|  |  |
| --- | --- |
| **Name:** | **Chi jao** |
| **Email:** |  |

**Question 1**

1. Name a (or more) cryptographic algorithm you would use to perform the following
   1. Encryption

* AES
* RSA
  1. non repudiation
* MD5
* SHA256
* + private key
  1. no tempering
* MD5
* SHA256

1. Name the following JWT registered claim names (see <https://tools.ietf.org/html/rfc7519#page-9>)
   1. unique JWT identifier

**"jti" (JWT ID) Claim**

* 1. cannot be used before a certain date

**"nbf" (Not Before) Claim**

* 1. issue date

**"iat" (Issued At) Claim**

* 1. token recipient

**"aud" (Audience) Claim**

* 1. mobile number

This is a claim. Since this is proprietary info, it goes into the payload

**Question 2**

You are developing a hotel reservation application. After your user have successfully booked a hotel, the application can (opt in) update a user’s Google calendar with the stay’s detail and alerts. The application needs to create, update and delete calendar entries.

What are the required steps to allows the reservation system to update the a user’s calendar?

* Enable API in google cloud
* Create API key and enable in app
* Authenticate user to login to application
* Get authorisation token from Google calendar
* Check for token validity
* Check if has access to resource
* Check policy if able to perform the action

**Question 3**

You and a few friends have co-founded a hot social media startup. Like any good social media startups, you will need a new feed. A news feed is a list of post that is constantly updated with stories, activities, polls, etc from your friends. A post content includes the poster, text, images, videos, simple questionnaires, links, locations, etc.

The post will also include likes, the number of people reacted to it.

Two REST endpoints have been designated for users to publish and retrieve their feeds.

Publish a post

POST /api/v1/feed/me

Retrieving a feed

GET /api/v1/feed/me

The endpoints are secured with JWT.

Each user of your social site can have up to a total of 1000 friends/followers. You anticipate 5 million daily active users with about 70% of them posting at least 1 post.

Design a system that will support your REST endpoints along with the given requirements. Be as details a possible with your design.

A whiteboard with writing on it

Description automatically generated with medium confidence

* Split post into different constituents
  + Static
    - Media: s3 bucket (with CDN)
    - (hardly updates) Post: document db e.g. mongodb, in json format indicating
      * Userid
      * Date
      * Text
      * mediaID (to reference to the media in s3)
  + Dynamic
    - like: store in Redis (has a list/map datastructure, key-value db)
      * postId
      * comment list
      * like
      * happy
      * sad
      * list of followers (to know who to post to)
      * …
* Put poster into queue (i.e. user A posts this post)
* Workers (could be using lambda) reads off this queue, go to graph db and look up all the user’s followers and add to follow’s feed list (e.g. using Redis)
  + Set timing for redis to delete ID in the list, e.g. after 10min can delete. When users scroll back, the feed list has to be regenerated
* Unique case scenarios
  + High follower users (e.g. celebrities)
  + Inactive accounts
    - Save time by not actively POST to user’s feed
    - Should pull instead (i.e. when follower logs in, then look up friends and pull post from there)
    - Downside of this is when the user opens the app, it will be slower

**Submission**

Copy this Word document to your repository and commit it.

git add .

git commit -m ‘worksheet03’

git push origin master