TWITTER SYSTEM DESIGN

NFR

1. Read heavy
2. Fast rendering
3. Fast tweet
4. Lag OK in some cases
   § like if someone has posted a tweet, OK to see after 2sec etc.
5. Scale
   - 150 Mil. DAU
   - 350 Mil. MAU
   - 1.5 Bil Accounts
   - 500 Mil. tweets / day
   - 5700 tweets / second
   ≈ at peak → 12,000 tweets / second.

FR

1. Tweet
2. Re-tweet
3. Follow
4. Search

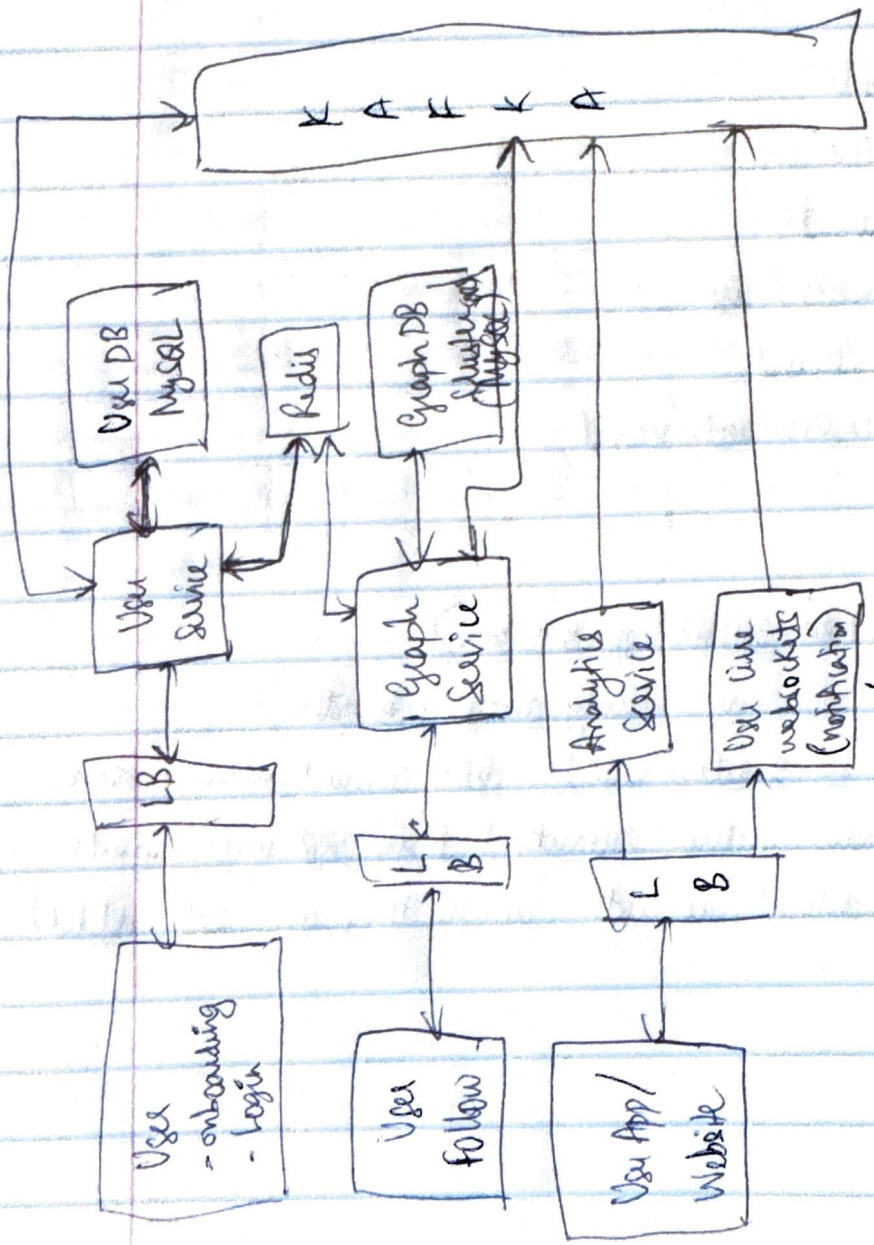
Users

1. Famous (Politician, Celebrity, sports person)
2. Active (ppl who are actively accessing twitter)
3. Live → (subset of active users) → like current online persons
4. Passive → have active accounts but are not very active
5. Inactive → deleted accounts. (usually they are soft deleted)

3 Flows

1. User onboarding flow
2. Tweeting flow
3. Analytics flow

→ Whenever we are designing read-heavy systems, we have to precompute lots of things (or) use cache for better performance. (so that there is low latency).



Using short-URL, as a tweet allows only 140 characters in link in a tweet. so short-URL service is required

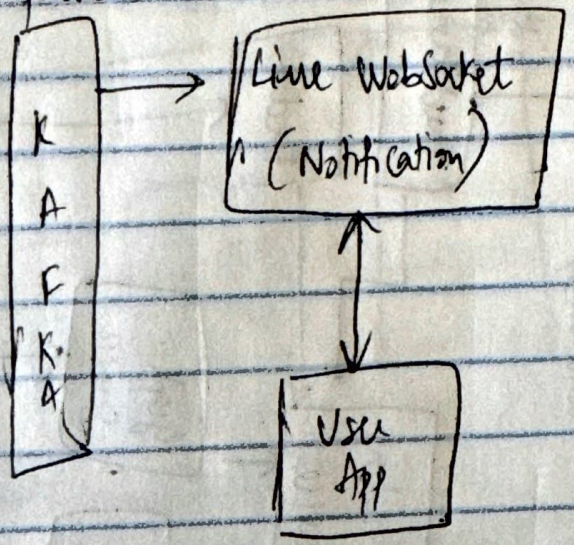
What happens when somebody posts a tweet?

Tweet can contain either text, image or video.
(multi-media content)

Problems in this design

We are using Redis to store so much data.
But Redis is an in-memory solution. so we need a lot of RAM. So, that's inefficient.
(So, we will store only active users data in Redis).

(So Active & Passive users are handled as per our design)
How can we optimize all of this for our live users ??
After previous talks



Now, for celebrities → say if the celebrity tweets then say there are 75M followers. Then each time the celebrity tweets then 75M updates has to be done, which is impractical. And celebrities will also follow celebrities.

So,

Fan-out on Write

When a user posts a tweet, the system immediately "fans out" that tweet into the timeline of every follower.
Usually used for normal users.

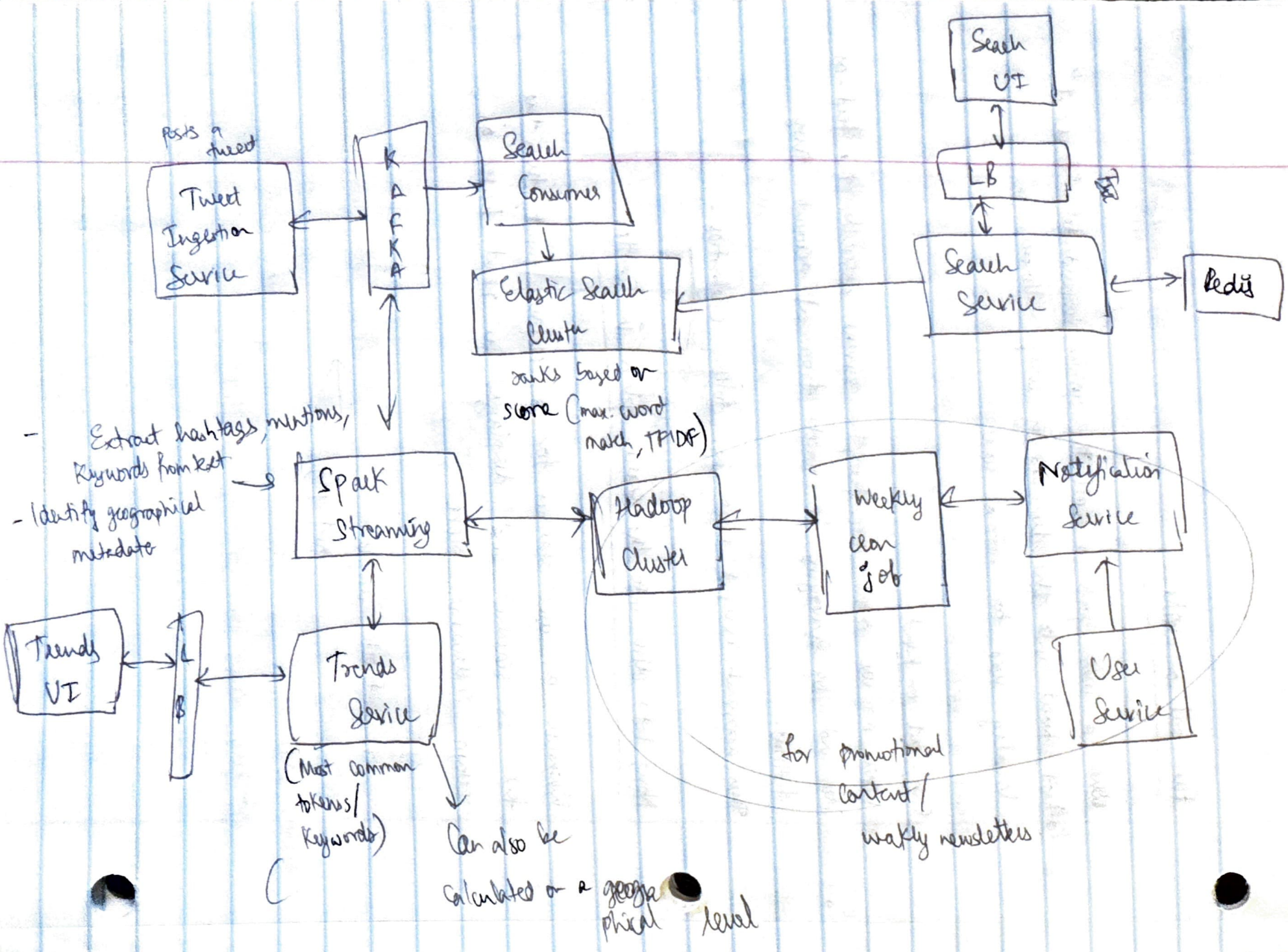
Fan-out on Read

When a user opens their feed, the system assembles the feed on demand by querying for tweets from all accounts they follow.
This approach is used when we follow celebrities.

Now, let's dive into the search option in Twitter

Whenever a user searches for a certain tweet, then the text is searched and displayed the results based on certain relevance score.

Block diagram → next page


Search UI
LB
Search Service
Redis
Search Consumer
Elastic Search Cluster
ranks based on score (max. word match, TF IDF)
Posts a tweet
Tweet Ingestion Service
KAFKA
- Extract hashtags, mentions, Keywords from text
- Identify geographical metadata
Spark Streaming
Hadoop Cluster
Weekly clean job
Notification Service
User Service
Trends UI
LB
Trends Service
(Most common tokens/ keywords)
Can also be calculated on a geographical level
for promotional content / weekly newsletters