**Explanations of Software System Architecture**

Our software system is designed as a micro-services architecture. It helps to scale easily to handle a large amount of requests. With this design, different teams with different skills can join to develop it.

**The components in this design:**

- **Load balancer** is NGINX. It is powerful and we can configure it to limit the total requests per hour per client. It is useful to protect against DDoS attack and crawler tools. This load balancer distributes the incoming requests to the less busy instances of API Gateway.

- **API Gateway**: This is the entry service of our system. It has about 3 or 4 instances. It takes care of the followings:

* Security: configure https certificates
* Authentication: Sign Up, Sign In, Sign Out, Reset Password
* Authorization: Checking permissions of user when trying to do something on Post and Activity... (the resources owned by user)
* Adaptor: Consolidate requests to multiple micro services into 1 single call and return result to client
* Router and load balancer: to busy micro services. if a micro service is busy, we can make multi-copies of it. Use Round-Robin algorithm

- **Cache**: Using an instance of Redis database to cache user session information. Trending Posts, Analytics and Reports, Personalized Recommendations are cached for 1 hour. After it expired, we send request to those micro services. Because these queries take much time to process.

- **CDN**: Standing for **C**ontent **D**elivery **N**etwork. It is very useful and needed for streaming videos and quickly loading images. There are many providers such as Amazon S3 + Cloud Front, Google Cloud CDN, Microsoft Azure... However, I prefer using Cloudinary for easy to use and very useful for loading images with different sizes. Klook app is using it as well.

- **Database**: I choose using Elastic Search database. Its performance is very good for searching posts with full text search. And another reason is scaling. Grab app is using it as well.

- **Google Firebase**: I choose this for sending Push Notifications to clients. It has many advantages and built-in dashboard, report tool.

- **Micro services:**

1. **Timeline CRUD and Search**: Provide core functions in order to Create, Read, Update, Delete and Search posts. This micro service is often used. So we can consider to deploy it on 1 - 3 machines(instances).
2. **Personalized Recommendations**: Suggest users the related post that is suitable to user's habits, current location, interests.
3. **Trending Posts**: Find out the popular posts that have many interactions and attract users' attention.
4. **Analytics and reports**:

+ Analyze the user activities and generate reports about social interactions, trending keywords

+ Analyze logging information to generate reports about Traffic per day, per hour, per week or average response time

1. **Notifications**: Push notifications to users to engage user experience.

**Suggestions of hardware**

As requirement, our system needs to serve ***1000 requests per second***

- **Load Balancer**: 1 instance

+ OS: Ubuntu Linux 16.04

+ Software: NGINX

+ RAM: 16Gb

+ CPU: Intel Xeon E5-2673 V3 2.40 GHz / 30MB / 12 Cores 24 Threads/ Socket 2011

+ Network: 2x Intel XL710 40 GbE QSFP+ (rev 01)

+ SSD: 256Gb

- **API Gateway**: 3 instances

+ OS: Ubuntu Linux 16.04

+ Software: We can write the software by using GoLang, Elixir(Phoenix framework), Swift(Perfect framework)

+ RAM: 32Gb

+ CPU: Intel Xeon E5-2673 V3 2.40 GHz / 30MB / 12 Cores 24 Threads/ Socket 2011

+ Network: 2x Intel XL710 40 GbE QSFP+ (rev 01)

+ SSD: 256Gb

- **Micro-services**: 6 or 8 instances

+ OS: Ubuntu Linux 16.04

+ Software: We can write the software by using GoLang, Elixir(Phoenix framework), Swift(Perfect framework)

+ RAM: 32Gb

+ CPU: Intel Xeon E5-2673 V3 2.40 GHz / 30MB / 12 Cores 24 Threads/ Socket 2011

+ Network: 2x Intel XL710 40 GbE QSFP+ (rev 01)

+ SSD: 256Gb

- **CDN**: The provider is Cloudinary

- **Elastic Search**: The provider is [https://www.elastic.co](https://www.elastic.co/)

**Programming languages and Frameworks**

1, **GoLang**: For good performance and ability to handle a large amount of concurrent requests. Suggest for building API Gateway. Because we need to calculate SHA256 Hash, encode, decode data

2, **Elixir / Phoenix framework**: Running on Erlang virtual machine. Ability to handle a large amount of concurrent requests. Elixir is a dynamic, functional language designed for building scalable and maintainable applications. Suggest for building Timeline micro service.

3, **Swift / Perfect framework**: Not only for writing iOS client apps. We can use it for writing server by using the Perfect framework. Swift has a good performance in calculating. So I suggest Swift for building the micro service "Analytics and Reports"