Design and implementation of an online Book store CS677 lab2 project

Group member: Shahrooz Pouryousef
University of Massachusetts Amherst
shahrooz@cs.umass.edu

ABSTRACT

This paper explains our implementation of a two-tier web design - a front-end and a back-end - and use microservices at each tier for an online book store. The front-end tier will accept user requests and perform initial processing. The backend consists of two components: a catalog server and an order server. The catalog server maintains the catalog (which currently consists of the above four entries). For each entry, it maintains the number of items in stock, cost of the book and the topic of the book.

1 RUNNING THE SYSTEM

There are two separate subdirectories for the distributed and single machine version of the system in the root directory of the project. In the distributed version of the system, the script for each microservice will be copied to the remote machine and then run. For the local and single machine versions of the system, the scripts will not be moved. Each version has its own script for measuring the end-to-end response time. You can set the IP address of each server in the "each_server_IP.txt" file.

2 EVALUATION

In this section, we evaluate the end-to-end response time and specific component-time using our implementation when servers are running on different machines or all on one machine

2.1 End-to-end response time

The goal of this experiment is to measure the end-to-end response time for each client's request. We repeat the experiment 1000 times. Each request identifies a run of the system where only one of the clients wants to send a request to the front-end tier microservice. In the distributed version of the system, we run each microservice on a different machine on Emulab servers [1]. We send the request to the front-end server from a machine in the UMASS CICS building.

On the single machine version of the system, we run all of the microservices on our local machine each with a different IP address.

Figure 1 shows a CDF on the end-to-end response time cross all requests in our experiment. For the case that each microservice is running on a different machine, the end-to-end response time for around %99 of our transactions is around 100 milliseconds. For this case of the evaluation, as the propagation delay is high in comparison to the processing delay on each microservice, the end-to-end response time is mostly the propagation delay. However, the DELETE request that is about buying an item has a higher end-to-end response time in comparison to other requests. The reason is that for DELETE requests, the order microservice has to communicate with the catalog server to first check if there is enough number of that item and then ask it to update its records.

The end-to-end response time for the case that we have used one single machine for all the microservices is shown in figure 1b. We do not have a high delay of propagation delay for this case.

2.2 Component-specific response time

The goal of this experiment is to check the component-specific response time. For this experiment, each microservice is running on a different physical machine each with 8 cores.

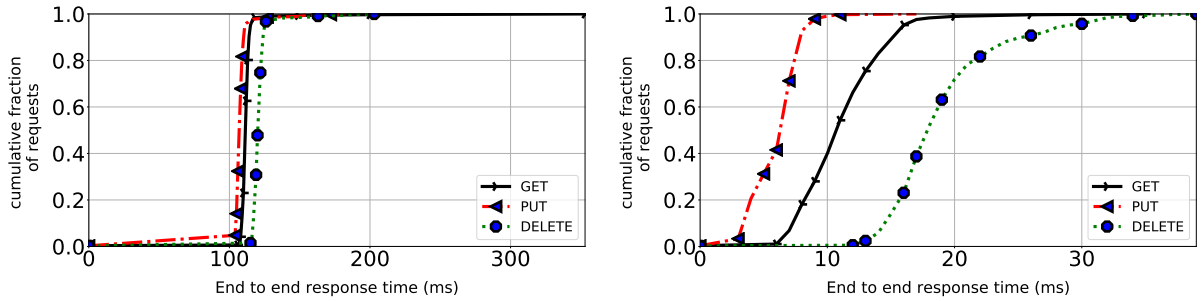
Figure 2 shows the average processing time for each type of request on our catalog and order microservices. Most of the requests have a processing delay of 1 or 2 milliseconds. As expected, the processing delay of a request is higher when we run all the microservices on a single machine.

3 AN ELECTRONIC COPY OF OUTPUT

Figure 3 shows an electronic copy of the output generated by running my program.

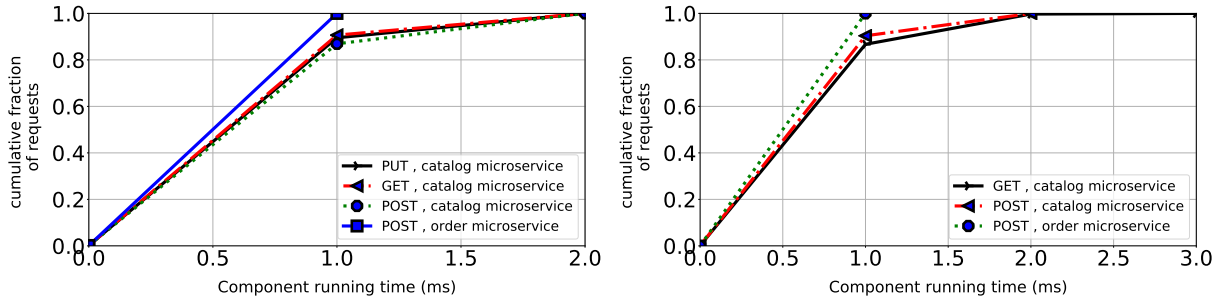
REFERENCES

[1] education A platform for research, development in distributed systems, and networks. 2021. https://www.emulab.net/portal/frontpage.php. [Online; accessed 20-March-2021].



(a) Each microservice running on a different machine **(b) All microservices running on a single machine**

Figure 1: A CDF on the end-to-end response time. In figure 1a the end to end response time is dominated by the propagation delay from CICS building to the Emulab servers that our microservices are running.



(a) Each microservice running on a different machine **(b) All microservices running on a single machine**

Figure 2: Processing delay of each request on each microservice. The processing delay of a request is higher when we run all the microservices on a single machine

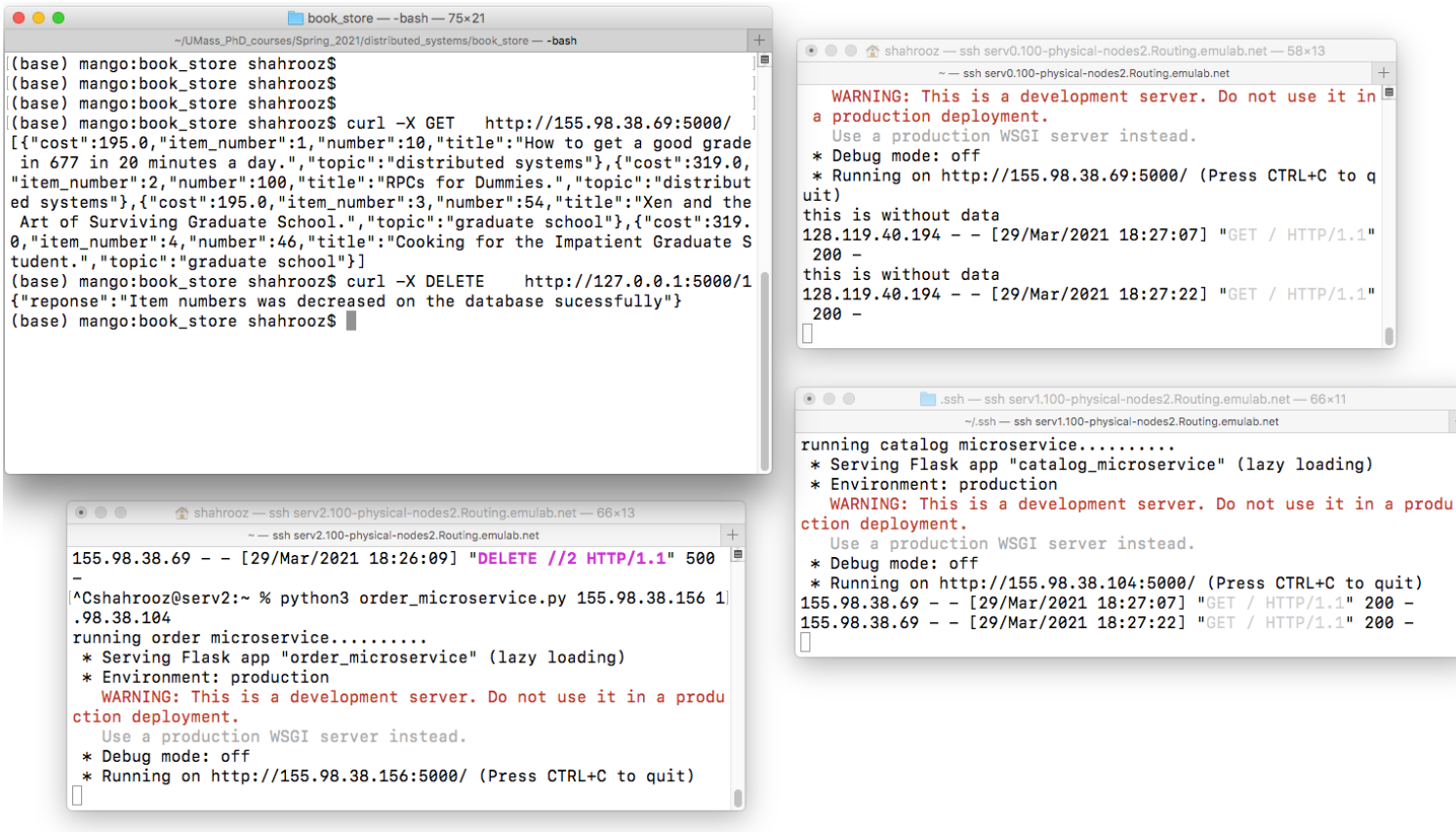


Figure 3: An electronic copy of the output