Peaks system design report

This is my report for Peaks' system design assignment. The assignment describes requirements for a simple messaging service, in essence quite similar to a basic version of Twitter. As such, I will be using Twitter's terms such as the word 'timeline' throughout this document for ease of communication. I will define two timelines: a 'user timeline', which contains all the tweets that a user has sent, and a 'home timeline', which is a collection of all the messages by the people that you follow.
Since this system is quite similar to a basic Twitter in requirements and functionality, significant inspiration was taken from the design of Twitter itself. Most importantly, this document (link) describing an early architecture of Twitter was heavily used as a source.

Summary

The core functionality of this system is to ingest messages from producers, and distribute these messages to the home timelines of many consumers who follow them. The main challenge here is not handling the content of the messages, or the viewing of this content, but rather how to place new messages on a consumer's home timeline in an efficient way.
To make things simpler, we split up a message into its metadata and content directly after ingestion. The content will go straight to CockroachDB to be stored persistently, while the metadata is passed on for further processing to ultimately place the message in a consumer's home timeline.
The metadata will initially enter a RabbitMQ pub/sub exchange. From here, it is written directly to the producer's user timeline in Redis. It is also passed to a service which is responsible for updating the consumers' home timelines.
This service will retrieve all the followers of the publisher, and create tasks in a RabbitMQ work queue for every follower. These tasks will be responsible for adding the message metadata to the followers' home timelines, also in Redis.
When a user retrieves a timeline, a service will populate all the metadata from Redis with the content from CockroachDB, and then return this full timeline to the user.
((TODO: HTTPS/API keys))

Architecture