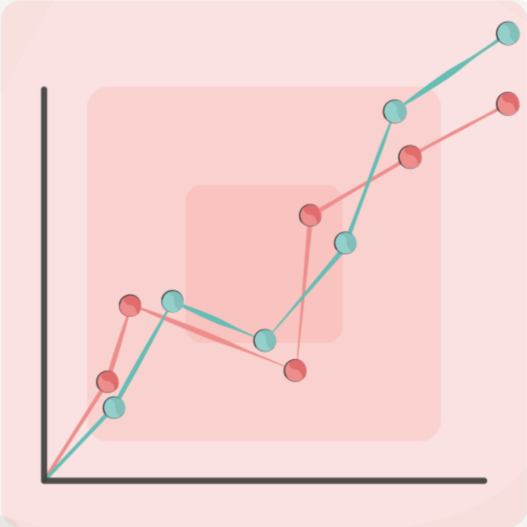
Level 5 Data Engineer
Module 5 Topic 3

Containers and Orchestration

Welcome to today's webinar.



Session objectives

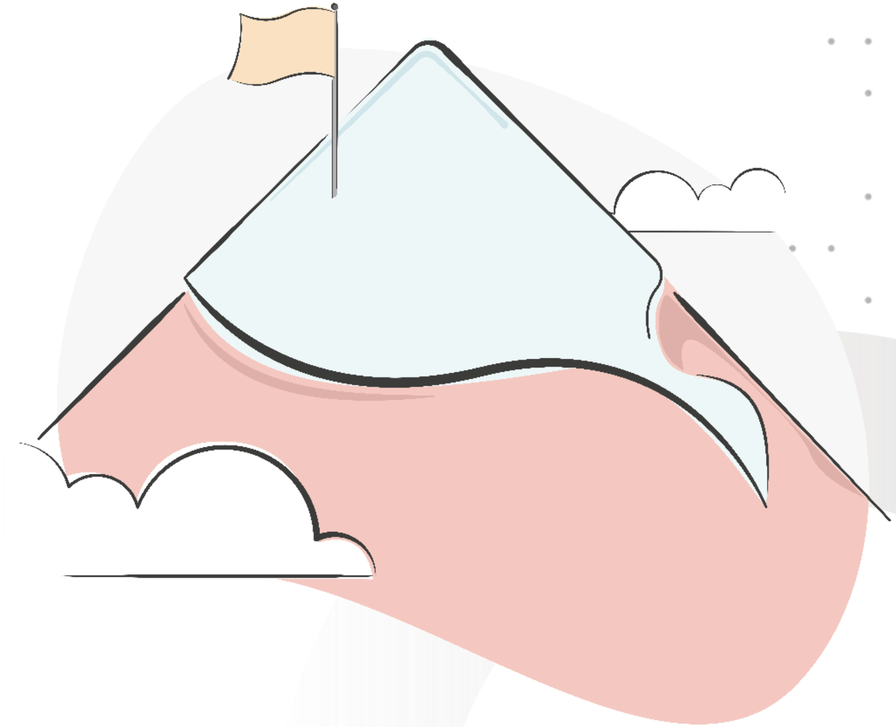
Building Careers Through Education
Everybody Matters.
Stronger Together.
Student, Learner and Client Centric.
Trust and Respect.
Embrace Change.

This webinar supports the following learning outcomes:

- Discuss how cloud automation and orchestration can help streamline the management and maintenance of cloud-based systems and applications;

- Analyse how the use of tools and technologies such as Ansible, Puppet, and Chef may be used to automate routine tasks in order to reduce the potential for human error in cloud-based systems and applications;

- Use tools and technologies such as Kubernetes, Docker Swarm, and Mesos to manage containers and balance workloads to ensure they are working together effectively and high availability exists.

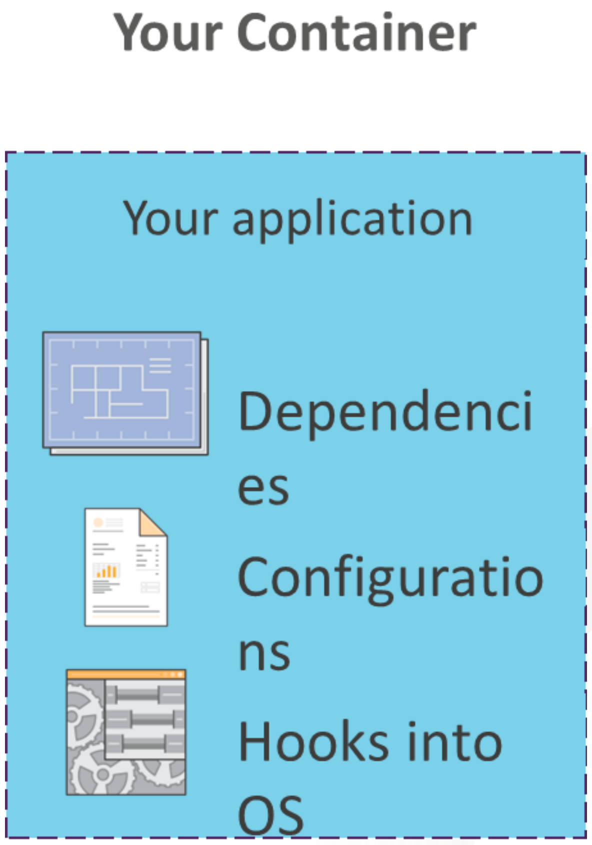


Container basics

Containers are a method of operating system virtualisation

Benefits –

- Repeatable.
- Self-contained environments.
- Software runs the same in different environments.
  - Developer's laptop, test, production.
- Faster to launch and stop or terminate than virtual machines



Your Container

Your application

Dependencies

Configurations

Hooks into OS

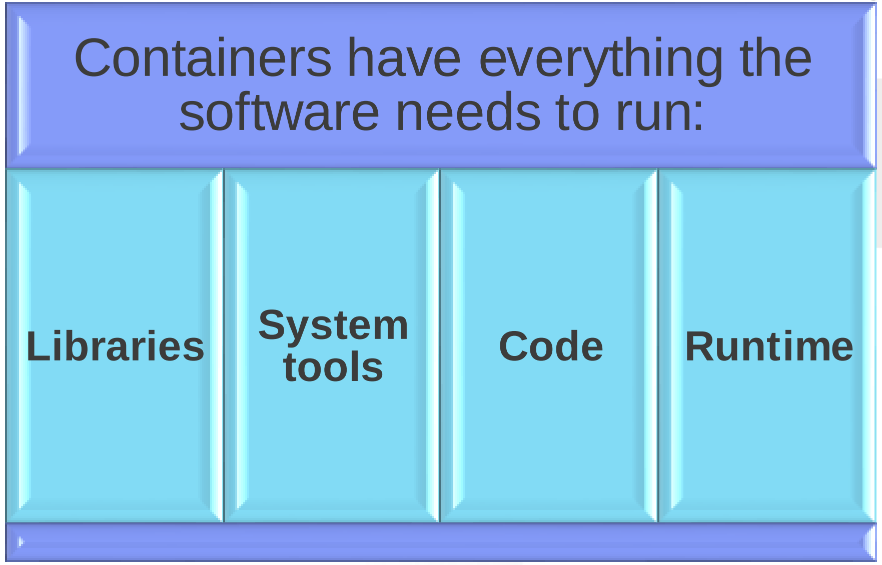
What is Docker?

- **Docker** is a software platform that enables you to build, test, and deploy applications quickly.

- You run containers on Docker.
  - Containers are created from a template called an *image*.

- A **container** has everything a software application needs to run.

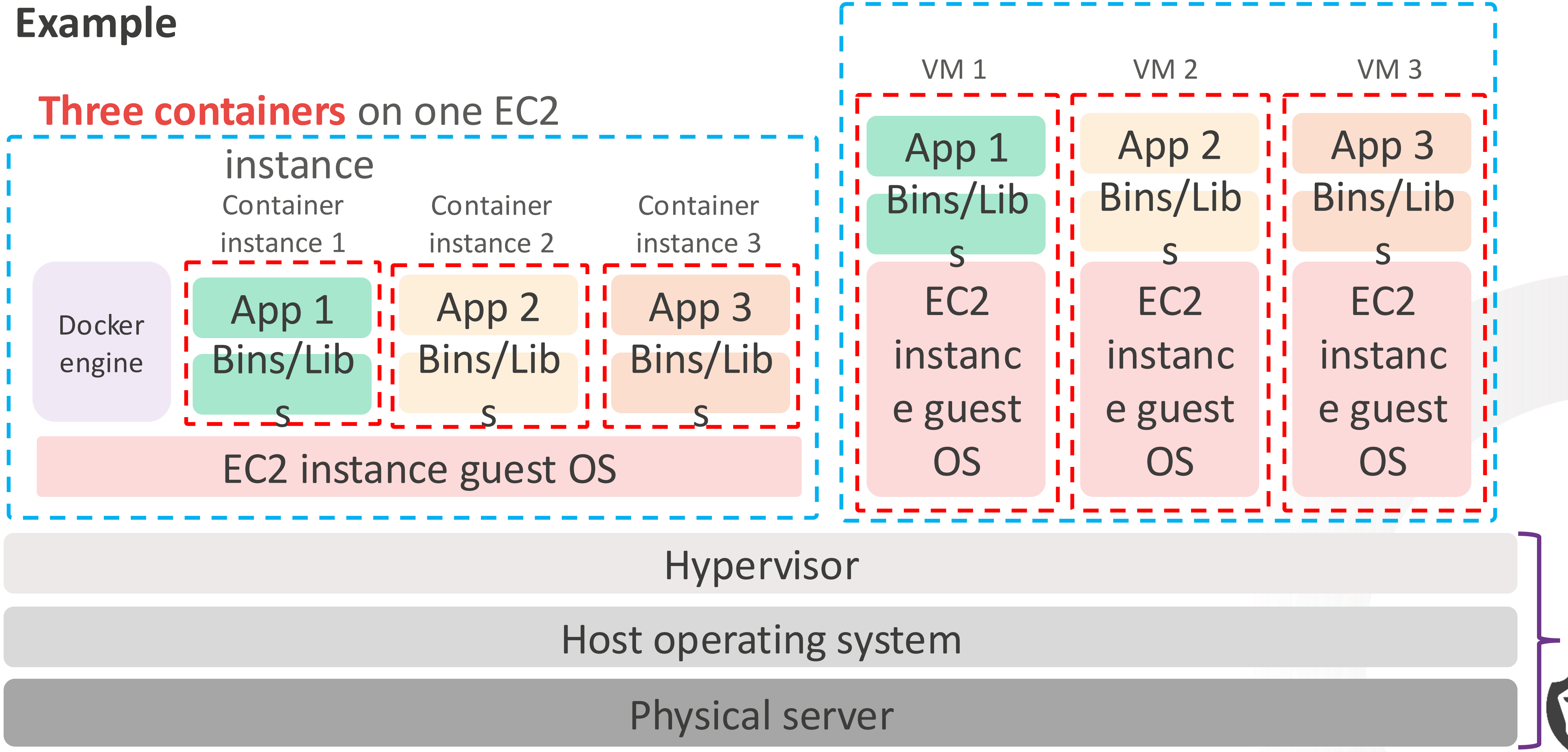
Container



Containers have everything the software needs to run:

Containers versus virtual machines

Example



Three containers on one EC2 instance

Container instance 1 | Container instance 2 | Container instance 3

Docker engine

App 1, Bins/Libs | App 2, Bins/Libs | App 3, Bins/Libs

EC2 instance guest OS

Three virtual machines on three EC2 instances

VM 1 | VM 2 | VM 3

App 1, Bins/Libs, EC2 instance guest OS | App 2, Bins/Libs, EC2 instance guest OS | App 3, Bins/Libs, EC2 instance guest OS

Hypervisor

Host operating system

Physical server

Part of AWS Global Infrastructure

Amazon Elastic Container Service (ECS)

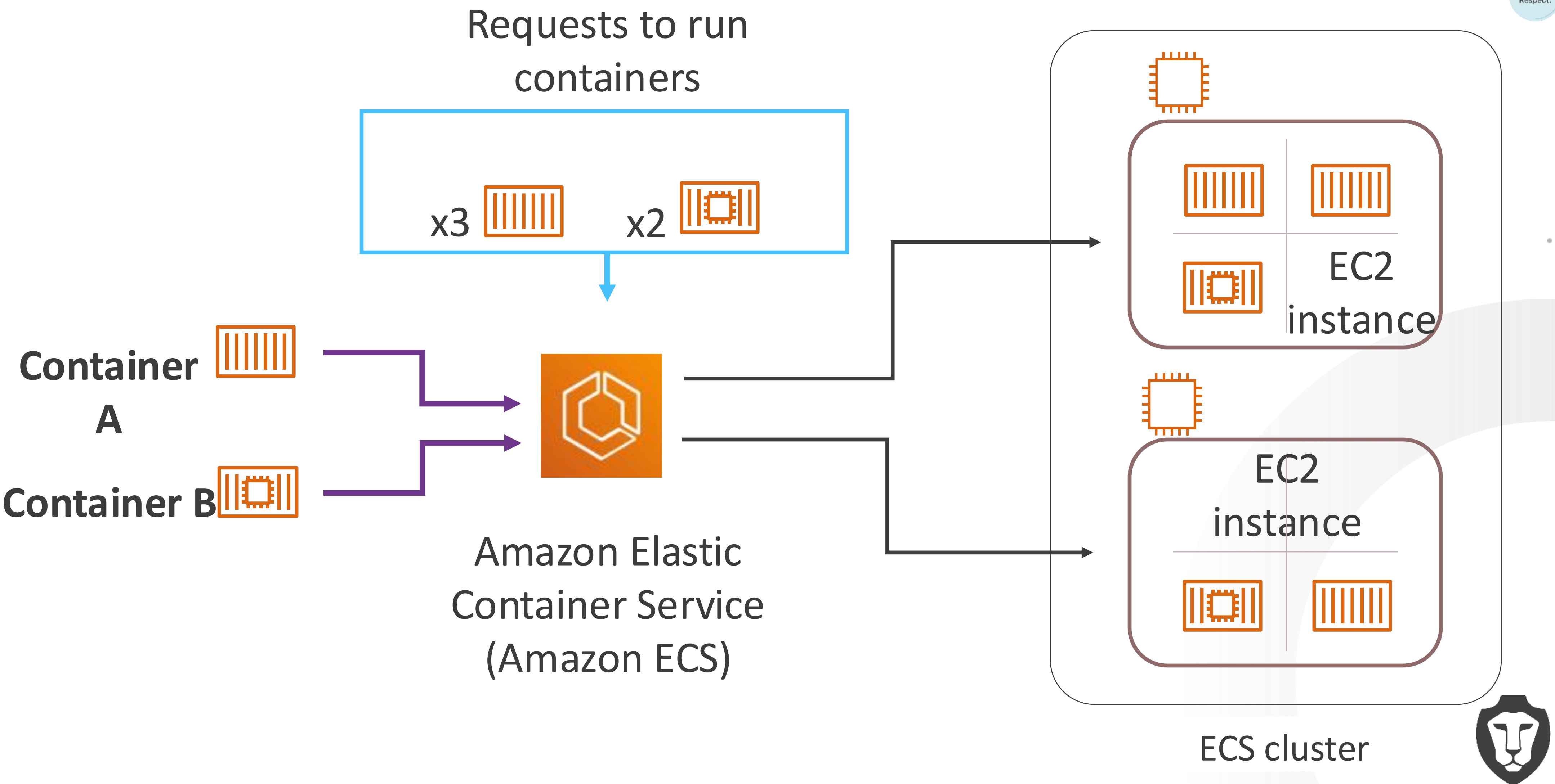
- Amazon Elastic Container Service (**Amazon ECS**) –
  - A highly scalable, fast, container management service

- Key benefits:
  - Orchestrates the running of Docker containers
  - Maintains and scales the fleet of nodes that run your containers
  - Removes the complexity of standing up the infrastructure

- Integrated with features that are familiar to Amazon EC2 service users:
  - Elastic Load Balancing
  - Amazon EC2 security groups
  - Amazon EBS volumes
  - IAM roles

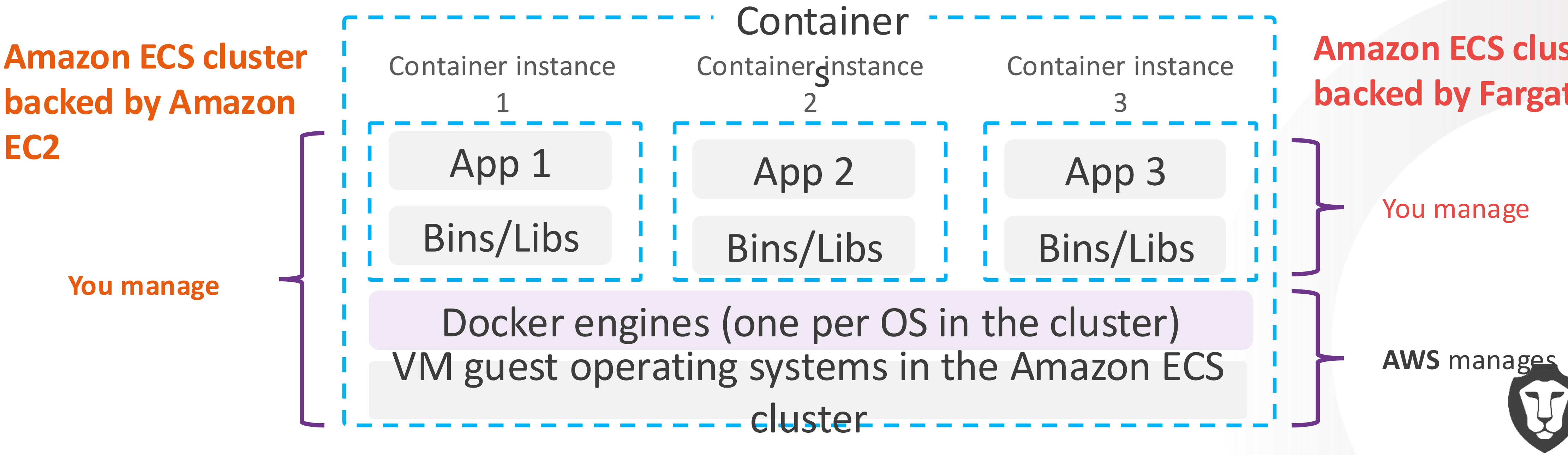
Amazon Elastic Container Service

Amazon ECS orchestrates containers


Requests to run containers
x3
x2
Container A
Container B
Amazon Elastic Container Service (Amazon ECS)
EC2 instance
EC2 instance
ECS cluster

Amazon ECS cluster options

- **Key question**: Do ***you*** want to manage the Amazon ECS cluster that runs the containers?
    - If **yes**, create an **Amazon ECS cluster backed by Amazon EC2** (provides more granular control over infrastructure)
    - If no, create an Amazon ECS cluster backed by AWS Fargate (easier to maintain, focus on your applications)



Amazon ECS cluster backed by Amazon EC2

You manage

Containers

Container instance 1

App 1

Bins/Libs

Container instance 2

App 2

Bins/Libs

Container instance 3

App 3

Bins/Libs

Docker engines (one per OS in the cluster)

VM guest operating systems in the Amazon ECS cluster

Amazon ECS cluster backed by Fargate

You manage

AWS manages

What is Kubernetes?

- **Kubernetes is open source software for container orchestration.**
  - Deploy and **manage containerised applications** *at scale*.
  - The same toolset can be used on premises and in the cloud.
- **Complements Docker.**
  - Docker enables you to run multiple containers on a single OS host.
  - Kubernetes **orchestrates** multiple Docker hosts (nodes).
- **Automates –**
  - Container provisioning.
  - Networking.
  - Load distribution.
  - Scaling.

Activity

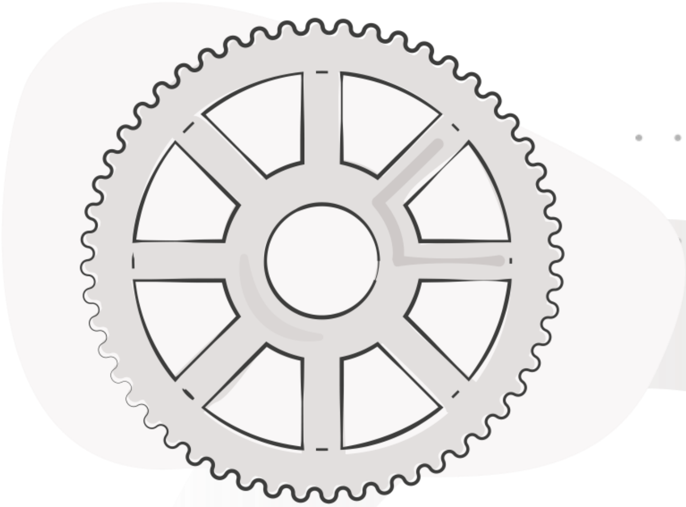
We are now going to:

1. Discuss how cloud automation and orchestration can help streamline the management and maintenance of cloud-based systems and applications;

2. Analyse how the use of tools and technologies such as Ansible, Puppet, and Chef may be used to automate routine tasks in order to reduce the potential for human error in cloud-based systems and applications;

3. Use tools and technologies such as Kubernetes, Docker Swarm, and Mesos to manage containers and balance workloads to ensure they are working together effectively and high availability exists.

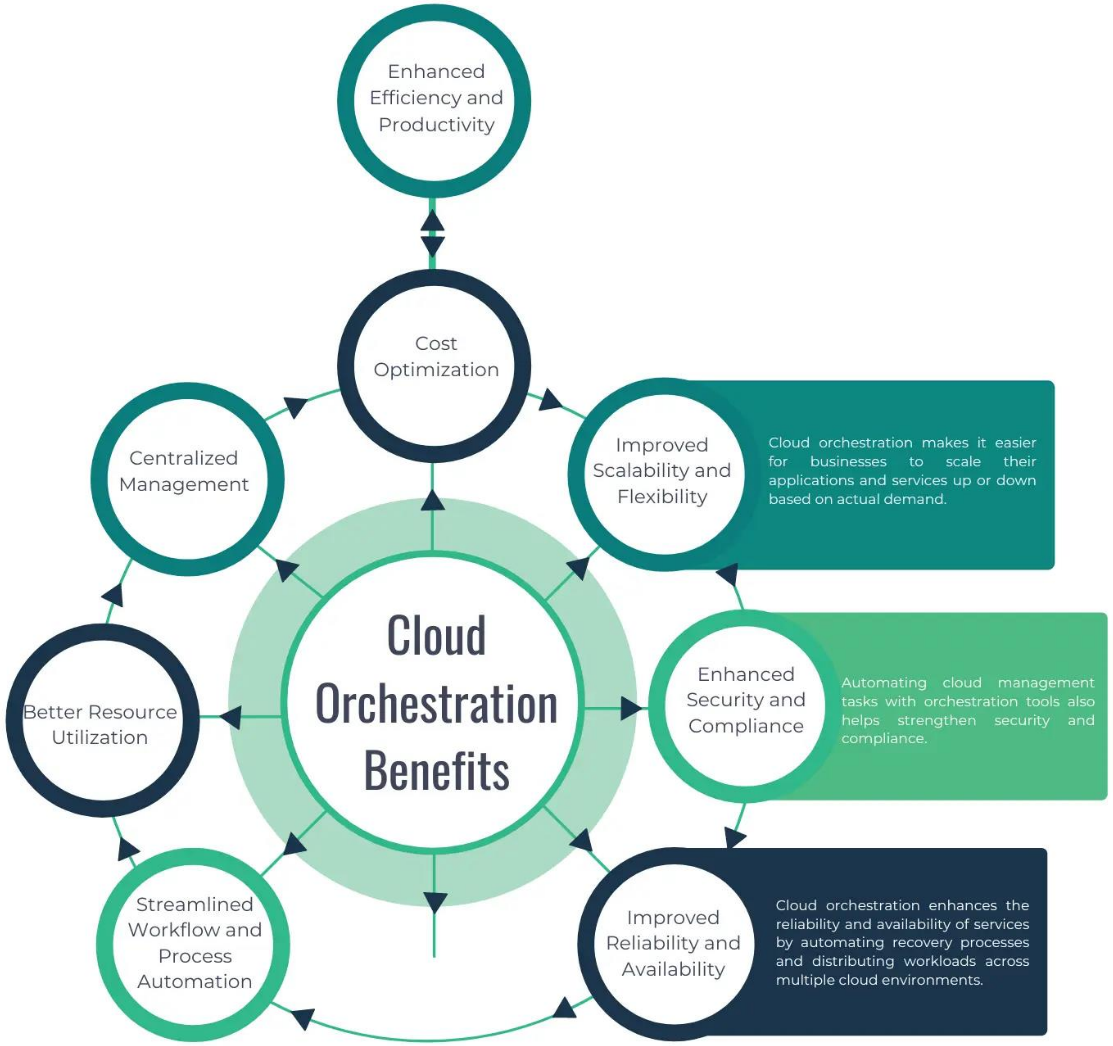
4. Demonstration of Kubernetes on Azure



Introduction to Cloud orchestration

Building Careers Through Education

Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.



Cloud Orchestration Benefits

Enhanced Efficiency and Productivity

Cost Optimization

Centralized Management

Improved Scalability and Flexibility

Cloud orchestration makes it easier for businesses to scale their applications and services up or down based on actual demand.

Better Resource Utilization

Enhanced Security and Compliance

Automating cloud management tasks with orchestration tools also helps strengthen security and compliance.

Streamlined Workflow and Process Automation

Improved Reliability and Availability

Cloud orchestration enhances the reliability and availability of services by automating recovery processes and distributing workloads across multiple cloud environments.

- **Source:** https://www.mega.com/blog/what-is-cloud-orchestration

Discussion

Building Careers Through Education

Everybody Matters.

Stronger Together.

Student, Learner and Client Centric.

Trust and Respect.

Embrace Change.

Use cases:

- Automated Deployment and Scaling of Web Applications
- Continuous Integration and Continuous Deployment (CI/CD)
- Infrastructure as Code (IaC)
- Disaster Recovery and Backup Automation
- Automated Security Compliance and Patching
- Multi-Cloud Management
- Automated Compliance Auditing

BPP

Automation tools – Chef, Puppet & Ansible

Ansible:

- Agentless architecture
- Uses YAML for defining playbooks
- Easy to learn and use

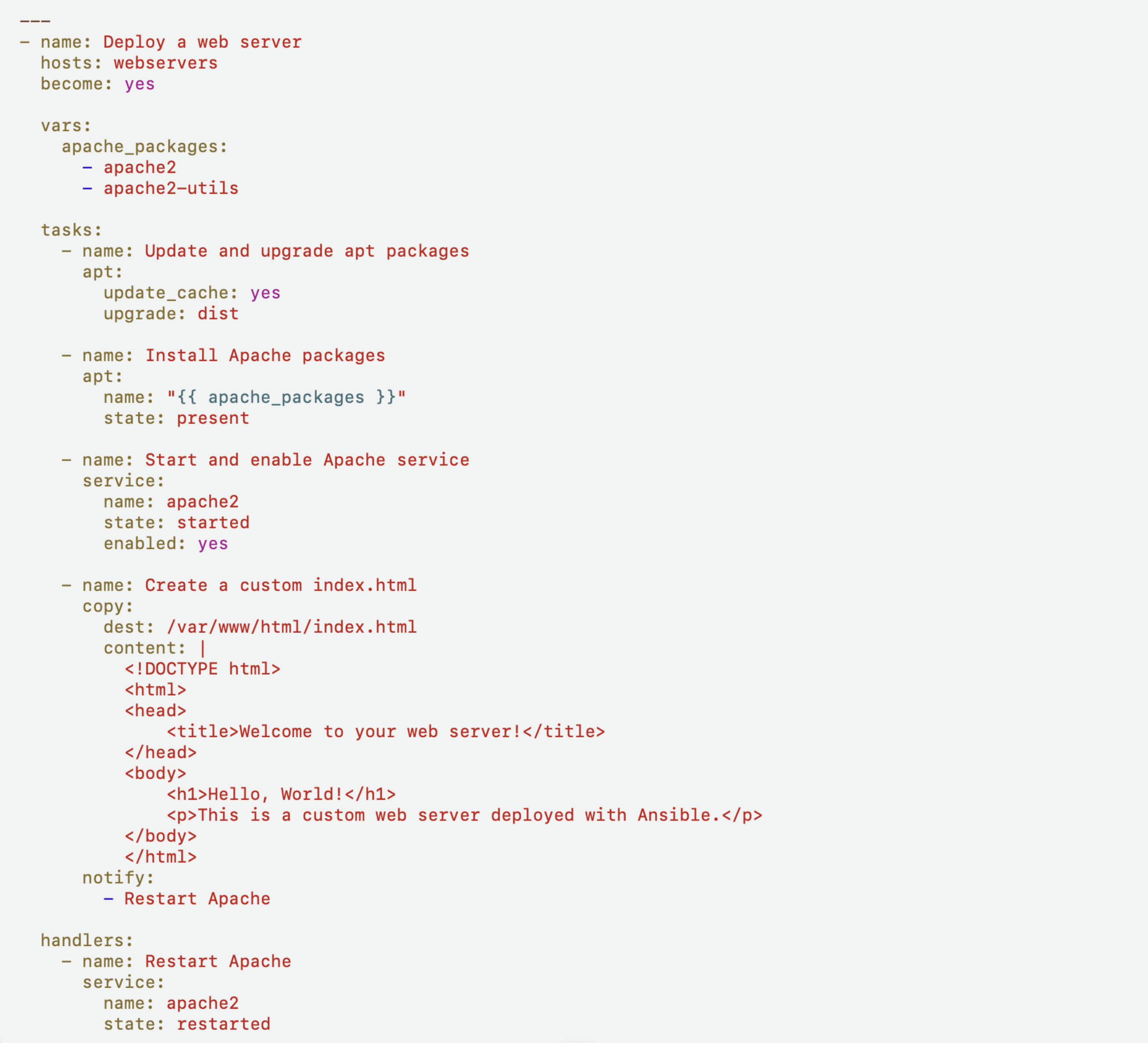
Puppet:

- Declarative language for defining system configurations
- Uses a client-server model
- Suitable for managing large infrastructures

Chef:

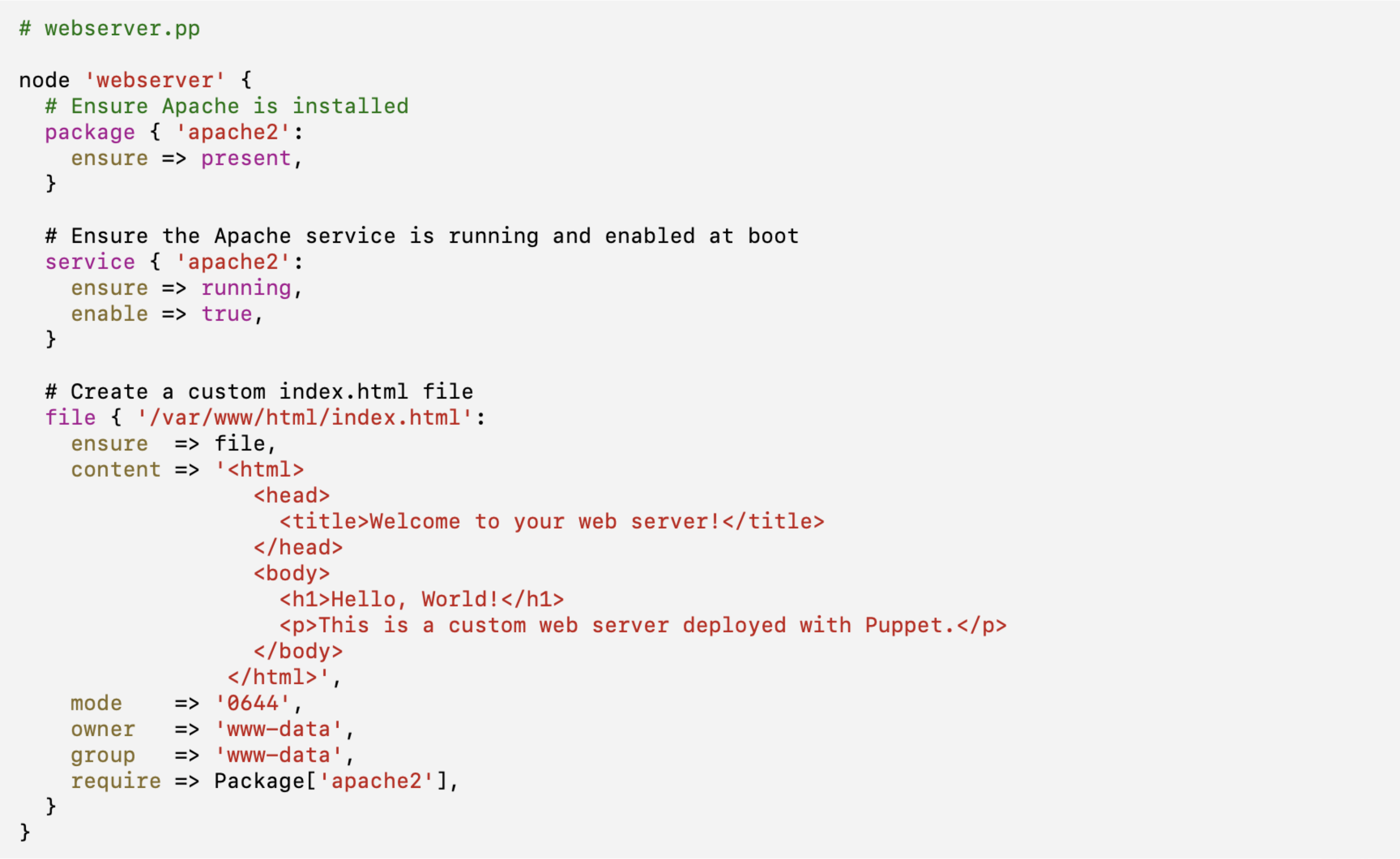
- Uses Ruby-based DSL for writing configuration scripts (cookbooks)
- Follows a client-server architecture
- Focuses on infrastructure as code

Ansible example – Deploy a webserver



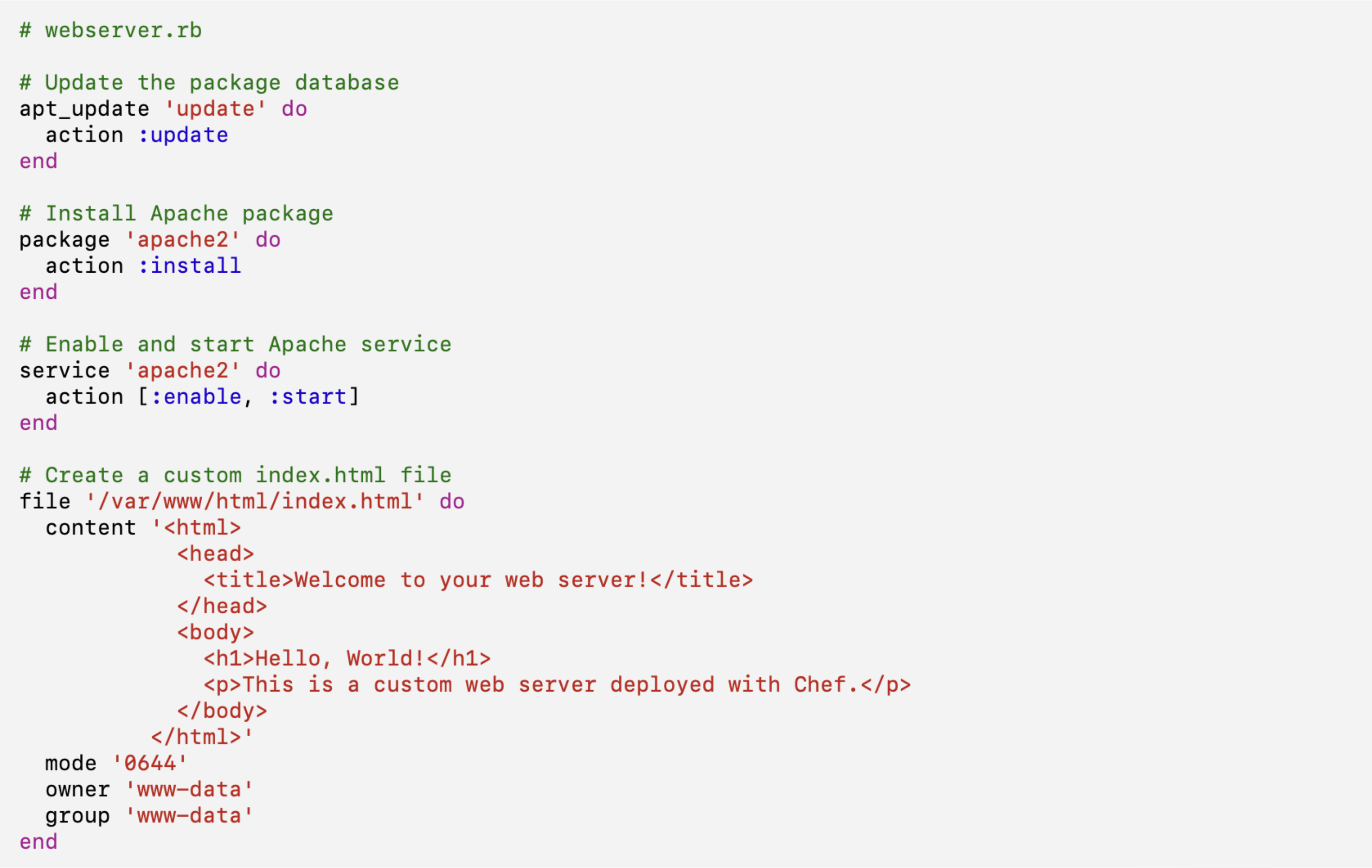
```yaml
---
- name: Deploy a web server
  hosts: webservers
  become: yes

  vars:
    apache_packages:
      - apache2
      - apache2-utils

  tasks:
    - name: Update and upgrade apt packages
      apt:
        update_cache: yes
        upgrade: dist

    - name: Install Apache packages
      apt:
        name: "{{ apache_packages }}"
        state: present

    - name: Start and enable Apache service
      service:
        name: apache2
        state: started
        enabled: yes

    - name: Create a custom index.html
      copy:
        dest: /var/www/html/index.html
        content: |
          <!DOCTYPE html>
          <html>
          <head>
              <title>Welcome to your web server!</title>
          </head>
          <body>
              <h1>Hello, World!</h1>
              <p>This is a custom web server deployed with Ansible.</p>
          </body>
          </html>
      notify:
        - Restart Apache

  handlers:
    - name: Restart Apache
      service:
        name: apache2
        state: restarted
```

Puppet manifest – deploy a webserver



```
# webserver.pp

node 'webserver' {
  # Ensure Apache is installed
  package { 'apache2':
    ensure => present,
  }

  # Ensure the Apache service is running and enabled at boot
  service { 'apache2':
    ensure => running,
    enable => true,
  }

  # Create a custom index.html file
  file { '/var/www/html/index.html':
    ensure  => file,
    content => '<html>
                  <head>
                    <title>Welcome to your web server!</title>
                  </head>
                  <body>
                    <h1>Hello, World!</h1>
                    <p>This is a custom web server deployed with Puppet.</p>
                  </body>
                </html>',
    mode    => '0644',
    owner   => 'www-data',
    group   => 'www-data',
    require => Package['apache2'],
  }
}
```

Chef recipe – deploy a webserver

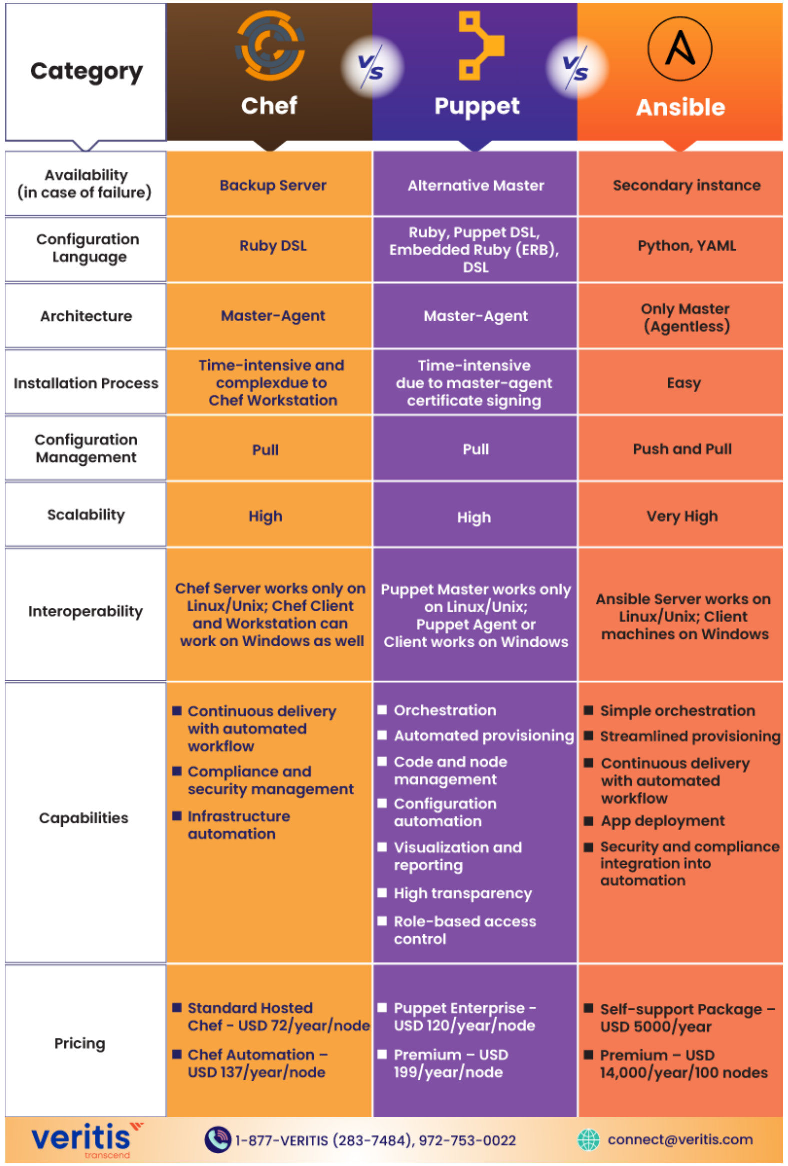


```
# webserver.rb

# Update the package database
apt_update 'update' do
  action :update
end

# Install Apache package
package 'apache2' do
  action :install
end

# Enable and start Apache service
service 'apache2' do
  action [:enable, :start]
end

# Create a custom index.html file
file '/var/www/html/index.html' do
  content '<html>
            <head>
              <title>Welcome to your web server!</title>
            </head>
            <body>
              <h1>Hello, World!</h1>
              <p>This is a custom web server deployed with Chef.</p>
            </body>
          </html>'
  mode '0644'
  owner 'www-data'
  group 'www-data'
end
```

Ansible, Puppet & Chef - Differences

Building Careers Through Education

Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.



Category	Chef	Puppet	Ansible
Availability (in case of failure)	Backup Server	Alternative Master	Secondary instance
Configuration Language	Ruby DSL	Ruby, Puppet DSL, Embedded Ruby (ERB), DSL	Python, YAML
Architecture	Master-Agent	Master-Agent	Only Master (Agentless)
Installation Process	Time-intensive and complexdue to Chef Workstation	Time-intensive due to master-agent certificate signing	Easy
Configuration Management	Pull	Pull	Push and Pull
Scalability	High	High	Very High
Interoperability	Chef Server works only on Linux/Unix; Chef Client and Workstation can work on Windows as well	Puppet Master works only on Linux/Unix; Puppet Agent or Client works on Windows	Ansible Server works on Linux/Unix; Client machines on Windows
Capabilities	■ Continuous delivery with automated workflow ■ Compliance and security management ■ Infrastructure automation	■ Orchestration ■ Automated provisioning ■ Code and node management ■ Configuration automation ■ Visualization and reporting ■ High transparency ■ Role-based access control	■ Simple orchestration ■ Streamlined provisioning ■ Continuous delivery with automated workflow ■ App deployment ■ Security and compliance integration into automation
Pricing	■ Standard Hosted Chef - USD 72/year/node ■ Chef Automation – USD 137/year/node	■ Puppet Enterprise - USD 120/year/node ■ Premium – USD 199/year/node	■ Self-support Package – USD 5000/year ■ Premium – USD 14,000/year/100 nodes

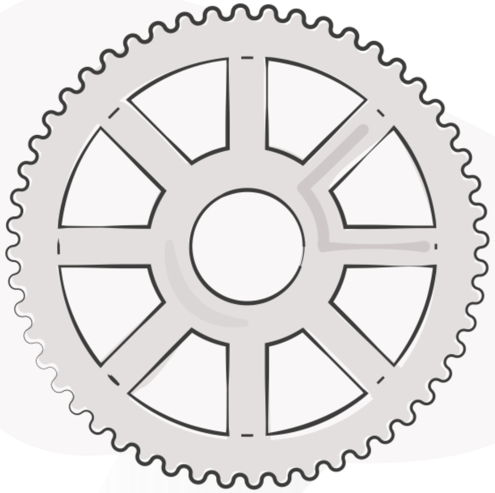
veritis transcend ☎ 1-877-VERITIS (283-7484), 972-753-0022 connect@veritis.com

- Source: https://www.veritis.com/blog/chef-vs-puppet-vs-ansible-comparison-of-devops-management-tools/

BPP

Best practices for automation tools

1. Modular Design
2. Version Control
3. Idempotency
4. Testing
5. Documentation
6. Security
7. Consistency
8. Monitoring and Logging
9. Collaboration
10. Continuous Integration/Continuous Deployment (CI/CD)



Managing and balancing workloads

Key Objectives:

- Automate Deployment: Simplify the process of deploying applications.
- Scale Applications: Dynamically adjust the number of running containers based on demand.
- Manage Resources: Optimise the allocation and use of computing resources.

Benefits of Using Orchestration Tools:

- High Availability: Ensure that applications are always available, even during failures.
- Load Balancing: Distribute traffic evenly across multiple containers to prevent overload.
- Self-Healing: Automatically detect and replace failed containers.
- Flexibility: Support a wide range of applications and workloads.

Popular Orchestration Tools:

- Kubernetes: Advanced orchestration features, suitable for complex, large-scale applications.
- Docker Swarm: Simple setup and management, integrates seamlessly with Docker.
- Apache Mesos: Suitable for large-scale distributed systems and mixed workloads.

Kubernetes overview

Open-source platform for automating deployment, scaling, and operations of application containers

Manages clusters of containers

Key Features:

- **Self-Healing:** Automatically replaces failed containers
- **Horizontal Scaling:** Scales applications up or down based on demand
- **Service Discovery and Load Balancing:** Distributes traffic across containers

Use Cases:

- Managing microservices architectures
- Scaling web applications

.

Docker Swarm Overview

Native clustering and orchestration solution for Docker

Simple setup and integration with Docker tools

Key Features:

- **Declarative Service Model:** Define the desired state of services
- **Multi-Host Networking:** Connects containers across multiple hosts
- **Load Balancing:** Distributes incoming requests across container instances

Use Cases:

- Small to medium-sized deployments
- Simple, straightforward container orchestration

.

Apache Mesos Overview

Cluster manager for large-scale distributed systems

Supports various container formats and workloads

Key Features:

- **Resource Isolation:** Allocates resources dynamically based on needs
- **Multi-Framework Support:** Runs different frameworks (e.g., Marathon, Spark)
- **High Availability:** Ensures continuous operation even if some nodes fail

Use Cases:

- Large-scale data processing
- Mixed workloads and environments

Discussion

Building Careers Through Education
Everybody Matters.
Stronger Together.
Student, Learner and Client Centric.
Trust and Respect.
Embrace Change.

AWS and orchestration

1. **How does AWS handle orchestration?**

Submit your responses to the chat!

BPP

Amazon Elastic Kubernetes Service (EKS)

- Amazon Elastic Kubernetes Service (**Amazon EKS**)
  - Enables you to run Kubernetes on AWS
  - Certified Kubernetes conformant (supports easy migration)
  - Supports Linux and Windows containers
  - Compatible with Kubernetes community tools and supports popular Kubernetes add-ons
- Use Amazon EKS to –
  - Manage clusters of Amazon EC2 compute instances
  - Run containers that are orchestrated by Kubernetes on those instances

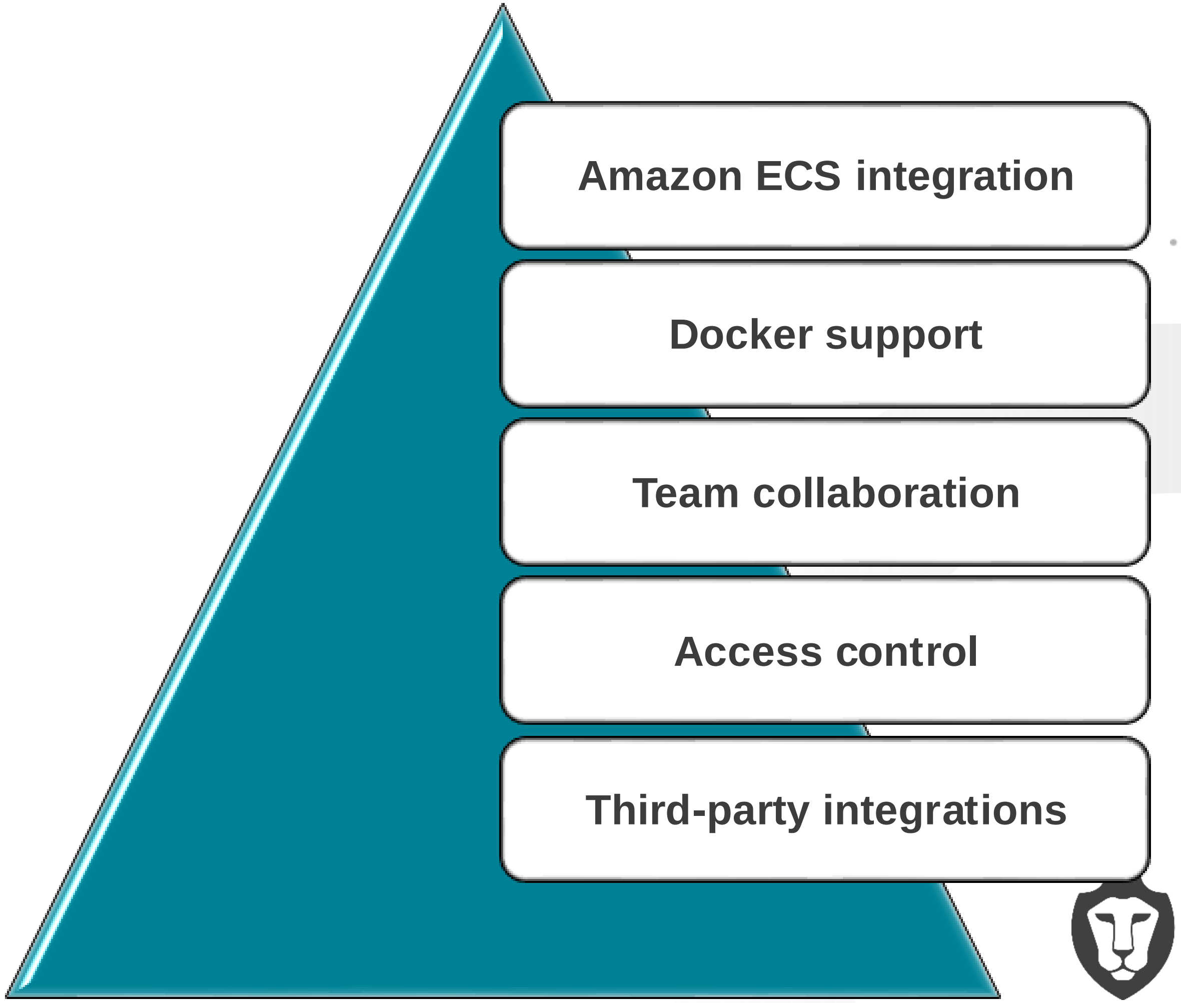
Amazon Elastic Kubernetes Service

Amazon Elastic Container Registry (Amazon ECR)

Amazon ECR is a fully managed Docker container registry that makes it easy for developers to store, manage, and deploy Docker container images.

Amazon Elastic Container Registry

Image Registry



Amazon ECS integration

Docker support

Team collaboration

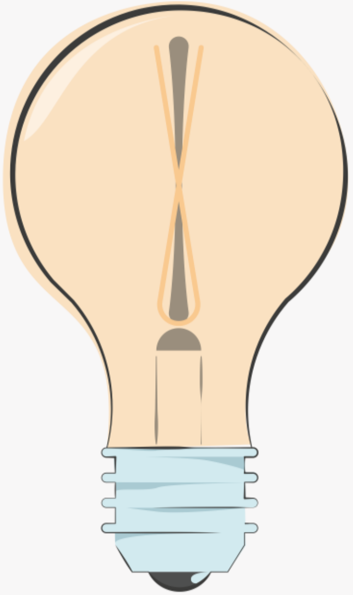
Access control

Third-party integrations

Key takeaways

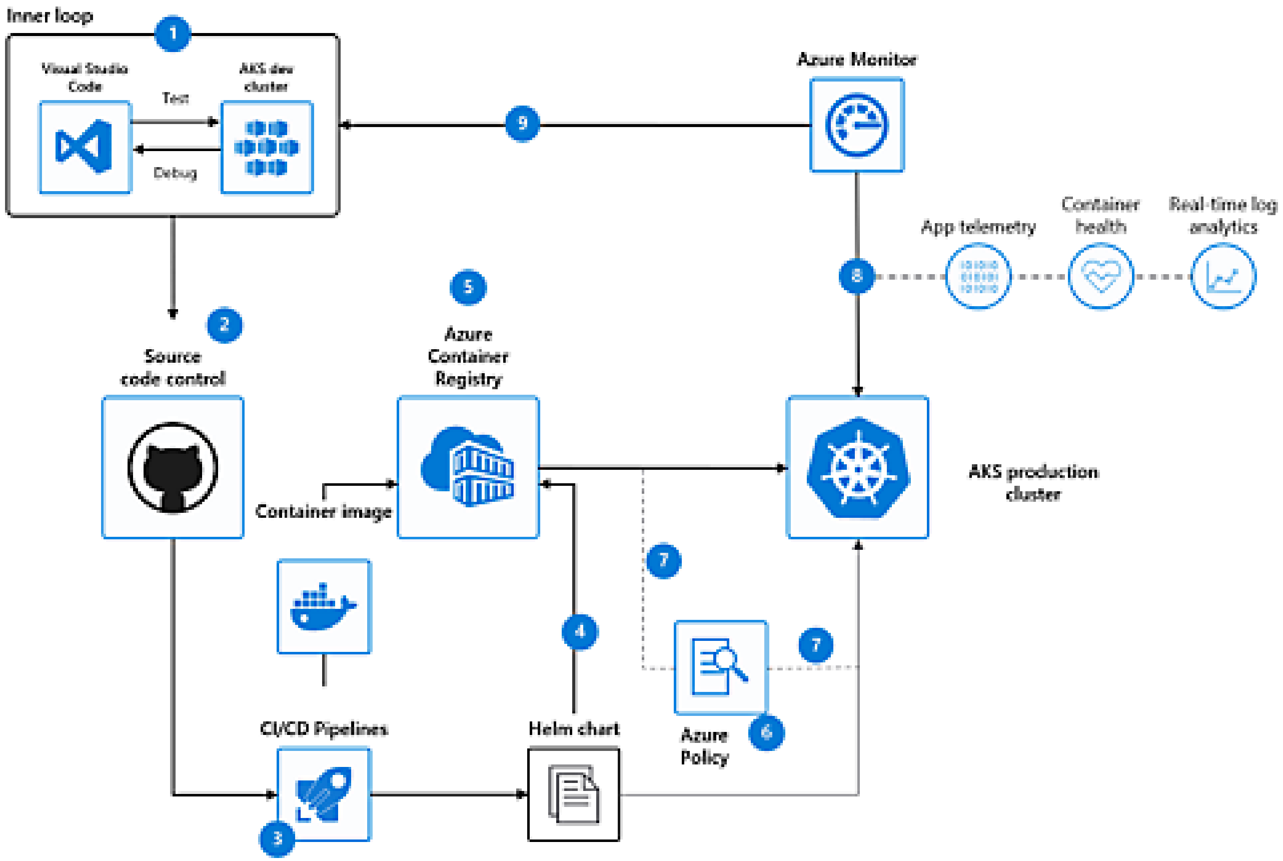
The key takeaways from this section are as follows:

- Containers can hold everything that an application needs to run.
- Docker is a software platform that packages software into containers.
- A single application can span multiple containers.
- Amazon Elastic Container Service (Amazon ECS) orchestrates the running of Docker containers.
- Kubernetes is open-source software for container orchestration.
- Amazon Elastic Kubernetes Service (Amazon EKS) enables you to run Kubernetes on AWS
- Amazon Elastic Container Registry (Amazon ECR) enables you to store, manage, and deploy your Docker containers.



Discussion

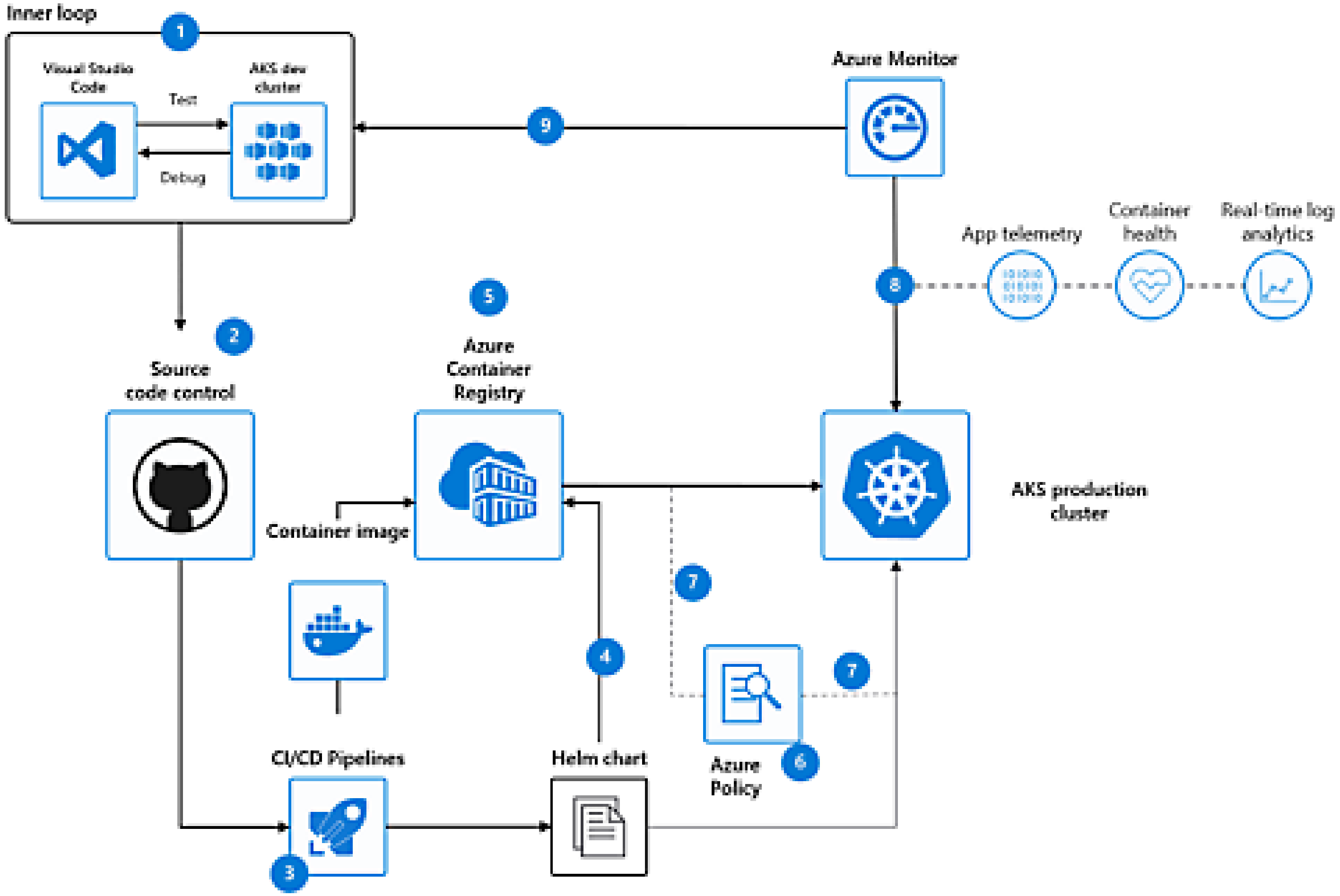
1. How does Azure handle Kubernetes?

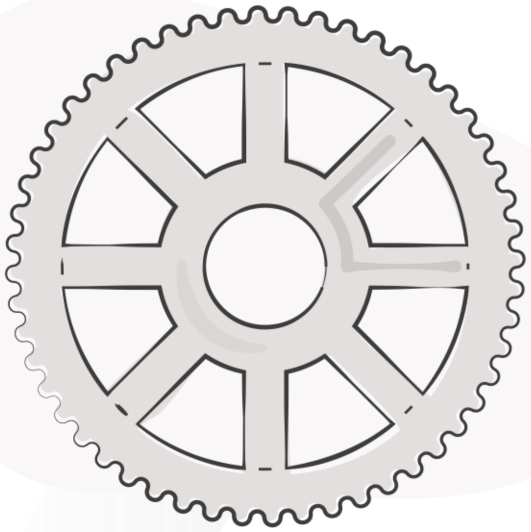


Submit your responses to the chat!

Demo walkthrough

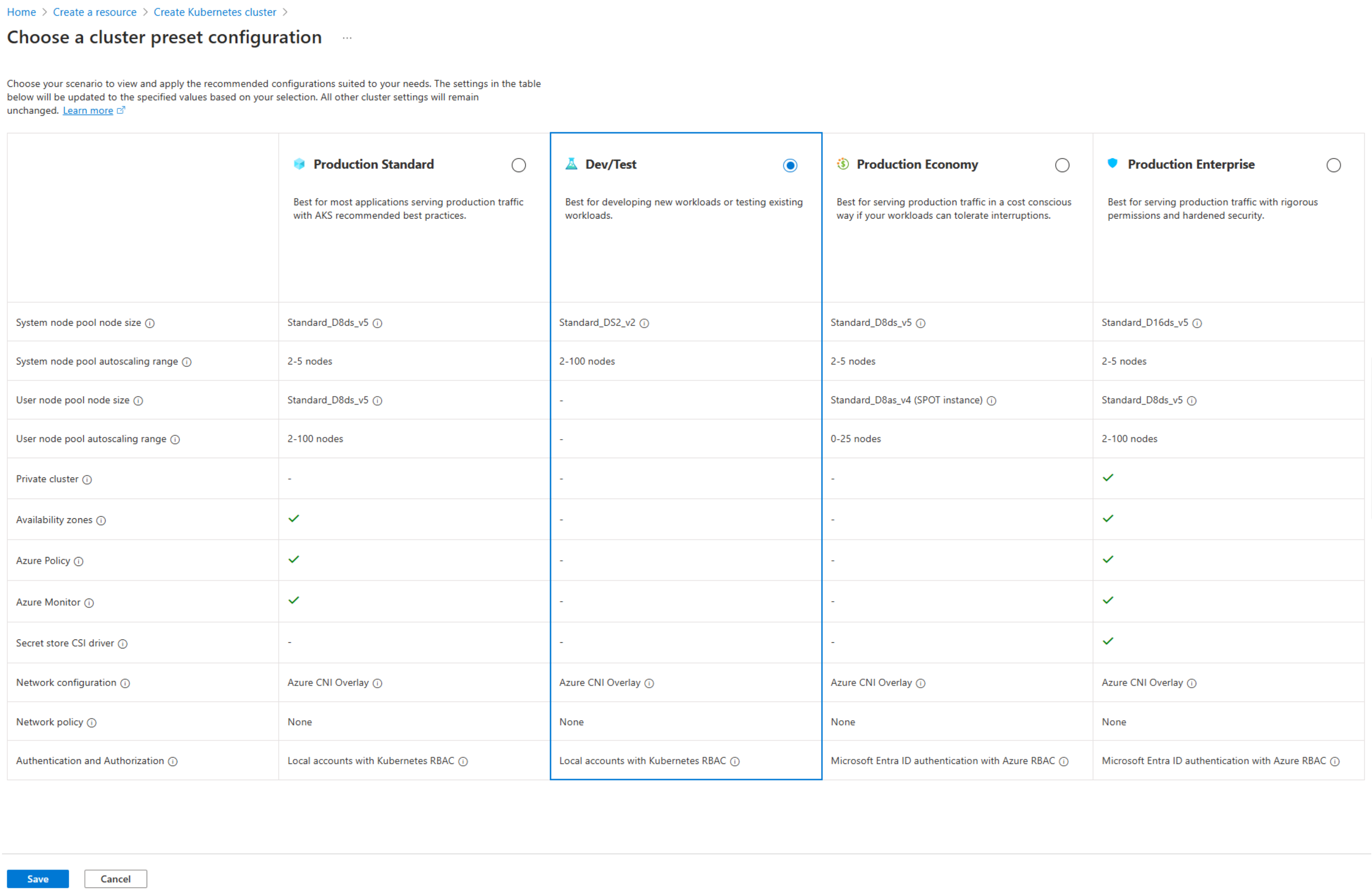
Kubernetes on Azure





Demo walkthrough

Kubernetes on Azure



Home > Create a resource > Create Kubernetes cluster >

Choose a cluster preset configuration ···

Choose your scenario to view and apply the recommended configurations suited to your needs. The settings in the table below will be updated to the specified values based on your selection. All other cluster settings will remain unchanged. Learn more

	Production Standard ○ Best for most applications serving production traffic with AKS recommended best practices.	Dev/Test ● Best for developing new workloads or testing existing workloads.	Production Economy ○ Best for serving production traffic in a cost conscious way if your workloads can tolerate interruptions.	Production Enterprise ○ Best for serving production traffic with rigorous permissions and hardened security.
System node pool node size ⓘ	Standard_D8ds_v5 ⓘ	Standard_DS2_v2 ⓘ	Standard_D8ds_v5 ⓘ	Standard_D16ds_v5 ⓘ
System node pool autoscaling range ⓘ	2-5 nodes	2-100 nodes	2-5 nodes	2-5 nodes
User node pool node size ⓘ	Standard_D8ds_v5 ⓘ	-	Standard_D8as_v4 (SPOT instance) ⓘ	Standard_D8ds_v5 ⓘ
User node pool autoscaling range ⓘ	2-100 nodes	-	0-25 nodes	2-100 nodes
Private cluster ⓘ	-	-	-	✓
Availability zones ⓘ	✓	-	-	✓
Azure Policy ⓘ	✓	-	-	✓
Azure Monitor ⓘ	✓	-	-	✓
Secret store CSI driver ⓘ	-	-	-	✓
Network configuration ⓘ	Azure CNI Overlay ⓘ	Azure CNI Overlay ⓘ	Azure CNI Overlay ⓘ	Azure CNI Overlay ⓘ
Network policy ⓘ	None	None	None	None
Authentication and Authorization ⓘ	Local accounts with Kubernetes RBAC ⓘ	Local accounts with Kubernetes RBAC ⓘ	Microsoft Entra ID authentication with Azure RBAC ⓘ	Microsoft Entra ID authentication with Azure RBAC ⓘ

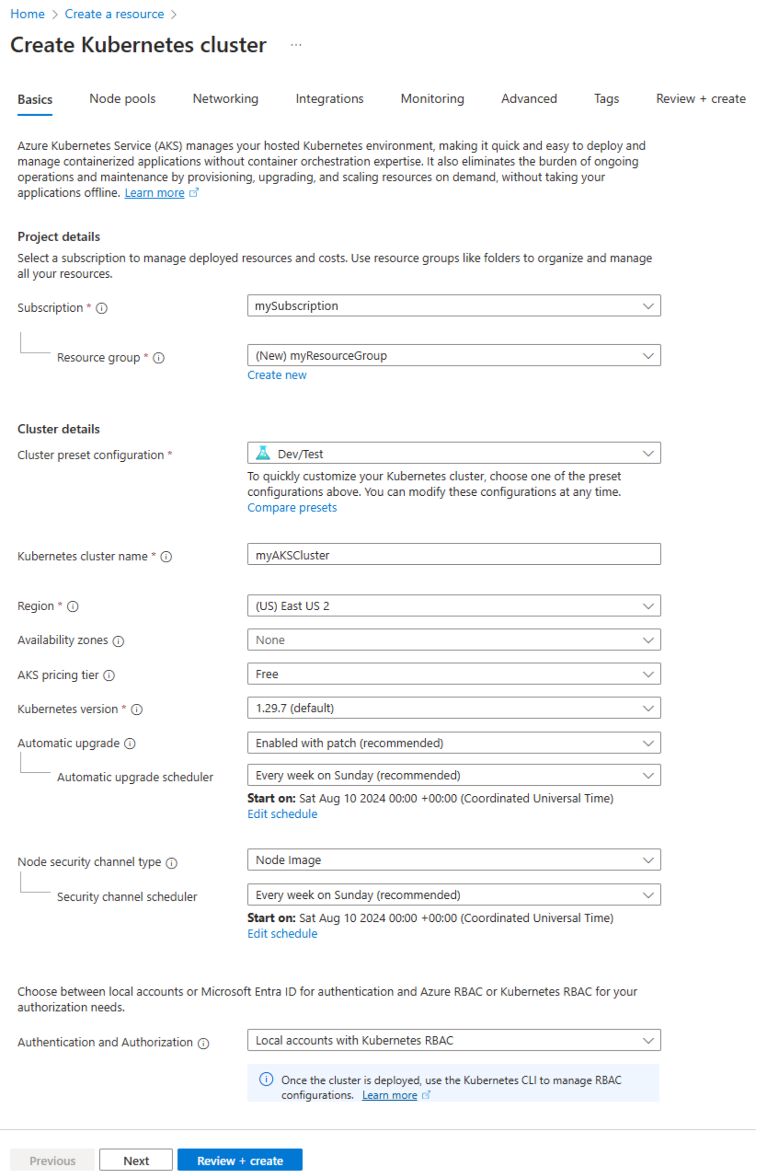
Building Careers Through Education

Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.

BPP

Demo walkthrough

Kubernetes on Azure



Home > Create a resource >

Create Kubernetes cluster ...

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment, making it quick and easy to deploy and manage containerized applications without container orchestration expertise. It also eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand, without taking your applications offline. Learn more

Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ: mySubscription

Resource group * ⓘ: (New) myResourceGroup
Create new

Cluster details

Cluster preset configuration *: Dev/Test
To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.
Compare presets

Kubernetes cluster name * ⓘ: myAKSCluster

Region * ⓘ: (US) East US 2

Availability zones ⓘ: None

AKS pricing tier ⓘ: Free

Kubernetes version * ⓘ: 1.29.7 (default)

Automatic upgrade ⓘ: Enabled with patch (recommended)

Automatic upgrade scheduler: Every week on Sunday (recommended)
Start on: Sat Aug 10 2024 00:00 +00:00 (Coordinated Universal Time)
Edit schedule

Node security channel type ⓘ: Node Image

Security channel scheduler: Every week on Sunday (recommended)
Start on: Sat Aug 10 2024 00:00 +00:00 (Coordinated Universal Time)
Edit schedule

Choose between local accounts or Microsoft Entra ID for authentication and Azure RBAC or Kubernetes RBAC for your authorization needs.

Authentication and Authorization ⓘ: Local accounts with Kubernetes RBAC

ⓘ Once the cluster is deployed, use the Kubernetes CLI to manage RBAC configurations. Learn more

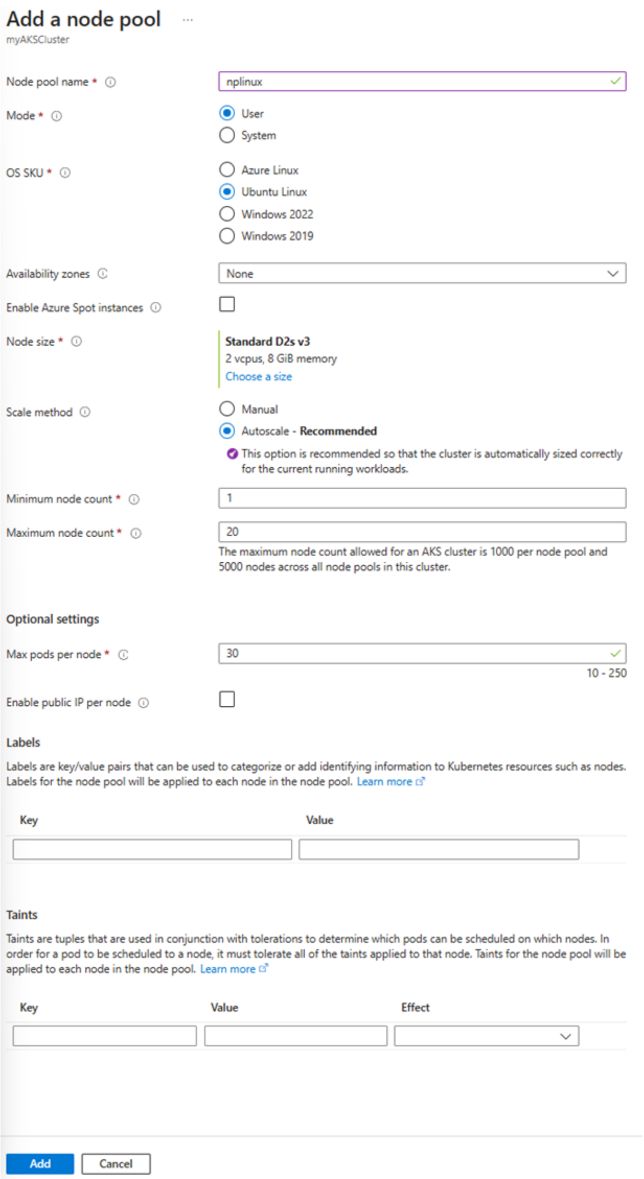
Building Careers Through Education

Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.

BPP

Demo walkthrough

Kubernetes on Azure



Add a node pool ···

myAKSCluster

Node pool name * ⓘ	nplinux ✓
Mode * ⓘ	● User ○ System
OS SKU * ⓘ	○ Azure Linux ● Ubuntu Linux ○ Windows 2022 ○ Windows 2019
Availability zones ⓘ	None
Enable Azure Spot instances ⓘ	□
Node size * ⓘ	**Standard D2s v3** 2 vcpus, 8 GiB memory Choose a size
Scale method ⓘ	○ Manual ● Autoscale - **Recommended** This option is recommended so that the cluster is automatically sized correctly for the current running workloads.
Minimum node count * ⓘ	1
Maximum node count * ⓘ	20 The maximum node count allowed for an AKS cluster is 1000 per node pool and 5000 nodes across all node pools in this cluster.

Optional settings

Max pods per node * ⓘ	30 ✓ 10 - 250
Enable public IP per node ⓘ	□

Labels

Labels are key/value pairs that can be used to categorize or add identifying information to Kubernetes resources such as nodes. Labels for the node pool will be applied to each node in the node pool. Learn more

Key	Value

Taints

Taints are tuples that are used in conjunction with tolerations to determine which pods can be scheduled on which nodes. In order for a pod to be scheduled to a node, it must tolerate all of the taints applied to that node. Taints for the node pool will be applied to each node in the node pool. Learn more

Key	Value	Effect

Add Cancel

Building Careers Through Education

Everybody Matters.

Stronger Together.

Student, Learner and Client Centric.

Trust and Respect.

Embrace Change.

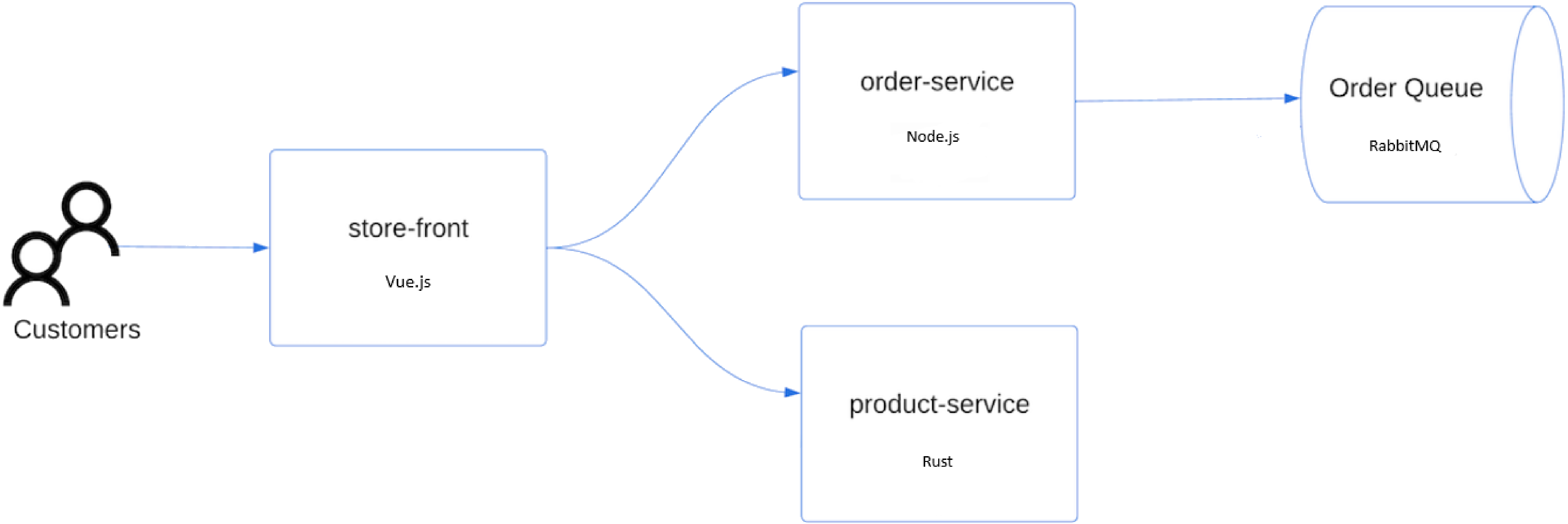
BPP

Demo walkthrough

Kubernetes on Azure

Building Careers Through Education

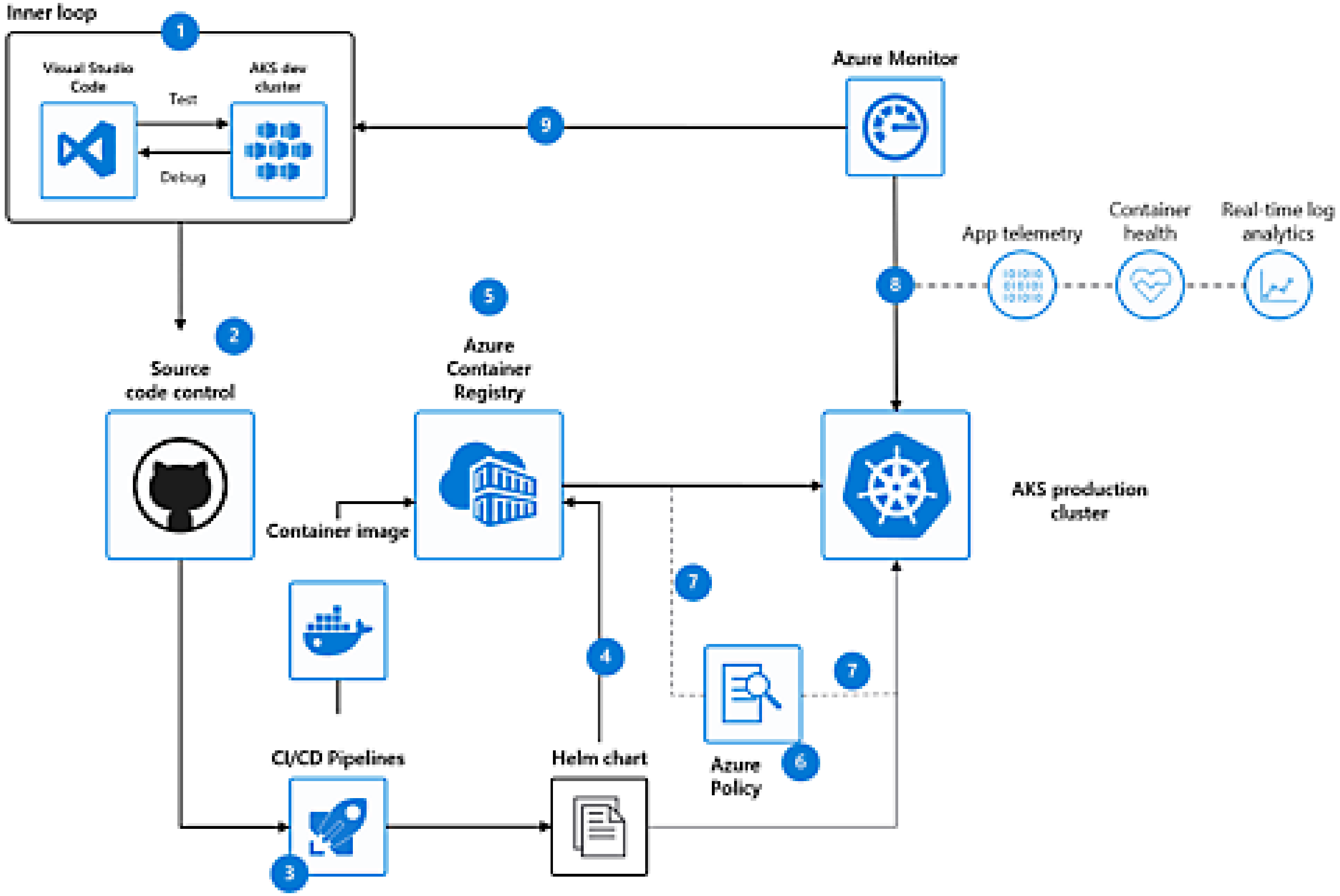
Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.



BPP

Demo walkthrough

The debrief



Building Careers Through Education

Everybody Matters.

Stronger Together.

Student, Learner and Client Centric.

Trust and Respect.

Embrace Change.

BPP

Best practices for container orchestration

Use Health Checks:

- Define health checks to ensure containers are running as expected
- Automatically restart unhealthy containers

Implement Monitoring and Logging:

- Use tools like Prometheus, Grafana, and ELK stack for monitoring and logging
- Monitor resource usage and application performance

Optimise Resource Allocation:

- Define resource limits and requests for containers
- Use node affinity and anti-affinity rules to optimise placement

Ensure Security:

- Implement network policies and secrets management
- Regularly update and patch container images

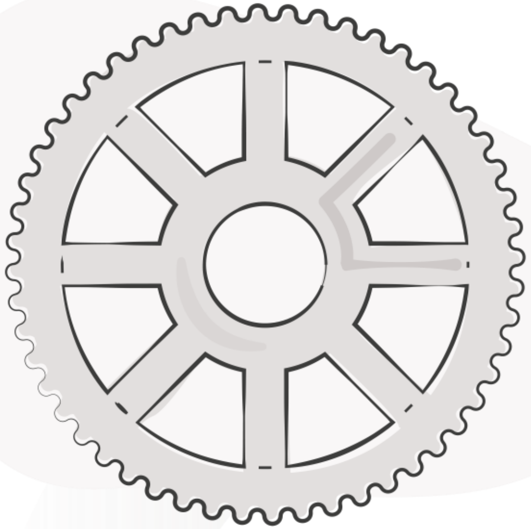
Demo walkthrough

Deploying a Python microservices application using Docker

Deploying a simple Python microservices application using Docker, Kubernetes and serverless deployments

- https://www.eksworkshop.com/

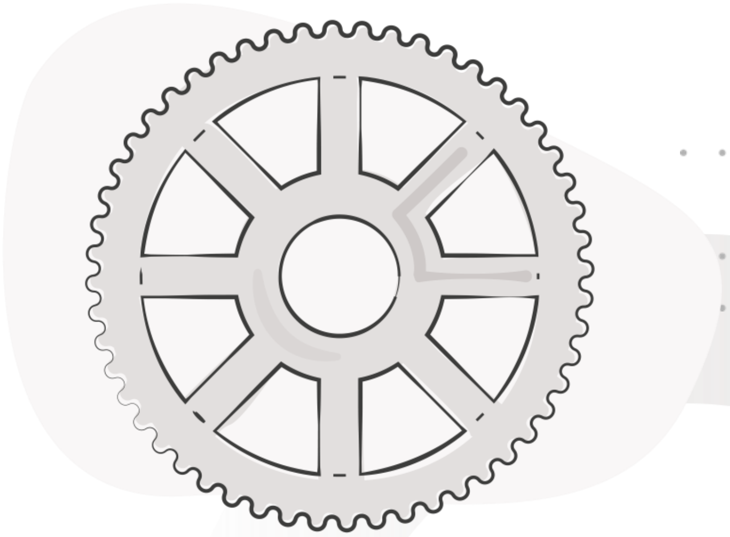
- https://webapp.serverlessworkshops.io/



Post-webinar consolidation

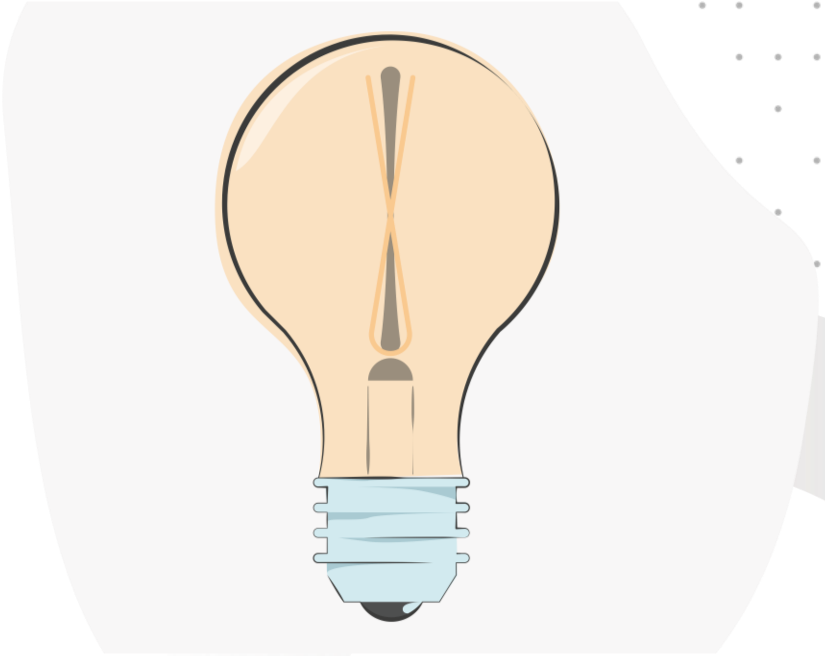
Apply…

- **Task 1:** Deploy a sample microservices application using Docker and Kubernetes
- **Task 2:** Write a reflection on resource optimisation strategies for sustainable growth
- **Task 3:** Reflect on the benefits of containerisation in application deployment in your organisation



Session objective review

- Discuss how cloud automation and orchestration can help streamline the management and maintenance of cloud-based systems and applications;
- Analyse how the use of tools and technologies such as Ansible, Puppet, and Chef may be used to automate routine tasks in order to reduce the potential for human error in cloud-based systems and applications;
- Use tools and technologies such as Kubernetes, Docker Swarm, and Mesos to manage containers and balance workloads to ensure they are working together effectively and high availability exists.



BPP

Thank you

Do you have any questions, comments, or feedback?