

DevOps Workshop

Objective

This workshop is designed to train the participants on the Azure DevOps platform. This workshop will help the participants to understand the Azure DevOps platform by creating the Azure Repos and Build and Release pipelines.

Participants profile

Developers with experience on Visual Studio and Git. Participants must have knowledge on the .NET/.NET Core platform to implement the steps in the workshop.

Tools and services required:

- ◆ Git
- ◆ .NET Core 3.1
- ◆ Visual Studio 2019 Community/Enterprise
- ◆ Azure DevOps Subscription
- ◆ Azure Subscription
- ◆ Visual Studio Code (Optional)

Case Study:

BST Bank is currently using the on-premise servers for deploying databases and web applications. BST Bank currently using a monolithic application developed in .NET framework. They want to migrate the application and data from on-premise to Azure cloud to benefit the features of the cloud infrastructure. They want to redesign the monolithic application in to cloud based application. As the first step they have created a Web API based application for the back-end and an Angular client for the user interface. They also decided to leverage the Azure DevOps pipelines to enable continuous integration and deployment of the application into Azure services.

Module 01: Creating Azure DevOps Project

- 1) Open the browser and navigate to <https://dev.azure.com>
- 2) Sign up for the Azure DevOps account, if you don't have it already. Click on the 'Sign in to Azure DevOps' to login if you already have an account.
- 3) After you have logged in successfully, you can create your first DevOps project. Specify the name of the DevOps project as 'WS-BST-Workshop' and provide a description. Select project visibility as 'Private' and click Create. The project uses 'Git' as the default version control system and 'Agile' as the work item process type. You can change it in the 'Advanced' section. Click on 'Create project' button.

Create a project to get started

Project name *

 ✓

Description

Visibility



Public

Anyone on the internet can view the project. Certain features like TFVC are not supported.



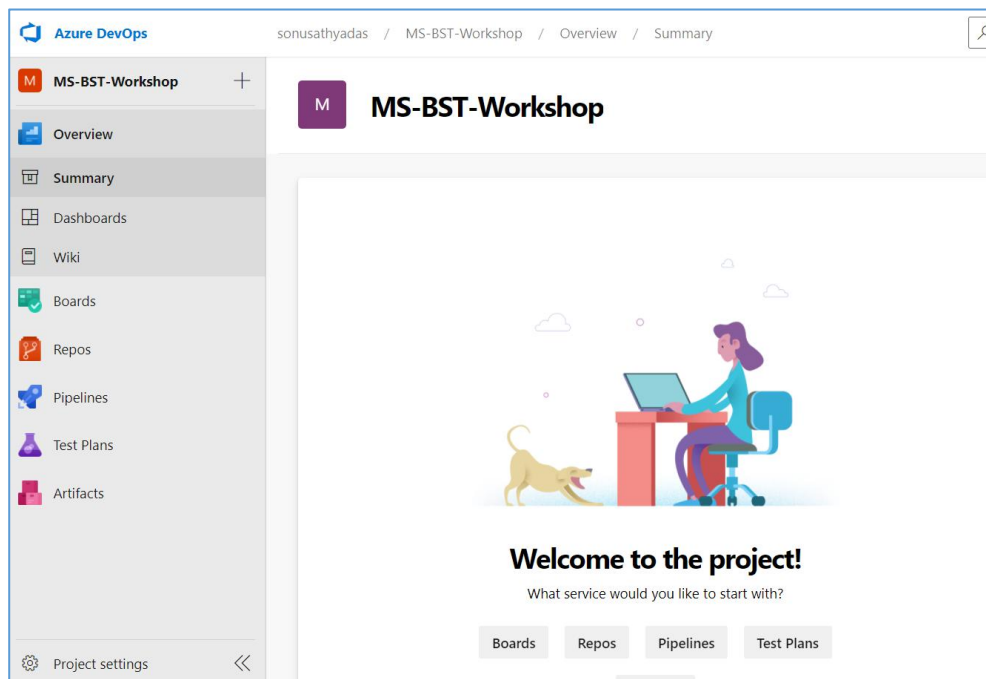
Private

Only people you give access to will be able to view this project.

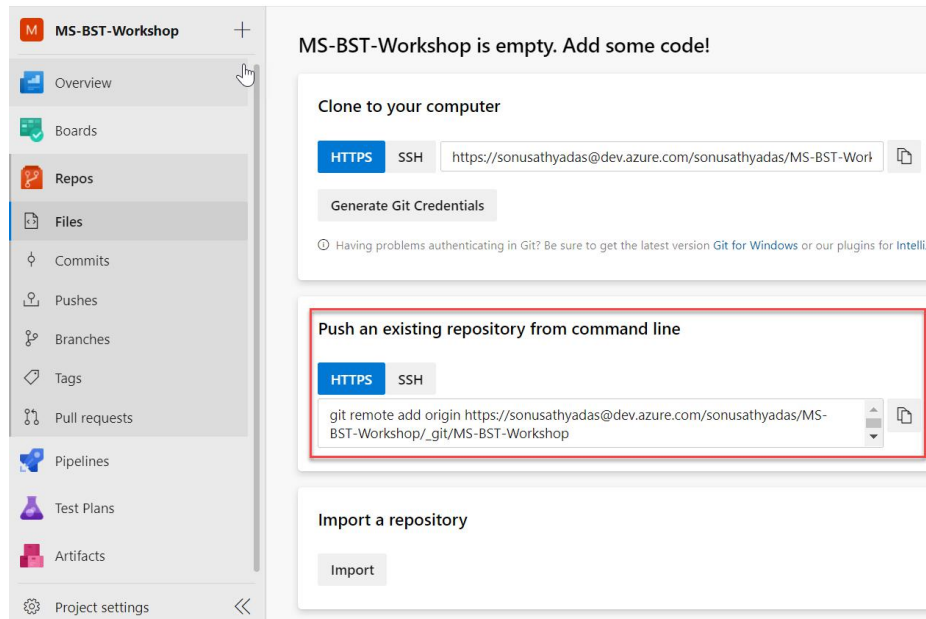
Advanced

+ Create project

- 4) After the project is created, you will be able to see the summary page of the project. You can navigate between Repos, Pipelines and Boards using the navigation menu on the left side.



- 5) Click on the 'Repos' menu item. You will see the options to configure the git project to the Azure DevOps repository.



Module 02: Creating Azure App Service and SQL database

- 1) Open Azure portal by navigation to <https://portal.azure.com> and login with your Azure credentials.
- 2) Click on the Resource Groups icon and create a new Resource Group with the name **"MS-BST-Workshop-Group"**. Select the location whichever you prefer.
- 3) Create a new Azure Database account by selecting 'Create a resource > SQL Database'.
- 4) Select the active subscription and resource group as **'MS-BST-Workshop-Group'**. In the database details section, specify the database name as **'BankDb'**.
- 5) For the server, click on the **'Create new'** link and create a new logical server. Specify the server name as **'xxx-workshop-sql'**. Replace **'xxx'** with your id/name. Specify the username as **'workshopuser'** and password as **'Password@123'**. Choose the same location of the resource group and click **'OK'**.
- 6) Select **'No'** for **'Want to use SQL elastic pool?'** and **'Compute and storage'** as **'General Purpose, Gen 5, 2VCores, 32GB'**
- 7) Select **'Backup storage redundancy'** value as **'Locally-redundant backup storage'** and click on **'Next:Networking'**.
- 8) In the networking page, select **'Public endpoint'** for **'Connectivity method'**. Leave the other values as default and click on **'Review and Create'**. After the validation click on **'Create'** button.
- 9) After the database is created open the SQL database instance and click on the **'Set server firewall'** button from the overview page.
- 10) In the firewall configuration page add a new rule with the name **'Allow All'**. Specify the start IP as **0.0.0.0** and end IP as **255.255.255.255**.
- 11) Click on the **'Save'** button to save the firewall settings.
- 12) No, Navigate to the Home page of the Portal and create a new Azure App Service Web App by selecting **'Create a resource > Web App'**.
- 13) Select the Azure subscription name and the resource group. Select the same resource group which you have created above.
- 14) Specify the name as **'xxx-workshop-bst-web'**. Replace **'xxx'** with your id/name. Select publish method as **'Code'** and runtime stack as **'.NET Core 3.1'**. Select **'Windows'** operating system and region as the same location of resource group.

- 15) Create a new App service plan by clicking on '**Create new**' link. Specify the name as '**workshop-app-plan**'.
- 16) For '**Sku and size**' leave the default '**Standard S1**'. Click '**Next: Monitoring**' button.
- 17) In the monitoring page ensure the '**Application insights**' is enabled and Click '**Review and Create**'. After the validation click on the '**Create**' button.

Module 03: Setting up the application in Visual Studio

- 1) Navigate to the <https://github.com/sonu-trainingmaterials/devops-workshop> url and download the repository to your local machine and extract it.
- 2) Open the '**BSTBankService**' project from the '**Resource-01**' directory. Project will open in Visual Studio.
- 3) You need to create a database with the name '**BankDb**' in your local SQL Server instance. Open the '**appsettings.json**' file and update the value for '**DefaultConnection**'. DefaultConnection will be the connection string to your '**BankDb**' database.
- 4) Press F5 to run the application. It will open the Swagger UI in the browser page. You can test the APIs by creating a new user (customer) and creating an account for the user. Hint: For creating a customer you can use the Swagger UI. Click on the '**Try it out**' button of the '**POST /api/Auth/register**' API and specify a user detail and click '**Execute**'.

Sample data: {

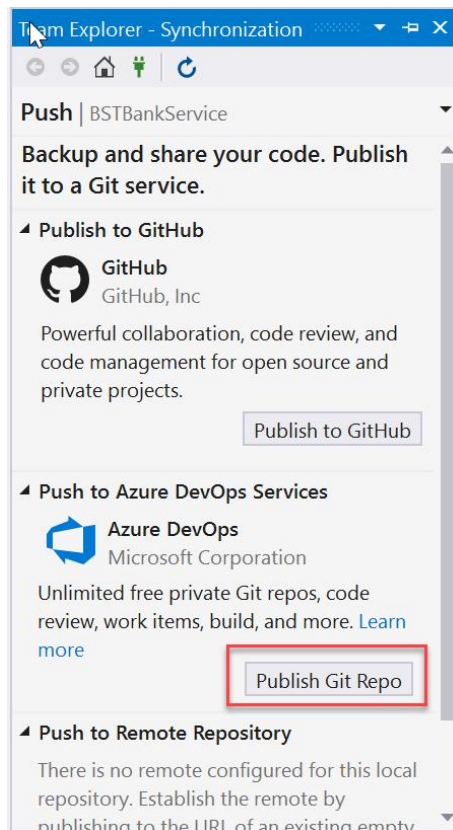
```

    "email": "workshopuser@gmail.com",
    "password": "Password@123",
    "firstName": "Test",
    "lastName": "User",
    "pictureUrl": "",
    "signImageUrl": "",
    "address": "My Address",
    "city": "My City",
    "state": "My State",
    "country": "My Country",
    "pincode": "123456",
    "aadharNo": "12345789012",
    "panNo": "ABCDE1234F"
  }

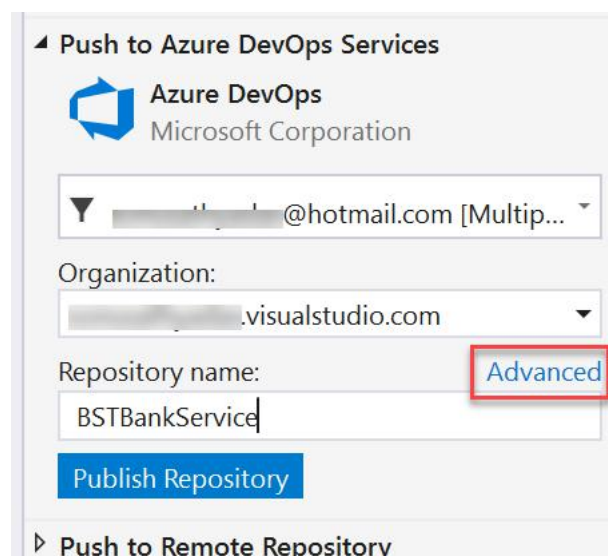
```

After the user is created successfully, it returns a status message and user object. User object contains an **identityId** field. You need to note this id to a text file. This is required whenever you perform any customer related operations later.

- 5) You can now enable version control for the project and push the code to the Azure DevOps project Repos. To do so, click on the '**Add to Source Control**' button on the bottom right corner of the Visual Studio and select '**Git**'.
- 6) Now, in the Team Explorer window, you will see the options to push the code to a remote repository. Click on the '**Publish Git Repo**' button from the '**Push to Azure DevOps Services**' section.



- 7) Select the DevOps account from the Accounts dropdown list. If account is not listing there you can select the '**Add account**' button to add the DevOps account to Visual Studio. Select the organization name and click on the '**Advanced**' button.



- 8) Select the '**MS-BST-Workshop**' project from the Project dropdown list. Specify the repository name as '**MS-BST-Workshop**' (same as DevOps project name) click on the '**Publish Repository**' button.

Push to Azure DevOps Services

Azure DevOps
Microsoft Corporation

[Multip...]

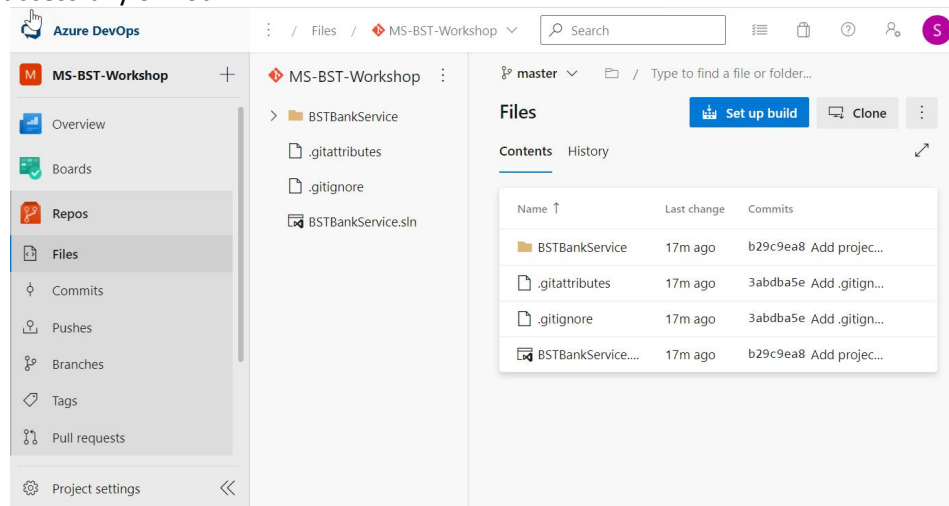
Organization:

Project:
 [Simple](#)

Repository name:

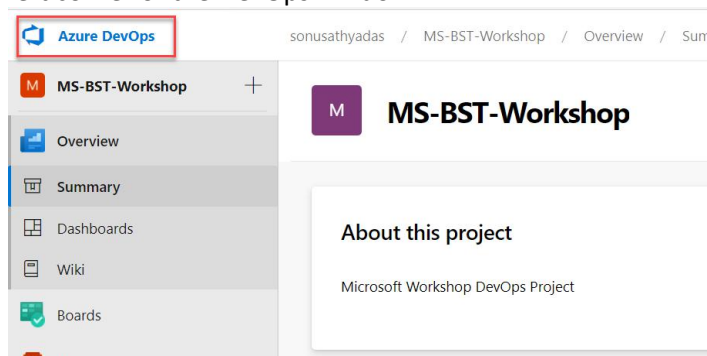
[Publish Repository](#)

- 9) Once the code is published to the Azure repository, you can go back to the Azure DevOps project and select the '**Repos**' option to verify whether the files are uploaded successfully or not.



Module 04: Add users to the Project Team in Azure DevOps

- 1) Navigate to the Azure DevOps project and click on the title 'Azure DevOps' on the top left corner of the DevOps window.



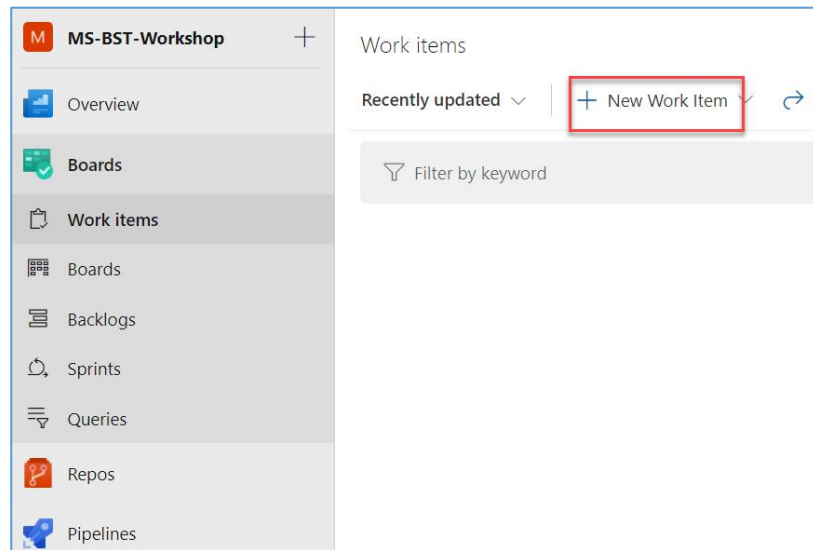
- 2) In the DevOps home page, click on the '**Organization Settings**' button from the bottom left corner.
- 3) In the '**Organization Settings**' page, select '**Users**'. You will see the existing list of users on the right. You can add a new user by clicking on the '**Add Users**' button on the right.
- 4) Type the user email and select Access level as '**Basic**'. Add the user to the '**WS-BST-Workshop**' project and select the '**Azure DevOps Groups**' value as '**Contributer**'. Click on 'Add' button to create the user.

You can add more users if you want.

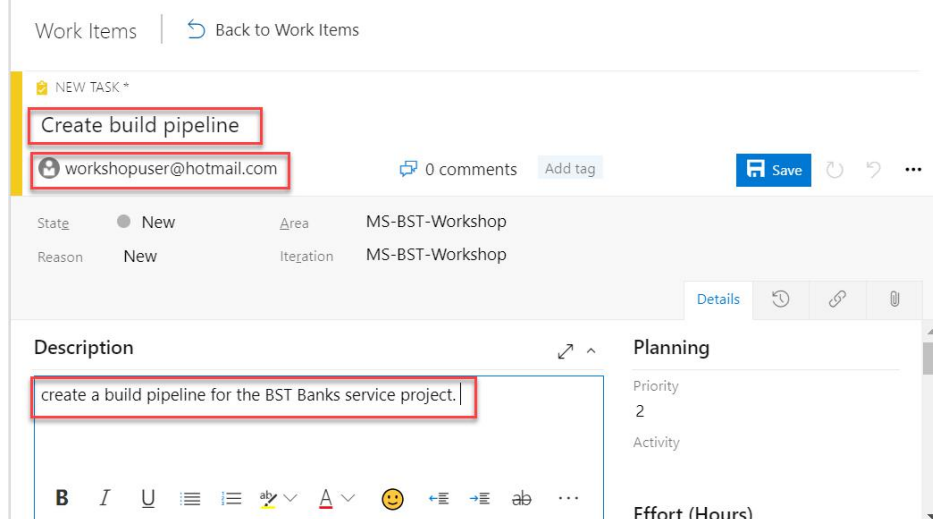
- 5) Open the **WS-BST-Workshop** DevOps project. Navigate to the '**Project Settings**'.
- 6) Click on the '**Teams**' from the left side and click on the default project team name (MS-BST-Workshop Team)

- 7) You will be able to see the existing team members there. You can add new members to the team by clicking on the '**Add**' button on the right side.
- 8) Search for the user you have created in the above step by email and add to the team.

- 9) Click on the '**Boards**' from the left side. Click on the '**New Work Item**' button and select '**Tasks**' from the list.



- 10) Specify the Title and assign the task to the user. To select a user type the name/email of the user in text box. Provide a description and click on '**Save**'.



Module 05: Create a build pipeline

- 1) Click on the Pipelines on left side and you will see an empty list. Click on the '**Create pipeline**' button to create a new build Pipeline.
- 2) In the Connect tab, click '**Use classic editor**' link from the bottom.

New pipeline

Where is your code?

- Azure Repos Git** YAML
Free private Git repositories, pull requests, and code search
- Bitbucket Cloud** YAML
Hosted by Atlassian
- GitHub** YAML
Home to the world's largest community of developers
- GitHub Enterprise Server** YAML
The self-hosted version of GitHub Enterprise
- Other Git**
Any generic Git repository
- Subversion**
Centralized version control by Apache

[Use the classic editor](#) to create a pipeline without YAML.

- 3) Select '**Azure Repos Git**' as the source and choose '**MS-BST-Workshop**' repository from the repositories list. Select '**Master**' branch and click '**Continue**'.

Select a source

- Azure Repos Git**
- GitHub**
- GitHub Enterprise Server**
- Subversion**
- Bitbucket Cloud**
- Other Git**

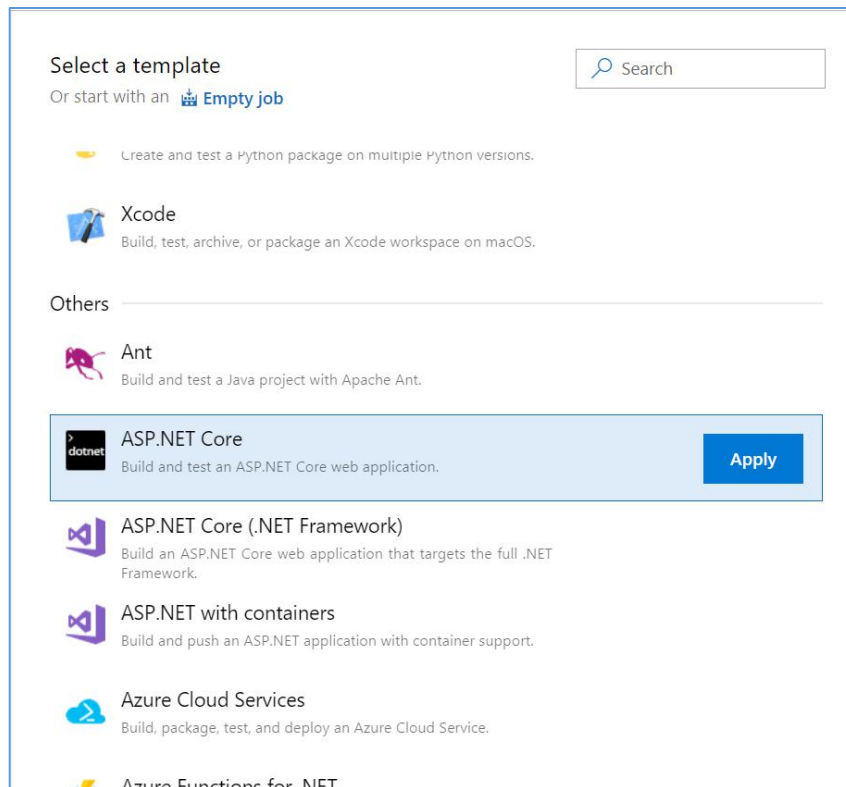
Team project

Repository

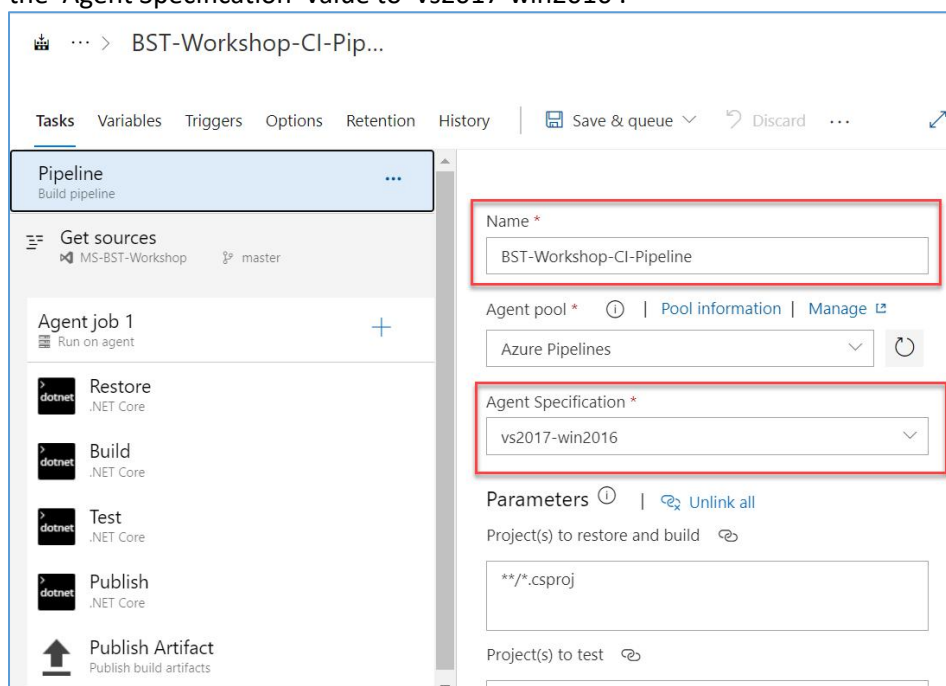
Default branch for manual and scheduled builds

Continue

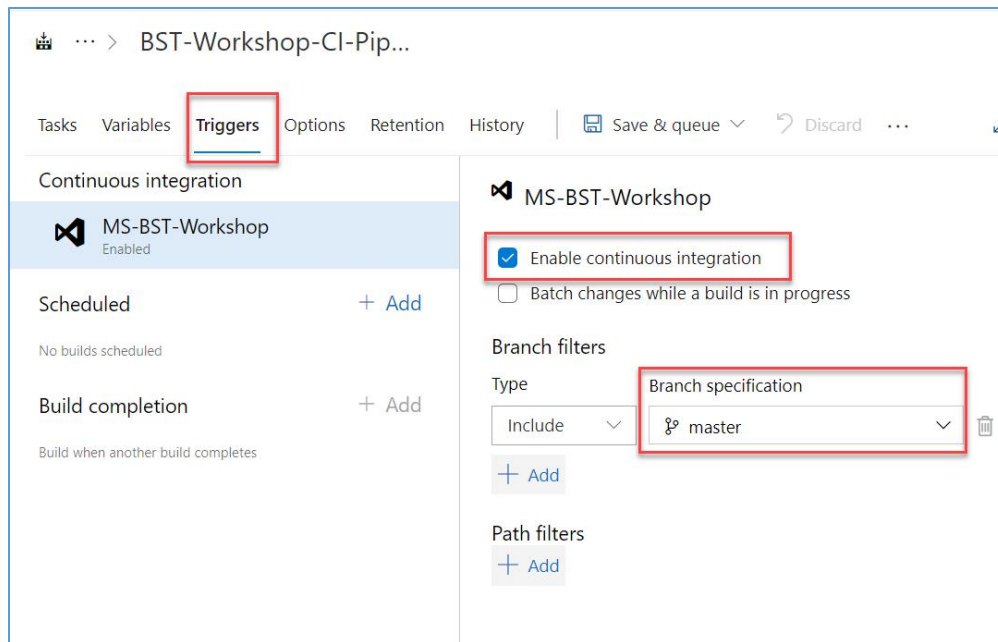
- 4) Select '**ASP.NET Core**' from the build template section.



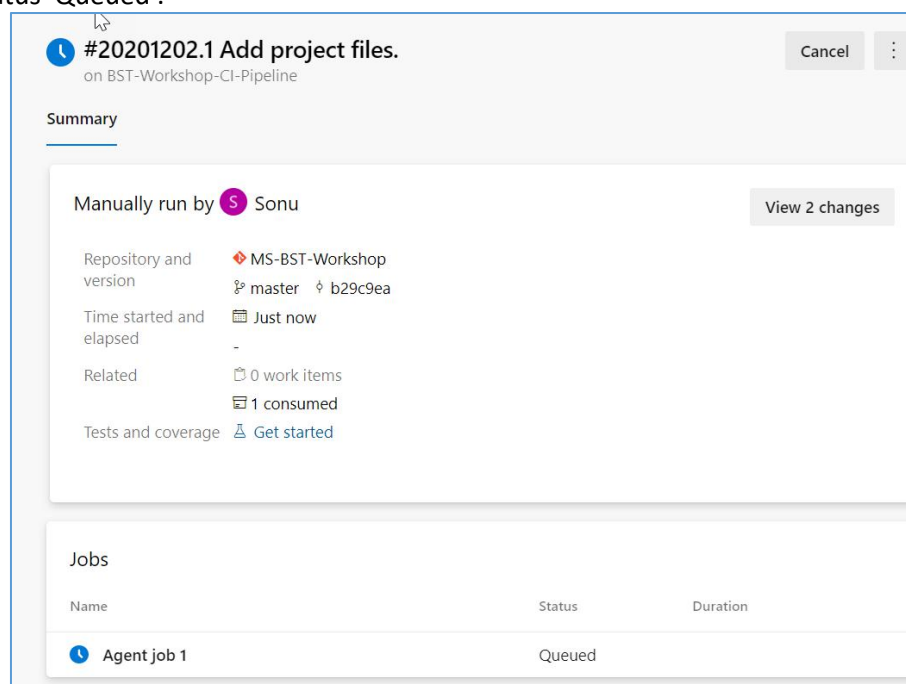
- 5) Change the Pipeline name to '**BST-Workshop-CI-Pipeline**' on the name textbox. Change the 'Agent Specification' value to '**vs2017-win2016**'.



- 6) Select the '**Test**' task from the Tasks list on the left side and click the '**Remove**' link button on top right side.
- 7) Click on the '**Triggers**' tab and select the check box for '**Enable continuous integration**'. Ensure the branch name is selected as '**Master**'.



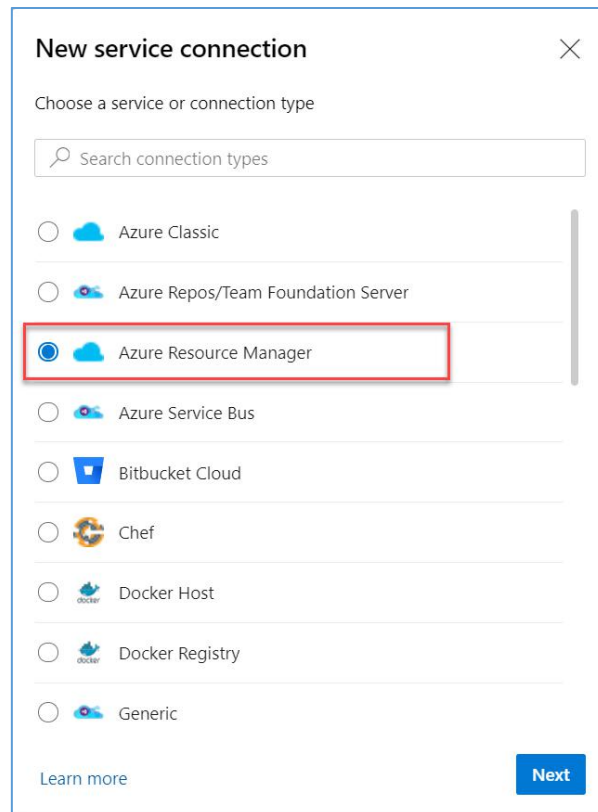
- 8) Click on the '**Save & Queue**' button on top. Enter a comment and click on the '**Save and run**' button on the pop up dialog.
- 9) This will create a new build pipeline and you will see a new Agent job started with status 'Queued'.



- 10) Once the build process is completed, the status will be updated to '**Success**'.

Module 06: Create a service connection for ARM

- 1) Click on the '**Project Settings**' on the bottom left corner. In the 'Project Settings' page scroll down the left side panel and select '**Service Connections**'.
- 2) The service connections list will be empty. You can create a new service connection by clicking in the '**Create service connection**' button.
- 3) Select 'Azure Resource Manager' from the Service connectio types and click **Next**.



New service connection ✕

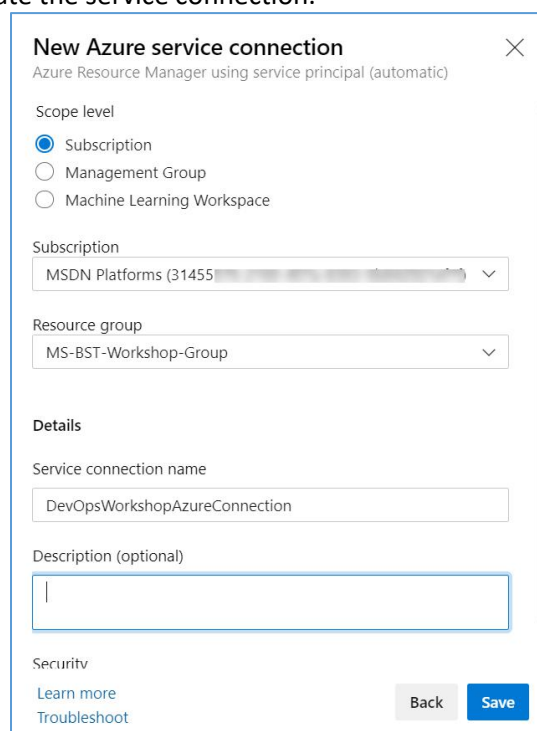
Choose a service or connection type

Search connection types

- ☐ Azure Classic
- ☐ Azure Repos/Team Foundation Server
- ☒ **Azure Resource Manager**
- ☐ Azure Service Bus
- ☐ Bitbucket Cloud
- ☐ Chef
- ☐ Docker Host
- ☐ Docker Registry
- ☐ Generic

[Learn more](#) **Next**

- 4) In the Authentication method page select '**Service principal (automatic)**' and click **Next**.
- 5) In the next page, Select **Subscription** and choose your Azure subscription from the subscriptions dropdown list. If the subscription is not found, you can Add a new subscription by selecting the '**New Azure Subscription**' option. Select the '**MS-BST-Workshop-Group**' in the resource group dropdown. Specify the service connection name as '**DevOpsWorkshopAzureConnection**' and provide a description. Click '**Save**' to create the service connection.



New Azure service connection ✕

Azure Resource Manager using service principal (automatic)

Scope level

- ☒ **Subscription**
- ☐ Management Group
- ☐ Machine Learning Workspace

Subscription

MSDN Platforms (31455...) ▼

Resource group

MS-BST-Workshop-Group ▼

Details

Service connection name

DevOpsWorkshopAzureConnection

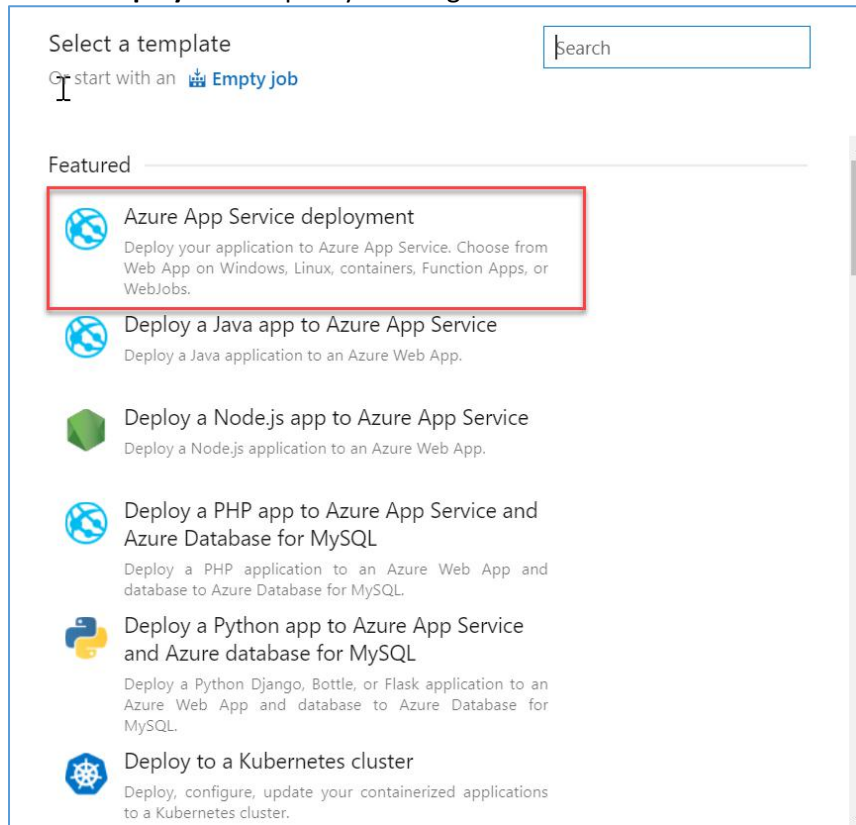
Description (optional)

Security

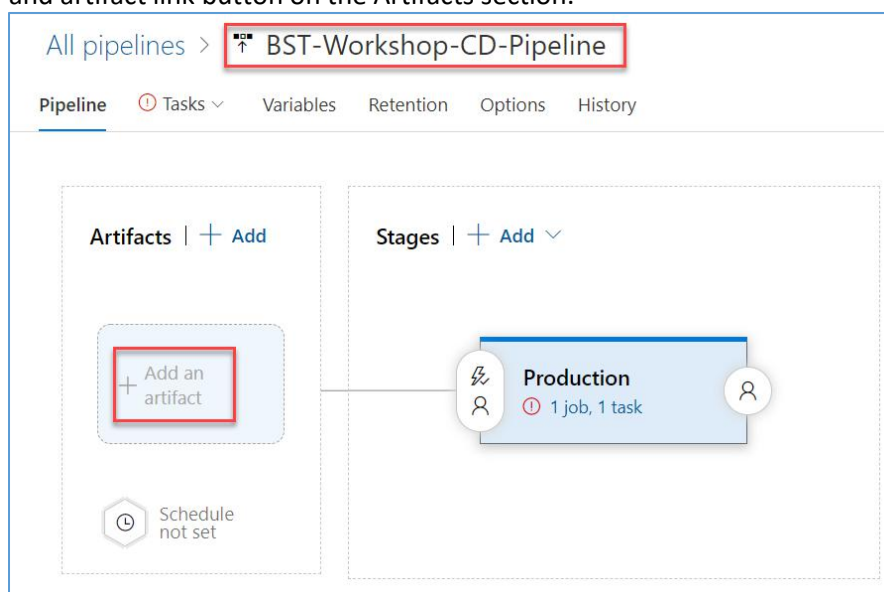
[Learn more](#) [Troubleshoot](#) **Back** **Save**

Module 07: Create a release pipeline

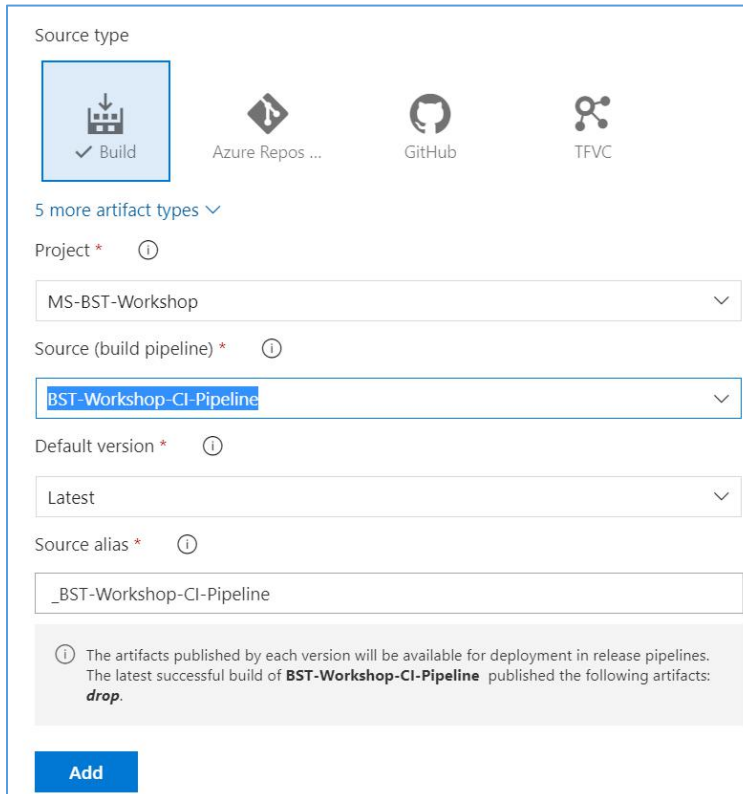
- 1) Click on the **Pipelines** on the left side and chose **Releases**. Click on the '**New pipeline**' button to create a new release pipeline.
- 2) It will open a Release pipeline creation page. From the popped list select '**Azure App Service deployment**'. Specify the Stage name as '**Production**'.



- 3) Update the name of the pipeline to '**BST-Workshop-CD-Pipeline**' and click on the Add and artifact link button on the Artifacts section.



- 4) Select artifact source type as **Build** and select the '**BST-Workshop-CI-Pipeline**' which you have created in the build pipeline module. Leave the other values as default and click 'Add'.



Source type

✓ Build Azure Repos ... GitHub TFVC

5 more artifact types ▾

Project * ⓘ

MS-BST-Workshop ▾

Source (build pipeline) * ⓘ

BST-Workshop-CI-Pipeline ▾

Default version * ⓘ

Latest ▾

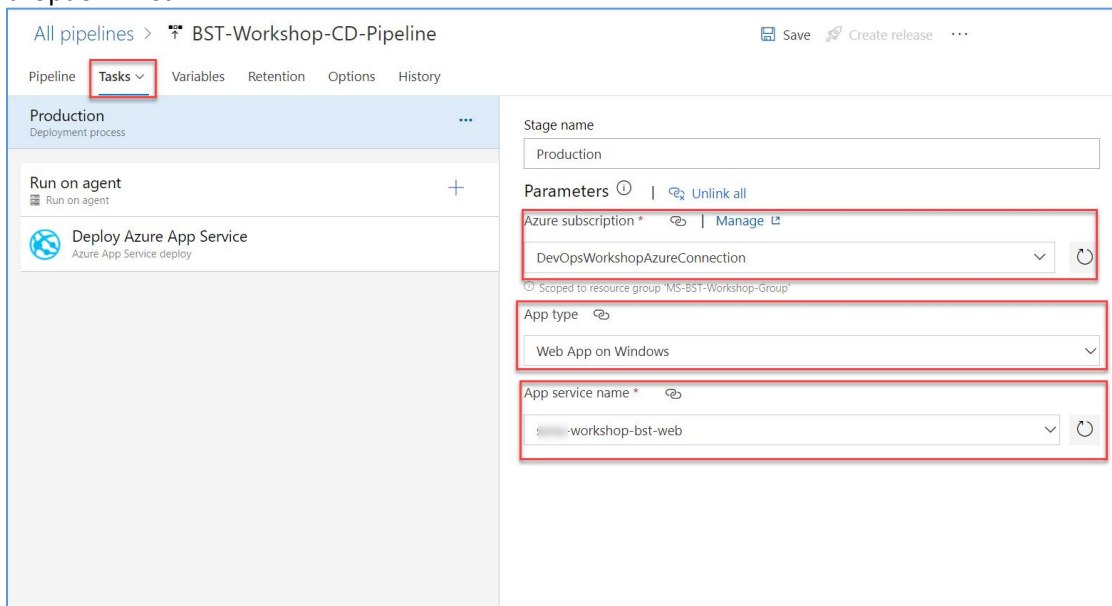
Source alias * ⓘ

_BST-Workshop-CI-Pipeline

ⓘ The artifacts published by each version will be available for deployment in release pipelines. The latest successful build of **BST-Workshop-CI-Pipeline** published the following artifacts: **drop**.

Add

- 5) Click on the **Tasks** tab and select the service connection name which you have created in the previous name for the Azure subscription. Select the App type as '**Web App on Windows**'. Select the App Service web app name from the '**App Service name**' dropdown list.



All pipelines > BST-Workshop-CD-Pipeline Save Create release ...

Pipeline Tasks Variables Retention Options History

Production Deployment process ...

Run on agent +

Deploy Azure App Service Azure App Service deploy

Stage name

Production

Parameters ⓘ | Unlink all

Azure subscription * ⓘ | Manage

DevOpsWorkshopAzureConnection ▾ ↻

ⓘ Scoped to resource group 'MS-BST-Workshop-Group'

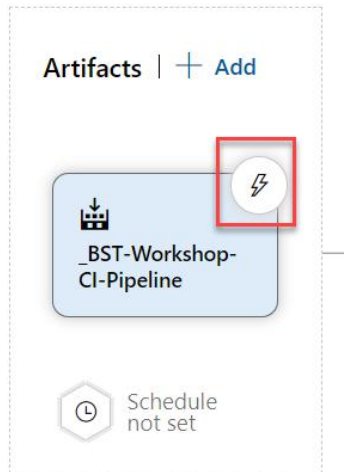
App type ⓘ

Web App on Windows ▾

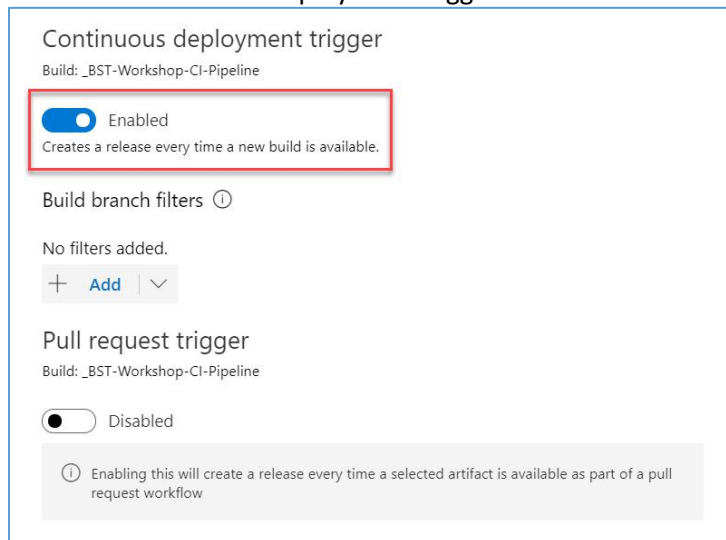
App service name * ⓘ

...-workshop-bst-web ▾ ↻

- 6) Switch back to the 'Pipelines' tab and click on the 'Continues deployment trigger' icon on the 'Artifact' you have selected.



7) Enable the Continuous deployment trigger switch.



8) Click on the **Save** button on the top.

9) Click on the **'Create release'** button next to the **Save** button. In the popped dialog box click on the **'Create'** button.

Create a new release

BST-Workshop-CD-Pipeline

⚡ Pipeline ^
Click on a stage to change its trigger from automated to manual.

⚡ Production

Stages for a trigger change from automated to manual. ⓘ

📦 Artifacts ^
Select the version for the artifact sources for this release

| Source alias | Version |
|---------------------------|------------|
| _BST-Workshop-CI-Pipeline | 20201202.1 |

Release description

Create
Cancel

- Once the release pipeline is created, it will start deploying the application to the App Service web app.

Module 08: Update the Database connection string for App Service

- You have successfully deployed the application to the App Service. But to successfully running the application we need to update the database connection string in the App service Configuration.
- Open the SQL Database you have created and click on the '**Show database connection strings**' from the **Overview** page. You can also select '**Connection Strings**' from the **Settings** section.
- Copy the connectionstring from the ADO.NET section.
- Open the App Service Web App you have created for deploying the .net core web application. Scroll down the left panel and select **Configuration** from the '**Settings**'.
- Under the **Application Settings** tab you can see the **Connection Strings** section.

Connection strings

Connection strings are encrypted at rest and transmitted over an encrypted channel.

+ New connection string
👁 Show values
✎ Advanced edit

🔍 Filter connection strings

| Name | Value | Source |
|------|-------|--------|
|------|-------|--------|

- 6) Click on the '**New Connection String**' button. It will open a dialog box. Specify the name of the connection string as '**DefaultConnection**'. (It should match with the connection string name in the appsettings.json file of your application). Paste the copied connection string in the **Value** textbox and update the database password in the connection string. Select the **Type** as **SQLServer** and click **OK**.

Add/Edit connection string >

| | |
|--|---|
| Name | DefaultConnection |
| Value | Server=tcp:--workshop-sql.database.windows.net,1433;Initial Catalog=WS-WorkshopDb;Persist Se... |
| Type | SQLServer |
| <input type="checkbox"/> Deployment slot setting | |

OK **Cancel**

- 7) This will update the application configuration and the application will restart. You can now navigate to the **Overview** page and browse the application by navigating to the web app url.
- 8) It will open the Swagger UI page as you seen in the local machine. Create a new user and saving account for the user. Check the application is working correctly.