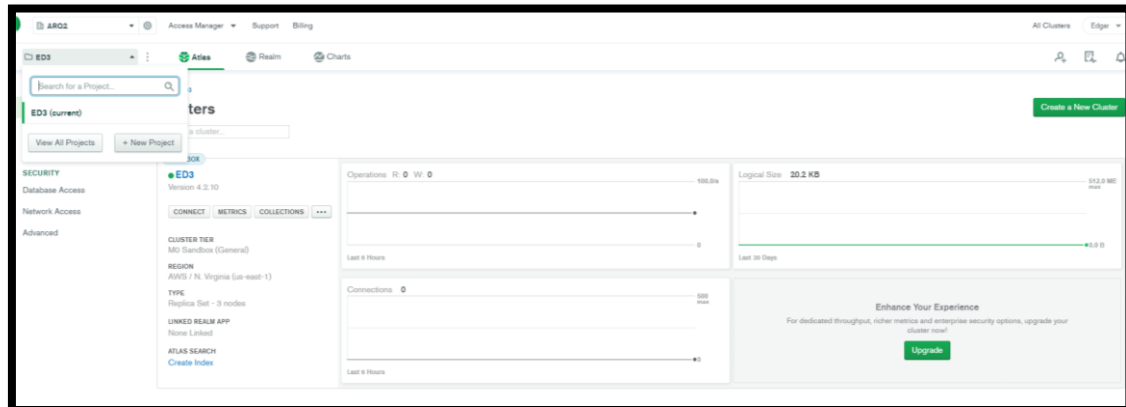


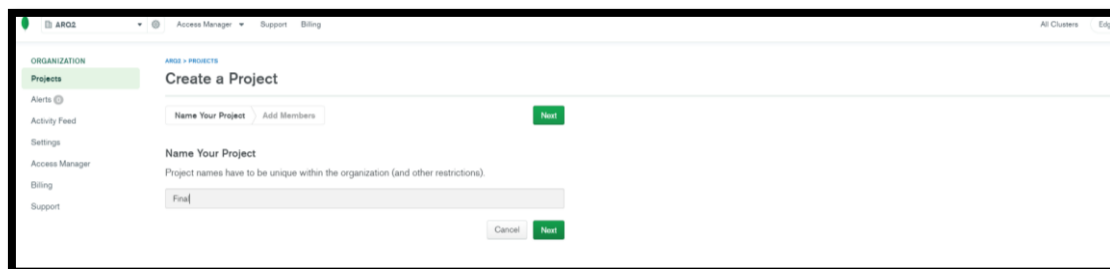
# Manual de configuraciones

## Creación y configuración de la base de datos

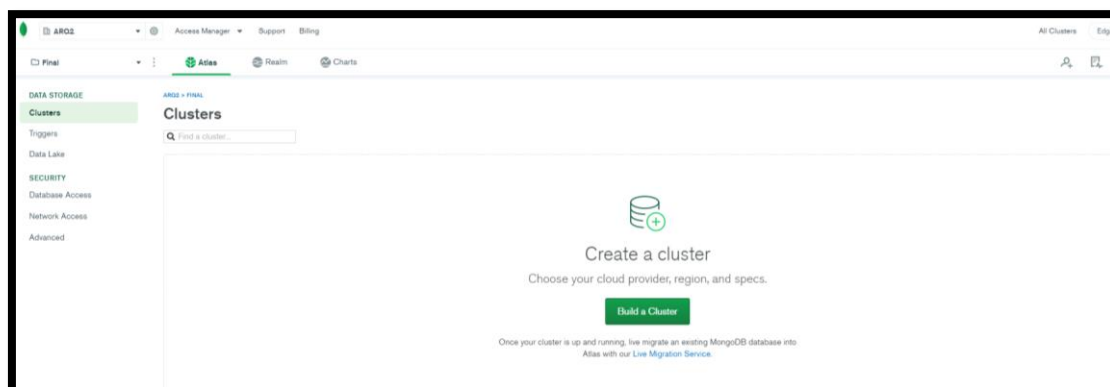
Se crea un nuevo proyecto.



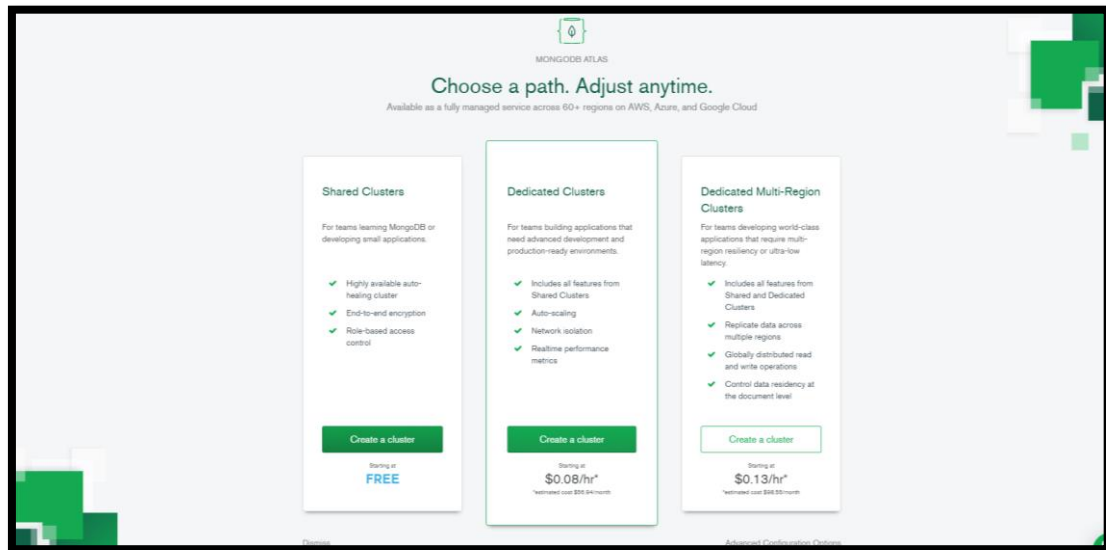
Se asigna el nombre, luego next y luego crear.



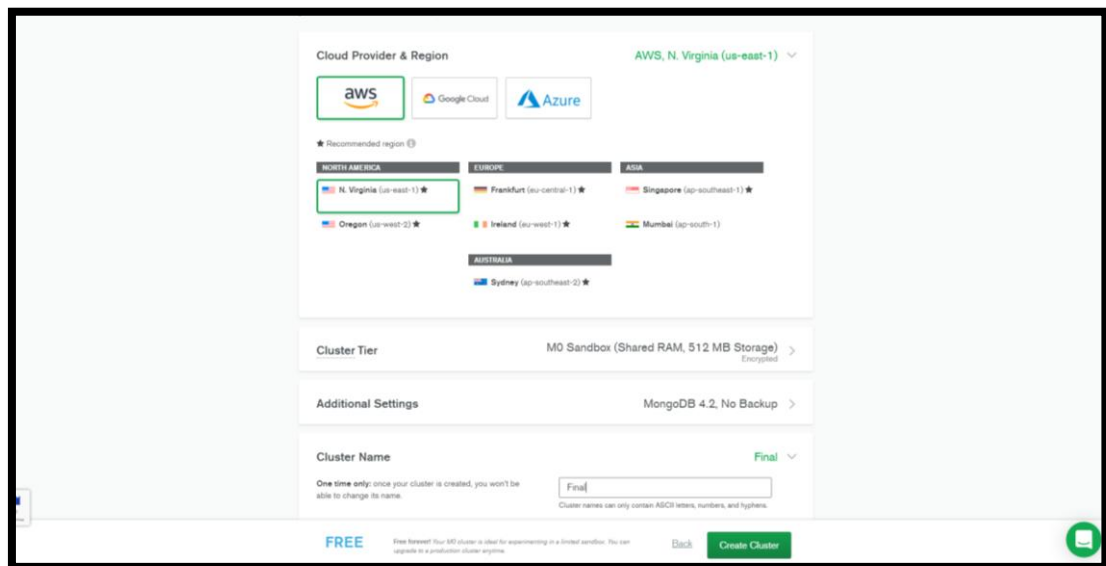
Se crea un nuevo cluster.



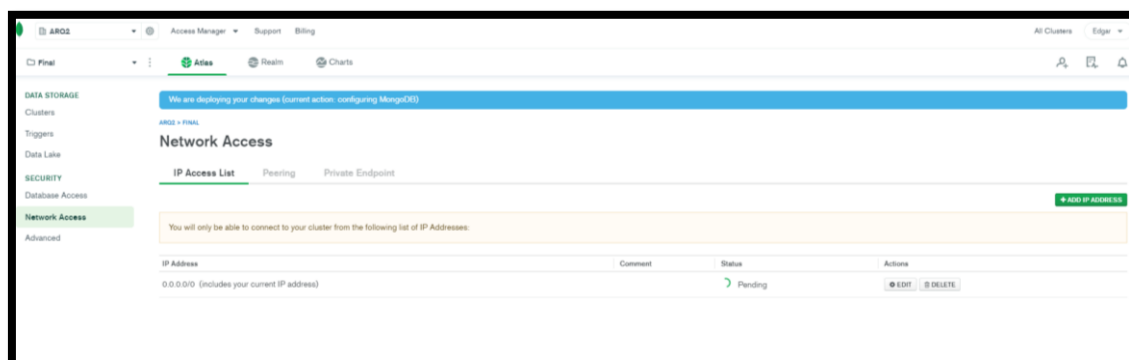
Se selecciona la opción que se desee, en este caso usaremos la opción gratuita.



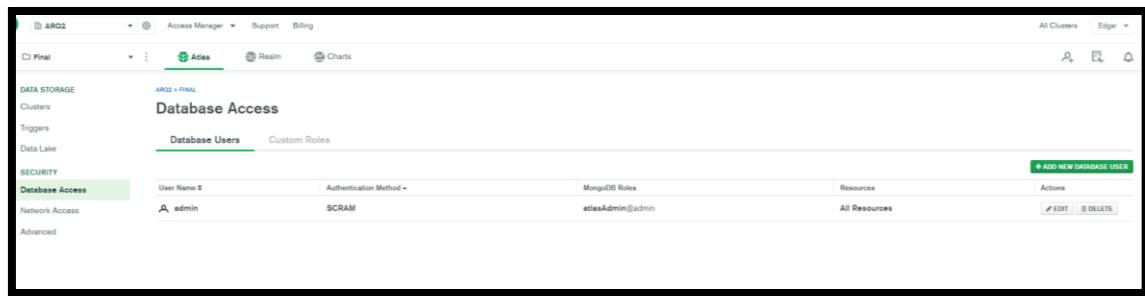
Se selecciona el proveedor, se asigna el nombre y clic en crear cluster.



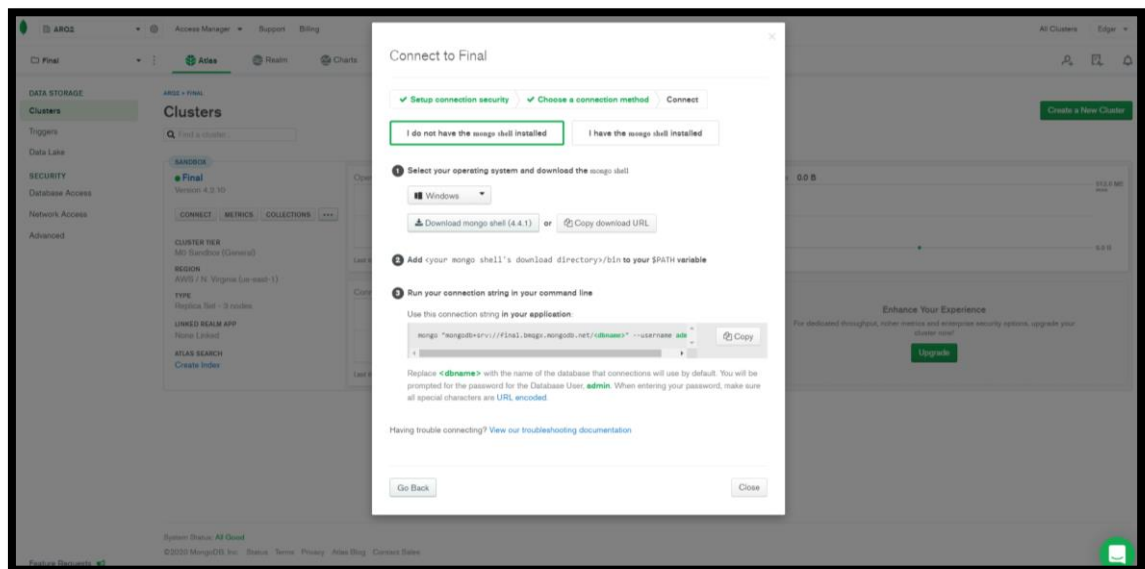
Se agrega un acceso una dirección ip.



Se crea un usuario con permisos de administrador.



Nos dirigimos a clusters y luego a connect donde se desplegarán las opciones de conexión, se selecciona la deseada. En este caso tenemos la siguiente url: `mongodb+srv://admin:1234@final.bmqgx.mongodb.net/Final?retryWrites=true&w=majority`



## Frontend

Se desarrollo una pagina web estática con html5, css y javascript.

### Registro

#### Crear Usuario

Usuario

Contraseña

Repetir Contraseña

Elegir archivo

No se ha seleccionado ningún archivo

Registrar

Iniciar Sesion

### Login

#### Inicias sesion

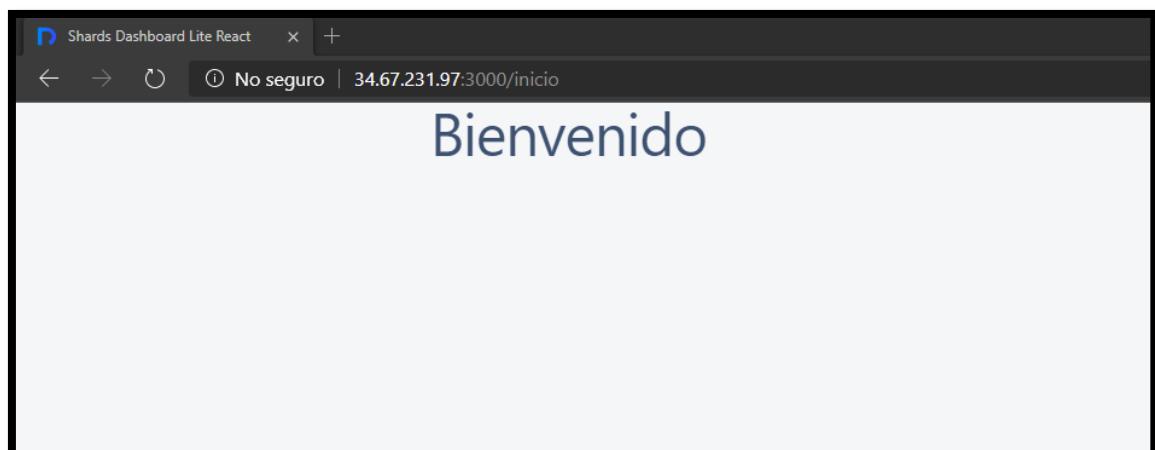
Usuario

Contraseña

Iniciar Sesion

Crear Cuenta

### Inicio



## Backend

Aplicación en nodejs con las siguientes rutas

35.188.121.170:5000/Login: utilizada para loguear al usuario en la plataforma.

35.188.121.170:5000/registro: Utilizada para el registro de usuarios.

## Kubernetes

ymls

Backend

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: back-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: back
  template:
    metadata:
      labels:
        app: back
    spec:
      containers:
        - name: server-back
          image: gcr.io/sopes-295304/back:1.0
          ports:
            - containerPort: 5000
```

Frontend

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: front-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: front
  template:
```

```
metadata:
  labels:
    app: front
spec:
  containers:
  - name: server-front
    image: gcr.io/sopes-295304/front2:1.0
    ports:
    - containerPort: 3000
```

Balanceador de carga backend

```
apiVersion: v1
kind: Service
metadata:
  name: back-service
spec:
  selector:
    app: back
  ports:
  - port: 5000
    targetPort: 5000
  type: LoadBalancer
```

Balanceador de carga frontend

```
apiVersion: v1
kind: Service
metadata:
  name: front-service
spec:
  selector:
    app: front
  ports:
  - port: 3000
    targetPort: 3000
  type: LoadBalancer
```

## Creación Pods y balanceador de carga

Para iniciar con la creación de los pods es necesario primero construir las imágenes que se utilizaran en los pod, para eso se utiliza el siguiente comando

Se le asigna el nombre y versión deseada.

Para el frontend se aplican los mismos comandos.

```
arnolso201@cloudshell:~/S01-Final (sopes-295304) $ docker build -t gcr.io/sopes-295304/back:1.0 .
```

Luego de construir las plantillas se les hace push para posteriormente utilizar la plantilla.

```
arnolso201@cloudshell:~/S01-Final (sopes-295304) $ docker push gcr.io/sopes-295304/back:1.0
```

Con las plantillas listas solamente queda ejecutar los yml que se encargaran de la construcción de los pods y los balanceadores de carga. Se debe de realizar la construcción del frontend y backend antes que los balanceadores de carga. Para la construcción de los yml se utiliza el siguiente comando

```
arnolso201@cloudshell:~/S01-Final (sopes-295304) $ kubectl apply -f deploymentBack.yml
```

Con los siguientes comandos es posible ver los pods y servicios que están corriendo.

```
arnolso201@cloudshell:~/S01-Final (sopes-295304)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
back-deployment-7d8459b59-czrws     1/1     Running   0           12s
back-deployment-7d8459b59-lzvrh     1/1     Running   0           12s
back-deployment-7d8459b59-nf5x2     1/1     Running   0           12s
front-deployment-567d5d465f-5m9tw   1/1     Running   0           41h
front-deployment-567d5d465f-kkxhz   1/1     Running   0           41h
front-deployment-567d5d465f-pfvn9   1/1     Running   0           41h
arnolso201@cloudshell:~/S01-Final (sopes-295304)$ kubectl get services
NAME      TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
back-service  LoadBalancer  10.4.6.104    35.188.121.170  5000:30129/TCP   42h
front-service  LoadBalancer  10.4.7.16     34.67.231.97   3000:31799/TCP   42h
kubernetes   ClusterIP      10.4.0.1      <none>         443/TCP           4d16h
arnolso201@cloudshell:~/S01-Final (sopes-295304)$
```