Cloud Native

Introduction

About me

Hi there, my name is Bilal and I will Welcome you to DevOps boot camp! I am thrilled to have you join us for this exciting journey of learning and discovery.

In this boot camp, we will be exploring the principles and practices of DevOps, which is a set of methodologies and tools that aims to bridge the gap between software development and operations. DevOps is an increasingly important area in the field of software engineering, as it helps organizations to streamline their processes, improve their agility, and deliver better value to their customers.

By the end of this boot camp, you will have gained a comprehensive understanding of DevOps and its key concepts, as well as practical skills in areas such as infrastructure automation, continuous integration and delivery, monitoring and logging, and more. You will be equipped with the knowledge and tools to apply DevOps principles in your own work and contribute to the success of your organization.

I am always looking to connect with other professionals in the field, share ideas and insights, and stay up to date on the latest trends and developments. I welcome the opportunity to connect with you and explore ways in which we can collaborate and support each other.

GitHub : https://github.com/BilalMaz/DevOps-Architect-BootCamp
LinkedIn : https://www.linkedin.com/in/bilalmazhar-cyber-security-consultant/

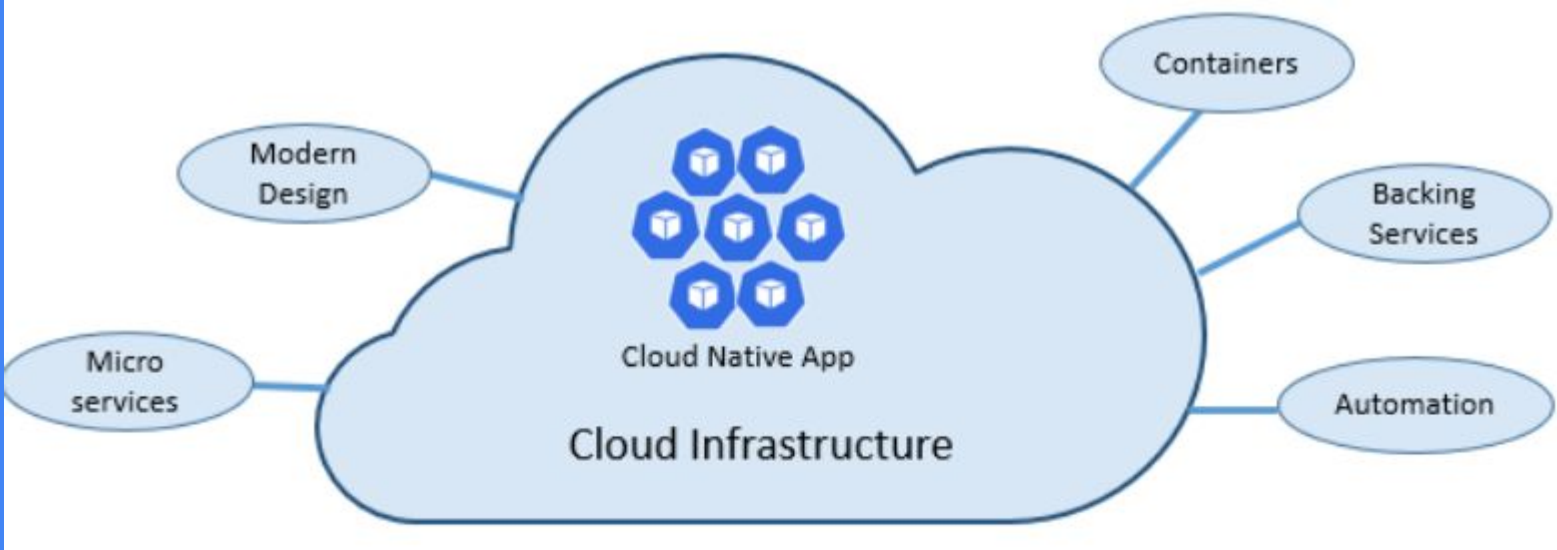
What is Cloud Native?

The term cloud native refers to the concept of building and running applications to take advantage of the distributed computing offered by the cloud delivery model. Cloud native apps are designed and built to exploit the scale, elasticity, resiliency, and flexibility the cloud provides.

As defined by the Cloud Native Computing Foundation (CNCF), Cloud native technologies empower organizations to build and run scalable applications in public, private, and hybrid clouds. Features such as containers, service meshes, **microservices**, immutable infrastructure, and declarative application programming interfaces (APIs) best illustrate this approach.

These features enable loosely coupled systems that are resilient, manageable, and observable. They allow engineers to make high-impact changes frequently and with minimal effort.

The pillars of cloud native



The Cloud

Cloud computing plays a **critical role** in cloud-native architecture. Cloud computing provides the infrastructure, services, and tools needed to build, deploy, and manage cloud-native applications. Some of the key roles that cloud computing plays in cloud-native architecture are:

- **Infrastructure:** Cloud computing provides the underlying infrastructure needed to run cloud-native applications, including compute, storage, and networking resources. Cloud infrastructure services such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP) provide the scalability and flexibility required to run microservices-based applications.
- **Platform:** Cloud platforms such as Kubernetes, Istio, and OpenShift provide the tools and services needed to deploy, manage, and scale containerized applications. These platforms offer features such as automatic load balancing, service discovery, and container orchestration to simplify the management of microservices-based applications.
- **Services:** Cloud providers offer a range of services that can be used to build cloud-native applications, including databases, messaging systems, and machine learning services. By leveraging these services, developers can reduce the amount of time and effort required to build and deploy cloud-native applications.
- **DevOps:** Cloud providers offer a range of DevOps services, including continuous integration and delivery (CI/CD) tools, monitoring and logging services, and automation tools. These services help to streamline the development, deployment, and management of cloud-native applications, enabling developers to deliver new features and services more quickly and efficiently.

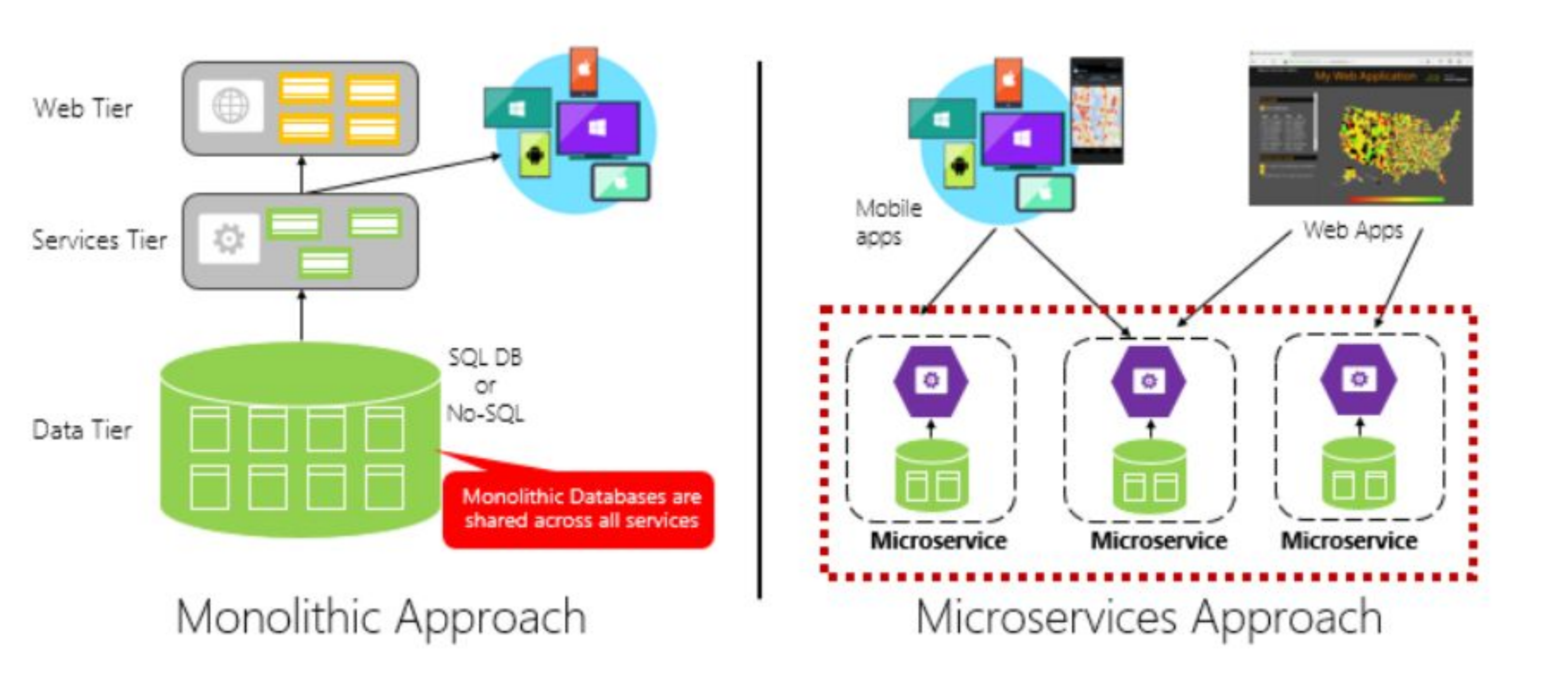
Modern design

How would you design a cloud-native app? What would your architecture look like? To what principles, patterns, and best practices would you adhere? What infrastructure and operational concerns would be important?

- **The Twelve-Factor Application :** A widely accepted methodology for constructing cloud-based applications is the Twelve-Factor Application. It describes a set of principles and practices that developers follow to construct applications optimized for modern cloud environments.
  - Youtube : https://www.youtube.com/watch?v=4yaSftn57Q0&t=1086s
  - Presentation: : https://github.com/BilalMaz/DevOps-Architect-BootCamp/blob/main/PDFs/12%20factor%20app.pdf

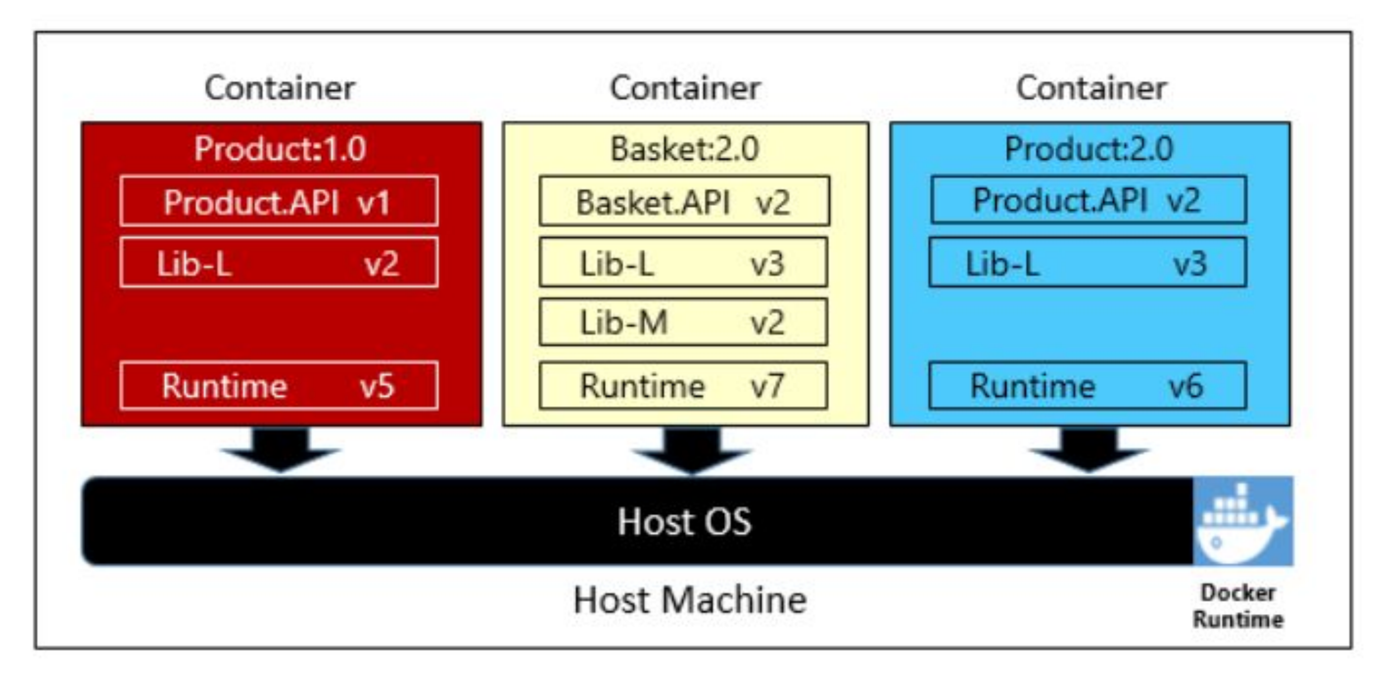
Microservices:

Microservices are the foundational building blocks of cloud-native applications. **Microservices are small, independent,** and **loosely coupled services** that can be developed, deployed, and **scaled independently**. By using microservices, developers can build complex applications more quickly and efficiently.



Containers:

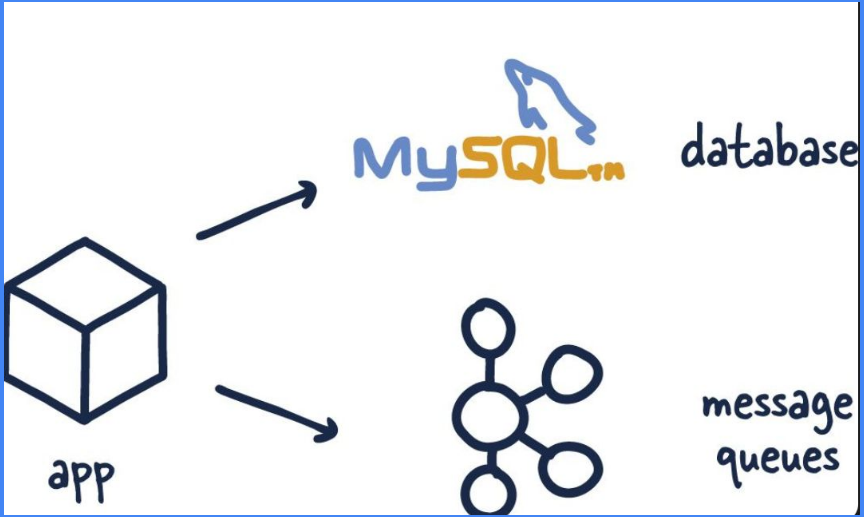
Containers are used to **package** and **deploy individual microservices**, along with their **dependencies** and **configurations**, in a lightweight and portable manner. Containers are designed to be easily moved between different environments, including development, testing, and production.



Backing services

Cloud-native systems typically rely on a variety of **ancillary resources,** such as **databases**, **message brokers**, **monitoring services**, and **identity services**. These resources are referred to as backing services in the context of cloud-native architecture. Backing services provide critical functionality and infrastructure that enable cloud-native applications to operate at scale and with high reliability.

By treating backing services as attached resources and following the Twelve-Factor Application principles, cloud-native applications can easily integrate with these services and maintain their scalability and resilience.

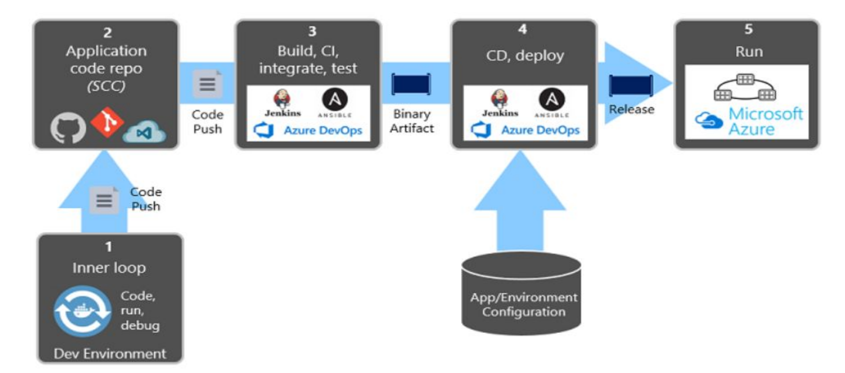


Automation

As you've seen, cloud-native systems embrace microservices, containers, and modern system design to achieve speed and agility. But, that's only part of the story. How do you provision the cloud environments upon which these systems run? How do you rapidly deploy app features and updates? How do you round out the full picture?

" DevOps "

DevOps is a set of practices that combines software development and IT operations to improve the speed and quality of software delivery. By automating key processes such as testing, deployment, and monitoring, DevOps enables developers to deliver new features and services more quickly and reliably


2
Application code repo (SCC)
Code Push
3
Build, CI, integrate, test
Jenkins
ANSIBLE
Azure DevOps
Binary Artifact
4
CD, deploy
Jenkins
ANSIBLE
Azure DevOps
Release
5
Run
Microsoft Azure
Code Push
1
Inner loop
Code, run, debug
Dev Environment
App/Environment Configuration