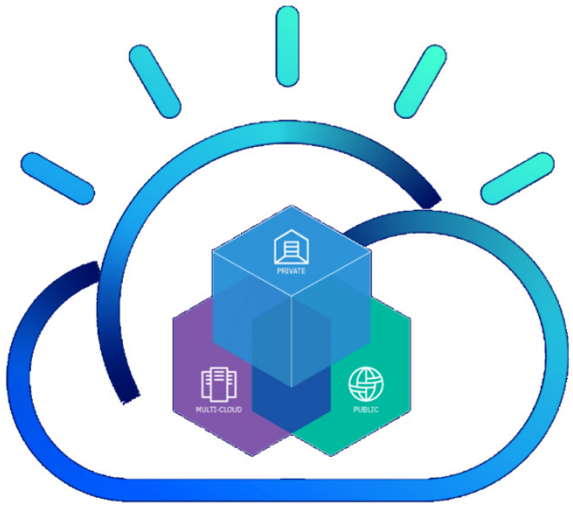
Watson Studio Local

Administration

Management



Feb. 18, 2019

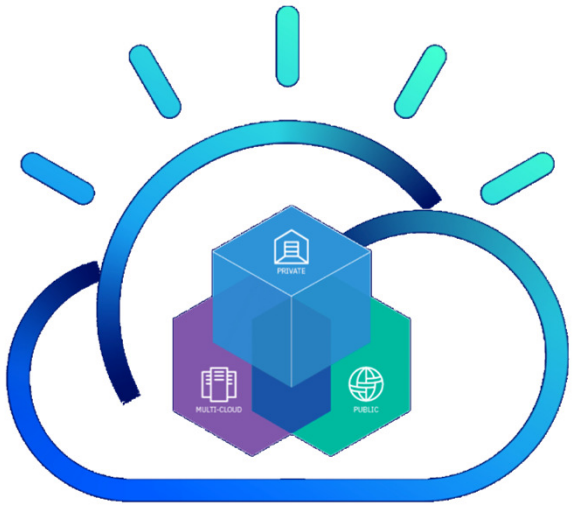
© 2019 IBM Corporation

Agenda

- **Architecture**
- **Cluster Administration**
  - Users
  - Nodes
  - Services
  - Pods
  - Alerts
  - Security & Auditing
  - SSHD
  - Replace Nodes
- **Manage Libraries and Packages**
- **Hadoop Integration**
- **Troubleshooting**
- **Health Check**
- **Image Management**
- **Deployment**
  - Assets Deployment
  - Manage a deployed job
  - Deployment dashboard
- **Additional Topics**
  - Kubernetes 101
  - Best Practices

© 2019 IBM Corporation

Watson Studio Local
Administration Architecture



© 2019 IBM Corporation

Schedule Plan

09:00 – 10:30 Session I

10:30 – 10:45 Break

10:45 – 11:45 Session II

11:45 – 13:00 Lunch

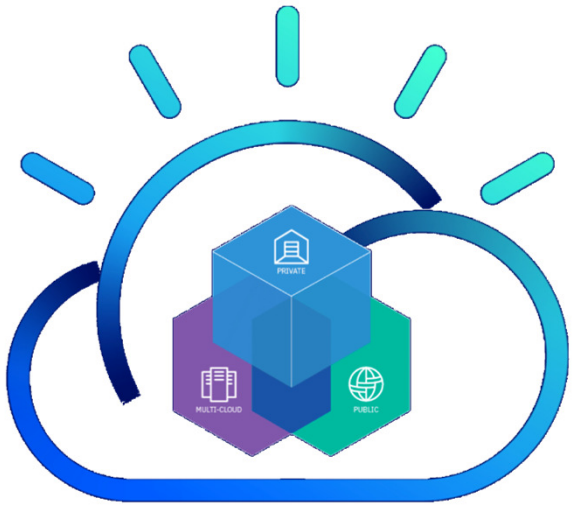
13:00 – 14:00 Session III

14:00 – 14:10 Break

14:10 – 15:30 Session IV

15:30 – 15:40 Break

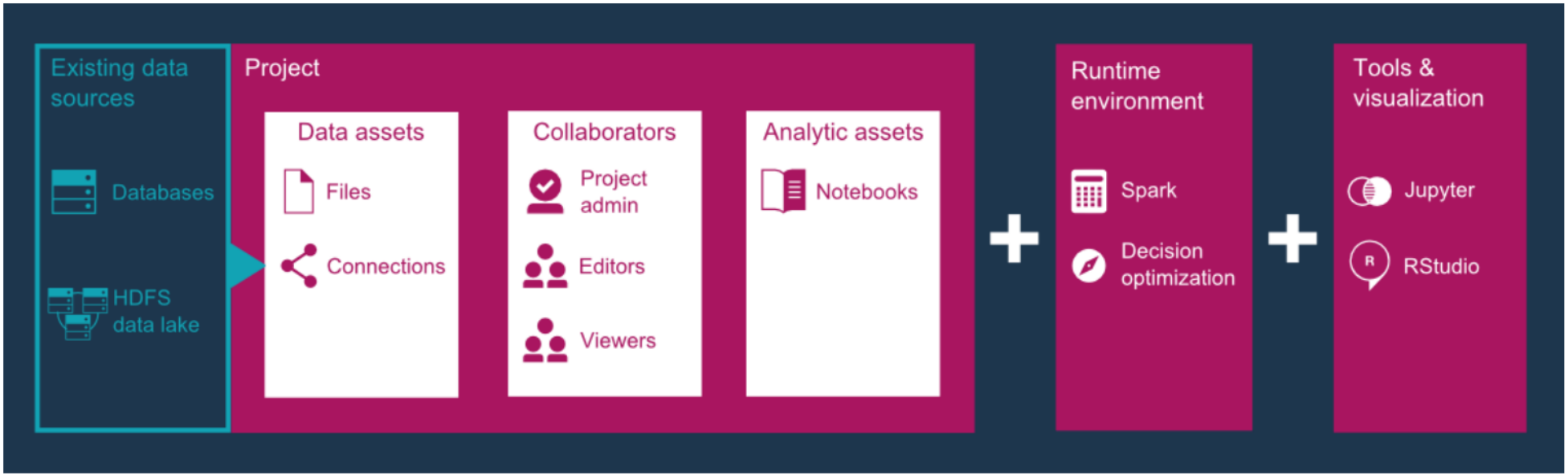
15:40 – 17:00 Session V



© 2019 IBM Corporation

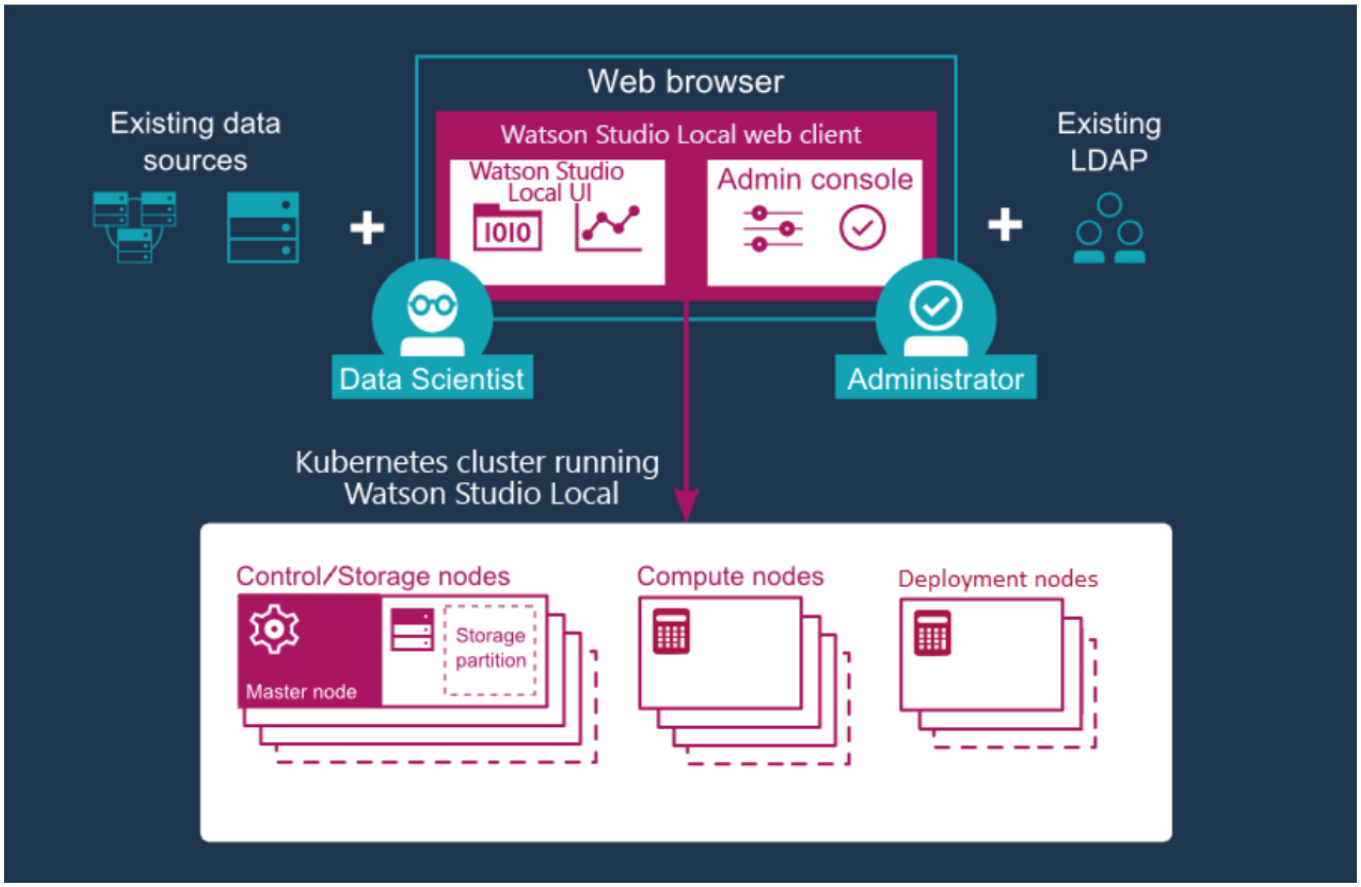
IBM Watson Studio Local (WSL) – Overview

- IBM Watson Studio Local is an out-of-the-box enterprise solution for data scientists and data engineers.
- It offers a suite of data science tools, such as RStudio, Spark, Jupyter, and Zeppelin notebooks, that are integrated with proprietary IBM technologies.



© 2019 IBM Corporation

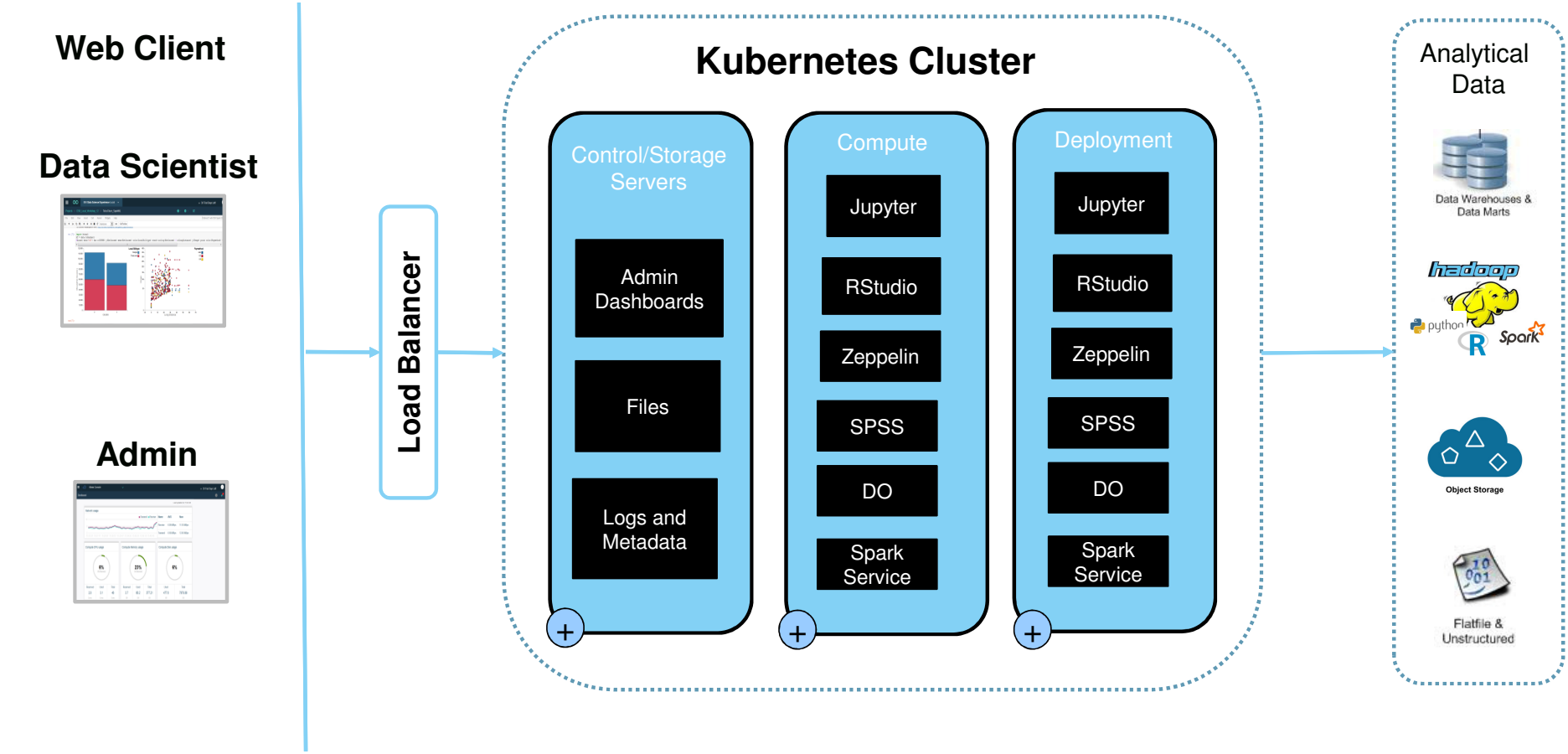
IBM Watson Studio Local (WSL) – Architecture



1. **HA is available by default. The deployments cannot be made HA for a single deployment node.**
2. **Vertical and horizontal scalability. Control/Compute and Deployment nodes can be added at any time.**
3. **LDAP authentication supported. Native authentication available.**
4. **Connect to data using JDBC, API or built-in connectors.**
5. **Supports Remote execution of Spark, Python, in R in a Hadoop cluster**

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Architecture



© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Cluster Configuration

- WSL cluster has several types of nodes (systems)
  - Control/Storage: manage the system and store data/metadata
  - Compute: run development workload (Notebooks, RStudio, etc.)
  - Deployment: dedicated to running production workload
- Cluster Configurations

Cluster Type	# Node breakdown	Notes
3 nodes	3 shared control / compute	Unable to deploy assets.
4 nodes	3 shared control / compute + 1 deploy	
5 nodes	3 shared control / compute + 2 deploy	
7 nodes	3 shared control + 3 compute + 1 deploy	
8 nodes	3 shared control + 3 compute + 2 deploy	
11 nodes	3 shared control + 6 compute + 2 deploy	

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Kubernetes

- WSL Local deploys into its own Kubernetes cluster or ICP (IBM Cloud Private)
  - Deployment into other clusters is not supported
- Kubernetes provides
  - High availability for services
  - Scalability
  - Monitor and display system status

© 2019 IBM Corporation

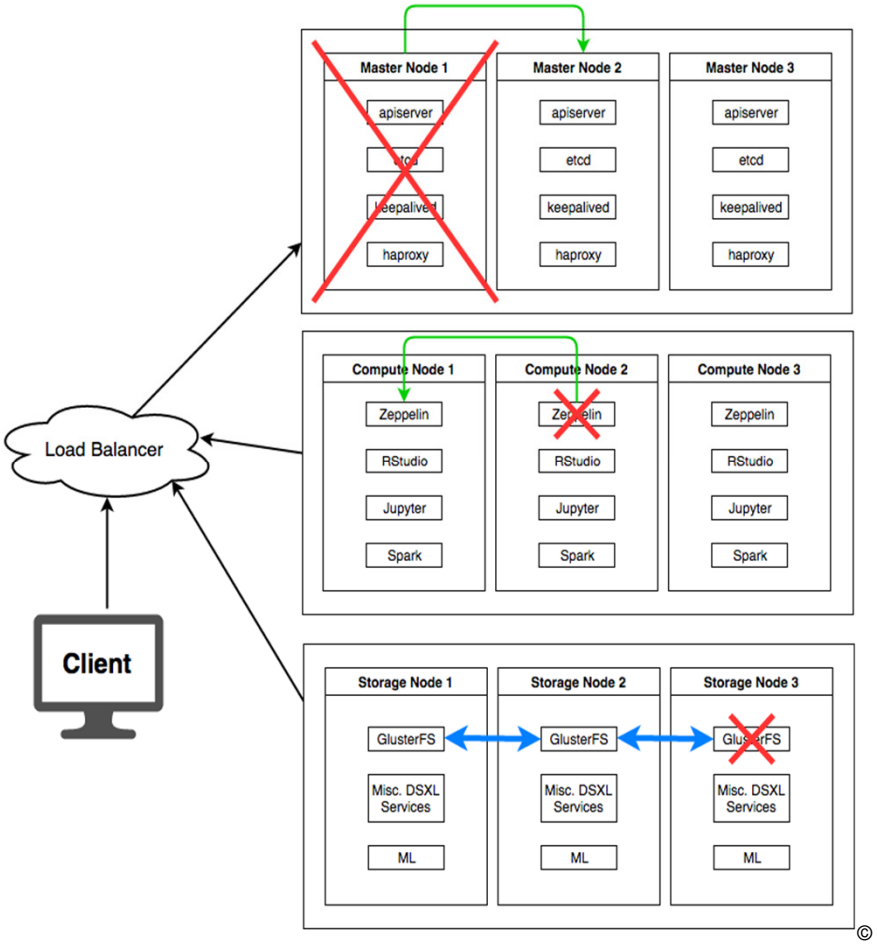
IBM Watson Studio Local (WSL) – High Availability

- Platform HA
  - By default, the development nodes (Control/Compute/Storage) are configured for HA
    - 3 nodes in 4-node configuration
    - 6 nodes in a 7-node configuration
  - For Deployment nodes, the customer can configure active/active or active/passive HA
    - Active/active HA: at least 2 Deployment nodes
    - Active/passive HA: 1 Deployment node and 1 cold standby

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – High Availability

- Platform HA
  - WSL can tolerate failure of
    - 1 node in a 3-node configuration
    - 3 nodes in a 6-node configuration



© 2019 IBM Corporation

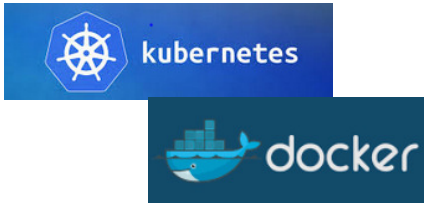
IBM Watson Studio Local (WSL) – High Availability

- Service HA
  - For services that are deployed as pods, Kubernetes will monitor and redeploy services when they are down
  - No session failover, which means that some services may be down for a few minutes
    - ✓ Any type of assets that requires a “session” (for example, batch scoring), will need to be restarted
  - For online scoring, it’s possible to do deployment to multiple environments
    - ✓ Ensures uninterrupted online scoring

© 2019 IBM Corporation

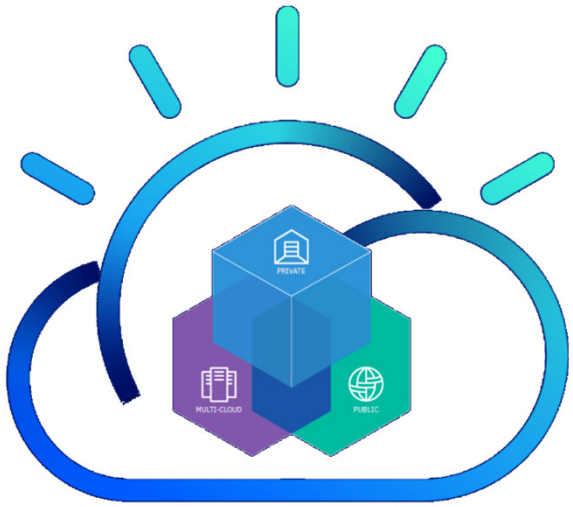
IBM Watson Studio Local (WSL) – Kubernetes and Docker

- Kubernetes and Docker are open source technologies
  - Used for deploying applications with microservices architecture
- Docker provides a container for services
  - Once the service is “dockerized”, it can run in any environment managed by Docker
- Kubernetes is a platform for managing *containers* (including Docker)
  - Deployment
  - Scalability
  - Environment status



© 2019 IBM Corporation

Watson Studio Local
Cluster Management



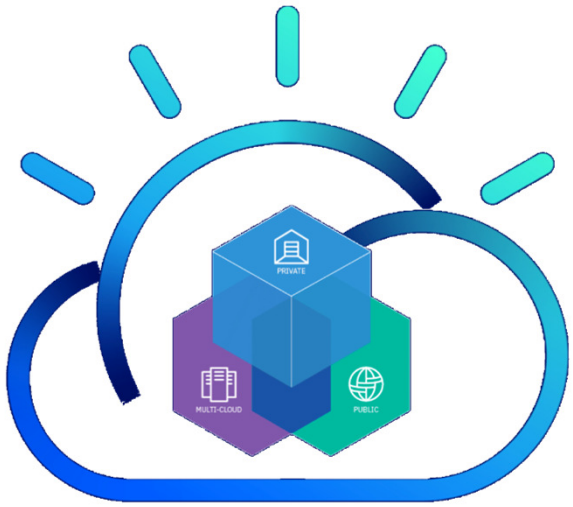
© 2019 IBM Corporation

Cluster Administration

- The Admin Console enables system administrators to manage users and to monitor the health of the application and the Kubernetes deployment environment
- Kubernetes and Docker do a lot of the work for you to manage your Watson Studio Local deployment. You don't need to be familiar with these technologies to manage Watson Studio Local
- To open the Admin Console:
  - Sign in to the WSL client from a web browser
  - Use the dropdown menu in the banner to select **Admin Console**

© 2019 IBM Corporation

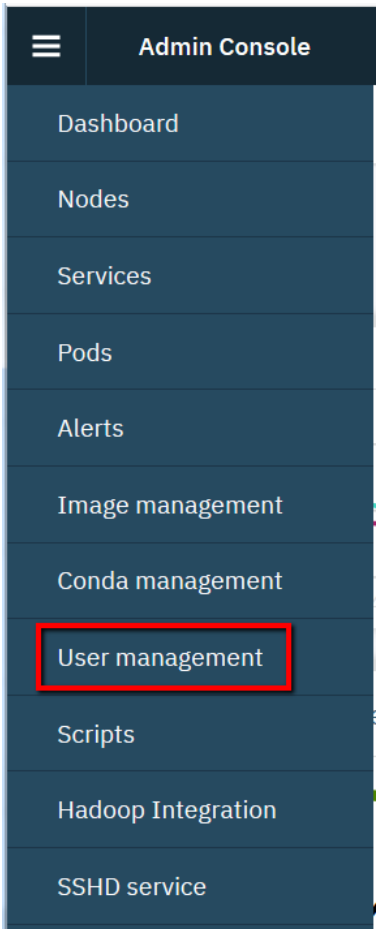
Watson Studio Local
Cluster Management
Users



© 2019 IBM Corporation

Manage Users

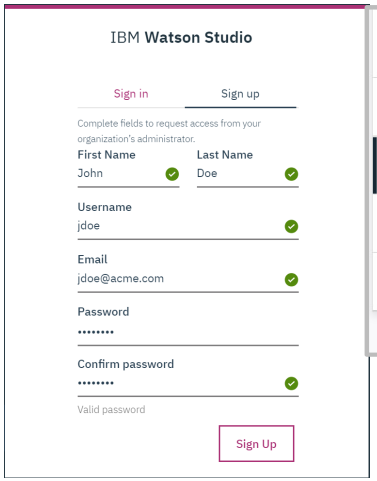
- WSL users can be managed from either an external LDAP server (recommended) or an internal repository database
- In the Admin Console, click the menu icon ( ) and click User Management to approve sign-up requests, add users, filter them, edit them, assign permissions to them, or delete them

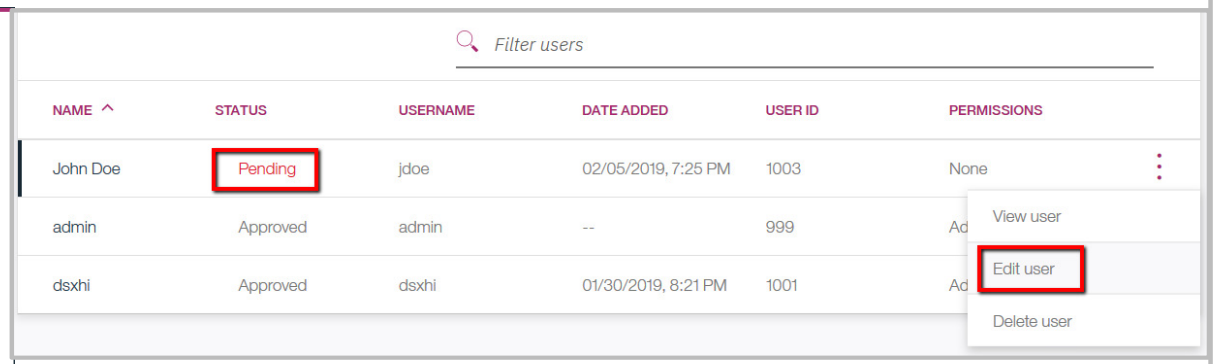


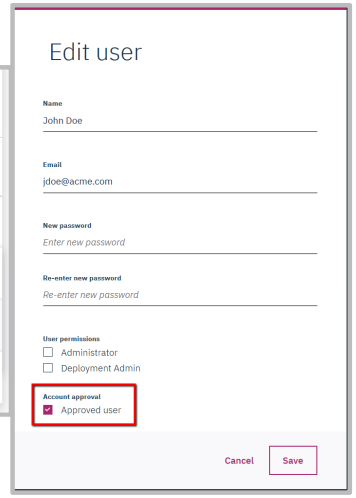
© 2019 IBM Corporation

Manage Users

- The user permissions are as follows:
  - **Admin:** Can sign in to both the admin dashboard and the WSL client
  - **Deployment Admin:** Can create project releases in IBM Watson Machine Learning
  - **User:** Can sign in to the WSL client only. This is the default permission if neither Admin check box is selected
- If new users request an account, you must approve them by editing them and selecting Approved User
  - Unapproved users have a status of pending







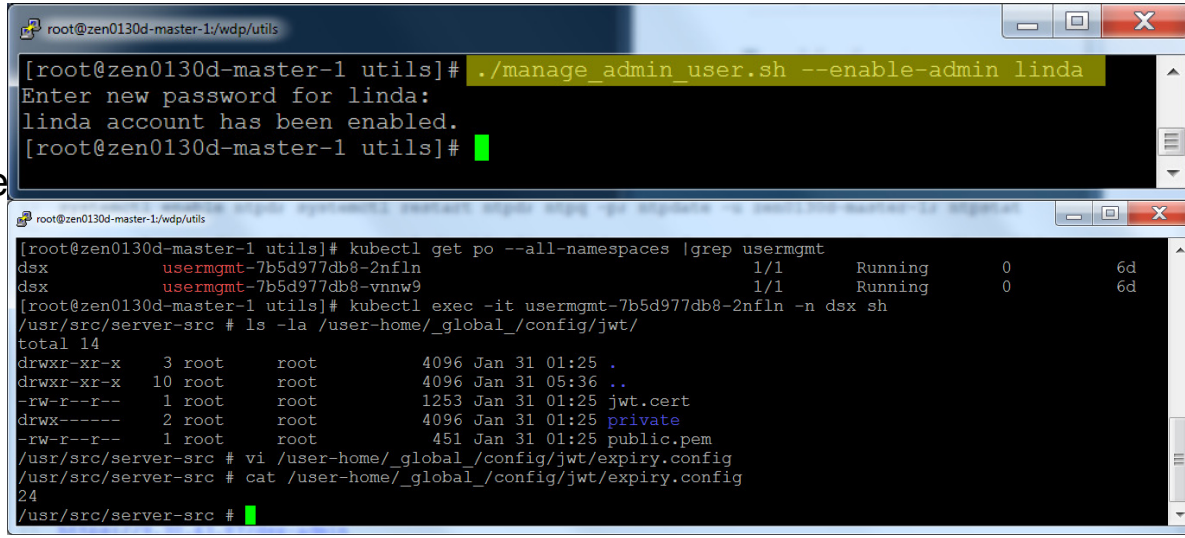
© 2019 IBM Corporation

Manage Users & Others

- Enable and set a new password for the WSL admin, enter the following command:
  - /wdp/utils/manage_admin_user.sh --enable-admin <userid>
- Disable a WSL admin, enter the following command:
  - /wdp/utils/manage_admin_user.sh --disable-admin <userid>
- Modify the session expiration time
  1. SSH to WSL cluster node
  2. Run kubectl exec to a usermgmt pod.
  3. Edit or create the following file to specify the number of hours. (default is 12 hours)

     - $cat /user-home/_global_/config/jwt/expiry.config

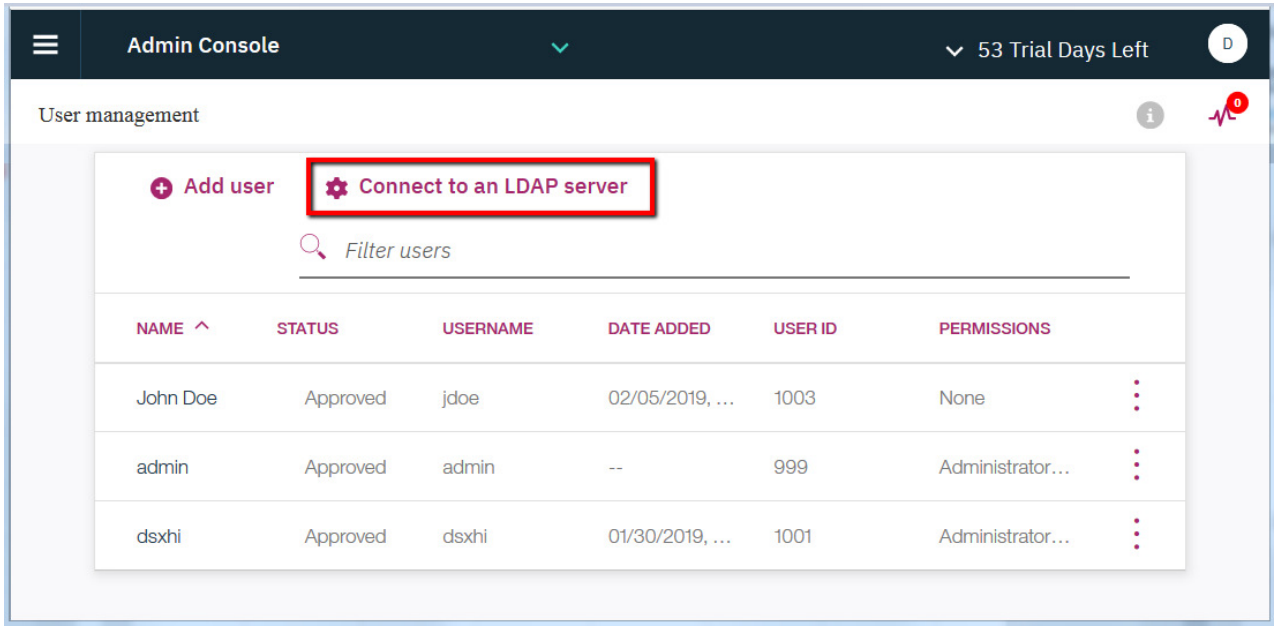
     24
  4. Restart usermgmt pod by deleting all of the current usermgmt pods.



© 2019 IBM Corporation

Manage Users – LDAP Server

- By default, WSL user records are stored in its internal repository database
- Alternatively, you can use your own external LDAP server instead. To set up your own LDAP server, click Connect to an LDAP server



© 2019 IBM Corporation

Manage Users – LDAP Server

- In the LDAP host field, use the *ldap://* prefix for a non-secure port and the *ldaps://* prefix for a secure port
- If you opt to authenticate LDAP with search, then specify the domain search user, password, and base. If you opt to authenticate by distinguished name without search, ensure the **LDAP Prefix and LDAP Suffix** fields match the distinguished name exactly. For example, uid for the prefix and ou = dsxusers, dc = ibm, dc = com for suffix for the setup to succeed
- When finished, click the **Set up LDAP** button. If the LDAP setup succeeds, WSL no longer displays password fields whenever you sign up a new user in the Admin Console.

© 2019 IBM Corporation

Manage Users – LDAP Server

- Because the WSL user records are stored in the external LDAP server, only the LDAP administrator can perform user management tasks like password resets and changes
- Note that after LDAP is enabled, both local and LDAP users can sign in, but only LDAP users can be added.
- **Tip**: To save time from approving LDAP users, you can select **Auto-Signup** to automatically approve all LDAP user sign-up requests.

© 2019 IBM Corporation

Manage Users – LDAP Server

- **Example "with search" configuration**

- **Notes:**
  - Domain search user and password are optional for testing connectivity
  - Username and Password for testing must be valid LDAP user

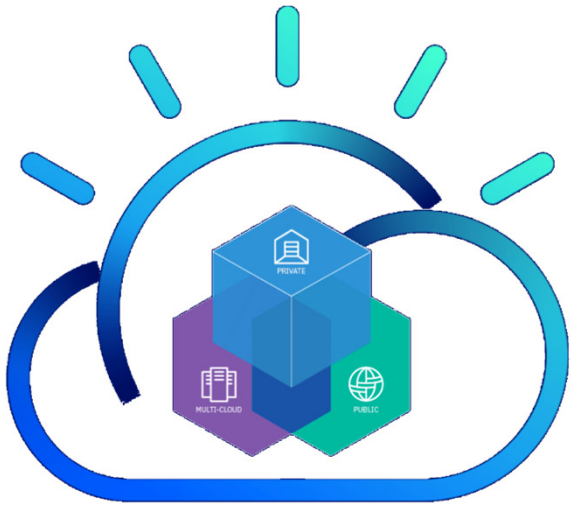
IBM Corporation

Manage Users – LDAP Server

- To verify that your LDAP connection works, type in an existing LDAP user in the Username for testing and Password for testing fields, then click the Test LDAP button
- When finished, click the Set up LDAP button
- If the LDAP setup succeeds, WSL no longer displays password fields whenever you sign up a new user in the admin dashboard
- Because the WSL user credentials are stored in the external LDAP server, only the LDAP administrator can perform LDAP user password resets and changes

© 2019 IBM Corporation

Watson Studio Local
Administration
Mange Nodes



© 2019 IBM Corporation

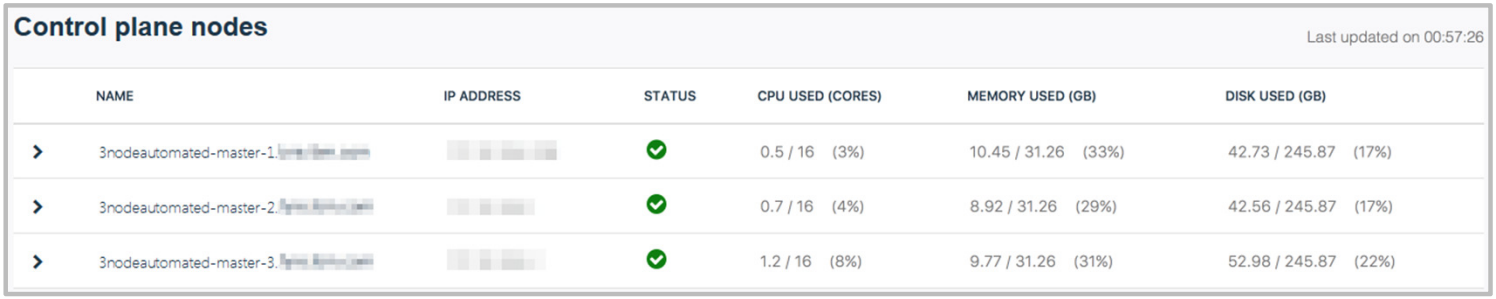
Monitor Cluster Nodes

- Your IBM Watson Studio Local deployment consists of up to three types of nodes
  - *Control/Storage/Compute Nodes (required)*
  - *Additional Compute Nodes (optional)*
  - *Additional Deployment Nodes (optional)*
- The cluster requires a minimum of three machines—set up as Control/Storage/Compute “master” nodes in the minimum three machine configuration—for ensuring platform high availability
- Storage is always part of the master nodes. In current versions of WSL, there are no independent storage nodes
- Additional node machines can be added to provide greater compute and deployment capability

© 2019 IBM Corporation

Manage Cluster Nodes

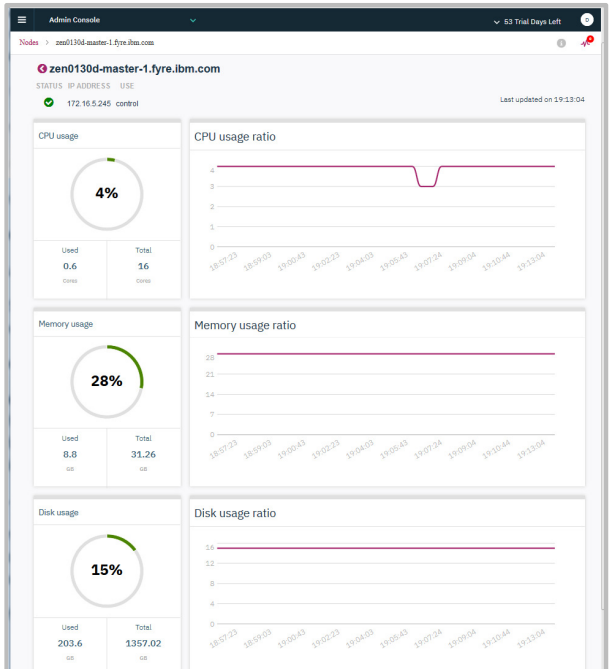
- You can check whether cluster nodes are running from the Nodes page
- You can access the Nodes page from the menu icon ( )
- For each node in the cluster, you can see:
  - The IP address of the node
  - Whether the node is running or down
  - How many CPU cores the node is using
  - The GB of memory the node is using
  - The GB of storage (disk) the node is using



© 2019 IBM Corporation

Manage Cluster Nodes

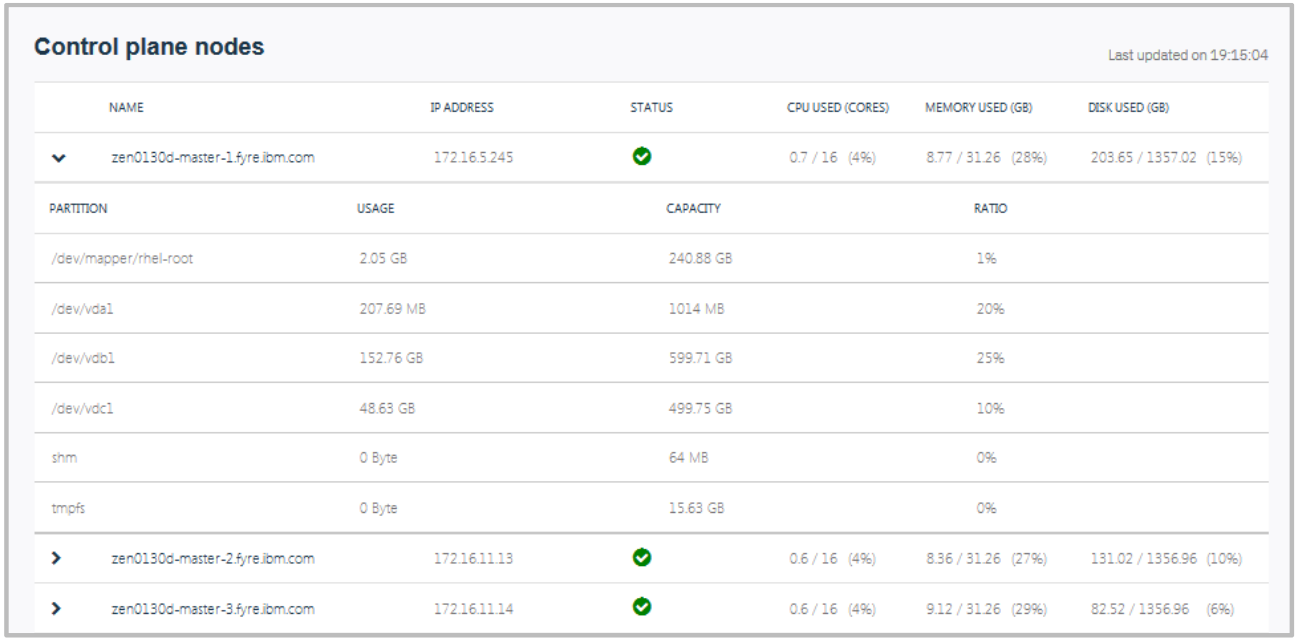
- What if you need more detailed information?
- If you click the node name, you can see the CPU, storage, and memory consumption for the node over the last 20 minutes
  - For example, if we would have clicked on the a specific node from the previous screen this is what we would have seen



© 2019 IBM Corporation

Manage Cluster Nodes

- Lets “zoom out” and go back to our high level view where you can see all the nodes
- If you expand the node name (ie clicking the arrow to the left of the node), you can see the operating system partition and disk partition where the node is running.
  - For example, in the below image you can see the size of the partition and how much of the partition is in use



Control plane nodes

Last updated on 19:15:04

NAME	IP ADDRESS	STATUS	CPU USED (CORES)	MEMORY USED (GB)	DISK USED (GB)
zen0130d-master-1.fyre.ibm.com	172.16.5.245	✓	0.7 / 16 (4%)	8.77 / 31.26 (28%)	203.65 / 1357.02 (15%)

PARTITION	USAGE	CAPACITY	RATIO
/dev/mapper/rhel-root	2.05 GB	240.88 GB	1%
/dev/vda1	207.69 MB	1014 MB	20%
/dev/vdb1	152.76 GB	599.71 GB	25%
/dev/vdc1	48.63 GB	499.75 GB	10%
shm	0 Byte	64 MB	0%
tmpfs	0 Byte	15.63 GB	0%

zen0130d-master-2.fyre.ibm.com	172.16.11.13	✓	0.6 / 16 (4%)	8.36 / 31.26 (27%)	131.02 / 1356.96 (10%)
zen0130d-master-3.fyre.ibm.com	172.16.11.14	✓	0.6 / 16 (4%)	9.12 / 31.26 (29%)	82.52 / 1356.96 (6%)

© 2019 IBM Corporation

Manage Cluster Nodes

- Kubernetes is designed to handle node outages, so you can still use WSL even if some of your nodes are down
- However, if a node is down, you must attempt to diagnose and resolve the problem to avoid outages if other nodes fail
- Contact IBM Software Support if you cannot determine why the node failed or how to restart the node
- Also, down the road if you want to expand the usage / resources for WSL you can add additional compute or deployment nodes
  - **Restriction**: WSL does not support adding additional nodes with GPU

© 2019 IBM Corporation

Monitor Cluster Nodes

- Your WSL deployment consists of three types of nodes:

1. **Control Nodes:**
   - Nodes that manage your Kubernetes cluster and your DSX Local deployment
   - By default, the cluster has three control nodes. If you notice that a node is down, attempt to restore it to prevent outages.
   - The cluster can continue to run if one node fails. However, if two nodes fail, your cluster fails.

© 2019 IBM Corporation

Monitor Cluster Nodes

2. Storage Nodes: (Combine with Control node in latest version)

- Nodes where WSL metadata and any data that you load into WSL is stored
- By default, the cluster has three storage nodes. The data on these nodes is replicated across each node, so that if a node fails, you can still access the data. WSL can continue to run if two nodes fail

– **Tip:** If you run out of space on your storage nodes, add XFS-formatted disks to each node and extend the Logical Volume Management (LVM) partition to include the disks. If possible, ensure that the disks are the same size.

© 2019 IBM Corporation

Monitor Cluster Nodes

3. Compute Nodes:

- Nodes where WSL services, such as Spark, run
- Unlike storage nodes, compute nodes are not replicated. When a new process starts, Kubernetes determines which node has sufficient capacity to run the process.
- WSL can continue to run when multiple compute nodes fail. However, you might notice that performance decreases when multiple nodes are down
- Additionally, if a node fails, Kubernetes attempts to bring any active processes up on another node. While Kubernetes attempts to bring up the processes, you might experience an outage
    - **If Kubernetes cannot bring the processes up on another node and the outage continues, contact IBM Software Support**

© 2019 IBM Corporation

Monitor Cluster Nodes

4. Deployment Nodes: (Optional**)

- Deployment nodes are production versions of the compute nodes
- Asset deployment requires at least one deployment node.
- If the deployment nodes fail all the deployments will go offline and remain offline until nodes are repaired or replaced.

Recommended configuration examples

Cluster type	# node breakdown	Notes
3 nodes	3 shared control/compute	Unable to deploy assets.
4 nodes	3 shared control/compute + 1 deploy	
5 nodes	3 shared control/compute + 2 deploy	
7 nodes	3 control + 3 compute + 1 deploy	
8 nodes	3 control + 3 compute + 2 deploy	
11 nodes	3 control + 6 compute + 2 deploy	

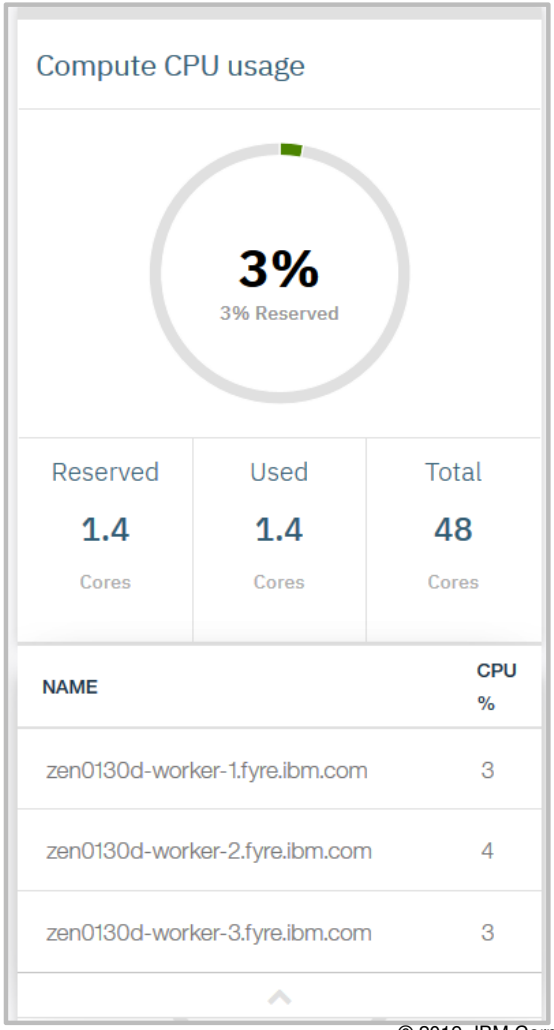
Node Health

- If you want a high-level overview of the status of your cluster, you can monitor the health of your cluster nodes from the Dashboard page
- You can access the Dashboard page from the menu icon ()
- Specifically, you can monitor:
  - CPU Usage
  - Memory Usage
  - Disk Usage

© 2019 IBM Corporation

Node Health

- For compute nodes, the usage of CPU and memory is measured against the CPU and memory that the WSL users reserved
- Each card on the Dashboard page shows the average usage across all of the nodes.
- However, you can expand the cards to see the specific usage for each node.



© 2019 IBM Corporation

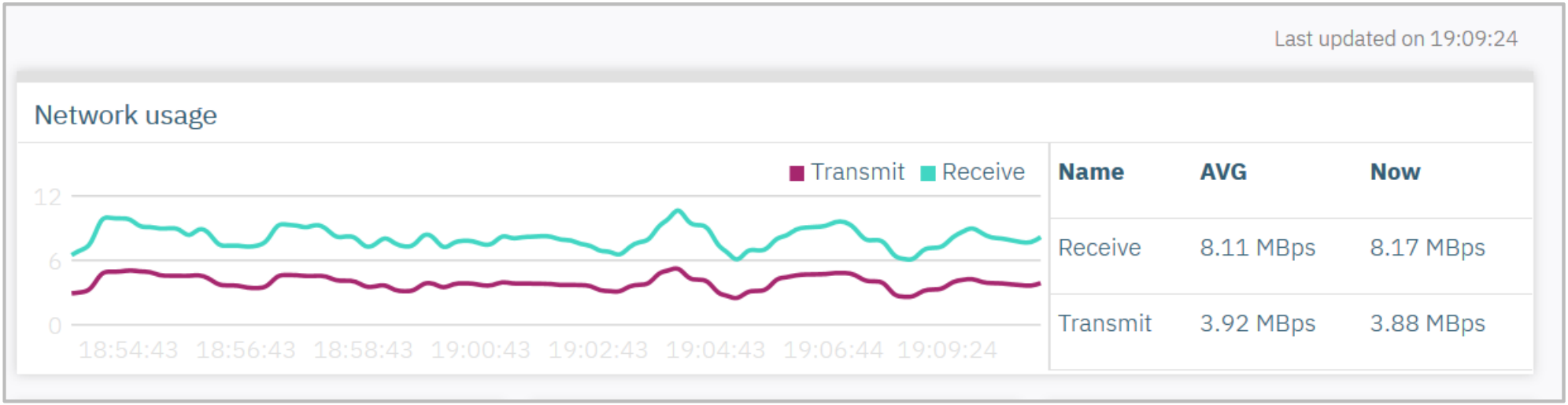
Node Health

- Data is refreshed every 10 seconds
- By default, Kubernetes attempts to balance the load across servers
- Contact IBM Software Support if you notice that all of the nodes in a group are overloaded for extended time
  - Nodes are overloaded when they run above 90% usage
- Nodes can become overloaded when:
  - You have more users than your cluster configuration can handle
    - For example, your cluster doesn't have sufficient CPU, memory, or storage
  - A node fails and other nodes need to handle requests that would normally be handled by that node.

© 2019 IBM Corporation

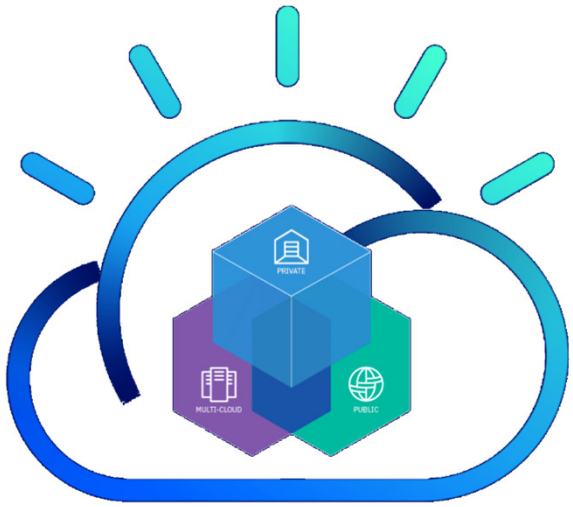
Monitor Network Usage

- If you encounter an issue with WSL, you can view the recent network traffic in your cluster on the Dashboard page
- You can view the number of megabytes that were sent and received across the nodes of the cluster over the last 20 minutes



© 2019 IBM Corporation

Watson Studio Local
Administration
Mange Services



© 2019 IBM Corporation

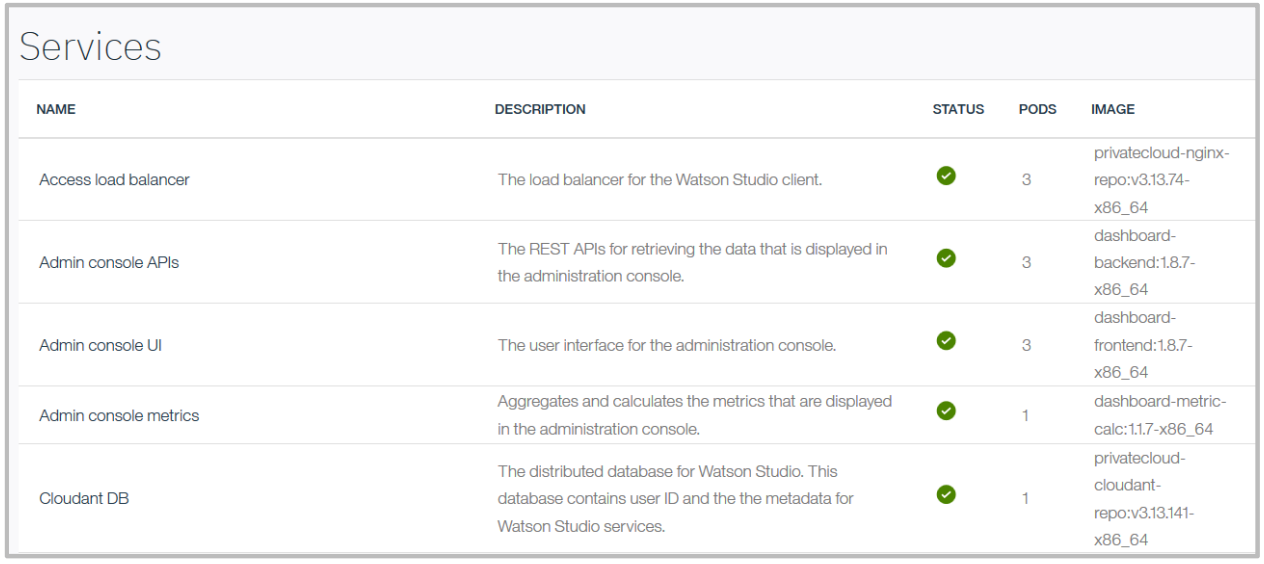
Manage Services

- Watson Studio Local is composed of multiple processes that run in a set of distributed pods.
- Watson Studio Local uses services to enable communication between pods.
- You can view the list of services that are used in your WSL deployment from the **Services page**
- You can access the Services page from the menu icon ()

© 2019 IBM Corporation

Manage Services

- **Click the service name to see:**
  - The list of pods that the service communicates with
  - The status of each pod
  - The nodes where each pod is deployed



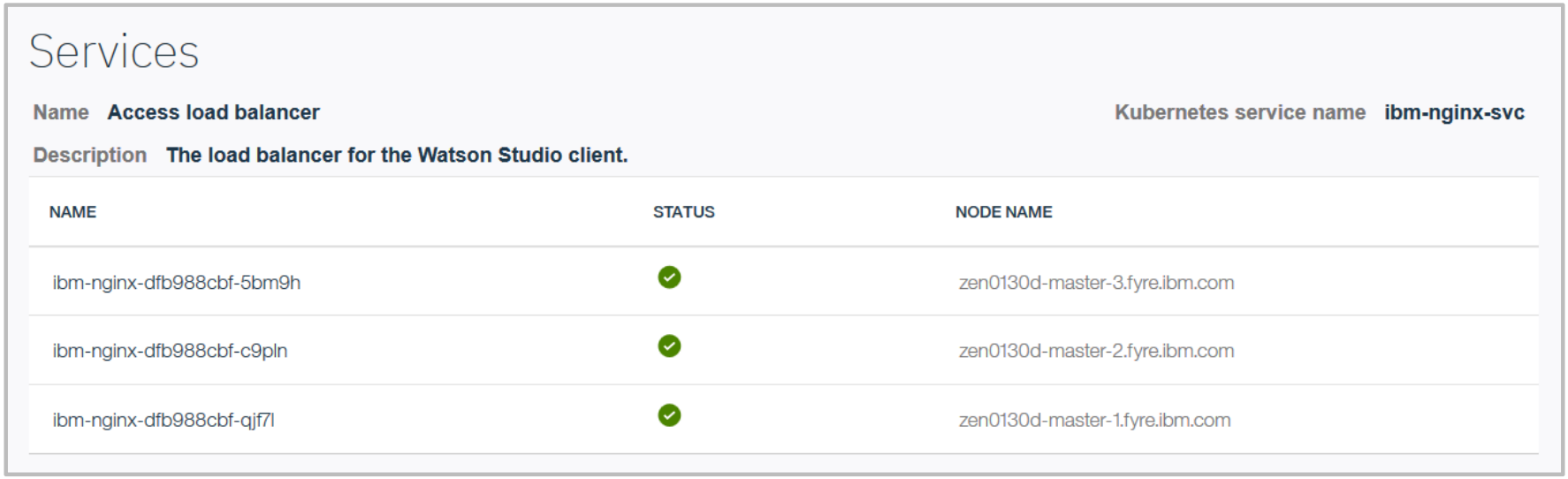
Services

NAME	DESCRIPTION	STATUS	PODS	IMAGE
Access load balancer	The load balancer for the Watson Studio client.	✓	3	privatecloud-nginx-repo:v3.13.74-x86_64
Admin console APIs	The REST APIs for retrieving the data that is displayed in the administration console.	✓	3	dashboard-backend:1.8.7-x86_64
Admin console UI	The user interface for the administration console.	✓	3	dashboard-frontend:1.8.7-x86_64
Admin console metrics	Aggregates and calculates the metrics that are displayed in the administration console.	✓	1	dashboard-metric-calc:1.1.7-x86_64
Cloudant DB	The distributed database for Watson Studio. This database contains user ID and the the metadata for Watson Studio services.	✓	1	privatecloud-cloudant-repo:v3.13.141-x86_64

© 2019 IBM Corporation

Manage Services

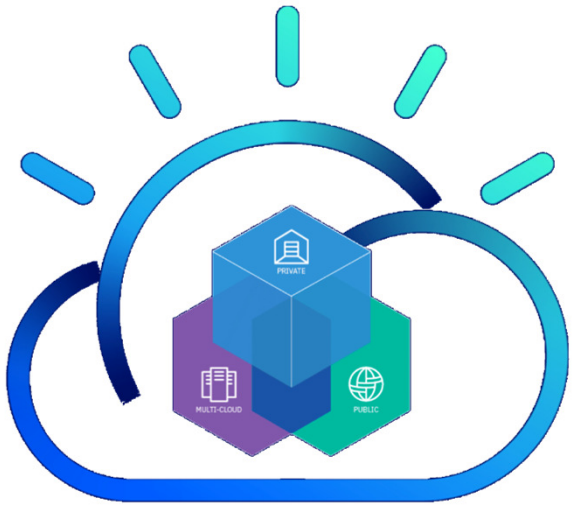
- You can also click each pod name to get more details about each pod



- If you encounter an issue with any of the services in WSL, contact IBM Software Support for assistance with resolving the issue

© 2019 IBM Corporation

Watson Studio Local
Administration
Mange Pods



© 2019 IBM Corporation

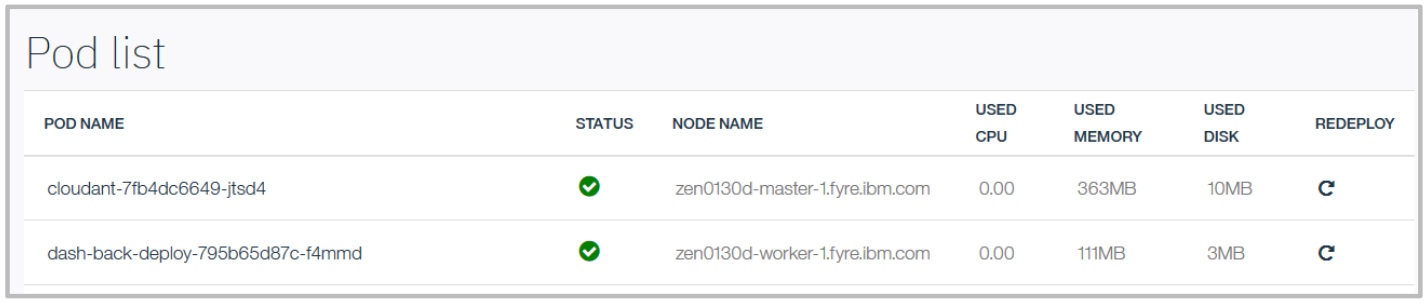
Manage Pods

- In Kubernetes, a *pod* is a process that is running on your Kubernetes cluster
- A pod is a runnable unit of work, which can be a either a stand-alone application or a microservice
- In Watson Studio Local, if a process needs to run on multiple nodes, more pods are created and deployed for extra users or extra notebook servers and IDEs.
  - For example, Spark is deployed as a cluster with at least one pod in each compute node.

© 2019 IBM Corporation

Manage Pods

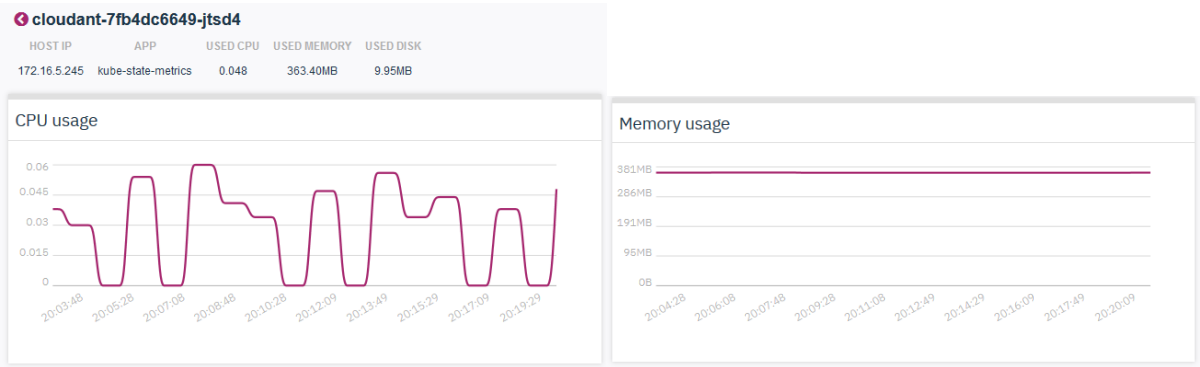
- You can view the list of pods that are used in your Watson Studio Local deployment from the **Pods** page
- You can access the **Pods** page from the Menu icon ( )
- From the **Pods** page you can see:
  - The names of the pods
  - The status of the pods
  - The node where each pod is deployed
  - The CPU, memory, and disk usage for each pod



© 2019 IBM Corporation

Manage Pods

- Additionally, you can click on the name of each pod to see:
  - The IP address of the host where the pod is deployed
  - The name of the service that is running on the pod
  - The amount of CPU the pod is using and a graph of the CPU usage over the last 15 minutes
  - The amount of memory that the pod is using and a graph of the memory usage over the last 15 minutes
  - The amount of disk space that the pod is using and a graph of the disk space usage over the last 15 minutes



© 2019 IBM Corporation

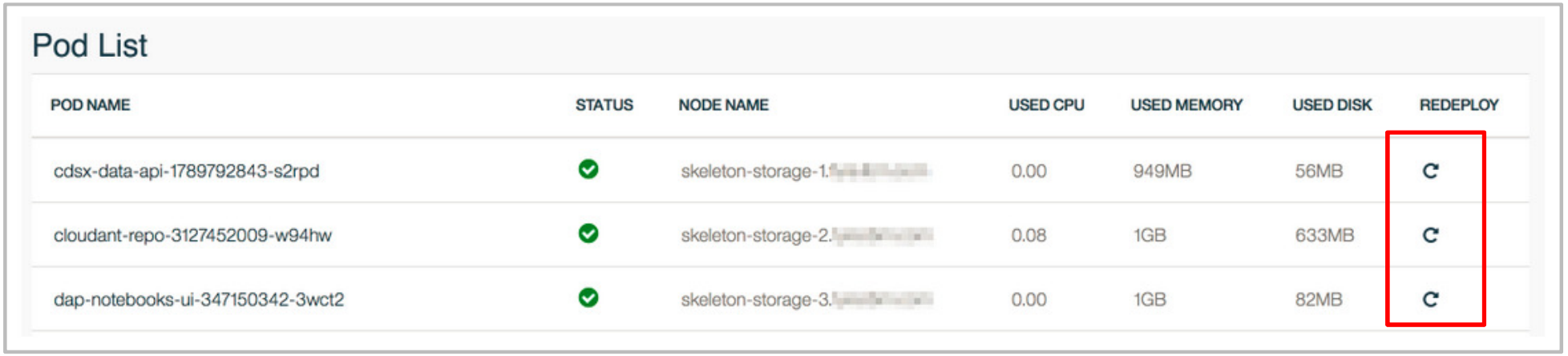
Pod Status

- Typically, all of the pods in your environment are running
- However, you might notice that the pods in your environment are in one of the following states:
  - **Pending ( ):** The Kubernetes cluster is creating the Container images that are included in the pod
  - **Succeeded ( ):** All of the Containers in the pod stopped successfully. The Containers are not restarted
  - **Failed ( ):** All of the Containers in the pod stopped successfully, but at least one Container stopped with an error or was stopped by Kubernetes
  - **Unknown ( ):** The state of the pod cannot be determined, which typically occurs when there is a communication issue with the node where the pod is deployed
  - **Running ( ):** The pod is deployed on a node. At least one Container is running or is in the process of starting or restarting.

© 2019 IBM Corporation

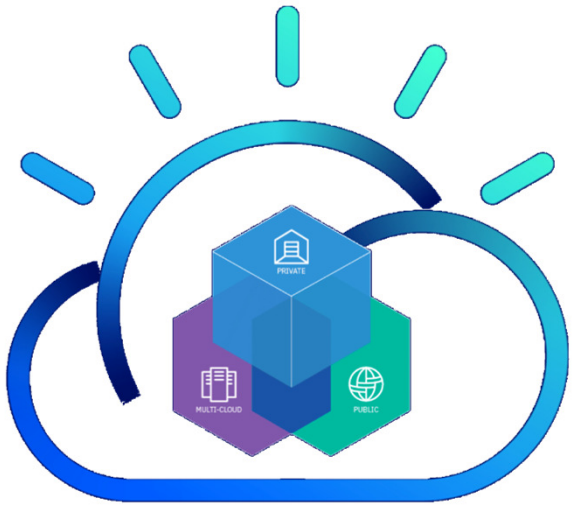
Pod Status

- If you encounter an issue with a service, you might need to redeploy the pod
- You can redeploy the pod by clicking the redeploy button
- If you cannot successfully redeploy a pod from the Pods page, contact IBM Software Support for assistance with resolving the issue



© 2019 IBM Corporation

Watson Studio Local
Administration
View Alerts



© 2019 IBM Corporation

View Alerts

- IBM Watson Studio Local automatically notifies you when a node or pod goes down or when you ‘re at risk of overloading a resource.
- By default, WSL issues alerts when:
  - CPU usage on a node goes above 90%
  - Memory usage on a node goes above 90%
  - Disk usage on a node goes above 90%
  - A node in the cluster goes down
  - A pod fails
  - A pod is not running or is in an unknown state for more than 5 minutes
- If you want to change the threshold at which alerts are issued, you can configure them on the Settings page.

© 2019 IBM Corporation

View Alerts

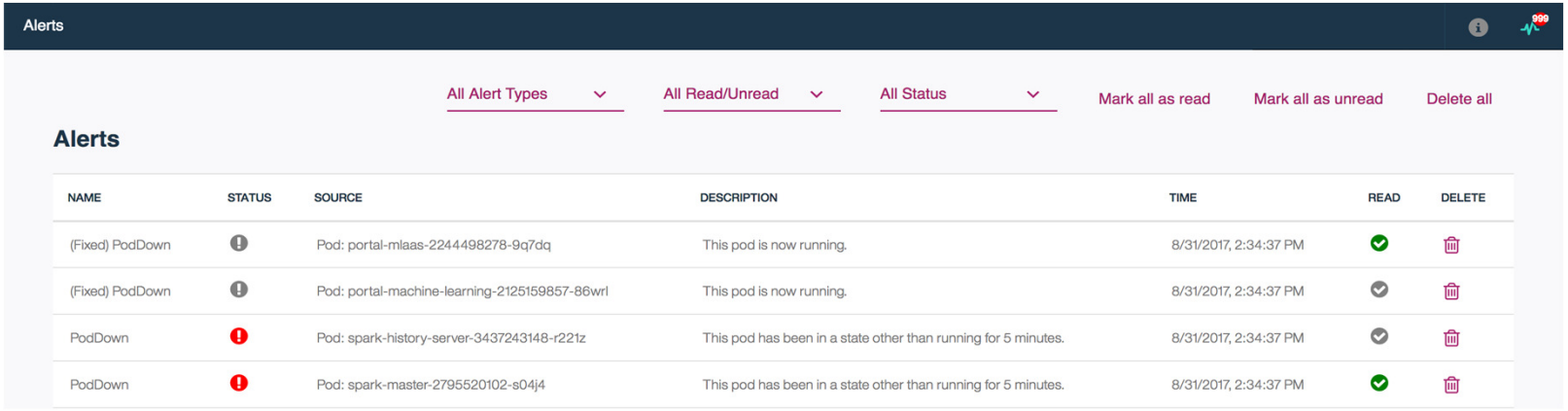
- When you have alerts, the alert icon displays the number of unread alerts in your queue:

- You can access alerts in either of the following ways:
  - If you want a quick peek at your alerts, click the alert icon in the menu bar ( )
  - If you want to manage your alerts, you can access the Alerts page from the menu icon ( )
- **Tip**: When you delete an alert, you can't access it again. Make sure that you don't need it before you delete it

© 2019 IBM Corporation

View Alerts

- From the Alerts page you can:
  - Filter alerts by type
  - Filter alerts based on whether they were read or not
  - Filter alerts by status
  - Mark alerts as read or unread
  - Delete alerts



© 2019 IBM Corporation

Change alert threshold

- In the Watson Studio Local client, an administrator can optionally adjust when alerts are generated, how long log files and metrics are stored, and how frequently the metrics on the dashboard are refreshed.
- To configure refresh and retention settings:
  1. From the Admin user profile icon, select Settings
  2. In the Refresh and alert settings, adjust the appropriate settings

     Log retention (days) – the number of days to keep logs before they are automatically deleted. **Default is 10 days.**

     Metrics retention (days) – the number of days to keep metrics history. **Default is 1 day.**

     Dashboard refresh (seconds) – the frequency with which the data in the dashboard is refreshed. **Default is 10 seconds.**

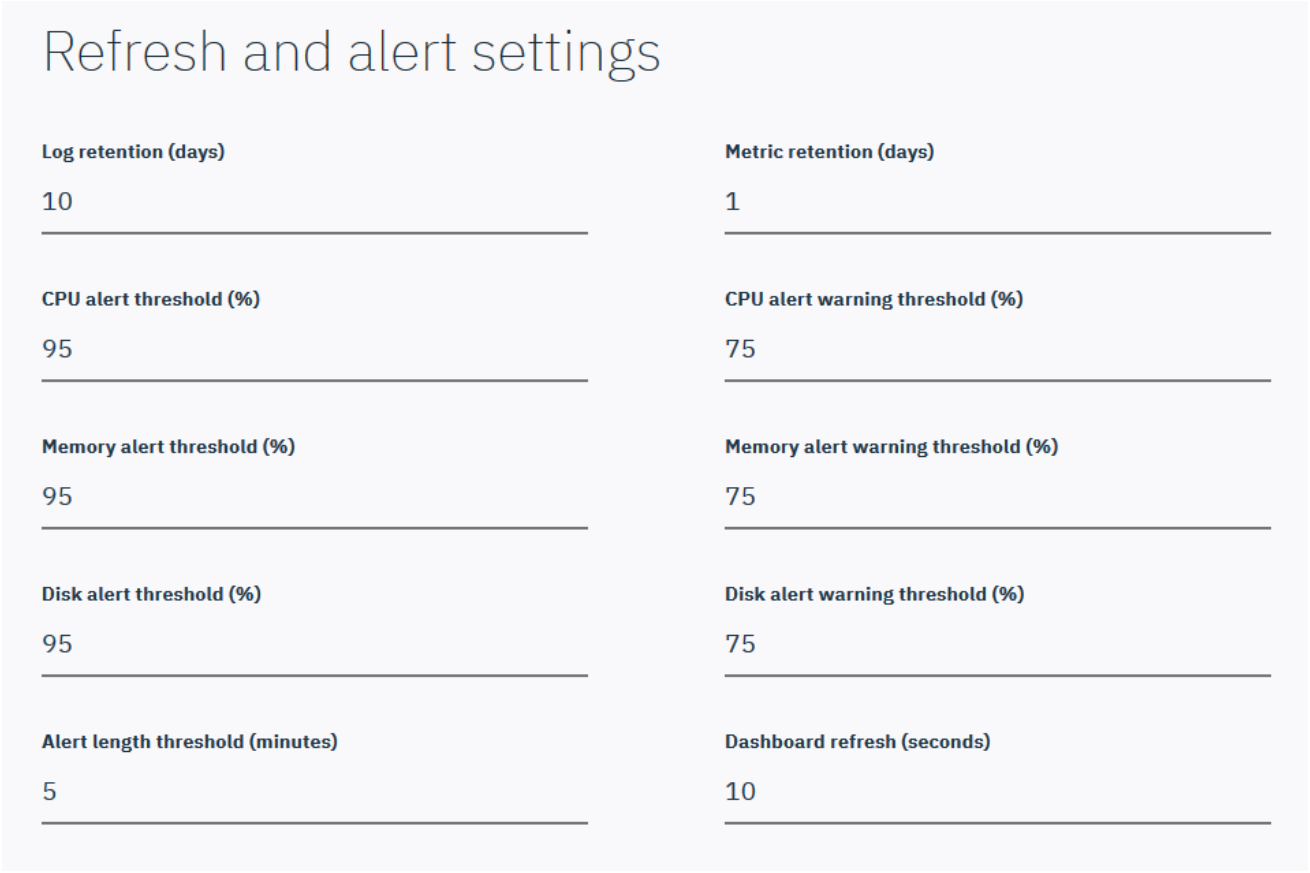
     Alert threshold (%) – the usage threshold at which an alert is triggered. **Default is 90%.**

     Alert warning threshold (%) – the usage threshold at which a warning is triggered and the node color changes to yellow. **Default is 70%.**

  3. Click Save

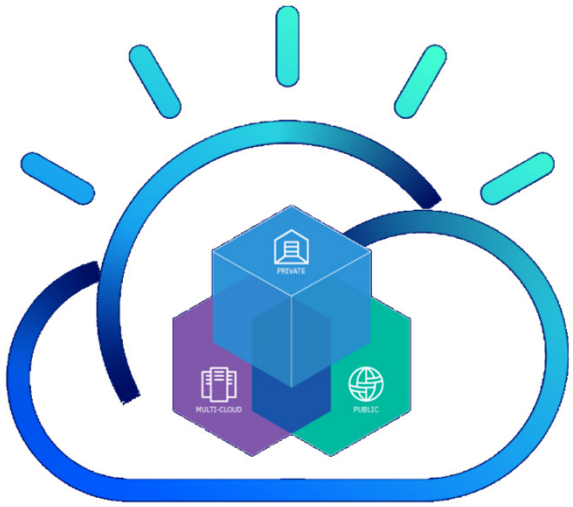
© 2019 IBM Corporation

Change alert threshold



© 2019 IBM Corporation

Watson Studio Local
Administration
Security



© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Security

Network security

- WSL is exposed via 443 port (SSL is supported by default)
- All REST APIs require bearer token or model deployment Token
- WSL supports firewall (vyatta) around cluster, not around individual nodes
- SSO is not yet supported

Data security

- WSL supports full disk encryption via LUKS (must be done prior to install)
- WSL encrypts the credentials for external data sources (AES 256)
- WSL hashes all the user passwords stored in Cloudant

Authentication

- LDAP/AD(See Compatibility Reports for supported LDAP servers)

Authorization

- Pre-defined roles: Administrator, Deployment Administrator, User
- Project roles: Administrator, Editor, Viewer

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Security

Watson Studio Local comes equipped with an array of security features you can use. Including:

- SAML
  - An administrator can setup SAML2.0 to redirect WSL users to sign in securely through your identity provider's login page
- SSH
  - You can install WSL using a private SSH key file for the credentials
  - A WSL administrator can enable SSHD so that users (using a public SSH key pair) can ssh directly to the WSL cluster through a secure port.
- SSL
- Tokens
- Authentication
- Authorization
- Data

© 2019 IBM Corporation

Audit Log

- WSL logs an audit record of user login attempts (either from the sign in page or using the validateAuth REST API) to the WSL cluster
- The record is stored as JSON, and contains the following information about each login attempt:

Template:

```
{
   "DSX_AUDIT_RECORD": {
      "version":"1.0",
      "event":"user_auth",
      "timestamp": Request timestamp in ISO8601 DateTime with
Timeszone format
      "user_auth_object": {
          "user_auth_object_version":"1.0",
          "uri":Request URI
          "username":The user name used to attempt the login,
          "status":Result Status code (e.g. 200: OK, 401:
Authorization failure)
      }
   }
}
```

Example:

```
{
   "DSX_AUDIT_RECORD": {
      "version":"1.0",
      "event":"user_auth",
      "timestamp":"2018-03-08T06:38:00+00:00",
      "user_auth_object": {
          "user_auth_object_version":"1.0",
          "uri":"\/v1\/preauth\/validateAuth",
          "username":"admin",
          "status":200
      }
   }
}
```


© 2019 IBM Corporation

Audit Records

- All audit records are retained for a default of 10 days before they are automatically deleted
- The WS administrator can modify the log retention settings by clicking Settings from the admin user profile icon and under Refresh and alert settings, typing in a new duration in the Log retention (days) field
- **Tip**: The admin could download the user login audit records using the REST API periodically, before they are automatically deleted for long term storage

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Security

Get User Bearer Tokens

- WSL requires a bearer token to authenticate the user calling the REST APIs.
- The token lasts for 13 hours. The following example shows how to retrieve and set a bearer token from the user management service:

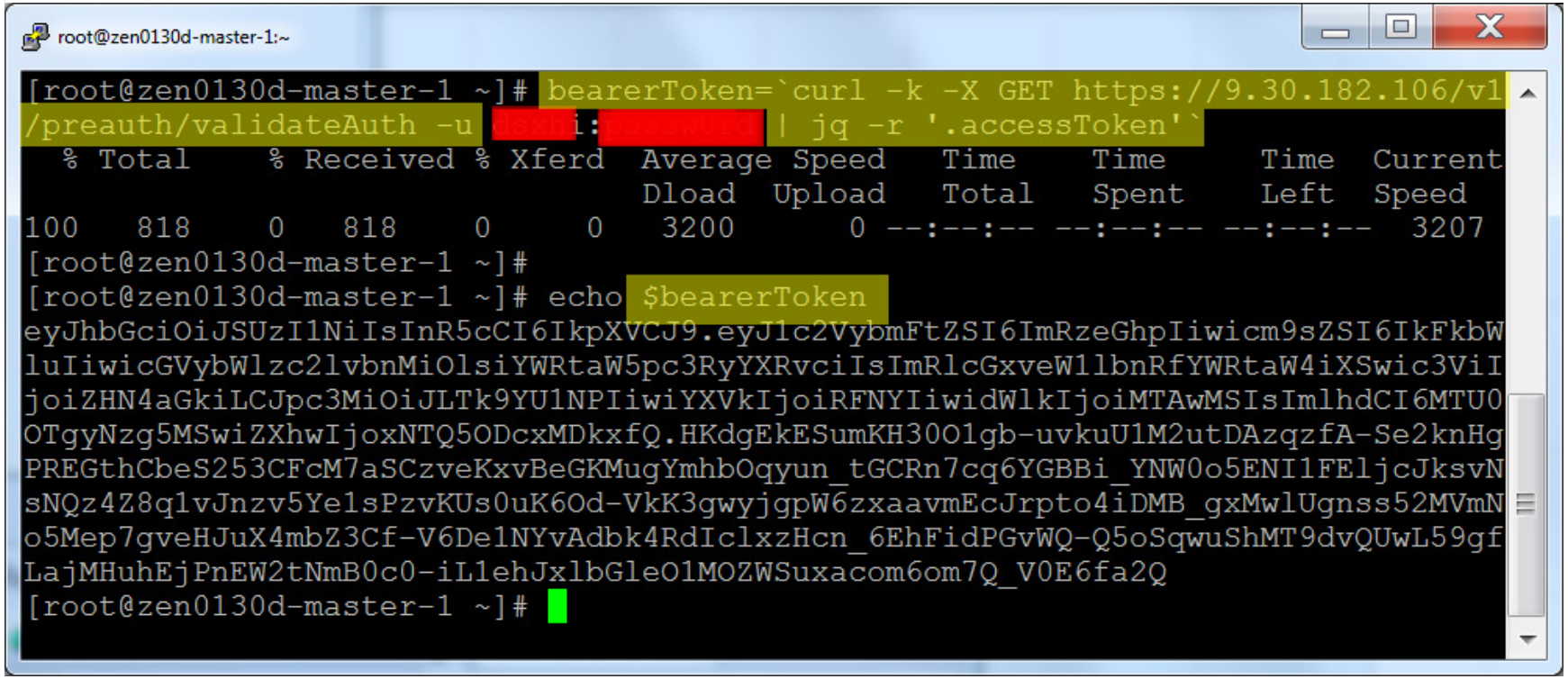
  Example

  # bearerToken=`curl -k -X GET <WSL_URL>/v1/preauth/validateAuth -u <uid>:<password> | jq -r '.accessToken'`

© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Security

Get User Bearer Tokens



© 2019 IBM Corporation

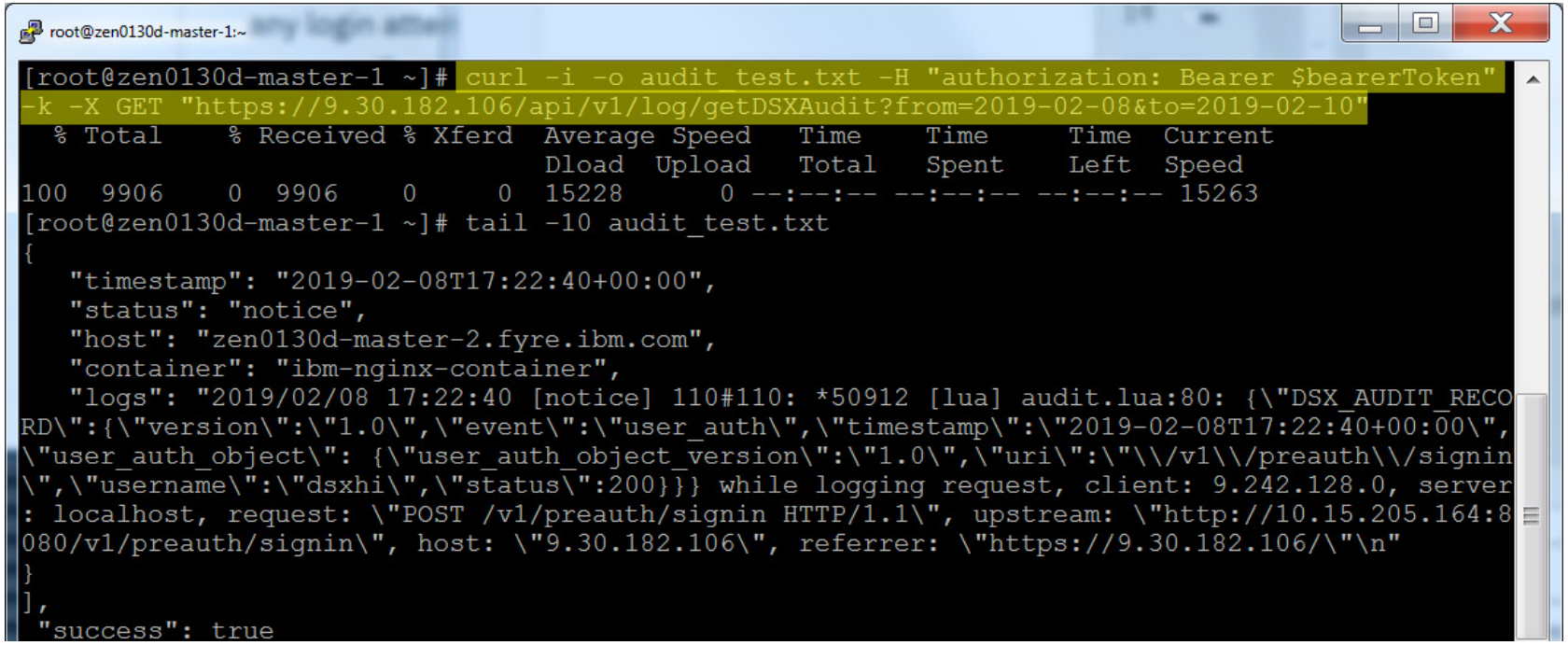
IBM Watson Studio Local (WSL) – Security

Get Server Audit Logs

Example

curl -i -o audit_test.txt -H "authorization: Bearer $bearerToken" -k -X GET

“<WSL_URL>/api/v1/log/getDSXAudit?from=2019-02-08&to=2019-02-10"



© 2019 IBM Corporation

IBM Watson Studio Local (WSL) – Security

Usage Report

- Every day at 1AM (machine time) WSL creates and stores a report of users signing into the system.
- To retrieve these usage report files, run the script “/wdp/utils/get_usage_reports.sh” on any one of the master nodes.
- The script saves zipped usage reports in the ws_usage_reports directory.
- Each usage report is a comma delimited CSV file with the column: Username, Login timestamp.
- Example report:

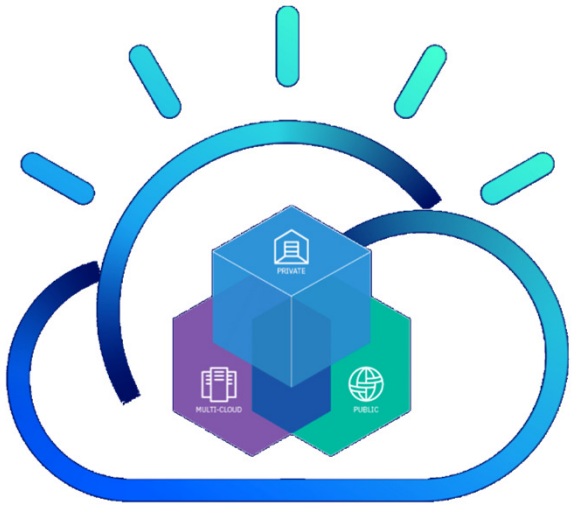
```
test_user_1_36, 2018-12-12T18:04:23+00:00
test_user_1_29, 2018-12-12T18:04:23+00:00
test_user_1_62, 2018-12-12T18:04:22+00:00
test_user_1_2, 2018-12-12T18:04:22+00:00
test_user_1_83, 2018-12-12T18:04:21+00:00
test_user_1_3, 2018-12-12T18:04:21+00:00
test_user_1_46, 2018-12-12T18:04:20+00:00
```


© 2019 IBM Corporation

Watson Studio Local

Administration

SSHD



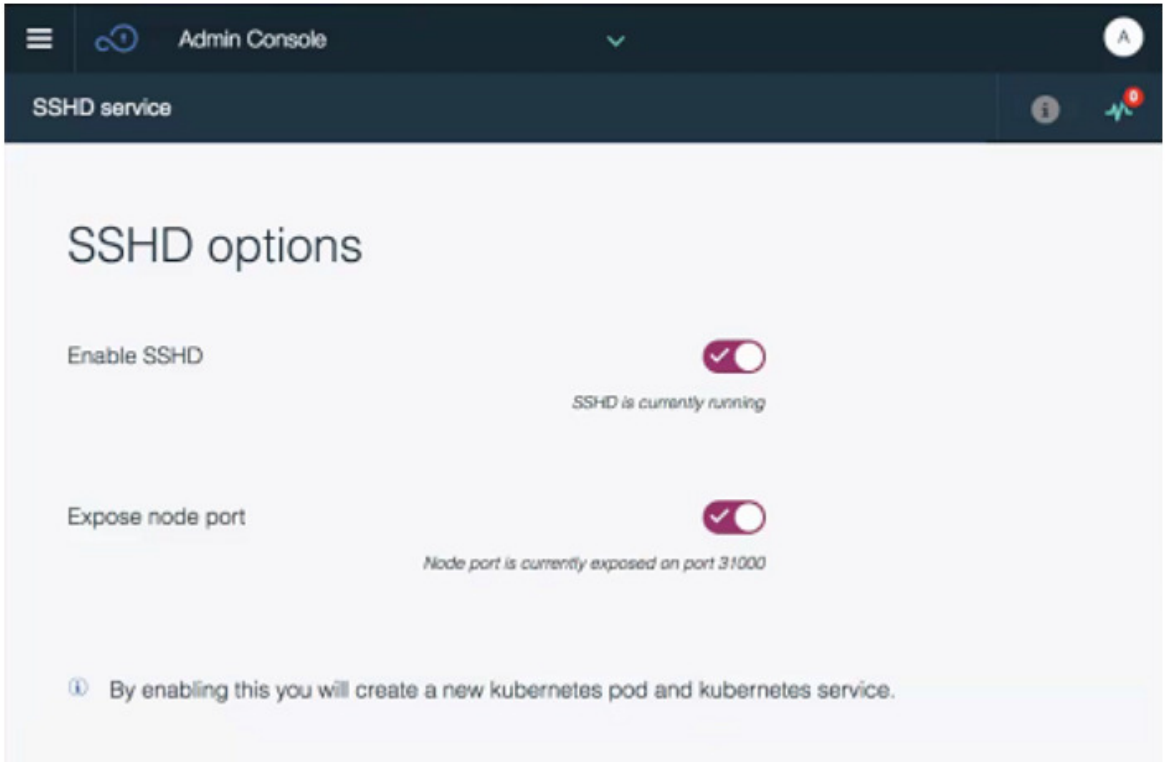
© 2019 IBM Corporation

SSHD Service

- SSH connection to Watson Studio Local is disabled by default. To enable the connection, do the following steps:
  - In the Admin Console, click the menu icon () and click SSHD service. The SSHD options window opens.
  - For Enable SSHD, switch SSHD on. A new kubernetes pod and service starts.
  - For Expose node port switch the node port on to expose a secure kubernetes port.

© 2019 IBM Corporation

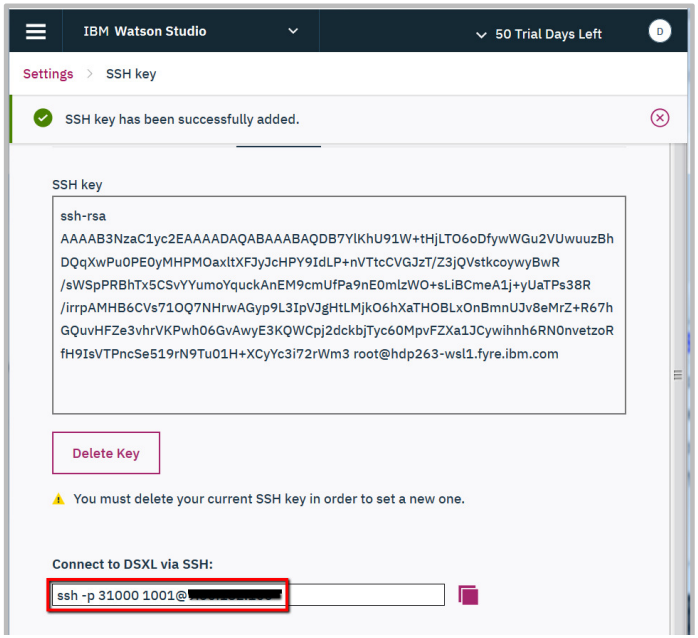
Enable SSHD Service



© 2019 IBM Corporation

Configure the SSH key in Watson Studio Local

- To setup an SSH connection to WSL, a WSL user must complete the following steps:
  - In the WSL client, go to **Settings** and click the **SSH** key tab.
  - In the SSH key field, paste in the public SSH key. Click the Add Key button. The public SSH key is automatically saved in the user's home directory on the WSL cluster.

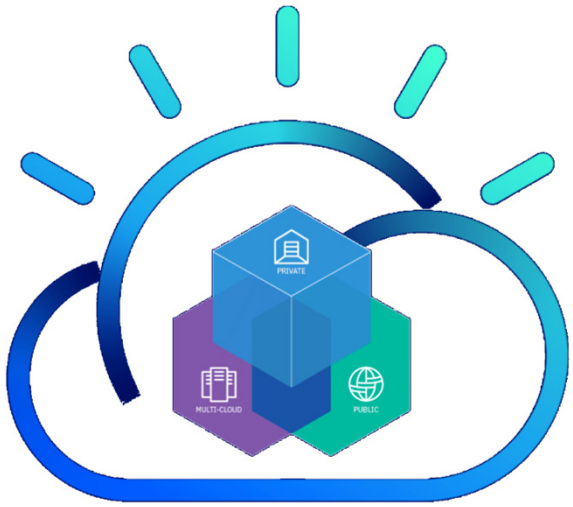


  - In the Connection via SSH field, copy the ssh command and paste it into the SSH client.
  - WSL user can then ssh to the Watson Studio Local cluster through the secure port.

Tip: To upload large files, place them in /user-home/_global_/libraryProjects/<YourLibrary>/datasets.

© 2019 IBM Corporation

Watson Studio Local
Administration
Replace Nodes



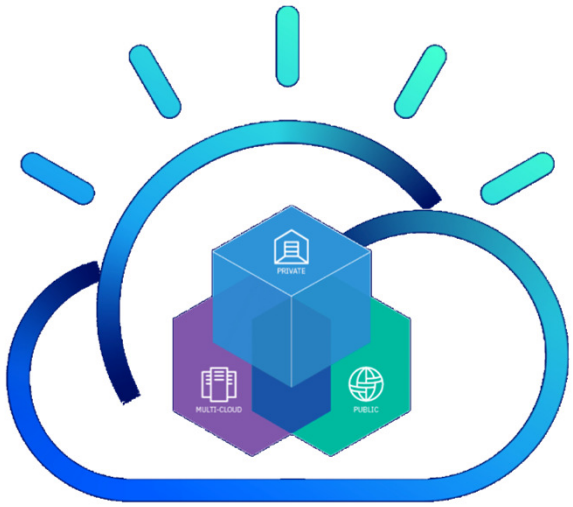
© 2019 IBM Corporation

Replace Nodes – Master Node

- If a master or compute node fails, a WSL administrator can manually replace it at the command line
- The process to do so is quite involved for any of the nodes, so it will not be covered here; the URL where the process is documented is found below:
  - https://content-dsxlocal.mybluemix.net/docs/content/SSAS34_current/local/replacenodes.html

© 2019 IBM Corporation

Watson Studio Local
Administration
Manage
Library & Package



© 2019 IBM Corporation

Install Custom or 3rd Party Library – Notebooks

- You can add processing capabilities to Apache Spark in WSL by installing external or third-party libraries
- WSL includes many pre-installed libraries
- Before you install a library, check the list of preinstalled libraries. Run the appropriate command from a notebook cell:

```
• Python: !pip list --isolated
• R: installed.packages()
```

- If the library you want to use is not listed, then you can use the commands on the following slides to install it

© 2019 IBM Corporation

Install Local Python Library – Notebooks

- Use the Python pip package installer command to install Python libraries to your notebook
  - For example, using the command "*!pip install prettyplotlib*" installs the prettyplotlib library into a pod's conda and removes the package once the pod is terminated
  - Or using the command "*!pip install –user prettyplotlib*" installs the prettyplotlib library in your user home and the package exists even after the pod is terminated
- Use the Python import command to import the library components
  - Going off our example above, you would then use the command "*import prettyplotlib as ppl*"
- Restart the kernel

© 2019 IBM Corporation

Install Global Library & Package

- A WSL administrator can install Python or R packages in global directories. These packages are then made available to all users
- To install a global Python library:
  - Log in to WSLL as admin and create a Python notebook
  - Use the Python pip package installer command to install Python libraries to your notebook for example see the image below:

```
!pip install --target
/user-home/_global_/python-2.7 prettyplotlib
```

- The installed packages can be used by all notebook users that use the same Python version in the Spark service
- Notebook users can now use the Python import command to import the library components.
  - For example, users can run the following command in a code cell:

```
import prettyplotlib as ppl
```


© 2019 IBM Corporation

Install Global Library & Package

- To install a global R package:
  - Log in to WSL as admin and create an R notebook
  - Use the R install.packages() function to install new R packages. For example, running the following command would install the ggplot2 package

```
install.packages("ggplot2")
```

- The imported package can be used by all R notebooks that are running in the Spark service
- Now, users can use the R library() function to load the installed package
  - For example, a user can run the following command in a code cell:

```
library("ggplot2")
```


© 2019 IBM Corporation

Manage Packages

- **Administrator**

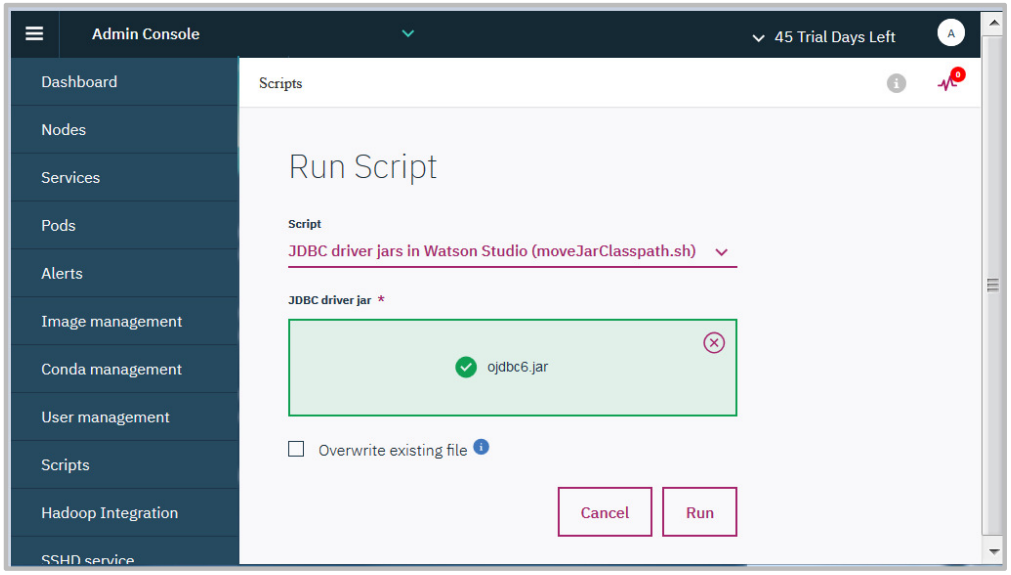
  A WSL administrator can install Python or R packages in global directories. These packages are available to all users on the cluster

  - **Add a global Spark JAR file**
    - In the Scripts panel of the Admin console, an administrator can select “Add Spark jars in Watson Studio (moveJarClasspath.sh) to upload JAR files to the /user-home/_global_/spark/jars/ directory for use with Spark.

© 2019 IBM Corporation

Manage Packages

- **Administrator**
  - **Add a JDBC driver**
    - In the Admin Console, click the menu icon ( ) and click Scripts.
    - In the Scripts pull-down menu, select JDBC driver jars in Watson Studio (moveJarClasspath.sh), and click add jar.



© 2019 IBM Corporation

Manage Packages

- **Administrator**
  - **Install a global Python package**
    - Log in to WSL as the default administrator (ie. Admin) and create a Python nodebook.
    - Use the Python pip package installer command to install Python libraries in the Python notebook.

      Example:
      !pip install –target /user-home/_global_/python-2.7 prettyplotlib
    - Once the package is installed, it can be used by all notebook users. Example from above installation, users can then use the library simply do: Ex: import prettyplotlib as ppl

© 2019 IBM Corporation

Manage Packages

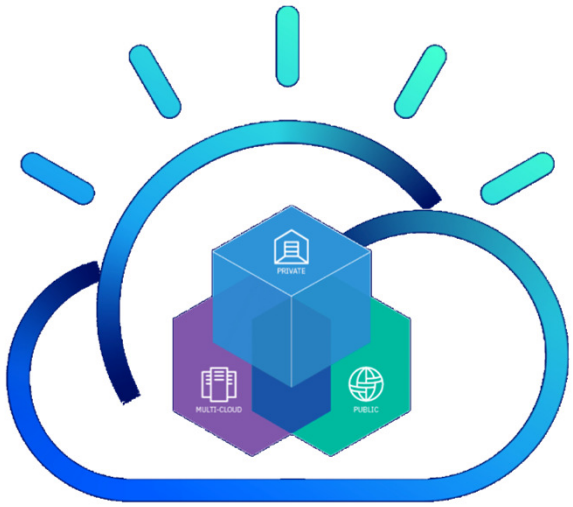
- **WSL Local User**
  - Local users can add additional packges to a base image, which can then be reused by other users.
  - WSL includes many preinstalled libraries. Check the list before you install a library. (or type command: “pip list” for Python, “installed.packages() for R.)
  - **Install a Python library**
    - Use the Python conda install. Ex: !conda install prettyplotlib
  - Use custom functions/libraries in Python scripts
    - **Example:**
      sys.path.insert(0, ‘../scripts/’)
      from your_custom_script import *
  - **Install a R library**
    - Example: install.packages(“ggplot2”)

© 2019 IBM Corporation

Watson Studio Local

Administration

Hadoop Integration



© 2019 IBM Corporation

Objective Overview

- Hadoop infrastructure is a huge investment but the tools in the ecosystem are not intuitive for business analyst and users. Most of your data is probably available within your HDFS along with your huge compute capacity but access to it is difficult.
- Build machine learning and analytics on Watson Studio that provides variety of open source libraries and frameworks such as Spark, R, Keras, TensorFlow, Scikit-learn as a service within a kubernetes based platform behind your firewall. Along with cloud agility the platform provides end-end data management and analytics capability with collaboration and lifecycle management functions.
- Hadoop connector provides seamless access to compute power available along with easy access to hdfs data.
- Accelerate your machine learning deployments in hours as oppose to months. Build quick to market business decision making Artificial Intelligence systems easily and quickly.

© 2019 IBM Corporation

Hadoop Integration – Overview

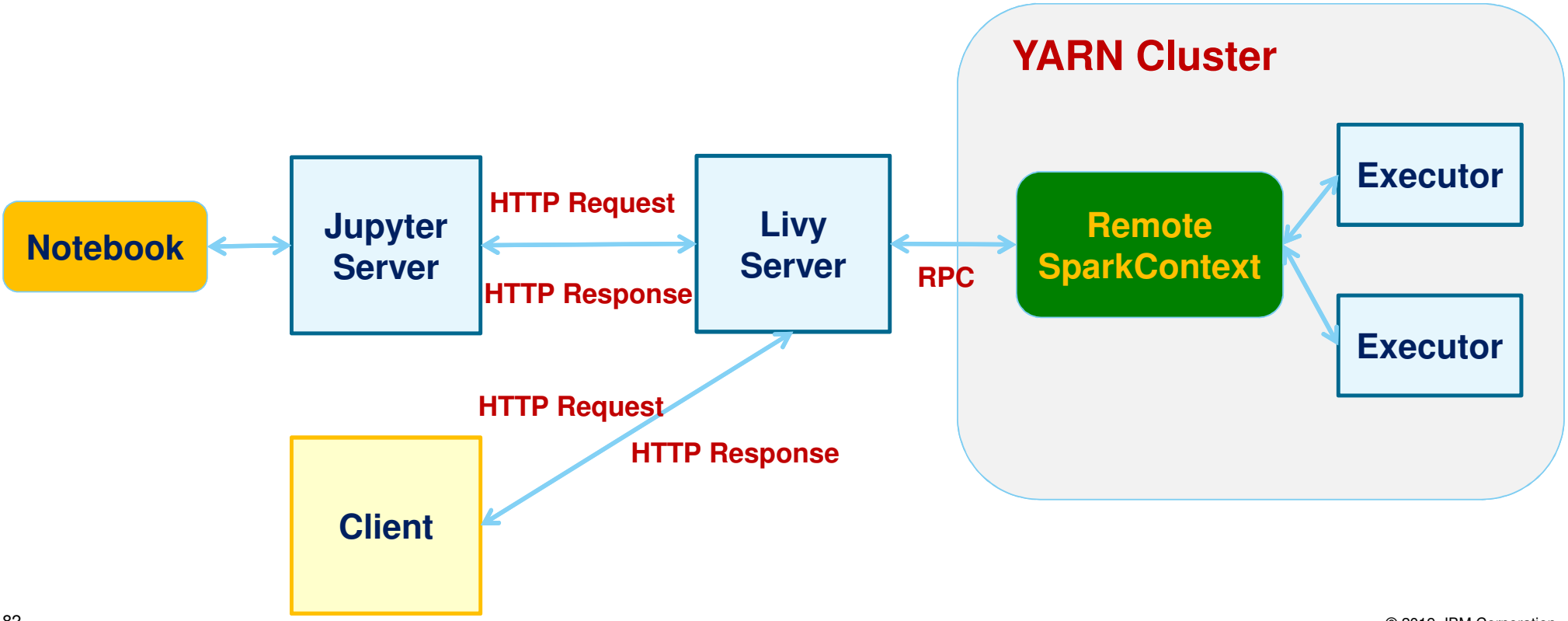
- The Watson Studio Local Hadoop Integration Service is a registration service that can be installed on a Hadoop edge node.
- WSL Version 1.2 or later clusters can securely access data residing on the Hadoop cluster, submit interactive Spark jobs, build models, and schedule jobs that run as a YARN application.

Important: Your Hadoop admin must install the Hadoop registration service on a Hadoop cluster, add the Watson Studio Local cluster to the Hadoop registration to be managed, and provide the Hadoop registration service user ID and the URLs.

© 2019 IBM Corporation

Livy Interface with Spark

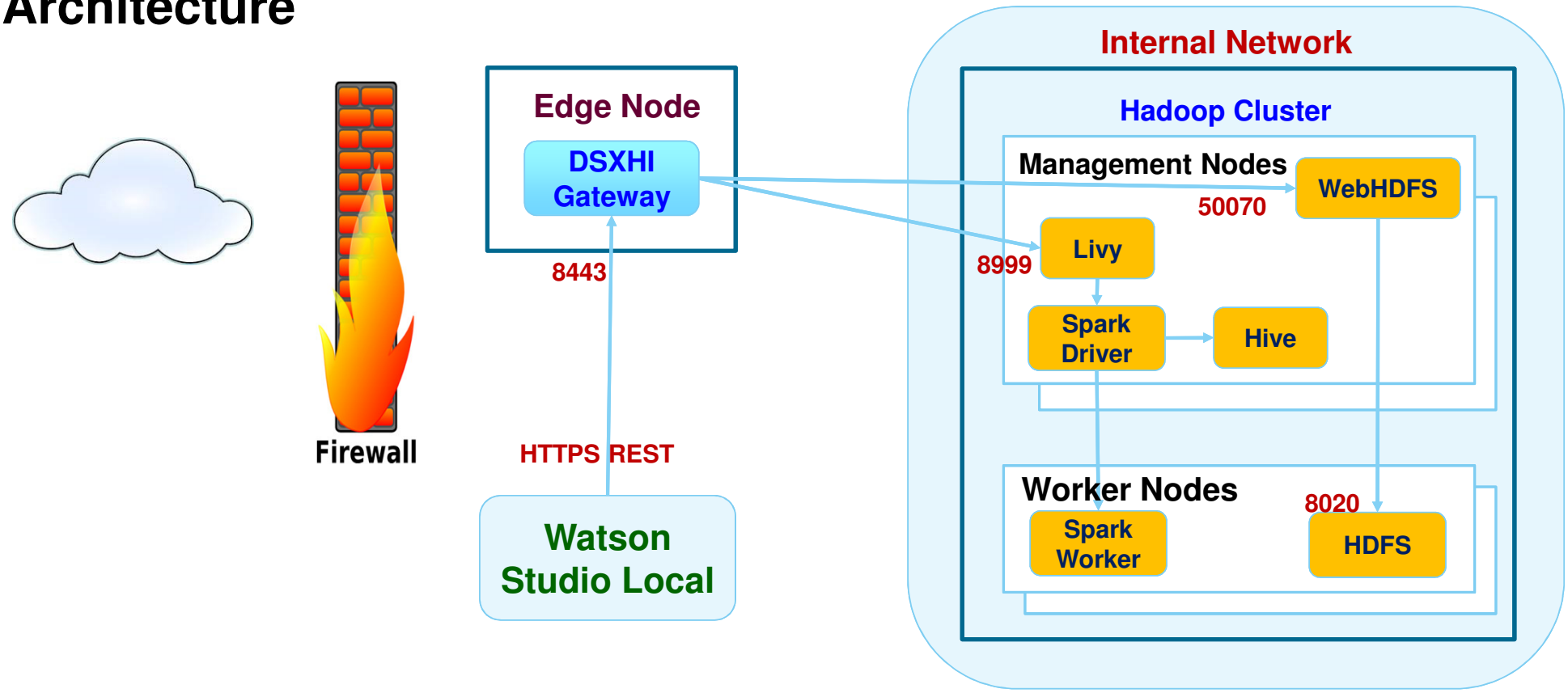
Architecture I



© 2019 IBM Corporation

Hadoop Connector Installation & Setup

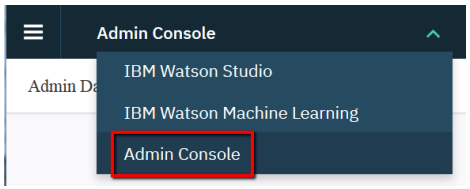
Architecture

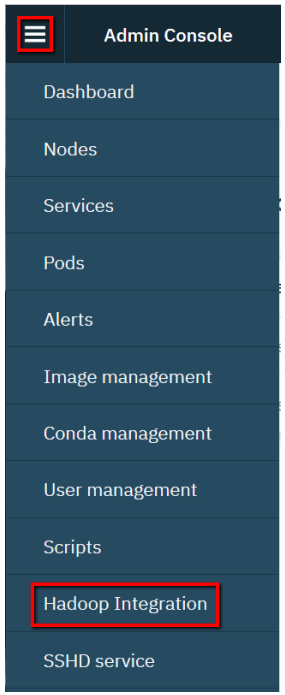


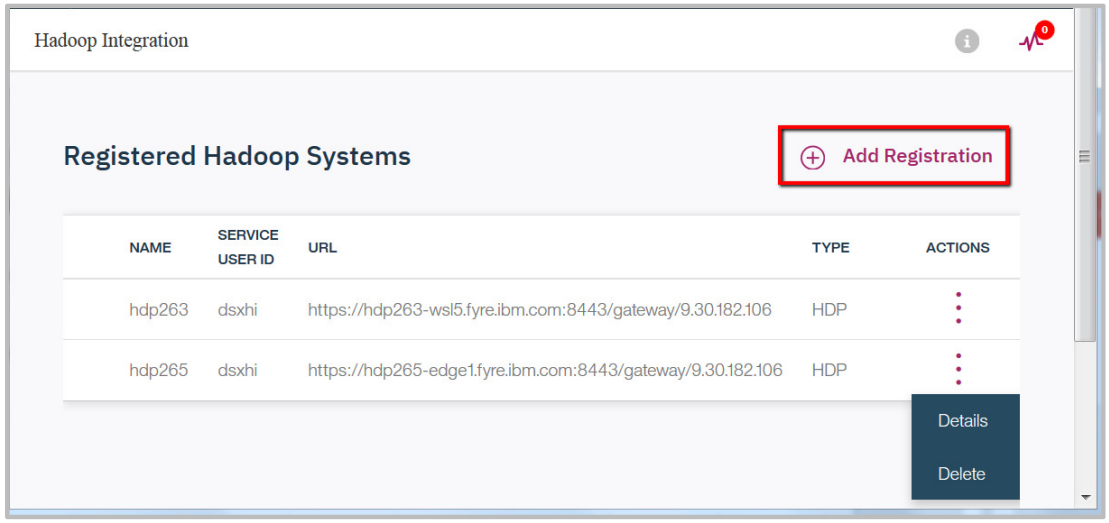
© 2019 IBM Corporation

Hadoop Integration – Register a Hadoop Cluster

- In the Admin Console, click the menu icon (☰) and click **Hadoop Integration**.
- HDFS and Hive data sources are automatically created when a Hadoop registration service is registered in WSL admin console.







Tip: If the registration fails, view the kubectl logs of the utils-api pod.

© 2019 IBM Corporation

Hadoop Integration – View details about the registered Hadoop cluster

If Livy for Spark and/or Livy for Spark2 services are exposed, WSL users can list these endpoints through dsx_core_utils and dsxCoreUtilsR, and use them as defined Livy endpoint in Jupyter, RStudio and Zeppelin notebooks.

Python syntax:

```
%python
import dsx_core_utils;
dsx_core_utils.list_dsxhi_livy_endpoints();
```

R syntax:

```
library('dsxCoreUtilsR');
dsxCoreUtilsR::listDSXHILivyEndpoints()
```


© 2019 IBM Corporation

Hadoop Integration

FYI. When a Watson Studio Local administrator pushes a custom image to a registered Hadoop system, a notebook can initialize a remote Spark Livy session which runs within an environment associated with that remote custom image:

```
import dsx_core_utils
%load_ext sparkmagic.magics

# Retrieve a list of registered Hadoop Integration systems.
DSXHI_SYSTEMS = dsx_core_utils.get_dsxhi_info(showSummary=True)
```

```
Available Hadoop systems:

  systemName  LIVYSPARK  LIVYSPARK2 imageId
0     hdp263  livyspark  livyspark2
1     hdp265  livyspark  livyspark2
```

```
session_name = 'spark2_0205'
livy_endpoint = HI_CONFIG['LIVY']
webhdfs_endpoint = HI_CONFIG['WEBHDFS']
%spark add -s $session_name -l python -u $livy_endpoint
```

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
12	application_1549042673081_0016	pyspark	idle	Link	Link	✓

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

- **Pre-req**
  - Install Hadoop clients on an edge node (**Recommended) or a node that will hos gateway service
  - Create a gateway service id on all Hadoop cluster nodes & edge node
    - ✓ Ex. # useradd -u 3001 -g hdfs dsxhi (all Hadoop cluster nodes**)
  - Create HDFS directories
    - ✓ Ex. # su - hdfs -c 'hdfs dfs -mkdir -p /user/dsxhi/environments'
    - ✓ # su - hdfs -c 'hdfs dfs -chown -R dsxhi:hdfs /user/dsxhi'
  - Collect port information for HDFS, WebHDFS, Livy** (ex. 8020, 50070, 8999)

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

- **Pre-req**
  - Add Hadoop service parameters for gateway service id (listed in below)
    - ✓ 1. HDFS custom core-site

      hadoop.proxyuser.<service_id>.groups=*

      hadoop.proxyuser.<service_id>.hosts=*
    - ✓ 2. Hive custom webhcat-site

      webhcat.proxyuser. <service_id>.hosts=*

      webhcat.proxyuser.<service_id>.groups=*
    - ✓ 3. Spark2 custom livy2-conf

      livy.superusers=<service_id>

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

- **Install Livy connector (dsxhi) rpm package**
  - Download connector rpm file from Passport Advantage
  - Install rpm file (Ex. rpm -ivh dsxhi-icp4data-dsp-1.1.1.0-64.noarch.rpm)
- **Configure Livy connector**
  - Modify configuration files
    - ✓ # cp /opt/ibm/dsxhi/conf/dsxhi_install.conf.template.HDP /opt/ibm/dsxhi/conf/dsxhi_install.conf
    - ✓ # vi /opt/ibm/dsxhi/conf/dsxhi_install.conf

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

✓ Example:

root@hdp263-wsl5:/opt/ibm/dsxhi/conf

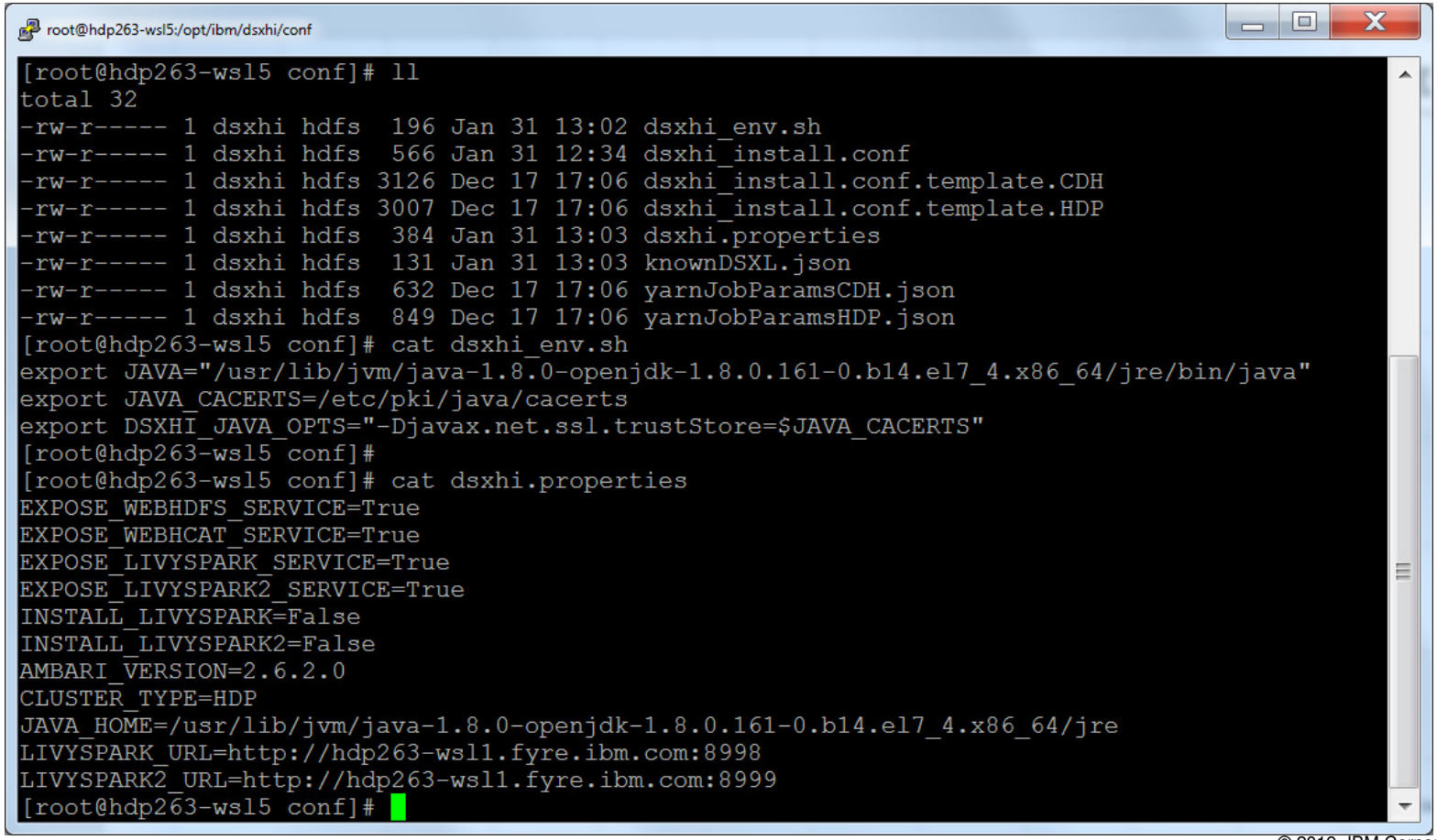
```
1  dsxhi_license_acceptance=a
2  dsxhi_serviceuser=dsxhi
3  dsxhi_serviceuser_group=hdfs
4  dsxhi_gateway_port=8443
5  dsxhi_rest_port=8082
6  cluster_manager_url=http://hdp263-wsl1.fyre.ibm.com:8080
7  cluster_admin=admin
8
9  exposed_hadoop_services=webhdfs,webhcat,livyspark,livyspark2
10
11 existing_webhcat_url=http://hdp263-wsl1.fyre.ibm.com:50111/templeton/v1
12 existing_webhdfs_url=http://hdp263-wsl1.fyre.ibm.com:50070/webhdfs/v1
13
14 existing_livyspark_url=http://hdp263-wsl1.fyre.ibm.com:8998
15 existing_livyspark2_url=http://hdp263-wsl1.fyre.ibm.com:8999
16
17 known_dsx_list=https://9.30.182.106
```


© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

✓ Example:



© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Installation

- **Install Livy connector (dsxhi)**
  - Install connector

    # /opt/ibm/dsxhi/bin/install.py --dsxhi_gateway_master_password=<password> --password=<password>

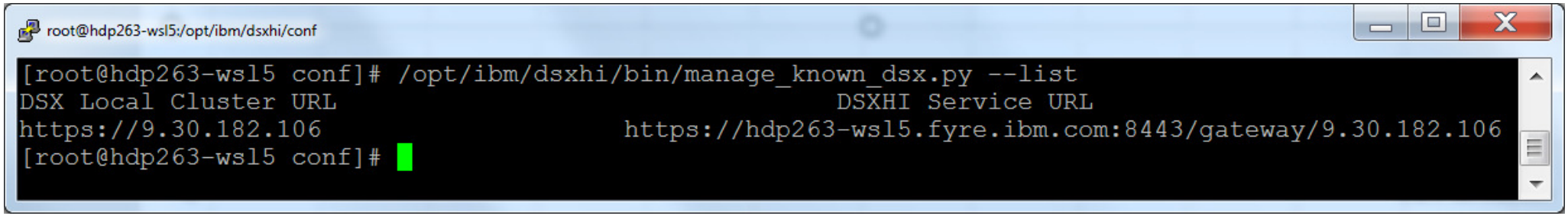
```
IBM Data Science Experience Hadoop Integration Service
------------------------------------------------------
Terms and Conditions: http://www14.software.ibm.com/cgi-
bin/weblap/lap.pl?la_formnum=&li_formnum=L-KLSY-
AVZW2D&title=IBM+Data+Science+Experience+Hadoop+Integration&l=en

--Determining properties
--Running the prechecks

--Install Livy for Spark / Spark 2
--Configure gateway
--Create template for gateway
--Setting up known DSX Local clusters
--Install dsxhi_rest
--Start all services
--Install finished. Check status in /var/log/dsxhi/dsxhi.log
```


© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Installation & Setup

Verify Installation

- **Livy connector process status check**
  - /opt/ibm/dsxhi/bin/status.py
- **List registered WSL cluster(s)**
  - /opt/ibm/dsxhi/bin/manage_known_dsx.py –list
- **Register/Add new WSL cluster**
  - /opt/ibm/dsxhi/bin/manage_known_dsx.py --add https://<WSL_FQDN>



© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

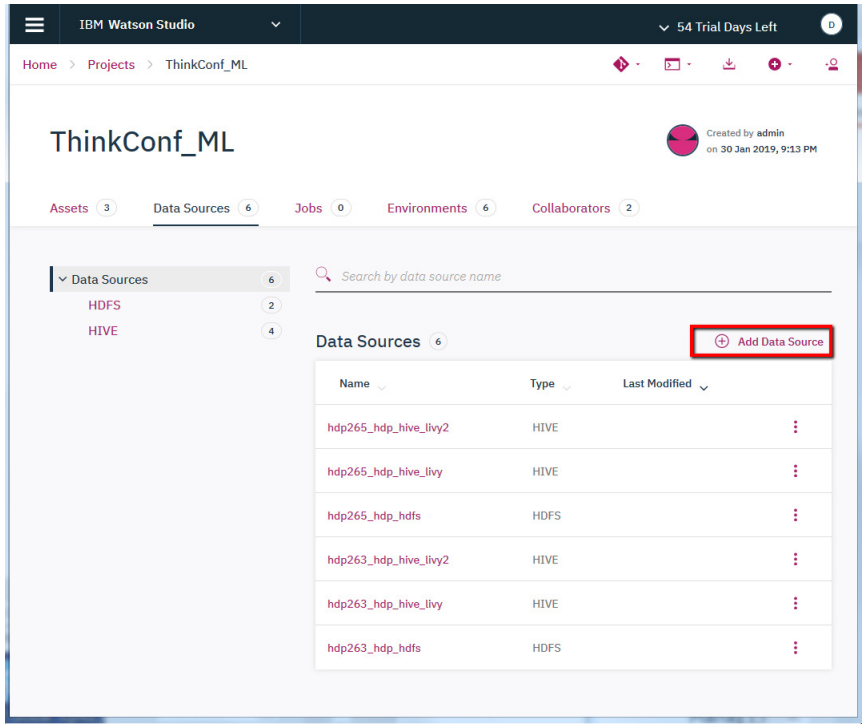
- Retrieve data from HDFS file system
- Access to Hive database table data
- Push down Spark job/task to the remote Spark2 server on Hadoop cluster

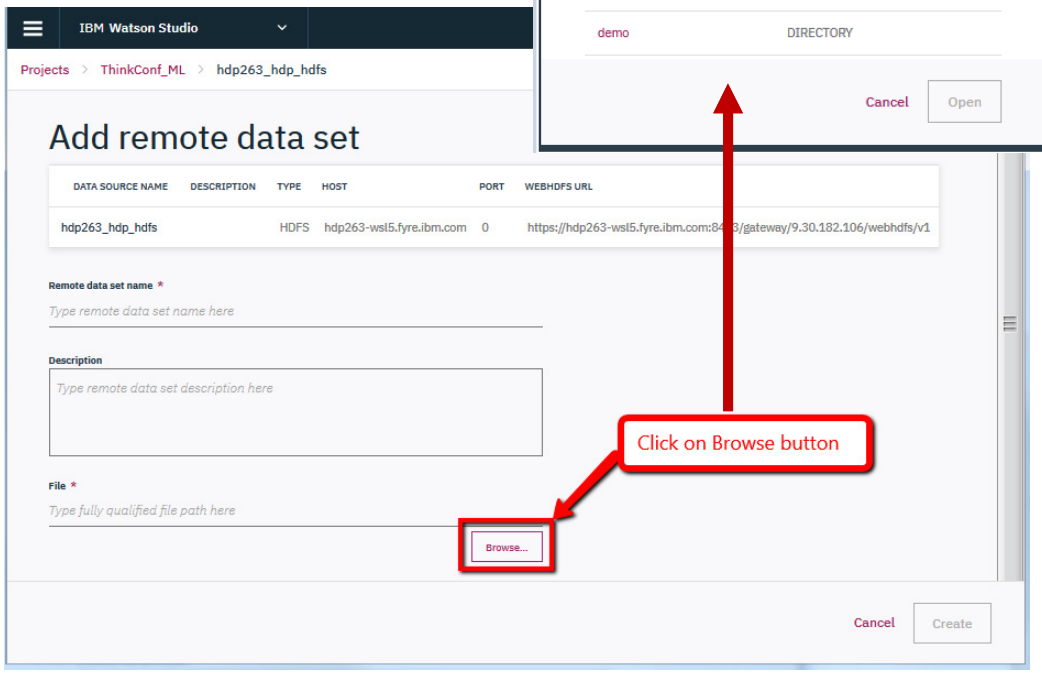
© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

1. Retrieving data from HDFS file system

- Add a remote data source from project dashboard



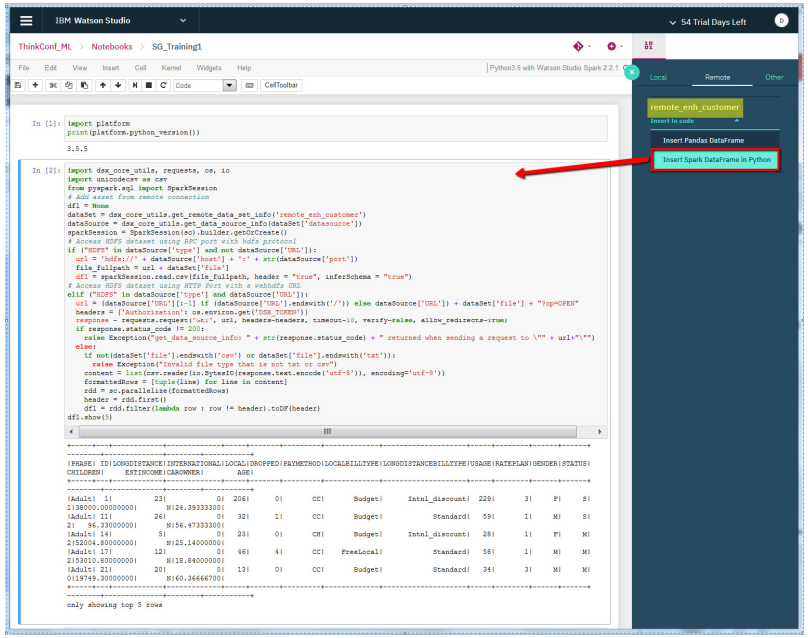


© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

2. Use added remote HDFS data source in Notebooks

- Click on icon ( ) and select the Remote data source created from previous instruction; then select “Insert Spark DataFrame in Python”.



© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

3. Load HDFS dataset via remote Spark session

- Example code:

```
import dsx_core_utils, requests, jaydebeapi, os, io, sys
import pandas as pd
url = 'https://hdp263-wsl5.fyre.ibm.com:8443/gateway/9.30.182.106/webhdfs/v1/data/demo/customer/enhanced_customer.csv?OP=OPEN'
headers = {'Authorization': os.environ.get('DSX_TOKEN')}
response = requests.request("GET", url, headers=headers, timeout=10, verify=False, allow_redirects=True)
df = pd.read_csv(io.StringIO(response.text, newline=None), sep=',')
df.head()
```

In [6]:

```
import dsx_core_utils, requests, jaydebeapi, os, io, sys
import pandas as pd
url = 'https://hdp263-wsl5.fyre.ibm.com:8443/gateway/9.30.182.106/webhdfs/v1/data/demo/customer/enhanced_customer.csv?OP=OPEN'
headers = {'Authorization': os.environ.get('DSX_TOKEN')}
response = requests.request("GET", url, headers=headers, timeout=10, verify=False, allow_redirects=True)
df = pd.read_csv(io.StringIO(response.text, newline=None), sep=',')
df.head()
```

Out[6]:

	PHASE	ID	LONGDISTANCE	INTERNATIONAL	LOCAL	DROPPED	PAYMETHOD	LOCALBILLTYPE	LONGDISTANCEBILLTYPE	USAGE	RATEPLAN	GENDER
0	Adult	1	23	0	206	0	CC	Budget	Intnl_discount	229	3	F
1	Adult	11	26	0	32	1	CC	Budget	Standard	59	1	M
2	Adult	14	5	0	23	0	CH	Budget	Intnl_discount	28	1	F
3	Adult	17	12	0	46	4	CC	FreeLocal	Standard	58	1	M
4	Adult	21	20	0	13	0	CC	Budget	Standard	34	3	M

Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

4. Load HDFS dataset via “PyWebHdfs” package

```
import json
from pywebhdfs.webhdfs import PyWebHdfsClient
headers = {'Authorization': os.environ.get('DSX_TOKEN')}
url = 'https://hdp263-wsl5.fyre.ibm.com:8443/gateway/9.30.182.106/webhdfs/v1'
hdfs = PyWebHdfsClient(base_uri_pattern=url, request_extra_opts={'verify': False}, request_extra_headers=head
print(json.dumps(hdfs.list_dir('/'), indent=1))
```

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

5. Access Hive data

- %%spark -s <session_name> -c sql -o <local_dataframe>

  select gender, count(gender) from ext_customer group by gender order by gender

  print(type(df_ex_cust_local))

  print(df_ex_cust_local)

In [9]:
```
%%spark -s $session_name -c sql
show tables
```

Out[9]:

	database	tableName	isTemporary
0	default	del2	False
1	default	ext_customer	False

In [12]:
```
%%spark -s $session_name -o df_ex_cust_local -c sql
select gender, count(gender) from ext_customer group by gender order by gender
```

Out[12]:

	gender	count(gender)
0	F	1316
1	GENDER	1
2	M	750

In [15]:
```
print(type(df_ex_cust_local))
print(df_ex_cust_local)
```

```
<class 'pandas.core.frame.DataFrame'>
   gender  count(gender)
0       F           1316
1  GENDER              1
2       M            750
```


© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

6. Push down Spark processing

In [25]:
```
import dsx_core_utils
%load_ext sparkmagic.magics
# Retrieve a list of registered Hadoop Integration systems.
DSXHI_SYSTEMS = dsx_core_utils.get_dsxhi_info(showSummary=True)
```

```
The sparkmagic.magics extension is already loaded. To reload it, use:
  %reload_ext sparkmagic.magics
Available Hadoop systems:

  systemName  LIVYSPARK  LIVYSPARK2 imageId
0     hdp263  livyspark  livyspark2
1     hdp265  livyspark  livyspark2
```

In [26]:
```
# Set up sparkmagic to connect to the selected registered HI systemName above.
myConfig={
"queue": "default",
"driverMemory": "512M",
"numExecutors": 1,
"executorMemory":"512M"
}
HI_CONFIG = dsx_core_utils.setup_livy_sparkmagic(
system="hdp263",
livy="livyspark2",
imageId=None,
addlConfig=myConfig)
# (Re-)load spark magic to apply the new configs.
%reload_ext sparkmagic.magics
```

```
sparkmagic has been configured to use https://hdp263-wsl5.fyre.ibm.com:8443/gateway/9.30.182.106/livy2/v1
success configuring sparkmagic livy.
```

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

6. Push down Spark processing

In [27]:
```
%spark cleanup
session_name = 'spark2_0205a'
livy_endpoint = HI_CONFIG['LIVY']
webhdfs_endpoint = HI_CONFIG['WEBHDFS']
%spark add -s $session_name -l python -u $livy_endpoint
```

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
4	application_1549042673081_0007	pyspark	idle	Link	Link	✓

```
SparkSession available as 'spark'.
```

In [28]:
```
%spark info
```

```
Info for running Spark:
    Sessions:
        Name: spark2_0205a      Session id: 4   YARN id: application_1549042673081_0007 Kind: pyspark   State: idle
        Spark UI: http://hdp263-wsl2.fyre.ibm.com:8088/proxy/application_1549042673081_0007/
        Driver Log: http://hdp263-wsl5.fyre.ibm.com:8042/node/containerlogs/container_e02_1549042673081_0007_01_000001/dsxhi
    Session configs:
        {'conf': {'spark.yarn.appMasterEnv.HI_UTILS_PATH': '/user/dsxhi/environments/pythonAddons/hi_core_utils.zip'}, 'proxyUser': 'ds
xhi', 'numExecutors': 1, 'queue': 'default', 'driverMemory': '512M', 'executorMemory': '512M'}
```


© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

6. Push down Spark processing

In [29]:
```
%%spark -s $session_name
import socket
print("Remote livy session driver: {}".format(socket.gethostname()))
```

```
Remote livy session driver: hdp263-wsl5.fyre.ibm.com
```

In [30]:
```
%%spark -s $session_name
import socket
print("Remote livy session driver: {}".format(socket.gethostname()))
```

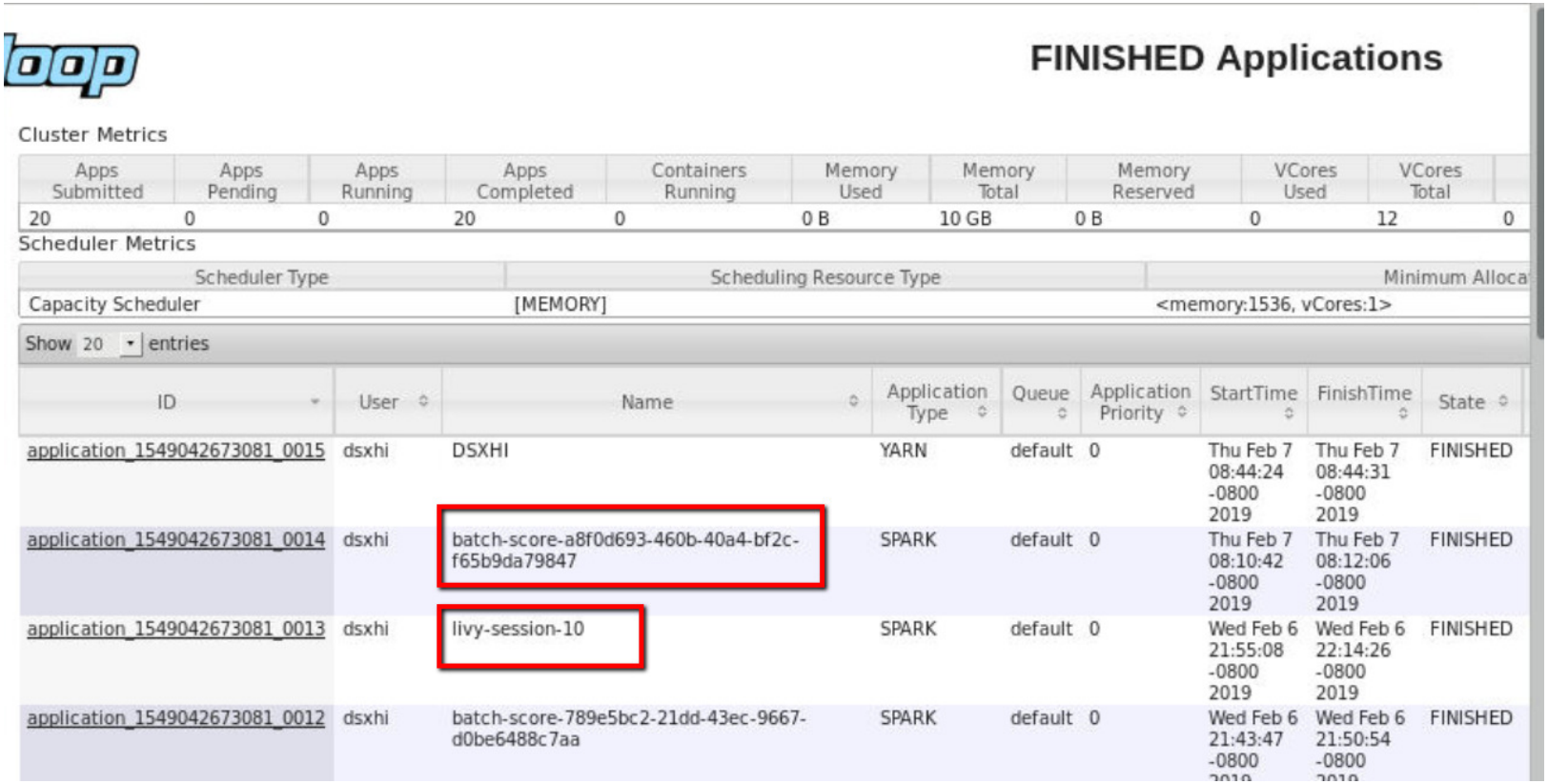
```
Remote livy session driver: hdp263-wsl5.fyre.ibm.com
```

In [32]:
```
%%spark

file_customer = "hdfs:///data/demo/customer/enhanced_customer.csv"
df_cust = spark.read.format("org.apache.spark.sql.execution.datasources.csv.CSVFileFormat").option("header", "true").option("inferSchema", "true").load(file_customer)
df_cust.printSchema()

print("Total number of customer dataset row count: {}\n".format(df_cust.count()))
```

```
root
 |-- PHASE: string (nullable = true)
 |-- ID: integer (nullable = true)
 |-- LONGDISTANCE: integer (nullable = true)
 |-- INTERNATIONAL: integer (nullable = true)
 |-- LOCAL: integer (nullable = true)
 |-- DROPPED: integer (nullable = true)
 |-- PAYMETHOD: string (nullable = true)
 |-- LOCALBILLTYPE: string (nullable = true)
 |-- LONGDISTANCEBILLTYPE: string (nullable = true)
 |-- USAGE: integer (nullable = true)
 |-- RATEPLAN: integer (nullable = true)
 |-- GENDER: string (nullable = true)
 |-- STATUS: string (nullable = true)
 |-- CHILDREN: integer (nullable = true)
 |-- ESTINCOME: double (nullable = true)
 |-- CAROWNER: string (nullable = true)
 |-- AGE: double (nullable = true)

Total number of customer dataset row count: 2066
```

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

6. Push down Spark processing – Review the remote Spark application



FINISHED Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	
20	0	0	20	0	0 B	10 GB	0 B	0	12	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Alloca
Capacity Scheduler	[MEMORY]	<memory:1536, vCores:1>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State
application_1549042673081_0015	dsxhi	DSXHI	YARN	default	0	Thu Feb 7 08:44:24 -0800 2019	Thu Feb 7 08:44:31 -0800 2019	FINISHED
application_1549042673081_0014	dsxhi	batch-score-a8f0d693-460b-40a4-bf2c-f65b9da79847	SPARK	default	0	Thu Feb 7 08:10:42 -0800 2019	Thu Feb 7 08:12:06 -0800 2019	FINISHED
application_1549042673081_0013	dsxhi	livy-session-10	SPARK	default	0	Wed Feb 6 21:55:08 -0800 2019	Wed Feb 6 22:14:26 -0800 2019	FINISHED
application_1549042673081_0012	dsxhi	batch-score-789e5bc2-21dd-43ec-9667-d0be6488c7aa	SPARK	default	0	Wed Feb 6 21:43:47 -0800	Wed Feb 6 21:50:54 -0800	FINISHED

© 2019 IBM Corporation

Hadoop Integration – Hadoop Gateway Testing & Troubleshoot

Delete the remote Spark session**

- %spark delete -s <session_name>

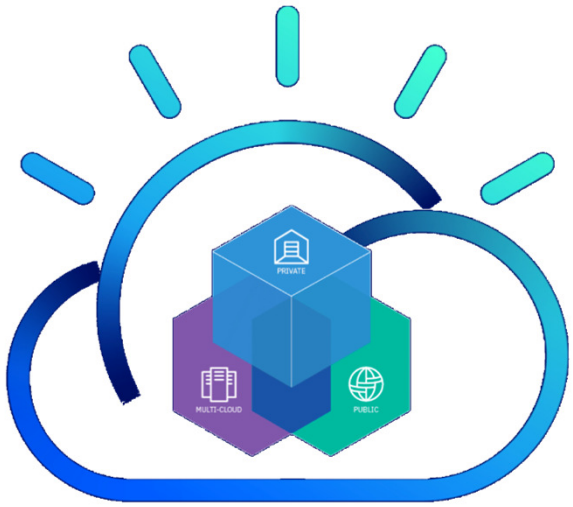
```
In [38]: %spark delete -s $session_name
```

```
In [23]: %spark cleanup
```

```
In [*]: %%javascript
        Jupyter.notebook.session.delete();
```


© 2019 IBM Corporation

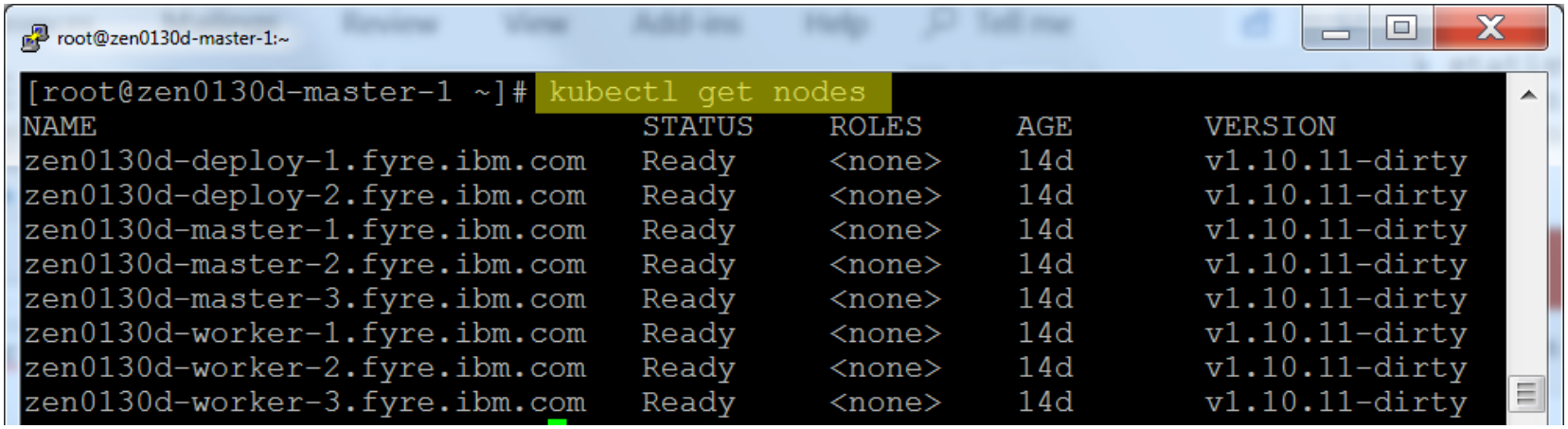
Watson Studio Local
Administration
Health Check



© 2019 IBM Corporation

Health Check

- Services need to be up and running on all cluster nodes
  - docker, kubelet, glusterd (ex. # systemctl enable docker; systemctl start docker)
- Verify gluster peer status (make sure communication among cluster nodes is working)
  - # gluster peer status
- Verify all cluster nodes status
  - # kubectl get nodes (FYI. Cluster has issues If any cluster node status is not shown “Ready”)



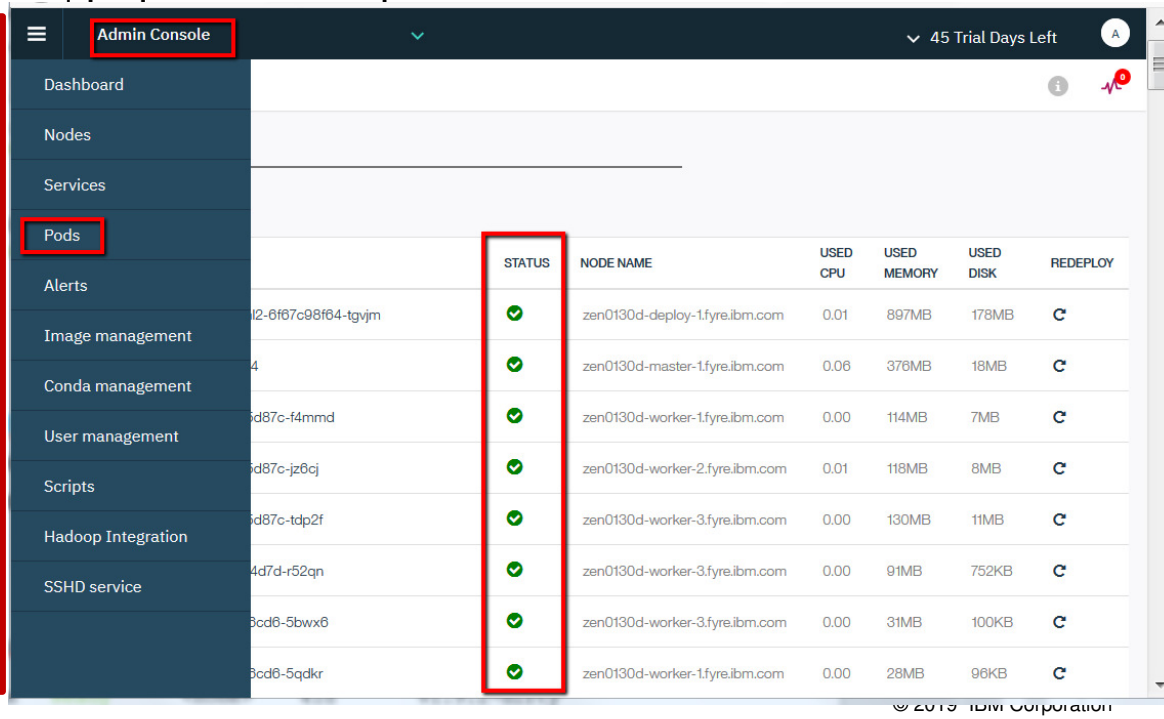
© 2019 IBM Corporation

Health Check

- Verify if Pods are all up and running
  - Verify from Admin Console GUI
  - OR via command line: # kubectl get [pods | po] --all-namespaces

Tip: If a pod is not “Running”, you can:

- Delete the Pod (it will auto create a new instance).
  Ex. kubectl delete po [--force] -n <namespace> <pod>
- Run describe command to review more detail
  Ex. kubectl describe po <pod> -n <namespace>
- Review Pod logs to investigate errors.
  Ex. kubectl logs <pod> -n <namespace>



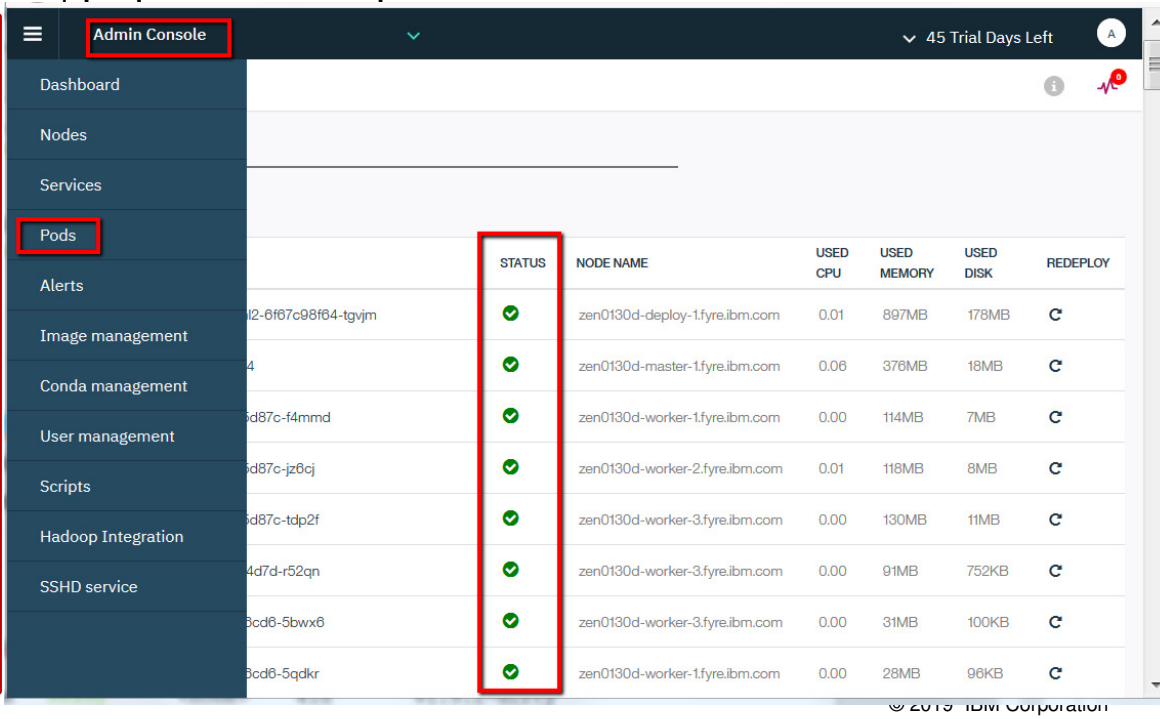
© 2019 IBM Corporation

Health Check

- Verify if Pods are all up and running
  - Verify from Admin Console GUI
  - OR via command line: # kubectl get [pods | po] --all-namespaces

Tip: If a pod is not “Running”, you can:

- Delete the Pod (it will auto create a new instance).
  Ex. kubectl delete po [--force] -n <namespace> <pod>
- Run describe command to review more detail
  Ex. kubectl describe po <pod> -n <namespace>
- Review Pod logs to investigate errors.
  Ex. kubectl logs <pod> -n <namespace>



© 2019 IBM Corporation

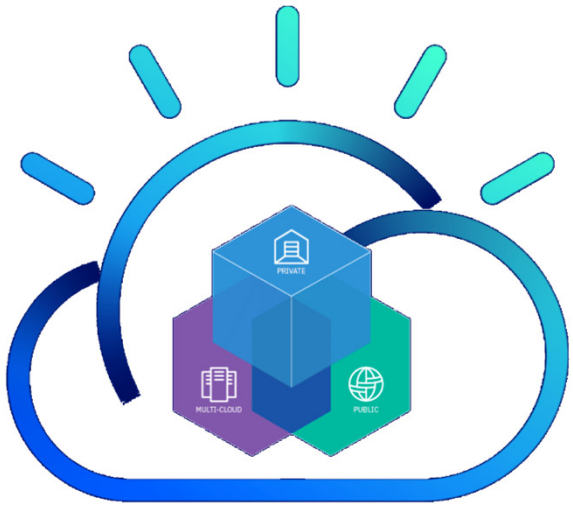
Health Check

- **Admin utility scripts**
  - Automatically troubleshoot and repair common problems in your Watson Studio Local cluster:
    - /wdp/utils/cluster_repair_utility.sh [-h]
  - Troubleshoot and print out results in a zipped tar file:
    - /wdp/utils/admin-utils.sh –user <username> --password –port <port> --key-pair <ssh_key_pair_file>

      Ex. /wdp/utils/admin-utils.sh --user root --password --key-pair /root/.ssh/id_rsa

© 2019 IBM Corporation

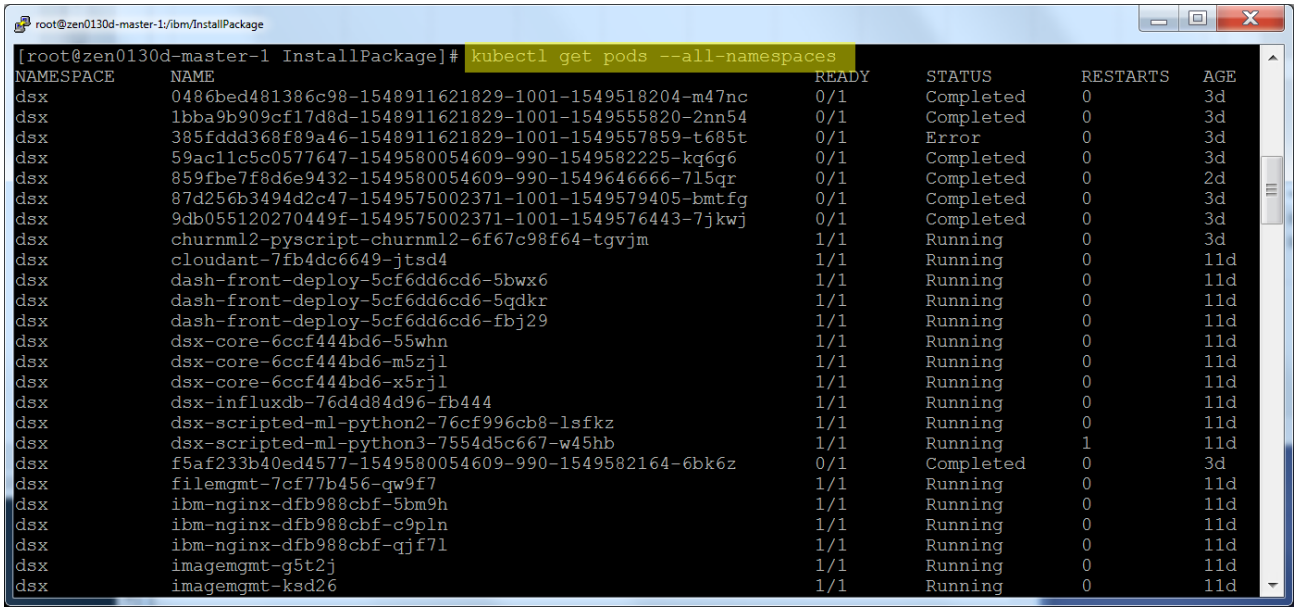
Watson Studio Local
Administration
Troubleshoots



© 2019 IBM Corporation

Access Pods

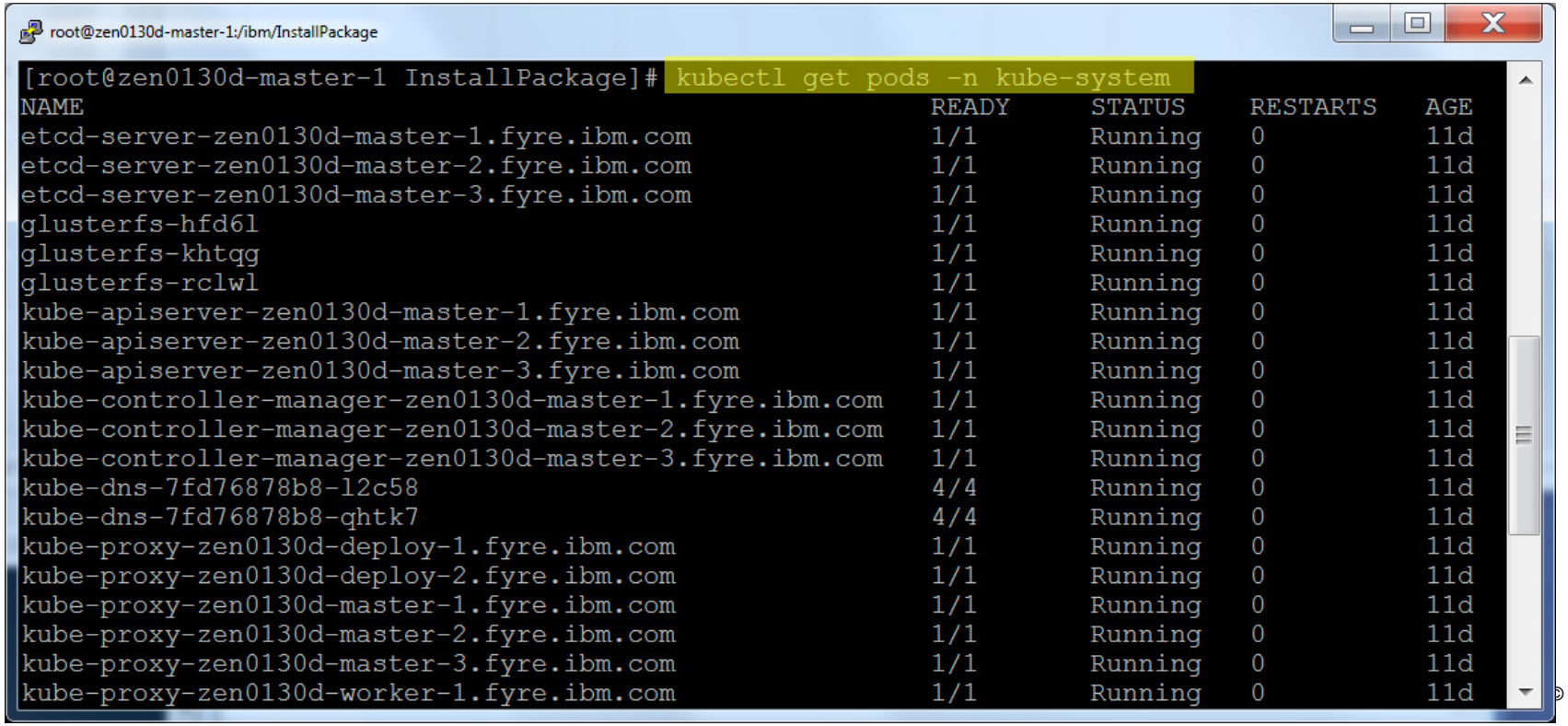
- List all Pods in the cluster
  - $ kubectl get pods --all-namespaces



© 2019 IBM Corporation

Access Pods

- List all Pods in the cluster based on namespace
  - $ kubectl get pods -n kube-system



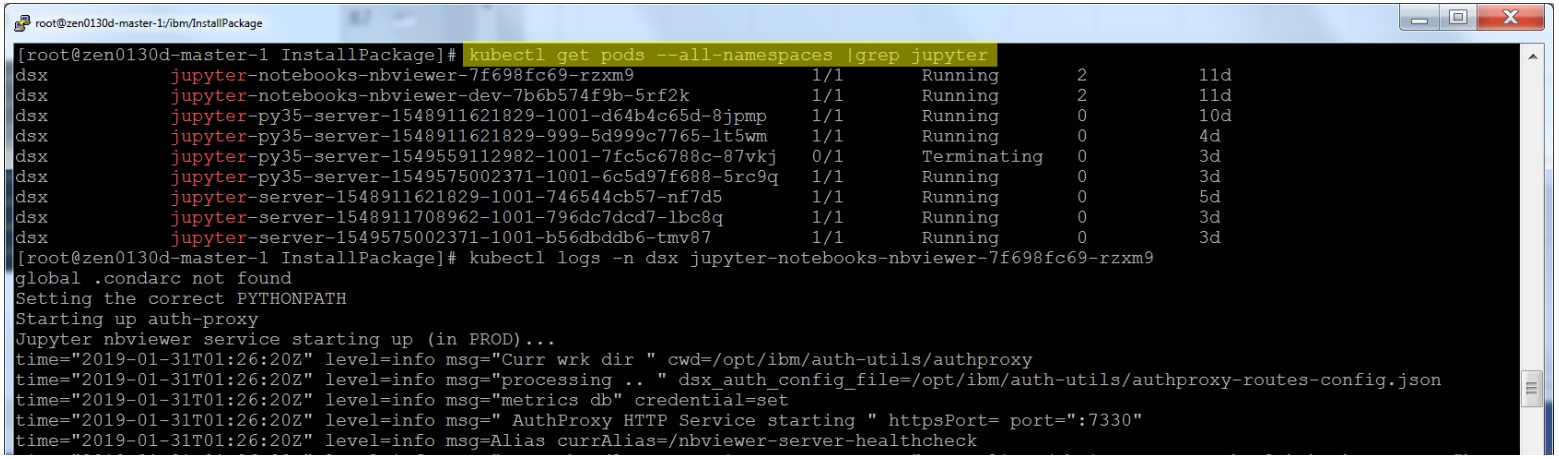
© 2019 IBM Corporation

Access Pods

- Log in to a Pod with shell
  - $ kubectl exec -it -n <namespace> <pod_name> [bash|sh]

    Example: $ kubectl exec -it -n kube-system glusterfs-hfd6l bash
- Access Pod's logs
  - $ kubectl logs –n <namespace> <pod-name>
- Restart a Pod
  - $ kubectl delete pod -n <namespace> <pod_name>

    Example: $ kubectl delete pod -n dsx jupyter-notebooks-nbviewer-7f698fc69-rzxm9



© 2019 IBM Corporation

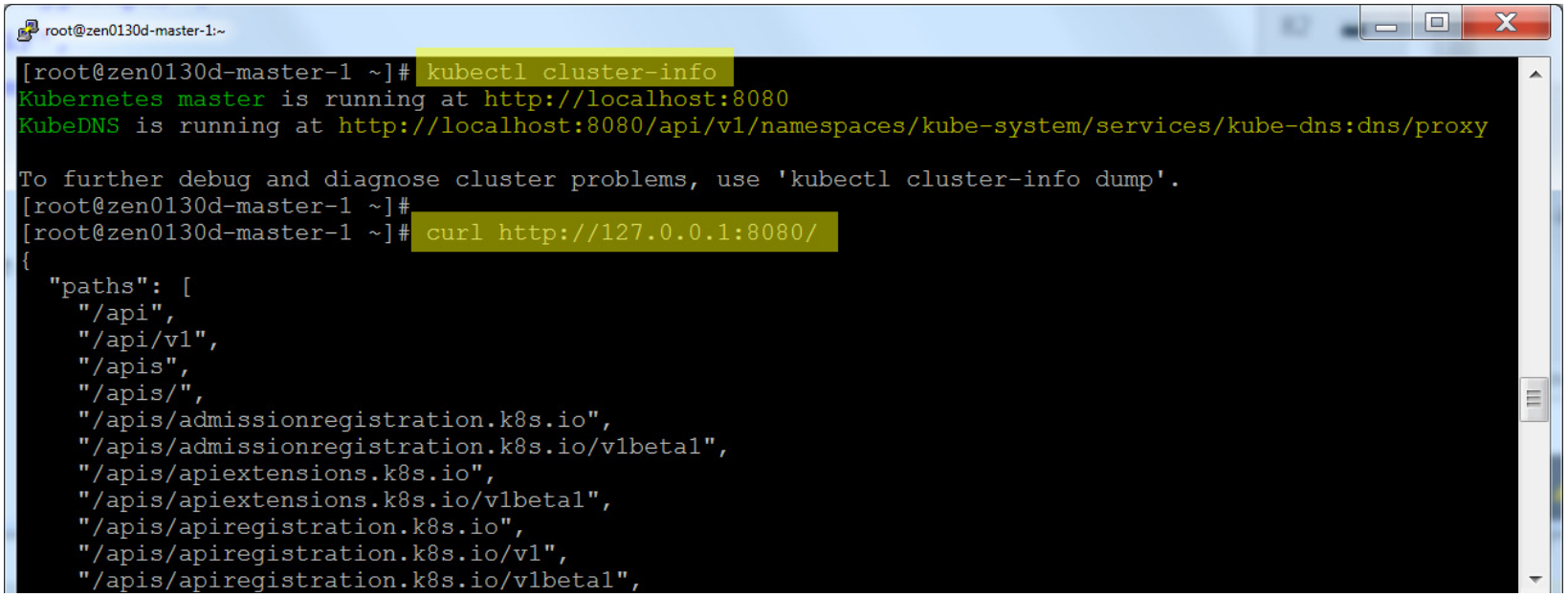
Docker 101

- Docker image vs container
  - A docker container is the runtime instance of an image
- Useful Docker commands
  - $ docker images → list all images that are locally stored with the Docker
  - $ docker rmi <image_name> → delete an image from the local image store
  - $ docker kill <container> → kill a running container
  - $ docker ps → list all “running” docker containers
  - $ docker ps -a → list all containers (running and not running)
  - $ docker inspect <container_id> → list detailed info about a docker container
  - $ docker logs <container_id> → show logs of a given container
  - $ docker build -t <image_name> . → build an image from the Dockerfile in the current directory and tag the image
  - $ docker [start | stop | rm] <container> → start: Launch a container previously stopped.
  - $ docker run <image> → create a new container of an image, and execute the container.
  - $ docker exec -it <container> [command] → ex. docker exec -it jupyter-notebooks-nbviewer-

©2019 IBM Corporation

Cluster information

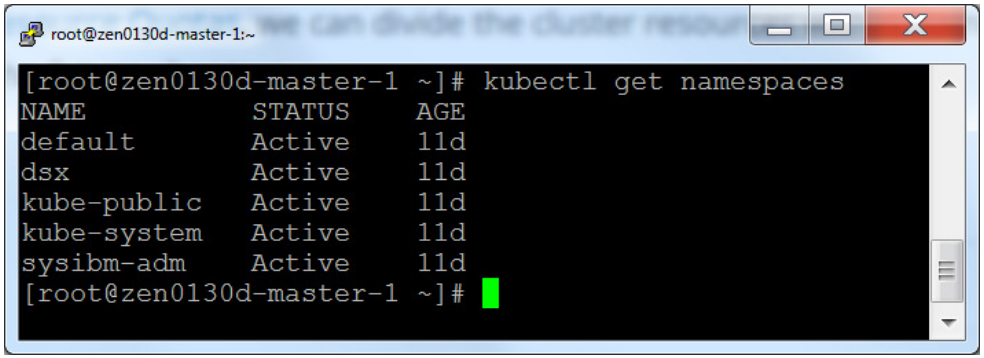
- Show cluster information
  - $ kubectl cluster-info
- List all API endpoint
  - $ curl http://localhost:8080/

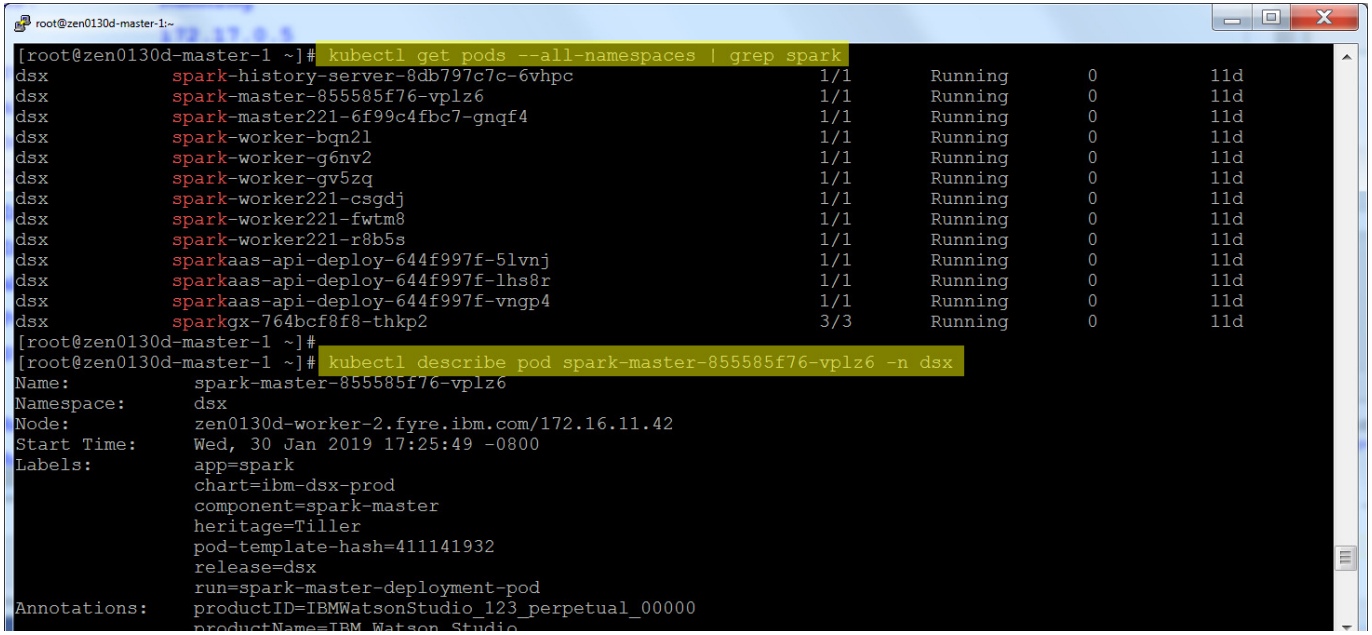


© 2019 IBM Corporation

Cluster / Pod information

- List all namespaces in the cluster
  - $ kubectl get namespaces
- Look at a Pod's details
  - $ kubectl describe pod <pod_name> -n <namespace>





Troubleshooting

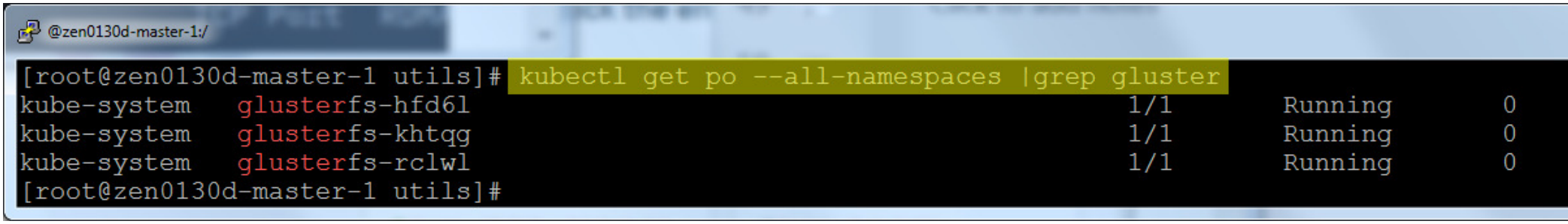
- Watson Studio Local provides various methods, scripts, and support for troubleshooting your cluster
- The areas we are going to cover include:
  - Troubleshoot common problems
  - Troubleshoot your system performance
  - Troubleshoot your system with administration utilities

© 2019 IBM Corporation

Troubleshoot Common Problems

- **Symptom:**
  - Watson Studio Local does not start, pods stuck creating or crashing, due to glusterfs in read-only or offline
- **Diagnostic Procedure:**
  1. Verify glusters bricks are online by using the following command:

     kubectl get po --all-namespaces | grep gluster



© 2019 IBM Corporation

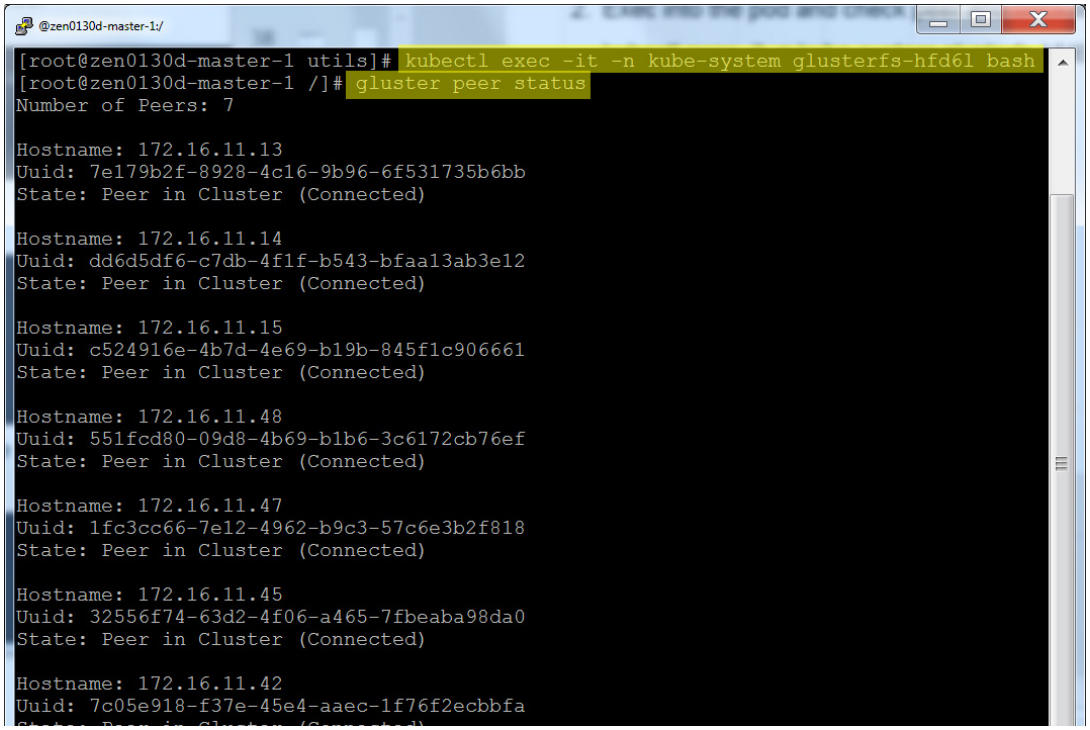
Troubleshoot Common Problems

- **Diagnostic Procedure (cont):**

  2. Exec into the pod and check peer and brick status using the following command:

     kubectl exec -it -n kube-system glusterfs-hfd6l bash

     gluster peer status

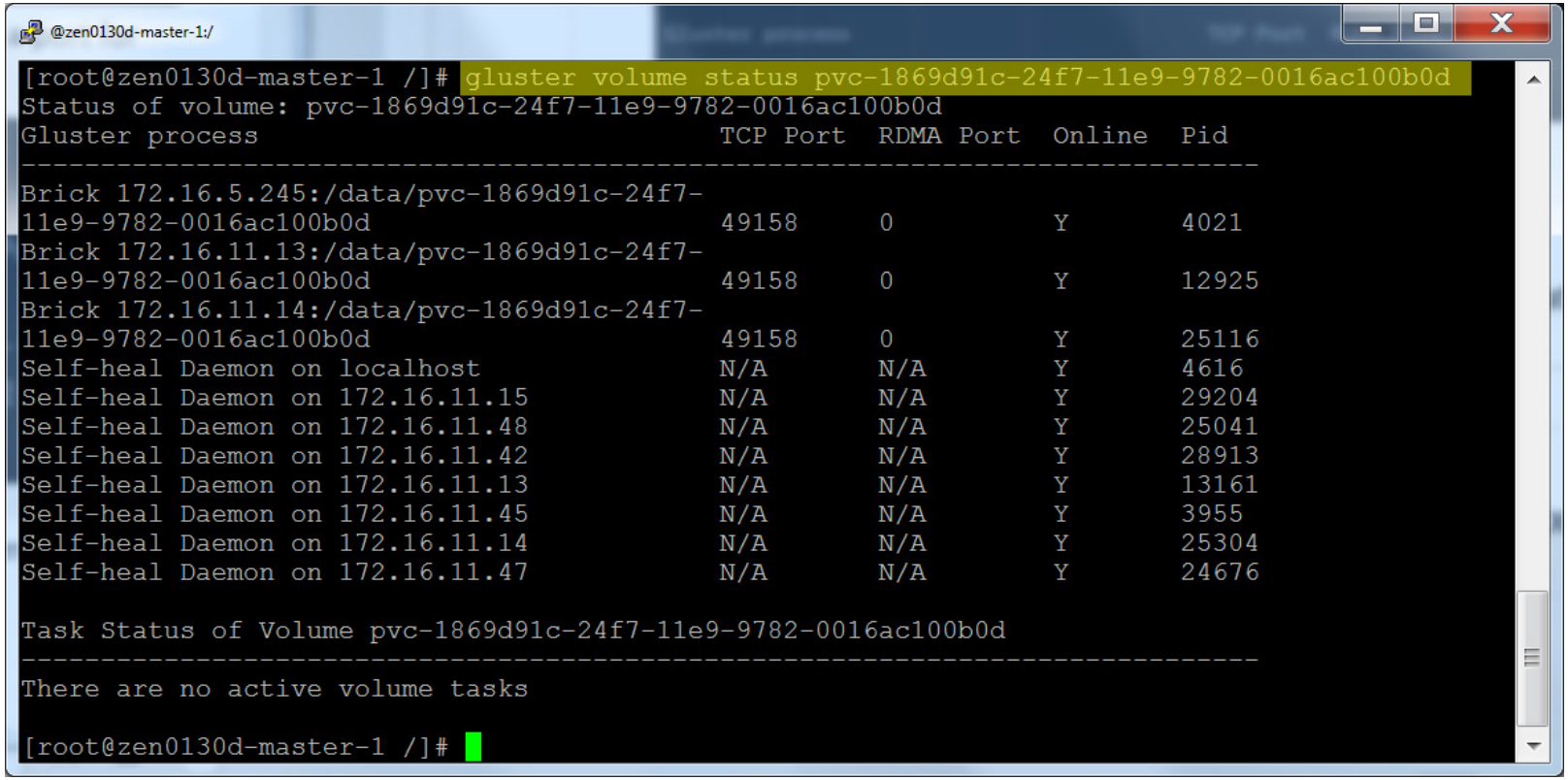


© 2019 IBM Corporation

Troubleshoot Common Problems

- **Diagnostic Procedure (cont):**

3. Enter gluster volume status. Verify that you see 3 bricks and show a port for the brick that is online:



© 2019 IBM Corporation

Troubleshoot Common Problems

- **Diagnostic Procedure (cont):**

4. If you do not, then run the following to attempt to connect the volumes that are not online:

   gluster volume stop <volume name>

5. Select Y to stop the volume
6. Enter gluster volume start <volume name>
7. Verify with gluster volume status, correct for all the volumes and then WSL should recover

© 2019 IBM Corporation

Troubleshoot Common Problems

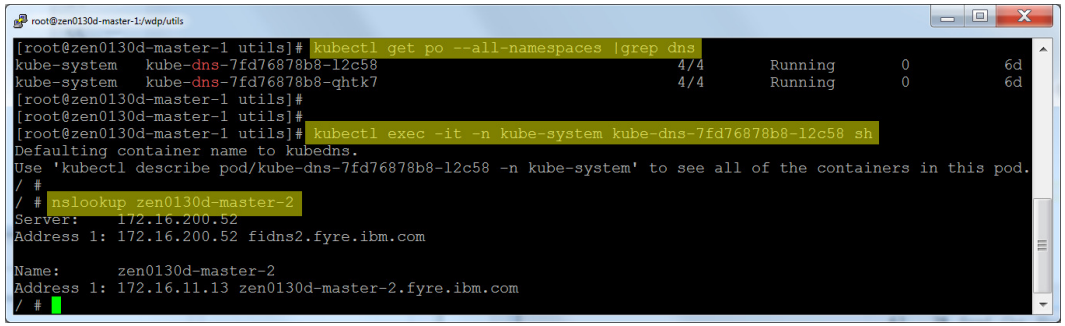
- **Symptom:**
  - Kubectl get no returns the following ‘The connection to the server localhost:8080 was refused – did y specify the right host or port?
- **Diagnostic Procedure:**
  1. Verify that at least two master nodes are online
  2. Check the status of docker:
  3. Check the status of kubelet:
  4. Check that the kube-apiserver is running: docker ps | grep kube-apiserver
  5. If not shown: docker ps -a | grep kube-apiserver, then check the logs: docker logs <api server contai
  6. If the logs look fine, check to verify that there is adequate space in the installer partition: df -lh
  7. Check logs for etcd container, and verify that you don't see messages about the time being out of sy

© 2019 IBM Corporation

Troubleshoot Common Problems

- **Symptom:**
  - ibm-nginx pods CrashLoopBack/Error
- **Diagnostic Procedure:**

1. Check logs for kubedns.kube-system.svc.cluster.local or usermgmt-svc.dsx.svc.cluster.local
2. Check logs of kubedns pods / restart kubedns
3. On the host, check the kubedns: nslookup <cluster_node>
4. Exec into dsx-core pod, and check dns resolution in the pod:

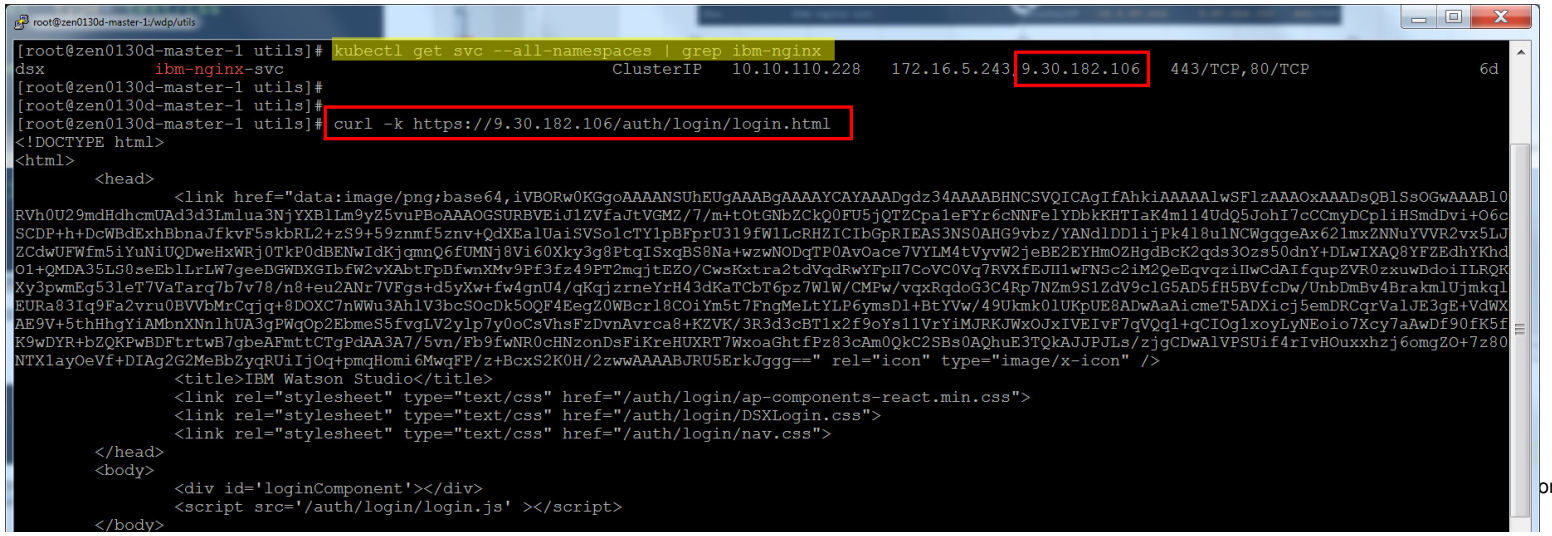


5. If resolving names doesn't work, check /etc/resolv.conf setting; and confirm that outside DNS lookups work: nslookup <hostname on network>

© 2019 IBM Corporation

Troubleshoot Common Problems

- **Symptom:**
  - Connecting to WSL from browser fails but ALL pods show as running
- **Diagnostic Procedure:**
  1. Verify that you can ping the proxy IP or hostname
  2. Connect to one of the nodes and attempt to connect to the site internally
  3. Verify site IP:



Troubleshoot Common Problems

- **Diagnostic Procedure (cont):**

  If the page comes up as previous screenshot, Watson Studio Local works internally.

  4. Check if firewalls are enabled:

     systemctl status firewalld

     Systemctl status iptables

     Verify that there are no firewalls, proxies, or blocked 443 port from the system.

© 2019 IBM Corporation

Troubleshoot System Performance

- If performance slows for computing jobs, or stalls to the point where no new projects or assets can be created anymore, reserve more CPU and memory for your runtime in the Environments page
- Additionally, you can stop the runtime environment for idle notebooks to free up more resource

```
In [10]: %spark cleanup
```

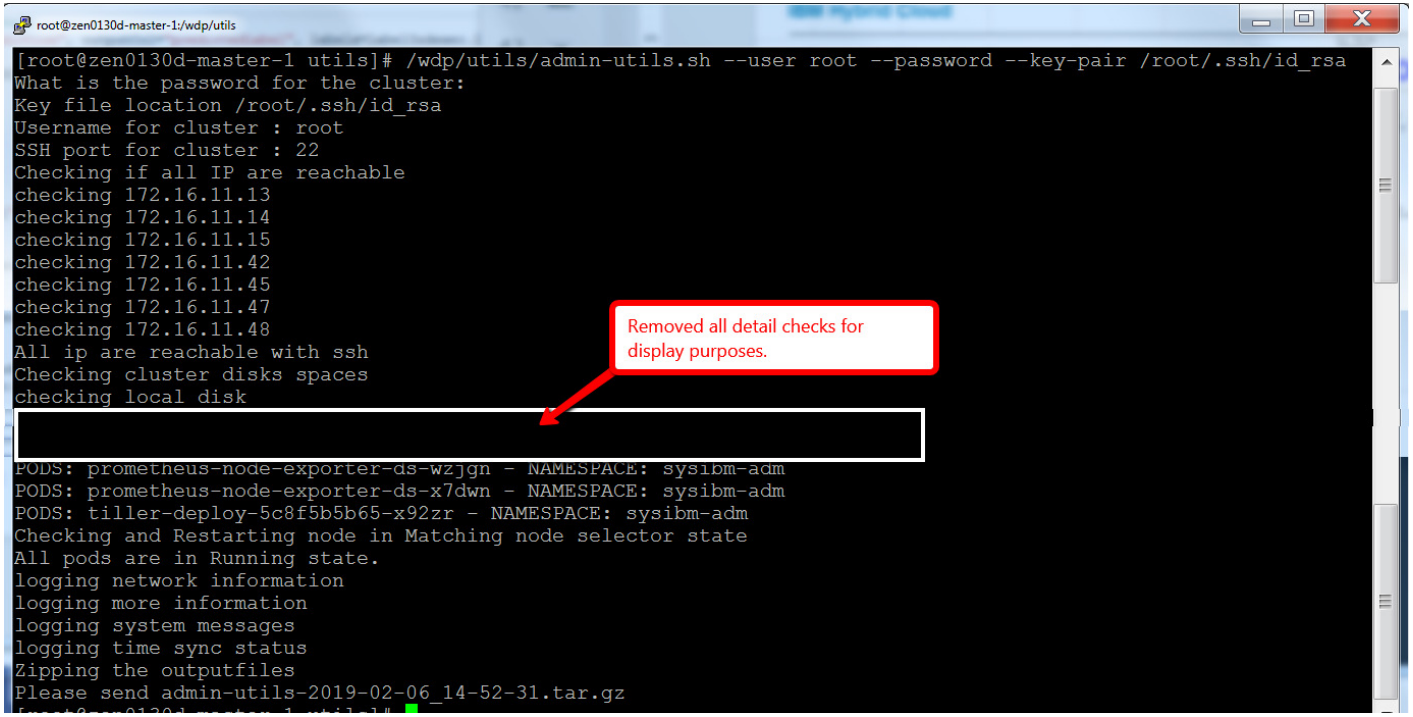
```
In [ ]: %%javascript
        Jupyter.notebook.session.delete();
```


© 2019 IBM Corporation

Troubleshoot using Administration Utilities

- To automatically troubleshoot your entire WSL cluster, run the */wdp/utils/admin-utils.sh* script by entering the following command:

  /wdp/utils/admin-utils.sh --user <userid> --password <--port port_number> --key-pair <ssh_key_file>



© 2019 IBM Corporation

Troubleshoot using Administration Utilities

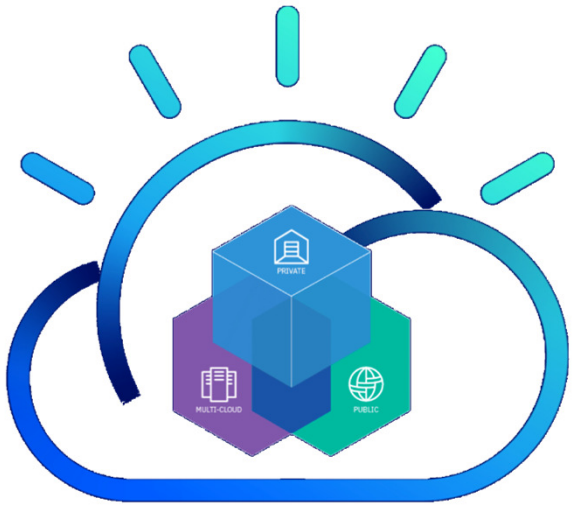
- The command performs the following tasks:
  - Checks whether all nodes are up and accessible with SSH
  - Checks free disk space on all nodes
  - Checks whether the Docker and Kubelet are running
  - Checks whether Gluster is installed and the volumes are up
- If all of the checks pass, then the script automatically logs everything that is not in the state ‘Ready’ or ‘Running’ into the following compressed file:
  - */wdp/utils/admin-utils-<timestamp>.tar.gz*
- You can then send this file to IBM Support to diagnose or you can take a look for yourself

© 2019 IBM Corporation

Watson Studio Local

Administration

Image Management



© 2019 IBM Corporation

Manage Images

- WSL administrator can create an image that everyone in the cluster can use.
- Images contain a specific runtime (for example, a Jupyter notebook, RStudio, or Zeppelin notebook) and a set of packages and libraries
- In the Admin Console, click the menu icon () and click Image Management to augment the runtime images that Watson Studio Local users can get started from.
- When the WSL administrator deploys them, users can conveniently select them from the Environments page of their project without having to upload any of the packages and libraries themselves
  - Alternatively, you can have your admin install packages globally which we covered in a previous training session

© 2019 IBM Corporation

Manage Images

Complete the following steps to customize an image:

1. Download an existing image
2. Modify the existing image
3. Upload the customized image
4. Approve the customized image
5. Delete the customized image

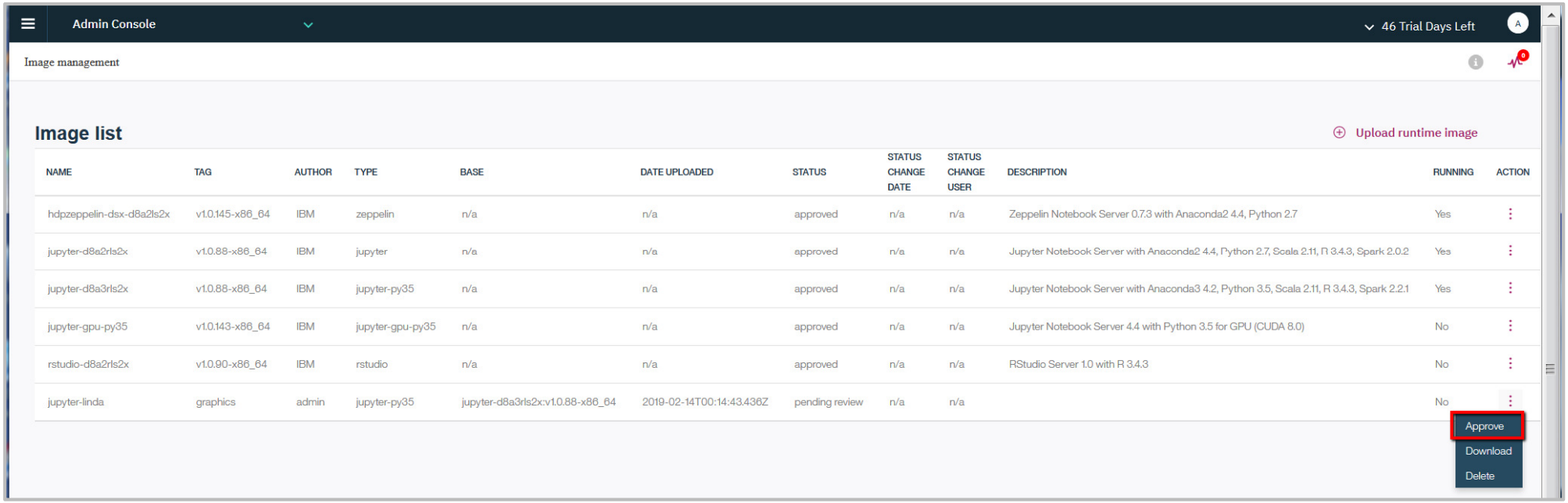
Tip: you can only delete a non-IBM image (that is NOT running) by clicking Delete next to it. If you do NOT see the Delete button for a non-IBM image, that means the image is currently in use by at least one user. To stop a runtime environment from using the image, go to the project's environment page, find the environment that is using the image, and click the Stop icon ( ) on it.

© 2019 IBM Corporation

Manage Images

Download an existing Image

- In the Image Management page, to to the Image List and click Download next to the runtime image you want to customize.



© 2019 IBM Corporation

Manage Images

Modify the downloaded Image

- Install Docker if not already. Download Docker Community Edition for details.
- Change the working directory to the folder with downloaded image.
- Load the image with command: # docker load -i <image_name>.tar.gz
- Create a file named Dockerfile and use the loaded image as the base image. Then add/customize with your need. **Example**:

```
[root@hdp265-edge1 p2]# docker load -i jupyter-d8a3rls2x_v1.0.88-x86_64.tar.gz

Loaded image: idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64
[root@hdp265-edge1 p2]#
[root@hdp265-edge1 p2]#
[root@hdp265-edge1 p2]# cat Dockerfile
FROM idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64
RUN apt-get update
RUN apt-get install -y graphviz
RUN pip install pydot
[root@hdp265-edge1 p2]# docker build -t idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64-Linda .
Sending build context to Docker daemon 3.613 GB
```


© 2019 IBM Corporation

Manage Images

Modify the downloaded Image

- Build the image with a new identifier. **Example**:

```
[root@hdp265-edge1 p2]# docker load -i jupyter-d8a3rls2x_v1.0.88-x86_64.tar.gz

Loaded image: idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64
[root@hdp265-edge1 p2]#
[root@hdp265-edge1 p2]#
[root@hdp265-edge1 p2]# cat Dockerfile
FROM idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64
RUN apt-get update
RUN apt-get install -y graphviz
RUN pip install pydot
[root@hdp265-edge1 p2]# docker build -t idp-registry.sysibm-adm.svc.cluster.local:31006/jupyter-d8a3rls2x:v1.0.88-x86_64-Linda .
Sending build context to Docker daemon 3.613 GB
```

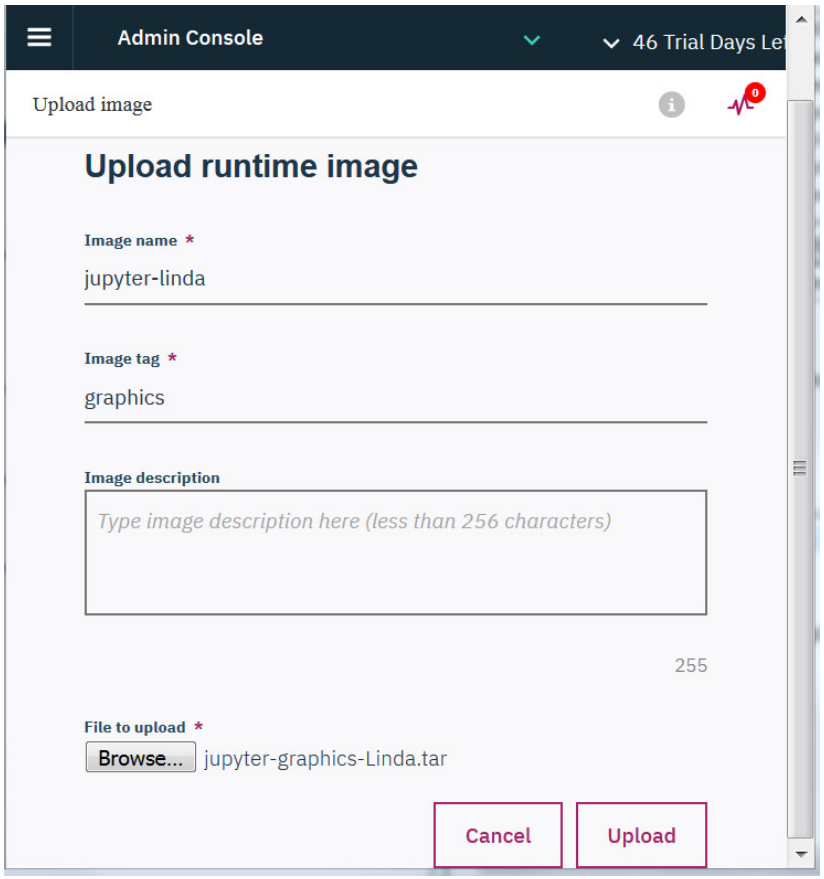
- Save the image and compress it as a tar file. **Example**: # docker save <modified_img_name> | gzip > <modified_name>.tar.gz

© 2019 IBM Corporation

Manage Images

Upload the customized image

- In the Admin console Image Management page, click Upload runtime image. **Example**:
- Type in a new image name, tag, and description.
- From the browse button, select the TAR.GZ file to be uploaded and click the Upload button.

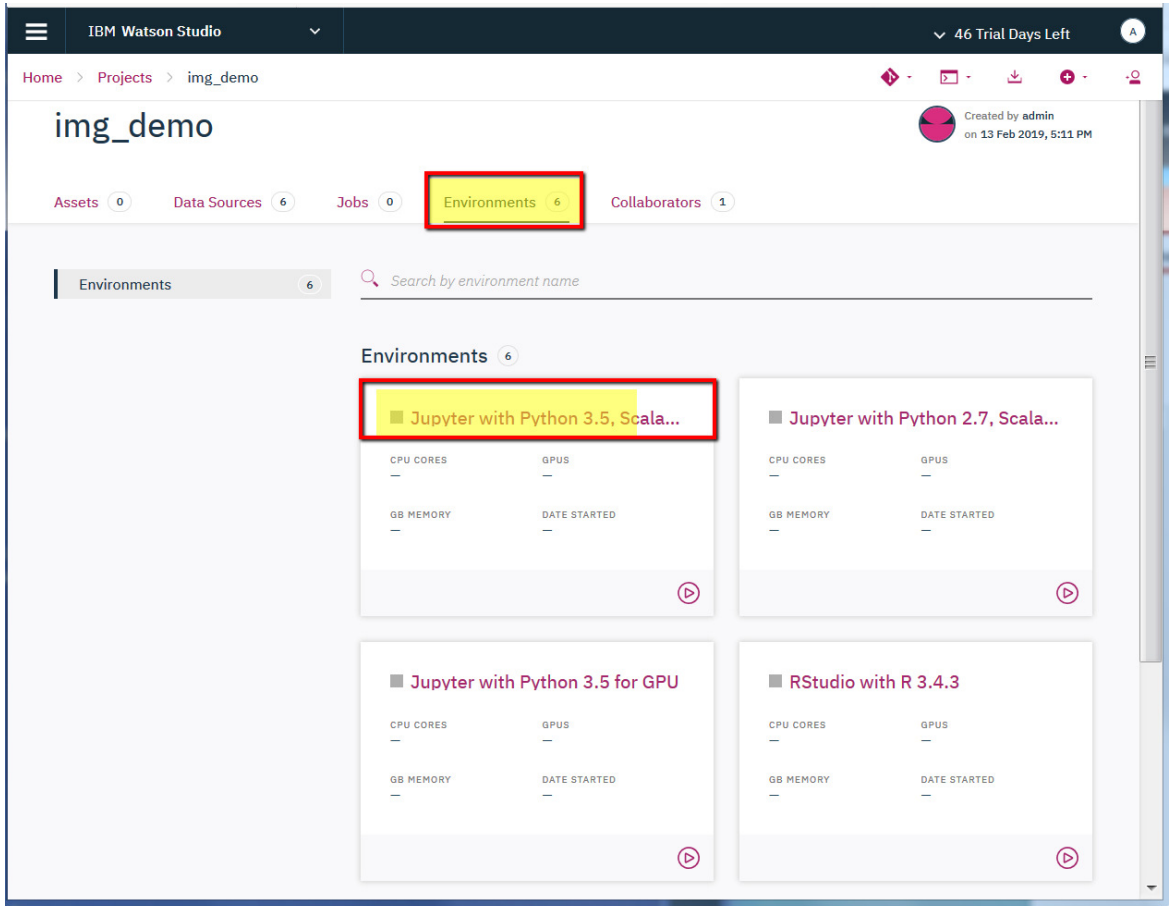


© 2019 IBM Corporation

Manage Images

Edit Environment

- From the project **Edit Environment** page, the newly uploaded image should display “Pending” status.
- To stop a runtime environment from using the image, click the Stop icon from the environment page. ( )

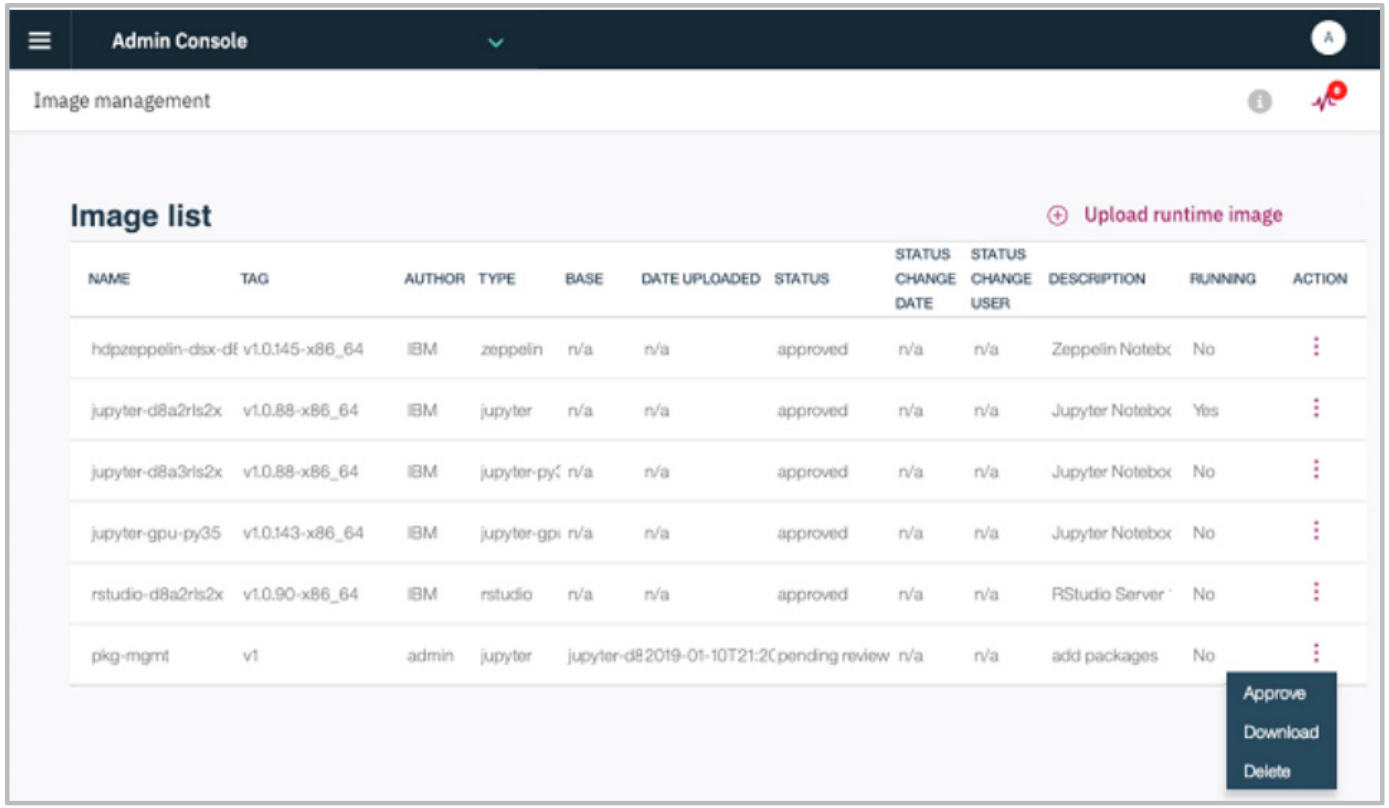


© 2019 IBM Corporation

Manage Images

Approve / Un-approve / Delete an Image

- A WSL administrator can click **Approve** / **Up-approve** / **Delete** next to the newly uploaded image.
- If you do not see the Delete in the dropdown list for a non-IBM image, that means the image is currently in use by at least one user.

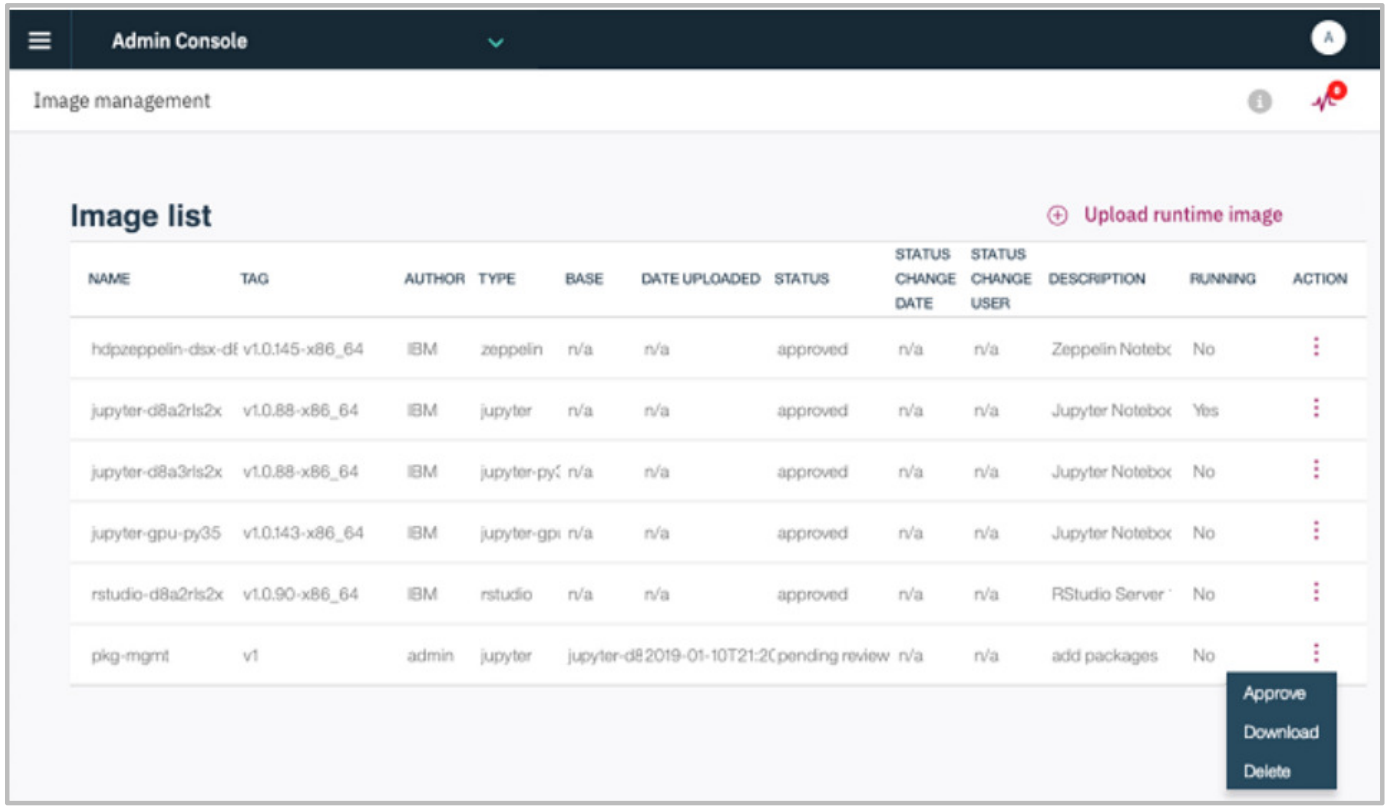


© 2019 IBM Corporation

Manage Images

Approve / Un-approve / Delete an Image

- A WSL administrator can click **Approve** / **Up-approve** / **Delete** next to the newly uploaded image.
- If you do not see the Delete in the dropdown list for a non-IBM image, that means the image is currently in use by at least one user.

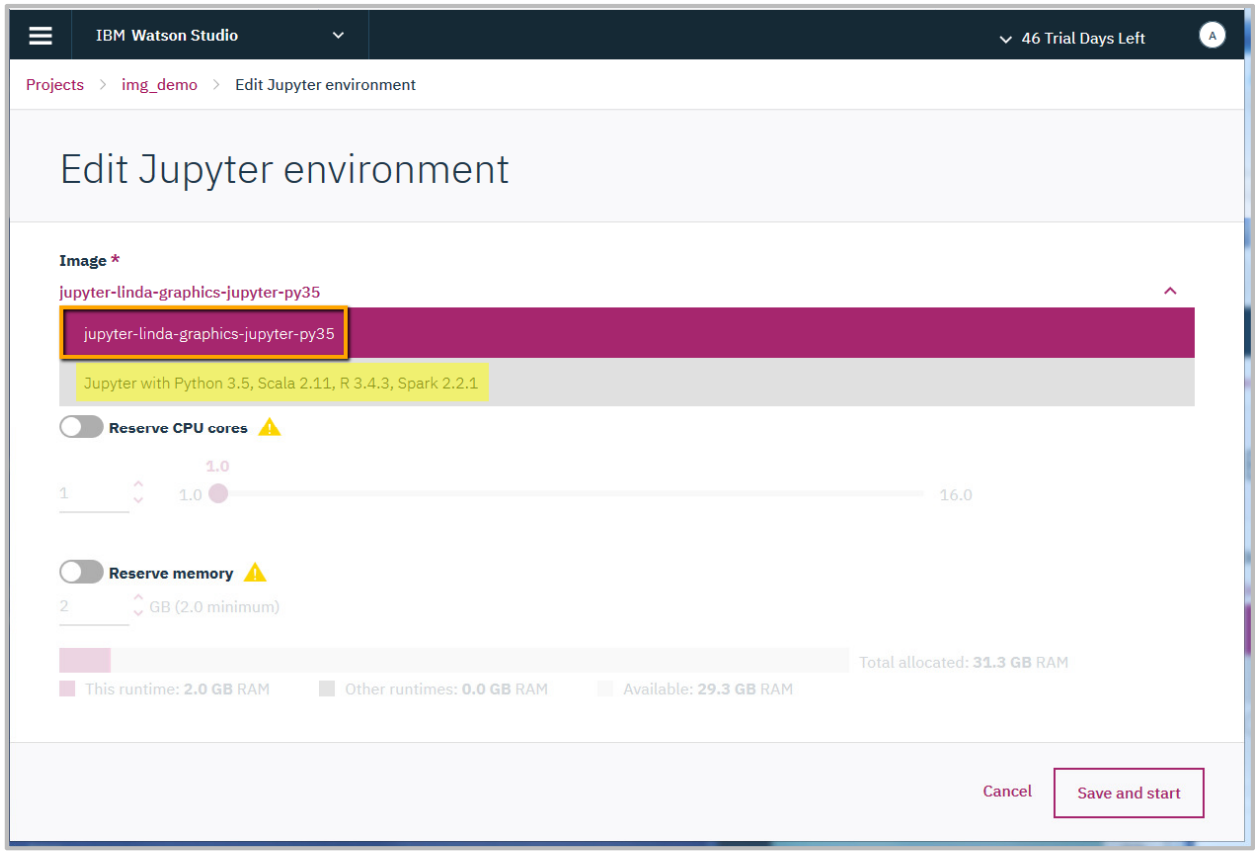


© 2019 IBM Corporation

Manage Images

Use custom image

- In your project **Environment** page, click the environment name to edit.
- For Image, select the custom image and click the Save and restart button.

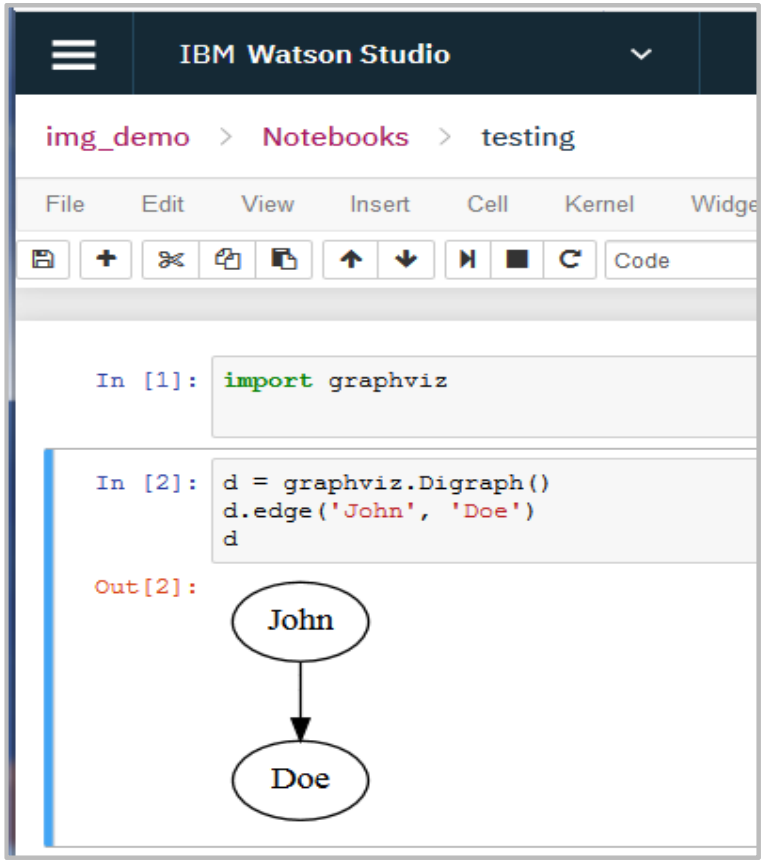


© 2019 IBM Corporation

Manage Images

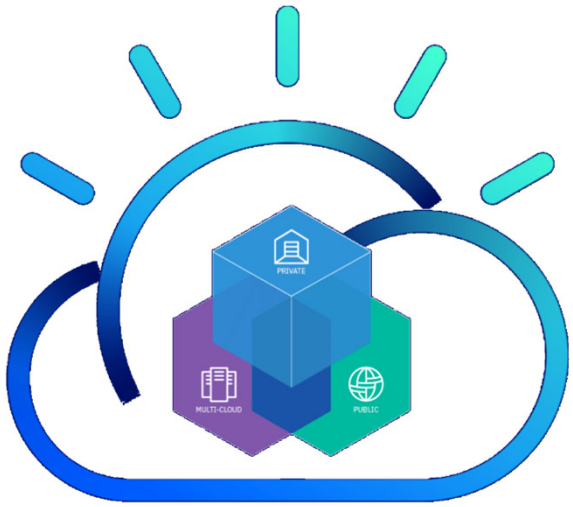
Use custom image

- Go to your project **Assets** page, click **Notebooks**, and click **Add Notebook** button.
- Start using the new customized image. **Example**:



© 2019 IBM Corporation

Watson Studio Local
Administration
Model Management & Deployment



© 2019 IBM Corporation

Model Management and Deployment in Watson Studio Local

- To expose a checkpoint of assets to outside users, a Deployment Admin can create a project release and deploy the assets within it.
- A project release represents a project tag that can be launched as a production environment within Watson Studio Local.
- The Watson Machine Learning client requires Deployment Admin permissions to create a project release.
- Only the WSL administrator can assign **Deployment Admin** permissions to a WSL user.
- **Prerequisite**: At least one WSL deployment node must be installed and available for a project release to launch successfully. If no deployment node was installed, the WSL administrator can add a new deployment node (that meets the software requirements) from the Nodes page of the Admin Console.

© 2019 IBM Corporation

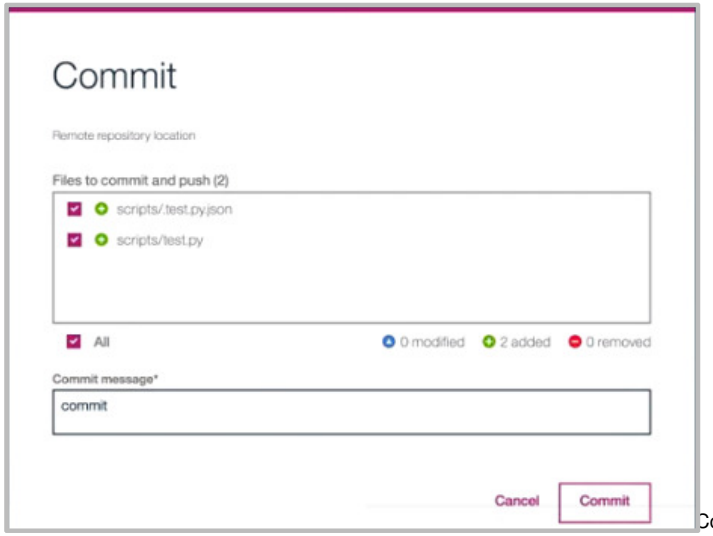
Model Management and Deployment in Watson Studio Local

- The Watson Machine Learning client provides the following tasks:
  - Creating a Project Release
  - Assign Members
  - Deploy Assets
  - Launch a Project Release
  - Manage a Project Release
  - Manage Deployments
  - View Data Sources
  - View Active Environments
  - Edit workers
- We will go over each of these tasks in the following slides

© 2019 IBM Corporation

Model Management & Deployment – Create a Project Release

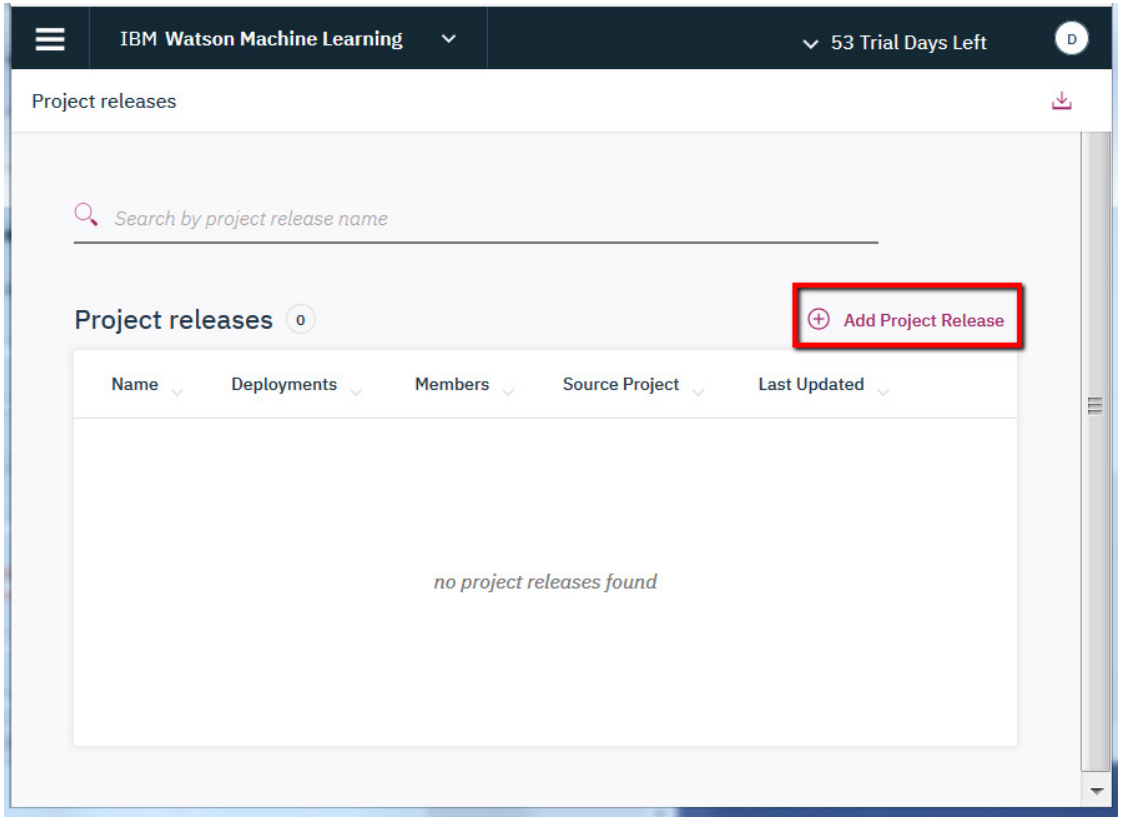
- This is the first step required when wanting to deploy an asset
- A Deployment Admin creates a project release out of the state of a project at a given point in time as denoted by a user-defined tag.
- After the project release is created, assets within this project release can then be deployed
- If you create a project release from a Watson Studio Local source project, this source project must have a tag (specified from the Commit and push window)



Corporation

Model Management & Deployment – Create a Project Release

1. In the Watson Machine Learning client, go to the Project releases page and click Add Project Release.



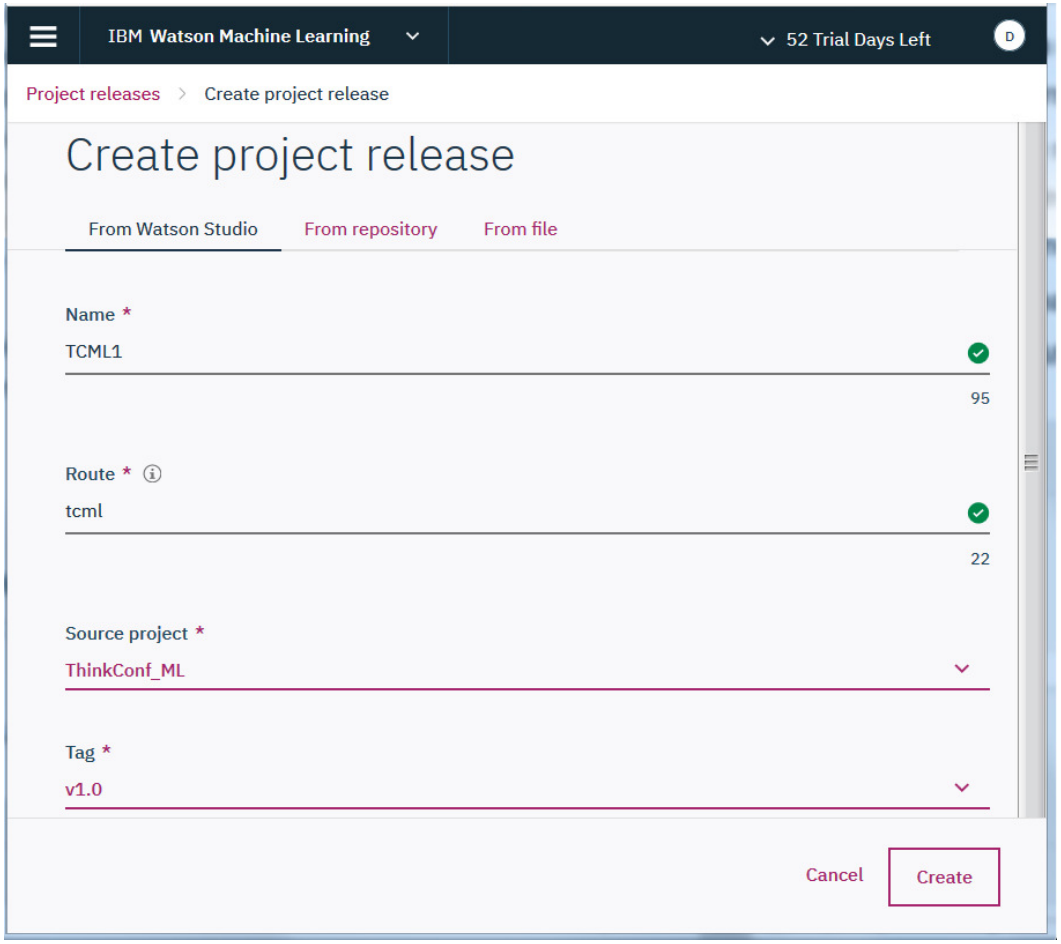
© 2019 IBM Corporation

Model Management & Deployment – Create a Project Release

2. Click **From Watson Studio Local** and select a source project and tag in WSL to associate the release to. Click **From repository** to import a release from a Git repo (a personal access token is required). Click **From file** to import a pre-existing project from your local device (note that the file must be an exported project). Ensure the file name and release name do not contain a space.
3. Type in a name that is descriptive and unique. The name can contain hyphens but not special characters such as a period (.).
4. Type the route to use. The route is the unique ID for the project release, and is used within the deployments' REST paths and URLs. It can contain at least 2 and at most 26 lowercase alphanumeric characters and hyphens, and must start with a letter and end with a letter or number
5. Click Create
6. Watson Machine Learning creates an offline release, unlocked for editing.

© 2019 IBM Corporation

Model Management & Deployment – Create a Project Release



© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

- An Admin can deploy an asset into a project release
- The deployment action creates a read-only snapshot of the asset
- The following types of assets can be deployed:

Asset	Job	Web service	App	Notes
Notebooks	Y	N	Y	Only Jupyter notebooks can be deployed. Zeppelin and H2O Flows are not supported by Watson Machine Learning.
Models	N	Y	N	Requires that you associate a batch scoring script with it. You cannot deploy Custom Batch models as a web service (must be Custom Online).
R Shiny	N	N	Y	Deploys the R code inside of an R pod.
Flows	Y	N	N	Flows from SPSS Modeler.
Decision Optimization Models	N	Y	N	Provides a REST API to submit and execute optimization jobs.
Scripts	Y	Y	N	
Model groups	N	Y*	N	* Can be deployed as a web service group.

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

- When the project release launches, each enabled deployment automatically generates either a permalink URL or REST API endpoint where the asset can be accessed:
  - **Job:** Generates a REST API endpoint to start, stop, and get status for that job. The job can be scheduled (but not unscheduled), and is visible to authenticated users only
  - **Web Service:** Generates a REST API endpoint for an external application to call. The web service is visible to authenticated users only

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

- **Web Service Group:** Generates a REST API endpoint for an external application to call. The web service group is visible to authenticated users only. In the Edit test settings page, you can set the leader and enable the following routing configurations:
  - **Leader:** Allows API requests to the leader of the group. The Routing-Option header in the API request must be set to leader. This is the default.
  - **Specific Model:** Allows API requests to a specific model version in the group. The Routing-Option header in the API request must be set to specific.
  - **Random Model:** Allows API requests to a randomly selected model version in the group. The Routing-Option header in the API request must be set to random.
  - **All Models:** Allows requests to every enabled model version in the group. The Routing-Option header in the API request must be set to all.

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

- **App**: Generates a URL that runs the asset code as an interactive UI session. You can specify the Shared with setting to be either:
  - **Anyone with the link:** Anyone outside person can view it
  - **Any authenticated user:** Only signed users can view it
  - **Deployment Admin:** Only the creator of the project release can view it.

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

A web service generates a REST API endpoint for an external application to call.

The following asset types can be deployed as a web services:

- Models
- Scripts

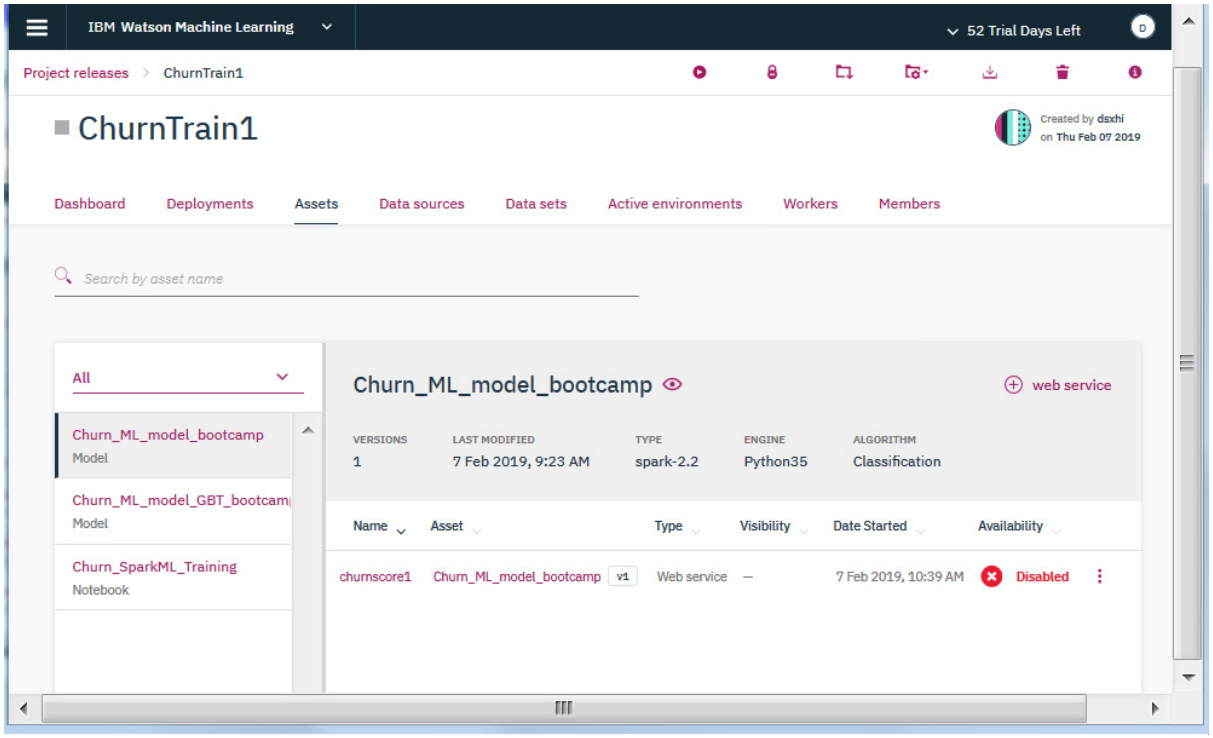
Restriction: You cannot deploy custom batch models as web service (must be custom online)

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

- To deploy an asset, complete the following steps:

1. In the Watson Machine Learning client, go to your project release and click the **Assets** tab.



© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets

2. Select a model from the left side of the assets table, and then, in the right side, click the button to create a web service. The Create web service deployment window opens.

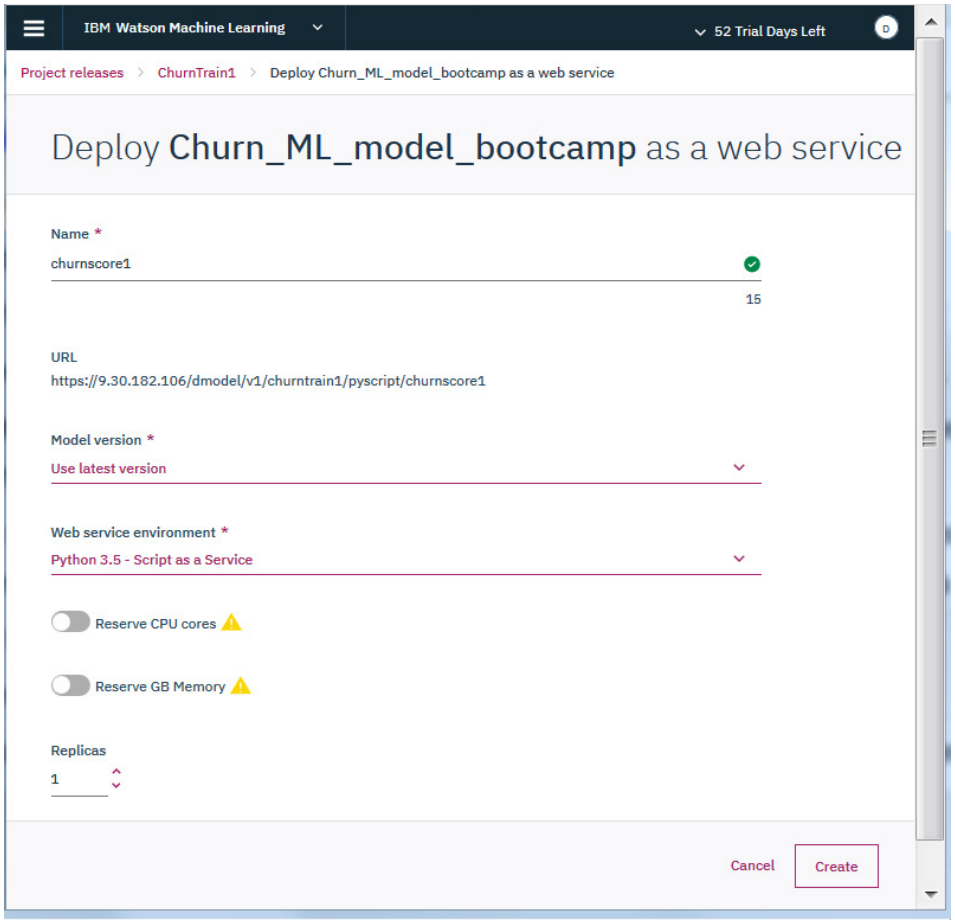
   You are prompted with the following fields:

   - Name – Specify a name for the asset that will be deployed. Use only lowercase alphanumeric characters and hyphens, up to 26 characters.
   - URL – View the generated URL for the web service.
   - Model version – Select from available versions. The latest is the default.
   - Web service environment – Select from one of the generated scripts, or select Custom for a custom script that you generate.
   - Reserve resources
   - Replicas

3. Click the **Create** button

© 2019 IBM Corporation

Model Management & Deployment – Deploy Assets



- All deployments remain **disabled** until the project release is launched
- If the project release is already online, your newly created deployment starts immediately

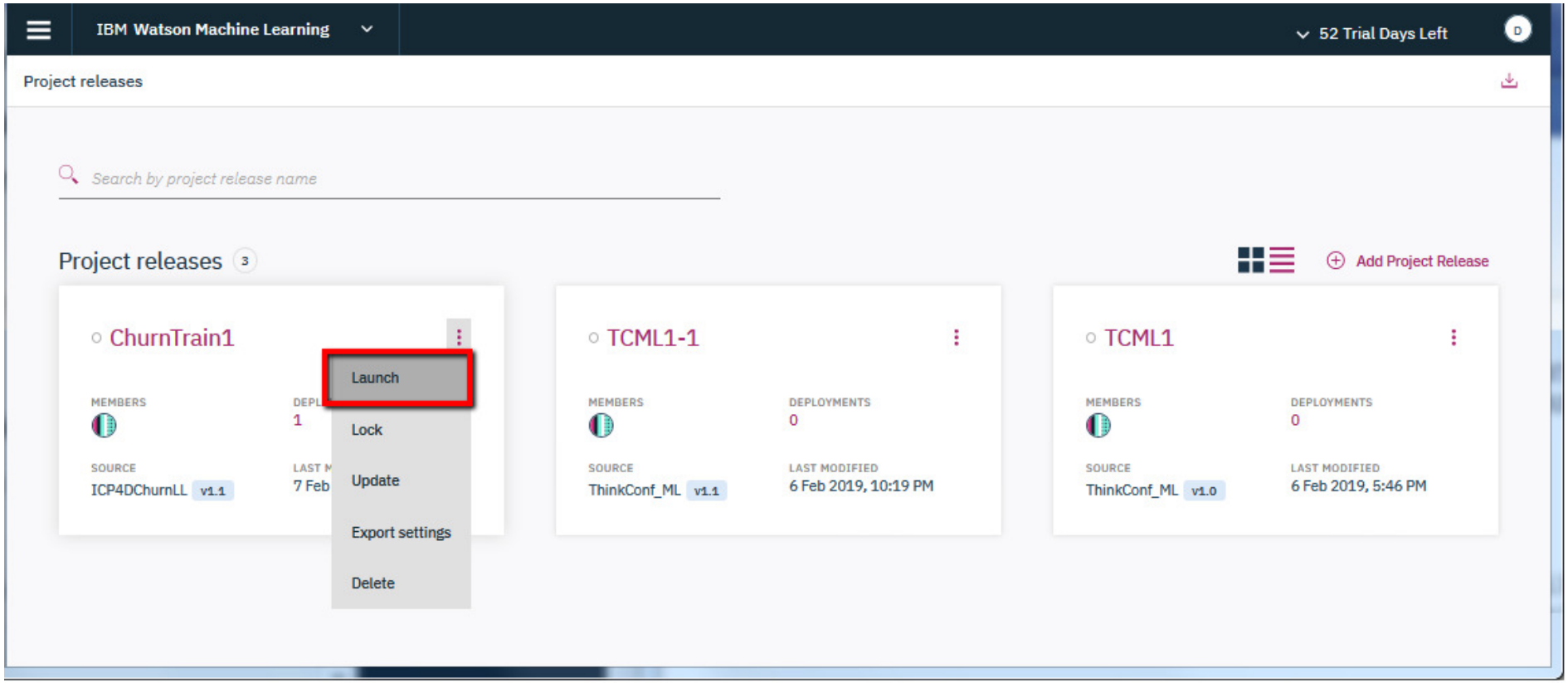
© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release

- An Admin can click the Launch icon to bring the release online
- Wait about a minute for the pods to start and each enabled deployment to activate (you can monitor the status in the Active environments panel) before you click any actions for the deployment
- You might also see a similar time delay when you stop or start a script for model deployment
- If you deployed a Job:
  - Click it to view the REST API endpoint for an external application or to run the job on demand
  - When the release is online, you can click the API tab and submit a request to Start, Stop, or get Status for the job
  - You can also click generate code to copy/paste the curl command with the deployment bearer token in it.
  - You cannot take a release offline; you can only delete the release

© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release



© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release

- If you deployed a Job:
  - A job deployment generates a REST API endpoint to start, stop, and get status for that job. The job can be scheduled (but not unscheduled), and is visible to authenticated users only.
  - The following asset types can be deployed as a job:
    - Scripts
    - Notebooks

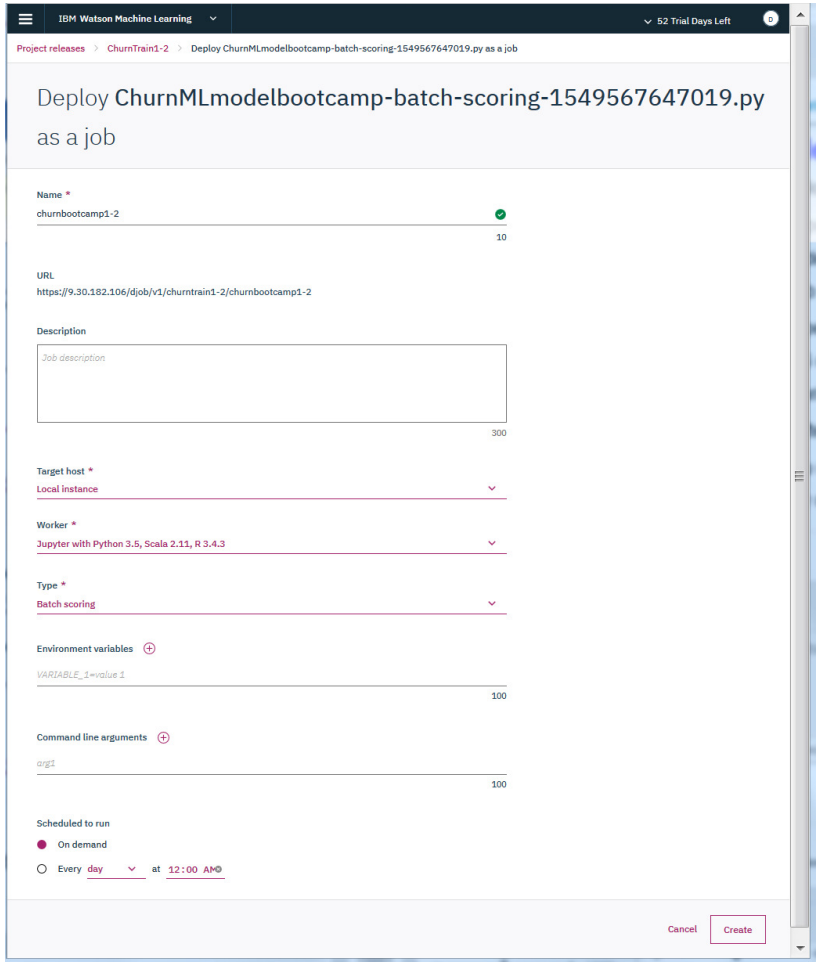
© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release

- Deploy a job following this steps:
  - Open the Assets tab in the Project release details page. Select a script or notebook from the left side of the assets table, and then, in the right side, click the button to create a job. The Create job deployment window opens.
  - Specify a name for the job that contains lowercase alphanumeric characters and hyphens, up to 26 characters. Then specify other details for the job, including workers, job type, environment variables, and command line arguments. You can set up a schedule for the job.
  - After you create a job, you will see the job deployment in the deployments list.
  - To use the test API page in the Deployment details page, you must first bring the release online by clicking the Launch button from the Project release details page.

© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release



- All deployments remain **disabled** until the project release is launched
- If the project release is already online, your newly created deployment starts immediately

© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release

- If you deployed a Web Service:
  - It will have a REST API endpoint for an external application to call
  - From the deployment details API tab, you can submit a JSON request or click generate code to copy/paste the curl command with the deployment bearer token in it
  - For an R script, ensure RStudio environment is in running state before you run the curl command (otherwise you might receive a '502 Bad Gateway' error)
  - You cannot take a release offline; you can only delete the release.

© 2019 IBM Corporation

Model Management & Deployment – Launch a Project Release

- If you deployed a Job:
  - Click it to view the REST API endpoint for an external application or to run the job on demand
  - When the release is online, you can click the API tab and submit a request to Start, Stop, or get Status for the job
  - You can also click generate code to copy/paste the curl command with the deployment bearer token in it.
  - You cannot take a release offline; you can only delete the release

© 2019 IBM Corporation

Model Management & Deployment – Manage a Project Release

- **Update:**
  - To pull in the latest assets from the source project and update the deployments with them, click **Update** next to the release
  - This action only updates modified assets and adds new assets to the release. It does not delete assets from the release
- **Export:**
  - To export the deployment settings of a release (for use in another release), click **Export deployment settings** next to it
  - You can then upload these same deployment setting into another offline release by clicking the Import deployment settings icon and dragging the settings file into the Input pane.
  - Click the **Validate File** button to compare these imported settings with the current settings for your release
  - If the **Differences** look acceptable, you can click **Submit** to update your release manifest accordingly. Existing settings will be updated, new settings will be added, and unaffected settings will remain unchanged

© 2019 IBM Corporation

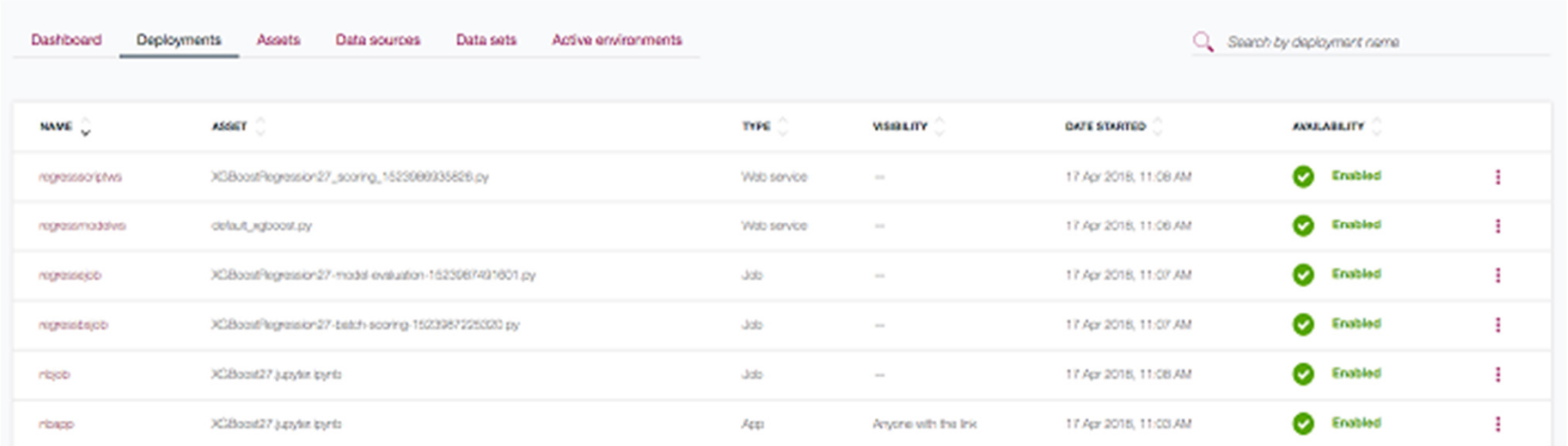
Model Management & Deployment – Manage a Project Release

- **Lock**:
  - To lock a release so that no one can edit it, click **Lock** next to it
  - You can click **Unlock** to reopen the release for editing.
- **Delete**:
  - To delete a release and all of its deployments, click **Delete** next to it
  - Alternatively, you can click the **Delete** icon from the release details page.

© 2019 IBM Corporation

Model Management & Deployment – Manage Deployments

- From the Deployments tab of your project release, an Admin, Developer, or Viewer can click a deployment for an overview of its details including request metrics captured at the function level
- Only the Admin can start, stop, delete, or redeploy any deployment in the release. Note that if you redeploy a new version of any of the assets, the URL does not change
- For an app, the Admin can edit its settings, edit is visibility, and view its route



© 2019 IBM Corporation

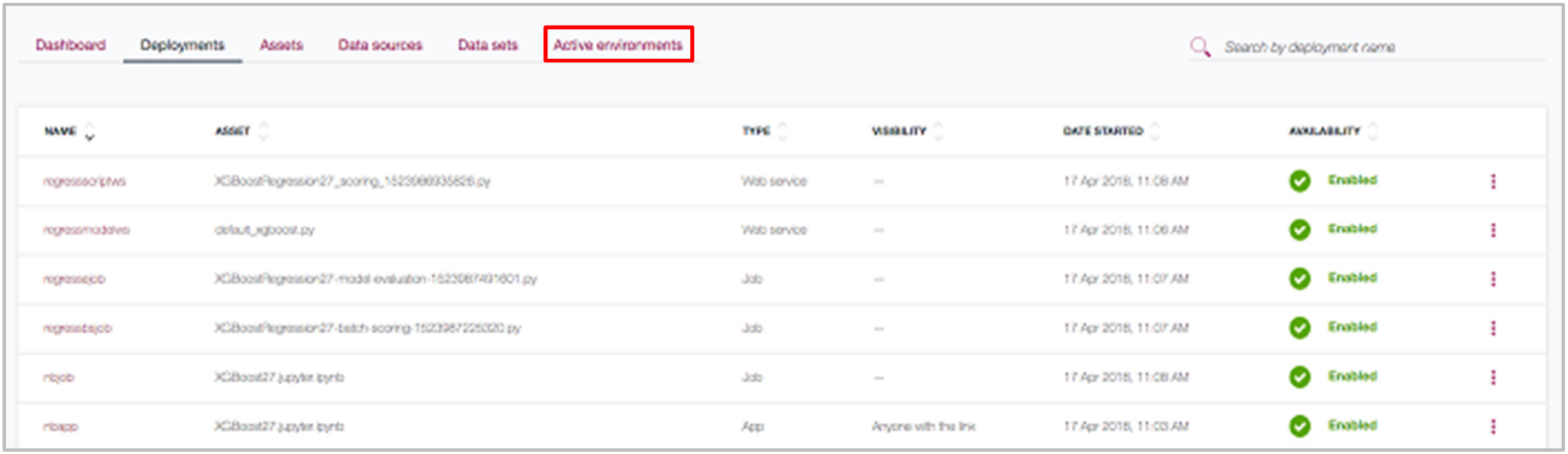
Model Management & Deployment – View Data Sources

- In the IBM Deployment Manager, click the Data sources to view your project data sources
- You can also edit the production data source connection string and credentials by clicking Edit credentials next to the data source
- Important: Database credentials must be input again after creating a project release. Otherwise, the data source might not work

© 2019 IBM Corporation

Model Management & Deployment – View Active Environments

- In the Watson Machine Learning client, click the **Active environments** tab to monitor all runtime environments, workers, and services to WSL
  - This is synonymous with using the menu icon to click on **All Active Environments**



© 2019 IBM Corporation

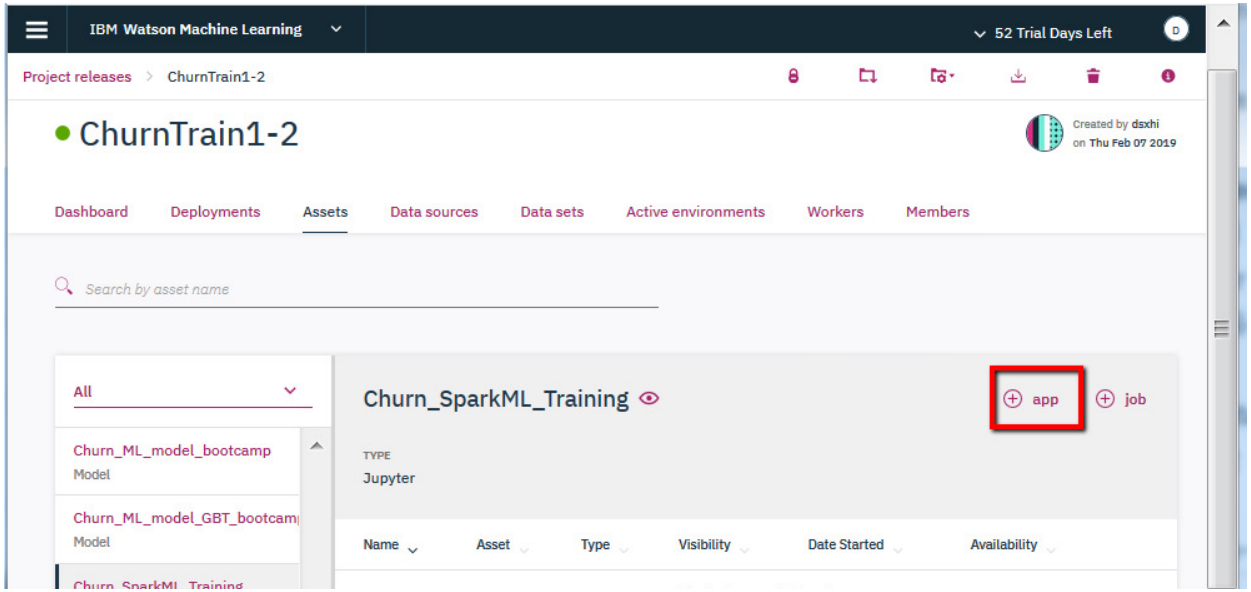
Model Management & Deployment – Deploy Asset as an App

- An app deployment generates a URL that runs the asset code as an interactive UI session
- The following asset types can be deployed as an app:
  - Shiny Servers
  - Jupyter Notebooks

© 2019 IBM Corporation

Model Management & Deployment – Deploy Notebook as an App

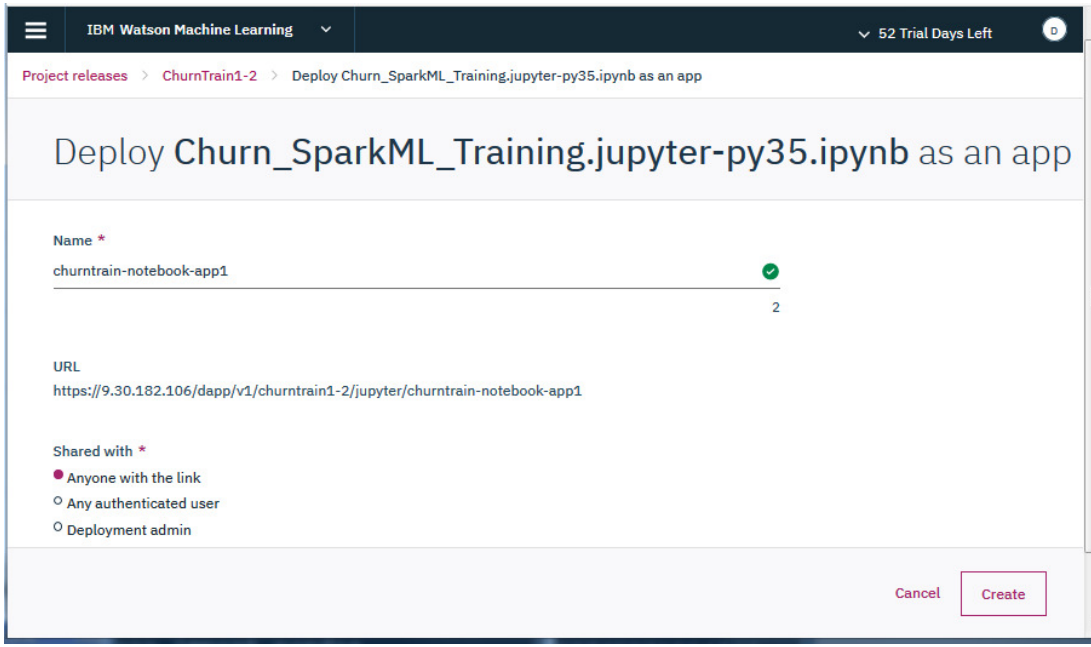
- **Requirement**: Ensure that you run your notebook before you deploy it as an app. Otherwise, cell results such as graphs do not appear
- You can navigate to the Assets tab in the Project release details page
- When a Jupyter notebook is selected in the left panel of the assets table, the right panel of the assets table provides the button to create an app, which brings you to the Create app deployment screen.



© 2019 IBM Corporation

Model Management & Deployment – Deploy Notebook as an App

- You are prompted with the following fields:
  - A **Name** for the asset to be deployed. It can contain lowercase alphanumeric characters and hyphens, up to 26 characters
  - A generated **URL** is shown
  - You can select the **Visibility** option from the radio group. Visibility determines who is app to interact with the app after it is deployed



© 2019 IBM Corporation

Model Management & Deployment – View App

- When an app deployment is created and the project is live, you can view the app by going to the Deployments tab on the Release details page
- You can click the name of the app deployment and are brought to the app page.

© 2019 IBM Corporation

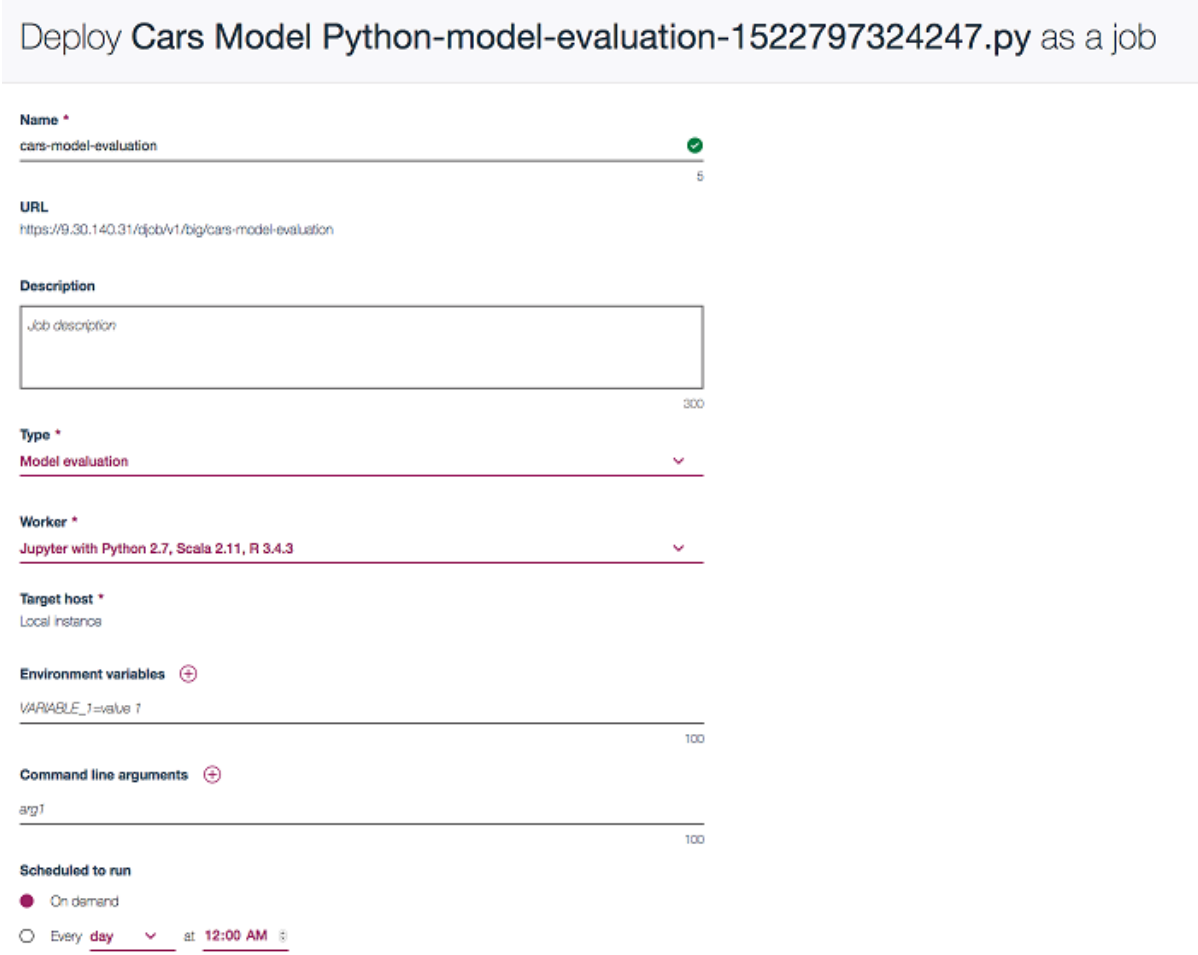
Model Management & Deployment – Deploy Asset as a Job

- A job generates a REST API endpoint to start, stop, and get status for that job
- The job can be scheduled (but not unscheduled), and is visible to authenticated users only
- The following asset types can be deployed as a job:
  - Scripts
  - Notebooks

© 2019 IBM Corporation

Model Management & Deployment – Deploy Script or Notebook as a Job

- You can navigate to the Assets tab in the Project release details page
- When a script is selected in the left panel of the assets table, the right panel of the assets table provides the button to create a job, which brings you to the Create job deployment page.



Model Management & Deployment – Deploy Script or Notebook as a Job

- You are prompted with the following fields:
  - A **Name** for the job to be deployed. It might contain lowercase alphanumeric characters and hyphens, up to 26 characters
  - A generated **URL** is shown
  - An optional **description** is shown
  - You are able to select the **Worker** from the pull-down of available images
  - You are able to select the **job type** from the pull-down for the various job types
  - You have the option to provide **environment variables** for the job and **command line arguments**
  - You have the option of providing a time for the job to be **scheduled to run**

© 2019 IBM Corporation

Model Management & Deployment – Deploy Script or Notebook as a Job

- After you create a job, you will see the job deployment in the deployments list
- To use the test API page in the deployment details page, you must first bring the release online by clicking the Launch button in the upper right of the Project release details page

© 2019 IBM Corporation

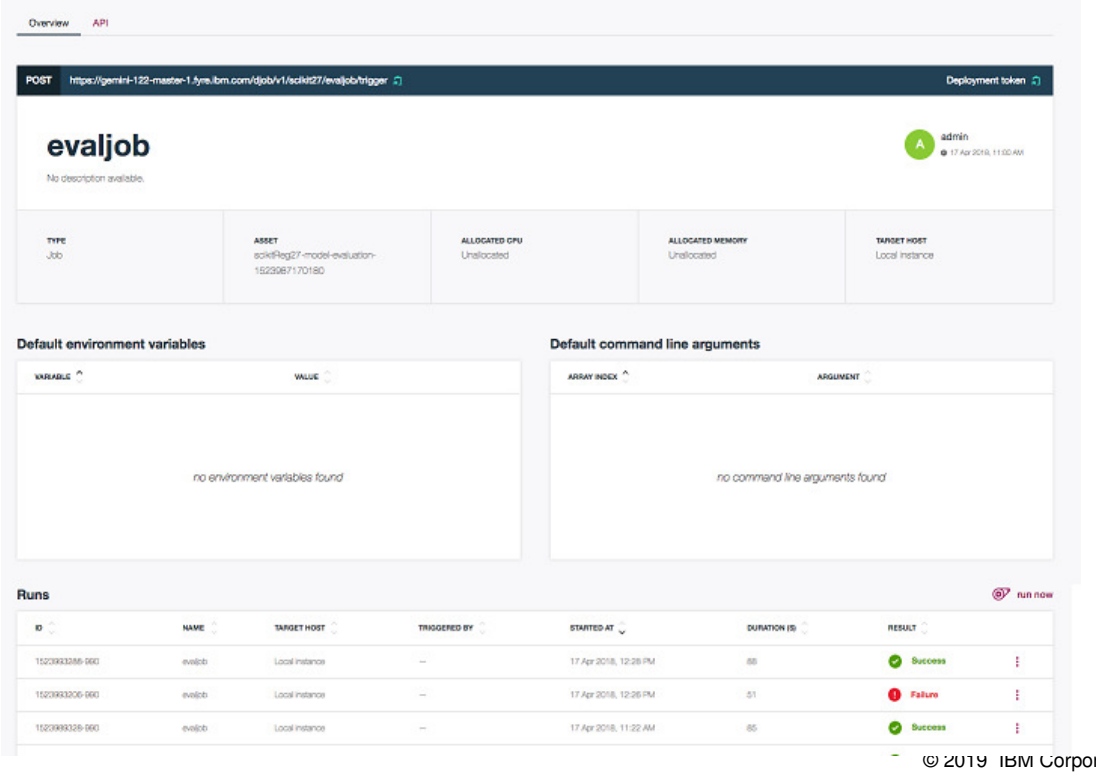
Model Management & Deployment – Manage a Deployed Job

- After you create a job deployment and selecting a job deployment from the deployments page, Watson Studio Local displays a Deployment details page
- The Deployment details page has two tabs:
  - Overview
  - API
- **Important**: You cannot start, stop, or get the status of a job if the release is not live

© 2019 IBM Corporation

Model Management & Deployment – Overview Tab

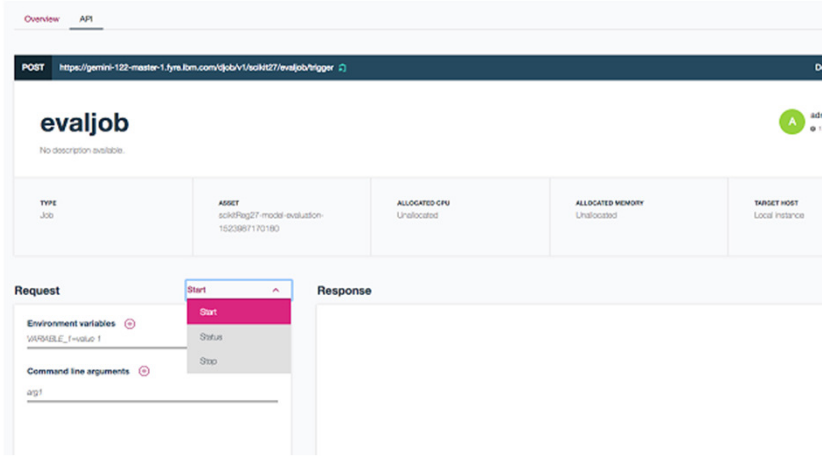
- A table of environments and arguments for the current job
- A table of all the runs for that particular job
- A button run now that triggers a job run



© 2019 IBM Corporation

Model Management & Deployment – API Tab

- In the API tab, you have three options of API calls to interact with a job with the pull-down menu:
  - Start calls the **trigger** api call to start a job run
  - Stop calls the **cancel** api call to stop a job run
  - Status calls the **status** api call to get information about the status. Note if you do not supply a run ID in the text box when you call the status, the API call gets all the runs for the current job
  - A **generate code** button, which displays the proper curl command based on the selected status and the data in the **input** box
  - A result box that shows the **JSON** output from the API call



© 2019 IBM Corporation

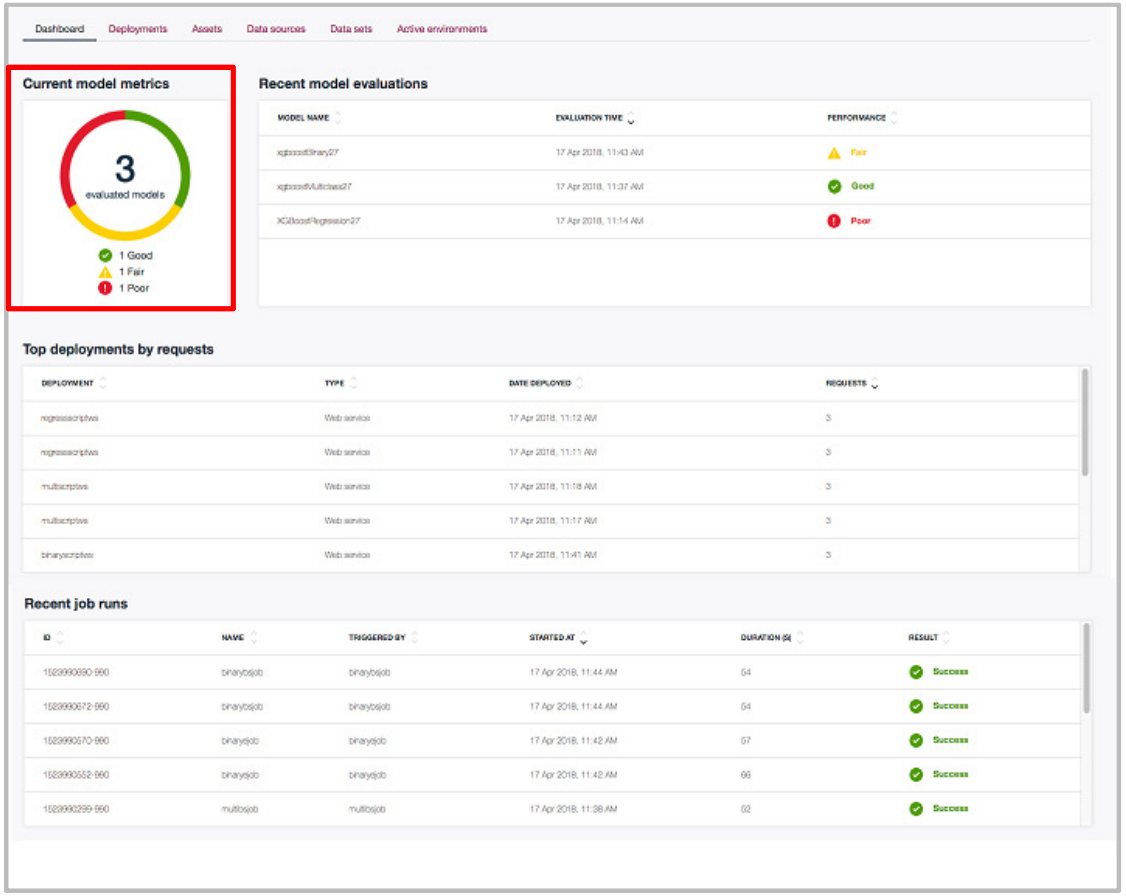
Model Management & Deployment – Project Release Dashboard

- Deployment dashboard provides an overview about the release
- It has five components:
  - Current Metrics
  - Recent Model Evaluations
  - Top Deployments by Requests
  - Recent Job Runs

© 2019 IBM Corporation

Model Management & Deployment – Current Model Metrics

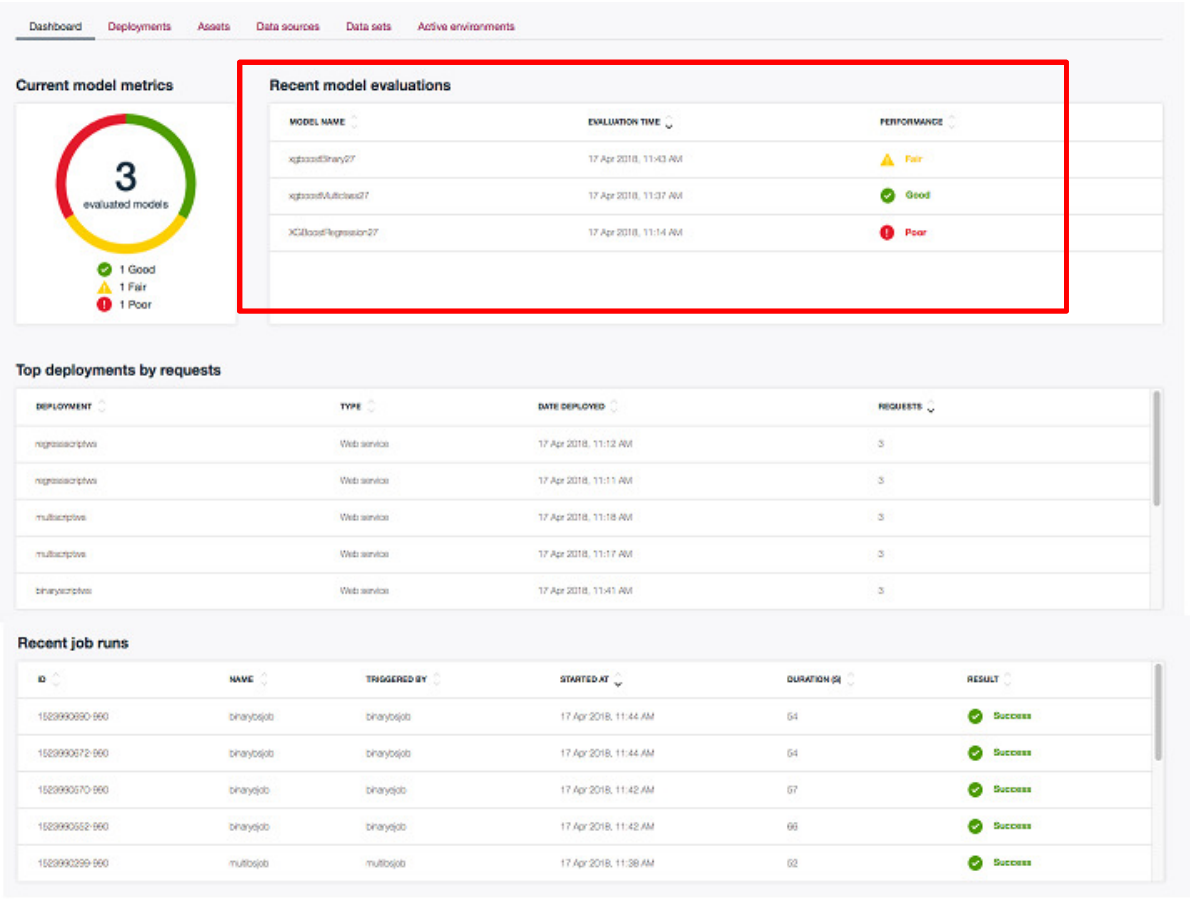
- The performance of all models across the release. The model's performance is calculated on model's score and thresholds. There are three different types of performance:
  - Good
    - Green
  - Fair
    - Yellow
  - Poor
    - Red



© 2019 IBM Corporation

Model Management & Deployment – Recent Model Evaluations

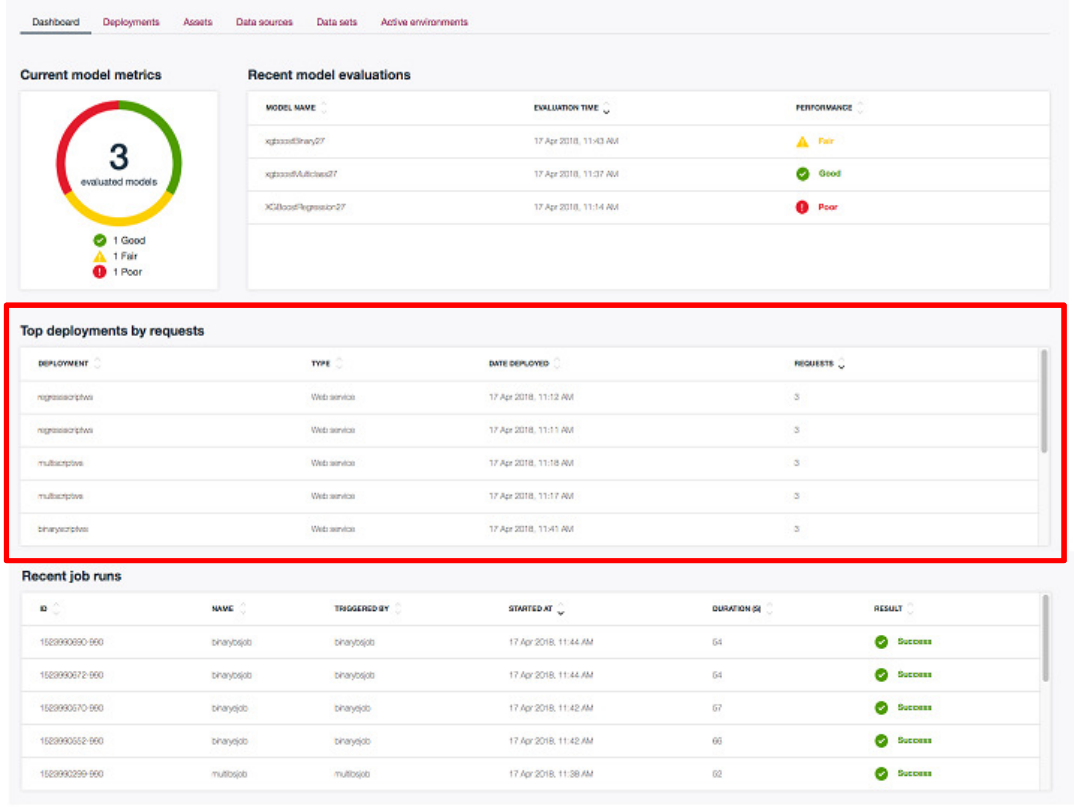
- This component shows five latest model's evaluations



© 2019 IBM Corporation

Model Management & Deployment – Top Deployments by Requests

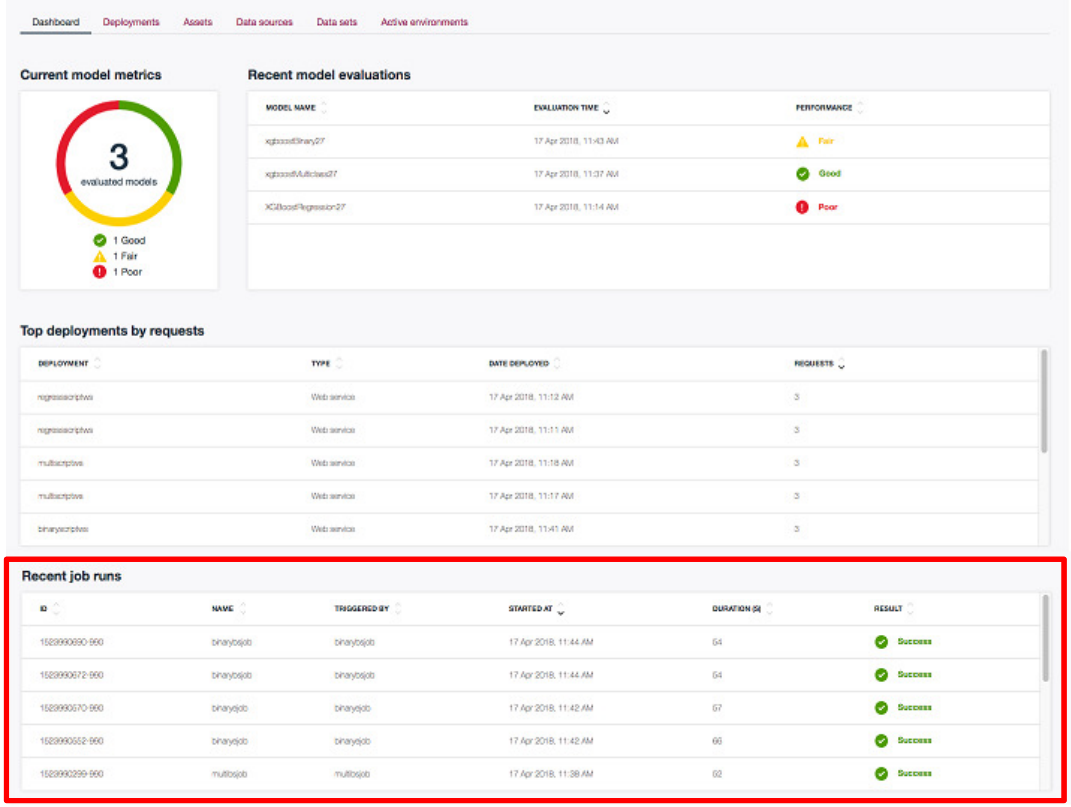
- This component shows five deployments that have the most number of requests
- There are three different types of deployments:
  - Job
  - Web Service
  - App



© 2019 IBM Corporation

Model Management & Deployment – Recent Job Runs

- This component shows five most recent job runs

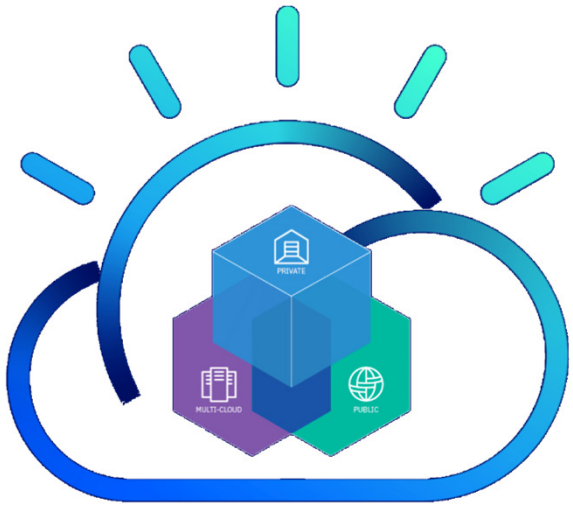


© 2019 IBM Corporation

Watson Studio Local

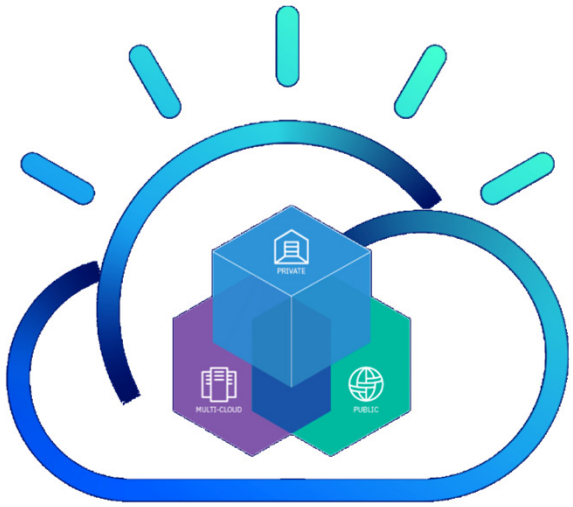
Administration

Additional Topics



© 2019 IBM Corporation

Watson Studio Local
Administration
Kubernetes 101



© 2019 IBM Corporation

Kubernetes 101

- What is Kubernetes?
  - Kubernetes is an open source orchestration tool developed by Google for managing microservices and containerized applications across a distributed cluster.
  - Kubernetes provides highly resilient infrastructure with zero downtime deployment capabilities, automatic rollback, scaling, and self-healing of containers.
  - The main objective of Kubernetes is to hide the complexity of managing a fleet of containers by providing REST APIs for the required functionalities.
  - Kubernetes is portable in nature, meaning it can run on various public or private cloud platforms such as AWS, Azure, OpenStack, Google Cloud, or Apache Mesos. It can also run on bare metal machines.

FYI. Kubernetes is a part of Cloud Native Computing Foundation (CNCF) project hosted by The Linux Foundation. https://www.cncf.io

© 2019 IBM Corporation

Kubernetes 101

- What is K8s?
  - Developers are lazy and somewhere in the mid-late 80s they started abbreviating the words based on their first letter, last letter, and number of letters in between.
  - For example **i18n** is equivalent to **internationalization** and **l10n** for **localization.**
  - So K8s is Kubernetes. ☺

FYI. Kubernetes is an open source project written in Go language, and licensed under the Apache License Version 2.0. The current stable version is 1.9 (as of February 2018)

© 2019 IBM Corporation

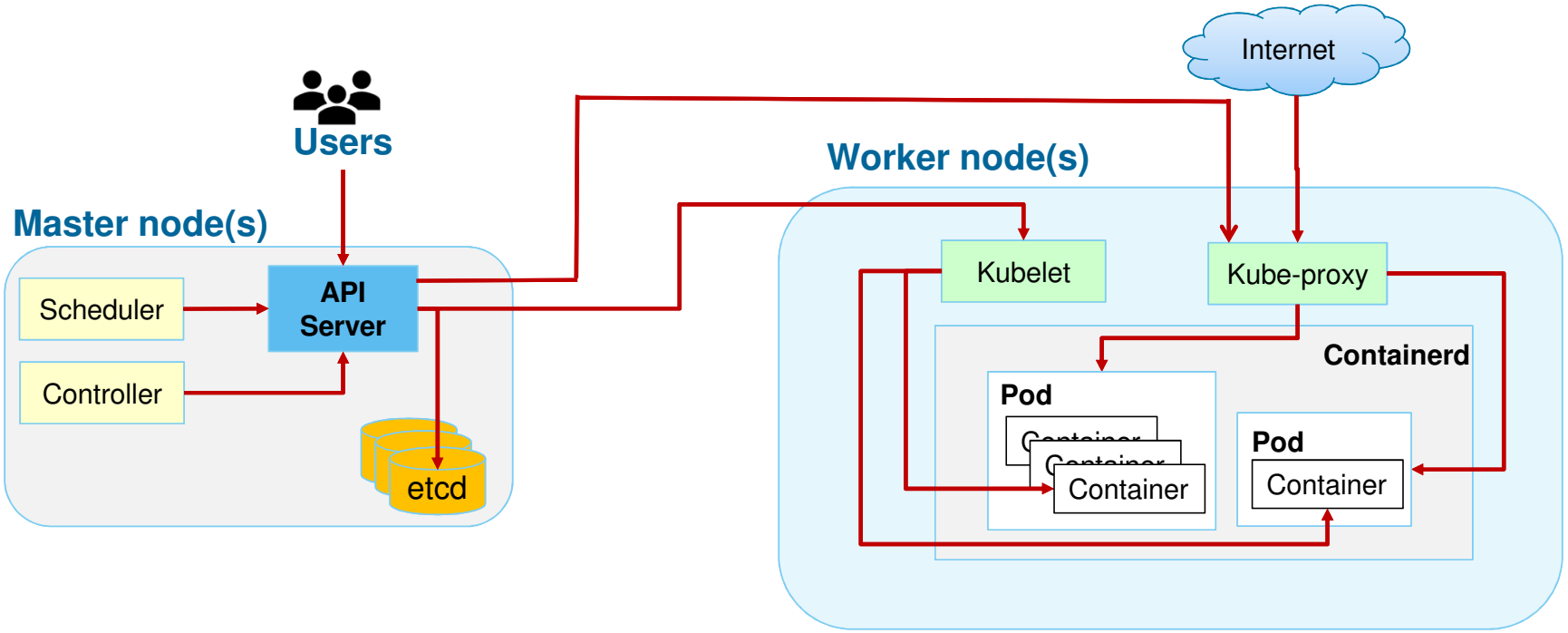
Kubernetes 101

- Most popular container orchestrators nowadays
  - Docker Swarm – provided by Docker, Inc. www.docker.com
  - Kubernetes – Cloud Native Computing Foundation. https://www.cncf.io/
  - Mesos Marathon – Apache Mesos. http://mesos.apache.org/
  - Amazon ECS – Amazon EC2 Container Service https://aws.amazon.com/ecs/
  - Hashicorp Nomad – provided by HashiCorp. https://www.hashicorp.com/

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - Kubernetes follows a client-server architecture. You can have multi-master setup (for HA).



© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **Master node**
    - The master node is responsible for the management of Kubernetes cluster.
    - The master node is the one taking care of orchestrating the worker nodes, where the actual services are running
    - We can communicate to the master node via the CLI, the GUI (Dashboard), or via APIs.
    - Only one of the master nodes would be the leader, performing all the operations. The others are followers.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **Worker node**

    A worker node is a machine which runs the applications using Pods and is controlled by the master node. Pods are scheduled on the worker nodes, which have the necessary tools. It's a logical collection of one or more containers which are always scheduled together.
  - Docker
    - Docker runtime (ie. Containerd) runs on each of the worker nodes, and runs the configured pods. It handles downloading the images and starting the containers
  - Kubelet
    - Kubelet is an agent which runs on each worker node and communicates with the master node. It gets the configuration of a pod from the API server and ensures that the described containers are up and running.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - Kube-proxy
    - Kube-proxy acts as a network proxy and a load balancer for a service on a single worker node. It takes care of the network routing for TCP and UDP packets.
    - It listens to the API server for each Service endpoint creation/deletion. It setup the routes so that it can reach to it.
  - Kubectl
    - A command line tool to communicate with the API service and send commands to the master node.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **API server**
    - The API server is the entry point for all the REST commands used to control the cluster. It processes the REST requests, validates them, and executes the bound business logic.
  - **Scheduler**
    - The scheduler schedules the work to worker nodes.
    - The scheduler has the information regarding resources available on the members of the cluster, as well as the ones required for the configured service to run and hence is able to decide where to deploy a specific service.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **Controller-manager**
    - A controller uses API server to watch the shared state of the cluster and make corrective changes to the current state to change it to the desired one.
    - You may have multiple different kind of controllers, for example, the replication controller which takes care of number of pods in the system. Other examples like, endpoints controller, namespace controller, and service accounts controller, etc.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **Pod**
    - Kubernetes targets the management of elastic applications that consist of multiple microservices communicating with each other. Often those microservices are tightly coupled forming a group of containers that would typically, in a non-containerized setup run together on one server. This group, the smallest unit that can be scheduled to be deployed through K8s is called a pod.
    - This group of containers share storage, Linux namespaces, IP addresses. Containers reach each other via localhost.
    - Pods are not intended to live long. They are created, destroyed and re-created on demand, based on the state of the server and the service.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **Service**
    - As pods have a short lifetime, there is not guarantee about the IP address they are served on. This could make the communication of microservices difficult.
    - K8s has introduced the concept of a service, which is an abstraction on top of a number of pods, communicating with it via a Virtual IP address.
    - This is where you can configure load balancing for your numerous pods and expose them via a service.

© 2019 IBM Corporation

Kubernetes 101

- Kubernetes Components and Architecture
  - **etcd storage**
    - etcd is a simple, distributed, consistent key-value store. It's mainly used for shared configuration and service discovery.
    - It provides a REST API for CRUD operations as well as an interface to register watchers on specific nodes, which enables a reliable way to notify the rest of the cluster about configuration changes.
    - An example of data stored by Kubernetes in etcd is jobs being scheduled, created and deployed, pod/service details and state, namespaces and replication information, etc.

© 2019 IBM Corporation

Kubernetes 101

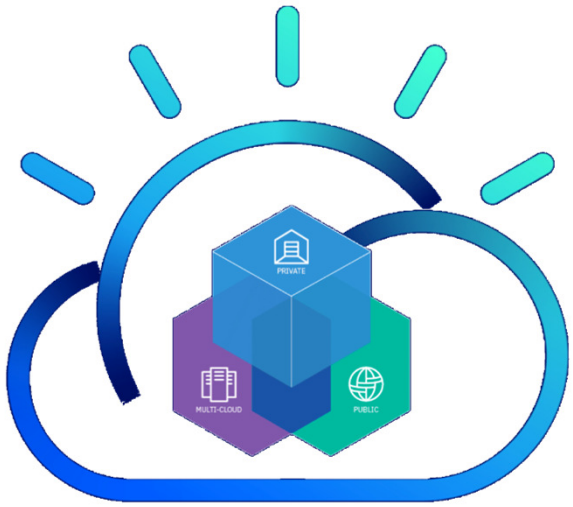
- Kubernetes Components and Architecture
  - **GlusterFS**
    - Gluster is a scalable, distributed file system that aggregates disk storage resources from multiple servers into a single global namespace.
    - Advantages
      - Scales to several petabytes
      - Handles thousands of clients
      - POSIX compatible
      - Uses commodity hardware
      - Can use any ondisk filesystem that supports extended attributes
      - Accessible using industry standard protocols like NFS and SMB
      - Provides replication, quotas, geo-replication, snapshots and bitrot detection
      - Allows optimization for different workloads
      - Open source

© 2019 IBM Corporation

Watson Studio Local

Administration

Best Practices



© 2019 IBM Corporation

Best Practices

- **Access large amount of data – with relational databases**

  When you load gigabytes of data from tables, no amount of memory you reserve will be adequate. For better practice, consider following

  – You can load and process by batches. Instead of fetchall() use batch similar to the following example:

```
cur.execute('select * from table1')
for row in cur:
    print row
    #perform processing…
```


© 2019 IBM Corporation

Best Practices

- **Access large amount of data – with relational databases**
  - Or add criteria in you query filter. Example

```
cur = conn.cursor()
cur.execute("SELECT * FROM large_table WHERE geo = 'NA'")
```

  - Using SparkSQL and DataFrame to provide faster data access compared to other database JDBC clients.
  - Consider letting RDBMS process data as much as possible, and extract only the subset.
  - Define stored procedures that can perform most of the filtering and processing on your relational database.

© 2019 IBM Corporation

Best Practices

- **Access large amount of data – with Spark and Hadoop**
  - As a best practice, processing should be close to your data. If you need to use large amount of data in HDFS or Hive, It's best to use the Spark running on Hadoop cluster through Livy.
  - Take advantage of data localization, and avoids data transfer to speed up your processing.
  - All runtime environments in WSL (Jupyter, Zeppelin and RStudio) support accessing Hadoop Spark through Livy. More details refer to: remote spark documentation

© 2019 IBM Corporation

Best Practices

- **Performance Tuning**
  - Generally, WSL reads and writes to relational databases using JDBC and modules JayDebeApi and RJDBC/SparkR.
  - Using database vendor specific clients such as cx_Oracle and pymssql can provide comparable performance to SparkSQL.
  - WSL writes database specific data frames using SQLAlchemy and database vendor specific adapters such as ibm_db_sa.
  - WSL does NOT support ODBC as the moment.

© 2019 IBM Corporation

Best Practices

- **Performance Tuning – Tune Zeppelin notebook**

  Set Zeppelin interpreter property and increase memory

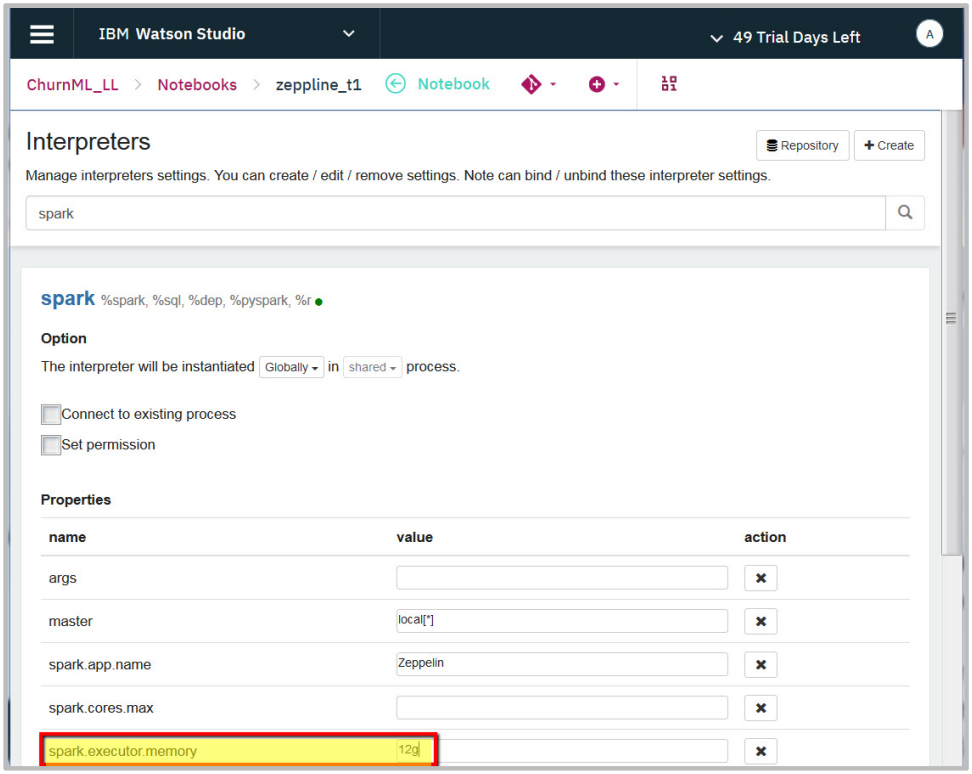
  - Go to the Zeppelin interpreters by clicking on the wheel at the right top of the Zeppelin notebook window.
  - Click on the Interpreter link to open the Interpreter settings window.
  - Search for spark and select edit on the upper right.
  - Set spark.executor.memory to 12g for example. You might have to try increasing this as much possible to test.

© 2019 IBM Corporation

Best Practices

- **Performance Tuning – Tune Zeppelin notebook**

  Set Zeppelin interpreter property and increase memory



© 2019 IBM Corporation

Best Practices

- **Performance Tuning** – **Tune Spark applications**
  - **Spark executor configuration** – **Change the resources for the Spark application**
    - Stop the pre-created spark context, and then create a new spark context with the proper resource configuration. For example:

```
sc.stop()
from pyspark import SparkConf, SparkContext
conf = (SparkConf()
    .set("spark.cores.max", "15")
    .set("spark.dynamicAllocation.initialExecutors", "3")
    .set("spark.executor.cores", "5")
    .set("spark.executor.memory", "6g"))
sc=SparkContext(conf=conf)
```


© 2019 IBM Corporation

Best Practices

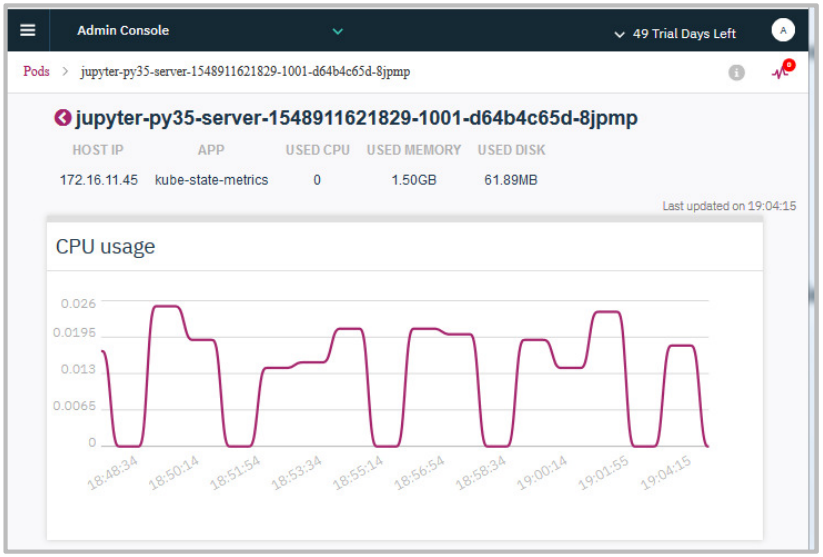
- **Performance Tuning – Tune Spark applications**
  - **Spark executor configuration – Change the resources for the Spark application**
    - Verify the new settings by running the following command in a cell using the new Spark context

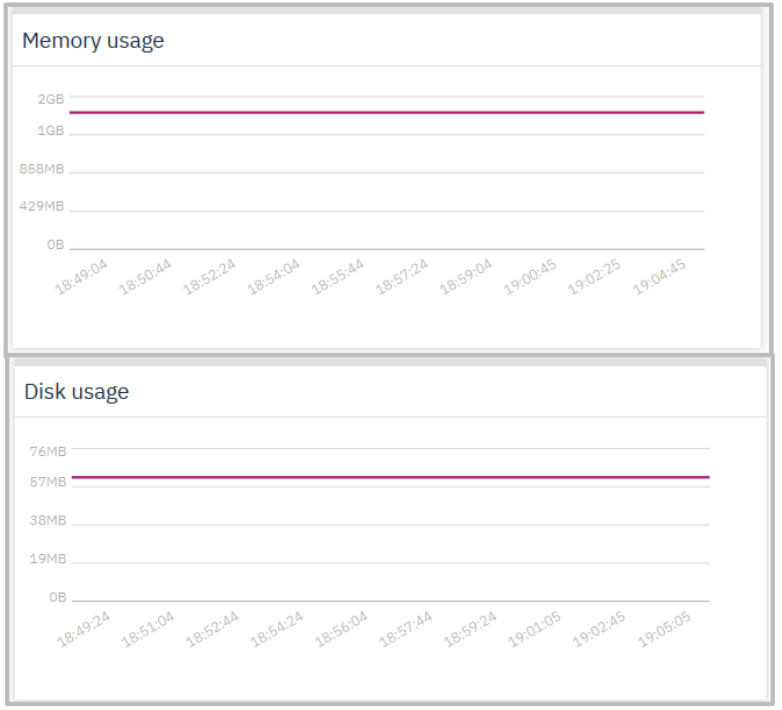
```
for item in sorted(sc._conf.getAll()): print(item)
```


© 2019 IBM Corporation

Best Practices

- **Performance Tuning – Tune Spark applications**
  - **Monitoring**
    - You can monitor the per-pod CPU/memory/disk usage from the dashboard in the **Admin Console**.





© 2019 IBM Corporation

Labs

1. **Lab #1** Instruction: WatsonStudio_Deployment.pdf
2. **Lab #2**: Use the same notebook developed in Lab #1 and convert the local Spark to remote Spark push down jobs.

© 2019 IBM Corporation