REST vs GraphQL: When to Use What

In modern web and mobile application development, APIs are essential for client-server communication. Two dominant API styles are REST and GraphQL. Understanding their differences, strengths, and appropriate use cases is critical for designing scalable, efficient systems.

🔗 What is GraphQL?

GraphQL is a query language for APIs and a runtime for executing those queries, developed by Facebook. Unlike REST, the client specifies exactly what data it needs.

Key Characteristics:

- **Schema-based and strongly typed**
- **Single endpoint** (e.g., `/graphql`)
- Client-defined queries (nested and selective)
- Supports **real-time subscriptions**
- Avoids over-fetching and under-fetching

🧠 When to Use REST

Use REST when:

- The API is **simple and CRUD-based**
- You want **easy caching** via HTTP
- Existing tools or clients are REST-optimized
- You need **low complexity** and **broad compatibility**
- You're exposing resources with well-defined URIs

⚖️ Summary

Use Case	Best Fit
Simple CRUD APIs	REST
Mobile clients needing efficiency	GraphQL
Public APIs needing HTTP caching	REST
APIs with complex data relationships	GraphQL
Long-lived clients with changing needs	GraphQL

🔚 Conclusion

REST and GraphQL serve different needs. REST shines in simplicity, standardization, and compatibility. GraphQL excels in flexibility and efficiency, especially for modern frontend-driven development. Choosing the right one depends on your application’s requirements, data complexity, and development velocity.