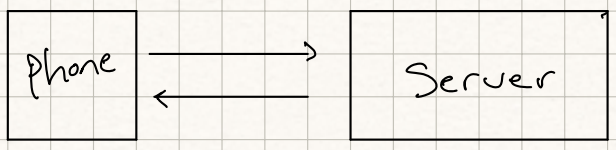



* some code is running on this computer → code is like any other program
  1. takes an input
  2. gives out an output

* lets say people are willing to pay you to use this code...
  - ↳ so instead of handing out servers we expose our code using an API (application programming interface)
    - ↳ the api will take in a request and then the api will send back a response

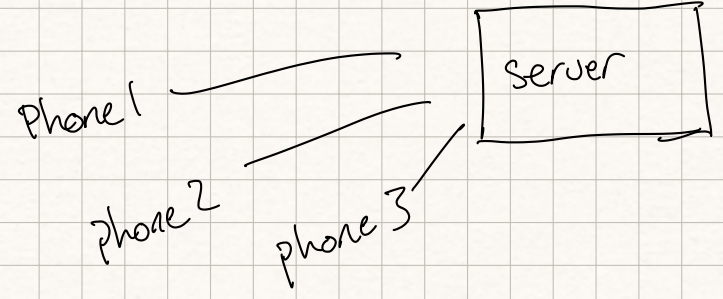
* when we set up this server it could require a
  - Database
  - configure the endpoints
  - what happens if the server goes down?

what's the difference between a desktop & cloud?

* the cloud is a set of computers working together
  - ↳ ex) AWS

AWS takes care of maintaining your service

w/ AWS taking care of maintenance we can focus on the business side... if we have a large number of users what do we do?




→ with an increase in users our service will start to slow down

so what do we do?

# Scalability

* to ensure our service maintains fast speed we have 2 solutions

① buy bigger machines
1 server w/ more ram
hard drive
CPU
etc

↓

this is called
vertical scaling

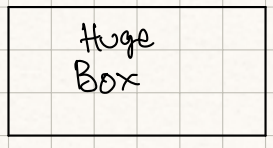
② buy more machines

* since there are more machines large amounts of requests can be handled and the service remains fast

↓

horizontal scaling

## vertical scaling




- No load balancing required b/c it's a single machine
- single point of failure
- Interprocess Communication (fast)
- Data Consistency is an issue
  * lose transactional guarantee
  you lose data when bouncing it from server to server
- This scales almost linear...
  as users increase so do servers

## horizontal scaling

- Load Balancing required to evenly distribute the work between all the machines
- If a single machine fails, you can redirect the requests elsewhere
- Network calls (RPC) - (slow)
  Remote Procedure calls
- Data is consistent b/c it runs on a single machine
- Hardware limit at some point can't just keep increasing

How do we pick?

We choose both → we take good qualities from vertical scaling

- fast inter process communication
- data consistency

→ we take good qualities from Horizontal Scaling

- scales well
- resilient → back-up servers there in case of a crash

The hybrid solution is essentially horizontal scaling only where each machine is maxed out

* initially you should start vertically, then as your users grow ... so do the numbers of servers.