Session 1: Basics of Cloud Computing

**Introduction**

In today's digital age, cloud computing has revolutionized the way we store, access, and utilize data and applications. It has become an integral part of various industries, offering numerous benefits and opportunities for businesses and individuals alike. This article aims to provide students with a comprehensive understanding of the basics of cloud computing, shedding light on its key concepts and components.

**What is Cloud Computing?**

Cloud computing refers to the delivery of computing services, such as servers, storage, databases, software, and networking, over the Internet. Instead of relying on local servers or personal devices, users can access and utilize these resources from anywhere, at any time, using any internet-connected device.

**Essentials of Cloud Computing**

The essentials of cloud computing include the following key components:

* **On-Demand Self-Service:** Cloud computing enables users to provision computing resources, such as processing power, storage, and network access, on-demand without requiring human interaction with the service provider. Users can easily access and manage resources as needed.
* **Broad Network Access:** Cloud services are available over the network and can be accessed by users through various devices, such as laptops, tablets, and smartphones. This allows for ubiquitous access to cloud resources from different locations and platforms.
* **Resource Pooling:** Cloud providers consolidate computing resources to serve multiple users simultaneously, dynamically allocating and reallocating resources based on demand. The infrastructure is shared, and users typically have no control or knowledge of the physical location of the resources they are using.
* **Rapid Elasticity:** Cloud systems have the ability to quickly scale resources up or down based on workload demands. Users can easily request additional resources during peak periods and release them when they are no longer needed. This elasticity allows for efficient resource utilization and cost optimization.
* **Measured Service:** Cloud computing provides transparency and control over resource usage by enabling measurement, monitoring, and reporting of resource consumption. This allows users to understand and optimize their resource usage and provides the basis for pay-as-you-go pricing models.
* **Multi-tenancy:** The cloud infrastructure is designed to support multiple users or tenants on the same physical resources. The resources are logically isolated to ensure security, privacy, and performance for each user. Multi-tenancy enables economies of scale and resource efficiency.
* **Service Models:** Cloud computing offers different service models to cater to various user needs:

1. **Infrastructure as a Service (IaaS):** Users have access to virtualized computing resources, such as virtual machines, storage, and networks, allowing them to deploy and manage their applications.
2. **Platform as a Service (PaaS):** Users can develop, deploy, and manage applications using programming languages, libraries, and tools provided by the cloud provider without worrying about the underlying infrastructure.
3. **Software as a Service (SaaS):** Users can access and use software applications provided by the cloud provider over the internet, without the need for installation or management on their local devices.

These essentials of cloud computing form the foundation for the flexible, scalable, and cost-effective delivery of computing services over the Internet.

**Need for Cloud Computing**

* **Reduced Cost:**Cloud computing reduces costs for businesses through pay-as-you-go pricing, infrastructure cost reduction, operational efficiency, scalability, and resource optimization, reduced energy consumption, reduced maintenance and upgrade costs, and robust disaster recovery and business continuity capabilities. Businesses can avoid upfront investments in hardware, eliminate the need for managing physical infrastructure, automate routine tasks, optimize resource utilization, reduce energy consumption, avoid expensive maintenance and upgrades, and benefit from the cloud provider's data redundancy and high availability. These cost-saving advantages allow businesses to optimize their IT budgets, focus on strategic initiatives, and achieve growth and innovation.
* **Scalability:** Cloud computing is essential for scalability because it allows for dynamic allocation and reallocation of computing resources based on demand. It enables rapid deployment, follows a pay-as-you-go pricing model, offers global availability, and provides scalable services. This enables organizations to quickly and efficiently scale their applications and resources as needed, without upfront investments or underutilization risks.
* **Remote Access:**Cloud computing is crucial for remote access because it allows users to access data and applications from anywhere, provides scalability to meet changing demands, offers cost efficiency by avoiding upfront infrastructure investments, ensures security through robust measures implemented by cloud providers, enables collaboration and productivity among remote teams, and facilitates seamless remote work.
* **Disaster Relief:**Cloud computing is crucial in disaster relief because it provides data backup and recovery, enables rapid deployment of infrastructure, supports scalability, facilitates collaboration and communication, allows for remote access and mobility, aids in information sharing and analytics, and offers cost efficiency for relief organizations.
* **Ease of implementation:**Cloud computing provides ease of implementation through simplified infrastructure management, rapid resource provisioning, preconfigured services, easy integration, flexibility and agility, accessibility and collaboration, and cost-effective pay-as-you-go pricing. This streamlines the implementation process, reduces complexity, and accelerates the adoption of cloud-based solutions.

**History of Cloud Computing**

**Client-Server Technology**

Client-server technology plays a significant role in the history and development of cloud computing. It forms the foundation for distributed architecture and enables the seamless delivery of cloud services to end users. Understanding the concept of client-server technology provides valuable insights into the evolution of cloud computing.

In the early days of computing, a centralized mainframe computer served as the sole provider of processing power and data storage. Users interacted with the mainframe through "dumb" terminals that had limited capabilities and relied heavily on the mainframe for all processing tasks. This model, known as the mainframe era, had several limitations, including high costs, limited accessibility, and a lack of flexibility.

The advent of client-server technology in the 1980s marked a significant shift in computing architecture. The client-server model introduced a decentralized approach where computing resources were distributed between two distinct components: the client and the server.

The client, typically a personal computer or workstation, was responsible for running the user interface and handling user interactions. It allowed users to perform tasks locally, such as data entry, data manipulation, and running specific applications. The client provided a more interactive and user-friendly experience compared to the earlier dumb terminals.

On the other hand, the server acted as a central hub responsible for managing and storing data, performing complex computations, and delivering services to clients. Servers were equipped with robust processing power, high-capacity storage, and specialized software to handle various tasks efficiently. They acted as the backbone of the system, ensuring data integrity, security, and reliability.

The client-server architecture revolutionized computing by decentralizing processing power and introducing a more flexible and scalable approach. This shift set the stage for the evolution of cloud computing.

As technology advanced, the Internet became widely available, enabling connectivity and communication between clients and servers over vast distances. This connectivity paved the way for the emergence of cloud computing. The client-server model, combined with internet connectivity, laid the foundation for the cloud computing infrastructure we know today.

Cloud computing takes the client-server concept to the next level by extending the capabilities of servers beyond a single physical machine. Instead of relying on local servers, cloud computing leverages virtualization technologies to create a pool of shared computing resources that can be dynamically allocated and scaled based on demand. Cloud service providers manage these virtualized resources, allowing users to access and utilize them over the Internet.

In summary, client-server technology played a crucial role in the history of cloud computing. It introduced a distributed architecture, empowering users with more capable clients and centralized servers. The shift from centralized mainframes to decentralized client-server systems set the stage for the scalable, flexible, and accessible nature of cloud computing. The concepts and principles of client-server technology continue to influence and shape the development of cloud computing as it continues to evolve.

**Peer-to-Peer Approach**

The peer-to-peer (P2P) approach in cloud computing has its roots in the early days of networking and distributed computing. Before the advent of centralized cloud infrastructures, the P2P approach was an influential concept that laid the foundation for the development of cloud computing as we know it today.

The concept of P2P computing emerged in the 1980s with the advent of early networking technologies. In a P2P network, individual computers, known as peers, collaborate and share resources directly with each other, without relying on a central server or authority. This approach aimed to harness the collective power and capabilities of multiple machines to achieve distributed computing tasks.

During the 1990s, P2P gained popularity with the rise of file-sharing applications like Napster and BitTorrent. These applications allowed users to directly exchange files with each other without relying on a central server. This decentralized model of sharing files opened new possibilities for efficient data distribution and collaboration.

**P2P and Cloud Computing:**  
The P2P approach influenced the development of cloud computing in several ways:

**Resource Sharing:** The P2P model demonstrated the value of resource sharing among peers. In cloud computing, this concept is extended to shared infrastructure, platforms, and software services, enabling users to access and utilize resources on demand.

**Scalability and Redundancy:** P2P networks demonstrated the ability to scale by adding more peers to the network. Similarly, cloud computing leverages the concept of scalability, allowing resources to be dynamically allocated or deallocated based on demand. Redundancy is also a shared characteristic, as both P2P networks and cloud computing ensure data and services are distributed across multiple nodes or servers for increased reliability.

**Decentralization:** P2P networks challenged the traditional client-server model by decentralizing control and authority. Cloud computing also aims to distribute resources across multiple data centers and locations, reducing single points of failure and enhancing reliability.

**User Collaboration:**P2P networks encouraged user collaboration and participation. Cloud computing platforms often provide collaboration features that allow multiple users to work together on shared documents or projects in real time.

**Hybrid Models:** P2P and cloud computing can also be combined in hybrid models. For example, some cloud storage services utilize a combination of centralized servers and P2P protocols to optimize data transfer and availability.

It is important to note that while P2P influenced the development of cloud computing, the modern cloud infrastructure primarily relies on centralized data centers operated by cloud service providers. These providers offer standardized and highly scalable services, allowing users to access computing resources on demand.

**Distributed Computing**

Distributed computing is a concept closely related to the history and evolution of cloud computing. It laid the foundation for the development of cloud computing by addressing the challenges of sharing computing resources and enabling collaboration among multiple users.

Historically, computing resources were centralized within large mainframe computers, which were expensive and accessible to a limited number of users. As computer networks emerged, distributed computing emerged as a solution to leverage the power of multiple interconnected machines.

In the 1970s and 1980s, the concept of distributed computing gained traction with the development of Local Area Networks (LANs). LANs allowed multiple computers within a limited area to share resources, such as printers and storage devices. This decentralized approach improved efficiency and collaboration within organizations but was limited to a specific physical location.

In the 1990s, the Internet became widely accessible, leading to the expansion of distributed computing beyond local networks. The emergence of the World Wide Web and the development of web-based applications enabled users to access services and resources remotely. This marked the shift toward a more distributed computing model on a global scale.

The term "cloud computing" emerged in the early 2000s, with the concept being popularized by companies like Amazon, Google, and Salesforce. These companies recognized the potential of delivering computing resources and services over the Internet. They began offering infrastructure, platforms, and software as services, forming the foundation of the cloud computing paradigm.

Cloud computing builds upon the principles of distributed computing by leveraging the internet to provide on-demand access to a wide range of resources. It enables users to access computing power, storage, and software applications from remote data centers, eliminating the need for local infrastructure.

By distributing resources across multiple servers and data centers, cloud computing enhances scalability, reliability, and availability. It allows users to scale their resources up or down based on demand, ensuring efficient utilization of computing power. Moreover, the redundancy and fault tolerance built into cloud infrastructure minimizes the risk of service disruptions and data loss.

In summary, distributed computing served as the precursor to cloud computing, paving the way for the development of a globally accessible and scalable computing model. Cloud computing expanded on the distributed computing concept, enabling the delivery of various services and resources over the Internet. The evolution from centralized mainframe computers to distributed networks and, ultimately, to cloud computing has transformed the way we store, access, and utilize computing resources in today's interconnected world.

**Evolution of Cloud Computing from Grid Computing**

Cloud computing has evolved from the concept of grid computing, which was another significant milestone in the history of distributed computing. Grid computing focused on harnessing the collective power of geographically distributed computers to solve complex problems and perform large-scale computations.

Grid computing emerged in the late 1990s as a response to the increasing need for massive computational power required by scientific research, data analysis, and simulations. It aimed to create a virtual infrastructure by connecting geographically dispersed resources, including computers, storage systems, and scientific instruments.

Grid computing systems were typically designed for specific research projects or scientific communities. They emphasized sharing and collaboration among institutions, allowing them to combine their computing resources to achieve higher performance and efficiency. The key characteristics of grid computing included resource sharing, coordinated problem-solving, and decentralized control.

While grid computing provided a scalable and distributed computing environment, it had limitations in terms of scalability, ease of use, and resource allocation. These limitations led to the evolution of cloud computing, which addressed these challenges and offered a more user-friendly and scalable model.

**Platform Virtualisation**

**Platform virtualization**, also known as **hardware virtualization**, is a technology that enables the creation and management of virtual machines (VMs) on a single physical computer. It allows multiple operating systems (OS) and applications to run independently and simultaneously on a shared hardware platform.

The primary goal of platform virtualization is to maximize the utilization of physical computing resources by dividing them into isolated and independent virtual environments. Each virtual machine acts as a self-contained entity, running its own OS and applications, while sharing the underlying hardware resources.

**Key Components of Platform Virtualization:**

* **Hypervisor:** The hypervisor, also referred to as a virtual machine monitor (VMM), is the core software responsible for managing and allocating hardware resources among multiple virtual machines. It sits between the physical hardware and the virtual machines, controlling their access to CPU, memory, storage, and network resources. The hypervisor ensures isolation and provides an interface for configuring and managing virtual machines.
* **Virtual Machine:** A virtual machine represents a complete and independent computing environment encapsulated within a software container. It includes a virtualized set of hardware components, such as virtual processors, memory, storage, and network interfaces. Each virtual machine runs its own instance of an operating system, allowing multiple OS environments to coexist on the same physical hardware.
* **Virtual Machine Manager:** A virtual machine manager is a management tool or interface used to create, configure, monitor, and manage virtual machines. It allows administrators to allocate hardware resources, control network connectivity, and perform operations like starting, stopping, and migrating virtual machines.

**Benefits of Platform Virtualization:**

* **Server Consolidation:**Platform virtualization enables efficient server consolidation by running multiple virtual machines on a single physical server. This consolidation reduces hardware costs, power consumption, and physical space requirements while increasing resource utilization and scalability.
* **Isolation and Security:** Each virtual machine operates in an isolated environment, ensuring that failures or issues in one virtual machine do not affect others. This isolation enhances security by preventing unauthorized access and minimizing the impact of potential security breaches.
* **Resource Optimization:** Virtualization allows for flexible allocation and reallocation of hardware resources among virtual machines. It enables dynamic adjustment of resource allocation based on workload demands, optimizing resource utilization and improving performance.
* **Testing and Development:** Platform virtualization provides a sandbox-like environment for software development, testing, and debugging. Developers can create multiple virtual machines with different configurations, OS versions, or software stacks to simulate various scenarios without affecting the production environment.
* **High Availability and Disaster Recovery:** Virtualization facilitates easy replication and migration of virtual machines, enabling quick disaster recovery and ensuring high availability of critical applications. Virtual machines can be moved to different physical servers in case of hardware failures or maintenance requirements.

Platform virtualization has transformed IT infrastructure management by offering flexibility, scalability, and efficiency. It has become an essential technology in data centers, enabling organizations to optimize resource utilization, improve agility, and reduce costs.

**Service Oriented Architecture(SOA)**

Service-Oriented Architecture (SOA) is an architectural approach that facilitates the development and integration of software systems by encapsulating functionalities as services. It promotes loose coupling, reusability, and interoperability among different software components and systems, both within an organization and across multiple organizations.

**Key Concepts in Service-Oriented Architecture:**

* **Service:**In SOA, a service is a self-contained unit of functionality that performs a specific task or business function. Services are designed to be independent, modular, and reusable. They expose well-defined interfaces (e.g., using web services standards like SOAP or REST) that allow other systems or components to interact with them.
* **Service Provider:** A service provider is responsible for implementing and maintaining one or more services. It encapsulates the underlying functionality and provides the service to consumers. The service provider determines the behavior, access rules, and quality of service parameters for the service.
* **Service Consumer:**A service consumer is a software component or system that utilizes the services provided by service providers. Consumers interact with services by invoking their interfaces and consuming their functionality. Service consumers may be other services, applications, or end-users.
* **Service Registry:**A service registry is a centralized directory or repository that stores information about available services within an SOA environment. It provides a way for service consumers to discover and locate services based on their functional capabilities, interfaces, and other metadata.
* **Service Composition:** Service composition involves combining multiple services to create higher-level composite services that offer more complex functionality. It allows organizations to assemble services in a modular fashion to meet specific business requirements. Service composition is typically achieved through orchestration or choreography mechanisms.

**Advantages of Service-Oriented Architecture:**

* **Modularity and Reusability:**Services in an SOA are designed to be modular and self-contained. This modularity allows for better reusability of services across different applications, reducing development time and effort.
* **Interoperability:** SOA promotes interoperability by defining standard interfaces and protocols for service communication. This enables services developed using different technologies and platforms to seamlessly interact with each other.
* **Scalability and Flexibility:** SOA allows for the scalable deployment of services. As demand grows, additional instances of a service can be deployed, or new services can be added without disrupting existing systems. This flexibility enables organizations to adapt to changing business needs.
* **Improved Agility:** SOA enables organizations to respond quickly to business changes and market demands by leveraging existing services to create new composite services. It enhances agility by facilitating service reuse and promoting a more adaptable IT infrastructure.
* **Service Lifecycle Management:**SOA provides a framework for managing the entire lifecycle of services, from design and development to retirement. This includes aspects such as service versioning, monitoring, security, and governance.

SOA has become a widely adopted architectural style in enterprise computing, enabling organizations to build flexible and scalable systems that can adapt to evolving business needs. It facilitates integration, collaboration, and agility, allowing organizations to leverage existing assets while embracing new technologies and innovations.

Session 2: Benefits and Types of Cloud Computing

## Benefits of Cloud Computing

Cloud computing offers numerous benefits to organizations and individuals. Here are some key advantages of cloud computing:

* **Cost Savings:** Cloud computing eliminates the need for upfront investments in hardware and infrastructure. Instead of purchasing and maintaining physical servers, businesses can pay for cloud services on a pay-as-you-go basis. This reduces capital expenses and allows organizations to convert IT costs into operational expenses, leading to significant cost savings.
* **Scalability and Flexibility:** Cloud services provide the ability to scale resources up or down based on demand. Whether it's increasing computing power during peak periods or reducing resources during off-peak times, cloud computing offers the flexibility to match resource allocation with actual needs. This scalability allows businesses to respond quickly to changing requirements and handle fluctuating workloads efficiently.
* **Accessibility and Mobility:**Cloud computing enables users to access their data and applications from anywhere, using any internet-connected device. This accessibility promotes remote work and collaboration, allowing individuals and teams to work seamlessly across different locations. It also facilitates easy sharing and synchronization of files and information, enhancing productivity and mobility.
* **Reliability and Redundancy:** Cloud service providers maintain multiple data centers and employ redundant infrastructure to ensure high availability. This redundancy minimizes the risk of service disruptions and data loss. Cloud providers typically have robust disaster recovery mechanisms in place, ensuring that data and applications remain accessible even in the event of hardware failures or natural disasters.
* **Security and Data Protection:** Cloud providers invest heavily in security measures and employ industry-leading practices to protect customer data. They implement advanced encryption techniques, access controls, and monitoring systems to ensure the security and privacy of data. Cloud services often offer built-in backup and recovery options, enhancing data protection and disaster recovery capabilities.
* **Automatic Software Updates:** Cloud providers handle software updates and maintenance tasks, relieving organizations from the burden of managing and applying patches and updates themselves. This allows businesses to focus on core activities rather than IT maintenance, ensuring that they have access to the latest features and security enhancements without interrupting operations.
* **Collaboration and Sharing:** Cloud computing facilitates easy collaboration and file sharing among individuals and teams. Multiple users can access and work on the same document simultaneously, enabling real-time collaboration and reducing version control issues. This enhances teamwork and productivity, particularly for geographically dispersed teams.
* **Innovation and Time-to-Market:** Cloud computing provides a platform for rapid application development and deployment. It offers a wide range of pre-built services and APIs that developers can leverage to build and scale applications quickly. This accelerates the time-to-market for new products and services, enabling businesses to stay ahead of the competition and promote innovation.

Cloud computing has revolutionized the IT landscape, offering numerous benefits that drive efficiency, agility, and cost savings. By leveraging cloud services, organizations can focus on their core competencies, scale resources as needed, and access the latest technologies without the upfront investment and maintenance associated with traditional infrastructure.

## Service Models in Cloud Computing Environment

Cloud computing has revolutionized the way organizations consume and deliver IT services. It offers a range of service models that cater to different needs and requirements. Understanding these service models is essential for businesses and individuals looking to leverage the cloud effectively. In this article, we will explore the three primary service models in cloud computing: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

### Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) is the foundation of cloud computing, providing virtualized computing resources over the internet. With IaaS, organizations can outsource the infrastructure needed to support their applications and systems, including servers, storage, and networking components. Key points to consider about IaaS include:

* **Flexibility and Scalability:** IaaS offers on-demand access to computing resources, allowing businesses to scale up or down based on their requirements. This flexibility eliminates the need to invest in and manage physical infrastructure.
* **Resource Management:**IaaS providers handle the management and maintenance of hardware infrastructure, including server provisioning, storage allocation, and network configuration. Users have control over the virtual machines and operating systems running on the infrastructure.
* **Cost Efficiency:**With IaaS, organizations only pay for the resources they consume, enabling cost optimization. It eliminates the need for upfront capital investments, as well as ongoing hardware maintenance and replacement costs.

#### **Examples of IaaS:**

Examples of Infrastructure as a Service (IaaS) providers include:

* **Amazon Web Services (AWS):** AWS offers a wide range of IaaS services, such as Amazon Elastic Compute Cloud (EC2) for virtual servers, Amazon Simple Storage Service (S3) for storage, and Amazon Virtual Private Cloud (VPC) for networking.
* **Microsoft Azure:** Azure provides IaaS capabilities through services like Azure Virtual Machines, Azure Storage, and Azure Virtual Network. It allows users to create and manage virtual machines, storage resources, and networking infrastructure.
* **Google Cloud Platform (GCP):** GCP offers IaaS solutions like Google Compute Engine for virtual machines, Google Cloud Storage for storage, and Google Virtual Private Cloud (VPC) for networking. It provides scalable infrastructure resources for computing and storage needs.
* **IBM Cloud:**IBM Cloud provides IaaS offerings, including IBM Virtual Servers for virtual machine instances, IBM Cloud Object Storage for scalable object storage, and IBM Virtual Private Cloud (VPC) for network isolation and security.
* **Oracle Cloud Infrastructure (OCI):** OCI offers IaaS services such as Oracle Compute for virtual machine instances, Oracle Object Storage for scalable storage, and Oracle Virtual Cloud Network (VCN) for network configuration and management.
* **DigitalOcean:**DigitalOcean is a popular IaaS provider known for its simplicity and developer-friendly approach. It offers virtual private servers (Droplets) with various configurations, block storage, and networking capabilities.

These are just a few examples of IaaS providers, and each offers a range of services to help users provision and manage their infrastructure needs without the burden of physical hardware management.

### Platform as a Service (PaaS)

Platform as a Service (PaaS) builds upon IaaS by providing a complete platform for developing, deploying, and managing applications. PaaS offers a development environment in the cloud, enabling developers to focus on application logic without worrying about the underlying infrastructure. Key points to consider about PaaS include:

* **Application Development:** PaaS provides a platform with preconfigured development tools, libraries, and frameworks, allowing developers to build applications more efficiently. It supports the entire application lifecycle, from development to testing, deployment, and scalability.
* **Automatic Scaling:** PaaS platforms typically offer automatic scaling capabilities, adjusting the computing resources based on application demand. This ensures optimal performance during peak periods and cost savings during low-traffic times.
* **Collaboration and Integration:**PaaS facilitates collaboration among development teams by providing shared development environments, version control, and team management features. It also enables integration with other cloud services and APIs, simplifying the development of connected and hybrid applications.

#### **Examples of PaaS:**

Examples of Platform as a Service (PaaS) providers include:

* **Heroku:**Heroku is a cloud-based PaaS platform that simplifies application deployment and management. It supports various programming languages and frameworks, such as Ruby, Python, Node.js, and Java, and provides services like automatic scaling, database management, and integrated logging.
* **Microsoft Azure App Service:**Azure App Service is a PaaS offering by Microsoft that enables developers to build, deploy, and scale web, mobile, and API applications. It supports multiple programming languages, provides integration with various Azure services, and offers features like auto-scaling, continuous deployment, and application insights.
* **Google App Engine:**Google App Engine is a fully managed PaaS platform that allows developers to build and deploy applications on Google's infrastructure. It supports multiple programming languages, provides automatic scaling, load balancing, and integrates with other Google Cloud services.
* **AWS Elastic Beanstalk:**Elastic Beanstalk is a PaaS service offered by Amazon Web Services (AWS) that simplifies the deployment and management of applications. It supports multiple programming languages and frameworks, handles infrastructure provisioning and configuration, and offers automated scaling and monitoring capabilities.
* **Salesforce Platform:**Salesforce Platform is a PaaS offering that provides a development environment for building custom enterprise applications. It includes tools and services for creating and integrating applications, managing data, and building user interfaces, all within the Salesforce ecosystem.
* **IBM Cloud Foundry:**IBM Cloud Foundry is a PaaS solution that allows developers to build, deploy, and scale applications quickly. It supports multiple programming languages, provides built-in services for data management, caching, and messaging, and integrates with other IBM Cloud services.

These are just a few examples of PaaS providers, and each offers a range of services and features to support developers in building and deploying applications without the need to manage the underlying infrastructure.

### Software as a Service (SaaS)

Software as a Service (SaaS) is a cloud computing model that delivers software applications over the internet on a subscription basis. With SaaS, users can access and use applications hosted in the cloud without the need for installation or maintenance. Key points to consider about SaaS include:

* **Accessibility and Convenience:**SaaS applications are accessible from any internet-connected device, eliminating the need for local installations. Users can access the applications through web browsers or dedicated client applications, making them highly convenient and platform-independent.
* **Maintenance and Upgrades:**SaaS providers handle all software maintenance, including updates, bug fixes, and security patches. This relieves users from the burden of managing software versions and ensures they have access to the latest features and enhancements.
* **Cost-Effectiveness:**SaaS follows a subscription-based pricing model, where users pay for the software on a per-user or per-usage basis. This eliminates the need for upfront software license fees and reduces maintenance and support costs, making it a cost-effective option for businesses.

#### **Examples of SaaS:**

Examples of Software as a Service (SaaS) applications include:

* **Salesforce:**Salesforce is a popular SaaS platform that offers customer relationship management (CRM) software. It provides organizations with tools for sales, marketing, customer service, and analytics, accessible through a web browser or mobile app.
* **Google Workspace:**Google Workspace (formerly G Suite) is a suite of SaaS applications that includes Gmail, Google Docs, Google Sheets, Google Slides, and more. It provides cloud-based productivity and collaboration tools for email, document editing, file storage, and real-time collaboration.
* **Microsoft 365:**Microsoft 365 (previously Office 365) is a suite of SaaS applications that includes Microsoft Word, Excel, PowerPoint, Outlook, and other productivity tools. It offers cloud-based document creation, collaboration, email, and communication services.
* **Dropbox:**Dropbox is a cloud-based file hosting and synchronization service. It allows users to store and share files across multiple devices and collaborate with others by providing access to files and folders through a web interface or desktop/mobile applications.
* **Slack:**Slack is a team collaboration platform that offers real-time messaging, file sharing, and project management features. It enables teams to communicate, collaborate, and organize their work in channels and provides integration with various third-party applications.
* **Zoom:**Zoom is a cloud-based video conferencing and communication platform. It allows users to conduct virtual meetings, webinars, and online events with features like screen sharing, chat, and recording.
* **Adobe Creative Cloud:**Adobe Creative Cloud is a suite of SaaS applications for creative professionals. It includes tools like Photoshop, Illustrator, InDesign, Premiere Pro, and more, offering cloud-based access to industry-standard software for graphic design, video editing, and multimedia production.

These are just a few examples of SaaS applications, and there is a wide range of SaaS offerings available across various industries and use cases, providing software functionality and services accessible through the internet without the need for local installation or management.

## Types of Cloud Computing Environment

Cloud computing offers different types of cloud environments that organizations can choose from based on their specific needs and requirements. Each type has its own characteristics, advantages, and disadvantages. In this article, we will delve into the four primary types of cloud environments: Public Cloud, Private Cloud, Hybrid Cloud, and Multi-Cloud.

### Public Cloud

Public cloud refers to a type of cloud computing deployment model in which cloud services and infrastructure are provided and managed by third-party service providers and made available to the general public over the internet. In a public cloud, multiple organizations or users share the same pool of computing resources, such as virtual machines, storage, and networking infrastructure.

#### **Characteristics of Public Cloud:**

* **Shared Infrastructure:** Resources such as servers, storage, and networks are shared among multiple organizations and users.
* **Scalability:**Public cloud services offer the ability to scale resources up or down as per demand, allowing organizations to pay for what they use.
* **Cost Efficiency:**Users pay for cloud services on a pay-as-you-go basis, avoiding upfront investments in infrastructure and maintenance costs.

#### **Advantages of Public Cloud:**

* **Cost Savings:**Public cloud eliminates the need for upfront capital expenditures, as organizations only pay for the resources and services they use.
* **Accessibility:**Public cloud services are easily accessible from anywhere with an internet connection, promoting remote work and collaboration.
* **Maintenance and Updates:**Service providers handle infrastructure maintenance, security updates, and software upgrades, relieving organizations from these tasks.

#### **Disadvantages of Public Cloud:**

* **Security and Privacy:**Data stored in the public cloud may raise concerns about security and privacy, especially for sensitive information or regulated industries.
* **Limited Customization:**Public cloud services may have limitations on customization and configuration options, as they are designed to serve a broad user base.

### Private Cloud

A private cloud is a computing environment that is dedicated to a single organization or entity. It is designed to offer the benefits of cloud computing, such as scalability, flexibility, and self-service provisioning while providing a higher level of control, security, and privacy compared to public cloud offerings. In a private cloud, the infrastructure and resources are owned and operated by the organization itself or by a third-party provider exclusively serving that organization.

#### **Characteristics of Private Cloud:**

* **Dedicated Infrastructure:** Resources are solely used by a single organization, providing higher levels of control, security, and customization.
* **Data Isolation:**Private cloud ensures that data is stored and processed within the organization's boundaries, enhancing security and compliance.

**Advantages of Private Cloud:**

* **Enhanced Security:**Private cloud offers greater control over data security and compliance, making it suitable for organizations with stringent regulatory requirements.
* **Customization and Control:**Organizations have full control over infrastructure, allowing for customization and tailored configurations based on specific needs.
* **Performance:**Private cloud provides predictable performance and low-latency connections, as resources are not shared with other organizations.

#### **Disadvantages of Private Cloud:**

* **Higher Costs:**Private cloud requires upfront investments in hardware, infrastructure, and maintenance, leading to higher costs compared to public cloud.
* **Limited Scalability:**Private cloud may have limited scalability compared to public cloud, as organizations are responsible for provisioning and managing resources.

### Community Cloud

A community cloud is a type of cloud computing deployment model that is shared by multiple organizations or entities with common interests or requirements. It is designed to provide a collaborative and shared computing environment while addressing the specific needs and concerns of a particular community. In a community cloud, the infrastructure, applications, and resources are accessible and used by multiple organizations that belong to the same community.

#### **Characteristics of Community Cloud:**

* **Shared Infrastructure:**Multiple organizations within a specific community share computing resources and infrastructure provided by a common cloud service provider.
* **Common Requirements:**Community Clouds are designed to meet the unique needs, regulations, and compliance standards of a specific industry or community.
* **Collaborative Environment:**Community Clouds foster collaboration and information sharing among organizations within the community.

#### **Advantages of Community Cloud:**

* **Customized Services:**A Community Cloud is specifically designed to address the requirements and challenges of a particular industry or community, ensuring that the services and infrastructure provided are tailored to their needs.
* **Cost Sharing:**By sharing infrastructure and resources, organizations within the community can achieve cost savings compared to building and managing their own dedicated infrastructure.
* **Enhanced Collaboration:**Community Clouds promote collaboration and knowledge sharing among organizations within the community, leading to improved efficiency and innovation.

#### **Disadvantages of Community Cloud:**

* **Limited Scalability:**Community Clouds may have limitations on scalability compared to public or hybrid clouds since the infrastructure is shared among a specific community, which could affect resource availability during peak demands.
* **Dependence on Community Trust:**As organizations within the community share the same infrastructure, there is a dependency on trust among community members to ensure data security, compliance, and privacy.

### Hybrid Cloud

Hybrid cloud is a cloud computing environment that combines the use of both public and private cloud services, allowing organizations to leverage the benefits of both models. It enables seamless integration and orchestration between public and private clouds, creating a unified infrastructure that offers flexibility, scalability, and control. In a hybrid cloud, applications, data, and workloads can be dynamically distributed between public and private cloud resources based on specific requirements.

#### **Characteristics of Hybrid Cloud:**

* **Integration:**Hybrid cloud enables seamless integration between public and private cloud environments, providing flexibility and scalability.
* **Data Mobility:**Organizations can move workloads and data between public and private clouds based on requirements and cost-effectiveness.

#### **Advantages of Hybrid Cloud:**

* **Flexibility:**Hybrid cloud allows organizations to choose the most suitable environment for each workload, optimizing cost, security, and performance.
* **Scalability:**Hybrid cloud provides the ability to scale resources by leveraging the public cloud during peak periods while maintaining critical data in a private cloud.

#### **Disadvantages of Hybrid Cloud:**

* **Complexity:**Managing and integrating multiple cloud environments can be complex, requiring expertise in hybrid cloud architecture and configuration.
* **Data Governance:**Data governance and security policies must be carefully implemented to ensure consistent protection and compliance across both cloud environments.

Week 2

Session 1: Introduction to GCP

## Overview of Google Cloud Platform (GCP)

Google Cloud Platform (GCP) is a comprehensive suite of cloud computing services provided by Google. It offers a wide range of infrastructure, platforms, and software services that enable businesses to build, deploy, and scale applications and services effectively. GCP leverages the same infrastructure and technology that powers Google's own products, including Google Search, YouTube, and Gmail.

Here's a detailed overview of the key components and services offered by the Google Cloud Platform:

* **Compute: GCP provides several options for computing resources:**
  + **Google Compute Engine:**It offers virtual machines (VMs) in the cloud, allowing you to run applications on Google's infrastructure.
  + **Google Compute Engine:**It offers virtual machines (VMs) in the cloud, allowing you to run applications on Google's infrastructure.
  + **App Engine:** A fully managed platform for building and hosting web applications without worrying about infrastructure management.
  + **Cloud Functions:** A serverless compute platform that lets you run event-driven functions in response to triggers.
* **Storage and Databases: GCP offers a variety of storage and database services**:
  + **Cloud Storage:** A scalable and durable object storage service for storing and accessing data.
  + **Cloud SQL:** Managed MySQL and PostgreSQL databases with automatic backups, replication, and scaling.
  + **Cloud Spanner:**A horizontally scalable, globally distributed relational database service.
  + **Cloud Firestore:** A NoSQL document database for web, mobile, and server applications.
  + **Bigtable:** A fully managed, massively scalable NoSQL database for large analytical and operational workloads.
* **Networking: GCP provides robust networking capabilities:**
  + **Virtual Private Cloud (VPC):**A logically isolated virtual network that enables you to define subnets, firewalls, and routing.
  + **Load Balancing:** GCP offers regional and global load balancers to distribute traffic across instances and services.
  + **Cloud CDN:** A content delivery network that caches and delivers content closer to end-users, reducing latency.
  + **Cloud Interconnect:** You can connect your on-premises network to GCP through dedicated or partner interconnects.
* **Big Data and Machine Learning: GCP offers various services for data processing and analysis:**
  + **BigQuery:** A fully managed data warehouse for running fast, SQL-like queries on large datasets.
  + **Cloud Dataflow:**A serverless, fully managed service for processing and analyzing streaming and batch data.
  + **Cloud Dataproc:**Managed Apache Spark and Hadoop service for large-scale data processing.
  + **Cloud Pub/Sub:**A messaging service for building event-driven systems and real-time analytics.
  + **AI Platform:**Provides tools and frameworks for building, training, and deploying machine learning models.
* **Identity and Security: GCP offers robust security features:**
  + **Cloud Identity and Access Management (IAM):**Enables you to manage access to resources and control permissions.
  + **Cloud Security Scanner:**Scans web applications for vulnerabilities and provides security recommendations.
  + **Cloud Armor:**Provides distributed denial of service (DDoS) protection and web application firewall (WAF) capabilities.
  + **Cloud Key Management Service (KMS):**A centralized key management system to create, import, and manage cryptographic keys.
* **Management Tools: GCP provides several tools for managing and monitoring your resources:**
  + **Cloud Console:**A web-based interface for managing your GCP resources, including monitoring and logging.
  + **Stackdriver:**A suite of monitoring, logging, and diagnostics tools for cloud-based applications and infrastructure.
  + **Cloud Deployment Manager:**Allows you to define, deploy, and manage complex GCP resources using templates.
  + **Cloud Billing:**Provides detailed usage and billing reports, budget alerts, and cost optimization recommendations.

## Setting up a GCP Account

The first step is to create a [Google Cloud console](https://console.cloud.google.com/) account. This section describes how to create a Google Cloud account.

#### **To create a Google Cloud account:**

* Open the [Google Cloud console](https://console.cloud.google.com/) in a browser.
* When prompted to sign in, create a new account by clicking **Create Account.**
* Follow the instructions to register your corporate email address as a [Google account](https://support.google.com/accounts/answer/27441). Alternatively, you can use a Gmail account or another Google account.
* Continue to the [Google Cloud console](https://console.cloud.google.com/) and accept the Google Cloud terms presented.

#### **Set up Cloud Identity**

If you'd prefer to use Cloud Identity to manage your organizational identity with Apigee, please take note of the following:

* General information on Cloud Identity can be found in [What is Cloud Identity](https://support.google.com/cloudidentity/answer/7319251) or in the [Cloud Identity Help Center](https://support.google.com/cloudidentity).
* Cloud Identity provides multiple ways to create users from your organization in bulk, including sync tools with your [LDAP server](https://support.google.com/cloudidentity/answer/106368) and the ability to configure [SSO](https://support.google.com/cloudidentity/topic/7558768).
* Using Cloud Identity with Google Cloud provides you with a Google Cloud Organization: this allows for more advanced resource hierarchy configuration. Although not needed for Apigee hybrid, it might be useful if you plan to use other Google Cloud services or have multiple members of your organization access the product.
* To use Cloud Identity, you must [verify ownership of the domain](https://support.google.com/cloudidentity/answer/7331243) that you plan to use.

## Understanding GCP console and Cloud Shell

#### **Here are a few things you need to know about the Google Cloud Console:**

* The Cloud Console is under continuous development, so occasionally the graphical layout changes. This is most often to accommodate new Google Cloud features or changes in the technology, resulting in a slightly different workflow.
* You can perform most common Google Cloud actions in the Cloud Console, but not all actions. In particular, very new technologies or sometimes detailed API or command options are not implemented (or not yet implemented) in the Cloud Console. In these cases, the command line or the API is the best alternative.
* The Cloud Console is extremely fast for some activities. The Cloud Console can perform multiple actions on your behalf that might require many CLI commands. It can also perform repetitive actions. In a few clicks you can accomplish activities that would require a lot of typing and would be susceptible to typing errors.
* The Cloud Console is able to reduce errors by offering only valid options through its menus. It can leverage access to the platform "behind the scenes" through the SDK to validate configuration before submitting changes. A command line can't do this kind of dynamic validation.

#### **Now, let's have a look at the Cloud Shell**

Cloud Shell is an interactive shell environment for Google Cloud that lets you learn and experiment with Google Cloud and manage your projects and resources from your web browser.

With Cloud Shell, the [Google Cloud CLI](https://cloud.google.com/sdk/gcloud) and other utilities you need are pre-installed, fully authenticated, up-to-date, and always available when you need them. Cloud Shell comes with a built-in code editor with an integrated Cloud Code experience, allowing you to develop, build, debug, and deploy your cloud-based apps entirely in the cloud. You can also launch interactive tutorials, open cloned repositories, and preview web apps on a Cloud Shell virtual machine instance.

#### **How Cloud Shell Works**

Cloud Shell provisions a Compute Engine virtual machine running a Debian-based Linux operating system for your temporary use. This virtual machine is owned and managed by Google Cloud, so will not appear within any of your GCP projects.

Cloud Shell instances are provisioned on a per-user, per-session basis. The instance persists while your Cloud Shell session is active; after an hour of inactivity, your session terminates and its VM is discarded. For more on usage quotas, refer to the [limitations guide](https://cloud.google.com/shell/docs/limitations#usage_limits).

With the default Cloud Shell experience, you are allocated with an ephemeral, pre-configured VM and the environment you work with is a Docker container running on that VM. You can also customize your environment automatically on VM boot to ensure that your Cloud Shell instance includes your preferred tools.

## Projects, Billings, and IAM basics in GCP

### Projects in GCP

#### **What is a Project?**

A project serves as a central hub for organizing your Google Cloud resources. In Google Cloud Storage, all data is associated with a project. A project encompasses a group of users, a set of APIs, and configurations for billing, authentication, and monitoring of those APIs. For instance, your Cloud Storage buckets and objects, along with user permissions, are housed within a project. You have the flexibility to create a single project or multiple projects to categorize and manage your Google Cloud resources, including your Cloud Storage data, in a structured manner.

#### **Projects and Permissions**

For each project, you use [Identity and Access Management (IAM)](https://cloud.google.com/storage/docs/access-control/iam) to grant the ability to manage and work on your project. When you grant an IAM *role* to a *principal*, such as a Google Account, that principal obtains certain *permissions* that allow them to perform actions. When you grant a role at the project level, the access provided by the role applies to every bucket and object within the project. Alternatively, when you grant a role for an individual bucket, the access provided by the role is limited to just that bucket and the objects the bucket contains.

For a list of available roles that apply to Cloud Storage, as well as a discussion about how a special set of roles, called *basic roles*, apply to Cloud Storage, see [Cloud Storage IAM roles](https://cloud.google.com/storage/docs/access-control/iam-roles).

### Identity and Access Management (IAM)

IAM (Identity and Access Management) provides you with the means to manage access to your Google Cloud project's resources. These resources encompass Cloud Storage buckets and the objects stored within them, as well as other entities like [Compute Engine instances](https://cloud.google.com/compute).

In IAM, principals define the "who" and can be individuals, groups, domains, or even the general public. Principals are assigned [roles](https://cloud.google.com/storage/docs/access-control/iam#roles), which grant them the ability to perform actions within Cloud Storage and Google Cloud as a whole. Roles are comprised of one or more [permissions](https://cloud.google.com/storage/docs/access-control/iam#permissions), which are the fundamental units of IAM. Each permission enables a specific action.

For instance, the ***storage.objects.create*** permission allows users to create objects. This permission can be found in roles like **Storage Object Creator**, which provides the necessary permissions for object creation, or **Storage Object Admin**, which grants a broader range of object-related permissions.

The aggregation of IAM roles assigned to a resource is known as an IAM policy. The access granted by these roles extends to both the resource itself and any nested resources it contains. For example, you can set an IAM policy on a bucket to grant a user administrative control over the bucket and its objects. Additionally, you can set an IAM policy at the project level, enabling another user to view objects in any bucket within that project.

In case you have a Google Cloud [organization resource](https://cloud.google.com/resource-manager/docs/cloud-platform-resource-hierarchy#organizations), you can utilize[IAM deny policies](https://cloud.google.com/iam/docs/deny-overview)to prohibit access to resources. When a deny policy is attached to a resource, the principal specified in the policy is unable to use the designated permission to access the resource or any sub-resources, regardless of the roles they possess. Deny policies take precedence over any IAM allow policies.

### Billing in Google Cloud Platform

Cloud Billing is a [collection of tools](https://cloud.google.com/billing/docs/concepts) that help you track and understand your Google Cloud spending, pay your bill, and optimize your costs.

A [Cloud Billing account](https://cloud.google.com/billing/docs/how-to/manage-billing-account) defines who pays for a given set of Google Cloud resources. To use Google Cloud services, you must have a valid Cloud Billing account, and must link it to your Google Cloud projects. Your project's Google Cloud usage is charged to the linked Cloud Billing account.

You must have a valid Cloud Billing account even if you are in your [free trial period](https://cloud.google.com/free/docs/gcp-free-tier#free-trial) or if you only use Google Cloud resources that are covered by the [Google Cloud Free Tier](https://cloud.google.com/free/docs/gcp-free-tier#free-tier).

You also need a Cloud Billing account to pay for your use of the [Google Maps Platform APIs](https://developers.google.com/maps/billing/gmp-billing).

## Introduction to GCP Regions and Zones

### Regions and Zones

Compute Engine resources are distributed across various worldwide locations, which consist of regions and zones. A region represents a specific geographic area where you can deploy your resources. Each region is composed of three or more zones. For example, the ***us-west1*** region denotes a region situated on the west coast of the United States, encompassing three zones: ***us-west1-a, us-west1-b, and us-west1-c***.

Zonal resources are the resources that reside within a particular zone, such as [virtual machine instances](https://cloud.google.com/compute/docs/instances)or zonal [persistent disks](https://cloud.google.com/compute/docs/disks). On the other hand, regional resources, like [static external IP addresses](https://www.geeksforgeeks.org/batch/cloud-bootcamp/track/week-2-track/article/static%20external%20IP%20addresses), are accessible to any resource within the same region, regardless of the zone they are in.

To illustrate, if you wish to attach a zonal persistent disk to an instance, both resources must be located in the same zone. Similarly, if you want to assign a static IP address to an instance, the instance must belong to the same region as the static IP address.

By distributing resources across different zones within a region, the risk of infrastructure outages impacting all resources simultaneously is reduced. Placing resources in different regions offers an even higher level of protection against failures. This enables the design of resilient systems with resources distributed across multiple failure domains.

It's important to note that only specific resources are specific to a region or zone. Other resources, such as images, are global resources that can be utilized by any resources across any location. For detailed information on global, regional, and zonal Compute Engine resources, please refer to the documentation on [Global, Regional, and Zonal Resources](https://cloud.google.com/compute/docs/regions-zones/global-regional-zonal-resources).

### Zones and Clusters

Compute Engine introduces a layer of abstraction between zones and the physical clusters where these zones are hosted. A cluster represents a distinct physical infrastructure situated within a data center. Each zone is hosted in one or more clusters, and Compute Engine autonomously maps zones to clusters for each organization. It's important to note that the mapping of zones to clusters can differ between organizations. For instance, the ***us-central1-a*** zone for one organization might not correspond to the same cluster as the ***us-central1-a*** zone for another organization.

The decoupling of zones from clusters brings several benefits to both you and Compute Engine:

* **Resource Balancing:**Compute Engine can effectively balance resources across the clusters within a region, ensuring optimal utilization.
* **Manageable Zone Selection:**As Compute Engine expands its regions by adding more clusters over time, the list of available zones remains manageable for users.

For most organizations, Compute Engine maintains a consistent zone-to-cluster mapping across all projects within the organization. In scenarios where organizations employ **VPC Network Peering** or **Private services access** to share networks or services with other organizations, Compute Engine endeavors to maintain a consistent zone-to-cluster mapping among the peered organizations. However, in the case of large-scale SaaS providers, there might be instances where Compute Engine cannot guarantee a consistent mapping for all peered organizations. Nevertheless, Compute Engine ensures that peered projects have a consistent zone-to-cluster mapping.

To delve deeper into zone-to-cluster mapping and zone virtualization concepts, I recommend referring to the [Zone Virtualization](https://cloud.google.com/compute/docs/regions-zones/zone-virtualization) document, which provides comprehensive information on the topic.

Session 2: Exploring GCP Products and Services

## Compute Services: Compute Engine, App Engine, Kubernetes Engine

### Compute Engine

Compute Engine is a [computing and hosting service](https://cloud.google.com/docs/overview/cloud-platform-services#computing-hosting) that lets you create and run virtual machines on Google infrastructure. Compute Engine offers scale, performance, and value that lets you easily launch large compute clusters on Google's infrastructure. There are no upfront investments, and you can run thousands of virtual CPUs on a system that offers quick, consistent performance.

Compute Engine instances can run the [public images](https://cloud.google.com/compute/docs/images) for Linux and Windows Server that Google provides as well as private custom images that you can [create](https://cloud.google.com/compute/docs/images/create-delete-deprecate-private-images) or [import from your existing systems](https://cloud.google.com/compute/docs/import/importing-virtual-disks). You can also [deploy Docker containers](https://cloud.google.com/compute/docs/containers/deploying-containers), which are automatically launched on instances running the [Container-Optimized OS](https://cloud.google.com/container-optimized-os/docs) public image.

You can choose the machine properties of your instances, such as the number of virtual CPUs and the amount of memory, by using a set of [predefined machine types](https://cloud.google.com/compute/docs/machine-types#predefined_machine_types) or by creating your own [custom machine types](https://cloud.google.com/compute/docs/instances/creating-instance-with-custom-machine-type).

#### **Instances and Projects**

Each instance belongs to a [Google Cloud console](https://console.cloud.google.com/) project, and a project can have one or more instances. When you create an instance in a project, you specify the zone, operating system, and machine type of that instance. When you delete an instance, it is removed from the project.

#### **Instances and Storage options**

By default, each Compute Engine instance has a small [boot persistent disk](https://cloud.google.com/compute/docs/disks/create-root-persistent-disks) that contains the operating system. When applications running on your instance require more storage space, you can add additional [storage options](https://cloud.google.com/compute/docs/disks) to your instance.

#### **Instances and Networks**

Each network interface of a Compute Engine instance is associated with a subnet of a unique VPC network. For more information about VPCs, see [Network overview](https://cloud.google.com/compute/docs/networking/network-overview) and [VPC quotas](https://cloud.google.com/vpc/docs/quota).

#### **Instances and Containers**

Compute Engine instances support a declarative method for launching your applications using [containers](https://cloud.google.com/containers). When creating a VM or an instance template, you can provide a Docker image name and launch configuration. Compute Engine will take care of the rest including supplying an up-to-date [Container-Optimized OS](https://cloud.google.com/container-optimized-os/docs) image with Docker installed and launching your container when the VM starts up. See [Deploying containers on VMs and managed instance groups (MIGs)](https://cloud.google.com/compute/docs/containers/deploying-containers) for more information.

### App Engine

App Engine is a fully managed, serverless platform for developing and hosting web applications at scale. You can choose from several popular languages, libraries, and frameworks to develop your apps, and then let App Engine take care of provisioning servers and scaling your app instances based on demand.

App Engine varies from language to language.

To check the documentation for your preferred language, [**click here**](https://cloud.google.com/appengine/docs).

### Kubernetes Engine

Google Kubernetes Engine (GKE) is a managed [Kubernetes](https://kubernetes.io/) service that you can use to deploy and operate containerized applications at scale using Google's infrastructure.

GKE is a Google-managed implementation of the [Kubernetes](https://kubernetes.io/) open-source container orchestration platform. Kubernetes was developed by Google, drawing on years of experience operating production workloads at scale on [Borg](https://research.google/pubs/pub43438/), our in-house cluster management system.

#### **When to use GKE?**

GKE is ideal if you need a platform that lets you configure the infrastructure that runs your containerized apps, such as networking, scaling, hardware, and security. GKE provides the operational power of Kubernetes while managing many of the underlying components, such as the control plane and nodes, for you.

#### **Benefits of GKE**

**The following are the benefits of GKE:**

* **Platform Management:**
  + Fully-managed nodes in GKE [Autopilot mode](https://cloud.google.com/kubernetes-engine/docs/concepts/autopilot-overview) with built-in hardening and best practice configurations automatically applied.
  + Managed upgrade experience with [release channels](https://cloud.google.com/kubernetes-engine/docs/concepts/release-channels) to improve security, reliability, and compliance.
  + Flexible [maintenance windows and exclusions](https://cloud.google.com/kubernetes-engine/docs/concepts/maintenance-windows-and-exclusions) that let you configure upgrade type and scope to meet business needs and architecture constraints.
  + In GKE Standard mode, flexible [node upgrade strategies](https://cloud.google.com/kubernetes-engine/docs/concepts/node-pool-upgrade-strategies) optimize availability and manage disruptions.
  + Automatic scaling of nodes based on the number of Pods in the cluster with Autopilot mode or with node auto-provisioning in Standard mode.
  + [Node auto-repair](https://cloud.google.com/kubernetes-engine/docs/concepts/node-auto-repair) to maintain node health and availability.
  + Built-in [logging and monitoring](https://cloud.google.com/monitoring/kubernetes-engine).
  + Google Cloud integrated CI/CD options with [Cloud Build](https://cloud.google.com/build) and [Cloud Deploy](https://cloud.google.com/deploy).
* **Improved Security Posture:**
  + Hardened node operating system for apps: [Container-Optimized OS](https://cloud.google.com/container-optimized-os/docs/concepts/security).
  + [Built-in security measures](https://cloud.google.com/kubernetes-engine/docs/concepts/autopilot-security).
  + [Automatic upgrades](https://cloud.google.com/kubernetes-engine/docs/concepts/release-channels) to new GKE versions.
  + Integrated security posture monitoring tooling with the [security posture dashboard](https://cloud.google.com/kubernetes-engine/docs/concepts/about-security-posture-dashboard).
  + Google Cloud logging and monitoring integrations with [Google Cloud's operations suite](https://cloud.google.com/stackdriver/docs/solutions/gke).
* **Cost Optimization:**
  + In Autopilot mode, pay only for the compute resources your running Pods request.
  + In GKE Standard mode, you pay for all resources on nodes, regardless of Pod requests.
  + Save costs by running fault-tolerant workloads, such as batch jobs, on [Spot Pods](https://cloud.google.com/kubernetes-engine/docs/how-to/autopilot-spot-pods).
  + Minimized operational overhead in Autopilot mode because Google manages both the nodes and the control plane.
* **Reliability and Availability:**
  + >99% [monthly uptime SLO](https://cloud.google.com/kubernetes-engine/sla).
  + Pod-level SLA in Autopilot clusters because Google manages the nodes.
  + The highly-available control plane and worker nodes in [Autopilot mode](https://cloud.google.com/kubernetes-engine/docs/concepts/autopilot-overview) and in [regional Standard clusters](https://cloud.google.com/kubernetes-engine/docs/concepts/regional-clusters).
  + [Proactive monitoring and recommendations](https://cloud.google.com/kubernetes-engine/docs/deprecations) to mitigate potential workload disruptions caused by upcoming deprecations.
  + Multi-cluster Service capabilities.

#### **Use cases for GKE**

GKE and Kubernetes are used in a variety of industries, including robotics, healthcare, retail, education, gaming, and financial services. The range of applications includes AI and ML operations, data processing at scale, operating scalable online games platforms, and running reliable applications under heavy load.

For case studies by industry and application, refer to [Google Cloud customers](https://cloud.google.com/customers#more-customer-stories).

#### **How GKE Works**

A GKE environment consists of *nodes*, which are [Compute Engine virtual machines (VMs)](https://cloud.google.com/compute), that are grouped together to form a *cluster*. You package your apps (also called *workloads*) into containers. You deploy sets of containers as *Pods* to your nodes. You use the Kubernetes API to interact with your workloads, including administering, scaling, and monitoring.

Kubernetes clusters have a set of management nodes called the *control plane*, which runs system components such as the Kubernetes API server. In GKE, Google manages the control plane and system components for you. In Autopilot mode, which is the recommended way to run GKE, Google also manages your worker nodes. Google automatically upgrades component versions for improved stability and security, ensuring high availability, and ensuring the integrity of data stored in the cluster's persistent storage.

## Storage services: Cloud Storage, Persistent Disk

### Cloud Storage

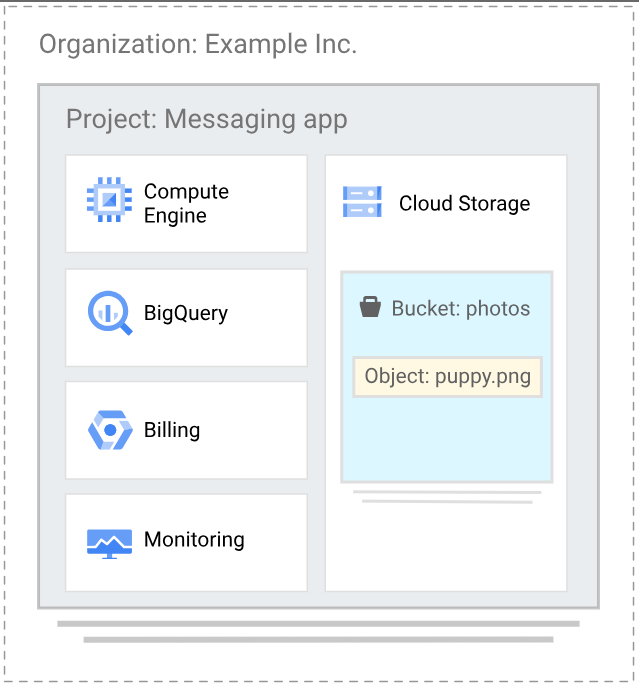
Cloud Storage allows worldwide storage and retrieval of any amount of data at any time. You can use Cloud Storage for a range of scenarios including serving website content, storing data for archival and disaster recovery, or distributing large data objects to users via direct download.

Cloud Storage is a service for storing your [*objects*](https://cloud.google.com/storage/docs/objects) in Google Cloud. An object is an immutable piece of data consisting of a file of any format. You store objects in containers called [*buckets*](https://cloud.google.com/storage/docs/buckets). All buckets are associated with a [*project*](https://cloud.google.com/storage/docs/projects), and you can group your projects under an [*organization*](https://cloud.google.com/resource-manager/docs/cloud-platform-resource-hierarchy#organizations). Each project, bucket, and object in Google Cloud is a resource in Google Cloud, as are things such as [Compute Engine instances](https://cloud.google.com/compute/docs/instances).

After you create a project, you can [create Cloud Storage buckets](https://cloud.google.com/storage/docs/creating-buckets), [upload objects](https://cloud.google.com/storage/docs/uploading-objects) to your buckets, and [download objects](https://cloud.google.com/storage/docs/downloading-objects) from your buckets. You can also grant permissions to make your data accessible to principals you specify, or - for certain use cases such as hosting a website - [accessible to everyone on the public internet](https://cloud.google.com/storage/docs/access-control/making-data-public).

#### **The Google Cloud Hierarchy**

The Cloud Storage structure looks like this:

Google Cloud Hierarchy

Here's how the Cloud Storage structure can apply to a real-world case:

**Organization**: Your company, called Example Inc., creates a Google Cloud organization called exampleinc.org.

**Project**: Example Inc. is building several applications, and each one is associated with a project. Each project has its own set of Cloud Storage APIs, as well as other resources.

**Bucket**: Each project can contain multiple buckets, which are containers to store your objects. For example, you might create a photo bucket for all the image files your app generates and a separate video bucket.

**Object**: An individual file, such as an image called ***puppy.png***.

#### **Basic Tools for Cloud Storage**

Here are some basic ways you can interact with Cloud Storage:

* **Console**: The [Google Cloud console](https://console.cloud.google.com/storage/browser) provides a visual interface for you to manage your data in a browser.
* **Google Cloud CLI**: The [gcloud CLI](https://cloud.google.com/sdk/gcloud" \t "_blank) allows you to interact with Cloud Storage through a terminal using [gcloud storage commands](https://cloud.google.com/sdk/gcloud/reference/storage" \t "_blank).
* **Client libraries**: The Cloud Storage [client libraries](https://cloud.google.com/storage/docs/reference/libraries) allow you to manage your data using one of your preferred languages, including C++, C#, Go, Java, Node.js, PHP, Python, and Ruby.
* **REST APIs**: Manage your data using the [JSON](https://cloud.google.com/storage/docs/json_api) or [XML](https://cloud.google.com/storage/docs/xml-api/overview) API.
* **Terraform**: [Terraform](https://www.terraform.io/) is an infrastructure-as-code (IaC) tool that you can use to provide the infrastructure for Cloud Storage.
* **Cloud Storage FUSE**: [Cloud Storage FUSE](https://cloud.google.com/storage/docs/gcs-fuse) allows you to mount Cloud Storage buckets to your local file system. This enables your applications to read from a bucket or write to a bucket by using standard file system semantics.

### Persistent Disk

Persistent Disk volumes are durable network storage devices that your instances can access like physical disks in a desktop or a server. The data on each Persistent Disk volume is distributed across several physical disks. Compute Engine manages the physical disks and the data distribution for you to ensure redundancy and optimal performance.

Persistent Disk volumes are located independently from your virtual machine (VM) instances, so you can detach or move Persistent Disk volumes to keep your data even after you delete your instances. Persistent Disk performance scales automatically with size, so you can resize your existing Persistent Disk volumes or add more Persistent Disk volumes to a VM to meet your performance and storage space requirements.

#### **Persistent Disk Types**

When you configure a persistent disk, you can select one of the following disk types:

* **Standard persistent disks** (pd-standard)
  + Suitable for large data processing workloads that primarily use sequential I/Os.
  + Backed by standard hard disk drives (HDD).
* **Balanced persistent disks** (pd-balanced)
  + An alternative to performance (pd-ssd) persistent disks
  + Balance of performance and cost. For most VM shapes, except very large ones, these disks have the same maximum IOPS as SSD persistent disks and lower IOPS per GB. This disk type offers performance levels suitable for most general-purpose applications at a price point between that of standard and performance (pd-ssd) persistent disks.
  + Backed by solid-state drives (SSD).
* **Performance (SSD) persistent disks** (pd-ssd)
  + Suitable for enterprise applications and high-performance databases that require lower latency and more IOPS than standard persistent disks provide.
  + Designed for single-digit millisecond latencies; the observed latency is application specific.
  + Backed by solid-state drives (SSD).
* **Extreme persistent disks** (pd-extreme)
  + Offer consistently high performance for both random access workloads and bulk throughput.
  + Designed for high-end database workloads.
  + Allow you to provision the target IOPS.
  + Backed by solid-state drives (SSD).
  + Available with a limited number of [machine types](https://cloud.google.com/compute/docs/disks/extreme-persistent-disk#machine_shape_support).

## Big Data Services: BigQuery, Dataflow

### BigQuery

BigQuery is Google Cloud's fully managed, petabyte-scale, and cost-effective analytics data warehouse that lets you run analytics over vast amounts of data in near real-time. With BigQuery, there's no infrastructure to set up or manage, letting you focus on finding meaningful insights using GoogleSQL and taking advantage of flexible pricing models across on-demand and flat-rate options.

BigQuery maximizes flexibility by separating the compute engine that analyzes your data from your storage choices. You can store and analyze your data within BigQuery or use BigQuery to assess your data where it lives. Federated queries let you read data from external sources while streaming supports continuous data updates. Powerful tools like BigQuery ML and BI Engine let you analyze and understand that data.

BigQuery interfaces include Google Cloud console interface and the BigQuery command-line tool. Developers and data scientists can use client libraries with familiar programming including Python, Java, JavaScript, and Go, as well as BigQuery's REST API and RPC API to transform and manage data. ODBC and JDBC drivers provide interaction with existing applications including third-party tools and utilities.

As a data analyst, data engineer, data warehouse administrator, or data scientist, the BigQuery ML documentation helps you discover, implement, and manage data tools to inform critical business decisions.

#### **Use cases of BigQuery**

* **Migrating data warehouses to BigQuery**
* **Visualizing BigQuery data in a Jupyter notebook**

### Dataflow

Dataflow is a managed service for executing a wide variety of data processing patterns. The documentation on this site shows you how to deploy your batch and streaming data processing pipelines using Dataflow, including directions for using service features.

The Apache Beam SDK is an open source programming model that enables you to develop both batch and streaming pipelines. You create your pipelines with an Apache Beam program and then run them on the Dataflow service. The [Apache Beam documentation](https://beam.apache.org/documentation/) provides in-depth conceptual information and reference material for the Apache Beam programming model, SDKs, and other runners.

#### **Use cases of Dataflow**

* **Building production-ready data pipelines using Dataflow**
* **Detecting anomalies in financial transactions by using AI Platform, Dataflow, and BigQuery**
* **Deploying production-ready log exports to Splunk using Dataflow**

## ****AI and Machine Learning Services****

**The following are some of the famous and most used AI tools for Developers:**

#### **AutoML**

Google Cloud's AutoML is a suite of AutoML products and services provided by Google Cloud Platform (GCP). It offers a range of automated machine-learning tools that enable users to build custom machine-learning models without extensive expertise in data science or programming.

Google Cloud AutoML simplifies the process of training, deploying, and managing machine learning models. It provides various AutoML solutions tailored to specific use cases, including image recognition, natural language processing, translation, and tabular data analysis.

#### **Cloud Natural Language**

Google Cloud's Cloud Natural Language is a cloud-based natural language processing (NLP) service provided by Google Cloud Platform (GCP). It offers a range of powerful tools and APIs for analyzing and understanding text data, extracting insights, and performing various language-related tasks. Cloud Natural Language utilizes machine learning algorithms and models trained on vast amounts of text data to provide advanced NLP capabilities.

#### **Dialogflow**

Google Cloud Dialogflow is a conversational AI platform that allows developers to build and deploy interactive chatbots, virtual agents, and natural language understanding systems. It provides a set of tools and capabilities to create conversational interfaces for various applications, including customer support, virtual assistants, and voice-enabled applications.

Dialogflow utilizes natural language processing (NLP) and machine learning techniques to understand and interpret user input in the form of text or voice. It enables developers to define conversational agents, known as chatbots or agents, that can understand user intents, extract important information, and provide relevant responses.

## Networking Services: VPC and Cloud Load Balancing

### VPC

Google Cloud Virtual Private Cloud (VPC) provides networking functionality to Compute Engine virtual machine (VM) instances, Google Kubernetes Engine (GKE) containers, and the App Engine flexible environment. VPC provides networking for your cloud-based services that is global, scalable, and flexible.

**Use cases for VPC:**

* **Architecting disaster recovery for locality-restricted workloads**
* **Building a multi-cluster service mesh on GKE with shared control-plane, single-VPC architecture**

#### **VPC Networks**

You can think of a VPC network the same way you'd think of a physical network, except that it is virtualized within Google Cloud. A VPC network is a global resource that consists of a list of regional virtual subnetworks (subnets) in data centers, all connected by a global wide area network. VPC networks are logically isolated from each other in Google Cloud.

A VPC network does the following:

* Provides connectivity for your [Compute Engine virtual machine (VM) instances](https://cloud.google.com/compute/docs/instances), including [Google Kubernetes Engine (GKE) clusters](https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-architecture), [App Engine flexible environment](https://cloud.google.com/appengine/docs/flexible) instances, and other Google Cloud products built on Compute Engine VMs.
* Offers built-in internal passthrough Network Load Balancers and proxy systems for internal Application Load Balancers.
* Connects to on-premises networks by using Cloud VPN tunnels and VLAN attachments for Cloud Interconnect.
* Distributes traffic from Google Cloud external load balancers to backends.

### Cloud Load Balancer

Cloud Load Balancing allows you to put your resources behind a single IP address that is externally accessible or internal to your Virtual Private Cloud (VPC) network.

#### **Load Balancer Types:**

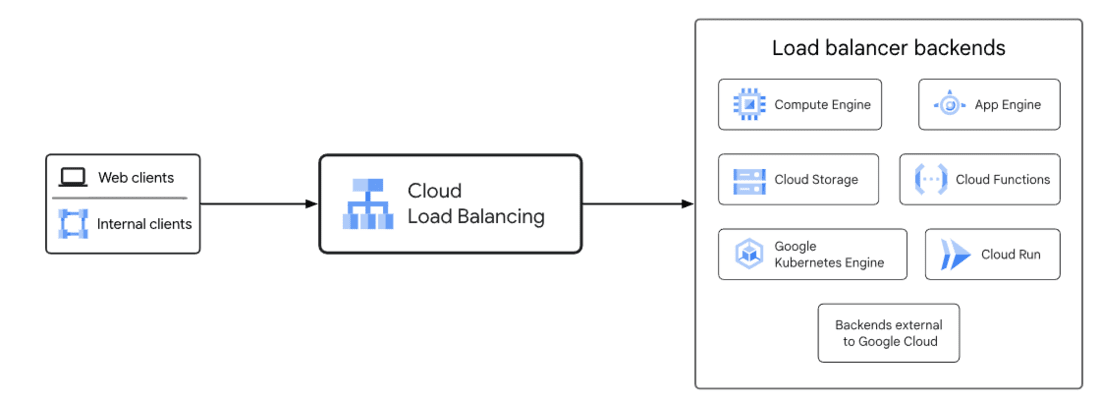
* External Application Load Balancer
* Internal Application Load Balancer
* External passthrough Network Load Balancer
* Internal passthrough Network Load Balancer
* External proxy Network Load Balancer
* Internal proxy Network Load Balancer

#### **Use cases for Cloud Load Balancer:**

* Request routing to a multi-region external Application Load Balancer
* Use UDP with Network Load Balancers
* Faster web performance and improved web protection for load balancing
* Delivering HTTP and HTTPS content over the same published domain
* Optimizing application latency with load balancing
* Cross-region load balancing using Microsoft IIS backends
* Using load balancing for highly available applications

A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues. Google's Cloud Load Balancing is built on reliable, high-performing technologies such as Maglev, Andromeda, Google Front Ends, and Envoy—the same technologies that power Google's own products.

Cloud Load Balancing offers the most comprehensive portfolio of application and network load balancers. Use our global proxy load balancers to distribute millions of requests per second among backends in multiple regions with our Google Front End fleet in over 80 distinct locations worldwide—all with a single, anycast IP address. Implement strong jurisdictional control with our regional proxy load balancers, keeping your backends and proxies in a region of your choice without worrying about TLS/SSL offload. Use our passthrough load balancers to quickly route multiple protocols to backends with the high performance of Direct Server Return.

Cloud Load Balancing Overview

#### **Cloud Load Balancing offers the following load-balancing features:**

* **Single anycast IP address.** With Cloud Load Balancing, a single anycast IP address is the frontend for all of your backend instances in regions around the world. It provides cross-region load balancing, including automatic multi-region failover, which moves traffic to failover backends if your primary backends become unhealthy. Cloud Load Balancing reacts instantaneously to changes in users, traffic, network, backend health, and other related conditions.
* **Software-defined load balancing.** Cloud Load Balancing is a fully distributed, software-defined, managed service for all your traffic. It is not an instance-based or device-based solution, so you won't be locked into a physical load-balancing infrastructure or face the high availability, scale, and management challenges inherent in instance-based load balancers.
* **Seamless autoscaling.** Cloud Load Balancing can scale as your users and traffic grow, including easily handling huge, unexpected, and instantaneous spikes by diverting traffic to other regions in the world that can take traffic. Autoscaling does not require pre-warming: you can scale from zero to full traffic in a matter of seconds.
* **Layer 4 and Layer 7 load balancing.** Use Layer 4-based load balancing to direct traffic based on data from network and transport layer protocols such as [TCP, UDP, ESP, GRE, ICMP, and ICMPv6](https://cloud.google.com/load-balancing/docs/network/networklb-backend-service). Use Layer 7-based load balancing to add request routing decisions based on attributes, such as the HTTP header and the uniform resource identifier.
* **External and internal load balancing.** You can use external load balancing when your users reach your applications from the internet. You can use internal load balancing when your clients are inside of Google Cloud.
* **Global and regional load balancing.** You can distribute your load-balanced resources in single or multiple regions to terminate connections close to your users and to meet your high availability requirements.
* **Advanced feature support.** Cloud Load Balancing supports features such as IPv6 global load balancing, [source IP-based traffic steering](https://cloud.google.com/load-balancing/docs/network/networklb-backend-service#traffic-steering), [weighted load balancing](https://cloud.google.com/load-balancing/docs/network/networklb-backend-service#weighted-lb), WebSockets, user-defined request headers, and protocol forwarding for private virtual IP addresses (VIPs).

Week 3

Session 1: Compute options, Storage options

**Creating VM Instances and Instance Types in Google Cloud Platform's Compute Engine**

Google Cloud Platform's Compute Engine offers a robust infrastructure-as-a-service (IaaS) solution, allowing users to create and manage virtual machine (VM) instances on Google's global infrastructure.

**Creating VM Instances:**

Creating VM instances in Compute Engine is a straightforward process that offers flexibility and customization options. Follow these steps to create a VM instance:

**Step 1: Accessing the Compute Engine Interface**

Navigate to the Google Cloud Console and select the project in which you want to create the VM instance. Open the Compute Engine section from the left-hand navigation menu to access the Compute Engine interface.

**Step 2: Creating a VM Instance**

Click on the "Create" button to start the VM instance creation process. You will be prompted to specify various configuration settings for the instance.

**Step 3: Configuration Settings**

* **Instance Details:** Provide a name for the VM instance, choose the region and zone where it should be located, and select the desired machine type.
* **Boot Disk:** Select the operating system image for the VM instance. You can choose from a range of pre-configured images provided by Google or use your own custom image.
* **Networking:** Configure network settings, including specifying the network, subnet, and firewall rules for the VM instance.
* **Management, Security, and Additional Options:** Configure additional settings such as enabling Stackdriver monitoring, setting up SSH access, and adding any required metadata.

**Step 4: Confirm and Create**

Review the configuration settings and click on the "Create" button to create the VM instance. Compute Engine will provision the instance and make it available for use.

**Instance Types**

Compute Engine offers a wide range of instance types to cater to diverse workload requirements. Each instance type varies in terms of computing power, memory capacity, and storage capabilities. It's crucial to choose the right instance type to ensure optimal performance and cost efficiency. Here are some commonly used instance types:

* General Purpose Instances
* Memory-Optimized Instances
* Compute-Optimized Instances
* Accelerator-Optimized Instances

**Instances Customization**

**Compute Engine also provides the option to customize VM instances based on specific requirements. Users can define custom machine types by specifying the desired number of virtual CPUs (vCPUs) and memory capacity. This allows for fine-grained control over resource allocation, ensuring optimal performance for unique workloads.**

**Kubernetes Engine: Introduction to Kubernetes, deploying applications**

Kubernetes has emerged as a powerful container orchestration platform, revolutionizing the way applications are deployed, managed, and scaled. Google Cloud Platform's Kubernetes Engine, commonly known as GKE, offers a managed environment for leveraging the capabilities of Kubernetes seamlessly.

**Understanding Kubernetes:**

Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It provides a robust framework for containerized workloads, enabling efficient resource utilization, high availability, and effortless scaling. Key concepts of Kubernetes include:

* **Containers:** Kubernetes leverages containers, such as Docker, to package and isolate application components, ensuring consistency and portability across different environments.
* **Pods:**The basic unit of deployment in Kubernetes is a pod, which encapsulates one or more containers. Pods share network and storage resources, facilitating communication between containers within the same pod.
* **Deployments:** Deployments define the desired state of applications in Kubernetes. They ensure that a specified number of replicas of a pod are running and handle rolling updates and rollbacks seamlessly.
* **Services:**Kubernetes services enable communication between different pods and provide a stable network endpoint for accessing applications. Services abstract the underlying pod IPs and ensure load balancing.
* **Scaling:** Kubernetes allows for both horizontal and vertical scaling of applications. Horizontal scaling involves adding or removing pod replicas to meet fluctuating demand, while vertical scaling adjusts the resources allocated to individual pods.

**Deploying Applications with GKE:**

Google Cloud Platform's Kubernetes Engine simplifies the process of deploying and managing Kubernetes clusters. Let's explore the steps involved in deploying applications using GKE:

**Step 1: Creating a GKE Cluster:**  
Access the Google Cloud Console and navigate to the Kubernetes Engine section. Click on "Create Cluster" to specify cluster details such as the number and type of nodes, region, and node pool configuration. GKE automatically provisions the cluster and sets up the necessary infrastructure.

**Step 2: Preparing Application Deployment:**  
Prepare your application for deployment on Kubernetes by containerizing it using a containerization tool like Docker. Create a Docker image containing your application and push it to a container registry, such as Google Container Registry.

**Step 3: Deploying the Application:**  
In the GKE interface, click on "Workloads" and then "Deployments." Click on "Create Deployment" to configure the deployment settings. Specify the Docker image, container ports, resource requirements, and any necessary environment variables.

**Step 4: Exposing the Application:**  
To make the application accessible externally, create a Kubernetes service. In the GKE interface, navigate to "Workloads" and then "Services." Click on "Create Service" and choose the appropriate service type. Specify the port mappings and target port to route traffic to the application.

**Step 5: Scaling and Monitoring:**  
GKE offers automatic scaling capabilities based on CPU utilization or custom metrics. You can configure autoscaling policies for your application to ensure efficient resource allocation. GKE also integrates with monitoring and logging solutions like Stackdriver, enabling you to gain insights into the performance and health of your deployed applications.

**App Engine: Deployment, versioning, traffic splitting**

Google Cloud Platform's App Engine offers a fully managed platform-as-a-service (PaaS) environment, enabling developers to focus on building and deploying applications without worrying about infrastructure management. App Engine simplifies the deployment process, provides versioning capabilities, and allows for efficient traffic splitting.

**Deployment Process in App Engine:**

App Engine simplifies the deployment of applications with a few straightforward steps. Let's walk through the process:

**Step 1: Application Configuration:**

Configure your application by defining the necessary files and dependencies. App Engine supports multiple programming languages, including Java, Python, Node.js, Go, and more. Create the required configuration files, such as the app.yaml file, to specify settings like runtime environment, instance scaling, and resource allocation.

**Step 2: Deploying the Application:**

Using the **gcloud** command-line tool or the Google Cloud Console, initiate the deployment process by specifying the application directory or the deployment package. App Engine automatically provisions the necessary infrastructure and deploys the application based on the defined configuration.

**Step 3: Continuous Deployment:**

App Engine supports continuous deployment, allowing for seamless updates to your application. As you make changes to your application's codebase, App Engine automatically detects the updates and deploys the latest version without interruption. This ensures a smooth development and deployment workflow.

**Versioning in App Engine:**

App Engine offers powerful versioning capabilities, enabling developers to manage and control different iterations of their applications. Key aspects of versioning in App Engine include:

* **Multiple Versions:** You can deploy multiple versions of your application simultaneously. Each version can have its own configuration, environment variables, and resource settings.
* **Traffic Allocation:** App Engine allows you to allocate traffic percentages to different versions. This feature enables gradual rollouts and A/B testing by directing a portion of the traffic to specific versions while keeping the remaining traffic on the current stable version.
* **Traffic Splitting:** App Engine's traffic splitting feature provides fine-grained control over routing user requests to different versions or services. You can define rules based on IP address, cookie values, or HTTP headers to distribute traffic intelligently.
* **Rollback and Cleanup:** If any issues arise with a new version, App Engine enables you to roll back to a previous version quickly. It also offers cleanup options to remove unused versions, ensuring efficient resource management.

**Efficient Traffic Spitting:**

App Engine's traffic splitting functionality plays a vital role in managing user traffic distribution across multiple versions or services. Key features of traffic splitting in App Engine include:

**Splitting Rules:** You can define rules to split traffic based on various parameters, such as IP address ranges, cookies, or HTTP headers. This allows for targeted traffic routing and testing of different versions or services.

**Gradual Rollouts and Rollbacks:**The app Engine allows you to gradually route traffic to a new version to ensure stability and catch any issues early. If necessary, you can quickly roll back to a previous version with minimal disruption.

**Monitoring and Insights:**App Engine provides built-in monitoring and logging capabilities, allowing you to track the performance and behavior of different versions or services. This helps you make informed decisions regarding traffic splitting and version management.

**Cloud Storage: Buckets, object versioning, data transfer**

Cloud Storage, a highly scalable and durable object storage service provided by Google Cloud Platform (GCP), offers a flexible and reliable solution for storing and accessing data in the cloud.

**Buckets in Cloud Storage:**

Buckets are the fundamental containers used to store data in Cloud Storage. A bucket serves as a logical unit for organizing and managing objects, which can be files, images, videos, or any other type of data. Key aspects of buckets in Cloud Storage include:

* **Naming and Access Control:** Buckets are identified by globally unique names. Access control lists (ACLs) and Cloud Identity and Access Management (IAM) policies enable granular control over who can access and modify buckets and the objects within them.
* **Fine-Grained Permissions:** Cloud Storage allows for defining fine-grained access permissions at the object level, enabling secure and controlled data access for different users or applications.
* **Data Redundancy and Durability:** Cloud Storage automatically replicates data within a region or across multiple regions to ensure high availability, durability, and data redundancy. This helps protect against data loss and ensures data integrity.

**Object Versioning in Cloud Storage:**

Cloud Storage provides object versioning capabilities, allowing users to keep multiple versions of an object over time. This feature offers increased data protection, simplified backup and restore processes, and easy recovery from accidental deletions or modifications. Key aspects of object versioning in Cloud Storage include:

* **Versioning Configuration:** Users can enable versioning for a specific bucket, specifying whether to create new versions when objects are overwritten or deleted. Once enabled, Cloud Storage automatically generates a unique version ID for each version of an object.
* **Managing Object Versions:** Cloud Storage provides APIs and tools to manage object versions. Users can list, retrieve, and restore previous versions of objects, providing enhanced data control and protection.
* **Lifecycle Management:**Object lifecycle policies in Cloud Storage allow users to automatically transition or delete object versions based on predefined rules. This helps optimize storage costs and ensures compliance with data retention policies.

**Data Transfer Options in Cloud Storage:**

Cloud Storage offers various options for efficient data transfer into and out of the storage service. These options are designed to streamline large-scale data transfers and optimize network performance. Key data transfer options in Cloud Storage include:

* **Transfer Service:** Cloud Storage Transfer Service enables seamless data transfers from on-premises systems or other cloud storage providers to Cloud Storage. It provides features like scheduling, data validation, and parallel transfers, ensuring fast and secure data migration.
* **Transfer Appliance:** For large-scale offline data transfers, Cloud Storage Transfer Appliance offers a hardware-based solution. Users can transfer terabytes or petabytes of data by shipping the physical appliance, ensuring rapid and reliable data ingestion.
* **Transfer Acceleration:**Cloud Storage Transfer Acceleration utilizes Google's global network infrastructure to optimize data transfer speed. It leverages edge caching and routing techniques to accelerate data uploads and downloads, especially for long-distance transfers.

**Cloud SQL and Bigtable**

Google Cloud Platform (GCP) offers a range of managed database solutions to cater to different application requirements. Two prominent options are Cloud SQL and Bigtable, both designed to provide scalable, highly available, and managed database services.

**Cloud SQL: Relational Database as a Service**

Cloud SQL is GCP's managed relational database service that supports popular database engines such as MySQL, PostgreSQL, and SQL Server. Key features of Cloud SQL include:

* **Ease of Use:** Cloud SQL simplifies database management by handling infrastructure provisioning, patching, backups, and replication. It provides a user-friendly interface for database administration tasks.
* **Horizontal Scalability:** Cloud SQL supports automatic scaling vertically (upgrading to more powerful machines) and manual horizontal scaling (adding read replicas). This allows you to accommodate increasing workloads while maintaining performance.
* **Data Protection:** Cloud SQL offers automated backups, point-in-time recovery, and the option to enable high availability with regional replication. These features ensure data durability and provide disaster recovery capabilities.
* **Integration with GCP Services:** Cloud SQL seamlessly integrates with other GCP services, such as Compute Engine, App Engine, and Dataflow. This enables smooth data access and application development workflows.

**Use cases:** Cloud SQL is well-suited for a wide range of applications, including web and mobile applications, e-commerce platforms, content management systems, and traditional relational database workloads.

**Bigtable: NoSQL Wide-Column Database**

Bigtable is GCP's fully managed NoSQL wide-column database service, designed for handling massive amounts of structured data with low latency and high scalability. Key features of Bigtable include:

* **Scalability:** Bigtable is built to handle petabytes of data with high read and write throughput. It can scale horizontally by adding nodes to the cluster, allowing you to handle increasing data volumes and user traffic.
* **Low Latency:** Bigtable provides single-digit millisecond latency for read and write operations, making it ideal for real-time applications and use cases that require rapid data access.
* **Replication and Durability:** Bigtable automatically replicates data across multiple zones within a region for high availability and durability. This ensures data resilience and minimizes the risk of data loss.
* **Integration with Big Data Tools:** Bigtable seamlessly integrates with other GCP services like BigQuery, Dataflow, and Dataproc. This enables data processing and analysis at scale, making it suitable for analytics and data-intensive workloads.

**Use cases:** Bigtable is well-suited for applications that require high scalability, low-latency data access, and real-time analytics. It is commonly used in areas such as IoT data processing, time series data analysis, recommendation systems, and AdTech platforms.

Session 2: Networking, IAM in GCP

## Virtual Private Cloud (VPC)

Google Cloud Platform (GCP) offers a robust and feature-rich networking solution called Virtual Private Cloud (VPC) to help organizations deploy, manage, and scale cloud-based applications and services.

### What is Virtual Private Cloud (VPC)?

Virtual Private Cloud (VPC) is a virtual network within GCP that enables you to securely manage your cloud resources. It offers a private and isolated space for your services, which is essential for configuring the network in a way that meets security and compliance requirements. With VPC, you can define your IP address range, create subnets, configure routing tables, and set up network gateways.

### Key features of VPC in Google Cloud Platform

* **Global Scope**  
  Unlike some cloud providers that limit VPC to a single region, Google Cloud VPC has a global scope. This means that you can have resources in different regions all within the same VPC. This global nature eliminates the need to set up VPNs or configure network peering between regions, thus simplifying network management.
* **Scalability**  
  Google Cloud VPC is designed to scale effortlessly. As your business needs evolve, you can add or remove instances and services to your VPC without the need to change the underlying network infrastructure. This flexibility allows you to scale your operations efficiently.
* **Security**  
  Security is a top priority for any cloud deployment. With Google Cloud VPC, you can set up firewalls, control network access, and use Identity and Access Management (IAM) to define who has access to your network resources. Moreover, VPC Service Controls allow you to establish a secure perimeter around your VPC to mitigate data exfiltration risks.
* **Fine-grained Controls**  
  Google Cloud VPC provides fine-grained controls over networking aspects such as IP addressing, routing, and network peering. You can create subnets, allocate static IP addresses, and configure custom routes for sophisticated network topologies.
* **Integration with Google Cloud Services**  
  VPC is natively integrated with Google Cloud’s services such as Compute Engine, Kubernetes Engine, and Cloud Functions. This integration allows for seamless communication between services and facilitates a streamlined development process.
* **Shared VPC**  
  Shared VPC enables resources from multiple projects to be securely connected to a common VPC network. This means that you can centralize network administration and management across several projects, thereby creating a more efficient and controlled environment.

### Use Cases of VPC in Google Cloud Platform

* **Hybrid Cloud Deployments**  
  Organizations that have a mix of on-premises and cloud resources can use Cloud VPN and Cloud Interconnect to securely connect their on-premises data centers to Google Cloud VPC.
* **Secure Application Deployment**  
  Deploying applications in a VPC ensures a secured network perimeter. Businesses can isolate their sensitive applications and data in private networks and use VPC firewalls to control inbound and outbound traffic.
* **Global Deployments**  
  For businesses operating globally, VPC’s global scope allows for the easy deployment of resources across multiple regions while maintaining a single network. This simplifies management and enhances performance by reducing latency.

## Load balancing and Cloud CDN

In an era where data drives businesses and application performance is critical, it’s essential to implement solutions that ensure high availability and low latency for users. Google Cloud Platform (GCP) offers two robust services for this purpose: Load Balancing and Cloud Content Delivery Network (CDN).

### Load Balancing in Google Cloud Platform

Load balancing is the process of distributing network traffic across multiple servers to ensure that no single server is overwhelmed with too much traffic. This ensures high availability, reliability, and performance of web applications.

### Key Features of Load Balancing in GCP

* **Global and Regional Load Balancing:**GCP offers both global and regional load balancing. Global load balancing distributes traffic across multiple regions, while regional load balancing does so within a single region.
* **Intelligent Traffic Distribution:** GCP load balancers distribute traffic based on several factors, including the proximity of the user, the health of the VM instances, and the global backend capacity, ensuring optimal performance.
* **Scalability:**Load balancing can handle traffic ranging from a few requests per second to millions of requests per second without any manual intervention.
* **Integrated DDoS Protection:** Google Cloud Load Balancers are equipped with DDoS protection, mitigating the risk of DDoS attacks.

### Types of Load Balancing in GCP

* **HTTP(S) Load Balancing:** Primarily for HTTP/HTTPS traffic, this global load balancer distributes traffic based on the HTTP(S) request data.
* **TCP/SSL Proxy Load Balancing:**This is a global, non-proxied load balancer suitable for TCP traffic.
* **Network Load Balancing:** This regional load balancer is best for distributing TCP/UDP traffic to backends within the same region.
* **Internal TCP/UDP Load Balancing:**This is used for balancing internal traffic within a GCP VPC.
* **Internal HTTP(S) Load Balancing:**A proxy-based, regional load balancer for distributing HTTP(S) traffic among backends in the same region.

### Cloud CDN in Google Cloud Platform

Cloud CDN (Content Delivery Network) uses Google's globally distributed edge points to cache external HTTP(S) load-balanced content closer to the users. This reduces latency by delivering content more quickly.

### Key Features of Cloud CDN in GCP

* **Global Distribution:** Google Cloud CDN leverages Google's edge network, which spans across numerous locations globally, ensuring content is served with low latency no matter where the user is.
* **Cache-Control:** It provides fine-grained cache controls, allowing you to specify what content is cached and for how long.
* **Integration with Load Balancing:** Cloud CDN is integrated with Google Cloud Load Balancing, allowing you to enable CDN capabilities for your load balancer with just a few clicks.
* **Logging and Monitoring:**With Stackdriver integration, Cloud CDN provides detailed logs and monitoring insights into cache performance and utilization.

### Use Cases for Cloud CDN

* **Media Content Delivery:** For websites with rich media content like videos, images, and scripts, Cloud CDN ensures that this content is delivered quickly.
* **Software Distribution:**Cloud CDN is ideal for distributing software patches and updates to a global user base.
* **API Acceleration:**Caching API responses at the edge can significantly improve the performance of API calls.

### Combining Load Balancing and Cloud CDN for Optimal Performance

When used together, Load Balancing and Cloud CDN provide a powerful combination for web application deployment. Load Balancing ensures that your application is highly available and scalable, while Cloud CDN ensures that your content is delivered with low latency. This combination is especially beneficial for applications with a global user base, as it ensures that all users, regardless of location, have a fast and reliable experience.

## Cloud DNS in Google Cloud Platform

In the vast world of the internet, the Domain Name System (DNS) serves as an essential foundation that makes it easy for users to interact with websites and services. Instead of remembering complex IP addresses, users can type in an easily readable domain name which the DNS resolves to the appropriate IP. For enterprises and developers deploying applications on the cloud, managing DNS effectively is vital.

### Introduction to Cloud DNS

Google Cloud DNS is a scalable, reliable, and managed DNS service running on the same infrastructure as Google. It provides domain name resolution services, allowing you to publish your domain names by translating them into IP addresses.

### Key Features of Cloud DNS

* **High Performance**: Cloud DNS caches queries at the edge of Google's network, providing low-latency DNS resolution for users globally.
* **Scalability**: It can handle internet-scale traffic, from just a few queries per second to over a million queries per second, without any provisioning.
* **Reliability**: Cloud DNS is designed for high availability and guaranteed uptime, leveraging Google's infrastructure.
* **Security**: DNSSEC integration ensures the authenticity and integrity of DNS responses, protecting against DNS cache poisoning attacks.
* **Simple Management**: With an intuitive interface and powerful APIs, managing DNS records is streamlined.

### Getting started with Cloud DNS

**Step 1: Set Up Cloud DNS in Google Cloud Platform**

* Navigate to the Google Cloud Console and select the project you want to work on.
* In the left-hand menu, navigate to “Network Services” and then select “Cloud DNS.”
* Click on the “Create Zone” button to set up a new DNS zone.

**Step 2: Configure DNS Zone**

* Select the zone type, Public for external or Private for internal Google Cloud resources.
* Enter a Zone Name and DNS name, which is usually the domain you want to manage.
* Optional: Configure DNSSEC for added security.

**Step 3: Adding DNS Records**

* Inside your DNS Zone, click on “Add Record Set.”
* Enter the DNS name and select the record type (A, CNAME, MX, etc.).
* Provide the appropriate data for the record type and click “Create.”

**Step 4: Updating Domain Registrar**

* Note the Cloud DNS name servers listed in your zone details.
* Go to your domain registrar and update the name servers with the ones provided by Cloud DNS.

### Advantages of using Cloud DNS

* **Enhanced Performance:** By leveraging Google's global network of caching servers, Cloud DNS ensures low-latency DNS resolution for users around the world.
* **Easy Integration:** Cloud DNS integrates seamlessly with other Google Cloud services, making it easy to manage and deploy applications on Google Cloud.
* **DNSSEC Protection:** The support for Domain Name System Security Extensions (DNSSEC) means that you can secure your domain against cache poisoning and other types of DNS forgeries.
* **Audit and Monitoring:**Integration with Cloud Audit Logs and Cloud Monitoring allows you to keep track of changes and monitor the performance of your DNS zones.

### Use Cases of Cloud DNS

* **Web Applications:** For web applications hosted on Google Cloud, Cloud DNS provides an efficient way to manage domain names.
* **Microservices Architecture:** Within complex microservices environments, Cloud DNS can be used to manage internal service discovery through private DNS zones.
* **Multi-regional Deployment:**For applications deployed across multiple regions, Cloud DNS ensures consistent, low-latency access to the right regional endpoints.

## Identity and Access Management (IAM) roles and policies

As enterprises embrace the cloud to build and deploy applications, securing and managing access to cloud resources is paramount. Google Cloud Platform (GCP) offers a comprehensive set of tools and services for managing identity and access.

### Introduction to IAM in GCP

Identity and Access Management (IAM) in GCP enables administrators to manage access to GCP resources securely. With IAM, you can create and manage identities, define permissions, and control access to GCP resources at a granular level.

### Core Components of IAM

* **Members:**These are the identities that have access to GCP resources. Members can be individual users, groups, service accounts, or even entire G Suite domains.
* **Roles:**A role is a collection of permissions that can be assigned to members. Permissions determine what actions are allowed on resources.
* **Policies:** These are bindings that link members to roles. A policy, when attached to a resource, defines who has what kind of access.

### Understanding Roles

In GCP, roles are central to how permissions are granted. There are three types of roles:

* **Primitive Roles:** These include Owner, Editor, and Viewer. These roles existed before IAM and provide broad permissions across all GCP resources.
* **Predefined Roles:** These are roles that GCP has created to correspond to common use cases (e.g., Compute Engine Instance Admin, Storage Object Viewer). They offer a finer granularity than Primitive roles.
* **Custom Roles:** These allow you to create a set of permissions tailored to your specific needs. Custom roles can be created at the project or organization level.

### Managing Policies

A policy in GCP IAM is a set of statements that define who has what type of access. A policy consists of one or more bindings, where a binding is an association between a role and the members who are granted that role.

### Best Practices for Managing IAM in GCP

#### **Principle of Least Privilege**

Assign users the fewest number of permissions they need to perform their job. Instead of using Primitive Roles which are broad, opt for Predefined or Custom Roles which can be tailored to specific needs.

#### **Use Groups and Service Accounts**

Assign roles to groups or service accounts instead of individual users. This makes it easier to manage access for a large number of users.

#### **Regularly Audit IAM Policies**

Regularly review and audit IAM policies and roles to ensure that they align with the current requirements. Remove any redundant permissions or access rights.

#### **Use Resources Hierarchies**

GCP resources are organized hierarchically ***(Organization -> Folder -> Project -> Resource)***. Policies can be applied at any level in this hierarchy. Understand and leverage this hierarchy to efficiently manage access across your resources.

#### **Enable IAM Role Recommendations**

GCP provides recommendations for IAM roles based on the actual usage. By enabling this, you can receive suggestions on granting or revoking access.