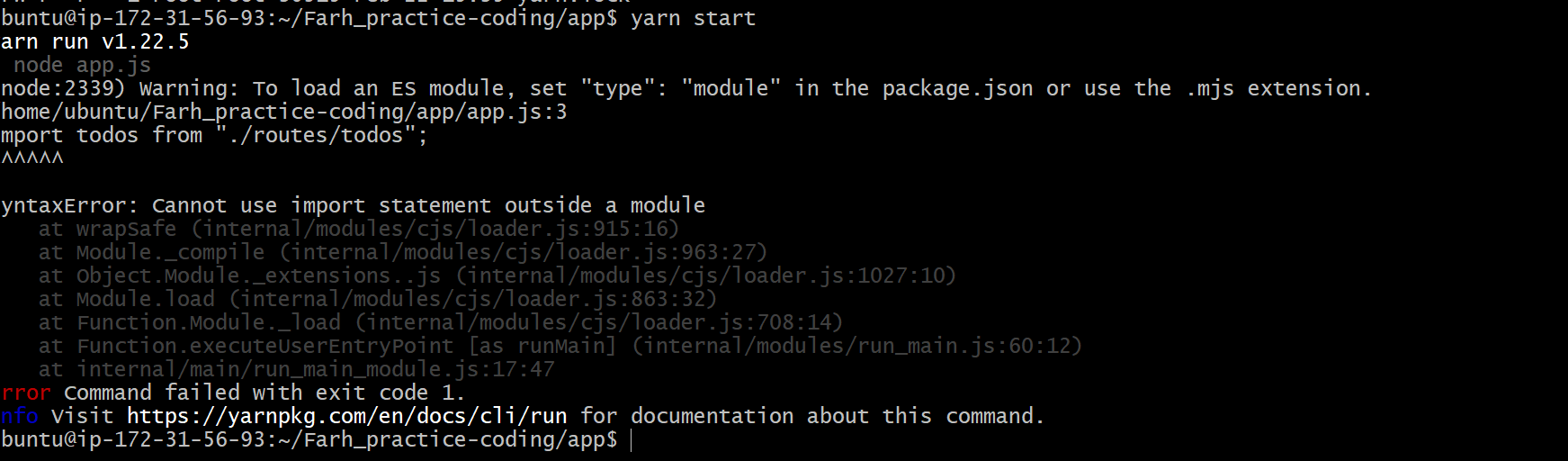
This is a documentation to build a fully automated infrastructure and CI/CD solution for a simple web-based application (Nodejs and mondodb)

This coding challenge is interesting.

* I have used GitHub, AWS, Docker, terraform, Kubernetes and helm to build a fully automated infrastructure and CI/CD solution
* I have used Github as a SCM tool to mange my code, added the dockerfile , created the image using github actions and pushed the same to docker repository(docker hub) completing my continuous integration.
* I have used terraform to build the infrastructure resources such as VPC, subnets, EC2 ELB, security groups etc. Then used KOPS to build Kubernetes cluster.
* I Created the kompose file to bring the nodejs and mongodb application up.
* I have Installed the helm charts to deploy the application (helm files can be found in https://github.com/Farhad207/Farh\_practice-coding/tree/master/nodejs)

Git repo –> <https://github.com/Farhad207/Farh_practice-coding.git>

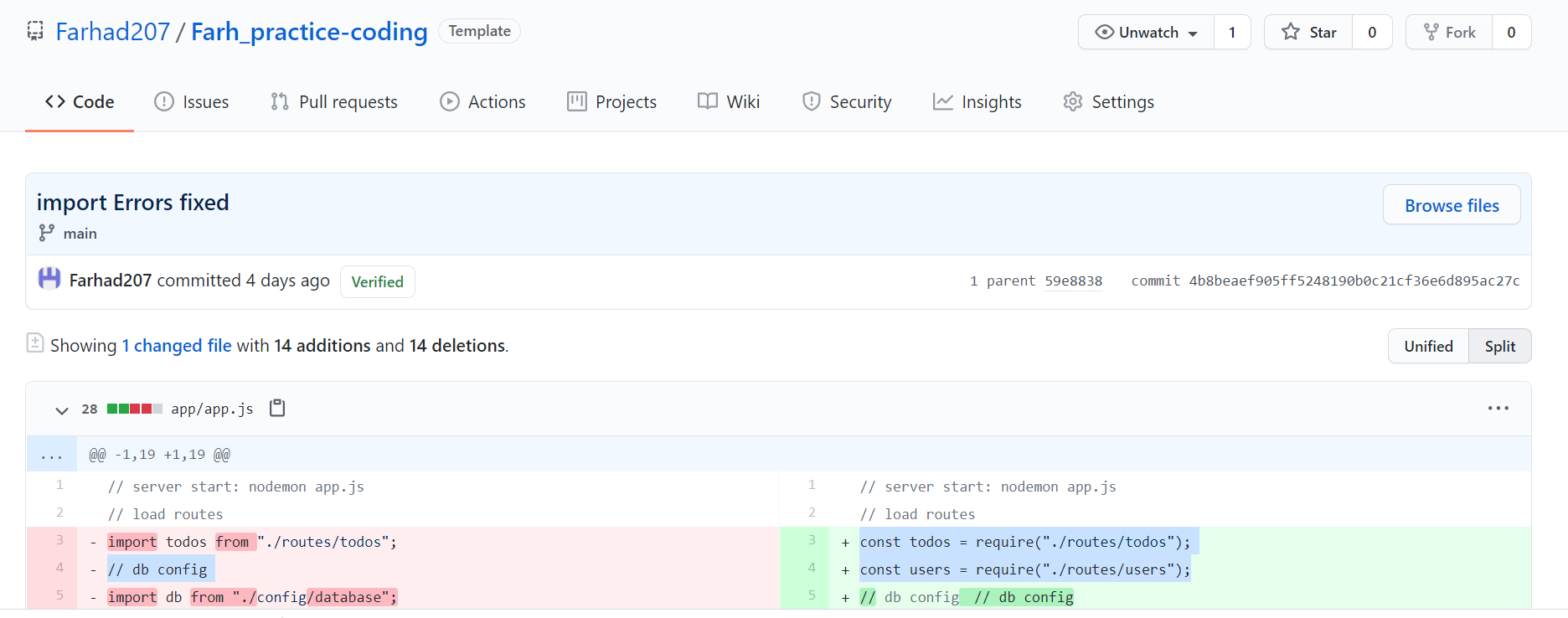
1. I cloned the Tradeling’s Git hub Repository and tried to run the yarn, and Yarn start, I got the import errors as shown below:

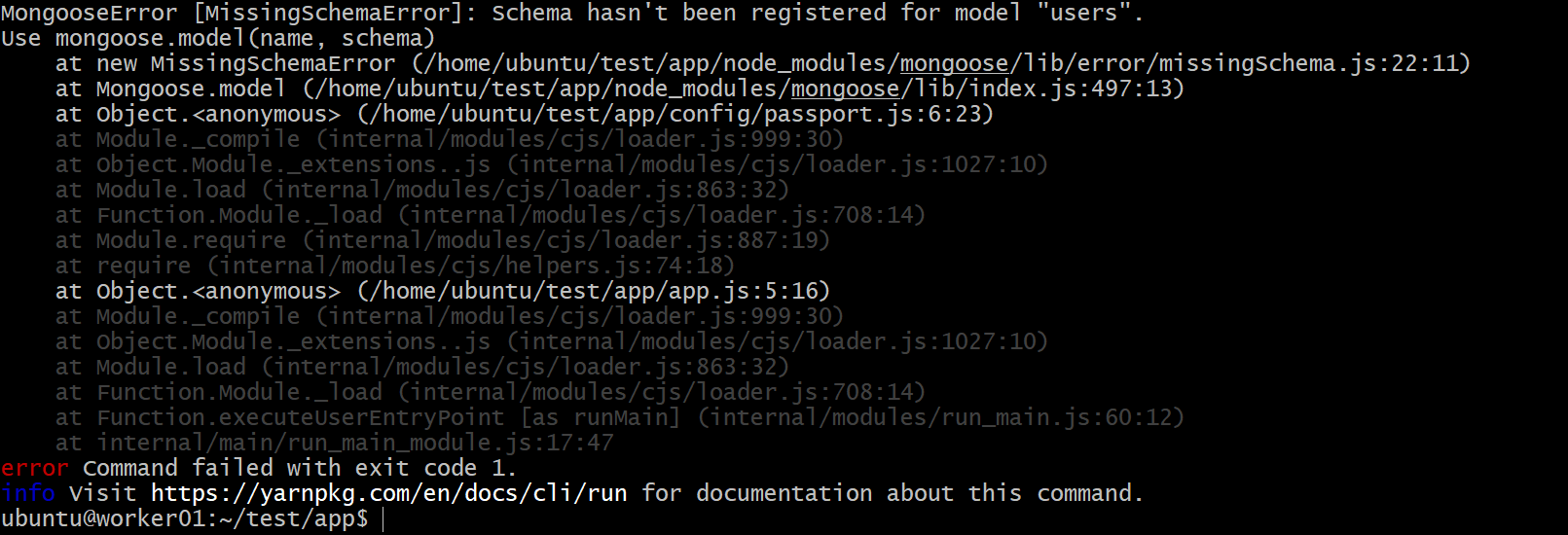


**Resolution for above Error:** I added conts todos and users as shown below, I used require instead of from and const instead of import. This can be found in the commit (commit id - 4b8beaef905ff5248190b0c21cf36e6d895ac27c)

*const todos = require("./routes/todos");*

*// db config const users = require("./routes/users");*



1. When I again ran the yarn start I received the Mangoose schema not registered errors, as shown below 

ERROR throw new mongoose.Error.MissingSchemaError(name);

^

MongooseError [MissingSchemaError]: Schema hasn't been registered for model "users".

Use mongoose.model(name, schema)

at new MissingSchemaError (/home/ubuntu/Farh\_practice-coding/app/node\_modules/mongoose/lib/error/missingSchema.js:22:11)

at Mongoose.model (/home/ubuntu/Farh\_practice-coding/app/node\_modules/mongoose/lib/index.js:497:13)

at Object.<anonymous> (/home/ubuntu/Farh\_practice-coding/app/config/passport.js:6:23)

at Module.\_compile (internal/modules/cjs/loader.js:999:30)

at Object.Module.\_extensions..js (internal/modules/cjs/loader.js:1027:10)

at Module.load (internal/modules/cjs/loader.js:863:32)

at Function.Module.\_load (internal/modules/cjs/loader.js:708:14)

at Module.require (internal/modules/cjs/loader.js:887:19)

at require (internal/modules/cjs/helpers.js:74:18)

at Object.<anonymous> (/home/ubuntu/Farh\_practice-coding/app/app.js:5:16)

at Module.\_compile (internal/modules/cjs/loader.js:999:30)

at Object.Module.\_extensions..js (internal/modules/cjs/loader.js:1027:10)

at Module.load (internal/modules/cjs/loader.js:863:32)

at Function.Module.\_load (internal/modules/cjs/loader.js:708:14)

at Function.executeUserEntryPoint [as runMain] (internal/modules/run\_main.js:60:12)

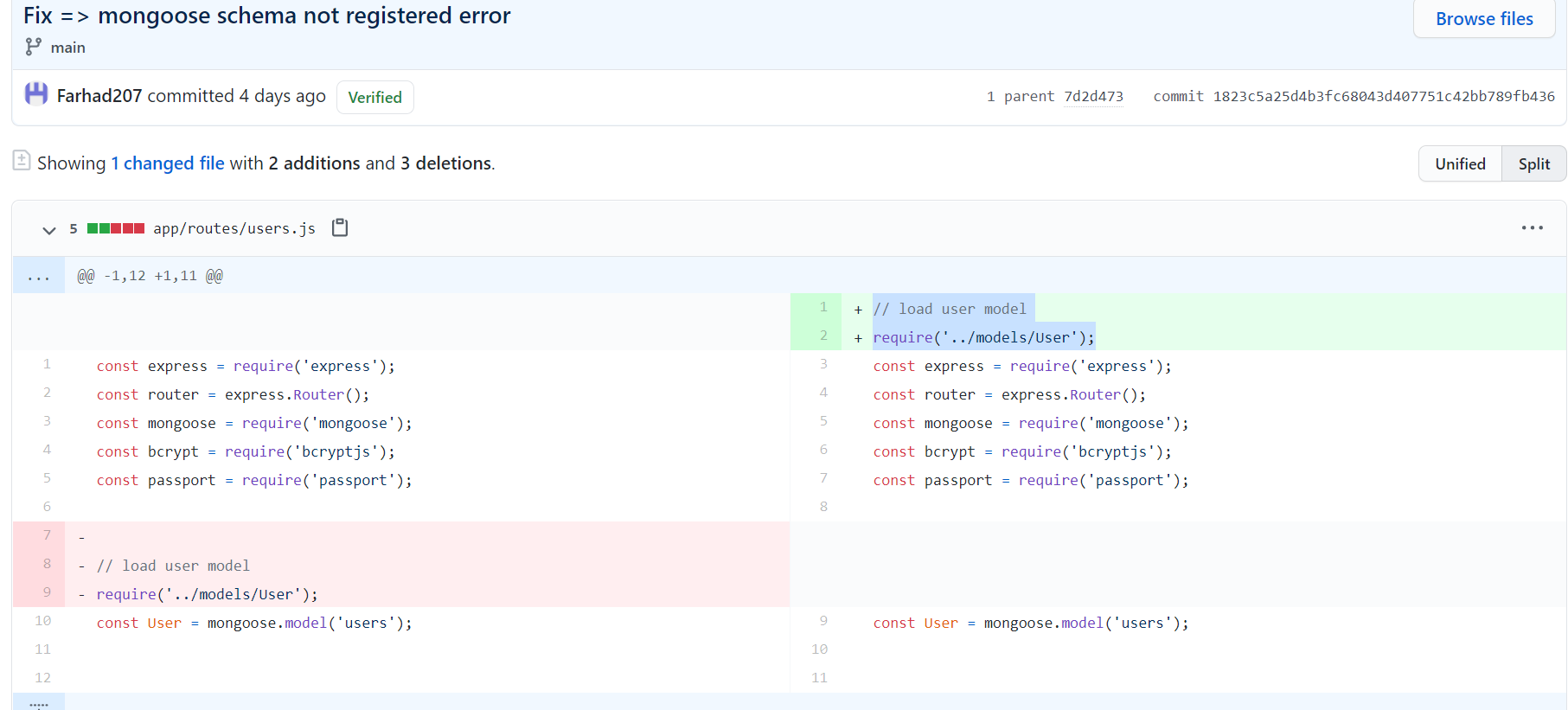
at internal/main/run\_main\_module.js:17:47

error Command failed with exit code 1.

info Visit https://yarnpkg.com/en/docs/cli/run for documentation about this command.info Visit https://yarnpkg.com/en/docs/cli/run for documentation about this command.

Resolution

The above errors were fixed by adding the require(“../ module/user’) above the express as shown in the screenshot (CommitID from github 1823c5a25d4b3fc68043d407751c42bb789fb436)



**After the above changes I was able to run the application locally.**

Continuous integration

1. Containerize the application.

For containerizing the application, I used docker as a runtime engine and wrote a Dockerfie as shown below. The same is updated in GitHub .

*FROM node:current-buster-slim*

*RUN mkdir -p /code*

*WORKDIR /code*

*ADD . /code*

*RUN npm install yarn && \*

*yarn install*

*CMD [ "yarn", "start" ]*

1. Build the image and push it to dockerhub

I used github actions here >>write your own workflows option. I added the below code to build the image and did the integration with Dockerhub to push it as and when we have a commit.

|  |
| --- |
|  |
|  |  |
|  |  |
| **Github -> dockerhub integration**   1. Go to your github acount click on builds 2. Click on link to github 3. Give your source repository, select the repo from where you want to push the images 4. Select the branch and save the settings.   Screenshots shown below: |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Build the resources using Terraform (Infrastructure as a code)**  I have used Kops to create a Kubernetes cluster on AWS. Below are the steps followed   1. Create an AWS instance and install the required packages such as pip, aws-cli . terraform then configure aws cli using AWS access Key ID and secret access key.  * sudo apt-get install python2-pip * sudo pip install awscli * Install the kubectl binaries * curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl" * mv ./kubectl ~/usr/bin     Similarly, download the latet version of Kops binaries and copy to the bin directory |  |
| curl -LO https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag\_name | cut -d '"' -f 4)/kops-linux-amd64 |  |
| **Install terraform v12**  ubuntu@ip-172-31-56-93:~$ terraform -v  Terraform v0.12.24  Your version of Terraform is out of date! The latest version  is 0.14.7. You can update by downloading from <https://www.terraform.io/downloads.html>  **Public key**  ubuntu@ip-172-31-56-93:~$ ls -lrt ~/.ssh/farhadkop.pub  -rw-r--r-- 1 ubuntu ubuntu 404 Feb 26 10:15 /home/ubuntu/.ssh/farhadkop.pub |  |
|  |  |
|  |  |
|  |  |
|  |  |
| Create hosted zone |  |
| **Create Kubernetes cluster using kops** |  |
|  |  |
| kops create cluster \  --name=testkop.farhad.com \  --state=s3://kops.farhad.com \  --authorization RBAC \  --zones=us-east-1a\  --node-count=1 \  --node-size=t2.micro \  --master-size=t2.medium \  --master-count=1 \  --dns-zone=farhad.com \  --out=kopskubernetes\_terraform \  --target=terraform \  --ssh-public-key=~/.ssh/farhadkop.pub |  |
|  |  |
|  |  |
|  |  |
| cd kopskubernetes\_terraform  /kopskubernetes\_terraform$ |  |
|  |  |

ls -lrt

total 28

-rw-r--r-- 1 ubuntu ubuntu 23769 Feb 28 07:37 **kubernetes.tf**

drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 28 07:37 data

**-terraform init**

ubuntu@ip-172-31-56-93:~/kopskubernetes\_terraform

**please execute the terraform plan command to show the infrastructure**

$ terraform plan -out=kops\_fartest.com

Refreshing Terraform state in-memory prior to plan...

The refreshed state will be used to calculate this plan, but will not be

persisted to local or remote state storage.

------------------------------------------------------------------------

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

# aws\_autoscaling\_group.master-us-east-1a-masters-testkop-farhad-com will be created

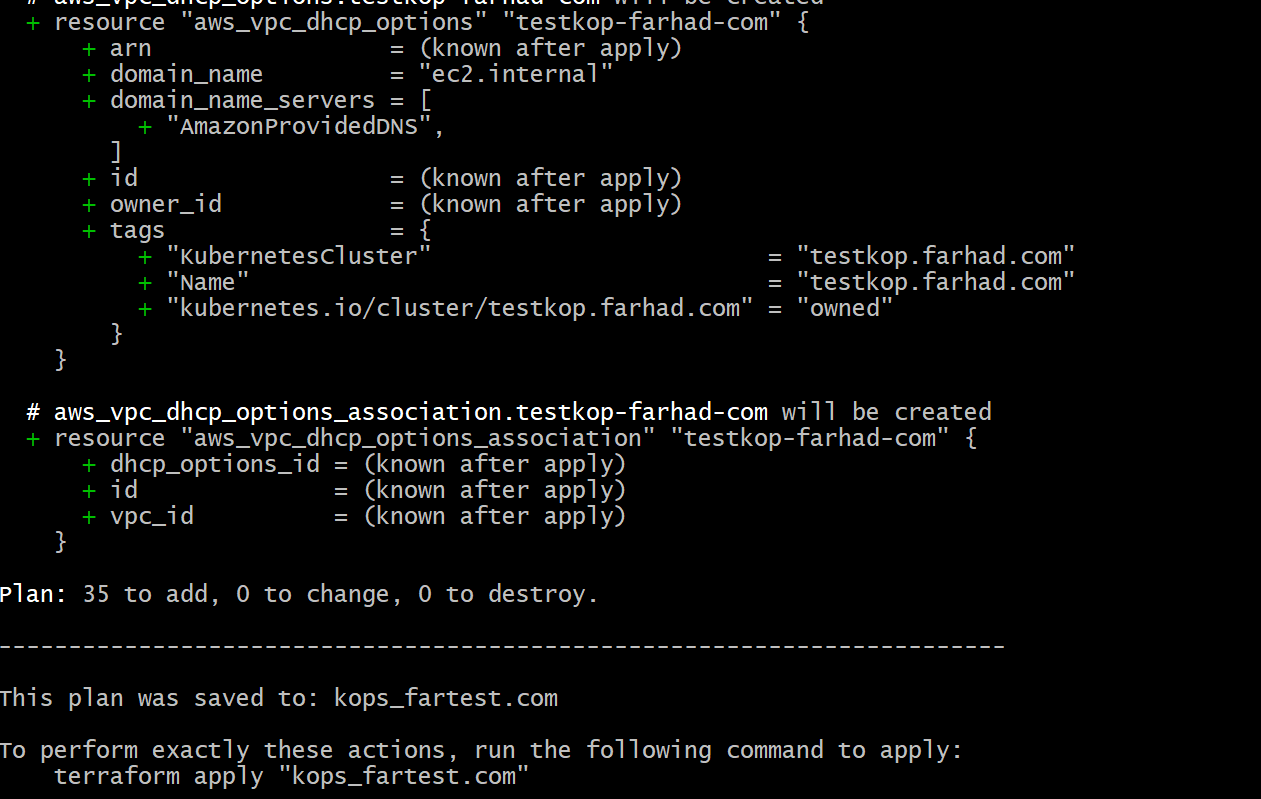
+ resource "aws\_autoscaling\_group" "master-us-east-1a-masters-testkop-farhad-com" {

+ arn = (known after apply)

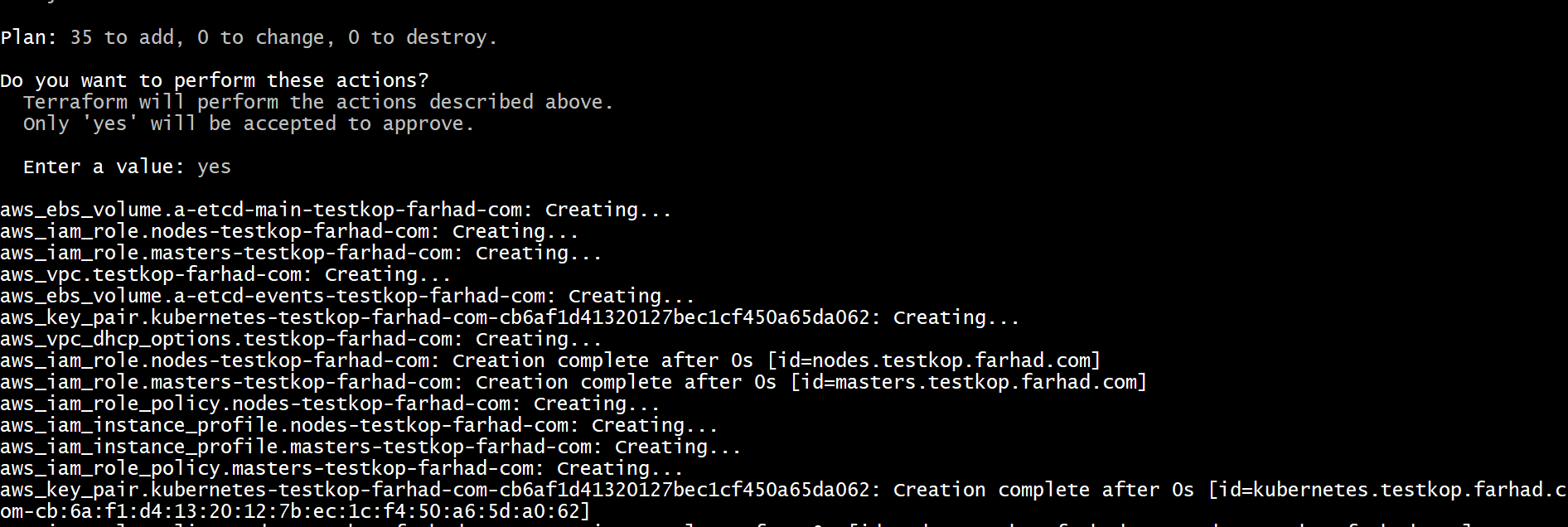
+ availability\_zones = (known after apply)

+ default\_cooldown = (known after apply)

+ desired\_capacity = (known after apply)



:~/kopskubernetes\_terraform$ terraform apply



**# aws\_launch\_template.master-us-east-1a-masters-testkop-farhad-com** will be created

+ resource "aws\_launch\_template" "master-us-east-1a-masters-testkop-farhad-com" {

+ arn = (known after apply)

+ default\_version = (known after apply)

+ id = (known after apply)

+ image\_id = "ami-0074ee617a234808d"

+ instance\_type = "t2.medium"

+ key\_name = (known after apply)

+ latest\_version = (known after apply)

+ name = "master-us-east-1a.masters.testkop.farhad.com"

+ tags = {

+ "KubernetesCluster" = "testkop.farhad.com"

+ "Name" = "master-us-east-1a.masters.testkop.farhad.com"

+ "k8s.io/cluster-autoscaler/node-template/label/kops.k8s.io/instancegroup" = "master-us-east-1a"

+ "k8s.io/cluster-autoscaler/node-template/label/kubernetes.io/role" = "master"

+ "k8s.io/cluster-autoscaler/node-template/label/node-role.kubernetes.io/master" = ""

+ "k8s.io/role/master" = "1"

+ "kops.k8s.io/instancegroup" = "master-us-east-1a"

+ "kubernetes.io/cluster/testkop.farhad.com" = "owned"

}

**# aws\_security\_group\_rule.ssh-external-to-master-0-0-0-0--0** will be created

+ resource "aws\_security\_group\_rule" "ssh-external-to-master-0-0-0-0--0" {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ from\_port = 22

+ id = (known after apply)

+ protocol = "tcp"

+ security\_group\_id = (known after apply)

+ self = false

+ source\_security\_group\_id = (known after apply)

+ to\_port = 22

+ type = "ingress"

}

**# aws\_security\_group\_rule.ssh-external-to-node-0-0-0-0--0** will be created

+ resource "aws\_security\_group\_rule" "ssh-external-to-node-0-0-0-0--0" {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ from\_port = 22

+ id = (known after apply)

+ protocol = "tcp"

+ security\_group\_id = (known after apply)

+ self = false

+ source\_security\_group\_id = (known after apply)

+ to\_port = 22

+ type = "ingress"

}

**# aws\_subnet.us-east-1a-testkop-farhad-com** will be created

+ resource "aws\_subnet" "us-east-1a-testkop-farhad-com" {

+ arn = (known after apply)

+ assign\_ipv6\_address\_on\_creation = false

+ availability\_zone = "us-east-1a"

+ availability\_zone\_id = (known after apply)

+ cidr\_block = "172.20.32.0/19"

+ id = (known after apply)

+ ipv6\_cidr\_block\_association\_id = (known after apply)

+ map\_public\_ip\_on\_launch = false

+ owner\_id = (known after apply)

+ tags = {

+ "KubernetesCluster" = "testkop.farhad.com"

+ "Name" = "us-east-1a.testkop.farhad.com"

+ "SubnetType" = "Public"

+ "kubernetes.io/cluster/testkop.farhad.com" = "owned"

+ "kubernetes.io/role/elb" = "1"

}

+ vpc\_id = (known after apply)

}

**# aws\_vpc.testkop-farhad-com** will be created

+ resource "aws\_vpc" "testkop-farhad-com" {

+ arn = (known after apply)

+ assign\_generated\_ipv6\_cidr\_block = false

+ cidr\_block = "172.20.0.0/16"

+ default\_network\_acl\_id = (known after apply)

+ default\_route\_table\_id = (known after apply)

+ default\_security\_group\_id = (known after apply)

+ dhcp\_options\_id = (known after apply)

+ enable\_classiclink = (known after apply)

+ enable\_classiclink\_dns\_support = (known after apply)

+ enable\_dns\_hostnames = true

+ enable\_dns\_support = true

+ id = (known after apply)

+ instance\_tenancy = "default"

+ ipv6\_association\_id = (known after apply)

+ ipv6\_cidr\_block = (known after apply)

+ main\_route\_table\_id = (known after apply)

+ owner\_id = (known after apply)

+ tags = {

+ "KubernetesCluster" = "testkop.farhad.com"

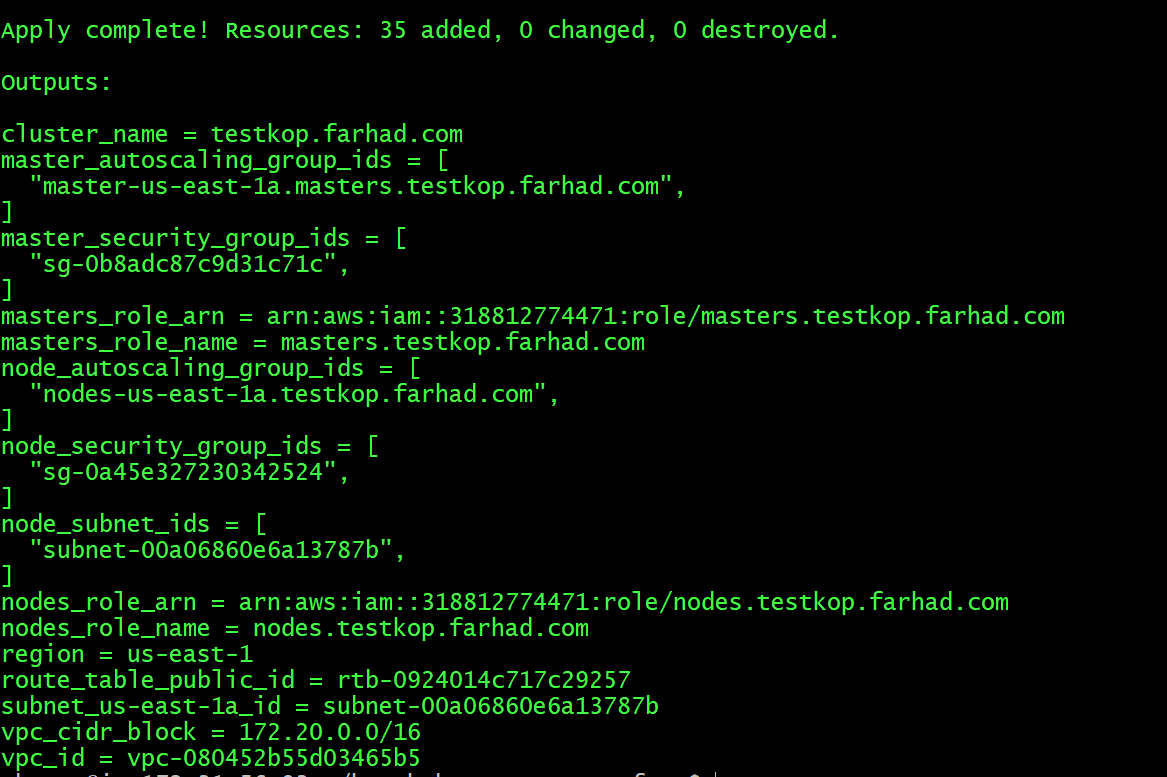
+ "Name" = "testkop.farhad.com"

+ "kubernetes.io/cluster/testkop.farhad.com" = "owned"

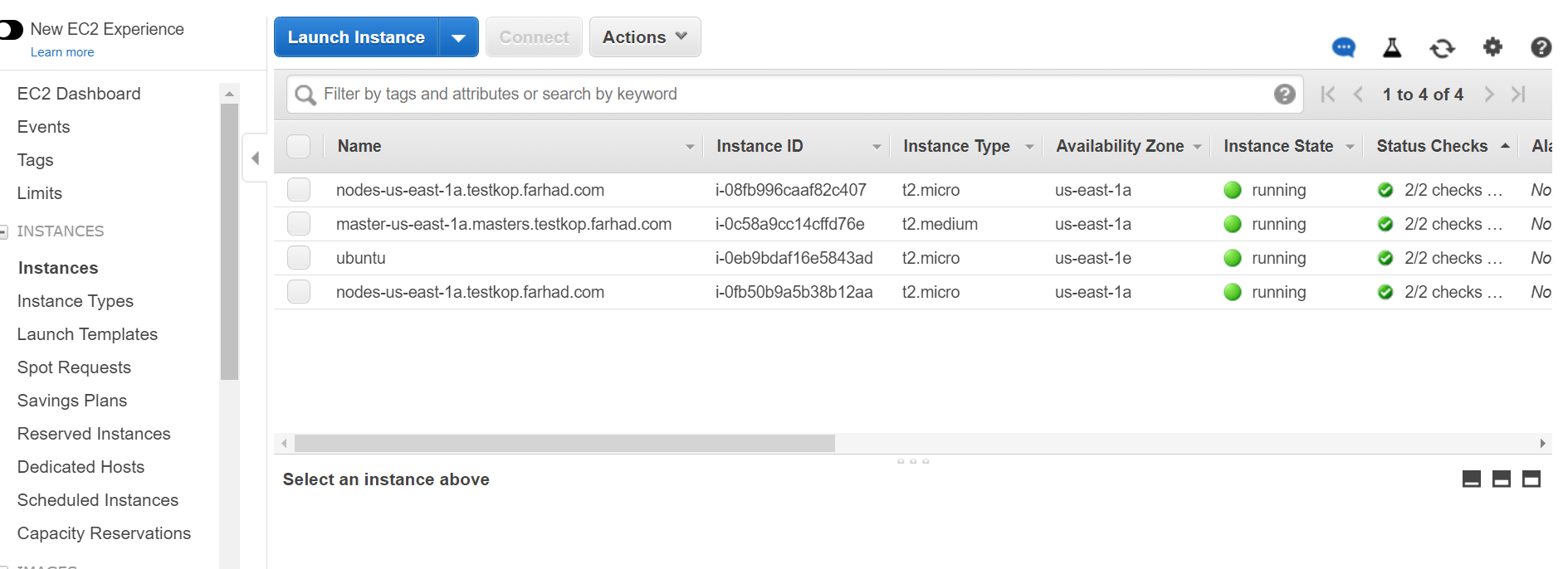
}

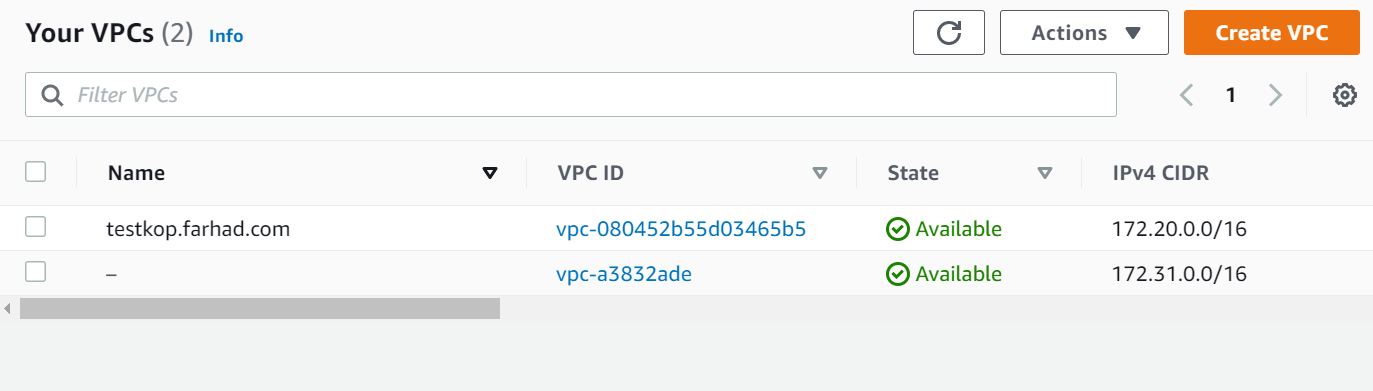
}

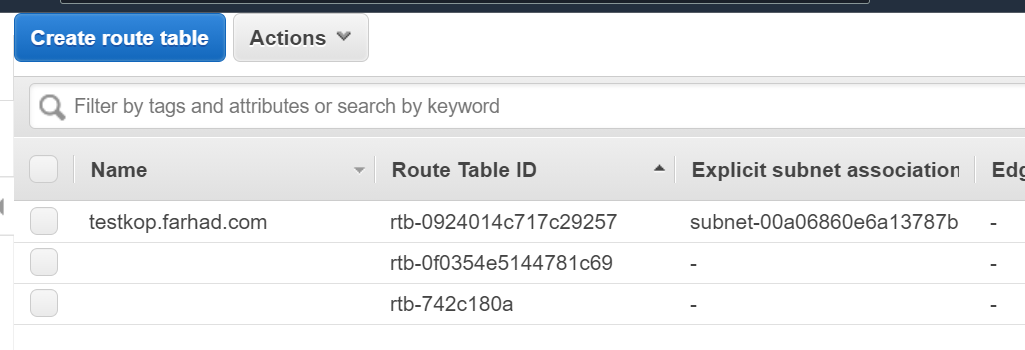
**I created VPC, subnets internet gateway , security groups and Ec2 instances from terraform**



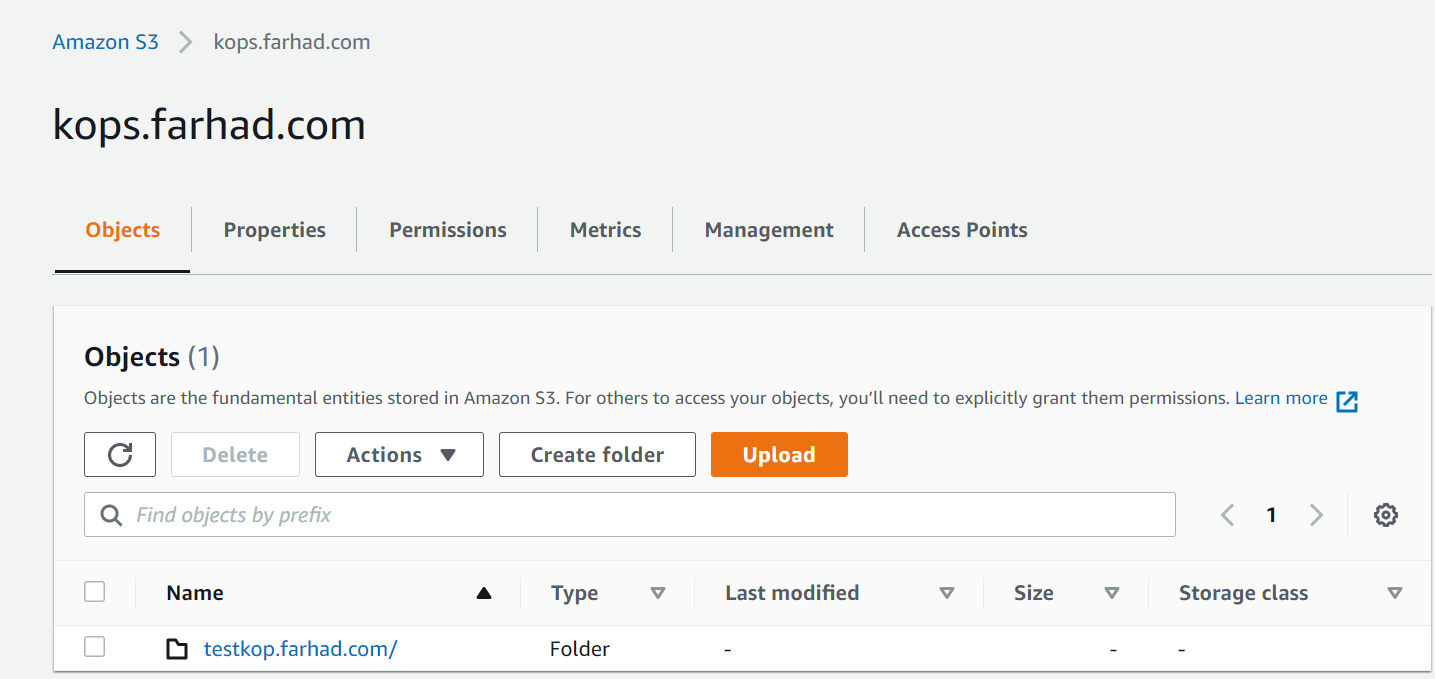
**I could create Kubernetes cluster as shown below**

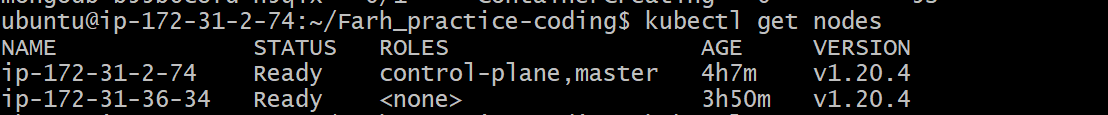






**I saved the state.tf file in a S3 bucket as shown below**





**Continuous deployment**

**Once the image is pushed from github to Docker hub through continuous integration , Helm will pull the image and deploy the application**

**I used Kubernetes to deploy nodejs and mongodb app using helm .**

**Below are the steps to be followed**

**Installing helm 3**

curl -fsSL -o get\_helm.sh <https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3>

chmod 700 get\_helm.sh

./get\_helm.sh

helm version

version.BuildInfo{Version:"v3.5.2", GitCommit:"167aac70832d3a384f65f9745335e9fb40169dc2", GitTreeState:"dirty", GoVersion:"go1.15.7

**Database not connected error fixed**

When I deployed the application with mongodb://localhost/tododbdev in database config , The application did not connect to the DB. Hence I changed the localhost to mongodb which fixed the errors

**Before**

module.exports = {

mongoURI: 'mongodb://localhost/tododb-dev'

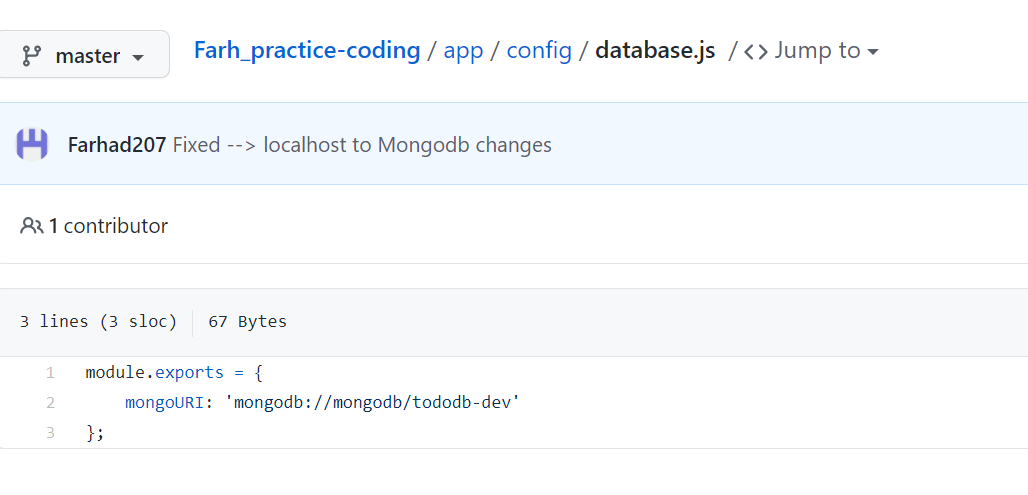
};

**After fixing**

module.exports = {

mongoURI: 'mongodb://**mongodb**/tododb-dev'

};



**Deploy application using helm**

**I have created helm charts in the folder nodejs.**

git clone <https://github.com/Farhad207/Farh_practice-coding.git>

:~/Farh\_practice-coding$ ls -lrt

total 12

drwxrwxr-x 8 ubuntu ubuntu 4096 Feb 28 18:47 app

-rw-rw-r-- 1 ubuntu ubuntu 22 Feb 28 18:47 README.md

drwxrwxr-x 3 ubuntu ubuntu 4096 Feb 28 18:50 **nodejs**

Farh\_practice-coding/nodejs# vi values.yaml

# Default values for nodejs.

# This is a YAML-formatted file.

# Declare variables to be passed into your templates.

replicaCount: 1

image:

repository: **farhad207/nodejs-mongodb**

# Overrides the image tag whose default is the chart appVersion.

tag: "latest"

port:

container: 5000

service:

type: NodePort

port: 5000

mongopv:

storageclassname: manual

hostpath: "/mnt/mongo\_data"

provisioner:

storageclassname: manual

storageclasspath: k8s.io/minikube-hostpath

mongodeploy:

imagename: mongo

mongosvc:

type: NodePort

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$ **helm install farhadnodejsapp nodejs/**

NAME: farhadnodejsapp

LAST DEPLOYED: Sun Feb 28 18:58:05 2021

NAMESPACE: default

STATUS: deployed

REVISION: 1

NOTES:

1. Get the application URL by running these commands:

export NODE\_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services farhadnodejsapp)

export NODE\_IP=$(kubectl get nodes --namespace default -o jsonpath="{.items[0].status.addresses[0].address}")

echo http://$NODE\_IP:$NODE\_PORT

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$ **kubectl get po**

NAME READY STATUS RESTARTS AGE

app-7c54c5d78d-nmbz9 1/1 Running 0 38s

mongodb-b55b6c8fd-f2qkt 1/1 Running 0 38s

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$ **kubectl get svc**

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

app NodePort 10.111.92.87 <none> 5000:31110/TCP 48s

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 3h51m

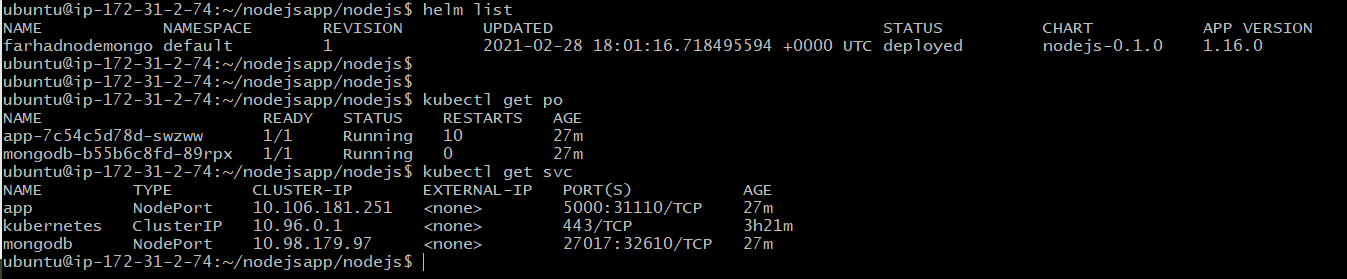
mongodb NodePort 10.101.206.191 <none> 27017:30655/TCP 48s

ubuntu@ip-172-31-2-74:~/Farh\_practice-coding$ **helm list**

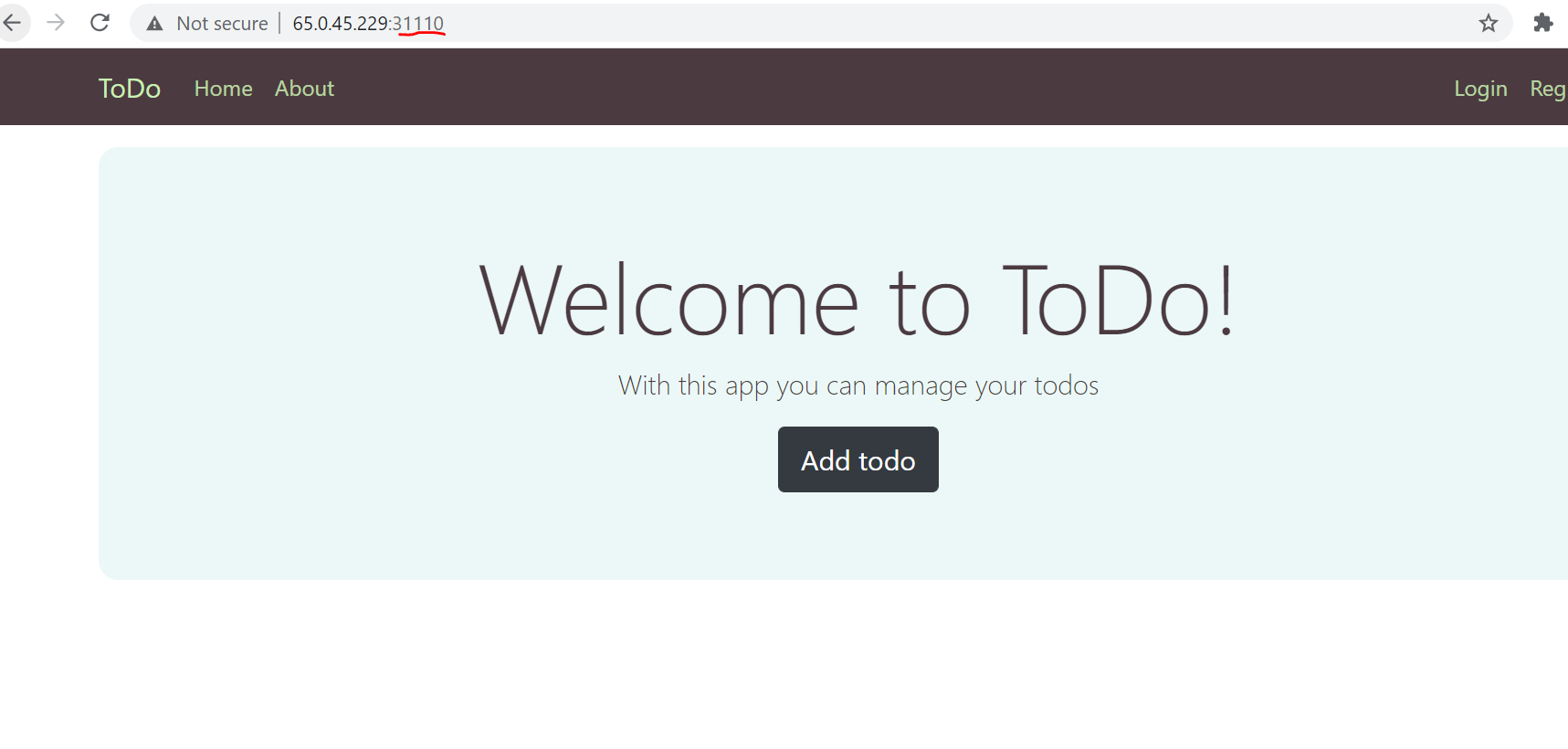
NAME NAMESPACE REVISION UPDATED STATUS CHART APP VERSION

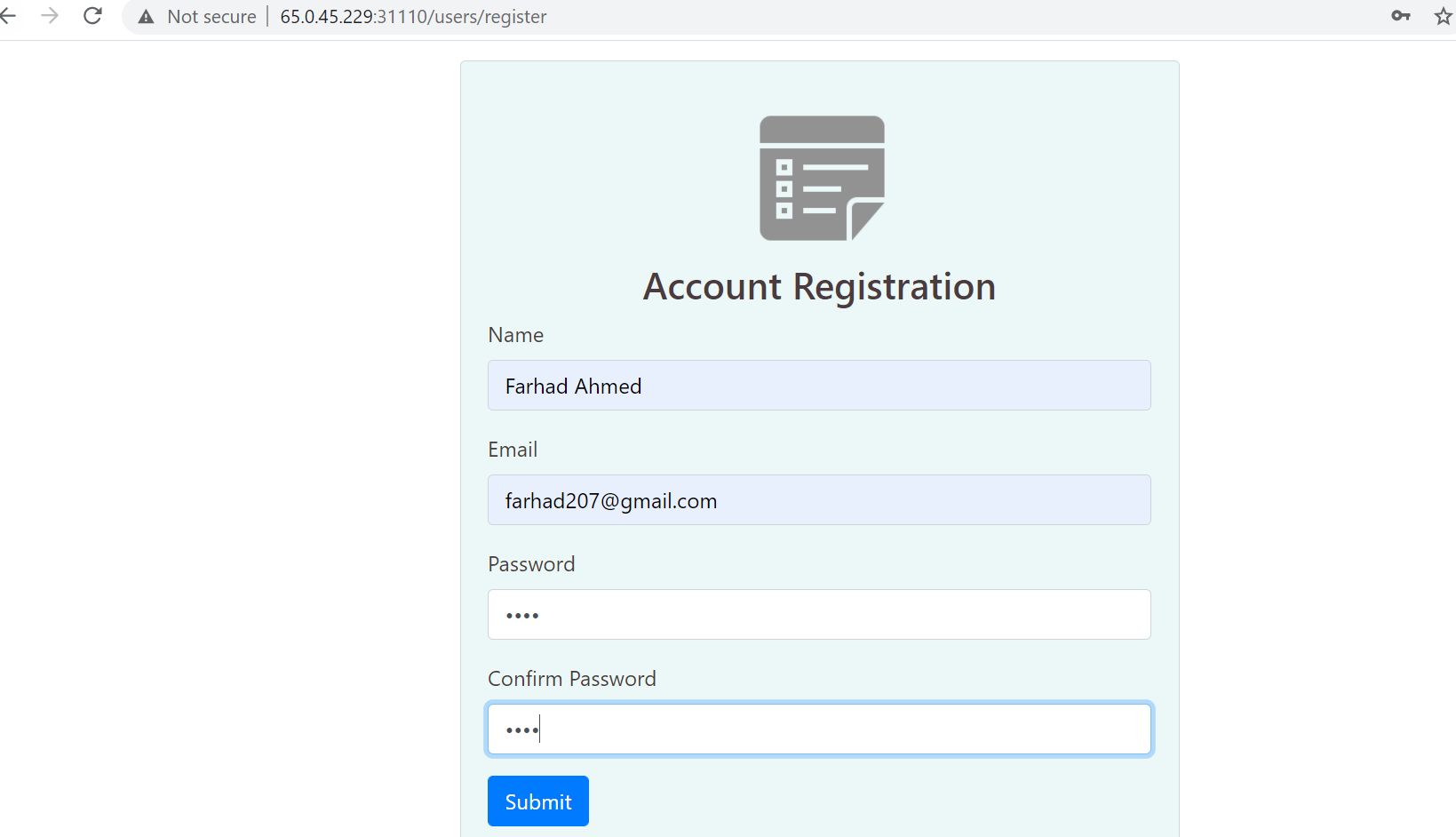
farhadnodejsapp default 1 2021-02-28 18:58:05.653730659 +0000 UTC deployed nodejs-0.1.0 1.16.0

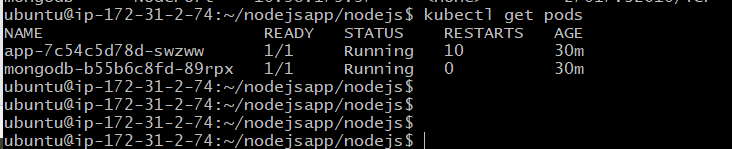


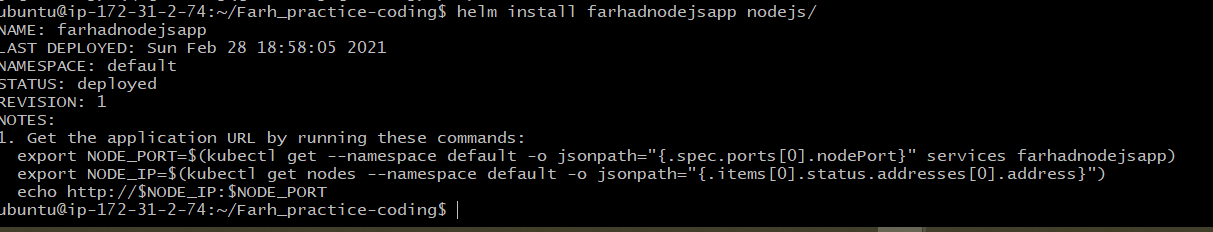


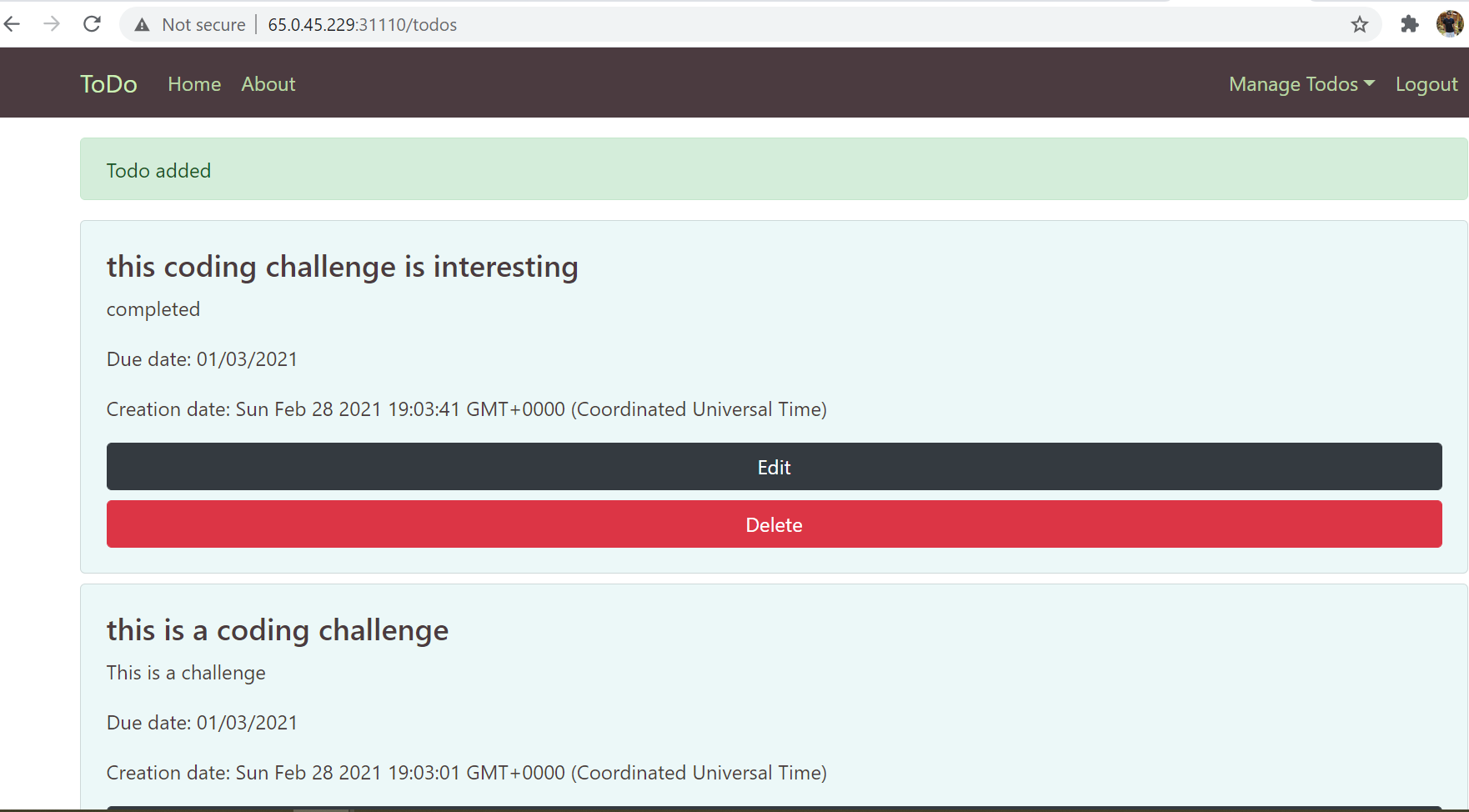
The app is exposed on 31110 port











Thank you , Happy coding !!