Mid sem notes -CC

Module 1-Cloud Computing Architecture

1. What is Cloud Computing?

Definition: for anything that involves delivering hosted services over the Internet.

Cloud computing is the delivery of computing services (storage, processing, networking, software) over the internet ("the cloud"), enabling on-demand access to shared resources without direct user management.

Core Characteristics:

1. **Storing/accessing data & programs** on remote servers (e.g., Dropbox for file storage).
2. **Internet-based computing** (services accessed via browsers/APIs).
3. **Resources provided as a service** (e.g., renting servers from AWS instead of buying physical hardware).
4. **Transparency, scalability, security & intelligent monitoring** (automatic resource allocation, threat detection).

Real-life Example:
Netflix uses AWS cloud services to stream content globally. Users access movies via browsers/apps (frontend), while AWS handles storage, servers, and security (backend).

A Service Level Agreement (SLA) is a formal contract between a service provider and a client that defines the expected level of service. In cloud computing, SLAs are crucial for setting clear expectations and ensuring accountability.

2. Cloud Architecture Foundation

Combines two paradigms:

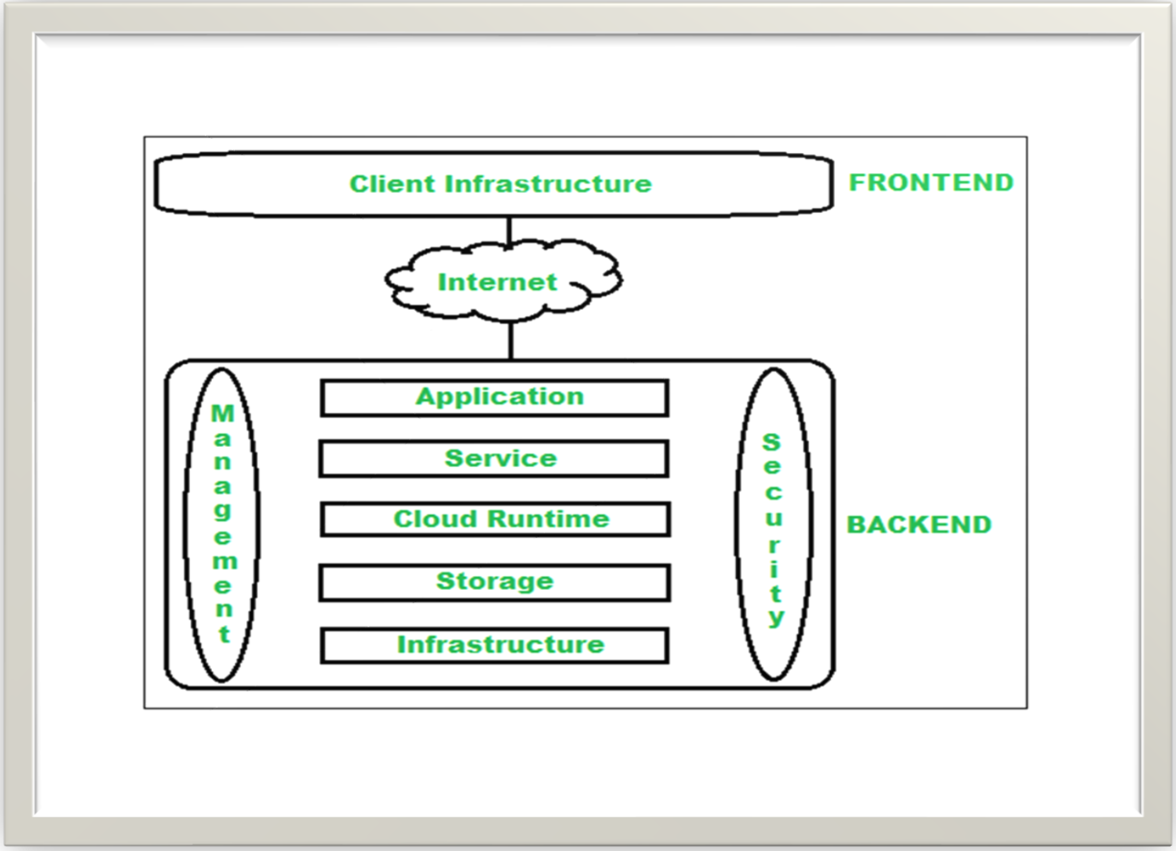
- **SOA (Service-Oriented Architecture)**:
  - Breaks down services into reusable components (e.g., "authentication service" used by multiple apps).
- **EDA (Event-Driven Architecture)**:
  - Responds to real-time events (e.g., processing a payment triggers an order-confirmation email).

Example: Uber uses SOA for modular services (maps, payments) and EDA to dispatch drivers when ride requests occur.

When a Cloud is made available in a pay-as-you-go manner to the public… The service bein
is Utility Computing."
This statement means that cloud providers (like AWS, Azure, Google Cloud) are operating lik
utility company, and their product—computing power—is a utility similar to electricity, wate
natural gas.

3. Cloud Architecture Components

Divided into **frontend** (client-facing) and **backend** (cloud infrastructure):



Frontend

- **Definition**: Interfaces users interact with to access cloud services.
- **Components**:
    1. **Client Infrastructure**:
        - Applications/GUIs used to access the cloud (e.g., web browser, mobile app).
        - *Example*: Using Chrome to access Google Docs.
    2. **User Interfaces**:
        - Dashboards, APIs, or command-line tools (e.g., AWS Management Console).

Backend

- **Definition**: The cloud itself, managing resources, security, and data.
- **Components**:
  1. **Application**:
     - Software/platform accessed by users (e.g., Salesforce CRM).
  2. **Service**:
     - **SaaS** (Software as a Service): Ready-to-use apps (e.g., Gmail).
     - **PaaS** (Platform as a Service): Development platforms (e.g., Heroku for app deployment).
     - **IaaS** (Infrastructure as a Service): Virtualized hardware (e.g., AWS EC2 virtual servers). eg. google and kaggle offers GPU in colab notebooks
  3. **Runtime Cloud**:
     - Execution environment for apps (e.g., Java apps running on Google App Engine).
  4. **Storage**:
     - Scalable storage (e.g., Amazon S3 for storing user files).
  5. **Infrastructure**:
     - Hardware/software (servers, virtualization, network devices).
  6. **Management**:
     - Coordinates resources (e.g., auto-scaling in Azure during traffic spikes).
  7. **Security**:
     - Tools like firewalls, encryption (e.g., AWS IAM for access control).
  8. **Database**:
     - Managed databases (e.g., Google Cloud SQL for structured data).
  9. **Networking**:
     - Connectivity services (e.g., AWS VPC for isolated cloud networks).
  10. **Internet**:

        - Bridge between frontend and backend.

Real-life Workflow:
A user uploads a photo to Instagram (frontend). The backend processes it:

- Storage saves the image (AWS S3).
- Database records metadata (Google Cloud SQL).
- CDN (Networking) delivers it globally.

---

4. Benefits of Cloud Architecture

Benefit	**Description**	**Example**
Simplifies System	Abstracts complexity; single interface for users	AWS Management Console controls all services
Improves Data Processing	Scalable compute for big data tasks	Spotify analyzes user data for recommendations
High Security	Centralized mechanisms (encryption, monitoring)	Bank apps use Azure Security Center
Modularity	Independent components for easy updates	Updating a payment service without downtime
Disaster Recovery	Automated backups across regions	Slack restores data after outages via Google Cloud
Accessibility	Access services anywhere via internet	Remote teams collaborate on Microsoft 365
Cost Reduction	Pay-as-you-go model; no physical hardware	Startups use AWS instead of data centers

Benefit	Description	Example
Reliability	99.9% uptime SLAs	Netflix streams 24/7 via AWS
Scalability	Instantly handle demand spikes	Airbnb scales servers during holiday seasons

5. Real-World Applications

- **Healthcare**: Hospitals use **SaaS** (e.g., Epic EHR) for patient records with **backend security** (HIPAA compliance).
- **E-commerce**: Shopify (**PaaS**) hosts online stores; scales during Black Friday sales.
- **IoT**: Smart home devices send data to **cloud storage** (e.g., Google Cloud IoT) for analysis.

Key Takeaways:

- **Frontend** = User access points (GUI, apps).
- **Backend** = Cloud infrastructure (services, storage, security).
- Cloud architecture enables **flexibility**, **cost savings**, and **innovation** (e.g., AI/ML via cloud GPUs).

Control Automation

Four functional areas :
Self-Configuration
Automatic configuration of components.

Self-Healing
Automatic discovery, and correction of faults.

Self-Optimization
Automatic monitoring and control of resources to ensure the optimal functioning with re
the defined requirements.

Self-Protection
Proactive identification and protection from arbitrary attacks.

Detailed Notes on Cloud Computing Framework

1. What is a Cloud Computing Framework?

Definition:
A structured approach providing tools and technologies to *design, deploy, manage, and optimize* cloud-based applications and services. It acts as a blueprint for building cloud solutions.

Core Components:

Component	**Purpose**	**Examples**
Development Tools	Build/test cloud apps	AWS Cloud9, Azure DevOps
Middleware	Connects apps/data across cloud environments	Red Hat JBoss, MuleSoft Anypoint Platform n8n
Administration Software	Monitors/manages cloud resources	VMware vRealize, IBM Cloud Pak

Real-World Analogy:
Like a "factory assembly line" for cloud apps:

- **Development tools** = Raw materials (code, APIs)
- **Middleware** = Conveyor belts (data integration)
- **Admin software** = Quality control robots (performance monitoring)

2. Framework Phases BAE

Phase 1: Analysis

Evaluates feasibility and requirements:

- **Cost Analysis**:
  - *Example*: Netflix migrated to AWS to save $1B/year vs. maintaining data centers.
- **Security Analysis**:

- *Example*: Banks use Azure Security Center to audit compliance (GDPR, HIPAA).

- **Accounting Analysis**:
  - Tracks usage-based billing (e.g., Google Cloud's per-second VM pricing).
- **Risk/Benefit Analysis**:
  - *Trade-off*: Cloud scalability vs. dependency on internet connectivity.

Phase 2: Evaluation

Assesses solutions against business needs:

Evaluation Type	Focus	Real Application
Investment	ROI of cloud migration	Dropbox saved $75M over 2 years by moving to AWS
Risk	Downtime/data loss probability	Slack uses AWS multi-region backups for 99.99% uptime
ROI	Cost savings vs. on-premises	Capital One reduced TCO by 30% with AWS
Scenario	"What-if" testing (e.g., traffic spikes)	Zoom scales servers during global events
Security	Vulnerability assessments	Shopify uses automated penetration testing

3. Why Businesses Adopt Cloud Frameworks

Key Drivers:

1. **Cost Reduction (61% of large IT companies)**:
   - *Mechanism*: Pay-as-you-go model eliminates upfront hardware costs.
   - *Example*: Airbnb avoids $200M+ in data center expenses using AWS.

2. **Enhanced Security**:
   - *Mechanism*: Enterprise-grade firewalls + encryption + access controls.
   - *Example*: Pfizer stores COVID vaccine data in IBM Cloud with end-to-end encryption.
3. **Remote Work Enablement**:
   - *Mechanism*: Centralized cloud access from any device/location.
   - *Example*: GitLab's 1,500+ remote employees collaborate via Google Workspace.
4. **High Reliability**:
   - *Mechanism*: Geographically distributed servers + failover systems.
   - *Example*: Salesforce guarantees 99.999% uptime for financial clients.
5. **Elastic Scalability**:
   - *Mechanism*: Instantly add/remove resources (CPU, storage).
   - *Example*: Instagram handles 4M+ uploads/hour by auto-scaling on AWS.

4. Real-World Industry Applications

Industry	**Cloud Framework Use Case**	**Outcome**
Healthcare	Epic EHR on Azure: Secure patient data sharing	250M+ patient records accessed globally
Retail	Shopify (PaaS): E-commerce store hosting	1M+ stores scale during Black Friday sales
Finance	Capital One on AWS: Fraud detection algorithms	Reduced false positives by 70%
Manufacturing	Siemens MindSphere (IoT cloud): Predictive maintenance	30% fewer machine failures

5. Challenges & Mitigations

Challenge	Framework Solution
Data Privacy Concerns	Encryption-at-rest + regional compliance (e.g., EU data in Azure Germany)
Vendor Lock-in	Hybrid/multi-cloud strategies (e.g., Anthos on AWS + GCP)
Skill Gaps	Managed services (e.g., AWS Managed Services)

Service Models

Cloud Service Models (SaaS, PaaS, IaaS)

1. Software as a Service (SaaS)

Definition:

- Software delivered over the internet on a subscription basis.
- **Key Feature**: No local installation/maintenance (accessed via web browser).
- **Billing Model**: Pay-as-you-go.
- **Nicknames**: *Web-based software*, *On-demand software*, *Hosted software*.

Real-World Examples:

- **Salesforce**: CRM for sales teams
- **Microsoft Office 365**: Productivity suite
- **Dropbox**: Cloud file storage

Advantages:

Benefit	Explanation	Real Application
Cost-Effective	No hardware costs; pay per user/month	Startups use Gmail instead of Exchange servers
Zero Installation	Accessible via browser instantly	Doctors access patient records on Epic EHR from any hospital computer
Automatic Updates	Provider handles patches/upgrades	Adobe Creative Cloud users get new features automatically

Benefit	Explanation	Real Application
Accessibility	Use anywhere with internet	Remote teams collaborate on Google Docs
Scalability	Add/remove users instantly	Zoom scales licenses during conference season

Disadvantages:

1. **Limited Customization**
   - *Issue*: Can't modify core functionality (e.g., Shopify stores can't alter checkout code).
   - *Workaround*: Use APIs for partial integrations (e.g., connect Mailchimp to Salesforce).
2. **Internet Dependency**
   - *Impact*: Offline work impossible (e.g., construction sites with poor connectivity can't access Autodesk BIM 360).
3. **Security Risks**
   - *Incident*: 2023 Microsoft breach exposed SaaS customer data.
   - *Mitigation*: Enable MFA and data encryption (e.g., Box Enterprise Key Management).
4. **Data Control Concerns**
   - *Regulatory Challenge*: HIPAA-compliant healthcare orgs avoid SaaS for sensitive patient data processing.

2. Platform as a Service (PaaS)

Definition:

- Cloud platform for developing, testing, and deploying applications.
- **Key Feature**: Manages OS, servers, storage – developers focus *only* on code.
- **Analogy**: Renting a fully equipped kitchen (PaaS) vs. building one (IaaS).

Real-World Examples:

- **AWS Elastic Beanstalk**: Deploy web apps without server config
- **Google App Engine**: Build scalable Python/Java apps
- **Heroku**: Deploy container-based apps

Advantages:

Benefit	Explanation	Real Application
Faster Development	Pre-configured tools (DBs, SDKs, runtimes)	Spotify built backends in days using Google App Engine
Cost Reduction	No server maintenance costs	Duolingo saved 60% vs. on-premises servers
Lifecycle Support	End-to-end: build → test → deploy → update	Netflix uses PaaS for continuous deployment
High-Level Abstraction	Focus on business logic, not infrastructure	Airbnb developers ignore server scaling rules

Disadvantages:

1. **Vendor Lock-in**
   - *Issue*: Apps built on Salesforce PaaS can't easily migrate to Azure.
   - *Solution*: Use Kubernetes for hybrid cloud portability.
2. **Limited Infrastructure Control**

    - *Consequence*: Can't optimize OS/kernel for high-frequency trading apps.

3. **Provider Dependency**
    - *Outage Impact*: 2021 AWS outage paralyzed PaaS users like Slack for 5 hours.

---

3. Infrastructure as a Service (IaaS)

Definition:

- Virtualized computing resources (servers, storage, networking) over the internet.
- **Key Feature**: Full control over OS/apps; provider manages *physical hardware only*.
- **Nickname**: *Hardware as a Service (HaaS)*.

Real-World Examples:

- **AWS EC2**: Virtual servers
- **Azure Virtual Machines**: Windows/Linux VMs
- **Google Cloud Storage**: Scalable object storage

Advantages:

Benefit	Explanation	Real Application
Cost Efficiency	Pay per hour/GB (e.g., $0.10/hr for Linux VM)	Pinterest saved $20M/year migrating to AWS
Hosting Flexibility	Custom web server configurations	NASA hosts Mars imagery on AWS with custom CDN

Benefit	Explanation	Real Application
Enterprise Security	Better than most on-premises setups	JPMorgan uses Azure for FedRAMP-compliant banking apps
Zero Maintenance	Provider handles hardware failures/upgrades	Tesla avoids data center staff for Autopilot training

Disadvantages:

1. **Steep Learning Curve** Non technical can't handle
   - *Challenge*: Requires DevOps skills (e.g., configuring AWS VPC networks).
   - *Solution*: Use managed services like AWS Lightsail.
2. **Security Responsibility**
   - *Shared Model*: Provider secures hardware; *you* secure OS/apps/data.
   - *Mistake*: 2022 Uber breach occurred due to misconfigured IAM permissions.
3. **Geographic Limitations**
   - *Restriction*: Chinese companies can't use AWS in Shanghai due to GFW policies.

Comparison: SaaS vs. PaaS vs. IaaS