IBM Integration bus and MQ container via IBM Bluemix Continuous delivery

Introduction

This tutorial and associated video shows how to deploy a Docker container containing IBM Integration Bus (IIB) V10 and IBM MQ from scratch, using the IBM Bluemix DevOps Continuous Delivery feature.

The tutorial takes the artefacts from a GitHub project (https://github.com/peterajessup/IIB-MQ) that provides the Docker assets to build the container.

This GitHub project is a merge of two independent projects https://github.com/ot4i/iib-docker and https://github.com/ibm-messaging/mq-docker.

The container additionally shows how to configure IIB with an ODBC data source, and to enable the integration node for callable flows.

The following sections provide a step-by-step set of instructions that guide you through the process.

Learning Objectives

- Understand the ease of use of Bluemix Continuous Delivery tooling
- Familiarisation with IIB and MQ in a Docker Container

Time Required

This tutorial should take no longer than 60 minutes.

Skill Level

Any

Audience

IIB and MQ Developers

Requirements

This tutorial assumes you have an active Bluemix account or a Bluemix trial account.

Expected results

A running container is created in Bluemix with an active instance of IIB and MQ.

Procedure

Pre-work:

As a one-time setup you may need to create a namespace for your Bluemix container registry.

Note : If you have already created a container previously in Bluemix you can skip to Step 1.

From the Bluemix dashboard select ‘catalogue’, then ‘Containers’ from the Apps Menu. (See Figure 1.)

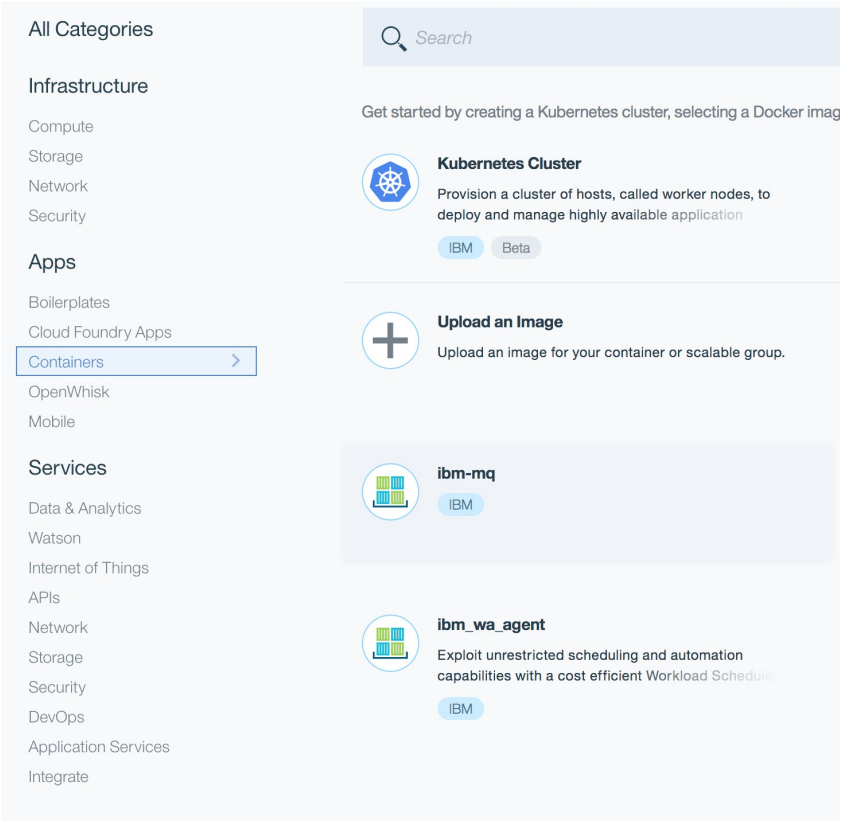

Figure 1: Bluemix Containers

Click the **ibm-mq** container tile; a pop-up will appear to prompt you for a registry namespace. (See Figure 2.)

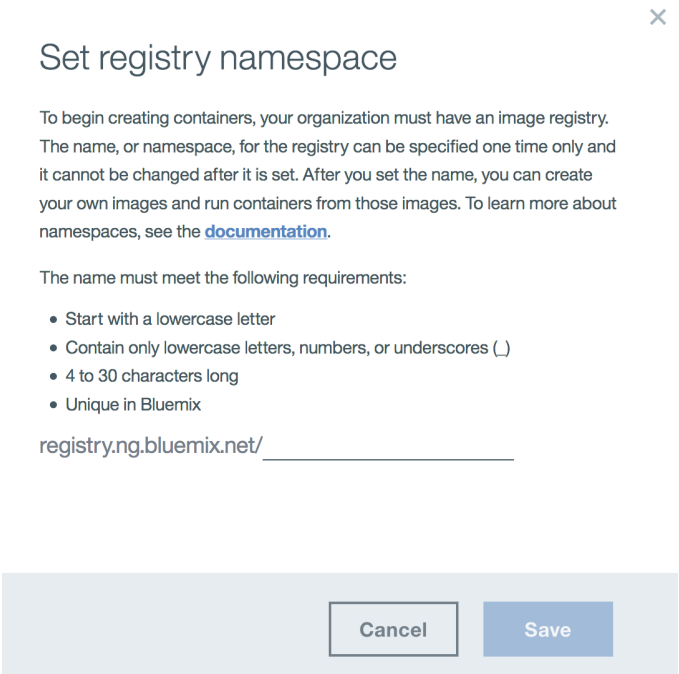

Figure 2: Bluemix Organisation Container namespace

Enter a string to uniquely identify your registry, then click **Save**.

Select the dashboard from the left hand side menu to cancel out of this screen, because we only want to create the namespace for now.

Log out of Bluemix.

Step 1.

Log into the Bluemix environment, and from the Catalog search for the DevOps category.

Select the Continuous Delivery service under DevOps. This is a new service that provides tools to rapidly develop and deploy applications on Bluemix.

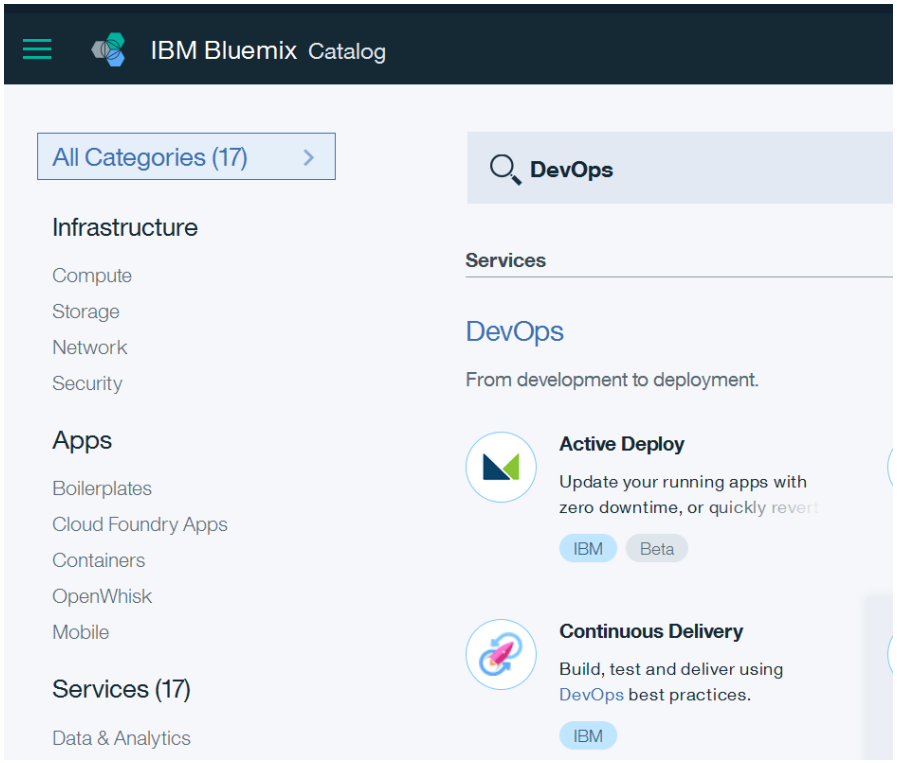

Figure 3: Bluemix Devops Continuous Delivery

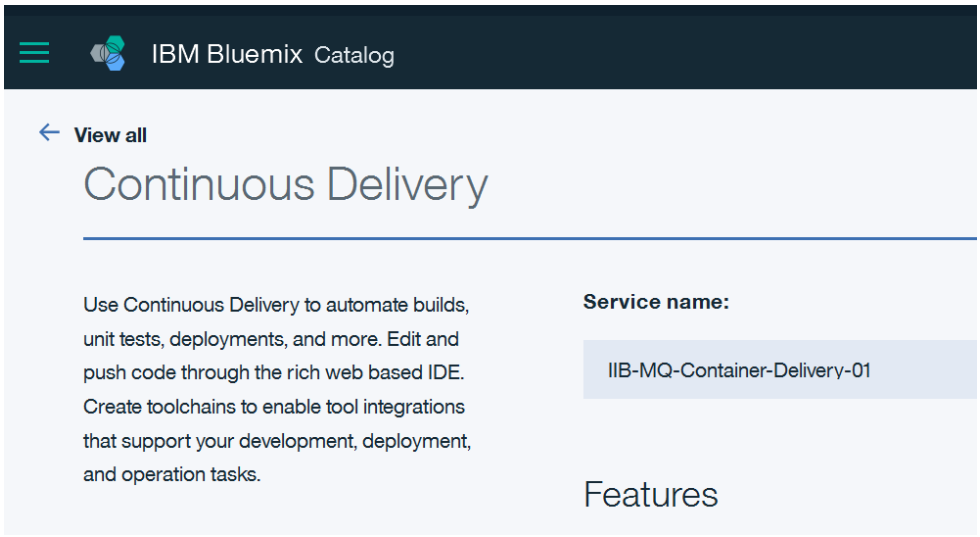

Figure 4: Continuous Delivery Service Name

Give the new service a name, and then select 'Start from a Toolchain template'. This provides a comprehensive list of templates which accelerate tool chain configuration.

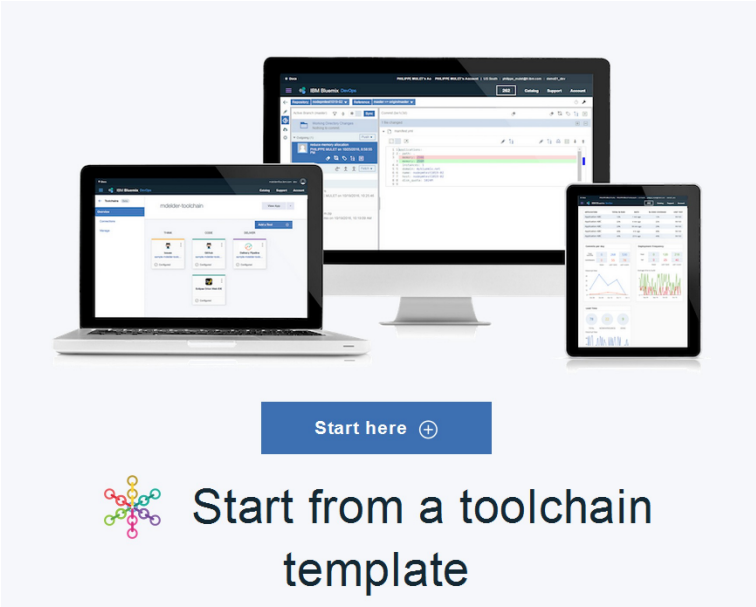

Figure 5: Toolchain template

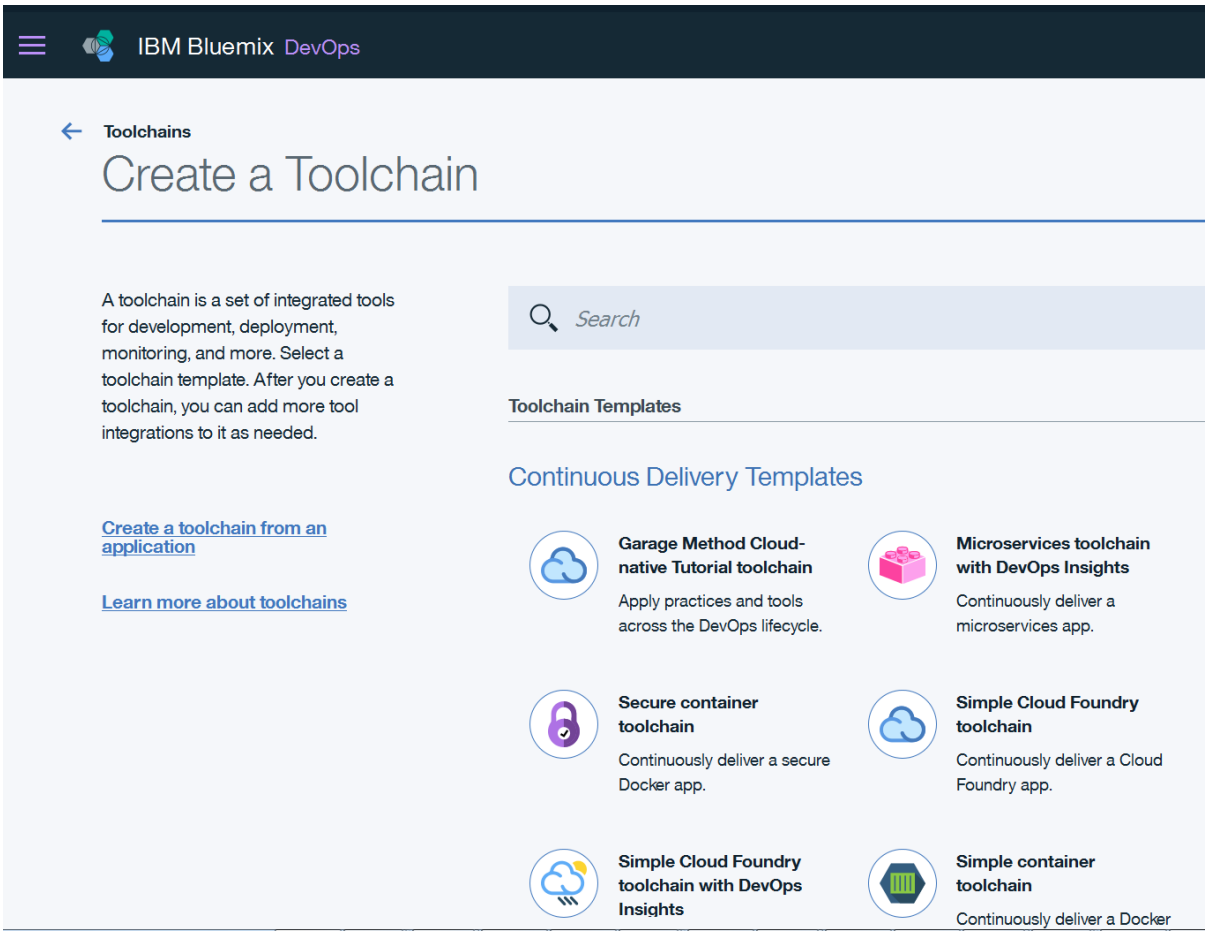

Figure 6: Creating a toolchain from a template

Step 2.

Select the Simple Container toolchain, and then give the toolchain a name.

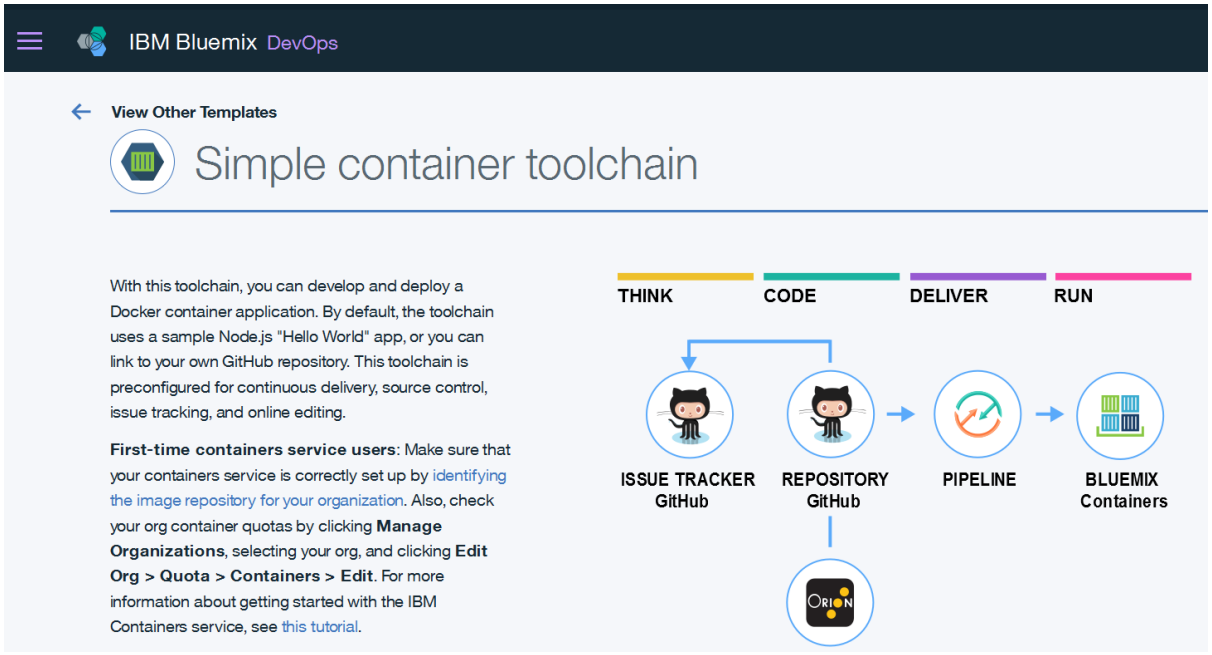

Figure 7: Simple Container Toolchain

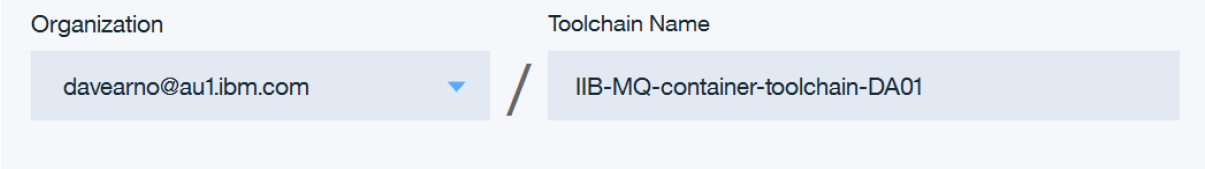

Figure 8: GitHub properties

For each of the components (GitHub, Eclipse Orion Web IDE, and Delivery Pipeline) we can now configure the specific parameters for each to customise the behaviour of the tool chain.

Step 3.

Click the GitHub tile. If this is the first time you have created a toolchain in your account, you may see the following prompt below the GitHub component.

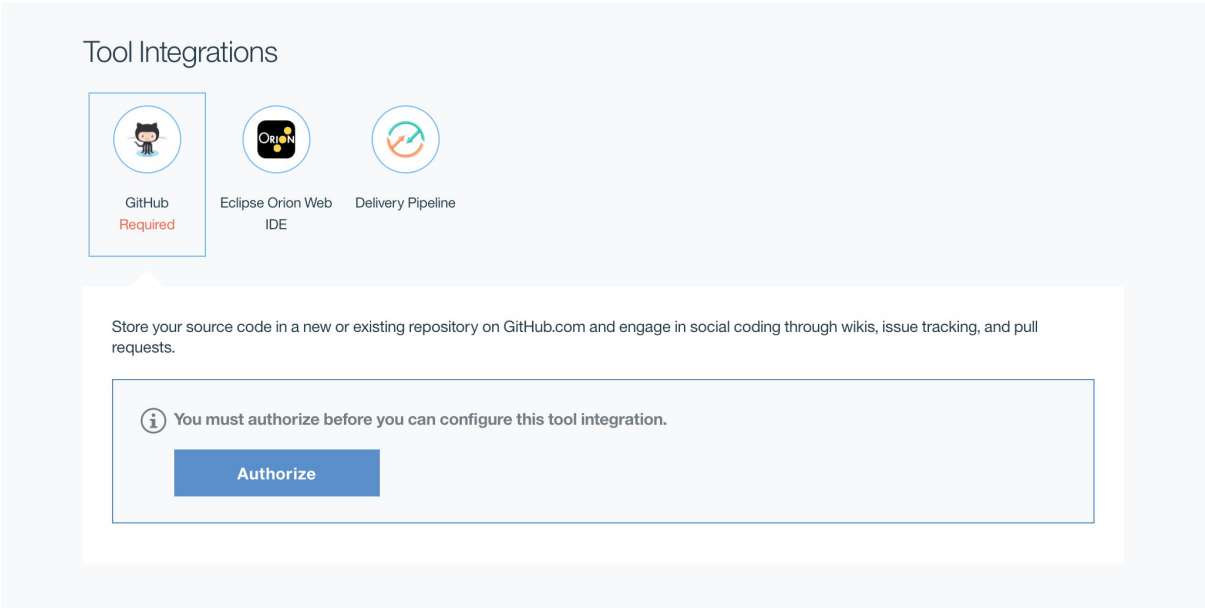

Figure 9: GitHub authorisation

Click **Authorize**. If you are already logged into GitHub in the browser, Bluemix uses your credentials; otherwise you are prompted to log into GitHub with your credentials.

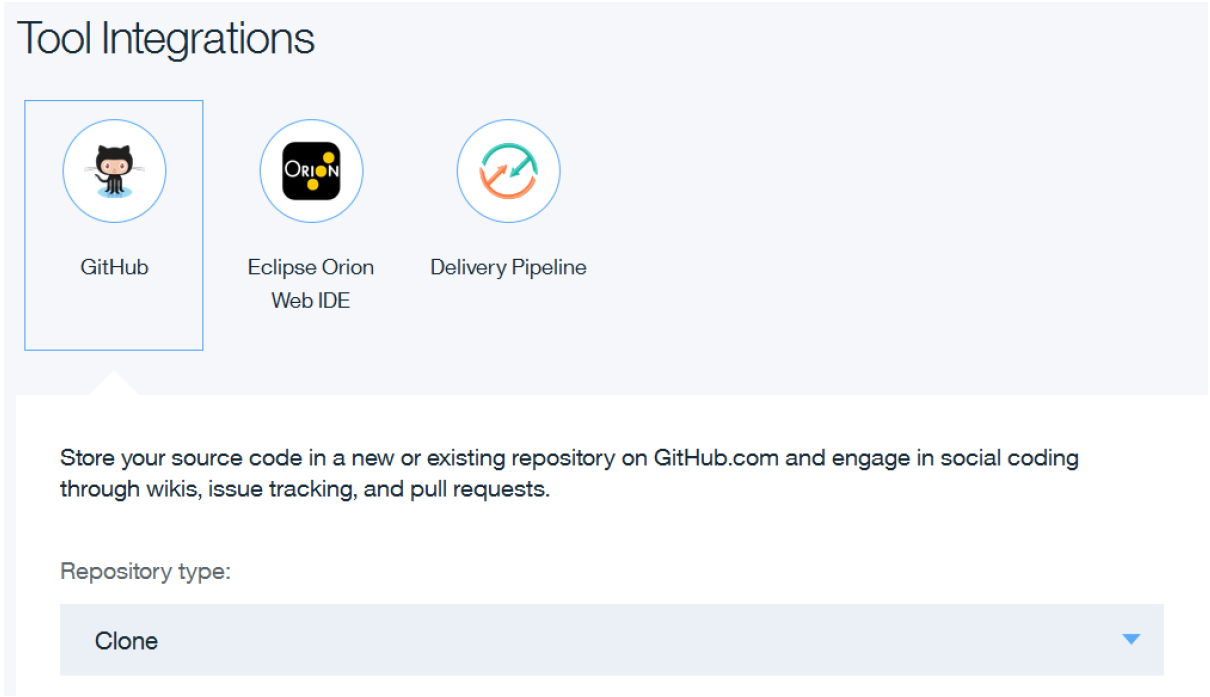

Figure 10: GitHub repository configuration

For the repository type, select **Clone**.

For the repository name, chose something meaningful. This will be the repository in your own GitHub account where the artefacts will be cloned.

For the source repository URL, use https://github.com/peterajessup/IIB-MQ.

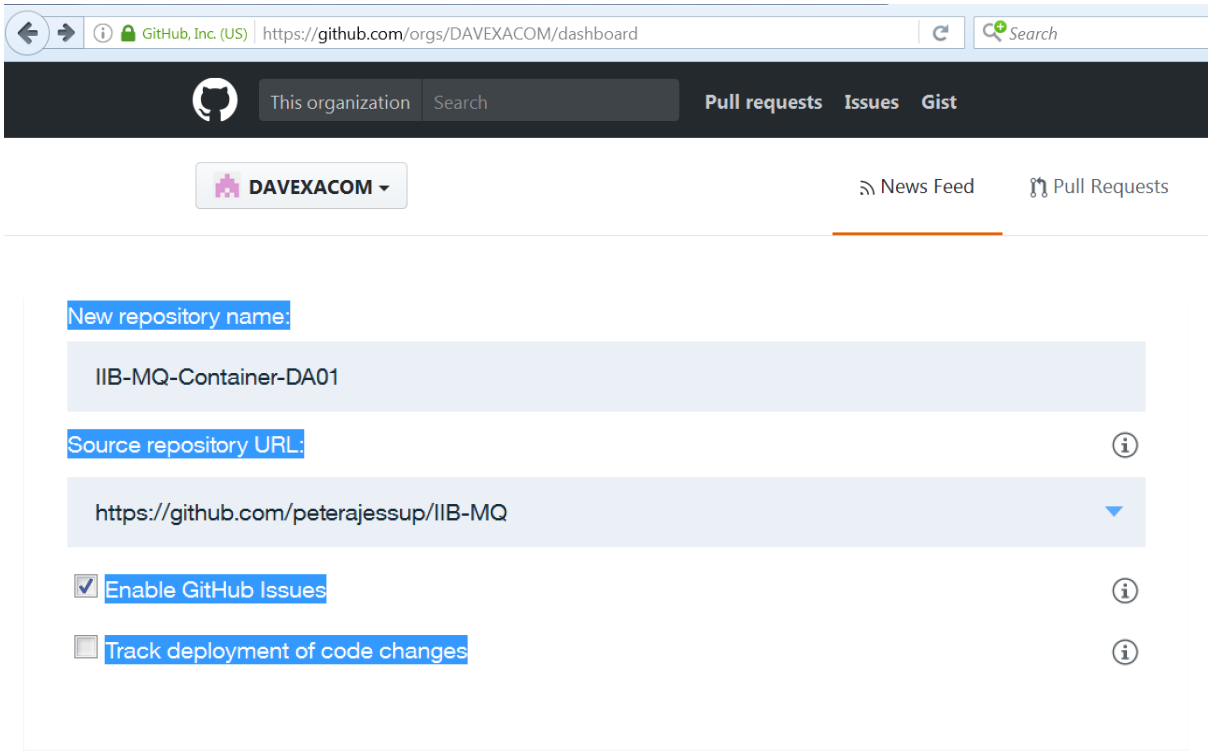

Figure 11: GitHub configuration

Step 4.

In a new browser tab, navigate to this source repository, and then explore the IIB and MQ artefacts. These artefacts will build an MQ and IIB container instance.

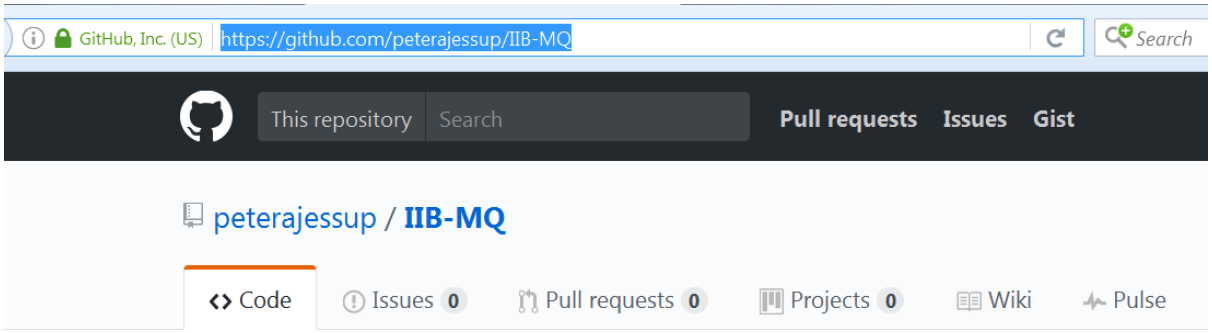

Figure 12: GitHub source repository

Step 5.

Back in Bluemix, click the Delivery Pipeline tile and then give the App a name. Ensure you use lowercase characters because uppercase is not permitted.

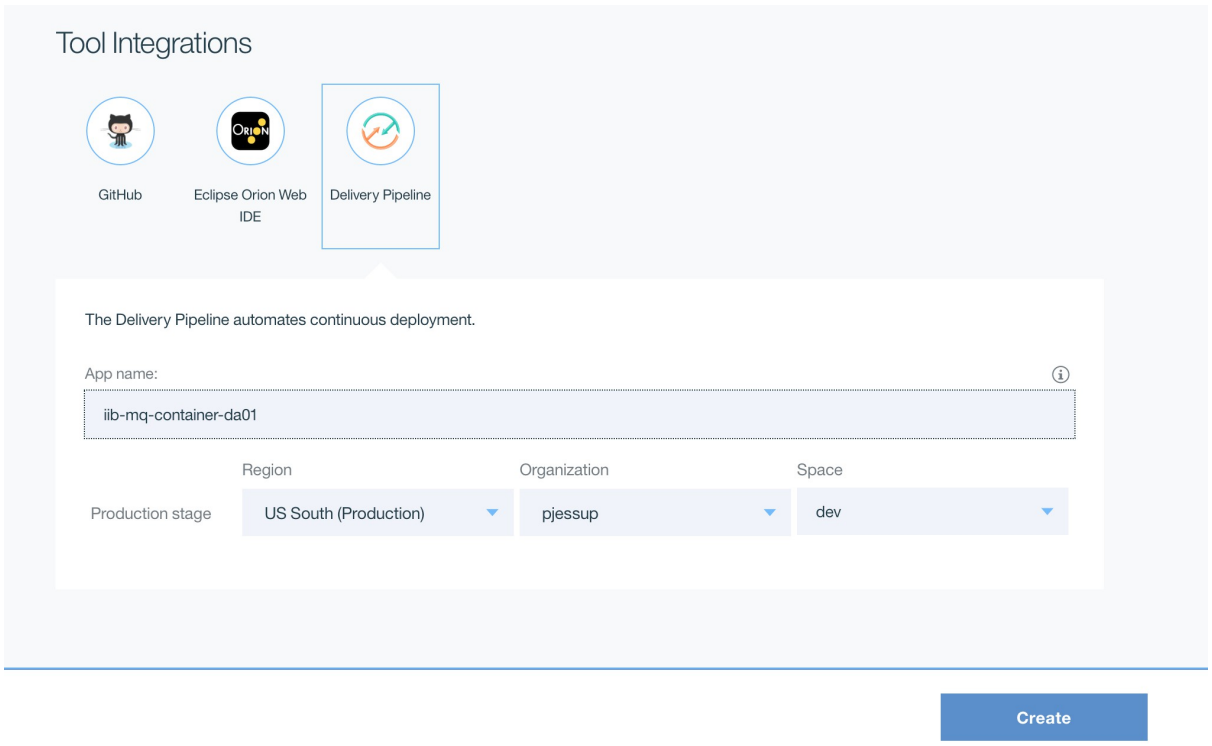

Figure 13: Pipeline configuration

Click the **Create** button to create your toolchain.

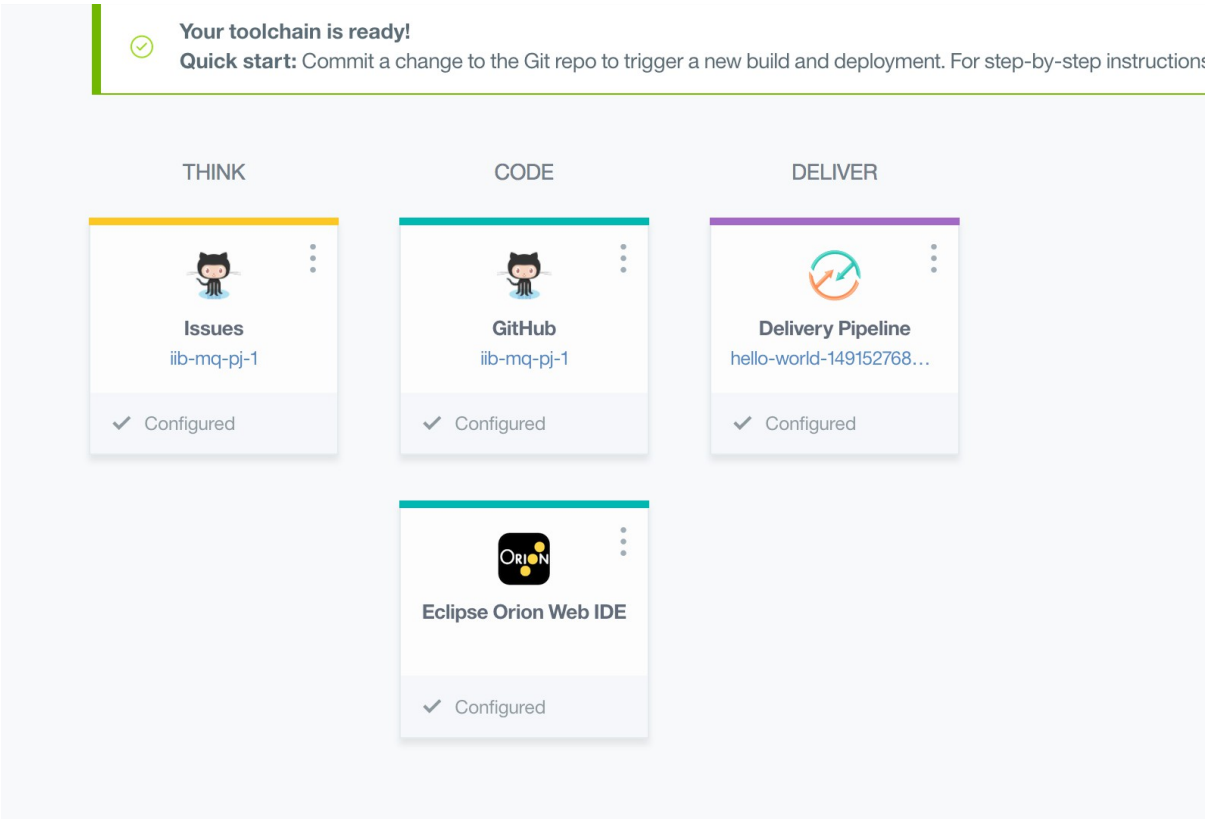

Figure 14: Completed toolchain

Step 6.

The toolchain immediately tries to build and deploy the project, but we're not ready to do that yet, so click the Delivery Pipeline tile and then cancel the build stage, which will have already started.

Cancel the build by clicking the **Stop** button.

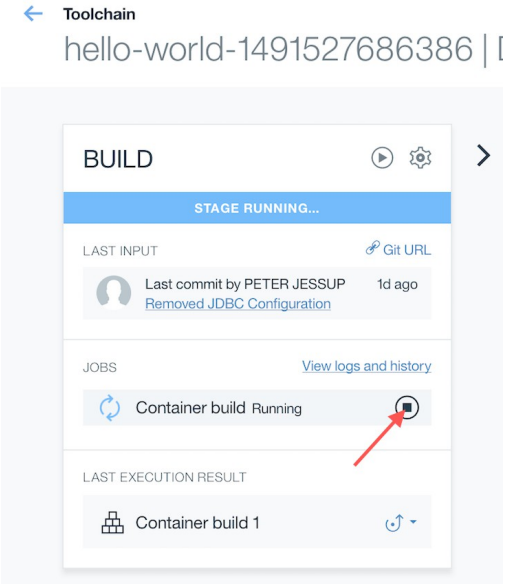

Figure 15: Initial build

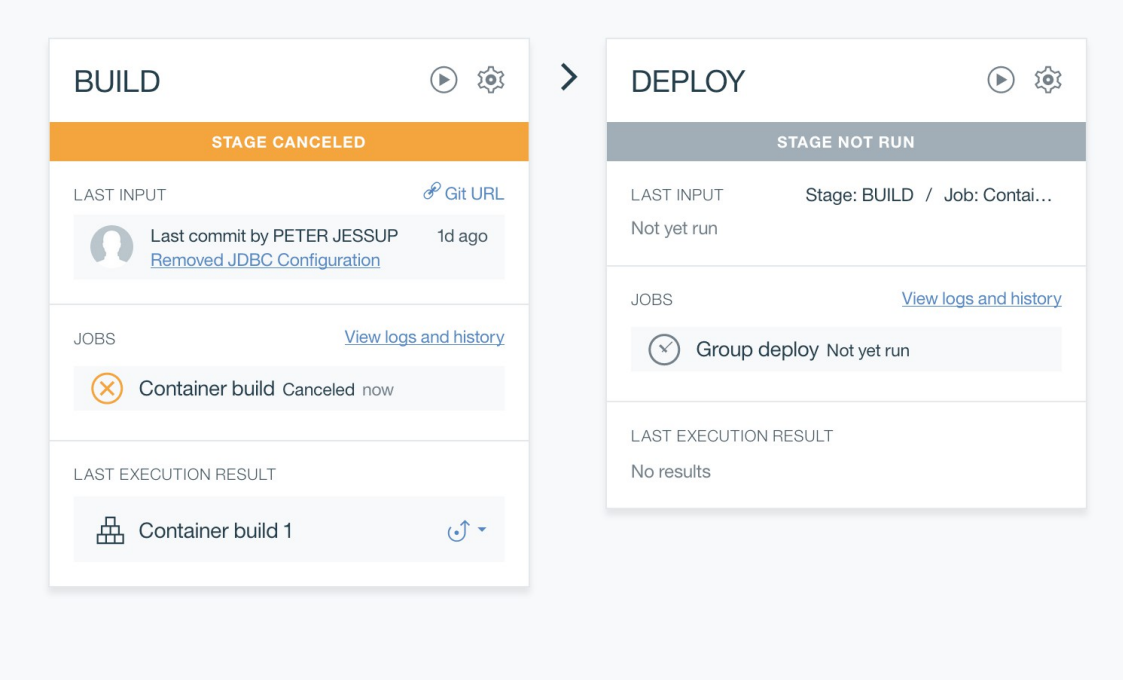

Figure 16: Build cancelled

Step 7.

Click the settings icon (top right of the tile) on the deploy tile, and then select **Configure Stage**.

Modify the Port, Optional Deploy Arguments, and Deploy Script as follows.

- The ports we want to expose on the container for IIB and MQ as shown below.

Figure 17: Deploy stage port configuration

- The optional deploy arguments are specified as two environment variables which accept the IIB and MQ licenses and provide a queue manager name for the queue manager creation and configuration script which runs during the deploy phase.

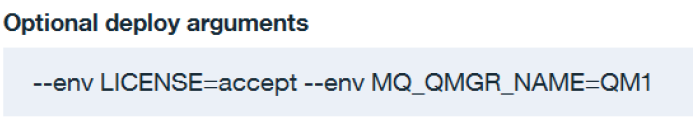

Figure 18: Deploy stage arguments

- The deployer script defaults to creating a container group. For now we want to create a single container. Comment out the line `/bin/bash deployscripts/deploygroup.sh` and uncomment the line `/bin/bash deployscripts/deploycontainer.sh` as shown below.

Figure 19: Deploy script - comment out group deployment

Save your changes, and then start the build again.

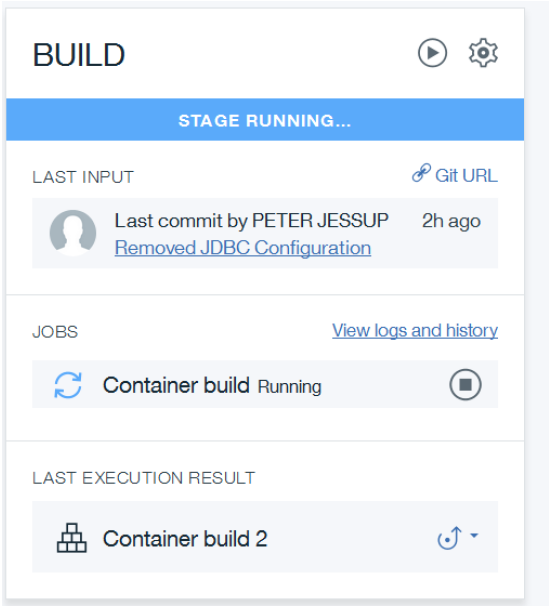

Figure 20: Build restart

The build will run, then pass control of the pipeline to the deploy task. After the stages have run you should see both stages having passed.

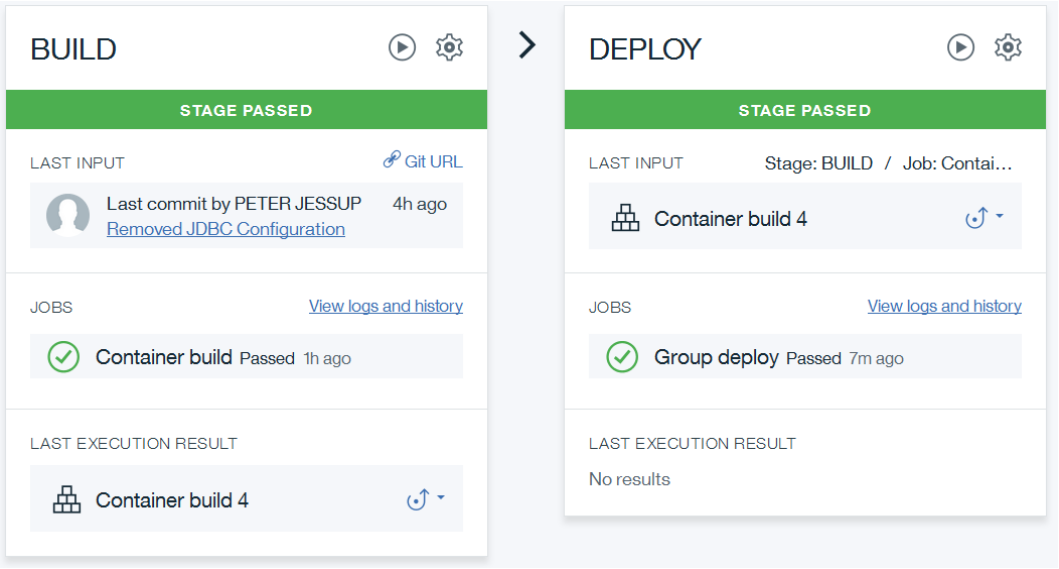

Figure 21: Build and Deploy success

Step 8.

In your IBM Bluemix dashboard, find your container, which should now be running.

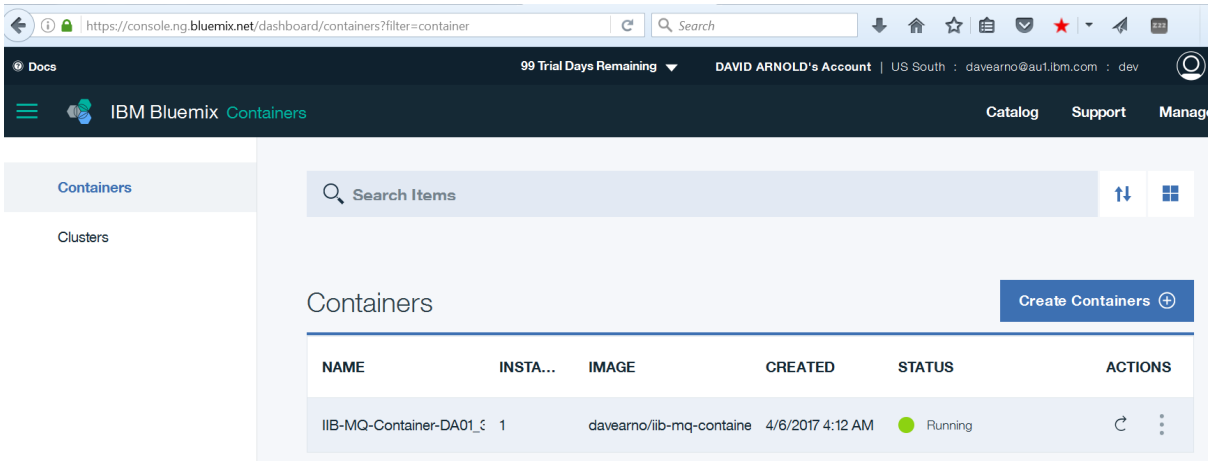

Figure 22: Bluemix container in dashboard

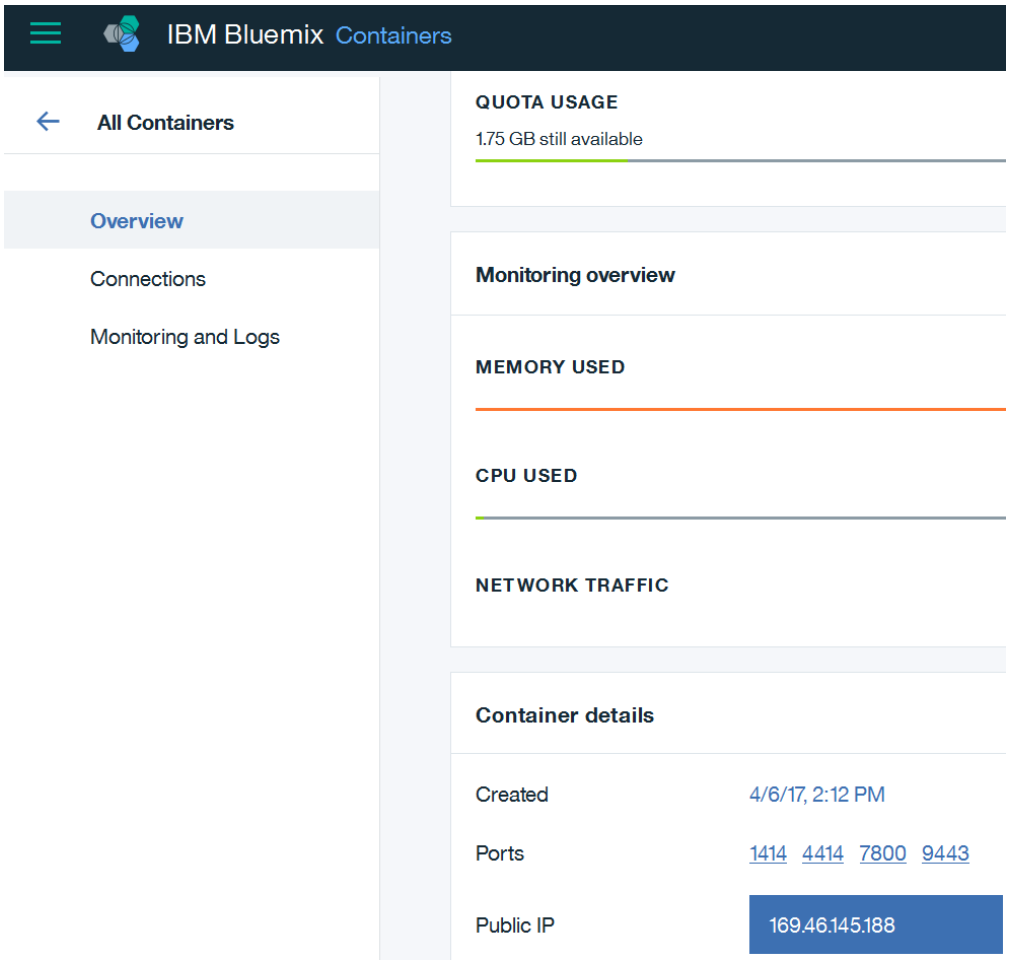

Figure 23: Container detail view

Identify the Public IP Address of the container and log into the IIB web console to check the status of the IIB instance.

In this case the URL is `169.46.145.188:4414`

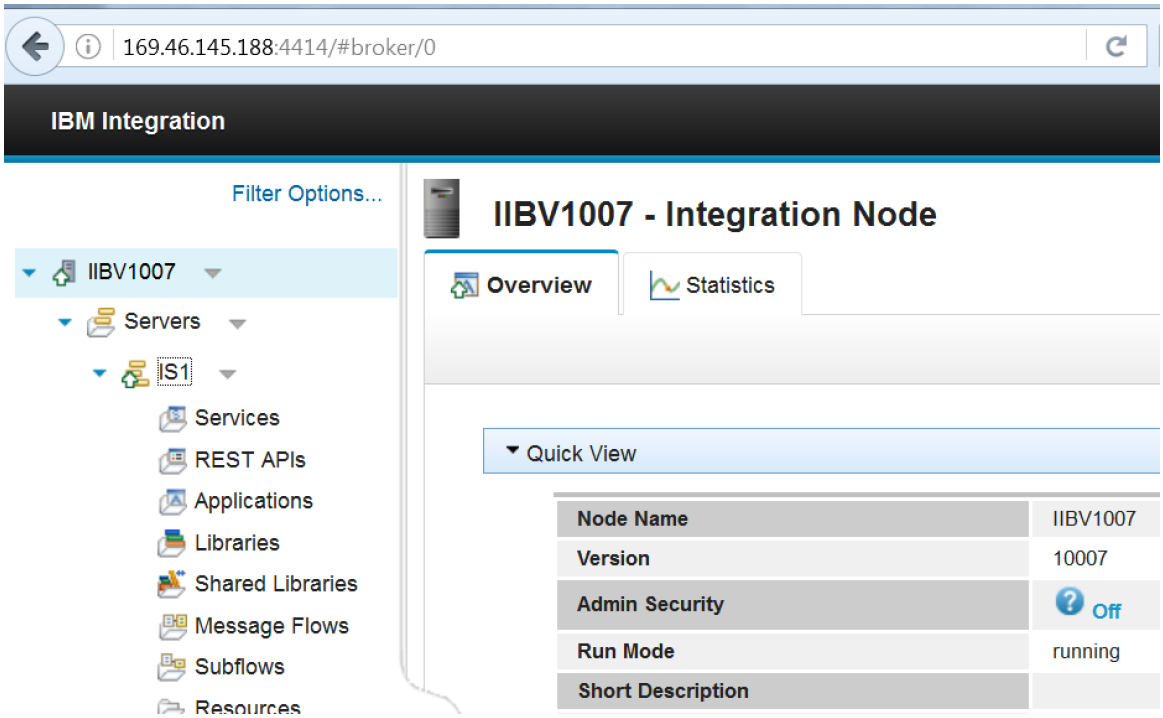

Figure 24: IIB web console

Step 9.

Use the same Public IP Address of the container and log into the Web Console of MQ to check the status of the MQ instance. In this case the URL is https://169.46.145.188:9443.

If see a security exception as the console is delivered with a self-signed certificate, replace this certificate with a certificate of your own organisation.

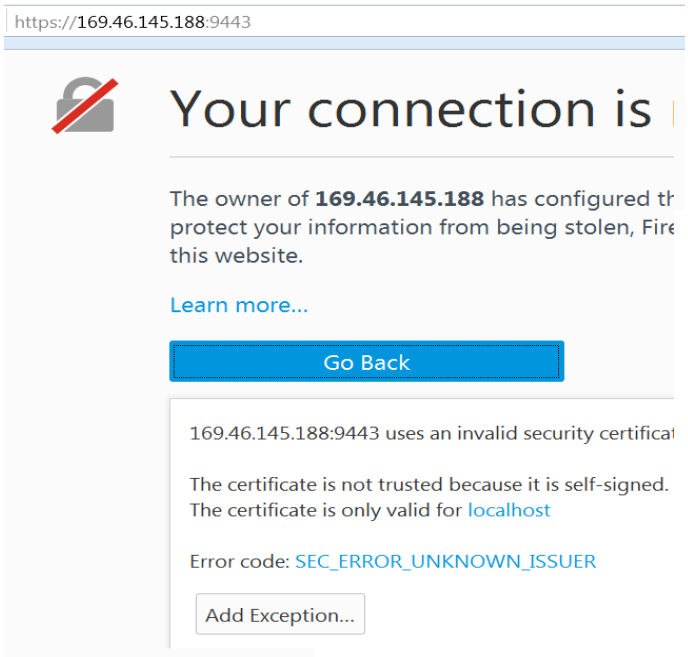

Figure 25: Security warning

Login with default credentials admin/passw0rd. This password can be changed using an environment variable for the deploy stage.

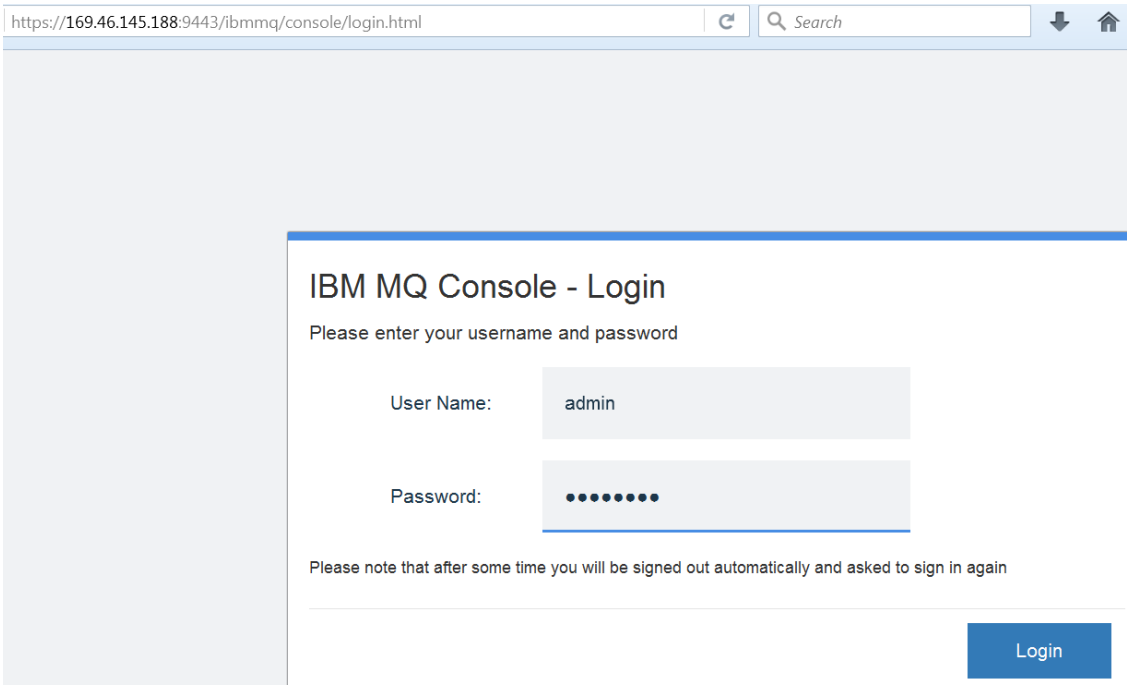

Figure 26: IBM MQ Console Login

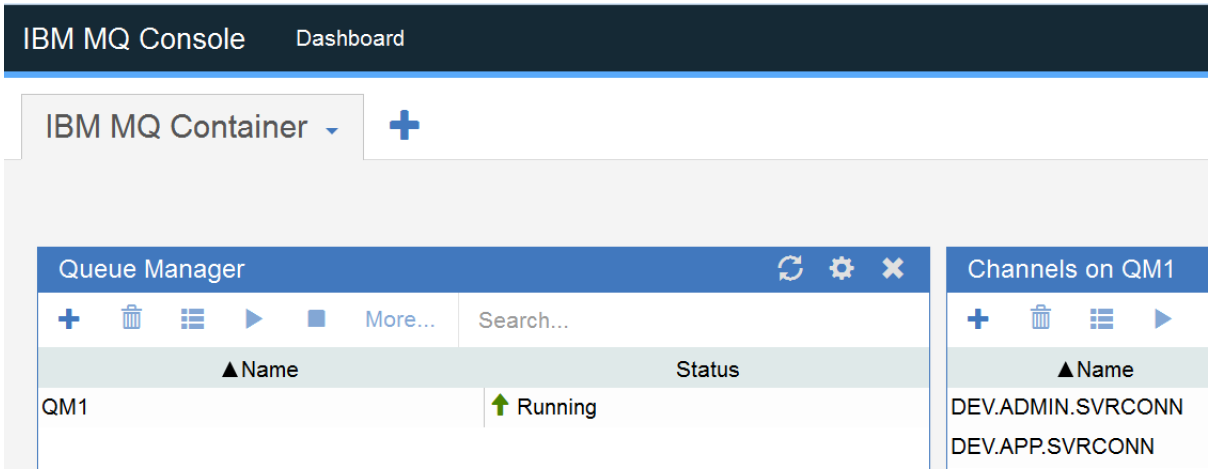

Figure 27: IBM MQ web console

Conclusion

Congratulations, you have successfully deployed a Bluemix-hosted Docker Container with a running instance of IBM Integration bus and IBM MQ by:

1. Creating a Continuous delivery Service in Bluemix
2. Configuring GitHub and the build pipeline to clone,build and deploy the artefacts
3. Customising the deploy script to configure the container.
4. Testing the IIB and MQ instances in a web browser.

For more information about IIB and Docker: https://developer.ibm.com/integration/blog/2015/12/02/ibm-integration-bus-now-supports-docker-for-production-use/

For more information about IBM MQ and Docker: https://www.ibm.com/developerworks/community/blogs/messaging/entry/mq-docker-supported?lang=en

For more information about Bluemix Continuous Delivery: https://www.ibm.com/blogs/bluemix/2016/11/bluemix-continuous-delivery-is-now-live/