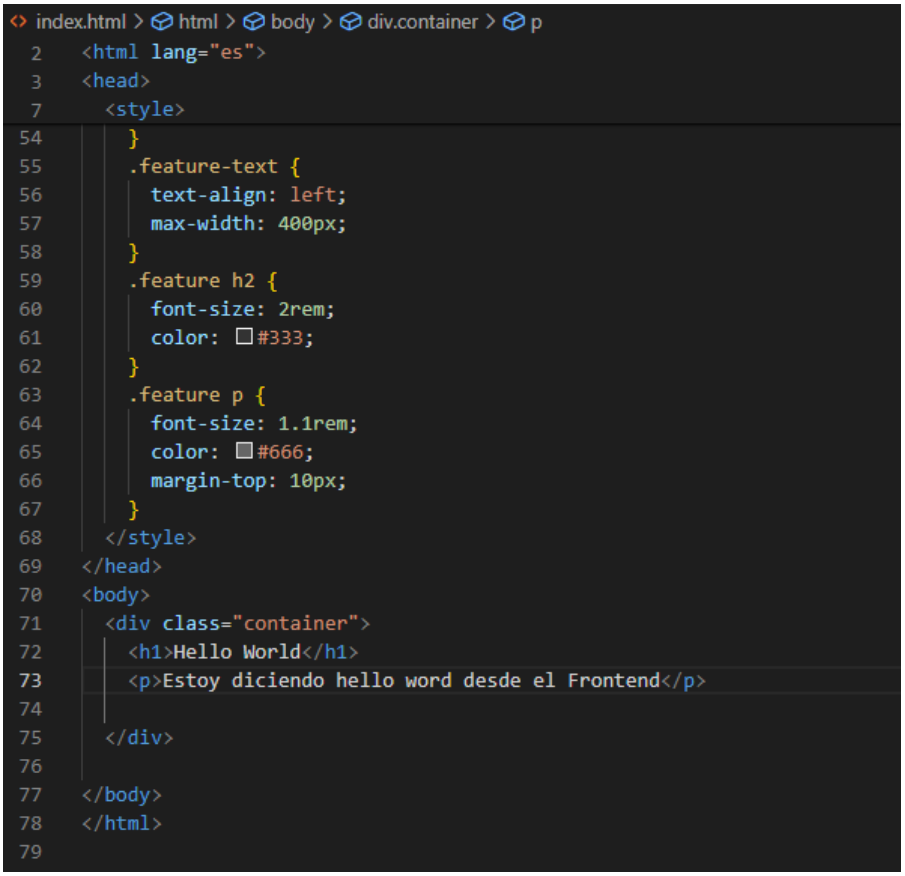
JASON BRIAN GIRÓN ZAMORA – 1135220

PARCIAL 2

Primero se creo el código del frontend y backend para que dijeran **hello world.** En el caso de mongoDB no fue necesario.

Frontend (Código)



Backend (Código)

```
const express = require('express');
const app = express();

// Endpoint para saludar
app.get('/hello', (req, res) => {
  res.send('Hello, World!');
});

// Puerto en el que escuchará el servidor
const PORT = process.env.PORT || 3000;

// Iniciar el servidor
app.listen(PORT, () => {
  console.log(`Servidor escuchando en el puerto ${PORT}`);
});
```

Seguido se creo el dockerfile para cada una, esto para poder subir el código como imágenes a Dockerhub ya que desde allí vamos a mandar a llamar las imágenes, en el caso de mongoDB no fue necesario, pues se uso el ya publicado en Dockerhub

Frontend (Dockerfile):

```
Dockerfile > ...
1   #servidor ngnix
2   FROM nginx:alpine
3
4   # Copia el contenido al directorio de trabajo de nginx
5   COPY index.html /usr/share/nginx/html/index.html
6
7   # Expón el puerto en el que la aplicación se ejecuta
8   EXPOSE 5000
```

Backend (Dockerfile):

Backend > Dockerfile > ...

```dockerfile
# Usa la imagen oficial de Node.js como base
FROM node:18.13.0

# Establece el directorio de trabajo en la carpeta de la aplicación
WORKDIR /app

# Copia los archivos del proyecto al contenedor
COPY package.json ./
COPY app.js ./

# Instala las dependencias
RUN npm install

# Expón el puerto en el que la aplicación se ejecuta
EXPOSE 3000

# Comando para ejecutar la aplicación
CMD ["node", "app.js"]
```

Al tener esto se procedió a crear las imágenes y subirlas a Dockerhub, para esto se utilizaron los comandos:

Frontend:

docker build -t jasongiron/front:latest .

docker run -d -p 5000:5000 jasongiron/ front:latest

docker push jasongiron/ front:latest

```
 => => naming to docker.io/jasongiron/front:latest                                                    0.0s

View build details: docker-desktop://dashboard/build/default/default/kfwa3049ksqwgplgba1tt1va8

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>docker run -d -p 5000:5000 jasongiron/ f
ront:latest
docker: invalid reference format.
See 'docker run --help'.

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>docker run -d -p 5000:5000 jasongiron/fr
ont:latest
253d8060111d252487a6e4dea578f4d143e723c511a5e7339bd79ac37c207957

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>docker push jasongiron/front:latest
The push refers to repository [docker.io/jasongiron/front]
dffaf768ea93: Pushed
ce495f7b0b7d: Mounted from jasongiron/web
9c70f446fbe2: Mounted from jasongiron/web
5be225e16e44: Mounted from jasongiron/web
3d04ead9b400: Mounted from jasongiron/web
af5598fef05f: Mounted from jasongiron/web
8fbd5a835e5e: Mounted from jasongiron/web
75061be64847: Mounted from jasongiron/web
d4fc045c9e3a: Mounted from jasongiron/web
latest: digest: sha256:bd23990e791cc21b5c0b1095d8759fa937343fde895898071a4ae0d07f1f3a9d size: 2196

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>
```

Backend:

docker build -t jasongiron/back:latest .

docker run -d -p 3037:3037 jasongiron/ back:latest

docker push jasongiron/back:latest

```
 => CACHED [5/5] RUN npm install                                                                      0.0s
 => exporting to image                                                                                0.1s
 => => exporting layers                                                                               0.0s
 => => writing image sha256:fb4d24a88ed25796fbe9b56b960c8686b80e927d0a998d7718ab62ca4e349ec2          0.0s
 => => naming to docker.io/jasongiron/back:latest                                                     0.0s

View build details: docker-desktop://dashboard/build/default/default/7gijt2auy77vwl9baaatm160n

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>docker run -d -p 3000:3000 jason
giron/ back:latest
docker: invalid reference format.
See 'docker run --help'.

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>docker run -d -p 3000:3000 jason
giron/back:latest
aae66ff6b673a5f863ca5b3be4523cd7a662ddf06c1b5d3a8829f86e3bd11c93

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>docker push jasongiron/ back:lat
est
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage:  docker push [OPTIONS] NAME[:TAG]

Upload an image to a registry

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>
```

Ahora creamos nuestros archivos Terraform main.tf para poder

Para esto desarrollamos tanto el deployment como el service de cada uno

Frontend

```
# Deployment para el frontend
resource "kubernetes_deployment" "front" {
  metadata {
    name = "front"
  }

  spec {
    replicas = 1

    selector {
      match_labels = {
        app = "front"
      }
    }

    template {
      metadata {
        labels = {
          app = "front"
        }
      }

      spec {
        container {
          image = "jasongiron/front:latest"
          name  = "front"
          port {
            container_port = 5000
          }
        }
      }
    }
```

```
# Service para el front
resource "kubernetes_service" "front-service" {
  metadata {
    name = "front-service"
  }

  spec {
    selector = {
      app = "front"
    }

    port {
      port        = 5000
      target_port = 5000
    }
  }
}
```

Backend

```
# Deployment backend
resource "kubernetes_deployment" "back" {
  metadata {
    name = "back"
  }

  spec {
    replicas = 1

    selector {
      match_labels = {
        app = "back"
      }
    }

    template {
      metadata {
        labels = {
          app = "back"
        }
      }

      spec {
        container {
          image = "jasongiron/back:latest"
          name  = "back"
          port {
            container_port = 3037
          }
        }
      }
    }
```

```
# Service para el back
resource "kubernetes_service" "back-service" {
  metadata {
    name = "back-service"
  }

  spec {
    selector = {
      app = "back"
    }

    port {
      port        = 3037
      target_port = 3037
    }
  }
}
```

MongoDB

```
# Deployment MongoDB
resource "kubernetes_deployment" "mongo" {
  metadata {
    name = "mongo"
  }

  spec {
    replicas = 1

    selector {
      match_labels = {
        app = "mongo"
      }
    }

    template {
      metadata {
        labels = {
          app = "mongo"
        }
      }

      spec {
        container {
          image = "mongo:latest"
          name  = "mongo"
          port {
            container_port = 27017
          }
        }
      }
    }
```

```
# Service para MongoDB
resource "kubernetes_service" "mongo-service" {
  metadata {
    name = "mongo-service"
  }

  spec {
    selector = {
      app = "mongo"
    }

    port {
      port        = 27017
      target_port = 27017
    }
  }
}
```

El cluster que utilizaré es minikube corriendo con la máquina virtual de Docker.

Para esto iniciamos el cluster con el comando

minikube start

```
C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>minikube start
W0514 20:42:56.147679   14452 main.go:291] Unable to resolve the current Docker CLI context "default": context "default"
: context not found: open C:\Users\jbgir\.docker\contexts\meta\37a8eec1ce19687d132fe29051dca629d164e2c4958ba141d5f4133a3
3f0688f\meta.json: El sistema no puede encontrar la ruta especificada.
* minikube v1.33.1 en Microsoft Windows 10 Home Single Language 10.0.19045.4412 Build 19045.4412
! Ambos driver=docker y vm-driver=virtualbox han sido establecidos.

 vm-driver ya es obsoleto, el por defecto de minikube será driver=docker.

 Si vm-driver está establecido en la configuracion global, ejecuta "minikube config unset vm-driver" para resolver esta
advertencia.

* Using the docker driver based on user configuration
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.44 ...
* Creating docker container (CPUs=2, Memory=2700MB) ...
* Preparando Kubernetes v1.30.0 en Docker 26.1.1...
  - Generando certificados y llaves
  - Iniciando plano de control
  - Configurando reglas RBAC...
* Configurando CNI bridge CNI ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Complementos habilitados: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>
```

Luego en la ruta en donde tenemos el archivo main.tf

Ingresamos los comandos:

Terraform init

Terraform apply

```
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial\Backend>cd C:\Users\jbgir\Documents\Docs
\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/kubernetes...
- Installing hashicorp/kubernetes v2.30.0...
- Installed hashicorp/kubernetes v2.30.0 (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.

C:\Users\jbgir\Documents\Docs\INGENIERIA_INFORMATICA\2024\Vitualización\Parcial>
```

```
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

kubernetes_service.front-service: Creating...
kubernetes_service.back-service: Creating...
kubernetes_service.mongo-service: Creating...
kubernetes_deployment.back: Creating...
kubernetes_deployment.mongo: Creating...
kubernetes_deployment.front: Creating...
kubernetes_service.front-service: Creation complete after 0s [id=default/front-service]
kubernetes_service.back-service: Creation complete after 0s [id=default/back-service]
kubernetes_service.mongo-service: Creation complete after 0s [id=default/mongo-service]
kubernetes_deployment.mongo: Still creating... [10s elapsed]
kubernetes_deployment.front: Still creating... [10s elapsed]
kubernetes_deployment.back: Still creating... [10s elapsed]
kubernetes_deployment.front: Still creating... [20s elapsed]
kubernetes_deployment.back: Still creating... [20s elapsed]
kubernetes_deployment.mongo: Still creating... [20s elapsed]
kubernetes_deployment.back: Still creating... [30s elapsed]
kubernetes_deployment.front: Still creating... [30s elapsed]
kubernetes_deployment.mongo: Still creating... [30s elapsed]
kubernetes_deployment.front: Still creating... [40s elapsed]
kubernetes_deployment.back: Still creating... [40s elapsed]
kubernetes_deployment.mongo: Still creating... [40s elapsed]
kubernetes_deployment.mongo: Still creating... [50s elapsed]
kubernetes_deployment.back: Still creating... [50s elapsed]
kubernetes_deployment.front: Still creating... [50s elapsed]
kubernetes_deployment.mongo: Still creating... [1m0s elapsed]
```

Esto levantará el código y será visible el front, backend y mongodb

Frontend

Backend