You can't survive ***BACKEND*** if you don't know these *concept*

1 REST vs GraphQL

➢ REST (Representational State Transfer)

- Uses fixed endpoints like /api/users, /api/products.
- Each endpoint returns all data, even if you need a small part.
- Simple, widely used - great for most CRUD apps.

➢ GraphQL

- Uses a single endpoint (usually /graphql).
- You can query exactly what you need - nothing extra.

2 JWT Authentication (JSON Web Token)

- Secure way to handle user authentication in APIs.
- After login, backend sends a signed token to the client.
- The token is included in every future request (Authorization: Bearer <token>).
- Backend verifies it - no need to check DB every time.
- Tokens have header + payload + signature (base64 encoded).

3 Database Relationships

Understanding relationships is core backend knowledge

➢ One-to-One

One record linked to exactly one other.

- Example: User ↔ UserProfile

➢ One-to-Many

One record linked to many others.

- Example: User ↔ Posts

➢ Many-to-Many

Multiple records linked both ways.

- Example: Students ↔ Courses

➢ Importance:

- Keeps your data organized & normalized.
- Reduces duplication.
- Improves query performance.

4 Caching (Redis)

- Stores frequently used data in memory instead of hitting the DB every time.
- Makes responses 10-100x faster.
- Example:

  1. Request comes → Check Redis cache.
  2. If found return instantly.
  3. If not fetch from DB, then save in cache.

- Used for: sessions, API responses, leaderboards, etc.

5 Message Queues (Kafka /RabbitMQ)

- Used for asynchronous communication between services.
- Instead of calling another service directly, you send a message to a queue.
- The other service picks it up later and processes it.
- Prevents systems from crashing under heavy load.
- Example:
  - User signs up →
    send "Welcome Email" message →
    Email service handles it later.

6 Load Balancing

- Distributes incoming traffic across multiple backend servers.
- Ensures no single server gets overloaded.
- Increases availability, scalability, and performance.

  Common tools: Nginx, HAProxy, AWS Elastic Load Balancer.