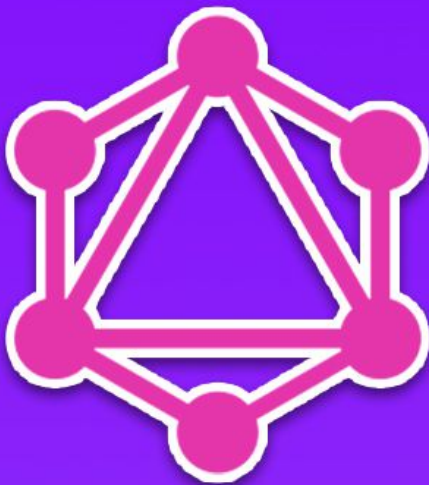




Hamza Nadeem
@hamza-nadeem-09

GraphQL VS REST

Real-Life Decisions in Production



Let's swipe to see

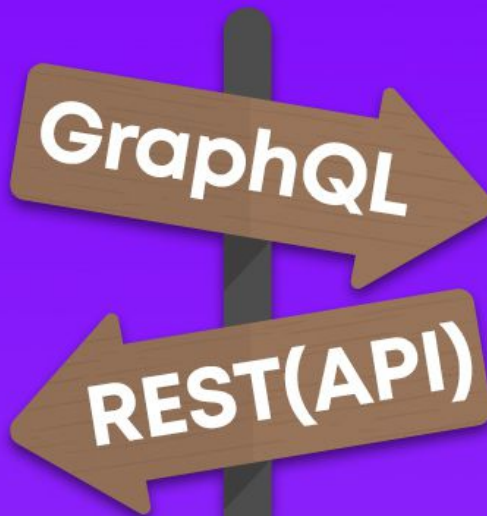




Hamza Nadeem
@hamza-nadeem-09

The API Dilemma

Choosing between GraphQL and REST isn't just about tech preference — it's about scalability, team dynamics, performance, and use case.





Hamza Nadeem
@hamza-nadeem-09

Why We Chose REST

- Stable endpoints
- Simpler caching via HTTP
- Easier to onboard new devs
- Works great for CRUD-heavy apps



Use Case:

Traditional e-commerce backend with standard data flow.

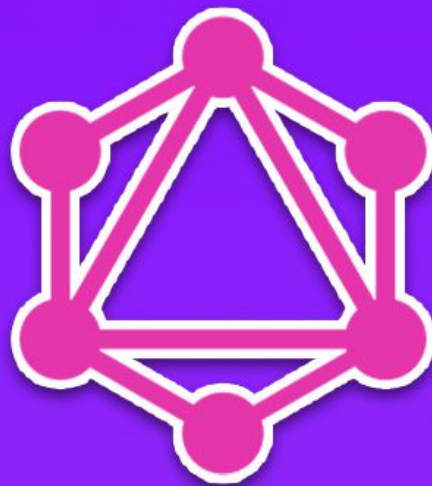




Hamza Nadeem
@hamza-nadeem-09

Why We Chose GraphQL

- Complex querying made easy
- Reduced over-fetching
- Strongly typed schema
- Great for mobile clients



Use Case:

A dashboard-heavy analytics platform with nested data needs.





Hamza Nadeem

@hamza-nadeem-09

Pros & Cons

	GraphQL	REST
Flexibility	High	Medium
Learning Curve	Steep for beginners	Lower
Query Control	Precise data fetching	Can over/under-fetch
Caching	Requires custom solutions	Native HTTP caching support



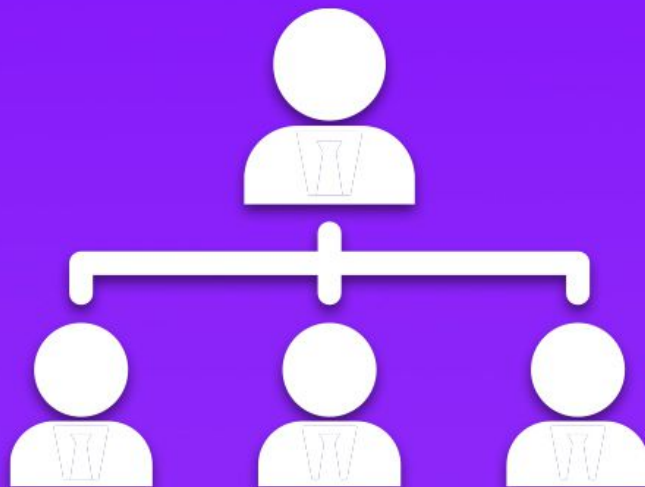


Hamza Nadeem
@hamza-nadeem-09

Team & Org Impact

Your choice impacts:

- Developer onboarding
- Tooling & ecosystem
- Testing complexity
- Long-term maintainability



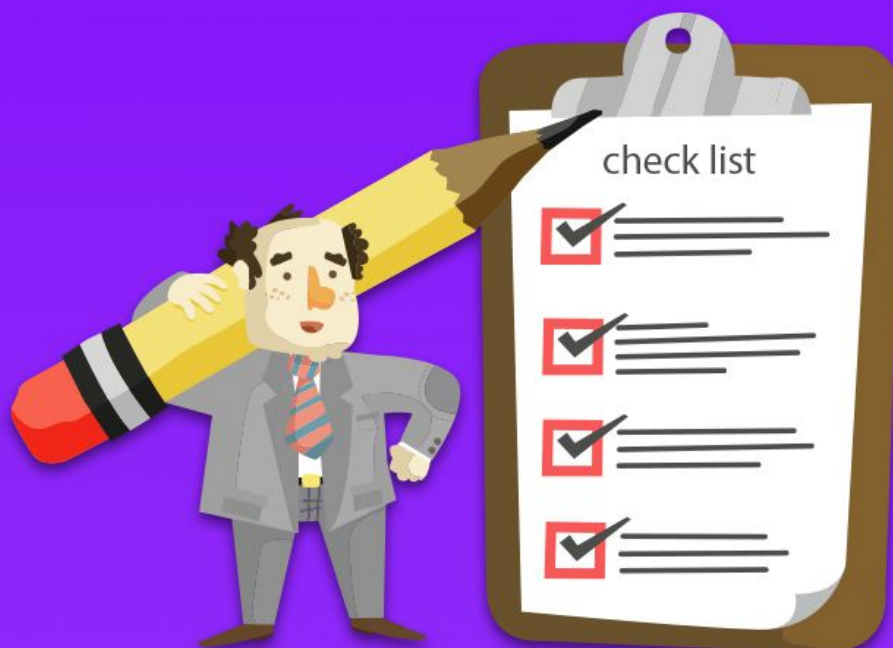


Hamza Nadeem

@hamza-nadeem-09

Lessons Learned

- Understand the data access patterns first
- Talk to frontend + backend teams
- Build internal tooling early (esp. with GraphQL)
- There's no "one-size-fits-all" , context is king





Hamza Nadeem

@hamza-nadeem-09

REST or GraphQL?

Use REST if:

- You need simplicity and stability
- Your app is CRUD-heavy

Use GraphQL if:

- You need dynamic querying
- You're serving multiple client types with varying data needs

Every API decision is a tradeoff. Choose what serves your product, team, and users best.



Have you made this choice before?

Share your experience in the comments!



Let's connect!



Hamza Nadeem
@hamza-nadeem-09