**System Design**

**System Design is the process of designing the architecture, components, and interfaces for a system so that it meets the end-user requirements.   
  
  
Why Learn System Design?**

**System design is important for anyone who wants to build a robust, scalable, and efficient software application. Whether you are building a small-scale application or a large one, understanding system design allows you to architect solutions that can handle real-world complexities.**

**Scalability**

**Mobile -> request -> server -> response  
client and server**

**Single user no problem if request get from 1000 user it will handle then now 100000 user sent to server**

**Issue came Scaling   
  
Scaling  
  
if 1 user using that server will give response and if 100000 user using that time alose the give response with same performance and sability will provide that server  
  
2 types of Scaling :  
horizontal scaling and vertical scaling   
  
Server is a system if that system have move ram it will handle more request   
  
eg if mobile phone have 2 gb ram it handle 5 applications at same time and it will lage  
but if system have 8 gb it will handle easly   
same comcept for server also   
  
we need to buy server with concept but AWS server is cost high   
so we cost effective and same need to handle all the server so we use Scaling Concepts   
  
  
  
VS – eg I share server link to friends it will run1000 but now 5000 users need to increase ram and storeage vg – 2 gm 20gm ssd to 4gb ram and 40gb ssd we can improve – upgrade machine.**

**HS- we can buy 2 system same 2 gm and 20 ssd  
  
we can extend the machine .**

**Dis Advantages :**

**VS - if I extended system if the user came 50000 next user reduce but over all user high but daily user less so we want only 2gm ram and we can’t reduce system ram and storage. We pay more cost eg: 1 lakhs user**

**We can’t predict the user**

**HS – we can buy same ram with ssd 2 same if the user reduce we can remove the system it reduce AWS Cost**

**Company prefer both based on requirement small company HS   
  
VS – same server for 10000 user same database**

**HS Muliple server Communication with help of Load Balancer**

**Load Balancer will handle muliple request and split to server   
  
it have separate cost we user both but must have for HS**

**We don’t need for VS single server  
  
HS – data inconsistency occur VS – data consistency   
  
we have handle file releated HS sent request to free server and next request will sent another server  
  
 but in VS Same server it will not affect   
  
initially HS and VS later**

**HS for public server and VS for private user**

**Vs 92 gb maximum but hs we can use with split**

**Load Balancing**

**Eg : Traffic police**

**Handle for too many request**

**Request 10000 server and response not handle without Load Balancer it will fail**

**Request -> Load Balancer -> server**

**Server have multiple instance so load balance will decide which server is free and sent to then**

**For these they have algorithms**

1. **Round Rober -> line by line they sent the request to server**
2. **A least request server -> check which server have lowest request have to sent**
3. **Least no of connections -> check request which is process quickly**

**Eg : upload a video it will take time like that it will check which fast**

**Profile view simple so fast, like edit, upload**

**They have consistent hashing**

**Request id dived by server and sent to them**

**Time and space complex it will check 50000 have 5 server it will take time so it add another instance**

1. **Reduce overtime**
2. **2.scalable**
3. **Redendence**
4. **Flexiabilty**

**Api gate way vs load balancer**

**Api gate way -> load balancer**

**Message queue :**