**DATA MIGRATION FOR AZUREDEVOPS GUIDE**

**Hello everyone. Welcome to the data migration Guide.**

The 1st step to migrate TFS to Azure is to have the latest TFS update. You can get the latest update through this link.

<https://docs.microsoft.com/en-us/azure/devops/server/release-notes/azuredevops2019u1?view=azure-devops>

Once you have the latest TFS update you must download the Migration tool and the Migration guide (helpful for your reference). The Migrator tool can be downloaded from the link below:

<https://www.microsoft.com/en-us/download/details.aspx?id=54274>

When the download is complete, please open the “Command prompt” and run as administrator. Later navigate to the path of the downloaded file.

**For example**: If you have TFS “2019 update 1.1” please select “DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758” or if you have TFS “2019 update 2019.1RWT “ please select the folder “DataMigrationTool\_AzureDevOps2019.1RTW\_17.153.10129122” and navigate to the particular folder in “Command prompt”.

Once you have navigated to the folder run the below command to validate:

**Migrator validate /collection:(URL)**

**For example: I have TFS 2019 update 1.1 so have selected “DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758” folder.**

**C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool>Migrator validate /collection:http://test-demo-vm:8080/tfs/DefaultCollection/**

**Note**: for all the migration commands a log file will be generated in the path from where you run the commands.

Once you run the above command usually if you have the latest update of TFS you will get an output stating “Validation Passed” if you have the older version and you try to validate you get an error. So, if there occurs an error you have to 1st fix the errors and then re-run the “Validate Command” as shown above.

If the validation is passed, you must save the validation process. For this you must run the below command in the command prompt from the location path you ran the “Validate Command”. This save process helps to save all the process at the organization level.

**Migrator validate /collection:(URL)/SaveProcesses**

**For example, the command looks like:**

**Migrator validate /collection:http://localhost:8080/tfs/DefaultCollection/SaveProcesses**

So once the SaveProcesses is completed you will have to prepare to start the migration. For that you will need “. json” file and “.csv” file. To extract those two files, you must run the below command:

**Migrator prepare /collection:URL /accoutRegion:<regionname> /tenantDomainName:<domainname>**

**For Example: I will select the regionname as South India (MA) and the domainname as ecanarys.com so the command looks like**

**Migrator prepare /collection:http://localhost:8080/tfs/DefaultCollection /accoutRegion:MA /tenantDomainName:ecanarys.com**

When you hit enter you will be prompted to login to the domain credentials and the process of preparing the documents kicks off.

**Note:** You can get the Region code from the link: <https://docs.microsoft.com/en-us/azure/devops/migrate/migration-import?view=azure-devops#supported-azure-regions-for-import>

Once you run the command, “. json and .csv files” will be extracted to the same file path where the migration logs are documented.

After the extraction of .json ad .csv file we will have to take the collection offline that means we must detach the collection from “team foundation server administrator console”. Before detaching you must take a backup of the database.

**Note**: If you are just performing a Dryrun, once the SQL database backup of the fully detached team project collection has fully completed, you can attach the team project collection again to

make it available to your team members and you can proceed with the further steps.

Now you will have to create a .DACPAK ie (Data Tier AppliCation Package) and run the SqlPackge.exe For that we will need to download and install SqlPakage. You can download it form the below link:

<https://docs.microsoft.com/en-us/sql/tools/sqlpackage-download?view=sql-server-ver15>

Once the SqlPackage is downloaded and installed we will have to run the SqlPackage command form the path where the Sqlpackage.exe is present to extract the DACPACK file. Below is the command used to run the SqlPackage:

**SqlPackage.exe /sourceconnectionstring:”Data Source=localhost;Initial**

**Catalog=Tfs\_Foo;Integrated Security=True” /targetFile:C:\DACPAC\**

**Tfs\_Foo.dacpac /action:extract /p:ExtractAllTableData=true**

**/p:IgnoreUserLoginMappings=true**

**/p:IgnorePermissions=true**

**/p:Storage=Memory**

In the above command we will have to make some small changes in the following parameters:

* **DataSource**- ie the name of your localhost instance

Example my localhost instance name is “SQLEXPRESS” so the DataSource parameter should be “localhost\ SQLEXPRESS”.

* **Initial Catalog**- this should be the name of your collection

Example my collection name is “DefaultCollection” so the Initial Catalog should be “Tfs\_ DefaultCollection”.

* **targetFile**- this is the place where you want your DACPAC to be extracted.

So, the final command should be something like this after setting the parameters.

**For Example: SqlPackage.exe /sourceconnectionstring:"Data Source=localhost\SQLEXPRESS;Initial Catalog=Tfs\_DefaultCollection;Integrated Security=True" /targetFile:C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool\Logs\DefaultCollectionTfs\_DataColletion.dacpac /action:extract /p:ExtractAllTableData=true /p:IgnoreUserLoginMappings=true /p:IgnorePermissions=true /p:Storage=Memory**

**Note:** If Migrator warned that your collection was too big, you will not want to generate a DACPAC backup of your SQL database. There is an alternate method that you will need to take which is setting up your own SQL Server in the same Azure datacenter. For more info about that Please check the link: - <https://docs.microsoft.com/en-us/azure/devops/migrate/migration-troubleshooting?view=azure-devops#resolve-size-warnings>

Now you have all the data necessary ie. “.jason file”, “.csv file” and “DACPAC file”. Now, we must create a blob storage container in azure account. You can follow the steps is the below link to create a storage container: -

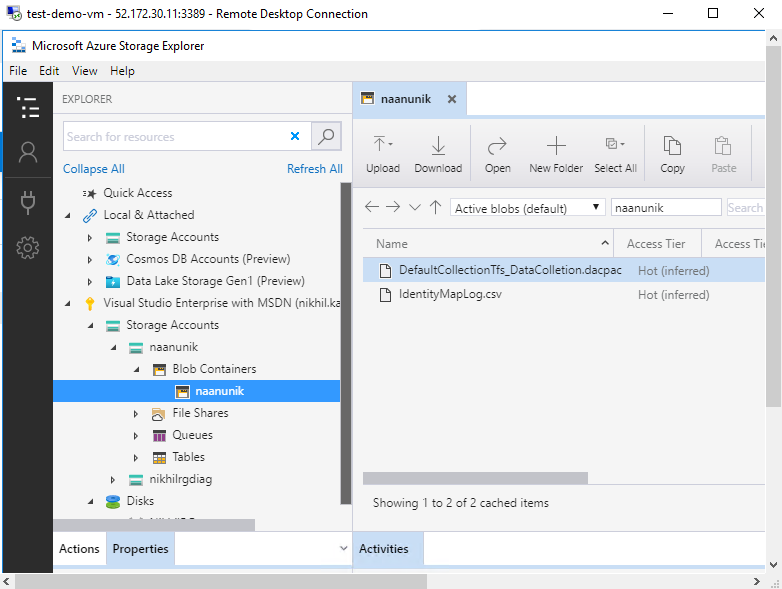
<https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-portal>

After the creation of blob storage container, we will have to upload the .csv file and DACPAC file to the storage container. We can use “Microsoft Azure Storage Explorer” to upload the files to the storage container. You can download the “Microsoft Azure Storage Explorer” for the below link:

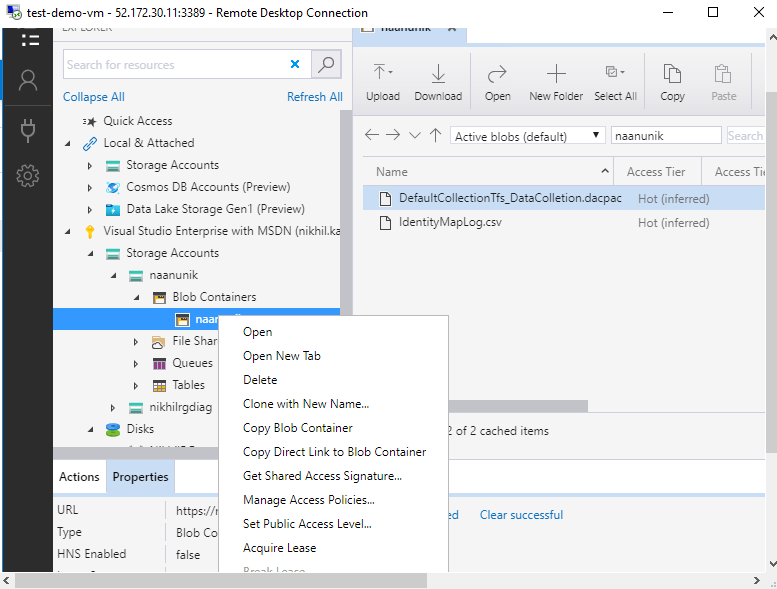
<https://azure.microsoft.com/en-us/features/storage-explorer/>

Once you upload the .csv file and DACPAC file to the storage container we will have to create a SASkey (Shared Access Signature) of the storage container. For that we will have to open the “Microsoft Azure Storage Explorer” >> “navigate to the blob storage name”>> “right click” >> and select “Get Shared Access Signature. You can follow the screenshots attached below:

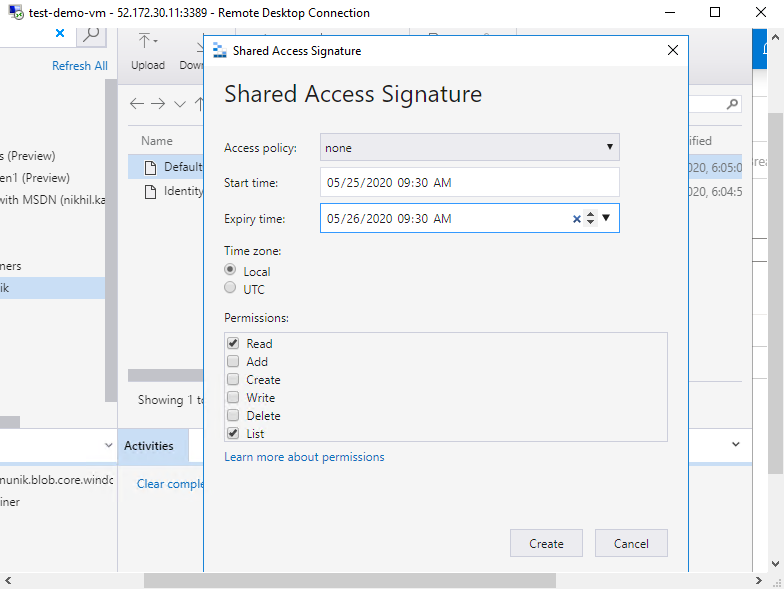
**Step 1:** Navigate to your blob storage where you have uploaded the files in the Azure Storage Explorer. My storage name is “naanunik”.



**Step 2:** After navigating to the blob storage, right click on the blob storage name to view the option “Get Shared Access Signature”



**Step 3**: When you click on “Get Shared Access Signature” another dialogue box will be prompted to set the expiry of the SAS key and to set the permissions. Please select the permission as List and Read. Please refer the screenshot below:



**Step 4:** After setting the expiry and permissions please click on create and you will see a URL. Please copy that URL for reference.

**For example URL looks like**: <https://naanunik.blob.core.windows.net/naanunik?sv=2019-02-02&st=2020-05-25T05%3A54%3A52Z&se=2020-05-31T05%3A54%3A00Z&sr=c&sp=rl&sig=vBlfpZRinD5eRSRdZseFbUbTxtJFyC%2BkbeOTPdtI%2BoQ%3D>

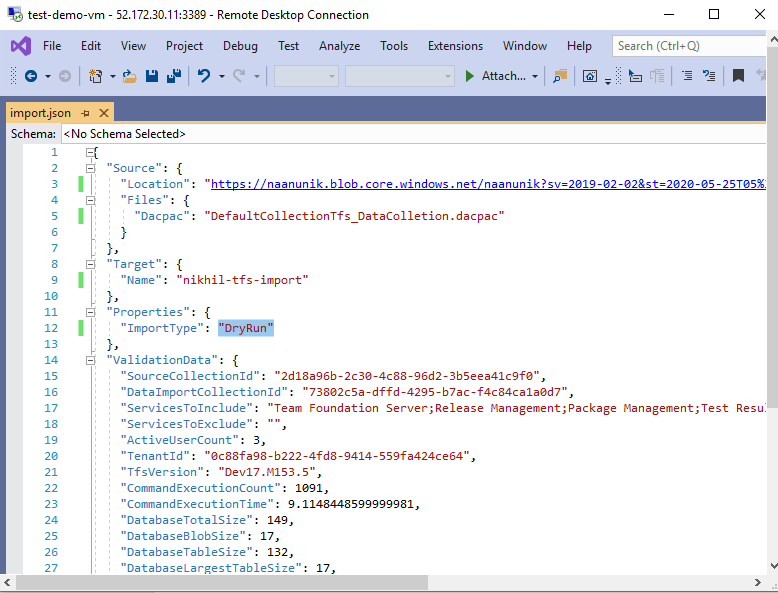
Now we will have to open the .json to make some changes so that we can run an import. So, now we will have to make changes in the below parameter:

* **Location**: Here in location we will have to insert the SAS key that we have generated.
* **Dacpac**: Here you must mention the name of the dacpac file that we had generated earlier.
* **Name**: Here we must specify a name that is not taken earlier by others.
* **ImportType**: Here we will have to input "DryRun" or “ProductionRun”.

So, after the changes are made save the .json file.

**Note:** As a good practice we will have to 1st perform “Dryrun” rather than directly on “production”. DryRun is a kind of testing phase to check everything is working fine.

So, the .json file looks like:



Once the changes are made and saved in the .json file, we will have to run the import command from the data migrator path ie the same path where you had run the “Validation” command. The import command is given below:

**Migrator import /importFile:(path of the .json file)**

**For Example: My .json file is in the path “C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool\Logs\DefaultCollection\20200521\_095019\import.json”**

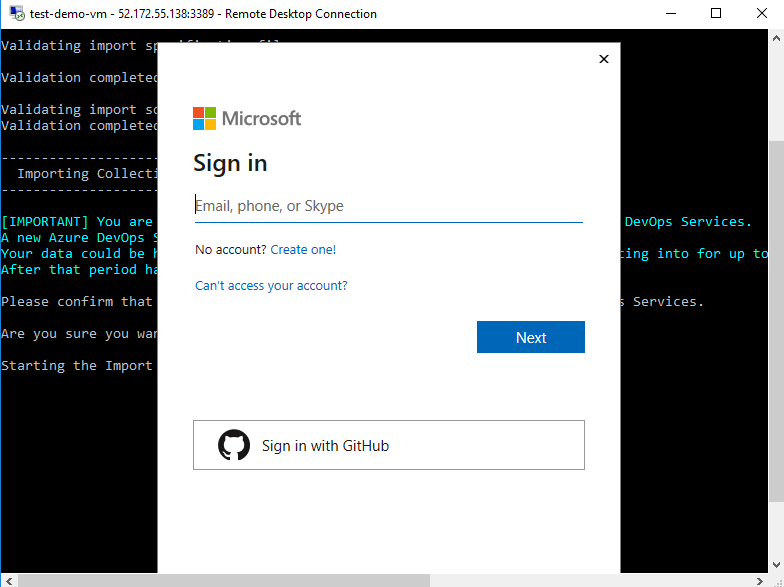
**So, the command looks like:**

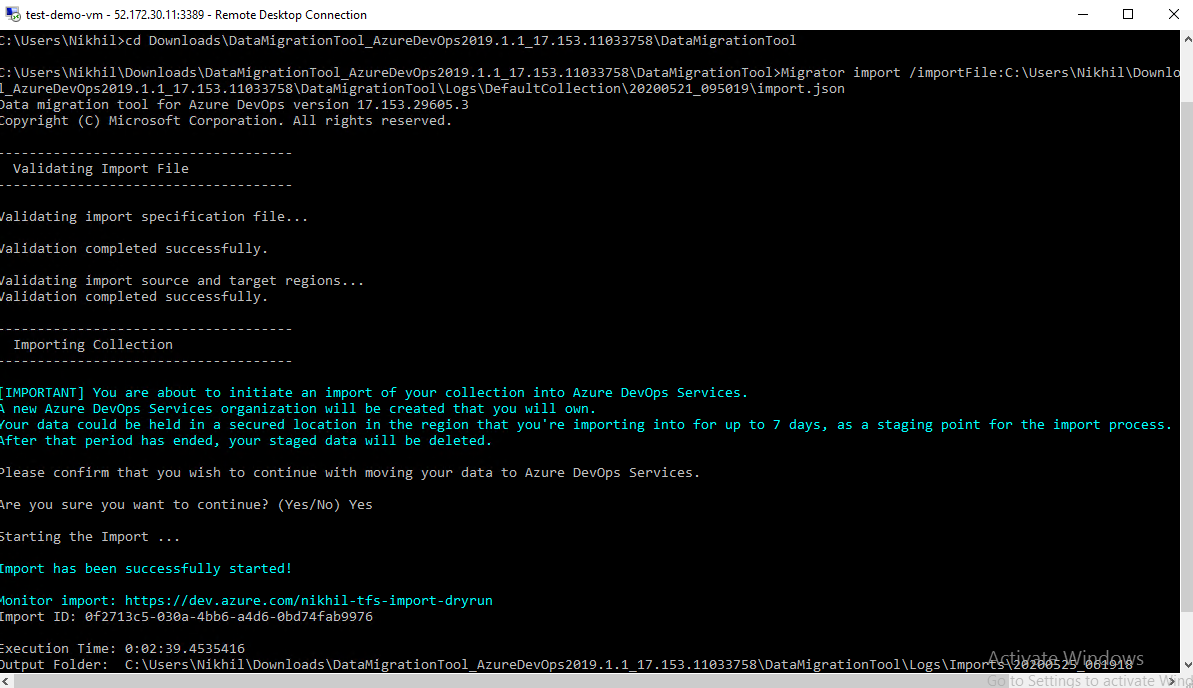
**Migrator import /importFile:C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool\Logs\DefaultCollection\20200521\_095019\import.json**

Once you run the command it will prompt a question stating

“Please confirm that you wish to continue with moving your data to Azure DevOps Services.

Are you sure you want to continue? (Yes/No)”. Please enter “**Yes**” and you will be prompted to enter your credentials. Once you signup you will get an email notification about the import status. Please refer the screenshots below: -





So, once you enter the input as Yes and enter your credentials, the import will start and will take few minutes to complete successfully.

Once the import is successful, it will take several minutes to move our data to the cloud. You will be prompted a mail when the import kicks off and finishes. Once you receive a mail stating “Your Azure DevOps Services dry run import organization is ready” you can hit the URL present in the Monitor Import parameter in the command prompt.

**For example: My import was successful, and the data has been moved to the cloud. Now I can access it with the URL present in the Monitor Import in the above image.**

<https://dev.azure.com/nikhil-tfs-import-dryrun>

Once you hit the URL you can validate if all the data from your TFS has been migrated to the cloud. If everything looks good you can go ahead with the production part.

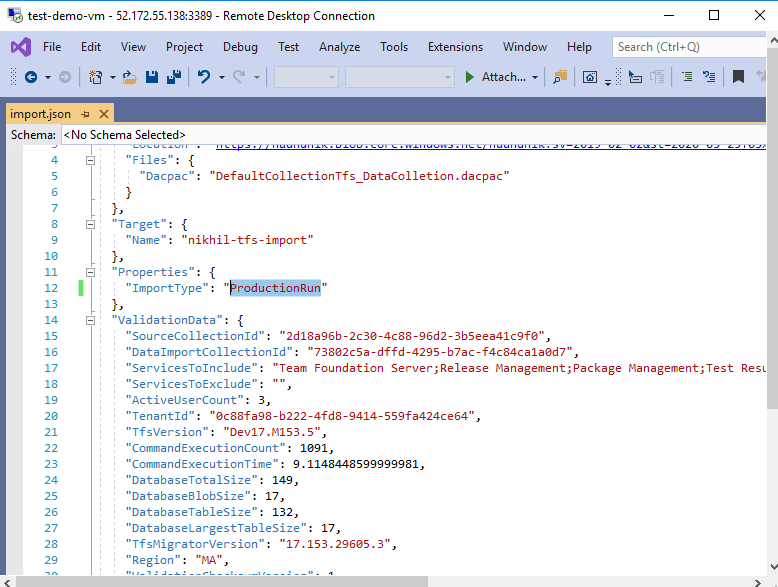
**Note**: Before you can run a second DryRun import or the final production import, you will

need to make sure you delete any previous Azure DevOps Services organizations

that were created in a previous dry run.

Now, for the Production part we will have to make a change in the .json file in the parameter “**ImportType”.**

Before in the .json file we had mentioned “DryRun” for test purpose in the parameter “ImportType”. Now we must change it to “ProductionRun” and keep the other parameters same as we changed for DryRun. Please refer the screenshot of the updated .json file: -



So, once the changes are made save it.

Now, before we run the import command, we will have to delete the “DryRun” organization ie “nikhil-tfs-import-dryrun” from our organization list. If we do not delete the “DryRun” organization, we can not proceed with the “ProductionRun”.

Once the “DryRun” organization is deleted we can proceed with the “import” command as you have done for “DryRun”.

**Migrator import /importFile:(path of the .json file)**

**For Example: My .json file is in the path “C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool\Logs\DefaultCollection\20200521\_095019\import.json”**

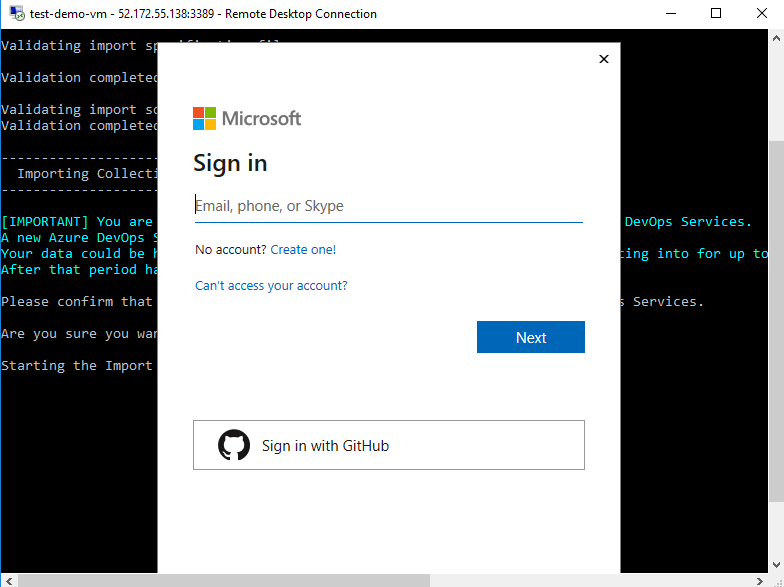
**So, the command looks like:**

**Migrator import /importFile:C:\Users\Nikhil\Downloads\DataMigrationTool\_AzureDevOps2019.1.1\_17.153.11033758\DataMigrationTool\Logs\DefaultCollection\20200521\_095019\import.json**

Once you run the command it will prompt a question stating

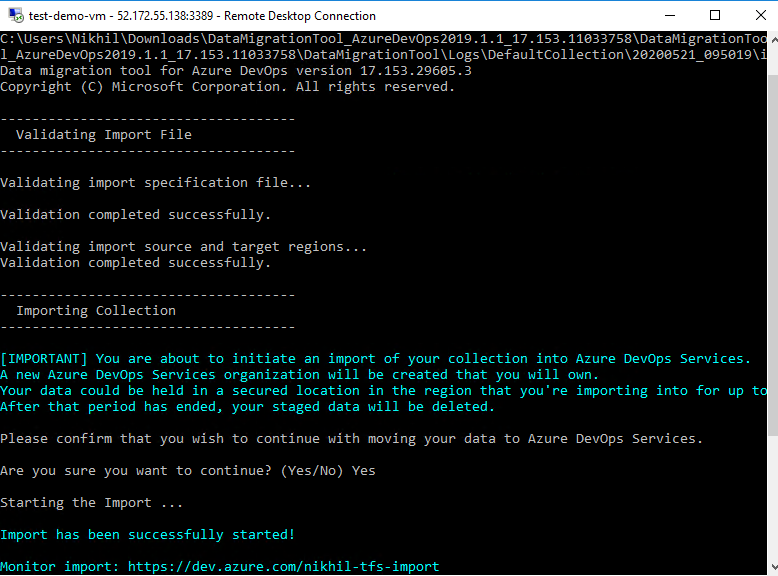
“Please confirm that you wish to continue with moving your data to Azure DevOps Services.

Are you sure you want to continue? (Yes/No)”. Please enter “**Yes**” and you will be prompted to enter your credentials. Once you signup you will get an email notification about the import status. Please refer the screenshots: -

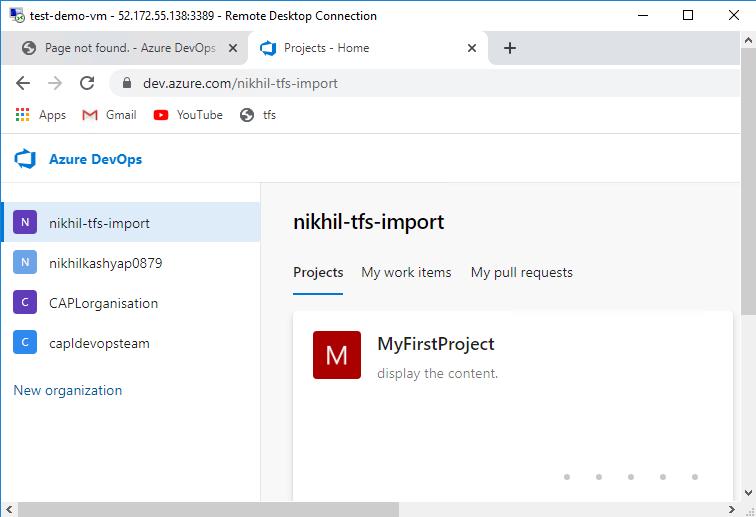


So, once you enter the input as Yes and give the credentials, the import will start and will take few minutes to complete successfully.

Once the import is successful, it will take several minutes to move our data to the cloud. You will be prompted a mail when the import kicks off and finishes. Once you receive a mail stating “Your Azure DevOps Services production import organization is ready” you can hit the URL present in the Monitor Import parameter in the command prompt ie is “<https://dev.azure.com/nikhil-tfs-import>”



These are the steps involved in the migration of TFS to cloud. Once the migration or the import is finished you can hit the link which is present you will be able to see all your data related to organizations or projects. Please have a look at the below screenshot for your reference: -



Now all your data from TFS on premise has been migrated to the Azure cloud. You can easily access from anywhere any system.

**Note: Important Link of Migration for new learners:**

[**https://www.youtube.com/watch?v=K1ynelpy\_OQ&feature=youtu.be**](https://www.youtube.com/watch?v=K1ynelpy_OQ&feature=youtu.be)