



Lab 8: Implement Azure Functions

At the end of each lab, any resources you created in your account will be preserved. Some Azure resources, such as VM instances, may be automatically shut down, while other resources, such as storage services will be left running. Keep in mind that some Azure features cannot be stopped and can still incur charges (i.e. Azure Bastion). To minimize your costs, delete all resources and recreate them as needed to test your work during a session.

The screenshot shows the 'Azure for Students' subscription page. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, and Events. The main content area has a top bar with a search box and action links: Upgrade, Cancel subscription, Rename, Change directory, Transfer billing ownership, and Feedback. Below this is a warning banner about checking remaining credit. The 'Essentials' section displays subscription details in two columns.

Essentials	
Subscription ID	[Redacted]
Subscription name	: Azure for Students
Directory	: Seneca (seneca.onmicrosoft.com)
Current billing period	: 9/13/2021-10/12/2021
My role	: Account admin
Currency	: CAD
Offer	: Azure for Students
Status	: Active
Offer ID	: MS-AZR-0170P
Secure score	: Not available

Reference: [AZ-900T0X-MICROSOFTAZUREFUNDAMENTALS](#)

08 - Implement Azure Functions

In this walkthrough, we will create a Function App to display a Hello message when there is an HTTP request.

Task 1: Create a Function app (5 min)

In this task, we will create a Function app.

1. Sign in to the [Azure portal](#).
2. In the **Search resources, services, and docs** text box at the top of the portal, search for and select **Function App** and then, from the **Function App** blade, click + **Add**.
3. On the **Basic** tab of the **Function App** blade, specify the following settings (replace **xxxx** in the name of the function with letters and digits such that the name is globally unique and leave all other settings with their default values):

Settings	Value
Subscription	the name of your Azure subscription
Resource group	the name of a new resource group myRGFunction
Function App name	<studentID>-function-xxxx (example: dtrinh1-function-1234)
Publish	Code

Settings	Value
Runtime stack	.NET Core
Version	3.1
Region	East US

4. **Note** - Remember to change the **xxxx** so that it makes a unique **Function App name**
5. Click **Review + Create** and, after successful validation, click **Create** to begin provisioning and deploying your new Azure Function App.
6. Wait for the notification that the resource has been created.
7. Navigate back to the **Function App** blade, click **Refresh** and verify that the newly created function app has the **Running** status.

Home > Function App

Function App

Default Directory

+ Add Manage view Refresh Export to CSV | Assign tags Start Restart Stop Delete

Filter by name... Subscription == all Resource group == all Location == all Add filter

Showing 1 to 1 of 1 records.

<input type="checkbox"/> Name	Status	Location	Pricing Tier	App Service Plan
<input type="checkbox"/> function-9007	Running	East US	Dynamic	ASP-myRGFunction-a50c


Task 2: Create a HTTP triggered function and test


In this task, we will use the Webhook + API function to display a message when there is an HTTP request.


1. On the **Function App** blade, click the newly created function app.
2. On the function app blade, in the **Functions** section, click **Functions** and then click **+ Add**.

function-9007 | Functions

App Service

 Diagnose and solve problems

 Security

 Events







Functions


 Functions

 App keys

 App files

 Proxies

 Add  Develop Locally  Refresh |  Enable  Disable  Delete

 Filter by name...

Name ↑↓

Trigger ↑↓

Status ↑↓

No results.

- On the **Templates** tab of the **New Function** blade, click **HTTP trigger**.

The screenshot displays the Azure Functions portal interface. On the left, the navigation pane shows the 'Functions' section highlighted. In the main area, the '+ Add' button is circled in red. To the right, the 'New Function' blade is open, showing a grid of function templates. The 'HTTP trigger' template is highlighted with a red border. Below it, other templates like 'Timer trigger', 'Azure Queue Storage trigger', 'Azure Service Bus Queue trigger', 'Azure Service Bus Topic', and 'Azure Blob Storage' are visible.

Home > Function-9007 | Functions

Function-9007 | Functions
App Service

Search (Ctrl+/) << **+ Add** </> Develop Locally Refresh | ✓ Enable ○ Disable

Filter by name...

Name ↑↓ Trigger ↑↓

No results.

New Function

Create a new function in this function app. Start by selecting a template below.

Templates Details

Search by template name

HTTP trigger
A function that will be run whenever it receives an HTTP request, responding based on data in the body or query string

Timer trigger
A function that will be run on a specified schedule

Azure Queue Storage trigger
A function that will be run whenever a message is added to a specified Azure Storage queue

Azure Service Bus Queue trigger
A function that will be run whenever a message is added to a specified Service Bus queue

Azure Service Bus Topic

Azure Blob Storage

4. On the **Details** tab of the **New Function** blade, accept the default **New Function** name and **Authorization level**, and then click **Create Function**.

Home > Function App > function-9007 | Functions

function-9007 | Functions

App Service

Search (Ctrl+/) <<

+ Add </> Develop Locally ↺ Refresh | ✓ Enable ⌵

Filter by name...

Name ↑↓	Trigger ↑↓
No results.	

New Function

Create a new function in this function app. Start by selecting a template below.

[Templates](#) [Details](#)

New Function*


HttpTrigger1

Authorization level* ⓘ

Function

Create Function

5. On the **HttpTrigger1** blade, in the **Developer** section, click **Code + Test**.
6. On the **HttpTrigger1 | Code + Test** blade, review the auto-generated code and note that the code is designed to run an HTTP request and log information. Also, notice the function returns a Hello message with a name.

 **HttpTrigger1 | Code + Test**
Function

[Overview](#)






Developer

[Code + Test](#)

[Integration](#)

[Monitor](#)

[Function Keys](#)

 Save  Discard  Refresh  Test  **Get function URL**

function-9007 \ HttpTrigger1 \

```
1  #r "Newtonsoft.Json"
2
3  using System.Net;
4  using Microsoft.AspNetCore.Mvc;
5  using Microsoft.Extensions.Primitives;
6  using Newtonsoft.Json;
7
8  public static async Task<IActionResult> Run(HttpRequest req, ILogger log)
9  {
10     log.LogInformation("C# HTTP trigger function processed a request.");
11
12     string name = req.Query["name"];
13
14     string requestBody = await new StreamReader(req.Body).ReadToEndAsync();
15     dynamic data = JsonConvert.DeserializeObject(requestBody);
16     name = name ?? data?.name;
17
18     return name != null
19         ? (ActionResult)new OkObjectResult($"Hello, {name}")
20         : new BadRequestObjectResult("Please pass a name on the query string or in the request body");
21 }
22
```

- Click **Get function URL** from the top section of function editor.
- Ensure that the value in the **Key** drop-down list is set to **default** and click **Copy** to copy the function URL.



HttpTrigger1 | Code + Test

Function

Search (Ctrl+/)



Save



Discard



Refresh



Test



Get function URL

Overview

Developer

Code + Test

Integration

Monitor

Function Keys

function-9007 \ HttpTrigger1 \ run.csx

```
1 #r "Newtonsoft.Json"
2
3 using System.Net;
4 using Microsoft.AspNetCore.Mvc;
5 using Microsoft.Extensions.Primitives;
6 using Newtonsoft.Json;
7
8 public static async Task<IActionResult> Run(HttpRequest req, ILogger log)
9 {
10     log.LogInformation("C# HTTP trigger function processed a request.");
11
12     string name = req.Query["name"];
13
14     string requestBody = await new StreamReader(req.Body).ReadToEndAsync();
15     dynamic data = JsonConvert.DeserializeObject(requestBody);
16     name = name ?? data?.name;
```

Get function URL

Key

default

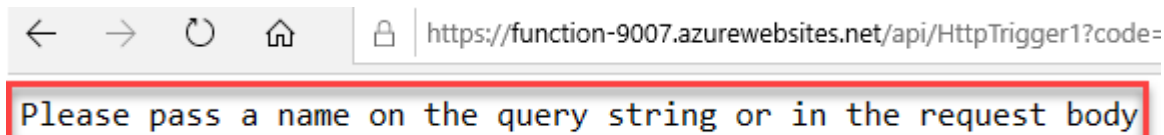
URL

https://function-9007.azurewebsites.n...



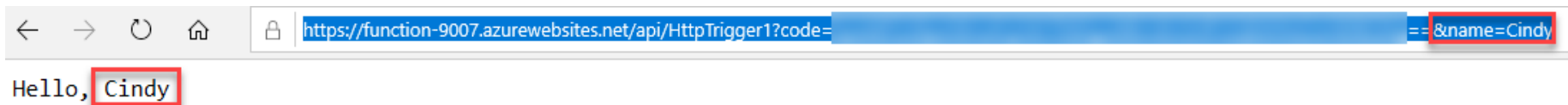
or in the request body");

9. Open a new browser tab and paste the copied function URL into your web browser's address bar. When the page is requested the function will run. Notice the returned message stating that the function requires a name in the request body.




10. Append **&name=yourname** to the end of the URL.

Note: Replace **yourname** with your first name. For example, if your name is Cindy, the final URL will resemble the following `https://azfuncxxx.azurewebsites.net/api/HttpTrigger1?code=X9xx9999xXXXXX9x9xxxXX==&name=cindy`



11. When your function runs, every invocation is traced. To view the traces in Azure portal, return to the **HttpTrigger1 | Code** + **Test** blade and click **Monitor**.

**HttpTrigger1 | Monitor**
Function

[Overview](#)

Developer

[Code + Test](#)

[Integration](#)

Monitor

[Function Keys](#)

[Invocations](#) [Logs](#)

Success Count
✔ 2
Last 30 Days

Error Count
✖ 0
Last 30 Days

Invocation Traces
The twenty most recent function invocation traces. For more advanced analysis, run the query in Application Insights.

[Run query in Application Insights](#) [Refresh](#)

Date (UTC)	Success	Result Code	Duration (ms)	Operation Id
2020-05-15 01:38:54.716	✔ Success	200	33	9a05b4de7403af448662232ab9df808e
2020-05-15 01:36:18.615	✔ Success	400	167	205f016ac879f54bbf5f5e0700915225

Congratulations! You have created a Function App to display a Hello message when there is an HTTP request.

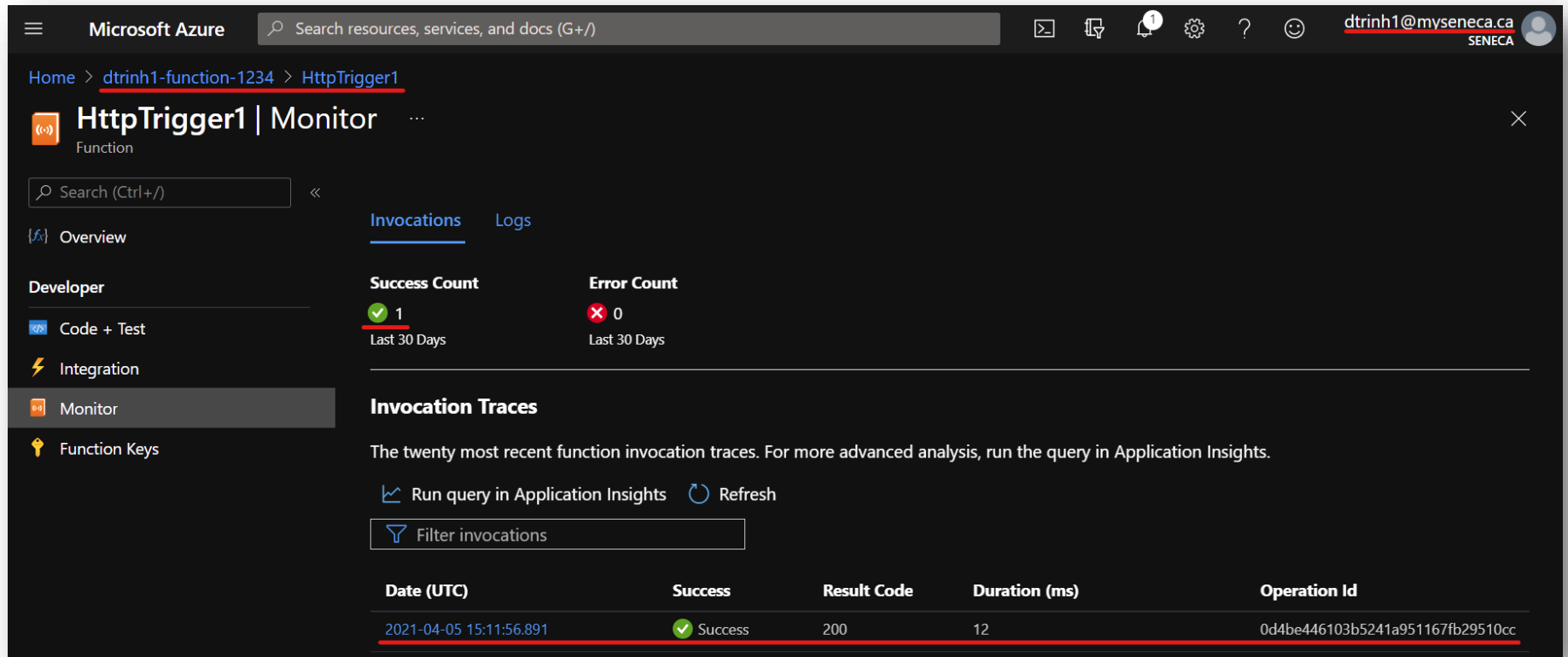
Note: To avoid additional costs, you can remove all resources in the resource group. Search for resource groups, click your resource group, and then delete the resources within the resource group. **DO NOT DELETE YOUR RESOURCE GROUP.**

Submission Requirements

Submit a screenshot with the following information:

Screenshot #1:

- The success count of the HttpTrigger from your function app
- The Azure Portal with your login ID



The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Microsoft Azure logo, a search bar, and user information (dtrinh1@myseneca.ca). The main content area displays the 'HttpTrigger1 | Monitor' page for the function app 'dtrinh1-function-1234'. The page shows the success count (1) and error count (0) for the function app. Below this, the 'Invocation Traces' section displays a table of recent function invocation traces.

Date (UTC)	Success	Result Code	Duration (ms)	Operation Id
2021-04-05 15:11:56.891	✓ Success	200	12	0d4be446103b5241a951167fb29510cc

Screenshot #2:

- Successful deletion of resources within resource group. **DO NOT DELETE YOUR RESOURCE GROUP!**

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Microsoft Azure logo, a search bar, and user information for 'dtrinh1@myseneca.ca'. The main content area is titled 'Resource groups' and shows the 'myRG' resource group. The left sidebar contains navigation links for 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Resource visualizer', 'Events', 'Settings', 'Deployments', 'Security', 'Policies', 'Properties', and 'Locks'. The 'Resources' tab is selected, showing a table with 0 records. The 'Essentials' section displays subscription information and deployment status.

Resource groups
Seneca (seneca.onmicrosoft.com)

myRG
Resource group

Essentials

Subscription (Move)
Azure for Students

Subscription ID
3e6685e5-073e-4397-8a34-b9022c3952d9

Deployments
No deployments

Location
East US

Resources Recommendations

Showing 0 to 0 of 0 records. ☐ Show hidden types

No grouping List view

Name	Type	Location
------	------	----------