Step 1 - Scope of problem (5 minutes) :05pm

* specific feature(s)/product/service are we building? Why?
* Users
  * **How many?**
  * How do they interface with the system? Web? API? Native?
  * How frequently do they access the system?
  * Is there an expectation of growth? How often (weeks, months)?
  * Peak usage hours?
  * Are there super users? Or celebrity users? Or tiers of users?
* Any special requirements?
* technology stack?
* Can we leverage any specific infrastructure? E.g., CDN?
* Are there any constraints/key tradeoffs?: Technology/Servers? Budget? Restrictions?

Step 2 - Assumptions (5 minutes) :10pm

* Gather maximums
* Caching/data freshness requirements?
* Any “deal breakers”?
* What’s the optimal access and organisation of data?
* Availability/reliability - up time?

Step 3 - Draw Components (10 minutes) :20pm

* Draw major components

* Do back of envelope calculations

* Check reliability

* Identify future things - (out of scope) - like AI stuff

* **Get agreement before continuing**

Step 4 - identify key issues (5 mins) :25pm

* **Bottlenecks**:
  * Bandwidth, throughput, latency
  * Read/write/Synchronise operations
  * Tradeoffs?
* **single points of failure** - Quality of service? Reliability/unreliability of clocks?
* Rate limiting?
* security issues?
* Analytics? Privacy?
* **Agree: did we miss anything critical?**

Step 5 - Redesign for key issues (15 mins) :40pm

Step 6 - Wrap up (5 mins) - :45pm

TOOLS

* Workers
* Message queues
* Database - relational or NoSQL/GraphDB
* CDN
* Other external services

* Horizontal (more servers!)/Vertical scaling (more CPU/memory)
* Load balancer
* Caches
* Servers/shards
* CAP: consistency <=vs=> availability <=vs=> partition tolerance