BITS Pilani
Pilani Campus

CLOUD COMPUTING

Session 16

Arun Vadekkedhil

Session Agenda

Cloud Computing – Engagements in the Cloud

- Serverless Computing
- Backend as a Service (BaaS)
- Mobile BaaS (MBaaS)
- Function as a Service (FaaS)

Future Directions in Cloud

- After Migration, What Next
- Evolution of EDGE computing
- Multi-clouds, a de facto standard

Course Wrap Up

- Review BEL & AR & IR Leases
- Exam Pattern discussion

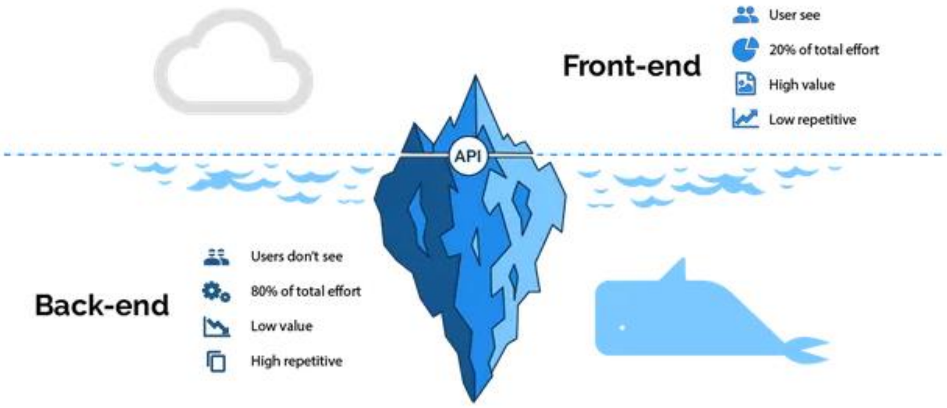
Cloud Deployments

Perception

Behind every **web or mobile application**, there is an array of **backend services**.

They **support** the applications the **frontend consumers are using** and seeing every day. However, the amount of **work required** to create and manage this **backend** is **never simple** and **straightforward**.

At the same time, most of the **business organizations** want to **save themselves the money and time** required to redevelop the wheel **from scratch every time**. That’s why they are preferring to go with an effective **Backend as a Service solution** to get the things done effectively.



BaaS Defined

What

A platform that

- automates backend side development
- takes care of the cloud infrastructure

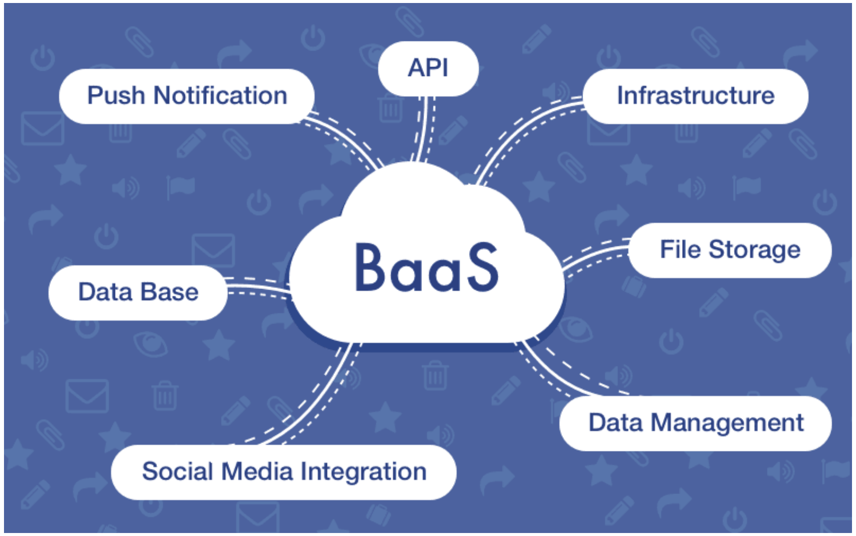
App teams WHO?

- outsource the responsibilities of running and maintaining servers to a third party
- focus on the frontend or client-side development

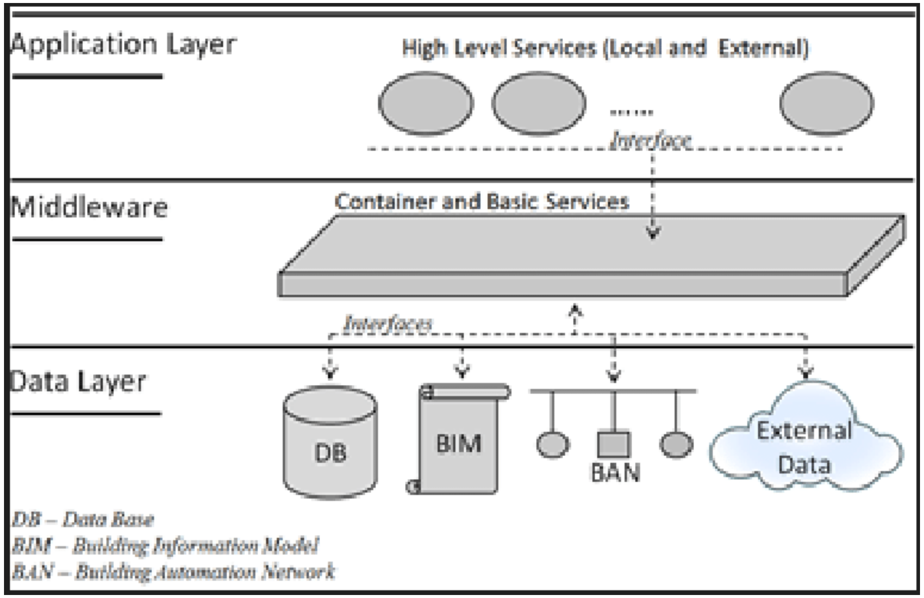
HOW?

Provides a set of tools to help developers to create a backend code speedily with help of ready to use features.

BaaS Capabilities



BaaS Architecture


Application Layer
High Level Services (Local and External)
......
Interface
Middleware
Container and Basic Services
Interfaces
Data Layer
DB
BIM
BAN
External Data
DB – Data Base
BIM – Building Information Model
BAN – Building Automation Network

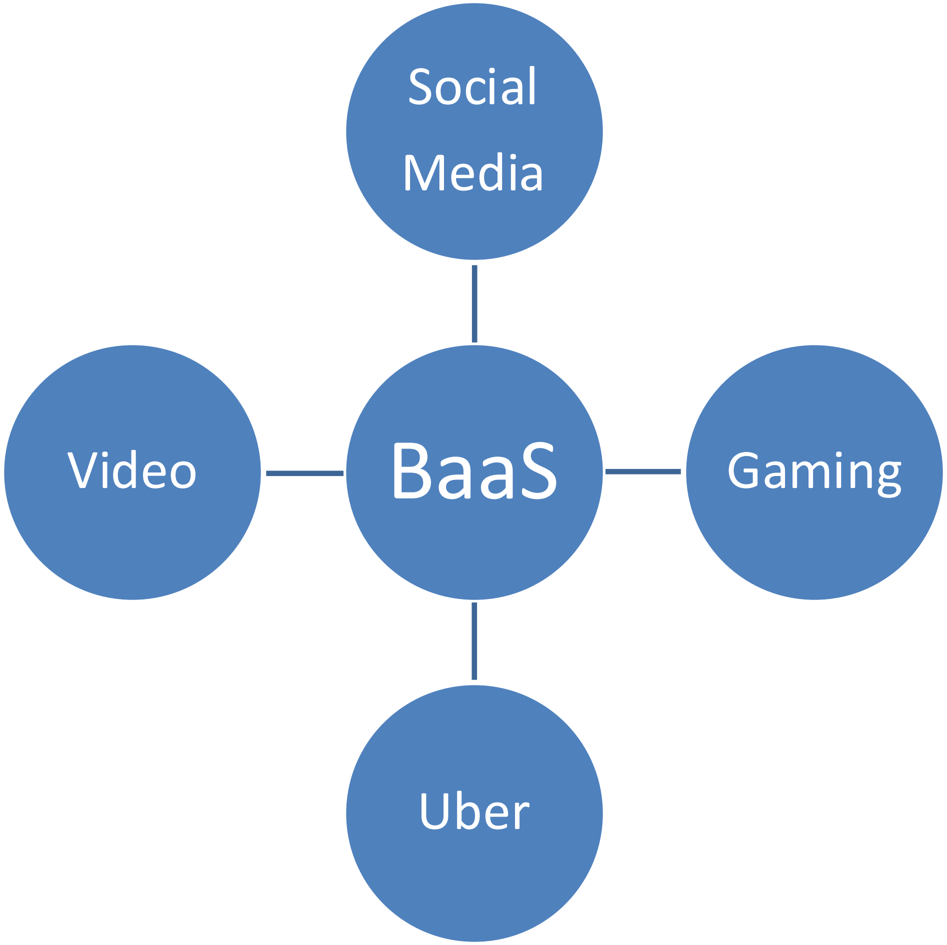
The **first layer** is the foundation and contains the database servers.

A **database cluster** will have at least two servers to replicate data and a backup routine to retrieve data. Most BaaS providers use NoSQL databases on their technology stacks due to scaling flexibility, but there is a growing trend to use SQL databases like Postgres.

The **second layer** is the **application cluster** and contains multiple servers to process requests. The quantity of servers fluctuates throughout the time of the day, and auto-scaling procedures are necessary to fulfill the group with the correct amount of servers.

The **third layer** connects the application servers to the Internet, and it’s composed of **load balancers and CDNs**.

BaaS Examples



Social Media

Login Panels, multi varied data formats to store & process, user authentication

Uber

Push notification, location based services, user authentication

Video Streaming

Get watchlist, curated play list creation, subscription services

Gaming Platform

User authentication, database management, logs

BaaS vs IaaS

Imagine you would like to **build a new software project** and that you will **not use a BaaS**. The first step before you start developing the backend side code is to **set up the servers**. Here is how it will work:

- Login on AWS or any other cloud.
- Go to Instances
- Launch Instance
- Select the Operating System
- Instance Size, Type
- Configure Instance Details
  - Number of instances ,Network, IP, Monitoring, Other settings like Auto Scaling, IAM, etc
- Add Storage
- Security Settings

All right, your **instance is up and running**, and now you can start coding! **Not really**! That is only the **first step** of the process, and you will still need to **install the web-server, database, framework, etc**.

After all that is done, you can start coding. The **time** to perform this process can range from a **few hours** (for a small project with skilled backend developers) to more than a **day for large environments**.

This same process using a **backend as a service will be done with a few clicks and take no more than a few minutes**.

BaaS Pro & Con

Advantages

Speedy Development

Reduced Development price

Serverless, and no need to manage infrastructure

Cloud code Functions
File storage
Backup routine
Social Login
Security
Database
Servers
BaaS
Launch Successful APP
Frontend

Disadvantages

Less flexible as compared to custom coding / deployments

Less customization in comparison to a custom backend

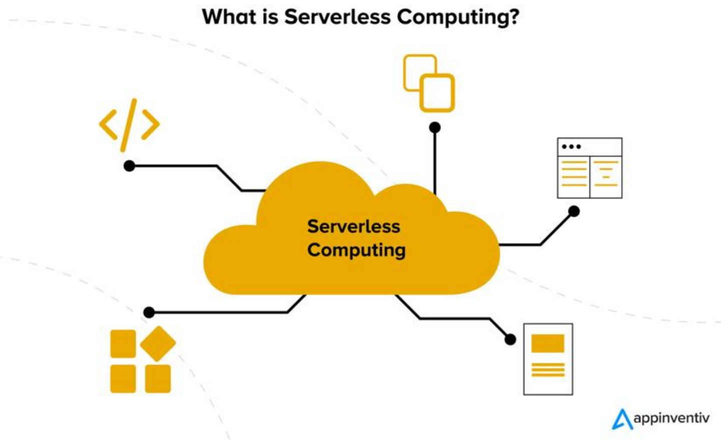
Vendor lock-in possible

innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Serverless Computing



Serverless Computing

Serverless computing is a method of providing **backend services on an as-used basis**.

A serverless provider allows users to write and deploy code without the hassle of worrying about the underlying infrastructure.

A company that gets backend services from a serverless vendor is **charged** based on **their computation** and do not have to **reserve and pay for a fixed amount of bandwidth or number of servers**, as the service is auto-scaling.

Note that *despite the name serverless, physical servers are still used but developers do not need to be aware of them.*

BaaS vs Serverless

There is some overlap between **BaaS and serverless computing**, because in both the developer only has to write their application code and doesn't think about the backend. In addition, many BaaS providers also offer serverless computing services. However, there are significant operational differences between applications built using BaaS and a true serverless architecture.

How the application is constructed

The backends of serverless applications are broken up into functions, each of which responds to events and performs one action only. BaaS server-side functionalities, meanwhile, are constructed however the provider wants, and developers don't have to concern themselves with coding anything other than the frontend of the application.

When code runs

Serverless architectures are event-driven, meaning they run in response to events. Each function only runs when it is triggered by a certain event, and it does not run otherwise. Applications built with BaaS are usually not event-driven, meaning that they require more server resources.

BaaS vs Serverless

Where code runs

Serverless functions can be run from anywhere on any machine, as long as they are still in communication with the rest of the application, which makes it possible to incorporate edge computing into the application's architecture by running code at the network's edge. BaaS is not necessarily set up to run code from anywhere, at any time (although it can be, depending on the provider).

How the application scales

Scalability is one of the biggest differentiators separating serverless architectures from other kinds of architecture. In serverless computing, the application automatically scales up as usage increases. The cloud vendor's infrastructure starts up ephemeral instances of each function as necessary. BaaS applications are not set up to scale in this way unless the BaaS provider also offers serverless computing and the developer builds this into their application.

Serverless on AWS

Serverless is a way to describe the services, practices, and strategies that enables building more agile applications to foster innovations and responses to the changes faster

With serverless computing, infrastructure management tasks like capacity provisioning and patching are handled by AWS, developer can focus on only writing code that serves customers

Serverless services like AWS Lambda come with

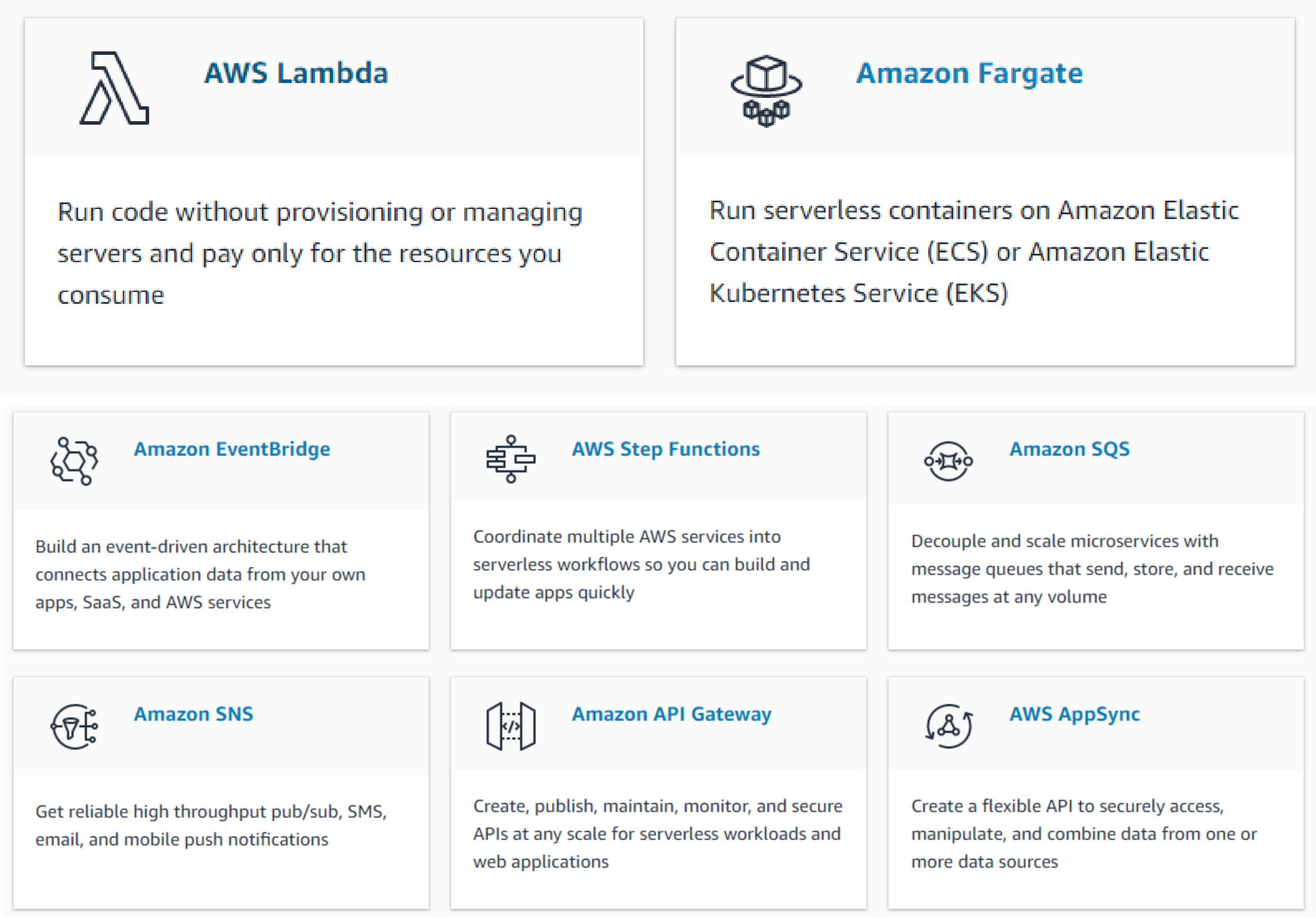
automatic scaling

built-in high availability

and a pay-for-value billing model

Lambda is an event-driven compute service that enables to run code in response to events from over 150 natively-integrated AWS and SaaS sources all without managing any servers

Serverless on AWS



AWS Lambda

Run code without provisioning or managing servers and pay only for the resources you consume

Amazon Fargate

Run serverless containers on Amazon Elastic Container Service (ECS) or Amazon Elastic Kubernetes Service (EKS)

Amazon EventBridge

Build an event-driven architecture that connects application data from your own apps, SaaS, and AWS services

AWS Step Functions

Coordinate multiple AWS services into serverless workflows so you can build and update apps quickly

Amazon SQS

Decouple and scale microservices with message queues that send, store, and receive messages at any volume

Amazon SNS

Get reliable high throughput pub/sub, SMS, email, and mobile push notifications

Amazon API Gateway

Create, publish, maintain, monitor, and secure APIs at any scale for serverless workloads and web applications

AWS AppSync

Create a flexible API to securely access, manipulate, and combine data from one or more data sources

Serverless Advantages

1. No server management

Serverless computing does not mean that there are no servers, but it definitely means that developers or companies do not have to worry about these servers. These servers are managed by vendors and developers can focus on expanding their application without being worried about their server capacity.

2. Lower cost

This is, obviously, very cost effective because a developer is only paying for the server space they are using.

3. Flexible scalability

This is one of the important advantages of serverless computing. A developer does not have to worry about scalability of their web application because serverless computing has the ability to scale according to traffic volumes.

4. Fewer things to worry

A company or developer does not have to worry about security, patch, bugs and many other things because their servers are managed by vendors.

Serverless Disadvantages

1. Not suitable for long term tasks:

Serverless are a good option for short term or real time tasks. If a task takes longer time to complete then the company might end up paying more for compute time. For example, if a large file has to be uploaded that takes more time then it will require additional functions till it's complete and the developer ends up paying more for that.

2. High non-performance penalty:

A business pays for what they use, but if they end up not using these functions then there will be a high non-performance penalty. These functions may suffer from cold start penalty as well and can be very slow when used after a certain period of time.

3. Vendors lock in:

When a business depends on a vendor for all its backend services for a web application, it ends up losing control over their hardware, updates and run times. If they want to switch then it gets very difficult. A business or developer has to re-engineer if they wish to switch to another service provider.

4. Security concerns:

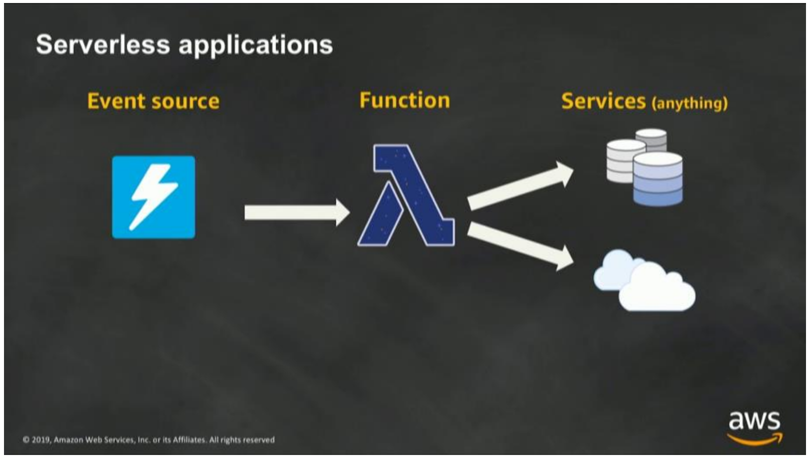
New security concerns are introduced when a vendor provides servers to a business. This can be a huge problem if the application contains personal or sensitive information like credit card details. This happens because companies are not given their own physical servers, vendors will be running code for many customers on a single server.

innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Function as a Service



Serverless applications

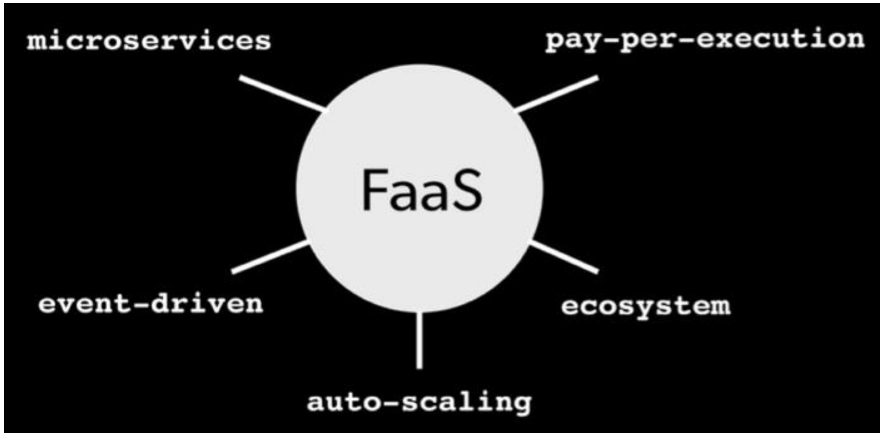
Event source

Function

Services (anything)

FaaS

Function-as-a-Service (FaaS) is a serverless way to execute modular pieces of code on the edge. FaaS lets developers write and update a piece of code on the fly, which can then be executed in response to an event, such as a user clicking on an element in a web application. This makes it easy to scale code and is a cost-efficient way to implement microservices.



Microservice

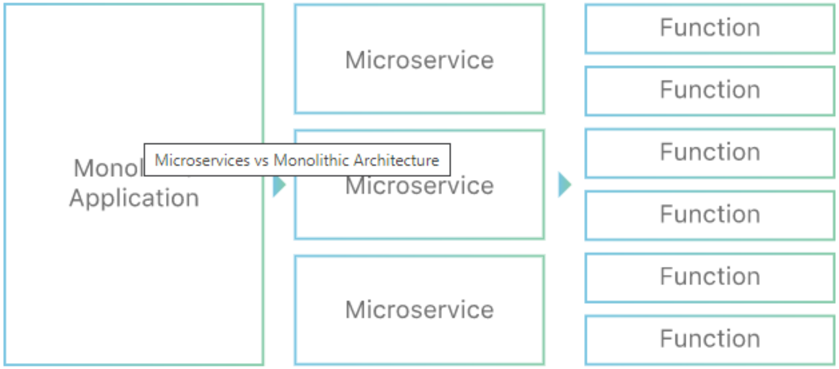
The approach of building an application out of a set of modular components is known as microservice architecture.

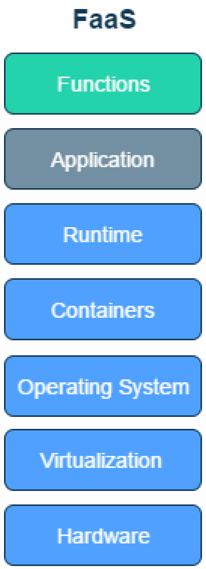
Dividing an application into microservices is appealing to developer because it means they can create and modify small pieces of code which can be easily implemented into their codebases.

This is in contrast to monolithic architecture, in which all the code is interwoven into one large system.

With large monolithic systems, even a minor changes to the application requires a hefty deploy process.

FaaS eliminates this deploy complexity. Using serverless code like FaaS, web developers can focus on writing application code, while the serverless provider takes care of server allocation and backend services.





FaaS Benefits

Focus more on code, not infrastructure:

With FaaS, can divide the server into functions that can be scaled automatically and independently so don’t have to manage infrastructure

This allows to focus on the app code and can dramatically reduce time-to-market

Pay only for the resources you use, when you use them:

With FaaS, you pay only when an action occurs

When the action is done, everything stops—no code runs, no server idles, no costs are incurred

FaaS is, therefore, cost-effective, especially for dynamic workloads or scheduled tasks

FaaS also offers a superior total-cost-of-ownership for high-load scenarios

Scale up or down automatically:

With FaaS, functions are scaled automatically, independently, and instantaneously, as needed

When demand drops, FaaS automatically scales back down

Get all the benefits of robust cloud infrastructure:

FaaS offers inherent high availability because it is spread across multiple availability zones per geographic region

can be deployed across any number of regions without incremental costs

innovate achieve lead

BITS Pilani

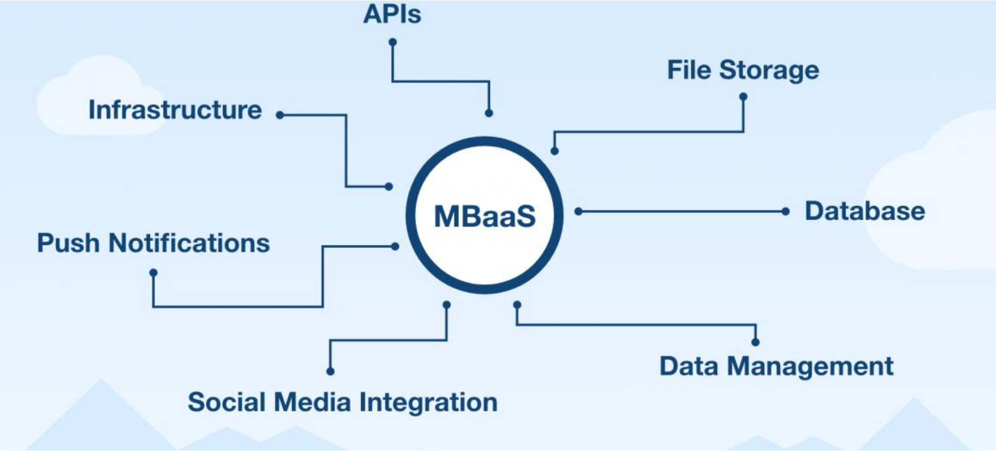
Pilani|Dubai|Goa|Hyderabad

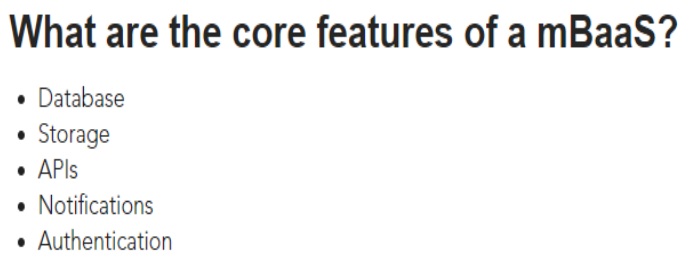
Mobile BaaS

APIs
Infrastructure
File Storage
MBaaS
Push Notifications
Database
Social Media Integration
Data Management

MBaaS

mBaaS – Mobile backend as a service helps developers in linking their applications, either mobile or websites, to the cloud via application programming interfaces (API). Other than this, the mobile backend as a service is also making grounds for helping developers in accelerating backend development, improve user management, provides push notifications, and much more. Common mBaaS features include database graphical interface, APIs, email verification, reset password, push-notifications, and more.




What are the core features of a mBaaS?

- Database
- Storage
- APIs
- Notifications
- Authentication

MBaaS Architecture

The first layer - Database

Is the foundation and contains the database servers

A database cluster has at least two servers to replicate data and a backup routine to retrieve data

The second layer - Application

Is the application cluster and contains multiple servers to process requests

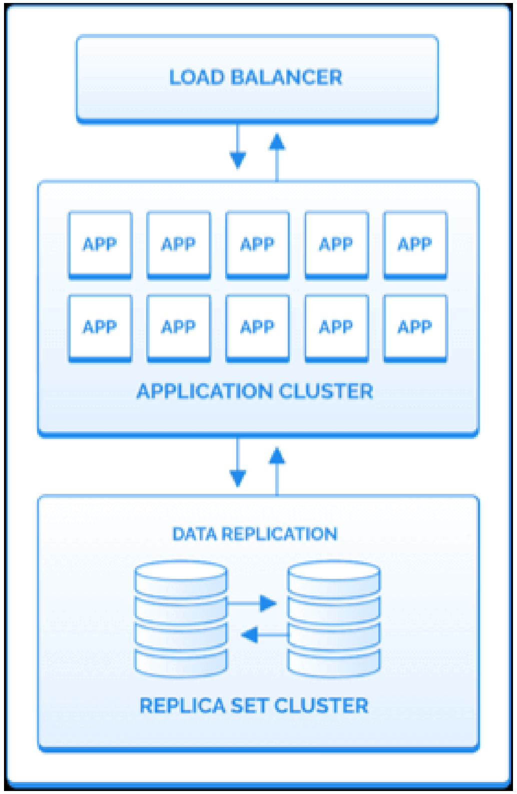
Quantity of servers fluctuates throughout the time of the day

Auto-scaling procedures are necessary to fulfill the correct number of servers

The third layer - Gateway

Connects the application servers to the Internet

Composed of load balancers and CDNs



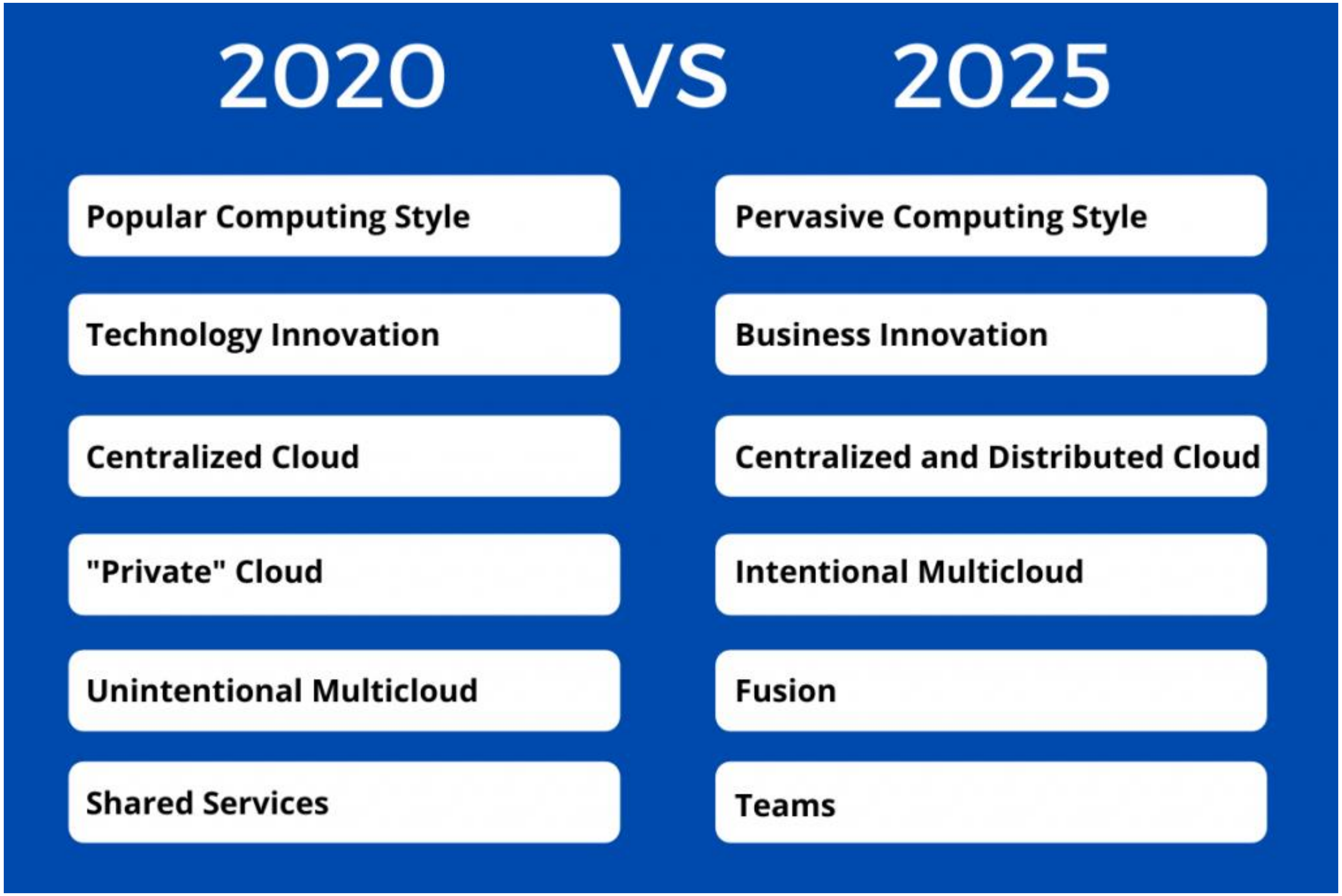
innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Future Directions

What Next in Cloud



2020	2025
Popular Computing Style	Pervasive Computing Style
Technology Innovation	Business Innovation
Centralized Cloud	Centralized and Distributed Cloud
"Private" Cloud	Intentional Multicloud
Unintentional Multicloud	Fusion
Shared Services	Teams

EDGE Computing

Edge computing is a networking philosophy focused on bringing computing as close to the source of data as possible in order to reduce latency and bandwidth use.

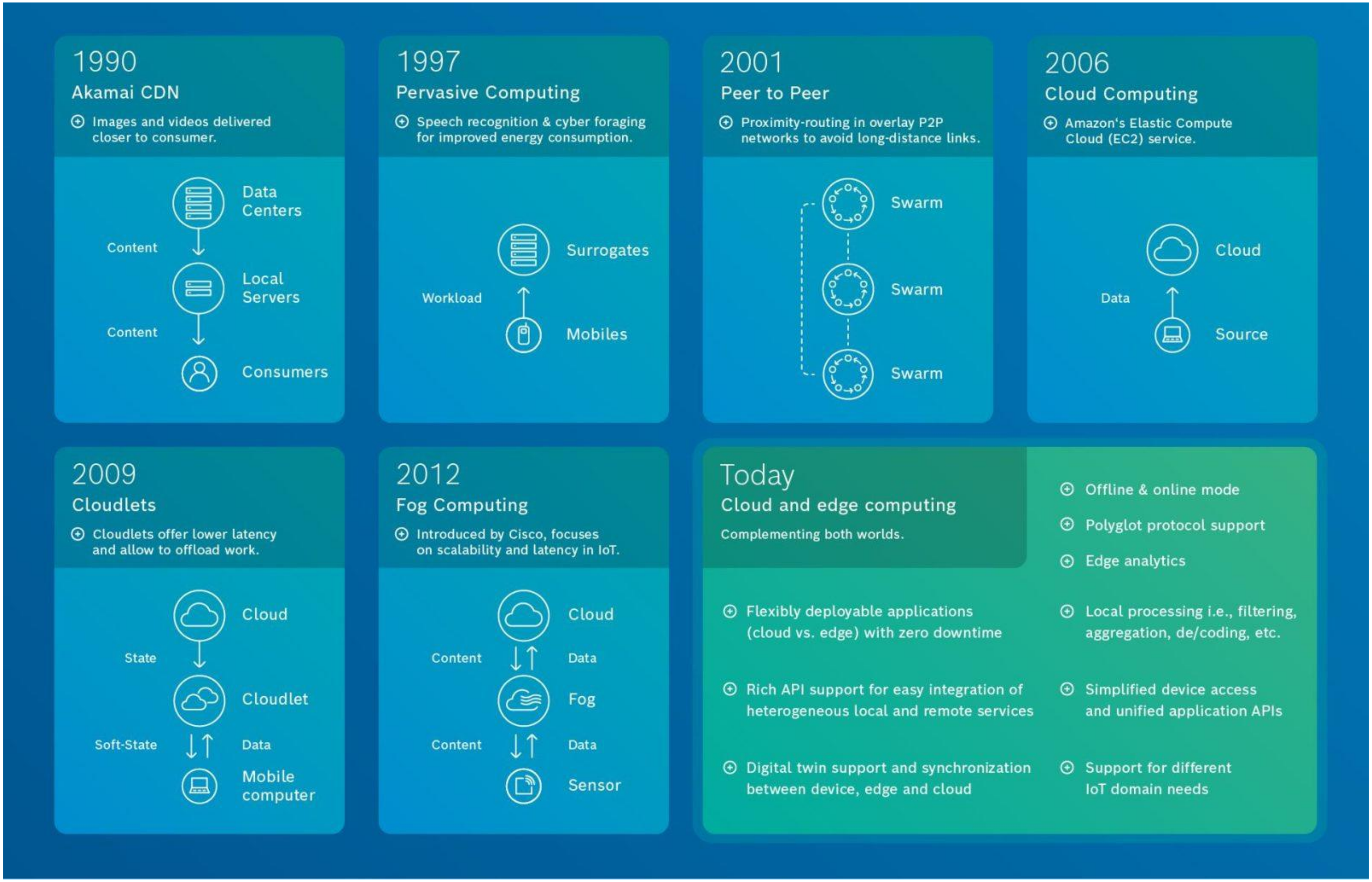
In simpler terms, edge computing means running fewer processes in the cloud and moving those processes to local places, such as on a user’s computer, an IoT device, or an edge server.

Bringing computation to the network’s edge minimizes the amount of long-distance communication that has to happen between a client and server.

What is the Edge?

For Internet devices, the network edge is where the device, or the local network containing the device, communicates with the Internet. The edge is a bit of a fuzzy term; for example a user’s computer or the processor inside of an IoT camera can be considered the network edge, but the user’s router, ISP, or local edge server are also considered the edge. The important takeaway is that the edge of the network is geographically close to the device, unlike origin servers and cloud servers, which can be very far from the devices they communicate with.

EDGE Computing



1990
Akamai CDN
⊕ Images and videos delivered closer to consumer.

1997
Pervasive Computing
⊕ Speech recognition & cyber foraging for improved energy consumption.

2001
Peer to Peer
⊕ Proximity-routing in overlay P2P networks to avoid long-distance links.

2006
Cloud Computing
⊕ Amazon's Elastic Compute Cloud (EC2) service.

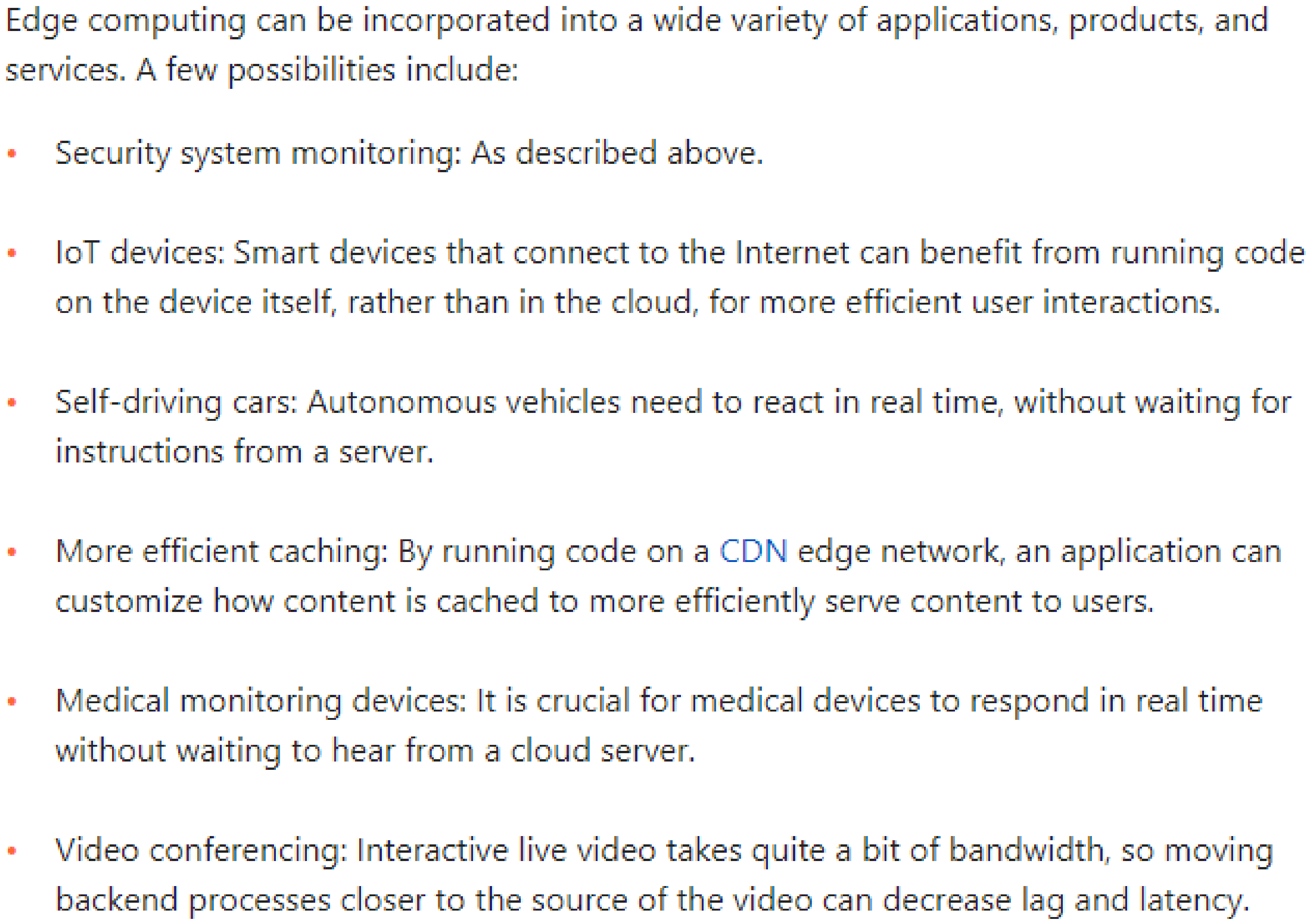
2009
Cloudlets
⊕ Cloudlets offer lower latency and allow to offload work.

2012
Fog Computing
⊕ Introduced by Cisco, focuses on scalability and latency in IoT.

Today
Cloud and edge computing
Complementing both worlds.

- ⊕ Offline & online mode
- ⊕ Polyglot protocol support
- ⊕ Edge analytics
- ⊕ Flexibly deployable applications (cloud vs. edge) with zero downtime
- ⊕ Local processing i.e., filtering, aggregation, de/coding, etc.
- ⊕ Rich API support for easy integration of heterogeneous local and remote services
- ⊕ Simplified device access and unified application APIs
- ⊕ Digital twin support and synchronization between device, edge and cloud
- ⊕ Support for different IoT domain needs

EDGE Computing Use Cases



Edge computing can be incorporated into a wide variety of applications, products, and services. A few possibilities include:

- Security system monitoring: As described above.
- IoT devices: Smart devices that connect to the Internet can benefit from running code on the device itself, rather than in the cloud, for more efficient user interactions.
- Self-driving cars: Autonomous vehicles need to react in real time, without waiting for instructions from a server.
- More efficient caching: By running code on a CDN edge network, an application can customize how content is cached to more efficiently serve content to users.
- Medical monitoring devices: It is crucial for medical devices to respond in real time without waiting to hear from a cloud server.
- Video conferencing: Interactive live video takes quite a bit of bandwidth, so moving backend processes closer to the source of the video can decrease lag and latency.

In Conclusion..

Points to Ponder – ChatGPT's Prediction

1. **Multi-cloud**: The use of multiple cloud providers is becoming increasingly common, as companies seek to avoid vendor lock-in and take advantage of the unique capabilities of different cloud platforms.

2. **Serverless Computing**: Serverless computing is gaining in popularity, as it allows developers to write and run applications without having to worry about managing infrastructure.

3. **Edge Computing**: With the growth of IoT devices and real-time applications, there is a need for computing resources to be located closer to the source of data. Edge computing involves processing data at or near the edge of the network, rather than sending it back to the cloud.

4. **Artificial Intelligence and Machine Learning**: Cloud providers are incorporating AI and machine learning capabilities into their platforms, making it easier for developers to build intelligent applications.

Points to Ponder – ChatGPT's Prediction

5.**Quantum Computing**: Quantum computing is still in its early stages, but it has the potential to transform cloud computing by providing a significant boost in computing power.

6.**Green Computing**: With the growing concern for the environment, cloud providers are making efforts to reduce their carbon footprint by using renewable energy and implementing energy-efficient technologies.

7.**Hybrid Cloud**: As more companies adopt cloud computing, hybrid cloud solutions that combine on-premises and cloud-based infrastructure are becoming increasingly popular.

8.**Security and Compliance**: With the increasing number of data breaches and cyber threats, cloud providers are investing in advanced security and compliance measures to protect their customers' data.

The technology you use impresses no one. The experience you create with it is everything

Sean Gerety

Hey !

To reach the top...

You don't have to be a book worm !

Jus' stay cool 'n' give your Best !

My best wishes

are with you !