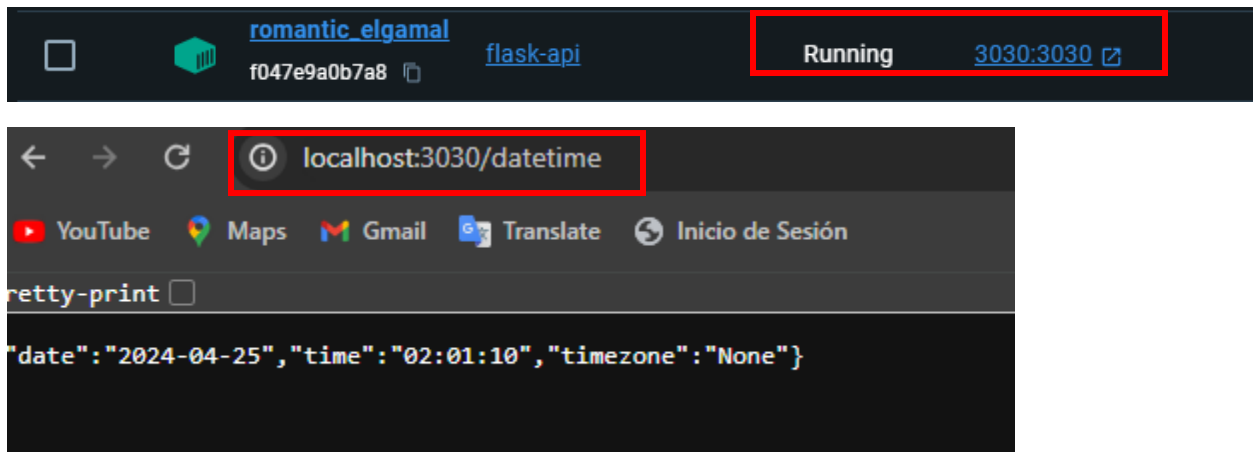


1 Cree un API con un endpoint que devuelva la fecha y hora del sistema

```
app.py x Dockerfile ! kind-config.yaml main.tf
app.py > get_datetime
1 from flask import Flask, jsonify
2 import datetime
3 import pytz
4
5 app = Flask(__name__)
6
7 @app.route('/datetime', methods=['GET'])
8 def get_datetime():
9     now = datetime.datetime.now()
10    response = {
11        'date': now.strftime("%Y-%m-%d"),
12        'time': now.strftime("%H:%M:%S"),
13        'timezone': str(now.tzinfo)
14    }
15    return jsonify(response)
16
17 if __name__ == '__main__':
18     app.run(host='0.0.0.0', port=3030)
19
```

2 Genere una imagen de docker que exponga el API anterior a través del servidor de su predilección (NGINX, Apache, Kestrel, etc)



--A partir de aquí, use Terraform para desarrollar los siguientes pasos

A partir de aquí todo se hizo usando Terraform, kind, kubernetes y nginx para exposición de servicios

A lo largo del documento, se observaran screenshots de como se configuro un servicio a través de la interacción entre Kubernetes usando Terraform, mediante la exposición de un NGINX en el cluster de kubernetes.

Se configuro un archivo .tf para desplegar dos pods de NGINX en el cluster de kubernetes

```
Administrator: Windows PowerShell

    }
  }
}

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

kubernetes_deployment.nginx: Creating...
kubernetes_deployment.nginx: Still creating... [10s elapsed]
kubernetes_deployment.nginx: Still creating... [20s elapsed]
kubernetes_deployment.nginx: Still creating... [30s elapsed]
kubernetes_deployment.nginx: Still creating... [40s elapsed]
kubernetes_deployment.nginx: Still creating... [50s elapsed]
kubernetes_deployment.nginx: Still creating... [1m0s elapsed]
kubernetes_deployment.nginx: Creation complete after 1m10s [id=default/scalable-nginx-example]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Administrador\Documents\U\Noveno ciclo\Virtualizacion\TerraformCloud\learn-terraform-deploy-nginx-kubernetes
> kubectl get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
scalable-nginx-example 2/2     2            2           110s
PS C:\Users\Administrador\Documents\U\Noveno ciclo\Virtualizacion\TerraformCloud\learn-terraform-deploy-nginx-kubernetes
>
```

Posterior se expone NGINX en el nodo a través del puerto 30201:

```
Administrator: Windows PowerShell

+ type                                = "NodePort"

+ port {
+   node_port = 30201
+   port      = 80
+   protocol  = "TCP"
+   target_port = "80"
+ }
}

Plan: 1 to add, 0 to change, 0 to destroy.

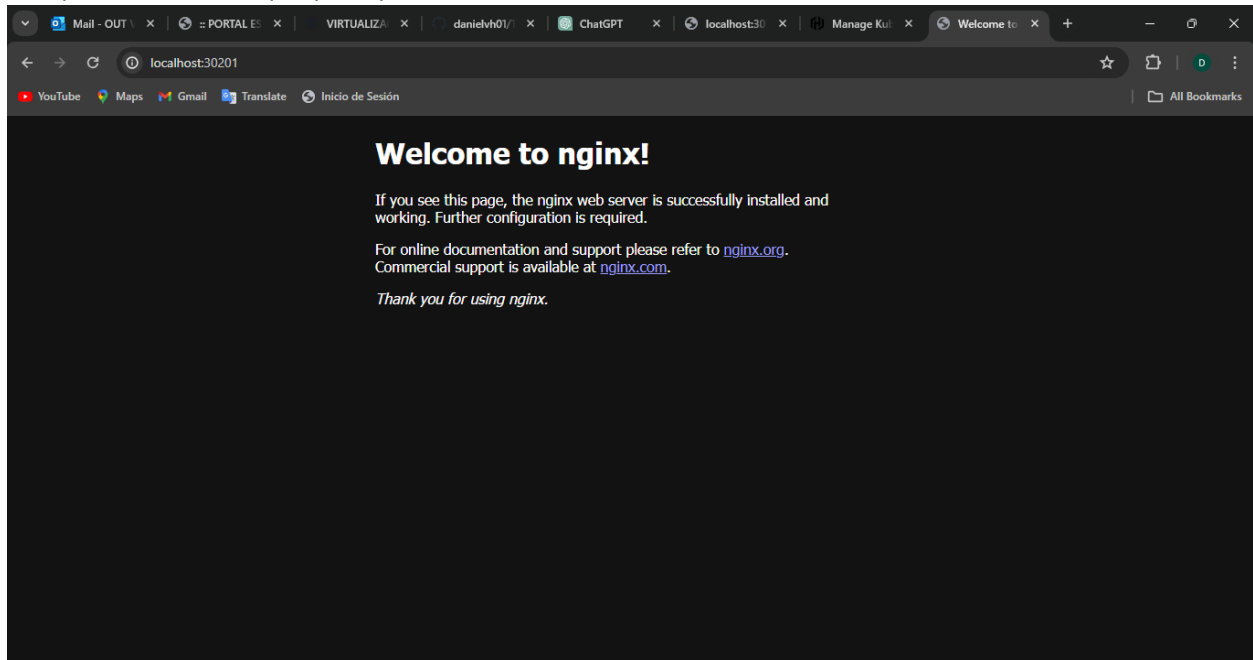
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

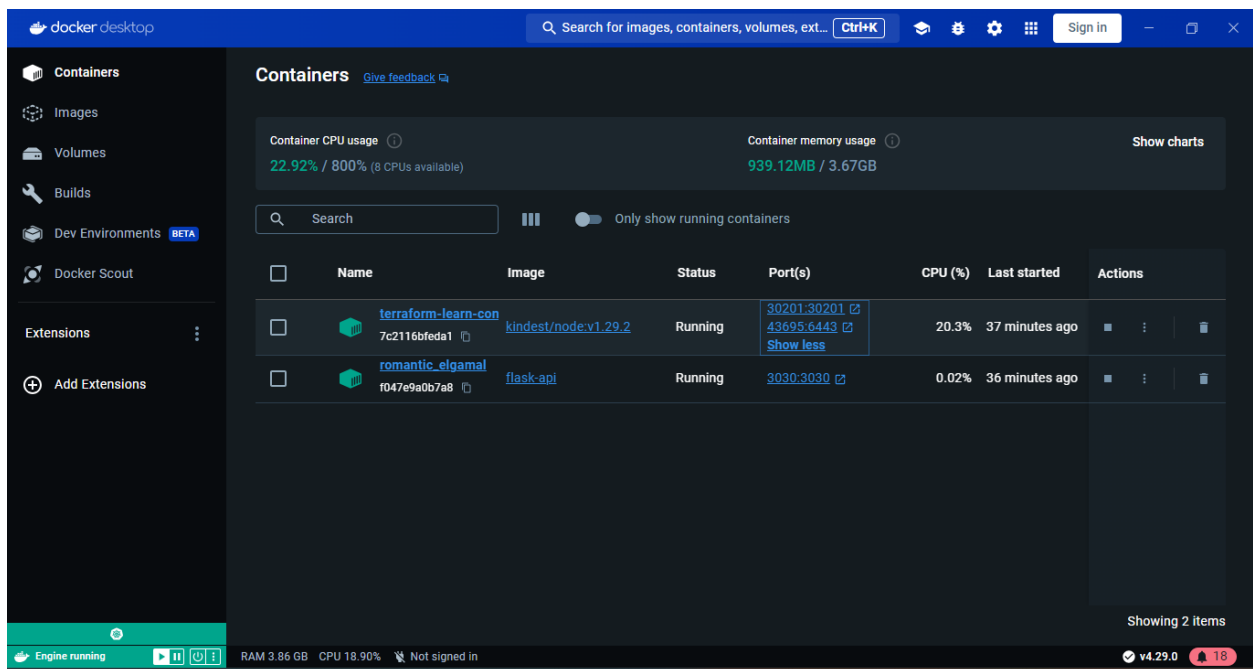
kubernetes_service.nginx: Creating...
kubernetes_service.nginx: Creation complete after 0s [id=default/nginx-example]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Administrador\Documents\U\Noveno ciclo\Virtualizacion\TerraformCloud\learn-terraform-deploy-nginx-kubernetes
> kubectl get services
NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes          ClusterIP   10.96.0.1    <none>        443/TCP          6h13m
nginx-example       NodePort    10.96.77.199 <none>        80:30201/TCP     26s
PS C:\Users\Administrador\Documents\U\Noveno ciclo\Virtualizacion\TerraformCloud\learn-terraform-deploy-nginx-kubernetes
>
```

Y se puede visualizar que ya se puede establecer conexión a dicha instancia:



7. Usando terraform reconstruya su ambiente y muestre evidencia del cambio publicado en el recurso.

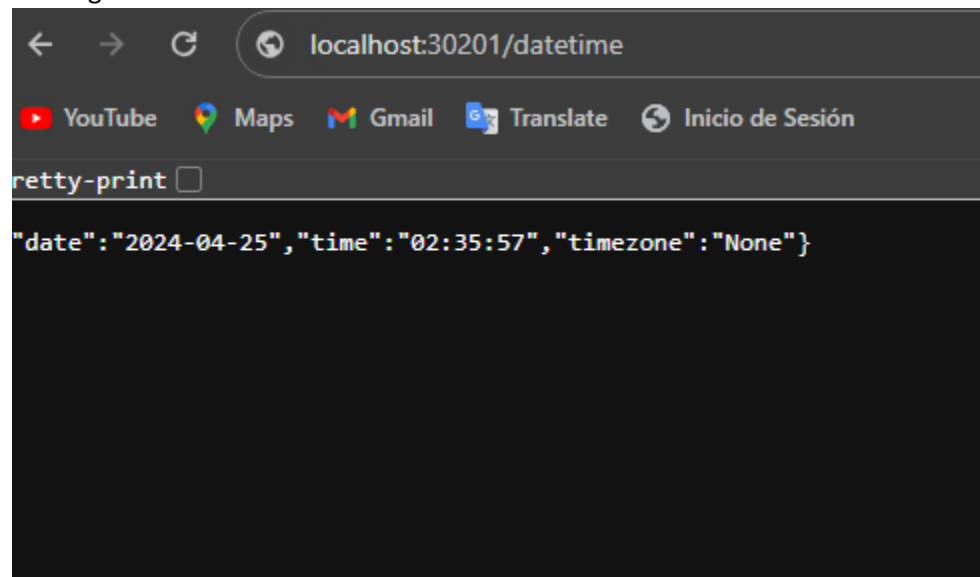


Por ultimo, se vuelve a desplegar la API ahora en NGINX que se acaba de desplegar:

Para ello se agrego lo siguiente al archivo .td:

```
18
19 resource "kubernetes_ingress" "nginx" {
20   metadata {
21     name = "nginx-ingress"
22     annotations = {
23       "kubernetes.io/ingress.class" = "nginx"
24     }
25   }
26   spec {
27     rule {
28       host = "localhost"
29       http {
30         path {
31           path = "/datetime"
32           backend {
33             service_name = kubernetes_service.api.metadata[0].name
34             service_port = kubernetes_service.api.spec[0].port[0].port
35           }
36         }
37       }
38     }
39   }
40 }
```

Y se logra acceder a la API finalmente.



The screenshot shows a web browser window with the address bar displaying `localhost:30201/datetime`. Below the address bar, there are several icons for YouTube, Maps, Gmail, Translate, and a login button labeled "Inicio de Sesión". A "pretty-print" checkbox is visible. The main content area of the browser displays the JSON response: `"date": "2024-04-25", "time": "02:35:57", "timezone": "None"}`.