HPE GreenLake for private cloud

Utilizing application blueprints to simplify deployment of a multinode application using GreenLake for private cloud
Contents

1: Executive summary ......................................................................................................................... 4
2: HPE GreenLake for private cloud .................................................................................................. 5
3: eShop Multi-Tiered Application .................................................................................................... 7
4: Configuration of Ansible and Cypher Services ............................................................................ 8
   Integrate Ansible ................................................................................................................................. 8
   Creating Cypher Keys for Secure Secret Management ................................................................. 11
   Using Cypher Keys in Ansible Playbooks ....................................................................................... 13
5: Create Automation Tasks and Provisioning Workflows to Consume Ansible Playbooks ............. 14
   Create Automation Task for Database Tier ....................................................................................... 14
   Create Automation Task for App Tier ............................................................................................... 15
   Create Automation Task for Web Tier .............................................................................................. 15
   Create Workflow for Database Tier ................................................................................................. 16
   Create Workflow for App Tier .......................................................................................................... 17
   Create Workflow for Web Tier .......................................................................................................... 18
6. Creating instances, virtual machines, applications and application blueprints............................ 19
   Creating Instance Type for Web Tier .............................................................................................. 20
   Creating Layout for Web Tier ........................................................................................................... 21
   Associating Node to Layout for Web Tier ....................................................................................... 22
7: Summary ........................................................................................................................................ 37
8: Resources and additional links ...................................................................................................... 38
1: Executive summary

In the early days of cloud computing, private clouds promised the scalability, elasticity, and manageability of public clouds, combined with the security and control within on-premises data center environments. Providing a private cloud turned out to be harder than everyone expected, and some of the early private clouds implementations didn’t offer the scalability, elasticity, and resilience that were the most important characteristics of the cloud environment. Over time, however, vendor offerings improved.

Today, while private clouds still retain their identity as a separate offering, they are more likely to be one part of a broader hybrid cloud strategy. One of the primary drivers of a hybrid cloud strategy is the realization that public clouds don’t meet every enterprise need – and that a ‘cloud first’ or ‘cloud migration’ strategy need not be all about public cloud offerings. There’s a growing realization that it’s often better to create the cloud closer to where the data is processed, than the expense of moving the data to the public cloud.

The role of private clouds becomes part of the hybrid cloud strategy discussion – more an implementation option than a strategic decision in its own right. It’s really more about the application and the outcome and transforming those things to take advantage of modern systems than a light-and-dark contest between public and private cloud. Depending on the exact scenario, it could be cheaper to run [a workload] in private cloud versus public cloud [or vice versa], but it is very dependent on the workload and outcome.

This workload centricity is at the heart of hybrid cloud, as workloads connect the infrastructure to the applications that IT puts in front of customers. Distinctions of public vs. private vs. hybrid eventually become implementation details that organizations can configure as a matter of policy, while supporting the changing needs of customers in the digital era.

Today’s IT brings complex deployments and challenges. IT organizations are required to deal with clustered applications, multiple datacenters, public, private, and hybrid clouds, and applications with complex dependencies. You need a solution and approach that can orchestrate and automate your processes simply and ensure that all tasks happen in the proper order.

Traditionally, deploying and operating enterprise workloads was a time-consuming and manual process. It often involved repetitive tasks, such as sizing, provisioning and configuring resources like virtual machines (VMs); establishing VM clusters and load balancing; creating storage logical unit numbers (LUNs); invoking virtual networks; making the actual deployment; and then monitoring and managing availability and performance.

Although each of these repetitive and manual processes is effective, they are inefficient and often fraught with errors. These errors can lead to troubleshooting, which delays the workload’s availability. They might also expose security vulnerabilities that can put the enterprise at risk.

With cloud automation, an organization eliminates these repetitive and manual processes for workload deployment and management. To achieve cloud automation, an IT team needs to use orchestration and automation tools that run on top of their virtualized environment. Orchestration enables an administrator to codify the various steps and processes involved with workload deployment and management, while automation invokes those steps without human intervention.

This whitepaper describes the application blueprint capabilities inherently built with the HPE GreenLake for private cloud service that enables savings in both time and money, elimination of errors, and construction of more predictable and reliable services beyond bare VM’s.
2: HPE GreenLake for private cloud

At the center of HPE GreenLake for hybrid cloud is a SaaS platform for managing hybrid cloud environments, called the HPE GreenLake Central platform. HPE GreenLake Central platform is a hub for HPE GreenLake services designed to deliver and help manage a hybrid cloud.

One of the services offered through HPE GreenLake Central platform is HPE GreenLake for private cloud. This service is:

- An HPE designed, implemented, owned and operated private cloud that is deployed at a customer site
- Offered as a consumption based service that allows customers to better align costs to outcomes
- Provides an intuitive self-service portal UI to create and manage private cloud services such as compute, storage, and network, as well as application blueprints (example described in this whitepaper)
- Provides rich set of capabilities, automation and reusable application blueprints to orchestrate infrastructure, application and services across the private cloud
This whitepaper explains how to use HPE GreenLake for private cloud portal to design and deploy a multi tiered application blueprint using Ansible playbooks.

At a high level, the steps required to create the application blueprint are:

1. **Integrate code repository** – Provide a connection to a git repo containing automation artifacts (e.g. Ansible playbooks).
2. **Create automation tasks & workflows** – Tasks are individual automation elements (e.g. an individual playbook or a script). Workflows consist of one or more tasks. Tasks and workflows can be integrated into HPE GreenLake for private cloud deployments individually or in combination.
3. **Create instance types & layouts** – Instances define a set of VMs that can correlate to a single horizontally scalable entity (like a database). Layouts are attached to instance types, and are a way of associating underlying VM templates and structures.
4. **Create application blueprint** – Application blueprints connect automation tasks, workflows, instance types, and layouts together to form a singular entity for deployment and lifecycle management. In this example, the blueprint we create will contain all required information for deploying and configuring the various VMs and applications needed to create our eShop application.
3: eShop Multi-Tiered Application

Before we jump in, let's spend a minute on understanding the architecture of the sample multi-tiered eShop application which will be used in this document.

This eShop application includes three tiers namely Web Tier, App Tier and Database Tier.

- **Web Tier** - The web tier is the front end layer in this architecture and consists of the user interface. The user interface is accessible through web browser. The web tier communicates with other layers through API calls.
- **App Tier** - The application tier contains the functional business logic that drives an application's core capabilities. These include, for example, identity services, Catalog services, ordering services, etc.
- **Database Tier** - The database tier comprises of the databases and data access layer. For example, SQL Server, MongoDB, CosmosDB, etc. Data is accessed by the application layer via API calls.

The example code repository contains Ansible playbooks to bring up each tier individually as depicted below (database.yml, api.yml and web.yml). The details of the individual playbooks is not important to understanding the application blueprint structure, so we have not included it here.
4: Configuration of Ansible and Cypher Services

Ansible requires access to port 22 on Linux and port 5985 on Windows to function. In many enterprise environments, these ports are blocked due to security policies, which makes Ansible unusable. Using the agent command bus feature, HPE GreenLake for private cloud is able to get around this problem and make Ansible usable in such environments. Using the agent command bus feature, Ansible playbooks can still be executed using secure port 443.

Integrate Ansible

The first step is to add the Ansible integration in the HPE GreenLake for private cloud portal and point to the Git repository where the Ansible playbooks reside.
- Navigate to Administration > Integrations
• Click on NEW INTEGRATION and select Integration Type 'Ansible'

• Populate the following fields, as shown in the figure below:

**NAME:** Name of the Ansible Integration in GLPL Portal  
**ENABLED:** Enabled by default  
**ANSIBLE GIT URL:** https URL format of Ansible Git repo to use.  
**PLAYBOOKS PATH:** Path of the Playbooks relative to the Git URL.  
**ROLES PATH:** Path of the Roles relative to the Git URL.  
**GROUP VARIABLES PATH:** Path of the Group Variables relative to the Git URL.  
**HOST VARIABLES PATH:** Path of the Host Variables relative to the Git URL.  
**ENABLE VERBOSE LOGGING:** Enable to output verbose logging for Ansible task history  
**USE MORPHEUS AGENT COMMAND BUS:** Enable for Ansible Playbooks to be executed via Morpheus Agent Command Bus instead of SSH
## EDIT ANSIBLE INTEGRATION

<table>
<thead>
<tr>
<th>NAME</th>
<th>#shop</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLED</td>
<td></td>
</tr>
<tr>
<td>ANSIBLE GIT URL</td>
<td><a href="https://github.com/ansible-examples-eshop.git">https://github.com/ansible-examples-eshop.git</a></td>
</tr>
<tr>
<td>PLAYBOOKS PATH</td>
<td>ecommerce_application</td>
</tr>
<tr>
<td>ROLES PATH</td>
<td>ecommerce_application/roles</td>
</tr>
<tr>
<td>GROUP VARIABLES PATH</td>
<td>ecommerce_application/group_vars</td>
</tr>
<tr>
<td>HOST VARIABLES PATH</td>
<td>ecommerce_application/hosts/hosts</td>
</tr>
<tr>
<td>USE ANSIBLE GALAXY</td>
<td></td>
</tr>
<tr>
<td>ENABLE VERBOSE LOGGING</td>
<td></td>
</tr>
<tr>
<td>USE MORPHEUS AGENT COMMAND BUS</td>
<td></td>
</tr>
</tbody>
</table>

**Git Options**

<table>
<thead>
<tr>
<th>KEYPAIR</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>USERNAME</td>
<td></td>
</tr>
<tr>
<td>PASSWORD</td>
<td></td>
</tr>
<tr>
<td>ACCESS TOKEN</td>
<td></td>
</tr>
</tbody>
</table>

- Click on “Save Changes”
The second step is to install Ansible on the HPE GreenLake for private cloud appliance. In most cases, it should be automatically installed. You can run `ansible --version` command in the Ansible integration details page to verify this (Administration > Integrations > Select Ansible Integration). Just run `--version` as Ansible is already included in the command.

![Ansible Integration Details](image)

Now that the setup is complete, it is possible run the Ansible playbooks from multiple locations in HPE GreenLake for private cloud.

**Creating Cypher Keys for Secure Secret Management**

The next step is to store the secrets used in the playbooks in a secured place. HPE GreenLake for private cloud has a built-in service called Cypher. Cypher is a secure key/value store, which allows the storage of secret data like credentials or variable sets in a highly encrypted way for future retrieval. When leveraging HPE GreenLake for private cloud and Ansible to execute playbooks at any stage (provisioning, future, or ad hoc), these variables can be seamlessly passed to the playbooks, so variables are maintained and managed only in one place.

- Navigate to Tools - Cypher and select ‘+ADD’ key
- Configure one of the following types of Keys:

1. **Password**: A Cypher password generates a secure password of specified character length in the key pattern with symbols, numbers, upper case, and lower case letters (i.e. password/15/mypass generates a 15 character password).

2. **Secret**: A Cypher secret is the standard secret module that stores a key/value in encrypted form.

3. **UUID**: A Cypher UUID Returns a new UUID by key name when requested and stores the generated UUID by key name for a given lease timeout period.

4. **Key**: A Cypher Key generates a Base 64 encoded AES Key of specified bit length in the key pattern (i.e. key/128/mykey generates a 128-bit key)

For this example, we have chosen to use secret key type, as shown below:
Using Cypher Keys in Ansible Playbooks

To use a Cypher secret in Ansible playbook, use the following syntax:

- `{{ lookup('cypher',secret='secret/MYSQL_SA_PASSWORD') }}`

For example, in our Ansible playbooks for eShop application, we will use the cypher secret as shown below:

- `MYSQL_SA_PASSWORD={{ lookup('cypher',secret='secret/MYSQL_SA_PASSWORD') }}`
5. Create Automation Tasks and Provisioning Workflows to Consume Ansible Playbooks

Provisioning workflows allows users to execute configured tasks in sequence and automate many parts of the application deployment process. Tasks are individual automation elements, and can come from many sources including scripts written directly in the library, as well as those consumed into HPE GreenLake for private cloud through integration of automation services. Workflows consist of one or more automation tasks, as well as additional steps that may be needed to successfully complete the flow (e.g. pre/post processing). In this section, we will create a task and workflow, based on Ansible playbooks, which can be consumed by the instances in the application blueprint.

The first step is to create the tasks for the three application tiers:
- Navigate to Provisioning > Automation > Tasks
- Click "+ADD" and provide a name as it will appear within the list of tasks in HPE GreenLake for private cloud UI
- Finally, identify the repo we integrated at the start, the specific playbook ("database.yml" for database tier, "api.yml" for application tier, "web.yml" for web tier) and set the EXECUTE TARGET to "Resource".

Create Automation Task for Database Tier
Create Automation Task for App Tier

Create Automation Task for Web Tier
The next step will be to create a provisioning workflow that executes the tasks that were created in the previous step.

- Navigate to Provisioning > Automation > WORKFLOWS
- Click "+ADD" > Provisioning Workflow
- Finally, provide a name as it should appear in the list of workflows within HPE GreenLake for private cloud UI. In the Tasks > Provisioning field, we will begin typing "ecommerce_database" to locate our task and select it. Similarly, we will create workflows for application and web tier as well:

Create Workflow for Database Tier

![EDIT WORKFLOW](image-url)
Create Workflow for App Tier

EDIT WORKFLOW

NAME: ecommerce_API

DESCRIPTION:

PLATFORM: All

Tasks

- Pre Provision

- Provision

  1. ecommerce_api

- Post Provision

- Start Service

- Stop Service

- Pre Deploy

- Deploy

- Reconfigure

- Teardown

SAVE CHANGES
Create Workflow for Web Tier

With workflows created to consume the ansible playbooks, we can now use them as part of an application blueprint.
6. Creating **instances, virtual machines, applications and application blueprints**

Instances is a great starting point for taking advantage of self service features and provisioning virtual machines. It’s important to understand the definition of a few terms used within the HPE GreenLake for private cloud application environment.

**Instance:** An instance is a representation of a resource or service. In the most basic form, this a VM with only an OS loaded on it. Taking it further it could be even be a machine with some application or a specific tier installed - for example WordPress service or Database service.

**Virtual Machine:** A node is generic representation of a virtual machine. Instances (defined above) can have many nodes.

**Application:** An application is a collection of instances linked together via application tiers. Tiers allow the user to define segregated sections of instances within an application.

**Application Blueprint:** Application blueprints allow a user to define an application structure for easy reproducibility and deployment into various environments. They can be used to mix and match various instance types to provision an application dependent on multiple layers of services. To consume an application blueprint, you simply navigate to Provisioning > Apps, and select the blueprint you would like to provision.

Below is an example of simple application blueprint, that consists of a web tier running Tomcat, and a database tier running MySQL.
Creating Instance Type for Web Tier

- Navigate to Provisioning > Library > Instance Types
- Click on "ADD". Create with the following settings:

**NEW INSTANCE TYPE**

- **Name**: Web Server
- **Code**: 1WBSR
  - Useful shortcode for provisioning naming schemes and export reference.
- **Category**: Web
- **Icon**: WebServer.png
  - Suggested Dimensions: 150 x 51

**Option Types**

Search option types

**Advanced**

**Environment Prefix**

Used for exportable environment variables when tying instance types together in app contexts. If not specified a name will be generated.

**Environment Variables**

- **Enable Settings**
- **Enable Scaling (Horizontal)**
- **Support Deployments**
  - Requires a data volume be configured on each version. Files will be copied into this location.

**SAVE CHANGES**
Creating Layout for Web Tier

- Create the layout by clicking "Add Layout" and configure the following:

  - **NAME**: ShopWeb
  - **VERSION**: Latest
  - **DESCRIPTION**: 
  - **TECHNOLOGY**: VMware
  - **MINIMUM MEMORY**: This will override any memory requirement set on the virtual image
  - **WORKFLOW**: Select Workflow
  - **ENVIRONMENT VARIABLES**: Name, Value
  - **Option Types**: Search option types
  - **Nodes**: Search nodes

[Image of the layout configuration interface]
Associating Node to Layout for Web Tier

The final step is to create and associate a Node to layout.

- Click "+Add VM Type" and fill out accordingly. Be sure to select an Ubuntu 16.x virtual image.

Creating Instance Type for App Tier

- Repeat the first two steps with the following configurations for application layer.
### Technical white paper

**NEW INSTANCE TYPE**

- **NAME**: Application Server
- **CODE**: <App>

Useful shortcode for provisioning naming schemes and export reference.

- **DESCRIPTION**
- **CATEGORY**: Apps
- **ICON**: ApplicationServer.png

Suggested Dimensions: 130 x 51

**Option Types**

- **Search option types**

**Advanced**

- **ENVIRONMENT PREFIX**: APP_TIER

Used for exportable environment variables when tying instance types together in app contexts. If not specified a name will be generated.

- **ENVIRONMENT VARIABLES**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
</table>

- **ENABLE SETTINGS**
- **ENABLE SCALING (HORIZONTAL)**
- **SUPPORT DEPLOYMENTS**

Requires a data volume be configured on each version. Files will be copied into this location.

**SAVE CHANGES**

- Navigate to Provisioning > Library > Instance Types and Click on "+ADD". Create with the following settings:

---

**Note**

Note the Environment Prefix being set. Using this prefix, HPE GreenLake for private cloud will set APP_TIER_IP and APP_TIER_HOST environment variables in the instances and these variables will be used by instances in web tier to communicate with application layer.
Creating Layout for App Tier

Note the nodes being selected. As there is no changes in the node type configuration, we will be using the same node type for application instance.

Creating Instance Type for Database Tier

- Repeat the first two steps with the following configurations for Database layer
- Navigate to Provisioning > Library > Instances Types and Click on "+ADD". Create with the following settings:
EDIT INSTANCE TYPE

NAME: Database Server

CODE: database

Useful shortcode for provisioning naming schemes and export reference.

DESCRIPTION:

CATEGORY: SQL

ICON: DatabaseServer.jpg

Suggested Dimensions: 150 x 51

Option Types

Search option types

Advanced

ENVIRONMENT PREFIX: DATABASE_TIER

Used for exportable environment variables when tying instance types together in app contexts. If not specified a name will be generated.

ENVIRONMENT VARIABLES

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
</table>

- ENABLE SETTINGS
- ENABLE SCALING (HORIZONTAL)
- SUPPORT DEPLOYMENTS

Requires a data volume be configured on each version. Files will be copied into this location.

SAVE CHANGES
Technical white paper

Note the environment prefix being set. Using this prefix, HPE GreenLake for private cloud will set DATABASE_TIER_IP and DATABASE_TIER_HOST environment variables in the instances and these variables will be used by instances in application tier to communicate with database layer.

Creating Layout for Database Tier

Note the nodes being selected. As there is no changes in the node type configuration, we will be using the same node type for application instance type.
Creating application blueprints

In an HPE GreenLake for private cloud application blueprint there are a few structural concepts to be aware of. Firstly there is a concept of a tier. A tier is a grouping of instances within an app blueprint. Tiers can be used for a variety of things including sequenced booting of instances or even properly creating endpoint groups. An example of a tier structure might be a web tier and a database tier. These tiers can also be marked as connected such that network communication rules can appropriately be defined.

A key value to HPE GreenLake for private cloud is the application based approach. HPE GreenLake for private cloud portal can deploy single VMs or a single tier of an application with multiple VMs as an instance. So in other words, we treat an instance as the top level to a VM. With an application blueprint, we treat that application as the top level to multiple instances. This is why we created the web, application and database instance types. The next step is to stitch them together in an application blueprint. Also note, these can be easily created and modified via JSON or YAML if you would like to automate these application blueprint builds.

- **Create Blueprint**
  - Navigate to Provisioning > Blueprints
  - Select +ADD
  - Enter a NAME for the blueprint, Select *Morpheus* type and Click on Next
  - Optionally add a description, category, and image for the blueprint.

Add Tiers

- In the STRUCTURE section, select + to add a Tier
- Select or enter a Tier Name.
• Select the tier to set boot order. For this sample eShop application, we need database VM to be up first followed by application server and then the web server. So the boot order should be 0 for database tier, 1 for application tier and 2 for web tier.

• Select the tier to connect to other tiers. For this sample eshop application, application tier should be connected to database tier, web tier should be connected to application tier. **Note:** When Tiers are connected, the instances in a Tier will import the evars from instances in a connected tier.

Create Database Tier
Create App Tier

Create Web Tier

Add Instances to Tiers

- In the STRUCTURE section, select + in a tier to add an instance
- Select an instance type
- Optionally add a name for the instance. Instances with blank names will automatically be named based off the app name. You can use the variable `${app.name}` in your instance naming convention to reference the name of the application you are deploying.

Note that database tier is pointed to database instance type we created in the previous step. Similarly, application tier and web tier are pointed to respective instance types.

Add configurations to Instances

- In the STRUCTURE section, select + in an instance to add a configuration
- Select at least one option from group, doud and environment.
- Select ADD CONFIG to create the configuration
- Populate the configuration
  - Configurations can be fully or partially populated
  - Fields can be locked by selecting the lock icon next to the field. Locking prevent the field from being editable when provisioning an app using the blueprint.
  - **IMPORTANT:** Expand the automation section and select the specific workflow. In our example, "ecommerce_DB" for database tier, "ecommerce_API" for app tier and "ecommerce_WEB" for web tier.
Once all desired tiers, instances and configurations are added, select save. The blueprint will be created and will available in the apps section for provisioning.
Deploying applications from application blueprints

Applications can be created from application blueprints:

- Click +ADD on the right side of the main Apps section in Provisioning.
- Select an existing application blueprint and click NEXT.
- Enter a Name of the App and select a Group, Default Cloud and Env can also be selected.
- Click NEXT. Blueprint configurations matching the Group, Cloud and Environment selections will auto-populate the configurations of the instances in the App.
- Configure your instances. Depending on the blueprint configurations settings, instances may already be fully configured. Fields that are locked in a blueprint cannot be edited when creating an app.

Note: Once an instance is fully configured, a green checkmark will appear next to instance. Instances that have required fields that need to be filled in will have a red X and must be completed. If your blueprint is fully configured you can simply select NEXT.

Review the options selected and select COMPLETE. The app will created and the instances will begin provisioning.
Technical white paper

**NEW APP**

**Review App**

**APP**

<table>
<thead>
<tr>
<th>INSTANCE OPTIONS</th>
<th>VOLUMES</th>
<th>NETWORKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME: eShop Application - eApp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOUD: CumuloCloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TYPE: eApp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLAN: Standard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VERSION: Latest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAYOUT: eShopApp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROOT: 60 GB Auto - Datatree SCS: 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VWINOS-693-VIRTUALWARE-4.510-5000-BLUE-NET: BlueAppPool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**WEB**

<table>
<thead>
<tr>
<th>INSTANCE OPTIONS</th>
<th>VOLUMES</th>
<th>NETWORKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME: eShop Application - cweb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOUD: CumuloCloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TYPE: cweb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLAN: customplan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VERSION: Latest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAYOUT: eShopWeb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROOT: 60 GB Auto - Datatree SCS: 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VWINOS-693-VIRTUALWARE-4.510-5000-BLUE-NET: BlueAppPool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DATABASE**

<table>
<thead>
<tr>
<th>INSTANCE OPTIONS</th>
<th>VOLUMES</th>
<th>NETWORKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME: eShop Application - eDatabase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOUD: CumuloCloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TYPE: eDatabase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLAN: customplan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VERSION: Latest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAYOUT: eShopDatabase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROOT: 60 GB Auto - Datatree SCS: 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VWINOS-693-VIRTUALWARE-4.510-5000-BLUE-NET: BlueAppPool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note**
Database tier is created first as it has a boot order of 0
Application finished installing!
7: Summary

Application blueprints are a powerful feature of HPE GreenLake for private cloud, and can be used to maintain application infrastructure “as code” by pulling scripts or, in this case, automation playbooks directly from code repositories. Whether your use case is simple VM provisioning, or complete application provisioning through “infrastructure as code”, HPE GreenLake for private cloud provides can meet your needs by providing a managed, simple to use, on-premise private cloud.

Learn more by exploring the resources below or signing up for a free trial.
8: Resources and additional links

Learn more about cloud services from HPE GreenLake
Video: HPE GreenLake Central platform