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

* **Answers:**
* Documentation for the system architecture:

Our infrastructure is designed to operate efficiently and smoothly on AWS. The project has a modular structure including the following separate files:

**1) main.tf** – to set up AWS provider and define all the resources of our system

**2) variables.tf** – to define variables or customizable settings separately

**3) output.tf** – to define outputs from the terraform configuration

**4) install\_nginx.sh** – to define the shell script for nginx installation

* Resources defined in **main.tf** are as follows:

1. ***Auto Scaling Group (ASG****):*

The ASG allows us to automatically scale our application up or down thus helps to utilize the power of cloud computing in form of its elasticity. The ASG automatically adjusts the number of EC2 instances based on demand ensuring optimal performance and cost-efficiency.

In our Terraform code, we defined various properties for the ASG like initial desired capacity, minimum and maximum capacity and the subnet where we want to create instances elastically.

In our code, these properties are defined as variables. Specific values for them are provided in the separate variables.tf file. which the ASG refers to.

1. ***Launch Configuration (LC****):*

This is a template or blueprint for EC2 instance creation as part of the Autoscaling service. It includes various details like name, AMI, instance type, keypair, Security Group, software configuration and startup behavior etc. We have included a Launch Configuration as part of the ASG consisting of these properties.

In our code, most of these properties are defined as variables with specific values provided in the dedicated variables.tf file that the Launch Configuration resource refers to.  
  
The configuration also uses a Shell Script for Nginx installation so that every EC2 instance of the Autoscaling Group will automatically have Nginx installed on them.

* ***Shell Script*:**

In our Project, we have separately created a Shell Script for Nginx installation in the **install\_nginx.sh** file. It is used as the reference file as part of the EC2 Launch Configuration resource.

***3)* *Elastic Load Balancer (ELB****):*

ELB distributes network or application traffic across multiple servers (EC2 instances) to ensure no single server becomes overwhelmed thus enhancing the availability, responsiveness and reliability of the application. As we are using Auto Scaling Group (ASG) in our system so we needed an ELB also.

The ELB resource block defines listener and health check. Listener handles incoming traffic and routes it to the backend instances. Health Check monitors the health of the instances.

* Listener specifies that both the ELB and the instances are listening on port 80 through HTTP protocol.
* Health Check specifies port and protocol to perform health check, interval, timeout, healthy threshold and unhealthy threshold.
* Port and Protocol are set to “HTTP:80/”
* Interval (time interval between each health check) is set to 30 seconds
* Timeout (amount of time to wait for a response before considering health check fails) is set to 5 seconds
* Healthy threshold (number of consecutive successful health checks required before an instance is considered healthy) is set to 2
* Unhealthy threshold (number of consecutive failed health checks required before an instance is considered unhealthy) is set to 2

ELB resource block also defines Security Group, Availability Zones (AZ) etc. as variables.

1. ***Security groups:***

security groups are robust security setups. They act as virtual firewalls to control inbound and outbound traffic to and from our resources thus safeguarding our system against unauthorized access. We have included Security Groups for both EC2 instances and Elastic Load Balancer (ELB) for extra security.

* ELB Security Group allows inbound traffic to the ELB on port 80 (for HTTP access) from anywhere. ELB distributes traffic to multiple instances.
* Instance Security Group allows inbound traffic to individual instances on port 22 (for SSH access) and 80 (for HTTP access) from anywhere.
* Outbound rule for both ELB Security Group and Instance Security Group set from\_port and to\_port to 0 (allowing outbound traffic to use any port), protocol to “-1” (allowing outbound traffic through any protocol) and cidr block as “0.0.0.0/0” (allowing outbound traffic to any IP address on the internet)
* Variables defined/specified in **variables.tf** are as follows:
* AWS region (default set as “us-east-1”)
* Availability Zones (default set as “us-east-1a”)
* Subnet ids (default set as the ‘default subnet’ in “us-east-1a” AZ)
* AMIs (default set as Amazon Linux 2)
* EC2 instance type (default set as t2.micro)
* Keypair for SSH access (needed during Terraform apply and run)
* ASG desired capacity (default set as 2)
* ASG maximum capacity (default set as 3)
* ASG minimum capacity (default set as 1)
* AWS access key/accessId (needed during Terraform apply and run)
* AWS secret access key (needed during Terraform apply and run)
* output.tf file content:

The DNS (Domain Name System) of a load balancer plays a crucial role in directing client requests to the load balancer (which then distributes these requests to the backend servers or instances). The DNS name of a load balancer acts as the entry point for all incoming traffic to our application. Instead of exposing the IP addresses of individual servers, we provide the DNS name of the load balancer.

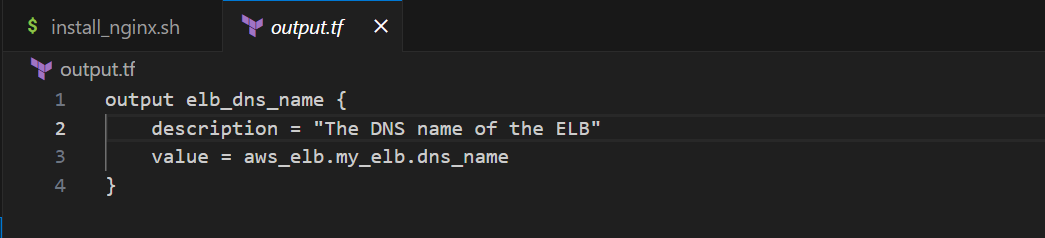
Output.tf file consists of a code that shows the DNS name of our Elastic Load Balancer as the output in the Terraform CLI when we do ‘terraform apply’ in our project directory and the code is properly executed thus successfully creating all the resources described above.

* Screenshots:

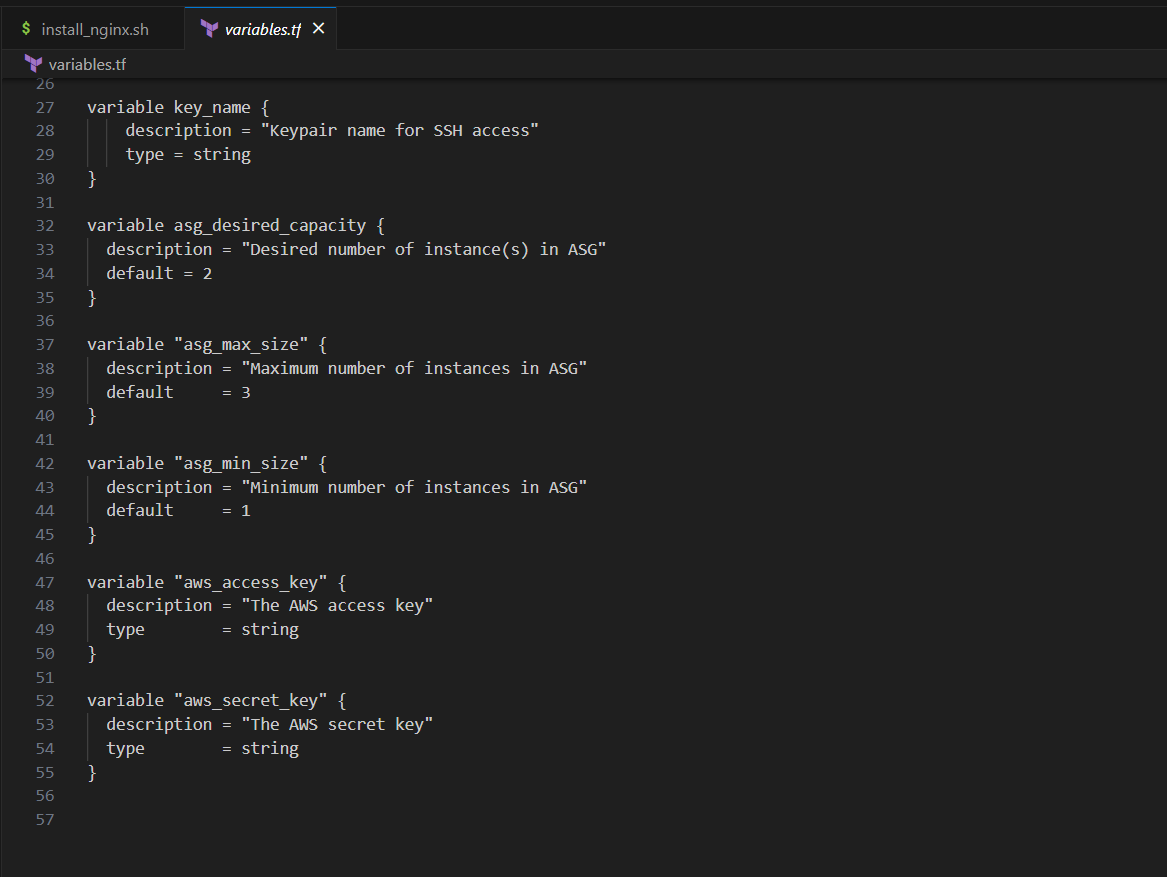
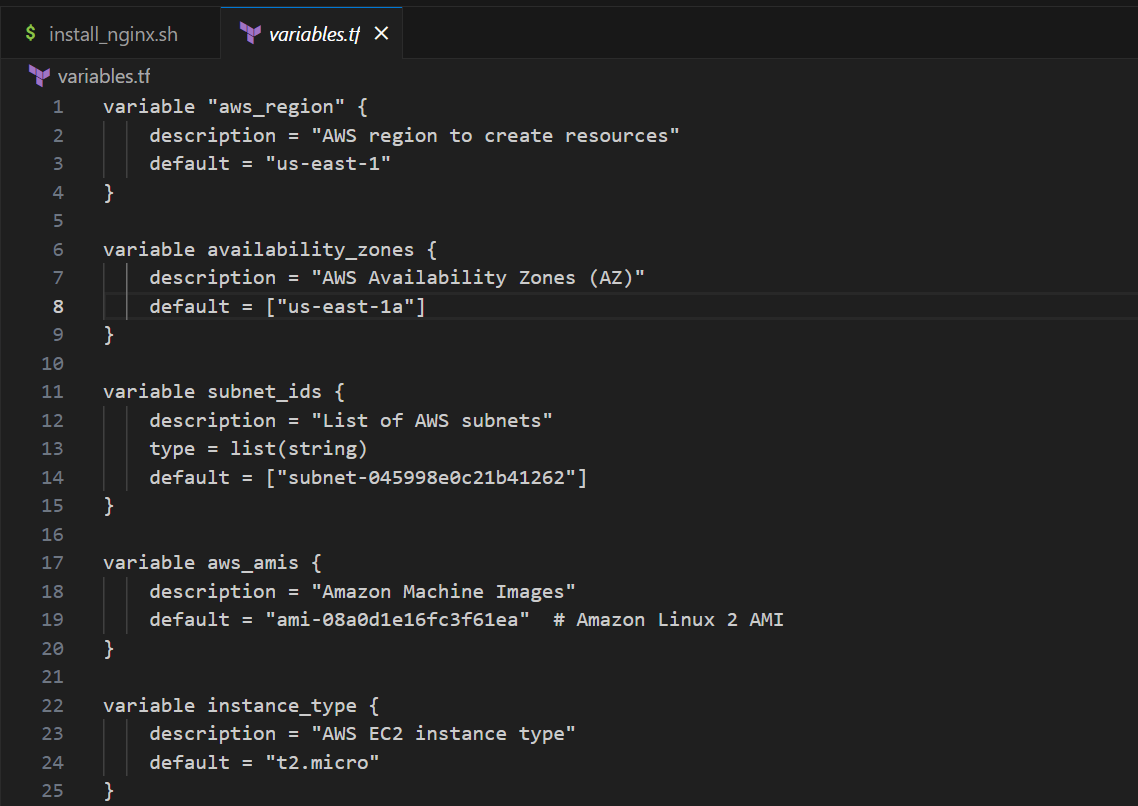
Shell Script for nginx installation –



Output file (DNS name) –

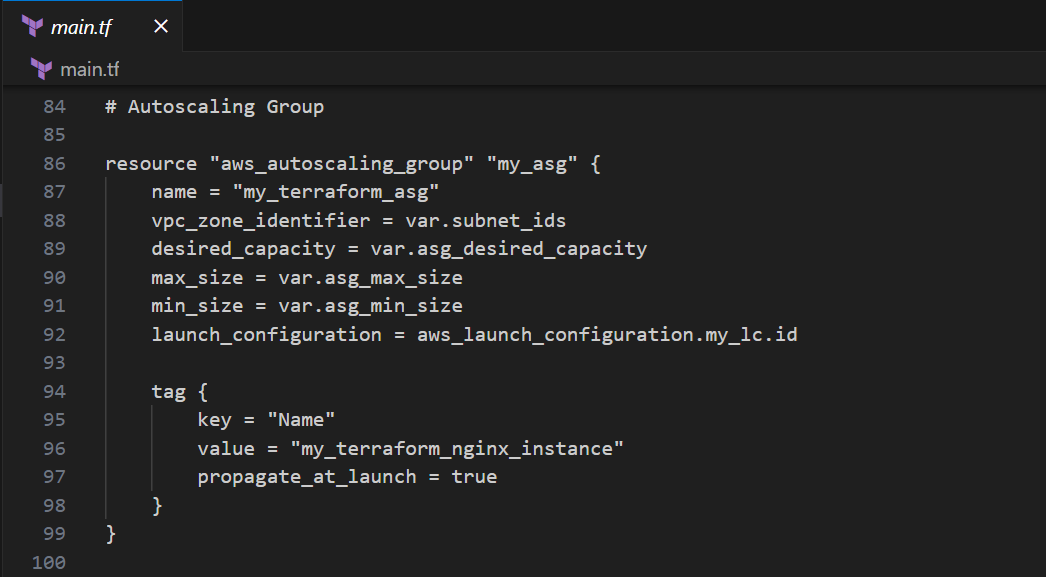


Variables file –

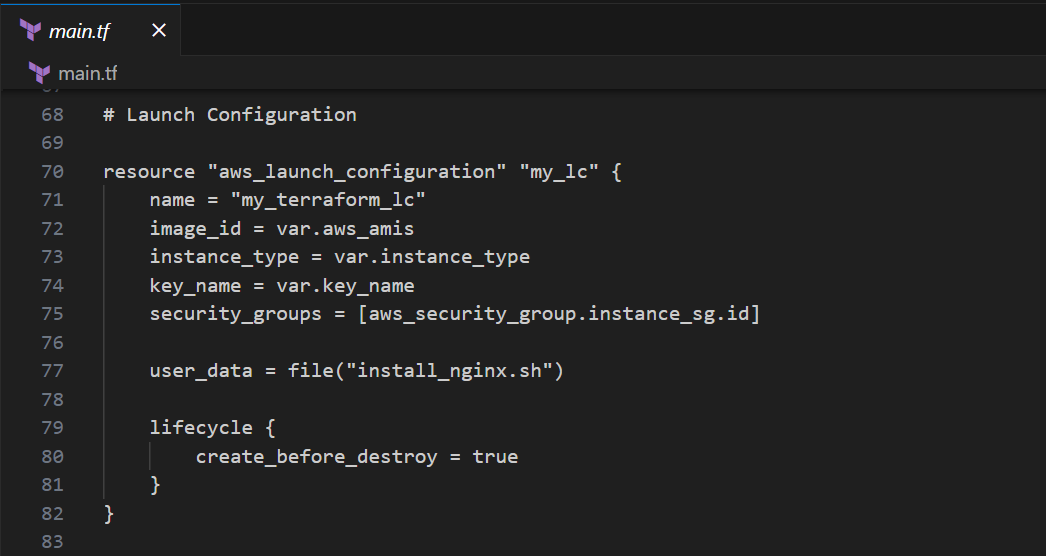


Terraform script resources:

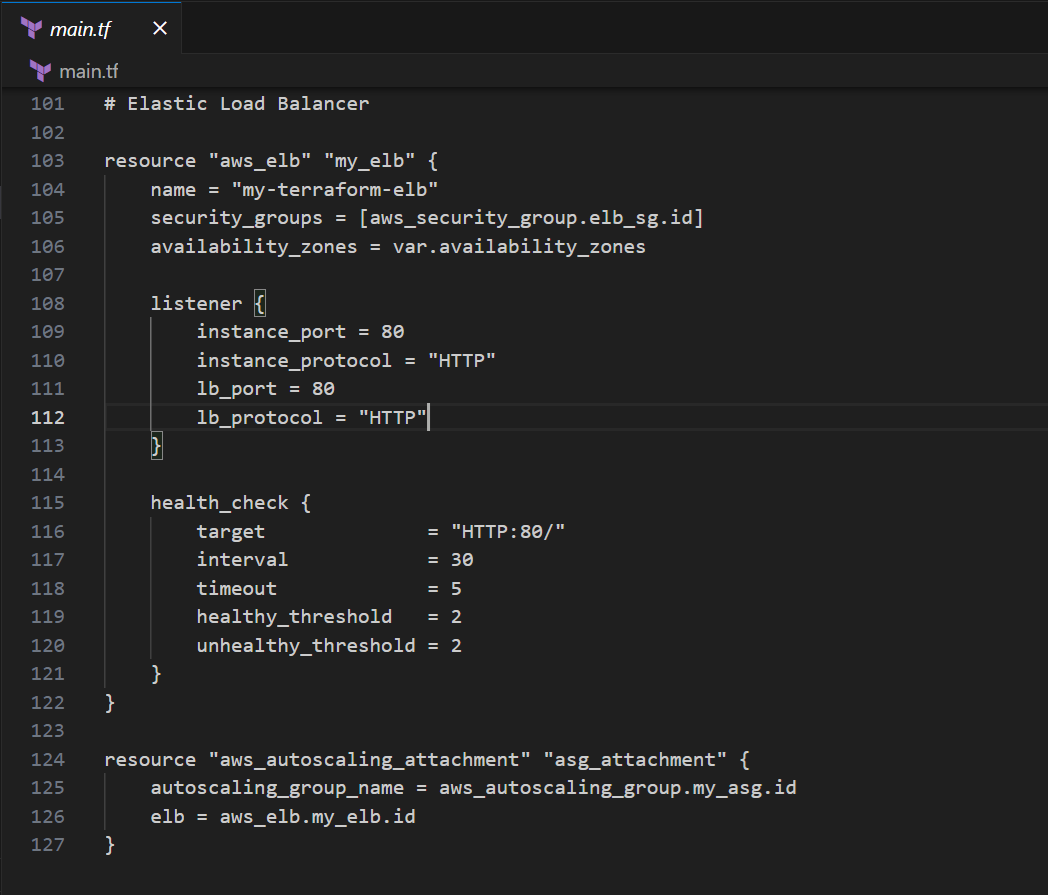
Auto Scaling Group (ASG) –



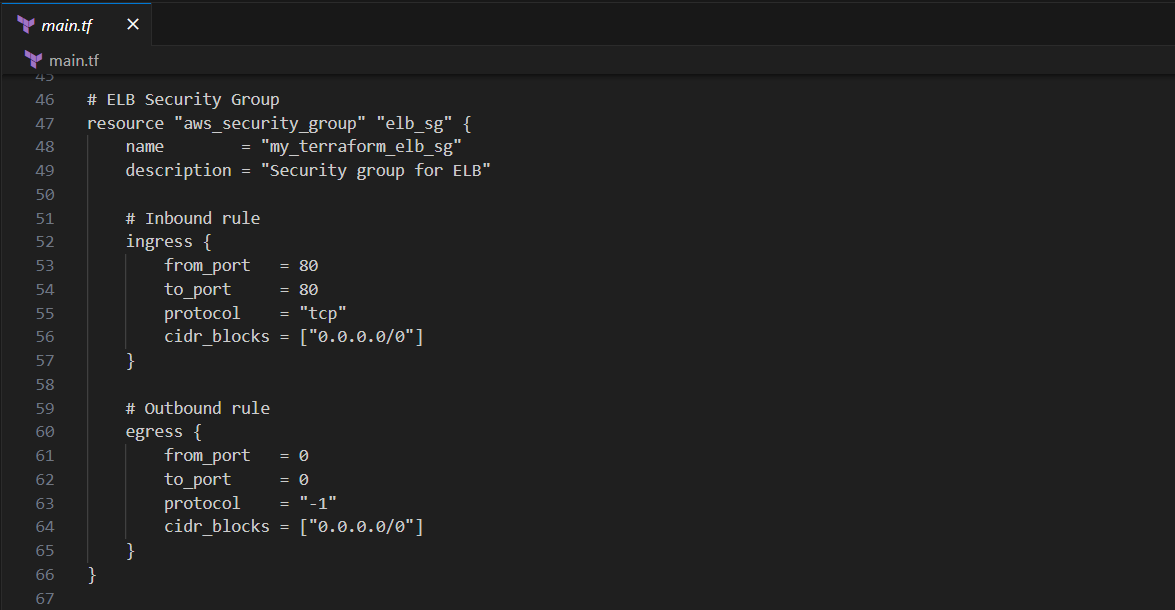
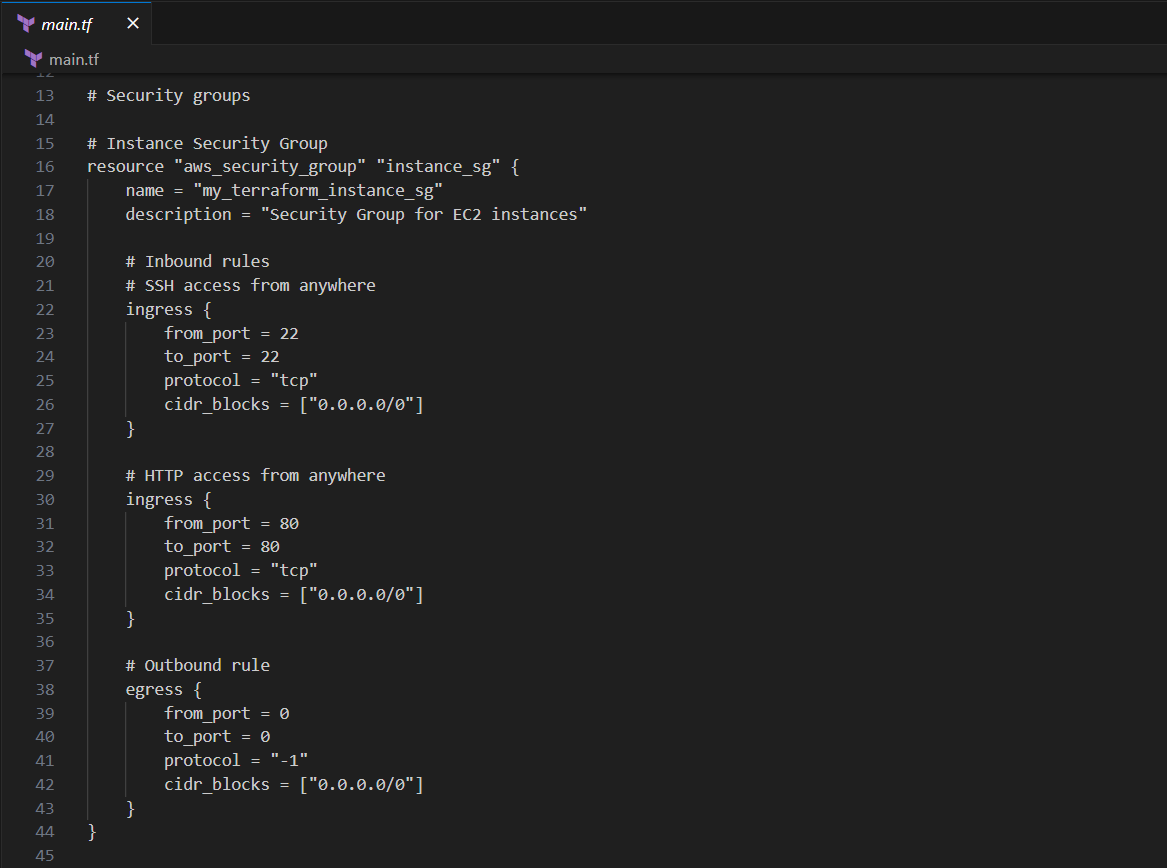
Launch Configuration (LC) –



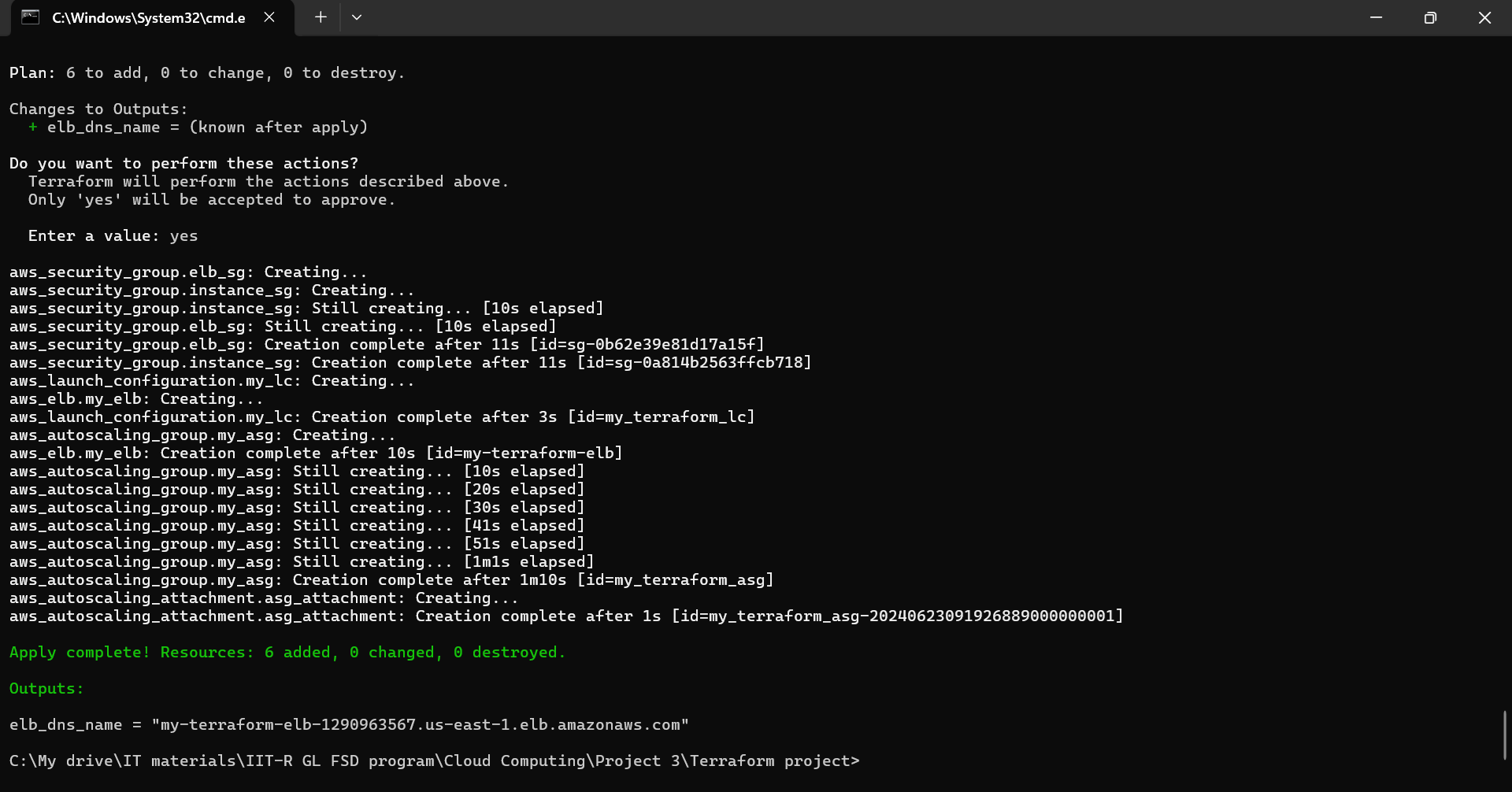
Elastic Load Balancer (ELB) –



Security Groups (SG) –



Executing of Terraform script and output –



------- \*\*\*\*\* -------