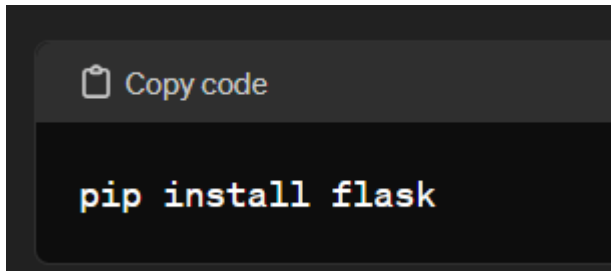
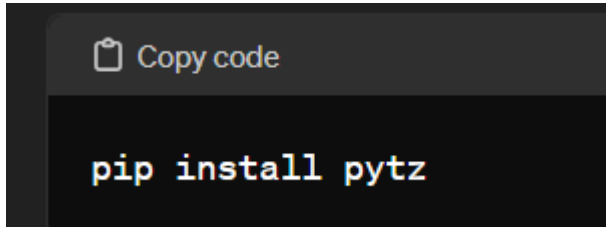


Se iniciara con crear un servidor de Python



```
Copy code

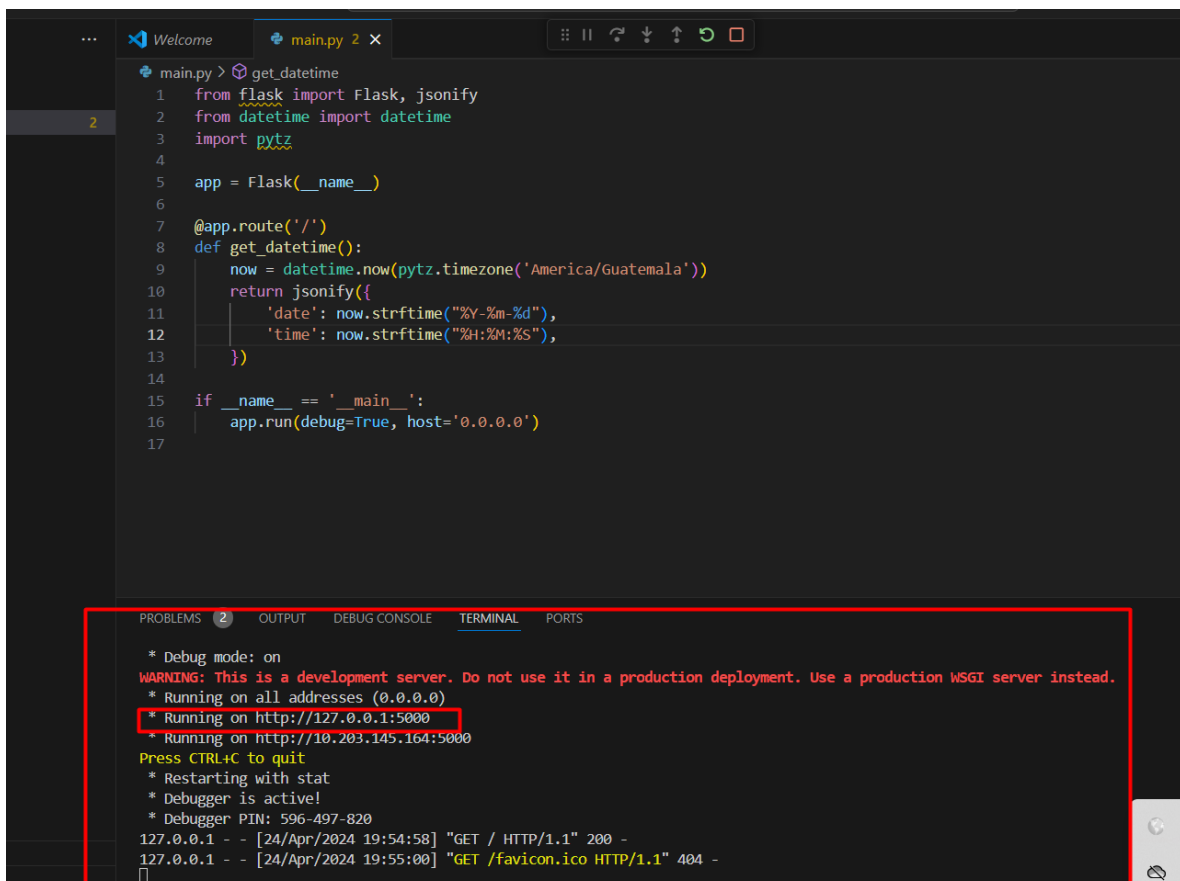
pip install flask
```



```
Copy code

pip install pytz
```

Primero instalamos las librerías que necesitamos

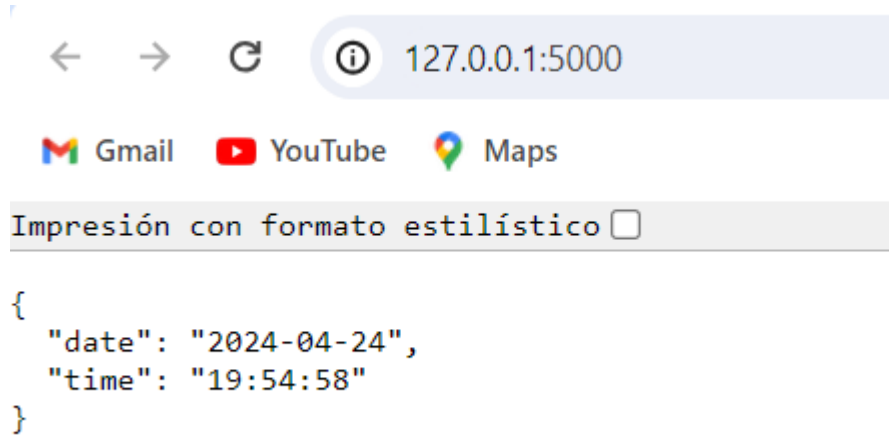


```
main.py > get_datetime
1 from flask import Flask, jsonify
2 from datetime import datetime
3 import pytz
4
5 app = Flask(__name__)
6
7 @app.route('/')
8 def get_datetime():
9     now = datetime.now(pytz.timezone('America/Guatemala'))
10     return jsonify({
11         'date': now.strftime("%Y-%m-%d"),
12         'time': now.strftime("%H:%M:%S"),
13     })
14
15 if __name__ == '__main__':
16     app.run(debug=True, host='0.0.0.0')
17
```

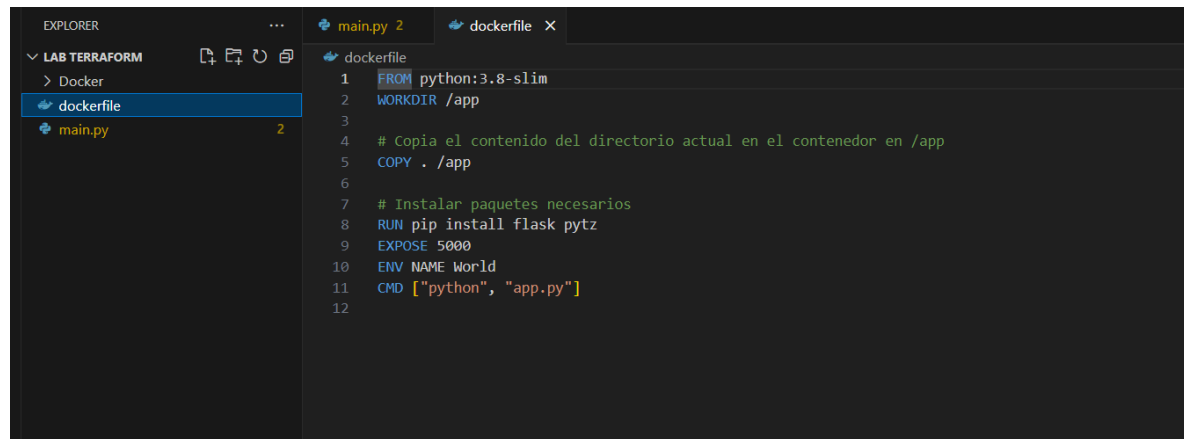
```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://10.203.145.164:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 596-497-820
127.0.0.1 - - [24/Apr/2024 19:54:58] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Apr/2024 19:55:00] "GET /favicon.ico HTTP/1.1" 404 -
```

Lo corremos y en este puerto podemos ver lo que devuelve



Generamos el archivo Docker



Y lo corremos (teniendo Docker instalado y con powershell)

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\WINDOWS\system32> cd "C:\Users\DELL\Desktop\U\Lab Terraform"
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker build -t app_flask .
ERROR: error during connect: in the default daemon configuration on windows, the docker client must be run with elevated privileges to connect: Get "http://localhost:2376/v1.42/ping?X-Real-IP=127.0.0.1": open //./pipe/docker_engine: The system cannot find the file specified.
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker build -t app_flask .
[+] Building 22.8s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/python:3.8-slim
=> [internal] load .dockerignore
=> [internal] transfer context: 2B
=> [1/4] FROM docker.io/library/python:3.8-slim@sha256:2f911e2866173a52104dc10b5e42b7069c2eb05eb78556d18b1ca66508dc445
=> resolve docker.io/library/python:3.8-slim@sha256:2f911e2866173a52104dc10b5e42b7069c2eb05eb78556d18b1ca66508dc445
=> sha256:2f911e2866173a52104dc10b5e42b7069c2eb05eb78556d18b1ca66508dc445: 1.85kB / 1.85kB
=> sha256:11c49f621b99ca802f21c354a404088f0b4f8e5948db6004311da779c774 1.37kB / 1.37kB
=> sha256:5b29acced8b80702bca011e25b3d631090f7ffe35fe2a4977abf53eb3954 6.95kB / 6.95kB
=> sha256:7291442418c3eb060b1308b0b01d307b2afdc51725b065caf29c335b31dc7 3.51kB / 3.51kB
=> sha256:545ebfaa7506d83d4862d6b0ca34e531e2e906431f42a8e59968cc372099695 11.60kB / 11.60kB
=> sha256:80ee18b20840648abeedaab21c9d7a6b4319sec8f362d48d55195e0402ebfb 243B / 243B
=> sha256:d361726ad06f2bc2e1928c9b5d4f7f31b10226d544315ca2408b7d9e2dcd16 3.14kB / 3.14kB
=> extracting sha256:10b0cf838012453b7c1539a084215a7bccc0d788e2bcccff2a03af6dbdd4f7 29.15kB / 29.15kB
=> extracting sha256:7291442418c3eb060b1308b0b01d307b2afdc51725b065caf29c335b31dc7 3.51kB / 3.51kB
=> extracting sha256:545ebfaa7506d83d4862d6b0ca34e531e2e906431f42a8e59968cc372099695 11.60kB / 11.60kB
=> extracting sha256:80ee18b20840648abeedaab21c9d7a6b4319sec8f362d48d55195e0402ebfb 243B / 243B
=> extracting sha256:d361726ad06f2bc2e1928c9b5d4f7f31b10226d544315ca2408b7d9e2dcd16 3.14kB / 3.14kB
=> [internal] load build context
=> transferring context: 1.30kB
=> [2/4] WORKDIR /app
=> [3/4] COPY . /app
=> [4/4] RUN pip install flask pytz
=> exporting to image
=> exporting layers
=> writing image sha256:1a7f38eb7c6e064da9a8d10b30e37a38fc1b63c367aef108d21b338757df5
=> naming to docker.io/library/app_flask

View build details: docker-desktop://dashboard/build/default/default/q/evu1uv6962xf2xx6xmks18j

What's Next?
1. Sign in to your Docker account + docker login
2. View a summary of image vulnerabilities and recommendations + docker scout quickview
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker run -p 5000:5000 app_flask
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 263-818-055
```

Lo corremos y podemos ver que con las configuraciones también nos levanta en el mismo puerto.

```
View build details: docker-desktop://dashboard/build/default/default/q/evu1uv6962xf2xx6xmks18j

What's Next?
1. Sign in to your Docker account + docker login
2. View a summary of image vulnerabilities and recommendations + docker scout quickview
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker run -p 5000:5000 app_flask
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 263-818-055
```

← → ↺ ⓘ

127.0.0.1:5000

Gmail

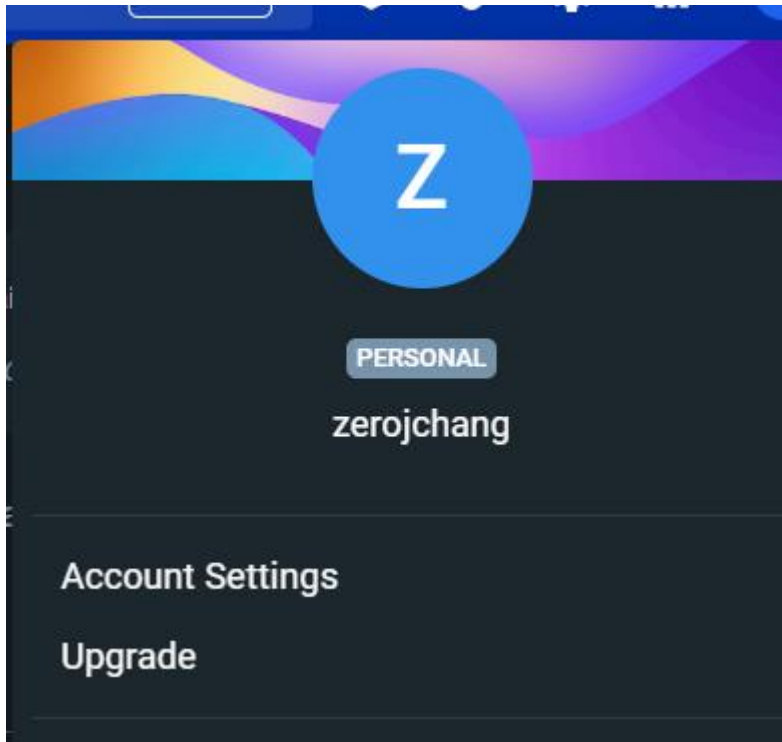
YouTube

Maps

Impresión con formato estilístico ☐

```
{
  "date": "2024-04-24",
  "time": "19:54:58"
}
```

Ahora para levantar en la nube necesitamos correr nuestro usuario.



```
Administrador: Windows PowerShell

* Debugger is active!
* Debugger PIN: 263-818-055
172.17.0.1 - - [25/Apr/2024 02:04:05] "GET / HTTP/1.1" 200 -
PS C:\Users\DELL\Desktop\U\Lab Terraform> ^C
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker build -t zerojchang/flask-app-nube:latest .
[+] Building 0.0s (0/0) docker:default
2024/04/24 20:08:24 http2: server: error reading preface from client //./pipe/docker_engine: file has already been close
[+] Building 0.9s (10/10) FINISHED
=> [internal] load build definition from dockerfile                                0.0s
=> => transferring dockerfile: 276B                                              0.0s
=> [internal] load metadata for docker.io/library/python:3.8-slim                0.8s
=> [auth] library/python:pull token for registry-1.docker.io                    0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                    0.0s
=> [1/4] FROM docker.io/library/python:3.8-slim@sha256:2f911e2866173a52104dc16b5e42b7069c2eba05eb78556d18b1ca665 0.0s
=> [internal] load build context                                                  0.0s
=> => transferring context: 148B                                                 0.0s
=> CACHED [2/4] WORKDIR /app                                                      0.0s
=> CACHED [3/4] COPY . /app                                                       0.0s
=> CACHED [4/4] RUN pip install flask pytz                                       0.0s
=> exporting to image                                                            0.0s
=> => exporting layers                                                            0.0s
=> => writing image sha256:51c0f59c3563c3d68d4f0860a83fc3d5ba697e6c6ef04b939b1c0c9be55d7db7 0.0s
=> => naming to docker.io/zerojchang/flask-app-nube:latest                     0.0s

View build details: docker-desktop://dashboard/build/default/default/ie1h7uwr7leiwb3b25eg3x12

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\DELL\Desktop\U\Lab Terraform>

See 'docker run --help'.
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker run -d -p 5050:5000 zerojchang/flask-app-nube:latest
7e5e83ce8315dabe3b2cd8fd556ef7fe011bad207dca72bf9f832d817e60642e
PS C:\Users\DELL\Desktop\U\Lab Terraform>
```

Para usar una nube debemos loguearnos en este caso Google cloud (se escogió este debido a que por el curso Ingeniería de Software 2 tenemos 1 mes de prueba).

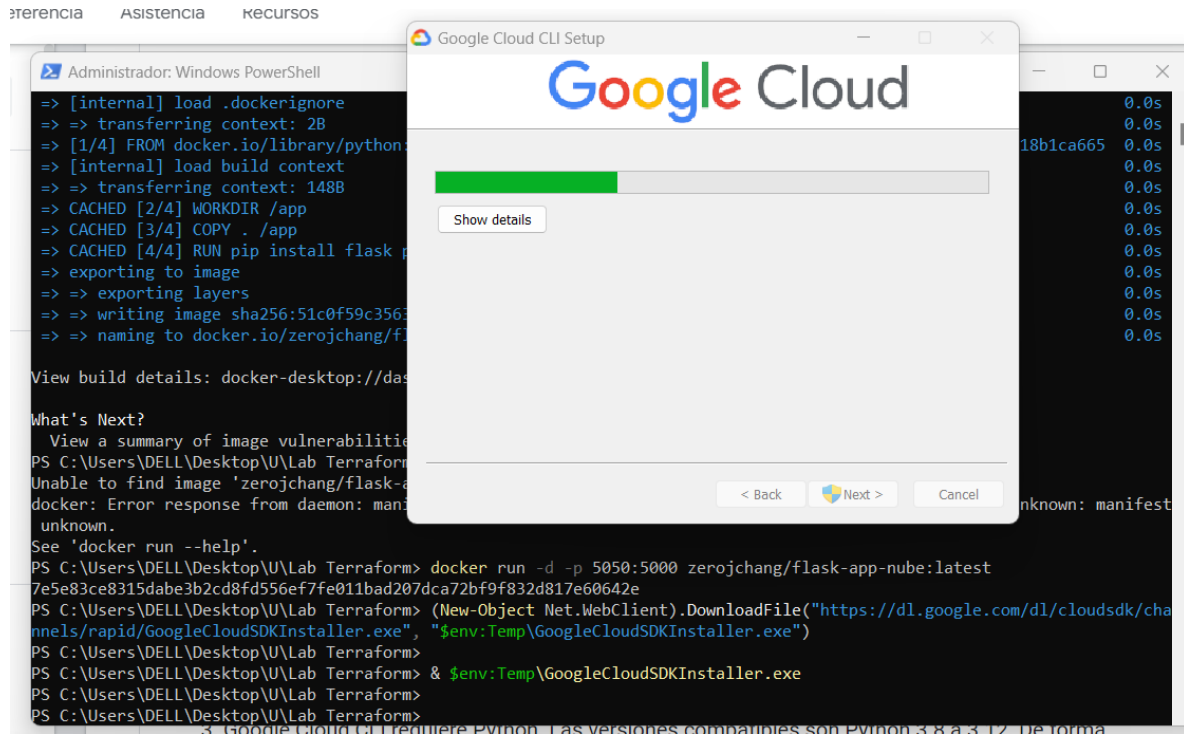
posteriores.

1. Descarga el [instalador de la Google Cloud CLI](#).

Como alternativa, abre una terminal de PowerShell y ejecuta los siguientes comandos:

```
(New-Object Net.WebClient).DownloadFile("https://dl.google.com/dl/cloudsdk/channels/rapid/cmdline-windows-latest")  
& $env:Temp\GoogleCloudSDKInstaller.exe
```

Lo instalamos



```
Administrador: C:\WINDOWS\SYSTEM32\cmd.exe
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

You must log in to continue. Would you like to log in (Y/n)? y

Your browser has been opened to visit:

https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=32555940559.apps.googleusercontent.com&redirect_uri=http%3A%2F%2Flocalhost%3A8085%2F&scope=openid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fappengine.admin+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fsqlservice.login+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcompute+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Faccounts.reauth&state=hdZJ4kDgR03rPwZ4G4V7W0mKXmnWHf&access_type=offline&code_challenge=0x_OIEQRlu0ZE3QfCqs51uw5wMU2ml0h2cj42CgH904&code_challenge_method=S256


You are logged in as: [jdeleonchang@gmail.com].

Pick cloud project to use:
[1] comentando-1530169741507
[2] Enter a project ID
[3] Create a new project
Please enter numeric choice or text value (must exactly match list item): 1

Your current project has been set to: [comentando-1530169741507].


Not setting default zone/region (this feature makes it easier to use
[gcloud compute] by setting an appropriate default value for the
--zone and --region flag).
See https://cloud.google.com/compute/docs/gcloud-compute section on how to set
default compute region and zone manually. If you would like [gcloud init] to be
able to do this for you the next time you run it, make sure the
```

Nos pedirá elegir un proyecto en el cual trabajar.



comentando

[← Detalles del producto](#)



Kubernetes Engine API

[Google Enterprise API](#)

Builds and manages container-based applications, powered by the open source Kubernetes technology.

[HABILITAR](#)[PROBAR ESTA API](#)

[DESCRIPCIÓN GENERAL](#)[DOCUMENTACIÓN](#)[PRODUCTOS RELACIONADOS](#)

Habilitamos Kubernetes en nuestro proyecto

Crea un clúster de Autopilot

Aspectos básicos del clúster

Especifica un nombre y una región para crear un clúster de Autopilot. Después de crear el clúster, puedes implementar tu carga de trabajo a través de Kubernetes y nosotros nos encargaremos del resto, incluidos los siguientes aspectos:

- ✓ **Nodos:** Escalamiento, mantenimiento y aprovisionamiento automático de nodos
- ✓ **Herramientas de redes:** Enrutamiento del tráfico nativo de la VPC para clústeres públicos o privados
- ✓ **Seguridad:** Nodos de GKE protegidos y Workload Identity
- ✓ **Telemetría:** Registro y supervisión de Cloud Operations

Nombre
autopilot-cluster-1

Región
us-central1

Los nombres de los clústeres deben comenzar con una letra minúscula seguida por un máximo de 39 letras minúsculas, números o guiones. No puede terminar con un guion. No puedes cambiar el nombre del clúster una vez creado.

La ubicación regional en la que se encuentran el plano de control y los nodos de tu clúster. No puedes cambiar la región del clúster una vez creada.

[SIGUIENTE: REGISTRO DE FLOTAS](#) [RESTABLECER CONFIGURACIÓN](#)

Habilitamos un cluster

Notificaciones

- ✓ **Crear clúster "cluster-1" de Kubernetes Engine** - Hace unos instantes - comentando
- ✓ **Habilitar servicio: container.googleapis.com** - hace 12 horas - comentando

[VER TODAS LAS ACTIVIDADES](#)

Kubernetes Engine

Clústeres de Kubernetes Engine

Administración de recursos

- Descripción general
- Clústeres**
- Cargas de trabajo
- Equipos **NUEVO**

Administrador: Windows PowerShell

```
unknown.  
$see 'docker run --help'.  
PS C:\Users\DELL\Desktop\U\Lab Terraform> docker run -d -p 5050:5000 zerojchang/flask-app-nube:latest  
7e5e83ce8315dabe3b2cd8fd556ef7fe011bad207dca72bf9f832d817e60642e  
PS C:\Users\DELL\Desktop\U\Lab Terraform> (New-Object Net.WebClient).DownloadFile("https://dl.google.com/dl/cloudsdk/channels/rapid/GoogleCloudSDKInstaller.exe", "$env:Temp\GoogleCloudSDKInstaller.exe")  
PS C:\Users\DELL\Desktop\U\Lab Terraform>  
PS C:\Users\DELL\Desktop\U\Lab Terraform> & $env:Temp\GoogleCloudSDKInstaller.exe  
PS C:\Users\DELL\Desktop\U\Lab Terraform>  
PS C:\Users\DELL\Desktop\U\Lab Terraform> gcloud container clusters get-credentials cluster-1 --zone us-central1
```

y ejecutamos

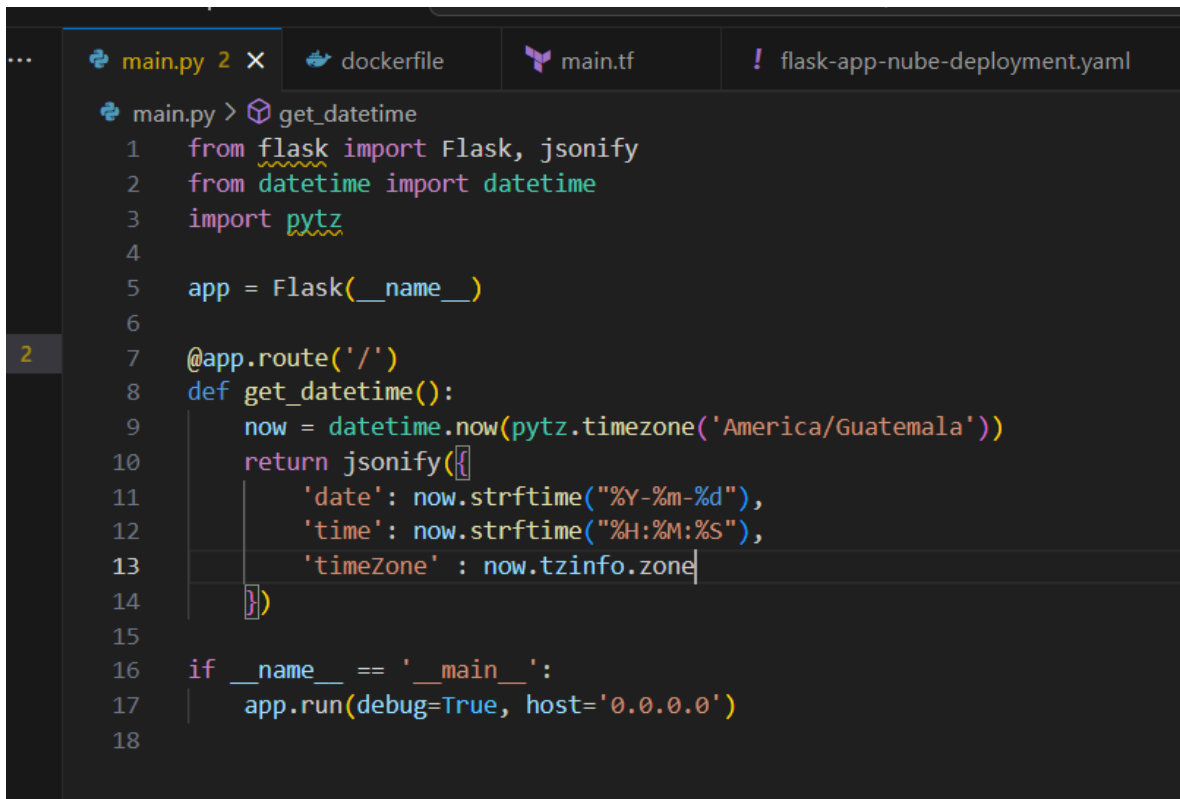
Creamos archivo .yaml

```
main.tf
1  provider "google"{
2      project = "comentando"
3      region = "us-central1"
4  }
5
6  resource "google_container_cluster" "cluster-cloud"{
7      name = "cluster-1"
8      location = "us-central1"
9  }
```

```
apiVersion: v1
kind: Service
metadata:
  name: flask-app-nube
spec:
  selector:
    app: flask-app-nube
  ports:
    - protocol: TCP
      port: 80
      targetPort: 5000
  type: LoadBalancer
```

Y configuraciones para Docker

Antes de proceder cambiamos el archivo .py para mostrar el dato faltante.



```
main.py 2 x  dockerfile  main.tf  ! flask-app-nube-deployment.yaml

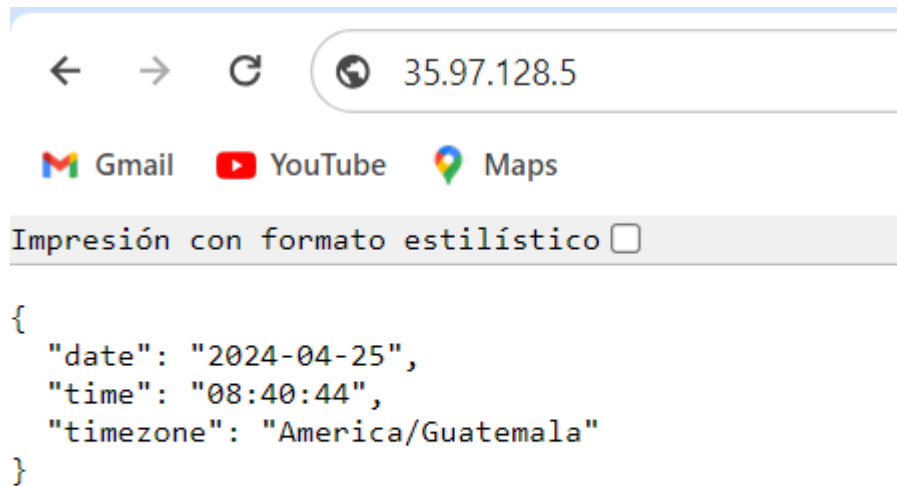
main.py > get_datetime
1  from flask import Flask, jsonify
2  from datetime import datetime
3  import pytz
4
5  app = Flask(__name__)
6
7  @app.route('/')
8  def get_datetime():
9      now = datetime.now(pytz.timezone('America/Guatemala'))
10     return jsonify({
11         'date': now.strftime("%Y-%m-%d"),
12         'time': now.strftime("%H:%M:%S"),
13         'timeZone' : now.tzinfo.zone
14     })
15
16 if __name__ == '__main__':
17     app.run(debug=True, host='0.0.0.0')
18
```

Ejecutaos todos los pasos anteriores

```
PS C:\Users\DELL\Desktop\U\Lab Terraform> kubectl apply -f flask-app-nube-deploy.yaml
```

Corremos el comando para el despliegue

Este nos dará un link para poder acceder a nuestro API endpoint en la nube



Como podemos ver al acceder al link nos muestra el resultado de nuestro endpoint