Problem Statement - Create an Autoscaling group using Terraform on AWS. The instances in the group should have Nginx installed on them. Parameters in the ASG can be set at your discretion and should be mentioned in the solution documentation.

Any variables in the script should be placed in a separate variables.tf file. The output of the script should be the DNS of the associated load balancer.

Grading Criteria

1. Documentation explaining the solution architecture and template created - 10 marks
2. Shell script used to install Nginx 5 marks
3. Separate variables file 5 marks
4. Output file showing DNS of load balancer 10 marks
5. Terraform script with given components
   1. Security groups 2 marks
   2. EC2 instances 3 marks
   3. Elastic Load balancer 5 marks
   4. Autoscaling group 5 marks
6. Screenshot showing successful execution of Terraform script 5 marks

Solution

1. The architecture contains a VPC with two public subnets in different regions (us-east-1a and us-east-1b), one private subnet (us-east-1c), one internet gateway, a custom route table with destination as “anywhere” and target as internet gateway associated with two public subnets. It also contains a NAT gateway in public subnet 1 associated with private subnet through a custom route table for private subnet separately.

The instances in the public subnets are maintained by autoscaling group. It launches/terminates the instances depending on the auto scaling group setting and autoscaling group policies. The instances have a predefined launch configuration as described in aws\_launch\_configuration.

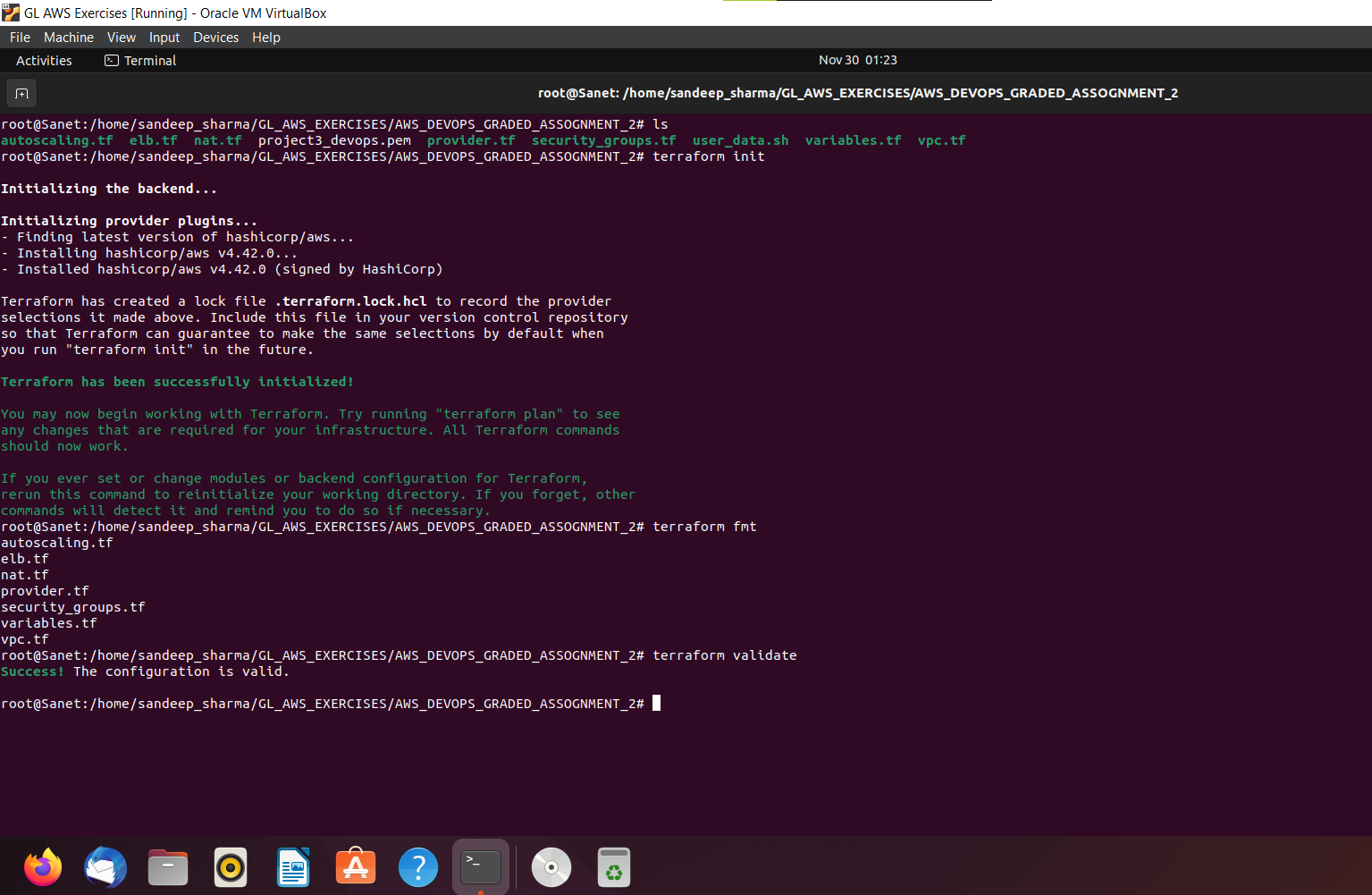
There is an elastic load balancer which distributes the traffic between the instances launched by autoscaling group.

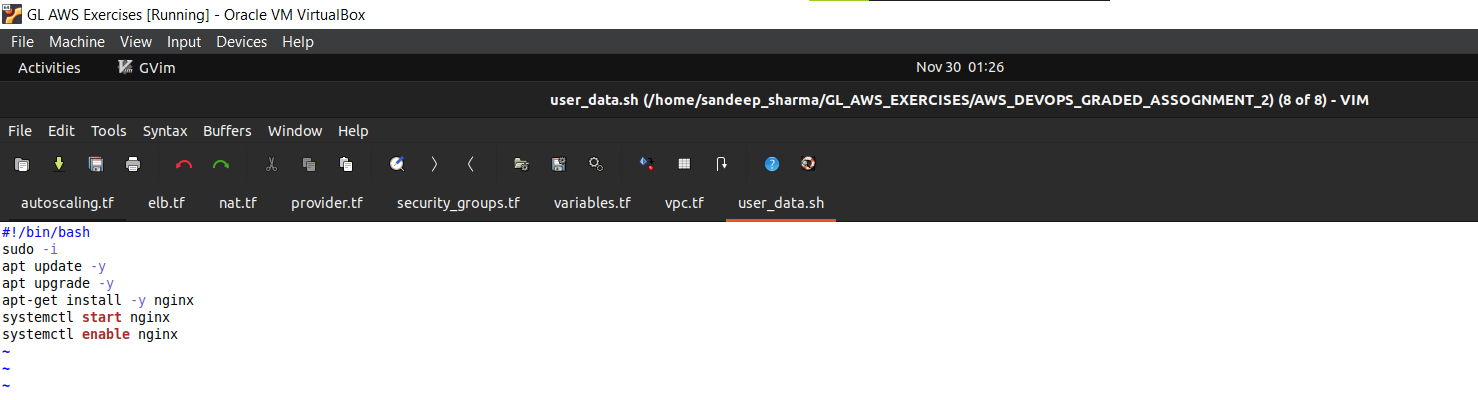
There are two security groups defined, for instances in public subnets (inbound : port 80 – http, port 22 – ssh || outbound : anywhere) and for elastic load balancer (inbound : port 80 – http || outbound : anywhere). Both security groups have been associated with VPC.

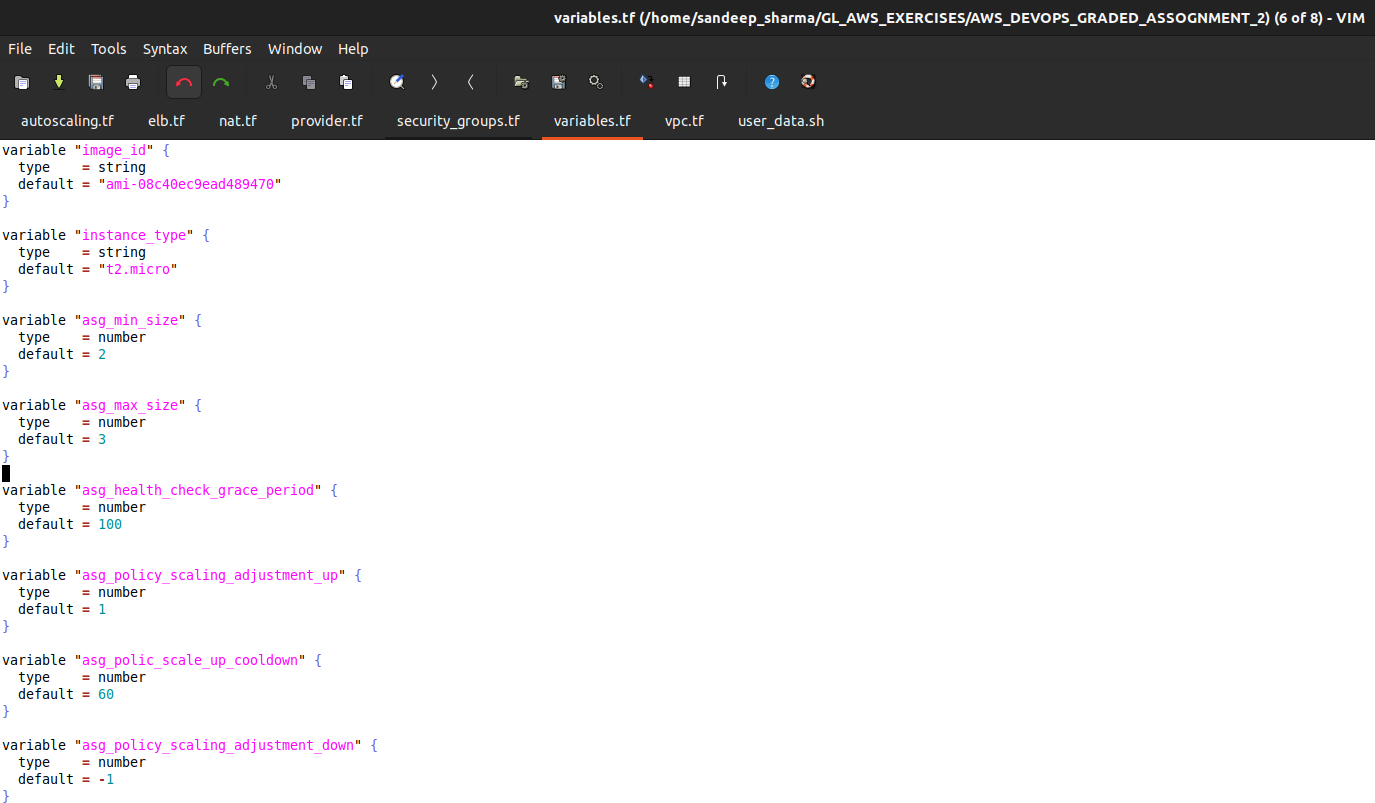
FILES:

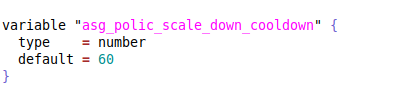
* 1. autoscaling.tf – contains “aws\_launch\_configuration”, “aws\_autoscaling\_group”, “aws\_autoscaling\_policy” for scaleup and scaledown.
  2. elb.tf – has “aws\_elb” (Elastic Load Balancer) and “output” defined.
  3. nat.tf – cotains “aws\_nat\_gateway”, “aws\_route\_table” for private subnet and nat gateway connection, “aws\_route\_table\_association” to associate route table to private subnet and “aws\_eip” to give an IP to nat gateway after creation.
  4. provider.tf – defines which cloud provider plugin to use while terraform executes.
  5. Security\_groups.tf – has security groups “aws\_security\_group” defined for instances and elb.
  6. user\_data.sh – a shell script to update the libraries and setup nginx after instances have been created and launched.
  7. vpc.tf – contains “aws\_vpc”, “aws\_subnet” for 2 public and 1 private, “aws\_internet\_gateway”, “aws\_route\_table” for 2 public subnets, and “aws\_route\_table\_association” for both public subnets to get associated with the route\_table.
  8. variables.tf – defines variables used in other resources and assigns their values.

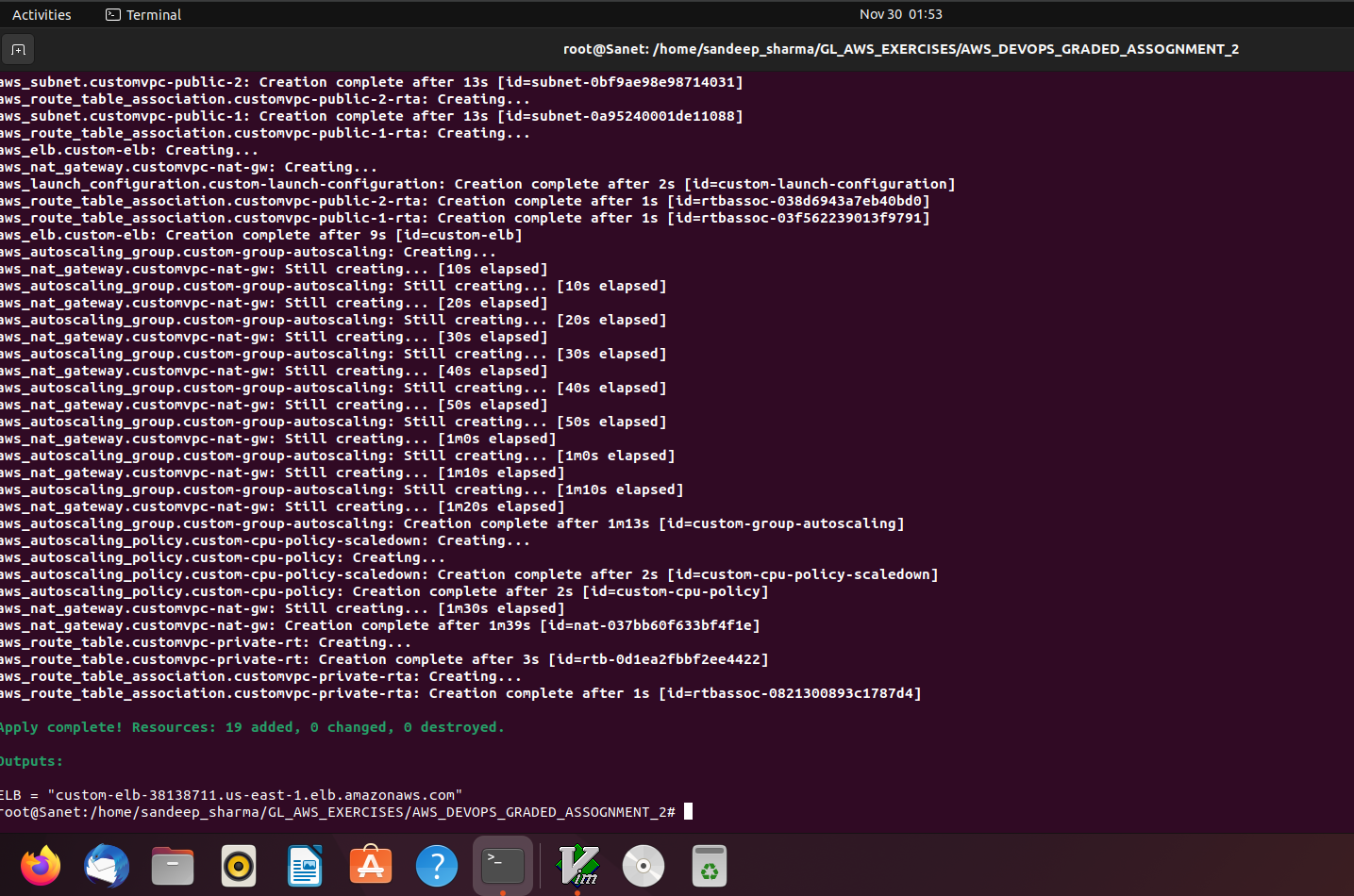
Initial setup:

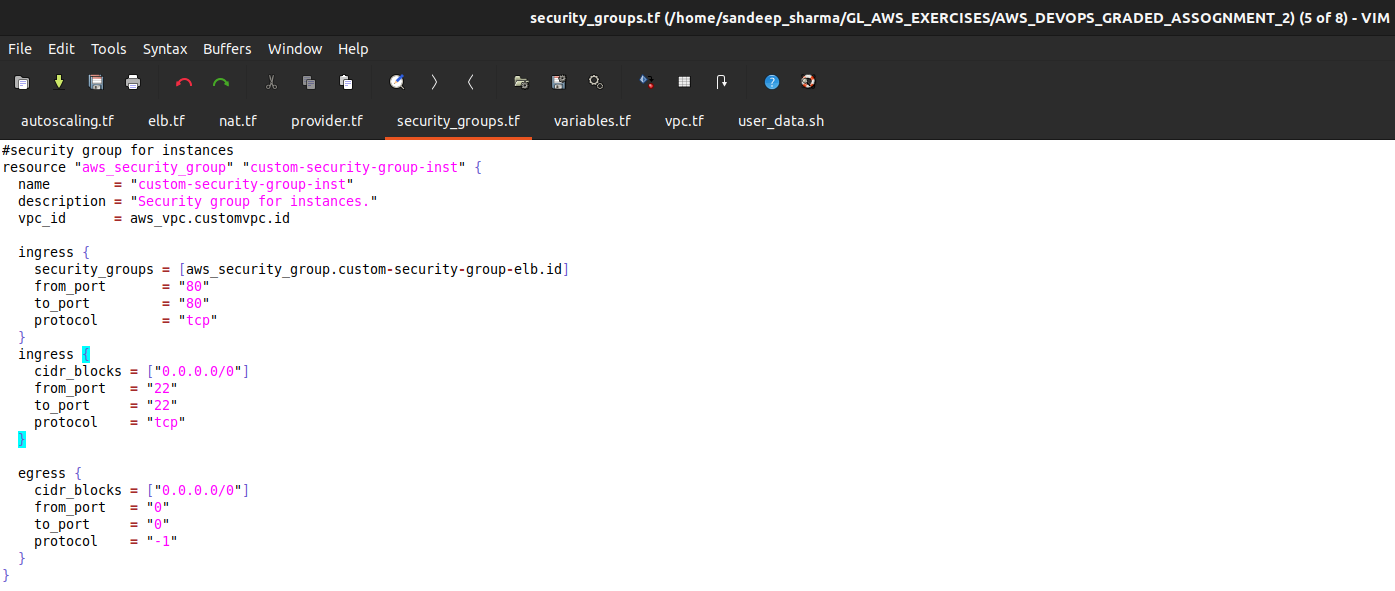


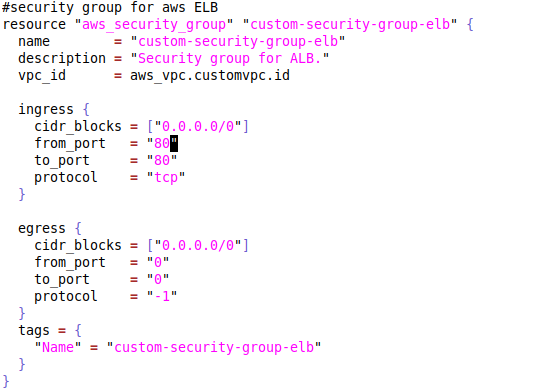




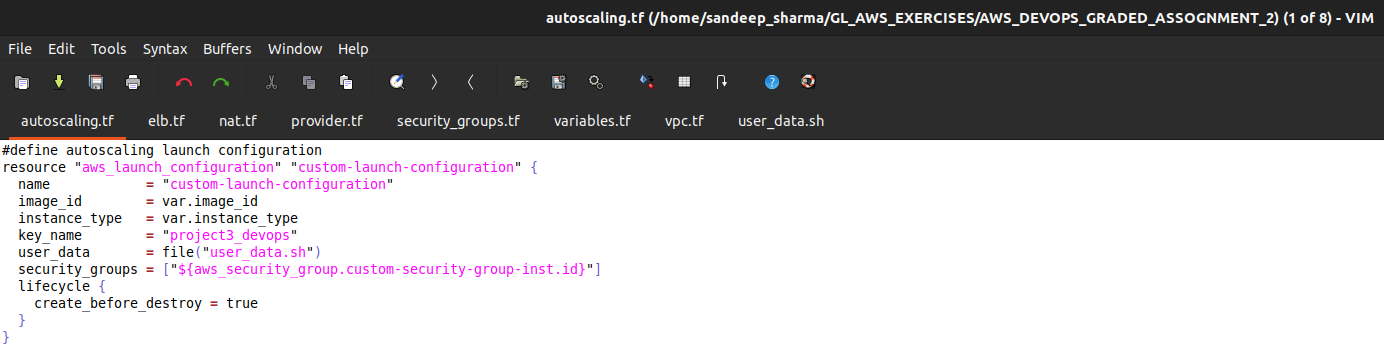


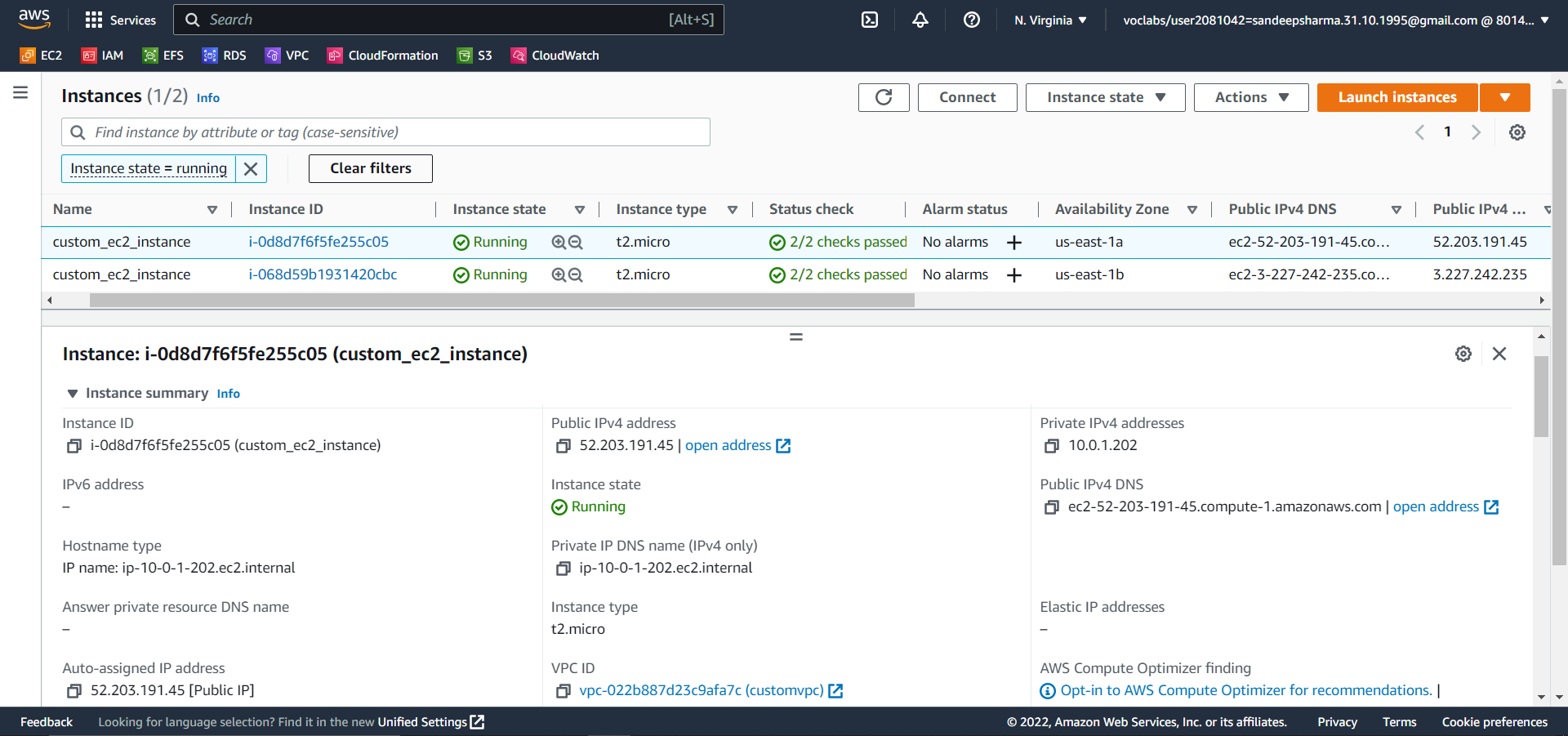


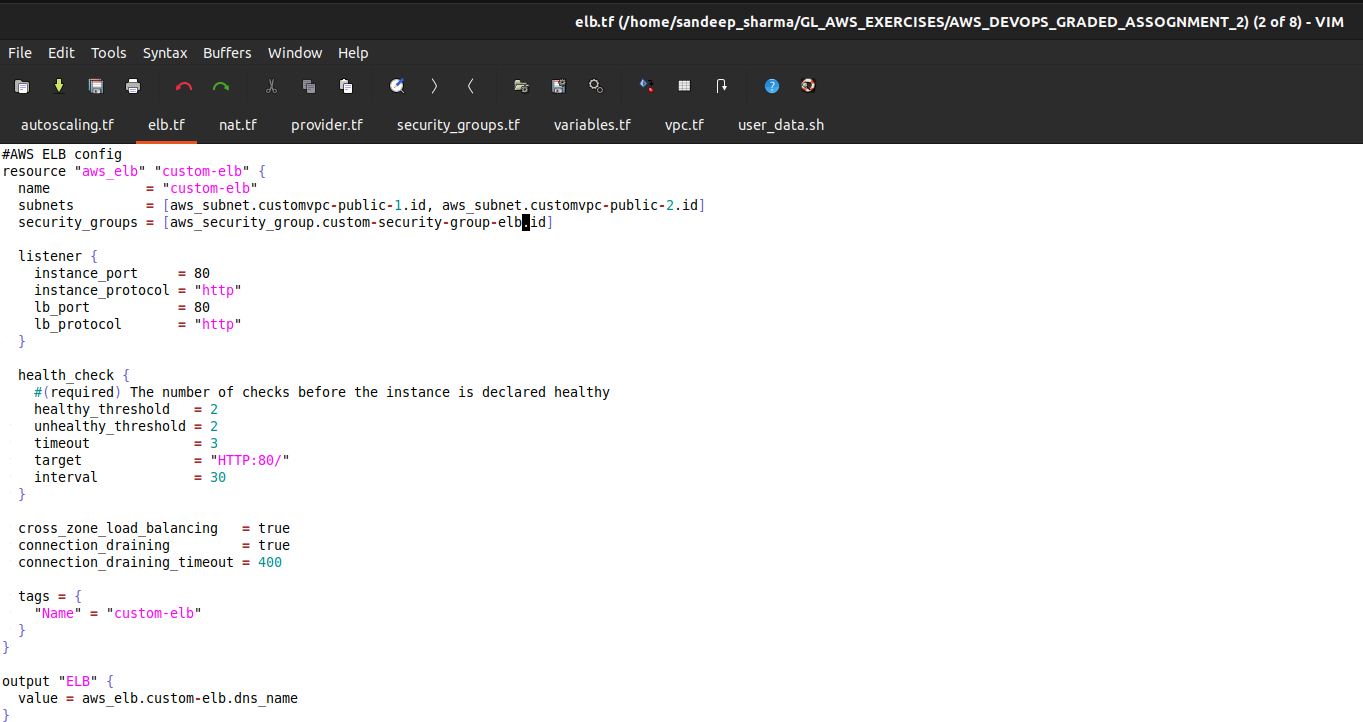
1. Terraform scripts:
   1. Security groups: 

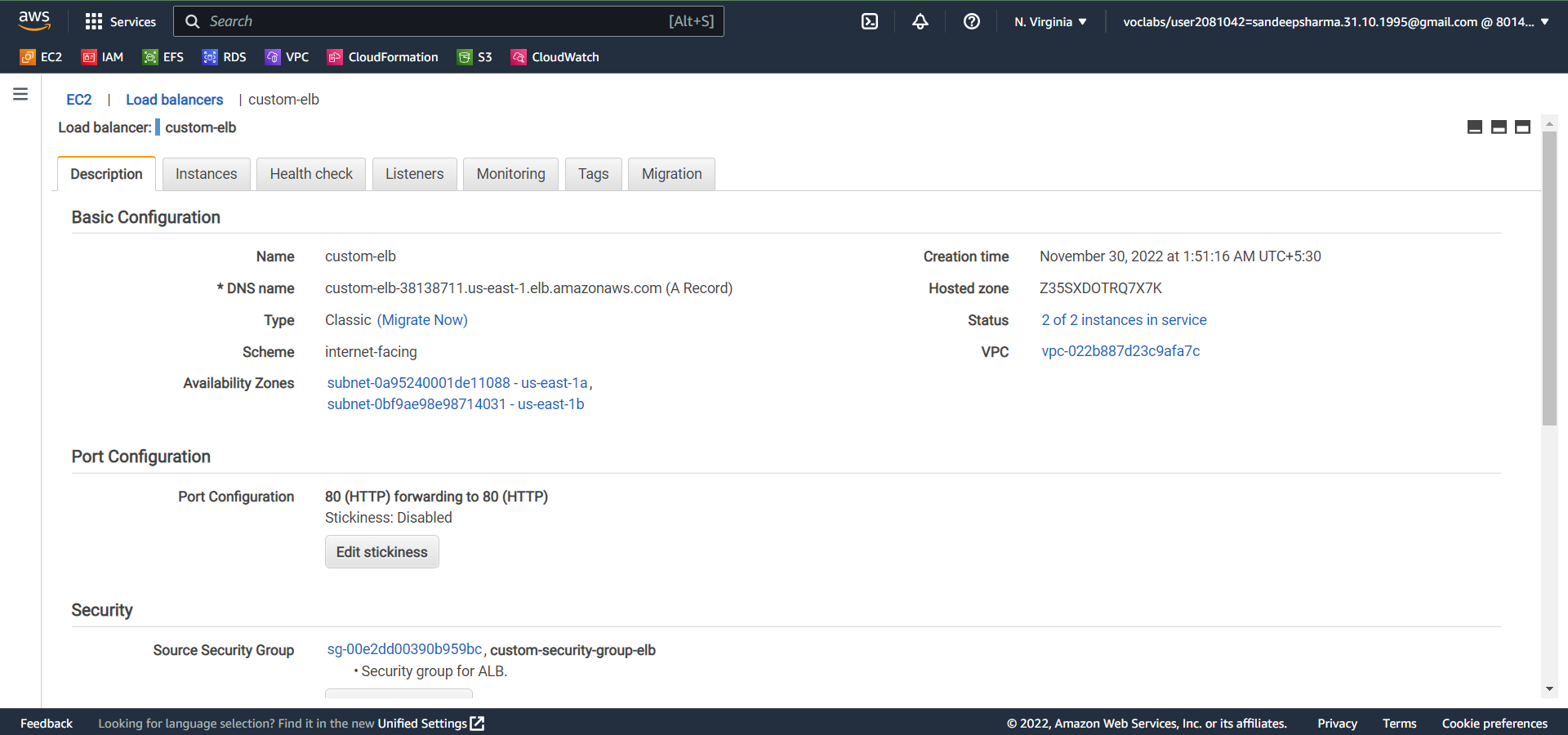


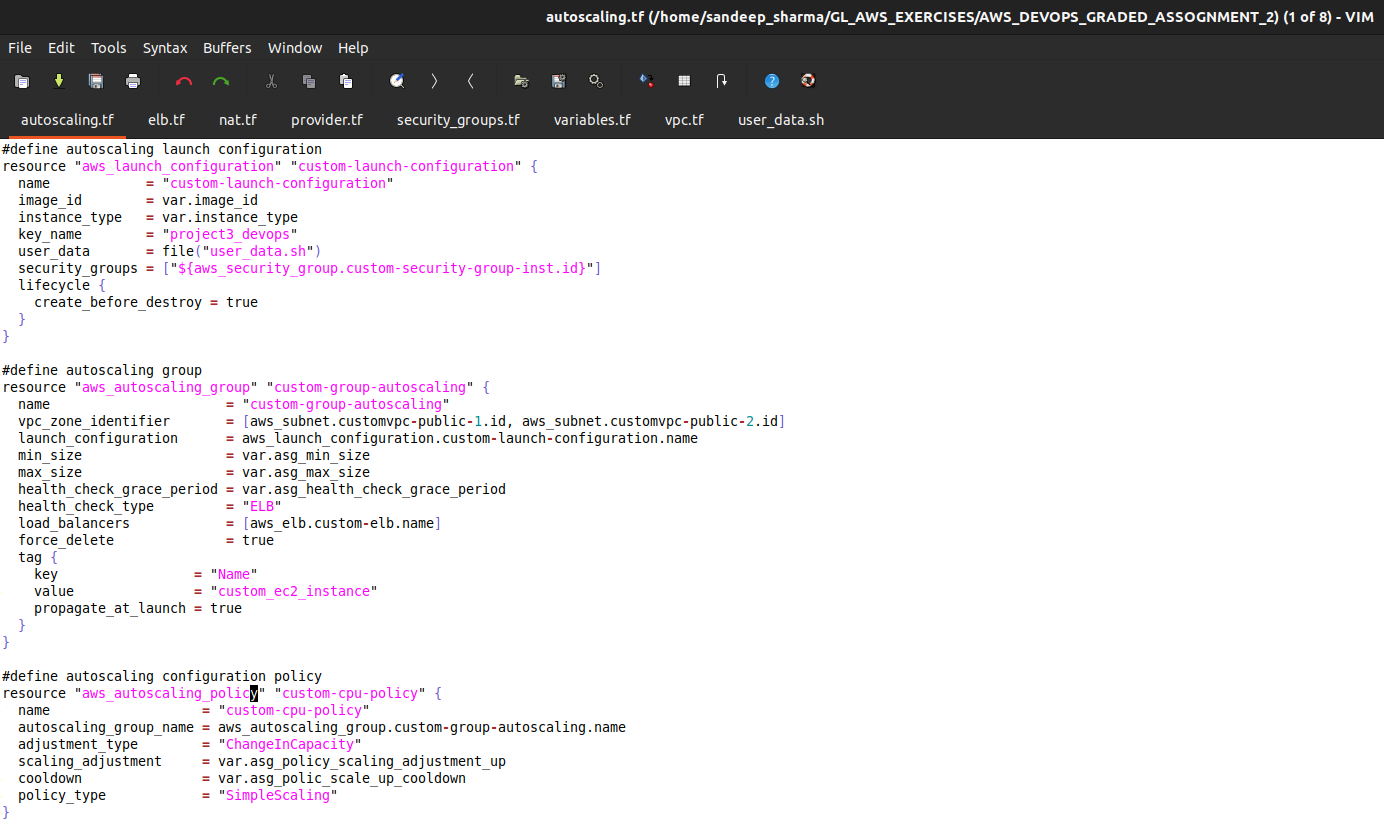
* 1. EC2 instances (instances are launched by auto scaling group itself.. I have not created aws\_instancce separately) – created using launch configuration

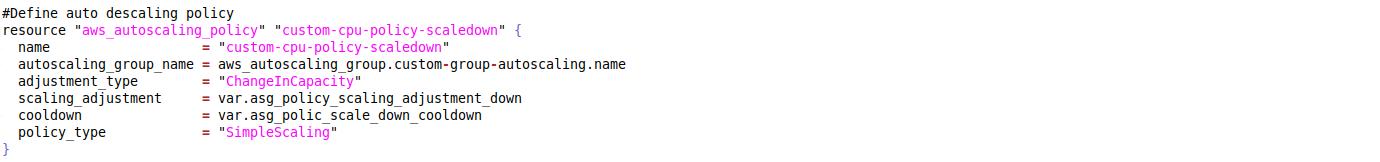


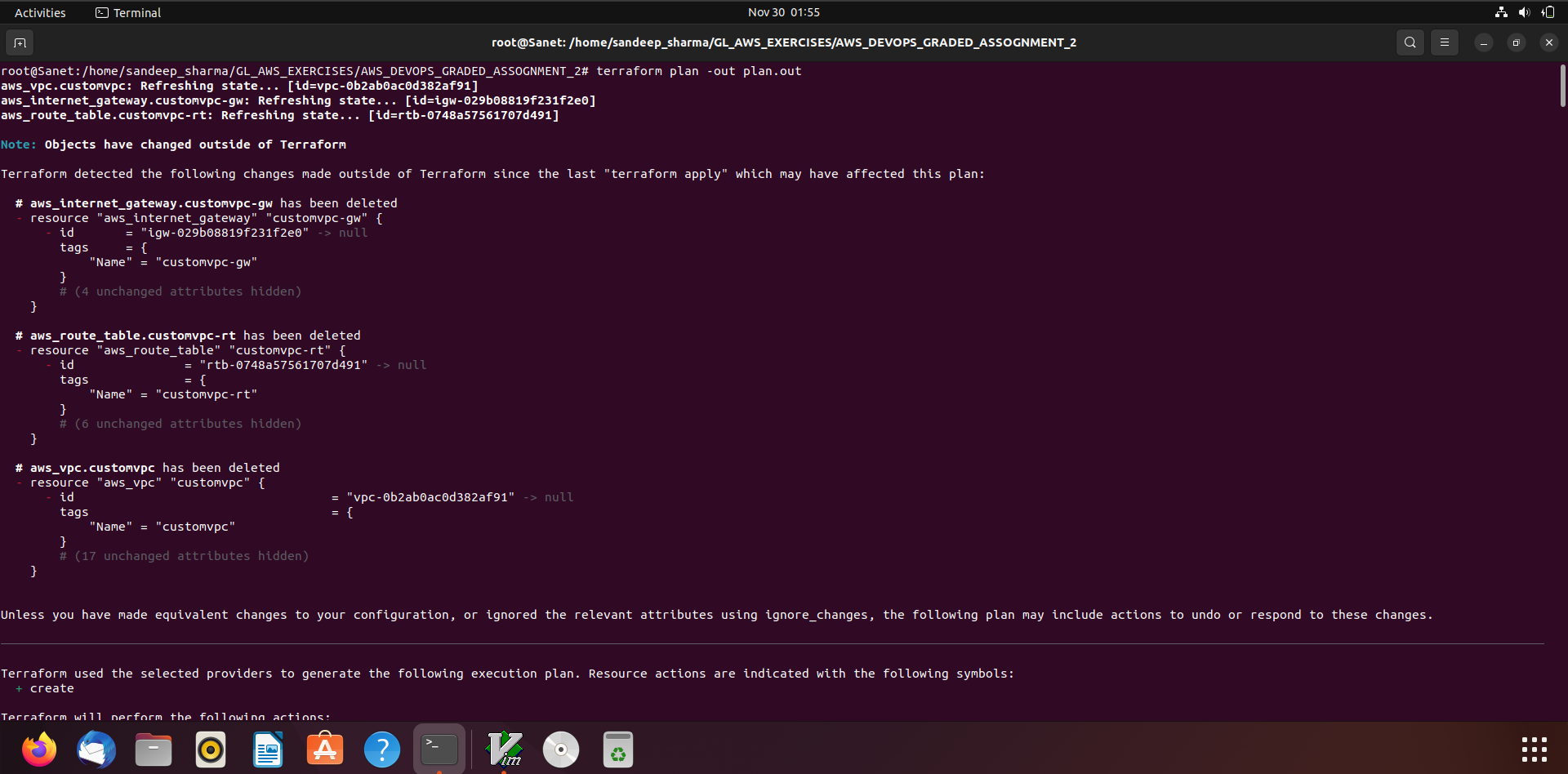


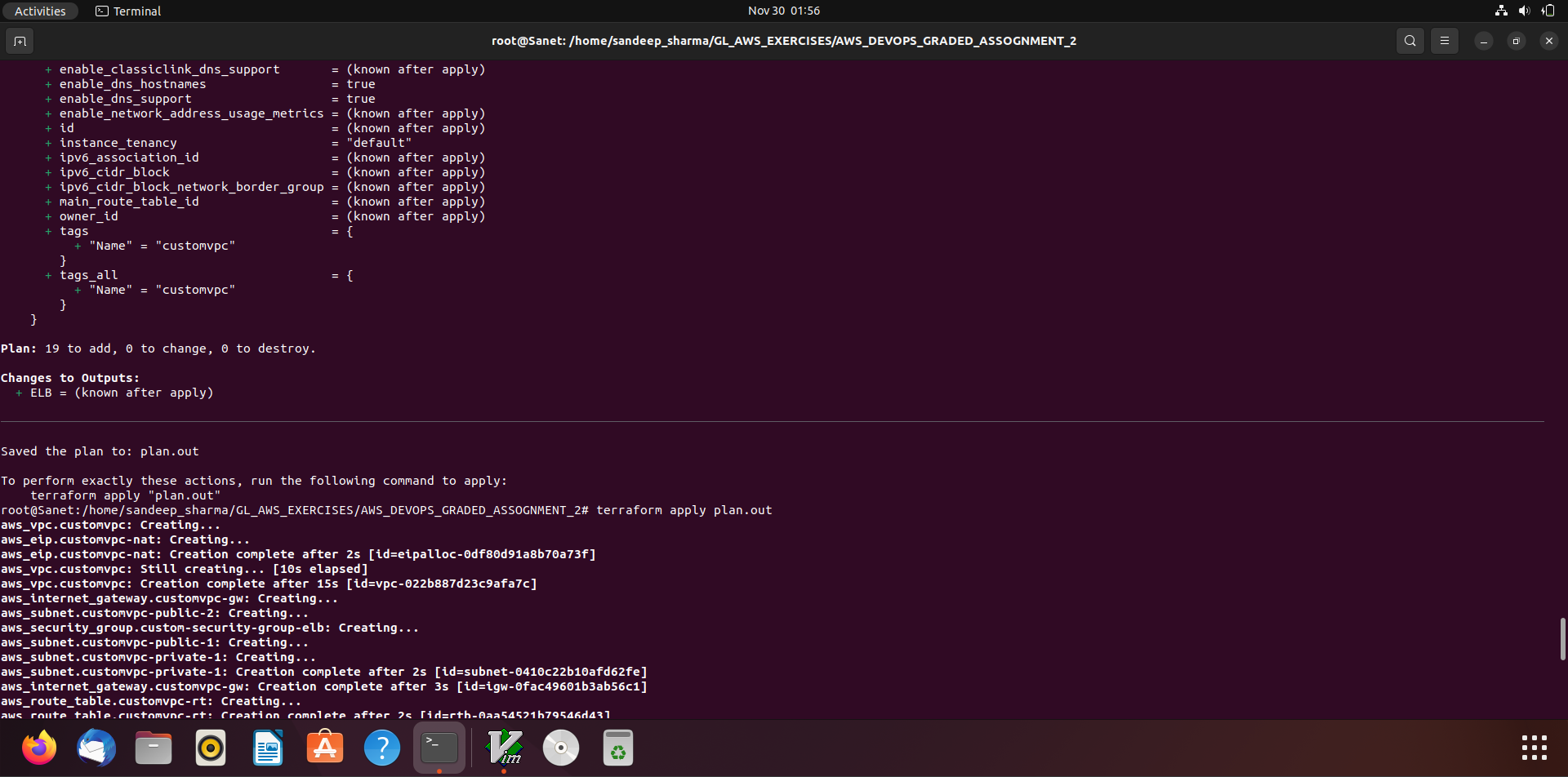


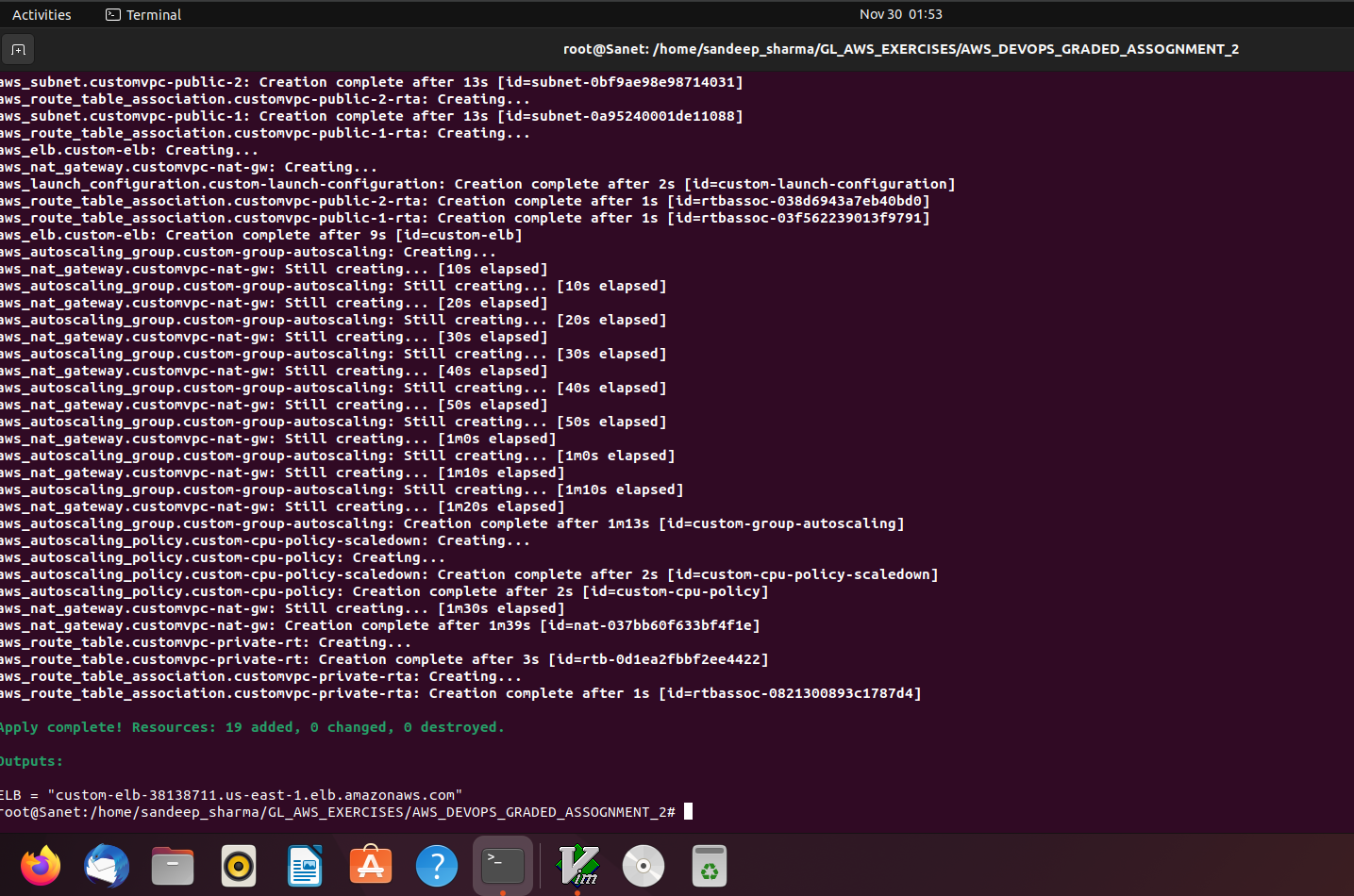


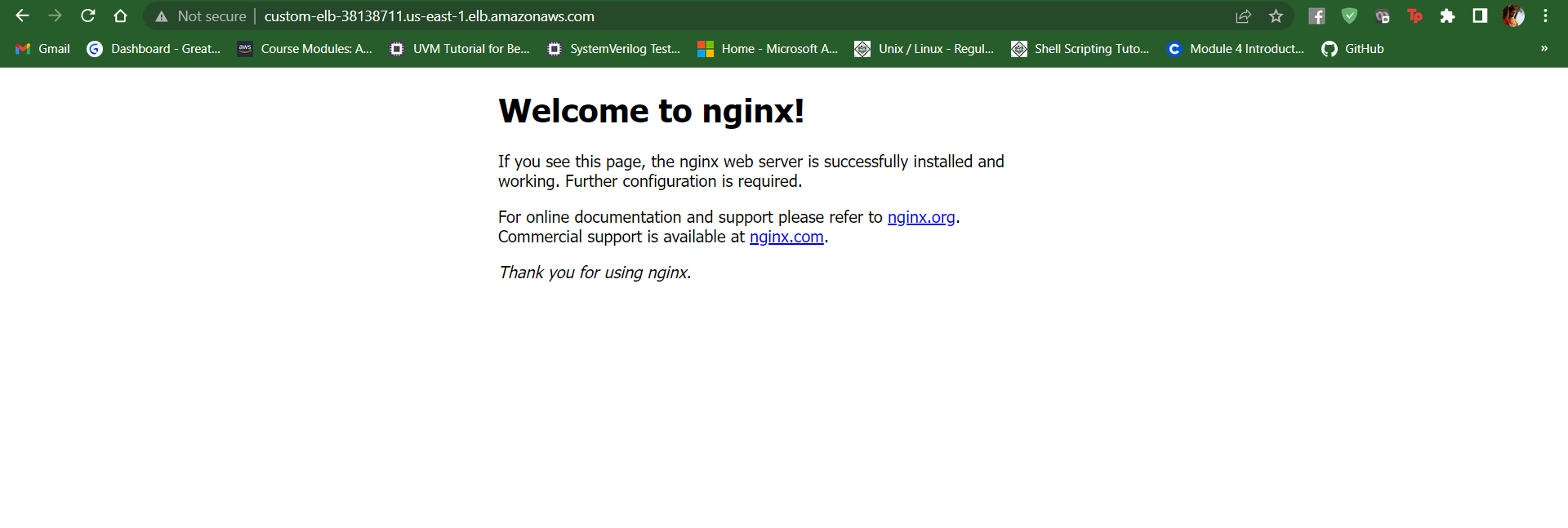




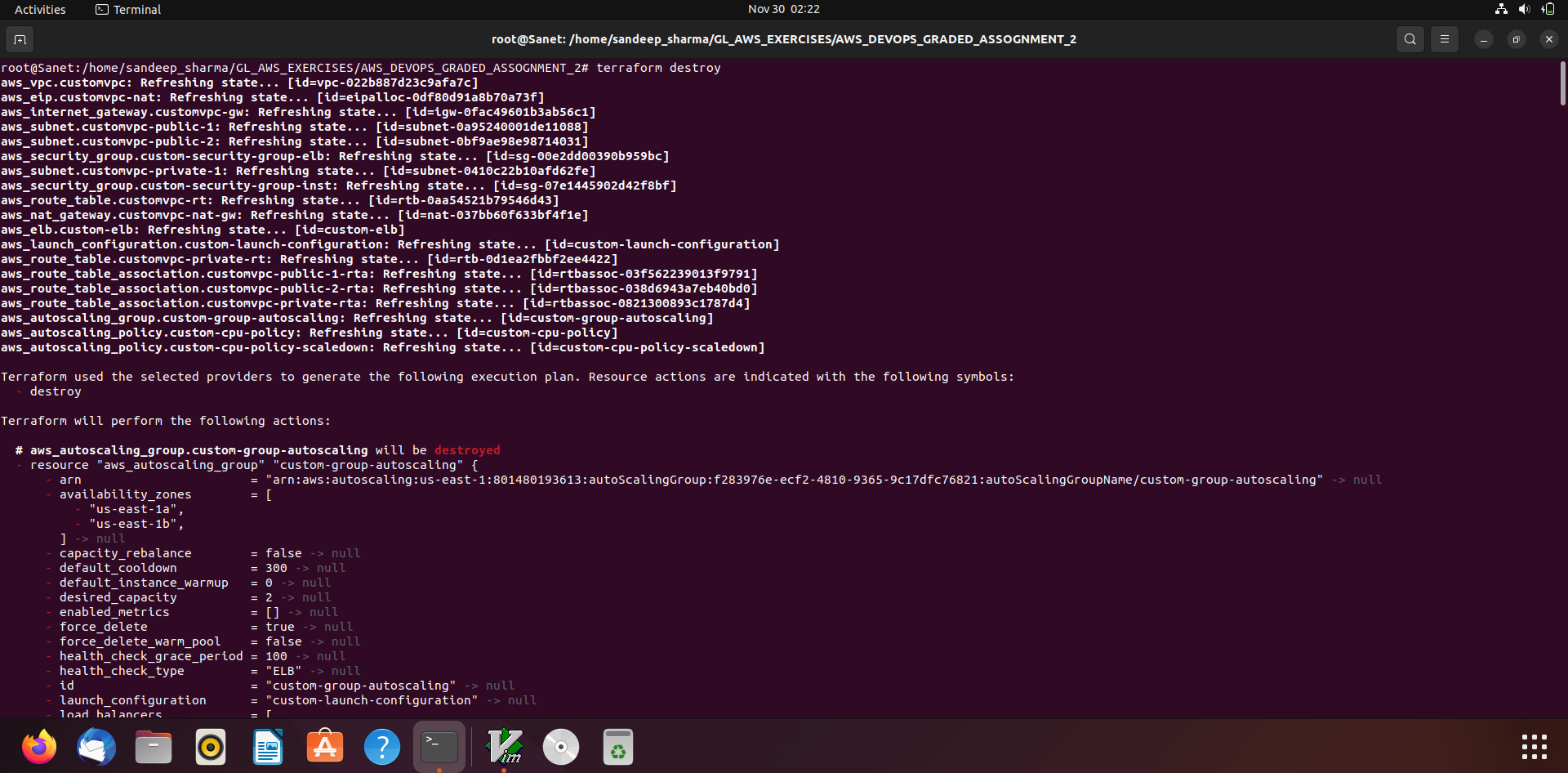


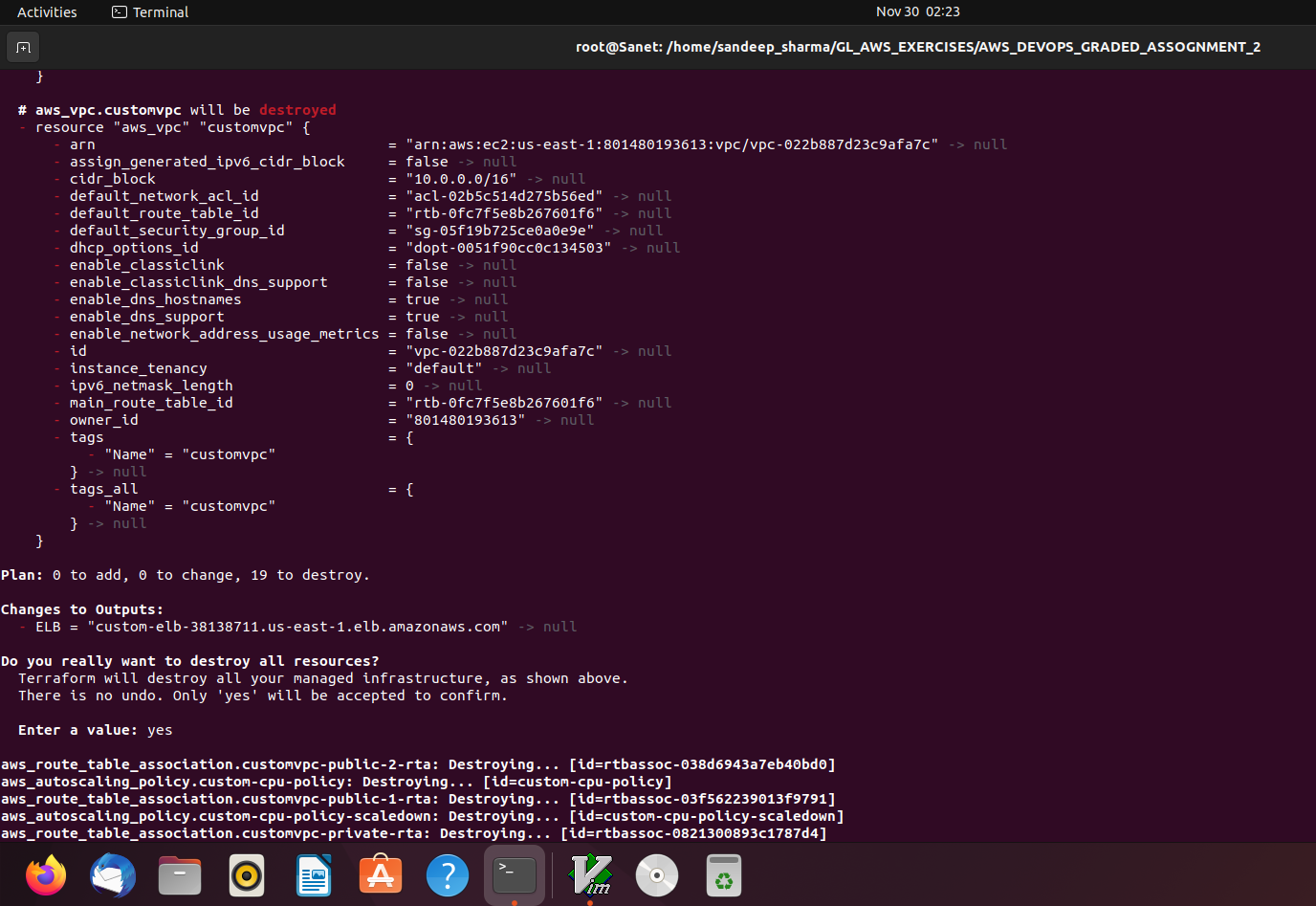


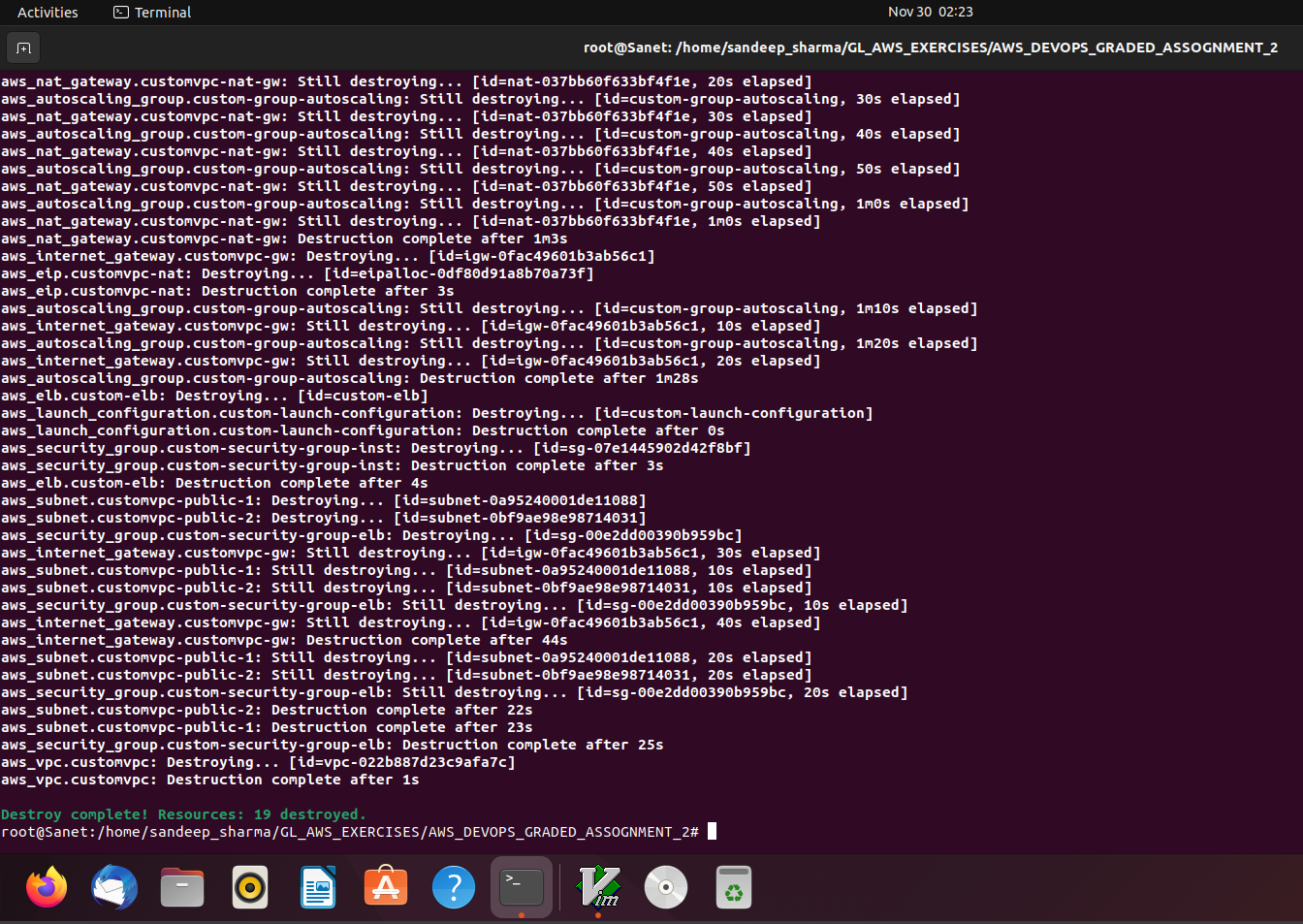




Destroying all procured resources:







Codes uploaded to github also:

***https://github.com/Sandeep-Sharma-3/Sandeep\_Sharma\_Cloud\_Computing\_Graded\_Assignment\_2.git***