System Design for Recommendations & Search

(*) Model

- Deep Learning
  - Neural Collaborative Filtering.
  - Transformers for Recommendation

(*)

(*) Model Design — How it fits in wholistic design?

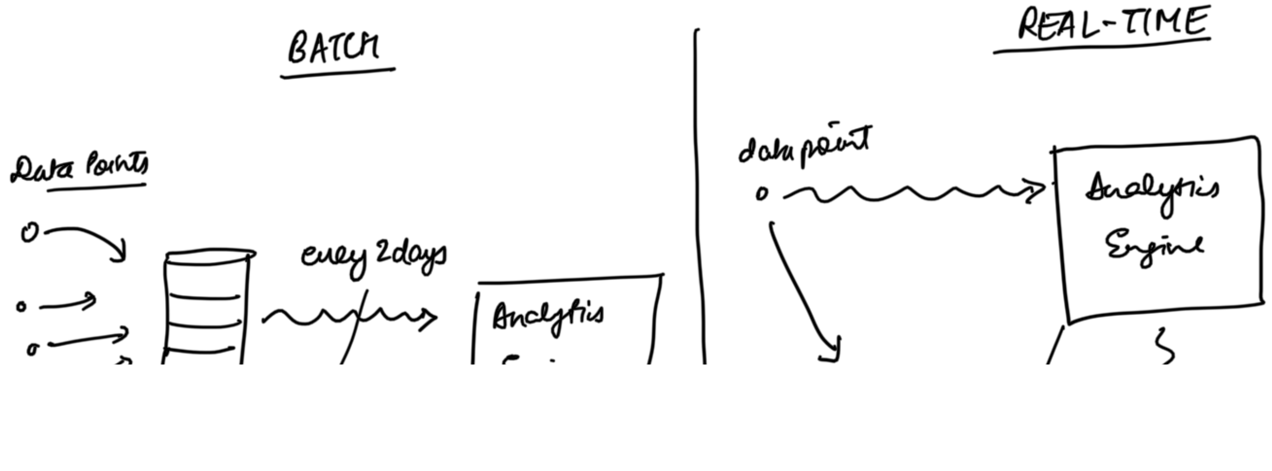
→ Move from Batch to Real-Time
↓
has constraints

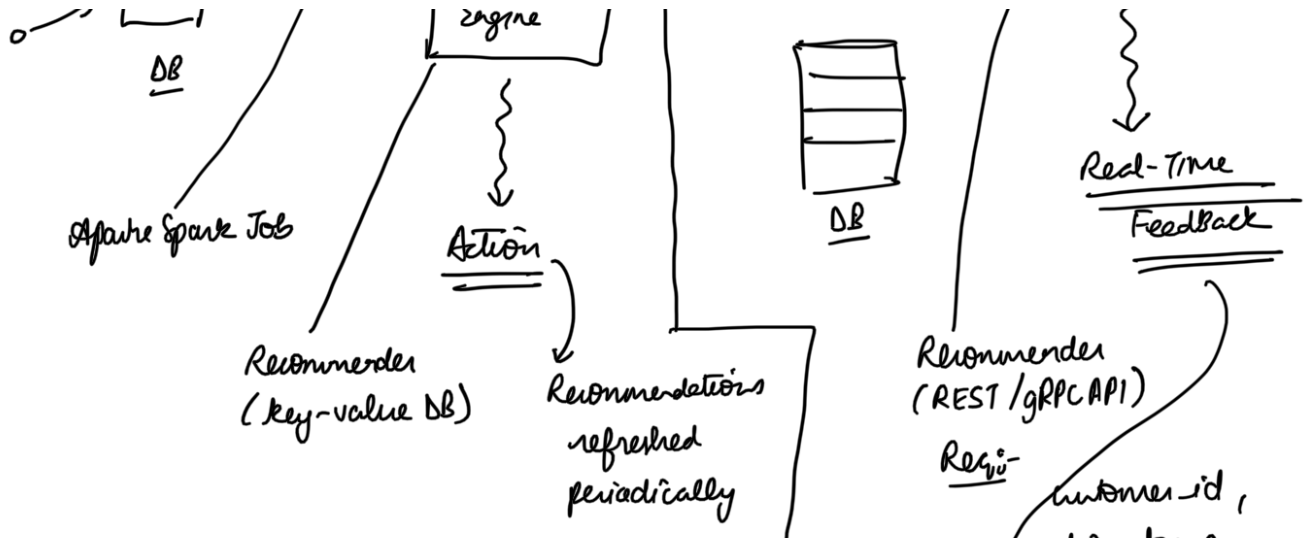
(*) Basic Fundamental Design that applies to 90% of all Recommendation Based Search.

How we do this in Industry?

- Candidate Retreival
- Ranking

BATCH v/s REAL-TIME





customer_id,
timestamp,
what they have
browsed through in past

Recommendations generated
in real-time

Pros

- Pre-Computed (saved somewhere in DB)
- Decouple Compute from Serving
- Lower Operational Load

if spark job breaks,
we can show yesterday's
recommendations.
(stale data)

we don't
need to maintain
much for
* low latency
* high throughput
∴ system is not
up 24/7

∴ we create
recommendations
of all users.

Pros

- Responsive to Time-Sensitive Context
- Reduced Cost for non-operative users.

- works well for 4* customers
- guesses mood of buyer &
  updates recommendations
  in home page.
- makes customers sticky on
  the app.

Cons: API should
have:-

- low latency
- high throughput
- expensive
  in infra
  &
  developer time
  (Ops)

REAL-TIME /ON-DEMAND /ONLINE

... offline env. to

BATCH/ OFFLINE

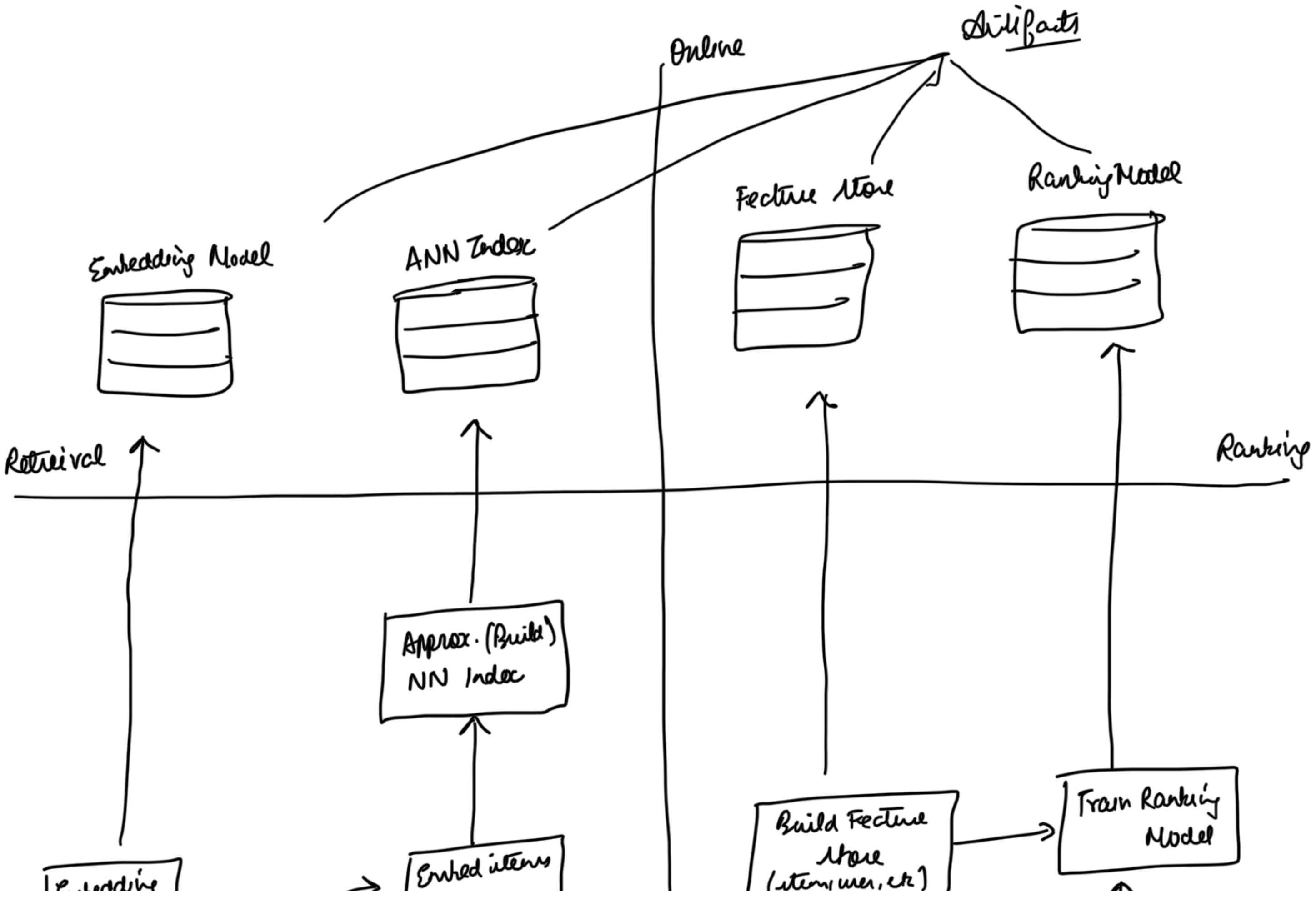
- Most Batch Processes (Training / Indexing / Graph Building)
- Load data in feature stores.

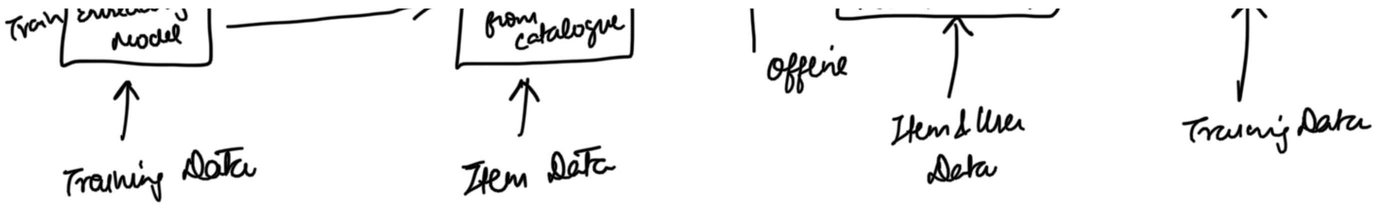
- uses artifacts from offline
serve requests

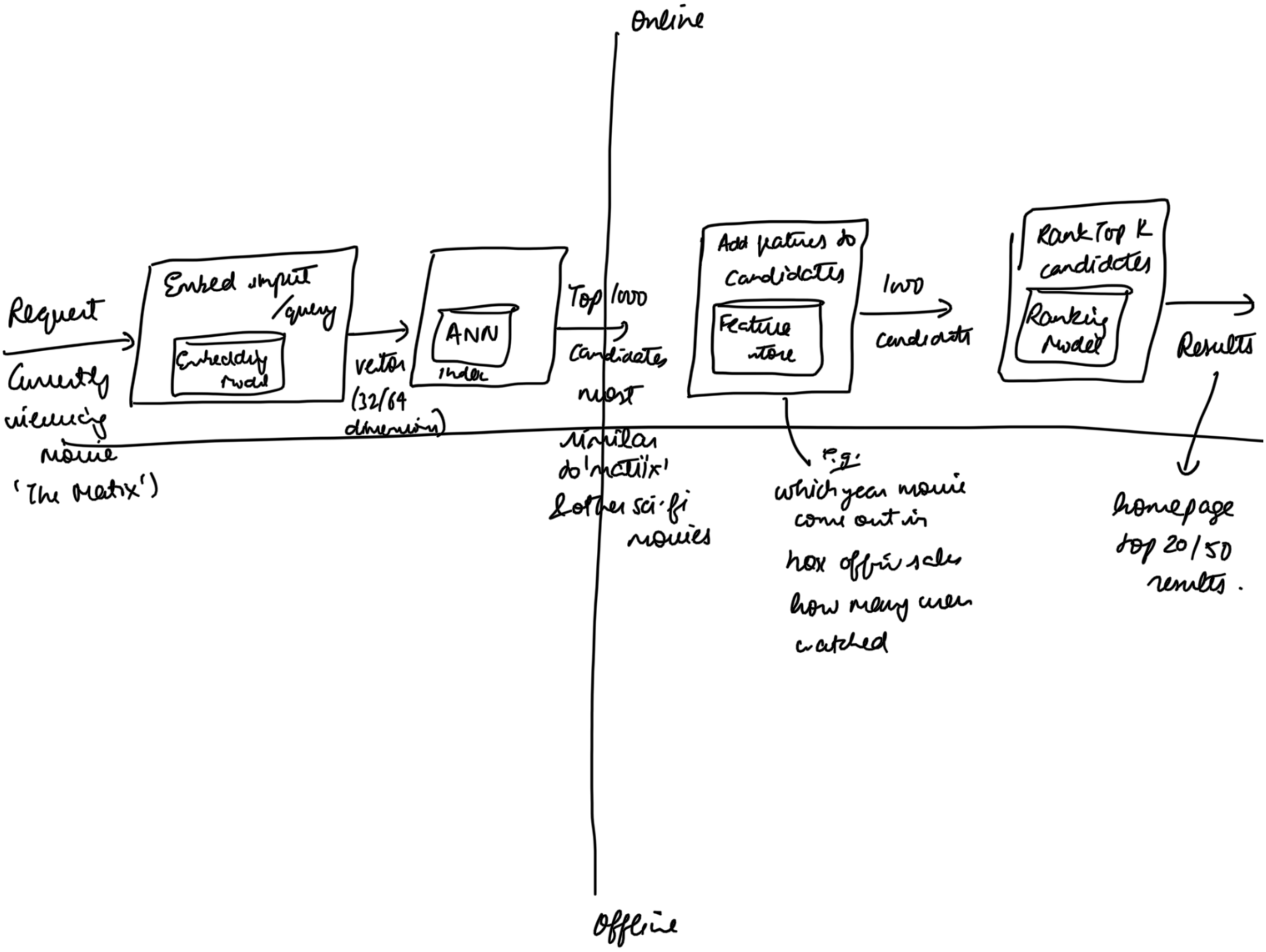
- _ CANDIDATE RETREIVAL
  _ RANKING.

Approximate
- using Nearest Neighbours (NN), graphs, etc
- searches millions of items to get hundreds of req.
- fast but coarse (not precise)

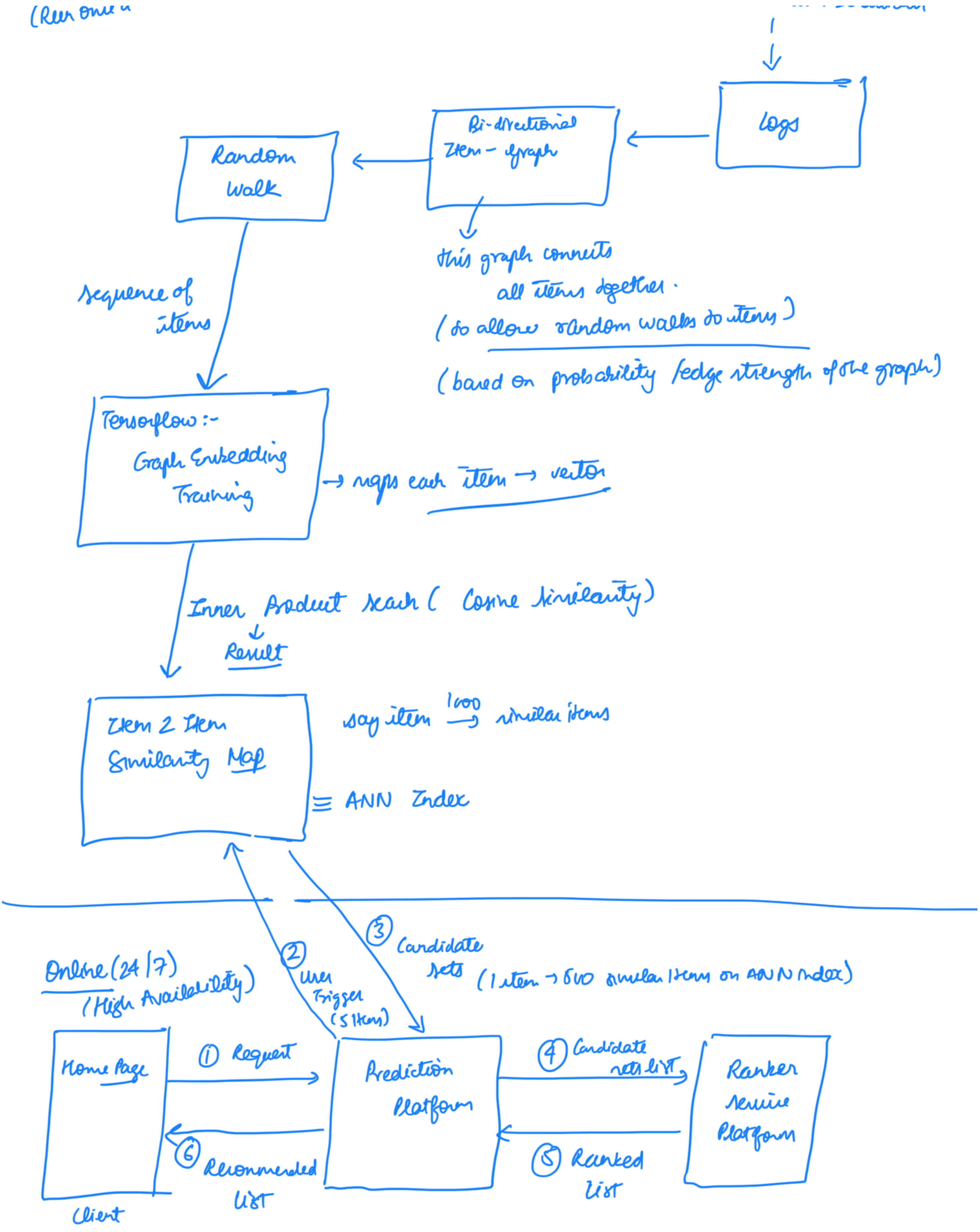
- slow but precise
- ranks hundreds of candidates
- adds more features (users, items, context (day / month), more engagement with user)



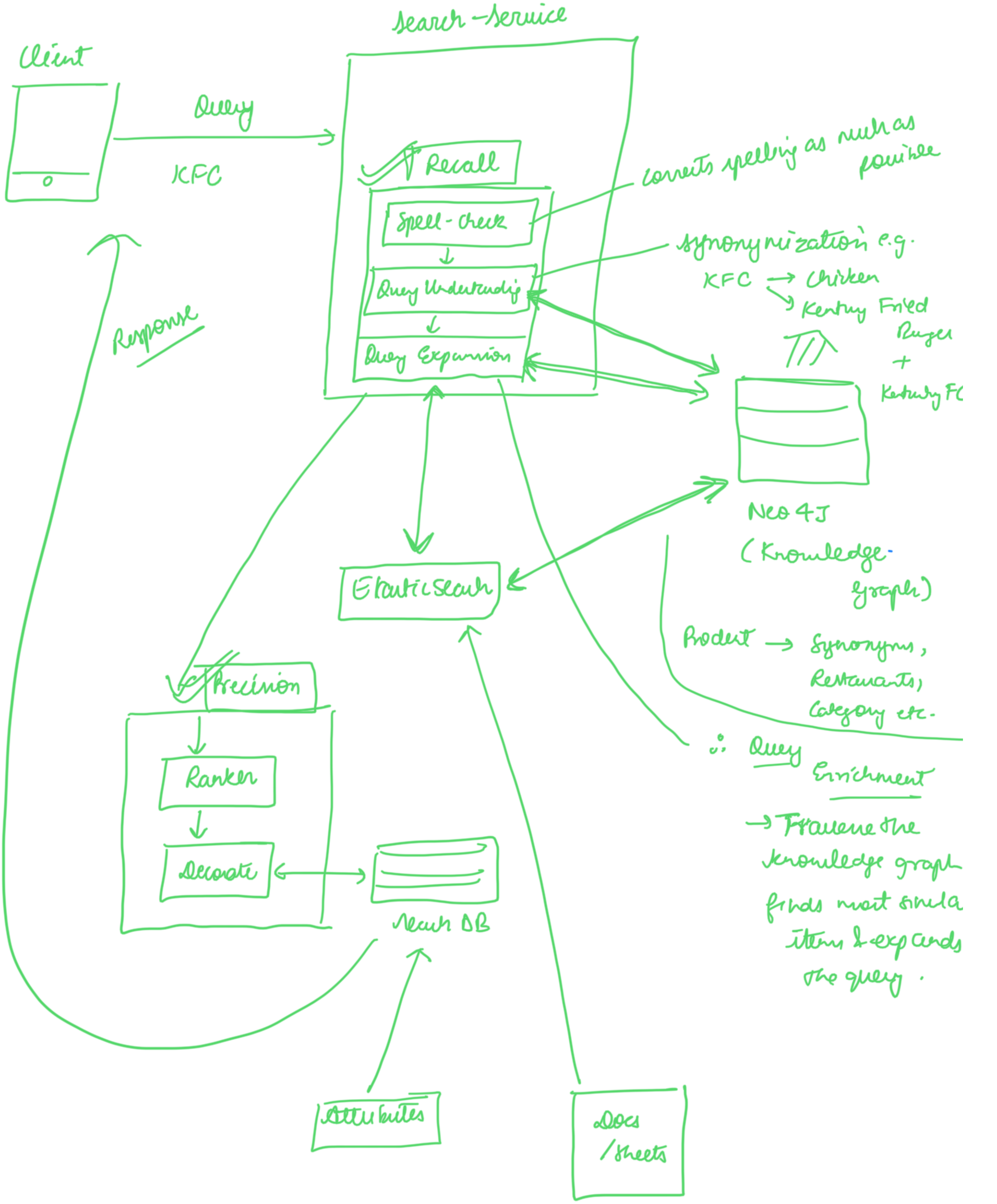




Industry Examples


Logs
Bi-directional Item-graph
this graph connects all items together.
(to allow random walks to items)
(based on probability /edge strength of the graph)
Random Walk
sequence of items
Tensorflow :- Graph Embedding Training
→ maps each item → vector
Inner Product search (Cosine similarity)
Result
Item 2 Item Similarity Map
say item $\xrightarrow{1000}$ similar items
≡ ANN Index
Online (24/7)
(High Availability)
② User Trigger (5 items)
③ Candidate sets (1 item → 500 similar items on ANN index)
Home Page
Client
① Request
⑥ Recommended list
Prediction Platform
④ Candidate rec list
⑤ Ranked list
Ranker service Platform

Building graphs for Query Expansion & Retrieval



corrects spelling as much as possible

synonymization e.g.
KFC → Chicken
→ Kentucky Fried Burger
+
Kentucky FC

Product → Synonyms, Restaurants, Category etc.

∴ Query Enrichment

→ Traverse the knowledge graph finds most simila items & expands the query.