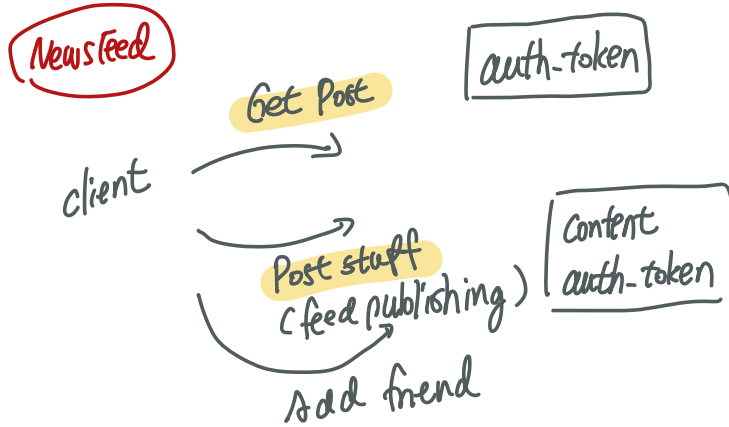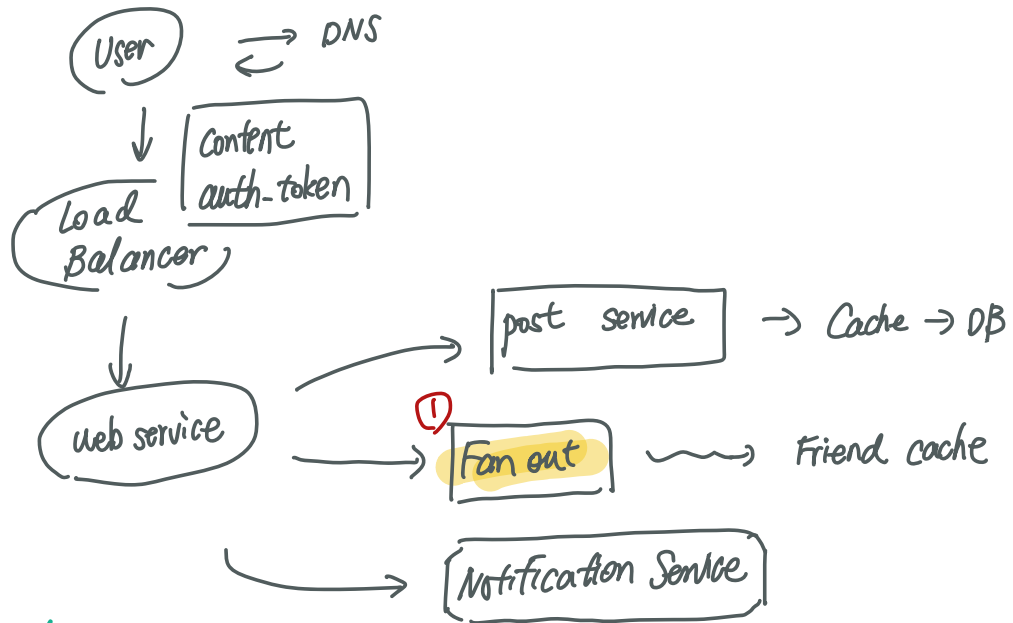Features:
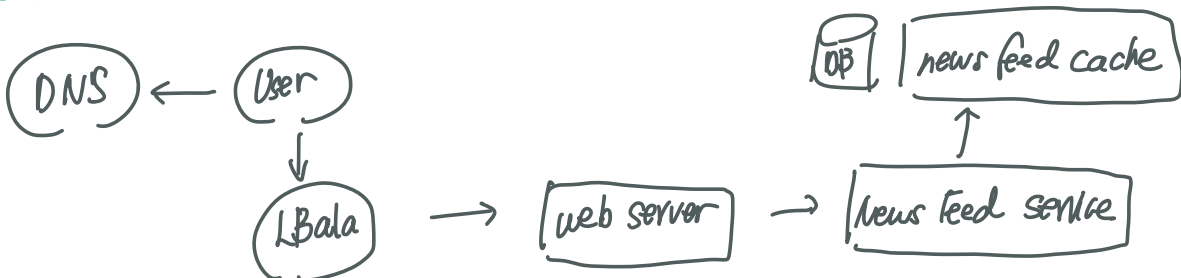
1). feed publishing: write into cache database ⟹ populate to friends
2). news feed building: aggregate friends' post in reverse chronological

News Feed

Get Post

auth-token

client

Post stuff
(feed publishing)

Content
auth-token

Add friend

1). Feed Publishing:

User ⇄ DNS

↓

Content
auth-token

Load Balancer

↓

web service

→ post service → Cache → DB

→ (1) Fan out ⤳ Friend cache

→ Notification Service

2. News retrieval

DNS ← User

↓

LBala → web server → News Feed service

DB | news feed cache

↑

① **Fan out:** Deliver post to all friends.
- Fan-out on write: pulish 以后立刻 publish
- Fan-out on read: read 时再 generate
.

In the world of message-oriented middleware solutions, **fan-out** is a messaging pattern used to model A model an information exchange that implies the delivery of a message to one or multiple destinations possibly in parallel, and not halting the process that executes the message to wait for any response to that message

- **Fan-out on write:** pulish 以后立刻 publish

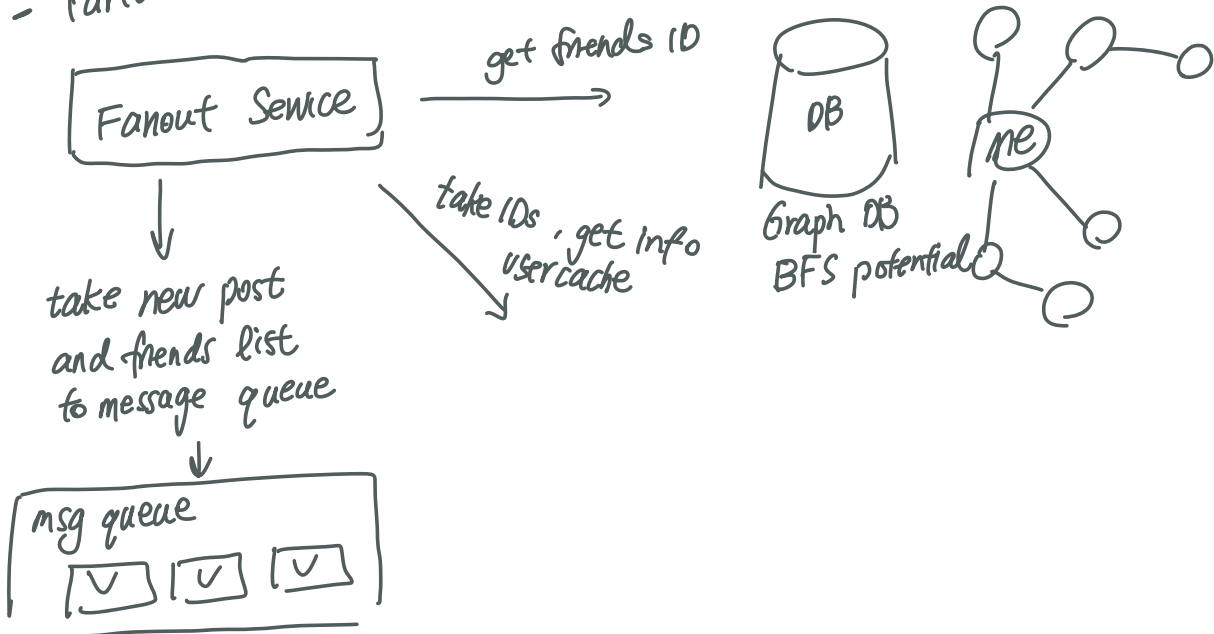 Pros:→ 好处是 push 快
  Cons: 如果人很多, 就慢得不行了.
- **Fan-out on read:** read 时再 generate
- . – Pros: if there's lots of inactive users, this saves compute resource.
  – Cons: you have to fetch it upon querying

[BIG Q.] how mix them?

For celebrity users are
– Fanout on read,

For nomal user:
– Fan out on write.

get friends ID

Fanout Service →

take IDs, get info
user cache

DB

Graph DB
BFS potential

me

take new post and friends list to message queue

↓

msg queue
[∨] [∨] [∨]

worker can look like
post-id, user-id
post-id, user-id

*
work append
id into this.

Can be elaborated.

News retrieval.

* for fast retrieval
of image, videos.

CDN

DNS ← User

Get Request

LB

/v1/me/feed

web server | Authentification
Rate limit

News feed Service →(Get posts)→ user cache → user DB

News feed Cache

post cache → Post DB

DB | news feed cache
↑
News feed service →