1. **Agenda**:
   1. What is **containerization technology** heavily used in **Docker Eco-System**?
2. First, Let’s discuss what is virtual machine before discussing about what is container and how does it work?
   1. As you know, **Cloud Providers** such as **AWS** provides you a **Virtual Server** with the concept of **Virtual Machine**.  
      AWS calls it **EC2** instance which is **E**lastic **C**ompute **C**loud.
   2. Now suppose, you want to deploy your app or install any software such as DBMS, WebLogic, what you used to do in traditional manner (during the concept of virtual machine)?

You buy a virtual server with 8GB RAM, 500GB Hard-Disk. Then along with the app, you install all the dependencies needed by that app such as Java, Spring Dependencies, MySQL, Tomcat Server.

**For Example (With Traditional Approach)**:

If you have three services such as Loan, Account, Card, you buy three virtual servers such as VM1, VM2, VM3 from Cloud Provider AWS.   
Then you install all these services along with their dependencies on the three virtual servers separately.

* 1. **Problem with this approach**:

|  |
| --- |
| **Problems with Virtual Server (Traditional Approach in Cloud without Containers)** |
| 1. **Cost**: You’ve to pay for three different virtual machines which are not necessary for these small services. |
| 1. **Time & Cost**: Scaling up will take good amount of time as you need to buy a new virtual machine with OS installed with all its dependencies, then install your dependencies for the app you want to run then install your app such as DBMS, WebLogic or deploy your web-app. |
| 1. **Time**: Restarting a service also takes a good amount of time as each virtual machine has its own OS and OS takes time to restart. |

But Virtual machine style is good for monolithic app as at max you have one app you want to maintain either manually or you can automate using DevOps as per your need.  
But if you say you have 100 microservices and you’re going to buy 100 virtual machines, it is going to be a nightmare for me and it involves a lot of cost.

* 1. So, with all these issues, a new concept called container evolved that was greatly adopted by industries especially for building and deploying microservices.
  2. So, let’s see what containers are and how they are different from virtual machines.

1. **Let’s see the differences between Virtual Machine and Container**.  
   Diagram

   Description automatically generated
   1. **Container**:
      1. See in the diagram, there is a single Physical Server.
      2. On top of it, Host operating system (any like Linux, Windows).
      3. On top of it, we can install Docker Engine or any equivalent engine to create and manage our containers.
      4. Then Docker Engine is responsible to allocate resources (RAM, Hard-Disk) as per the demand by the containers.
      5. No Guest OS in any container. So easy to restart. So, scaling up/down is easy in case of containers.
   2. **Virtual Machine**:
      1. The Hypervisor is responsible to allocate RAMs, Hard-Disk.
2. In next lecture, let’s see what is container, role played by docker in this containerization technology.