When to Choose

🧠 **“When to Choose What?” — System Design Decisions**

1. Monolith vs Microservices

🧱 Monolith — কখন নেবে?

✅ Small team
✅ Early-stage product
✅ Fast development দরকার
✅ Low traffic

❌ কখন না:

- Team বড়
- Independent scaling দরকার

📌 Line:

“We start with monolith for simplicity and move to microservices as scale grows.”

🧩 Microservices — কখন নেবে?

✅ Large team
✅ High traffic
✅ Independent scaling দরকার
✅ Different tech stacks

❌ Cons:

- Complex

- Debugging hard

📌 Line:

“Microservices add complexity but give scalability and team autonomy.”

2. SQL vs NoSQL

🗄️ SQL — কখন?

✅ Strong consistency দরকার
✅ Complex joins
✅ Financial / transactional data

Examples:

- Banking
- Payments

📦 NoSQL — কখন?

✅ Huge scale
✅ Flexible schema
✅ Read/write heavy system

Examples:

- Social media
- Analytics

📌 Line:

“We choose NoSQL for scalability, accepting eventual consistency.”

3. Cache vs Database

⚡ Cache — কখন?

✅ Read-heavy system
✅ Low latency critical
✅ Repeated data access

❌ Risk:

- Stale data

🗄️ Database — কখন?

✅ Source of truth
✅ Data durability দরকার

📌 Line:

“Cache improves performance but database remains the source of truth.”

4. Sync vs Async Communication

🔁 Sync — কখন?

✅ Immediate response দরকার
✅ Simple flow

Examples:

- Login
- Payment confirmation

📨 Async — কখন?

✅ Background processing
✅ High latency tolerate করা যায়

Examples:

- Email
- Notifications

5. REST vs GraphQL

🌐 REST — কখন?

✅ Simple CRUD APIs
✅ Cache-friendly

🔍 GraphQL — কখন?

✅ Client needs flexibility
✅ Over-fetching issue

6. Strong vs Eventual Consistency

🔒 Strong — কখন?

✅ Correctness critical
✅ Money-related

⏳ Eventual — কখন?

✅ High availability দরকার
✅ Social features

🏆 FAANG Cheat Rule 🧠

Choose simplest thing that works for current scale, design for future scale.

🔚 One-line Summary

“When to choose what” = context + constraints + trade-offs