



# Midterm Report

## Create serverless applications

10727211 林彥輝  
10727216 李品毅

# Intro

Model	1	2	3	4	5	6	7	8	9	10	11
Sandbox	Intro										Local
110-1-CS456L	Intro										Local
							VS (with lib)	Github	Node.js Core Tools VS Code (with lib)		Golang Azurite extension
Exercise		5 green circles					2 pink circles	2 pink circles	2 pink circles	2 pink circles	2 pink circles
Schedule	← 80min (Hands-on) →					← 10min (Intro) →					

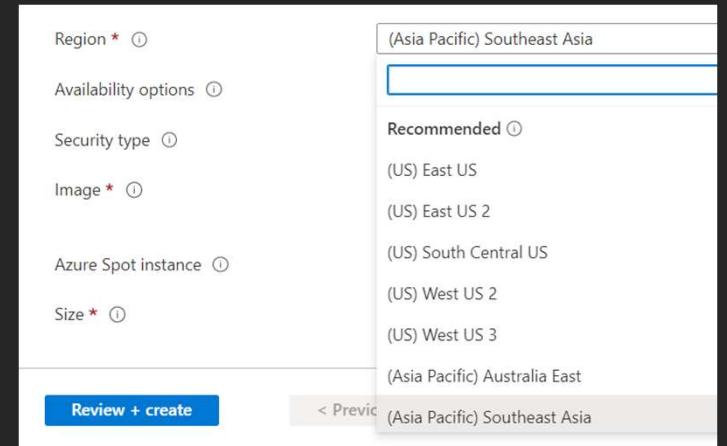
Azure Portal (110-1-CS456L) : <https://portal.azure.com/#home>

Microsoft Learn : <https://docs.microsoft.com/en-us/learn/patterns/create-serverless-applications/>

# Intro

- 本報告279頁，請把PPT載下來對照（台上只講大流程）

- Region （不要通通集中在Japan East）
- Resource Naming Rule （記得學號開頭）
- 備忘稿要看，裡面有提醒（有code、截圖提示！）
- Another Resource : [https://www.youtube.com/playlist?list=PLMXvlcTmSbxHVQ2pUfZkgyu\\_0mum7jiKH](https://www.youtube.com/playlist?list=PLMXvlcTmSbxHVQ2pUfZkgyu_0mum7jiKH)  
**Sandbox Demo Video**  
( Model 2 3 4 5 6 8 · ~103min )



01

// Choose the best Azure service to  
automate your business processes

# Azure Functions



<https://docs.microsoft.com/en-us/learn/modules/create-serverless-logic-with-azure-functions/2-decide-if-serverless-computing-is-right-for-your-business-need>

<https://docs.microsoft.com/en-us/learn/modules/build-serverless-api-with-functions-api-management/2-import-function-app-api-management>

## What is serverless compute?

Serverless compute can be thought of as a **function as a service (FaaS)**, or a microservice that is hosted on a cloud platform. You don't have to manually provision or scale infrastructure. The cloud provider manages infrastructure. Your app is automatically scaled out or down depending on load.

## What is Azure Functions?

Azure Functions is a service that enables serverless architectures in Azure. You can write functions, without worrying about the supporting infrastructure, in many different languages, including C#, Java, JavaScript, PowerShell, and Python. You can also use libraries from NuGet and the Node Package Manager (NPM), and authenticate users with the OAuth standard from providers such as Active Directory, Facebook, Google, and Microsoft Account.

## Benefits

1. Avoids over-allocation of infrastructure
2. Stateless logic
3. Event driven
4. Functions can be used in traditional compute environments

Azure Functions



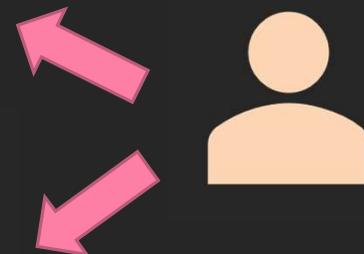
SaaS

PaaS

IaaS

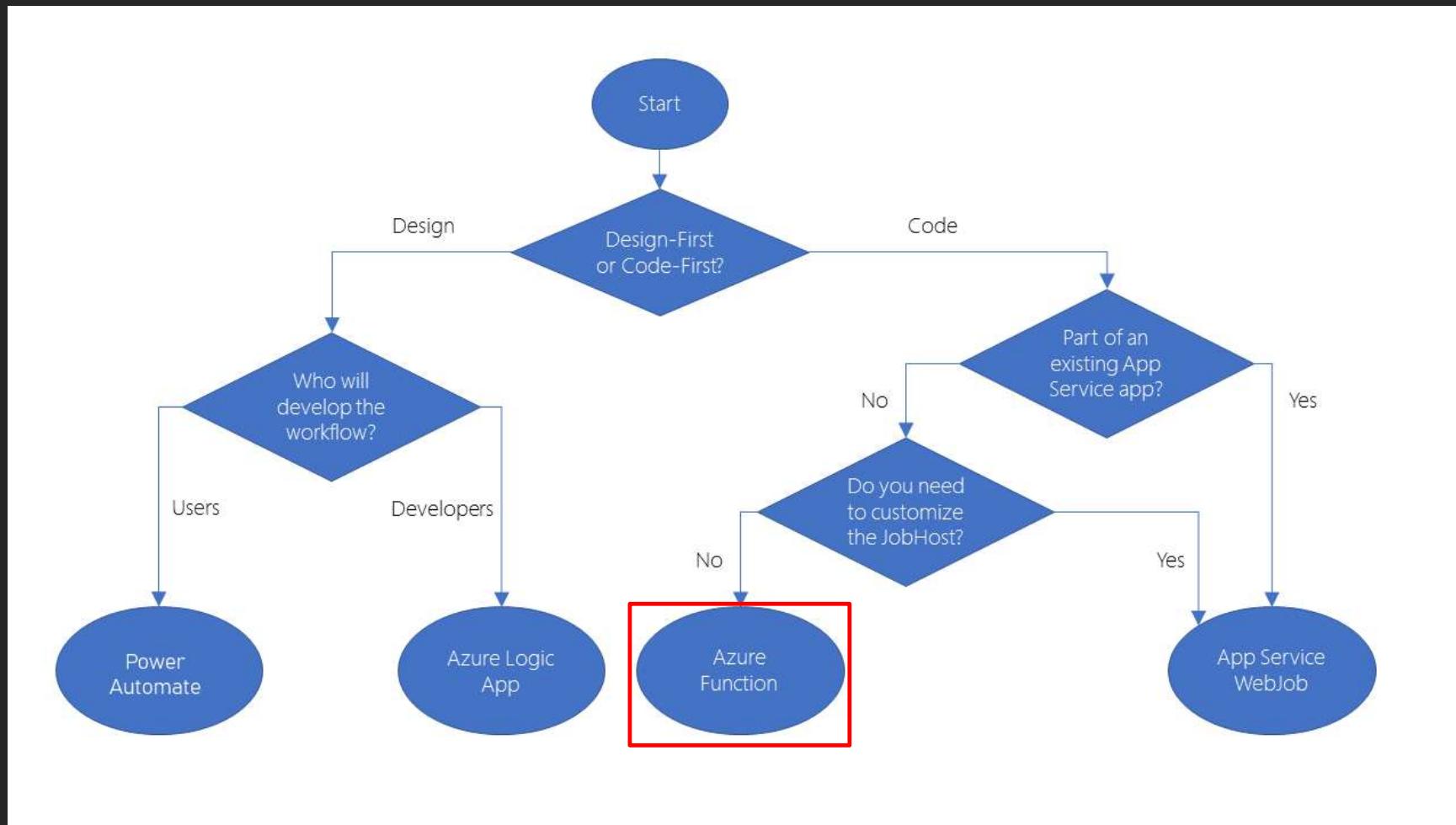
A{  
FaaS

Function url ↗



# Choose a Azure Serverless Service

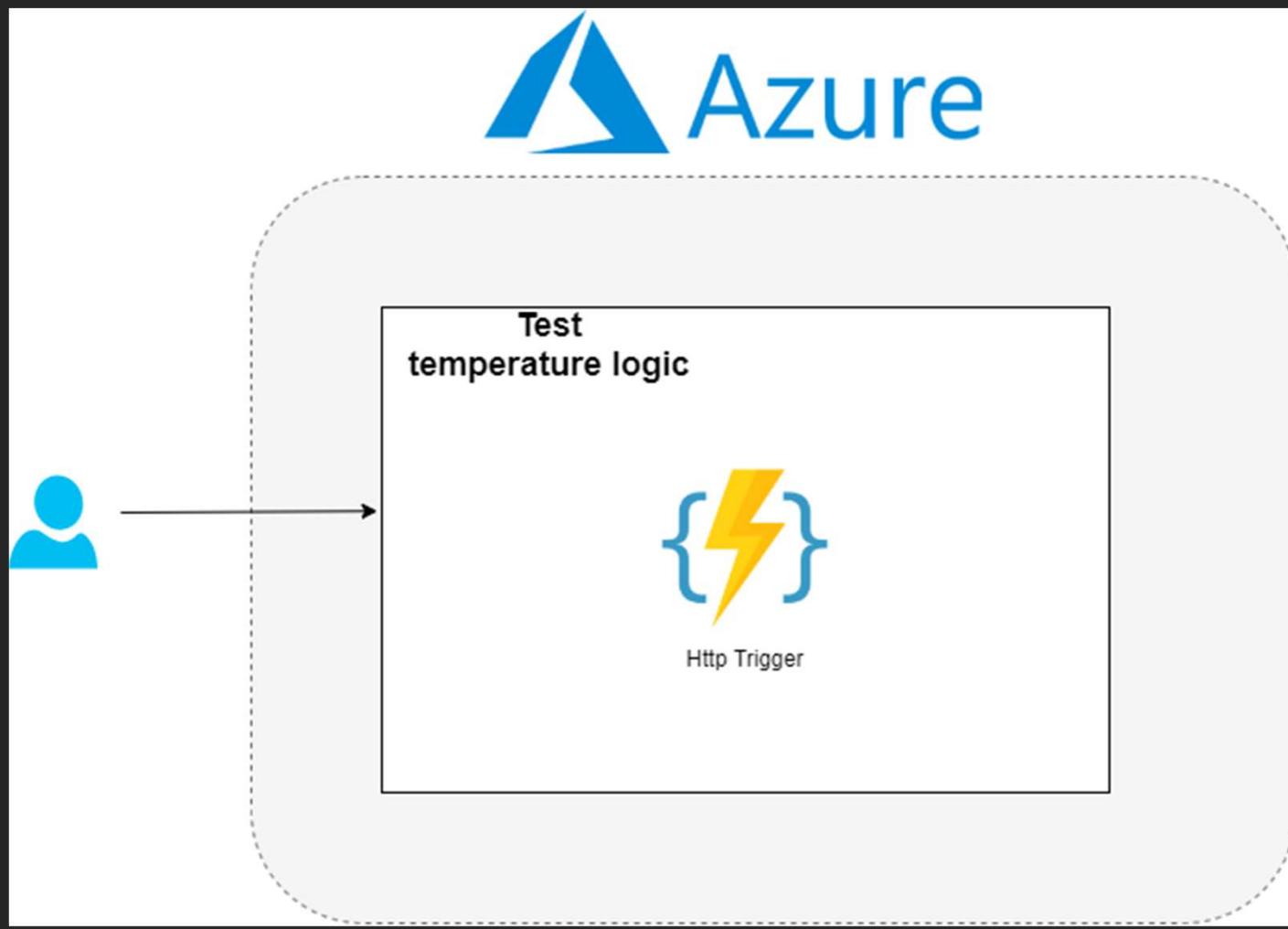
<https://docs.microsoft.com/en-us/learn/modules/choose-azure-service-to-integrate-and-automate-business-processes/3-analyze-the-decision-criteria>



02

// Create serverless logic with Azure  
Functions

# Abstract



# Create Azure Function in the Azure Portal

The screenshot shows the Microsoft Azure Portal interface. At the top, there is a navigation bar with various links and a search bar. On the right side of the header, there is a user profile section with a red box highlighting it. Below the header, the main content area is titled "Azure services". A large button labeled "Create a resource" with a plus sign icon is highlighted with a yellow box. To its right are icons for Container registries, App Services, Container instances, Resource groups, All resources, Quickstart Center, Virtual machines, Storage accounts, and More services. Below this section is a "Recent resources" table:

Name	Type	Last Viewed
110-1-CS456L	Resource group	3 minutes ago
10727211storage	Storage account	2 hours ago
110-1-CS456L-雲端計算平台實務-鍾武君	Resource group	3 days ago

Below the table is a "See all" link. Further down, there is a "Navigate" section with links for Subscriptions, Resource groups, All resources, and Dashboard. At the bottom, there is a "Tools" section with links for Microsoft Learn, Azure Monitor, Security Center, and Cost Management. The taskbar at the bottom of the screen shows various pinned application icons.

A Create a resource - Microsoft A x +

portal.azure.com/#create/hub

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+/)

Home > Create a resource ...

Get started

Recently created

Categories

- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers
- Databases
- Developer Tools
- DevOps
- Identity
- Integration
- Internet of Things
- IT & Management Tools
- Media
- Migration
- Mixed Reality

Search services and marketplace

Getting Started? Try our Quickstart center

Popular products See more in Marketplace

 Virtual machine  
Create | Learn more

 Virtual machine scale set  
Create | Learn more

 Kubernetes Service  
Create | Docs | MS Learn

 Function App  
Create | Docs

 Datadog  
Set up + subscribe | Learn more

 Ubuntu Server 20.04 LTS  
Create | Learn more

 Red Hat Enterprise Linux 8.2 (LVM)  
Create | Learn more

 CentOS-based 8.2

15°C 英 2021/11/23

A Create Function App - Microsoft Azure +

portal.azure.com/#create/Microsoft.FunctionApp

應用程式 中原e點靈\_e17 i-learning 中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+)

Home > Create a resource >

## Create Function App

or resources. Functions lets you execute your code in a serverless environment without having to first create a VM or publish a web application.

### Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ① 中原大學

Resource Group \* ① 110-1-CS456L

Create new

### Instance Details

Function App name \* 10727211-FunctionApp .azurewebsites.net

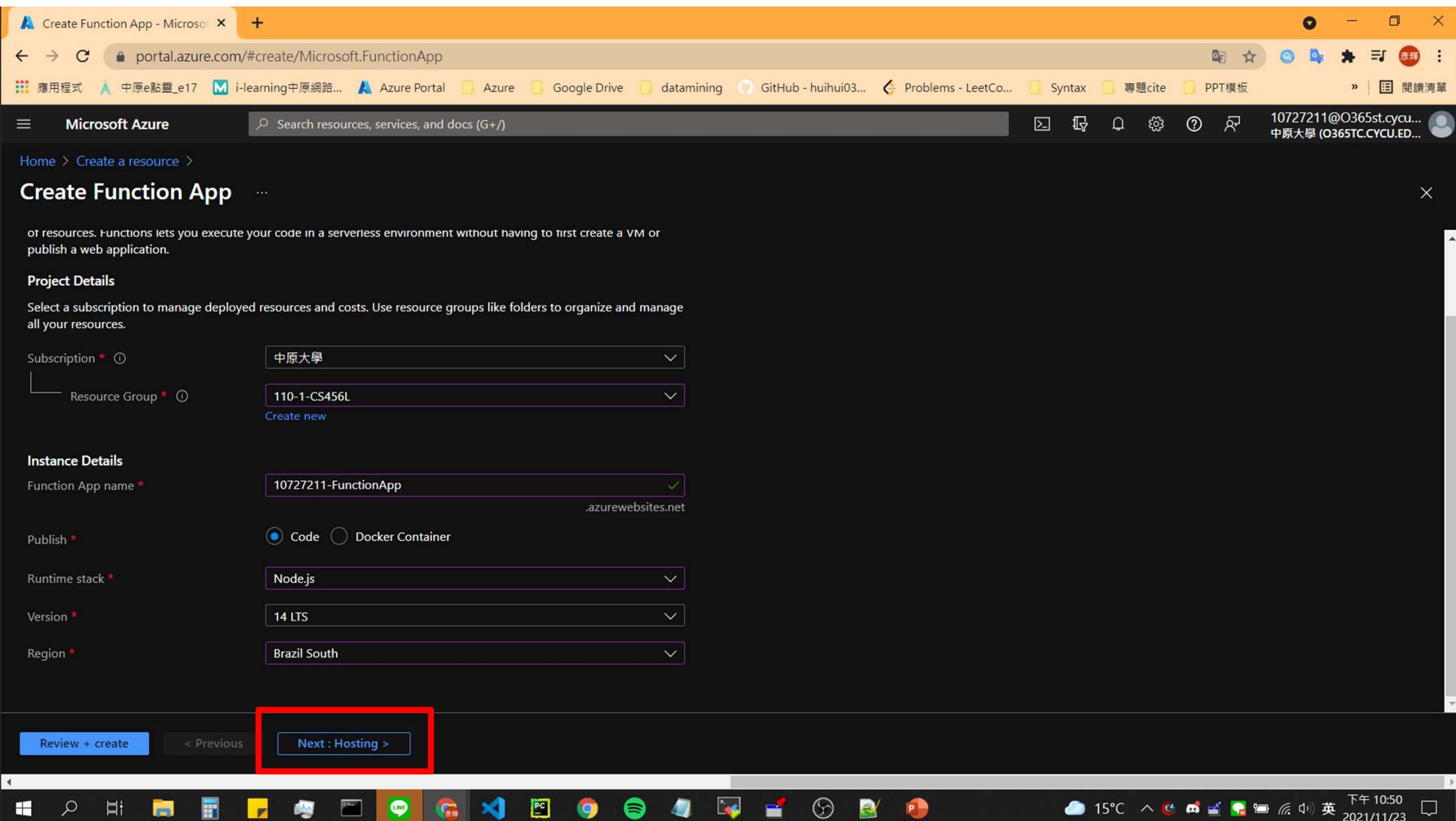
Publish \*  Code  Docker Container

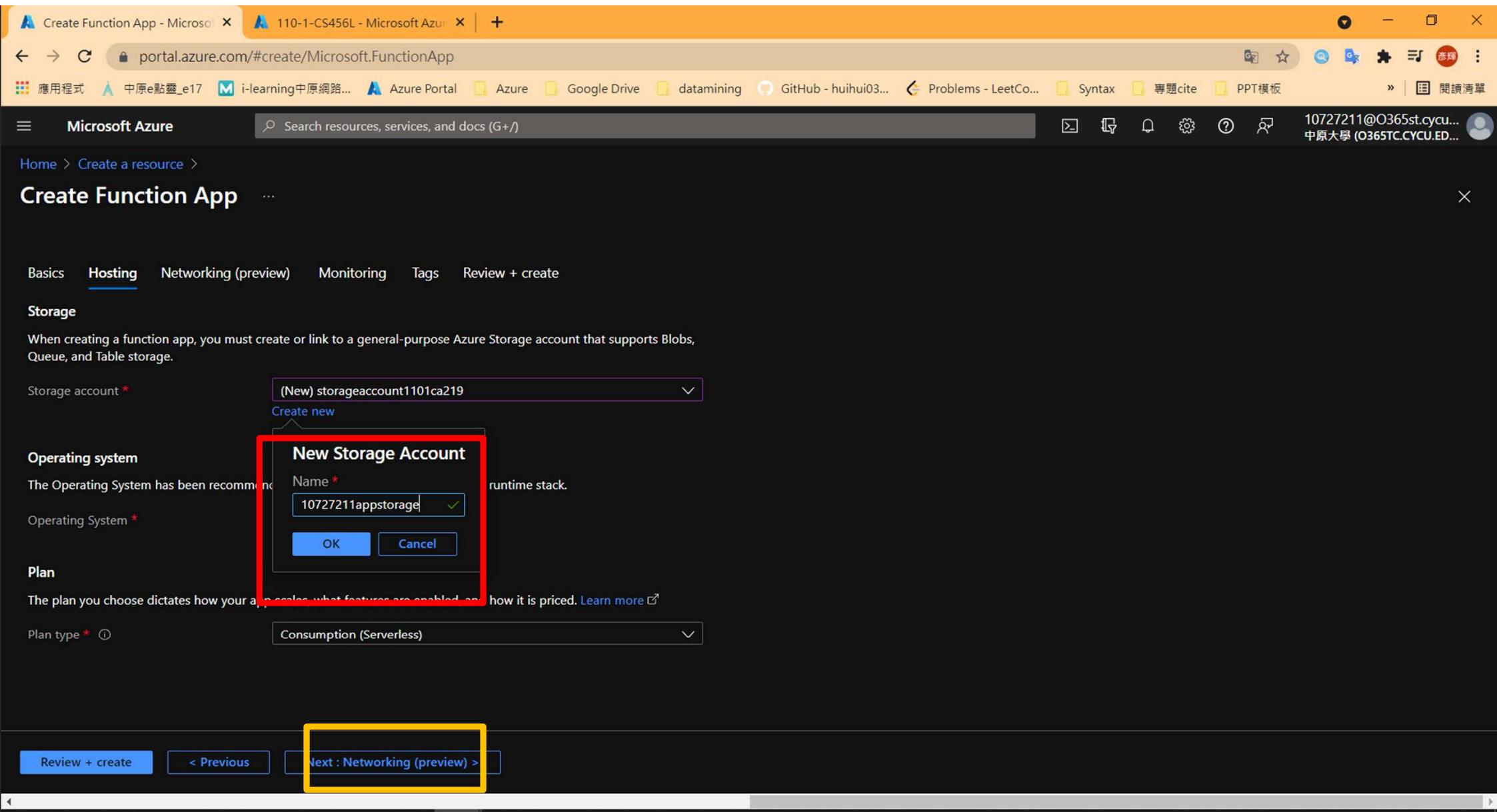
Runtime stack \* Node.js

Version \* 14 LTS

Region \* Brazil South

Review + create < Previous Next : Hosting >





A Create Function App - Microsoft Azure | A 110-1-CS456L - Microsoft Azure | +

portal.azure.com/#create/Microsoft.FunctionApp

應用程式 中原e點靈\_e17 i-learning 中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+/)

Home > Create a resource >

## Create Function App

Basics Hosting Networking (preview) Monitoring Tags Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups.

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name ⓘ	Value ⓘ	Resource
10727211	:	3 selected
	:	3 selected

Review + create < Previous Next : Review + create >

Review + create

15°C 英 2021/11/23

A 10727211-FunctionApp - Microsoft Azure A 110-1-CS456L - Microsoft Azure +

portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourcegroups/110-1-CS456L/providers/Microsoft.Web/sites/10727211-FunctionApp

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCode... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+/)

Home > 10727211-FunctionApp Function App

Search (Ctrl+ /) Browse Refresh Stop Restart Swap Get publish profile Reset publish profile Download app content Delete Send us your feedback

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Security Events (preview)

Resource group (Move) : 110-1-CS456L Status : Running Location : Brazil South Subscription (Move) : 中原大學 Subscription ID : ba9f5e30-1488-49db-bae0-9026dacc11b0 Tags (Edit) : 10727211 URL : https://10727211-functionapp.azurewebsites.net Operating System : Windows App Service Plan : ASP-1101CS456L-a252 (Y1: 0) Properties : See More Runtime version : 4.0.1.16815

JSON View

Essentials

Metrics Features (9) Notifications (1) Quickstart

Memory working set

100B
90B
80B
70B
60B
50B
40B

Function Execution Count

100
90
80
70
60
50
40

15°C 英 2021/11/23

# Add a function to your function app

The screenshot shows the Microsoft Azure portal interface. The title bar has two tabs: '10727211-FunctionApp - Micro' and '110-1-CS456L - Microsoft Azur'. The address bar shows the URL: 'portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourcegroups/110-1-CS456L/providers/Microsoft.Web/sites/10727211-FunctionApp'. The top navigation bar includes links for 'Search resources, services, and docs (G+)', 'Home', 'Microsoft Azure', and user information.

**Overview**

Click here to access Application Insights for monitoring and profiling for your app. [JSON View](#)

**Essentials**

Resource group (Move)	: 110-1-CS456L	URL	: <a href="https://10727211-functionapp.azurewebsites.net">https://10727211-functionapp.azurewebsites.net</a>
Status	: Running	Operating System	: Windows
Location	: Brazil South	App Service Plan	: ASP-1101CS456L-a252 (Y1: 0)
Subscription (Move)	: 中原大學	Properties	: <a href="#">See More</a>
Subscription ID	: ba9f5e30-1488-49db-bae0-9026dacc11b0	Runtime version	: 4.0.1.16815
Tags (Edit)	: 10727211 :		

**Metrics** [Metrics](#) Features (9) Notifications (1) Quickstart

**Memory working set**

Value
100B
90B
80B
70B
60B
50B
40B

**Function Execution Count**

Value
100
90
80
70
60
50
40

Windows taskbar icons include: File Explorer, Task View, Start, Search, Edge, Google Chrome, Spotify, FileZilla, Visual Studio Code, and others. System tray shows: Cloud, 15°C, battery, signal, volume, language (English), date (2021/11/23), and a notification icon.

A Create function - Microsoft Azure +

portal.azure.com/#@O365tc.cygu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourcegroups/110-1-CS456L/providers/Microsoft.Web/sites/

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+/)

Home > 10727211-FunctionApp

## 10727211-FunctionApp | Functions

Function App

+ Create Refresh Delete

Search (Ctrl+ /) Filter

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Security Events (preview)

Functions

Functions App keys App files Proxies

Deployment

Deployment slots Deployment Center

Settings

### Create function

Use a template to create a function. Triggers describe the type of events that invoke your functions. [Learn more](#)

Template	Description
HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
Timer trigger	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 儲存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
Azure Blob Storage trigger	Blob 每次新增至指定的容器時，將會執行的函式
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

No results.

Name ↑

Template details

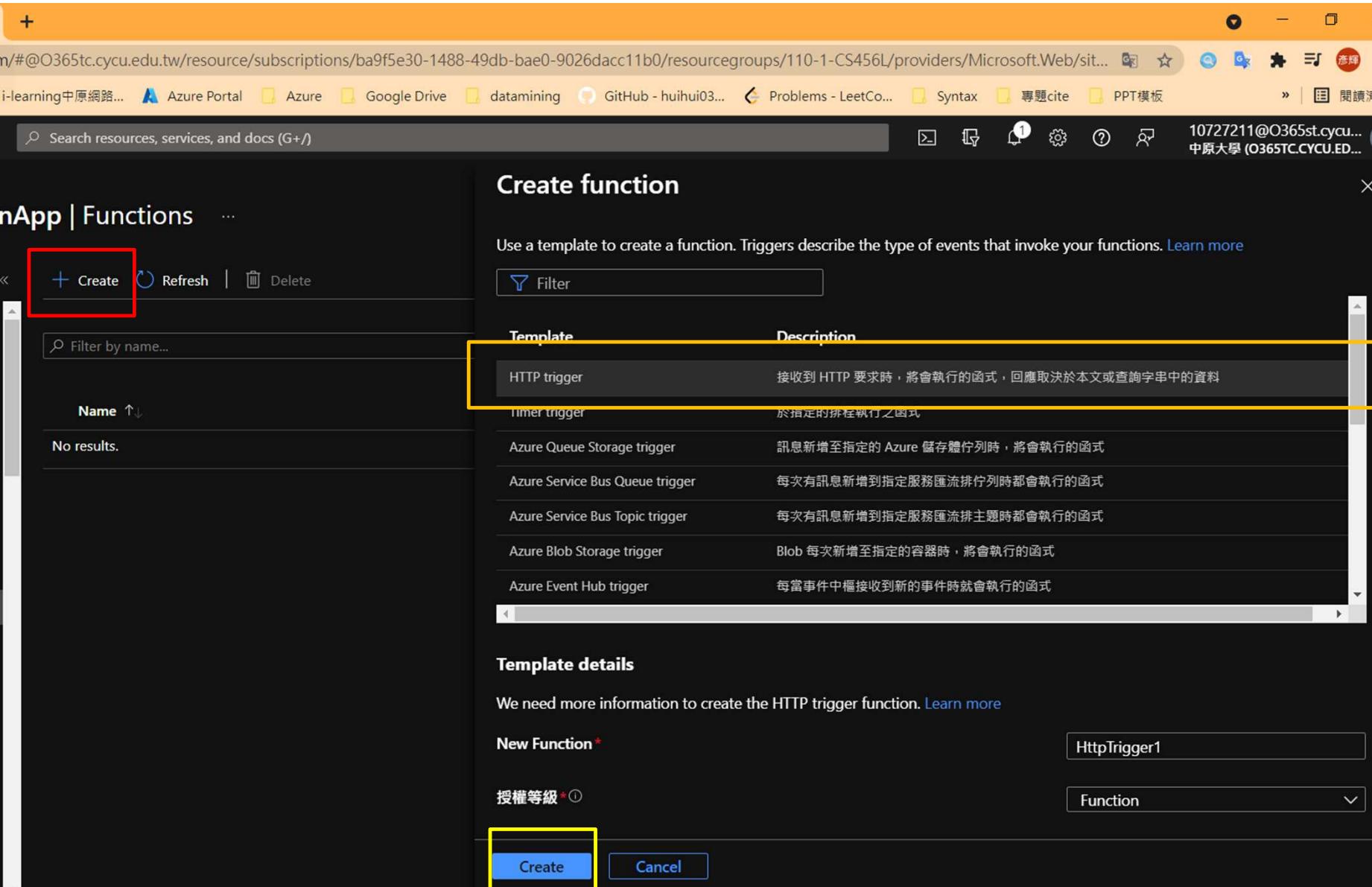
We need more information to create the HTTP trigger function. [Learn more](#)

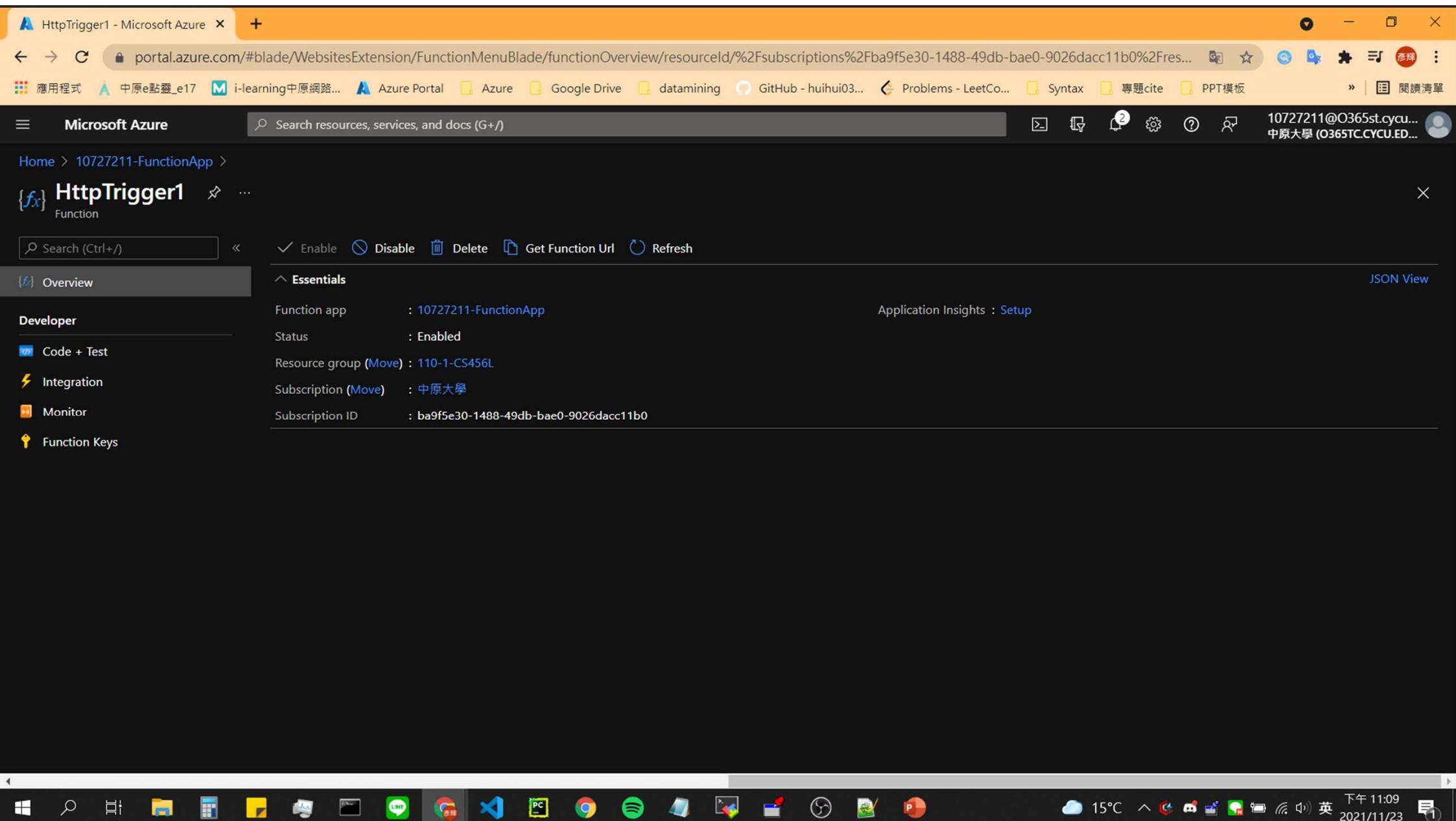
New Function\*

授權等級\*

Create Cancel

15°C 11:04 2021/11/23





# Get Function URL

The screenshot shows the Microsoft Azure portal interface. In the top navigation bar, there are three tabs: "HttpTrigger1 - Microsoft Azure", "練習 - 將邏輯新增至函數應用程... (未命名)", and "Exercise - Add logic to the func...". The main content area is titled "HttpTrigger1" under the "Function" category. On the left, a sidebar lists "Overview", "Developer", "Code + Test", "Integration", "Monitor", and "Function Keys". The "Developer" section is active, showing a search bar, "Enable", "Disable", "Delete", and "Get Function Url" buttons. The "Get Function Url" button is highlighted with a red box. A modal dialog box is displayed in the center, titled "Get Function Url". It contains a dropdown menu set to "default (function key)" and a URL field containing "https://10727211-functionapp.azurewebsites.net/api/HttpTrigger1?code=KIUjh7DBaezfoAALnHoSsOlQ...". Below the URL is an "OK" button. To the right of the URL field, a yellow box encloses the URL and is labeled "Store-1". At the bottom of the screen, the Windows taskbar shows various pinned icons and the system tray displays the date and time as "下午 11:18 2021/11/23".

# Secure HTTP triggers

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for 'HttpTrigger1 - Microsoft Azure', '練習 - 將邏輯新增至函數應用程...', and 'Exercise - Add logic to the func...'. Below the navigation bar, the address bar shows the URL: portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/functionKeys/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2Fresource... . The main content area is titled 'HttpTrigger1 | Function Keys' under the 'Function' category. On the left, a sidebar lists 'Overview', 'Developer' (with 'Code + Test', 'Integration', and 'Monitor' options), and 'Function Keys' (which is highlighted with a red box). The central area is titled 'Function Keys' and contains the following text: 'Function keys are scoped to this function and can be used to access this function.' Below this are two buttons: '+ New function key' and 'Hide values' (which is highlighted with a yellow box). A search bar labeled 'Search (Ctrl+ /)' and a refresh button are also present. A filter bar for 'Filter functions keys' is shown. A table lists one function key: 'Name' (default) and 'Value' (KIUjh7DBaezfoAALnHoSsOIQMSDUUBvDEBxDUNJZVguivXN8p2Hdg==). To the right of the value column are icons for 'Copy' (highlighted with a yellow box) and 'Renew key value'. The bottom right corner of the table cell containing the value is labeled 'Store-2' in green. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray.

# Test Function

```
curl --header "Content-Type: application/json" --header "x-functions-key: <your-function-key>"  
--request POST --data "{\"name\": \"Azure Function\"}" <your-https-url>
```



```
curl --header "Content-Type: application/json" --header "x-functions-key:  
KIUjh7DBaezfoAALnHoSsOlQMSDUUBvDEBcxDUNJZVguivXN8p2Hdg==" --request POST --data  
"{\"name\": \"Azure Function\"}" https://10727211-  
functionapp.azurewebsites.net/api/HttpTrigger1?code=KIUjh7DBaezfoAALnHoSsOlQMSDUUBvDEBc  
xDUNJZVguivXN8p2Hdg==
```

# Test Function - 1

The screenshot shows the Microsoft Azure Functions developer portal interface. At the top, there are three tabs: "HttpTrigger1 - Microsoft Azure", "練習 - 將邏輯新增至函數應用程...", and "Exercise - Add logic to the func...". The main content area is titled "HttpTrigger1 | Code + Test" and shows the "Code + Test" tab selected. On the left, a sidebar lists "Overview", "Developer" (with "Code + Test" selected), "Integration", "Monitor", and "Function Keys". The main workspace displays the function's code in "index.js":

```
1 module.exports = async function (context, req) {
2     context.log('JavaScript HTTP trigger function processed a request.');
3     const name = (req.query.name || (req.body && req.body.name));
4     const responseMessage = name;
5 }
```

Below the code, there is a "Logs" section with a "Filesystem Logs" dropdown and a "Log Level" dropdown set to "Information". The logs show the following entries:

```
Connected!
2021-11-23T15:26:15 Welcome, you are now connected to log-streaming service. The default timeout is 2 hours. Change the timeout with the App Setting SCM_LOGSTREAM_TIMEOUT (in seconds).
2021-11-23T15:26:26.976 [Information] Executing 'Functions.HttpTrigger1' (Reason='This function was programmatically called via the host APIs.', Id=59b2585c-578e-425b-8ff7-566b9d1b60d9)
2021-11-23T15:26:26.980 [Information] JavaScript HTTP trigger function processed a request.
2021-11-23T15:26:26.980 [Information] Executed 'Functions.HttpTrigger1' (Succeeded, Id=59b2585c-578e-425b-8ff7-566b9d1b60d9, Duration=4ms)
```

A red box highlights the "Code + Test" tab in the sidebar and the "Logs" section in the main content area.

The screenshot shows a terminal window with a yellow border. The command entered is:

```
azuser@Azure:~$ curl --header "Content-Type: application/json" --header "x-functions-key: KIUjh7DBaezfoAALnHoSs0lQMSDUUBvDEBcxDUNJZVguivXN8p2Hdg==" --request POST --data '{"name": "Azure Function"}' https://10727211-functionapp.azurewebsites.net/api/HttpTrigger1?code=KIUjh7DBaezfoAALnHoSs0lQMSDUUBvDEBcxDUNJZVguivXN8p2Hdg==
```

The output of the command is:

```
Hello, Azure Function. This HTTP triggered function executed successfully.azuser@Azure:~$
```

The terminal window has a dark theme and includes icons for file operations like copy, paste, and close.

# Test Function - 2

The screenshot shows the Microsoft Azure Functions portal interface. The top navigation bar includes tabs for "HttpTrigger1 - Microsoft Azure", "練習 - 將邏輯新增至函數應用程...", and "Exercise - Add logic to the func...". The main title is "HttpTrigger1 | Code + Test". The left sidebar has sections for "Overview", "Developer", "Code + Test" (which is selected), "Integration", "Monitor", and "Function Keys". The code editor displays the following JavaScript code:

```
1 module.exports = async function (context, req) {
2     context.log('JavaScript HTTP trigger function processed a request.')
3
4     const name = (req.query.name || (req.body && req.body.name));
5     const responseMessage = name
6     ? `Hello, ${name}! This HTTP-triggered function executed successfully.`
7     : `This HTTP-triggered function executed successfully. Pass a name in the query or in the request body`
8
9     context.res = {
10         // status: 200, /* Defaults to 200 */
11         body: responseMessage
12     };
13 }
```

The "Test/Run" button in the toolbar is highlighted with a red box. The "Body" input field contains a curl command:

```
1 [
2     curl --header "Content-Type: application/json" --header "x-func"
3 ]
```

The "Run" button at the bottom of the test panel is highlighted with a yellow box.

# Test Function - 2

The screenshot shows the Microsoft Azure Functions test interface for an HttpTrigger1 function. The code editor displays the index.js file:

```
1 module.exports = async function(context, req) {
2     context.log('JavaScript HTTP trigger function processed a request.')
3
4     const name = (req.query.name || (req.body && req.body.name));
5     const responseMessage = name
6     ? `Hello, ${name}! This HTTP-triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response.
7     : "This HTTP-triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response.
8
9     context.res = {
10         // status: 200, /* Defaults to 200 */
11         body: responseMessage
12     };
13 }
```

The logs panel shows the following output:

```
Connected!
2021-11-23T15:33:15 No new trace in the past 2 min(s).
2021-11-23T15:33:37.262 [Information] Executing 'Functions.HttpTrigger1' (Reason='This function was programmatically called via the host APIs.', Id=45fc885-aa83-45b3-a79b-78ca7f0ed9d1)
2021-11-23T15:33:37.265 [Information] JavaScript HTTP trigger function processed a request.
2021-11-23T15:33:37.265 [Information] Executed 'Functions.HttpTrigger1' (Succeeded, Id=45fc885-aa83-45b3-a79b-78ca7f0ed9d1, Duration=3ms)
```

The test results panel shows the output tab selected, displaying:

HTTP response code  
200 OK

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

# Replace index.js

The screenshot shows the Microsoft Azure Functions code editor interface. The title bar indicates the current project is "HttpTrigger1 - Microsoft Azure". The main area displays the "index.js" file content, which is a Node.js script for an HTTP trigger function. The code uses context.log to log temperature readings and sets a status based on the reading's temperature. A red box highlights the dropdown menu next to the file name, which is currently set to "index.js". The left sidebar shows developer tools: "Code + Test" (selected), "Integration", "Monitor", and "Function Keys". The bottom navigation bar includes icons for various Windows applications like File Explorer, Task View, and Edge browser.

```
1 module.exports = function(context, req) {
2     context.log('Drive Gear Temperature Service triggered');
3     if (req.body && req.body.readings) {
4         req.body.readings.forEach(function(reading) {
5             if(reading.temperature<=25) {
6                 reading.status = 'OK';
7             } else if (reading.temperature<=50) {
8                 reading.status = 'CAUTION';
9             } else {
10                 reading.status = 'DANGER';
11             }
12             context.log('Reading is ' + reading.status);
13         });
14     }
15     context.res = {
16         // status: 200, /* Defaults to 200 */
17         body: {
18             "readings": req.body.readings
19         }
20     };
21 }
22 else {
23 }
```

# Replace Test Body

The screenshot shows the Microsoft Azure portal interface for the `HttpTrigger1` function. The top navigation bar includes tabs for `HttpTrigger1 - Microsoft Azure`, `練習 - 將邏輯新增至函數應用程式`, and `Exercise - Add logic to the func`. The main content area displays the `index.js` file code:

```
1 module.exports = function (context, req) {
2     context.log('Drive Gear Temperature Service triggered');
3     if (req.body && req.body.readings) {
4         req.body.readings.forEach(function(reading) {
5             if(reading.temperature<=25) {
6                 reading.status = 'OK';
7             } else if (reading.temperature<=50) {
8                 reading.status = 'CAUTION';
9             } else {
10                 reading.status = 'DANGER';
11             }
12             context.log('Reading is ' + reading.status);
13         });
14         context.res = {
15             // status: 200, /* Defaults to 200 */
16             body: {
17                 "readings": req.body.readings
18             }
19         };
20     }
21     else {
22     }
23 }
```

The toolbar above the code editor includes `Save`, `Discard`, `Refresh`, `Test/Run` (which is highlighted with a red box), `Upload`, and other options.

To the right of the code editor, there are sections for `headers` and `Body`. The `Body` section contains the following JSON data, which is highlighted with a yellow box:

```
1 [
2     {
3         "readings": [
4             {
5                 "driveGearId": 1,
6                 "timestamp": 1534263995,
7                 "temperature": 23
8             },
9             {
10                 "driveGearId": 3,
11                 "timestamp": 1534264048,
12                 "temperature": 45
13             },
14             {
15                 "driveGearId": 18,
16                 "timestamp": 1534264050,
17                 "temperature": 55
18             }
19         ]
20     }
21 ]
```

At the bottom of the interface are `Run` and `Close` buttons.

The taskbar at the bottom of the screen shows various pinned application icons, and the system tray indicates the date and time as `下午 11:42 2021/11/23`.

# Result

The screenshot shows the Microsoft Azure portal interface for an HttpTrigger1 function. The left sidebar has 'Code + Test' selected under 'Developer'. The main area displays the index.js code:

```
1 module.exports = function (context, req) {
2     context.log('Drive Gear Temperature Service triggered');
3     if (req.body && req.body.readings) {
4         req.body.readings.forEach(function(reading) {
5             if(reading.temperature<=25) {
6                 reading.status = 'OK';
7             } else if (reading.temperature<=50) {
8                 reading.status = 'CAUTION';
9             } else {
10                 reading.status = 'DANGER';
11             }
12             context.log('Reading is ' + reading.status);
13         });
14     };
}
```

The 'Logs' tab is open, showing the following log entries:

```
(Reason='This function was programmatically called via the host APIs.', Id=b3fe22b2-a9b5-4ded-9f4e-16891fe84d3b)
2021-11-23T15:43:55.089 [Information] Drive Gear Temperature Service triggered
2021-11-23T15:43:55.089 [Information] Reading is OK
2021-11-23T15:43:55.090 [Information] Reading is CAUTION
2021-11-23T15:43:55.090 [Information] Reading is DANGER
2021-11-23T15:43:55.095 [Information] Executed 'Functions.HttpTrigger1' (Succeeded, Id=b3fe22b2-a9b5-4ded-9f4e-16891fe84d3b, Duration=31ms)
```

The 'Output' tab shows the HTTP response code 200 OK and the HTTP response content:

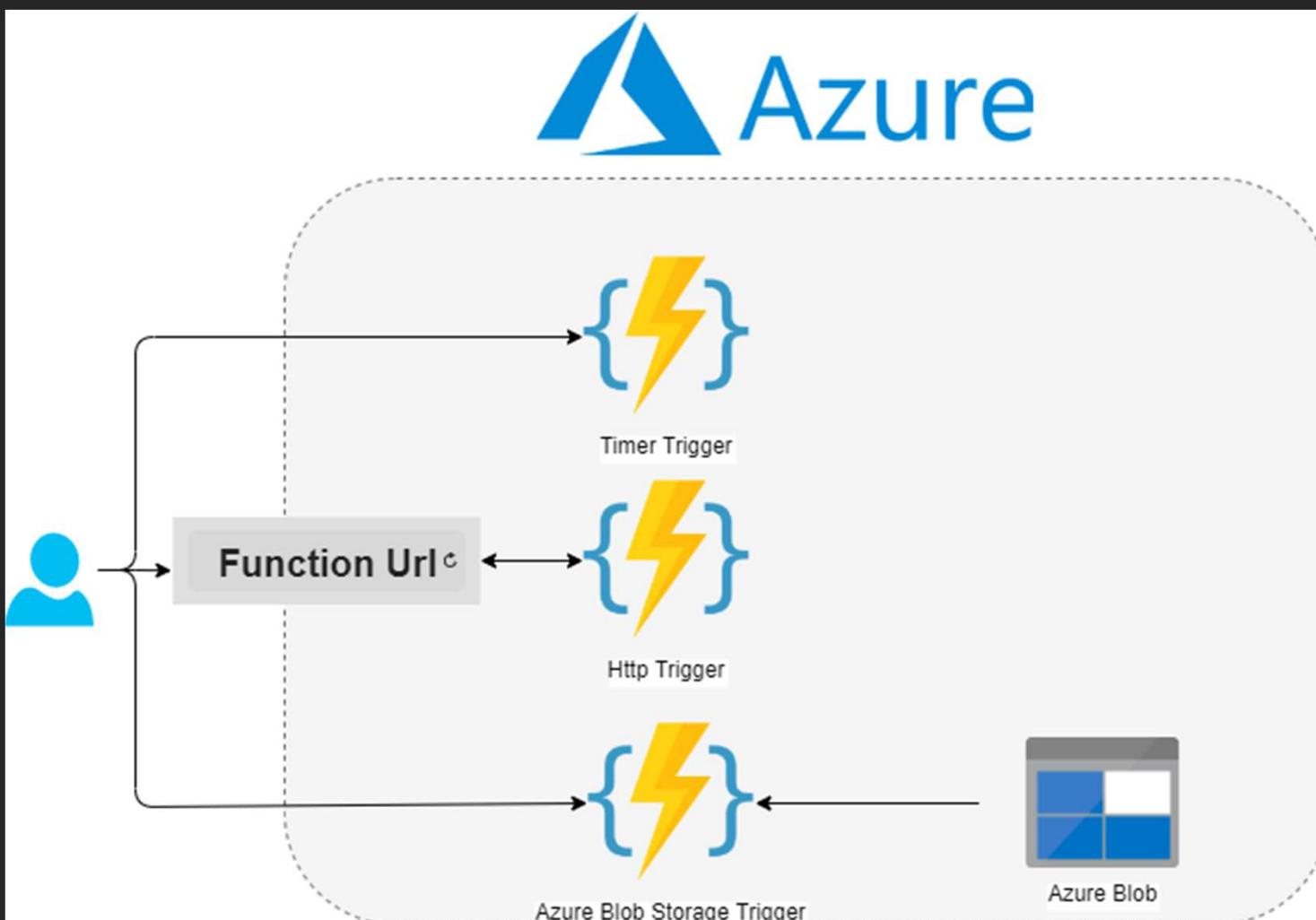
```
{
  "readings": [
    {
      "driveGearId": 1,
      "timestamp": 1534263995,
      "temperature": 23,
      "status": "OK"
    },
    {
      "driveGearId": 3,
      "timestamp": 1534264048,
      "temperature": 45,
      "status": "CAUTION"
    },
    {
      "driveGearId": 18,
      "timestamp": 1534264050,
      "temperature": 55,
      "status": "DANGER"
    }
  ]
}
```

The bottom status bar shows system icons and the date/time: 15°C, 2021/11/23, 下午 11:44.

03

// Execute an Azure Function with  
triggers

# Abstract



# Create a timer-triggered function

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar is collapsed, and the main area displays the 'Create function' dialog.

**Left Sidebar (Functions):**

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Security
- Events (preview)
- Functions** (highlighted with a red box)
- App keys
- App files
- Proxies
- Deployment
- Deployment slots
- Deployment Center
- Settings

**Main Area:**

**Create function**

Use a template to create a function. Triggers describe the type of events that invoke your functions. [Learn more](#)

**Template**

Template	Description
HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
<b>Timer trigger</b>	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 儲存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
Azure Blob Storage trigger	Blob 每次新增至指定的容器時，將會執行的函式
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

**Template details**

We need more information to create the Timer trigger function. [Learn more](#)

New Function\*

排程\*

**Buttons:**

**Create** (highlighted with a blue box) | **Cancel**

**System Tray:**

15°C 11:52 2021/11/23

# Configure the timer trigger

TimerTrigger1 | Integration

Overview

Developer

Code + Test

Integration **Integration**

Monitor

Function Keys

Search (Ctrl+ /)

Refresh

Integration

Edit the trigger and choose from a selection of inputs and outputs for your function, including Azure Blob Storage, Cosmos DB and others.

**Trigger**

計時器 (myTimer)

**Inputs**

No inputs defined  
+ Add input

**Function**

TimerTrigger1

**Outputs**

No outputs defined  
+ Add output

**Edit Trigger**

Save Discard Delete

Binding Type: 計時器

時間戳記參數名稱: myTimer

排程: \*/20 \* \* \* \*

# Test the timer

The screenshot shows the Microsoft Azure Function portal interface for the TimerTrigger1 function. The left sidebar has a red box around the 'Code + Test' tab, which is currently selected. The main area displays the function code in index.js:

```
1 module.exports = async function (context, myTimer) {  
2     var timestamp = new Date().toISOString();  
3 };
```

Below the code is a log viewer with a green box highlighting the log output. The logs show several executions of the function, with the most recent one being successful:

```
(Reason='Timer fired at 2021-11-23T15:58:40.0132827+00:00', Id=60c3b3b2-9e23-4356-86e9-19fabc92cdbe)  
2021-11-23T15:58:40.030 [Information] JavaScript timer trigger function ran! 2021-11-23T15:58:40.012Z  
2021-11-23T15:58:40.032 [Information] Executed 'Functions.TimerTrigger1' (Succeeded, Id=60c3b3b2-9e23-4356-86e9-19fabc92cdbe, Duration=19ms)  
2021-11-23T15:58:42.819 [Information] Executing 'Functions.TimerTrigger1' (Reason='This function was programmatically called via the host APIs.', Id=327f189d-b92f-4160-8c4f-bb818369995f)  
2021-11-23T15:58:42.822 [Information] JavaScript timer trigger function ran! 2021-11-23T15:58:42.814Z  
2021-11-23T15:58:42.822 [Information] Executed 'Functions.TimerTrigger1' (Succeeded, Id=327f189d-b92f-4160-8c4f-bb818369995f, Duration=4ms)  
2021-11-23T15:59:00.005 [Information] Executing 'Functions.TimerTrigger1' (Reason='Timer fired at 2021-11-23T15:59:00.0056242+00:00', Id=181ff029-6944-40cc-9cb1-7e7c052123f3)  
2021-11-23T15:59:00.008 [Information] JavaScript timer trigger function ran! 2021-11-23T15:59:00.004Z  
2021-11-23T15:59:00.009 [Information] Executed 'Functions.TimerTrigger1' (Succeeded, Id=181ff029-6944-40cc-9cb1-7e7c052123f3, Duration=3ms)  
2021-11-23T15:59:20.018 [Information] Executing 'Functions.TimerTrigger1' (Reason='Timer fired at 2021-11-23T15:59:20.0180996+00:00', Id=64cd6ce0-26ff-484d-8b31-4fb1e77ccb9)
```

To the right of the logs, there is a test panel with tabs for 'Input' and 'Output'. It includes a 'Key' dropdown set to 'master (Host key)' and a 'Body' input field. A yellow box highlights the 'Run' button at the bottom of the panel.

# Create a http-triggered function

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar is dark-themed and includes sections for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, Events (preview), Functions (highlighted with a red box), App keys, App files, Proxies, Deployment slots, Deployment Center, and Settings. The main content area has a light background and displays the 'Create function' page. At the top, there are three tabs: 'Create function - Microsoft Az...', 'Exercise - Create an HTTP trig...', and 'Create serverless logic with Az...'. Below the tabs, the URL in the address bar is [portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-cs456l/providers/Microsoft.Web/sites/10727211-FunctionApp/functions](https://portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-cs456l/providers/Microsoft.Web/sites/10727211-FunctionApp/functions). The top navigation bar includes links for 應用程式, 中原e點靈\_e17, i-learning中原網路..., Azure Portal, Azure, Google Drive, datamining, GitHub - huihui03..., Problems - LeetCo..., Syntax, 專題cite, PPT模板, 閱讀清單, and 10727211@O365st.cycu... 中原大學 (O365TC.CYCU.ED...).

The central part of the screen shows the 'Create function' dialog. It includes a search bar, a 'Create' button (highlighted with a yellow box), a 'Refresh' button, and a 'Delete' button. A 'Filter' input field is also present. The 'Template' section lists various trigger types:

Template	Description
HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
Timer trigger	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 賴存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
Azure Blob Storage trigger	Blob 每次新增至指定的容器時，將會執行的函式
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

The 'HTTP trigger' row is highlighted with a yellow box. The 'Name' dropdown shows 'HttpTrigger1' and 'TimerTrigger1'. The 'Template details' section indicates that more information is needed to create the function. The 'New Function\*' input field contains 'M2U6-HttpTrigger' and the 'Authorization level\*' dropdown is set to 'Anonymous' (highlighted with a green box). The bottom right of the dialog has 'Create' and 'Cancel' buttons, with the 'Create' button highlighted with a blue box.

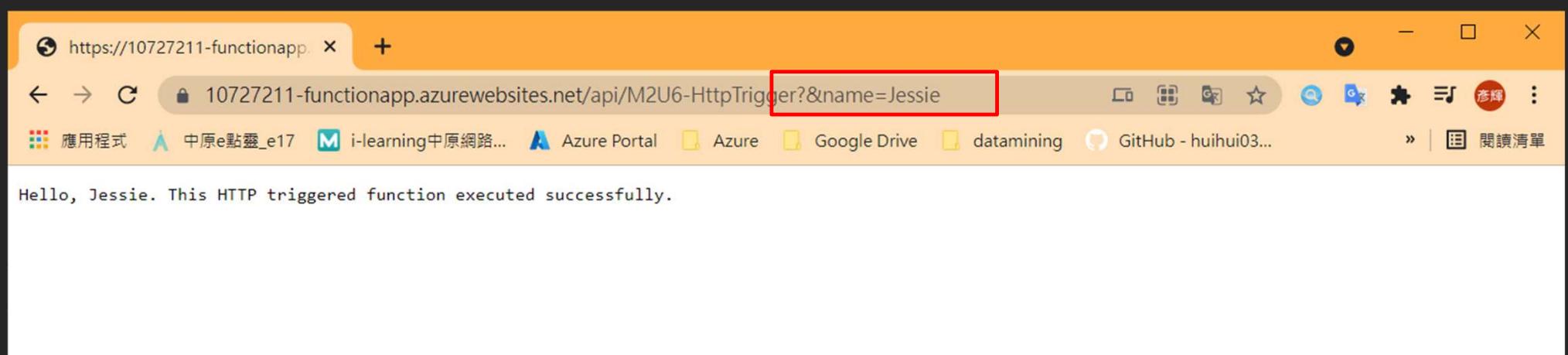
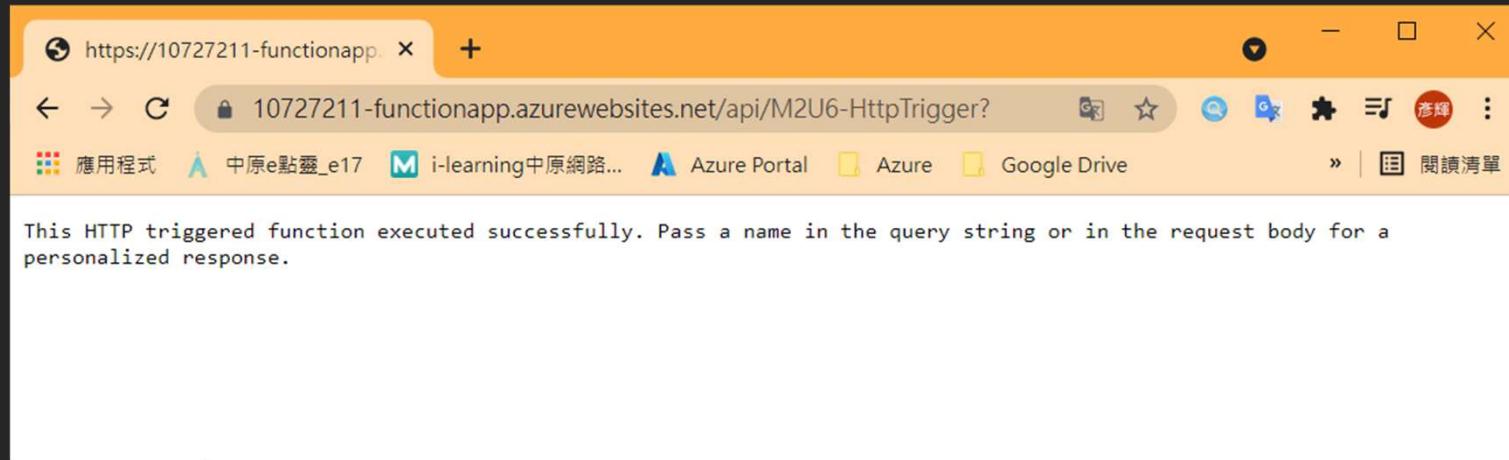
# Get Function Url

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for 'M2U6-HttpTrigger - Microsoft', 'Exercise - Create an HTTP trigger', and 'Create serverless logic with Az...'. The address bar shows the URL [portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/functionOverview/resourceId%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2Fres...](https://portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/functionOverview/resourceId%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2Fres...). The user's profile is visible on the right.

The main content area displays the 'M2U6-HttpTrigger' function details. On the left, a sidebar lists 'Overview', 'Developer', 'Code + Test', 'Integration', 'Monitor', and 'Function Keys'. The 'Developer' section is active, showing a dropdown menu set to 'default (function key)' and a URL input field containing <https://10727211-functionapp.azurewebsites.net/api/M2U6-HttpTrigger?>. A red box highlights the 'Get Function Url' button. Below the URL input, there are 'OK' and 'Cancel' buttons. At the bottom of the developer panel, it shows 'Subscription (Move) : 中原大學' and 'Subscription ID : ba9f5e30-1488-49db-bae0-9026dacc11b0'.

The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time as '2021/11/24 上午 12:10'.

# Test Function Url



# Create a blog trigger

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with navigation links like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, Events (preview), Functions, App keys, App files, Proxies, Deployment slots, Deployment Center, and Settings. The main area is titled "Create function". At the top right of the main area, there's a "Create" button highlighted with a red box. Below it, a list of trigger templates is shown, with "Azure Blob Storage trigger" highlighted with a yellow box. The "Template details" section below lists the trigger type as "Azure Blob Storage trigger", the name as "M2U8-BlobTrigger", the path as "samples-workitems/{name}", and the storage account connection as "10727211appstorage\_STORAGE (new)". A green box highlights the "New" button next to the storage account dropdown. At the bottom of the dialog, there are "Create" and "Cancel" buttons.

Template

Template	Description
HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
Timer trigger	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 儲存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
<b>Azure Blob Storage trigger</b>	<b>Blob 每次新增至指定的容器時，將會執行的函式</b>
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

Template details

We need more information to create the Azure Blob Storage trigger function. [Learn more](#)

New Function\*

路徑\*

儲存體帳戶連線\*  [New](#)

[Create](#) [Cancel](#)

# Create a blog container (create new tab)

A M2U8-BlobTrigger - Microsoft | A New container - Microsoft Azure | Exercise - Create a Blob trigger | Create serverless logic with Az... | +

portal.azure.com/#@O365tc.cygu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/providers/Microsoft.Storage...

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+)

Home > 110-1-CS456L > 10727211appstorage

10727211appstorage | Storage Explorer (preview)

Storage account

Search (Ctrl+ /) Search

BLOB CONTAINERS Create blob container Refresh

FILE SHARES

QUEUES TABLES

Overview Activity log Tags Diagnose and solve problems Access Control (IAM) Data migration Storage Explorer (preview) **Storage Explorer (preview)**

Name \* samples-workitems

Public access level Private (no anonymous access)

Advanced

Create Discard

The screenshot shows the Azure Storage Explorer (preview) interface. On the left, there's a navigation sidebar with various options like Overview, Activity log, and Storage Explorer (preview). The Storage Explorer (preview) option is highlighted with a yellow box. The main area shows a storage account named '10727211appstorage'. Under the 'Containers' section, there are two existing containers named 'azure-web' and a 'Create blob container' button. A yellow box highlights the 'Create blob container' button. To the right, a 'New container' dialog box is open, prompting for a name ('samples-workitems') which is also highlighted with a yellow box. Below the name input, there are dropdowns for 'Public access level' (set to 'Private (no anonymous access)') and an 'Advanced' section. At the bottom of the dialog are 'Create' and 'Discard' buttons, with the 'Create' button being highlighted with a blue box.

# Turn on your blog trigger

The screenshot shows the Microsoft Azure portal interface for managing a Function App named 'M2U8-BlobTrigger'. The top navigation bar includes tabs for 'M2U8-BlobTrigger - Microsoft', '10727211appstorage - Microsoft', 'Exercise - Create a Blob trigger', and 'Create serverless logic with Az...'. The main search bar says 'Search resources, services, and docs (G+)'. The user is signed in as '10727211@O365st.cy... 中原大學 (O365TC.CYCU.ED...)'. The breadcrumb navigation shows 'Home > 110-1-CS456L > 10727211-FunctionApp > M2U8-BlobTrigger'.

The left sidebar has sections for 'Overview', 'Developer' (with 'Code + Test' highlighted by a red box), 'Integration', 'Monitor', and 'Function Keys'. The 'Code + Test' section contains a code editor for 'index.js':

```
1 module.exports = async function (context, myBlob) {  
2     context.log(`JavaScript blob trigger function processed blob\n Blob: ${myBlob}\n Blob Size: ${myBlob.length} Bytes`);  
3 };
```

The 'Test/Run' button in the toolbar is highlighted with a yellow box. Below the code editor is a 'Logs' panel with a yellow border, showing the message 'Connected!' and the log entry '2021-11-23T16:27:18 Welcome, you are now connected to log-streaming service. The default timeout is 2 hours. Change the timeout with the App Setting SCM\_LOGSTREAM\_TIMEOUT (in seconds.)'.

The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time as '2021/11/24 上午 12:27'.

# Create a blob

The screenshot shows the Microsoft Azure Storage Explorer (preview) interface. On the left, the navigation pane includes links for Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, and Storage Explorer (preview). Under Data storage, there are sections for Containers, File shares, Queues, and Tables. Under Security + networking, there are sections for Networking, Azure CDN, and Access keys.

The main area displays a storage account named "10727211appstorage". In the "BLOB CONTAINERS" section, a container named "samples-workitems" is selected and highlighted with a red box. The "Upload" button in the toolbar above the list is also highlighted with a yellow box. The list table shows the following columns: NAME, ACCESS TIER, ACCESS TIER LAST MODIFIED, LAST MODIFIED, BLOB TYPE, CONTENT TYPE, and SIZE. A search bar at the top of the list table contains the text "samples-workitems".

To the right, a modal window titled "Upload blob" is open. It shows a file selection dialog with the path "samples-workitems/" and the file name "HITACHI.pdf" selected. There is a checkbox for "Overwrite if files already exist" and an "Advanced" section. At the bottom of the modal is a large blue "Upload" button, which is highlighted with a green box.

The taskbar at the bottom of the screen shows various pinned icons, including Microsoft Edge, Google Chrome, Spotify, and File Explorer. The system tray indicates the date as "2021/11/24" and the time as "上午 12:30".

# Result

The screenshot shows the Azure Functions portal interface for the 'M2U8-BlobTrigger' function. The left sidebar has 'Code + Test' selected under 'Developer'. The main area displays the 'index.js' file content:

```
1 module.exports = async function(context, myBlob) {
2     context.log(`JavaScript blob trigger function processed blob ${myBlob.name}. Blob size: ${myBlob.length} Bytes`);
3 };
```

Below the code editor is a 'Logs' section. The log stream shows several entries, with the last four highlighted by a red box:

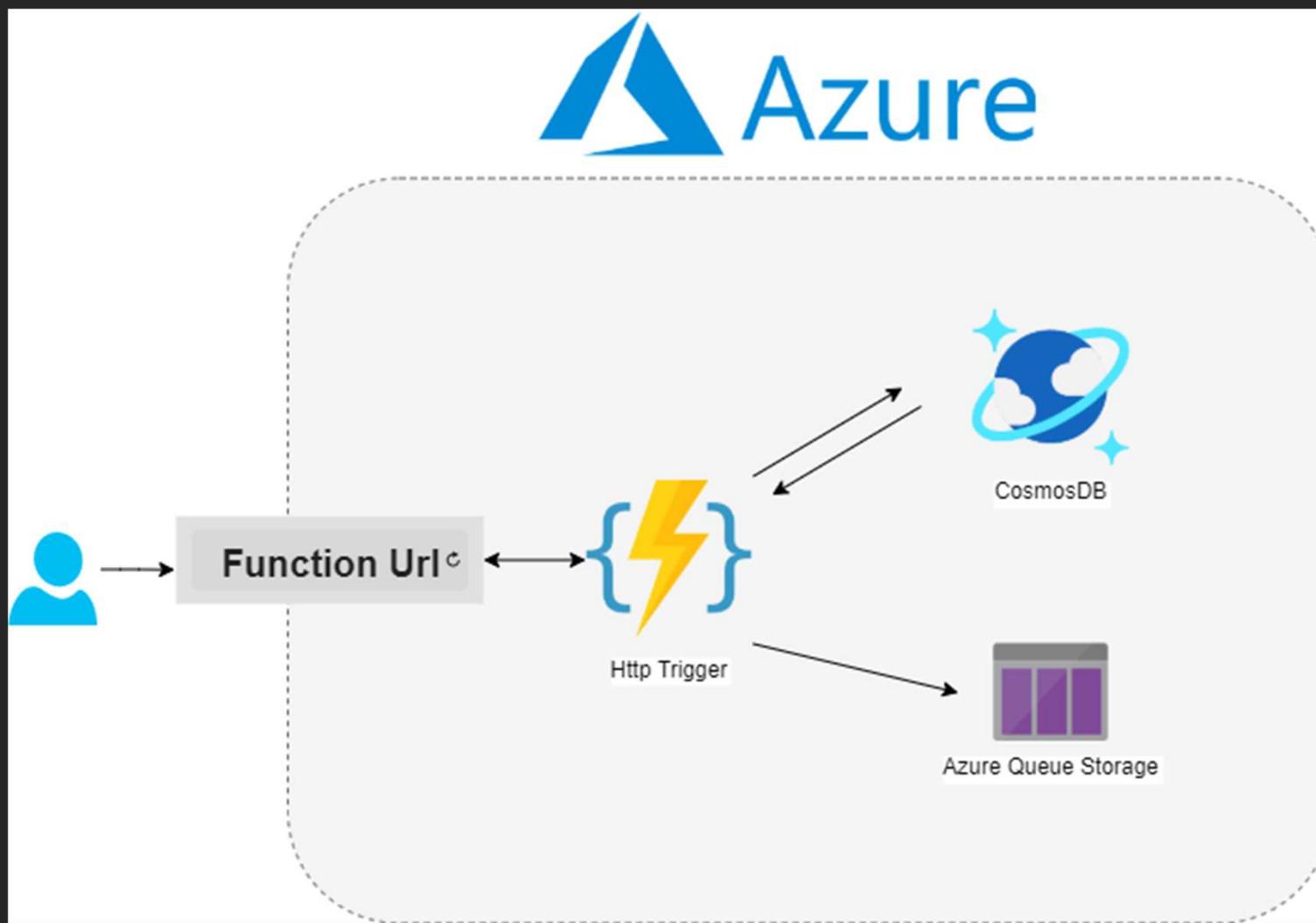
```
Connected!
2021-11-23T16:27:18 Welcome, you are now connected to log streaming service. The default timeout is 2 hours. Change the timeout with the App Setting SCM_LOGSTREAM_TIMEOUT (in seconds).
2021-11-23T16:28:18 No new trace in the past 1 min(s).
2021-11-23T16:29:18 No new trace in the past 2 min(s).
2021-11-23T16:30:18 No new trace in the past 3 min(s).
2021-11-23T16:31:18 No new trace in the past 4 min(s).
2021-11-23T16:31:33.612 [Information] Executing 'Functions.M2U8-BlobTrigger' (Reason='New blob detected: samples-workitems/HITACHI.pdf', Id=2ec459f0-41b0-48ea-aa6c-95257056cd1)
2021-11-23T16:31:33.613 [Information] Trigger Details: MessageId: 6d11d766-f153-41d0-9b85-fa798f2f1827, DequeueCount: 1, InsertionTime: 2021-11-23T16:31:33.000+00:00, BlobCreated: 2021-11-23T16:31:30.000+00:00, BlobLastModified: 2021-11-23T16:31:30.000+00:00
2021-11-23T16:31:33.651 [Information] JavaScript blob trigger function processed blob Blob: samples-workitems/HITACHI.pdf Blob Size: 371576 Bytes
2021-11-23T16:31:33.652 [Information] Executed 'Functions.M2U8-BlobTrigger' (Succeeded, Id=2ec459f0-41b0-48ea-aa6c-95257056cd1, Duration=124ms)
```

The system tray at the bottom of the screen shows various icons and the date/time: 16°C, 12:31, 2021/11/24.

04

// Chain Azure Functions together  
using input and output bindings

# Abstract



# Create a http trigger function

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar is for the '10727211-FunctionApp' function app, with the 'Functions' section highlighted by a red box. The main area is titled 'Create function' and displays a list of templates. The 'HTTP trigger' template is selected and highlighted by a yellow box. The 'Template details' section below it shows the function name 'M4U3-HttpTrigger' and the authorization level 'Function'. The bottom right corner of the screenshot shows the Windows taskbar with the date and time.

Microsoft Azure

Home > 110-1-CS456L > 10727211-FunctionApp | Functions

Search resources, services, and docs (G+/)

Microsoft Azure

10727211@O365st.cycu... 中原大學 (O365TC.CYCU.ED...

Search (Ctrl+ /) Create Refresh Delete

Filter by name...

Template Description

HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
Timer trigger	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 儲存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
Azure Blob Storage trigger	Blob 每次新增至指定的容器時，將會執行的函式
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

Name ↑

HttpTrigger1  
M2U6-HttpTrigger  
M2U8-BlobTrigger  
TimerTrigger1

Template details

We need more information to create the HTTP trigger function. Learn more

New Function\* M4U3-HttpTrigger

授權等級\*① Function

Create Cancel

下午 07:11  
2021/11/24

# Create Azure CosmosDB - 1

The screenshot shows the Microsoft Azure 'Create a resource' interface. On the left, there's a sidebar with a red box around the 'Databases' category. In the main area, under 'Popular products', the 'Azure Cosmos DB' item is highlighted with a yellow box. The Azure Cosmos DB card includes icons for Create, Docs, and MS Learn.

Get started  Getting Started? Try our Quickstart center

Recently created

Categories

- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers
- Databases **Selected**
- Developer Tools
- DevOps
- Identity
- Integration
- Internet of Things
- IT & Management Tools
- Media
- Migration
- Mixed Reality

Popular products [See more in Marketplace](#)

- Azure SQL Managed Instance [Create](#) [Docs](#) [MS Learn](#)
- SQL Database [Create](#) [Docs](#) [MS Learn](#)
- Azure Cosmos DB** [Create](#) [Docs](#) [MS Learn](#)
- Azure Database for PostgreSQL [Create](#) [Docs](#) [MS Learn](#)
- Azure Database for MySQL [Create](#) [Docs](#) [MS Learn](#)
- HVR for Microsoft Azure [Create](#) [Learn more](#)
- SQL Server 2017 Enterprise Windows Server 2016 [Create](#) [Learn more](#)
- Azure Cache for Redis [Create](#) [Docs](#) [MS Learn](#)

portal.azure.com/#create/hub

10727211@O365st.cy... 中原大學 (O365TC.CYCU.ED...

下午 07:19  
2021/11/24

# Create Azure CosmosDB - 2

The screenshot shows a Microsoft Edge browser window with the URL [portal.azure.com/#create/Microsoft.DocumentDB](https://portal.azure.com/#create/Microsoft.DocumentDB). The title bar says "Select API option - Microsoft Azure". The page is titled "Select API option" and displays six options for creating a new Azure Cosmos DB account:

- Core (SQL) - Recommended**: Fully managed database service for apps written with SQL query language and client libraries for .NET, JavaScript, Python, and Java. This option is highlighted with a red border.
- Azure Cosmos DB API for MongoDB**: Fully managed database service for apps written for MongoDB. Recommended if you have existing MongoDB workloads that you plan to migrate to Azure Cosmos DB.
- Cassandra**: Fully managed Cassandra database service for apps written for Apache Cassandra. Recommended if you have existing Cassandra workloads that you plan to migrate to Azure Cosmos DB.
- Azure Table**: Fully managed database service for apps written for Azure Table storage. Recommended if you have existing Azure Table storage workloads that you plan to migrate to Azure Cosmos DB, but do not want to re-write your application to use the SQL API.
- Gremlin (Graph)**: Fully managed graph database service using the Gremlin query language, based on Apache TinkerPop project. Recommended for new workloads that need to store relationships between data.
- PostgreSQL**: Build new applications using PostgreSQL APIs with integrated features like JSONB, geospatial support, rich indexing, dozens of extensions, and high-performance scale-out.

Each option has a "Create" button and a "Learn more" link. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time (下午 07:20, 2021/11/24).

# Create Azure CosmosDB - 3

The screenshot shows the Microsoft Azure portal interface for creating a new Cosmos DB account. The top navigation bar includes the title 'Create Azure Cosmos DB Account', the URL 'portal.azure.com/#create/Microsoft.DocumentDB', and various browser tabs and icons.

The main content area displays the 'Create Azure Cosmos DB Account - Core (SQL)' configuration page. It starts with a note about selecting a subscription for managing resources and costs. The 'Subscription' dropdown is set to '中原大學'. Below it, the 'Resource Group' dropdown is set to '110-1-CS456L', with a 'Create new' option available.

The 'Instance Details' section contains the following fields:

- Account Name \***: 10727211cosmosdb
- Location \***: (Europe) France Central
- Capacity mode**:  Provisioned throughput  Serverless  
Learn more about capacity mode

A note below states: "With Azure Cosmos DB free tier, you will get the first 1000 RU/s and 25 GB of storage for free in an account. You can enable free tier on up to one account per subscription. Estimated \$64/month discount per account."

Below this, there are two options: "Apply Free Tier Discount" with radio buttons for "Apply" (selected) and "Do Not Apply", and a checkbox for "Limit total account throughput". A tooltip for the throughput limit explains: "This limit will prevent unexpected charges related to provisioned throughput. You can update or remove this limit after your account is created."

At the bottom of the page, there are two buttons: 'Review + create' (highlighted with a red border) and 'Next: Global Distribution'.

The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time as '下午 07:23 2021/11/24'.

# Create Azure CosmosDB - Result

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the URL 'portal.azure.com/#@O365tc.cygu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourcegroups/110-1-CS456L/providers/Microsoft.DocumentDB/accounts/10727211cosmosdb'. The left sidebar has a dark theme with various service icons and the 'Overview' section selected. The main content area displays the '10727211cosmosdb' account details, including its status as 'Online', resource group '110-1-CS456L', subscription '中原大學', and URI 'https://10727211cosmosdb.documents.azure.com:443/'. It also shows 'Read Locations' and 'Write Locations' both set to 'France Central'. The 'Containers' section indicates no containers are present. Below this are sections for 'Monitoring' (with time ranges from 1 hour to 30 days) and 'Estimated Cost (hourly)'.

# Add a Container

The screenshot shows the Microsoft Azure Data Explorer interface for the database account '10727211cosmosdb'. The left sidebar is collapsed, and the main area displays the 'Data Explorer' tab. A yellow box highlights the 'New Container' button in the top navigation bar. Another yellow box highlights the 'New Container' dialog box on the right side of the screen.

**New Container Dialog Box:**

- Database id:** func-io-learn-db (radio button selected: Create new)
- Share throughput across containers
- Container id:** Bookmarks
- Partition key:** /id
- Container throughput (autoscale):** Autoscale (radio button selected)
- Estimate your required RU/s with [capacity calculator](#).
- Container Max RU/s:** 4000
- Your container throughput will automatically scale from **400 RU/s** (10% of max RU/s) - 4000 RU/s based on usage.

# Add test data - 1

The screenshot shows the Microsoft Azure Data Explorer interface for the database account '10727211cosmosdb'. The left sidebar lists various management options like Overview, Activity log, Access control (IAM), Tags, and Data Explorer. The main area is titled 'func-io-learn-db' under the 'DATA' section. A red box highlights the 'func-io-learn-db' database. The 'Items' table is selected, showing a single item with the ID 'docs'. The item's JSON representation is displayed in the bottom right, with a green box around it. The top navigation bar includes a search bar, a 'New Item' button (highlighted with a yellow box), a 'Save' button (highlighted with a green box), and other standard browser controls.

```
{  
  "id": "docs",  
  "url": "https://docs.microsoft.com/azure"  
}
```

# Add test data - 2

The screenshot shows the Microsoft Azure Data Explorer interface for the database account '10727211cosmosdb'. The left sidebar is the 'Data Explorer' settings menu, with 'Data Explorer' selected. The main area displays a JSON document under the 'func-io-learn-db' database. The document has the following structure:

```
1  {
2    "id": "docs",
3    "url": "https://docs.microsoft.com/azure",
4    "_rid": "AzAvAN1Djj8BAAAAAAA==",
5    "_self": " dbs/AzAvAA==/colls/AzAvAN1Djj8=/docs/AzAvAN1Djj8BAAAAAAA==/",
6    "_etag": "\"0200a30d-0000-0e00-0000-619e245e0000\"",
7    "_attachments": "attachments/",
8    "_ts": 1637753950
9 }
```

The browser address bar shows the URL for the Azure portal, and the system tray at the bottom indicates the date and time as '下午 07:39 2021/11/24'.

# Add test data - 3

The screenshot shows the Microsoft Azure Data Explorer interface for the database '10727211cosmosdb'. The left sidebar lists various Azure services, and the main area shows the SQL API interface.

In the center, under the 'func-io-learn-db' database, there is a table named 'Bookmarks' with the following data:

id	/id
docs	docs
portal	portal
learn	learn
marketplace	marketplace
blog	blog

A red box highlights the rows for 'portal', 'learn', and 'marketplace'.

On the right side, four JSON documents are displayed, each representing a bookmark entry:

```
{ "id": "portal", "url": "https://portal.azure.com" }
```

```
{ "id": "learn", "url": "https://docs.microsoft.com/learn" }
```

```
{ "id": "blog", "url": "https://azure.microsoft.com/blog", "rid": "AzAvAN1Djj8FAAAAAAAA==", "self": "dbs/AzAvAA=/colls/AzAvAN1Djj8=/docs/AzAvAN1Djj8FAAAAAAAA==/", "etag": "\\"0200a70d-0000-0e00-0000-619e24d50000\\\"", "attachments": "attachments/", "c": "1627754050" }
```

```
{ "id": "marketplace", "url": "https://azuremarketplace.microsoft.com/marketplace/apps" }
```

```
{ "id": "blog", "url": "https://azure.microsoft.com/blog" }
```

# Add an Azure Cosmos DB input binding -1

The screenshot shows the Microsoft Azure portal interface for managing an Azure Function named "M4U3-HttpTrigger". The left sidebar has a red box around the "Integration" tab, which is currently selected. The main area displays the function's integration configuration. A yellow box highlights the "Inputs" section under the "Trigger" configuration, which shows "No inputs defined" and a button "+ Add input". On the right, a modal window titled "Create Input" is open, also with a yellow box around it. The "Binding Type" dropdown is set to "Azure Cosmos DB", also highlighted with a yellow box. The "Azure Cosmos DB details" section contains fields for "Cosmos DB 帐户連線" (ConnectionString) with a note "No existing connections available" and a required field indicator "This field is required", and "New" button. Below that are fields for "文件參數名稱" (File Parameter Name) with value "inputDocument" and "資料庫名稱" (Database Name) with value "inDatabase". At the bottom of the modal are "OK" and "Cancel" buttons.

M4U3-HttpTrigger - Microsoft

portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/integration/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2FresourceG...

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCode... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure

Search resources, services, and docs (G+/-)

Home > 10727211-FunctionApp > M4U3-HttpTrigger

**M4U3-HttpTrigger | Integration**

Function

Search (Ctrl+/)

Overview

Developer

Code + test

**Integration**

Monitor

Function Keys

**Integration**

Edit the trigger and choose from a selection of inputs and outputs for your function, including Azure Blob Storage, Cosmos DB and others.

**Trigger**

HTTP (req)

**Inputs**

No inputs defined

+ Add input

**Function**

M4U3-HttpTrigger

**Outputs**

HTTP (res)

+ Add output

**Create Input**

Start by selecting the type of input binding you want to add.

**Binding Type**

Azure Cosmos DB

**Azure Cosmos DB details**

Cosmos DB 帳戶連線

No existing connections available

This field is required

New

文件參數名稱

inputDocument

資料庫名稱

inDatabase

OK Cancel

18°C 11/24 07:57

# Add an Azure Cosmos DB input binding -2

M4U3-HttpTrigger - Microsoft + New

portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/integration/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2FresourceG...

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+/)

Home > 10727211-FunctionApp > M4U3-HttpTrigger

### M4U3-HttpTrigger | Integration

Function

Search (Ctrl+ /) Refresh

Overview

Developer

Code + Test

Integration (Selected)

Monitor

Function Keys

**Integration**

Edit the trigger and choose from a selection of inputs and outputs for your function, including Azure Blob Storage, Cosmos DB and others.

**Trigger**: HTTP (req)

**Function**: M4U3-HttpTrigger

**Outputs**: HTTP (res) + Add output

**Inputs**: No inputs defined + Add input

**Create Input**

10727211cosmosdb\_DOCUMENTDB ...

New

文件參數名稱 \*①  
bookmark

資料庫名稱 \*①  
func-io-learn-db

集合名稱 \*①  
Bookmarks

文件識別碼 (選擇性) ①  
id

分割區素引鍵 (選用) ①  
/id

OK Cancel

```
graph LR; Trigger[Trigger: HTTP (req)] --> Function[Function: M4U3-HttpTrigger]; Function --> Outputs[Outputs: HTTP (res), + Add output];
```

# Replace index.js

The screenshot shows the Microsoft Azure portal interface for managing a function app. The top navigation bar includes links for portal.azure.com, Microsoft Azure, and various Azure services like Azure Portal, Google Drive, and GitHub. The main content area is titled "M4U3-HttpTrigger | Code + Test". On the left, there's a sidebar with developer tools: Code + Test (selected), Integration, Monitor, and Function Keys. The main workspace displays the "index.js" file content:

```
1 module.exports = function(context, req) {
2   var bookmark = context.bindings.bookmark
3
4   if(bookmark){
5     context.res = {
6       body: { "url": bookmark.url },
7       headers: {
8         'Content-Type': 'application/json'
9       }
10    };
11  }
12
13  else {
14}
```

A yellow box highlights the "Save" button in the toolbar above the code editor. A red box highlights the code editor area where the "index.js" file is displayed.

# Replace function.json

The screenshot shows the Microsoft Azure portal interface for managing a function app. The current view is the 'Code + Test' section of the 'M4U3-HttpTrigger' function within the '10727211-FunctionApp'. The URL in the browser bar is <https://10727211-functionapp.azurewebsites.net/>.

The function.json file contains the following code:

```
7     "name": "req",
8     "methods": [
9       "get",
10      "post"
11    ],
12  },
13  {
14    "type": "http",
15    "direction": "out",
16    "name": "res"
17  },
18  {
19    "name": "bookmark",
20    "direction": "in",
21    "type": "cosmosDB",
22    "databaseName": "func-io-learn-db",
23    "collectionName": "bookmarks",
24    "connectionStringSetting": "10727211cosmosdb_DOCUMENTDB",
25    "id": "{id}",
26    "partitionKey": "{id}"
27  }
28]
```

A red box highlights the line `"connectionStringSetting": "10727211cosmosdb_DOCUMENTDB"`, which is the connection string for the CosmosDB output binding.

# Result

The screenshot shows the Microsoft Azure portal interface. In the center, there is a code editor for the 'M4U3-HttpTrigger' function. The code is as follows:

```
7  ...  "name": "req",  
8  ...  "methods": [  
...]
```

Below the code editor, there is a 'Get function URL' section. It includes a dropdown menu labeled 'Key' with 'default' selected, and a URL field containing <https://10727211-functionapp.azurewebsites.net/api/M4U3-HttpTrigger?code=wuWZF7ldkmpCQa/OD9iRY00NPH7523JplnVpkS99FF/lZuP7JNfOJu==&id=docs>. A red box highlights the copy icon next to the URL field.

The screenshot shows a browser window displaying the response from the Azure function URL. The URL in the address bar is the same as the one in the Azure portal: <https://10727211-functionapp.azurewebsites.net/api/M4U3-HttpTrigger?code=wuWZF7ldkmpCQa/OD9iRY00NPH7523JplnVpkS99FF/lZuP7JNfOJu==&id=docs>. The response body contains the following JSON object, with a yellow box highlighting the entire response area:

```
{  
  "url": "https://docs.microsoft.com/azure"  
}
```

# Result

The screenshot shows the Azure Portal interface for the function app "M4U3-HttpTrigger". The left sidebar has a red box around the "Integration" tab under the "Developer" section. The main area displays the function's architecture: an "HTTP (req)" trigger connects to a "Function" named "M4U3-HttpTrigger", which then connects to an "Outputs" section. The "Outputs" section shows an "HTTP (res)" binding with a yellow box around the "+ Add output" button. A pink box highlights the path from the trigger to the function. A large pink box labeled "/id" is overlaid on the "Outputs" section, with a pink arrow pointing from it towards the "+ Add output" button. A yellow box surrounds the "Create Output" dialog box on the right, which contains fields for "文件參數名稱" (newbookmark), "資料庫名稱" (func-io-learn-db), and "集合名稱" (Bookmarks). It also includes a toggle for "若為 true，則建立 Cosmos DB 資料庫..." (No) and a checkbox for "分割區索引鍵 (選用)".

M4U3-HttpTrigger - Microsoft Edge

portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/integration/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2FresourceG...

應用程式 中原e點靈\_e17 i-learning中原網路... Azure Google Drive datamining GitHub - huihui03... Problems - LeetCo... Syntax 專題cite PPT模板 閱讀清單

Microsoft Azure Search resources, services, and docs (G+)

Home > 10727211-FunctionApp > M4U3-HttpTrigger

**M4U3-HttpTrigger | Integration**

Trigger  
HTTP (req)

Function  
M4U3-HttpTrigger

Inputs  
Azure Cosmos DB (bookmark)  
+ Add input

Outputs  
HTTP (res)  
+ Add output

Create Output

10727211cosmosdb\_DOCUMENTDB

文件參數名稱 \*①  
newbookmark

資料庫名稱 \*①  
func-io-learn-db

集合名稱 \*①  
Bookmarks

若為 true，則建立 Cosmos DB 資料庫... \*①  
No

分割區索引鍵 (選用)

/id

# Add an Azure Queue Storage output binding

The screenshot shows the Microsoft Azure portal interface for managing an Azure Function named "M4U3-HttpTrigger". The left sidebar is titled "Developer" and includes options like "Code + Test", "Integration" (which is selected), "Monitor", and "Function Keys". The main area is titled "Integration" and contains a flow diagram: "Trigger" (HTTP (req)) → "Function" (M4U3-HttpTrigger) → "Outputs". The "Outputs" section lists "HTTP (res)" and "Azure Cosmos DB (newbookmarks)". A red box highlights the "+ Add output" button in this section. A modal window titled "Create Output" is open on the right, asking "Start by selecting the type of output binding you want to add." It shows the "Binding Type" set to "Azure Queue Storage" and the "Azure Queue Storage details" section, which includes "Storage account connection" (set to "10727211appstorage\_STORAGE") and "Queue name" (set to "bookmarks-post-process").

M4U3-HttpTrigger - Microsoft

portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/integration/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2FresourceG...

Microsoft Azure

Search resources, services, and docs (G+/)

Home > 10727211-FunctionApp > M4U3-HttpTrigger

M4U3-HttpTrigger | Integration

Trigger

HTTP (req)

Inputs

Azure Cosmos DB (bookmark)  
+ Add input

Function

M4U3-HttpTrigger

Outputs

HTTP (res)  
Azure Cosmos DB (newbookmarks)  
+ Add output

Create Output

Start by selecting the type of output binding you want to add.

Binding Type

Azure Queue Storage

Azure Queue Storage details

儲存體帳戶連線 \*①

10727211appstorage\_STORAGE

New

訊息參數名稱 \*①

newmessage

佇列名稱 \*①

bookmarks-post-process

OK Cancel

# Replace index.js

The screenshot shows the Microsoft Azure portal interface for managing an Azure Function. The function name is "M4U3-HttpTrigger". The "Code + Test" tab is selected in the left sidebar under the "Developer" section. The main area displays the "index.js" file content:

```
1 module.exports = function (context, req) {
2     var bookmark = context.bindings.bookmark
3     if(bookmark){
4         context.res = {
5             status: 422,
6             body : "Bookmark already exists.",
7             headers: {
8                 'Content-Type': 'application/json'
9             }
10        };
11    }
12    else {
13        // Create a JSON string of our bookmark.
14        var bookmarkString = JSON.stringify({
15            id: req.body.id,
16            url: req.body.url
17        });
18        // Write this bookmark to our database.
19        context.bindings.newbookmark = bookmarkString;
20    }
21    // ...
22    // ...
23}
```

The lines from 1 to 23 are highlighted with a red rectangle. The "Save" button in the toolbar is also highlighted with a yellow box.

# Writing message to queue - 1

The screenshot shows the Microsoft Azure portal interface for the 'M4U3-HttpTrigger' function. The top navigation bar includes tabs for 'M4U3-HttpTrigger - Microsoft' and 'bookmarks-post-process - Microsoft'. The main page title is 'M4U3-HttpTrigger | Code + Test'. The left sidebar has sections for 'Overview', 'Developer' (selected), 'Code + Test' (highlighted in blue), 'Integration', 'Monitor', and 'Function Keys'. The 'Code + Test' section displays the 'index.js' file content:

```
1 module.exports = function (context, req) {
2     var bookmark = context.bindings.bookmark
3     if(bookmark){
4         context.res = {
5             status: 422,
6             body : "Bookmark already exists.",
7             headers: {
8                 'Content-Type': 'application/json'
9             }
10        };
11    }
12    else{
13        // Create a JSON string of our bookmark.
14        var bookmarkString = JSON.stringify({
15            id: req.body.id,
16            url: req.body.url
17        });
18        // Write this bookmark to our database.
19        context.bindings.newbookmark = bookmarkString;
20    }
21}
22
23
```

The 'Logs' tab is also visible at the bottom left. On the right, the 'Test/Run' interface is shown with the following configuration:

- HTTP method:** POST
- Key:** master (Host key)
- Query:** (empty)
- Headers:** (empty)
- Body:** (highlighted with a yellow box)

```
1 [
2     {
3         "id": "github",
4         "url": "https://www.github.com"
5     }
]
```

At the bottom right of the test interface are 'Run' and 'Close' buttons.

# Writing message to queue - 2

The screenshot shows the Microsoft Azure Functions developer interface for a function named 'M4U3-HttpTrigger'. The left sidebar has 'Code + Test' selected. The main area displays the 'index.js' code:

```
1 module.exports = function (context, req) {
2     var bookmark = context.bindings.bookmark
3     if(bookmark){
4         context.res = {
5             status: 422,
6             body : "Bookmark already exists.",
7             headers: {
8                 'Content-Type': 'application/json'
9             }
10    };
11 }
12 else{
13
14
15     // Create a JSON string of our bookmark.
16     var bookmarkString = JSON.stringify({
17         id: req.body.id,
18         url: req.body.url
19     });
20
21     // Write this bookmark to our database.
22     context.bindings.newbookmark = bookmarkString;
23 }
```

To the right, a red box highlights the 'Output' tab of the test results. It shows the 'HTTP response code' as '200 OK' and the 'HTTP response content' as 'bookmark added!'. At the bottom right are 'Run' and 'Close' buttons.

# Verify that a message is written to the queue - 1

The screenshot shows the Microsoft Azure Storage Queues page for the storage account '10727211appstorage'. The left sidebar navigation bar is visible, with 'Queues' selected. The main content area displays a table of queues. Two entries are present:

Queue	Url	More
azur...-functionapp	https://10727211appstorage.queue.core.windows.net/azur...-functionapp	...
bookmarks-post-process	https://10727211appstorage.queue.core.windows.net/bookmarks-post-process	...

# Verify that a message is written to the queue - 2

The screenshot shows the Microsoft Azure portal interface for managing a storage queue named 'bookmarks-post-process'. The queue is currently empty, as indicated by the 'Overview' tab being selected. A single message is listed in the table below:

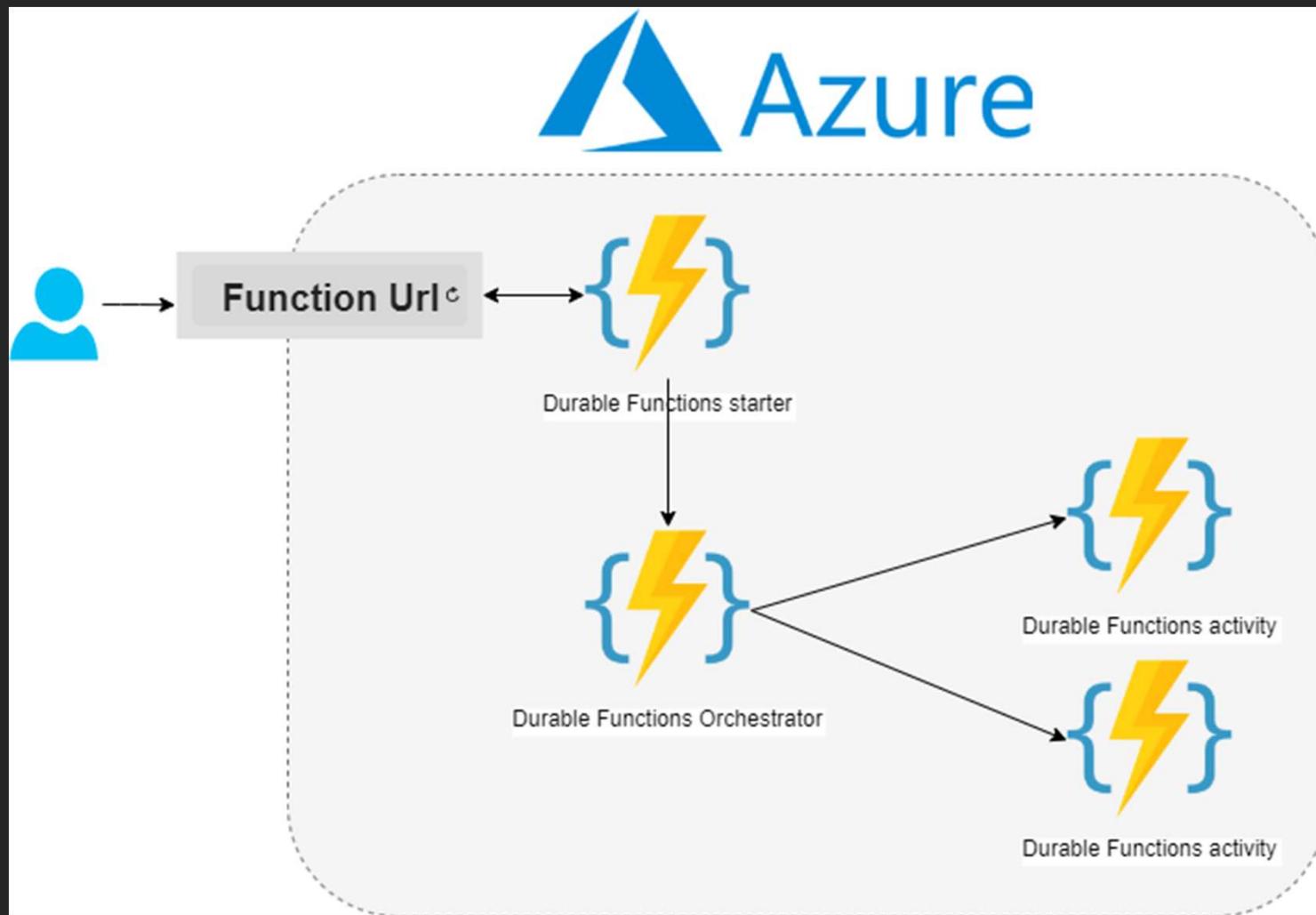
ID	Message text	Insertion time	Expiration time	Dequeue count
fa043a8d-5ac0-47fc-... (redacted)	{ "id": "github", "url": "https://www.github.com" }	11/24/2021, 8:38:28 PM	12/1/2021, 8:38:28 PM	0

The message text contains a JSON object with an 'id' field set to 'github' and a 'url' field set to 'https://www.github.com'. The insertion time is 11/24/2021, 8:38:28 PM, and the expiration time is 12/1/2021, 8:38:28 PM. The dequeue count is 0.

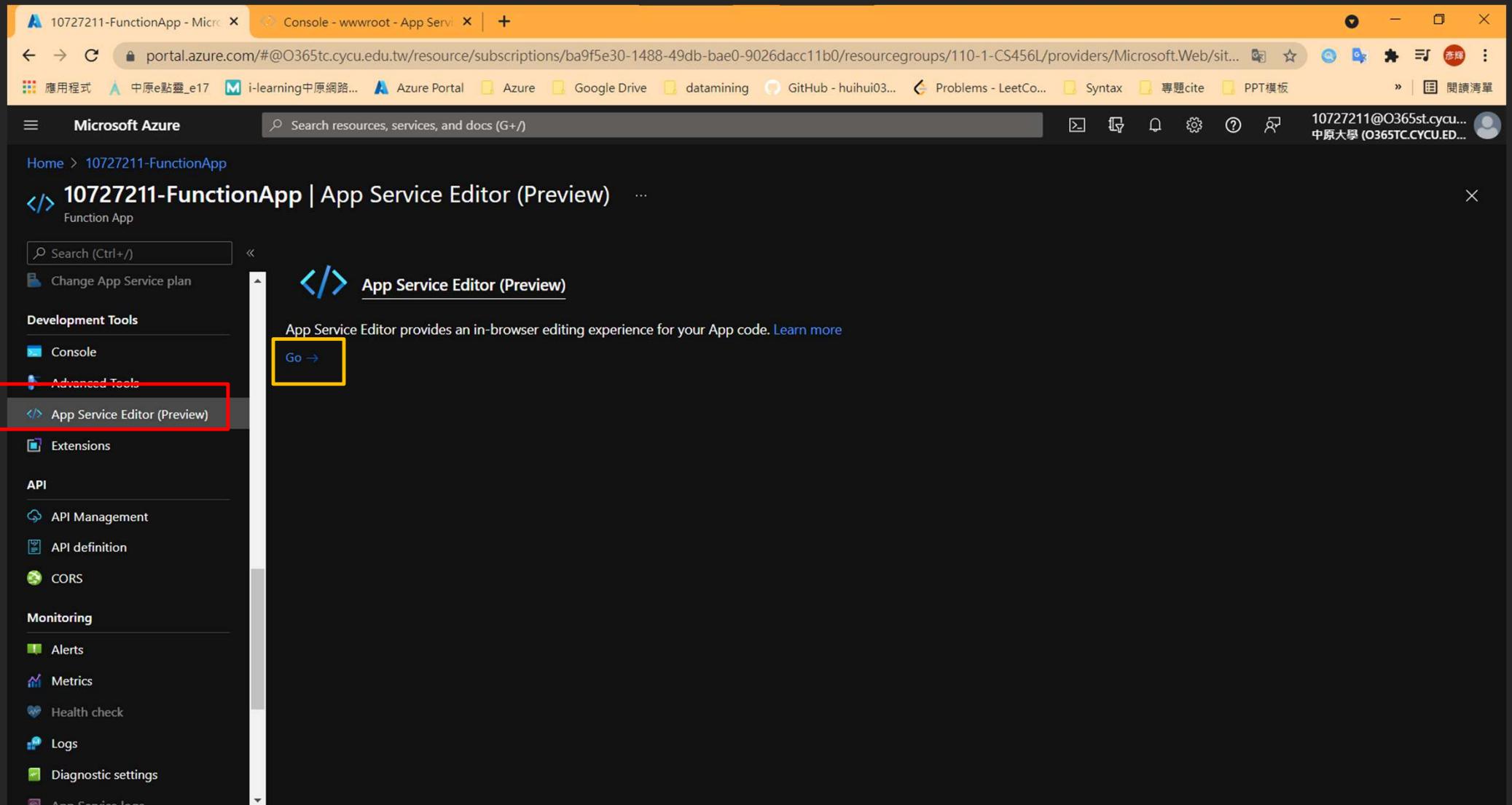
05

// Create a long-running serverless  
workflow with Durable Functions

# Abstract



# Install the durable-functions npm package - 1



# Install the durable-functions npm package - 2

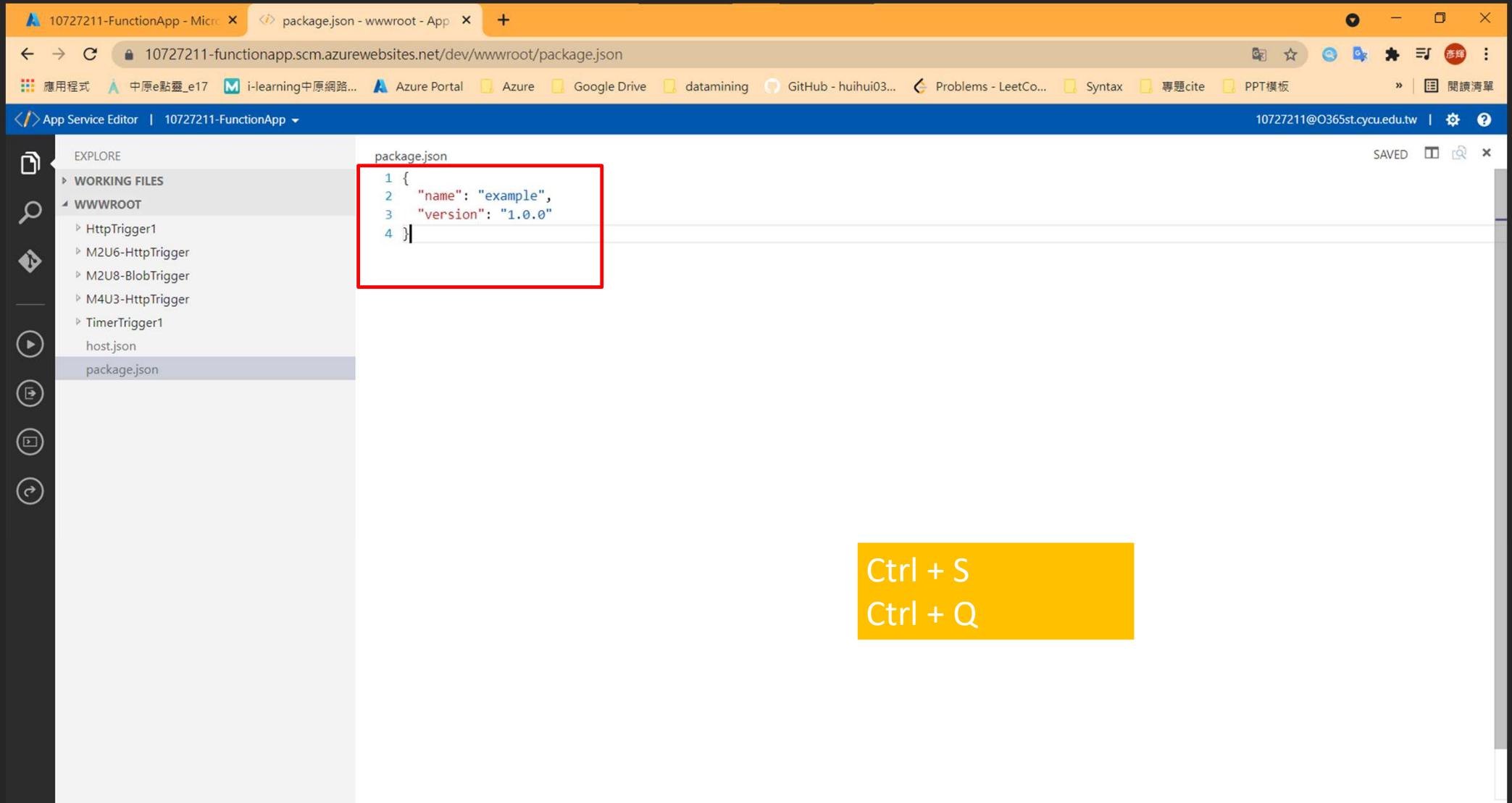
The screenshot shows the Azure App Service Editor interface with a terminal window open. The terminal window title is "Console - wwwroot - App Serv". The URL in the browser is "10727211-functionapp.scm.azurewebsites.net/dev/wwwroot/vs.console/4". The terminal output shows the user navigating to the wwwroot directory and listing files. A red box highlights the "Copy" icon in the toolbar. A yellow box highlights the last two commands entered:

```
\> ls
Volume in drive C is Windows
Volume Serial Number is B815-B799

Directory of C:\home\site\wwwroot

11/23/2021  02:58 PM    <DIR>      .
11/23/2021  02:58 PM    <DIR>      ..
11/23/2021  02:59 PM            141 host.json
11/23/2021  03:06 PM    <DIR>      HttpTrigger1
11/23/2021  04:09 PM    <DIR>      M2U6-HttpTrigger
11/23/2021  04:21 PM    <DIR>      M2U8-BlobTrigger
11/24/2021  11:13 AM    <DIR>      M4U3-HttpTrigger
11/23/2021  03:53 PM    <DIR>      TimerTrigger1
                           1 File(s)           141 bytes
                           7 Dir(s)  5,497,557,417,984 bytes free
\>
\>
\>
\> touch package.json
\> open package.json
```

# Install the durable-functions npm package - 3



The screenshot shows the Azure App Service Editor interface for a function app named "10727211-FunctionApp". The left sidebar lists "WORKING FILES" and "WWWROOT" sections, with "package.json" selected. The main area displays the contents of the package.json file:

```
1 {
2   "name": "example",
3   "version": "1.0.0"
4 }
```

A red box highlights the code block. In the bottom right corner of the editor window, there is a yellow button with the following text:

Ctrl + S  
Ctrl + Q

# Install the durable-functions npm package – 4

The screenshot shows the Microsoft Azure portal interface for a Function App named "10727211-FunctionApp". The left sidebar contains navigation links for Development Tools, API, Monitoring, and Logs. The "Console" link is highlighted with a red box. The main area displays a terminal window titled "10727211-FunctionApp | Console". The terminal shows the command "D:\home\site\wwwroot>npm install durable-functions" being run, followed by several npm warning messages about deprecated packages and missing peer dependencies. The output ends with "D:\home\site\wwwroot>".

```
D:\home\site\wwwroot>ls
HttpTrigger1
M2U6-HttpTrigger
M2U8-BlobTrigger
M4U3-HttpTrigger
TimerTrigger
host.json
package.json

D:\home\site\wwwroot>npm install durable-functions
npm WARN deprecated uid@3.3.3: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN typedoc@0.17.8 requires a peer of typescript@>=3.8.3 but none is installed. You must install peer dependencies yourself.
npm WARN example@1.0.0 No description
npm WARN example@1.0.0 No repository field.
npm WARN example@1.0.0 No license field.

D:\home\site\wwwroot>
```

# Create Durable Functions Http start

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar navigation includes Home, 10727211-FunctionApp, Functions, App keys, App files, Proxies, Deployment slots, Deployment Center, and Settings. The main content area is titled "Create function" and displays a list of function templates. The "Durable Functions HTTP starter" template is selected and highlighted with a yellow box. The "New Function\*" input field contains "M5-HttpStart" and the "Authorization level" dropdown is set to "Function". The "Create" button at the bottom is highlighted with a green box.

Microsoft Azure

Home > 10727211-FunctionApp

10727211-FunctionApp | Functions

Function App

Search (Ctrl+ /)

Create Refresh Delete

Filter by name...

Name ↑

- HttpTrigger1
- M2U6-HttpTrigger
- M2U8-BlobTrigger
- M4U3-HttpTrigger
- TimerTrigger1

Durable Functions Entity HTTP starter  
會在每次收到執行協調器函式的 HTTP 要求時觸發的函式。

**Durable Functions HTTP starter**  
會在每次收到執行協調器函式的 HTTP 要求時觸發的函式.

Durable Functions activity  
會在每次協調器函式呼叫活動時執行的函式。

Durable Functions entity  
用以儲存狀態的 Durable Functions 實體。

Durable Functions orchestrator  
會在序列中叫用活動函式的協調器函式。

Kafka output  
將訊息傳送至指定 Kafka 主題的函式

Kafka trigger  
每次有訊息新增到指定 Kafka 主題時都會執行的函式

RabbitMQ trigger  
每當訊息新增至指定的 RabbitMQ 挪列時，將會執行的函式

SignalR negotiate HTTP trigger  
由 HTTP 觸發的函式。SignalR 用於當前子以呼叫，以開始連線方法。

Template details

We need more information to create the Durable Functions HTTP starter function. [Learn more](#)

New Function\*

授權等級\*①

Create Cancel

# Create the orchestrator function - 1

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar shows the 'Functions' section selected. The main area is titled 'Create function' under 'Select a template'. A yellow box highlights the 'Create' button in the top-left of the main content area. Another yellow box highlights the 'Durable Functions orchestrator' template in the list. A third yellow box highlights the 'M5-OrchFunction' input field for the 'New Function' name. The 'Create' button at the bottom is also highlighted with a green border.

**Create function - Microsoft Azure**

portal.azure.com/#@O365tc.cyku.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourcegroups/110-1-CS456L/providers/Microsoft.Web/sites/10727211-FunctionApp

Microsoft Azure

Home > 10727211-FunctionApp

10727211-FunctionApp | Functions

Function App

Search (Ctrl+ /)

Create Refresh Delete

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Security

Events (preview)

Functions

Functions

App keys

App files

Proxies

Deployment

Deployment slots

Deployment Center

Settings

Create

Refresh

Delete

Filter by name...

Name ↑

HttpTrigger1

M2U6-HttpTrigger

M2U8-BlobTrigger

M4U3-HttpTrigger

M5-HttpStart

TimerTrigger1

Durable Functions Entity HTTP starter

Durable Functions HTTP starter

Durable Functions activity

Durable Functions entity

Durable Functions orchestrator

Kafka output

Kafka trigger

RabbitMQ trigger

SignalR negotiate HTTP trigger

Template details

We need more information to create the Durable Functions orchestrator function. [Learn more](#)

New Function\*

M5-OrchFunction

Create Cancel

# Create the orchestrator function - 2

The screenshot shows the Azure portal interface for the M5-OrchFunction function. The top navigation bar includes links for Microsoft Azure, portal.azure.com, and various application icons. The main header displays "Microsoft Azure" and the current function name, "M5-OrchFunction | Code + Test". Below the header, there are tabs for "Overview", "Code + Test" (which is selected), "Integration", "Monitor", and "Function Keys". The "Code + Test" tab has buttons for "Save" (highlighted with a yellow box), "Discard", "Refresh", "Test/Run", and "Upload". The code editor displays the "index.js" file content:

```
1 const df = require("durable-functions");
2
3 module.exports = df.orchestrator(function*(context) {
4     const outputs = [];
5
6     /*
7      * We will call the approval activity with a reject and an approved to simulate both
8     */
9
10    outputs.push(yield context.df.callActivity("M5-Approval", "Approved"));
11    outputs.push(yield context.df.callActivity("M5-Approval", "Rejected"));
12
13    return outputs;
14});
```

A red box highlights the entire code block, and a blue box highlights the line `outputs.push(yield context.df.callActivity("M5-Approval", "Approved"));`.

# Create the activity function - 1

The screenshot shows the Microsoft Azure portal interface for creating a new function. On the left, the navigation menu is visible with the 'Functions' option selected under '10727211-FunctionApp'. In the center, the 'Create function' dialog is open, showing the 'Select a template' section. A red box highlights the 'Create' button in the top-left corner of the main content area. A yellow box highlights the 'Durable Functions activity' template, which is described as '會在每次協調器函式呼叫活動時執行的函式。' (Will execute the function every time the orchestrator function calls an activity). Below the template list, the 'Template details' section shows the placeholder 'New Function\*' with the value 'M5-Approval' entered. A green box highlights the 'Create' button at the bottom of the dialog.

Home > 10727211-FunctionApp

**10727211-FunctionApp | Functions**

Function App

Search (Ctrl+ /) Create Refresh Delete

Filter by name...

Name ↑

- HttpTrigger1
- M2U6-HttpTrigger
- M2U8-BlobTrigger
- M4U3-HttpTrigger
- M5-HttpStart
- M5-OrchFunction
- TimerTrigger1

**Create function**

**Select a template**

Use a template to create a function. Triggers describe the type of events that invoke your functions. [Learn more](#)

Filter

Template	Description
Durable Functions Entity HTTP starter	會在每次收到執行協調器函式的 HTTP 要求時觸發的函式。
Durable Functions HTTP starter	會在每次收到執行協調器函式的 HTTP 要求時觸發的函式。
<b>Durable Functions activity</b>	<b>會在每次協調器函式呼叫活動時執行的函式。</b>
Durable Functions entity	用以儲存狀態的 Durable Functions 實體。
Durable Functions orchestrator	會在序列中叫用活動函式的協調器函式。
Kafka output	將訊息傳送至指定 Kafka 主題的函式
Kafka trigger	每次有訊息新增到指定 Kafka 主題時都會執行的函式
RabbitMQ trigger	每當訊息新增至指定的 RabbitMQ 佇列時，將會執行的函式
SignalR negotiate HTTP trigger	由 HTTP 觸發的函式，SignalR 用戶端會予以呼叫，以開始連線交涉

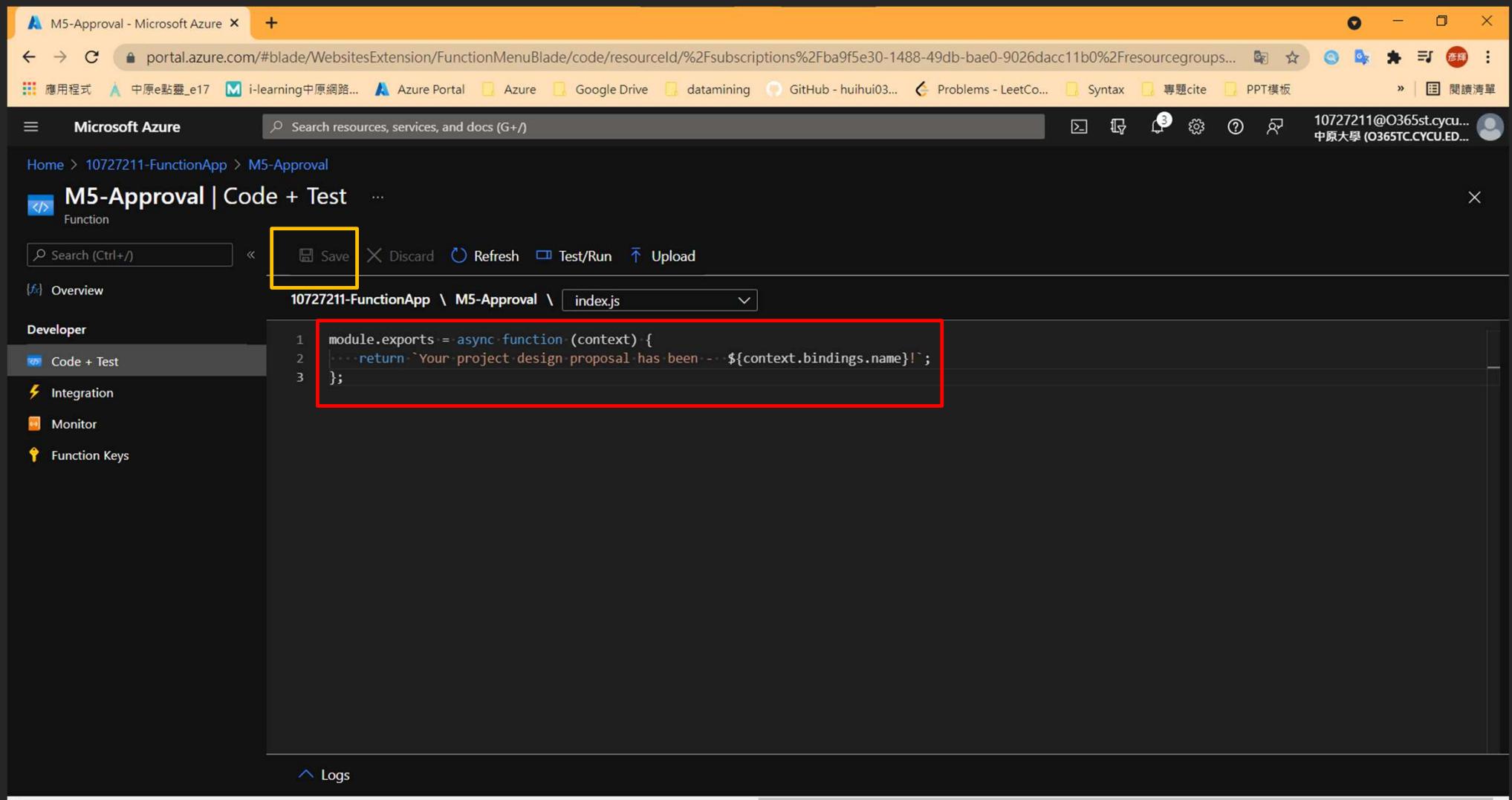
**Template details**

We need more information to create the Durable Functions activity function. [Learn more](#)

New Function\* M5-Approval

Create Cancel

# Create the activity function - 2



The screenshot shows the Microsoft Azure portal interface for a Function App named "M5-Approval". The "Code + Test" tab is selected in the developer sidebar. The main area displays the "index.js" file content:

```
1 module.exports = async function(context) {  
2     return `Your project design proposal has been ${context.bindings.name}!`;  
3 };
```

The second line of code, which contains the string interpolation, is highlighted with a red box. Above the code editor, the "Save" button is highlighted with a yellow box.

# Check Point : HttpStart 、 OrchFunction 、 Approval

The screenshot shows the Microsoft Azure Functions blade for the "10727211-FunctionApp" function app. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, Events (preview), Functions (selected), App keys, App files, Proxies, Deployment slots, Deployment Center, and Settings.

The main area displays a list of functions:

Name	Trigger	Status
HttpTrigger1	HTTP	Enabled
M2U6-HttpTrigger	HTTP	Enabled
M2U8-BlobTrigger	Blob	Enabled
M4U3-HttpTrigger	HTTP	Enabled
M5-Approval	Activity	Enabled
M5-HttpStart	HTTP	Enabled
M5-OrchFunction	Orchestration	Enabled
TimerTrigger1	Timer	Enabled

The functions M5-Approval, M5-HttpStart, and M5-OrchFunction are highlighted with a red box around their row in the table.

# Verify that the durable functions workflow starts - 1

The screenshot shows the Microsoft Azure portal interface. The top navigation bar has a tab labeled "M5-HttpStart - Microsoft Azure". Below the navigation bar, there's a search bar and a list of recent items including "中原e點靈\_e17", "i-learning中原網路...", "Azure Portal", "Google Drive", "datamining", "GitHub - huihui03...", "Problems - LeetCo...", "Syntax", "專題cite", "PPT模板", and "10727211@O365st.cy... 中原大學 (O365TC.CYCU.ED...)". The main content area shows a "Microsoft Azure" header with "Home > 10727211-FunctionApp > M5-HttpStart". Below this, there are buttons for "Enable", "Disable", "Delete", "Get Function Url" (which is highlighted with a yellow box), and "Refresh". A "Get Function Url" dialog box is open, showing a dropdown menu set to "default (function key)" and a URL field containing "https://10727211-functionapp.azurewebsites.net/api/orchestrators/{functionName}?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUTGWW...". There is also a "Copy" button next to the URL field, which is also highlighted with a yellow box. At the bottom of the dialog, there are "OK" and "Subscription (Move)" buttons.

https://10727211-  
functionapp.azurewebsites.net/api/orchestrators/{functionName}?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUT  
GWWRd8s3aYAcJlgVEiKHEfnQ==

https://10727211-functionapp.azurewebsites.net/api/orchestrators/M5-  
OrchFunction?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUTGWWRd8s3aYAcJlgVEiKHEfnQ==

## Verify that the durable functions workflow starts - 2

The screenshot shows a browser window with three tabs open:

- M5-HttpStart - Microsoft Azure
- https://10727211-functionapp.x
- https://10727211-functionapp.x

The third tab displays a JSON object representing a durable function instance. A red box highlights the first few lines of the JSON output:

```
{  
  "id": "87842fc008b3425eacbd9095f6417461",  
  "statusQueryGetUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461?",  
  "taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
  "sendEventPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461/raiseEvent/{eventName}?",  
  "taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
  "terminatePostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461/terminate?reason={text}&taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
  "rewindPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461/rewind?reason={text}&taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
  "purgeHistoryDeleteUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461?",  
  "taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
  "restartPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461/restart?",  
  "taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA="}
```

# Verify that the durable functions workflow starts - 3

The screenshot shows a browser window with three tabs open:

- M5-HttpStart - Microsoft Azure
- https://10727211-functionapp - (selected tab)
- https://10727211-functionapp

The URL in the selected tab is <https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/87842fc008b3425eacbd9095f6417461?taskHub=t10727211FunctionApp&connecti...>. The page content displays the following JSON response:

```
{"name": "M5-OrchFunction", "instanceId": "87842fc008b3425eacbd9095f6417461", "runtimeStatus": "Completed", "input": null, "customStatus": null, "output": ["Your project design proposal has been - Approved!", "Your project design proposal has been - Rejected!"], "createdTime": "2021-11-24T13:51:06Z", "lastUpdatedTime": "2021-11-24T13:51:11Z"}
```

# Add moment npm package to your function app

The screenshot shows the Microsoft Azure portal interface for a Function App named "10727211-FunctionApp". The left sidebar contains navigation links for App Service plan, Quotas, Change App Service plan, Development Tools (Console selected), Advanced Tools, App Service Editor (Preview), Extensions, API (API Management, API definition), and Monitoring (Alerts, Metrics, Health check). The main area is titled "10727211-FunctionApp | Console" and displays a terminal window. The terminal output shows the following commands and their results:

```
C:\home\site\wwwroot>pwd
/c/home/site/wwwroot

C:\home\site\wwwroot>npm install typescript
npm WARN example@1.0.0 No description
npm WARN example@1.0.0 No repository field.
npm WARN example@1.0.0 No license field.

C:\home\site\wwwroot>npm install moment
npm WARN example@1.0.0 No description
npm WARN example@1.0.0 No repository field.
npm WARN example@1.0.0 No license field.

C:\home\site\wwwroot>
```

The first command, "npm install typescript", is highlighted with a red border, and the second command, "npm install moment", is highlighted with a yellow border.

# Add an escalation activity to your function app - 1

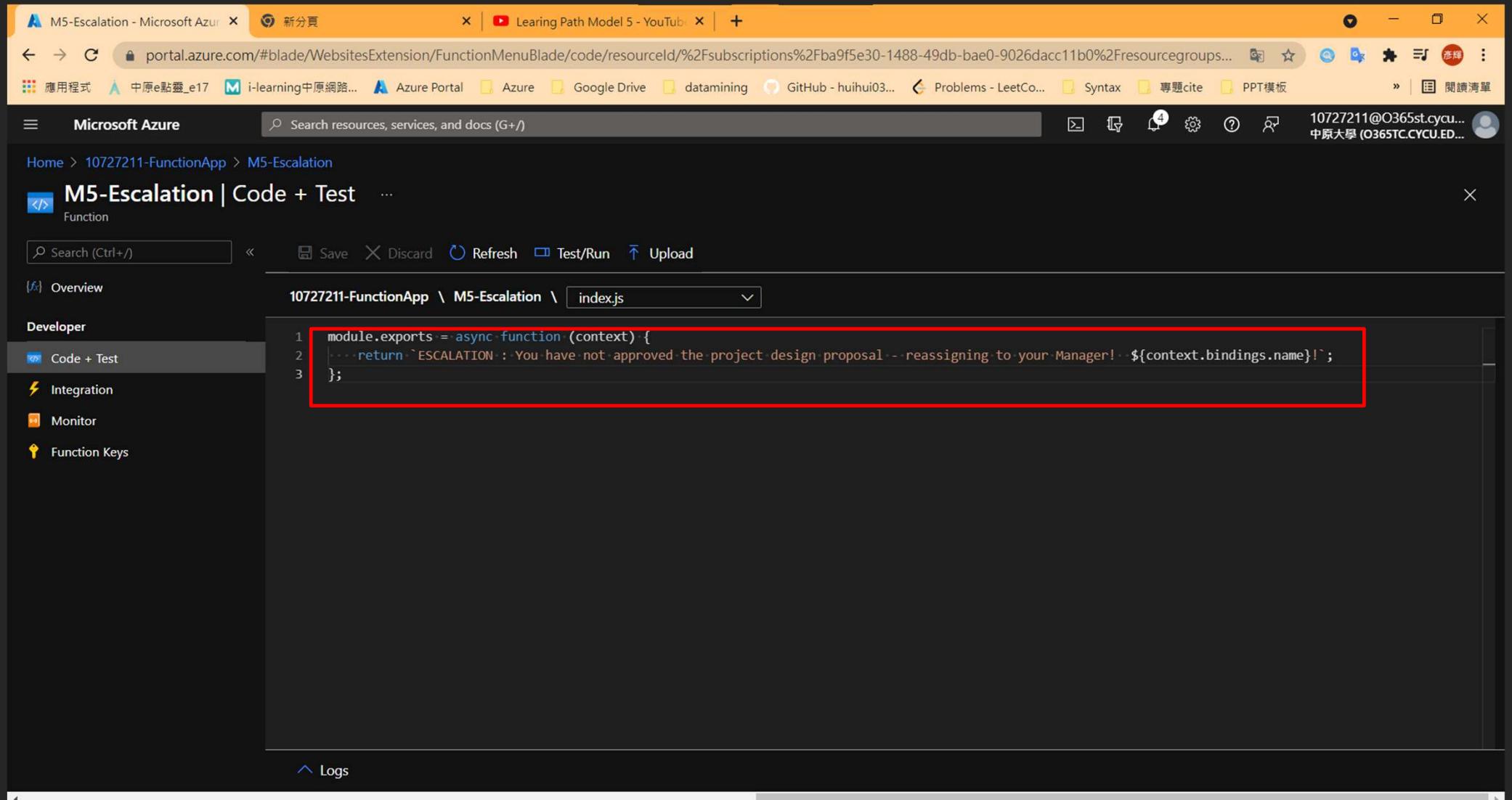
The screenshot shows the Microsoft Azure portal interface for creating a new function. On the left, the sidebar is visible with sections like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Security, Events (preview), Functions (selected), App keys, App files, Proxies, Deployment (Deployment slots, Deployment Center), and Settings.

The main area is titled "Create function" and "Select a template". A red box highlights the "+ Create" button in the top-left of the list view. A yellow box highlights the "Durable Functions activity" template in the list, which is described as "會在每次協調器函式呼叫活動時執行的函式。" (Will execute the function every time the orchestrator function calls the activity).

In the "Template details" section, a green box highlights the "New Function\*" input field where "M5-Escalation" is typed. The "Create" button at the bottom is also highlighted with a green border.

At the very top of the browser window, there are tabs for "Create function - Microsoft Azure" and "Learing Path Model 5 - YouTube", along with the address bar showing the URL of the Azure portal.

# Add an escalation activity to your function app - 2

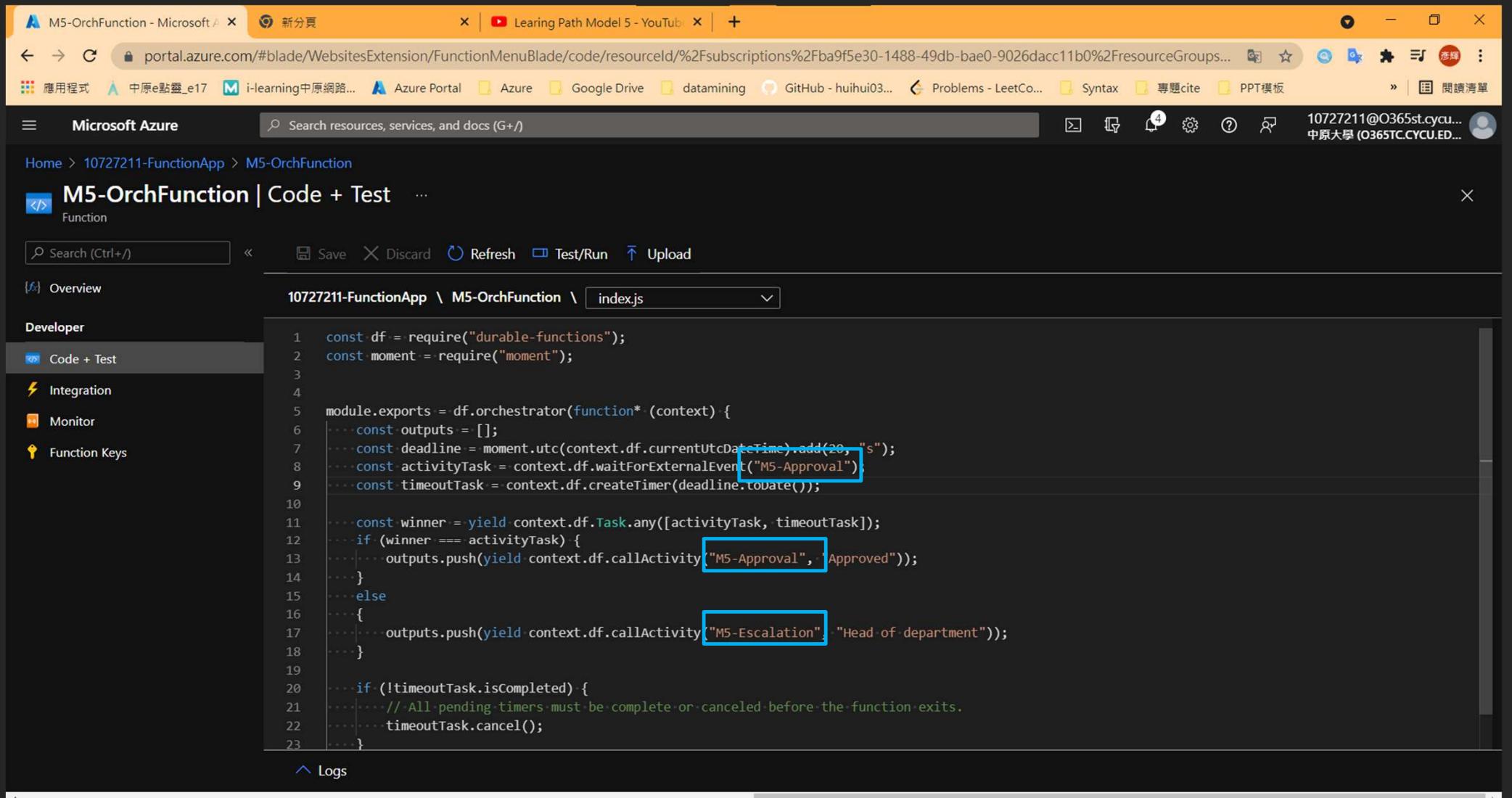


The screenshot shows the Microsoft Azure portal interface for a Function App named "M5-Escalation". The code editor displays the following JavaScript code:

```
1 module.exports = async function (context) {
2     // ...return `ESCALATION: You have not approved the project design proposal -- reassigning to your Manager! - ${context.bindings.name}`;
3 };
```

The code is highlighted with a red box, specifically around the second line which contains a comment about an unapproved project design proposal.

# Update the orchestration function to use the escalation function



The screenshot shows the Microsoft Azure portal interface for managing a Function App named "M5-OrchFunction". The current view is the "Code + Test" tab for the "index.js" file. The code editor displays the following JavaScript code:

```
1 const df = require("durable-functions");
2 const moment = require("moment");
3
4 module.exports = df.orchestrator(function*(context) {
5     const outputs = [];
6     const deadline = moment.utc(context.df.currentUtcDateTime).add(20, "s");
7     const activityTask = context.df.waitForExternalEvent("M5-Approval");
8     const timeoutTask = context.df.createTimer(deadline.toDate());
9
10    const winner = yield context.df.Task.any([activityTask, timeoutTask]);
11    if (winner === activityTask) {
12        outputs.push(yield context.df.callActivity("M5-Approval", "Approved"));
13    }
14    else {
15        outputs.push(yield context.df.callActivity("M5-Escalation", "Head of department"));
16    }
17    if (!timeoutTask.isCompleted) {
18        // All pending timers must be complete or canceled before the function exits.
19        timeoutTask.cancel();
20    }
21});
```

The code uses the Durable Functions library to define an orchestrator function. It waits for an external event ("M5-Approval") and a timer to trigger. If the approval event occurs first, it calls the "M5-Approval" activity with the parameter "Approved". If the timer triggers first, it calls the "M5-Escalation" activity with the parameter "Head of department". Both activities are defined elsewhere in the function app.

# Verify that the Durable Functions workflow starts - 1

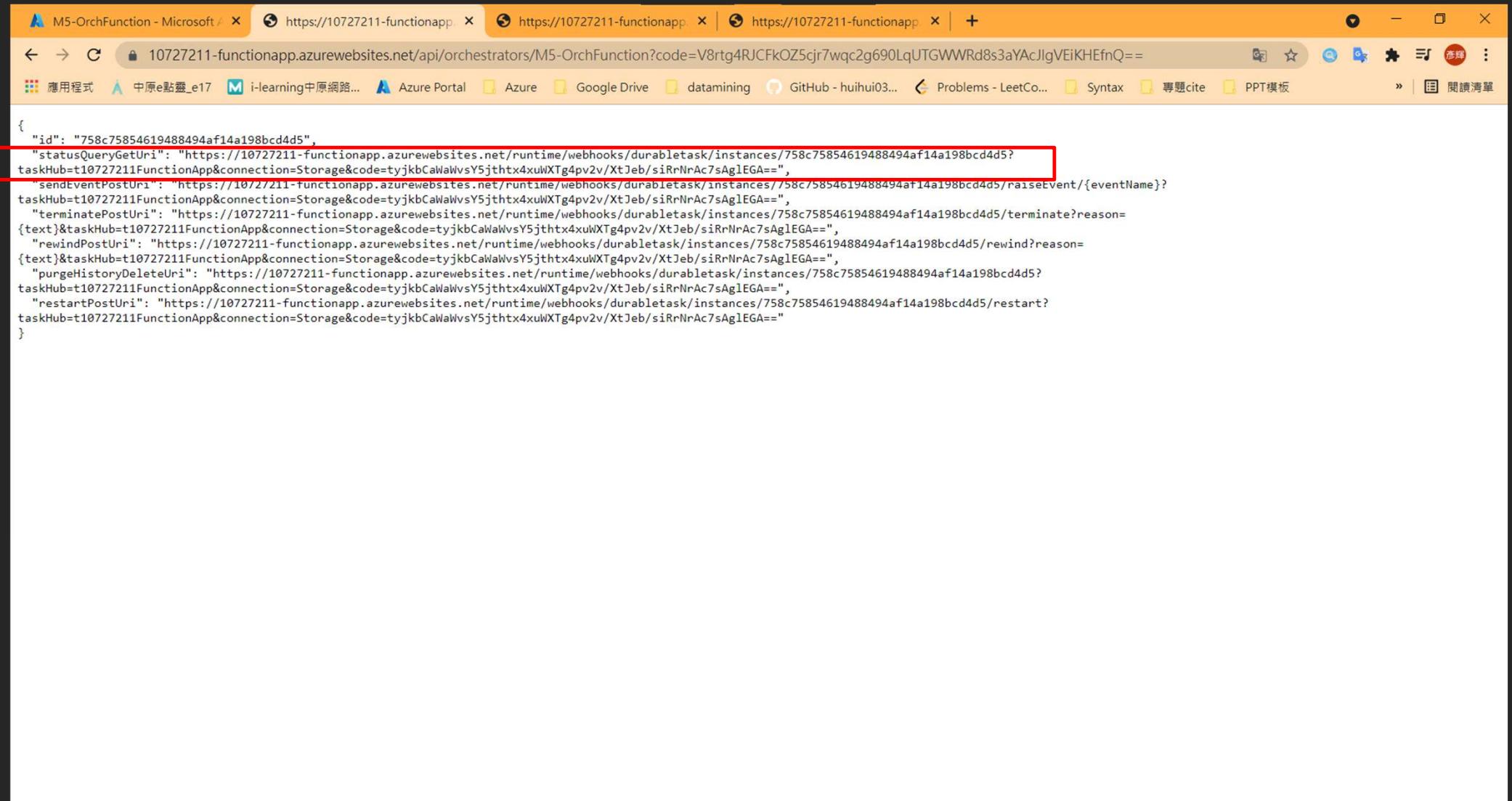
The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes links for portal.azure.com, Microsoft Azure, and various application icons. The main page title is "M5-HttpStart - Microsoft Azure". Below the title, there's a breadcrumb navigation: Home > 10727211-FunctionApp > M5-HttpStart. The left sidebar has sections for Overview, Developer, Code + Test, Integration, Monitor, and Function Keys. The "Overview" section is currently selected. A search bar at the top says "Search resources, services, and docs (G+)". Below the search bar are buttons for Enable, Disable, Delete, Get Function Url, and Refresh. A modal window titled "Get Function Url" is open, showing a dropdown menu set to "default (function key)" and a URL field containing "https://10727211-functionapp.azurewebsites.net/api/orchestrators/{functionName}?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUTGWW...". There are "OK" and "Cancel" buttons at the bottom of the modal. At the very bottom of the portal page, there's a status bar with icons for battery, signal, and other system information.

https://10727211-  
functionapp.azurewebsites.net/api/orchestrators/{functionName}?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUT  
GWWRD8s3aYAcJlgVEiKHEfnQ==



https://10727211-functionapp.azurewebsites.net/api/orchestrators/M5-  
OrchFunction?code=V8rtg4RJCFkOZ5cjr7wqc2g690LqUTGWWRD8s3aYAcJlgVEiKHEfnQ==

## Verify that the Durable Functions workflow starts - 2



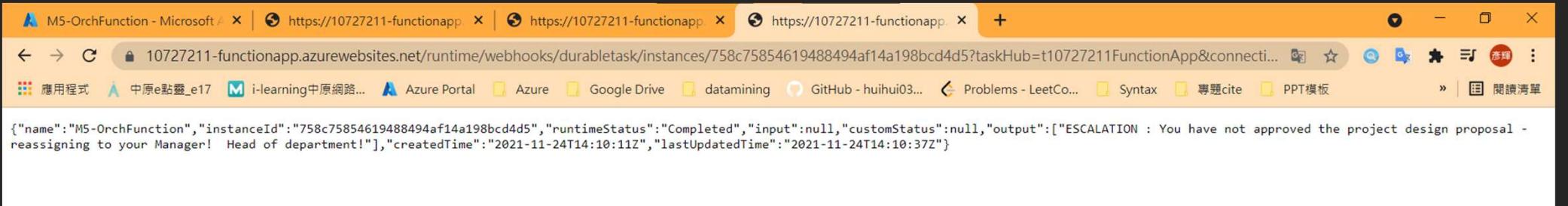
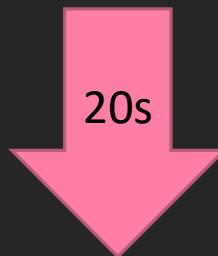
```
{  
    "id": "758c75854619488494af14a198bcd4d5",  
    "statusQueryGetUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5?  
taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
    "sendEventPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5/raiseEvent/{eventName}?  
taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
    "terminatePostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5/terminate?reason=  
{text}&taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
    "rewindPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5/rewind?reason=  
{text}&taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
    "purgeHistoryDeleteUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5?  
taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA=",  
    "restartPostUri": "https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5/restart?  
taskHub=t10727211FunctionApp&connection=Storage&code=tyjkbCaWaWvsY5jthtx4xuWXTg4pv2v/XtJeb/siRrNrAc7sAg1EGA="}  
}
```

# Verify that the Durable Functions workflow starts - 3



A screenshot of a Microsoft Edge browser window. The address bar shows four tabs, all pointing to `https://10727211-functionapp.azurewebsites.net/runtime/webhooks/durabletask/instances/758c75854619488494af14a198bcd4d5?taskHub=t10727211FunctionApp&connecti...`. The main content area displays the following JSON response:

```
{"name": "M5-OrchFunction", "instanceId": "758c75854619488494af14a198bcd4d5", "runtimeStatus": "Running", "input": null, "customStatus": null, "output": null, "createdTime": "2021-11-24T14:10:11Z", "lastUpdatedTime": "2021-11-24T14:10:11Z"}
```



A screenshot of a Microsoft Edge browser window, identical to the one above but showing the result after a 20-second delay. The address bar and tabs are the same. The main content area now displays a more complex JSON response:

```
{"name": "M5-OrchFunction", "instanceId": "758c75854619488494af14a198bcd4d5", "runtimeStatus": "Completed", "input": null, "customStatus": null, "output": ["ESCALATION : You have not approved the project design proposal - reassigning to your Manager! Head of department!"], "createdTime": "2021-11-24T14:10:11Z", "lastUpdatedTime": "2021-11-24T14:10:37Z"}
```

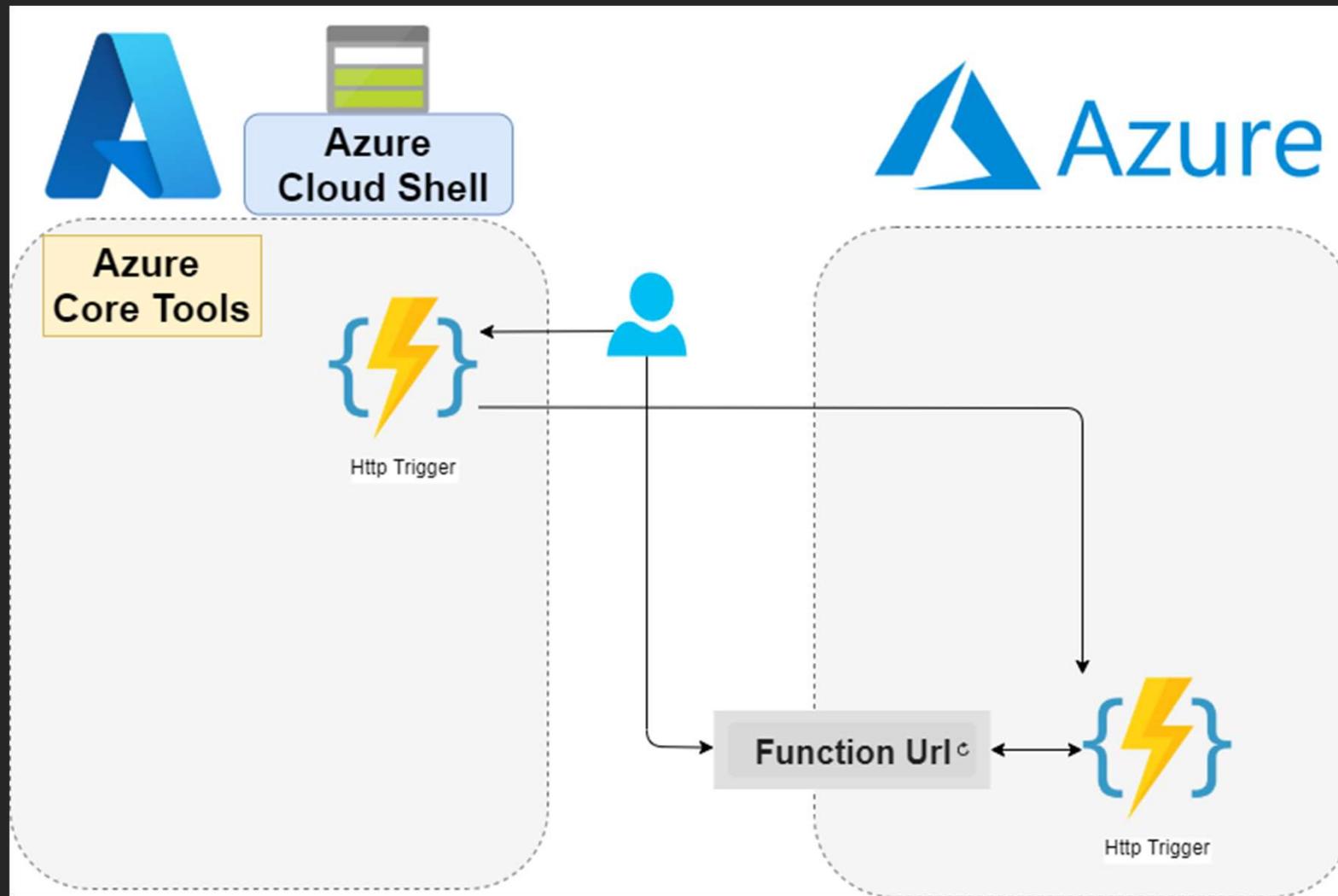
# 06 // Develop, test, and publish Azure Functions by using Azure Functions Core Tools

# Starter

Warning : Azure Cloud Shell 無法實現功能 · 請開啟Sandbox

網址：<https://docs.microsoft.com/en-us/learn/modules/develop-test-deploy-azure-functions-with-core-tools/3-exercise-create-function-core-tools>

# Abstract



# Create a local Azure Functions project

```
azureuser@Azure:~$ mkdir ~/loan-wizard
azureuser@Azure:~$ cd ~/loan-wizard
azureuser@Azure:~/loan-wizard$ func init
Select a number for worker runtime:
1. dotnet
2. node
3. python
4. powershell
5. custom
Choose option: 2
node
Select a number for language:
1. javascript
2. typescript
Choose option: 1
javascript
Writing package.json
Writing .gitignore
Writing host.json
Writing local.settings.json
Writing /home/azureuser/loan-wizard/.vscode/extensions.json
```

# Create an HTTP-triggered function

```
azureuser@Azure:~/loan-wizard$ func new ←  
Select a number for template:  
1. Azure Blob Storage trigger  
2. Azure Cosmos DB trigger  
3. Durable Functions activity  
4. Durable Functions HTTP starter  
5. Durable Functions orchestrator  
6. Azure Event Grid trigger  
7. Azure Event Hub trigger  
8. HTTP trigger ←  
9. IoT Hub (Event Hub)  
10. Kafka output  
11. Kafka trigger  
12. Azure Queue Storage trigger  
13. RabbitMQ trigger  
14. SendGrid  
15. Azure Service Bus Queue trigger  
16. Azure Service Bus Topic trigger  
17. SignalR negotiate HTTP trigger  
18. Timer trigger  
Choose option: 8 ←  
HTTP trigger  
Function name: [HttpTrigger] simple-interest ←  
Writing /home/azureuser/loan-wizard/simple-interest/index.js  
Writing /home/azureuser/loan-wizard/simple-interest/function.json
```

# Replace index.js

The screenshot shows the Azure Cloud Shell interface. On the left, a file tree displays a local repository structure:

- 檔案
- .vscode
- simple-interest
  - function.json
  - index.js
- .gitignore
- host.json
- local.settings.json
- package.json

The `index.js` file is open in the code editor, which has a yellow border around its content area. The code is as follows:

```
1 module.exports = async function(context, req) {
2     // Try to grab principal, rate, and term from the query string
3     // parse them as numbers
4     const principal = parseFloat(req.query.principal);
5     const rate = parseFloat(req.query.rate);
6     const term = parseFloat(req.query.term);
7
8     if ([principal, rate, term].some isNaN)) {
9         // If any empty or non-numeric values, return a 400 response
10        // error message
11        context.res = {
12            status: 400,
13            body: "Please supply principal, rate and term in the query
14        };
15    } else {
16        // Otherwise set the response body to the product of the three
17        context.res = { body: principal * rate * term };
18    }
19}
```

At the bottom, the terminal window shows the command `code .` being run, with the entire command highlighted by a red box.

**Ctrl+S**  
**Ctrl+Q**

# Run the function locally - 1

```
⟳ | Azure Cloud Shell
azureuser@Azure:~/loan-wizard$ func start ←
Azure Functions Core Tools (2.7.2936 Commit hash: c00b06616a741dcdf70a3061fc415adde8
4010f8)
Function Runtime Version: 2.0.14494.0
AZURE_FUNCTIONS_ENVIRONMENT: Development
Hosting environment: Development
Content root path: /home/azureuser/loan-wizard
Now listening on: http://0.0.0.0:7071
Application started. Press Ctrl+C to shut down.

Functions:

    simple-interest: [GET,POST] http://localhost:7071/api/simple-interest

For detailed output, run func with --verbose flag.
[2021-11-24T14:52:09.860] Worker process started and initialized.
[2021-11-24T14:52:14.627] Host lock lease acquired by instance ID '0000000000000000
0000000EAFDF55D'.
^CApplication is shutting down...
```

## Run the function locally - 2

```
azureuser@Azure:~/loan-wizard$ func start &> ~/output.txt & ←  
[1] 261  
azureuser@Azure:~/loan-wizard$ curl "http://localhost:7071/api/simple-interest" -w "  
\n" ←  
Please supply principal, rate and term in the query string  
azureuser@Azure:~/loan-wizard$ curl "http://localhost:7071/api/simple-interest?princ  
ipal=5000&rate=.035&term=36" -w "\n" ←  
6300  
azureuser@Azure:~/loan-wizard$ pkill func ←  
azureuser@Azure:~/loan-wizard$ ←  
[1]+ Done func start &> ~/output.txt ←  
azureuser@Azure:~/loan-wizard$ code ~/output.txt ←
```

# Output.txt

⟳ | Azure Cloud Shell

檔案 檔案

.vscode

simple-interest

function.json

index.js

.gitignore

host.json

local.settings.json

package.json

output.txt

```
1 Azure Functions Core Tools (2.7.2936 Commit hash: c00b06616a741dc...| ...
2 Function Runtime Version: 2.0.14494.0
3 AZURE_FUNCTIONS_ENVIRONMENT: Development
4 Hosting environment: Development
5 Content root path: /home/azureuser/loan-wizard
6 Now listening on: http://0.0.0.0:7071
7 Application started. Press Ctrl+C to shut down.

9 Functions:
10
11     simple-interest: [GET,POST] http://localhost:7071/api/simple...
12
13 For detailed output, run func with --verbose flag.
14 [2021-11-24T14:53:02.421] Worker process started and initialized.
15 [2021-11-24T14:53:03.266] Executing 'Functions.simple-interest' (|
16 [2021-11-24T14:53:03.501] Executed 'Functions.simple-interest' (|
17 [2021-11-24T14:53:07.259] Host lock lease acquired by instance ID |
18 [2021-11-24T14:53:08.759] Executing 'Functions.simple-interest' (|
19 [2021-11-24T14:53:08.766] Executed 'Functions.simple-interest' (| ...
```

# Create a function app

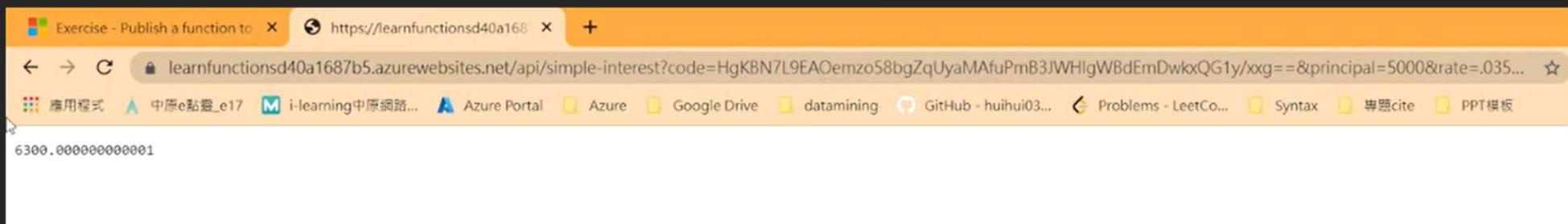
```
azureuser@Azure:~/loan-wizard$ RESOURCEGROUP="learn-0f6b9075-14e1-4f58-9cb9-cd8a0f437c8a"
azureuser@Azure:~/loan-wizard$ STORAGEACCT=learnstorage$(openssl rand -hex 5)
azureuser@Azure:~/loan-wizard$ FUNCTIONAPP=learnfunctions$(openssl rand -hex 5)
azureuser@Azure:~/loan-wizard$ 
azureuser@Azure:~/loan-wizard$ az storage account create \
>   --resource-group "$RESOURCEGROUP" \
>   --name "$STORAGEACCT" \
>   --kind StorageV2 \
>   --location centralus
```

```
azureuser@Azure:~/loan-wizard$ az functionapp create \
>   --resource-group "$RESOURCEGROUP" \
>   --name "$FUNCTIONAPP" \
>   --storage-account "$STORAGEACCT" \
>   --runtime node \
>   --consumption-plan-location centralus \
>   --functions-version 3
```

# Publish to Azure

```
azureuser@Azure:~/loan-wizard$ cd ~/loan-wizard ←
azureuser@Azure:~/loan-wizard$ func azure functionapp publish "$FUNCTIONAPP" ←
You're trying to publish to a non-v2 function app from v2 tooling.
You can pass --force to force update the app to v2, or switch to v1 or v3 tooling for
publishing
azureuser@Azure:~/loan-wizard$ func azure functionapp publish "$FUNCTIONAPP" --force ←
Getting site publishing info...
Creating archive for current directory...
Uploading 1.35 KB [#####
Upload completed successfully.
Deployment completed successfully.
Syncing triggers...
Functions in learnfunctionsd40a1687b5:
    simple-interest - [httpTrigger]
        Invoke url: https://learnfunctionsd40a1687b5.azurewebsites.net/api/simple-int
erest?code=HgKBN7L9EA0emzo58bgZqUyaMAfuPmB3JWHlgWBdEmDwkxQG1y/xxg==
```

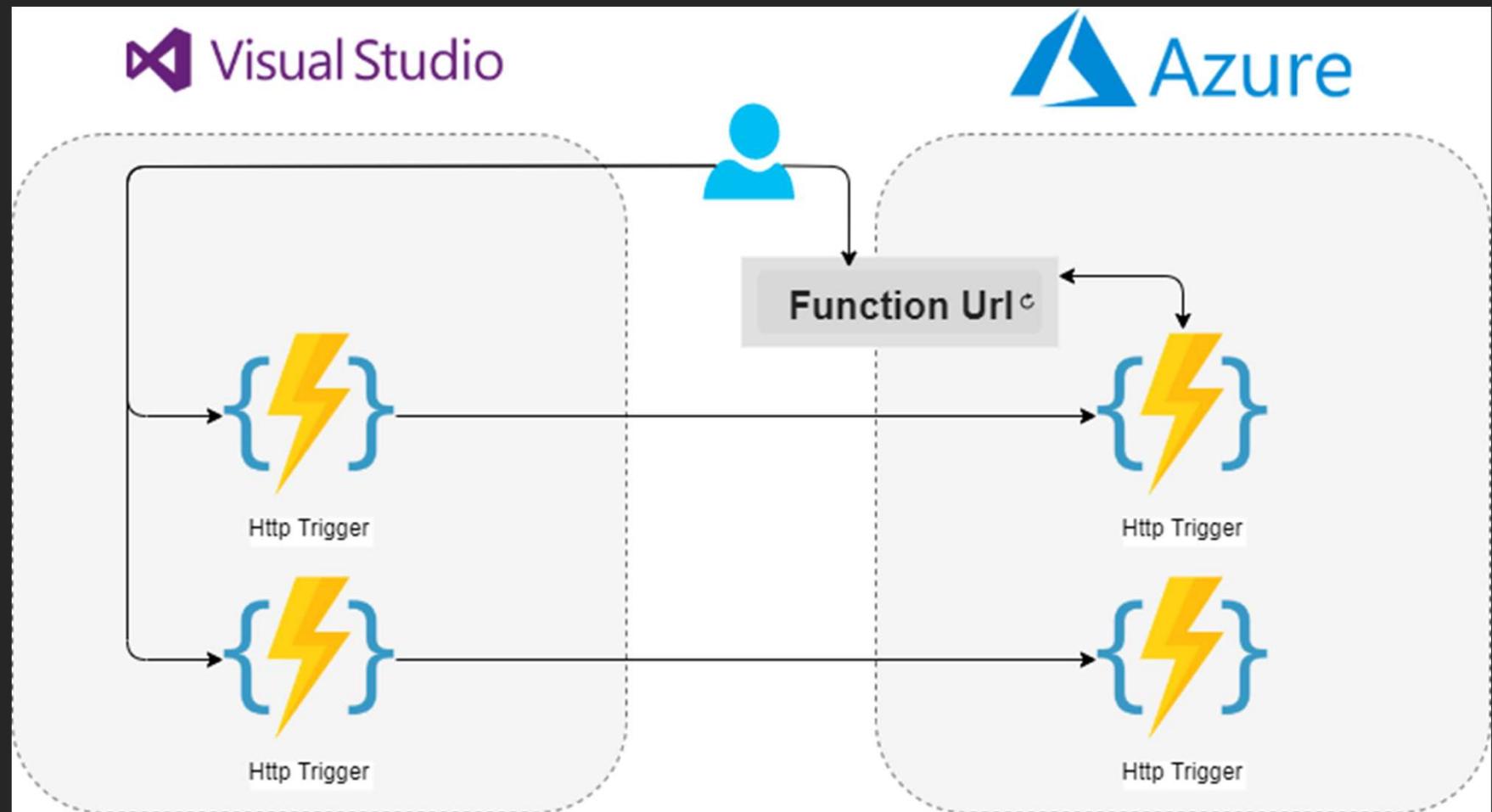
# Run the function



07

// Develop, test, and deploy an Azure  
Function with Visual Studio

# Abstract



# Starter - Requirement

Install Visual Studio & package

# Open Visual Studio Installer - 1

Visual Studio Installer

已安裝 可用

**Visual Studio Community 2019**  
16.11.7  
功能強大的 IDE、免費供學生、開放原始碼參與者及個人使用  
[版本資訊](#)

修改(M) 修改(M)

啟動(L)

更多 ▾

**開發人員新聞**

Visual Studio 2022 Launch videos available on-demand  
Visual Studio 2022 Launch videos are now availab...  
2021年11月9日

Visual Studio 2022 is now available  
We've reached general availability for Visual Studi...  
2021年11月8日

Announcing .NET 6 – The Fastest .NET Yet  
.NET 6 is now available. It is easier to use, runs fas...  
2021年11月8日

[檢視更多 Microsoft 開發人員新聞...](#)

需要協助嗎? 查看 [Microsoft 開發人員社群](#) 或是  
透過 [Visual Studio 支援](#) 與我們連絡。

安裝程式版本 2.11.52.58712

# Open Visual Studio Installer - 2

The screenshot shows the Visual Studio Installer interface. The top navigation bar includes 'Visual Studio Installer', '修改中 — Visual Studio Community 2019 — 16.11.7', and tabs for '工作負載', '個別元件', '語言套件', and '安裝位置'. The main content area is divided into sections: 'Web 與雲端 (4)' and '傳統型與行動裝置 (5)'. In the 'Web 與雲端' section, 'ASP.NET 與網頁程式開發' and 'Azure 開發' are highlighted with red boxes around their descriptions and checkboxes. In the '傳統型與行動裝置' section, '.NET 桌面開發' and '使用 C++ 的桌面開發' are shown. A yellow callout box labeled '在此步驟下載相關套件' is positioned over the right side of the installer window. The bottom navigation bar includes '位置' (C:\Program Files (x86)\Microsoft Visual Studio\2019\Community), '繼續進行即表示您同意所選 Visual Studio 版本的授權。我們也可讓您使用 Visual Studio 下載其他軟體。此軟體為分開授權，如同協力廠商聲明或其隨附的授權中所述。繼續進行即表示您也同意該授權。', '總共所需空間 24 MB', '在下載時安裝', and '關閉(C)'.

修改中 — Visual Studio Community 2019 — 16.11.7

工作負載 個別元件 語言套件 安裝位置

Web 與雲端 (4)

ASP.NET 與網頁程式開發  
使用 ASP.NET Core、ASP.NET、HTML/JavaScript 及容器 (包括 Docker 支援) 建立 Web 應用程式。

Azure 開發  
用於使用 .NET 和 .NET Framework 開發雲端應用程式及建立資源的 Azure SDK、工具及專案。同時包含用於將應用...

Python 開發  
對 Python 進行編輯、偵錯、互動式開發及原始檔控制。

Node.js 開發  
使用非同步的事件驅動 JavaScript 執行階段 Node.js 建置可調整的網路應用程式。

傳統型與行動裝置 (5)

.NET 桌面開發  
使用 C#、Visual Basic 及 F#，利用 .NET 和 .NET Framework 建置 WPF、Windows Forms 與主控台應用程式。

使用 C++ 的桌面開發  
使用您選擇的工具 (包括 MSVC、Clang、CMake 或 MSBuild)，建置適用於 Windows 的新式 C++ 應用程式。

通用 Windows 平台開發  
使用 C#、VB 或選用 C++，來建立適用於通用 Windows 平台的應用程式。

使用 .NET 進行行動開發  
使用 Xamarin 建置適用於 iOS、Android 或 Windows 的跨平台應用程式。

在此步驟下載相關套件

位置 C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

繼續進行即表示您同意所選 Visual Studio 版本的授權。我們也可讓您使用 Visual Studio 下載其他軟體。此軟體為分開授權，如同協力廠商聲明或其隨附的授權中所述。繼續進行即表示您也同意該授權。

總共所需空間 24 MB

在下載時安裝 關閉(C)

# Create an Azure Function App - 1

Visual Studio Installer

已安裝 可用

 Visual Studio Community 2019  
16.11.7  
功能強大的 IDE、免費供學生、開放原始碼參與者及個人使用  
[版本資訊](#)

修改(M)  
啟動(L)  
更多 ▾

開發人員新聞

Visual Studio 2022 Launch videos available on-demand  
Visual Studio 2022 Launch videos are now availab...  
2021年11月9日

Visual Studio 2022 is now available  
We've reached general availability for Visual Studi...  
2021年11月8日

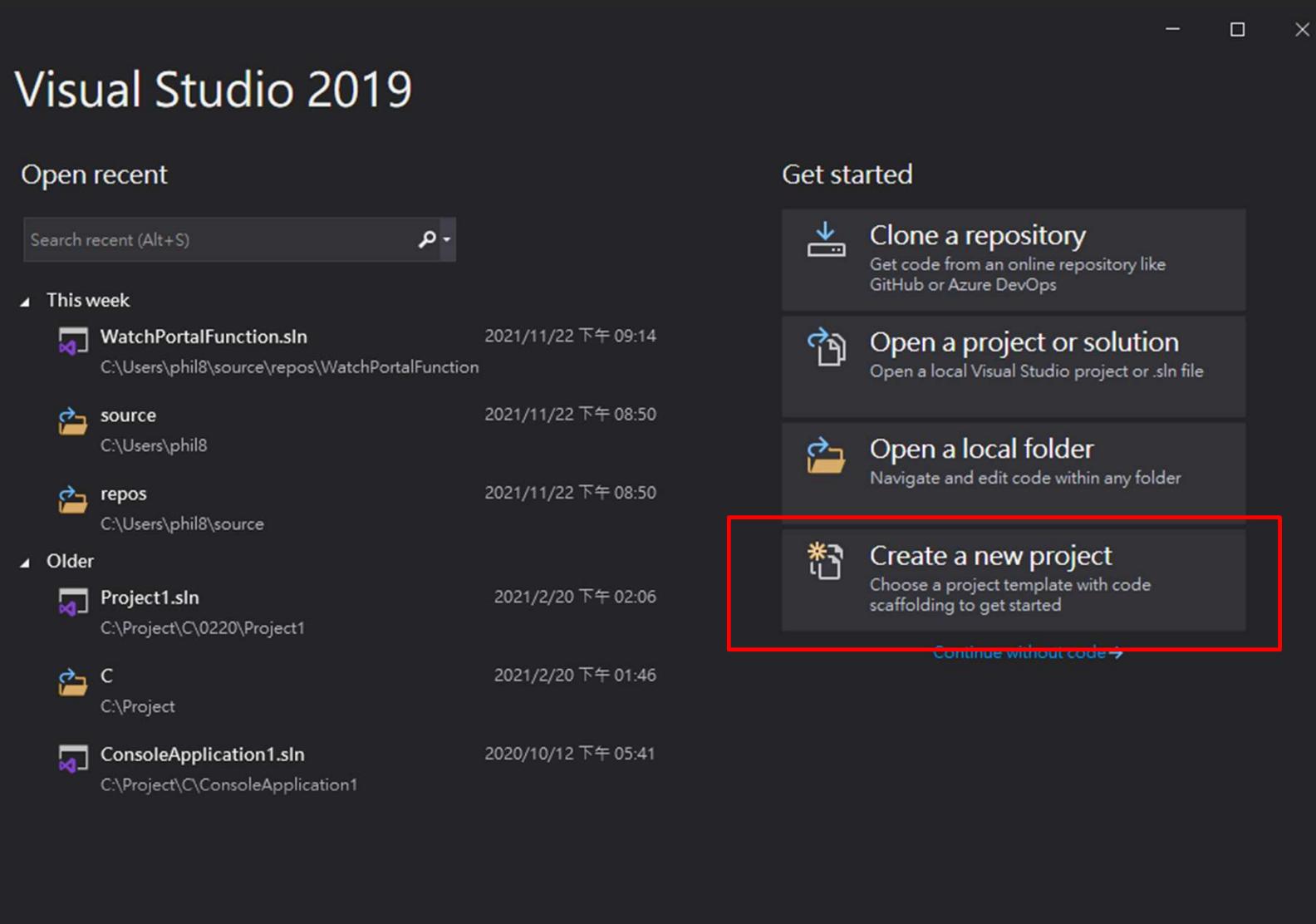
Announcing .NET 6 – The Fastest .NET Yet  
.NET 6 is now available. It is easier to use, runs fas...  
2021年11月8日

[檢視更多 Microsoft 開發人員新聞...](#)

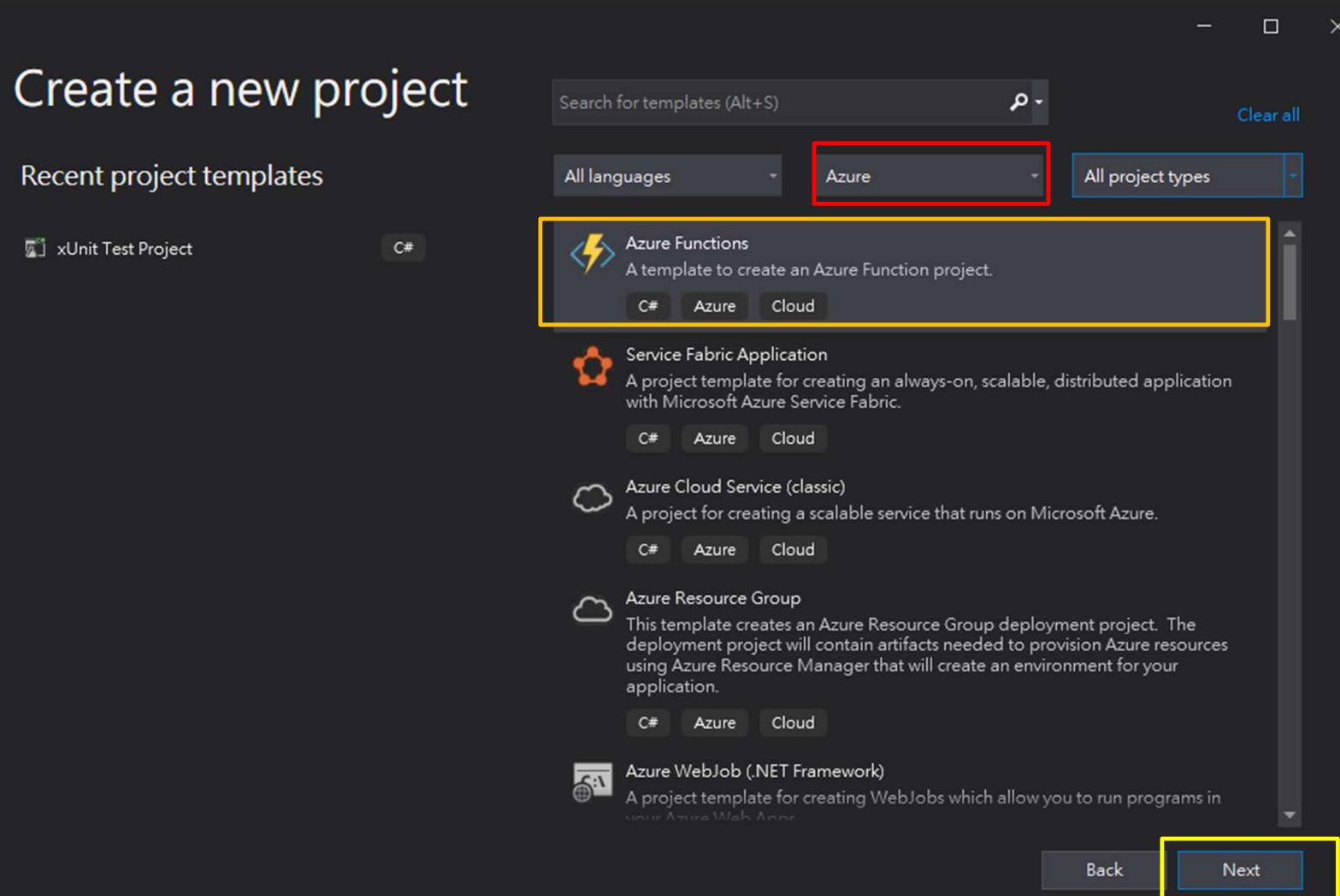
需要協助嗎? 查看 [Microsoft 開發人員社群](#) 或是  
透過 [Visual Studio 支援](#) 與我們連絡。

安裝程式版本 2.11.52.58712

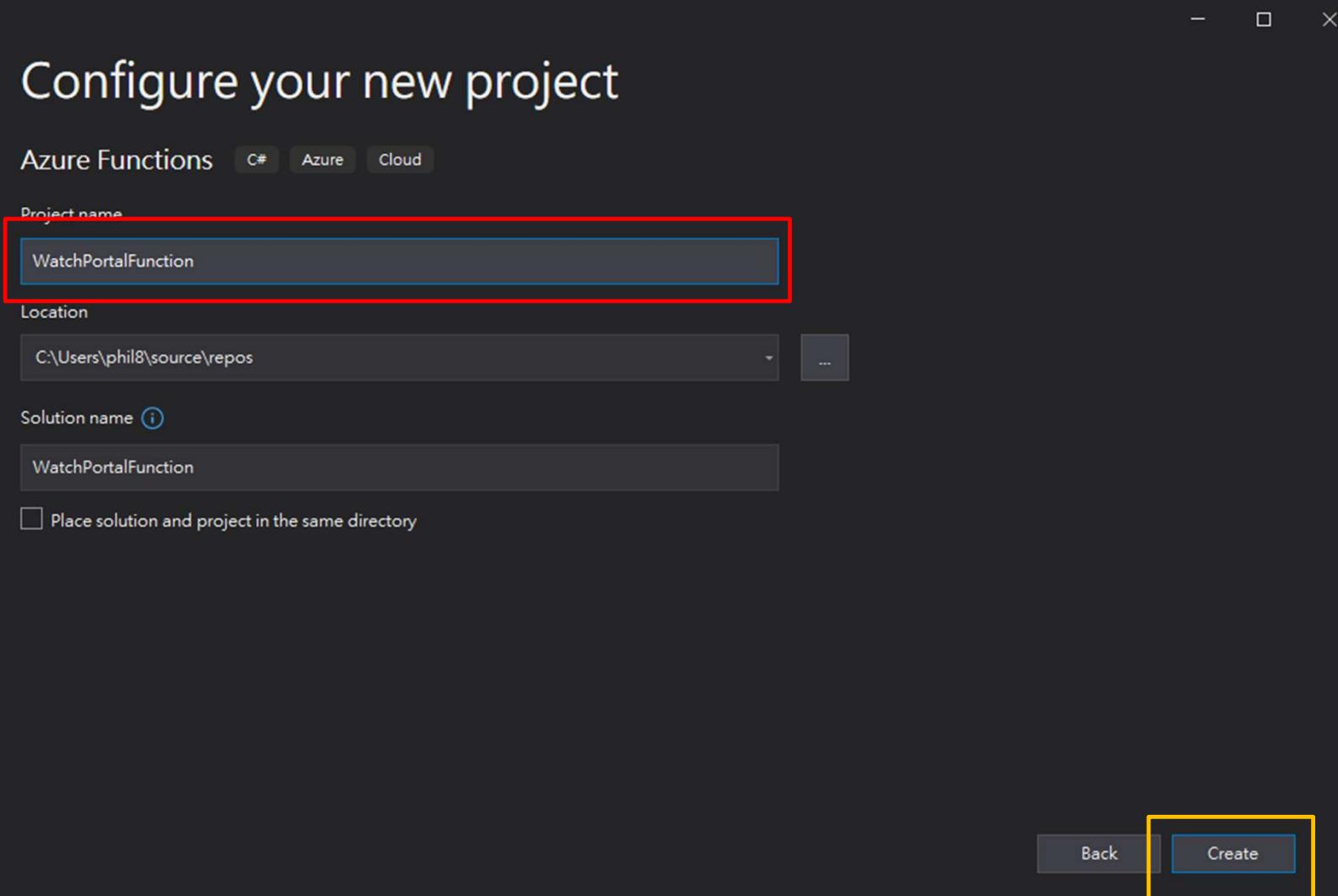
# Create an Azure Function App - 2



# Create an Azure Function App - 3



# Create an Azure Function App - 4



# Create an Azure Function App - 5

Create a new Azure Functions application

.NET Core 3 (LTS)

Empty

Creates an Azure Function project with no triggers. Function triggers can be added during development.

Service Bus Queue trigger

A C# function that will be run whenever a message is added to a specified Service Bus queue

Http trigger

A C# function that will be run whenever it receives an HTTP request

Http trigger with OpenAPI

A C# function that will be run whenever it receives an HTTP request and is preconfigured to generate and render OpenAPI document

Timer trigger

A C# function that will be run on a specified schedule

Queue trigger

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

Storage account (AzureWebJobsStorage)

Storage emulator

Some capabilities may require an Azure storage account.

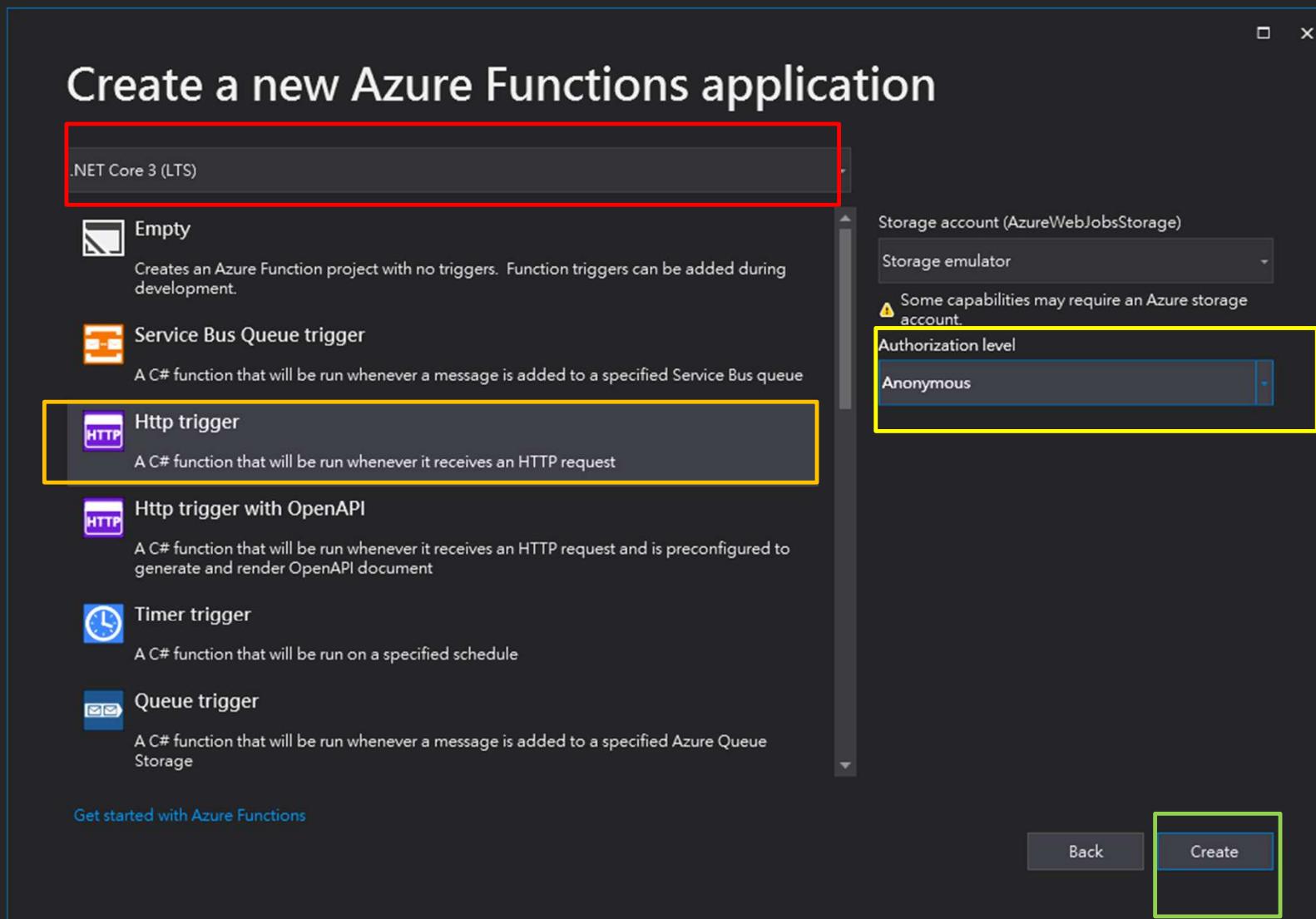
Authorization level

Anonymous

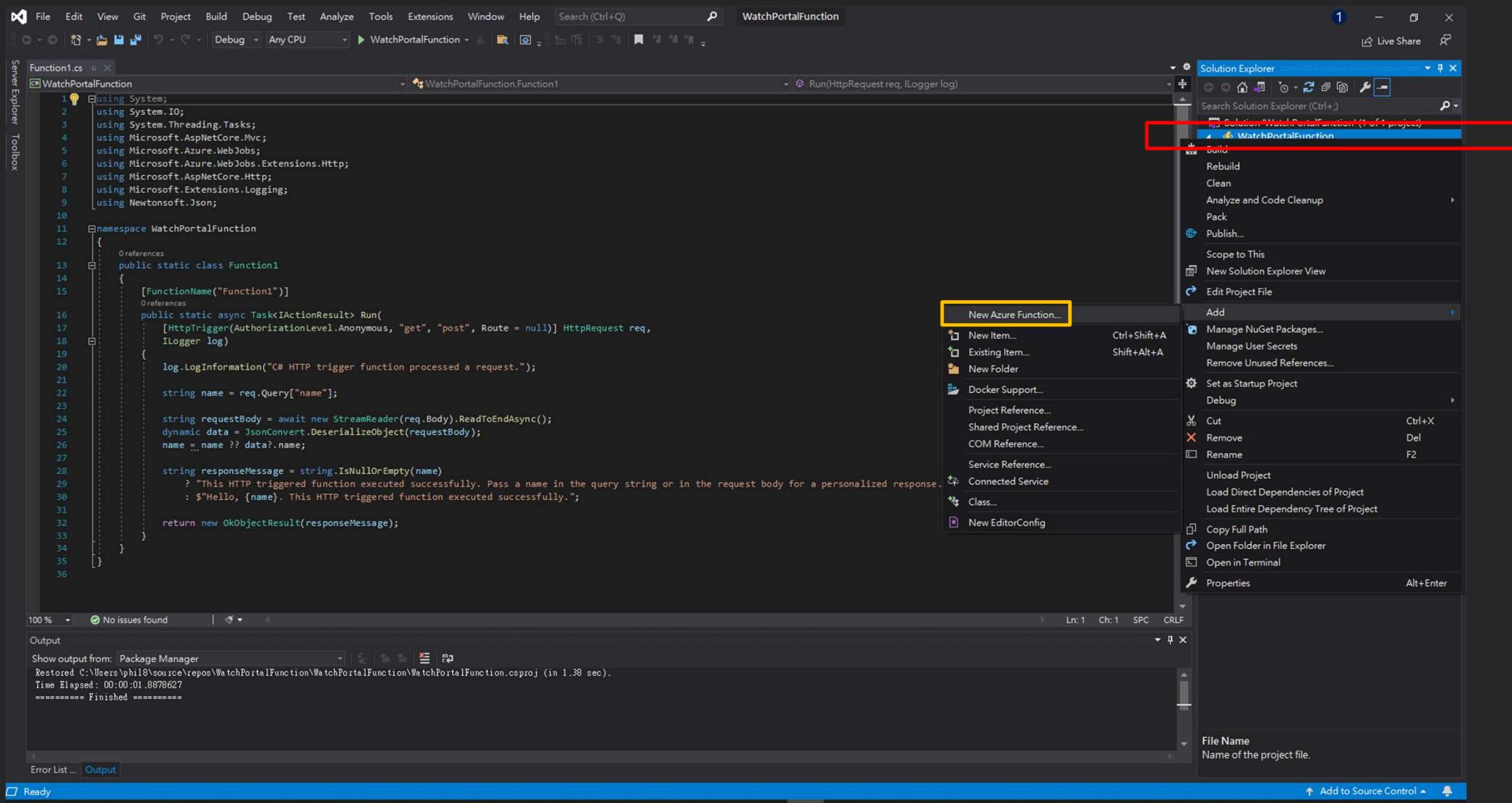
Get started with Azure Functions

Back

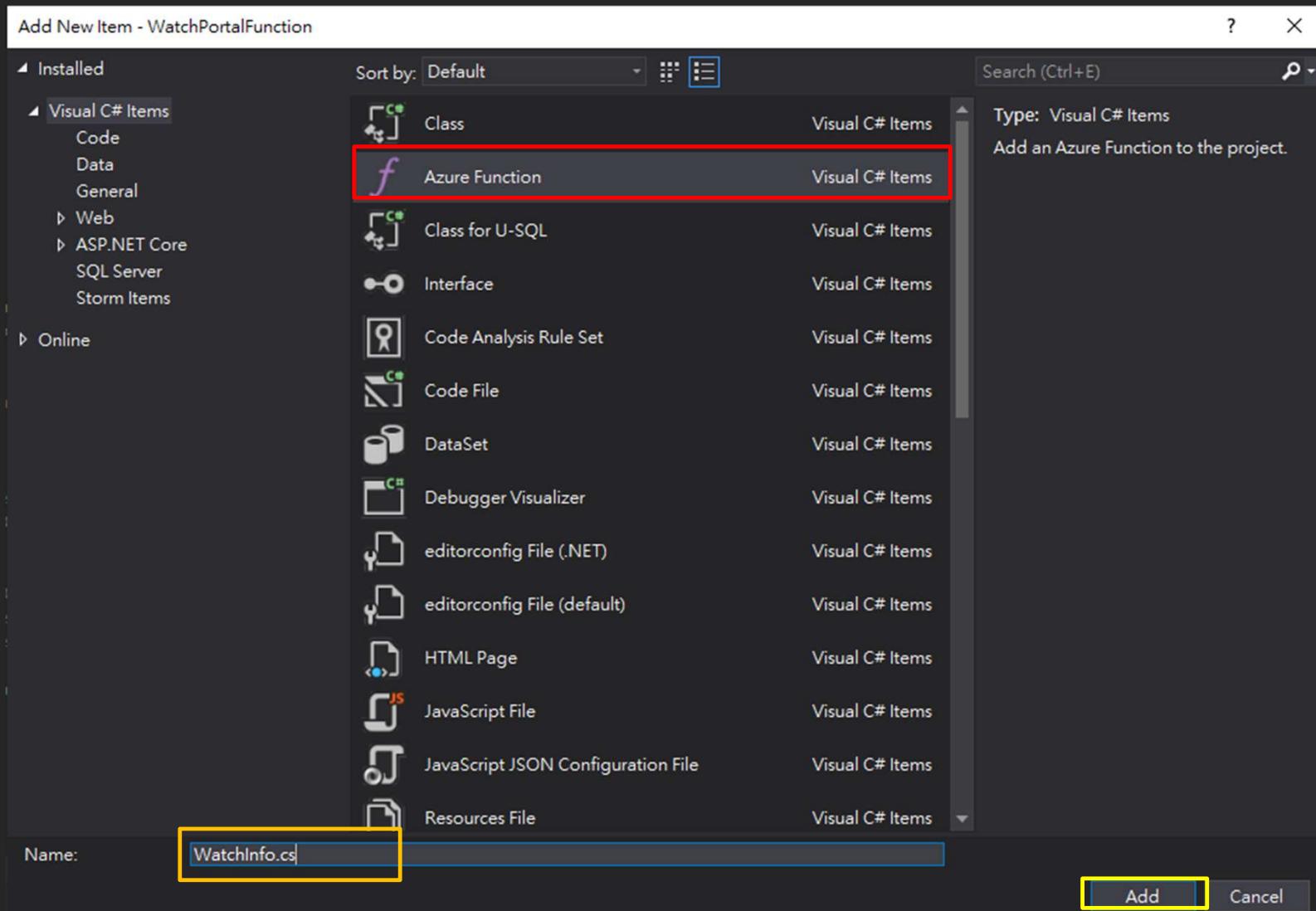
Create



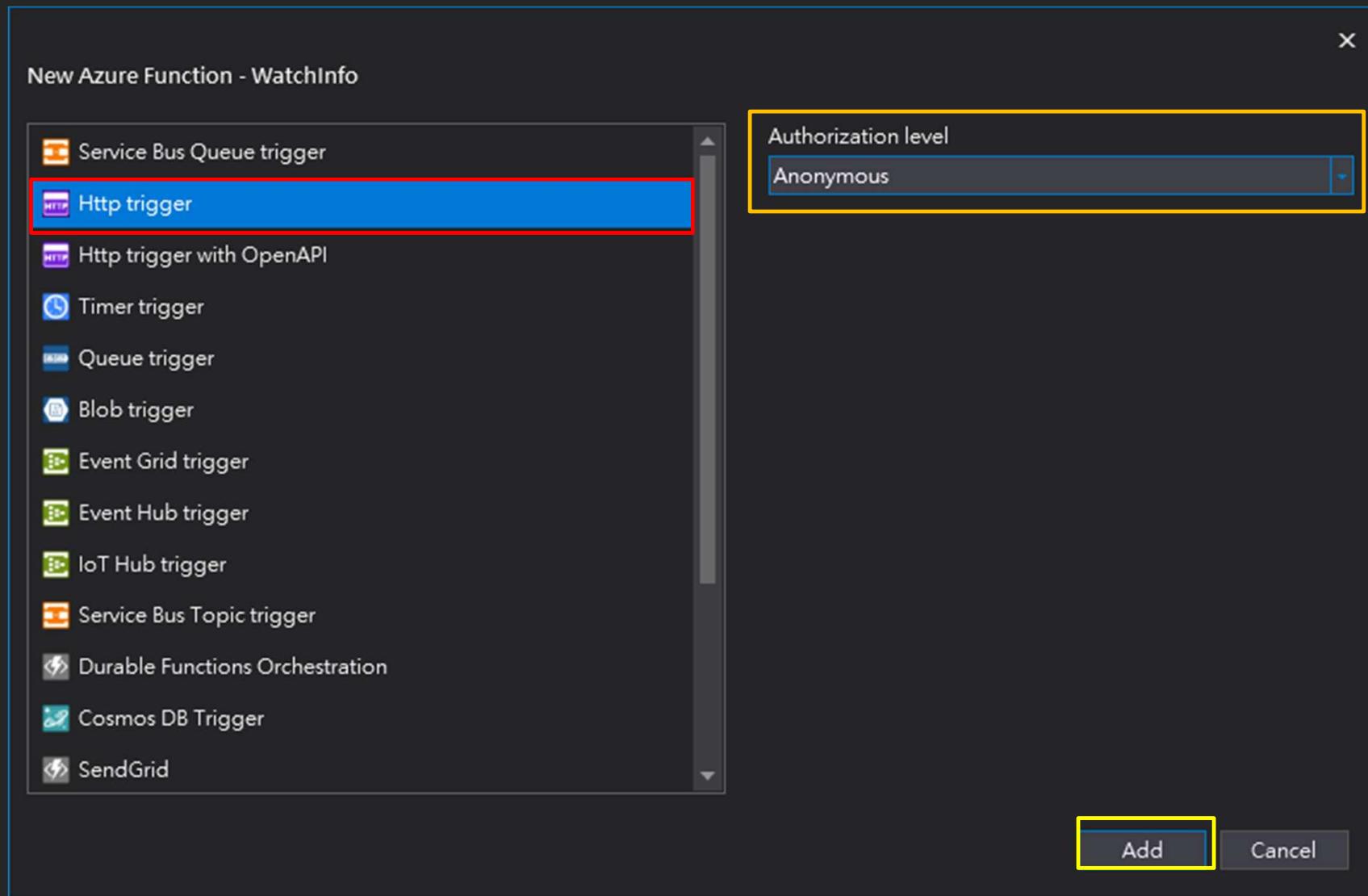
# Create the WatchInfo Azure Function - 1



## Create the WatchInfo Azure Function - 2



# Create the WatchInfo Azure Function - 3

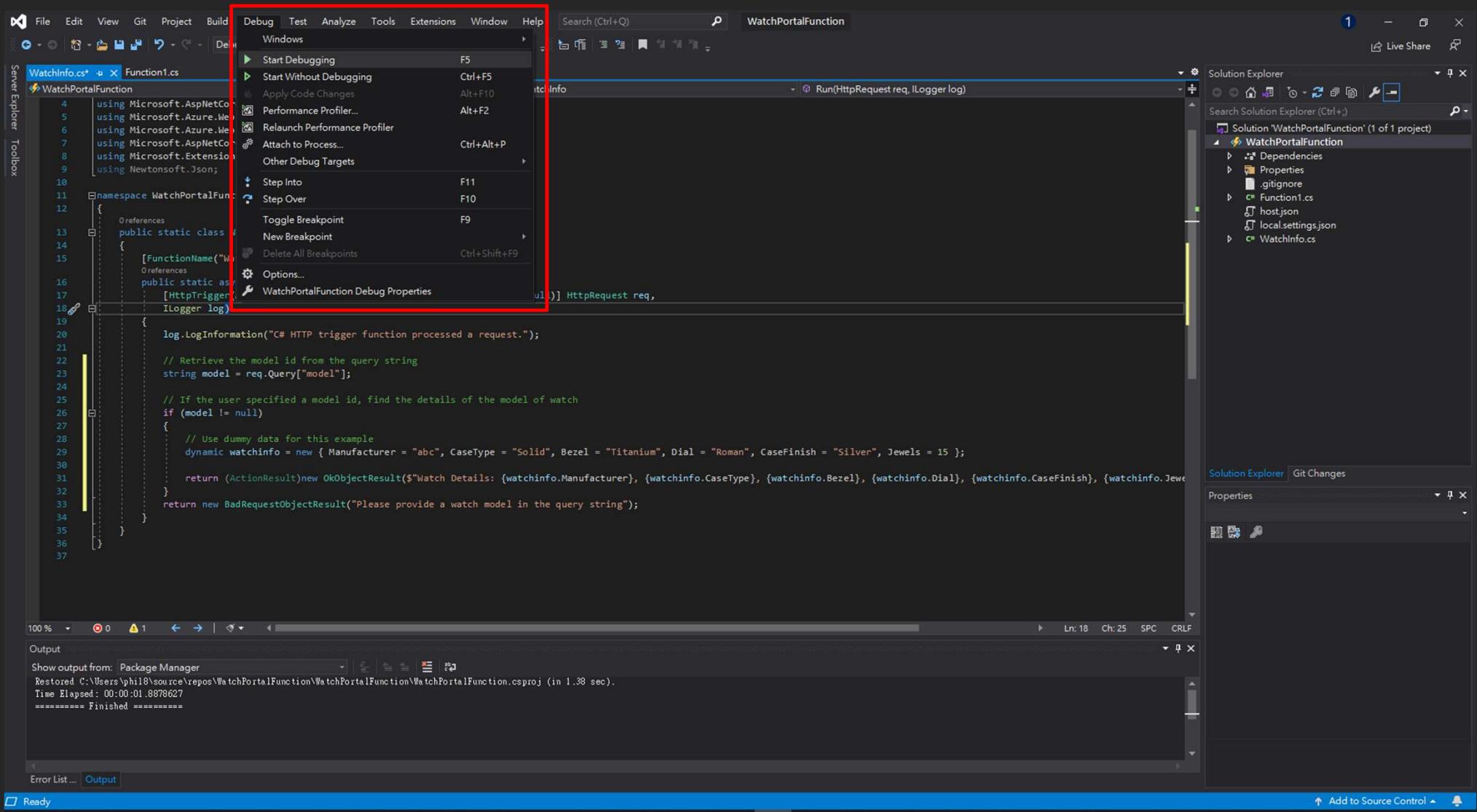


# Create the WatchInfo Azure Function - 4

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Toolbox:** Server Explorer, Toolbox.
- Code Editor:** WatchInfo.cs (Function1.cs) is open, showing C# code for an Azure Function. A yellow box highlights the entire code area. The code defines a static class WatchInfo with a Run method that handles HTTP triggers for "get" and "post" methods. It retrieves a model ID from the query string, uses dummy data for watch details, and returns an OKObjectResult or BadRequestObjectResult.
- Solution Explorer:** Shows the Solution 'WatchPortalFunction' (1 of 1 project) containing the WatchPortalFunction folder with files: Dependencies, Properties, .gitignore, host.json, local.settings.json, and WatchInfo.cs (which is selected and highlighted with a red box).
- Output Window:** Shows the output of a package restore command, indicating success with a time elapsed of 00:00:01.8878627 and a finished message.
- Status Bar:** Ready.

# Test the Azure Function locally - 1



# Test the Azure Function locally - 2

The screenshot displays a Windows desktop environment with a browser and a terminal window.

**Browser:** The address bar shows `localhost:7071/api/WatchInfo?model=abc`. A red box highlights this URL. The page content is "Watch Details: abc, Solid, Titanium, Roman, Silver, 15".

**Terminal Window:** The title bar says "C:\Users\phil8\AppData\Local\AzureFunctionsTools\Releases\3.30.1\cli\_x64\func.exe". The output shows:

```
Azure Functions Core Tools
Core Tools Version: 3.0.3904 Commit hash: c345f7140a8f968c5dbc621f8a8374d8e3234206 (64-bit)
Function Runtime Version: 3.3.1.0

[2021-11-25T12:55:42.740Z] Found C:\Users\phil8\source\repos\WatchPortalFunction\WatchPortalFunction\WatchPortalFunction.csproj. Using user secrets file configuration.

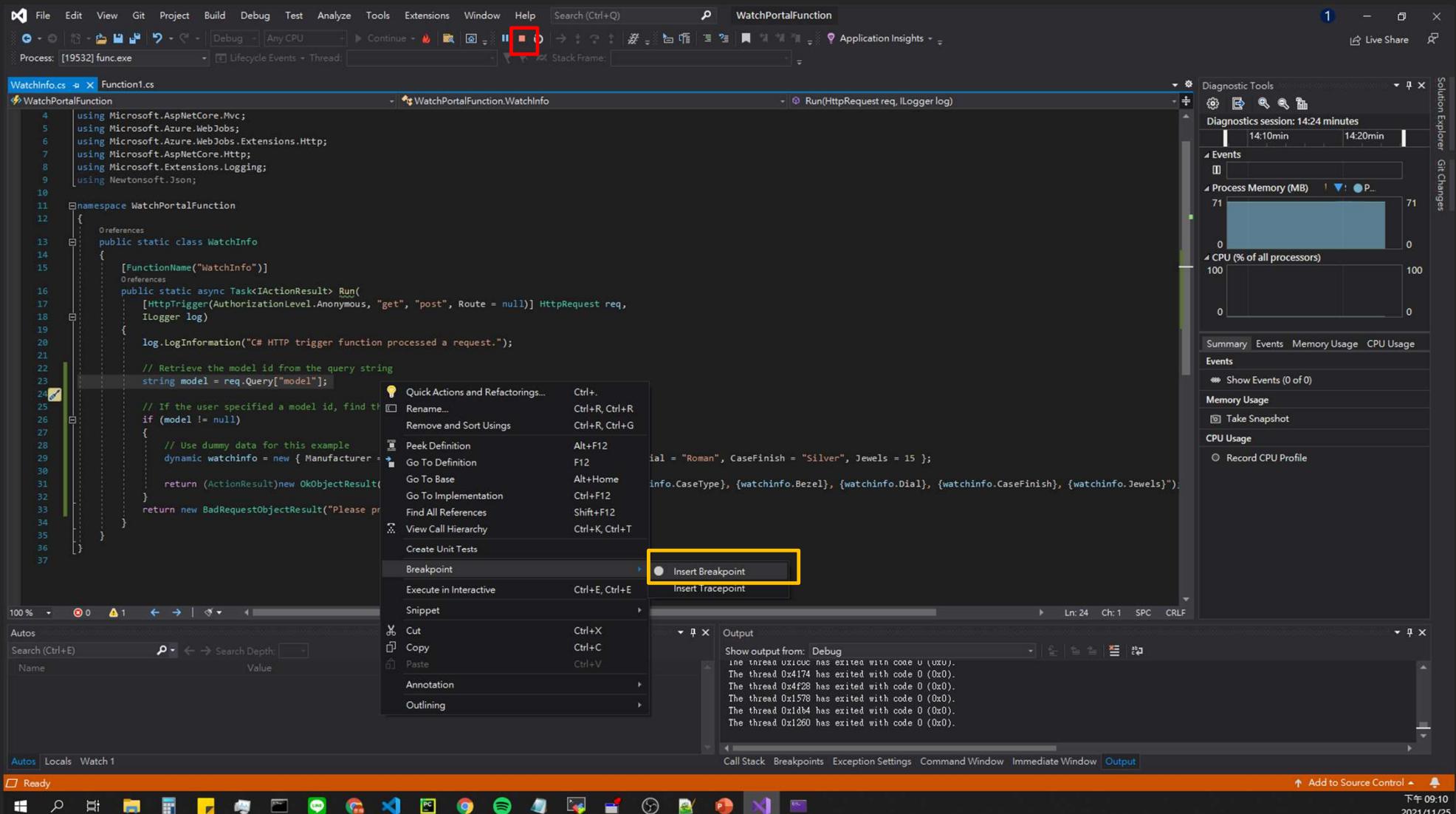
Functions:

    Function1: [GET,POST] http://localhost:7071/api/Function1
    WatchInfo: [GET,POST] http://localhost:7071/api/WatchInfo

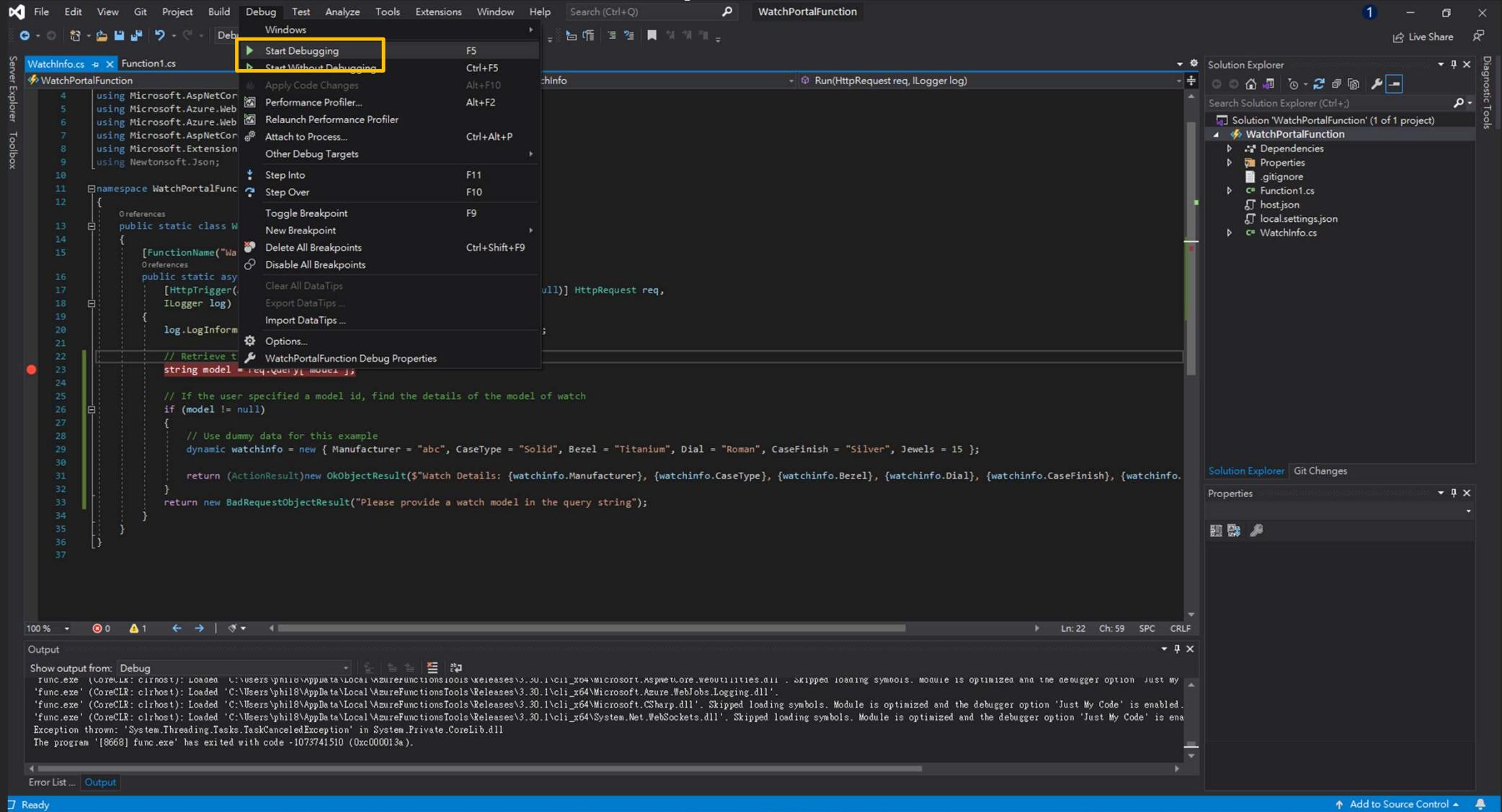
For detailed output, run func with --verbose flag.
[2021-11-25T12:55:49.818Z] Host lock lease acquired by instance ID '000000000000000000000000000000002784BC3D'.
[2021-11-25T12:56:10.798Z] Executing 'WatchInfo' (Reason='This function was programmatically called via the host APIs.', Id=0806939c-d0d0-41e7-bfa5-8af2e1020715)
[2021-11-25T12:56:10.813Z] C# HTTP trigger function processed a request.
[2021-11-25T12:56:10.860Z] Executed 'WatchInfo' (Succeeded, Id=0806939c-d0d0-41e7-bfa5-8af2e1020715, Duration=72ms)
```

**Taskbar:** The taskbar shows various pinned icons, including File Explorer, Microsoft Edge, and Visual Studio Code. The system tray indicates the date as "2021/11/25" and the time as "下午 08:56".

# Test the Azure Function locally - 3



# Test the Azure Function locally - 4



# Test the Azure Function locally - 5

The screenshot shows the Microsoft Visual Studio interface during local testing of an Azure Function. The code editor displays the `WatchInfo.cs` file, which contains the function definition:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.Logging;
using Newtonsoft.Json;

namespace WatchPortalFunction
{
    public static class WatchInfo
    {
        [FunctionName("WatchInfo")]
        public static async Task<IActionResult> Run(
            [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
            ILogger log)
        {
            log.LogInformation("C# HTTP trigger function processed a request.");

            // Retrieve the model id from the query string
            string model = req.Query["model"];

            // If the user specified a model id, find the details of the model of watch
            if (model != null)
            {
                // Use dummy data for this example
                dynamic watchinfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };

                return (ActionResult)new OkObjectResult($"Watch Details: {watchinfo.Manufacturer}, {watchinfo.CaseType}, {watchinfo.Bezel}, {watchinfo.Dial}, {watchinfo.CaseFinish}, {watchinfo.Jewels}");
            }
            return new BadRequestObjectResult("Please provide a watch model in the query string");
        }
    }
}
```

A red callout box highlights the URL and F5 instructions:

在Browser  
http://localhost:7071/api/WatchInfo?model=abc  
F5重新整理一次

The Diagnostic Tools window shows a session duration of 12 seconds. The CPU usage chart indicates a peak of 100% for 10 seconds.

The Output window shows the command-line output of the function execution.

The Autos window shows the current variable values:

Name	Type	Value
log	Microsoft.Extensions.Logging.Logger	{Microsoft.Extensions.Logging.Logger}
model	string	null

# Test the Azure Function locally - 6

The screenshot shows the Microsoft Visual Studio IDE interface during the local testing of an Azure Function. The main window displays the code for `WatchInfo.cs`, which contains the definition for the `WatchInfo` class and its `Run` method. The code uses `Microsoft.AspNetCore.Mvc`, `Microsoft.Azure.WebJobs`, and `Microsoft.Azure.WebJobs.Extensions.Http` namespaces. The `Run` method is triggered by an `[HttpTrigger]` attribute with `AuthorizationLevel.Anonymous` and routes `"get"` and `"post"`. It logs a message and retrieves a model ID from the query string. If a model ID is provided, it creates a `watchInfo` object with dummy data and returns it as a JSON response. Otherwise, it returns a `BadRequestObjectResult` with an error message.

The Diagnostic Tools window on the right shows a summary of the diagnostics session, including events, process memory usage, and CPU usage over 12 seconds. The CPU usage chart indicates minimal activity, staying near 0%.

The Autos window at the bottom left shows the current values of variables: `model` is set to `"abc"`. The Output window at the bottom right shows the command-line output of the function execution, including the loading of DLLs and the exit code `0 (0x0)`.

```
4  using Microsoft.AspNetCore.Mvc;
5  using Microsoft.Azure.WebJobs;
6  using Microsoft.Azure.WebJobs.Extensions.Http;
7  using Microsoft.AspNetCore.Http;
8  using Microsoft.Extensions.Logging;
9  using Newtonsoft.Json;
10
11 namespace WatchPortalFunction
12 {
13     public static class WatchInfo
14     {
15         [FunctionName("WatchInfo")]
16         public static async Task<IActionResult> Run(
17             [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
18             ILogger log)
19         {
20             log.LogInformation("C# HTTP trigger function processed a request.");
21
22             // Retrieve the model id from the query string
23             string model = req.Query["model"];
24
25             // If the user specified a model id, find the details of the model of watch
26             if (model != null) {sim elapsed}
27             {
28                 // Use dummy data for this example
29                 dynamic watchInfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };
30
31                 return (ActionResult)new OkObjectResult($"Watch Details: {watchInfo.Manufacturer}, {watchInfo.CaseType}, {watchInfo.Bezel}, {watchInfo.Dial}, {watchInfo.CaseFinish}, {watchInfo.Jewels}");
32             }
33             return new BadRequestObjectResult("Please provide a watch model in the query string");
34         }
35     }
36 }
37
```

# Test the Azure Function locally - 7

The screenshot shows the Visual Studio IDE interface during the development of an Azure Function. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a Search (Ctrl+Q) field. The title bar displays "WatchPortalFunction". The main editor window shows the C# code for the "WatchInfo.cs" file, which contains an Azure Function named "WatchInfo" with a single HTTP trigger. The code retrieves a model ID from the query string and returns a dummy watch object if provided, or a BadRequestObjectResult otherwise. The code editor has syntax highlighting and various status indicators (e.g., breakpoints, code coverage). To the right of the editor is the Diagnostic Tools window, which displays performance metrics for the current session: Events (1.69s), Process Memory (MB) (62), and CPU (% of all processors) (0). Below the editor are the Autos, Locals, and Watch 1 windows, showing the variable "model" with a value of "abc". The Output window at the bottom shows the command-line output of the function's execution.

```
4  using Microsoft.AspNetCore.Mvc;
5  using Microsoft.Azure.WebJobs;
6  using Microsoft.Azure.WebJobs.Extensions.Http;
7  using Microsoft.AspNetCore.Http;
8  using Microsoft.Extensions.Logging;
9  using Newtonsoft.Json;
10
11 namespace WatchPortalFunction
12 {
13     public static class WatchInfo
14     {
15         [FunctionName("WatchInfo")]
16         public static async Task<IActionResult> Run(
17             [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
18             ILogger log)
19         {
20             log.LogInformation("C# HTTP trigger function processed a request.");
21
22             // Retrieve the model id from the query string
23             string model = req.Query["model"];
24
25             // If the user specified a model id, find the details of the model of watch
26             if (model != null)
27             {
28                 // Use dummy data for this example
29                 dynamic watchinfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };
30
31                 return (ActionResult)new OkObjectResult($"Watch Details: {watchinfo.Manufacturer}, {watchinfo.CaseType}, {watchinfo.Bezel}, {watchinfo.Dial}, {watchinfo.CaseFinish}, {watchinfo.Jewels}");
32             }
33             else
34             {
35                 return new BadRequestObjectResult("Please provide a watch model in the query string");
36             }
37         }
38     }
39 }
```

Output

```
func.exe (CoreCLR: clrhost): Loaded 'C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\Common7\IDE\Remote Debugger\x64\Runtime.dll'. Skipped loading symbols. Module was built without debugging information.
func.exe (CoreCLR: clrhost): Loaded 'C:\Users\phi118\AppData\Local\AzureFunctionsTools\Releases\3.30.1\cli_x64\System.ComponentModel.Annotations.dll'.
func.exe (CoreCLR: clrhost): Loaded 'C:\Users\phi118\AppData\Local\AzureFunctionsTools\Releases\3.30.1\cli_x64\Microsoft.AspNetCore.WebUtilities.dll'.
func.exe (CoreCLR: clrhost): Loaded 'C:\Users\phi118\AppData\Local\AzureFunctionsTools\Releases\3.30.1\cli_x64\Microsoft.Azure.WebJobs.Logging.dll'.
func.exe (CoreCLR: clrhost): Loaded 'C:\Users\phi118\AppData\Local\AzureFunctionsTools\Releases\3.30.1\cli_x64\Microsoft.CSharp.dll'.
```

# Test the Azure Function locally - 8

The screenshot shows the Visual Studio IDE interface during local testing of an Azure Function. The code editor displays `WatchInfo.cs` with the following content:

```
1 using Microsoft.AspNetCore.Mvc;
2 using Microsoft.Azure.WebJobs;
3 using Microsoft.Azure.WebJobs.Extensions.Http;
4 using Microsoft.AspNetCore.Http;
5 using Microsoft.Extensions.Logging;
6 using Newtonsoft.Json;
7
8 namespace WatchPortalFunction
9 {
10     public static class WatchInfo
11     {
12         [FunctionName("WatchInfo")]
13         public static async Task<ActionResult> Run(
14             [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
15             ILogger log)
16         {
17             log.LogInformation("C# HTTP trigger function processed a request.");
18
19             // Retrieve the model id from the query string
20             string model = req.Query["model"];
21
22             // If the user specified a model id, find the details of the model of watch
23             if (model != null)
24             {
25                 // Use dummy data for this example
26                 dynamic watchinfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };
27
28                 return (ActionResult)new OkObjectResult($"Watch Details: {watchinfo.Manufacturer}, {watchinfo.CaseType}, {watchinfo.Bezel}, {watchinfo.Dial}, {watchinfo.CaseFinish}, {watchinfo.Jewels}");
29             }
30             else
31             {
32                 return new BadRequestObjectResult("Please provide a watch model in the query string");
33             }
34         }
35     }
36 }
```

The Diagnostic Tools window on the right shows a summary of the current session, including memory usage and CPU usage over time. The Output window at the bottom shows the command-line logs for the function execution.

# Test the Azure Function locally - 9

The screenshot shows the Microsoft Visual Studio IDE interface during the development of an Azure Function. The code editor displays the `WatchInfo.cs` file, which contains the following C# code:

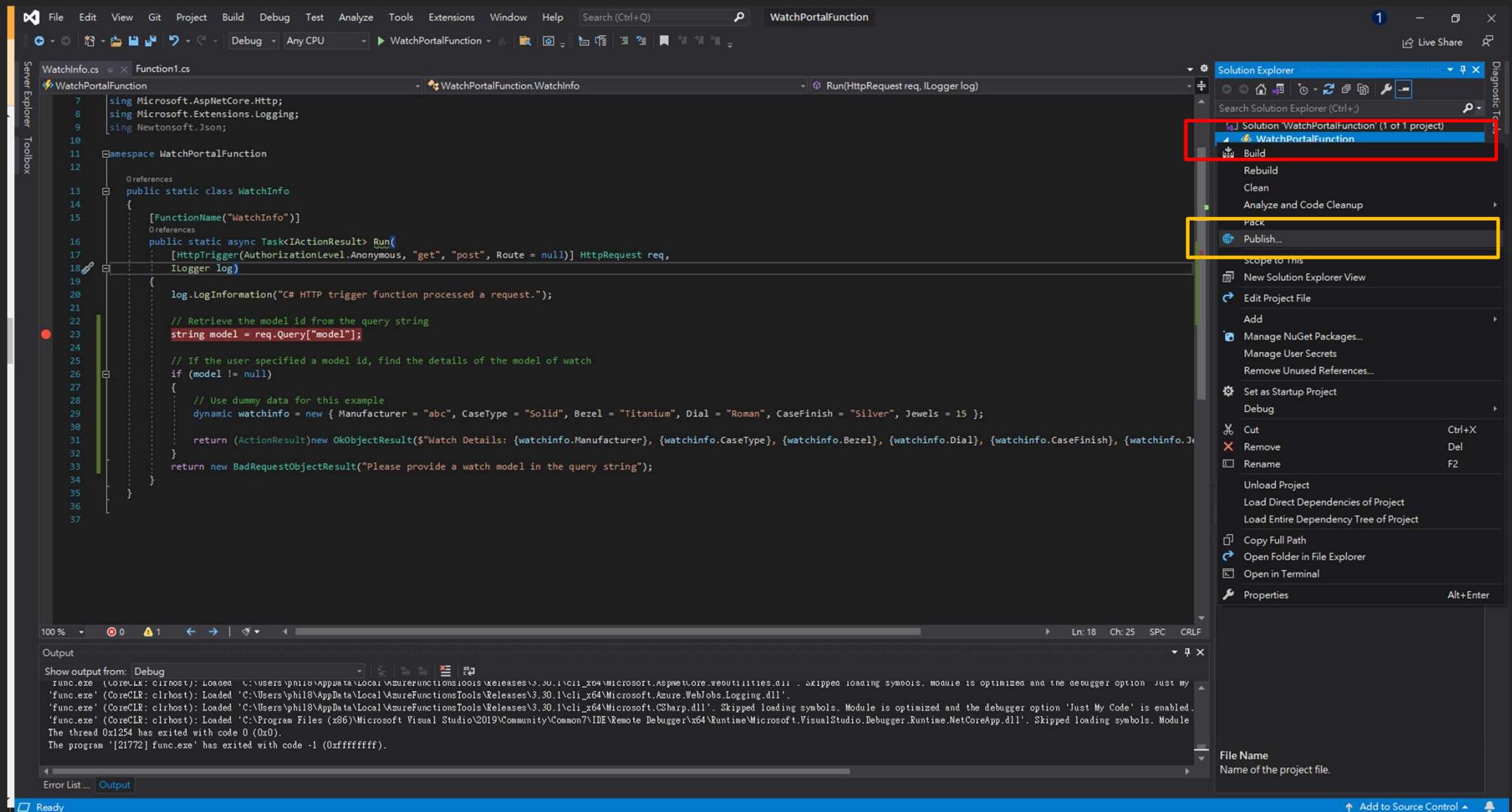
```
4  using Microsoft.AspNetCore.Mvc;
5  using Microsoft.Azure.WebJobs;
6  using Microsoft.Azure.WebJobs.Extensions.Http;
7  using Microsoft.AspNetCore.Http;
8  using Microsoft.Extensions.Logging;
9  using Newtonsoft.Json;
10 
11 namespace WatchPortalFunction
12 {
13     public static class WatchInfo
14     {
15         [FunctionName("WatchInfo")]
16         public static async Task<IActionResult> Run(
17             [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
18             ILogger log)
19         {
20             log.LogInformation("HTTP trigger function processed a request.");
21 
22             // Retrieve the model id from the query string
23             string model = req.Query["model"];
24 
25             // If the user specified a model id, find the details of the model of watch
26             if (model != null)
27             {
28                 // Use dummy data for this example
29                 dynamic watchinfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };
30 
31                 return (ActionResult)new OkObjectResult($"Watch Details: {watchinfo.Manufacturer}, {watchinfo.CaseType}, {watchinfo.Bezel}, {watchinfo.Dial}, {watchinfo.CaseFinish}, {watchinfo.Jewels}");
32             }
33             return new BadRequestObjectResult("Please provide a watch model in the query string");
34         }
35     }
36 }
37 
```

The code editor includes several annotations: a red dot on line 23 indicating a breakpoint, a yellow lightbulb icon on line 31 suggesting a quick fix or refactor, and a green vertical bar on the left margin.

The Diagnostic Tools window on the right shows performance metrics for the current session, including a timeline, process memory usage (63 MB), and CPU usage (0% of all processors).

The Output window at the bottom displays the command-line output of the Azure Functions host, showing the loading of the host application and various DLLs.

# Deploy the WatchInfo function to the Azure Function App - 1



# Deploy the WatchInfo function to the Azure Function App - 2

Where are you publishing today?

Target →

- Azure**  
Publish your application to the Microsoft cloud
- Docker Container Registry  
Publish your application to any supported Container Registry that works with Docker images
- Folder  
Publish your application to a local folder or file share
- Import Profile  
Import your publish settings to deploy your app

Back **Next** Finish Cancel

# Deploy the WatchInfo function to the Azure Function App - 3

Publish

Which Azure service would you like to use to host your application?

Target

Specific target

Azure Function App (Windows)  
Publish your application code to a serverless compute that scales dynamically and runs code on-demand

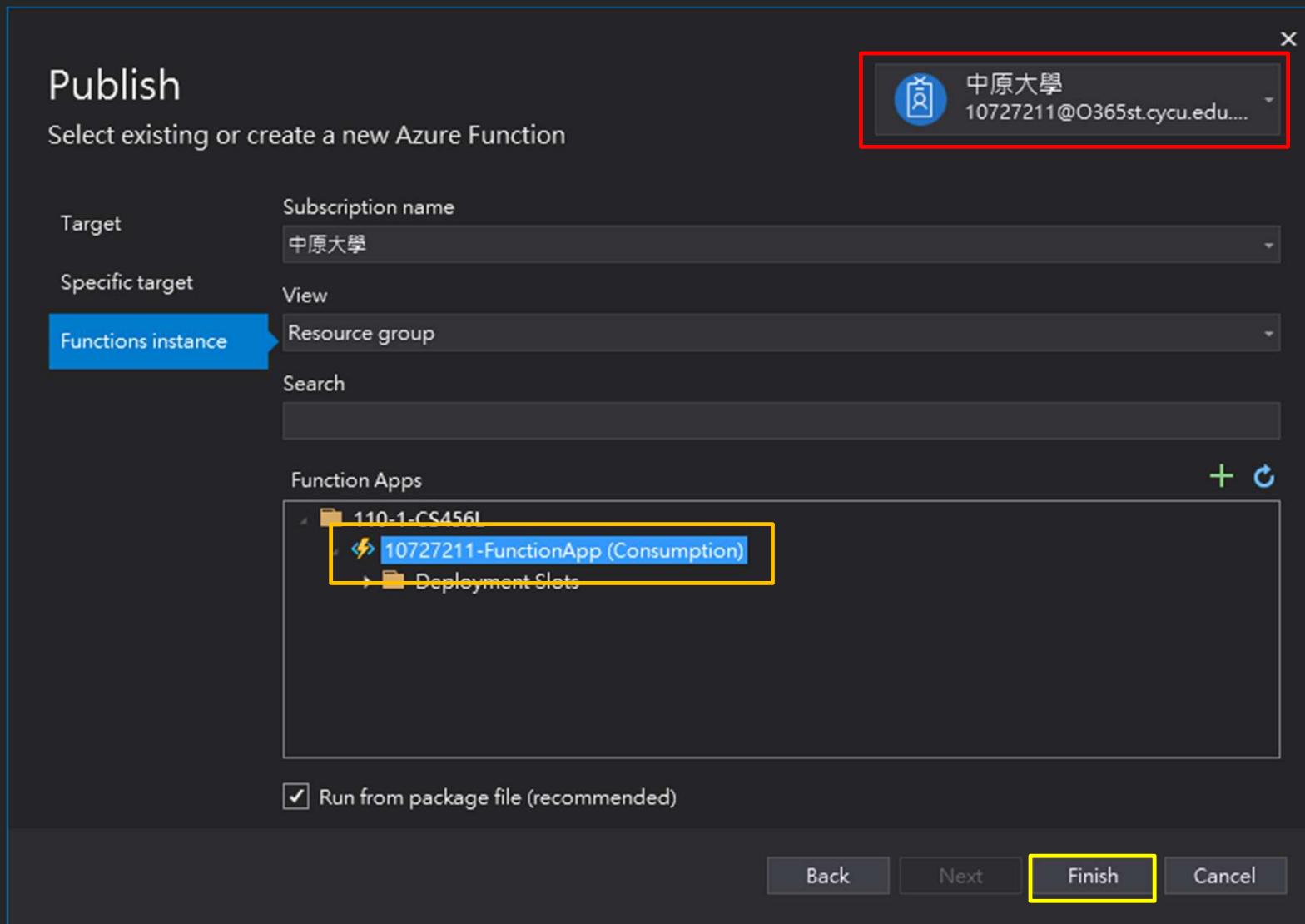
Azure Function App (Linux)  
Publish your application code to a serverless compute that scales dynamically and runs code on-demand

Azure Function App Container  
Publish your application as a Docker image to Azure Container Registry and run it on Azure Function App

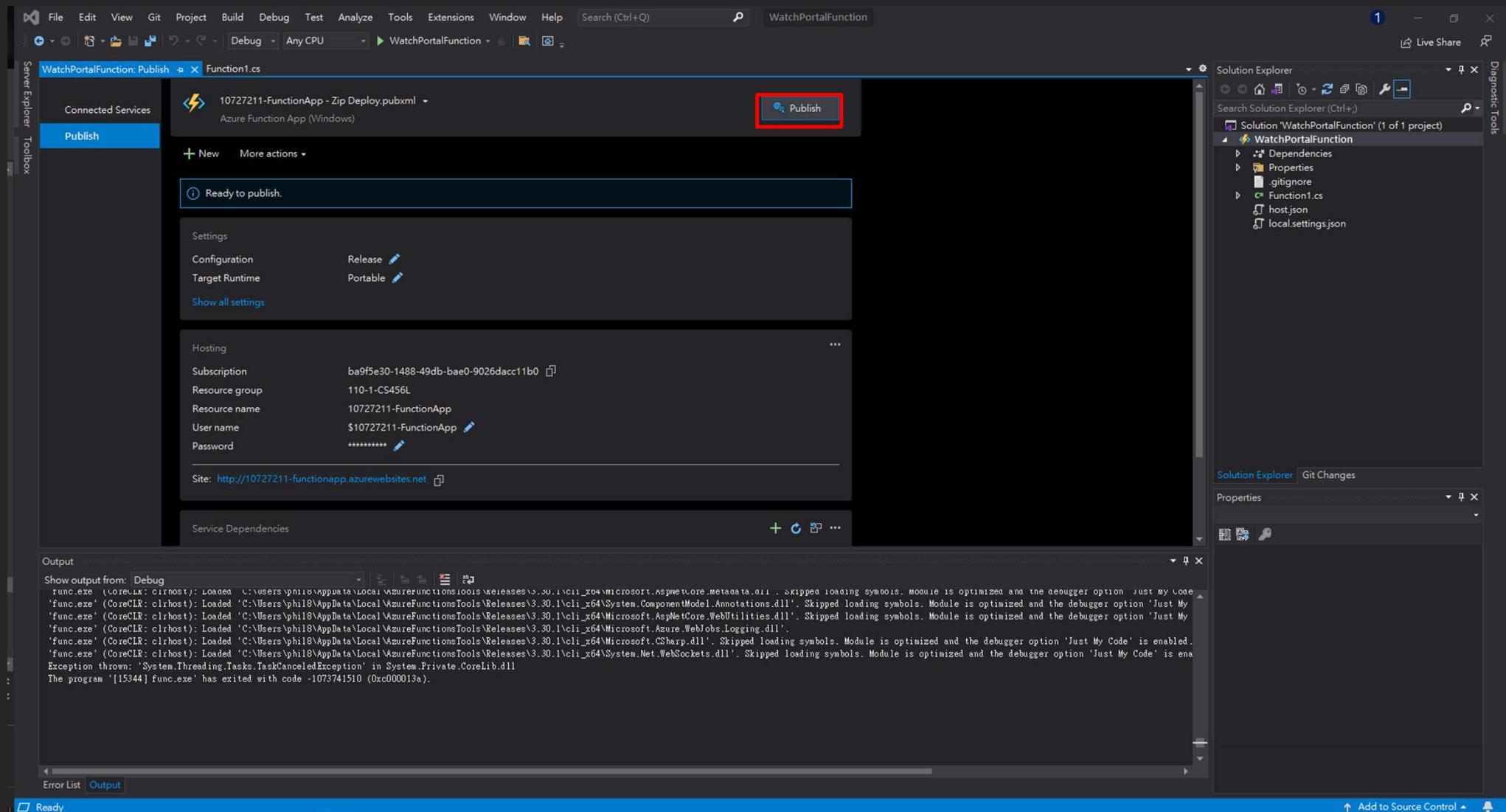
Azure Container Registry  
Publish your application as a Docker image to Azure Container Registry

Back Next Finish Cancel

# Deploy the WatchInfo function to the Azure Function App - 4



# Deploy the WatchInfo function to the Azure Function App - 5



# Deploy the WatchInfo function to the Azure Function App – 6

The screenshot shows the deployment process of an Azure Function App named "10727211-FunctionApp".

**Visual Studio (Left Side):**

- Solution Explorer:** Shows the project structure with files like WatchPortalFunction.csproj, WatchPortalFunction, Function1.cs, host.json, local.settings.json, and WatchInfo.cs.
- Server Explorer:** Publish tab shows a successful publish message: "Successfully published on 2021/11/25 at 下午 10:00." It also lists the publish profile: "10727211-FunctionApp - Zip Deploy.pubxml" (Azure Function App (Windows)).
- Output:** Displays build and publish logs. A red box highlights the output for the "Publish started" step, which shows the deployment process and successful zip deployment.

**Azure Portal (Right Side):**

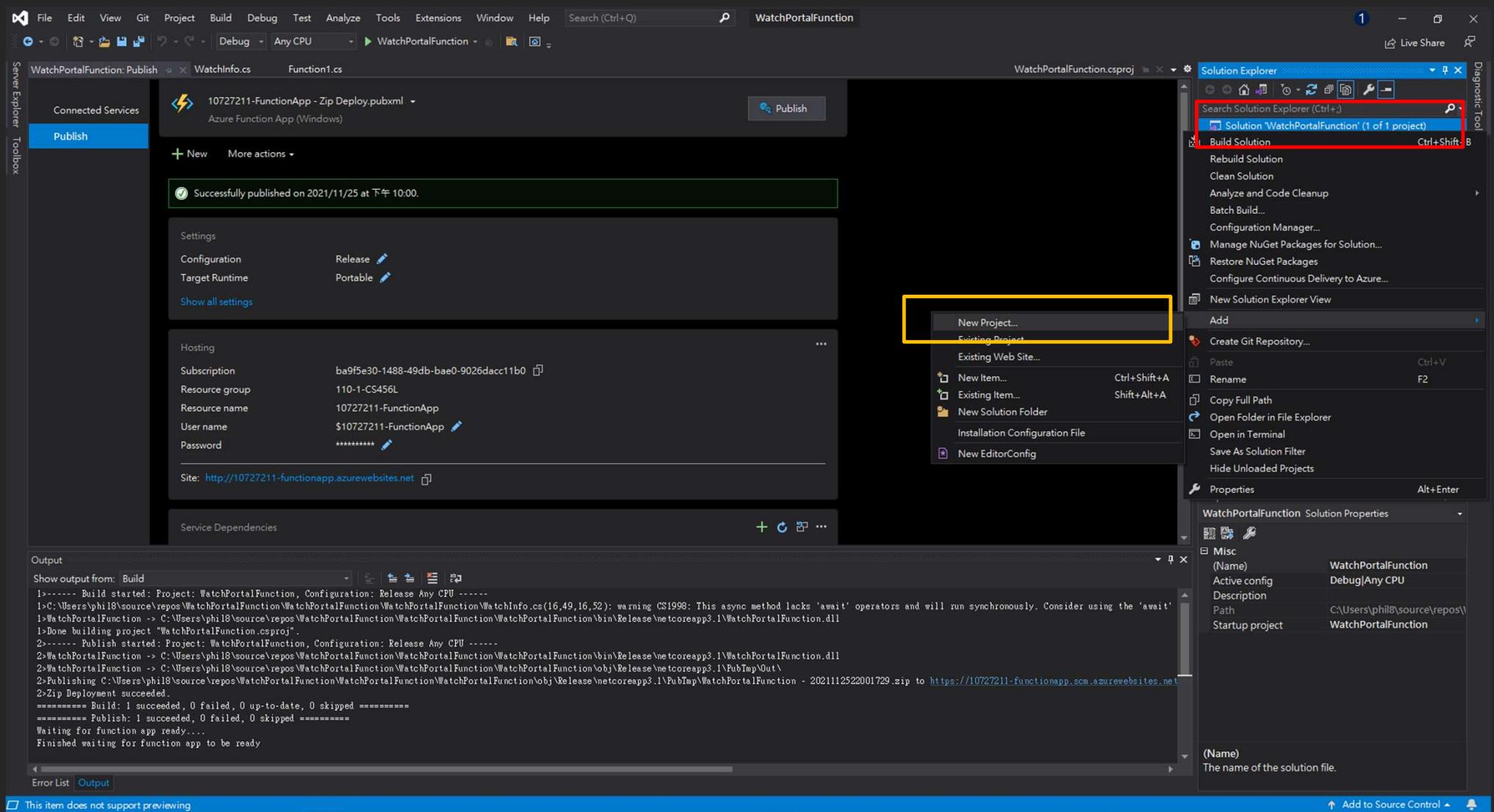
- Dashboard:** Shows the Azure Functions blade for the "10727211-FunctionApp" function app.
- Functions:** List of functions: "Function1" and "WatchInfo". The "WatchInfo" function is highlighted with a yellow border.
- Events (preview):** Shows a warning message: "Your app is currently in read only mode because you are running from a package file. To make changes, switch to 'Development' mode or run your app locally."

# Verify the functions have been deployed

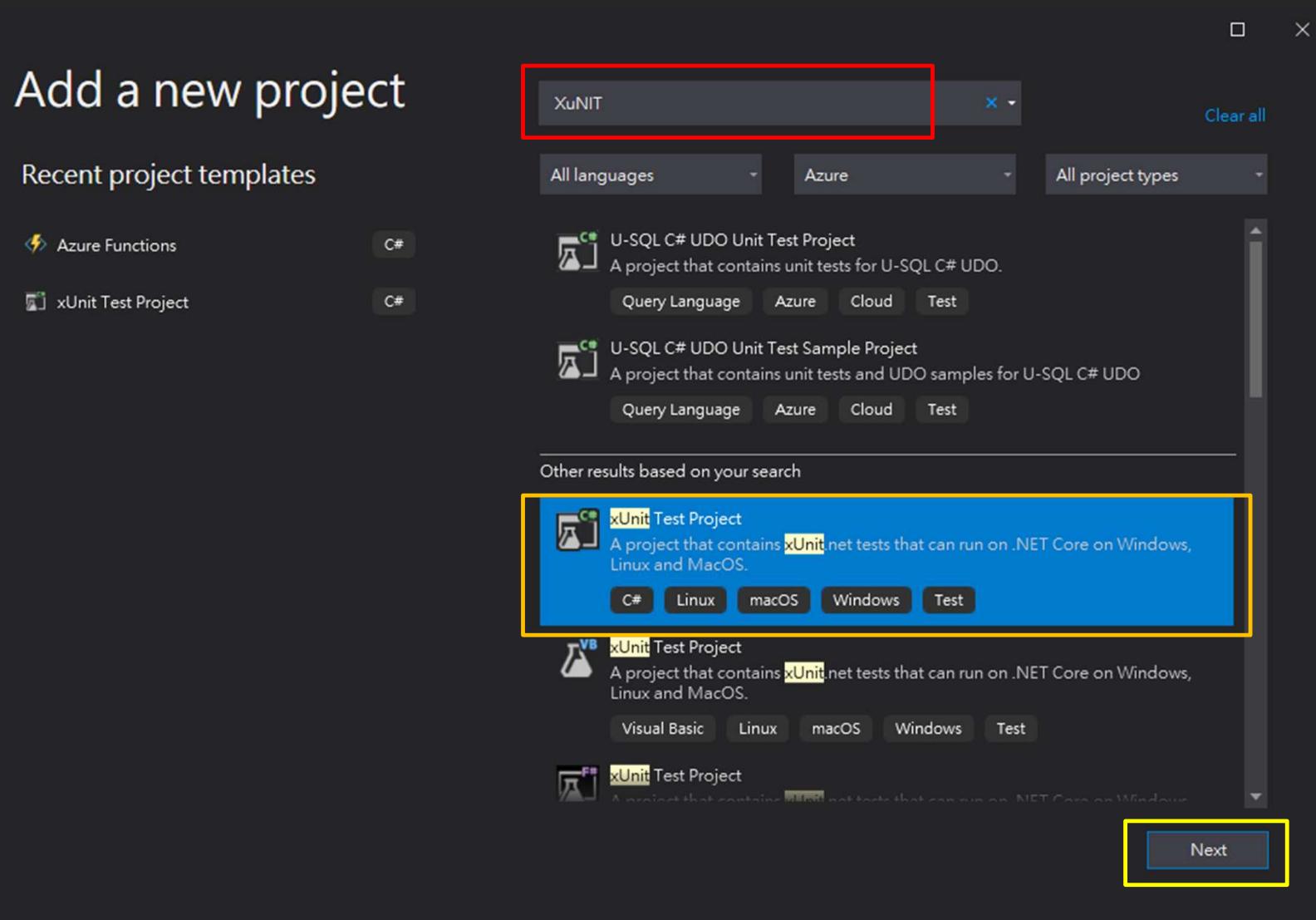
The screenshot shows the Microsoft Azure portal interface. In the top navigation bar, there are three tabs: 'Exercise - Publish a simple Azu...', '練習 - 發佈簡單的 Azure 函式 - l...', and 'WatchInfo - Microsoft Azure'. The 'WatchInfo - Microsoft Azure' tab is active. The URL in the address bar is <https://portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/functionOverview/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2Fresource...>. The main content area displays the 'Function App' blade for '10727211-FunctionApp'. A sub-blade titled 'WatchInfo' is open. The 'Get Function Url' button is highlighted with a yellow box. Below it, the function URL is shown as <https://10727211-functionapp.azurewebsites.net/api/WatchInfo?>, with a 'Copy to clipboard' button next to it, also highlighted with a yellow box. On the left sidebar, under the 'Developer' section, there are links for 'Code + Test', 'Integration', 'Monitor', and 'Function Keys'. The 'Monitor' link is highlighted with a red box.

The screenshot shows a web browser window with the URL <https://10727211-functionapp.azurewebsites.net/api/WatchInfo?model=abc>. The page content displays the response from the function: 'Watch Details: abc, Solid, Titanium, Roman, Silver, 15'. The browser's address bar and toolbar are visible at the top, and the bottom navigation bar includes links for 'Exercise - Publish a simple Azu...', '練習 - 發佈簡單的 Azure 函式 - l...', 'WatchInfo - Microsoft Azure', and other Azure services like 'Azure Portal', 'Google Drive', and 'GitHub'.

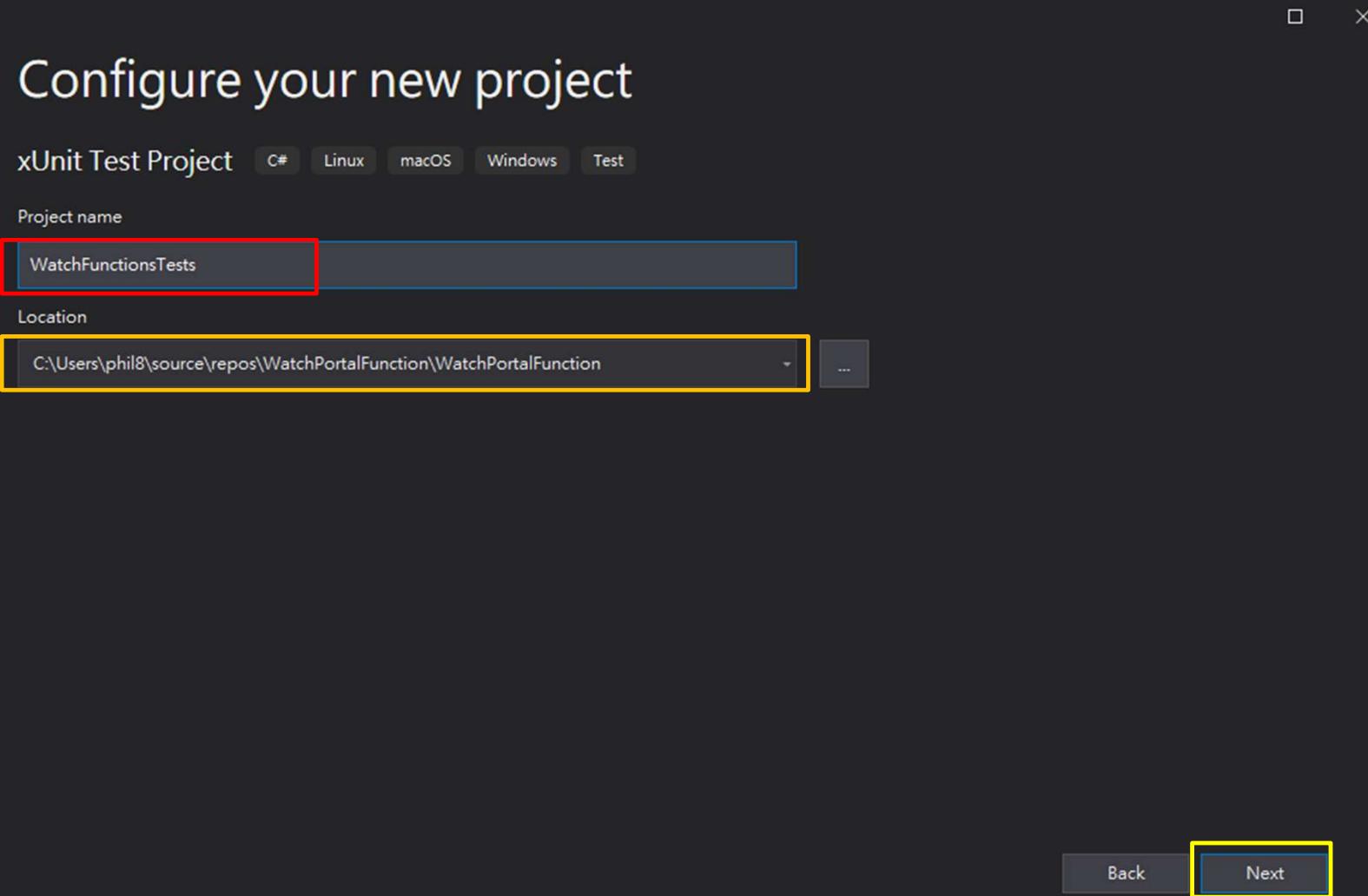
# Unit test an Azure Function - 1



# Unit test an Azure Function - 2



# Unit test an Azure Function - 3



# Unit test an Azure Function - 4

## Additional information

xUnit Test Project

C# Linux macOS Windows Test

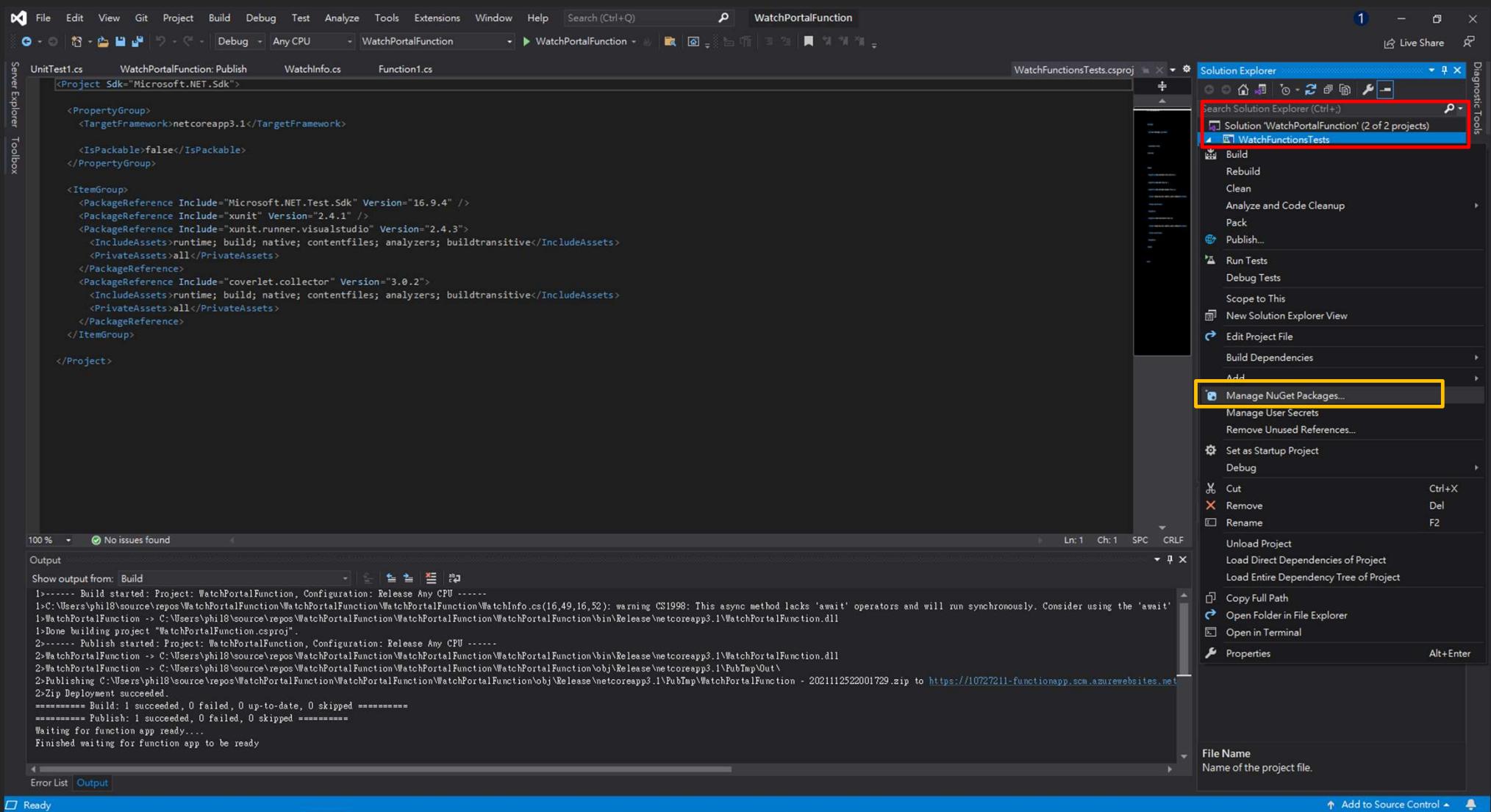
Target Framework ⓘ

.NET Core 3.1 (Long-term support)

Back

Create

# Unit test an Azure Function - 5

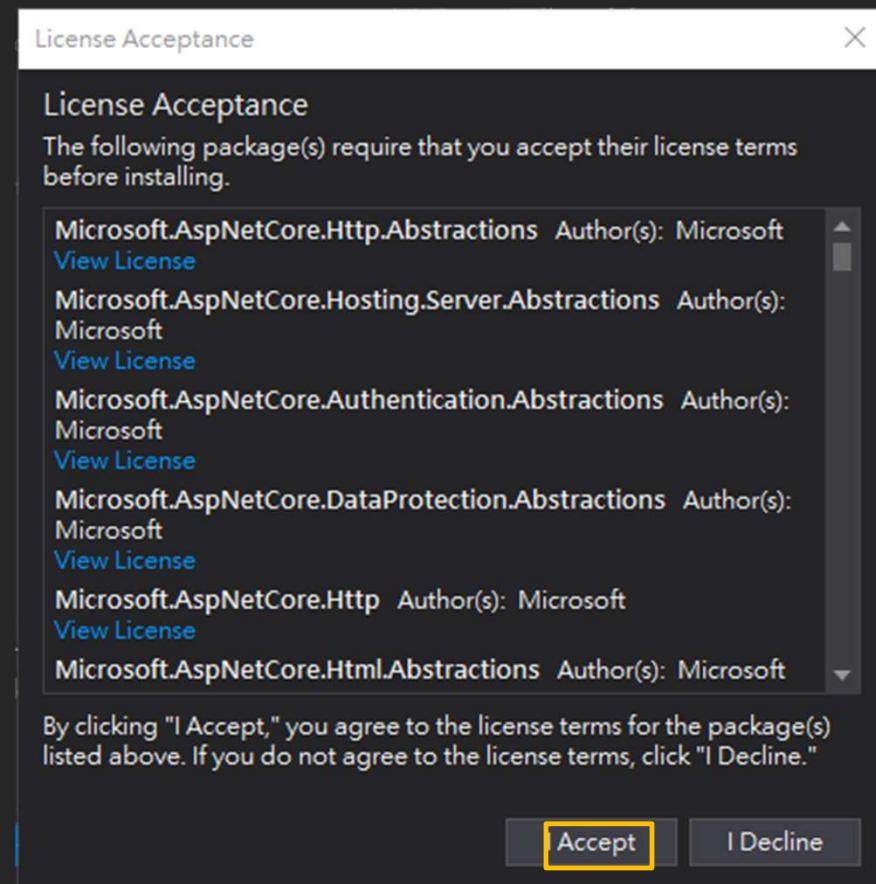
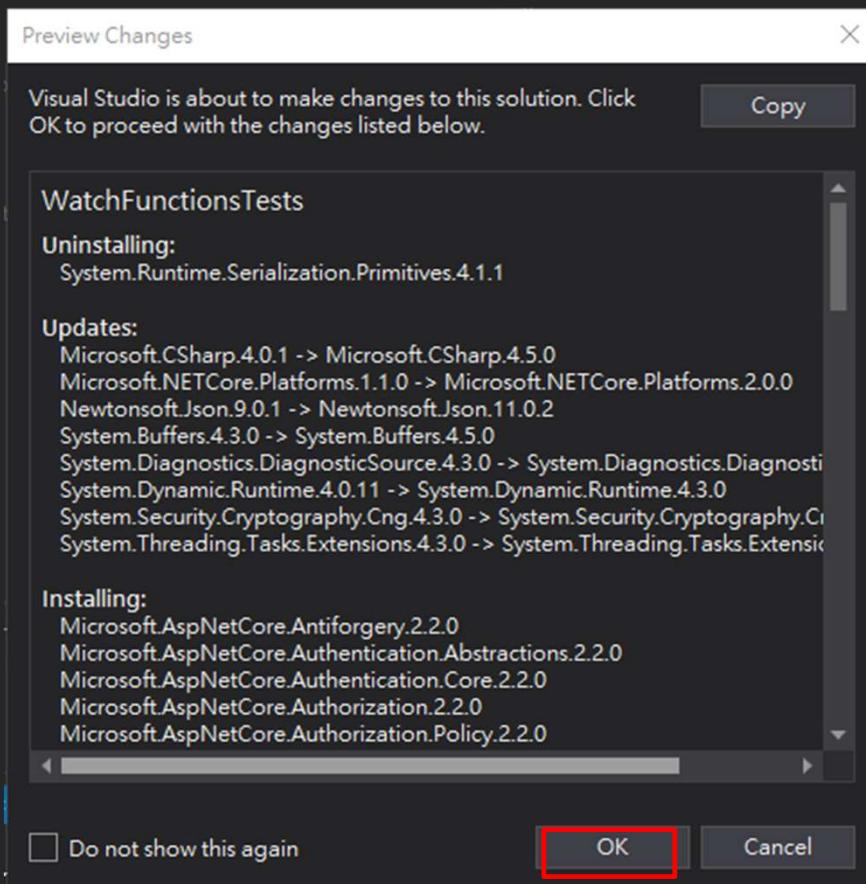


# Unit test an Azure Function - 6

The screenshot shows the Visual Studio interface with the following details:

- Toolbar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Solution Explorer:** Shows two projects: WatchFunctionsTests and WatchPortalFunction.
- NuGet Package Manager:** The current tab is "WatchFunctionsTests".
  - Left pane:** Shows the "Browse" tab selected (highlighted with a red box). It lists several NuGet packages, with **Microsoft.AspNetCore.Mvc** version 2.2.0 highlighted (also with a yellow box).
    - Description:** ASP.NET Core MVC is a web framework that gives you a powerful, patterns-based way to build dynamic websites and web APIs. ASP.NET Core MVC enables a clean separation of concerns and gives you full control over markup.
    - Version:** 2.2.0
    - Downloads:** 106M
  - Right pane:** Details for **Microsoft.AspNetCore.Mvc** version 2.2.0.
    - Version:** Latest stable 2.2.0 (highlighted with a yellow box)
    - Install** button (highlighted with a yellow box)
    - Description:** ASP.NET Core MVC is a web framework that gives you a powerful, patterns-based way to build dynamic websites and web APIs. ASP.NET Core MVC enables a clean separation of concerns and gives you full control over markup.
    - Version:** 2.2.0
    - Author(s):** Microsoft
    - License:** [View License](#)
    - Date published:** Tuesday, December 4, 2018 (12/4/2018)
    - Project URL:** <https://asp.net/>
    - Report Abuse:** <https://www.nuget.org/packages/Microsoft.AspNetCore.Mvc/2.2.0/ReportAbuse>
    - Tags:** aspnetcore, aspnetcoremvc
    - Dependencies:**
      - .NETStandardVersion=v2.0
      - Microsoft.AspNetCore.Mvc.Analyzers (>= 2.2.0)
- Output Window:** Shows the build and publish logs.
- Error List:** Ready.
- Add to Source Control:** Add to Source Control button.

# Unit test an Azure Function - 7

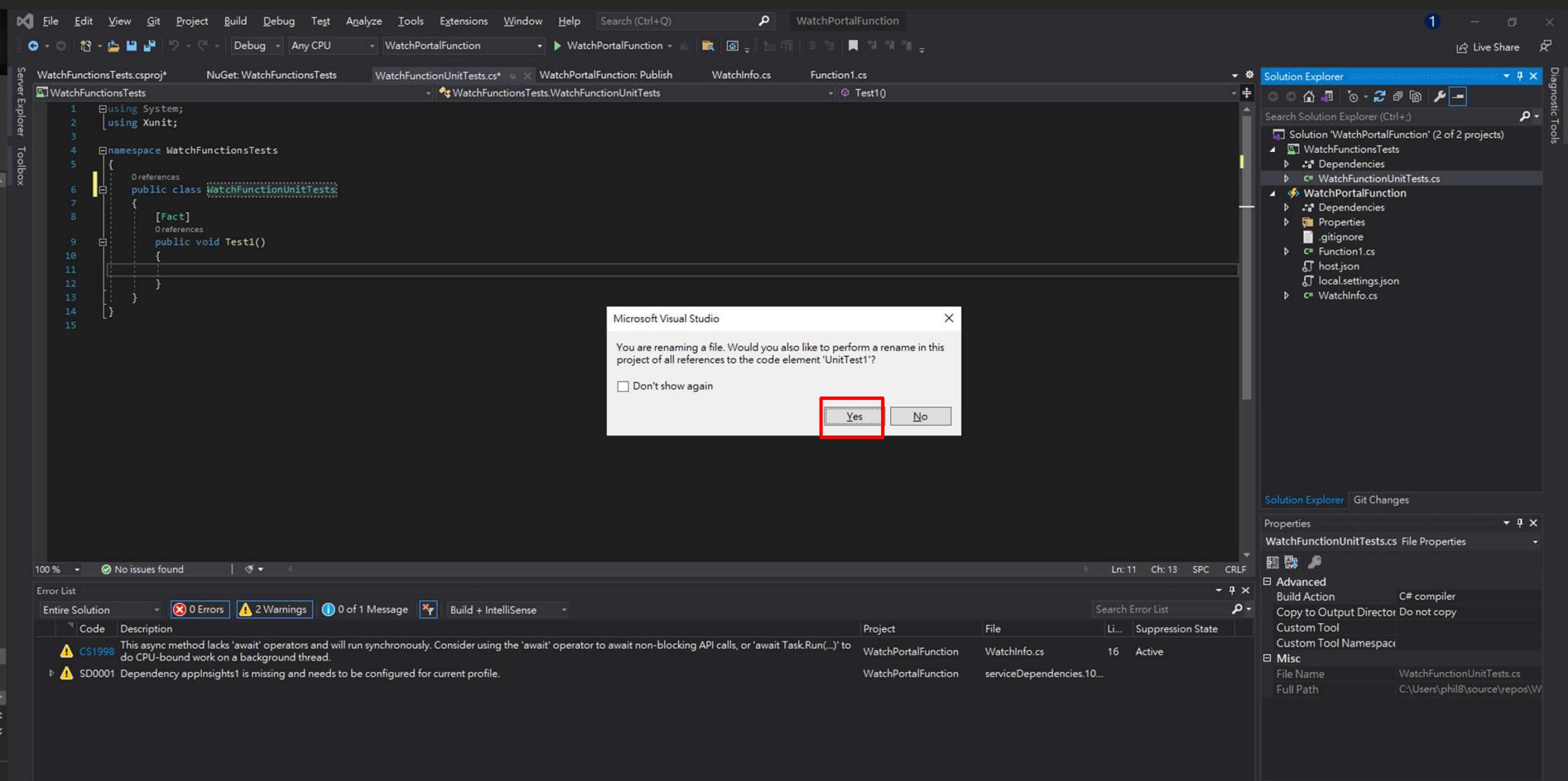


# Unit test an Azure Function - 8

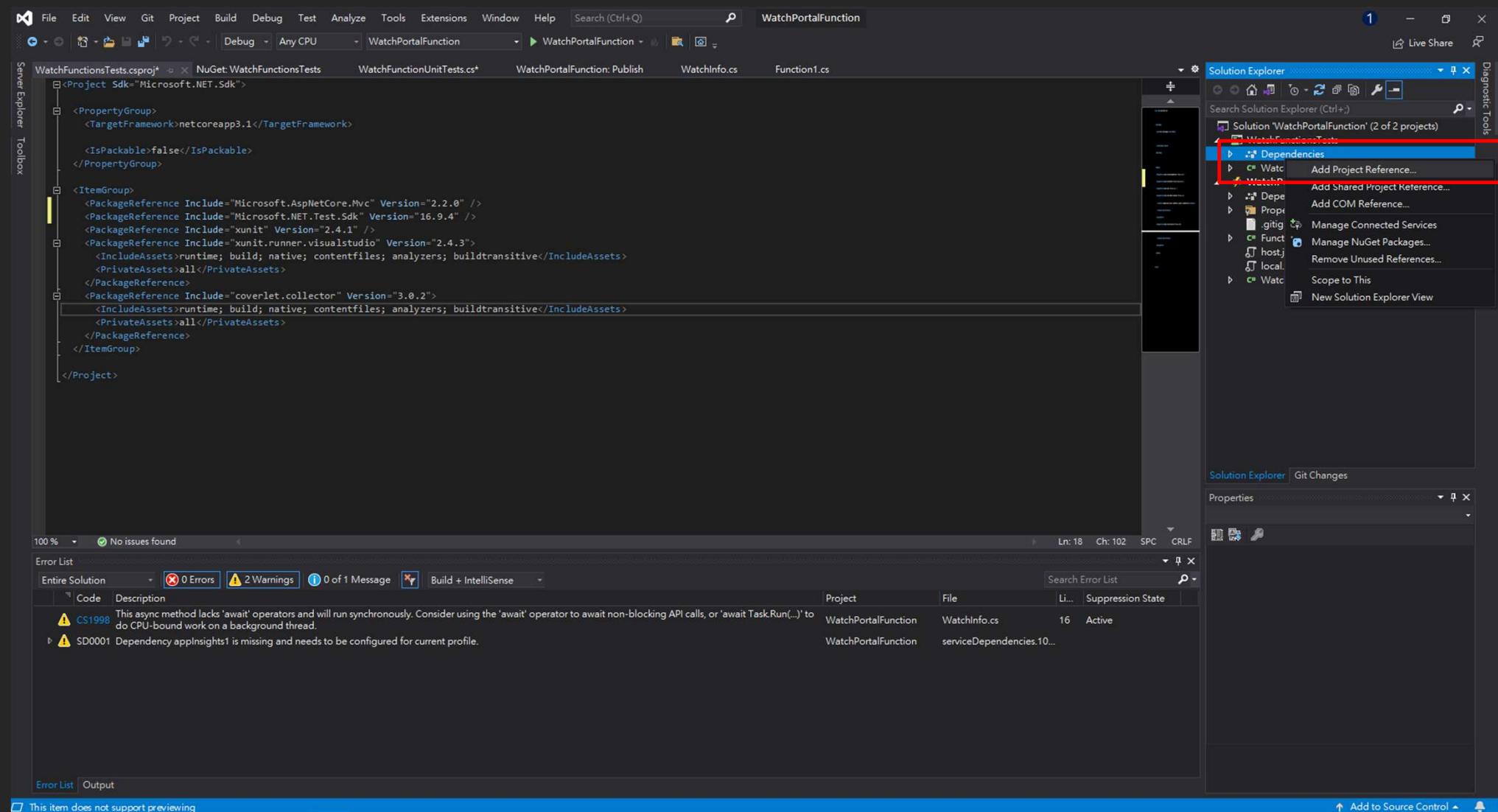
The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Toolbars:** Standard, Debug, Task List.
- Solution Explorer:** Shows two projects: WatchFunctionsTests (2 of 2 projects) and WatchPortalFunction. In WatchFunctionsTests, UnitTest1.cs is selected. A context menu is open over UnitTest1.cs, with "Rename" highlighted.
- Properties Window:** Shows properties for UnitTest1.cs under the "Advanced" section, including Build Action (C# compiler), Copy to Output Directory (Do not copy), and Custom Tool (None). It also lists File Name (UnitTest1.cs) and Full Path (C:\Users\philb\source\repos\WatchFunctionsTests\UnitTest1.cs).
- Error List:** Shows 2 Warnings:
  - CS1998: This async method lacks 'await' operators and will run synchronously. Consider using the 'await' operator to await non-blocking API calls, or 'await Task.Run(...)' to do CPU-bound work on a background thread.
  - SD0001: Dependency apliInsights1 is missing and needs to be configured for current profile.
- Output Tab:** Ready.

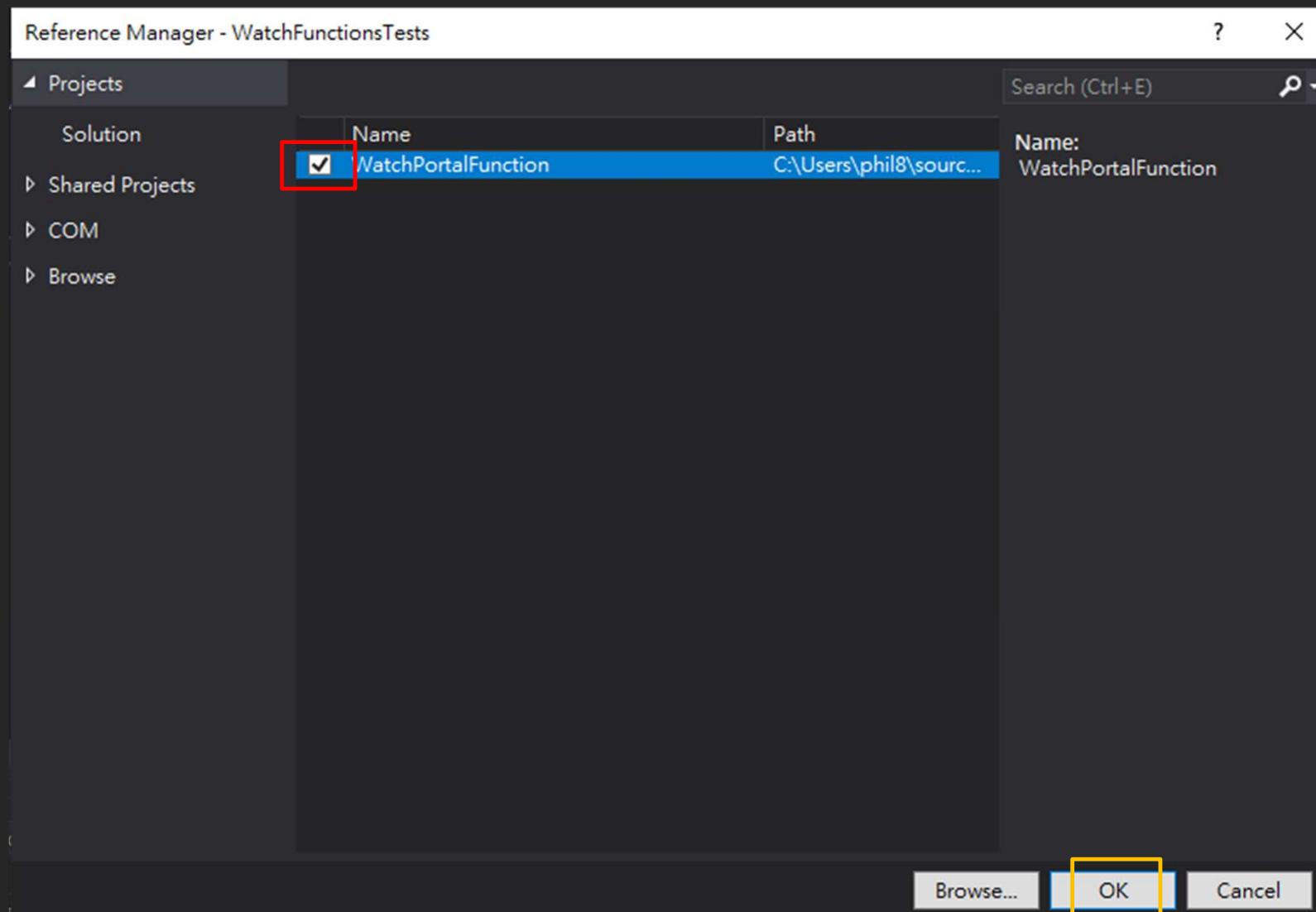
# Unit test an Azure Function - 9



# Unit test an Azure Function - 10



# Unit test an Azure Function - 11

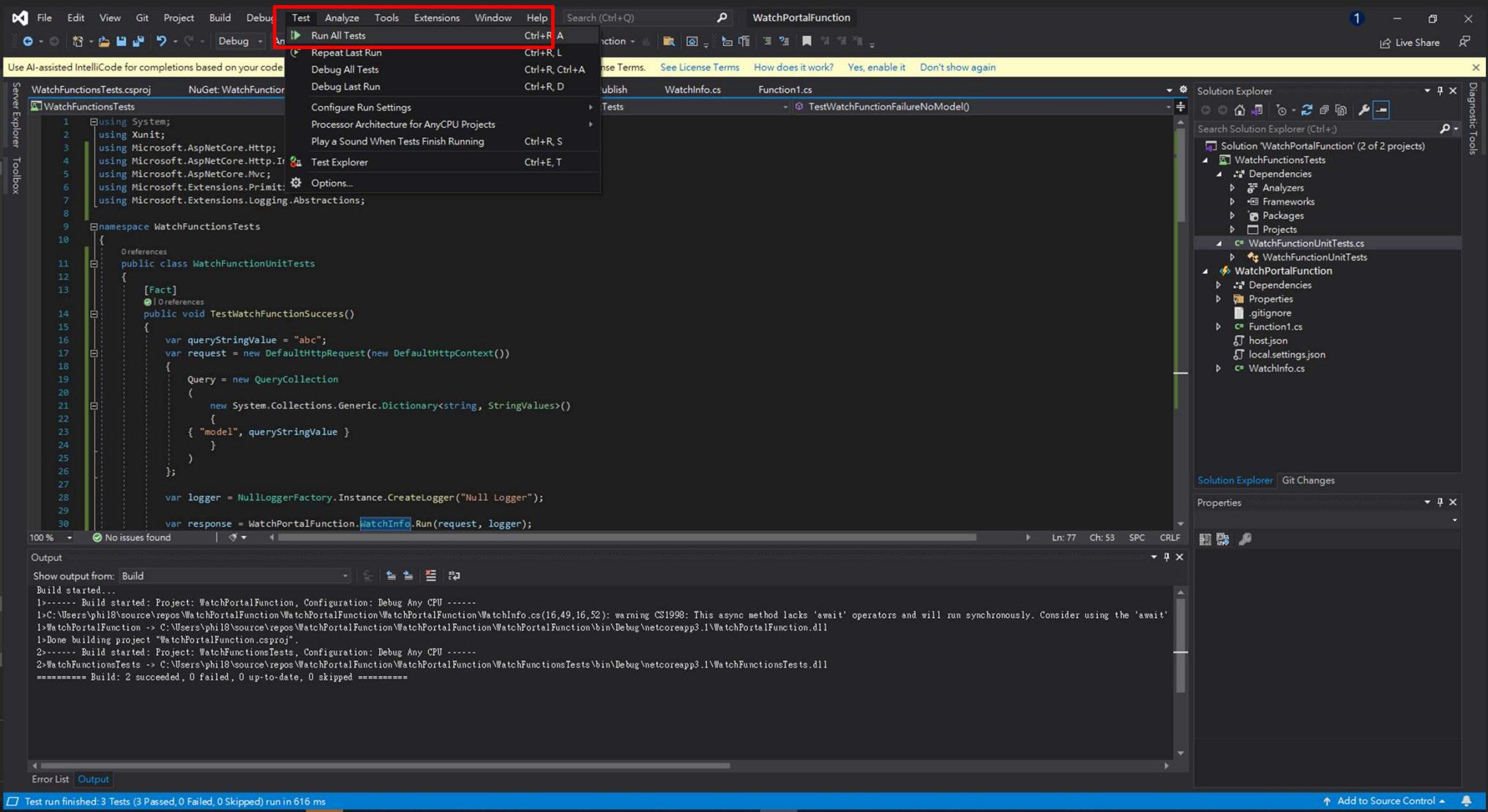


# Unit test an Azure Function - 12

The screenshot shows the Visual Studio IDE interface with the following details:

- Code Editor:** The main window displays the `WatchFunctionUnitTests.cs` file, which contains unit test code for an Azure Function. The code uses XUnit and Microsoft.AspNetCore.Http libraries. A yellow box highlights the entire code block.
- Solution Explorer:** On the right, the Solution Explorer shows two projects:
  - WatchFunctionsTests:** Contains files like `WatchFunctionUnitTests.cs` (highlighted with a red box) and `WatchFunctionUnitTests.cs`.
  - WatchPortalFunction:** Contains files like `host.json`, `local.settings.json`, and `WatchInfo.cs`.
- Error List:** At the bottom left, the Error List shows 0 Errors, 2 Warnings, and 0 Messages. One warning is listed: CS1998 (This async method lacks 'await' operators and will run synchronously. Consider using the 'await' operator to await non-blocking API calls, or 'await Task.Run(...)' to do CPU-bound work on a background thread.) and SD0001 (Dependency appInsights1 is missing and needs to be configured for current profile).
- Status Bar:** The status bar at the bottom indicates "Item(s) Saved".

Unit test an Azure Function - 13



## Unit test an Azure Function - 14

Test Explorer

3 3 0

Search Test Explorer (Ctrl+E)

Test	Duration	Traits	Error Message
WatchFunctionsTests (3)	52 ms		
WatchFunctionsTests (3)	52 ms		
WatchFunctionUnitTests (3)	52 ms		
TestWatchFunctionFailureNo...	4 ms		
TestWatchFunctionFailureNo...	9 ms		
TestWatchFunctionSuccess	39 ms		

# Unit test an Azure Function - 15

名字隨意，單純測Error用

```
7  using Microsoft.AspNetCore.Http;
8  using Microsoft.Extensions.Logging;
9  using Newtonsoft.Json;
10 
11 namespace WatchPortalFunction
12 {
13     public static class WatchInfo
14     {
15         [FunctionName("WatchInfo")]
16         [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
17         ILogger log
18     {
19         log.LogInformation("C# HTTP trigger function processed a request.");
20 
21         // Retrieve the model id from the query string
22         string model = req.Query["model=nonsenseword"];
23 
24         // If the user specified a model id, find the details of the model of watch
25         if (model != null)
26         {
27             // Use dummy data for this example
28             dynamic watchinfo = new { Manufacturer = "abc", CaseType = "Solid", Bezel = "Titanium", Dial = "Roman", CaseFinish = "Silver", Jewels = 15 };
29 
30             return (ActionResult)new OkObjectResult($"Watch Details: {watchinfo.Manufacturer}, {watchinfo.CaseType}, {watchinfo.Bezel}, {watchinfo.Dial}, {watchinfo.CaseFinish}, {watchinfo.Jewels}");
31         }
32         else
33             return new BadRequestObjectResult("Please provide a watch model in the query string");
34     }
35 }
36 }
```

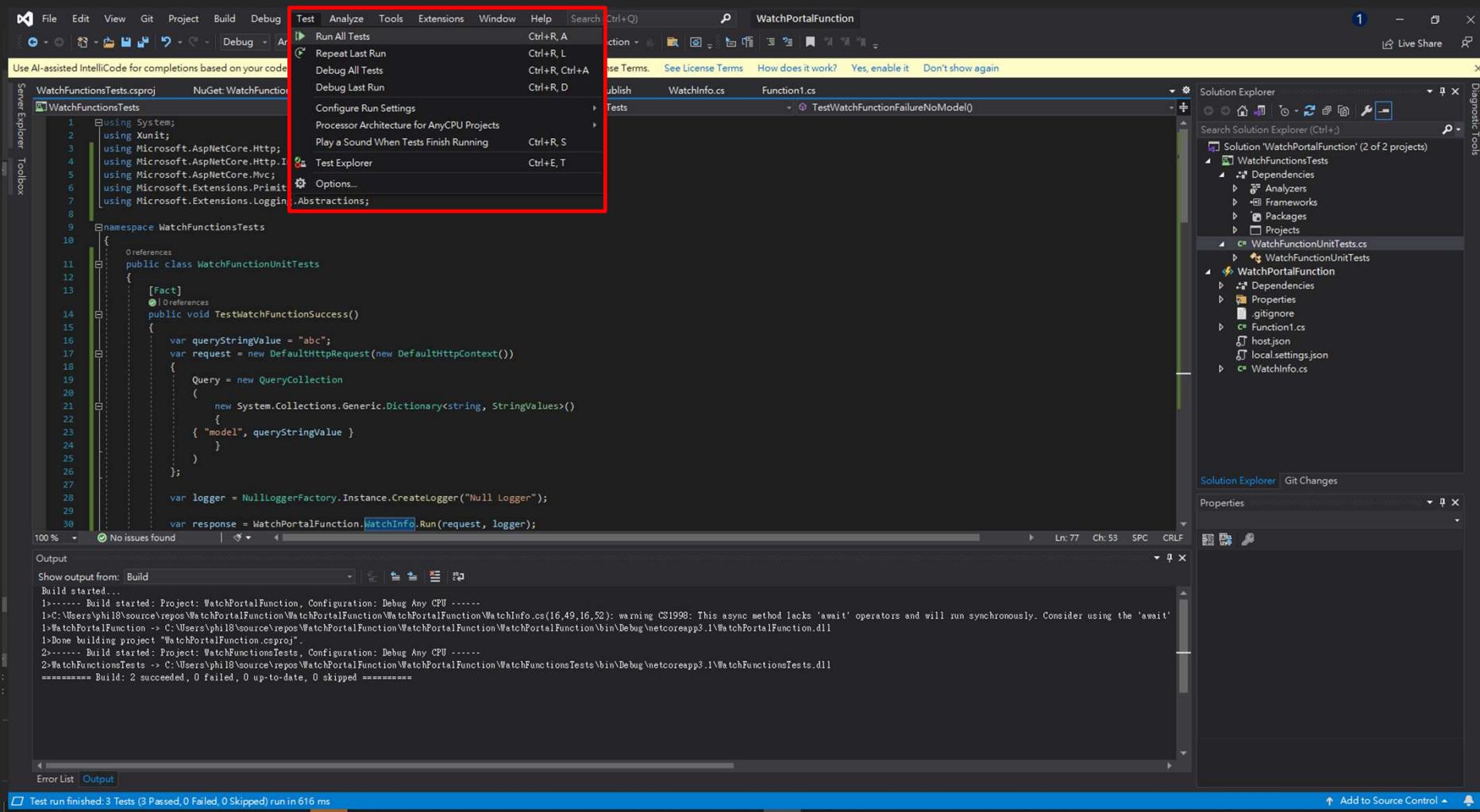
Output

```
Show output from: Build
Build started...
1>----- Build started: Project: WatchPortalFunction, Configuration: Debug Any CPU -----
1>C:\Users\phi18\source\repos\WatchPortalFunction\WatchPortalFunction\WatchInfo.cs(16,49,16,52): warning CS1998: This async method lacks 'await' operators and will run synchronously. Consider using the 'await'
1>WatchPortalFunction -> C:\Users\phi18\source\repos\WatchPortalFunction\WatchPortalFunction\bin\Debug\netcoreapp3.1\WatchPortalFunction.dll
1>Done building project "WatchPortalFunction.csproj".
2>----- Build started: Project: WatchFunctionsTests, Configuration: Debug Any CPU -----
2>WatchFunctionsTests -> C:\Users\phi18\source\repos\WatchPortalFunction\WatchFunctionsTests\bin\Debug\netcoreapp3.1\WatchFunctionsTests.dll
===== Build: 2 succeeded, 0 failed, 0 up-to-date, 0 skipped ======
```

Error List ... Output

Ready Add to Source Control

# Unit test an Azure Function - 16



# Unit test an Azure Function - 17

Test Explorer

3 2 1

Search Test Explorer (Ctrl+E)

Test	Duration	Traits	Error Message
WatchFunctionsTests (3)	17 ms		
WatchFunctionsTests (3)	17 ms		
WatchFunctionUnitTests (3)	17 ms		
TestWatchFunctionFailureNo... (3)	3 ms		
TestWatchFunctionFailureNo... (3)	8 ms		
TestWatchFunctionSuccess (3)	6 ms		Assert.IsNotNull() Failure E...

Group Summary

WatchFunctionsTests

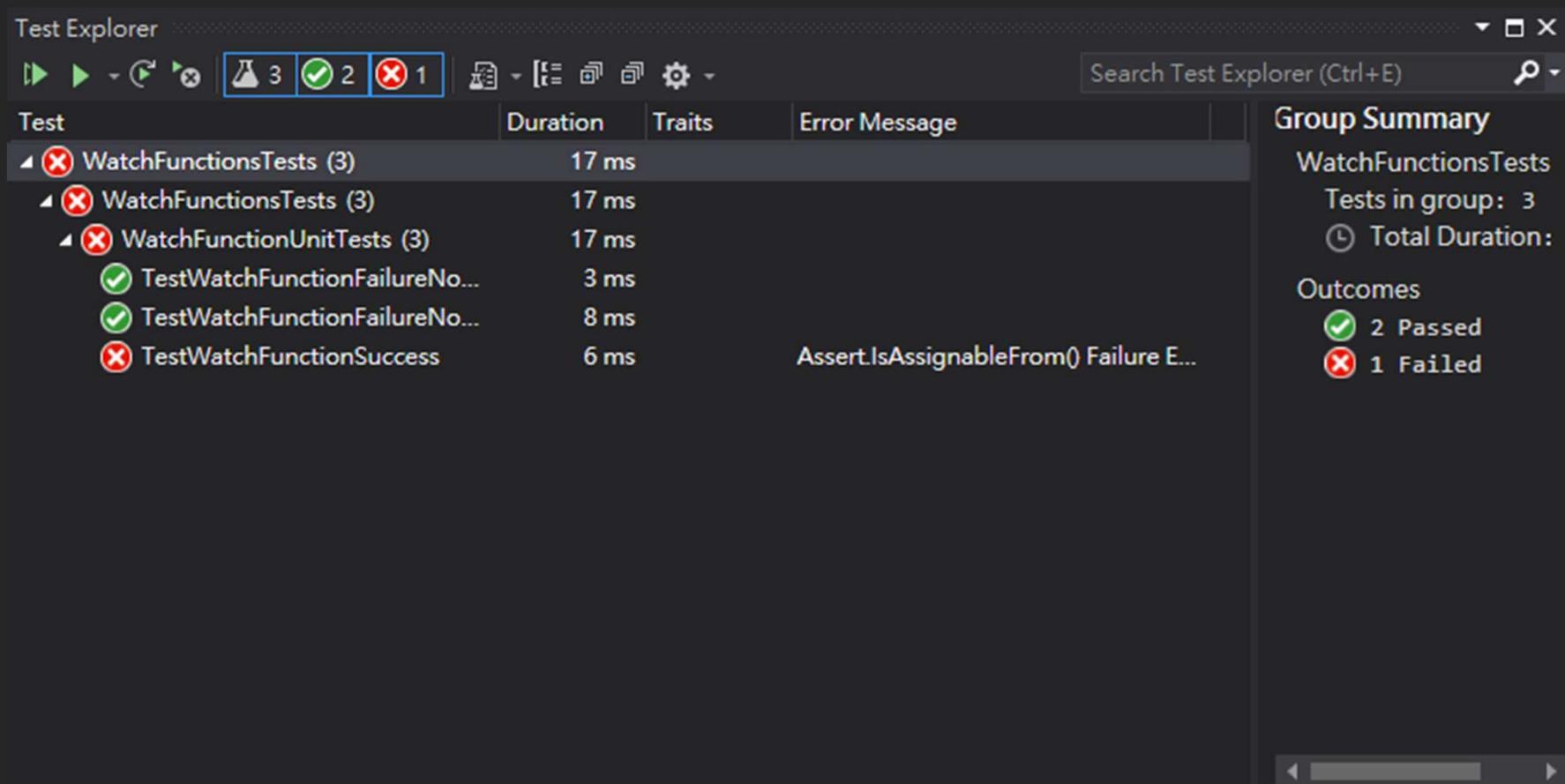
Tests in group: 3

Total Duration:

Outcomes

2 Passed

1 Failed



# Resource Check

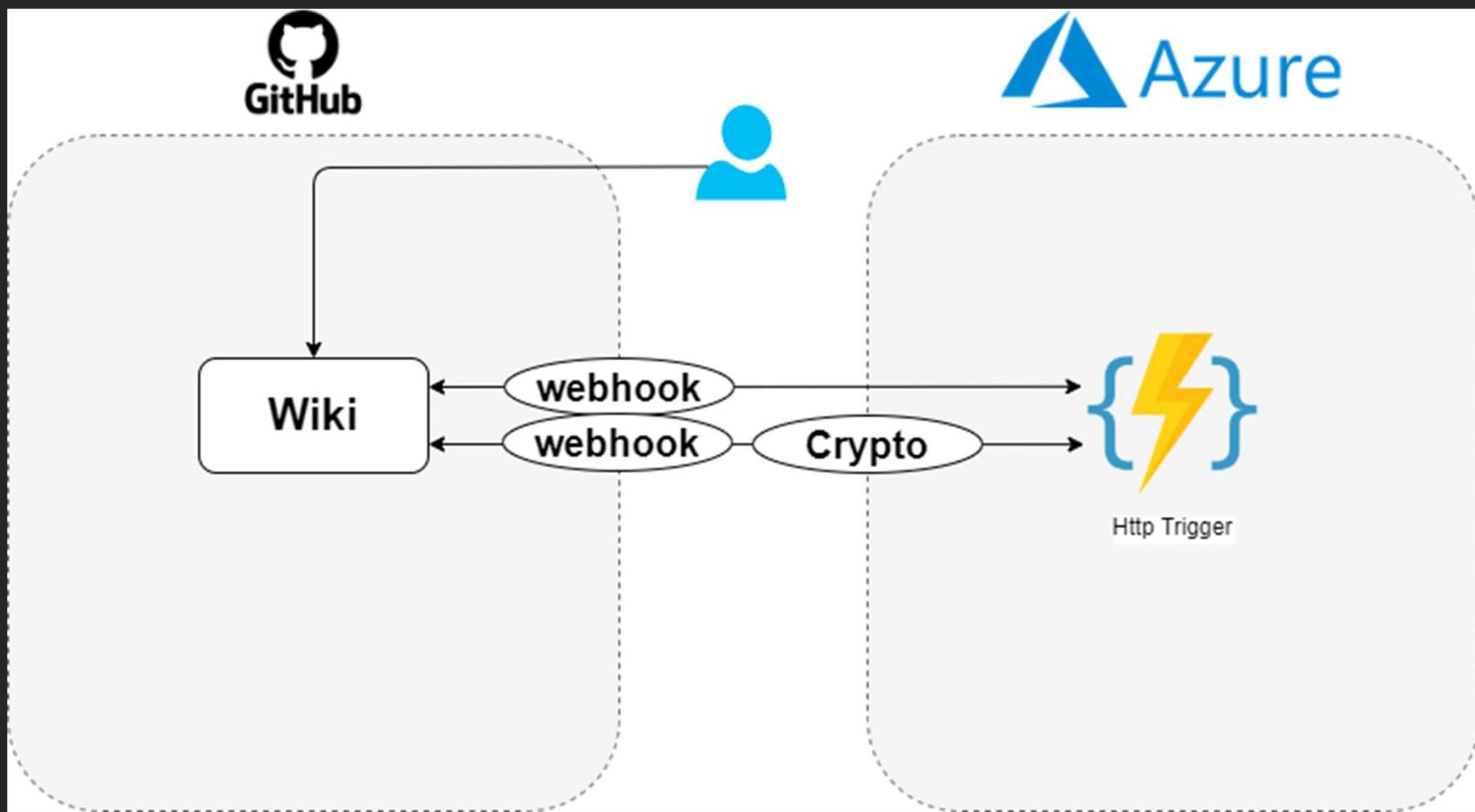
## Remove Resource

- cosmosdb
- functionapp

08

// Monitor GitHub events by using a  
webhook with Azure Functions

# Abstract



# Create a webhook triggered function

The screenshot shows the Microsoft Azure portal interface for creating a new function. The left sidebar navigation includes 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Security', 'Events (preview)', 'Functions' (selected), 'App keys', 'App files', 'Proxies', 'Deployment' (with 'Deployment slots' and 'Deployment Center'), and 'Settings' (with 'Configuration', 'Authentication', 'Application Insights', 'Identity', 'Backups', 'Custom domains', and 'TLS/SSL settings').

The main content area is titled 'Create function'. It starts with a 'Select development environment' section, which is set to 'Develop in portal'. Below that is a 'Select a template' section, where the 'HTTP trigger' template is highlighted with a red box. The 'Template details' section below it contains a note about needing more information to create the function, a 'New Function\*' input field containing 'M8-HttpTrigger', and a dropdown for '權限等級' (Authorization Level) set to 'Function'. A yellow box highlights the 'Create' button at the bottom.

Template	Description
HTTP trigger	接收到 HTTP 要求時，將會執行的函式，回應取決於本文或查詢字串中的資料
Timer trigger	於指定的排程執行之函式
Azure Queue Storage trigger	訊息新增至指定的 Azure 儲存體佇列時，將會執行的函式
Azure Service Bus Queue trigger	每次有訊息新增到指定服務匯流排佇列時都會執行的函式
Azure Service Bus Topic trigger	每次有訊息新增到指定服務匯流排主題時都會執行的函式
Azure Blob Storage trigger	Blob 每次新增至指定的容器時，將會執行的函式
Azure Event Hub trigger	每當事件中樞接收到新的事件時就會執行的函式

# Set up a webhook for a GitHub repository - 1

The screenshot shows the GitHub homepage. At the top right, there is a 'New' button with a plus sign, which is highlighted with a red box. A dropdown menu is open from this button, listing options: 'New repository', 'Import repository', 'New gist', 'New organization', and 'New project'. The rest of the page includes a search bar, navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore', and sections for 'Repositories' and 'Recent activity'.

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

[Read the guide](#) [Start a project](#)

huihui0310tt/CYCU-Project  
huihui0310tt/FL-Project  
AutumnLai/AIMFL

All activity

Introduce yourself

The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:

huihui0310tt / README.md

```
1 - 🌟 Hi, I'm @huihui0310tt
2 - 💬 I'm interested in ...
3 - 🚀 I'm currently learning ...
4 - 🤝 I'm looking to collaborate on ...
5 - 📧 How to reach me ...
6
```

[Dismiss this](#) [Continue](#)

Discover interesting projects and people to populate your personal news feed.

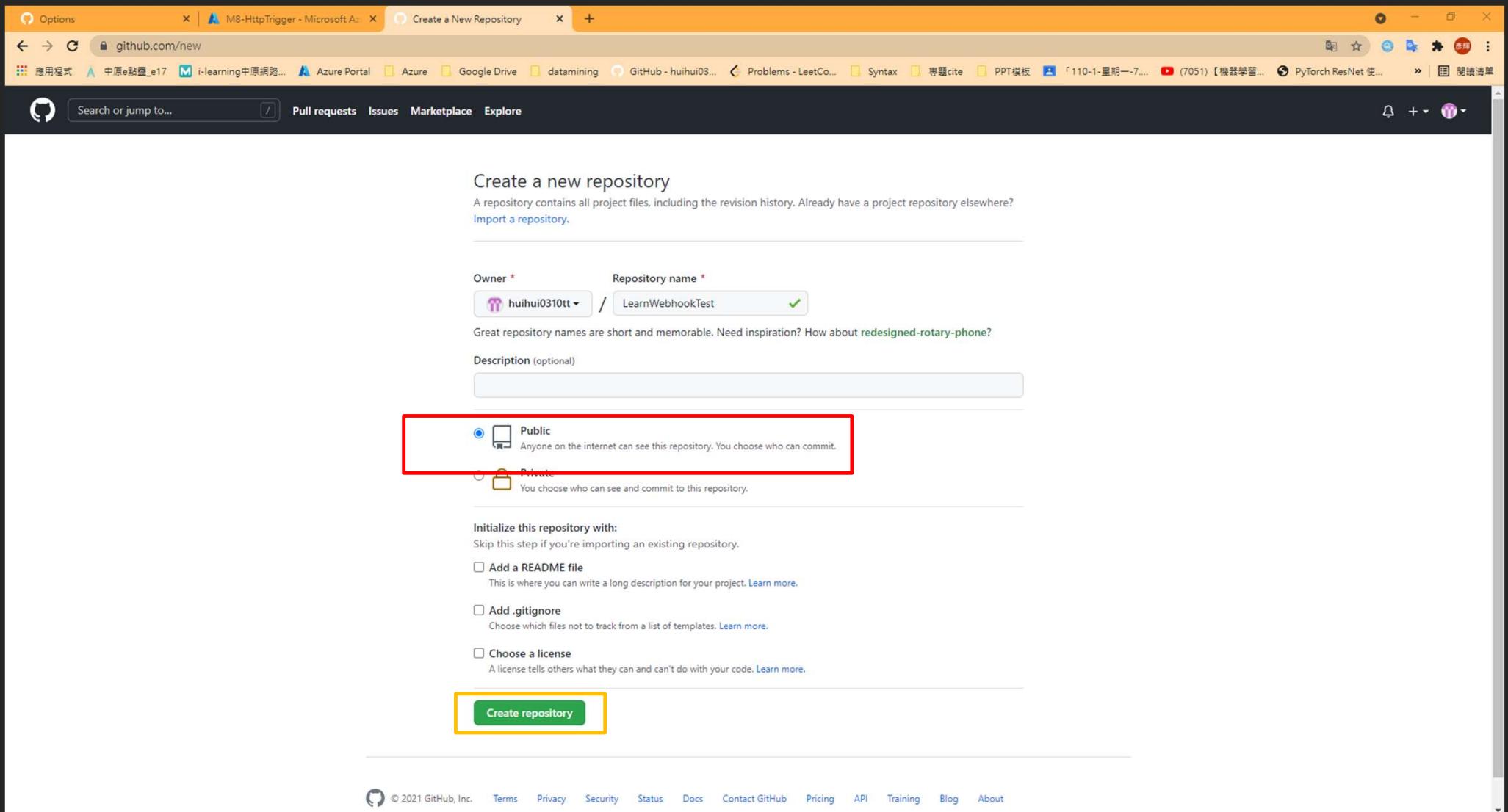
Your news feed helps you keep up with recent activity on repositories you [watch](#) or [star](#) and people you [follow](#).

[Explore GitHub](#)

💡 ProTip! The feed shows you events from people you follow and repositories you [watch](#) or [star](#).  
[Subscribe to your news feed](#)

<https://github.com/new>

# Set up a webhook for a GitHub repository - 2



The screenshot shows the GitHub 'Create a new repository' interface. At the top, there's a navigation bar with tabs like 'Options', 'Create a New Repository', and a search bar. Below the navigation is a toolbar with links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The main content area is titled 'Create a new repository' and asks if the user already has a project repository elsewhere, with a link to 'Import a repository'. It then prompts for the 'Owner' (set to 'huihui0310tt') and 'Repository name' ('LearnWebhookTest'). A note suggests short and memorable names. The 'Description (optional)' field is empty. Below this, there are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option is described as allowing anyone on the internet to see the repository. The 'Private' option is described as allowing the user to choose who can see and commit. A red box highlights the 'Public' option. Further down, there's a section for initializing the repository with 'README file', '.gitignore', and 'license', each with an unchecked checkbox. At the bottom is a large green 'Create repository' button.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner \* Repository name \*

huihui0310tt / LearnWebhookTest ✓

Great repository names are short and memorable. Need inspiration? How about [redesigned-rotary-phone](#)?

Description (optional)

 Public  
Anyone on the internet can see this repository. You choose who can commit.

 Private  
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file  
This is where you can write a long description for your project. [Learn more](#).

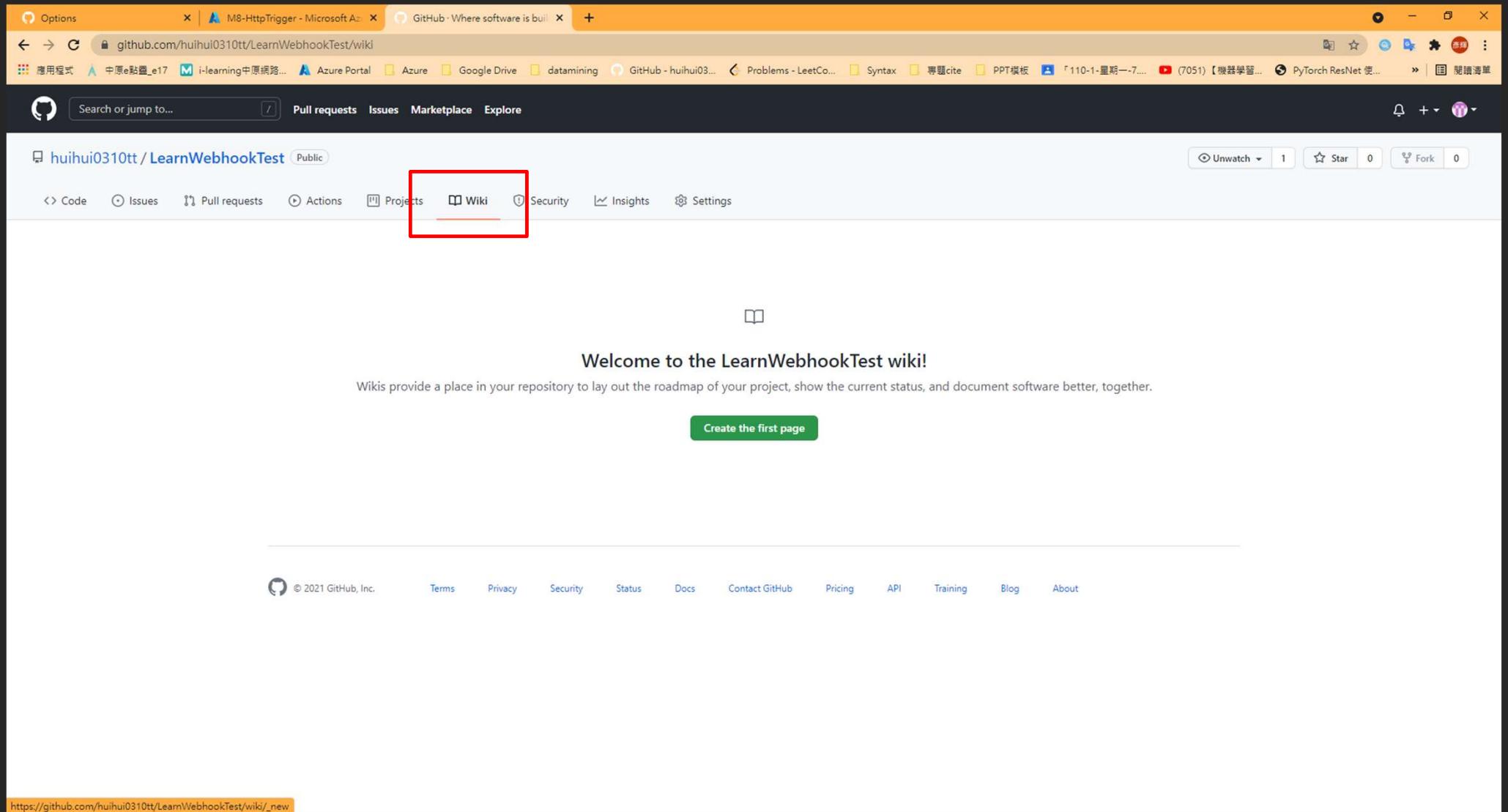
Add .gitignore  
Choose which files not to track from a list of templates. [Learn more](#).

Choose a license  
A license tells others what they can and can't do with your code. [Learn more](#).

[Create repository](#)

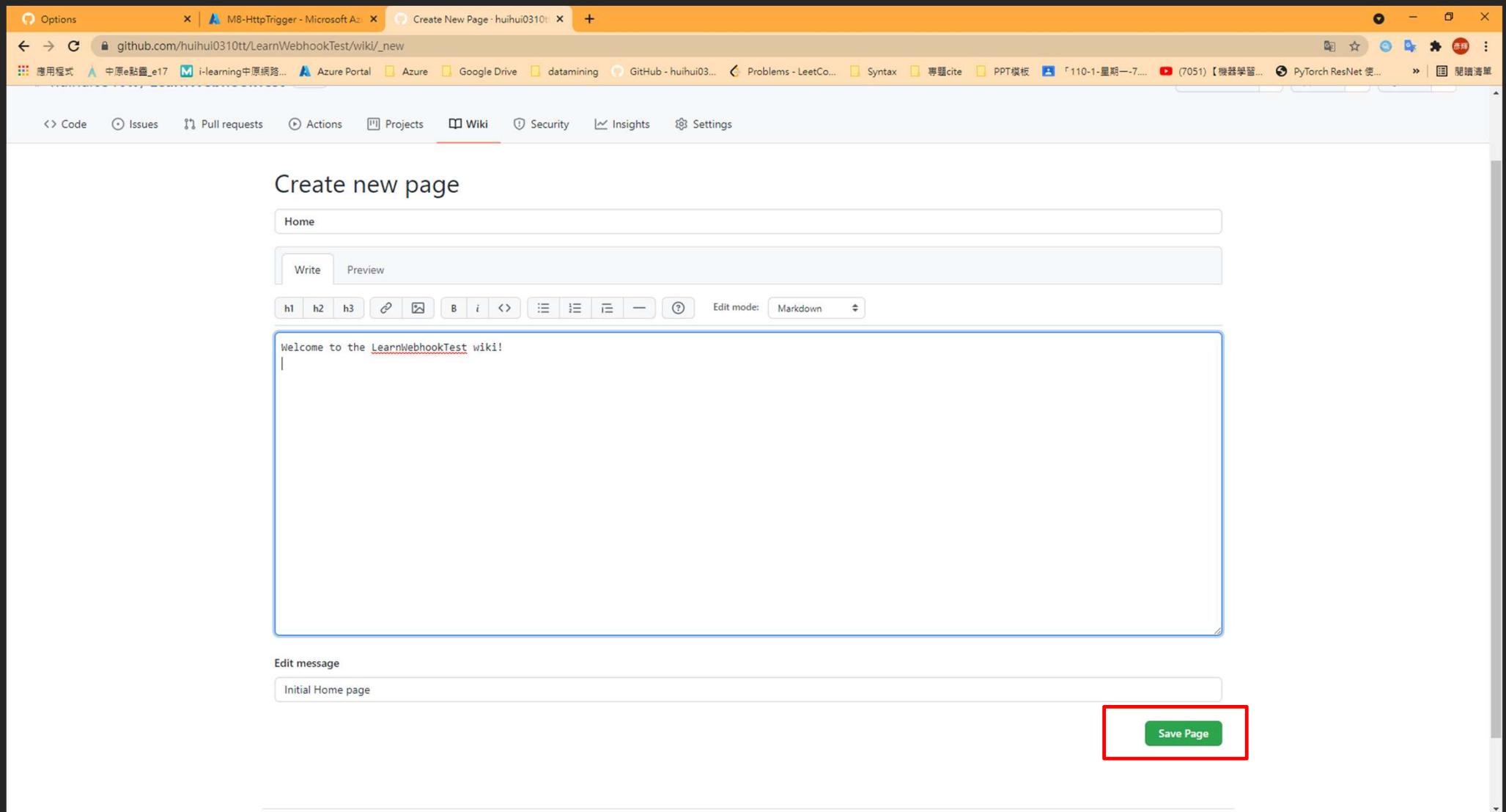
© 2021 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

# Add a webhook for the Gollum event - 1



The screenshot shows a GitHub repository page for 'huihui0310tt/LearnWebhookTest'. The 'Wiki' tab is highlighted with a red box. The page displays a welcome message: 'Welcome to the LearnWebhookTest wiki! Wikis provide a place in your repository to lay out the roadmap of your project, show the current status, and document software better, together.' A green button labeled 'Create the first page' is visible. The bottom of the page includes standard GitHub footer links: Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About. The URL 'https://github.com/huihui0310tt/LearnWebhookTest/wiki/\_new' is shown at the bottom left.

## Add a webhook for the Gollum event - 2



# Add a webhook for the Gollum event - 3

The screenshot shows a browser window with multiple tabs open. The active tab is 'Webhooks' under the 'Settings' section of a GitHub repository named 'huihui0310tt/LearnWebhookTest'. The left sidebar menu is highlighted with a yellow box around the 'Webhooks' item. The main content area displays the 'Webhooks' configuration page, which includes a descriptive text about what webhooks do and a prominent 'Add webhook' button. A red box highlights the 'Settings' tab in the top navigation bar. A yellow box also highlights the 'Add webhook' button.

github.com/huihui0310tt/LearnWebhookTest/settings/hooks

Options | M8-HttpTrigger - Microsoft Azure | Webhooks

Search or jump to... Pull requests Issues Marketplace Explore

huihui0310tt / LearnWebhookTest (Public)

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

Options  
Manage access  
Security & analysis  
Webhooks  
Notifications  
Integrations  
Deploy keys  
Actions  
Environments  
Secrets  
Pages  
Moderation settings

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

<https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks/new>

# Add a webhook for the Gollum event - 4

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for 'Options', 'M8-HttpTrigger - Microsoft Azure', and 'Add webhook'. Below the navigation bar is a toolbar with icons for 'Search resources, services, and docs (G+)', 'Home', 'Microsoft Azure', and user information. The main content area displays the 'M8-HttpTrigger' function details. On the left, there is a sidebar with sections for 'Developer' (Code + Test, Integration, Monitor, Function Keys), 'Overview', and a search bar. The main panel shows the 'Get Function Url' dialog. It has a dropdown menu set to 'default (function key)' and a URL input field containing 'https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKiTCO8fyrtlaPvp7sSA6WNuGoKor85jpjQwKkU...'. A yellow box highlights the URL input field. To the right of the URL is a 'Copy to clipboard' button, also highlighted with a yellow box. Below the URL, it shows 'Subscription (move) 中原大學' and 'Subscription ID ba9f5e30-1488-49db-bae0-9026dacc11b0'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

# Add a webhook for the Gollum event - 5

The screenshot shows a browser window with the URL [github.com/huihui0310tt/LearnWebhookTest/settings/hooks/new](https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks/new). The page is titled "Add webhook". On the left, there is a sidebar with the following options: Options, Manage access, Security & analysis, Webhooks (which is selected and highlighted in red), Notifications, Integrations, Deploy keys, Actions, Environments, Secrets, Pages, and Moderation settings. The main content area has a heading "Webhooks / Add webhook" and a sub-instruction: "We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation." Below this, there is a "Payload URL \*" input field containing the URL <https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger1>, which is highlighted with a red box. Under "Content type", there is a dropdown menu set to "application/json", which is highlighted with a yellow box. Further down, there is a "Secret" input field, an "SSL verification" section with a note about verifying SSL certificates and two radio buttons for "Enable SSL verification" (selected) and "Disable (not recommended)", and a section for selecting events. The "Let me select individual events" option is highlighted with a yellow box. Under this, there are several checkboxes for different events, each with a brief description:

- Branch or tag creation  
Branch or tag created.
- Branch or tag deletion  
Branch or tag deleted.
- Branch protection rules  
Branch protection rule created, deleted or edited.
- Check runs  
Check run is created, requested, rerequested, or completed.
- Check suites
- Code scanning alerts

# Add a webhook for the Gollum event - 6

The screenshot shows a Microsoft Edge browser window with the URL [github.com/huihui0310tt/LearnWebhookTest/settings/hooks/new](https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks/new). The title bar says "Add webhook". The page displays a list of GitHub events:

- Pushes**: Git push to a repository.
- Releases**: Release created, edited, published, unpublished, or deleted.
- Repository imports**: Repository import succeeded, failed, or cancelled.
- Secret scanning alerts**: Secrets scanning alert created, resolved, or reopened.
- Statuses**: Commit status updated from the API.
- Visibility changes**: Repository changes from private to public.
- Wiki**: Wiki page updated.
- Workflow runs**: Workflow run requested or completed on a repository.

A red box highlights the "Wiki" checkbox, and another red box highlights the "Add webhook" button at the bottom.

**只選Wiki  
其他不勾**

At the bottom of the form, there is a note: "We will deliver event details when this hook is triggered." Below this note is a green "Add webhook" button, which is also highlighted with a yellow border.

At the very bottom of the page, there is a footer with links: Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About.

# Test the webhook - 1

The screenshot shows a GitHub repository page for 'huihui0310tt/LearnWebhookTest'. The 'Wiki' tab is selected. On the right side of the page, there are two buttons: 'Edit' (gray) and 'New Page' (green). A red box highlights the 'Edit' button. Below these buttons is a sidebar with a 'Pages' section containing a single entry for 'Home'. There are also sections for adding a custom footer and a custom sidebar. At the bottom of the page, there is a 'Clone this wiki locally' section with a URL: <https://github.com/huihui0310tt/LearnWebhookTest/wiki>. The browser's address bar at the bottom also displays this URL.

Options M8-HttpTrigger - Microsoft Az... Home : huihui0310tt/LearnWebhookTest/wiki

github.com/huihui0310tt/LearnWebhookTest/wiki

應用程式 中原e點通\_e17 i-learning中原網路... Azure Portal Azure Google Drive datamining GitHub - huihui03... Problems - LeetCode... Syntax 專題cite PPT模板 「110-1-星期一-7.... (7051) 【機器學習... PyTorch ResNet 使... 閱讀清單

Search or jump to... Pull requests Issues Marketplace Explore

huihui0310tt / LearnWebhookTest Public

Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

## Home

huihui0310tt edited this page 8 minutes ago · 1 revision

Welcome to the LearnWebhookTest wiki!

+ Add a custom footer

+ Add a custom sidebar

Pages 1

Find a Page...  
Home

Clone this wiki locally

<https://github.com/huihui0310tt/LearnWebhookTest/wiki>

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

[https://github.com/huihui0310tt/LearnWebhookTest/wiki/Home/\\_edit](https://github.com/huihui0310tt/LearnWebhookTest/wiki/Home/_edit)

# Test the webhook - 2

The screenshot shows a GitHub repository interface. The top navigation bar includes tabs for Options, M8-HttpTrigger - Microsoft Azure, and Editing Home · huihui0310tt/LearnWebhookTest · Wiki · Home · \_edit. Below the navigation is a toolbar with links to various services like GitHub, LeetCode, and PPT templates. The main content area is titled "Editing Home". It features a header input field containing "Home", a toolbar with "Write" and "Preview" buttons, and a rich text editor with a blue border. Inside the editor, there is a message: "Welcome to the LearnWebhookTest wiki!  
Some words update for webhook". Below the editor is an "Edit message" section with a placeholder "Write a small message here explaining this change. (Optional)". A green "Save Page" button is located at the bottom right of this section, which is highlighted with a red rectangular box.

# Test the webhook - 3

The screenshot shows a Microsoft Edge browser window displaying the GitHub settings page for the repository "huihui0310tt / LearnWebhookTest". The URL in the address bar is [github.com/huihui0310tt/LearnWebhookTest/settings/hooks](https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks). The page is titled "Webhooks" and contains a list of configured webhooks. A red box highlights the first webhook entry, which has a green checkmark and the URL <https://10727211-functionapp.azurewebsites.net/>. To the right of this entry are two buttons: "Edit" (highlighted with a yellow box) and "Delete". On the left, a sidebar menu lists various GitHub settings options: Options, Manage access, Security & analysis, Webhooks (which is currently selected and highlighted with a red box), Notifications, Integrations, Deploy keys, Actions, Environments, Secrets, Pages, and Moderation settings.

# Test the webhook - 4

github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries

Search or jump to... Pull requests Issues Marketplace Explore

huihui0310tt / LearnWebhookTest Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Webhooks / Manage webhook

Settings Recent Deliveries

Delivery ID	Time	...
544b87e0-4e02-11ec-9055-0e9581be86d	2021-11-25 23:14:06	[...]
7a75d840-4e01-11ec-808b-8f21c8e823ad	2021-11-25 23:08:01	[...]

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

# Test the webhook - 5

Options

Manage access

Security & analysis

**Webhooks**

Notifications

Integrations

Deploy keys

Actions

Environments

Secrets

Pages

Moderation settings

## Webhooks / Manage webhook

Settings Recent Deliveries

✓ 544b87e0-4e02-11ec-9055-0e9581be86d1 2021-11-25 23:14:06 ...

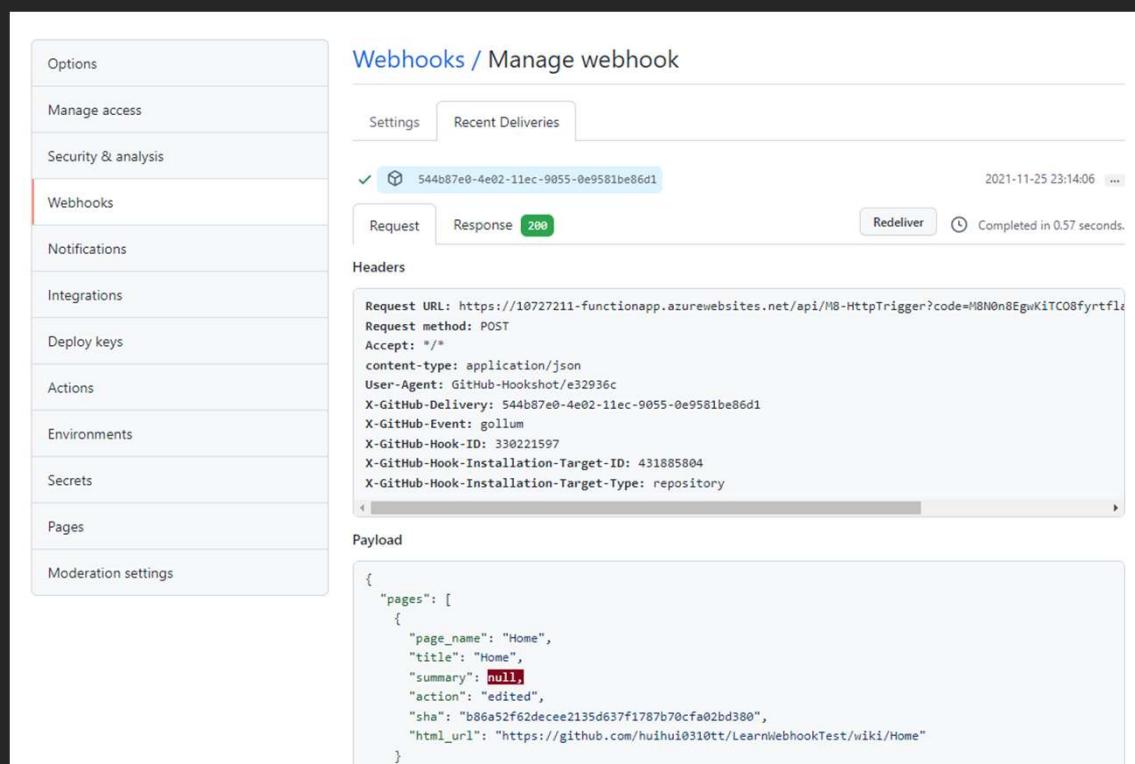
Request Response 200 Redeliver Completed in 0.57 seconds.

Headers

```
Request URL: https://1027211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKiTC08fyrtfl...
Request method: POST
Accept: */*
content-type: application/json
User-Agent: Github-Hookshot/e32936c
X-GitHub-Delivery: 544b87e0-4e02-11ec-9055-0e9581be86d1
X-GitHub-Event: gollum
X-GitHub-Hook-ID: 330221597
X-GitHub-Hook-Installation-Target-ID: 431885804
X-GitHub-Hook-Installation-Target-Type: repository
```

Payload

```
{ "pages": [ { "page_name": "Home", "title": "Home", "summary": null, "action": "edited", "sha": "b86a52f62decee2135d637f1787b70cfa02bd380", "html_url": "https://github.com/huihui0310tt/LearnWebhookTest/wiki/Home" } ] }
```



ki Security Insights Settings

## Webhooks / Manage webhook

Settings Recent Deliveries

✓ 544b87e0-4e02-11ec-9055-0e9581be86d1 2021-11-25 23:14:06 ...

Request Response 200 Redeliver Completed in 0.57 seconds.

Headers

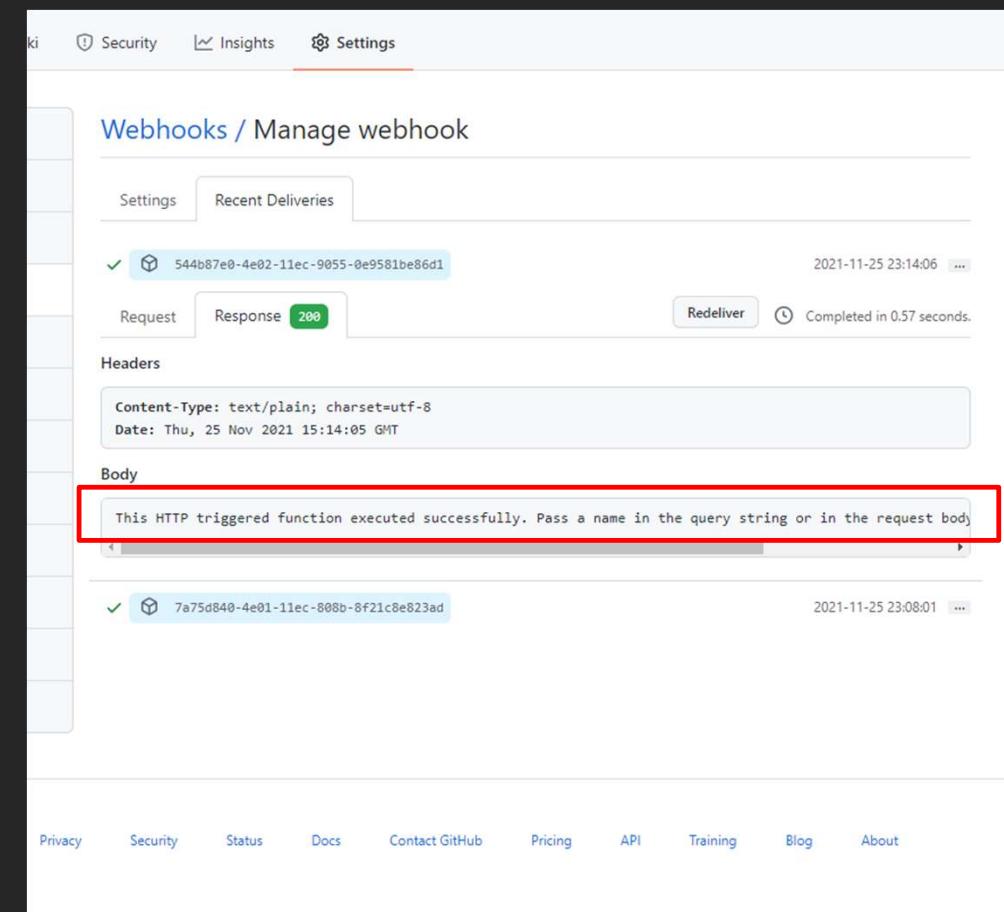
```
Content-Type: text/plain; charset=utf-8
Date: Thu, 25 Nov 2021 15:14:05 GMT
```

Body

```
This HTTP triggered function executed successfully. Pass a name in the query string or in the request body
```

✓ 7a75d840-4e01-11ec-808b-8f21c8e823ad 2021-11-25 23:08:01 ...

Privacy Security Status Docs Contact GitHub Pricing API Training Blog About



# Exercise - Trigger an Azure Function with a GitHub event - 1

The screenshot shows the Microsoft Azure portal interface for managing Azure Functions. The top navigation bar includes tabs for Options, M8-HttpTrigger - Microsoft Azure, and Webhook Deliveries. Below the navigation is a toolbar with icons for Save, Discard, Refresh, Test/Run, Upload, and Get function URL.

The main content area displays the Azure Function details for "M8-HttpTrigger". On the left, a sidebar titled "Developer" lists "Code + Test" (which is selected), "Integration", "Monitor", and "Function Keys".

The central area shows the code editor for the "index.js" file. The code is as follows:

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

    const name = (req.query.name || (req.body && req.body.name));
    const responseMessage = name
        ? `Hello, ${name}. This HTTP triggered function executed successfully.`
        : `This HTTP triggered function executed successfully. Pass a name in the query string or in the request body for a personalized response.`;

    if (req.body.pages[0].title) {
        context.res = {
            body: `Page is ${req.body.pages[0].title}, Action is ${req.body.pages[0].action}, Event Type is ${req.headers['x-github-event']}`
        };
    } else {
        context.res = {
            status: 400,
            body: ("Invalid payload for Wiki event")
        };
    }
}
```

A red box highlights the main logic block from line 10 to line 20, which handles the request body to determine the response message and structure.

At the bottom of the code editor, there is a "Logs" tab.

# Exercise - Trigger an Azure Function with a GitHub event - 2

The screenshot shows a browser window with several tabs open. The active tab is 'Webhook Deliveries' under 'M8-HttpTrigger - Microsoft AZ'. The URL is [github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries](https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries). The GitHub sidebar on the left has 'Webhooks' selected. The main content area shows a 'Webhooks / Manage webhook' page for the repository 'huihui0310tt / LearnWebhookTest'. A recent delivery is listed, showing a successful '200' response. The 'Redeliver' button for this delivery is highlighted with a red box. Below it, the 'Headers' section shows 'Content-Type: text/plain; charset=utf-8' and 'Date: Thu, 25 Nov 2021 15:14:05 GMT'. The 'Body' section contains the message 'This HTTP triggered function executed successfully. Pass a name in the query string or in the request body.'

Options | M8-HttpTrigger - Microsoft AZ | Webhook Deliveries | +

github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries

Search or jump to... Pull requests Issues Marketplace Explore

huihui0310tt / LearnWebhookTest Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Webhooks / Manage webhook

Options Manage access Security & analysis Webhooks Notifications Integrations Deploy keys Actions Environments Secrets Pages Moderation settings

Recent Deliveries

544b87e0-4e02-11ec-9055-0e9581be86d1 2021-11-25 23:14:06 ...

Request Response 200 Redeliver Completed in 0.57 seconds.

Headers

Content-Type: text/plain; charset=utf-8  
Date: Thu, 25 Nov 2021 15:14:05 GMT

Body

This HTTP triggered function executed successfully. Pass a name in the query string or in the request body.

7a75d840-4e01-11ec-808b-8f21c8e823ad 2021-11-25 23:08:01 ...

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

# Exercise - Trigger an Azure Function with a GitHub event - 3

The screenshot shows a browser window with multiple tabs open. The active tab is 'Webhook Deliveries' for a repository named 'huihui0310tt/LearnWebhookTest'. A modal dialog box titled 'Redeliver payload?' is displayed in the center. The dialog contains the following text:

The payload will be delivered to <https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKITCO8fyrtflaPYp7sSA6WNuGoKor8SjpJQwKkUjhOrg==> using the current webhook configuration.

At the bottom of the dialog is a button labeled 'Yes, redeliver this payload' which is highlighted with a red border.

Below the dialog, the main page shows the webhook configuration details:

- Request**: Response 200
- Redeliver**: Completed in 0.57 seconds
- Headers**: Content-Type: text/plain; charset=utf-8, Date: Thu, 25 Nov 2021 15:14:05 GMT
- Body**: This HTTP triggered function executed successfully. Pass a name in the query string or in the request body
- Event Log**: 7a75d840-4e01-11ec-808b-8f21c8e823ad, 2021-11-25 23:08:01

At the bottom of the page, there are links for GitHub's footer: Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About.

# Exercise - Trigger an Azure Function with a GitHub event - 4

The screenshot shows a browser window with multiple tabs open. The active tab is 'Webhook Deliveries' for a GitHub repository named 'huihui0310tt/LearnWebhookTest'. The page displays a list of webhook deliveries. One delivery is highlighted with a red box:

Event ID	Action	Timestamp
544b87e0-4e02-11ec-9055-0e9581be86d1	redelivery	2021-11-25 23:18:21
544b87e0-4e02-11ec-9055-0e9581be86d1		2021-11-25 23:14:06
7a75d840-4e01-11ec-808b-8f21c8e823ad		2021-11-25 23:08:01

The left sidebar shows the GitHub settings menu with 'Webhooks' selected. The right side of the page shows the webhook details, including headers and body content.

GitHub Settings Sidebar (Left):

- Options
- Manage access
- Security & analysis
- Webhooks** (highlighted)
- Notifications
- Integrations
- Deploy keys
- Actions
- Environments
- Secrets
- Pages
- Moderation settings

Webhooks / Manage webhook (Right):

Recent Deliveries

Event ID	Action	Timestamp
544b87e0-4e02-11ec-9055-0e9581be86d1	redelivery	2021-11-25 23:18:21
544b87e0-4e02-11ec-9055-0e9581be86d1		2021-11-25 23:14:06
7a75d840-4e01-11ec-808b-8f21c8e823ad		2021-11-25 23:08:01

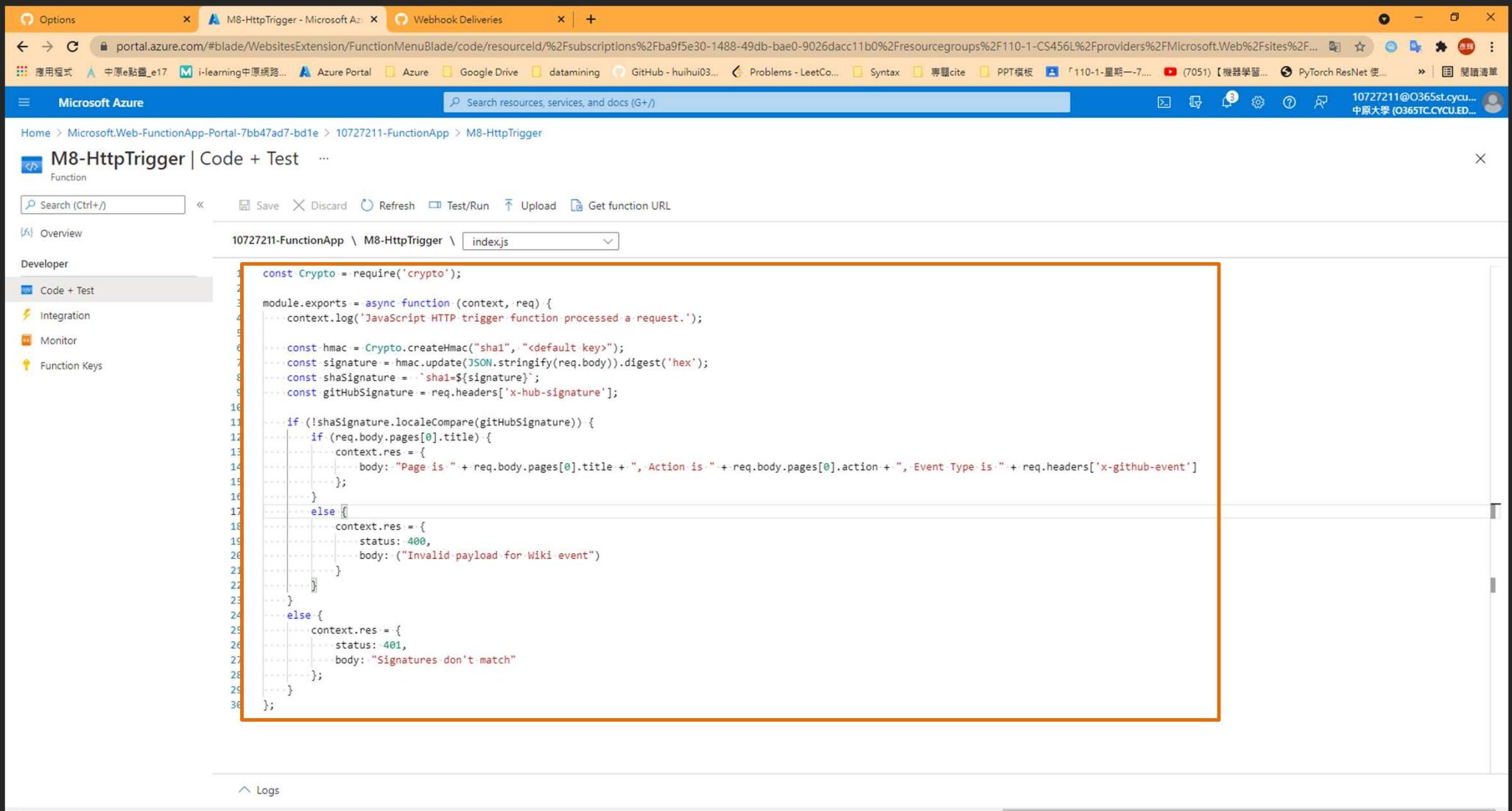
Request Response 200 Redeliver Headers Body

Content-Type: text/plain; charset=utf-8  
Date: Thu, 25 Nov 2021 15:18:21 GMT

Page is Home, Action is edited, Event Type is gollum

© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

# Get a key for your Azure Function



The screenshot shows the Microsoft Azure portal interface for managing Azure Functions. The current function is "M8-HttpTrigger". The developer tools sidebar is open, with "Code + Test" selected. The main area displays the "index.js" file content, which contains Node.js code for handling GitHub webhook requests. A red box highlights the entire code block.

```
const Crypto = require('crypto');

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

    const hmac = Crypto.createHmac("sha1", "<default key>");
    const signature = hmac.update(JSON.stringify(req.body)).digest('hex');
    const shaSignature = `sha1=${signature}`;
    const gitHubSignature = req.headers['x-hub-signature'];

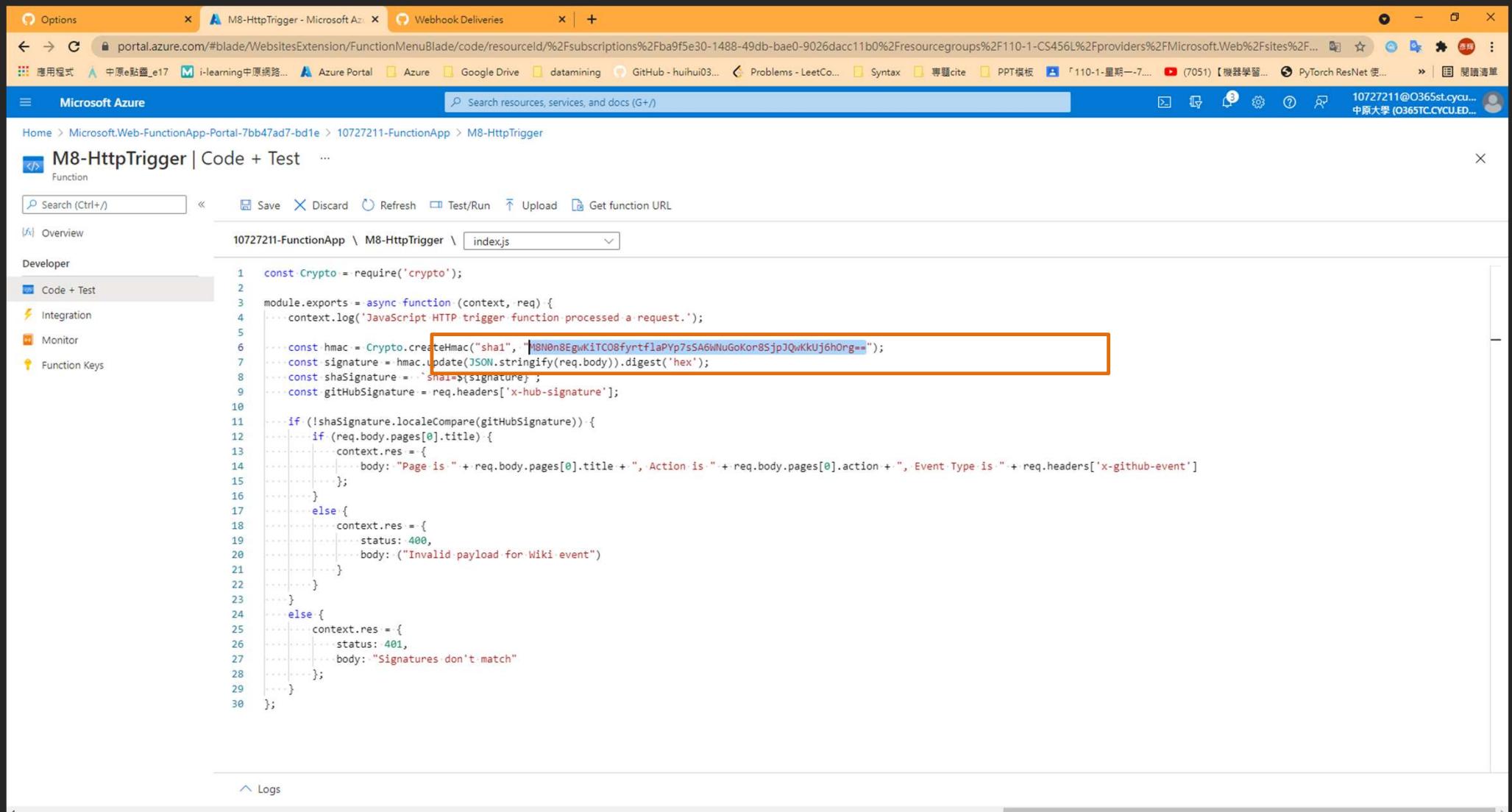
    if (!shaSignature.localeCompare(gitHubSignature)) {
        if (req.body.pages[0].title) {
            context.res = {
                body: "Page is " + req.body.pages[0].title + ", Action is " + req.body.pages[0].action + ", Event Type is " + req.headers['x-github-event']
            };
        } else {
            context.res = {
                status: 400,
                body: ("Invalid payload for Wiki event")
            };
        }
    } else {
        context.res = {
            status: 401,
            body: "Signatures don't match"
        };
    }
};
```

# Update the webhook secret - 1

The screenshot shows the Microsoft Azure portal interface. The URL in the address bar is `portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/functionKeys/resourceId/%2Fsubscriptions%2Fba9f5e30-1488-49db-bae0-9026dacc11b0%2Fresourcegroups%2F110-1-CS456L%2Fproviders%2FMicrosoft.Web%2Fs...`. The title bar says "M8-HttpTrigger - Microsoft Azure". The main content area is titled "M8-HttpTrigger | Function Keys" and shows the "Function Keys" section. The "Function Keys" table has one row with "Name" as "default" and "Value" as "M8N0n8EgwKiTCO8fyrtflaPYp7sSA6WNuGoKor8SjpJQwKkJUj6hOrg==". To the right of the "Value" column, there is a "Renew key value" button, which is highlighted with a red box.

Name	Value	
default	M8N0n8EgwKiTCO8fyrtflaPYp7sSA6WNuGoKor8SjpJQwKkJUj6hOrg==	

# Update the webhook secret - 2



The screenshot shows the Microsoft Azure portal interface for managing an Azure Function. The function name is "M8-HttpTrigger". The code editor displays the "index.js" file, which contains the following JavaScript code:

```
1 const Crypto = require('crypto');
2
3 module.exports = async function (context, req) {
4     context.log('JavaScript HTTP trigger function processed a request.');
5
6     const hmac = Crypto.createHmac("sha1", "M8N0n8EgwKiTC08fyrtflaPyg7sSA6WNuGoKor8SjpJQwKkUj6h0rg==");
7     const signature = hmac.update(JSON.stringify(req.body)).digest('hex');
8     const shaSignature = `sha1=${signature}`;
9     const gitHubSignature = req.headers['x-hub-signature'];
10
11     if (!shaSignature.localeCompare(gitHubSignature)) {
12         if (req.body.pages[0].title) {
13             context.res = {
14                 body: "Page is " + req.body.pages[0].title + ", Action is " + req.body.pages[0].action + ", Event Type is " + req.headers['x-github-event']
15             };
16         } else {
17             context.res = {
18                 status: 400,
19                 body: ("Invalid payload for Wiki event")
20             };
21         }
22     } else {
23         context.res = {
24             status: 401,
25             body: "Signatures don't match"
26         };
27     };
28 };
29
30};
```

The line of code that generates the HMAC signature is highlighted with an orange rectangle:

```
const signature = hmac.update(JSON.stringify(req.body)).digest('hex');
```

# Test the webhook and the Azure Function - 1

The screenshot shows the GitHub 'Webhooks / Manage webhook' settings page for a repository named 'LearnWebhookTest'. The left sidebar menu is visible, showing various options like Options, Manage access, Security & analysis, and Webhooks (which is currently selected). The main area displays the configuration for a webhook. A red box highlights the 'Secret' input field, which contains the value '3N0n8EqwKiTCO8fyrtflaPYp7sSA6WNuGoKor8SjpJQwKkUj6hOrg=='. A large green button with the text 'Update Webhook' is overlaid on the right side of the page. The URL for the payload is set to 'https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger'. The content type is set to 'application/json'. SSL verification is enabled. The events section has the 'Let me select individual events' option selected.

github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597

Search or jump to... Pull requests Issues Marketplace Explore

Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Webhooks / Manage webhook

Settings Recent Deliveries

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

Payload URL \*

https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger

Content type

application/json

Secret

3N0n8EqwKiTCO8fyrtflaPYp7sSA6WNuGoKor8SjpJQwKkUj6hOrg==

SSL verification

By default, we verify SSL certificates when delivering payloads.

Enable SSL verification  Disable (not recommended)

Which events would you like to trigger this webhook?

Just the push event.  
 Send me everything.  
 Let me select individual events.

Update Webhook

# Test the webhook and the Azure Function - 2

The screenshot shows a browser window with multiple tabs open. The active tab is 'Webhook Deliveries' on the GitHub page for the repository 'huihui0310tt/LearnWebhookTest'. The URL in the address bar is [github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries](https://github.com/huihui0310tt/LearnWebhookTest/settings/hooks/330221597/deliveries). The GitHub sidebar on the left has 'Webhooks' selected. The main content area displays a list of webhook deliveries. One delivery is highlighted with a red box around its status icon and ID: '544b87e0-4e02-11ec-9055-0e9581be86d1'. Below this, under the 'Request' section, there is a green button labeled 'Redeliver' with a circular arrow icon, which is also highlighted with an orange box.

**Webhooks / Manage webhook**

Settings Recent Deliveries

Request Response 200 Redeliver (completed in 0.59 seconds)

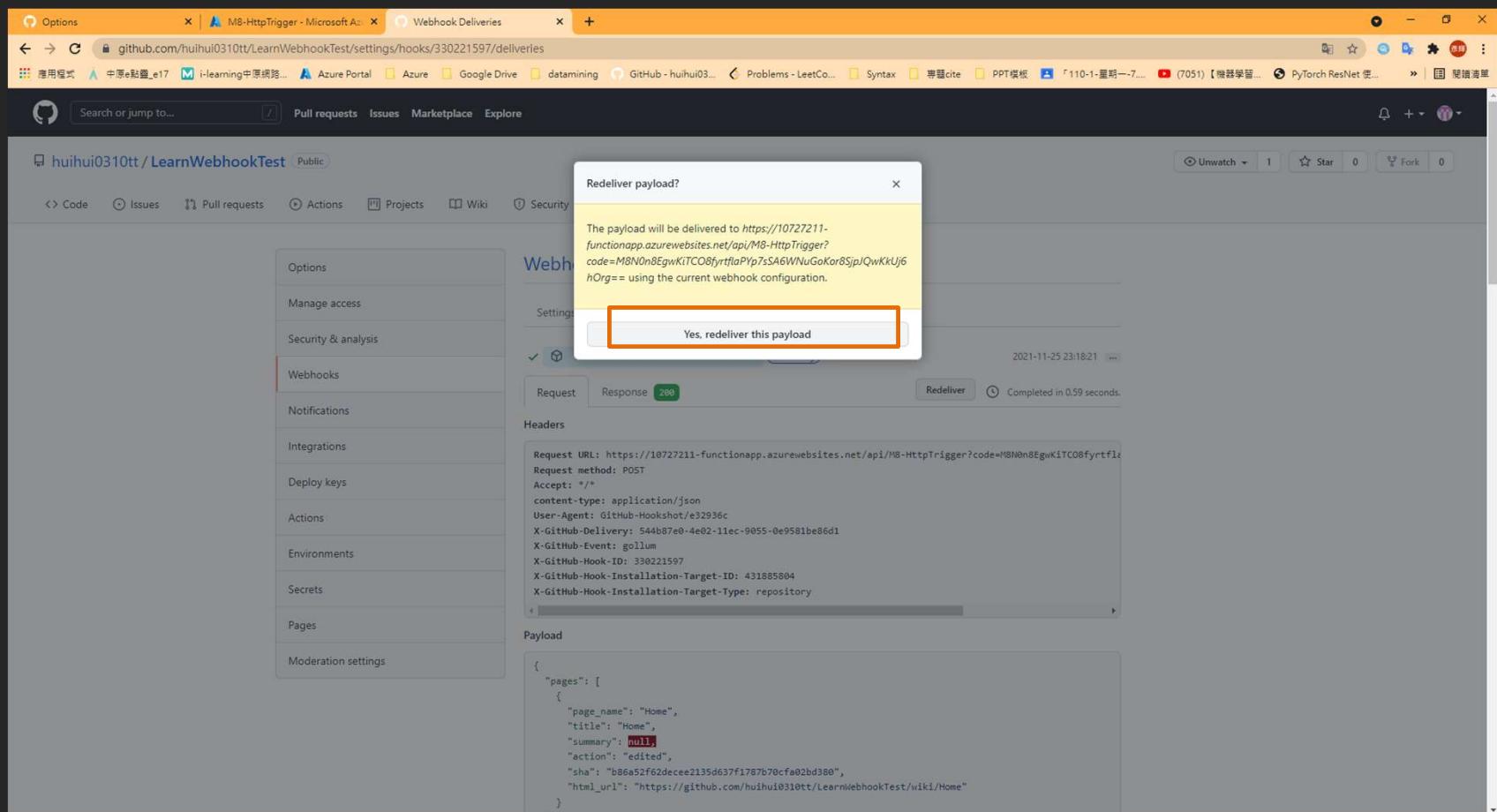
Headers

```
Request URL: https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKitC08fyrtfl&...  
Request method: POST  
Accept: */*  
Content-Type: application/json  
User-Agent: GitHub-Hookshot/e32936c  
X-GitHub-Delivery: 544b87e0-4e02-11ec-9055-0e9581be86d1  
X-GitHub-Event: gollum  
X-GitHub-Hook-ID: 330221597  
X-GitHub-Hook-Installation-Target-ID: 431885804  
X-GitHub-Hook-Installation-Target-Type: repository
```

Payload

```
{  
  "pages": [  
    {  
      "page_name": "Home",  
      "title": "Home",  
      "summary": null,  
      "action": "edited",  
      "sha": "b86a52f62dece2135d637f1787b70cfa02bd380",  
      "html_url": "https://github.com/huihui0310tt/LearnWebhookTest/wiki/Home"  
    }  
  ]}
```

# Test the webhook and the Azure Function - 3



# Test the webhook and the Azure Function - 4

Webhooks / Manage webhook

Recent Deliveries

544b87e0-4e02-11ec-9055-0e9581be86d1 (redelivery) 2021-11-25 23:23:52 ...

Request Response 200 Redeliver Completed in 0.55 seconds.

Headers

```
Content-Type: text/plain; charset=utf-8
Date: Thu, 25 Nov 2021 15:23:51 GMT
```

Body

```
Page is Home, Action is edited, Event Type is gollum
```

544b87e0-4e02-11ec-9055-0e9581be86d1 (redelivery) 2021-11-25 23:22:26 ...

544b87e0-4e02-11ec-9055-0e9581be86d1 (redelivery) 2021-11-25 23:18:21 ...

544b87e0-4e02-11ec-9055-0e9581be86d1 2021-11-25 23:14:06 ...

7a75d840-4e01-11ec-808b-8f21c8e823ad 2021-11-25 23:08:01 ...

Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

Webhooks / Manage webhook

Recent Deliveries

544b87e0-4e02-11ec-9055-0e9581be86d1 (redelivery) 2021-11-25 23:23:52 ...

Request Response 200 Redeliver Completed in 0.55 seconds.

Headers

```
Request URL: https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKitC08fyrtfl&
Request method: POST
Accept: */*
content-type: application/json
User-Agent: GitHub-Hookshot/e32936c
X-GitHub-Delivery: 544b87e0-4e02-11ec-9055-0e9581be86d1
X-GitHub-Event: gollum
X-GitHub-Hook-ID: 330221597
X-GitHub-Hook-Installation-Target-ID: 431885804
X-GitHub-Hook-Installation-Target-Type: repository
X-Hub-Signature: sha1=9c1a48c1b14eaf14d26b2974f1b36117d0c7f4dd
X-Hub-Signature-256: sha256=53fe862b052157ea105a2333a63a6bc4f3320ec0905fd920e0aea6b56935bbb3
```

Payload

```
{
  "pages": [
    {
      "page_name": "Home",
      "title": "Home",
      "summary": null,
      "action": "edited",
      "sha": "b86a52f62dece2135d637f1787b70cfa02bd380",
      "html_url": "https://github.com/huihui0310tt/LearnWebhookTest/wiki/Home"
    }
  ]
}
```

# Test an invalid signature - 1

随便輸入一串字

Update webhook

Webhooks / Manage webhook

Payload URL \*

https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger

Content type

application/json

Secret

If you've lost or forgotten this secret, you can change it, but be aware that any integrations using this secret will need to be updated. — [Change Secret](#)

SSL verification

By default, we verify SSL certificates when delivering payloads.

Enable SSL verification    Disable (not recommended)

Which events would you like to trigger this webhook?

Just the push event.

Send me everything.

Unwatch 1 Star 0 Fork 0

# Test an invalid signature - 2

The screenshot shows a GitHub repository page for "huihui0310tt/LearnWebhookTest". The user is interacting with a webhook configuration. A modal dialog titled "Redeliver payload?" is open, containing the message: "The payload will be delivered to <https://10727211-functionapp.azurewebsites.net/api/M8-HttpTrigger?code=M8N0n8EgwKiTCO8fyrtflaPyP7sSA6WNuGoKor8SjpQwKkJ6hOrg==> using the current webhook configuration." Below this message is a button labeled "Yes, redeliver this payload", which is highlighted with a red rectangle.

On the right side of the screen, the "Webhooks / Manage webhook" section is visible. It shows a list of recent deliveries. One delivery is highlighted with a yellow border and a red rectangle, indicating an error. The details for this delivery show a status of "401" and the message "Signatures don't match". Other deliveries in the list are marked with green checkmarks and "redelivery" status indicators.

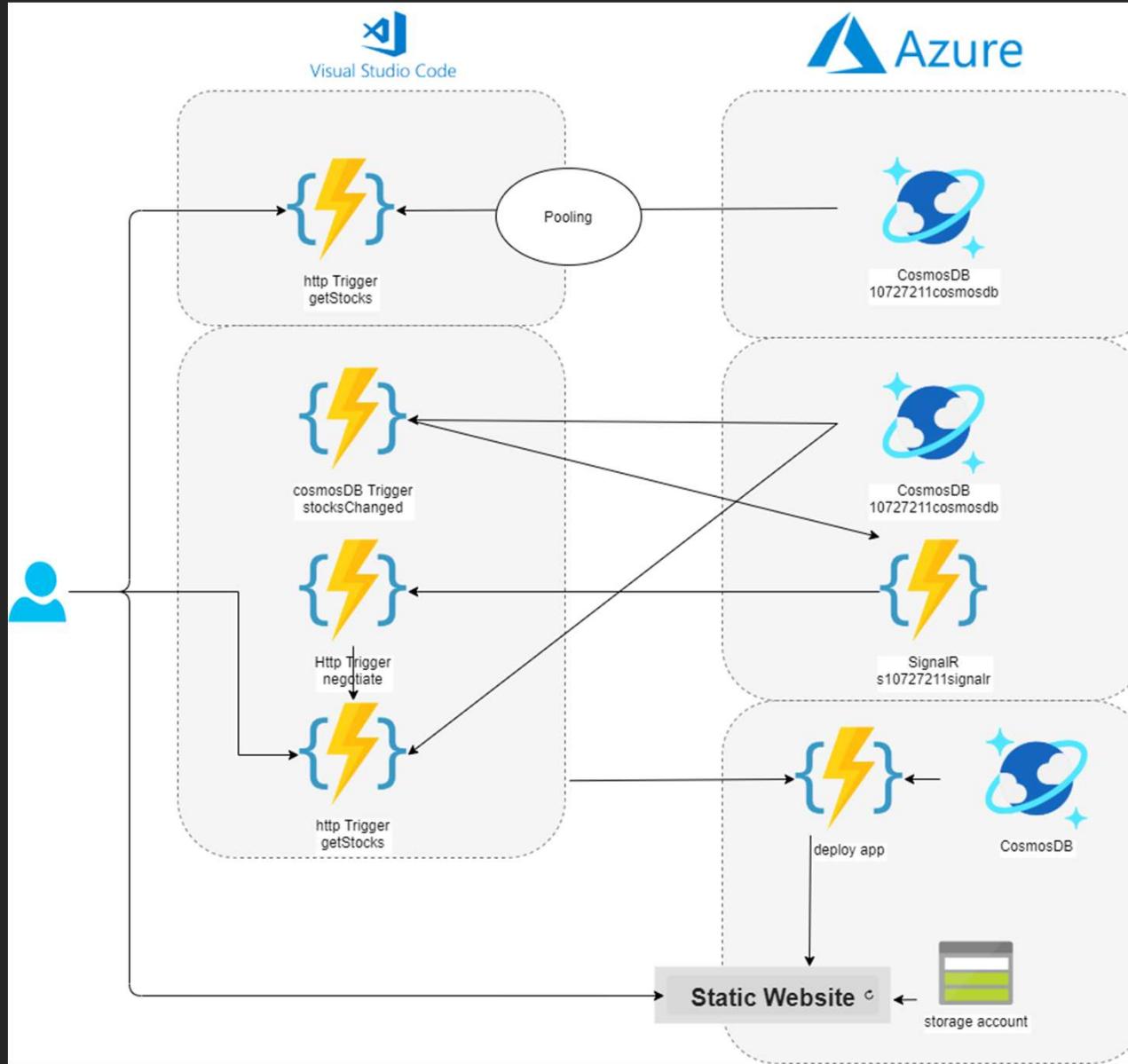
Key elements visible in the interface include:

- GitHub navigation bar: Options, M8-HttpTrigger - Microsoft Azure, Webhook Deliveries.
- Repository header: huihui0310tt / LearnWebhookTest (Public).
- Left sidebar: Options, Manage access, Security & analysis, Webhooks (selected), Notifications, Integrations, Deploy keys, Actions, Environments, Secrets, Pages, Moderation settings.
- Central area: Redeliver payload? dialog, Request and Response tabs, Headers, Payload (JSON object).
- Right area: Webhooks / Manage webhook, Settings tab, Recent Deliveries table.

09

// Enable automatic updates in a web  
application using Azure Functions  
and SignalR Service

# Abstract



# Create a Storage account and Azure Cosmos DB account - 1

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for 'Exercise - Analyze the limitation' and '110-1-CS456L - Microsoft Azure'. The URL in the address bar is [portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/overview](https://portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/overview). The main content area displays the '110-1-CS456L' resource group overview. The 'Essentials' section shows the subscription details: 'Subscription (Move) : 中原大學', 'Subscription ID : ba9f5e30-1488-49db-bae0-9026dacc11b0', 'Tags (Edit) : Click here to add tags', 'Deployments : 3 Failed, 79 Succeeded', and 'Location : Southeast Asia'. Below this, the 'Resources' section lists 'Recommendations (4)' with a filter bar for 'Type == all' and 'Location == all'. At the bottom, a terminal window titled 'Bash' shows the command 'azurerm user@Azure:~\$' followed by a prompt. The status bar at the bottom indicates 'Requesting a Cloud Shell. Succeeded.' and 'Connecting terminal...'. The overall theme is a light blue and white color scheme.

## Create a Storage account and Azure Cosmos DB account - 2

```
export STORAGE_ACCOUNT_NAME=10727211model9storage  
echo "Storage Account Name: $STORAGE_ACCOUNT_NAME"
```

```
az storage account create \  
--name $STORAGE_ACCOUNT_NAME \  
--resource-group 110-1-CS456L \  
--kind StorageV2 \  
--sku Standard_LRS
```

```
az cosmosdb create \  
--name 10727211cosmosdb \  
--resource-group 110-1-CS456L
```

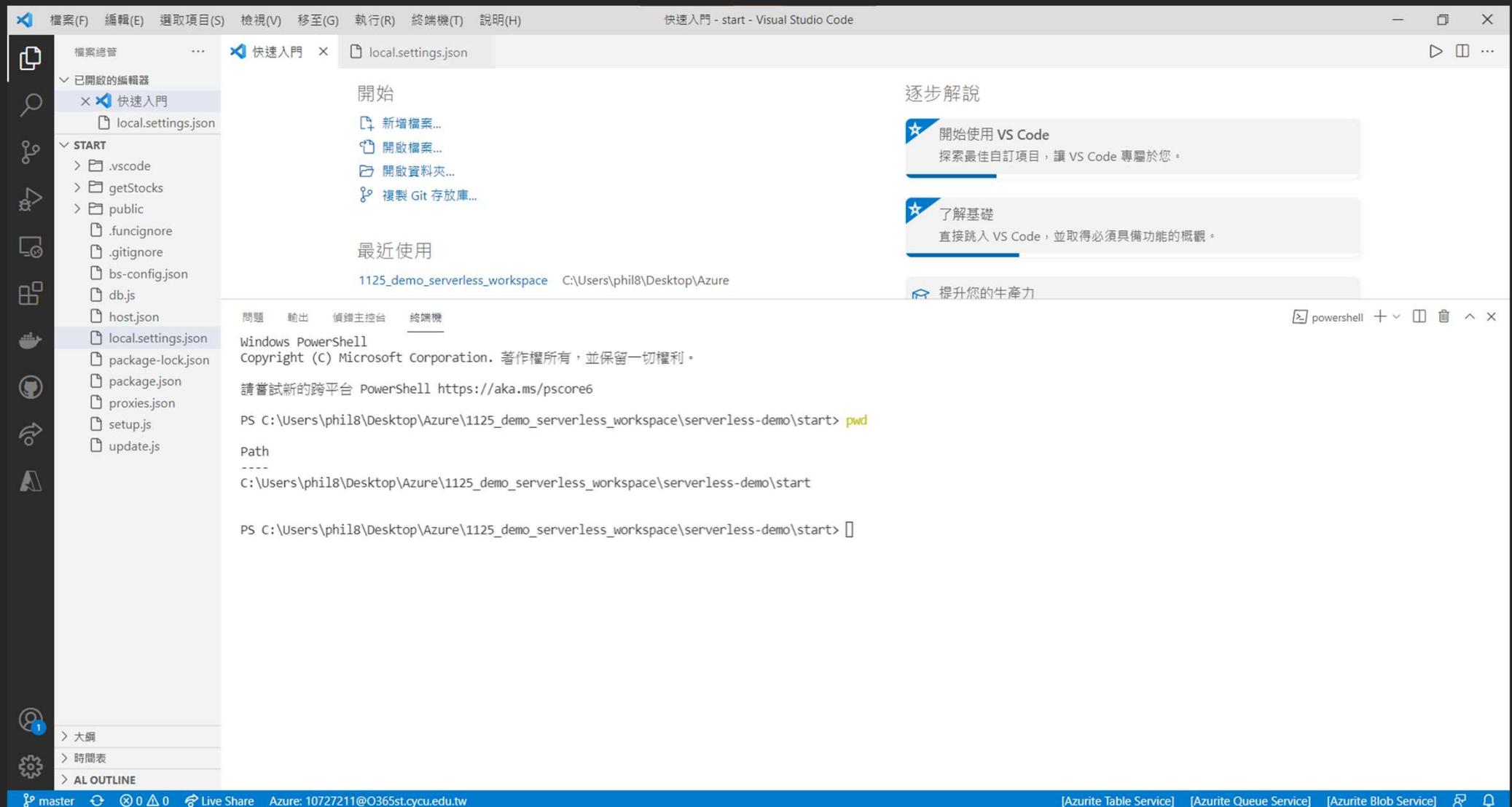
# Download sample app code & switch directory

The screenshot shows the Visual Studio Code interface with a terminal window open. The terminal is running a Windows PowerShell session. The user has cloned a GitHub repository named 'serverless-demo' into their current directory and then switched to it.

```
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace> git clone https://github.com/MicrosoftDocs/mslearn-advocates.azure-functions-and-signalr.git serverless-demo
Cloning into 'serverless-demo'...
remote: Enumerating objects: 151, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (32/32), done.
Receiving objects: 87% (132/151) 0 (delta 0), pack-reused 119  eceiving objects: 86% (130/151)
Receiving objects: 100% (151/151), 97.79 KiB | 370.00 KiB/s, done.
Resolving deltas: 100% (83/83), done.
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace> code .\serverless-demo\start\
```

The interface includes a sidebar with file management, a 'Recent' section, and a 'Help' icon. The main area features a 'Quick Start' guide and productivity tips. The bottom status bar shows the current branch ('master'), commit count ('0'), and other system information.

# Check path



# Update local settings - 1

The screenshot shows the Microsoft Azure portal interface. At the top, there are two tabs: "Exercise - Analyze the limitation" and "110-1-CS456L - Microsoft Azure". The main content area displays the "Overview" of the resource group "110-1-CS456L". The "Essentials" section provides summary information:

- Subscription (Move) : 中原大學
- Subscription ID : ba9f5e30-1488-49db-bae0-9026dacc11b0
- Tags (Edit) : Click here to add tags
- Deployments : 3 Failed, 79 Succeeded
- Location : Southeast Asia

The "Resources" section shows recommendations (4). Below the main content is a terminal window titled "Bash" with the following output:

```
Bash | ⌂ ? ⓘ ⌂ {} ⌂ ⌂
Requesting a Cloud Shell. Succeeded.
Connecting terminal...
azureuser@Azure:~$
```

# Update local settings - 2

```
STORAGE_CONNECTION_STRING=$(az storage account show-connection-string \  
--name $(az storage account list \  
--resource-group 110-1-CS456L \  
--query [0].name -o tsv) \  
--resource-group 110-1-CS456L \  
--query "connectionString" -o tsv)
```

```
COSMOSDB_ACCOUNT_NAME=$(az cosmosdb list \  
--resource-group 110-1-CS456L \  
--query [0].name -o tsv)
```

```
COSMOSDB_CONNECTION_STRING=$(az cosmosdb list-connection-strings \  
--name $COSMOSDB_ACCOUNT_NAME \  
--resource-group 110-1-CS456L \  
--query "connectionStrings[?description=='Primary SQL Connection String'].connectionString" -o tsv)
```

```
COSMOSDB_MASTER_KEY=$(az cosmosdb list-keys \  
--name $COSMOSDB_ACCOUNT_NAME \  
--resource-group 110-1-CS456L \  
--query primaryMasterKey -o tsv)
```

```
printf "\n\nReplace <STORAGE_CONNECTION_STRING> with:\n$STORAGE_CONNECTION_STRING\n\nReplace <COSMOSDB_CONNECTION_STRING>  
with:\n$COSMOSDB_CONNECTION_STRING\n\nReplace <COSMOSDB_MASTER_KEY> with:\n$COSMOSDB_MASTER_KEY\n\n"
```

# Update local settings - 3

The screenshot shows a Microsoft Azure Bash terminal window. The user is executing commands to retrieve connection strings and master keys from an Azure Cosmos DB account and then replace placeholder variables in a local configuration file.

```
Bash
> --resource-group 110-1-CS456L \
> --query [0].name -o tsv
azureuser@Azure:~$ 
azureuser@Azure:~$ COSMOSDB_CONNECTION_STRING=$(az cosmosdb list-connection-strings \
> --name $COSMOSDB_ACCOUNT_NAME \
> --resource-group 110-1-CS456L \
> --query "connectionStrings[?description=='Primary SQL Connection String'].connectionString" -o tsv)
WARNING: This command has been deprecated and will be removed in a future release. Use 'cosmosdb keys list --type connection-strings' instead.
azureuser@Azure:~$ 
azureuser@Azure:~$ COSMOSDB_MASTER_KEY=$(az cosmosdb list-keys \
> --name $COSMOSDB_ACCOUNT_NAME \
> --resource-group 110-1-CS456L \
> --query primaryMasterKey -o tsv)
WARNING: This command has been deprecated and will be removed in a future release. Use 'cosmosdb keys list' instead.
azureuser@Azure:~$ 
azureuser@Azure:~$ printf "\n\nReplace <STORAGE_CONNECTION_STRING> with:\n$STORAGE_CONNECTION_STRING\n\nReplace <COSMOSDB_CONNECTION_STRING> with:\n$COSMOSDB_CONNECTION_STRING\n\nReplace <COSMOSDB_MASTER_KEY> with:\n$COSMOSDB_MASTER_KEY\n\n"
Replace <STORAGE_CONNECTION_STRING> with:
DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=10727211model9storage;AccountKey=WqQEWC/HVsikytntzzkt7b6egjMevS0rrm5Pb034CPJXta1wCU2QmLRS7Qmm1isFiQriP45idMpY7EEPU4OpQ==

Replace <COSMOSDB_CONNECTION_STRING> with:
AccountEndpoint=https://10727211cosmosdb.documents.azure.com:443/;AccountKey=v1S7EXnw6xCg0HzIYqTLuIpPTY7g6bihim2K7uOhKwIwN99bF42wQJECs9Pt9rJ0LHD2e2bB2E60hzdiQUhoHQ==;

Replace <COSMOSDB_MASTER_KEY> with:
v1S7EXnw6xCg0HzIYqTLuIpPTY7g6bihim2K7uOhKwIwN99bF42wQJECs9Pt9rJ0LHD2e2bB2E60hzdiQUhoHQ==

azureuser@Azure:~$
```

# Update local settings - 4

The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** 檔案(F) 編輯(E) 選取項目(S) 檢視(V) 移至(G) 執行(R) 終端機(T) 說明(H)
- Editor:** local.settings.json - start - Visual Studio Code
- Left Sidebar:** 檔案總管, 入門, 已開啟的編輯器 (local.settings.json), START (node\_modules/public/favicon.ico, index.html, index.html.css, index.html.js), stocksChang..., .funcignore, .gitignore, bs-config.json, db.js, host.json, local.setting..., package-lock.json, package.json, proxies.json, setup.js, update.js.
- Code Editor Content:**

```
1  {
2      "IsEncrypted": false,
3      "Values": {
4          "AzureWebJobsStorage": "DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=10727211model9storage;AccountKey=WqQEWC/Hvsik",
5          "AzureCosmosDBConnectionString": "AccountEndpoint=https://10727211cosmosdb.documents.azure.com:443/;AccountKey=v1S7EXnw6xCg0HzIYqTLuIpTTY7g6bihi",
6          "AzureCosmosDBMasterKey": "v1S7EXnw6xCg0HzIYqTLuIpTTY7g6bihiM2K7u0hKwIwN99bF42wQJECs9Pt9rJ0LHD2e2bB2E60hzdiQuhoHQ==",
7          "AzureSignalRConnectionString": "<SIGNALR_CONNECTION_STRING>",
8          "FUNCTIONS_WORKER_RUNTIME": "node"
9      },
10     "Host" : {
11         "LocalHttpPort": 7071,
12         "CORS": "http://localhost:8080",
13         "CORScredentials": true
14     }
15 }
```
- Terminal:** powershell
- Bottom Status Bar:** master\*, Go 1.17.3, Live Share, Azure: 10727211@O365st.cycu.edu.tw, [Azurite Table Service], [Azurite Queue Service], [Azurite Blob Service], 第 16 行, 第 1 欄, 空格: 2, CRLF, JSON

# Run the application - 1

The screenshot shows a Visual Studio Code interface. On the left is the sidebar with icons for file management, search, and code navigation. The main area has a tab bar with "local.settings.json" and "start - Visual Studio Code". The "local.settings.json" tab is active, displaying a JSON configuration file with various Azure connection strings and settings. Below it is a terminal window titled "Windows PowerShell" showing the command-line output of running npm scripts to set up the application.

```
1  {
2    "IsEncrypted": false,
3    "Values": {
4      "AzureWebJobsStorage": "DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=10727211model9storage;AccountKey=kDLWkdjioQZM",
5      "AzureCosmosDBConnectionString": "AccountEndpoint=https://aci-cosmos-db-26309.documents.azure.com:443/;AccountKey=e92z5ydiZNoHPkG72i5zd3fLov39ga",
6      "AzureCosmosDBMasterKey": "e92z5ydiZNoHPkG72i5zd3fLov39gaVawZqottc1HNZd7tEyUtIGI4DdqJQAtAt0kNC61tkd55HAPCjnghp10Q==",
7      "AzureSignalRConnectionString": "<SIGNALR_CONNECTION_STRING>",
8      "FUNCTIONS_WORKER_RUNTIME": "node"
9    },
10   "Host" : {
```

問題 輸出 備錯主控台 終端機

```
Windows PowerShell
Copyright (C) Microsoft Corporation. 著作權所有，並保留一切權利。

請嘗試新的跨平台 PowerShell https://aka.ms/pscore6

PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start> npm -v
6.14.15
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start> node -v
v14.18.1
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start> npm install

> microsoft-learn-func-signalr-start@1.0.0 postinstall C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start
> npm run setup

> microsoft-learn-func-signalr-start@1.0.0 setup C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start
> node setup.js

Database created.
Collection created.
Seed data added.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

added 336 packages from 282 contributors and audited 404 packages in 9.84s
found 48 vulnerabilities (13 moderate, 34 high, 1 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start>
```

master\* 0 △0 Live Share Azure: 10727211@O365st.cyu.edu.tw [Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] 第 16 行, 第 1 欄 空格: 2 UTF-8 CRLF JSON

# Run the application - 2

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the `local.settings.json` file.
- Terminal:** Displays the command `func host start` and its output:
  - Azure Functions Core Tools
  - Core Tools Version: 3.0.3904 Commit hash: c345f7140a8f968c5dbc621f8a8374d8e3234206 (64-bit)
  - Function Runtime Version: 3.3.1.0
  - Functions:**
  - `getStocks: [GET] http://localhost:7071/api/getStocks`
  - For detailed output, run func with --verbose flag.
  - [2021-11-25T16:21:08.816Z] Debugger listening on ws://127.0.0.1:9229/dd7dd690-fadd-4f4d-a41e-df3c413fbafb
  - [2021-11-25T16:21:08.817Z] For help, see: <https://nodejs.org/en/docs/inspector>
  - info:** Microsoft.AspNetCore.Hosting.Diagnostics[1]
  - Request starting HTTP/2 POST http://127.0.0.1:53893/AzureFunctionsRpcMessages.FunctionRpc/EventStream application/grpc
  - info:** Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
  - Executing endpoint 'gRPC - /AzureFunctionsRpcMessages.FunctionRpc/EventStream'
  - [2021-11-25T16:21:09.017Z] Worker process started and initialized.
  - [2021-11-25T16:21:09.147Z] Debugger attached.
  - [2021-11-25T16:21:14.539Z] Host lock lease acquired by instance ID '0000000000000000000000002784BC8D'.

# Run the application - 3

S 檔案(F) 編輯(E) 選取項目(S) 檢視(V) 移至(G) 執行(R) 終端機(T) 說明(H) local.settings.json - start - Visual Studio Code

Attach to Node Functions (start) 快速入門 local.settings.json M X

變數

問題 輸出 偵錯主控台 終端機

PS C:\Users\phil8\Desktop\Azure\1125\_demo\_serverless\_workspace\serverless-demo\start> npm start

> microsoft-learn-func-signalr-start@1.0.0 start C:\Users\phil8\Desktop\Azure\1125\_demo\_serverless\_workspace\serverless-demo\start

> lite-server --baseDir="public"

\*\* browser-sync config \*\*

```
{  
  injectChanges: false,  
  files: ['./public/**/*.{html,htm,css,js}'],  
  watchOptions: { ignored: 'node_modules' },  
  server: {  
    baseDir: './public',  
    middleware: [ [Function (anonymous)], [Function (anonymous)] ]  
  },  
  port: 8080  
}  
[Browsersync] Access URLs:  
-----  
  Local: http://localhost:8080  
  External: http://192.168.56.1:8080  
-----  
  UI: http://localhost:3001  
  UI External: http://localhost:3001  
-----  
[Browsersync] Serving files from: ./public  
[Browsersync] Watching files...  
21.11.26 00:21:50 200 GET /index.html  
21.11.26 00:21:50 200 GET /index.html.css  
21.11.26 00:21:50 200 GET /index.html.js
```

powershell host start... powershell

Stocks | Enable automatic update X +  
localhost:8080 應用程式 中原e點靈\_e17 i-learning中原網路... Azure Portal 閱讀清單

Stocks

ABC: \$100.00  
CHANGE: +1.00

DEF: \$45.89  
CHANGE: -1.25

GHI: \$156.21  
CHANGE: +6.81

master\* 0 Δ 0 Attach to Node Functions (start) Live Share Azure: 10727211@O365st.cycu.edu.tw [Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] 第 16 行, 第 1 欄 空格: 2 UTF-8 CRLF JSON

# Run the application - 4

S 檔案(F) 編輯(E) 選取項目(S) 檢視(V) 移至(G) 執行(R) 終端機(T) 說明(H) local.settings.json - start - Visual Studio Code

問題 輸出 儲錯主控台 終端機

變數

```
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start> npm run update-data
> microsoft-learn-func-signalr-start@1.0.0 update-data C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start
> node update.js

Read data from database.

Data updated: {"id":"e0eb6e85-176d-4ce6-89ae-1f699aaa0bab","symbol":"ABC","price":90.52,"change":9.48,"changeDirection":"-","_rid":"dYUaALKsFUoBAAAAAAA==","_self":"dbs/dYUaAA==/colls/dYUaALKsFUo=/docs/dYUaALKsFUoBAAAAAAA==/","_etag":"\"c7008e82-0000-1800-0000-619fb7d90000\"","_attachments":"attachments/","_ts":1637857241}
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start> npm start
> microsoft-learn-func-signalr-start@1.0.0 start C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo\start
> lite-server --baseDir="public"

** browser-sync config **
{
  injectChanges: false,
  files: [ './public/**/*.{html,htm,css,js}' ],
  watchOptions: { ignored: 'node_modules' },
  server: {
    baseDir: './public',
    middleware: [ [Function (anonymous)], [Function (anonymous)] ]
  },
  port: 8080
}
[Browsersync] Access URLs:
-----
Local: http://localhost:8080
External: http://192.168.56.1:8080
-----
UI: http://localhost:3001
UI External: http://localhost:3001
[Browsersync] Serving files from: ./public
[Browsersync] Watching files...
21.11.26 00:24:07 304 GET /index.html
21.11.26 00:24:07 304 GET /index.html.css
21.11.26 00:24:07 304 GET /index.html.js
```

host sta... ✓  
powershell

監看

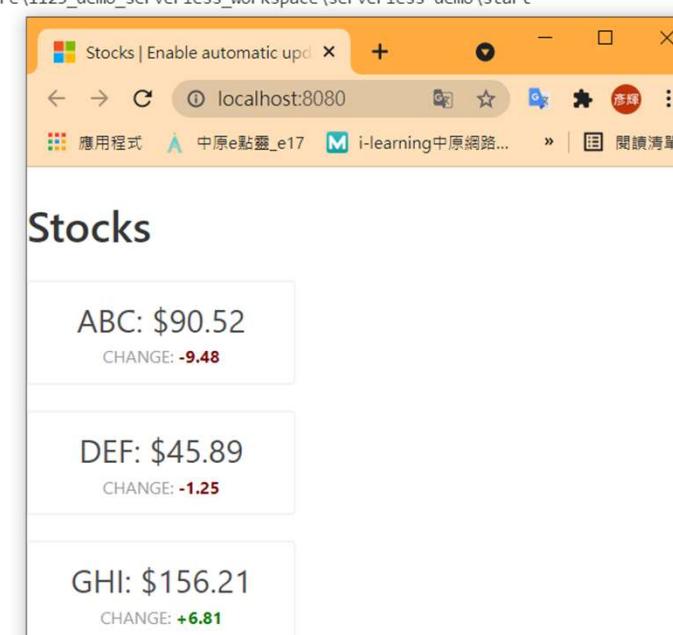
呼叫堆疊 Attach ... 正在執行

已載入的指令碼

中斷點  
 Caught Exceptions  
 Uncaught Except...

Attach to Node Functions (start) Live Share Azure: 10727211@O365st.cycu.edu.tw

Azure Table Service | Azure Queue Service | Azure Blob Service | 第 16 行, 第 1 欄 | 全格: 2 | UTF-8 | CRLF | JSON | 🔍 | 🗑



The screenshot shows a Visual Studio Code interface with a terminal window and a browser window. The terminal window displays the execution of a Node.js application, including npm commands and logs from Browsersync and lite-server. The browser window shows a simple 'Stocks' application with three stock entries: ABC (\$90.52, change -9.48), DEF (\$45.89, change -1.25), and GHI (\$156.21, change +6.81).

# Run the application - 5

When you're done, stop the running processes.

- To stop the web server, select the **kill process** (trash can icon) on the terminal window that is running the web server.
- To stop the functions app, select **Stop** or press **Shift+F5**.

The screenshot shows the Visual Studio Code interface with the following details:

- Terminal Tab:** The terminal tab is active, displaying the command `microsoft-learn-func-signalr-start@1.0.0 start C:\Users\phil8\Desktop\Azure\1125\_demo\_serverless\_workspace\serverless-demo\start` followed by `lite-server --baseDir="public"`. Below this, a configuration block for `browser-sync` is shown, including `injectChanges: false`, `files: ['./public/\*\*/\*.{html,htm,css,js}' ]`, `watchOptions: { ignored: 'node\_modules' }`, and `server: { baseDir: './public' }`.
- Status Bar:** The status bar at the bottom shows the current branch as `master\*`, file status icons, and connection information: `Live Share Azure: 10727211@O365st.cycu.edu.tw`.
- Sidebar:** The sidebar on the left includes sections for `已載入的指令碼` (Loaded Commands), `中斷點` (Breakpoints), and checkboxes for `Caught Exceptions` and `Uncaught Except...`.
- Right Sidebar:** The right sidebar contains icons for `powershell` and other integrated development environment features.

# Create a SignalR account - 1

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for 'Exercise - Analyze the limitation' and '110-1-CS456L - Microsoft Azure'. The URL in the address bar is [portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/overview](https://portal.azure.com/#@O365tc.cycu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/overview). The page title is '110-1-CS456L - Microsoft Azure'.

The left sidebar contains a navigation menu with items like 'Home', 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Resource visualizer', 'Events', 'Settings', and 'Deployments'. The 'Overview' tab is selected.

The main content area displays the 'Essentials' section for the '110-1-CS456L' resource group. It shows the following details:

- Subscription (Move) : 中原大學
- Subscription ID : ba9f5e30-1488-49db-bae0-9026dacc11b0
- Tags (Edit) : Click here to add tags
- Deployments : 3 Failed, 79 Succeeded
- Location : Southeast Asia

Below the essentials section, there is a 'Resources' section with a count of 4 recommendations. A terminal window at the bottom shows the output of a command:

```
Bash Requesting a Cloud Shell. Succeeded.
Connecting terminal...
azureuser@Azure:~$
```

# Create a SignalR account - 2

```
SIGNALR_SERVICE_NAME=s10727211signalr
```

```
az signalr create \  
--name $SIGNALR_SERVICE_NAME \  
--resource-group 110-1-CS456L \  
--sku Free_DS2 \  
--unit-count 1
```

```
az resource update \  
--resource-type Microsoft.SignalRService/SignalR \  
--name $SIGNALR_SERVICE_NAME \  
--resource-group 110-1-CS456L \  
--set properties.features[flag=ServiceMode].value=Serverless
```

```
SIGNALR_CONNECTION_STRING=$(az signalr key list \  
--name $(az signalr list \  
--resource-group 110-1-CS456L \  
--query [0].name -o tsv) \  
--resource-group 110-1-CS456L \  
--query primaryConnectionString -o tsv)  
  
printf "\n\nReplace <SIGNALR_CONNECTION_STRING> with:\n$SIGNALR_CONNECTION_STRING\n\n"
```

# Update local settings - 1

The screenshot shows the Microsoft Azure Portal interface. The top navigation bar includes links for portal.azure.com, Azure Portal, Google Drive, datamining, GitHub, Problems, Syntax, and PPT模板. The user's email, 10727211@O365st.cy... 中原大學 (O365TC.CYCU.ED..., is visible on the right.

The main content area displays the '110-1-CS456L' resource group details. Below the search bar are buttons for Create, Edit columns, Delete resource group, Refresh, Export to CSV, Open query, Assign tags, Move, Delete, Export template, and more.

A terminal window titled 'Bash' is open, showing the following command history:

```
systemData": {  
    "createdAt": "2021-11-25T16:29:54.0172356Z",  
    "createdBy": "10727211@O365st.cy...edu.tw",  
    "createdByType": "User",  
    "lastModifiedAt": "2021-11-25T16:31:53.8547603Z",  
    "lastModifiedBy": "10727211@O365st.cy...edu.tw",  
    "lastModifiedByType": "User"  
},  
"tags": null,  
"type": "Microsoft.SignalRService/SignalR"  
}  
azureuser@Azure:~$  
azureuser@Azure:~$ SIGNALR_CONNECTION_STRING=$(az signalr key list \  
> --name $(az signalr list \  
> --resource-group 110-1-CS456L \  
> --query [0].name -o tsv) \  
> --resource-group 110-1-CS456L \  
> --query primaryConnectionString -o tsv)  
azureuser@Azure:~$  
azureuser@Azure:~$ printf "\n\nReplace <SIGNALR_CONNECTION_STRING> with:\n$SIGNALR_CONNECTION_STRING\n\n"
```

A callout box highlights the command output:

```
Replace <SIGNALR_CONNECTION_STRING> with:  
Endpoint=https://s10727211.signalr.service.signalr.net;AccessKey=fueKhvJ9R7DQGK6SkWwEFyipZA+DvUgUNJ9Jcx8yz50=;Version=1.0;
```

The terminal prompt ends with 'azureuser@Azure:~\$'.

# Update local settings - 2

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** local.settings.json - start - Visual Studio Code
- File Explorer:** Shows a tree view with "local.settings.json > ...".
- Editor:** Displays the contents of the local.settings.json file. The file is a JSON object with two main sections: "Values" and "Host".

```
1  {
2    "IsEncrypted": false,
3    "Values": {
4      "AzureWebJobsStorage": "DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=10727211model9storage;AccountKey=kDLWkdjioQZMAigvCvWTB4T4EBFxCp8IsTsk2woEe1+qVDioyrGlibgZc",
5      "AzureCosmosDBConnectionString": "AccountEndpoint=https://aci-cosmos-db-26309.documents.azure.com:443/;AccountKey=e92z5ydiZNoHPkG72i5zd3fLov39gaVawZqottc1HNZd7tEyUtIGI4DdqJQAtAtOKNC6ltd55HAPCjnghp10Q==",
6      "AzureCosmosDBMasterKey": "e92z5ydiZNoHPkG72i5zd3fLov39gaVawZqottc1HNZd7tEyUtIGI4DdqJQAtAtOKNC6ltd55HAPCjnghp10Q==",
7      "AzureSignalRConnectionString": "Endpoint=https://s10727211signalr.service.signalr.net;AccessKey=fueKhvJ9R7DQGK6SkWwEFyipZA+DvUgUNJ9Jcx8yz50=;Version=1.0;",
8      "FUNCTIONS_WORKER_RUNTIME": "node"
9    },
10   "Host" : {
11     "LocalHttpPort": 7071,
12     "CORS": "http://localhost:8080",
13     "CORS Credentials": true
14   }
15 }
16 }
```
- Sidebar:** Includes icons for File, Find, Replace, and other development tools.
- Bottom Status Bar:** Shows the current branch (master), file status (0 changes), and other details like "Attach to Node Functions (start)" and "Live Share".
- Bottom Right:** Includes links to Azure services (Table Service, Queue Service, Blob Service) and file metadata (Line 16, 16x24, UTF-8, CRLF, JSON).

# Manage client connections- 1

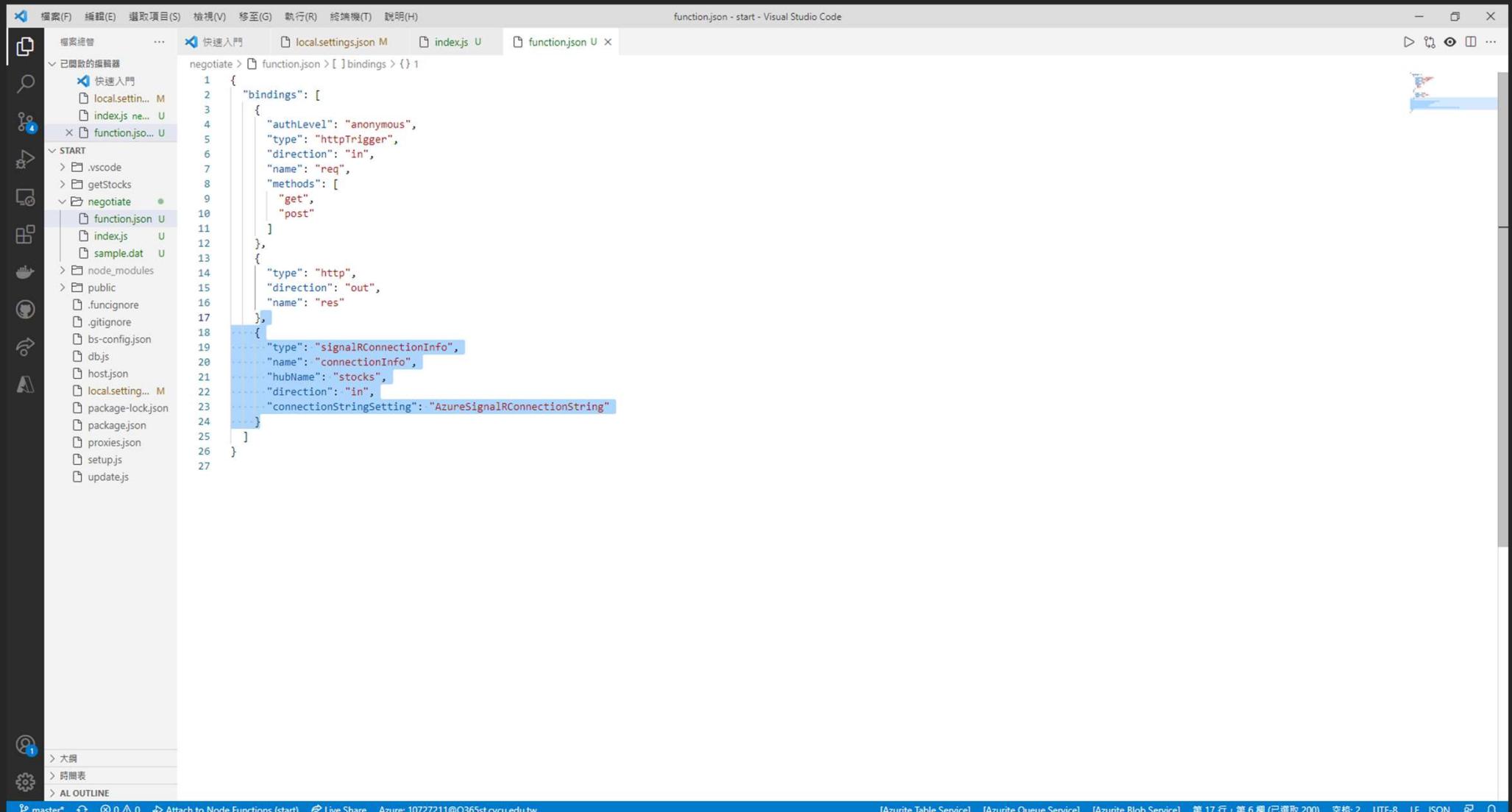
The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** 檔案(F)、編輯(E)、擷取項目(S)、檢視(V)、移至(G)、執行(R)、終端機(T)、說明(H)
- Toolbar:** Attach to Node Functions (start), Live Share, Azure: 10727211@O365st.cycu.edu.tw
- Left Sidebar:** 显示了监视器和呼叫堆栈。
- Code Editor:** 显示了 `local.settings.json` 文件的内容，包含 Azure Functions 配置。
- Context Menu:** 在文件编辑器中打开，包含以下选项：
  - Azure Functions: Open in Portal
  - Azure Storage: Deploy to Static Website via Azure Storage...
  - Azure Storage: Configure Static Website...
  - Azure Functions: Upload Local Settings...
  - Azure Functions: Deploy to Function App...
  - Azure: Sign In
  - Azure: Sign Out
  - Azure Functions: Create Function...
  - Azure: Start Queue Service
  - Azure Storage: Open in Storage Explorer
  - Azure Storage: Open in File Explorer...
  - Azure Functions: Create New Project...
  - 設定顯示語言
  - Configure Display Language
  - C/C++: 編輯組態 (UI)
  - C/C++: Edit Configurations (UI)
  - C/C++: 組建及偵錯使用中的檔案
  - C/C++: Build and Debug Active File
- Bottom Status Bar:** [Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] 第 16 行, 第 1 列 空格: 2 UTF-8 CRLF JSON

## Manage client connections- 2

Azure Functions: Create Function	
Template	HTTP Trigger
Name	negotiate
Authorization level	Anonymous

# Manage client connections- 3



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure with files like `index.js`, `function.json`, `sample.dat`, and `getStocks`.
- Editor:** The main editor area displays the `function.json` file content. The code defines a function named `negotiate` with the following configuration:

```
1  {
2    "bindings": [
3      {
4        "authLevel": "anonymous",
5        "type": "httpTrigger",
6        "direction": "in",
7        "name": "req",
8        "methods": [
9          "get",
10         "post"
11       ]
12     },
13     {
14       "type": "http",
15       "direction": "out",
16       "name": "res"
17     },
18     {
19       "type": "signalRConnectionInfo",
20       "name": "connectionInfo",
21       "hubName": "stocks",
22       "direction": "in",
23       "connectionStringSetting": "AzureSignalRConnectionString"
24     }
25   ]
26 }
```
- Bottom Status Bar:** Shows the current branch is `master`, the file count is 0, the line count is 0, and the status message `Attach to Node Functions (start)`. It also includes links for `Live Share` and `Azure: 10727211@O365st.cycu.edu.tw`.
- Bottom Right:** Includes links for `Azurite Table Service`, `Azurite Queue Service`, and `Azurite Blob Service`, along with the current line number (17), character position (6), and encoding information (UTF-8, LF, JSON).

# Manage client connections- 4

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure under "negotiate":
  - index.js
  - sample.dat
  - function.json
- Code Editor:** The main area displays the content of index.js:

```
module.exports = async function (context, req, connectionInfo) {
    context.res.body = connectionInfo;
};
```
- Status Bar:** At the bottom, it shows "master" and "Attach to Node Functions (start)" along with connection information: "Azure: 10727211@O365st.cycu.edu.tw".
- Bottom Navigation:** It includes links for "Azurite Table Service", "Azurite Queue Service", and "Azurite Blob Service".

# Detect and broadcast database changes - 1

	<b>Azure Functions: Create Function</b>
Template	Azure Cosmos DB Trigger
Name	stocksChanged
App setting for your Azure Cosmos DB account	AzureCosmosDBConnectionString
Database name	stocksdb
Collection name	stocks
Collection name for leases	leases
Create lease collection if not exists	true

# Detect and broadcast database changes - 2

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "已開啟的編輯器". The "stocksChanged" folder contains "function.json" (selected), "index.js", and "sample.dat". The "negotiate" folder contains "function.json", "index.js", and "sample.dat". The "stocksChan..." folder contains "function.json", "index.js", and "sample.dat". Other files like ".funcignore", "bs-config.json", "db.js", "host.json", "local.setting.json", "package-lock.json", "package.json", "proxies.json", "setup.js", and "update.js" are also listed.
- Code Editor:** The "function.json" file is open in the editor. It defines two bindings: one for a Cosmos DB trigger named "documents" and another for a SignalR hub named "stocks".

```
1  {
2    "bindings": [
3      {
4        "type": "cosmosDBTrigger",
5        "name": "documents",
6        "direction": "in",
7        "leaseCollectionName": "leases",
8        "connectionStringSetting": "AzureCosmosDBConnectionString",
9        "databaseName": "stocksdb",
10       "collectionName": "stocks",
11       "createLeaseCollectionIfNotExists": "true",
12       "feedPollDelay": 500
13     },
14     {
15       "type": "signalR",
16       "name": "signalRMessages",
17       "connectionString": "AzureSignalRConnectionString",
18       "hubName": "stocks",
19       "direction": "out"
20     }
21   ]
22 }
```
- Bottom Status Bar:** Displays "master" branch, file status (0 changes), "Attach to Node Functions (start)", "Live Share", and connection information "Azure: 10727211@O365st.cycu.edu.tw".
- Bottom Right:** Includes links to [Azurite Table Service], [Azurite Queue Service], [Azurite Blob Service], and footer text "第 23 行, 第 1 檔 空格: 2 UTF-8 LF JSON" and icons for copy, cut, and paste.

# Detect and broadcast database changes - 3

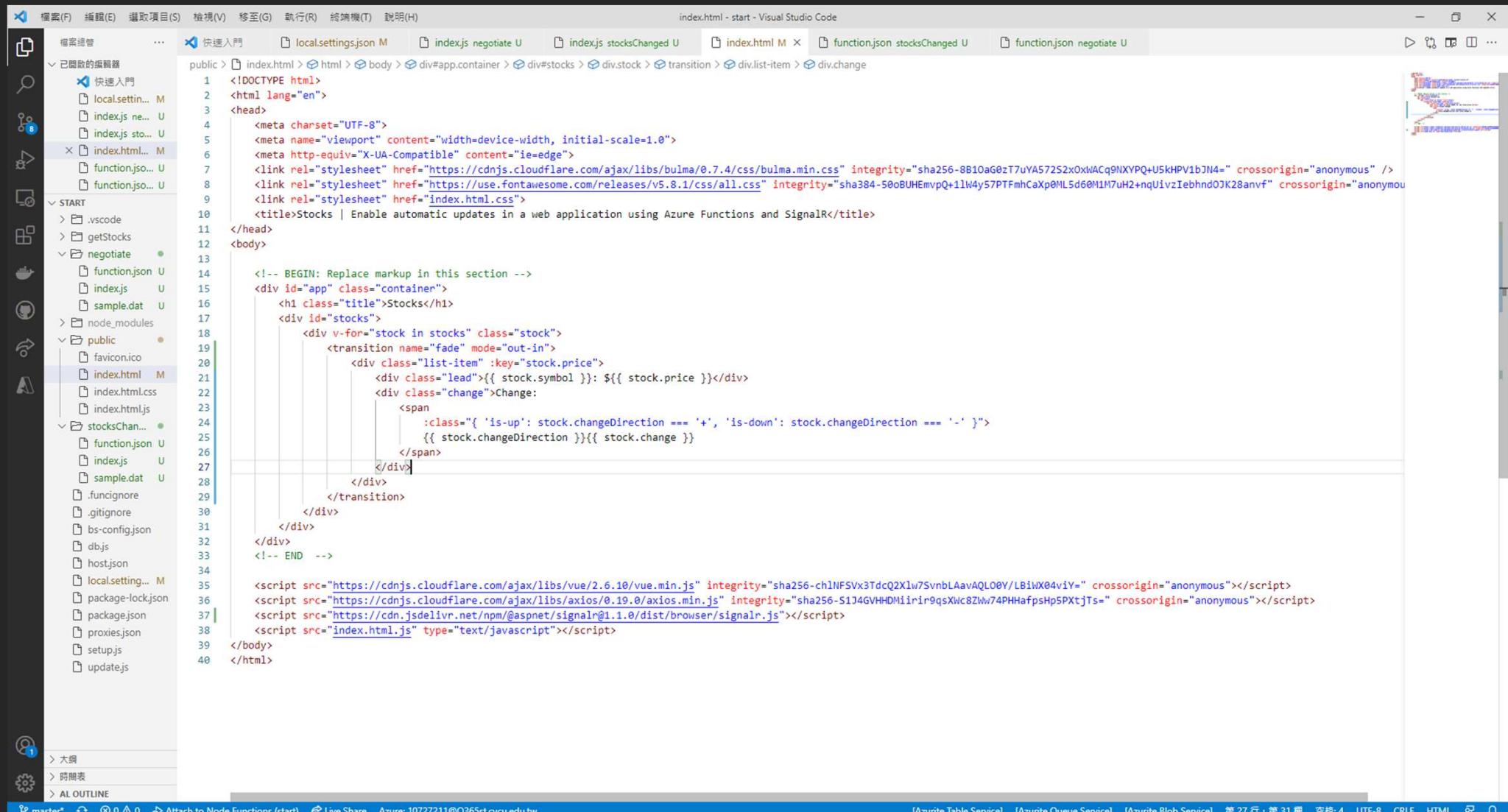
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a file tree with several folders and files. Notable files include `index.js`, `function.json`, `local.settings.json`, and `host.json`.
- Editor:** The main editor area displays the `index.js` file under the `stocksChanged` folder. The code is as follows:

```
1 module.exports = async function (context, documents) {
2     const updates = documents.map(stock => ({
3         target: 'updated',
4         arguments: [stock]
5     }));
6 
7     context.bindings.signalRMessages = updates;
8     context.done();
9 }
```

- Quick Open:** A search bar at the top labeled "快速入門" is visible.
- Status Bar:** At the bottom, it shows "master" and "Attach to Node Functions (start)" along with connection information for Azure.
- Bottom Bar:** Includes links for "Azure Table Service", "Azure Queue Service", and "Azure Blob Service".

# Update the web application - 1

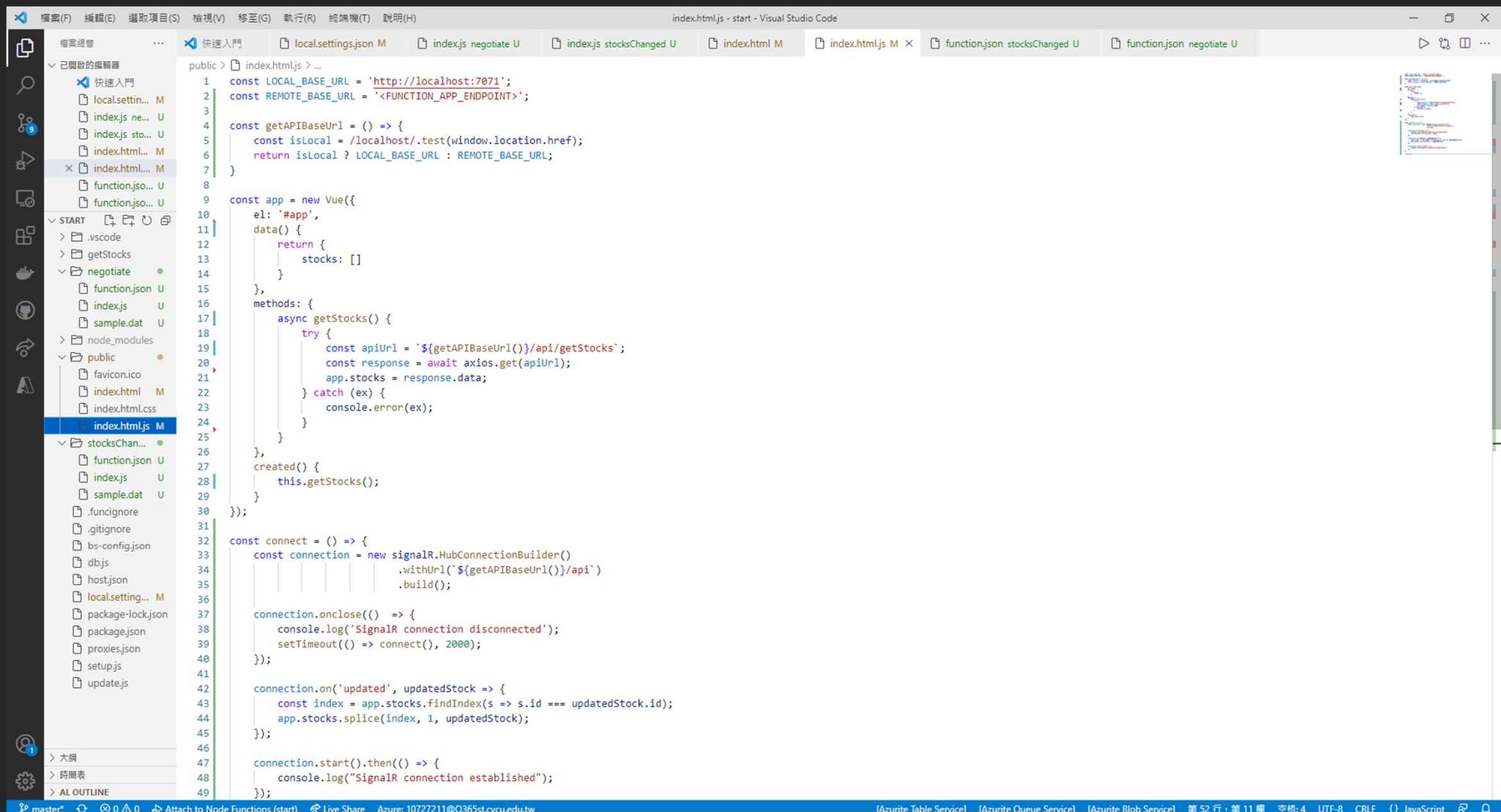


The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "START".
  - Folder ".vscode"
  - File "getStocks"
  - Folder "negotiate" (marked with a green dot)
    - File "function.json"
    - File "index.js"
    - File "sample.dat"- Editor (Center):** Displays the "index.html" file content.

```
public > index.html > html > body > div#app.container > div#stocks > div.stock > transition > div.list-item > div.change
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <meta http-equiv="X-UA-Compatible" content="ie=edge">
7      <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.7.4/css/bulma.min.css" integrity="sha256-8B10aG0zT7uYA572S2x0xwAc9NXPQ+U5kHPV1bJN4=" crossorigin="anonymous" />
8      <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.8.1/css/all.css" integrity="sha384-50oBUHEmpQ+llW4y57PTFnhCaXp0ML5d60M1M7uH2+nQuivZiebhnd0JK28anvf" crossorigin="anonymous" />
9      <link rel="stylesheet" href="index.html.css">
10     <title>Stocks | Enable automatic updates in a web application using Azure Functions and SignalR</title>
11 
12 </head>
13 
14 <!-- BEGIN: Replace markup in this section -->
15 <div id="app" class="container">
16     <h1 class="title">Stocks</h1>
17     <div id="stocks">
18         <div v-for="stock in stocks" class="stock">
19             <transition name="fade" mode="out-in">
20                 <div class="list-item" :key="stock.price">
21                     <div class="lead">{{ stock.symbol }}: {{ stock.price }}</div>
22                     <div class="change">Change:<br/>
23                         <span :class="{'is-up': stock.changeDirection === '+', 'is-down': stock.changeDirection === '-' }">
24                             {{ stock.changeDirection }}{{ stock.change }}
25                         </span>
26                     </div>
27                 </div>
28             </transition>
29         </div>
30     </div>
31 </div>
32 <!-- END -->
33 
34 
35 <script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.10/vue.min.js" integrity="sha256-ch1NFSVx3TdcQ2Xlw7SvnLAavAQL00Y/LBiWX04viY=" crossorigin="anonymous"></script>
36 <script src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.19.0/axios.min.js" integrity="sha256-S1j4GVHDMiirir9qsXNc8Zlw74PHHafpsHpSPXtjTs=" crossorigin="anonymous"></script>
37 <script src="https://cdn.jsdelivr.net/npm/@aspnet/signalr@1.1.0/dist/browser/signalr.js"></script>
38 <script src="index.html.js" type="text/javascript"></script>
39 
40 </body>
</html>
```
- Bottom Status Bar:** Shows the current branch ("master"), file status ("0 0 0"), and other details like "Attach to Node Functions (start)", "Live Share", and connection information.

# Update the web application - 2



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows the project structure. The file `index.html.js` is currently selected.
- Code Editor:** The main area displays the `index.html.js` file content. The code is a Vue.js application that fetches stock data from a local API endpoint (`localhost:7071`) or a remote endpoint (`<FUNCTION_APP_ENDPOINT>`). It uses Axios to make the request and SignalR to handle real-time updates.
- Status Bar:** At the bottom, it shows the current branch is "master", there are no changes (0), and the file is attached to Node Functions (start). It also includes links for "Live Share" and "Azure: 10727211@O365st.cycu.edu.tw".
- Bottom Navigation:** It includes links for "Azurite Table Service", "Azurite Queue Service", "Azurite Blob Service", and the current file path "第 52 行, 第 11 檔 空格:4 UTF-8 CRLF () JavaScript".

```
const LOCAL_BASE_URL = 'http://localhost:7071';
const REMOTE_BASE_URL = '<FUNCTION_APP_ENDPOINT>';

const getAPIBaseUrl = () => {
    const isLocal = /localhost/.test(window.location.href);
    return isLocal ? LOCAL_BASE_URL : REMOTE_BASE_URL;
}

const app = new Vue({
    el: '#app',
    data() {
        return {
            stocks: []
        }
    },
    methods: {
        async getStocks() {
            try {
                const apiUrl = `${getAPIBaseUrl()}/api/getStocks`;
                const response = await axios.get(apiUrl);
                app.stocks = response.data;
            } catch (ex) {
                console.error(ex);
            }
        }
    },
    created() {
        this.getStocks();
    }
});

const connect = () => {
    const connection = new signalR.HubConnectionBuilder()
        .withUrl(`${getAPIBaseUrl()}/api`)
        .build();

    connection.onclose(() => {
        console.log('SignalR connection disconnected');
        setTimeout(() => connect(), 2000);
    });

    connection.on('updated', updatedStock => {
        const index = app.stocks.findIndex(s => s.id === updatedStock.id);
        app.stocks.splice(index, 1, updatedStock);
    });

    connection.start().then(() => {
        console.log("SignalR connection established");
    });
}
```

# Run the application - 1

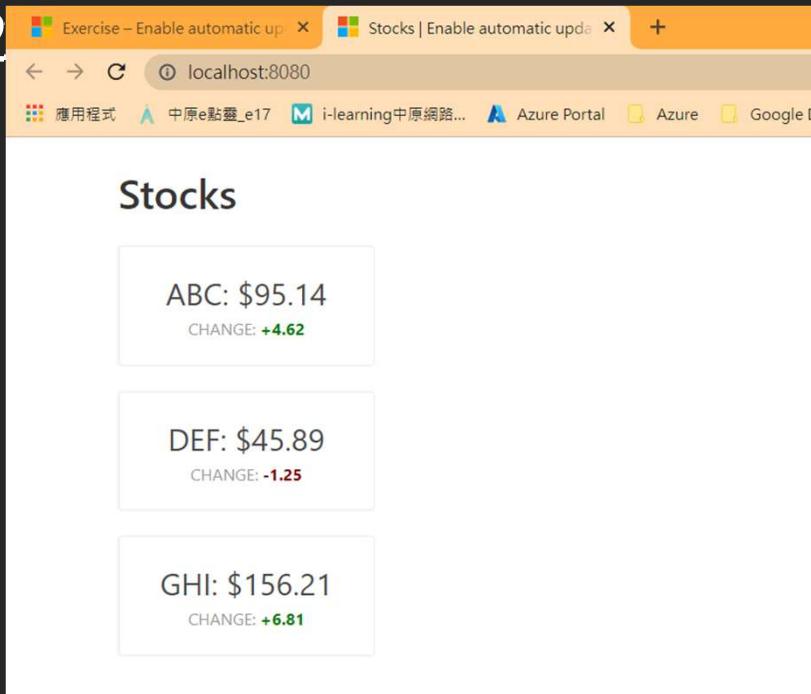
The screenshot shows a web browser window with the address bar set to `localhost:8080`. The browser's toolbar includes icons for back, forward, search, and refresh, along with a pinned tab for the Stocks application.

The Stocks application interface displays three stock entries:

- ABC:** \$90.52  
CHANGE: **-9.48**
- DEF:** \$45.89  
CHANGE: **-1.25**
- GHI:** \$156.21  
CHANGE: **+6.81**

The background of the browser window shows a blurred view of other tabs and the operating system's desktop environment.

# Run the application - 2



```
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo> npm run update-data
> microsoft-learn-func-signalr-start@1.0.0 update-data C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo
> node update.js

Read data from database.

Data updated: {"id": "e0eb6e85-176d-4ce6-89ae-1f699aaa0bab", "symbol": "ABC", "price": 95.14, "change": 4.62, "changeDirection": "+", "_rid": "dYUaALKsFUoBAAAAAAA==", "_self": "dbs/dYUaAA==/colls/dYUaALKsFUo=/docs/dYUaALKsFUoBAAAAA
AAAAAA==/", "_etag": "\"c700e882-0000-1000-619fb8970000\"", "_attachments": "attachments/", "_ts": 1637857431}
PS C:\Users\phil8\Desktop\Azure\1125_demo_serverless_workspace\serverless-demo>
```

The screenshot shows a Windows PowerShell window with several tabs at the top: 問題, 輸出, 傷標主控台, 編輯器. The main pane displays a command being run to update stock data from a database. The command is `npm run update-data` and it triggers a script named `update.js`. The output shows the data being updated from the database, including the stock symbol, price, and change.

# Login

F1

> Azure Sign in

檔案(F) 編輯(E) 選取項目(S) 檢視(V) 執行(R) 終端機(T) 說明(H)

index.html.js - start - Visual Studio Code

AZURE

> RESOURCE GROUPS

> HELP AND FEEDBACK

APP SERVICE

Select Subscriptions...

< FUNCTIONS

Select Subscriptions...

> Local Project start

< STORAGE

Select Subscriptions...

> Attached Storage...

< VIRTUAL MACHINES

Select Subscriptions...

< DATABASES

Select Subscriptions...

> Attached Datab...

1 const LOCAL\_BASE\_URL = 'http://localhost:7071';  
2 const REMOTE\_BASE\_URL = '<FUNCTION\_APP\_ENDPOINT>';  
3  
4 const getAPIBaseUrl = () => {  
5 const isLocal = /localhost/.test(window.location.href);  
6 return isLocal ? LOCAL\_BASE\_URL : REMOTE\_BASE\_URL;  
7 }  
8  
9 const app = new Vue({  
10 el: '#app',  
11 data() {  
12 return {  
13 stocks: []  
14 }  
15 },  
16 methods: {  
17 async getStocks() {  
18 try {  
19 const apiUrl = `\${getAPIBaseUrl()}/api/getStocks`;  
20 const response = await axios.get(apiUrl);  
21 app.stocks = response.data;  
22 } catch (ex) {  
23 console.error(ex);  
24 }  
25 },  
26 created() {  
27 this.getStocks();  
28 }  
29 },  
30 };  
31  
32 const connect = () => {  
33 const connection = new signalR.HubConnectionBuilder()  
34 .withUrl(`\${getAPIBaseUrl()}/api`)  
35 .build();  
36  
37 connection.onclose(() => {  
38 console.log('SignalR connection disconnected');  
39 setTimeout(() => connect(), 2000);  
40 });  
41  
42 connection.on('updated', updatedStock => {  
43 const index = app.stocks.findIndex(s => s.id === updatedStock.id);  
44 app.stocks.splice(index, 1, updatedStock);  
45 });  
46  
47 connection.start().then(() => {  
48 console.log('SignalR connection established');  
49 });  
50 };

[Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] 第 17 行 · 第 28 檔 空格:4 UTF-8 CRLF {} JavaScript

# Select Subscriptions

The screenshot shows the Visual Studio Code interface with the Azure Functions extension installed. The left sidebar contains navigation links for AZURE, RESOURCE GROUPS, HELP AND FEEDBACK, APP SERVICE, FUNCTIONS, STORAGE, VIRTUAL MACHINES, and DATABASES. The FUNCTIONS section has a 'Select Subscriptions...' option. The main editor area displays index.js code:

```
public > index.html.js > app > methods > getStocks
1 const LOCAL_BASE_URL = 'http://localhost:7071';
2 const REMOTE_BASE_URL = '<FUNCTION_APP_ENDPOINT>';
3
4 const getAPIBaseUrl = () => {
5     const isLocal = /localhost/.test(window.location.href);
6     return isLocal ? LOCAL_BASE_URL : REMOTE_BASE_URL;
7 }
8
9 const app = new Vue({
10     el: '#app',
11     data() {
12         return {
13             stocks: []
14         }
15     },
16     methods: {
17         async getStocks() {
18             try {
19                 const apiUrl = `${getAPIBaseUrl()}/api/getStocks`;
20                 const response = await axios.get(apiUrl);
21                 app.stocks = response.data;
22             } catch (ex) {
23                 console.error(ex);
24             }
25         },
26         created() {
27             this.getStocks();
28         }
29     });
30 }
31
32 const connect = () => {
33     const connection = new signalR.HubConnectionBuilder()
34         .withUrl(`${getAPIBaseUrl()}/api`)
35         .build();
36
37     connection.onclose(() => {
38         console.log('SignalR connection disconnected');
39         setTimeout(() => connect(), 2000);
40     });
41
42     connection.on('updated', updatedStock => {
43         const index = app.stocks.findIndex(s => s.id === updatedStock.id);
44         app.stocks.splice(index, 1, updatedStock);
45     });
46
47     connection.start().then(() => {
48         console.log("SignalR connection established");
49     });
}
```

A modal dialog titled 'Select Subscriptions' is displayed, containing a list box with one item: '中原大學 ba9f5e30-1488-49db-bae0-9026dacc11b0'. There are '已選擇 0' and '確定' buttons at the bottom of the modal.

The status bar at the bottom shows the current file path: 'index.html.js - start - Visual Studio Code', and other service status indicators: 'Azure: 10727211@O365st.cycu.edu.tw', '[Azurite Table Service]', '[Azurite Queue Service]', '[Azurite Blob Service]', '第 17 行, 第 28 檔', '空格:4', 'UTF-8', 'CRLF', 'JavaScript', and icons for Attach to Node Functions, Live Share, and Scanning.

# Deploy the function app - 1

Azure Functions: Deploy to Function App	
Template	Create new Function App in Azure... Advanced
Name	10727211-deployfunctionapp
Running Stack	Node.js 8 LTS
Resource Group	110-1-CS456L
Location for new resource	東南亞 (for CS456L)
Hosting Plan	Consumption
storage	10727211model9storage
Application Insights resource	Skip for now

# Deploy the function app - 2

The screenshot shows the Visual Studio Code interface with the Azure extension loaded. The left sidebar displays the Azure resource groups, including 'APP SERVICE' (中原大學), 'FUNCTIONS' (中原大學), 'STORAGE' (中原大學, Attached Storage...), and 'VIRTUAL MACHINES' (中原大學). The main editor area shows the 'index.html.js' file content:

```
public > index.html.js > [app] > data
1 const LOCAL_BASE_URL = 'http://localhost:7071';
2 const REMOTE_BASE_URL = '<FUNCTION_APP_ENDPOINT>';
3
4 const getAPIBaseUrl = () => {
5   const isLocal = /localhost/.test(window.location.href);
6   return isLocal ? LOCAL_BASE_URL : REMOTE_BASE_URL;
7 }
8
9 const app = new Vue({
10   el: '#app',
11   data() {
12     return {
13       stocks: []
14     }
15   },
16   methods: {
17     async getStocks() {
18       try {
19         const apiUrl = `${getAPIBaseUrl()}/api/getStocks`;
20         const response = await axios.get(apiUrl);
21         app.stocks = response.data;
22       } catch (ex) {
23         console.error(ex);
24       }
25     },
26   },
27   created() {
28     this.getStocks();
29   }
30 });

```

The bottom status bar shows deployment logs:

```
上午1:03:23: Ensuring App Service plan "ASP-10727211-deployfunctionapp-2bd8" exists...
上午1:03:24: Creating App Service plan "ASP-10727211-deployfunctionapp-2bd8"...
上午1:03:31: Successfully created App Service plan "ASP-10727211-deployfunctionapp-2bd8".
上午1:03:31: Creating new function app "10727211-deployfunctionapp"...
上午1:03:34: Successfully created function app "10727211-deployfunctionapp": https://10727211-deployfunctionapp.azurewebsites.net
上午1:04:39 10727211-deployfunctionapp: Starting deployment...
上午1:04:46 10727211-deployfunctionapp: Creating zip package...
```

A message in the bottom right corner says: 'Deploying to "10727211-deployfunctionapp"... Check output window for details.'

Bottom navigation bar:

```
git master* 0 0 Attach to Node Functions (start) Live Share (...) Scanning... Azure: 10727211@O365st.cycu.edu.tw [Azure Table Service] [Azure Queue Service] [Azure Blob Service] 第 11 行, 第 13 列 空格: 4 CRLF {} JavaScript
```

# Deploy the function app - 3

The screenshot shows the Visual Studio Code interface with the Azure Functions extension open. The left sidebar displays the Azure portal navigation pane with sections like RESOURCE GROUPS, APP SERVICE, FUNCTIONS, STORAGE, VIRTUAL MACHINES, and DATABASES. The main editor area shows the `index.js` file for a function named "negotiate". The code defines a Vue component that fetches stock data from an API. The bottom status bar indicates the deployment process is in progress, with messages such as "Scanning..." and "Azure: 10727211@O365st.cycu.edu.tw". The bottom right corner shows the Azure Functions tab is active.

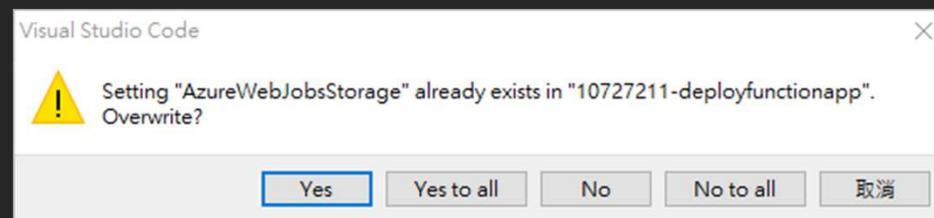
```
public > index.html.js > (e) REMOTE_BASE_URL
1 const LOCAL_BASE_URL = 'http://localhost:7071';
2 const REMOTE_BASE_URL = 'https://10727211-deployfunctionapp.azurewebsites.net';
3
4 const getAPIBaseUrl = () => {
5   const isLocal = /localhost/.test(window.location.href);
6   return isLocal ? LOCAL_BASE_URL : REMOTE_BASE_URL;
7 }
8
9 const app = new Vue({
10   el: '#app',
11   data() {
12     return {
13       stocks: []
14     }
15   },
16   methods: {
17     async getStocks() {
18       try {
19         const apiUrl = `${getAPIBaseUrl()}/api/getStocks`;
20         const response = await axios.get(apiUrl);
21         app.stocks = response.data;
22       } catch (ex) {
23         console.error(ex);
24       }
25     },
26     created() {
27       this.getStocks();
28     }
29   }
});
```

```
問題 跳出 儲存主控台 終端機
+-----+
上午1:03:31: Creating new function app "10727211-deployfunctionapp"...
上午1:04:34: Successfully created function app "10727211-deployfunctionapp": https://10727211-deployfunctionapp.azurewebsites.net
上午1:04:39 10727211-deployfunctionapp: Starting deployment...
上午1:04:46 10727211-deployfunctionapp: Creating zip package...
上午1:05:28 10727211-deployfunctionapp: Zip package size: 13.6 MB
上午1:05:31 10727211-deployfunctionapp: Updating submodules...
上午1:05:31 10727211-deployfunctionapp: Preparing deployment for commit id 'a672f2e727'.
上午1:05:32 10727211-deployfunctionapp: Skipping build. Project type: Run-From-Zip
上午1:05:32 10727211-deployfunctionapp: Skipping post build. Project type: Run-From-Zip
上午1:05:32 10727211-deployfunctionapp: Triggering recycle (preview mode disabled).
上午1:05:33 10727211-deployfunctionapp: Deployment successful.
上午1:05:52 10727211-deployfunctionapp: Syncing triggers...
上午1:06:01 10727211-deployfunctionapp: Querying triggers...
上午1:06:04 10727211-deployfunctionapp: HTTP Trigger Urls:
  getStocks: https://10727211-deployfunctionapp.azurewebsites.net/api/getstocks
  negotiate: https://10727211-deployfunctionapp.azurewebsites.net/api/negotiate
```

[Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] 第 2 行, 第 78 標 空格: 4 UFT-8 CRLF {} JavaScript

## Deploy the function app - 4

	Azure Functions: Upload local settings
Function App in Azure	10727211-deployfunctionapp



# Configure static websites in Azure Storage && Deploy the web application to Azure Storage

	<b>Azure Storage: Configure Static Website</b>
Storage account	10727211model9storage
Default file	Index.html
Error document	Index.html



	<b>Azure Storage: Deploy to static website via Azure Storage</b>
Storage account	10727211model9storage
Select the folder to deploy	Select <b>browse</b> and choose the <i>public</i> subfolder containing the web app.

# Set up CORS in the function app - 1

The screenshot shows the Azure Storage account settings for the '10727211model9storage' account. The left sidebar lists various storage management options like Geo-replication, Data protection, Object replication, Blob inventory, Static website, Lifecycle management, and Azure search. Under 'Settings', there are sections for Configuration, Data Lake Gen2 upgrade, Resource sharing (CORS), Advisor recommendations, Endpoints, Locks, Monitoring (with Insights, Alerts, Metrics, Workbooks, Diagnostic settings (preview), and Logs (preview)), and Monitoring (classic). The main content area is titled '10727211model9storage | Static website'. It includes a note about enabling static websites on the blob service, a status indicator for 'Static website' (Enabled), and a note that an Azure Storage container has been created to host the static website. The 'Primary endpoint' is listed as `https://10727211model9storage.z23.web.core.windows.net/`. Below this, fields for 'Index document name' (set to 'index.html') and 'Error document path' (also set to 'index.html') are shown. A 'Save' button is visible at the top of the form.

# Set up CORS in the function app - 2

The screenshot shows the Microsoft Azure portal interface for managing a Function App named "10727211-FunctionApp". The left sidebar contains a navigation menu with various sections like API, Monitoring, Automation, and Support + troubleshooting. The "CORS" section is currently selected under the API category. The main content area displays the CORS configuration settings. It includes a "Request Credentials" section with a checked checkbox for "Enable Access-Control-Allow-Credentials". Below it is a "Allowed Origins" section where the URL "https://10727211model9storage.z23.web.core.windows.net" has been entered into a text input field. The browser's address bar shows the URL "portal.azure.com/#@O365tc.cygu.edu.tw/resource/subscriptions/ba9f5e30-1488-49db-bae0-9026dacc11b0/resourceGroups/110-1-CS456L/providers/Microsoft.Web/sites/10727211-FunctionApp/apiCors".

# Run the deployed application

The screenshot shows a Microsoft Edge browser window with three tabs open:

- 10727211-deployfunctionapp
- Exercise - Use a storage account
- Stocks | Enable automatic update

The address bar displays the URL: `10727211model9storage.z23.web.core.windows.net`. The page content is titled "Stocks" and lists three stocks with their current price and change:

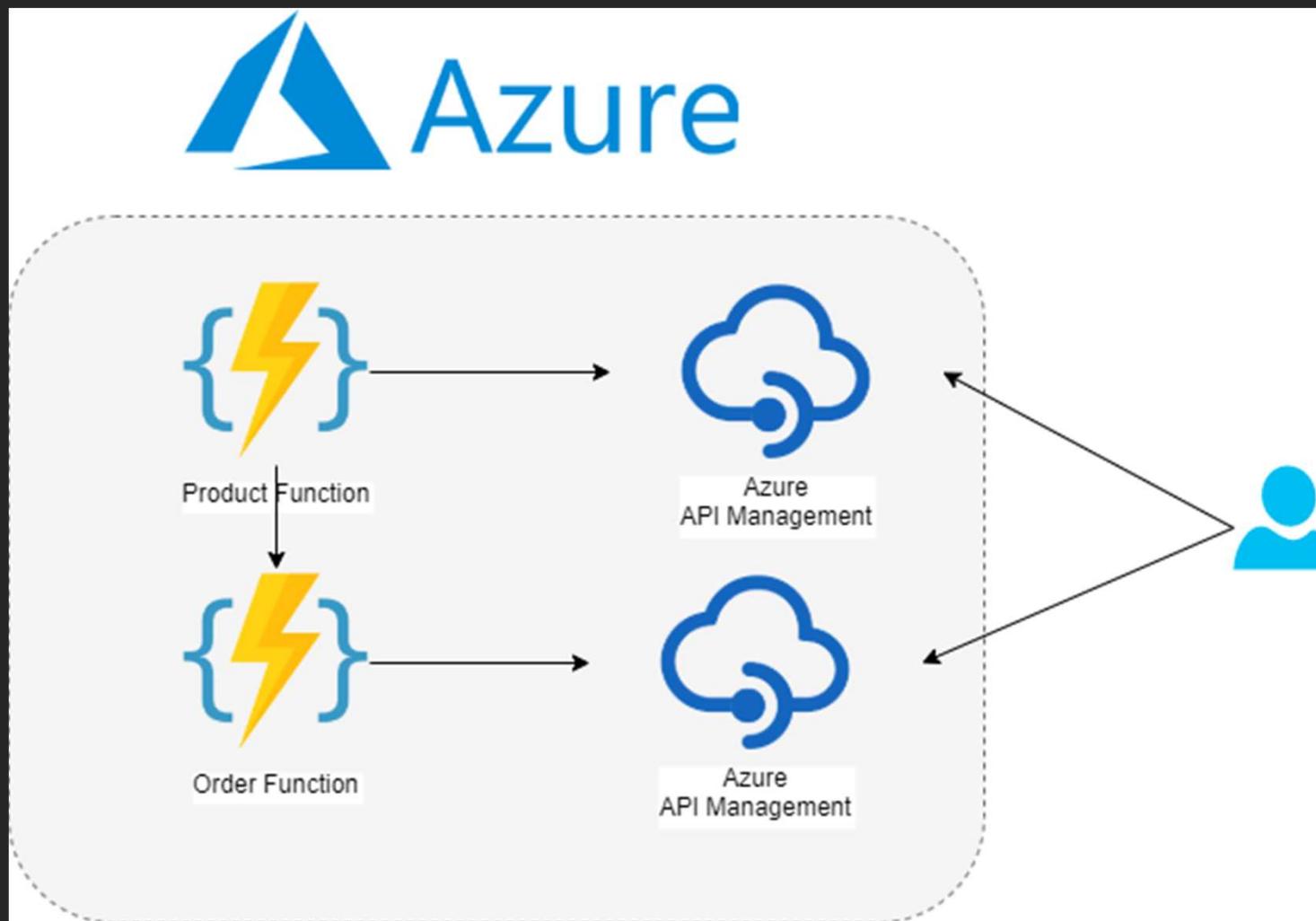
Stock	Price	Change
ABC	\$100.49	+8.4
DEF	\$45.89	-1.25
GHI	\$156.21	+6.81

The browser interface includes standard navigation buttons (back, forward, search), a ribbon menu, and a toolbar with various icons.

10

// Expose multiple Azure Function  
apps as a consistent API by using  
Azure API Management

# Abstract



# Environment

Sandbox recommend

官方提供腳本 Create Resource 時，名稱會是random (管理不易)

網址：<https://docs.microsoft.com/en-us/learn/modules/build-serverless-api-with-functions-api-management/3-exercise-import-function-app-api-management>

# Create functions - 1

```
Bash    ▾ | ⌂ ? ⚙ ⌂ ⌂ {} ⌂ ... — □ ×  
Requesting a Cloud Shell.Succeeded.  
Connecting terminal...  
  
Welcome to Azure Cloud Shell  
  
Type "az" to use Azure CLI  
Type "help" to learn about Cloud Shell  
  
azureuser@Azure:~$ git clone https://github.com/MicrosoftDocs/mslearn-apim-and-functions.git  
~/OnlineStoreFuncs ← Red arrow  
Cloning into '/home/azureuser/OnlineStoreFuncs'...  
remote: Enumerating objects: 39, done.  
remote: Counting objects: 100% (39/39), done.  
remote: Compressing objects: 100% (34/34), done.  
remote: Total 39 (delta 16), reused 9 (delta 4), pack-reused 0  
Unpacking objects: 100% (39/39), done.  
azureuser@Azure:~$  
azureuser@Azure:~$ cd ~/OnlineStoreFuncs ← Yellow arrow  
azureuser@Azure:~/OnlineStoreFuncs$ bash setup.sh ← Yellow arrow  
  
The resource group is called learn-cbff884e-d559-4845-b052-78a4882fad4e and is located in we  
stus  
  
Creating a storage account for the functions...  
[- Running ..
```

# Test the product details function - 1

The screenshot shows the Microsoft Azure Functions blade for the 'ProductFunction9146e8ab3a' Function App. The left sidebar lists various resources, with 'ProductFunction9146e8ab3a' highlighted and a red box around it. Below it, 'Functions' is highlighted with a yellow box. The main content area displays the 'ProductDetails' function, which was created in a local environment and cannot be edited in the portal. The function has an HTTP trigger and is currently enabled.

ProductFunction9146e8ab3a | Functions

Function App

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Security Events (preview)

Filter by name...

Name ↑	Trigger ↑	Status ↑
ProductDetails	HTTP	Enabled

ProductDetails

Functions

App keys App files Proxies

Deployment slots Deployment Center

# Test the product details function - 2

The screenshot shows the Microsoft Azure portal interface for testing an Azure Function named 'ProductDetails'. The URL in the browser is [portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/code/resourceId/%2Fsubscriptions%2F2eb288f0-205d-43c5-9356-449b90daee2e%2FresourceGroups...](https://portal.azure.com/#blade/WebsitesExtension/FunctionMenuBlade/code/resourceId/%2Fsubscriptions%2F2eb288f0-205d-43c5-9356-449b90daee2e%2FresourceGroups...).

The page title is 'ProductDetails - Microsoft Azure'. The top navigation bar includes links for 'Search resources, services, and docs (G+)', 'Azure Portal', 'Google Drive', 'datamining', 'GitHub - huihui03...', 'Problems - LeetCo...', 'Syntax', '專題cite', 'PPT模版', and 'MICROSOFT LEARN SANDB...'. The user account '10727211@O365st.cycu...' is also visible.

The main content area shows the 'ProductDetails | Code + Test' blade. On the left, under 'Developer', the 'Code + Test' option is selected and highlighted with a red box. The 'Test/Run' button is also highlighted with a yellow box. The code editor displays the 'function.json' file:

```
1  {
2     "generatedBy": "Microsoft.NET.Sdk.Functions-1.0.24",
3     "configurationSource": "attributes",
4     "bindings": [
5         {
6             "type": "httpTrigger",
7             "methods": [
8                 "get",
9                 "post"
10            ],
11            "authLevel": "function",
12            "name": "req"
13        }
14    ],
15    "disabled": false,
16    "scriptFile": "../bin/ProductDetailsFunc.dll",
17    "entryPoint": "ProductDetailsFunc.ProductDetails.Run"
18 }
```

To the right, the 'Input' tab is selected. The 'HTTP method' dropdown is set to 'GET' and is highlighted with a yellow box. The 'Key' dropdown is set to 'master (Host key)'. The 'Query' section contains a parameter 'id' with a value of '3', which is highlighted with a green box. The 'Headers' section is empty. At the bottom, there are 'Run' and 'Close' buttons, with 'Run' being highlighted with a blue box.

# Test the product details function - 3

The screenshot shows the Microsoft Azure portal interface for testing a function named "ProductDetails".

**Code + Test** tab is selected in the developer sidebar.

The function.json file content is:

```
1  {
2    "generatedBy": "Microsoft.NET.Sdk.Functions-1.0.24",
3    "configurationSource": "attributes",
4    "bindings": [
5      {
6        "type": "httpTrigger",
7        "methods": [
8          "get",
9          "post"
10         ],
11       }
12     ],
13   }
```

The logs pane shows the execution of the function:

```
2021-11-29T09:41:54.220 [Information] Executing 'ProductDetails' (Reason='This function was programmatically called via the host APIs.', Id=b8857850-519e-4a89-bf3f-032763c34d6f)
2021-11-29T09:41:54.223 [Information] C# HTTP trigger function processed a request.
2021-11-29T09:41:54.228 [Information] Executed 'ProductDetails' (Succeeded, Id=b8857850-519e-4a89-bf3f-032763c34d6f, Duration=8ms)
```

The test results pane is highlighted with a red box and shows the following output:

**Output** tab is selected.

**HTTP response code**: 200 OK

**HTTP response content**:

```
{
  "ID": 3,
  "Name": "Smart Dimmer Switch",
  "Price": 59.99,
  "PartNumber": "DS728"
}
```

**Run** and **Close** buttons are at the bottom of the test results pane.

# Test the product details function - 4

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes links for portal.azure.com, Azure Portal, Google Drive, datamining, GitHub, Problems, Syntax, and PPT模板. The user's email, 10727211@O365st.cycu..., and Microsoft Learn Sandboxed profile are also visible.

The main content area displays the "ProductDetails" function configuration page. The left sidebar lists Overview, Developer, Code + Test, and Integration. The "Get Function Url" section shows the URL https://productfunction9146e8ab3a.azurewebsites.net/api/ProductDetails?code=DQCoeNYTge5RTKl1s0A0HpEZ4pJiUvVQ723/palu5YoAgQj0sbaeFQ==&id=3. A "Copy to clipboard" button is available next to the URL.

The screenshot shows a browser window with the address bar containing the URL https://productfunction9146e8ab3a.azurewebsites.net/api/ProductDetails?code=DQCoeNYTge5RTKl1s0A0HpEZ4pJiUvVQ723/palu5YoAgQj0sbaeFQ==&id=3. The page content displays the JSON response: {"ID":3,"Name":"Smart Dimmer Switch","Price":59.99,"PartNumber":"DS728"}.

# Expose function app as an API using Azure API Management - 1

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for '練習 - 在 API 管理中從函數應用' (Practice - Exposing Functions via API Management), 'ProductFunction9146e8ab3a -' (the current active tab), and 'https://productfunction9146e8ab3a.azurewebsites.net'.

The main content area displays the 'ProductFunction9146e8ab3a | API Management' page for a Function App. On the left, the 'All resources' sidebar lists various Azure resources, with 'ProductFunction9146e8ab3a' highlighted by a red box. The 'API' section of the sidebar has 'API Management' highlighted by a yellow box.

The right panel contains the 'API Management' configuration section. It features a heading 'API Management' with the subtext 'Expose your HTTP trigger Functions through Azure API Management - manage, protect, secure, and publish the APIs.' Below this is a note 'Select API Management instance and API for your App.' and a dropdown menu labeled 'API Management \*' with a 'Create new' button highlighted by a yellow box. Other options in the dropdown include 'Import Functions' (checked) and 'Enable Application Insights' (unchecked). At the bottom is a 'Link API' button.

# Expose function app as an API using Azure API Management - 2

The screenshot shows the Microsoft Azure portal interface for creating a new API Management service. The browser tabs include '練習 - 在 API 管理中從函數應用' (Practice - From Function App), 'API Management service - Mic...' (selected), and 'https://productfunction9146e...'. The address bar shows the URL for creating a new API Management resource.

The main content area is titled 'API Management service' and contains the following fields:

- Name \***: ProductFunction9146e8ab3a-apim (with a green checkmark)
- Subscription \***: Concierge Subscription
- Resource group \***: learn-cbff884e-d559-4845-b052-78a4882fad4e (with a dropdown arrow)
- Create new** (link)
- Location \***: (US) West US (with a dropdown arrow)
- Organization name \***: OnlineStore (highlighted with a red border)
- Administrator email \***: 10727211@O365st.cycu.edu.tw (with a green checkmark)
- Pricing tier (View full pricing details)**: Consumption (99.95% SLA) (with a dropdown arrow)

A blue button labeled 'Export' is highlighted with a yellow border at the bottom left.

# Expose function app as an API using Azure API Management - 3

The screenshot shows the Microsoft Azure portal interface. The left sidebar displays 'All resources' under 'Microsoft Learn Sandbox'. A list of resources is shown, including 'cloudshell284238723', 'OrderFunction30124ea1de', 'ProductFunction9146e8ab3a' (which is selected), and 'storestorage8e89c05b8a'. The main content area is titled 'ProductFunction9146e8ab3a | API Management' and shows the 'Function App' blade. On the right, the 'API Management' section is visible, featuring a summary message: 'Expose your HTTP trigger Functions through Azure API Management - manage, protect, secure, and publish the APIs.' Below this, there's a note: 'Your App is linked to the API Management instance: 'ProductFunction9146e8ab3a-apim''. A dropdown menu for selecting an API is present, with the option '-- Create New --'. There are also checkboxes for 'Import Functions' (checked) and 'Enable Application Insights' (unchecked). A prominent blue button at the bottom right is labeled 'Link API', which is highlighted with a red rectangular box.

# Expose function app as an API using Azure API Management - 4

The screenshot shows the Microsoft Azure portal interface. The browser tab bar includes '練習 - 在 API 管理中從函數應用', 'Import Azure Functions - Micro...', and 'https://productfunction9146e...'. The address bar shows the URL for a specific Azure Function. The top navigation bar has links for '應用程式', '中原e點靈\_e17', 'i-learning中原網路...', 'Azure Portal', 'Azure', 'Google Drive', 'datamining', 'GitHub - huihui03...', 'Problems - LeetCode...', 'Syntax', '專題cite', 'PPT模板', and '閱讀清單'. The main header says 'Microsoft Azure' and 'Search resources, services, and docs (G/+)'. On the right, there's a user profile for '10727211@O365st.cy... MICROSOFT LEARN SANDB...'.

The page title is 'Import Azure Functions' under 'API Management service'. A message box says: 'Don't see an Azure Function? Azure API Management requires Azure Functions to use the HTTP trigger and Function or Anonymous authorization level setting.' Below is a search bar and a table:

Name	HTTP methods	URL template
<input checked="" type="checkbox"/> ProductDetails	GET, POST	ProductDetails

A yellow box highlights the 'Select' button at the bottom left of the table.

# Expose function app as an API using Azure API Management - 5

The screenshot shows the Microsoft Azure portal interface. The left sidebar lists various resources under 'All resources' for the 'ProductFunction9146e8ab3a' service. The main panel displays the configuration for 'ProductFunction9146e8ab3a | API Management'. The 'Basic' tab is selected, showing the following fields:

- \* Function App: ProductFunction9146e8ab3a
- \* Display name: ProductFunction9146e8ab3a
- \* Name: productfunction9146e8ab3a
- API URL suffix: products (highlighted with a red box)
- Base URL: https://productfunction9146e8ab3a-apim.azure-api.net/products

At the bottom right, there are 'Create' and 'Cancel' buttons. The browser address bar shows the URL: https://productfunction9146e8ab3a.apim.azure-api.net/.

# Test the OnlineStore products endpoint - 1

The screenshot shows the Microsoft Azure API Management interface for a Function App named "ProductFunction9146e8ab3a". The left sidebar is expanded, showing various monitoring and diagnostic settings. The main area displays the "ProductDetails" endpoint under the "API Management" tab. The "Test" tab is selected, highlighted with a yellow box. The "GET" method for the "ProductDetails" endpoint is also highlighted with a yellow box. The "Host" section shows the name "productfunction9146e8ab3a-apim.azure-api.r". The "Query parameters" section contains a table with one row:

NAME	VALUE	TYPE	DESCRIPTION
id	3	string	Additional parameter.

A green box highlights the first row of the table. At the bottom, there is a "Send" button highlighted with a blue box and a "Bypass CORS proxy" checkbox.

# Test the OnlineStore products endpoint - 2

The screenshot shows the Microsoft Azure API Management console for a Function App named "ProductFunction9146e8ab3a". The left sidebar navigation includes "Console", "Advanced Tools", "App Service Editor (Preview)", "Extensions", "API", "API Management" (which is selected), "API definition", "CORS", "Monitoring", "Alerts", "Metrics", "Health check", "Logs", "Diagnostic settings", and "App Service logs". The main content area displays the "ProductDetails" endpoint under "API Management". The "Test" tab is active, showing a "REVISION 1" entry created on Nov 29, 2021, at 5:59:42 PM. The "GET" method is highlighted. The response body is displayed in a code editor-like view, with the JSON content of the response highlighted by a red box:

```
ProductFunction9146e8ab3a > ProductDetails > Console
ocp-apim-apiid: productfunction9146e8ab3a
ocp-apim-operationid: get-productdetails
ocp-apim-subscriptionid: master
ocp-apim-trace-location: https://apimstufaqnq9najukgl8jj.blob.core.windows.net/apimsticiqzgtzructd8xdgk-inspector/YJTyn080CxmbcQ2-1?sv=2018-03-28&sr=c&sig=M0Pl7DX%2B6g1J9XzM0ZROy%2BTsRHMuJjuHZKrQZNSbao%3D&se=2022-11-29T09%3A57%3A45Z&sp=racwdl&traceId=fd0e424e83c0cd887dccf13f
request-context: appId=cid-v1:13a43f06-22c8-4b2c-b240-f3c093870063
transfer-encoding: chunked
vary: Accept-Encoding,Origin
{
  "ID": 3,
  "Name": "Smart Dimmer Switch",
  "Price": 59.99,
  "PartNumber": "DS728"
}
```

At the bottom of the interface are "Send" and "Bypass CORS proxy" buttons.

# Test the OrderDetails function - 1

The screenshot shows the Microsoft Azure portal interface. The left sidebar lists various resources under 'All resources' for the 'Microsoft Learn Sandbox'. One resource, 'OrderFunction30124ea1de', is selected and highlighted with a red box. On the right, the main content area displays the 'OrderFunction30124ea1de | Functions' page for a 'Function App'. The 'Functions' section shows a table with one item:

Name	Trigger	Status
OrderDetails	HTTP	Enabled

The 'OrderDetails' row is highlighted with a yellow box. The entire screenshot is framed by a large yellow border.

# Test the OrderDetails function - 2

The screenshot shows the Microsoft Azure portal interface for testing an Azure Function named 'OrderDetails'. The top navigation bar includes tabs for '練習 - 將另一個 Azure Function...' and 'OrderDetails - Microsoft Azure'. The URL in the address bar is <https://productfunction9146e02c9d44f8a3--orderdetails.netlify.app/>.

The main content area displays the 'OrderDetails | Code + Test' page. On the left, a sidebar under 'Developer' shows 'Code + Test' selected. The main pane shows the function.json file content:

```
1  {
2    "generatedBy": "Microsoft.NET.Sdk.Functions-1.0.24",
3    "configurationSource": "attributes",
4    "bindings": [
5      {
6        "type": "httpTrigger",
7        "methods": [
8          "get",
9          "post"
10         ],
11        "authLevel": "function",
12        "name": "req"
13      }
14    ],
15    "disabled": false,
16    "scriptFile": "../bin/OrderShippingFunc.dll",
17    "entryPoint": "OrderShippingFunc.OrderDetails.Run"
18 }
```

The 'Test/Run' button in the top navigation bar is highlighted with a red box. The 'Input' tab is selected in the test configuration panel, which contains fields for 'HTTP method' (set to 'GET'), 'Key' (set to 'master (Host key)'), and a 'Query' table with a single entry: 'name' (Value: 'Chiba'). The 'Run' button at the bottom of the test panel is highlighted with a green box.

# Test the OrderDetails function - 3

The screenshot shows the Microsoft Azure portal interface for testing an Azure Function named 'OrderDetails'. The URL in the browser is <https://productfunction9146e.azurewebsites.net/api/OrderDetails>.

The Azure portal navigation bar includes links for 'Home', 'All resources', 'OrderFunction30124ea1de', and 'OrderDetails'.

The main area displays the 'Code + Test' view for the 'OrderDetails' function. On the left, there's a 'Developer' sidebar with options: 'Code + Test' (selected), 'Integration', 'Monitor', and 'Function Keys'. The 'Code + Test' section shows the 'function.json' file content:

```
1  {
2    "generatedBy": "Microsoft.NET.Sdk.Functions-1.0.24",
3    "configurationSource": "attributes",
4    "bindings": [
5      {
6        "type": "httpTrigger",
7        "methods": [
8          "get",
9          "post"
10        ],
11        "route": "OrderDetails"
12      }
13    ]
14 }
```

Below the code editor is a log window showing execution details:

```
2021-11-29T10:12:03.378 [Information] Executing 'OrderDetails' (Reason='This function was programmatically called via the host APIs.', Id=ef6e1030-7bc1-4ffc-97f7-4fec2aafa5c5)
2021-11-29T10:12:03.378 [Information] C# HTTP trigger function processed a request.
2021-11-29T10:12:03.378 [Information] Executed 'OrderDetails' (Succeeded, Id=ef6e1030-7bc1-4ffc-97f7-4fec2aafa5c5, Duration=0ms)
```

To the right, a 'Test' panel is open with tabs for 'Input' and 'Output'. The 'Output' tab shows the HTTP response code as '200 OK' and the response content as a JSON object:

```
{
  "ID": 72945,
  "CustomerFirstName": "Yuki",
  "CustomerLastName": "Chiba",
  "Total": 442.5,
  "Shipped": false
}
```

At the bottom of the test panel are 'Run' and 'Close' buttons.

# Add a function to an existing API - 1

The screenshot shows the Microsoft Azure portal interface. On the left, the 'All resources' sidebar lists various Azure resources, with 'OrderFunction30124ea1de' highlighted and surrounded by a red box. In the main content area, the title is 'OrderFunction30124ea1de | API Management'. The 'API' section is expanded, showing options like 'API Management' (which is also highlighted with a yellow box), 'API definition', 'CORS', 'Monitoring', 'Logs', 'Diagnostic settings', 'App Service logs', 'Log stream', and 'Process explorer'. Below the API section, there's a form for 'API Management' configuration, with a dropdown menu set to 'Create new' (also highlighted with a yellow box). There are also checkboxes for 'Import Functions' (checked) and 'Enable Application Insights' (unchecked). A 'Link API' button is visible at the bottom of this form.

# Add a function to an existing API - 2

Microsoft Azure Search resources, services, and docs (G+ /)

Home > All resources > OrderFunction30124ea1de >

## API Management service

OrderFunction30124ea1de-apim .azure-api.net

Subscription \*

Concierge Subscription

Resource group \*

learn-cbff884e-d559-4845-b052-78a4882fad4e

Create new

Location \*

(US) West US

Organization name \* ⓘ

OnlineStore

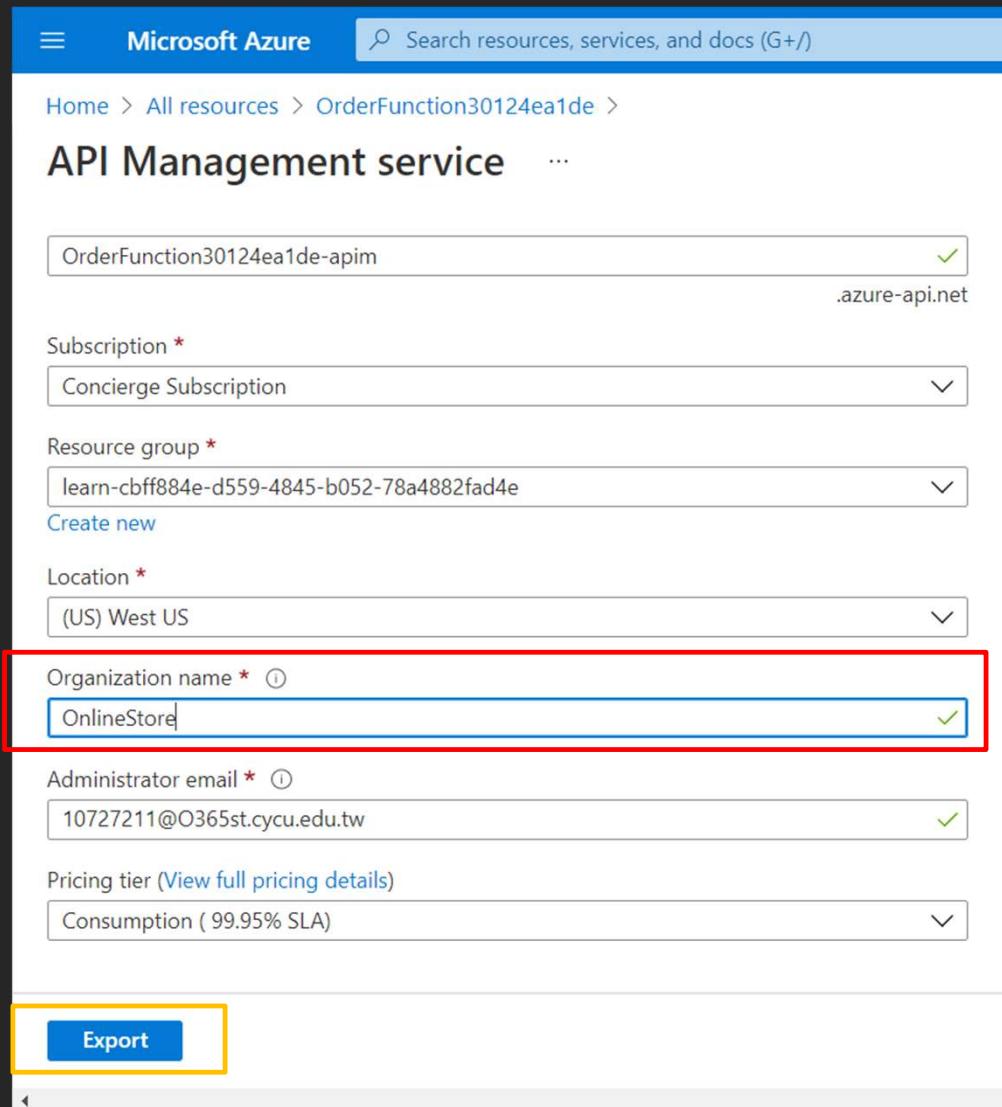
Administrator email \* ⓘ

10727211@O365st.cycu.edu.tw

Pricing tier (View full pricing details)

Consumption ( 99.95% SLA )

Export



# Add a function to an existing API - 3

The screenshot shows the Microsoft Azure portal interface. The main title is "OrderFunction30124ea1de | API Management". On the left, there's a sidebar titled "All resources" with a "Function App" icon. The main content area has a "Search (Ctrl+/" input field. Below it, there are two buttons: "Go to API Management" (highlighted with a red box) and "Unlink API Management". The "API Management" section contains a sub-section titled "API Management" with the subtext "Expose your HTTP trigger Functions through Azure API Management - manage, protect, secure, and publish the APIs. Learn more". It also displays a message: "Your App is linked to the API Management instance: 'OrderFunction30124ea1de-apim'". There are sections for "Select API for your App.", "API \*", which includes a dropdown menu with "-- Create New --", and checkboxes for "Import Functions" (checked) and "Enable Application Insights" (unchecked). At the bottom is a blue "Link API" button.

# Add a function to an existing API - 4

Screenshot of the Microsoft Azure portal showing the API Management service for the OrderFunction30124ea1de-apim resource.

The left sidebar shows the navigation path: Home > All resources > OrderFunction30124ea1de > OrderFunction30124ea1de-apim.

The main content area displays four representation formats for APIs:

- OpenAPI**: Standard, language-agnostic interface to REST APIs.
- WADL**: Standard XML representation of your RESTful API.
- WSDL**: Standard XML representation of your SOAP API.

Below these, there is a section titled "Create from Azure resource" with four options:

- Logic App**: Scalable hybrid integrations and workflows.
- App Service**: API hosted on App Service.
- Function App**: Serverless, event driven experience on App Service. This option is highlighted with a yellow border.
- Container App**: Serverless containers for microservices.

The "APIs" item in the left sidebar is highlighted with a red border, indicating it is the current selected category.

# Add a function to an existing API - 5

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for '練習 - 將另一個 Azure Functions' (Practice - Add another Azure Function), 'OrderFunction30124ea1de-api' (the current page), and a link to 'https://productfunction9146e...'. Below the navigation bar is a search bar and a ribbon menu with various links like '應用程式', '中原e點靈\_e17', 'i-learning中原網路...', 'Azure Portal', 'Google Drive', 'datamining', 'GitHub - huihui03...', 'Problems - LeetCode...', 'Syntax', '專題cite', 'PPT模板', and '閱讀清單'.

The main content area displays the 'OrderFunction30124ea1de-apim | APIs' page under the 'API Management service'. On the left, there's a sidebar with links for 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Events (preview)', 'Properties', 'Locks', and 'APIs' (which is currently selected). The main panel shows a 'Create from Function App' dialog. The dialog has two tabs: 'Basic' (selected) and 'Full'. It contains fields for 'Function App' (with a placeholder 'Please select Function App' and a 'Browse' button highlighted with a red box), 'Display name' (e.g. 'Http Bin'), 'Name' (e.g. 'httpbin'), 'API URL suffix' (e.g. 'httpbin'), and 'Base URL' (https://orderfunction30124ea1de-apim.azure-api.net). At the bottom right of the dialog are 'Create' and 'Cancel' buttons.

# Add a function to an existing API - 6

The screenshot shows a Microsoft Azure portal window with the following details:

- Address Bar:** https://productfunction9146e...  
portal.azure.com/#@learn.docs.microsoft.com/resource/subscriptions/2eb288f0-205d-43c5-9356-449b90daee2e/resourcegroups/learn-cbfff884e-d559-4845-b052-78a...
- Header:** Microsoft Azure, Search resources, services, and docs (G+/-)
- Breadcrumbs:** Home > All resources > OrderFunction30124ea1de > OrderFunction30124ea1de-apim >
- Title:** Import Azure Functions
- Sub-Title:** API Management service
- Information Bar:** Don't see an Azure Function? Azure API Management requires Azure Functions to use the HTTP trigger and Function or Anonymous authorization level setting.
- Form Fields:**
  - Function App: A dropdown menu with a "Select" button highlighted by a red box.
  - Search to filter items...
  - Name: An input field with a checkbox.
  - HTTP methods: A list of checkboxes.
  - URL template: An input field.
- Results:** No results
- Buttons:** Select (at the bottom left) and a large blue Select button (at the top right of the configuration area).

# Add a function to an existing API - 7

**Select Azure Function App**

API Management service

Name	Resource group	Location
OrderFunction30124ea1de	learn-cbff884e-d559-4845-b052-78a4882fa...	West US
ProductFunction9146e8ab3a	learn-cbff884e-d559-4845-b052-78a4882fa...	West US

**Select**

# Add a function to an existing API - 8

The screenshot shows the Microsoft Azure portal interface for importing Azure Functions into an API Management service. The URL in the browser is <https://productfunction9146e0c0d4f449b90daee2e/resourcegroups/learn-cbfff884e-d559-4845-b052-78a...>.

The page title is "Import Azure Functions - Microsoft Azure". The top navigation bar includes links for Microsoft Learn, Azure Portal, Google Drive, GitHub, and Microsoft Learn Sandboxed.

The main content area displays the "Import Azure Functions" configuration screen. It shows a message: "Don't see an Azure Function? Azure API Management requires Azure Functions to use the HTTP trigger and Function or Anonymous authorization level setting." Below this, there is a search bar and a table for configuring required settings.

The table has three columns: "Name", "HTTP methods", and "URL template". There are two rows:

Name	HTTP methods	URL template
<input checked="" type="checkbox"/> OrderDetails	GET, POST	OrderDetails

A yellow box highlights the "Select" button at the bottom left of the table. A red box highlights the "OrderDetails" row, specifically the checkbox column.

# Add a function to an existing API - 9

The screenshot shows the Microsoft Azure portal interface for managing APIs. The URL in the browser is <https://productfunction9146e.azurewebsites.net/>. The left sidebar shows the navigation path: Home > All resources > OrderFunction30124ea1de > OrderFunction30124ea1de-apim.

The main content area displays the 'OrderFunction30124ea1de-apim | APIs' page. A modal dialog titled 'Create from Function App' is open. The 'Basic' tab is selected. The form fields are as follows:

- Function App: OrderFunction30124ea1de
- Display name: OrderFunction30124ea1de
- Name: orderfunction30124ea1de
- API URL suffix: orders (highlighted with a red box)
- Base URL: https://orderfunction30124ea1de-apim.azure-api.net/orders

The 'Create' button at the bottom right of the dialog is highlighted with a yellow box.

# Test the OnlineStore orders endpoint in the portal - 1

The screenshot shows the Microsoft Azure API Management service interface. On the left, the navigation menu is visible with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview), Properties, Locks, APIs, Products, Subscriptions, and Named values. The APIs section is currently selected.

In the main content area, the title is "OrderFunction30124ea1de-apim | APIs". A revision history shows "REVISION 1" created on Nov 29, 2021, at 6:17:32 PM. The "Test" tab is selected. The "All APIs" list contains one item: "OrderFunction30124ea1de...". Below it, the "OrderDetails" endpoint is shown with two methods: "GET OrderDetails ..." and "POST OrderDetails ...". The "POST" method is highlighted with a yellow box.

The "OrderDetails" endpoint configuration includes:

- Host:** Name: orderfunction30124ea1de-apim.
- Query parameters:**

NAME	VALUE	TYPE	DESCRIPTION
name	Chiba	string	Additional parameter.

+ Add parameter
- Send** button (highlighted with a blue box)
- Bypass CORS proxy

# Test the OnlineStore orders endpoint in the portal - 2

The screenshot shows the Microsoft Azure API Management service interface. The left sidebar is titled "OrderFunction30124ea1de-apim | APIs" and includes sections for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview), Properties, Locks, APIs, Products, Subscriptions, and Named values. The main content area is titled "REVISION 1" (CREATED Nov 29, 2021, 6:17:32 PM) and shows the "Test" tab selected. It displays the "OrderFunction30124ea1de > OrderDetails > Console" section. The "GET OrderDetails" and "POST OrderDetails" operations are listed. The "POST" operation's response body is highlighted with a red box and contains the following JSON:

```
{  
    "ID": 72945,  
    "CustomerFirstName": "Yuki",  
    "CustomerLastName": "Chiba",  
    "Total": 442.5,  
    "Shipped": false  
}
```

At the bottom right of the main content area are "Send" and "Bypass CORS proxy" buttons.

# Test the combined API - 1

The screenshot shows the Microsoft Azure 'All resources' blade. The URL in the browser is <https://productfunction9146e8ab3a.azurewebsites.net/>. The page displays a list of 8 resources, each with a checkbox, name, type, resource group, location, and subscription information. The 'OrderFunction30124ea1de-apim' resource is highlighted with a red box.

Name	Type	Resource group	Location	Subscription
cloudshell284238723	Storage account	learn-cbff884e-d559-4845-b052-78...	Central India	Concierge Subscription
OrderFunction30124ea1de	Application Insights	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
OrderFunction30124ea1de	Function App	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
OrderFunction30124ea1de-apim	API Management service	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
ProductFunction9146e8ab3a	Application Insights	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
ProductFunction9146e8ab3a	Function App	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
storestorage8e89c05b8a	Storage account	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription
WestUSPlan	App Service plan	learn-cbff884e-d559-4845-b052-78...	West US	Concierge Subscription

# Test the combined API - 2

The screenshot shows the Microsoft Azure portal interface. At the top, there are three tabs: '練習 - 將另一個 Azure Function' (练习 - 将另一个 Azure Function), 'OrderFunction30124ea1de-api' (OrderFunction30124ea1de-api), and a third tab showing a URL. The main navigation bar includes links for '應用程式' (App), '中原e點靈\_e17', 'i-learning中原網路...', 'Azure Portal', 'Google Drive', 'datamining', 'GitHub - huihui03...', 'Problems - LeetCode...', 'Syntax', '專題cite', 'PPT模版', '閱讀清單' (Reading List), and user information '10727211@O365st.cycu... MICROSOFT LEARN SANDB...'.

In the center, the 'Microsoft Azure' search bar contains 'Search resources, services, and docs (G+/-)'. To its right are several icons: a magnifying glass, a gear, a question mark, a person icon, and a refresh symbol. A yellow box highlights the refresh icon.

The main content area displays the 'OrderFunction30124ea1de-apim' API Management service details. The left sidebar has a 'Search (Ctrl+/' input field and a list of options: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview). The 'Overview' option is selected. The right pane shows the 'Essentials' section with the following details:

- Resource group (Move) : learn-cbff884e-d559-4845-b052-78a4882fad4e
- Status : Online
- Location : West US
- Subscription (Move) : Concierge Subscription
- Subscription ID : 2eb288f0-205d-43c5-9356-449b90daee2e
- Tags (Edit) : Click here to add tags
- Tier : Consumption
- Gateway URL : <https://orderfunction30124ea1de-apim.azure-api.net> (highlighted with a red box)

Below the essentials section are 'Properties', 'Get started', 'Learn more', 'Monitor', and 'Recommendations (0)' buttons. The 'Properties' button is underlined.

At the bottom, a terminal window titled 'Bash' shows the command: `azureuser@Azure: $ GATEWAY_URL=https://orderfunction30124ea1de-apim.azure-api.net`. This command is also highlighted with a yellow box.

# Test the combined API - 3

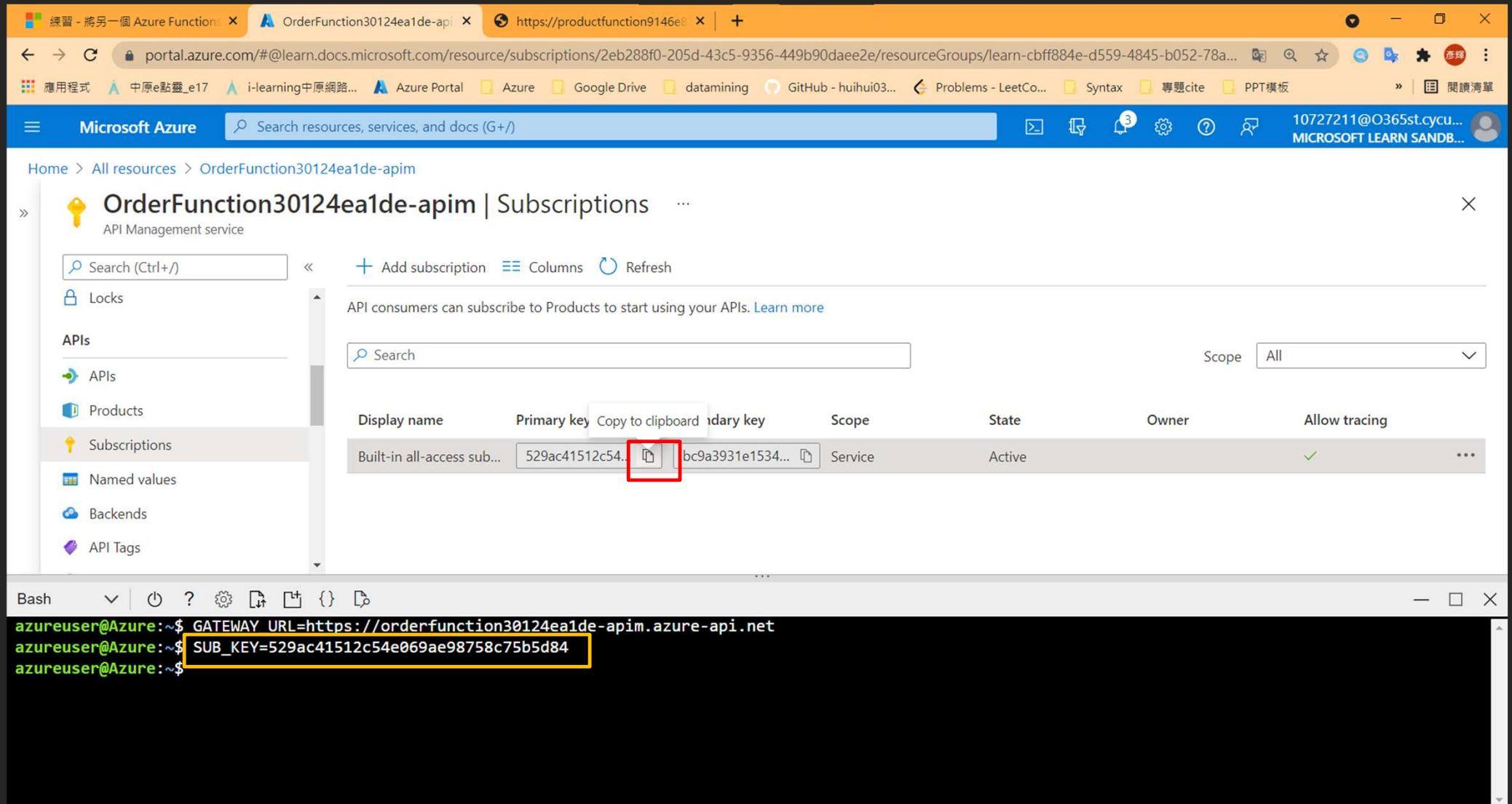
The screenshot shows the Microsoft Azure portal interface. The user is on the 'Subscriptions' page for the API Management service 'OrderFunction30124ea1de-apim'. The left sidebar shows navigation options like Home, All resources, and OrderFunction30124ea1de-apim. The main area displays a table of subscriptions with columns: Display name, Primary key, Secondary key, Scope, State, Owner, and Allow tracing. A context menu is open over the first row of the table, with the 'Show/hide keys' option highlighted.

Display name	Primary key	Secondary key	Scope	State	Owner	Allow tracing
Built-in all-access sub...	.....	.....	Service	Active		<span>Show/hide keys</span>

Context menu options (highlighted 'Show/hide keys'):   
Activate subscription  
Submit subscription  
Suspend subscription  
Reject subscription  
Cancel subscription  
Delete subscription  
Regenerate primary key  
Regenerate secondary key

```
azureuser@Azure:~$ GATEWAY_URL=https://orderfunction30124ea1de-apim.azure-api.net
azureuser@Azure:~$
```

# Test the combined API - 4



The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for '練習 - 將另一個 Azure Functions' (练习 - 将另一个 Azure Functions), 'OrderFunction30124ea1de-api' (OrderFunction30124ea1de-api), and a browser tab for 'https://productfunction9146e...'. The main content area is titled 'OrderFunction30124ea1de-apim | Subscriptions' and is described as an 'API Management service'. On the left, a sidebar lists 'Locks', 'APIs' (selected), 'Products', 'Subscriptions' (highlighted in blue), 'Named values', 'Backends', and 'API Tags'. The main pane displays a table of subscriptions. The first row, 'Built-in all-access sub...', has columns for 'Display name', 'Primary key' (containing '529ac41512c54e069ae98758c75b5d84'), 'Copy to clipboard' (with a red box around it), 'Secondary key' (containing 'bc9a3931e1534...'), 'Scope' (Service), 'State' (Active), 'Owner' (empty), and 'Allow tracing' (checked). Below the table is a Bash terminal window showing command-line output:

```
azureuser@Azure:~$ GATEWAY_URL=https://orderfunction30124ea1de-apim.azure-api.net
azureuser@Azure:~$ SUB_KEY=529ac41512c54e069ae98758c75b5d84
azureuser@Azure:~$
```

# Test the combined API - 5

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for '練習 - 將另一個 Azure Function' (练习 - 将另一个 Azure Function), 'OrderFunction30124ea1de-api' (OrderFunction30124ea1de-api), and a browser tab for 'https://productfunction9146e...'.

The main content area is titled 'OrderFunction30124ea1de-apim | Subscriptions' and displays a list of API consumer subscriptions. The table columns include 'Display name', 'Primary key', 'Secondary key', 'Scope', 'State', 'Owner', and 'Allow tracing'. One row is visible for a 'Built-in all-access sub...' with keys '529ac41512c54...' and 'bc9a3931e1534...', scope 'Service', state 'Active', owner 'MICROSOFT LEARN SANDB...', and allow tracing checked.

On the left sidebar under 'APIs', the 'Subscriptions' option is selected. A terminal window at the bottom shows the following command sequence:

```
Bash
azureuser@Azure:~$ GATEWAY_URL=https://orderfunction30124ea1de-apim.azure-api.net
azureuser@Azure:~$ SUB_KEY=529ac41512c54e069ae98758c75b5d84
azureuser@Azure:~$ curl -X GET "$GATEWAY_URL/orders/OrderDetails?name=Henri" -H "Ocp-Apim-Subscription-Key:$SUB_KEY"
{"ID":56224,"CustomerFirstName":"Pascale","CustomerLastName":"Henri","Total":307.98,"Shipped":true}
```

# Test the combined API - 6

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes tabs for '練習 - 將另一個 Azure Function' (Exercise - Create another Azure Function), 'ProductFunction9146e8ab3a-apim' (the current active tab), and 'https://productfunction9146e8ab3a...'. The address bar shows the URL of the current page. The main content area is titled 'ProductFunction9146e8ab3a-apim | Subscriptions' under 'API Management service'. On the left, a sidebar lists 'APIs', 'Products', 'Subscriptions' (which is selected and highlighted in grey), 'Named values', and 'Backends'. The main table displays subscription details:

Display name	Primary key	Secondary key	Scope	State	Owner	Allow tracing
Built-in all-access subs...	1ebc383f472f4f...	173b7cf26fce47...	Service	Active		<input checked="" type="checkbox"/>

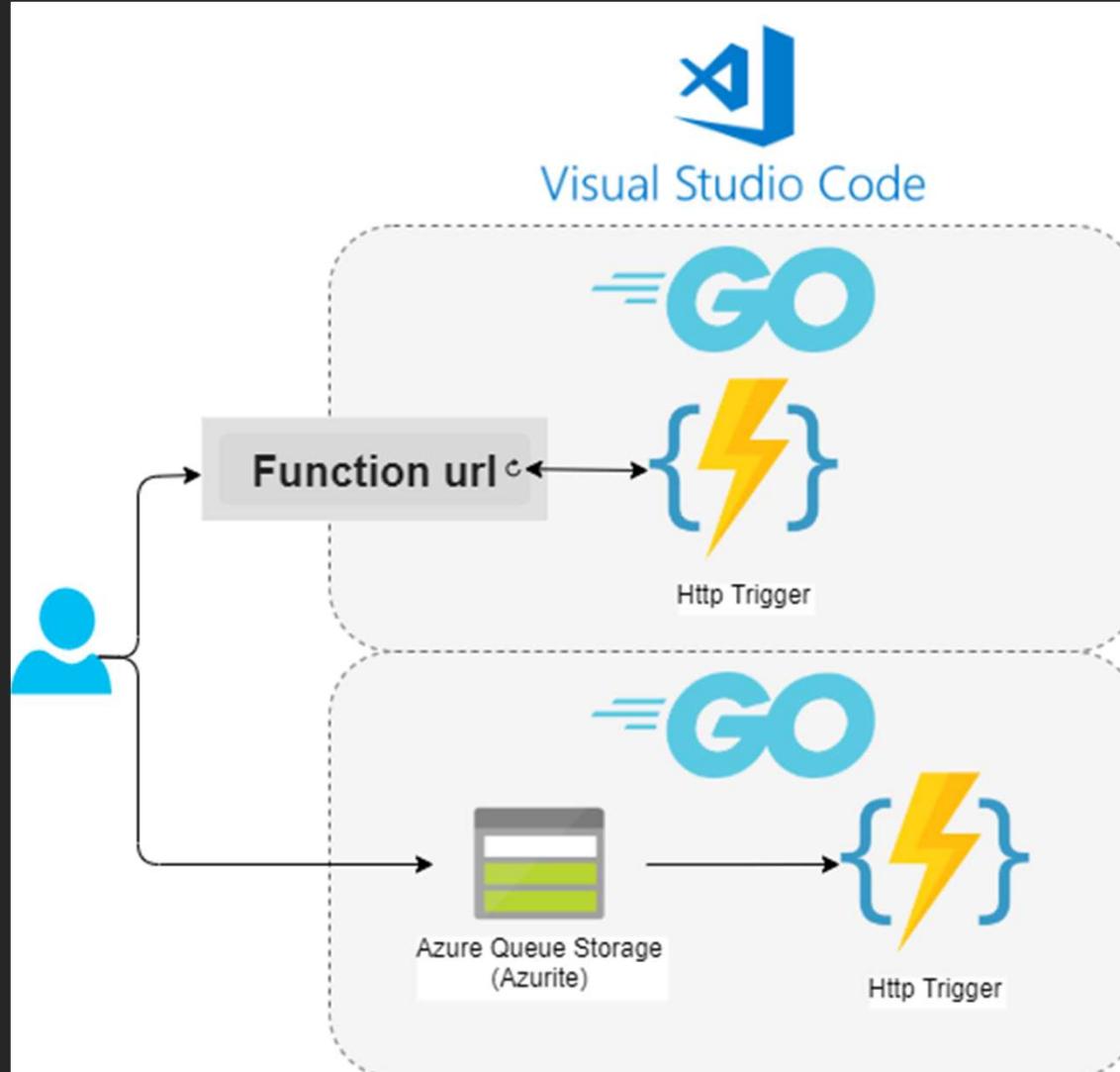
Below the table is a 'Bash' terminal window showing command-line interactions:

```
azureuser@Azure:~$ GATEWAY_URL=https://orderfunction30124ea1de-apim.azure-api.net
azureuser@Azure:~$ SUB_KEY=529ac41512c54e069ae98758c75b5d84
azureuser@Azure:~$ curl -X GET "$GATEWAY_URL/orders/OrderDetails?name=Henri" -H "Ocp-Apim-Subscription-Key:$SUB_KEY"
azureuser@Azure:~$ 
azureuser@Azure:~$ curl -X GET "$GATEWAY_URL/products/ProductDetails?id=2" -H "Ocp-Apim-Subscription-Key:$SUB_KEY"
{"ID":2,"Name":"Home Security Camera","Price":105.99,"PartNumber":"SC967"}azureuser@Azure:~$
```

A red box highlights the text '切換至productfunction apim，再做一次 Test the combined API' (Switch to productfunction apim, do it again). A yellow box highlights the second curl command in the terminal.

# 11 // Build serverless apps with Go

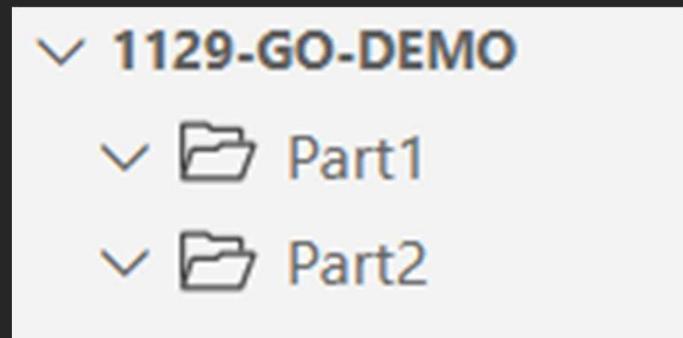
# Abstract



# Starter

Env : Visual Studio Code

/Go-Demo  
  ├ Part1  
  └ Part2



# Part1

切換至Part1目錄

Cmd :

Code <path>

Eg: code .

Eg: code.\Part1\

```
PS C:\Users\phil8\Desktop\Azure\1129-go-demo>
PS C:\Users\phil8\Desktop\Azure\1129-go-demo>
PS C:\Users\phil8\Desktop\Azure\1129-go-demo> code .\Part1\
```

## Scaffold the app

Azure Functions: Create New Project	
Select the folder that will contain your function project	{Part1}
Select a language	Custom Handler
Select a template for your first function	HttpTrigger
Provide a function name	hello
授權等級	Anonymous

# Create the app

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a tree view of files and folders. A red box highlights the '+' icon next to the 'PART1' folder, indicating where new files can be added. A yellow box highlights the 'server.go' file in the 'hello' folder.
- Code Editor:** The main area displays the 'server.go' file content. The code is written in Go and defines a server that listens on port 8080 and handles a GET request for '/api/hello' by writing 'hello world' to the response.
- Status Bar:** At the bottom, it shows the current branch ('master'), the Go version ('Go 1.17.3'), the number of changes ('0'), the Live Share status, and the Azure connection information ('Azure: 10727211@O365st.cycu.edu.tw').
- Bottom Navigation:** It includes links for [Azurite Table Service], [Azurite Queue Service], [Azurite Blob Service], and the current file location ('第 28 行, 第 22 欄').

```
1 package main
2
3 import (
4     "fmt"
5     "io/ioutil"
6     "log"
7     "net/http"
8     "os"
9 )
10
11 func main() {
12     customHandlerPort, exists := os.LookupEnv("FUNCTIONS_CUSTOMHANDLER_PORT")
13     if !exists {
14         customHandlerPort = "8080"
15     }
16     mux := http.NewServeMux()
17     mux.HandleFunc("/api/hello", helloHandler)
18     fmt.Println("Go server Listening on: ", customHandlerPort)
19     log.Fatal(http.ListenAndServe(": "+customHandlerPort, mux))
20 }
21
22 func helloHandler(w http.ResponseWriter, r *http.Request) {
23     w.Header().Set("Content-Type", "application/json")
24     if r.Method == "GET" {
25         w.Write([]byte("hello world"))
26     } else {
27         body, _ := ioutil.ReadAll(r.Body)
28         w.Write(body)
29     }
30 }
31
```

# Run the app - 1

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure under "PART1". The "server.go" file is selected and open in the editor.
- Editor:** The main editor area displays the content of "server.go".

```
1 package main
2
3 import (
4     "fmt"
5     "io/ioutil"
6     "log"
7     "net/http"
```
- Terminal:** At the bottom, a terminal window is open with the command "go build server.go" entered and executed.

```
PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part1> go build server.go
PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part1>
```

# Run the app - 2

host.json - Part1 - Visual Studio Code

檔案(F) 編輯(E) 選取項目(S) 檢視(V) 移至(G) 執行(R) 終端機(T) 說明(H)

已開啟的編輯器

PART1

host.json

```
1  {
2    "version": "2.0",
3    "logging": {
4      "applicationInsights": {
5        "samplingSettings": {
6          "isEnabled": true,
7          "excludedTypes": "Request"
8        }
9      },
10     "extensionBundle": {
11       "id": "Microsoft.Azure.Functions.ExtensionBundle",
12       "version": "[2.*, 3.0.0)"
13     },
14     "customHandler": {
15       "description": {
16         "defaultExecutablePath": "./server.exe",
17         "workingDirectory": "",
18         "arguments": []
19       },
20       "enableForwardingHttpRequest" : true
21     }
22   }
23 }
```

2. Open the `host.json` file and find the `defaultExecutablePath` element inside the `customHandler` one. Specify `./server` on macOS and Linux, or `.\server.exe` on a Windows OS.

```
17   "defaultExecutablePath": ".\server.exe",
18   "workingDirectory": "",
19   "arguments": []
20 },
21   "enableForwardingHttpRequest" : true
22 }
23 }
```

問題 輸出 備註主控台 終端機

Windows PowerShell  
Copyright (C) Microsoft Corporation. 著作權所有，並保留一切權利。  
請嘗試新的跨平台 PowerShell <https://aka.ms/powershell>

PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part1> func start

Azure Functions Core Tools  
Core Tools Version: 3.0.3904 Commit hash: c345f7140a8f968c5dbc621f8a8374d8e3234206 (64-bit)  
Function Runtime Version: 3.3.1.0

Unable to parse host configuration file 'C:\Users\phil8\Desktop\Azure\1129-go-demo\Part1\host.json'.  
PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part1>

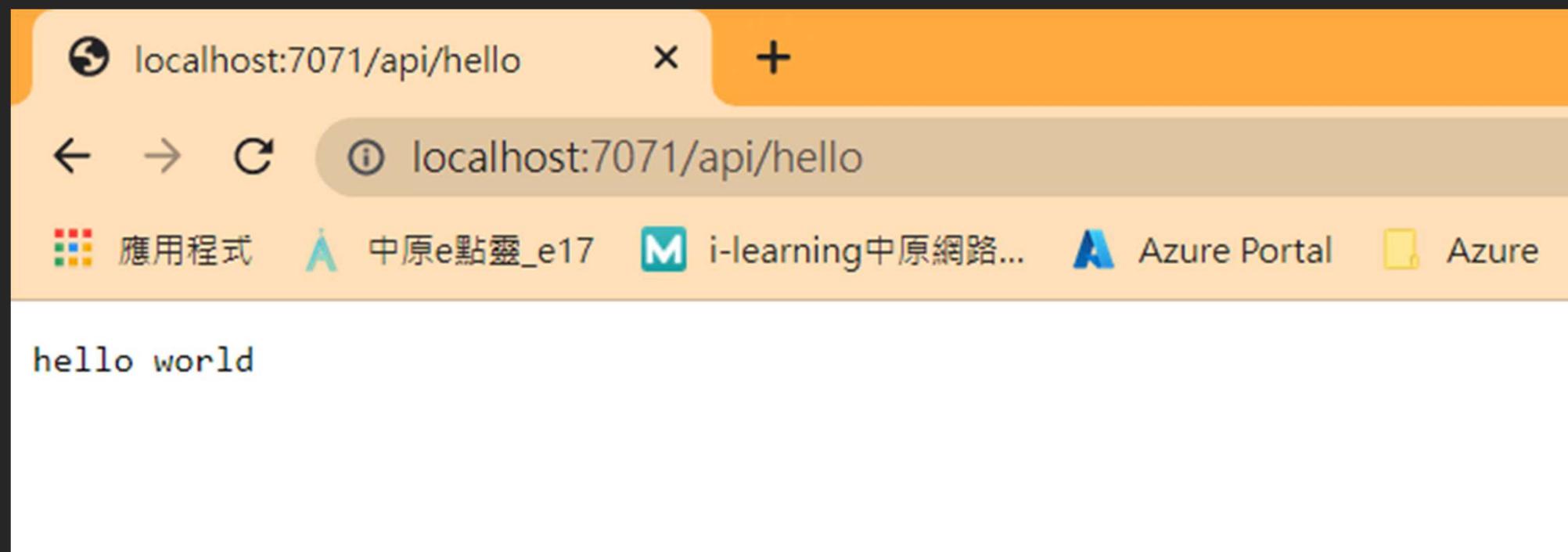
master\* Go 1.17.3 Live Share Azure: 10727211@O365st.cyu.edu.tw [Azurite] [Azurite Table Service] [Azurite Queue Service]

# Run the app - 3

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure under "PART1". The "server.go" file is selected.
- Editor:** The "server.go" file is open, displaying Go code for an Azure Function. The code includes imports for fmt, io/ioutil, log, net/http, and os, and defines a main function that sets up a custom handler port if specified via environment variable or defaults to 8080, then creates a mux and handles the "/api/hello" route.
- Terminal:** A terminal window at the bottom shows the command "func start" being run, which starts the Azure Functions Core Tools. The output shows the server listening on port 58639 and a worker process starting. It also logs host lock lease acquisition and the execution of the "hello" function.
- Status Bar:** Shows the current branch is "master", the Go version is 1.17.3, and the commit hash is 10727211@O365st.cycu.edu.tw.

## Run the app - 4



## Part2

切換至Part2目錄

Cmd :

Code <path>

Eg: code .

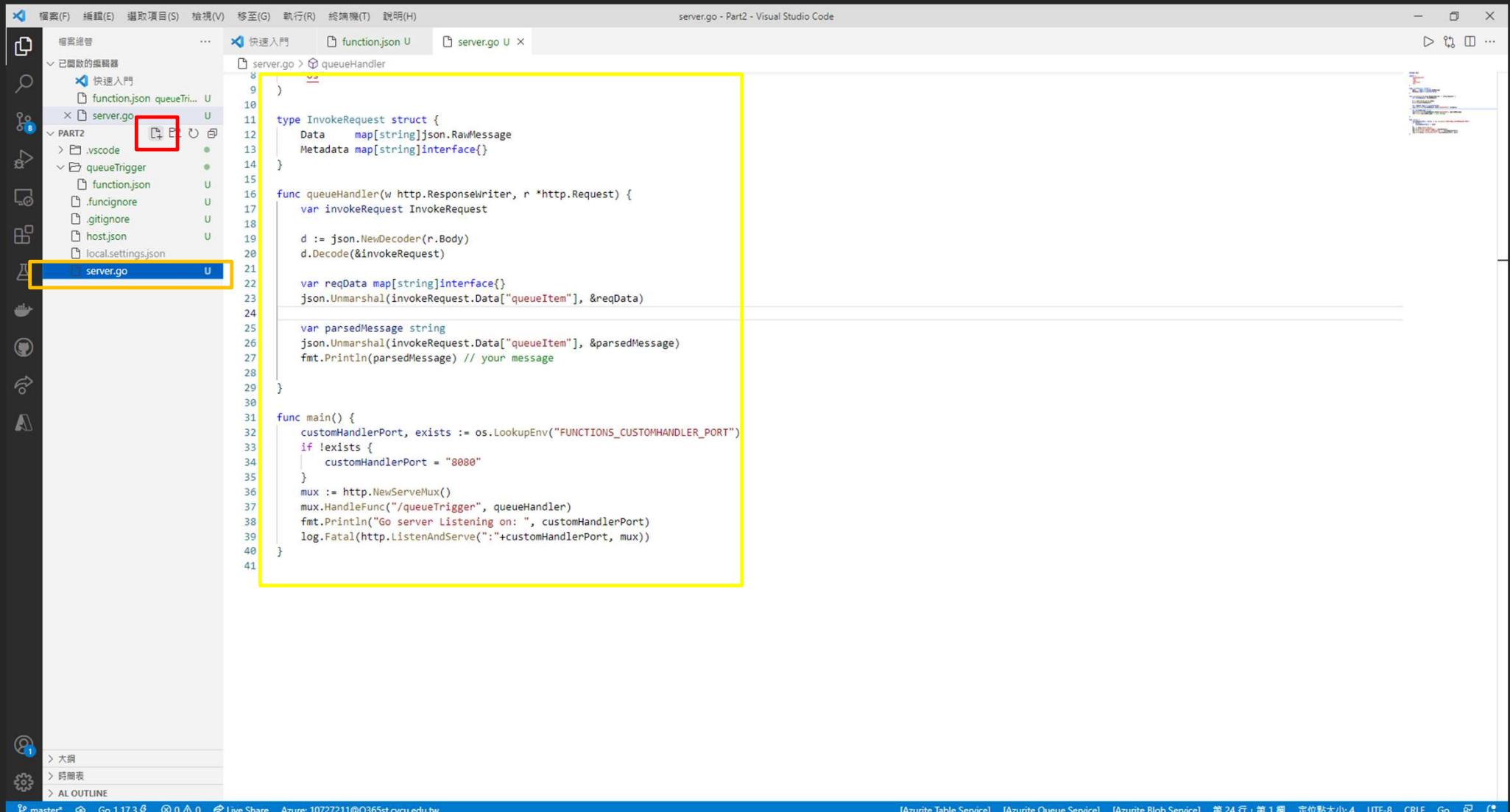
Eg: code.\Part2\

```
PS C:\Users\phil8\Desktop\Azure\1129-go-demo>
PS C:\Users\phil8\Desktop\Azure\1129-go-demo>
PS C:\Users\phil8\Desktop\Azure\1129-go-demo> code .\Part2\
```

## Scaffold the app

	<b>Azure Functions: Create New Project</b>
Select the folder that will contain your function project	{Part2}
Select a language	Custom Handler
Select a template for your first function	HttpTrigger
Provide a function name	queueTrigger
授權等級	Anonymous

# Create the app - 1



```
server.go - Part2 - Visual Studio Code
server.go > queueHandler

8
9
10
11 type InvokeRequest struct {
12     Data map[string]json.RawMessage
13     Metadata map[string]interface{}
14 }
15
16 func queueHandler(w http.ResponseWriter, r *http.Request) {
17     var invokeRequest InvokeRequest
18
19     d := json.NewDecoder(r.Body)
20     d.Decode(&invokeRequest)
21
22     var reqData map[string]interface{}
23     json.Unmarshal(invokeRequest.Data["queueItem"], &reqData)
24
25     var parsedMessage string
26     json.Unmarshal(invokeRequest.Data["queueItem"], &parsedMessage)
27     fmt.Println(parsedMessage) // your message
28
29 }
30
31 func main() {
32     customHandlerPort, exists := os.LookupEnv("FUNCTIONS_CUSTOMHANDLER_PORT")
33     if !exists {
34         customHandlerPort = "8080"
35     }
36     mux := http.NewServeMux()
37     mux.HandleFunc("/queueTrigger", queueHandler)
38     fmt.Println("Go server Listening on: ", customHandlerPort)
39     log.Fatal(http.ListenAndServe(": "+customHandlerPort, mux))
40 }
41 
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure under the 'PART2' folder. The 'server.go' file is selected and highlighted with a yellow box. A red box highlights the 'PART2' folder itself.
- Code Editor:** The main area displays the 'server.go' file content. The code defines a struct for 'InvokeRequest' and a function 'queueHandler' that decodes JSON from the request body into this struct. It then extracts the 'queueItem' field and unmarshals it into a string variable 'parsedMessage'. Finally, it prints the value of 'parsedMessage'. The 'main' function starts an HTTP server on port 8080 using the 'queueHandler' function as a route handler.
- Bottom Status Bar:** Shows the current branch ('master'), Go version ('1.17.3'), and other status information like 'Live Share' and connection details ('Azure: 10727211@O365st.cycu.edu.tw').
- Bottom Taskbar:** Includes links for 'Azurite Table Service', 'Azurite Queue Service', 'Azurite Blob Service', and file navigation options like '第 24 行, 第 1 欄', '定位點大小: 4', 'UTF-8', 'CRLF', 'Go', and 'Find'.

# Create the app - 2

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like `function.json`, `host.json`, `local.settings.json`, and `server.go`.
- Code Editor:** The `server.go` file is open, containing Go code for handling queue triggers.
- Terminal:** A PowerShell window is open at the bottom, showing the command `go build server.go` being run.

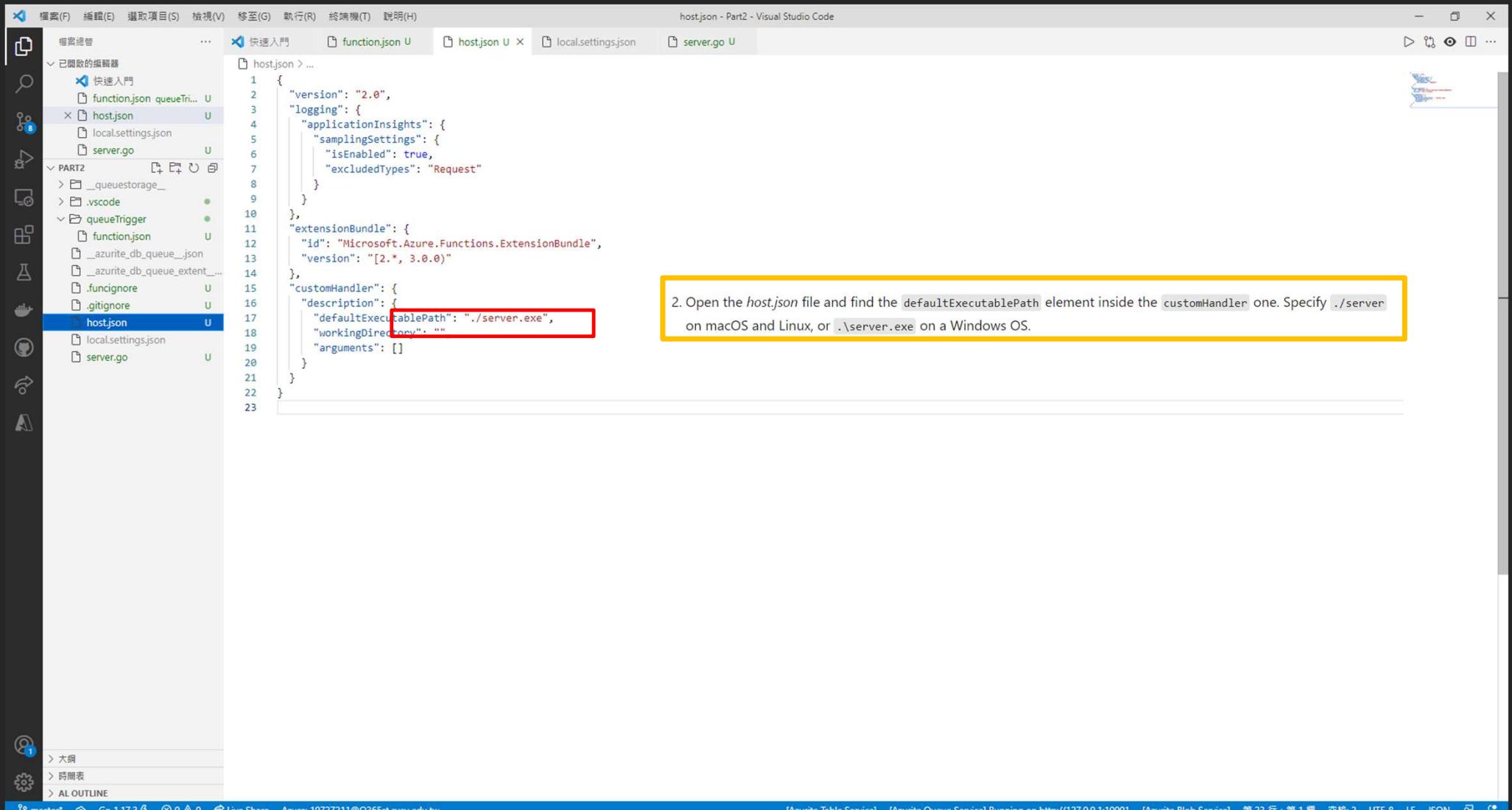
```
server.go - Part2 - Visual Studio Code
server.go > InvokeRequest
8 }
9 }
10 type InvokeRequest struct {
11     Data map[string]json.RawMessage
12     Metadata map[string]interface{}
13 }
14
15
16 func queueHandler(w http.ResponseWriter, r *http.Request) {
17     var invokeRequest InvokeRequest
18
19     d := json.NewDecoder(r.Body)
20     d.Decode(&invokeRequest)
21
22     var reqData map[string]interface{}
23     json.Unmarshal(invokeRequest.Data["queueItem"], &reqData)
24
25     var parsedMessage string
26     json.Unmarshal(invokeRequest.Data["queueItem"], &parsedMessage)
27     fmt.Println(parsedMessage) // your message
28
29 }
30
31 func main() {
32 }
```

Windows PowerShell  
Copyright (C) Microsoft Corporation. 著作權所有，並保留一切權利。  
請嘗試新的跨平台 PowerShell <https://aka.ms/pscore6>

```
PS C:\Users\philip\Desktop\Azure\1129-go-demo\Part2> go build server.go
```

底部状态栏显示：master\* Go 1.17.3 Live Share Azure: 10727211@O365st.cycu.edu.tw [Azurite Table Service] [Azurite Queue Service] Running on http://127.0.0.1:10001 [Azurite Blob Service] 第 11 行, 第 28 檔 定位點大小: 4 UTF-8 CRLF Go

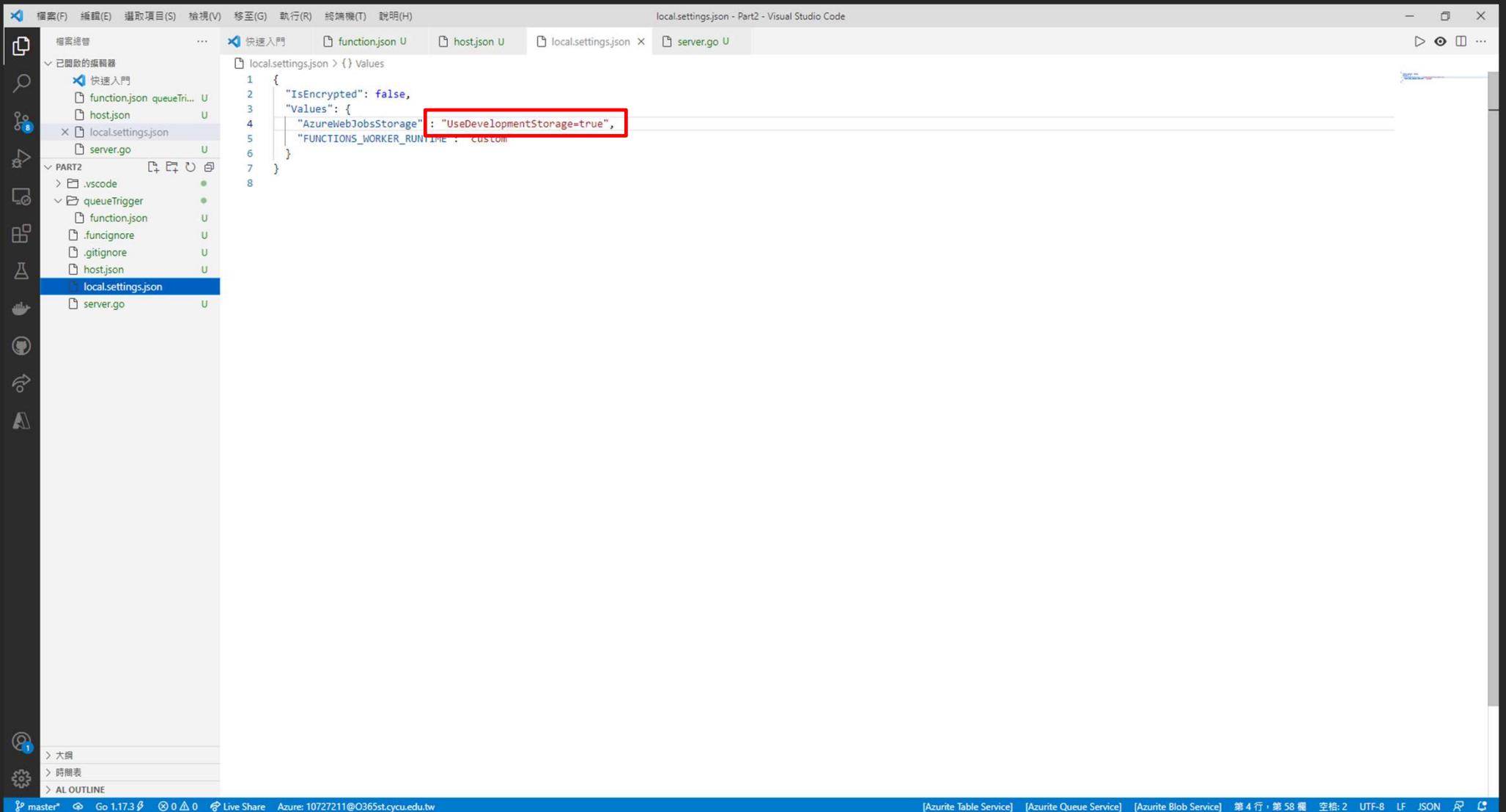
# Configure the environment - 1



```
host.json > ...
1  {
2    "version": "2.0",
3    "logging": {
4      "applicationInsights": {
5        "samplingSettings": {
6          "isEnabled": true,
7          "excludedTypes": "Request"
8        }
9      }
10 },
11 "extensionBundle": {
12   "id": "Microsoft.Azure.Functions.ExtensionBundle",
13   "version": "[2. *, 3.0.0)"
14 },
15 "customHandler": {
16   "description": {
17     "defaultExecutablePath": "./server.exe",
18     "workingDirectory": ""
19   }
20 },
21 }
22 }
```

2. Open the `host.json` file and find the `defaultExecutablePath` element inside the `customHandler` one. Specify `./server` on macOS and Linux, or `.\server.exe` on a Windows OS.

# Configure the environment - 2



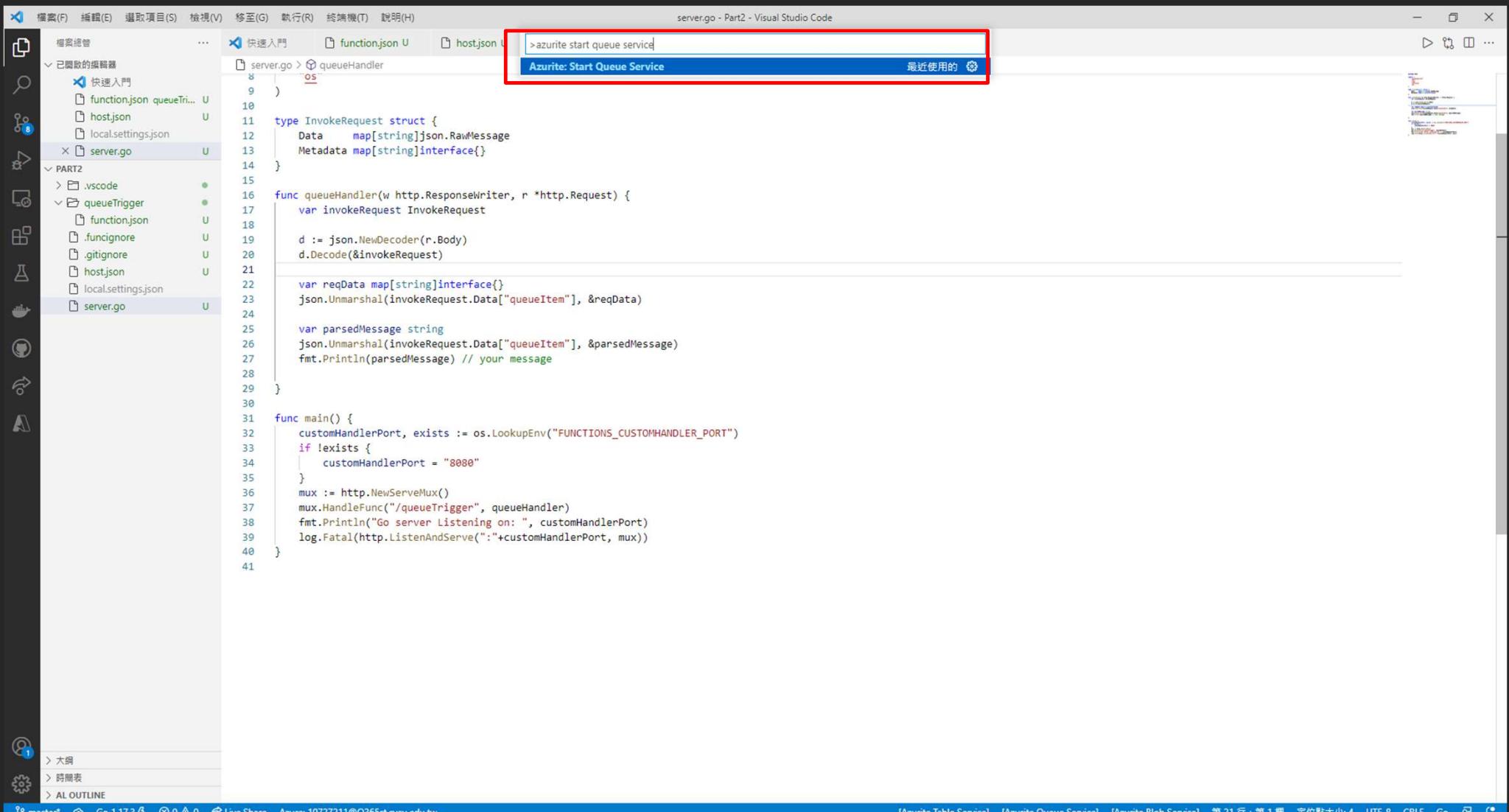
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure with folders like "PART2", "queueTrigger", and files such as "function.json", "host.json", "server.go", and "local.settings.json".
- Code Editor:** Displays the content of the "local.settings.json" file. The file contains the following JSON code:

```
1  {
2    "IsEncrypted": false,
3    "Values": {
4      "AzureWebJobsStorage": "UseDevelopmentStorage=true",
5      "FUNCTIONS_WORKER_RUNTIME": "CUSTOM"
6    }
7 }
```

The line "AzureWebJobsStorage": "UseDevelopmentStorage=true" is highlighted with a red rectangular box.
- Bottom Status Bar:** Shows the current branch ("master"), Go version ("Go 1.17.3"), Live Share status, and the Azure connection string ("Azure: 10727211@O365st.cycu.edu.tw").
- Bottom Right:** Includes links for [Azurite Table Service], [Azurite Queue Service], and [Azurite Blob Service].

# Configure the environment - 3



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure with files like `function.json`, `host.json`, and `server.go`.
- Code Editor:** Displays the `server.go` file containing Go code for a queue trigger function.
- Command Palette:** A search bar at the top right contains the text `>azurite start queue service`. Below it, a list of suggestions includes `Azurite: Start Queue Service`, which is highlighted with a red rectangle.
- Bottom Status Bar:** Shows the current branch (`master`), commit hash (`Ga 1.173`), and other status indicators.

# Configure the environment - 4

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "PART2".
- Code Editor:** The "server.go" file is open, containing Go code for an Azure Function. The code defines a struct for InvokeRequest and implements a queueHandler function. It also contains a main() function that starts a custom handler on port 10001 if no environment variable is set.
- Terminal:** Shows the command "master\* Go 1.17.3 beta" and other status indicators.
- Status Bar:** Shows "Live Share" and "Azure: 10727211@O365st:cycu.edu.tw".
- Bottom Status Bar:** Shows "[Azurite Table Service] [Azurite Queue Service] Running on http://127.0.0.1:10001 [Azurite Blob Service]" and "第 11 行, 第 28 檔 定位點大小: 4 UTF-8 CRLF Go ⌂".
- Callout:** A yellow callout box highlights the text "Port : 10001".
- Message Bar:** A message bar at the bottom right indicates "Azurite Queue Service successfully listens on http://127.0.0.1:10001".

```
server.go - Part2 - Visual Studio Code
server.go > InvokeRequest
8
9
10
11 type InvokeRequest struct {
12     Data map[string]json.RawMessage
13     Metadata map[string]interface{}
14 }
15
16 func queueHandler(w http.ResponseWriter, r *http.Request) {
17     var invokeRequest InvokeRequest
18
19     d := json.NewDecoder(r.Body)
20     d.Decode(&invokeRequest)
21
22     var reqData map[string]interface{}
23     json.Unmarshal(invokeRequest.Data["queueItem"], &reqData)
24
25     var parsedMessage string
26     json.Unmarshal(invokeRequest.Data["queueItem"], &parsedMessage)
27     fmt.Println(parsedMessage) // your message
28
29 }
30
31 func main() {
32     customHandlerPort, exists := os.LookupEnv("FUNCTIONS_CUSTOMHANDLER_PORT")
33     if !exists {
34         customHandlerPort = "8080"
35     }
36     mux := http.NewServeMux()
37     mux.HandleFunc("/queueTrigger", queueHandler)
38     fmt.Println("Go server Listening on: ", customHandlerPort)
39     log.Fatal(http.ListenAndServe(": "+customHandlerPort, mux))
40 }
41
```

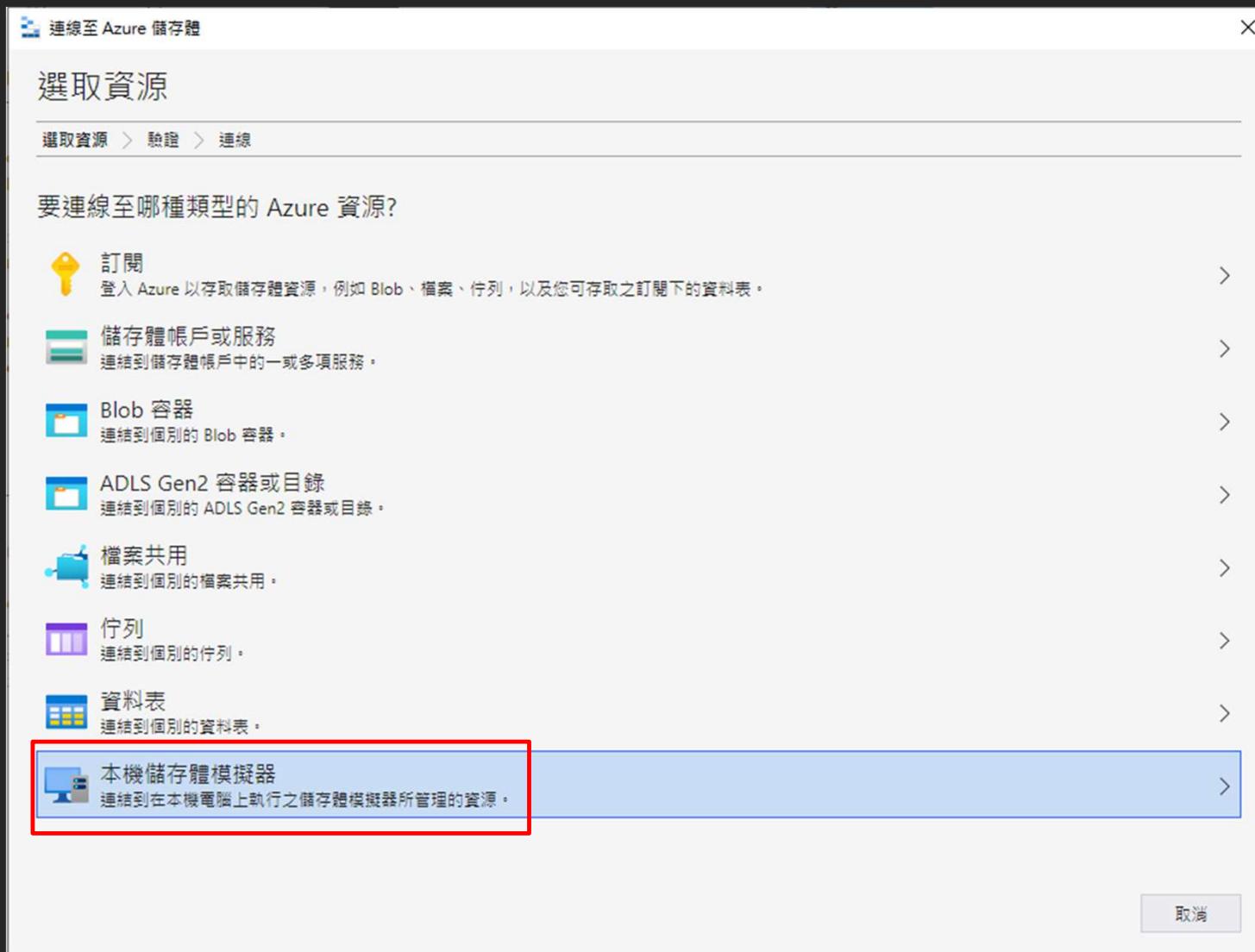
## Configure the environment - 5

Microsoft Azure Storage Explorer

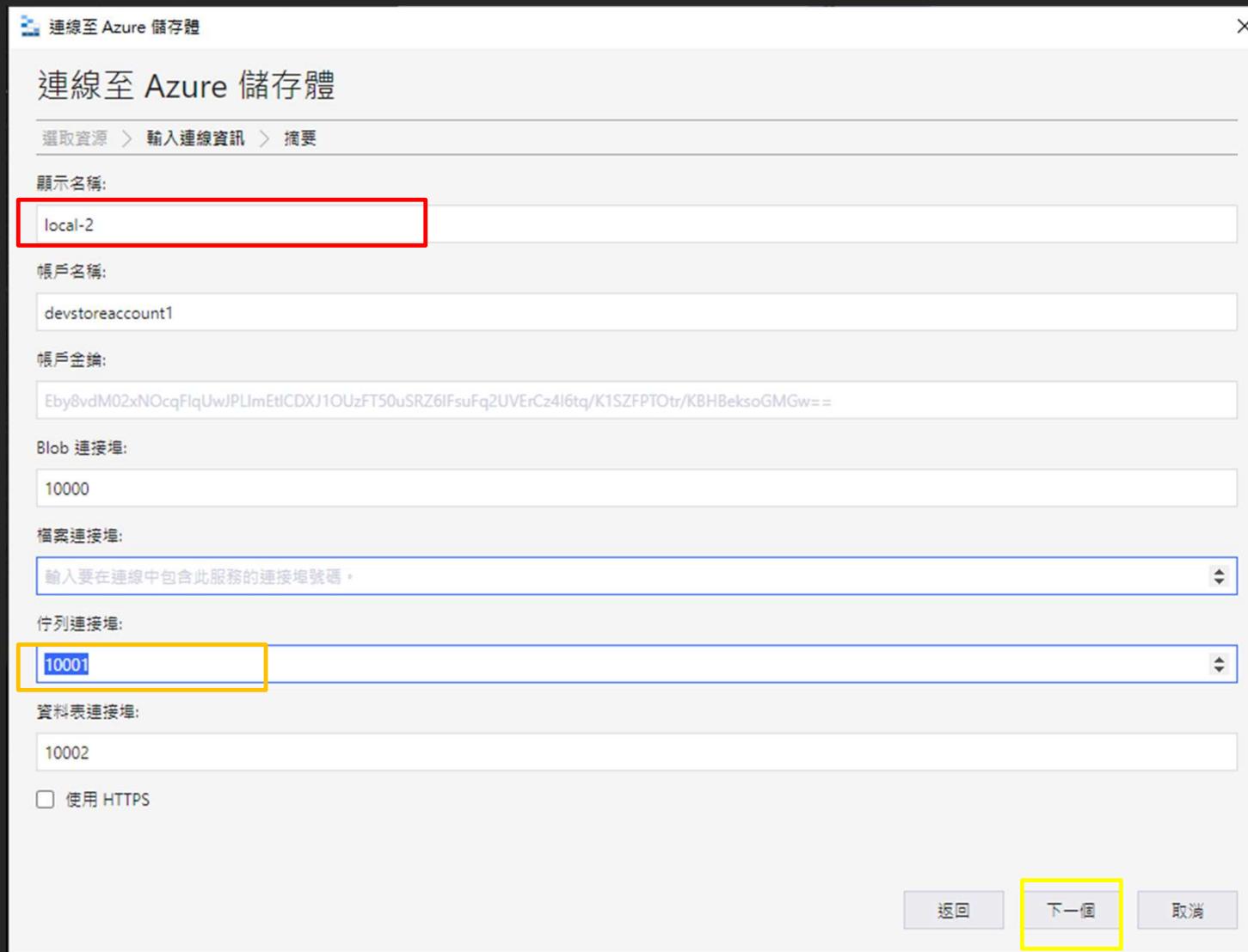


v1.21.3

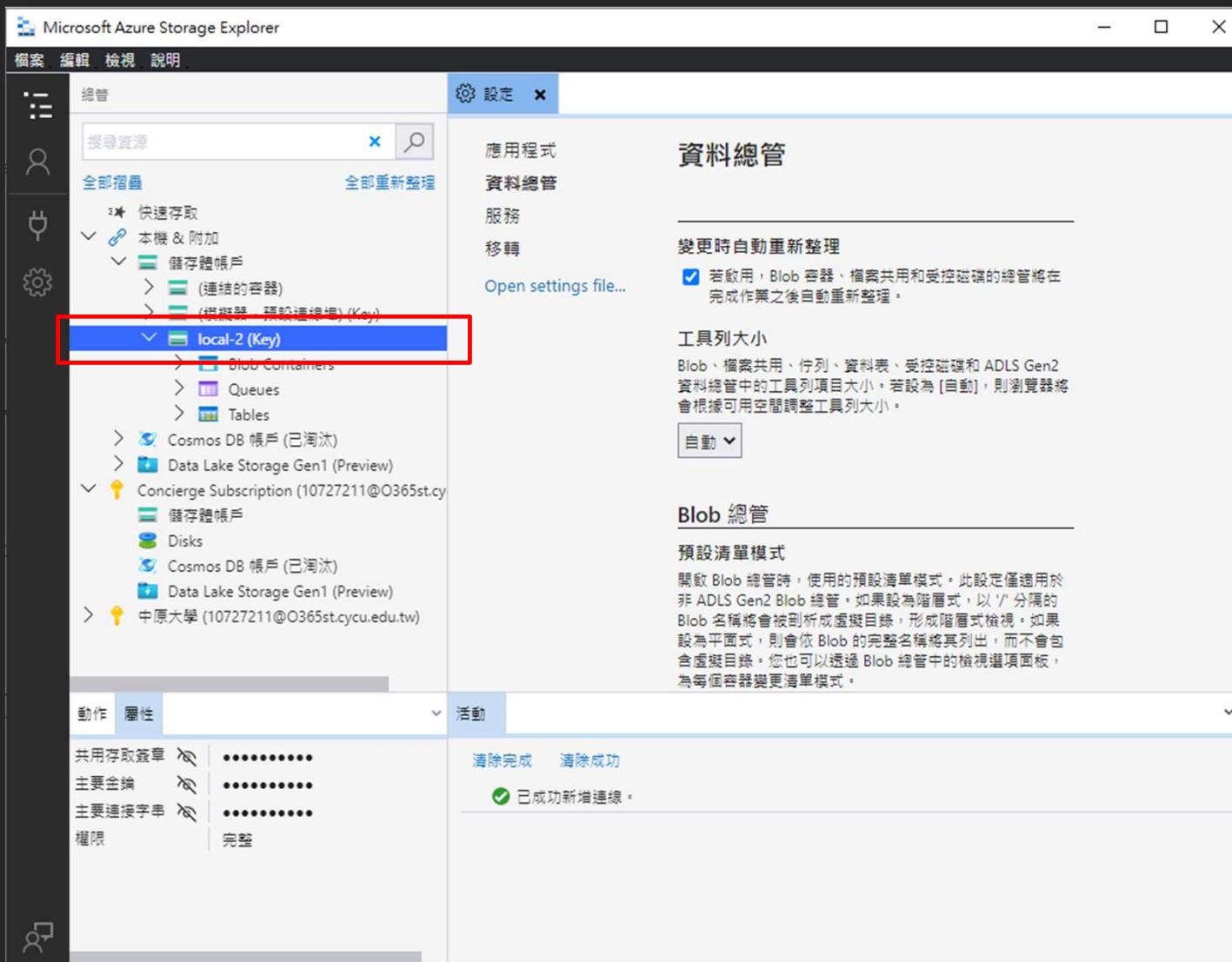
# Configure the environment - 6



# Configure the environment - 7



# Configure the environment - 8



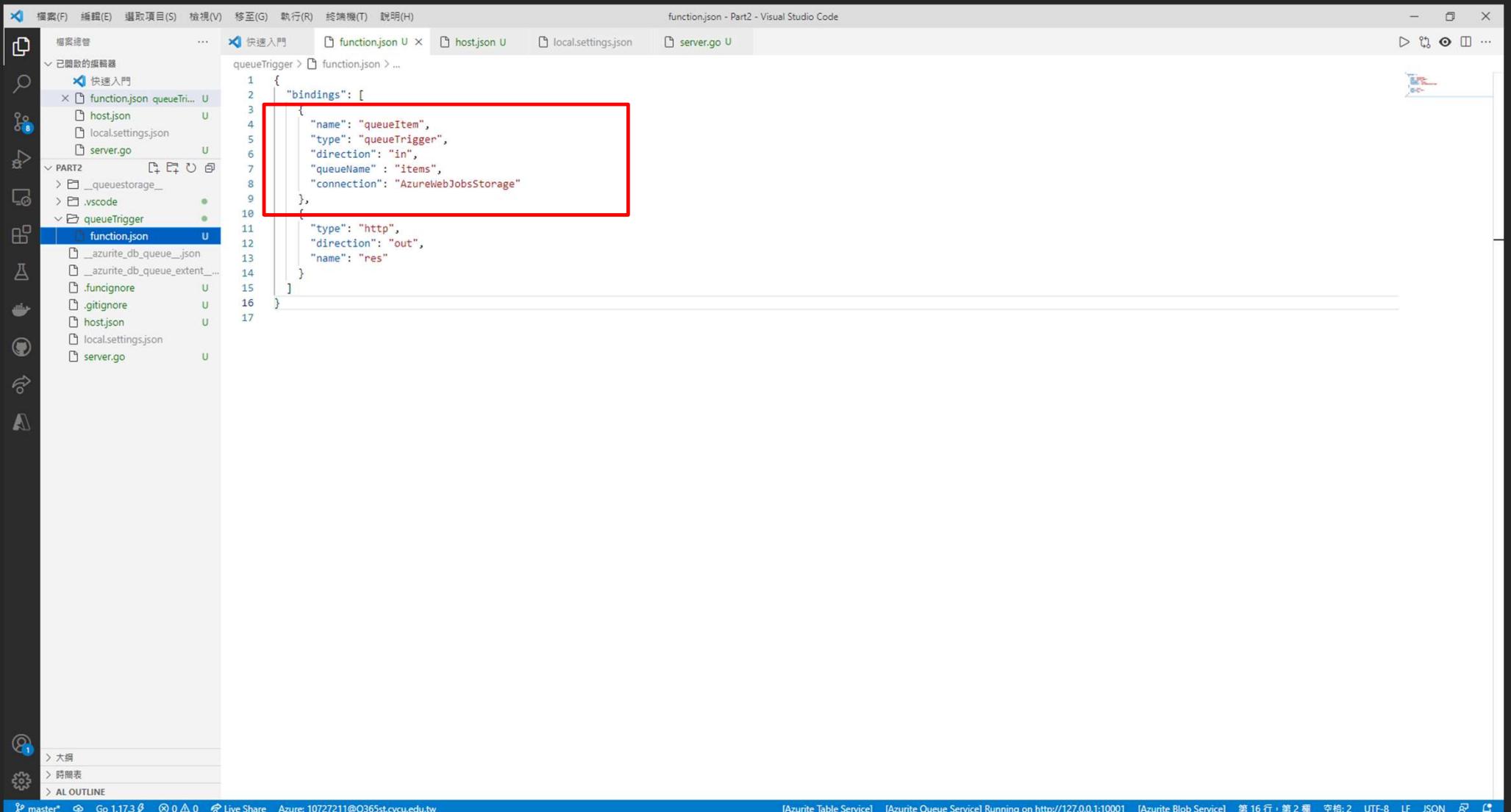
# Configure the environment - 9



# Configure the environment - 10



# Configure the environment - 11



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure with folders like "PART1", "queueTrigger", and files such as "function.json", "host.json", "local.settings.json", and "server.go".
- Code Editor:** Displays the "function.json" file content. A red box highlights the "queueTrigger" section.
- Terminal:** At the bottom, it shows the command "master\* Go 1.17.3 ⚡ ⚡ 0 ▲ 0 Live Share Azure: 10727211@O365st.cycu.edu.tw".
- Status Bar:** At the bottom right, it shows "[Azurite Table Service] [Azurite Queue Service] Running on http://127.0.0.1:10001 [Azurite Blob Service] 第 16 行, 第 2 檔 空格: 2 UFT-8 LF JSON ⚡ 🔍".

```
function.json - Part2 - Visual Studio Code
queueTrigger > function.json > ...
1 {
2   "bindings": [
3     {
4       "name": "queueItem",
5       "type": "queueTrigger",
6       "direction": "in",
7       "queueName": "items",
8       "connection": "AzureWebJobsStorage"
9     },
10    {
11      "type": "http",
12      "direction": "out",
13      "name": "res"
14    }
15  ]
16 }
```

# Run the app - 1

The screenshot shows a Visual Studio Code interface with the following details:

- Project Explorer:** Shows the project structure under "已開啟的編輯器" (Open Editors). It includes "快速入門" (Quick Start) and "PART2" sections. "PART2" contains "queueTrigger" and "function.json" files.
- Code Editor:** Displays the content of "function.json". The code defines a queue trigger named "queueItem" and a response binding named "res".

```
1  {
2    "bindings": [
3      {
4        "name": "queueItem",
5        "type": "queueTrigger",
6        "direction": "in",
7        "queueName": "items",
8        "connection": "AzureWebJobsStorage"
9      },
10     {
11       "type": "http",
12       "direction": "out",
13       "name": "res"
14     }
15   ]
16 }
```
- Terminal:** A Windows PowerShell window at the bottom shows the command "func start" being run. The output indicates the application is listening on port 56075 and the worker process has started.

```
PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part2> func start

Azure Functions Core Tools
Core Tools Version: 3.0.3904 Commit hash: c345f7140a8f968c5dbc621f8a8374d8e3234206 (64-bit)
Function Runtime Version: 3.3.1.0

Functions:

queueTrigger: queueTrigger

For detailed output, run func with --verbose flag.
[2021-11-29T14:24:51.231Z] Go server Listening on: 56075
[2021-11-29T14:24:51.639Z] Worker process started and initialized.
```
- Bottom Status Bar:** Shows the current branch is "master", the file version is "Go 1.17.3", and the status bar also includes "Live Share" and connection information.

# Run the app - 2

Microsoft Azure Storage Explorer

檔案 編輯 檢視 說明

總管 設定 items 新增訊息 檢視訊息 清除佇列訊息 將訊息重新排入佇列 移動訊息 清除佇列 重新整理

搜尋資源 全部重新整理

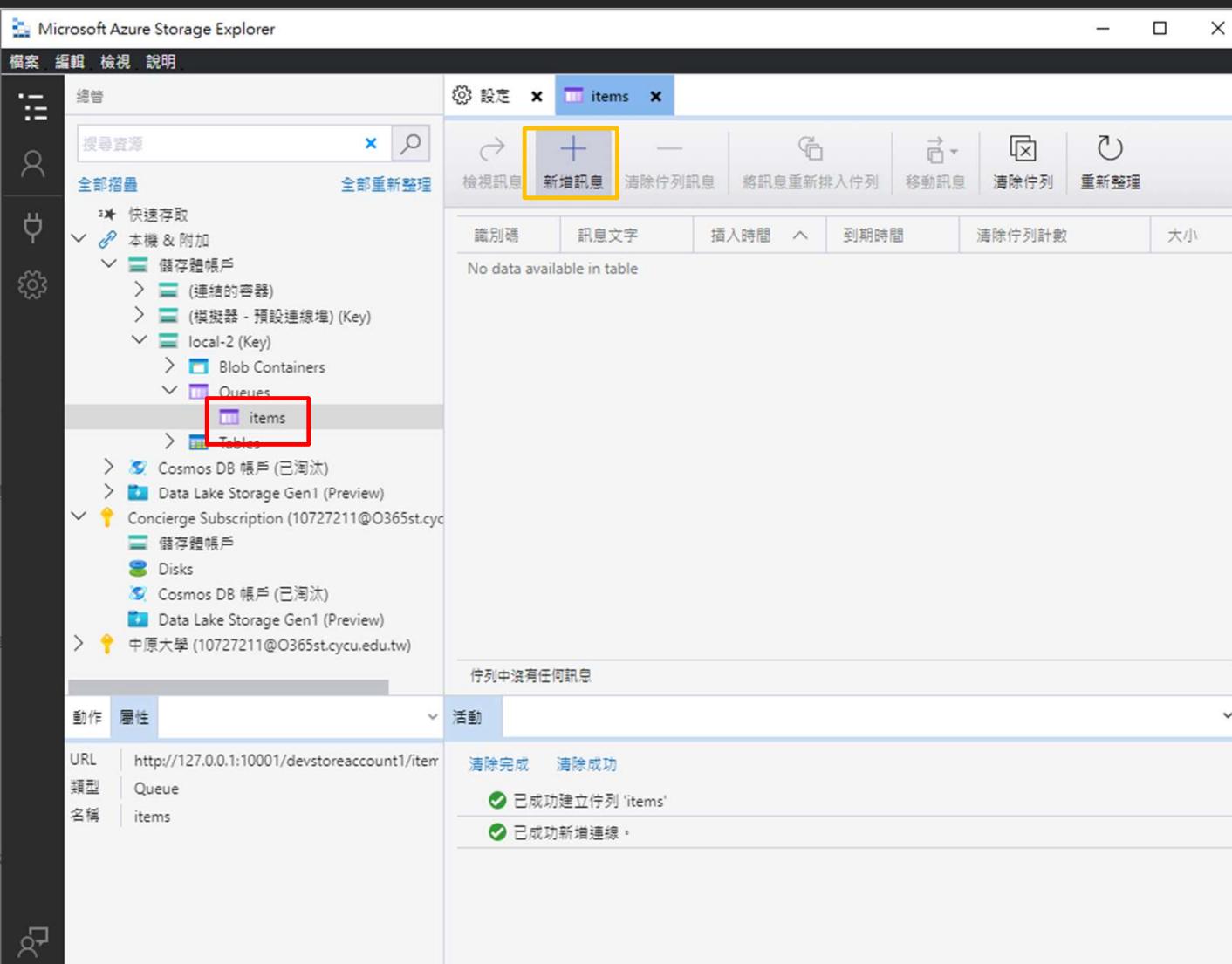
全部擴展 快速存取 本機 & 附加 儲存體帳戶 (連接的容器) (模擬器 - 預設連線埠) (Key) local-2 (Key) Blob Containers Queues items Tables Cosmos DB 帳戶 (已淘汰) Data Lake Storage Gen1 (Preview) Concierge Subscription (10727211@O365st.cyc 儲存體帳戶 Disks Cosmos DB 帳戶 (已淘汰) Data Lake Storage Gen1 (Preview) 中原大學 (10727211@O365st.cycu.edu.tw)

No data available in table

動作 屬性 活動

URL: http://127.0.0.1:10001/devstoreaccount1/item  
類型: Queue  
名稱: items

活動: 清除完成 清除成功  
已成功建立佇列 'items'  
已成功新增連線。



Microsoft Azure Storage Explorer

### 加入訊息

訊息文字:

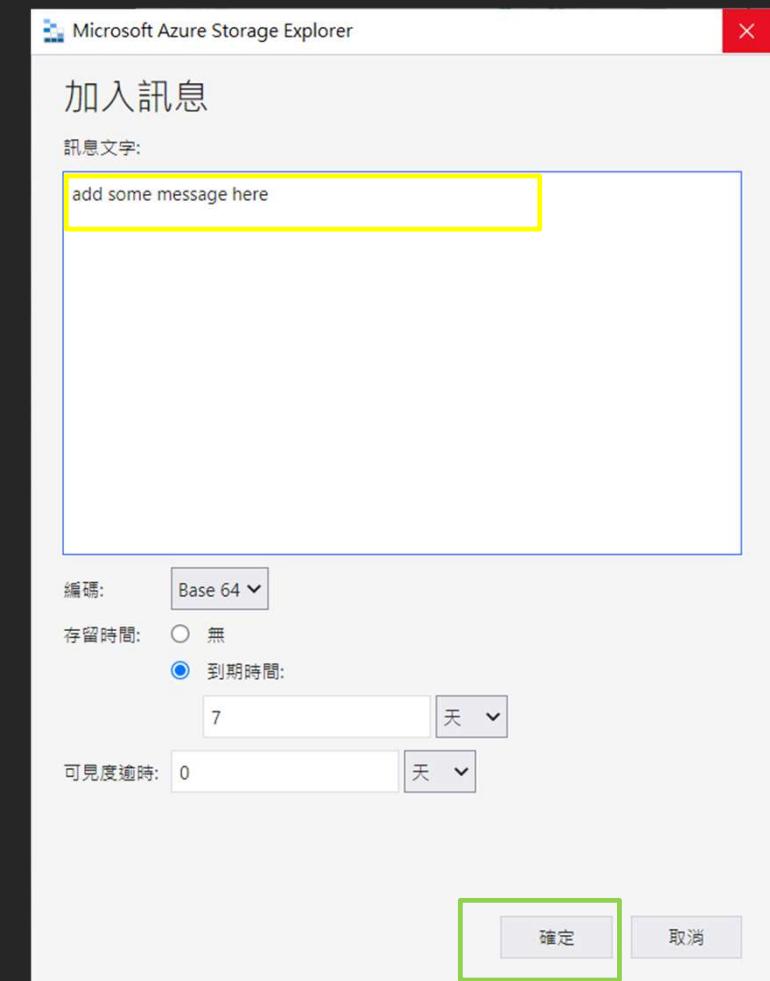
add some message here

編碼: Base 64

存留時間:  無  到期時間:  
7 天

可見度逾時: 0 天

確定 取消



# Run the app - 3

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "PART2".
  - function.json (selected)
  - host.json
  - local.settings.json
  - server.go
- Code Editor:** Displays the contents of function.json.

```
1 {
2   "bindings": [
3     {
4       "name": "queueItem",
5       "type": "queueTrigger",
6       "direction": "in",
7       "queueName": "items",
8       "connection": "AzureWebJobsStorage"
9     },
10    {
11      "type": "http",
12      "direction": "out",
13      "name": "res"
14    }
15  ]
16 }
```
- Terminal:** Shows the Windows PowerShell window with the following output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. 著作權所有，並保留一切權利。
請嘗試新的跨平台 PowerShell https://aka.ms/pscore6

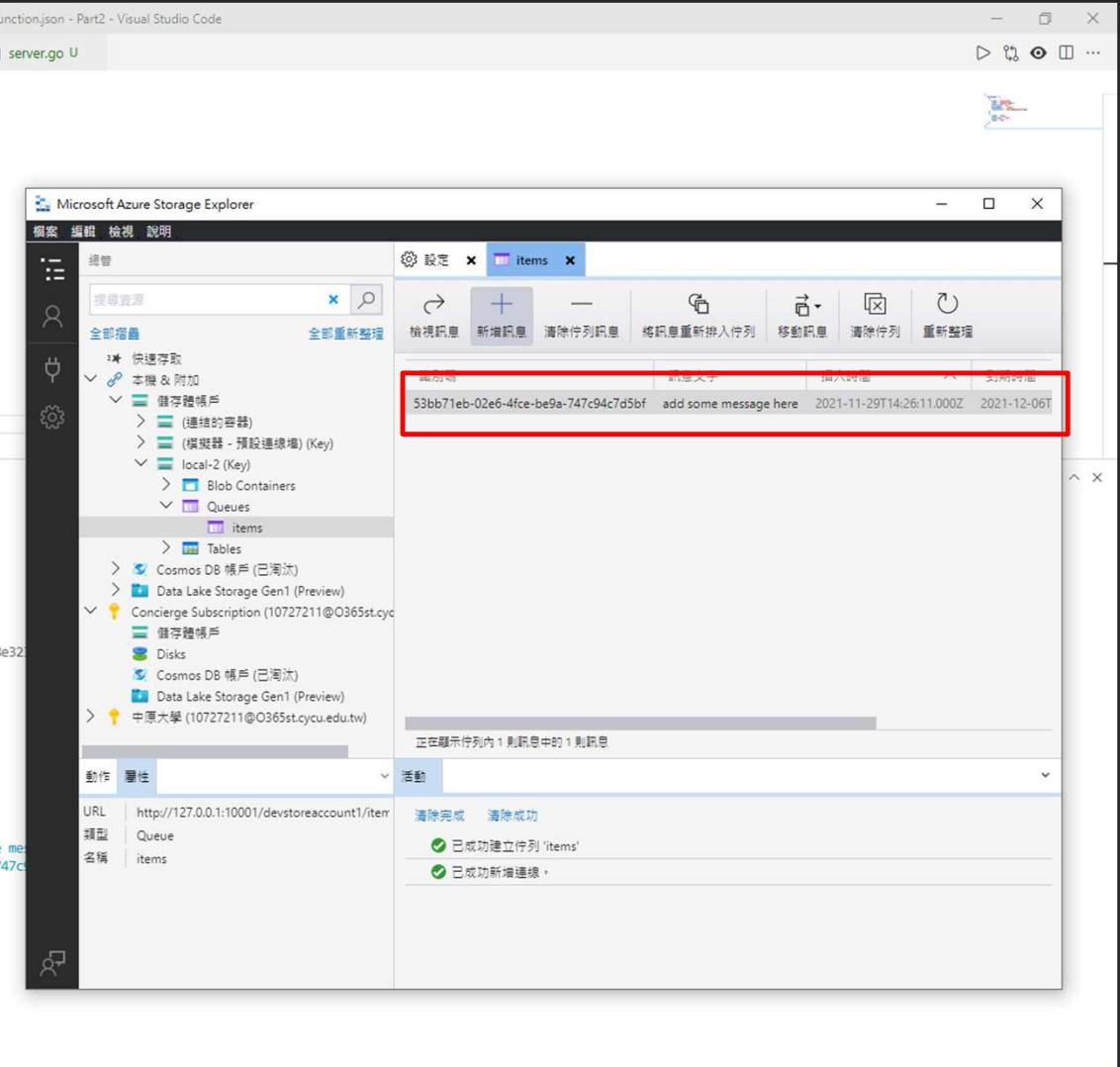
PS C:\Users\phil8\Desktop\Azure\1129-go-demo\Part2> func start

Azure Functions Core Tools
Core Tools Version: 3.0.3904 Commit hash: c345f7140a8f968c5dbc621f8a8374
Function Runtime Version: 3.3.1.0

Functions:

queueTrigger: queueTrigger

For detailed output, run func with --verbose flag.
[2021-11-29T14:24:51.231Z] Go server Listening on: 56075
[2021-11-29T14:24:51.639Z] Worker process started and initialized.
[2021-11-29T14:26:13.418Z] Executing 'Functions.queueTrigger' (Reason='New queue message')
[2021-11-29T14:26:13.459Z] Trigger Details: messageId: 350071e0-0260-47ce-bea
[2021-11-29T14:26:13.459Z] "add some message here"
```





Thanks for  
your reading!