

# LAB GUIDE

## Lab: Deploying a Docker based web application to Azure App Service

### Learning Objectives

- How to build custom Docker images using Azure DevOps Hosted Linux agent
- How to push and store the Docker images in a private repository
- How to Deploy and run the images inside the Docker Containers

### Pre-requisites

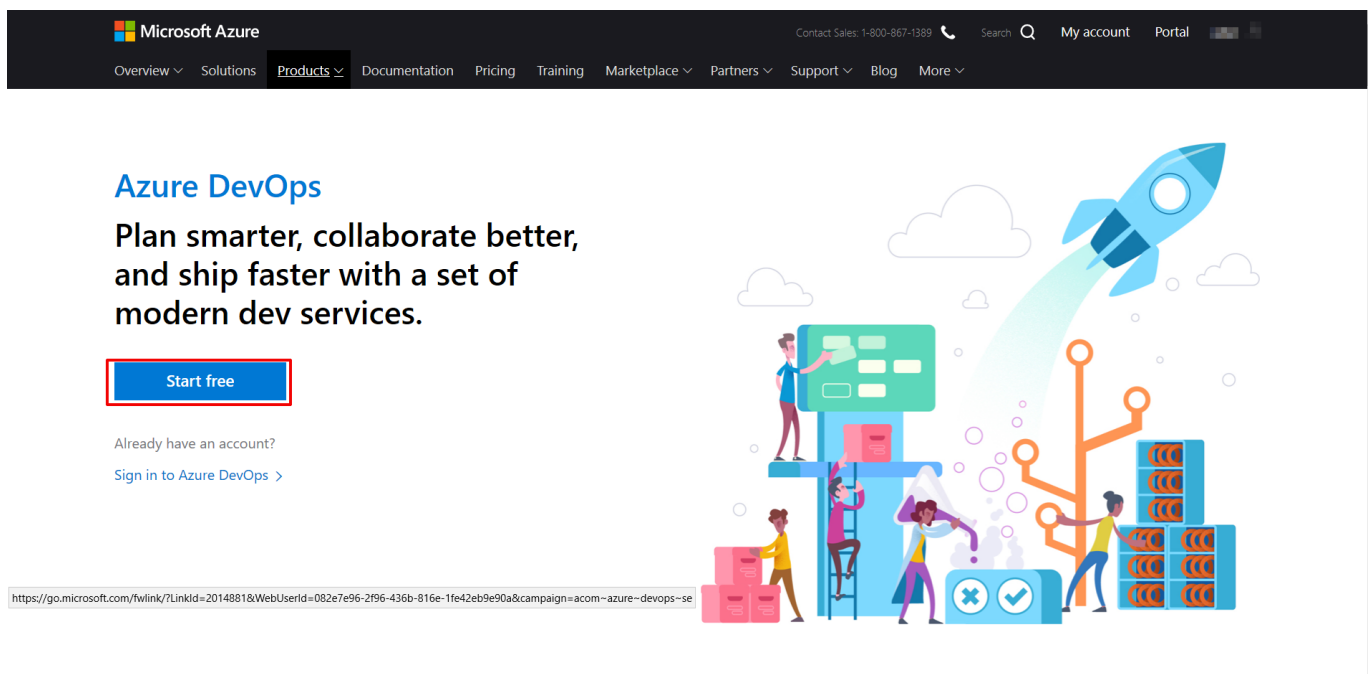
- Microsoft Azure Account: You'll need a valid and active Azure account for the Azure labs.
- You'll need an Azure DevOps account.

### Length

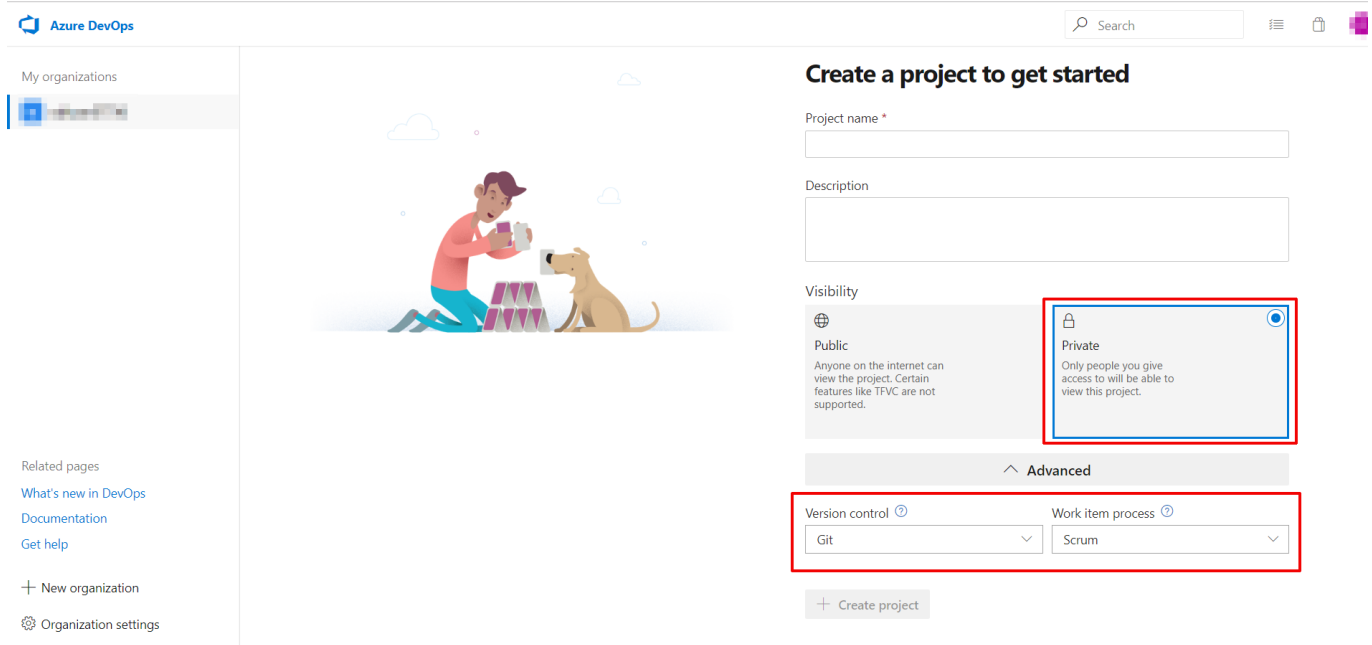
40 minutes

## Exercise 1: Create a new project in Azure DevOps

1. Enter in [Azure DevOps](#) and log in clicking in **Start free** button



2. Set the **Project name** you want
3. In **visibility** select **Private**
4. Click on **Advanced**, in **Version control** select **GIT** and **Work item process** select Scrum then click on **+ Create project**



Azure DevOps

My organizations

Related pages

- What's new in DevOps
- Documentation
- Get help

+ New organization

Organization settings

### 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.

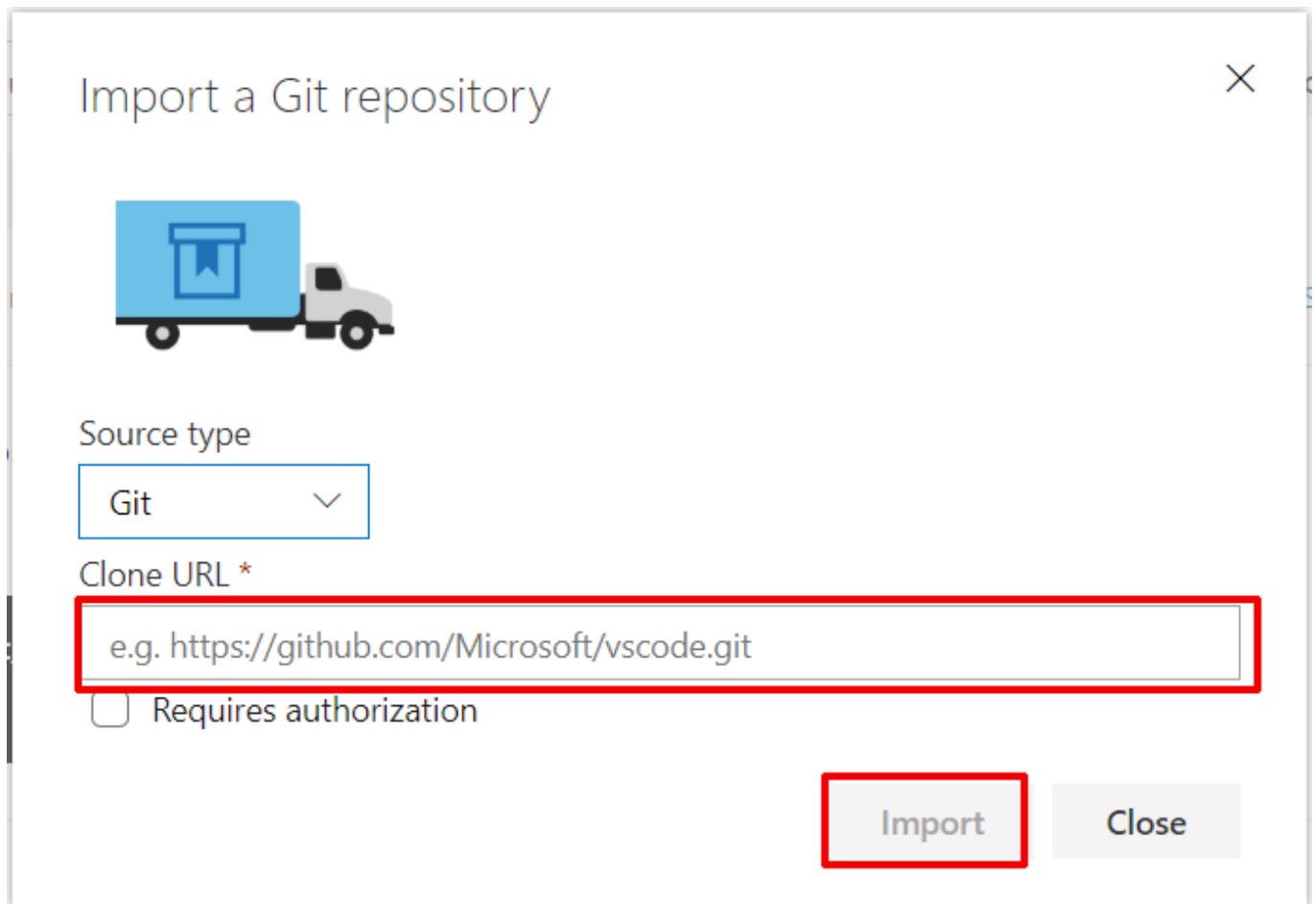
Advanced

Version control ?  
Git


Work item process ?  
Scrum

+ Create project

5. Once your project has been created click on **Repos** option
6. In Repos page you'll see many options to add some code, click on **Import** from **Or import a repository** option
7. in **Clone URL** option put this URL then click on import <https://github.com/MSTecs/Azure-DevDay-lab4-demoProject.git>



## Import a Git repository



Source type

Git

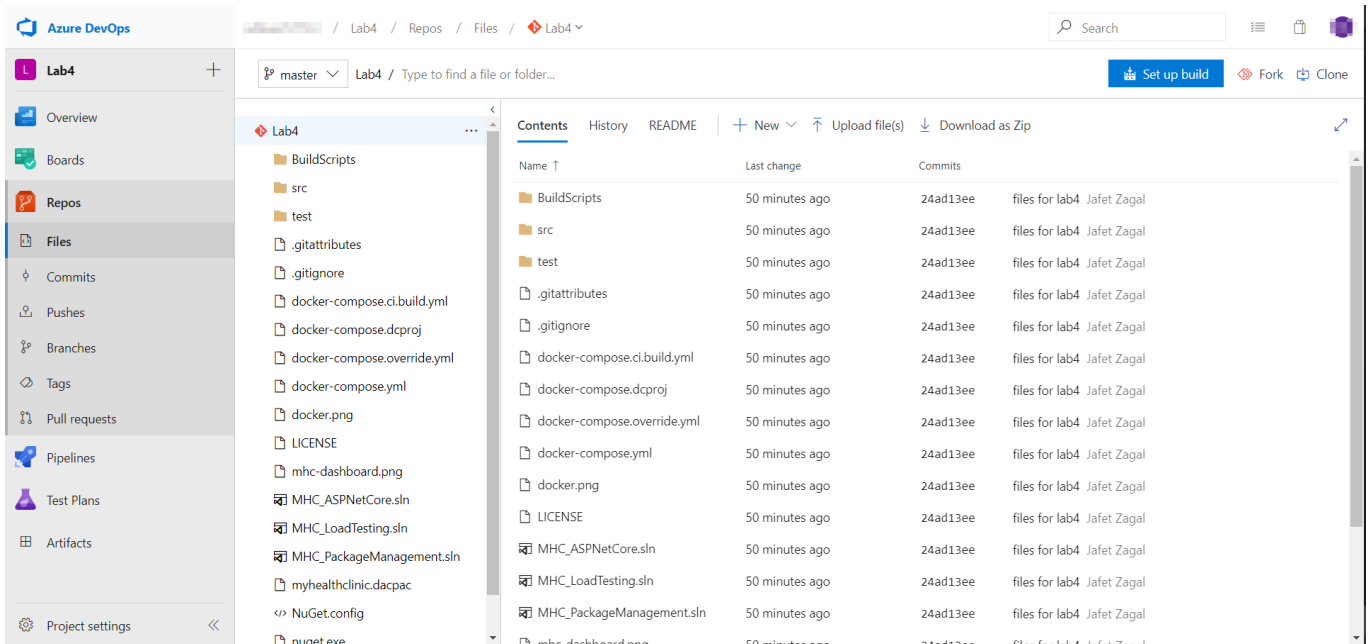
Clone URL \*

e.g. <https://github.com/Microsoft/vscode.git>

☐ Requires authorization

Import Close

8. Now if you click again on files you will see the code in your page



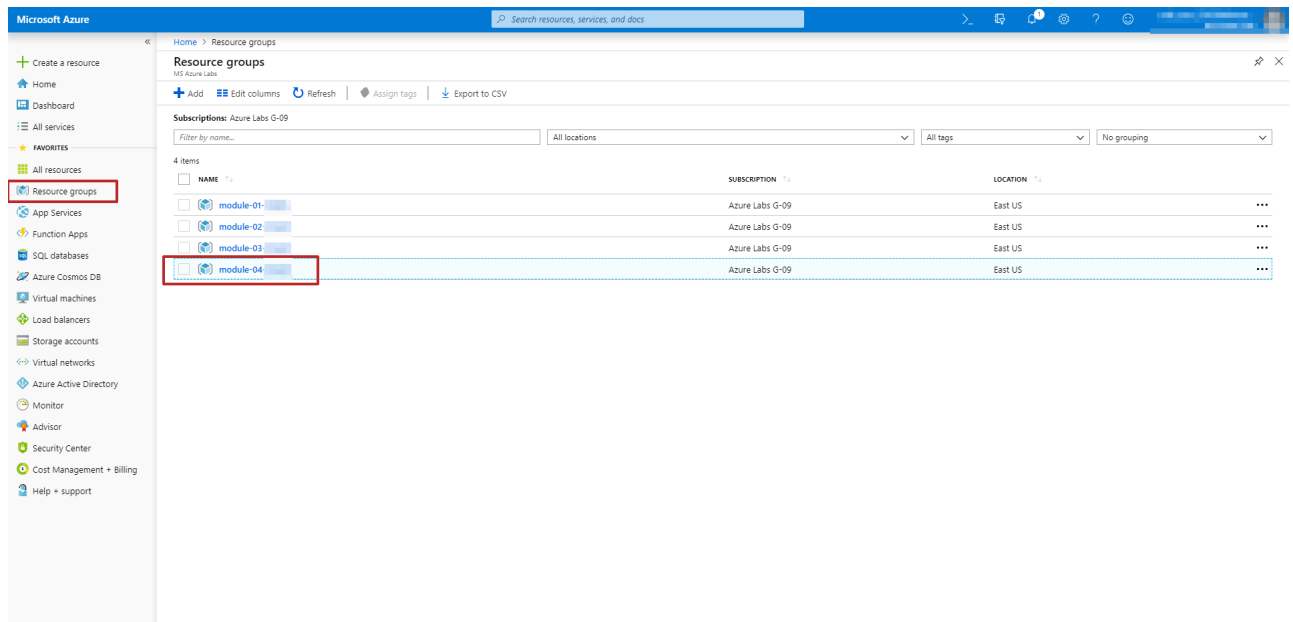
## Exercise 2: Configure Continuous Integration

### Task 1: Configure your basic Pipeline DevOps

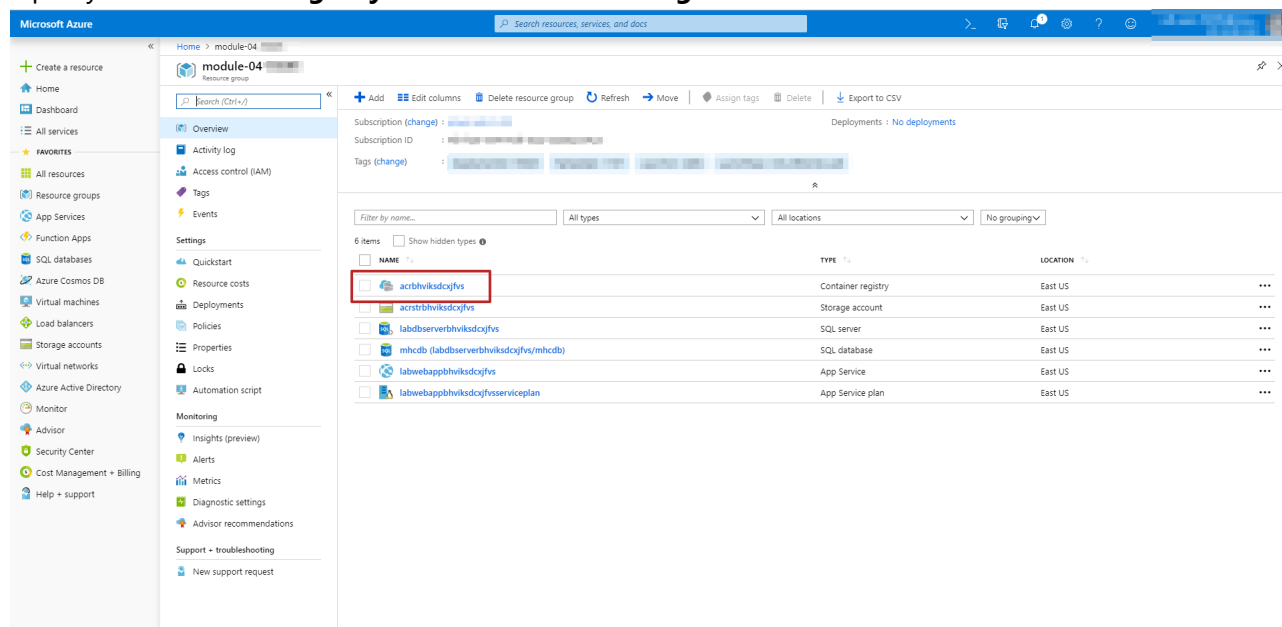
1. Log in portal Azure clicking here



2. Go to resources groups and select the resource named module-04-xxxxx Where xxxxx is the number obtained for the lab

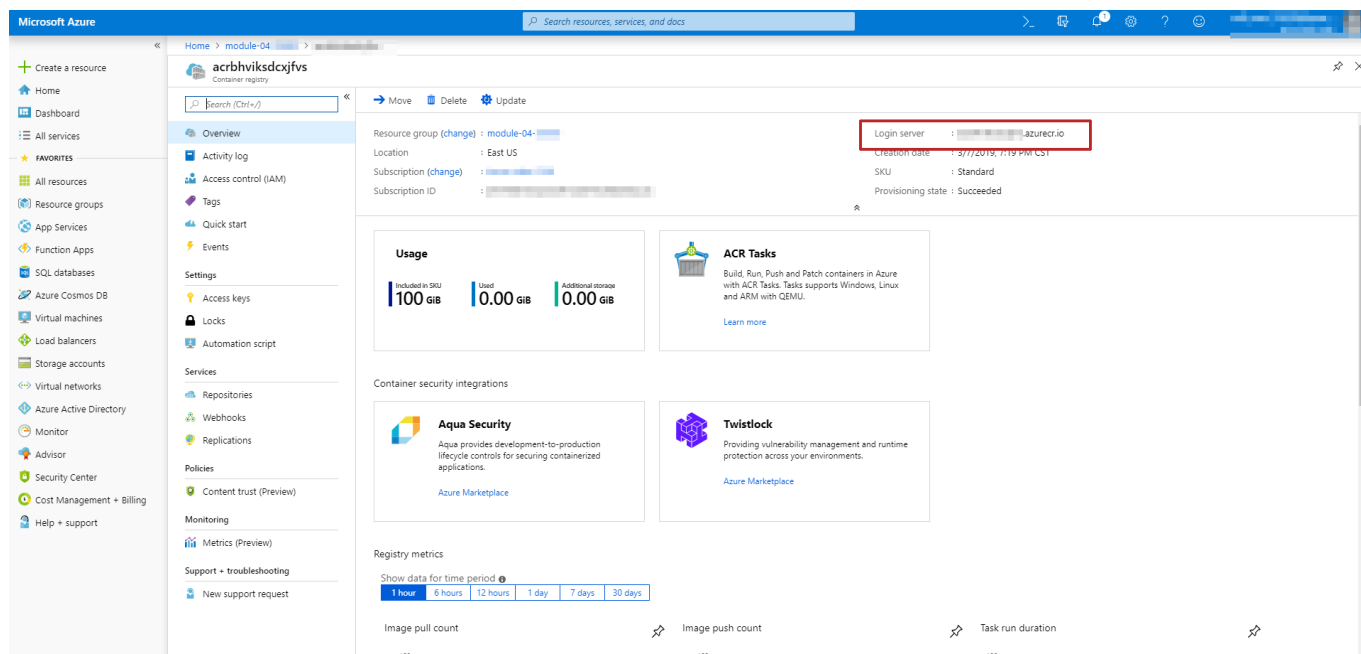


### 3. Open your **container registry** and take note of the **login server**



The screenshot shows the Microsoft Azure portal interface. The left sidebar contains navigation links for 'Create a resource', 'Home', 'Dashboard', 'All services', and 'FAVORITES'. The main area displays the 'module-04' resource group. The 'Overview' tab is active, showing a list of resources. The 'acrshvskdcjvjs' container registry is highlighted with a red box. The table below lists the resources in the group:

NAME	TYPE	LOCATION
acrshvskdcjvjs	Container registry	East US
acrshvskdcjvjs	Storage account	East US
labdbserverbhvskdcjvjs	SQL server	East US
mhcdb (labdbserverbhvskdcjvjs/mhcdb)	SQL database	East US
labwebappbhvskdcjvjs	App Service	East US
labwebappbhvskdcjvjs/serviceplan	App Service plan	East US



The screenshot shows the Microsoft Azure portal interface for the 'acrshvskdcjvjs' container registry. The 'Overview' tab is active, displaying various metrics and tasks. The 'Login server' is highlighted with a red box. The table below lists the metrics for the registry:

Metric	Value
Used	100 GiB
Additional storage	0.00 GiB

The 'ACR Tasks' section shows a task named 'Login server' with a status of 'Succeeded'. The 'Container security integrations' section lists 'Aqua Security' and 'Twistlock' as integrated services.

### 4. Go back to your resources open your **SQL database** and take note of the **Server name**

Microsoft Azure portal showing the 'module-04' resource group. The 'Overview' tab is selected, displaying a list of resources. The resource 'mhcdb (labdbserverbhvikscjfv/mhcdb)' is highlighted with a red box.

NAME	TYPE	LOCATION
acrbhvikscjfv	Container registry	East US
acrstrbhvikscjfv	Storage account	East US
labdbserverbhvikscjfv	SQL server	East US
<b>mhcdb (labdbserverbhvikscjfv/mhcdb)</b>	SQL database	East US
labwebappbhvikscjfv	App Service	East US
labwebappbhvikscjfvserviceplan	App Service plan	East US

Microsoft Azure portal showing the 'mhcdb' SQL database. The 'Overview' tab is selected, displaying details about the database. The 'Server name' field is highlighted with a red box.

Resource group (change): module-04

Status: Online

Location: East US

Subscription (change):

Tags (change):

Server name: **labdbserverbhvikscjfv.database.windows.net**

Elastic pool: No elastic pool

Connection strings: Show database connection strings

Pricing tier: Basic

Oldest restore point: 2019-03-08 01:32 UTC

Resource utilization (mhcdb)

Database data storage

Used space: 4 MB

Allocated space: 16 MB

Max size: 1 GB

0.39% USED SPACE

Notifications (0)

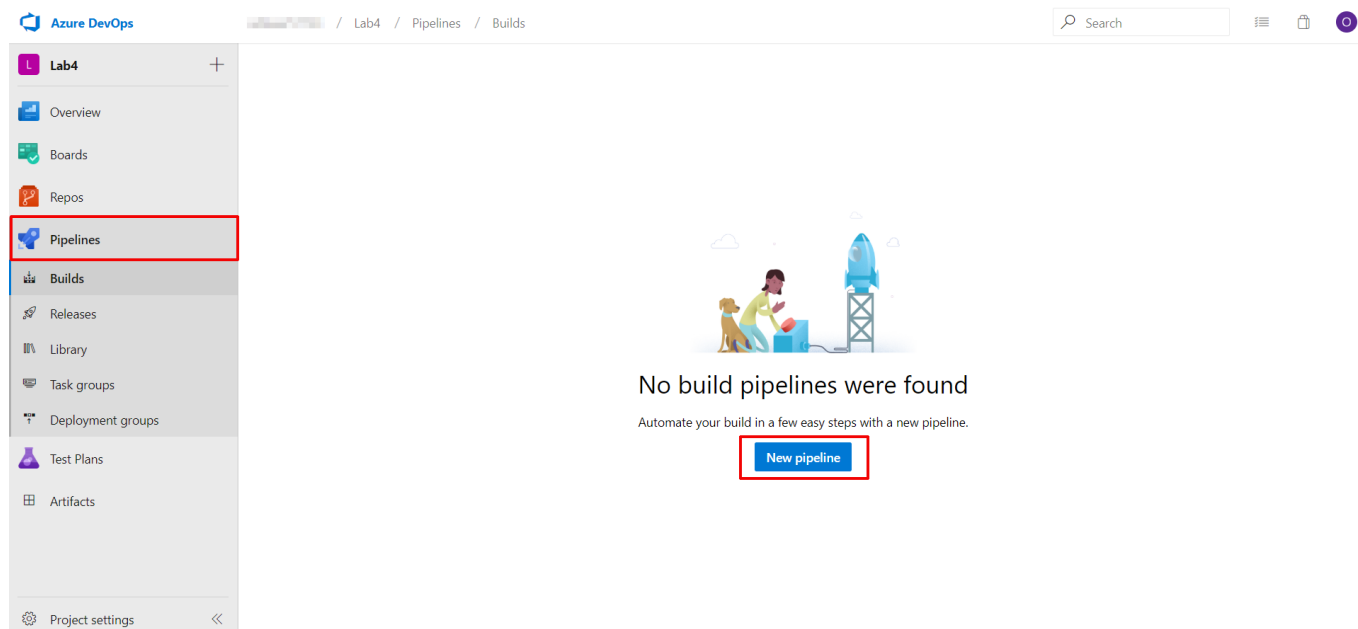
Database features (6)

Transparent data encryption: CONFIGURED

Advanced Data Security: NOT CONFIGURED

5. Return to [Azure DevOps](#)

6. Navigate to the **Builds** option under the **Pipelines** tab and select new pipeline



## 7. Select **Use the classic editor** option

Connect      Select      Configure      Create pipeline

New pipeline

## Where is your code?



**Azure Repos Git** YAML

Free private Git repositories, pull requests, and code search



**Bitbucket Cloud**

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 Internet-facing Git repository




**Subversion**

Centralized version control by Apache

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


## 8. Select **Azure Repos Git** and select your project and click on Continue





Select your repository


Tell us where your sources are.  
You can customize how to get these sources from the repository later.


Select a source


 Azure Repos Git

 GitHub

 GitHub Enterprise Server

 Subversion

 Bitbucket Cloud

 External Git

Team project

Lab4

Repository


Lab4

Default branch for manual and scheduled builds

master

Continue


## 9. Select the **Empty job** option



Choose a template


Choose a template that builds your kind of app.  
Don't worry if it's not an exact match;  
you can add and customize the tasks later.

Select a template


Or start with an  **Empty job**


Search


Configuration as code


 **YAML**  
Looking for a better experience to configure your pipelines using YAML files? Try the new YAML pipeline creation experience. [Learn more](#)


Featured


 **.NET Desktop**  
Build and test a .NET or Windows classic desktop solution.

 **Android**  
Build, test, sign, and align an Android APK.

 **ASP.NET**  
Build and test an ASP.NET web application.

 **Azure Web App for ASP.NET**  
Build, package, test, and deploy an ASP.NET Azure Web App.

 **Docker container**  
Build a Docker image and push it to a container registry.

 **Maven**  
Build and test a Java project with Apache Maven.

10. Click on the **Agent job 1** and change the **display name** to Docker

11. On **Execution Plan** Section set 1 on **Job cancel timeout** input

Lab4-CI

Tasks Variables Triggers Options Retention History Save & queue Discard Summary Queue

Pipeline Build pipeline

Get sources Lab4 master

Agent job 1 Run on agent

Display name \* Agent job 1

Agent selection ^

Agent pool | Manage <inherit from pipeline>

Demands

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

+ Add

Execution plan ^

Parallelism

☒ None ☐ Multi-configuration ☐ Multi-agent

Timeout \* 0

Job cancel timeout \* 0

Your configuration should look like this

Lab4-CI

Tasks Variables Triggers Options Retention History Save & queue Discard Summary Queue

Pipeline Build pipeline

Get sources Lab4 master

Docker Run on agent

Agent job

Display name \* Docker

Agent selection ^

Agent pool | Manage <inherit from pipeline>

Demands

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

+ Add

Execution plan ^

Parallelism

☒ None ☐ Multi-configuration ☐ Multi-agent

Timeout \* 0

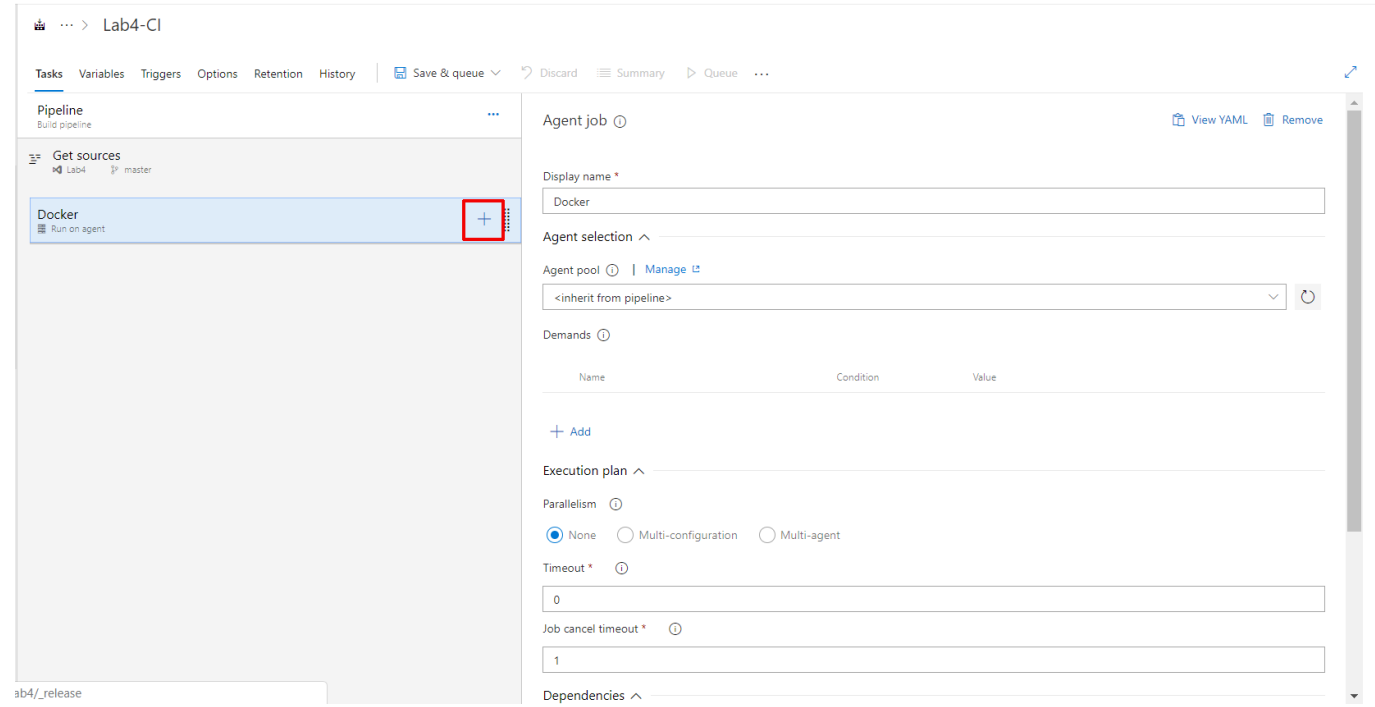
Job cancel timeout \* 1

Dependencies ^

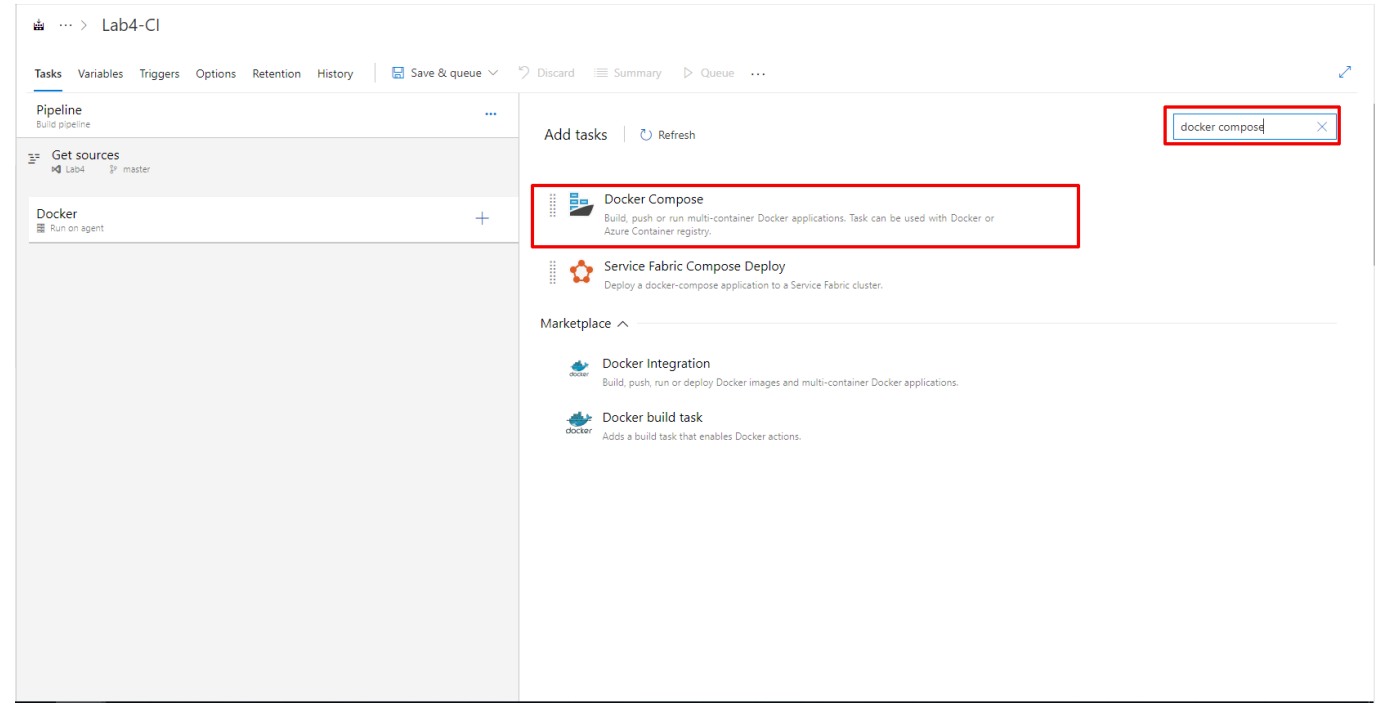
## Task 2: create your Run Services

1. Click on plus button + on your Docker agent

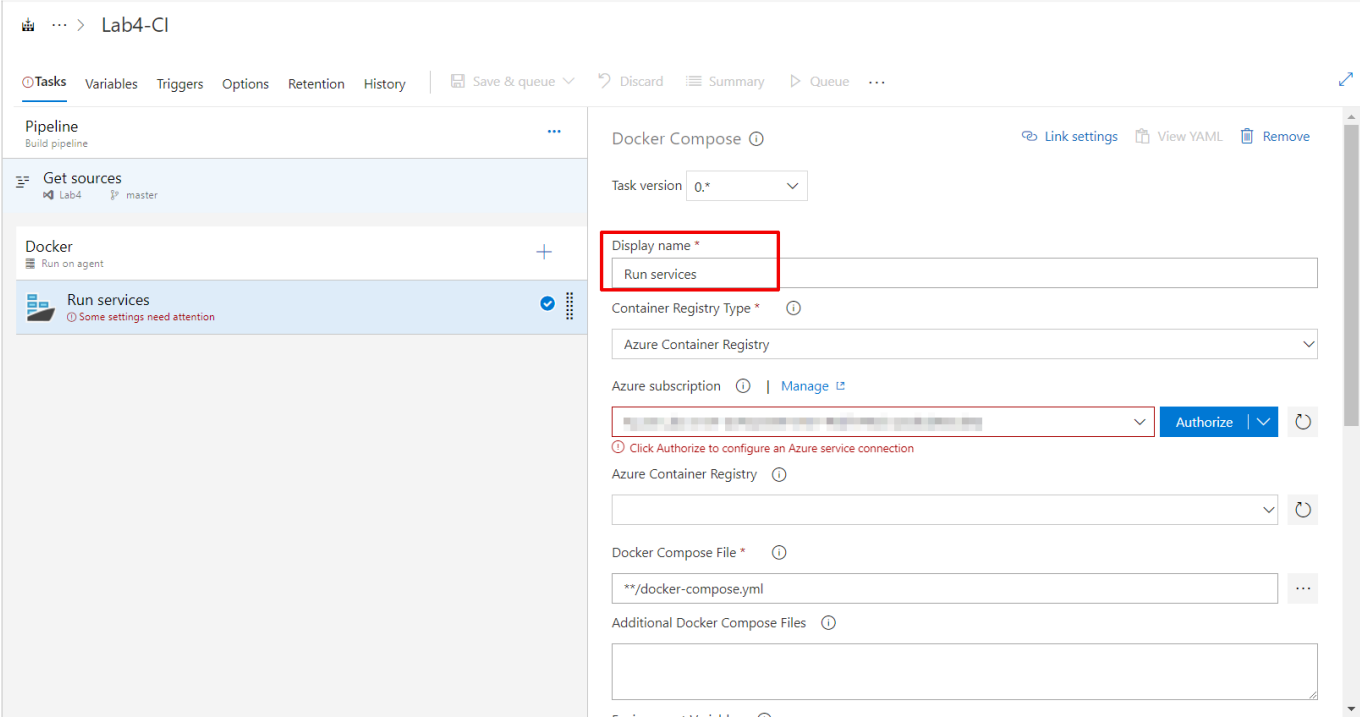




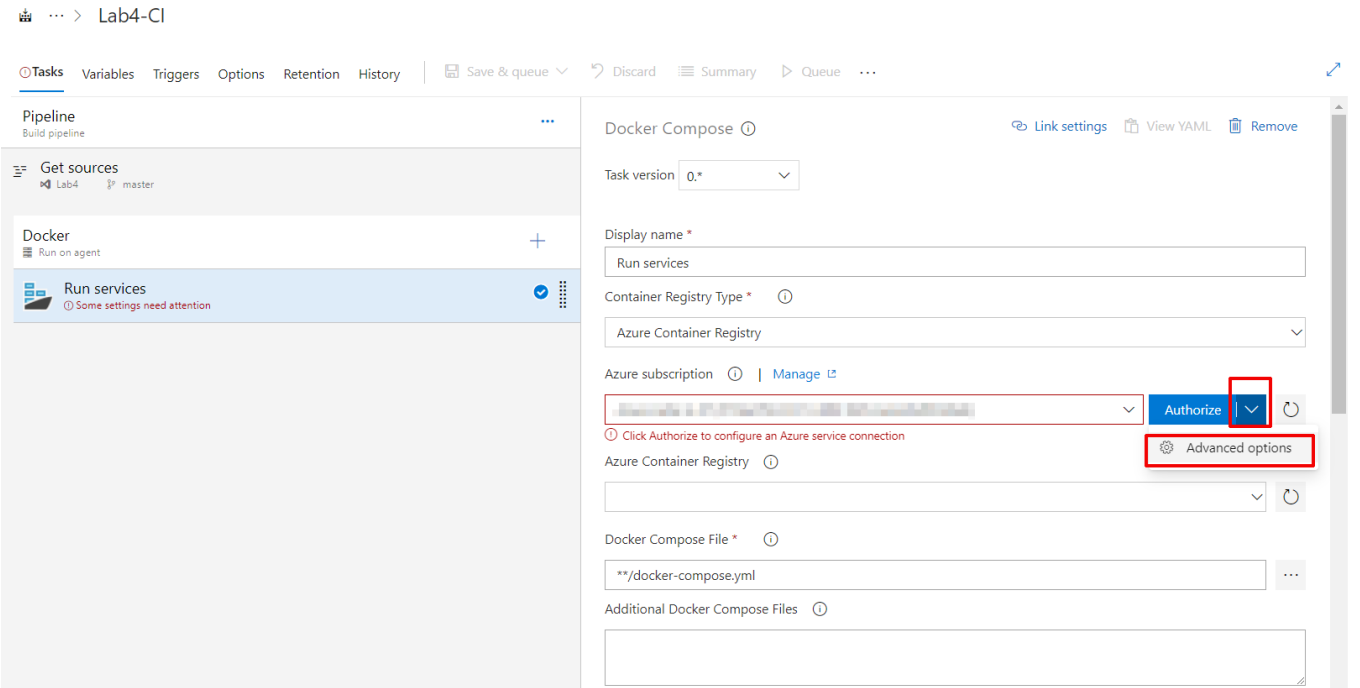
2. Search for **docker compose** and select the first option



3. Set **Run services** on **Display name** input and select your azure subscription



4. Click **Advanced options** as the image shown



5. Click on **use the full version of the service connection dialog** option

×

## Add an Azure Resource Manager service connection

☒ Service Principal Authentication ☐ Managed Identity Authentication

Connection name

Scope level

Subscription

▼

Subscription

Resource Group

▼

Subscriptions listed are from Azure Cloud

A new Azure service principal will be created and assigned with "Contributor" role, having access to all resources within the subscription. Optionally, you can select the Resource Group to which you want to limit access.

If your subscription is not listed above, or your organization is not backed by Azure Active Directory, or to specify an existing service principal, [use the full version of the service connection dialog.](#)

☒ Allow all pipelines to use this connection.

OK

Close

6. Set `serviceConnection` on **Connection name** input

7. Put your **Service Principal Details** given at the beginning of the lab and click in **Ok**

- **Application/Client Id**
- **Application Secret Key**

×

## Add an Azure Resource Manager service connection

☒ Service Principal Authentication   ☐ Managed Identity Authentication

Connection name

ServiceConnection

Environment

AzureCloud

Scope level

Subscription

Subscription ID

Subscription name

Service principal client ID

☒ Service principal key   ☐ Certificate

Service principal key

.....

Tenant ID

Connection: Not verified

Verify connection

For help on creating an Azure service principal, see [service connections](#).

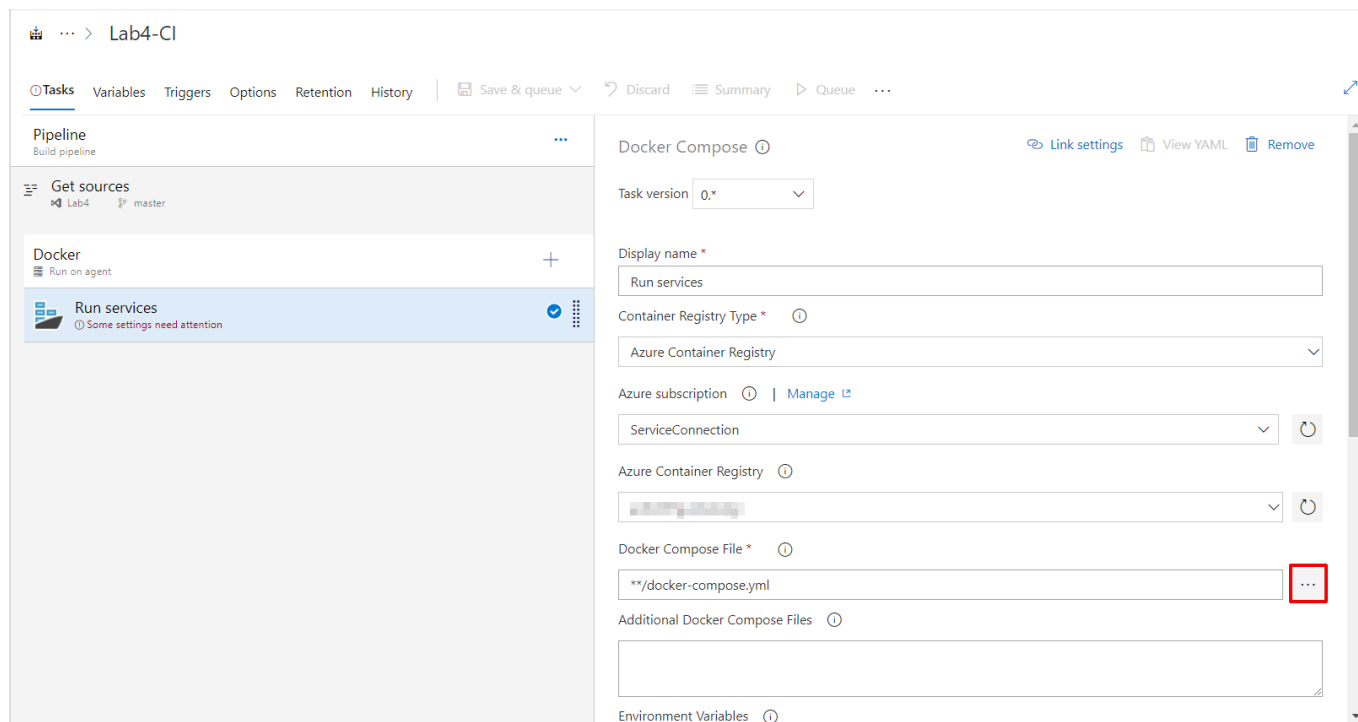
OK

Close

8. Select your container registry

The screenshot displays the Azure DevOps web interface for configuring a pipeline task. On the left, the 'Pipeline' sidebar shows the task list: 'Get sources', 'Lab4', 'master', 'Docker', and 'Run services' (which is highlighted with a blue bar and a warning icon indicating 'Some settings need attention'). The main area is titled 'Docker Compose' and contains several configuration fields: 'Task version' (0.\*), 'Display name' (Run services), 'Container Registry Type' (Azure Container Registry), 'Azure subscription' (ServiceConnection, highlighted with a red box), 'Azure Container Registry' (a text field with a red box around it), 'Docker Compose File' (\*\*/docker-compose.yml), 'Additional Docker Compose Files' (empty), and 'Environment Variables' (empty). At the top of the main area, there are links for 'Link settings', 'View YAML', and 'Remove'.

9. Click on ellipsis button (...) and search the file **docker-compose.ci.build.yml**



The screenshot shows the Azure DevOps pipeline configuration for 'Lab4-CI'. The 'Run services' task is selected, and the 'Docker Compose' configuration is visible. The 'Docker Compose File' field is highlighted with a red box, showing the path '\*\*/docker-compose.yml'.

## Select path

- Lab4
  - BuildScripts
  - src
  - test
  - .gitattributes
  - .gitignore
  - docker-compose.ci.build.yml**
  - docker-compose.dcpj
  - docker-compose.override.yml
  - docker-compose.yml
  - docker.png

OK

Cancel

- Go down in your **Run services** configuration and select **Run service images** on **Action** option
- Uncheck the **Run in Background** option

## 12. In **Output Variables** set as **Task1** in **Reference name** input

The screenshot shows the Jenkins configuration page for a task named 'Run service images'. The left sidebar shows the pipeline structure with 'Run services' selected. The main configuration area on the right has several fields: 'Project Name' is set to '\$(Build.Repository.Name)'; 'Qualify Image Names' is checked; 'Action' is set to 'Run service images'; 'Build Images' is checked; 'Run in Background' is unchecked; 'Abort on Container Exit' is checked. Under 'Output Variables', the 'Reference name' field is set to 'Task1'. A message at the bottom states 'There are no output variables associated with this task'.

Lab4-CI

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue

Pipeline  
Build pipeline

Get sources  
Lab4 master

Docker  
Run on agent

Run services  
Docker Compose

Project Name  
\$(Build.Repository.Name)

Qualify Image Names

Action \*  
Run service images

Build Images

Run in Background

Abort on Container Exit

Advanced Options

Control Options

Output Variables

Reference name  
Task1

Variables list  
There are no output variables associated with this task [more information](#)

## Task 3: create your Build services

1. Click on plus button (+) and search again for Docker compose

The top screenshot shows the 'Run services' task configuration in the Azure DevOps pipeline editor. The task is named 'Run services' and uses the 'Docker Compose' provider. The configuration includes fields for Project Name, Action (set to 'Run service images'), and checkboxes for 'Qualify Image Names', 'Build Images', 'Run in Background', and 'Abort on Container Exit'. The 'Advanced Options', 'Control Options', and 'Output Variables' sections are collapsed. The 'Reference name' is set to 'Task1'.

The bottom screenshot shows the 'Add tasks' section of the pipeline editor. A search bar at the top right contains the text 'docker compose'. Below the search bar, the 'Docker Compose' task is highlighted with a red box. The task description reads: 'Build, push or run multi-container Docker applications. Task can be used with Docker or Azure Container registry.' Other tasks listed include 'Service Fabric Compose Deploy', 'Docker Integration', and 'Docker build task'.

2. Set **Build services** on **Display name** input
3. Select your **ServiceConnection** *previously created* on **Azure subscription**
4. Select your **Azure Container registry**
5. Search the **docker-compose.yml** file clicking on the Ellipsis button (...), selecting the file and clicking **Ok**
6. Put this line **DOCKER\_BUILD\_SOURCE=** on your **Environment Variables**



## Your config should look like this

The screenshot shows the configuration for a pipeline task named 'Build services'. The task is selected in the left-hand 'Pipeline' view. The configuration details on the right are as follows:

- Display name \***: Build services
- Container Registry Type \***: Azure Container Registry
- Azure subscription**: ServiceConnection
- Azure Container Registry**: [Redacted]
- Docker Compose File \***: docker-compose.yml
- Additional Docker Compose Files**: [Empty]
- Environment Variables**: DOCKER\_BUILD\_SOURCE=
- Project Name**: [Empty]

7. Go down and select **Build service images** on **Action** dropdown

8. Set as **Task2** on **Reference name** on **Output Variables** section

The screenshot shows the configuration for a pipeline task named 'Build service images'. The task is selected in the left-hand 'Pipeline' view. The configuration details on the right are as follows:

- \$(Build.Repository.Name)**: [Empty]
- Qualify Image Names**: ☒
- Action \***: Build service images
- Additional Image Tags**: [Empty]
- Include Source Tags**: ☐
- Include Latest Tag**: ☐
- Advanced Options**: [Expanded]
- Control Options**: [Expanded]
- Output Variables**: [Expanded]
- Reference name**: Task2
- Variables list**: There are no output variables associated with this task [more information](#)

## Task 4: Create your Push services

1. Click on plus button (+) and search again for Docker compose

The top screenshot shows the 'Build services' task configuration in the 'Lab4-CI' pipeline. The task is 'Build services' using the 'Docker Compose' provider. The 'Action' is set to 'Build service images'. The 'Reference name' is 'Task2'. The 'Variables list' shows no output variables associated with this task.

The bottom screenshot shows the 'Add tasks' interface. A search box contains 'docker compose'. The 'Docker Compose' task is highlighted, with a description: 'Build, push or run multi-container Docker applications. Task can be used with Docker or Azure Container registry.' Other tasks listed include 'Service Fabric Compose Deploy', 'Docker Integration', and 'Docker build task'.

2. Set **Push services** on **Display name** input
3. Select your **ServiceConnection** *previously created* on **Azure subscription**
4. Select your **Azure Container registry**
5. Search the **docker-compose.yml** file clicking on the Ellipsis button (...), selecting the file and clicking **Ok**
6. Put this line **DOCKER\_BUILD\_SOURCE=** on your **Environment Variables**

## Your config should look like this

The screenshot shows the configuration for a pipeline named 'Lab4-CI'. The left sidebar lists tasks: 'Get sources', 'Run services', 'Build services', and 'Push services'. The 'Push services' task is selected. The main configuration area on the right includes the following fields:

- Display name \***: Push services
- Container Registry Type \***: Azure Container Registry
- Azure subscription**: ServiceConnection
- Azure Container Registry**: [Redacted]
- Docker Compose File \***: docker-compose.yml
- Additional Docker Compose Files**: [Empty]
- Environment Variables**: DOCKER\_BUILD\_SOURCE=
- Project Name**: [Empty]

7. Go down and select **Push service images** on **Action** dropdown

8. Set as **Task3** on **Reference name** on **Output Variables** section

The screenshot shows the configuration for the 'Push service images' task. The left sidebar lists tasks: 'Get sources', 'Run services', 'Build services', and 'Push services'. The 'Push services' task is selected. The main configuration area on the right includes the following fields:

- \$(Build.Repository.Name)**: [Empty]
- Qualify Image Names**: ☒
- Action \***: Push service images
- Additional Image Tags**: [Empty]
- Include Source Tags**: ☐
- Include Latest Tag**: ☐
- Advanced Options**: [Expanded]
- Control Options**: [Expanded]
- Output Variables**: [Expanded]
- Reference name**: Task3
- Variables list**: There are no output variables associated with this task [more information](#)

## Task 5: create your Publish Artifact

1. Click on plus button (+) and search for Publish build artifacts

Lab4-CI

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue

Pipeline Build pipeline

Get sources Lab4 master

Docker Run on agent

Run services Docker Compose

Build services Docker Compose

Push services Docker Compose

\$(Build.Repository.Name)

☒ Qualify Image Names

Action \* Push service images

Additional Image Tags

☐ Include Source Tags

☐ Include Latest Tag

Advanced Options

Control Options

Output Variables

Reference name Task3

Variables list

There are no output variables associated with this task [more information](#)

Lab4-CI

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue

Pipeline Build pipeline

Get sources Lab4 master

Docker Run on agent

Run services Docker Compose

Build services Docker Compose

Push services Docker Compose

Add tasks Refresh

publish build artifacts

Publish Build Artifacts

Publish build artifacts to Azure Pipelines/TFS or a file share

Deprecated tasks

2. Set **Publish Artifact** on **Display name** input
3. Set **myhealthclinic.dacpac** on **Path to publish** input
4. Set **dacpac** on **Artifact name** input
5. Set **PublishBuildArtifacts2** as your **Reference name** in the **Output variables**

Lab4-CI

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

Pipeline Build pipeline

Get sources Lab4 master

Docker Run on agent

Run services Docker Compose

Build services Docker Compose

Push services Docker Compose

Publish Artifact Publish Build Artifacts

Display name \* Publish Artifact

Path to publish \* myhealthclinic.dacpac

Artifact name \* dacpac

Artifact publish location \* Azure Pipelines/TFS

Control Options

Output Variables

Reference name PublishBuildArtifacts2

Variables list

There are no output variables associated with this task [more information](#)

## Task 6: Set your variables

1. Click on **Variables** tab
2. Create these variables clicking on **+ Add** button

Name: **BuildConfiguration** Value: **Release** check the *settable at queue time* Name: **BuildPlatform** Value: **Any CPU**

Lab4-CI

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

Pipeline variables

Variable groups

Predefined variables

Name ↑	Value	Settable at queue time
BuildConfiguration	Release	<input checked="" type="checkbox"/>
BuildPlatform	Any CPU	<input checked="" type="checkbox"/>
system.collectionId	6548c617-84e7-42eb-9a62-270617775037	<input checked="" type="checkbox"/>
system.debug	false	<input checked="" type="checkbox"/>
system.definitionId	< No pipeline id yet >	
system.teamProject	Lab4	

+ Add

## Task 7: config your triggers

1. Click on **Triggers** tab
2. Check the **Enable continuous integration** option

Lab4-CI

Tasks Variables **Triggers** Options Retention History | Save & queue Discard Summary Queue ...

Continuous integration

Lab4 Enabled

Scheduled + Add

No builds scheduled

Build completion + Add

Build when another build completes

Lab4

☒ Enable continuous integration

☐ Batch changes while a build is in progress

Branch filters

Type Branch specification

Include \*/master

+ Add

Path filters

+ Add

## Task 8: config your options

1. Click on **Options** tab
2. Put this line `$(date:yyyyMMdd)$(rev:.r)` on **Build number format**
3. Set as `0` on **Build job timeout in minutes**

Lab4-CI

Tasks Variables Triggers **Options** Retention History | Save & queue Discard Summary Queue ...

Build properties

Define general build pipeline settings

Description

Build number format ⓘ

`$(date:yyyyMMdd)$(rev:.r)`

The new build request is processing

☒ Enabled - queue and start builds when eligible agent(s) available

☐ Paused - queue new builds but do not start

☐ Disabled - do not queue new builds

Automatically link new work in this build

When a build completes successfully, create links to all work items linked to associated changes.

☐ Disabled

Create work item on failure

Create a work item for each failed build

☐ Disabled

Status badge

Azure Pipelines set up now

Build job

Define build job authorization and timeout settings

Build job authorization scope ⓘ

Project collection

Build job timeout in minutes ⓘ

0

Build job cancel timeout in minutes ⓘ

5

Demands

Specify which capabilities the agent must have to run this pipeline.

Name	Condition
+ Add	

## Task 9: config your Retention

1. Click on **Retention** tab
2. Click **Keep for 10 days, 1 good build**

### 3. Click on Delete

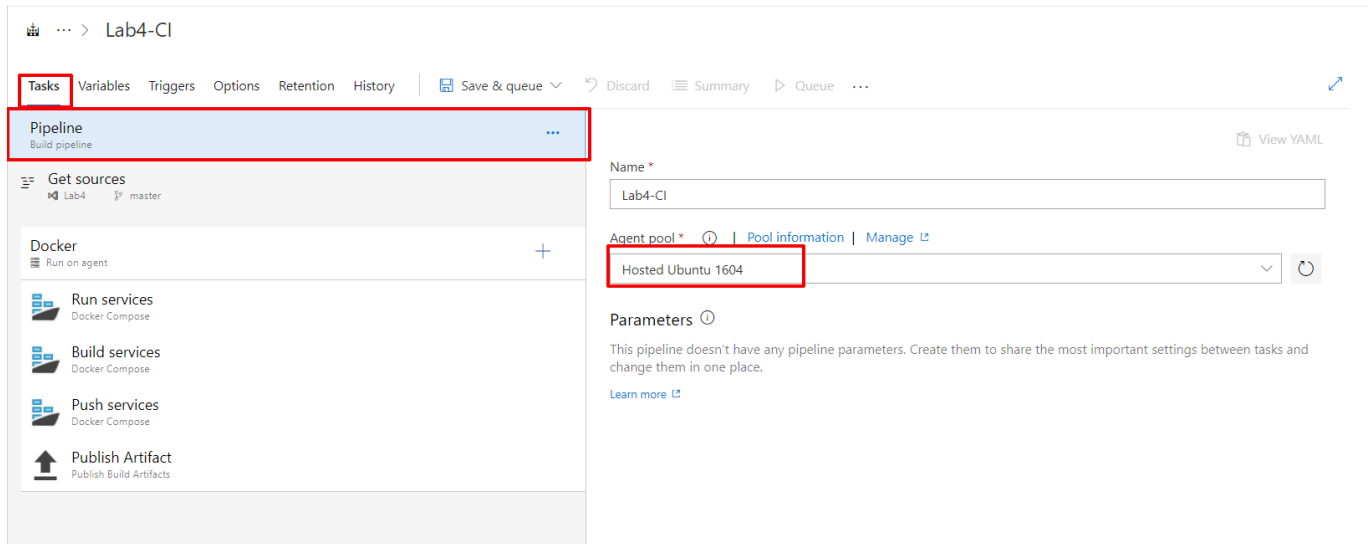
The screenshot shows the Jenkins Lab4-CI configuration page with the 'Retention' tab active. The 'Policies' section on the left lists two policies: 'Keep for 10 days, 1 good build' (highlighted with a red box) and 'Keep for 30 days, 10 good builds'. The 'Settings' section on the right shows branch filters and cleanup options. The 'Delete' button in the top right of the settings panel is also highlighted with a red box.

### Your Retention should look like this

The screenshot shows the Jenkins Lab4-CI configuration page with the 'Retention' tab active. The 'Policies' section on the left lists two policies: 'Keep for 10 days, 1 good build' and 'Keep for 30 days, 10 good builds' (highlighted with a blue box). The 'Settings' section on the right shows branch filters and cleanup options. The 'Days to keep' is set to 30 and 'Minimum to keep' is set to 10.

## Task 10: Set your host

1. Click on **Tasks** tab
2. Click on Pipelines
3. Select **Hosted Ubuntu 1604** option from **Agent Pool** dropdown



4. Click on **Save & queue** dropdown and select the **save** option
5. Put any comment you want

## Exercise 3: configure your Continuous Delivery

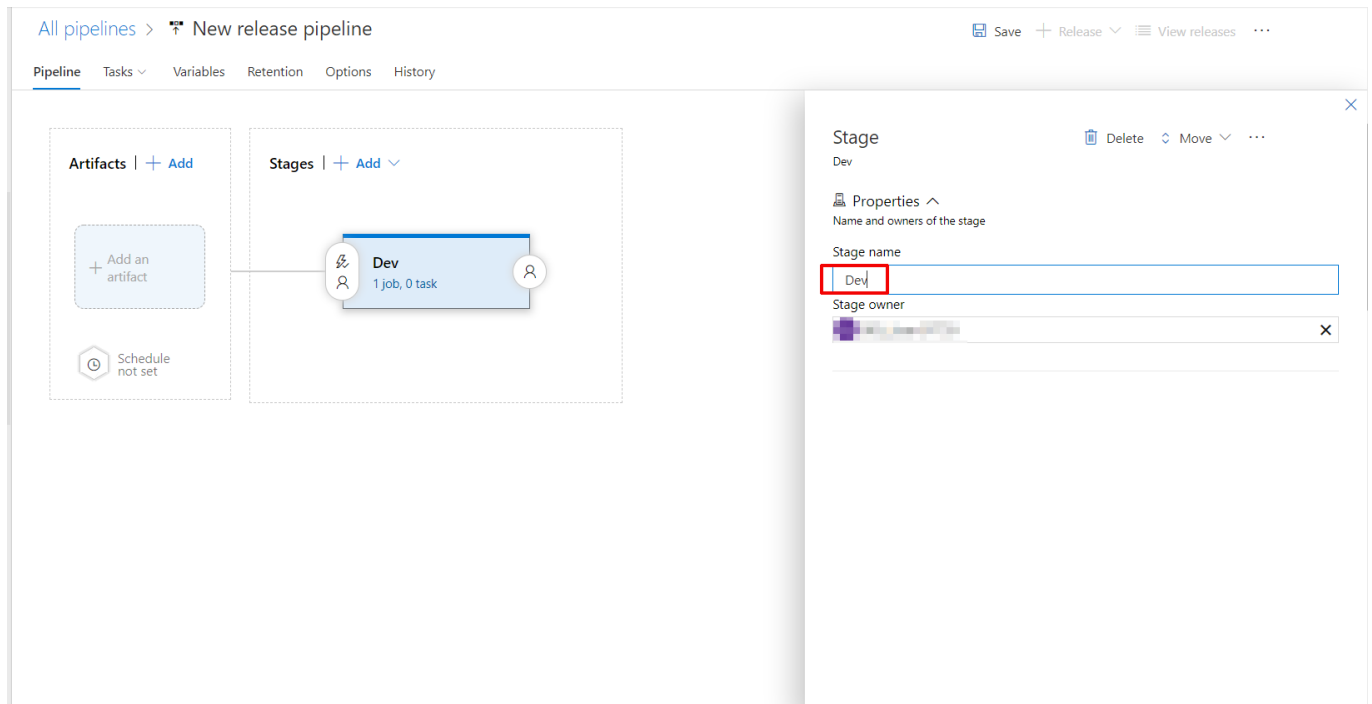
### Task 1: create the pipeline

1. Navigate to the **Releases** section under the **Pipelines** tab and select **New pipeline** and select **Empty job**



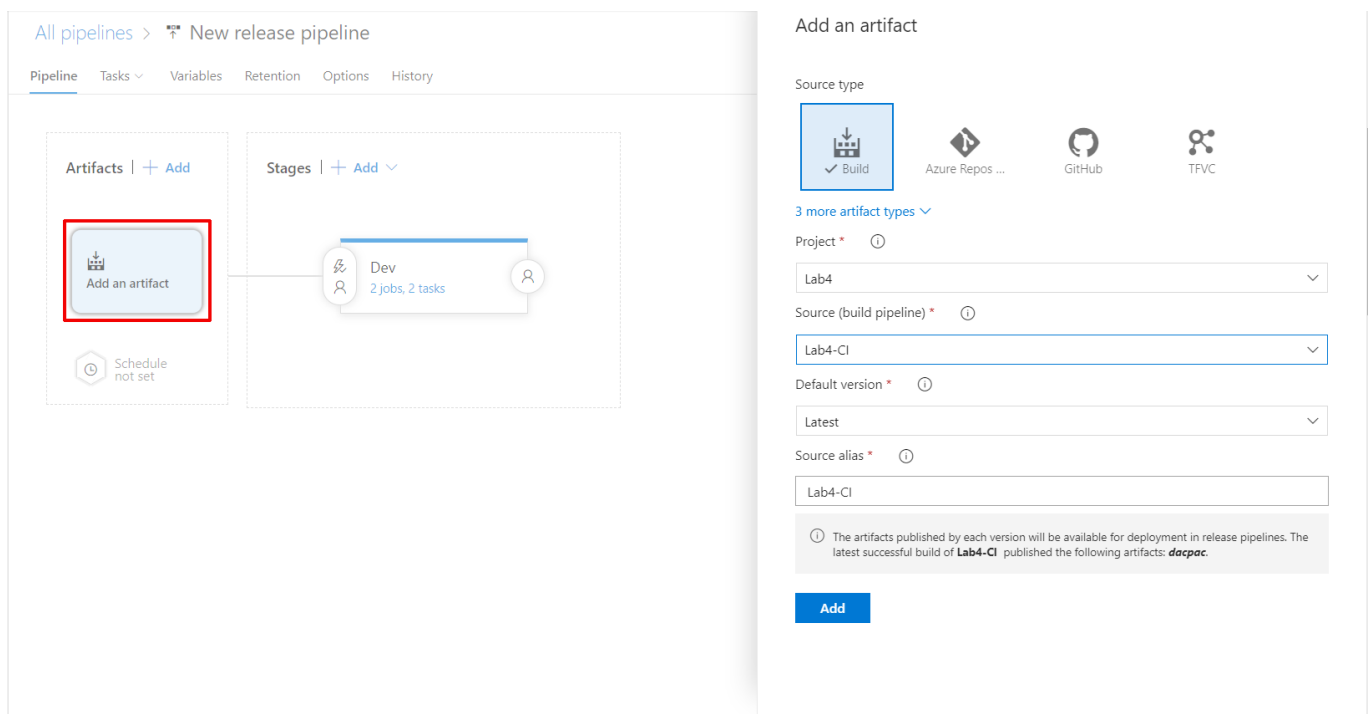
The image shows two screenshots of the Azure DevOps interface. The top screenshot shows the 'Releases' section in the left-hand navigation pane, which is highlighted with a red box. The main area displays a message: 'No release pipelines found' with a subtext 'Automate your release process in a few easy steps with a new pipeline' and a 'New pipeline' button, also highlighted with a red box. The bottom screenshot shows the 'New release pipeline' wizard. The 'Select a template' step is highlighted with a red box, showing the option 'Or start with an Empty job'. The 'Featured' section on the right lists various deployment templates, including 'Azure App Service deployment', 'Deploy a Java app to Azure App Service', 'Deploy a Node.js app to Azure App Service', 'Deploy a PHP app to Azure App Service and Azure Database for MySQL', 'Deploy a Python app to Azure App Service and Azure database for MySQL', 'Deploy to a Kubernetes cluster', and 'IIS website and SQL database deployment'.

## 2. Set **Dev** on **Stage name** option



The screenshot shows the 'New release pipeline' configuration page. On the left, the 'Artifacts' section has a button labeled 'Add an artifact'. The 'Stages' section shows a stage named 'Dev' with '1 job, 0 task'. On the right, a 'Stage' properties panel is open for the 'Dev' stage. The 'Stage name' field is highlighted with a red box, showing the text 'Dev'.

3. Click on **Add an artifact** and set the inputs as below



The screenshot shows the 'Add an artifact' dialog. The 'Source type' is set to 'Build'. The 'Project' is 'Lab4', 'Source (build pipeline)' is 'Lab4-CI', and 'Default version' is 'Latest'. The 'Source alias' is 'Lab4-CI'. A blue 'Add' button is at the bottom.

4. Set Continuous deployment trigger clicking on the ray icon and switching to enable the Continuous deployment trigger

Continuous deployment trigger  
Build: Lab4-CI  
**Enabled**  
Creates a release every time a new build is available.

Build branch filters ⓘ  
No filters added.  
[+ Add](#) | [v](#)

Pull request trigger  
Build: Lab4-CI  
**Disabled**  
Enabling this will create a release every time a selected artifact is available as part of a pull request workflow

## Task 2: Config your DB deployment

1. Click on **Tasks** tab
2. Click on **Agent job**
3. set **DB Deployment** on **Display name** input
4. Create this variable clicking on **+ Add** button **Name:** `sqlpackage` **Condition:** `exists`
5. Click on plus button (+) and search for `azure SQL database deployment`
6. Select the first option then add

Add tasks | Refresh

Search:

**Azure SQL Database Deployment**  
Deploy Azure SQL DB using DACPAC or run scripts using SQLCMD  
by Microsoft Corporation [Learn more](#) **Add**

**Azure Database for MySQL Deployment**  
Run your scripts and make changes to your Azure Database for MySQL

Marketplace ^

**Entity Framework Core Migrations Script Generator**  
Tool for projects that use Entity Framework Core Code-First. Generates migration scripts which can be used to update a database (for instance with the task 'Azure SQL Database Deployment').

7. Select your **Azure SQL DacpacTask**
8. Put **Execute Azure SQL : DacpacTask** as the **Display name**
9. Select your **ServiceConnection** on **Azure Subscription**
10. Set `$(SQLserver)` on **Azure SQL Server** input
11. Set `$(DatabaseName)` on **Database** input
12. Set `$(SQLadmin)` (with a blank space at the beginning) on **Login** input
13. Set `$(Password)` on **Password** input

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Dev  
Deployment process

DB deployment  
Run on agent

Execute Azure SQL : DacpacTask  
Some settings need attention

ServiceConnection

SQL Database

Authentication Type \*  
SQL Server Authentication

Azure SQL Server \*  
\$(SQLServer)

Database \*  
\$(DatabaseName)

Login \*  
\$(SQLadmin)

Password \*  
\$(Password)

Deployment Package

Deploy type \*  
SQL DACPAC File

Action \*  
Publish

14. Go down and set `$(System.DefaultWorkingDirectory)/**/*.dacpac` as your **DACPAC File** on **Deployment Package** section

15. Set `SqlAzureDacpacDeployment1` as your **Reference name** on **Output Variables** section

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Dev  
Deployment process

DB deployment  
Run on agent

Execute Azure SQL : DacpacTask  
Azure SQL Database Deployment

Action \*  
Publish

DACPAC File \*  
\$(System.DefaultWorkingDirectory)/\*\*/\*.dacpac

Publish Profile

Additional SqlPackage.exe Arguments

Firewall

Control Options

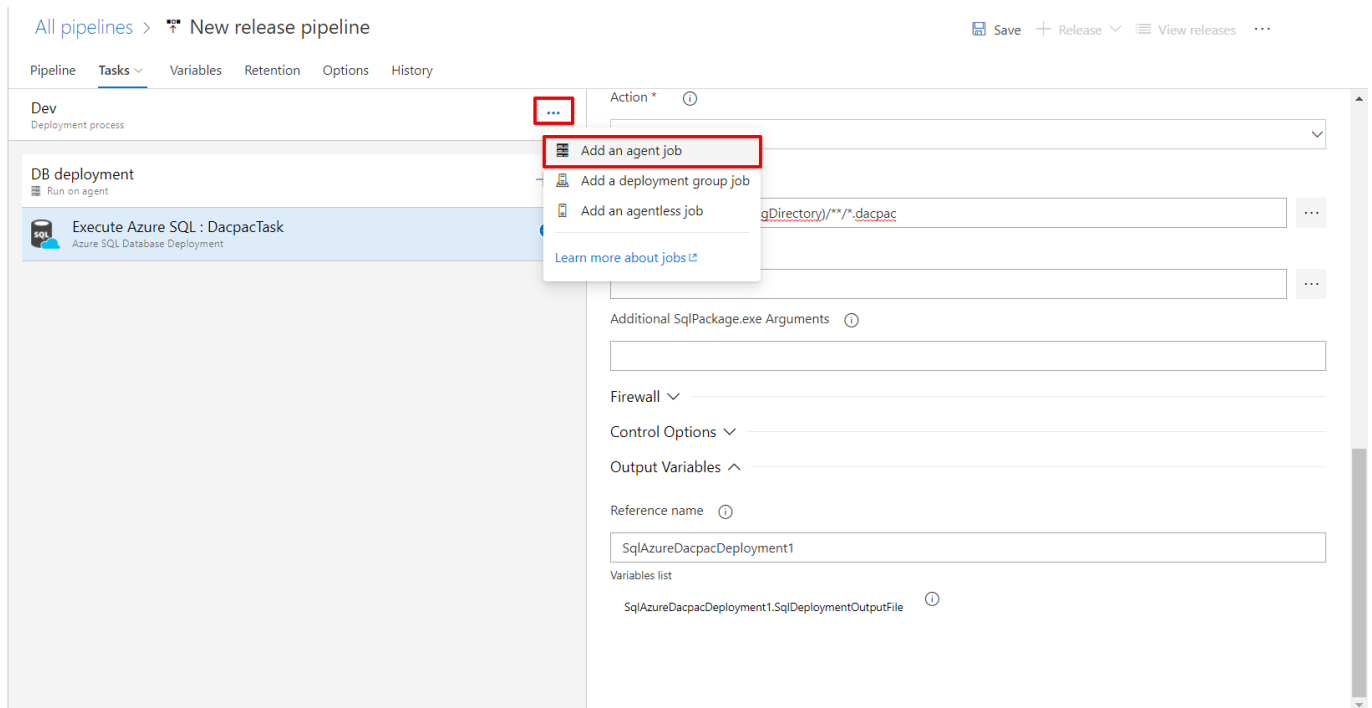
Output Variables

Reference name  
SqlAzureDacpacDeployment1

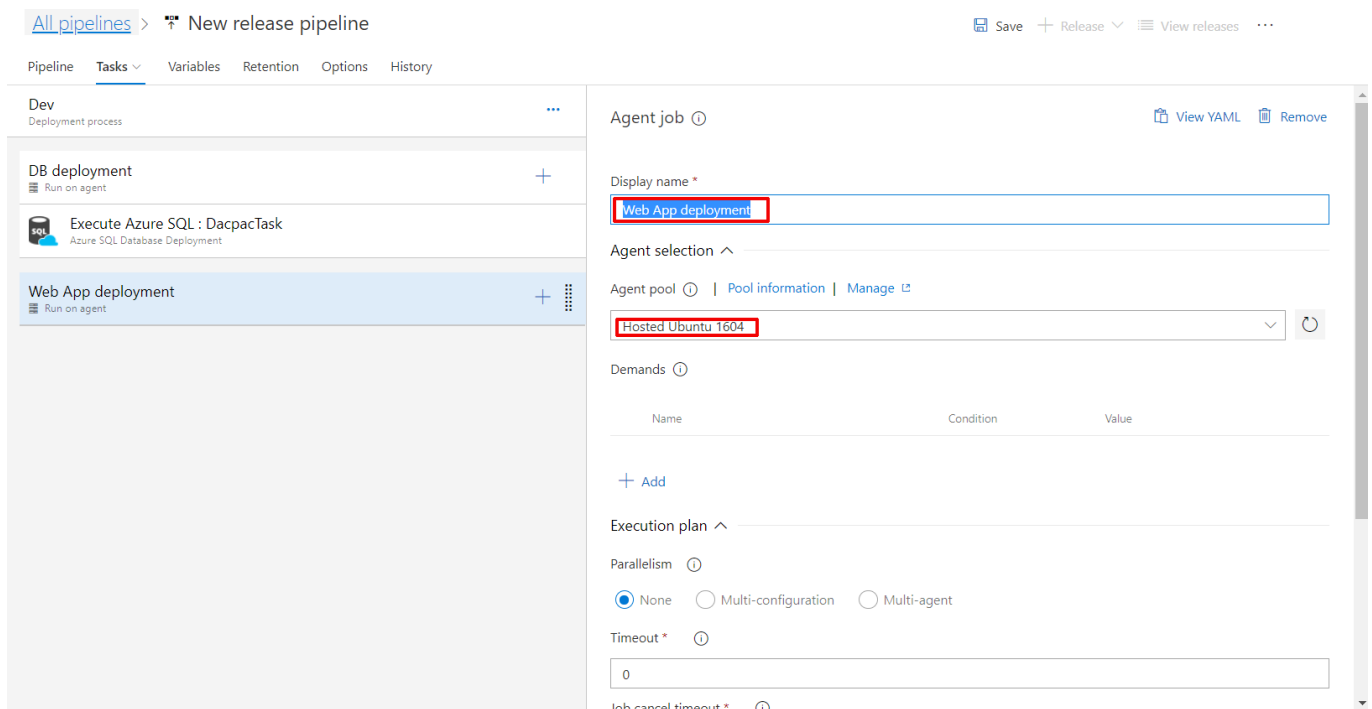
Variables list  
SqlAzureDacpacDeployment1.SqlDeploymentOutputFile

## Task 3: Config your Web App deployment

1. Click on Ellipsis button (...) on **Dev** (Deployment process) and select **Add an Agent Job**



2. Click on your new **Agent job**
3. Set **Web App deployment** as your **Display name**
4. Select the **Hosted Ubuntu 1604** option from **Agent pool** dropdown



5. Click on plus button (+) and search for **Azure App Service Deploy**
6. Select the first one then click on **Add**

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Dev  
Deployment process

DB deployment  
Run on agent

Execute Azure SQL : DacpacTask  
Azure SQL Database Deployment

Web App deployment  
Run on agent

Add tasks | Refresh

Azure App Service Deploy

Update Azure App Services on Windows, Web App on Linux with built-in images or Docker containers, ASP.NET, .NET Core, PHP, Python or Node.js based Web applications, Function Apps on Windows or Linux with Docker Containers, Mobile Apps, API applications, Web Jobs using Web Deploy / Kudu REST APIs

by Microsoft Corporation

Learn more

Marketplace

Azure App Service: Set App settings  
Add settings to the App Settings of an Azure App Service.

Azure App Service: Add or remove IP Restrictions  
Add or remove IP Restrictions, use custom address or add the IP of the build agent.

Octopus Deploy Integration  
Build and Release tasks and other features for integrating with Octopus Deploy. Octopus is great for deploying ASP.NET or Core apps to on IIS or Azure, SQL databases, Windows services and much more.

Kudu ZipDeploy  
Deploy a Zip package to the Azure App Service using the Kudu zipdeploy api

7. Click on your new **Azure App Service Deploy**:
8. Select **3.\*** option from **Task version** dropdown
9. Set **Azure App Service Deploy** as your **Display name**
10. Select **ServiceConnection** on **Azure subscription**
11. Select **Linux web App** on **App type**
12. Select your **webapp** on **App Service Name**
13. Set **\$(ACR)** on **Registry or Namespace** input

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Dev  
Deployment process

DB deployment  
Run on agent

Execute Azure SQL : DacpacTask  
Azure SQL Database Deployment

Web App deployment  
Run on agent

Azure App Service Deploy  
Some settings need attention

Azure App Service Deploy

Task version 3.\*

Display name \*  
Azure App Service Deploy

Azure subscription \* | Manage  
ServiceConnection

App type \*  
Linux Web App

App Service name \*  
[Redacted]

☐ Deploy to slot

Image Source  
Container Registry

Registry or Namespace \*  
\$(ACR)

Image \*

14. Put **myhealth.web** in the **Image** input
15. Put **\$(BUILD.BUILDID)** in **Tag** input
16. Put **AzureRmWebAppDeployment1** in **Reference name** on **Output Variables** section

Dev  
Deployment process

DB deployment  
Run on agent

Execute Azure SQL : DacpacTask  
Azure SQL Database Deployment

Web App deployment  
Run on agent

Azure App Service Deploy  
Azure App Service Deploy

\$(ACR)

Image \*  
myhealth.web

Tag  
\$(BUILD.BUILDID)

Startup command

Post Deployment Action

Application and Configuration Settings

Output

Control Options

Output Variables

Reference name  
AzureRmWebAppDeployment1

Variables list  
There are no output variables associated with this task [more information](#)

## Task 4: Config your variables

1. Click on **Variables** tab
2. Add these variables clicking on + **Add** button

**Name:** ACR **Value:** YOUR\_ACR.azurecr.io **Scope:** Release **Name:** DatabaseName **Value:** mhcdb **Scope:** Release **Name:** Password **Value:** P2ssw0rd1234 **Scope:** Release **Name:** SQLadmin **Value:** sqladmin **Scope:** Release **Name:** SQLserver **Value:** YOUR\_DBSERVER.database.windows.net **Scope:** Release

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Pipeline variables

Variable groups

Predefined variables

Filter by keywords

Scope

List Grid

Name	Value	Scope
ACR	YOUR_ACR.azurecr.io	Release
DatabaseName	mhcdb	Release
Password	P2ssw0rd1234	Release
SQLadmin	sqladmin	Release
SQLserver	YOUR_DBSERVER.database.windows.net	Release

+ Add

3. Click on **save** button

All pipelines > New release pipeline Save Release View releases ...

Pipeline Tasks Variables Retention Options History

Pipeline variables

Variable groups

Predefined variables

Filter by keywords Scope

Name	Value	Scope
ACR	[redacted].azurecr.io	Release
DatabaseName	mhcdb	Release
Password	P2ssw0rd1234	Release
SQLadmin	sqladmin	Release
SQLserver	[redacted].database.windows.net	Release

+ Add

## Exercise 4: Initiate the CI Build and Deployment through code commit

1. Click on **Files** section under the **Repos** tab and navigate to the Docker/src/MyHealth.Web/Views/Home folder and open the Index.cshtml file for editing

Azure DevOps

/ DockerLab / Repos / Files / Docker

master Docker / src / MyHealth.Web / Views / Home / Index.cshtml

Contents History Compare Blame Edit Rename Delete Download

```

1  @{
2      ViewData["Title"] = "HealthClinic";
3  }
4
5  <header>
6      <nav class="navbar navbar-default">
7          <div class="container-fluid mh-nav">
8              <div class="navbar-header">
9                  <button type="button" class="navbar-toggle hamburger collapsed" data-toggle=
10                     <span class="sr-only"></span>
11                     <span class="icon-bar"></span>
12                     <span class="icon-bar"></span>
13                     <span class="icon-bar"></span>
14                 </button>
15                 <a class="navbar-brand" href="/">
16                     
17                 </a>
18             </div>
19
20             <div class="collapse navbar-collapse id="mh-menu">
21                 <div class="navbar-header visible-xs-block">
22                     
25
26                 <ul class="nav navbar-nav">
27                     <li class="active"><a href="#">HOME</a></li>
28                     <li><a href="#">JOIN US</a></li>
29                     <li><a href="#">TOUR INSIDE</a></li>
30                 </ul>
31
32                 <ul class="nav navbar-nav login-container hidden-xs">
33                     <li><a asp-controller="Account" asp-action="Login">Login</a></li>

```

2. Modify the text **JOIN US** to **CONTACT US** on the line number 28 and then click on the **Commit** button. This action would initiate an automatic build for the source code





3. After clicking **Commit** button, add a comment and click on **Commit**

A screenshot of a 'Commit' dialog box. The title bar says 'Commit' with a close button (X) on the right. Inside the dialog, there is a 'Comment' section with a text input field containing the text 'Updated Index.cshtml'. Below this is a 'Branch name' section with a text input field containing the text 'master'. At the bottom is a 'Work items to link' section with a search input field containing the text 'Search work items by ID or title' and a dropdown arrow on the right. At the bottom right of the dialog are two buttons: 'Commit' (in blue) and 'Cancel' (in grey).

4. Click on **Builds** tab, and subsequently select the commit name

Azure DevOps interface showing the 'Builds' tab for the 'MHCDocker.build' pipeline. The left sidebar highlights 'Builds'. The main area shows a table with one build entry: 'Updated Index.cshtml' (Commit) for build #20190209.2 on the master branch. The build is in a 'Queued' state.

#20190209.2: Updated Index.cshtml  
Triggered today at 23:35 for DC Courses Docker master 586269e

Logs Summary Tests

**Docker Job** Started: 08/02/2019, 23:39:48  
Pool: Hosted Ubuntu 1604 · Agent: Hosted Agent 3m 29s

Step	Status	Duration
Initialize job	succeeded	1s
Initialize Agent	succeeded	<1s
Checkout	succeeded	8s
Run services	succeeded	3m 4s

Build services 15s

```

Starting: Build services
=====
Task       : Docker Compose
Description: Build, push or run multi-container Docker applications. Task can be used with Docker or Azure Container registry.
Version    : 0.4.26
Author     : Microsoft Corporation
Help       : [More Information](https://go.microsoft.com/fwlink/?linkid=848006)
=====
[command]/usr/local/bin/docker-compose -f /home/vsts/work/1/s/docker-compose.yml -f /home/vsts/agents/2.146.0/.docker-compose.1549690983352.yml -p Docker build
Building myhealth.web
  
```

- The Build will generate and push the docker image of the web application to the Azure Container Registry. Once the build is completed, the build summary will be displayed.

#20190209.2: Updated Index.cshtml

[Release](#)
[Artifacts](#)

Triggered today at 23:35 for DC Courses · Docker master · 586269e Retained by release

[Logs](#)
[Summary](#)
[Tests](#)

---

### Docker Job

Pool: Hosted Ubuntu 1604 · Agent: Hosted Agent Started: 08/02/2019, 23:39:48  
... 4m 9s

✓ Prepare job · succeeded	<1s
✓ Initialize job · succeeded	1s
✓ Initialize Agent · succeeded	<1s
✓ Checkout · succeeded	8s
✓ Run services · succeeded	3m 4s
✓ Build services · succeeded	32s
✓ Push services · succeeded	18s
✓ Publish Artifact · succeeded	3s
✓ Post-job: Checkout · succeeded	<1s

6. Navigate to the [Azure Portal](#) and click on the **App Service** that was created at the beginning of this lab. Select the **Container Settings** option and provide the information as suggested and then click the **Save** button

Microsoft Azure

+

 Create a resource

Home

Dashboard

All services

FAVORITES

All resources

Resource groups

App Services

Function Apps

SQL databases

Azure Cosmos DB

Virtual machines

Load balancers

Storage accounts

Virtual networks

Azure Active Directory

Monitor

Advisor

Security Center

Cost Management + Bill...

Search resources, services, and docs

Azure services

See all (+100) >

Virtual machines

Storage accounts

App Services

SQL databases

Azure Database for PostgreSQL

Azure Cosmos DB

Kubernetes services

Function Apps

Azure Databricks

Cognitive Services

Make the most out of Azure

Learn Azure with free online courses by Microsoft

Microsoft Learn

Monitor your apps and infrastructure

Azure Monitor

Secure your apps and infrastructure

Security Center

Optimize performance, reliability, security, and costs

Azure Advisor

Connect to Azure via an authenticated browser-based shell

Cloud Shell

Recent resources

See all your recent resources > See all your resources >

NAME	TYPE	LAST VIEWED
mhwebapp	App Service	Just now
mhcdb (mhcdbaseserver/mhcdb)	SQL database	40 min ago
acrmhc	Container registry	40 min ago
DockerRG	Resource group	2 h ago
MyVNETD	Virtual network	3 h ago
Pase para Azure: patrocino	Subscription	3 h ago

Useful links

Get started or go deep with technical docs

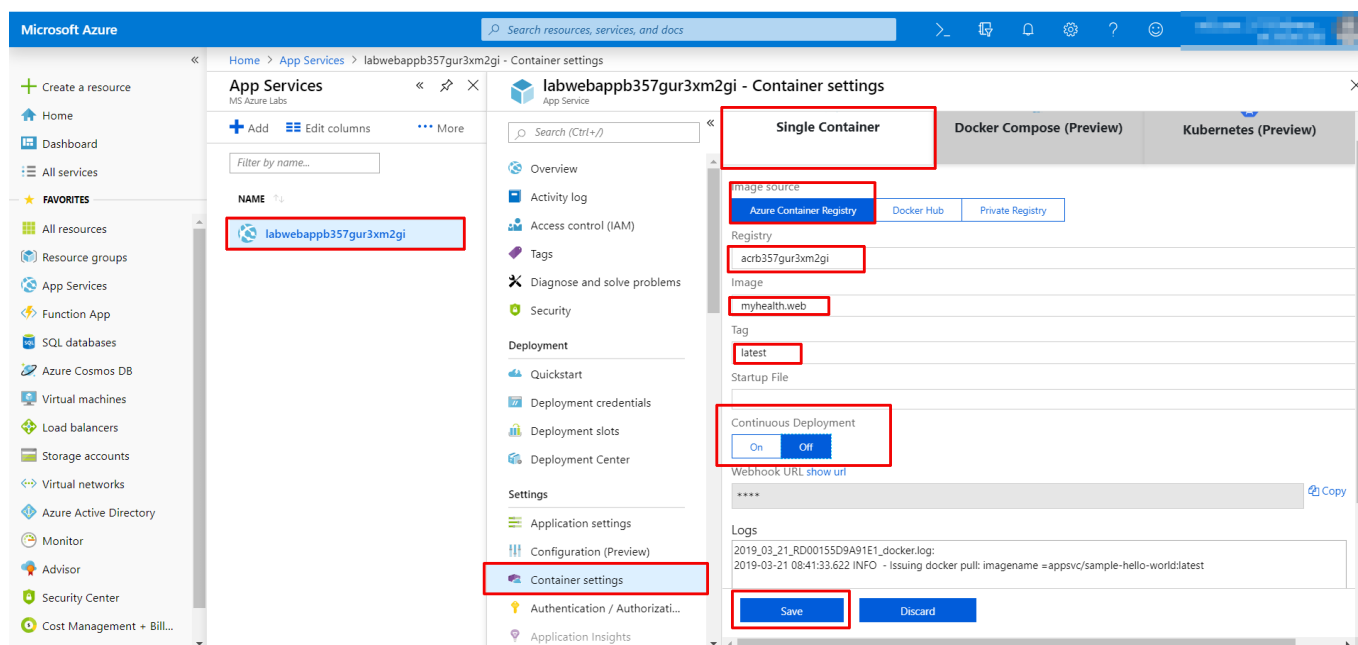
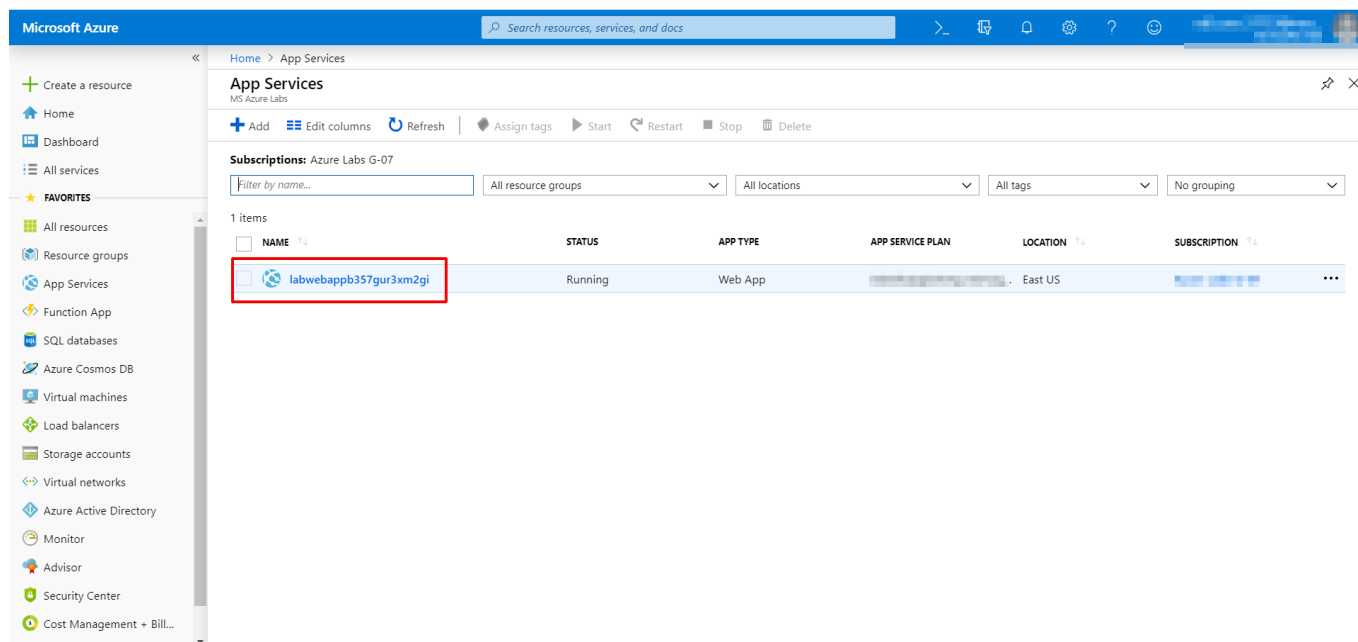
Our articles include everything from quickstarts, samples, and tutorials to help you get started, to SDKs and architecture guides for designing applications.

Discover Azure products

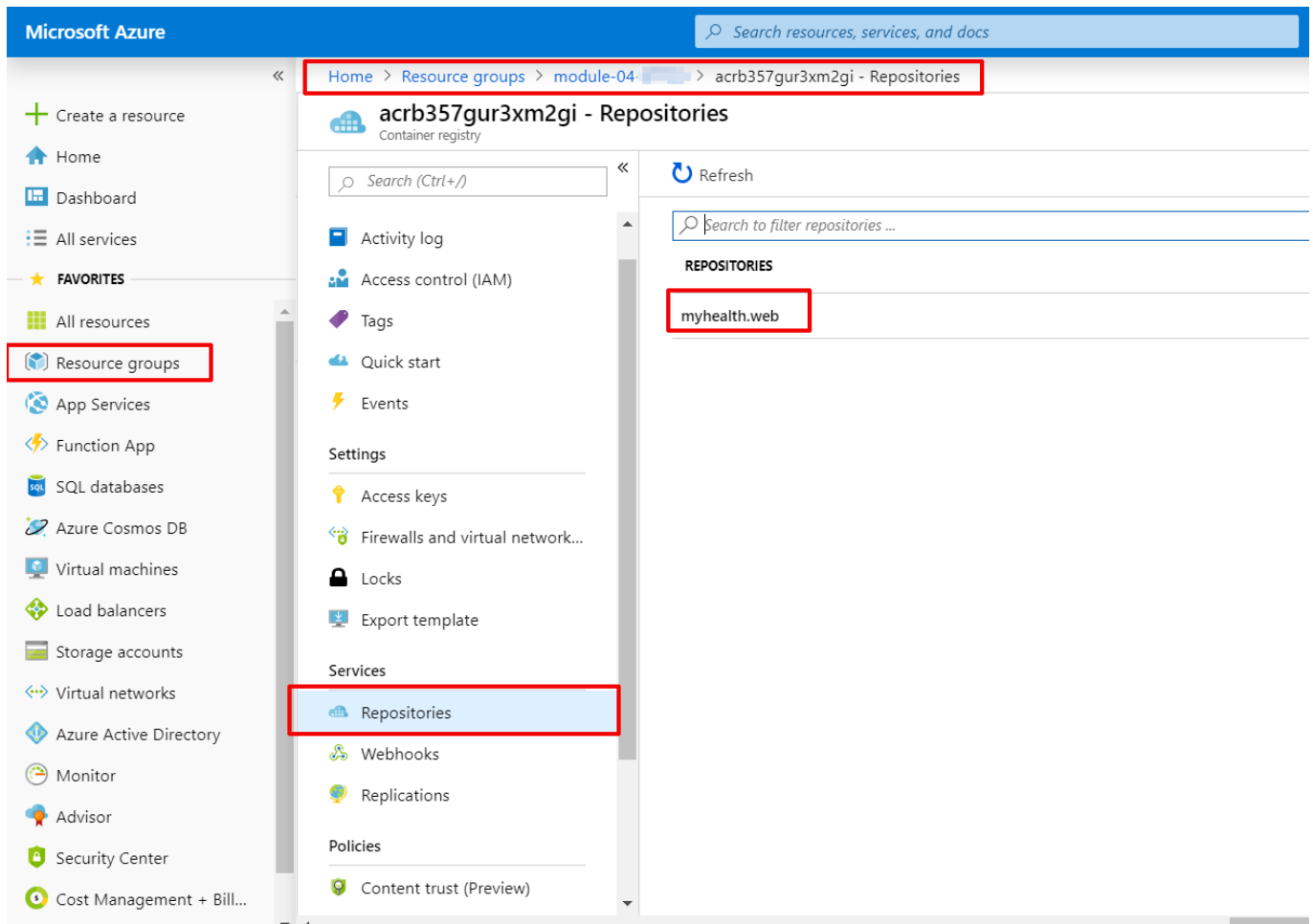
Explore Azure offers that help turn ideas into solutions, and get info on support, training, and pricing.

Keep current with Azure updates

Learn more and what's on the roadmap and subscribe to notifications to stay informed. Azure.Source wraps up all the news from last week in a...

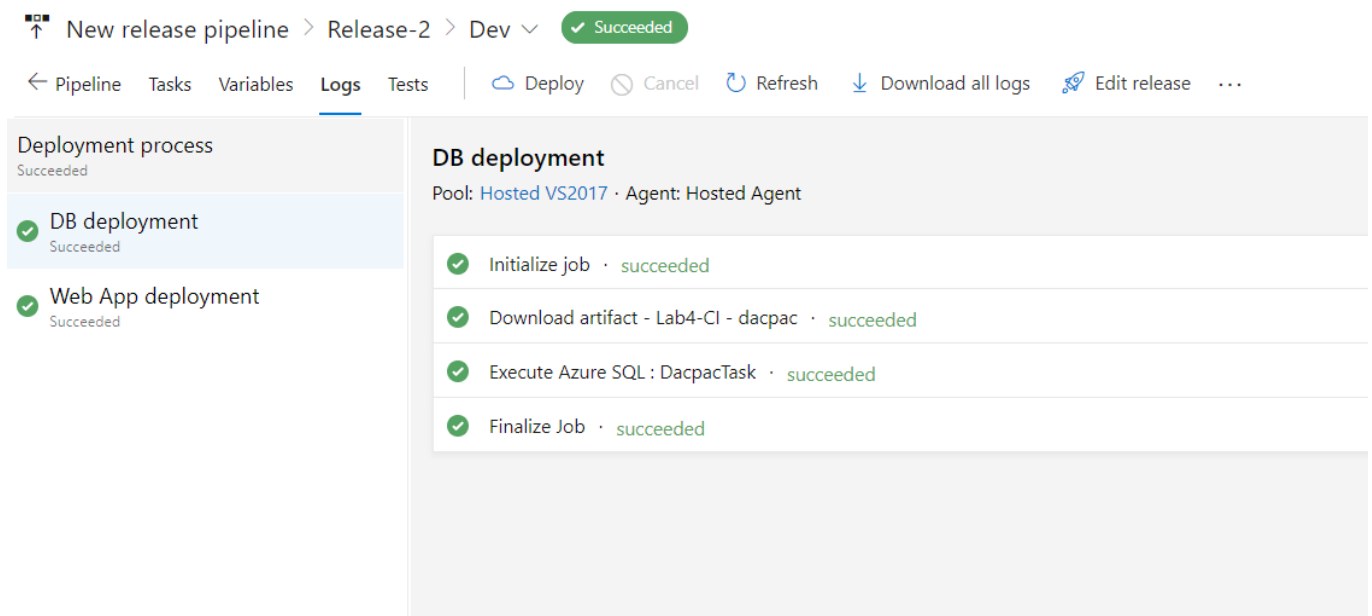


7. Navigate to the **Azure Container Portal** and then select the **Repositories** option to view the generated docker images



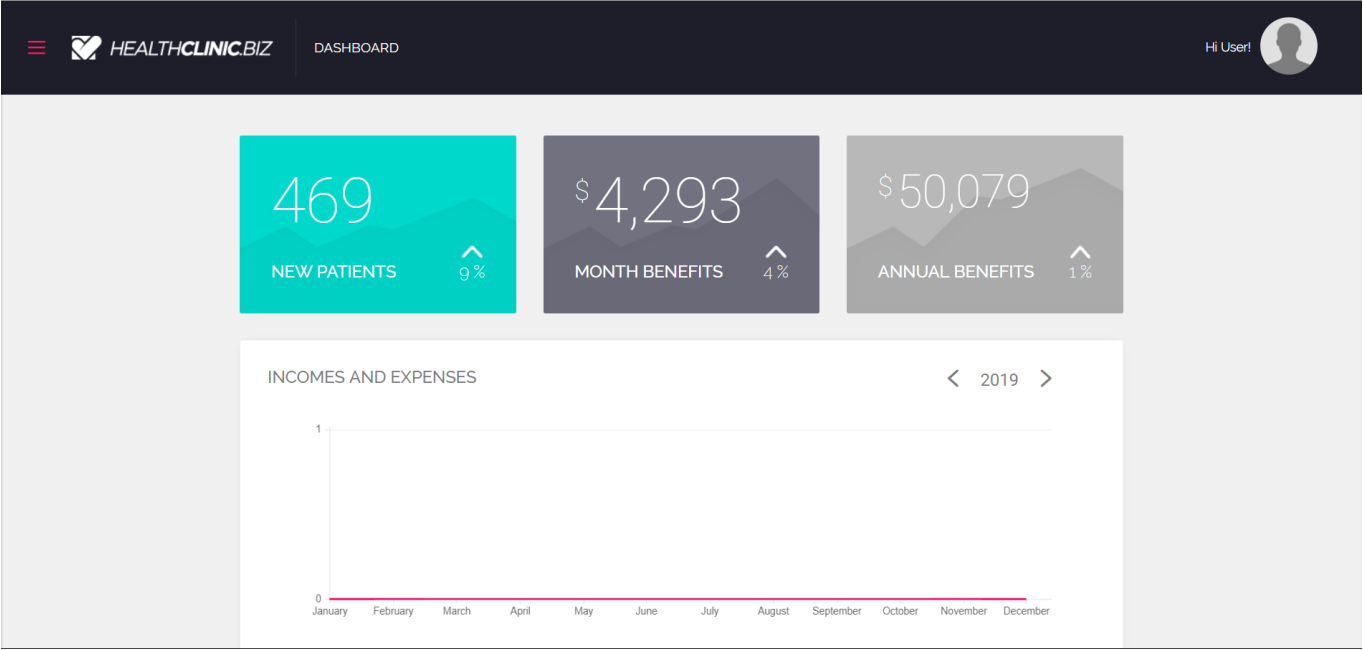
8. Navigate to the **Releases** section in **Azure DevOps** under **Pipelines** tab and double-click on the latest release displayed on the page. Click on Logs to view the details of the release in progress

**note** In case doesn't exist any release you can create a new one clicking on **create a release** and selecting the **Dev** from the pipeline



9. Navigate back to the [Azure Portal](#) and click on the **Overview** section of the **App Service**. Click on the link displayed under the **URL** field to browse the application and view the changes

10. Use the credentials **Username:** user and **Password:** P2ssw0rd@1 to login to the HealthClinic web application.



End of the lab