### **Design the Twitter newsfeed**

Design the backend system for the Twitter newsfeed. We want to be able to post tweets, follow other users, and favorite tweets. A user should be able to see a feed of tweets of his/her followers.

**Use cases scope:**

1) User posts a tweet

2) User views the home timeline (activity from people the user is following)

3) Fav tweets

**General Assumptions:**

1. Traffic is not evenly distributed
2. Posting a tweet should be fast
3. Fanning out a tweet to all of your followers should be fast100 million active users
4. Twitter is more read heavy than write heavy
5. Optimize for fast reads of tweets

**User post a tweet:**

1. The user posts a tweet to the Web Server, running as a reverse proxy
2. The Web Server forwards the request to the Write API server
3. The Write API stores the tweet in the user's timeline on a SQL database
4. The Write API contacts the Fan Out Service, which does the following:
   1. Queries the User Graph Service to find the user's followers stored in the Memory Cache
   2. Stores the tweet in the home timeline of the user's followers in a Memory Cache
   3. O(n) operation: 1,000 followers = 1,000 lookups and inserts
   4. Stores the tweet in the Search Index Service to enable fast searching
   5. Stores media in the Object Store
   6. Uses a Queue to asynchronously send out notifications

**User views the home timeline:**

1. The Client posts a home timeline request to the Web Server
2. The Web Server forwards the request to the Read API server
3. The Read API server contacts the Timeline Service, which does the following:
   1. Gets the timeline data stored in the Memory Cache, containing tweet ids and user ids - O(1)
   2. Queries the Tweet Info Service with a multiget to obtain additional info about the tweet ids - O(n)
   3. Queries the User Info Service with a multiget to obtain additional info about the user ids - O(n)

**Fav the tweet Workflow:**

1. The Client posts request to the Web Server to favorite the tweets.
2. The Web Server forwards the request to the Write API server
3. The Write API stores this information in SQL database.
4. The Write API contacts the Fan Out Service, which does the following:
   1. Queries the User Graph Service to find the user's followers stored in the Memory Cache
   2. Stores the tweet in the home timeline of the user's followers in a Memory Cache
   3. O(n) operation: 1,000 followers = 1,000 lookups and inserts
   4. Stores the tweet in the Search Index Service to enable fast searching
   5. Stores media in the Object Store
   6. Uses a Queue to asynchronously send out notifications

**Storage:**

We could store the user's own tweets to populate the user timeline (activity from the user) in a SQL or NoSQL database.

Delivering tweets and building the home timeline (activity from people the user is following) is trickier. Fanning out tweets to all followers (60 thousand tweets delivered on fanout per second) will overload a traditional relational database. We'll probably want to choose a data store with fast writes such as a NoSQL database or Memory Cache. Reading 1 MB sequentially from memory takes about 250 microseconds, while reading from SSD takes 4x and from disk takes 80x longer.

Storing media such as photos or videos on an Object Store.