

# Serverless | AFs

Running and publish your first Azure Functions project



# Prerequisites and more!

Install *Visual Studio Code*.

Install *Git*, *NodeJS8+* and *NPM*.

Install *Azure Functions Core Tools*.

Install *Azure Functions extension* for *VS Code*.

An *Azure subscription* (free account).

Main supported languages:

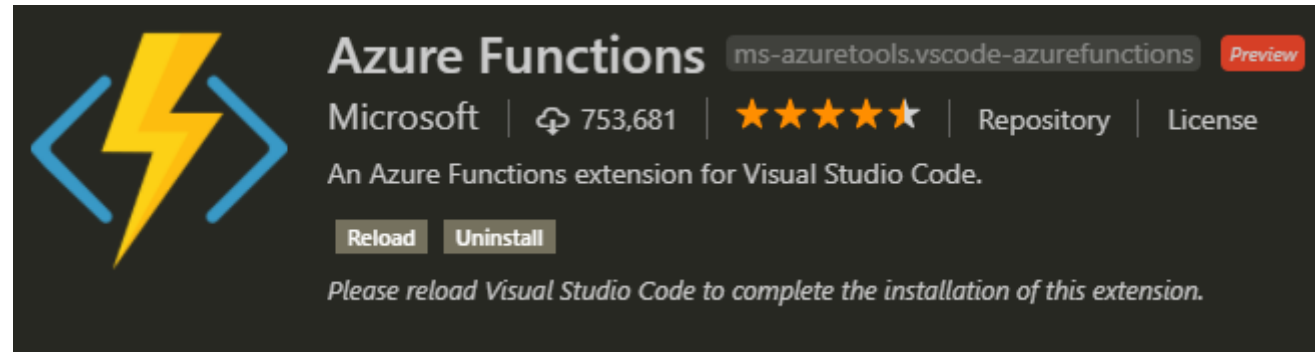
Language	Extension
C#	C# for VS Code
Java	Debugger for Java * + Maven 3+
Typescript	.Net Core 2.2 (preview)
Javascript	Node 8.0+

# Prerequisites and more!

The following steps use download SDK and npm to install Core Tools on Windows. You can also use Chocolatey.

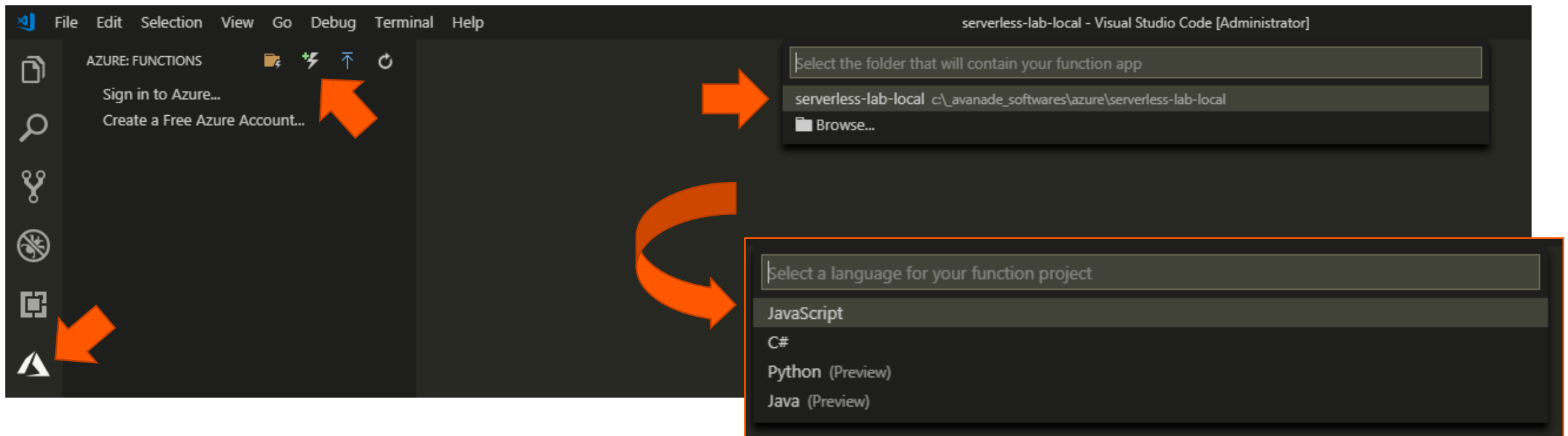
Create path *"serverless-lab-local"*.

Install *npm -g azure-functions-core-tools*



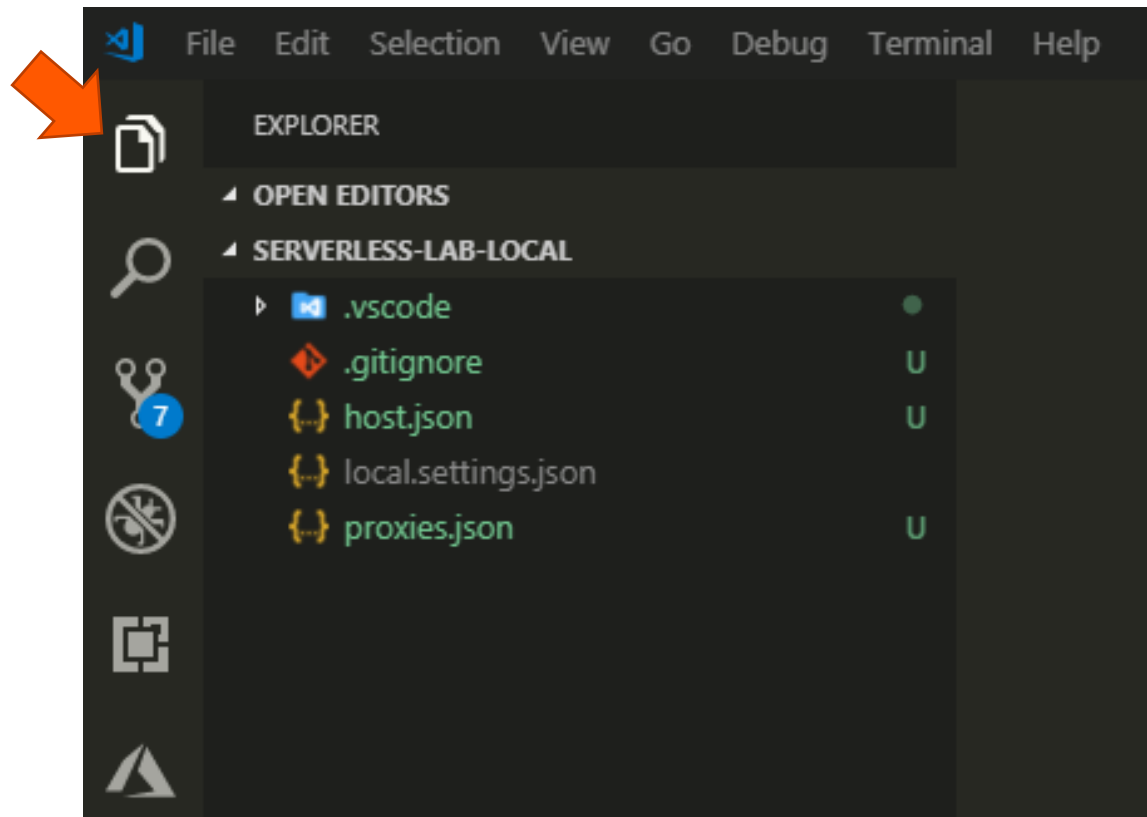
# Create an Azure Function Project

The Azure Functions project template in Visual Studio Code creates a project that can be published to a function app in Azure. A function app lets you group functions as a logical unit for management, deployment, and sharing of resources.



# Create an Azure Function Project

The result can be viewed like image by clicking the *Explorer* icon.

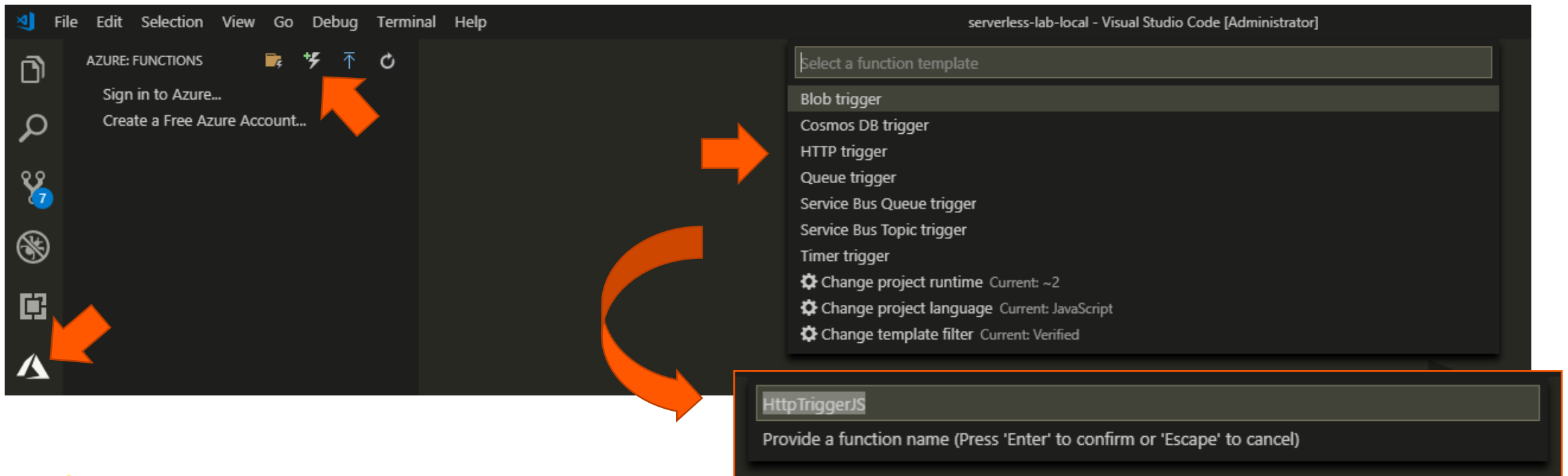


# Create an Azure Function Project

From *Azure: Functions view*, choose the *Create Function icon*.

Select the folder with your function app project and select the *HTTP trigger* function template.

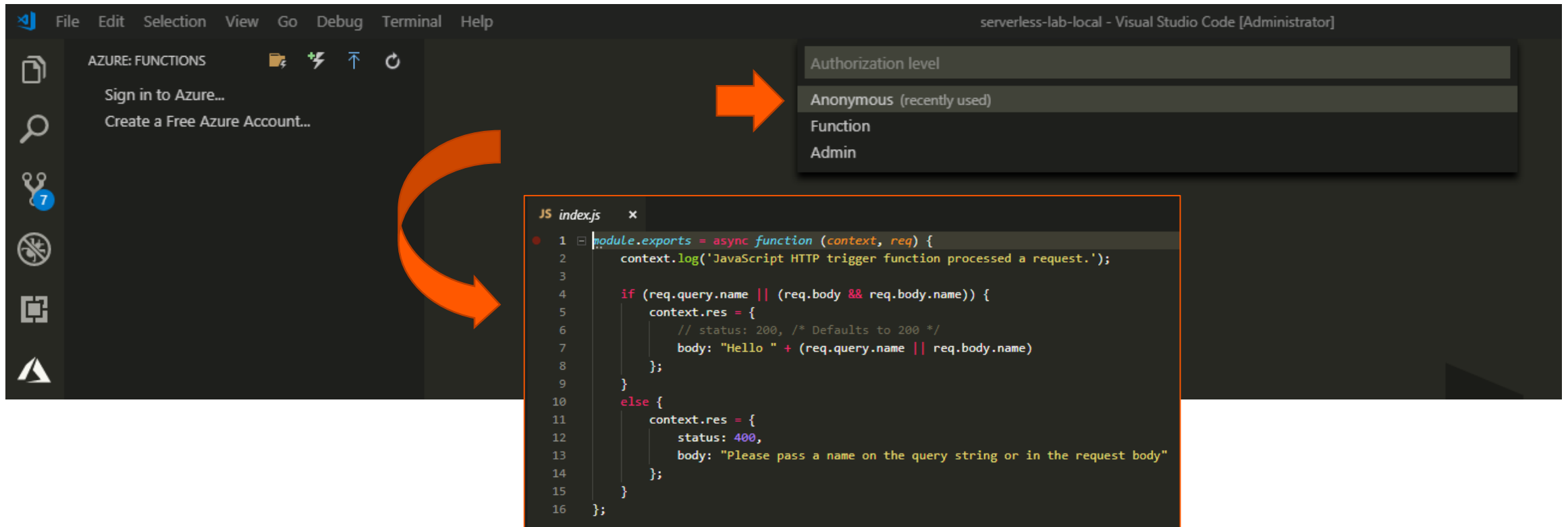
Type *HTTPTriggerJS* for the function name and press *Enter*.



# Create an Azure Function Project

In the list, select *Anonymous* authentication.

A function is created in your chosen language using the template for an HTTP-triggered function.

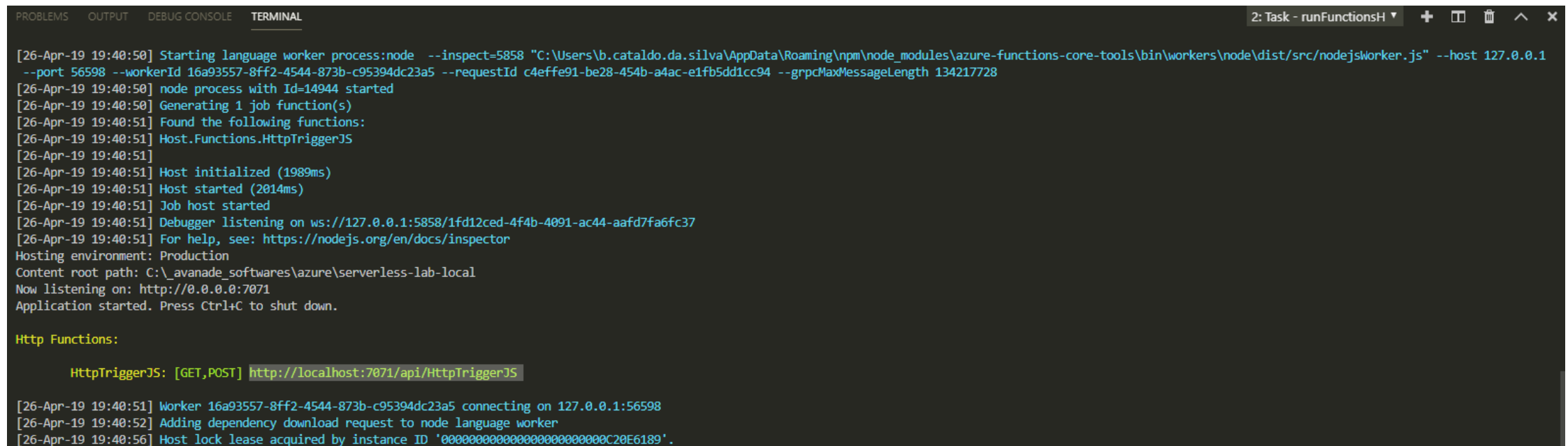


# Let's ride!

Let's you run an Azure Functions project on your local development computer.

To test your function, set a *breakpoint* in the function code and *press F5* to start the function app project.

Output from Core Tools is displayed in the Terminal panel or Prompt Command.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Task - runFunctionsH + [ ] X

[26-Apr-19 19:40:50] Starting language worker process:node --inspect=5858 "C:\Users\b.cataldo.da.silva\AppData\Roaming\npm\node_modules\azure-functions-core-tools\bin\workers\node\dist/src/nodejsWorker.js" --host 127.0.0.1 --port 56598 --workerId 16a93557-8ff2-4544-873b-c95394dc23a5 --requestId c4effe91-be28-454b-a4ac-e1fb5dd1cc94 --grpcMaxMessageLength 134217728
[26-Apr-19 19:40:50] node process with Id=14944 started
[26-Apr-19 19:40:50] Generating 1 job function(s)
[26-Apr-19 19:40:51] Found the following functions:
[26-Apr-19 19:40:51] Host.Functions.HttpTriggerJS
[26-Apr-19 19:40:51]
[26-Apr-19 19:40:51] Host initialized (1989ms)
[26-Apr-19 19:40:51] Host started (2014ms)
[26-Apr-19 19:40:51] Job host started
[26-Apr-19 19:40:51] Debugger listening on ws://127.0.0.1:5858/1fd12ced-4f4b-4091-ac44-aafd7fa6fc37
[26-Apr-19 19:40:51] For help, see: https://nodejs.org/en/docs/inspector
Hosting environment: Production
Content root path: C:\_avanade_softwares\azure\serverless-lab-local
Now listening on: http://0.0.0.0:7071
Application started. Press Ctrl+C to shut down.

Http Functions:

    HttpTriggerJS: [GET,POST] http://localhost:7071/api/HttpTriggerJS

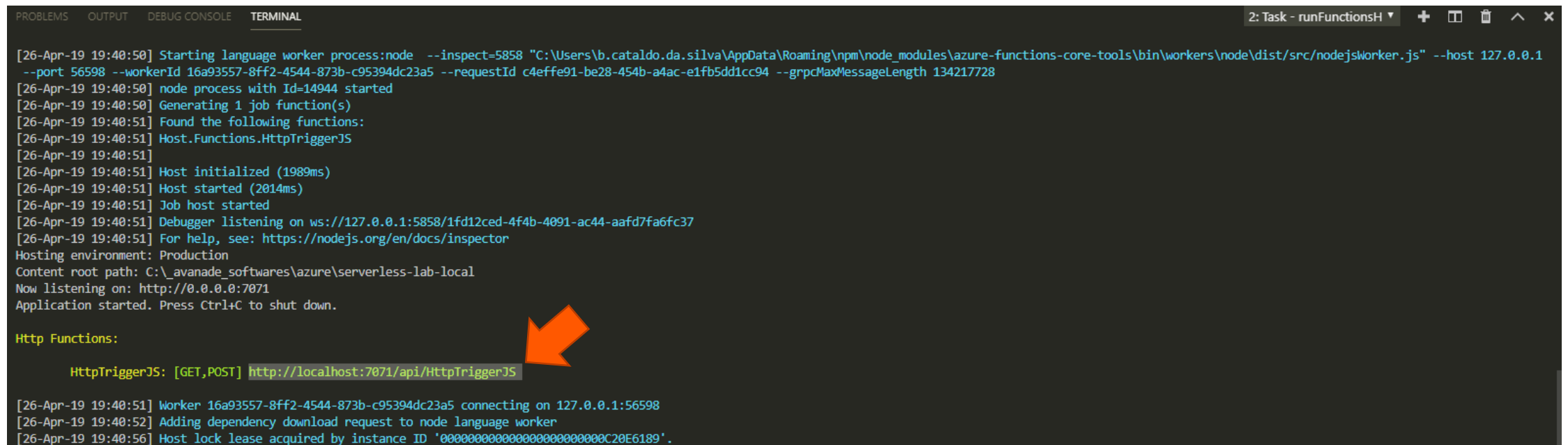
[26-Apr-19 19:40:51] Worker 16a93557-8ff2-4544-873b-c95394dc23a5 connecting on 127.0.0.1:56598
[26-Apr-19 19:40:52] Adding dependency download request to node language worker
[26-Apr-19 19:40:56] Host lock lease acquired by instance ID '00000000000000000000000000000000C20E6189'.
```



# Let's ride!

In the **Terminal** panel, copy the URL endpoint of your HTTP-triggered function.

Paste the URL for the HTTP request into your browser's address bar. Append the query string *?name=<yourname>* to this URL and execute the request.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Task - runFunctionsH + [ ] [ ] ^ X
[26-Apr-19 19:40:50] Starting language worker process:node --inspect=5858 "C:\Users\b.cataldo.da.silva\AppData\Roaming\npm\node_modules\azure-functions-core-tools\bin\workers\node\dist/src/nodejsWorker.js" --host 127.0.0.1 --port 56598 --workerId 16a93557-8ff2-4544-873b-c95394dc23a5 --requestId c4effe91-be28-454b-a4ac-e1fb5dd1cc94 --grpcMaxMessageLength 134217728
[26-Apr-19 19:40:50] node process with Id=14944 started
[26-Apr-19 19:40:50] Generating 1 job function(s)
[26-Apr-19 19:40:51] Found the following functions:
[26-Apr-19 19:40:51] Host.Functions.HttpTriggerJS
[26-Apr-19 19:40:51]
[26-Apr-19 19:40:51] Host initialized (1989ms)
[26-Apr-19 19:40:51] Host started (2014ms)
[26-Apr-19 19:40:51] Job host started
[26-Apr-19 19:40:51] Debugger listening on ws://127.0.0.1:5858/1fd12ced-4f4b-4091-ac44-aafd7fa6fc37
[26-Apr-19 19:40:51] For help, see: https://nodejs.org/en/docs/inspector
Hosting environment: Production
Content root path: C:\_avanade_softwares\azure\serverless-lab-local
Now listening on: http://0.0.0.0:7071
Application started. Press Ctrl+C to shut down.

Http Functions:

    HttpTriggerJS: [GET,POST] http://localhost:7071/api/HttpTriggerJS

[26-Apr-19 19:40:51] Worker 16a93557-8ff2-4544-873b-c95394dc23a5 connecting on 127.0.0.1:56598
[26-Apr-19 19:40:52] Adding dependency download request to node language worker
[26-Apr-19 19:40:56] Host lock lease acquired by instance ID '000000000000000000000000C20E6189'.
```

# Let's ride!

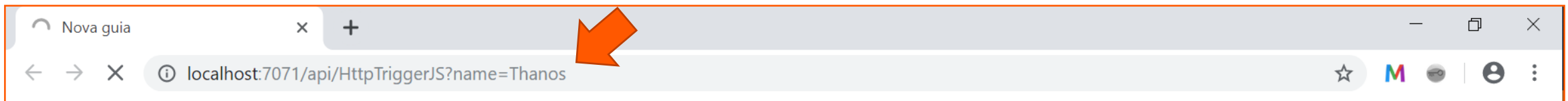
In the **Terminal** panel, copy the URL endpoint of your HTTP-triggered function.

Paste the URL for the HTTP request into your browser's address bar. Append the query string *?name=<yourname>* to this URL and execute the request.



```
1 module.exports = async function (context, req) {
2     context.log('JavaScript HTTP trigger function processed a request.');
```

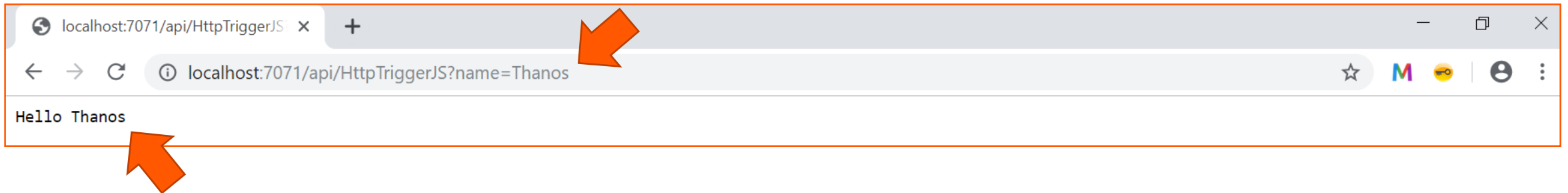
```
4     if (req.query.name || (req.body && req.body.name)) {
5         context.res = {
6             // status: 200, /* Defaults to 200 */
7             body: "Hello " + (req.query.name || req.body.name)
8         };
9     }
10    else {
11        context.res = {
12            status: 400,
13            body: "Please pass a name on the query string or in the request body"
14        };
15    }
16};
```



# Let's ride!

When you continue the execution, the following shows the response in the browser to the GET request.

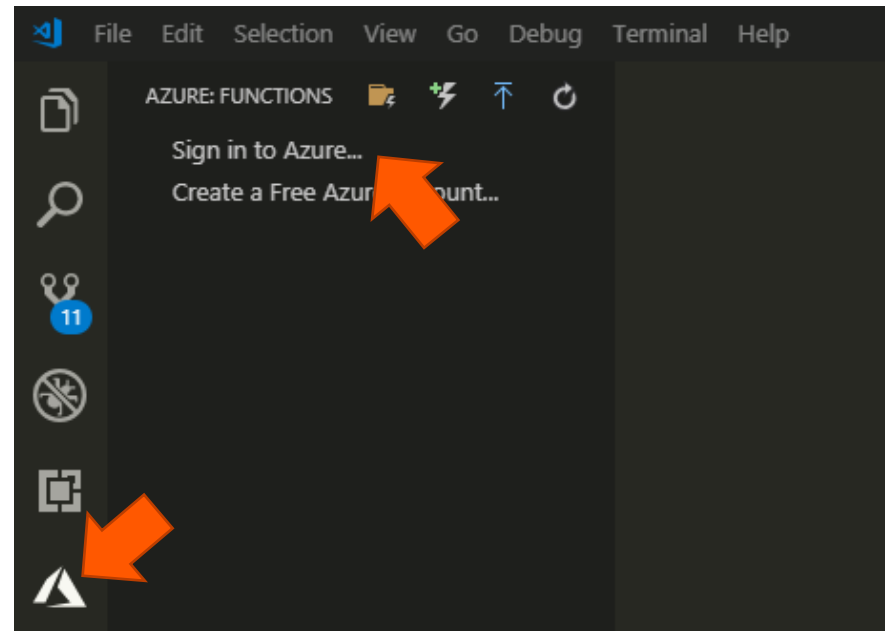
After you've verified that the function runs correctly on your local computer, it's time to publish the project to Azure.



# Sign in to Azure

Before you can publish your app, you must sign in to Azure.

In the *Azure: Functions View*, choose *Sign in to Azure....* If you don't already have one, you can Create a free Azure account.

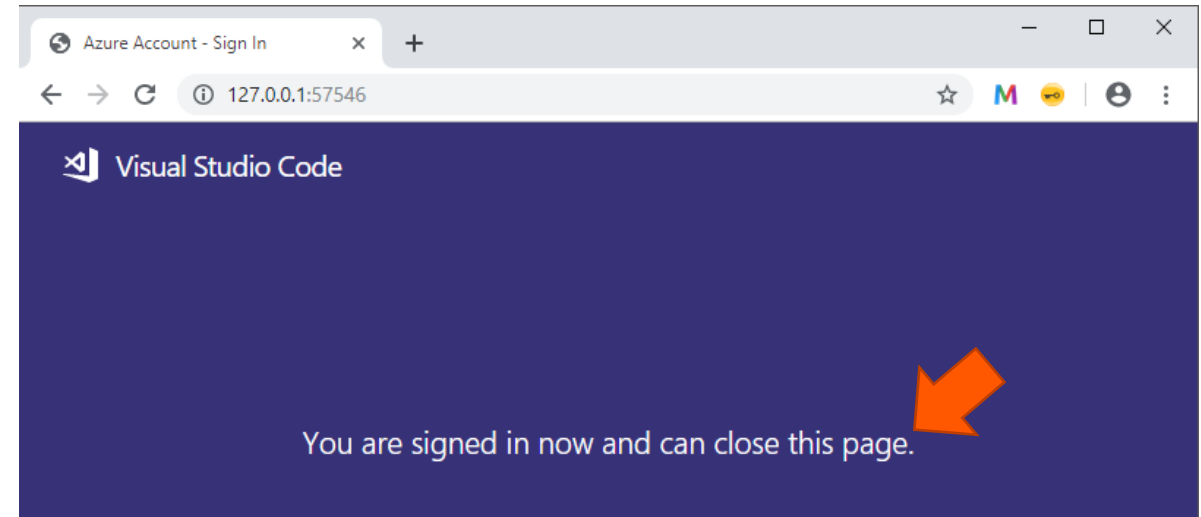


# Sign in to Azure

When prompted, select Copy&Open, or copy the displayed code and open <https://aka.ms/devicelogin> in your browser.

Paste the copied code in the *Device Login* page, verify the sign in for *Visual Studio Code*, then select *Continue*.

Complete the sign in using your *Azure account credentials*. After you have successfully signed in, you can close the browser.

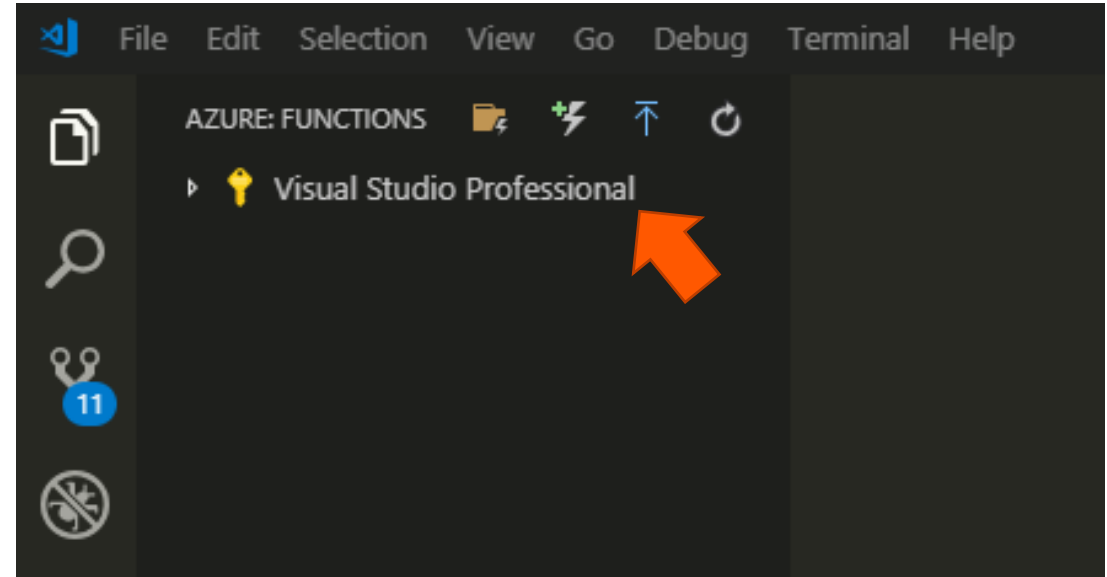


# Sign in to Azure

When prompted, select Copy&Open, or copy the displayed code and open <https://aka.ms/devicelogin> in your browser.

Paste the copied code in the *Device Login* page, verify the sign in for *Visual Studio Code*, then select *Continue*.

Complete the sign in using your *Azure account credentials*. After you have successfully signed in, you can close the browser.

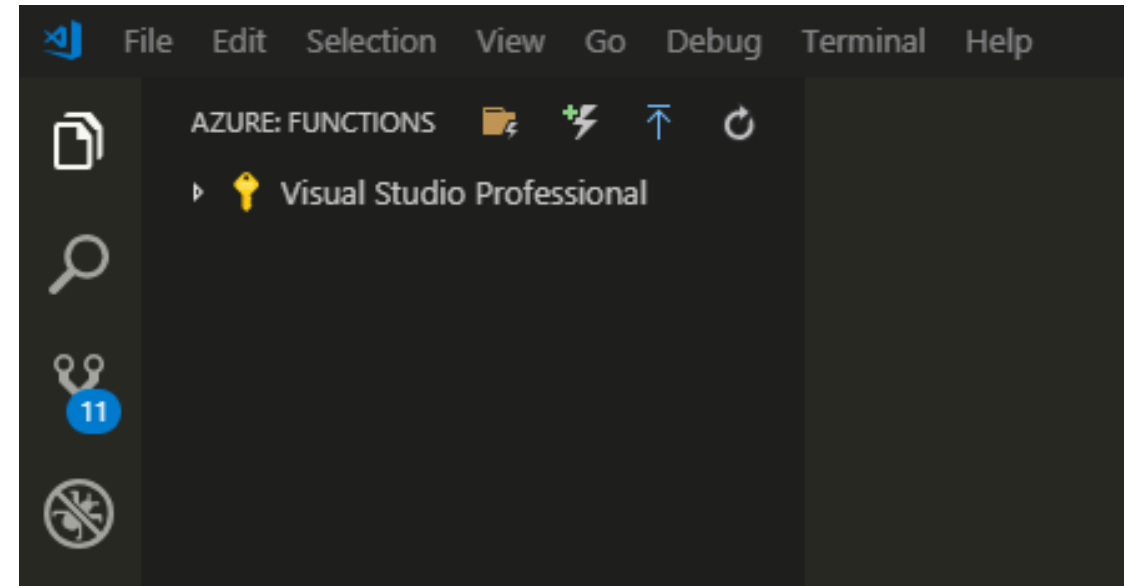


# Publish the project to Azure | VS Code

If not signed-in, you are prompted to Sign in to Azure.

You can also Create a free Azure account.

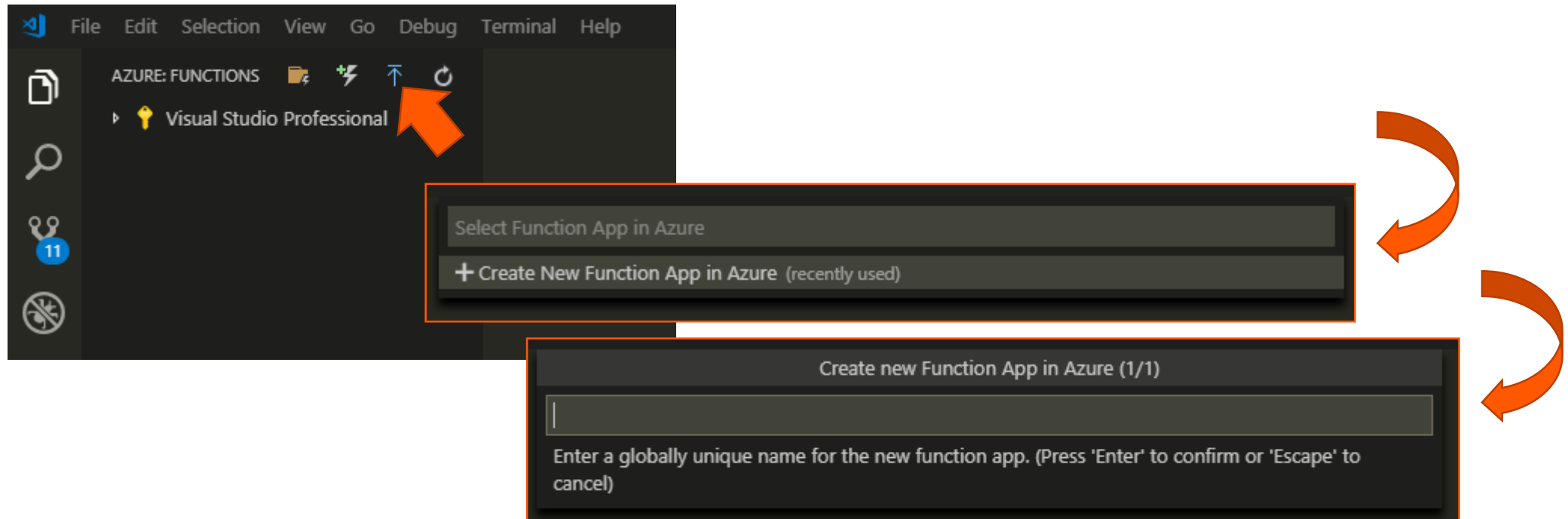
After successful sign in from the browser, go back to Visual Studio Code.



# Publish the project to Azure | VS Code

In the *Azure: Functions View*, select the Deploy to Function App icon.

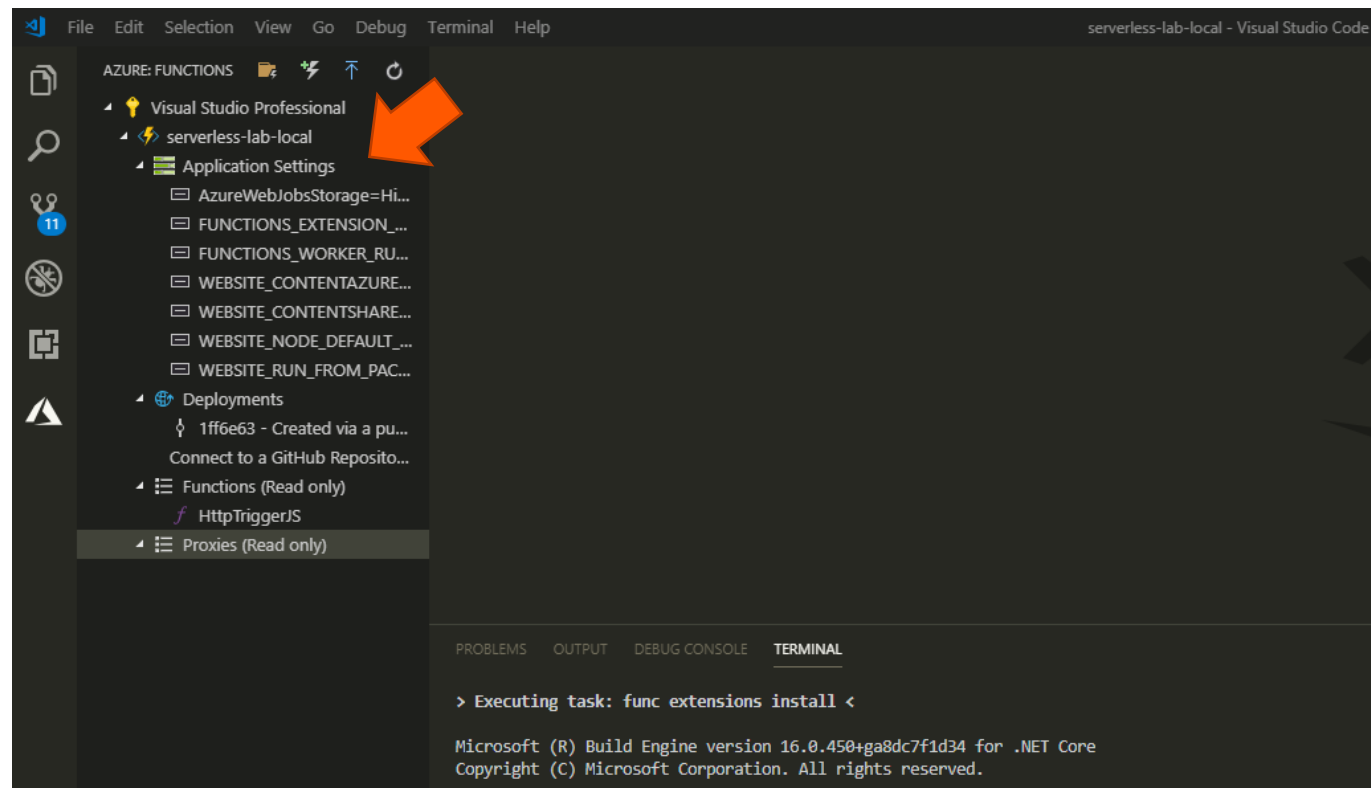
Type a globally unique name that identifies your function app and press Enter. Valid characters for a function app name are a-z, 0-9, and -.





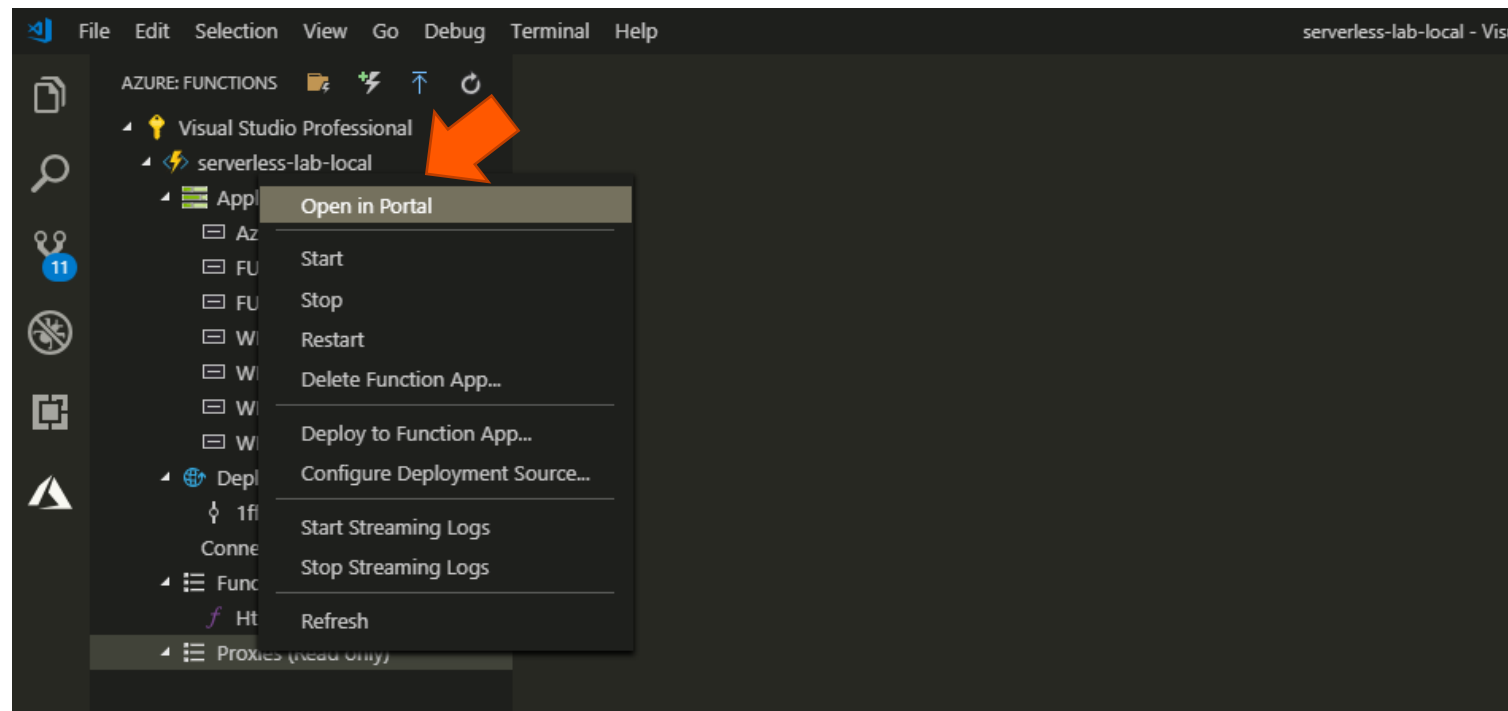
# Publish the project to Azure | VS Code

The result can be viewed like image by clicking the *Azure: Function View*.



# Publish the project to Azure | VS Code

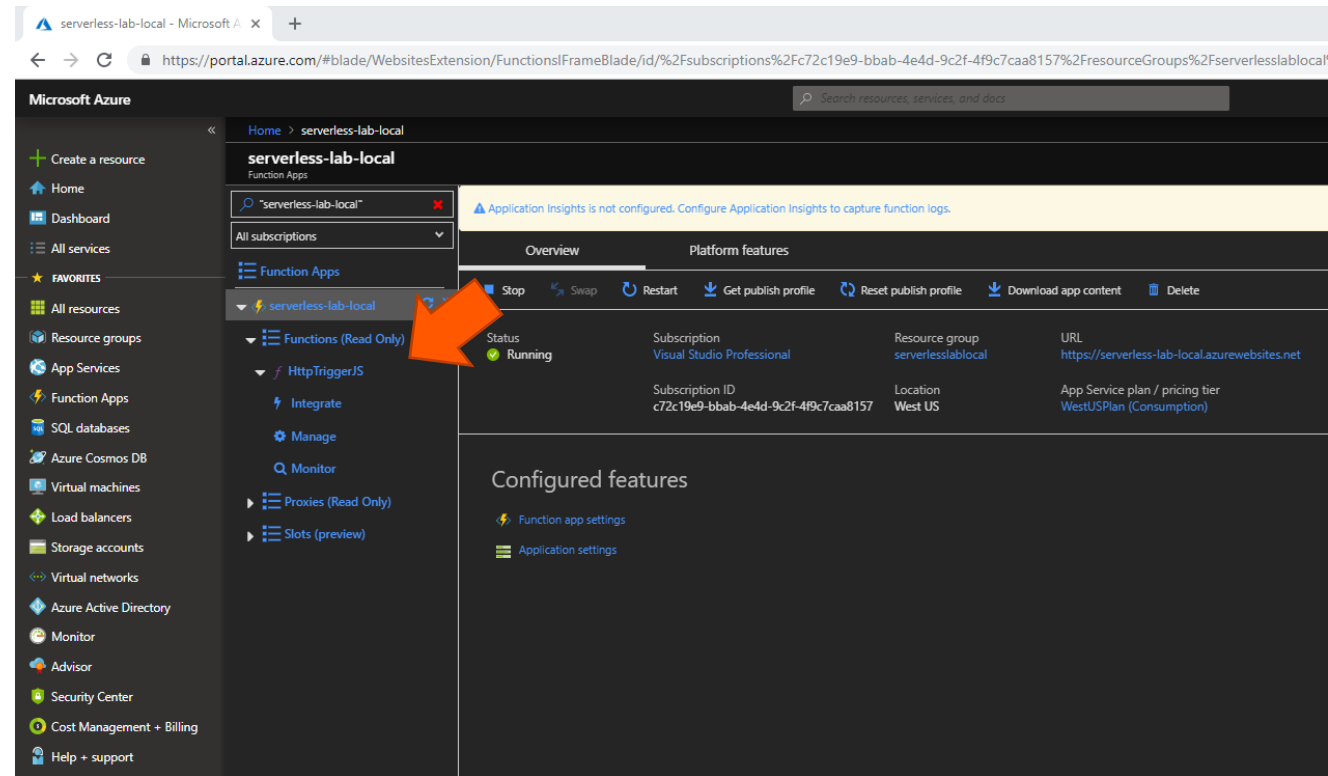
Right-click mouse button in *"serverless-lab-local"* to open the *Azure Function* menu.  
In the list, select *Open in Portal* option.



# Publish the project to Azure | VS Code

When the site loads, you can view your Azure Function publication on the Azure Platform.

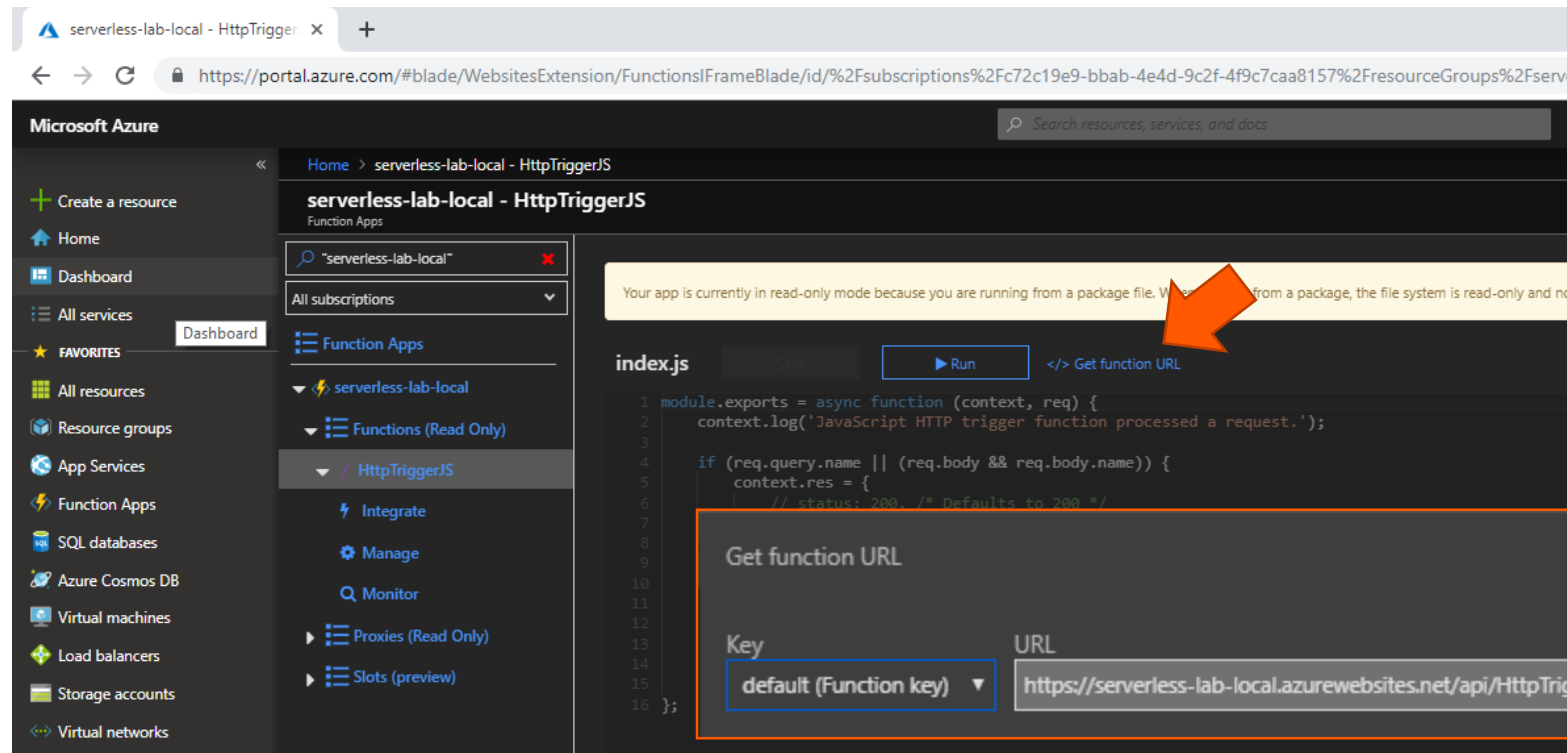
Select the *Functions* options and after the *HttpTriggerJS* item in the list to view the image below.



# Publish the project to Azure | VS Code

Click the "</> Get function URL" link like the image below.

Using the button to *copy* the URL.



# Publish the project to Azure | VS Code

Paste the URL for the HTTP request into your browser's address bar. Append the query string *?name=<yourname>* to this URL and execute the request.

The following shows the response in the browser to the *GET* request.

