



TOP



Things to consider during

# SYSTEM DESIGN INTERVIEW



REVANTH MURIGIPUDI

## \*Disclaimer\*

- System design is purely discussion-driven, there is no one fixed architecture for a system, & it varies based on requirements as well.
- From this pdf, you can expect "getting started" topics that one should consider before designing a scalable system.

#1

# SET THE EXPECTATIONS RIGHT

- **Functional requirements** - Talk about the features or use cases your system will handle, & always take inputs from the interviewer
- **Nonfunctional requirements** - Discuss things like "how much storage is required, latency, availability, performance, scalability", and so on...

EXPECTATIONS

**#2**

## **DO CAPACITY ESTIMATIONS**

Capacity estimations are done to understand the scale of the system at hand

Generally, you can talk about storage, **network bandwidth** and these estimations make you realise about the wrong assumptions in non functional requirements, if you have made any

ESTIMATIONS

## EXAMPLE

Consider that you're designing a messaging application like **Whatsapp**. Here's one probable way you can drive the discussion

Let's assume there are **100 million** daily active users (DAU), and every user sends at least 50 text and 10 media daily messages which means around **6 billion** messages every day !!



Continue on next slide...

## CONTINUE...

If we consider **50 bytes** for a text message and **500 bytes** for a media message, then we need **0.75 terabytes** of storage every day.

And for data warehousing, if we store the messages for 10 years, then we need **2.7 petabytes** of storage

There's also other storage that we need to consider like metadata, userdata, groupdata etc etc. Likewise you can easily drive the discussion for network bandwidth as well

E X A M P L E

**#3**

## **DATABASE SCHEMA**

Start considering the trade-offs between a relational and a non-relational database. Look for the functional & nonfunctional requirements again and understand which one suits the best!

### **EXAMPLE**

Let's say you're designing some sort of transaction system, then you need **ACID properties**, which hints you towards a relational database!

DATABASE

Sounds like a **simple example?**  
but you get the gist, right?

Also you can talk about the tables, attributes, primary & foreign keys, explain relationships between the tables, and avoid having data redundancy across multiple tables.



E X A M P L E

#4

## ARCHITECTURE DESIGN

Start with the **simplest component** and **gradually build** your way to the high-level design keeping the requirements in mind

You might also have to talk about **synchronization** and concurrency if the discussion requires it, so be ready with that as well

# EXAMPLE

Start off with a simple client and a load balancer, then add the services required & explain what they'll do. Add in caching if required, message queues for async/batch processing, establish connections with database. Also talk about replication & availability

## PS :

Remember that these architecture designs are discussion driven! There is no right or wrong design, so, discuss with the interviewer + take insights from them

E X A M P L E

**#5**

# API CONTRACT DESIGN

Talk about the essential APIs needed for your system. Remember a few points when writing them down :

- Specify the **URL** path, query & path parameters necessary
- Mention the method and its signature handling the **API endpoint**
- Specify the **HTTP** Method, request & response body

# #6

## DEEP DIVE INTO FEW COMPONENTS

You are given **limited time** during the interview, so it's not possible to drill down into every single component.

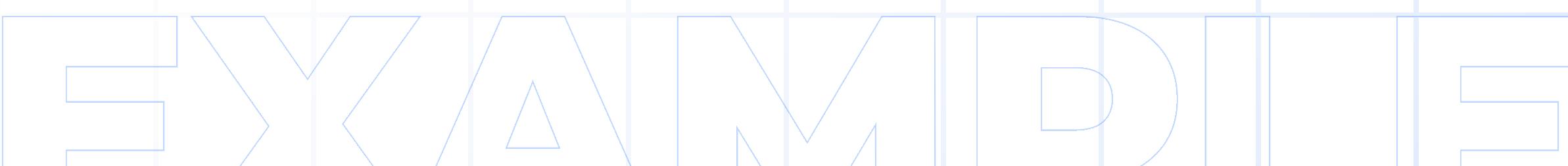
Clarify which component the **interviewer** wants to discuss more about and dig deep into designing it

## EXAMPLE

Imagine **designing** a notification system during the interview, & you specified about using message queues for sending async updates .

**Deep diving**, you can talk about how notifications are pushed onto queue, how the consumers are configured, how the final notifications are validated by a rule engine before propagating it to downstream user client devices

**PS :** Don't forget to talk about the failure conditions across components, how you can ensure they're reliable, failure tolerant & maintain availability by scaling





Recruiters in seek engineers who can design scalable systems

**System design** is one of the important ingredients along with **DSA, CS Fundamentals**



**BOSSCODER ACADEMY**  
**HAS GOT YOU COVERED ENTIRELY**

Within a span of 7 months, you will **develop skills + confidence + hands-on experience** to grab your dream job



## WHY BOSSCODER?

 **400+** Alumni placed at Top Product-based companies.

 More than **136% hike** for every **2 out of 3** working professional.

 Average package of **22LPA**.

The syllabus is most up-to-date and the list of problems provided covers all important topics.

Lavanya  




Course is very well structured and streamlined to crack any MAANG company

Rahul .  




[EXPLORE MORE](#)