Load

10 M users per month with 300 translations / month => 3 Billion translations per month

1Month = 2628 * 10^3 seconds ~= 3000 * 10^3

3Billions translations per month => per second calculation
(3 * 10^9)
—---------- = 1000 requests / second
(3000 * 10^3)

Infrastructure Estimation

UserCreation service and History Service (refer diagram in the later section) both would have to do 1 DB call at the least. This is assuming there's a cache miss or there's no cache at all, i.e worst case scenario. Assuming the DB's are co-located geographically in the same Data Centeres (DC), i.e when the app server is say located in the Europe region, the DB also resides in the Europe region, the Round trip time would ~ 100ms, With app logic taking some time, I'd estimate roughly 500ms for each request.

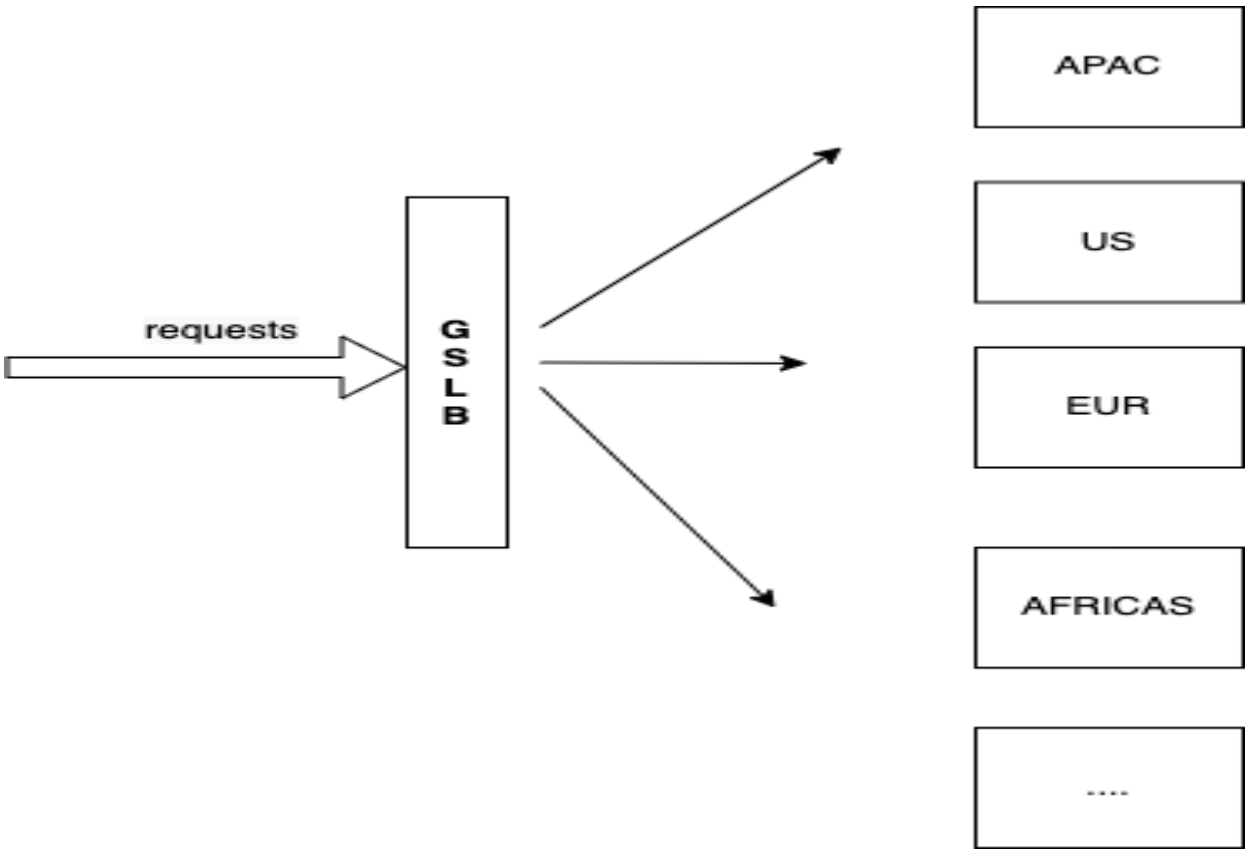
Assuming 1JVM process runs in a server with 100 threads (from what I've seen PROD machines run without much thread swapping, off course this number is subject to change depending on the host), the number of hosts required would be as follows,

100 threads for processing
500 ms for 1 request
200 requests can be served by 1 running instance of a host

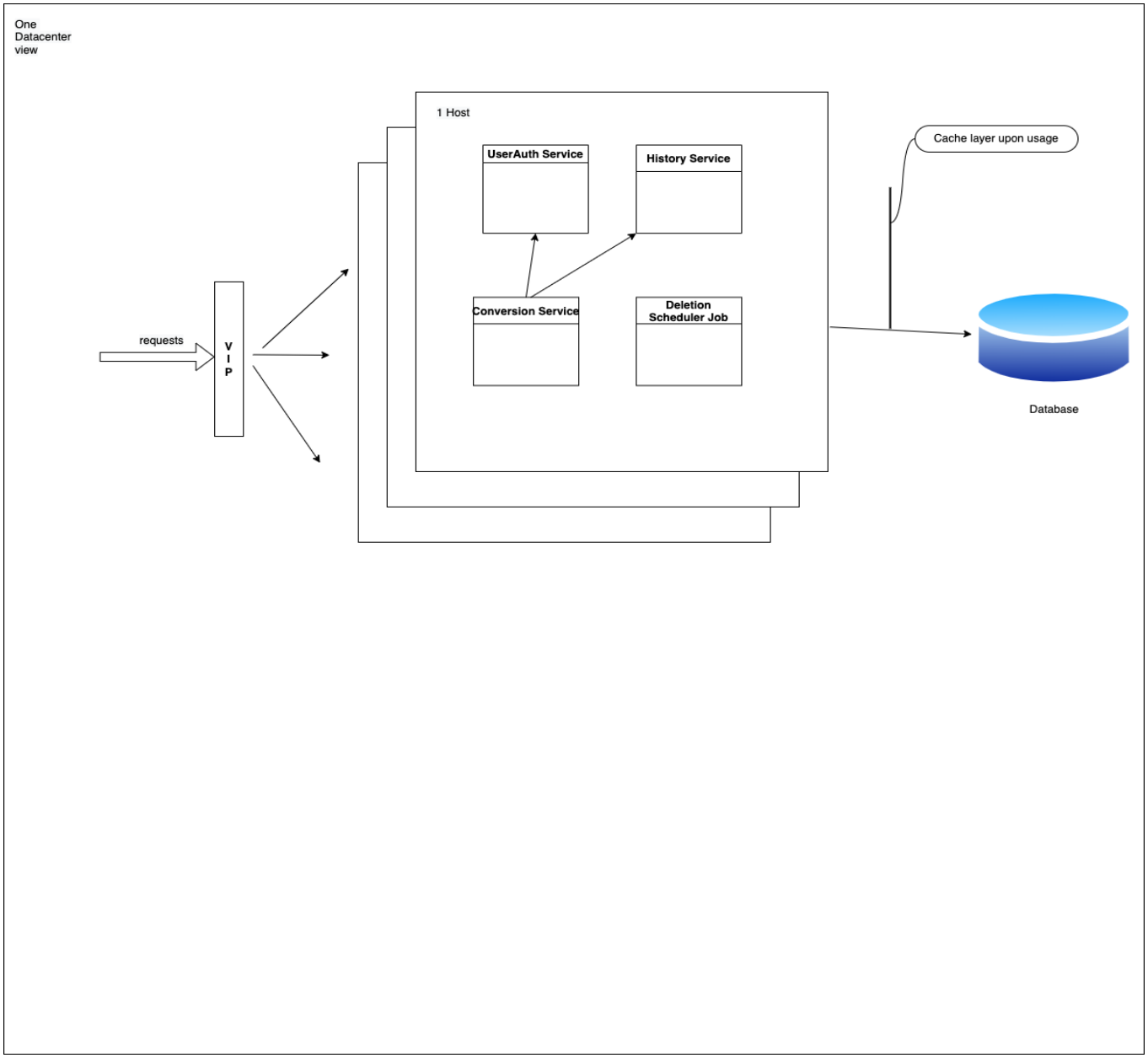
From our Load estimation, we know we'd have to be ready to server 1000 requests / second, so
1000
—---- = 5 hosts would be needed
200

Addressing Geographic loads

In order to address geographic load balancing, we can use GSLB to distribute load across multiple regions/Data Centers. This way we can distribute load across the host fleet we plan to deploy.



System Components



Function Prototypes

Conversion Service

This service can be the facade which gets called for user registration, history retrieval from transaction table, etc and communicate with the various other components such as authentication and history retrieval services/methods in the system.

Deletion Job

This job can actually perform the hard deletion on the DB table. This can be run occasionally like, every month or so and look for data records that are > 30 day timestamp. This way of deleting occasionally is done to avoid creating a lot of churn in the DB table space. If there are too many holes created in the DB table, then the performance of the DB goes down when operating on those tables.

UserAuthentication Service

authenticateUser()
createUser()
deleteUser()

History Service

Since the results become invalid after 30Days, we'd have to make sure we're querying/using the right records that are < 30Days. This service could actually just be a method stub that performs this action of filtering. Based on the need, this could communicate with a cache layer to fetch the translated records instead of hitting the DB. The introduction of cache is contingent upon the cache hit ratio as we don't want to introduce an additional layer if we're not getting a good cache hit ratio.

User Table

userId	userName	Email Id	phoneNumber	usercreatedTime

Transaction Table

TransId	userId	transcreateTime