

How to Locally Develop a GCP Cloud Run Function that Calls an External API

COE-558: Cloud and Edge Computing
KFUPM - Term 241 - Nov 2024

Mohammad Al-Mohsin
mohammad.mohsin@kfupm.edu.sa

This guide describes how to prepare a local environment on your machine or on a server to develop **HTTP-triggered Node.js GCP Cloud Run functions** (formerly known as Cloud Functions). For other languages and more details, refer to the links in the “References” section.

On this guide, we will create a Cloud Run function called “**time**” that returns the current date and time from an external API.

1. Install Node.js

Install Node.js on your machine if it is not already installed. Node.js allows you to run server-side JavaScript. You can use the following page for various ways to install Node.js on macOS, Windows, or Linux. The installation also includes Node Package Manager (npm).

<https://nodejs.org/en/download/prebuilt-installer>

2. Local Directory

Create a directory (e.g. **time-service**) on your machine and change to that directory:

```
mkdir time-service  
cd time-service
```

3. Initialize Node.js

Initialize Node.js application and accept default answers. This will create **package.json** file for you:

```
npm init --yes
```

4. Install GCP Functions Framework

Install GCP Functions Framework library, which will allow you to run Cloud Run Functions on your local machine or server. It will be installed in **node_modules** directory and will be added as a dependency in **package.json** file.

```
npm install @google-cloud/functions-framework
```

5. Install Axios

Install Axios, which is a [promise-based](#) HTTP Client. We will use it to make HTTP requests to the external API. It will be installed in [node_modules](#) directory and will be added as a dependency in [package.json](#) file.

```
npm install axios
```

6. Start Script

Add the following [start](#) script in [package.json](#) file in the [scripts](#) block replacing the [test](#) script, if it exists. The [target](#) parameter specifies the name of the function to execute (e.g. [time](#))

```
"scripts": {  
  "start": "npx functions-framework --signature-type=http --target=time"  
}
```

7. Write the Code for your Node.js Cloud Run Function

Create [index.js](#) file with the following content using your preferred IDE or text editor:

```
const functions = require('@google-cloud/functions-framework');  
const axios = require('axios');  
const api_url = 'https://timeapi.io/api/time/current/zone?timeZone=Asia%2FRiyadh';  
  
functions.http('time', (req, res) => {  
  switch (req.method) {  
    case 'GET':  
      try {  
        axios.get(api_url).then( (timeServiceResponse) => {  
          console.log(timeServiceResponse);  
          const currentTime = timeServiceResponse.data;  
          result = {  
            'year': currentTime.year,  
            'month': currentTime.month,  
            'day': currentTime.day,  
            'hour': currentTime.hour,  
            'minute': currentTime.minute,  
            'dayOfWeek': currentTime.dayOfWeek,  
            'date': currentTime.date,  
            'timeZone': currentTime.timeZone  
          }  
          console.log(result);  
          res.status(200).send(result);  
        });  
      }  
      catch (error) {  
        res.status(500).send('error getting current time');  
      }  
      break;  
    default:  
      res.status(405).send('Only GET method is allowed');  
      break;  
  }  
});
```

8. Start your Cloud Run Function Locally

Run the following command to start your Node.js application. It will run on port 8080 by default.

```
npm start
```

You should get the following output indicating that your application is running on <http://localhost:8080>

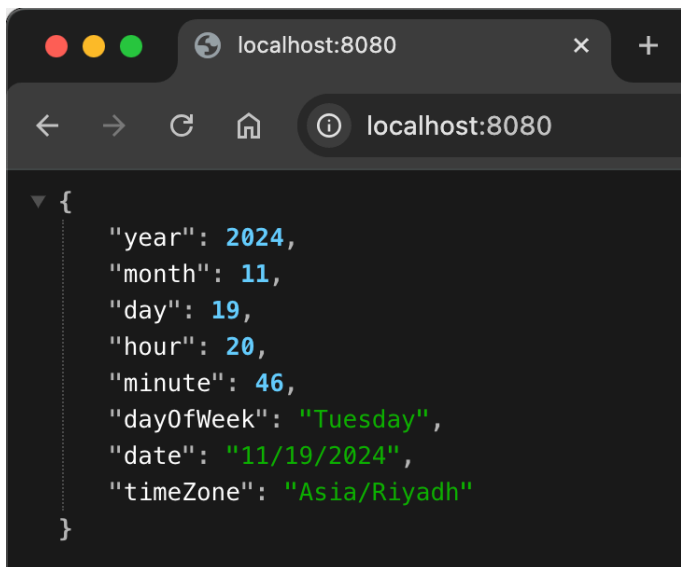
```
> time@1.0.0 start
> npx functions-framework --signature-type=http --target=time

Serving function...
Function: time
Signature type: http
URL: http://localhost:8080/
```

9. Test your Cloud Run Function Locally

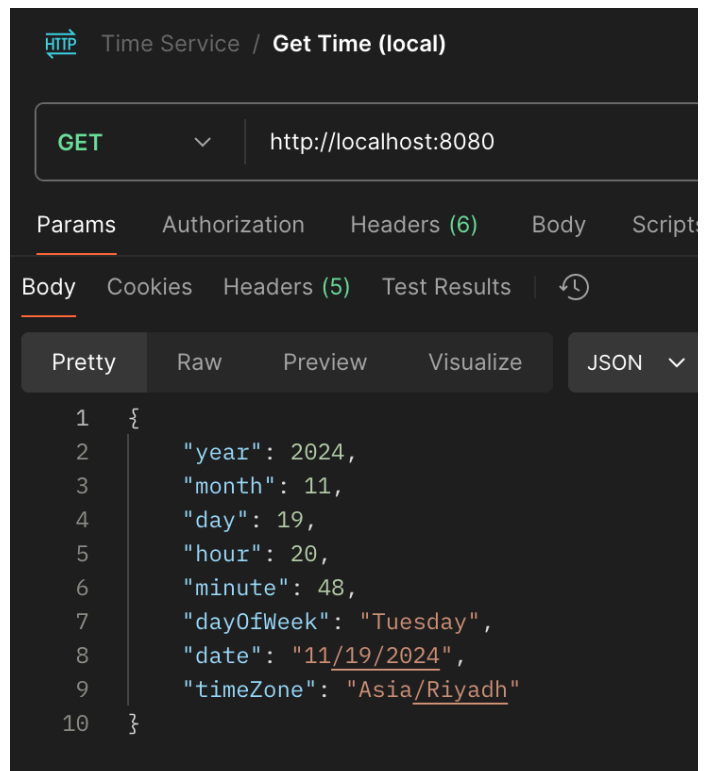
Use your browser, Postman, or curl command and make an HTTP GET request to <http://localhost:8080/> to test your Cloud Run Function.

You should get a JSON output similar to the following:



A screenshot of a web browser window with the address bar showing 'localhost:8080'. The main content area displays a JSON object with the following fields: 'year' (2024), 'month' (11), 'day' (19), 'hour' (20), 'minute' (46), 'dayOfWeek' ('Tuesday'), 'date' ('11/19/2024'), and 'timeZone' ('Asia/Riyadh').

```
{
  "year": 2024,
  "month": 11,
  "day": 19,
  "hour": 20,
  "minute": 46,
  "dayOfWeek": "Tuesday",
  "date": "11/19/2024",
  "timeZone": "Asia/Riyadh"
}
```



A screenshot of the Postman application showing a GET request to 'http://localhost:8080'. The 'Body' tab is selected, displaying the JSON response in a 'Pretty' format. The JSON object is identical to the one shown in the browser screenshot.

```
1 {
2   "year": 2024,
3   "month": 11,
4   "day": 19,
5   "hour": 20,
6   "minute": 48,
7   "dayOfWeek": "Tuesday",
8   "date": "11/19/2024",
9   "timeZone": "Asia/Riyadh"
10 }
```

When you complete your testing, you can stop the running Node.js application by pressing Ctrl+C

10. Deploy your Cloud Run Function on GCP

- Go to GCP Console (<https://console.cloud.google.com>)
 - Go to Cloud Run Functions
 - Click on Create Function
 - Write the name of the function (e.g. `time`)
 - Under Trigger:
 - Select **HTTPS** trigger type.
 - Under Authentication:
 - Select **Allow unauthenticated invocations** to allow access to the function without authentication.
 - If you select **Require authentication**:
 - You will need to pass the following HTTP header with your HTTP request:
 - **Authorization: Bearer** `token`
 - You can get the token value by executing the following **gcloud** command (e.g. from GCP Cloud Shell):
- ```
gcloud auth print-identity-token
```
- The authenticated user must have Cloud Run Invoker role for the token to be accepted and be able to run the function.
  - Once the function is deployed, you can change the authentication settings from **Cloud Run** by updating the corresponding Cloud Run service that is hosting the function.
- Click next and update the code for **index.js** as in your local machine.
- Update the entry point of the function (e.g. `time`).
- Update **package.json** by adding axios dependency as defined in your local machine. The file should be similar to this:

```
{
 "dependencies": {
 "@google-cloud/functions-framework": "^3.0.0",
 "axios": "^1.7.7"
 }
}
```

- Click **Deploy** and wait for building and deploying the function.

## 11. Test your Cloud Run Function on GCP

Copy the URL of the Cloud Run Function and test it with your favorite HTTP client (browser, Postman, curl, ...etc).