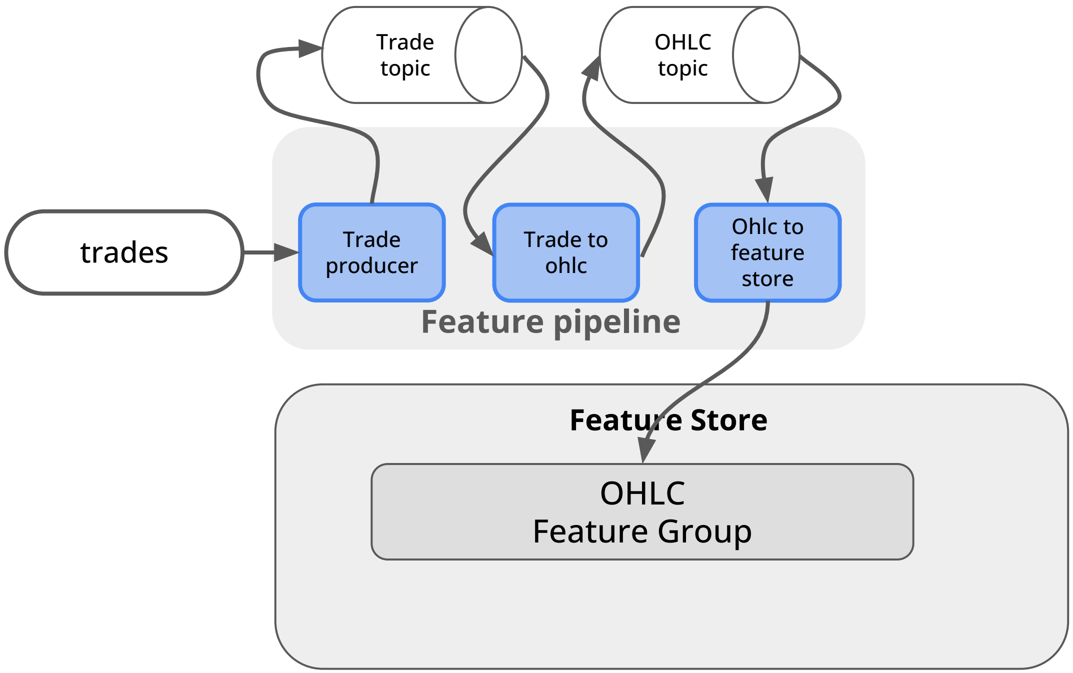
Google Cloud
Pub/Sub

Redpanda

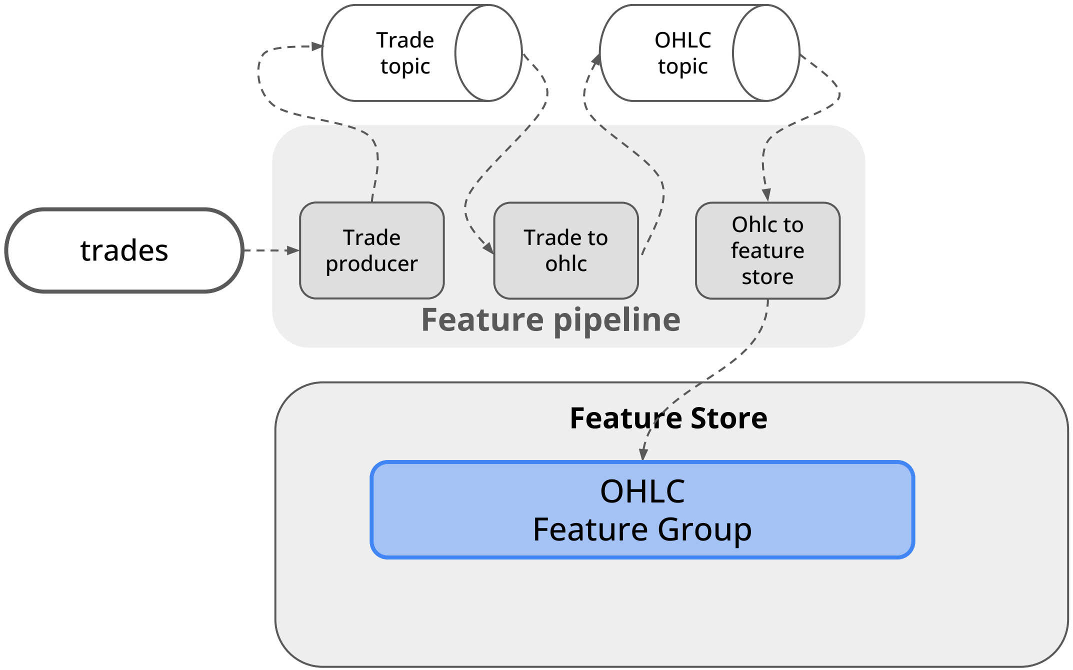
Backfill historical features for last 90 days



Backfill historical features for last 90 days

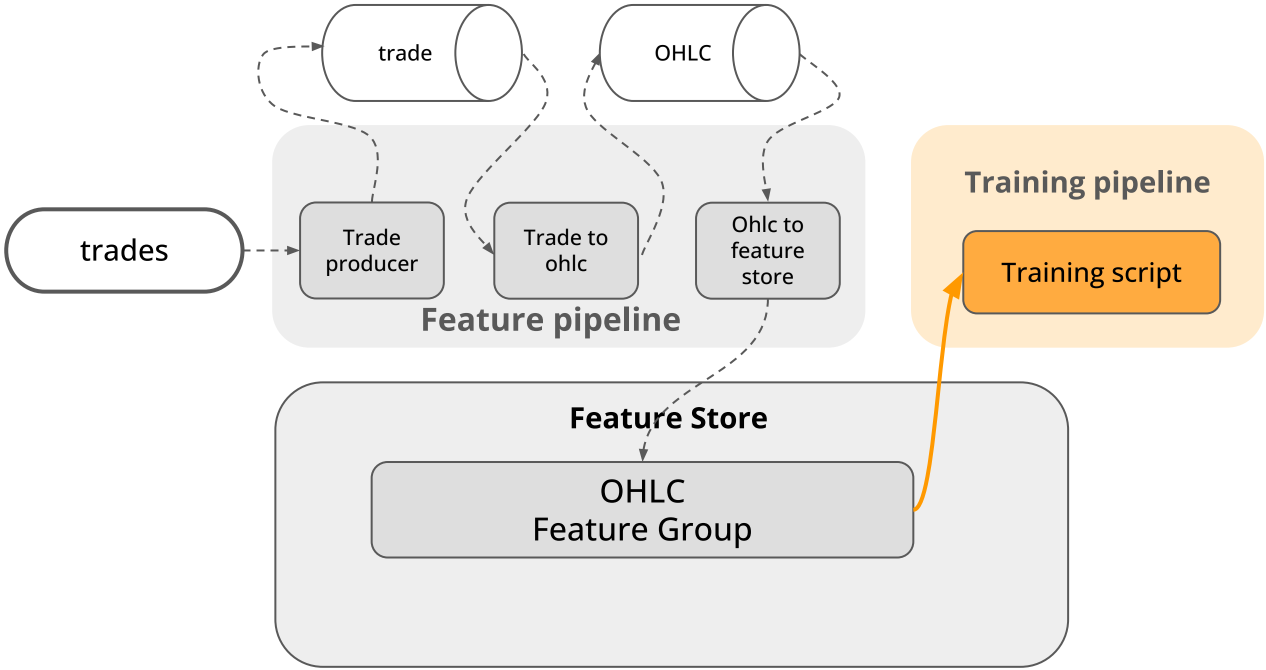


This is our training data



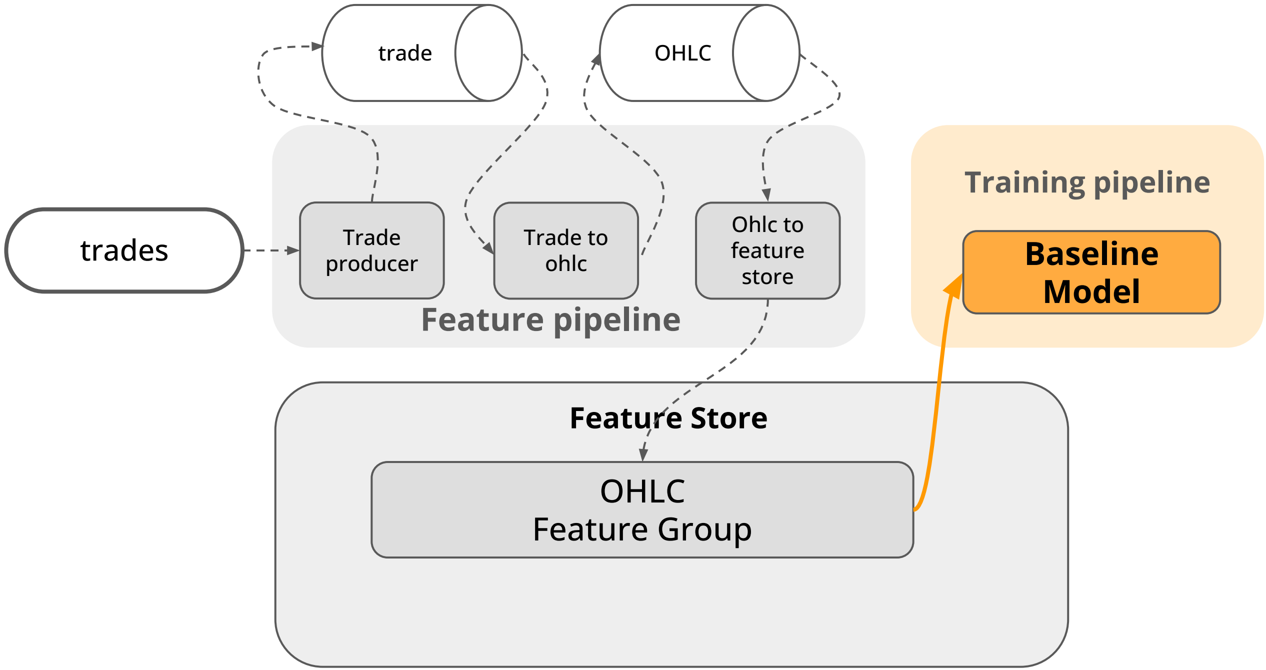
Build ML model from this training data

to predict price 5 minutes into the future



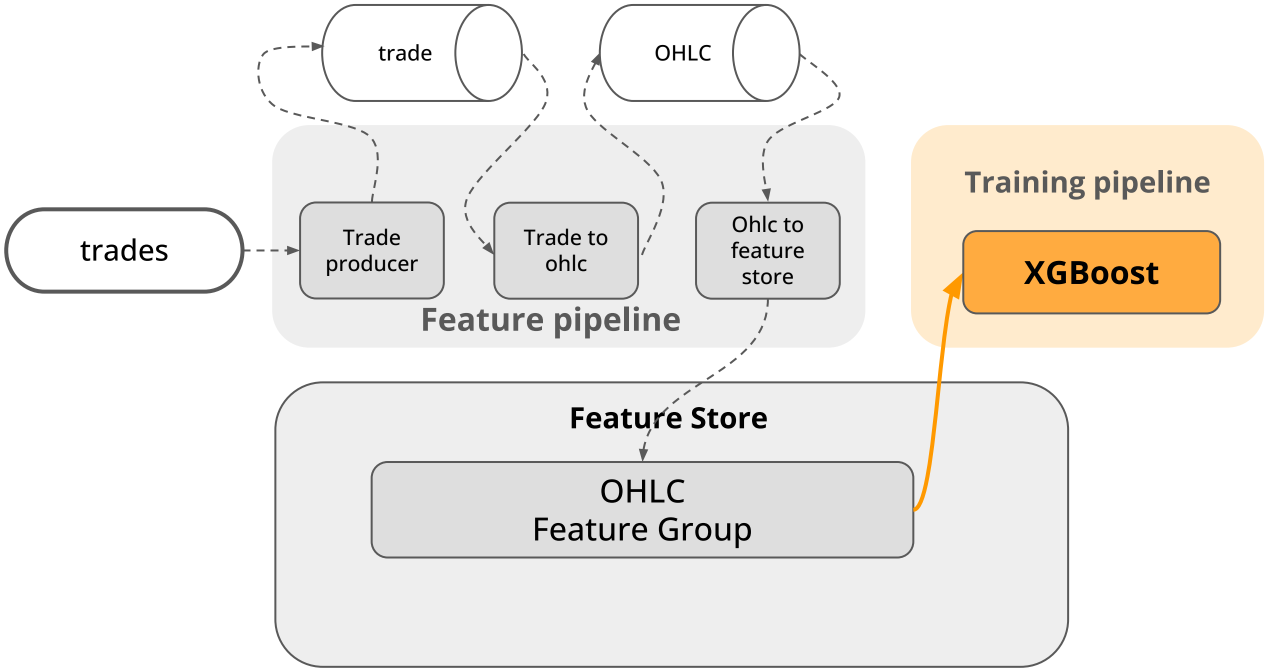
Model baseline

to predict price 5 minutes into the future



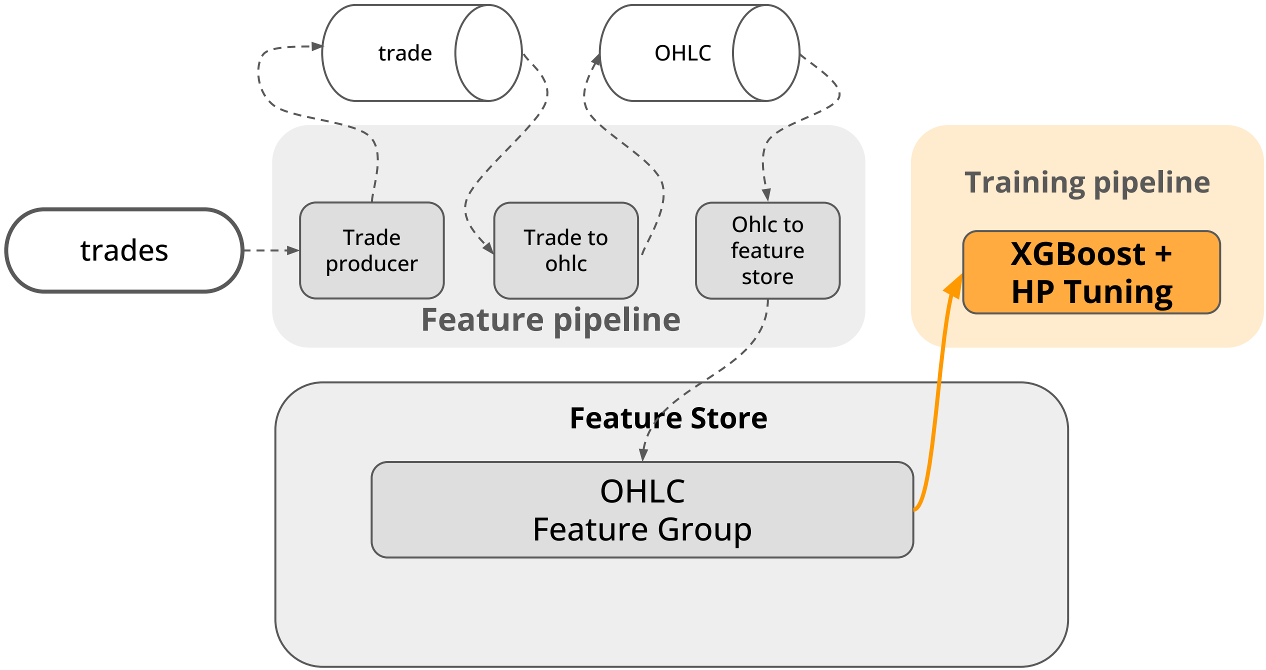
XGBoost

to predict price 5 minutes into the future



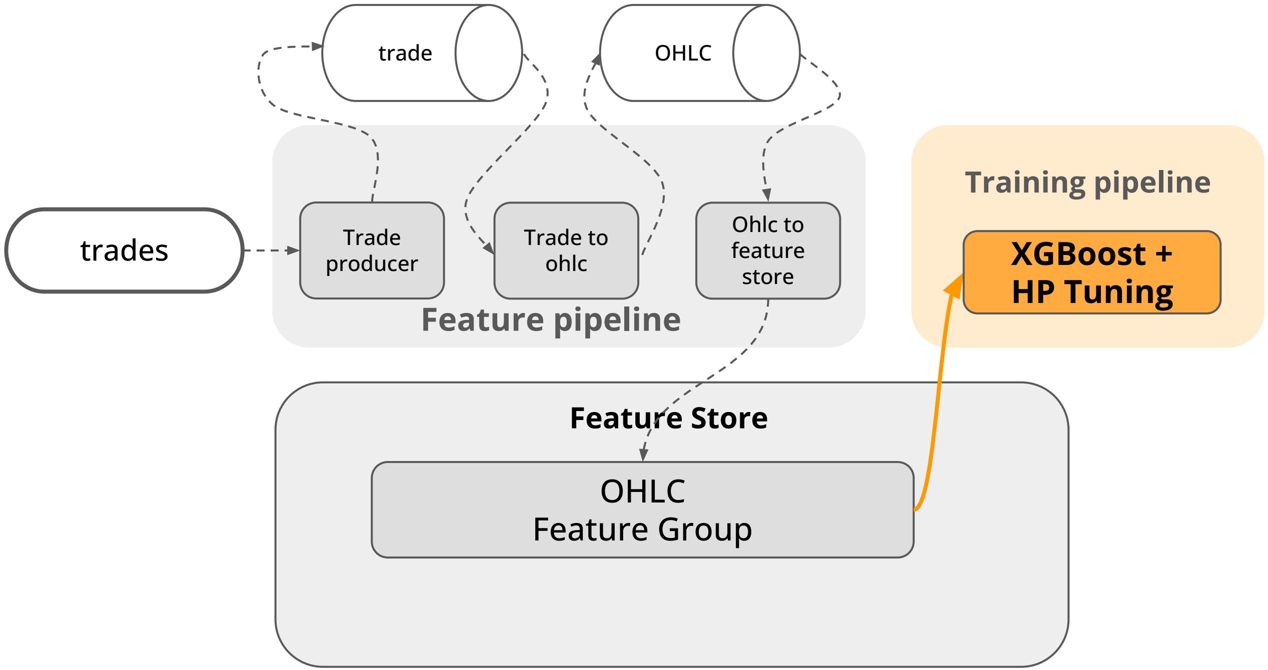
XGBoost + Hyperparameter Tuning

to predict price 5 minutes into the future

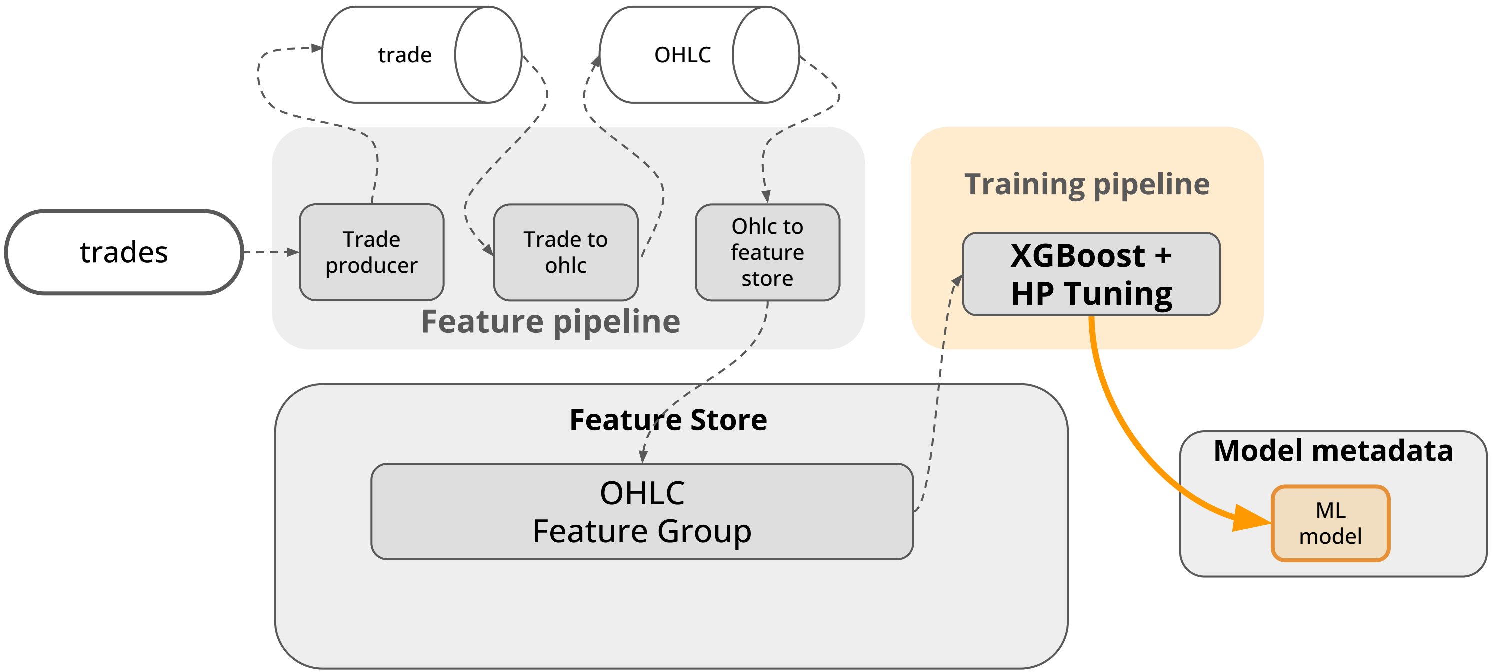


XGBoost + Hyperparameter Tuning

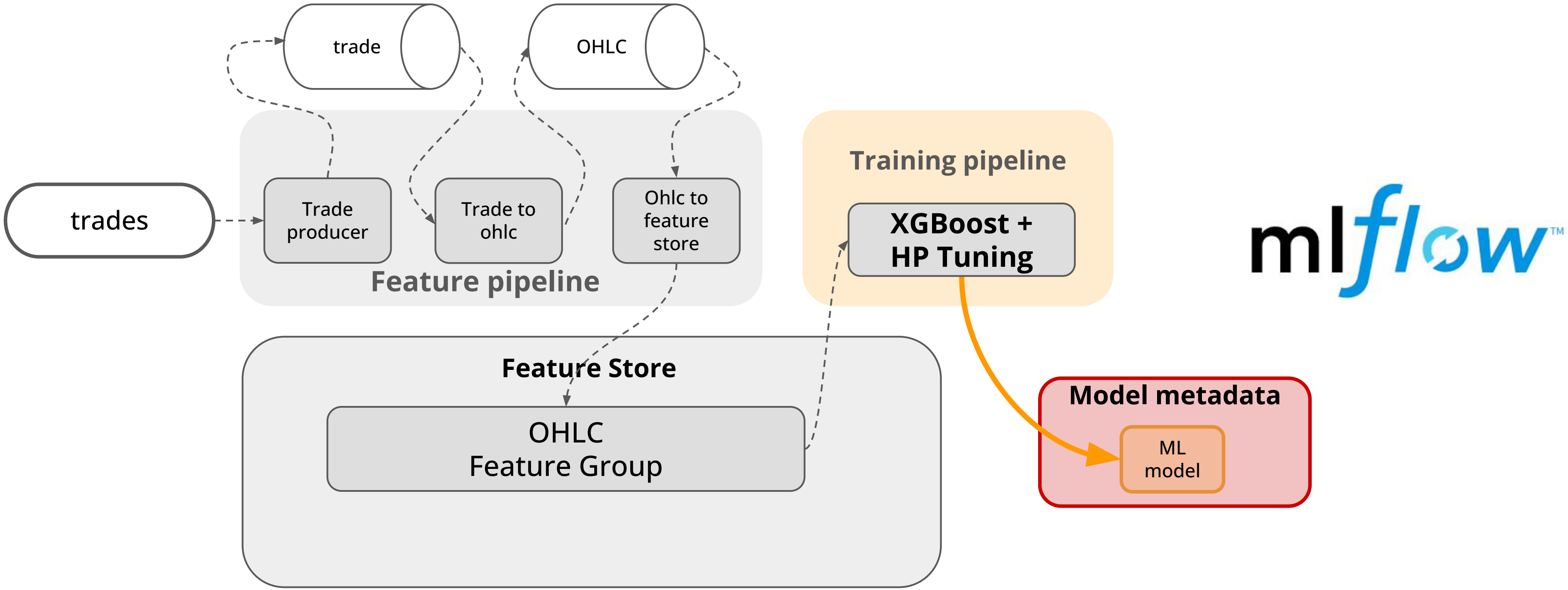
to predict price 5 minutes into the future



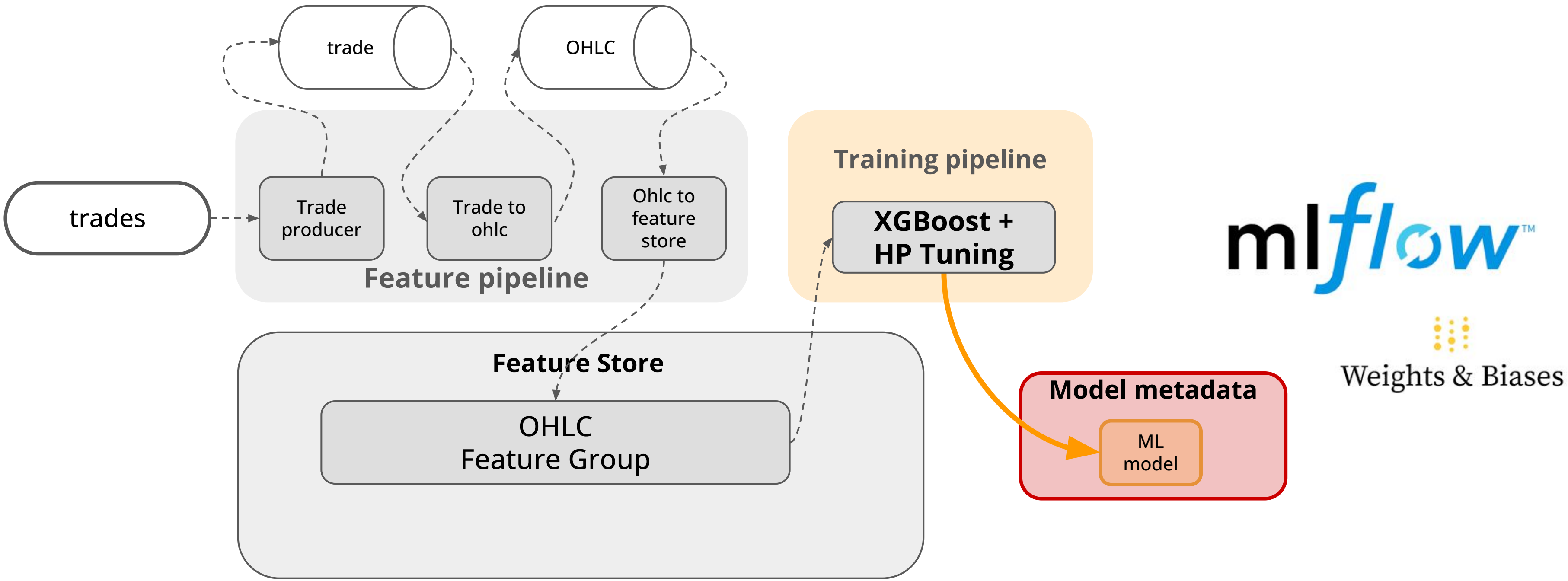
Push model to Model Registry



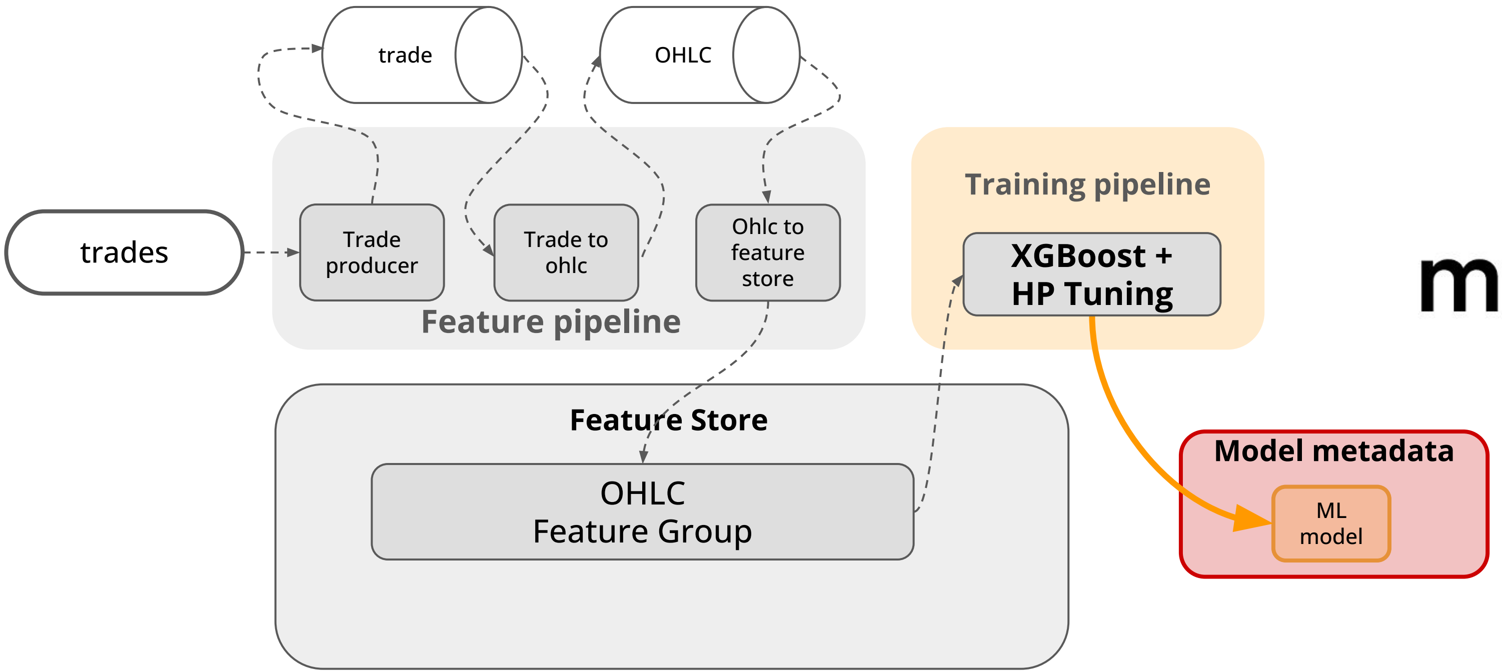
Push model to Model Registry



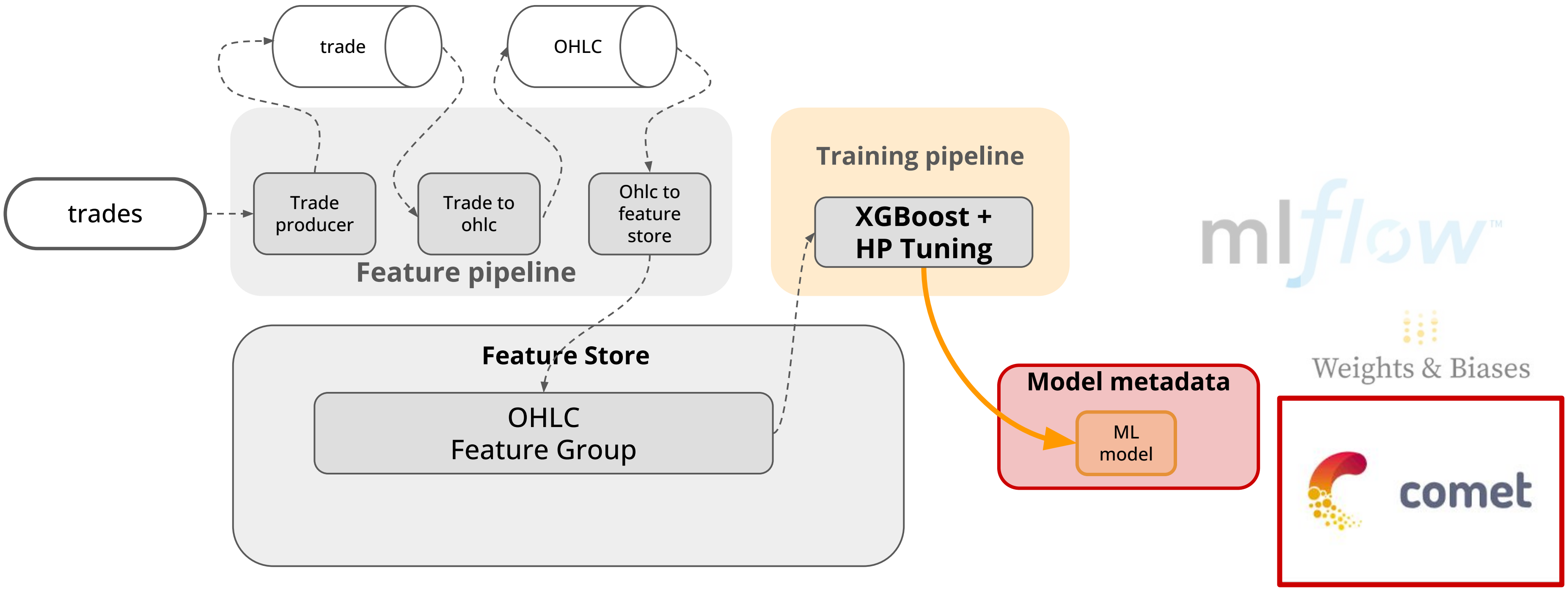
Push model to Model Registry



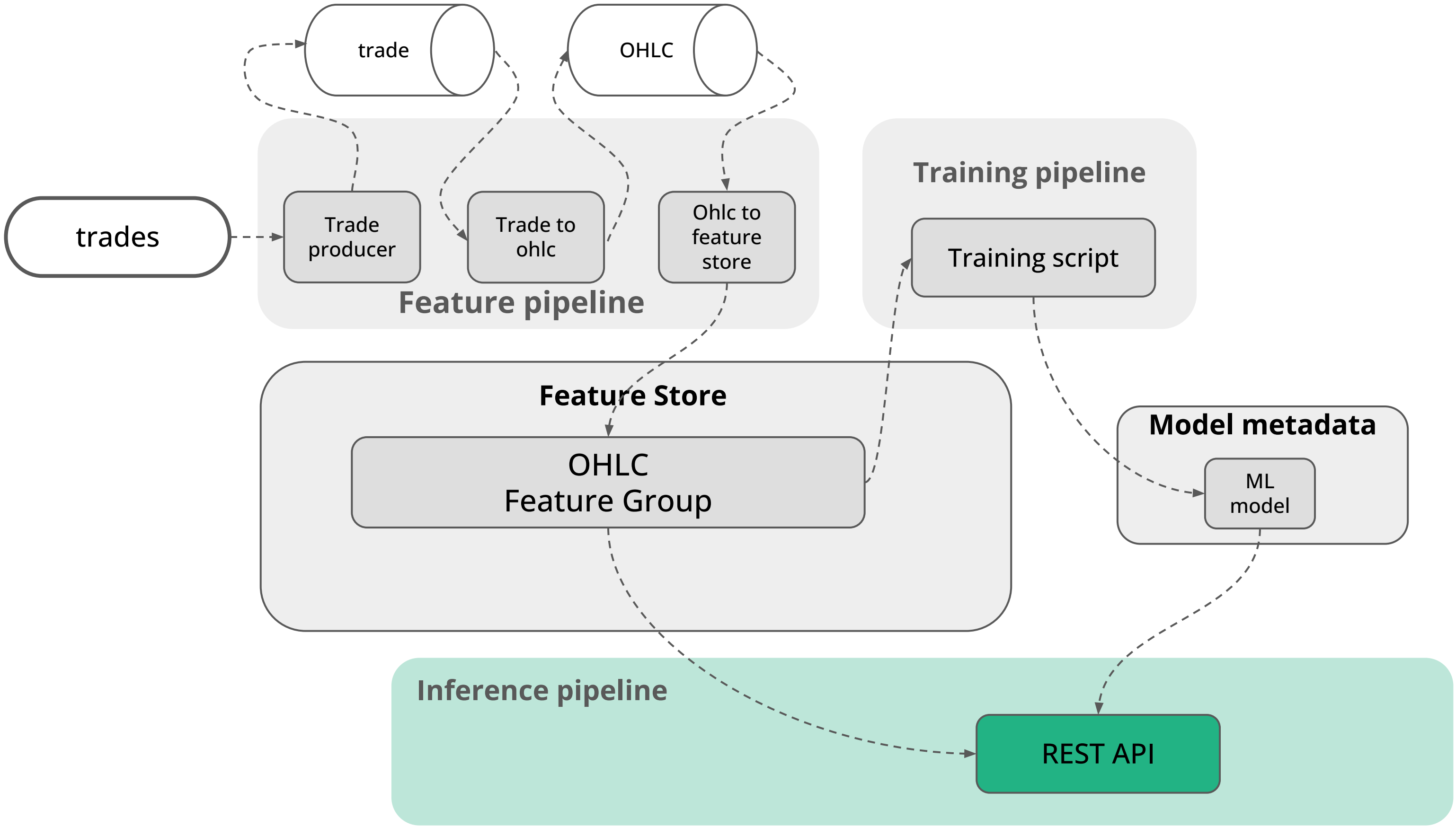
Push model to Model Registry



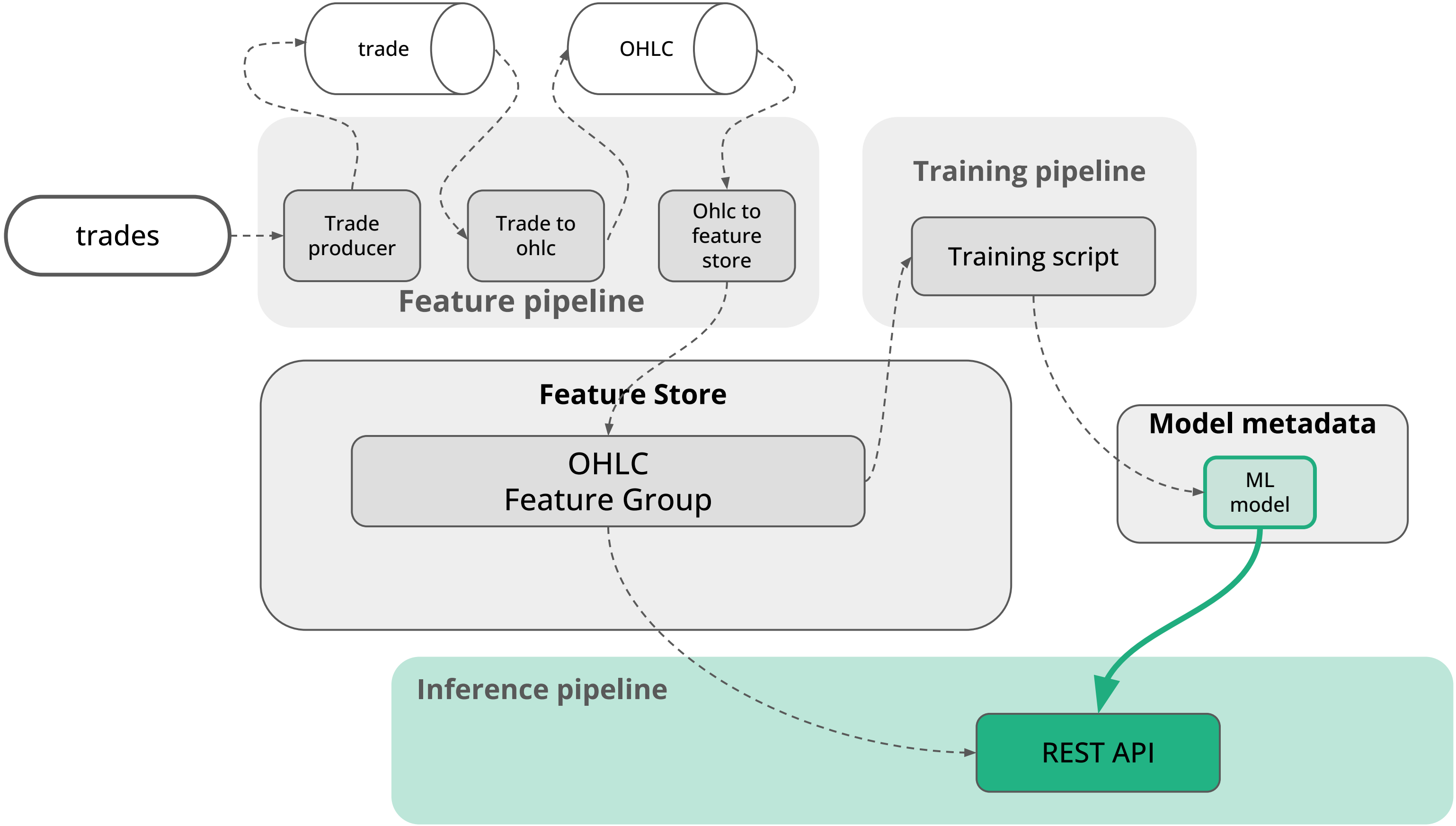
Push model to Model Registry



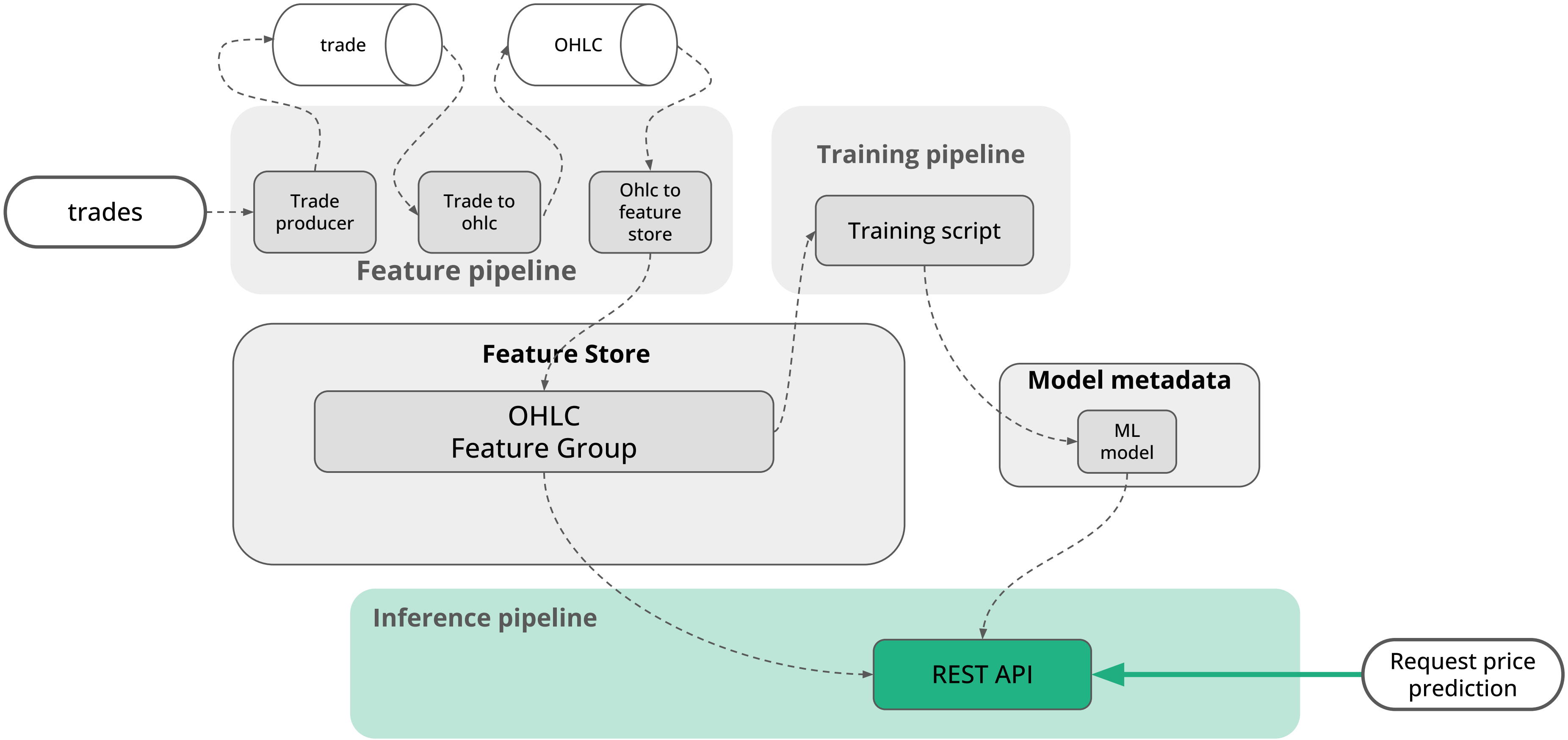
Start building the Inference pipeline



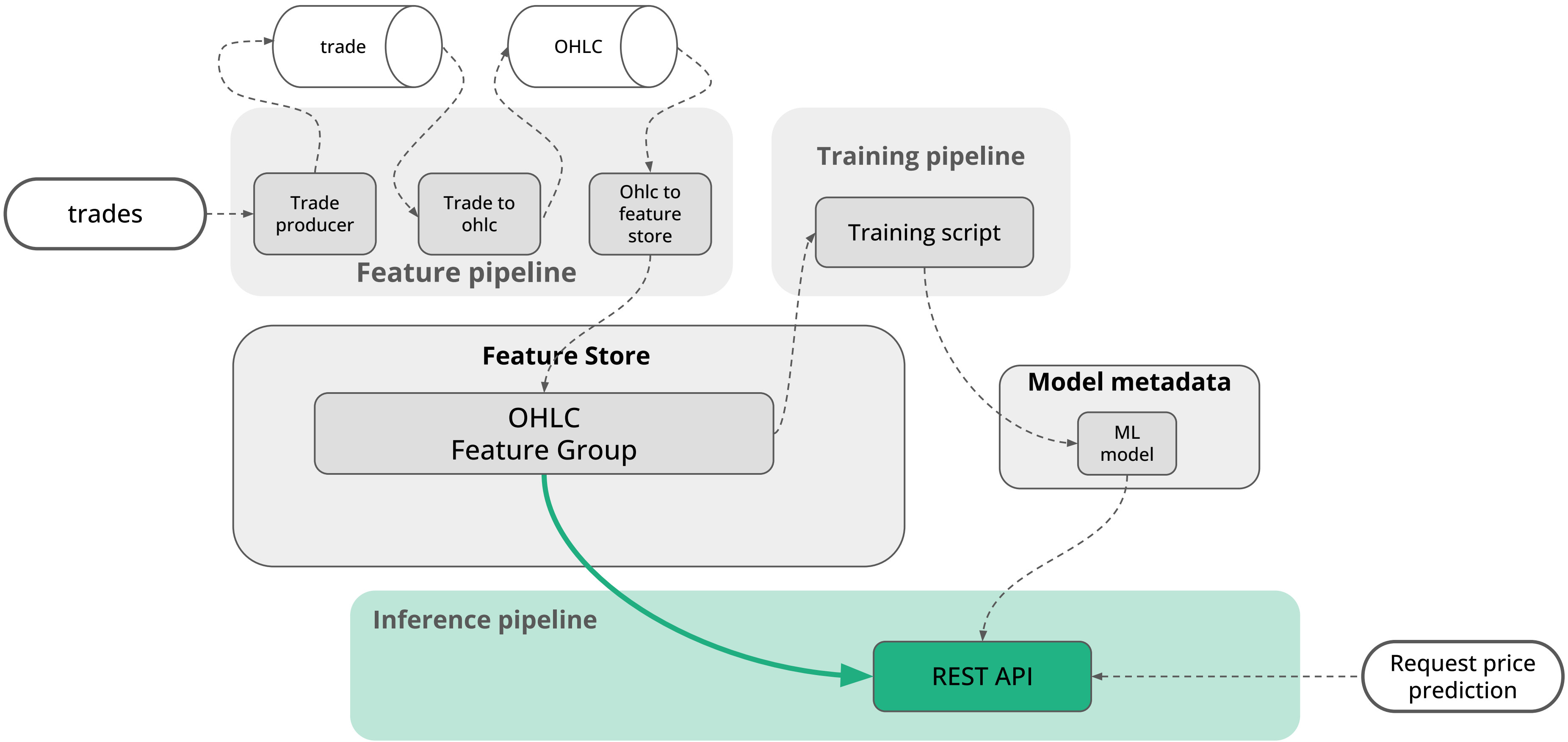
Load Production model from registry



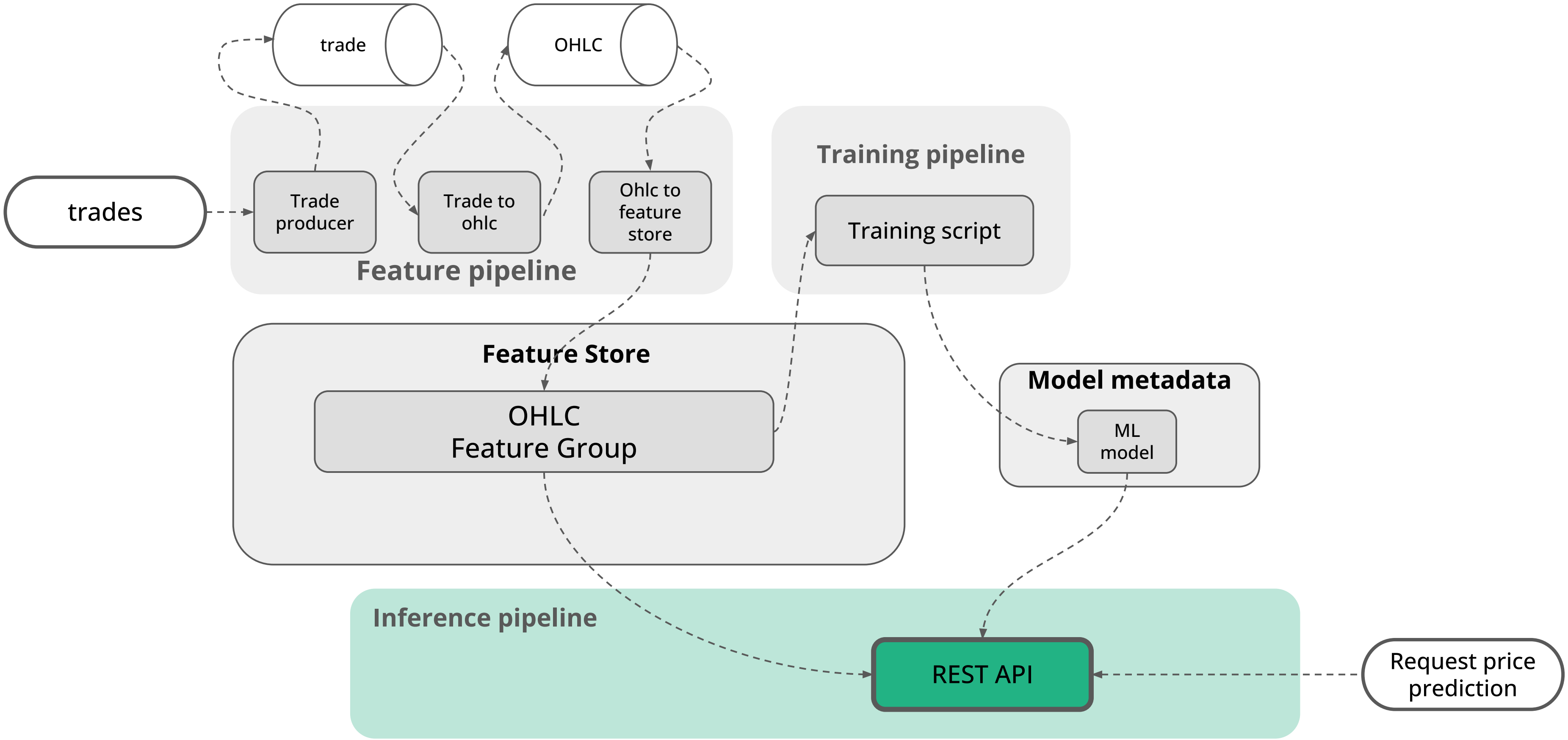
Listens to incoming requests



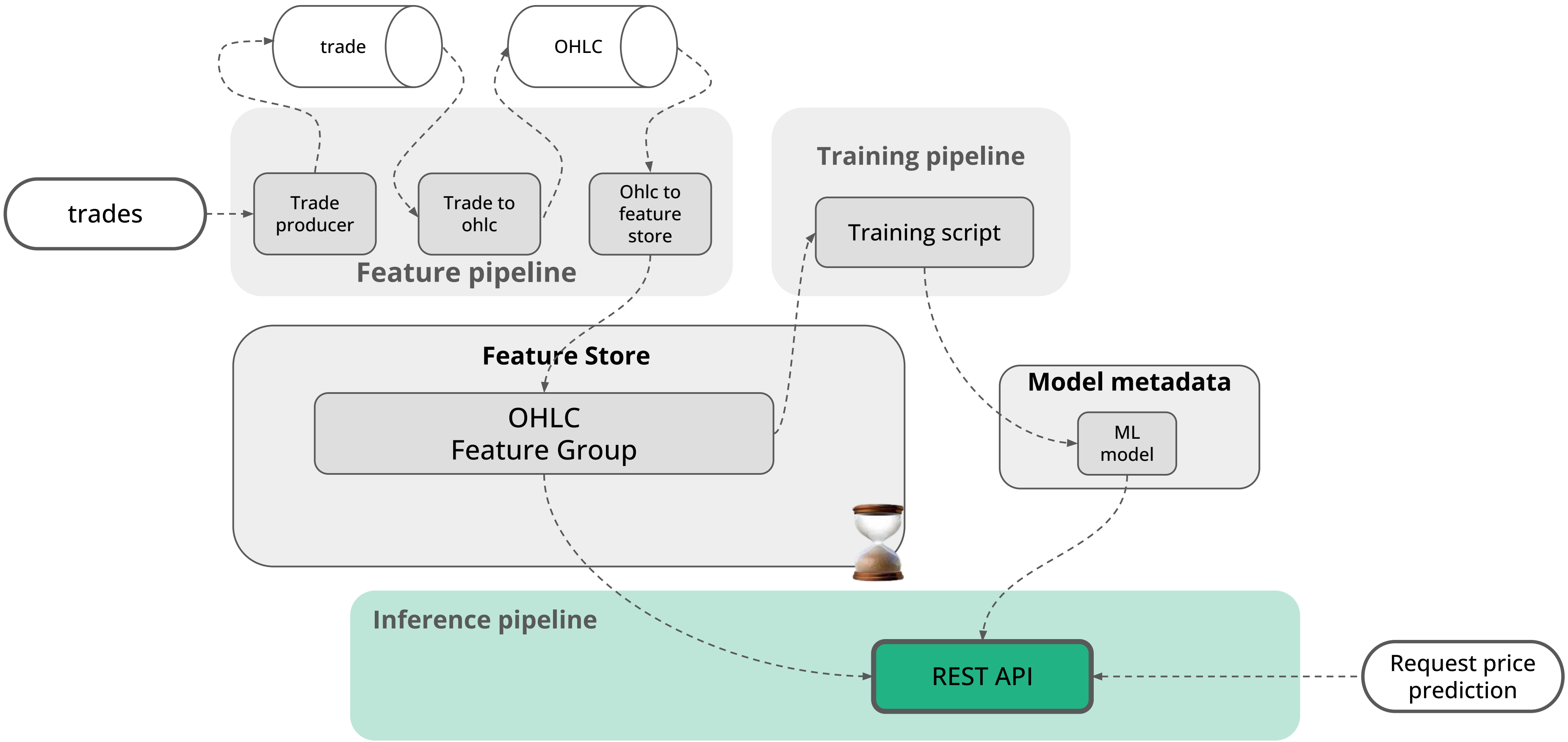
Gets fresh features from the store



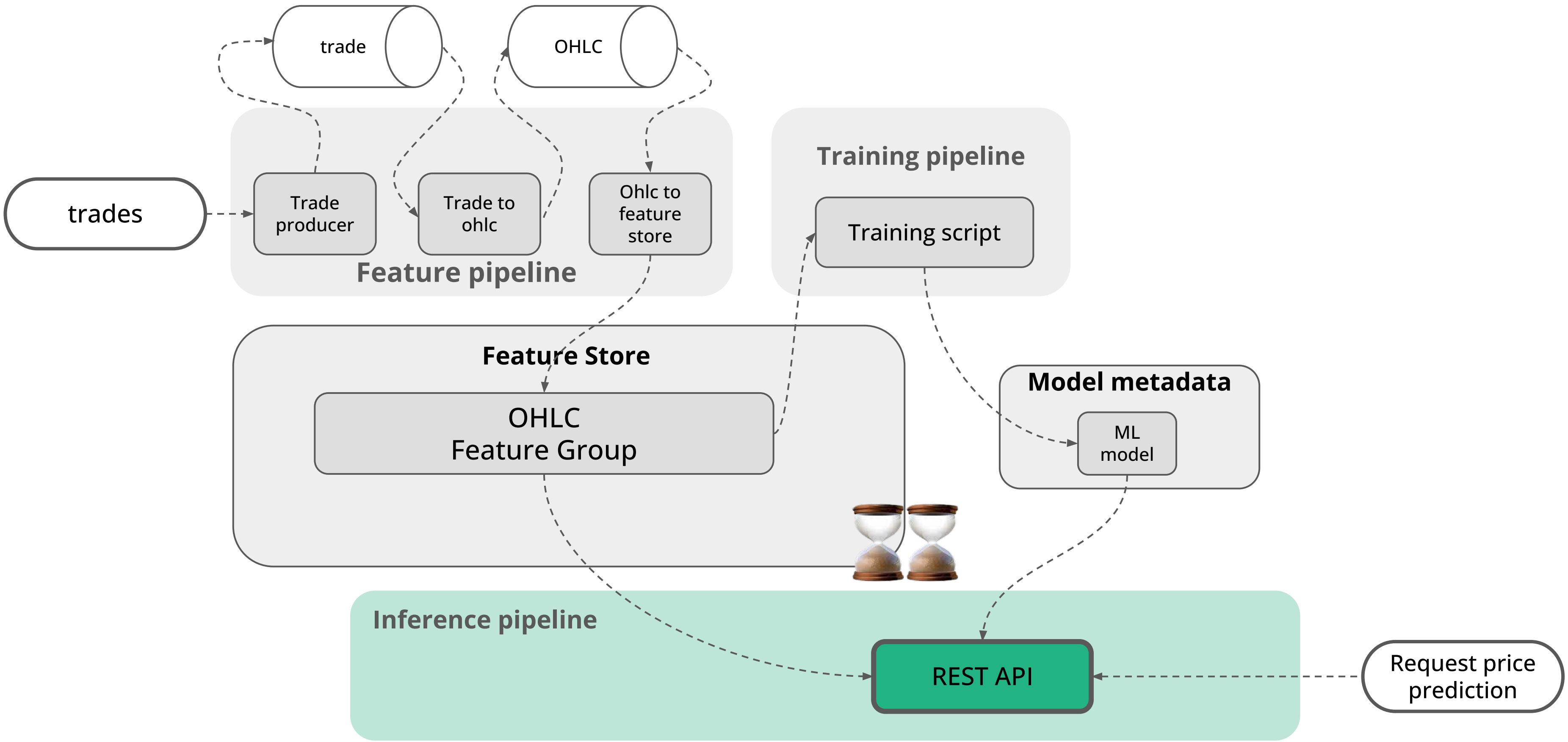
Computes prediction with ML Model



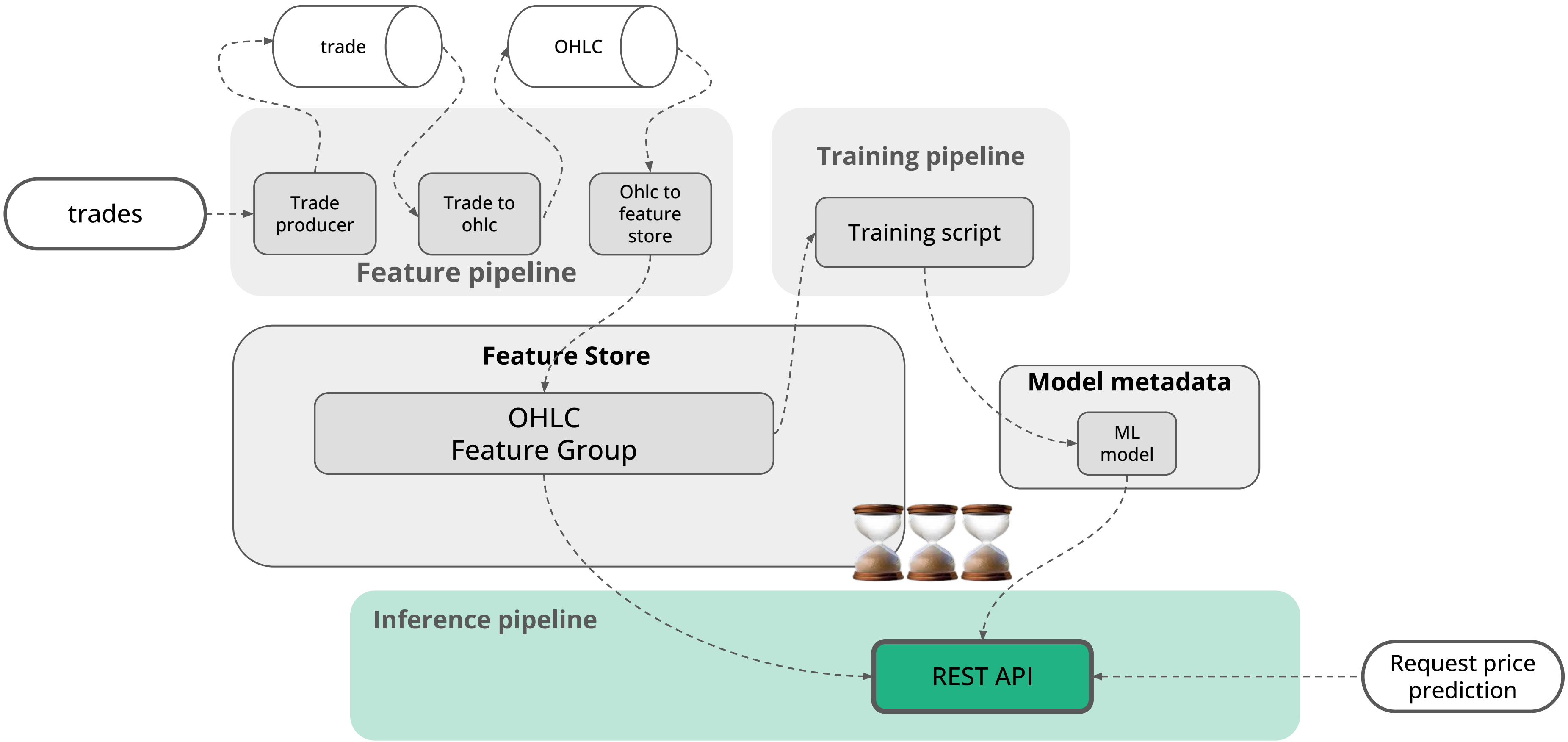
Computes prediction with ML Model



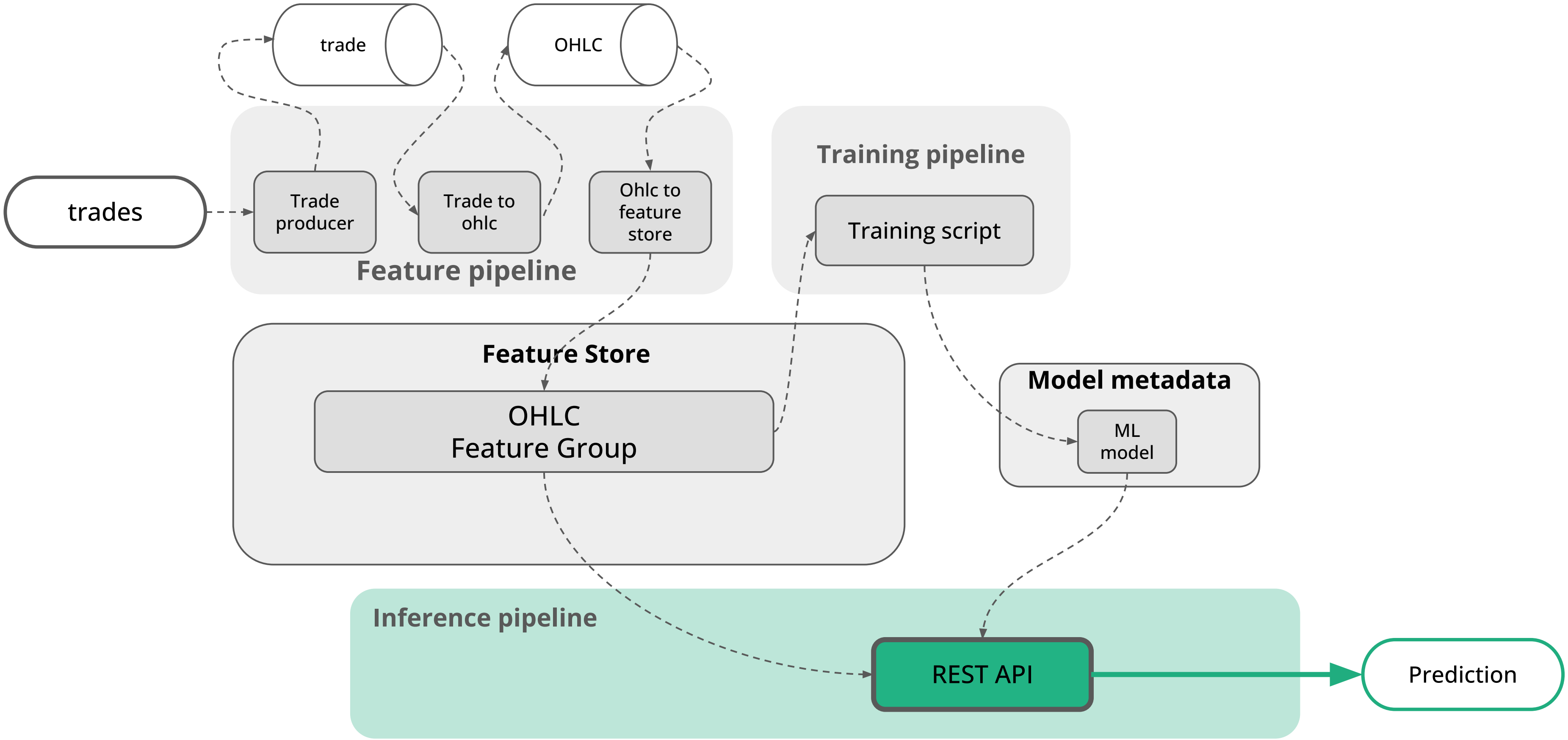
Computes prediction with ML Model



Computes prediction with ML Model

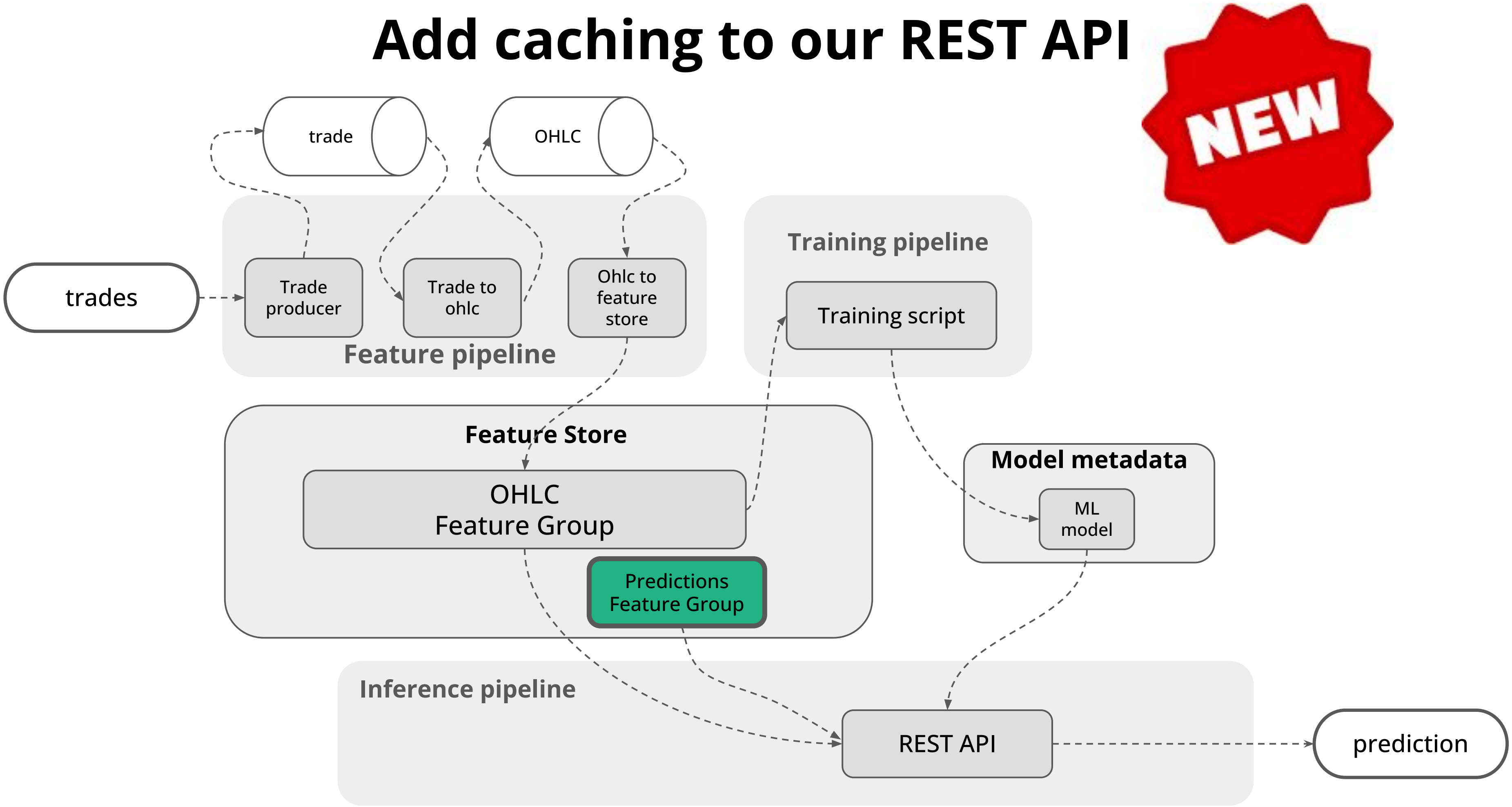


Computes prediction with ML Model

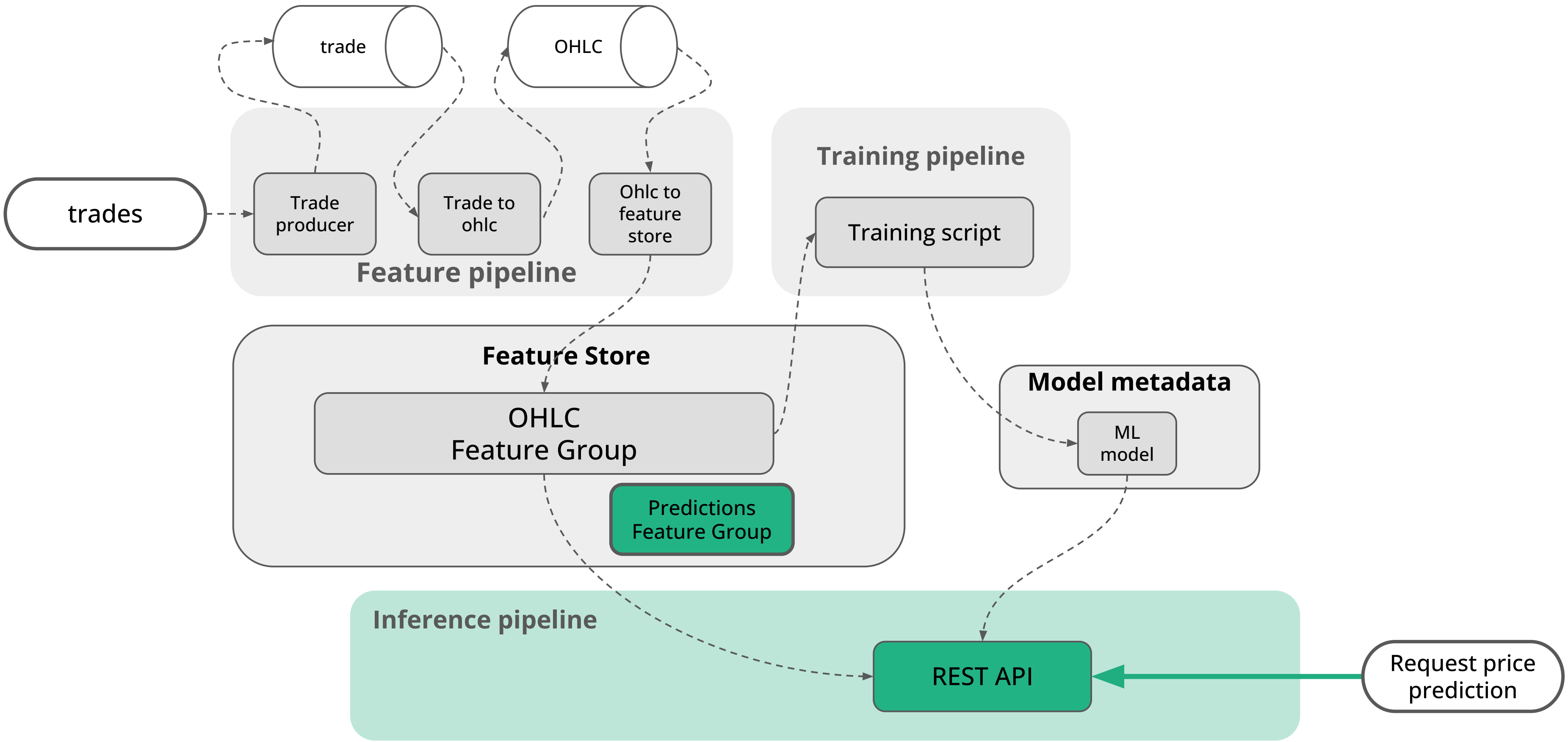


REST API caching

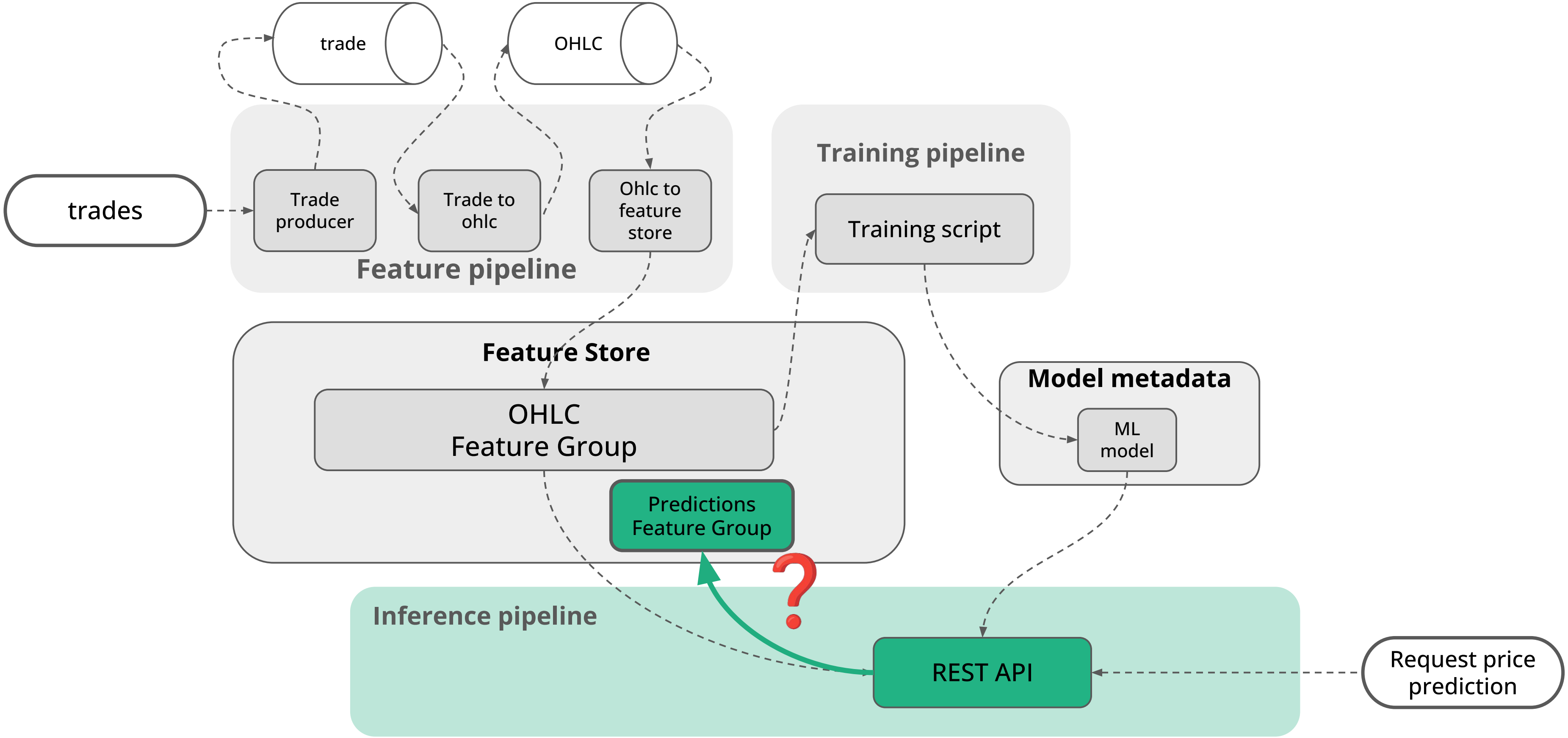
Add caching to our REST API



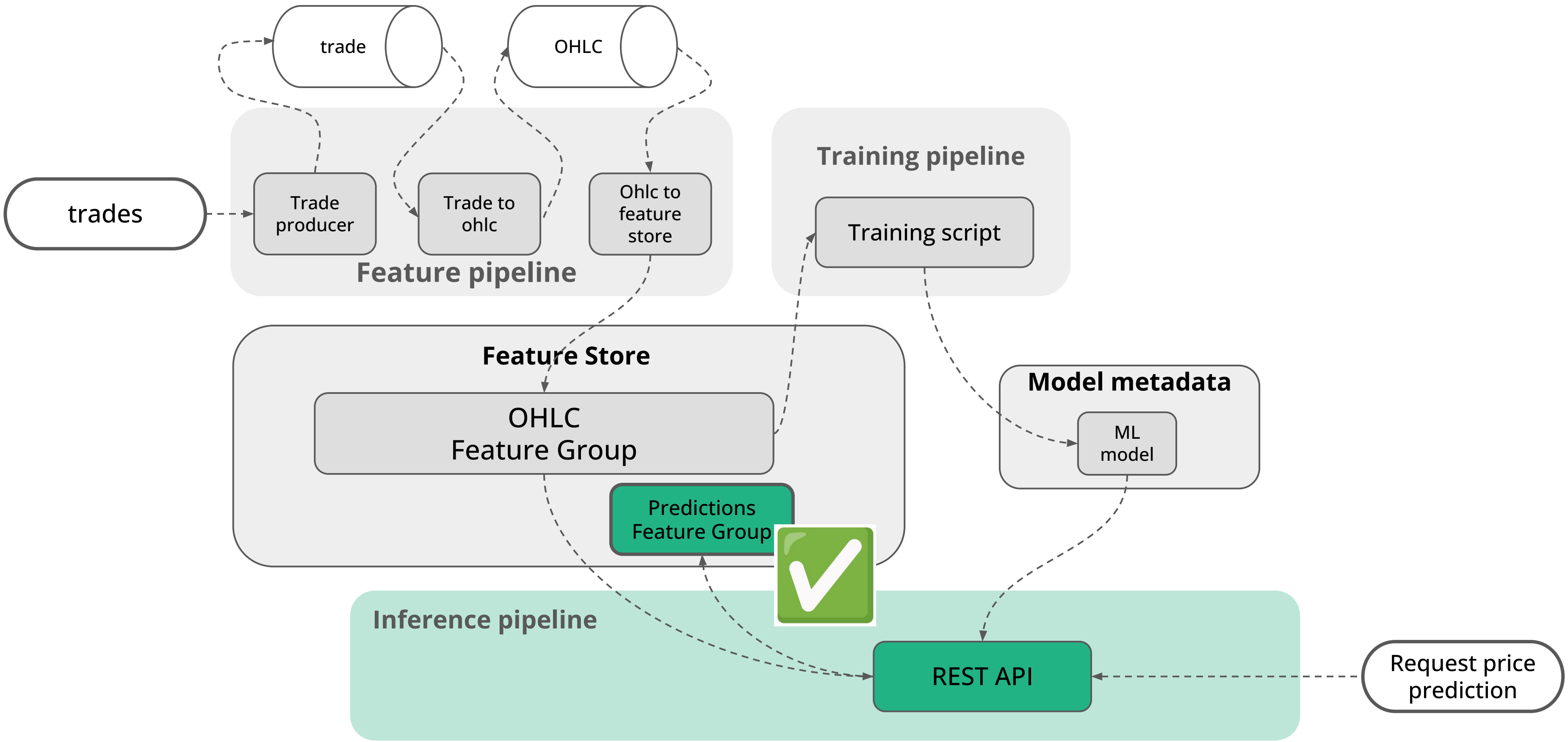
Listens to incoming requests



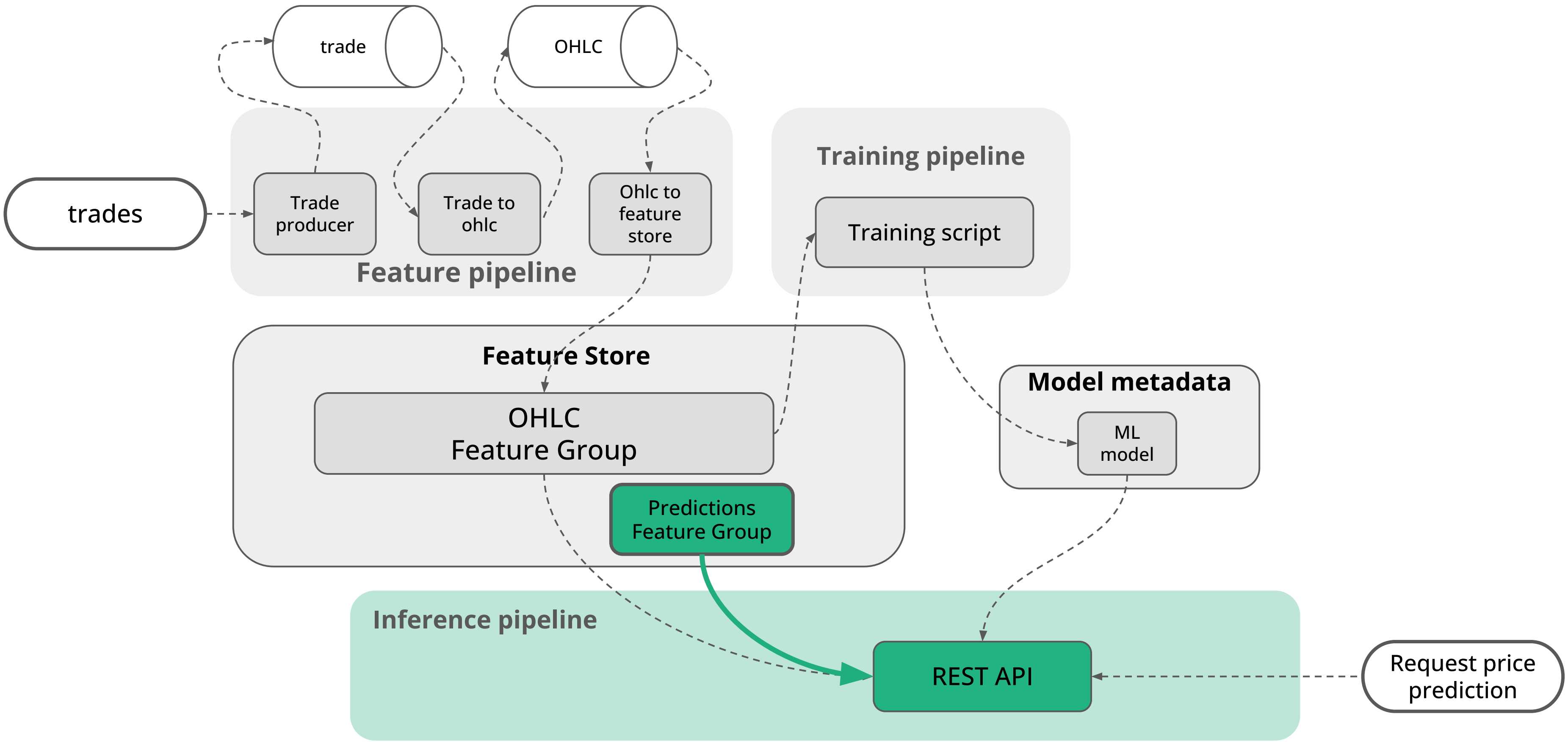
Checks if request is cached



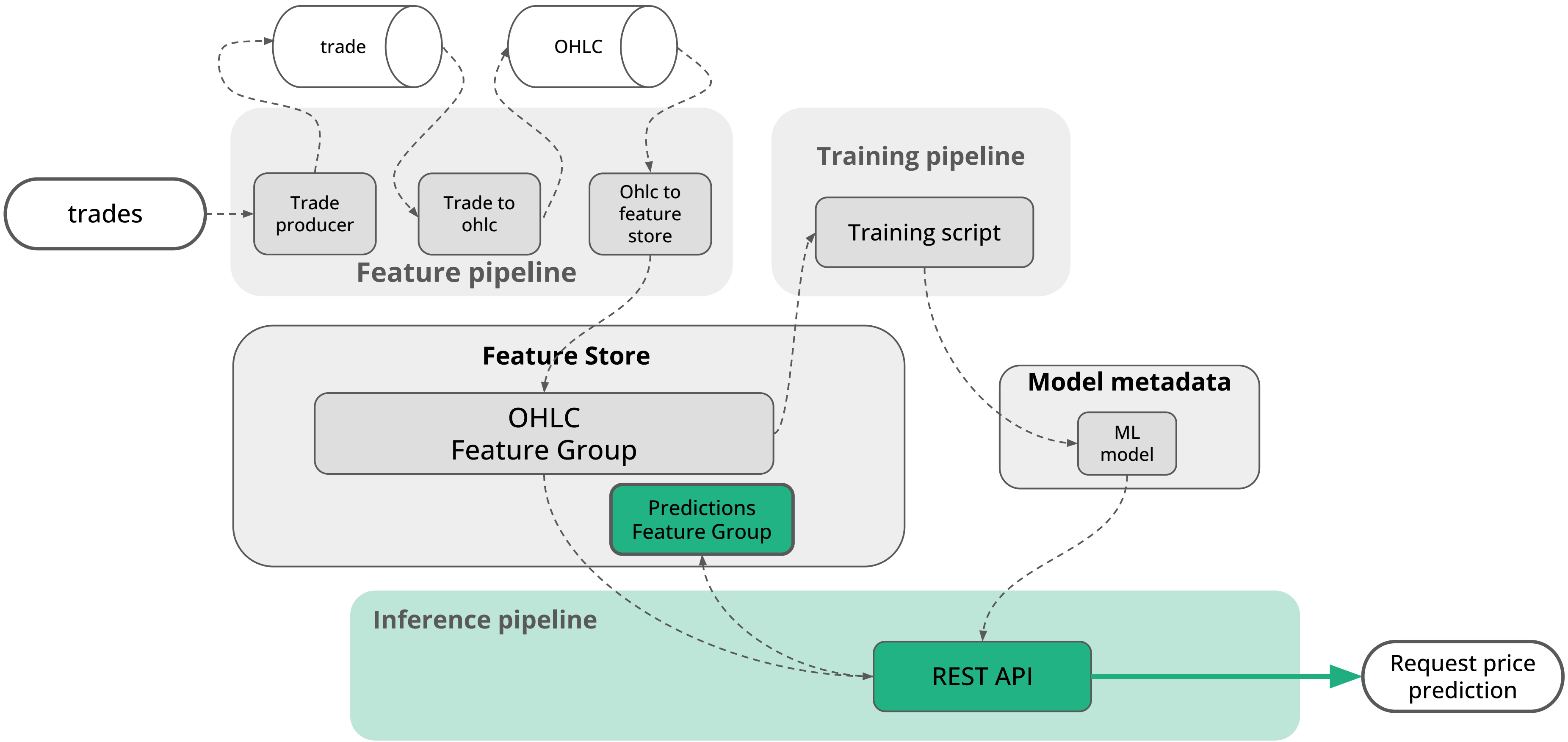
If YES


trade
OHLC
trades
Trade producer
Trade to ohlc
Ohlc to feature store
Feature pipeline
Training pipeline
Training script
Feature Store
OHLC Feature Group
Predictions Feature Group
Model metadata
ML model
Inference pipeline
REST API
Request price prediction

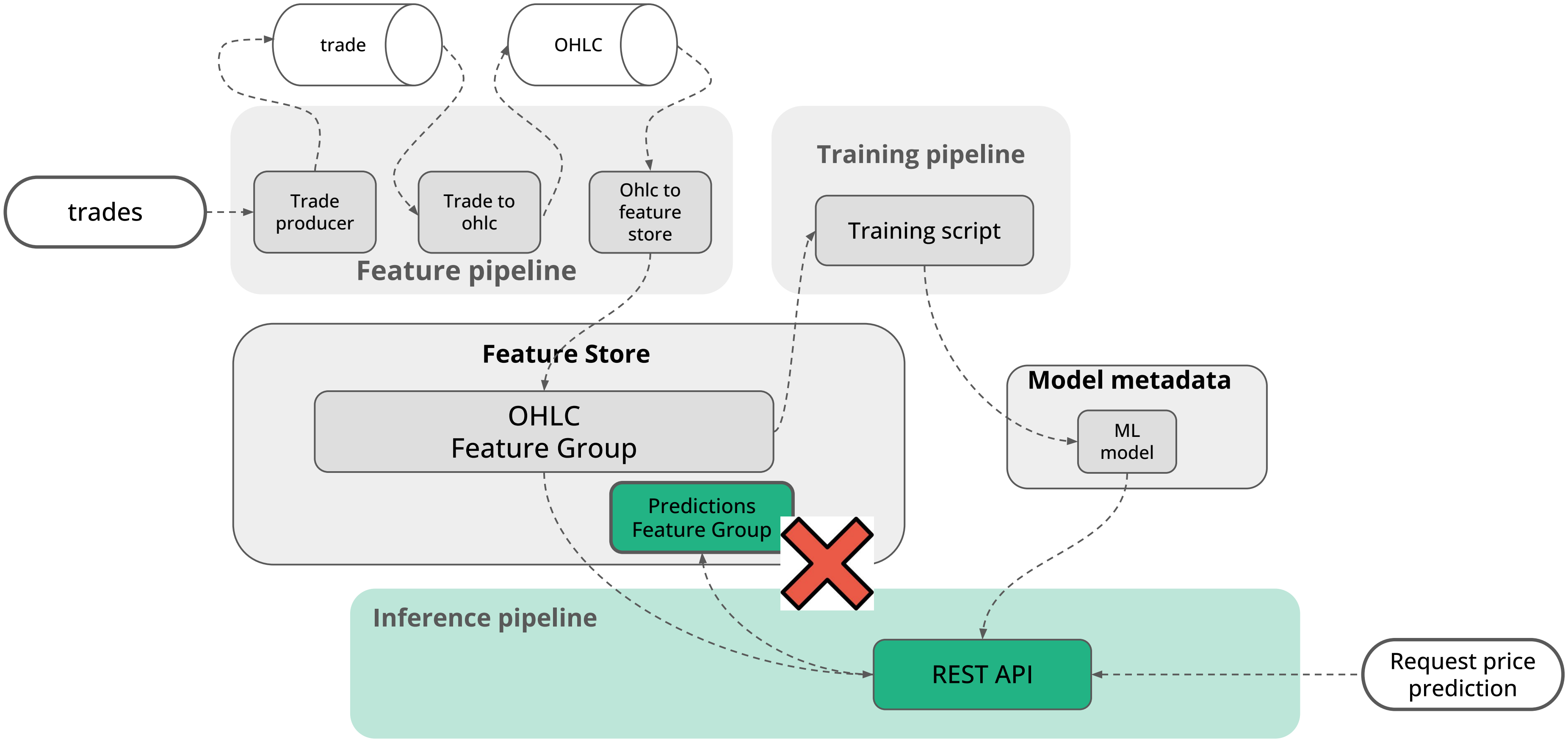
Fetch cached prediction



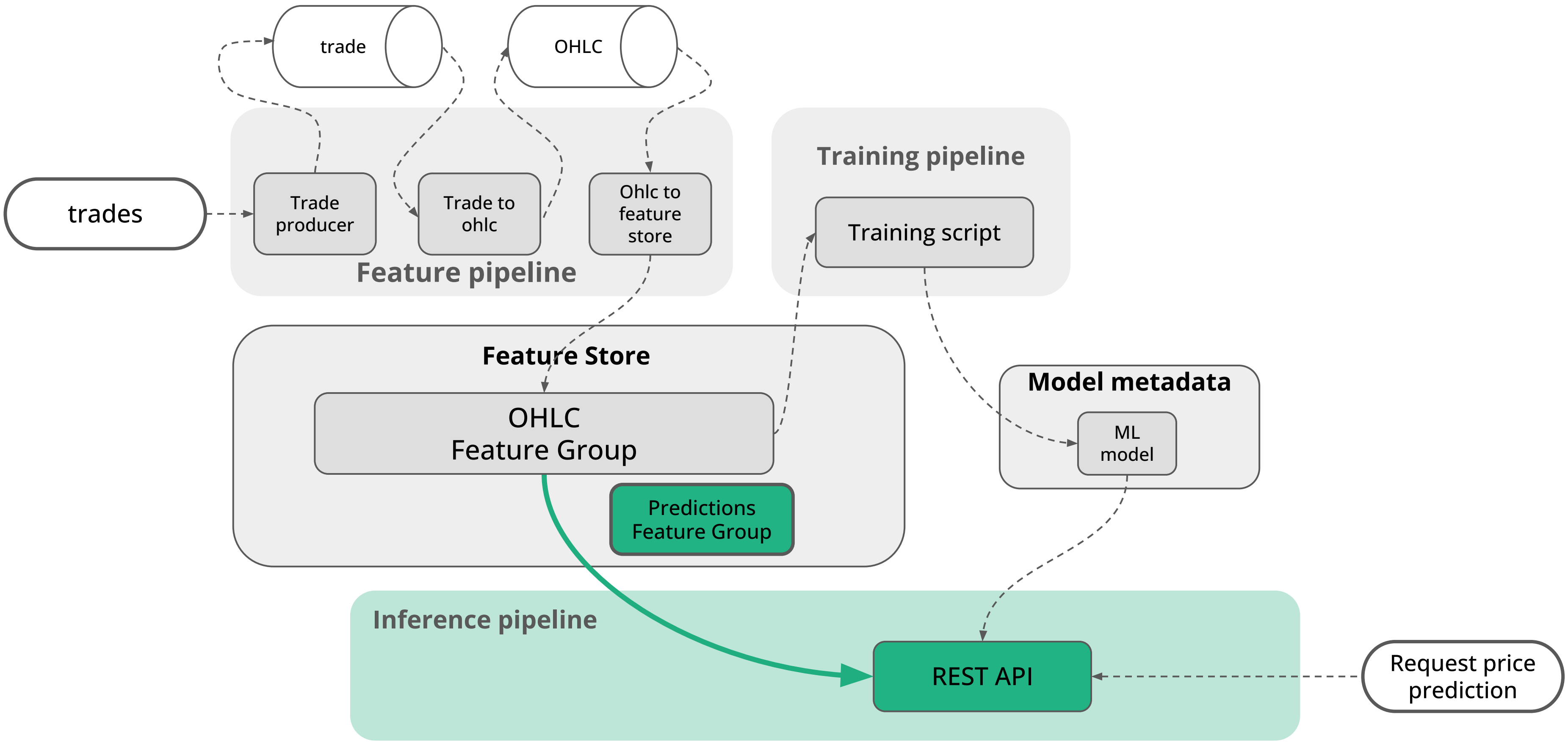
Return cached prediction



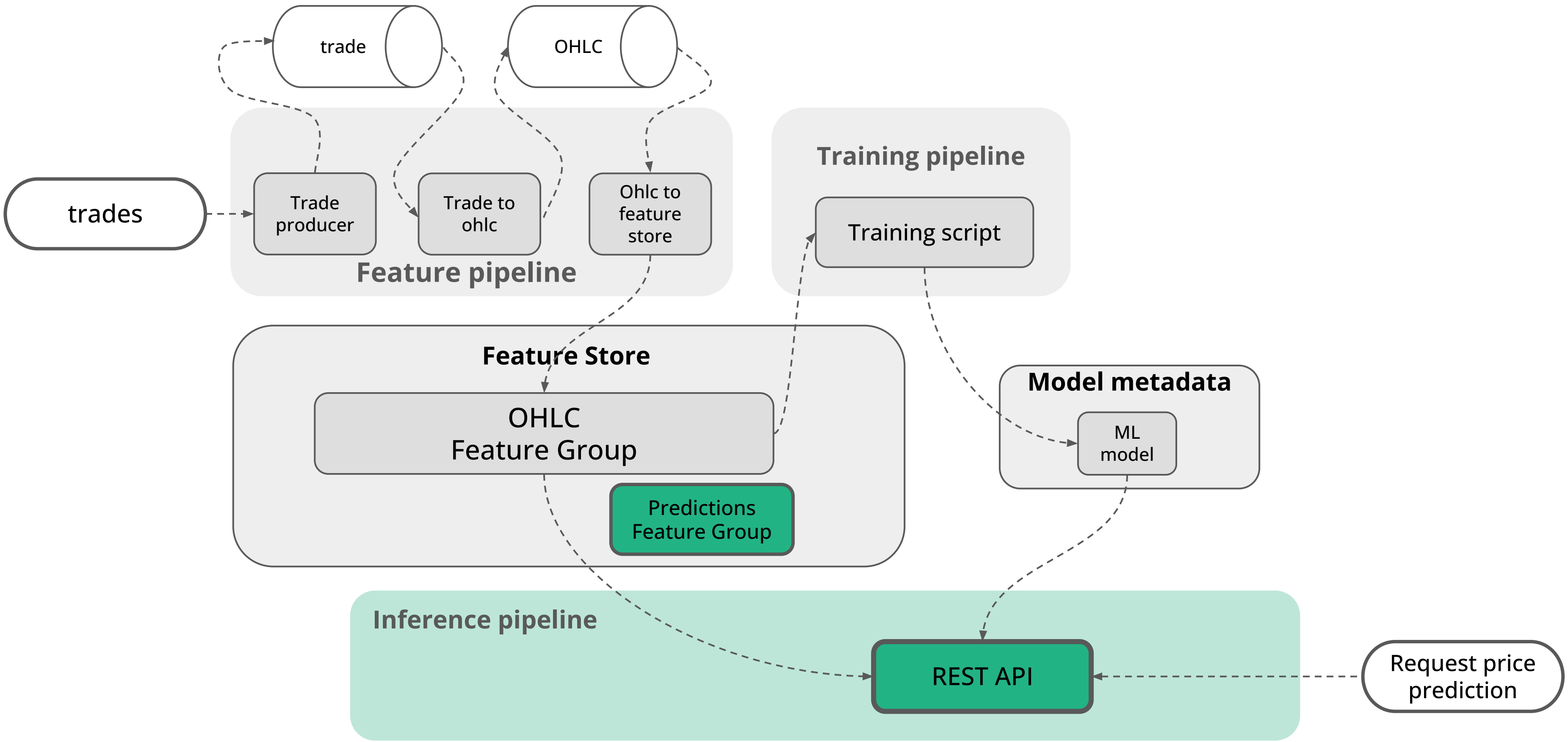
If NO


trade
OHLC
trades
Trade producer
Trade to ohlc
Ohlc to feature store
Feature pipeline
Training pipeline
Training script
Feature Store
OHLC Feature Group
Predictions Feature Group
Model metadata
ML model
Inference pipeline
REST API
Request price prediction

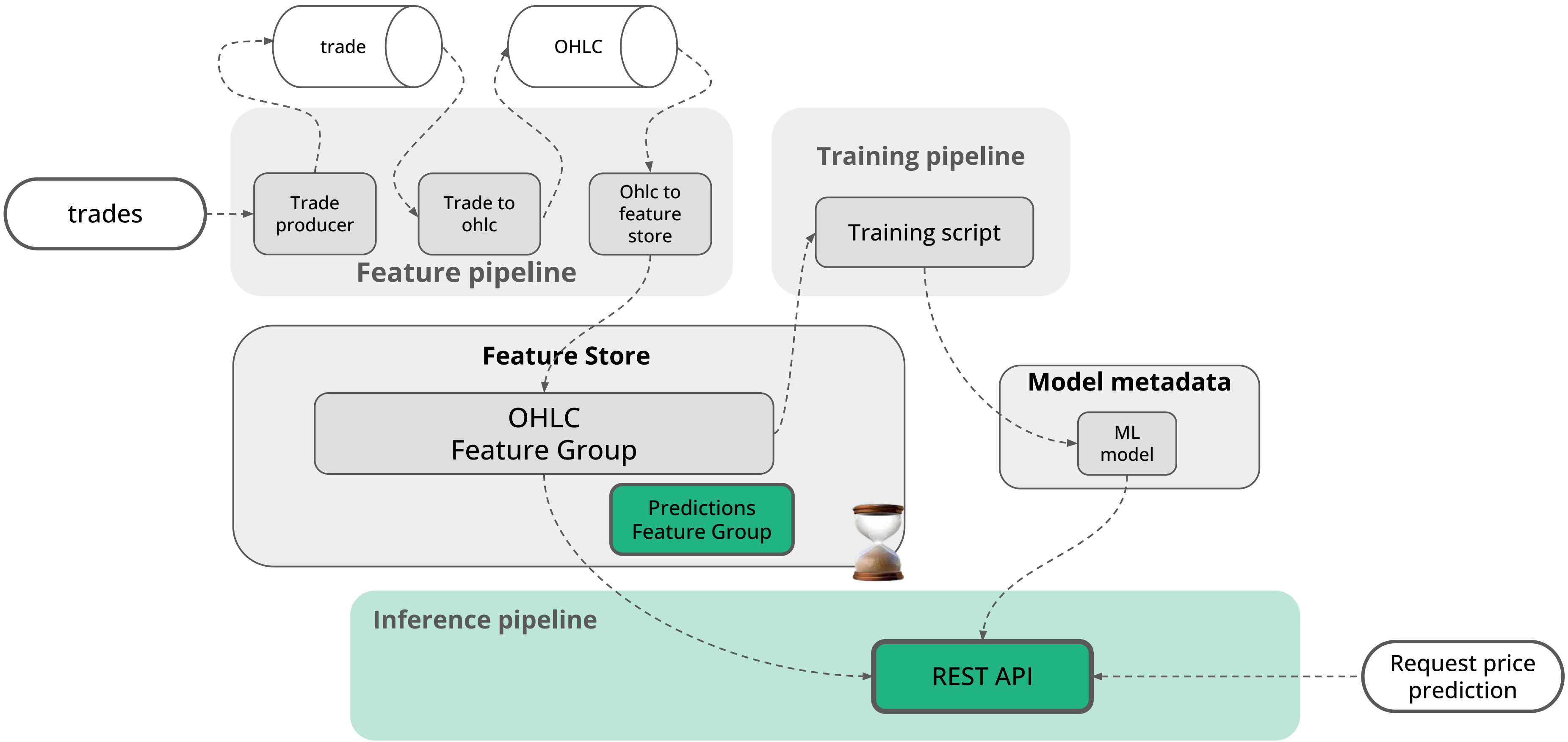
Gets fresh features from the store



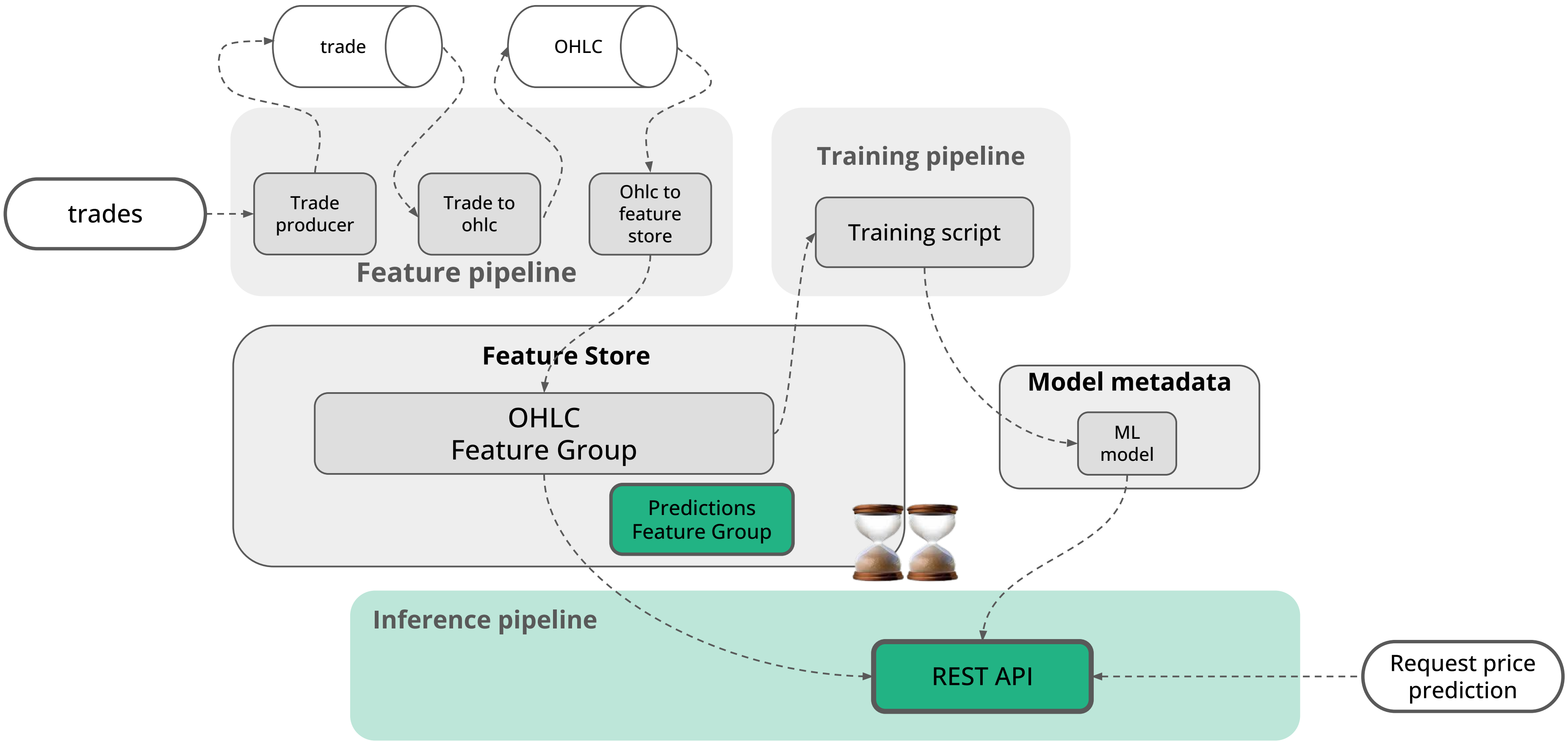
Computes prediction with ML Model



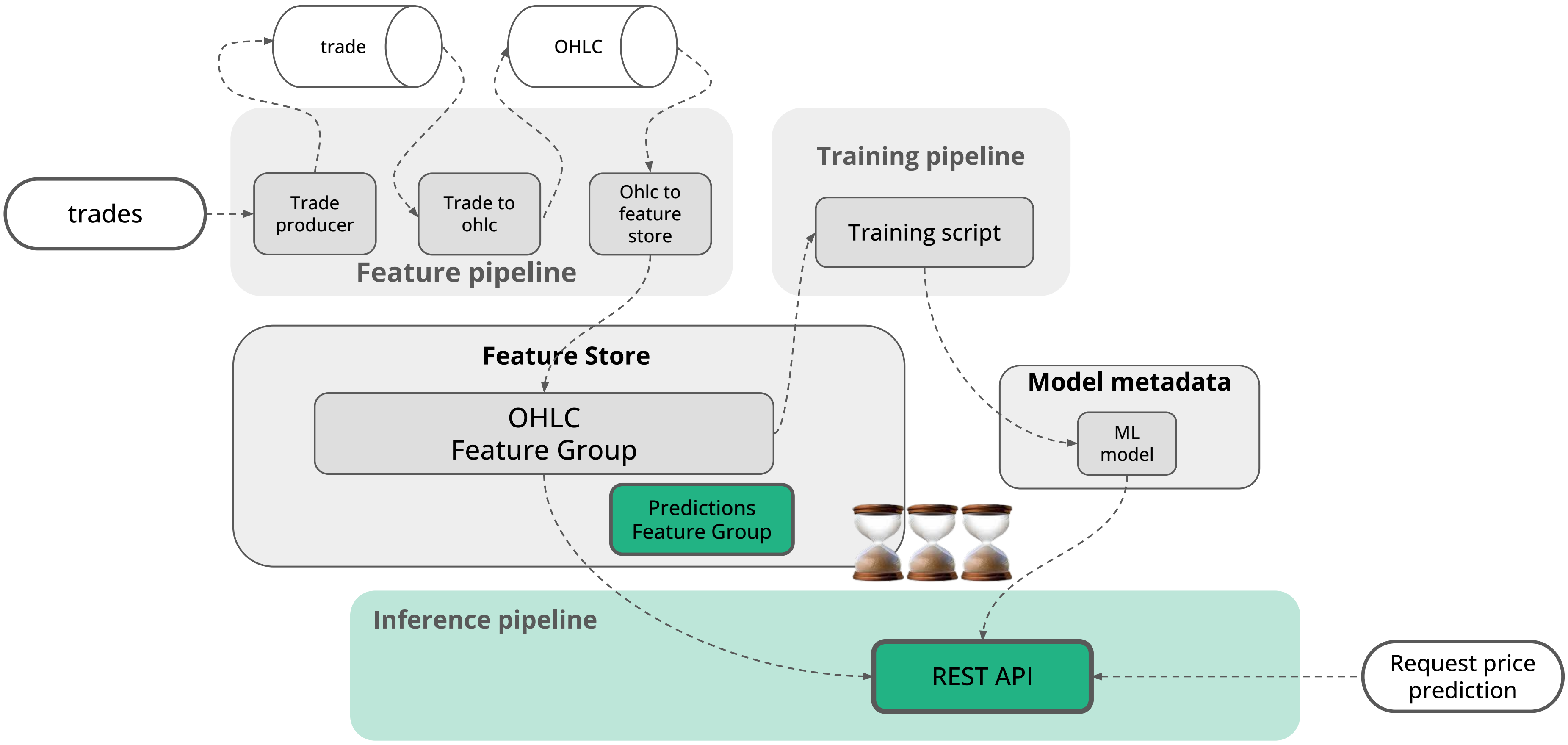
Computes prediction with ML Model



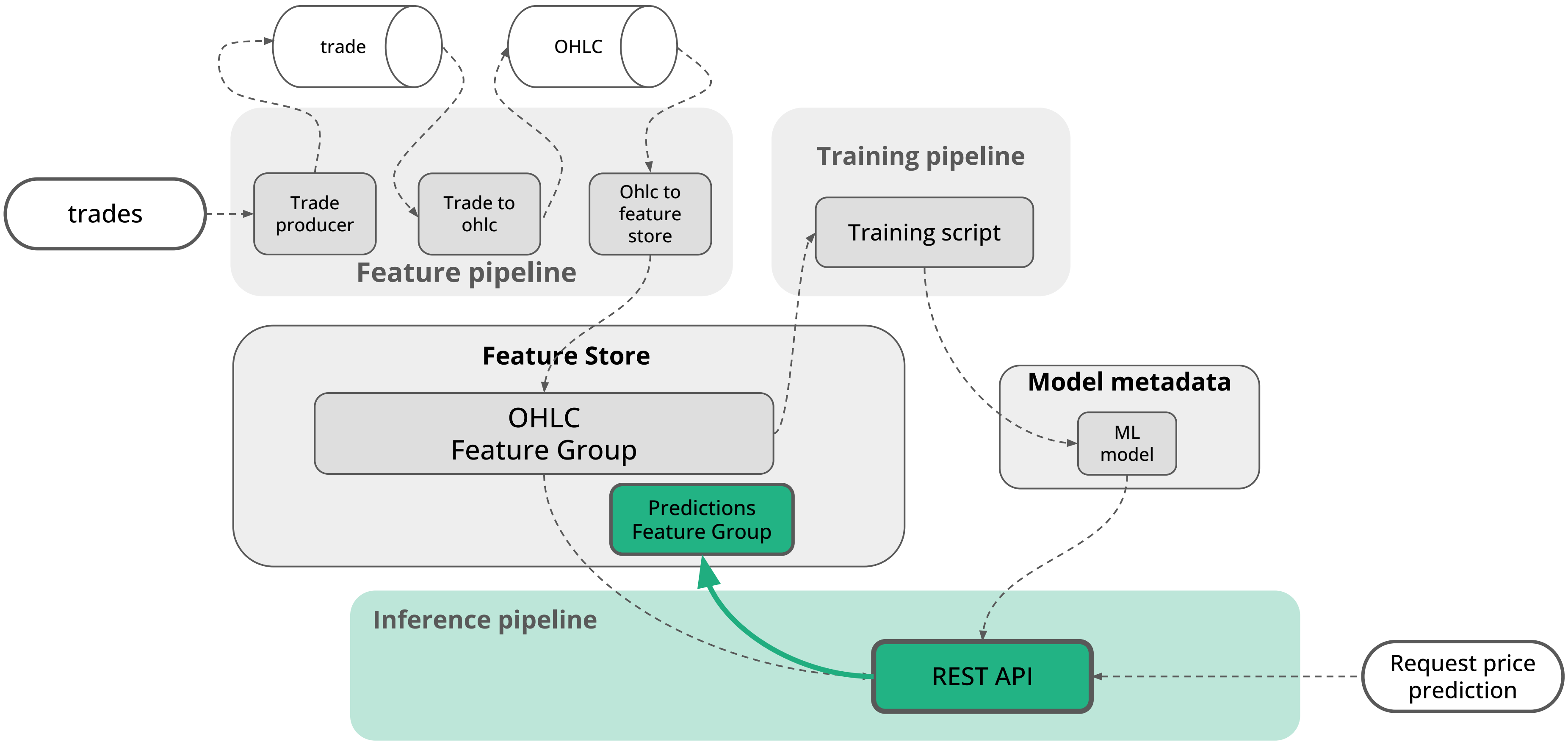
Computes prediction with ML Model



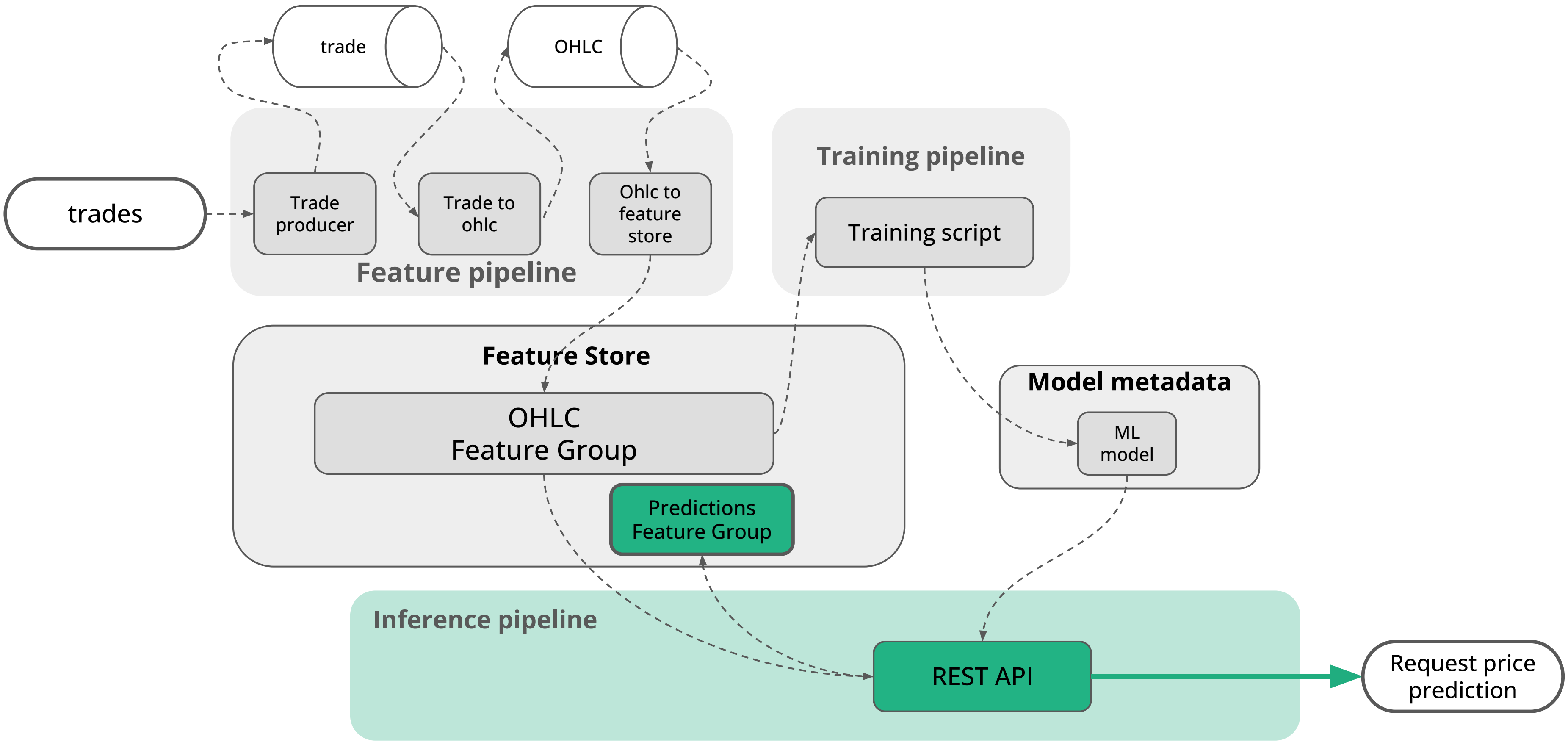
Computes prediction with ML Model



Pushes prediction to cache

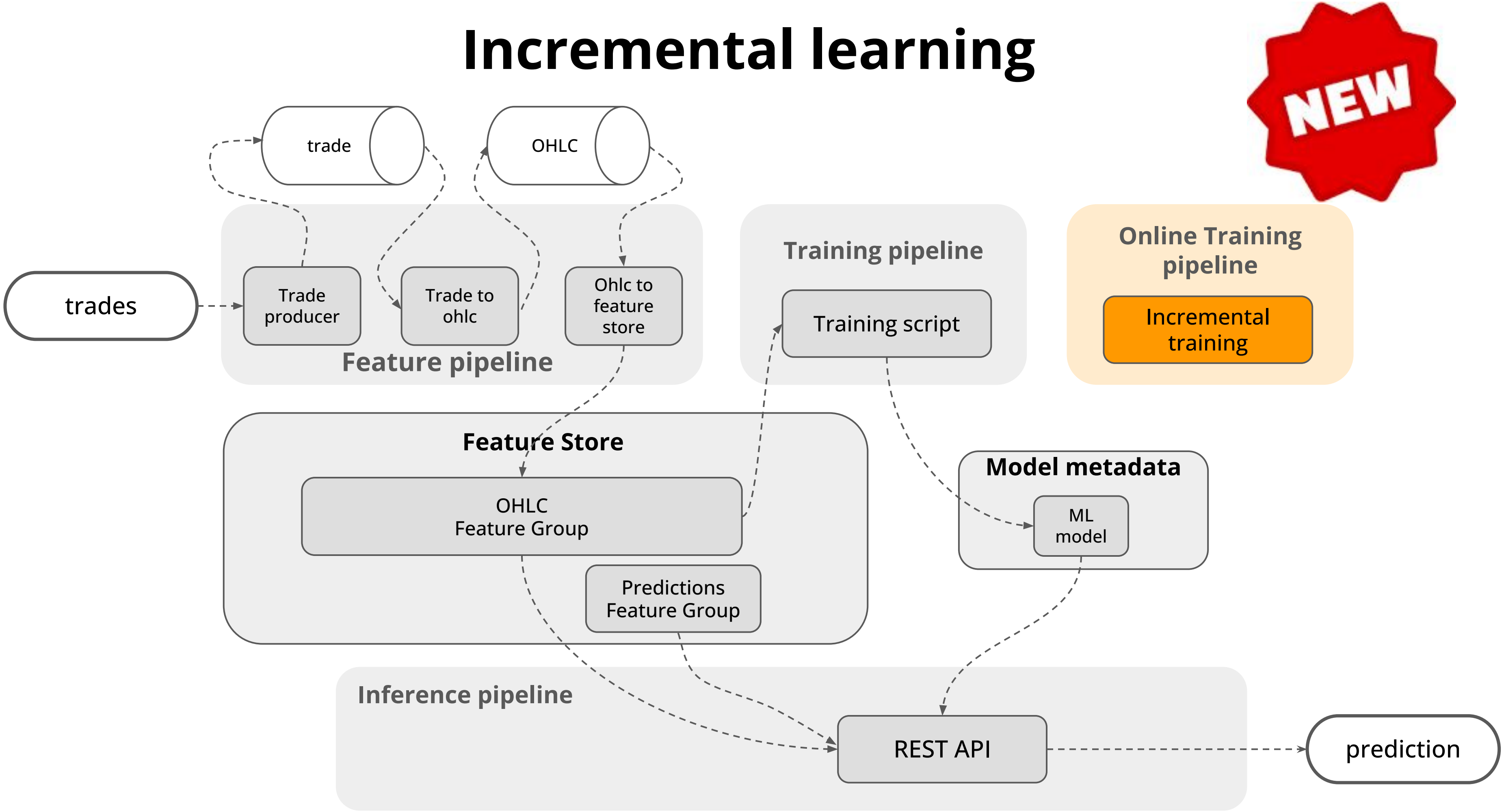


Returns prediction

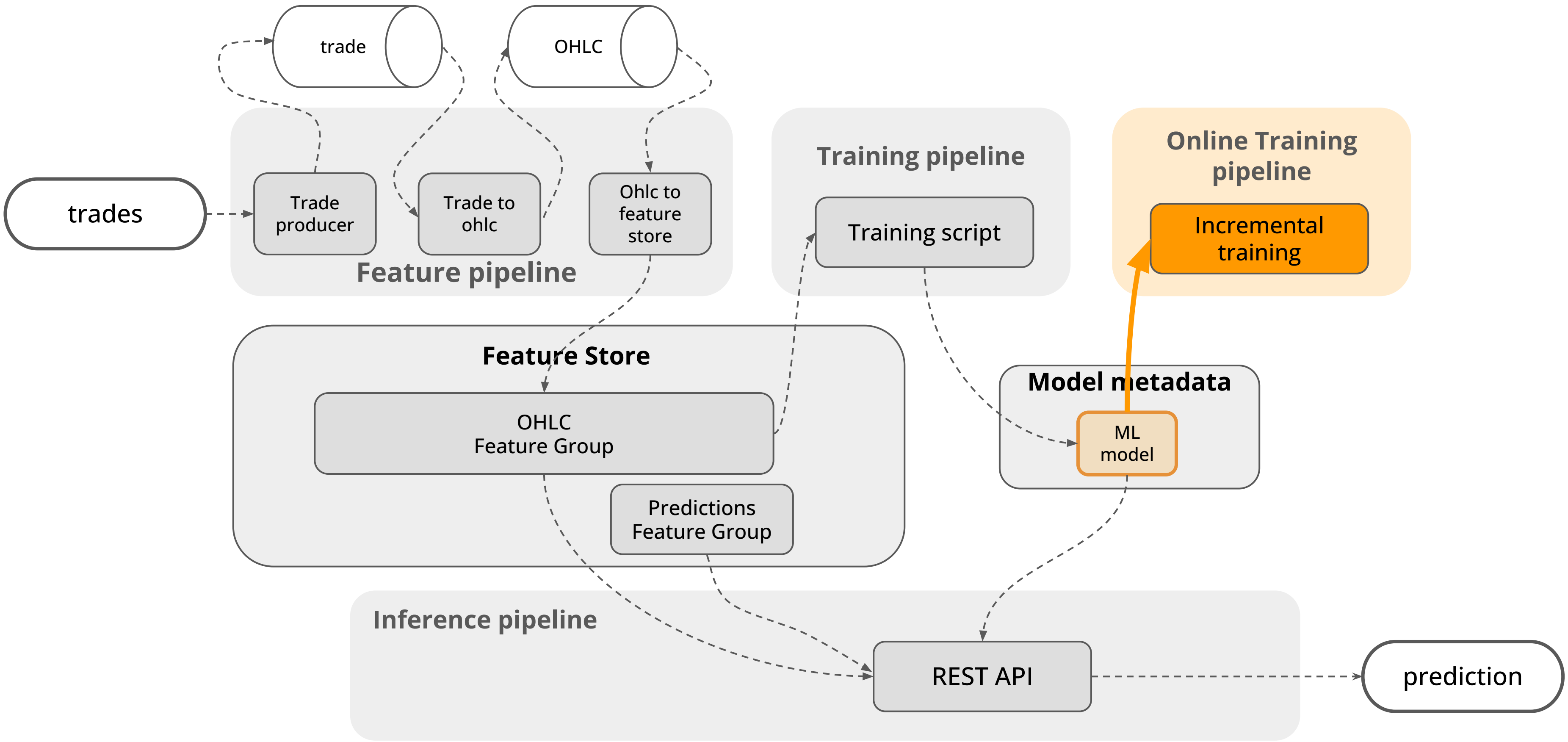


Incremental learning

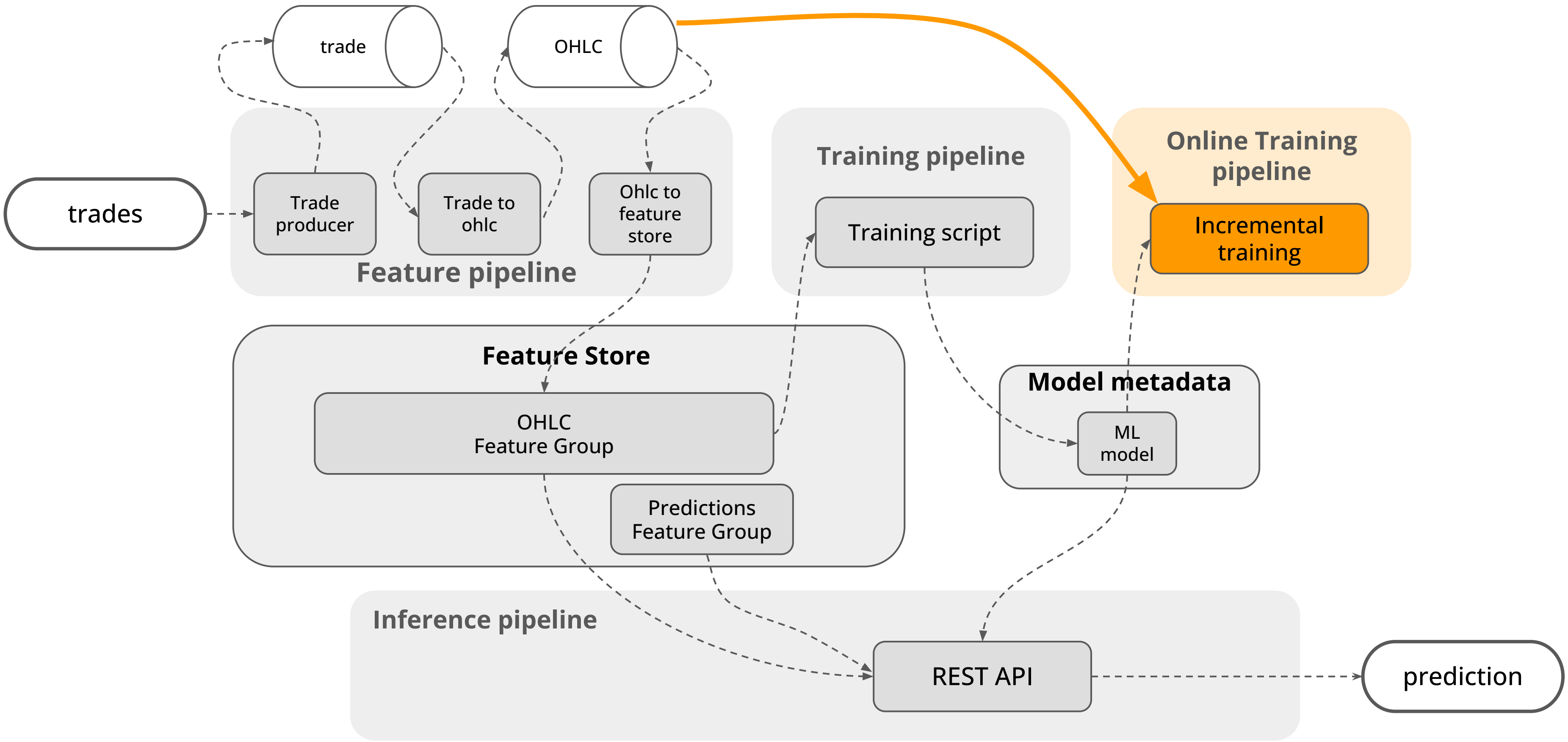
Incremental learning



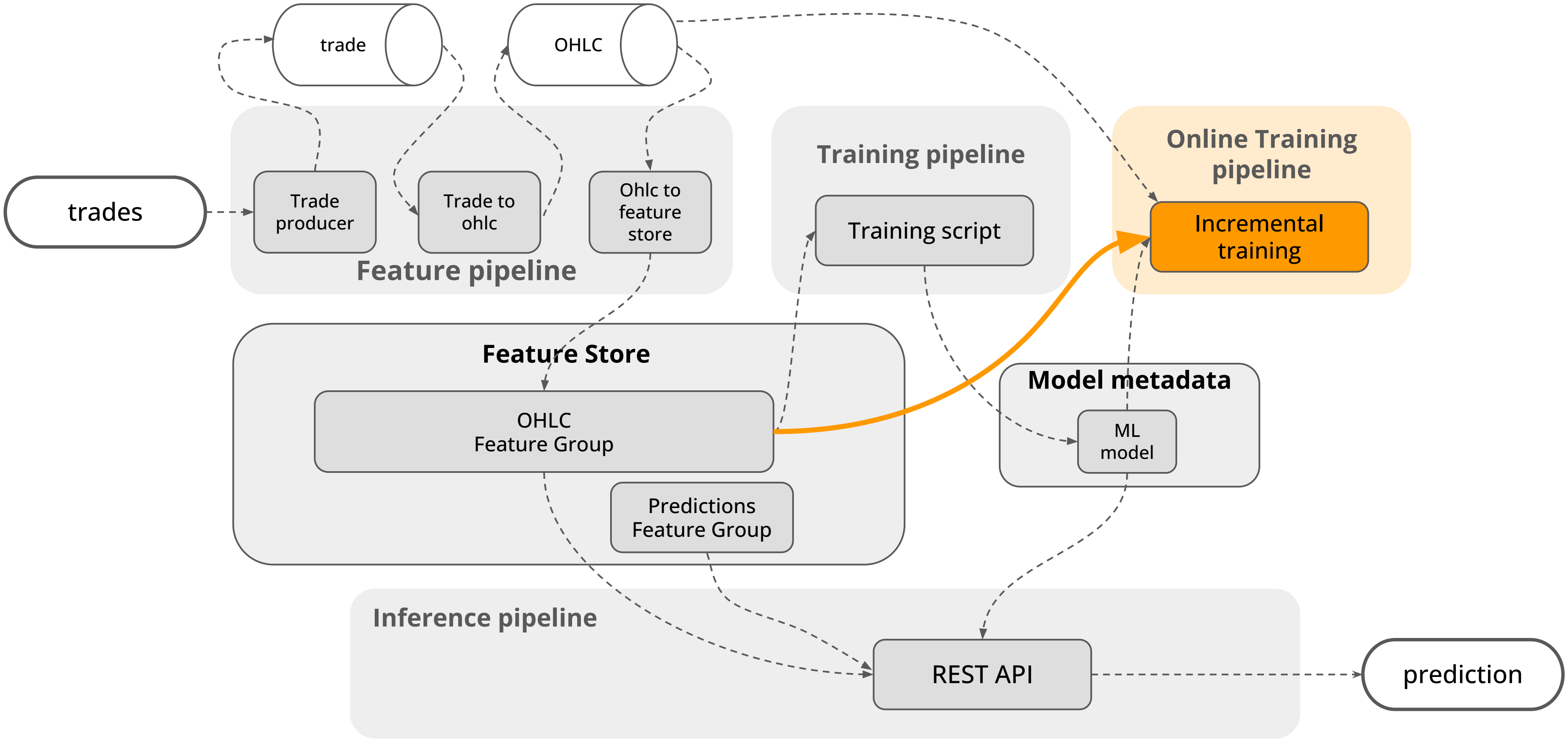
Loads PRODUCTION model from registry



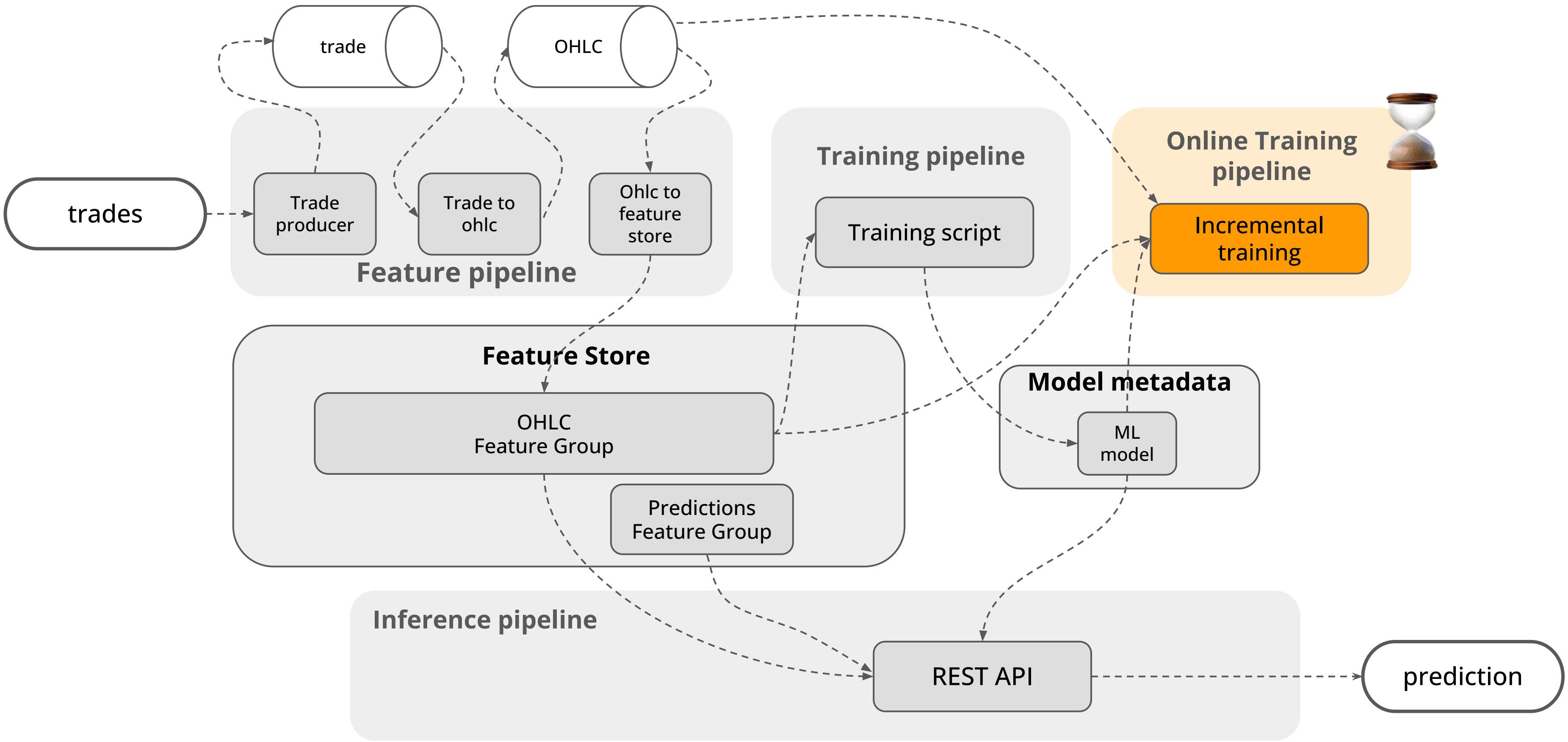
Consumes newest OHLC candle



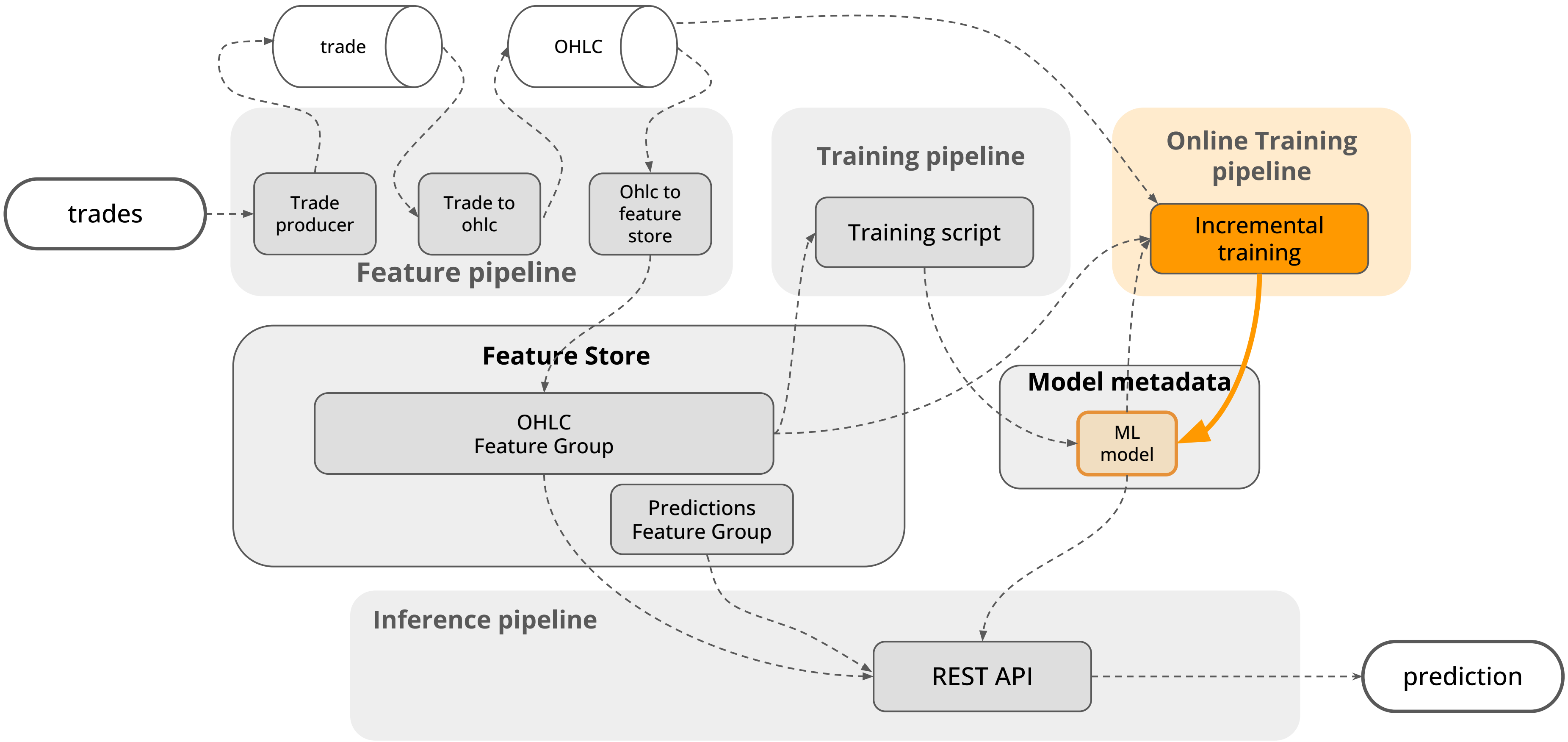
Fetches features from store



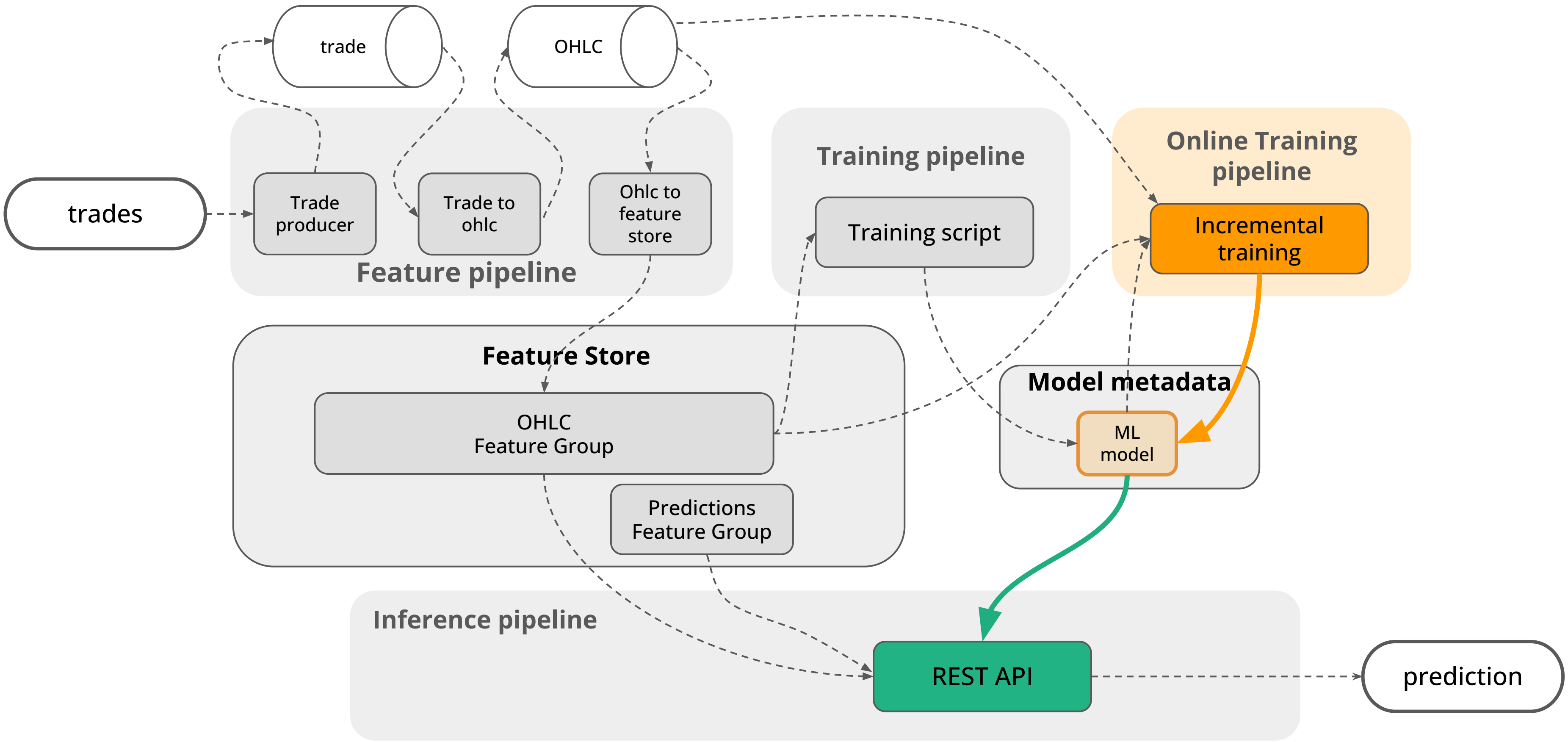
Updates the model parameters



Pushes updated model to registry

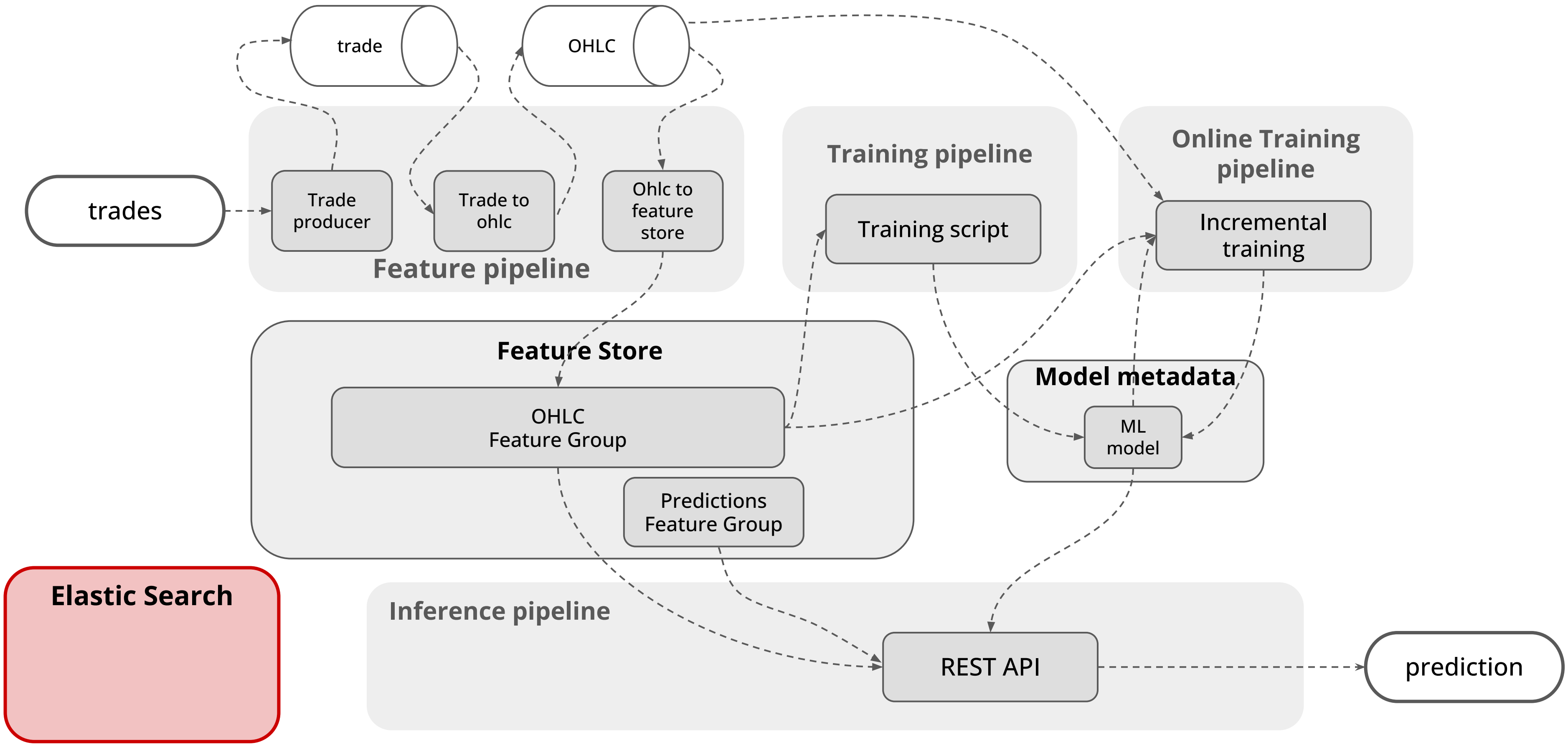


REST API periodically refreshes model

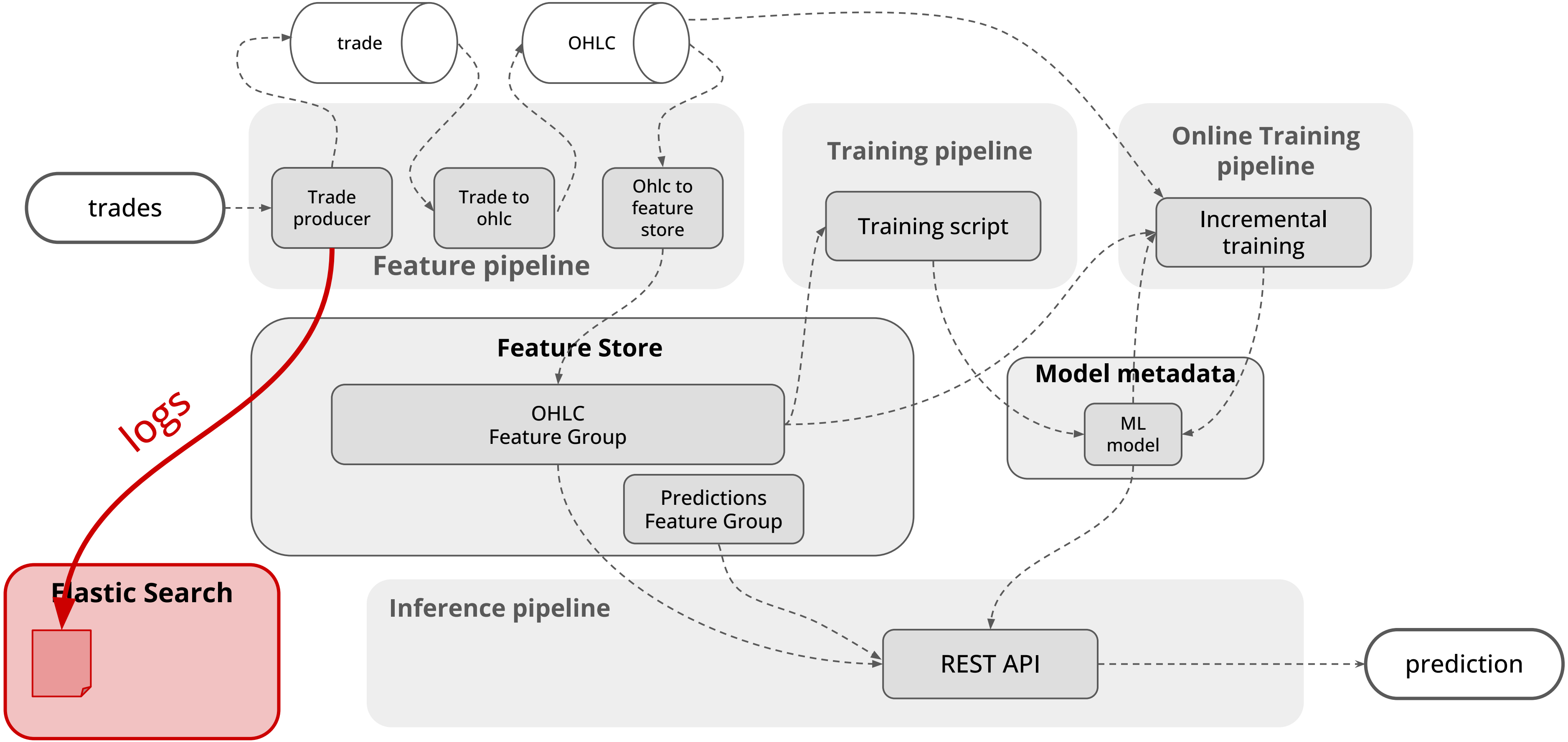


Monitoring

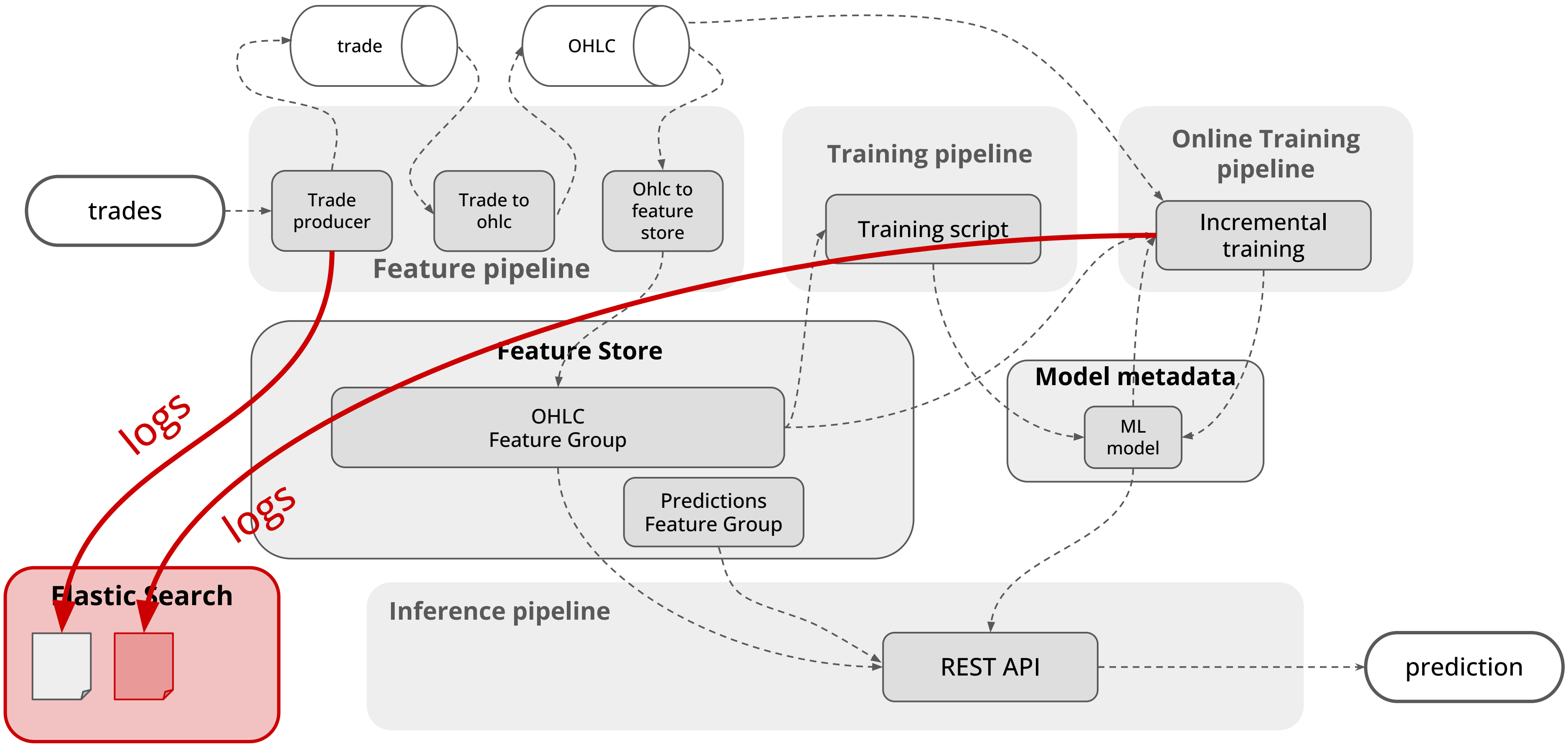
Add Elasticsearch cluster



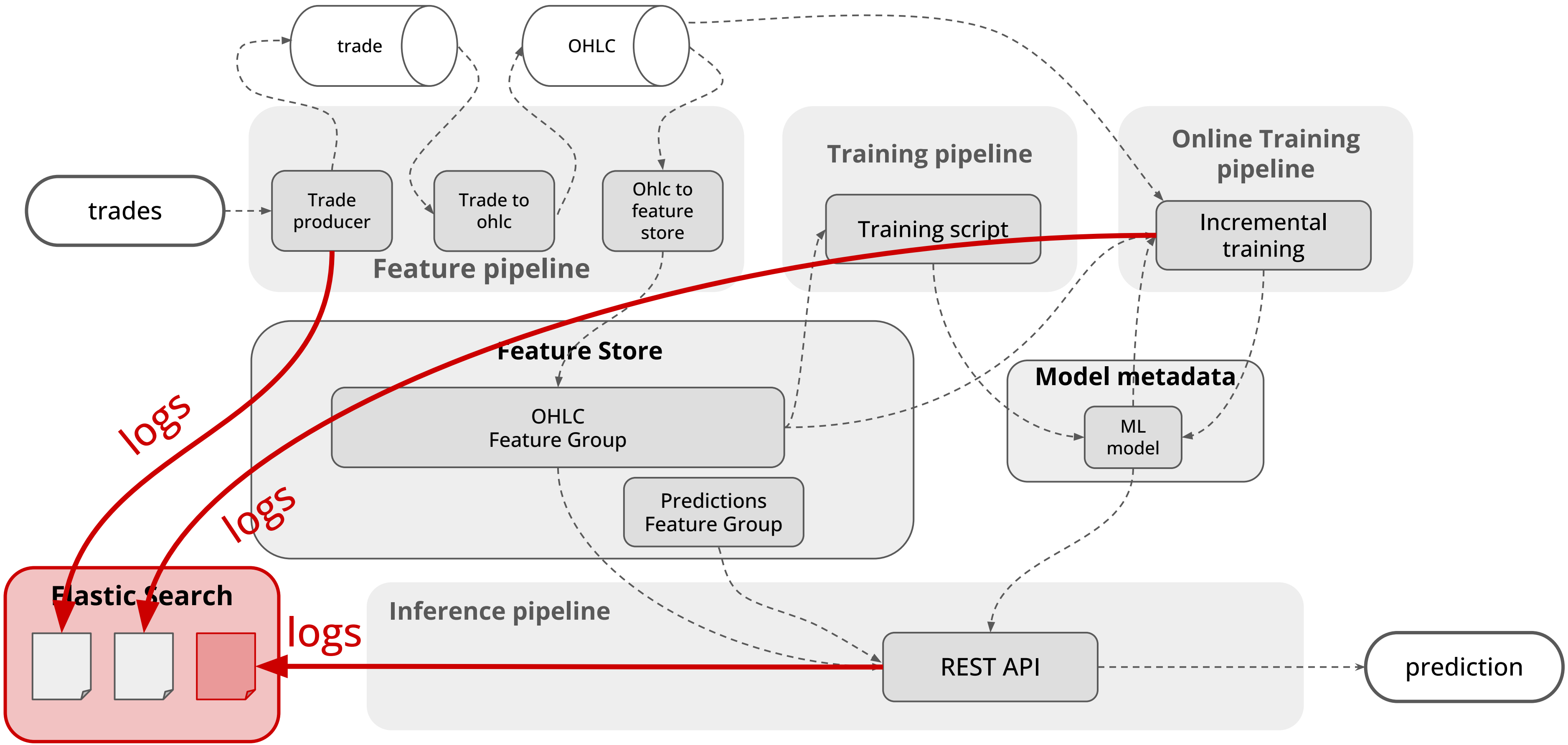
Logs for `trade_producer`



Logs for `incremental_trainer`

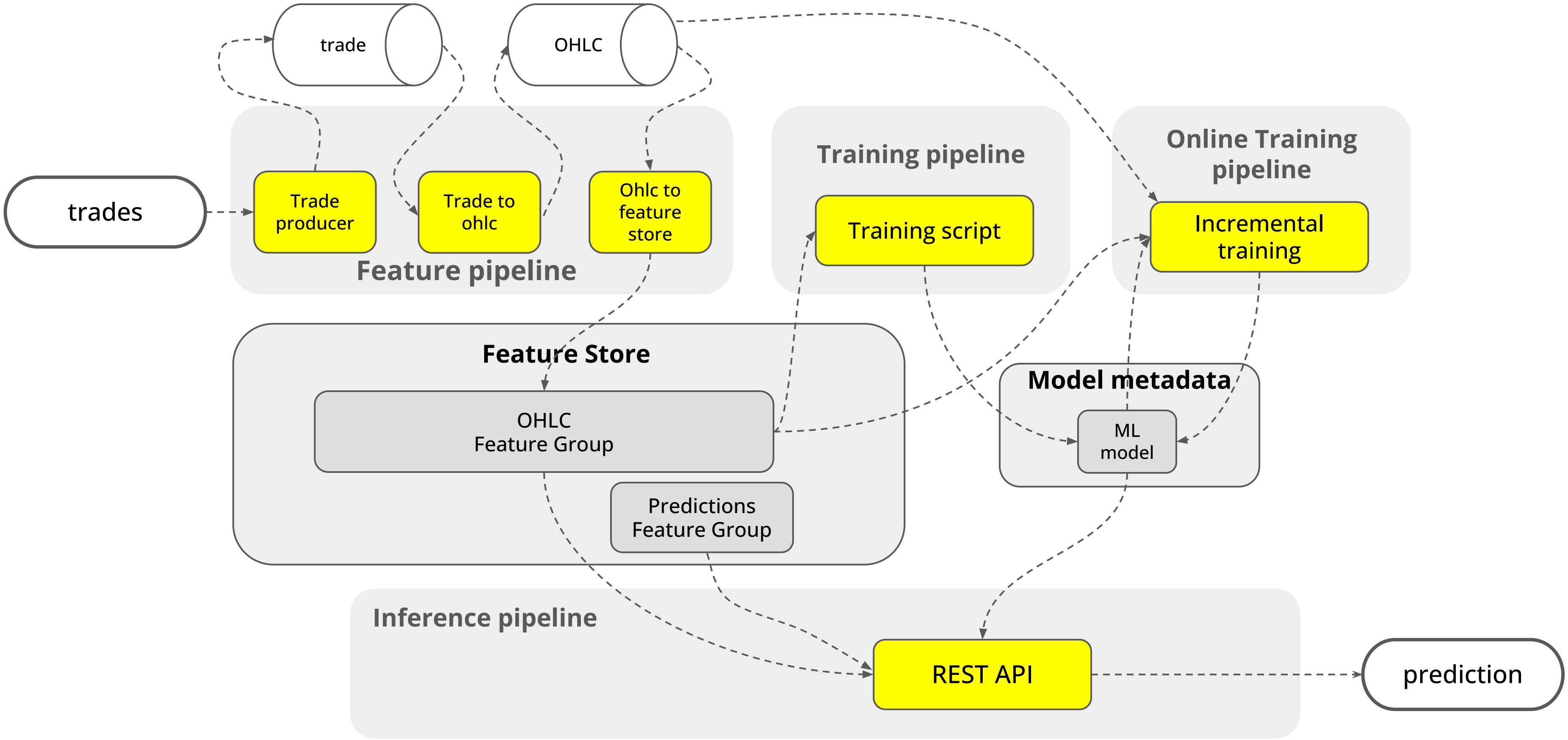


Logs for `REST API`

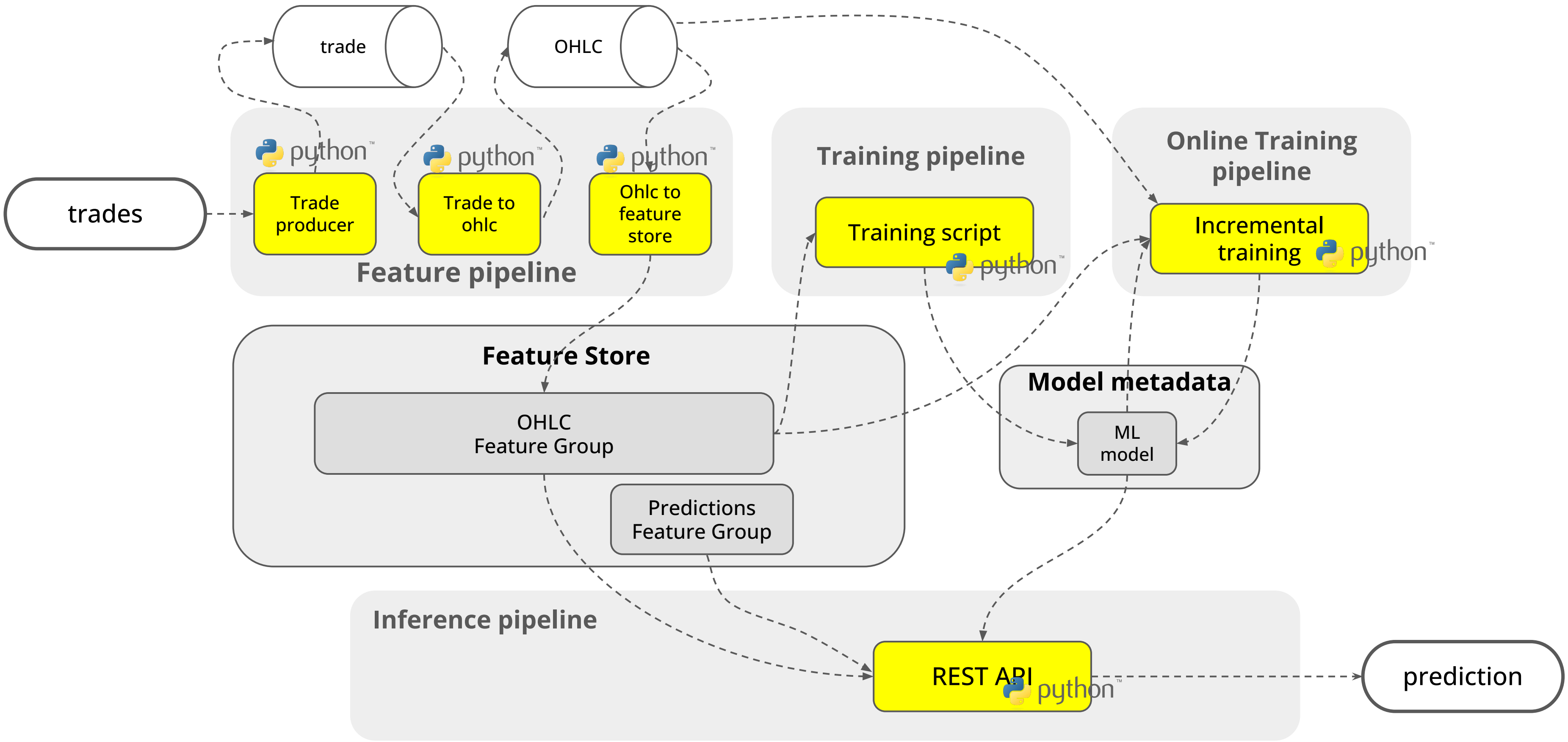


Production ready
from DAY 1

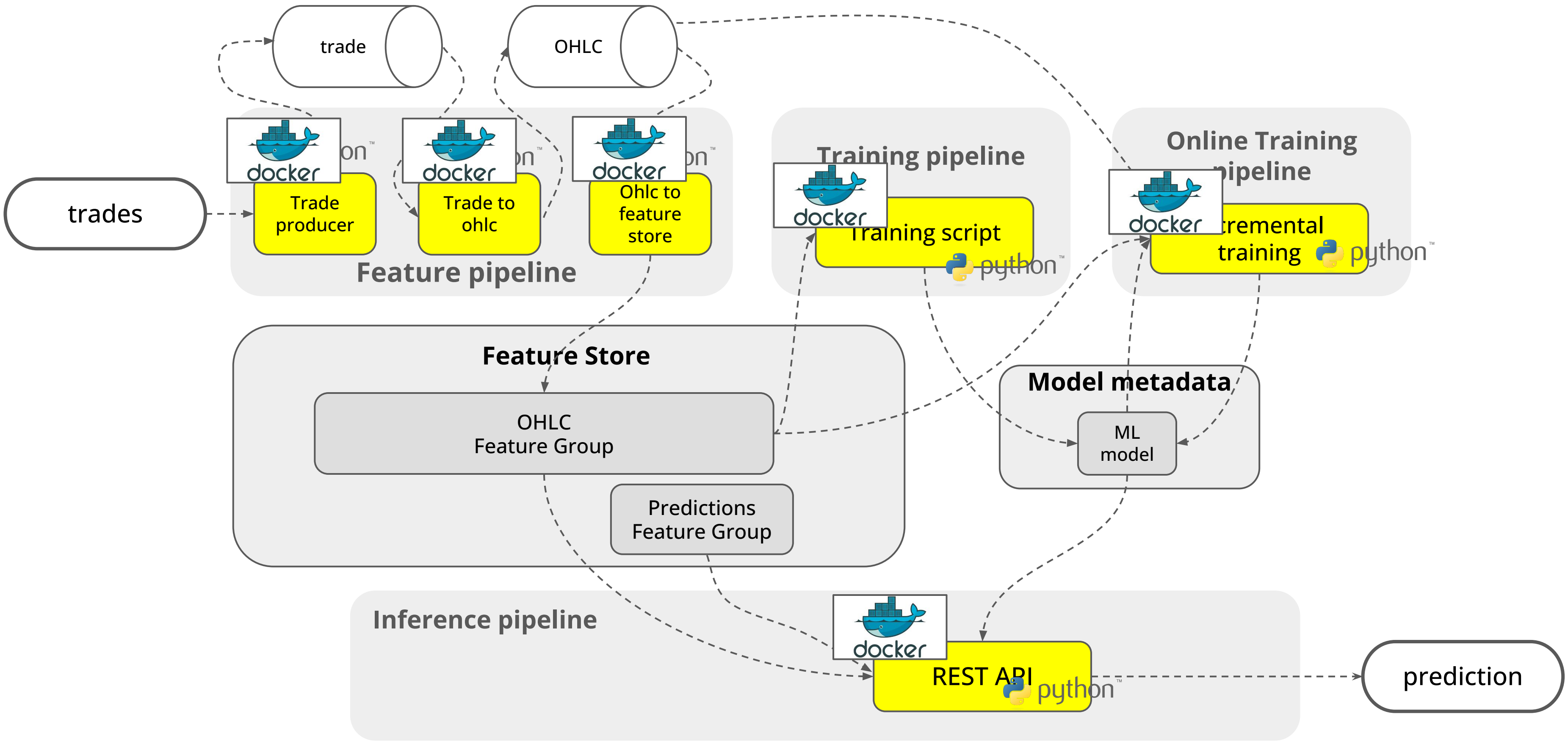
6 microservices



in Python

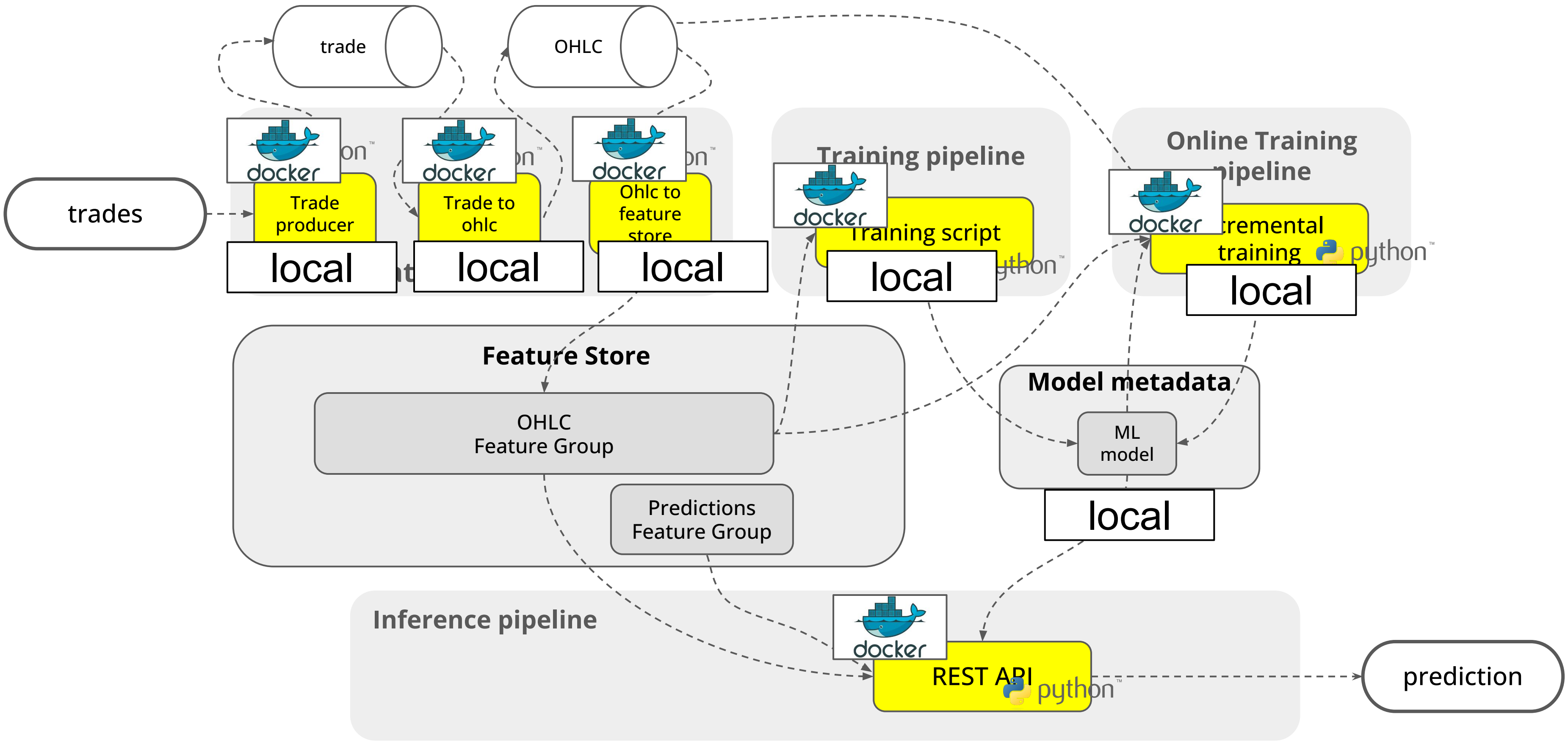

trade
OHLC
trades
Trade producer
Trade to ohlc
Ohlc to feature store
Feature pipeline
Training pipeline
Training script
Online Training pipeline
Incremental training
Feature Store
OHLC Feature Group
Predictions Feature Group
Model metadata
ML model
Inference pipeline
REST API
prediction

Dockerized

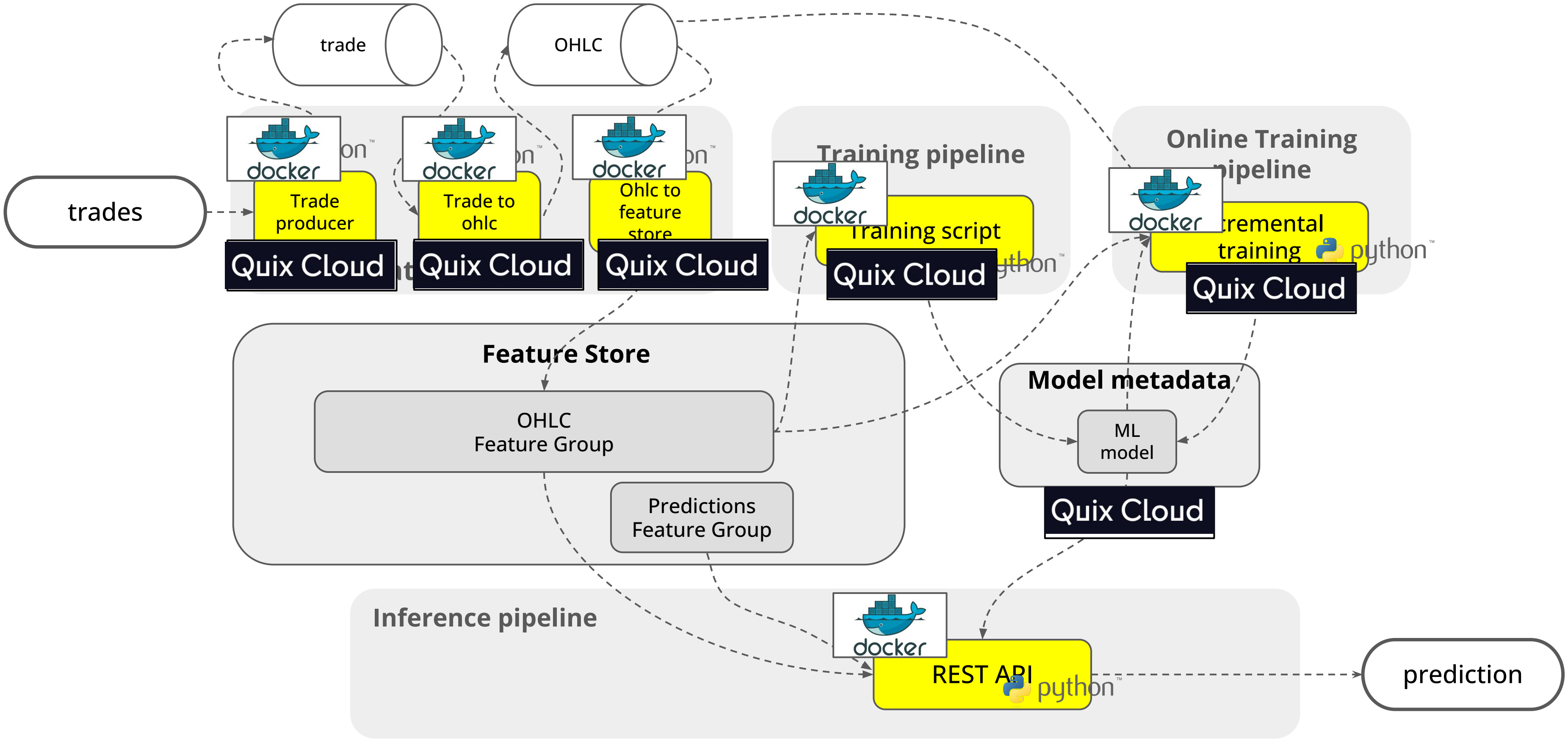


trade
OHLC
trades
docker
Trade producer
Trade to ohlc
Ohlc to feature store
Feature pipeline
Training pipeline
Training script
python
Online Training pipeline
Incremental training
Feature Store
OHLC Feature Group
Predictions Feature Group
Model metadata
ML model
Inference pipeline
REST API
prediction

Develop locally

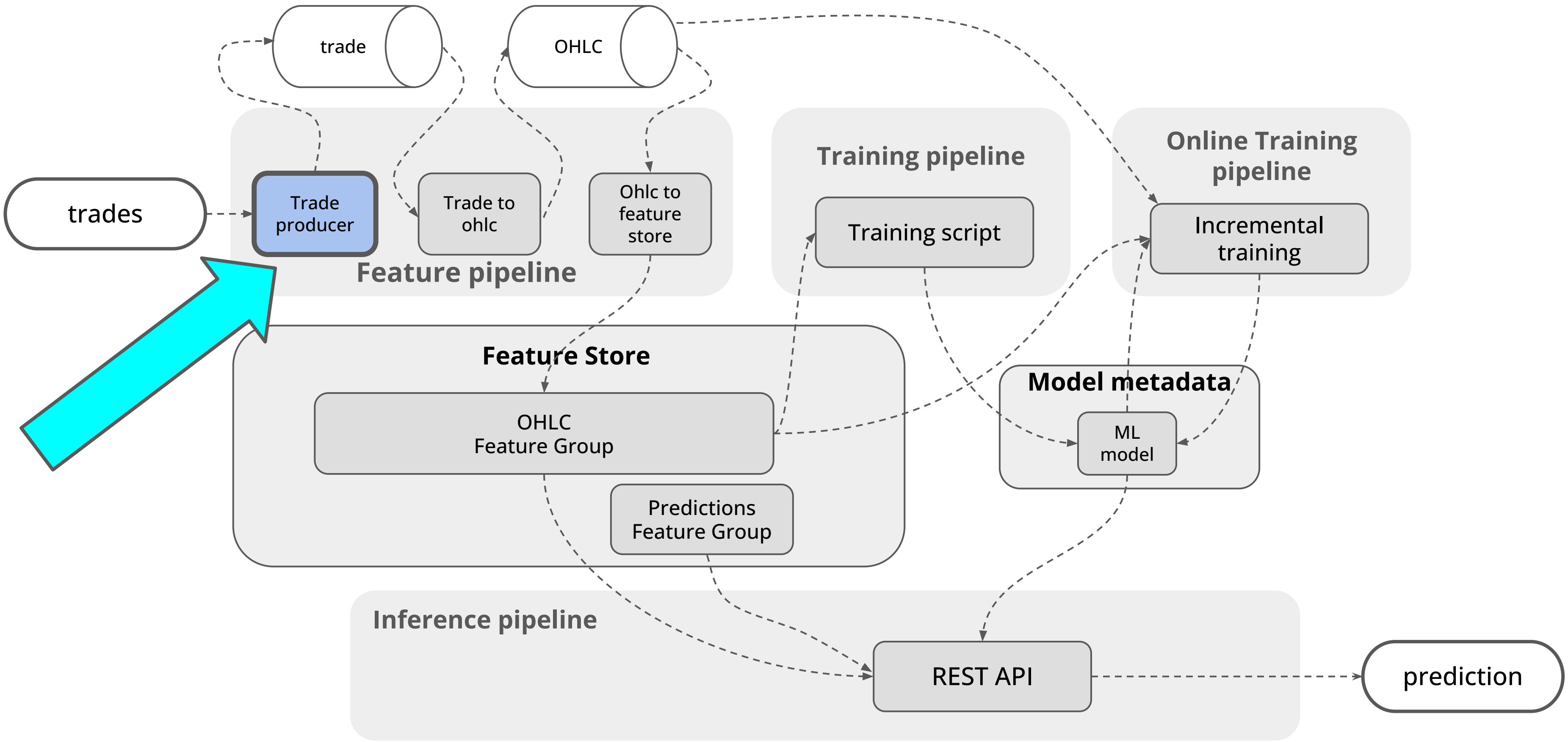


Deploy to Quix Cloud

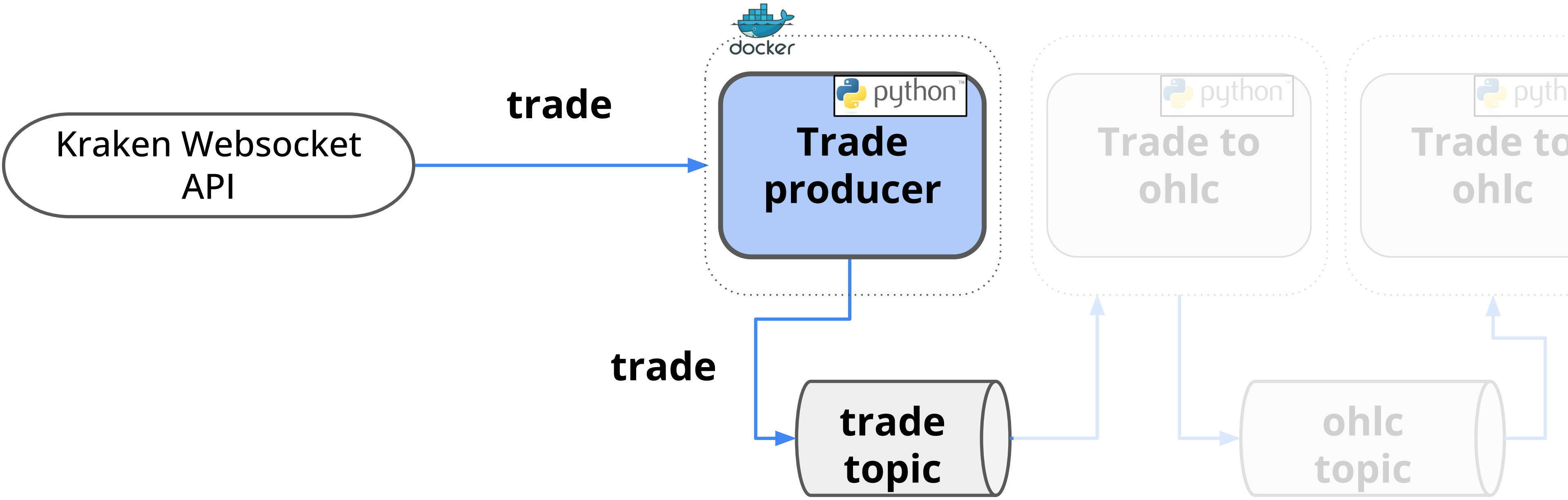


Let's start building!

This is where we start



Today we will build the **trade producer**



On Thursday we will build the other 2 steps