

think 2018

IBM



Lab Center – Hands-on Lab

Session: Scheduled Lab 2500 / Open Lab 9060

Session Title: Deploy an application to IBM Cloud and IBM Cloud Private using Kubernetes and IBM UrbanCode Deploy

Hollis Chui, IBM, hollisc@ca.ibm.com

Steven M Cotugno, IBM, steve.cotugno@us.ibm.com

Table of Contents

Disclaimer	4
Introduction	6
Business Scenario	6
Application Environment Architecture.....	6
Objectives	7
DevOps Architecture	7
Lab Overview	8
What is Already Completed	8
Lab Exercises	9
Lab 0: Create IBM Cloud and GitHub accounts	9
Lab 0.1: Create IBM Cloud trial account.....	9
Lab 0.2: Create GitHub account	9
Lab 1: Clone Required GitHub Repositories	10
Lab 1.1: Clone the THINK2018_SL2500_OL9060_MultiCloud_Deployment GitHub Repository.....	10
Lab 1.2: Create a nodejs-cloudant-demo GitHub Repository.....	10
Lab 1.3: Clone nodejs-cloudant-demo GitHub repository.....	10
Lab 1.4: Push clone content into your GitHub repository	10
Lab 1.5: Delete the clone repository directory.....	10
Lab 1.6: Clone your GitHub repository as a local workspace	10
Lab 2: Provision an instance of Cloudant and create a Service Credential	12
Lab 3: Create and configure a Kubernetes cluster in IBM Cloud.....	15
Lab 3.1: Create a Kubernetes cluster in IBM Cloud	15
Lab 3.2: Configure the Kubernetes cluster in IBM Cloud with a DEV Namespace	17
Lab 3.3: Configure the Kubernetes cluster in IBM Cloud with a TEST Namespace	19
Lab 3.4: Configure the Kubernetes cluster in IBM Cloud with a QA Namespace	19
Lab 4: Create a Registry Namespace, generate an API key and find the IBM Account ID	21
Lab 5: GitHub and Jenkins integration	24
Lab 5.1: Configure the GitHub and Jenkin integration	24
Lab 6: IBM UrbanCode Deploy and Jenkins integration	26
Lab 6.1: Start IBM UrbanCode Deploy Server and Agent as root.....	26
Lab 6.2: Configure the IBM UrbanCode Deploy and Jenkins integration	26
Lab 7: Jenkins Configuration	27

Lab 7.1: Update Jenkins Job	27
Lab 8: IBM UrbanCode Deploy Configuration	29
Lab 8.1: Update Environment Properties for DEV environment	29
Lab 8.2: Update Environment Properties for TEST environment	30
Lab 8.3: Update Environment Properties for QA environment.....	32
Lab 9: Update IBM UrbanCode Deploy Resource.....	34
Lab 10: Run the CI/CD pipeline	44

Disclaimer

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.

The development, release, and timing of any future features or functionality described for our products remains at our sole discretion I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results like those stated here.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed "as is" without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts.

In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply."

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may

need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.

© 2018 International Business Machines Corporation. No part of this document may be reproduced or transmitted in any form without written permission from IBM.

U.S. Government Users Restricted Rights — use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

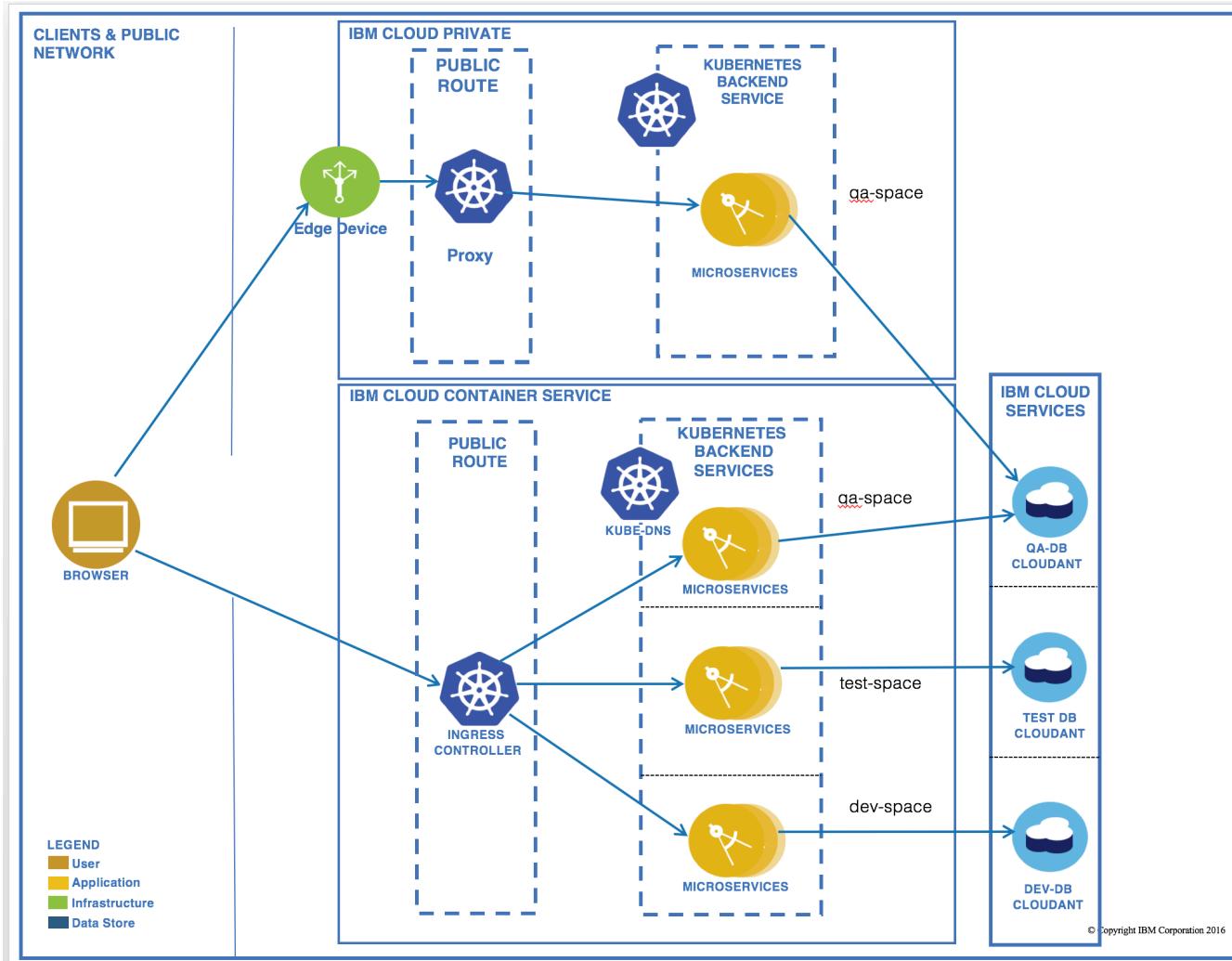
Introduction

The purpose of this lab is to walk through a multi-Cloud deployment solution using IBM UrbanCode Deploy. The lab will go over the architecture, configuration and running of the continuous integration and continuous deployment pipeline.

Business Scenario

Businesses recognize that applications being deployed to multiple Cloud platforms will be common place and a solution that can address complex deployment workflows is required. The DevOps solution covered by this lab provides a solution for deploying containerized applications on Kubernetes on both IBM Cloud Container Service and IBM Cloud Private.

Application Environment Architecture



Objectives

The objective of the lab is to run through the end to end workflow of the multi-Cloud deployment solution. The continuous integration and continuous deployment process will build a Docker image based on the latest code in GitHub, push the image into the Container Registry and IBM UrbanCode Deploy will deploy the container into Kubernetes clusters representing the different environments.

For this specific lab, the tool stack consists of:

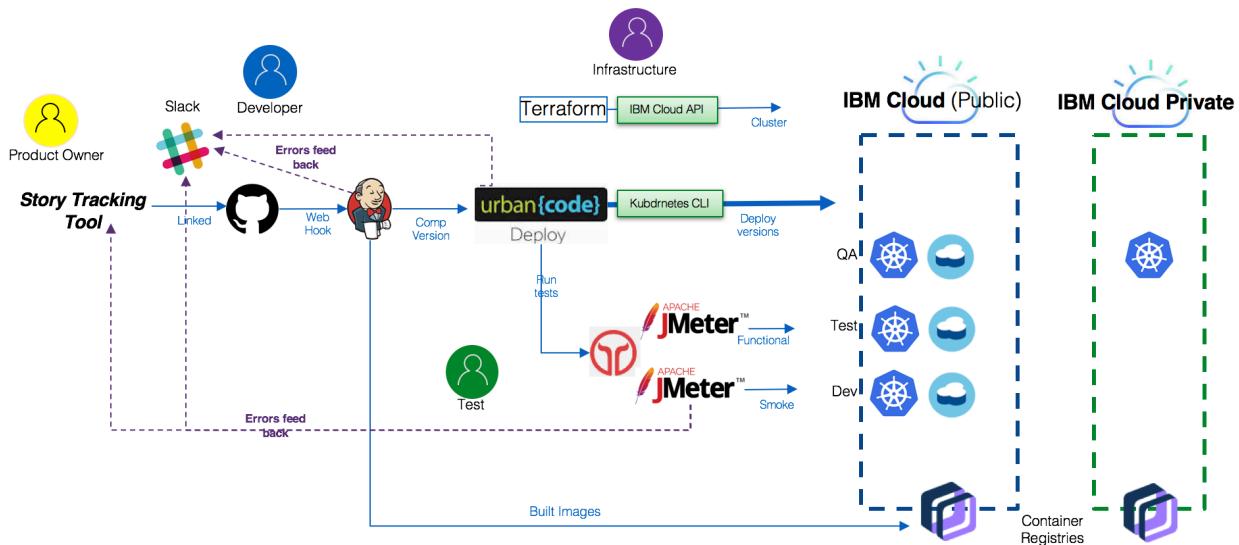
- GitHub Enterprise is the SCM system
- Jenkins is the build tool
- IBM UrbanCode Deploy is the deployment tool

The deployment process in IBM UrbanCode Deploy will perform the following:

- Deploy application container into the IBM Container Service Kubernetes cluster under the DEV namespace
- Placeholder to run smoke tests for target application
- Deploy application container into the IBM Container Service Kubernetes cluster under the TEST namespace
- Placeholder to run functional tests for target application

For the QA deployment, the application container will be deployed into the IBM Container Service Kubernetes cluster under the QA namespace as well as into IBM Cloud Private. This is a manual deploy but it can be easily automated as part of the pipeline process. This lab is meant to provide a framework to perform multiple Cloud platforms running on Kubernetes.

DevOps Architecture



Lab Overview

What is Already Completed

The lab will be run on two VMs hosted in Skytap. Below are user / password for the accounts used to access the pre-installed software.

IBM Cloud Private VM: Boot / Master Node (10.0.0.1):

- **Accounts:**
 - VM: skytap / A1rb0rn3
 - IBM Cloud Private: admin / admin
- **Installed Software:**
 - IBM Cloud Private 2.1.0.1

IBM UrbanCode Deploy VM: DevOps (10.0.0.2):

- **Accounts:**
 - VM: devops / devops
 - Jenkins: admin / admin
 - IBM UrbanCode Deploy: admin /admin
- **Installed Software:**
 - IBM UrbanCode Deploy 6.2.2 + local agent
 - Jenkins 2.19

Lab Exercises

To achieve the objectives of this lab, the following exercises must be completed:

Lab 0: Create IBM Cloud and GitHub accounts

Lab 0.1: Create IBM Cloud trial account

1. If you already have an active IBM Cloud account, skip this task.
2. Open a web browser and go to <https://console.ng.bluemix.net/>.
3. Click on the “**Create a free account**” button.
4. Follow the directions to fill out the form and make a note of the password specified. Note, you will need access to the email to confirm the account creation.
5. Click “**Create Account**” and IBM Cloud will send a confirmation email to the account specified.
6. Login into the email account specified and open the email with the subject: **Action Required: Confirm your IBM Cloud account**.
7. Click on the “**Confirm Account**” button.
8. You now have an active IBM Cloud trial account.
9. When specifying the name of your IBM Cloud Org, do not use any special characters.

Lab 0.2: Create GitHub account

1. If you already have a GitHub account, skip this task.
2. Open a web browser and go to <https://github.com/>.
3. Follow the directions to fill out the form and make a note of the password specified. Note, you will need access to the email to confirm the account creation.
4. Click on the “**Sign up for GitHub**” button and GitHub will send a confirmation email to the account specified.
5. Login into the email account specified and open the email from GitHub with the subject: **Please verify your email address**.
6. Click on the “**Verify email address**” link.
7. You now have an active GitHub account.

Lab 1: Clone Required GitHub Repositories

Lab 1.1: Clone the THINK2018_SL2500_DL9060_MultiCloud_Deployment GitHub Repository

1. Open a terminal window in the DevOps VM and change to the /home/devops directory.
2. Create a directory to store your work using the command “**mkdir <Folder name>**”.
3. Change into the directory and clone the lab repository using the command “**git clone https://github.com/hollisc/THINK2018_SL2500_DL9060_MultiCloud_Deployment.git**”.

Lab 1.2: Create a nodejs-cloudant-demo GitHub Repository

1. Launch a web browser and go to <https://github.com>.
2. Click the “Sign in” link and log in using the account created in Lab 0.
3. Click the “Start a project” button.
4. On the Create a new repository page, enter **nodejs-cloudant-demo** and click the “Create repository” button.

Lab 1.3: Clone nodejs-cloudant-demo GitHub repository

1. Open a terminal window and change to the /home/devops/<Folder name> directory.
2. Clone the lab repository using the command “**git clone <https://github.com/hollisc/nodejs-cloudant-demo.git>**”.

Lab 1.4: Push clone content into your GitHub repository

1. From the terminal window, change into the clone repository directory /home/devops/<Folder name>/nodejs-cloudant-demo.
2. Push the contents into the GitHub repository created in Lab 2.1 by running the command: “**git push --mirror https://github.com/<GitHub_Username>/nodejs-cloudant-demo.git**”.
3. Enter your GitHub username and password.

```
devops@devops:~/Documents/Hollis/nodejs-cloudant-demo$ git push --mirror https://github.com/hollisc-chui/nodejs-cloudant-demo.git
Username for 'https://github.com': hollis-chui
Password for 'https://hollis-chui@github.com':
Counting objects: 300, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (140/140), done.
Writing objects: 100% (300/300), 279.39 KiB | 0 bytes/s, done.
Total 300 (delta 145), reused 300 (delta 145)
remote: Resolving deltas: 100% (145/145), done.
To https://github.com/hollis-chui/nodejs-cloudant-demo.git
 * [new branch]      master -> master
 * [new branch]      origin/HEAD -> origin/HEAD
 * [new branch]      origin/master -> origin/master
```

Lab 1.5: Delete the clone repository directory

1. From the terminal window, change to the /home/devops/<Folder name> directory.
2. Remove the directory by running the command ‘**rm -rf nodejs-cloudant-demo**’.

```
devops@devops:~/Documents/Hollis$ 
devops@devops:~/Documents/Hollis$ rm -rf nodejs-cloudant-demo/
devops@devops:~/Documents/Hollis$
```

Lab 1.6: Clone your GitHub repository as a local workspace

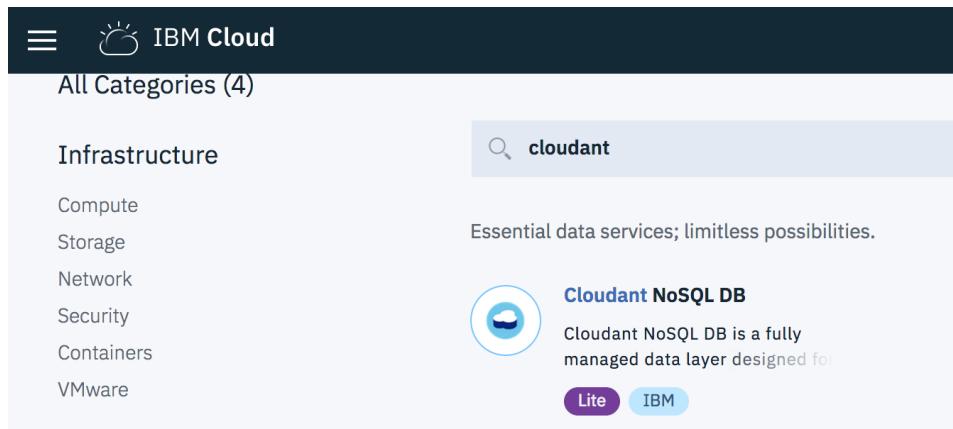
1. From the terminal window, change to the /home/devops/<Folder name> directory.

2. Clone your GitHub repository using the command “***git clone https://github.com/<GitHub_Username>/nodejs-cloudant-demo.git***”.

```
devops@devops:~/Documents/Hollis$ git clone https://github.com/hollis-chui/nodejs-cloudant-demo.git
Cloning into 'nodejs-cloudant-demo'...
remote: Counting objects: 300, done.
remote: Compressing objects: 100% (140/140), done.
remote: Total 300 (delta 145), reused 300 (delta 145), pack-reused 0
Receiving objects: 100% (300/300), 279.39 KiB | 0 bytes/s, done.
Resolving deltas: 100% (145/145), done.
Checking connectivity... done.
```

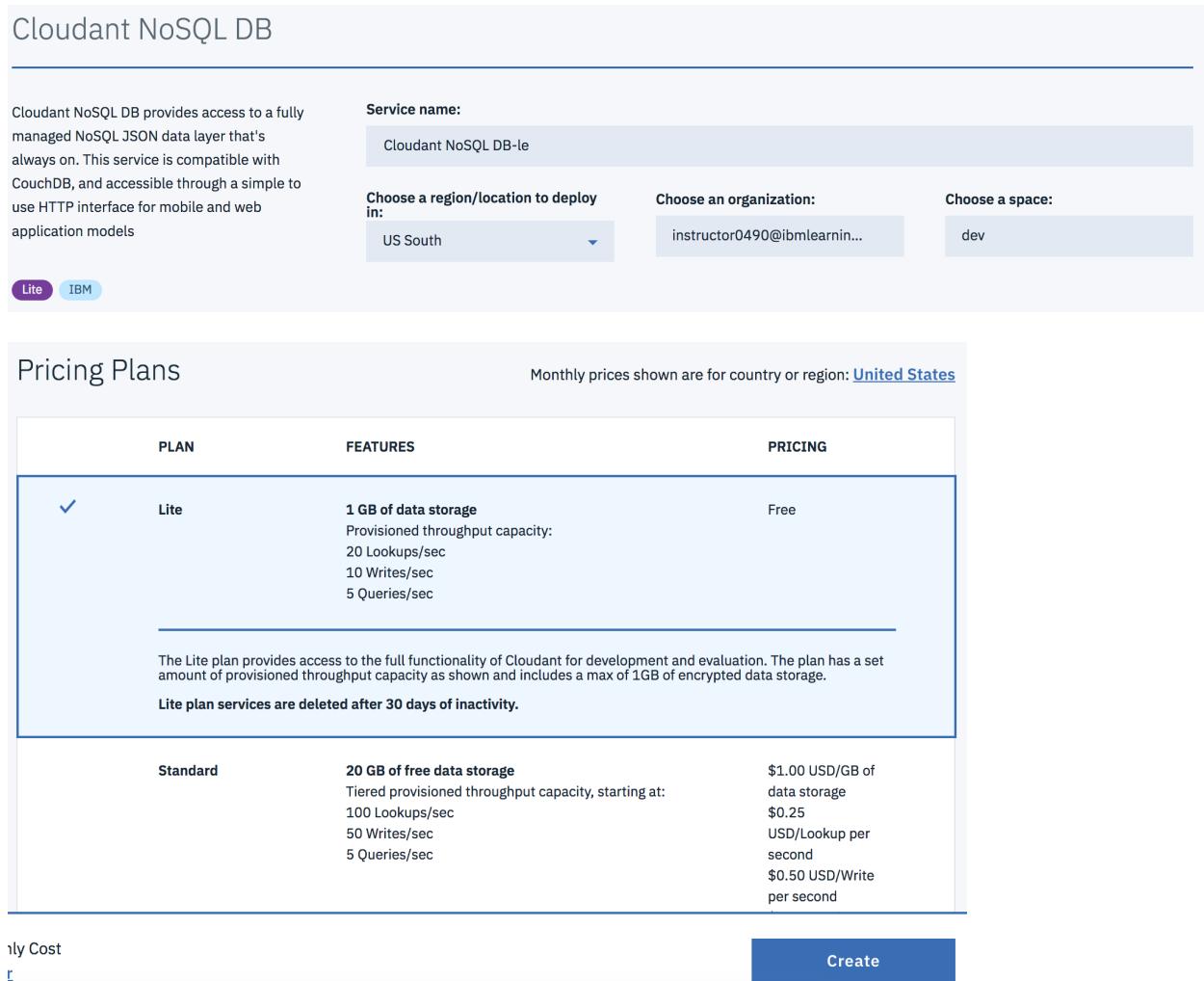
Lab 2: Provision an instance of Cloudant and create a Service Credential

1. From the IBM Cloud dashboard, click on Catalog and search for the Cloudant NoSQL DB tile.



The screenshot shows the IBM Cloud Catalog interface. At the top, there's a search bar with the word "cloudant" typed into it. Below the search bar, the results are displayed under the heading "Infrastructure". On the left, there's a sidebar with categories like Compute, Storage, Network, Security, Containers, and VMware. The "Cloudant NoSQL DB" service is listed as the first result. It has a small icon of a database, a brief description, and two buttons at the bottom: a purple "Lite" button and a blue "IBM" button.

2. Use the default options, select the Lite pricing plan and click Create.



The screenshot shows the "Cloudant NoSQL DB" creation page. On the left, there's a descriptive text about the service. On the right, there are several input fields: "Service name" (set to "Cloudant NoSQL DB-1e"), "Choose a region/location to deploy in" (set to "US South"), "Choose an organization" (set to "instructor0490@ibmlearnin..."), and "Choose a space" (set to "dev"). Below these fields, there are two buttons: "Lite" (highlighted in purple) and "IBM".

Pricing Plans

Monthly prices shown are for country or region: [United States](#)

PLAN	FEATURES	PRICING
✓ Lite	1 GB of data storage Provisioned throughput capacity: 20 Lookups/sec 10 Writes/sec 5 Queries/sec	Free
<p>The Lite plan provides access to the full functionality of Cloudant for development and evaluation. The plan has a set amount of provisioned throughput capacity as shown and includes a max of 1GB of encrypted data storage.</p> <p>Lite plan services are deleted after 30 days of inactivity.</p>		
Standard	20 GB of free data storage Tiered provisioned throughput capacity, starting at: 100 Lookups/sec 50 Writes/sec 5 Queries/sec	\$1.00 USD/GB of data storage \$0.25 USD/Lookup per second \$0.50 USD/Write per second

Finally Cost USD

Create

3. On the browser, select “Service credentials” and click “New credential”.

The screenshot shows the IBM Cloudant NoSQL DB service credentials page. The left sidebar has options: Manage, Service credentials (which is selected and highlighted in blue), Plan, and Connections. The main content area shows the service name "Cloudant NoSQL DB-le" with location "US South", organization "instructor0490@ibmlearning.org", and space "dev". Below this, there's a "Service credentials" section with a note about JSON format and a "View More" link. At the bottom right of this section is a "New credential" button with a plus sign and three dots. A horizontal line separates this from the footer.

4. Use the default name and click Add.

The screenshot shows the "Add new credential" dialog box. It has a title bar with a close button (X). The main area has a "Name:" label with a text input field containing "Credentials-1". Below it is a section titled "Add Inline Configuration Parameters (Optional):" with an information icon. At the bottom right are "Cancel" and "Add" buttons.

5. Copy the json of the generated Cloudant service credential and paste the contents into the <path to THINK2018_SL2500_DL9060>/Cloudant_Credentials/binding file.

KEY NAME	DATE CREATED	ACTIONS
Credentials-1	Mar 16, 2018 - 04:09:09	View credentials ▾ 

```

    "username": "dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix",
    "password": "e3035cd0cc8f7dadfe0d217baf3bfad65baf23daea57669dd8f8f8b132dae36f",
    "host": "dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix.cloudant.com",
    "port": 443,
    "url": "https://dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix:e3035cd0cc8f7dadfe0
d217baf3bfad65baf23daea57669dd8f8f8b132dae36f@dcc57faf-4ef0-4fba-88ca-9ef80a9126cf
-bluemix.cloudant.com"
}

```

```

devops@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Cloudant_Credentials$ cat binding
{
  "username": "dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix",
  "password": "e3035cd0cc8f7dadfe0d217baf3bfad65baf23daea57669dd8f8f8b132dae36f",
  "host": "dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix.cloudant.com",
  "port": 443,
  "url": "https://dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix:e3035cd0cc8f7dadfe0d217baf3bfad65baf23daea57669dd8f8
f8b132dae36f@dcc57faf-4ef0-4fba-88ca-9ef80a9126cf-bluemix.cloudant.com"
}

```

Lab 3: Create and configure a Kubernetes cluster in IBM Cloud

Lab 3.1: Create a Kubernetes cluster in IBM Cloud

1. Open a web browser and log in to IBM Cloud (<https://console.ng.bluemix.net/>) using your account. This will bring you to the IBM Cloud Dashboard.

The screenshot shows the IBM Cloud Dashboard. At the top, there are navigation links for Catalog, Docs, Support, and Manage. Below that, the dashboard header includes 'REGION US South', 'CLOUD FOUNDRY ORG instructor0490@i...arning.org', and 'CLOUD FOUNDRY SPACE dev'. A search bar and a 'Create resource' button are also present. The main content area features a 'Getting Started Tutorials' section with several links: Watson (Conversation, Discovery, Language Translator, Natural Language Understanding, Personality Insights, Tone Analyzer), Internet of Things (Internet of Things Platform, Push Notifications), Mobile, and DevOps (Availability Monitoring).

2. From the dashboard, click on the hamburger menu on the top left and select Containers.

This screenshot is similar to the previous one, but the sidebar on the left is open, showing a list of options: Containers (selected), Infrastructure, VMware, Dashboard (selected), APIs, Application Services, and Blockchain. The rest of the dashboard interface is identical to the first screenshot, displaying the Getting Started Tutorials section.

3. Select “Cluster type = Free” and “Cluster name = mycluster”.

Region

US South

Cluster type

Free

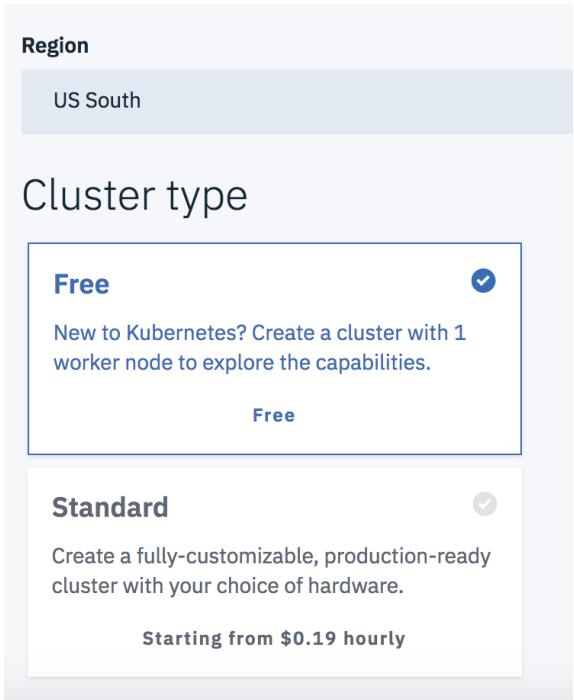
New to Kubernetes? Create a cluster with 1 worker node to explore the capabilities.

[Free](#)

Standard

Create a fully-customizable, production-ready cluster with your choice of hardware.

[Starting from \\$0.19 hourly](#)



Cluster details

Cluster name

mycluster

[Create Cluster](#)

4. Click Create Cluster.

The screenshot shows the IBM Cloud interface. On the left, a sidebar menu has 'Access' selected. The main content area displays a message about Kubernetes version 1.5 being deprecated. Below this, it shows the 'Clusters / mycluster' section. A cluster icon labeled 'mycluster' with a status of 'Deploying' is shown. A large heading says 'Gain access to your cluster'. Under 'Prerequisites', it states: 'To gain access to your cluster, download and install a few CLI tools and the IBM Cloud Container Service plug-in.' It lists three steps: 1. Download the [IBM Cloud CLI](#), 2. Download the [Kubernetes CLI](#), and 3. Install the container service plugin. A command line interface (CLI) box contains the command: `bx plugin install container-service -r Bluemix`.

5. The cluster will take ~ 25 minutes to provision and the cluster will have an indicator of “Ready” status when it complete.

The screenshot shows the same IBM Cloud interface as above, but the cluster status has changed. The 'mycluster' entry now shows a green dot next to the word 'Ready', indicating the cluster is fully provisioned and operational.

Lab 3.2: Configure the Kubernetes cluster in IBM Cloud with a DEV Namespace

1. From the browser, select “mycluster” on the IBM Cloud Clusters Overview page and select “Access” on the left menu.

Access

Gain access to your cluster

1. Log in to your IBM Cloud account.

```
bx login -a https://api.ng.bluemix.net
bx cs region-set us-south
```

If you have a federated ID, use bx login --sso to log in to the IBM Cloud CLI.

2. Set the context for the cluster in in your CLI.

- a. Get the command to set the environment variable and download the Kubernetes configuration files.

```
bx cs cluster-config mycluster
```

2. Open a terminal window and run as the root user via “sudo bash”. Follow the instructions under the “Gain access to your cluster” section using a terminal window to access the Kubernetes cluster.
3. Create a DEV namespace in the cluster using the command.

```
kubectl create -f dev_namespace.yml
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl create -f dev_namespace.yml
namespace "dev-space" created
```

4. Create a pull image secret for the DEV namespace.

```
kubectl get secret bluemix-default-secret -o yaml | sed 's/default/dev-space/g' | kubectl -n dev-space create -f -
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl get secret bluemix-default-secret -o yaml | sed 's/default/dev-space/g' | kubectl -n dev-space create -f -
secret "bluemix-dev-space-secret" created
```

5. Patch the existing service account with the pull image secret for the DEV namespace.

```
kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-dev-space-secret"}]}' --namespace=dev-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-dev-space-secret"}]}' --namespace=dev-space
serviceaccount "default" patched
```

6. Create a Kubernetes Secret to store the Cloudant credentials.

```
kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=dev-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Cloudant_Credentials# kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=dev-space
secret "binding-rates-cloudant-cred1" created
```

Lab 3.3: Configure the Kubernetes cluster in IBM Cloud with a TEST Namespace

1. Create a TEST namespace in the cluster using the command.

```
kubectl create -f test_namespace.yml
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl create -f test_namespace.yml
namespace "test-space" created
```

2. Create a pull image secret for the TEST namespace.

```
kubectl get secret bluemix-default-secret -o yaml | sed 's/default/test-space/g' | kubectl -n test-space create -f -
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl get secret bluemix-default-secret -o yaml | sed 's/default/test-space/g' | kubectl -n test-space create -f -
secret "bluemix-test-space-secret" created
```

3. Patch the existing service account with the pull image secret for the TEST namespace.

```
kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-test-space-secret"}]}' --namespace=test-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-test-space-secret"}]}' --namespace=test-space
serviceaccount "default" patched
```

4. Create a Kubernetes Secret to store the Cloudant credentials.

```
kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=test-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Cloudant_Credentials# kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=test-space
secret "binding-rates-cloudant-cred1" created
```

Lab 3.4: Configure the Kubernetes cluster in IBM Cloud with a QA Namespace

1. Create a QA namespace in the cluster using the command.

```
kubectl create -f qa_namespace.yml
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl create -f qa_namespace.yml
namespace "qa-space" created
```

2. Create a pull image secret for the QA namespace.

```
kubectl get secret bluemix-default-secret -o yaml | sed 's/default/qa-space/g' | kubectl -n qa-space create -f -
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl get secret bluemix-default-secret -o yaml | sed 's/default/qa-space/g' | kubectl -n qa-space create -f -
secret "bluemix-qa-space-secret" created
```

3. Patch the existing service account with the pull image secret for the QA namespace.

```
kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-qa-space-secret"}]}' --namespace=qa-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Kubernetes_YAML# kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "bluemix-qa-space-secret"}]}' --namespace=qa-space  
serviceaccount "default" patched
```

4. Create a Kubernetes Secret to store the Cloudant credentials.

```
kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=qa-space
```

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Cloudant_Credentials# kubectl create secret generic binding-rates-cloudant-cred1 --from-file=./binding --namespace=qa-space  
secret "binding-rates-cloudant-cred1" created
```

Lab 4: Create a Registry Namespace, generate an API key and find the IBM Account ID

1. From the IBM Cloud dashboard, click on the hamburger menu on the top left and select Containers.
2. Click on Registry > Quick Start (<https://console.bluemix.net/containers-kubernetes/registry/start>) and follow the instructions to add a custom IBM Container Service Namespace.

`bx cr namespace-add <IBM Container Service Namespace>`

The screenshot shows the IBM Cloud Registry > Quick Start interface. On the left, there's a sidebar with links like Overview, Clusters, Registry, Quick Start (which is selected), Private Repositories, IBM Public Repositories, Vulnerability Advisor, Solutions, Helm Repositories, and Helm Charts. The main content area has a heading "Let's get started by installing the needed CLIs, setting up your first private registry namespace, and pushing your first image." It lists five steps:

1. [Install the IBM Cloud CLI.](#)
2. [Install the Docker CLI.](#)
3. Install the Container Registry plug-in.
A command box contains: `bx plugin install container-registry -r Bluemix`
4. Log in to your IBM Cloud account.
A command box contains: `bx login -a https://api.ng.bluemix.net`
5. Choose a name for your first namespace, and create that namespace. Use this namespace for the rest of the Quick Start.
A command box contains: `bx cr namespace-add <my_namespace>`

```
root@devops:~/Lab2500/THINK2018_SL2500_OL9060_MultiCloud_Deployment/Cloudant_Credentials# bx cr namespace-add labdemo
Plugin version '0.1.304' is now available. To update, run: bx plugin update container-registry -r Bluemix
Adding namespace 'labdemo'...
Successfully added namespace 'labdemo'
OK
```

3. On the IBM Cloud dashboard, click Manage > Security > Platform API Keys and click Create.

The screenshot shows the IBM Cloud Manage > Security > Platform API Keys interface. At the top, there are tabs for Catalog, Docs, Support, and Manage. Under Manage, there are sub-links for Account, Billing and Usage, and Security. The "Platform API Keys" link is highlighted in blue. The main content area shows a table with two rows: "Identity and Access" and "Platform API Keys".

The screenshot shows the 'Platform API Keys' section under 'Identity & Access'. A 'Create' button is visible at the bottom right.

4. Enter a name for the API key and click Create.

The dialog box has a 'Name' field containing 'devops' and a 'Description' field which is empty. It includes 'Cancel' and 'Create' buttons.

5. Click the Show button and download the API Key. This will be needed in Lab 7.

The dialog box displays the message 'API key successfully created.' It shows the API key value 'ic0V7vnojKLOVFb0Bh_aFHomifCLk8nAMb0OYbx30CI' with a 'Hide' link, and 'Close' and 'Download' buttons.

6. On the IBM Cloud dashboard, click Manage > Billing and Usage and select Billing on the left menu. Note down the Account ID as this will be used in Lab 7.

The screenshot shows a sidebar menu on the left with the following options:

- Profile
- Platform Notifications
- Usage Dashboard
- Billing** (selected)
- Cloud Foundry Orgs
- Resource Groups

The main content area is titled "Billing" and contains the following information:

Account
IBM 
ID: 343021f8c8c74357d5e000fe3216d76e

Lab 5: GitHub and Jenkins integration

Lab 5.1: Configure the GitHub and Jenkins integration

1. Launch a web browser and go to <https://github.com>.
2. Click the “Sign in” link and log in using the account created in Lab 0.
3. Click the dropdown menu on the top right corner and select “Your profile”.
4. Select the Repositories tab and click on the “nodejs-cloudant-demo” repository. Select the Settings tab and click on “Webhooks”.

The screenshot shows the GitHub repository settings for 'nodejs-cloudant-demo'. The 'Webhooks' tab is selected in the sidebar. The main content area displays information about Webhooks, stating they allow external services to be notified when certain events happen, and provides a link to the 'Webhooks Guide'. There is a 'Add webhook' button at the top right of this section.

5. The URL for Jenkins is published as a service from the Skytap VM. This URL is unique for each VM instance. Follow the steps below to find the published service URL for Jenkins on your VM instance. Note, if you are using an on-premise install of Jenkins, simply use the URL to access the Jenkins UI.
6. Go to the IBM Cloud Conference Scheduler with the link to launch your VM.
7. There is a table listing the internal ports and corresponding external IP and ports. The Jenkins application is using port 8080 on the VM. The corresponding external IP and Port will be used for the Webhook URL.

The screenshot shows the 'CONNECT TO THIS VM' interface. It lists network adapters and their published services:
- **from the Internet (3):**
 Published Services (3)
 Port Address Network adapter
 7918 services-uscentral.sktap.com:12461 host-1 - 10.0.0.2
 8080 services-uscentral.sktap.com:12467 host-1 - 10.0.0.2
 8443 services-uscentral.sktap.com:12476 host-1 - 10.0.0.2
- **from a non-NAT source (1 connected network adapter):**
 Network Network adapter
 Blue Demos host-1 - 10.0.0.2
- **from a WAN NAT source (0 WANs):**
- **from an ICNR NAT source (0 networks):**

8. Go back to your web browser and click the **Add webhook** button.

9. In the Payload URL, enter the Published Service URL ‘`http://<External IP>:<External Port>/github-webhook/`’ and click the “Add webhook” button. Note, the ending “/” is required in the URL.

Webhooks / **Manage webhook**

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *

`http://services-uscentral.skytap.com:12467/github-webhook/`

Content type

`application/x-www-form-urlencoded`

Secret

Which events would you like to trigger this webhook?

Just the push event.

10. Verify that a green checkmark appears beside the added webhook.

Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

<input checked="" type="checkbox"/> http://services-uscentral.skytap.com:12467/github-webhook/ (push)	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
---	-------------------------------------	---------------------------------------

Lab 6: IBM UrbanCode Deploy and Jenkins integration

Lab 6.1: Start IBM UrbanCode Deploy Server and Agent as root

1. Open a terminal window and type in “**sudo bash**” to run as root. When prompted for the password for the devops user, enter devops.
2. Go to the UCD Server bin directory (**/opt/ibm-ucd/server/bin**) and run ‘**./server start**’ to start the server.

```
root@devops:/opt/ibm-ucd/server/bin# ./server start  
root@devops:/opt/ibm-ucd/server/bin#
```

3. Go to the UCD Agent bin directory (**/opt/ibm-ucd/agent/bin**) and run ‘**./agent start**’ to start the agent.

```
root@devops:/opt/ibm-ucd/agent/bin# ./agent start  
root@devops:/opt/ibm-ucd/agent/bin#
```

Lab 6.2: Configure the IBM UrbanCode Deploy and Jenkins integration

1. From within the DevOps VM, launch the Firefox browser and click on the Jenkins bookmark

|  **Dashboard [Jenkins]**

2. Login to Jenkins using **admin / admin**.
3. Select the “**Manage Jenkins**” link and click on the “**Configure System**” link.
4. Search for “**IBM UrbanCode Deploy Server**” and verify the properties match the following:
 - i) Profile Name = Local UCD Server
 - ii) IBM UrbanCode Deploy URL = <https://devops:8443>
 - iii) User Name = admin
 - iv) Password = admin

IBM UrbanCode Deploy Server

Profile Name	Local UCD Server
IBM UrbanCode Deploy URL	https://devops:8443
User Name	admin
Password	*****

Delete **Test Connection**

5. Click the “**Test Connection**” button and verify there is a successful connection.

IBM UrbanCode Deploy Server

Profile Name	Local UCD Server
IBM UrbanCode Deploy URL	https://devops:8443
User Name	admin
Password	*****

Success **Test Connection**

Lab 7: Jenkins Configuration

Lab 7.1: Update Jenkins Job

- From the Firefox browser, bring up the Jenkins Dashboard and click on the “Build” link.

The screenshot shows the Jenkins dashboard with a single job listed. The job is named "Build". It has a status icon of a sun (yellow), indicating it is healthy. The last success was marked as "N/A". Below the dashboard, there is a link labeled "Icon: S M L" which allows for scaling the job icon.

- Click on the “Configure” link on the left menu.



- On the “General” tab, update the “GitHub project > Project url” property with your GitHub repository URL in HTTPS format.

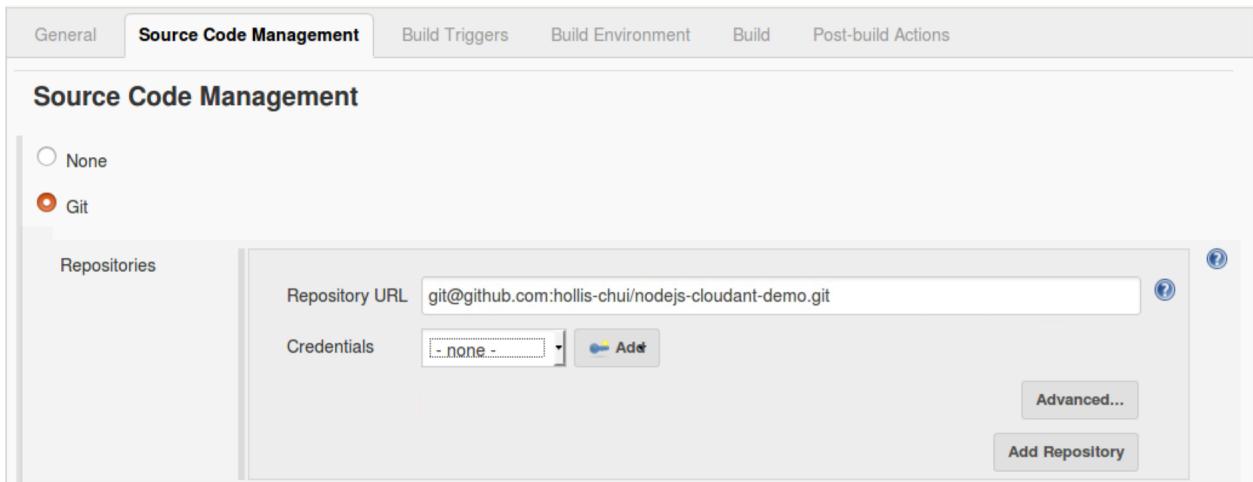
The screenshot shows the "General" configuration page for the "Build" job. The "Project name" is set to "Build". The "Description" field contains a detailed list of tasks:

- 1. Download files from GitHub repository (nodejs-cloudant-demo).
- 2. Build Docker image using Dockerfile in GitHub repo
- 3. Publish Docker image to IBM Cloud and IBM Cloud Private container registries.
- 4. Trigger IBM UrbanCode Deploy's Multi-Cloud Deployment process.

Below the description, there are two checkboxes: "Discard old builds" (unchecked) and "GitHub project" (checked). The "Project url" field is populated with "https://github.com/hollis-chui/nodejs-cloudant-demo.git". A "Preview" button is available to see the rendered description. An "Advanced..." button is located at the bottom right.

Note: The GitHub repository URL in HTTPS format can be found on the GitHub repository page and clicking on the “Clone or download” button. Ensure HTTPS is selected.

- Click on the “Source Code Management” tab and update the Git Repository URL property with your GitHub repository URL in SSH format.



Note: The GitHub repository URL can be found by going to the GitHub repository page and clicking on the “Clone or download” button. Ensure SSH is selected.

5. Click the “Build” tab > Inject environment variables section and update the following variables in the Properties Content.
 - BMX_USER=<IBM Cloud account>
 - BMX_ACCTID=<IBM Account ID>
 - BMX_ORG=<IBM Cloud account email>
 - BMX_SPACE=dev
 - BMX_APIKEY=<API Key>
 - ICS_NAMESPACE=<IBM Container Service Namespace>

```

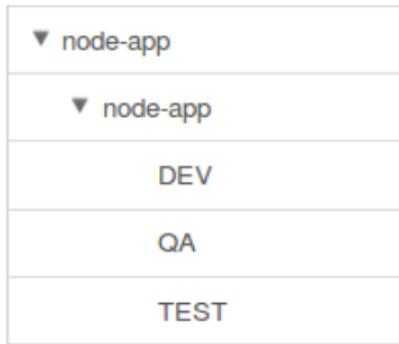
APPNAME=node-app
BMX_API=api.ng.bluemix.net
BMX_USER=instructor0490@ibmlearning.org
BMX_ACCTID=343021f8c8c74357d5e000fe3216d76e
BMX_ORG=instructor0490@ibmlearning.org
BMX_SPACE=dev
BMX_APIKEY=ic0V7vnojKLOVFb0Bh_aFHomifCLk8nAMb0OYbx30CPR
ICS_REGISTRY=registry.ng.bluemix.net
ICS_NAMESPACE=labdemo
ICS_SCRIPT_NAME=deployment_nodejs-cloudant-demo_bmx.yml
ICP_SCRIPT_NAME=deployment_nodejs-cloudant-demo_icp.yml
  
```

6. Click the “Save” button.

Lab 8: IBM UrbanCode Deploy Configuration

Lab 8.1: Update Environment Properties for DEV environment

1. From within the DevOps VM, launch the Firefox browser and click on the IBM UCD bookmark.
2. Login to IBM UrbanCode Deploy using **admin / admin**.
3. Select the “**Configuration**” tab and verify there are 3 environments listed under node-app.



4. Click on DEV environment and check that the “**Resources**” section matches the configuration in the screen capture.

The screenshot shows the "Configuration of node-app in DEV" page. Under the "Resources" section, there is a table with columns: Name, Inventory, Status, and Description. The table contains the following data:

Name	Inventory	Status	Description
Resource Name			
Node_Application / DEV			
BMX			
devops_local_agent (View Agent)		Online	
bmx			
Node_Application (View Component)			

At the bottom of the page, there are "Refresh" and "Print" buttons.

5. In the “**Environment Properties**” section, update the value for the following properties.

- a. BMX_USER = <IBM Cloud User Email>
- b. BMX_ACCTID = <use the IBM Cloud Account ID from Lab 4 Step 6>
- c. BMX_ORG = <the default value is the same as the BMX_USER value>
- d. ICS_NAMESPACE = <IBM Container Service Namespace>
- e. BMX_APIKEY = <use the API Key generated from Lab 4 Step 5>
- f. KUBENAMESPAC = dev-space

Environment Properties	
Use Batch Edit Mode	
KUBECLUSTER	mycluster
KUBECONFIG	/home/devops/.bluemix/plugins/container-service/clu...
BMX_API	api.ng.bluemix.net
BMX_USER	instructor0490@ibmlearning.org
BMX_ACCTID	343021f8c8c74357d5e000fe3216d76e
BMX_ORG	instructor0490@ibmlearning.org
BMX_SPACE	dev
ICS_REGISTRY	registry.ng.bluemix.net
ICS_NAMESPACE	demo
BMX_PWD
BMX_APIKEY
KUBE_SCRIPT_REPLICA_SET	2
KUBE_NAMESPACE	dev-space

- In the “Environment Properties (All Components)” section, check that the property matches the property defined in the screen capture.

Environment Properties (All Components)			
Version 2 of 2			
Add Property Batch Edit			
Name	Value	Description	Actions
<input type="text"/>	<input type="text"/>		
version.prop.name	<code> \${p?:currentVersion.name}</code>		Edit Delete

Lab 8.2: Update Environment Properties for TEST environment

- Click on TEST environment and check that the “Resources” section matches the configuration in the screen capture.

Configuration of node-app in TEST

Resources

Add Base Resources Select All... Actions... Show

	Name	Inventory	Status	Description
	Resource Name Tags			
	Node_Application / TEST			
::	BMX bmx x			
::	devops_local_agent (View Agent)		Online	

Refresh Print

2. In the “**Environment Properties**” section, update the value for the following properties.

- a. BMX_USER = <IBM Cloud User Email>
- b. BMX_ACCTID = <use the IBM Cloud Account ID from Lab 4 Step 6>
- c. BMX_ORG = <the default value is the same as the BMX_USER value>
- d. ICS_NAMESPACE = <IBM Container Service Namespace>
- e. BMX_APIKEY = <use the API Key generated from Lab 4 Step 5>
- f. KUBENAMESPAE = test-space

Environment Properties

Use Batch Edit Mode

KUBECLUSTER	mycluster
KUBECONFIG	/home/devops/.bluemix/plugins/container-service/clu:
BMX_API	api.ng.bluemix.net
BMX_USER	instructor0490@ibmlearning.org
BMX_ACCTID	343021f8c8c74357d5e000fe3216d76e
BMX_ORG	instructor0490@ibmlearning.org
BMX_SPACE	dev
ICS_REGISTRY	registry.ng.bluemix.net
ICS_NAMESPACE	demo
BMX_PWD
BMX_APIKEY
KUBE_SCRIPT_REPLICA_SET	2
KUBE_NAMESPACE	test-space

3. In the “Environment Properties (All Components)” section, check that the property matches the property defined in the screen capture.

Environment Properties (All Components)			
Version 2 of 2			
<input type="button" value="Add Property"/> <input type="button" value="Batch Edit"/>			
Name	Value	Description	Actions
version.prop.name	\${p?:currentVersion.name}		Edit Delete

Lab 8.3: Update Environment Properties for QA environment

1. Click on TEST environment and check that the “Resources” section matches the configuration in the screen capture.

Configuration of node-app in QA						
Resources						
		<input type="button" value="Add Base Resources"/> <input type="button" value="Select All..."/> <input type="button" value="Actions..."/> <input type="button" value="Show"/>				
		Name	Inventory	Status	Description	
		Resource Name				
		Tags				
		Node_Application / QA				
..	..	AWS				
..	..	BMX	bmx			
..	..	devops_local_agent	(View Agent)	Online		
..	..	Node_Application	(View Component)			
..	..	ICP	icp			
..	..	devops_local_agent	(View Agent)	Online		
..	..	Node_Application	(View Component)			

2. In the “Environment Properties” section, update the value for the following properties.

- BMX_USER = <IBM Cloud User Email>
- BMX_ACCTID = <use the IBM Cloud Account ID from Lab 4 Step 6>
- BMX_ORG = <the default value is the same as the BMX_USER value>
- ICS_NAMESPACE = <IBM Container Service Namespace>
- BMX_APIKEY = <use the API Key generated from Lab 4 Step 5>

f. KUBENAMESPAE = dev-space

Environment Properties	
Use Batch Edit Mode	
KUBECLUSTER	mycluster
KUBECONFIG	/home/devops/.bluemix/plugins/container-service/clu...
BMX_API	api.ng.bluemix.net
BMX_USER	instructor0490@ibmlearning.org
BMX_ACCTID	343021f8c8c74357d5e000fe3216d76e
BMX_ORG	instructor0490@ibmlearning.org
BMX_SPACE	dev
ICS_REGISTRY	registry.ng.bluemix.net
ICS_NAMESPACE	demo
BMX_PWD	****
BMX_APIKEY	****
KUBE_SCRIPT_REPLICA_SET	2
KUBE_NAMESPACE	qa-space

3. In the “Environment Properties (All Components)” section, check that the property matches the property defined in the screen capture.

Environment Properties (All Components)			
Version 2 of 2			
Add Property Batch Edit			
Name	Value	Description	Actions
<input type="text"/>	<input type="text"/>		
version.prop.name	\$[p?:currentVersion.name]		Edit Delete

Lab 9: Update IBM UrbanCode Deploy Resource

1. From within the DevOps VM, launch the Firefox browser and click on the IBM UCD bookmark.
2. Login to IBM UrbanCode Deploy using **admin / admin**.
3. Click on Resources tab and delete the “node_app_demo” component under <Environment> > BMX > devops_local_agent path where Environment = DEV, TEST and QA.

The screenshot shows the IBM UrbanCode Deploy Resources tree. The 'node_app_demo' component is selected in the QA environment under the BMX environment. A context menu is open over the component, with the 'Delete' option highlighted.

4. Add the “node-app” component to the resource tree for DEV environment. Hover the mouse over the “devops_local_agent” and click on Actions > Add Component.

The screenshot shows the IBM UrbanCode Deploy Resources tree. The 'node_app_demo' component is selected in the QA environment under the BMX environment. A context menu is open over the component, with the 'Add Component' option highlighted.

Select Component = node-app and click Save.

Create Resource [?](#) X

Component * node-app

Name * node-app

Description

Inherit Teams From Parent

Teams

Default Impersonation

Default impersonation can be configured here. Any steps which do not specify their own impersonation settings will fall back to the settings provided here.

Role Properties: node-app

Property	Description	Value	Actions
Show Filters			
No properties are available on this role.			

[Refresh](#)

Save **Cancel**

Repeat the steps for the TEST and QA resource tree.

5. In the UCD UI, update the Application processes to reference the updated component.

For example, click on Applications > node-app > processes and select Pipeline_Deployment.

Dashboard | Components | Applications | Configuration | Processes | Resources | Calendar |

Home > Applications > node-app

Application: node-app [\(show details\)](#)

Environments | History | Configuration | Components | Blueprints | Snapshots | **Processes** | Cal

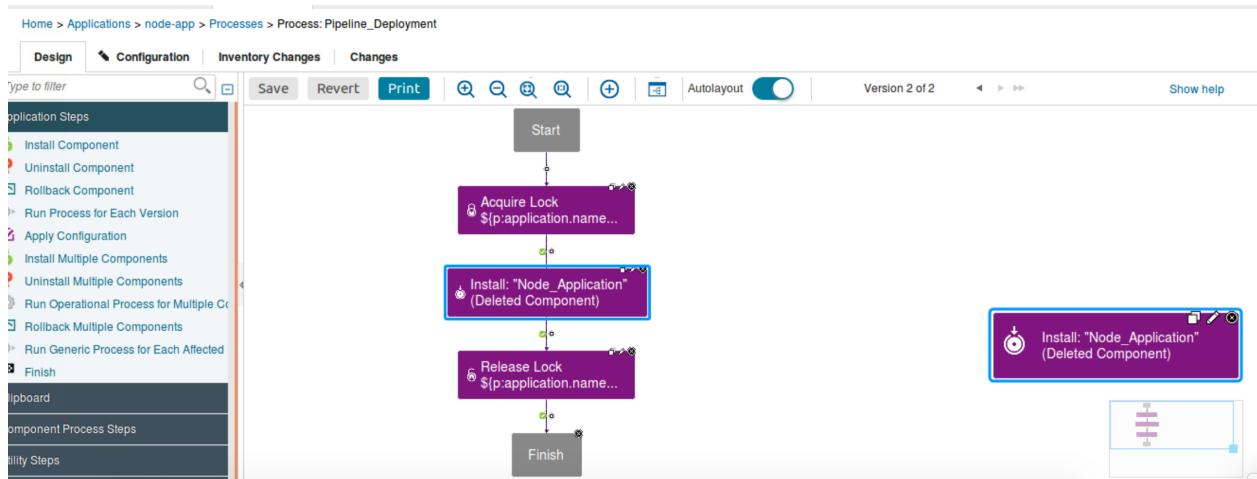
Create Process

Process	Description
BMX_Uninstall	
Clear_Env_Inventory	
Deploy_Cloud	
ICP_Uninstall	
Pipeline_Deployment	
Run_Functional_Test	
Run_Smoke_Test	

7 records - [Refresh](#) [Print](#)

1 / 1

Select the "Install: Node_Application" step and click the pencil icon to edit the step.



Select the following and click OK, then click Save.

Component = node-app
Component Process = Pipeline_Deployment (Template)

Edit Properties for Install Component

Name *	Install: "Node_Application"
Component *	node-app
Use Versions Without Status *	Active
Component Process *	Pipeline_Deployment (Template)
Limit to Resource Tag	bmx
Maximum number of concurrent processes *	-1

6. Repeat the steps to update the Component for the following Application processes:
a. BMX_Uninstall:

Component = node-app
Component Process = Uninstall (Template)

Edit Properties for Uninstall Component

Name *	Uninstall: "Node_Application"
Component *	node-app
Remove Versions With Status *	Active
Component Process *	Uninstall (Template)
Limit to Resource Tag	bmx
Uninstall Type *	All Selected For Process
Maximum number of concurrent processes *	-1

OK **Cancel**

b. Clear_Env_Inventory

Component = node-app
Component Process = Uninstall (Template)

Edit Properties for Uninstall Component

Name *	Uninstall: "Node_Application"
Component *	node-app
Remove Versions With Status *	Active
Component Process *	Uninstall (Template)
Limit to Resource Tag	
Uninstall Type *	All Existing
Maximum number of concurrent processes *	-1

OK **Cancel**

c. Deploy_Cloud

Component = node-app
Component Process = Deploy_Cloud (Template)

Edit Properties for Install Component

Name *	Install: "Node_Application" on IBM Cloud
Component *	node-app
Use Versions Without Status *	Active
Component Process *	Deploy_Cloud (Template)
Limit to Resource Tag	bmx
Maximum number of concurrent processes *	-1

OK Cancel

Component = node-app
Component Process = Deploy_Cloud (Template)

Edit Properties for Install Component

Name *	Install: "Node_Application" on ICP
Component *	node-app
Use Versions Without Status *	Active
Component Process *	Deploy_Cloud (Template)
Limit to Resource Tag	icp
Maximum number of concurrent processes *	-1

OK Cancel

d. ICP_Uninstall

Component = node-app
Component Process = Uninstall (Template)

Edit Properties for Uninstall Component

Name *	Uninstall:"Node_Application"
Component *	node-app
Remove Versions With Status *	Active
Component Process *	Uninstall (Template)
Limit to Resource Tag	icp
Uninstall Type *	All Selected For Process
Maximum number of concurrent processes *	-1

OK Cancel

e. Run_Functional_Test

Component = node-app

Component Process = Run_Functional_Test (Template)

Edit Properties for Run Process for Each Version

Name *	Functional Test
Component *	node-app
Component Process *	Run_Functional_Test (Template)
Limit to Resource Tag	bmx
Maximum number of concurrent processes *	-1
Fail Fast	<input type="checkbox"/>
Run on First Online Resource Only	<input type="checkbox"/>

OK Cancel

f. Run_Smoke_Test

Component = node-app

Component Process = Run_Smoke_Test (Template)

Edit Properties for Run Process for Each Version

Name *	Smoke Test
Component *	node-app
Component Process *	Run_Smoke_Test (Template)
Limit to Resource Tag	bmx
Maximum number of concurrent processes *	-1
Fail Fast	<input type="checkbox"/>
Run on First Online Resource Only	<input type="checkbox"/>

OK **Cancel**

- Click on Components > Templates > Multi-Cloud-Deployment-Component-Template > Processes and select the Uninstall process.

Dashboard Components Applications Configuration Processes Resources

Home > Component Templates > Multi-Cloud-Deployment-Component-Template

Component Template: Multi-Cloud-Deployment-Component-Temp

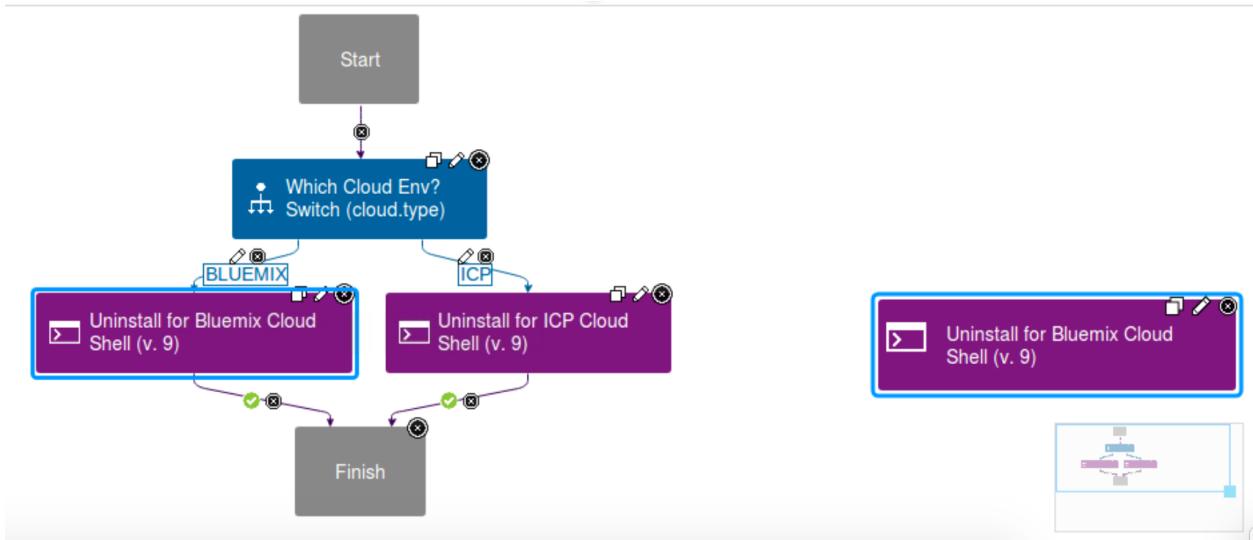
Components Configuration **Processes** Changes

Create Process

Process	Description
Deploy_Cloud	Deploy to IBM Cloud and/or AWS
Pipeline_Deployment	Pipeline Deployment
Remove_Inventory	Delete all component versions deployed to an env
Run_Functional_Test	Run Functional Test
Run_Smoke_Test	Run Smoke Test
Uninstall	Uninstalls for Kube for Bluemix or AWS

6 records - Refresh Print

Click Edit and select the “Uninstall for IBM Cloud” process step. Click the pencil icon to edit.



Select the Shell Script and delete “-p \${p:BMX_PWD}” from the bx login command.

Edit Script

```

Save | ⌘ ⌘ ⌘ ⌘ ⌘ ⌘ Go to line [ ] | ⌚ ⌚ | #
1 echo "Remove version ${p:version.name} from inventory."
2 echo "No operations required since Kube script deployed using kubectl apply command."
3
4 # Add Bluemix CLI to PATH
5 #export PATH=$PATH:/usr/local/bin
6
7 # Bluemix Login
8 #bx login -a ${p:BMX_API} -u ${p:BMX_USER} -o ${p:BMX_ORG} -s ${p:BMX_SPACE} -c ${p:BMX_ACCTID}
9
10 # Initialize IBM Container Service
11 #bx cs init
12
13 # Download config file and certificate for Kubernetes Cluster

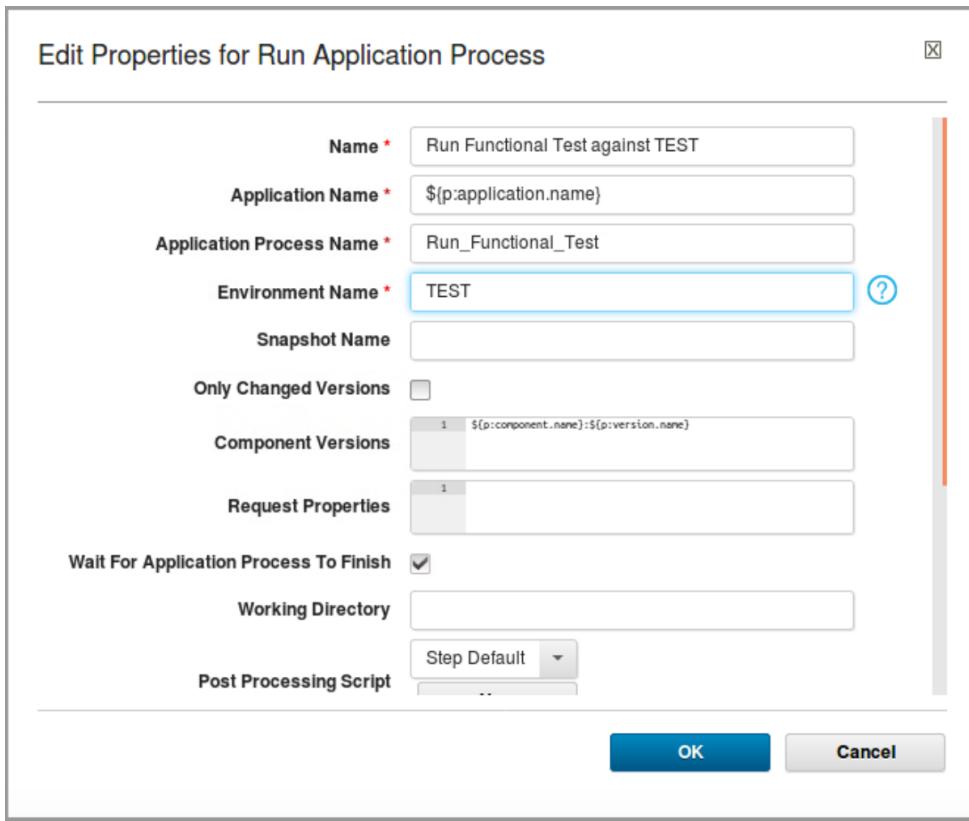
```

8. Update the Pipeline_Deployment component process. Fom the UCD UI, select Component > Templates > Multi-Cloud-Deployment-Component-Template > Processes > Pipeline_Deployment.

Process	Description
Deploy_Cloud	Deploy to IBM Cloud and/or AWS
Pipeline_Deployment	Pipeline Deployment
Remove_Inventory	Delete all component versions deployed to an env
Run_Functional_Test	Run Functional Test
Run_Smoke_Test	Run Smoke Test
Uninstall	Uninstalls for Kube for Bluemix or AWS

Edit the “Run Functional Test against TEST” component process step and edit Environment Name to

“TEST”. Click OK and Save.



Assign custom port to DEV vs TEST vs QA for the nodePort deployment using Kubernetes. Fom the UCD UI, select Component > Templates > Multi-Cloud-Deployment-Component-Template > Processes > Deploy_Cloud and click Edit.

Home > Component Templates > Multi-Cloud-Deployment-Component-Template

Component Template: Multi-Cloud-Deployment-Component-Template (show details)

Components | Configuration | **Processes** | Changes

Create Process

Process	Description
Deploy_Cloud	Deploy to IBM Cloud and/or AWS
Pipeline_Deployment	Pipeline Deployment
Remove_Inventory	Delete all component versions deployed to an env

Click the “Retrieve Kube cluster config from IBM Cloud” process step and select the pencil icon to edit the step. Within the shell script, replace the following script and click Save, OK and Save.

Before:

```
if [[ ${p:environment.name} == "TEST" ]];
```

```

then
    echo "Set port = 30081 for TEST"
    sed -i.bak s%nodePort: 30080%nodePort: 30081%g ${ICS_SCRIPT_NAME}
elif [[ ${p:environment.name} == "QA" ]];
then
    echo "Set port = 30082 for QA"
    sed -i.bak s%nodePort: 30080%nodePort: 30082%g ${ICS_SCRIPT_NAME}
else
    # Leave port = 30080 for DEV
    echo "Set port to 30080 for DEV"
fi

```

After:

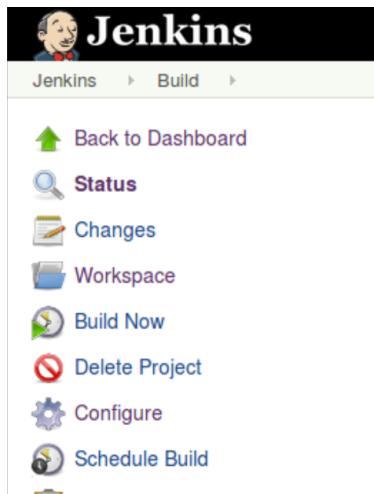
```

if [[ ${p:environment.name} == "TEST" ]];
then
    echo "Set port = 30081 for TEST"
    sed -i.bak s%30080%30081%g ${ICS_SCRIPT_NAME}
elif [[ ${p:environment.name} == "QA" ]];
then
    echo "Set port = 30082 for QA"
    sed -i.bak s%30080%30082%g ${ICS_SCRIPT_NAME}
else
    # Leave port = 30080 for DEV
    echo "Set port to 30080 for DEV"
fi

```

Lab 10: Run the CICD pipeline

1. On the DevOps VM, open a browser and click on the Jenkins bookmark. Select the Build job and click "Build now".



2. Click on the new build and review the Console Output.

```
604c78617f34: Pushed
53c779688d06: Layer already exists
60a0858edcd5: Layer already exists
b6ca02dfe5e6: Layer already exists
fa18e5ffd316: Pushed
1: digest: sha256:9c6765a126dfb18af99c5d5d55995ff1a829447f1e57bebcf39320c31306aea2 size: 2423
Logging out...
OK
Update Kubernetes script files for Bluemix and AWS
Creating new component version 'node-app:1' on component 'node-app'
Successfully created component version with UUID 'abd97701-2a6d-4964-9eb2-65c1eb0bd35e'
Uploading files to version 'node-app:1' on component 'node-app'
Successfully uploaded files
Setting properties for version 'node-app:1' on component 'node-app'
Creating component version link 'Jenkins Build #1' to URL 'http://devops:8080/job/Build/1/'
Deploying component versions '{node-app=[node-app:1]}'
Starting deployment process 'Pipeline_Deployment' of application 'node-app' in environment 'DEV'
Deployment request id is: '1623a742-6e39-4ddf-96bb-a200839b91b8'
Deployment is running. Waiting for UCD Server feedback.
Finished the deployment in 6 seconds
The deployment result is SUCCEEDED. See the UrbanCode Deploy deployment logs for details.
Finished: SUCCESS
```

3. Using the IBM Cloud cli, check to make sure a Docker image for the application got pushed into the Container Registry. The command to retrieve the list of images is:

```
bx cr login
bx cr images
```

```

root@devops:~# bx cr images
Plugin version '0.1.304' is now available. To update, run: bx plugin update container-registry -r Bluemix
Listing images...

REPOSITORY                                     NAMESPACE   TAG    DIGEST          CREATED        SIZE
VULNERABILITY STATUS
registry.ng.bluemix.net/labdemo/node-app      labdemo    1      9c6765a126df  5 minutes ago  263 MB
Vulnerable

```

4. Go to the IBM UrbanCode Deploy bookmark from a browser and click Components > node-app > Versions and verify that a component version for “node-app:<build number>” has been created.

The screenshot shows the IBM UrbanCode Deploy interface. The top navigation bar includes links for Dashboard, Components, Applications, Configuration, and Processes. Below this, a breadcrumb trail indicates the current location: Home > Components > node-app. The main content area is titled "Component: node-app (show details)". Under the "Versions" tab, there is a table with two columns: "Version" and "Statuses". The table contains a single row with the value "node-app:1" in the Version column and "Statuses" in the Statuses column. A note at the bottom left says "1 record - Refresh Print".

5. From the browser, review the IBM UrbanCode Deploy pipeline process that got triggered by Jenkins. Click Applications > node-app > History. The pipeline process will have deployed the application to DEV, run smoke tests, deploy the application to TEST, and run functional tests.

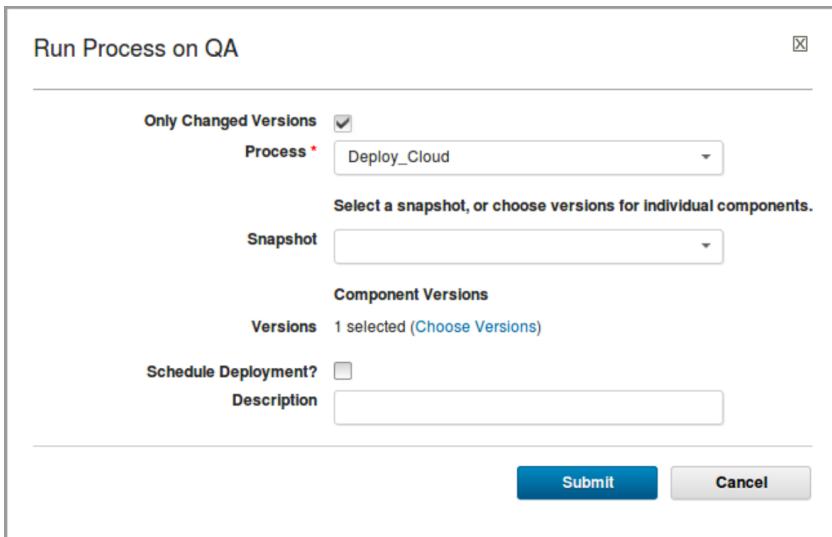
Process	Environment	Snapshot	Scheduled For	By	Status	Actions
Run_Functional_Test	TEST		3/18/2018, 3:38 PM	admin	Success	View Request
BMX_Uninstall	TEST		3/18/2018, 3:37 PM	admin	Success	View Request
Deploy_Cloud	TEST		3/18/2018, 3:37 PM	admin	Success	View Request
Run_Smoke_Test	DEV		3/18/2018, 3:36 PM	admin	Success	View Request
BMX_Uninstall	DEV		3/18/2018, 3:36 PM	admin	Success	View Request
Deploy_Cloud	DEV		3/18/2018, 3:35 PM	admin	Success	View Request
Pipeline_Deployment	DEV		3/18/2018, 3:35 PM	admin	Success	View Request

6. Assuming the deployment to QA is manual and not automated, a user can run the deployment process in IBM UrbanCode Deploy to promote the build to the QA environment.

In the UCD UI, select Applications > node-app and click the play button for the QA environment.

The screenshot shows the IBM UrbanCode Deploy interface. The top navigation bar includes links for Environments, History, Configuration, and Components. Below this, a breadcrumb trail indicates the current location: Home > Applications > node-app. The main content area is titled "Application: node-app (show details)". Under the "Environments" tab, there is a table with three rows. Each row contains icons for play, stop, and refresh, followed by the environment name: DEV, TEST, and QA respectively. The environments are color-coded: yellow for DEV, orange for TEST, and blue for QA.

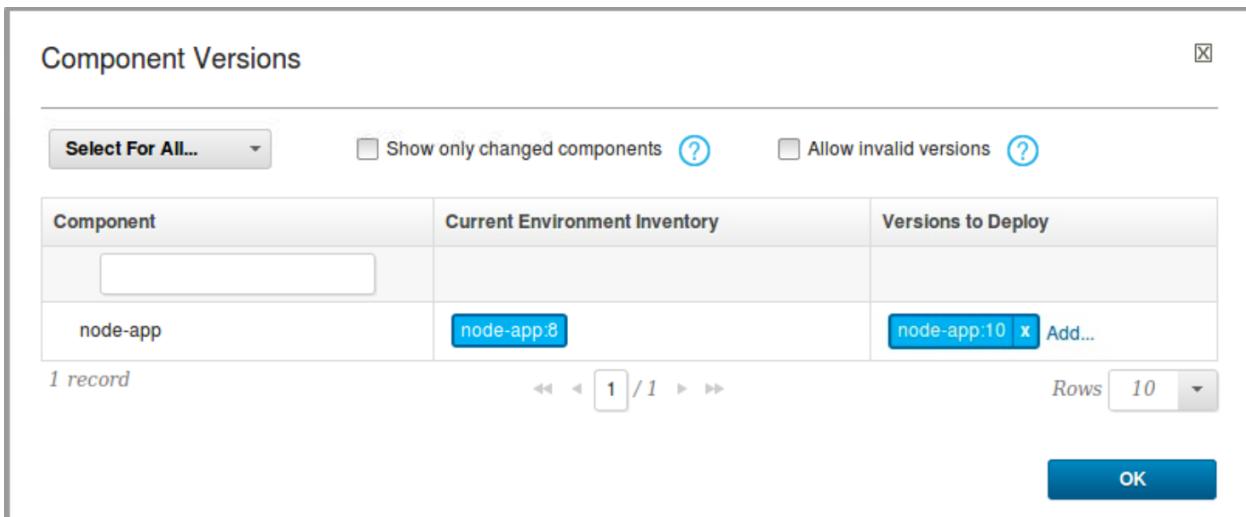
In the “Run Process on QA” window, select the “**Deploy_Cloud**” process and click Choose Version.



The dialog box titled "Run Process on QA" contains the following fields:

- Only Changed Versions**: A checked checkbox.
- Process ***: A dropdown menu set to "Deploy_Cloud".
- Select a snapshot, or choose versions for individual components.**
- Snapshot**: A dropdown menu.
- Component Versions**
- Versions**: "1 selected" with a link to "Choose Versions".
- Schedule Deployment?**: An unchecked checkbox.
- Description**: An empty text input field.
- Submit** and **Cancel** buttons at the bottom.

Under the “Select For All...” dropdown, select the “Latest available” and click OK and Submit.



The dialog box titled "Component Versions" contains the following interface:

- Select For All...**: A dropdown menu.
- Show only changed components**: An unchecked checkbox.
- Allow invalid versions**: An unchecked checkbox.
- Component**, **Current Environment Inventory**, and **Versions to Deploy** tables.
- node-app** in the Component column.
- node-app:8** in the Current Environment Inventory column.
- node-app:10** in the Versions to Deploy column, with a delete icon and "Add..." button.
- 1 record** status.
- Pagination controls: <<, <, 1 / 1, >, >>.
- Rows**: A dropdown menu set to 10.
- OK** button at the bottom right.

Once the process finishes running, check the component versions in the UCD Environments and the same node-app version should be deployed on all the environments now.

<p>Snapshot: None Blueprint: DEV Compliancy 1 / 1 3 Environment</p> <table border="1"> <thead> <tr> <th>Component</th><th>Version</th><th>Date Deployed</th><th>Compliancy</th><th>Actions</th></tr> </thead> <tbody> <tr> <td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>node-app</td><td>node-app:10 (View Details)</td><td>3/18/2018, 3:35 PM</td><td>Compliant (1/1)</td><td>View Request</td></tr> </tbody> </table> <p>Refresh Print</p>	Component	Version	Date Deployed	Compliancy	Actions						node-app	node-app:10 (View Details)	3/18/2018, 3:35 PM	Compliant (1/1)	View Request	<p>Snapshot: None Blueprint: TEST Compliancy 1 / 1</p> <table border="1"> <thead> <tr> <th>Component</th><th>Version</th><th>Date Deployed</th><th>Compliancy</th><th>Actions</th></tr> </thead> <tbody> <tr> <td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>node-app</td><td>node-app:10 (View Details)</td><td>3/18/2018, 3:37 PM</td><td>Compliant (1/1)</td><td>View Request</td></tr> </tbody> </table> <p>Refresh Print</p>	Component	Version	Date Deployed	Compliancy	Actions						node-app	node-app:10 (View Details)	3/18/2018, 3:37 PM	Compliant (1/1)	View Request	<p>Snapshot: None Blueprint: QA Compliancy 1 / 1</p> <table border="1"> <thead> <tr> <th>Component</th><th>Version</th><th>Date Deployed</th><th>Compliancy</th><th>Actions</th></tr> </thead> <tbody> <tr> <td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>node-app</td><td>node-app:10 (View Details)</td><td>3/18/2018, 3:40 PM</td><td>Compliant (1/1)</td><td>View Request</td></tr> </tbody> </table> <p>Refresh Print</p>	Component	Version	Date Deployed	Compliancy	Actions						node-app	node-app:10 (View Details)	3/18/2018, 3:40 PM	Compliant (1/1)	View Request
Component	Version	Date Deployed	Compliancy	Actions																																											
node-app	node-app:10 (View Details)	3/18/2018, 3:35 PM	Compliant (1/1)	View Request																																											
Component	Version	Date Deployed	Compliancy	Actions																																											
node-app	node-app:10 (View Details)	3/18/2018, 3:37 PM	Compliant (1/1)	View Request																																											
Component	Version	Date Deployed	Compliancy	Actions																																											
node-app	node-app:10 (View Details)	3/18/2018, 3:40 PM	Compliant (1/1)	View Request																																											

7. Check the pods running in the “dev-space”, “test-space” and “qa-space” namespace within the IBM Cloud Kubernetes cluster and verify kubernetes pods are running successfully.

Using the IBM Cloud cli commands from earlier, verify the pods are running successfully in each of the three namespaces.

kubectl get pods --namespace=dev-space

```
root@devops:~/Documents/Hollis/nodejs-cloudant-demo# kubectl get pods --namespace=dev-space
NAME                               READY   STATUS    RESTARTS   AGE
nodejs-cloudant-demo-deployment-cd778f68-25sht   1/1     Running   0          5m
nodejs-cloudant-demo-deployment-cd778f68-prpkf   1/1     Running   0          5m
root@devops:~/Documents/Hollis/nodejs-cloudant-demo#
```

kubectl get pods --namespace=test-space

```
root@devops:~/Documents/Hollis/nodejs-cloudant-demo# kubectl get pods --namespace=test-space
NAME                               READY   STATUS    RESTARTS   AGE
nodejs-cloudant-demo-deployment-cd778f68-hkqzf   1/1     Running   0          6m
nodejs-cloudant-demo-deployment-cd778f68-ml5df   1/1     Running   0          6m
root@devops:~/Documents/Hollis/nodejs-cloudant-demo#
```

kubectl get pods --namespace=qa-space

```
root@devops:~/Documents/Hollis/nodejs-cloudant-demo# kubectl get pods --namespace=qa-space
NAME                               READY   STATUS    RESTARTS   AGE
nodejs-cloudant-demo-deployment-cd778f68-cgw7m   1/1     Running   0          4m
nodejs-cloudant-demo-deployment-cd778f68-slhnw   1/1     Running   0          4m
root@devops:~/Documents/Hollis/nodejs-cloudant-demo#
```

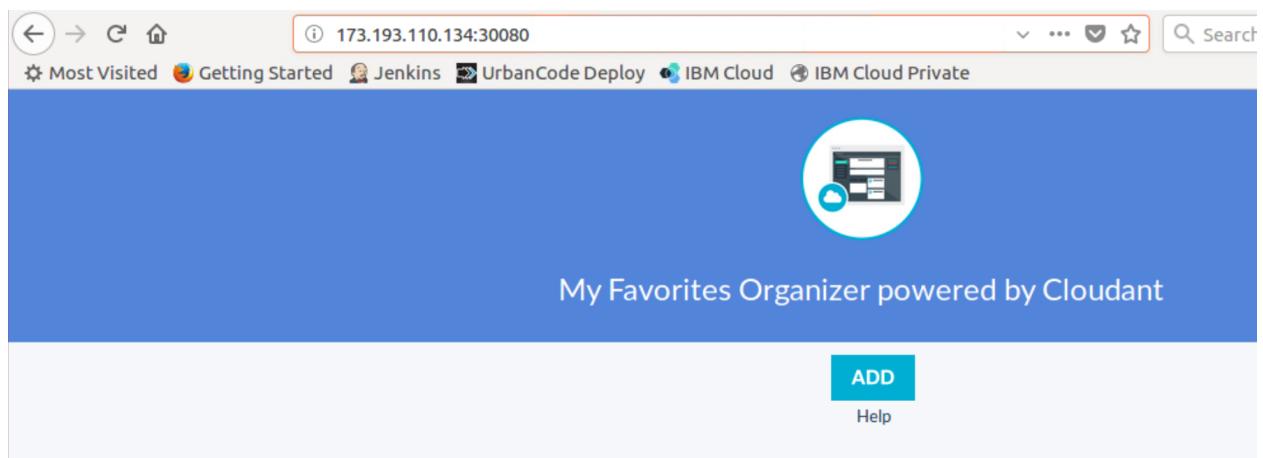
8. Retrieve the public ip of the worker node and the nodePort used to access the running application. bring up the URL to the node.js application using nodePort.

bx cs workers mycluster

```
root@devops:~/Documents/Hollis/nodejs-cloudant-demo# bx cs workers mycluster
OK
ID                           Public IP      Private IP    Machine Type  State   Status  Zone
Version
kube-hou02-pa63b83389868a4cf4b3f1f592b8caeef63-w1  173.193.110.134  10.47.79.241  free        normal  Ready   hou02
1.8.8_1507
```

Here are the ports used for the different environments DEV (30080), TEST (30081) and QA (30082).

Open a browser and go to the URL `http://<Worker node public ip>:<nodePort>` to get to the application in the different environments.



Congratulations, you have completed all the lab exercises.

We Value Your Feedback!

- Don't forget to submit your Think 2018 session and speaker feedback! Your feedback is very important to us – we use it to continually improve the conference.
- Access the Think 2018 agenda tool to quickly submit your surveys from your smartphone, laptop or conference kiosk.