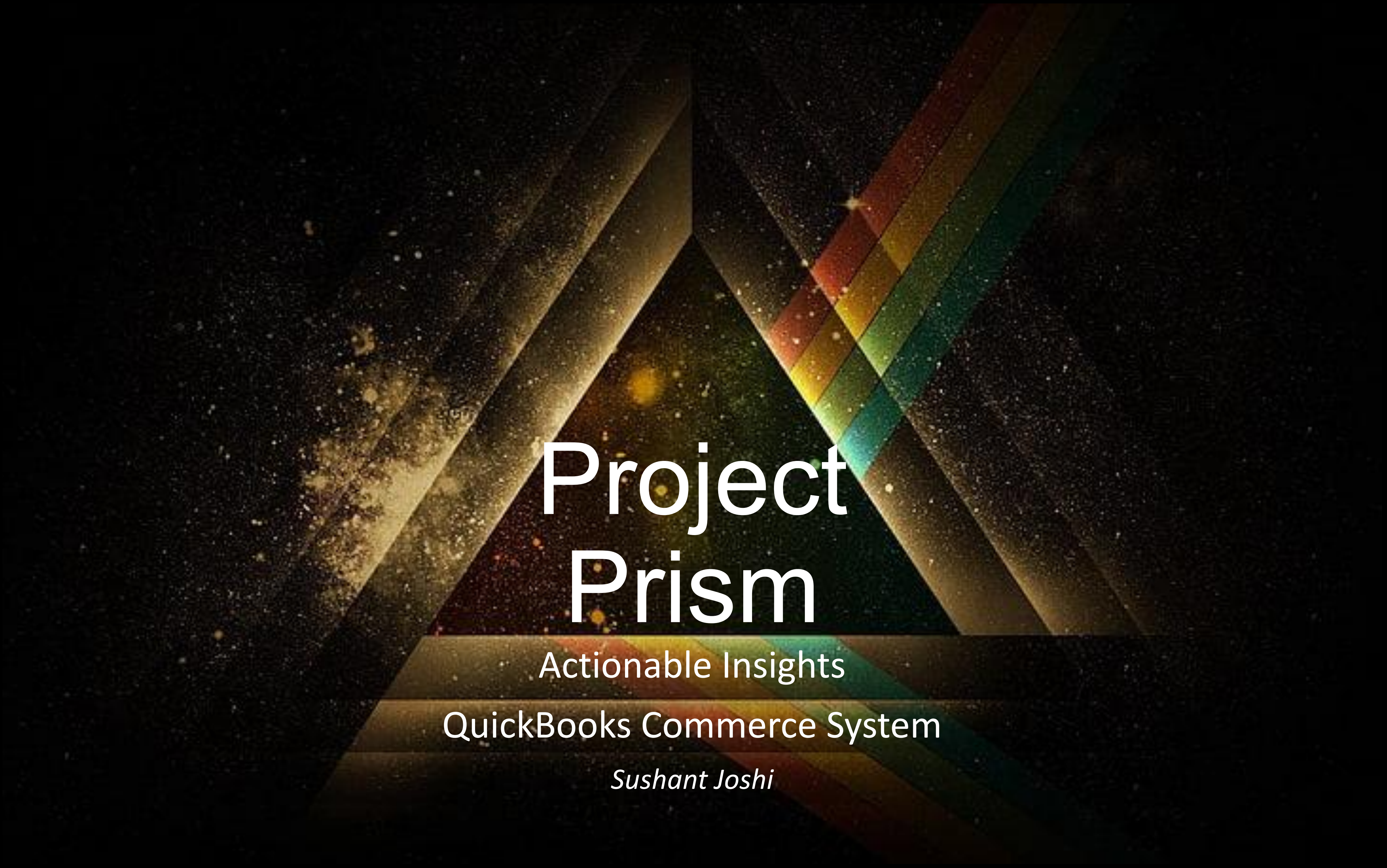


Project Prism

Actionable Insights

QuickBooks Commerce System

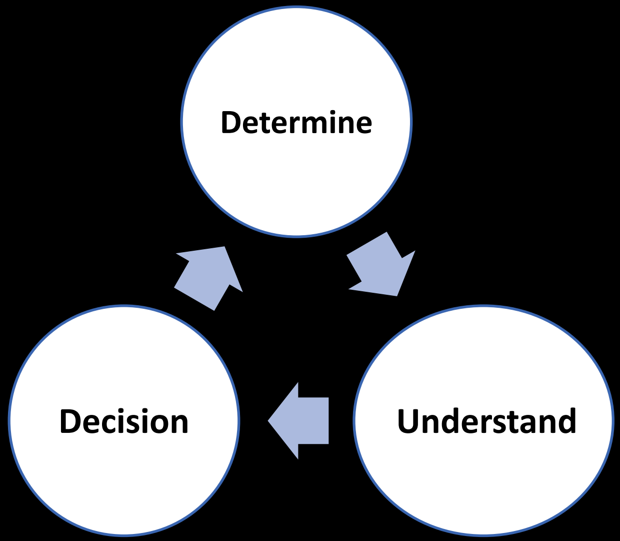
Sushant Joshi

Agenda

- Problem Statement
- The Product Managers Hat
- Requirements
- Measure of Success
- Architecture
- High Level Design
    - Data Ingestion
    - Data Storage
    - Prediction Engine
    - Serving Layer
    - Dashboard
- Operational Excellence
- Application Cost
- Appendix

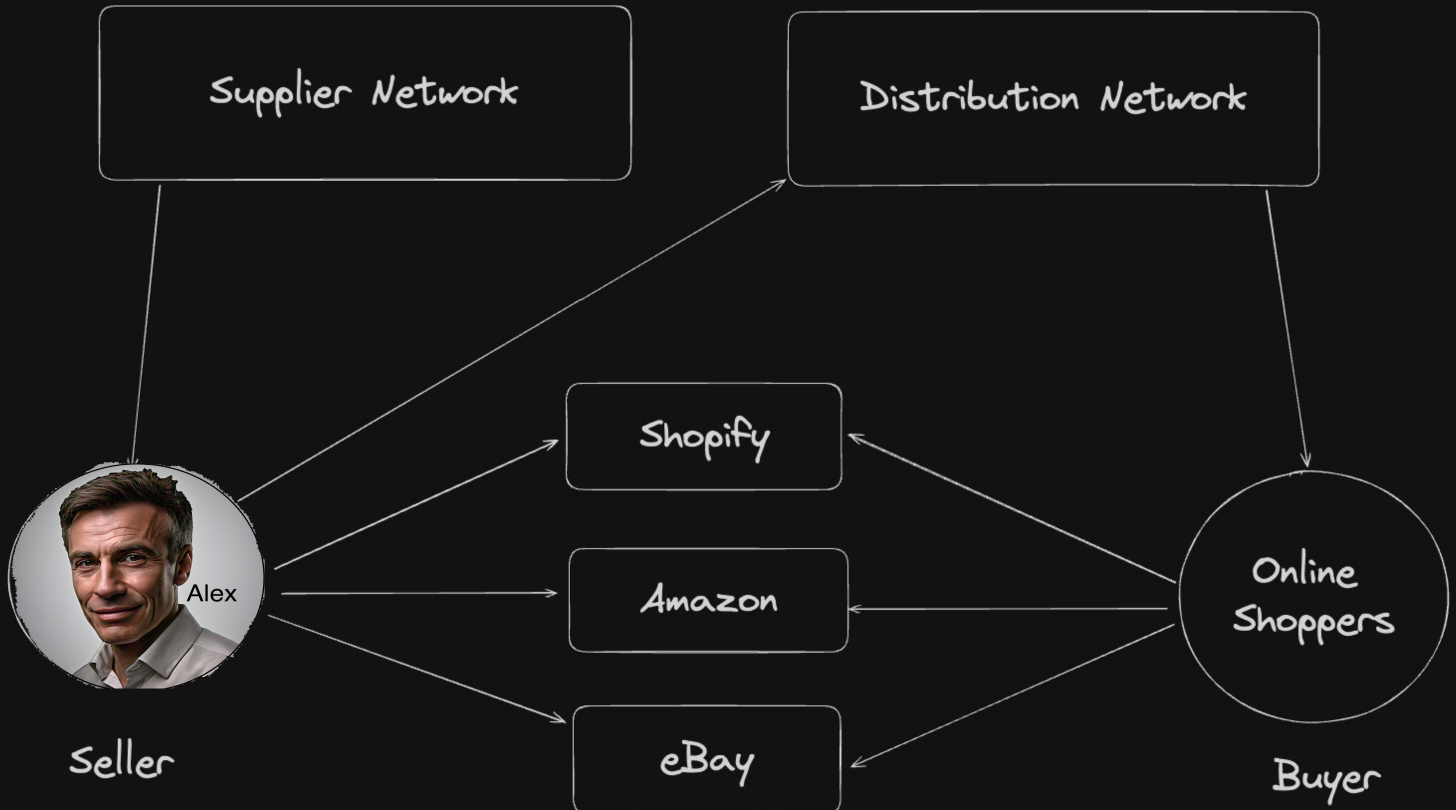
Problem Statement

“QuickBooks Commerce users need an efficient way to forecast their top-selling products and amounts across different categories. This feature should help users to better prepare and adjust their business strategies accordingly.”



The Product Managers Hat

Alex's story



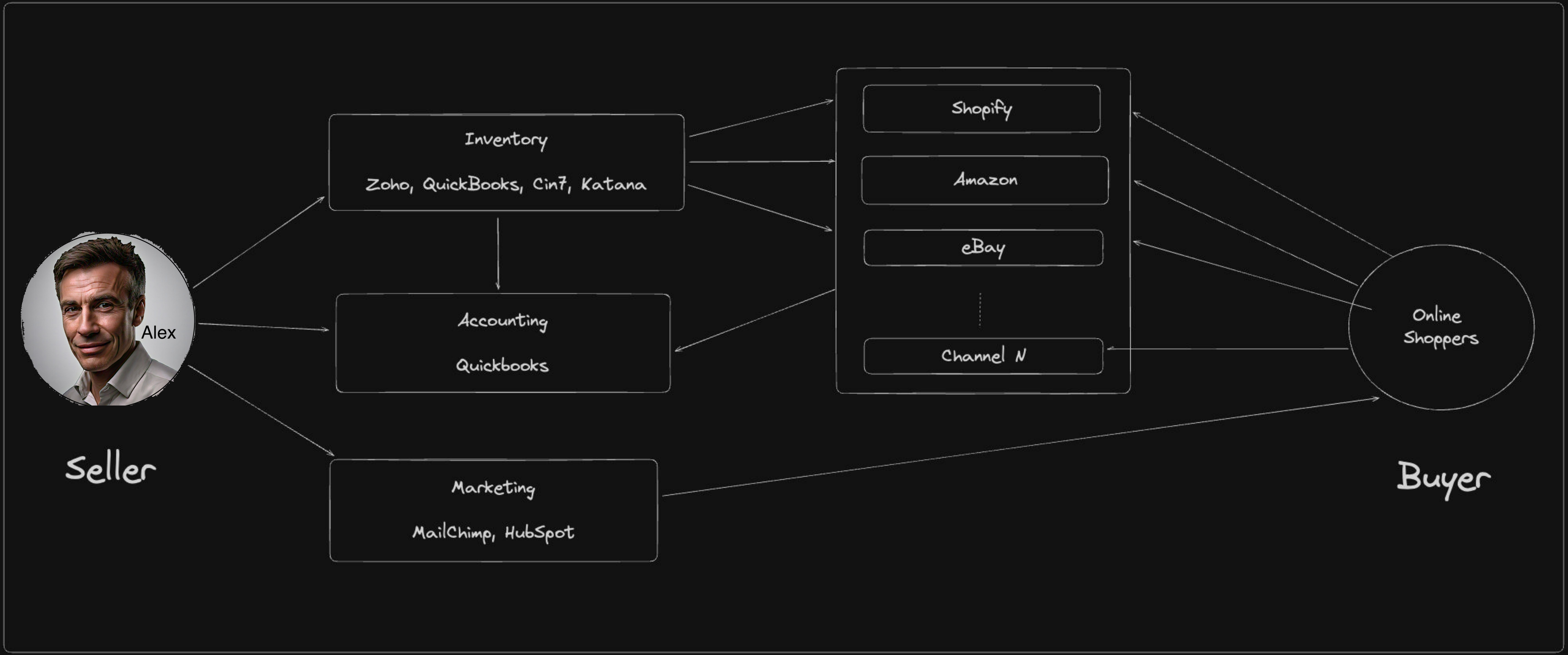
Merchant's life cycle

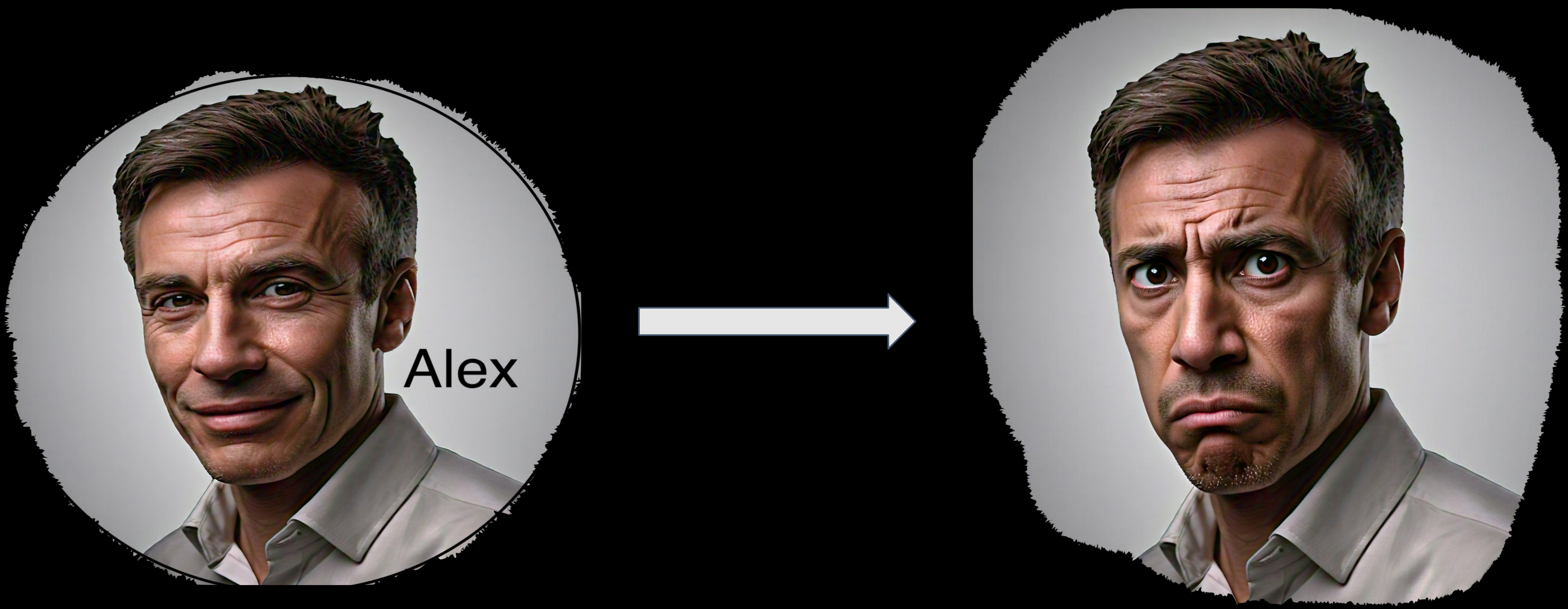
- Onboarding on Shopify, Amazon, Etsy, eBay, Walmart, etc - too many options.
- Setting up Shop / Online Catalog / Pricing / Segments / Checkout and Payment Gateways
- Scaling Operations
- Access to multi-channel selling
- Driving the conversion
- Order Management and Fulfilment
- Abandoned Carts and Returns

Merchants Deeply Care About

- Establishing their brand
- Easily reaching customers and cohorts
- Growth
- Retention

Alex’s (growth) story




Alex

Alex’s top of mind

- How much inventory should I plan for upcoming Valentines Day ?
- Which categories and products should I focus on ?
- What channels are driving the most revenue and profits ?
- What should my marketing strategy be for the next quarter ?
- How do I bundle my products for maximizing revenue ?
- How do I reach my target customer base for my product lines ?
- …

Product Requirement Document

Proposal: A sales forecasting feature within QuickBooks Commerce that predicts future sales based on historical data, trends, and seasonality. This will enable users to:

- **Proactively manage inventory:** Optimize stock levels to meet demand and minimize holding costs.
- **Identify growth opportunities:** Spot trends and capitalize on popular product categories.
- **Improve business planning:** Make informed decisions about pricing, marketing, and resource allocation.

MVP Scope:

- **Limited User Base:** Initial launch to 1,000 selected sellers across the US representing diverse business sizes and industries.
- **Core Forecasting:** Focus on predicting top-selling products and quantities across major categories for the next 3 months.
- **Basic Visualization:** Simple charts and graphs to display forecasted sales data.
- **Feedback Mechanism:** In-app surveys and feedback forms to gather user input for iterative improvements

PRD (contd)

Scaling Plan:

- **Phase 1:** Expand to 100,000 US sellers based on initial MVP feedback and performance data. (3 months post-MVP)
- **Phase 2:** Scale to 1 million US sellers, incorporating advanced forecasting features like custom date ranges and seasonality adjustments. (6 months post-Phase 1)
- **Phase 3:** Full rollout to 20 million US sellers, with continuous optimization and feature enhancements based on user feedback and market trends. (12 months post-Phase 2)

Success Metrics:

- **Feature Adoption Rate:** Percentage of eligible users actively utilizing the sales forecasting tool.
- **Forecast Accuracy:** Measure the deviation between predicted and actual sales.
- **User Satisfaction:** Track user feedback and satisfaction scores related to the forecasting feature.
- **Inventory Efficiency:** Monitor improvements in inventory turnover and stockout reduction.

This phased approach allows for iterative development, data-driven optimization, and a scalable solution that meets the needs of millions of QuickBooks Commerce users.

Domain Deep Dive

Inventory

- Seasonality
- Channel-wise distribution
- Geography
  - State, City, Zip Code
  - Country & Region
- Holiday events
  - Thanksgiving, CMBF
  - Family Oriented Events
  - Labor Day, Memorial Day
  - Chinese New Year
- Special Events
  - NFL, Superbowl
- Supply Chain Linearity
- Forward Deployment
- Dropship vs Direct to Store
- Supply Imbalances

Pricing

- Optimize for Profit
- Optimize for Revenue
- Excess Supply
- Discounts
- Promotions
- Deals
- Bundles
- Dynamic Pricing
  - Competitor Pricing
- Extreme Weather
  - Events Hurricanes, Storms
- Strikes
- Variances in Holidays
  - Offsets
  - Thanksgiving
- Cannibalization

Marketing Strategies

- Advertisements
  - Display Ads
  - Sponsored Ads
- Automated emails
  - Sign-Ups, Abandoned Carts
- Campaigns
- Newsletters
- A/B Testing
- Social Media Posts
- Promotional Mails
- Manufacturer Discounts
- Personalized Messaging
- Customer Understanding
- Channel Understanding
- Rate Limiting
- Engagement

Functional Requirements

From PRD: Focus on predicting top-selling products and quantities across major categories for the next 3 months.

- Dashboard requirements
    - Users should be able to sign in to a portal integrated with QuickBooks Commerce
    - Users should be able to view the inventory forecast of top selling product over a given time period
    - Users should be able to view the top performing categories
    - Users should be able to view the top performing products within given categories
    - Users should be able to select different time ranges for the dashboard - 1 week, 1 month, 1 quarter, 1 year
    - Dashboards should provide a Role Based Access
- API requirements
    - Design a scalable API microservice that will power the above dashboards.
    - API should provide inventory predictions for the top selling product over a given time period (3 months) for the authenticated user.
        - Given a sellerId and channelId, get top most category for channel, then get top selling product for that category, and finally get inventory predictions for that product for the next 3 months
    - API should return predictions that take into account Historical Trends, Seasonality, Holidays and Channels.
    - API should return daily level forecast data for the given product
        - Input forecast range: 1 week – Return: 7 days of forecast
        - Input forecast range: 1 month - Return: 30 days of forecast
        - Input forecast range: 1 quarter - Return: 90 days of data
        - Input forecast range: 1 year - Return: 365 days of data
    - API should be integrated with experimentation platform to enable A/B experimentation involving model and feature improvements.
    - API should be secure and should only return relevant sellers’ forecasts.

Non-Functional Requirements

Scalable

- Supports 10s of millions of sellers across geographies
- Supports 10s of channels
- Supports 1000s of products across 10s of categories per seller

Performant

- Low Latency
- Dashboards load up in milliseconds
- Reactive to changes in data patterns
- SLA: 250 msec @ 95th percentile

Available

- System as a whole is Highly Available
- Redundancies carefully crafted
- Stateless and loosely coupled

Fault Tolerant

- Handles Systemic Failures
- Handles Data Delays
- Provides Fail-Safe Mode of Operations

Secure | Cloud Native | Cost Sensitive

Architecture

Blueprint

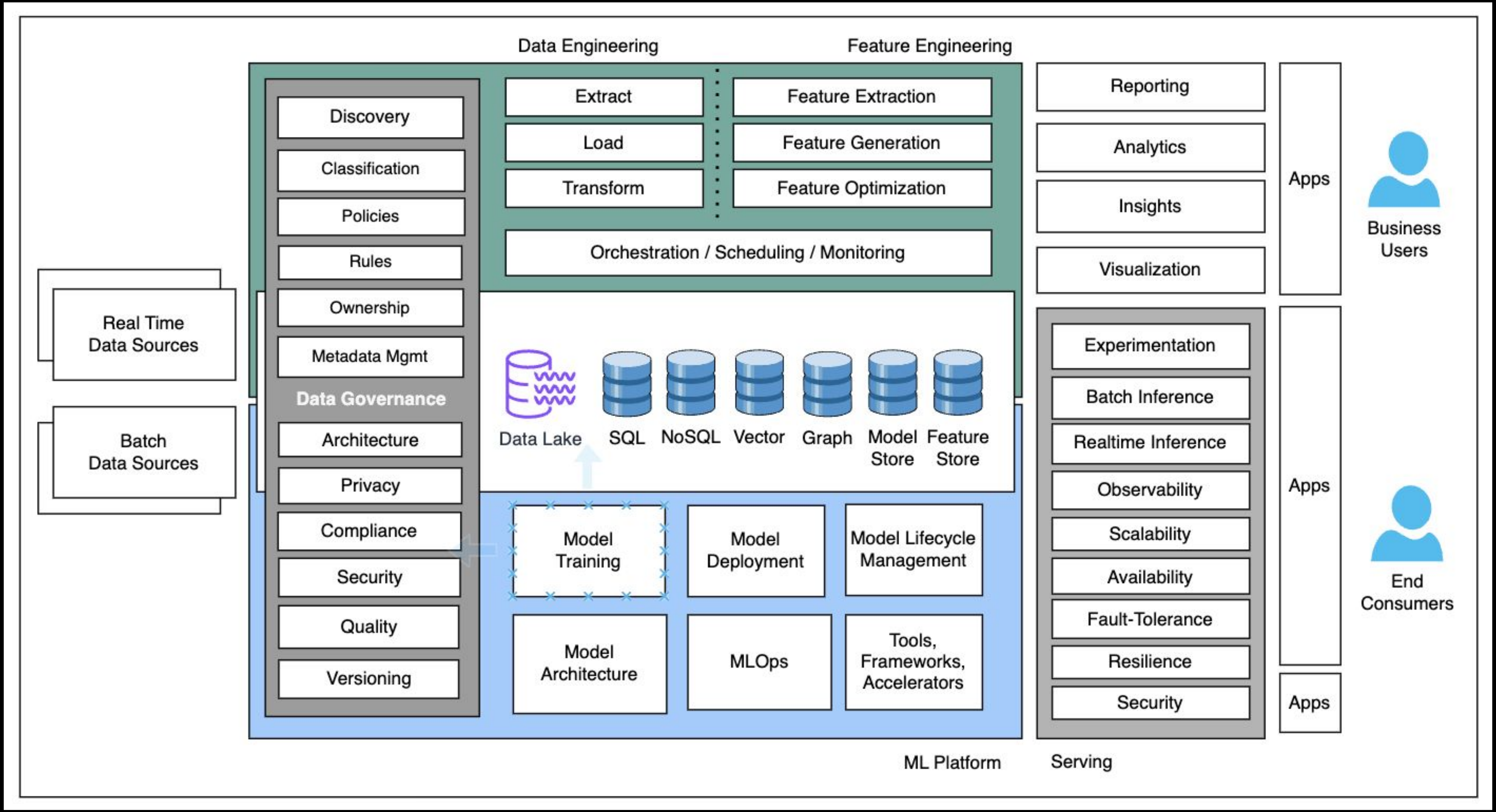


Fig: Architecture Blueprint

Data

Data

- Type
  - Transactional Data
  - Order Data
  - Invoice Data
  - Online Data
  - Store Data
  - Payment Data
  - POS Data
  - Click Stream Data
  - Sales Reports
    - Structured or Unstructured

- Data Access Pattern
  - Push Based
  - Pull Based
  - Drop Box
  - API
  - Messaging
- Data Issues
  - Delayed Data
  - Missing Data
  - Lost Data
  - Duplicate Data
  - Garbage/Invalid Data

Key Data Attributes

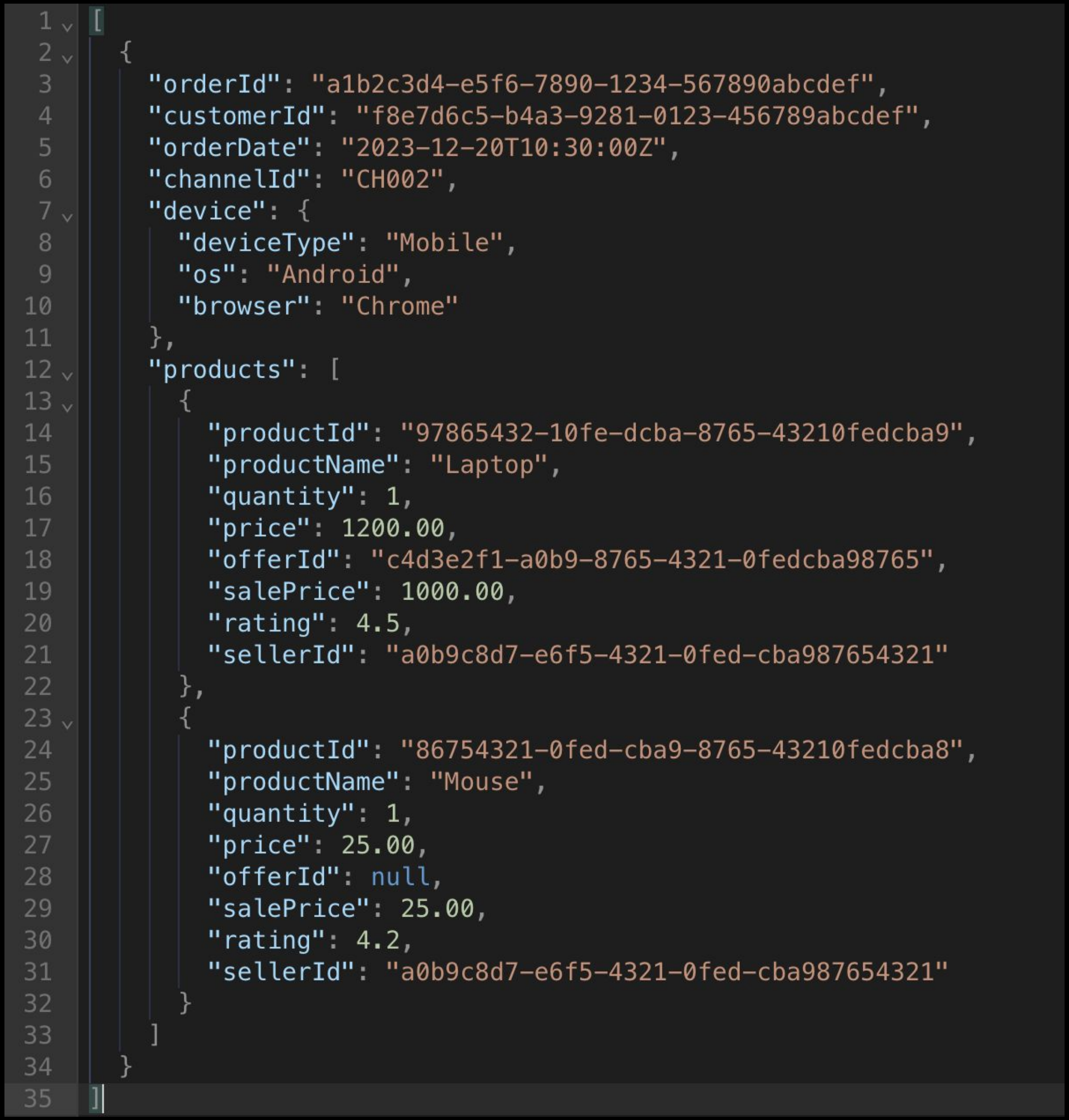
- Date
  - Transaction Date
  - Order Date
  - Invoice Date
  - Delivered Date
  - Returned Date
- Order
  - Order Id
  - Total Order Value
- Channel
  - Channel Id
  - Channel Name
- Subscription
  - SubscriberID
- Device
  - IP Address
  - Device Type

- Item
  - Item Id
  - Item name
  - Item Price
  - Item Description
  - SKU Number
  - UPC
  - Offer Id
  - Bundle Id
  - Variant Information
  - Unit Count
  - Product URL
  - Ratings
  - Reviews
  - Price
    - Item Price
    - Sale Price

- Category
  - Category Id
  - Category Name
  - Sub Category Id
  - Sub Category Name
  - Category Hierarchy
- Customer
  - Customer Id
  - Customer Name
- Geographical
  - City
  - State
  - Country
  - Zip
  - Latitude
  - Longitude

Sample Order Payload

data ingestion



```json
[
  {
    "orderId": "a1b2c3d4-e5f6-7890-1234-567890abcdef",
    "customerId": "f8e7d6c5-b4a3-9281-0123-456789abcdef",
    "orderDate": "2023-12-20T10:30:00Z",
    "channelId": "CH002",
    "device": {
      "deviceType": "Mobile",
      "os": "Android",
      "browser": "Chrome"
    },
    "products": [
      {
        "productId": "97865432-10fe-dcba-8765-43210fedcba9",
        "productName": "Laptop",
        "quantity": 1,
        "price": 1200.00,
        "offerId": "c4d3e2f1-a0b9-8765-4321-0fedcba98765",
        "salePrice": 1000.00,
        "rating": 4.5,
        "sellerId": "a0b9c8d7-e6f5-4321-0fed-cba987654321"
      },
      {
        "productId": "86754321-0fed-cba9-8765-43210fedcba8",
        "productName": "Mouse",
        "quantity": 1,
        "price": 25.00,
        "offerId": null,
        "salePrice": 25.00,
        "rating": 4.2,
        "sellerId": "a0b9c8d7-e6f5-4321-0fed-cba987654321"
      }
    ]
  }
]
```

The Three Vs - Velocity, Variety, Volume

Velocity

- Continuous Stream
  - Real Time
  - Near Real Time
- MicroBatches
  - Accumulated Data
- Batches
  - Hourly Data
  - Daily Data
  - Weekly Data
  - Monthly Data
- Irregular Data Flow

Variety

- Structured
  - CSV
  - JSON
  - AVRO
  - XML
- Unstructured
  - TEXT
  - IMAGES
  - PDF
- Compressed
- Encoded
- Encrypted

Volume

(next slide)

Volume

Estimates

- Number of sellers - 20 Million
- Number of channels - 3 (AMAZON, eBAY, SHOPIFY)
- Number of products sold / seller / day - 100 (Avg) across 3 channels
- Each order payload - 1 kilobyte raw uncompressed

Daily Data Generation

- Daily Orders
  - 20,000,000 sellers * 100 orders/seller/day
  - 2,000,000,000 orders/day
- Daily Volume
  - 2,000,000,000 orders/day * 1 kilobyte/order * 0.35 CR
  - 700,000,000 kilobytes/day
  - **~ 650 GB / day**

Yearly Data Generation

- Yearly Volume
  - 650 GB/day * 365 days/year
  - **≈ 230 Terabytes /year**

Data Ingestion Considerations

- Retention Policy
  - Raw Data Retention Policy
  - Client Tier Specific Configuration
    - Premier User, Business User, Standard User
- Archival Policy
  - Cold Storage
  - Client Tier Specific Configuration
- Data Compression and Overhead
  - Compression Techniques
  - Data Schema Standardization

- Replication Strategy
  - Optimal replica count
  - Replicas distribution
    - Across Data Center vs Cloud Region
- Sharding / Partitioning
  - Date Level
  - Channel Level
  - Seller Level
  - Nested Hierarchical
- Growth Projections
  - Growth over next 5 years, 10 years, 15 years…

High Level Design

HLD

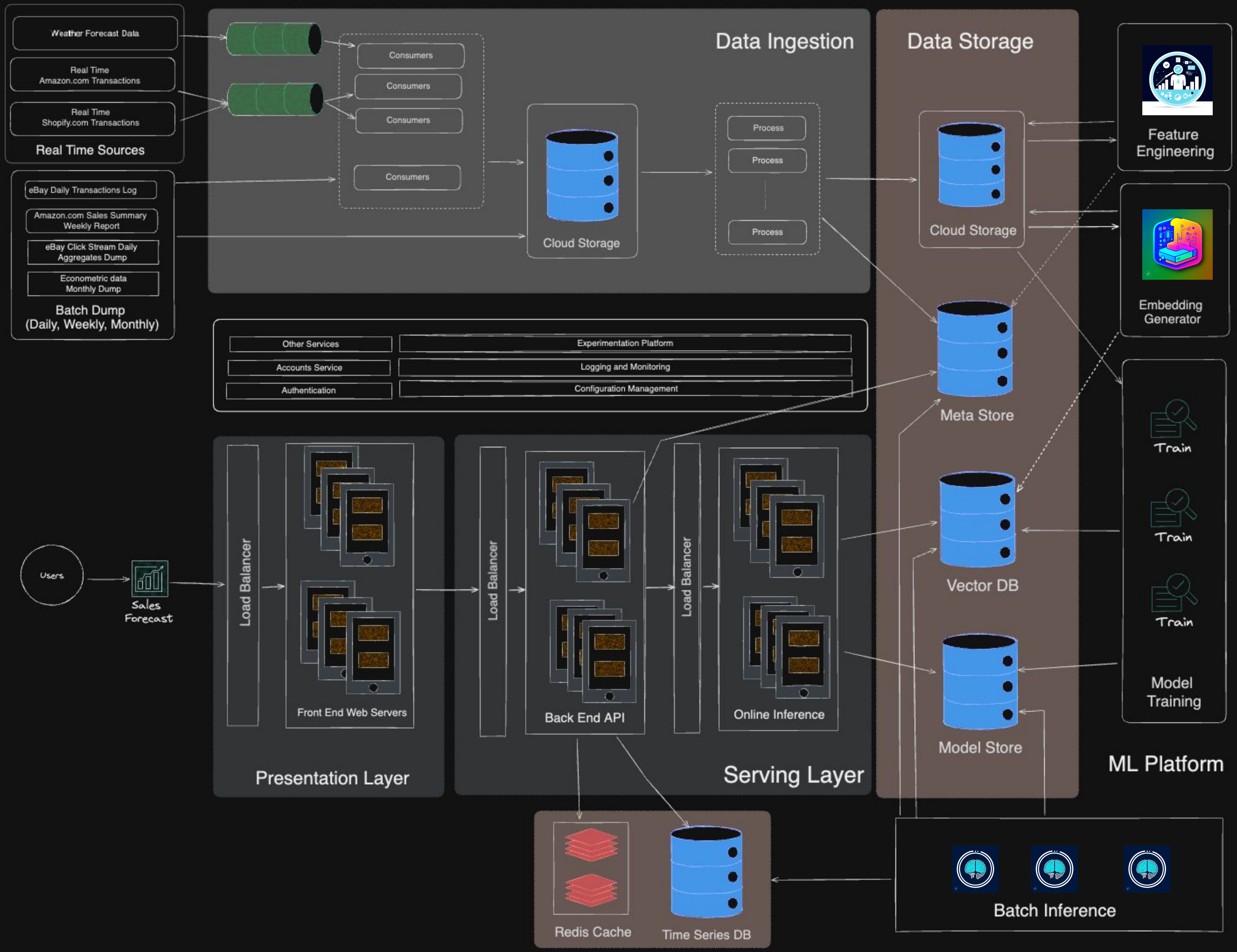
Data Ingestion

Data Storage

ML Platform

Serving Layer

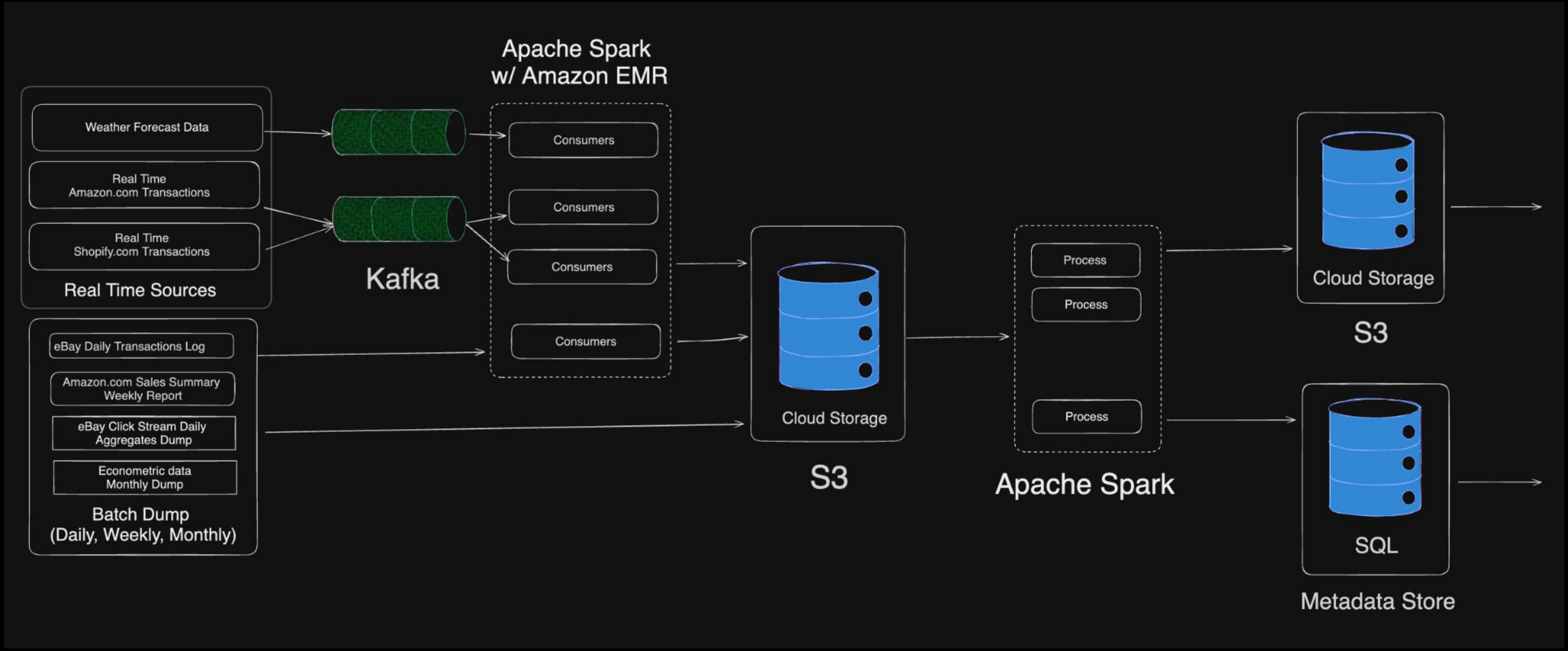
Presentation Layer



1. Data Ingestion

2. ML Platform
3. Serving Layer

Data Ingestion



1. Data Ingestion
2. ML Platform
3. Serving Layer

Framing the ML Problem

Objective

Develop a machine learning model to accurately forecast the top-selling products and their corresponding sales quantities across various product categories for QuickBooks Commerce users.

Machine Learning Task

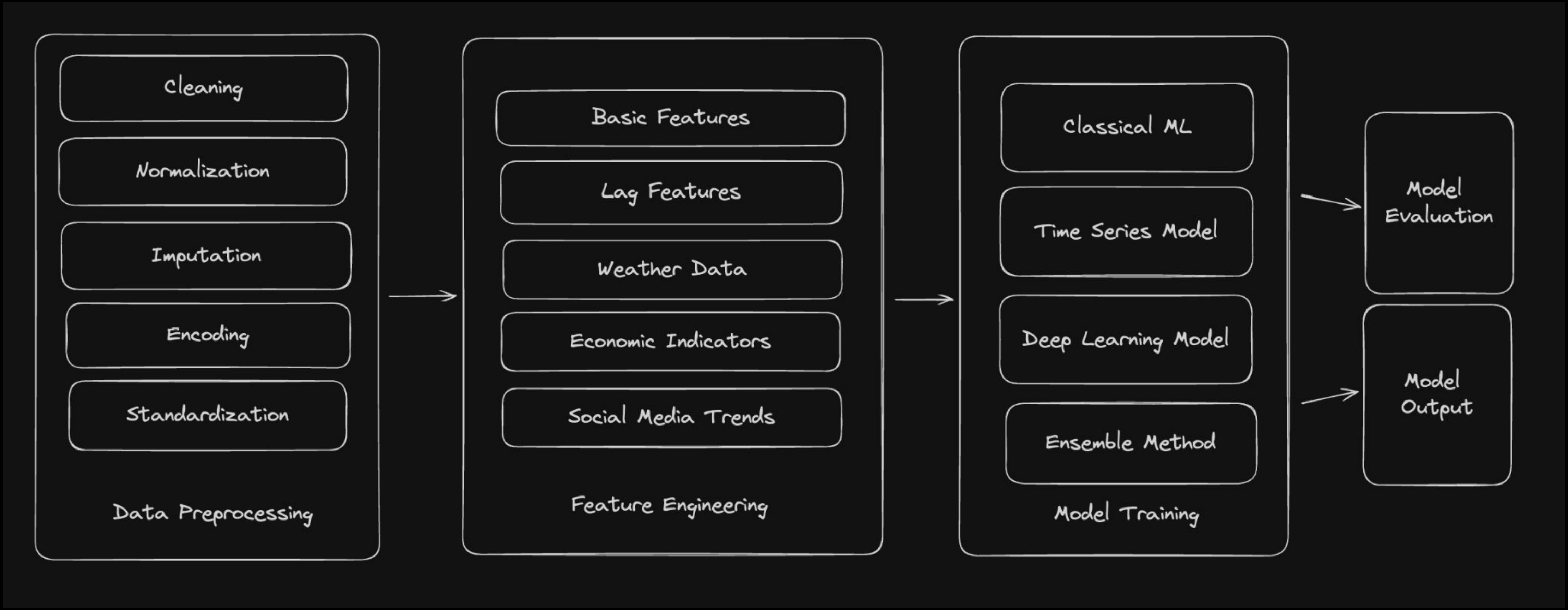
- **Time series forecasting** problem.
  - Predict future values (sales quantity) based on past values and other features.

Feature Engineering

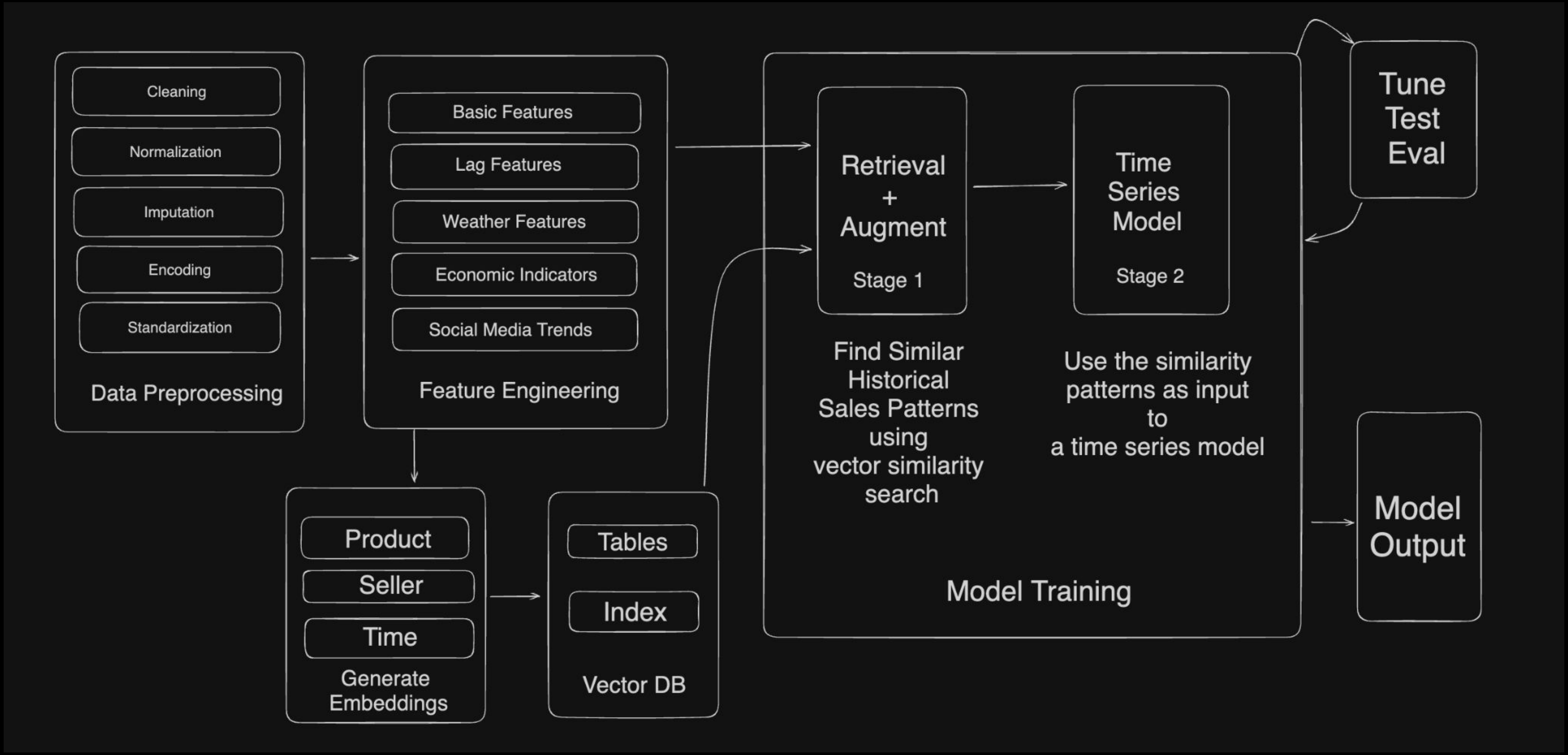
- **Temporal Features**
    - Day of week
    - Week of month
    - Week of the year
    - Month of the year
    - Holidays
    - Seasonality indicators
    - Cyclical patterns (back-to-school)
- **Product Features**
    - Name
    - Description
    - Product Reviews
    - Product Ratings
- **Category Features:**
    - Product categories
    - Product sub-categories
- **Seller Features**
    - Seller Ratings
- **Geographic Features**
    - city
    - state
    - country
    - zip
    - regions
- **Sales Channel Features:**
    - amazon
    - eBay
    - shopify

{ product + category + geography + seller + temporal + … }

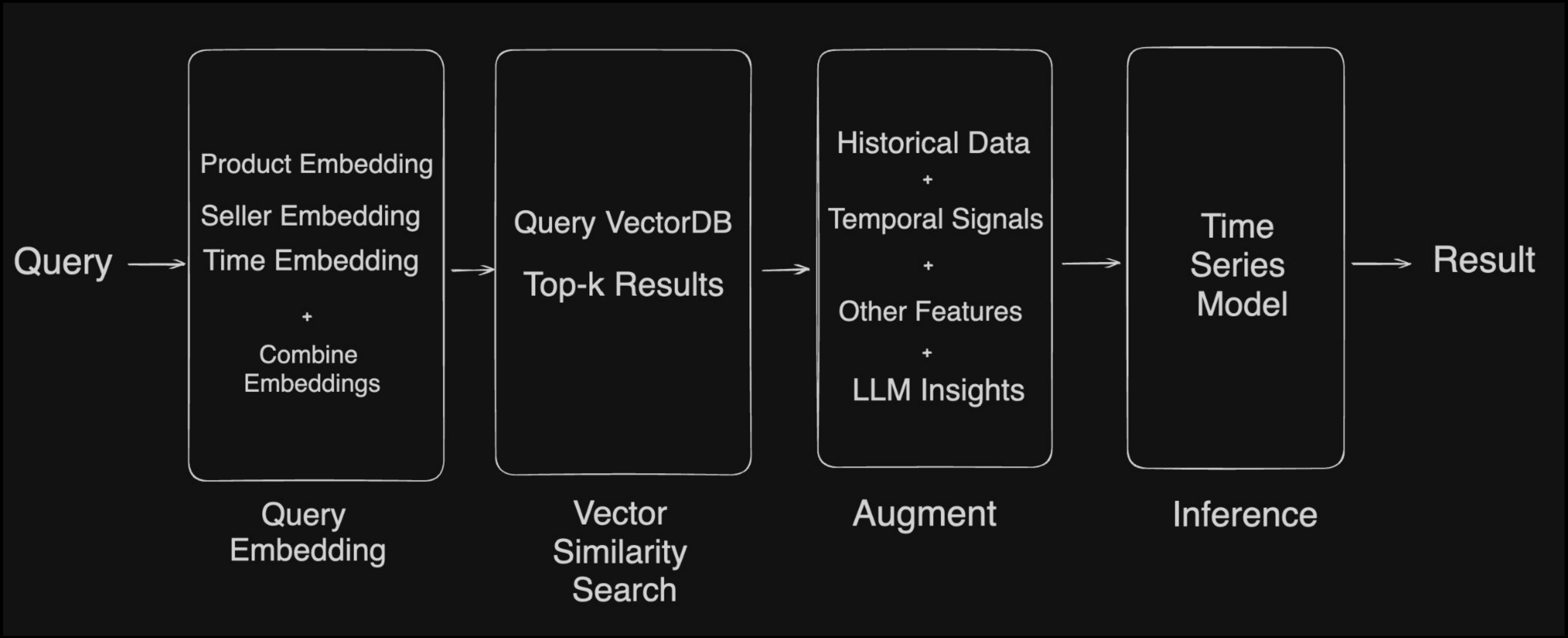
Model Training



Model Training



Inference



Query

Product Embedding

Seller Embedding

Time Embedding

+

Combine Embeddings

Query Embedding

Query VectorDB

Top-k Results

Vector Similarity Search

Historical Data

+

Temporal Signals

+

Other Features

+

LLM Insights

Augment

Time Series Model

Inference

Result

Model Evaluation Metrics

- **Mean Absolute Error (MAE)**
  - Measures the average absolute difference between predicted and actual sales.
- **Root Mean Squared Error (RMSE)**
  - Penalizes larger errors.
- **Mean Absolute Percentage Error (MAPE)**
  - Measures the average percentage error in the predictions.

ML Model Considerations

Ethical Considerations

- Train the model across sellers, channels, geographies ?

Data Sparsity

- Products might have sparse and limited historical data.
  - Easter Eggs, Christmas Trees, Halloween Candy
- Flash Sales

Cold Start Problem

- Predicting sales for new products
- Predicting for new sellers on platform

Seasonality and Trends

- Capturing complex seasonal patterns and trends

External Factors

- Incorporating and quantifying the impact of external factors
  - Elections, Wars, Extreme Events, Natural Disasters, Pandemic

Explainability

- Providing users with insights into why the model made certain predictions.

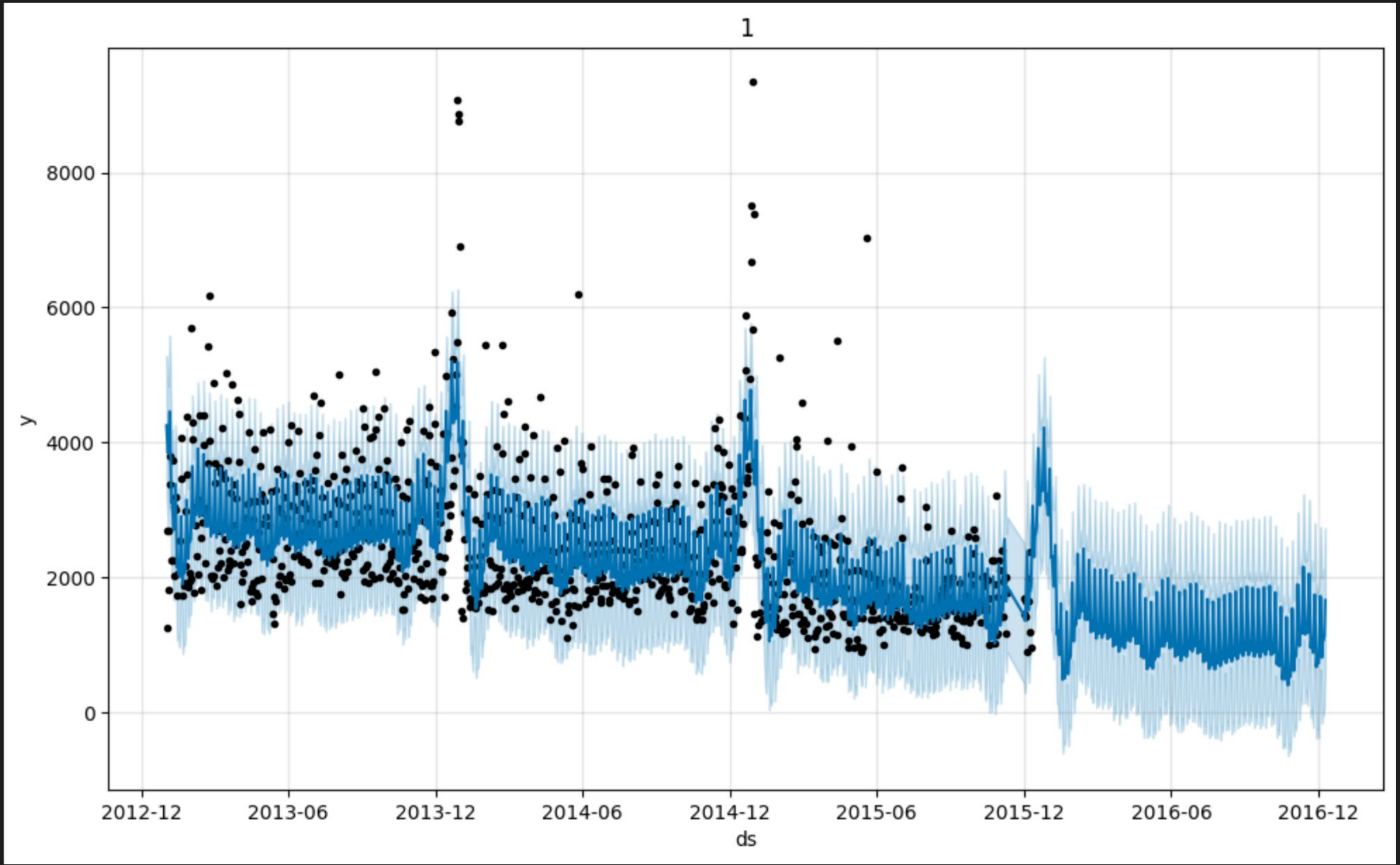


Figure: Sample Plot forecasting next 12 months from model trained using Prophet Time Series Model over 2012 to 2015 dataset

1. Data Ingestion
2. Prediction Engine

3. Serving Layer

API Specification

`Endpoint: GET /v1/predict/top-sales`

Request

Field	Data Type	Description
fRange	string	Forecast range (1d, 1w, 1m, 1q, 1y)
sellerId	GUID	Seller ID

Response

Field	Data Type	Description
result	array	Array of sales predictions for each channel
channelId	string	Channel ID
products	array	Array of products
productId	string	Product ID
productName	string	Product name
categoryName	string	Category name
categoryId	string	Category ID
predictions	array	Array of sales data
date	string	Forecast date (MM-DD-YYYY)
quantity	integer	Sales units

Error Codes

Error Code	Description
400	Bad Request (invalid request body or parameters)
404	Not Found (seller ID or channel not found)
500	Internal Server Error (prediction model error or database issue)

Request

```
{
  "fRange": "1q",
  "sellerId": "123e4567-e89b-12d3-a456-426655440000"
}
```

Response

```
{
  "result": [
    {
      "channelId": "CH001",
      "products": [
        {
          "productId": "12345",
          "productName": "Laptop",
          "categoryName": "Electronics",
          "categoryId": "C001",
          "predictions": [
            {
              "date": "01-10-2025",
              "quantity": 120
            },
            {
              "date": "01-11-2025",
              "quantity": 150
            },
            …
            {
              "date": "03-26-2025",
              "quantity": 70
            }
          ]
        }
      ]
    }
  ]
}
```

API Specification (Additional Endpoints)

1. Get a list of all channels for seller

`Endpoint: GET /v1/sellers`

- Request Parameters:
  - sellerId (query parameter)
  - limit (query parameter, optional, default=3)
- Response: JSON array of all the channels for the sellerId

2. Get Top Categories for each channel

`Endpoint: GET /v1/channels/{channelId}/top-categories`

- Request Parameters:
  - channelId (path parameter)
  - sellerId (query parameter)
  - limit (query parameter, optional, default=1)
- Response: JSON array of top categories for the channel

3. Get Top Selling Products for top 3 categories

`Endpoint: GET /v1/categories/{categoryId}/top-products`

- Request Parameters:
  - sellerId (query parameter)
  - channelId (query parameter)
  - categoryId (path parameter)
  - limit (query parameter, optional, default=1)
- Response: JSON array of top selling products for the category

4. Get Top Selling Products for Seller (Query)

`Endpoint: GET /v1/products`

- Request Parameters:
  - sellerId (query parameter, required)
  - channelLimit(query parameter, optional, default=3)
  - categoryLimit (query parameter, optional, default=1)
  - productLimit (query parameter, optional, default=1)
- Response: JSON array of top selling products for seller

5. Get Inventory Predictions for Product (Forecast)

`Endpoint: GET /v1/products/{productId}/forecast`

- Request Parameters:
  - sellerId (query parameter)
  - productId (path parameter)
  - forecastRange (query parameter, default="1q")
- Response: JSON object with inventory prediction data for the product

Serving Database Selection

Relational Databases (SQL)

- Structured data
- Strong querying capabilities
- Robust tools
- ACID properties

–

- Predefined Schema
- Scaling challenges

Strong Querying and Structured Data

Time-Series Databases

- Optimized for time-series data
- Support time-based queries
- High ingestion rates
- Built-in functions

–

- Limited non-time-series data
- Limited querying flexibility

Optimized for time-series handling

NoSQL Databases

- Semi-structured data
- Schema flexibility
- Simple Querying
- Horizontally scalable

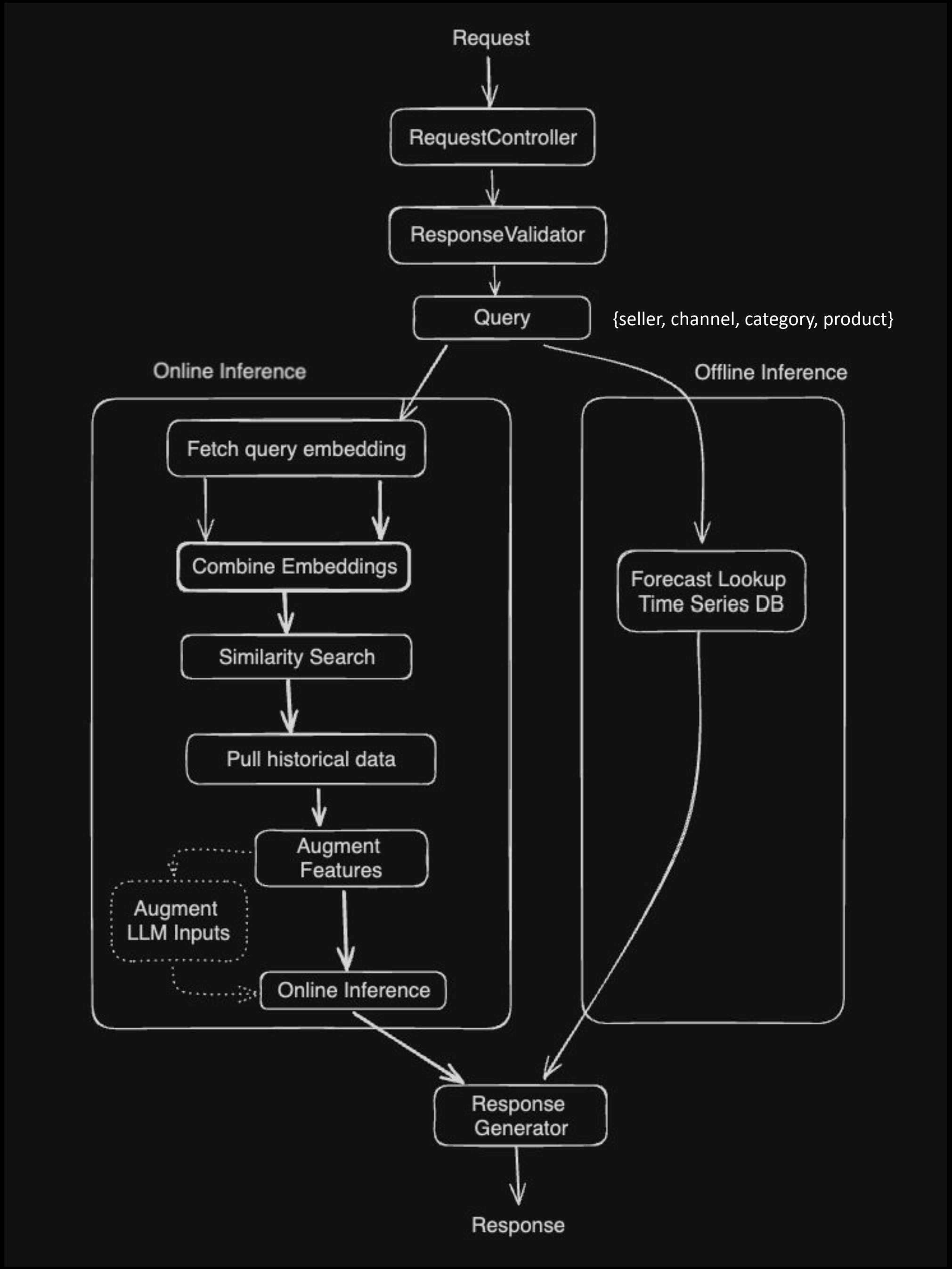
–

- Limited query complexity
- Data consistency challenges

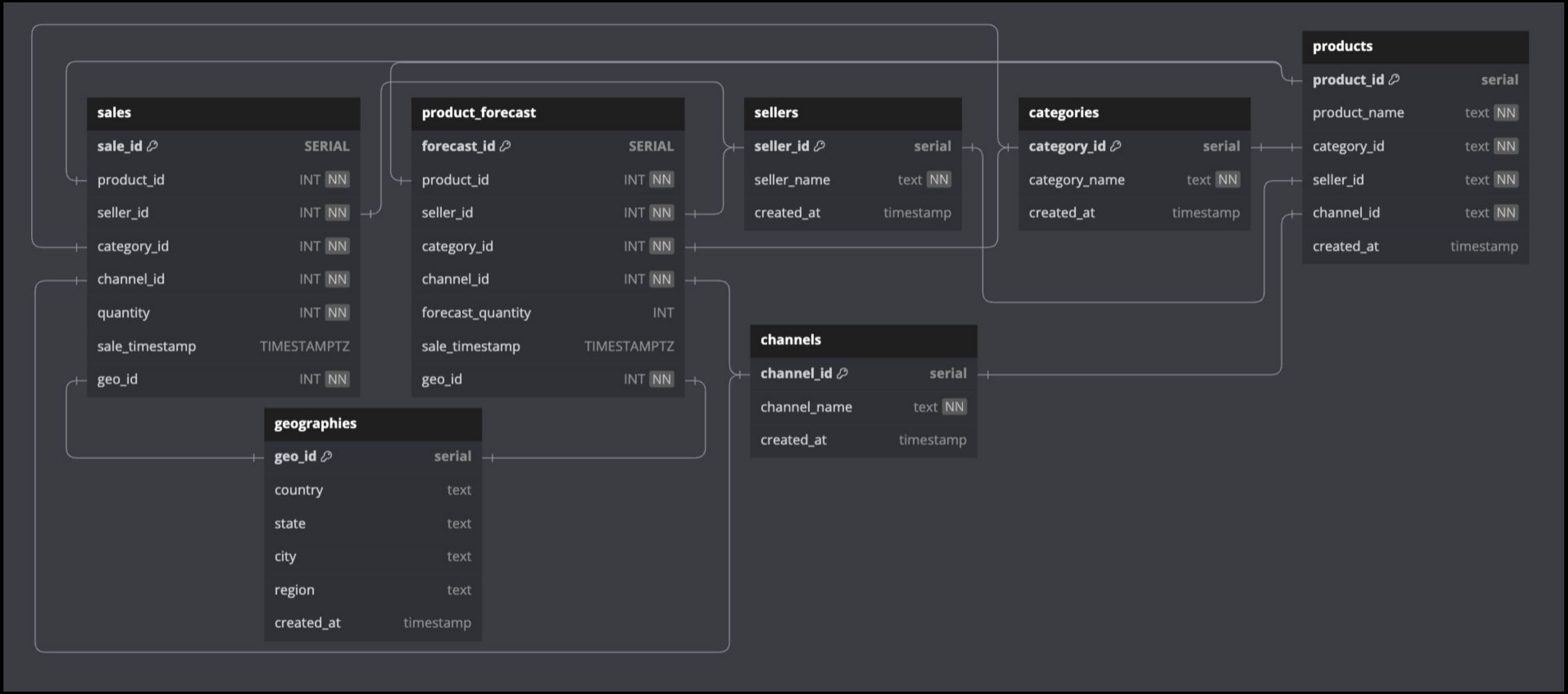
Maximum flexibility and Scalability

Serving Layer Flow

- Request Controller
  - Accept Input Request
  - Set RequestContext
- Request Validator
  - Validate Query Parameters and Request Body
  - Apply A/B Experiment Context
  - Update RequestContext
- Query
  - Get the all channels for seller Id
  - Get the top category within each channel
  - Get top product within the top category
  - {sellerId, channelId, productId}
  - Update RequestContext
- Online Inference
  - Real-time updates
  - Flexibility
  - Computational Cost
  - Latency Impact
  - Scalability Challenges
- Offline Inference
  - Low Latency
  - Low Computational Cost
  - No Real Time Updates
  - Stale Predictions
- Hybrid
  - Best of both worlds
  - Tunable for Tiered Client Configuration
    - Data Update Frequency for Client
    - Sensitivity to Accuracy
- Response Generator
  - Add Metadata
  - Massage Response



Entity Relationship Diagram



sales		
sale_id 🔑	SERIAL	
product_id	INT	NN
seller_id	INT	NN
category_id	INT	NN
channel_id	INT	NN
quantity	INT	NN
sale_timestamp	TIMESTAMPTZ	
geo_id	INT	NN

product_forecast		
forecast_id 🔑	SERIAL	
product_id	INT	NN
seller_id	INT	NN
category_id	INT	NN
channel_id	INT	NN
forecast_quantity	INT	
sale_timestamp	TIMESTAMPTZ	
geo_id	INT	NN

sellers		
seller_id 🔑	serial	
seller_name	text	NN
created_at	timestamp	

categories		
category_id 🔑	serial	
category_name	text	NN
created_at	timestamp	

products		
product_id 🔑	serial	
product_name	text	NN
category_id	text	NN
seller_id	text	NN
channel_id	text	NN
created_at	timestamp	

channels		
channel_id 🔑	serial	
channel_name	text	NN
created_at	timestamp	

geographies		
geo_id 🔑	serial	
country	text	
state	text	
city	text	
region	text	
created_at	timestamp	

TimeScale DB: Scaling and Performance

- Indexing
- Compression
- Continuous Aggregates
- Multi-Node Cluster
- Sharding
- Partitioning

Table	Partitioned by	Sharded By	indexed on
sales	sale_timestamp	seller_id	seller_id, sale_timestamp category_id, sale_timestamp
product_forecast	forecast_timestamp	seller_id, category_id	seller_id, forecast_timestamp category_id, forecast_timestamp

TimescaleDB: Capacity Estimation

sales

- Total **orders per day** 20 million **sellers** × 100 line items per day = **2 billion orders per day**.
- **Storage per day** = 2 billion rows × 44 bytes = **88 GB per day**.
- **Storage per Year: 88** GB/day × 365 days = **32 TB per year**.
- Over 5 years: **5-year sales data** = 160.6 TB/year × 5 years = **160 TB**.

product_forecast

- **Forecast entries per seller per year** = 5 categories × 5 products × 365 days = **9,125 forecast entries per year**.
- **Total forecast entries per year** = 20 million sellers × 9,125 forecast entries = **182.5 billion forecast entries per year**.
- **Storage per year** = 182.5 billion entries × 44 bytes ≈ **7.8 TB per year**
- .**5-year forecast data** = 7.8 TB/year × 5 years = **39 TB**.

Vector Database Schema

Field Name	Data Type	Description
seller_id	INT64	Unique identifier for the seller
embedding	FLOAT_VECTOR	Vector representation of the seller's features (dim depends on model)
timestamp	INT64	Unix timestamp of when the embedding was generated

Field Name	Data Type	Description
channel_id	INT64	Unique identifier for the channel
embedding	FLOAT_VECTOR	Vector representation of the channel’s features (dim depends on model)
timestamp	INT64	Unix timestamp of when the embedding was generated

Serving Compute Estimation

Incoming API Traffic Estimation

- Total Number of Sellers: 20 million
- Daily Active Users: 4 million (20%)
- ~ 4,000,000 / (10 * 60 * 60)
- ~ 120 QPS

Capacity Per Server

- Max Time Budget per API Call - 250 msec
- 3 requests per core per second
- 4 core, 16 GB Memory, 100 GB Disk Node
  - 12 requests per node per second

Total Number of Servers per data center / cloud region

- 10 servers per cloud region
- Redundancy - 3 Cloud regions
- Total - 30 servers

API Caching: Redis

What to Cache

- {seller_id, channel_id, category_id, product_id}
- {seller_id, product_id, fRange, forecast}

Why to Cache

- Avoid expensive database lookup
- Avoid expensive inference execution
- Improve API throughput
- Reduce Latency

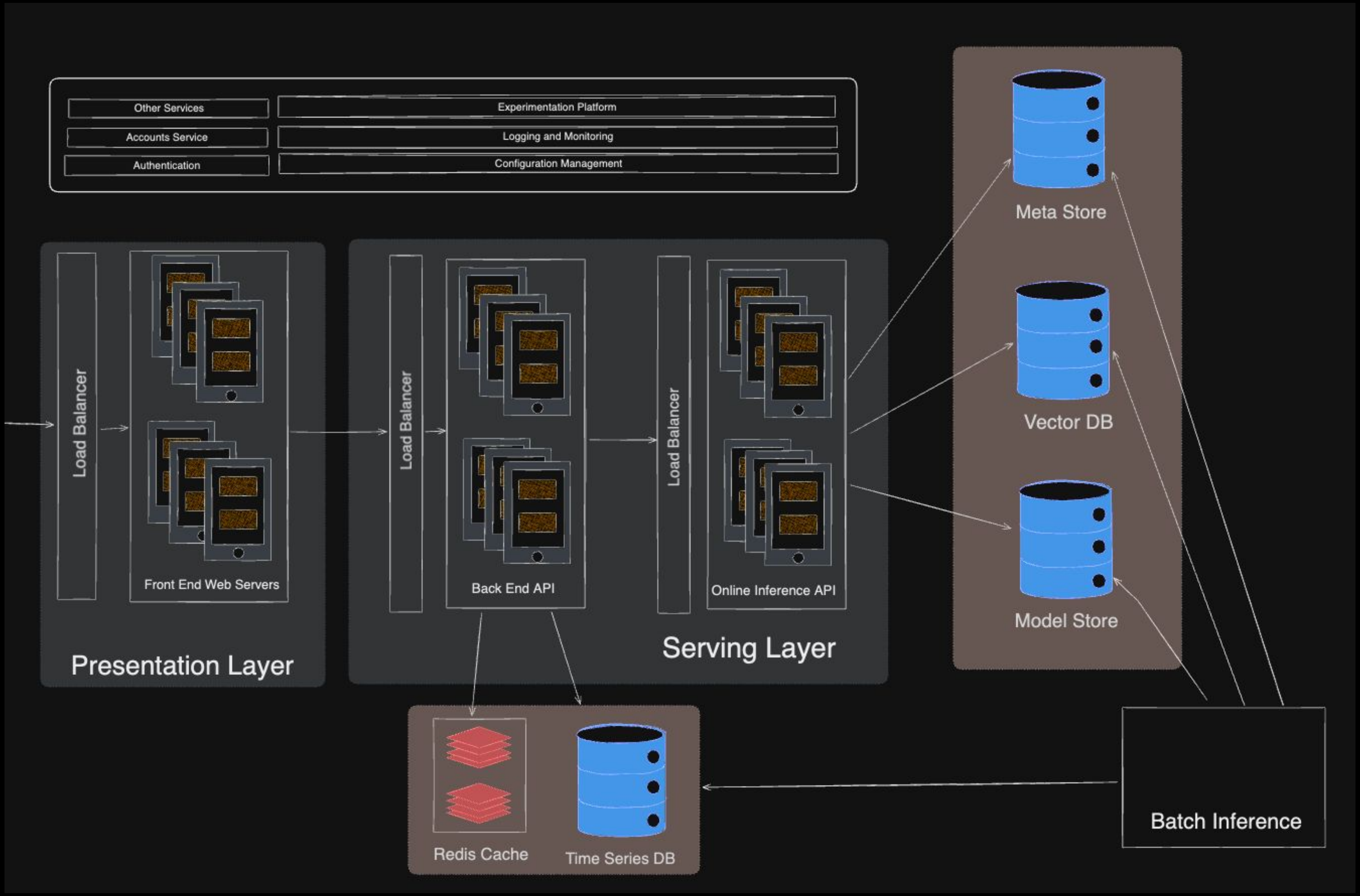
How Long to Cache

- Cache TTL
    - Different strategies possible
        - Fixed time
            - 1 hour, 1 day, etc
        - Greater than 95th percentile of max session time
        - Needs to be tuned

Cache Ratio

- Assume 2% DAU of 20 Million Users
- It Depends
    - Tune and obtain the right ratio of requests to be cached

Serving Layer



HLD

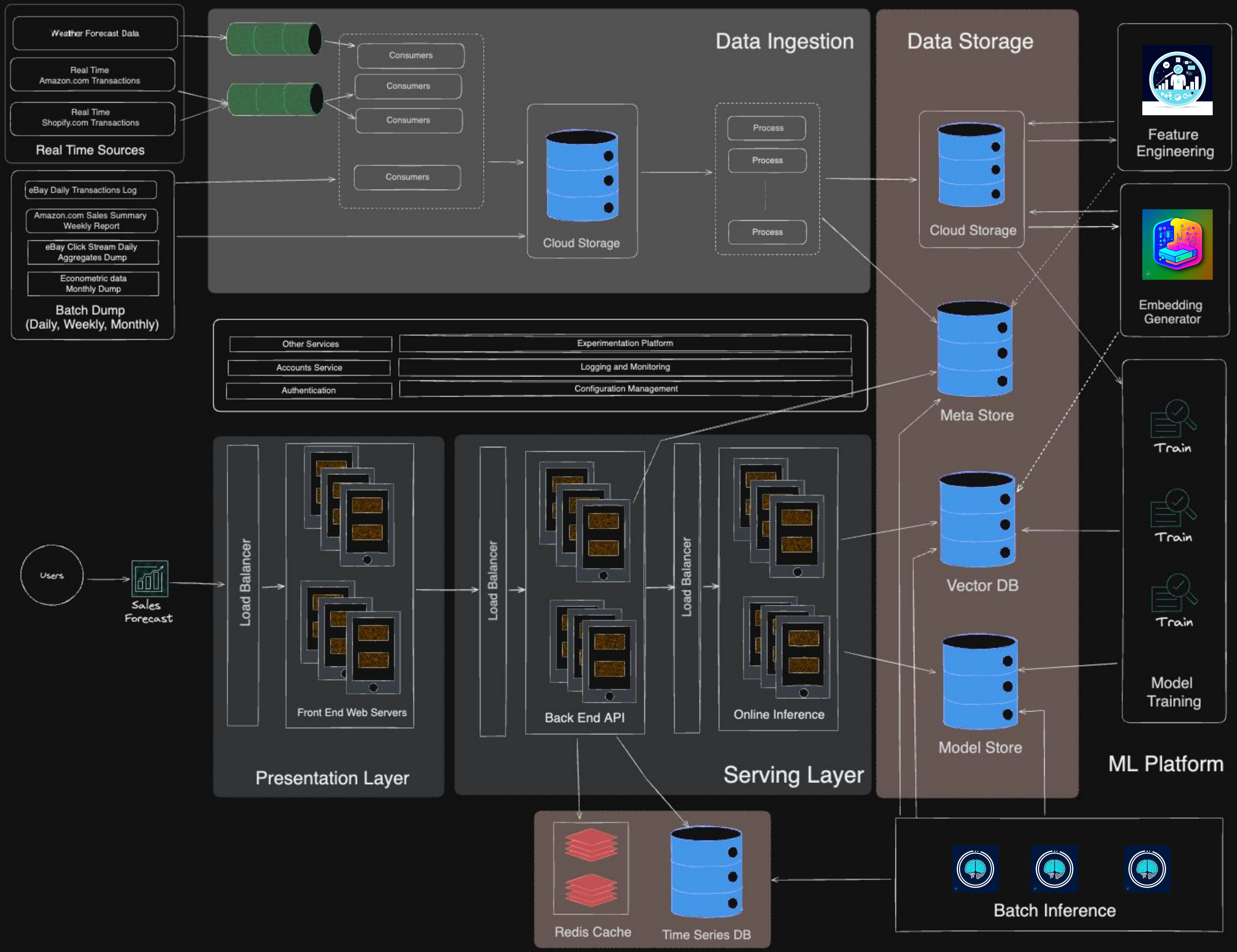
Data Ingestion

Data Storage

ML Platform

Serving Layer

Presentation Layer



Application Cost

AWS S3	AWS Estimate
Active Storage (S3 Standard) with 3 years Retention	$250,000
Archival in Glacier Storage	$60,000
Total Annual Cost	**~ $320,000/year**
AWS PostgreSQL	AWS Estimate
Storage (40 TB)	$40,00/year
Compute (2 x r5.24 xlarge instances)	$30,000/year
Data Transfer (egress, 1TB/month)	$1,080/year
High Availability (Multi-AZ)	$50,000/year
Total Annual Cost	~ $70,000/year
Serving Layer	AWS Estimate
Annual EC2 cost	$50,000
Annual EBS storage cost	$3,000
Annual Data transfer cost	$100
Total Annual Cost	~ $55,000 per year

* Not included: ML Prediction and Inference Costs, Cache, VectorDB costs

Operational Excellence

Observability

Monitoring and Alerting Plan

- RUM
  - Bounce Rate
  - Login Rate
  - Session Time
  - Average number of operations per session
- Application metrics
  - Latency
  - Error rate
  - Exception Counts
- System Metrics
  - CPU
  - Memory
  - Disk
  - Network IO
- Data Metrics
  - Data Health
  - Data Lag
  - Data Availability
  - Data Volume
  - Data Schema

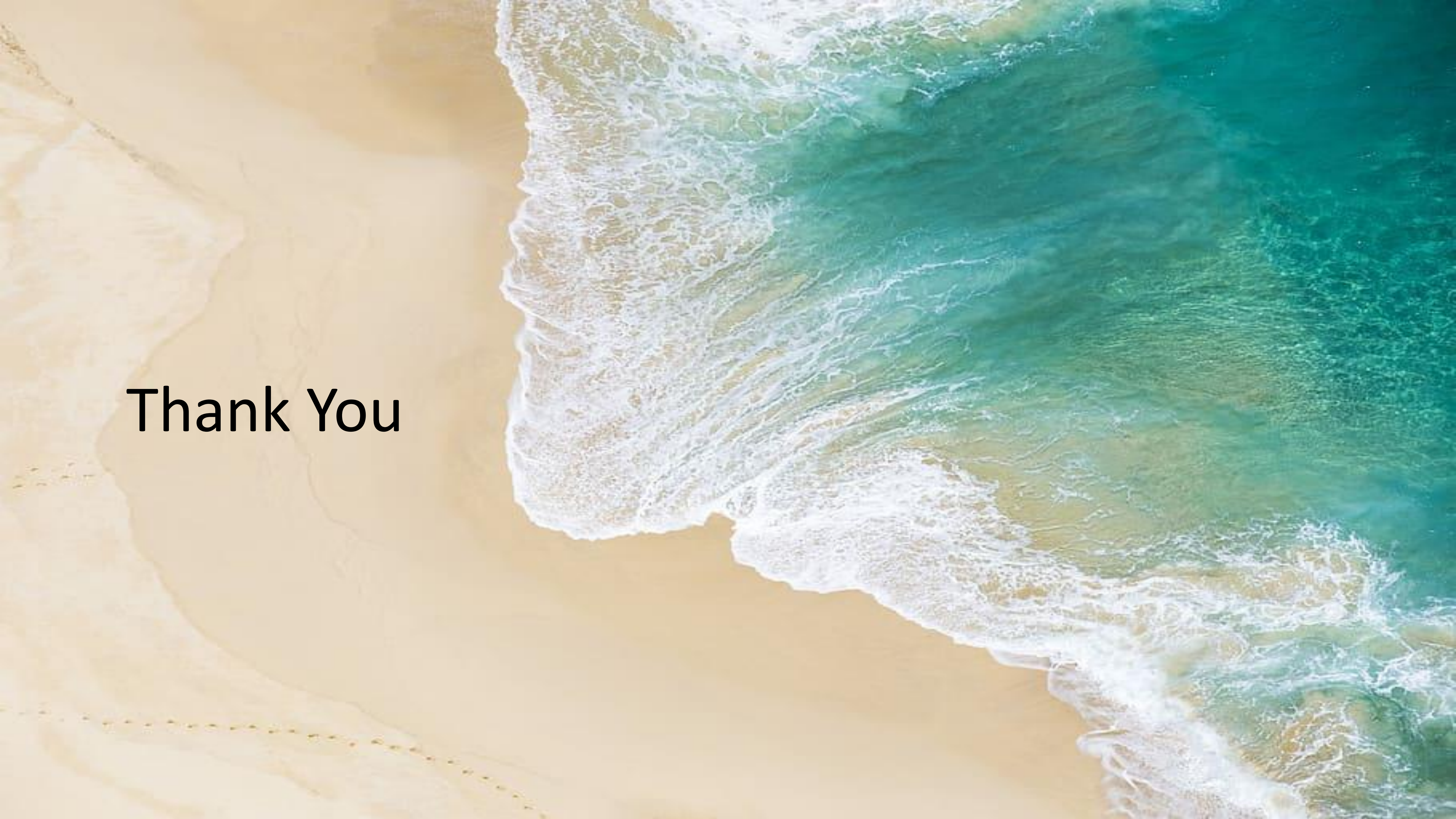
Monitoring and Alerting Tools

- Prometheus
  - For monitoring latency, throughput, and memory usage.
- Grafana
  - For visualizing performance metrics.
- New Relic
  - For monitoring API performance and identifying bottlenecks.
- Splunk
  - Log Aggregation Analysis, dashboarding and monitoring
- PagerDuty / xMatters / Slack
  - Alerting Tools

Playbooks

- Versioned Release
- Deployment
- A/B Experiments
  - Measuring Statistical Significance
  - Production Rollout
    - Data
    - Model
    - Application
- Production Issue Triaging
  - Application Issues
  - System Issues
  - Cloud Issues

- Feature Switches
- Fail Safe mode
- Failover Strategies
  - Automated Failover
  - Manual Failover
- Restore Operations
- Rollback



Thank You