1️⃣ Flash Sale / High Traffic (Simple Version)

Ask this:

If thousands of orders come at the same time during a sale, how will you make sure **no order is missed**, **no order is created twice**, and the website does not become slow?

Good answer sounds like:

- Orders are saved first, quickly
- Heavy work happens later in background
- System can handle repeated clicks safely
- Even if traffic is high, orders are not lost

Bad answer sounds like:

- "Node.js is fast, so it will be fine"
- "We'll handle it directly in one API"

2️⃣ Courier System Down (Very Important)

Ask this:

If one courier's system is slow or not working, what happens to our orders?
Will customers get stuck, or will the system handle it automatically?

Good answer sounds like:

- System waits only a short time
- Automatically tries another courier
- Orders don't stop because of one failure

- Problem courier is skipped temporarily

Bad answer sounds like:

- “We’ll just try again”
- No clear backup plan

3️⃣ Choosing the Best Courier (Business-Focused)

Ask this:

> How will the system decide which courier is best — not just cheapest, but also reliable and fast?

Good answer sounds like:

- Looks at past delivery performance
- Considers cost, speed, and success rate
- Improves decisions over time
- Automatically switches if one courier performs badly

Bad answer sounds like:

- “We’ll always choose the cheapest courier”
- No mention of learning from past data

4️⃣ Order Status Updates Everywhere

Ask this:

> When an order status changes (like shipped or delivered), how will tracking, billing, and notifications all update correctly without breaking each other?

Good answer sounds like:

- One change updates everything automatically
- Systems are connected but not dependent
- Even if one part fails, others still work
- Full history of order changes is saved

Bad answer sounds like:

- “We’ll update everything in one place”
- Too much dependency between systems

5️⃣ Reports vs Live Orders

Ask this:

> If many people are checking reports and dashboards, how do we make sure new orders are still created fast?

Good answer sounds like:

- Reports use separate data
- Live orders are always priority
- Dashboards are cached or updated in background
- No slowdown for real users

Bad answer sounds like:

- “Database can handle everything”
- No separation between reports and live data

6️⃣ Night-Time Failures (Trust Question)

Ask this:

> If something breaks at night and orders start failing, how will we know immediately and find the problem fast?

Good answer sounds like:

- Automatic alerts
- Clear error tracking
- Ability to see where and why it failed
- Not dependent on manual checking

Bad answer sounds like:

- “We’ll check logs in the morning”
- No alert or monitoring system