Design of a Newsfeed System

Learn how to design a newsfeed system.

We'll cover the following

Let’s discuss the high-level and detailed design of a newsfeed system based on the requirements discussed in the previous lesson.

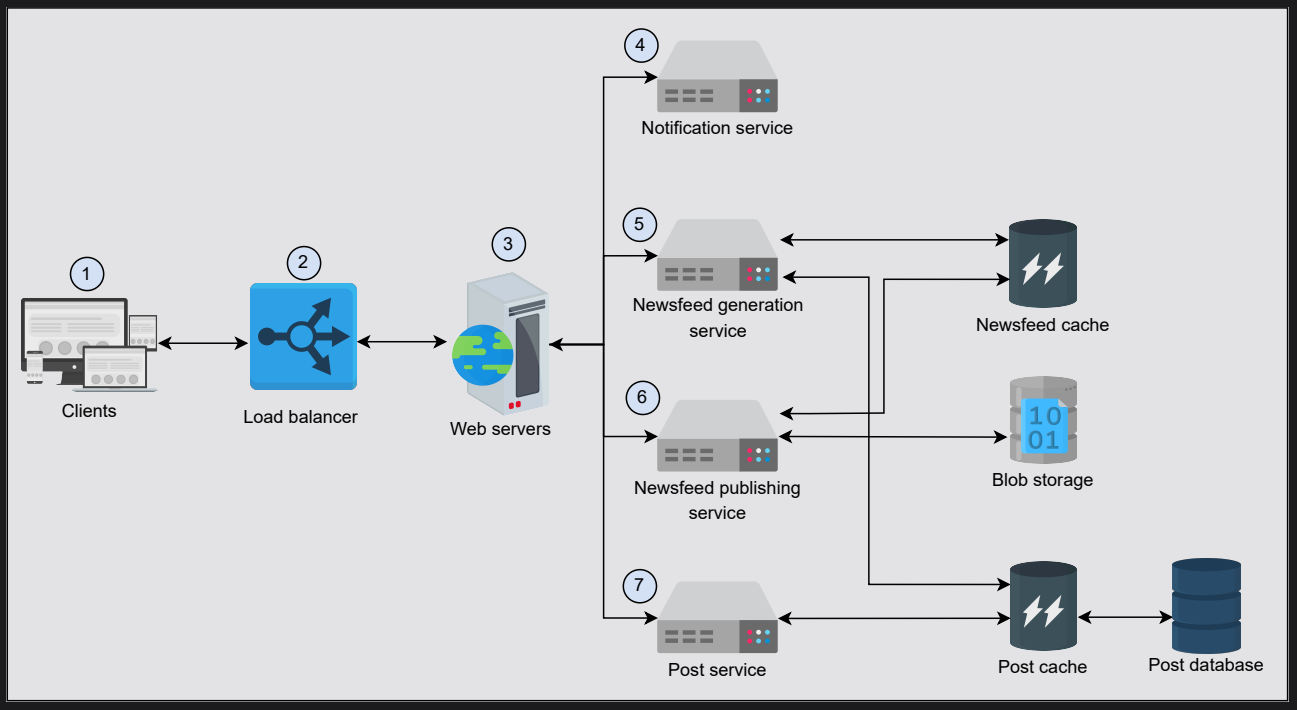
High-level design of a newsfeed system

Primarily, the newsfeed system is responsible for the following two tasks:

1. **Feed generation:** The newsfeed is generated by aggregating friends’ and followers’ posts (or feed items) based on some ranking mechanism.
2. **Feed publishing:** When a feed is published, the relevant data is written into the cache and database. This data could be textual or any media

content. A post containing the data from friends and followers is populated to a user's newsfeed.

Let's move to the high-level design of our newsfeed system. It consists of the above two essential parts, shown in the following figure:



High-level design of the Newsfeed system

Let's discuss the main components shown in the high-level design:

1. **User(s):** Users can make a post with some content or request their newsfeed.
2. **Load balancer:** It redirects traffic to one of the web servers.
3. **Web servers:** The web servers encapsulate the back-end services and work as an intermediate layer between users and various services. Apart from enforcing authentication and rate-limiting, web servers are responsible to redirect traffic to other back-end services.
4. **Notification service:** It informs the newsfeed generation service whenev a new post is available from one's friends or followers, and sends a push notification.
5. **Newsfeed generation service:** This service generates newsfeeds from the posts of followers/friends of a user and keeps them in the newsfeed

cache.

6. **Newsfeed publishing service:** This service is responsible for publishing newsfeeds to a users' timeline from the newsfeed cache. It also appends a thumbnail of the media content from the blob storage and its link to the newsfeed intended for a user.
7. **Post-service:** Whenever a user requests to create a post, the post-service is called, and the created post is stored on the post database and corresponding cache. The media content in the post is stored in the blob storage.

API design

APIs are the primary ways for clients to communicate with servers. Usually, newsfeed APIs are HTTP-based that allow clients to perform actions, including posting a status, retrieving newsfeeds, adding friends, and so on. We aim to generate and get a user's newsfeed; therefore, the following APIs are essential:

Generate user's newsfeed

The following API is used to generate a user's newsfeed:

```
generateNewsfeed(user_id)
```

This API takes users' IDs, and determines their friends and followers. This API generates newsfeeds that consist of several posts. Since internal system components use this API, therefore, it can be called offline to pre-generate newsfeeds for users. The pre-generated newsfeeds are stored on persistent storage and associated cache.

The following parameter is used in this API call:

Parameter	Description
`user_id`	A unique identification of the user for whom the newsfeed is generated.

Get user’s newsfeed

The following API is used to get a user’s newsfeed:

```
getNewsfeed(user_id, count)
```

The `getNewsfeed(.)` API call returns a JSON object consisting of a list of posts.

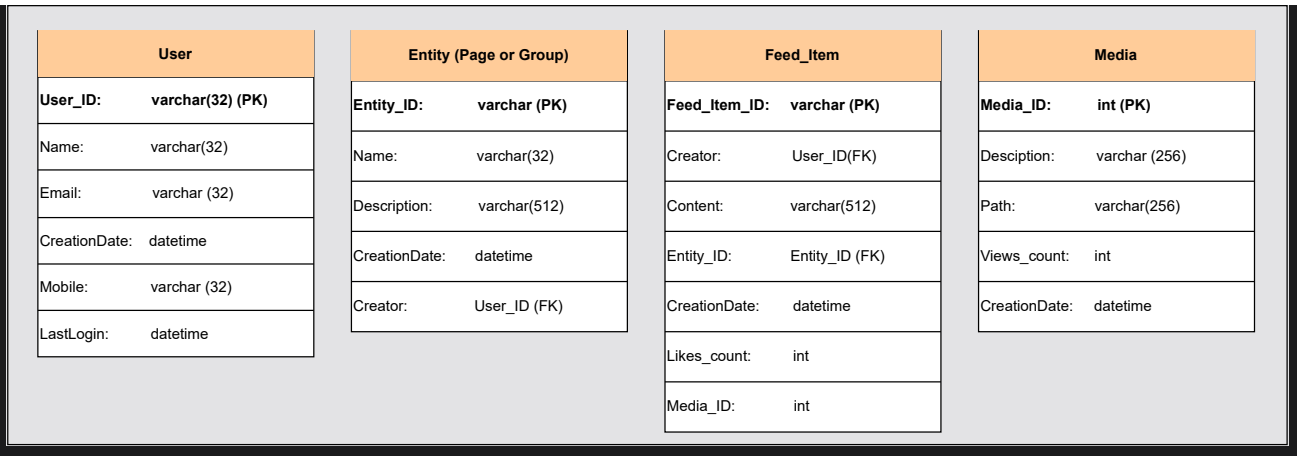
The following parameters are used for this API:

Parameter	Description
`user_id`	A unique identification of the user for whom the system will fetch the newsfeed.
`count`	The number of feed items (posts) that will be retrieved per request.

Storage schema

The database relations for the newsfeed system are as follows:

- **User:** This relation contains data about a user. A user can also be a follower or friend of other users.
- **Entity:** This relation stores data related to any entity, such as pages, groups, and so on.
- **Feed_item:** The data about posts created by users is stored in this relation.
- **Media:** The information about the media content is stored in this relation.



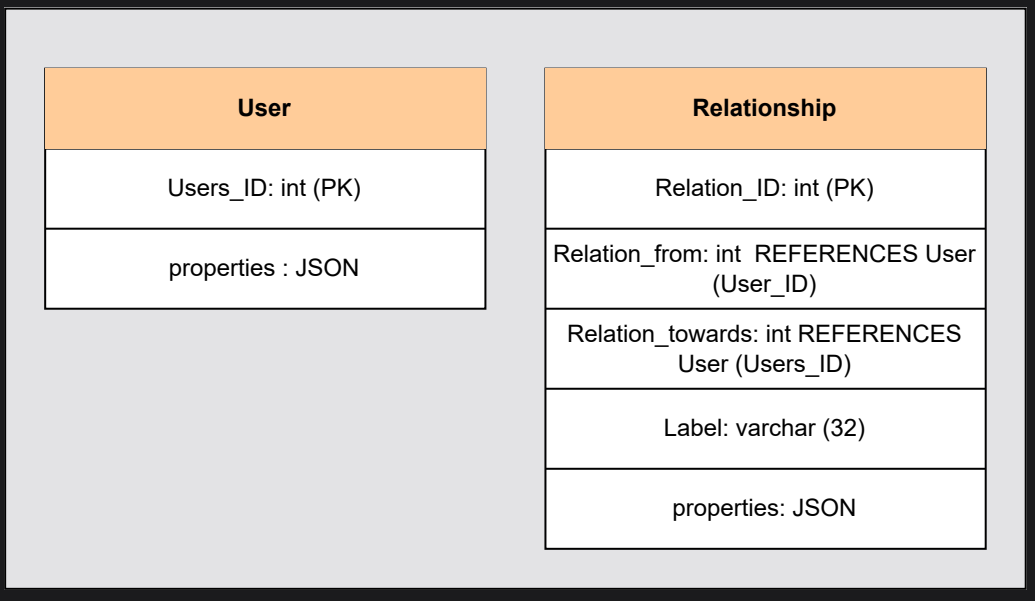
The database schema for the newsfeed system

The user and post data is structured, so we will use SQL-based databases to store it. We use a graph database to store relationships between users, friends, and followers. For this purpose, we follow the property graph model. We can think of a graph database consisting of two relational tables:

1. For vertices that represent users
2. For edges that denotes relationships among them

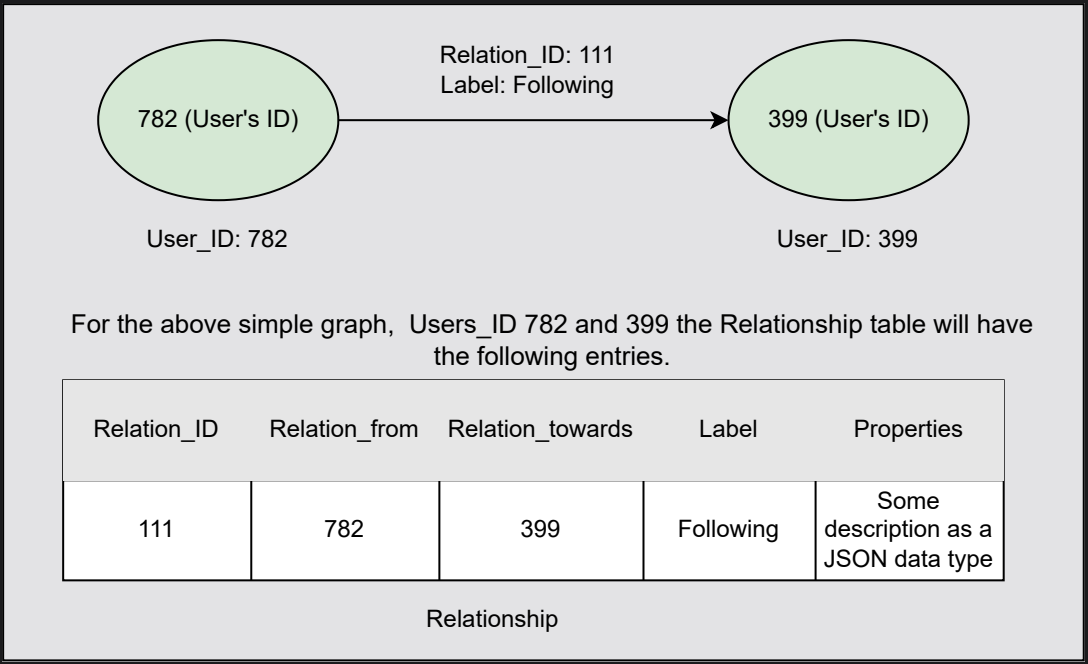
Therefore, we follow a relational schema for the graph store, as shown in the following figure. The schema uses the PostgreSQL JSON data type to store the properties of each vertex (user) or edge (relationship).

An alternative representation of a **User** can be shown in the graph database below. Where the `Users_ID` remains the same and attributes are stored in a JSON file format.



Schema of the graph database to hold relationships between users

The following figure demonstrates how graph can be represented using the relational schema:



A graph between two users consisting of two vertices and an edge

Detailed design

Let’s explore the design of the newsfeed system in detail.

As discussed earlier, there are two parts of the newsfeed system; newsfeed publishing and newsfeed generation. Therefore, we’ll discuss both parts, starting with the newsfeed generation service.

The newsfeed generation service

Newsfeed is generated by aggregated posts (or feed items) from the user’s friends, followers, and other entities (pages and groups).

In our proposed design, the **newsfeed generation service** is responsible for generating the newsfeed. When a request from a user (say Alice) to retrieve a newsfeed is received at web servers, the web server either:

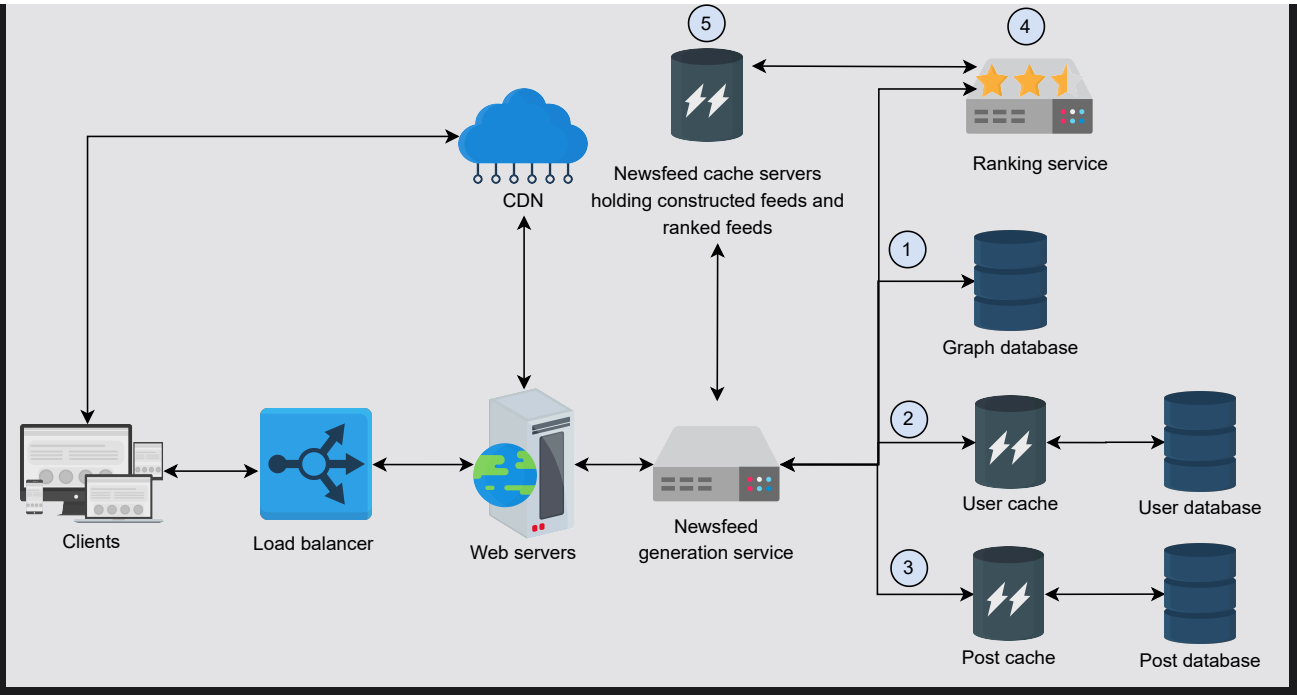
- Calls the newsfeed generation service to generate feeds because some users don’t often visit the platform, so their feeds are generated on their

request.
- It fetches the pre-generated newsfeed for active users who visit the platform frequently.

The following steps are performed in sequence to generate a newsfeed for Alice:

1. The newsfeed generation service retrieves IDs of all users and entities that Alice follows from the graph database.
2. When the IDs are retrieved from the graph database, the next step is to get their friends' (followers and entities) information from the user cache, which is regularly updated whenever the **users database** gets updated/modified.
3. In this step, the service retrieves the latest, most popular, and relevant posts for those IDs from the post cache. These are the posts that we might be able to display on Alice's newsfeed.
4. The **ranking service** ranks posts based on their relevance to Alice. This represents Alice's current newsfeed.
5. The newsfeed is stored in the the newsfeed cache from which top N posts are published to Alice's timeline. (The publishing process is discussed in detail in the following section).
6. In the end, whenever Alice reaches the end of her timeline, the next top N posts are fetched to her screen from the newsfeed cache.

The process is illustrated in the following figure:



Working of the newsfeed generation service

Point to Ponder

Question

The creation and storage of newsfeeds for each user in the cache requires an enormous amount of memory (step 5 in the above section). Is there any way to reduce this memory consumption?

Show Answer

The newsfeed publishing service

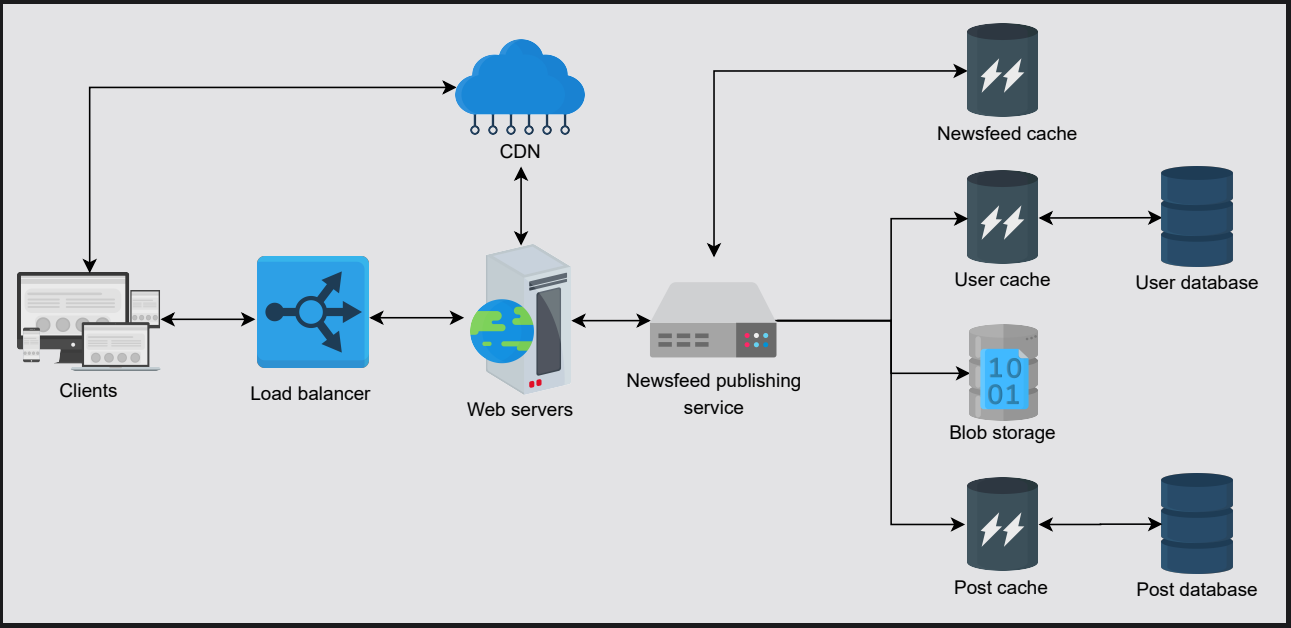
At this stage, the newsfeeds are generated for users from their respective friends, followers, and entities and are stored in the form of `<Post_ID, User_I` in the news feed cache.

Now the question is how the newsfeeds generated for Alice will be published to her timeline?

The **newsfeed publishing service** fetches a list of post IDs from the newsfeed cache. The data fetched from the newsfeed cache is a tuple of post and user IDs, that is., `<Post_ID, User_ID>`. Therefore, the complete data about posts and users are retrieved from the users and posts cache to create an entirely constructed newsfeed.

In the last step, the fully constructed newsfeed is sent to the client (Alice) using one of the **fan-out approaches**↴. The popular newsfeed and media content are also stored in CDN for fast retrieval.

> **What is the problem with generating a newsfeed upon a user's request (also called live updates)?**



The newsfeed publishing service in action

Point to Ponder

Question

How is the newsfeed of the friends and followers updated when a user creates a new post?

Show Answer

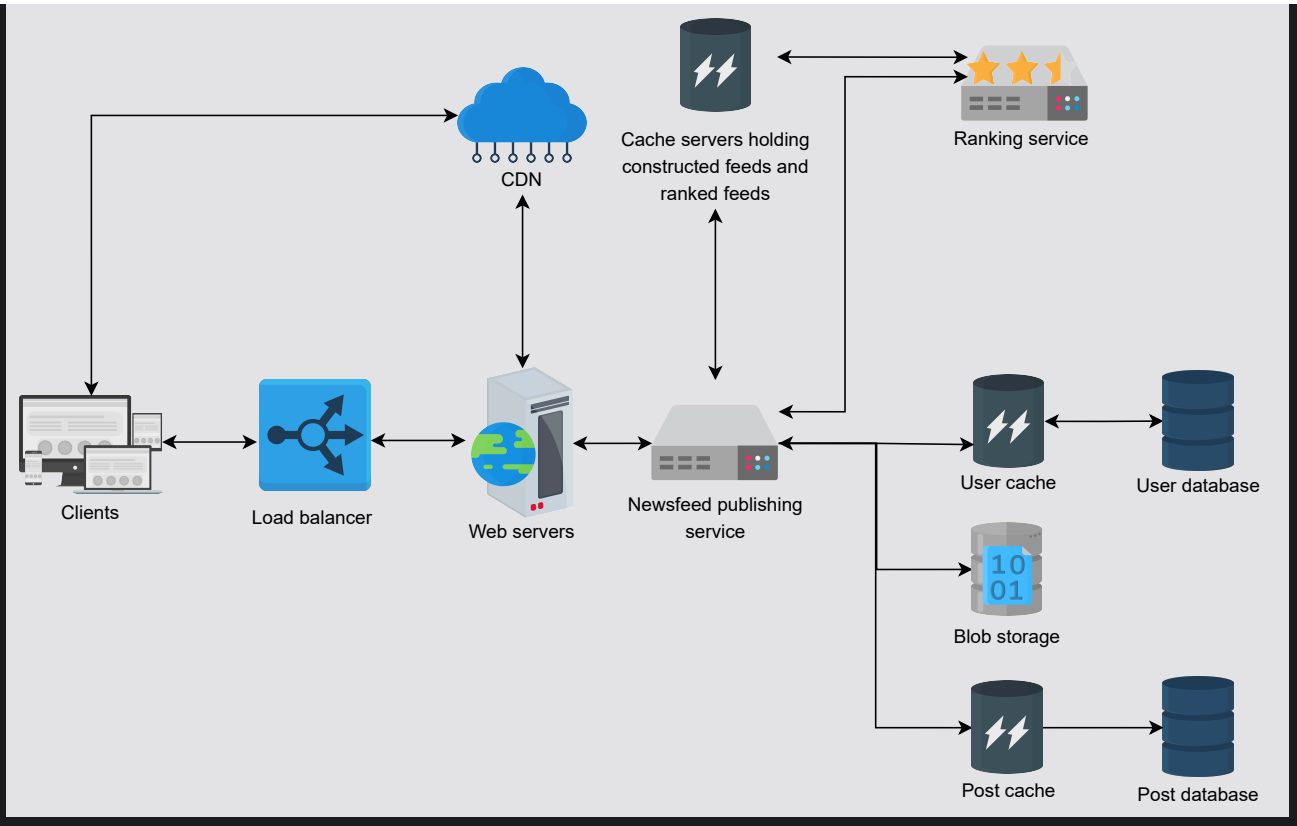
The newsfeed ranking service

Often we see the relevant and important posts on the top of our newsfeed whenever we log in to our social media accounts. This ranking involves multiple advanced ranking and recommendation algorithms.

In our design, the **newsfeed ranking service** consists of these algorithms working on various features, such as, a user's past history, likes, dislikes, comments, clicks, and many more. These algorithms also perform the following functions:

- Select "candidates" posts to show in a newsfeed.
- Eliminate posts including misinformation or clickbait from the candidate posts.
- Create a list of friends a user frequently interacts with.
- Choose topics on which a user spent more time.

The ranking system considers all the above points to predict relevant and important posts for a user.



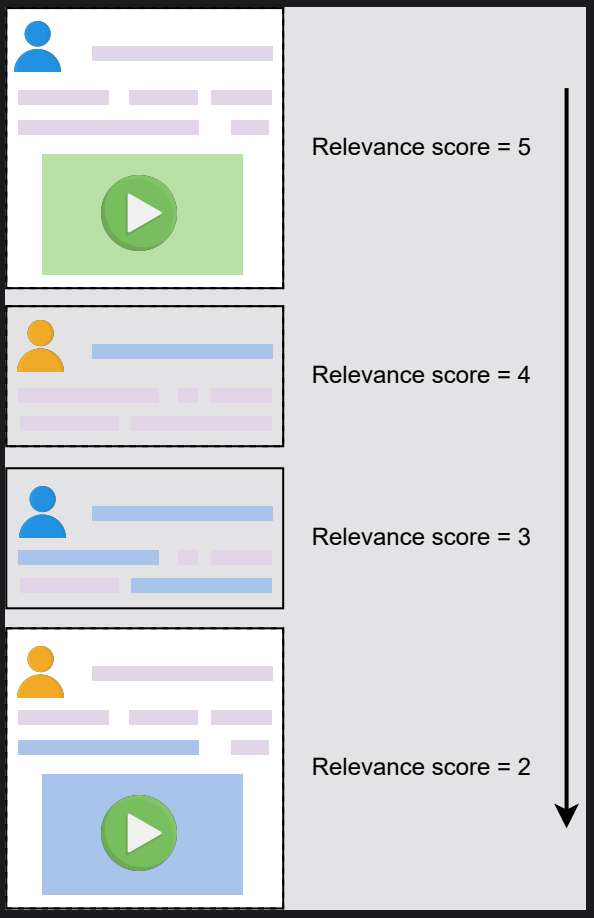
The ranked newsfeeds are stored in the cache servers

Posts ranking and newsfeed construction

The post database contains posts published by different users. Assume that there are 10 posts in the database published by 5 different users. We aim to rank only 4 posts out of 10 for a user (say Bob) who follows those five different users. We perform the following to rank each post and create a newsfeed for Bob:

1. Various features such as likes, comments, shares, category, duration, etc and so on, are extracted from each post.
2. Based on Bob's previous history, stored in the **user database**, the relevance is calculated for each post via different ranking and machine learning algorithms.
3. A relevance score is assigned, say from 1 to 5, where 1 shows the least relevant post and 5 means highly relevant post.
4. The top 4 posts are selected out of 10 based on the assigned scores.
5. The top 4 posts are combined and presented on Bob's timeline in decreasing order of the score assigned.

The following figure shows the top 4 posts published on Bob's timeline:



A newsfeed consisting of top 4 posts based on the relevance scores

Newsfeed ranking with various machine learning and ranking algorithms is a computationally intensive task. In our design, the **ranking service** ranks posts and constructs newsfeeds. This service consists of big-data processing systems that might utilize specialized hardware like graphics processing units (GPUs) and tensor processing units (TPUs).

> **Note: According to Facebook:**
>
> "For each person on Facebook, we need to evaluate thousands of features to determine what that person might find most relevant to predict what each of those people wants to see in their feed."
>
> This implies that we need enormous computational power and sophisticated learning algorithms to incorporate all the features to get good quality feed in a reasonably short time.

Putting everything together

The following figure combines all the services related to the detailed design of the newsfeed system: