Microsoft IoT Hackathon DX

Scenario 1 – Health

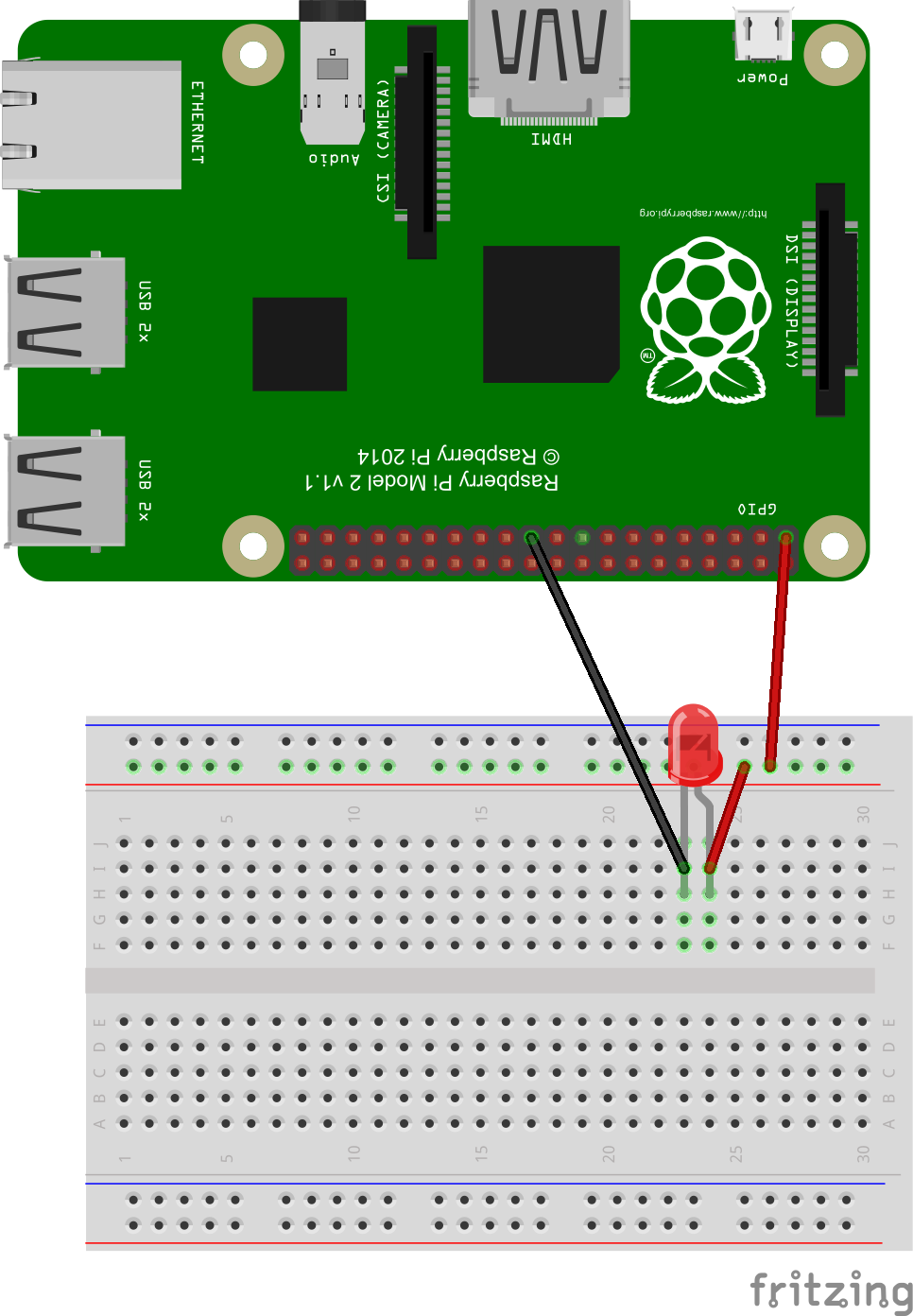
# Overview

This scenario simulates the tracking of vulnerable patients in a care home. You will be using an IBeacon and a Raspberry Pi equipped with a Bluetooth dongle to detect when the Raspberry Pi enters the range of an IBeacon. The Raspberry Pi could be a wearable device in the form of an electronic id bracelet worn by the patient and the IBeacons could be placed near exits of the building. Using this set up you could use this Scenario to detect when a patient is trying to leave the building.

When the Raspberry Pi detects the IBeacon a message is sent to an IoT Hub. Using Stream Analytics this data is then saved to an Azure SQL Database. In addition to this an Azure Web Job listens to the IoT Hub and when the IBeacon is detected as being within a certain range the Web Job sends a cloud to device message back to the Raspberry Pi. This message tells the Raspberry Pi to flash an LED connected to the device. The WebJob also uses Twilio to send an SMS alert about the patient’s proximity to an exit to a mobile phone carried by a nurse.

The Raspberry Pi in this scenario runs Raspbian and python.

# Circuit Diagram

Wire up your Raspberry Pi by following the circuit diagram. 

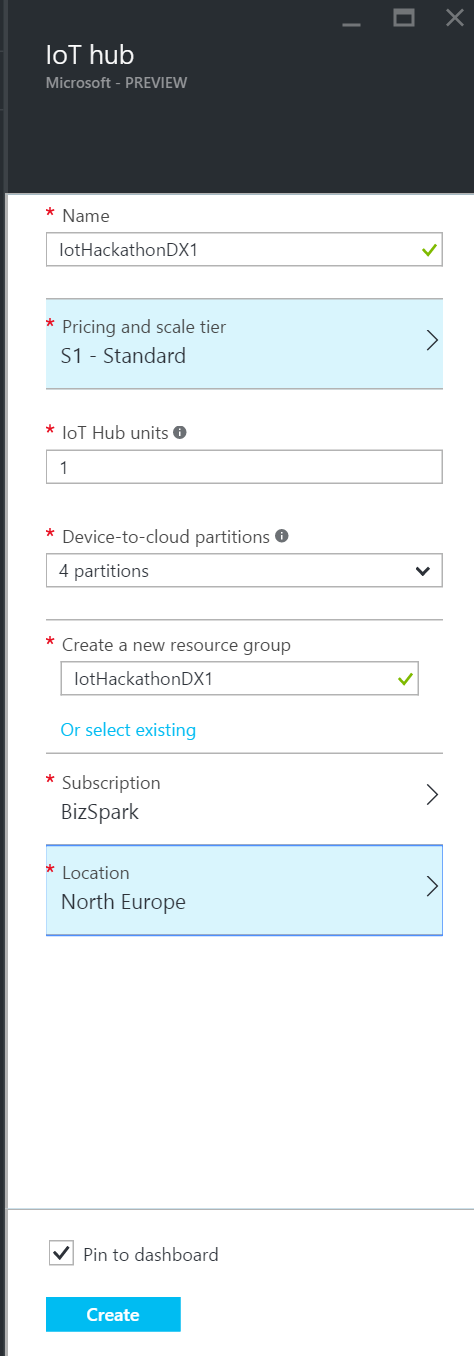
# Login to Azure

Azure has two management portals – the classic portal (<http://manage.windowsazure.com>) and a new portal that is in preview at <http://portal.azure.com>. You will be using the new portal for this challenge.

1. **Open a browser** and go to [**http://portal.azure.com**](http://portal.azure.com)
2. Enter your **Microsoft Account email address** and **password** for the Microsoft Account associated with your subscription.
3. You will now be in your Azure subscription (see opposite) and from here you can create and manage Azure services.

# Provision the IoT Hub

1. On the Preview Portal (remember you need to be in the Preview Portal), click **“+ NEW”** (top left) and select **Internet of Things.**



1. Select **IoT Hub** from the featured apps shown.
2. Enter a name for your hub. In this example I am using “IoTHackathonDX1”.
3. The IoT Hub is charged based on the number of messages per day you want to send to the hub. There is a **“free”** tier in this service that gives you 8,000 messages/day at no charge – select this option unless you are already using the **“free”** tier, in which case select “**S1 – Standard**”.

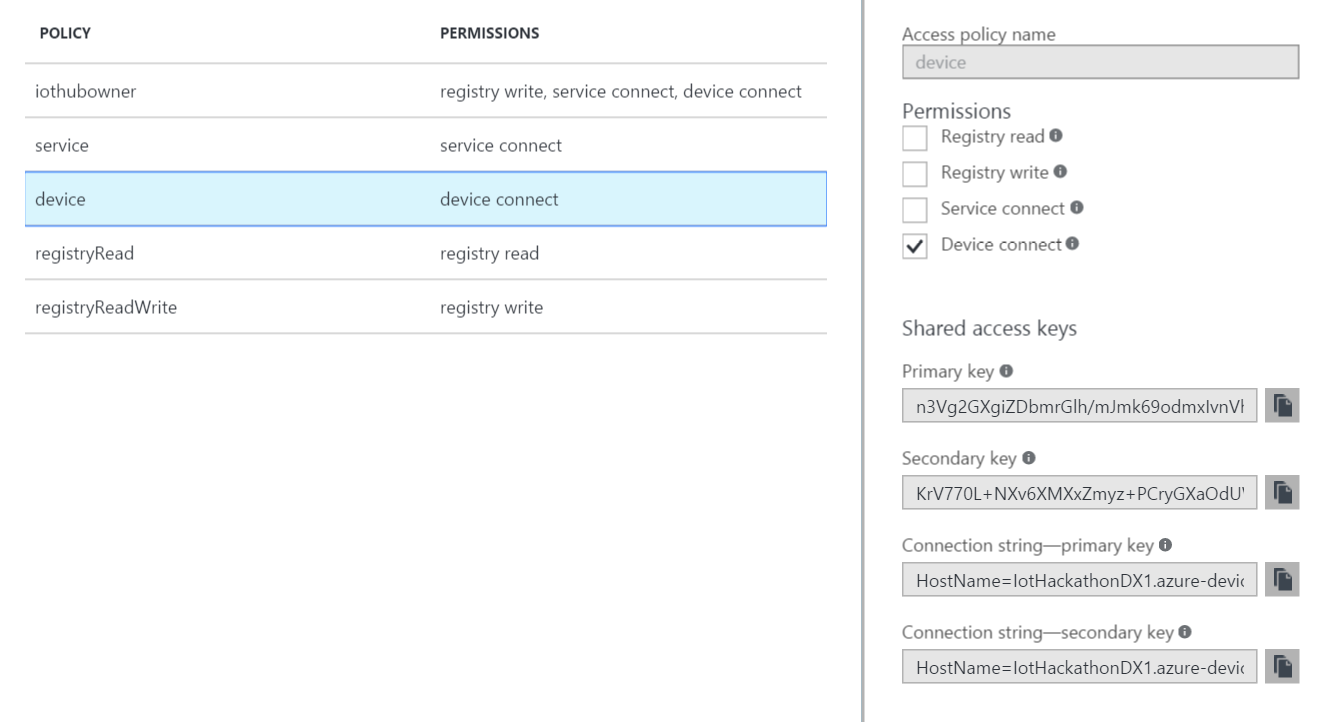
**Each Azure subscription may only have one free tier IoT Hub provisioned**

1. Leave the IoT Hub units as 1 and the number of partitions as 4.
2. Create a new resource group by clicking **“or create new”** and enter the name you want to call this resource group. We will re-use this resource group for provisioning additional services later on in this lab.
3. For the location select a location local to you where possible.
4. Click the **create button**. You will now be taken back to your Azure dashboard where you will see a new dashboard tile appear showing the status of the provision. IoT Hubs can take a few minutes to provision.

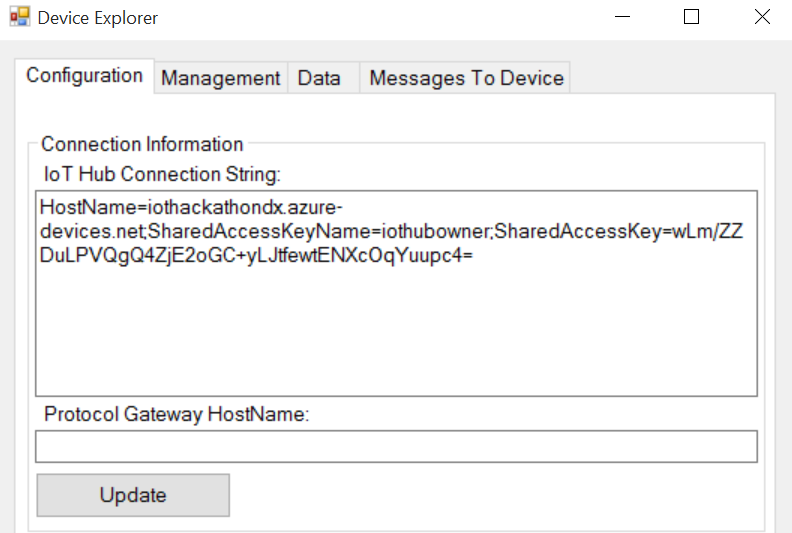
# Using Device Explorer

You can’t configure your IoT Hub devices from the Preview Portal however as part of the Azure IoT SDK there is an application called DeviceExplorer which you can use to manage devices, view messages being sent to your hub and send cloud to device messages. For this part of the lab you will need to have the Azure IoT SDK.

1. **Download** or clone the GitHub repo for the **Azure IoT SDK** (<https://github.com/Azure/azure-iot-sdks>). If downloading extract, the zip files to your machine.
2. Navigate to \tools\DeviceExplorer\ in your local repo and **open the DeviceExplorer solution** file in Visual Studio.
3. Run the project.
4. To use Device Explorer, the first thing you need to do is enter the connection string for your IoT Hub. In the Preview Portal click your IoT Hub tile that was created for your earlier.
5. Two panes should now appear for your IoT Hub. The right-hand side pane should be titled **“settings”**
6. Select the **“Shared access policies”** menu option and select the **“iothubowner”** policy in the shared access policies tile.



1. From the **“iothubowner”** tile copy the **“Connection string – primary key”.**
2. Back in Device Explorer paste into the **“IoT Hub connection string”** textbox in Device Explorer.
3. Click the **“Update”** button and a dialog confirming the update of the settings should appear.

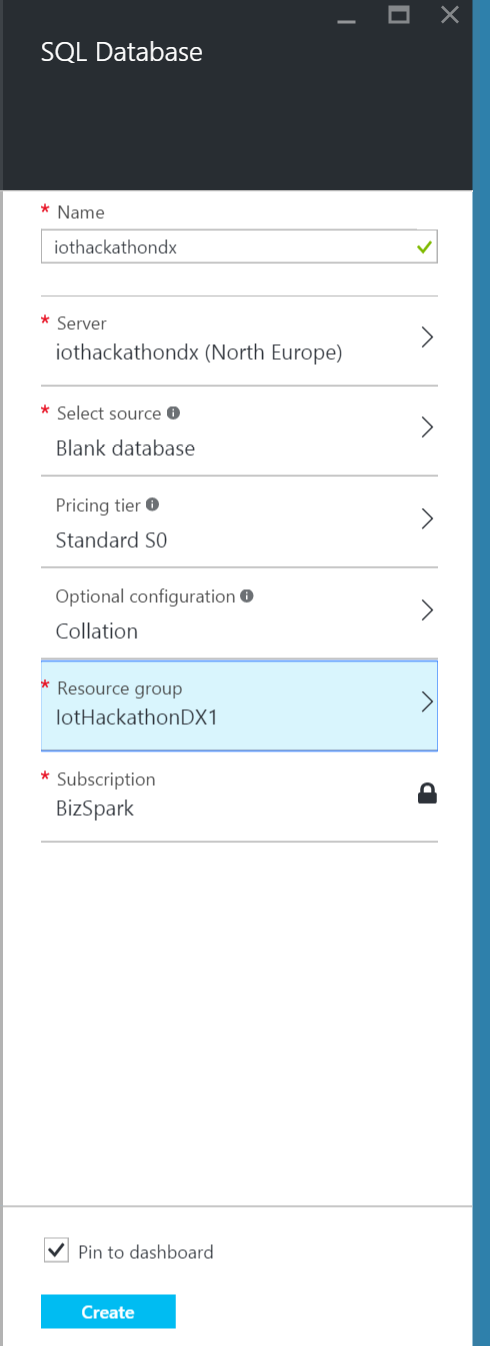


1. Select the **management tab** and click the **“Create”** button.
2. Enter a name for your device in the **“device id”** textbox. In this example I’ve called my device “raspberrypidevice” and click **“create”**. You should now see the device you created shown in the data grid. You can verify it was created by clicking the “refresh” button.

# Configuring your Raspberry Pi

# Provision an SQL Database

1. On the Preview Portal, click **“+ NEW”** (top left) and select **DATA AND STORAGE.**
2. Select **SQL Database** from the list of featured apps.



1. Enter the name of your database in the settings tile. In this example I am using IoTHackathonDX.
2. Next click **“configure server”** followed by **“create new server”**.
3. Enter the server name. In this example I am using IoTHackathonDX. I would suggest calling the server the same as the database to make it easy to remember. Enter a username and password making a note for these for later.
4. Select a suitable region for your server.
5. Click **“Ok”** to confirm the server settings.
6. Select a suitable pricing tier. By default, the pricing tier will be S0 – Standard but you can use the **“B – Basic”.**
7. Select a resource group for the database. I would recommend the same resource group you created for the IoT Hub earlier.
8. Click **“create”**. The SQL Database will now be provisioned. As with the IoT Hub you can see the status of the provisioning process on your dashboard.

## Configure the SQL Database

Next you need to create the table that the data from your IoT Hub will be saved to using Stream Analytics. You will use Visual Studio for this part of the exercise.

1. Open **Visual Studio 2015**.
2. If you don’t already have **Cloud Explorer** installed download and install it from <https://visualstudiogallery.msdn.microsoft.com/84e83a7c-9606-4f9f-83dd-0f6182f13add>
3. Add your Azure account to cloud explorer by clicking the **“Connect to Microsoft Azure”** link.
4. Enter your credentials in the Visual Studio login popup box.
5. You should now see a list of Azure components. Expand the SQL Databases node and right click on the database you created in the last step. Click the **“Open in SQL Object Explorer”** option and you will be prompted for the password for your database.
6. In the SQL Object Explorer navigate to the table node for your database and right click, selecting **“Add New Table…”**.
7. Enter the following SQL statement into the bottom T-SQL window: -

CREATE TABLE [dbo].[SensorData] (

[ReadingId] INT IDENTITY (1, 1) NOT NULL,

[ReadingDateTime] DATETIME CONSTRAINT [DF\_SensorData\_ReadingDateTime] DEFAULT (getutcdate()) NOT NULL,

[DeviceUDID] UNIQUEIDENTIFIER NOT NULL,

[Major] INT NOT NULL,

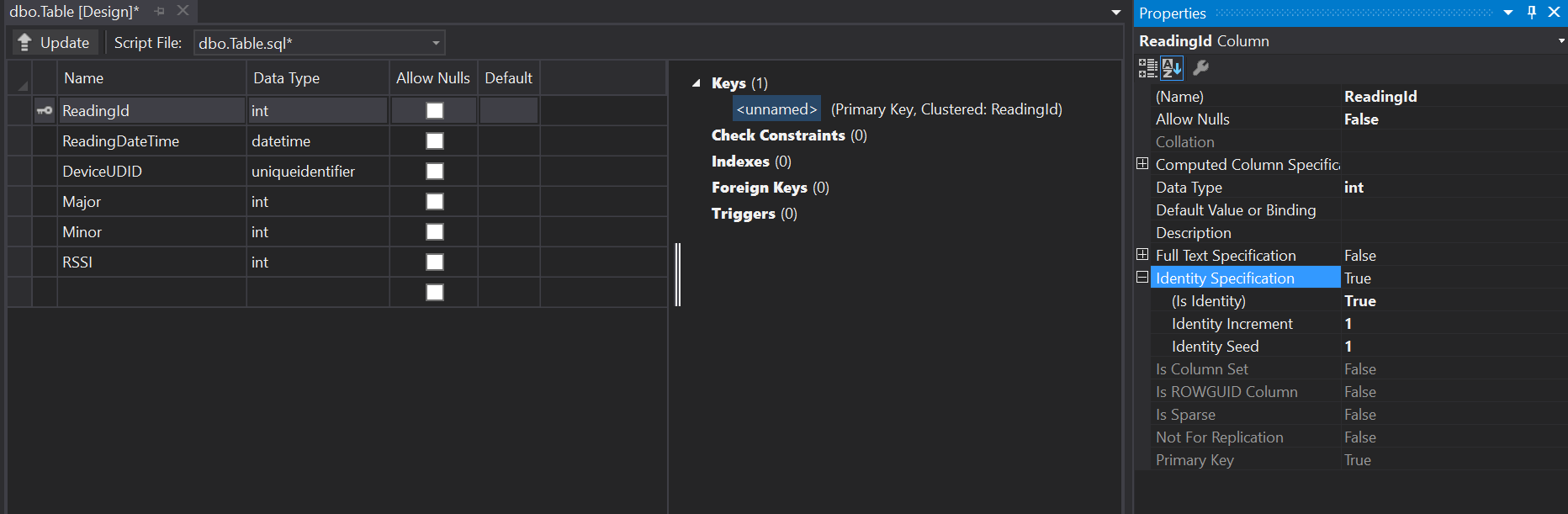
[Minor] INT CONSTRAINT [DF\_SensorData\_Minor] DEFAULT ((1)) NOT NULL,

[RSSI] INT NOT NULL,

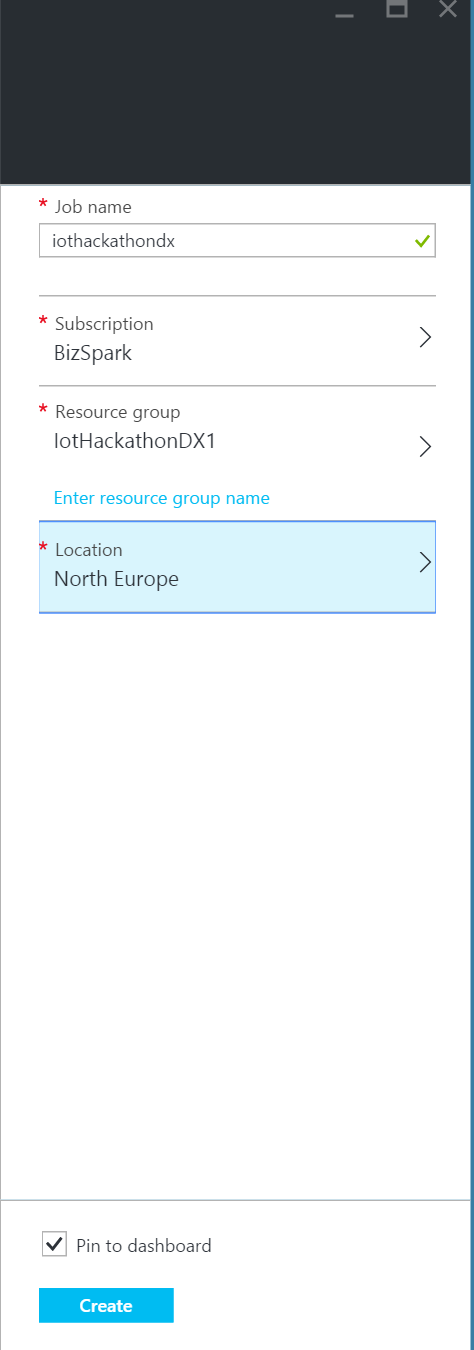
PRIMARY KEY CLUSTERED ([ReadingId] ASC)

);

1. Click the **“update”** button in the top left hand of the designer window.

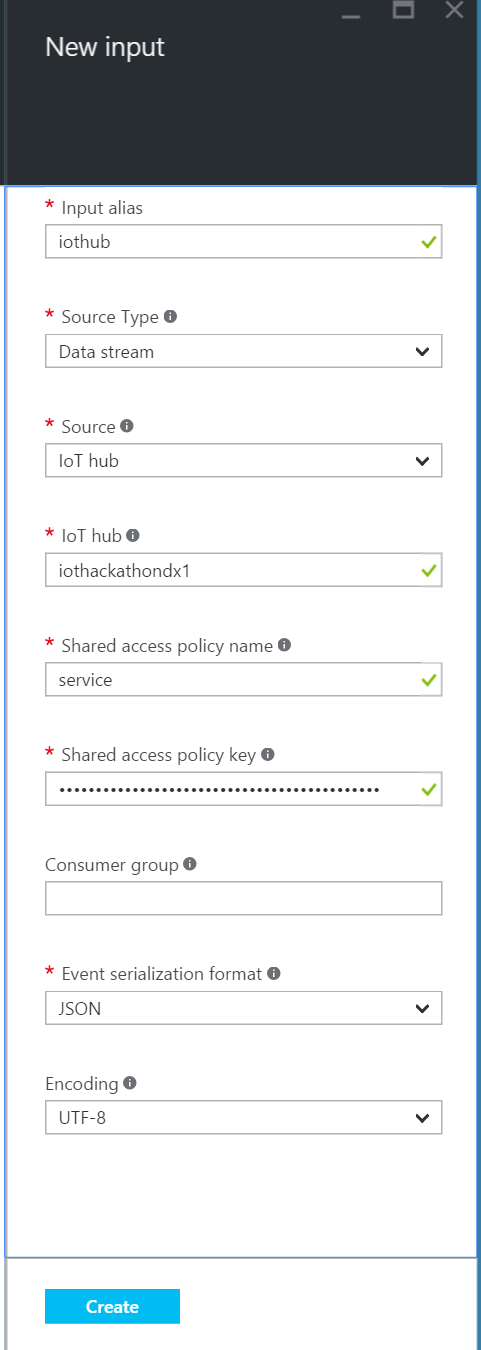


# Provision a Stream Analytics Job



You are now going to provision and configure the Azure Stream Analytics service. This will be used to take our data from your IoT Hub and save it into the SQL Database.

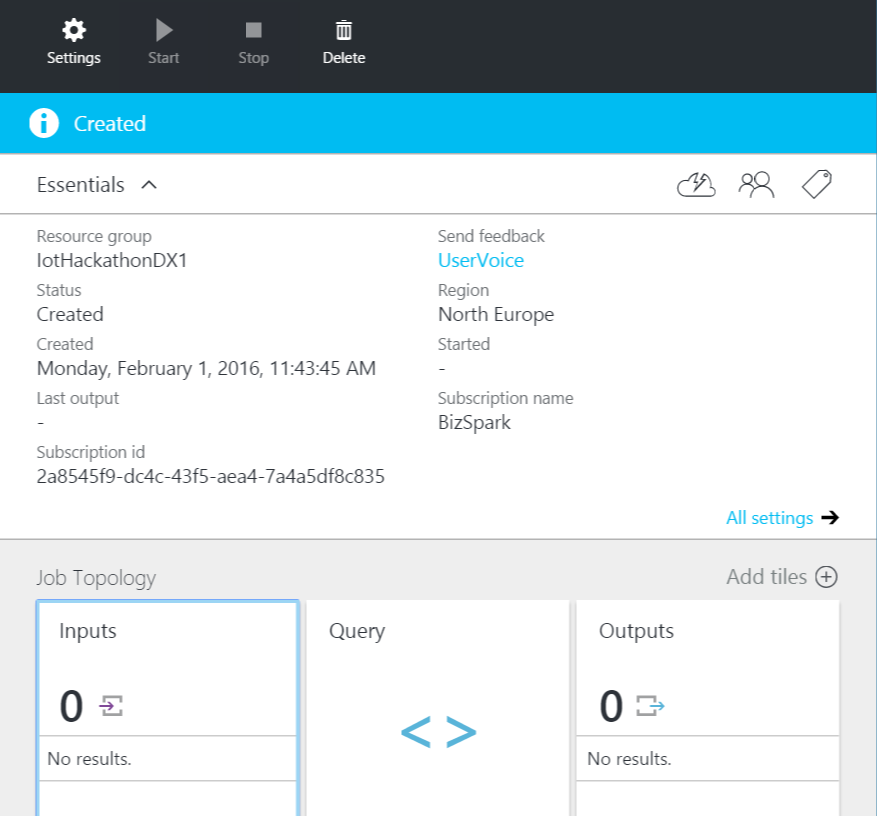
1. On the Preview Portal, click **“+ NEW”** (top left) and select **Internet of Things.**
2. Select **Stream Analytics Job** from the list of featured apps.
3. Enter a suitable name for the job.
4. Make sure a subscription is selected.
5. Select a resource group for the Job. I would recommend the same resource group you created for the IoT Hub earlier.
6. Select a suitable region for the Job.
7. Click the **“Create”** button. The Stream Analytics Job will now be provisioned. As with the IoT Hub you can see the status of the provisioning process on your dashboard.



## Configuring the Stream Analytics Job

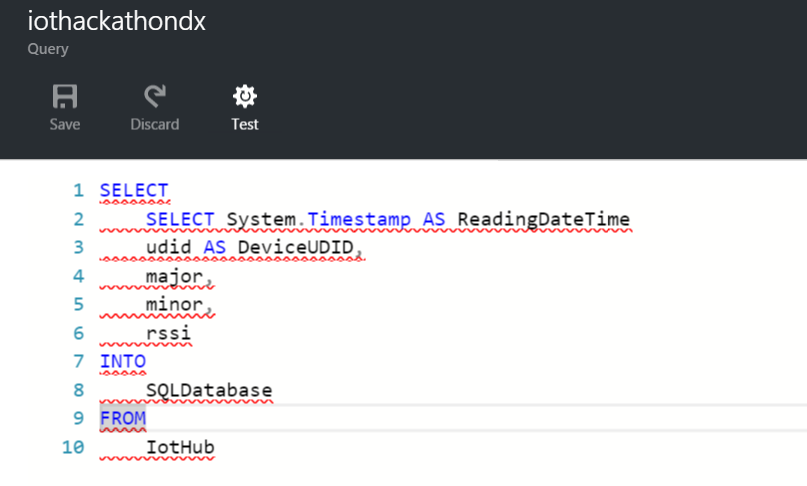
Once the job has provisioned the Preview Portal should automatically open up the job.

1. Before creating our first input you need to look up a shared access policy key for the IoT Hub you created earlier.
   1. From your **dashboard** select the IoT Hub you created earlier.
   2. From the settings tile select **“Shared access policies”.**
   3. Select the service policy.
   4. Copy the **“primary key”** for the policy.
   5. Close the IoT Hub tiles and select the Stream Analytics Job from your dashboard.
2. Under **“Job Topology”** click the “Inputs” box. You will then see a new tile open.
3. On the “Inputs” tile click the top left **“Add”** link.
4. In the **“Input Alias”** textbox enter“IoT Hub”.
5. Select **“Data stream”** as the Source Type.
6. Select **“IoT Hub”** as the Source.
7. Next enter the name of the IoT Hub you created earlier.
8. In the **“shared access policy name”** textbox enter **“service”**.
9. In the **“shared access policy key”** textbox paste the primary key you copied from the IoT Hub a few moments ago.
10. Leave the consumer group blank.
11. Make sure the Event serialization format is set to JSON and the Encoding is set to UFT-8.
12. Click the **“Create”** button. You will now see under the list on inputs the new input you just created.
13. Back under **“Job Topology”** click the “Outputs” box. You will then see a new tile open.
14. On the “Outputs” tile click the top left **“Add”** link.
15. In the **output alias** textbox enter **“SQLDatabase”**.



1. Select **SQL Database** from the sink drop down.
2. Enter the database name you created earlier in this exercise along with the server name, username and password. The server name should be in the format of **“<servername>.database.windows.net”**
3. In the **“table”** textbox enter **“sensordata”** which is the table you created a moment ago.
4. Click the **“Create”** button. You will now see under the list on outputs the new output you just created.
5. It’s now time to create your first query. Back under **“Job Topology”** click the **“Query”** box.
6. In the query editor enter the following query: -

SELECT



SELECT System.Timestamp AS ReadingDateTime

udid AS DeviceUDID,

major,

minor,

rssi

INTO

SQLDatabase

FROM

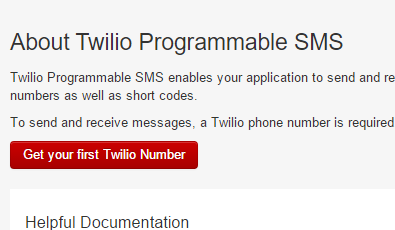
IotHub

1. Click the **“save”** button. Your Stream Analytics Job is now fully configured.

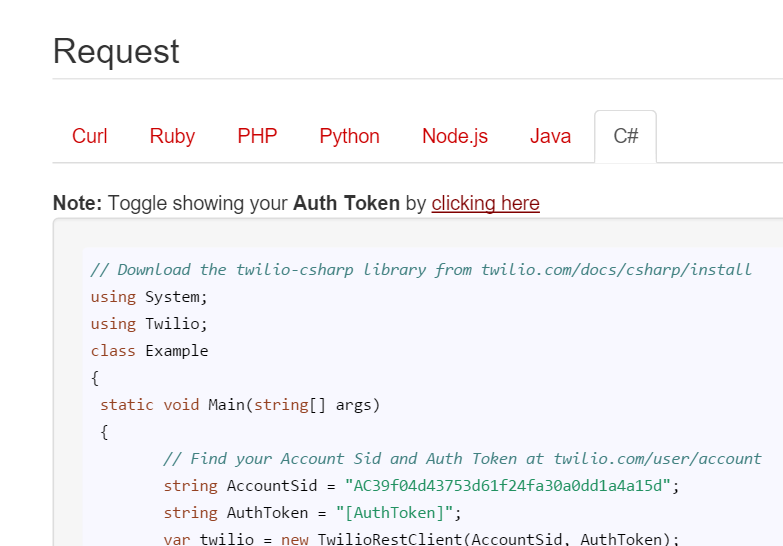
# Configuring Twilio

You need to setup a free Twilio account.

1. Go to <https://www.twilio.com/try-twilio> and complete the form.
2. Complete the verification step using your mobile number.
3. Click the **“Get your first Twilio number”** button and accept the number given by clicking the **“Choose this number”** button. Make a note of the number you have been assigned.



1. Next in the top right hand corner **click your user name** and select **“Account”** from the menu.
2. In the live Api credentials box click the padlock icon to reveal your **Auth Token**. You will need the **Auth Token** and the **AccountSID** in the next step.



# Configuring the WebJob

You will need to next configure an Azure WebJob. The WebJob is going to monitor the IoT Hub data stream and when the IBeacon is within a certain range will send a Cloud to Device message back to your Pi. The WebJob will also send an SMS to your mobile number using Twilio.

1. Open the Scenario1-WebJob solution in Visual Studio.
2. In solution explorer right click on the top solution node and select **“Restore Nuget Packages”**. This will restore the packages the project uses for Azure and Twilio.
3. In solution explorer double click the **“Properties”** node to open the project properties.
4. For the **“NurseNumber”** property enter your number.
5. For the **“TwilioNumber”** property enter your Twilio number.
6. For the **“TwilioAccountSid”** and **“TwilioAuthToken”** properties enter the values on your Twilio account page.
7. You now need to create a consumer group for your IoT Hub. Follow these instructions: -
   1. From your **dashboard** select the IoT Hub you created earlier.
   2. From the settings tile select **“Messaging”.**
   3. Under the consumer groups section in the empty text box enter “range” followed by the enter key. You should now see a green tick appear in the text box.
   4. Click the **“save”** button on the Messaging tile.
8. Back in the Web Job solution the **“ConsumerGroupName”** property should be set to “range”.
9. Set the **“IoTDeviceId”** property to the name you gave your device in the IoT Hub earlier.
10. Finally, in the **“IoTHubConnectionString”** property enter the **IoT Hub’s service connection string**. Details of how to get this are listed in the Configuring Stream Analytics.
11. Save and close the properties window.
12. Right click on the project node in the Solution Explorer again and this time chose **“Publish as Azure WebJob…”**.
13. From the Publish window make sure the **WebJob run mode** is set to **“Run Continuously”**.
14. Click “Ok”. You might now see a dialog saying installing WebJob Nuget Package.
15. Under “Select a publish target” select **“Microsoft Azure Web Apps”**.
16. Next click the **“New”** button.
17. Enter a name for the “Web App Name”.
18. Under “App Service plan” select **“Create new App Service plan”** and enter a name for the plan.
19. Select the resource group you used for the rest of the Azure services you have provisioned.
20. Select the same region as the rest of the Azure services you created earlier.
21. Select no database from the **“Database server”** drop down.
22. Click the **“Create”** button.
23. The Web App will now be provisioned and may take a few moments.
24. Once the Web App has been provisioned click the **“Publish”** button. This will publish your Azure WebJob to the website and start running the WebJob automatically.

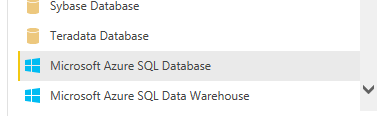
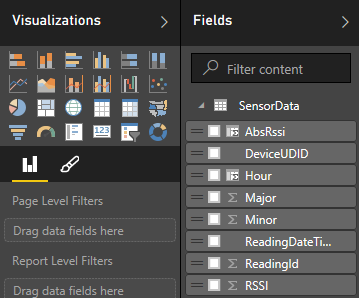
# Running the example

You have now fully configured the Raspberry Pi, IoT Hub, Stream Analytics, Web Job and SQL Database and we are ready to start receiving data.

1. Open **“PuTTY”** (or your own choice of SSH tool) on your computer. If you do not have PuTTy already installed it can be downloaded from <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
2. Enter the address of your RaspberryPi in the host name and make sure the port is set to 22.
3. Click the **“open”** button.
4. PuTTY will now connect to your Pi. Once it’s connected you will be prompted for the username and password. The username is Pi and the password is raspberry.
5. Now you are logged into an SSH session you are almost ready to run the code. First we need to navigate to the python code. Type **“cd /IoTHackathon/Scenario1/”**.
6. To run the code type **“sudo python scenario1.py”**.
7. You should now see the program running in your SSH session. When an IBeacon is detected you will see the IBeacon Id and RSSI data output to the window. When the IBeacon is moved around you will see the RSSI figure change.
8. To exit the program press **“Ctrl + Z”**

# Using PowerBi

In this section you are going to use PowerBi to visualise the data.

1. If you don’t already have PowerBi desktop edition installed on your machine download and install it from <https://powerbi.microsoft.com/en-us/desktop/>.
2. Open PowerBi and click on the **“Get Data”** link on the start-up screen.
3. From the list select **“Microsoft Azure SQL Database”** and click the **“connect”** button.  
   
4. Enter the database name you created earlier in this exercise along with the server name. The server name should be in the format of **“<servername>.database.windows.net”** and click **“Next”**.
5. Select **“Database”** from the side menu and enter the username and password for your database.
6. In the data navigator from the left hand side use the tree to navigate to the **“Sensor Data”** table and tick. Finally click the **“Load”** button.
7. From the tool bar click the “**New Column**” button and in the formula field enter **“AbsRssi = ABS(SensorData[RSSI])”** followed by the Enter key.
8. Click the “New Column” button again and this time enter **“Hour = Hour(SensorData[ReadingDateTime])”**, again followed by the Enter key.
9. On the right hand side of PowerBi you should see a panel as per the below screen shot: -  
   
10. Next click the **“Column chart”** icon (2nd Icon top row) under the Visualization section.
11. From the Fields section drag the **“Hour”** column to the axis place holder.
12. From the fields section drag the **“AbsRssi”** to the values placeholder.
13. In the value placeholder you should see a small drop down arrow, click this and select “Count” from the sub menu.
14. You should now end up with a chart looking a little like this: -  
    