**DAY 2 ANALYTICAL QUESTIONS**

1. **Sharing a single physical instance of a resource or an application among multiple customers and organizations at one time. It can be achieved by assigning a logical name to physical resources and providing a pointer to that physical resource on demand. Demonstrate the different types of services offered by cloud computing with its architecture.**

A. Virtualization is a technique which allows the single instance of a resource in application. It is done by assigning a logical name to a physical resource and providing a pointer to that physical resource on demand.

\*it is a process of creating a virtual version of something like computer hardware.

\*with the virtualization multiple os and applications can run on the same machine and its same hardware at the same time , increasing the utilization and flexibility of the hardware.

\*the term virtualization is often synonymous with hardware virtualization, which plays a fundamental role in delivering Iaas solutions for cloud computing.

+types of virtualization:

1.os virtualization

2.server virtualization

3.Hardware virtualization

4.Storage virtualization

\*apart from cloud computing in virtualization offers types of services:

1.information as a service: This layer provides the most basic blocks like virtual machines, storage, and networking. Users have control over the operating system and applications they install in vms.

2.platform as a service: this layer builds on top of iaas, providing pre-configured environments with os and development tools.

3.software as a service: This layer delivers complete applications over the internet. Users access these applications through a web browser or API without needing to manage any infrastructure or platforms.

Benefits of Cloud Architecture:

* Resource Sharing: Virtualization allows efficient sharing of physical resources, leading to cost savings and reduced environmental impact.
* Scalability: Users can easily scale their resources up or down on demand, paying only for what they use.
* Flexibility: Cloud services offer a wide range of options to meet diverse user needs.
* Accessibility: Cloud applications are accessible from anywhere with an internet connection.

In essence, cloud computing architecture uses logical names, pointers, and virtualization to create a multi-tenant environment where resources are shared efficiently and on-demand, delivering different services based on user needs.

**2. My organization prefers to deal with a cloud vendor that has implemented certain standards for quite a while. It will provide us with greater confidence in doing business with them. Is there any ISO standard out there related to Cloud?**

A. I think ISO standard cloud vector is a wise decision. It demonstrates a focus on security , reliability and consistent practices when dealing with cloud services.

\*there are some of the ISO standards specifically related to cloud computing:

1>security focus:

* ISO 27001:this is a foundational informational security standard application to any organization including cloud providers. It establishes best practices for managing information security risks.
* ISO/IEC 27017:building on ISO 27001,this standard offers specific guidance for information security controls in cloud services.it applies both providers and users.

2>Data protection:

* ISO/IEC 27018:This standard specifically focuses on protecting personally identifiable services, providing a clear understanding of the landscope.

3>Cloud famework:

* ISO/IEC 17789: this standard defines the terminology and architecture for cloud computing services , providing a clear understanding of the landscape.

ADVANTAGES:

* Enhanced security
* Improved reliability
* Increased transparency.

Choosing a cloud vendor with a strong ISO compliance track record fosters confidence in doing business with them and ensures a secure, reliable, and high-quality cloud service experience.

**3. The software defined framework is used to interact and be useful in cloud platforms as it allows easy implementation of it on the system. It removes the need to write full-fledged programs. It provides the instructions to make the communication between one or more applications. Demonstrate to implement this scenario in cloud computing with a suitable architecture diagram**

**A.** Software-Defined Frameworks (SDFs) are powerful tools in cloud computing. They offer a pre-built structure and tools for developers to interact with cloud services without writing complex code from scratch. Here's a breakdown of how this works and an illustrative architecture diagram for a cloud scenario:

**Benefits of SDFs in Cloud Computing:**

* **Rapid Development:** SDFs provide pre-defined functionalities, allowing developers to focus on application logic rather than low-level infrastructure details. This accelerates development and deployment times.
* **Abstraction:** They act as an abstraction layer, hiding the complexities of underlying cloud APIs and services. Developers can interact with the cloud platform through a familiar and consistent interface.
* **Reusability:** Many SDFs offer reusable components, promoting code reuse and reducing development effort.

**Cloud Architecture with SDF:**

Imagine you're building a cloud application that processes large datasets. Here's how an SDF can be implemented:

1. **Cloud Platform:** The foundation is your chosen cloud platform, like AWS, Azure, or GCP. This platform provides the underlying infrastructure and services.
2. **Software Defined Framework (SDF):** On top of the cloud platform sits the chosen SDF. This could be a framework like Apache Spark for data processing or Spring Cloud for microservices architecture.
3. **Development Environment:** Developers use their preferred IDE (Integrated Development Environment) to write application code. However, instead of low-level cloud service interactions, they utilize the functionalities provided by the SDF.
4. **Cloud Services:** The SDF interacts with various cloud services on behalf of the application. This could involve storage services (like S3 in AWS) for data processing or managed databases (like RDS in AWS) for storing processed results.

**Architecture Diagram:**

+-------------------+

| Client | (Web App, Mobile App)

+-------------------+

|

v

+-------------------+

| Development Env | (IDE with SDF libraries)

+-------------------+

|

v

+-------------------+

| Software Defined |

| Framework | (e.g., Apache Spark)

+-------------------+

|

v

+-------------------+

| Cloud Platform | (AWS, Azure, GCP)

+-------------------+

|

v

+-------------------+

| Cloud Services | (Storage, Databases etc.)

+-------------------+

**Explanation:**

* The client interacts with your cloud application (web app, mobile app, etc.).
* Developers use their IDE with the chosen SDF's libraries to write the application logic.
* The SDF interacts with various cloud services on the developer's behalf, leveraging the cloud platform's functionalities.

By utilizing an SDF in this architecture, developers can focus on building innovative applications without getting bogged down in the intricacies of cloud service APIs. The framework simplifies development, promotes code reuse, and streamlines cloud resource management.

**4. If my company has given the business case a go and the result says we should adopt the cloud. That said technical wise from top to bottom, what are key successful elements to consider the pros and cons of?**

**ANSWER:**

* **Planning and Strategy:** Define your cloud migration strategy, considering which applications and data are best suited for the cloud. Assess your current IT infrastructure and its compatibility with cloud services.
* **Security:** Ensure robust security measures are in place to protect your data and applications in the cloud environment. Evaluate the security practices of your chosen cloud provider and implement additional security controls as needed.
* **Cost Management:** Cloud services often offer a pay-as-you-go model. However, it's crucial to monitor and optimize cloud resource usage to avoid unexpected costs.
* **Network Connectivity:** A reliable and high-speed internet connection is essential for seamless cloud access and performance. Evaluate your current network bandwidth and consider any potential upgrades necessary.
* **Scalability:** The cloud offers on-demand scalability, allowing you to easily adjust resources based on your needs. Ensure your chosen cloud platform offers the required scaling capabilities to accommodate future growth.
* **Integration:** Consider how your existing on-premises IT infrastructure will integrate with the cloud environment. Plan for seamless data transfer and application integration between cloud and on-premises systems.
* **Skills and Expertise:** Evaluate your current IT team's skillset for cloud management. Consider training or hiring personnel with cloud expertise to ensure smooth operation and maintenance of your cloud environment.

By carefully considering these technical elements, your company can ensure a successful and secure transition to the cloud, maximizing the benefits of cloud computing while minimizing potential drawbacks.

## In computing, virtualization describes the process of abstracting a physical object into a logical object. The logical object is the digital replica of the real system. This replication is created in software to make a copy of the hardware. With virtualization technology, the resources can be used more effectively and efficiently to introduce new capabilities and to reduce operational costs. (i). Demonstrate your knowledge of the advantages gained when implementing this virtualization in the cloud.

## A. Advantages of Virtualization in the Cloud:

Virtualization offers numerous advantages when implemented in the cloud environment, making it a cornerstone technology for modern computing. Here are some key benefits:

* **Cost savings:** By consolidating multiple virtual machines (VMs) onto a single physical server, organizations can significantly reduce hardware costs. This optimization eliminates the need for underutilized physical servers, leading to lower power consumption and data center space requirements.
* **Increased Efficiency and Resource Utilization:** Virtualization allows for better utilization of computing resources. A single physical server can run multiple VMs, each with its own operating system and applications. This eliminates the need for dedicated servers for individual tasks, maximizing resource utilization and reducing hardware waste.
* **Scalability and Agility:** Cloud-based virtualization enables organizations to easily scale their resources up or down on-demand. This allows them to quickly adapt their computing capacity to meet fluctuating workloads without the need to purchase additional hardware. VMs can be easily provisioned and deployed, making it faster and easier to respond to business needs.
* **Improved Disaster Recovery:** Virtualization facilitates faster and more efficient disaster recovery. VMs can be easily replicated and backed up in the cloud, allowing for quick restoration in case of server failure or other disruptions. This minimizes downtime and ensures business continuity.
* **Enhanced Security:** Virtualization can improve security by isolating VMs from each other. This creates a secure environment where a security breach in one VM won't affect other VMs on the same physical server. Additionally, cloud providers often offer robust security features like firewalls and access controls to further protect your data and applications.
* **Simplified Management:** Virtualization tools automate many administrative tasks, simplifying IT operations and management. This allows IT teams to provision, configure, and monitor VMs from a centralized platform, saving time and effort.
* **Improved Testing and Development:** Virtualization provides a flexible environment for testing and development activities. Developers can easily create and deploy different VMs with specific configurations, allowing them to test applications and software efficiently without impacting production systems.

These are just some of the many advantages of using virtualization in the cloud. By leveraging this technology, organizations can achieve significant cost savings, improve efficiency, and gain greater agility and flexibility in their IT infrastructure.

## 7. Consider two cloud service systems: Google File System and Amazon S3. Explain how they achieve their design goals to secure data integrity and to maintain data consistency while facing the problems of hardware failure, especially concurrent hardware failures. A. Data Integrity and Consistency in Google File System (GFS) and Amazon S3

Both Google File System (GFS) and Amazon S3 prioritize data integrity and consistency, albeit with different approaches due to their design goals. Here's a breakdown of how they achieve these goals in the face of hardware failures, including concurrent failures:

**Google File System (GFS):**

* **Design Goal:** Highly available, write-once-append-many (WORM) file system for large datasets.
* **Data Integrity:**
  + **Checksums:** Data chunks are accompanied by checksums, allowing for verification during reads and writes. Any discrepancies trigger repairs.
  + **Replication:** Data is replicated across multiple machines (typically 3-6 replicas). Inconsistencies are resolved by comparing replicas during reads and writes.
* **Data Consistency:**
  + **Leases:** Clients acquire write leases that grant exclusive write access for a limited time. This prevents concurrent modifications from corrupting data.
  + **Versioning:** GFS maintains multiple versions of a file, allowing recovery from inconsistencies caused by failures.
* **Handling Hardware Failures:**
  + **Replication:** Replication ensures data availability even if one or more replicas are lost due to hardware failure.
  + **Master Election:** In case of a master server failure, a new master is elected from the remaining servers, minimizing downtime.
  + **Chunk Servers:** GFS can tolerate the failure of individual chunk servers by reconstructing lost data from remaining replicas.

**Amazon S3 (Simple Storage Service):**

* **Design Goal:** Highly scalable, low-cost object storage for a variety of data types.
* **Data Integrity:**
  + **Checksums:** Similar to GFS, S3 utilizes checksums to detect data corruption during upload and download operations.
  + **Parity Protection:** S3 offers optional parity protection (configurable with S3 Storage Classes) that allows reconstruction of lost data from redundant information stored across multiple servers.
* **Data Consistency:**
  + **Eventual Consistency:** S3 prioritizes availability over strict consistency. Writes are replicated asynchronously across geographically dispersed servers. This means there may be a slight delay before all replicas reflect the latest update.
  + **Read-After-Write Consistency:** S3 offers options like Read-After-Write Consistency for critical data, ensuring a read operation reflects the latest write before returning the data.
* **Handling Hardware Failures:**
  + **Redundancy:** Data objects are stored across multiple geographically dispersed servers, ensuring availability even if a single server fails.
  + **Self-Healing:** S3 automatically detects and repairs inconsistencies caused by hardware failures using redundant copies.

**Key Differences:**

* **Consistency Model:** GFS prioritizes strong consistency using leases and versioning, while S3 adopts an eventual consistency model for scalability and availability.
* **Replication:** GFS typically uses a smaller number of highly reliable replicas, while S3 leverages a larger number of geographically dispersed servers for redundancy.
* **Focus:** GFS is optimized for large datasets and write-once workloads, while S3 caters to a broader range of data types and access patterns.

**Concurrent Hardware Failures:**

Both GFS and S3 are designed to handle concurrent failures to a certain extent. GFS's replication and master election process can tolerate the loss of multiple chunk servers and the master server. S3's redundancy across geographically dispersed servers ensures data availability even with concurrent failures in a specific location.

However, the ability to withstand concurrent failures depends on the number of replicas and the chosen storage class in S3. Additionally, extensive hardware failures might still lead to data loss in either system.

**Conclusion:**

GFS and S3 demonstrate different approaches to data integrity and consistency based on their design goals. GFS prioritizes strong consistency for critical data, while S3 emphasizes availability and scalability for various use cases. The choice between them depends on the specific needs of the application regarding consistency requirements, data access patterns, and cost considerations.

## 8. Draw a layered diagram to relate the construction of IaaS, PaaS, and SaaS clouds from bare machine hardware to the user’s applications. Briefly list the representative cloud service offerings at each cloud layer from the major cloud providers that you know of. A. Layered Cloud Construction Diagram

**Top Layer:** User Applications

**Middle Layer (PaaS):** Platform as a Service

* **Representative Offerings:**
  + **AWS:** AWS Elastic Beanstalk, AWS Lambda
  + **Microsoft Azure:** Azure App Service, Azure Functions
  + **Google Cloud Platform (GCP):** App Engine, Cloud Functions

**Bottom Layer (IaaS):** Infrastructure as a Service

* **Representative Offerings:**
  + **AWS:** Amazon EC2 (virtual machines), Amazon S3 (storage)
  + **Microsoft Azure:** Azure Virtual Machines, Azure Blob Storage
  + **GCP:** Compute Engine (virtual machines), Cloud Storage

**Base Layer:** Bare Machine Hardware

**Connections:**

* User applications interact directly with the PaaS layer.
* The PaaS layer utilizes resources from the IaaS layer (e.g., virtual machines, storage) to run the applications.
* The IaaS layer manages the physical hardware infrastructure.

**9. This assignment requires you to combine the queuing and publish -subscribe paradigms within a single application. Organizations A and B are two businesses that use queuing for B2B transactions. Every transaction is stored (prior to forwarding) and has a 128 -bit UUID identifier associated with it. Within the organization, messages are delivered using publish -subscribe. Create five subscribers (sales, marketing, audit, packaging, and finance) within each organization; these subscribers do not have to log the messages again since there is a copy of that message already available. If the assignment is done in Java it is prescribed that the Java Message Service be used.**

**A.** Here's a design for a Java application utilizing JMS (Java Message Service) to combine queuing and publish-subscribe for B2B transactions between Organizations A and B:

**Components:**

1. **Message Queue:** A single queue will be used for B2B transactions. This queue acts as a buffer for messages before forwarding them to the receiving organization.
2. **JMS Message Producer (Organization A):** This program creates a transaction message with a UUID and relevant data. It then sends the message to the message queue.
3. **JMS Message Consumer (Organization B):** This program receives messages from the queue and forwards them to the appropriate topic within Organization B based on the transaction type.
4. **JMS Topic (Organization A & B):** Each organization will have an internal topic for distributing received B2B messages to its subscribers (sales, marketing, audit, packaging, finance).
5. **JMS Message Subscribers (Organization A & B):** These programs subscribe to their respective organization's topic and receive the forwarded B2B messages based on their interest.

**Implementation with JMS:**

* **Message Format:** The transaction message can be a Java object containing fields like UUID, transaction type, data payload, etc.
* **JMS Libraries:** Use libraries like Apache ActiveMQ or the built-in Java JMS API to interact with the messaging system.
* **Message Producer (Organization A):**
  + Creates a JMS connection factory and session.
  + Creates a temporary queue for the producer.
  + Creates a JMS message object containing the transaction details and UUID.
  + Sends the message to the message queue.
* **Message Consumer (Organization B):**
  + Creates a JMS connection factory and session.
  + Creates a consumer for the shared B2B transaction queue.
  + On receiving a message, extracts the transaction type and UUID.
  + Publishes the message to the appropriate topic within Organization B using a JMS message publisher.
* **Message Subscribers (Organization A & B):**
  + Each subscriber creates a connection factory and session.
  + Subscribes to the relevant topic within their organization.
  + On receiving a message, processes the data based on their function (e.g., sales might update CRM, finance might trigger payment processing).

**Benefits:**

* **Decoupled Communication:** Organizations A and B are decoupled, as the message producer doesn't need to know about the specific consumers.
* **Scalability:** The system can easily handle varying message volumes.
* **Reliable Delivery:** JMS offers features like message persistence and redelivery for reliable message handling.

## 10. Write an application code to run on the GAE platform for backup storage of a large amount of your personal, family, or company data and records such as photos, video, music, sales receipts, documents, news media, inventory, market records, financial matters, supply chain information, human resources, public data sets, and so forth. Note that strict privacy protection is required here. Minimizing the storage cost is another objective function to achieve. You should explain your code development experience and report the results in using the GAE platform.

## A. GAE Application for Backup Storage with Privacy and Cost Efficiency

**Development Experience:**

While I cannot directly access and develop code, I can provide a design outline and discuss considerations for a GAE application focused on secure and cost-effective data backup. My knowledge base includes various programming languages and frameworks, making me familiar with GAE development concepts.

**Application Design:**

1. **Data Upload:**
   * Users upload data through a secure web interface or API.
   * Client-side encryption can be implemented using libraries like AES-256 to encrypt data before upload.
2. **Data Storage:**
   * **Cloud Storage:** The primary storage option. It offers object-level access control (ACLs) for granular privacy control. Data can be stored in different storage classes based on access frequency (e.g., Standard Storage for frequently accessed data, Coldline Storage for infrequently accessed data).
   * **Cloud Datastore (Optional):** Consider Datastore for small, frequently accessed metadata associated with the backup files (e.g., filenames, timestamps). This can minimize Cloud Storage access charges.
3. **Data Access:**
   * Users authenticate through a secure mechanism (e.g., Google Sign-In) before accessing their data.
   * Access control lists (ACLs) restrict access to authorized users only.
4. **Cost Optimization:**
   * **Storage Classes:** Utilize different Cloud Storage classes based on access frequency.
   * **Lifecycle Management:** Set up lifecycle rules to automatically transition less frequently accessed data to cheaper storage classes.
   * **Cloud Functions:** Utilize serverless functions for tasks like data encryption/decryption, triggered by upload events. This minimizes persistent server costs associated with App Engine instances.

**Code Considerations:**

* **Security Libraries:** Use Google Cloud KMS (Key Management Service) to manage encryption keys securely.
* **Authentication:** Implement user authentication using Google Sign-In or other secure mechanisms.
* **Error Handling:** Implement robust error handling to ensure data integrity and user experience.

**Results on GAE (Hypothetical):**

Due to the inability to directly run code, results cannot be reported. However, based on the design:

* **Privacy:** Client-side encryption and access control lists (ACLs) ensure data privacy.
* **Cost Efficiency:** Storage class tiering and serverless functions minimize storage and compute costs.

**Disclaimer:** This is a high-level design. Actual implementation requires in-depth development and security considerations.

**Additional Notes:**

* Consider GAE standard environment for this application as it offers better scalability and cost-effectiveness for data storage compared to the flexible environment.
* Explore integrating with third-party backup solutions that offer pre-built functionalities for data encryption, scheduling, and versioning.

## 