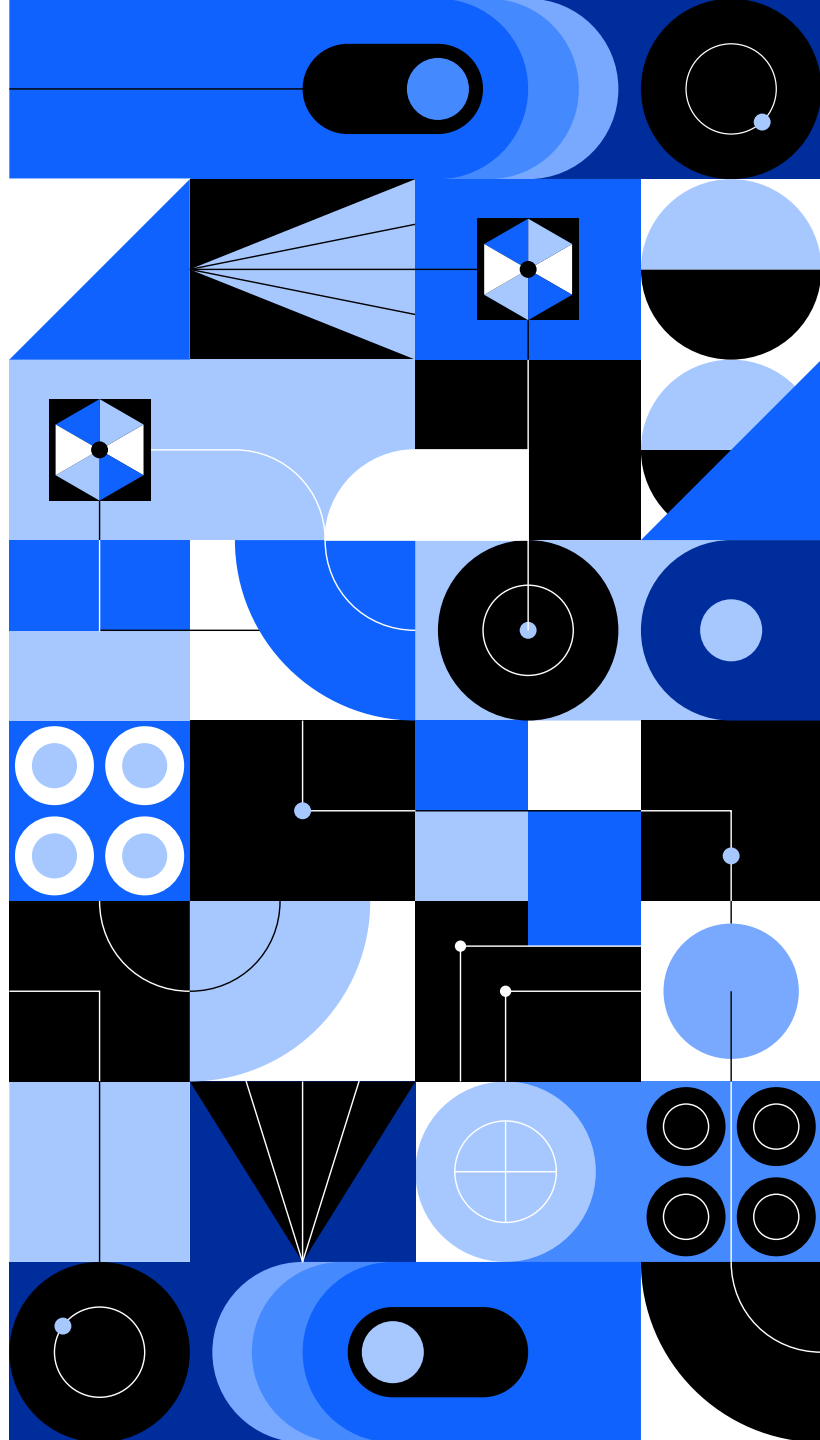
IBM Accelerate Software Developer Track

Wednesdays June 5th – July 24th
6:00pm – 8:00pm Eastern Time



• Optional practice test

Wednesday, July 24, 2024

Cloud Native Development Introduction

Welcome & Program Highlights: Sujeily P. Fonseca-Gonzalez

Sujeily P. Fonseca
Software Engineering Manager, Technical Leader, MultiCloud Saas Platform
SW Track Lead

Speakers: Shikha Srivastava, Snehalata Hegde

Shikha Srivatsava
Distinguished Engineer, IBM SWG Multicloud Saas Platform, Master Inventor IBM Infrastructure

Snehalata Hegde
Lead Software Engineer, IBM Cloud Pak for AIOps, IBM Software

Closure: Alan Bivens

Alan Bivens
VP, IBM MultiCloud Platform
SW Track Executive Sponsor

IBM Accelerate Software: Talent Identification Program
2024 Software Track Learning Curriculum

Executive Sponsors: Alan Bivens and Tim Humphrey; Track Lead: Sujeily Fonseca

June 5 – Week 1

Frontend Technology Intro

Presenters:
Multiple

Foundational Skills Topic:
Corporate Communications

June 12 – Week 2

JavaScript, React, and Styling

Presenters:
Multiple

Foundational Skills Topic:
Growth Behaviors

June 19 – Week 3

Event Handling Synchronicity & Testing

Presenters:
Multiple

Foundational Skills Topic:
Job Applications and Resumes

June 26 – Week 4

Client-Side versus Server-Side, Backend Intro, and Web App Security

Presenters:
Multiple

Foundational Skills Topic:
Personal Branding & LinkedIn Profile Refinement

July 3 – Week 5

Design Thinking

Presenters:
Multiple

Foundational Skills Topic:
Interviewing

July 10 – Week 6

Functional Backend Hosts, SDKs, and Design Patterns

Presenters:
Multiple

Foundational Skills Topic:
Bringing Your Whole Self to Work

July 17 – Week 7

Security & Compliance

Presenters:
Multiple

Foundational Skills Topic:
Ask Me Anything

July 24 – Week 8

Cloud Native Development

Presenters:
Multiple

Foundational Skills Topic:
Graduation Celebration

IBM Accelerate Software Recap

Software Development

Executive Sponsors

Track Leader

College Sophomores

IBM Instructors

IBM Coaches

IBM Coordinators

Software Development Session Structure –

Wednesdays, 6-8 PM Eastern

Pre-Session

- Up to 30 minutes of material to introduce that week's topic
- Specific tools needed for week
- Posted Monday before session

Lecture

- 1st hour of session
- Webex lecture with slides, demos
- Q&A ongoing via slack channel

Breakout & Lab

- 2nd hour of session
- Breakout rooms of 10 students with 2 coaches
- Practice exercise (lab) in GitHub

Post- session

- Attendance and NPS survey
- Optional practice test
- Up to 30 minutes of additional optional materials
- Lab solutions posted following week

Office Hour

- 1 hour
- Thursday (day after lecture)
- Optional session
- Hosted by that week's instructors

Slack Channel Q&A

- Ongoing Q&A
- Via slack channel
- Monitored by instructors & coaches

HackerRank Test

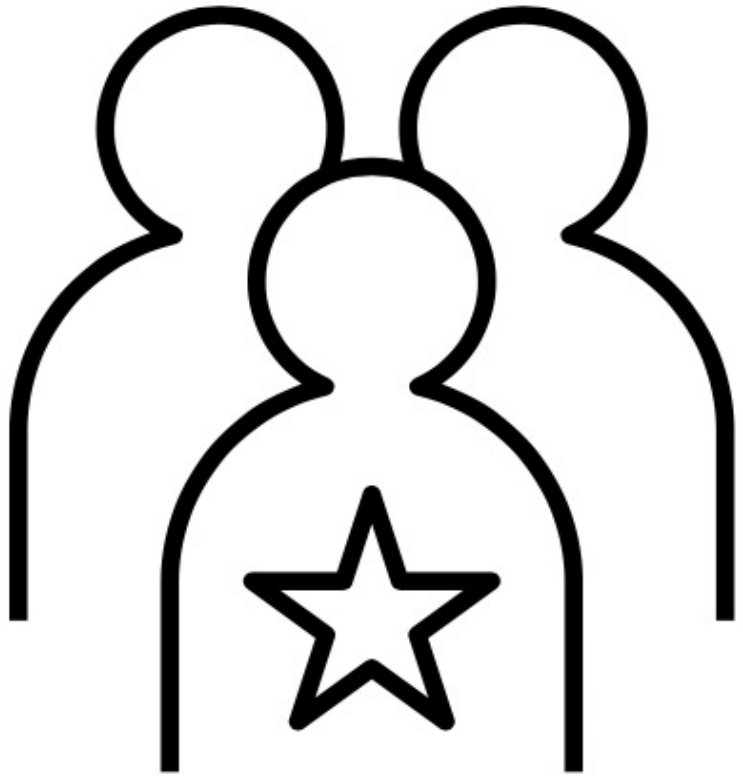
- Multiple choice
- 15-20 minutes
- After sessions 3, 6, 7, 8
- Must pass 3 of 4 to earn Badge
- Available after Thursday Office Hours
- Due the following Tuesday

Coding Practice Test

- Optional
- Given at end of Program
- Coding Test practice to prepare for Intern Application Assessment

Track Stars

– Peer-to-peer recognition



Ibrahim Shah
Brayden Nguyen
Christopher Kim

IBM

Thank you!

We couldn’t do it without you!

- 2 Executive Sponsors
- 1 Software Program Leader
- 18 Technical Leaders/Instructors
- 1 Instructor Coordinator
- 20 Coaches
- 1 Coach Coordinator
- 2 Coordinators
- 133 students across the U.S.

We achieved our goals together!

- Education around relevant Software topics and technologies.
- Increase student engagement and understanding through exposure to industry best practices and real-world scenarios.
- Enhance the program's reputation and effectiveness in nurturing future technical talent.
- Provide students with a more comprehensive understanding of relevant Software Development topics.
- Foster stronger networking connections between students and industry professionals, further aiding career development.

• Optional practice test

Wednesday, July 24, 2024

Cloud Native Development Introduction

Welcome & Program Highlights: Sujeily P. Fonseca-Gonzalez

Sujeily P. Fonseca
Software Engineering Manager, Technical Leader, MultiCloud Saas Platform
SW Track Lead

Speakers: Shikha Srivastava, Snehalata Hegde

Shikha Srivatsava
Distinguished Engineer, IBM SWG Multicloud Saas Platform, Master Inventor IBM Infrastructure

Snehalata Hegde
Lead Software Engineer, IBM Cloud Pak for AIOps, IBM Software

Closure: Alan Bivens

Alan Bivens
VP, IBM MultiCloud Platform
SW Track Executive Sponsor

Session Agenda

1. Cloud
2. Types of cloud Deployment
3. IBM Cloud
4. CloudNative development
5. Lab – Deploy to-do-list application on IBM cloud

01

Cloud

Cloud Native Development and Cloud

What is Cloud

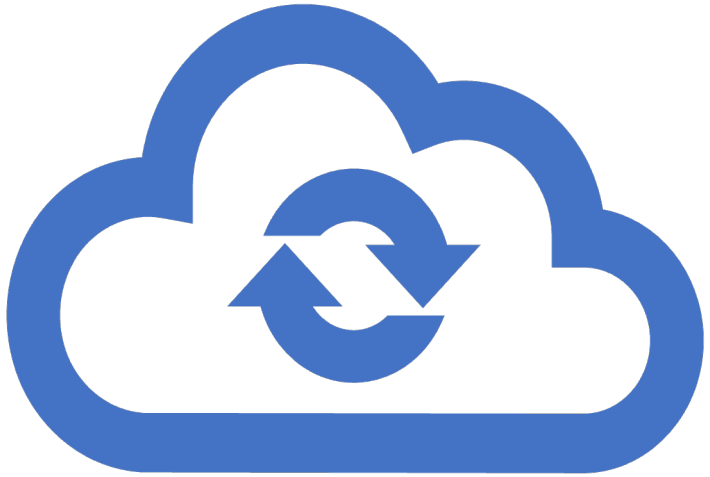
Why is Cloud important

What is Cloud-native development

Cloud-native development –vs- Regular development

What is Cloud?

Cloud computing describes the on-demand delivery of computing resources over the internet



Software
Databases
Servers
Storage

Why is Cloud important?

Offers much more flexibility and is quicker and easier to set up

You can use these resources without installing and maintaining them in your local datacenter

02

Types of cloud deployment

Types of Cloud deployment options

Public Cloud

Public cloud provides developers with the ability to provision resources on demand and only pay for what you use

Public cloud architectures are multi-tenant environments which allows customers to share computing resources.

Private Cloud

A cloud computing environment dedicated to a single customer

Combines many of the benefits of cloud computing with the security and control of on-premises IT infrastructure

Is a *single-tenant* environment, meaning all resources are accessible to one customer only

Hybrid Cloud

IT infrastructure that connects at least one public cloud and at least one private cloud, and provides orchestration, management and application portability between the two

Combines and unifies public cloud and private cloud services from multiple cloud vendors to create a single, flexible, cost-optimal IT infrastructure

03

IBM Cloud

IBM Cloud

IBM Cloud (*offers the most open and secure public cloud platform for business*)	IBM Red Hat OpenShift (ROKS) (*a fast and secure way to containerize and deploy enterprise workloads in Kubernetes clusters*)
• https://www.ibm.com/cloud/free • https://cloud.ibm.com/registration	• Ease of Deployment • Managed Environment

04

CloudNative Development

What is Cloud native development?

Cloud native development means build once, iterate rapidly and deploy anywhere

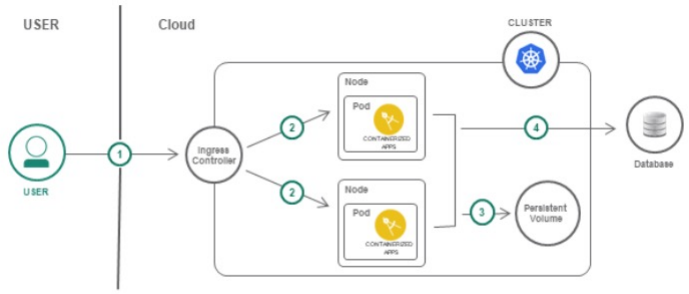
Cloud native services help you build for continuous innovation

A cloud native application consists of discrete, reusable components known as microservices that are designed to integrate into any cloud environment.

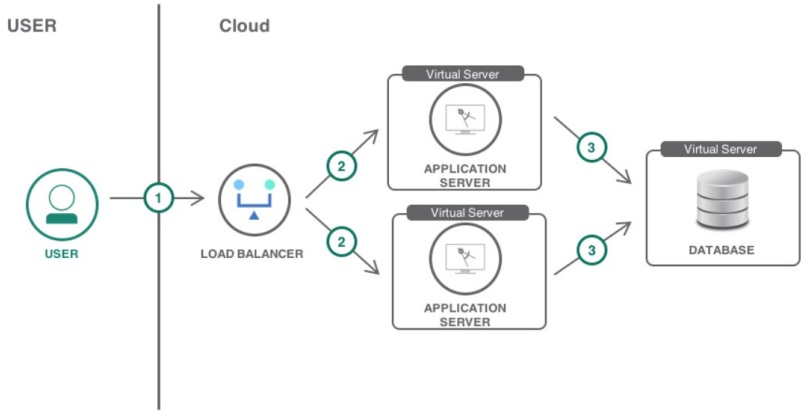
Cloud native applications often have quite specific functions

Cloud-native development –vs- Regular development

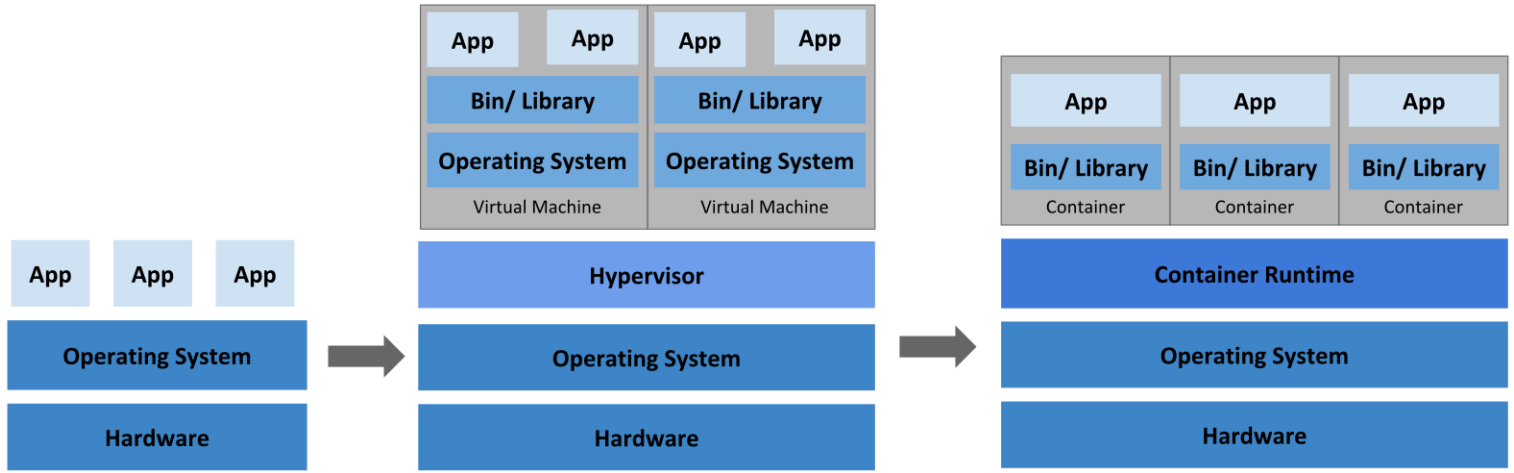
- Developers design cloud native applications to be scalable, platform agnostic, and comprised of microservices
- Rely on microservices, continuous integration and continuous delivery (CI/CD) and can be used via any cloud platform



Traditional applications are developed for deployment in a traditional data center; however, can later be changed so that it also could run in a cloud environment.



Evolution of application deployment



Traditional deployment : Applications are run on physical servers.

Virtualized deployment : Applications are run on a virtual machine.

Container deployment : Applications are run on containers.

What is Serverless development?

Serverless is a cloud-native development model that allows developers to build and run applications without having to manage servers.

Servers in serverless are abstracted away from app development.

A cloud provider handles the routine work of provisioning, maintaining, and scaling the server infrastructure. Developers can simply package their code in containers for deployment.

What are cloud providers?

Cloud service providers are companies that establish public clouds, manage private clouds or offer on-demand cloud computing components/cloud computing services like Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service(SaaS). Using a cloud service provider is a helpful way to access computing services such as

- Infrastructure: The foundation of every computing environment. This infrastructure could include networks, database services, data management, data storage servers etc.
- Platforms: The tools needed to create and deploy applications. These platforms could include operating systems like Linux, middleware, and runtime environments.
- Software: Ready-to-use applications. This software could be custom or standard applications provided by independent service providers.

Cloud service offerings

IaaS, PaaS and SaaS are the three most popular types of cloud service offerings.

- **IaaS**, or infrastructure as a service, is on-demand access to cloud-hosted physical and virtual servers, storage and networking - the backend IT infrastructure for running applications and workloads in the cloud.
- **PaaS**, or platform as a service, is on-demand access to a complete, ready-to-use, cloud-hosted platform for developing, running, maintaining and managing applications. The cloud services provider hosts, manages and maintains all the hardware and software included in the platform.
- **SaaS**, or software as a service, is cloud-hosted, ready-to-use application software. Users pay a monthly or annual fee to use a complete application from within a web browser, desktop client or mobile app. The application and all of the infrastructure required to deliver it - servers, storage, networking, middleware, application software, data storage - are hosted and managed by the SaaS vendor.

Cloud Native Deployment orchestration

CloudNative deployment and Orchestration is deploying and managing platforms and cloud-native applications on cloud infrastructure.

This involves procedural, or script-based tools which are most commonly a simple set of commands used to automatically deploy, configure, and manage networks, servers, and applications.

This approach allows to create and reproduce the exact same platform or application over and over again. This can also be done across multiple cloud providers using one single orchestration tool.

Cloud orchestration tools

There is a wide variety of cloud orchestration tools available for deploying and managing cloud infrastructure.

- Infrastructure-as-code tools : Terraform, Ansible, Helm, Puppet etc.
- Deployment & management tools : Kubernetes, Docker Swarm etc.

Infrastructure-as-code tool : Terraform basics

- Terraform is an infrastructure as code (IaC) tool that allows you to build, change, and version infrastructure safely and efficiently. This includes both low-level components like compute instances, storage, and networking, as well as high-level components like DNS entries and SaaS features.
- Terraform is distributed as a CLI and is used for writing declarative infrastructure as code.
- The main purpose of the Terraform language is declaring resources which represent infrastructure objects. Terraform *Blocks* are containers for other content and usually represent the configuration of some kind of object, like a resource.

```
terraform {
    required_providers {
      aws = {
        source  = "hashicorp/aws"
        version = "~> 1.0.4"
      }
    }
  }
  variable "aws_region" {}
  variable "base_cidr_block" {
    description = "A /16 CIDR range definition, such as 10.1.0.0/16, that the VPC will use"
    default = "10.1.0.0/16"
  }
  variable "availability_zones" {
    description = "A list of availability zones in which to create subnets"
    type = list(string)
  }
  variable "ingress" {
    description = "list of ingress ports"
    type = list(map(any))
    default = [{port1=90,port2=32,port3=443,port4=53}]
  }
  provider "aws" {
    region = var.aws_region
  }
```

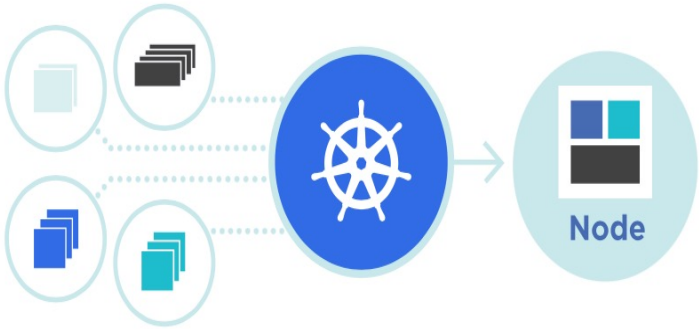
Deployment & management tools : Kubernetes basics

Kubernetes also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications.

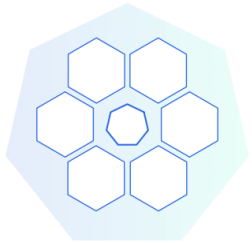
Containers are similar to VMs, but they are lightweight and have relaxed isolation properties to share the Operating System (OS) among the applications. They are portable across clouds and OS distributions. Containers are a good way to bundle and run your applications.

Kubernetes provides you with a framework to run distributed systems resiliently. It takes care of scaling and failover for your application, provides deployment patterns, and more.

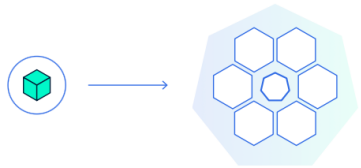
A Kubernetes cluster consists of a set of worker machines, called nodes that run containerized applications.



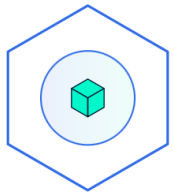
Kubernetes basic modules



1. Create a Kubernetes cluster

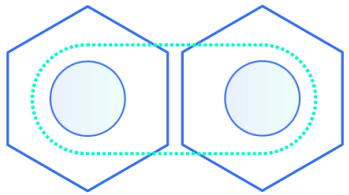


2. Deploy an app

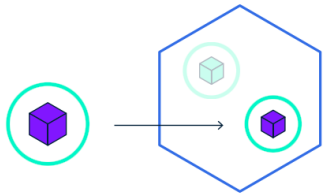


3. Explore your app

4. Expose your app publicly



5. Scale up your app



6. Update your app

Kubernetes cluster components

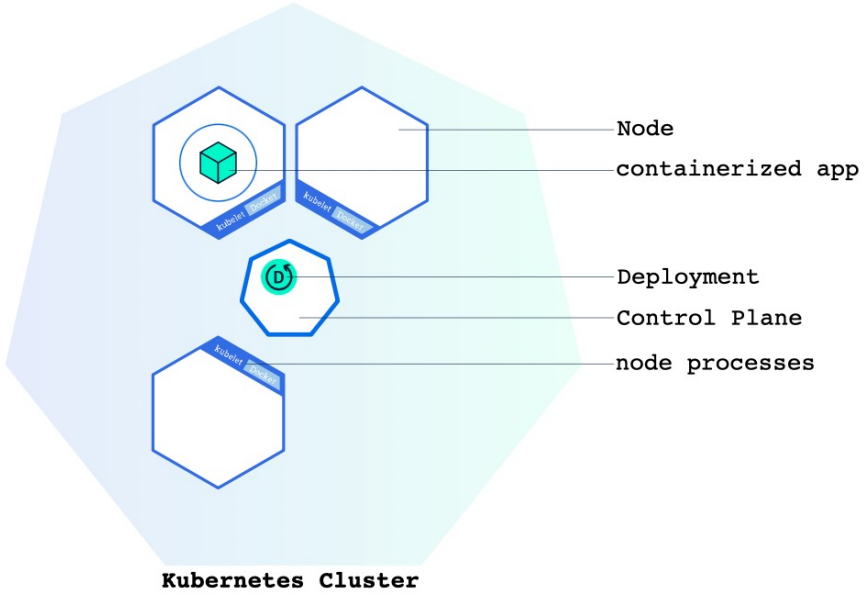
A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. Every cluster has at least one worker node.

The worker node(s) host the pods that are the components of the application workload.

The control plane manages the worker nodes and the Pods in the cluster.

Control Plane Components include kube-apiserver, etcd, kube-scheduler, kube-controller-manager, cloud-controller-manager.

Node Components include kubelet, kube-proxy, Container runtime



Kubernetes objects

Kubernetes objects are persistent entities in the Kubernetes system.

Every Kubernetes object includes two nested object fields that govern the object's configuration: the object ***spec*** and the object ***status***.

Example: in Kubernetes, a Deployment is an object that can represent an application running on your cluster.

Deployments can be created and managed by using the Kubernetes command line interface, **kubectl.**

Create a deployment

- Containerized applications can be deployed on a Kubernetes cluster.
- Provide the information to kubectl in a .yaml file. **kubectl** converts the information to JSON when making the API request.
- One way to create a Deployment using a .yaml file and use the kubectl apply command.
  'kubectl apply -f example-deployment.yaml '
- To list your deployments
  'kubectl get deployments '

example-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2 # tells deployment to run 2 pods matching the template
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Scale a deployment

- The Deployment creates the Pods for running an application.
- We will need to scale the application to keep up with user demand if the traffic increases.
- ***Scaling*** is accomplished by changing the number of replicas in a Deployment
- After change is applied, `n` instances of the applications available as provided in replicas.
- One way to scale a Deployment, by updating the replicas and using a .yaml file and use the kubectl apply command.
  'kubectl apply -f example-deployment.yaml '
- Another way is using kubectl scale command.
  'kubectl scale deployments/nginx-deployment --replicas=4'

! example-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 4
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Next Steps

1. Finish your lab(https://ibm.biz/accelerate-software-week8) by Monday.
2. If you need further help, ask in the slack channel (#ibm-accelerate-2023-software) or join the office hours (Thursday at 6:00 PM ET).
3. Slides and video recordings of the session are available in Week 8 folder.
4. Complete the **hackerrank** test.
5. Network and connect with your peers today and after today's session.

Project/Lab

Deploy to-do-list application on IBM Cloud (Optional)

1

Deploy a Kubernetes cluster from IBM cloud catalog

2

Deploy to-do-list app to the IBM Cloud deployment target

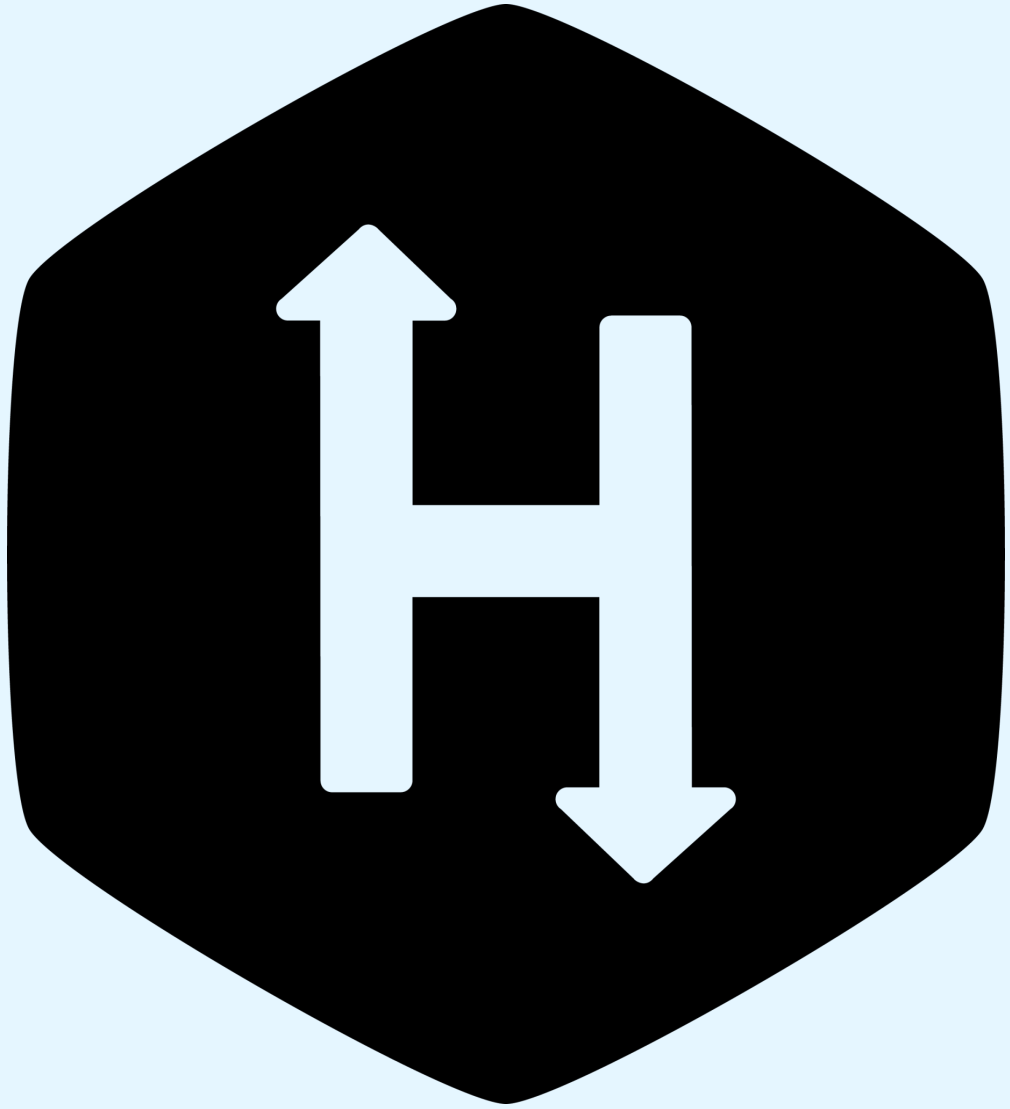
To-do-list Optional Lab: Instructions

- In this Lab, you will be following the instructions to get a cloud cluster and deploy the containerized to-do-list frontend application/microservice on the cloud cluster.

- Follow the detailed steps from **Readme.md** file from github project **to_do_list_week8**:

- Create IBM cloud ID using the instructions and IBM Cloud Feature Code shared via email or using the information here.
- Deploy and configure Kubernetes service cluster from IBM Cloud Catalog.
- Deploy the to-do list front-end application microservice(containerized image) on the provisioned Kubernetes cluster once ready.
- Access your to-do list front-end application endpoint hosted on the cloud cluster.

Required HackerRank Test



IBM

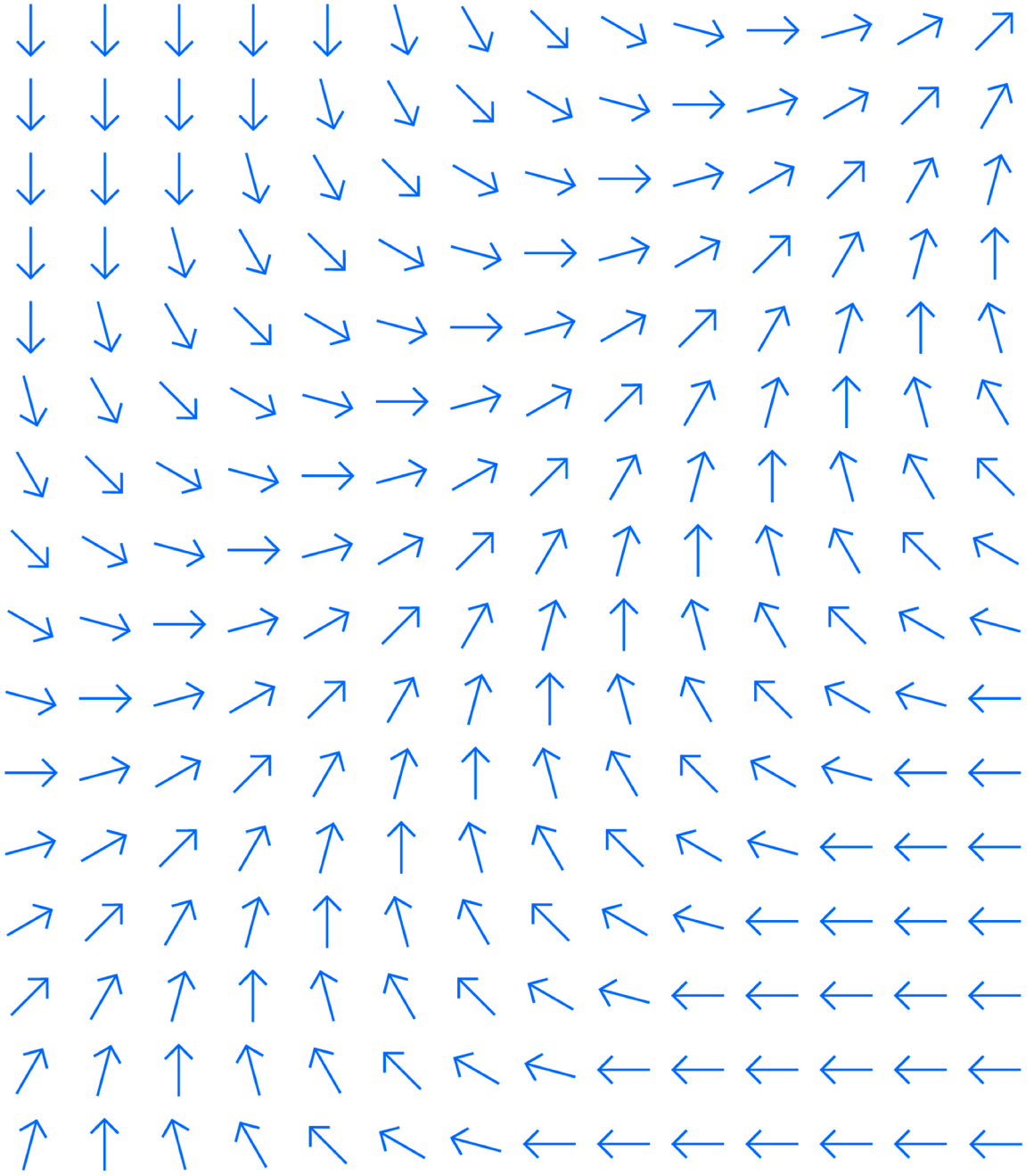
Required HackerRank Test for Week 6

- Tomorrow (Thursday), you should receive an email containing a link to a HackerRank quiz covering content from Week 8.
  - From IBM Corporation Hiring Team – support@hackerrankforwork.com
- Test will include **6 questions**
- Required to earn the IBM Accelerate Badge
- Passing score: 4/6
- Due by next Tuesday, July 30, 11:59 PM ET

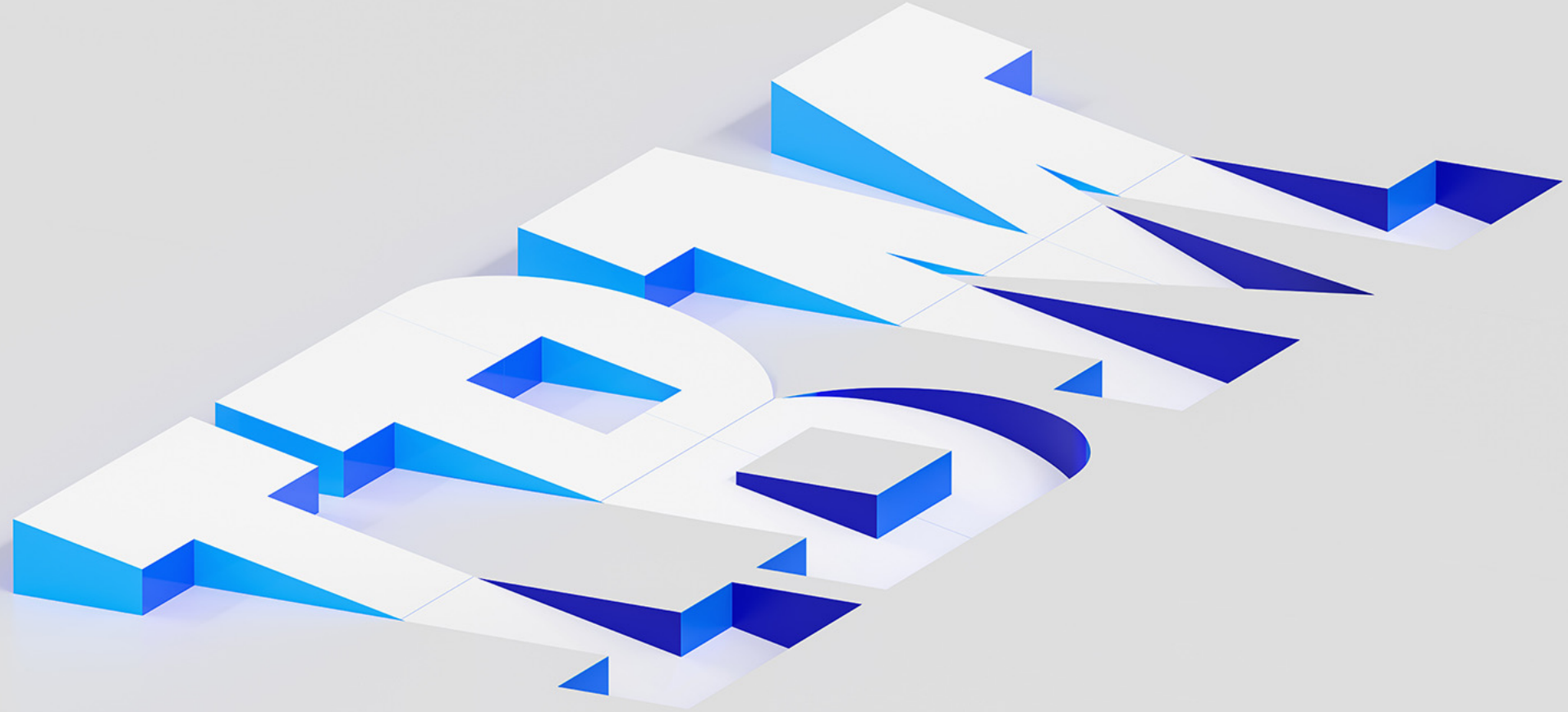
Required HackerRank Test: Preparation

- Review lab and presentation for Week 8
- Use Slack and/or Office Hours to answer any questions

Q&A



IBM



Thank You