ANALYSIS AND REPORTING

Document your hardware configuration (CPU speed, number of cores, RAM, Disk etc.) and software stack (RDB and version, Redis and version, programming language, libraries used, etc.)

The hardware configurations are 2.3 GHz 8-Core Intel Core i9, 16 GB of memory using Jedis verison jedis-4.4.0-m2 . We implemented the application in Java using the Jedis client to connect to the Redis database called tweet_db1.

Report TWO numbers in units of API calls per second:

API Method	**API Calls/Second**
postTweet	142.4
getHomeTimeline	1247.3

Describe any factors you think might have impacted your results.

1. **Generating the timelines** of each follower of the user X when X posted a Tweet is a costly operation. It involved having to build two lists in Redis - one for the followers of X, and another list (T) to represent the timelines of the followers of X. Without this postToTimeline call, our postTweet call was averaging 4500 tweets/sec. postTweet also calls postToTimelines, which pushes all the tweets into the user's timelines.

2. **Generating unique tweet ID**: We used a Redis method, incr to create unique tweet_ids. This may positively benefits our runtime because it creates ids in constant time as opposed to our previous implementation in SQL that required a counter in Java.

3. **Retrieving pre-made timelines.** Since we are sacrificing the runtime of posting tweets to Redis by simultaneously creating timelines for their followers, we save time on retrieving the timelines since they are already stored in Redis.