# AZURE DEVOPS TUTORIAL: BUILDING CI/CD PIPELINE FOR JAVA APP WITH MYSQL

Hitesh Sharma (hiteshs@kth.se) and Avijit Roy (avijit@kth.se)

KTH DEVOPS COURSE (DD2482)

# Table of Contents

## Overview:

Innovation and ability to compete in marketplace are some of the main factors to become successful in industry. Continuous integration and continuous delivery help companies to deliver well and deliver often and allows companies to compete in market. Azure DevOps makes this CI CD process setup easy and flexible. It can start with code available in git repository such as GitHub or by uploading the code in repo provided by Azure. You can build and deploy application in different azure services such as virtual machine, app service, Azure Kubernetes Service or even in on-premises. In this tutorial you will see how to setup a CI/CD pipeline starting from creating a project in azure to deploy application on live server.

### Things covered in this tutorial:

1. Creation of project with scrum settings in azure portal.
2. Uploading code in azure repo.
3. Deploy an Azure Container Registration repo.
4. Setup build pipeline (continuous integration) for a java web application with MySql database.
5. Run build pipeline.
6. Deploy an Azure web app for container
7. Deploy an Azure MySQL database server.
8. Setup release pipeline (continuous delivery) for staging and production environment.
9. Run release pipeline.

### Prerequisite:

1. Microsoft Azure account: To follow this tutorial you need to have an active Microsoft Azure Account. You can create free trial account.
2. Azure DevOps Account: You can create a DevOps account in Azure using this link.
3. A java web application with test project. You can use your own code, or the code used for this tutorial.
4. An Azure Container Registry service, an Azure web app service for container and an Azure MySql database server.

### Java app:  Overview

For this tutorial we will use Java maven project with MySQL to store and retrieve data. We will use this web app called **MyShuttle** where employees can login used login page and see their trip expenses. We need web server with tomcat 9 to deploy the web app and a MySQL database server. For this tutorial, we will create docker image to deploy application and use Azure web app service to host the app. dockerfile, Docker-compose file and database script file, all are included in code repo. You can download code from:

**https://aviroy1988@dev.azure.com/aviroy1988/My_Shuttel_Web_CI_CD/_git/MyShuttelWeb**

FUN FACT

*Why is Azure DevOps most popular over other cloud vendors?*

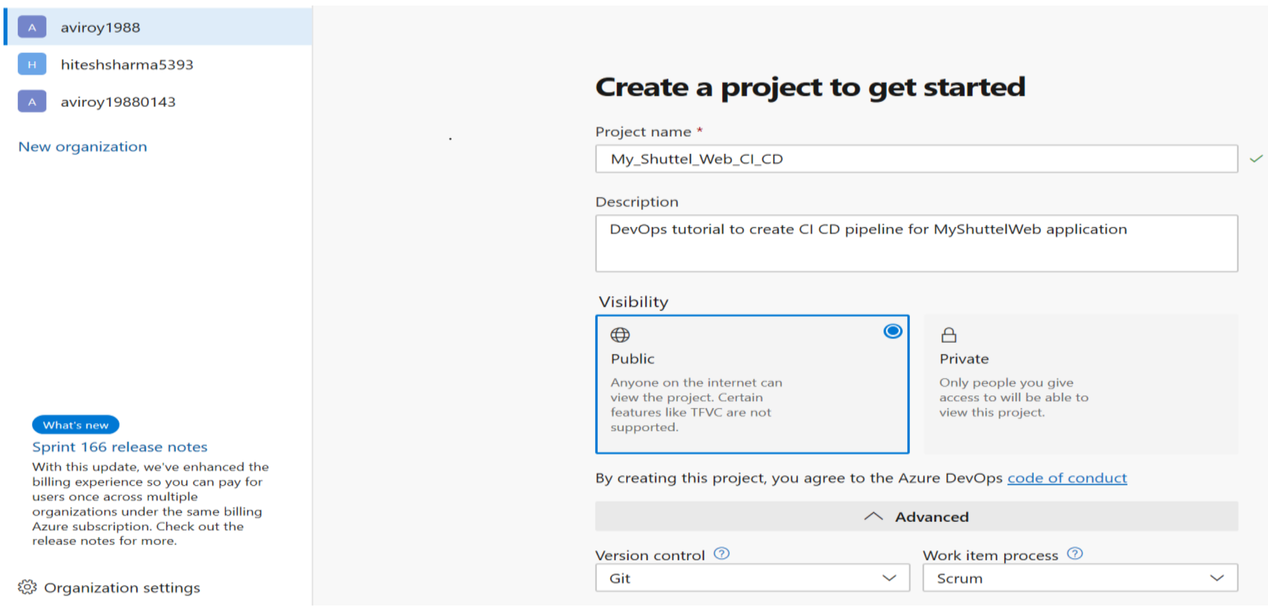
*It allows users to prepare tasks or utilize the existing once i.e. facilitates the Power of open source Community.*

*Azure DevOps gives unlimited private agile planning, Git hosting, continuous integration and release management for continuous delivery to the cloud and on-premises as opposed to its rival AWS.*

## Part 1: Project creation in Azure

In this part of this tutorial, we will create a new project in Azure.

1. Sign into the Microsoft Azure Portal.
2. Select organization from left panel and then click New Project. Give a project name, visibility type (Public or Private).
3. Choose version control as Git and work item process as Scrum.


A aviroy1988
H hiteshsharma5393
A aviroy19880143
New organization

What's new
Sprint 166 release notes
With this update, we've enhanced the billing experience so you can pay for users once across multiple organizations under the same billing Azure subscription. Check out the release notes for more.

Organization settings

Create a project to get started

Project name *
My_Shuttel_Web_CI_CD

Description
DevOps tutorial to create CI CD pipeline for MyShuttelWeb application

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.

By creating this project, you agree to the Azure DevOps code of conduct

Advanced

Version control: Git
Work item process: Scrum

*Figure 1: Project creation*

This project demonstrates the complete structure of a project that we do in a project with actual industrial setup. Azure DevOps provides us all those necessary tools under one umbrella called Azure DevOps. It is interesting and easy for any DevOps team to manage. Figure 1 shows summary of newly created project.

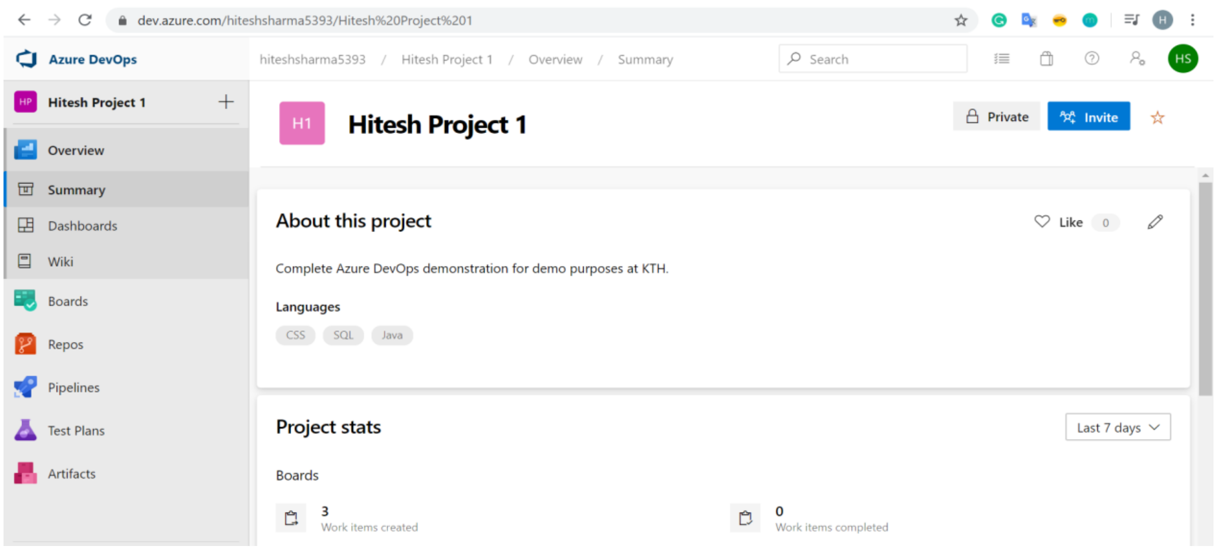


*Figure 2: Project Summary*

Azure provides azure boards that display the work items of the project depending on the work item process type. Scrum is used for this tutorial. Azure dashboard provides a customizable page to view work items, bugs, test cases and many more. You can add the widgets (what we call gadgets in Jira) from this side bar to get view of different items. It is just so easy method of drag and drop. It works in the same way just like any software development tool for Agile teams.

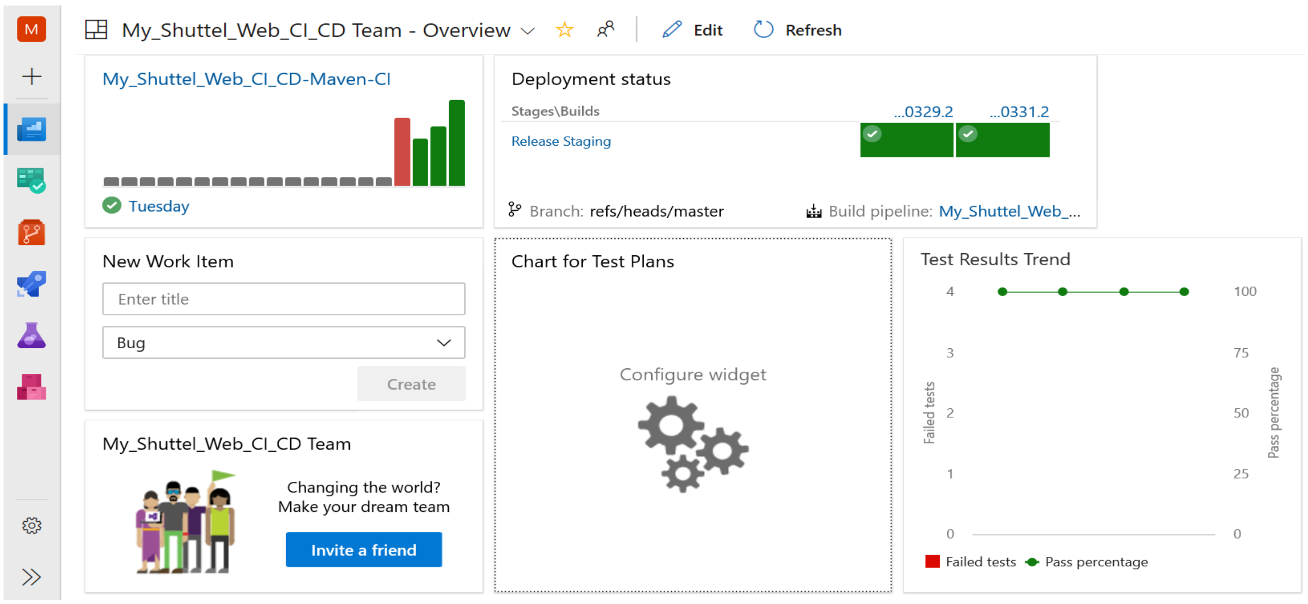
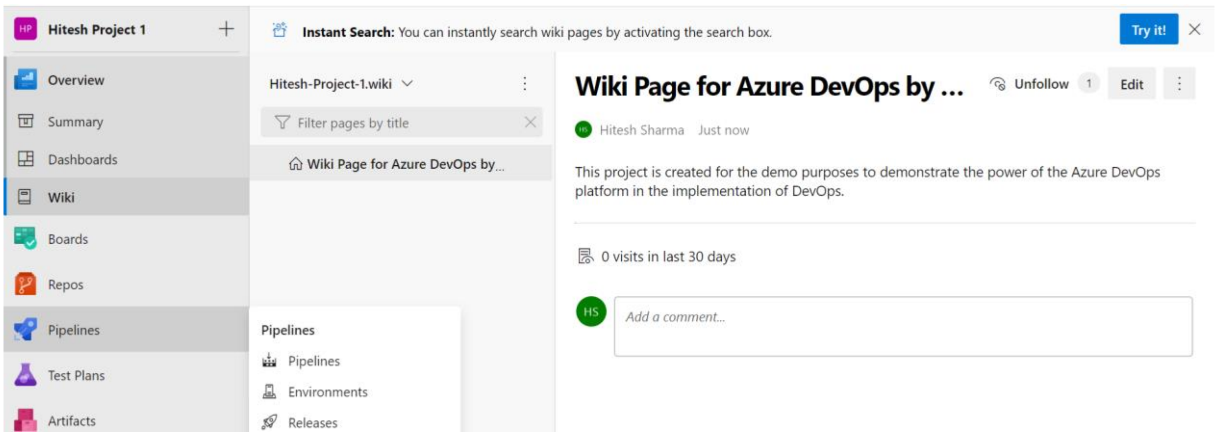


*Figure 3: Project Dashboard*

Wiki Page in Azure DevOps just like any other Wiki page used in Project documents of a company.



Wiki Page for Azure DevOps by ...

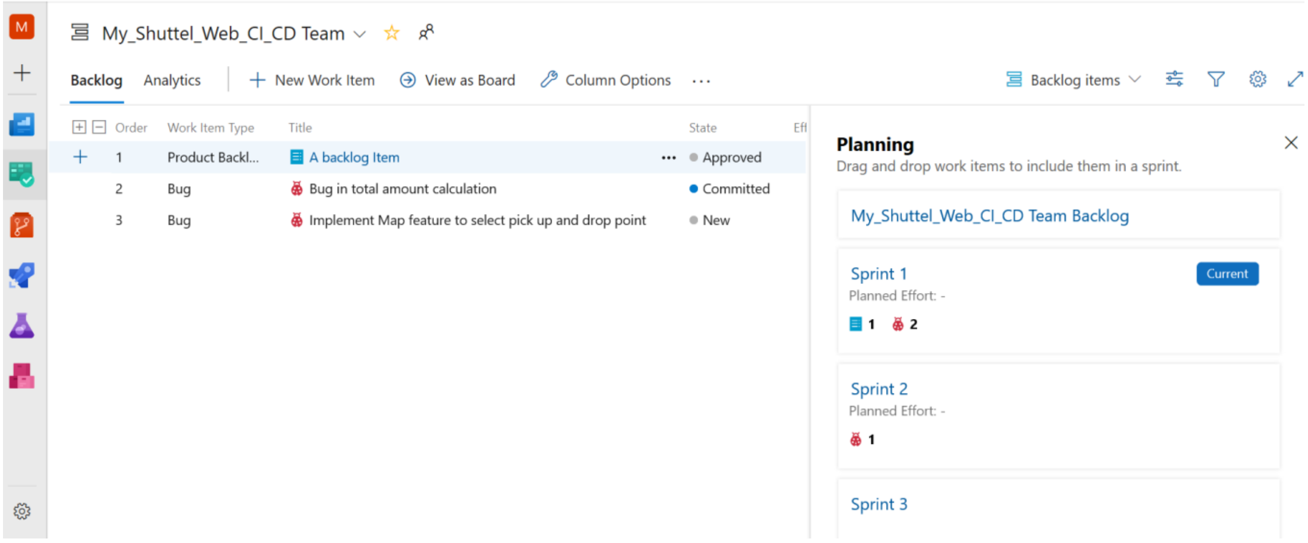
Hitesh Sharma Just now

This project is created for the demo purposes to demonstrate the power of the Azure DevOps platform in the implementation of DevOps.

0 visits in last 30 days

*Figure 4: Project Wiki*

Backlogs page provides backlogs view of the project.



| Order | Work Item Type | Title | State |
|---|---|---|---|
| 1 | Product Backl... | A backlog Item | Approved |
| 2 | Bug | Bug in total amount calculation | Committed |
| 3 | Bug | Implement Map feature to select pick up and drop point | New |

*Figure 5: Project Backlogs*

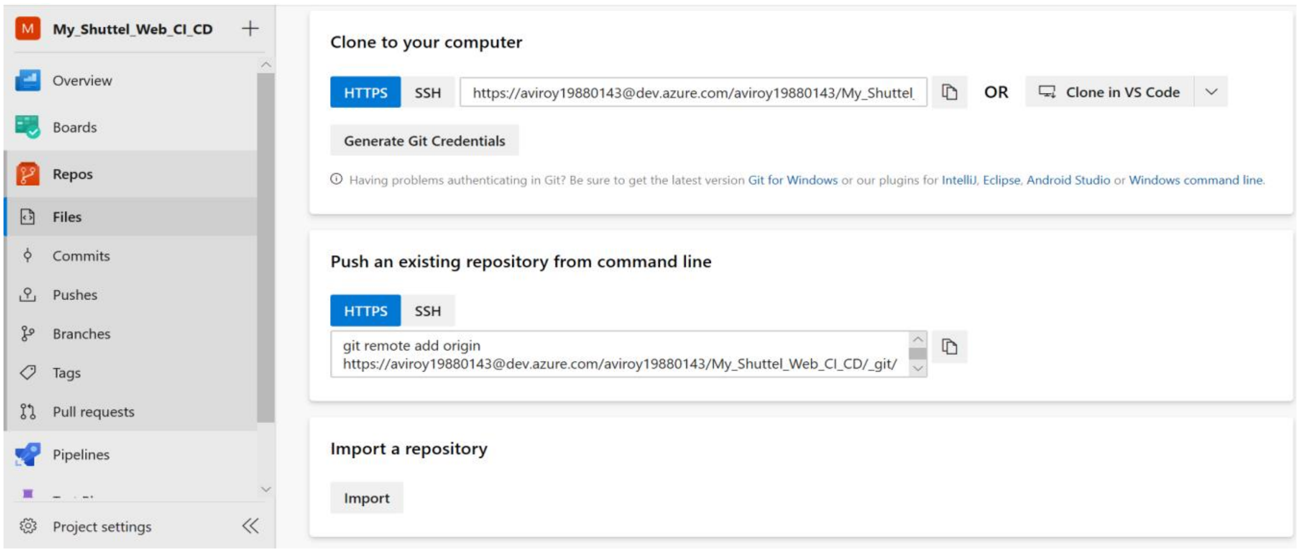
For making the document concise and meaningful for DevOps core viz. CI/CD. We jump to the next section now.

Jumping to the core of DevOps which are repos and pipelines that facilitates CI/CD in Azure DevOps.

## Part 2: Repo creation and code upload

For this tutorial a dummy code has been taken from a random repository on GitHub and added to the Azure. This is done to show that instead of just integrating any version control system, Azure DevOps also provides its own code collaboration and version control tools for offering repository management. This can be accessed as shown in figure bellow. We can push the latest revision of code easily just like in any repository. If one has knowledge of setup of any such system, then implementation of version control system is straightforward here.

1. Create new repo in Azure Git Repo.



Clone to your computer

https://aviroy19880143@dev.azure.com/aviroy19880143/My_Shuttel

Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Windows command line.

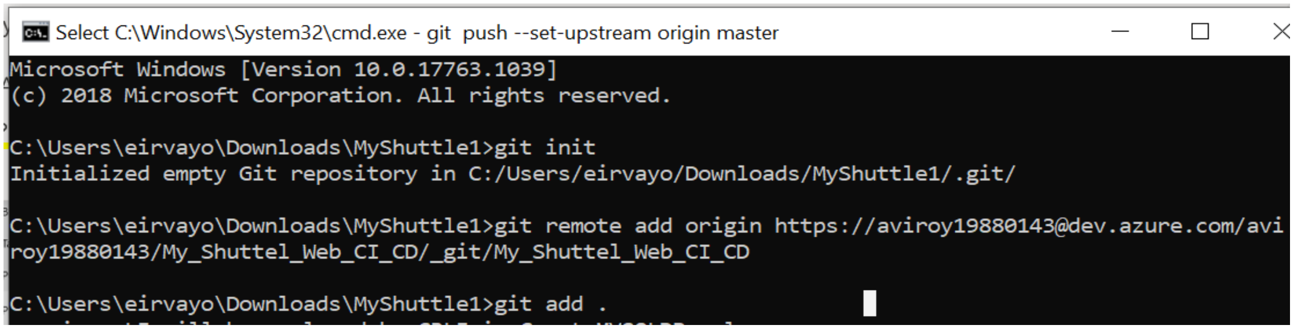
Push an existing repository from command line

```
git remote add origin
https://aviroy19880143@dev.azure.com/aviroy19880143/My_Shuttel_Web_CI_CD/_git/
```

Import a repository

*Figure 6: Azure Repo creation*

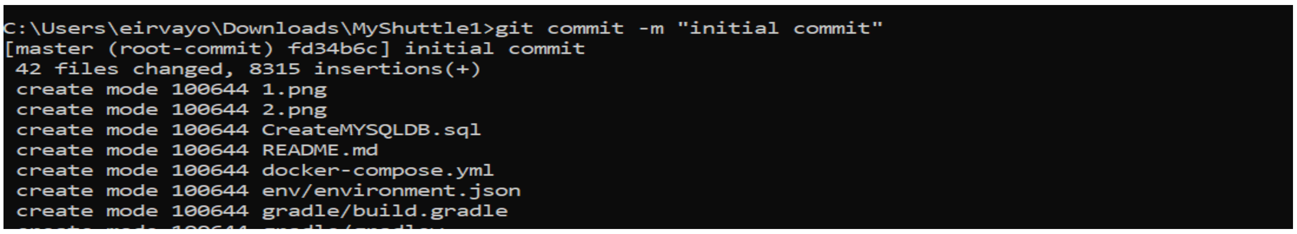
2. Initialize empty repo in local where code is present using **git init** command in command prompt.
3. Add remote repo as Azure repo (As instructed in Azure repo).
4. Use **git add** command to add all file in local git repo.



Select C:\Windows\System32\cmd.exe - git  push --set-upstream origin master

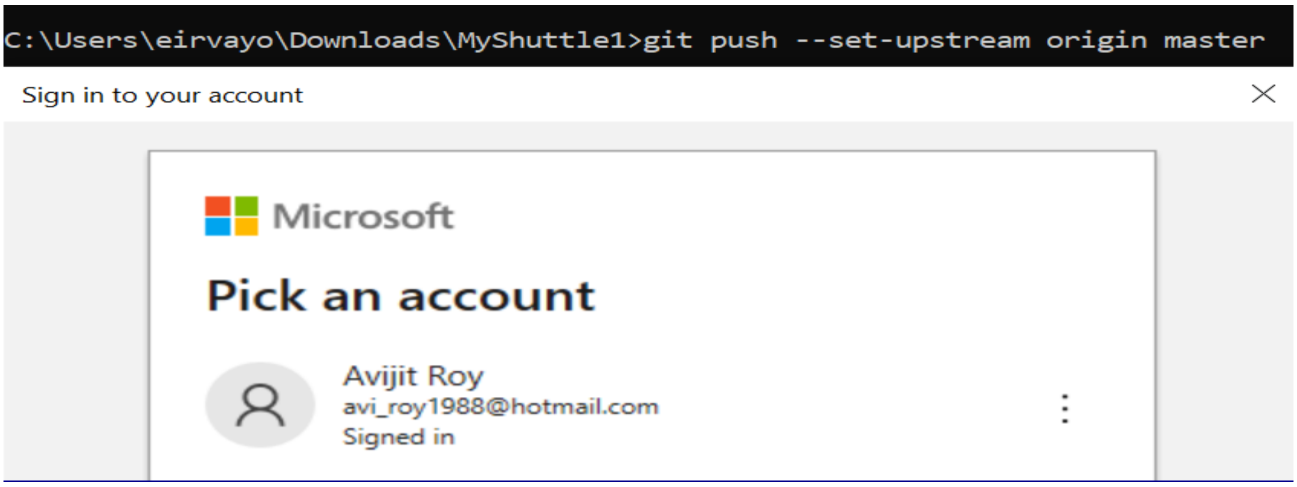
```
Microsoft Windows [Version 10.0.17763.1039]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\eirvayo\Downloads\MyShuttle1>git init
Initialized empty Git repository in C:/Users/eirvayo/Downloads/MyShuttle1/.git/

C:\Users\eirvayo\Downloads\MyShuttle1>git remote add origin https://aviroy19880143@dev.azure.com/avi
roy19880143/My_Shuttel_Web_CI_CD/_git/My_Shuttel_Web_CI_CD

C:\Users\eirvayo\Downloads\MyShuttle1>git add .
```

5. Commit new files in local git repo using **git commit**.



```
C:\Users\eirvayo\Downloads\MyShuttle1>git commit -m "initial commit"
[master (root-commit) fd34b6c] initial commit
 42 files changed, 8315 insertions(+)
 create mode 100644 1.png
 create mode 100644 2.png
 create mode 100644 CreateMYSQLDB.sql
 create mode 100644 README.md
 create mode 100644 docker-compose.yml
 create mode 100644 env/environment.json
 create mode 100644 gradle/build.gradle
```

6. Push files in remote git repo (Azure git). Authorize using Azure portal credential.



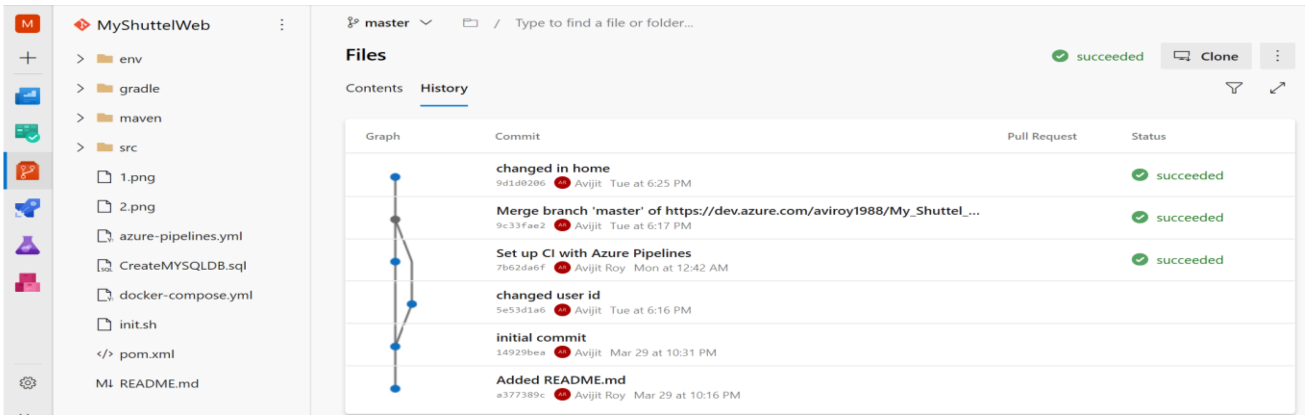
```
C:\Users\eirvayo\Downloads\MyShuttle1>git push --set-upstream origin master
```

Sign in to your account

Microsoft

Pick an account

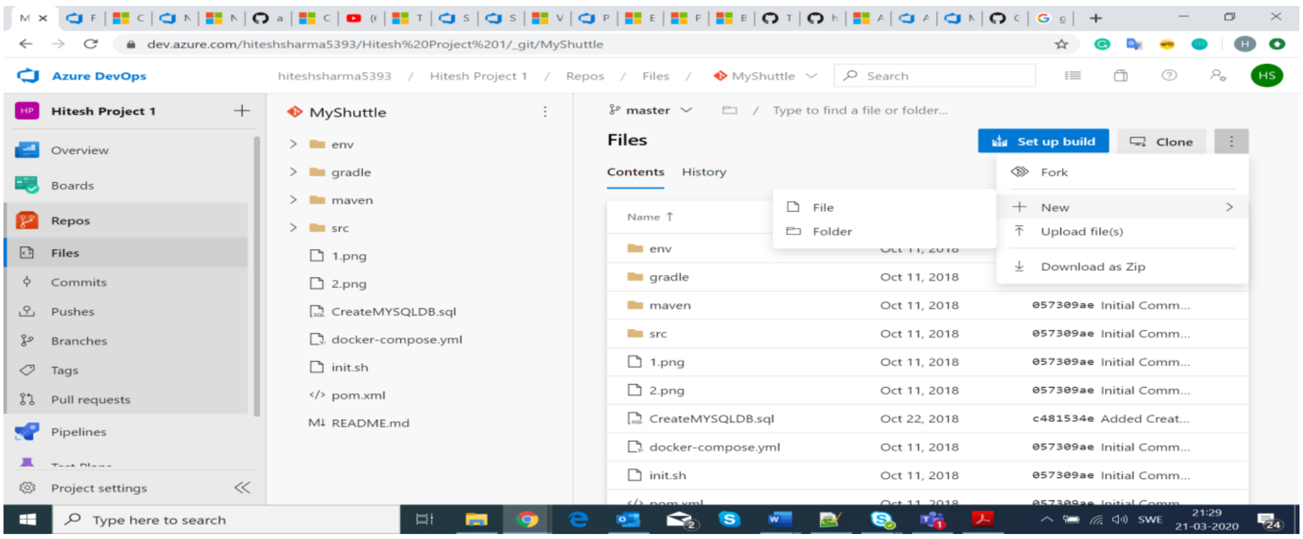
Avijit Roy
avi_roy1988@hotmail.com
Signed in



| Graph | Commit | Pull Request | Status |
|---|---|---|---|
| | changed in home<br>9d1d0206 Avijit Tue at 6:25 PM | | succeeded |
| | Merge branch 'master' of https://dev.azure.com/aviroy1988/My_Shuttel_...<br>9c33fae2 Avijit Tue at 6:17 PM | | succeeded |
| | Set up CI with Azure Pipelines<br>7b62da6f Avijit Roy Mon at 12:42 AM | | succeeded |
| | changed user id<br>5e53d1a6 Avijit Tue at 6:16 PM | | |
| | initial commit<br>14929bea Avijit Mar 29 at 10:31 PM | | |
| | Added README.md<br>a377389c Avijit Roy Mar 29 at 10:16 PM | | |

*Figure7: Repo view with uploaded code*

For adding new folder or file directly in repo.:



| Name ↑ | | |
|---|---|---|
| env | Oct 11, 2018 | |
| gradle | Oct 11, 2018 | 057309ae Initial Comm... |
| maven | Oct 11, 2018 | 057309ae Initial Comm... |
| src | Oct 11, 2018 | 057309ae Initial Comm... |
| 1.png | Oct 11, 2018 | 057309ae Initial Comm... |
| 2.png | Oct 11, 2018 | 057309ae Initial Comm... |
| CreateMYSQLDB.sql | Oct 22, 2018 | c481534e Added Creat... |
| docker-compose.yml | Oct 11, 2018 | 057309ae Initial Comm... |
| init.sh | Oct 11, 2018 | 057309ae Initial Comm... |

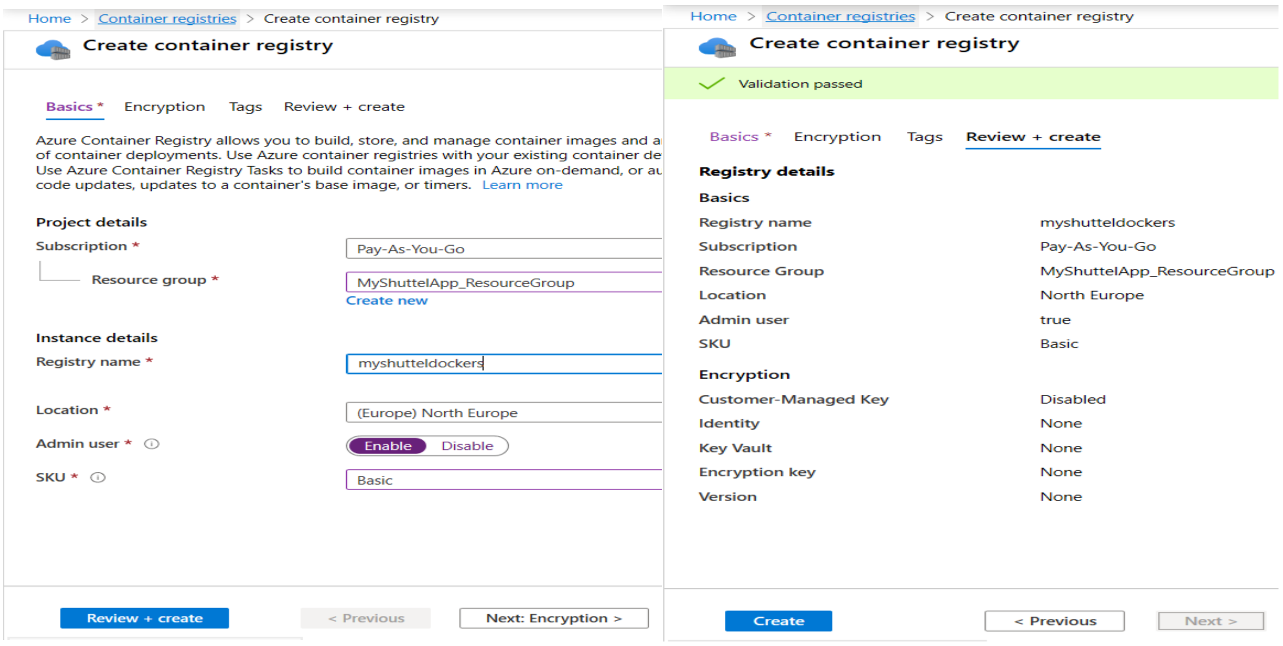
*Figure 8: Option to upload files or folder directly on Azure repo*

Now we have project and our repo ready, let go to next step to build the code. For this we will create build pipeline using pipelines option available in Azure.

## Part 3: Create Azure Container Registry

Azure container registry is a private repo to store docker images. We are using Azure Container registry instead of public/ private registry of dockerhub. That is needed in any company when you don’t want to expose your product on public registry and don’t have expertise in setting up all new registry server. Not only that, Azure never forces you to use its product, one is free to use other registry servers if you like. Integration with such services are comfortably enabled here. To create Azure Container Registry:

- Go to azure portal.
- Click “**Create New resource**”, search for “**container registries**” and hit enter.
- Click **+ Add** button to start creating new registry by providing subscription name, resource group name, registry name, location.
- Enable “**Admin User**” and select SKU as “**Basic**”.


Home > Container registries > Create container registry

**Create container registry**

Basics * | Encryption | Tags | Review + create

Azure Container Registry allows you to build, store, and manage container images and a
of container deployments. Use Azure container registries with your existing container de
Use Azure Container Registry Tasks to build container images in Azure on-demand, or au
code updates, updates to a container's base image, or timers. Learn more

**Project details**

| | |
|---|---|
| Subscription * | Pay-As-You-Go |
| Resource group * | MyShuttelApp_ResourceGroup |

**Instance details**

| | |
|---|---|
| Registry name * | myshutteldockers |
| Location * | (Europe) North Europe |
| Admin user * | [✓ Enable] Disable |
| SKU * | Basic |

Home > Container registries > Create container registry

**Create container registry**

Validation passed

Basics * | Encryption | Tags | Review + create

**Registry details**

**Basics**

| | |
|---|---|
| Registry name | myshutteldockers |
| Subscription | Pay-As-You-Go |
| Resource Group | MyShuttelApp_ResourceGroup |
| Location | North Europe |
| Admin user | true |
| SKU | Basic |

**Encryption**

| | |
|---|---|
| Customer-Managed Key | Disabled |
| Identity | None |
| Key Vault | None |
| Encryption key | None |
| Version | None |

*Figure 9: Azure container registry creation*

<u>FUN FACT</u>

*YAML is a human friendly format. It is the default configuration file format used by Ruby on Rails and used in a number of notable tools such as Ansible and Google App Engine. Here, we have used for Azure Tasks. It was released in 2001 but recently from 2015 it gained popularity.*

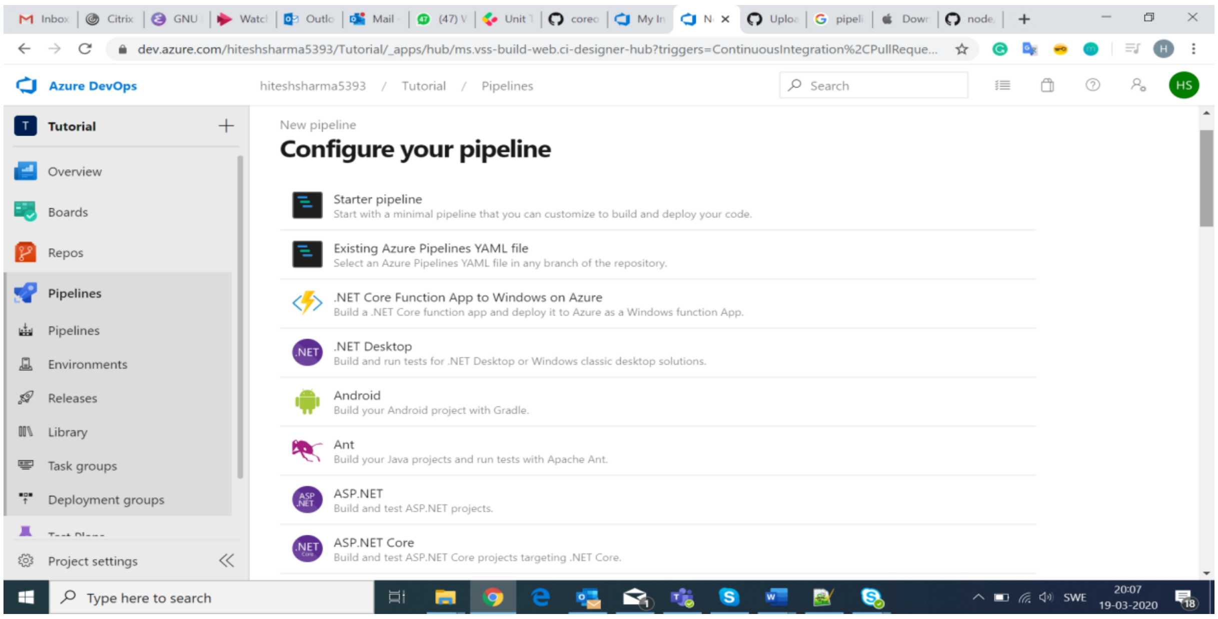
## Part 4: Create build pipeline

Going to the pipeline of DevOps that is quite famous. Azure DevOps provide a complete setup and facilitates the pipelines beautifully in yaml files (other formats are also allowed but it takes yaml by default) unlike the recommended Groovy of our old school Jenkins.

To start with pipeline, select pipelines from left menu, then create new pipeline. This will prompt to select repo for the build and deployment pipeline. Select appropriate repo from list of repos. Next step is to configure the pipeline.

In this step, you can find different option available to configure pipeline. These options are called tasks. Tasks are nothing but the building block to define the automation of build and release process. Task contains packaged scripts that are used to run a job, such as build or release. You can choose a predefined task from the list or create a custom task using yaml file. Figure 8 shows predefined tasks in azure pipeline. These tasks can either be drag and drop from the i) Azure Tasks or ii) its Marketplace or iii) can be self-created using YAML.

For this tutorial, All the following components were picked from the marketplace, it is just as easy as that. For any customisation required we need to modify its YAML and as we know YAML demand indentation rest it is so simple to understand and implement. This shows how Azure makes the so called ‘complex tasks’ really easy to implement. Code reusability is the power here.



New pipeline

Configure your pipeline

Starter pipeline
Start with a minimal pipeline that you can customize to build and deploy your code.

Existing Azure Pipelines YAML file
Select an Azure Pipelines YAML file in any branch of the repository.

.NET Core Function App to Windows on Azure
Build a .NET Core function app and deploy it to Azure as a Windows function App.

.NET Desktop
Build and run tests for .NET Desktop or Windows classic desktop solutions.

Android
Build your Android project with Gradle.

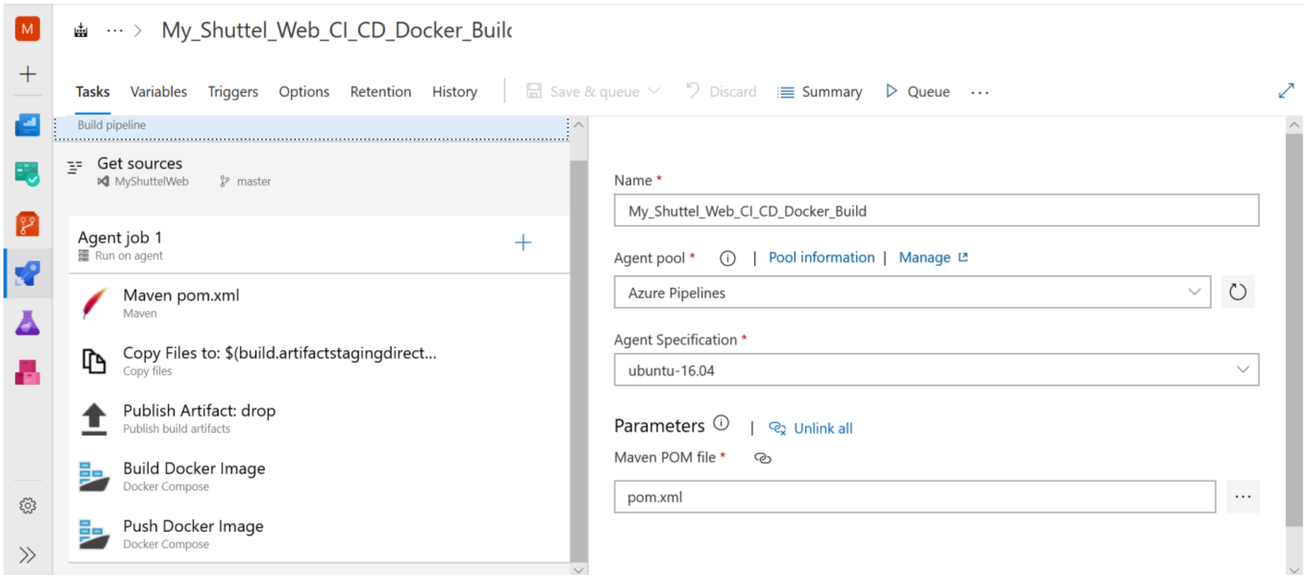
Ant
Build your Java projects and run tests with Apache Ant.

ASP.NET
Build and test ASP.NET projects.

ASP.NET Core
Build and test ASP.NET Core projects targeting .NET Core.

*Figure 10: Task list available in Azure pipelines*

For this project’s implementation we have used following tasks (figure 9) in build pipeline:



Build pipeline

Get sources
MyShuttelWeb master

Agent job 1
Run on agent

Maven pom.xml
Maven

Copy Files to: $(build.artifactstagingdirect...
Copy files

Publish Artifact: drop
Publish build artifacts

Build Docker Image
Docker Compose

Push Docker Image
Docker Compose

Name *
My_Shuttel_Web_CI_CD_Docker_Build

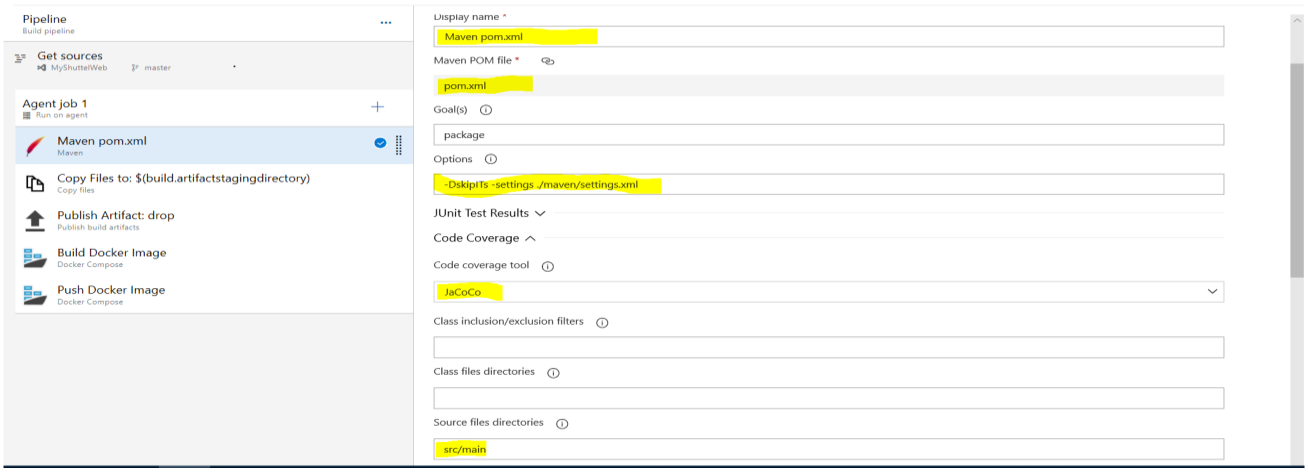
Agent pool * | Pool information | Manage
Azure Pipelines

Agent Specification *
ubuntu-16.04

Parameters | Unlink all
Maven POM file *
pom.xml

*Figure 11: Steps used for build pipeline*

- **Maven pom.xml**: here we have added our java code POM file. This will build the code and create package and change the highlighted fields.



Pipeline
Build pipeline

Get sources
MyShuttleWeb master

Agent job 1
Run on agent

Maven pom.xml
Maven

Copy Files to: $(build.artifactstagingdirectory)
Copy files

Publish Artifact: drop
Publish build artifacts

Build Docker Image
Docker Compose

Push Docker Image
Docker Compose

Display name *
Maven pom.xml

Maven POM file *
pom.xml

Goal(s)
package

Options
-DskipITs -settings ./maven/settings.xml

JUnit Test Results

Code Coverage

Code coverage tool
JaCoCo

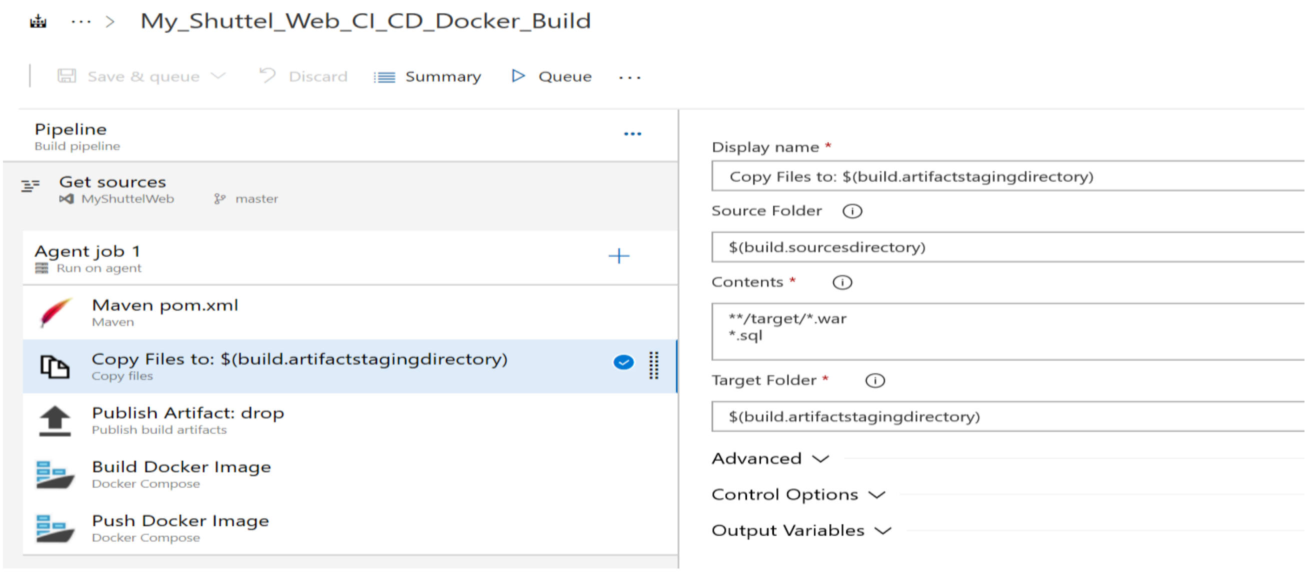
Class inclusion/exclusion filters

Class files directories

Source files directories
src/main

*Figure 12: Pom xml step confiuration*

- **Copy files:** To copy build package (war, jar files) into artifact library.



··· > My_Shuttel_Web_CI_CD_Docker_Build

Save & queue · Discard · Summary · Queue · ···

Pipeline
Build pipeline

Get sources
MyShuttleWeb master

Agent job 1
Run on agent

Maven pom.xml
Maven

Copy Files to: $(build.artifactstagingdirectory)
Copy files

Publish Artifact: drop
Publish build artifacts

Build Docker Image
Docker Compose

Push Docker Image
Docker Compose

Display name *
Copy Files to: $(build.artifactstagingdirectory)

Source Folder
$(build.sourcesdirectory)

Contents *
```
**/target/*.war
*.sql
```

Target Folder *
$(build.artifactstagingdirectory)

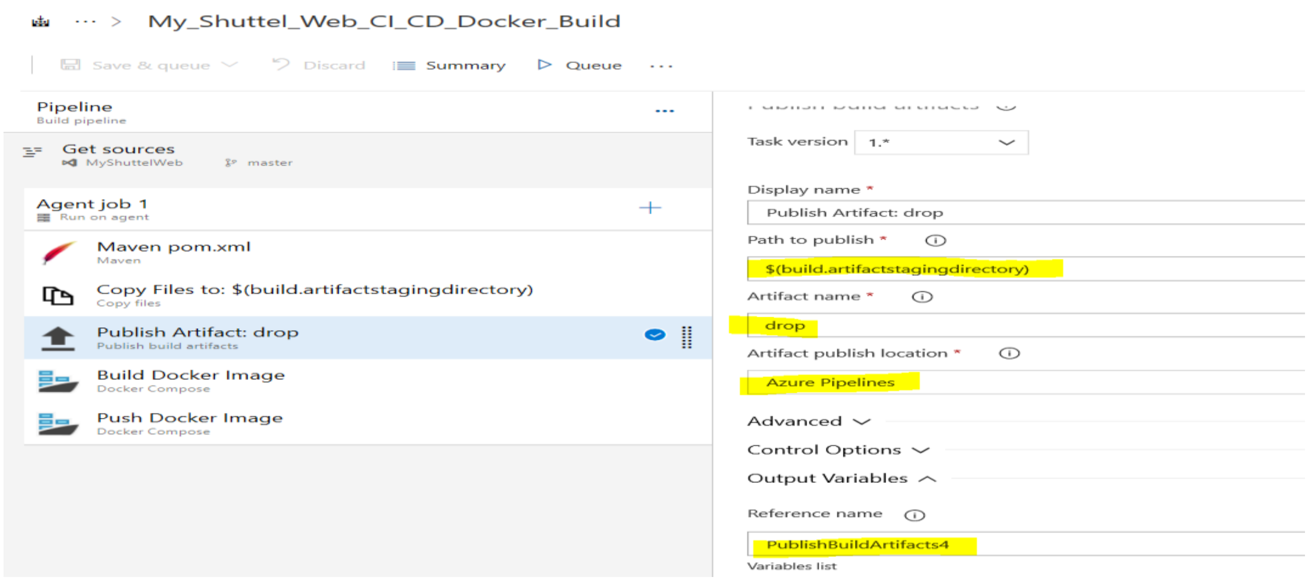
Advanced

Control Options

Output Variables

*Figure 13: Copy files step configuration*

- **Publish Artifact:** To publish artifacts from build into pipeline to make it available for release.



Pipeline
Build pipeline
Get sources
MyShuttelWeb master
Agent job 1
Run on agent
Maven pom.xml
Maven
Copy Files to: $(build.artifactstagingdirectory)
Copy files
Publish Artifact: drop
Publish build artifacts
Build Docker Image
Docker Compose
Push Docker Image
Docker Compose

Task version: 1.*
Display name *: Publish Artifact: drop
Path to publish *: $(build.artifactstagingdirectory)
Artifact name *: drop
Artifact publish location *: Azure Pipelines
Advanced
Control Options
Output Variables
Reference name: PublishBuildArtifacts4
Variables list

*Figure 14: Publish Build Artifacts step configuration*

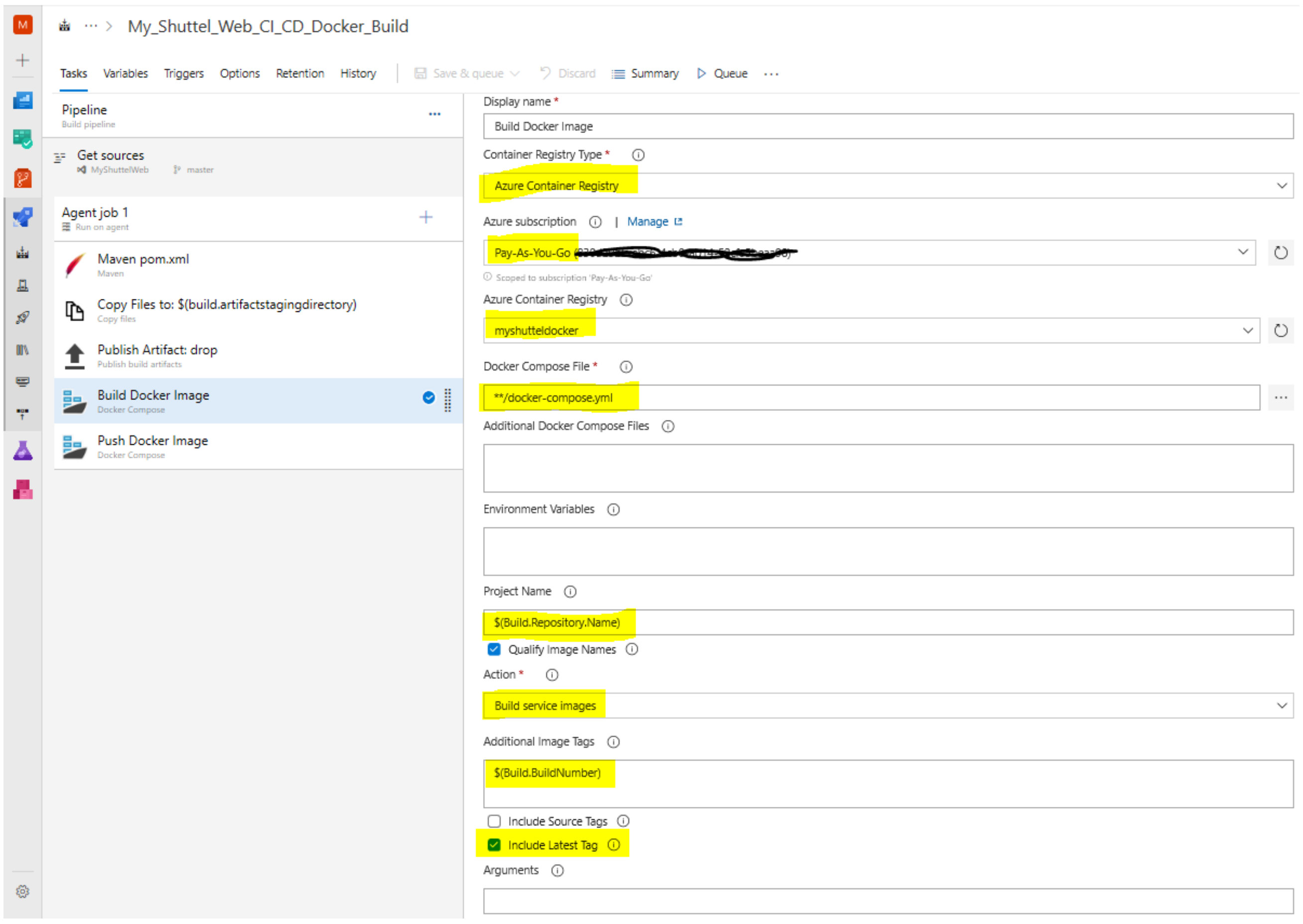
FUN FACT

*Containers packages up an application with all the dependencies, such as libraries, and deploy it as one single package. Docker is getting widely famous because it implements containers. So we have also included docker image in our tutorial.*

- **Build Docker image:** To build a docker image using the docker compose task. Refer docker compose file in repo to find docker file to create docker image. Make the following changes for this task.

| Parameter | Value | Comments |
|---|---|---|
| Azure Subscription | Your authorised subscription | |
| Container Registry Type | Azure Container Registry | |
| Azure Container Registry | Your container registry | |
| Docker Compose file | **/docker-compose.yml | Compose file present in repo. |
| Project Name | $(Build.Repository.Name) | Name of you project |
| Action | Build Service Images | Create Docker image for deployment |
| Additional Image tags | $(Build.BuildNumber) | |
| Include latest tag | True | |
| Qualify Image name | True | |

- **Push Docker image:** Push the docker image in container registry. You can clone "Build Docker image" task and only change "Action" to "Push service images".



Display name * : Build Docker Image

Container Registry Type * : Azure Container Registry

Azure subscription : Pay-As-You-Go [redacted]

Azure Container Registry : myshutteldocker

Docker Compose File * : **/docker-compose.yml

Additional Docker Compose Files :

Environment Variables :

Project Name : $(Build.Repository.Name)

[✓] Qualify Image Names

Action * : Build service images

Additional Image Tags : $(Build.BuildNumber)

[ ] Include Source Tags

[✓] Include Latest Tag

Arguments :

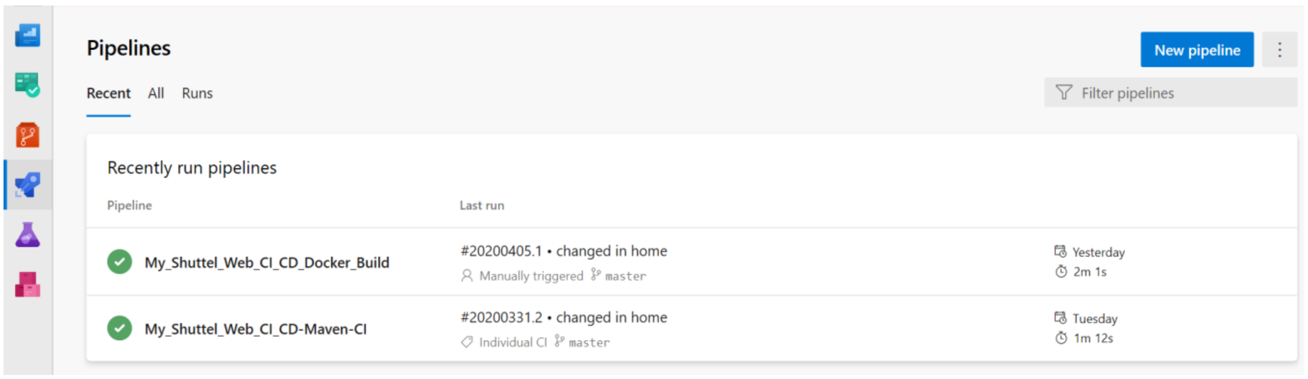
*Figure 15: Build Docker Image step configuration*

You can add these tasks using .yaml file other than user console. Here is the yaml file used for this pipeline.

```
 Job Agent YAML, this includes all the tasks of pipeline:

pool:

  name: Azure Pipelines

  demands: maven

#Your build pipeline references an undefined variable named
'Parameters.mavenPOMFile'. Create or edit the build pipeline for this
YAML file, define the variable on the Variables tab. See
https://go.microsoft.com/fwlink/?linkid=865972

steps:

- task: Maven@3
```

```
  displayName: 'Maven pom.xml'
  inputs:
    mavenPomFile: '$(Parameters.mavenPOMFile)'
    options: '-DskipITs -settings ./maven/settings.xml'
    codeCoverageToolOption: JaCoCo
    codeCoverageSourceDirectories: src/main

- task: CopyFiles@2
  displayName: 'Copy Files to: $(build.artifactstagingdirectory)'
  inputs:
    SourceFolder: '$(build.sourcesdirectory)'
    Contents: |
     **/target/*.war
     *.sql
    TargetFolder: '$(build.artifactstagingdirectory)'
  condition: succeededOrFailed()

- task: PublishBuildArtifacts@1
  displayName: 'Publish Artifact: drop'
  inputs:
    PathtoPublish: '$(build.artifactstagingdirectory)'
  condition: succeededOrFailed()

- task: DockerCompose@0
  displayName: 'Build Docker Image'
  inputs:
    azureSubscription: 'Pay-As-You-Go (XXXXXXXXXX-XXXXXXXXX-XXXXXXXX)'
    azureContainerRegistry: '{"loginServer":"myshutteldocker.azurecr.io",
"id" : "/subscriptions/830d29b5-ebc6-4cb0-ab14-
52cfa5beaa98/resourceGroups/MyShuttelApp_ResourceGroup/providers/Microsof
t.ContainerRegistry/registries/myshutteldocker"}'
    action: 'Build services'
    additionalImageTags: '$(Build.BuildNumber)'

    includeLatestTag: true

- task: DockerCompose@0

  displayName: 'Push Docker Image'


  inputs:

    azureSubscription: 'Pay-As-You-Go (XXXXXXXXXX-XXXXXXXXX-XXXXXXXX)'
azureContainerRegistry: '{"loginServer":"myshutteldocker.azurecr.io",
"id" : "/subscriptions/830d29b5-ebc6-4cb0-ab14-
52cfa5beaa98/resourceGroups/MyShuttelApp_ResourceGroup/providers/Microsof
t.ContainerRegistry/registries/myshutteldocker"}'

    action: 'Push services'

    additionalImageTags: '$(Build.BuildNumber)'

    includeLatestTag: true
```

Once the configuration is finished build pipelines is available in pipelines Azure pipelines. They are nothing but yaml files associated with the Azure 'Tasks'



| Pipeline | Last run | |
|---|---|---|
| My_Shuttel_Web_CI_CD_Docker_Build | #20200405.1 • changed in home<br>Manually triggered master | Yesterday<br>2m 1s |
| My_Shuttel_Web_CI_CD-Maven-CI | #20200331.2 • changed in home<br>Individual CI master | Tuesday<br>1m 12s |

*Figure 16: List of pipelines*

## Part 5: Running a build pipeline

Here it comes to running the pipeline that we just configured, and this is so easy just some click only.

You can run pipeline manually or setup to run automatically whenever there is any push to repository. For this setup trigger for the pipeline and enable continuous integration.

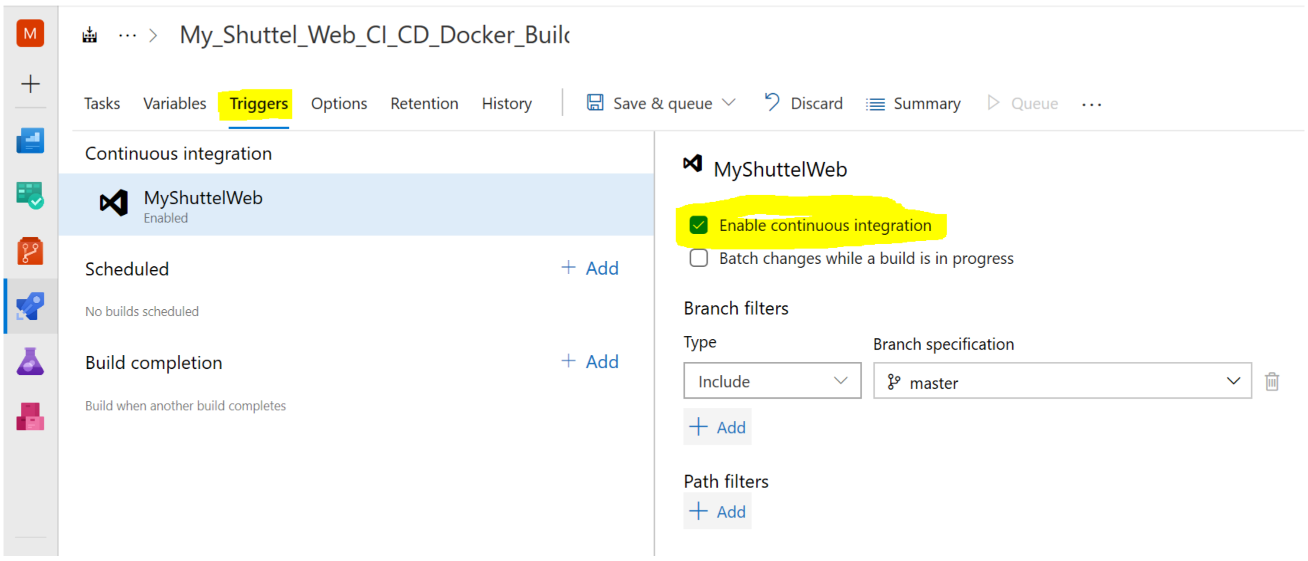


*Figure 17: Enable Continuous integration in build pipeline*

There are various conditions and parameters we can set like one has been shown in following diagram. By default, if all the previous steps in pipeline are success then forthcoming steps are initiated. However, we have opportunity to setup whether the next step in pipeline will be executed or not in case of previous job failure or success. These are self-explanatory options as shown in screenshot bellow:

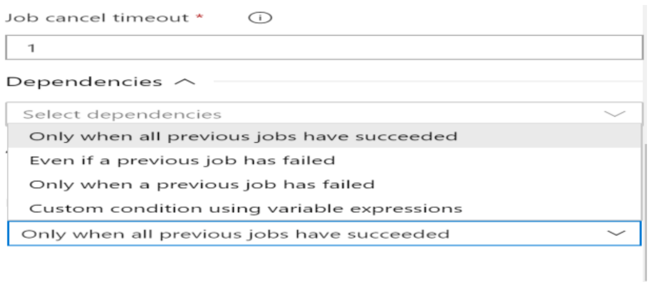

Job cancel timeout *
1
Dependencies
Select dependencies
Only when all previous jobs have succeeded
Even if a previous job has failed
Only when a previous job has failed
Custom condition using variable expressions
Only when all previous jobs have succeeded

*Figure 18: Dependency options to run a task*

To run the pipeline, click on the pipeline that you want to run and click Run pipeline at right top corner.

Here, you have option to choose either default or some specified Linux or Windows machine. Likewise, you can select the branch from which you want the CI pipeline to execute. This makes the pipeline dynamic and gives user a flexibility to graphically interact without altering the underlying pipeline code.

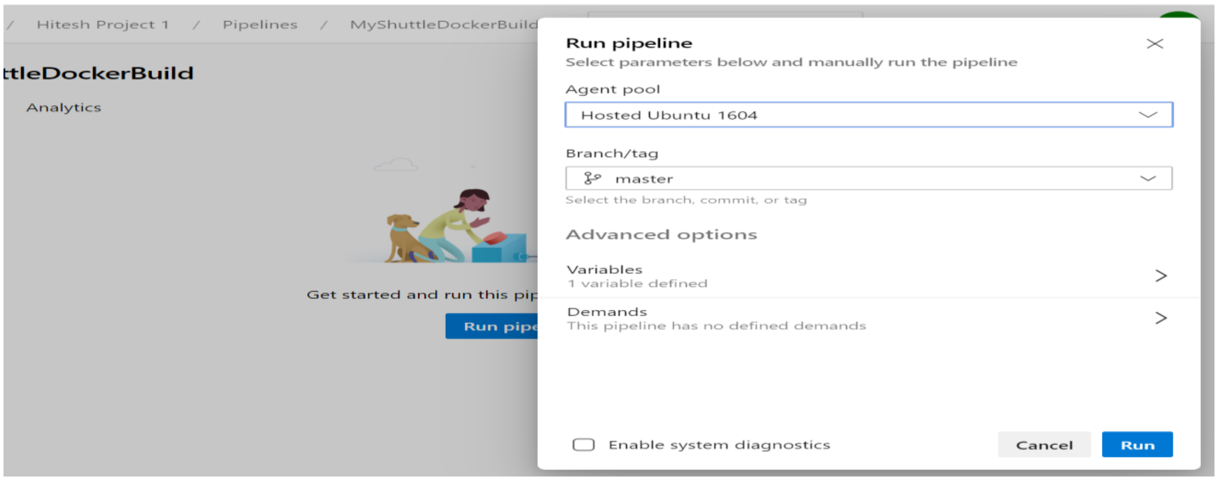
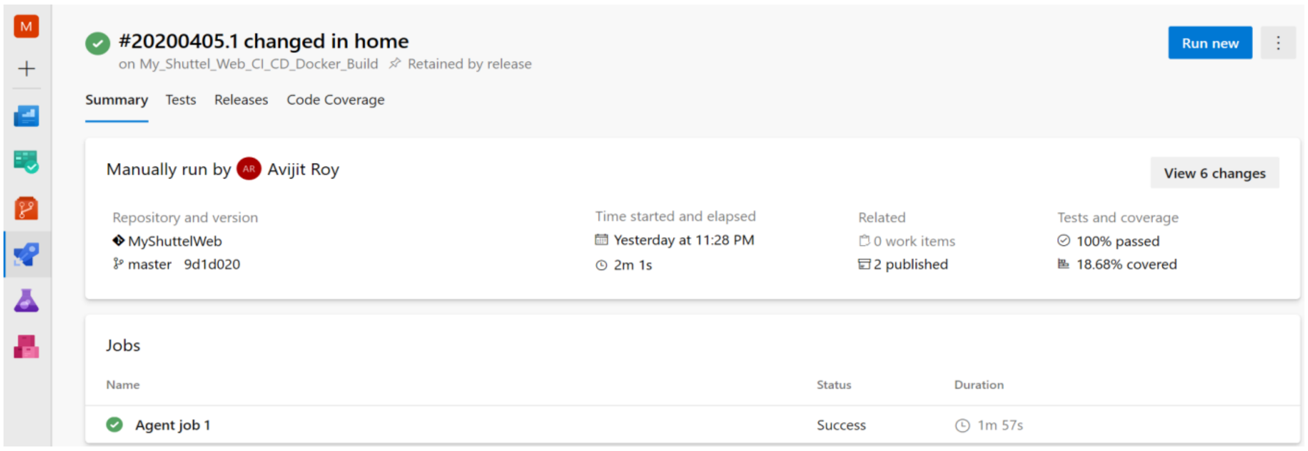

/ Hitesh Project 1 / Pipelines / MyShuttleDockerBuild
ttleDockerBuild
Analytics
Get started and run this pip
Run pipe
Run pipeline
Select parameters below and manually run the pipeline
Agent pool
Hosted Ubuntu 1604
Branch/tag
master
Select the branch, commit, or tag
Advanced options
Variables
1 variable defined
Demands
This pipeline has no defined demands
Enable system diagnostics
Cancel
Run

*Figure 19: Pipeline run configuration*

You can see the running status of all tasks form console. Moreover, once finished azure will send email notification about the run status.



#20200405.1 changed in home
on My_Shuttel_Web_CI_CD_Docker_Build Retained by release

Manually run by Avijit Roy

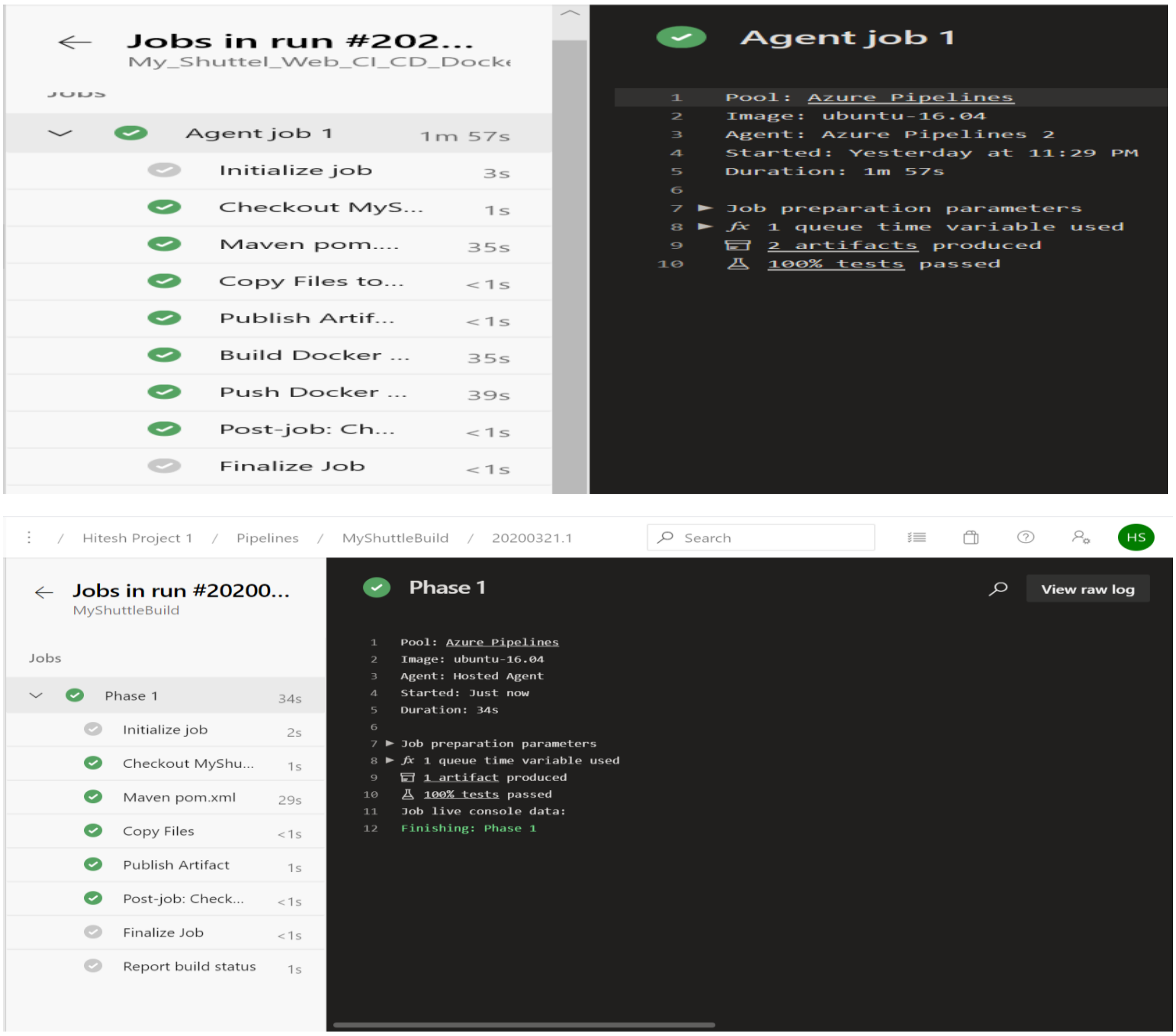
| Repository and version | Time started and elapsed | Related | Tests and coverage |
|---|---|---|---|
| MyShuttelWeb | Yesterday at 11:28 PM | 0 work items | 100% passed |
| master 9d1d020 | 2m 1s | 2 published | 18.68% covered |

Jobs

| Name | Status | Duration |
|---|---|---|
| Agent job 1 | Success | 1m 57s |

*Figure 20: Pipeline run status*

You can see the logs of every steps by clicking on "Agent job 1" for this run.



Jobs in run #202...
My_Shuttel_Web_CI_CD_Docke

| Agent job 1 | 1m 57s |
|---|---|
| Initialize job | 3s |
| Checkout MyS... | 1s |
| Maven pom.... | 35s |
| Copy Files to... | <1s |
| Publish Artif... | <1s |
| Build Docker ... | 35s |
| Push Docker ... | 39s |
| Post-job: Ch... | <1s |
| Finalize Job | <1s |

Agent job 1

```
Pool: Azure Pipelines
Image: ubuntu-16.04
Agent: Azure Pipelines 2
Started: Yesterday at 11:29 PM
Duration: 1m 57s

► Job preparation parameters
► fx 1 queue time variable used
  2 artifacts produced
  100% tests passed
```

Hitesh Project 1 / Pipelines / MyShuttleBuild / 20200321.1

Jobs in run #20200...
MyShuttleBuild

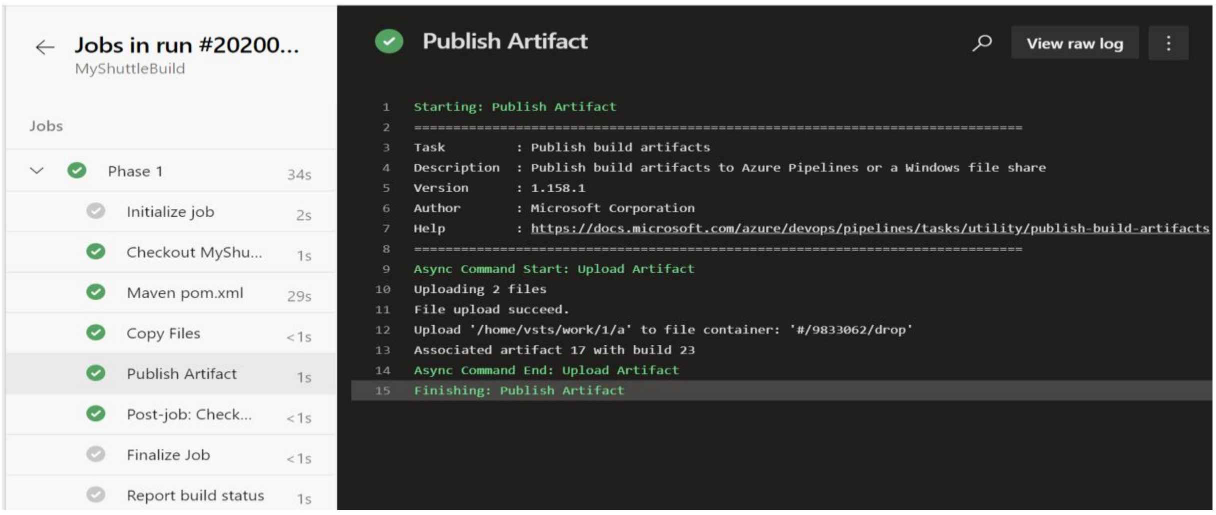
| Phase 1 | 34s |
|---|---|
| Initialize job | 2s |
| Checkout MyShu... | 1s |
| Maven pom.xml | 29s |
| Copy Files | <1s |
| Publish Artifact | 1s |
| Post-job: Check... | <1s |
| Finalize Job | <1s |
| Report build status | 1s |

Phase 1

```
Pool: Azure Pipelines
Image: ubuntu-16.04
Agent: Hosted Agent
Started: Just now
Duration: 34s

► Job preparation parameters
► fx 1 queue time variable used
  1 artifact produced
  100% tests passed
Job live console data:
Finishing: Phase 1
```

*Figure 21: Build pipeline build steps logs*

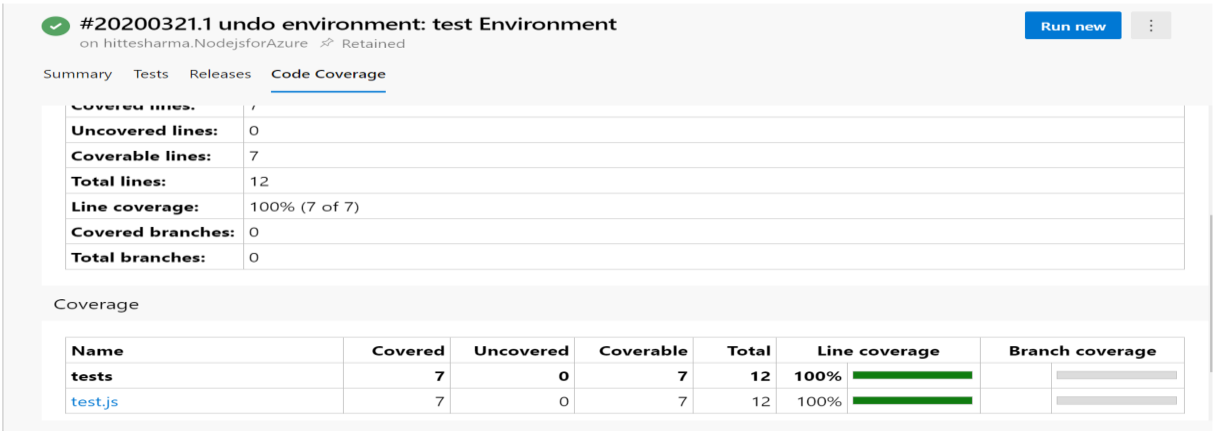
The deployable Artifact so produced could be seen in Azure ‘**Artifacts’** section. They are versioned and kept successively like logs. This is another DevOps tool that Azure provides seamlessly without requiring any infrastructure configuration explicitly.



| | |
|---|---|
| Phase 1 | 34s |
| Initialize job | 2s |
| Checkout MyShu... | 1s |
| Maven pom.xml | 29s |
| Copy Files | <1s |
| Publish Artifact | 1s |
| Post-job: Check... | <1s |
| Finalize Job | <1s |
| Report build status | 1s |

```
Starting: Publish Artifact
==============================================================================
Task         : Publish build artifacts
Description  : Publish build artifacts to Azure Pipelines or a Windows file share
Version      : 1.158.1
Author       : Microsoft Corporation
Help         : https://docs.microsoft.com/azure/devops/pipelines/tasks/utility/publish-build-artifacts
==============================================================================
Async Command Start: Upload Artifact
Uploading 2 files
File upload succeed.
Upload '/home/vsts/work/1/a' to file container: '#/9833062/drop'
Associated artifact 17 with build 23
Async Command End: Upload Artifact
Finishing: Publish Artifact
```

**Code coverage** can easily be seen if we have used a proper code coverage task for our code. It looks like this:



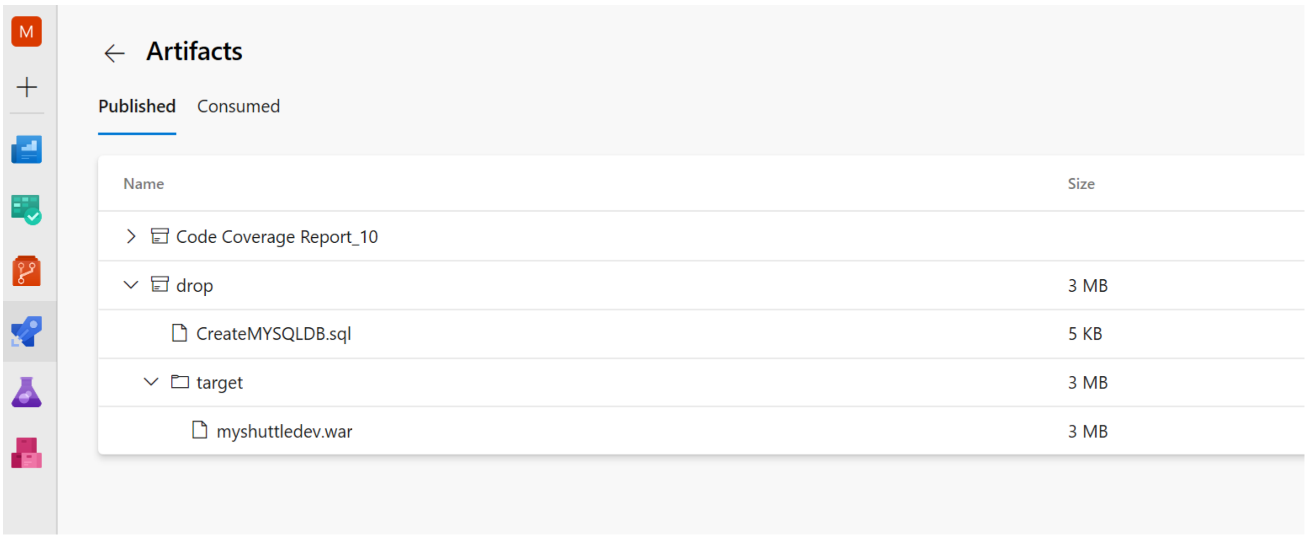
#20200321.1 undo environment: test Environment
on hittesharma.NodejsforAzure
Retained
Run new
Summary
Tests
Releases
Code Coverage

| | |
|---|---|
| **Covered lines:** | 7 |
| **Uncovered lines:** | 0 |
| **Coverable lines:** | 7 |
| **Total lines:** | 12 |
| **Line coverage:** | 100% (7 of 7) |
| **Covered branches:** | 0 |
| **Total branches:** | 0 |

Coverage

| Name | Covered | Uncovered | Coverable | Total | Line coverage | Branch coverage |
|---|---|---|---|---|---|---|
| **tests** | **7** | **0** | **7** | **12** | **100%** | |
| test.js | 7 | 0 | 7 | 12 | 100% | |

## Artifacts


Artifacts

Published Consumed

| Name | Size |
|---|---|
| Code Coverage Report_10 | |
| drop | 3 MB |
| CreateMYSQLDB.sql | 5 KB |
| target | 3 MB |
| myshuttledev.war | 3 MB |

*Figure 22: Artifacts after build run*

## Part 6: Release Pipeline

In this step you will create a release pipeline that will take the artifacts created in build pipeline and deploy. Our java application is a web application and you have created docker image for application. Now you will deploy an Azure web app for containers and Azure MySQL database server.

### Azure web app for container – For our shuttle app

- Go to azure portal and choose to create new resource.
- Search “Web app for containers” and hit enter.
- Provide details to create web app with container,

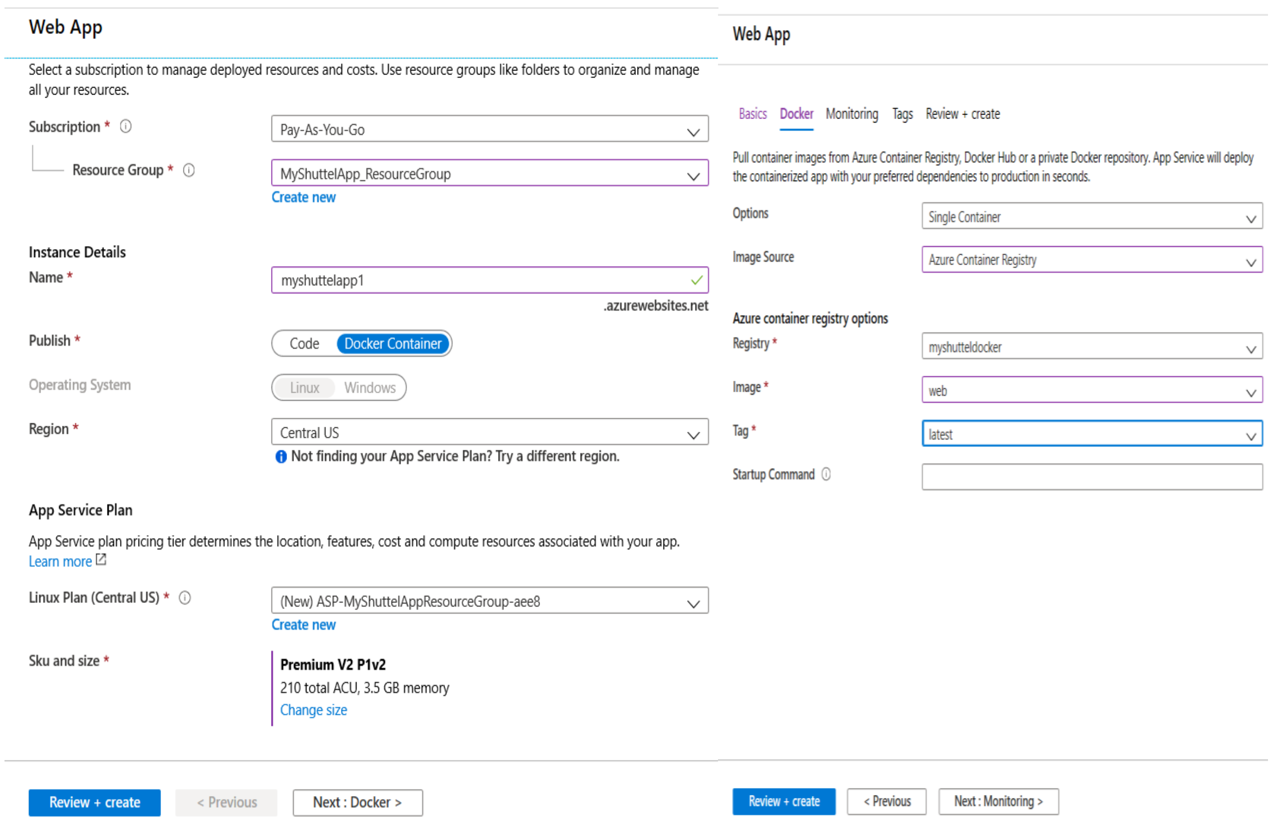
Azure Subscription name

Azure Resource Group name (should be linked with same azure subscription)

Name of the web app: **MyShuttelApp**

App service plan

In Docker tab, select image source, registry, image name, image tags


Web App

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *: Pay-As-You-Go

Resource Group *: MyShuttelApp_ResourceGroup

Create new

Instance Details

Name *: myshuttelapp1 .azurewebsites.net

Publish *: Code [✓ Docker Container]

Operating System: Linux Windows

Region *: Central US

Not finding your App Service Plan? Try a different region.

App Service Plan

App Service plan pricing tier determines the location, features, cost and compute resources associated with your app. Learn more

Linux Plan (Central US) *: (New) ASP-MyShuttelAppResourceGroup-aee8

Create new

Sku and size *: **Premium V2 P1v2** 210 total ACU, 3.5 GB memory Change size

Review + create | < Previous | Next : Docker >

Web App

Basics | Docker | Monitoring | Tags | Review + create

Pull container images from Azure Container Registry, Docker Hub or a private Docker repository. App Service will deploy the containerized app with your preferred dependencies to production in seconds.

Options: Single Container

Image Source: Azure Container Registry

Azure container registry options

Registry *: myshutteldocker

Image *: web

Tag *: latest

Startup Command:

Review + create | < Previous | Next : Monitoring >

*Figure 23: Azure Web app service creation*

Complete other details by pressing next and finally create the resource.

## Create Azure MySQL DB server

- Go to Azure portal and search “Azure Database for MySQL servers” and hit enter.
- Enter server details that you want to create as below

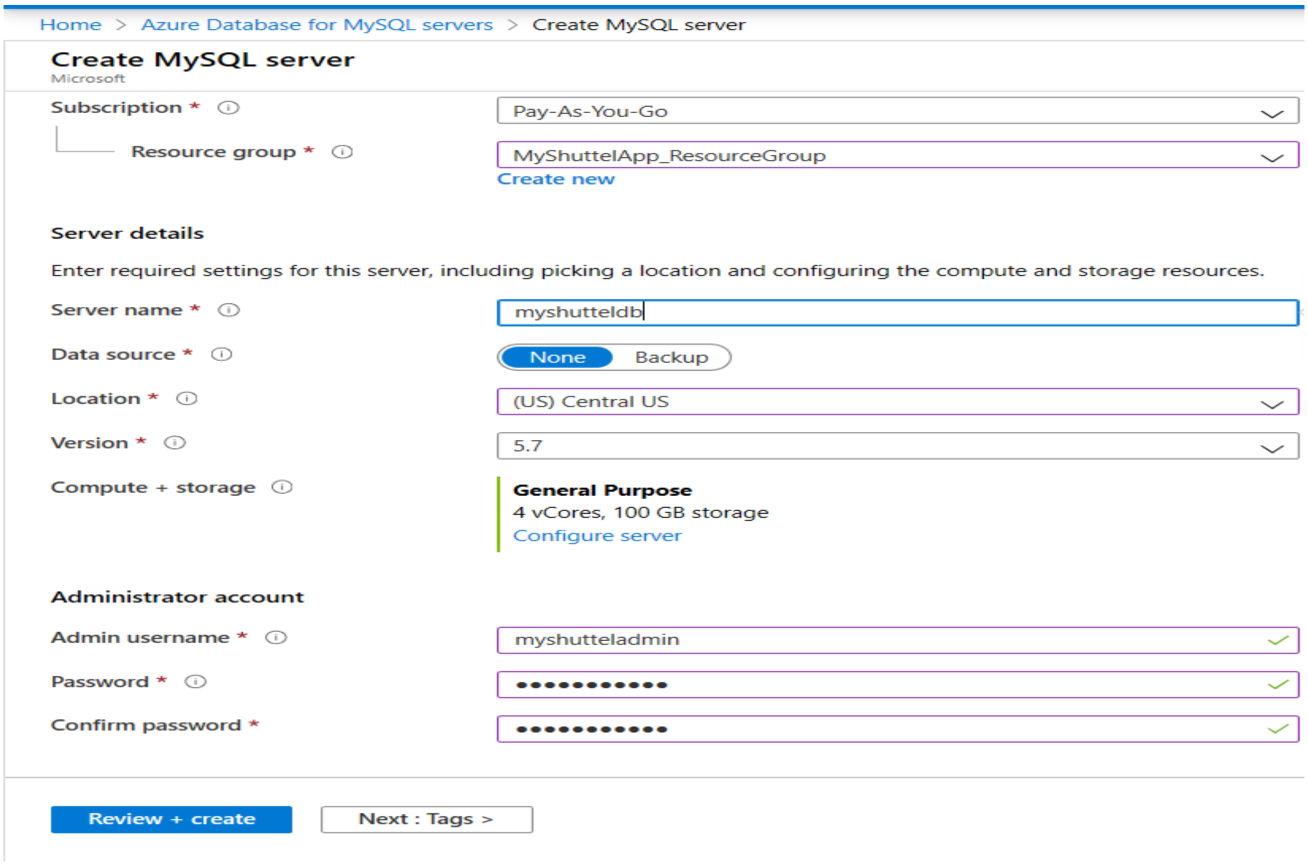
Subscription Name: {The subscription you are using for this tutorial}

Resource Group: {Where your other resources are present for this application}

Database Server Name: **myshutteldb**

Location, Version, Server configuration: {As per your deployment requirement}

Admin account details: username – **myshutteladmin**, password: {your_choice}


Home > Azure Database for MySQL servers > Create MySQL server

**Create MySQL server**
Microsoft

| | |
|---|---|
| Subscription * | Pay-As-You-Go |
| Resource group * | MyShuttelApp_ResourceGroup |
| | Create new |

**Server details**

Enter required settings for this server, including picking a location and configuring the compute and storage resources.

| | |
|---|---|
| Server name * | myshutteldb |
| Data source * | [✓ None] Backup |
| Location * | (US) Central US |
| Version * | 5.7 |
| Compute + storage | **General Purpose** 4 vCores, 100 GB storage Configure server |

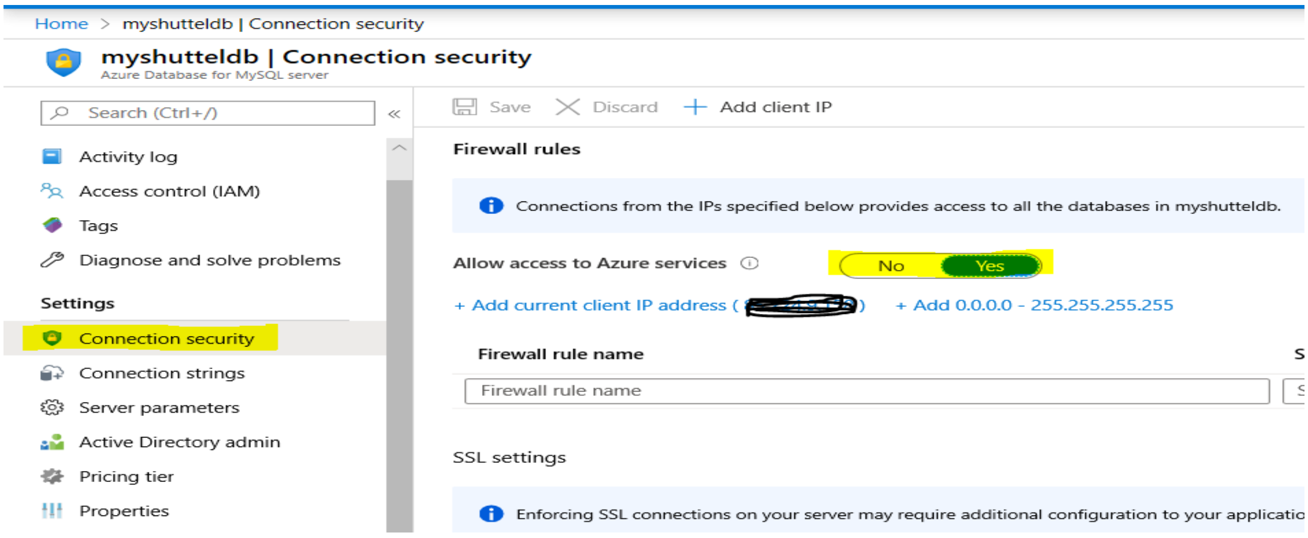
**Administrator account**

| | |
|---|---|
| Admin username * | myshutteladmin |
| Password * | ●●●●●●●●●●●● |
| Confirm password * | ●●●●●●●●●●●● |

Review + create | Next : Tags >

*Figure 24: Azure MySql server setup*

Once deployed, go to "myshutteldb" resource and change firewall rule to allow connection to database from other azure resource.


Home > myshutteldb | Connection security

**myshutteldb | Connection security**
Azure Database for MySQL server

**Firewall rules**

Connections from the IPs specified below provides access to all the databases in myshutteldb.

Allow access to Azure services: No [✓ Yes]

+ Add current client IP address ([redacted]) + Add 0.0.0.0 - 255.255.255.255

| Firewall rule name | S |
|---|---|
| Firewall rule name | S |

SSL settings

Enforcing SSL connections on your server may require additional configuration to your applicatio

*Figure 25: Enabling access to database*

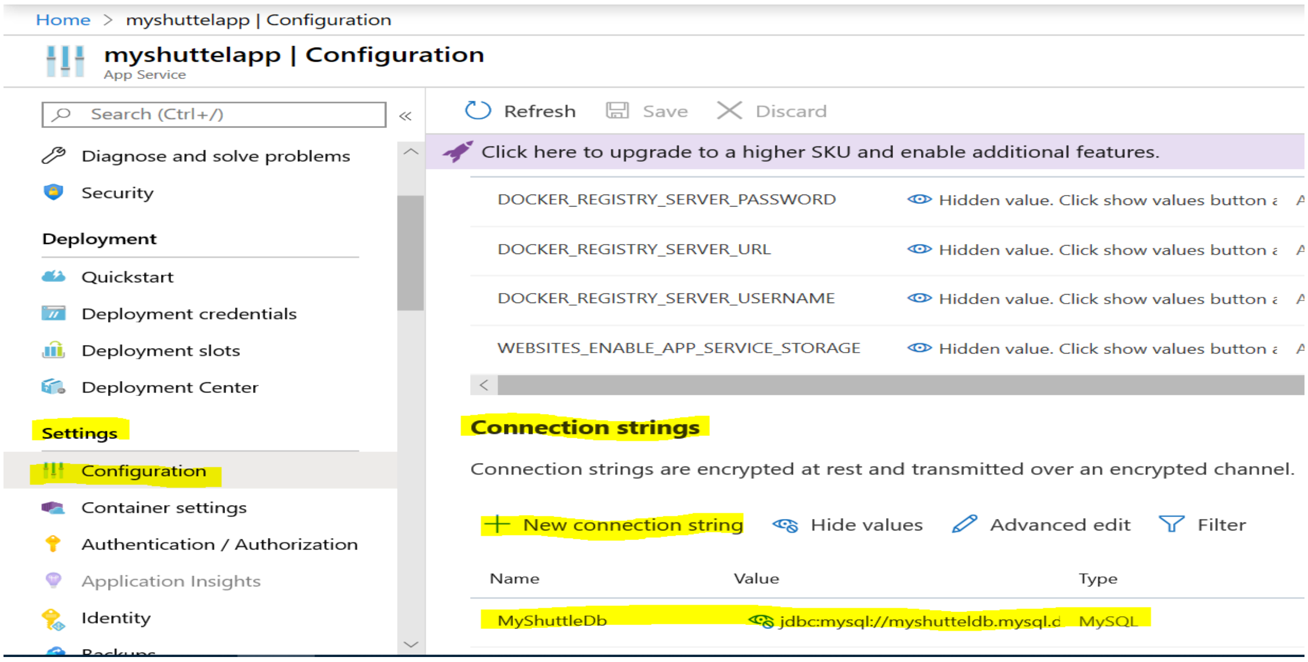
## Change connectionstring in web app

- Go to web app - "**myshuttelapp**" resource
- From left menu, open Configuration under Settings section and add a "New connection string" as below.

**Name**: **MyShuttelDB**

**Value**:
jdbc:mysql://myshutteldb.mysql.database.azure.com:3306/alm?useSSL=true&requireSSL=false&autoReconnect=true&user=myshutteladmin@myshutteldb&password={**Your_Password**}

Type: MySQL


Home > myshuttleapp | Configuration

myshuttleapp | Configuration
App Service

Refresh | Save | Discard

Click here to upgrade to a higher SKU and enable additional features.

| | |
|---|---|
| DOCKER_REGISTRY_SERVER_PASSWORD | Hidden value. Click show values button a |
| DOCKER_REGISTRY_SERVER_URL | Hidden value. Click show values button a |
| DOCKER_REGISTRY_SERVER_USERNAME | Hidden value. Click show values button a |
| WEBSITES_ENABLE_APP_SERVICE_STORAGE | Hidden value. Click show values button a |

Connection strings

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

New connection string | Hide values | Advanced edit | Filter

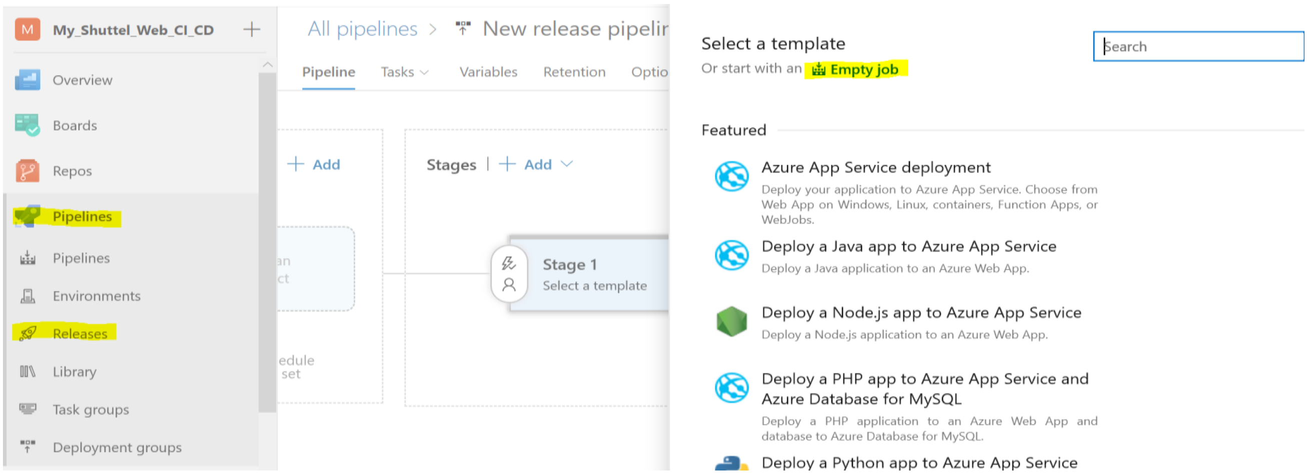
| Name | Value | Type |
|---|---|---|
| MyShuttleDb | jdbc:mysql://myshutteldb.mysql.d | MySQL |

*Figure 26: Adding database connection string for application*

## Release pipeline creation

Now all resources are set to create the release pipeline for your java app and finally deploy the artifacts created in build pipeline run.

To create release pipeline, go to **Pipelines** -> **Releases** in project menu and use **+ New** button. Start with **Empty job** from template selection.



Azure App Service deployment
Deploy your application to Azure App Service. Choose from Web App on Windows, Linux, containers, Function Apps, or WebJobs.

Deploy a Java app to Azure App Service
Deploy a Java application to an Azure Web App.

Deploy a Node.js app to Azure App Service
Deploy a Node.js application to an Azure Web App.

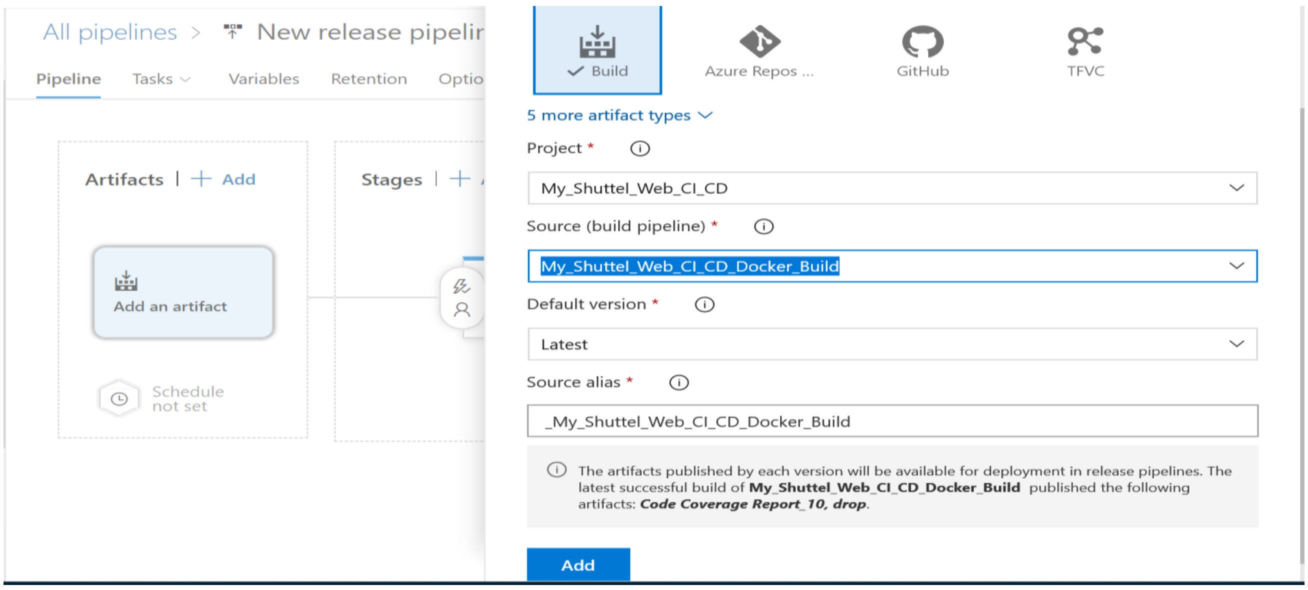
Deploy a PHP app to Azure App Service and Azure Database for MySQL
Deploy a PHP application to an Azure Web App and database to Azure Database for MySQL.

Deploy a Python app to Azure App Service

*Figure 27: Release pipeline jobs*

Now you have a screen with two section, "**Artifact**" and "**Stages**". In artifacts, you click add and give artifacts details. Here you should select artifact type as "Build" as you will deploy artifacts from your previous build pipeline.

Select other details as below and click **Add**.



Project * : My_Shuttel_Web_CI_CD

Source (build pipeline) * : My_Shuttel_Web_CI_CD_Docker_Build

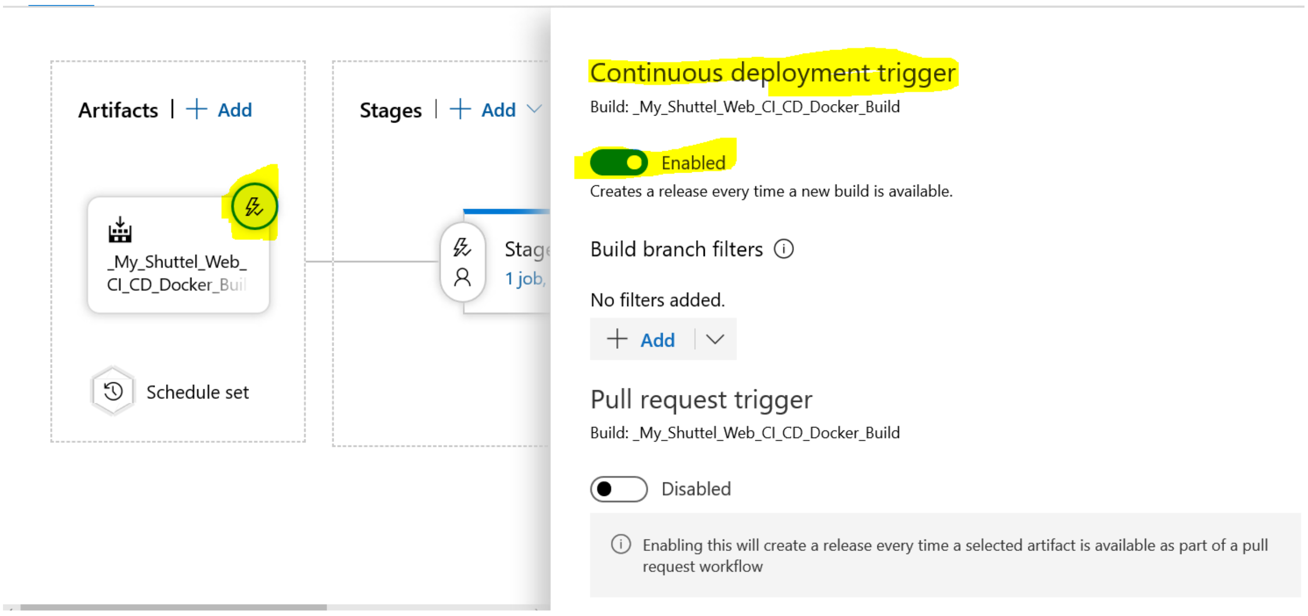
Default version * : Latest

Source alias * : _My_Shuttel_Web_CI_CD_Docker_Build

The artifacts published by each version will be available for deployment in release pipelines. The latest successful build of **My_Shuttel_Web_CI_CD_Docker_Build** published the following artifacts: ***Code Coverage Report_10, drop***.

*Figure 28: Setup artifacts for release pipeline*

To enable continuous delivery, click on lightning symbol in "Artifact" section and enable "Continuous Deployment Trigger".



Continuous deployment trigger

Build: _My_Shuttel_Web_CI_CD_Docker_Build

Enabled

Creates a release every time a new build is available.

Build branch filters

No filters added.

Pull request trigger

Build: _My_Shuttel_Web_CI_CD_Docker_Build

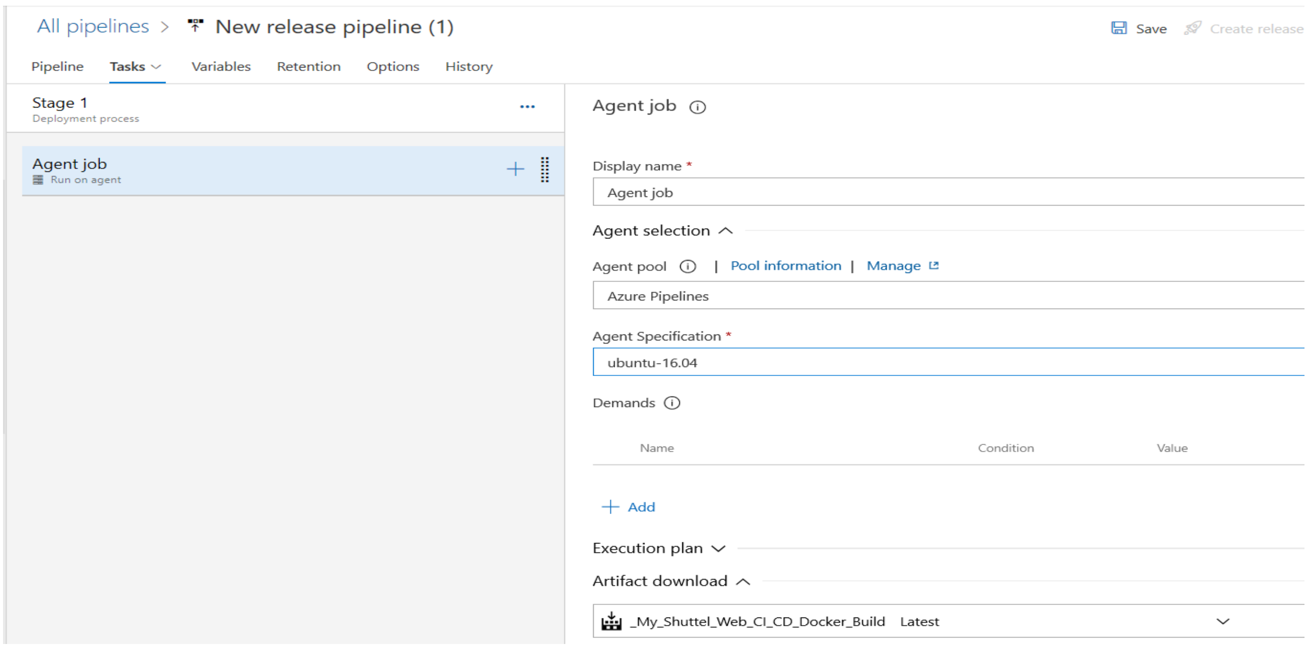
Disabled

Enabling this will create a release every time a selected artifact is available as part of a pull request workflow

*Figure 29: Enabling continuous deployment of release pipeline*

Next step is to configure release stages. Here you can setup different stages such as staging, Test, production and configure each stage with deployment tasks, what to deploy and where to deploy.

To setup deployment tasks, click on **task, job** link in stages section and configure Agent Job that where all deployment task will run. Set details as below


All pipelines > New release pipeline (1)

Save | Create release

Pipeline | Tasks | Variables | Retention | Options | History

Stage 1
Deployment process

Agent job
Run on agent

Agent job

Display name *
Agent job

Agent selection

Agent pool | Pool information | Manage

Azure Pipelines

Agent Specification *
ubuntu-16.04

Demands

| Name | Condition | Value |
|---|---|---|

+ Add

Execution plan

Artifact download

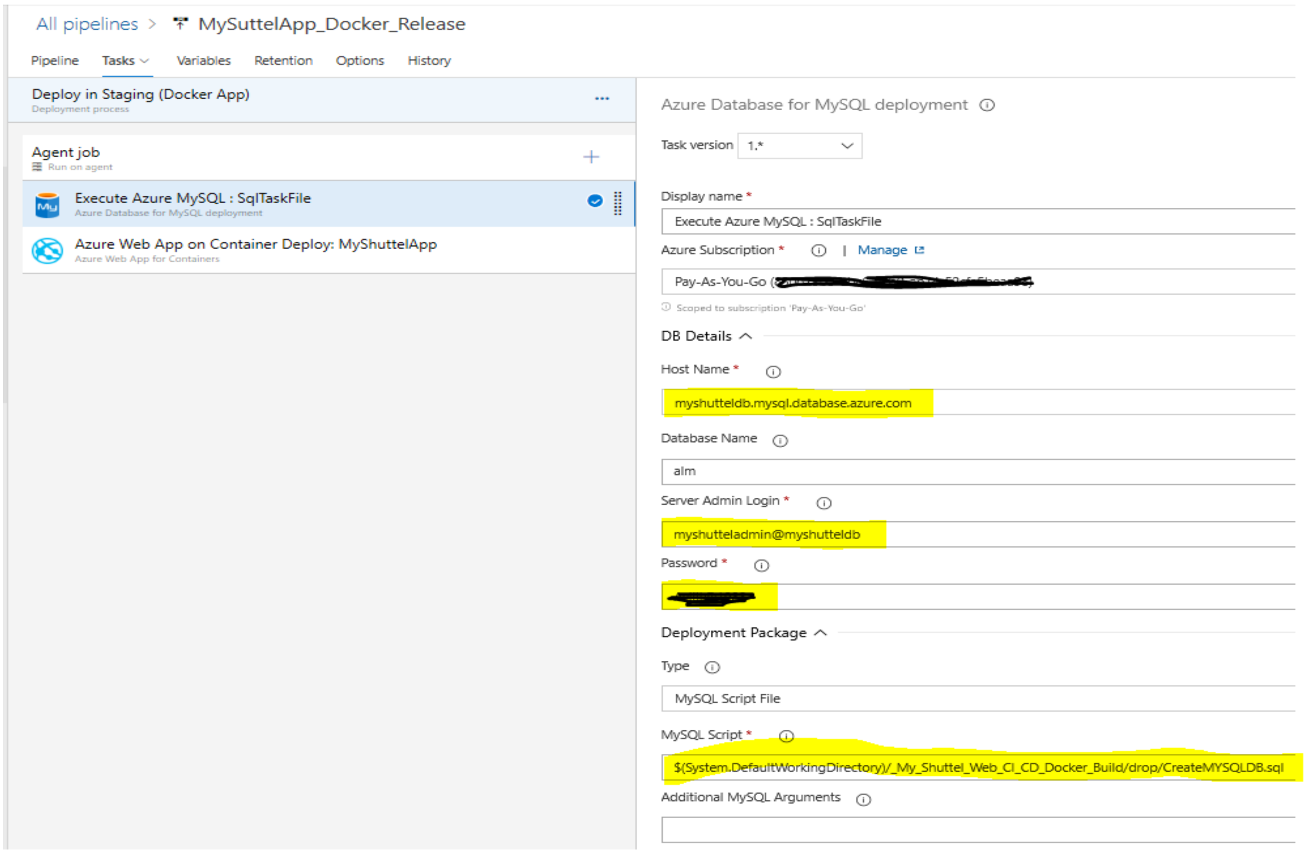
_My_Shuttel_Web_CI_CD_Docker_Build Latest

*Figure 30: Release agent configuration*

Click **+** sign on "Agent Job" to add new task. You need two tasks to complete the deployment process.

1. **Azure Database for MySQL deployment**: This task will deploy the database script on the "**myshutteldb**" database server to create a database named as "**alm**" and database schema with users and their expenses table.

Configure the task with details as below


All pipelines > MySuttelApp_Docker_Release

Pipeline | Tasks | Variables | Retention | Options | History

Deploy in Staging (Docker App)
Deployment process

Agent job
Run on agent

Execute Azure MySQL : SqlTaskFile
Azure Database for MySQL deployment

Azure Web App on Container Deploy: MyShuttelApp
Azure Web App for Containers

Azure Database for MySQL deployment

Task version 1.*

Display name *
Execute Azure MySQL : SqlTaskFile

Azure Subscription * | Manage
Pay-As-You-Go ([redacted])
Scoped to subscription 'Pay-As-You-Go'

DB Details

Host Name *
myshutteldb.mysql.database.azure.com

Database Name
alm

Server Admin Login *
myshutteladmin@myshutteldb

Password *
[redacted]

Deployment Package

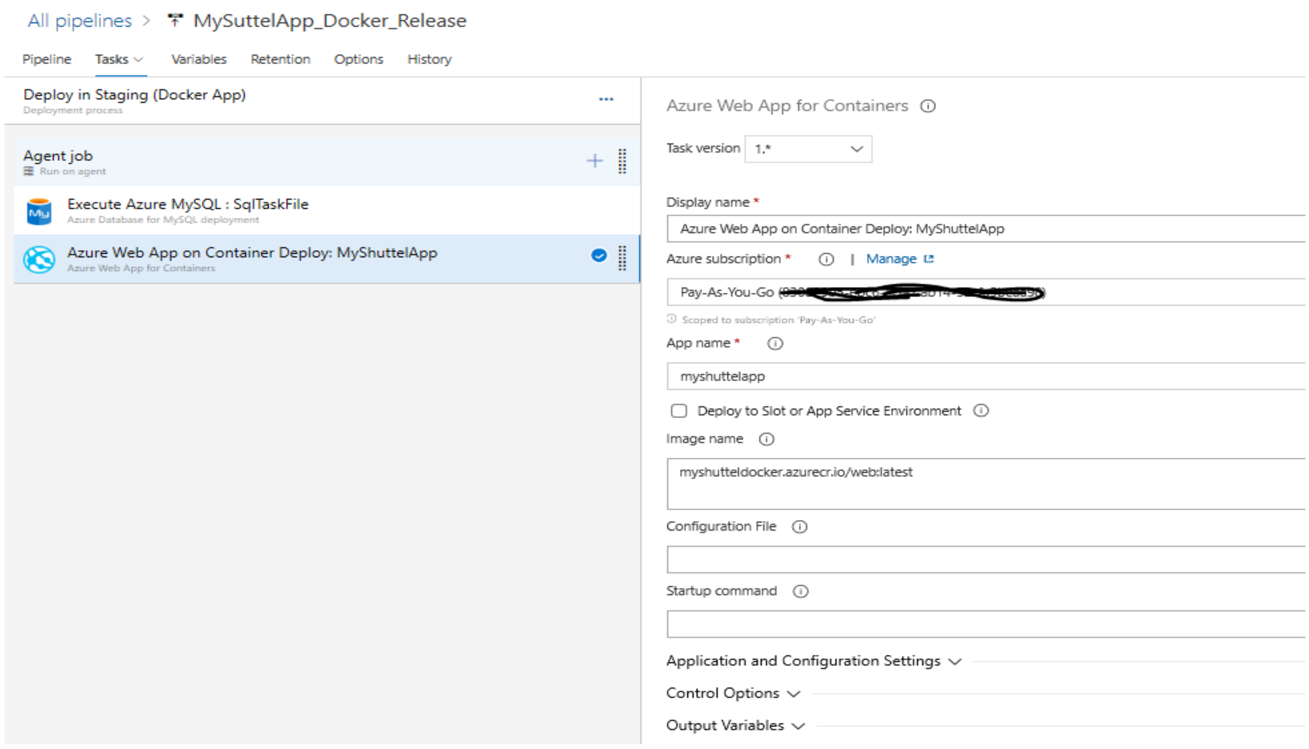
Type
MySQL Script File

MySQL Script *
$(System.DefaultWorkingDirectory)/_My_Shuttel_Web_CI_CD_Docker_Build/drop/CreateMYSQLDB.sql

Additional MySQL Arguments

*Figure 31: Database deployment task configuration*

2. **Azure Web App for Containers:** This task is to deploy the MyShuttelApp docker image into "**MyShuttelApp**" azure web app. Configure the task as shown below.


All pipelines > MySuttelApp_Docker_Release

Deploy in Staging (Docker App)
Deployment process

Agent job
Run on agent

Execute Azure MySQL : SqlTaskFile
Azure Database for MySQL deployment

Azure Web App on Container Deploy: MyShuttelApp
Azure Web App for Containers

Azure Web App for Containers

Task version: 1.*

Display name *: Azure Web App on Container Deploy: MyShuttelApp

Azure subscription * | Manage: Pay-As-You-Go [redacted]

Scoped to subscription 'Pay-As-You-Go'

App name *: myshuttelapp

☐ Deploy to Slot or App Service Environment

Image name: myshutteldocker.azurecr.io/web:latest

Configuration File:

Startup command:

*Figure 32: Web app deployment task configuration*

**Please Note**: in Image name, fully qualified name is required which you can find from your Azure Container registry.

Once task setup is completed, save the task using **Save** button.

<u>FUN FACT</u>

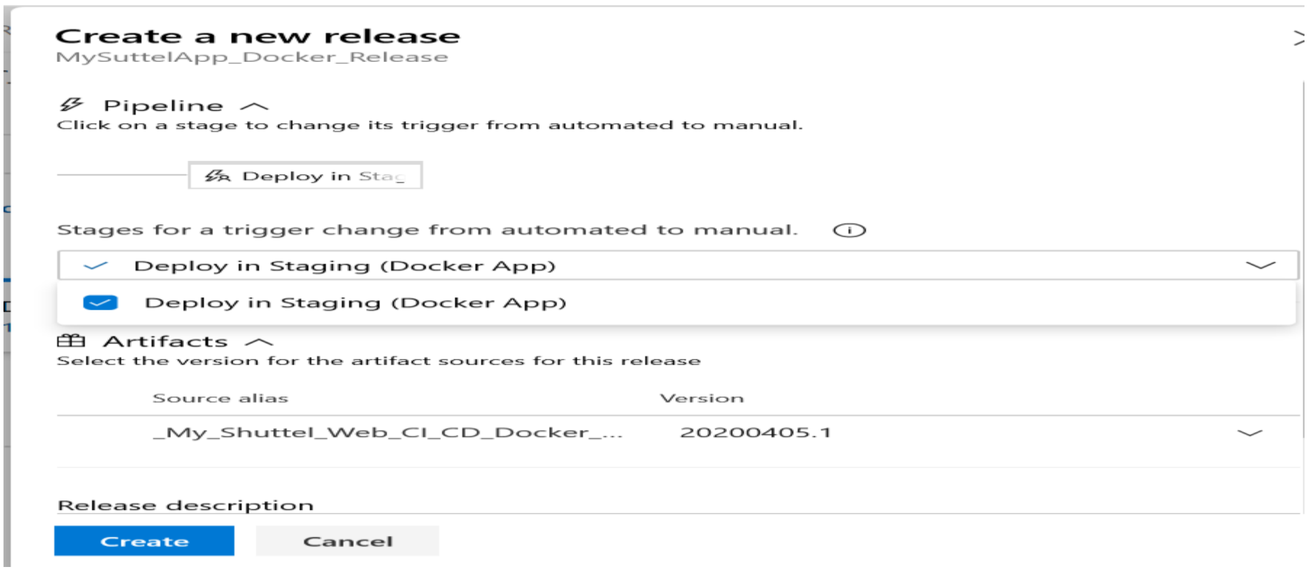
Why is pipeline named so?

*Pipeline is called so because it flows in the same fashion as a tubular water pipe flowing water. We can also say it performs compile, build and deploy one after the other where human generally see the input and the result at the two ends only, rest is all automated.*

## Part 7: Run Release pipeline

You can trigger release pipeline by creating a new build in build pipeline or trigger it manually. Here you will create a new release manually.

Click on "Create Release" button on release pipeline page and select the stage name that you want to run.



Create a new release
MySuttelApp_Docker_Release

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

Deploy in Stag

Stages for a trigger change from automated to manual.
Deploy in Staging (Docker App)
Deploy in Staging (Docker App)

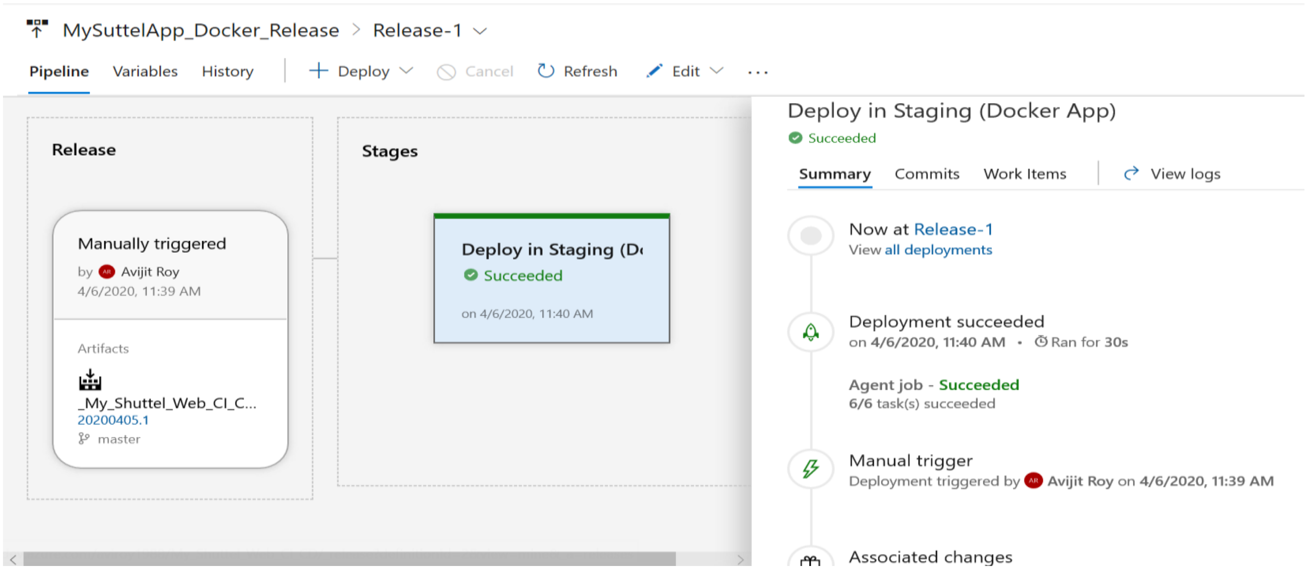
Artifacts
Select the version for the artifact sources for this release

| Source alias | Version |
|---|---|
| _My_Shuttel_Web_CI_CD_Docker_... | 20200405.1 |

Release description

*Figure 33: Adding release manually*

Once release is saved, it will trigger the release and you can see the status in Releases section of release pipeline.



MySuttelApp_Docker_Release > Release-1

Release
Manually triggered
by Avijit Roy
4/6/2020, 11:39 AM
Artifacts
_My_Shuttel_Web_CI_C...
20200405.1
master

Stages
Deploy in Staging (D
Succeeded
on 4/6/2020, 11:40 AM

Deploy in Staging (Docker App)
Succeeded

Now at Release-1
View all deployments

Deployment succeeded
on 4/6/2020, 11:40 AM • Ran for 30s

Agent job - Succeeded
6/6 task(s) succeeded

Manual trigger
Deployment triggered by Avijit Roy on 4/6/2020, 11:39 AM

Associated changes

*Figure 34: Running release pipeline*

Similarly, we can clone the already created release to be released into production environment. That means once the release is successful in pre-prod a.k.a. Staging Environment, we should be confident enough to push to production with prerequisite pre and post deployment approvals (that can be done in Azure, as explained before). This makes the complete CI/ CD pipeline under one roof of Azure, right from the requirement gathering in Agile board to the production. Azure made it simple and require not exhaustive expertise in every tool and technology.

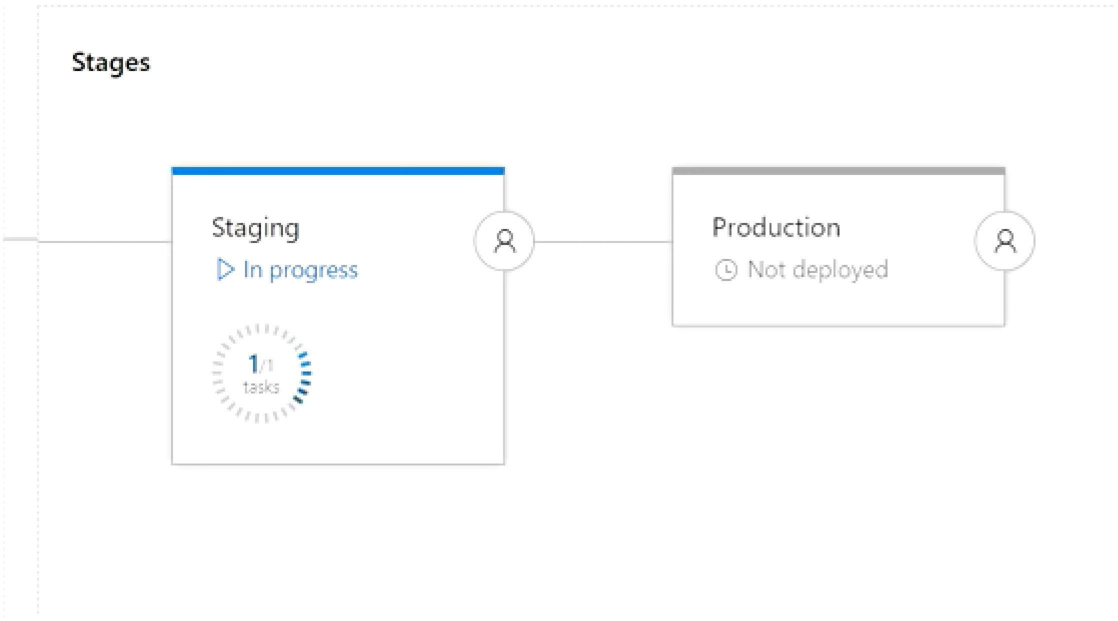
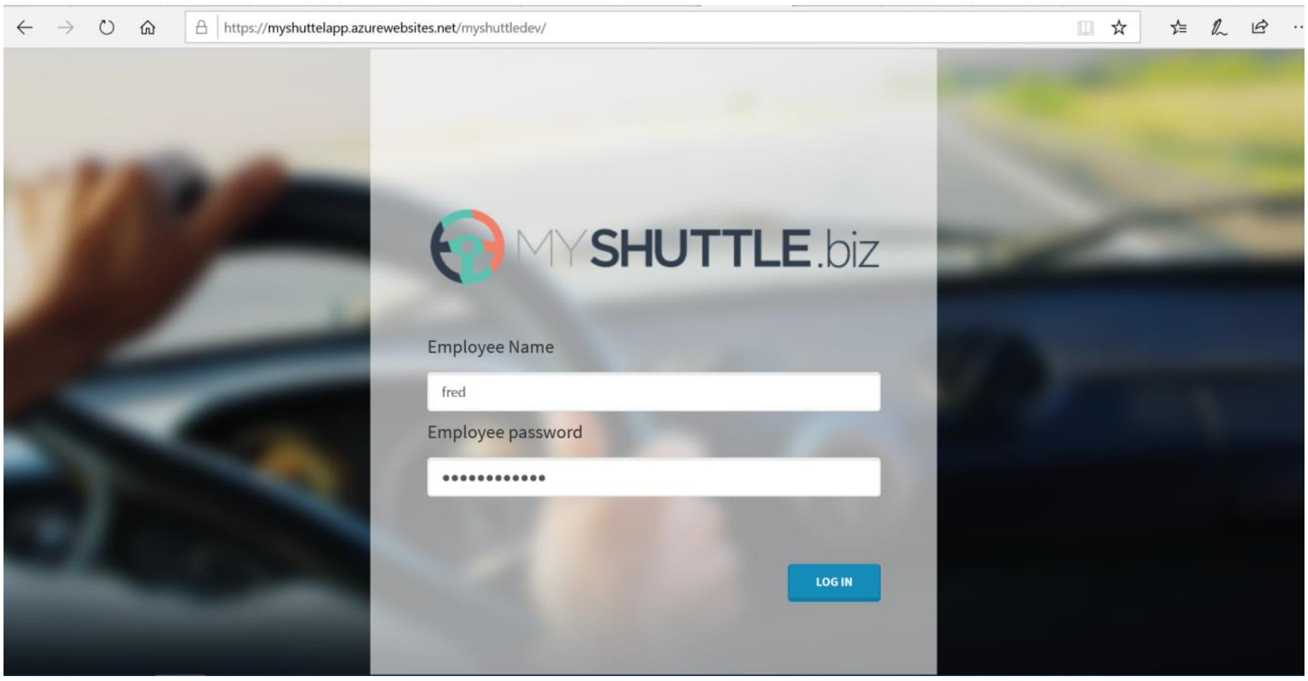
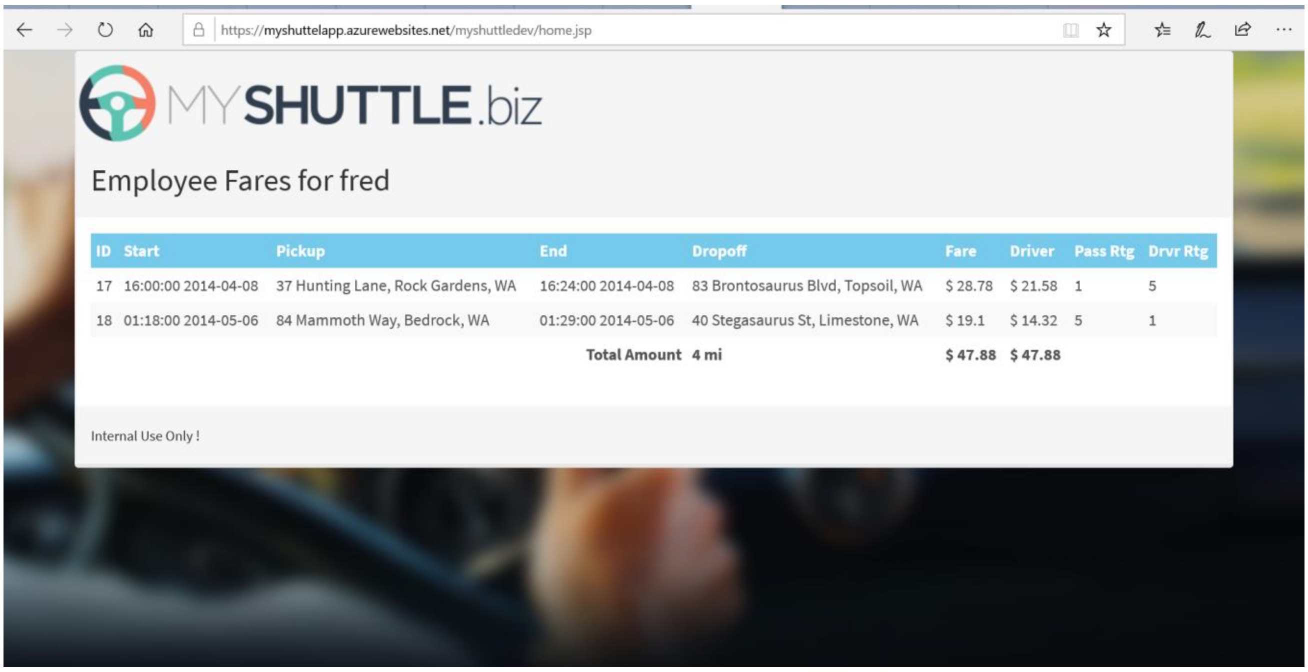


*Figure 35: Different stages of deployment*

Finally, MyShuttel app is deployed and ready to browse. You can go to MyshutteApp, Azure web app resource and use browse button or use following link to browse the tomcat default page.

https://myshuttelapp.azurewebsites.net/myshuttledev/




https://myshuttelapp.azurewebsites.net/myshuttledev/home.jsp

Employee Fares for fred

| ID | Start | Pickup | End | Dropoff | Fare | Driver | Pass Rtg | Drvr Rtg |
|---|---|---|---|---|---|---|---|---|
| 17 | 16:00:00 2014-04-08 | 37 Hunting Lane, Rock Gardens, WA | 16:24:00 2014-04-08 | 83 Brontosaurus Blvd, Topsoil, WA | $ 28.78 | $ 21.58 | 1 | 5 |
| 18 | 01:18:00 2014-05-06 | 84 Mammoth Way, Bedrock, WA | 01:29:00 2014-05-06 | 40 Stegasaurus St, Limestone, WA | $ 19.1 | $ 14.32 | 5 | 1 |
| | | | Total Amount | 4 mi | $ 47.88 | $ 47.88 | | |

Internal Use Only !

*Figure 36: Browsing application after deployment*

**In this demo** we presented the overall flow of Agile implementation of a project it is close to one that we practice in an industry. It gives clear idea of end to end flow starting from project planning to the deployment in Agile methodology.

References:

https://docs.microsoft.com/en-us/azure/devops-project/azure-devops-project-java?toc=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fdevops-project%2Ftoc.json&bc=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fbread%2Ftoc.json

https://github.com/hsachinraj/GitHub-AzureDevOps