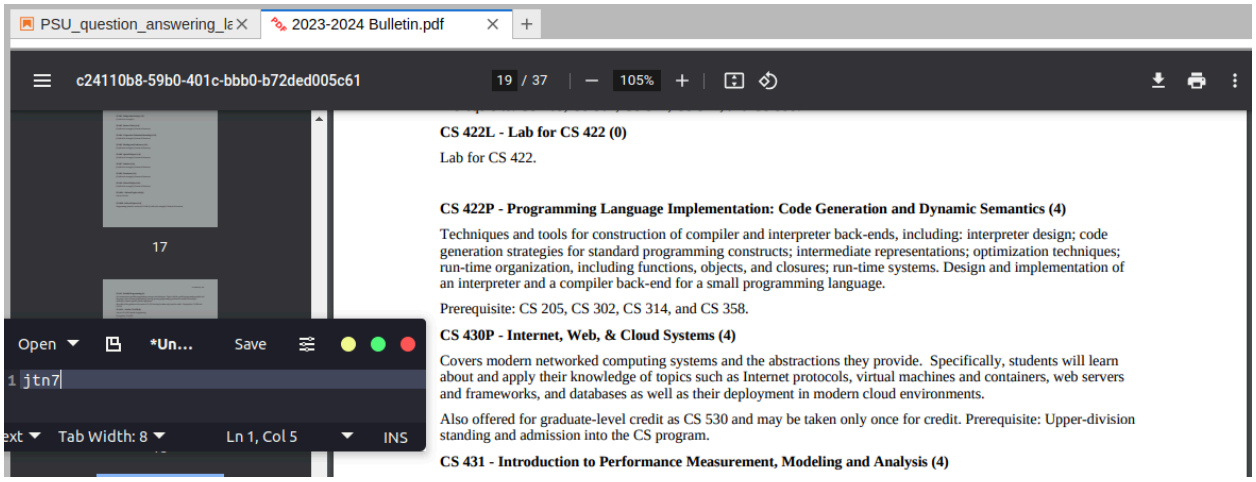
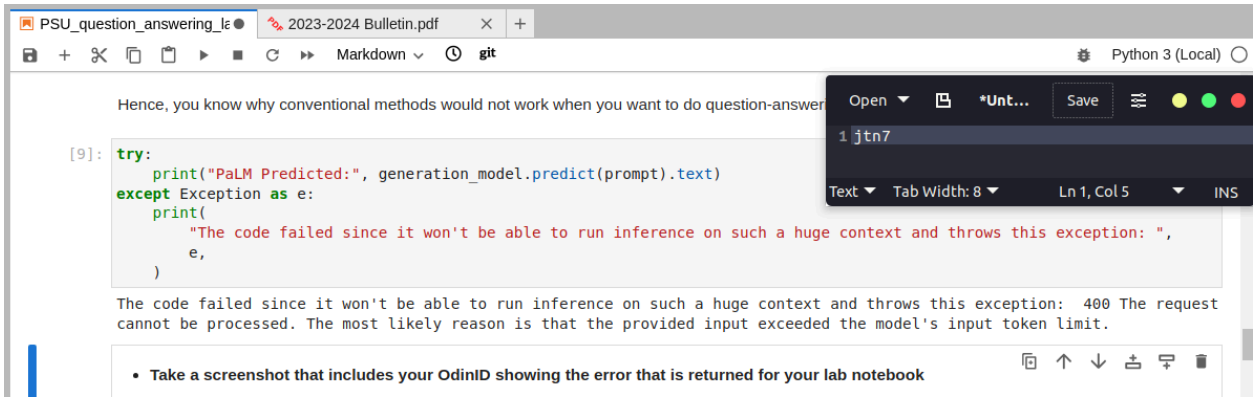
10.1g

- Take a screenshot that includes your OdinID showing the page number and the description of the class for your lab notebook



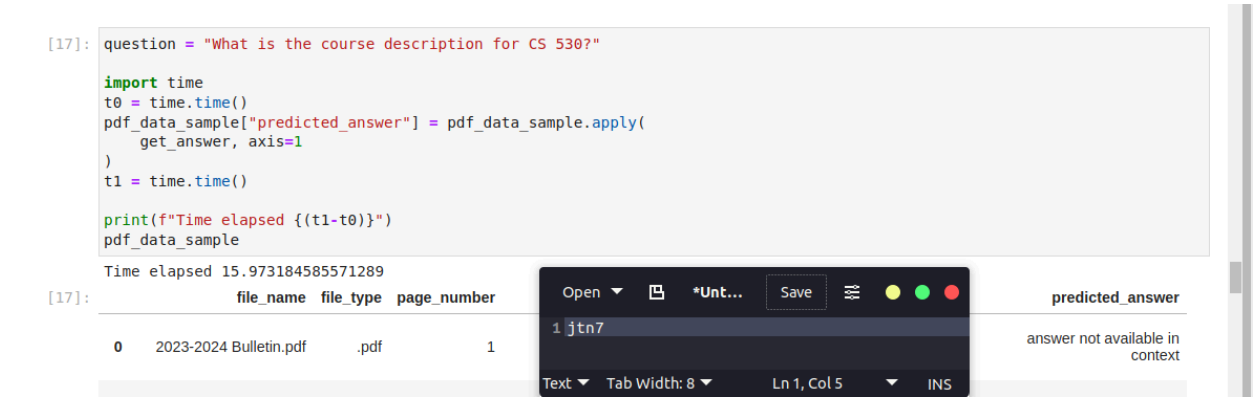
- Take a screenshot that includes your OdinID showing the error that is returned for your lab notebook



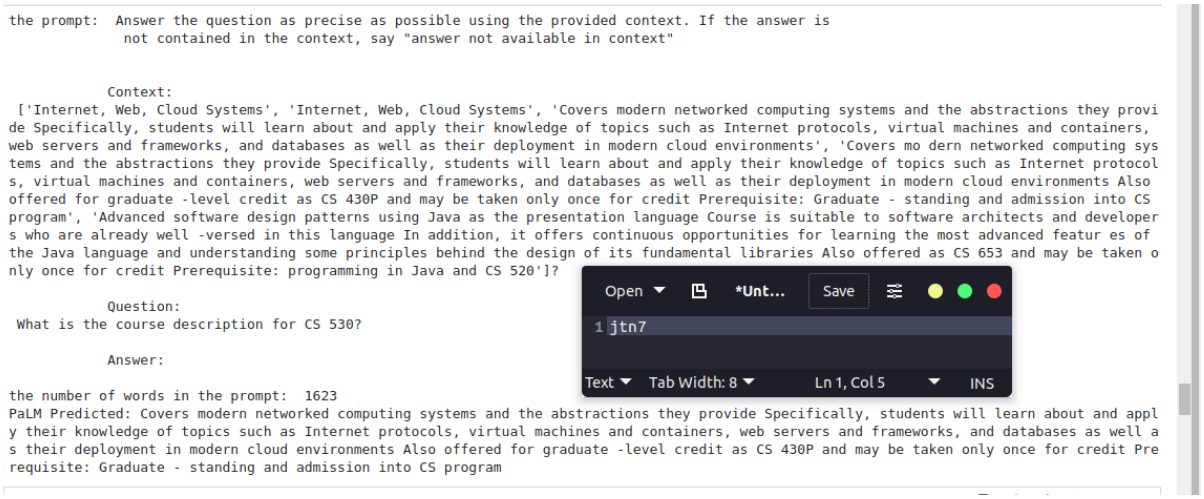
- Provide an explanation as to why the description is not returned for your lab notebook

The question passed in was asking about the course description for CS 530 but the first 5000 or so words in the document did not contain the answer to that question, thus it resulted in the program telling us that it did not find the answer.

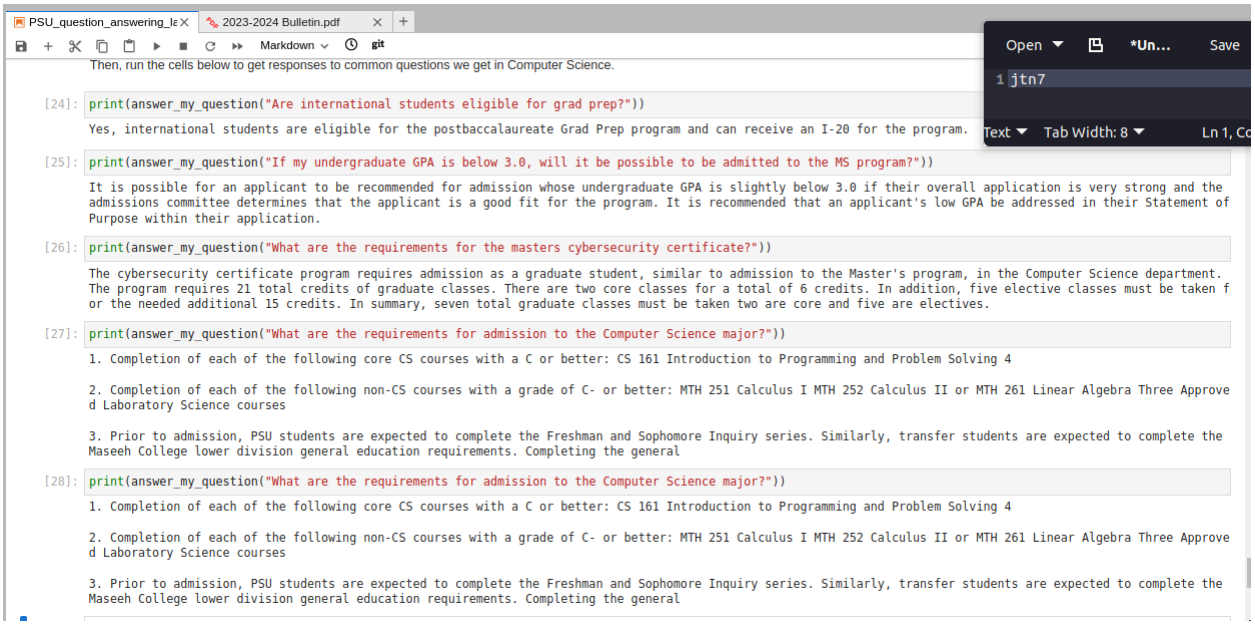
- Take a screenshot including your OdinID that shows how long it took to perform the prediction across every chunk



- How many chunks returned predictions? **5**
- Take a screenshot that includes your OdinID showing the result that is returned for your lab notebook



- Take a screenshot including your OdinID that shows the results of the queries



- Which of the approaches described would have issues with token limits on LLMs?

The first approach of stuffing where we did not set a limit to how many tokens to use and would pass all the data at once to query our question.

- Which of the approaches would result in the most queries for the LLM to handle? How many LLM requests are performed from a single user query in this approach?

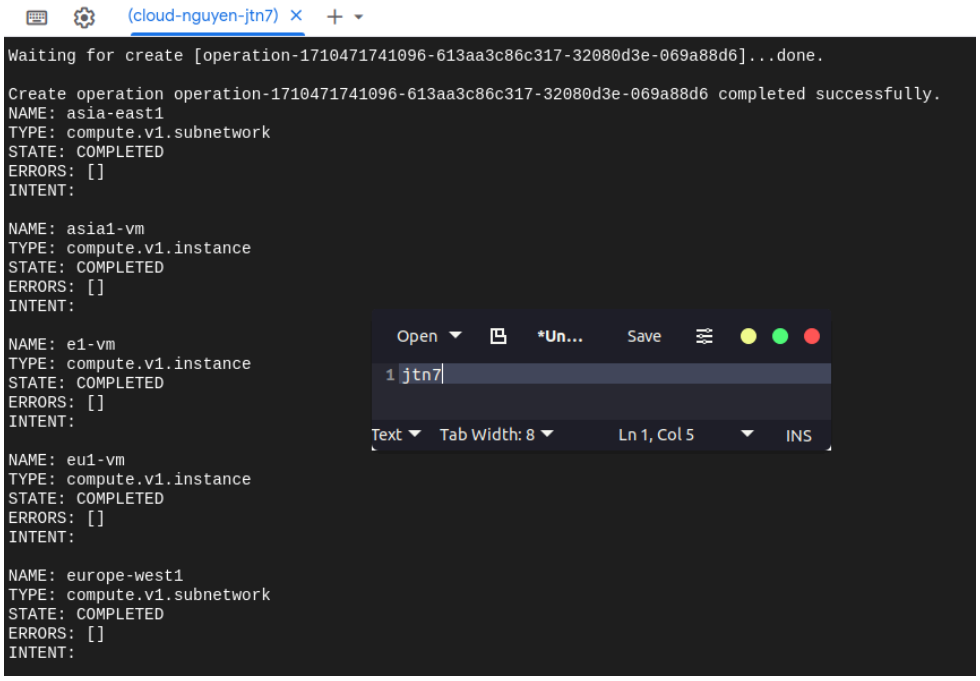
The second approach of map reduce where we had multiple api calls by querying each individual chunk had a lot of queries for the LLM to handle and this could slow down the process drastically especially if the data to query is very large.

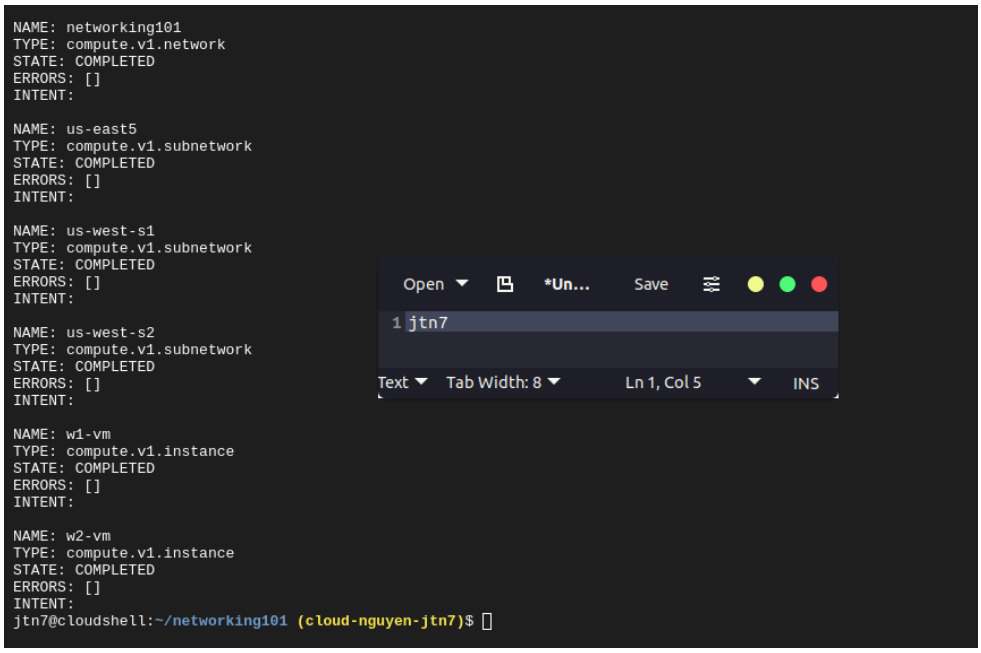
- Which of the approaches requires one to search a vector database for an appropriate context that is then sent to the LLM?

The map reduce with embeddings approach where we used an embedding model to create embedding from the chunks and then only passed in chunks that contained content similar to the question to the LLM when asking a question.

10.2g

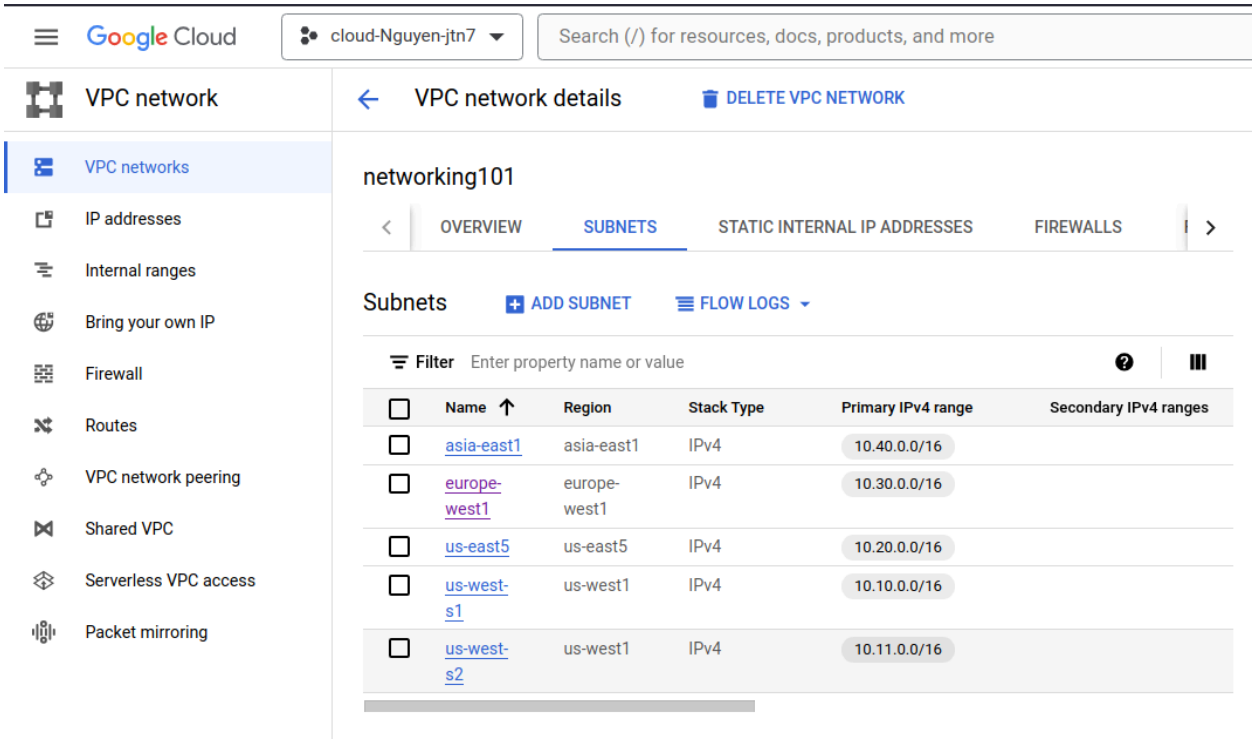
- Take a screenshot of the output to include in your lab notebook. How many networks, subnetworks, and VM instances have been created?



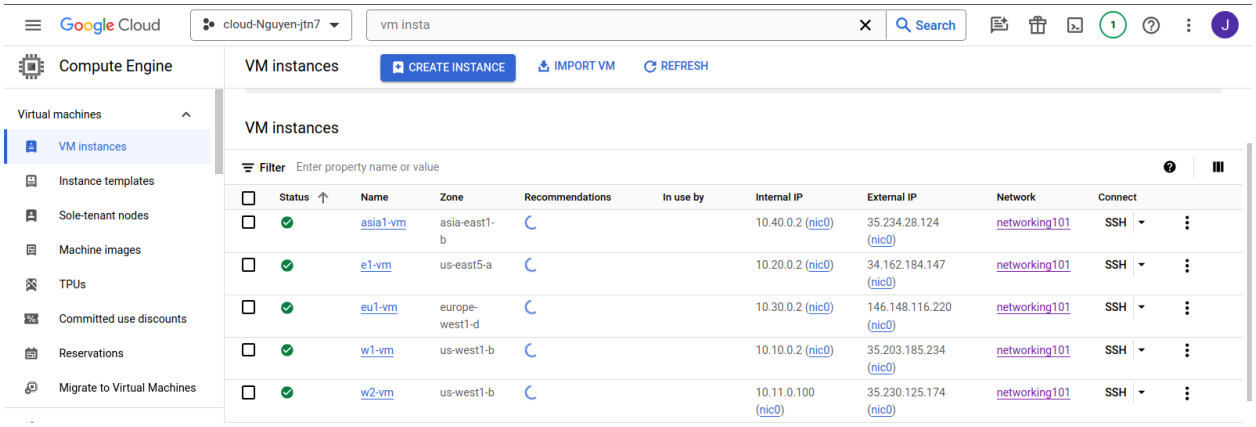


1 network and 5 subnetworks and 5 VM instances have been created.

- Visit the web console for VPC network and show the network and the subnetworks that have been created. Validate that it has created the infrastructure in the initial figure. Note the lack of firewall rules that have been created.



- Visit the web console for Compute Engine and show all VMs that have been created, their internal IP addresses and the subnetworks they have been instantiated on. Validate that it has created the infrastructure shown in the initial figure.

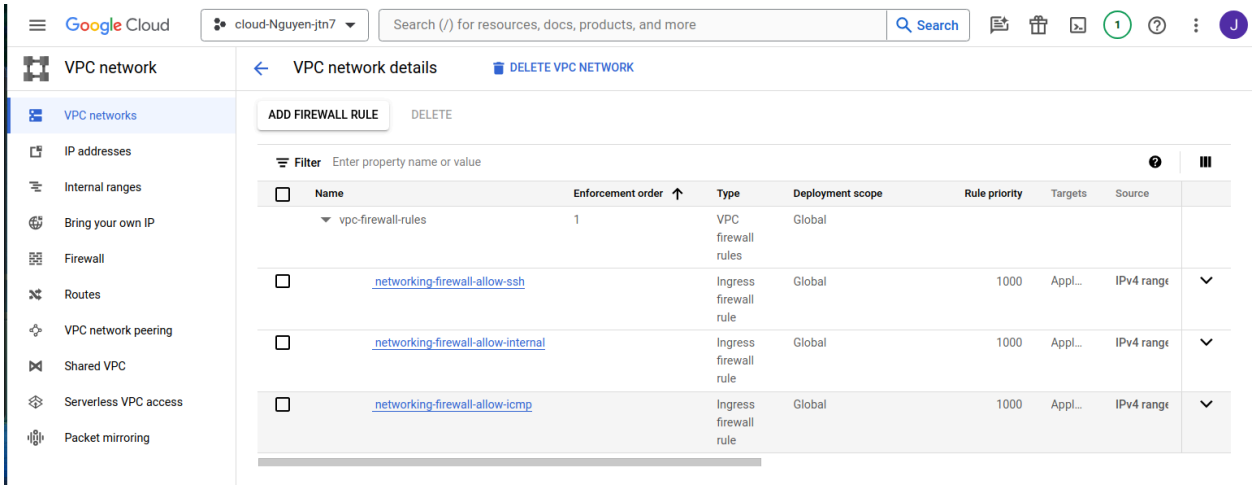


- Click on the ssh button for one of the VMs and attempt to connect. Did it succeed?

No, the connection failed.

Visit the networking101 VPC network in the web UI

- Take a screenshot that indicates the new rules have been deployed



- Given this, fill in the table with the measured latencies between the 6 pairs and include it in your lab notebook. Use the shortest latency measured for each pair.

Location pair	Ideal latency	Measured latency
us-west1 us-east5	~45 ms	~53 ms
us-west1 europe-west1	~93 ms	~134 ms
us-west1 asia-east1	~114 ms	~119 ms
us-east5 europe-west1	~76 ms	~88 ms
us-east5 asia-east1	~141 ms	~169 ms
europe-west1 asia-east1	~110 ms	~251 ms

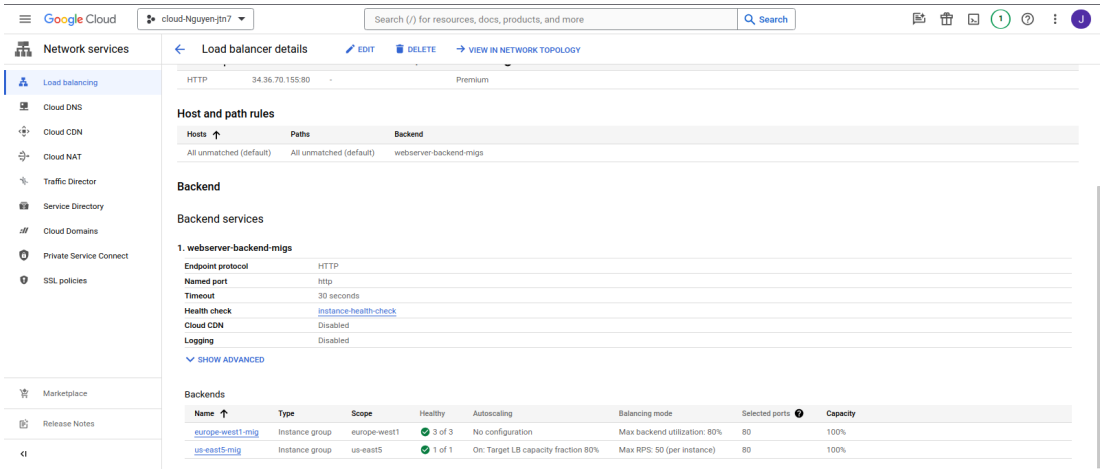
- Are the instances in the same availability zone or in different ones?

No, one is in us-east5-c while the other is in europe-west1-d zone.

- List all availability zones that your servers show up in for your lab notebook.

The us-east5 region only has one instance located in zone c while europe-west1 has 3 instances in zones b, c, d.

- Show a screenshot of the page that is returned. If you get an error, you may need to wait several minutes for the load balancer to finish deploying.



- Which availability zone does the server handling your request reside in?

us-east5-c

- Take a screenshot of the initial traffic distribution

Keep this window open for 5-10 minutes as the system adapts to the load and the UI updates.

- Take a screenshot of the UI as additional instances are brought up and show that the traffic distribution shifts

- Show a screenshot of the final traffic distribution.