1. Step1: Ask **functional requirements**.
    a. Who/what are the users?
    b. What is the purpose?
    c. What is the key features list down it?
    d. What are your assumptions?
    e. What are the constraints of the system?

2. Step2: ask **non-functional requirements**
    a. scalability, availability, performance, etc.

3. Step3: **Design high-level**
    a. Possible use cases
    b. Possible architecture components UI Backend, mobile app, website
    c. Possible services, read/write, Gateways, databases,
    d. Possible classes/data structures
    e. Redis cache, database, hash table, API server, load balancer, etc
    f. Network protocols and proxies
    g. Latency, throughput, and availability, performance, Load balancing
    h. Caching, Sharding, Polling, SSE, and WebSockets

4. Step4: **Drill down** on your design
    a. Should an API gateway be used to improve security?
    b. Will there be an authentication process to access the API?
    c. Will you use a REST API or something different?
    d. How would you modify the frequency of the system without compromising availability or increasing latency?
    e. Consider the fault tolerance of the system - How would you store the already built trie data structure so that in case of failure the system can be restored?
    f. Consider trade-offs, URL-encoding. Concurrency may cause problems, etc.
    g. Where will you place load balancers, and how will you cache URLs?

5. Step5: Bring it all together
    a. Sense checks your design, and confirm whether it has met all of the requirements you identified at the beginning of the interview