Assignment 02 Design Document

A1885211 – Ethan Dankiw

**Network Topology**

A diagram of a company

Description automatically generated with medium confidence

**Multi-Threading and Thread Safety**

The load balancer has a thread pool of aggregation server, with each aggregation server having a thread pool of 100 to process requests concurrently.

I used concurrent data structures like AtomicLong (LamportClock) and ConcurrentMap (ContentStore) in Java without the use of mutex’s to ensure that there is no race conditions. Since I did not use any mutex’s, I utilised the synchronized keywork for queueing PUT requests to the content store to ensure that there are no dead locks, and all method calls are being processed or are waiting.

An incoming request to the Load Balancer, first checks for the least loaded Aggregation Server to send the request to. If all servers are at capacity, a new server is spawned to handle the request. The client socket is then sent to the server to be processed. The aggregation handles the client request on one of its 100 threads in its pool. This allows for a maximum of 1000 requests to be handled concurrently.

**Number of Server Replicas**

The number of replicas is dependent on server load as I have both a LoadBalancer and ServerScaler.