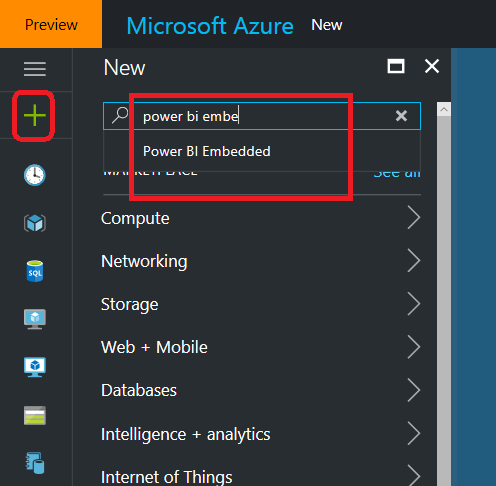
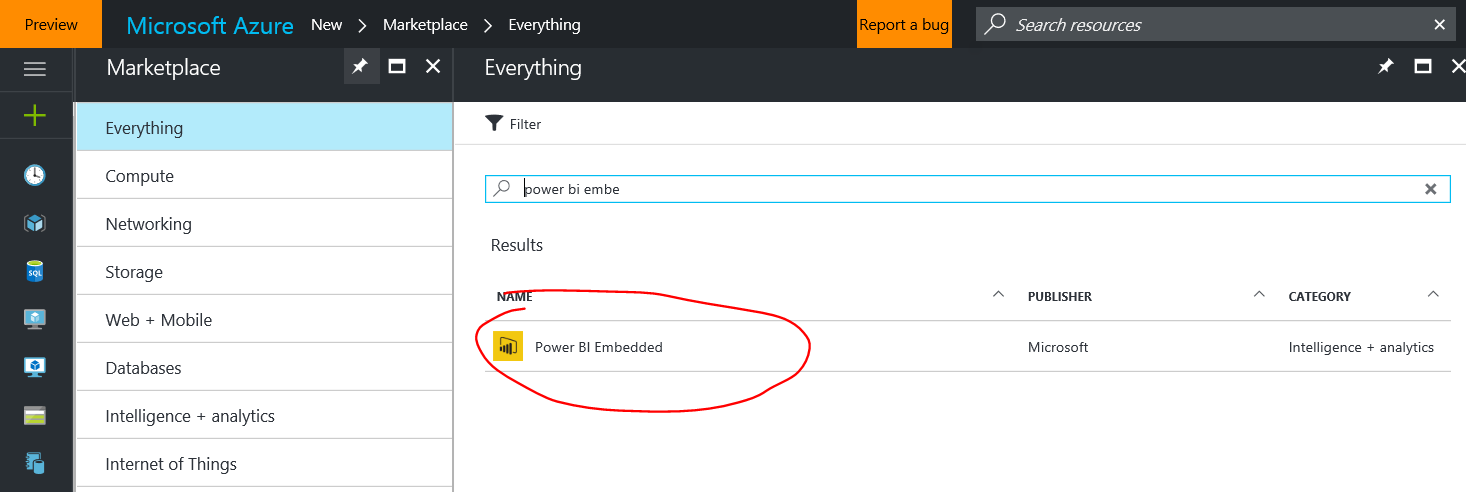
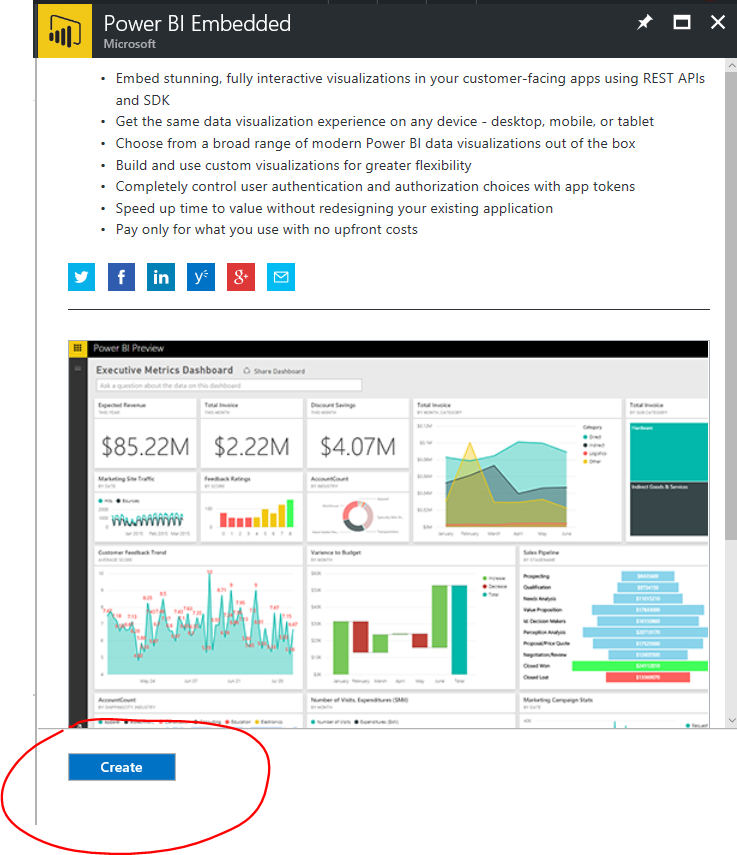
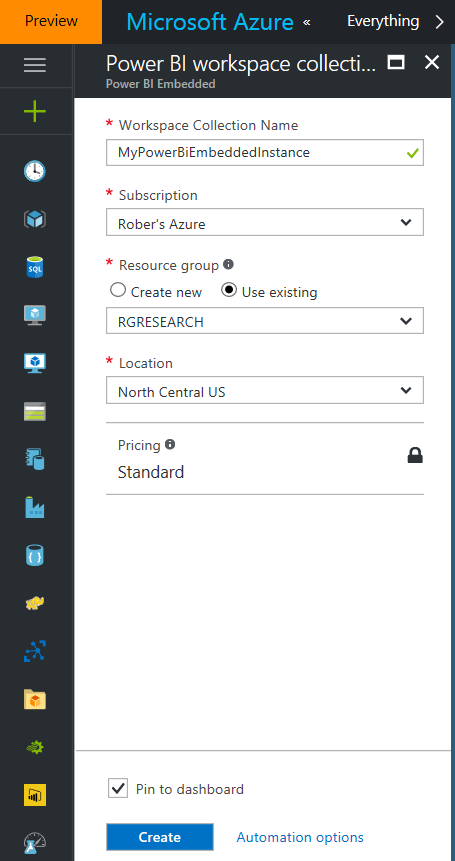
1. Create your Power BI embedded instance:



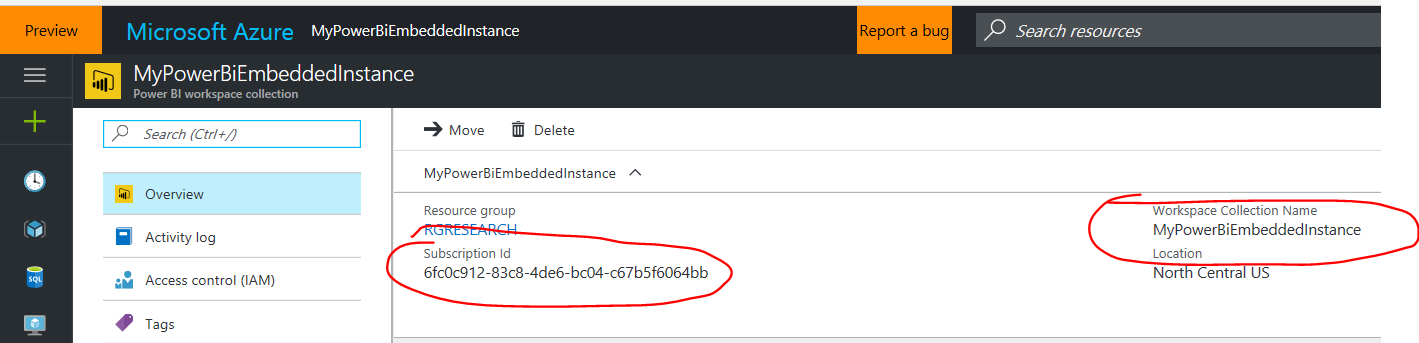






2 - Access your instance configuration at Azure portal:

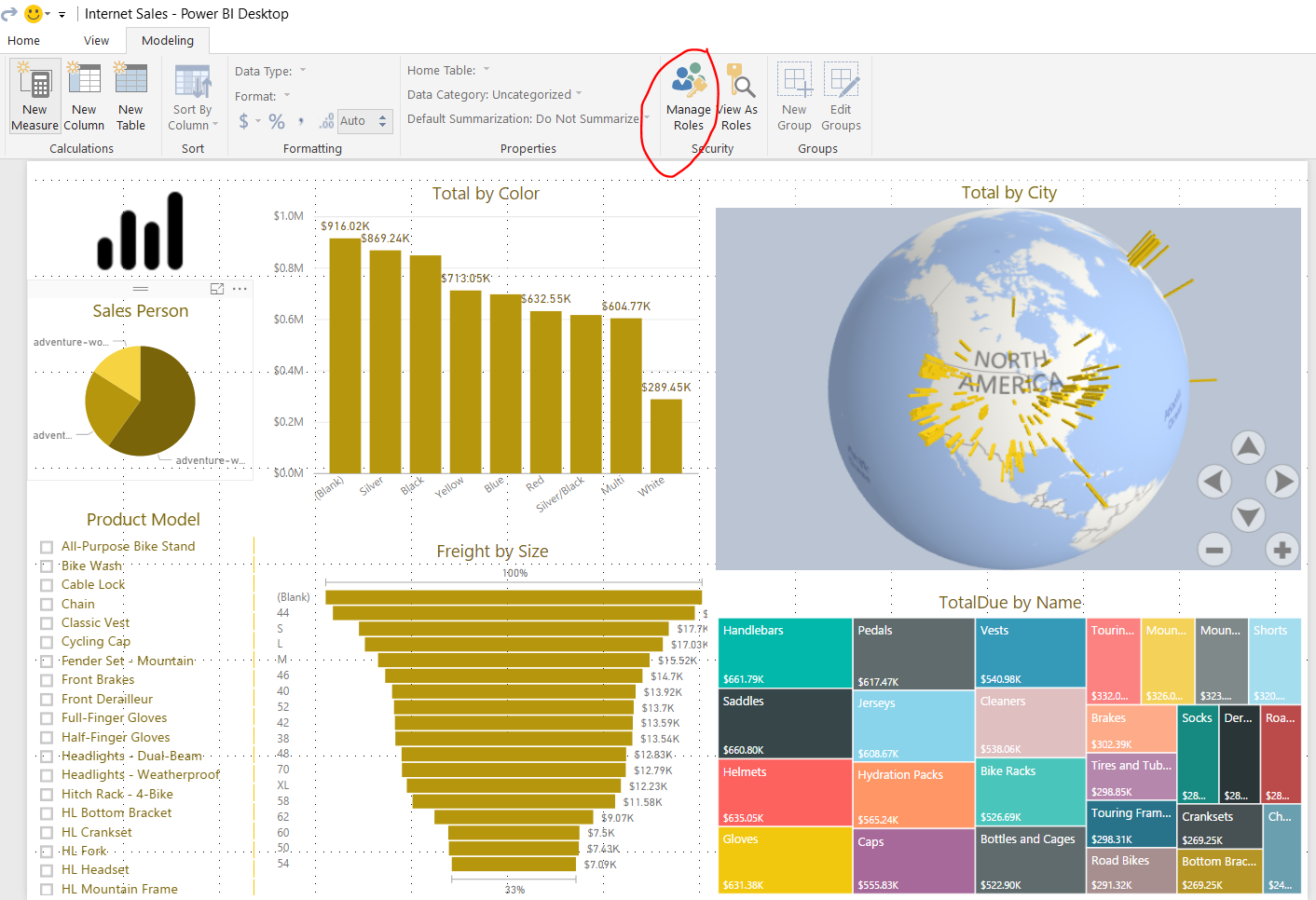
* Take notes of your subscription ID and Workspace name:



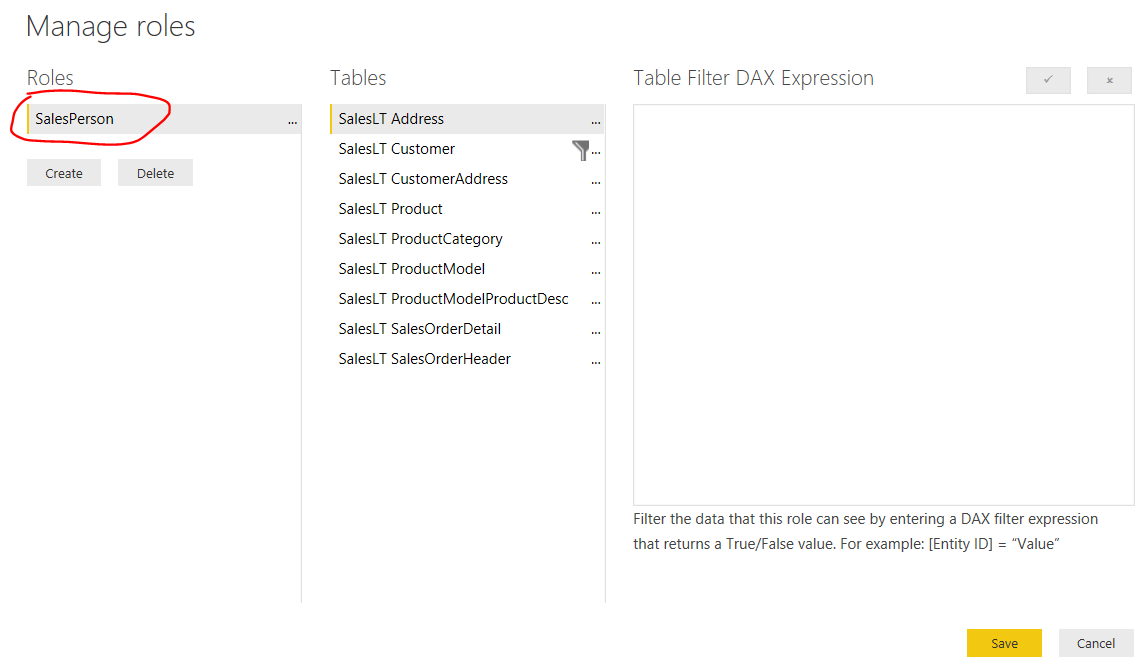
3 – Configure your Power BI filter parameters properly:

At the power bi sample, you can check the parameters configuration to know how to properly make the setup according your scenario. So, be aware of:

* Check the roles configuration:

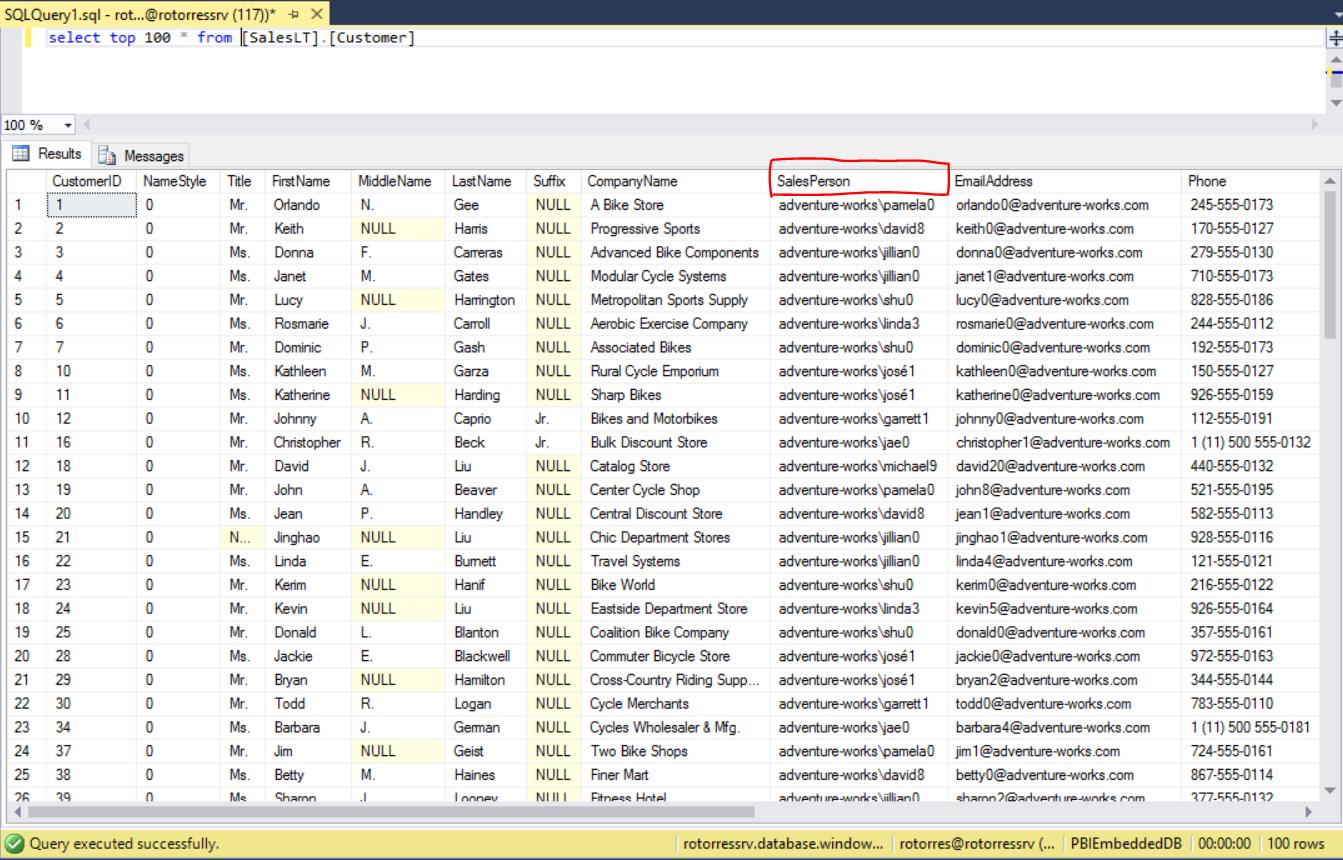


* The 2 main information of the dialog box:
  + The name of the Role: In my case, I’ve named the role as “SalesPerson”… but it could be anything… like “MyInsaneEasySampleRole” for example…feel free; Just pay attention because this is the name you will have to use as a parameter in your app;

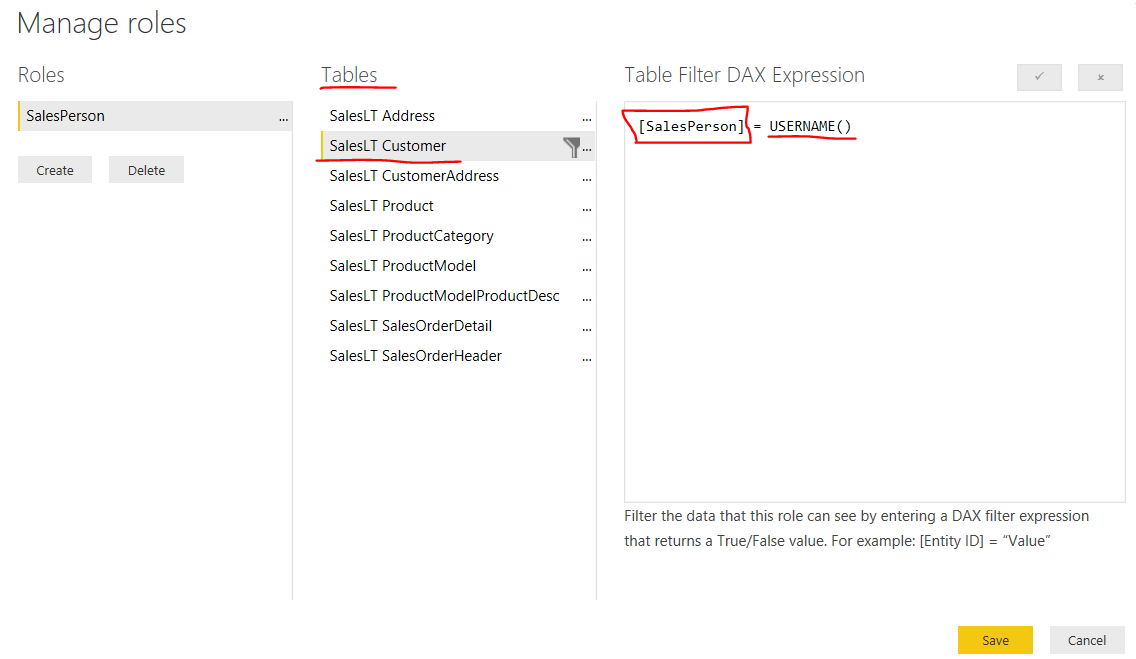


* + The table where you have a column containing information about users you can use to filter the data. That is the table where you must setup the filter using the function “USERNAME()”. It will be the variable containing the parameter passed by your app; In my case:

My sample table in the database, has the column SalesPerson, where the data about the user’s name are persisted:

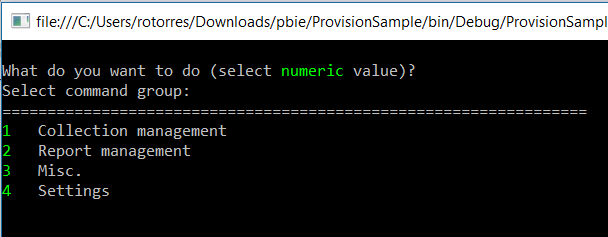


And so, at the configuration of the role filter, I’m using the syntax [SalesPerson]=USERNAME(). Considering that [SalesPerson] is the name of the field in my table where the name of the users are in, and, USERNAME() is the function that will receives the parameter I’m going to input through my app;

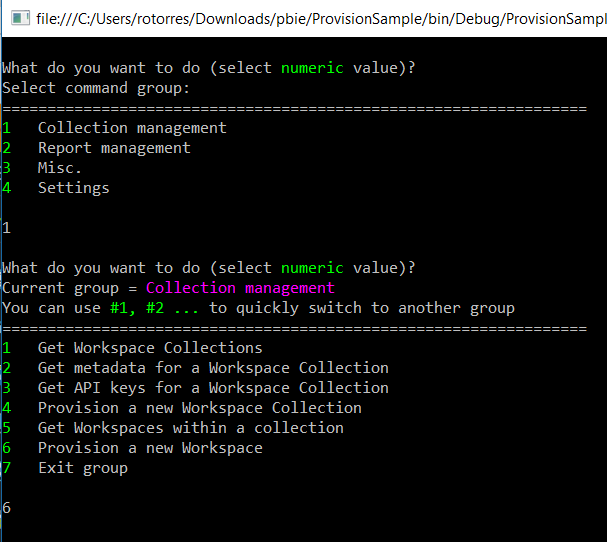


4 – Save your Power BI file, and go ahead with the upload:

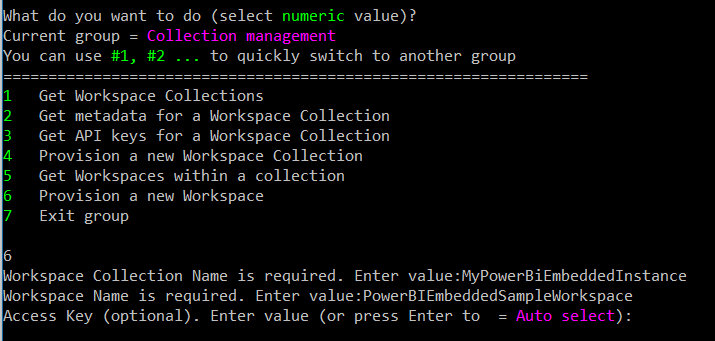
* Download the project available at the address <https://github.com/Azure-Samples/power-bi-embedded-integrate-report-into-web-app/tree/master/ProvisionSample>. Using Visual Studio, compile and execute it. The dialog below should appear:



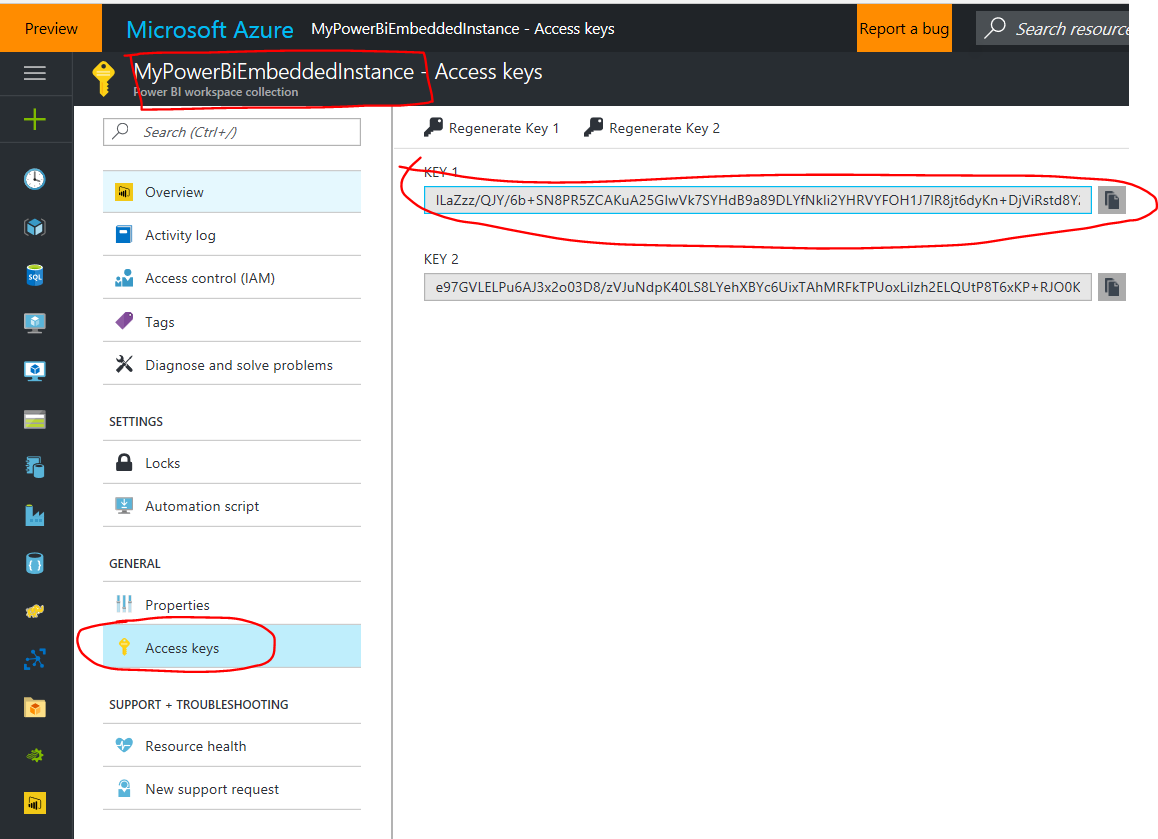
* Type 6



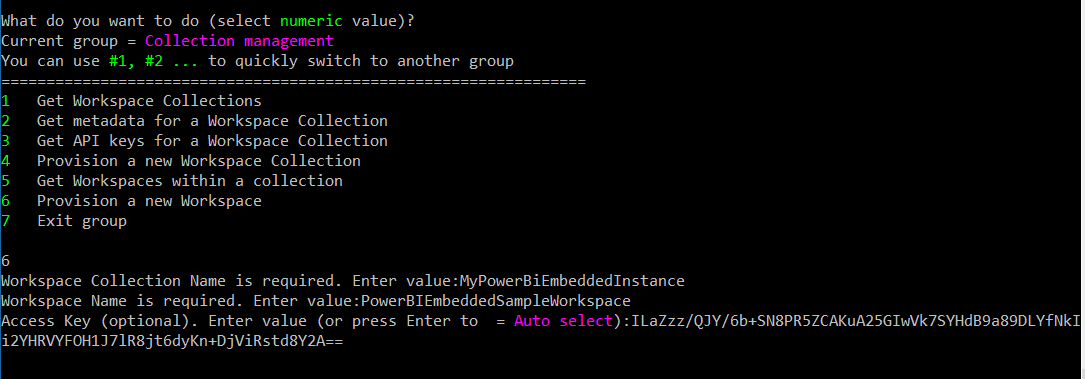
Inform the workspace collection name (the name you collected at the topic 2), and the name you want to use to give for your new workspace area:



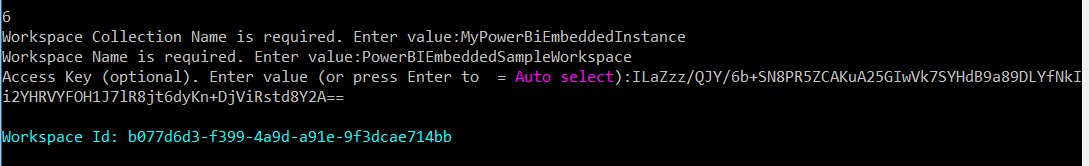
Get the access key: back to your Power BI Embedded instance at the Azure Portal, and locate the menu item “Access Keys”. The window containing your access keys should be showed. Copy the Key1:



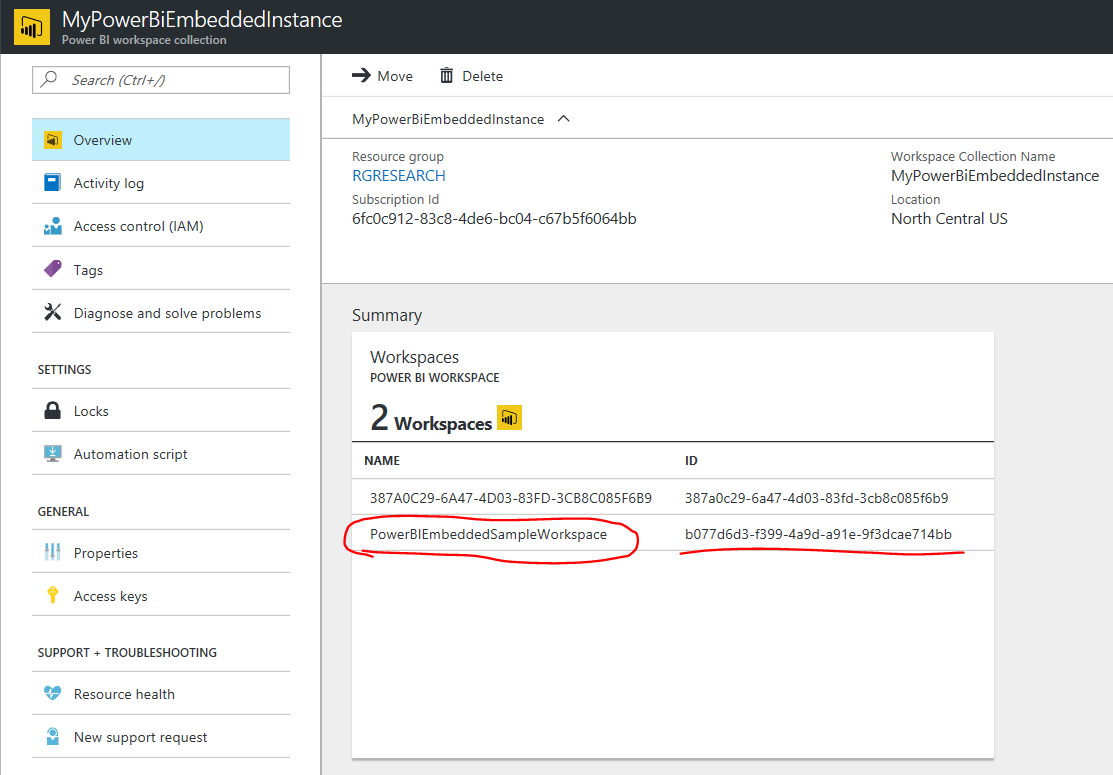
Past the access key at your prompt window and press Enter:



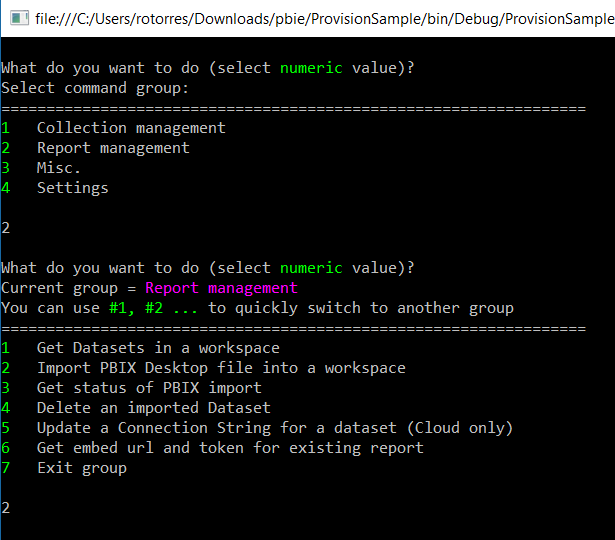
A new collection should be created at your instance:



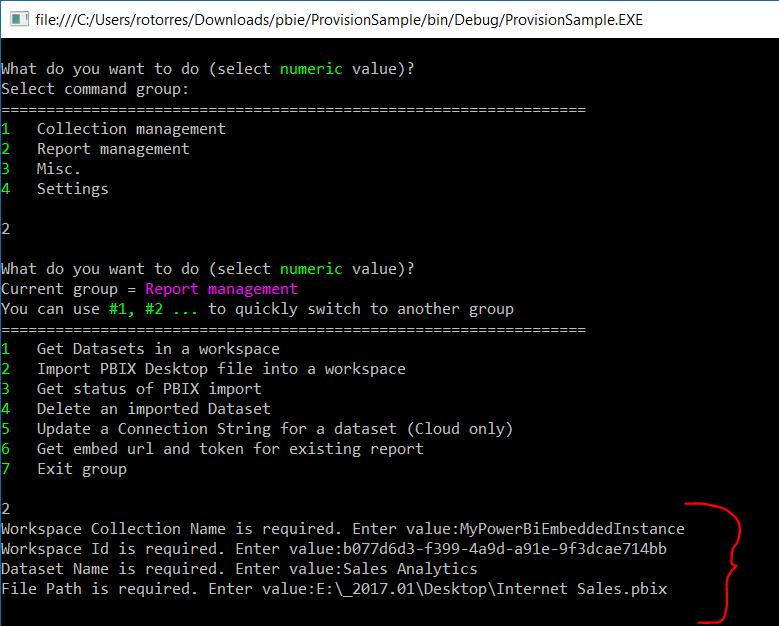
Back to your Azure Portal and go to the Power BI Embedded “Instance” (Workspace Collection) and take notes of your new workspace name and ID:



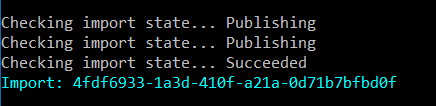
* Back to your prompt window and type 7 (exit group). The menu options will get back to the original options. Now, type 2 (Report Management). And again, 2 (Import PBIX…):



* Fill the new requested fields:
  + Workspace collection name: your power bi embedded instance name;
  + Workspace ID: the generated id for the created workspace in the precedent topic
  + Dataset Name: the name you want to use for the cached dataset on your report. It’s informational for a while, and it’s similar to the names used in the datasets at the Power BI portal when you publish a report using Power BI Desktop;
  + File Path: the full path to reach your Power BI file (pbix);

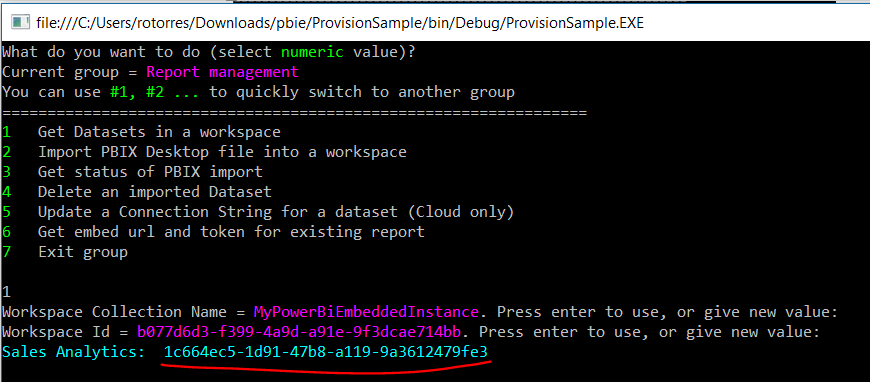


* + Press ENTER. If everything works fine, you will see messages like this:

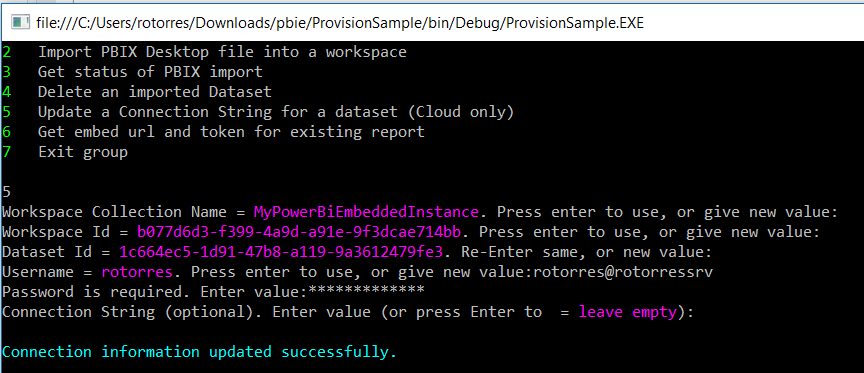


Your report is published now.

* Get the new dataset ID: Now, still in the Report Management group, type 1 (Get Datasets in a workspace). It will request or show you the already done configurations for Workspace Collection Name and Workspace ID. Confirm pressing ENTER and the existent Datasets must be showed. Take note of the ID

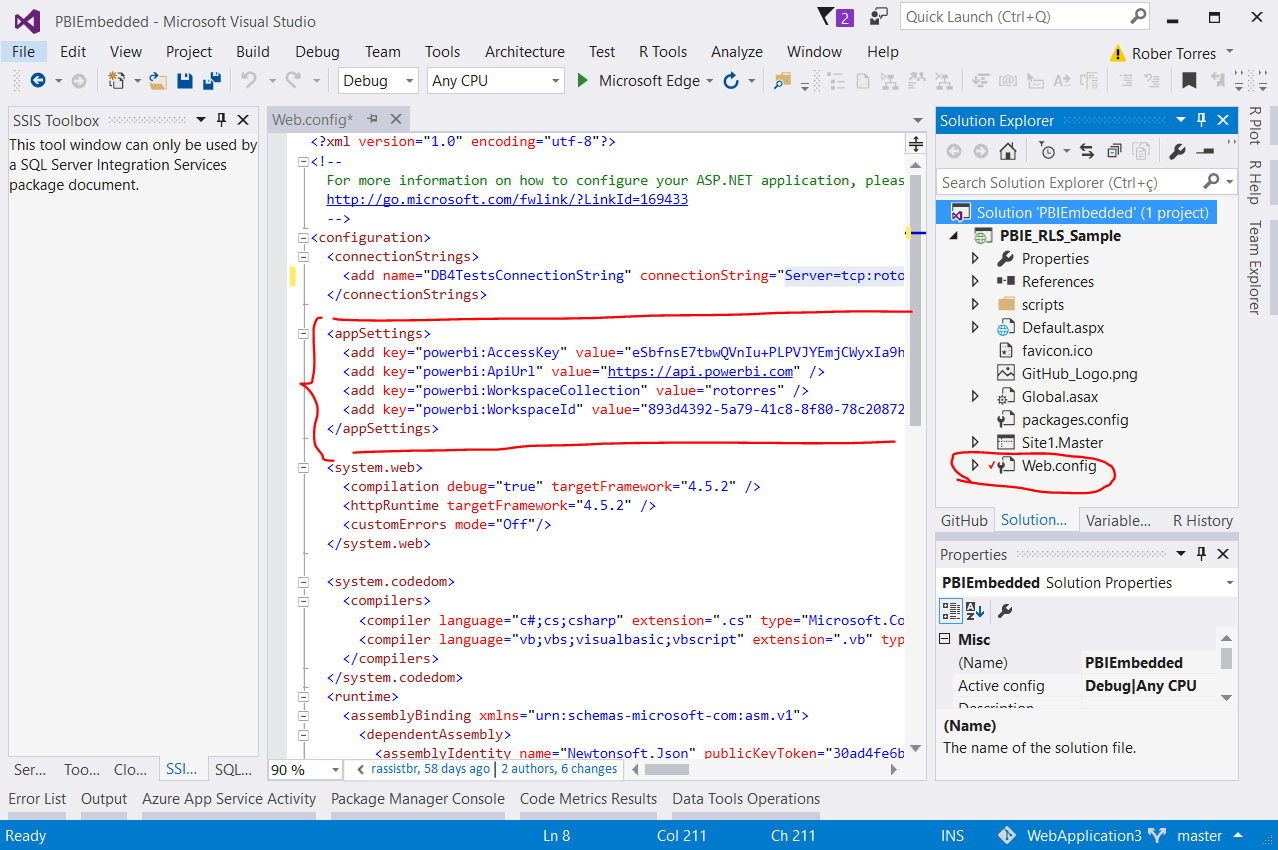


* Update the connection string: Now, still in the Report Management group, type 5 (update a connection string for a dataset) and press ENTER. It will request or show you the already done configurations for Workspace Collection Name, Workspace ID and Dataset ID. If not, fill the fields using the collected info in the past steps. Confirm pressing ENTER and inform the credentials (username and password) to access the database you are using at your Power BI file. For the connection string (next question), go ahead leaving it empty pressing ENTER:



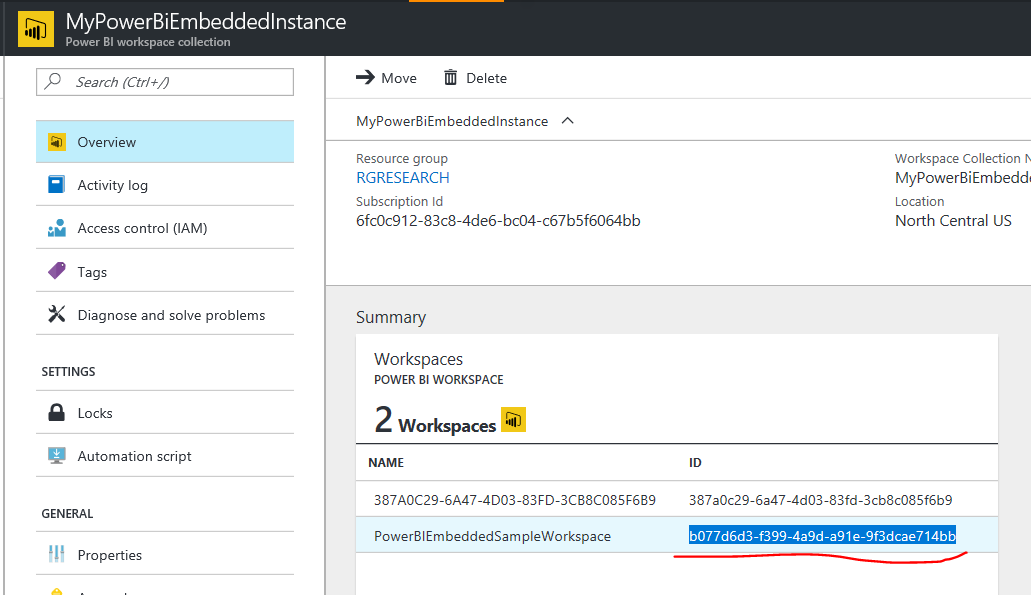
5 - Configure your application:

* Open the Sample app using Visual Studio and open the web.config file:



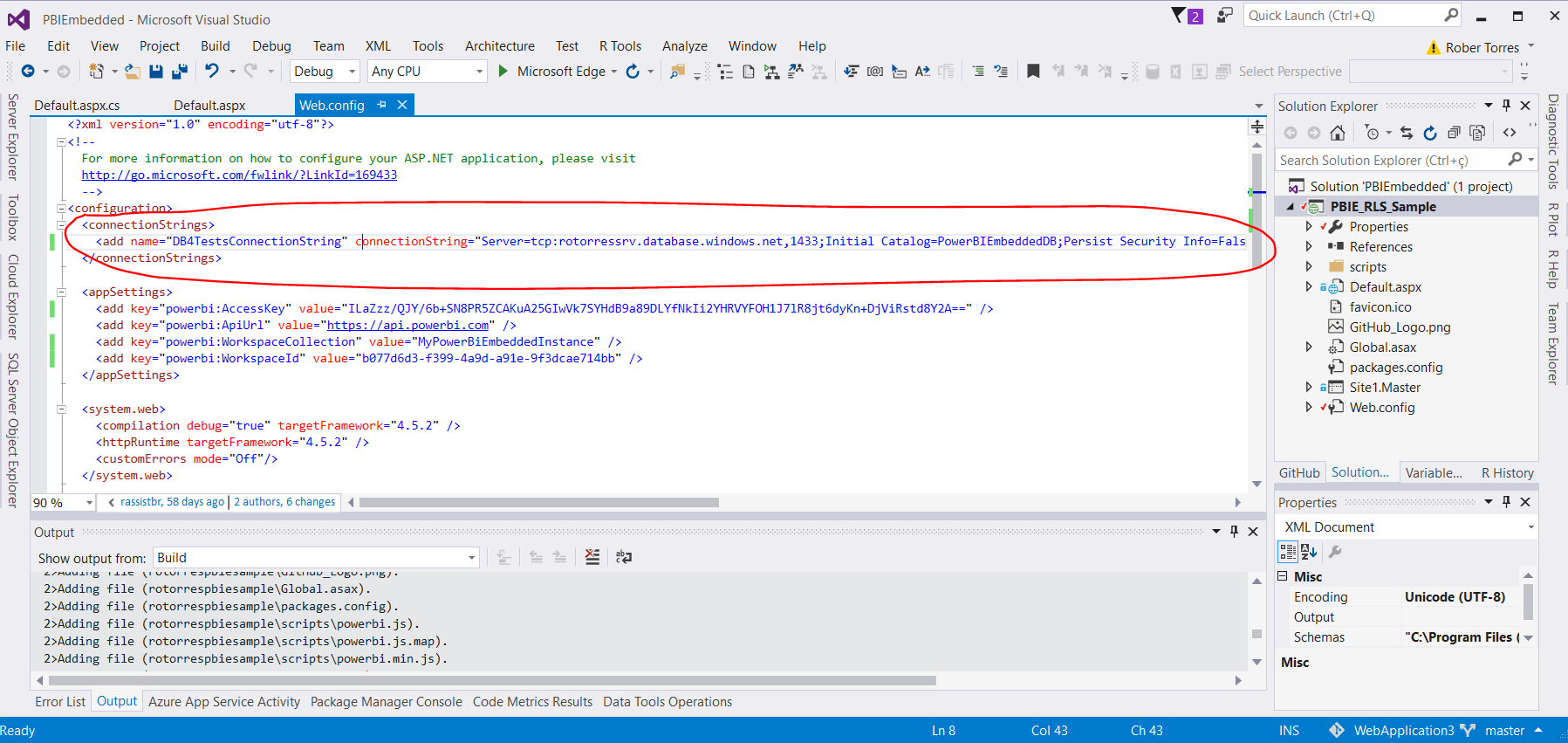
At this file, you have to locate and configure the setting keys:

* + AccessKey = the key you have collected at the beginning of the topic 4;
  + ApiURL = it’s default. You don’t need to change anything;
  + WorkspaceCollection = the workspace collection name (instance) you have collected at the topic 2;
  + Workspace ID = the ID you have collected during the topic 4:

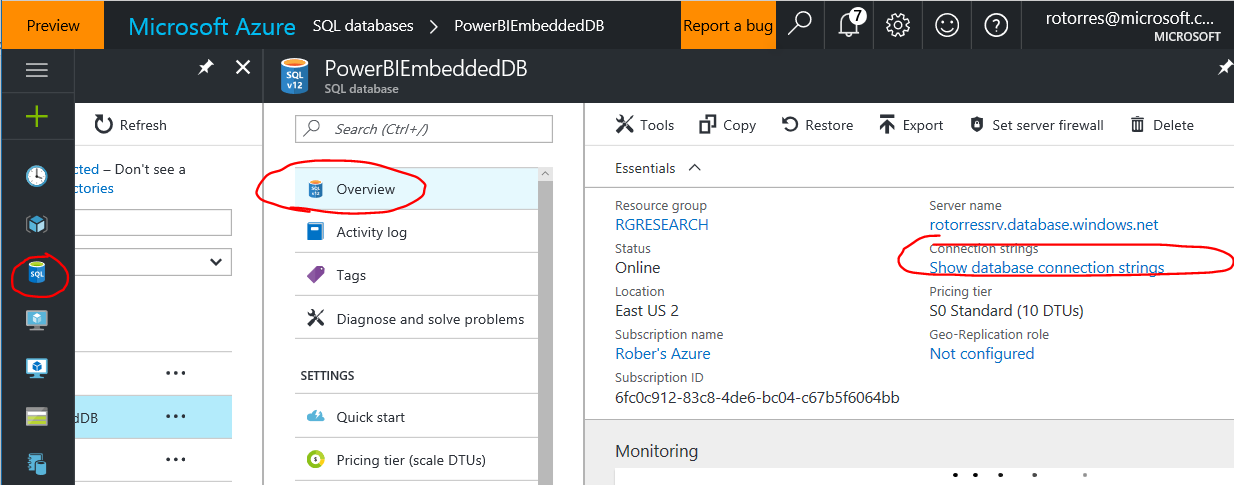


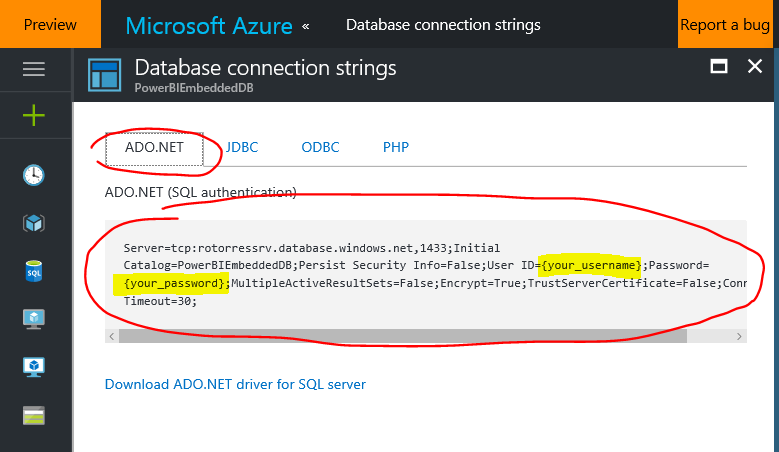
6 – Set your connection string properly:

* Back to the file web.config, locate and configure the setting key for the connection string DB4TestsConnectionString:



* To get the correct Connection String, back to the Azure Portal and locate the database you are using at the sample:





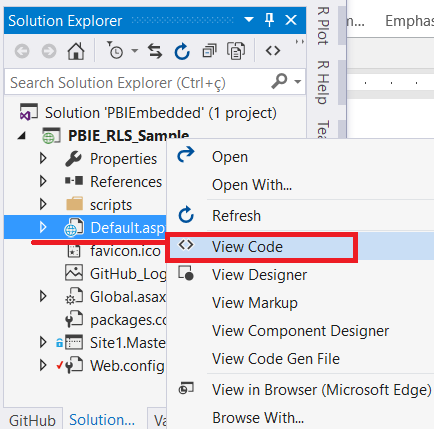
Don’t forget to fix User ID and Password:

In my case:

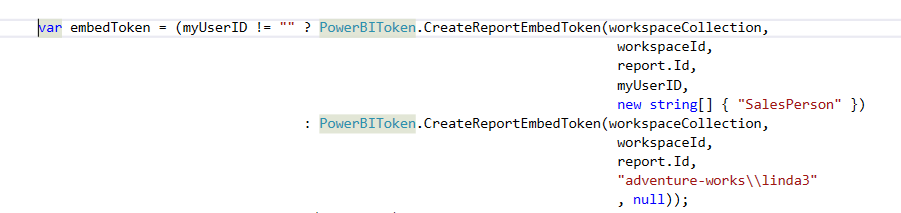
Server=tcp:rotorressrv.database.windows.net,1433;Initial Catalog=PowerBIEmbeddedDB;Persist Security Info=False;**User ID=rotorres**;**Password=R83ef042@2016**;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

7- Check the app code:

* Open the code of the file Default.aspx:



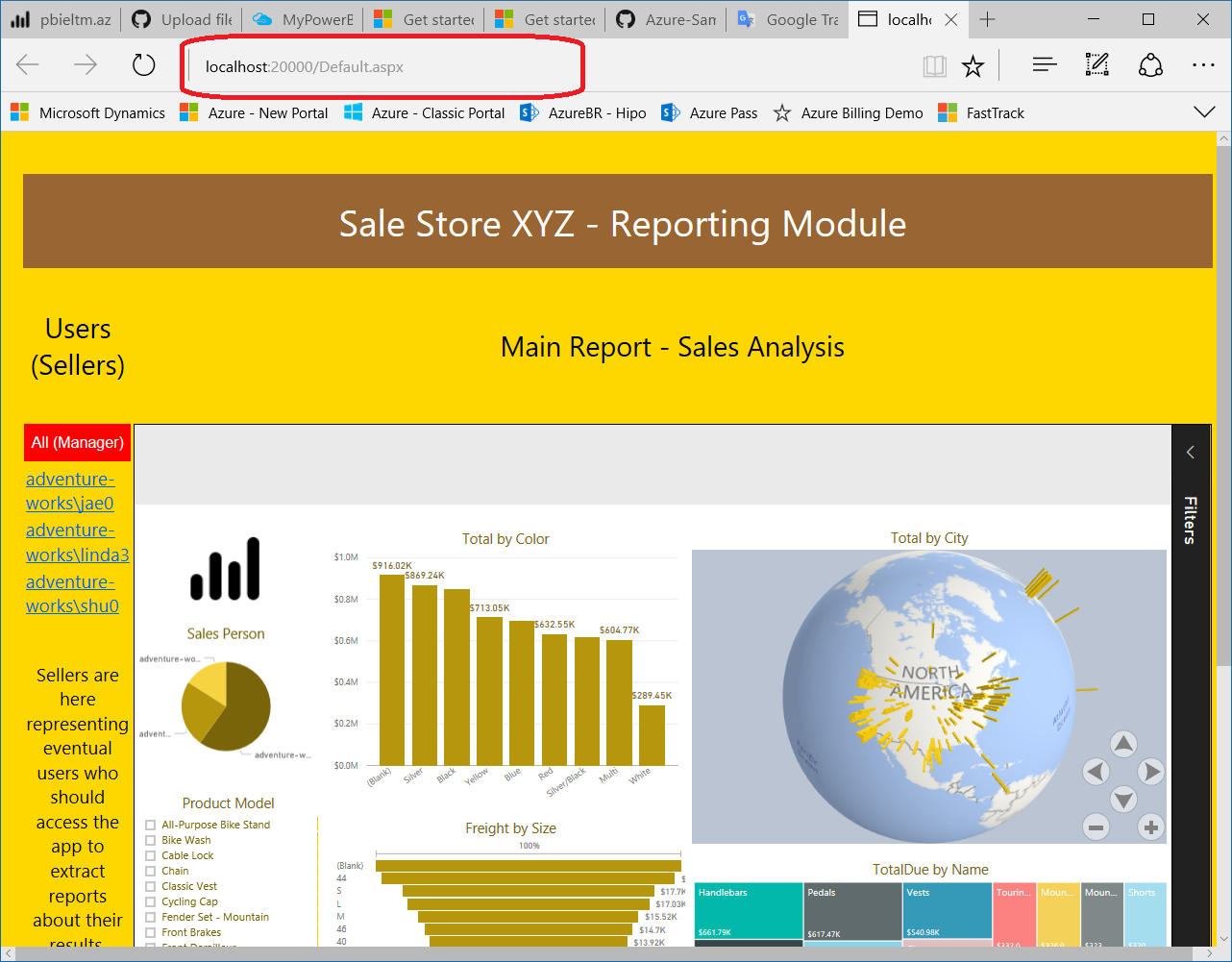
* The main code is in the line 57:



This code just evaluate if a UserName was collected, and so, if it’s different of “” (empty), we pass its value as parameter to the Power BI method that render our report. And so, 2 parameters are important: the USERNAME, what’s represented by the variable myUserID, and the name of the role (what we have configurated directly in the Power BI file, on the topic 3. In my case, the role name is “SalesPerson”.

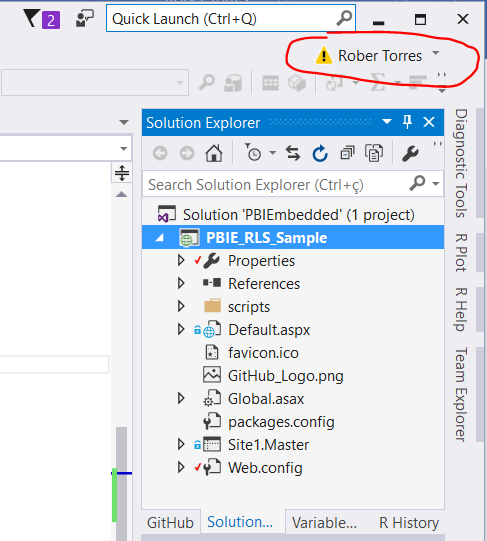
If the username isn’t important, for example, simulating an access of a Super User member, just to avoid any filter giving the user full access to the data, we can execute the code to render the report just passing null as a role name parameter. So, as username, you can input anything… in my case, I’ve used a registered sales person, but I could use anything… like God, or Evil…

Now, you can compile and execute your project to test:

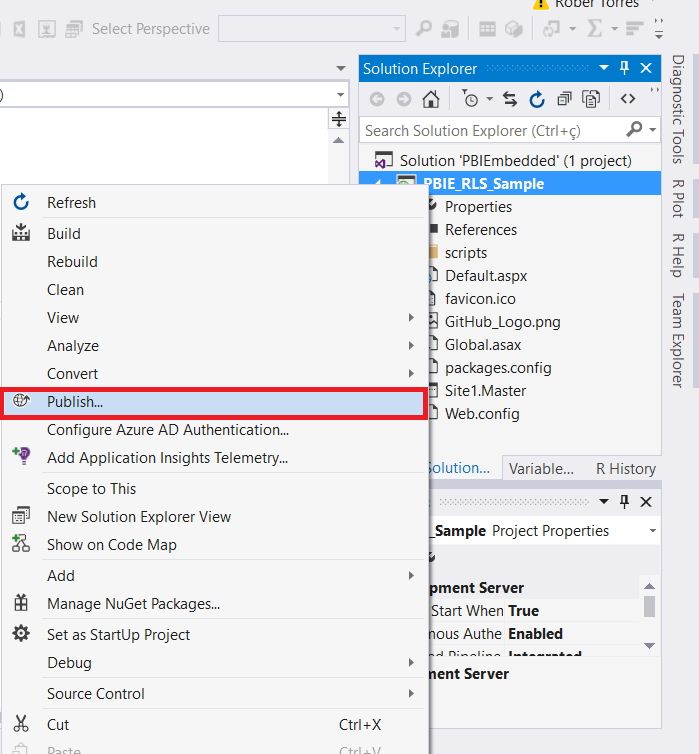


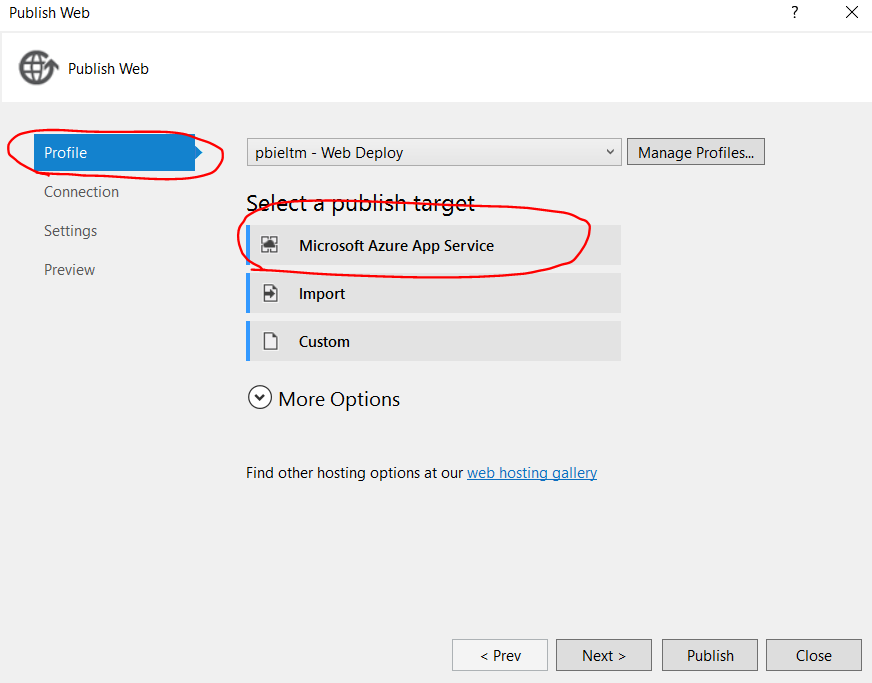
If your test is running well, you can publish your project:

* Check if you are logged in on the Visual Studio:

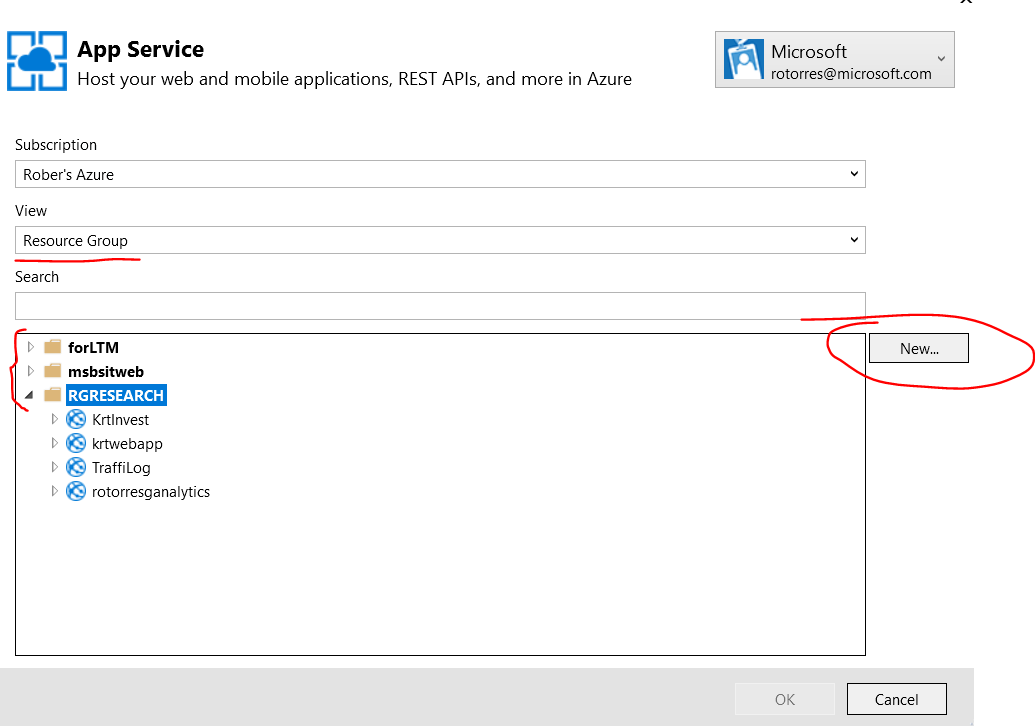


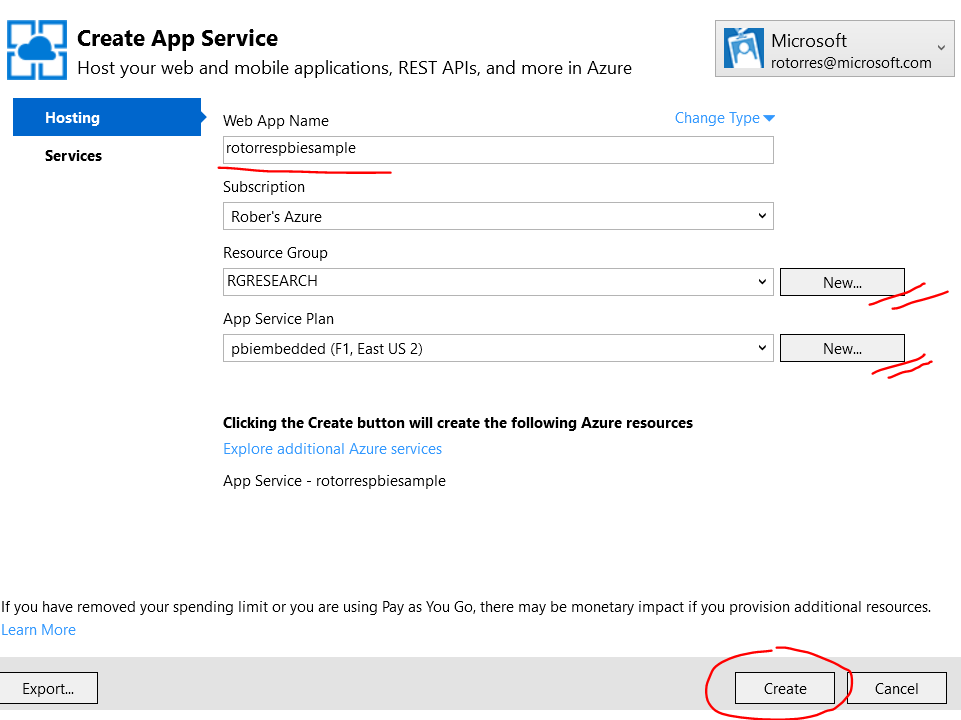
* And, follow the actions to publish the app as an Azure Web App:

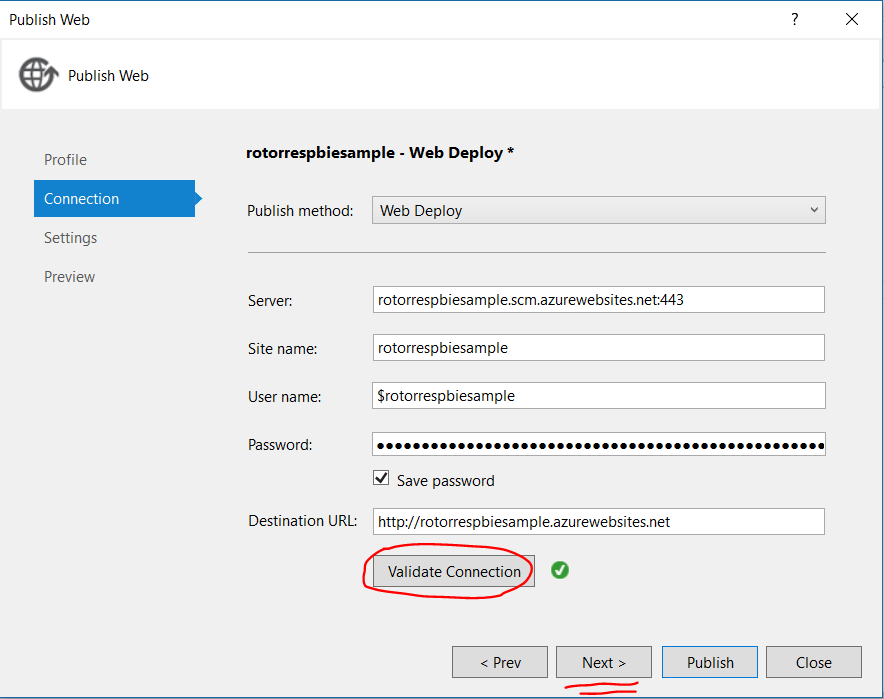


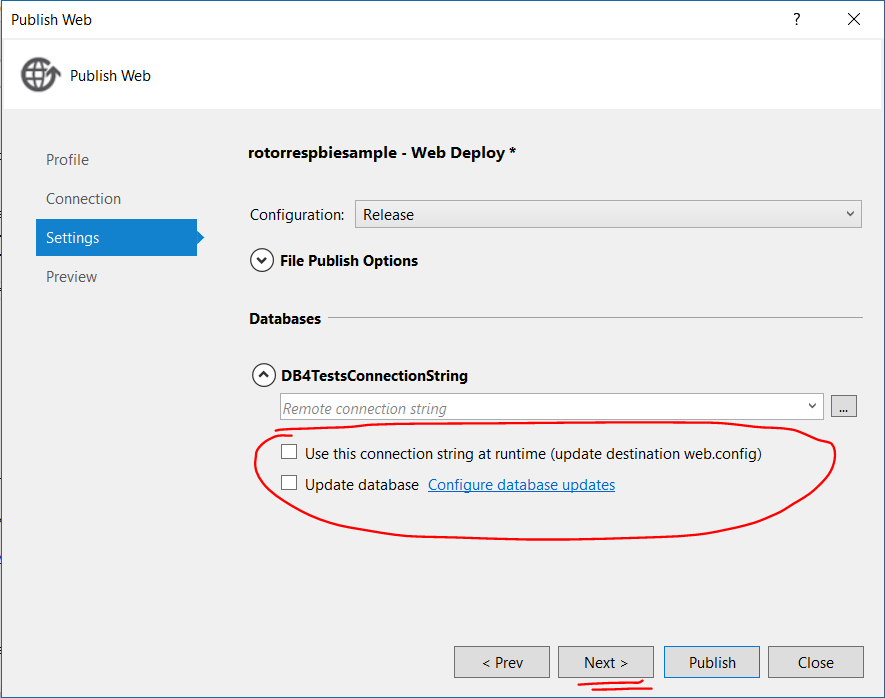


Choose an existent resource group and create a new app

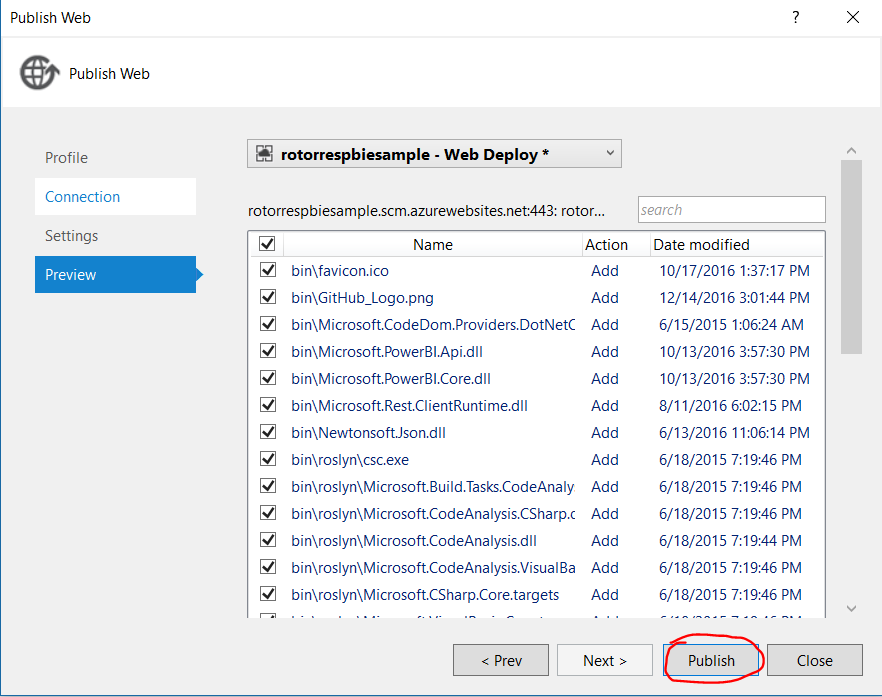




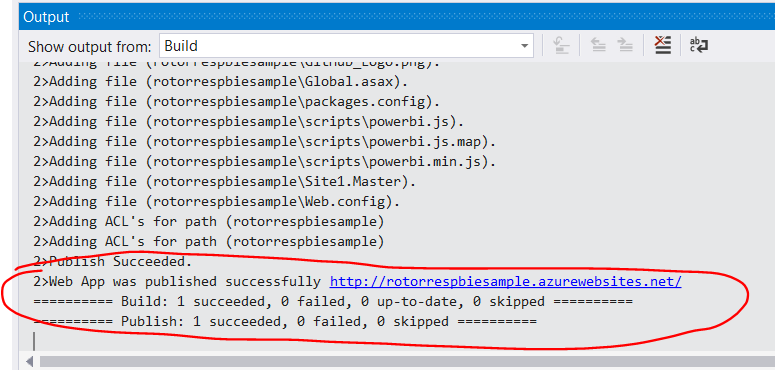




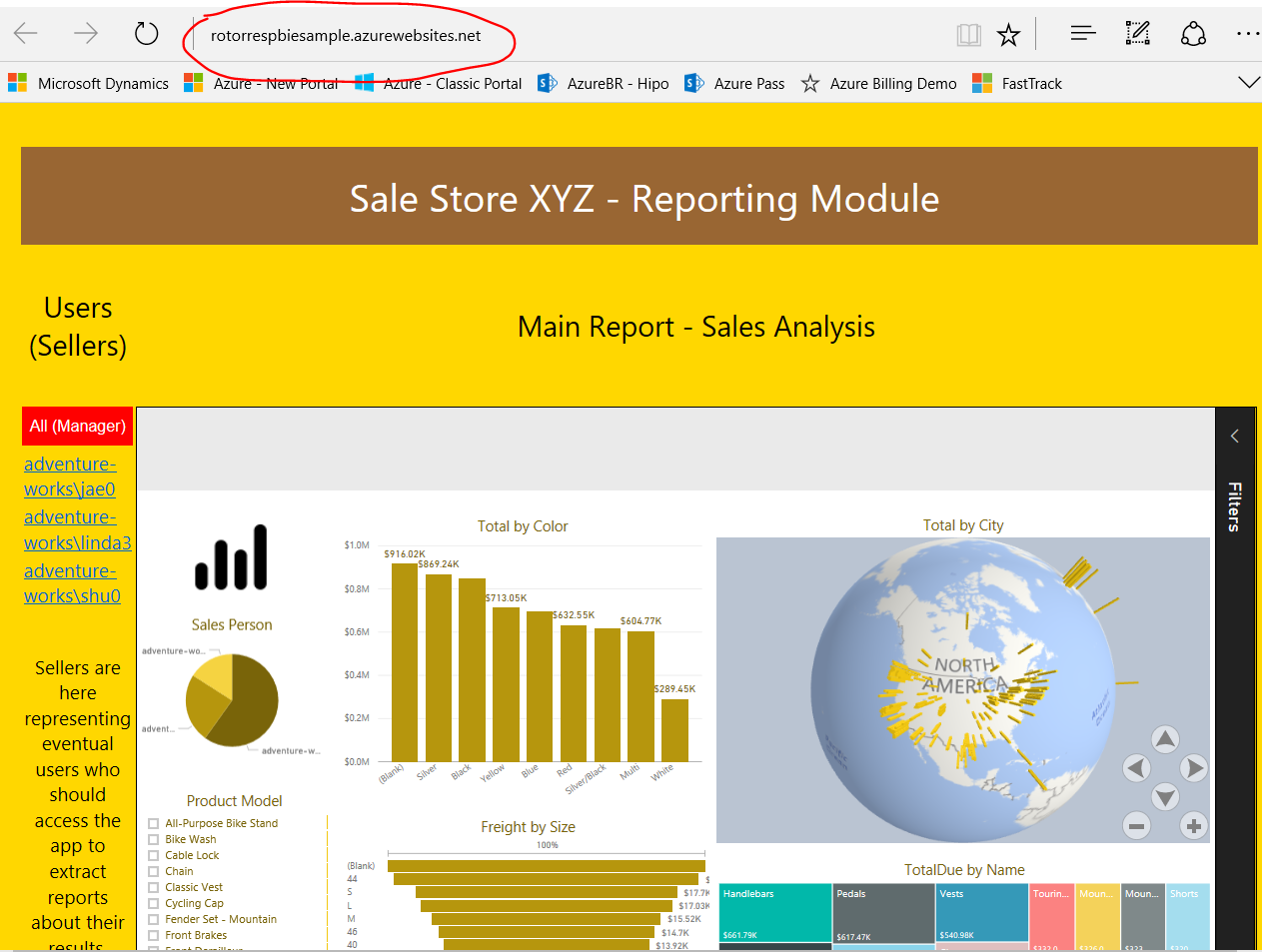
Click at the Preview Button at the next screen, and so, Publish



At Visual Studio Output frame you should see a success message like that:



Go to your new website address and check if everything is working fine:



Feel free to get in touch: **rotorres@microsoft.com**