System Design

- Vedant Dhiman

Introduction

- COE NSUT, Delhi 2019
- Sodtware Developer at Samsung
- Call and messaging part
- Linkedin- https://www.linkedin.com/in/v7dhiman/
- Phone - 9958744385

Course Goals

- Trade-off Evaluations: Reliability vs Scalability vs Maintainability
- Components of Distributed Systems
  - Design Strategies
  - Why they're important
  - When to use them
- System Design Frameworks
- Avoiding Common Pitfalls
- Review fundamental concepts and must-know concepts
- Learn common design strategies and when to use them
- Practice – As much as possible

The System Design Interview

Are system design interviews technical interviews ?

Requirement Analysis

Design a car

The System Design Interview

Vague Questions

- Ask clarifying questions
- Follow a framework
- Time-bound approach
- Stay on track

No Right Answer

- No good or bad decision
- Justify choice of components
- Articulate trade-offs

Two-way Dialogue

- Very critical
- Work with your interviewer
- Clarify requirements in beginning
- Check-in throughout
- Justify your choices
- Summarize your design
- Go into specifics as per interviewer questions

Types of jobs to target from the market

- Engineering Manager
- Technical Product Manager
- Technical Lead
- SDE 2/3/4

Expectations from Interviewee

- Time management (Blueprint)
- Do’s
- Donts
- Questions for interviewer

Blueprint for System Design Interview

1) Requirement gathering (~5 mins)

2) Estimations (~5 mins)

3) High-level design (~10 mins)

4) Deep dive/ looking into each component (~15 mins)

5) Core of the problem (~7 mins)

6) Scalability (2-3 mins)

7) Partition or Fault Tolerance (2-3 mins)

8) Security (Optional)

9) Logging (Optional)

Interview Do’s

- Ask clarifying questions
- Understand the User
- Follow a framework
- List functional and non-functional requirements
- Depth vs Breadth Design
- Don’t need to know everything about everything
- Two-way dialogue with the Interviewer is a must
- Take lead/drive the conversation
- Check-in regularly
- Identify bottlenecks, offer alternatives, and summarize the pros and cons

Interview Don’ts

- Don’t start solving the problem immediately
- Didn’t ask clarifying questions
- Design didn’t meet specs
- Failed to grasp the problem statement
- Failed to prioritize requirements
- Missed critical components of the architecture
- Vague design flow, didn’t explain the components
- Didn’t evaluate tradeoffs
- Couldn’t justify decisions

Questions For Interviewer

- When the interview asks do you want to ask me anything?
- Don’t ever say you don’t have any questions for them
- Don’t ask questions like feedback on your design or improvements
- Ask more about the company/role under consideration
- Try to get a sense of the team and the work culture
- Try to understand the challenges you’d be facing if you’d get selected

System Design Process

What to expect in an interview

System Design Framework

How to approach System Design Question ?

System Design Interview Format

- 45-60 minutes long. Few companies have 90 minutes round as well.
- A vague question prompt
- Companies ask questions they are actually facing. For example:
  - Google may focus on search-related questions
  - Facebook may focus on large file storage systems
  - Amazon may focus on concurrency systems
- Use design tools such as whiteboard, draw.io, Google Docs etc.

Step 1: Define your problem space

- Capture Functional and Non-Functional Requirements
- What features do we need to build?
- At what scale will this system need to work?
  - How many requests per second?
  - How many users?
- Under what conditions should this system work?
  - Bad network connectivity? / database consistency? / error handling ?
- Other non-functional requirements
  - Speed
  - Reliability
  - Maintainability
  - Cost

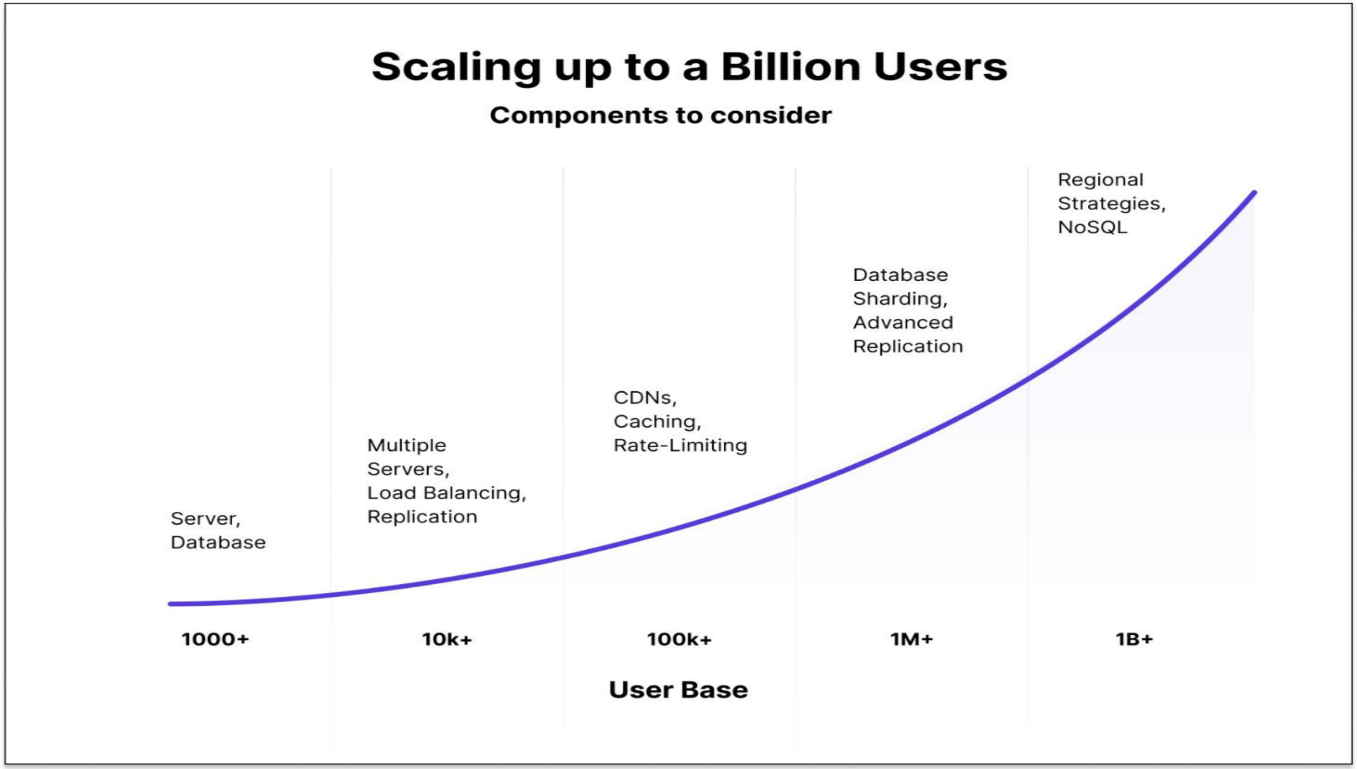
Step 2: High level design

- Identify critical requirements – explain the reasoning on why they are critical
- High level decisions to think about:
  - If it's an application, what coding framework will you use? (MVC, MVVM)
  - If it involves a database, what kind of database will you use? Does it need to be sharded?
  - If it involves a server and client, what kinds of APIs do you need to build?
  - If it requires lots of computation, do you need to parallelize the workload?
  - Will your architecture be monolithic (i.e. one service) or split into microservices?

Step 3: Deep-dive

- Deep-dive into the components of the system
- Define the interface between these systems:
  - APIs between server and client
  - Database tables and indexes
  - Background jobs (if any)
- Check with the interviewer if they want you to focus on one area or another

Step 4: Identify Bottlenecks and Scale



Step 5: Summarize

- Walk through your major decisions
- Justify each trade-off in terms of space, time and complexity
- Make sure user experience, business goals, requirements and system are aligned
- Think about futuristic product requirements and how the design will be updated accordingly