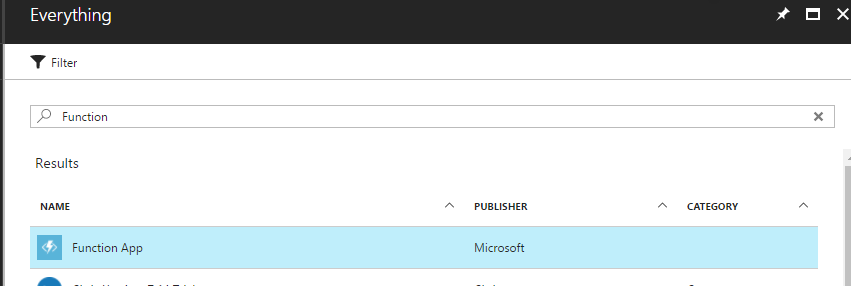
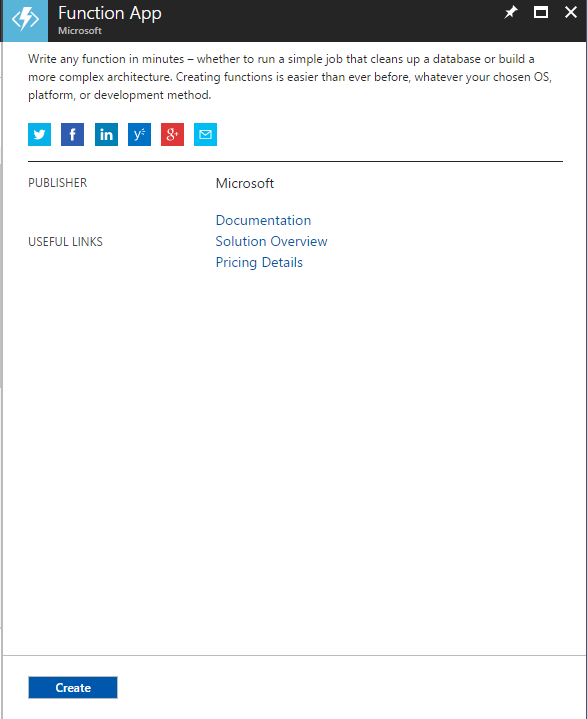
# Creating api using azure functions

## Email Service

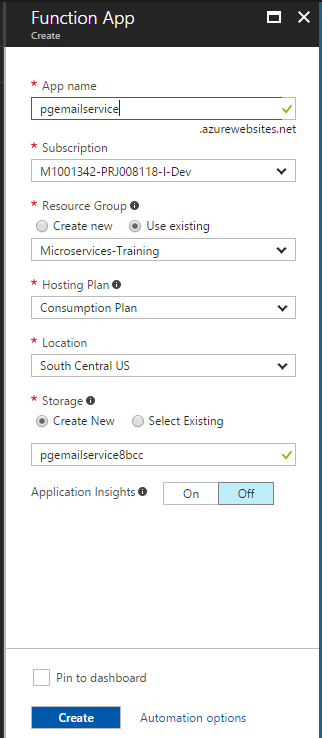
1. Login to Azure Portal with contributor and search for ‘Function App’ under specific resource group.



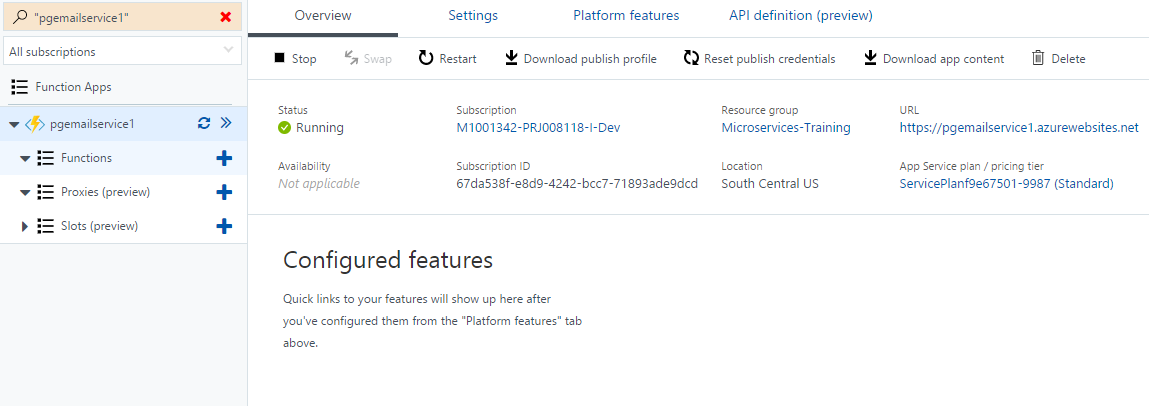
1. Click on ‘Create’ button as in below screen.



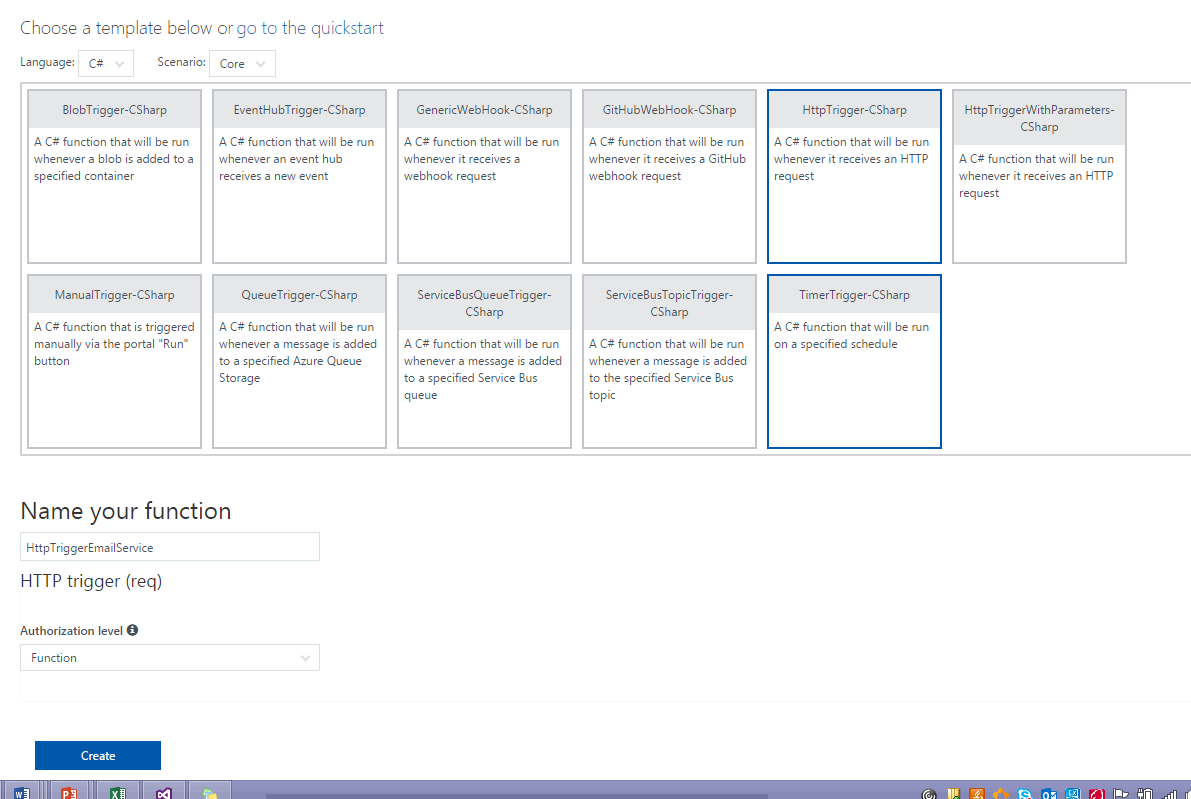
1. Enter Function details and click ‘Create’
   1. App Name
   2. Subscription – default
   3. Resource group – default
   4. Hosting plan – as per requirement
   5. Storage name



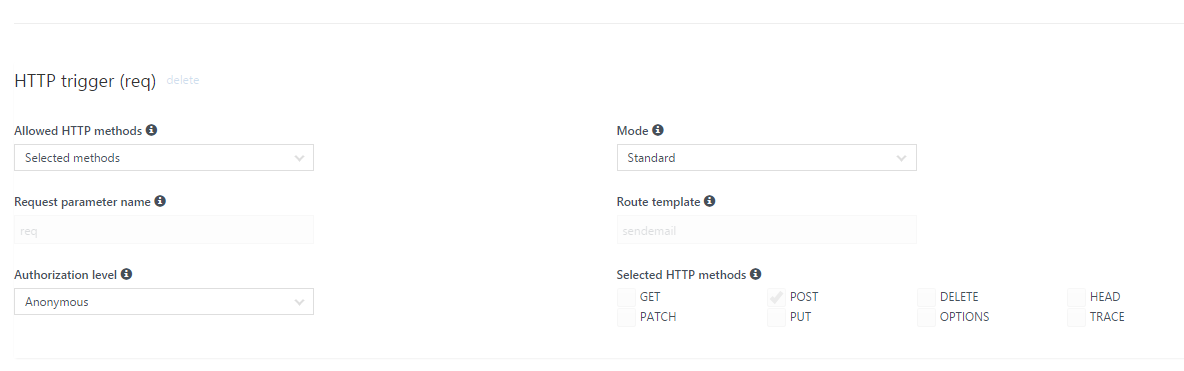
1. Below screen appears post successful creation.



1. Select ‘C#’ as language and HTTPTrigger as function type. Enter below fields:
   1. Name of your function
   2. Authorization Level - Function



1. Navigate to <Name of your function> -> Integrate.
   1. Select:
      1. Allowed HTTP methods – selected methods
      2. Required parameter name
      3. Route template
      4. Selected HTTP methods – POST (in this case)



1. Click on function name and edit ‘run.csx’. Enter code to send emails logic. Copy and paste below highlighted codes:

*using System.Net;*

*public static async Task<HttpResponseMessage> Run(HttpRequestMessage req)*

*{*

*bool output = false;*

*// parse query parameter*

*string fromemail = req.GetQueryNameValuePairs()*

*.FirstOrDefault(q => string.Compare(q.Key, "fromemail", true) == 0)*

*.Value;*

*string toemail = req.GetQueryNameValuePairs()*

*.FirstOrDefault(q => string.Compare(q.Key, "toemail", true) == 0)*

*.Value;*

*string subject = req.GetQueryNameValuePairs()*

*.FirstOrDefault(q => string.Compare(q.Key, "subject", true) == 0)*

*.Value;*

*string body = req.GetQueryNameValuePairs()*

*.FirstOrDefault(q => string.Compare(q.Key, "body", true) == 0)*

*.Value;*

*// Get request body*

*dynamic data = await req.Content.ReadAsAsync<object>();*

*// Set name to query string or body data*

*fromemail = fromemail ?? data?.fromemail;*

*toemail = toemail ?? data?.toemail;*

*subject = subject ?? data?.subject;*

*body = body ?? data?.body;*

*output = SendEmail(fromemail, toemail, subject, body);*

*return output == false*

*? req.CreateResponse(HttpStatusCode.BadRequest, "Please pass a name on the query string or in the request body")*

*: req.CreateResponse(HttpStatusCode.OK, "Email sent successfully.");*

*}*

*private static bool SendEmail(string fromemail, string toemail, string subject, string body)*

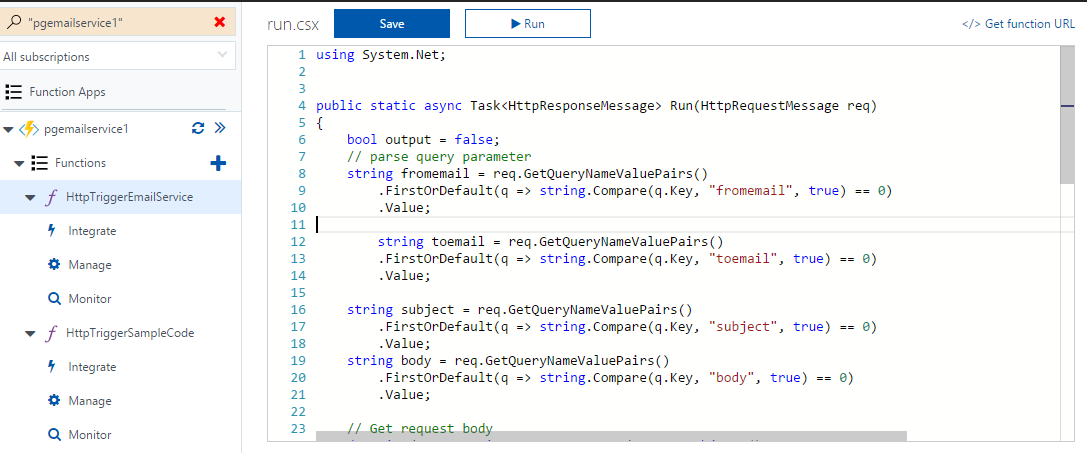
*{*

*//logic to send email*

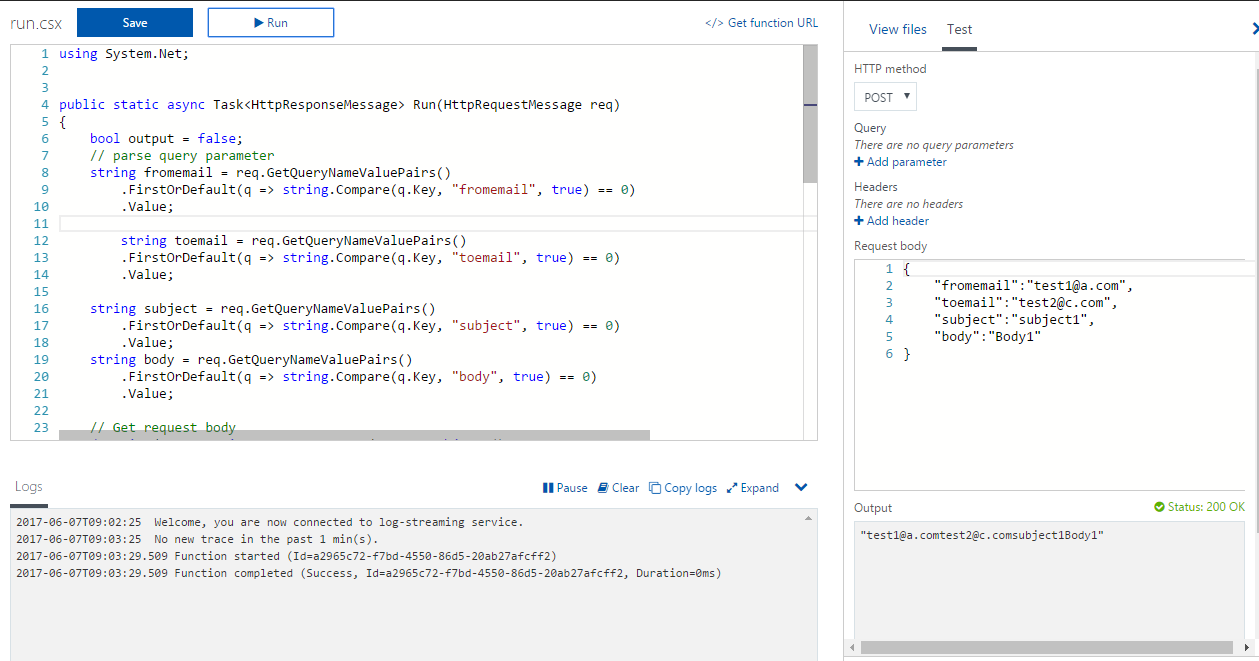
*bool isEmailSuccess = true;*

*return isEmailSuccess;*

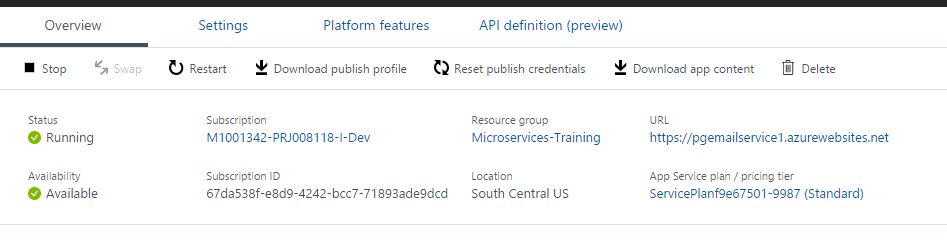
*}*



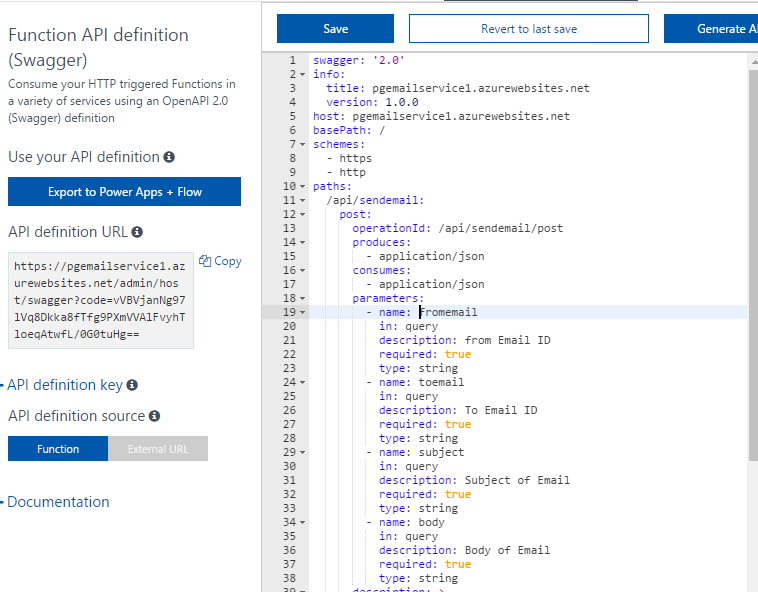
1. Click Save and Run to check the outcome of the function. Output status code 200 means operation is successful.



1. Adding Swagger metadata, from Azure Functions landing page, click on ‘API Definition (Preview)’.

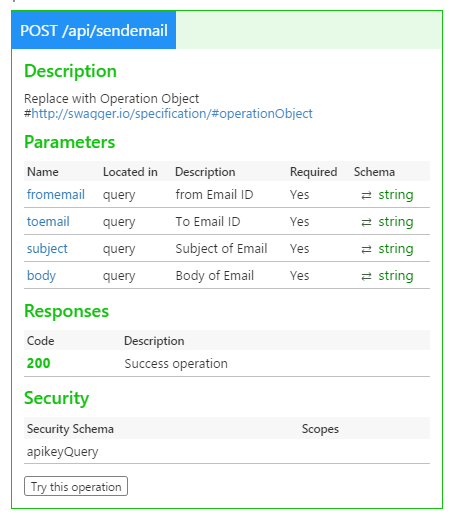


1. Click on ‘Generate API Definition template’ and replace below properties:



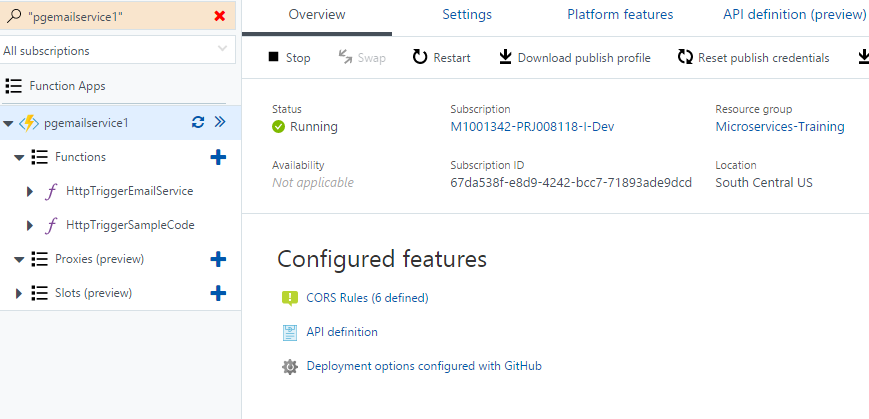
|  |  |
| --- | --- |
| operationId | /api/sendemail/post |
| produces | application/json |
| consumes | application/json |
| parameters | - name: fromemail  in: query  description: from Email ID  required: true  type: string  - name: toemail  in: query  description: To Email ID  required: true  type: string  - name: subject  in: query  description: Subject of Email  required: true  type: string  - name: body  in: query  description: Body of Email  required: true  type: string |

1. Click Save.
2. Try the swagger metadata from panel as:

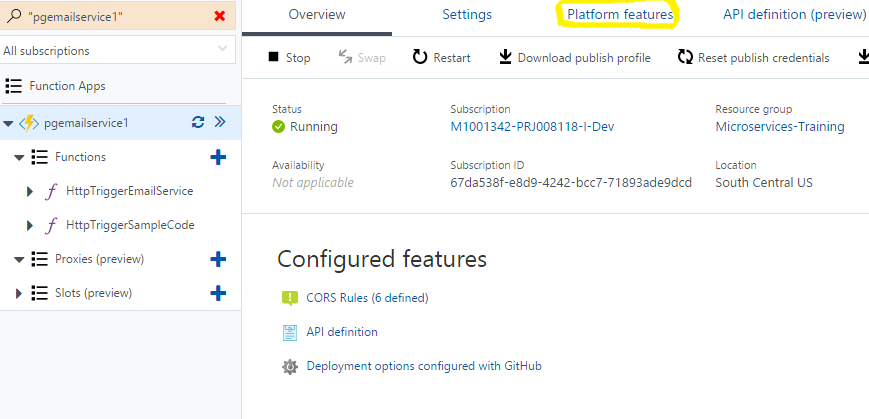


# Configure Continous deployment (if code is already checked in github)

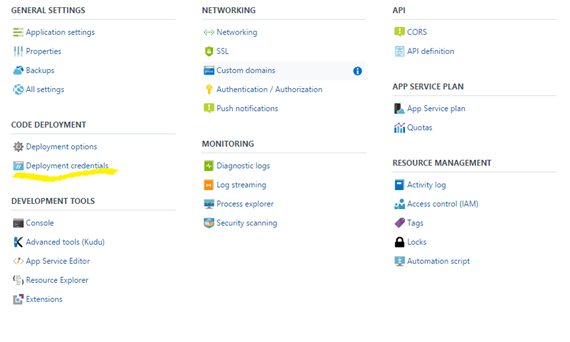
1. Login to Azure Portal and Navigate to Azure function ‘pgemailservice1’.



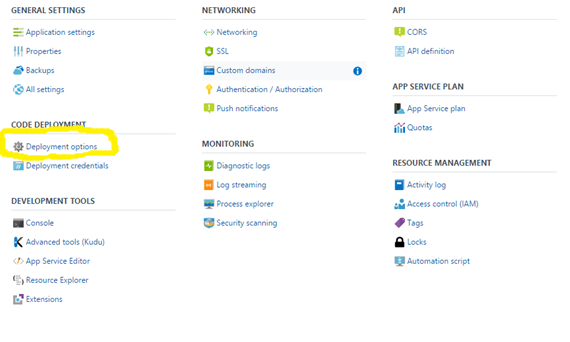
1. Click on ‘Platform features’.



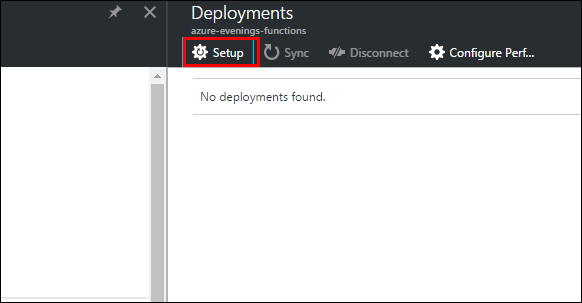
1. Enter deployment credential as and enter the credential needed for deployment.

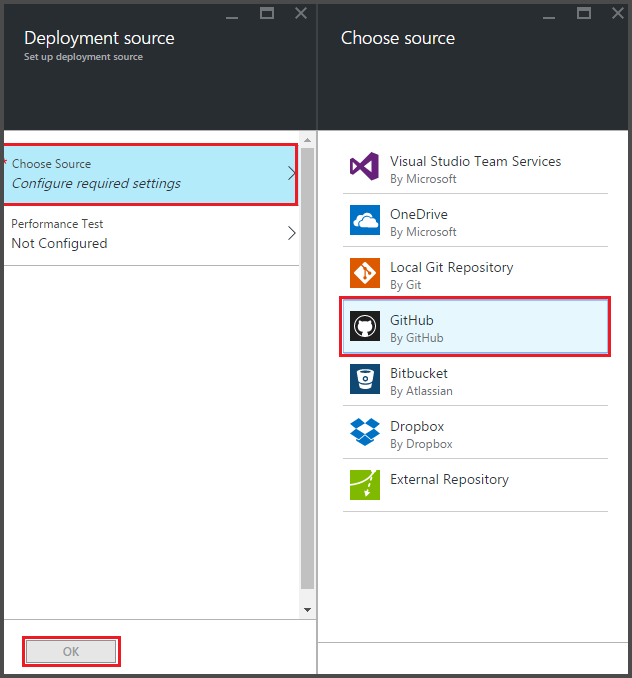


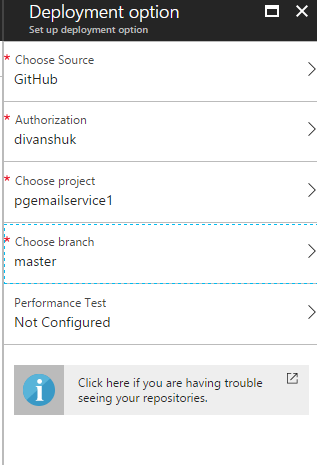
1. Click on Deployment Option as:



1. Follow the below screens for setting up CD:







# References

* <https://docs.microsoft.com/en-us/azure/azure-functions/functions-continuous-deployment>