**Automating Selenium Tests in Azure Pipelines**

**Overview**

[Selenium](http://www.seleniumhq.org/) is a portable open source software-testing framework for web applications. It has the capability to operate on almost every operating system. It supports all modern browsers and multiple languages including .NET (C#), Java.

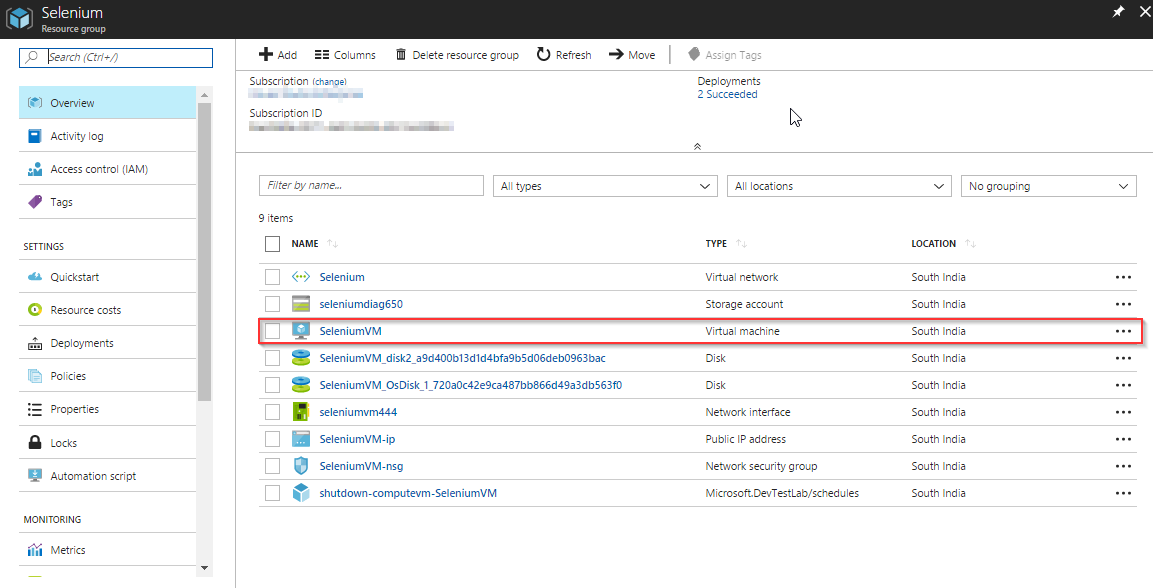
In this lab, you will learn how to execute selenium test cases on a C# web application, as part of the Azure DevOps Release pipeline.

### *Before you begin*

1. Click on the deploy to Azure button below, to provision a Windows Server 2016 virtual machine along with SQL Express 2017 and browsers - Chrome and FireFox.

<https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fraw.githubusercontent.com%2FMicrosoft%2Falmvm%2Fmaster%2Flabs%2Fvstsextend%2Fselenium%2Farmtemplate%2Fazuredeploy.json>

It should take approximately 20-25 minutes to provision the resources. Once the deployment is successful, you will see the resources as shown.



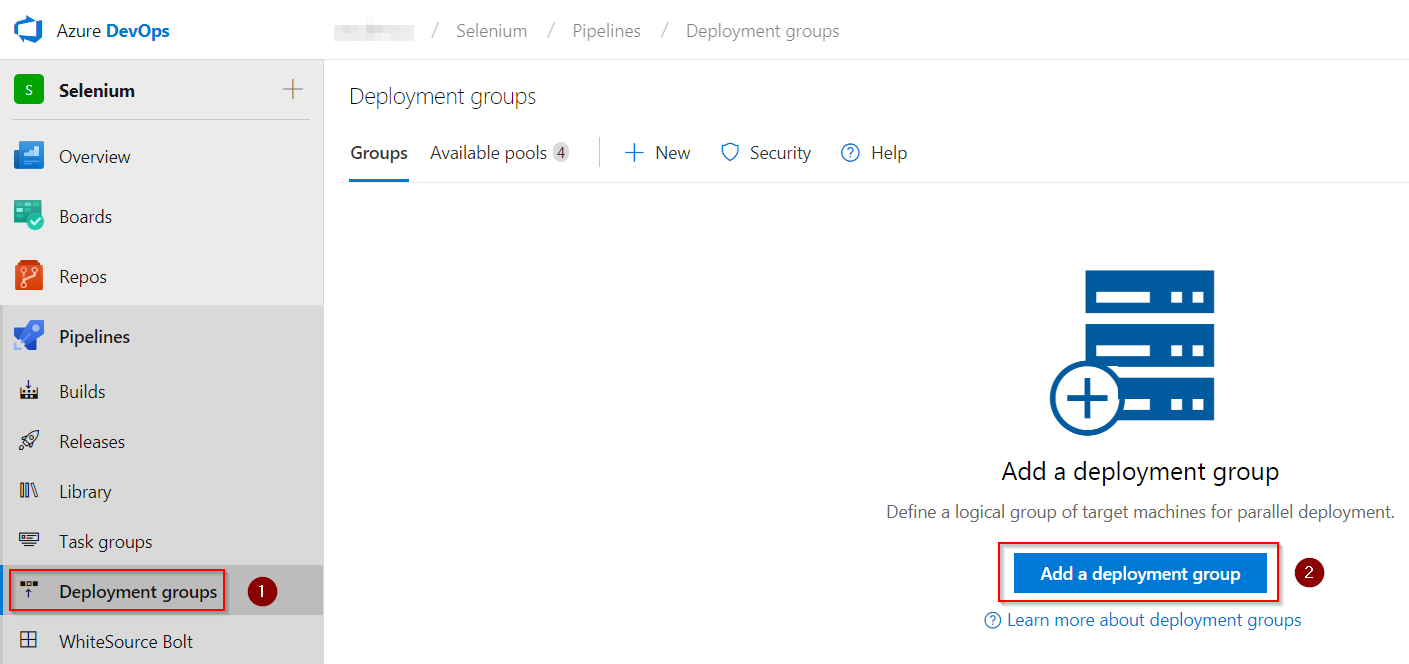
1. Use the [Azure DevOps Demo Generator](https://azuredevopsdemogenerator.azurewebsites.net/?Name=Selenium&TemplateId=77367) to provision the project on your Azure DevOps Organization.

## Exercise 1: Create a Deployment Group

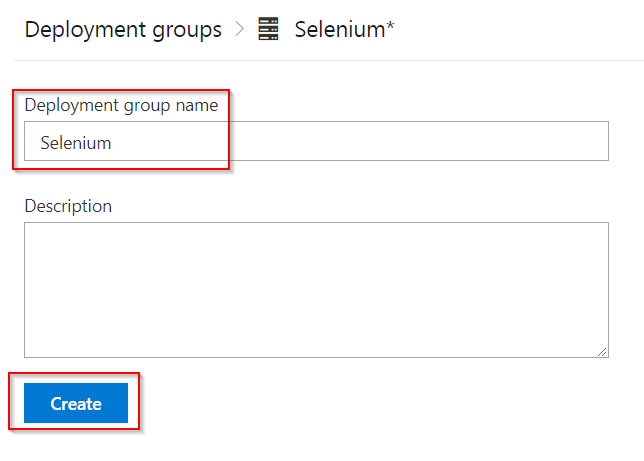
We will use Deployment Groups feature in Azure DevOps to deploy the application on a VM which was provisioned earlier to execute the Selenium test cases. [Deployment Groups](https://docs.microsoft.com/en-us/vsts/build-release/concepts/definitions/release/deployment-groups/) in Azure DevOps makes it easier to organize the servers that you use to host your app. A deployment group is a collection of machines with a Azure DevOps agent on each of them. Each machine interacts with Azure DevOps to coordinate the deployment of your applications.

We will also deploy the SQL database in the VM using Deployment Groups.

1. Go to **Deployment Groups** under **Pipelines** tab. Click on **Add a deployment group** .

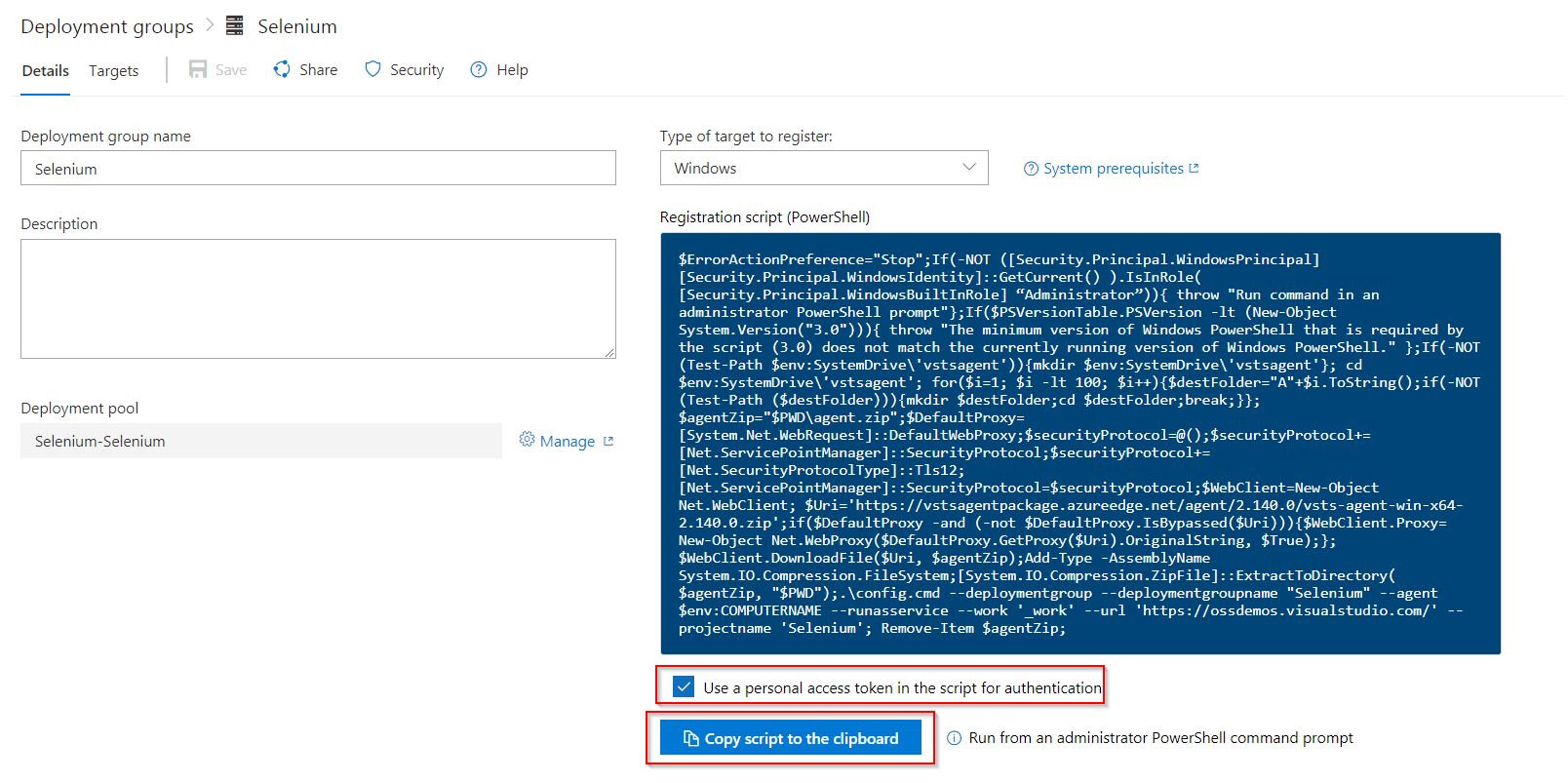


1. Provide deployment group name and click on **Create**.



This will generate a PowerShell script to associate the VM to this deployment group.

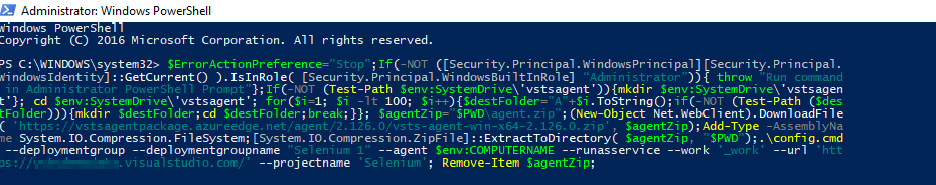
1. Select **Use a personal access token in the script for authentication** check the box, so that we will not have to provide a password every time the script is executed. Click on **Copy script to clipboard** to copy the script which will be used in the next exercise to associate the VM to the deployment group.



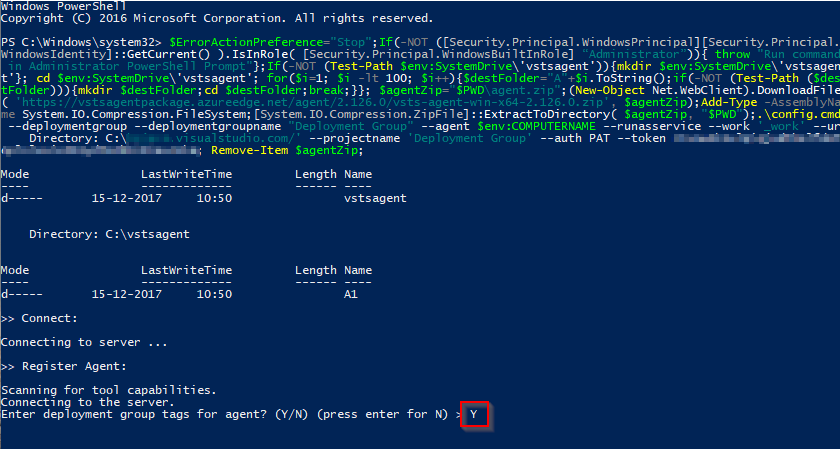
## Exercise 2: Associate the VM to Deployment Group

In this exercise, we will execute the **registration script** on the VM to associate with the deployment group.

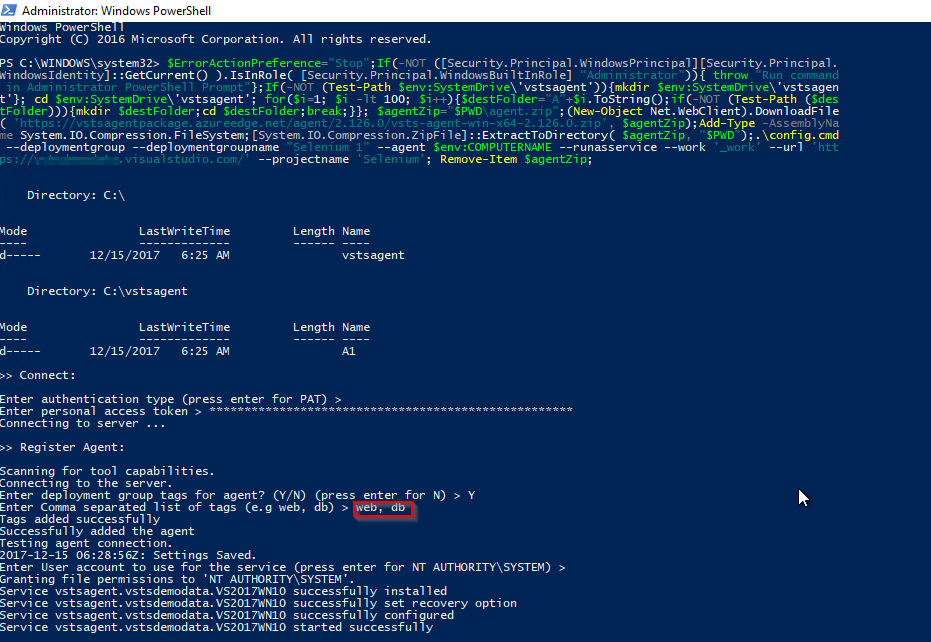
1. Login to the VM using [RDP](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/connect-logon) with the following credentials
   * **Username**: vmadmin
   * **Password**: P2ssw0rd@123
2. Open **Windows PowerShell** in **administrator** mode, paste the copied **Registration script** and hit **Enter**.



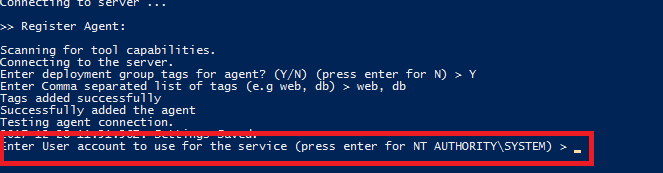
1. When the **Enter deployment group tags for agent? (Y/N) (press enter for N) >** message is prompted in the PowerShell window, type **Y** and hit enter.



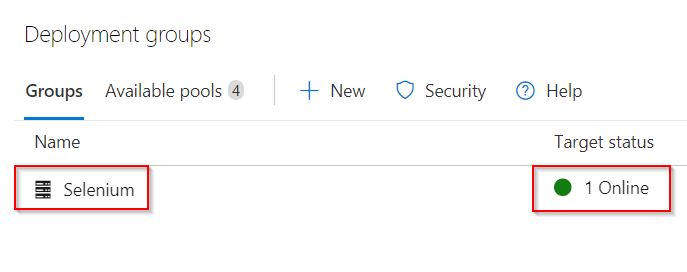
1. Enter **web, db** for the tags



1. When prompt -**Enter User account to use for the service (press enter for NT AUTHORITY\SYSTEM) >** is displayed, hit **Enter** to configure the service to run under **NT AUTHORITY\SYSTEM** account.



1. Refresh the Azure DevOps Deployment Groups page you will notice that your Deployment group is online.



## Exercise 3: Configure agent on the VM

Let us configure a **private** agent on this VM, since Selenium requires the agent to be run in **interactive** mode to execute the UI tests.

1. Go to the VM and open the folder **C:\VSTSwinAgent**.
2. Open a command prompt in **administrator mode**. Change the path to **C:\VSTSWinAgent** and type **Config.cmd** and hit **Enter**.
3. Provide the following details:
   * Enter server URL: Your Azure DevOps Organization URL
   * Authentication type: Press the **enter key** for **PAT** as the authentication type and paste the PAT in the next prompt.
   * Let us use the default options for the rest of the configuration. Press **Enter** for all prompts until the command execution completes.
   * Once the agent is registered, type **run.cmd** and hit **Enter** to start the agent.

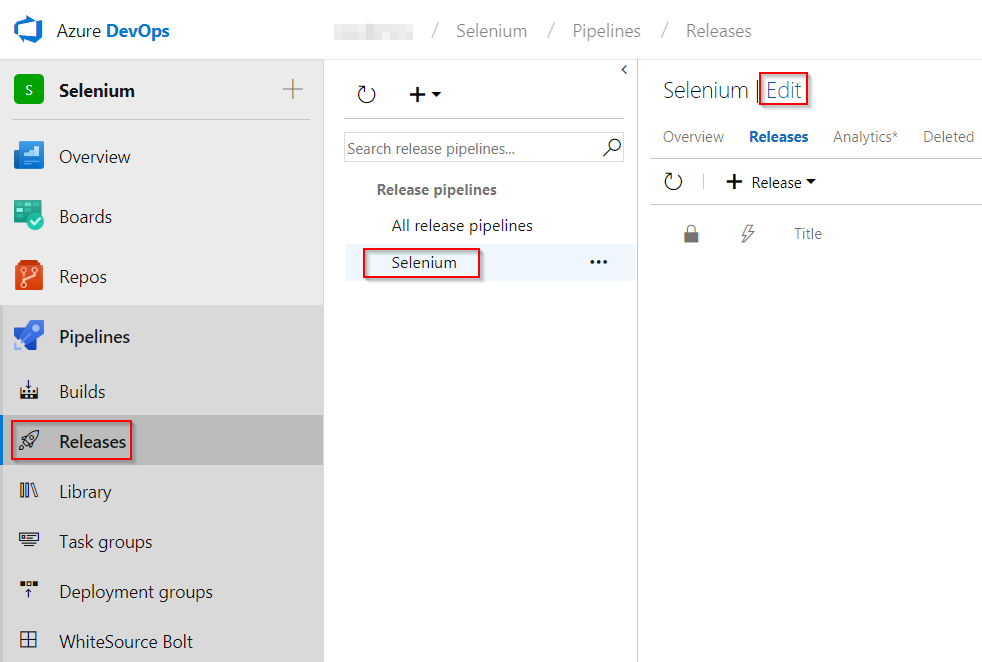
Click [here](https://docs.microsoft.com/en-us/vsts/build-release/actions/agents/v2-windows) for more information on how to configure the agent.



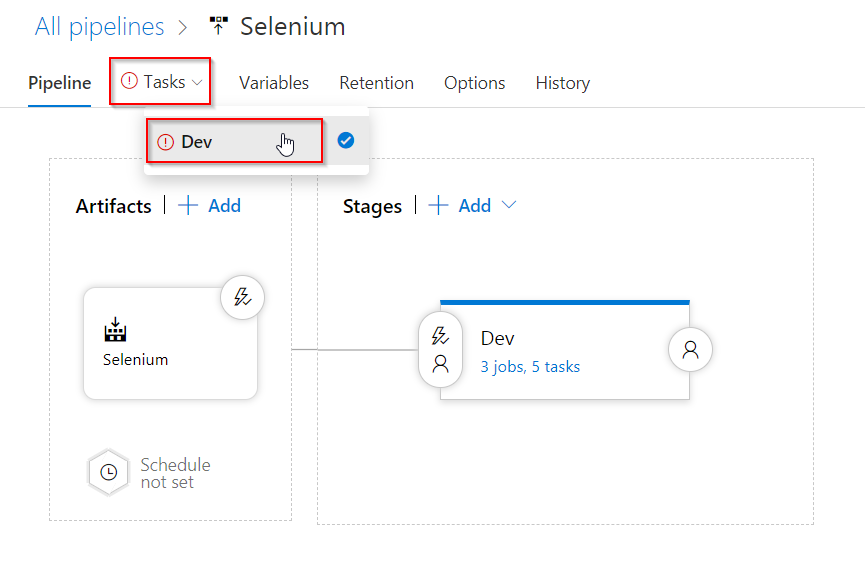
## Exercise 4: Configure Release Pipeline

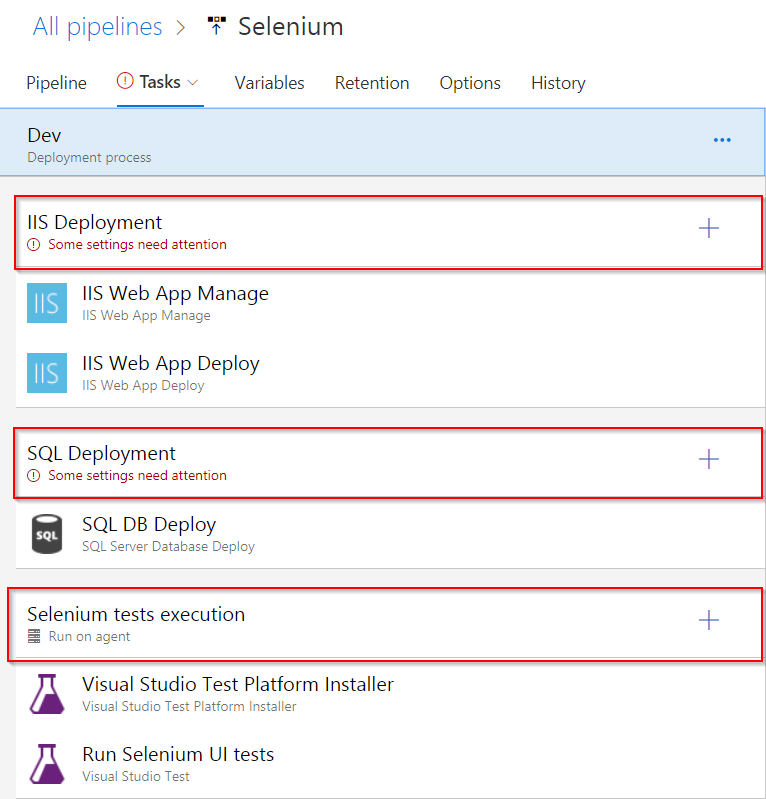
The target machine is available in the deployment group to deploy the applications and run selenium testcases. The release definition uses [**Phases**](https://docs.microsoft.com/en-us/vsts/build-release/concepts/process/phases) to deploy to target servers.

1. Go to **Releases** under **Pipelines** tab. Select **Selenium** release definition and click on **Edit**.



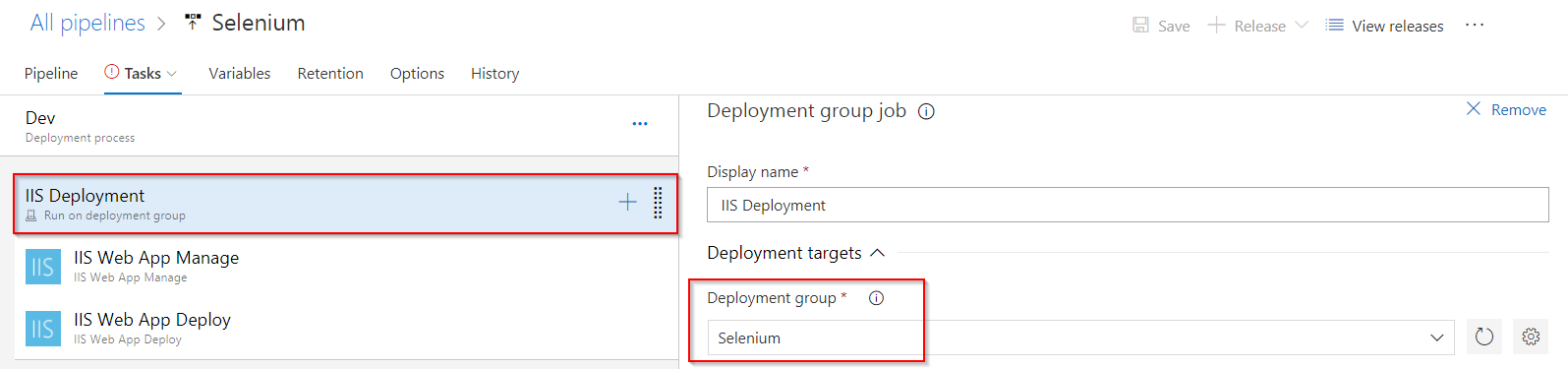
1. Open **Dev** environment to see the three deployment phases.



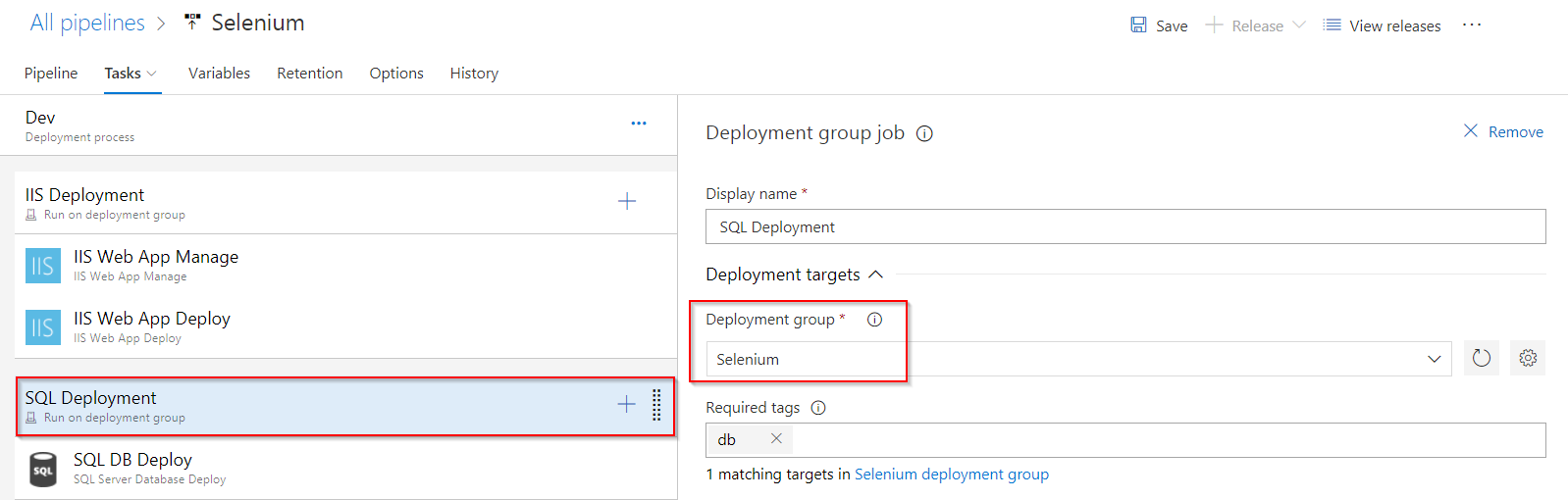


* + **IIS Deployment phase**: In this phase, we deploy application to the VM using the following tasks-
    - **IIS Web App Manage**: This task runs on the target machine registered with the Deployment Group. It creates a website and an Application Pool locally with the name **PartsUnlimited** running under the port **82** [**http://localhost:82**](http://localhost:82/)
    - **IIS Web App Deploy**: This task deploys the application to the IIS server using **Web Deploy**.
  + **Database deploy phase**: In this phase, we use [**SQL Server Database Deploy**](https://github.com/Microsoft/vsts-tasks/blob/master/Tasks/SqlDacpacDeploymentOnMachineGroup/README.md) task to deploy **[dacpac](https://docs.microsoft.com/en-us/sql/relational-databases/data-tier-applications/data-tier-applications" \t "_blank)** file to the DB server.
  + **Selenium tests execution**: Executing **UI testing** as part of the release process is a great way of detecting unexpected changes, and need not be difficult. Setting up automated browser based testing drives quality in your application, without having to do it manually. In this phase, we will execute Selenium tests on the deployed web application. The below tasks describes using Selenium to test the websites in the release pipeline.
    - **Visual Studio Test Platform Installer**: The [Visual Studio Test Platform Installer](https://docs.microsoft.com/en-us/azure/devops/pipelines/tasks/tool/vstest-platform-tool-installer?view=vsts) task will acquire the Microsoft test platform from nuget.org or a specified feed, and add it to the tools cache. It satisfies the ‘vstest’ demand and a subsequent Visual Studio Test task in a build or release pipeline can run without needing a full Visual Studio install on the agent machine.
    - **Run Selenium UI tests**: This [task](https://github.com/Microsoft/azure-pipelines-tasks/blob/master/Tasks/VsTestV2/README.md) uses **vstest.console.exe** to execute the selenium testcases on the agent machines.

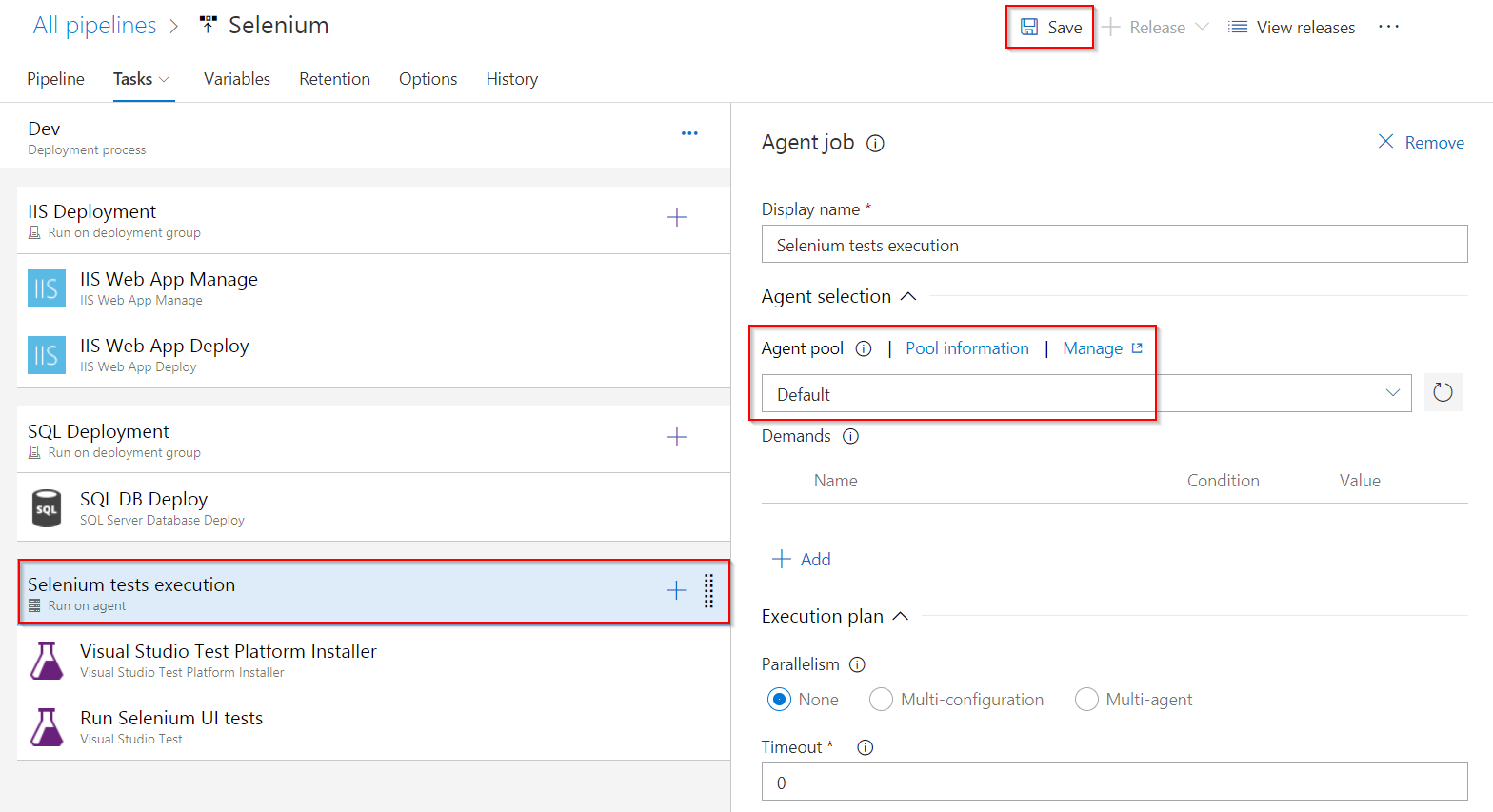
1. Click on **IIS Deployment** phase and select the Deployment Group which we have created in **Exercise 2**.



1. Repeat the above step for **SQL Deployment** phase



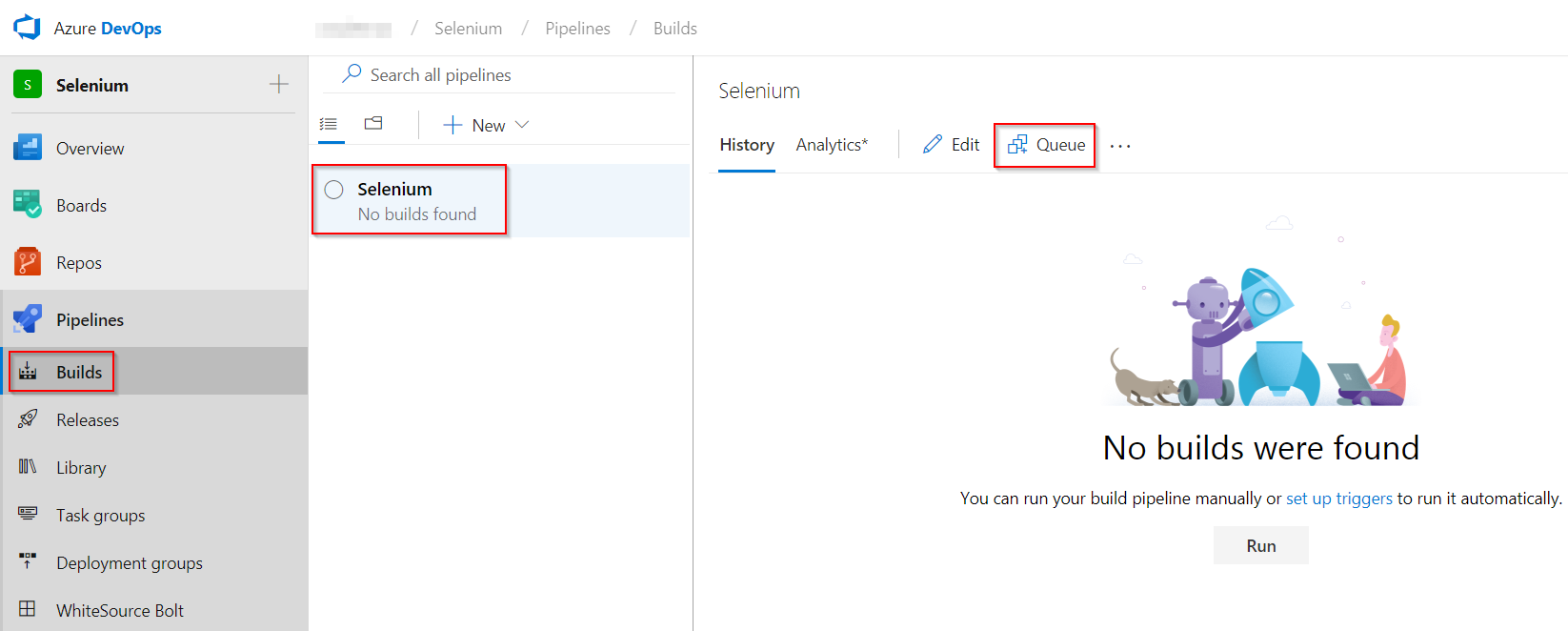
1. Click on **Selenium tests execution** phase and set Agent pool to **Default** then **Save** the changes.



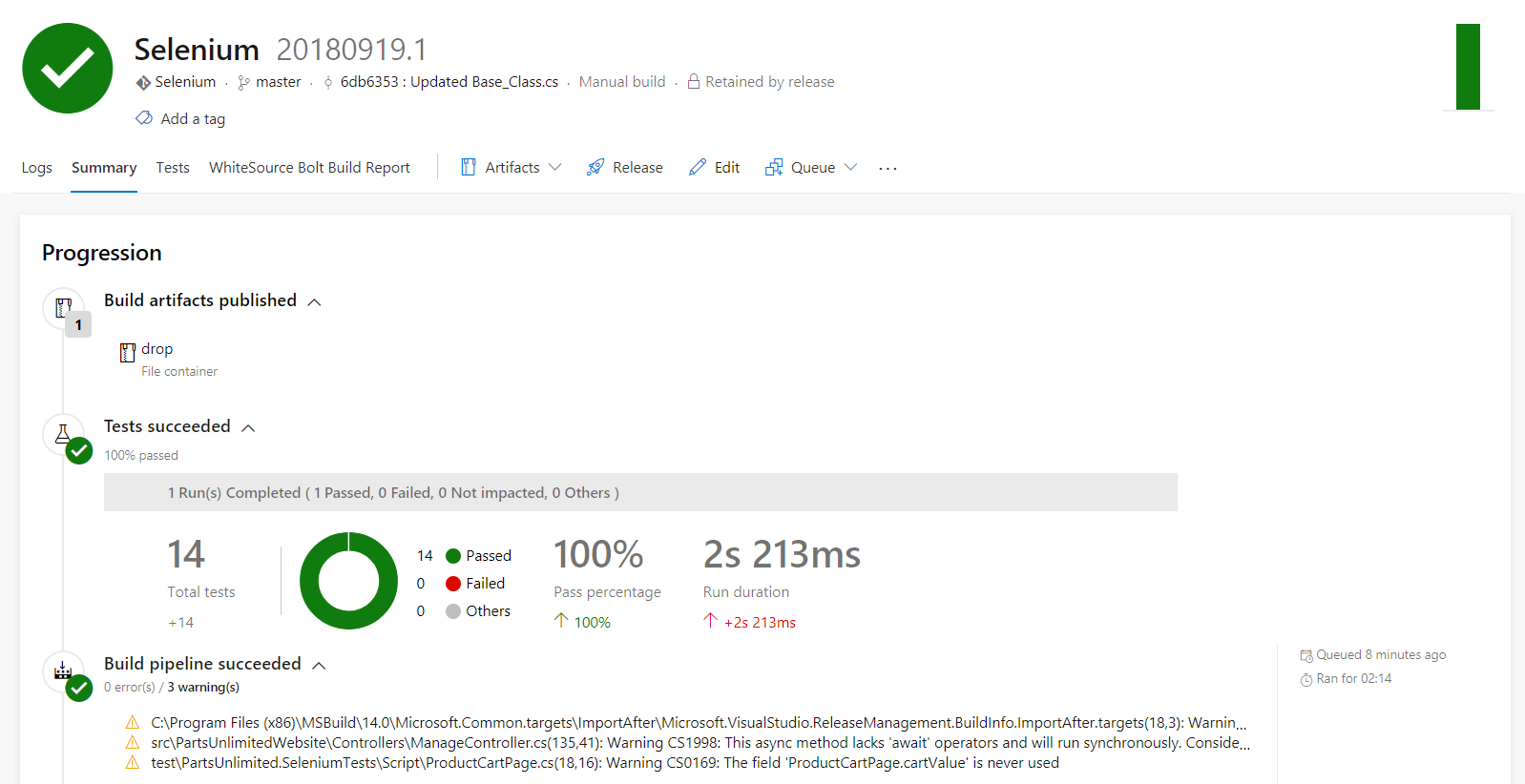
## Exercise 5: Trigger Build and Release

In this exercise, we will trigger the **Build** to compile Selenium C# scripts along with the Web application. The resulting binaries are copied to Azure VM and finally the selenium scripts are executed as part of the automated **Release**.

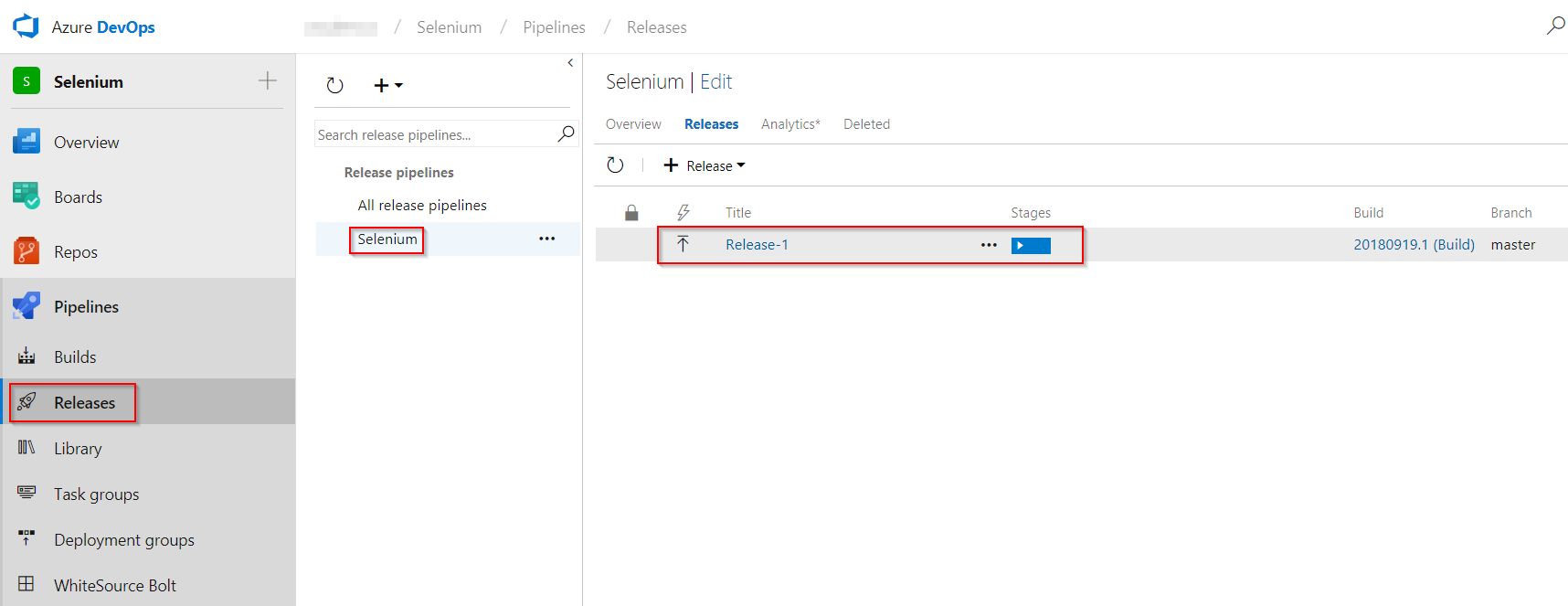
1. Navigate to **Builds** under **Pipelines** tab and queue the build.



1. This build will publish the test artifacts to Azure DevOps, which will be used in release.

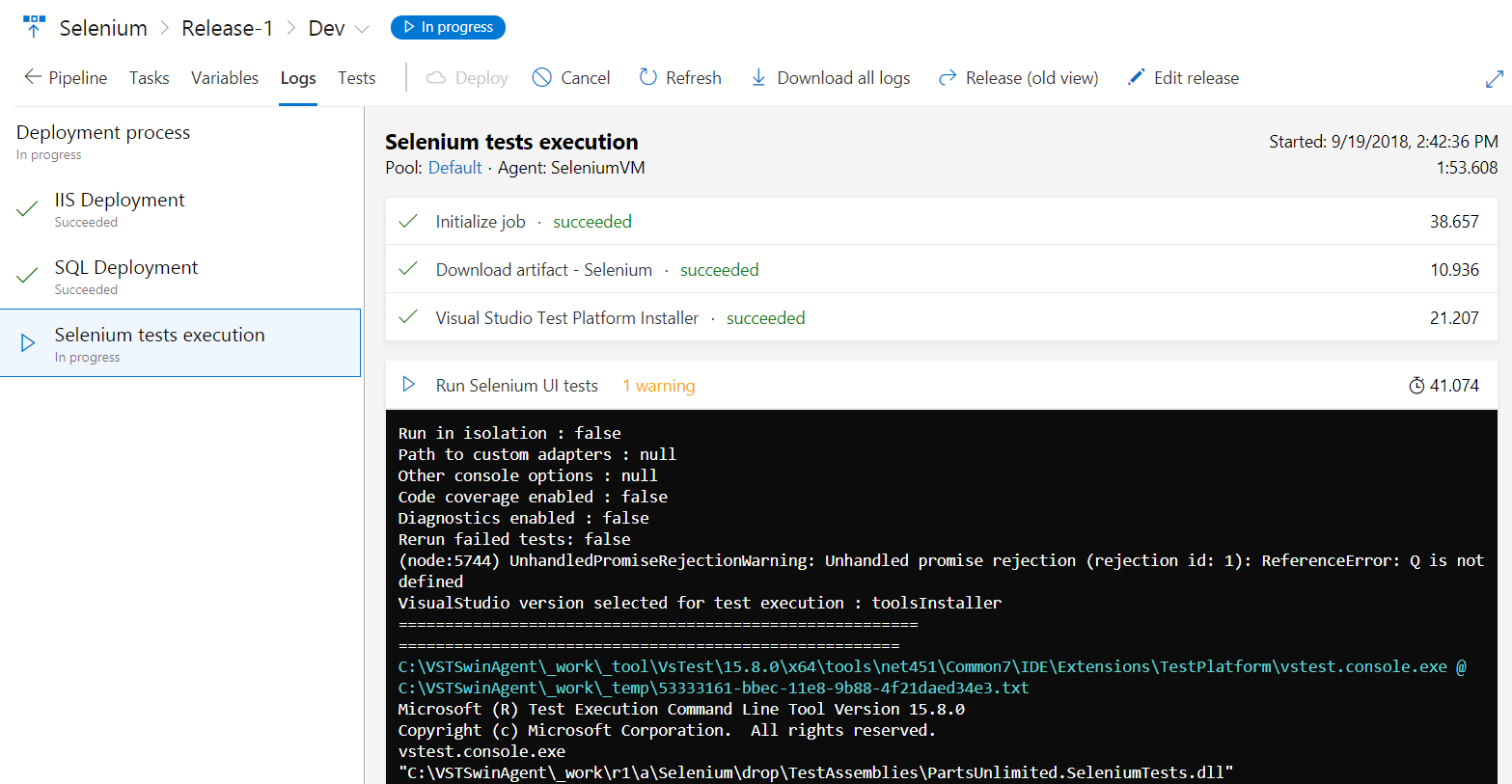


1. Once the build is successful, release will be triggered. Navigate to **Releases** tab to see the deployment in-progress.



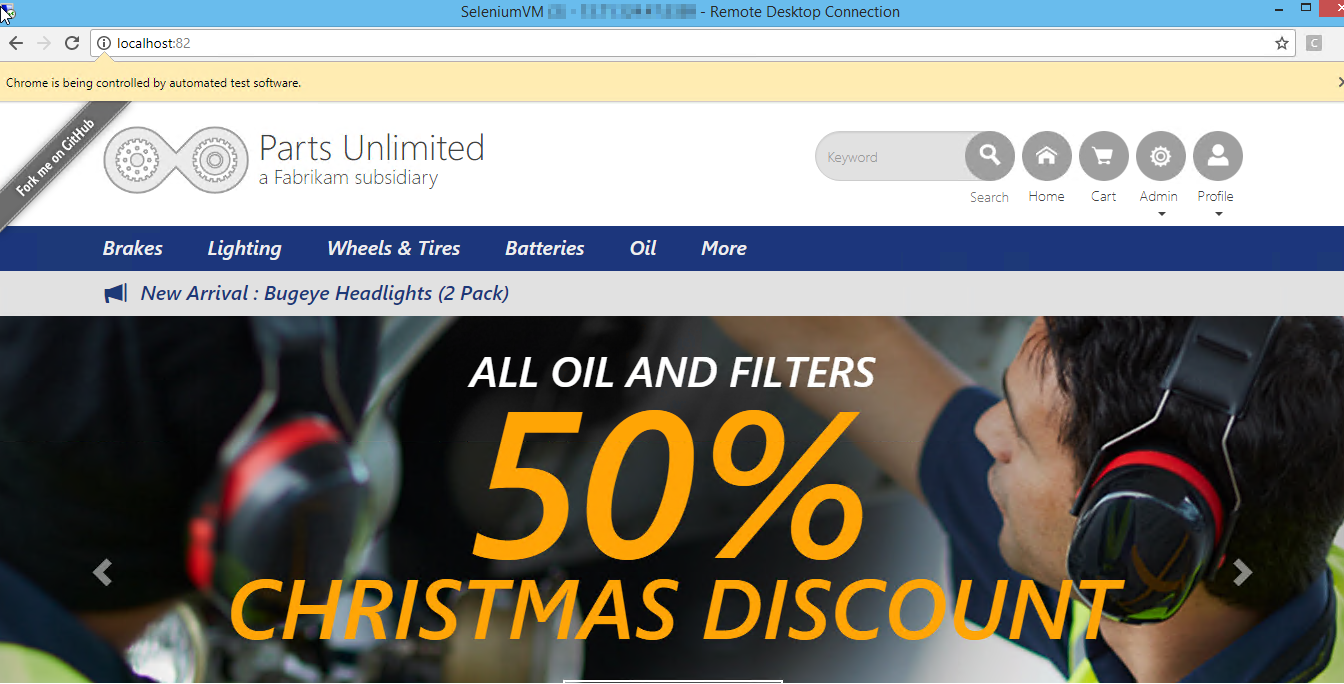
1. When **Selenium test execution** phase starts, connect back to the VM provisioned earlier to see UI tests execution.

**Note:** It takes approximately 8 minutes to deploy test agent for the first time on VM. Once this task is complete, you can connect to the VM to see actual test execution.

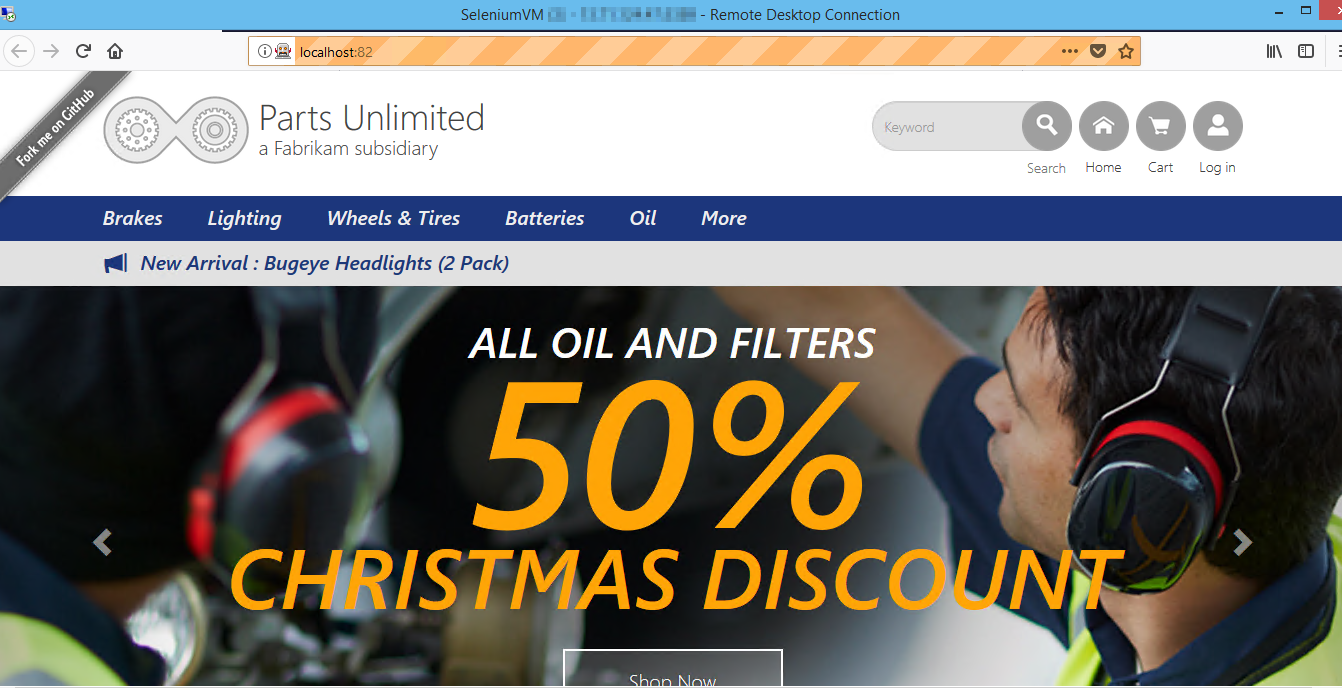


1. In this lab, we are executing **four** UI test scenarios configured to run on **Chrome** and **Firefox** browsers.

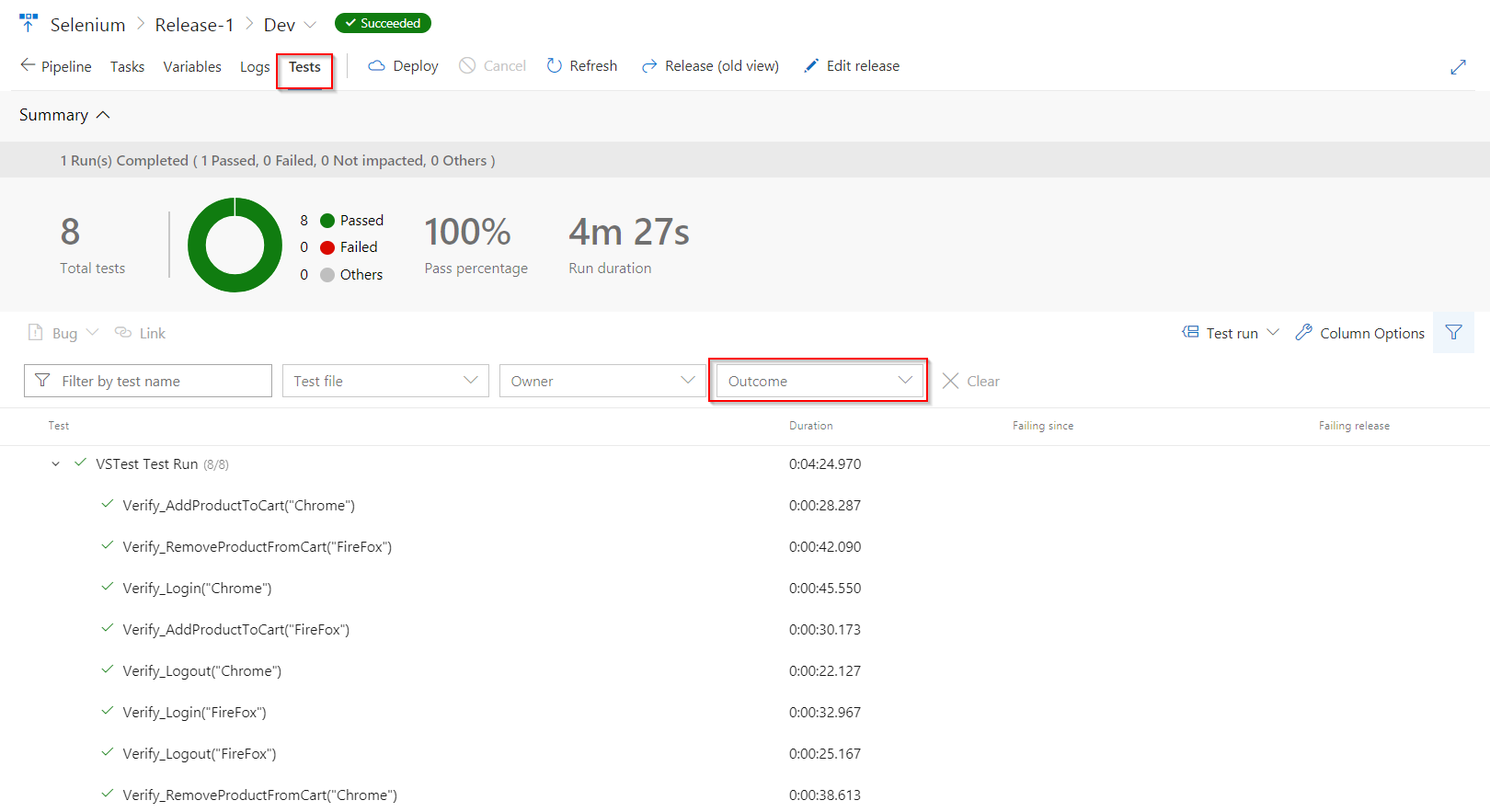
### *Tests running in Chrome*



### *Tests running in Firefox*



Once the release succeeds, click on the **Tests** tab to analyze the test results. Select the required filters from the dropdown in **Outcome** section to view the tests and their status.



## Summary

In this lab, you have learnt how to automate and execute Selenium UI test cases on different browsers on an Azure VM with Azure Pipelines.