A blue and black logo

Description automatically generated

ASSIGNMENT COVER SHEET

**Lecturer’s Name:**  Ruth Lennon

**Assessment Title:** Mini-Network assignment

**Work to be submitted to:**  Ruth Lennon

**Date for submission of work:**  19/11/2023 \_\_\_\_\_\_\_

**Place and time for submitting work:** Blackboard

**Student’s Name:** Shreya Raghuvanshi, L00179092

**Class:**  IAC - DevOps

**Subject/Module:**  As Above

**Word Count (where applicable):**  N/A

**I confirm that the work submitted has been produced solely through my own efforts.**

**Student’s signature:** Shreya Raghuvanshi **Date:** 19/11/2023\_

|  |
| --- |
| **Notes**  **Penalties:** The total marks available for an assessment is reduced by 15% for work submitted up to one week late. The total marks available are reduced by 30% for work up to two weeks late. Assessment work received more than two weeks late will receive a mark of zero. [Incidents of alleged plagiarism and cheating are dealt with in accordance with the Institute’s Assessment Regulations.]  **Plagiarism:** Presenting the ideas etc. of someone else without proper acknowledgement (see section L1 paragraph 8).  **Cheating:** The use of unauthorised material in a test, exam etc., unauthorised access to test matter, unauthorised collusion, dishonest behaviour in respect of assessments, and deliberate plagiarism (see section L1 paragraph 8).  **Continuous Assessment:** For students repeating an examination, marks awarded for continuous assessment, shall normally be carried forward from the original examination to the repeat examination. |

# MiniNetwork - CloudFormation

# Aims/Description:

# CloudFormation:

CloudFormation is a service on Amazon Web Services that utilizes template files to automatically setup the AWS resources using the concept of Stacks. It can be used to automate configurations of workloads that run on AWS. All the resources required for application can be deployed using either JSON or YAML templates.

# Templates in CloudFormation:

Templates are text-based files which can be written in YAML Ain’t Markup Language or JavaScript Object Notation. They specify the AWS resources which a user wants to create such as S3 bucket, EC2 instance, and more. Templates define connections and interdependencies among resources enabling users to tailor resource settings when creating a stack which makes it dynamic for different use cases.

**The aims were:**

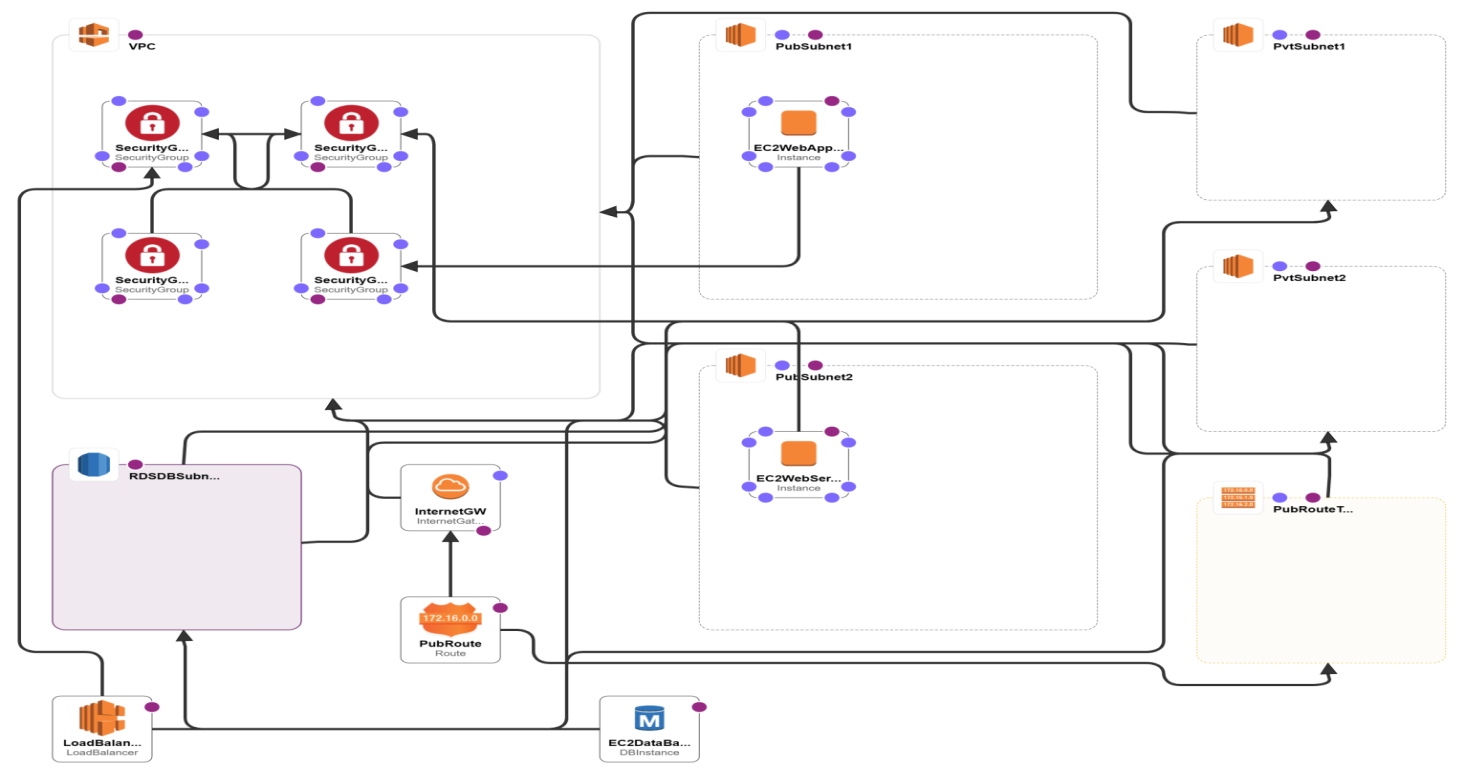
1. To establish an AWS CloudFormation based MiniNetwork utilizing AWS resources exclusively.
2. To craft an AWS template relying solely upon AWS documentation.
3. To produce a template that was both versatile and adaptable ensuring the company’s capacity to employ it seamlessly, even during expansion.
4. To incorporate best practices throughout the process.

# Method:

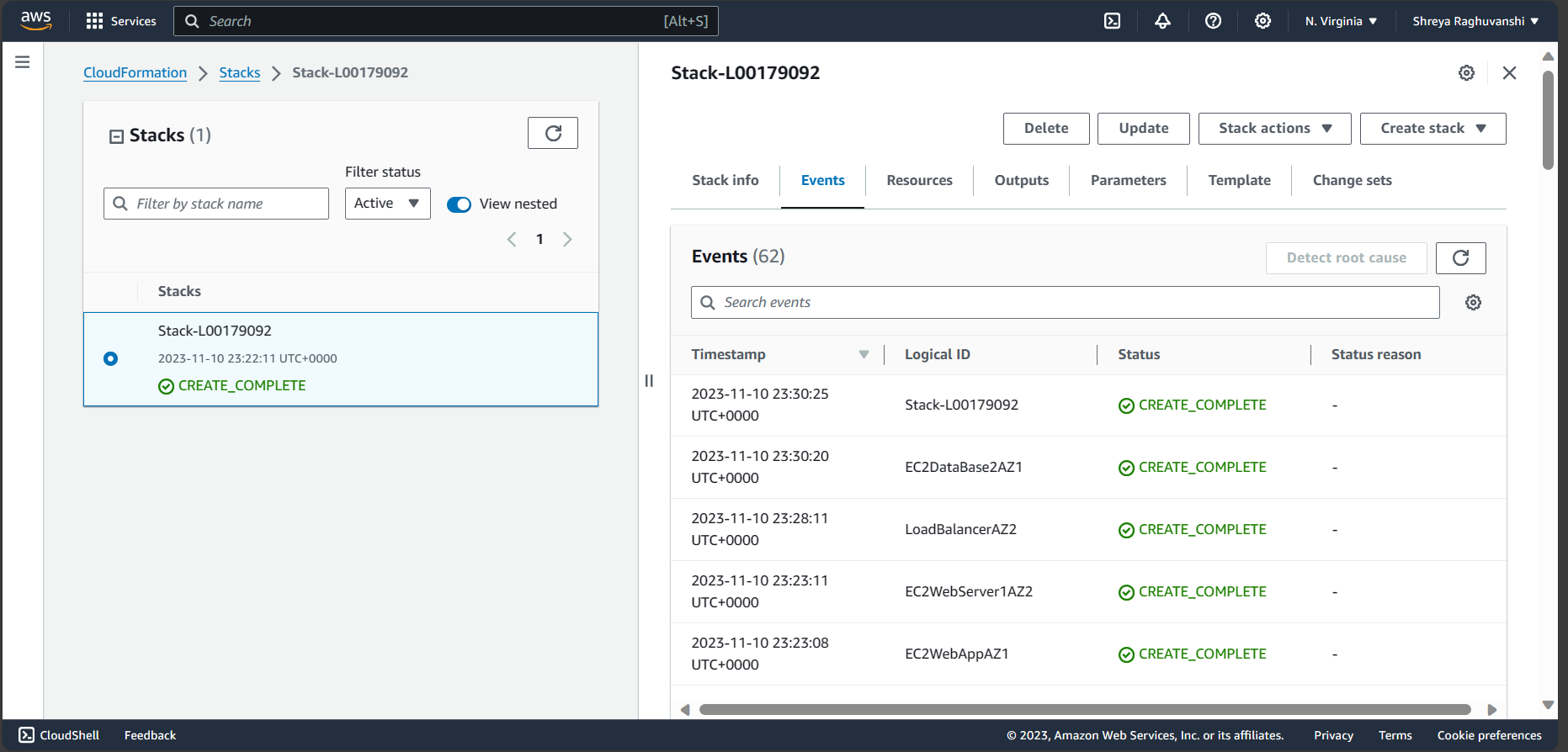
1. Objectives of CloudFormation infrastructure were defined and which AWS resources were needed was figured out. Factors such as performance, security and redundancy were considered.
2. The comprehensive physical architecture was designed including arrangement and interconnection of resources.
3. YAML template language was selected between JSON and YAML to be used for CloudFormation template.
4. IP Addressing scheme including subnets and address range was decided.
5. A YAML template was created including Parameters, Mappings and Resources sections. The YAML file was then saved on VS code and uploaded on AWS. Next, the template validated and diagram was made (Refer fig 1.1).
6. Moving ahead, a new Stack was created using the existing template (Refer fig 1.2).

# Results:

1. MiniNetwork Diagram:

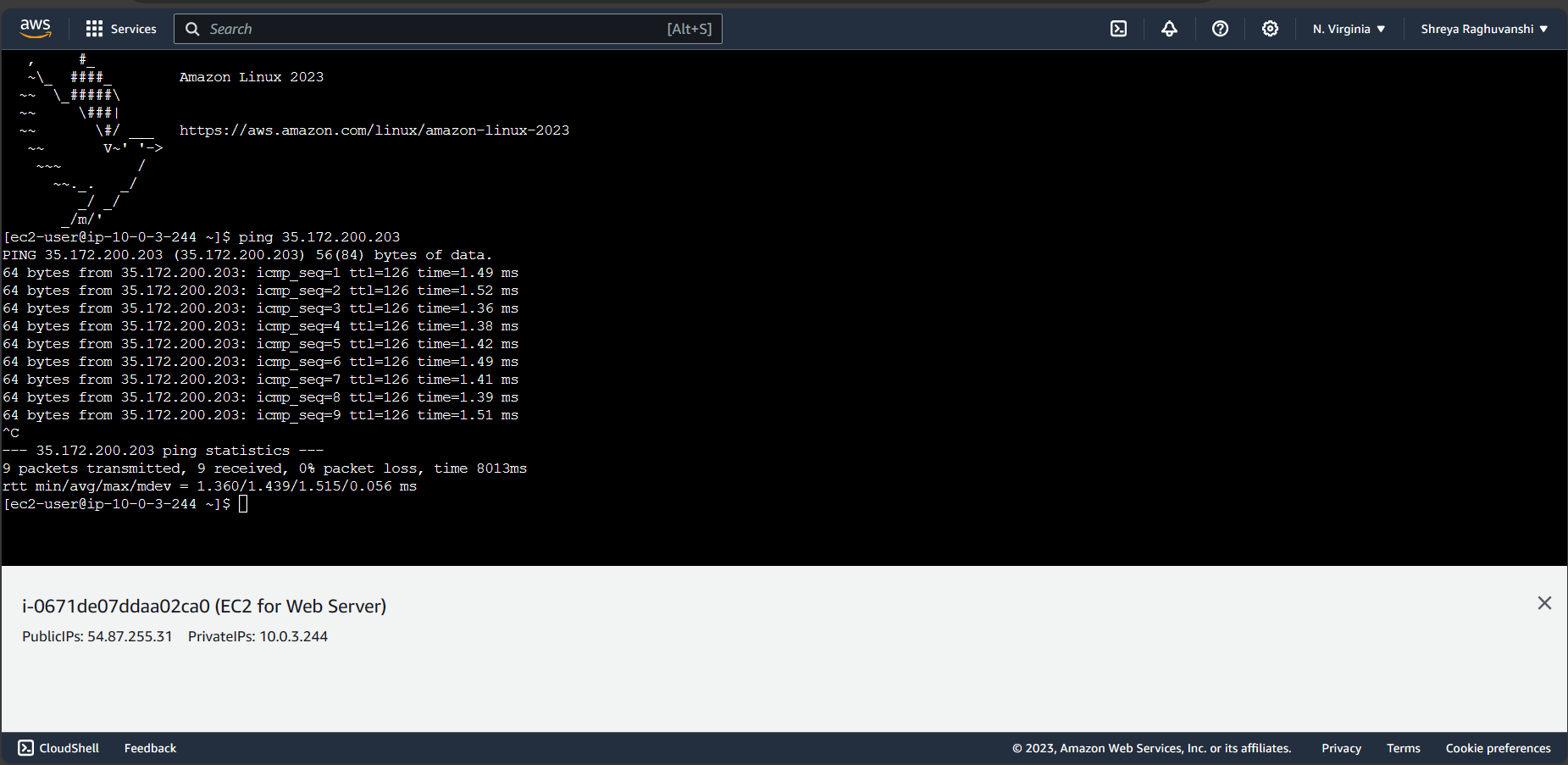


*fig 1.1 CloudFormation diagram*

1. Stack output:

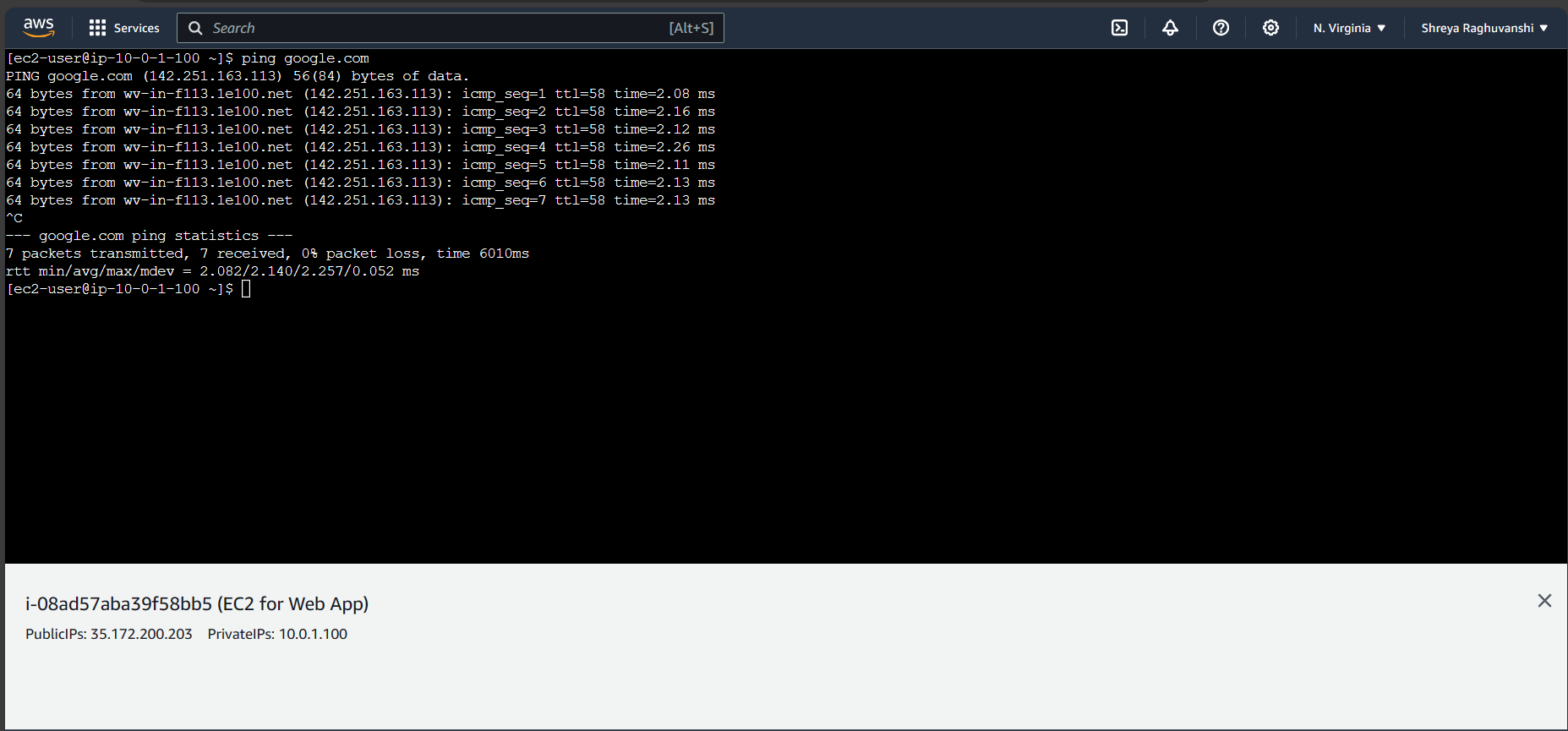
*fig 1.2 Stack creation in AWS*

3. Ping from Web Server to Web App:



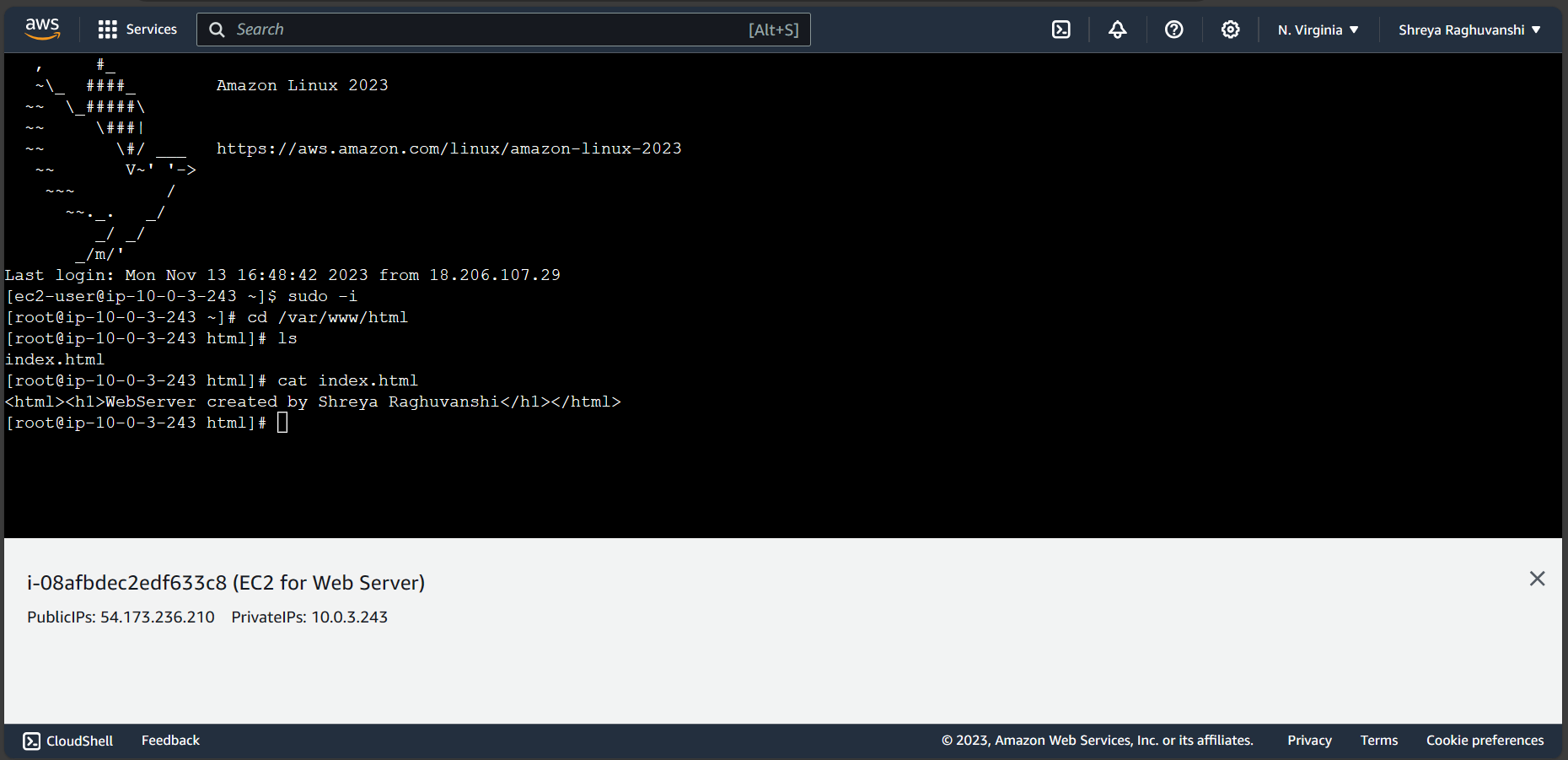
*fig 1.3 Ping Web App through Web Server*

4. Ping google.com from Web App:



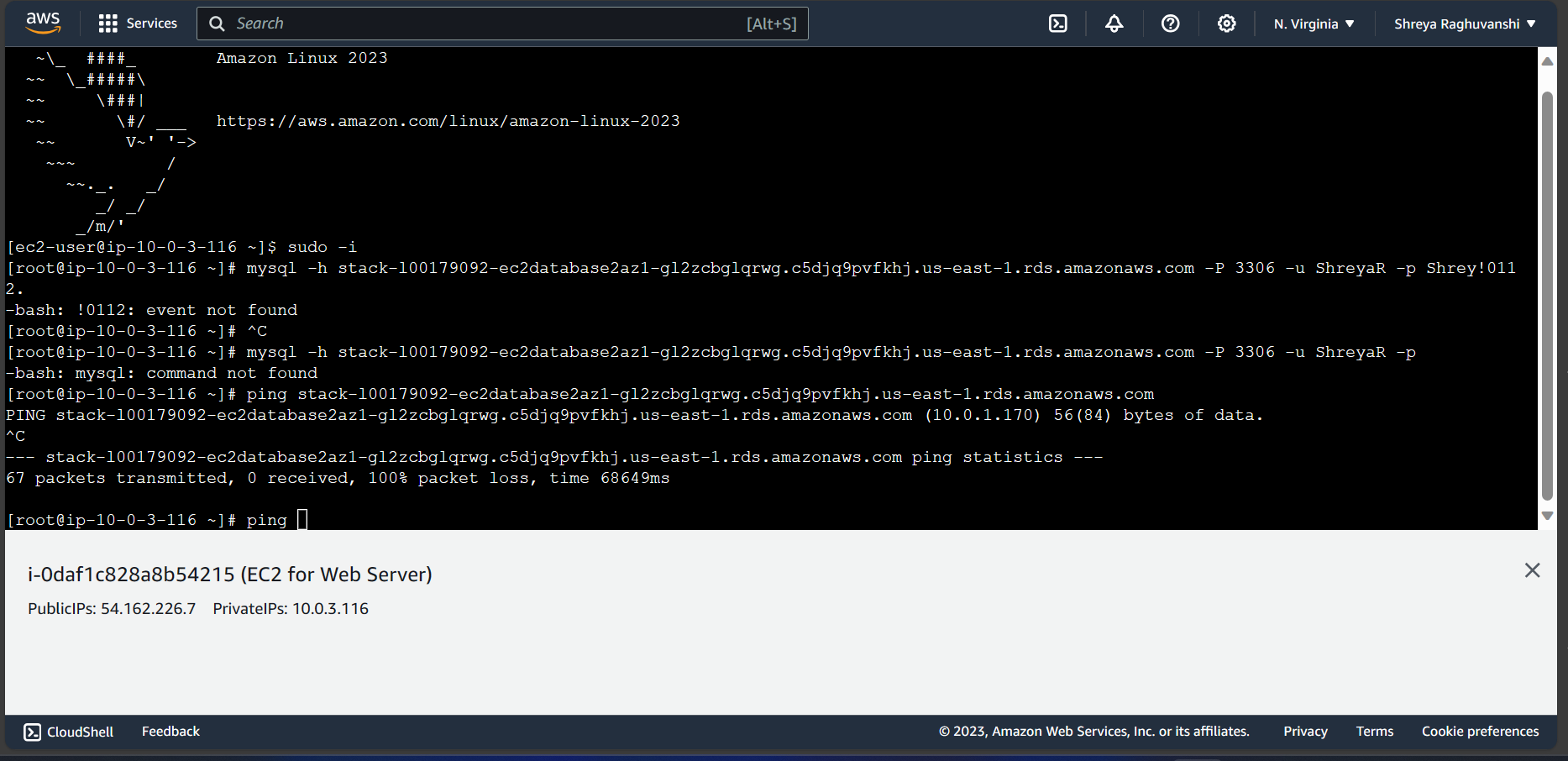
*fig 1.4 Ping google through Web App*

1. View contents of html file:



*fig 1.5 html file on Web Server instance*

1. Ping database instance through Web Server:



*fig 1.6 ping DBinstance through WebServer*

1. Link for GitHub repository where yaml file is present:

<https://github.com/ShreyaRaghuvanshi1/MiniNetwork_L00179092>

# Conclusion:

In conclusion, the AWS CloudFormation template created during the assessment provided a scalable and structured solution for creating an infrastructure. By utilizing Infrastructure as Code principles, the template ensured reproducibility and consistency. As suggested by Wood (2022), parameters and mappings are two key factors for enhancing reusability in the template. The template also emphasized on security through features like the definition of security groups that enables control over inbound and outbound traffic.

YAML template was used throughout the assessment for Mini-Network template creation. The YAML format offered a human-readable syntax, enhancing the ease of maintenance.

The main focus of this assignment was to create an architecture while also considering security. Both Security groups (SecurityGroupWebApp and SecurityGroupWebServer) allowed incoming traffic on port 22 (SSH) facilitating communication between WebApp and WebServer (refer fig 1.3). As both App and Server were present in the public subnet, they were able to ping to the internet (Eg. Google.com) (refer fig 1.4) or access any external resources. The VPC was configured with an Internet Gateway (InternetGW), and the public subnets were associated with a route table (PubRouteTable) that had a default route (0.0.0.0/0) which is connected to the IGW (Internet Gateway). This allowed traffic from the instances to reach the internet. However, EC2 instances could not ping the database as there was no security group created that allowed EC2 to communicate with Database (refer fig 1.6).

The creation of Mini-Network project commenced by creating Parameters. ‘Parameters’ were used as input values which were used to customize a CloudFormation template to make it more flexible and reusable. The first parameter was the VpcCIDR parameter that defined the Common Virtual Private Cloud (VPC) including all the resources within VPC. The users had the flexibility to redefine this CIDR block during stack creation.

Additionally, the PublicSubnetCIDR1 and PrivateSubnetCIDR1 parameters were responsible for the Public and Private Subnet CIDRs in Availability zone 1, while PublicSubnetCIDR2 and PrivateSubnetCIDR2 performed the same role for Availability zone 2. SSHLocation parameter was defined considering security for accessing web server and web app through SSH.

Furthermore, EC2 Key Pair (KeyName) was used for secure instance access. To set up Database, parameters like DBName, DBUserName, DBPassword, DBIstanceClass and DBAllocatedStorage were included.

The `Mappings` section of the AWS CloudFormation template contained a `RegionMap` that linked AWS regions with specific Amazon Machine Images (AMI). Each region had an associated AMI mapped. This mapping facilitated region-specific customization during stack creation.

The template also declared set of resources for the deployment of an infrastructure. The resources section consisted of a Virtual Private Cloud (VPC), public and private subnets across two Availability Zones, associated Route tables and an Internet Gateway. Security groups were configured to control traffic to different components, including web app, load balancer, web server, and database.

A number of authors including [Yusifova](https://dev.to/tiamatt) (2020) has emphasized on using Intrinsic functions in AWS. Intrinsic function ‘!Ref’ was used to refer to the value of a parameter or resource that allowed users to dynamically reference the properties of resources or parameters within CloudFormation templates. Another function which was used during resources specifications was Fn::GetAZs. It is an intrinsic function that returned an array of Availability Zone names for the specified AWS region. !Ref 'AWS::Region' referred to the AWS region where the CloudFormation stack is created.

RDS database instance was introduced in the template. The template also had the deployment of an Elastic Load Balancer across subnets, enhancing the infrastructure's resilience and distribution of traffic.

However, a different approach for this architecture would have been to use NAT gateway for private instances. As private instances cannot connect to the internet directly, it needs NAT gateway to access internet.

Through this experience, learners gained a deeper understanding of how to define, provision, and manage AWS resources using YAML/ JSON templates. The ability to orchestrate complex infrastructures, customize configurations with parameters and mappings, and ensure consistency across environments showcased the power of CloudFormation in AWS.

Overall, the assessment was a little tough for naive as it was a new concept and it was difficult to build a template from scratch within the timeframe. Concepts like parameters, mappings and resources were overwhelming for new learners. However, with practice and understanding, it becomes easier to create and manage the infrastructure.

The main issue with template designer was that it failed to provide proper error message. Another major issue was that of indentation, the AWS CloudFormation designer did not specify the line where the indentation error occurred which resulted in requiring users to search for indentation problems throughout the entire template.

During the assessment, multiple errors were faced. Some major issues are mentioned below for example:

Problem: One of the issues was the restrictions in ‘Allowed pattern’ property in Database password which did not allow ‘@’ to be used.

Solution: The issue was solved by avoiding the use of ‘@’ in allowed pattern for User Password.

Problem: Another error which was encountered during the stack creation stated ‘no default subnet detected in VPC’.

Solution: The above error was solved with the assistance of [Bhalodia](https://stackoverflow.com/users/22450739/kavya-bhalodia) (2023) by creating a default subnet in the availability zones through local CLI. ‘aws configure’ command was used to configure CLI with AWS account. Access key ID and Secret Access Key was entered followed by ‘aws ec2 create-default-subnet --availability-zone us-east-1a’ for AZ1 and similarly ‘aws ec2 create-default-subnet --availability-zone us-east-1b’ for AZ2.

To sum up, the project underscored the significance of CloudFormation in simplifying and automating the deployment and management of cloud infrastructure, making it a valuable tool for scalable and efficient operations in the AWS environment and the broader integration of AWS with DevOps.

# References:

Bhalodia, K. (2023) Stack Overflow*. Recreate AWS default subnets*. Available at: <https://stackoverflow.com/questions/71282141/recreate-aws-default-subnets> (Accessed: 17 November, 2023).

Wood, J. (2022). Telstra Purple. *Reusable CloudFormation Templates*. Available at: <https://purple.telstra.com/blog/reuasable-cloudformation-templates> (Accessed: 18 November, 2023).

‌ Yusifova, S. (2020). DEV Community. (2020). *Hands-on AWS CloudFormation - Part 2. Into to Intrinsic functions*. Available at: <https://dev.to/tiamatt/hands-on-aws-cloudformation-part-2-into-to-intrinsic-functions-4kj2> (Accessed: 19 November, 2023).