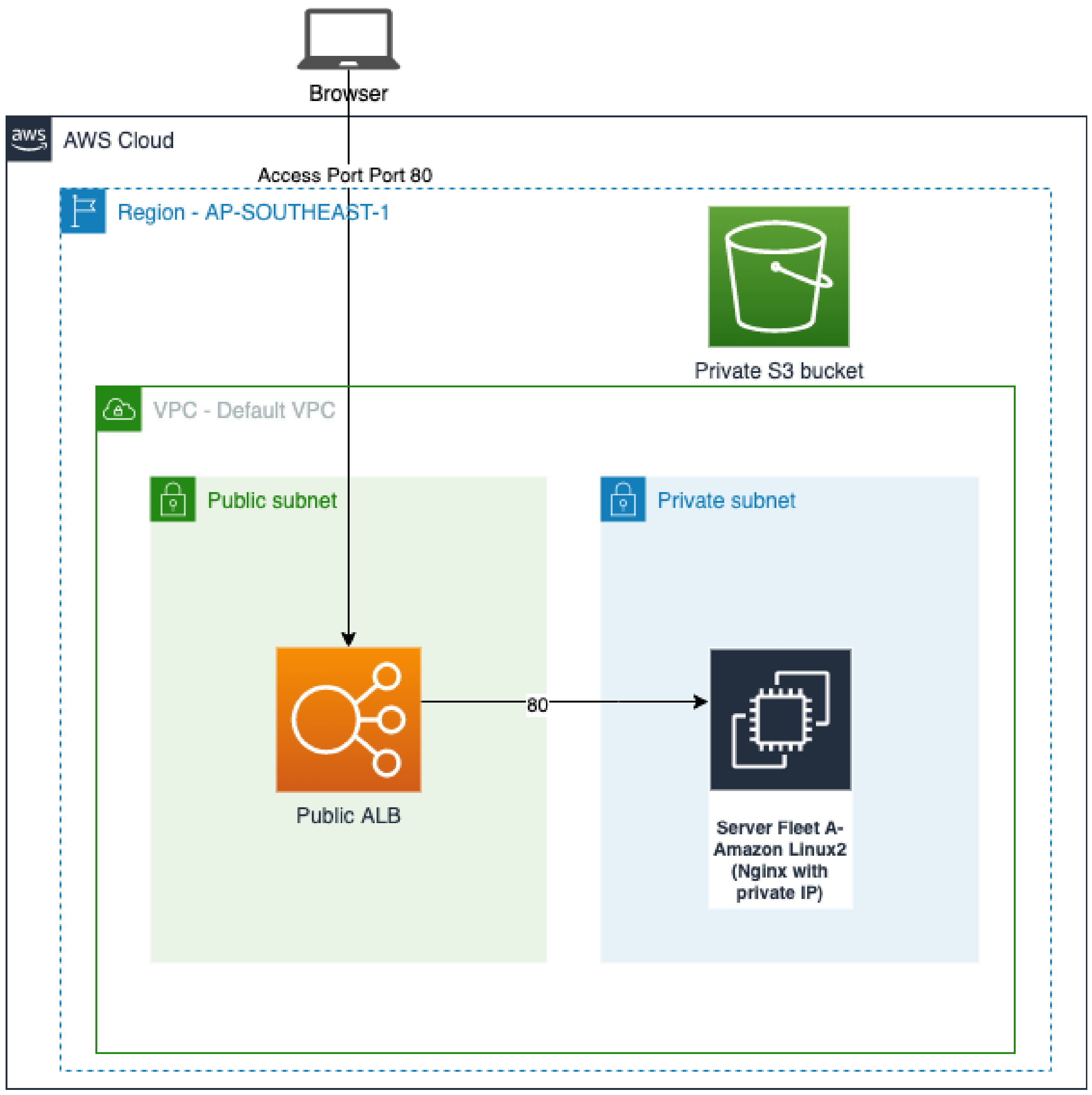
Scenario 1 - Terraform, AWS, CICD:

You've been tasked to set up an highly available auto-scaling group of 3 AWS EC2s (Server fleet A, t2.micro instance type) running nginx to serve public web content**,** behind a public ALB in the Singapore region based on the following network diagram using terraform. The EC2 servers when booting up should download its web content from a private S3 bucket. Use sample **Incremental and Decremental Counter** code from this website with this scenario:
https://www.geeksforgeeks.org/how-to-make-incremental-and-decremental-counter-using-html-css-and-javascript/

Next, set up a Gitlab pipeline which updates changes to website code stored in your Gitlab code repository to the above-mentioned S3 bucket upon merge to Master branch. Change all the text to upper case and push changes to S3 bucket using your pipeline. In your pipeline, trigger instance refresh of the autoscaling group of Server fleet A, after successful push of code to S3 bucket.
Ref: https://docs.aws.amazon.com/autoscaling/ec2/userguide/asg-instance-refresh.html

Note: You are encouraged to use the default VPC as it already comes with Internet Gateway, public subnet and default routing table. A private subnet would need to be created by your terraform code.



Deliverable:

- Restrict inbound access to both public ALB and Server Fleet A to only allow on port 80/TCP
- Install **Nginx** on Amazon Linux 2, Server Fleet A of EC2
- Able to view **Incremental and Decremental Counter** webpage from public internet served by Server Fleet A (Screenshot required for both the original page and modified uppercase text page)
- Describe in a README file, what further improvements to above architecture and setup would you recommend and why.
- Share the Gitlab pipeline with your interviewer
- Push your terraform, web codes and gitlab-ci.yml file to your gitlab repo to be shared with your interviewer

Scenario 2 - Kubernetes:

Your team decided that they want the **Incremental and Decremental Counter** to run on kubernetes and monitor the resources using Prometheus and Grafana. You have been tasked to deploy the **Incremental and Decremental Counter**, deploy a **Mysql server** to a Kubernetes cluster and monitor the resources. The **Incremental and Decremental Counter** to be able to communicate with your **Mysql server. (Create a table in a MySql database. Can be any table name)**

Deliverable:

- Make use of **scenario 1** code and display your <your first name> using an environment variable supplied to the container.
- Document in a README.md, the assumptions made when creating this site and provide us a script to spin up the Application using local kubernetes (**you can include any troubleshooting steps if any**).
- Ensure that the **Incremental and Decremental Counter** is able to connect to Mysql server. No code change is required from scenario 1.
- Monitor the resources using Prometheus and Grafana
- Push your code to your gitlab repo to be shared with your interviewer.

Note: Run your kubernetes cluster locally. You can use either Docker desktop kubernetes/minikube. You do not need to deploy it using gitlab for the CICD part.

Additional notes: You can also make use of helm chart for Prometheus and Grafana (eg. https://github.com/prometheus-community/helm-charts)