Deploying Oracle Database with IBM Terraform & Service Automation on Red Hat Openshift Container Platform and IBM PowerVC

March, 31 2021

Ralf Schmidt-Dannert

IBM Advanced Technology Group, ISV on Power - Oracle

This page intentionally left blank.

Table of Contents

1 Abstract

This paper builds on the Proof of Technology described in "Deploying Oracle® Database as a Service with IBM Cloud PowerVC Manager 1.3.1" (see References, page 93) where we demonstrate how IBM Power Virtualization Center (PowerVC) technology enables the implementation of Database-as-A-Service (DBaaS) for an Oracle® database.

In the first section of this paper, we demonstrate how to utilize IBM Terraform & Service Automation (T&SA), instead of (PowerVC), to provide the control point and end-user interface for DBaaS for an Oracle database, while re-using the existing PowerVC image developed in the earlier project.

We then expand the scope and illustrate the workload orchestration capabilities of T&SA by creating a deployable service which co-deploys two virtual machines (VMs) - a database server and an application server. The result of that orchestrated deployment is a running and pre-loaded Oracle database and a workload driver in the second VM ready for the end-user to connect to and drive simulated transactions against the database.

It is assumed that readers of this paper have at least some knowledge of Red Hat Openshift Container Platform 3.11 (OCP), IBM Terraform & Service Automation v4.2 (T&SA), IBM Power servers, IBM Power Virtualization Center (PowerVC) as well as AIX and Linux system administration. The creation of the deployable image for the Oracle database server, as described in the sister white paper, requires working knowledge of Oracle Database 12c software.

This paper is an updated version of the white paper "Deploying Oracle Database with IBM Cloud Automation Manager and IBM PowerVC" (https://www.ibm.com/support/pages/node/6355775) authored by Stephen Poon and Ralf Schmidt-Dannert and which is based on a deployment of IBM Cloud Automation Manager on top of IBM Cloud Private instead of Red Hat Openshift and IBM Cloud Pak for Multicloud Management.

2 Legal Disclaimer

No warranty:

Subject to any statutory warranties which cannot be excluded, IBM makes no warranties or conditions either express or implied, including without limitation, the warranty of non-infringement and the implied warranties of merchantability and fitness for a particular purpose, regarding the program or technical support, if any.

Limitation of Liability:

Neither IBM nor its suppliers will be liable for any direct or indirect damages, including without limitation, lost profits, lost savings, or any incidental, special, or other economic consequential damages, even if IBM is informed of their possibility. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above exclusion or limitation may not apply to you.

3 Comments / Feedback

Version: 1.0

The author is interested in any feedback or comments on this white paper and the approach taken to implement a service on IBM Terraform & Service Automation.

Contact information:
Ralf Schmidt-Dannert, IBM
dannert@us.ibm.com

4 Introduction

Application containerization has become a major trend in software development and deployment. A 2020 survey conducted by Forrester (*Forrester Analytics Business Technographics® Infrastructure Survey, 2020*) shows that 24% of customers in the study are already using containers today and that those customers expect to increase that usage to 50% in the next two or more years.

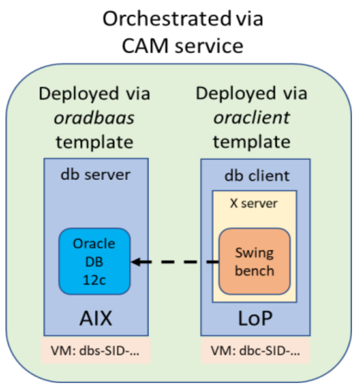
In support of this new development and deployment paradigm, Red Hat has created the Red Hat Openshift Container Platform (OCP) offering which is based on Kubernetes. Kubernetes is an open-source container-orchestration system for automating application deployment, scaling, and management. It was originally designed by Google and is now maintained by the Cloud Native Computing Foundation.

In 2019 IBM bought Red Hat®, Inc. and made the decision that Red Hat OpenShift will be the platform of choice to develop and deploy containerized applications going forward.

This white paper utilizes IBM Terraform & Service Automation 4.2 (T&SA) deployed into IBM Cloud Pak for Multicloud Management 1.2 (CP4MCM) in OCP 3.11 environment on IBM Power servers. The latest IBM Cloud Pak for Multicloud Management version includes T&SA 4.2. We used fix pack 4.2.0.1 of T&SA.

While more and more containerized applications are being developed and deployed, many critical applications today are still virtual machine (VM) based. For mixed environments, with both VM and containerized applications, IBM Terraform & Service Automation (T&SA) is a Docker container application that allows you to deploy existing VM-based applications in a multi-cloud environment. Since T&SA is an application that runs on Red Hat OpenShift as part of the IBM Multicloud Management Cloud Pak you can integrate management of new container-based services with existing mission critical workloads and provide a single software and services catalog for both containerized and VM-based applications.

The purpose of this paper is to illustrate how T&SA can be utilized to orchestrate the deployment of two VMs from within CP4MCM. All relevant configuration and setup steps are described in detail so the reader can reproduce the deployment and implementation steps in his own environment.
The graphic below illustrates the end state after a successful service deployment from T&SA.


Orchestrated via CAM service
Deployed via *oradbaas* template
Deployed via *oraclient* template
db server
db client
X server
Oracle DB 12c
Swing bench
AIX
LoP
VM: dbs-SID-...
VM: dbc-SID-...

The deployed environment consists of:

- A database server (db server) running Oracle Database 12c on AIX, using the image described in the IBM Whitepaper entitled "Deploying Oracle Database as a Service with IBM Cloud PowerVC Manager 1.3.1" (DBaaS whitepaper, see References, Section 11) and,
- A database client (db client) consisting of a Linux on Power (LoP) VM running the Swingbench load generator over Oracle Instant Client against the database server.

After deployment, workload data is automatically loaded into the db server's database from the db client. Once the load is complete, the user/requester can access the db client to run Swingbench load generator.
The only required input from the end-user, and even that is optional, to deploy this full functionality via a "single-click" service deployment is to provide the database name to be configured.

Note: The Swingbench tool was developed by Dominic Giles, Oracle and is freely available at:
http://www.dominicgiles.com/swingbench.html

The following are pre-requisites for this proof of technology and the setup or configuration therefore are not covered in this white paper.

- IBM® PowerVC for Private Cloud (PowerVC) has been installed and configured and can successfully manage, capture, deploy, start, stop, etc. images and virtual machines onto managed IBM Power servers and storage.
- A virtual machine image with Linux on Power as OS has been created and is deployable in PowerVC.
- A virtual machine image with AIX and Oracle database software has been created and is deployable in PowerVC. See "Deploying Oracle Database as a Service with IBM Cloud PowerVC Manager 1.3.1" (see References, page 93) for details on how to create this image. Please note that the relevant scripts / documentation are also available in a GitHub repository here: https://github.com/ppc64le/devops-automation/tree/master/terraform/oracle/powervc-oradbaas
- An NFS server with Oracle client 12c software and the swingbench install zip-file accessible during deployment of the *oraclient* template.
- Red Hat Openshift Container Platform 3.11 and IBM Cloud Pak for Multicloud Management 1.3.1 with IBM Terraform & Service Automation (T&SA) have been installed. Please see Installing Cloud Automation Manager Community Edition reference for the installation of T&SA 4.2 into CP4MCM 1.3.1 as utilized in this proof of technology.
- A GitHub repository to be used to store the T&SA designer templates.

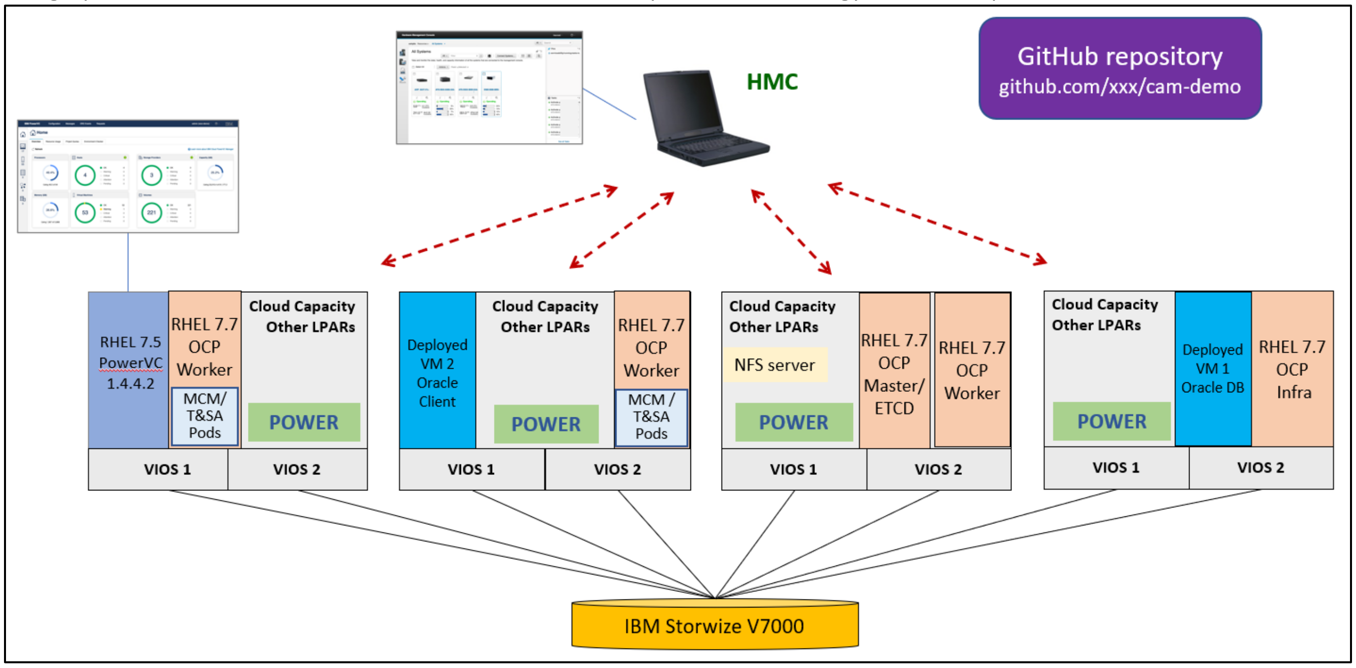
In the remainder of this white paper we provide:

- A description of the hardware / software environment in which this project was developed,
- A brief overview of Red Hat Openshift and IBM Terraform & Service Automation,
- A detailed description of how to create the Oracle DBaaS (*oradbaas*) template with the IBM Template Designer, part of T&SA software package, and deploy this template via IBM Terraform & Service Automation.
- A description on how to create the Oracle client (*oraclient*) template the same way, but not in as much detail. Template file content is available in the appendix.
- A detailed description of how to create and deploy a service, *Oracle 12C on AIX with Swingbench Client on LoP*, in T&SA using the two new templates *oradbaas* and *oraclient*.

Note that all related documentation and source code / scripts have been made available in a public GitHub repository. For latest versions of this document, script changes or new features please refer to that repository at: https://github.com/ppc64le/devops-automation/tree/master/terraform/oracle.

5 Lab Environment

The graphic below illustrates the environment in which this proof of technology was developed:



- Two POWER9 processor-based servers and two IBM POWER8 processor-based servers, all managed by an IBM Hardware Management Console (HMC),
- An IBM Storwize V7000 providing virtualized storage,
- Two fiber-based SAN switches connecting the storage to the servers (not shown in the graphic),
- The Power servers are fully virtualized and each was configured with two Virtual I/O Servers (VIOS),
- IBM Power Virtualization Center (PowerVC) as the OpenShift endpoint for VM deployments via Terraform,
- The IBM Cloud Private environment, with IBM Terraform & Service Automation (T&SA) deployed.

The software versions of OCP, T&SA and PowerVC in our lab environment at the time this paper was written were:

- Red Hat Openshift Container Platform 3.11.394
- IBM Cloud Pak for Multicloud Manager 1.3.1
- IBM Terraform & Service Automation Version 4.2.0.1
- IBM PowerVC Version 1.4.4.2

6 About Red Hat Openshift Container Platform and IBM Terraform & Service Automation

Red Hat Openshift Container Platform is an application platform for developing and managing containerized applications on-premises and public clouds. It is an integrated environment for managing containers that includes the container orchestrator Kubernetes, a private image registry, a management console, and monitoring frameworks.

Many applications today are still VM-based. For these environments, IBM Terraform & Service Automation (T&SA) is an application that consists of several containers executing on Red Hat OpenShift. T&SA allows you to deploy existing VM-based applications in a multi-cloud environment, add any AIX, IBM i, or Linux VM-based application to the IBM Cloud Private or Red Hat OpenShift catalog, integrate new services with existing mission critical workloads, and achieve a single catalog with coordinated orchestration.

Terraform & Service Automation uses Terraform, an open-source tool created by HashiCorp, to handle provisioning of cloud and infrastructure resources. These resources are described to Terraform using a high-level configuration language, stored in configuration files, or templates. From these templates, Terraform generates an execution plan describing what it will do, and executes it to build the described infrastructure.

T&SA includes two graphical user interface (GUI) tools, accessed through a web browser, for developing templates and services:

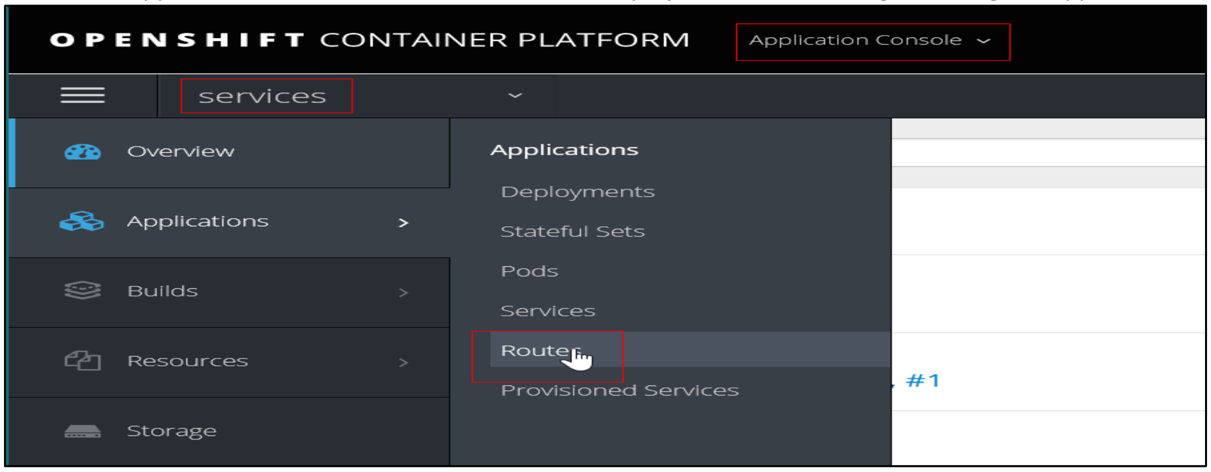
- Template Designer is used to develop Terraform templates featuring a "drag and drop" interface. Template Designer can also be installed on a local workstation using Docker Compose. Templates created using Template Designer are "pushed" to a Git repository and published or imported into the T&SA Template Library.
- Service Composer allows the creation of orchestrated services, via drag-and-drop interface, from T&SA (Terraform) templates based on a designed flow sequence or decision tree. Service Composer is part of the T&SA Service library.

Note that with IBM Multicloud Management Cloud Pak 2.x IBM Terraform & Service Automation, previously known as IBM Cloud Automation Manager, has been renamed to "Managed Services" under the umbrella term "Infrastructure and Service Management" (https://www.ibm.com/support/knowledgecenter/en/SSFC4F_2.1.0/cam/cam_intro.html).

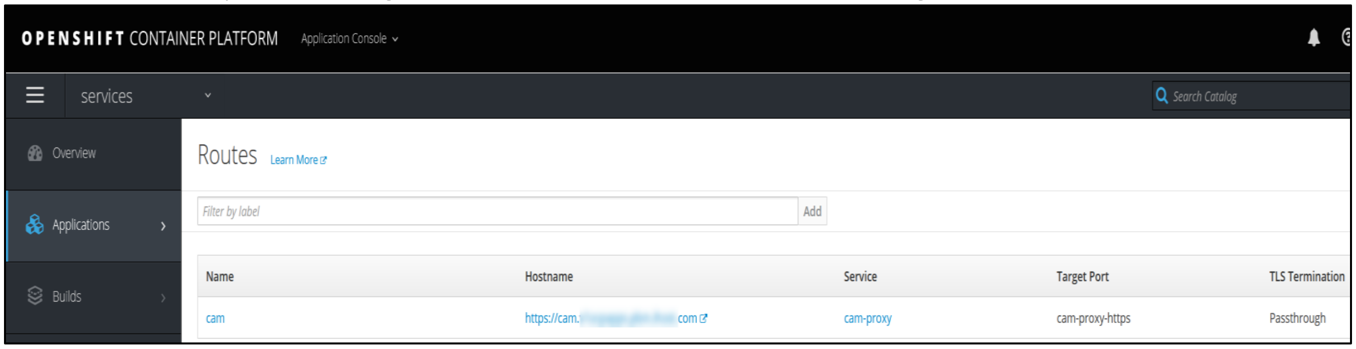
7 Working with Terraform & Automation Services

7.1 Accessing Terraform & Automation Services

Terraform & Automation Services (T&SA) is deployed as an add-on in CP4MCM and installed via a Helm Release. There is no direct link in OCP GUI to access T&SA, but there is an OCP route created at T&SA deploy time. To find that route open the OCP GUI -> Application Console and select **services** as the project. In the Hamburger menu go to Applications → Routes:

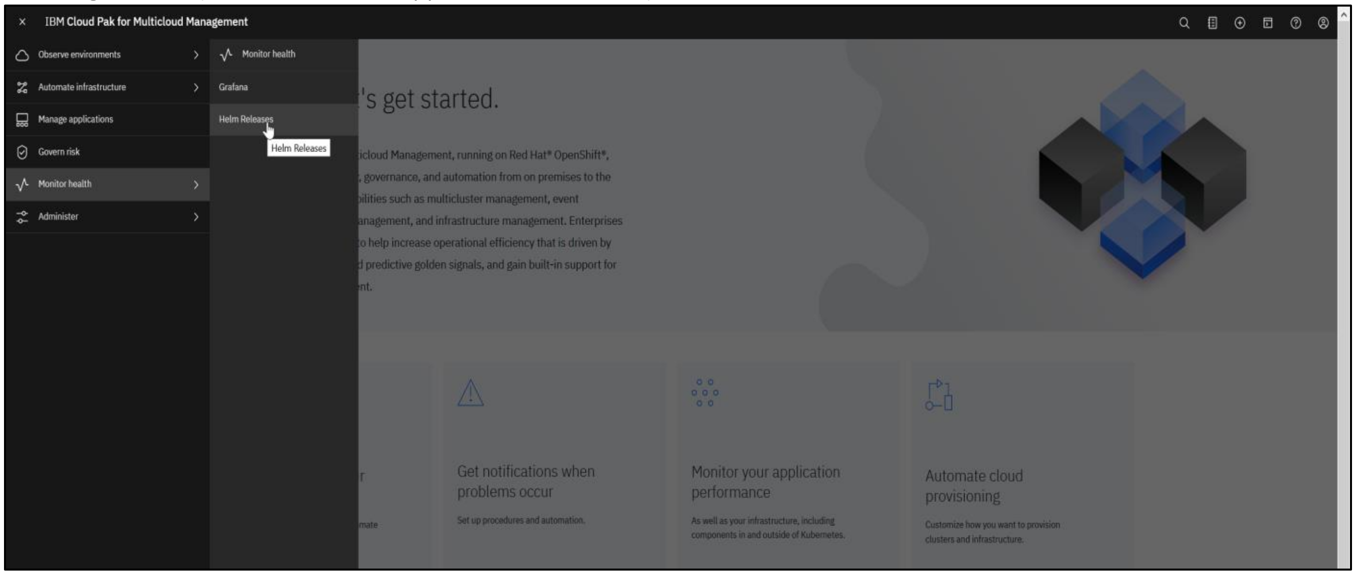


On the next screen you will then find the direct route to T&SA in the route where Name is **T&SA**. Copy / paste the value under Hostname into your browser to directly access T&SA. The entry is of the form **https://cam.<OCP subdomain>**. Note that the user ID and password to log in are from IBM Cloud Pak for Multicloud Management and not OCP!

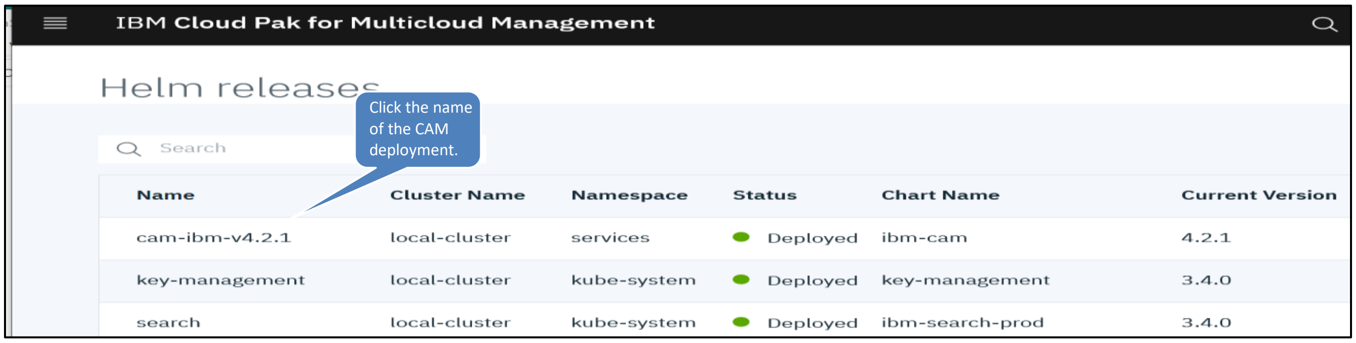


Name	Hostname	Service	Target Port	TLS Termination
cam	https://cam.[redacted].com	cam-proxy	cam-proxy-https	Passthrough

An alternative method to access T&SA is to go via the IBM Cloud Pak for Multicloud Management GUI. Access the hamburger menu (three bars on the upper left-hand corner), select: Monitor Health -> Helm Releases.


IBM Cloud Pak for Multicloud Management
Observe environments
Automate infrastructure
Manage applications
Govern risk
Monitor health
Administer
Monitor health
Grafana
Helm Releases
Helm Releases
's get started.
Get notifications when problems occur
Set up procedures and automation.
Monitor your application performance
As well as your infrastructure, including components in and outside of Kubernetes.
Automate cloud provisioning
Customize how you want to provision clusters and infrastructure.

Search for **cam-ibm** under Chart Name, or the name you used when you installed the T&SA Helm Release and then click the name of the T&SA deployment.

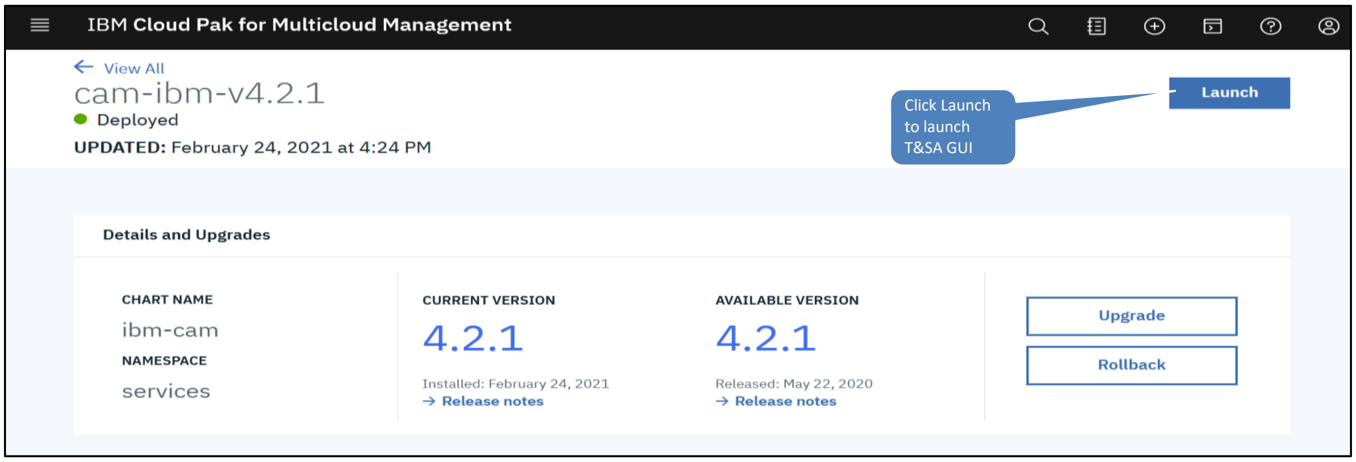

IBM Cloud Pak for Multicloud Management

Helm releases

Click the name of the CAM deployment.

Name	Cluster Name	Namespace	Status	Chart Name	Current Version
cam-ibm-v4.2.1	local-cluster	services	● Deployed	ibm-cam	4.2.1
key-management	local-cluster	kube-system	● Deployed	key-management	3.4.0
search	local-cluster	kube-system	● Deployed	ibm-search-prod	3.4.0

On the detail screen for the Helm Release you click on **Launch**. If asked for a user-ID and password, use the credentials defined in CP4MCM and not from OCP!


IBM Cloud Pak for Multicloud Management

View All

cam-ibm-v4.2.1

● Deployed

UPDATED: February 24, 2021 at 4:24 PM

Click Launch to launch T&SA GUI

Launch

Details and Upgrades

CHART NAME
ibm-cam
NAMESPACE
services

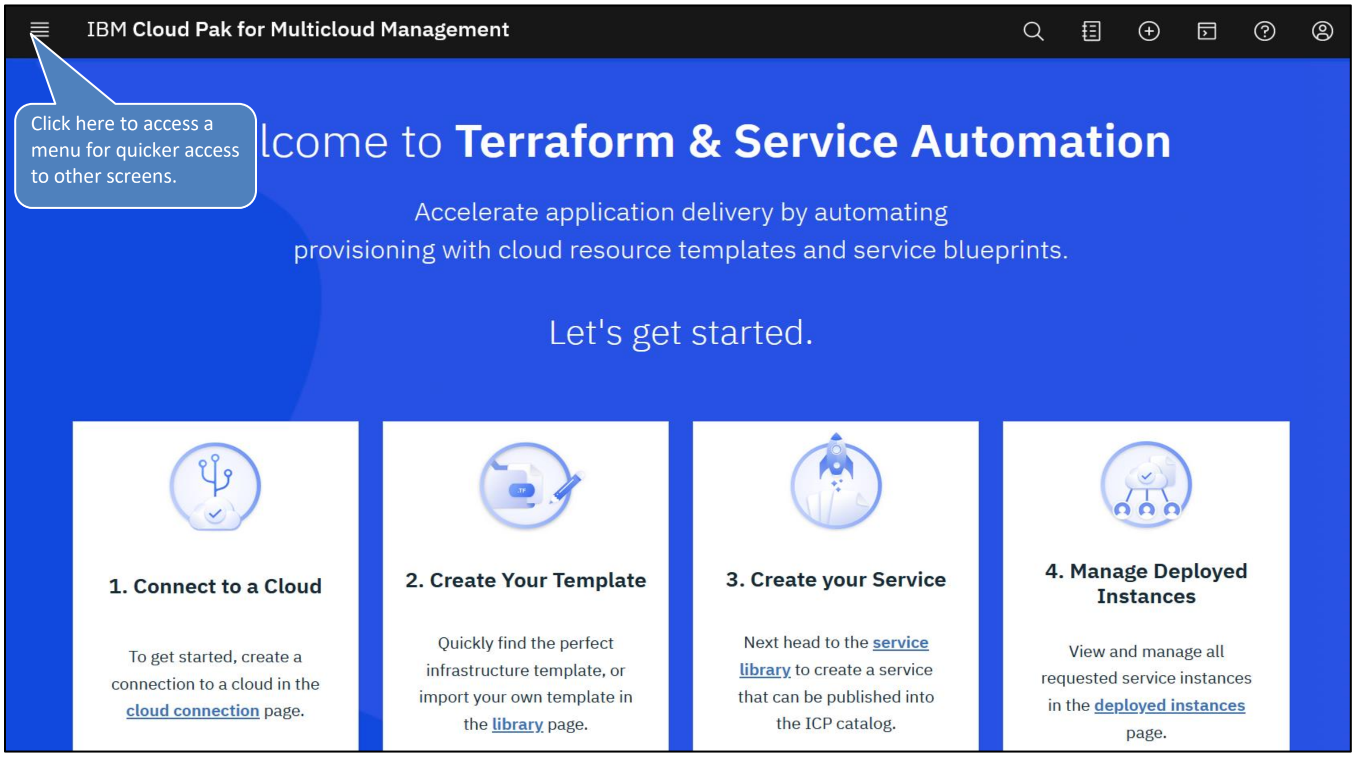
CURRENT VERSION
4.2.1
Installed: February 24, 2021
→ Release notes

AVAILABLE VERSION
4.2.1
Released: May 22, 2020
→ Release notes

Upgrade

Rollback

Once logged in, the **Welcome** screen or **Getting Started** screen is displayed as shown in the picture below.


IBM Cloud Pak for Multicloud Management

Click here to access a menu for quicker access to other screens.

lcome to **Terraform & Service Automation**

Accelerate application delivery by automating provisioning with cloud resource templates and service blueprints.

Let's get started.

1. Connect to a Cloud

To get started, create a connection to a cloud in the cloud connection page.

2. Create Your Template

Quickly find the perfect infrastructure template, or import your own template in the library page.

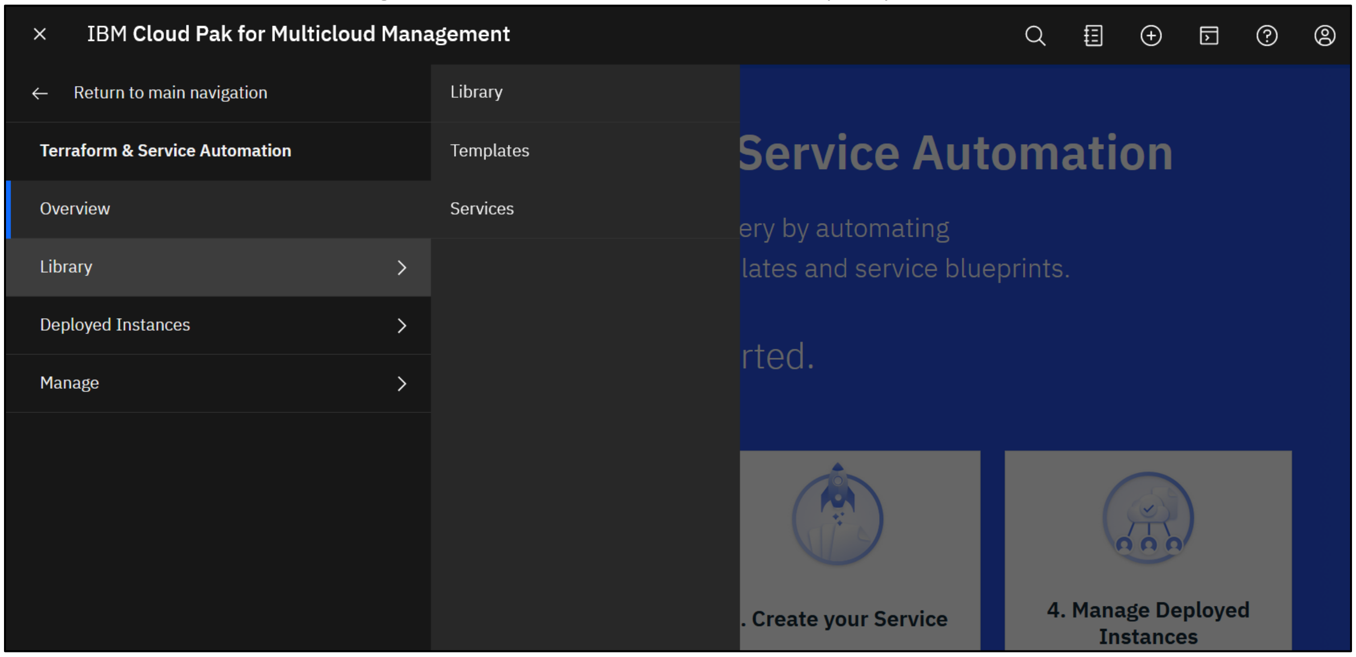
3. Create your Service

Next head to the service library to create a service that can be published into the ICP catalog.

4. Manage Deployed Instances

View and manage all requested service instances in the deployed instances page.

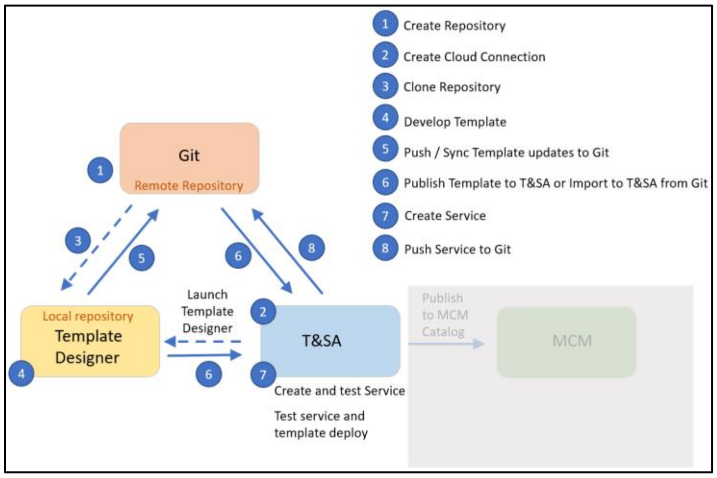
Note that the hamburger menu can be used to bring up the left-hand panel menu below. This menu can be used to access the functions shown on this **Getting Started** screen or return to this screen quickly from other screens.


IBM Cloud Pak for Multicloud Management
Return to main navigation
Terraform & Service Automation
Overview
Library
Deployed Instances
Manage
Library
Templates
Services
Service Automation
ery by automating
lates and service blueprints.
rted.
. Create your Service
4. Manage Deployed Instances

7.2 Development Workflow – Overview

Templates and content libraries that are created in the Template Designer are stored and versioned in a Git repository. GitHub, GitLab and Bitbucket server are supported in T&SA. For this paper, we used GitHub and have included the steps we performed to set up the repository in Appendix, Section 10.1, "Working with GitHub".

The picture below illustrates the sequence of steps we followed to prepare for, develop, test and deploy the two T&SA templates and the T&SA service for this proof of technology.



We started with the creation of a new GitHub repository, *cam-demo*, for this project under an existing GitHub account, then cloned that new repository (empty at this point) to the Template Designer. We then used Template Designer to create and modify the Terraform files defining the *oradbaas* and *oraclient* templates, committed our changes and pushed the updated files to the GitHub repository. Once that was done, we published the new templates (using Template Designer) to T&SA, where we then tested / successfully deployed the *oradbaas* template.

Once the templates had been tested, we used T&SA Service Composer to create and test the T&SA Service *Oracle 12C on AIX with Swingbench Client on LoP*. This new service is an orchestration of *oradbaas* and *oraclient* templates. We finalized the new service by pushing it into the GitHub repository and publish it to the OCP service catalog.

Pushing templates and services to a GitHub repository serves as backup and facilitates distribution and collaboration with other interested parties. Templates and services can be easily imported into other T&SA deployments, saving the effort of recreating templates and services in T&SA.

7.3 Terraform Console

Git provides change control and management that may not be needed when first starting Terraform template development. One way to quickly learn, create and test early prototypes of Terraform files is to install Terraform on your Windows, Mac or Linux workstation.

This blog, based on an article by Joe Cropper, IBM, describes how to install and run Terraform on your workstation.
http://gibsonnet.net/blog/cgaix/html/Using Terraform with PowerVC to deploy new AIX VMs.html

One useful function of locally installed Terraform software is the Terraform console, which we used to test and verify Terraform functions, including syntax checking. For example, on our workstation, the following was used to test the timestamp function used in our *oradbaas* template.
Note that the console functionality, at least on windows, seems to be broken with current version 0.12, but works with version 0.11.14 of Terraform.

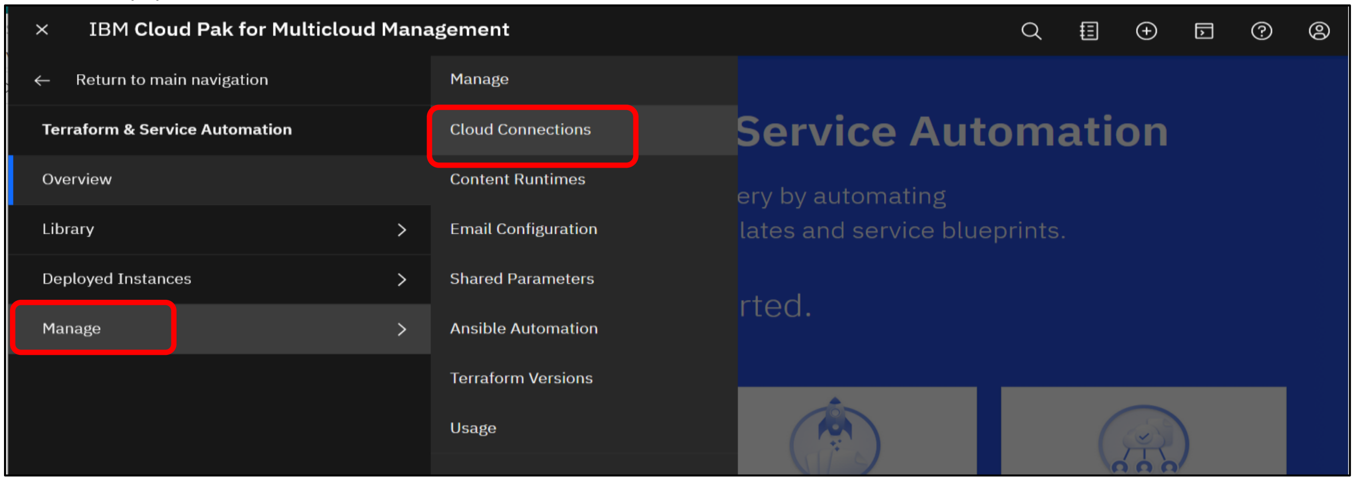
```
$ terraform console
> timestamp()
2019-05-23T18:41:34Z
> replace(replace(replace(replace(substr(timestamp(),0,19),"-","),"T",""),"T",""),":","")
20190523184013
> replace(replace(replace(substr(timestamp(),2,17 ),"-",""),"T","_"),":","")
190911_172625
> exit
$
```

7.4 Create Cloud Connection

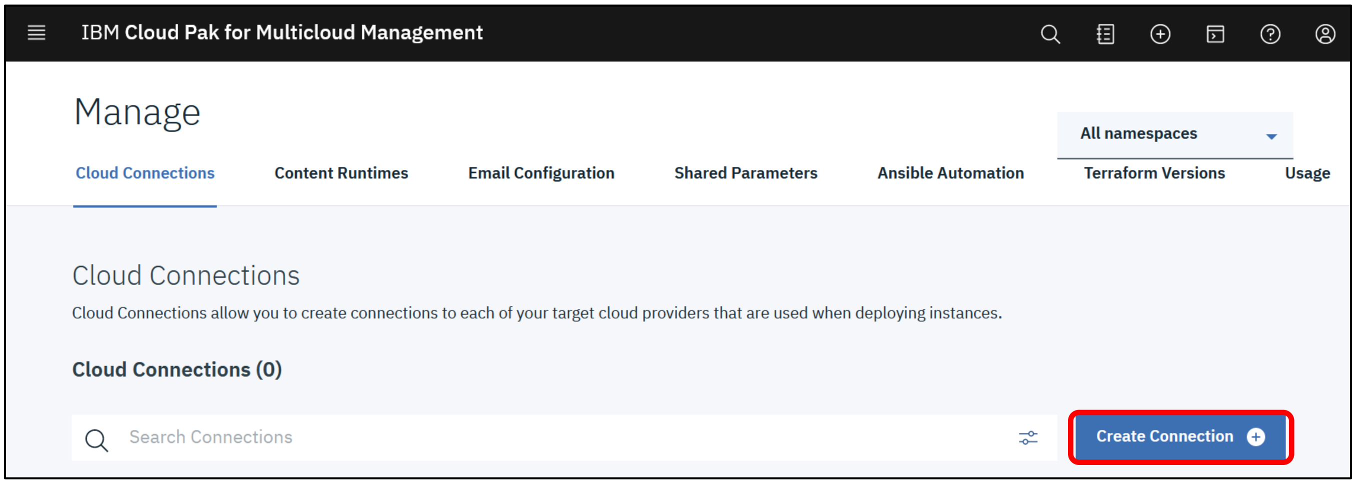
T&SA is designed to work efficiently in a multi-cloud environment with multiple cloud providers. To simplify the use of a specific cloud provider T&SA provides the concept of a Cloud Connection which stores the detailed connection related parameters (e.g. OpenStack user, password, auth_url, region, etc.). At deploy time – template or service – you can then simply specify which Cloud Connection to use without having to specify all the cloud provider connection details in the deployment.

To create a new Cloud Connection, click the left-side navigation panel, click **Manage** as shown on the screen below to expand the submenu and select **Cloud Connections**.

Alternatively, you can click **1. Connect to a Cloud** link on the Welcome screen



This switches the focus to the Manage (connections) screen where you click the **Create Connection** button,

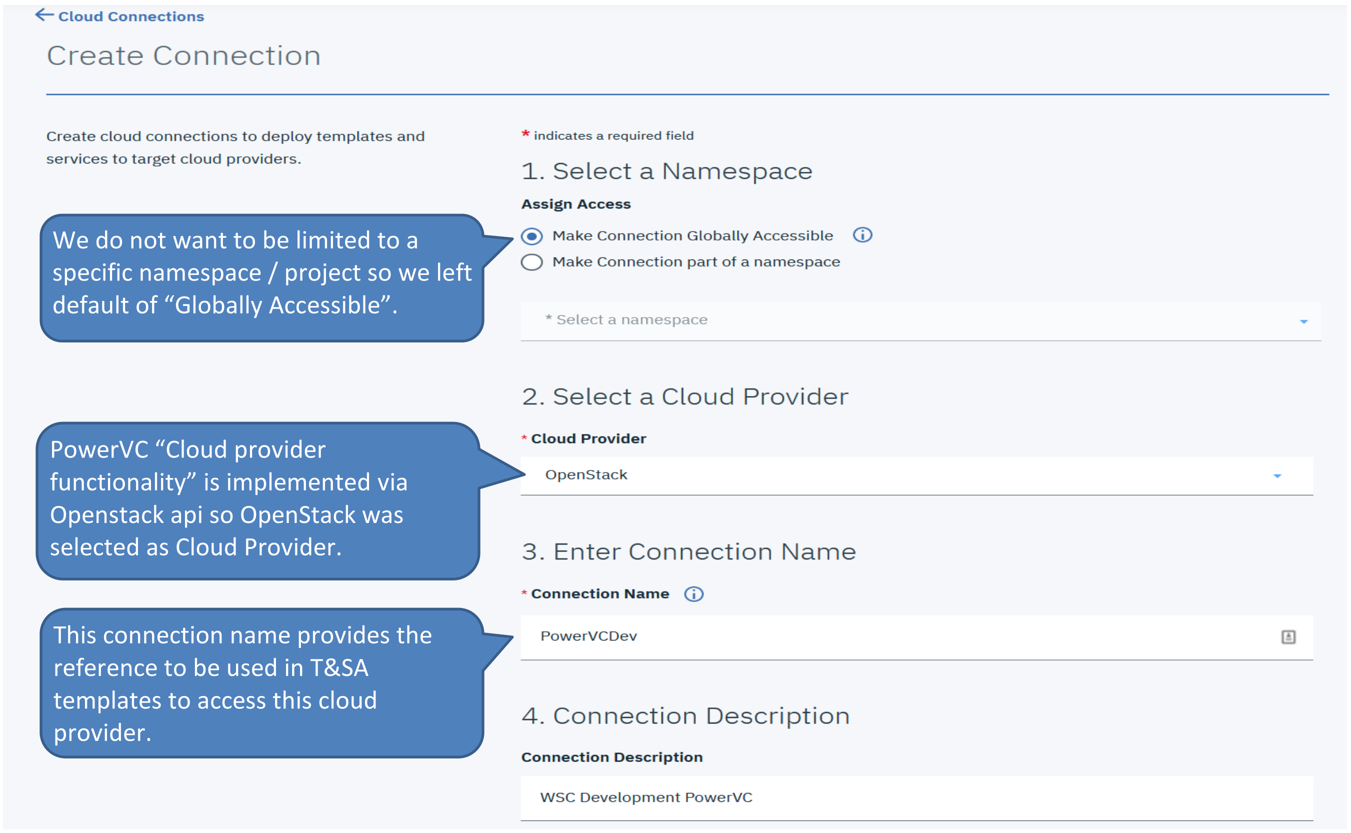


Cloud Connections

Cloud Connections allow you to create connections to each of your target cloud providers that are used when deploying instances.

Cloud Connections (0)

Fill in the Create Connection screen as described below. Important for a PowerVC environment is to select OpenStack as **Cloud Provider**. We also defined the new cloud connection to be available in all namespaces – Globally Accessible. Note that in this case the Select a namespace field is not editable.



Create Connection

Create cloud connections to deploy templates and services to target cloud providers.

* indicates a required field

1. Select a Namespace

Assign Access

(● Make Connection Globally Accessible) ○ Make Connection part of a namespace

* Select a namespace

2. Select a Cloud Provider

* Cloud Provider: OpenStack

3. Enter Connection Name

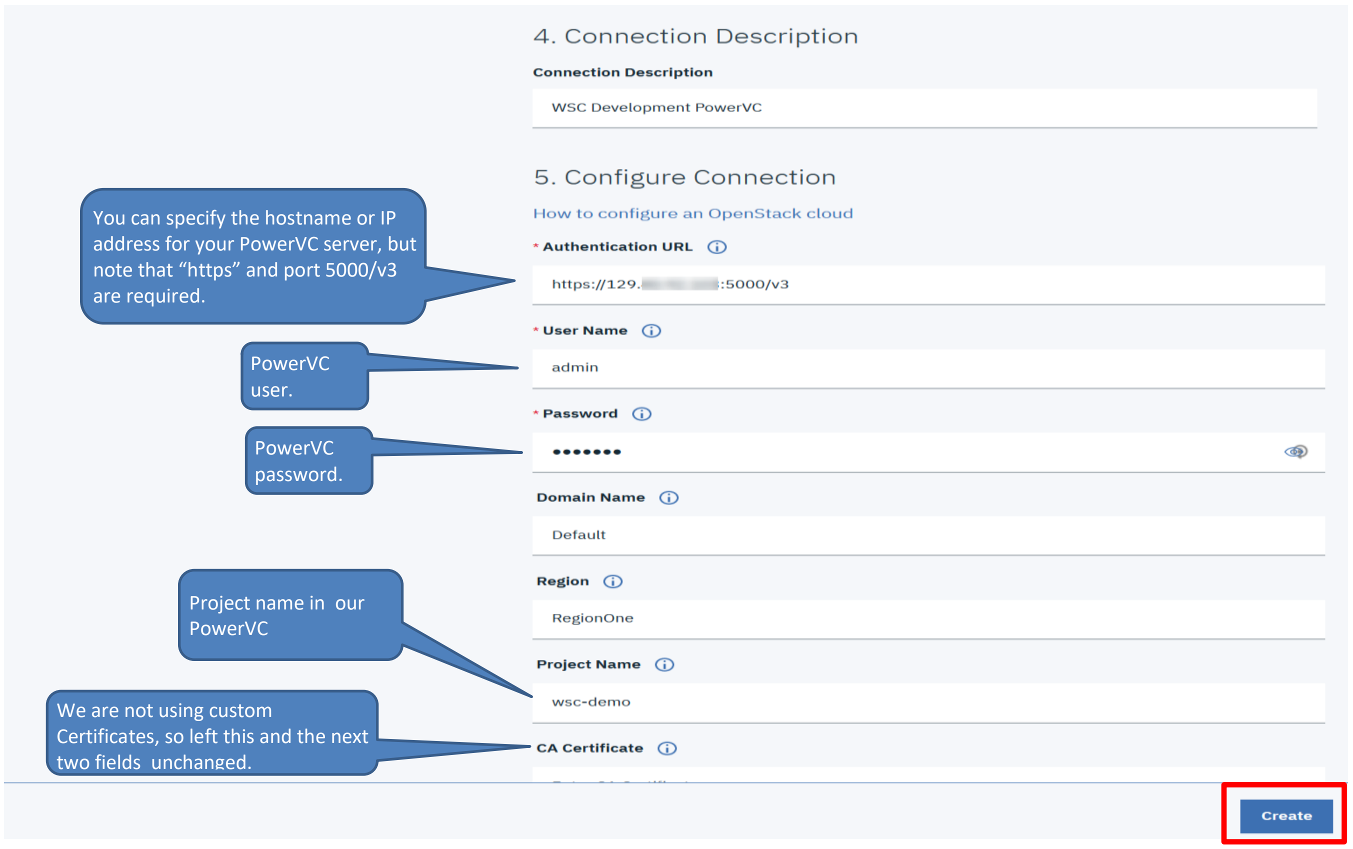
* Connection Name: PowerVCDev

4. Connection Description

Connection Description: WSC Development PowerVC

We do not want to be limited to a specific namespace / project so we left default of “Globally Accessible”.
PowerVC “Cloud provider functionality” is implemented via Openstack api so OpenStack was selected as Cloud Provider.
This connection name provides the reference to be used in T&SA templates to access this cloud provider.

… continued on next page.



4. Connection Description

Connection Description: WSC Development PowerVC

5. Configure Connection

How to configure an OpenStack cloud

* Authentication URL: https://129.[redacted]:5000/v3

* User Name: admin

* Password: •••••••

Domain Name: Default

Region: RegionOne

Project Name: wsc-demo

CA Certificate

You can specify the hostname or IP address for your PowerVC server, but note that "https" and port 5000/v3 are required.

PowerVC user.

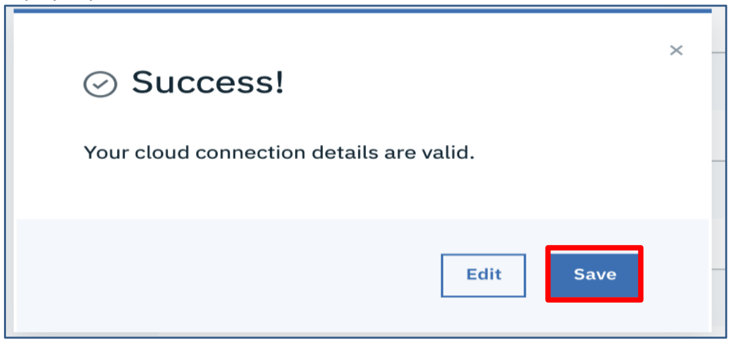
PowerVC password.

Project name in our PowerVC

We are not using custom Certificates, so left this and the next two fields unchanged.

When complete, click the **Create** button.

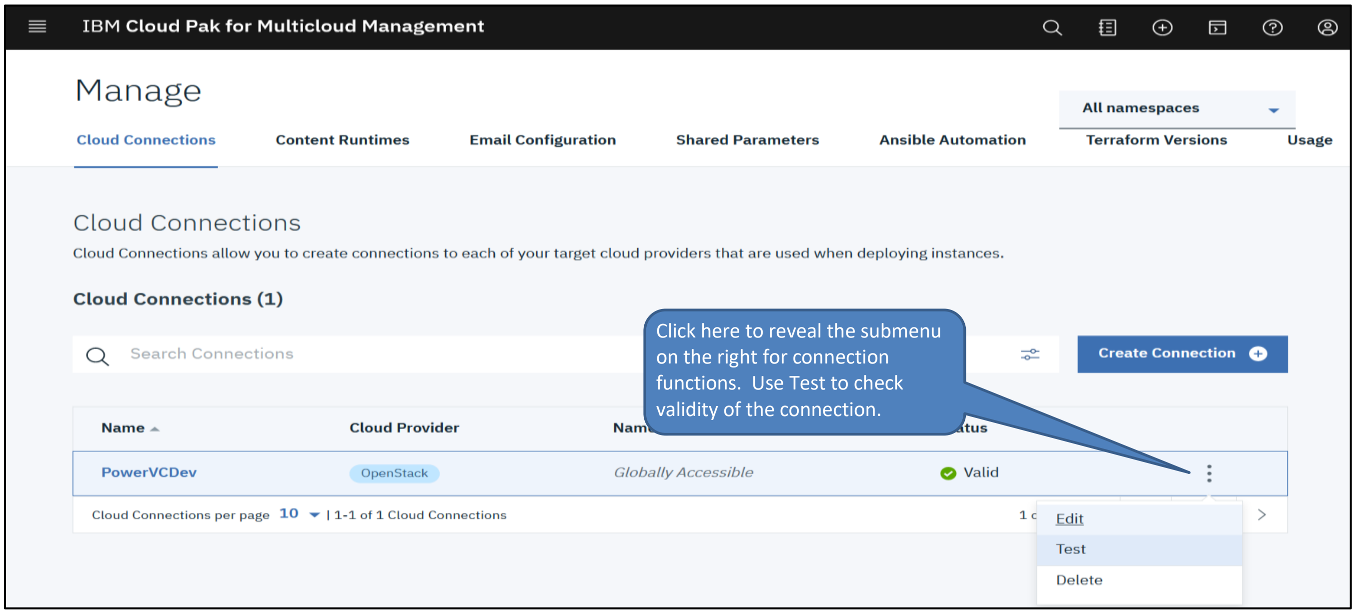
A pop-up window indicates successful creation of the connection. Click the **Save** button.



Success!

Your cloud connection details are valid.

The Manage screen now shows the just-created Cloud Connection.


IBM Cloud Pak for Multicloud Management

Manage

Cloud Connections | Content Runtimes | Email Configuration | Shared Parameters | Ansible Automation | Terraform Versions | Usage

All namespaces

Cloud Connections

Cloud Connections allow you to create connections to each of your target cloud providers that are used when deploying instances.

Cloud Connections (1)

Search Connections

Create Connection

Click here to reveal the submenu on the right for connection functions. Use Test to check validity of the connection.

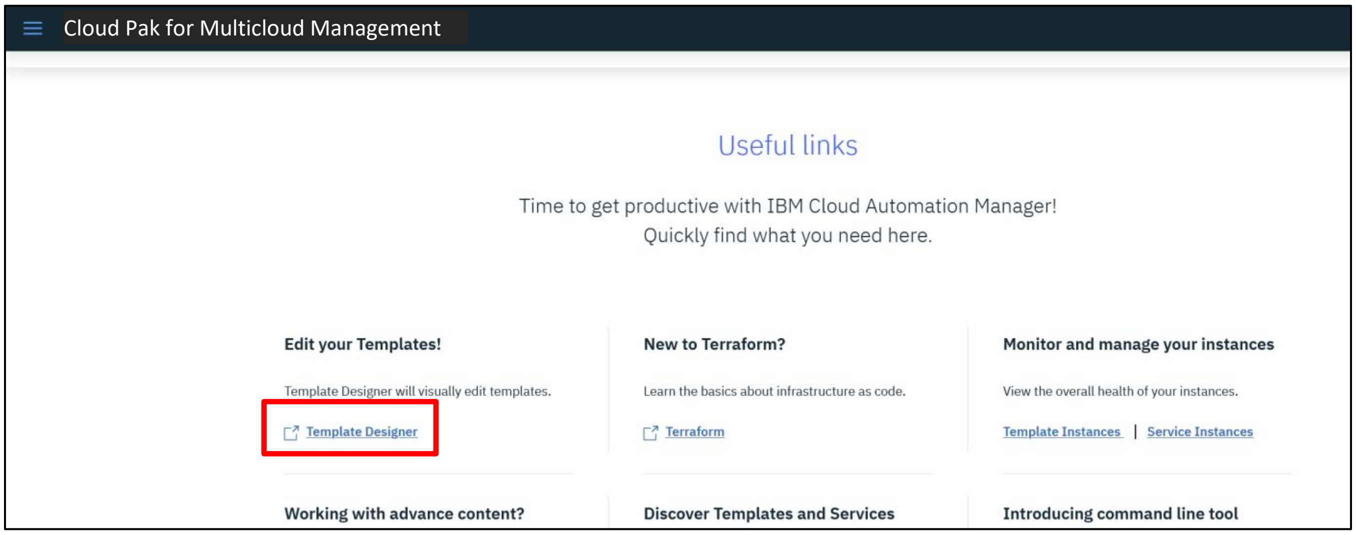
Name	Cloud Provider	Namespace	Status
PowerVCDev	OpenStack	Globally Accessible	Valid

Cloud Connections per page 10 | 1-1 of 1 Cloud Connections

7.5 Prepare Template Designer

In this section we describe the steps to configure Template Designer before starting to create the first template.

From the **Getting Started** screen, scroll down and launch **Template Designer**, as shown in the graphic below. Alternatively, click the hamburger menu, select Library -> Templates. Then click the **Create Template** button.


Cloud Pak for Multicloud Management

Useful links

Time to get productive with IBM Cloud Automation Manager!
Quickly find what you need here.

Edit your Templates!

Template Designer will visually edit templates.

Template Designer

New to Terraform?

Learn the basics about infrastructure as code.

Terraform

Monitor and manage your instances

View the overall health of your instances.

Template Instances | Service Instances

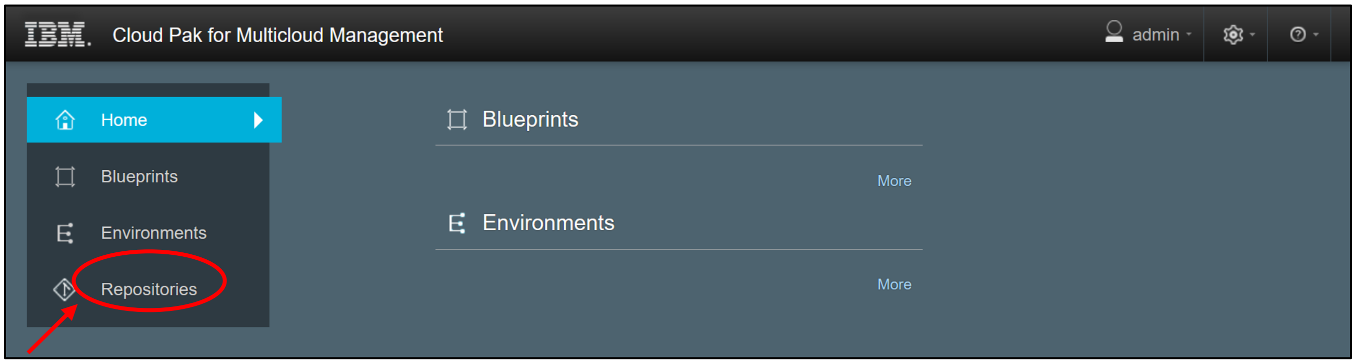
Working with advance content?

Discover Templates and Services

Introducing command line tool

After going through the OCP login screen again, template Designer will launch in a separate browser tab.

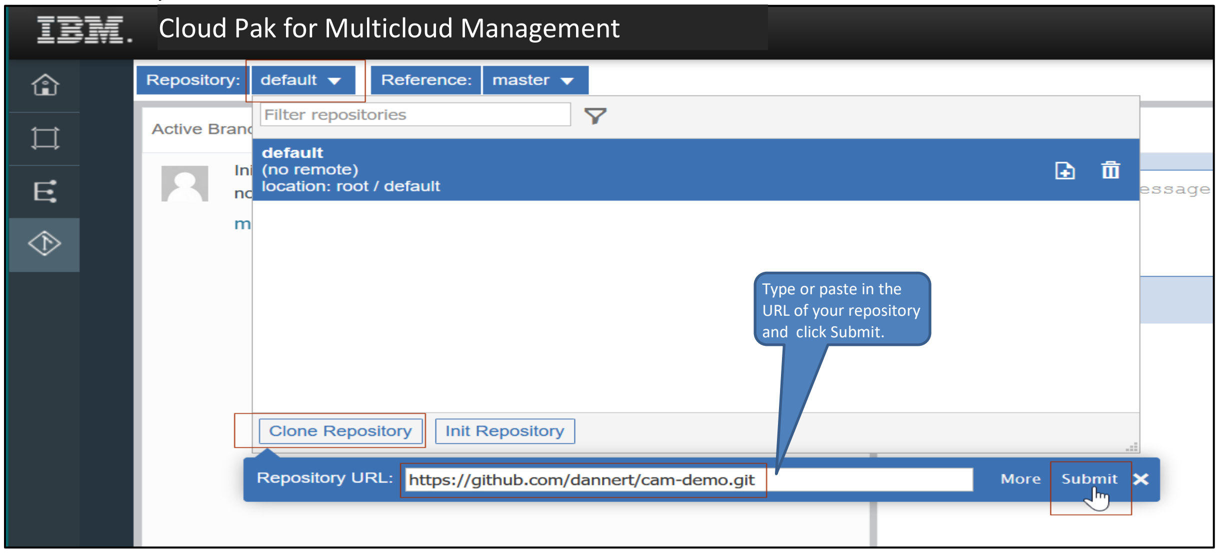
On the Template Designer browser tab, click Repositories (circled below).



7.5.1 Create Repository

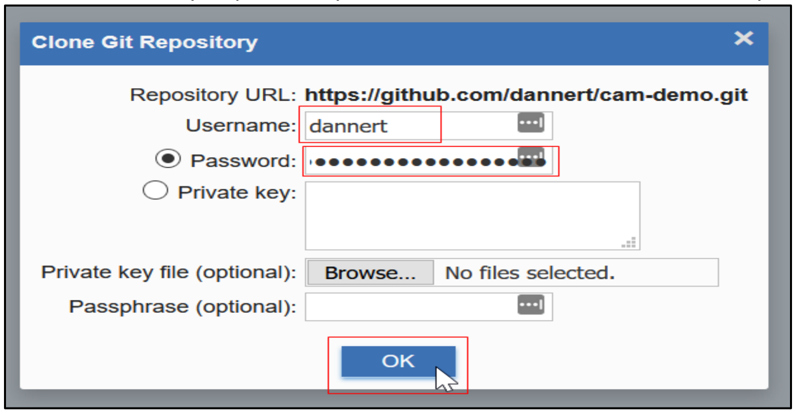
This section assumes that you have already created a Git repository for this project on github.com and have the URL and access token available. For reference see: [Working with GitHub](#) (Section: 10.1)

Click the down arrow next to Repository **default**, then click **Clone Repository**, fill in the GitHub repository URL into the input field and finally click on **Submit**.



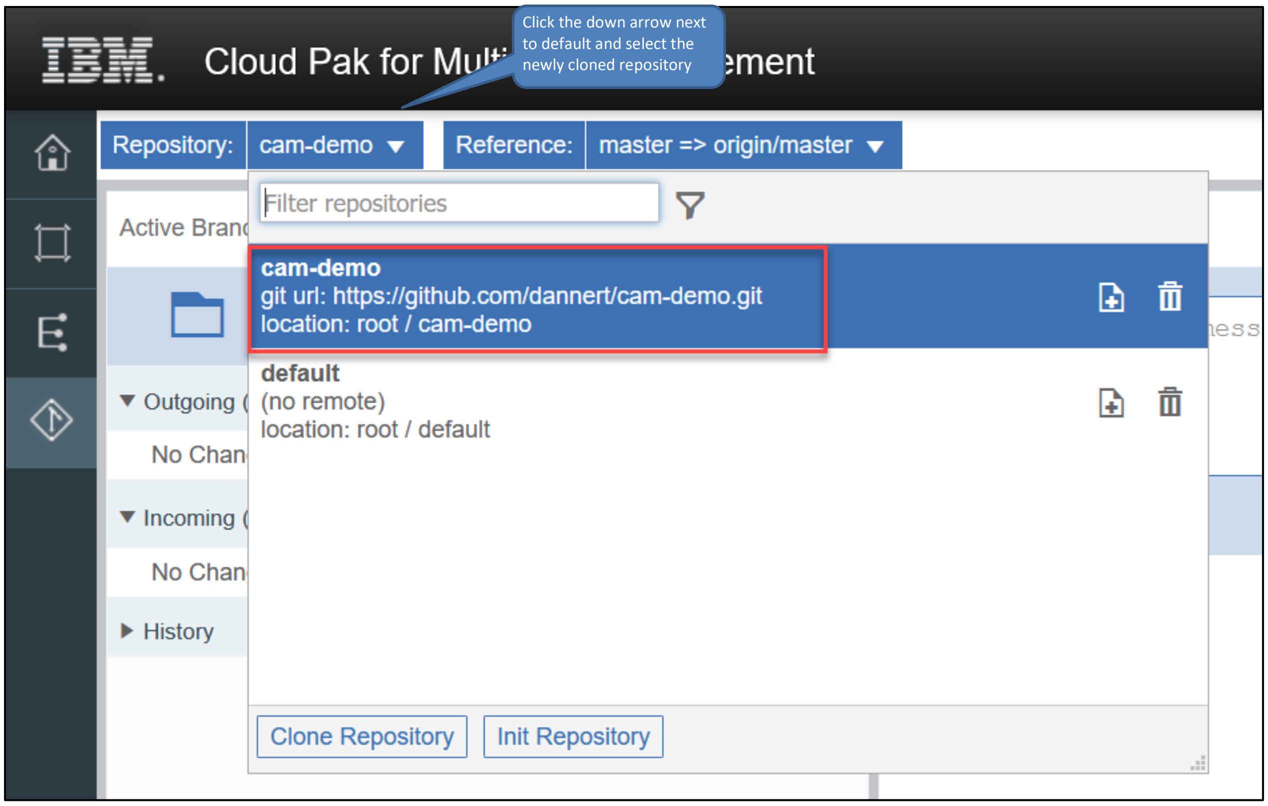
Repository URL: https://github.com/dannert/cam-demo.git

After clicking **Submit** the following window pops up. Enter your GitHub user ID into **Username** and, into the **Password** field, the access token you previously created on the GitHub website. Finally click **Ok**.



Repository URL: https://github.com/dannert/cam-demo.git
Username: dannert
● Password: ••••••••••••••••
○ Private key:
Private key file (optional): No files selected.
Passphrase (optional):

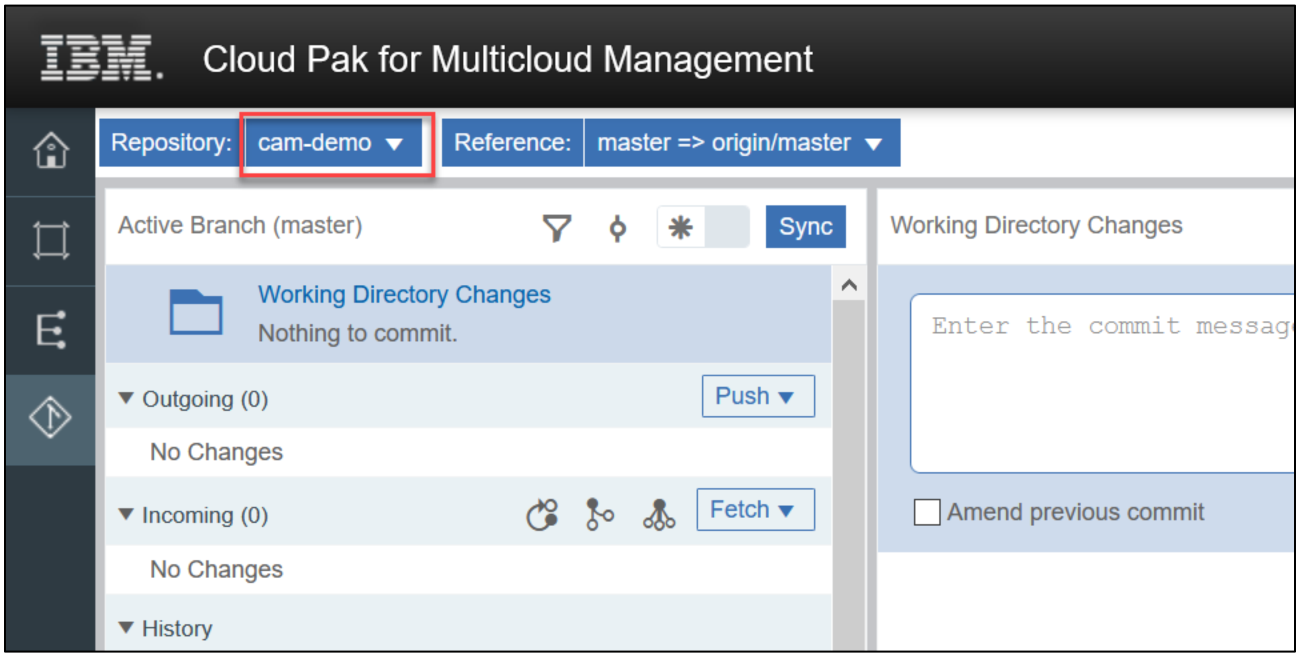
To set this new repository as the default repository click again onto the “default” drop-down and select the new project as the new default as shown below.


Click the down arrow next to default and select the newly cloned repository
IBM. Cloud Pak for Multicloud Management
Repository: cam-demo
Reference: master => origin/master
Filter repositories
Active Branch
Outgoing
No Changes
Incoming
No Changes
History
Clone Repository
Init Repository

cam-demo
git url: https://github.com/dannert/cam-demo.git
location: root / cam-demo

default
(no remote)
location: root / default

Now the Repository field shows **cam-demo** as selected. Note that we are referring to the master branch, but you could modify that to a different branch/fork for a project via the Reference field.



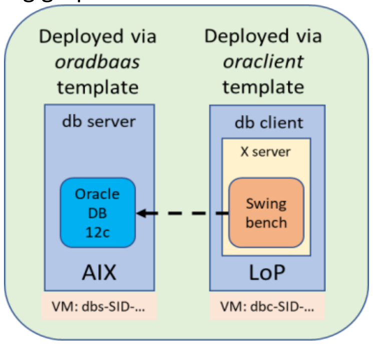
8 Developing our Templates

In this section we describe how to create the two T&SA templates used in this project to achieve the following:

a) Deploy an AIX VM running Oracle Database 12c (dbserver). The Oracle database name (dbname) is entered as an input parameter and used to create the database instance in the deployed VM. This template can be deployed by itself. We will call this template *oradbaas*.

b) Deploy a Linux on Power VM, set up Oracle Instant Client and the Swingbench load generator in the deployed VM (dbclient). Dbclient will wait for dbserver to complete post-deployment customization, which includes the database creation and startup, before loading the Swingbench Order Entry schema into the Oracle database. This template cannot be deployed by itself as it requires input parameters from the deployment of the oradbaas template. We will call this template *oraclient*.

The following graphic illustrates the end state after an orchestrated deployment of both templates via a service.



Note that it is possible to create only one template to deploy both VMs, but that would preclude the re-use of either template in other contexts / services. Separating the deployment into two different templates also allows us to demonstrate how required or dependent data is passed between several template deployments when deployed as an orchestrated service. An example of such required data exchange is the passing of the IP address of the dbserver and the dbname of the Oracle database to the dbclient. Those values are needed by dbclient to be able to successfully connect to the Oracle database on dbserver.

We will only describe the two-template design in this paper.

8.1 *oradbaas* Template Design

In this section we provide the step-by-step instructions on how to create the *oradbaas* template.

8.1.1 *oradbaas* template

The *oradbaas* template relies on the Oracle DbaaS image to be available in PowerVC for deployment. Details on how we created that image can be found in IBM Technical Whitepaper, *Deploying Oracle Database as a Service with IBM Cloud PowerVC Manager 1.3.1*. This image utilizes Openstack Cloud Init and accepts the Oracle database name (dbname) as an input parameter passed to the post-deployment customization scripts that are imbedded in the captured image.

The following provides a quick overview of the key configuration blocks in main.tf. Please find the actual settings used later in this section.
The *oradbaas* template has two key configuration blocks:

- openstack provider, and
- openstack compute resource (openstack_compute_instance_v2).

The **default** openstack provider configuration block in file main.tf contains the following configuration arguments:

```
provider "openstack" {
  user_name    = "<User id to log on to OpenStack server, e.g. PowerVC>"
  password     = "<User password>"
  tenant_name = "<PowerVC project, usually ibm-default>"
  domain_name = "<PowerVC domain, usually Default>"
  auth_url     = "<Identity authentication URL of OpenStack server>,
                   e.g. https://<IP address or hostname of OpenStack server>:5000/v3/>"
  insecure =     "<Trust self-signed SSL certificates>"
  version   = "~> 1.2"
}
```

All but the last two of the arguments are already specified in T&SA using the Cloud Connection and therefore only the last two arguments are needed in a T&SA template.

The compute resource (openstack_compute_instance_v2) requires the following arguments and they were all left as input parameters in our templates.

```
Resource "openstack_compute_instance_v2" "<name of compute instance resource>" {
  name       =    "<Name of VM>"
  image_name  =   "<Name of the captured image on PowerVC>"
  flavor_name =   "<Name of the OpenStack flavor, or compute template in PowerVC>"
  user_data =     "<Special scripts processed by cloud-init. Used here for dbname>"
  network {
    uuid = "<ID of the network in PowerVC to attach to the instance>"
  }
}
```

Notes:

- Use PowerVC GUI or CLI to obtain values for image_name / image ID: **openstack image list**
- Use PowerVC GUI or CLI to obtain values for flavor_name / flavor ID (compute templates): **openstack flavor list**
- Use PowerVC CLI to obtain network name / network ID: **openstack network list**

8.1.2 Input and Output Parameters

Template developers should always have re-usability and flexibility in mind when creating a new template. To achieve maximum flexibility for our new template we decided to define all our parameters as template input parameters which could be specified / overridden at deployment time. The table below list the parameters and their meaning for the *oradbaas* template.

Input Parameter	**Description**
Namespace	This is the OCP / Kubernetes namespace or Red Hat project in which the instance will be deployed.
Instance name	This is the name of the instance being deployed; displayed on the T&SA deployed instance screen. Choose any appropriate name.
dbserver_name	This is used to name the PowerVC VM. In our template, we concatenate this parameter to the dbname (from dbserver_user_data), and a timestamp to have a name that's unique and one that can be matched with the dbclient VM name. For example, **dbs**-testdb-<YYMM_hhmmss>.
Openstack_flavor_name	This is the PowerVC compute template which specifies the compute characteristics of the VM to be deployed. The flavor (OpenStack) or compute template (PowerVC) specifies VM specifications, including number of CPUs, RAM, etc. Alternative option would be to use the flavor id instead.
Openstack_image_name	This is the name of the deployable image in PowerVC. Alternative option would be to use image id.
Openstack_network_id	This is the network id in OpenStack. Alternatively, we can choose to use network name.
dbserver_user_data	This is used to code cloud-init cloud-config. In this template, we use this to pass the desired Oracle dbname. If not provided, our PowerVC image will take the given VM name, truncated to 8 characters.

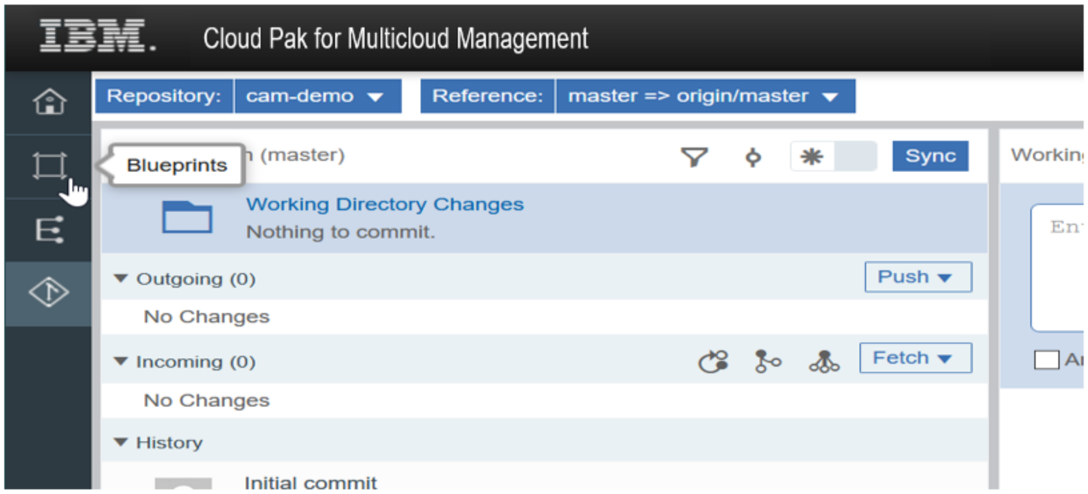
Note that image name, network name and flavor name can be changed via the GUI in PowerVC and it is therefore typically safer to use the respective ID value instead. For this project we wanted to demonstrate both options.

To be able to display deployment related information in the T&SA deployment screen, or to make information available to other templates during a service deployment, we defined the following output parameters in this *oradbaas* template.

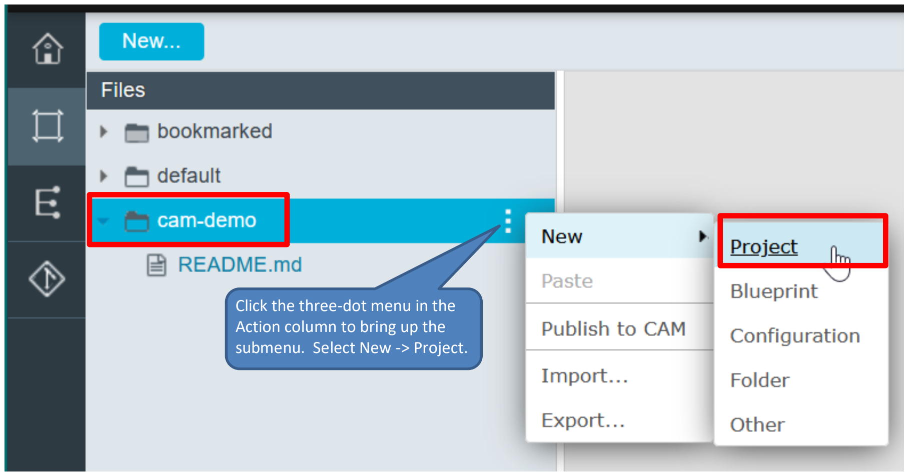
Output Parameter	Description
dbserver_ipaddr	The IP address of the db server. By not specifying a fixed ip address, the IP address is dynamically assigned by PowerVC from the pool of available IP addresses in the network specified by openstack_network_id input parameter.
dbserver_vm_name	This is the VM name in PowerVC, which is dynamically generated at deploy time based on input parameters and a local timestamp.
dbserver_ts	This output parameter is hidden and used to pass a local timestamp to another template. In our example, we pass the timestamp to oraclient for use in generating the VM name for the oraclient. This approach allows us to easily match DB server and client VMs in PowerVC to each other.
db_name	Database name to illustrate parameter passing to dbclient template

8.1.3 Create oradbaas template

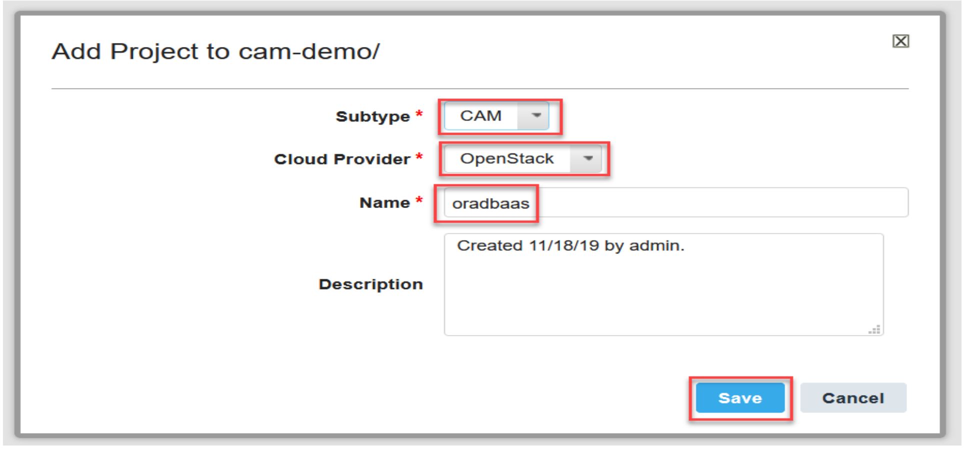
After we prepared Template Designer and defined the input and output parameters we want to use / expose in this template, it is now time to create the *oradbaas* template in Template Designer.
In Template Designer click the **Blueprints** icon on the left-hand panel,



Select the Cam-demo repository and click on New -> Project as shown below.



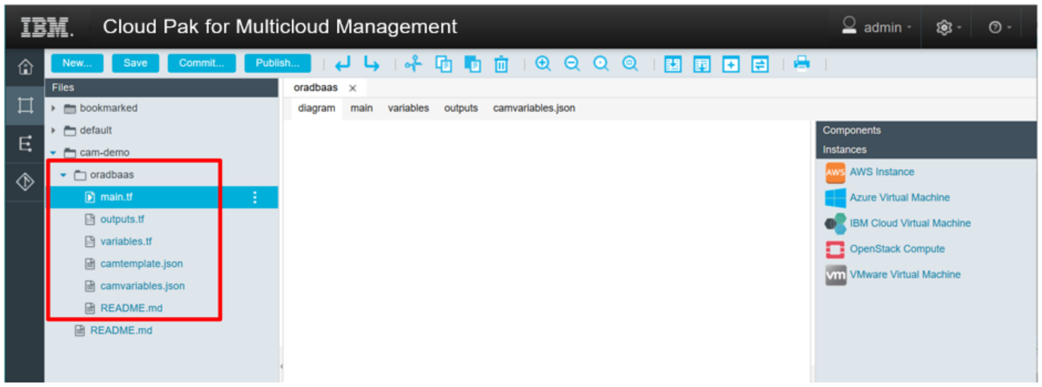
In the pop-up window below, set **Subtype**, **Cloud Provider** and **Name** exactly as shown and then click **Save**.



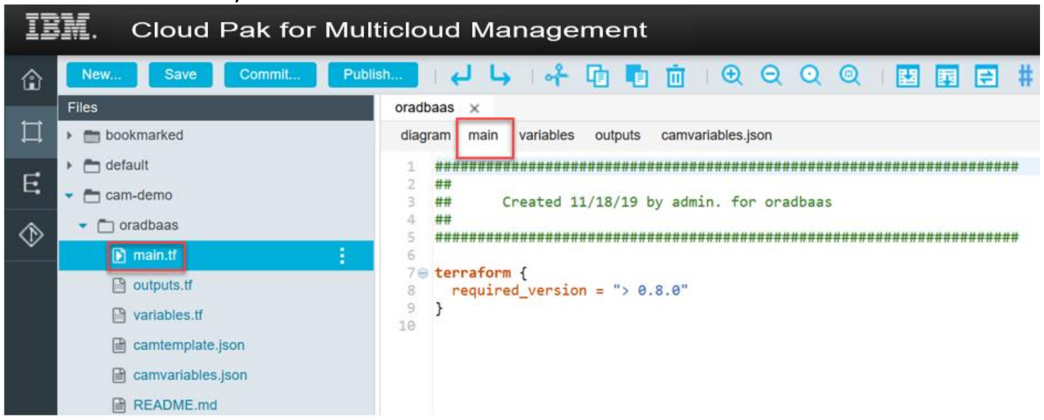
Add Project to cam-demo/

Subtype * CAM
Cloud Provider * OpenStack
Name * oradbaas
Description Created 11/18/19 by admin.

When you click **Save** Template Designer automatically creates a starting set of files for the template.

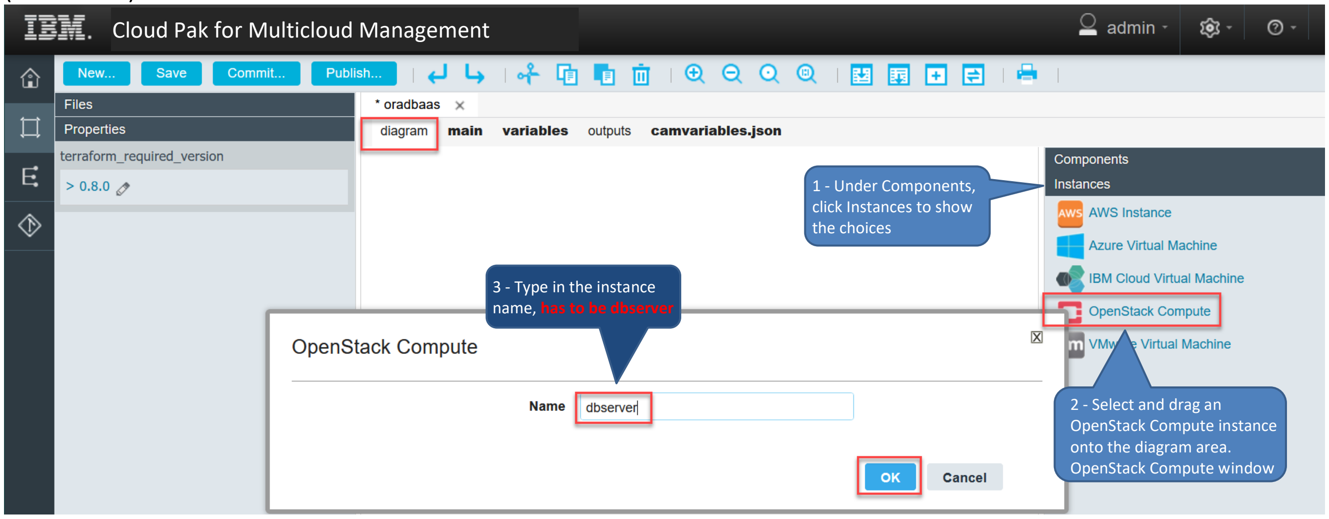


Select the main.tf entry to select the *oradbaas* template and then the main tab to examine the contents. Note that it's just a framework currently.


IBM. Cloud Pak for Multicloud Management
New... Save Commit... Publish...
Files
bookmarked
default
cam-demo
oradbaas
main.tf
outputs.tf
variables.tf
camtemplate.json
camvariables.json
README.md
oradbaas
diagram main variables outputs camvariables.json

```
##############################################################################
##
##      Created 11/18/19 by admin. for oradbaas
##
##############################################################################

terraform {
  required_version = "> 0.8.0"
}
```

To populate the template with required resources we utilize the drag-and-drop feature provided. Select the diagram tab, select **Instances** and then drag **OpenStack Compute** instance from the right-hand panel to the empty space in the middle (the canvas).


IBM. Cloud Pak for Multicloud Management
admin
New... Save Commit... Publish...
Files
Properties
terraform_required_version
> 0.8.0
* oradbaas
diagram main variables outputs camvariables.json
Components
Instances
AWS Instance
Azure Virtual Machine
IBM Cloud Virtual Machine
OpenStack Compute
VMware Virtual Machine
OpenStack Compute
Name
dbserver
OK
Cancel

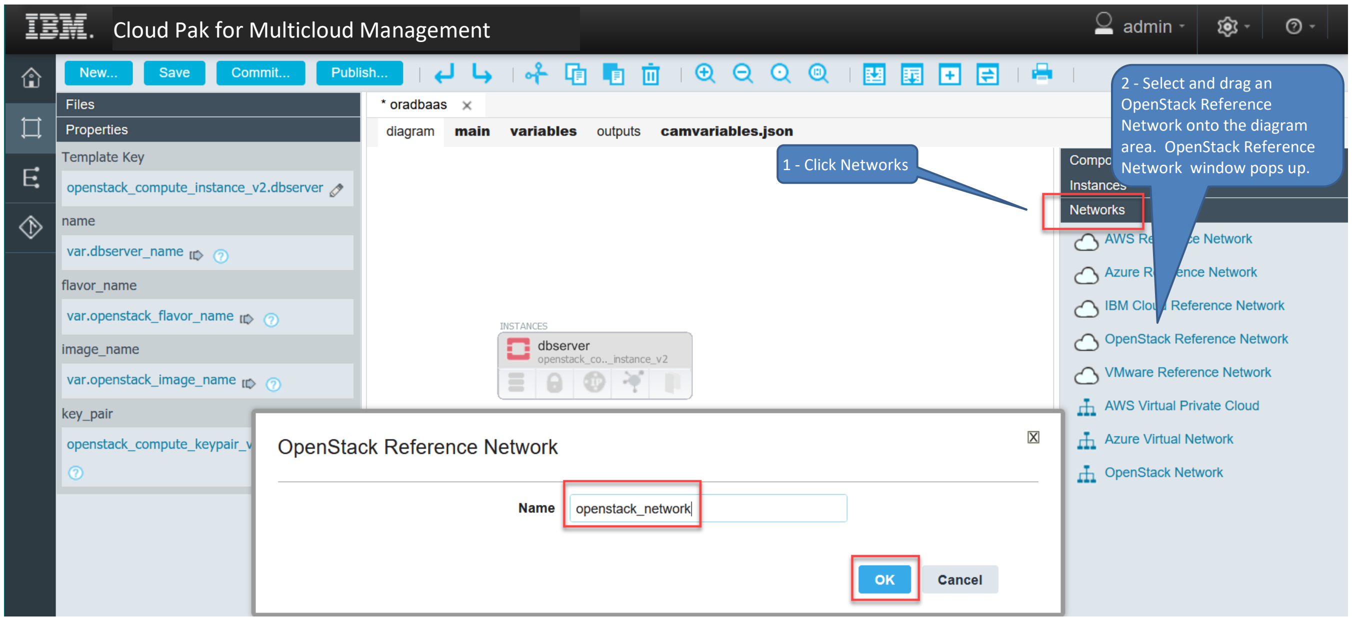
1 - Under Components, click Instances to show the choices

3 - Type in the instance name, has to be dbserver

2 - Select and drag an OpenStack Compute instance onto the diagram area. OpenStack Compute window

After typing in the OpenStack Compute instance name, **dbserver,** shown in the picture above, click **OK**. Please ensure that you enter the name exactly as shown as otherwise you will not be easily able to utilize the prepared template and service files as shown in the appendix. Note that this instance name becomes part of the generated template variable names, should not include spaces and should be meaningful.

Next, select **Networks** and drag **OpenStack Reference Network,** as shown below, to the canvas. We left the Name unchanged as “openstack_network” and click **OK**.

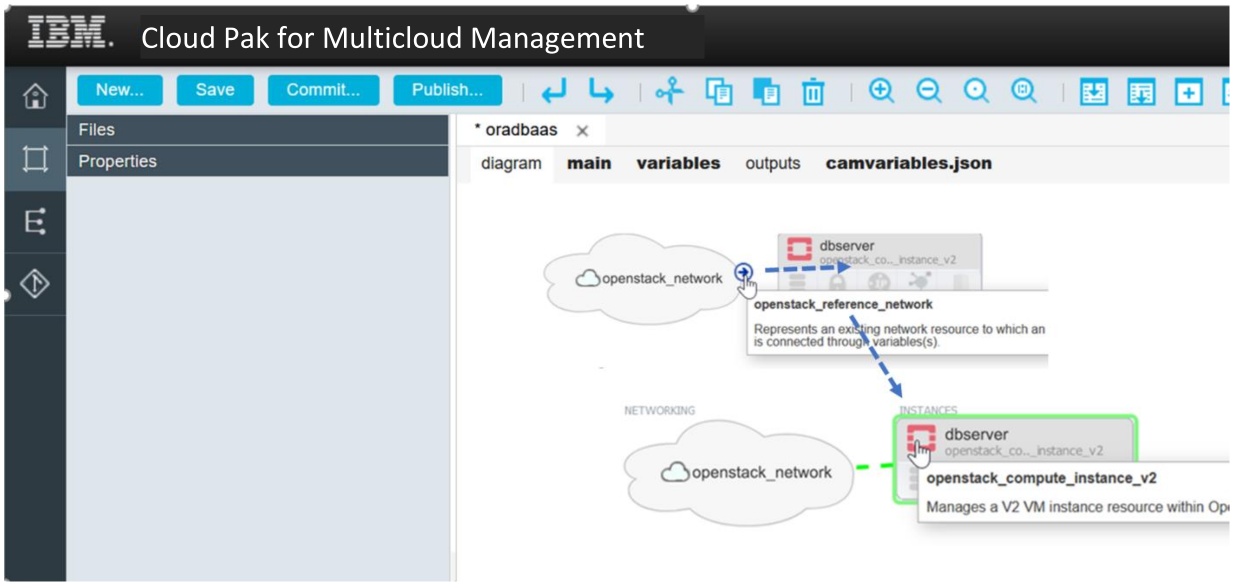

IBM. Cloud Pak for Multicloud Management
admin
New... Save Commit... Publish...
Files
Properties
* oradbaas
diagram main variables outputs camvariables.json
1 - Click Networks
2 - Select and drag an OpenStack Reference Network onto the diagram area. OpenStack Reference Network window pops up.
Compo
Instances
Networks
AWS Reference Network
Azure Reference Network
IBM Cloud Reference Network
OpenStack Reference Network
VMware Reference Network
AWS Virtual Private Cloud
Azure Virtual Network
OpenStack Network
INSTANCES
dbserver
openstack_co..._instance_v2
OK
Cancel

Template Key
openstack_compute_instance_v2.dbserver
name
var.dbserver_name
flavor_name
var.openstack_flavor_name
image_name
var.openstack_image_name
key_pair
openstack_compute_keypair_v

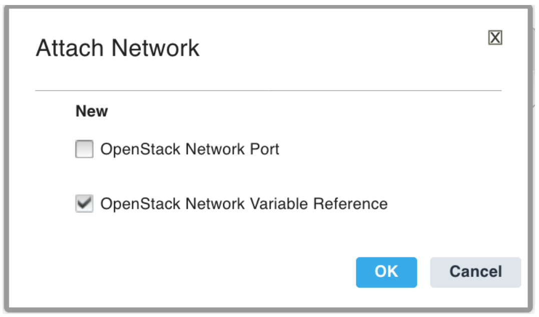
OpenStack Reference Network

Name openstack_network

Next hover your cursor over the **openstack_network** figure and point to the right arrow when it appears (a gloved hand appears, not shown on the screen capture), hold down the left mouse button and drag to the dbserver instance. Move the cursor around until the dbserver instance box's border turns green, as shown below, and then release the mouse button.


IBM. Cloud Pak for Multicloud Management
New... Save Commit... Publish...
Files
Properties
* oradbaas
diagram main variables outputs camvariables.json
openstack_network
dbserver
openstack_co..._instance_v2
openstack_reference_network
Represents an existing network resource to which an is connected through variables(s).
NETWORKING
INSTANCES
openstack_network
dbserver
openstack_co..._instance_v2
openstack_compute_instance_v2
Manages a V2 VM instance resource within Op

The Attach Network window pops up. **Check** OpenStack Network Variable Reference and click OK.



Attach Network

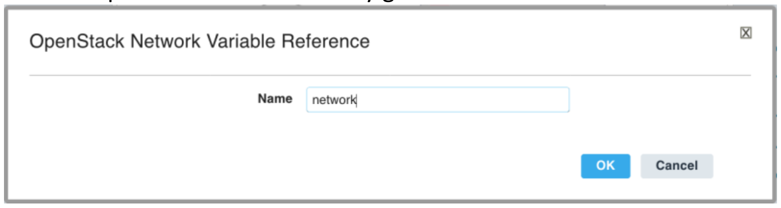
New

☐ OpenStack Network Port

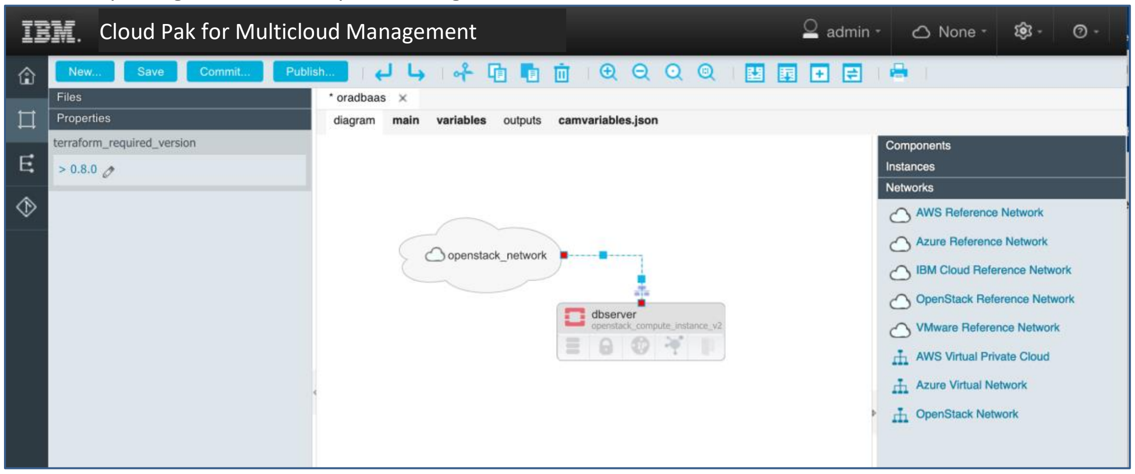
☑ OpenStack Network Variable Reference

OK Cancel

In the next pop-up window, OpenStack Network Variable Reference, set Name to **network** for this exercise. This value becomes part of the automatically generated variable names! Then click OK.



After completing the above steps, the diagram on the canvas looks like this:

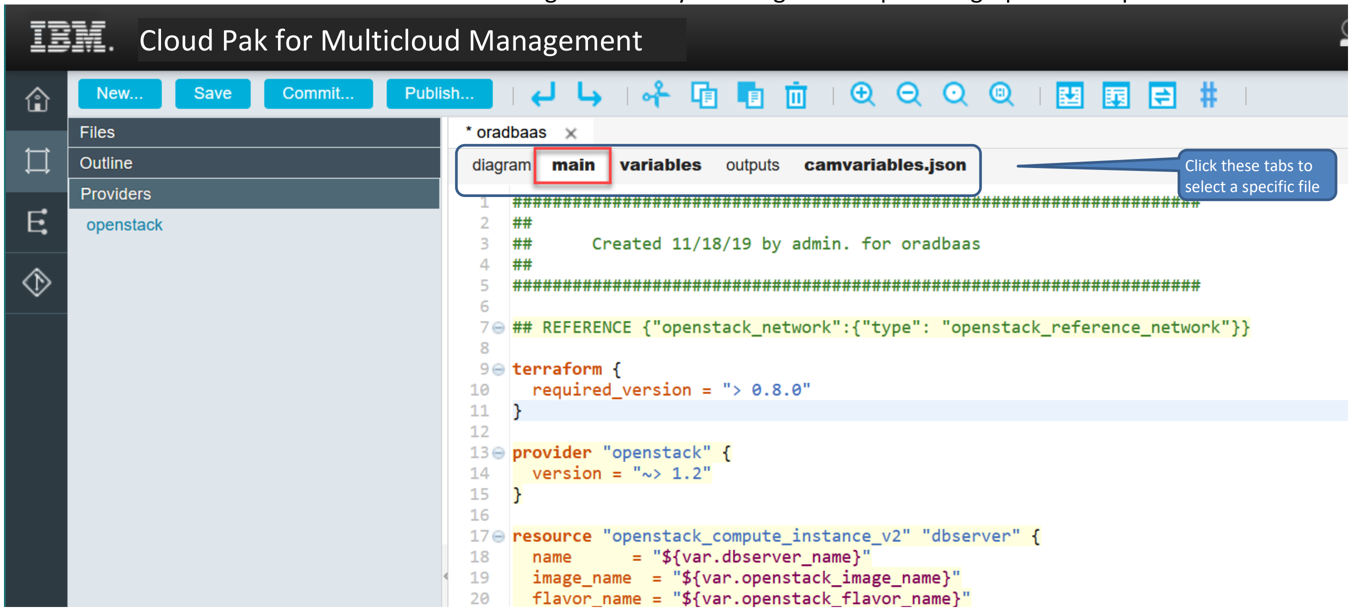


At this point Template Designer has updated the files defining this template, three terraform files (*.tf) and two metadata files (*.json).

For more information about the template structure, refer to:
https://www.ibm.com/support/knowledgecenter/en/SS2L37_3.2.1.0/cam_struct_template.html

The next step is to modify these files to suit our specific requirements.

Click the **main** tab and review the terraform code generated by the drag-and-drop of the graphical components.



```
############################################################################
##
##       Created 11/18/19 by admin. for oradbaas
##
############################################################################

## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
  required_version = "> 0.8.0"
}

provider "openstack" {
  version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbserver" {
  name        = "${var.dbserver_name}"
  image_name  = "${var.openstack_image_name}"
  flavor_name = "${var.openstack_flavor_name}"
```

We will first modify the main.tf file. Click the main tab as indicated in the picture above. Note that you can find a copy of the file in the appendix.

File: main.tf

Before

```
## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
 required_version = "> 0.8.0"
}

provider "openstack" {
   version = "~> 1.2"
}


resource "openstack_compute_instance_v2" "dbserver" {
 name     = "${var.dbserver_name}"                         ] 1
 image_name  = "${var.openstack_image_name}"
 flavor_name = "${var.openstack_flavor_name}"
 key_pair  = "${openstack_compute_keypair_v2.auth.id}"     ] 2
 network {
   uuid = "${var.openstack_network_id}"
 }
}

resource "tls_private_key" "ssh" {                         ]
   algorithm = "RSA"                                       ]
}                                                          ]
                                                           ] 2
resource "openstack_compute_keypair_v2" "auth" {           ]
   name = "${var.openstack_key_pair_name}"                 ]
   public_key = "${tls_private_key.ssh.public_key_openssh}" ]
}
```

After

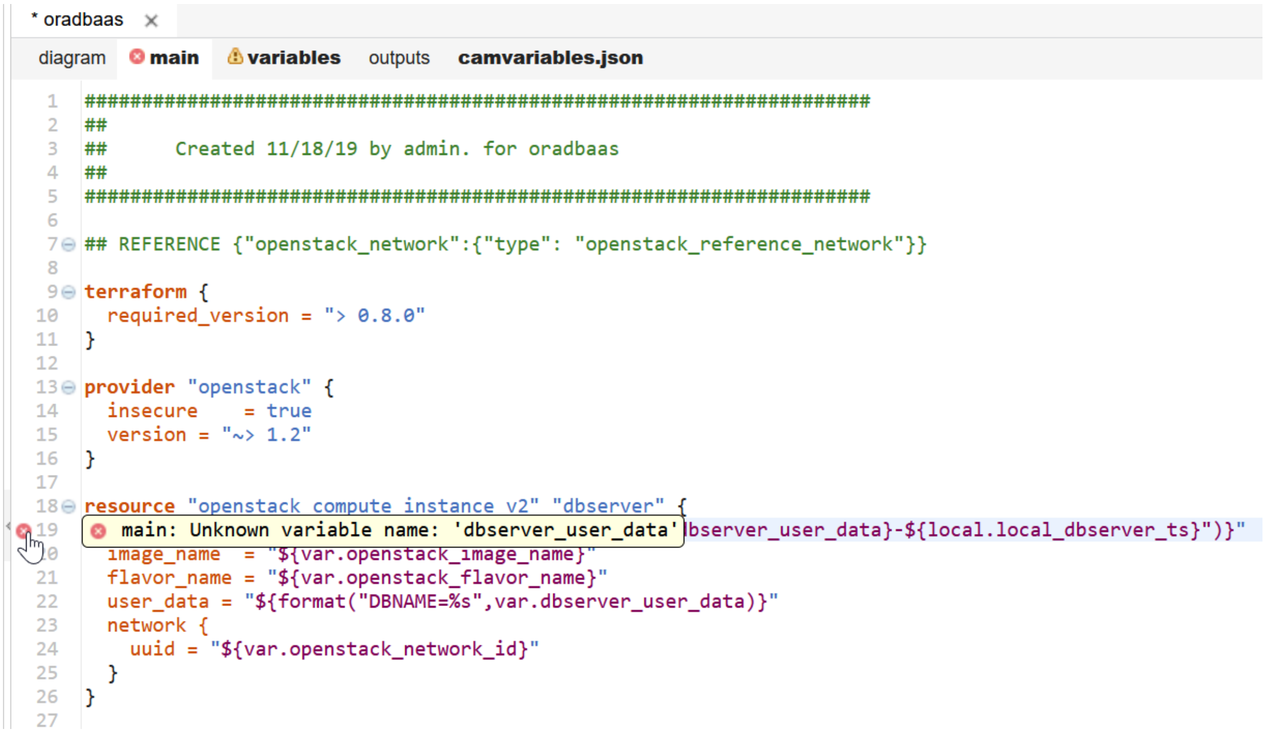
```
## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
 required_version = "> 0.8.0"
}

provider "openstack" {
 insecure    = true                  3
 version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbserver" {
 name     = "${format("${var.dbserver_name}-${var.dbserver_user_data}-        1
${local.local_dbserver_ts}")}"
 image_name  = "${var.openstack_image_name}"
 flavor_name = "${var.openstack_flavor_name}"
 user_data = "${format("DBNAME=%s",var.dbserver_user_data)}"    4
 network {
   uuid = "${var.openstack_network_id}"
 }
}
```

Notes:

1. This is the VM name in PowerVC. We will append dbserver_user_data and a time stamp to the instance name to make the VM name unique in PowerVC.
   - dbserver_user_data is an input variable used to hold the Oracle dbname
   - the time stamp is calculated in a local variable; see variables.tf.
   - Note that this is all 1 line!
2. Delete tls_private_key and key_pair related statements. These will not be used.
3. Add line to provider stanza.
4. Add user_data to pass cloud-config directives to Openstack cloud-init. Here we pass a text string containing the Oracle Database name (dbname) to the deployed instance in the form “DBNAME=<name>”. The deployed oradbaas image will parse this string to determine the name of the database to create.

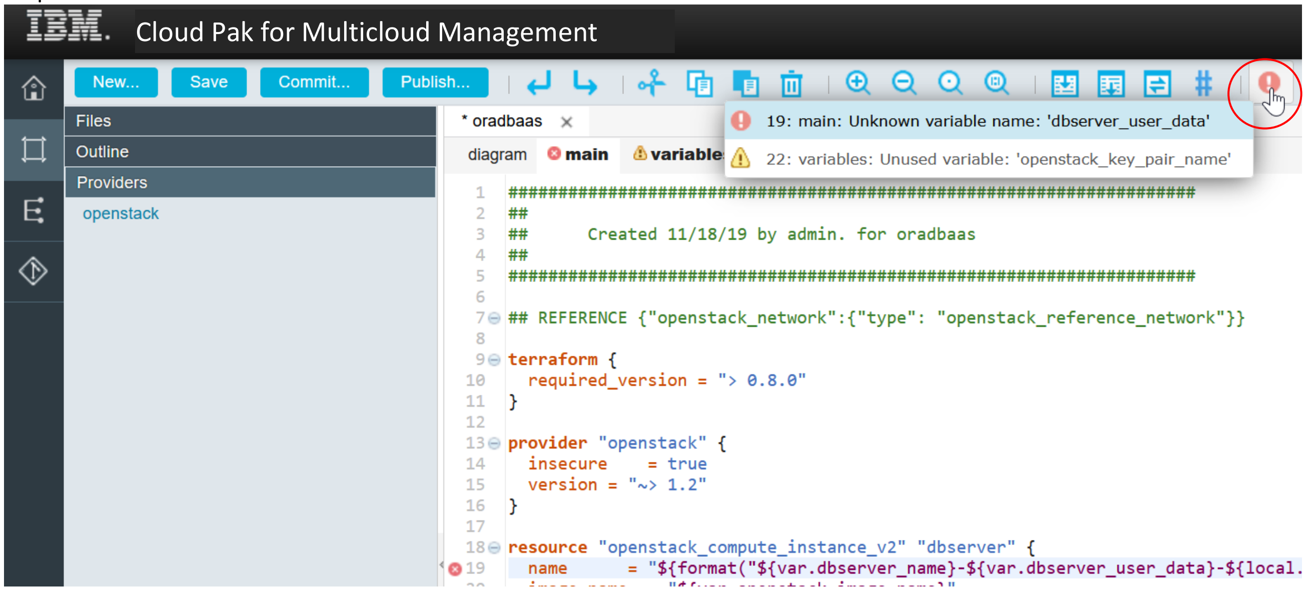
Refer to https://www.terraform.io/docs/providers/openstack/r/compute_instance_v2.html for further information on the openstack_compute-instance_v2 resource.

Note that once the above changes are made to main.tf, the Template Designer GUI will indicate errors and warnings as shown below. That is expected!



```
#####################################################################
##
##      Created 11/18/19 by admin. for oradbaas
##
#####################################################################

## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
  required_version = "> 0.8.0"
}

provider "openstack" {
  insecure     = true
  version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbserver" {
 main: Unknown variable name: 'dbserver_user_data' lbserver_user_data}-${local.local_dbserver_ts}")}"
  image_name  = "${var.openstack_image_name}"
  flavor_name = "${var.openstack_flavor_name}"
  user_data = "${format("DBNAME=%s",var.dbserver_user_data)}"
  network {
    uuid = "${var.openstack_network_id}"
  }
}
```

Hover your pointer over the red exclamation mark symbol as shown below to see all current errors and warnings. The next step is to correct variables.tf.



```
19: main: Unknown variable name: 'dbserver_user_data'
22: variables: Unused variable: 'openstack_key_pair_name'
```

```
#####################################################################
##
##      Created 11/18/19 by admin. for oradbaas
##
#####################################################################

## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
  required_version = "> 0.8.0"
}

provider "openstack" {
  insecure     = true
  version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbserver" {
  name       = "${format("${var.dbserver_name}-${var.dbserver_user_data}-${local.
```

Click on the **variables** tab and modify the contents as shown below.

File: variables.tf

Before:

```
variable "dbserver_name" {
  type = "string"
  description = "Generated"
}

variable "openstack_image_name" {
  type = "string"
  description = "Generated"
}

variable "openstack_flavor_name" {
  type = "string"
  description = "Generated"
}

variable "openstack_key_pair_name" {          1
  type = "string"
  description = "Generated"
}

variable "openstack_network_id" {
  type = "string"
  description = "Generated"
}
```

After:

```
variable "dbserver_name" {
  type = "string"
  description = " db server VM name prefix. Oracle dbname and a
timestamp will be appended in the PowerVC VM name."                2
}

variable "openstack_image_name" {
  type = "string"
  description = "PowerVC image name"    2
}

variable "openstack_flavor_name" {
  type = "string"
  description = "flavor name, or PowerVC compute template)"    2
}

variable "openstack_network_id" {
  type = "string"
  description = "Network id in PowerVC"    2
}

variable "dbserver_user_data" {          3
  type = "string"
  description = "Oracle DB name"
}

locals {                                                           4
  local_dbserver_ts =
"${replace(replace(replace(substr(timestamp(),2,17),"-
",""),"T","_"),":","")}"
}
```

Notes:

These are variable definitions used in main.tf and must be declared, removed , and made consistent with what's in main.tf.

1. Deleted. No longer needed since they were deleted in main.tf.
2. Change the descriptions as appropriate.
3. Added. Cloud-init cloud-config. We use this variable to pass the desired Oracle dbname to the deployed PowerVC VM.
4. Added. Local variable used to calculate and hold a timestamp. Our timestamp is formatted to the YYMMDD_hhmmss` format. Note that the content between '{' and '}' is all one line with no line breaks! For information on:
   - local variables: https://www.terraform.io/docs/configuration/locals.html
   - built-in functions (timestamp): https://www.terraform.io/docs/configuration/functions.html

Now switch to the tab **outputs** to modify the file: outputs.tf
The original Template Designer-generated outputs.tf is empty except for a comment block. The following will be added.

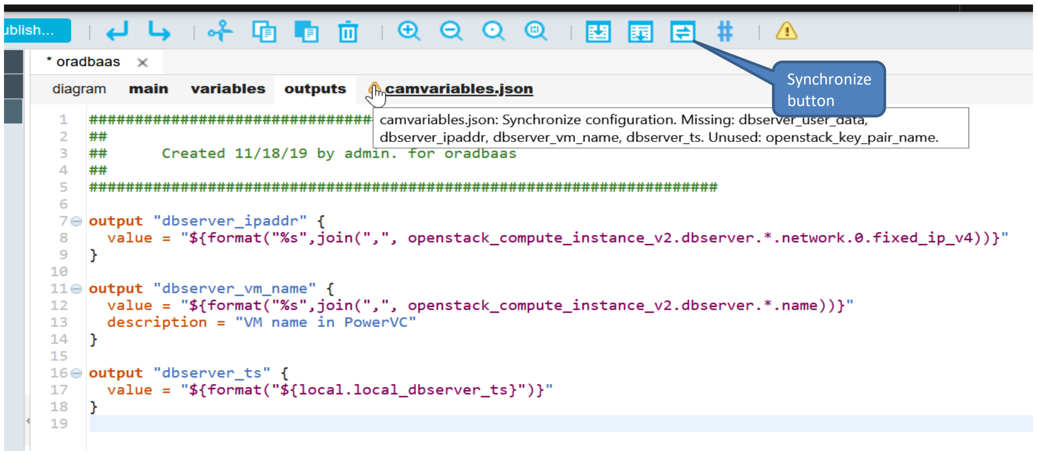
File: outputs.tf

```
output "dbserver_ipaddr" {
  value = "${format("%s",join(",",                                  1
openstack_compute_instance_v2.dbserver.*.network.0.fixed_ip_v4))}"
}

output "dbserver_vm_name" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.name))}"   2
  description = "VM name in PowerVC"
}

output "dbserver_ts" {
  value = "${format("${local.local_dbserver_ts}")}"   3
}

output "dbname" {
  value = "${var.dbserver_user_data}"   4
}
```

Notes:

1. The IP address is assigned by PowerVC and therefore not known until the instance is deployed. This output variable will be populated after the VM is deployed and is used to display the address on the user interface. This output variable is also referenced in the orchestrated service .
2. The VM name is dynamically created at deploy time, including a time stamp, and this variable will store that VM name for display in the user interface.
3. As a stand-alone template, the local timestamp may not be useful. However, we declare it as an output variable in order to have it available as input to the dbclient template deployment. This will be utilized when we create our service which orchestrates the co-deployment of the dbserver and the dbclient.
4. To be able to easily reference the database name in an orchestrated service deployment we make it available as an output variable as well.

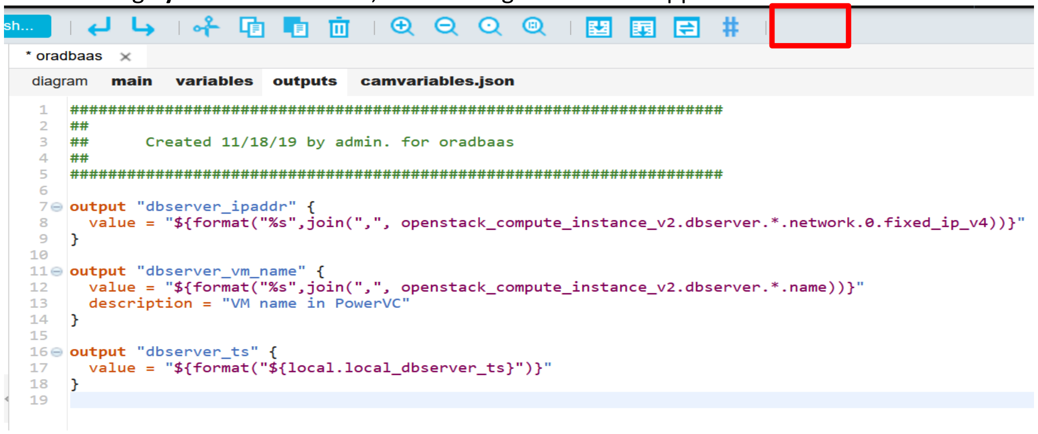
Now the only warning indicator left is against Camvariables.json. Click the **Synchronize** or **Save** button to synchronize the configuration. Note that the following two screen shots do not show the full content of Camvariables.json.



camvariables.json: Synchronize configuration. Missing: dbserver_user_data, dbserver_ipaddr, dbserver_vm_name, dbserver_ts. Unused: openstack_key_pair_name.

```
###############################
##
##      Created 11/18/19 by admin. for oradbaas
##
######################################################################

output "dbserver_ipaddr" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.network.0.fixed_ip_v4))}"
}

output "dbserver_vm_name" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.name))}"
  description = "VM name in PowerVC"
}

output "dbserver_ts" {
  value = "${format("${local.local_dbserver_ts}")}"
}

```

After clicking **Synchronize** or **Save**, the warning indicator disappears.



```
#####################################################################
##
##      Created 11/18/19 by admin. for oradbaas
##
#####################################################################

output "dbserver_ipaddr" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.network.0.fixed_ip_v4))}"
}

output "dbserver_vm_name" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.name))}"
  description = "VM name in PowerVC"
}

output "dbserver_ts" {
  value = "${format("${local.local_dbserver_ts}")}"
}

```

The file Camvariables.json contains metadata that describe what and how variables appear in the T&SA user interface. To suite our preferences, we modify **Camvariables.json**.

1. Modify the input groups.

Template Designer defined four input_groups and no output_groups. As we don't have many input or output variables we simplified the configuration to a single input and output group each. Note that below shows only the top code section in the file.

Before

```
{
  "input_datatypes": [ ],
  "input_namespaces": [ ],
  "output_namespace": "",
  "input_groups": [
    {
      "name": "Instances-
openstack_compute_instance_v2.dbserver",
      "label": "Instances -
openstack_compute_instance_v2.dbserver"
    },
    {
      "name": "Other",
      "label": "Other"
    },
    {
      "name": "instances",
      "label": "Instances"
    },
    {
      "name":
"openstack_compute_instance_v2_networks",
      "label":
"openstack_compute_instance_v2 Networks"
    }
  ],
  "output_groups": [ ],
```

After

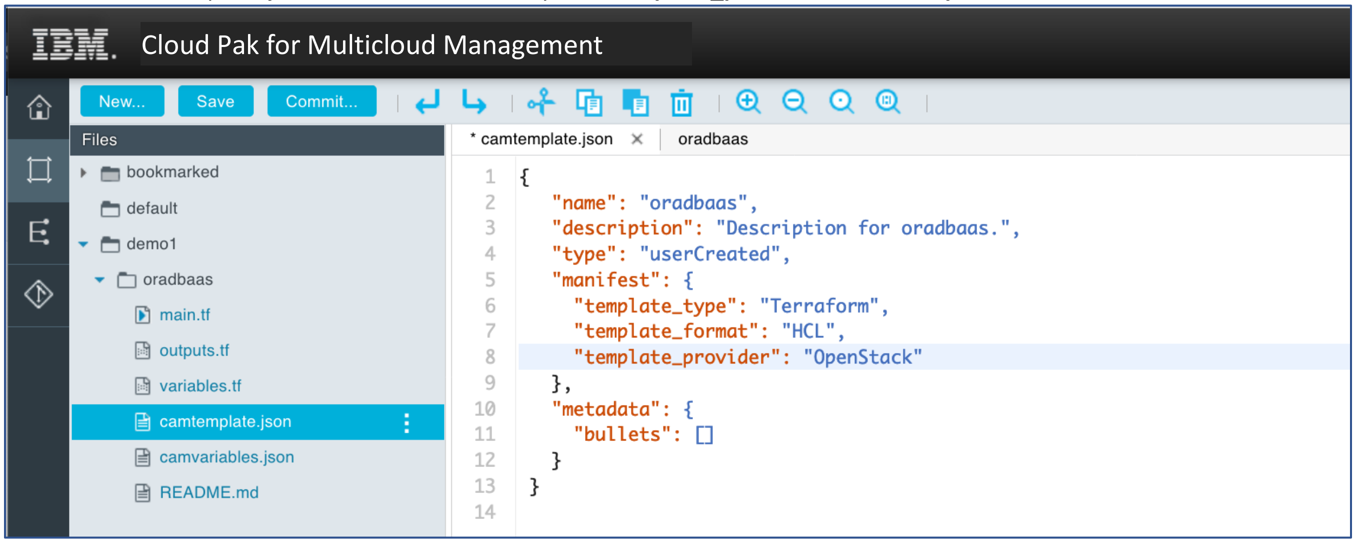
```
{
  "input_datatypes": [ ],
  "input_namespaces": [ ],
  "output_namespace": "",
  "input_groups": [
    {
      "name": "Input-parameters",
      "label": "Input parameters"
    }
  ],
  "output_groups": [
     {
      "name": "Output-parameters",
      "label": "Output parameters"
    }
  ],
```

2. Next, make the changes shown below to the following variables in each stanza as highlighted:
   - group_name,
   - label,
   - immutable_after_create, and
   - for **dbserver_ts** block in the **template_output_params** group, set hidden to true.
   - To illustrate default values, I specified RSDtest as default database name in the dbserver_user_data stanza.

When Camvariables.json was sync'd earlier, description labels were brought over from those specified in variables.tf. We adjusted them as well to be more descriptive.

```
 "template_input_params": [
    {
      "name": "dbserver_name",
      "label": "db server VM name prefix",
      "description": "db server VM name prefix. Oracle dbname and a timestamp will be
appended in the PowerVC VM name.",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_image_name",
      "label": "OpenStack Image Name",
      "description": "PowerVC image name",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_flavor_name",
      "label": "OpenStack Flavor Name",
      "description": "flavor name, or PowerVC compute template",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_network_id",
      "label": "OpenStack Network ID",
      "description": "Network id in PowerVC",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "dbserver_user_data",
      "label": "Oracle dbname",
      "description": "Oracle DB name",
      "type": "string",
      "default": "RSDtest",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    }
  ],
  "template_output_params": [
    {
      "name": "dbserver_ipaddr",
      "label": "dbserver IP address",
      "description": "dbserver IP address",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string"
    },
    {
      "name": "dbserver_vm_name",
      "label": "dbserver VM name",
      "description": "VM name in PowerVC",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string"
    },
    {
      "name": "dbserver_ts",
      "label": "dbserver ts",
      "description": "internal only - timestamp for output to dbclient",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": true,
      "shortname": "",
      "type": "string"
    }
  ]
```

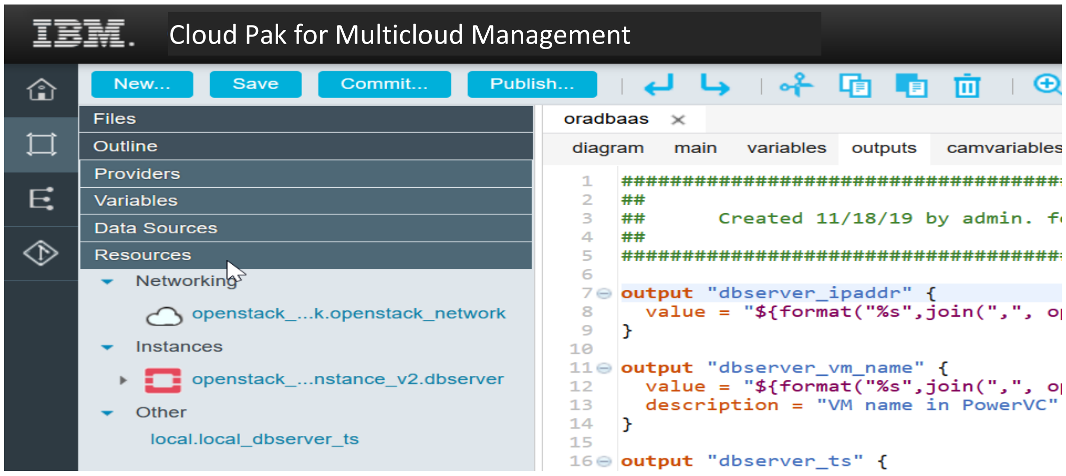
Next, click camtemplate.json under Files and verify that **template_provider** is set to **"OpenStack"** as shown below.



```
{
  "name": "oradbaas",
  "description": "Description for oradbaas.",
  "type": "userCreated",
  "manifest": {
    "template_type": "Terraform",
    "template_format": "HCL",
    "template_provider": "OpenStack"
  },
  "metadata": {
    "bullets": []
  }
}
```

Click the **Save** button to save your changes, making sure there are no warning or error indicators.

Note that you can drill down into the template details also by selection criteria like *Providers*, *Variables*, …

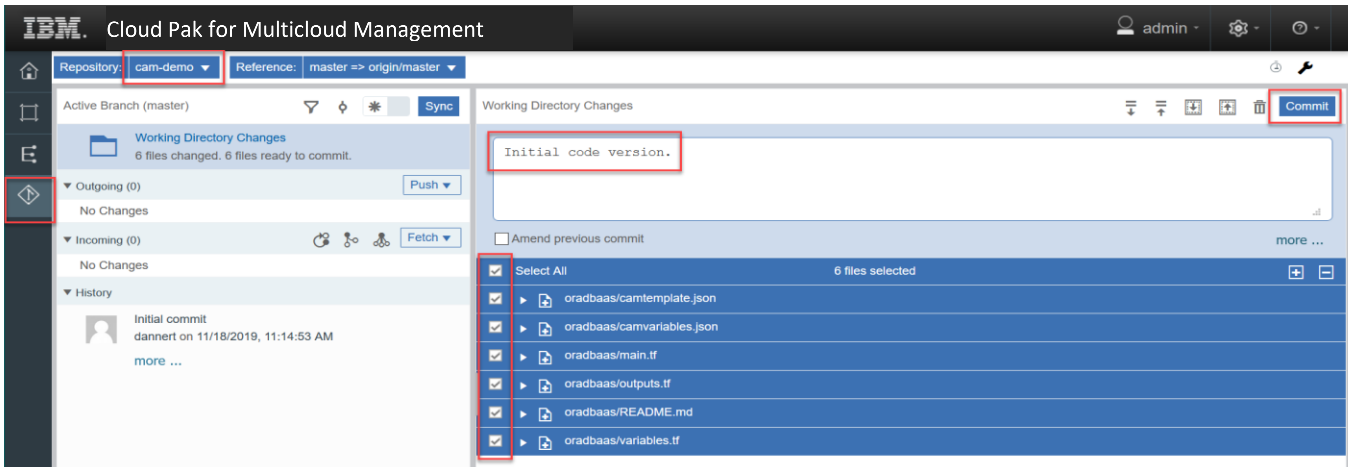

IBM. Cloud Pak for Multicloud Management
New... Save Commit... Publish...
Files
Outline
Providers
Variables
Data Sources
Resources
Networking
openstack_...k.openstack_network
Instances
openstack_...nstance_v2.dbserver
Other
local.local_dbserver_ts
oradbaas
diagram main variables outputs camvariables

```
#######################################
##
##      Created 11/18/19 by admin. f
##
#######################################

output "dbserver_ipaddr" {
  value = "${format("%s",join(",", o
}

output "dbserver_vm_name" {
  value = "${format("%s",join(",", o
  description = "VM name in PowerVC"
}

output "dbserver_ts" {
```

8.1.4 Commit *oradbaas* template to GitHub

After creating and modifying all the files defining the *oradbaas* template we now need to commit all those files back to the GitHub repository.

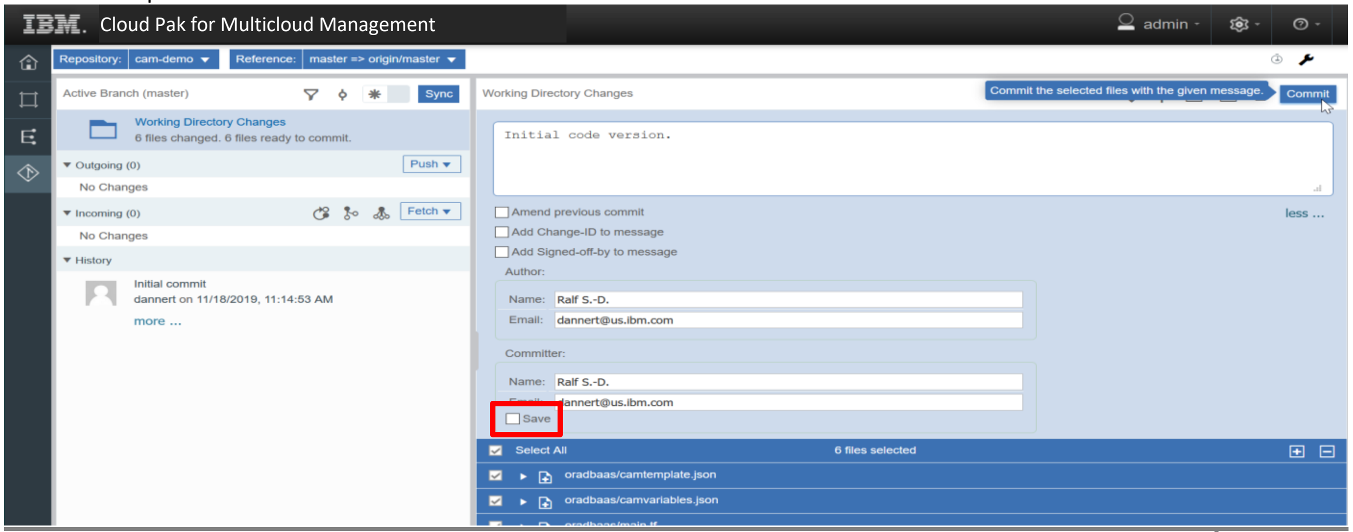
Return to the Repository menu via the lowest icon on the left-hand pane. **Ensure the correct repository is selected!** Enter a commit description, select all the files via Select All and finally click the Commit button.


IBM. Cloud Pak for Multicloud Management
admin
Repository: cam-demo
Reference: master => origin/master
Active Branch (master)
Sync
Working Directory Changes
6 files changed. 6 files ready to commit.
Outgoing (0)
Push
No Changes
Incoming (0)
Fetch
No Changes
History
Initial commit
dannert on 11/18/2019, 11:14:53 AM
more ...
Working Directory Changes
Commit
Amend previous commit
more ...
Select All
6 files selected

Initial code version.

- oradbaas/camtemplate.json
- oradbaas/camvariables.json
- oradbaas/main.tf
- oradbaas/outputs.tf
- oradbaas/README.md
- oradbaas/variables.tf

Fill in the name and email address of the author and committer, then click the Commit button. You can check the Save check box to preserve the Author and Committer name for future commits.


IBM. Cloud Pak for Multicloud Management
admin
Repository: cam-demo
Reference: master => origin/master
Active Branch (master)
Sync
Working Directory Changes
6 files changed. 6 files ready to commit.
Outgoing (0)
Push
No Changes
Incoming (0)
Fetch
No Changes
History
Initial commit
dannert on 11/18/2019, 11:14:53 AM
more ...
Working Directory Changes
Commit the selected files with the given message.
Commit
Amend previous commit
Add Change-ID to message
Add Signed-off-by to message
less ...
Save
Select All
6 files selected

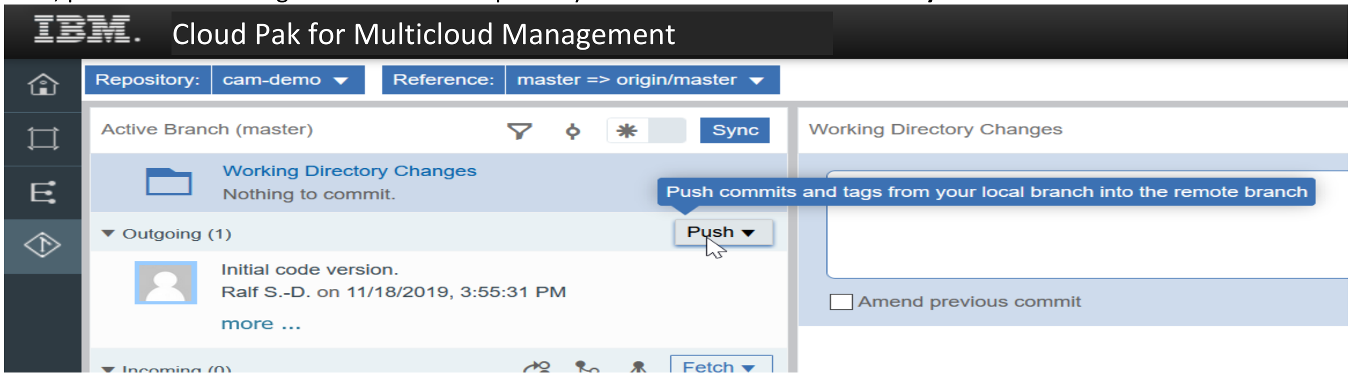
Initial code version.

Author:
Name: Ralf S.-D.
Email: dannert@us.ibm.com

Committer:
Name: Ralf S.-D.
Email: dannert@us.ibm.com

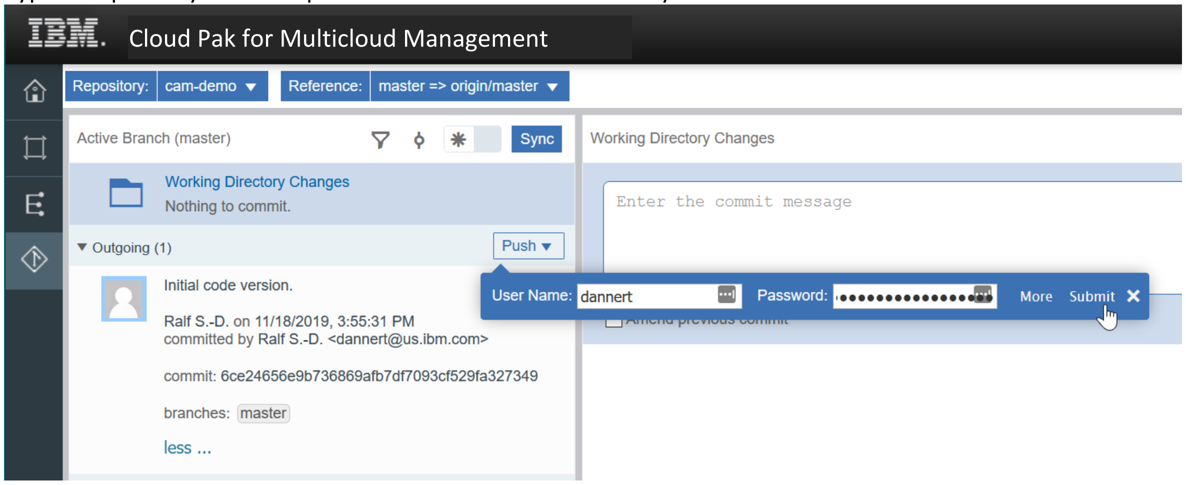
- oradbaas/camtemplate.json
- oradbaas/camvariables.json
- oradbaas/main.tf

Now, push the local changes to the GitHub repository. Click the **Push** button or the **Sync** button.



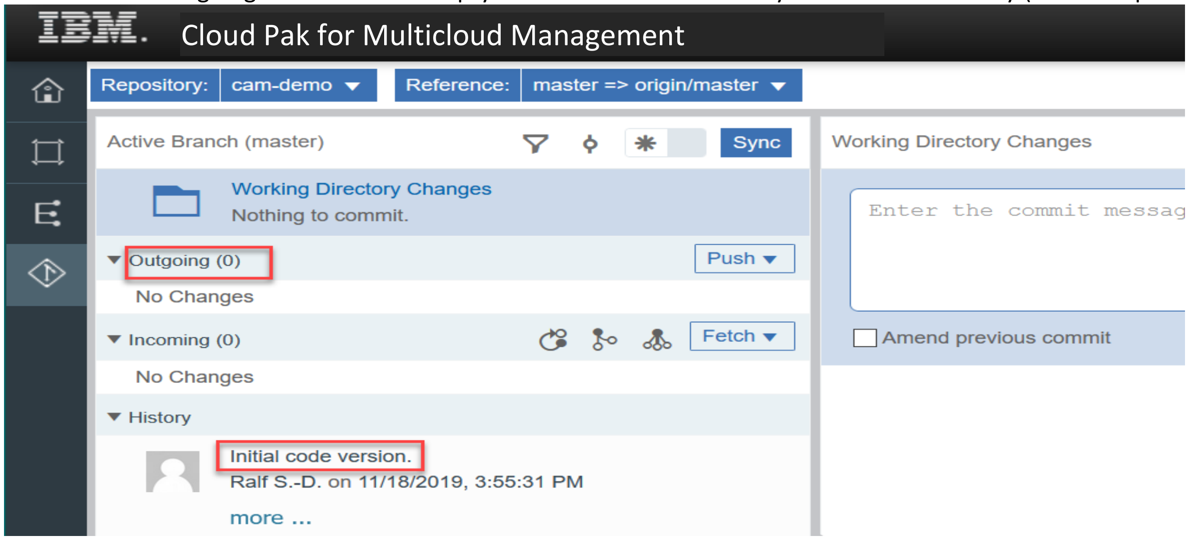
Cloud Pak for Multicloud Management
Repository: cam-demo
Reference: master => origin/master
Active Branch (master)
Sync
Working Directory Changes
Nothing to commit.
Push commits and tags from your local branch into the remote branch
Outgoing (1)
Push
Initial code version.
Ralf S.-D. on 11/18/2019, 3:55:31 PM
more ...
Amend previous commit
Fetch

Type in repository user and password. The GitHub token may be entered in the Password field. Click the **Submit** button.



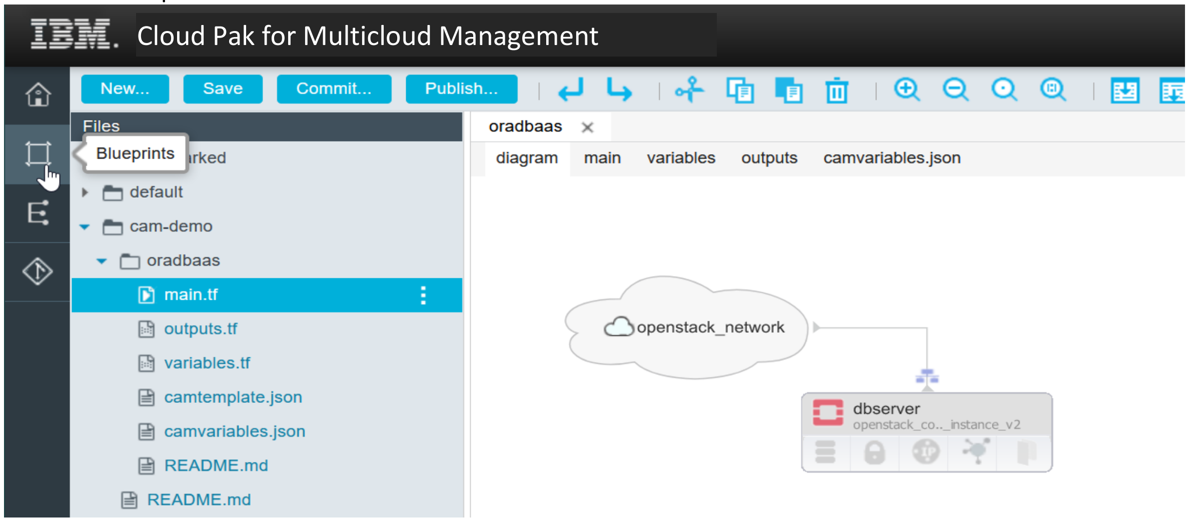
Cloud Pak for Multicloud Management
Repository: cam-demo
Reference: master => origin/master
Active Branch (master)
Sync
Working Directory Changes
Nothing to commit.
Enter the commit message
Outgoing (1)
Push
User Name: dannert
Password:
More
Submit
Initial code version.
Ralf S.-D. on 11/18/2019, 3:55:31 PM
committed by Ralf S.-D. <dannert@us.ibm.com>
commit: 6ce24656e9b736869afb7df7093cf529fa327349
branches: master
less ...

Note that the Outgoing section is now empty and the "Initial code' entry is now under History (left hand panel).

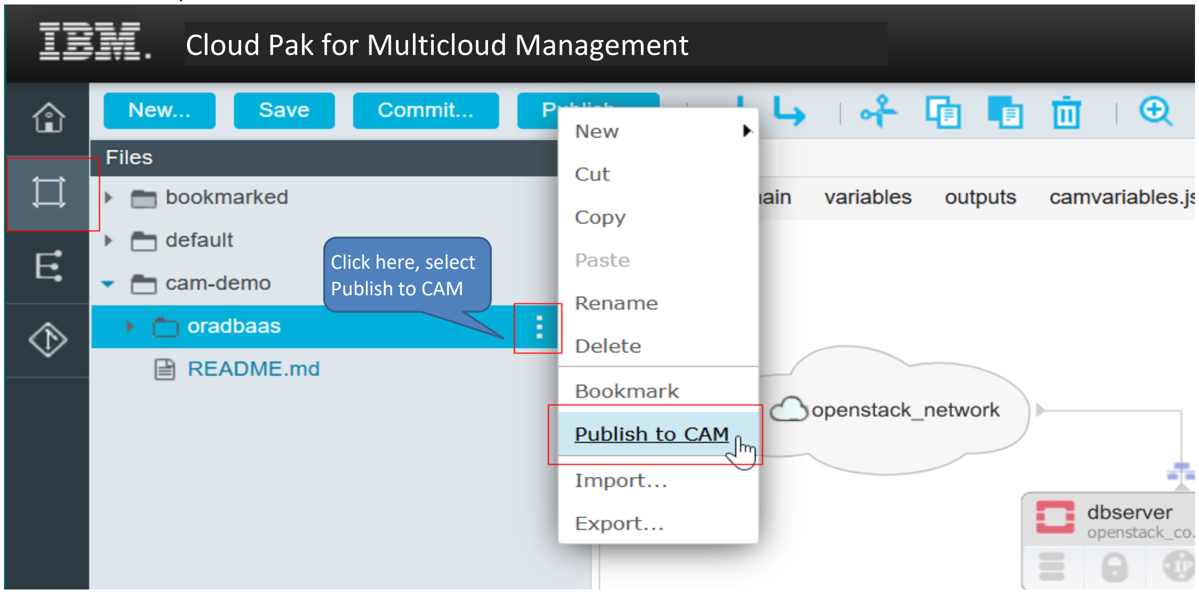


8.1.5 Publish the *oradbaas* template to Terraform & Service Automation

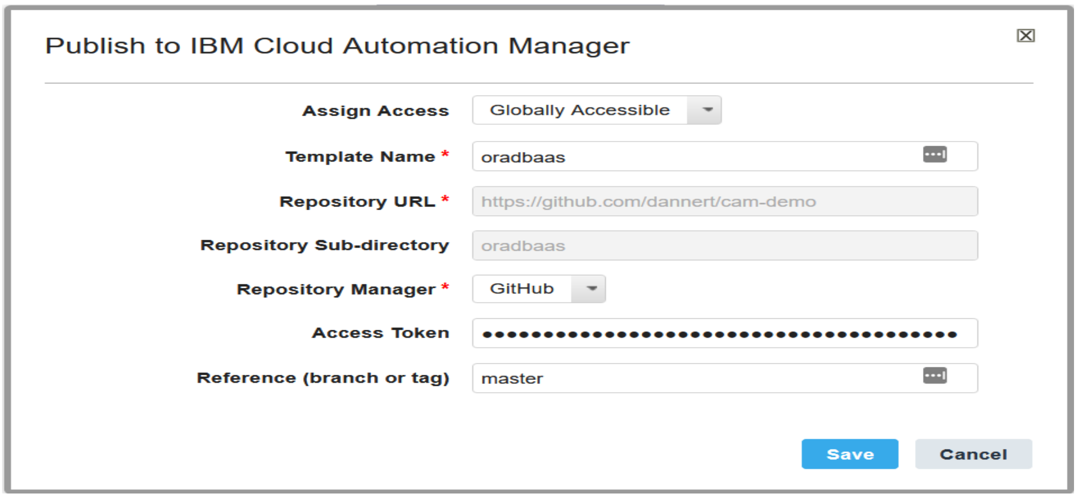
Return to Blueprints.



Select ***oradbaas***, then click the three-dot menu in the Action column. Select **Publish to T&SA**.

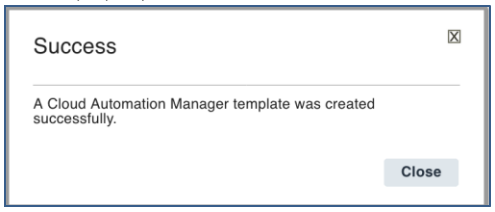


A pop-up window appears. Enter a template name *oradbaas* and the GitHub access token. Click **Save** button.


Publish to IBM Cloud Automation Manager

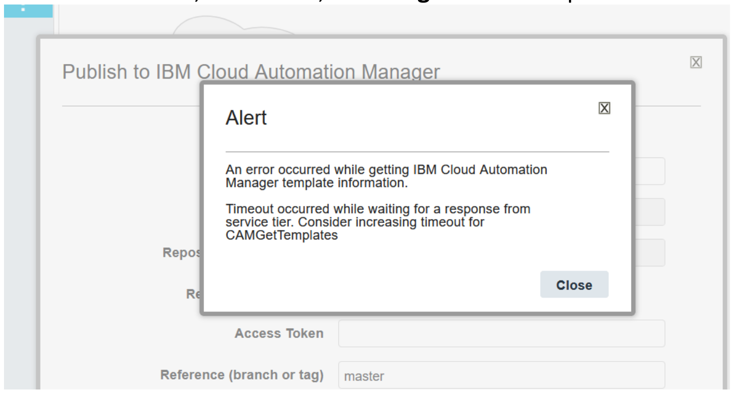
Assign Access	Globally Accessible
Template Name *	oradbaas
Repository URL *	https://github.com/dannert/cam-demo
Repository Sub-directory	oradbaas
Repository Manager *	GitHub
Access Token	●●
Reference (branch or tag)	master

A new pop-up window indicates successful creation of the T&SA template.


Success

A Cloud Automation Manager template was created successfully.

Note: In our testing we ran sporadically into challenges with the "Publish to CAM" functionality and it failed with the error screen below and, not visible, the **Assign Access** drop down field showed ERROR text.

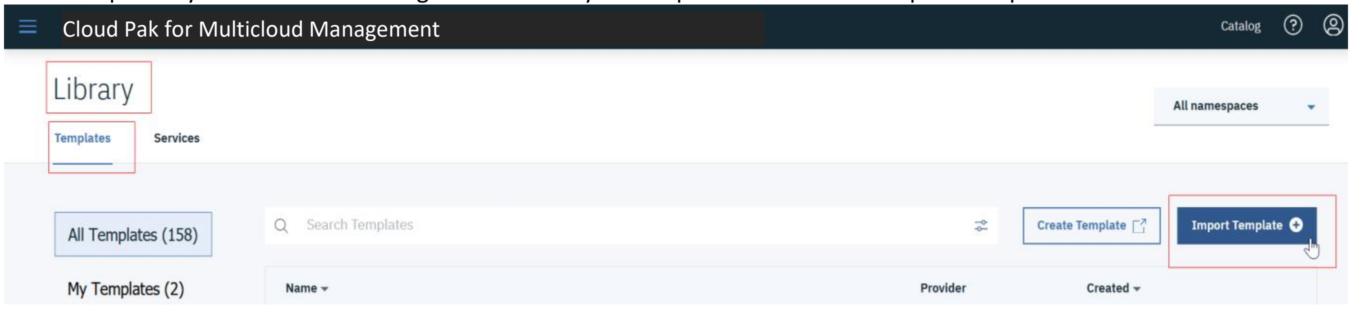

Publish to IBM Cloud Automation Manager

Alert

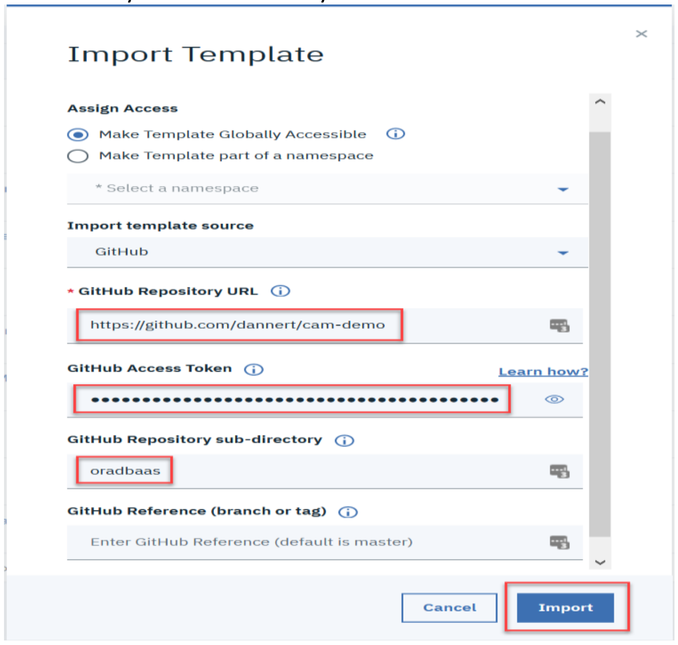
An error occurred while getting IBM Cloud Automation Manager template information.

Timeout occurred while waiting for a response from service tier. Consider increasing timeout for CAMGetTemplates

As workaround we utilized instead the **Import Template** functionality in T&SA to import the template definition from the GitHub repository into T&SA. In T&SA go to the Library -> Templates and click on Import Templates.



In the new pop-up window provide the relevant information to access GitHub and select the correct template, *oradbaas*, as indicated by the sub-directory we choose.


Import Template

Assign Access
- Make Template Globally Accessible (selected)
- Make Template part of a namespace

* Select a namespace

Import template source: GitHub

* GitHub Repository URL: https://github.com/dannert/cam-demo

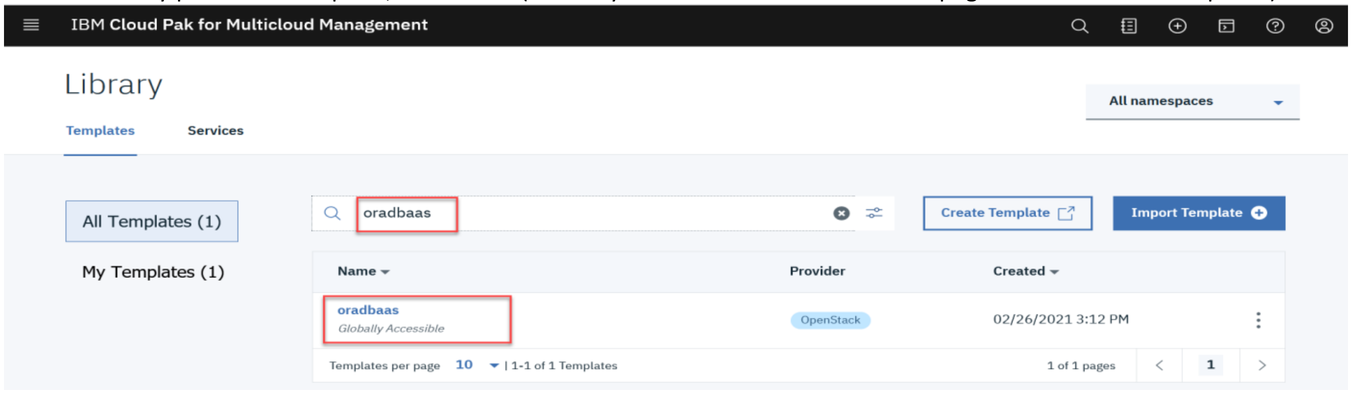
GitHub Access Token: ••

GitHub Repository sub-directory: oradbaas

GitHub Reference (branch or tag): Enter GitHub Reference (default is master)

After the Publish to T&SA or Import Template in T&SA go the Template Library.

Note the newly published template, *oradbaas*. (You may need to refresh the browser page to see the new template.)

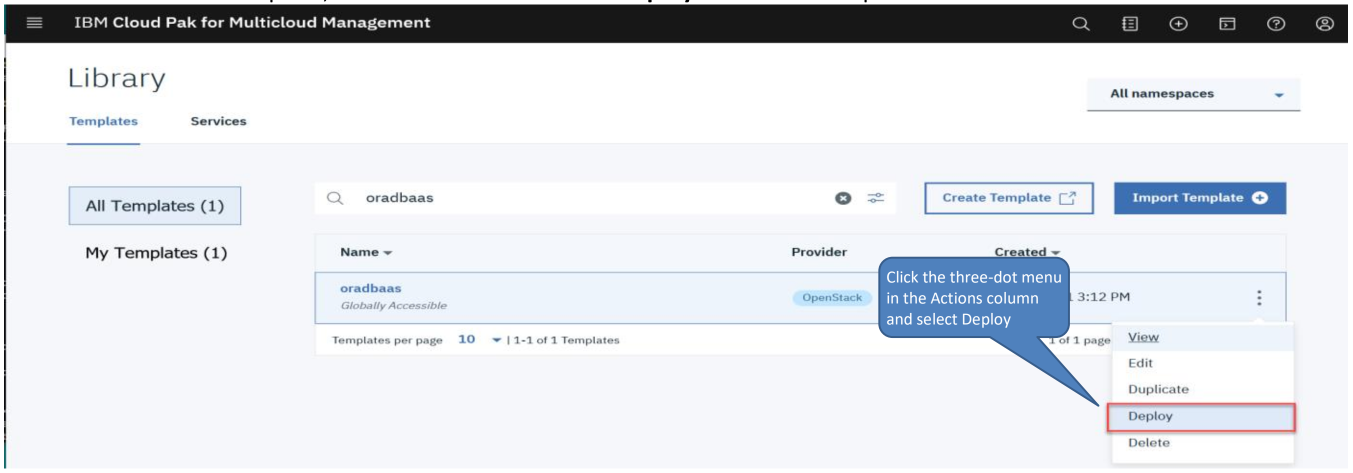


The new template can now be deployed in T&SA or be used in a service which can also be published to OCP catalog.

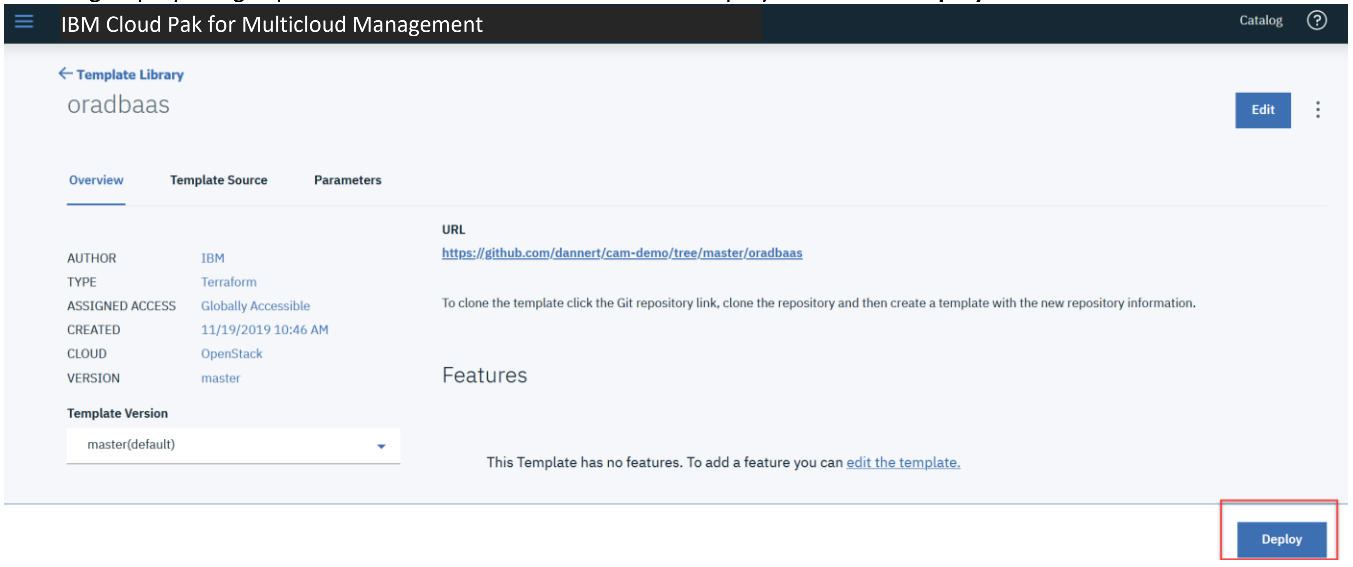
Note: The template Provider shown in the T&SA screen above shows **OpenStack**. This value is taken from **template_provider** in **camtemplate.json**. If this does not show Openstack then this can be corrected in T&SA by clicking the three-button menu on the right, selecting Edit and changing **Cloud Provider** to OpenStack.

8.2 Deploying the *oradbaas* template

Select the *oradbaas* template, click the menu and select **Deploy** to test the template.



Selecting Deploy brings up the screen below. To start the actual deploy click on the **Deploy** button.



AUTHOR	IBM
TYPE	Terraform
ASSIGNED ACCESS	Globally Accessible
CREATED	11/19/2019 10:46 AM
CLOUD	OpenStack
VERSION	master

Template Version: master(default)

URL
https://github.com/dannert/cam-demo/tree/master/oradbaas

To clone the template click the Git repository link, clone the repository and then create a template with the new repository information.

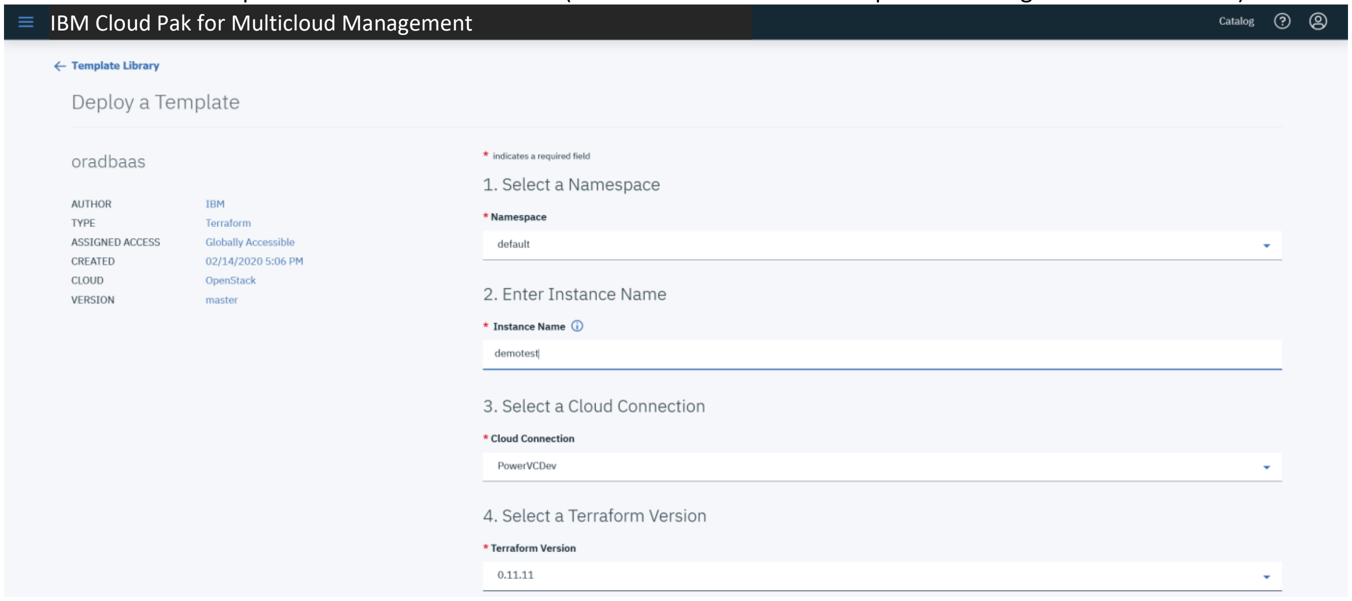
This Template has no features. To add a feature you can edit the template.

On the Deploy a Template screen, select the Namespace, e.g.: default.

Once Namespace is selected, other input parameters appear as shown on the screen below. Type in an **Instance Name**, e.g. demotest. This name is what will be listed under Deployed Instances in T&SA.

Note that our Cloud Connection, PowerVCDev has been filled in already (it's the only OpenStack connection we've defined).

Ensure the Terraform provider version is at 0.11.11 (0.12.13 fails with client and provider being at different levels!)



Deploy a Template

oradbaas

AUTHOR	IBM
TYPE	Terraform
ASSIGNED ACCESS	Globally Accessible
CREATED	02/14/2020 5:06 PM
CLOUD	OpenStack
VERSION	master

* indicates a required field

1. Select a Namespace

* Namespace: default

2. Enter Instance Name

* Instance Name: demotest

3. Select a Cloud Connection

* Cloud Connection: PowerVCDev

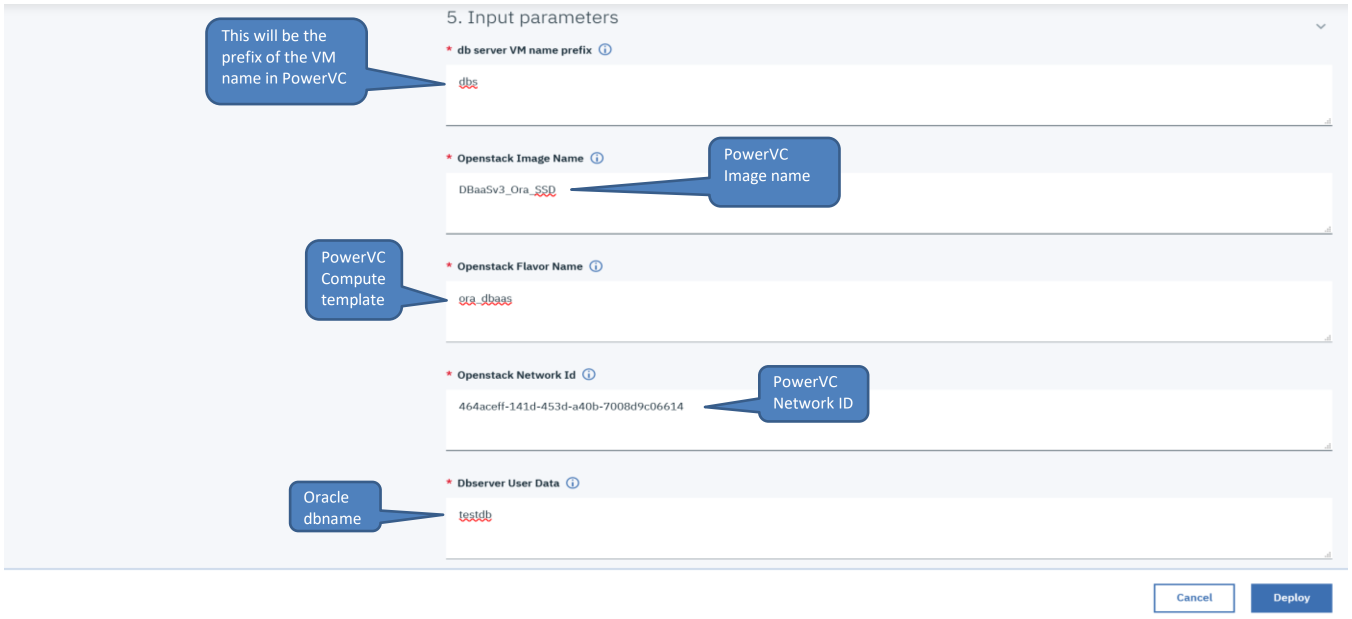
4. Select a Terraform Version

* Terraform Version: 0.11.11

Type in the Input Parameters as shown below.

Note:

- All our template inputs are grouped under **Input Parameters**, which is our one input group defined in camvariables.json.
- In this example, only the Oracle dbname field had a default value set – RSDtest – and therefore all other input parameters show as “”. Note that you can overwrite the specified default value as you can see below. For a real production environment, it probably would make sense to set default values for the other input fields as well. If desired, this can be done in the camvariables.json file.
- Image name, flavor name and network ID can be determined via PowerVC gui or CLI commands: openstack {image|flavor|network} list.



5. Input parameters

* db server VM name prefix: dbs

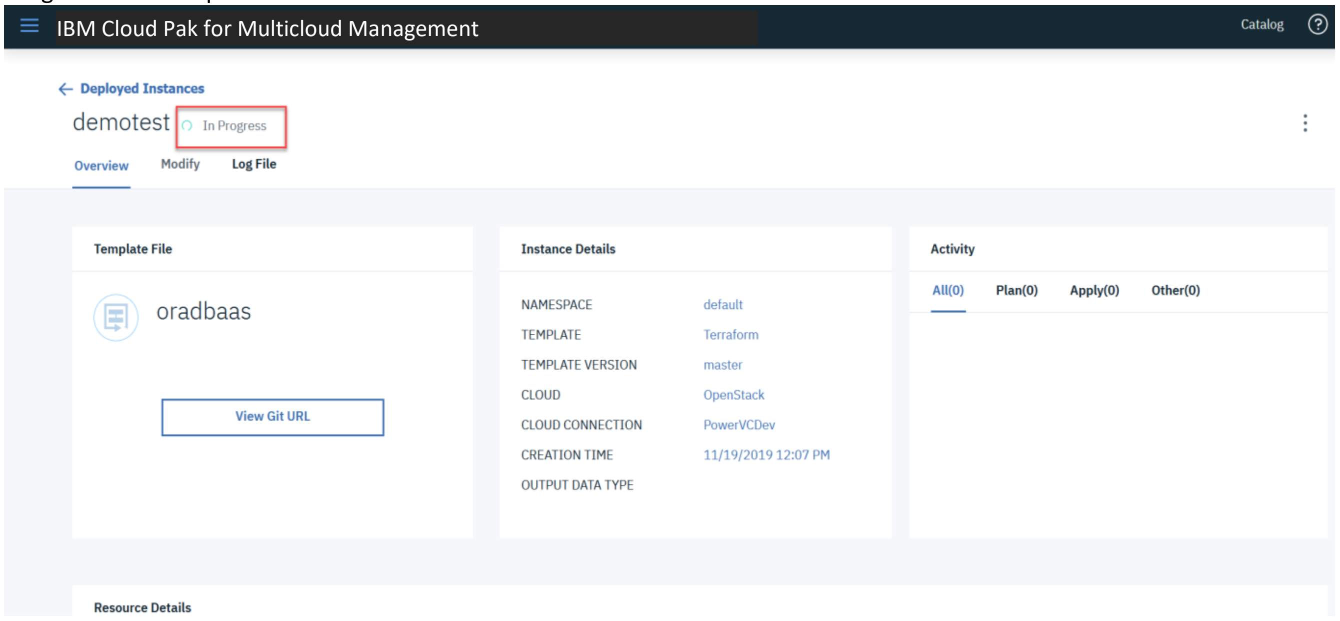
* Openstack Image Name: DBaaSv3_Ora_SSD

* Openstack Flavor Name: ora_dbaas

* Openstack Network Id: 464aceff-141d-453d-a40b-7008d9c06614

* Dbserver User Data: testdb

Click the **Deploy** button to deploy the template.
After clicking Deploy the following screen is displayed where you can observe the status of the deploy. Note the "In Progress" in the top left.


IBM Cloud Pak for Multicloud Management

Deployed Instances

demotest ○ In Progress

Overview Modify Log File

Template File

oradbaas

View Git URL

Instance Details

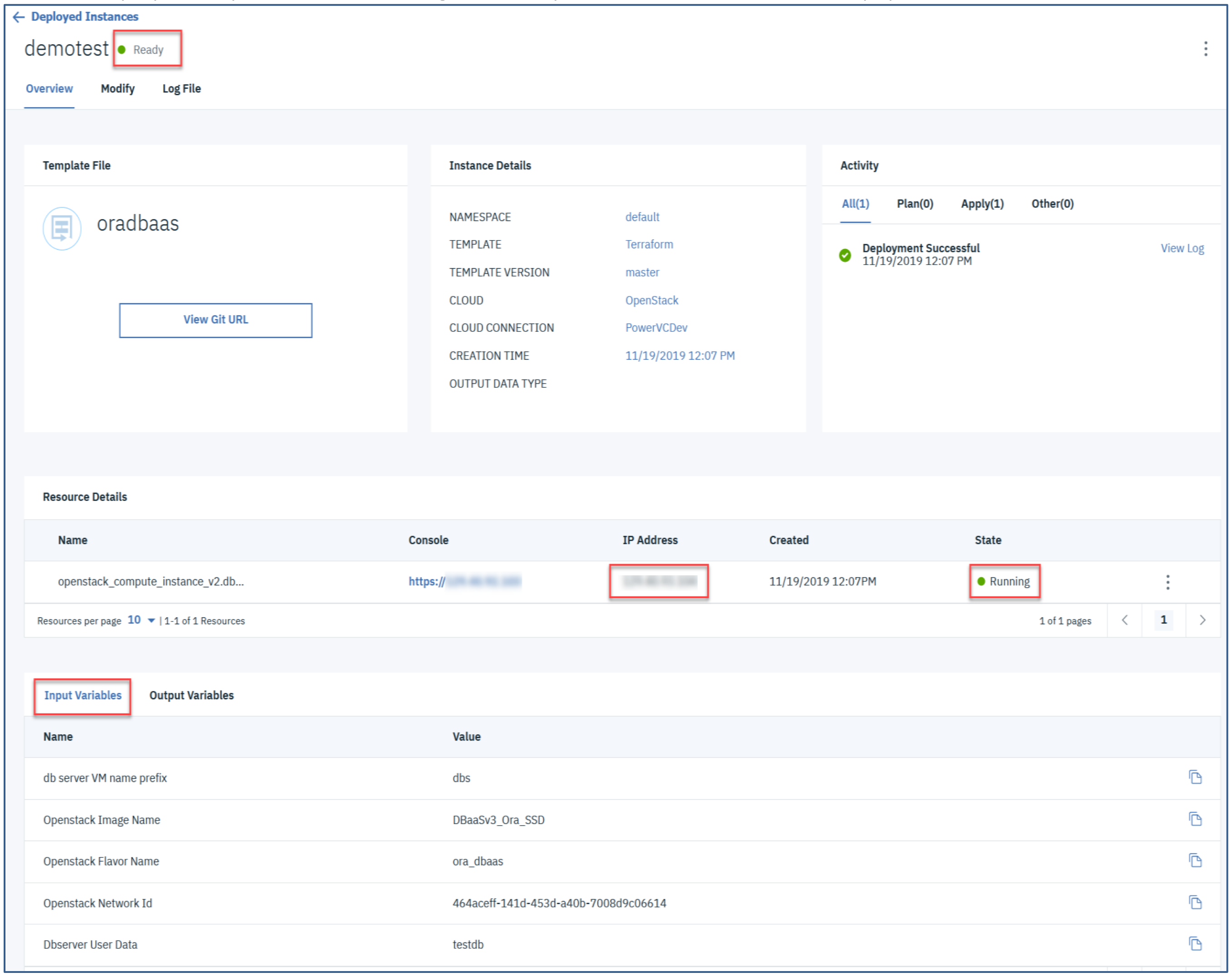
NAMESPACE	default
TEMPLATE	Terraform
TEMPLATE VERSION	master
CLOUD	OpenStack
CLOUD CONNECTION	PowerVCDev
CREATION TIME	11/19/2019 12:07 PM
OUTPUT DATA TYPE	

Activity

All(0) Plan(0) Apply(0) Other(0)

Resource Details

After the deploy is complete the status changes to Ready and VM related details are all populated.


Deployed Instances

demotest ● Ready

Overview Modify Log File

Template File

oradbaas

View Git URL

Instance Details

NAMESPACE	default
TEMPLATE	Terraform
TEMPLATE VERSION	master
CLOUD	OpenStack
CLOUD CONNECTION	PowerVCDev
CREATION TIME	11/19/2019 12:07 PM
OUTPUT DATA TYPE	

Activity

All(1) Plan(0) Apply(1) Other(0)

✓ Deployment Successful 11/19/2019 12:07 PM View Log

Resource Details

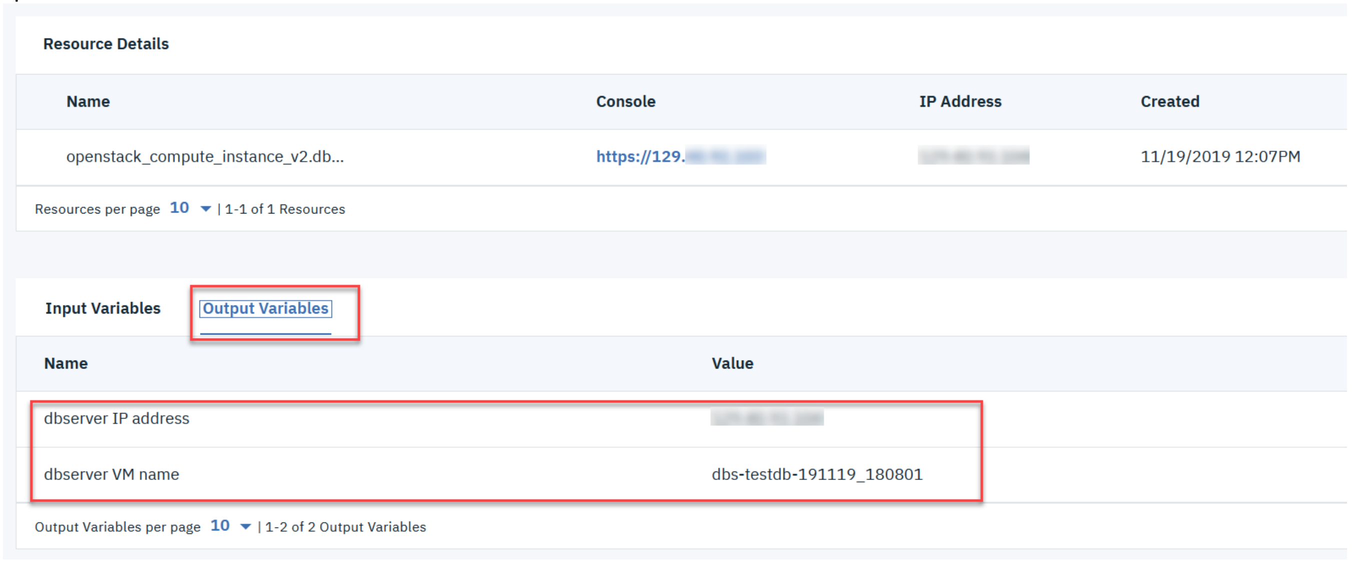
Name	Console	IP Address	Created	State
openstack_compute_instance_v2.db...	https://[redacted]	[redacted]	11/19/2019 12:07PM	● Running

Resources per page 10 | 1-1 of 1 Resources 1 of 1 pages 1

Input Variables Output Variables

Name	Value
db server VM name prefix	dbs
Openstack Image Name	DBaaSv3_Ora_SSD
Openstack Flavor Name	ora_dbaas
Openstack Network Id	464aceff-141d-453d-a40b-7008d9c06614
Dbserver User Data	testdb

To see the populated output variables, we specified in the "output parameters" section of the camvariables.json file of the *oradbaas* template, click **Output Variables**.
Note that the "hidden" output parameter, as expected, is not displayed. The VM image name was correctly constructed as specified.


Resource Details

Name	Console	IP Address	Created
openstack_compute_instance_v2.db...	https://129.		11/19/2019 12:07PM

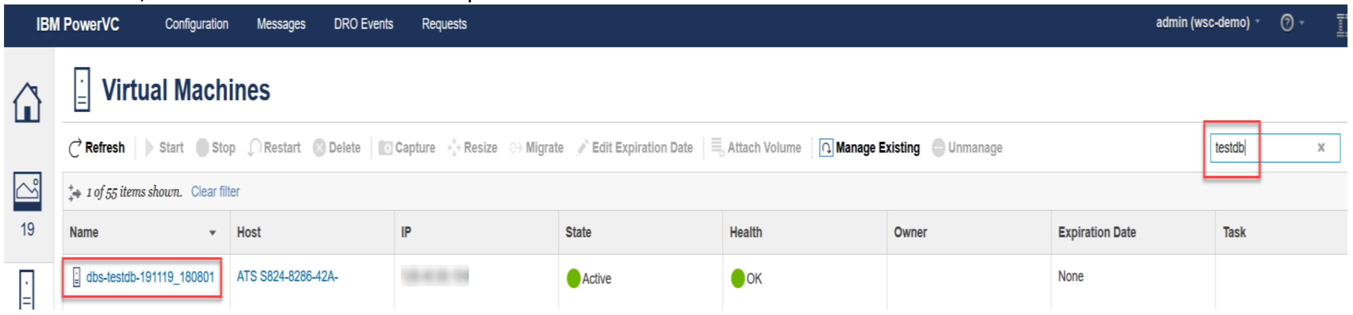
Resources per page 10 | 1-1 of 1 Resources

Input Variables | Output Variables

Name	Value
dbserver IP address	
dbserver VM name	dbs-testdb-191119_180801

Output Variables per page 10 | 1-2 of 2 Output Variables

On PowerVC, use the dbname or timestamp to filter the list of Virtual Machines.



Name	Host	IP	State	Health	Owner	Expiration Date	Task
dbs-testdb-191119_180801	ATS S824-8286-42A-		Active	OK		None	

8.3 *oraclient* Template Design

The *oraclient* template deploys a VM running Linux on Power. The PowerVC deployment image contains Red Hat Linux, configured with graphical user interface, VNC client and NFS client software, but there are no post-deployment scripts, or oracle / swingbench related software included in that image.

Important: This template is written to be co-deployed with the *oradbaas* template as it expects values like DB server IP and database name as parameter which are the result of a successful deploy of the *oradbaas* template.

This template illustrates how you can inject scripts, via cloud-init cloud-config, into a VM at deploy time and the use of a known NFS export to access required additional software to be installed via the injected post-processing scripts. This approach is very flexible as it does not need the creation of a new image in PowerVC every time a post-deployment script needs to be changed. The availability of the NFS server to stage the software is, of course, critical at deploy time.

In contrast, the PowerVC image used in the *oradbaas* template deployment included the oracle database software and several post-processing scripts to configure, create and start the oracle database at VM deployment. Cloud-init user data was only used to provide simple parameters like the Oracle database name to the post-deployment scripts.

Note: All template files for the *oraclient* template are available in the appendix of this document.

8.3.1 *oraclient* template

As mentioned above, the PowerVC deployment image does not include any client-specific post-deployment customization processing. Post-deployment performed by *oraclient* is done using cloud-init cloud-config which injects the following files into the deployed VM and executes several additional shell commands at deployment.

- Inject four script files using the write_files configuration statement. These four files are:
  - dbserver.cfg - contains export statements to set environment variables
  - sbsetup.sh - shell script to check availability of the database, then load the Swingbench Order Entry schema using Swingbench's oewizard in silent mode (non-GUI).
  - dropsoe.sh - shell script to drop the Swingbench Order Entry. Invokes the oewizard Java program in silent mode.
  - startsb.sh - shell script to start the Swingbench GUI load generator. This is intended to be invoked manually by an end user after deploy in the VNC viewer.
- run shell commands to do the following:
  - restart (reset) vncserver
  - mount an NFS drive and copy Oracle Instant Client and Swingbench software to local file system
  - add Oracle Instant Client binary path to root user's PATH in .bash_profile
  - wait for *oradbaas* deploy to complete post-deployment and load the Swingbench Order Entry schema by running sbsetup.sh

8.3.2 Input and Output parameters

oraclient template input parameters:

Input Parameter	Description
Namespace	This is the ICP namespace or Red Hat OpenShift project name in which the instance will be deployed.
Instance name	This is the name of the instance being deployed; displayed on the T&SA deployed instance screen. Choose any appropriate name.
dbclient_name	This is used to name the PowerVC VM. In our template, we concatenate this parameter to the dbname (dbserver_user_data), and a timestamp in order to have a unique name that can be matched up to the dbclient VM. For example, **dbc**-testdb-<YYMM_hhmmss>.
openstack_image_name	This is the PowerVC Compute template which specifies the compute characteristics of the VM to be deployed. The flavor (OpenStack) or compute template (PowerVC) specifies VM specifications, including number of CPUs, RAM, etc. Alternative option would be to use the flavor id instead.
openstack_flavor_name	This is the name of the deployable image in PowerVC. Alternative option would be to use image id.
openstack_network_name	This is the network name in OpenStack. Note that we'd typically use the network ID but also wanted to show the use of network name in this example.
dbserver_user_data	This parameter will be linked to oradbaas' input parameter of the same name, which is used for the Oracle dbname. The dbname is needed for the connect string to the Oracle database, and it is also needed to build the VM name. See dbclient_name above.
dbserver_ipaddr	dbserver_ipaddr is assigned by PowerVC, therefore is not known at deployment request time. dbserver_ipaddr is obtained as input from the *oradbaa*s output variable.
dbserver_ts	This is used to accept *oradbaas*' output parameter of the same name. See *oradbaas* output parameter above.

oraclient template output parameters are for display on the T&SA deployed instance screen:

Output Parameter	Description
dbclient_ipaddr	IP address of the dbclient VM, needed for user/requester access.
dbclient_vm_name	dbclient VM name in PowerVC, useful for verification and troubleshooting.
dbserver_connect	Oracle jdbc connect string. Used by the Swingbench load generator to connect to the database.
dbclient_vnc	Host and display information for access using VNC viewer.

8.3.3 Create *oraclient* template

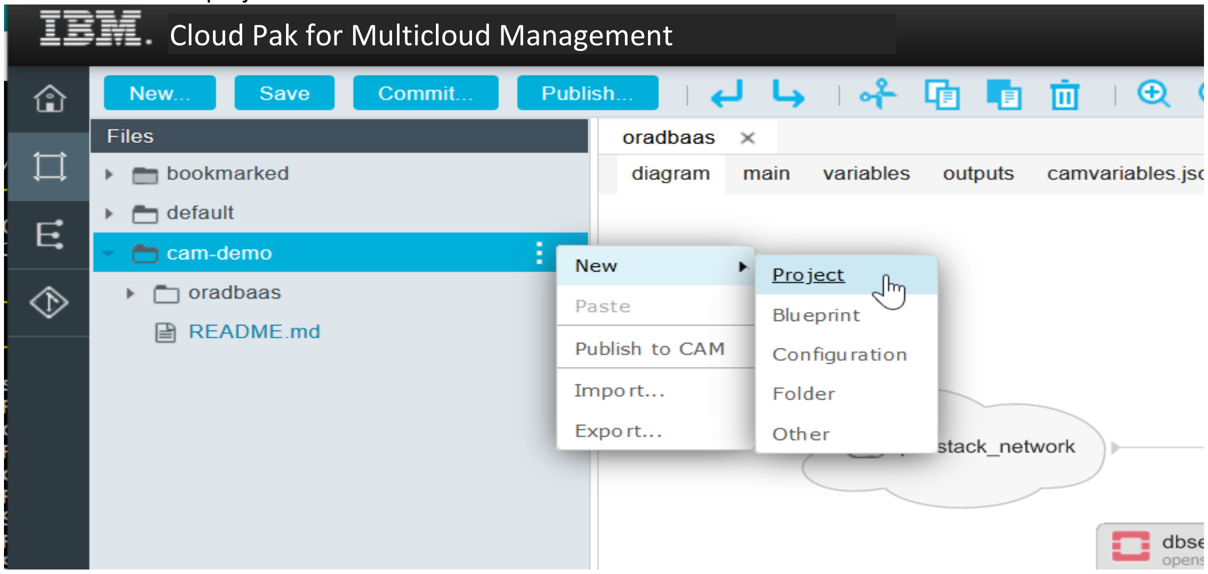
In this section we provide only the critical steps to create the oraclient template. The contents of the template files can be found in section 10.3, *oraclient* Template files, and can be copied / pasted into the correct locations in Template Designer.

It is important to closely look at the main.tf file and understand how the file injection is specified in the openstack compute instance resource definition.
Here a sample snippet which, at deploy time, creates the file "dbserver.cfg" in the directory /root within the VM.

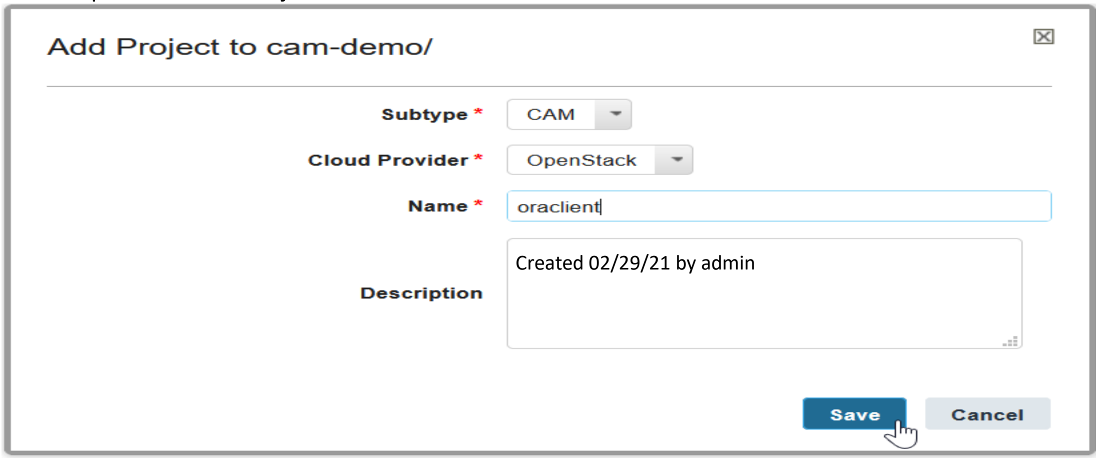
```
#cloud-config
write_files:
  - content: |
      # dbserver parameters
      ${format("export DBNAME=%s",var.dbserver_user_data)}
      ${format("export DBSERVERIPADDR=%s",var.dbserver_ipaddr)}
      export ORASYSPW=oracle
      export SWINGBENCHBINDIR=/opt/swingbench/bin
    path: /root/dbserver.cfg
```

Note how the content of the file is dynamically generated at deploy time based on the values of input parameters derived from the output of the *oradbaas* template deploy step. The parameter transfer between the *oradbaas* and *oraclient* templates is discussed in more detail in section 9, Developing our Service.

To create the *oraclient* template, we go back to the Template Designer, select Blueprints, our cam-demo repository and then create a new project as shown below.

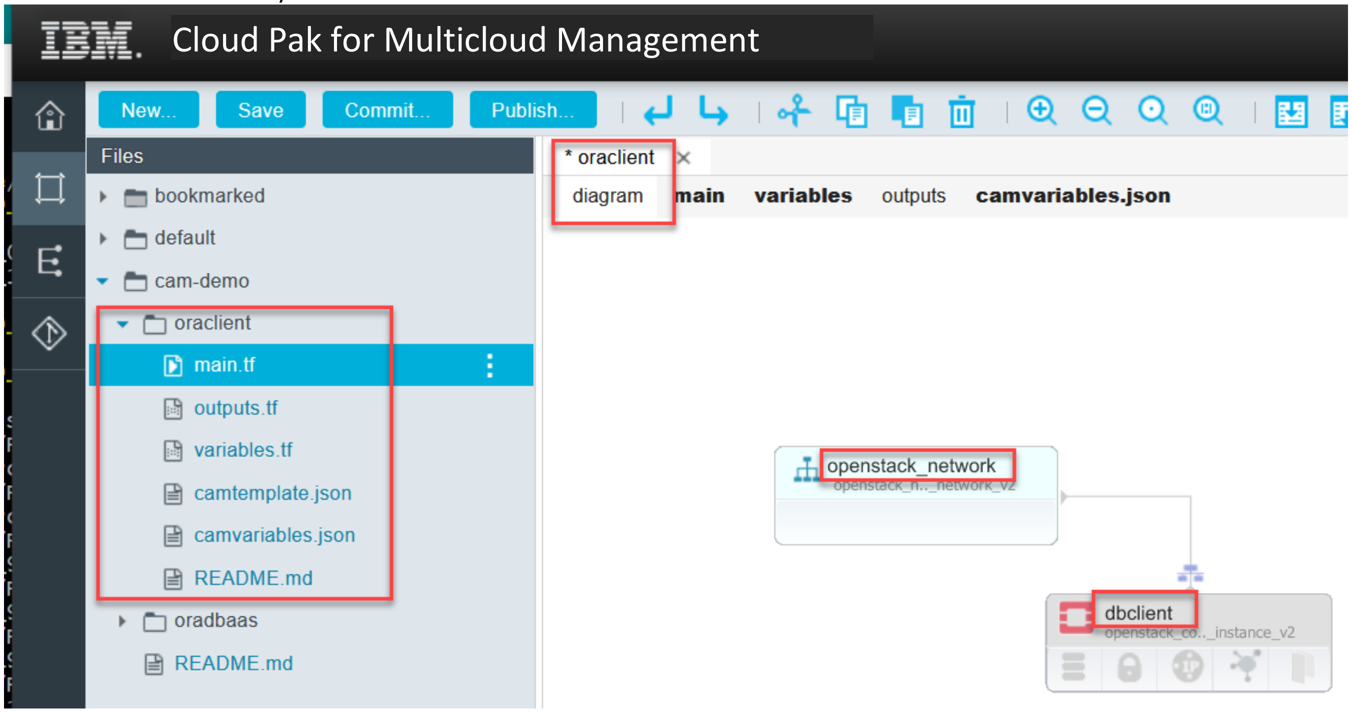


We complete the Add Project screen as shown below and then click Save.

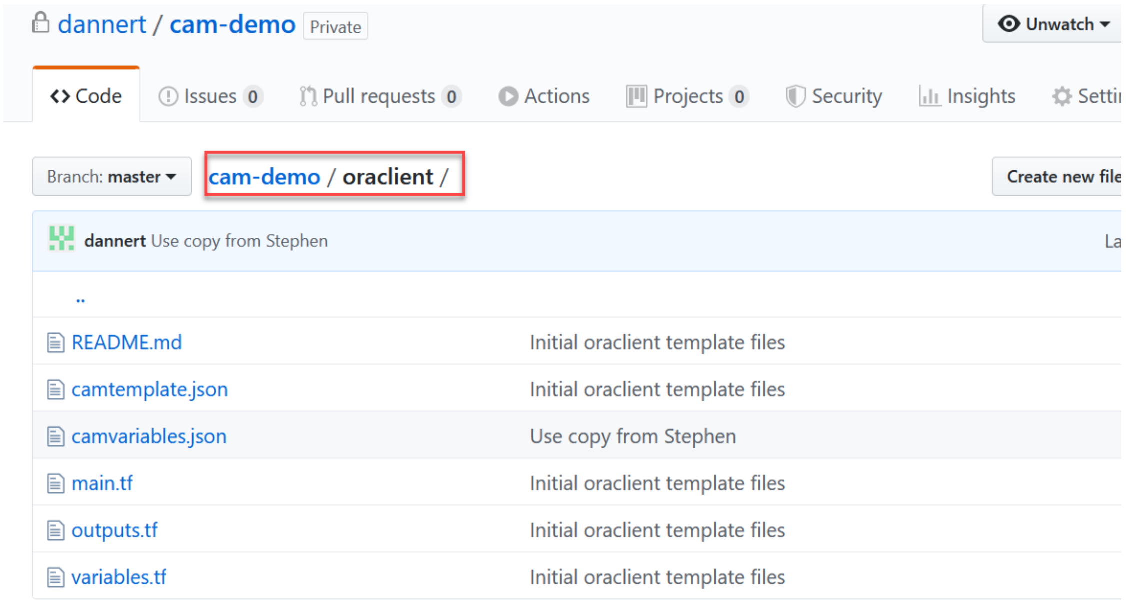

Add Project to cam-demo/

Subtype * CAM
Cloud Provider * OpenStack
Name * oraclient
Description Created 02/29/21 by admin

Similar to the steps performed when creating the *oradbaas template*, we then switch to the diagram view and create an openstack instance, openstack network and connect the two. Note the names we used for the openstack_network “**network**” and the openstack_instance **“dbclient**”. Those names are referenced in the variable names in the template files and need to be set exactly as mentioned.



We then copied the respective code from section 10.3 in the appendix into the respective files and finally committed the new project to the GitHub repository. GitHub now shows a new sub-directory for *oraclient* with the respective template files:


dannert / cam-demo Private

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Security | Insights | Setti

Branch: master | cam-demo / oraclient / | Create new file

dannert Use copy from Stephen

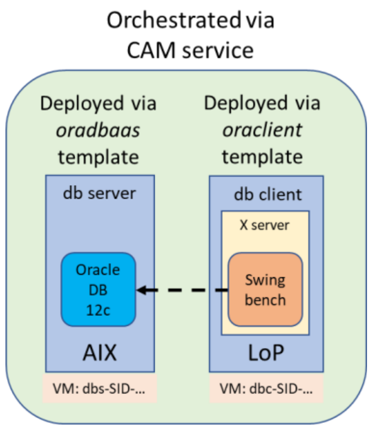
..

README.md	Initial oraclient template files
camtemplate.json	Initial oraclient template files
camvariables.json	Use copy from Stephen
main.tf	Initial oraclient template files
outputs.tf	Initial oraclient template files
variables.tf	Initial oraclient template files

As the last step we published the *oraclient* template under the template name *oraclient* to T&SA. For the required steps refer to the section *oradbaas* template.

9 Developing our Service

In this section we describe how to create a service to deploy the *oradbaas* and *oraclient* templates in an orchestrated fashion. In the end state, after the service has been deployed by the end user, the requester will be able to access the dbclient VM using VNC viewer and run Swingbench's Order Entry load generator against the database server.


Orchestrated via CAM service
Deployed via *oradbaas* template
Deployed via *oraclient* template
db server
Oracle DB 12c
AIX
VM: dbs-SID-...
db client
X server
Swing bench
LoP
VM: dbc-SID-...

9.1 Service Design

Use T&SA Service Composer to create the service to deploy the *oradbaas* and *oraclient* templates together. To minimize the number of parameters an end-user must fill in, we specify most of the input parameters required for the *oradbaas* and *oraclient* templates in the service definition. For example, this includes image and flavor names as well as network id and leaves only the Oracle database name as input parameter for the deployer of the service.

Input Parameter	Description
dbserver_user_data	Oracle dbname. This is linked to the oradbaas input parameter.

Create service output parameters for display on the deployed service screen.

Output Parameter	Description
dbserver_ipaddr	dbserver IP address
dbserver_vm_name	dbserver VM (in PowerVC) name. Expected value <prefix>-<dbname>-<timestamp>
dbclient_ipaddr	dbclient IP address
dbclient_vm_name	dbclient VM (in PowerVC) name. See dbserver_vm_name.
ora_dbname	Oracle database dbname.
oradb_connect	Oracle database connect string (jdbc). Expected value - //<ipaddress>/<dbname>
dbclient_vnc_addr	VNC host and display information. Expected value - <ipaddress>:<display#>.

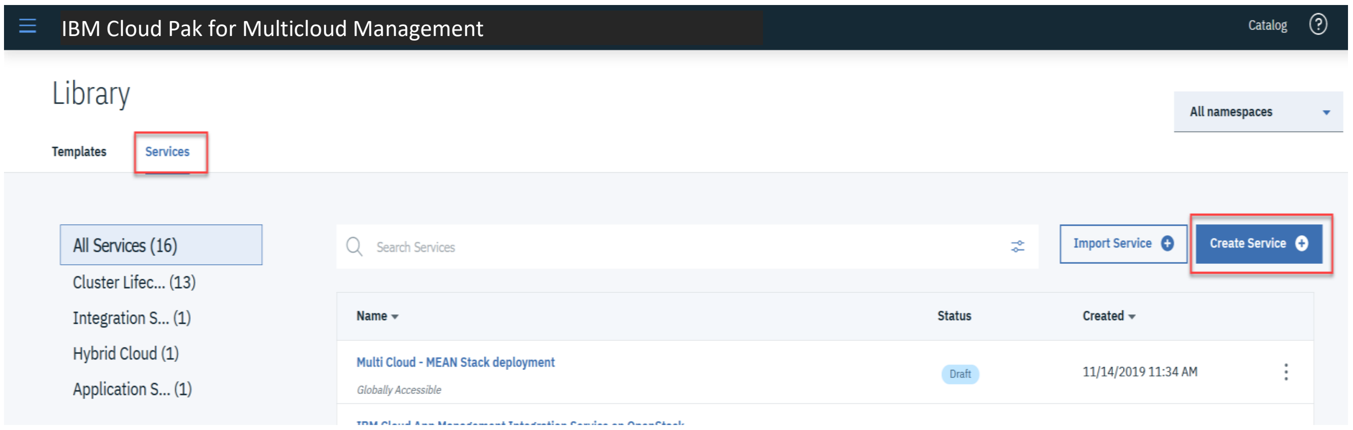
9.2 Creating the Service

The Service Composer, part of the Terraform & Service Automation Service library, enables the authoring of T&SA services. After T&SA templates are created, administrators can use the Service Composer to create a service, consisting of T&SA templates or Kubernetes Helm charts from IBM Cloud Private, based on a designed sequence or decision.

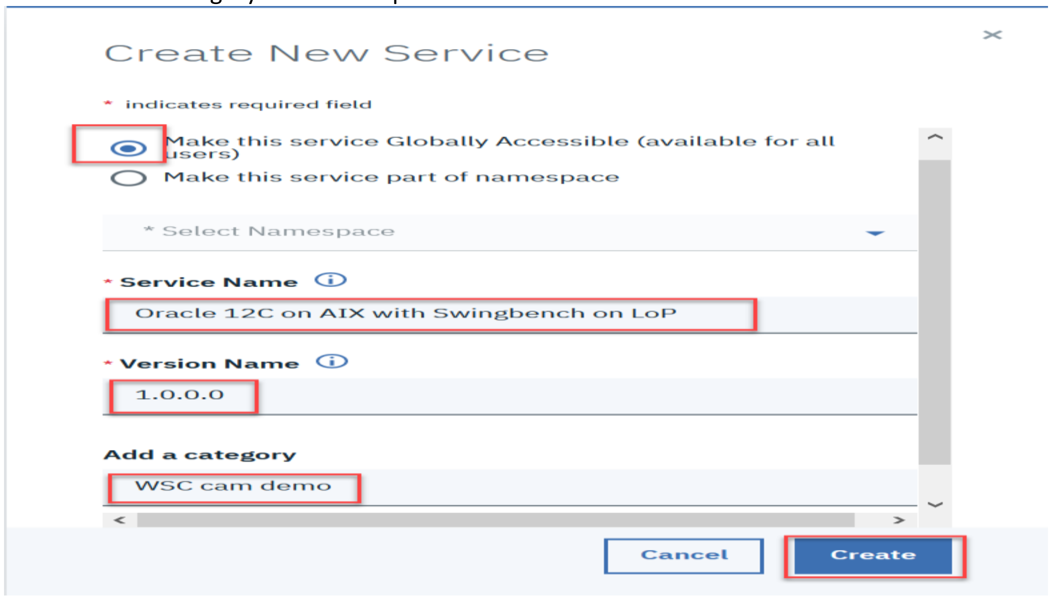
9.2.1 Create Service

The Service Composer is accessed via the IBM Terraform & Service Automation GUI and can be invoked from the Service Library. Access the Service Library by clicking the hamburger menu and select Library->Services.

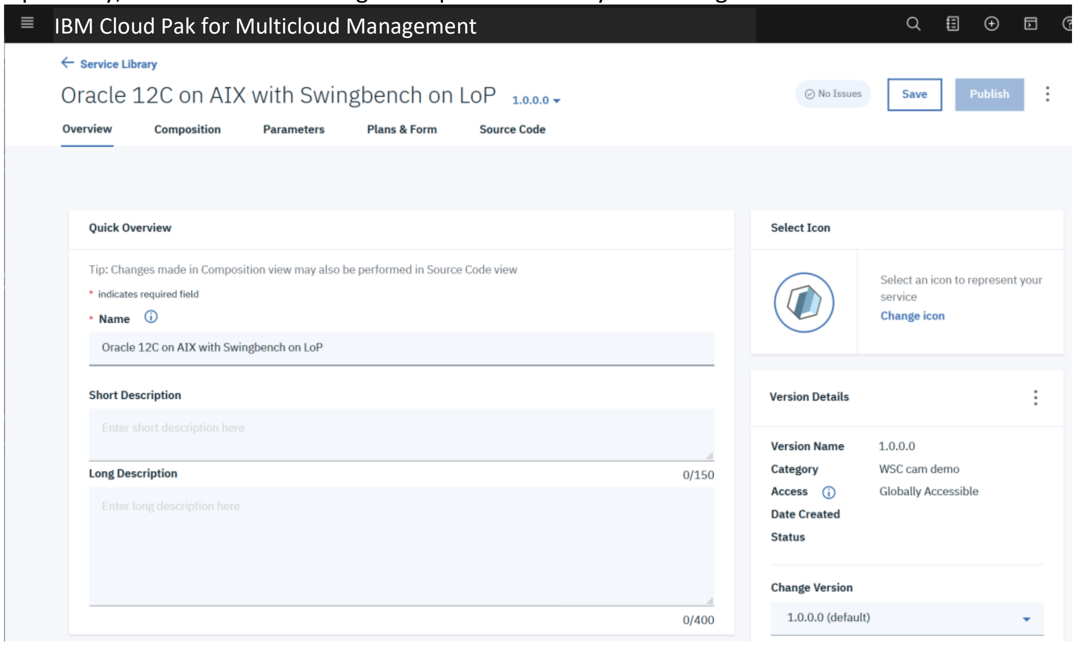
On the Service Library screen, click the **Create Service** button.



On the Create New Service window, enter the Service name and add a category or pick one from the drop-down list. We created a new category in our example.

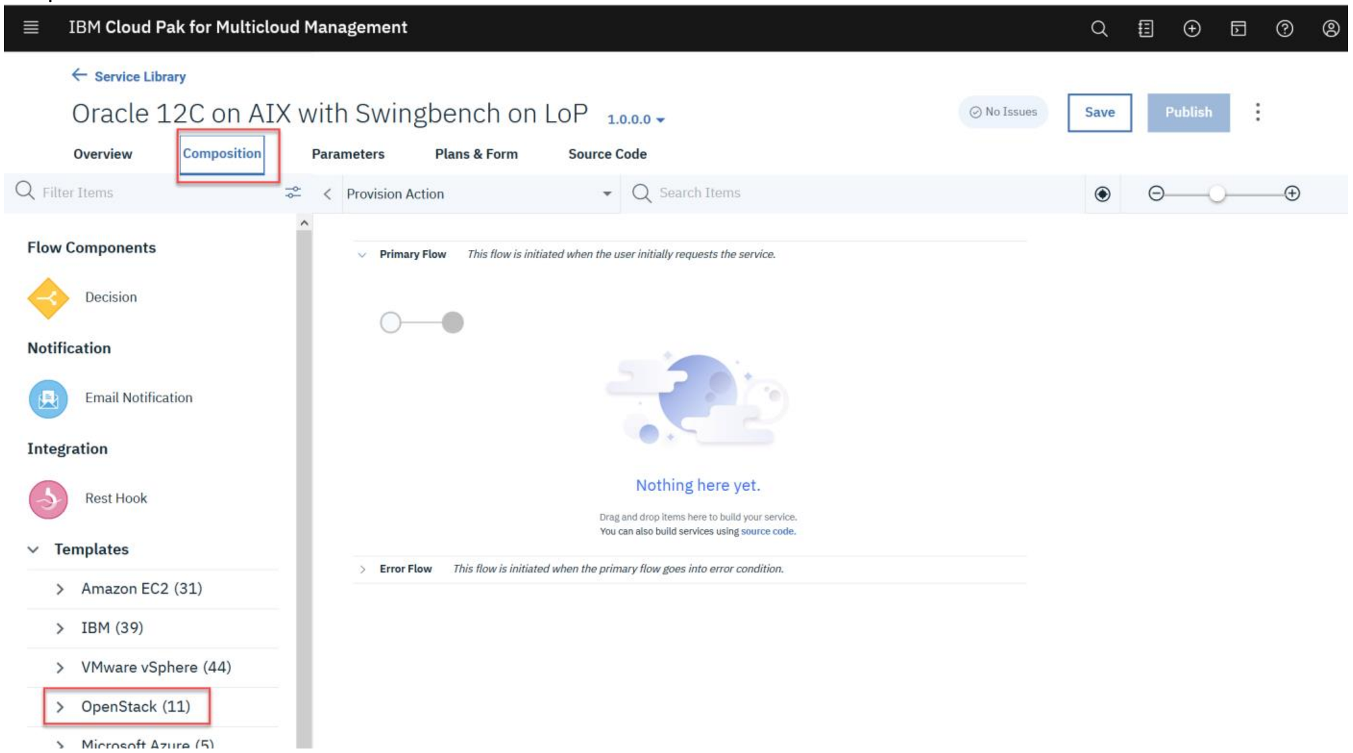

Create New Service
* indicates required field
Make this service Globally Accessible (available for all users)
Make this service part of namespace
* Select Namespace
* Service Name
Oracle 12C on AIX with Swingbench on LoP
* Version Name
1.0.0.0
Add a category
WSC cam demo
Cancel
Create

Optionally, enter a short and long description. You may also change the icon here if desired.

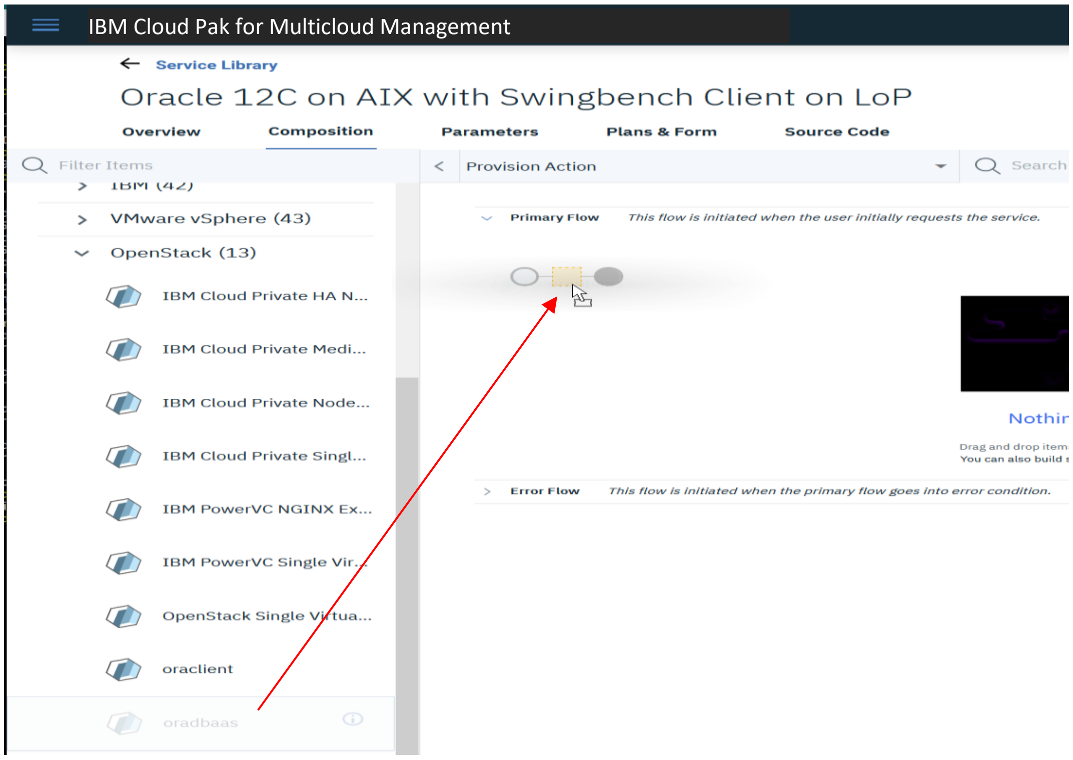

IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP
1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Quick Overview
Tip: Changes made in Composition view may also be performed in Source Code view
* indicates required field
* Name
Oracle 12C on AIX with Swingbench on LoP
Short Description
Enter short description here
0/150
Long Description
Enter long description here
0/400
Select Icon
Select an icon to represent your service
Change icon
Version Details
Version Name 1.0.0.0
Category WSC cam demo
Access Globally Accessible
Date Created
Status
Change Version
1.0.0.0 (default)

Click **Composition**.

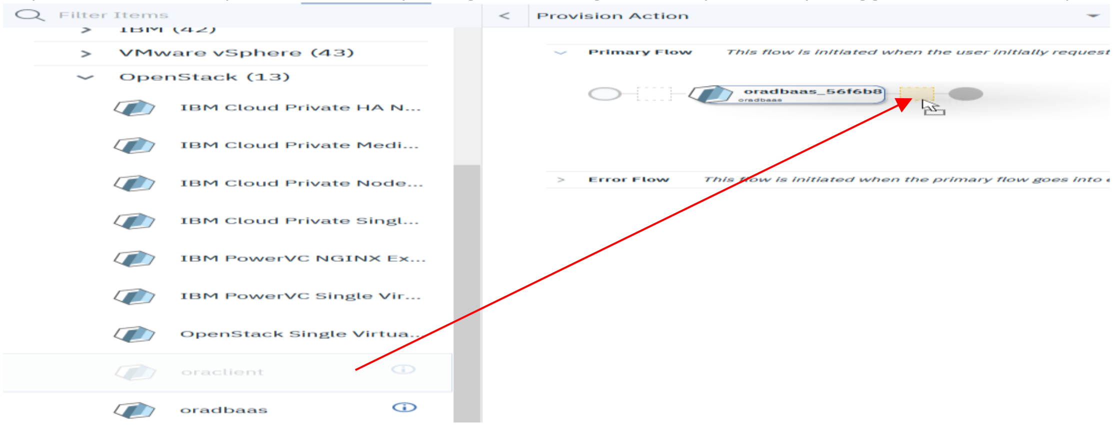
On the left-hand panel, expand the **OpenStack** group under Templates and scroll down to find the *oradbaas* and *oraclient* templates.



Point to the *oradbaas* template, left click and hold on the *oradbaas* template, and drag it to the space between the two dots, shown by the arrow below. A rectangular box will appear between the two dots when the dragged object approaches the line. Release the mouse button to drop the template into this rectangular box.

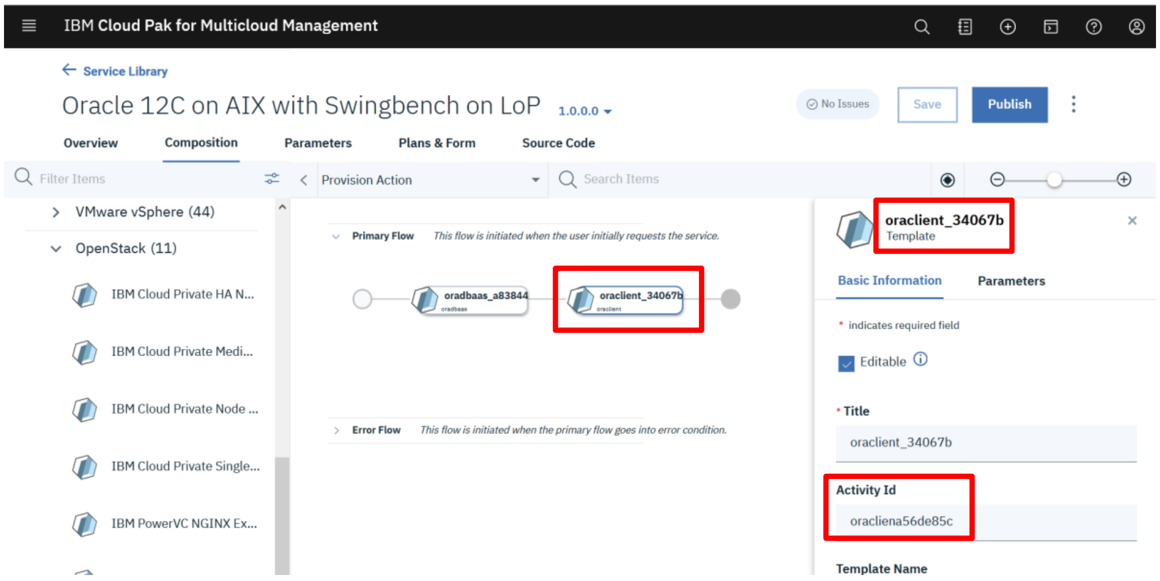

IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench Client on LoP
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search
IBM (42)
VMware vSphere (43)
OpenStack (13)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node...
IBM Cloud Private Singl...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Vir...
OpenStack Single Virtua...
oraclient
oradbaas
Primary Flow
This flow is initiated when the user initially requests the service.
Error Flow
This flow is initiated when the primary flow goes into error condition.

Repeat the same steps for oraclient, placing it to the right of the previously dragged oradbaas template.

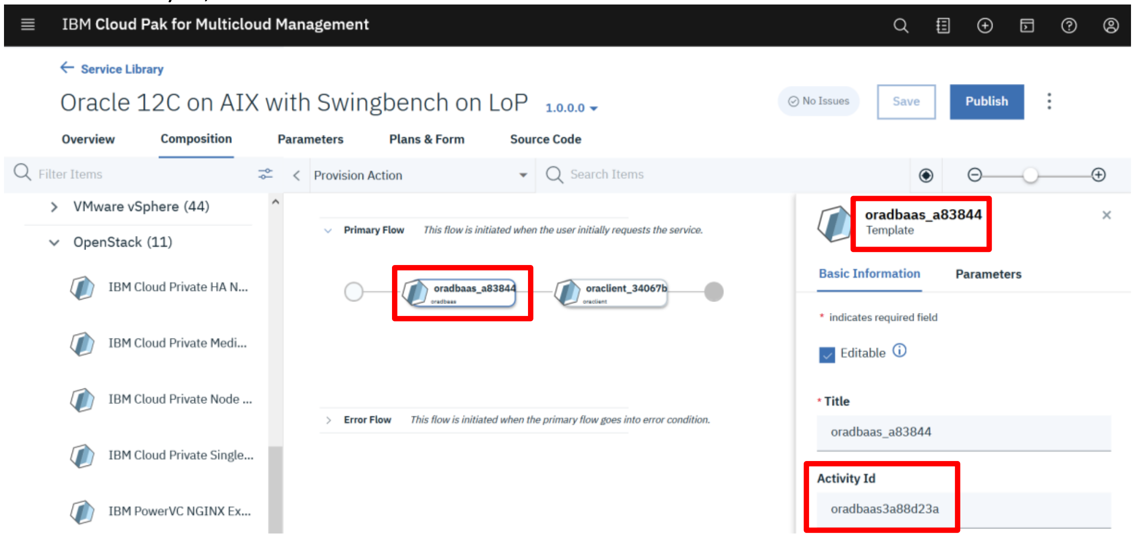

Filter Items
IBM (42)
VMware vSphere (43)
OpenStack (13)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node...
IBM Cloud Private Singl...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Vir...
OpenStack Single Virtua...
oraclient
oradbaas
Provision Action
Primary Flow
This flow is initiated when the user initially request
oradbaas_56f6b8
Error Flow
This flow is initiated when the primary flow goes into

When done, the two templates will be lined up between the two dots as shown below.

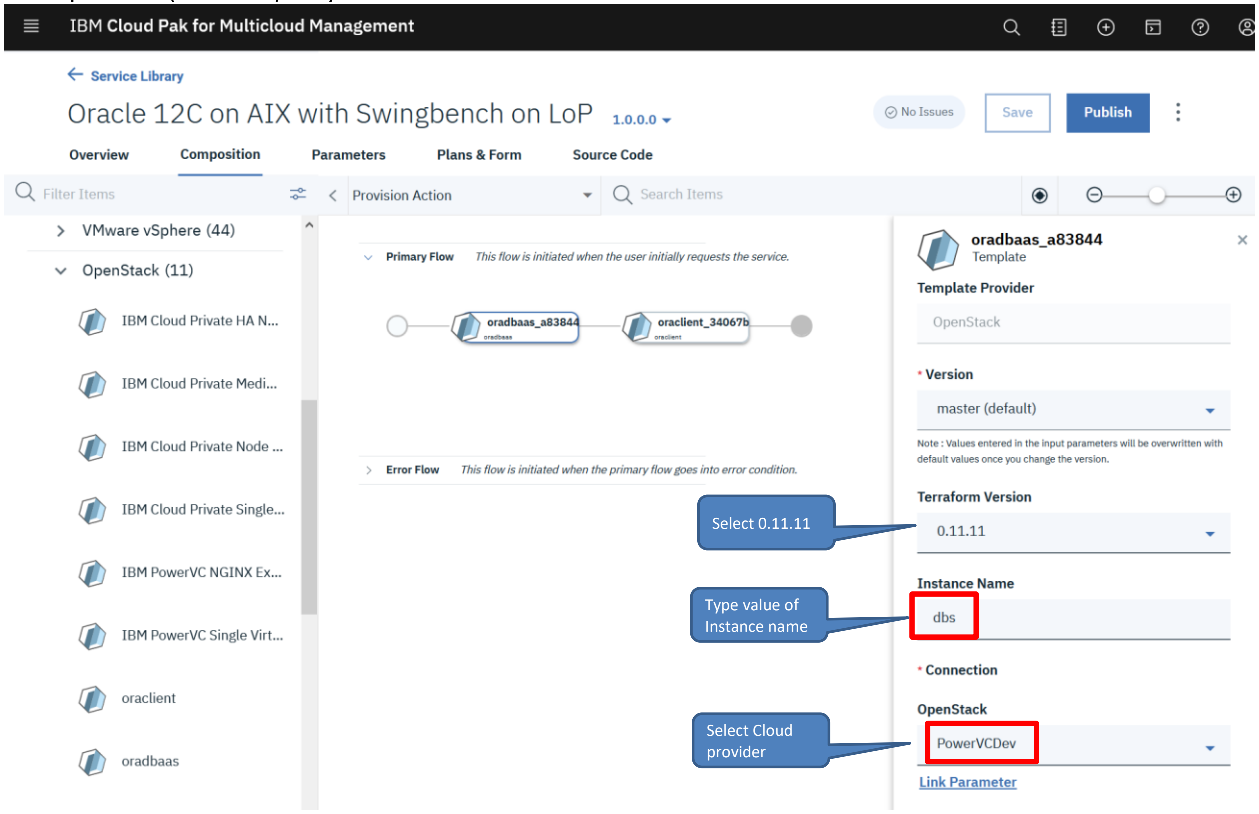
Since it's the most recently dragged/placed object, *oraclient* template's **Basic Information** tab will be shown on the right-hand panel. (If not, select Basic Information.) Save the value for **Activity Id,** which in this example, is `oracliena56de85c`, for later.



Click the *oradbaas* template icon to show *oradbaas*' **Basic Information** tab on the right-hand panel. As before, save the value of Activity Id, in this case: `oradbaas3a88d23a`.



With oradbaas still selected, scroll down the right-hand panel to find **Instance Name**. This will specify the deployment instance name in T&SA. Type in a suitable name, e.g. **dbs**. Also verify that the value under **Connection** reflects the correct cloud provider (PowerVC) for your environment.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow
This flow is initiated when the primary flow goes into error condition.
Select 0.11.11
Type value of Instance name
Select Cloud provider
Link Parameter

oradbaas_a83844
Template

Template Provider: OpenStack

* Version: master (default)

Note : Values entered in the input parameters will be overwritten with default values once you change the version.

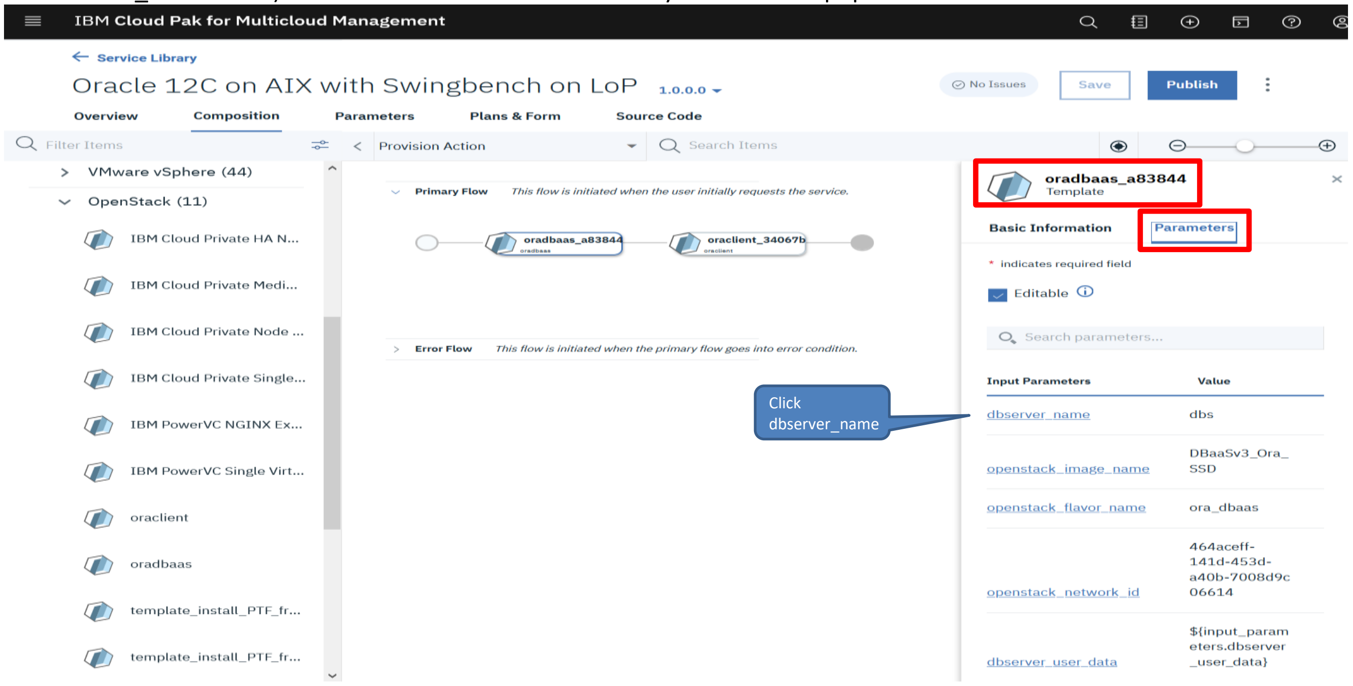
Terraform Version: 0.11.11

Instance Name: dbs

* Connection

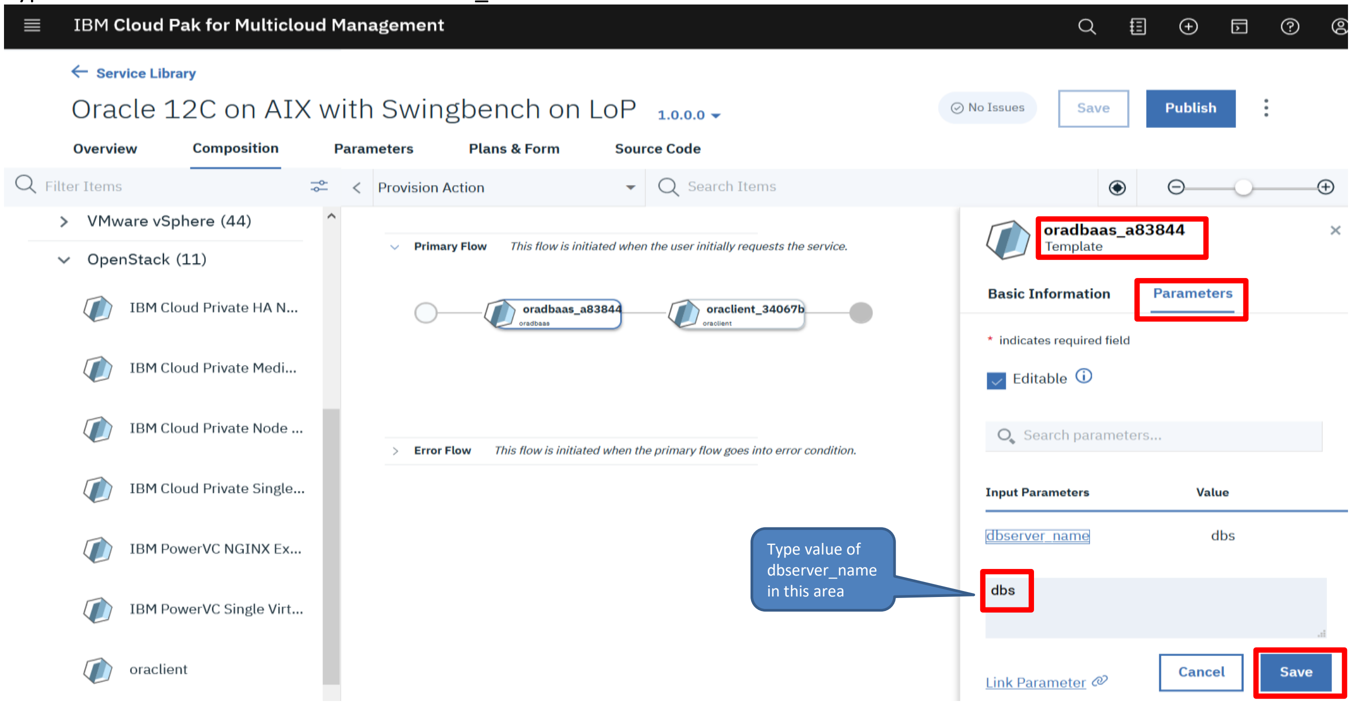
OpenStack: PowerVCDev

Select the Parameters tab and click dbserver_name under Input Parameters. This dbserver_name will be the prefix of the VM name in PowerVC. (Recall from earlier that we are building the dbserver VM name in the format <prefix>-<dbname>-<YYMMDD>_<hhmmss>.) Note that screenshot below already had all fields populated.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
template_install_PTF_fr...
template_install_PTF_fr...
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow
This flow is initiated when the primary flow goes into error condition.
Click dbserver_name
oradbaas_a83844
Template
Basic Information
Parameters
* indicates required field
Editable
Search parameters...

Input Parameters	Value
dbserver_name	dbs
openstack_image_name	DBaaSv3_Ora_SSD
openstack_flavor_name	ora_dbaas
openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614
dbserver_user_data	${input_parameters.dbserver_user_data}

Type dbs in the area shown below dbserver_name. Click Save.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow
This flow is initiated when the primary flow goes into error condition.
oradbaas_a83844
Template
Basic Information
Parameters
* indicates required field
Editable
Search parameters...
Type value of dbserver_name in this area
Link Parameter
Cancel
Save

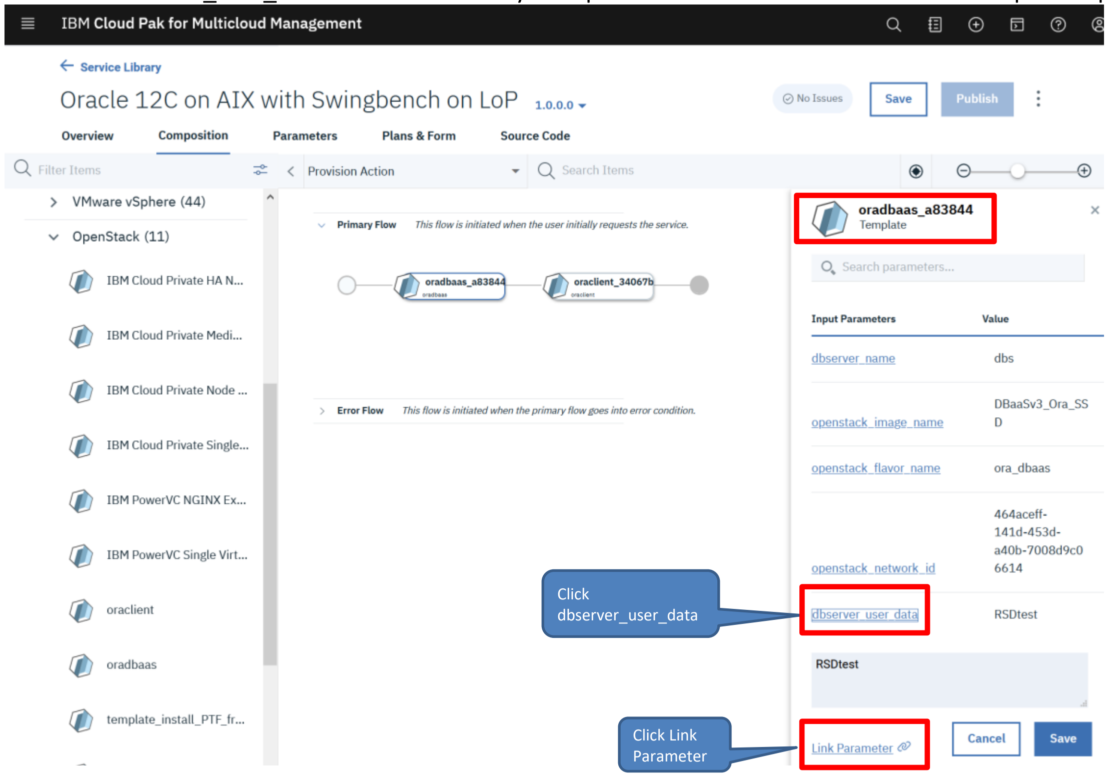
Input Parameters	Value
dbserver_name	dbs

dbs

Repeat the previous step to specify the other input parameters we want to pre-set for this service. You need to determine the respective values from your specific PowerVC environment!

Input Parameter	Values from PowerVC
openstack_image_name,	DBaaSv3_Ora_SSD
openstack_flavor_name	ora_dbaas
openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614

When completed, the right-hand panel of the screen will look like on the screen below.
Now click dbserver_user_data. Note that initially the specified default from the *oradbaas* template is pre-filled here.

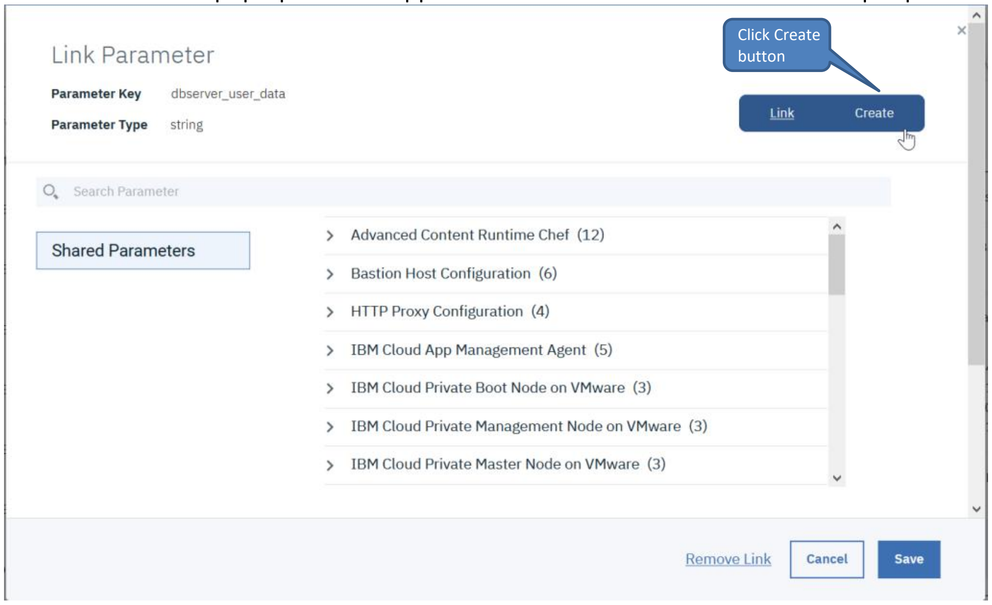

IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
template_install_PTF_fr...
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow
This flow is initiated when the primary flow goes into error condition.
oradbaas_a83844
Template
Search parameters...
Click dbserver_user_data
Click Link Parameter
Link Parameter
Cancel
Save

Input Parameters	Value
dbserver_name	dbs
openstack_image_name	DBaaSv3_Ora_SSD
openstack_flavor_name	ora_dbaas
openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614
dbserver_user_data	RSDtest

RSDtest

Under dbserver_user_data (may need to scroll down to find), click **Link Parameter.**

The Link Parameter pop-up window appears. Click Create button to create an input parameter for the service.


Link Parameter

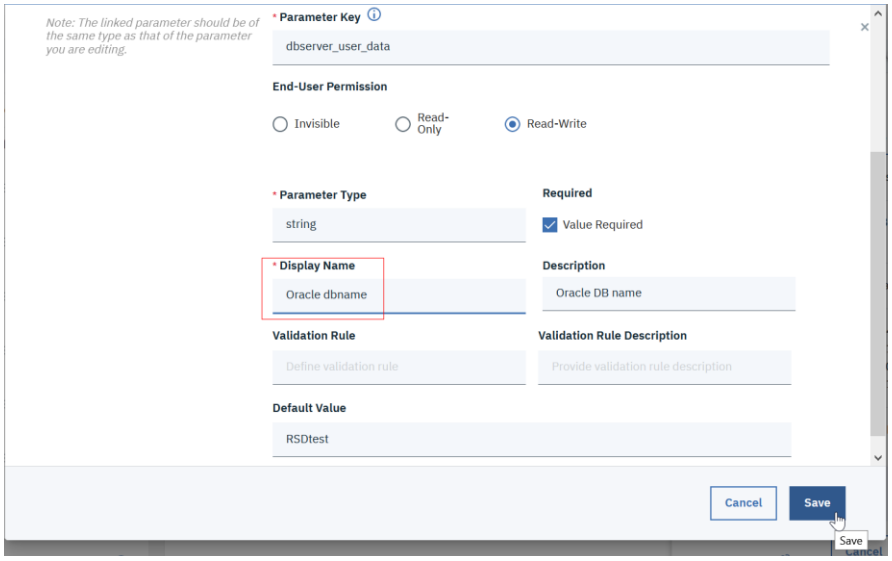
Parameter Key dbserver_user_data

Parameter Type string

- Advanced Content Runtime Chef (12)
- Bastion Host Configuration (6)
- HTTP Proxy Configuration (4)
- IBM Cloud App Management Agent (5)
- IBM Cloud Private Boot Node on VMware (3)
- IBM Cloud Private Management Node on VMware (3)
- IBM Cloud Private Master Node on VMware (3)

Scroll down to **Display Name** in the pop-up window and adjust to read “Oracle dbname”.

Review and click the **Save** button.


Note: The linked parameter should be of the same type as that of the parameter you are editing.

* Parameter Key: dbserver_user_data

End-User Permission: Invisible [✓ Read-Write] Read-Only

* Parameter Type: string

Required: [✓] Value Required

* Display Name: Oracle dbname

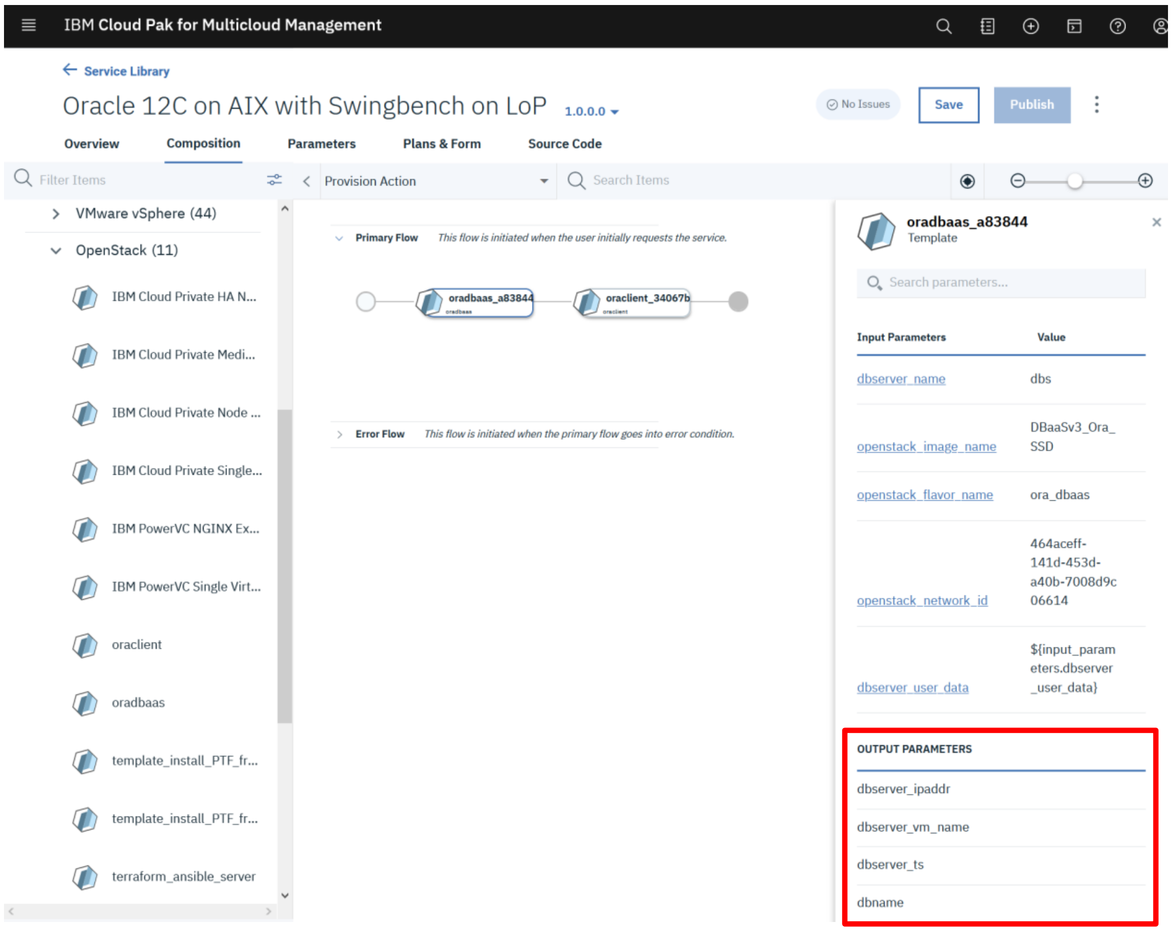
Description: Oracle DB name

Validation Rule: Define validation rule

Validation Rule Description: Provide validation rule description

Default Value: RSDtest

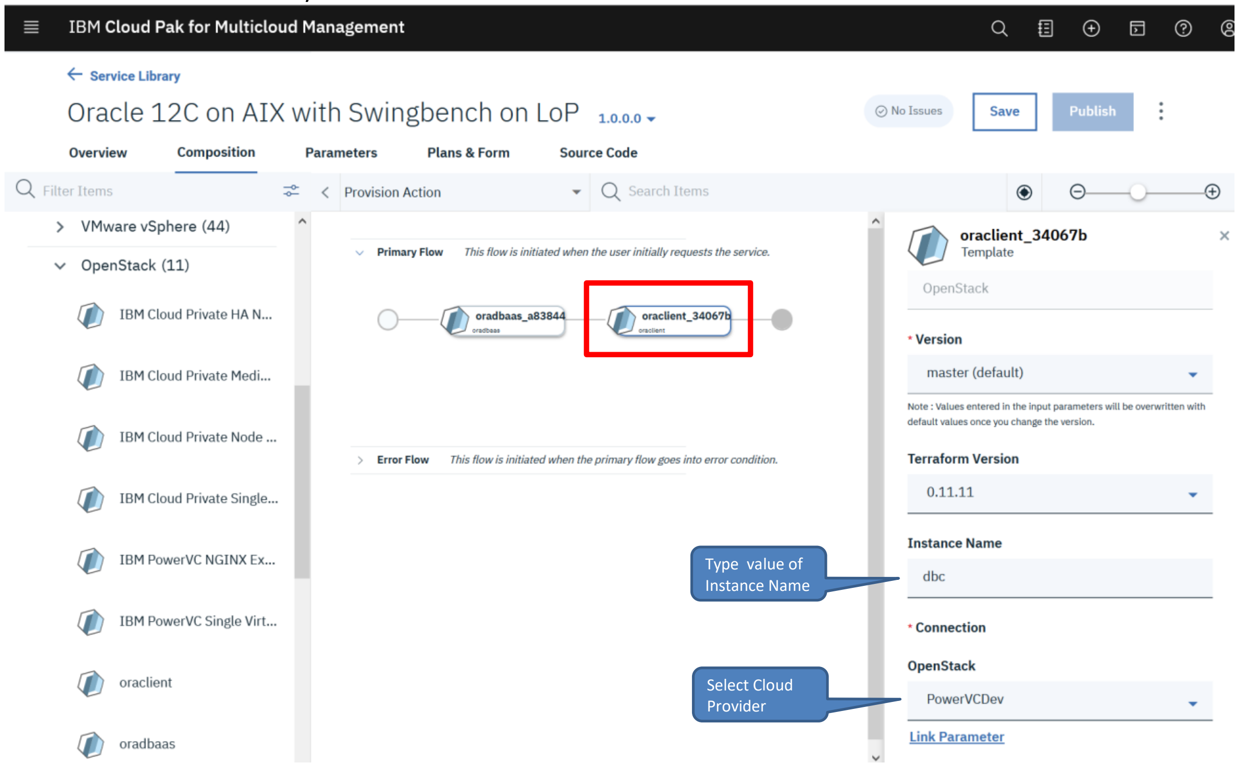
Note the Output Parameters (right-hand panel on next picture). These were defined in the *oradbaas* template (outputs.tf). Later, in section 9.2.2, we will create Service Output Parameters for dbserver_ipaddr and dbserver_vm_name, so that they will be displayed on the Services detailed information page.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
template_install_PTF_fr...
template_install_PTF_fr...
terraform_ansible_server
Primary Flow This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow This flow is initiated when the primary flow goes into error condition.
oradbaas_a83844
Template
Search parameters...

Input Parameters	Value
dbserver_name	dbs
openstack_image_name	DBaaSv3_Ora_SSD
openstack_flavor_name	ora_dbaas
openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614
dbserver_user_data	${input_parameters.dbserver_user_data}

OUTPUT PARAMETERS
dbserver_ipaddr
dbserver_vm_name
dbserver_ts
dbname

Next, we will set up the input parameters of the *oraclient* template.
Select the *oraclient* template by clicking the oraclient_34067b icon in the middle panel of the screen. Scroll down on the right-hand panel to find **Instance Name** and type in **dbc** as shown below. Also verify that the correct cloud provider is selected in the **Connection** field. If you decide to use a different cloud provider (still PowerVC), that is possible as long as there is network connectivity between the two VMs to be created.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
Primary Flow This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow This flow is initiated when the primary flow goes into error condition.
Type value of Instance Name
Select Cloud Provider
oraclient_34067b
Template
OpenStack

* Version: master (default)

Note : Values entered in the input parameters will be overwritten with default values once you change the version.

Terraform Version: 0.11.11

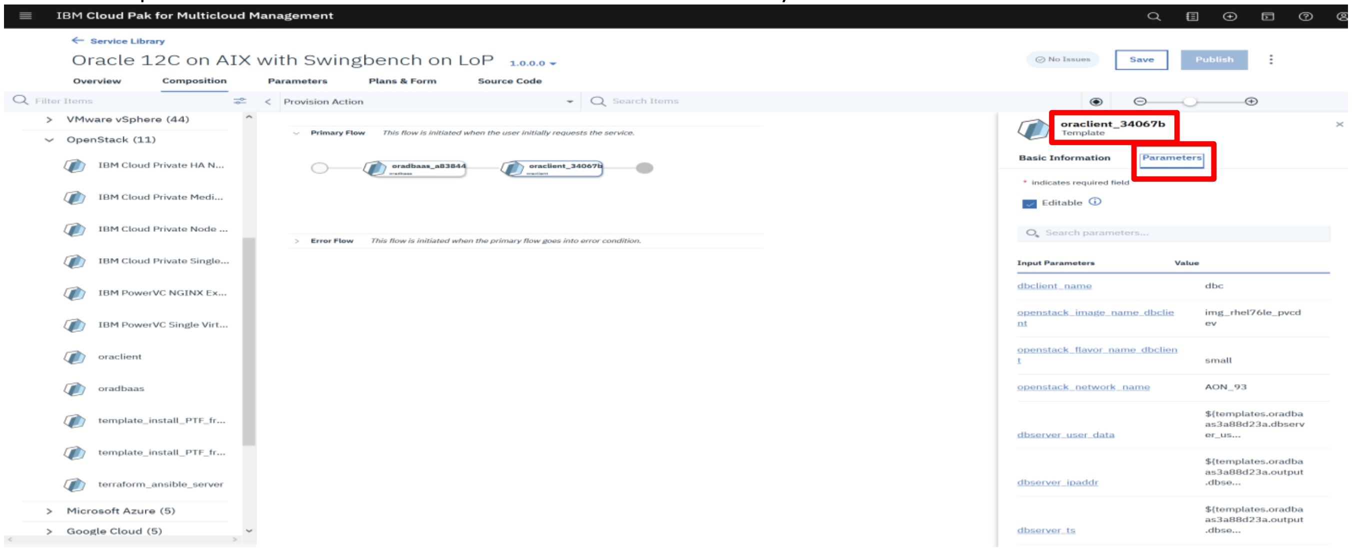
Instance Name: dbc

* Connection

OpenStack: PowerVCDev

Link Parameter

Scroll back up and click Parameters. Note that below screen shot already shows filled in values.


IBM Cloud Pak for Multicloud Management
← Service Library
Oracle 12C on AIX with Swingbench on LoP 1.0.0.0
Overview
Composition
Parameters
Plans & Form
Source Code
No Issues
Save
Publish
Filter Items
VMware vSphere (44)
OpenStack (11)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node ...
IBM Cloud Private Single...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Virt...
oraclient
oradbaas
template_install_PTF_fr...
template_install_PTF_fr...
terraform_ansible_server
Microsoft Azure (5)
Google Cloud (5)
Provision Action
Search Items
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_a83844
oraclient_34067b
Error Flow
This flow is initiated when the primary flow goes into error condition.
oraclient_34067b
Template
Basic Information
Parameters
* indicates required field
Editable
Search parameters...

Input Parameters	Value
dbclient_name	dbc
openstack_image_name_dbclient	img_rhel76le_pvcdev
openstack_flavor_name_dbclient	small
openstack_network_name	AON_93
dbserver_user_data	${templates.oradbaas3a88d23a.dbserver_us...
dbserver_ipaddr	${templates.oradbaas3a88d23a.output.dbse...
dbserver_ts	${templates.oradbaas3a88d23a.output.dbse...

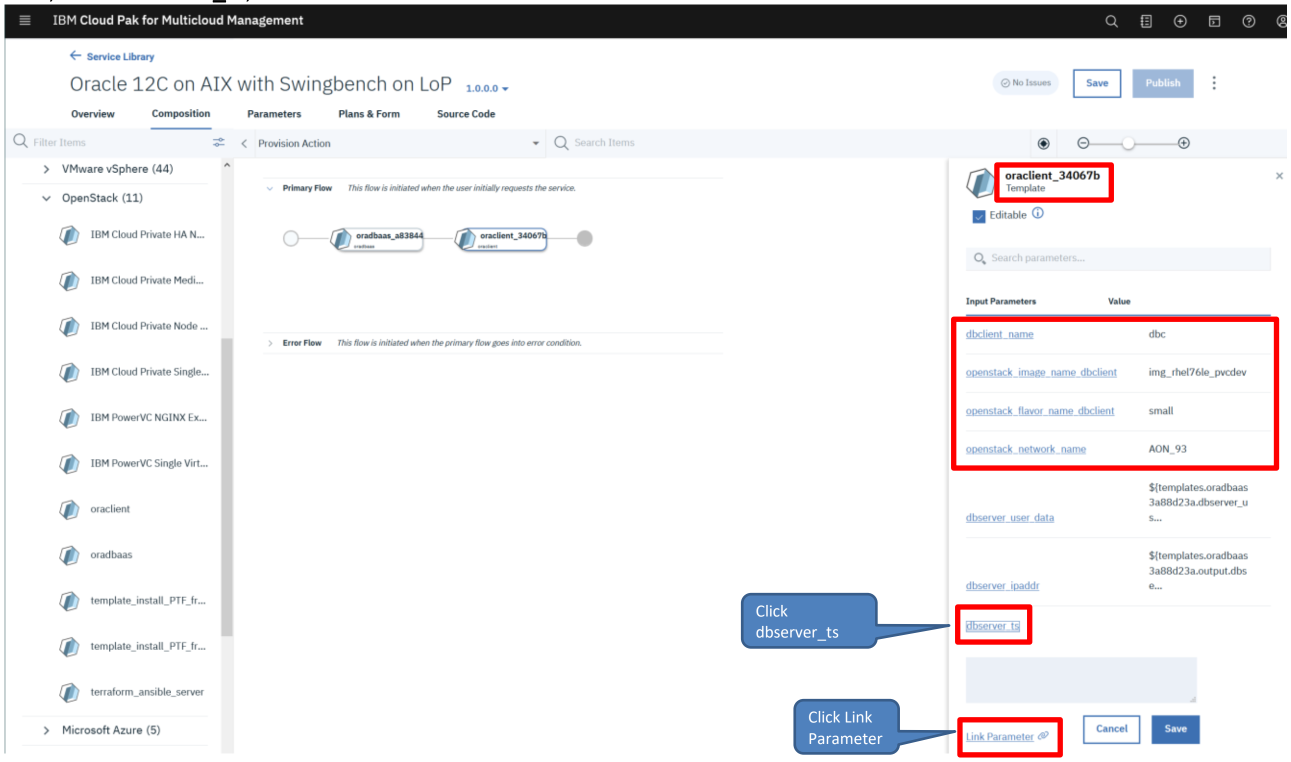
As we did before with the *oradbaas* template, click and fill in the values for your environment for the *oraclient* template. Note that at most the openstack_network_id will be identical to the *oradbaas* template values. Again, adjust the PowerVC specific values to your environment – rows 3-5 in the table below.

Input Parameter	Values
dbclient_name	dbc
openstack_image_name	img_rhel76le_pvcdev
dbclient_flavor_name	small
openstack_network_name	AON_93

Note that the *oraclient* template uses the network name instead of network ID (UUID). AON_93 in this case is the name for the same network we specified for the *oradbaas* template as UUID.

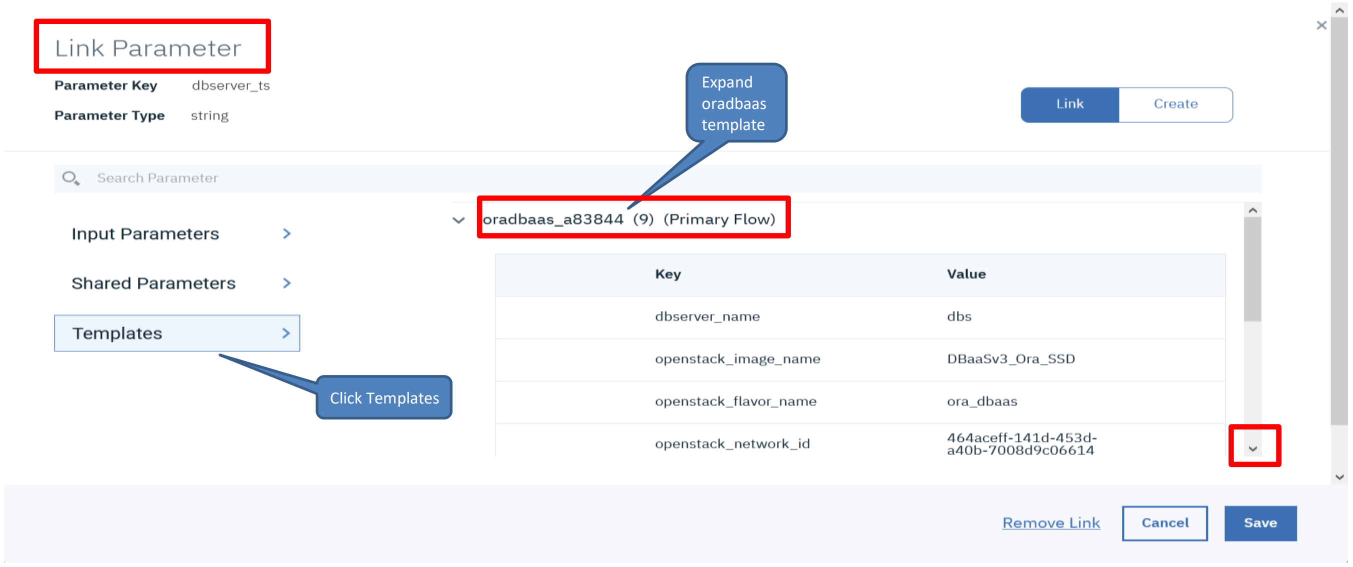
Be sure to click the Save button for each parameter.

Next, click **dbserver_ts**, then Link Parameter.



Input Parameters	Value
dbclient_name	dbc
openstack_image_name_dbclient	img_rhel76le_pvcdev
openstack_flavor_name_dbclient	small
openstack_network_name	AON_93
dbserver_user_data	${templates.oradbaas3a88d23a.dbserver_us...
dbserver_ipaddr	${templates.oradbaas3a88d23a.output.dbse...
dbserver_ts	

On the pop-up window for Link Parameters, click Templates and expand oradbaas_a83844.

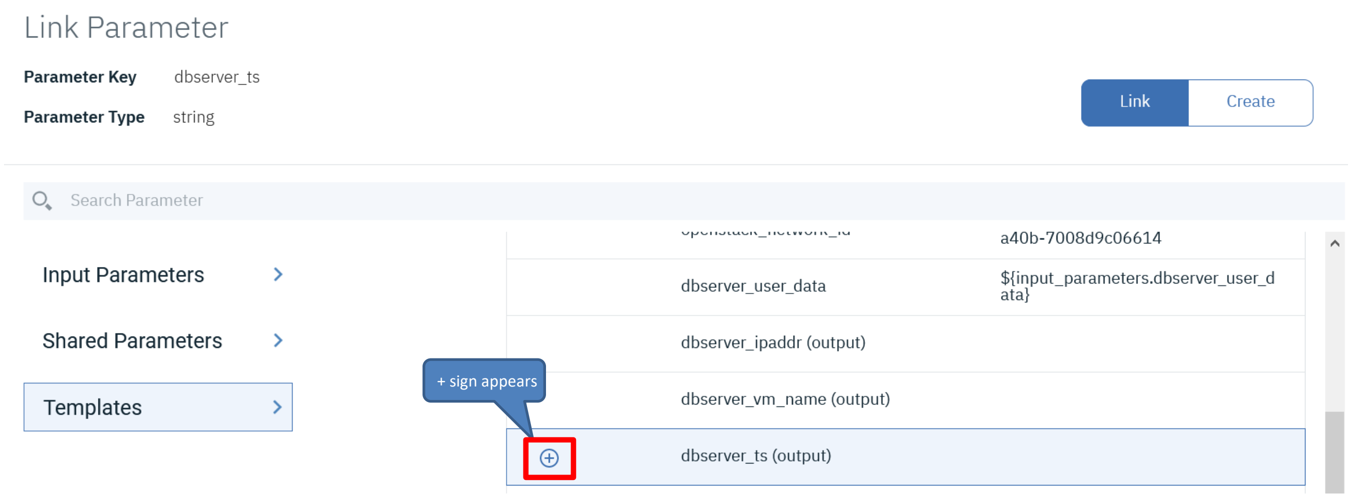

Link Parameter

Parameter Key	dbserver_ts
Parameter Type	string

oradbaas_a83844 (9) (Primary Flow)

Key	Value
dbserver_name	dbs
openstack_image_name	DBaaSv3_Ora_SSD
openstack_flavor_name	ora_dbaas
openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614

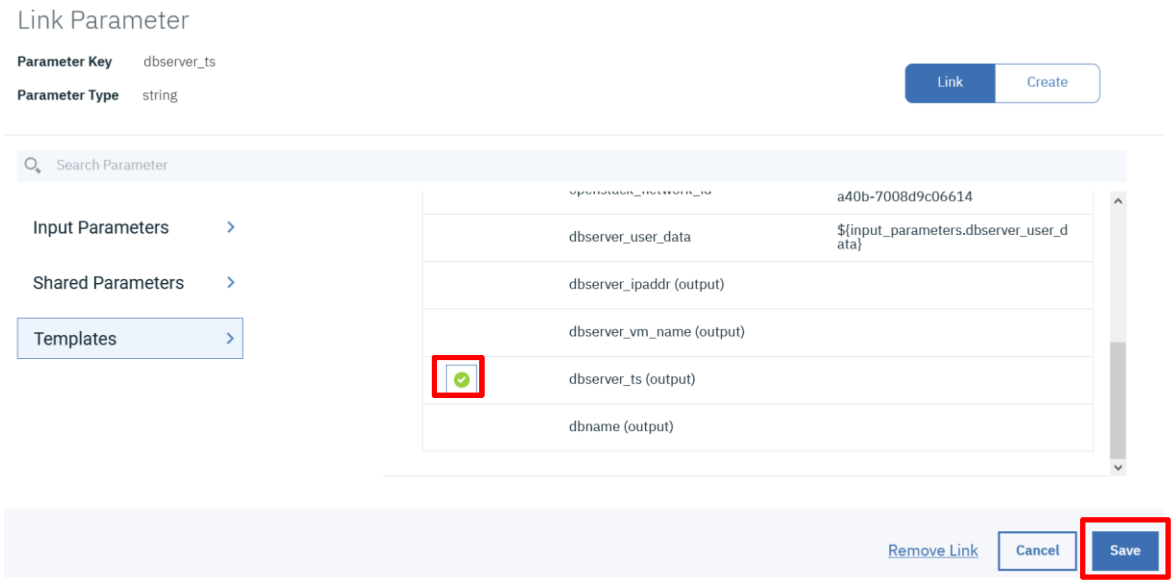
Scroll down to find the **dbserver_ts (output)** key and point the cursor to the dbserver_ts (output) row. The line becomes highlighted and a plus sign appears on the left at the beginning of the line.


Link Parameter

Parameter Key	dbserver_ts
Parameter Type	string

openstack_network_id	a40b-7008d9c06614
dbserver_user_data	${input_parameters.dbserver_user_data}
dbserver_ipaddr (output)	
dbserver_vm_name (output)	
dbserver_ts (output)	

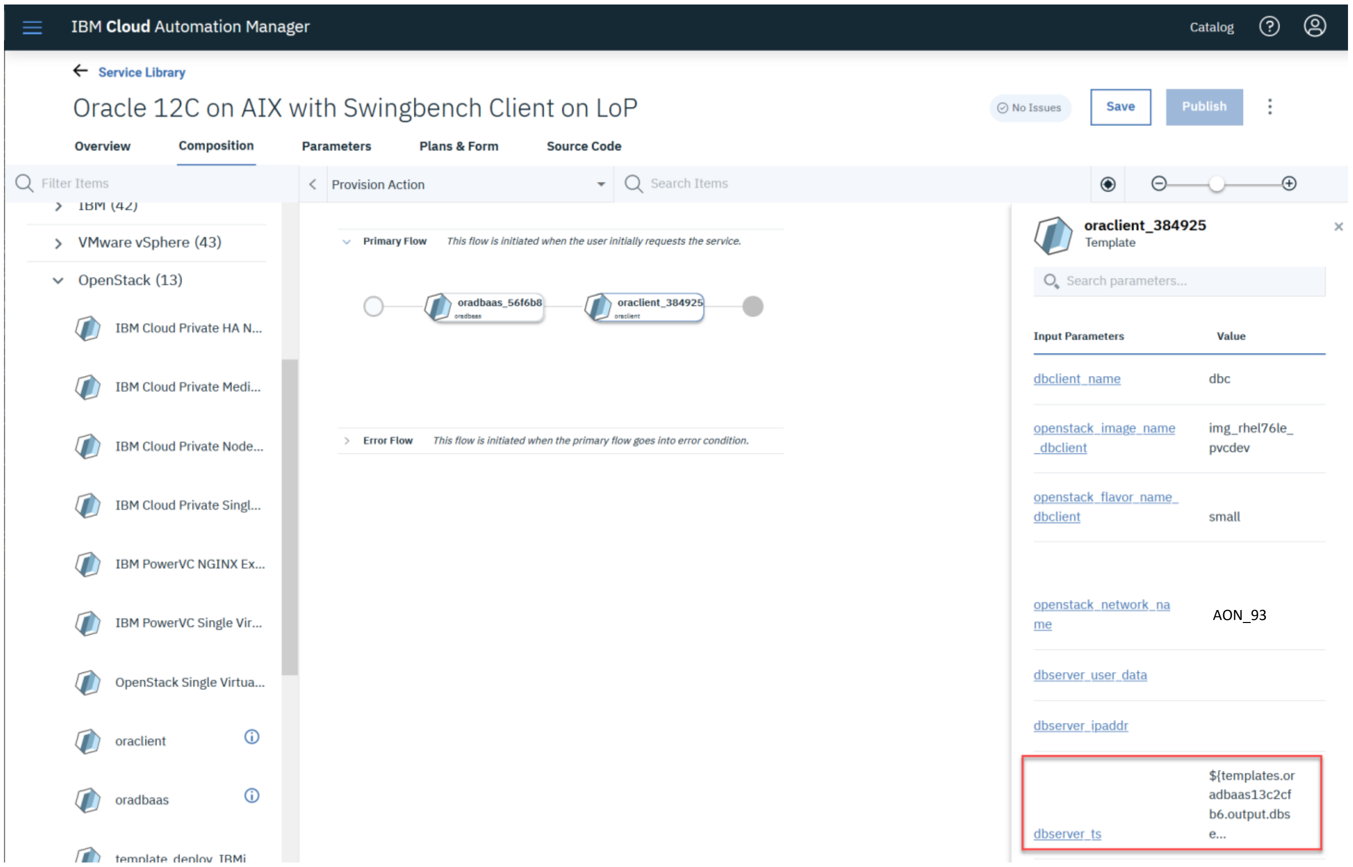
Click the plus sign and the check mark appears (signifying selection). Click the Save button.


Link Parameter

Parameter Key	dbserver_ts
Parameter Type	string

openstack_network_id	a40b-7008d9c06614
dbserver_user_data	${input_parameters.dbserver_user_data}
dbserver_ipaddr (output)	
dbserver_vm_name (output)	
dbserver_ts (output)	
dbname (output)	

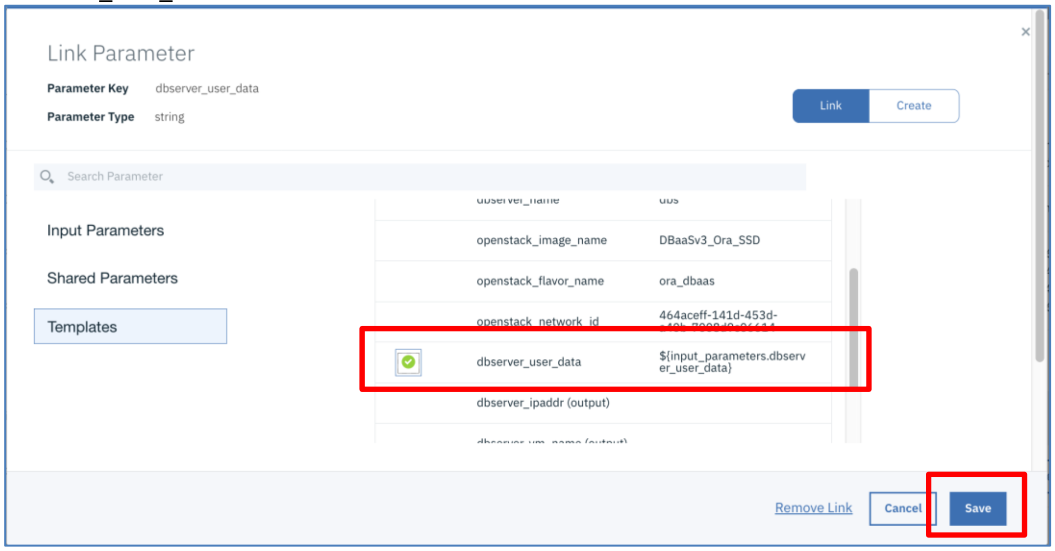
Notice that the value for **dbserver_ts** now points to the Activity ID of the *oradbaas* template. This step connects the output from *oradbaas* template to the input of *oraclient* template, allowing the VM name of the dbclient to have the same timestamp as that of dbserver.


IBM Cloud Automation Manager
Catalog
Service Library
Oracle 12C on AIX with Swingbench Client on LoP
No Issues
Save
Publish
Overview
Composition
Parameters
Plans & Form
Source Code
Filter Items
Provision Action
Search Items
IBM (42)
VMware vSphere (43)
OpenStack (13)
IBM Cloud Private HA N...
IBM Cloud Private Medi...
IBM Cloud Private Node...
IBM Cloud Private Singl...
IBM PowerVC NGINX Ex...
IBM PowerVC Single Vir...
OpenStack Single Virtua...
oraclient
oradbaas
template deploy IBMi
Primary Flow
This flow is initiated when the user initially requests the service.
oradbaas_56f6b8
oraclient_384925
Error Flow
This flow is initiated when the primary flow goes into error condition.
oraclient_384925
Template
Search parameters...

Input Parameters	Value
dbclient_name	dbc
openstack_image_name_dbclient	img_rhel76le_pvcdev
openstack_flavor_name_dbclient	small
openstack_network_name	AON_93
dbserver_user_data	
dbserver_ipaddr	
dbserver_ts	${templates.oradbaas13c2cfb6.output.dbse...

Repeat the above steps for dbserver_user_data and dbserver_ipaddr (only the pop-up windows for the Link Parameter step are shown).

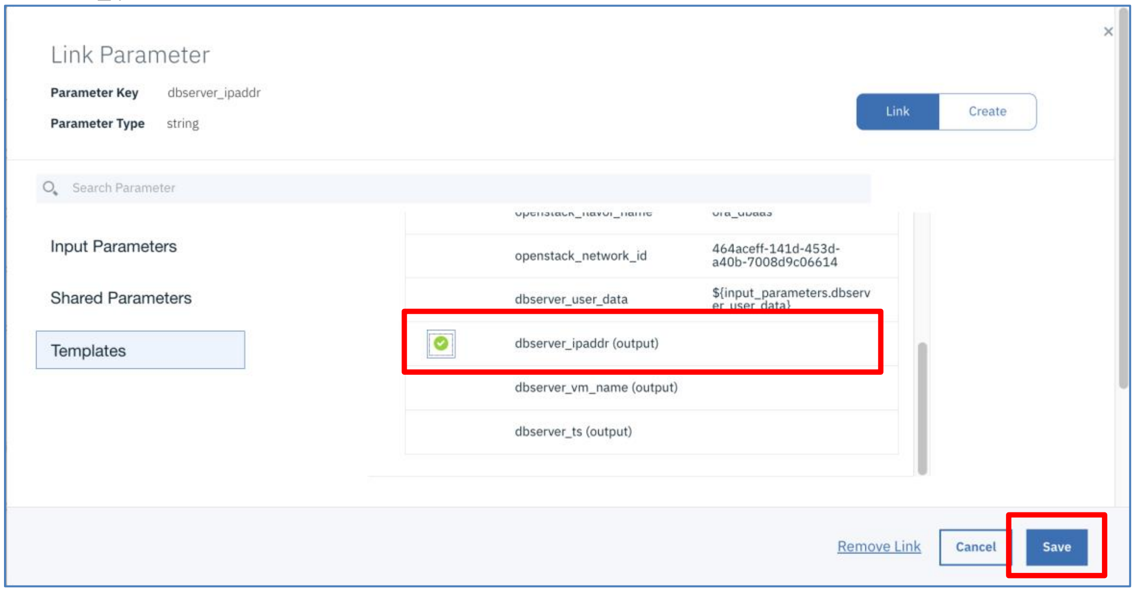
dbserver_user_data:



Parameter Key dbserver_user_data
Parameter Type string

	dbserver_name	dbs
	openstack_image_name	DBaaSv3_Ora_SSD
	openstack_flavor_name	ora_dbaas
	openstack_network_id	464aceff-141d-453d-
✓	dbserver_user_data	${input_parameters.dbserver_user_data}
	dbserver_ipaddr (output)	
	dbserver_vm_name (output)	

dbserver_ipaddr:

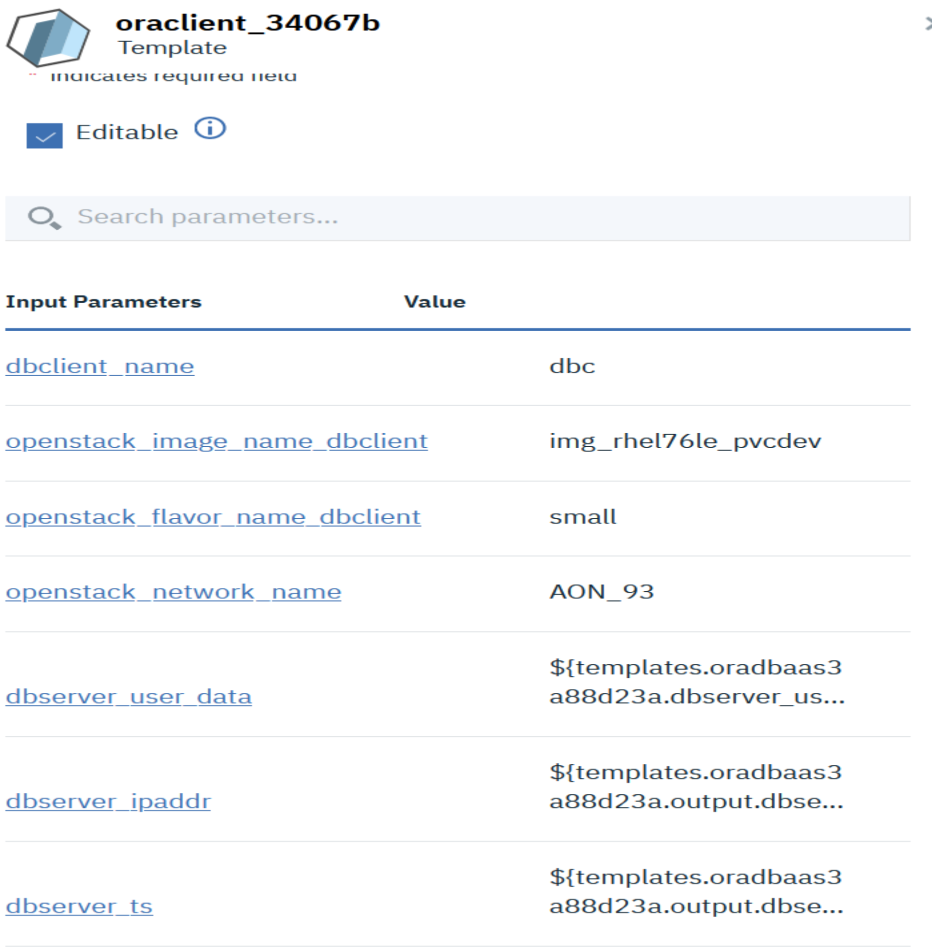


Parameter Key dbserver_ipaddr
Parameter Type string

openstack_network_id	464aceff-141d-453d-a40b-7008d9c06614
dbserver_user_data	${input_parameters.dbserver_user_data}
✓ dbserver_ipaddr (output)	
dbserver_vm_name (output)	
dbserver_ts (output)	

When done, the input parameters for *oraclient* template will look like the screen below.

In the screen shot below you also see the defined Output Parameters. In section 9.2.2, we create the Service's Output Parameters, which will display these parameters on the Service detailed information page.

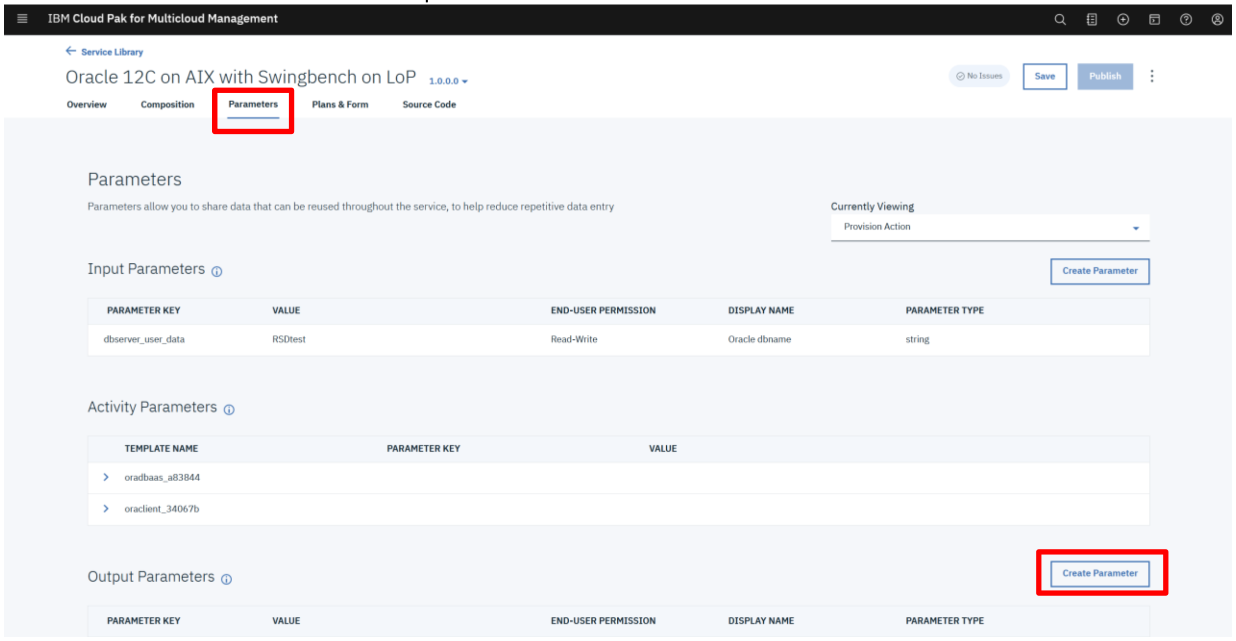


Input Parameters	Value
dbclient_name	dbc
openstack_image_name_dbclient	img_rhel76le_pvcdev
openstack_flavor_name_dbclient	small
openstack_network_name	AON_93
dbserver_user_data	${templates.oradbaas3a88d23a.dbserver_us...
dbserver_ipaddr	${templates.oradbaas3a88d23a.output.dbse...
dbserver_ts	${templates.oradbaas3a88d23a.output.dbse...

9.2.2 Create Service Output Parameters

An end user needs to be able to easily find the access information for the deployed service. To provide that data we create service output parameters which will be displayed on the deployed service screen.

On the service creation screen select the **Parameters** tab and then click the **Create Parameter** button for the **Output Parameter** section as indicated on the previous screen.



Parameters

Parameters allow you to share data that can be reused throughout the service, to help reduce repetitive data entry

Currently Viewing: Provision Action

Input Parameters — Create Parameter

PARAMETER KEY	VALUE	END-USER PERMISSION	DISPLAY NAME	PARAMETER TYPE
dbserver_user_data	RSDtest	Read-Write	Oracle dbname	string

Activity Parameters

TEMPLATE NAME	PARAMETER KEY	VALUE
oradbaas_a83844		
oraclient_34067b		

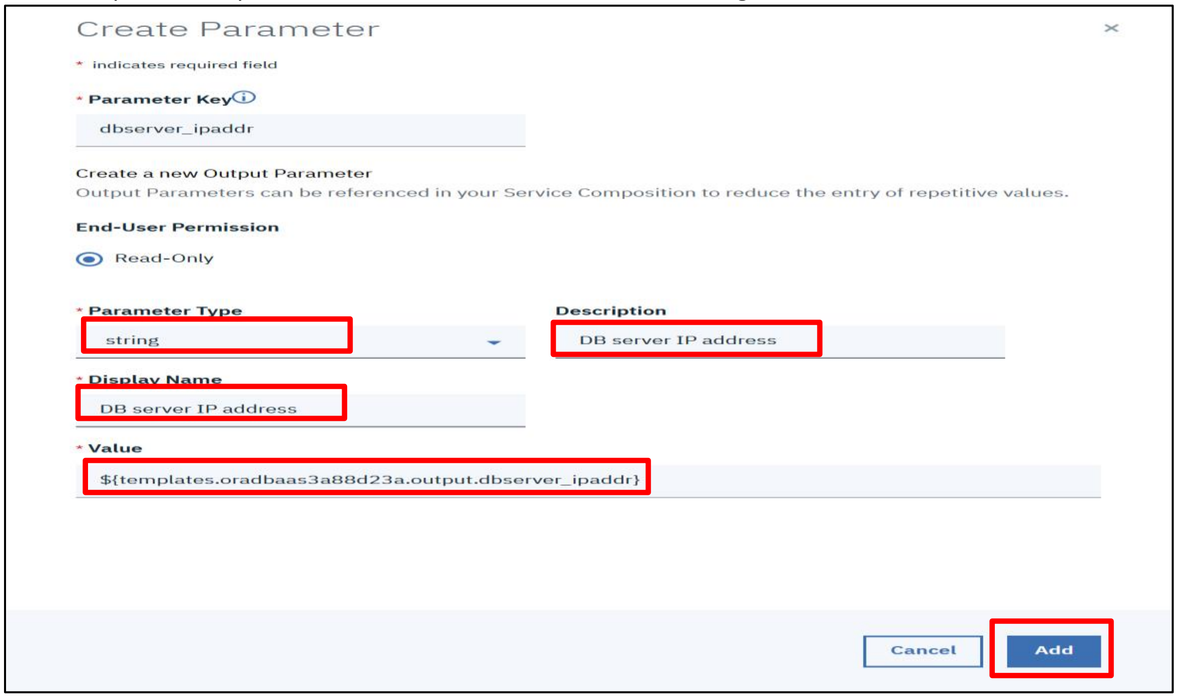
Output Parameters — Create Parameter

PARAMETER KEY	VALUE	END-USER PERMISSION	DISPLAY NAME	PARAMETER TYPE

Fill in the parameter shown below. Here we will need the Activity IDs saved earlier. If you need to find them again, go back to the **Composition** tab and select **Basic Information**, after selecting the correct template on the canvas.
Just to repeat – these Activity IDs will be different in your environment and need to be adjusted!

Template	Activity Id
oradbaas	oradbaas3a88d23a
oraclient	oracliena56de85c

For new output parameter dbserver_ipaddr, complete the definition screen as shown below. Note that you will need to substitute your Activity ID for ***oradbaas*** in the **Value** field. After filling the fields in click **Add**.



Create Parameter

* indicates required field

* Parameter Key: dbserver_ipaddr

Create a new Output Parameter
Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission: ● Read-Only

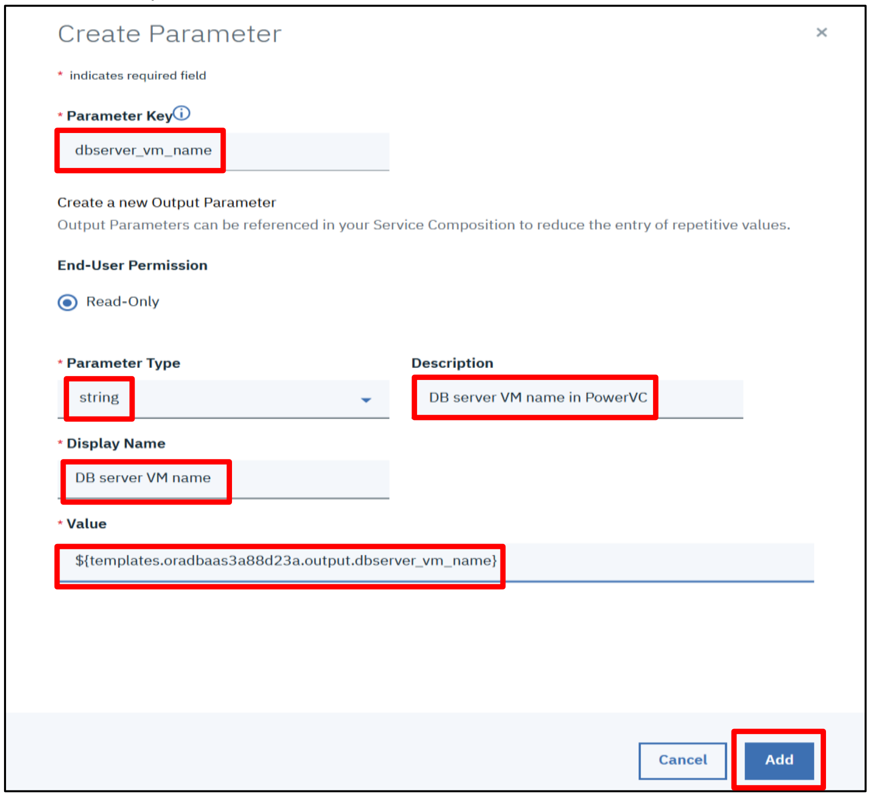
* Parameter Type: string

Description: DB server IP address

* Display Name: DB server IP address

* Value: ${templates.oradbaas3a88d23a.output.dbserver_ipaddr}

For new output parameter dbserver_vm_name, again ensure to replace the Activity ID with the correct value from your ***oradbaas*** template in **Value**.



Create Parameter

* indicates required field

* **Parameter Key**

dbserver_vm_name

Create a new Output Parameter

Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission

● Read-Only

* **Parameter Type**

string

Description

DB server VM name in PowerVC

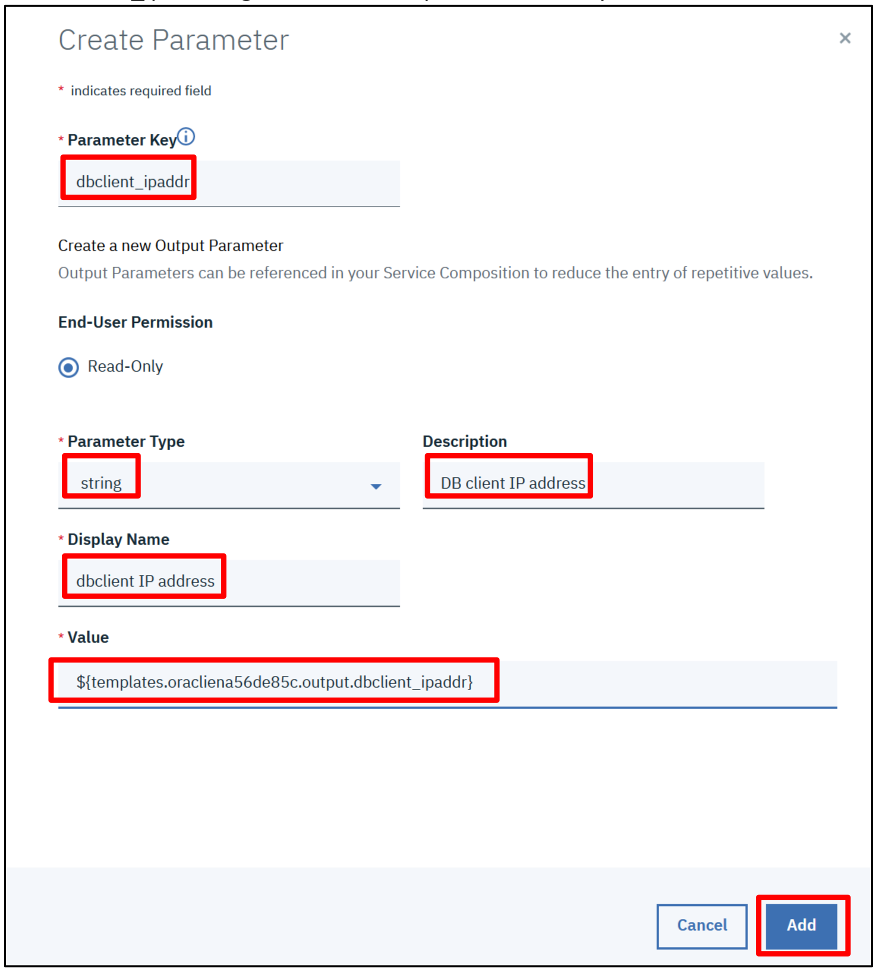
* **Display Name**

DB server VM name

* **Value**

${templates.oradbaas3a88d23a.output.dbserver_vm_name}

For dbclient_ipaddr, again ensure to replace the Activity ID with the correct value from your ***oraclient*** template in **Value**.



Create Parameter

* indicates required field

* **Parameter Key**

dbclient_ipaddr

Create a new Output Parameter

Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission

● Read-Only

* **Parameter Type**

string

Description

DB client IP address

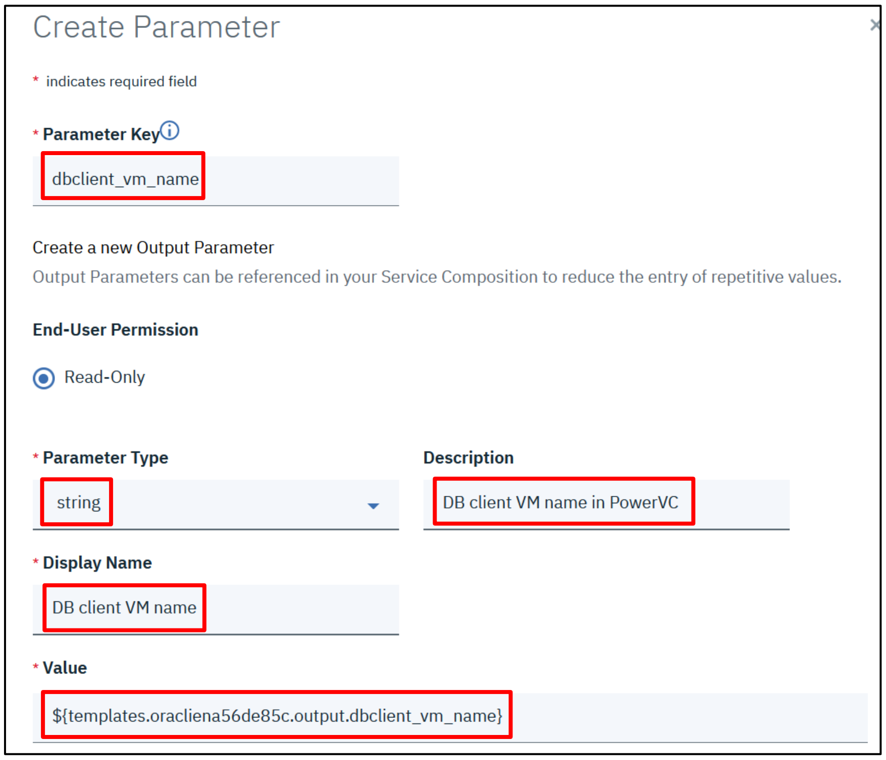
* **Display Name**

dbclient IP address

* **Value**

${templates.oracliena56de85c.output.dbclient_ipaddr}

For dbclient_vm_name, again ensure to replace the Activity ID with the correct value from your ***oraclient*** template in **Value**.



Create Parameter

* indicates required field

* **Parameter Key**

dbclient_vm_name

Create a new Output Parameter

Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission

● Read-Only

* **Parameter Type**

string

Description

DB client VM name in PowerVC

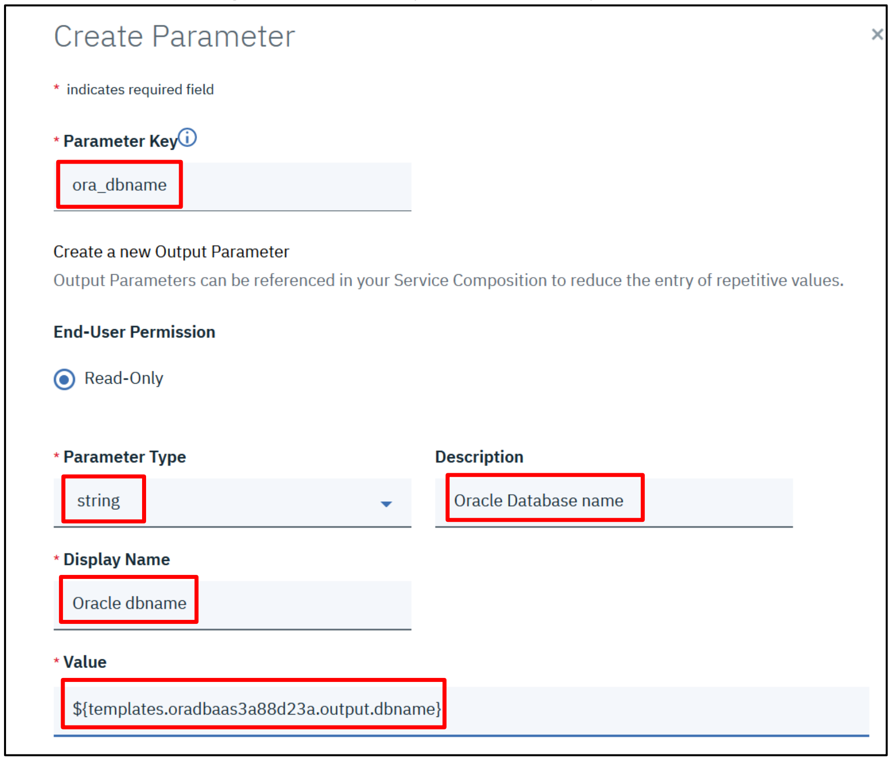
* **Display Name**

DB client VM name

* **Value**

${templates.oracliena56de85c.output.dbclient_vm_name}

For Oracle dbname, again ensure to replace the Activity ID with the correct value from your ***oradbaas*** template in **Value**.



Create Parameter

* indicates required field

* **Parameter Key**

ora_dbname

Create a new Output Parameter

Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission

● Read-Only

* **Parameter Type**

string

Description

Oracle Database name

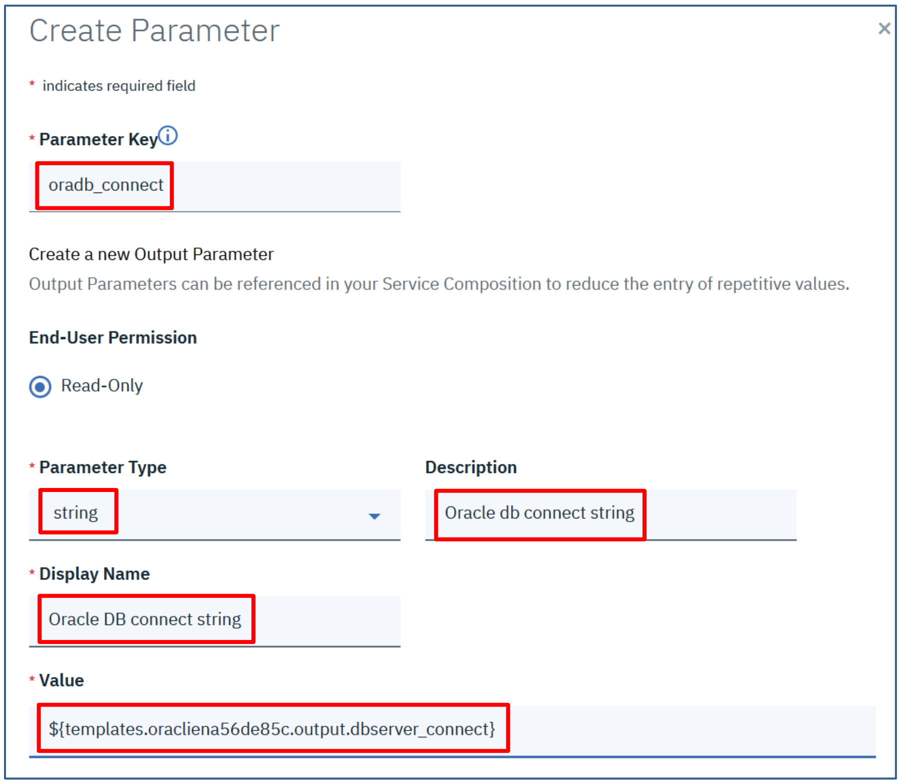
* **Display Name**

Oracle dbname

* **Value**

${templates.oradbaas3a88d23a.output.dbname}

And finally, for Oracle database connect string, be sure to replace the Activity ID with the correct value from your ***oraclient*** template in **Value**.


Create Parameter

* indicates required field

* **Parameter Key**

oradb_connect

Create a new Output Parameter

Output Parameters can be referenced in your Service Composition to reduce the entry of repetitive values.

End-User Permission

(●) Read-Only

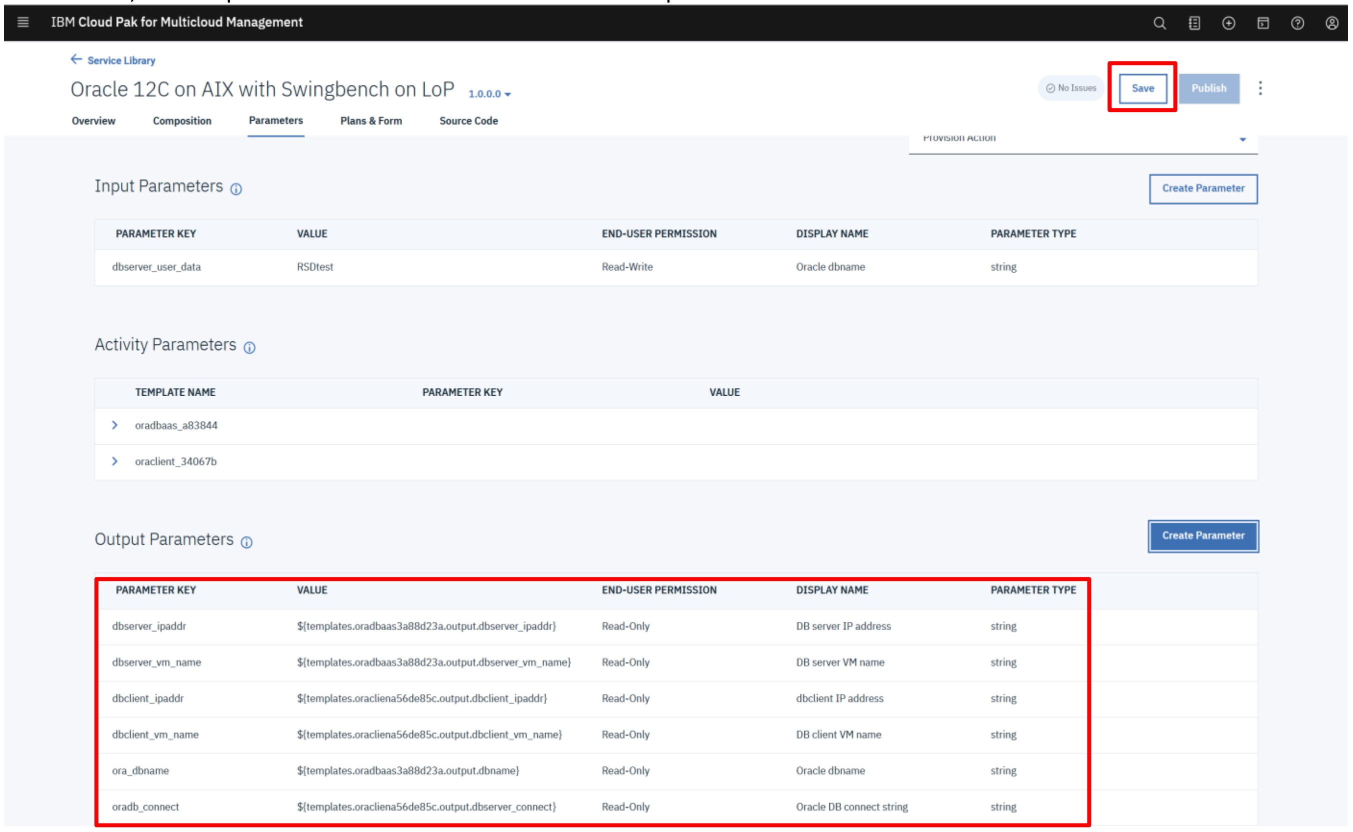
* **Parameter Type**: string

Description: Oracle db connect string

* **Display Name**: Oracle DB connect string

* **Value**: ${templates.oracliena56de85c.output.dbserver_connect}

When done, the Output Parameters section will look like the picture below.



Oracle 12C on AIX with Swingbench on LoP 1.0.0.0

Input Parameters

PARAMETER KEY	VALUE	END-USER PERMISSION	DISPLAY NAME	PARAMETER TYPE
dbserver_user_data	RSDtest	Read-Write	Oracle dbname	string

Activity Parameters

TEMPLATE NAME	PARAMETER KEY	VALUE
oradbaas_a83844		
oraclient_34067b		

Output Parameters

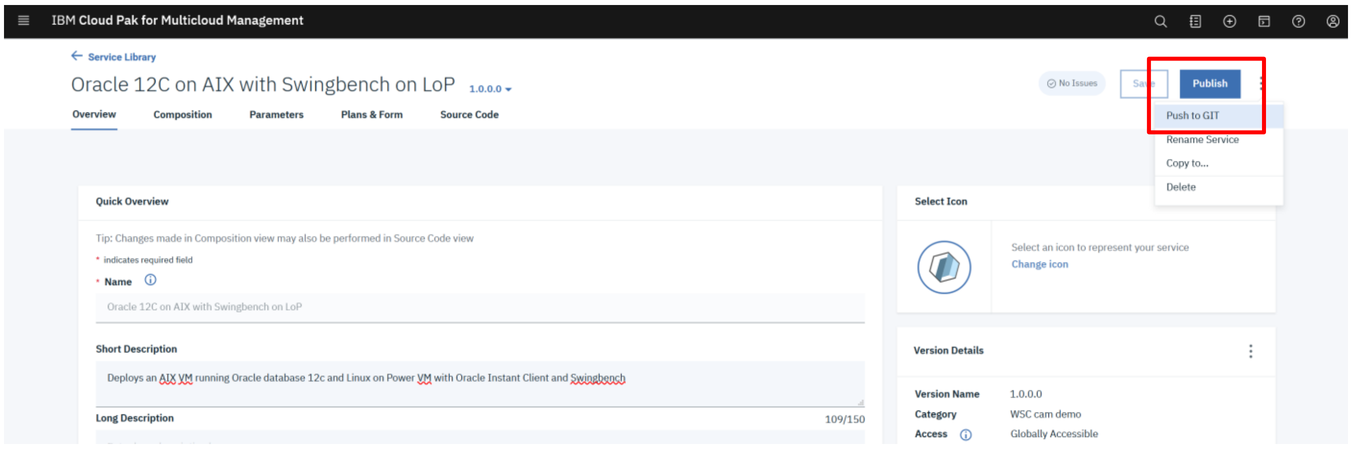
PARAMETER KEY	VALUE	END-USER PERMISSION	DISPLAY NAME	PARAMETER TYPE
dbserver_ipaddr	${templates.oradbaas3a88d23a.output.dbserver_ipaddr}	Read-Only	DB server IP address	string
dbserver_vm_name	${templates.oradbaas3a88d23a.output.dbserver_vm_name}	Read-Only	DB server VM name	string
dbclient_ipaddr	${templates.oracliena56de85c.output.dbclient_ipaddr}	Read-Only	dbclient IP address	string
dbclient_vm_name	${templates.oracliena56de85c.output.dbclient_vm_name}	Read-Only	DB client VM name	string
ora_dbname	${templates.oradbaas3a88d23a.output.dbname}	Read-Only	Oracle dbname	string
oradb_connect	${templates.oracliena56de85c.output.dbserver_connect}	Read-Only	Oracle DB connect string	string

Click the **Save** button as indicated above.
At this point you may want to Save (Push) the Service to a GitHub repository as shown in the next section Push Service to GitHub 9.3.

9.3 Push Service to GitHub

To facilitate distribution and collaboration without having to rebuild the service from scratch each time, push the service to a GitHub repository. To do this, access the Service Library and edit the service.

Click the three-dot menu next to the Publish button and select Push to GIT as shown below.


IBM Cloud Pak for Multicloud Management

← Service Library

Oracle 12C on AIX with Swingbench on LoP 1.0.0.0

Quick Overview

Tip: Changes made in Composition view may also be performed in Source Code view

* indicates required field

* Name: Oracle 12C on AIX with Swingbench on LoP

Short Description: Deploys an AIX VM running Oracle database 12c and Linux on Power VM with Oracle Instant Client and Swingbench

Long Description 109/150

Select Icon

Select an icon to represent your service

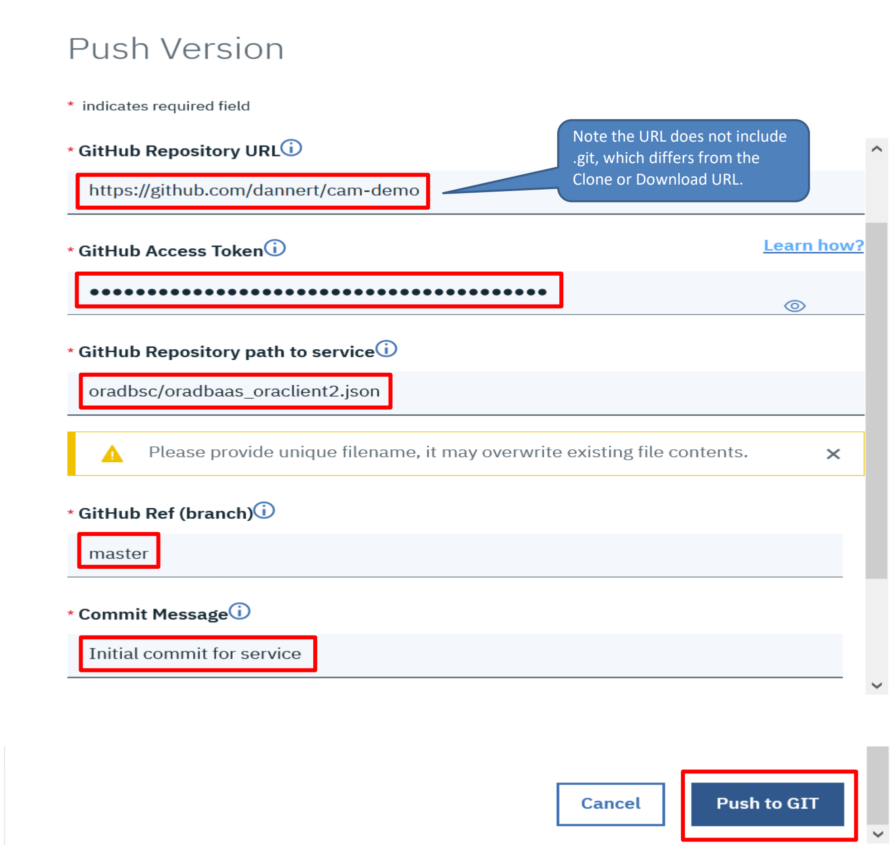
Change icon

Version Details

Version Name	1.0.0.0
Category	WSC cam demo
Access	Globally Accessible

Since we created the service using Service Composer, rather than importing from Git, we need to provide the Git repository details.

In our scenario, we created a new folder named **oradbsc** in the cam-demo project via GitHub WEB interface first and then performed the **Push to GIT**. On the pop-up screen for Push Version enter the appropriate repository details as shown in the example below. Be sure to also enter an appropriate Commit Message.


Push Version

* indicates required field

* GitHub Repository URL: https://github.com/dannert/cam-demo

Note the URL does not include .git, which differs from the Clone or Download URL.

* GitHub Access Token: ••

* GitHub Repository path to service: oradbsc/oradbaas_oraclient2.json

Please provide unique filename, it may overwrite existing file contents.

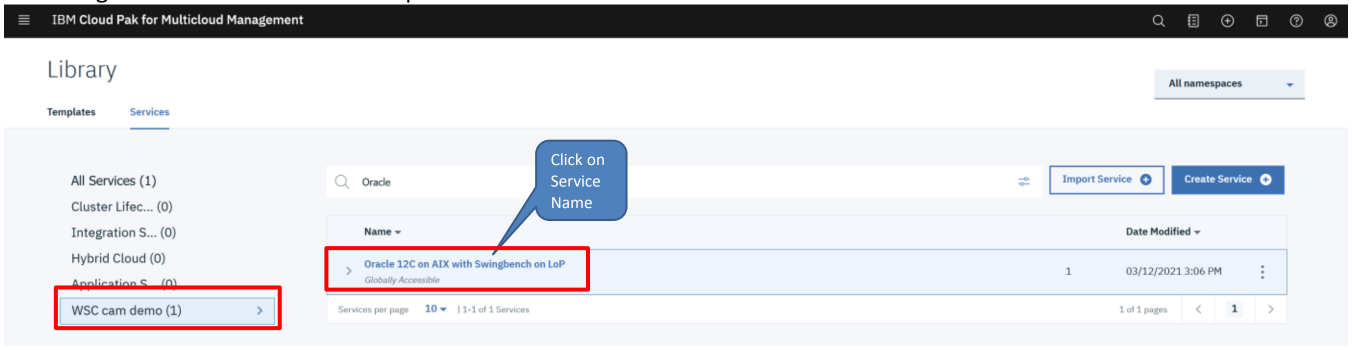
* GitHub Ref (branch): master

* Commit Message: Initial commit for service

9.4 Deploying the Service

Return to the Service Library by clicking the link on the left-hand upper corner. Alternatively, click the hamburger menu, then Library->Services. Select “WSC cam demo” to down-select to the correct service entry.

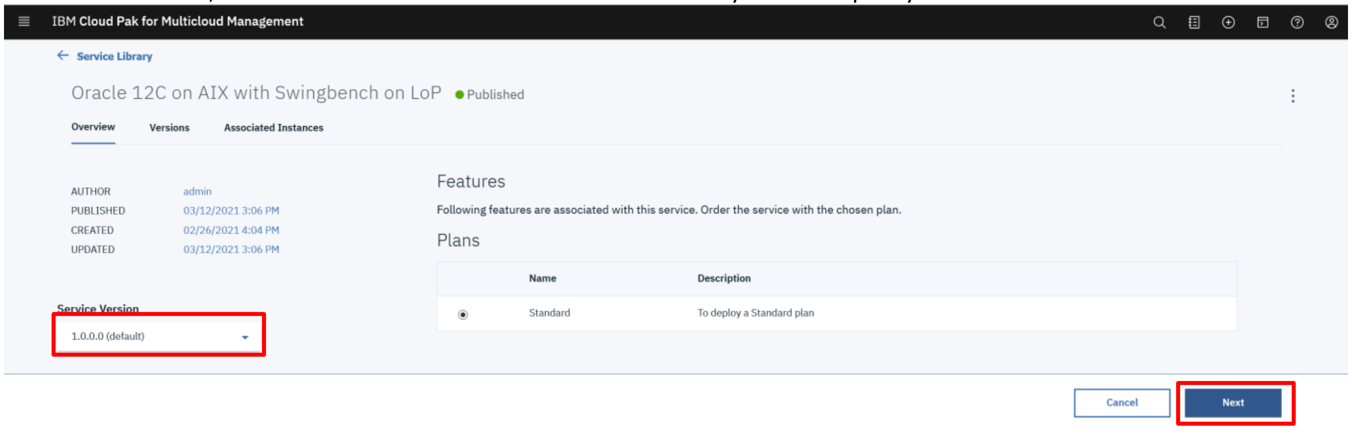
Then “Right Click” on the service to open the next menu.



Name		Date Modified
Oracle 12C on AIX with Swingbench on LoP *Globally Accessible*	1	03/12/2021 3:06 PM

Services per page 10 | 1-1 of 1 Services — 1 of 1 pages

On the next screen, click the **Next** button. Note that on this screen you could specify the service version.


IBM Cloud Pak for Multicloud Management
Service Library
Oracle 12C on AIX with Swingbench on LoP
Published
Overview
Versions
Associated Instances
Cancel
Next

AUTHOR	admin
PUBLISHED	03/12/2021 3:06 PM
CREATED	02/26/2021 4:04 PM
UPDATED	03/12/2021 3:06 PM

Service Version: 1.0.0.0 (default)

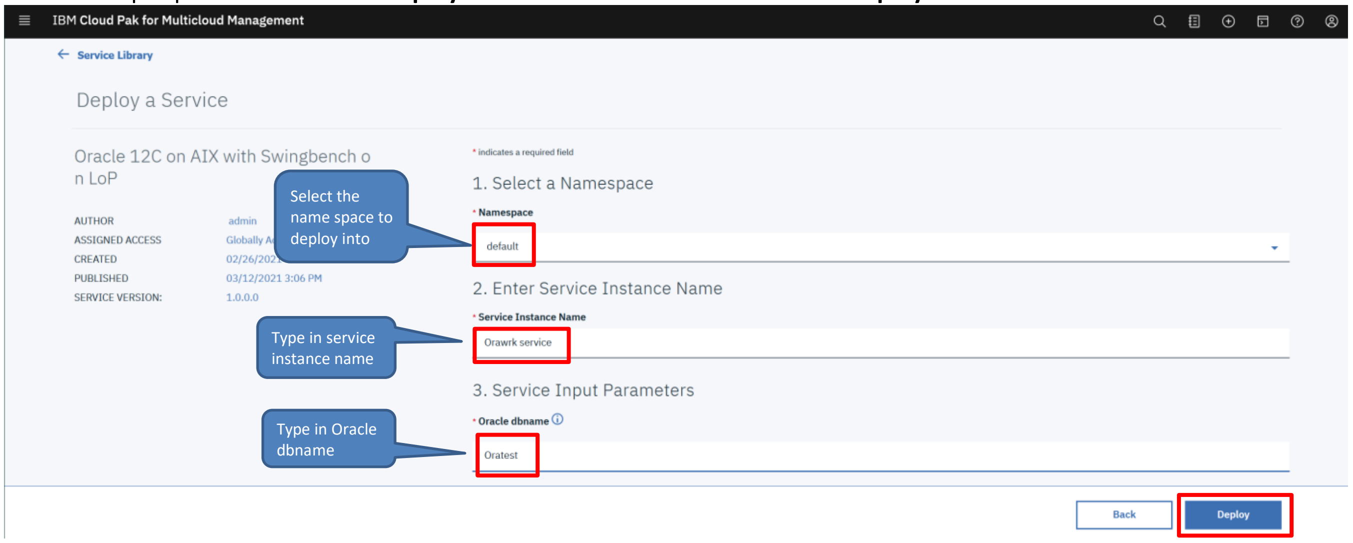
Features

Following features are associated with this service. Order the service with the chosen plan.

Plans

	Name	Description
●	Standard	To deploy a Standard plan

Enter the input parameters on the **Deploy a Service** screen and then click the **Deploy** button.


IBM Cloud Pak for Multicloud Management
Service Library
Deploy a Service
Oracle 12C on AIX with Swingbench on LoP
Select the name space to deploy into
Type in service instance name
Type in Oracle dbname
Back
Deploy

AUTHOR	admin
ASSIGNED ACCESS	Globally A
CREATED	02/26/202
PUBLISHED	03/12/2021 3:06 PM
SERVICE VERSION:	1.0.0.0

* indicates a required field

1. Select a Namespace

* Namespace: default

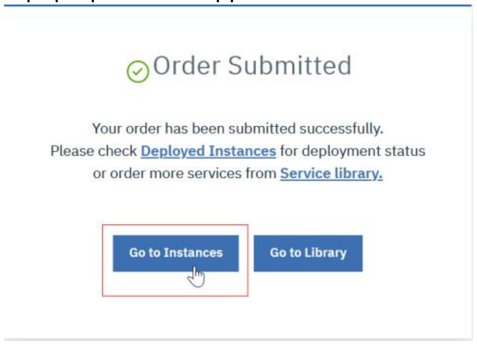
2. Enter Service Instance Name

* Service Instance Name: Orawrk service

3. Service Input Parameters

* Oracle dbname: Oratest

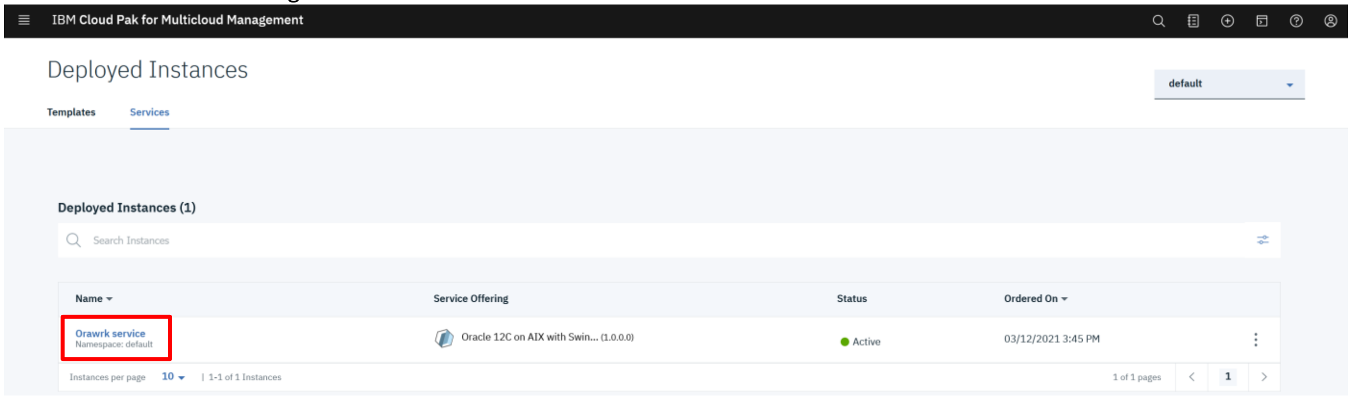
A pop-up window appears to indicate order submission. Click the **Go to Instances** button.



Order Submitted

Your order has been submitted successfully.
Please check Deployed Instances for deployment status
or order more services from Service library.

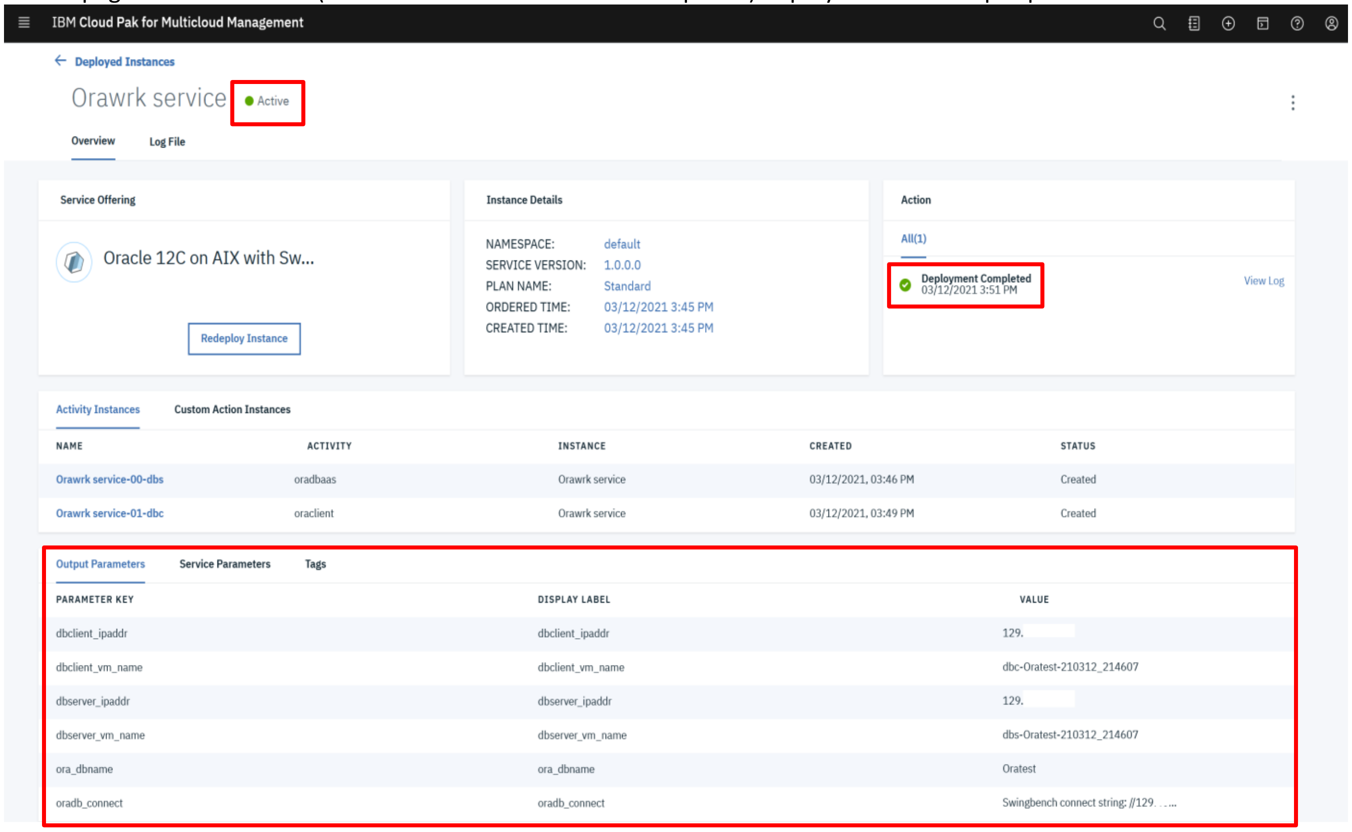
The **Deployed Instances** screen is displayed. Note the name under Name, **Orawrk service**, which we entered earlier. Click on **Orawrk service** for more details. Note that the screen shot was taken after service was already deployed. The initial status would show “In Progress”.



Name	Service Offering	Status	Ordered On
Orawrk service Namespace: default	Oracle 12C on AIX with Swin... (1.0.0.0)	● Active	03/12/2021 3:45 PM

Instances per page 10 | 1-1 of 1 Instances 1 of 1 pages 1

At this point, the *oradbaas* template is being deployed, and after successful service deploy and in status Active, the Service details page shows two VMs (from *oradbaas* and *oraclient* templates) deployed and all output parameters are filled in.



NAMESPACE: default
SERVICE VERSION: 1.0.0.0
PLAN NAME: Standard
ORDERED TIME: 03/12/2021 3:45 PM
CREATED TIME: 03/12/2021 3:45 PM

Deployment Completed 03/12/2021 3:51 PM

Activity Instances | Custom Action Instances

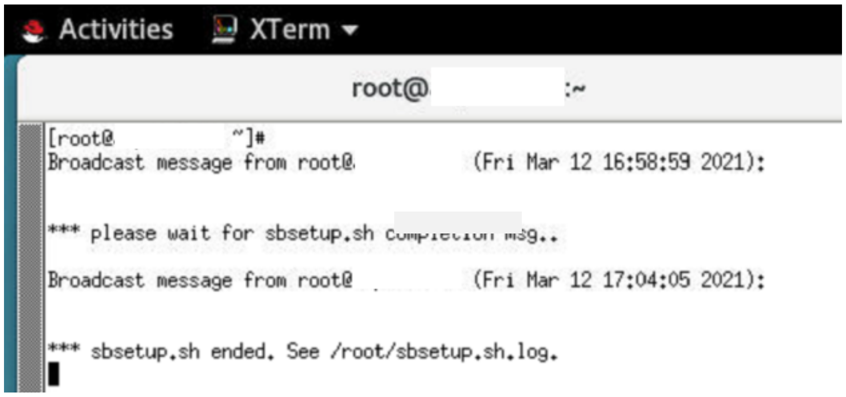
NAME	ACTIVITY	INSTANCE	CREATED	STATUS
Orawrk service-00-dbs	oradbaas	Orawrk service	03/12/2021, 03:46 PM	Created
Orawrk service-01-dbc	oraclient	Orawrk service	03/12/2021, 03:49 PM	Created

Output Parameters | Service Parameters | Tags

PARAMETER KEY	DISPLAY LABEL	VALUE
dbclient_ipaddr	dbclient_ipaddr	129.
dbclient_vm_name	dbclient_vm_name	dbc-Oratest-210312_214607
dbserver_ipaddr	dbserver_ipaddr	129.
dbserver_vm_name	dbserver_vm_name	dbs-Oratest-210312_214607
ora_dbname	ora_dbname	Oratest
oradb_connect	oradb_connect	Swingbench connect string: //129. . . ...

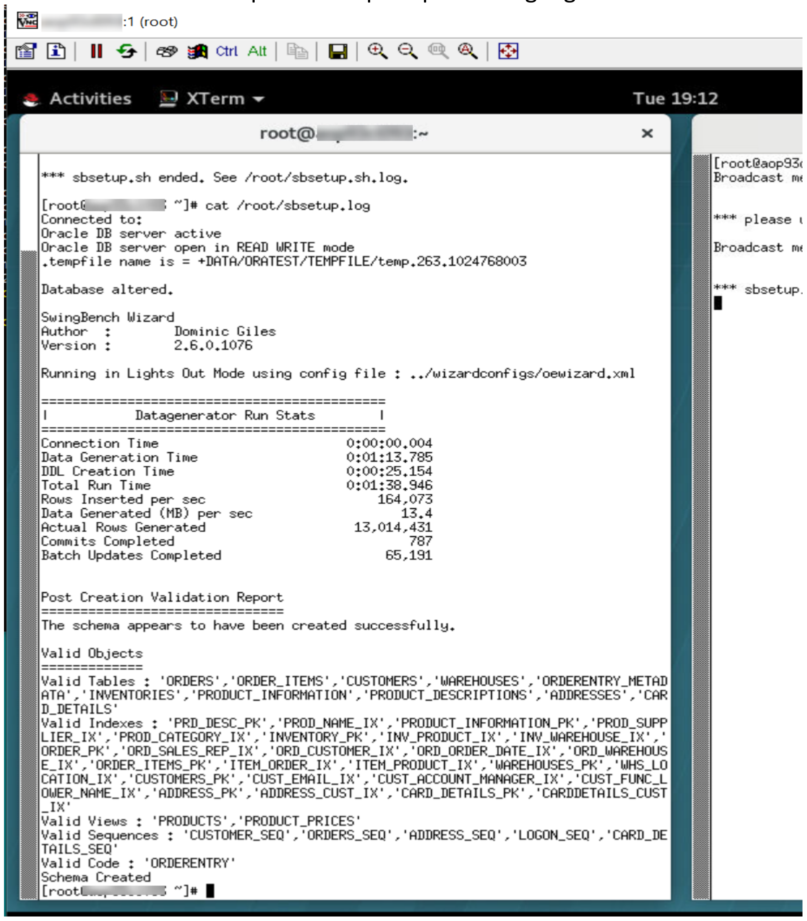
9.5 Test the deployed service

From the parameter output in the services screen we can utilize the value reported in dbclient_ipaddr to open a VNC viewer session to the dbclient server – <ip address>:1. This gets us to the X-session with an xterm running as root user in this case.
Note that the service deployment in T&SA will show as complete significantly before all post-processing in the dbclient server is actually complete. The service deployment, from the viewpoint of T&SA and PowerVC, is complete with the successful creation and start of the dbclient VM, regardless of the success of any post-deployment steps.
To account for this behavior, the client publishes the start and stop messages of when the data load started and when it completed. Only after completion, as shown below, can the swingbench driver workload be successfully executed.



```
[root@[redacted] ~]#
Broadcast message from root@[redacted] (Fri Mar 12 16:58:59 2021):

*** please wait for sbsetup.sh [redacted] msg..

Broadcast message from root@[redacted] (Fri Mar 12 17:04:05 2021):

*** sbsetup.sh ended. See /root/sbsetup.sh.log.
```

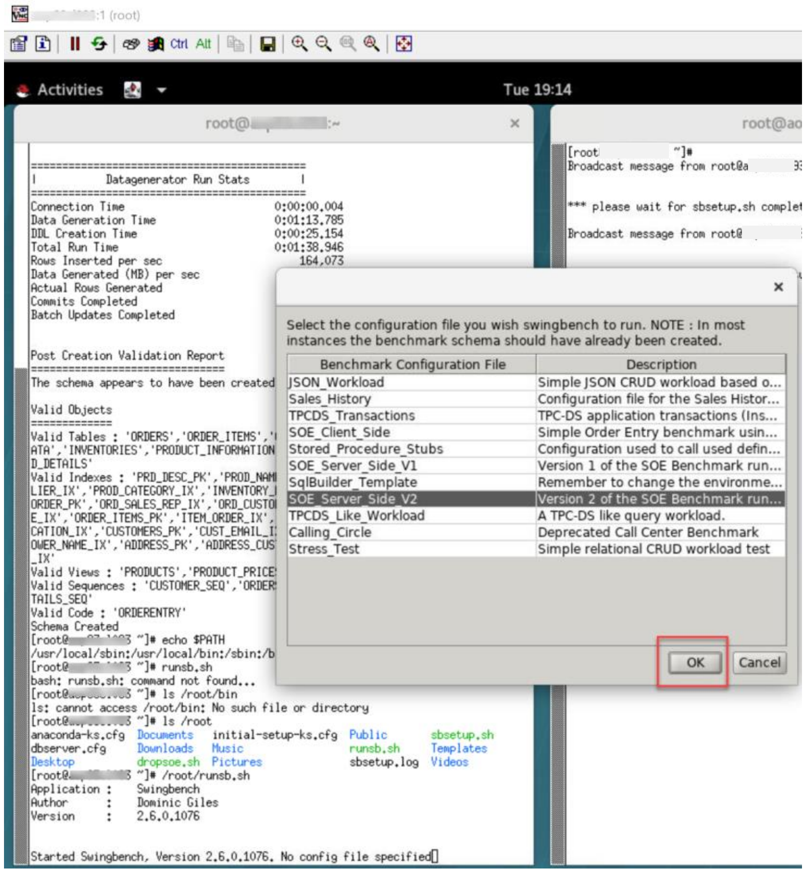
See below for an examp how the post-processing log looks like after successful deployment.



```
*** sbsetup.sh ended. See /root/sbsetup.sh.log.

[root@[redacted] ~]# cat /root/sbsetup.log
Connected to:
Oracle DB server active
Oracle DB server open in READ WRITE mode
.tempfile name is = +DATA/ORATEST/TEMPFILE/temp.263.1024768003

Database altered.

SwingBench Wizard
Author  :        Dominic Giles
Version :        2.6.0.1076

Running in Lights Out Mode using config file : ../wizardconfigs/oewizard.xml

============================================
|           Datagenerator Run Stats         |
============================================
Connection Time                           0:00:00.004
Data Generation Time                      0:01:13.785
DDL Creation Time                         0:00:25.154
Total Run Time                            0:01:38.946
Rows Inserted per sec                         164,073
Data Generated (MB) per sec                      13.4
Actual Rows Generated                      13,014,431
Commits Completed                                 787
Batch Updates Completed                        65,191


Post Creation Validation Report
===============================
The schema appears to have been created successfully.

Valid Objects
=============
Valid Tables : 'ORDERS','ORDER_ITEMS','CUSTOMERS','WAREHOUSES','ORDERENTRY_METAD
ATA','INVENTORIES','PRODUCT_INFORMATION','PRODUCT_DESCRIPTIONS','ADDRESSES','CAR
D_DETAILS'
Valid Indexes : 'PRD_DESC_PK','PROD_NAME_IX','PRODUCT_INFORMATION_PK','PROD_SUPP
LIER_IX','PROD_CATEGORY_IX','INVENTORY_PK','INV_PRODUCT_IX','INV_WAREHOUSE_IX','
ORDER_PK','ORD_SALES_REP_IX','ORD_CUSTOMER_IX','ORD_ORDER_DATE_IX','ORD_WAREHOUS
E_IX','ORDER_ITEMS_PK','ITEM_ORDER_IX','ITEM_PRODUCT_IX','WAREHOUSES_PK','WHS_LO
CATION_IX','CUSTOMERS_PK','CUST_EMAIL_IX','CUST_ACCOUNT_MANAGER_IX','CUST_FUNC_L
OWER_NAME_IX','ADDRESS_PK','ADDRESS_CUST_IX','CARD_DETAILS_PK','CARDDETAILS_CUST
_IX'
Valid Views : 'PRODUCTS','PRODUCT_PRICES'
Valid Sequences : 'CUSTOMER_SEQ','ORDERS_SEQ','ADDRESS_SEQ','LOGON_SEQ','CARD_DE
TAILS_SEQ'
Valid Code : 'ORDERENTRY'
Schema Created
[root@[redacted] ~]#
```

```
[root@aop93
Broadcast me

*** please

Broadcast me

*** sbsetup.
```

To illustrate that the service deployment achieved the expected end result, we ran the swingbench driver by executing the /root/runsb.sh script:



```
root@[redacted]:~

=============================================
|          Datagenerator Run Stats          |
=============================================
Connection Time                        0:00:00.004
Data Generation Time                   0:01:13.785
DDL Creation Time                      0:00:25.154
Total Run Time                         0:01:38.946
Rows Inserted per sec                      164,073
Data Generated (MB) per sec
Actual Rows Generated
Commits Completed
Batch Updates Completed


Post Creation Validation Report
===============================
The schema appears to have been created

Valid Objects
=============
Valid Tables : 'ORDERS','ORDER_ITEMS','
ATA','INVENTORIES','PRODUCT_INFORMATION
D_DETAILS'
Valid Indexes : 'PRD_DESC_PK','PROD_NAM
LIER_IX','PROD_CATEGORY_IX','INVENTORY_
ORDER_PK','ORD_SALES_REP_IX','ORD_CUSTO
E_IX','ORDER_ITEMS_PK','ITEM_ORDER_IX',
CATION_IX','CUSTOMERS_PK','CUST_EMAIL_I
OWER_NAME_IX','ADDRESS_PK','ADDRESS_CUS
_IX'
Valid Views : 'PRODUCTS','PRODUCT_PRICE
Valid Sequences : 'CUSTOMER_SEQ','ORDER
TAILS_SEQ'
Valid Code : 'ORDERENTRY'
Schema Created
[root@[redacted] ~]# echo $PATH
/usr/local/sbin:/usr/local/bin:/sbin:/b
[root@[redacted] ~]# runsb.sh
bash: runsb.sh: command not found...
[root@[redacted] ~]# ls /root/bin
ls: cannot access /root/bin: No such file or directory
[root@[redacted] ~]# ls /root
anaconda-ks.cfg  Documents  initial-setup-ks.cfg  Public       sbsetup.sh
dbserver.cfg     Downloads  Music                 runsb.sh     Templates
Desktop          dropsoe.sh Pictures              sbsetup.log  Videos
[root@[redacted] ~]# /root/runsb.sh
Application :   Swingbench
Author      :   Dominic Giles
Version     :   2.6.0.1076


Started Swingbench, Version 2.6.0.1076. No config file specified
```

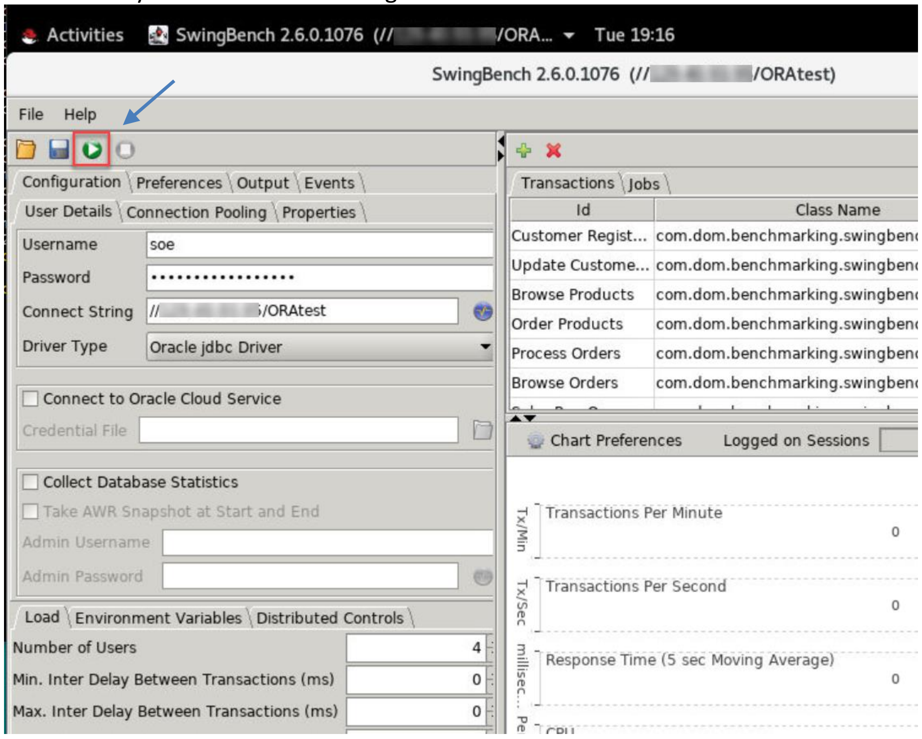
```
[root@[redacted] ~]#
Broadcast message from root@a[redacted]

*** please wait for sbsetup.sh complet

Broadcast message from root@[redacted]
```

Select the configuration file you wish swingbench to run. NOTE : In most instances the benchmark schema should have already been created.

Benchmark Configuration File	Description
JSON_Workload	Simple JSON CRUD workload based o...
Sales_History	Configuration file for the Sales Histor...
TPCDS_Transactions	TPC-DS application transactions (Ins...
SOE_Client_Side	Simple Order Entry benchmark usin...
Stored_Procedure_Stubs	Configuration used to call used defin...
SOE_Server_Side_V1	Version 1 of the SOE Benchmark run...
SqlBuilder_Template	Remember to change the environme...
SOE_Server_Side_V2	Version 2 of the SOE Benchmark run...
TPCDS_Like_Workload	A TPC-DS like query workload.
Calling_Circle	Deprecated Call Center Benchmark
Stress_Test	Simple relational CRUD workload test

Accept the selected default benchmark configuration file and click the OK button.
Note the connect string and other fields in the next screen have been pre-filled by the script.

Click the Play icon to start the load generator.



SwingBench 2.6.0.1076 (//[redacted]/ORAtest)

User Details:

Username	soe
Password	••••••••••••••••••
Connect String	//[redacted]/ORAtest
Driver Type	Oracle jdbc Driver

Connect to Oracle Cloud Service [ ]
Collect Database Statistics [ ]
Take AWR Snapshot at Start and End [ ]

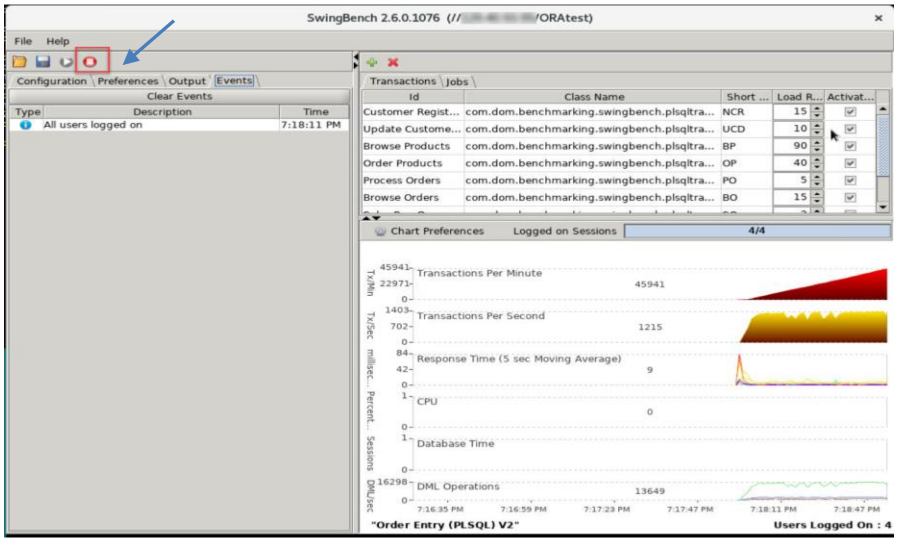
Load:

Number of Users	4
Min. Inter Delay Between Transactions (ms)	0
Max. Inter Delay Between Transactions (ms)	0

Transactions:

Id	Class Name
Customer Regist...	com.dom.benchmarking.swingbenc
Update Custome...	com.dom.benchmarking.swingbenc
Browse Products	com.dom.benchmarking.swingbenc
Order Products	com.dom.benchmarking.swingbenc
Process Orders	com.dom.benchmarking.swingbenc
Browse Orders	com.dom.benchmarking.swingbenc

And as a proof that it works you see that all users successfully logged in and are driving transactions against the database. At that point we stop the workload by clicking the stop button.


SwingBench 2.6.0.1076 (//[redacted]/ORAtest)

Type	Description	Time
●	All users logged on	7:18:11 PM

Id	Class Name	Short ...	Load R...	Activat...
Customer Regist...	com.dom.benchmarking.swingbench.plsqltra...	NCR	15	✓
Update Custome...	com.dom.benchmarking.swingbench.plsqltra...	UCD	10	✓
Browse Products	com.dom.benchmarking.swingbench.plsqltra...	BP	90	✓
Order Products	com.dom.benchmarking.swingbench.plsqltra...	OP	40	✓
Process Orders	com.dom.benchmarking.swingbench.plsqltra...	PO	5	✓
Browse Orders	com.dom.benchmarking.swingbench.plsqltra...	BO	15	✓

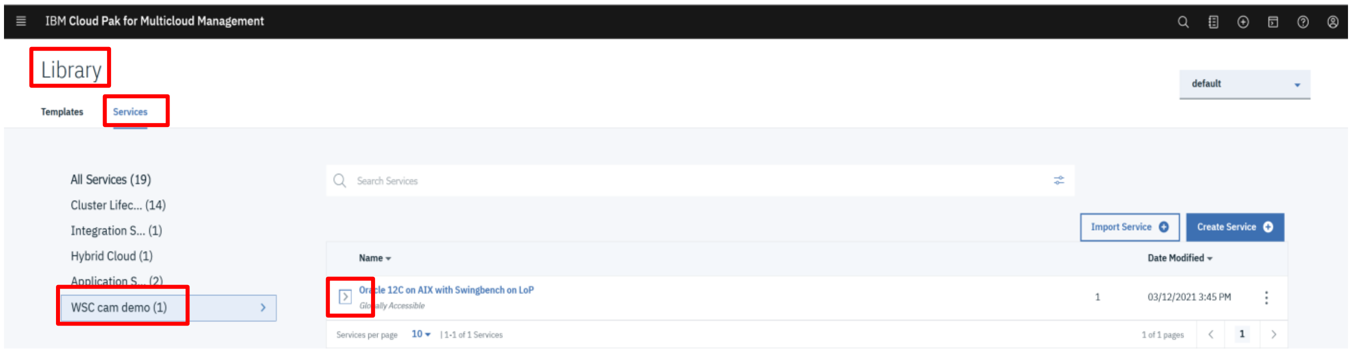
Logged on Sessions 4/4

"Order Entry (PLSQL) V2"   Users Logged On : 4

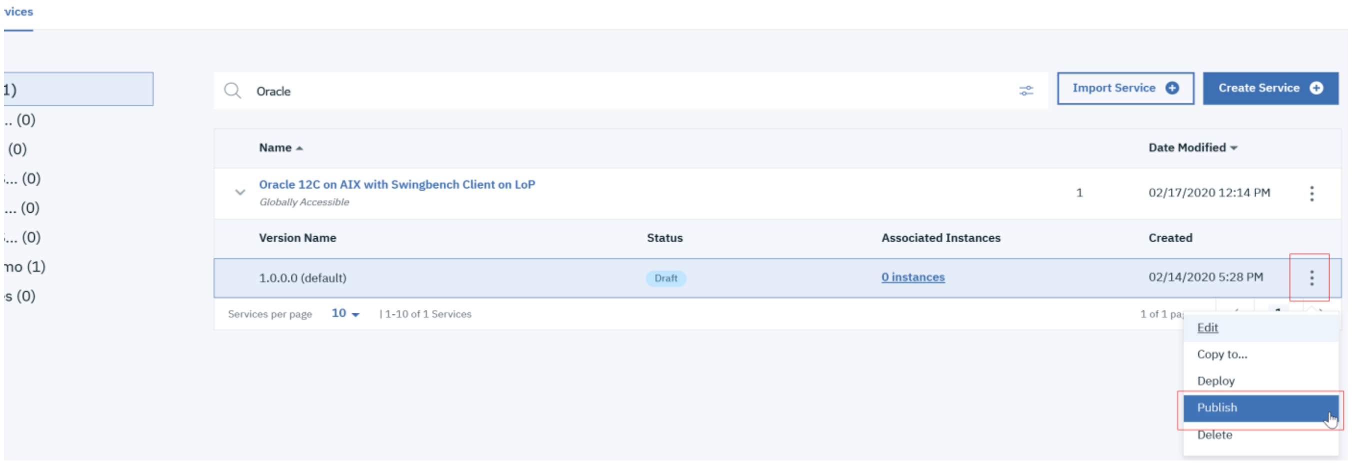
This completes the validation that our service, consisting of the dbserver and the dbclient, was successfully deployed and that parameter hand-over and orchestration worked as expected.

9.6 Publish the service to OCP Service Catalog and CP4MCM catalog

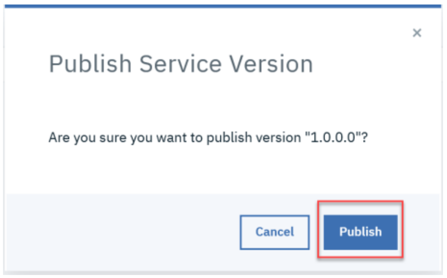
The service has so far been successfully deployed from the T&SA service library. Now we want to make this service also easily consumable for end-users of OCP via the Service Catalog.
To publish the service into the OCP Service Catalog, and the CP$MCM catalog, go in T&SA to Library -> Services, enter “Oracle” as filter and then click on the “>” character to the left of the service to expand the service version list



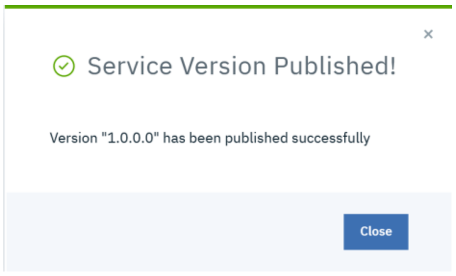
Then click on the three-dot menu to the right of the version of service you want to publish, in this case there is only one version, and then select Publish as shown in the screen shot below.



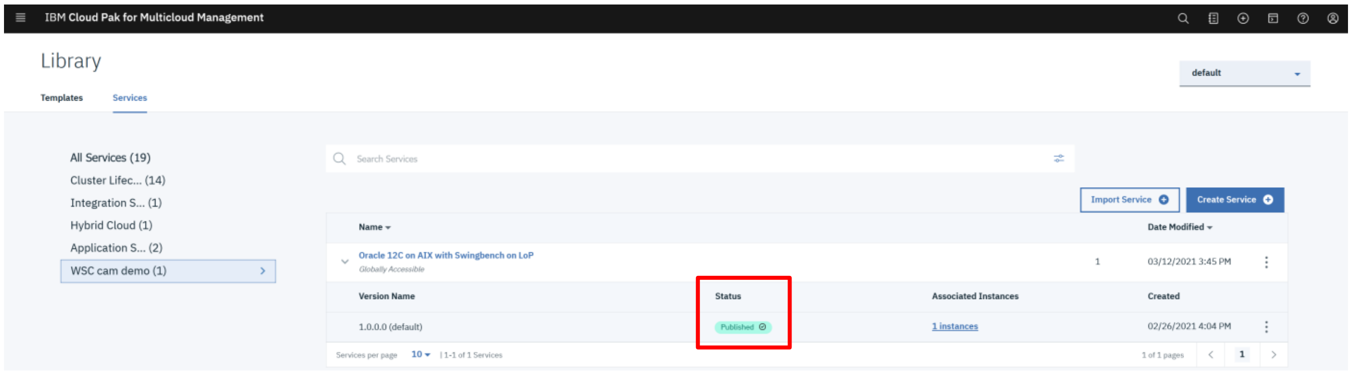
Confirm the Publish Service Version pop-up with **Publish.**


Publish Service Version

Are you sure you want to publish version "1.0.0.0"?


Service Version Published!

Version "1.0.0.0" has been published successfully

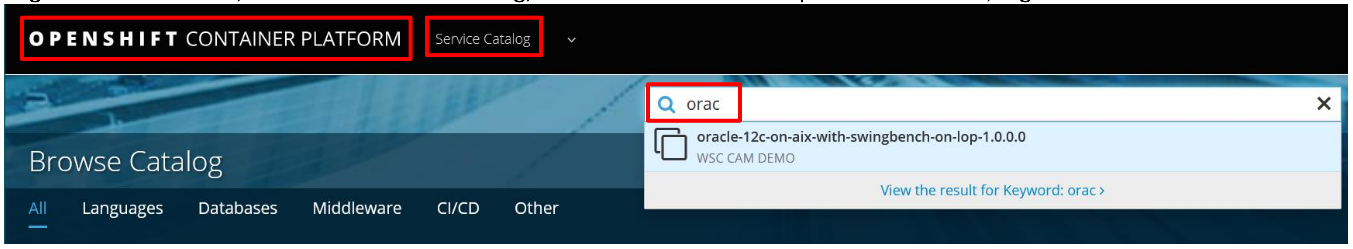


Name				Date Modified	
Oracle 12C on AIX with Swingbench on LoP *Globally Accessible*			1	03/12/2021 3:45 PM	⋮
Version Name	Status	Associated Instances		Created	
1.0.0.0 (default)	Published ✓	1 instances		02/26/2021 4:04 PM	⋮

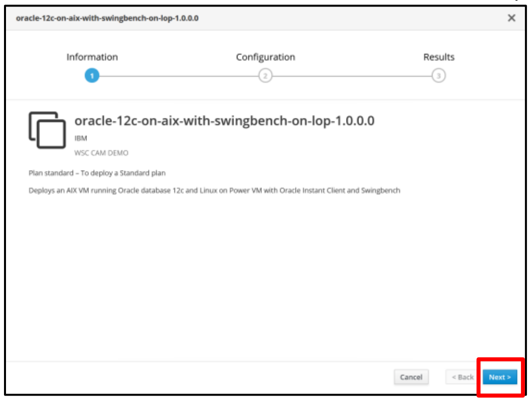
After the service version has been published the service can be deployed via the OCP Service Catalog or via the CP4MCM catalog as a Helm Chart. Wait a few minutes before trying to deploy the published service so the catalogs are updated.

In this document we only show the deployment via the OCP Service Catalog.

Log in to the OCP GUI, select the Service Catalog, and then search for the published service; e.g **orac**.


OPENSHIFT CONTAINER PLATFORM
Service Catalog
Browse Catalog
All
Languages
Databases
Middleware
CI/CD
Other
orac
oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0
WSC CAM DEMO
View the result for Keyword: orac ›

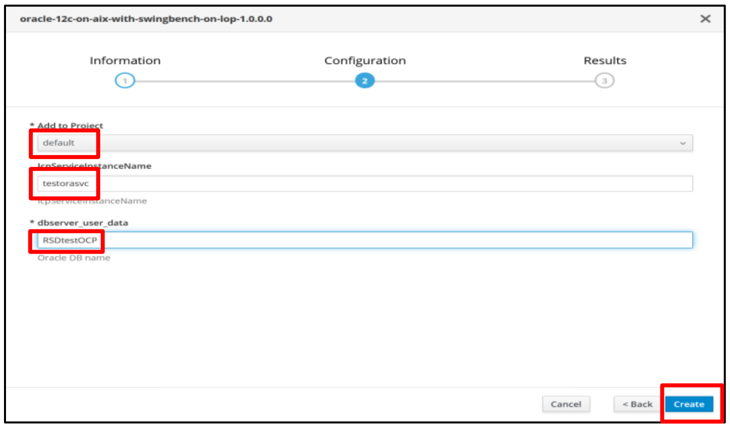
Click on the listed service and click Next on the pop-up window shown below.


oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0

oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0
IBM
WSC CAM DEMO

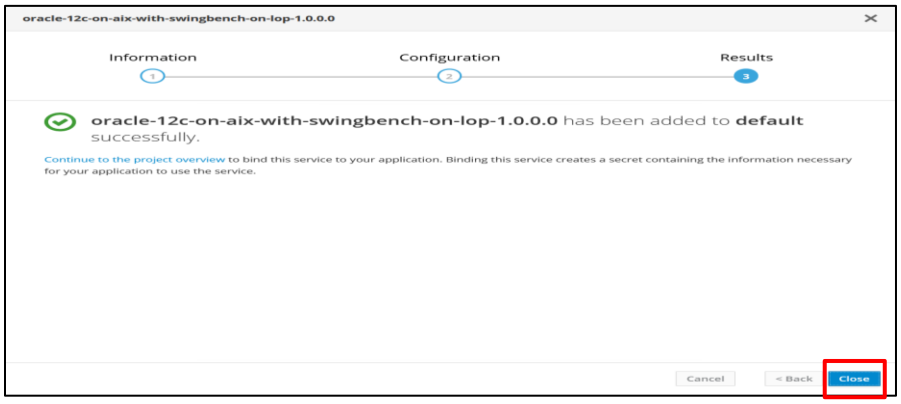
Plan standard – To deploy a Standard plan

Deploys an AIX VM running Oracle database 12c and Linux on Power VM with Oracle Instant Client and Swingbench


oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0

* Add to Project: default
IcpServiceInstanceName: testorasvc
* dbserver_user_data: RSDtestOCP
Oracle DB name

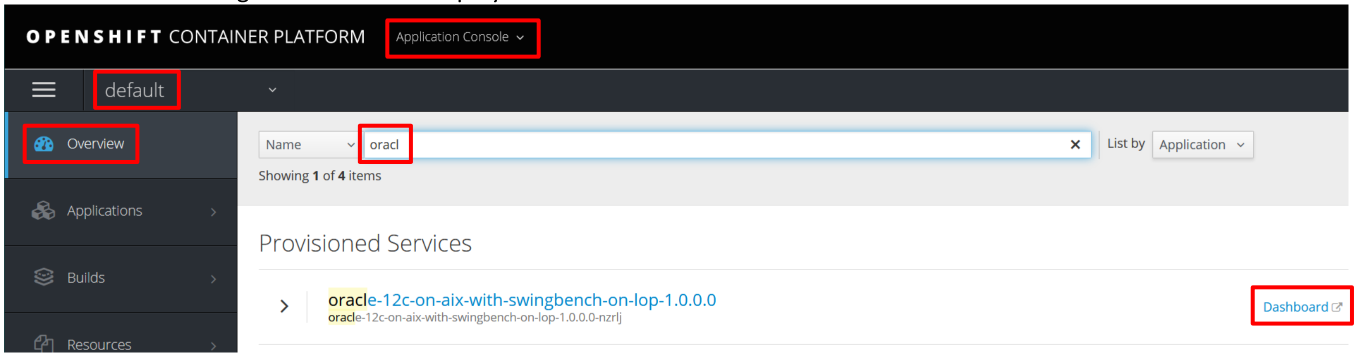
Select the project to deploy into, the service name to use and the database name and then click “Create”, followed by “Close”.


oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0

oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0 has been added to **default** successfully.

Continue to the project overview to bind this service to your application. Binding this service creates a secret containing the information necessary for your application to use the service.

After the deploy the service shows up in OCP → “Application Console”. Select the “default” project, select “Overview” and enter the search string **orac** to find the deployed service.

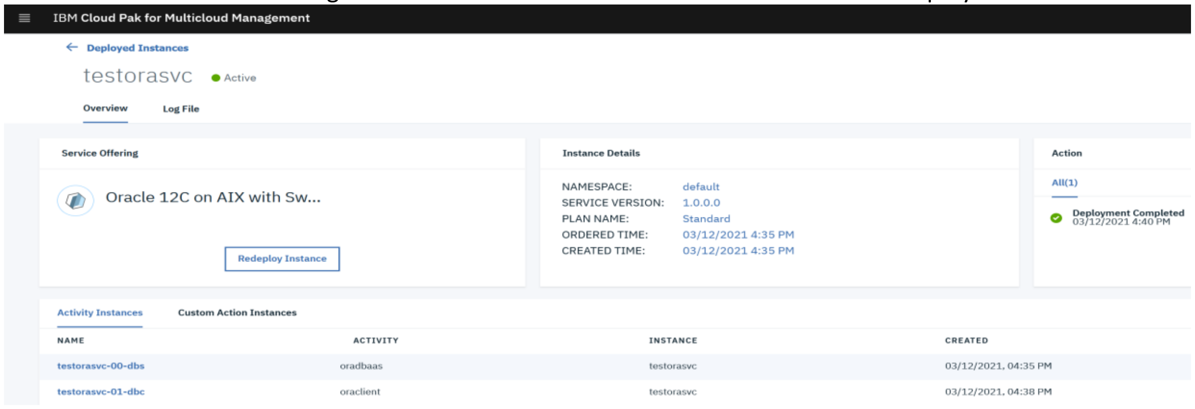

Name: oracl

Showing 1 of 4 items

Provisioned Services

oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0
oracle-12c-on-aix-with-swingbench-on-lop-1.0.0.0-nzrlj

Then click on “Dashboard” to get re-directed to T&SA and see all the details of the deployed service.


testorasvc Active

Service Offering: Oracle 12C on AIX with Sw...

Instance Details

NAMESPACE: default
SERVICE VERSION: 1.0.0.0
PLAN NAME: Standard
ORDERED TIME: 03/12/2021 4:35 PM
CREATED TIME: 03/12/2021 4:35 PM

Action

All(1)

Deployment Completed
03/12/2021 4:40 PM

Activity Instances

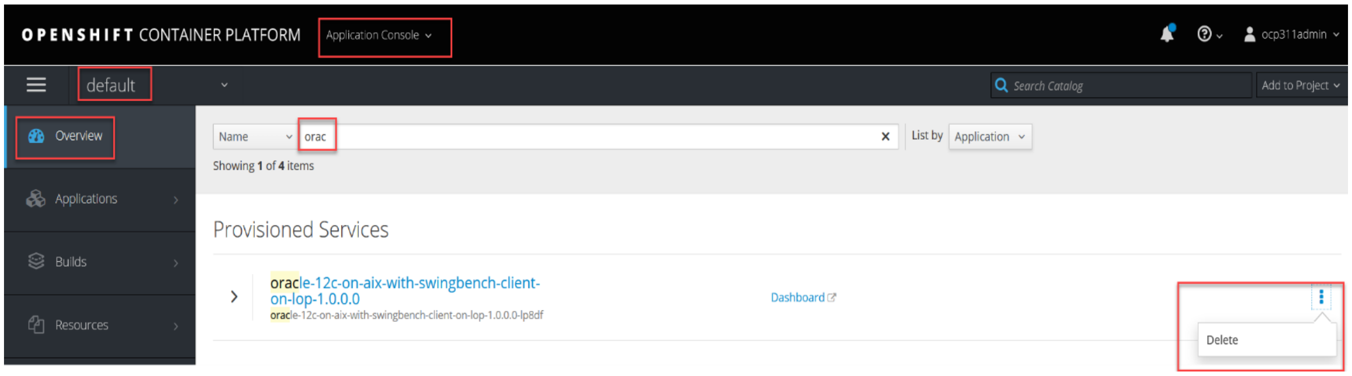
NAME	ACTIVITY	INSTANCE	CREATED
testorasvc-00-dbs	oradbaas	testorasvc	03/12/2021, 04:35 PM
testorasvc-01-dbc	oraclient	testorasvc	03/12/2021, 04:38 PM

9.7 Cleaning up

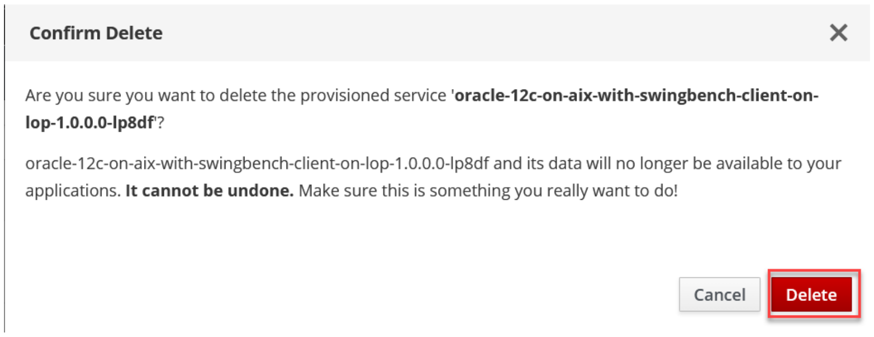
This section describes the required steps to delete deployed templates and services. In our testing we found that a specific sequence of steps is required to be able to correctly clean up the.

9.7.1 Deleting a Deployed Service Instance

If you deployed the service from the OCP Service Catalog you must start the deletion of the service from OCP as well, and not from T&SA. For example, the testorasvc deploy from OCP needs to be deleted like this – in OCP: Application Console -> "default" project -> Overview and search "orac". Select "Delete" in the three-dot menu from the drop-down



Accept the pop-up by clicking **Delete**.

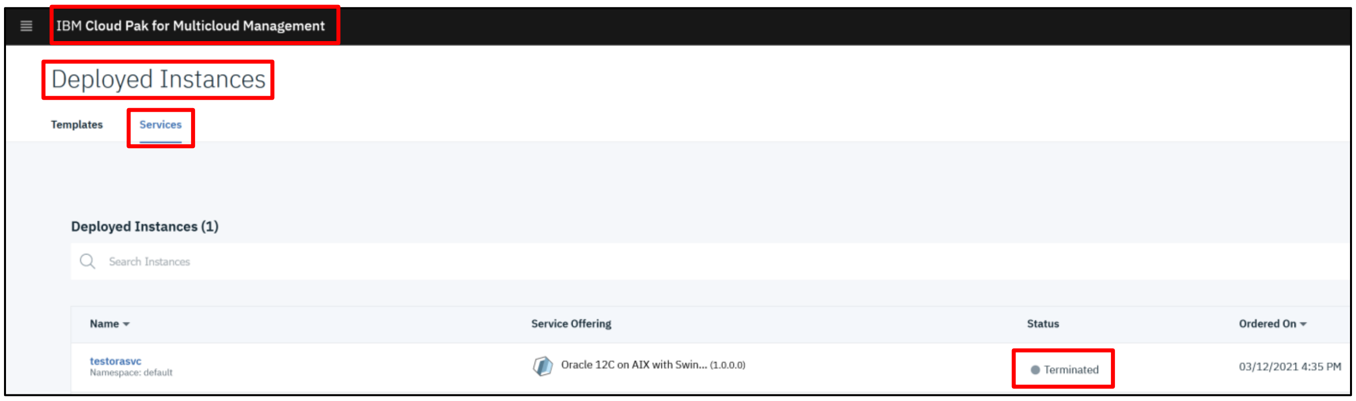


Confirm Delete

Are you sure you want to delete the provisioned service '**oracle-12c-on-aix-with-swingbench-client-on-lop-1.0.0.0-lp8df**'?

oracle-12c-on-aix-with-swingbench-client-on-lop-1.0.0.0-lp8df and its data will no longer be available to your applications. **It cannot be undone.** Make sure this is something you really want to do!

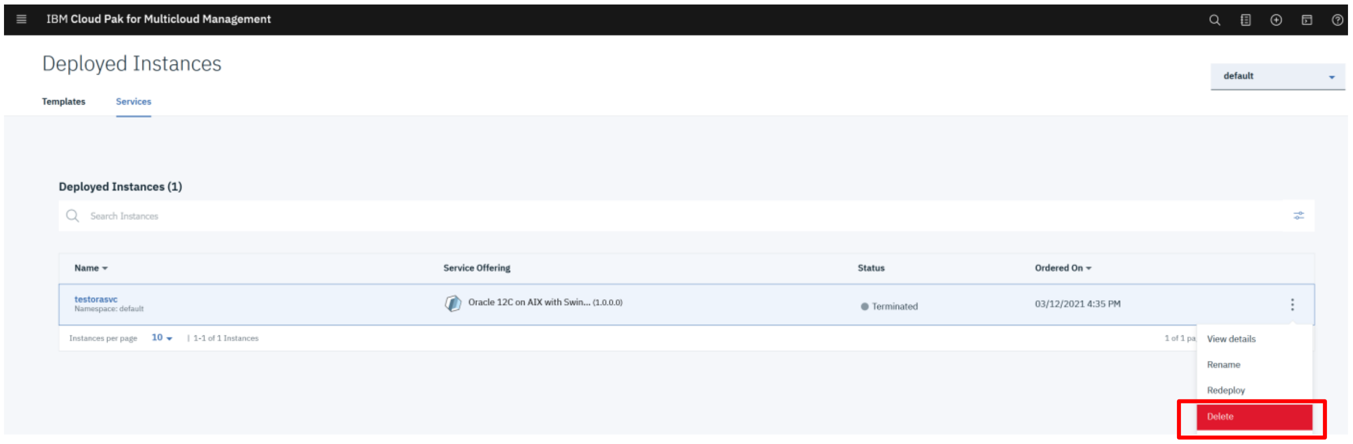
Note that this removes the service from the list of Provisioned Services and terminates the service deployment. This results in all VMs and storage being released. Note that the deployment of that service is not deleted from T&SA as the screenshot below illustrates.



Name	Service Offering	Status	Ordered On
testorasvc Namespace: default	Oracle 12C on AIX with Swin... (1.0.0.0)	● Terminated	03/12/2021 4:35 PM

Deleting a deployed service instance in T&SA requires two steps. The first step is to **Terminate** the service instance, or delete the Provisioned Services in OCP GUI, which deletes the VMs in PowerVC. After the instance is terminated, the next step is to **Delete** the deployed service instance which removes the deployed instance from T&SA. If you do the **Delete** first, the deployed instance is removed from T&SA, but the deployed VMs will persist, and you need to delete those VMs from PowerVC manually instead as T&SA has no longer any knowledge of those deployed VMs.

The final step to cleanly remove the deployed instance from T&SA is to **Delete** it as shown below.



IBM Cloud Pak for Multicloud Management

Deployed Instances

Templates | Services

default

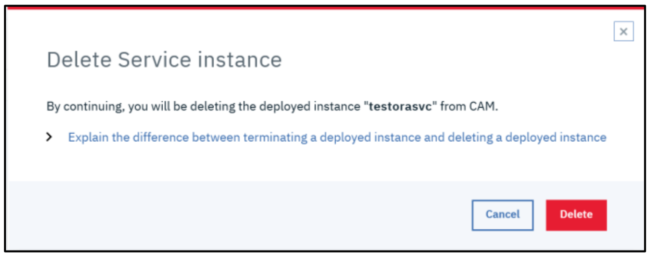
Deployed Instances (1)

Search Instances

Name	Service Offering	Status	Ordered On
testorasvc Namespace: default	Oracle 12C on AIX with Swin... (1.0.0.0)	Terminated	03/12/2021 4:35 PM

Instances per page 10 | 1-1 of 1 Instances

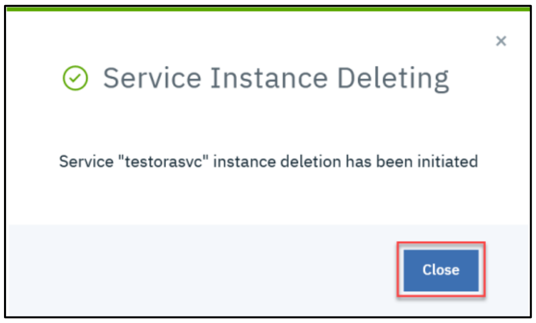
A pop-up window appears, requesting you to confirm. Click the **Delete** button.



Delete Service instance

By continuing, you will be deleting the deployed instance "**testorasvc**" from CAM.

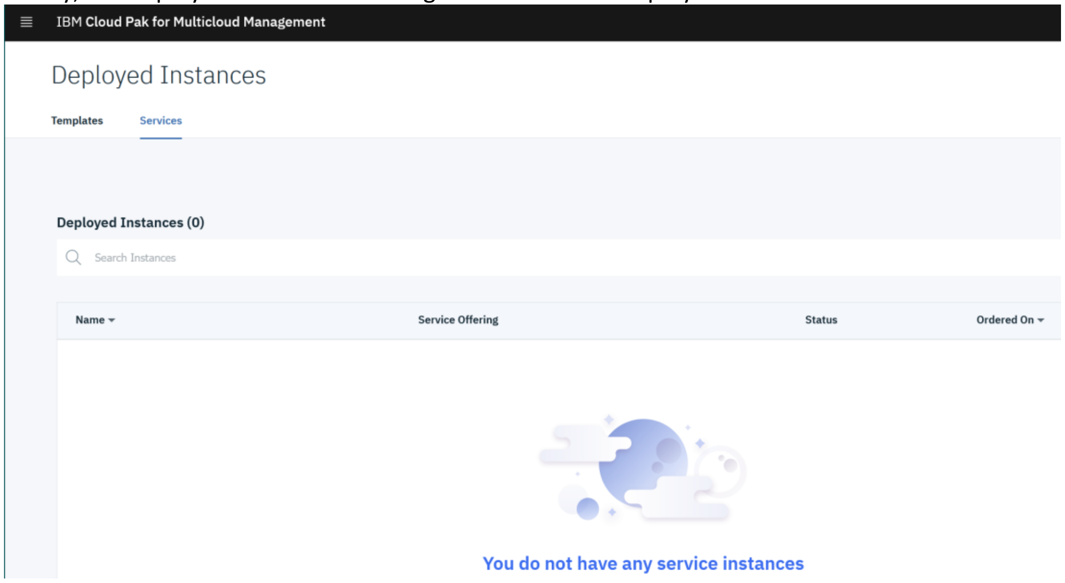
Explain the difference between terminating a deployed instance and deleting a deployed instance



Service Instance Deleting

Service "testorasvc" instance deletion has been initiated

Finally, the deployed instance is no longer shown on the Deployed Instance screen.



IBM Cloud Pak for Multicloud Management

Deployed Instances

Templates | Services

Deployed Instances (0)

Search Instances

Name	Service Offering	Status	Ordered On

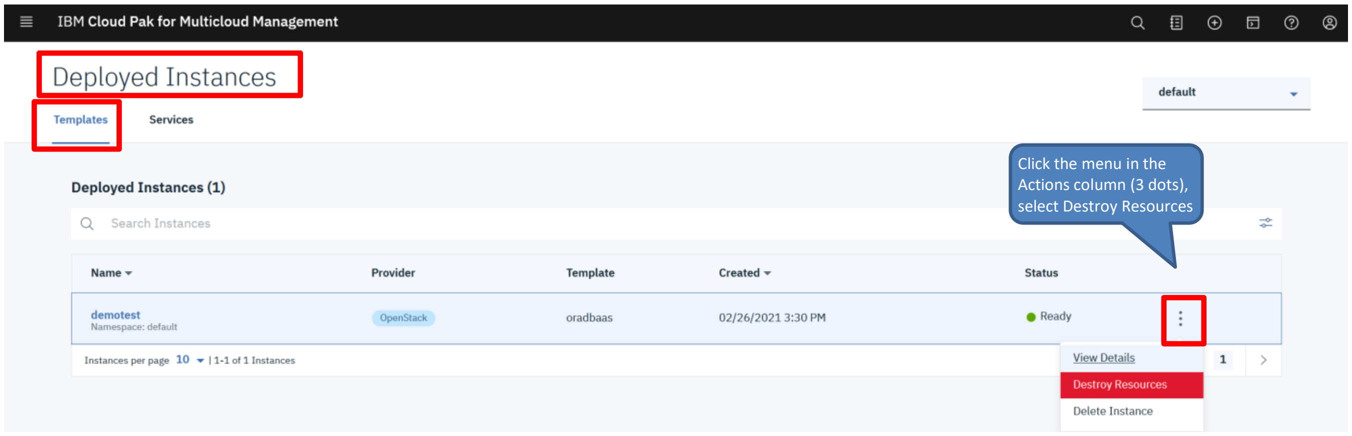
You do not have any service instances

9.7.2 Deleting a Deployed Template Instance

Deleting the deployed template instance requires two steps as well: Destroying the Resources and Deleting the Instance.

9.7.2.1 Destroy Resources

Go back to the T&SA screen, click **Deployed Instances**, or use the upper left hamburger menu, select Deployed Instances -> Templates. The **Destroy Resources** is similar to the **Terminate** for a Service in that it deletes the underlying VM(s) for the template in PowerVC but keeps the information about that template deployment intact in T&SA.

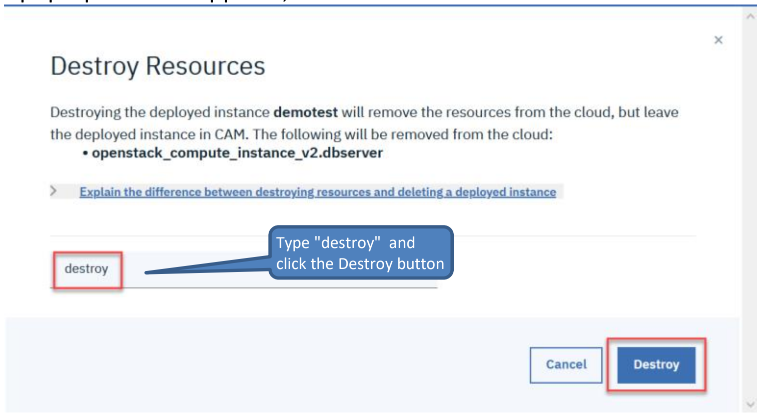

IBM Cloud Pak for Multicloud Management
Deployed Instances
Templates
Services
default
Click the menu in the Actions column (3 dots), select Destroy Resources
View Details
Destroy Resources
Delete Instance

Deployed Instances (1)

Name	Provider	Template	Created	Status
demotest Namespace: default	OpenStack	oradbaas	02/26/2021 3:30 PM	Ready

Instances per page 10 | 1-1 of 1 Instances

A pop-up window appears,

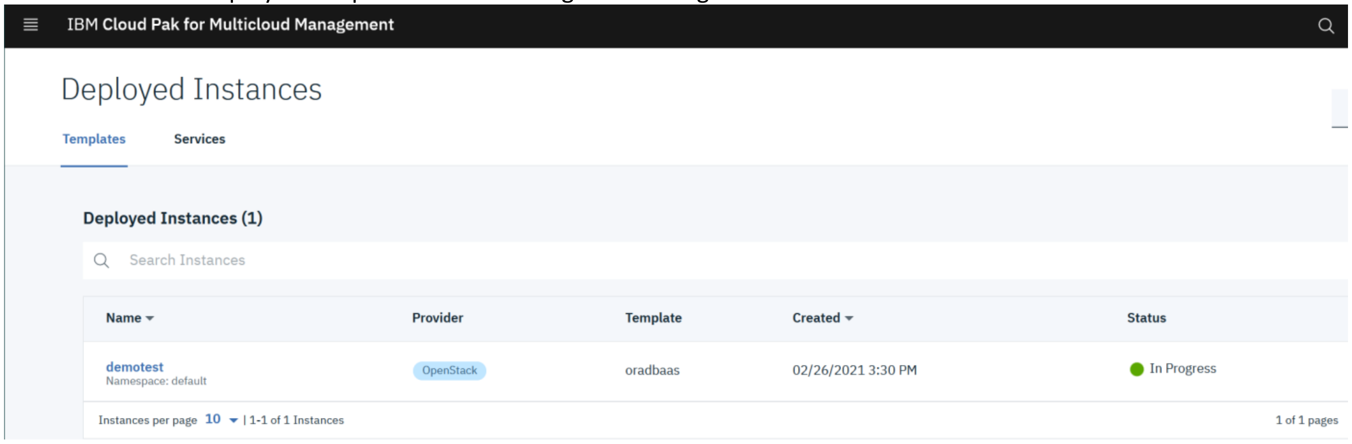


Destroy Resources

Destroying the deployed instance **demotest** will remove the resources from the cloud, but leave the deployed instance in CAM. The following will be removed from the cloud:
- **openstack_compute_instance_v2.dbserver**

Explain the difference between destroying resources and deleting a deployed instance

The status for the deployed template instance changes to In Progress.

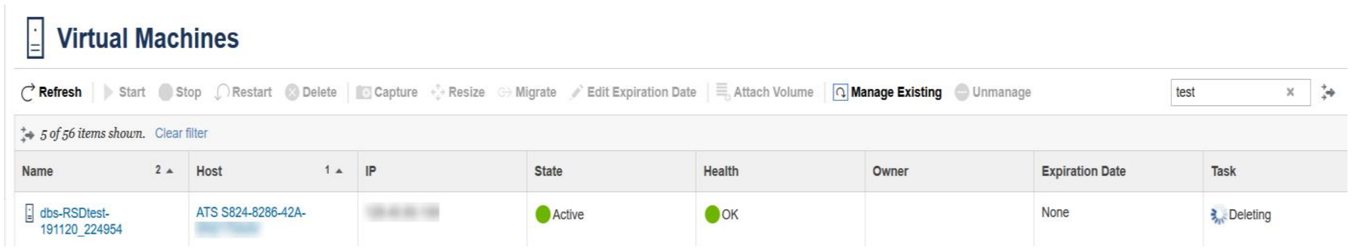


Deployed Instances (1)

Name	Provider	Template	Created	Status
demotest Namespace: default	OpenStack	oradbaas	02/26/2021 3:30 PM	In Progress

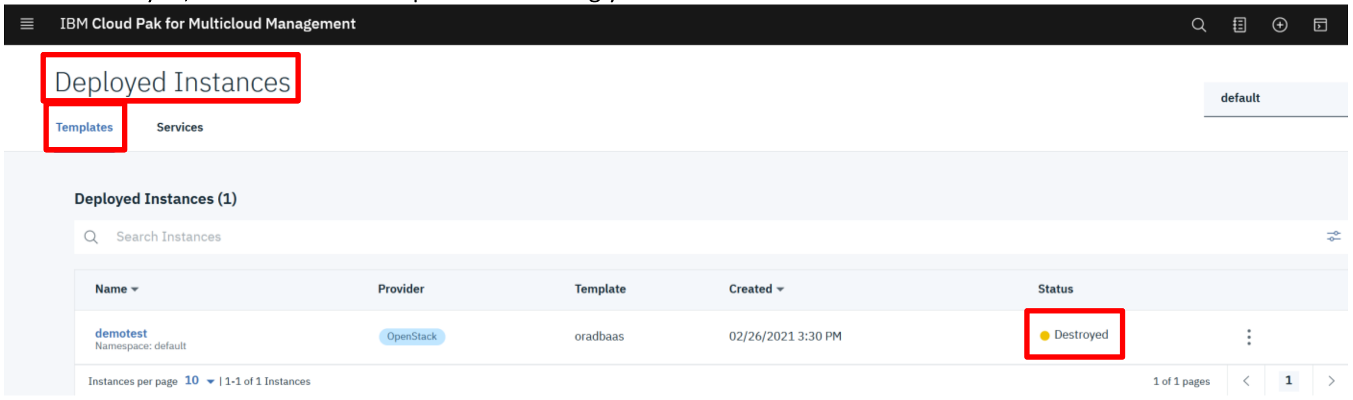
Instances per page 10 | 1-1 of 1 Instances   1 of 1 pages

Looking at PowerVC, we see the VM being deleted.



Name	Host	IP	State	Health	Owner	Expiration Date	Task
dbs-RSDtest-191120_224954	ATS S824-8286-42A-[redacted]	[redacted]	Active	OK		None	Deleting

When destroyed, the T&SA status is updated accordingly.



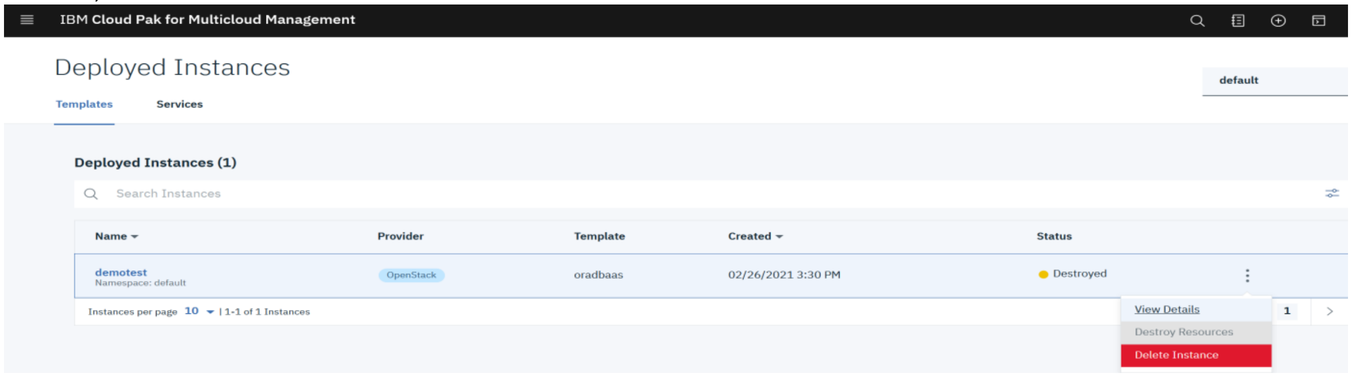
Name	Provider	Template	Created	Status
demotest Namespace: default	OpenStack	oradbaas	02/26/2021 3:30 PM	Destroyed

The deployed instance may now be deleted.

Tip: If you have a template with many parameters and you just want to re-deploy the template with a minor parameter change, you can Destroy the deployed template, then go to the "deployed template", click on the details and edit the deployment, re-plan it – means changing any parameters needed – and then activate the plan. This will then trigger a new deployment with the same deployment name and the changed parameters, without having to go through the full template deployment process.

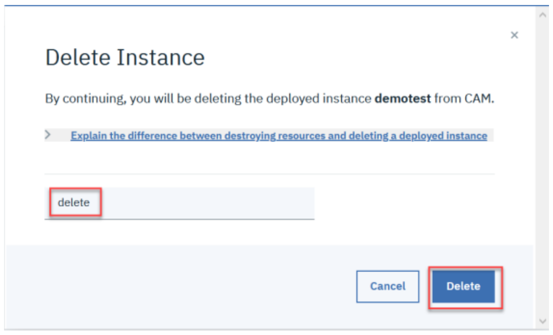
9.7.2.2 Delete Deployed Instance

Once the resources have been destroyed, the deployed instance can be deleted. Select the demotest deployed template instance, click on the three-dot menu and select **Delete Instance**.



Name	Provider	Template	Created	Status
demotest Namespace: default	OpenStack	oradbaas	02/26/2021 3:30 PM	Destroyed

Again, a pop-up window appears. Type "delete" and click the **Delete** button.



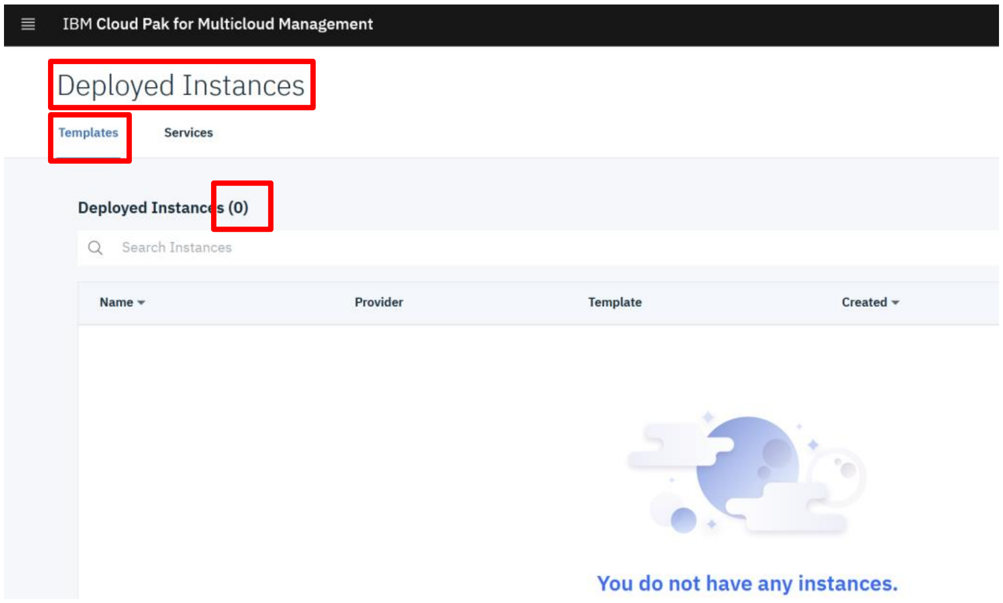
Delete Instance

By continuing, you will be deleting the deployed instance **demotest** from CAM.

Explain the difference between destroying resources and deleting a deployed instance

delete

The instance has been deleted.



IBM Cloud Pak for Multicloud Management

Deployed Instances

Deployed Instances (0)

Name	Provider	Template	Created

You do not have any instances.

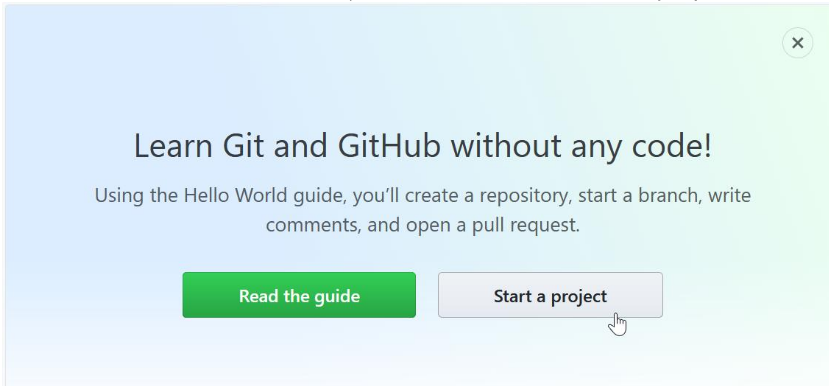
10 Appendix

10.1 Working with GitHub

10.1.1 Create GitHub repository

Access https://github.com. Sign up for an account on github.com or use your existing account and log in to GitHub.

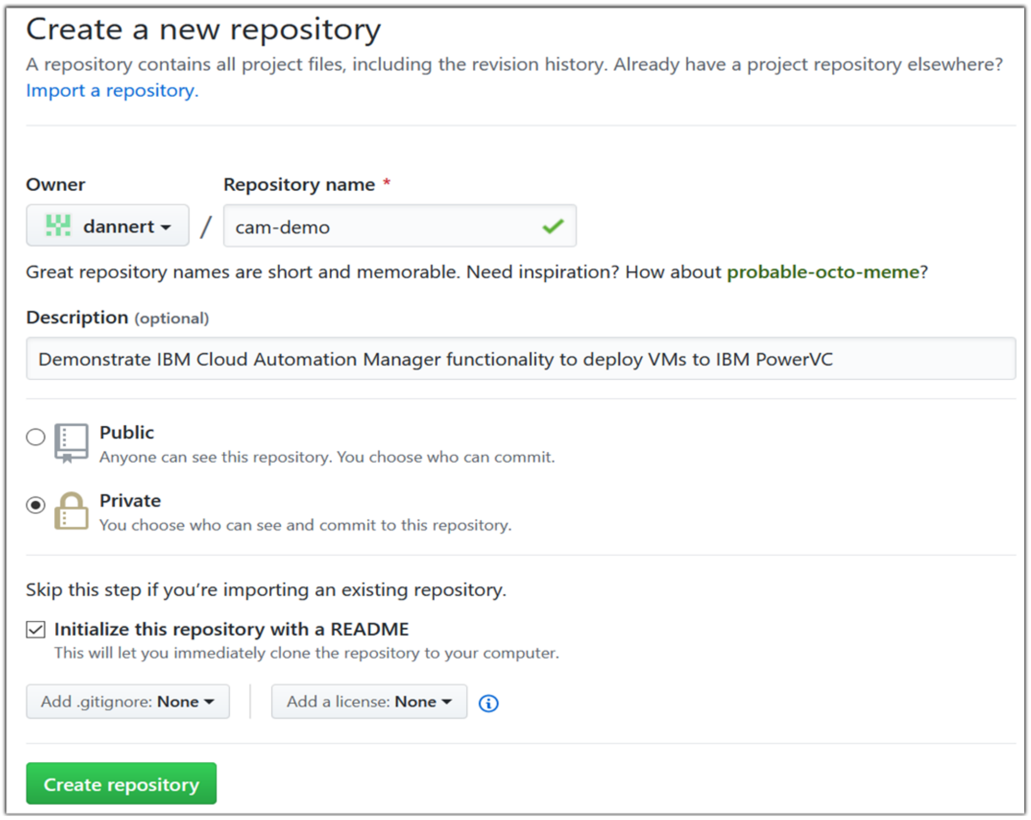
Click the **New** button next to Repositories or click on **Start a project** to create a new repository.



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Type in the name of your repository. In this example, we use the name **Cam-demo**. Select **Private** and **Initialize this repository with a README** and then click **Create Repository**.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: dannert / Repository name *: cam-demo

Great repository names are short and memorable. Need inspiration? How about probable-octo-meme?

Description (optional): Demonstrate IBM Cloud Automation Manager functionality to deploy VMs to IBM PowerVC

○ Public — Anyone can see this repository. You choose who can commit.

● Private — You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☑ Initialize this repository with a README — This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None

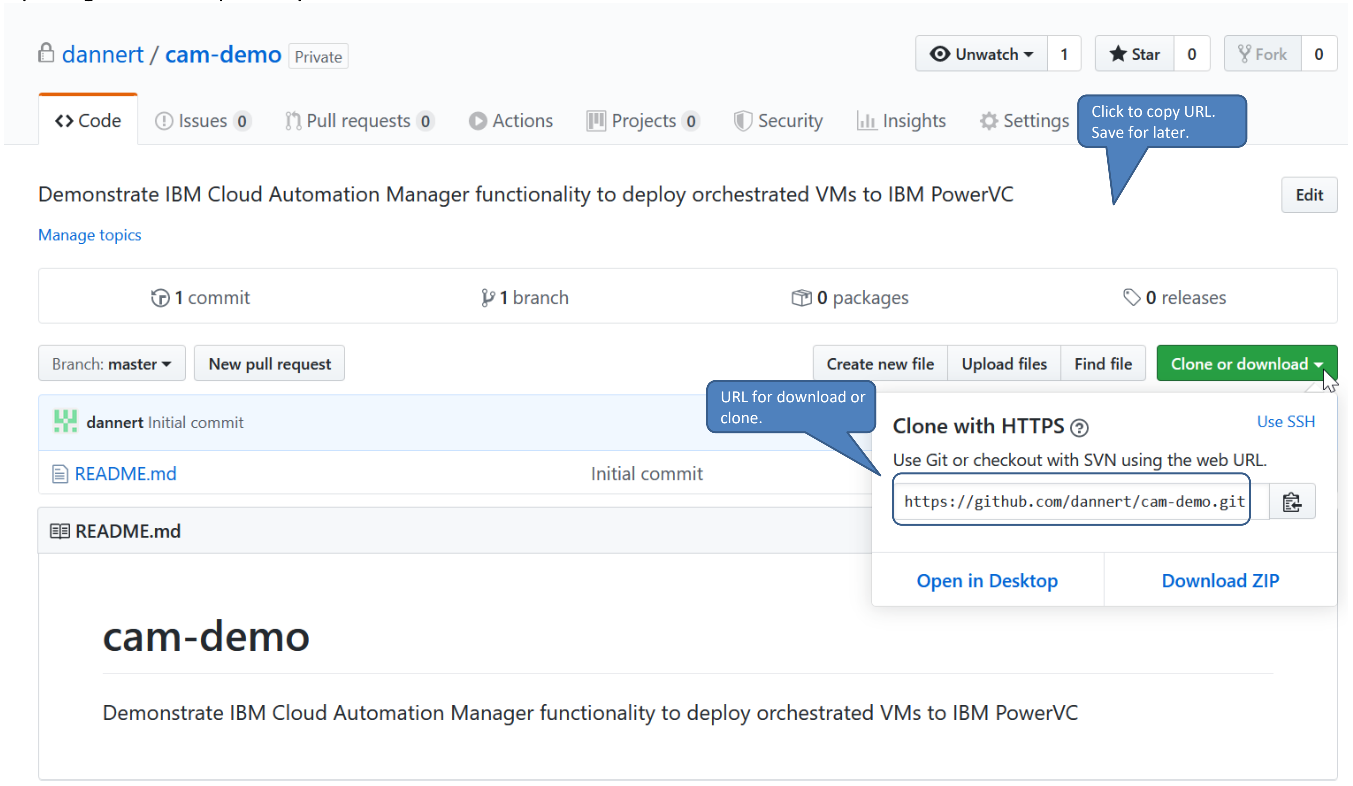
Information on Private repository may be found on:
https://help.github.com/en/articles/githubs-products

Review
https://github.com/pricing

for features comparison of the various GitHub offerings.

You can clone the code from the following public GitHub repoitory:
https://github.com/ppc64le/devops-automation/tree/master/terraform/oracle/cam-ora

Opening the new repoistory then looks like this:


dannert / cam-demo Private
Unwatch 1 Star 0 Fork 0
Code Issues 0 Pull requests 0 Actions Projects 0 Security Insights Settings
Click to copy URL. Save for later.
Edit
Manage topics
1 commit 1 branch 0 packages 0 releases
Branch: master New pull request Create new file Upload files Find file Clone or download
URL for download or clone.
Clone with HTTPS Use SSH
Open in Desktop Download ZIP
README.md
cam-demo

Demonstrate IBM Cloud Automation Manager functionality to deploy orchestrated VMs to IBM PowerVC

dannert Initial commit

README.md Initial commit

Use Git or checkout with SVN using the web URL.

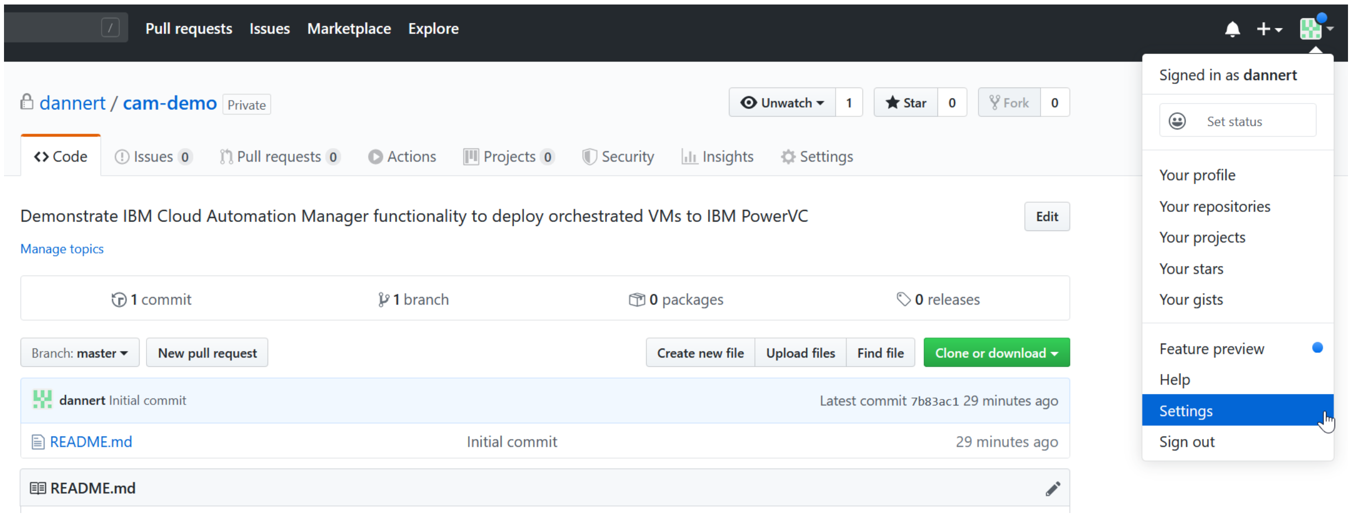
```
https://github.com/dannert/cam-demo.git
```

Demonstrate IBM Cloud Automation Manager functionality to deploy orchestrated VMs to IBM PowerVC

Copy the HTTPS URL shown as you will need that later in T&SA to register this GitHub repository for use in the project.

10.1.2 Create personal access token

Click the icon (a profile picture that can be changed on the **Settings** screen) on the upper right-hand corner to reveal and access the drop-down submenu. See picture below. Select **Settings**.

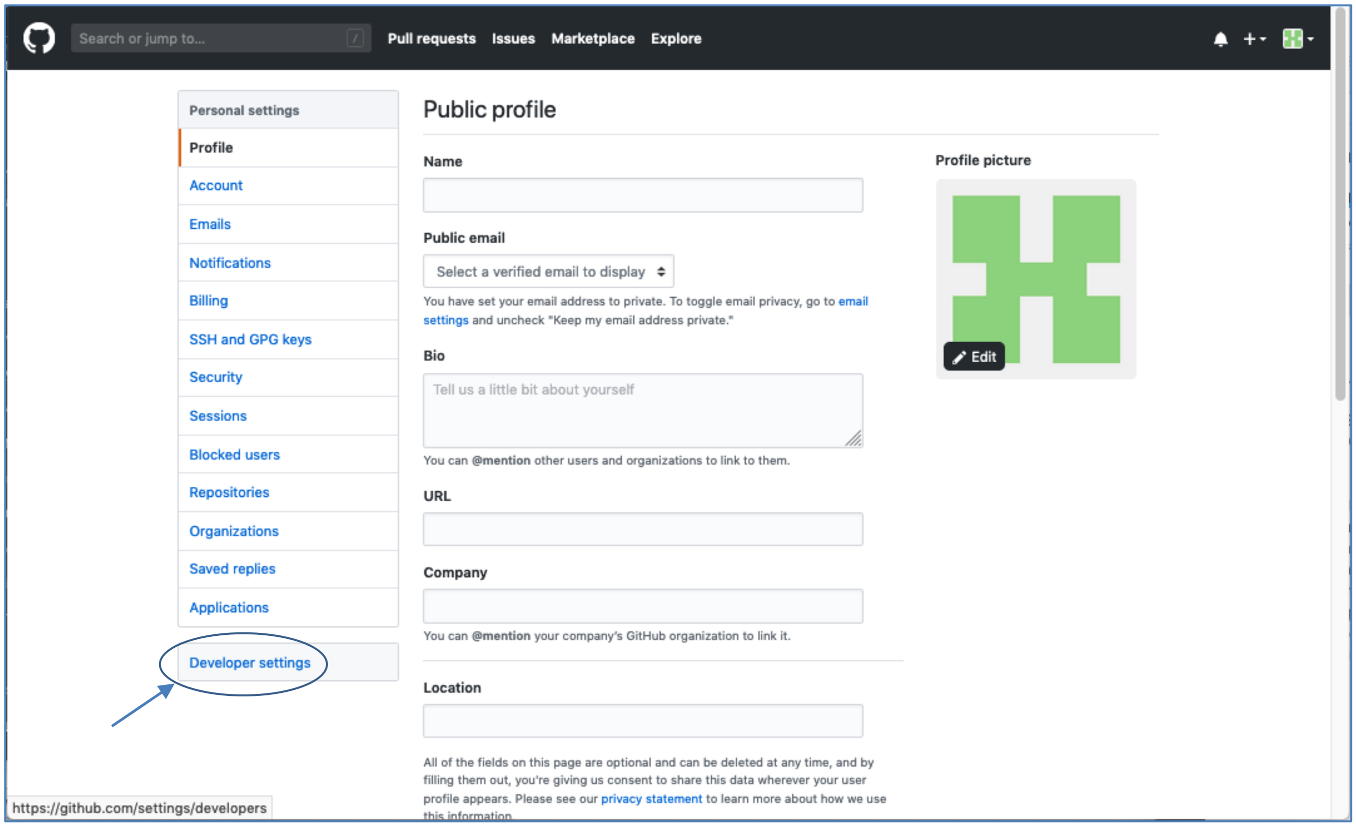

Pull requests Issues Marketplace Explore
dannert / cam-demo Private
Unwatch 1 Star 0 Fork 0
Code Issues 0 Pull requests 0 Actions Projects 0 Security Insights Settings
Edit
Manage topics
1 commit 1 branch 0 packages 0 releases
Branch: master New pull request Create new file Upload files Find file Clone or download
README.md
Signed in as dannert
Set status
Your profile
Your repositories
Your projects
Your stars
Your gists
Feature preview
Help
Settings
Sign out

Demonstrate IBM Cloud Automation Manager functionality to deploy orchestrated VMs to IBM PowerVC

dannert Initial commit Latest commit 7b83ac1 29 minutes ago

README.md Initial commit 29 minutes ago

Select **Developer settings** on the left.



Public profile

Name

Public email

Select a verified email to display

You have set your email address to private. To toggle email privacy, go to email settings and uncheck "Keep my email address private."

Bio

Tell us a little bit about yourself

You can @mention other users and organizations to link to them.

URL

Company

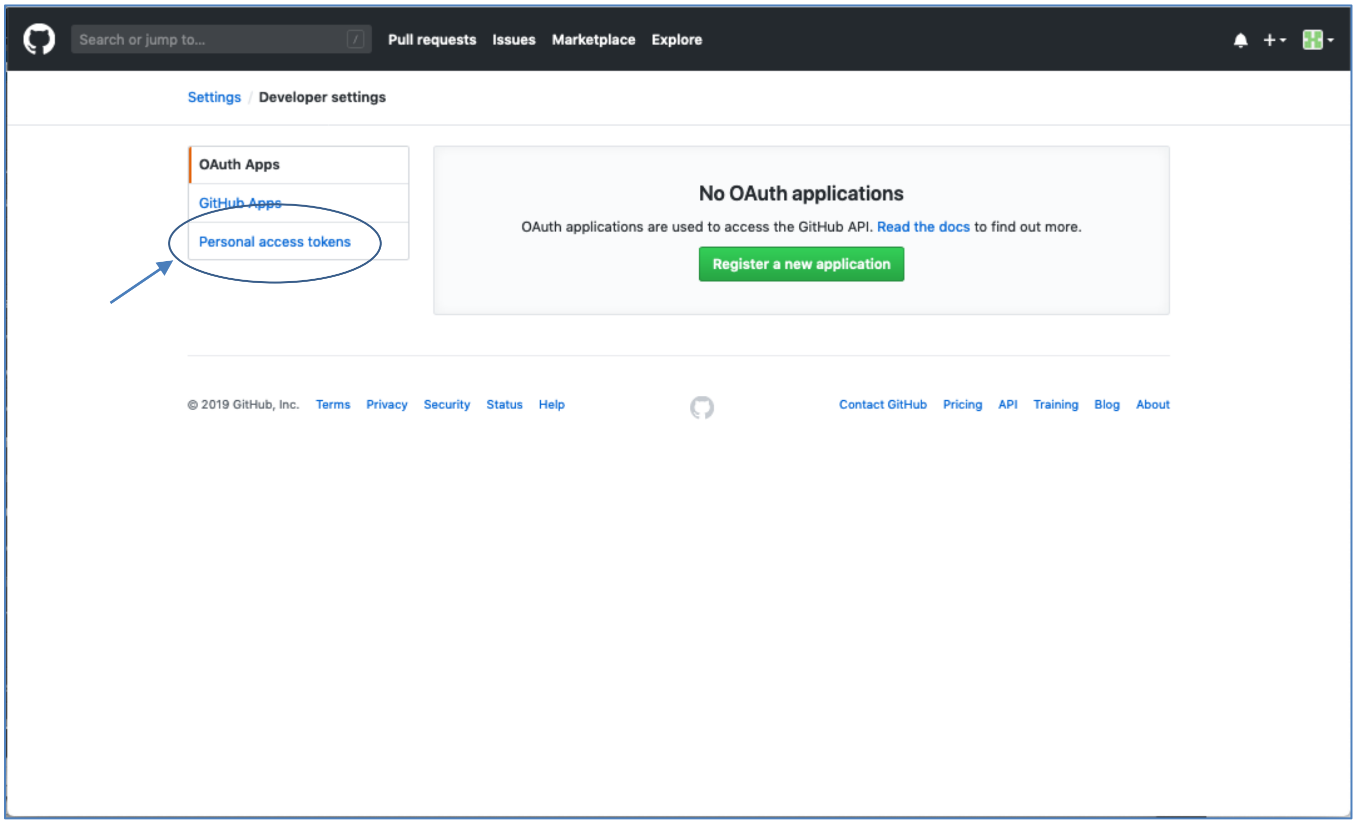
You can @mention your company's GitHub organization to link it.

Location

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our privacy statement to learn more about how we use this information.

Profile picture

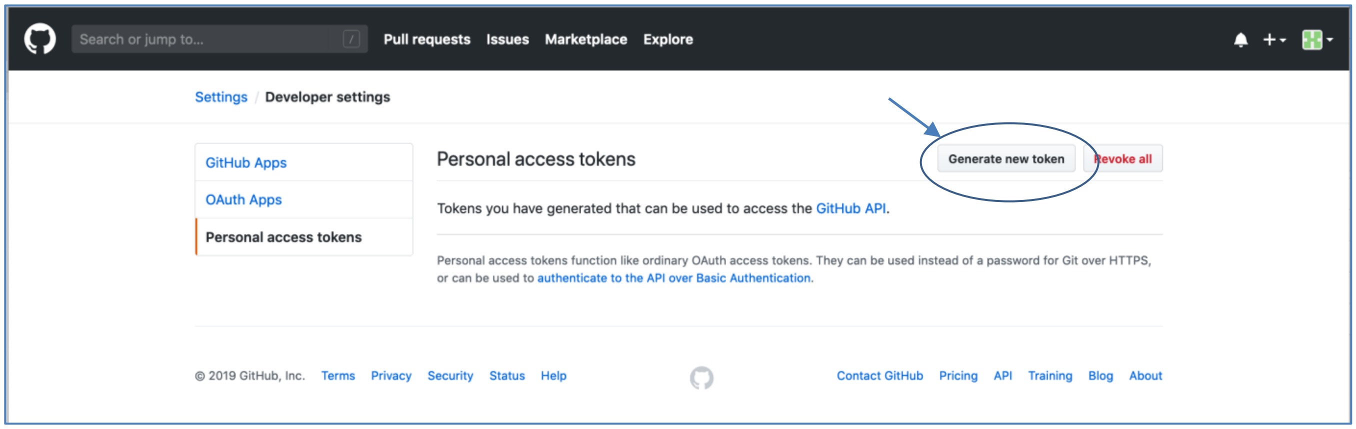
Select **Personal access tokens**.



No OAuth applications

OAuth applications are used to access the GitHub API. Read the docs to find out more.

Now click **Generate new token**.



Search or jump to... Pull requests Issues Marketplace Explore

Settings / Developer settings

GitHub Apps
OAuth Apps
Personal access tokens

Personal access tokens

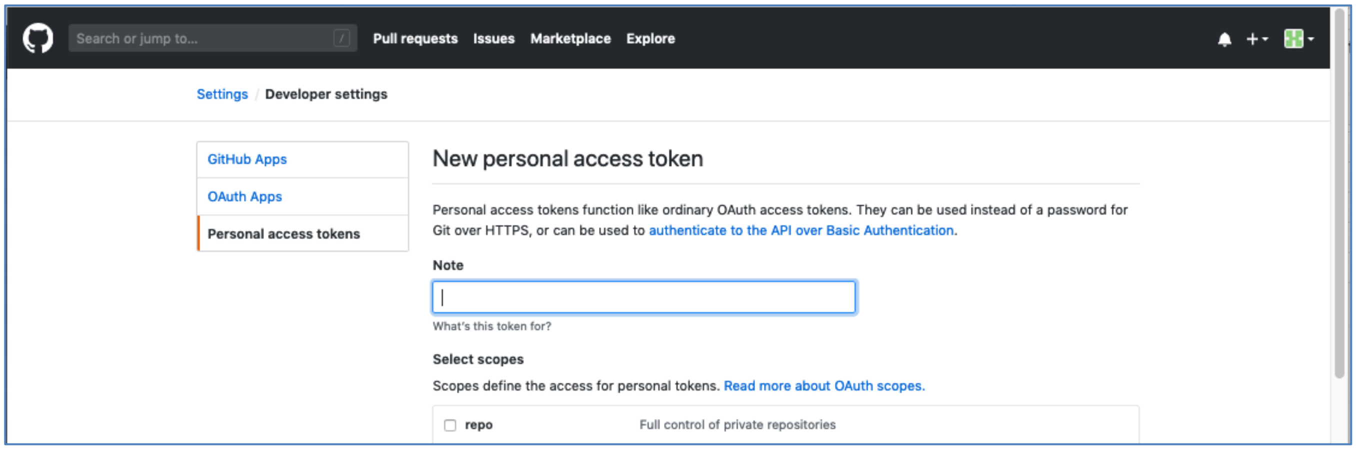
Generate new token Revoke all

Tokens you have generated that can be used to access the GitHub API.

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

© 2019 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Confirm the user password if requested to get to the next screen as shown below.
Type in an appropriate description for this token and scroll down.



Search or jump to... Pull requests Issues Marketplace Explore

Settings / Developer settings

GitHub Apps
OAuth Apps
Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note

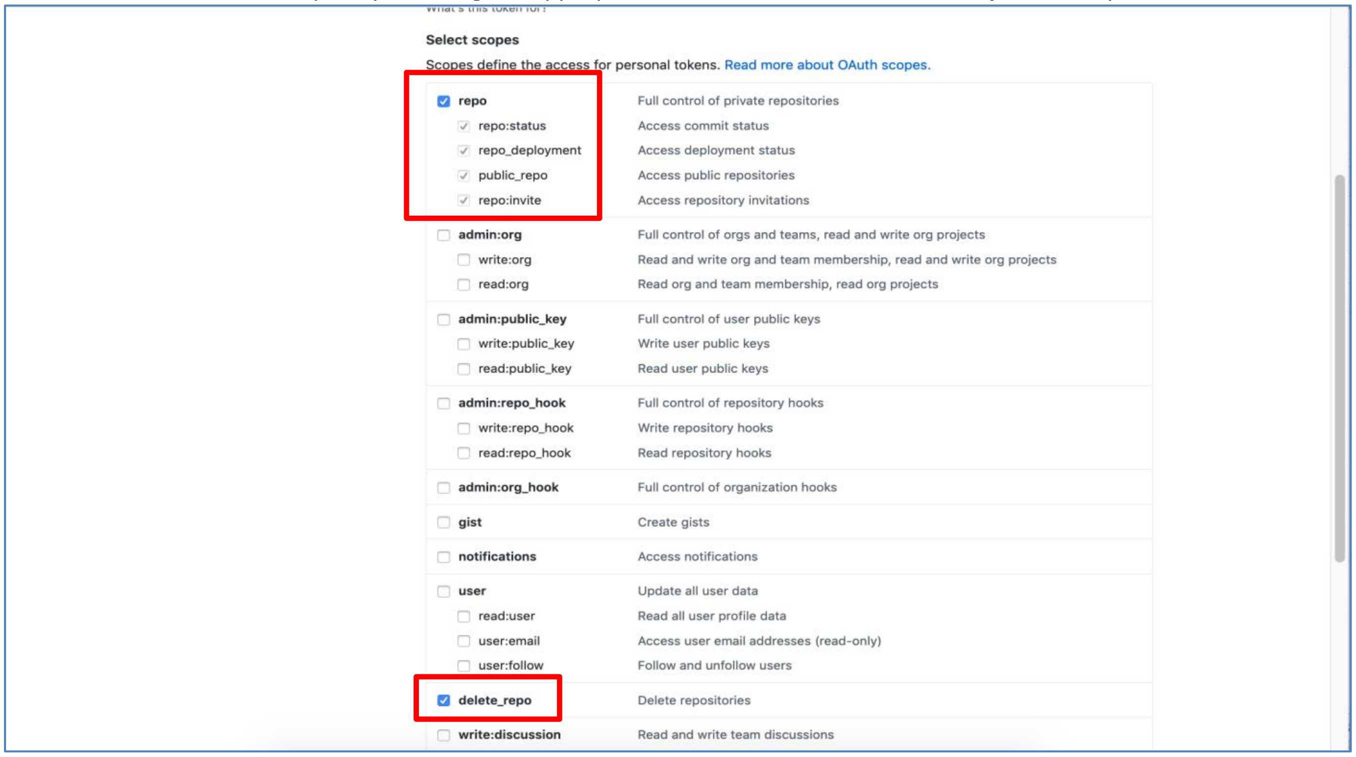
What's this token for?

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

repo Full control of private repositories

Select the desired access scopes by checking the appropriate boxes, but at minimum the **repo** related permissions.

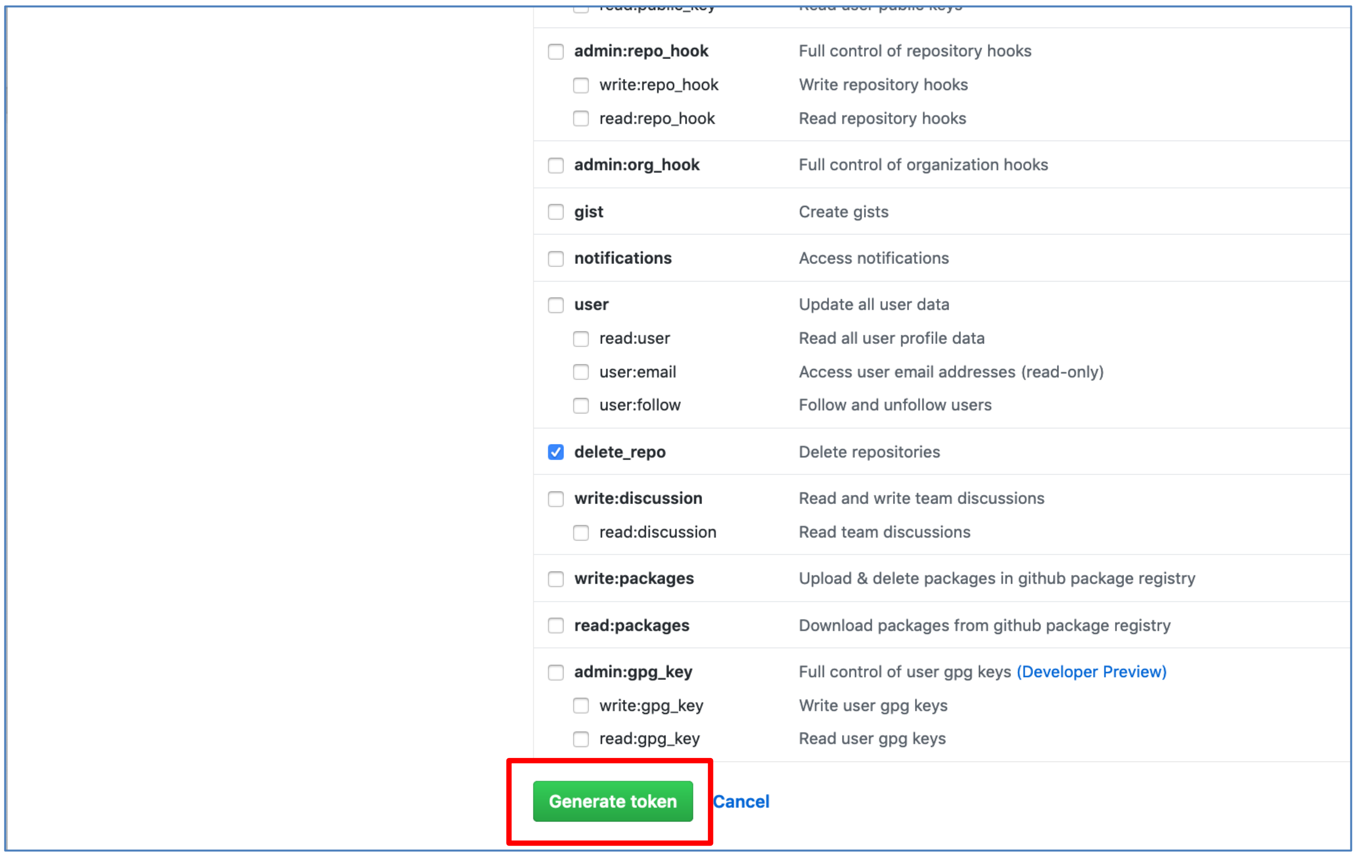


Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

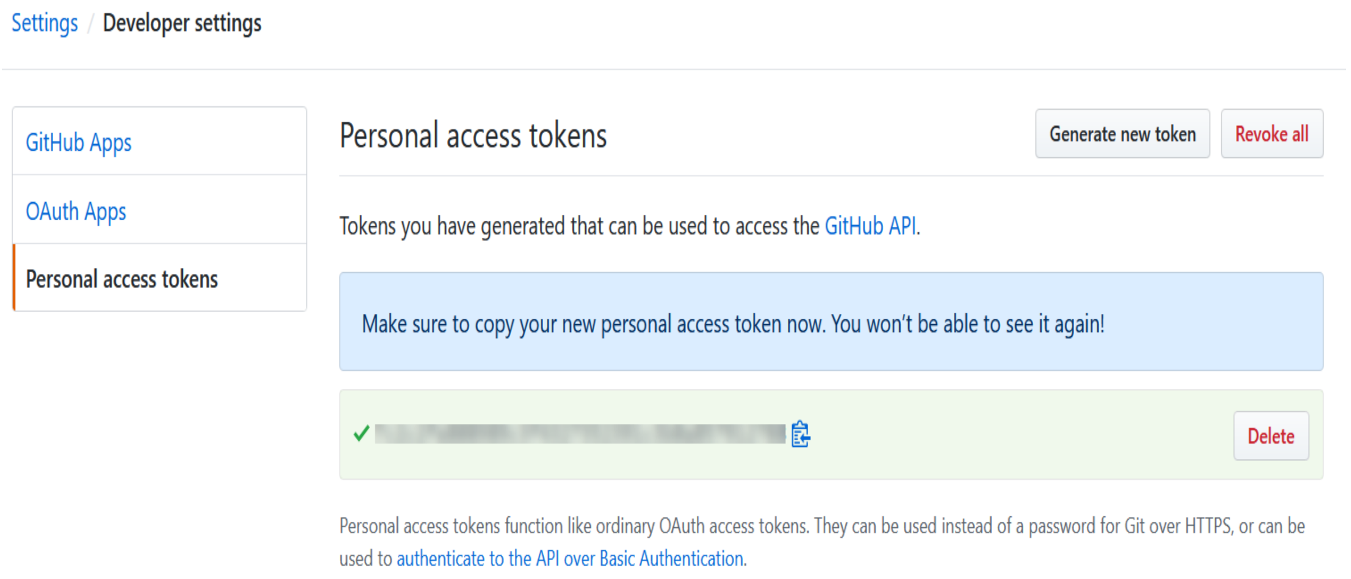
✓ repo	Full control of private repositories
✓ repo:status	Access commit status
✓ repo_deployment	Access deployment status
✓ public_repo	Access public repositories
✓ repo:invite	Access repository invitations
admin:org	Full control of orgs and teams, read and write org projects
write:org	Read and write org and team membership, read and write org projects
read:org	Read org and team membership, read org projects
admin:public_key	Full control of user public keys
write:public_key	Write user public keys
read:public_key	Read user public keys
admin:repo_hook	Full control of repository hooks
write:repo_hook	Write repository hooks
read:repo_hook	Read repository hooks
admin:org_hook	Full control of organization hooks
gist	Create gists
notifications	Access notifications
user	Update all user data
read:user	Read all user profile data
user:email	Access user email addresses (read-only)
user:follow	Follow and unfollow users
✓ delete_repo	Delete repositories
write:discussion	Read and write team discussions

Scroll down and click **Generate token.**



☐ admin:repo_hook	Full control of repository hooks
☐ write:repo_hook	Write repository hooks
☐ read:repo_hook	Read repository hooks
☐ admin:org_hook	Full control of organization hooks
☐ gist	Create gists
☐ notifications	Access notifications
☐ user	Update all user data
☐ read:user	Read all user profile data
☐ user:email	Access user email addresses (read-only)
☐ user:follow	Follow and unfollow users
☑ delete_repo	Delete repositories
☐ write:discussion	Read and write team discussions
☐ read:discussion	Read team discussions
☐ write:packages	Upload & delete packages in github package registry
☐ read:packages	Download packages from github package registry
☐ admin:gpg_key	Full control of user gpg keys (Developer Preview)
☐ write:gpg_key	Write user gpg keys
☐ read:gpg_key	Read user gpg keys

IMPORTANT: Save the personal access token for use later as T&SA will request that token for any interaction with the GitHub repository.



Personal access tokens

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your new personal access token now. You won't be able to see it again!

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

10.2 *oradbaas* Template files

10.2.1 main.tf

```
######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#----------------------------------------------------------------------
#####################################################################
##
## Deploy Oracle database as a service
##
#####################################################################
## REFERENCE {"openstack_network":{"type": "openstack_reference_network"}}

terraform {
  required_version = "> 0.8.0"
}

provider "openstack" {
  insecure    = true
  version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbserver" {
  name      = "${format("${var.dbserver_name}-${var.dbserver_user_data}-
${local.local_dbserver_ts}")}"
  image_name  = "${var.openstack_image_name}"
  flavor_name = "${var.openstack_flavor_name}"
  user_data = "${format("DBNAME=%s",var.dbserver_user_data)}"
  network {
    uuid = "${var.openstack_network_id}"
  }
}
```

10.2.2 variables.tf

```
######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
```

```
#
#-----------------------------------------------------------------------
######################################################################
##
## Deploy Oracle database as a service
##
######################################################################

variable "dbserver_name" {
  type = "string"
  description = " db server VM name prefix. Oracle dbname and a timestamp will be appended in
the PowerVC VM name."
}

variable "openstack_image_name" {
  type = "string"
  description = "PowerVC image name"
}

variable "openstack_flavor_name" {
  type = "string"
  description = "flavor name, or PowerVC compute template)"
}

variable "openstack_network_id" {
  type = "string"
  description = "Network id in PowerVC"
}

variable "dbserver_user_data" {
  type = "string"
  description = "Oracle DB name"
}

locals {
  local_dbserver_ts = "${replace(replace(replace(substr(timestamp(),2,17),"-
",""),"T","_"),":","")}"
}
```

10.2.3 outputs.tf

```
#######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#-----------------------------------------------------------------------
######################################################################
##
## Deploy Oracle database as a service
##
######################################################################

output "dbserver_ipaddr" {
  value = "${format("%s",join(",",
openstack_compute_instance_v2.dbserver.*.network.0.fixed_ip_v4))}"
}

output "dbserver_vm_name" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbserver.*.name))}"
  description = "VM name in PowerVC"
}

output "dbserver_ts" {
  value = "${format("${local.local_dbserver_ts}")}"
}

output "dbname" {
  value = "${var.dbserver_user_data}"
}
```

10.2.4 camtemplate.json

```
{
   "name": "oradbaas",
   "description": "Description for oradbaas.",
   "type": "userCreated",
   "manifest": {
     "template_type": "Terraform",
     "template_format": "HCL",
     "template_provider": "OpenStack"
   },
   "metadata": {
     "bullets": []
   }
}
```

10.2.5 camvariables.json

```
{
  "input_datatypes": [ ],
  "input_namespaces": [ ],
  "output_namespace": "",
  "input_groups": [
    {
      "name": "Input-parameters",
      "label": "Input parameters"
    }
  ],
  "output_groups": [
     {
      "name": "Output-parameters",
      "label": "Output parameters"
    }
  ],
  "template_input_params": [
    {
      "name": "dbserver_name",
      "label": "db server VM name prefix",
      "description": " db server VM name prefix. Oracle dbname and a timestamp will be
appended in the PowerVC VM name.",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_image_name",
      "label": "Openstack Image Name",
      "description": "PowerVC image name",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_flavor_name",
      "label": "Openstack Flavor Name",
      "description": "flavor name, or PowerVC compute template)",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_network_id",
      "label": "Openstack Network Id",
      "description": "Network id in PowerVC",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "dbserver_user_data",
      "label": "Dbserver User Data",
      "description": "Oracle DB name",
      "type": "string",
      "default": "RSDtest",
      "validation": "",
      "group_name": "Input-parameters",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    }
  ],
  "template_output_params": [
    {
      "name": "dbserver_ipaddr",
      "label": "dbserver IP address",
      "description": null,
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string"
    },
    {
      "name": "dbserver_vm_name",
      "label": "dbserver VM name",
      "description": "VM name in PowerVC",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string"
    },
    {
      "name": "dbserver_ts",
      "label": "dbserver ts",
      "description": "internal only - timestamp for output to dbclient",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": true,
      "shortname": "",
      "type": "string"
    },
    {
      "name": "dbname",
      "label": "dbname",
      "description": "Database name",
      "group_name": "Output-parameters",
      "secured": false,
      "hidden": true,
      "shortname": "",
      "type": "string"
    }
  ]
}
```

10.3 *oraclient* Template files

10.3.1 main.tf

Note that you need to replace the text <MY_NFS_SERVER_IP> to reflect the IP of your NFS server.

```
######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#-----------------------------------------------------------------------
#####################################################################
##
##      Deploy LOP with Oracle client and Swingbench for oraclient
##
#####################################################################
#
# Note that the text <MY_NFS_SERVER_IP> needs to be replaced with the
# hostname or IP of your NFS server in the line:
# - mount <MY_NFS_SERVER_IP>:/export/stage /mnt
#####################################################################

terraform {
  required_version = "> 0.8.0"
}

provider "openstack" {
  insecure    = true
  version = "~> 1.2"
}

resource "openstack_compute_instance_v2" "dbclient" {
  name  = "${format("${var.dbclient_name}-${var.dbserver_user_data}-${var.dbserver_ts}")}"
  image_name  = "${var.openstack_image_name_dbclient}"
  flavor_name = "${var.openstack_flavor_name_dbclient}"
  user_data = <<EOT
#cloud-config
write_files:
  - content: |
     # dbserver parameters
     ${format("export DBNAME=%s",var.dbserver_user_data)}
     ${format("export DBSERVERIPADDR=%s",var.dbserver_ipaddr)}
     export ORASYSPW=oracle
     export SWINGBENCHBINDIR=/opt/swingbench/bin
    path: /root/dbserver.cfg
  - content: |
     cd ~
     . ./.bash_profile
     . ./dbserver.cfg
     DBACTIVE=false
     MAXLOOP=30
     LOOPDELAY=10
     i=1
     while [ "$DBACTIVE" = false ] && [ "$i" -le $MAXLOOP ] ; do
       sqlplus system/$ORASYSPW@$DBSERVERIPADDR/$DBNAME < /dev/null | grep "Connected to"
       if [ "$?" -eq 0 ]; then
           echo "Oracle DB server active"
           TEMP=`sqlplus -s system/$ORASYSPW@$DBSERVERIPADDR/$DBNAME << EOF1
           set pages 0
           set head off
           set feed off
           select open_mode, database_status, shutdown_pending from v\\$database,
v\\$instance;
           exit
     EOF1`
       TEMP2=`echo $TEMP | tr -d ' '`
       if [ "$TEMP2" = "READWRITEACTIVENO" ]; then
            DBACTIVE=true
            echo "Oracle DB server open in READ WRITE mode"
       fi
       else
           if [ "$i" -eq 1 ]; then
                   /bin/echo -e "Waiting for Oracle DB server\c"
           fi
       fi
       sleep $LOOPDELAY
       /bin/echo -e ".\c"
       i=$(( $i + 1 ))
     done
     if [ "$DBACTIVE" = true ]; then
       TEMPFILE_NAME=`sqlplus -s system/$ORASYSPW@$DBSERVERIPADDR/$DBNAME << EOF
       set pages 0
       set head off
       set feed off
       select name from v\\$tempfile;
       exit
     EOF`
       echo tempfile name is = $TEMPFILE_NAME
       sqlplus -s system/$ORASYSPW@$DBSERVERIPADDR/$DBNAME << EOF2
       alter database tempfile '$TEMPFILE_NAME' resize 2G;
       exit
     EOF2
       cd $SWINGBENCHBINDIR
       ./oewizard -c ../wizardconfigs/oewizard.xml -bigfile -cl -create -cs
//$DBSERVERIPADDR/$DBNAME -dbap $ORASYSPW -scale 1 -u soe -p soe -ts soe -tc 16
     else
       echo "Unable to connect to database.. perhaps not active or incorrect connect
information"
       echo "Retries = $i"
     fi
    path: /root/sbsetup.sh
    permissions: '0755'
  - content: |
     cd
     . ./dbserver.cfg
     cd $SWINGBENCHBINDIR
     ./oewizard -c ../wizardconfigs/oewizard.xml -cl -drop -cs //$DBSERVERIPADDR/$DBNAME -
dbap $ORASYSPW -scale 1 -u soe -p soe -ts soe
    path: /root/dropsoe.sh
    permissions: '0755'
  - content: |
     cd
     . ./dbserver.cfg
     cd $SWINGBENCHBINDIR
     ./swingbench -cs "//$DBSERVERIPADDR/$DBNAME"
     cd
    path: /root/runsb.sh
    permissions: '0755'
runcmd:
  - systemctl restart vncserver@:1
  - mount <MY_NFS_SERVER_IP>:/export/stage /mnt
  - if [ `uname -p` = "ppc64le" ]; then cp /mnt/client/12g/instantclient*.leppc64.*.zip
/root; elif [[ `uname -p` = "ppc64" ]]; then cp /mnt/client/12g/instantclient*.ppc64.*.zip
/root; fi
```

```
  - unzip -o /mnt/Swingbench/swingbench261076.zip -d /opt
  - cd /root; for i in `ls | grep instantclient`; do unzip -o $i -d /opt; rm -f $i; done
  - chmod -R 755 /opt/swingbench /opt/instantclient_12_1
  - /bin/echo -e "export PATH=\$PATH:/opt/`ls /opt | grep instantclient` \nexport
LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/opt/`ls /opt | grep instantclient`" >> /root/.bash_profile
  - umount /mnt
  - cd /root; rm -f instantclient*.zip
  - /bin/echo -e "\n*** please wait for sbsetup.sh completion msg.." | wall
  - sleep 180; /root/sbsetup.sh >> /root/sbsetup.log 2>&1
  - /bin/echo -e "\n*** sbsetup.sh ended. See /root/sbsetup.sh.log." | wall
EOT

  network {
    name = "${var.openstack_network_name}"
  }
}
```

10.3.2 variables.tf

```
######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#----------------------------------------------------------------------
######################################################################
##
##      Deploy LOP with Oracle client and Swingbench for oraclient
##
######################################################################

variable "dbclient_name" {
  type = "string"
  description = "DB client name (VM) on PowerVC"
  default = "dbc"
}

variable "openstack_image_name_dbclient" {
  type = "string"
  description = "PowerVC image name"
}

variable "openstack_flavor_name_dbclient" {
  type = "string"
  description = "PowerVC Compute Template name"
}

variable "openstack_network_name" {
  type = "string"
  description = "PowerVC network name"
}

variable "dbserver_user_data" {
  type = "string"
  description = "Oracle DB name"
}

variable "dbserver_ipaddr" {
  type = "string"
  description = "DB server IP address"
}

variable "dbserver_ts" {
  type = "string"
  description = "Unique id to be appended to db client name"
}
```

10.3.3 outputs.tf

```
######################################################################
# Copyright: IBM Corp., 2020
#
# Written by: Stephen Poon, Ralf Schmidt-Dannert
# IBM North America Systems Hardware
# Power Technical Team, Oracle
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
#----------------------------------------------------------------------
#####################################################################
##
##      Deploy LOP with Oracle client and Swingbench for oraclient
##
#####################################################################

output "dbclient_ipaddr" {
  value = "${format("%s",join(",",
openstack_compute_instance_v2.dbclient.*.network.0.fixed_ip_v4))}"
}

output "dbclient_vm_name" {
  value = "${format("%s",join(",", openstack_compute_instance_v2.dbclient.*.name))}"
  description = "dbclient VM name"
}

output "dbserver_connect" {
  value = "${format("Swingbench connect string: //%s/%s",var.dbserver_ipaddr,
var.dbserver_user_data)}"
}
```

10.3.4 camtemplate.json

```
{
   "name": "oraclient",
   "description": "Description for oraclient.",
   "type": "userCreated",
   "manifest": {
     "template_type": "Terraform",
     "template_format": "HCL",
     "template_provider": "OpenStack"
   },
   "metadata": {
     "bullets": []
   }
}
```

10.3.5 camvariables.json

```
{
  "input_datatypes": [],
  "input_namespaces": [],
  "output_namespace": "",
  "input_groups": [
    {
      "name": "input-parms",
      "label": "Input parameters"
    }
  ],
  "output_groups": [
    {
      "name": "output_parms",
      "label": "Output parmeters"
    }
  ],
  "template_input_params": [
    {
      "name": "dbclient_name",
      "label": "db client VM name prefix",
      "description": "DB client VM name prefix on PowerVC",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_image_name",
      "label": "Openstack Image Name",
      "description": "PowerVC image name",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_flavor_name",
      "label": "Openstack Flavor Name",
      "description": "flavor name, or PowerVC Compute Template name",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "openstack_network_id",
      "label": "Openstack Network id",
      "description": "PowerVC network id",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "dbserver_user_data",
      "label": "DB Server's DB name",
      "description": "Oracle DB name",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "dbserver_ipaddr",
      "label": "DB Server ip address",
      "description": "DB server IP address",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": true,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": true
    },
    {
      "name": "dbserver_ts",
      "label": "unique id in dbclient VM name",
      "description": "Unique id to be appended to db client VM name prefix",
      "type": "string",
      "default": "",
      "validation": "",
      "group_name": "input-params",
      "required": false,
      "secured": false,
      "hidden": false,
      "immutable": false,
      "immutable_after_create": false
    }
  ],
  "template_output_params": [
    {
      "name": "dbclient_ipaddr",
      "label": "DB client IP address",
      "description": "dbclient IP address",
      "group_name": "output_parms",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string",
      "default": "",
      "validation": "",
      "required": true,
      "immutable": false,
      "immutable_after_create": false
    },
    {
      "name": "dbclient_vm_name",
      "label": "DB client VM name",
      "description": "dbclient VM name",
      "group_name": "output_parms",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string",
      "default": "",
      "validation": "",
      "required": true,
      "immutable": false,
      "immutable_after_create": false
    },
    {
      "name": "dbserver_connect",
      "label": "Oracle db jdbc connect string",
      "description": "Oracle database jdbc connect string",
      "group_name": "output_parms",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string",
      "default": "",
      "validation": "",
      "required": true,
      "immutable": false,
      "immutable_after_create": false
    },
    {
      "name": "dbclient_vnc",
      "label": "dbclient VNC host display",
      "description": "dbclient VNC server and display number",
      "group_name": "output_parms",
      "secured": false,
      "hidden": false,
      "shortname": "",
      "type": "string",
      "default": "",
      "validation": "",
      "required": true,
      "immutable": false,
      "immutable_after_create": false
    }
  ]
}
```

10.4 Service definition

Note that the below requires significant adjustments to match your specific environment and is only meant as illustration how the service configuration will look like. See the chapter on creating the service for more details.

10.4.1 oradbaas_oraclient.json

```
{
  "service": {
    "specVersion": "v3",
    "catalog_metadata": {
      "name": "Oracle 12C on AIX with Swingbench Client on LoP",
      "description": "Deploys an AIX VM running Oracle database 12c and a Linux on Power VM
with Oracle Instant Client and Swingbench.",
      "image": "serviceicon_1.svg",
      "category": "WSC T&SA demo",
      "bullets": [],
      "providerDisplayName": "IBM",
      "longDescription": "",
      "documentationUrl": "",
      "supportUrl": "",
      "bindable": false,
      "updatable": "",
      "systemTags": true,
      "author": "admin"
    },
    "tags": [
      {
        "name": "request_group",
        "label": "request_group",
        "type": "string",
        "immutable": true,
        "hidden": false,
        "required": false,
        "secured": false,
        "description": "The current context group id of the current user that requested the
provisioning of the template. This is important because a user may be part of more than one
group and this identifies the context of which group the user made the request.",
        "isSystemTag": true,
        "default": "${svc_instance.group}",
        "customtype": "string",
        "permission": "Read-Only"
      },
      {
        "name": "request_user",
        "label": "request_user",
        "type": "string",
        "immutable": true,
        "hidden": false,
        "required": false,
        "secured": false,
        "description": "The user id of the current user that requested the provisioning of
the template.",
        "isSystemTag": true,
        "default": "${svc_instance.owner}",
        "customtype": "string",
        "permission": "Read-Only"
      },
      {
        "name": "service_name",
        "label": "service_name",
        "type": "string",
        "immutable": true,
        "hidden": false,
        "required": false,
        "secured": false,
        "description": "Name provided for the service instance by the end user at time of
request",
        "isSystemTag": true,
        "default": "${svc_instance.name}",
        "customtype": "string",
        "permission": "Read-Only"
      },
      {
        "name": "service_identifier",
        "label": "service_identifier",
        "type": "string",
        "immutable": true,
        "hidden": false,
        "required": false,
        "secured": false,
        "description": "Unique identifer generated by IT&SA which is mapped to the service
instance of the template provisioned.",
        "isSystemTag": true,
        "default": "${svc_instance.service_instance_id}",
        "customtype": "string",
        "permission": "Read-Only"
      }
    ],
    "actions": [
      {
        "type": "provision",
        "name": "Provision",
        "description": "Default action for deployment of service",
        "input_parameters": [
          {
            "name": "dbserver_user_data",
            "label": "Oracle dbname",
            "customtype": "string",
            "type": "string",
            "immutable": false,
            "hidden": false,
            "required": true,
            "secured": false,
            "description": "Oracle DB name",
            "view": "create new",
            "default": "RSDtest"
          }
        ],
        "flow": {
          "conditions": [],
          "templates": [
            {
              "oradbaas": {
                "title": "oradbaas_56f6b8",
                "template_name": "oradbaas",
                "version": "master",
                "id": "oradbaas13c2cfb6",
                "template_type": "Terraform",
                "template_content_type": "OpenStack",
                "template_provider": "OpenStack",
                "instance_name": "dbs",
                "cloud_connection_name": "PowerVCDev",
                "template_data_objects": {},
                "template_params": {
                  "dbserver_name": "dbs",
                  "openstack_image_name": "DBaaSv3_Ora_SSD",
                  "openstack_flavor_name": "ora_dbaas",
                  "openstack_network_id": "464aceff-141d-453d-a40b-7008d9c06614",
                  "dbserver_user_data": "${input_parameters.dbserver_user_data}"
                },
                "outputs": [
                  "dbserver_ipaddr",
                  "dbserver_vm_name",
                  "dbserver_ts"
                ],
                "error": false,
                "isErrorFlow": false,
                "warning": false
              }
    },
    {
      "oraclient": {
        "title": "oraclient_384925",
        "template_name": "oraclient",
        "version": "master",
        "id": "oracliend1356a5d",
        "template_type": "Terraform",
        "template_content_type": "OpenStack",
        "template_provider": "OpenStack",
        "instance_name": "dbc",
        "cloud_connection_name": "PowerVCDev",
        "template_data_objects": {},
        "template_params": {
          "dbclient_name": "dbc",
          "openstack_image_name_dbclient": "img_rhel76le_pvcdev",
          "openstack_flavor_name_dbclient": "small",
          "openstack_network_name": "AON_93",
          "dbserver_user_data": "${templates.oradbaas13c2cfb6.dbserver_user_data}",
          "dbserver_ipaddr": "${templates.oradbaas13c2cfb6.output.dbserver_ipaddr}",
          "dbserver_ts": "${templates.oradbaas13c2cfb6.output.dbserver_ts}"
        },
        "outputs": [
          "dbclient_ipaddr",
          "dbclient_vm_name",
          "dbserver_connect"
        ],
        "error": false,
        "isErrorFlow": false,
        "warning": false
      }
    }
  ],
  "resthooks": [],
  "notifications": [],
  "sequence": {
    "0": "oradbaas13c2cfb6",
    "1": "oracliend1356a5d"
  },
  "error_sequence": {}
},
"output_parameters": [
  {
    "name": "dbserver_ipaddr",
    "label": "DB server IP address",
    "customtype": "string",
    "type": "string",
    "immutable": true,
    "hidden": false,
    "required": false,
    "secured": false,
    "description": "DB server IP address",
    "default": "${templates.oradbaas13c2cfb6.output.dbserver_ipaddr}"
  },
  {
    "name": "dbserver_vm_name",
    "label": "DB server VM name",
    "customtype": "string",
    "type": "string",
    "immutable": true,
    "hidden": false,
    "required": false,
    "secured": false,
    "description": "DB server VM name in PowerVC",
    "default": "${templates.oradbaas13c2cfb6.output.dbserver_vm_name}"
  },
  {
    "name": "dbclient_ipaddr",
    "label": "dbclient IP address",
          "customtype": "string",
          "type": "string",
          "immutable": true,
          "hidden": false,
          "required": false,
          "secured": false,
          "description": "DB client IP address",
          "default": "${templates.oracliend1356a5d.output.dbclient_ipaddr}"
        },
        {
          "name": "dbclient_vm_name",
          "label": "DB client VM name",
          "customtype": "string",
          "type": "string",
          "immutable": true,
          "hidden": false,
          "required": false,
          "secured": false,
          "description": "DB client VM name in PowerVC",
          "default": "${templates.oracliend1356a5d.output.dbclient_vm_name}"
        },
        {
          "name": "ora_dbname",
          "label": "Oracle dbname",
          "customtype": "string",
          "type": "string",
          "immutable": true,
          "hidden": false,
          "required": false,
          "secured": false,
          "description": "Oracle Database name",
          "default": "${templates.oradbaas13c2cfb6.output.dbname}"
        },
        {
          "name": "oradb_connect",
          "label": "Oracle DB connect string",
          "customtype": "string",
          "type": "string",
          "immutable": true,
          "hidden": false,
          "required": false,
          "secured": false,
          "description": "Oracle DB connect string",
          "default": "${templates.oracliend1356a5d.output.dbserver_connect}"
        }
      ]
    }
  ],
  "plans": [
    {
      "name": "Standard",
      "description": "To deploy a Standard plan",
      "actions": [],
      "plan_parameters": []
    }
  ]
 }
}
```

11 References

- Containers, 12 August 2019, IBM Cloud Education
  https://www.ibm.com/cloud/learn/containers

- IBM Terraform & Service Automation documentation
  https://www.ibm.com/support/knowledgecenter/en/SS2L37_4.2.0.0/kc_welcome.html

- Concepts and key components of Terraform & Service Automation
  https://www.ibm.com/cloud/garage/content/course/ibm-cloud-private-automation-manager/3

- HashiCorp Terraform Documentation
  https://www.terraform.io/docs/index.html

- HashiCorp Introduction to Terraform
  https://www.terraform.io/intro/index.html

- HashiCorp Terraform Configuration Syntax
  https://www.terraform.io/docs/configuration/syntax.html

- HashiCorp Terraform Documentation - openstack_compute_instance_v2
  https://www.terraform.io/docs/providers/openstack/r/compute_instance_v2.html

- Canonical Ltd., cloud-init Documentation
  https://cloudinit.readthedocs.io/en/latest/

- Canonical Ltd., Cloud Config Examples
  https://cloudinit.readthedocs.io/en/latest/topics/examples.html

- Oracle Instant Client, Oracle Corporation
  https://www.oracle.com/database/technologies/instant-client.html

- Dominic Giles, Swingbench
  http://www.dominicgiles.com/swingbench.html

- Technical Whitepaper, Deploying Oracle Database as a Service with IBM Cloud PowerVC Manager 1.3.1, Stephen R. Poon, Ralf Schmidt-Dannert, May 6, 2017
  https://www.ibm.com/support/pages/node/6355627

- Blog based on content from "Infrastructure as Code: Terraform meets PowerVC, Joe Cropper, published on June 29, 2017 / Updated on June 30, 2017"
  http://gibsonnet.net/blog/cgaix/html/Using_Terraform_with_PowerVC_to_deploy_new_AIX_VMs.html

12 Notices and Trademarks