

Uso de Funciones de Azure con Flow y PowerApps

En este lab se indica como crear una Función de Azure que expone su API por medio de un OpenAPI del Azure API Management, que, a su vez, puede ser utilizado desde Flow y/o PowerApps. El ejemplo a continuación es un sistema de reserva de salas de conferencias en una empresa: empleados pueden reservar una sala de conferencias desde una Lista de SharePoint; un Flow acoplado a la Lista pasa los datos de la reserva a una Función de Azure que calcula si la reservación es posible, o si el sitio está ocupado y retorna una indicación al respecto al Flow, el que retransmite la información a SharePoint y al usuario. Todo el sistema de reserva se puede utilizar también desde una aplicación independiente de PowerApps. El algoritmo para determinar si una sala está ocupada o no, no está implementado, solamente se ha simulado por medio de un generador random. Pero el ejemplo indica claramente el potencial que ofrece la combinación de los cuatro sistemas.

Creación de la infraestructura en Azure

1. Abra el portal de Azure (<https://portal.azure.com>) utilizando una cuenta con suficientes derechos para crear servicios. Una cuenta temporal puede ser creada desde el mismo sitio.
2. Cree un Grupo de Recursos: abra la sección de Grupos de Recursos (Resource Groups) y utilice el botón de "+Add" para crear uno nuevo.
3. Después de que el grupo ha sido creado, ábralo y agregue un servicio de storage: use el botón de "+Add" en el grupo de recursos, introduzca en la casilla de búsqueda "storage account", haga clic sobre el bloque de resultados correspondiente y haga clic sobre "Create". Introduzca los datos pedidos sobre la suscripción y el Grupo de Recursos creado anteriormente. Defina un nombre para el servicio y la localización de su centro de datos. Seleccione "Standard" para "Performance", "StorageV2" en "Account kind", "Read-acces geo-redundant" en "Replication" y "Hot" en "Access tier". Use el botón de "Review + create" y luego "Create" para crear el servicio.

Create storage account

Project details

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

* Subscription	Visual Studio Ultimate with MSDN	▼
* Resource group	CompartiMOSS_FunctionFlowPowerapps	▼

[Create new](#)

Instance details

The default deployment model is Resource Manager, which supports the latest Azure features. You may choose to deploy using the classic deployment model instead. [Choose classic deployment model](#)

* Storage account name ⓘ	storagefunctflowpowapp	✓
* Location	(Europe) West Europe	▼
Performance ⓘ	<input checked="" type="radio"/> Standard <input type="radio"/> Premium	
Account kind ⓘ	StorageV2 (general purpose v2)	▼
Replication ⓘ	Read-access geo-redundant storage (RA-GRS)	▼
Access tier (default) ⓘ	<input type="radio"/> Cool <input checked="" type="radio"/> Hot	

Review + create

< Previous

Next : Advanced >

4. Cree un App Service para alojar la Función de Azure: Regrese al Resource Group creado en el punto 2, use el botón de "+Add" y busque esta vez por "function app". Haga clic sobre el bloque correspondiente en los resultados y también sobre el botón de "Create". Defina el nombre para el servicio en la siguiente ventana, seleccione la suscripción y el Grupo de Recursos creado anteriormente. Seleccione "Windows" como "OS", "Consumption Plan" en "Hosting Plan", la localización de su centro de datos, ".NET Core" en "Runtime Stack" y el storage creado anteriormente. Si desea, remueva el servicio de "Application Insights" al final.

Function App



Create

* App name

functionappflowpowerapp



.azurewebsites.net

* Subscription

Visual Studio Ultimate with MSDN



* Resource Group



Create new



Use existing

CompartiMOSS_FunctionFlowPowerapps



* OS

Windows

Linux

* Hosting Plan

Consumption Plan



* Location

West Europe



* Runtime Stack

.NET Core



* Storage



Create new



Use existing

compfunflpappstorage0



Create

[Automation options](#)

5. Cree un servicio de API Management que es el que se va a encargar de convertir la Función en un recurso de OpenAPI: Regrese al Grupo de Recursos, use el botón de "+Add" y busque por "api management". Haga clic sobre el bloque de "API Management" y el botón de "Create". Defina un nombre para el servicio, seleccione la suscripción y el Grupo de Recursos, la localización de su centro de datos y defina un nombre para la organización y su email (estos dos datos no son

validados, pero son obligatorios). Seleccione "Consumption (99.9 SLA, %)" en el "Pricing tier". Si desea, seleccione la creación de un servicio de Application Insights.

API Management service



* Name

apimanagementfunctionflowpowapp



.azure-api.net

* Subscription

Visual Studio Ultimate with MSDN



* Resource group

CompartiMOSS_FunctionFlowPowerapps




[Create new](#)

* Location

West Europe



* Organization name 

gavd



* Administrator email 

gavd@whatever.com



Pricing tier ([View full pricing details](#))

Consumption (99.9 SLA, %)



Enable Application Insights

6. Cambie la definición de las funciones a versión 1. Funciones de Azure pueden ser configuradas como versión 1 o 2. La versión 1 funciona con el .NET Framework de Microsoft y la versión 2 con el .NET Core Framework. Como todos los APIs de SharePoint no son (todavía) compatibles con el .NET Core Framework, todas las Funciones de Azure que tengan que trabajar (o que potencialmente vayan a trabajar) con SharePoint tienen que ser de versión 1. Regrese al Grupo de Recursos y haga clic sobre el "App service" creado en el punto 4. Haga clic sobre "Platform features" y luego sobre "Function app settings" (sección "General Settings"). En la sección de "Runtime version" seleccione "-1".
7. Cree la función. Desde la ventana del App service, haga clic sobre el botón "+" al lado derecho de "Functions" para crear una nueva función. Haga clic sobre "C#" en el bloque "HTTP Trigger". Seleccione "C#" en "Language", defina un nombre para la función y seleccione "Function" en "Authorization level". Cree la nueva función.



HTTP trigger

New Function

Language:

Name:

HTTP trigger

Authorization level

Create

Cancel

8. Codificación de la función. La función es creada con algo de código por defecto que permite capturar los parámetros de entrada, sea que se hayan enviado por medio del QueryString o en el Body de la llamada. En el ejemplo se va a utilizar solamente el QueryString porque Flow y PowerApps tiene problemas haciendo llamadas REST que utilizan el Body para enviar parámetros.

Reemplace todo el código original por el siguiente fragmento:

```
1using System;using System.Net;public static async Task<HttpResponseBody>
Run(HttpRequestMessage req, TraceWriter log){ log.Info("Request for a Conference
Room is arrived"); // Parse query parameters string roomname =
req.GetQueryNameValuePairs().FirstOrDefault(q=> string.Compare(q.Key,
"roomname", true)== 0) .Value; log.Info("roomname = " + roomname); string persons
= req.GetQueryNameValuePairs().FirstOrDefault(q=> string.Compare(q.Key,
"persons", true)== 0) .Value; log.Info("persons = " +
persons); HttpResponseMessage myResponse =
null; if(string.IsNullOrEmpty(roomname)== true || string.IsNullOrEmpty(persons)==
true) { myResponse = req.CreateResponse(HttpStatusCode.BadRequest, "roomname
```

```

and number of persons are obligatory"); } else { // Faking a room algorithm
...    string[] roomState = { "Free", "Occupied" };    Random myRnd = new
Random();    int stateIndex = myRnd.Next(roomState.Length);    myResponse =
req.CreateResponse(HttpStatusCode.OK, roomState[stateIndex]); } log.Info("Returning
- " + myResponse); return myResponse;}

```

2

Testee la función para comprobar que no hay errores de compilación.

9. Configuración del API Management para exponer la función como un OpenAPI.
 Regrese al Grupo de Aplicaciones y haga clic sobre el servicio del API Management creado en el punto 5. En la nueva ventana, haga clic sobre "APIs" (menú vertical izquierdo) y luego sobre el botón de "Function App" en la sección de "Add a new API". En la nueva ventana, use el botón de "Browse" y luego el de "Function App", el que abre una ventana con una lista con todas las Function Apps en la suscripción. Elija la Function App correcta, lo que produce a su vez una lista con las funciones dentro de ella. Seleccione la función creada en el punto 8, y luego el botón "Select". Los nombres para utilizar aparecen automáticamente, extraídos del nombre del Function App, pero pueden ser configurados a discreción.

Nota: Una función puede ser exportada directamente a un servicio de API Management (desde la ventana de "Platform features" del App Service, vínculo "API Management" en la sección "API"), pero se pierde el control sobre nombres y configuración del servicio, por lo que es mejor hacerlo siguiendo este procedimiento. Igualmente, desde esta sección hay un botón para exportar la definición directamente a Flow y PowerApps, pero, como se indicó inicialmente, solamente funciona si tanto Azure como Office 365 comparten el mismo Directorio Activo (AD).

Create from Function App

Basic | [Full](#)

* Function App

compfunflpappfunction0

Browse

* Display name

RoomBooking

* Name

roombooking

API URL suffix

roombooking

Products

No products selected

i To publish the API, you must associate it with a product. [Learn more.](#)

Utilice el botón de "Create". La ventana de "Operations" aparece con toda la configuración del API para la función:

The screenshot displays the Azure API Management console interface. On the left sidebar, there is a search bar for APIs and a filter by tags section. Below this, the 'All APIs' section lists the 'RoomBooking' API. The main area is titled 'REVISION 1' and 'UPDATED Jul 14, 2019, 5:41:36 PM'. It features tabs for 'Design', 'Settings', 'Test', 'Revisions', and 'Change log'. The 'Design' tab is active, showing a 'Search operations' bar and a 'Filter by tags' section. Below this, there is an 'Add operation' button and a list of 'All operations'. The list contains two operations: 'POST BookRoom' and 'GET BookRoom'. The right-hand pane shows the 'Frontend' section and two policy sections: 'Inbound processing' and 'Outbound processing'. Each policy section has a 'Policies' dropdown menu with a 'base' policy selected.

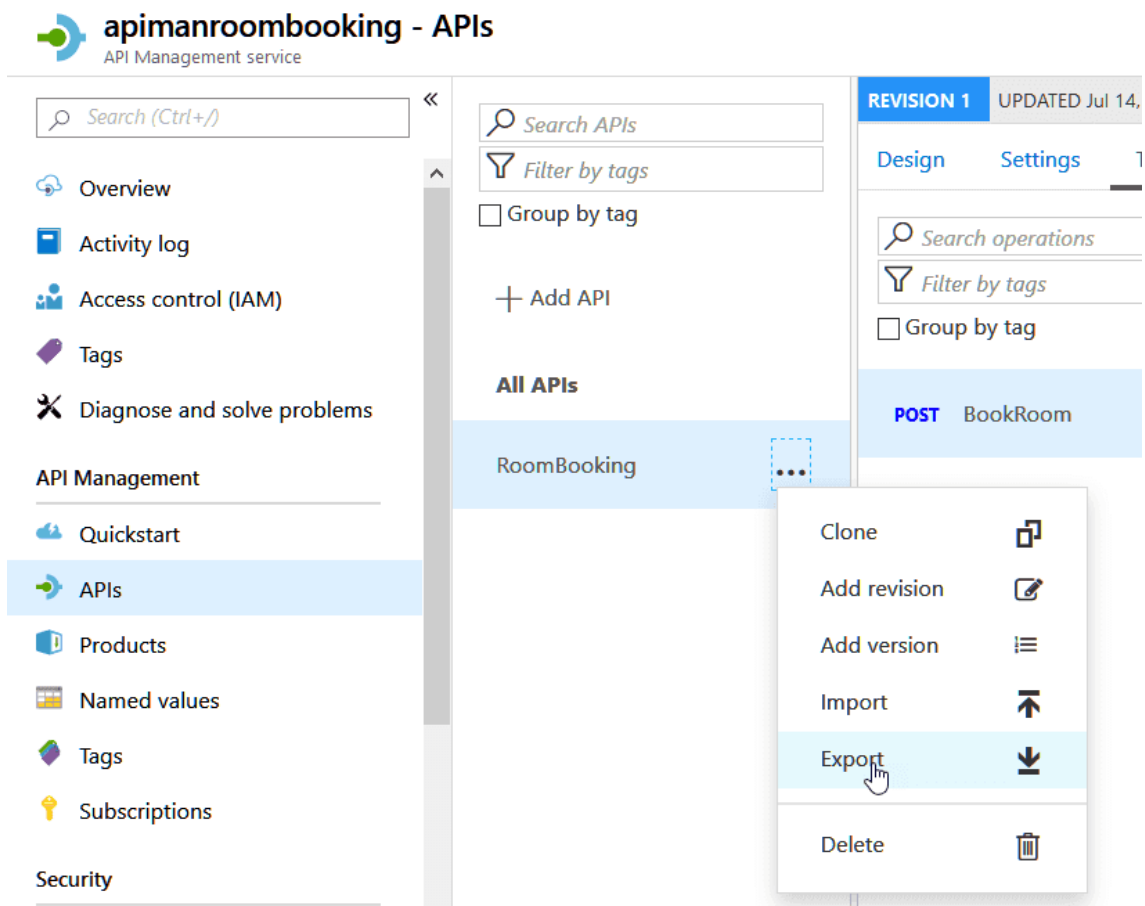
Haga clic sobre el método "POST" (bajo "All operations"). En el panel de "Frontend" haga clic sobre el botón con el lápiz para configurar los parámetros de entrada del QueryString. Seleccione la pestaña de "Query" y agregue (botón "+Add parameter") los nombres de los dos parámetros del QueryString que la función espera encontrar ("roomname" y "persons"). En la misma ventana se puede configurar el tipo de parámetro y un valor por defecto, pero solamente el nombre es obligatorio. Después de guardar los cambios, la ventana de configuración aparece de nuevo mostrando los dos nombres de los parámetros.

Para comprobar que el API funciona correctamente, seleccione la pestaña de "Test" con el método "POST" seleccionado, suministre valores para los dos parámetros, y use el botón de "Send". En la respuesta debe aparecer "Free" o "Occupied". La función presenta su funcionalidad desde este momento al mundo externo en forma de un método REST que es compatible con la definición de OpenAPI que se va a generar. En el siguiente artículo de la serie veremos cómo exportar la definición del método y como utilizarlo en Microsoft Flow.

Para darle más flexibilidad y capacidad de interacción con otros sistemas, Microsoft Flow y PowerApps pueden utilizar "conectores" a procedimientos externos. Azure Functions es el método ideal para crear esa funcionalidad.

Exportar la definición de OpenAPI

1. Abra el portal de Azure (<https://portal.azure.com>) donde se creó el Grupo de Recursos que contiene la función y la definición del API Management.
2. Abra el Grupo de Recursos (Resource Groups) donde se crearon los servicios del sistema.
3. Haga clic sobre el servicio de API Management y luego sobre el vínculo de "APIs" (menú vertical izquierdo). Seleccione el API creado y, haciendo clic sobre el botón de elipse ("..."), seleccione "Export".



Haga clic sobre el botón "OpenAPI v2 (JSON)". La ventana de descargas por defecto de Windows abre para guardar el archivo de descarga localmente. El archivo es en formato JSON y contiene todos los parámetros para poder hacer una llamada al servicio REST, el que, a su vez, manda la llamada a la Función de Azure. El archivo se puede abrir con cualquier editor ASCII.

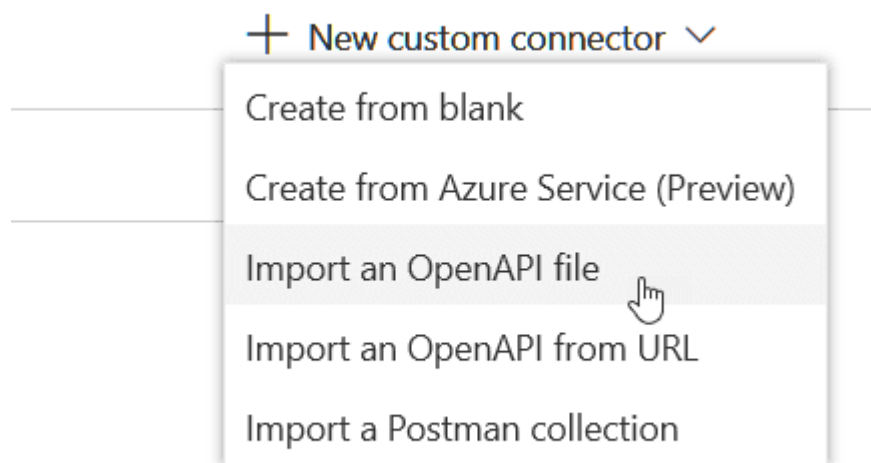
Crear una Lista de SharePoint

4. Para el ejemplo, el flujo se inicia desde una Lista de SharePoint. Desde alguna colección de sitios de SharePoint Online cree una Lista Personalizada con dos campos de texto sencillo, uno llamado "RoomName" y el otro "Persons".

Conectar el servicio a****PowerAutomate

5. Abra el sitio de PowerAutomate desde el portal de Office (<https://portal.office.com>), o desde el portal de PowerAutomate mismo (<https://powerautomate.microsoft.com>) y lóguese con credenciales que tengan permisos suficientes para crear y utilizar flujos.
6. Expanda el menú de "Data" (menú vertical al lado izquierdo) y haga clic sobre "Custom connectors". Use el botón "+New custom connector" en la esquina

superior derecha para expandir un nuevo menú. Haga clic sobre "Importar an




OpenAPI file"

Defina un nombre para el conector y seleccione el archivo descargado desde Azure en el punto 3.

Casi toda la información necesaria para configurar el conector es extraída automáticamente del archivo JSON e insertada en los campos del conector. Solamente hay que agregar el URL del host del servicio API. Regrese al portal de Azure, abra el Grupo de Aplicaciones donde están los servicios del proyecto, abra el servicio del API Management, clic sobre "APIs", seleccione el API indicado y seleccione la pestaña "Settings". Copie el dominio de la casilla "Base URL", es decir, si en la casilla aparece un URL de "https://apimanroombooking.azure-api.net/roombooking", copie solamente la parte de "apimanroombooking.azure-api.net".

Ahora regrese a la configuración que se tenía abierta en PowerAutomate y copie este dominio en la casilla "Host" de la pestaña "Settings". Todas las otras configuraciones se pueden dejar por defecto. Utilice el botón de "Create connector" en la esquina superior derecha de la ventana. Haga clic sobre "Custom connectors" de la sección "Data" de nuevo, y el nuevo conector debe aparecer en la lista de conectores disponibles.

7. Seleccione el vínculo "My flows" y luego cree un nuevo flujo por medio del botón "+New" - "Automated-from blank". Asígnele un nombre al flujo y seleccione "When an item is created SharePoint" en la casilla de "Choose your flow's trigger". Use el botón de "Create". Seleccione la dirección del sitio en la primera casilla del trigger, y el nombre de la Lista.
8. Clic sobre el botón "+New step" y seleccione la pestaña de "Custom". La lista de custom connectors aparece. Seleccione el conector creado en el punto 6.


 When an item is created ...

*Site Address ▼


*List Name ▼


[Show advanced options](#) ▼

↓

 Choose an action ×

All Built-in Standard Premium Custom


CallAzureFunction


RoomBooking

▼

Triggers Actions

We couldn't find any actions

9. Después de seleccionar el conector aparece su ventana de configuración para crear una conexión con el conector:

When an item is created

* Site Address: Test_Guitaca - https://m365x951476.sharepoint.com/sites/Test_Guitaca

* List Name: RoomBooking

Show advanced options

RoomBooking

* Connection Name: RoomBookingConnection

* API Key: The API Key for this api

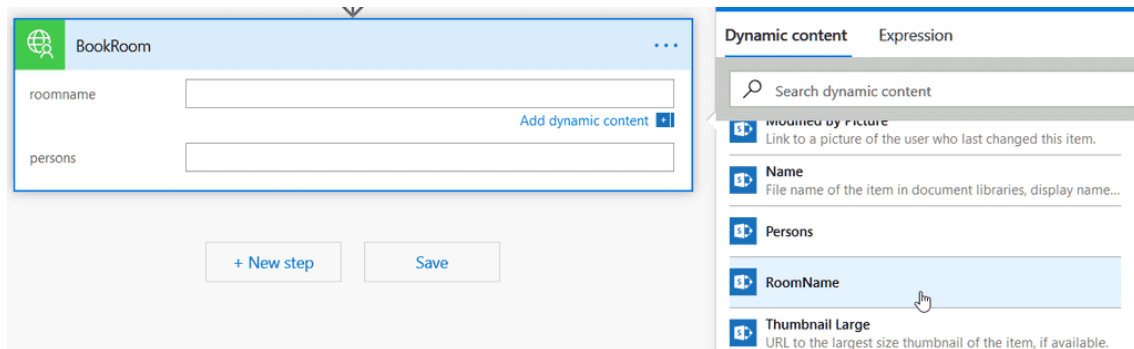
Create

Cuando se creó la función de Azure, se seleccionó que su autorización debería ocurrir por medio de una llave. Esta llave es indispensable para que las llamadas a la función sean aceptadas. Si se hubiera seleccionado "Anonymous", la llave no sería necesaria, y no aparecería en la ventana de la imagen 4, pero cualquier llamada al servicio sería aceptada sin validación. El API Management mantiene la llave, y expone otra llave que es necesaria para hacer llamadas a la función.

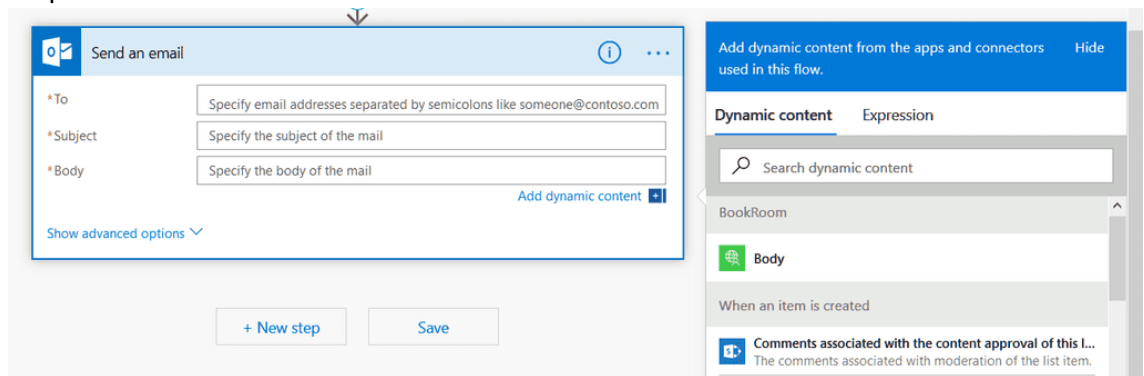
Para encontrar el valor de la llave, desde el portal de Azure abra el servicio del API Management y use el vínculo de "Subscriptions" en el menú vertical izquierdo. Utilice el botón de elipse ("...") al lado derecho de "Built-in-all-access" y seleccione "Show/hide keys". Use el botón con un icono de copiar de la "Primary key". Regrese al flujo y pegue el valor copiado en la casilla de "API key" de la imagen 4. Guarde los cambios en el flujo con el botón de "Create".

Note que este paso es necesario solamente la primera vez que se utiliza el conector. Si el conector ya ha creado una conexión anteriormente, este procedimiento inicial no es necesario. También es posible crear primero la conexión directamente desde la ventana de "Connections" en el portal de PowerAutomate, y usar el botón de "+New connection" para crear una conexión con el conector manualmente. En cualquiera de los dos casos, el procedimiento para encontrar la llave es igual al descrito.

Si los valores son aceptados (es decir, la llave es válida), la ventana cambia para mostrar los valores de las dos propiedades que la función va a recibir a través del API Management. Haciendo clic sobre "roomname" se abre la ventana de "Dynamic content" con las columnas de la Lista que contienen el valor a enviar. Seleccione "RoomName". Haga la misma operación para el campo de "persons":

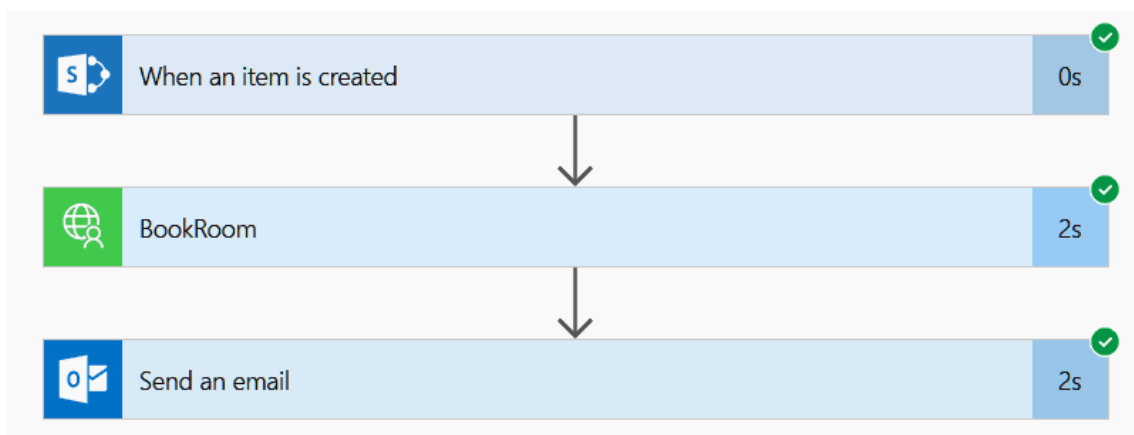


10. Agregue otro nuevo paso en el flujo. Utilice el botón de "+New step", use el botón de "Office 365 Outlook" y seleccione la acción de "Send an email". Configure las casillas de "To" y "Subject". Clic sobre la casilla de "Body" para abrir el "Dynamic content" y seleccione "Body" del conector. La respuesta de la función de Azure es enviada al API Management, y este la retransmite al flujo en el body de la respuesta de HTTP:



Use el "Flow Checker" (esquina superior derecha) para comprobar que el flujo no tiene errores, y guarde todo el flujo con el botón de "Save".

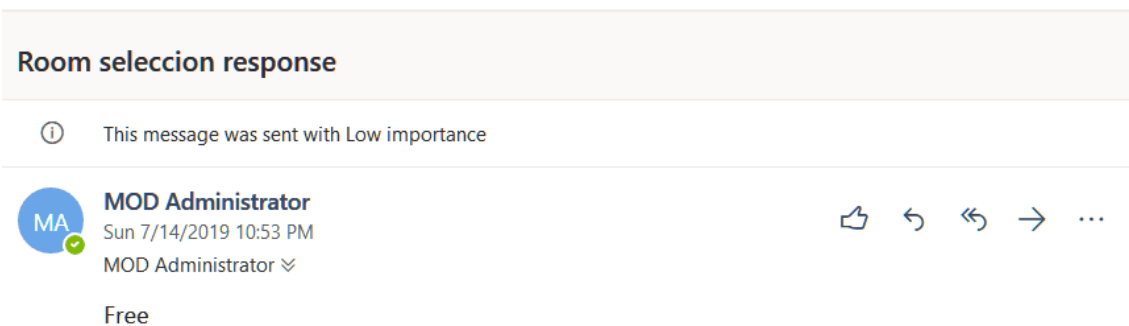
11. Para testear el funcionamiento de todo el sistema, cree un nuevo elemento en la Lista de SharePoint. Al crear el elemento se dispara el flujo, lo cual se puede seguir en la ventana de detalles del flujo mismo:



El flujo hace una llamada REST al Azure API Management, y este a su vez llama a la función. El funcionamiento de la función se puede seguir abriendo la función misma y su ventana de Logs:

```
Logs Errors and warnings Console
2019-07-14T20:52:16 Welcome, you are now connected to log-streaming service.
2019-07-14T20:53:16 No new trace in the past 1 min(s).
2019-07-14T20:53:55.605 [Info] Function started (Id=3cd5a7a3-5b0e-4ee2-a404-cbfb515b35a6)
2019-07-14T20:53:55.624 [Info] Request for a Conference Room is arrived
2019-07-14T20:53:55.624 [Info] roomname = Room number one
2019-07-14T20:53:55.624 [Info] persons = 37
2019-07-14T20:53:55.624 [Info] Returning - StatusCode: 200, ReasonPhrase: 'OK', Version: 1.1, Content:
System.Net.Http.ObjectContent`1[[System.String, mscorlib, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089]], Headers:
{
  Content-Type: application/json; charset=utf-8
```

Finalmente, el flujo envía un Email a la cuenta configurada indicando el valor generado por la función:



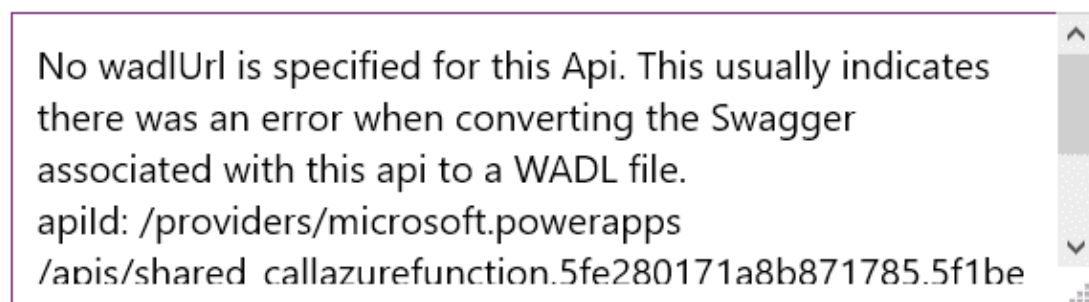
Modificar la definición de OpenAPI

Si se intenta utilizar el conector creado en el segundo artículo de la serie con la definición OpenAPI de la función, Power Apps lo rechaza con un error:

Error

There was a problem adding your service. Please try again later.

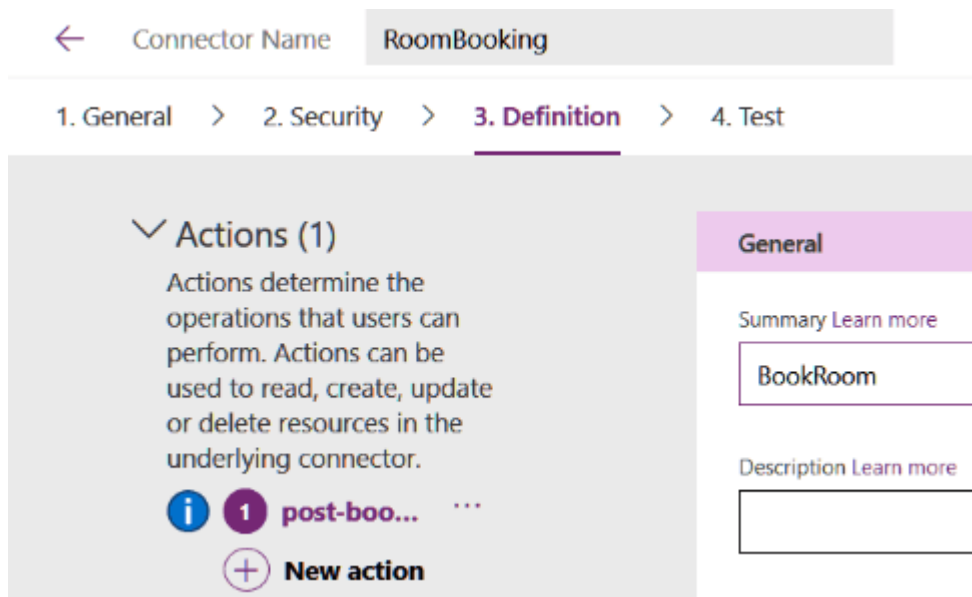
[Less](#)



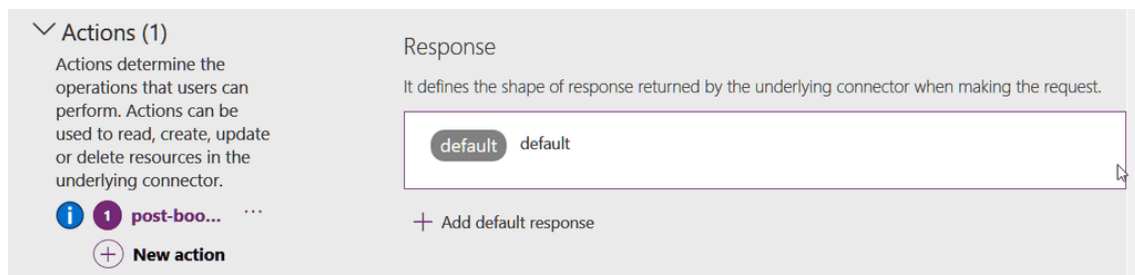
Close

El error es bastante críptico y no da ninguna indicación sobre el problema con la definición de OpenAPI. Para poder usar el conector en Power Apps, es indispensable solucionar primero el problema, como se indica a continuación.

1. Abra el sitio de Power Apps desde el portal de Office (<https://portal.office.com>), o desde el portal de Power Apps mismo (<https://powerapps.com>) y lóguese con credenciales que tengan permisos suficientes para crear y utilizar aplicaciones.
2. Expanda el menú de "Data" (menú vertical al lado izquierdo) y haga clic sobre "Custom connectors". Use el botón con el icono de un lápiz al lado del conector creado en el punto 6 del segundo artículo de la serie (conectores son iguales para Power Automate y Power Apps). Esto abre la configuración del conector en forma de edición. Haga clic sobre "3. Definition".

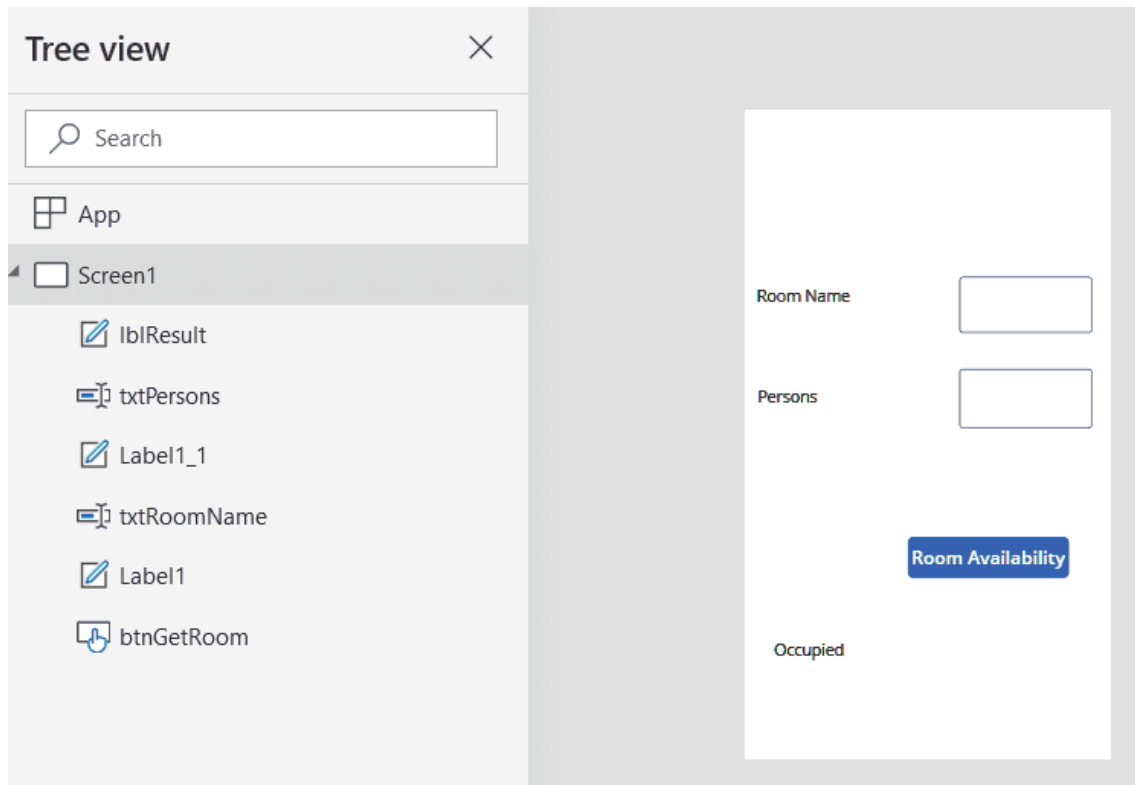


3. Primero haga clic sobre el botón de elipse al lado derecho del método "get-" y use "Delete". Este método no es utilizado y, además, contiene errores que impiden usarlo en Power Apps.
4. Luego seleccione el método "post-" y desplace la ventana hacia abajo, hasta la sección de "Response". Note que no hay una respuesta definida porque el OpenAPI exportado desde Azure API Management no detecta que haya una respuesta desde la función, y no crea una respuesta en la definición (esto hace que la definición no sea conforme al estándar de OpenAPI, y que Power Apps rechace el conector). Para solucionar el problema, use el botón de "+Add default response", acepte los valores por defecto y haga un update del conector. El resultado es una respuesta por defecto definida en el conector:



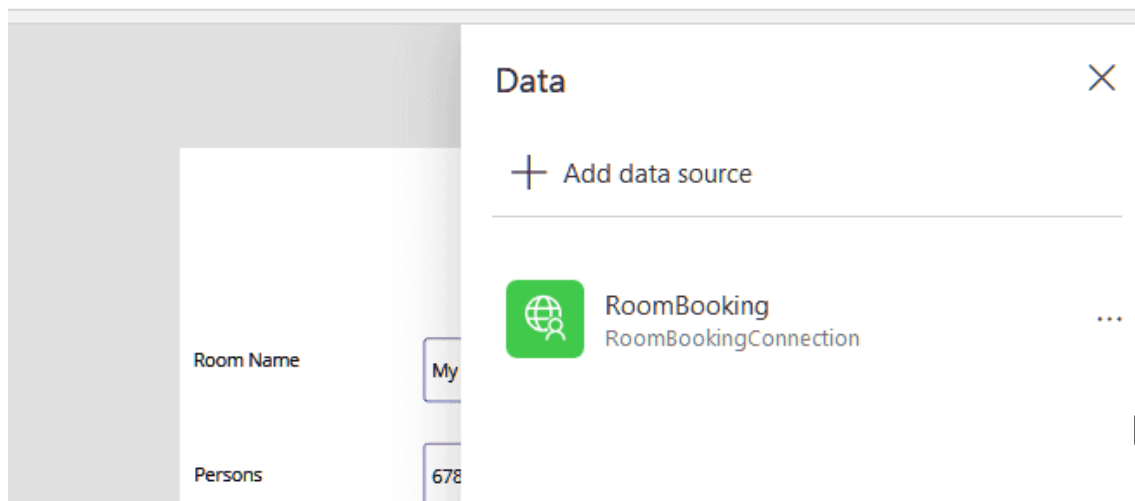
Crear la aplicación de Power Apps

- Desde el portal de Power Apps haga clic sobre "Apps" y sobre "+Create an app" - "Canvas" - "Start with a blank canvas" - "Phone layout" para crear una nueva Power App.
- Agregue dos labels ("Room Name" y "Persons"), dos text boxes ("txtRoomName" y "txtPesons"), un botón ("btnGetRoom") y un label ("lblResult") para mostrar el resultado.



- Agregue el conector. Seleccione la pestaña de "View" (menú horizontal superior) y "Data sources". En la nueva ventana use "+Add data source", busque el conector y haga clic sobre él. Si las modificaciones han sido correctas, el conector aparecerá en la lista de "Data" sin mostrar errores:

255, 1)

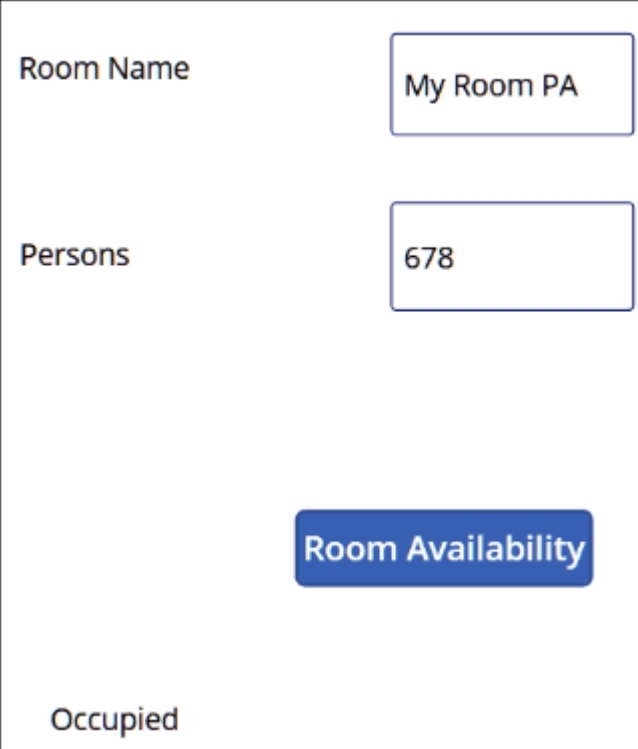


8. Cierre la ventana de "Data" y haga clic sobre el botón de la aplicación. En el método "OnSelect" utilice el siguiente fragmento de código:

```
UpdateContext({RoomAvailable:  
RoomBooking.postbookroom({roomname:txtRoomName.Text, persons:txtPersons.Text}}))
```

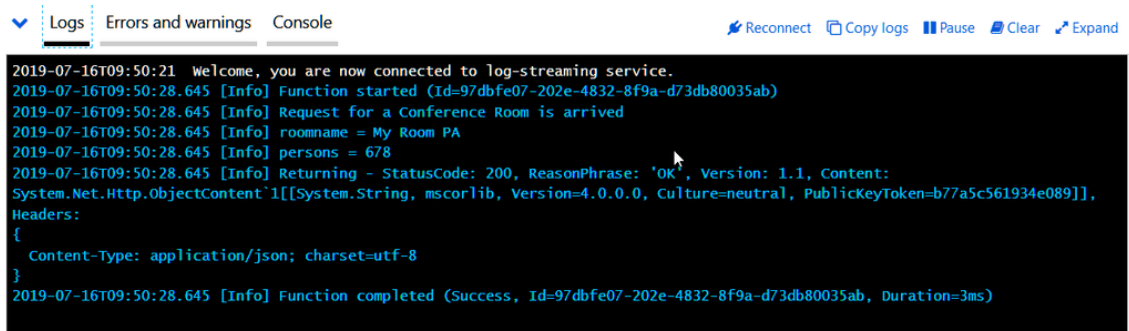
En este fragmento se crea una variable llamada "RoomAvailable" por medio del método "UpdateContext". A la variable se le asigna el resultado de llamar el método "RoomBooking.postbookroom", que utiliza dos parámetros de entrada: "txtRoomName.Text" para "roomname" y "txtPesons.Text" para "persons". Note que "RoomBooking" es el nombre del conector y "postbookroom" el nombre de la operación (sin "-") que indica la definición del conector. De igual forma, "roomname" y "persons" son los dos parámetros de entrada que el OpenAPI ha definido en el QueryString de entrada del conector.

9. Haga clic sobre el label para mostrar los resultados en la aplicación, y en su propiedad de "Text" seleccione "RoomAvailable". Esta es la variable que el botón crea y asigna un valor cuando la función de Azure es llamada
10. Use el "App Checker" (esquina superior derecha) para comprobar que la aplicación no tiene errores
11. Para testear el funcionamiento de la aplicación, use el botón de "Preview the app" en la esquina superior derecha, o use F5. Inserte dos valores en las dos casillas de texto y use el botón. El resultado deberá aparecer en el label de resultados:



The screenshot shows a web application interface with a light blue background. It contains two text input fields: the first is labeled "Room Name" and contains the text "My Room PA"; the second is labeled "Persons" and contains the text "678". Below these fields is a blue button with the text "Room Availability". At the bottom left of the form, there is a label "Occupied".

La aplicación hace una llamada REST al Azure API Management, y este a su vez llama a la función. El funcionamiento de la función se puede seguir abriendo la función misma y su ventana de Logs:



```
2019-07-16T09:50:21 welcome, you are now connected to log-streaming service.
2019-07-16T09:50:28.645 [Info] Function started (Id=97dbfe07-202e-4832-8f9a-d73db80035ab)
2019-07-16T09:50:28.645 [Info] Request for a Conference Room is arrived
2019-07-16T09:50:28.645 [Info] roomname = My Room PA
2019-07-16T09:50:28.645 [Info] persons = 678
2019-07-16T09:50:28.645 [Info] Returning - StatusCode: 200, ReasonPhrase: 'OK', Version: 1.1, Content:
System.Net.Http.ObjectContent`1[System.String, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089]],
Headers:
{
  Content-Type: application/json; charset=utf-8
}
2019-07-16T09:50:28.645 [Info] Function completed (Success, Id=97dbfe07-202e-4832-8f9a-d73db80035ab, Duration=3ms)
```

12. Note que también es posible crear una aplicación de Power Apps para la Lista de SharePoint utilizada en el segundo artículo. Pero en ese caso, la aplicación no es más que un reemplazo de la interfaz por defecto de la Lista en SharePoint, que lo que hace es crear un elemento en la Lista, el que dispara el flujo acoplado a ella.