About Cognixia

Cognixia- A Digital Workforce Solutions Company is dedicated to delivering exceptional trainings and certifications in digital technologies. Founded in 2014, we provide interactive, customized training courses to individuals and organizations alike, and have served more than 100,000 professionals across 37 countries worldwide.

Our team of more than 4,500 industry experts facilitate more than 400 comprehensive digital technologies courses, along with state-of-the-art infrastructure, to deliver the best learning experience for everyone. Our comprehensive series of instructor-led online trainings, classroom trainings and on-demand self-paced online trainings cover a wide array of specialty areas, including all of the following:

•  IoT
•  Big Data
•  Cloud Computing
•  Cyber Security
•  Machine Learning
•  AI & Deep Learning
•  Blockchain Technologies
•  DevOps

Cognixia is ranked amongst the top five emerging technologies training companies by various prestigious bodies. We’re also an MAPR Advantage Partner, Hortonworks Community Partner, RedHat Enterprise Partner, Microsoft Silver Learning Partner and an authorized training partner for Dell EMC, Pivotal, VMware and RSA technologies.
OUR AWARDS & AFFILIATIONS

Best Workplace Amongst Emerging Enterprises Awarded By Great Indian Workplace Awards – 2018 Mumbai, India

Best Training Provider of the Year Awarded By The Golden Globe Tiger Awards – 2018 Kuala Lumpur, Malaysia

Top 5 Big Data Training Institutes awarded By siliconindia

World HRD Congress has awarded Cognixia for Excellence in Training (Asia)

Top 10 IoT Training Companies for the Second consecutive year.

Top 10 IoT Training Institutes – 2017 HIGHER EDUCATION Review

Asian Training & Leadership Award, Dubai

ISO 9001:2015 Certified Quality Management System


Top 10 Linkedin Best of Pages, 2019

Cognixia is awarded as Training Company of the Year, 2018

Top 10 IoT Training Institutes – 2016 Analytics India Magazine

ISO 9001:2015 Certified Quality Management System


www.cognixia.com
AUTHORIZED TRAINING PARTNERS FOR

Red Hat

ITIL®

(ISC)²

Silver Partner

Microsoft

Automation Anywhere
Who should enroll for this bootcamp?

Kubernetes and Docker have become hugely popular among developers, especially in the DevOps world. The course would be very useful for professionals interested in learning more container orchestration and management. Microservices architecture and containers are immensely popular technologies in the market today, and this bootcamp will equip you with the right skills to build a successful career in the field.

Prerequisites

To enroll for this course, participants need to have:

- Basic command knowledge of Linux
- Basic understanding of DevOps
- Basic knowledge of YAML programming language (beneficial, not mandatory)

About Docker and Kubernetes

Kubernetes is an open-source orchestration system for automating management, placement, scaling and routing of containers. It provides an API to control how and where the containers would run. Docker is also an open-source container-file format for automating the deployment of applications as portable, self-sufficient containers that can run in the cloud or on-premises. Together, Kubernetes and Docker have become hugely popular among developers, especially in the DevOps world.

Both Docker and Kubernetes are huge open-source technologies, largely written in the Go programming language, that use human-readable YAML files to specify application stacks and their deployment.
Latest Industry Trends

- In 2019, 87% of the companies surveyed by the Cloud Native Computing Foundation reported using container orchestration solutions, and 40% were using Kubernetes.

- The application container market is expected to grow at a CAGR of 29% during 2020-2025.

- One of the world’s largest retail electronic payments networks – Visa, refactored two of their critical payment processing applications using Docker enterprise container platform. In six months, they achieved over 10x increase in scalability.

- Kubernetes is now nearly as popular in containerized environments on Azure as it is on GCP. Among organizations running containers in AWS, popularity of Kubernetes has doubled to over 45% in the past two years.
Course features

- 32 hours of online training with a live point of contact and hands-on assignments
- Interactive, online, live session with an industry expert trainer
- Dedicated technical team to answer your queries at anytime
- Best price guarantee, aligning with the quality of our course deliverables

Course objectives

The course will help participants understand:

- Fundamentals of Docker
- Fundamentals of Kubernetes
- Running Kubernetes instances on Minikube
- Creating and working with Kubernetes clusters
- Working with resources
- Creating and modifying workloads
- Working with Kubernetes API and key metadata
- Working with specialized workloads
- Scaling deployments and application security
- Understanding the container ecosystem
Docker Essentials

- Docker Introduction
- Docker Architecture
- Docker Installation on RedHat and Ubuntu OS
- Working with Images (Docker hub, docker Registry)
- Working with containers
- Container Networking.
- Working with Volumes and Persistent data
- Managing Container App using Docker Swarm
- Overview of Docker Enterprise Tool.

Kubernetes

Minikube

- Using Kubernetes Without Installation
  - Installing the Kubernetes CLI, kubectl
  - Installing Minikube to Run a Local Kubernetes Instance
  - Using Minikube Locally for Development
  - Starting Your First Application on Minikube
  - Accessing the Dashboard in Minikube

Kubernetes Cluster

- Installing kubeadm to Create a Kubernetes Cluster
- Bootstrapping a Kubernetes Cluster Using kubeadm
- Downloading a Kubernetes Release from GitHub
- Downloading Client and Server Binaries
- Using a hyperkube Image to Run a Kubernetes Master Node with Docker
- Writing a systemd Unit File to Run Kubernetes Components
- Creating a Kubernetes Cluster on Google Kubernetes Engine (GKE)
- Creating a Kubernetes Cluster on Azure Container Service (ACS)

Kubernetes Client

- Listing Resources
- Deleting Resources
- Watching Resource Changes with kubectl
- Editing Resources with kubectl
- Asking kubectl to Explain Resources and Fields

Creating and Modifying Workloads

- Creating a Deployment Using kubectl run
- Creating Objects from File Manifests
- Writing a Pod Manifest from Scratch
- Launching a Deployment Using a Manifest
- Updating a Deployment
Managing Pods with Kubernetes
- Replication Controller
- Replica Set
- ConfigMap
- Deployment
- Secrets
- DaemonSet
- Jobs

Services
- Creating a Service to Expose Your Application
- Verifying the DNS Entry of a Service
- Changing the Type of a Service
- Deploying an Ingress Controller on Minikube
- Making Services Accessible from Outside the Cluster

Exploring the Kubernetes API and Key Metadata
- Discovering API Endpoints of the Kubernetes API Server
- Understanding the Structure of a Kubernetes Manifest
- Creating Namespaces to Avoid Name Collisions
- Setting Quotas Within a Namespace
- Labeling an Object
- Using Labels for Queries
- Annotating a Resource with One Command

Managing Specialized Workloads
- Running a Batch Job
- Running a Task on a Schedule Within a Pod
- Running Infrastructure Daemons per Node
- Managing Stateful and Leader/Follower Apps
- Influencing Pods’ Startup Behavior

Volumes and Configuration Data
- Exchanging Data Between Containers via a Local Volume
- Passing an API Access Key to a Pod Using a Secret
- Providing Configuration Data to an Application
- Using a Persistent Volume with Minikube
- Understanding Data Persistency on Minikube
- Dynamically Provisioning Persistent Storage on GKE

Scaling
- Scaling a Deployment
- Automatically Resizing a Cluster in GKE
- Automatically Resizing a Cluster in AWS
- Using Horizontal Pod Autoscaling on GKE

Security
- Providing a Unique Identity for an Application
- Listing and Viewing Access Control Information
- Controlling Access to Resources
- Securing Pods
Detailed Curriculum: Modules

Monitoring and Logging
- Accessing the Logs of a Container
- Recover from a Broken State with a Liveness Probe
- Controlling Traffic Flow to a Pod Using a Readiness Probe
- Adding Liveness and Readiness Probes to Your Deployments
- Enabling Heapster on Minikube to Monitor Resources
- Using Prometheus on Minikube
- Using Elasticsearch–Fluentd–Kibana (EFK) on Minikube

Maintenance and Troubleshooting
- Enabling Autocomplete for kubectl
- Removing a Pod from a Service
- Accessing a ClusterIP Service Outside the Cluster
- Understanding and Parsing Resource Statuses
- Debugging Pods
- Getting a Detailed Snapshot of the Cluster State
- Adding Kubernetes Worker Nodes
- Draining Kubernetes Nodes for Maintenance
- Managing etcd

Developing Kubernetes
- Compiling from Source
- Compiling a Specific Component
- Using a Python Client to Interact with the Kubernetes API
- Extending the API Using Custom Resource Definitions (CRDs)

The Ecosystem
- Installing Helm, the Kubernetes Package Manager
- Using Helm to Install Applications
- Creating Your Own Chart to Package Your Application with Helm
- Converting Your Docker Compose Files to Kubernetes Manifests
- Creating a Kubernetes Cluster with kubicorn
- Storing Encrypted Secrets in Version Control
- Deploying Functions with kubeless
Capstone project for Docker and Kubernetes

**Project: Setting up the infrastructure for automating a CI/CD pipeline using Jenkins**

**Synopsis:**
This project would help the participant learn how to set up the required infrastructure for automating the CI/CD pipeline. The applications would be deployed on Docker containers. The front-end applications would be written in php, while the back-end applications would be written in MySQL.

Infrastructure requirements for the project:
- The developers would need to set up the development environment, and commit the code to GitHub. The code would also need to be automatically built and tested.
- Testing environment would test the build code as well as the integrated application.
- A UAT environment would need to be set up, which would be the same as the production environment. Here, the client would be able to perform the testing.
- The production environment for these applications would need to be set up using Docker and Kubernetes. Each node in the cluster would need to have anti-virus installed.
- To ensure zero downtime, the rolling update would need to be installed using the blue/green process.
- A backup process would need to be set up for the DB log files.
- Multiple masters would need to be created in the clusters for ensuring fault-tolerance.
- Applications need to be load balanced.
- An easy rollback of the applications to their previous versions should be possible, when required.
- Applications would be verified on the basis of their smooth functioning without any issues during production.

**Tools to be used in the project:**
- GitHub
- Maven
- Docker
- Kubernetes
- Jenkins
Cognixia USPs

- Lifetime LMS Access
- 24 x 7 Support
- Real-life Projects & Case Studies
- Industry Experts as Trainers
- Industry Standard Certificate
POTENTIAL CAREER OPTIONS

- DevOps developers
- DevOps engineers
- Java developers
- C#/.Net developers
- Software engineers
- Backend developers
- IoT architects
- QA engineers
Docker and Kubernetes Bootcamp