Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Sistemas Operativos 1
Guillermo Alfredo Peitzner Estrada - 201504468.

Manual de Configuraciones

Node.JS API REST

DynamoDB

```
const AWS = require("aws-sdk");
const DDB = new AWS.DynamoDB({
  accessKeyId: "AKIA5M3XKPWM4P24F63P",
  secretAccessKey: "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",
  region: "us-east-2",
});

function insert(params) {
  return new Promise((resolve, reject) => {
    DDB.putItem(params, (err, data) => {
      if (err) {
        reject(err);
      } else {
        resolve(data);
      }
    });
  });
}

function getAll(params) {
  return new Promise((resolve, reject) => {
    DDB.scan(params, (err, data) => {
      if (err) {
        reject(err);
      } else {
        resolve(data);
      }
    });
  });
}
```

JWT

```
function authenticateToken(req, res, next) {
  const authHeader = req.headers["authorization"];
  const token = authHeader && authHeader.split(" ")[1];
  if (token == null) return res.sendStatus(401);
  jwt.verify(token, TOKEN_SECRET, (err, user) => {
    if (err) return res.sendStatus(403);
    req.user = user;
    next();
  });
}

function generateAccessToken(username) {
  return jwt.sign(username, TOKEN_SECRET, { expiresIn: "1800s" });
}
```

Express

```
const express = require("express");
const bodyParser = require("body-parser");
const cors = require("cors");
const app = new express();
app.use(bodyParser.json());
app.use(cors());
app.get("/", authenticateToken, async (req, res) => {
  try {
    const params = {
      TableName: "FinalExam",
    };
    const results = await getAll(params);
    res.json(
      results.Items.map((data) => {
        return {
          Id: data.Id.S,
          Text: data.Text.S,
        };
      })
    );
  } catch (error) {
    res.json({ message: "error" });
  }
});
```

```
app.post("/", authenticateToken, async (req, res) => {
  try {
    const { text } = req.body;
    const params = {
      TableName: "FinalExam",
      Item: {
        Id: { S: Date.now().toString() },
        Text: { S: text },
      },
    };
    await insert(params);
    res.json(params);
  } catch (error) {
    res.json({ message: "error" });
  }
});

app.listen(8080);
```

Dockerfile

```
FROM node:latest
WORKDIR /code
COPY . .
RUN npm install
EXPOSE 8080
CMD [ "npm", "run", "start" ]
```

Deployment.yml

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

```
      labels:
        app: api
    spec:
      containers:
        - name: api-container
          image: gcr.io/peppy-aileron-292121/api:1.0
          ports:
            - containerPort: 8080
```

Service.yml

```
apiVersion: v1
kind: Service
metadata:
  name: api-service
spec:
  selector:
    app: api
  ports:
    - port: 8080
      targetPort: 8080
  type: LoadBalancer
```

Nginx + Angular.js

Dockerfile

```
FROM nginx
COPY ./dist/web /usr/share/nginx/html
```

Deployment.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web
```

```
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
        - name: web-container
          image: gcr.io/peppy-aileron-292121/web:1.0
          ports:
            - containerPort: 80
```

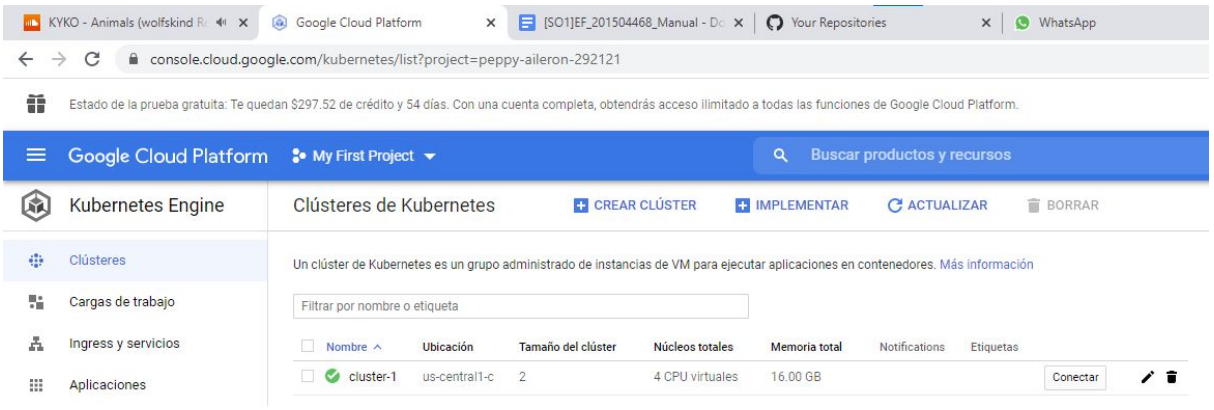
Service.yml

```
apiVersion: v1
kind: Service
metadata:
  name: web-service
spec:
  selector:
    app: web
  ports:
    - port: 80
      targetPort: 80
  type: LoadBalancer
```

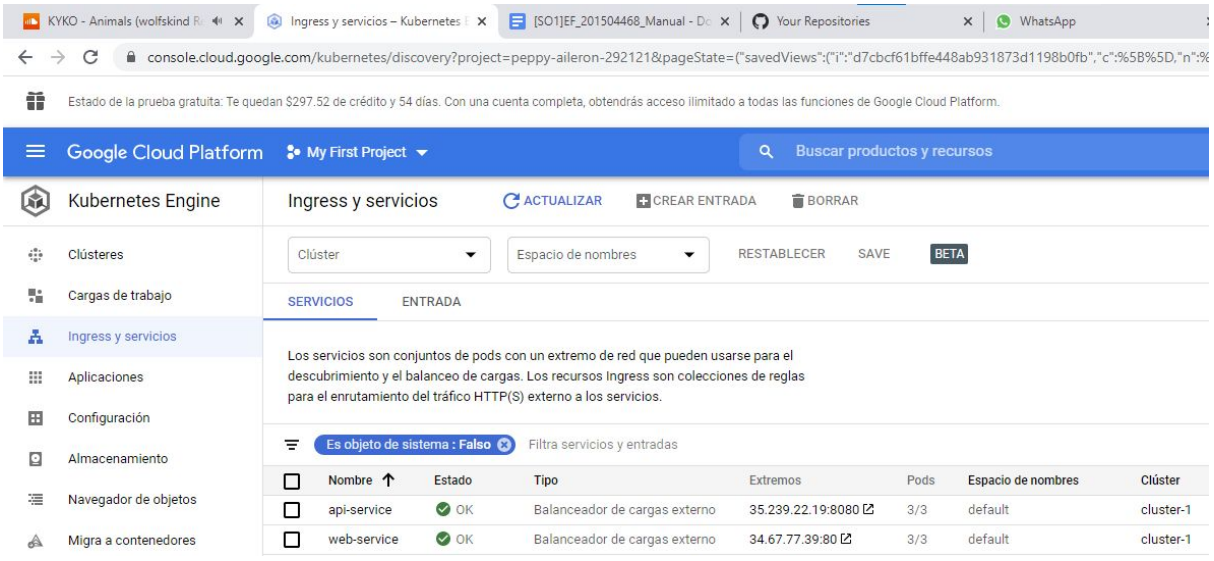
Google Kubernetes Engine

```
docker build -t gcr.io/$DEVSHELL_PROJECT_ID/api:1.0 .
gcloud auth configure-docker
docker push gcr.io/$DEVSHELL_PROJECT_ID/api:1.0
kubectl apply -f deployment.yml
kubectl apply -f service.yml

docker build -t gcr.io/$DEVSHELL_PROJECT_ID/web:1.0 .
gcloud auth configure-docker
docker push gcr.io/$DEVSHELL_PROJECT_ID/web:1.0
kubectl apply -f deployment.yml
kubectl apply -f service.yml

kubectl get deployments
kubectl get services
```

Cluster



Cargas de trabajo



Ingress y servicios