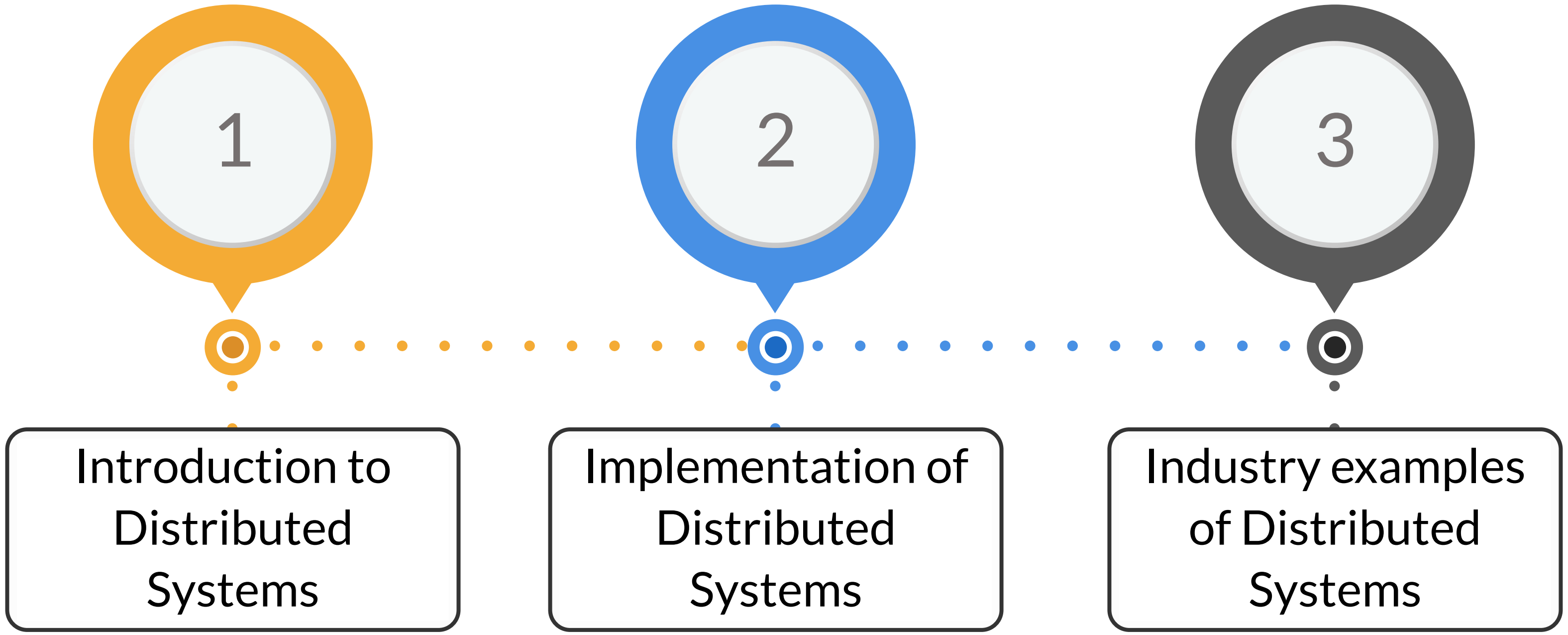
upGrad

Introduction to Distributed Systems

Session: Introduction to Distributed Systems

Instructor: Vishwa Mohan

MODULE INTRODUCTION



Introduction to Distributed Systems

Implementation of Distributed Systems

Industry examples of Distributed Systems

- What are distributed systems?
- Why do we need distributed systems?
- Expectations from a distributed system.

- Distributed systems architecture
- Challenges in implementation
- CAP theorem

- Case study on some well known distributed systems like Amazon S3, Apache Spark etc.

DISTRIBUTED SYSTEMS

- A distributed system is a collection of autonomous computing elements that appears to its users as a single coherent system.
- Distributed systems are used to solve problems that cannot be solved by a single system.
- These problems are broadly categorised into the following:

Computation

Storage

DISTRIBUTED SYSTEMS USE CASES

Computation

This seems like a trivial problem, but becomes complex when the volume and velocity of data increases to an extent that it cannot be efficiently handled by a single machine. Typical use cases include Machine Learning and Big Data problems.

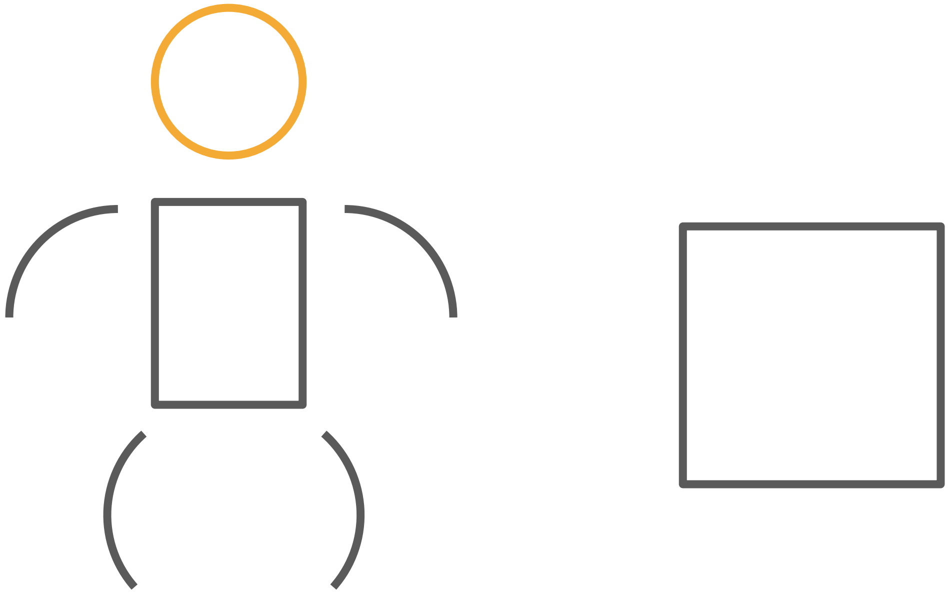
Storage

The storage in a single machine can only be expanded to a certain point. After that the cost of implementing a larger storage is so high that it becomes impractical.

DISTRIBUTED SYSTEMS - EXAMPLE

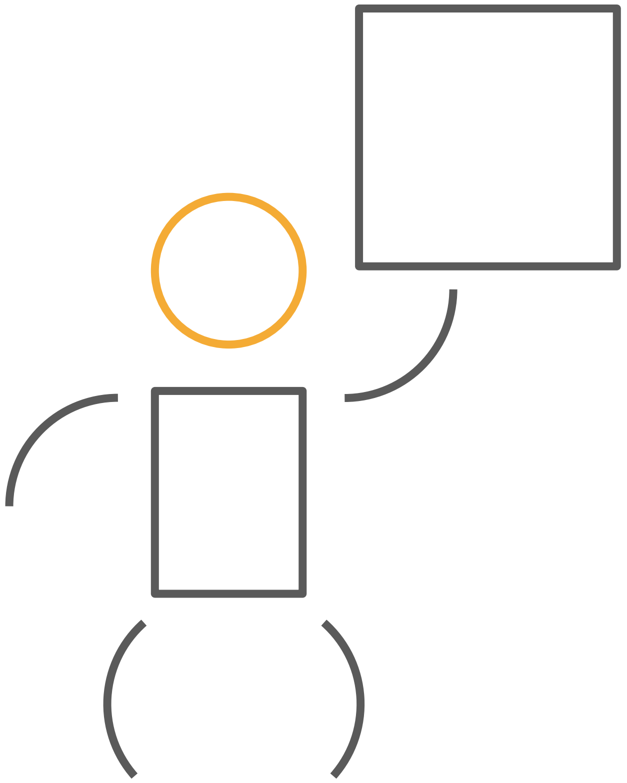
As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let's look at a simple example to understand this better:

Can Mr. X lift one block?



DISTRIBUTED SYSTEMS - EXAMPLE

As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let's look at a simple example to understand this better:

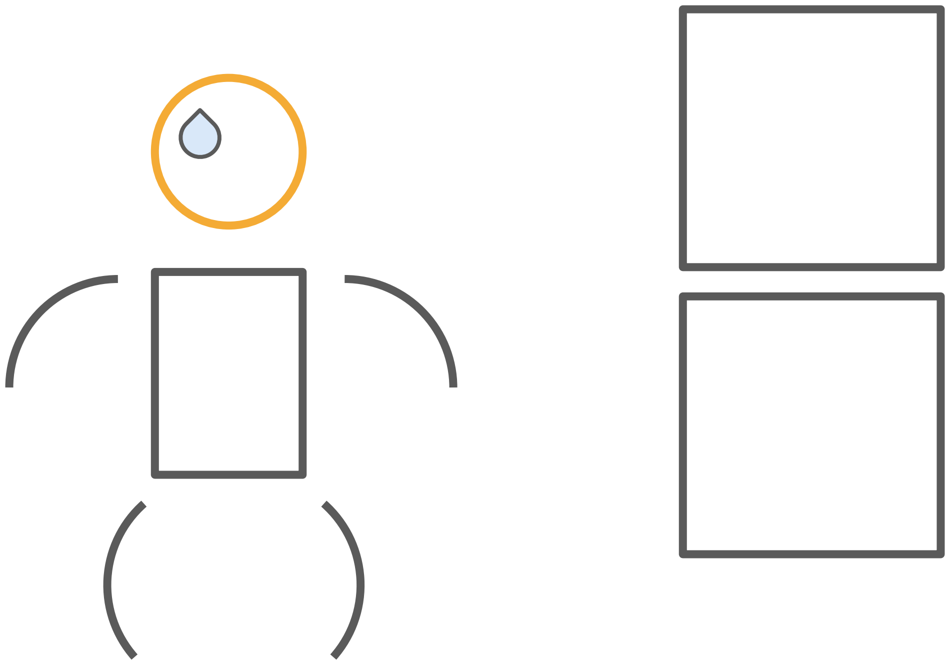


Can Mr. X lift one block?

Yes!

DISTRIBUTED SYSTEMS - EXAMPLE

As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let’s look at a simple example to understand this better:



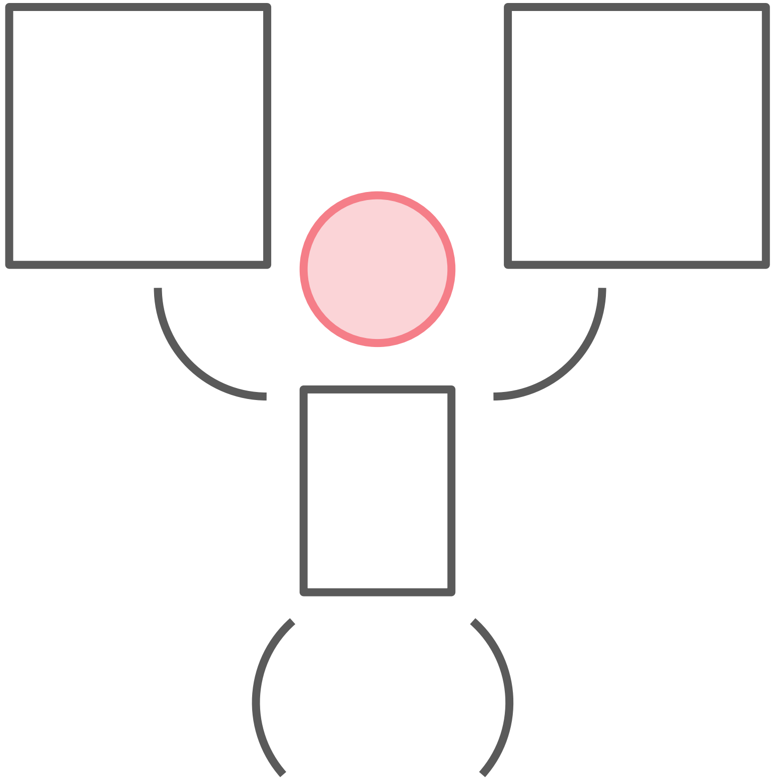
Can Mr. X lift one block?

Yes!

Can Mr. X lift two blocks?

DISTRIBUTED SYSTEMS - EXAMPLE

As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let's look at a simple example to understand this better:



Can Mr. X lift one block?

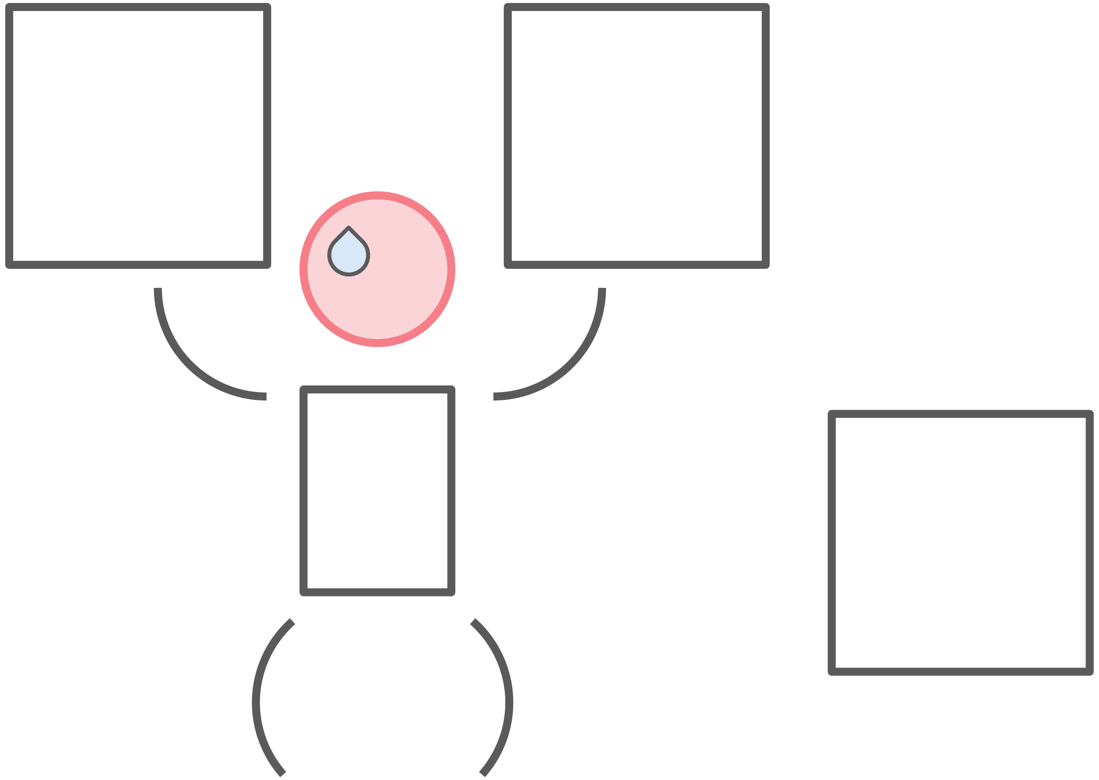
Yes!

Can Mr. X lift two blocks?

Yes?

DISTRIBUTED SYSTEMS - EXAMPLE

As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let's look at a simple example to understand this better:



Can Mr. X lift one block?

Yes!

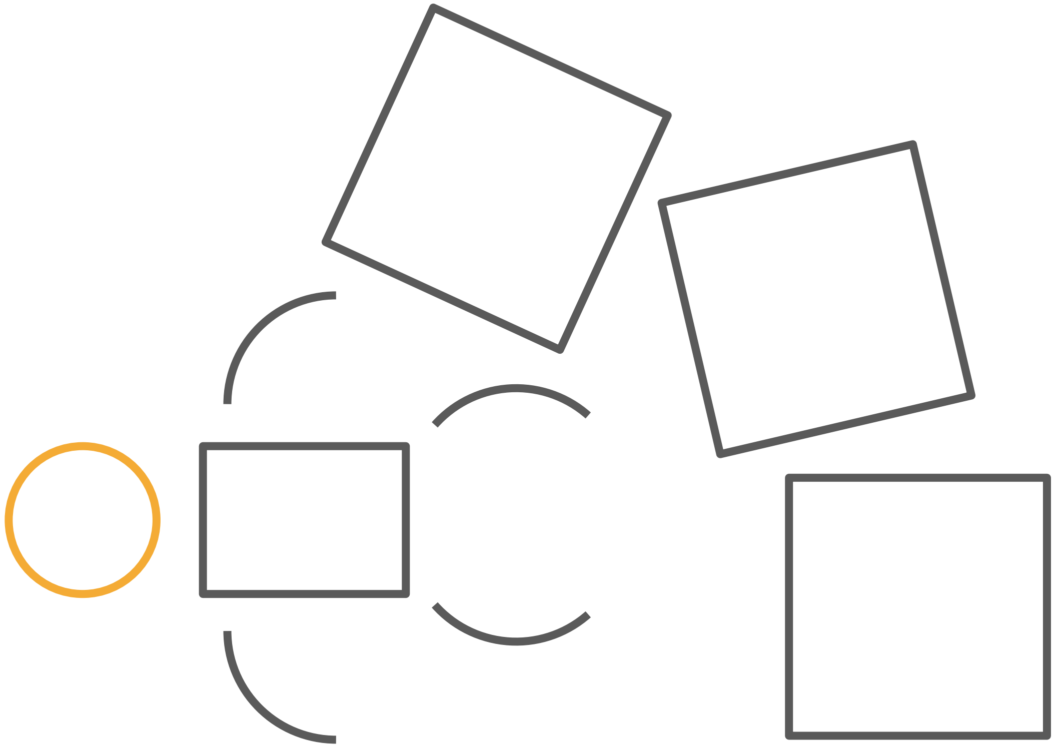
Can Mr. X lift two blocks?

Yes?

One more?

DISTRIBUTED SYSTEMS - EXAMPLE

As you know now, distributed systems are used when a problem becomes too big to be solved by a single machine. Let's look at a simple example to understand this better:



Can Mr. X lift one block?

Yes!

Can Mr. X lift two blocks?

Yes?

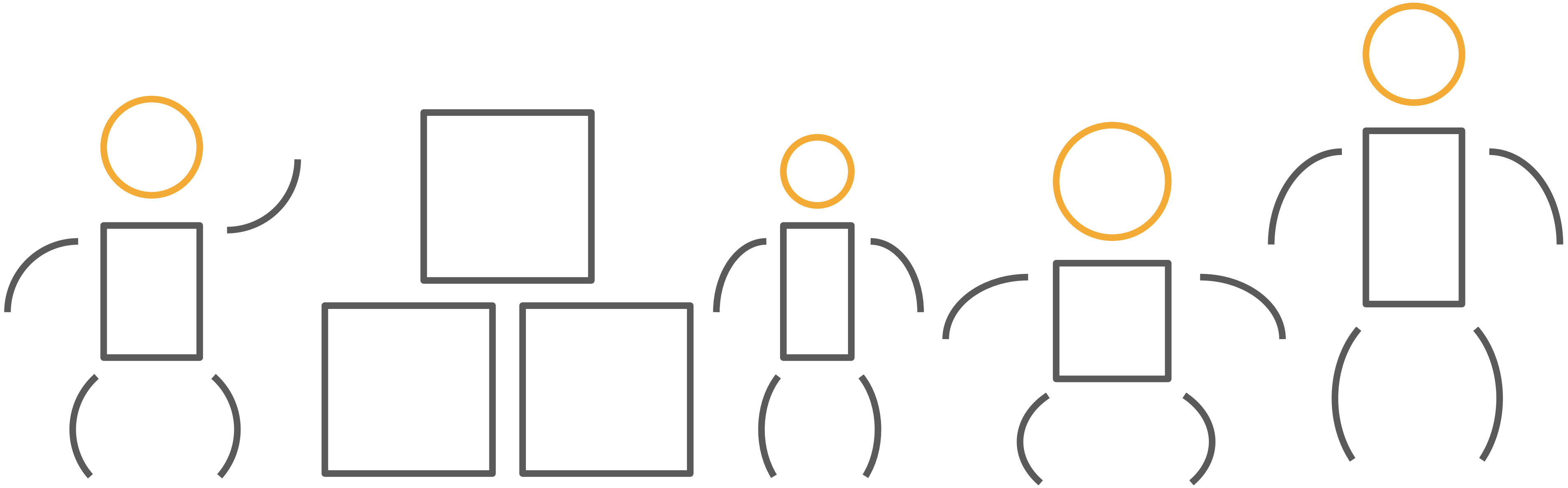
One more?

No

DISTRIBUTED SYSTEMS - EXAMPLE

Now, what if Mr. X called his friends to help him?

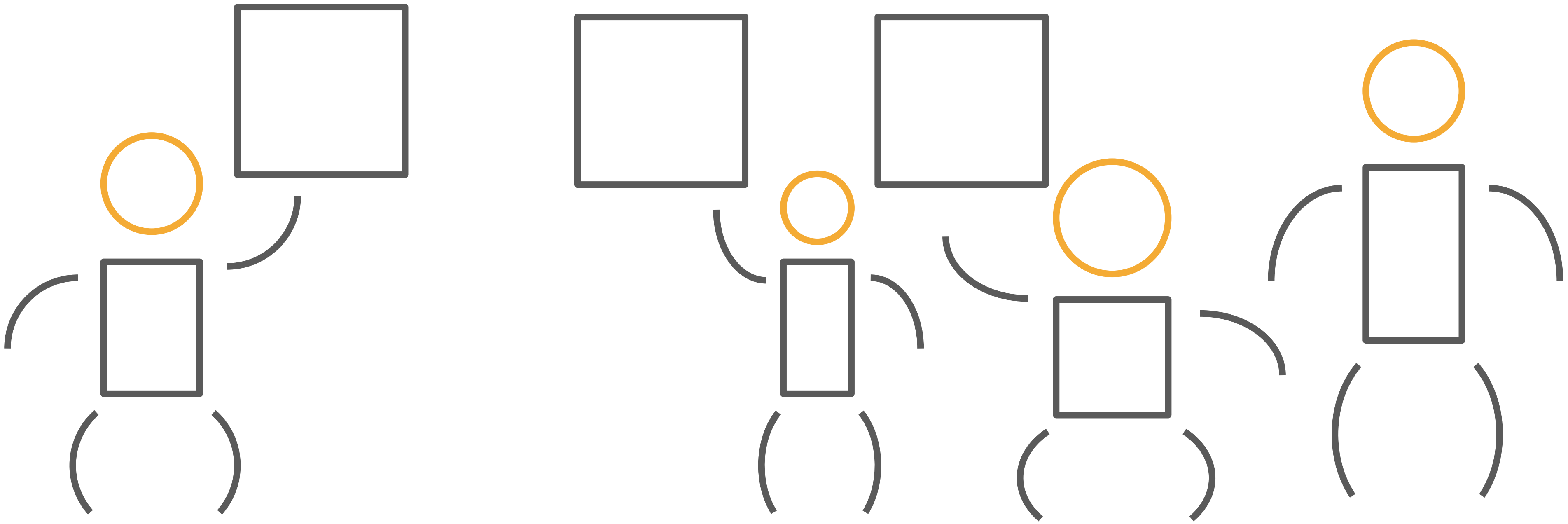
Can they lift three blocks?



DISTRIBUTED SYSTEMS - EXAMPLE

Now, what if Mr. X called his friends to help him?

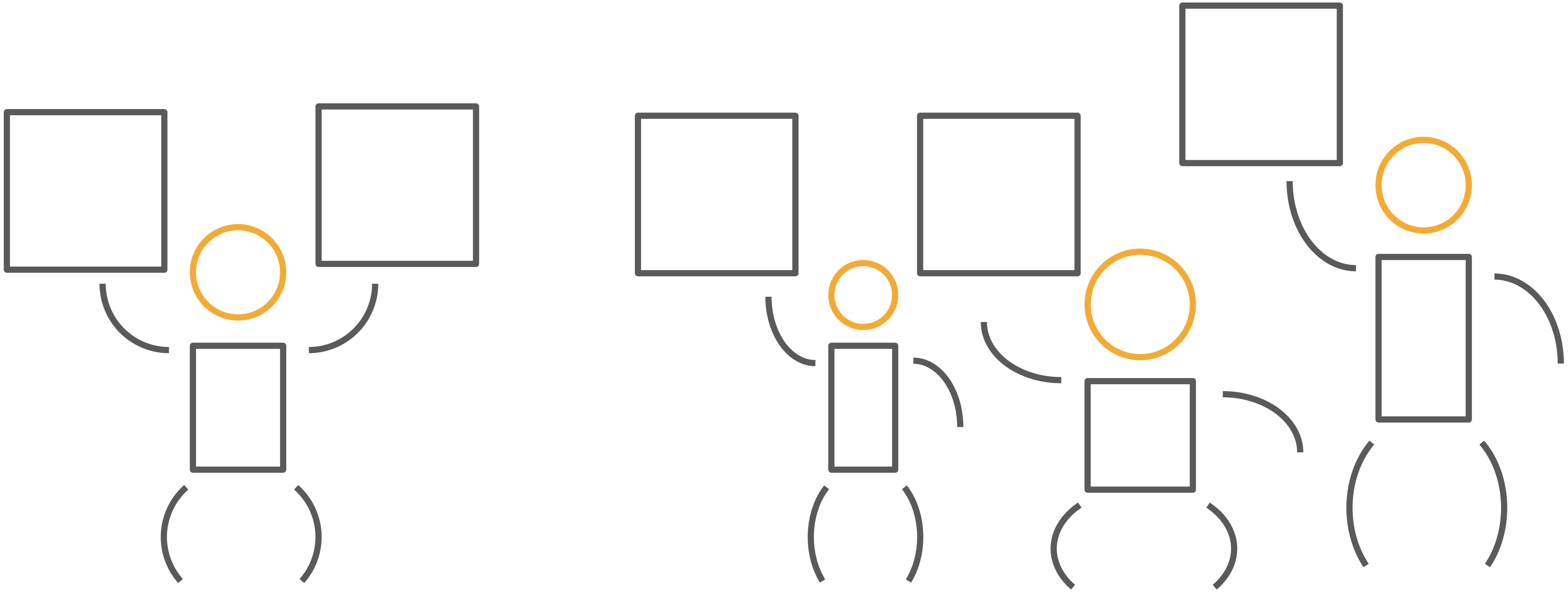
Can they lift three blocks? Yes!



DISTRIBUTED SYSTEMS - EXAMPLE

Now, what if Mr. X called his friends to help him?

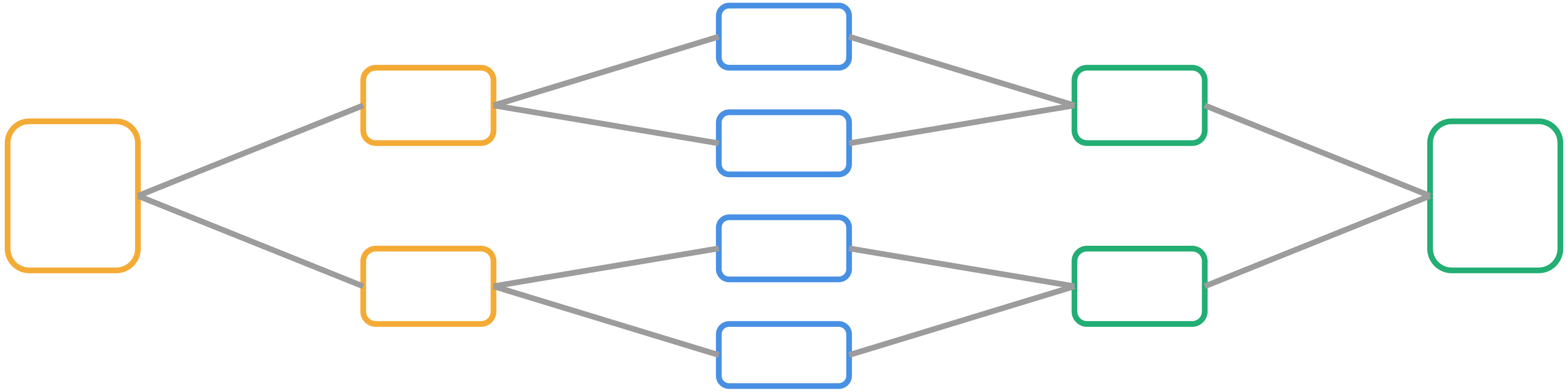
Can they lift three blocks? **Yes!** **And more!!**



DISTRIBUTED SYSTEMS - EXAMPLE

As we just saw, distributed systems are essentially based on the concept of divide and conquer:

- You take a problem that is too complex/huge for a single machine.
- Divide the problem into smaller sections manageable by a single machine.
- Distribute the workload among a number of autonomous machines and solve them.



WHY DO WE NEED DISTRIBUTED SYSTEMS?

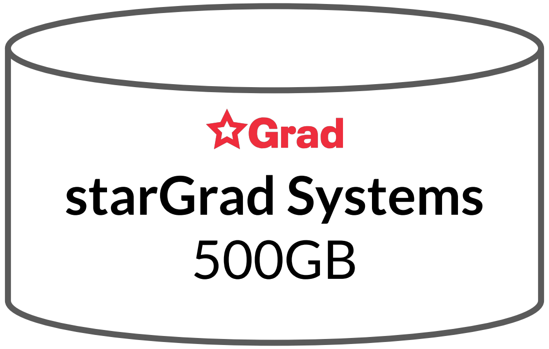
To better understand why we need distributed systems, let us study the following problem statement:

- You are an entrepreneur working on a startup called **starGrad.**
- It is a unique online higher education platform providing rigorous industry-relevant programs.
- Your website stores a ton of text-based content and also provides on-demand streaming services to your learners.
- Initially, you only provide services in your locality and your entire model is based on a **single centralised system with 500 GB of storage.**

WHY DO WE NEED DISTRIBUTED SYSTEMS?

Now owning to unique selling point of course, your starGrad is famous in no time.

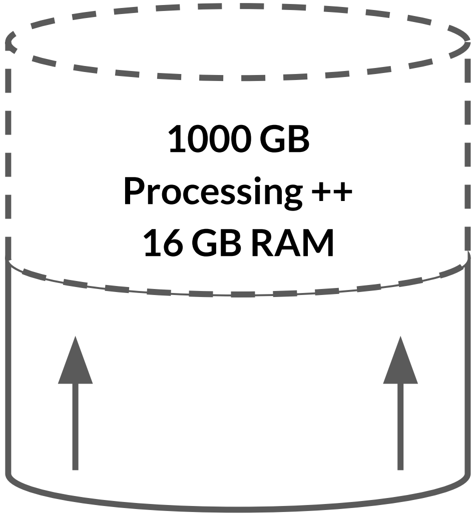
- The traffic in your website is growing rapidly, more and more learners are joining.
- As an excellent entrepreneur you also want to expand the number courses you are providing.
- The 500 GB of storage in your centralised system in not enough to hold the amount of content you intend to provide. You also need more computational power to handle the increased traffic.



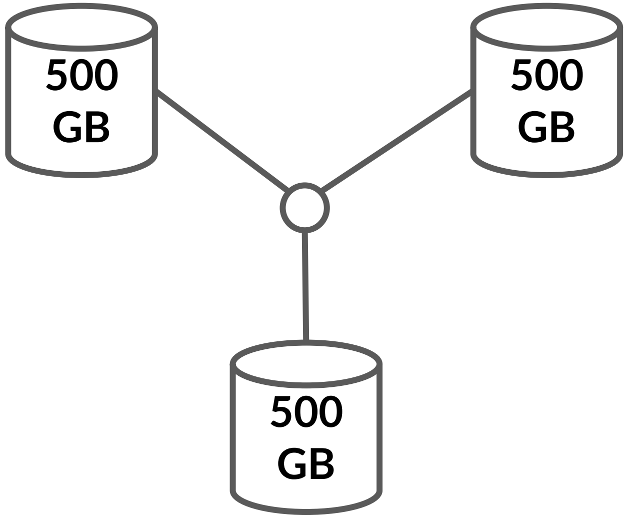
WHY DO WE NEED DISTRIBUTED SYSTEMS?

Given that your model is based on a centralised system, what can you do to scale your company?

1. Scale Vertically



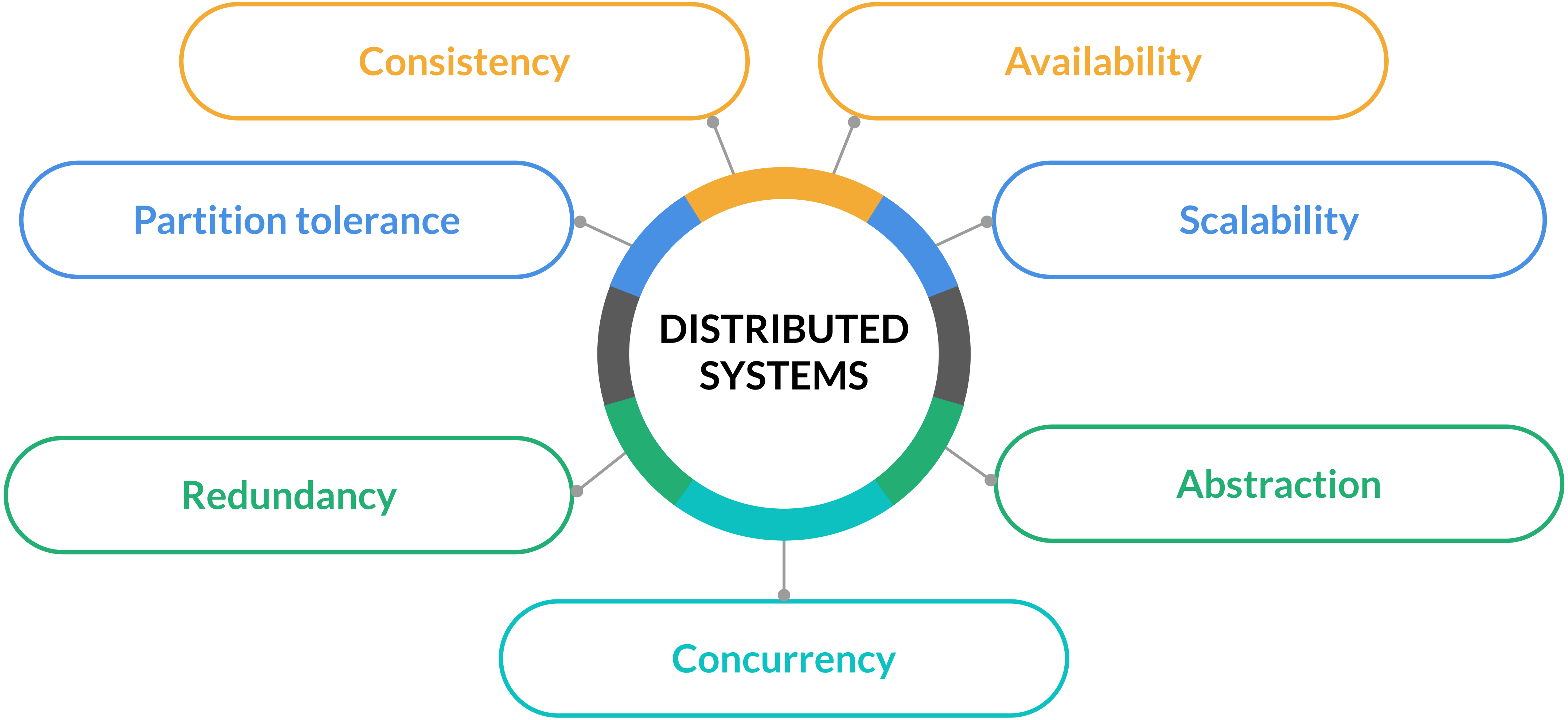
2. Shift to Distributed Systems



CENTRALISED VS DISTRIBUTED SYSTEMS

Centralised Systems	Distributed Systems
Limited storage capacity and computational capability	In theory no upper limit to storage and computational capabilities
Only vertically scalable which is very expensive after a limit and involves downtime	Both vertical and horizontal scalability, hence easily scalable
Single point of failure	No single point of failure
Not fault tolerant	Fault tolerant

EXPECTATIONS FROM DISTRIBUTED SYSTEMS



DISTRIBUTED SYSTEMS

Consistency

Availability

Partition tolerance

Scalability

Redundancy

Abstraction

Concurrency

REDUNDANCY

- **Redundancy** - Having more than one copy of the data.
- Helps in avoiding loss of data due to a new partition or node failure.
- For starGrad - Let’s say one of the machines in which your “Machine Learning” course content is stored crashes. If your distributed system has redundancy, it will have a backup copy of the course in another machine. This makes your model more reliable and failure proof.

WACOM

CONSISTENCY

- **Consistency** - At any given time, if two people request for the same data they should get the exact same result.
- Simply put, performing a read operation will return the value of the most recent write operation, causing all nodes to return the same data.
- For starGrad - Let’s say a learner reports a mistake in a graded question and you update your platform to rectify the error. Now, to be fair to everyone, the latest update in the question should be consistent for all the learners.

WACOM

AVAILABILITY

- **Availability** - The system should be responsive all the time.
- Distributed systems should ensure its availability, i.e. even if a component of the system fails, the system does not fail, and there is another component that replaces it and keeps the system running.
- For starGrad - Let’s say your nodes in Delhi are down every Friday for maintenance. This should not mean that the learner cannot access the platform on those days. Distributed system allows you to ensure platform availability at all times, which is essential for a great learning experience.

WACOM

PARTITION TOLERANCE

- **Partition tolerance** - The system should be responsive and working even if one partition is disconnected from the rest due to network/communication issues.
- A system that is partition-tolerant can sustain any amount of network failure that does not result in a failure of the entire network.
- For starGrad - Once you move to a distributed systems, your content will be spread across different machines in far away physical locations. Let’s say there are some network issues in Mumbai. This creates a partition in your network. If all the content of a particular course and its backups are stored in Mumbai, your system will not be partition tolerant.

WACOM

SCALABILITY

- **Scalability** - The system should be able to easily scale with increase in workload.
- By adding cheap commodity hardware, distributed systems can easily implement horizontal scaling when the workload increases.
- For starGrad - To accommodate more learners and to expand your current list of programs, your platform must be easily scalable. Distributed systems allows you to easily scale up your storage and computation capabilities by adding more machines to your system.

WACOM

CONCURRENCY

- **Concurrency** - The system should be able to perform parallel processing/computation.
- Concurrency in computation allows distributed systems to perform complex tasks in a fast and efficient manner.
- For starGrad - As the platform grows and the number of learners increase, the computational requirements will also rise. Through concurrency, distributed systems can easily tackle this issue.

WACOM

ABSTRACTION

- **Abstraction** - The system should seem like a single system to the user.
- Distributed systems provide a **global view** of all the underlying machines as a single machine and **hide the internal working** of the system from the end users.
- For starGrad - The learner should see your platform as single isolated service. He/she should not have to request data from the Bangalore server for the “Machine Learning” course and from the Mumbai server for the “Web Development” course.

WACOM

SUMMARY

01 What are distributed systems?

02 Why do we need distributed systems?

03 Centralised Vs Distributed systems

04 Expectations from distributed systems

Thank You