Technical Concepts Pt. 2

Agenda

1. System Design & Scalability
2. Technical Concepts
3. Object-Oriented Approach
4. Break-Out Rooms

1.

System Design & Scalability

How to Approach

What is System Design?

- Process of designing elements of a system
  - Architecture
  - Modules
  - Components
  - Interfaces
  - Data

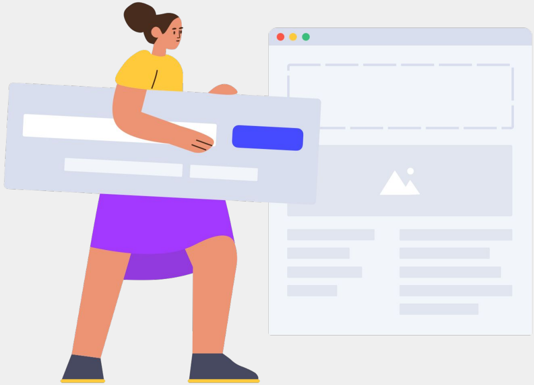
> *Actual experience with a wide range of tools and systems is an advantage, but being able to identify a need and suggest a common solution for it would get you a long way, even if you've never used it yourself.*
>
> *- Pramp*

System Design Question

- Similar to technical system design questions
- Brainstorming session
  - Open-ended questions
- Don’t have to know everything!
  - Thought process behind design choices is more important
- There is no optimal solution
  - Everything is a tradeoff

Step-by-Step

1. **Go to the whiteboard!**
   a. Write/draw all your thoughts
2. **Ask clarifying questions**
   a. Understand the goal
   b. Narrow the scope
   c. Share your assumptions
3. **List out features you will be designing for**
   a. Lay out which areas to focus on
   b. Use cases



Step-by-Step

4. **Draw out high-level approach**
    a. End-to-end user flow based on established goals
5. **Drill down into which data structures, algorithms, software solutions**
    a. Don’t spend too much time on the details
6. **Discuss tradeoffs and edge cases**
    a. Pros and cons of different approaches

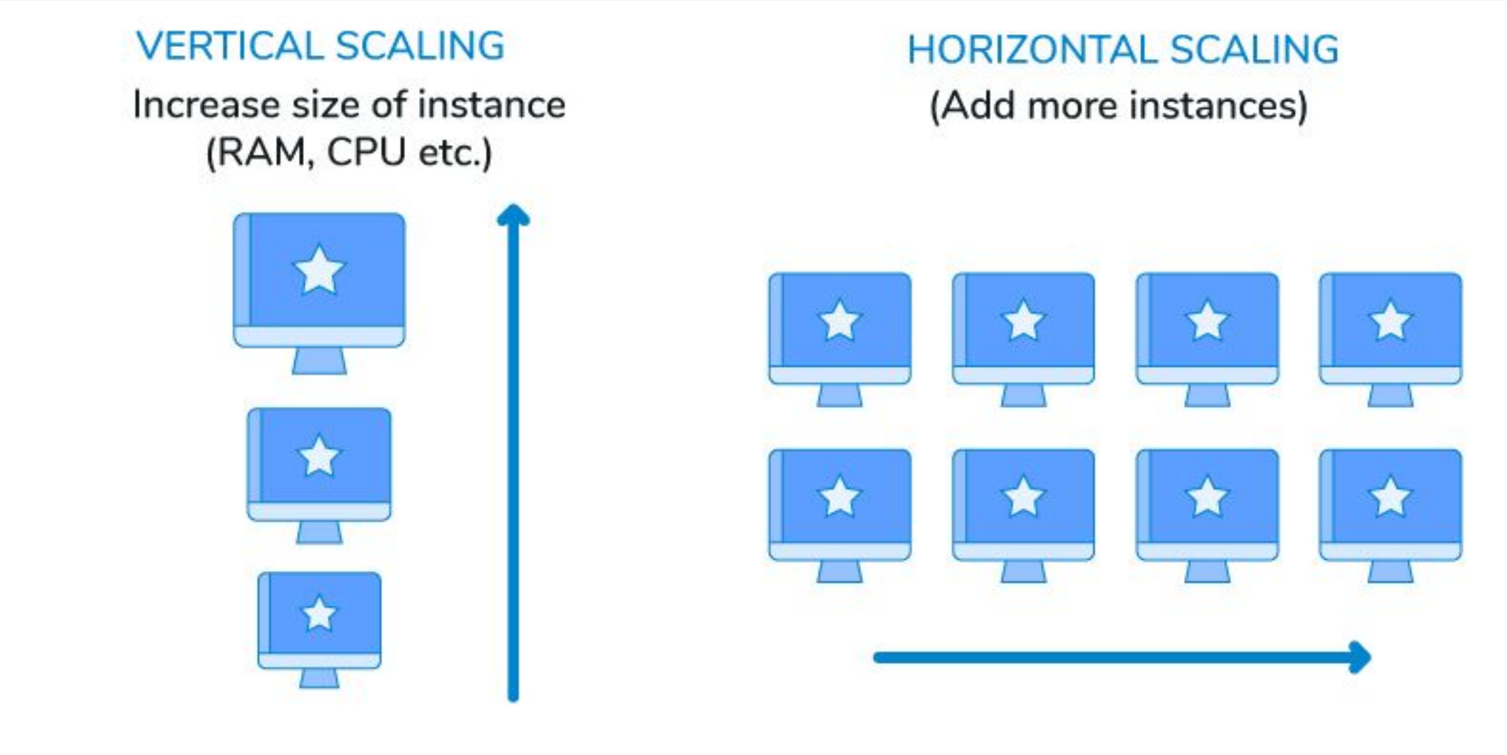
2.

Concepts

Be familiar with a broad range

> *And when it comes to distributed systems, it turns out, size really* does *matter.*
>
> *- Vaidehi Joshi, Medium*

Scalability

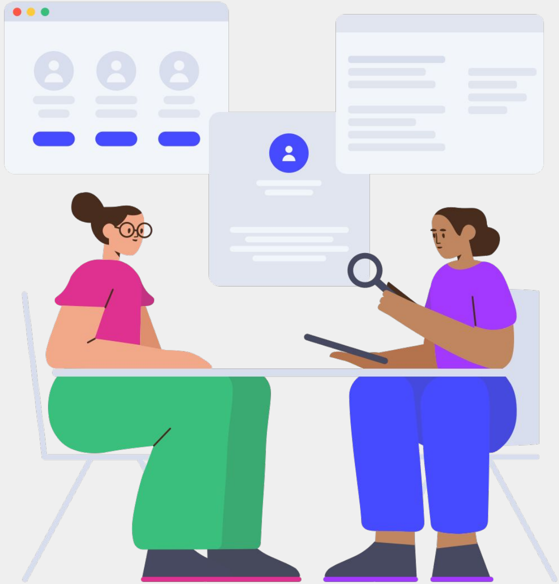


Pros & Cons

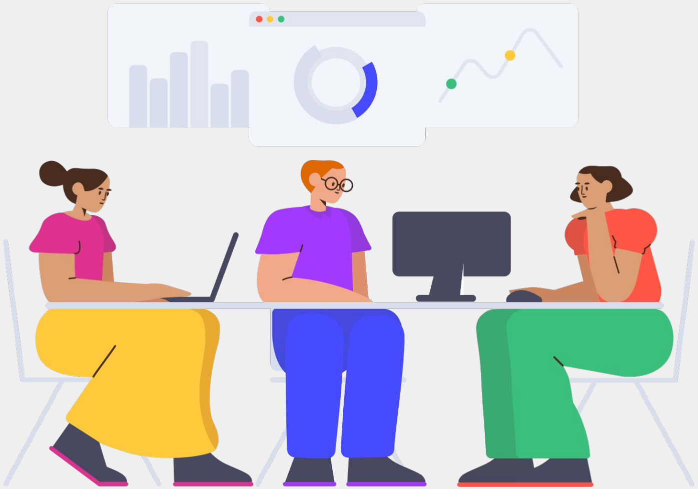
Horizontal Scaling	Vertical Scaling
Load Balancing Required	N/A
Resilient	Single point of failure
Network Calls	Inter-process communication
Data inconsistency	Consistent
Scales well as users increase	Hardware limitations

Main Takeaways

- Communicate clearly with your interviewer, but you’re the driver
- Explore different directions
- Ask the right questions
- Examine the tradeoffs
- Don’t drill down on details
- Be explicit about your assumptions



Technical Limitations



- Memory (RAM)
  - Data storage costs
- Execution time
  - Complexity of implementation

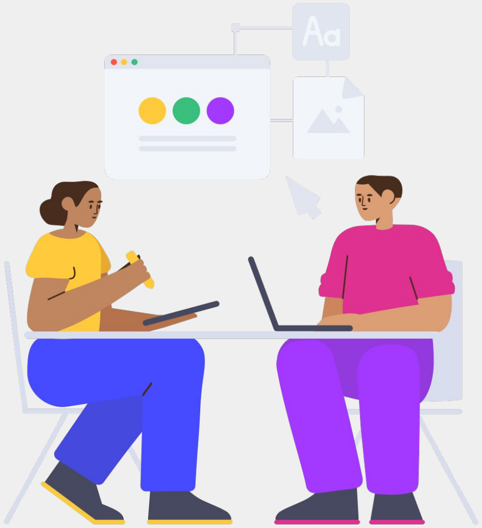
3.

Object-Oriented Approach

Java Time!

How does this relate to PM?

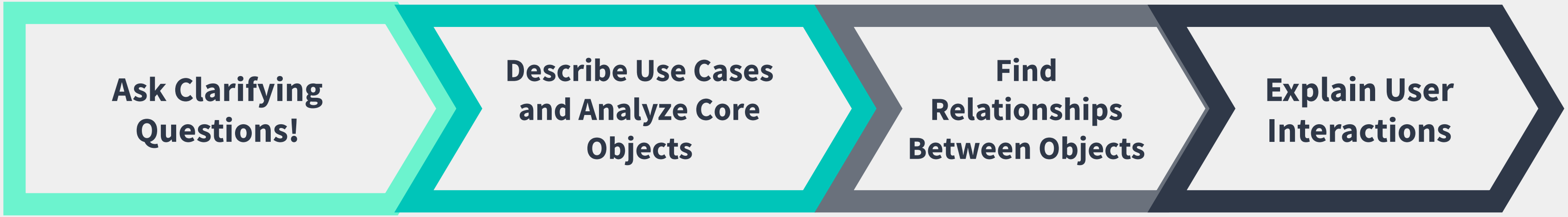
- Object-Oriented Design Questions
  - **Vision** for big picture
  - Show off your **technical** expertise
  - Ability to understand **end-user**
- Very similar to product design questions



Jukebox Example

- **System Components**
  - **Jukebox**
    - Plays songs and playlists
  - **CD**
    - Contains songs and artists
  - **Playlist**
    - Can add, delete, or queue songs in playlist
  - **Song**
    - Contains song name and artist name
  - **Artist**
    - Component of song

How to Approach



Ask Clarifying Questions!

Describe Use Cases and Analyze Core Objects

Find Relationships Between Objects

Explain User Interactions

Thanks!

Any questions?