Aplicación con Microservicios

Raúl Francisco Ochoa Díaz

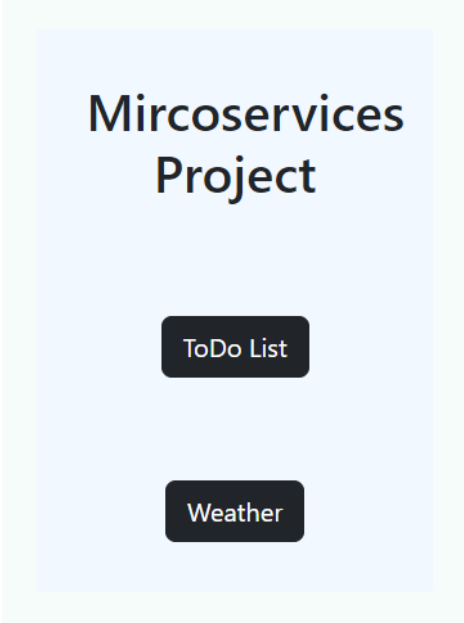
220790776

Dr. Michel Emmanuel López Franco

Computación Tolerante a Fallas D06

¿Cómo Funciona la Aplicación?

Una interfaz para acceder a los servicios.



Accesibles en cualquier momento.

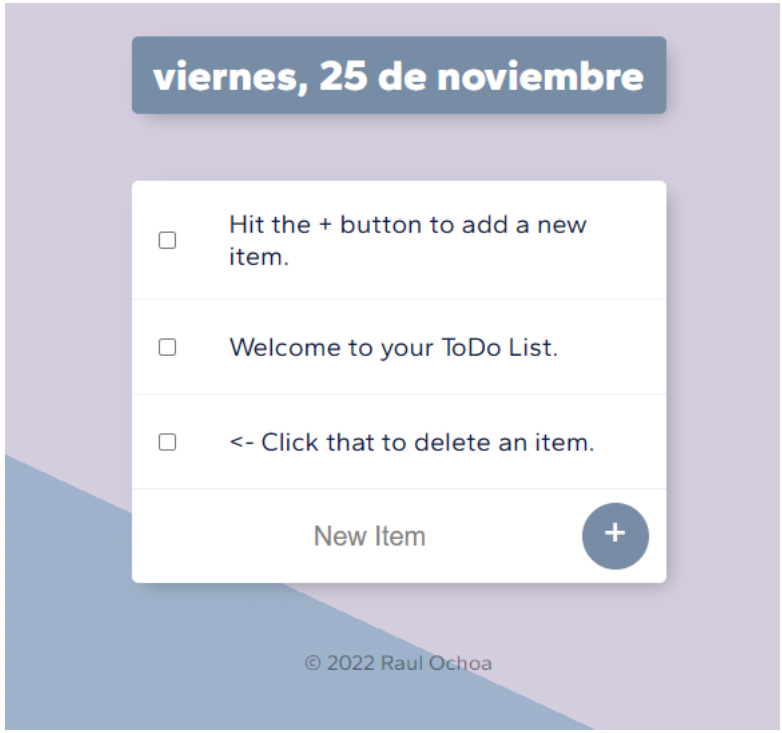
Cada uno alojado en un *pod* distinto.

Lista de Pendientes

Guarda las entradas en la base de datos remota de MongoDB.

La eliminación se realiza utilizando el *checkbox*.

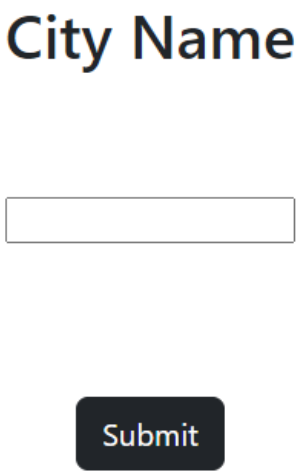
Es posible crear sublistas a través de las rutas.



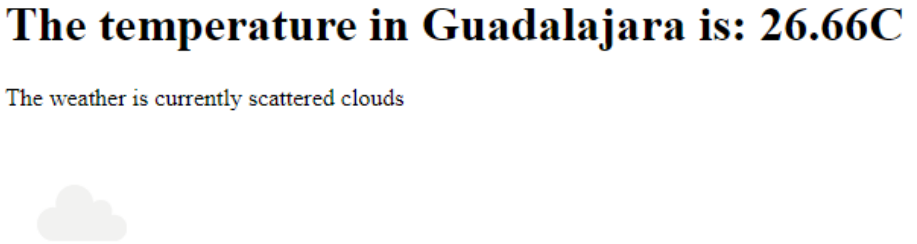
Programada con NodeJS y Express.

Aplicación del Clima

Solicita información del clima de una ciudad a una API.



Muestra los resultados en HTML junto a una pequeña imagen.



Proceso de Despliegue

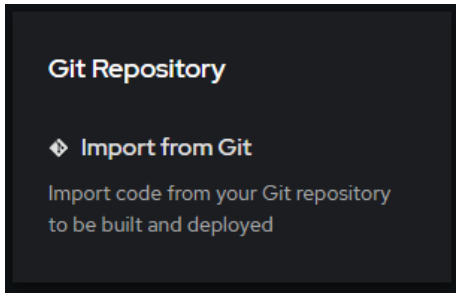
Cargar los tres programas a GitHub.

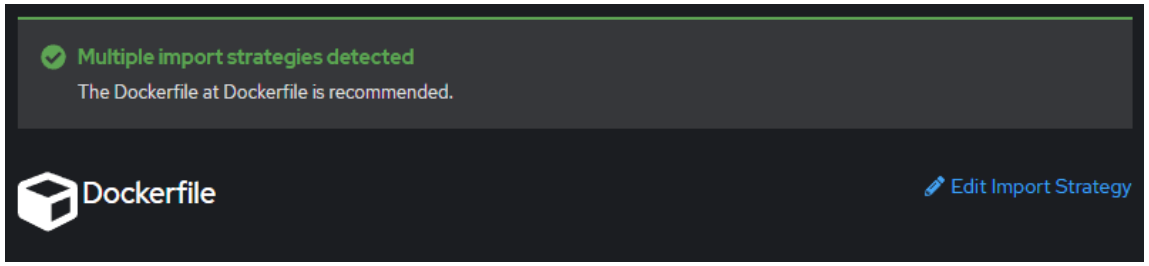
img	Agregar imagenes	7 hours ago
public	Add routes	14 days ago
src	Cambiar link clima	11 days ago
.gitignore	Initialize project using Create React App	15 days ago
README.md	Update README.md	7 hours ago
package-lock.json	Initialize project using Create React App	15 days ago
package.json	Initialize project using Create React App	15 days ago

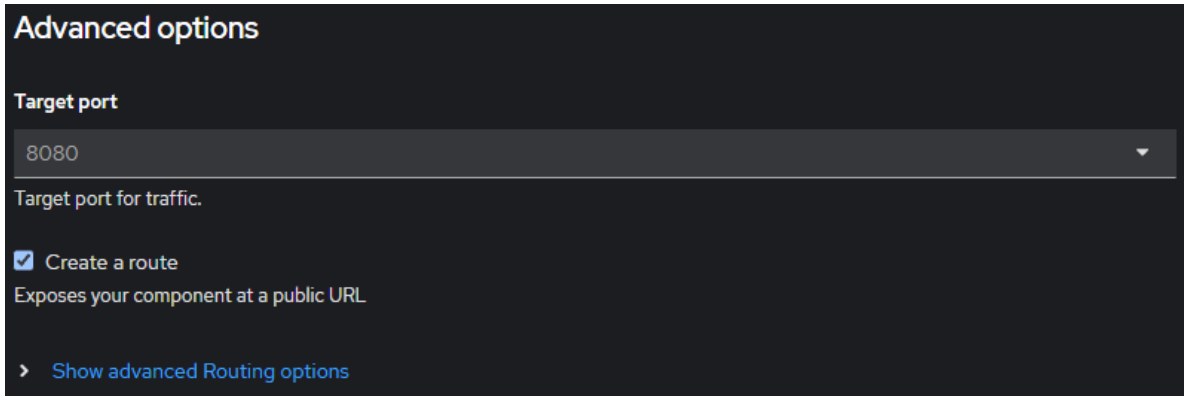
.vscode	Docker	14 days ago
node_modules	Docker	14 days ago
public	Initial commit.	5 months ago
views	Initial commit.	5 months ago
.dockerignore	Docker	14 days ago
.gitignore	Correct Database	14 days ago
Dockerfile	Modificar Dockerfile	11 days ago
app.js	Change node	14 days ago
date.js	Initial commit.	5 months ago
index.html	Initial commit.	5 months ago
package-lock.json	Change package	14 days ago
package.json	Change node	14 days ago

.vscode	Initial commit	14 days ago
public	Initial commit	14 days ago
.dockerignore	Initial commit	14 days ago
.gitignore	Initial commit	14 days ago
Dockerfile	Initial commit	14 days ago
app.js	Eliminar funcion terminar proceso	11 days ago
index.html	Agregar funcion falla	11 days ago
package-lock.json	Initial commit	14 days ago
package.json	Initial commit	14 days ago

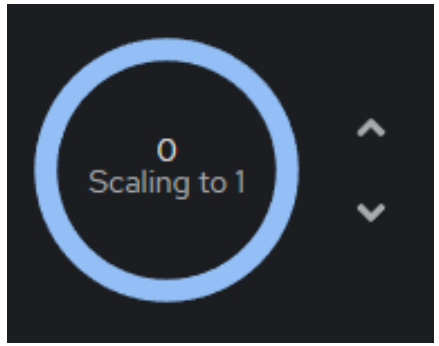
Usamos la interfaz de **OpenShift** para cargar los repositorios.

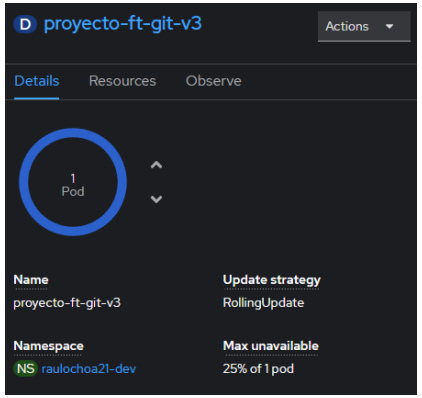






Determinamos el número de **pods,** y visualizamos el **grafo** de **despliegue**.





Enlace para acceder a la aplicación:

https://proyecto-ft-git-v3-raulochoa21-dev.apps.sandbox.x8i5.p1.openshiftapps.com/

Desplegar Localmente

Clonar el repositorio de GitHub.

```
$ git clone https://github.com/RaulF8a/proyecto-ft.git
```

Crear un **Dockerfile** para NodeJS.

```
FROM node:alpine
WORKDIR /app
COPY package.json ./
COPY package-lock.json ./
COPY ./ ./
RUN npm i
CMD ["npm", "run", "start"]
```

Crear la **imagen** en **Docker**.

```
$ docker build -t proyecto-ft .
[+] Building 88.5s (12/12) FINISHED
```

proyecto-ft	latest	3dcf7f523c4e	less than a minute ago	466.66 MB

Crear los archivos **YAML**.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: proyecto-ft
spec:
  replicas: 5
  selector:
    matchLabels:
      app: proyecto-ft
  template:
    metadata:
      labels:
        app: proyecto-ft
    spec:
      containers:
      - name: proyecto-ft-container
        image: proyecto-ft
        imagePullPolicy: Never
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 3000
          protocol: TCP
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: proyecto-ft-service
spec:
  type: ClusterIP
  selector:
    app: proyecto-ft
  ports:
  - port: 5000
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: proyecto-ft-ingress
  annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
  labels:
    name: proyecto-ft
spec:
  rules:
  - http:
      paths:
        - backend:
            service:
              name: proyecto-ft-service
              port:
                number: 5000
          path: /
          pathType: Prefix
```

Iniciamos **minikube** y cargamos la imagen.

```
$ minikube start
😄  minikube v1.27.1 on Microsoft Windows 10 Home 10.0.19044 Build 19044
```

```
$ minikube image load proyecto-ft
```

Usamos **kubectl** para aplicar los archivos YAML.

```
$ kubectl apply -f ./proy_deployment.yaml
deployment.apps/proyecto-ft created
```

```
$ kubectl apply -f ./proy_service.yaml
service/proyecto-ft-service created
```

```
$ kubectl apply -f ./proy_ingress.yaml
ingress.networking.k8s.io/proyecto-ft-ingress created
```

Obtenemos la **dirección IP** y accedemos a la aplicación.

```
$ kubectl get ing
NAME                  CLASS    HOSTS   ADDRESS         PORTS   AGE
proyecto-ft-ingress   <none>   *       192.168.49.2   80      67s
```

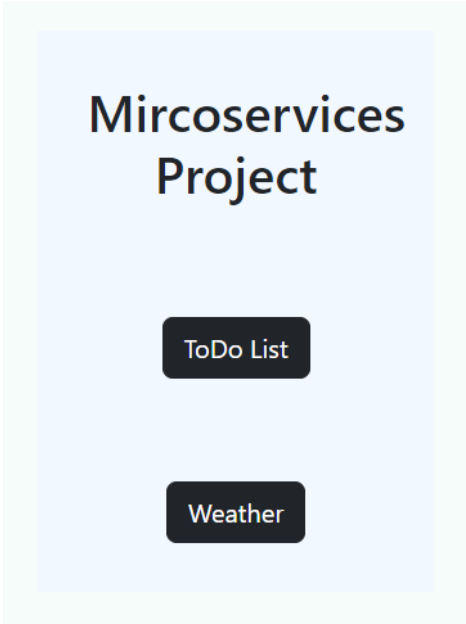


Diagrama de Arquitectura