**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:**

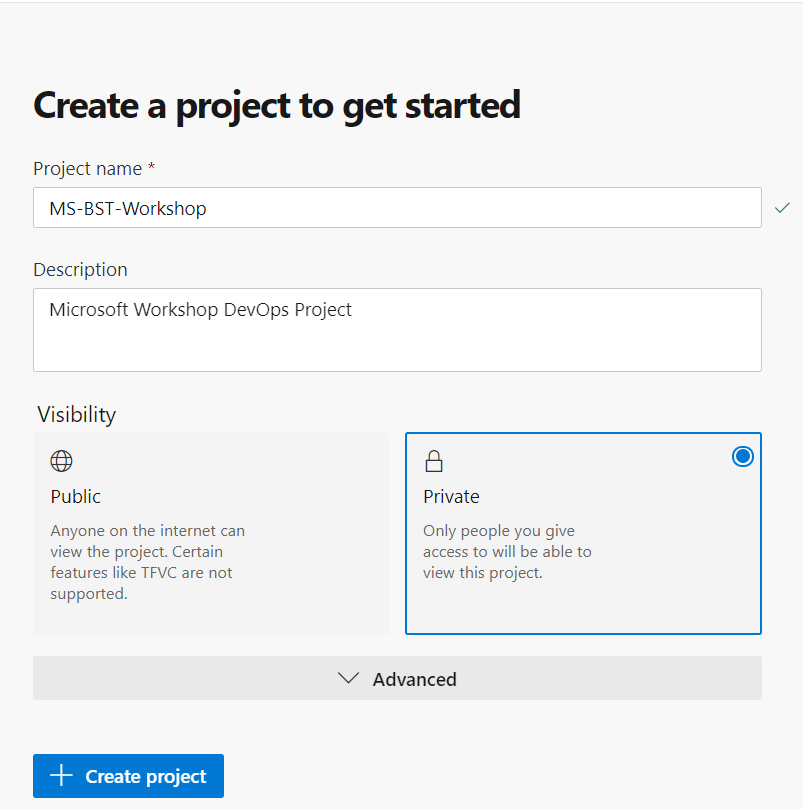
* Git
* .NET Core 3.1
* Visual Studio 2019 Community/Enterprise
* Azure DevOps Subscription
* Azure Subscription
* Visual Studio Code (Optional)

**Case Study:**

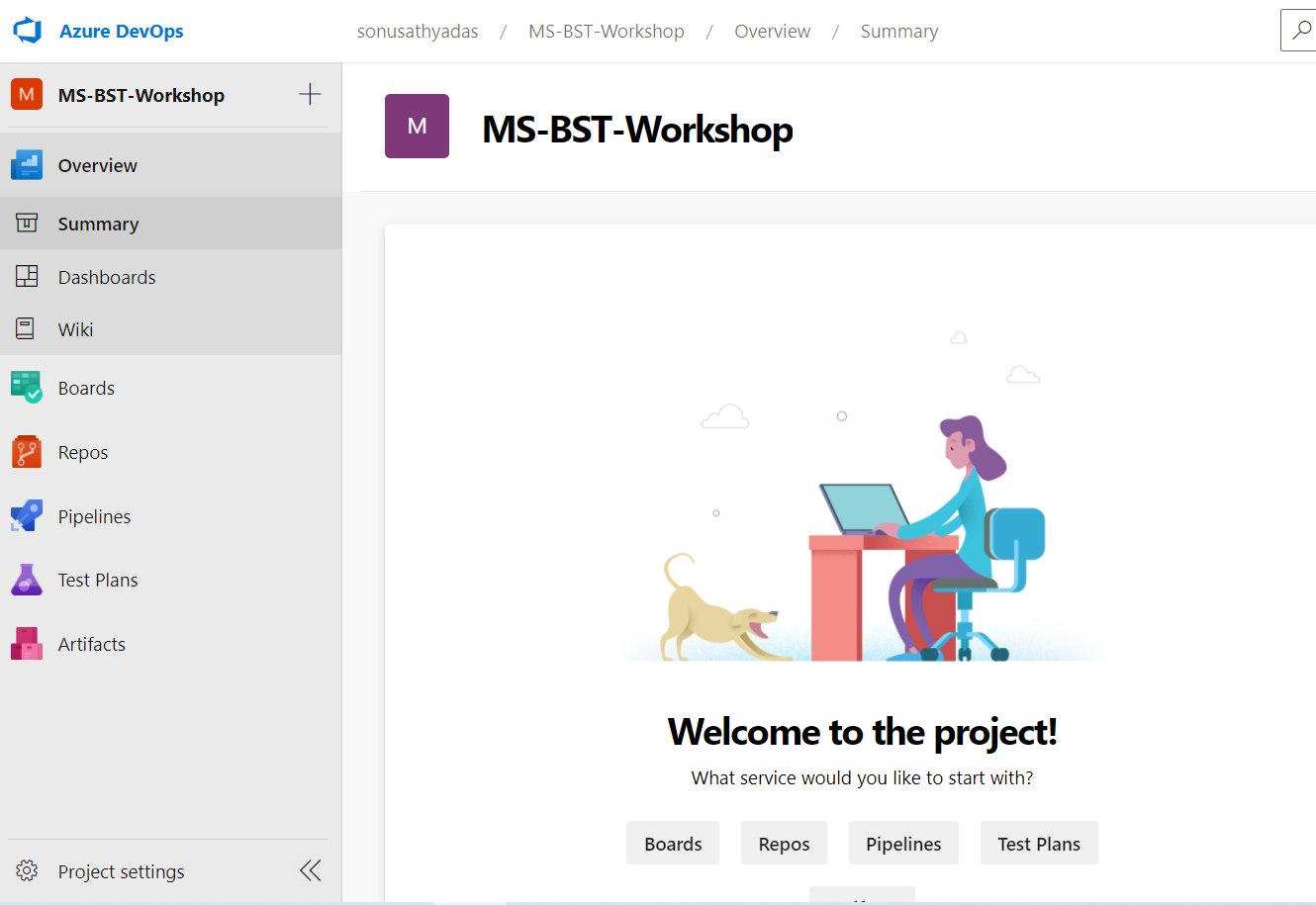
BST Bank is currently using the on-premise servers for deploying databases and web applications. BST Bank currently using a monolithic application developed in .NET framework. They want to migrate the application and data from on-premise to Azure cloud to benefit the features of the cloud infrastructure. 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 continues integration and deployment of the application into Azure services.

**Module 01: Creating Azure DevOps Project**

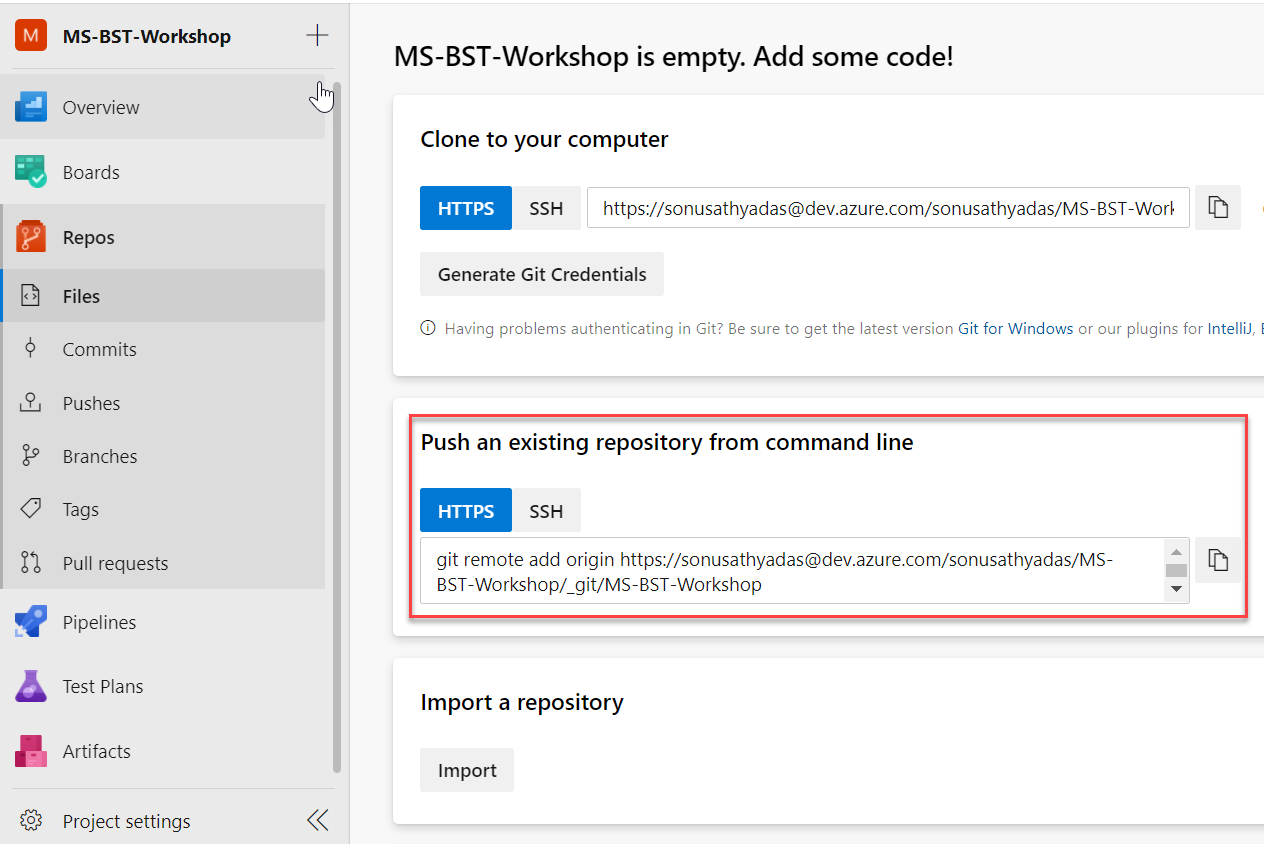
1. Open the browser and navigate to <https://dev.azure.com>
2. Sign up for the Azure DevOps account, if you don’t have it already. Click on the ‘Sign in to Azure DevOps’ to login if you already have an account.
3. After you have logged in successfully, you can create your first DevOps project. Specify 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.



1. 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.



1. 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 Resouce 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 rul 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 clone the git repository to your machine. You can also download the repository 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 staus message and user object. User object contains an **identityId** field. You need to note this id to a text file. This is require whenever you perform any customer related operations later.