**Deploy Java Application**

This lab will show how you will

* Create a new Azure App Service with a MySQL database server and configure the web app to use Apache Tomcat
* Use Azure App Service Task to deploy a WAR file

**Before you begin**

1. Use **MyShuttle** as a template to provision the new Azure DevOps project using the Azure DevOps Demo Generator

<https://azuredevopsdemogenerator.azurewebsites.net/?TemplateId=77371&Name=MyShuttle>

## **Exercise 1: Creating Azure Web App and MySQL database**

1. Launch the [Azure Cloud Shell](https://docs.microsoft.com/en-in/azure/cloud-shell/overview) from the portal. To deploy to a resource group, enter the following command

az group create --name MyResourceGroup --location westus

1. To create an App service plan

az appservice plan create --resource-group MyResourceGroup --name MyPlan --sku S1

1. Create the web app with a unique app name

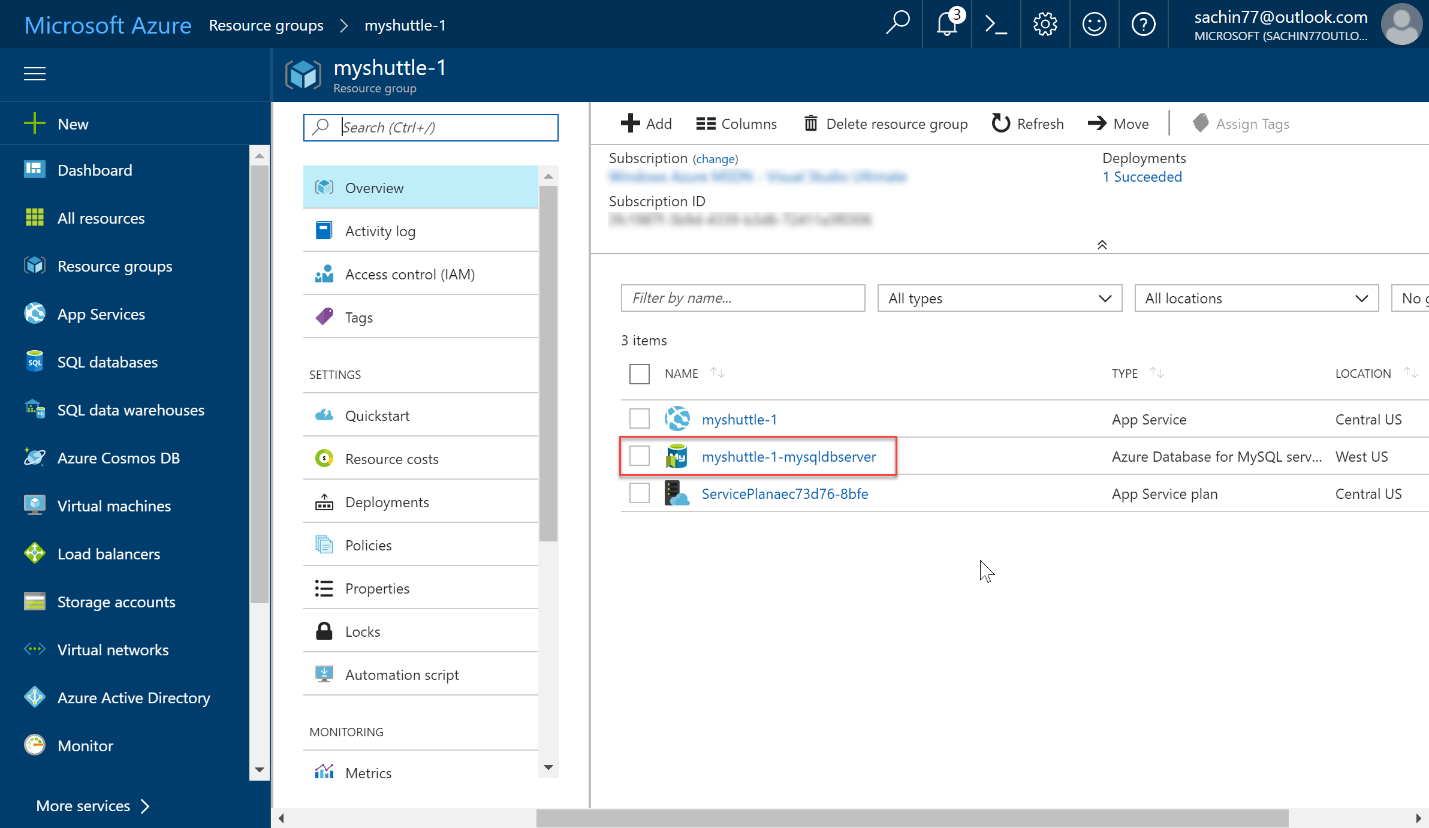
az webapp create --resource-group MyResourceGroup --plan MyPlan --name MyUniqueAppName

1. Finally, create the MySQL server with a unique server name.

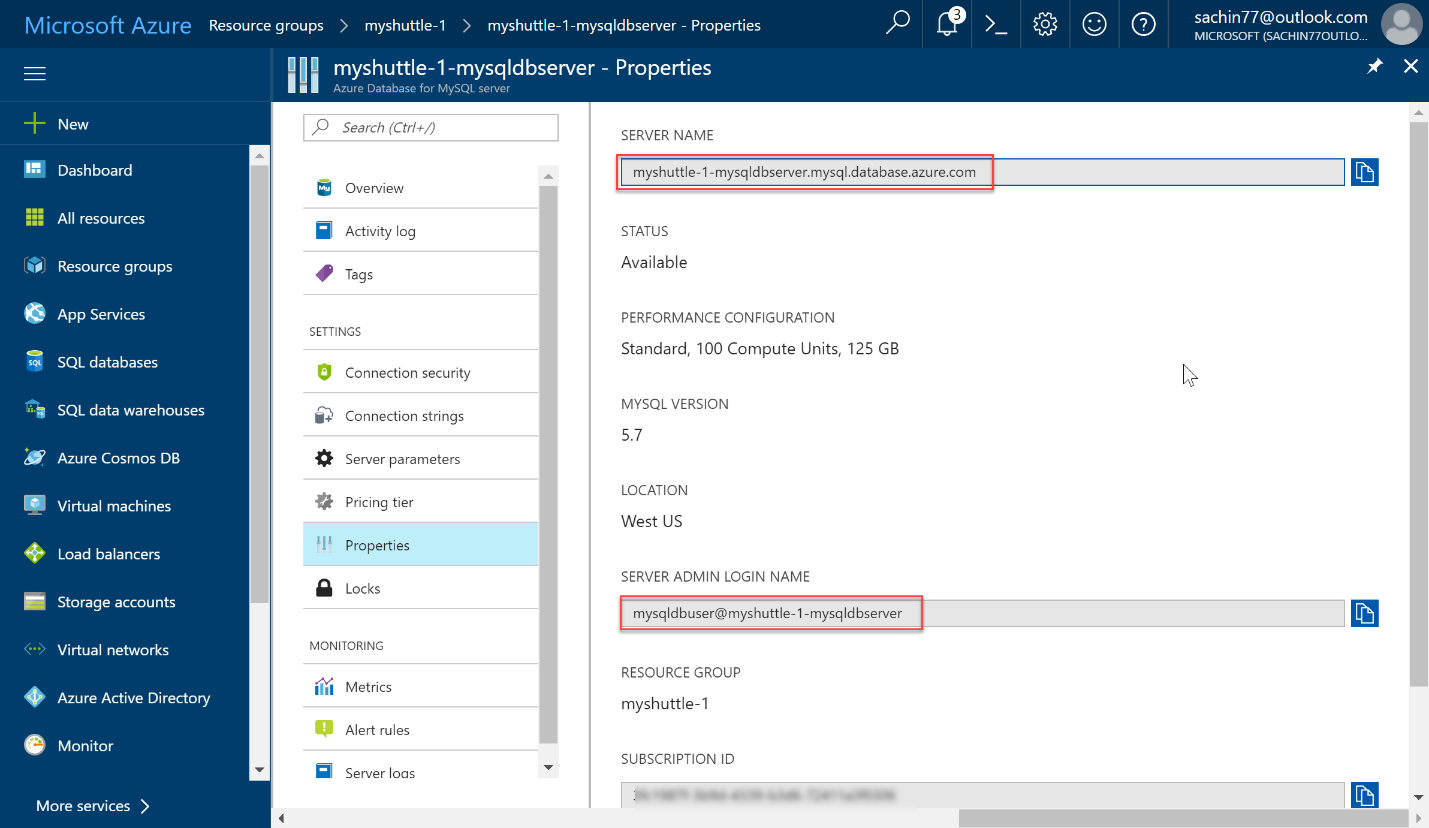
az mysql server create --resource-group MyResourceGroup --name mysqldbserver --admin-user mysqldbuser --admin-password P2ssw0rd@123 --sku-name GP\_Gen5\_2

**Important:** Enter a unique SQL server name. Since the Azure SQL Server name does not support **UPPER** / **Camel** casing naming conventions, use lowercase for the **DB Server Name** field value.

1. Navigate to the resource group that you have created. You should see a **Azure Database for MySQL server** provisioned. Select the database server.

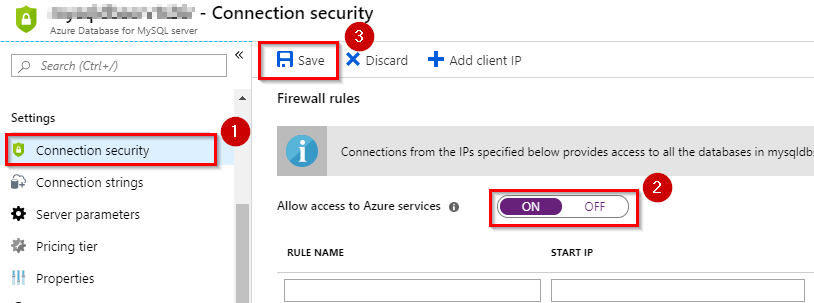


1. Select **Properties**. Save the **Server name** and **Server admin login name** to a notepad.



In this example, the server name is **myshuttle-1-mysqldbserver.mysql.database.azure.com** and the admin user name is **mysqldbuser@myshuttle-1-mysqldbserver**.

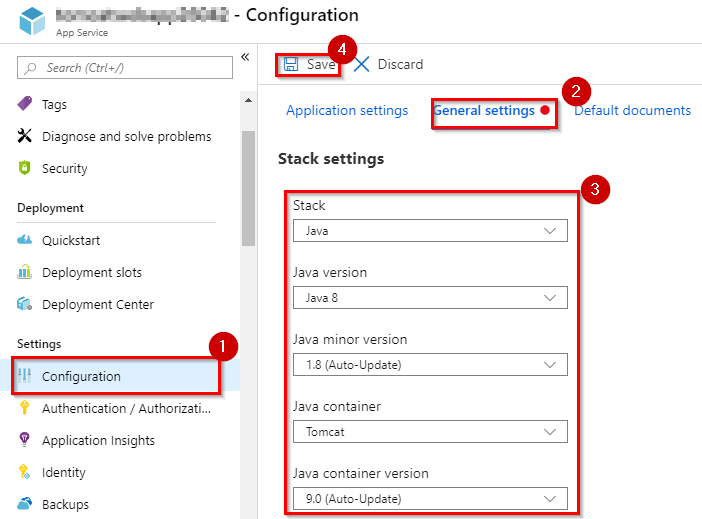
1. Select **Connection security**. Enable **Allow access to Azure services** toggle and **Save** the changes. This provides access to Azure services for all the databases in your MySQL server.



## **Exercise 2: Updating the App Settings for the Web App**

Next, navigate to the Web app that you have created. As you are deploying a Java application, you need to change the web app’s web container to Apache Tomcat.

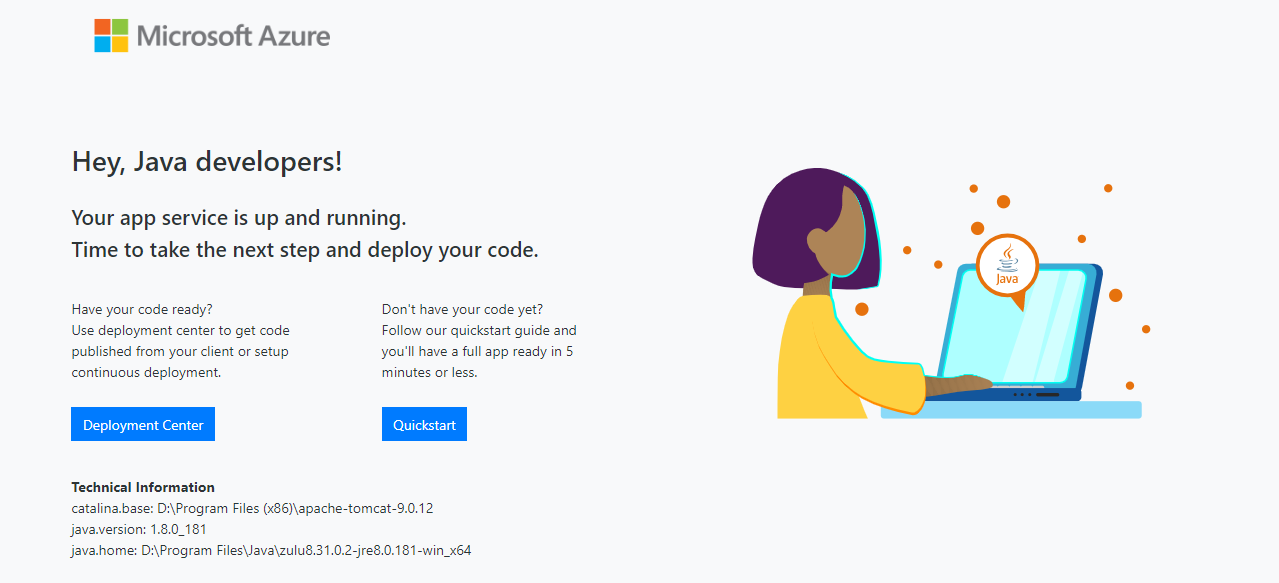
1. Select **Configuration**. Set the **Stack settings** as shown in below image and click **Save**.



1. Select **Overview** and click **Browse**.

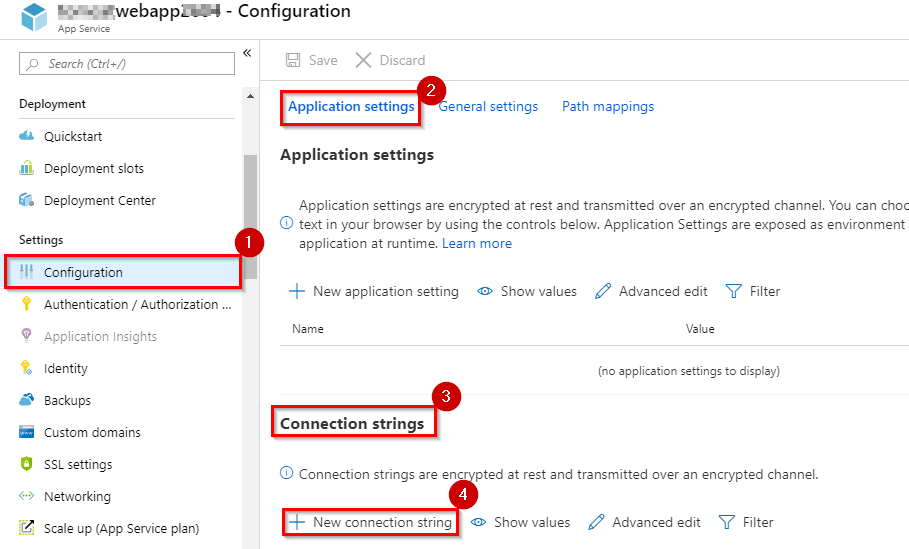


The web page will look like the below image.



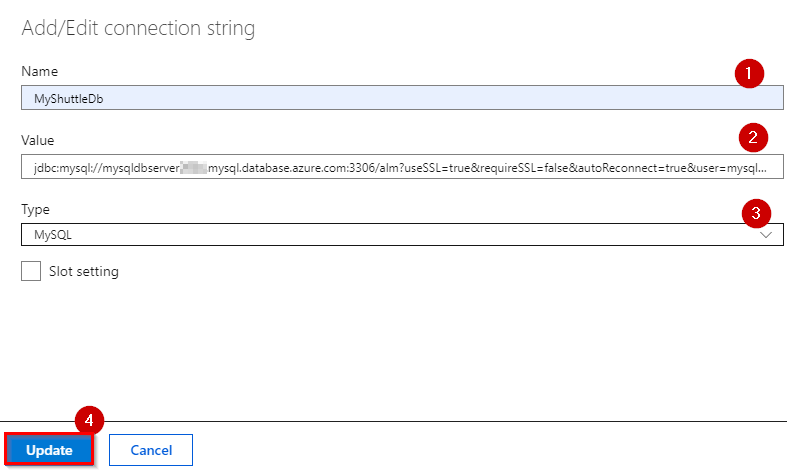
Next, you need to update the **connection strings** for the web app to connect to the database correctly. There are multiple ways you can do this - but for the purpose of this lab, you will take a simple approach by updating it directly on the Azure portal.

1. From the Azure portal, select the Web app you provisioned. Go to **Configuration | Application settings | Connection strings** and click on **+ New connection string**.



1. In **Add/Edit connection string** window, add a new **MySQL** connection string with **MyShuttleDb** as the name, paste the following string for the value and replace **MySQL Server Name**, **your user name** and **your password** with the appropriate values. Click **Update**.

jdbc:mysql://{MySQL Server Name}:3306/alm?useSSL=true&requireSSL=false&autoReconnect=true&user={your user name}&password={your password}



* + MySQL Server Name : Value that you copied previously from the MySQL server Properties.
  + your user name : Value that you copied previously from the MySQL server Properties.
  + your password : Value that you provided during the creation of MySQL database server.

1. Click on **Save** to save the connection string.

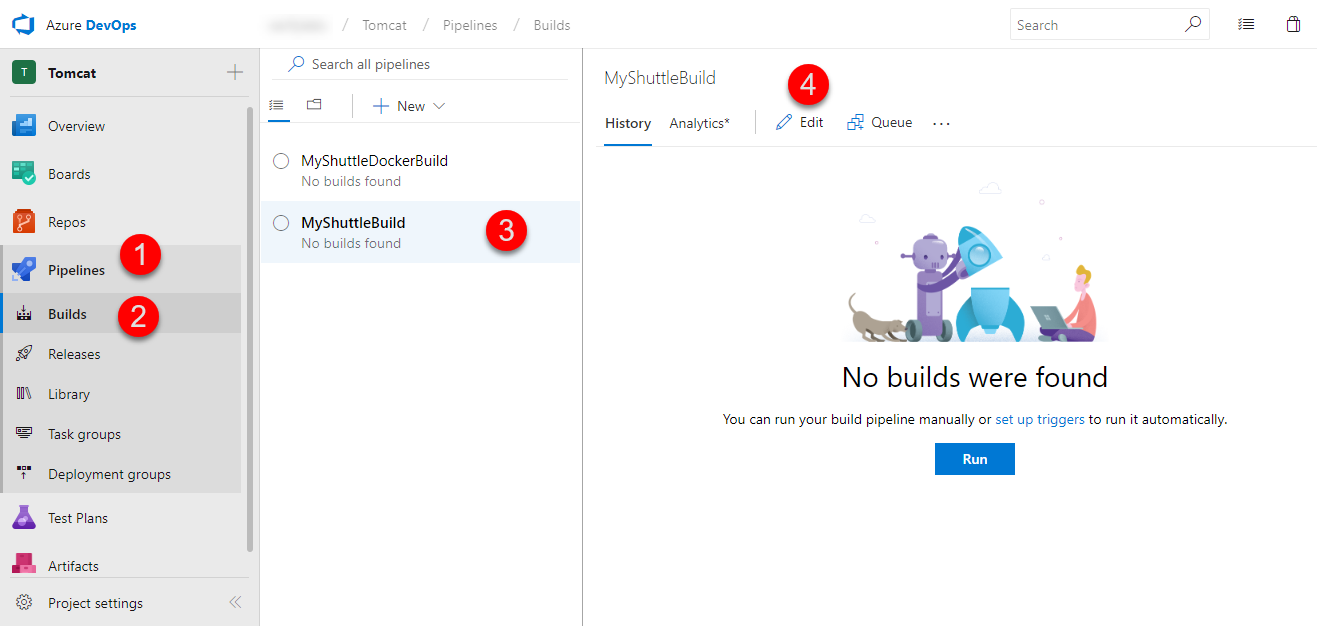
**Note:** Connection Strings configured here will be available as environment variables, prefixed with connection type for Java apps (also for PHP, Python and Node apps). In the **DataAccess.java** file under **src/main/java/com/microsoft/example** folder, we retrieve the connection string using the following code

String conStr = System.getenv("MYSQLCONNSTR\_MyShuttleDb");

You have now setup and configured all the resources that is needed to deploy and run the MyShuttle application.

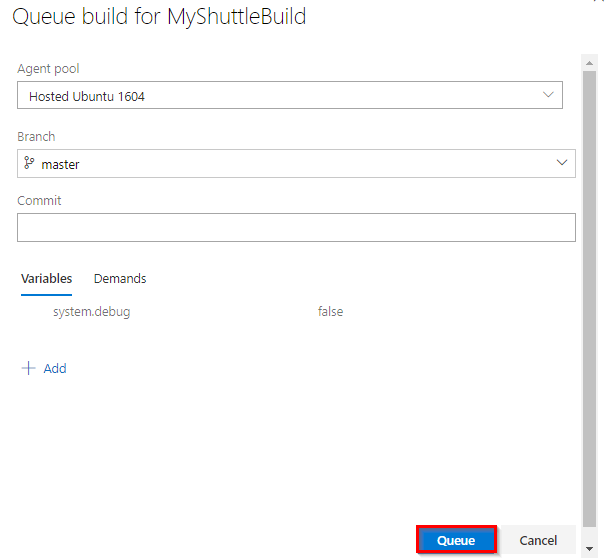
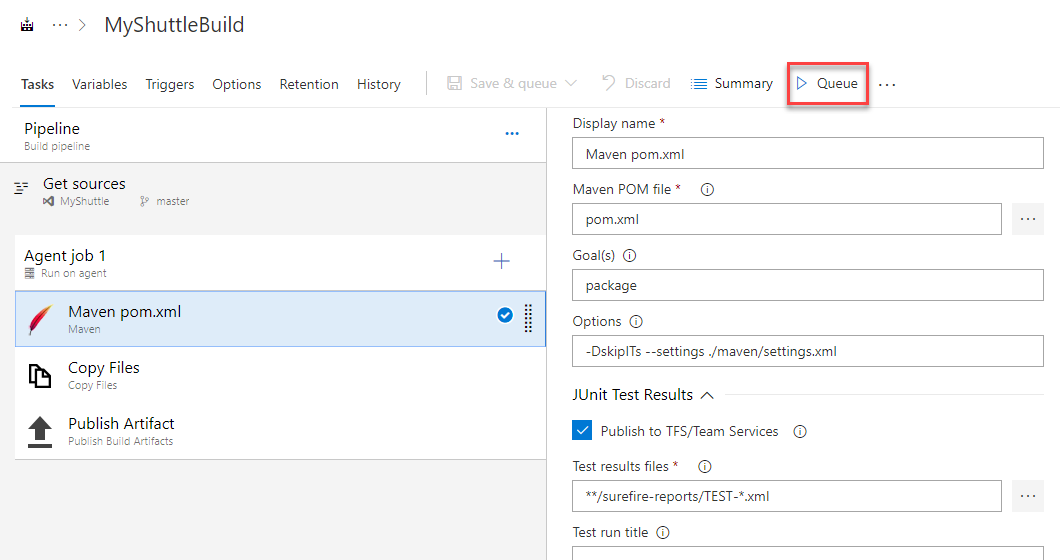
## **Exercise 3: Deploy the changes to Web App**

1. Select **Pipelines** and then, **Builds**. Choose the build **MyShuttleBuild** and click **Edit Pipeline** to view the build definition.



The lab uses the standard **Maven** build template to compile the code, copy and publish the resulting artifacts for deployment. An additional file which is copied here is the CreateMYSQLDB.sql file which creates a MySQL database and inserts a few records into it during the deployment.

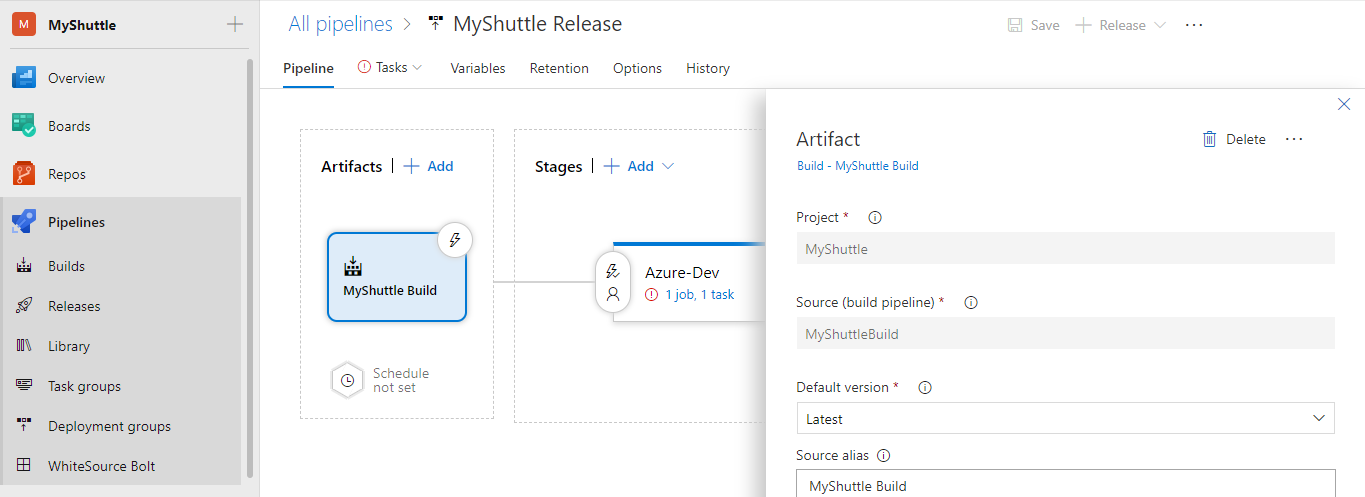
1. Click **Queue** to trigger the build and wait for the build to complete.



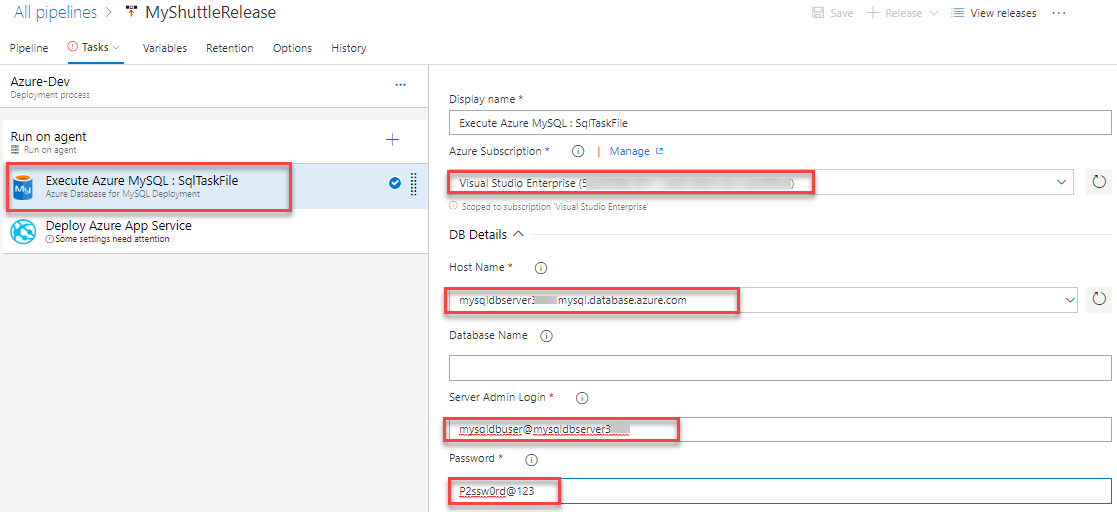
1. Once the build succeeds, select **Releases** under **Pipelines**.
2. Select **MyShuttle Release** and click **Edit** to open the release definition.

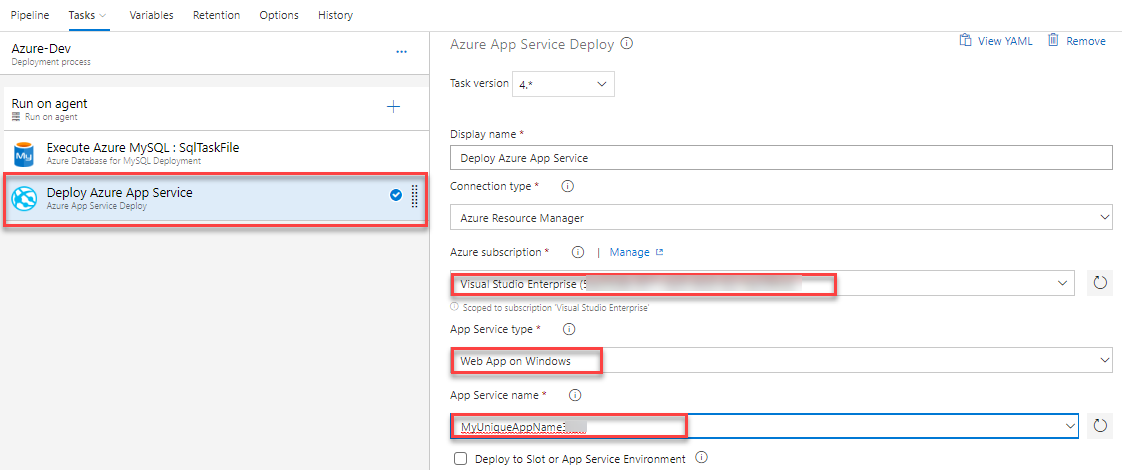


1. Make sure the artifact is pointing to the **Build** artifact as shown below. If you are following this lab from Jenkins hands-on-lab, make sure the artifact is pointing to Jenkins.



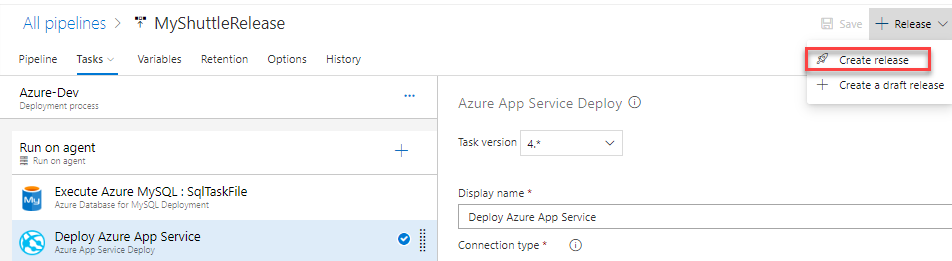
1. Click **Tasks**. Select **Execute Azure MySQL : SqlTaskFile** task and provide the following details.
   * Azure Subscription Details : Select the appropriate subscription. Click **Authorize** and login to your Azure subscription in the pop-up window.
   * Host Name : Select the **MySQL Database server** host name that was created.
   * Server Admin Login : Provide the **Server admin login name** that you noted down previously.
   * Password : Provide the password that you created during the creation of MySQL server in the Azure portal.



1. Select the **Deploy Azure App Service** task, choose the **Azure subscription** details and select **App Service name** from the dropdown. 

**Note:** We are using the **Deploy Azure App Service** task. This task is used to update Azure App Service to deploy Web Apps and WebJobs to Azure. The task works on cross platform agents running Windows, Linux or Mac and uses the underlying deployment technologies of Web Deploy and Kudu. The task works for ASP.NET, ASP.NET Core 1 and Node.js based web applications. Note that this task works with Azure Resource Manager APIs only.

1. Click on **Save** and then **+Release | Create Release** to start a new release



1. Wait for the release to complete. Then navigate to the Web App and select the **URL** from the overview blade. Add **/myshuttledev** context to the URL. For instance - <http://myshuttle1.azurewebsites.net/myshuttledev>
2. Select **Login** and try logging in to the site with any one of the following credentials.

| **Username** | **Password** |
| --- | --- |
| barney | barneypassword |
| fred | fredpassword |

1. If your database was setup correctly and the connection parameters are valid, you should be able to login to the portal.

**Warning:** If you encounter an error with **The specified CGI application encountered an error and the server terminated the process** message, check whether you have entered the app settings variables and values (for the database connection) correctly

