Phase 1

1️⃣ Capacity (Agility, Elasticity, Scalability)

- **Capacity** = How much your system can handle.
- **Agility** = How quickly you can adjust resources.
- **Elasticity** = Ability to **grow or shrink automatically** based on demand.
- **Scalability** = Ability to **handle more load** without breaking.

Example:
Imagine a website:

- On weekdays, 50 users visit → one small server is enough.
- On weekends, 5000 users visit → you add more servers automatically (elasticity).

2️⃣ Availability

- **Availability** = Your service stays **up and running**, even if something goes wrong.
- Think of it like a **backup plan** for failure.

Example:

- You have a website hosted in one data center. If that data center goes down, your site goes offline → low availability.
- If you have two data centers and traffic switches to the second when the first fails → high availability.

3️⃣ Blast Radius

- **Blast radius** = How much damage happens if something fails.
- Smaller blast radius → less impact when failure happens.

Example:

- One giant server crashes → your entire app goes down → huge blast radius.
- Many small servers → if one crashes, only part of your app is affected → small blast radius.

4️⃣ Disaster Recovery

- **Disaster recovery** = How quickly you can **restore your service after a big failure**.

Example:

- Your data center floods → you switch to another region and your website is back online.

5️⃣ Vertical Scaling vs Horizontal Scaling

Vertical Scaling (Scale Up)

- Add **more power to one server** (CPU, memory).
- **Problem:**
  - Often requires **downtime** to upgrade.

  - Not all apps can handle huge servers.
- **Example:**
  - Upgrade a small server from 2 CPUs → 16 CPUs → your server must restart.
  - If it crashes during upgrade → downtime.

Horizontal Scaling (Scale Out)

- Add **more servers** instead of making one bigger.
- **Better for cloud** → handles failure and traffic smoothly.
- **Example:**
  - Your website has 2 small servers → traffic increases → add 3 more servers automatically.
  - Traffic decreases → remove 1 server.
  - You **never go below 2 servers** to ensure availability.

Rule of thumb in cloud: Horizontal scaling is safer, more flexible, and keeps your service always available.

✅ Simple Analogy

- **Vertical Scaling** = One big pizza → hard to eat, and if it burns, you lose all.
- **Horizontal Scaling** = Many small pizzas → easy to share, if one burns, others are still fine.

6️⃣Capital Expenditure (CapEx)

- **What it is:** Buying physical resources upfront, which you own and depreciate over time.
- **Example in the real world:** Buying servers, storage devices, or networking equipment for your company's on-premises data center.
- **Simple analogy:** Like buying a car—you pay upfront, and it's yours for years.

In cloud context (Phase 1 / Azure labs):

- On-premises equivalent: If you wanted to practice Azure security but instead bought your own physical servers, firewalls, and networking gear to test labs.
- You spend **a lot upfront**, but you **own the hardware**.

7️⃣Operational Expenditure (OpEx)

- **What it is:** Paying only for what you use, usually subscription or consumption-based. No big upfront cost.
- **Example in the real world:** Using a streaming service like Netflix—you pay monthly only for what you watch.
- **Simple analogy:** Like renting a car—you pay only when you drive it.

In cloud context (Phase 1 / Azure labs):

- Using Azure free-tier or pay-as-you-go services for VMs, Storage, Key Vault, and AKS in your labs.
- You **pay based on usage**, scale up or down, and don't need to buy servers or networking devices upfront.

- Ideal for learning: You can practice security labs **without spending thousands on hardware**.

Key difference (super simple)

CapEx	OpEx
Buy and own (servers, devices)	Rent or pay-as-you-go (Azure services)
High upfront cost	Low/no upfront cost
Fixed capacity (limited by hardware)	Flexible capacity (scale up/down anytime)
Depreciates over time	Costs treated as ongoing expense