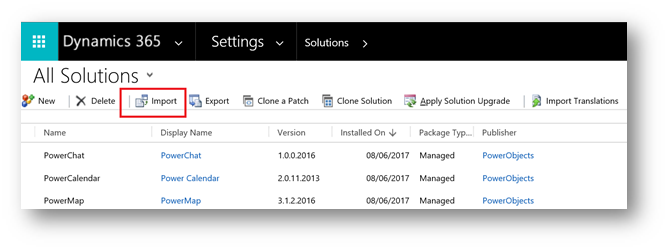
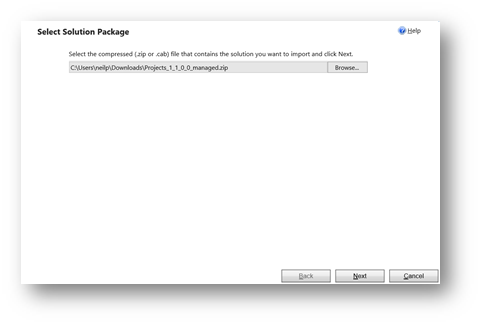
# Prerequisites for CICD Automation tool with DevOps:

## Deploy Solution in Instance

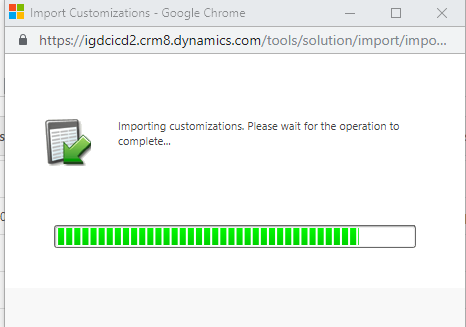
1. Download the attached zip file.



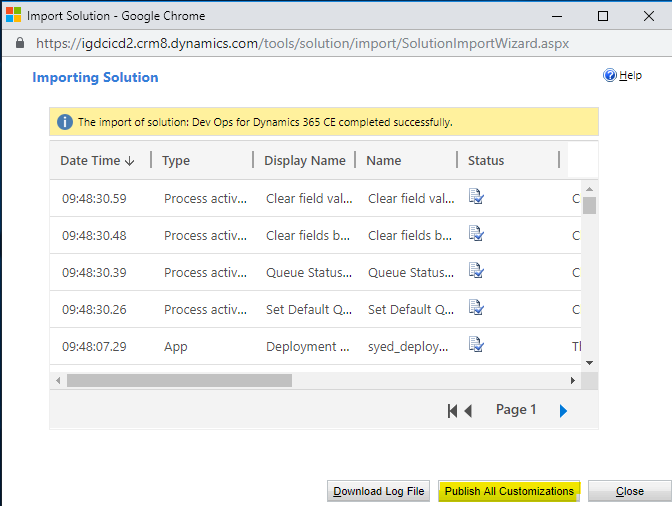
1. Login to Dynamics 365. Navigate to Settings-> Solutions.
2. Select the import option.
3. Now browse for a solutions file and click on Next



1. Click Import button. Then you will see below screen.



1. Once Import completed. Please click on Publish All Customization button.

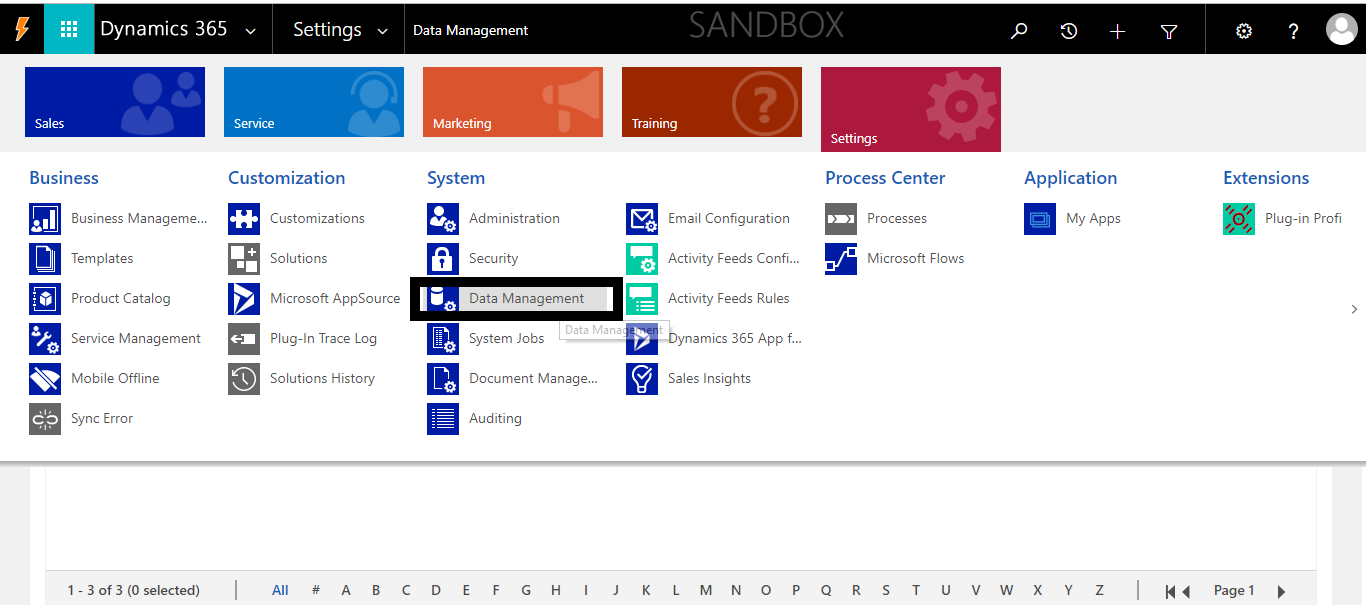


## Configuration Settings.

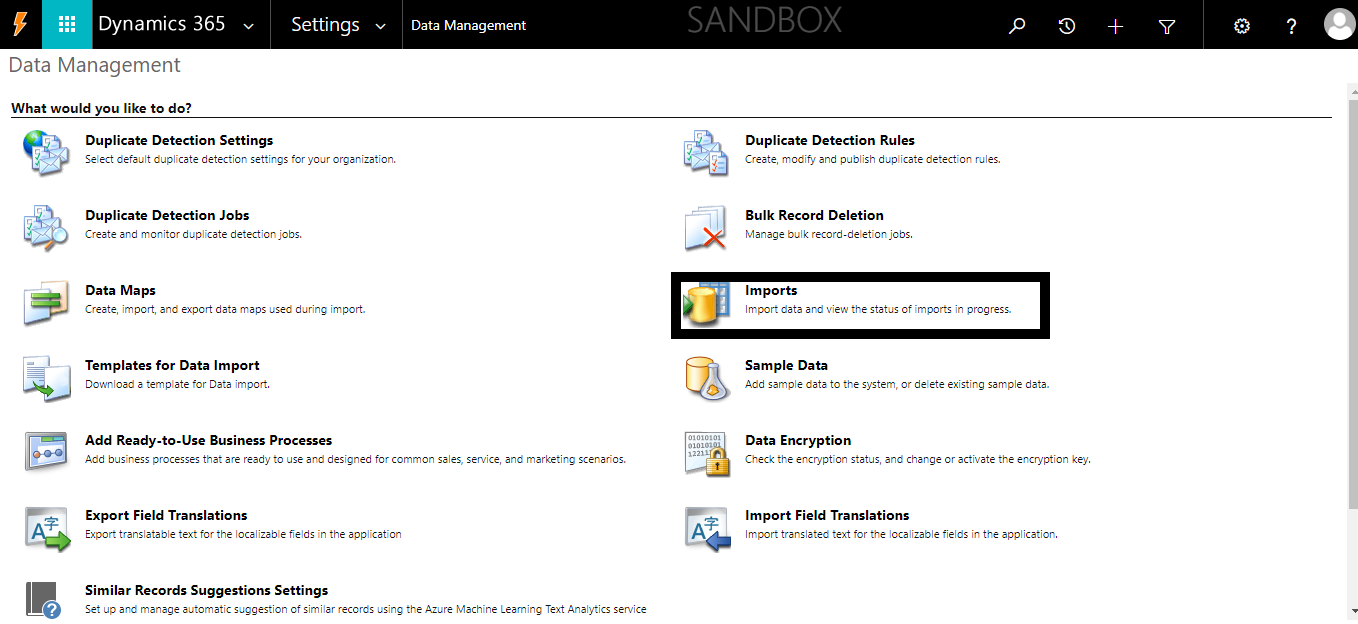
1. Download the attached configuration settings file.



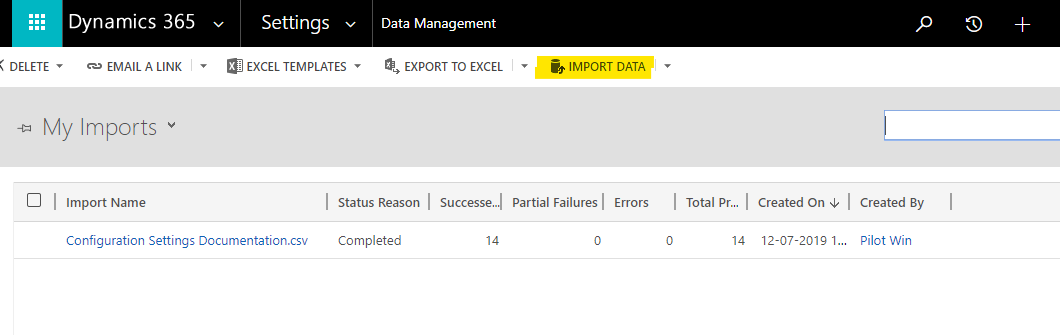
1. Fill the respective values for configuration key and save the file.
2. Once it is saved, open CRM instance in browser and navigate to Navigate to Settings->Data Management->Imports



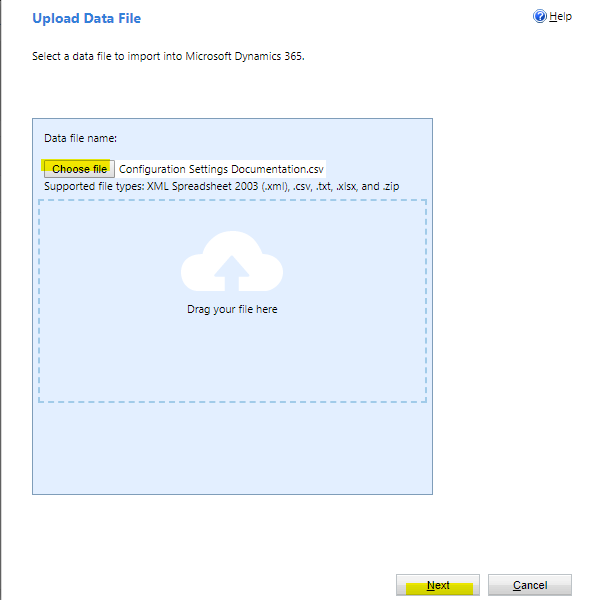
1. Click on Imports.



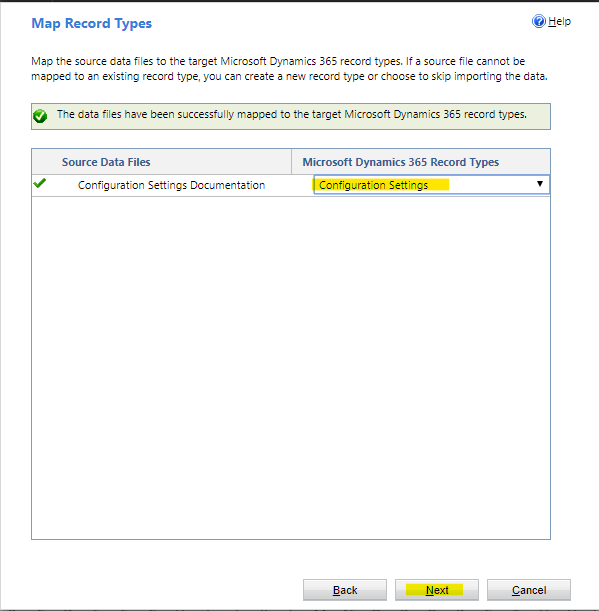
1. In Imports, Now click on Import Data.



1. Please select the Configuration File to be Imported. Then click on next.



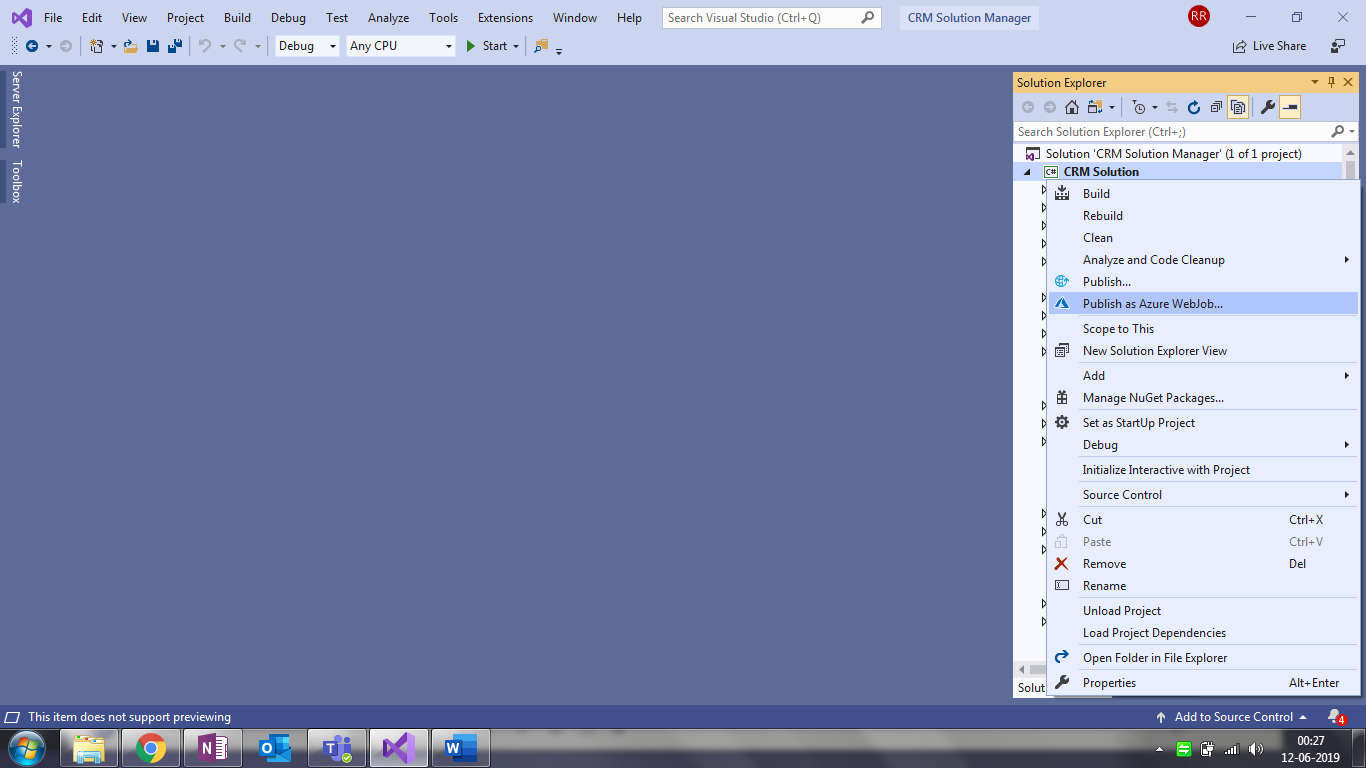
1. In Map Record section. Choose Configuration Settings and click on next and Submit.



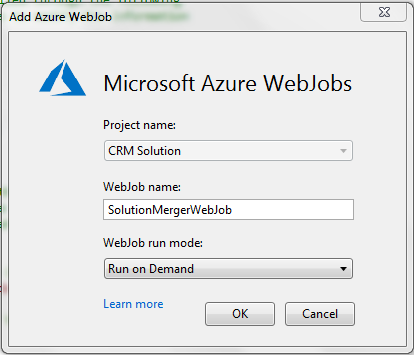
1. Once Data is imported, check the status of the Import.

## Publishing Console Application as an Azure WebJob:

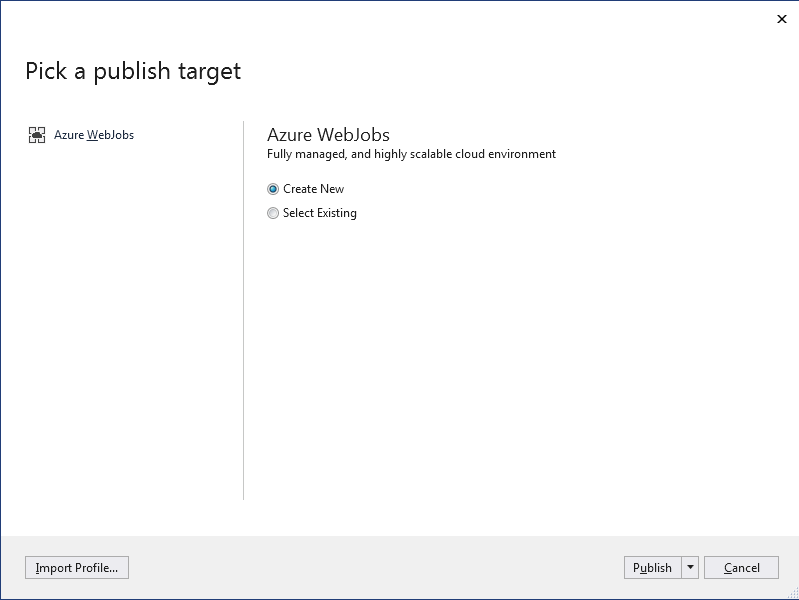
1. Open **‘CRM Solution Manager’** Solution in Visual Studio and provide values where keys are blank in **App.config** file.
2. Build the solution
3. After Successful build, go to Solution Explorer and right click on ‘CRM Solution Manager’ Project and then click on **‘Publish as Azure WebJob’**:



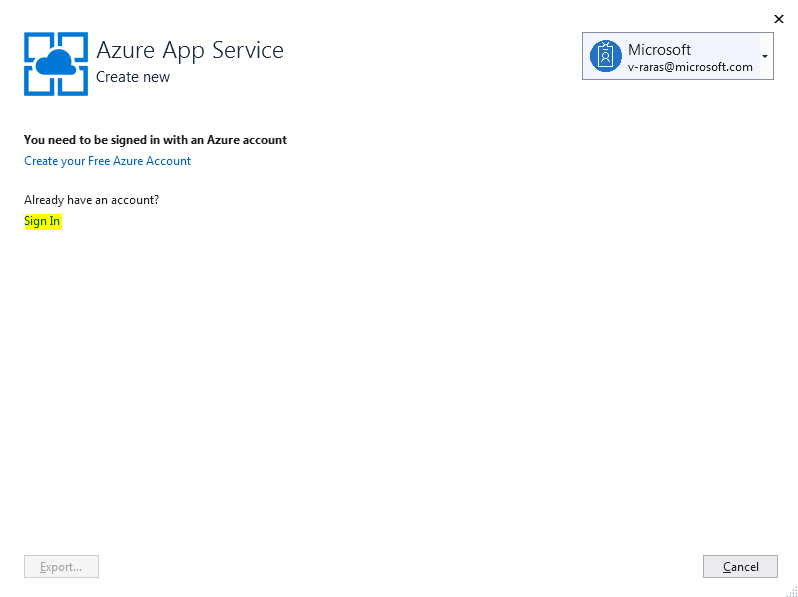
1. On the next screen, under **‘WebJob Name’**, give a name to the WebJob and select **‘WebJob run mode’** as **‘Run on Demand’** and click **OK** button:



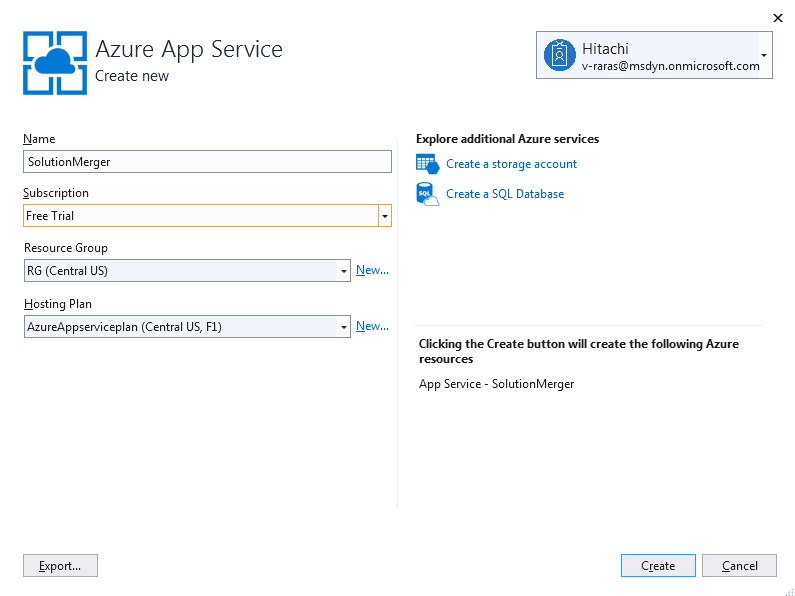
1. On the next screen, select **‘Create New’** and click on **‘Publish’**:



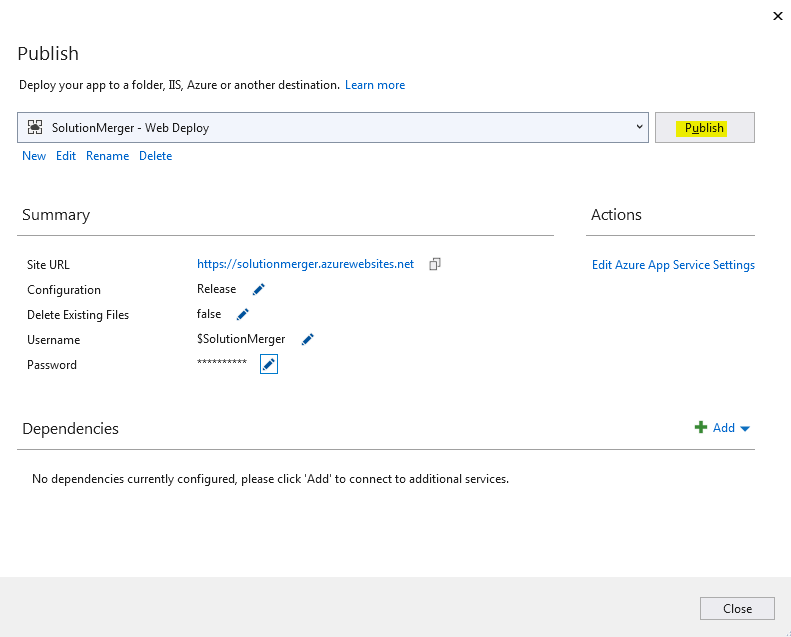
1. On the next screen, sign in to your azure portal by clicking on **Sign In** button:



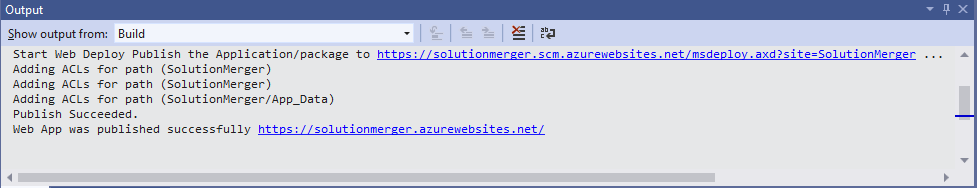
1. On the next screen, under **‘Name’**,give a name to the App Service, select **Subscription** As **‘Free Trial’**, select **Resource Group** for the App Service and click on **Create**:



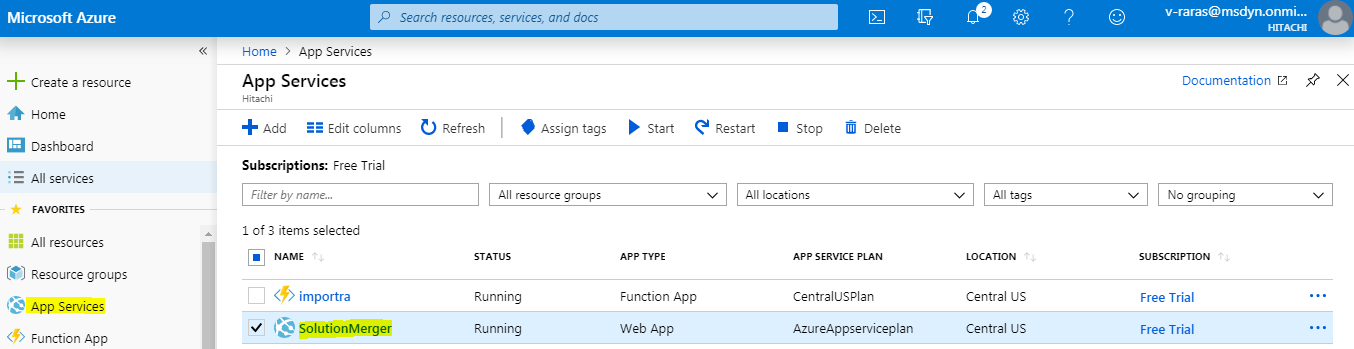
1. On the next screen, click on **Publish**:



This will **Publish the WebJob** **(Web App)** in **Azure**. You can see the output message in **Output Window** in Visual Studio:

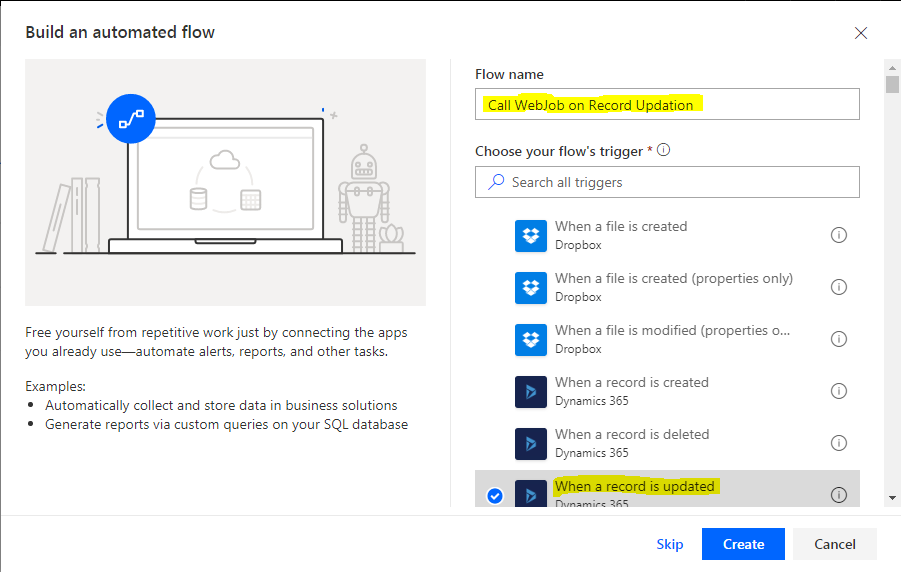


You can login to your Azure Portal and should be able to see your recently created **Webjob** under **App Services** tab:

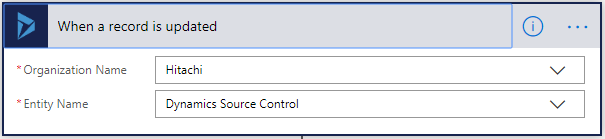


## Creating Microsoft Flow to call Azure WebJob:

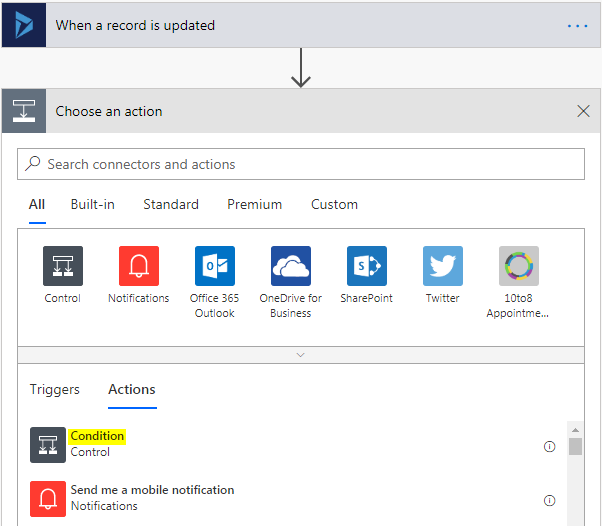
1. Go to [https://india.flow.microsoft.com](https://india.flow.microsoft.com/) and login to flow using CRM credentials where **DevOps** solution is installed.
2. Next, Click on **My Flows** tab under left pane, click on **New** and select **Automated-from blank**.
3. On the next screen, under **Flow Name**, give a name to your flow, under **Choose your flow’s trigger**, select **When a record is updated** and then click **Create**:



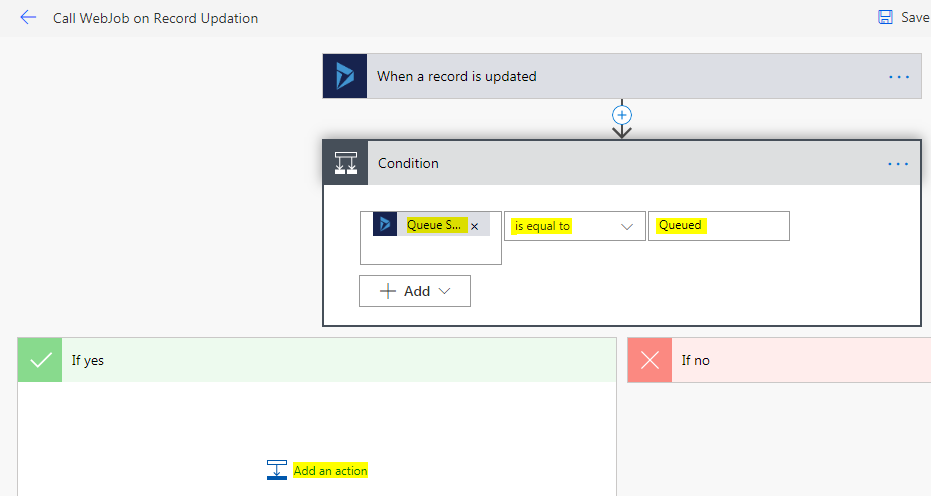
1. On the next screen, under **Organization Name**, select your organization from the drop down, under **Entity Name**, select **Dynamics Source Control** from the drop down and click **New Step**:



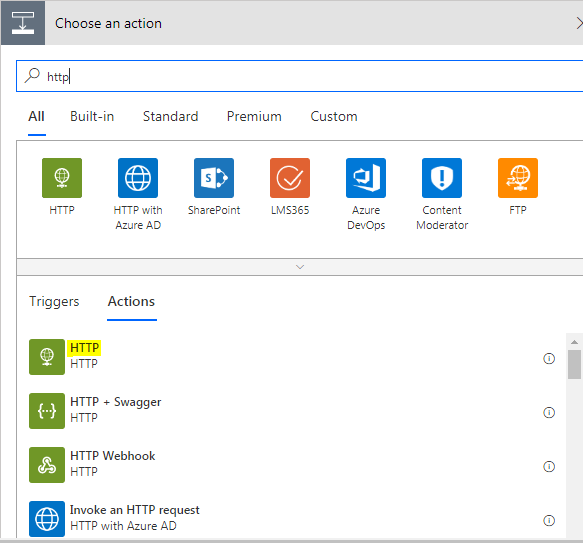
1. On the next screen, under **Actions,** select **Condition:**



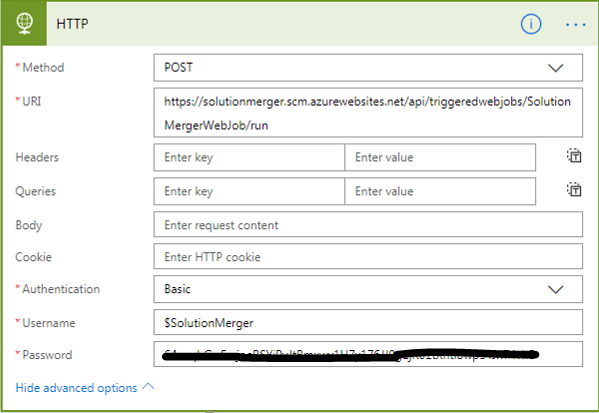
1. On the next screen, specify condition **Queue Status is equal to Queued** as given below and in **If Yes** step, click on **Add an action**:



1. On the next, screen, under **Actions,** search http and select **HTTP** Action:



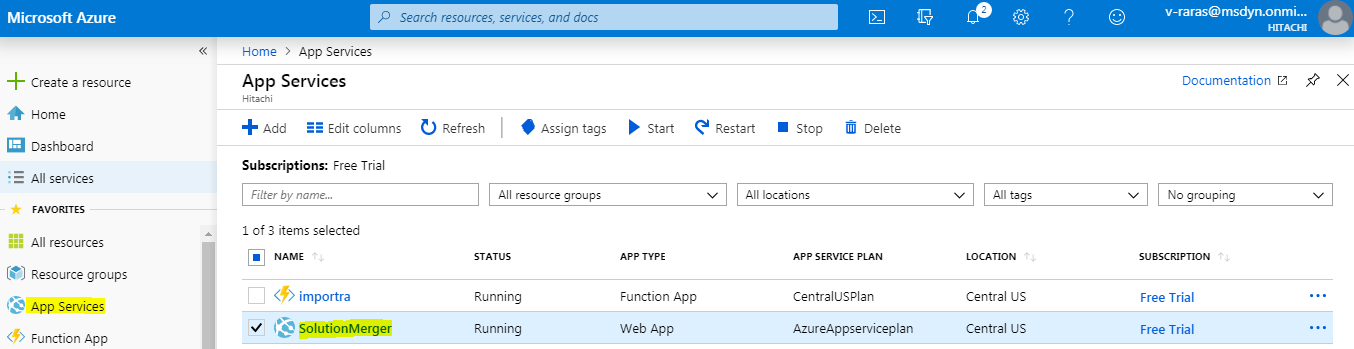
1. Choose **Method** as **POST**, under **Uri**, **Azure WebJob WEB HOOK** needs to be provided. You don’t need to set **Headers, Queries** and **Body**. Under **Advanced Options**, Choose **Authentication** as **Basic** and the **Username** and **Password** also need to be provided from **Azure WebJob** only. (Steps to get Azure WebJob WEB HOOK\*, Username\* and Password\* are provided at the last)



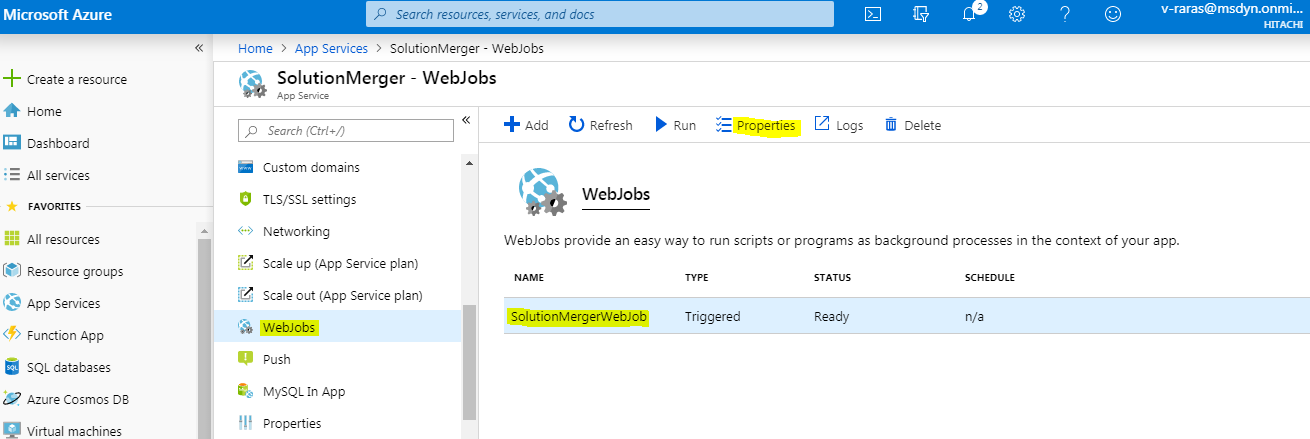
1. Next, click on **Save.** This will create the flow which will call our **Azure Webjob** when condition is satisfied.

**Steps To get Azure WebJob Uri, Username and Password:**

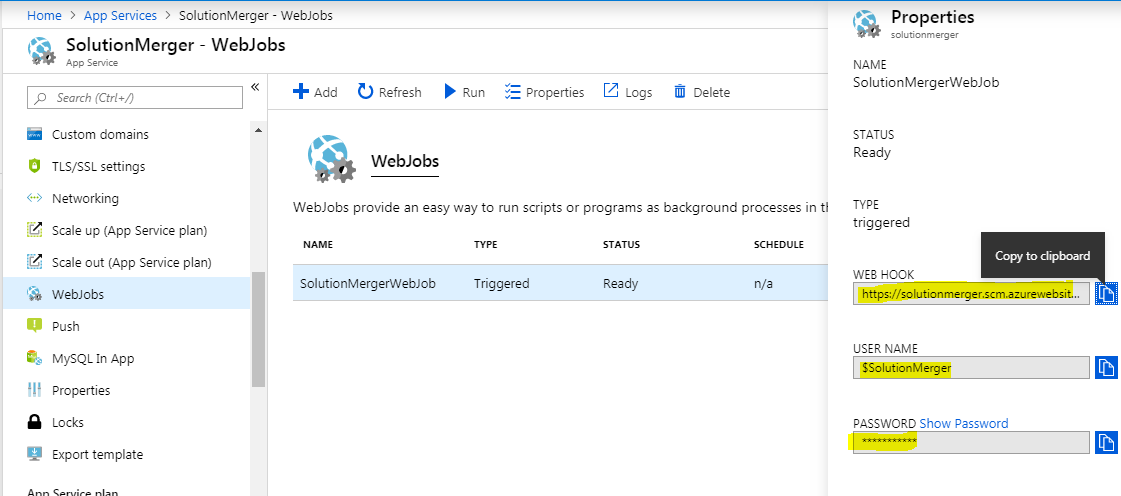
1. Login to Azure portal: [https://portal.azure.com](https://portal.azure.com/)
2. Click on your **Webjob** under **App Services** tab:



1. On the next screen, scroll down the middle pane, click on **Webjobs.** Now select the WebJob and click on **Properties:**



1. A new window will open at right, from where you can copy **WEB HOOK, Username and Password:**



## GitHub and CRM Configuration

* Login to GitHub account
* Create folder for JS, Images, HTML, Solution in Git Hub Repository.

**Note: If folders are already there, please ignore above step**.

* Once created, Login to CRM. Navigate to Settings->Configuration Settings-> Please follow the below table to update in respective configuration setting records.

|  |  |  |
| --- | --- | --- |
| **Configuration Key** | **Value (Example)** | **Description** |
| **BranchName** | master | Branch Name of Repository |
| **CloneRepositoryAlways** | TRUE | Default Value (Please don’t change) |
| **MultilpleSolutionsImport** | MultilpleSolutionsImport.ps1 | Default Value (Please don’t change) |
| **RemoteName** | origin | Remote Name of Repository |
| **RepositoryHtmlDirectory** | source\html\ | GitHub Repository HTML folder path |
| **RepositoryImagesDirectory** | source\images\ | GitHub Repository images folder path |
| **RepositoryJsDirectory** | source\js\ | GitHub Repository js folder path |
| **RepositoryLocalDirectory** | Solution | GitHub Repository CRM solution text file folder path  **Note**: (In solution.txt will contain list of solutions names for Deployed via Azure DevOps and It will be updated automatically) |
| **RepositorySolutionFolder** | Release\ | GitHub Repository Solution folder path |
| **RepositoryUrl** | https://github.com/\*\*\*.git | Repository URL |
| **SleepTimeoutInMillis** | 15000 | Default Value (Please don’t change) |
| **SolutionPackagerPath** | E:\tools\Tools\CoreTools\solutionpackager.exe | Default Value (Please don’t change) |
| **SolutionPackagerRelativePath** | CoreTools\solutionpackager.exe | Default Value (Please don’t change) |
| **SolutionToBeImported** | SolutionToBeImported.ps1 | Default Value (Please don’t change) |

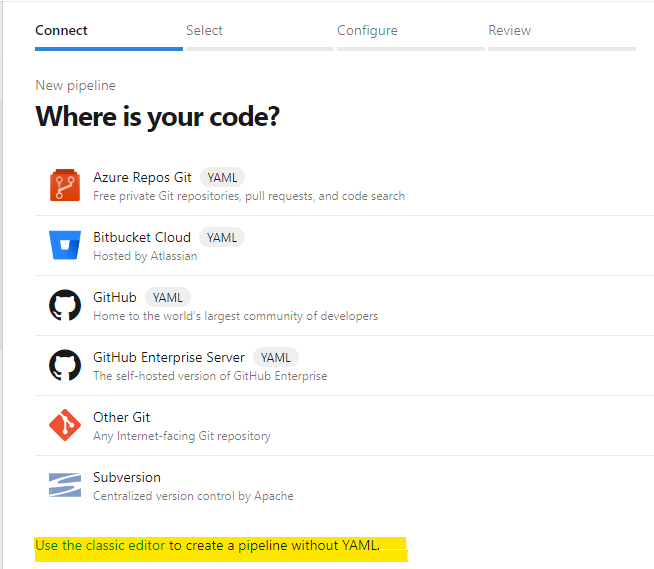
## Azure DevOps

### Prerequisites

* A GitHub Account.
* An Azure DevOps organization.

### Build Pipeline

1. Sign into your Azure DevOps organization and navigate to your project.
2. In your project, navigate to the **Pipelines** page. Then choose the action to create a new pipeline. Click on classic editor as shown below.



1. Walk through the steps of the wizard by first selecting **GitHub** as the location of your source code.
2. You might be redirected to GitHub to sign in. If so, enter your GitHub credentials.
3. When the list of repositories appears, select your desired sample app repository.
4. Azure Pipelines will analyse your repository. Select Save and run, then select Commit directly to the master branch, and then choose Save and run again.
5. A new run is started. Wait for the run to finish.
6. There are many templates available to build common project types. Everything is customizable, and you can even start with an empty pipeline.

#### PowerShell Scripts



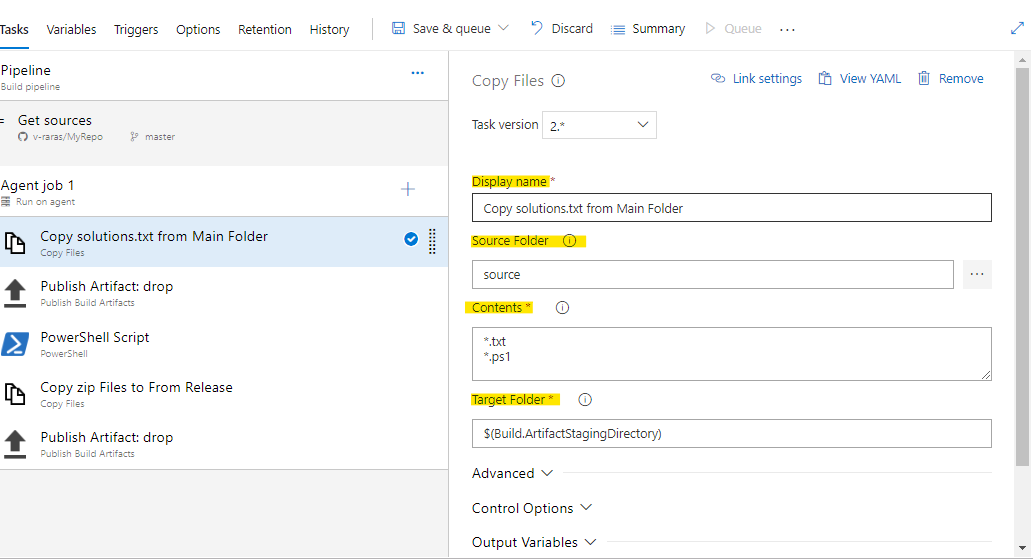
#### Add Task to Build Pipeline:

* 1. Click the Add task button.
  2. Search for **Copy Files** task.

Select Copy Files task to copy files from a source folder to a target folder.

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Source Folder | Folder that contains Solution.txt and PowerShell script |
| Contents | Specify match pattern filters that you want to apply to the list of files to be copied.  **Example**:  \*.txt  \*.ps1 |
| Target Folder | Folder where the files will be copied.  **Example**:  $(Build.ArtifactStagingDirectory) The local path on the agent where any artifacts are copied to before being pushed to their destination. |

Please refer below image.

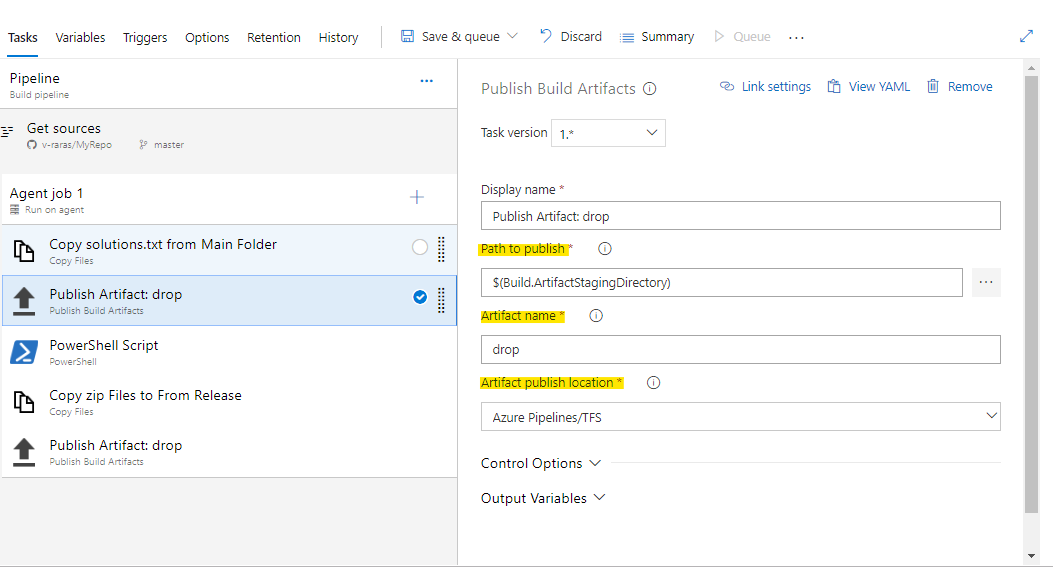


* 1. Search for **publish build artifacts** task.

Select **publish build artifacts** to build outputs to a staging directory and Publish staged artifacts.

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Path to publish | Path to the folder or file you want to publish.  **Example**:  $(Build.ArtifactStagingDirectory) The local path on the agent where any artifacts are stored. |
| Artifact name | Specify the name of the artifact that you want to create.  **Example**:  drop |
| Artifact publish location | Choose Azure Pipelines/TFS |

Please refer below image.

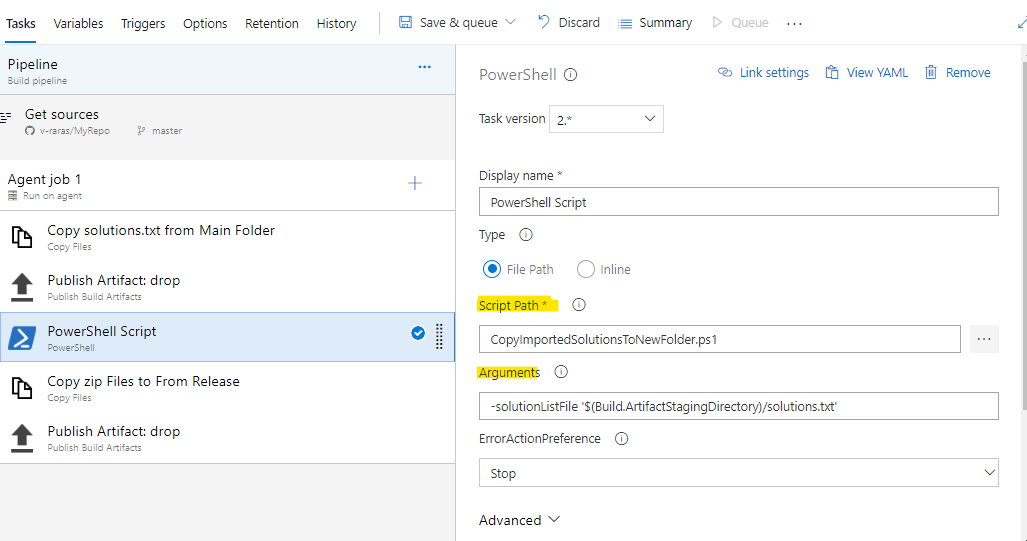


* 1. Search for **PowerShell** task.

Select **PowerShell** task to run a PowerShell script

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Type | Set as an inline script |
| Script Path | Contents of the script, Choose PowerShell script CopyImportedSolutionsToNewFolder.ps1 |
| Arguments | Arguments passed to the PowerShell script.  **Example:**  -solutionListFile'$(Build.ArtifactStagingDirectory)/solutions.txt' |

Please refer below image.

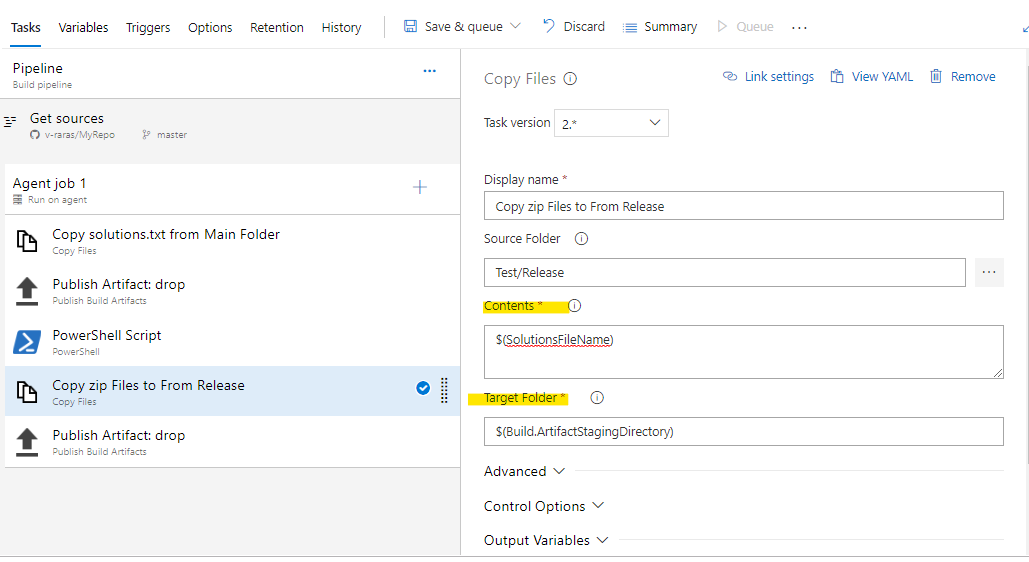


* 1. Search for **Copy Files** task.

Select Copy Files task to copy files from a source folder to a target folder.

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Source Folder | Folder that contains CRM Solutions to be deployed |
| Contents | Specify match pattern filters that you want to apply to the list of files to be copied.  **Example**:  $(SolutionsFileName) |
| Target Folder | Folder where the files will be copied.  **Example**:  $(Build.ArtifactStagingDirectory) The local path on the agent where any artifacts are copied to before being pushed to their destination. |

Please refer below image.

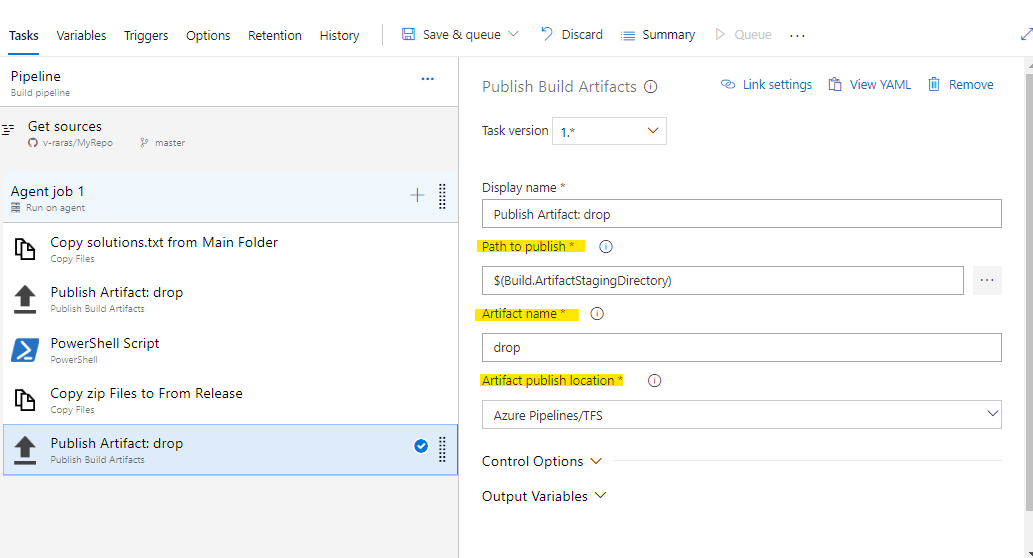


* 1. Search for **publish build artifacts** task.

Select **publish build artifacts** to build outputs to a staging directory and Publish staged artifacts.

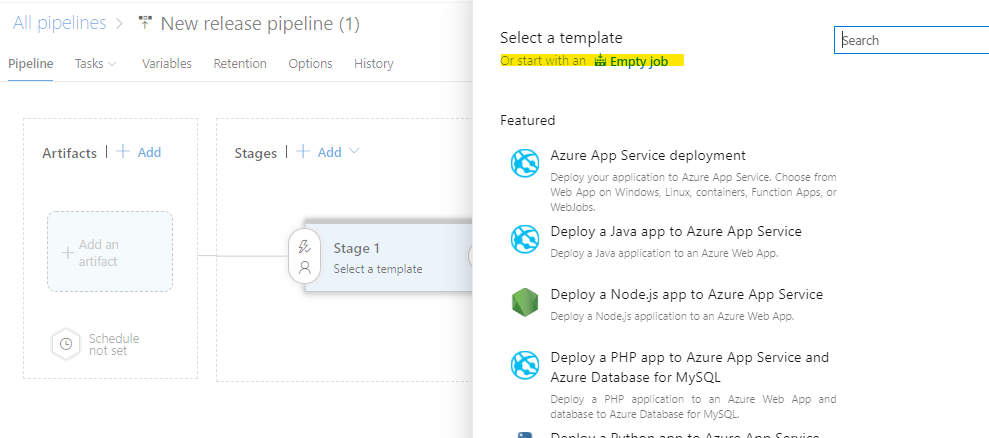
|  |  |
| --- | --- |
| **Argument** | **Description** |
| Path to publish | Path to the folder or file you want to publish.  **Example**:  $(Build.ArtifactStagingDirectory) The local path on the agent where any artifacts are stored. |
| Artifact name | Specify the name of the artifact that you want to create.  **Example**:  drop |
| Artifact publish location | Choose Azure Pipelines/TFS |

Please refer below image.

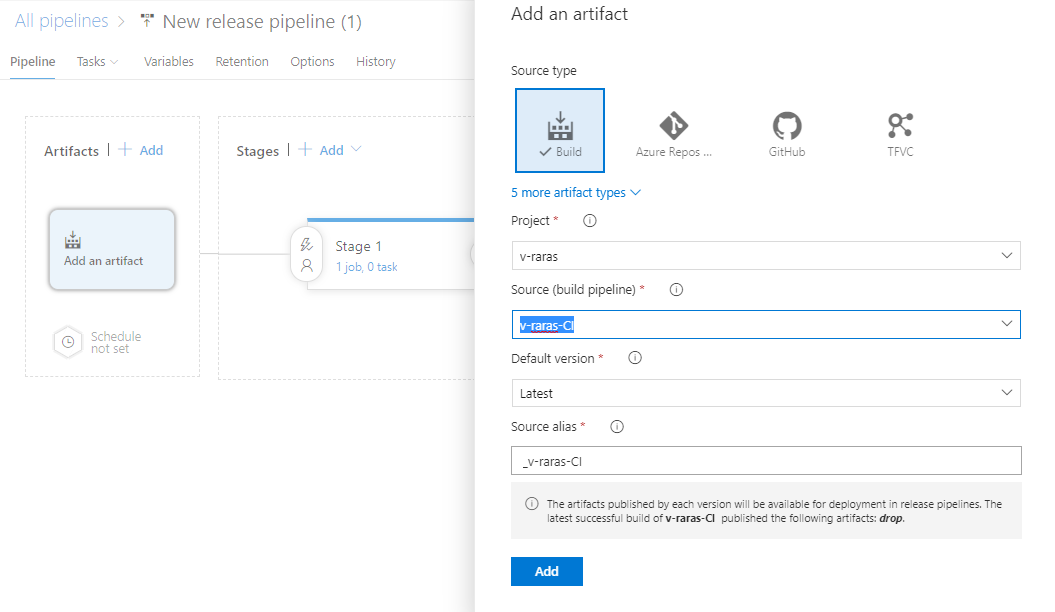


### Release Pipeline:

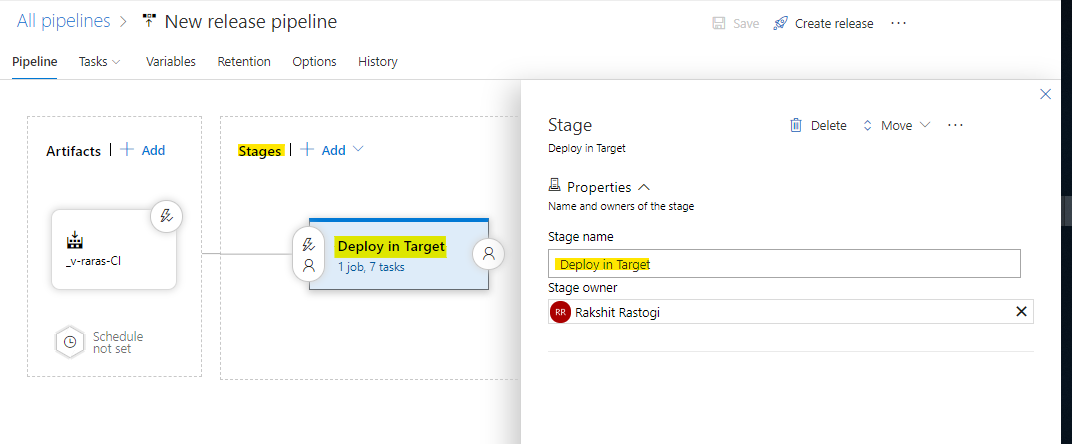
1. In your project, navigate to the Releases page. Then choose the action to create a new pipeline. In Select a Template as Empty Job.



1. Click on the Artifacts within the release pipeline. Choose the Build Pipeline that you have created.

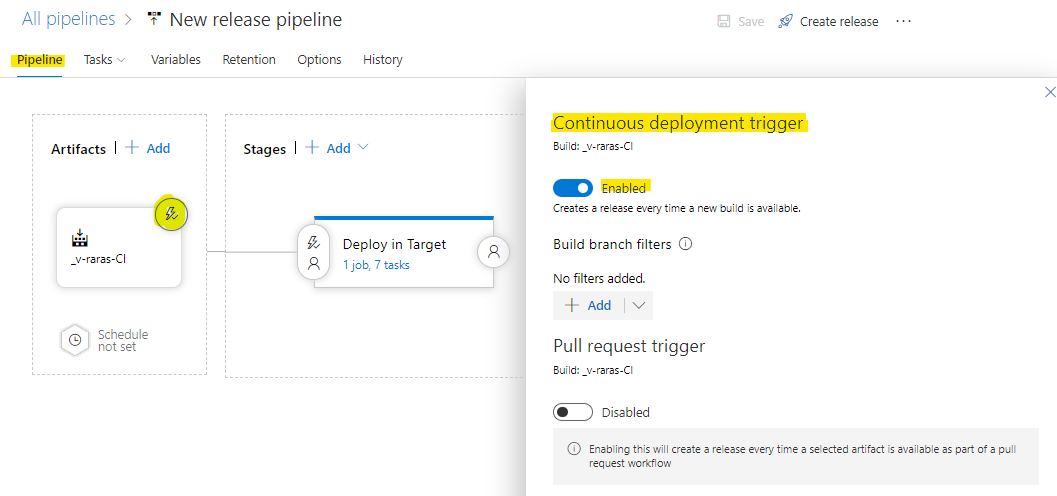


1. Click on the stage within the release pipeline and rename the Stage Name.



1. To set continuous deployment trigger. Click on the trigger as shown in below image.

Set that continuous deployment to enabled.

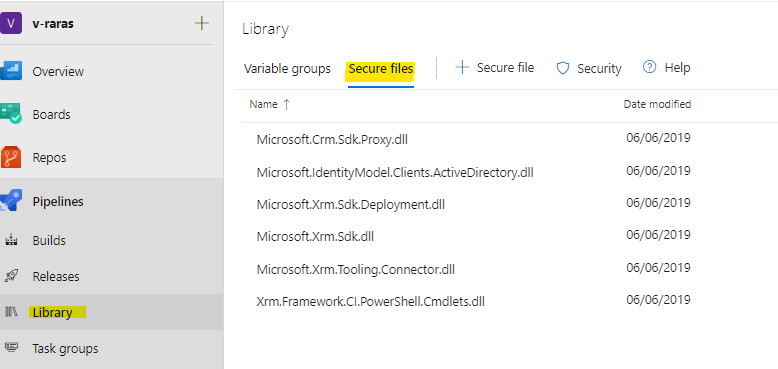


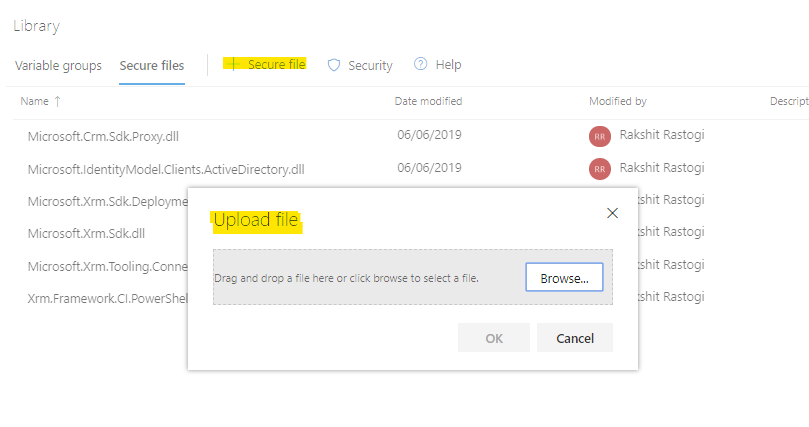
#### Add Secure File

1. Follow the steps to add listed .dll in Library.

* Microsoft.Xrm.Sdk.dll
* Microsoft.Crm.Sdk.Proxy.dll
* Microsoft.IdentityModel.Clients.ActiveDirectory.dll
* Microsoft.Xrm.Sdk.Deployment.dll
* Microsoft.Xrm.Tooling.Connector.dll
* Xrm.Framework.CI.PowerShell.Cmdlets.dll

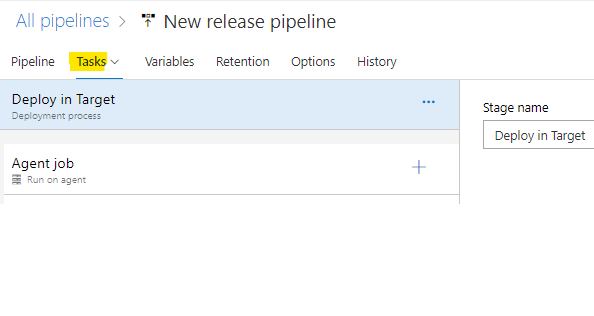


1. Navigate to the Library tab in Azure Pipelines and Select the Secure files tab at the top.
2. Click on the + Secure file and upload .dll and click ok.



#### Add Task to Release Pipeline:

1. In Release Pipeline, Click on Task as shown as below.

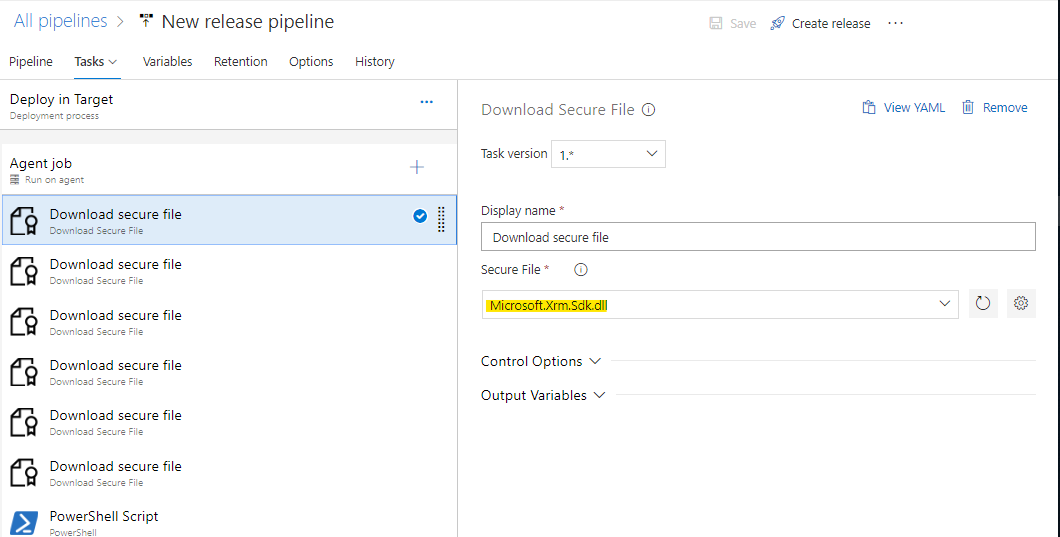


1. Search for **Download Secure File** task.

Select Download Secure File to download a secure file to a temporary location on the release agent.

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Secure File | Select the secure file to download to a temporary location.  **Example**: Microsoft.Xrm.Sdk.dll  **Note**: The file will be deleted after the build or release. |

1. Please refer below image.



1. Follow the Step:2 and add below secure files from Library.

* Microsoft.Crm.Sdk.Proxy.dll
* Microsoft.IdentityModel.Clients.ActiveDirectory.dll
* Microsoft.Xrm.Sdk.Deployment.dll
* Microsoft.Xrm.Tooling.Connector.dll
* Xrm.Framework.CI.PowerShell.Cmdlets.dll
  + Search for **PowerShell** task.

1. Select **PowerShell** task to run a PowerShell script

|  |  |
| --- | --- |
| **Argument** | **Description** |
| Type | Set as an inline script |
| Script Path | Contents of the script, Choose PowerShell script from the drop location.  **Example**: $(System.DefaultWorkingDirectory)/\_v-raras-CI/drop/MultilpleSolutionsImport.ps1 |
| Arguments | Arguments passed to the PowerShell script.  **Example:**  -solutionListFile '$(System.DefaultWorkingDirectory)/\_v-raras-CI/drop/  solutions.txt'  -solutionImportPath '$(System.DefaultWorkingDirectory)/\_v-raras-CI/drop'  -solutionContainer '$(System.DefaultWorkingDirectory)/\_v-raras-CI/drop/SolutionsToBeImported'  -crmConnectionString 'AuthType=Office365;Username=\*\*\*\*\*\*;  Password=\*\*\*\*\*\*;Url=\*\*\*\*\* '  -override 1  -publishWorkflows 1  -overwriteUnmanagedCustomizations 1  -skipProductUpdateDependencies 1  -convertToManaged 0  -holdingSolution 0  -AsyncWaitTimeout 0  -logsDirectory ''  -logFilename '' |

Please refer below image.

