**DevOps and Cloud**

**March 2025**



# Terraform Fundamentals

Homework (M4)

Vasil Atanasov

@VasAtanasov

## Environment Setup

A single VM will be set up with Vagrant. All tasks will be performed inside Debian VM.

Docker and Terraform installed on the Debian VM. This is convenient because we can leverage auto completion and generally, I feel better in the Linux Bash terminal than PowerShell.

## Terraform and Docker

### Remote Image with Local Mount

The files for the task are located inside terraform/task-1a folder. If we explore the folder we will see terraform code split into separate files: main.tf, variables.tf and terraform.tfvars.

First, we need to get the project. Clone the repo in terraform/task-1a. Later the a vagrant trigger will clean it up on destroy.

git clone https://github.com/shekeriev/bgapp.git

We will be using docker provider and to download it we need to initialize the project with:

terraform init

A screenshot of a computer program

AI-generated content may be incorrect.

To validate that our configuration is correct we use:

terraform validate



When we explore the main.tf file we will see the following definition:

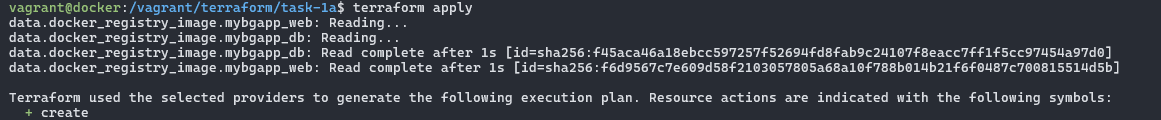
data "docker\_registry\_image" "mybgapp\_db" {  
 name = "${var.mybgapp\_db\_image\_repo}:${var.mybgapp\_db\_image\_tag}"  
}  
  
resource "docker\_image" "mybgapp\_db" {  
 name = data.docker\_registry\_image.mybgapp\_db.name

pull\_triggers = [data.docker\_registry\_image.mybgapp\_db.sha256\_digest]  
}

This gives the ability to update the image dynamically when there is a sha256 sum change. So, to work we need both docker\_image resource and docker\_image\_registry data. The same applies for the mybgaap\_web image.

To provision docker execute

terraform apply



A black background with white lines

AI-generated content may be incorrect.

To check if all is working go to <http://localhost:8000> on the Host OS.

A map of the country with a flag

AI-generated content may be incorrect.

### Local Image Build

The files for the task are located inside terraform/task-1b folder. If we explore the folder, we will see similar terraform code split into separate files: main.tf, variables.tf and terraform.tfvars like in the first task.

First, we need to get the project. While in the folder task-1b clone the repo. Later it will be cleaned up by a vagrant trigger on destroy.

git clone https://github.com/shekeriev/bgapp.git

We will perform the same steps as the previous task.

terraform init

terraform validate

Let’s explore the docker image setup

resource "docker\_image" "bgapp\_db" {  
 name = "${var.mybgapp\_db\_image\_repo}:${var.mybgapp\_db\_image\_tag}"

keep\_locally = true  
  
 build {  
 context = "${path.cwd}/bgapp"

dockerfile = "${path.cwd}/bgapp/Dockerfile.db"

}  
}

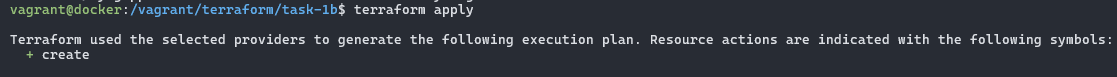
The file terraform.tfvars was also changed.

* Changed the db and web image names
* Changed the volume host path

Here we can see that the image needs to be built locally first before we can use it. The same applies for the bgapp\_web image.

We then provision docker

terraform apply



A screenshot of a computer code

AI-generated content may be incorrect.

If we brows <http://localhost:8000> on the Host OS, we will see the site loading.

A map of the country

AI-generated content may be incorrect.

## Terraform and AWS

### Setup AWS User

To not use the root user, I created an additional and more limited use account for the purpose of the task through the Identity and Access Management service. The user will be used only with the CLI it has no access to the console but an access key which will be saved to the environment.

A screenshot of a computer

AI-generated content may be incorrect.

### Setup AWS V2 CLI

To set up the aws cli tool and configure it with the access key I have saved the access key id and access key secret to environment variables on my Host OS (Windows 11) and load them in the Vagrantfile configuration which passes the variables to the aws-cli-setup.sh script. The script installs and configures the aws cli for the vagrant user if both AWS\_ACCESS\_KEY\_ID and AWS\_SECRET\_ACCESS\_KEY are present.

terraform.vm.provision 'install-aws-cli', type: :shell, privileged: false do |shell|  
 shell.env = {  
 'AWS\_ACCESS\_KEY\_ID' => "#{ENV['AWS\_ACCESS\_KEY\_ID']}",  
 'AWS\_SECRET\_ACCESS\_KEY' => "#{ENV['AWS\_SECRET\_ACCESS\_KEY']}",  
 'AWS\_DEFAULT\_REGION' => "#{ENV.fetch('AWS\_DEFAULT\_REGION', 'eu-central-1')}",  
 'AWS\_OUTPUT\_FORMAT' => "#{ENV.fetch('AWS\_OUTPUT\_FORMAT', 'json')}"

}  
 shell.path = "#{SCRIPTS\_DIR}/aws-cli-setup.sh"

end

A screen shot of a computer

AI-generated content may be incorrect.

### Explore Terraform Project Structure

A black background with white text

AI-generated content may be incorrect.

### Download the AWS Provider

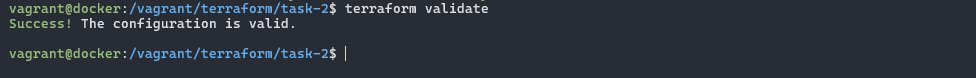
terraform init

A screenshot of a computer

AI-generated content may be incorrect.

### Validate Terraform Code

terraform validate



### Terraform Plan

To visually verify what will be created on AWS we execute.

terraform plan

A black background with white text

AI-generated content may be incorrect.

### Provision of the Infrastructure

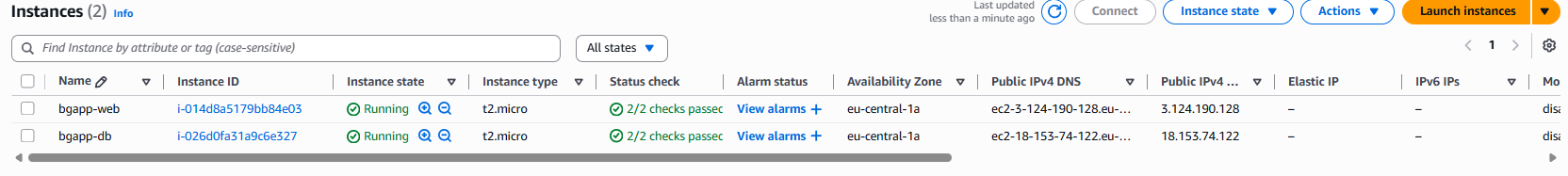
terraform apply

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



From the screenshots it is visible we can access the machines with SSH and make ping request in the private network.

### SSH Setup

To be able to login into the machines we need a keypair which I generated beforehand and stored on my Host machine. I will transfer it to the VM with the following command:

scp -P 2222 C:\Users\v.atanasov\.ssh\terraform-key.pem vagrant@127.0.0.1:/home/vagrant/.ssh/terraform-key.pem

Then restrict the key usage only to the owner user (vagrant)  
A screen shot of a computer

AI-generated content may be incorrect.

### DB Server Setup

I will ssh into the server machine and manually set up the database

#### SSH into DB Server

To be able to log in we need to specify the location of the key

ssh -i $HOME/.ssh/terraform-key.pem admin@ec2-18-153-74-122.eu-central-1.compute.amazonaws.com

A computer screen shot of a program

AI-generated content may be incorrect.

#### Install MariaDB Server

sudo apt update && sudo apt install -y mariadb-server git

#### Make the Database Accessible

By default, the server will start listening for connections on 127.0.0.1:3306. To be able to connect to the database from the web server we need to modify the configuration to accept connections from the outside. First remove the bind-address line and replace it with the new one.

sudo sed -i '/^\s\*bind-address\s\*=/d' /etc/mysql/mariadb.conf.d/50-server.cnf

sudo sed -i '/^\[mysqld\]/a bind-address = 0.0.0.0' /etc/mysql/mariadb.conf.d/50-server.cnf

#### Enable and start the MariaDB service

sudo systemctl enable --now mariadb

#### Download the project

git clone https://github.com/shekeriev/bgapp ~/bgapp

#### Install the database required for the web application

sudo mysql -u root < ~/bgapp/db/db\_setup.sql

At this point we should have our db running and ready to accept connections from web server.

### Web Server Setup

I will ssh into the server machine and manually set up the web server

#### SSH into Web Server

ssh -i $HOME/.ssh/terraform-key.pem admin@ec2-3-124-190-128.eu-central-1.compute.amazonaws.com

#### Install the required packages

sudo apt update && sudo apt install -y apache2 php php-mysql git

#### Enable and start the Apache2 service

sudo systemctl enable --now apache2

#### Download the project

git clone https://github.com/shekeriev/bgapp ~/bgapp

#### Copy the files related to the web application

sudo cp ~/bgapp/web/\* /var/www/html/

#### Substitute MariaDB related connection parameters

We don’t have name resolution so we need to change the host from DB to the IP address of the database server.

sudo sed -i 's/db/10.10.10.101/g' /var/www/html/config.php

A screenshot of a computer

AI-generated content may be incorrect.

### Destroy the Infrastructure

To destroy all configurations related to the infrastructure we execute

terraform destroy

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.