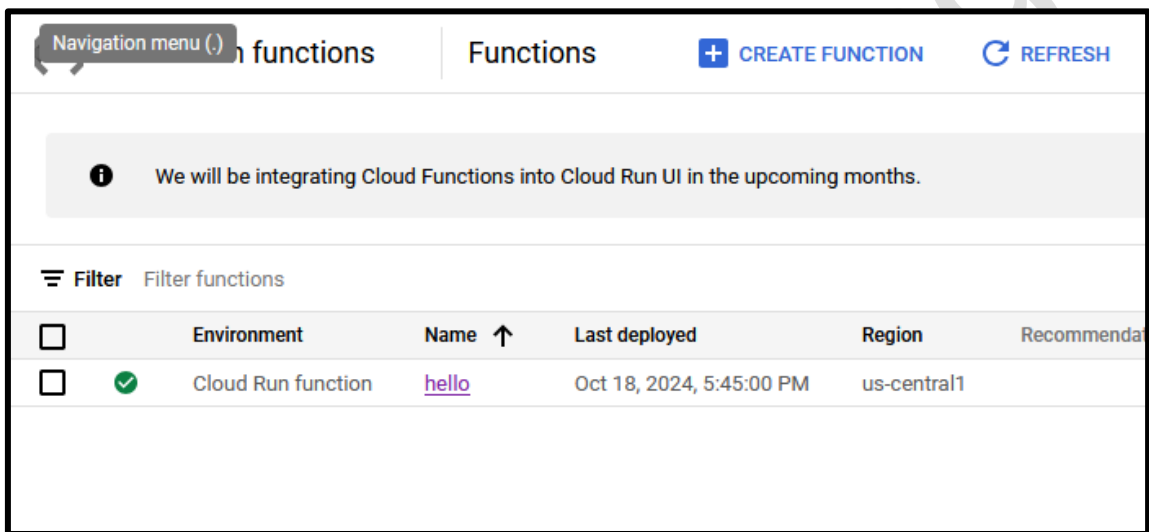


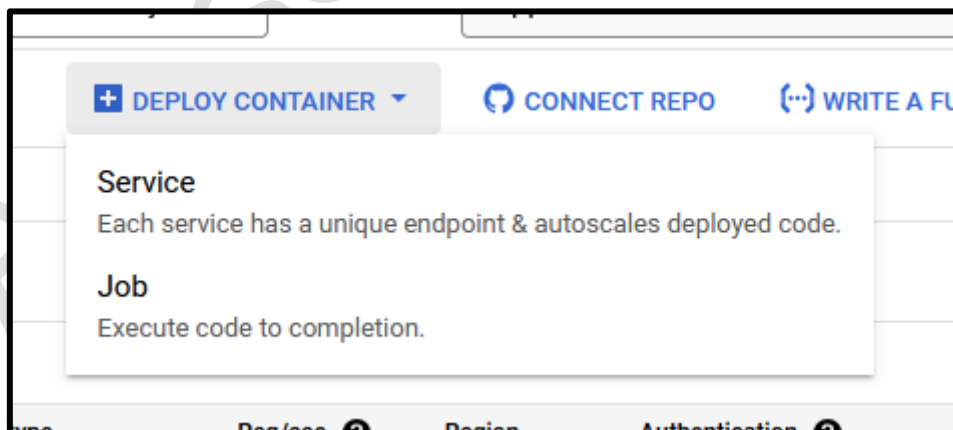
Google Cloud Platform Practices

1. Cloud Function

- In this example, we're going to create a Python-based Cloud Function. What the function will do is return a text in capital letters that is passed to it as an argument.
- We access Cloud Function. My screen may be different from the one you have



- Select "CREATE FUNCTION".



- The name of the function will be "upperText". We put an HTTPS trigger and allow full access.

Basics

Environment ▼ ?
Cloud Run function


Function name * ?
upperText

Region * ▼ ?
us-central1 (Iowa)

Trigger

Trigger type ▼
HTTPS

URL
https://us-central1-centering-oxide-437720-n3.cloudfunctions.net/upperText



Authentication ?

Allow unauthenticated invocations
Check this if you are creating a public API or website.

Require authentication
Manage authorized users with Cloud IAM.

- Click on the "NEXT".
- On the next screen we indicate Python3.9 as the language.
- Something similar to the following should come out.

Runtime
Python 3.9
▼ ?

Entry point *
hello_http
?

TEST FUNCTION

Source code

Inline Editor

main.py ✎ 🗑

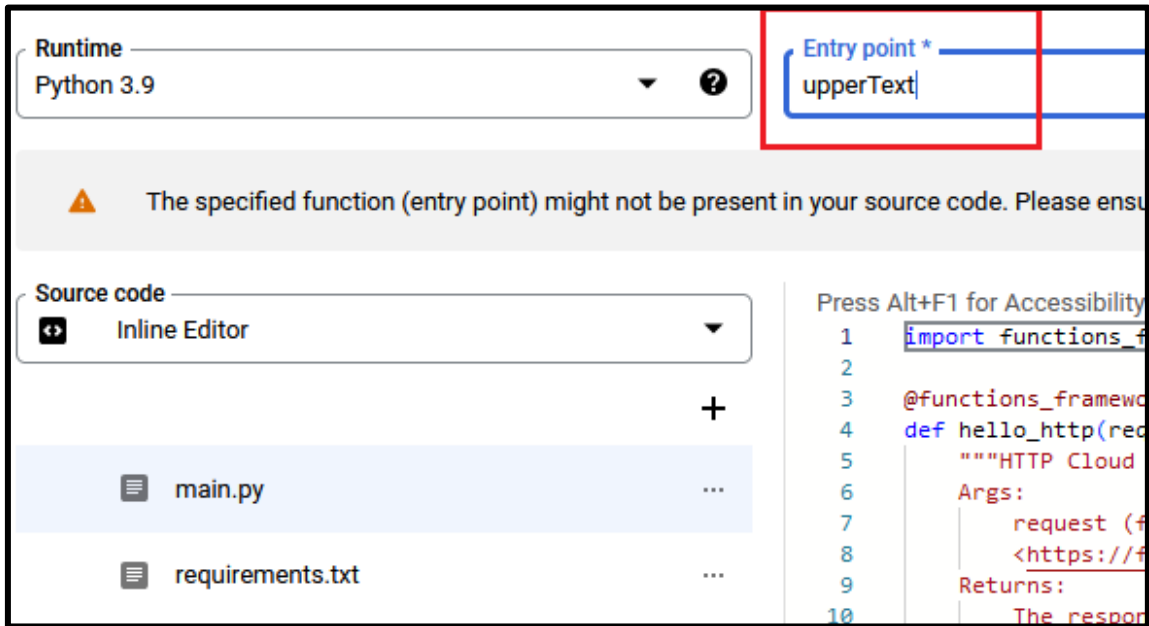
requirements.txt ⋮

Press Alt+F1 for Accessibility Options.

```

1 import FunctionsFramework
2
3 @functions_framework.http
4 def hello_http(request):
5     """HTTP Cloud Function.
6
7     Args:
8         request (flask.Request): The request object.
9         <https://flask.palletsprojects.com/en/1.1.x/api/#incoming-request-data>
10
11     Returns:
12         The response text, or any set of values that can be turned into a
13         Response object using 'make_response'
14         <https://flask.palletsprojects.com/en/1.1.x/api/#flask.make_response>.
15     """
16     request_json = request.get_json(silent=True)
17     request_args = request.args
18
19     if request_json and 'name' in request_json:
20         name = request_json['name']
21     elif request_args and 'name' in request_args:
22         name = request_args['name']
23     else:
24         name = 'World'
25     return 'Hello {}'.format(name)
                
```

- We changed the EntryPoint to the name of our function



- We changed the code of the function:

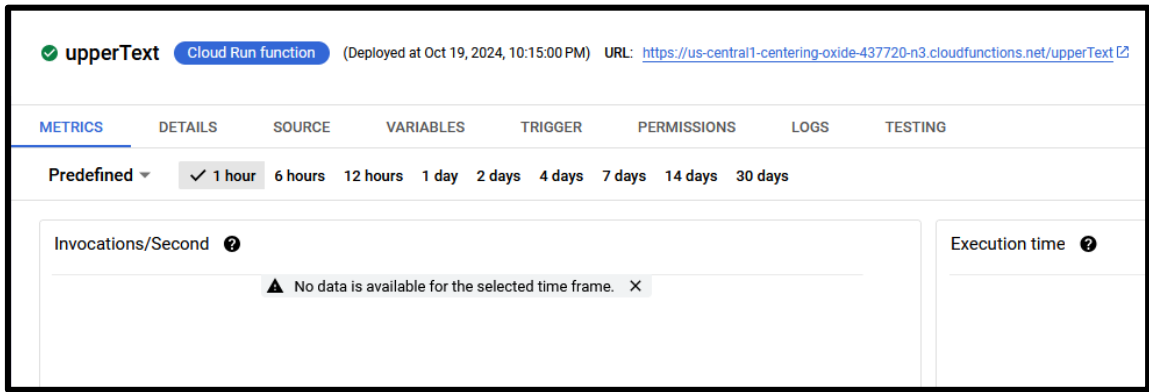
```
import functions_framework

@functions_framework.http
def upperText(request):
    request_json = request.get_json(silent=True)
    request_args = request.args

    # Check if text comes in JSON body
    if request_json and 'text' in request_json:
        text = request_json['text']
    elif request_args and 'text' in request_args:
        text = request_args['text']
    else:
        return 'No text provided', 400

    # Convert text to uppercase
    return text.upper()
```

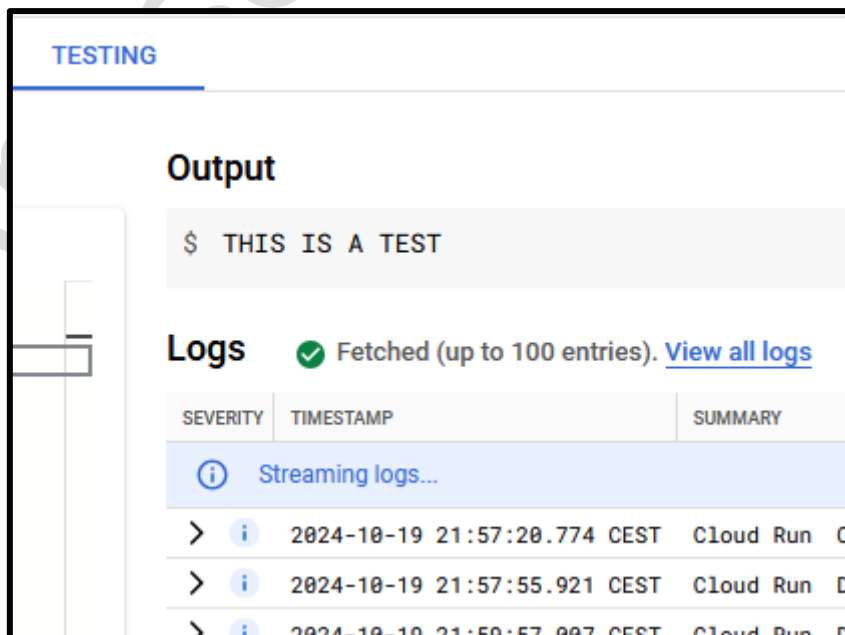
- Click on the "DEPLOY" button.
- If everything is correct, a screen similar to the following should appear:



- We're going to go to the "TESTING" tab to test our feature.
- We put any text in the function. We need to change the "name" parameter to "text".



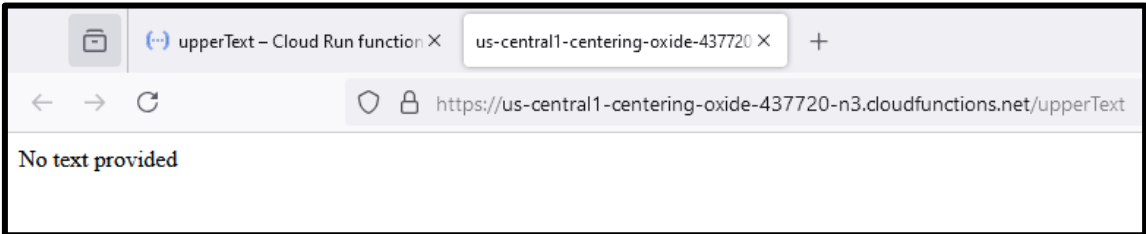
- Click on the "TEST THE FUNCTION" button.
- A result like the following should appear.



- We copy the URL to test the feature from a WEB browser

URL: <https://us-central1-centering-oxide-437720-n3.cloudfunctions.net/upperText>

- If we run it without any parameters, the following message should appear.



- Now we put an argument. We must put the symbol "?" followed by the argument "text" and the text to be converted into capital letters.
- For example

[URL?text=A simple Cloud Function](https://us-central1-centering-oxide-437720-n3.cloudfunctions.net/upperText?text=A simple Cloud Function)

- Result

