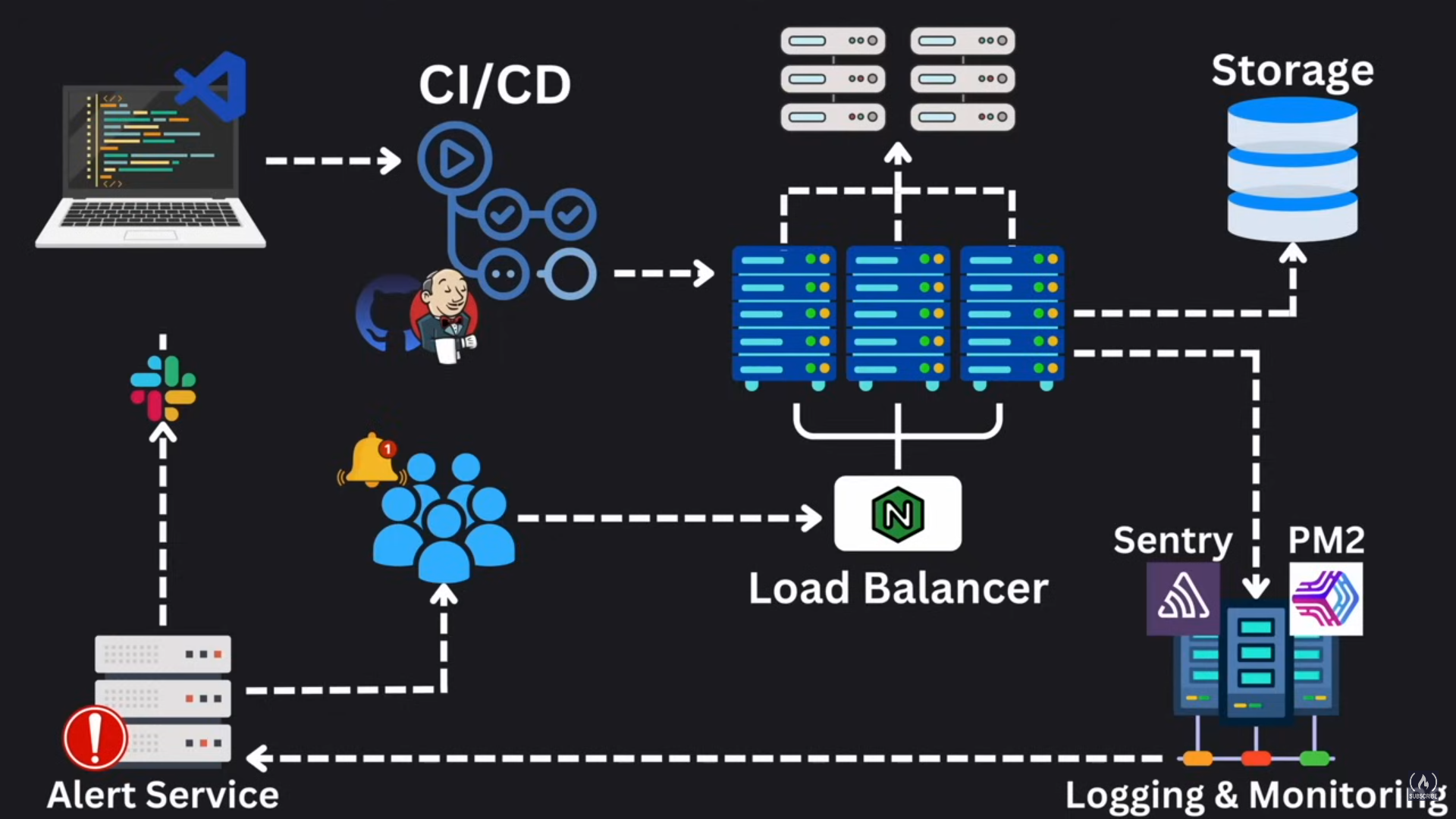
**System Design Concepts**

## **High-level architecture of a production app**

1. CI/CD pup-line (continuous integration and continuous deployment)
2. 

## **Good system design**

1. Scalability
2. Maintainability
3. Efficiency
4. Reliability

## **System design 3 key elements**

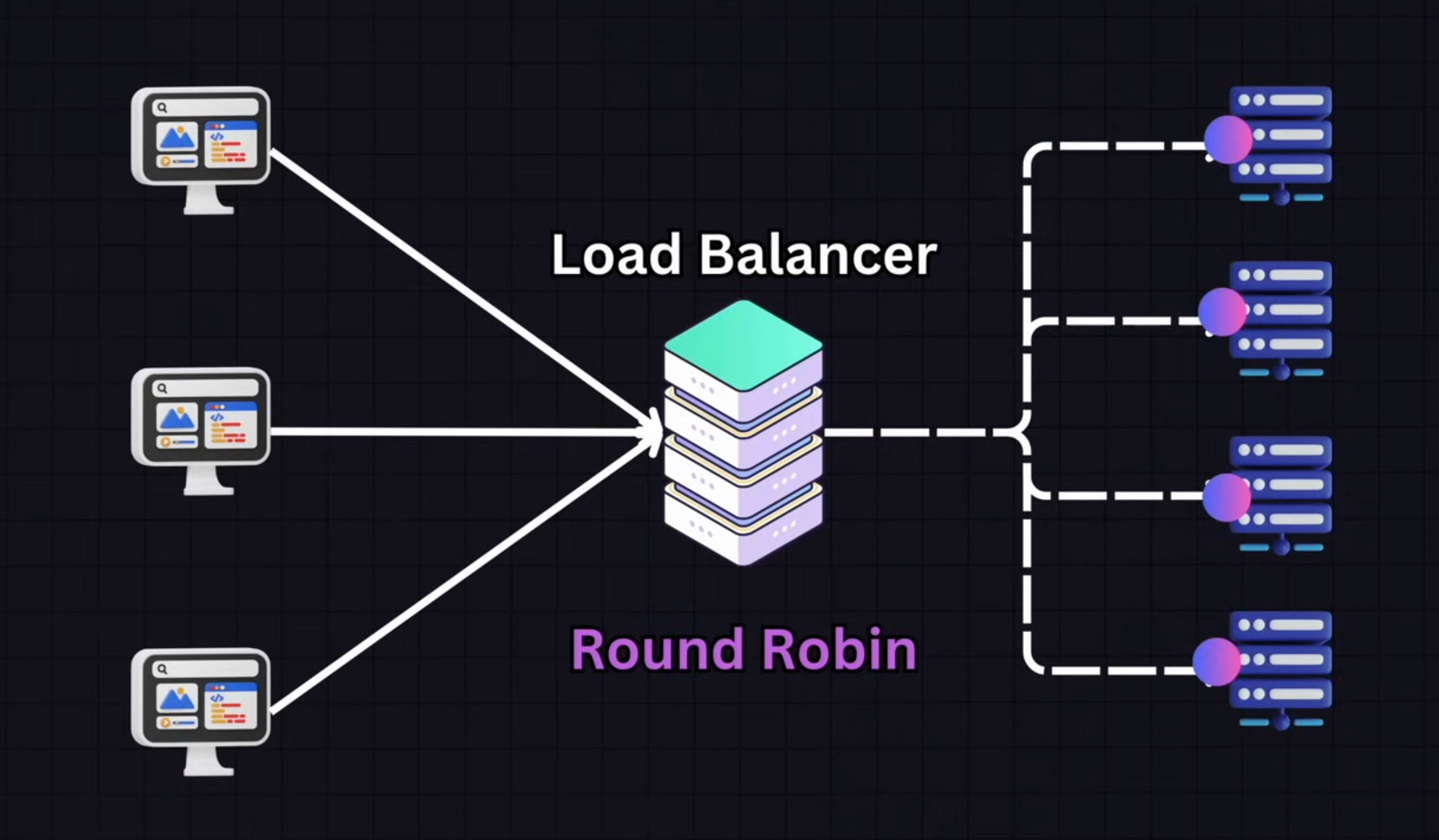
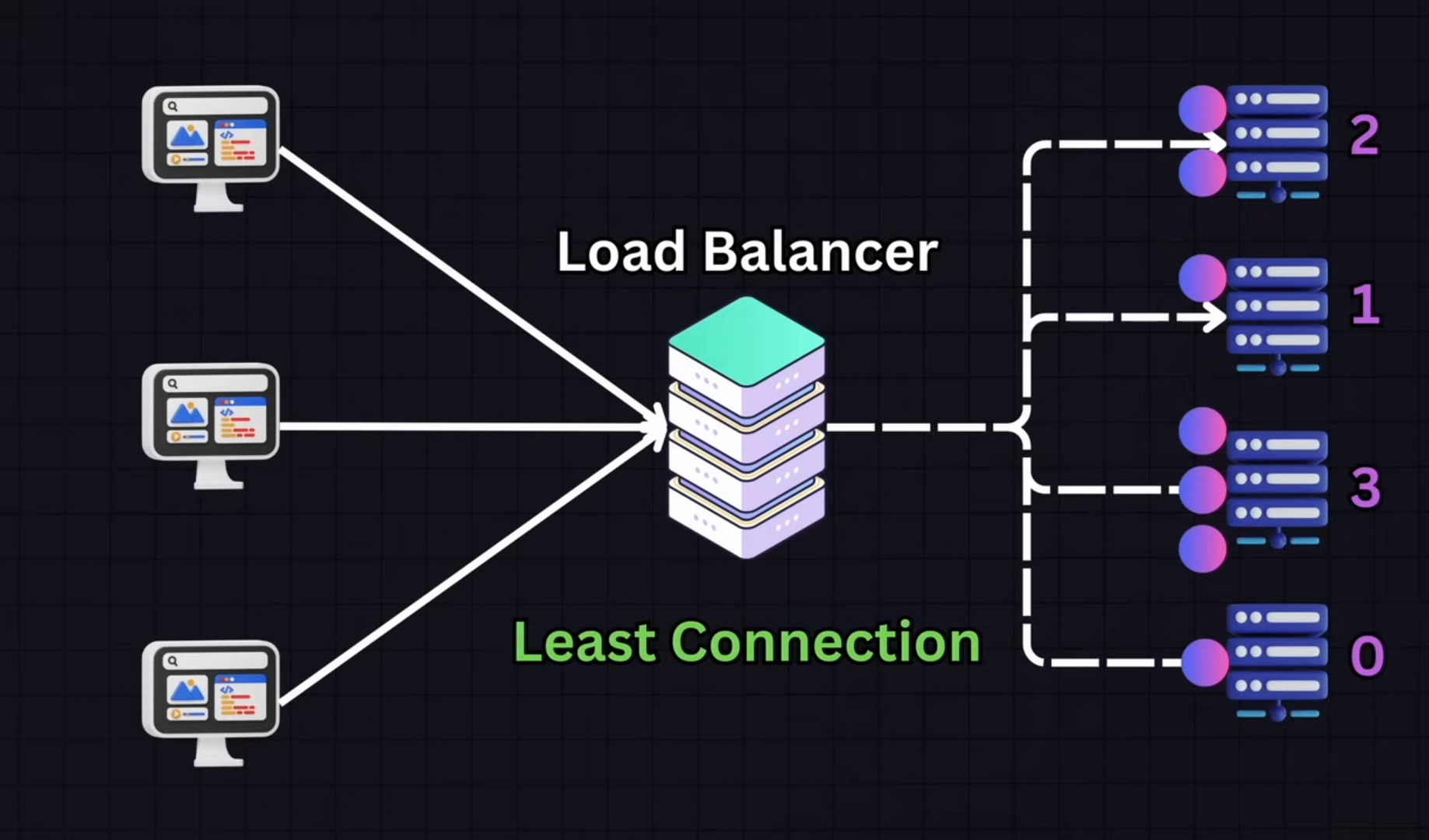
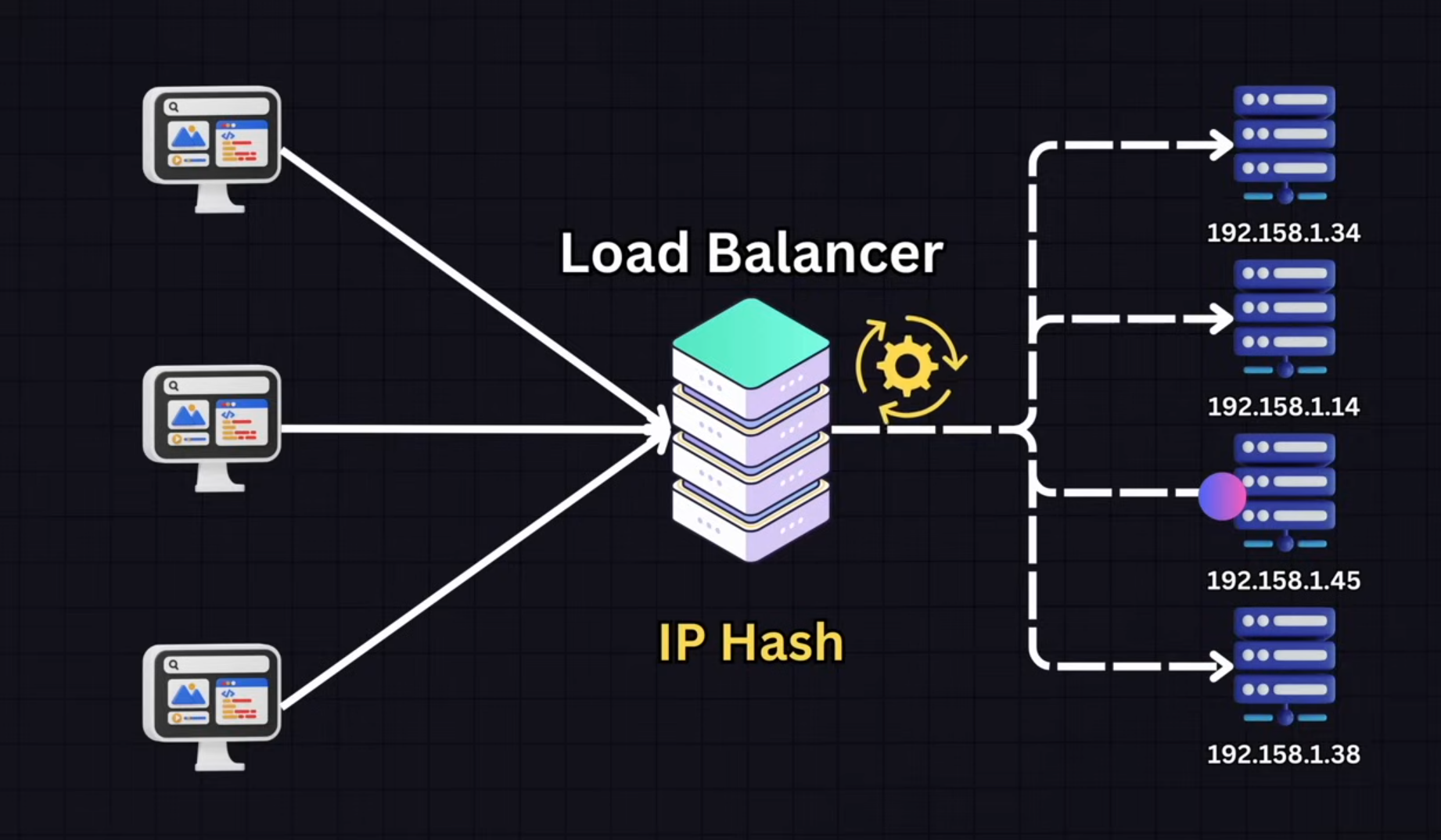
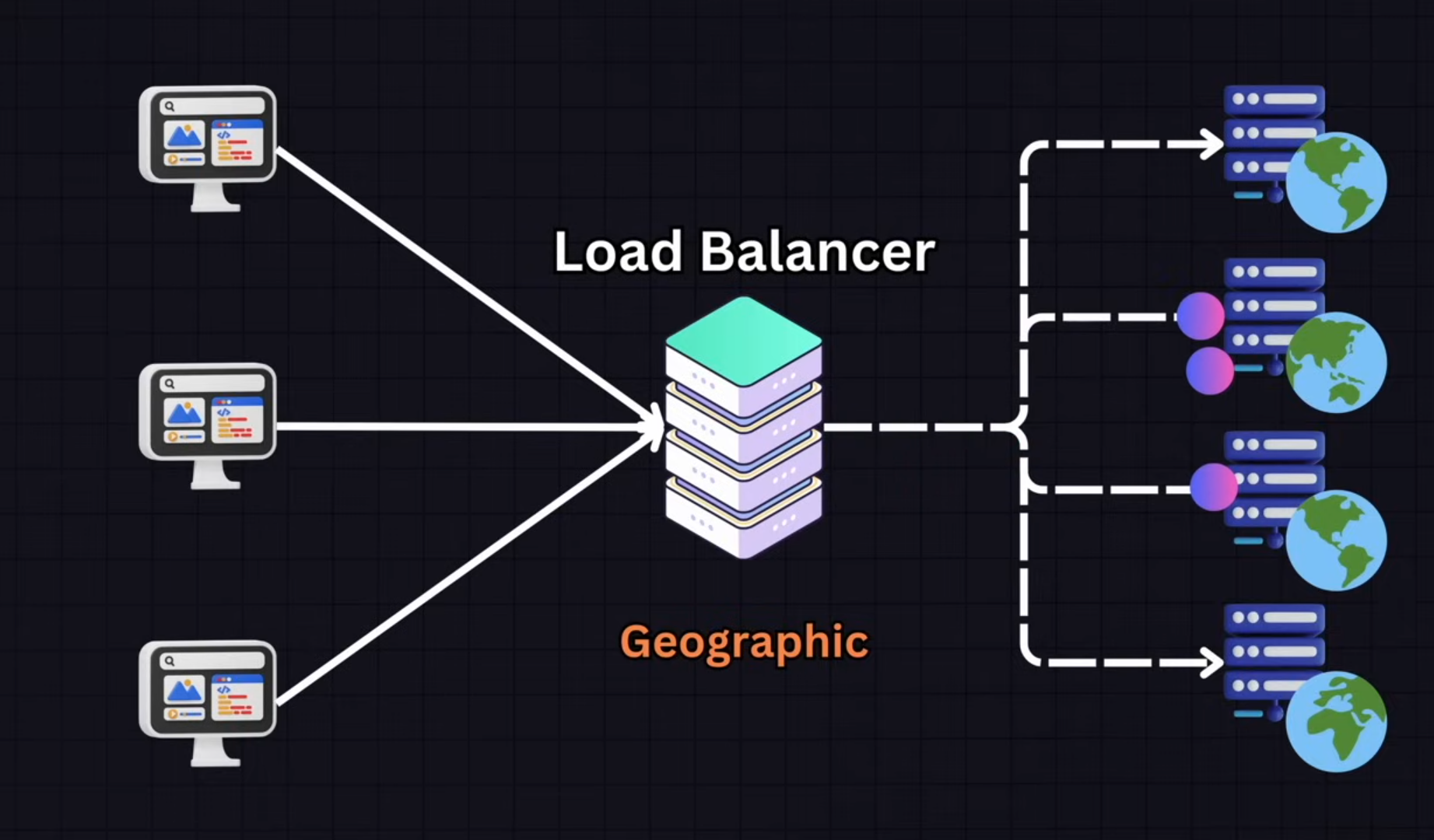
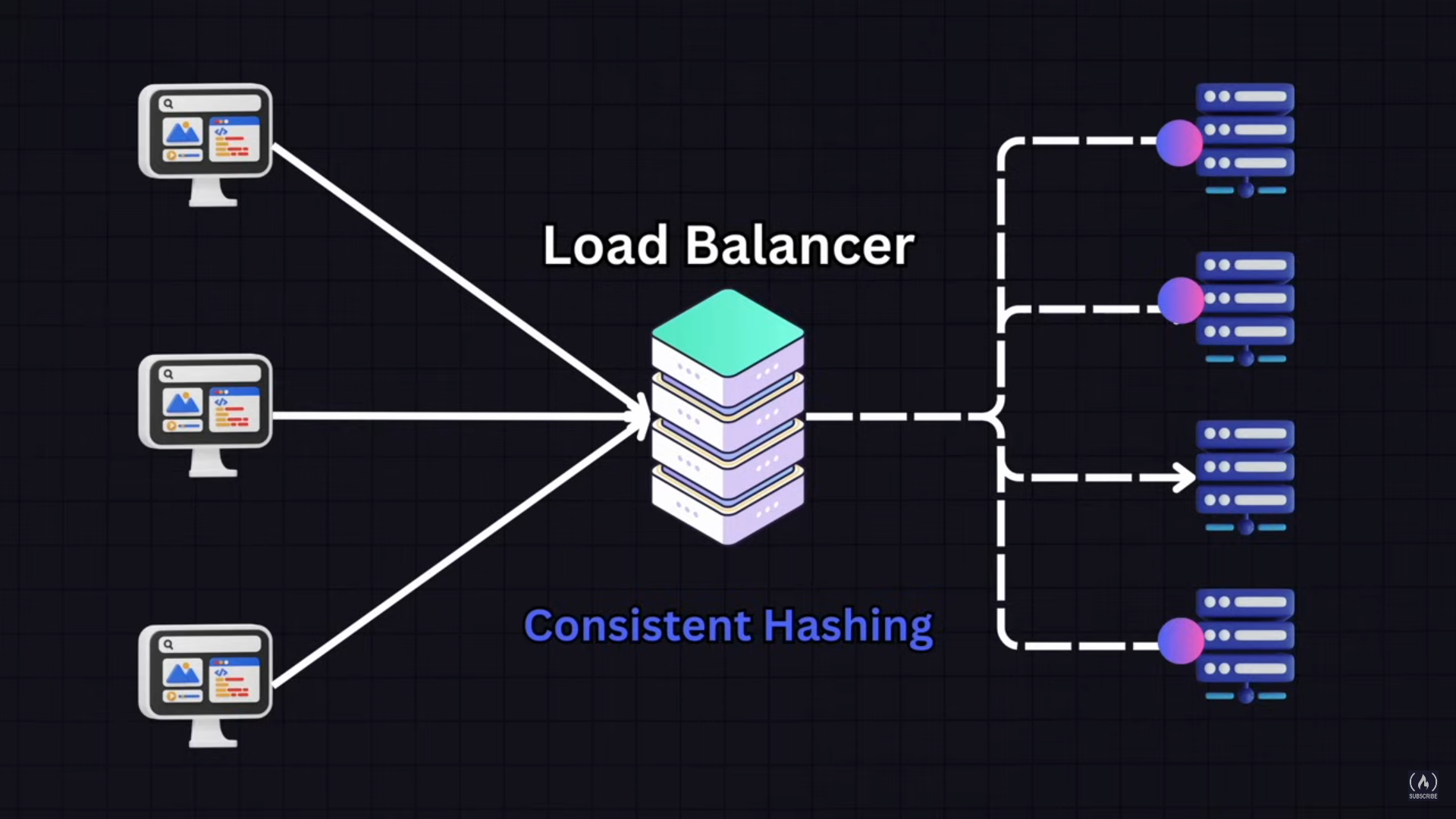
1. Moving data
   1. Json 🡪 server
   2. Server 🡪 json
2. Storing data
   1. Access patterns
   2. Indexing strategies
   3. Backup solutions
3. Transforming data
   1. Roll data from the meaningful information Json, log, SQL

## **CAP Theorem**

1. Consistency
   1. Same data same time
2. Availability
   1. Operational and response to request 24/7
   2. SLO – service level objectives
   3. SLA – service level Agreements
   4. Partition tolerance
   5. Continue functioning
   6. Reliability
   7. Fault tolerance
   8. Redundancy
   9. Speed
      1. Throughput: how much data our system can handle over a certain period of time
         1. Server throughput – unit RPS (request per second)
         2. DB throughput – unit QPS (query per second)
         3. Data throughput – B/s
      2. Latency: how long it takes to handle a single request

## **Brewer’s Theorem**

## **Networking basics**

1. IP – internet protocol
   1. IPv4 – 32bit
   2. IPv6 – 128bit
   3. IP header: bearing the IP information
2. Application layer
3. Transport layer
   1. TCP – transmission control protocol
      1. Reduce the possibility of the packet going missing.
      2. It will establish any connection.
   2. UDP – user datagram protocol
      1. It will not establish any connection.
      2. There is a higher possibility of the packet going missing.
   3. DNS – domain name system
      1. Yotube.com 🡪 192.168.5.1
4. Networking infrastructure
   1. Public IP address – 172.16.254.1
   2. Private IP address – 172.16.254.1
   3. Static IP address – 172.16.254.1
   4. Dynamic IP address – 172.16.254.1, 192.168.1.1
   5. Local area network (LAN) –
   6. PORTS
      1. HTTP – 80
         1. hypertext transfer protocol
         2. does not store any client context between requests
      2. HTTPS – 443
      3. SSH – 22
      4. MYSQL – 3306
   7. Status codes
      1. 1xx Informational
         1. **100 Continue:** The server has received the request headers, and the client should proceed to send the request body.
         2. **101 Switching Protocols:** The server is switching protocols as requested by the client.
      2. 2xx Success
         1. 200 OK: The request was successful, and the server is returning the requested data.
         2. 201 Created: The request was successful, and a new resource was created.
         3. 202 Accepted: The request has been accepted for processing, but the processing is not complete.
         4. 204 No Content: The request was successful, but there is no content to send in the response.
      3. 3xx Redirection
         1. 301 Moved Permanently: The requested resource has been moved permanently to a new URL.
         2. 302 Found: The requested resource has been temporarily moved to a different URL.
         3. 303 See Other: The response to the request can be found under a different URL using the GET method.
         4. 304 Not Modified: The resource has not been modified since the last request, so the client can use the cached version.
         5. 307 Temporary Redirect: The resource is temporarily located at a different URL, but future requests should still use the original URL.
         6. 308 Permanent Redirect: The resource has been permanently moved to a new URL, and future requests should use the new URL.
      4. 4xx Client Errors
         1. 400 Bad Request: The request was malformed or invalid.
         2. 401 Unauthorized: The request requires user authentication.
         3. 403 Forbidden: The server understood the request, but it refuses to authorize it.
         4. 404 Not Found: The requested resource could not be found.
         5. 405 Method Not Allowed: The request method is not allowed for the requested resource.
         6. 406 Not Acceptable: The requested resource is not capable of generating content acceptable according to the Accept headers sent in the request.
         7. 407 Proxy Authentication Required: The client must first authenticate itself with the proxy.
         8. 408 Request Timeout: The server timed out waiting for the request.
         9. 409 Conflict: The request could not be completed due to a conflict with the current state of the resource.
         10. 410 Gone: The requested resource is no longer available and will not be available again.
         11. 411 Length Required: The server refuses to accept the request without a defined Content-Length header.
         12. 412 Precondition Failed: One or more conditions in the request headers were not met.
         13. 413 Payload Too Large: The request payload is larger than the server is willing to process.
         14. 414 URI Too Long: The requested URI is too long for the server to process.
         15. 415 Unsupported Media Type: The media type of the request is not supported by the server.
         16. 416 Range Not Satisfiable: The requested range is not available for the resource.
         17. 417 Expectation Failed: The server cannot meet the expectations of the Expect header.
      5. 5xx Server Errors
         1. 500 Internal Server Error: The server encountered an unexpected condition that prevented it from fulfilling the request.
         2. 501 Not Implemented: The server does not support the functionality required to fulfill the request.
         3. 502 Bad Gateway: The server received an invalid response from the upstream server.
         4. 503 Service Unavailable: The server is currently unable to handle the request due to temporary overload or maintenance.
         5. 504 Gateway Timeout: The server did not receive a timely response from the upstream server.
         6. 505 HTTP Version Not Supported: The server does not support the HTTP protocol version used in the request.
   8. HTTP methods
      1. GET/orders
      2. POST/orders
      3. PUT/orders/{order\_id}
      4. DELETE/orders/{order\_id}
   9. webSockets
      1. real time update
   10. Email related protocols
       1. SMTP: email transmission across the internet
       2. IMAP: User to retrieve emails from a serve (Internet message access protocol)
       3. POP3: user for downloading emails from a (Post office protocol version 3)
   11. File transfer protocols
       1. FTP: for transferring files over the internet
       2. SSH: for command-line log
   12. Real time communication
       1. webRTC: Enables browser-to-browser applications for voice calling, video chat, and file sharing
       2. MQTT: lightweight messaging protocol (message Queuing telemetry transport)
       3. AMQP: protocol for message-oriented middleware (Advanced message queuing protocol)
   13. RPC-Remote procedure call
5. CRUD operations:
   1. Create – POST: /api/products
   2. Read – GET: /api/products
   3. Update – PUT: /api/products/:id
   4. Delete - DELECT: /api/products/:id
6. API paradigms
   1. REST - representational state transfer
      1. Stateless
      2. Standard HTTP methods
      3. Over-fetching under-fetching
      4. JSON for data exchange
   2. GraphQL
      1. Avoids over-fetching and under-fetching
      2. Strongly typed schema based queries
      3. Queries can impact serve performance
      4. Only POST requests
      5. Responds with HTTP 200
   3. GRPC
      1. Build on HTTP/2
      2. Multiplexing/server push
7. Types of proxy servers
   1. Forward proxy
   2. Reverse proxy
   3. open proxy
   4. transparent proxy
   5. anonymous proxy
   6. Distorting proxy
   7. High Anonymity proxy
8. Forward proxy use cases
   1. Instagram proxies
   2. Internet use control
   3. Anonymizing web access
9. Reverse proxy use cases
   1. Load balancers
   2. CDNs
   3. Firewalls (WAFs)
   4. SSL offloading
10. Load balancer
    1. Round robin  
       
    2. Least connection   
       
    3. IP Hash  
       
    4. Geographic  
       
    5. Consistent hashing  
       
11. Hardware load balancers
    1. F5
    2. Citrix
12. Software load balancers
    1. Haproxy
    2. Nginx
13. Cloud-based load balancers
    1. Aws’s elastic load balancing
    2. Microsoft Azure load balancer
    3. Google cloud’s load balancer
14. Redundancy
15. Health checks & monitoring
16. Auto-scaling and self-healing systems
17. DNS failover
18. Relational database
    1. Postgresql
    2. MySQL
    3. SQLite
    4. ACID properties
       1. A 🡪 Atomicity
       2. C 🡪 Consistency
       3. I 🡪 Isolation
       4. D 🡪 Durability
    5. NoSQL databases
       1. MongoDB - A document-oriented database known for its flexibility and scalability.
       2. Cassandra - A distributed database designed for handling large amounts of data across many commodity servers.
       3. Redis - An in-memory key-value store used for caching and real-time applications.
       4. CouchDB - A document-oriented database that uses a schema-free JSON document format.
       5. RavenDB - A document database designed for .NET applications with high performance and scalability.
       6. DynamoDB - Amazon's fully managed key-value and document database service.
       7. Neo4j - A graph database that focuses on the relationships between data points.
       8. Elasticsearch - A distributed search and analytics engine based on Lucene, often used for log and text search.
       9. HBase - A column-family store that runs on top of the Hadoop Distributed File System (HDFS).
       10. ArangoDB - A multi-model database that supports document, graph, and key-value data models.
       11. Couchbase - A NoSQL database with a distributed architecture supporting key-value and document data models.
       12. OrientDB - A multi-model database that supports document, object, and graph models.
    6. SQL database
       1. MySQL - An open-source relational database management system known for its speed and reliability.
       2. PostgreSQL - An open-source relational database with advanced features and extensibility.
       3. Microsoft SQL Server - A relational database management system developed by Microsoft with strong integration into Windows and .NET.
       4. Oracle Database - A powerful and widely used relational database system with advanced features for enterprise environments.
       5. SQLite - A lightweight, file-based database engine often used in embedded systems and mobile applications.
       6. MariaDB - A fork of MySQL with additional features and community-driven development.
       7. IBM Db2 - A family of data management products from IBM, including database servers designed for high availability and scalability.
       8. Amazon Aurora - A fully managed relational database service from AWS compatible with MySQL and PostgreSQL.
       9. SAP HANA - An in-memory relational database from SAP designed for real-time data processing.
       10. Google Cloud SQL - A fully managed SQL database service from Google Cloud, compatible with MySQL, PostgreSQL, and SQL Server.
    7. In-memory database
       1. Redis - A widely used in-memory key-value store that supports various data structures and is often used for caching.
       2. Memcached - A high-performance, distributed memory caching system used to speed up dynamic web applications.
    8. Vertical scaling (scale up)
       1. Increasing CPU power
       2. Adding more RAM
       3. Adding more disk storage
       4. Upgrading network
    9. Horizontal scaling (scale out)
       1. Database sharding
          1. Sharding strategies: range-based sharding: based on the range of a given key
          2. Directory-based Sharding: Lookup service to direct traffic to the database
          3. Geographical Sharding: Based on geographic location.
       2. Replication
    10. Database performance
        1. Caching
        2. Indexing
        3. Query optimization