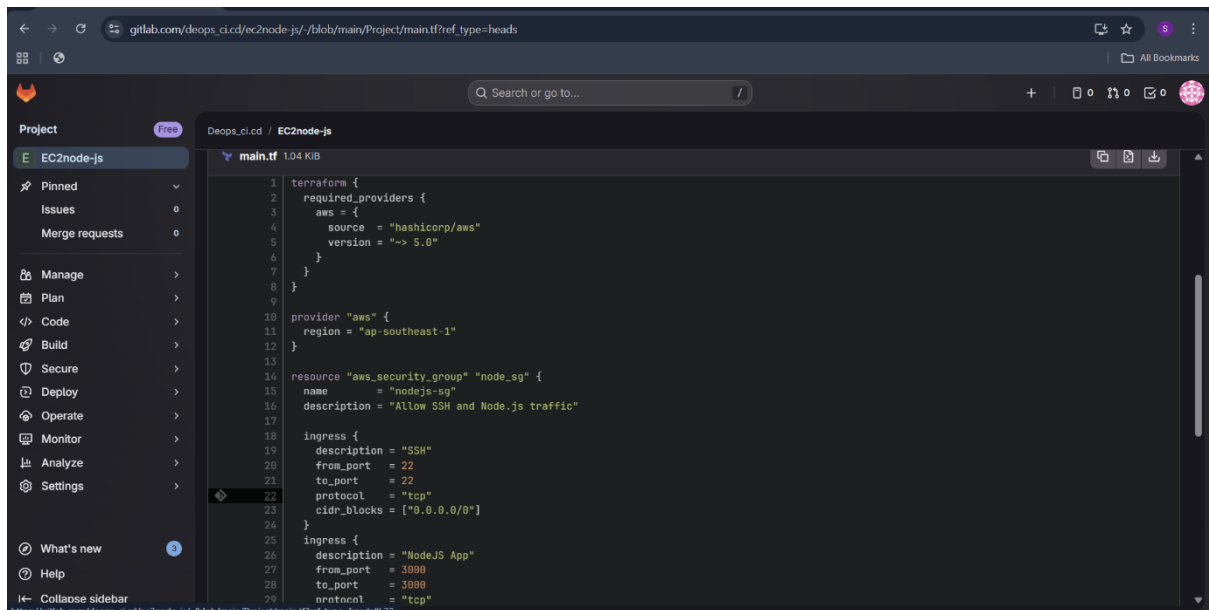


## Weekly Test

# Deploy the node js application in EC2 using terraform and gitlab ci/cd

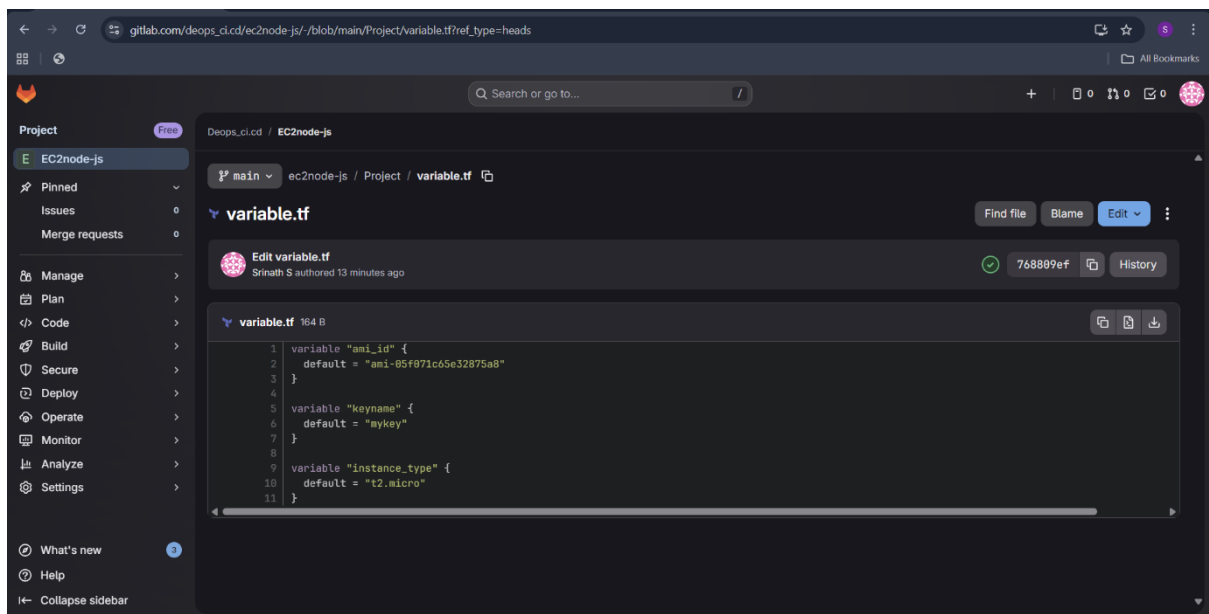
## Create main.tf in Gitlab:



The screenshot shows the GitLab web interface for a project named 'EC2node-js'. The file 'main.tf' is open, displaying Terraform configuration for an AWS environment. The configuration includes required providers for AWS, an AWS provider block for the 'ap-southeast-1' region, and an AWS security group resource named 'node\_sg' with two ingress rules: one for SSH (port 22) and one for Node.js (port 3080).

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "~> 5.0"
6     }
7   }
8 }
9
10 provider "aws" {
11   region = "ap-southeast-1"
12 }
13
14 resource "aws_security_group" "node_sg" {
15   name = "nodejs-sg"
16   description = "Allow SSH and Node.js traffic"
17
18   ingress {
19     description = "SSH"
20     from_port = 22
21     to_port = 22
22     protocol = "tcp"
23     cidr_blocks = ["0.0.0.0/0"]
24   }
25   ingress {
26     description = "NodeJS App"
27     from_port = 3080
28     to_port = 3080
29     protocol = "tcp"
30   }
31 }
```

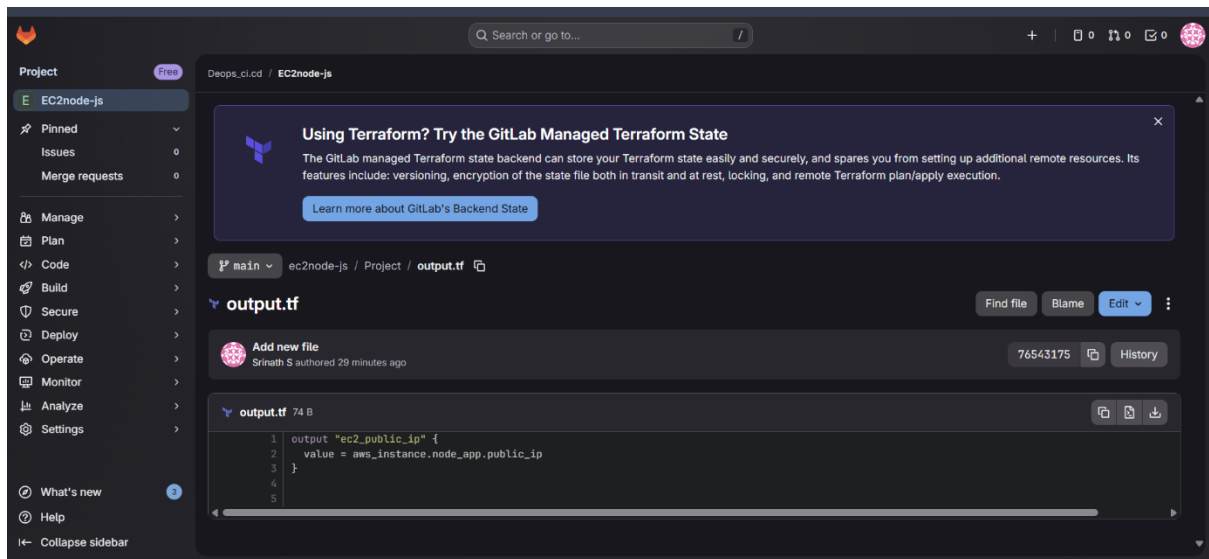
## Create Variable.tf file:



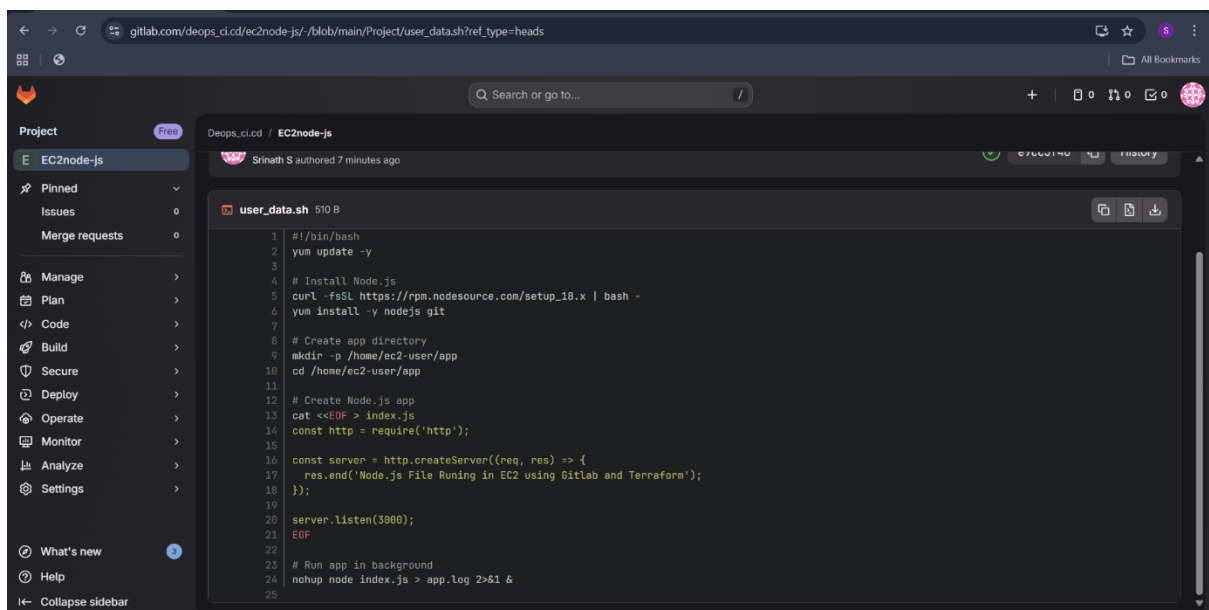
The screenshot shows the GitLab web interface for the same project, now displaying the 'variable.tf' file. The file contains three variable definitions: 'ami\_id' with a default value of 'ami-05f971c65e32875a8', 'keyname' with a default value of 'mykey', and 'instance\_type' with a default value of 't2.micro'.

```
1 variable "ami_id" {
2   default = "ami-05f971c65e32875a8"
3 }
4
5 variable "keyname" {
6   default = "mykey"
7 }
8
9 variable "instance_type" {
10  default = "t2.micro"
11 }
```

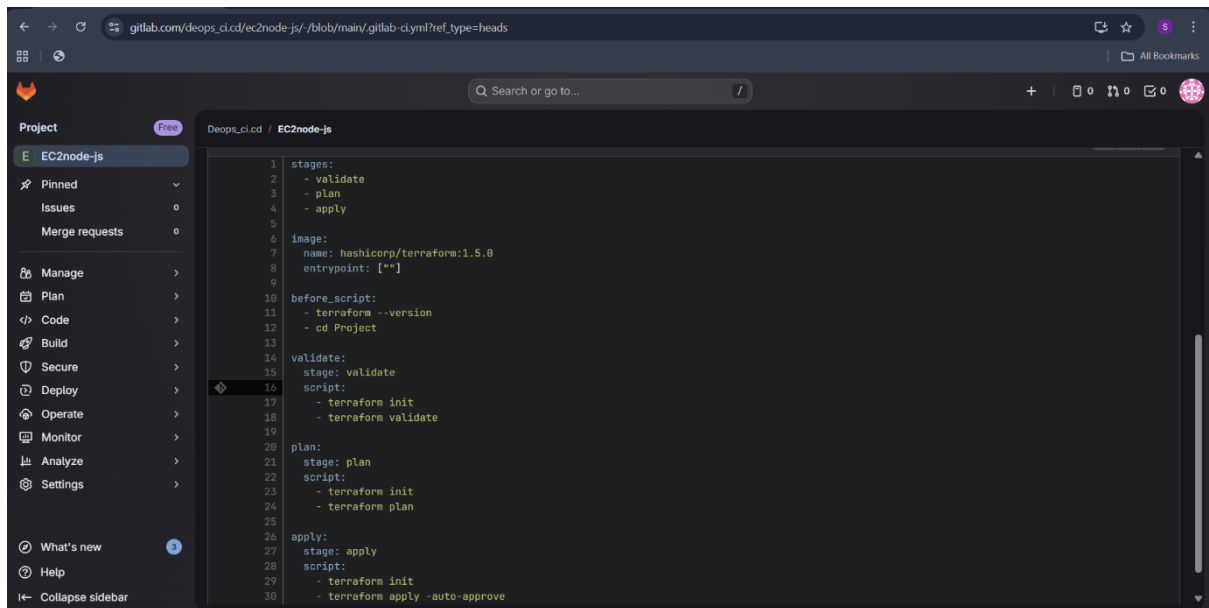
## Create output.tf file:



## Create user data File:



## Create .gitlab-ci.yml file:



Code:

stages:

- validate
- plan
- apply

image:

name: hashicorp/terraform:1.5.0  
entrypoint: [""]

before\_script:

- terraform --version
- cd Project

validate:

stage: validate

script:

- terraform init
- terraform validate

plan:

stage: plan

script:

- terraform init
- terraform plan

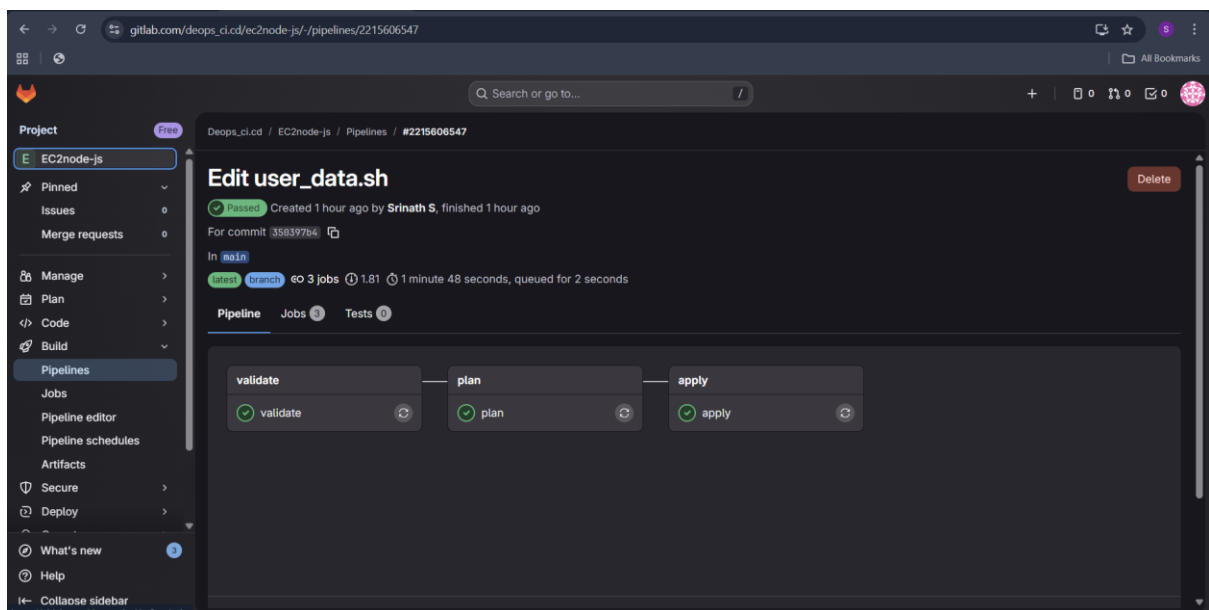
apply:

stage: apply

script:

- terraform init
- terraform apply -auto-approve

## CI/CD pipe line:



## Output:

