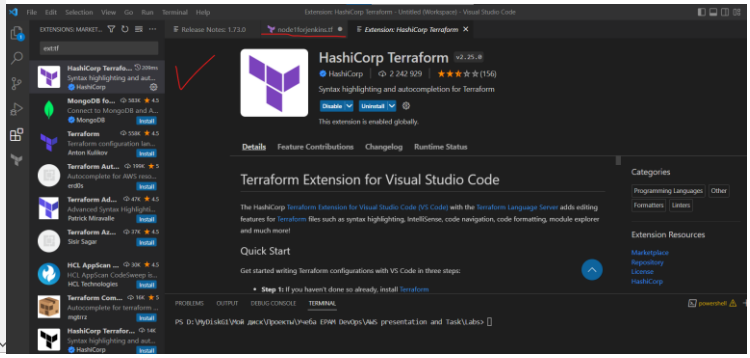
# Terraform

## Install and check on windows and Visual Studio Code

```
Администратор: C:\Windows\System32\cmd.exe

d:\Downloads>dir c:\terraform
 Volume in drive C has no label.
 Volume Serial Number is 889C-A276

 Directory of c:\terraform

23.11.2022  10:18    <DIR>          .
23.11.2022  10:18    <DIR>          ..
17.11.2022  19:45        62 128 376 terraform.exe
               1 File(s)     62 128 376 bytes
               2 Dir(s)  15 832 952 832 bytes free

d:\Downloads>echo %PATH%
C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Windows\System32\
OpenSSH\;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files\Intel\Intel(R) Manageme
nt Engine Components\DAL;C:\Program Files\dotnet\;c:\page;C:\Program Files\PuTTY\;C:\Program Files\Git\cmd;C:\Users\Влад
имир\AppData\Local\Programs\Python\Python38;C:\Users\Владимир\AppData\Local\Programs\Python\Python38\Scripts;C:\Users\Вл
адимир\AppData\Local\Microsoft\WindowsApps;C:\Users\Владимир\.dotnet\tools;C:\Program Files\JetBrains\PyCharm Community
Edition 2022.1.3\bin;C:\Users\Владимир\AppData\Local\Programs\Microsoft VS Code\bin;C:\Terraform

d:\Downloads>terraform version
Terraform v1.3.5
on windows_amd64

d:\Downloads>
```




## Create 2 files in plan1 dir:

Release Notes: 1.73.0 | instances.tf | main.tf

Terraform > tasks > plan1 > instances.tf > resource "aws_instance" "jenkins-node1"

```hcl
resource "aws_instance" "jenkins-node1" {
  ami           = "ami-0f15e0a4c8d3ee5fe"
  instance_type = "t2.micro"

  tags = {
    Name    = "jenkins-node1"
    Projecr = "homework"
  }
}
```

instances.tf | main.tf

Terraform > tasks > plan1 > main.tf > provider "aws"

```hcl
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

provider "aws" {
  region  = "eu-west-3"
}
```

## terraform init

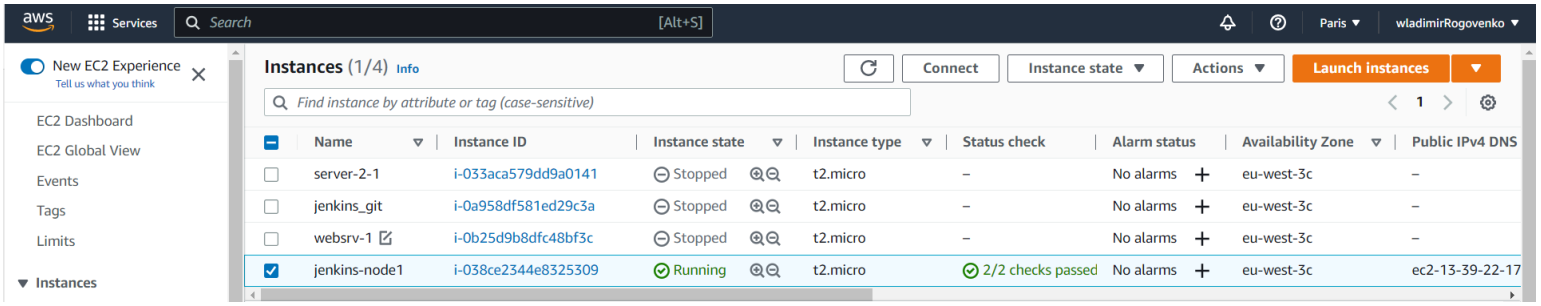
```
PS D:\MyDiskG1\Мой диск\Проекты\Учеба EPAM DevOps\Terraform\tasks\plan1> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 4.16"...
- Installing hashicorp/aws v4.40.0...
- Installed hashicorp/aws v4.40.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS D:\MyDiskG1\Мой диск\Проекты\Учеба EPAM DevOps\Terraform\tasks\plan1>
```

## terraform plan

```
PS D:\MyDiskG1\Мой диск\Проекты\Учеба EPAM DevOps\Terraform\tasks\plan1> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.jenkins-node1 will be created
  + resource "aws_instance" "jenkins-node1" {
      + ami                                  = "ami-0f15e0a4c8d3ee5fe"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_state                       = (known after apply)
```

## Terraform apply

```
PS D:\MyDiskG1\Мой диск\Проекты\Учеба EPAM DevOps\Terraform\tasks\plan1> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.jenkins-node1 will be created
  + resource "aws_instance" "jenkins-node1" {
      + ami                                  = "ami-0f15e0a4c8d3ee5fe"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name"    = "jenkins-node1"
          + "Projecr" = "homework"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)

      + capacity_reservation_specification {
          + capacity_reservation_preference = (known after apply)

          + capacity_reservation_target {
              + capacity_reservation_id = (known after apply)
            }
        }

      + cpu_options {
          + core_count       = (known after apply)
          + threads_per_core = (known after apply)
        }

      + ebs_block_device {
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + snapshot_id           = (known after apply)
          + tags                  = (known after apply)
          + throughput            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }

      + enclave_options {
          + enabled = (known after apply)
        }

      + ephemeral_block_device {
          + device_name  = (known after apply)
          + no_device    = (known after apply)
          + virtual_name = (known after apply)
        }

      + maintenance_options {
          + auto_recovery = (known after apply)
        }

      + metadata_options {
          + http_endpoint               = (known after apply)
          + http_put_response_hop_limit = (known after apply)
          + http_tokens                 = (known after apply)
          + instance_metadata_tags      = (known after apply)
        }

      + network_interface {
          + delete_on_termination = (known after apply)
          + device_index          = (known after apply)
          + network_card_index    = (known after apply)
          + network_interface_id  = (known after apply)
        }

      + private_dns_name_options {
          + enable_resource_name_dns_a_record    = (known after apply)
          + enable_resource_name_dns_aaaa_record = (known after apply)
          + hostname_type                        = (known after apply)
        }

      + root_block_device {
          + delete_on_termination = (known after apply)
          + device_name           = (known after apply)
          + encrypted             = (known after apply)
          + iops                  = (known after apply)
          + kms_key_id            = (known after apply)
          + tags                  = (known after apply)
          + throughput            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.jenkins-node1: Creating...
aws_instance.jenkins-node1: Still creating... [10s elapsed]
aws_instance.jenkins-node1: Still creating... [20s elapsed]
aws_instance.jenkins-node1: Creation complete after 22s [id=i-038ce2344e8325309]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```




Please note: using terraform I will show in my finally project for real example (create S3 bucket for tfstate, VPC and network subnets)

There are some parts tf script of them:

```
#======================== for Jenkins node 1 ================================
#defautl VPC
data "aws_vpc" "default-1" {
  default = true #its filter
}               #for read this var: data.aws_vpc.default-1.id

#default SubNet
data "aws_subnets" "default" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.default-1.id]
  }
  filter {
    name   = "default-for-az"
    values = [true]
  }
}

#create Security Group for 80, 8081, 22 ports
resource "aws_security_group" "secgrp-Linux_80_22" {
  name = "secgrp-Linux_80_22"
  description = "Allow 80 and 22 ports. For Linux web server"
  vpc_id      = data.aws_vpc.default-1.id
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
    ingress {
    from_port   = 8081
    to_port     = 8081
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port        = 0
```

```
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"]
    #ipv6_cidr_blocks = ["::/0"]
  }
  tags = {
    "Name" = "${var.env}-secgrp-Linux_80_22"
  }
}
```

```
# outputs vars

output "env" {
  description = "Name of project"
    value     = var.env  # dev / prod / test
}

output "aws_region" {
  description = "Current AWS region"
  value     = var.aws_region
}


output "aws_vpc_id" {
  description = "Current VPC"
  value     = data.aws_vpc.default-1.id
}

output "aws_subnet_id" {
  description = "Current Subnet in VPC"
  value     = sort(data.aws_subnets.default.ids)[0]
}


output "aws_security_group_id" {
  description = "Create new SG with ports 80,22"
  value     = aws_security_group.secgrp-Linux_80_22.id
}

output "aws_instance_Jenkins-node_public_ip" {
  description = "Public IP address of Jenkins-node agent instance"
  value     = aws_instance.jenkins-node-1.public_ip
}

output "aws_instance_Jenkins-node_private_ip" {
  description = "Private IP address of Jenkins-node agent instance"
  value = aws_instance.jenkins-node-1.private_ip
}
```

```
#================= create bucket s3 for Artifacts storage ======================================

resource "aws_s3_bucket" "s3_artifacts_storage" {
  bucket = "p-petclinic2023-jenkins-archive-artifacts"
  lifecycle {
    #prevent_destroy = true #prevent randomly delete backet
  }
  versioning {
    enabled = false
  }
  server_side_encryption_configuration {
    rule {
      apply_server_side_encryption_by_default {
        sse_algorithm = "AES256"
      }
    }
  }
  tags = {
    "project" = "petclinic"
  }
}


#================================================================================================
```

```
#================= create bucket s3 for TF state ================================================

resource "aws_s3_bucket" "terraform_state" {
  bucket = "tfstate-final-project-java2022"
  lifecycle {
    #prevent_destroy = true #prevent randomly delete backet
  }
  versioning {
    enabled = true
  }
  server_side_encryption_configuration {
    rule {
      apply_server_side_encryption_by_default {
        sse_algorithm = "AES256"
      }
    }
  }
}
```