18 Most-Used Linux Commands You Should Know

blog.bytebytego.com

ls List files and directories	**cd** Change current directory	**mkdir** Create new directory
rm Remove files or directories	**mv** Move or rename files or directories	**chmod** Change file or directory permission
cp Copy files or directories	**find** Search for files or directories	**grep** Search for a pattern in files
vi Edit files using text editor	**cat** Display the content of files	**tar** Manipulate tarball archive files
ps Display process information	**kill** Terminate process by sending a signal	**top** Display process and resource usage
ifconfig Configure network interfaces	**ping** Test network connectivity between hosts	**du** Estimate file space usage

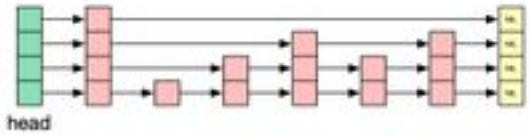
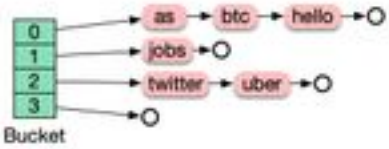
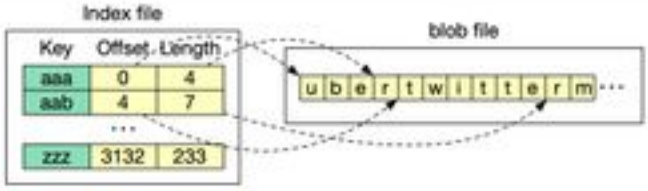
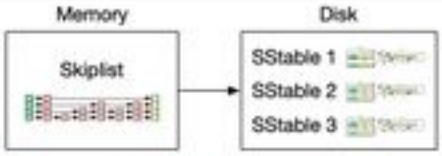
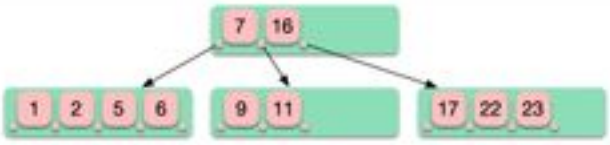
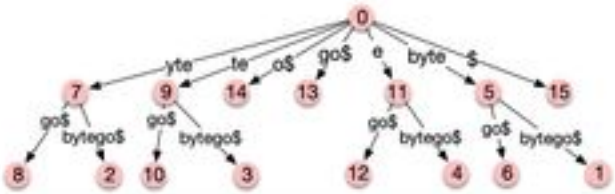
18 Key Design Patterns Every Developer Should Know

ByteByteGo.com

Abstract Factory Family creator Create groups of related items	**Builder** Lego master Build object step by step	**Prototype** Cloner Create copies from examples
Singleton The one and only With just one instance	**Adapter** Universal plug Connect different interfaces	**Bridge** Connector Link what is to how it works
Composite Tree builder Create tree-like structure	**Decorator** Customizer Add new features to existing object	**Facade** One-stop shop Single interface to all functions
Flyweight Space saver Share small, reusable items	**Proxy** Middle man Represent another object	**Chain of responsibility** Replayer Relay requests until it is handles
Command Task wrapper Turn a request into object	**Iterator** Explorer Assess element one by one	**Mediator** Hub Simplify communication between classes
Memento Capsule Capture and store object state	**Observer** Broadcaster Notify others about the change	**Visitor** Guests Explore an object without changing it

8 Data Structures That Power Your Databases

ByteByteGo.com

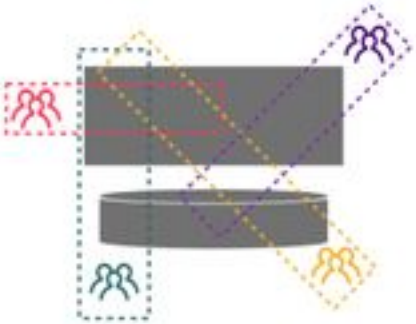
Types	Illustration	Use Case	Note
Skiplist		In-memory	used in Redis
Hash index		In-memory	Most common in-memory index solution
SSTable		Disk-based	Immutable data structure. Seldom used alone
LSM tree		Memory + Disk	High write throughput. Disk compaction may impact performance
B-tree		Disk-based	Most popular database index implementation
Inverted index		Search document	Used in document search engine such as Lucene
Suffix tree		Search string	Used in string search, such as string suffix match
R-tree		Search multi-dimension shape	Such as the nearest neighbor

Airbnb's Microservice Architecture

Monolith

2008 - 2017

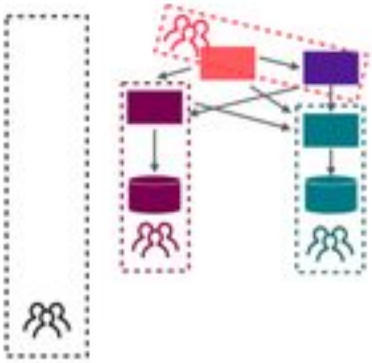
Confusing team ownership + unowned code



Microservices

2017 - 2020

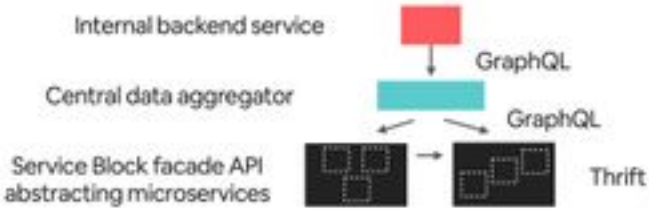
Dedicated service migration team



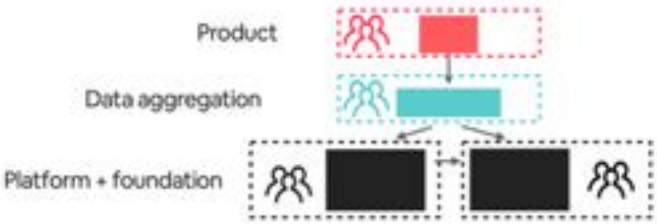
Micro + macroservices

2020 - present

Unified APIs



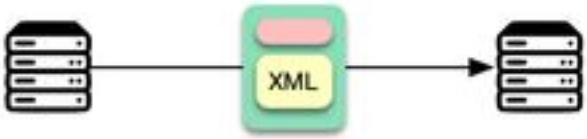
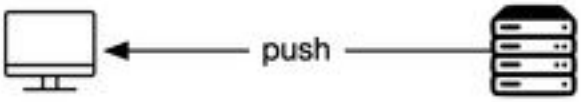
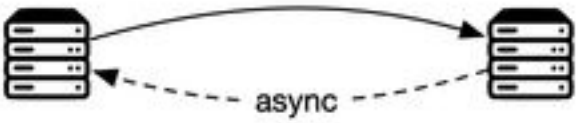
Specialized teams



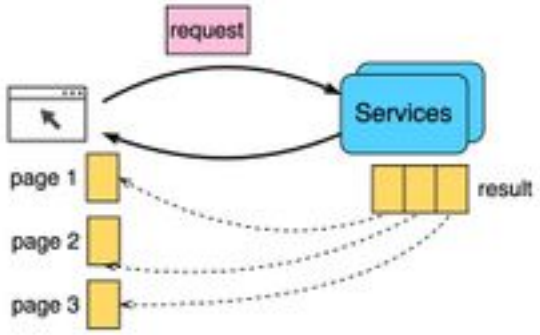
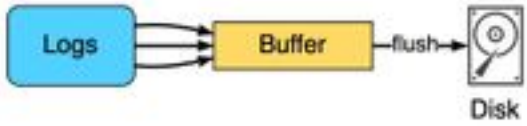
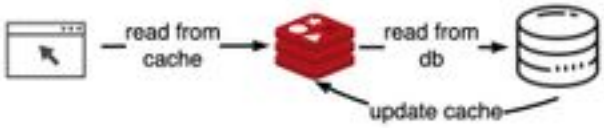
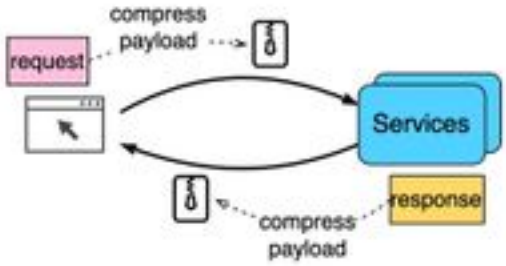
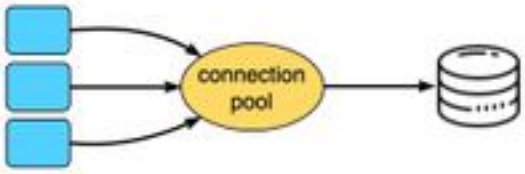
@jessicamtai

API Architecture Styles

ByteByteGo.com

Style	Illustration	Use Cases
SOAP		XML-based for enterprise applications
RESTful		Resource-based for web servers
GraphQL		Query language reduce network load
gRPC		High performance for microservices
WebSocket		Bi-directional for low-latency data exchange
Webhook		Asynchronous for event-driven application

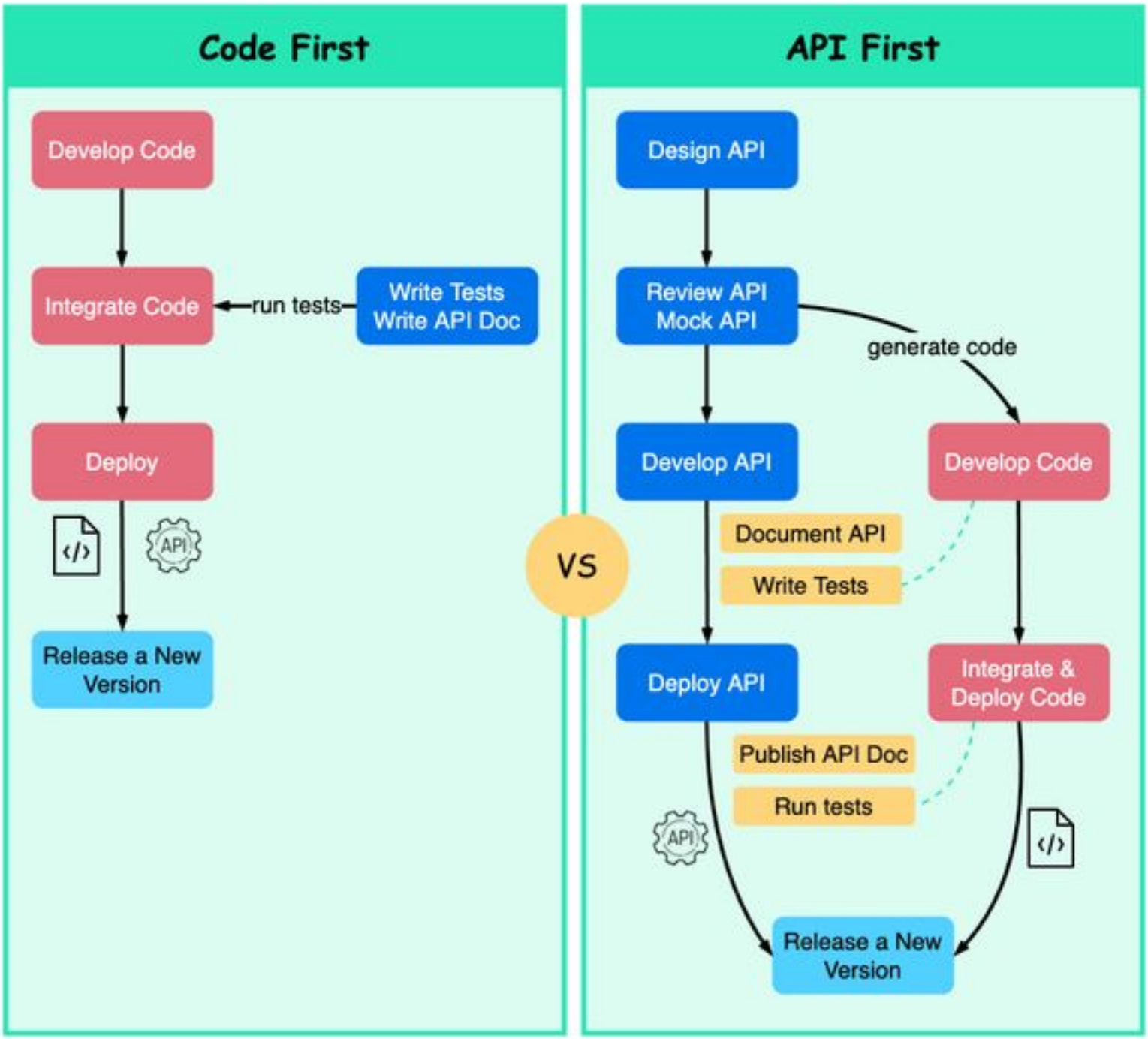
How to Improve API Performance?

PAGINATION		• an ordinal numbering of pages • handles a large number of results
ASYNC LOGGING		• send logs to a lock-free ring buffer and return • flush to the disk periodically • higher throughput and lower latency
CACHING		• store frequently used data in the cache instead of database • query the database when there is a cache miss
PAYLOAD COMPRESSION		• reduce the data size to speed up the download and upload
CONNECTION POOL		• opening and closing DB connections add significant overhead • a connection pool maintains a number of open connections for applications to reuse

Reference: Rapid API

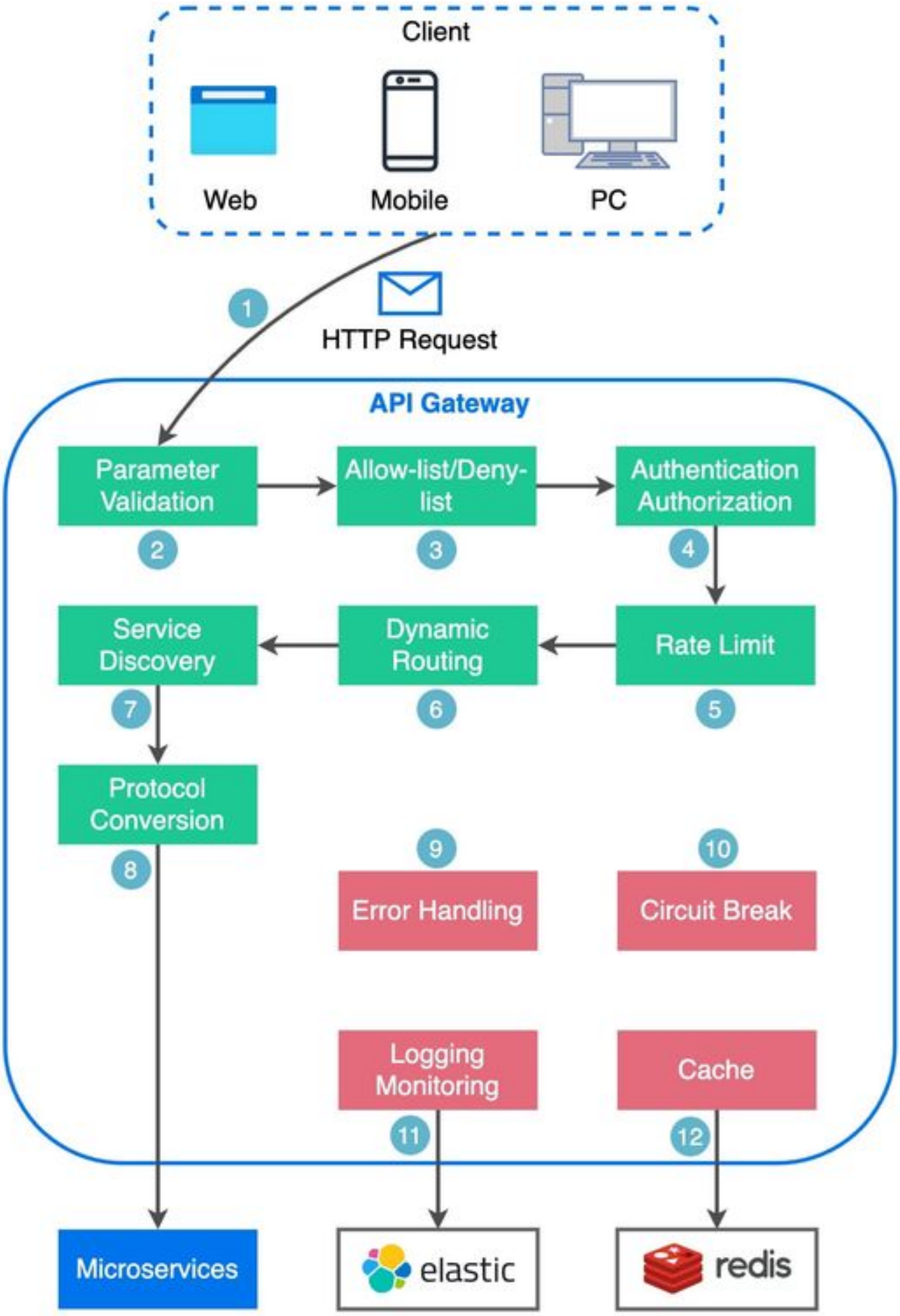
Code First v.s API First Development

blog.bytebytego.com



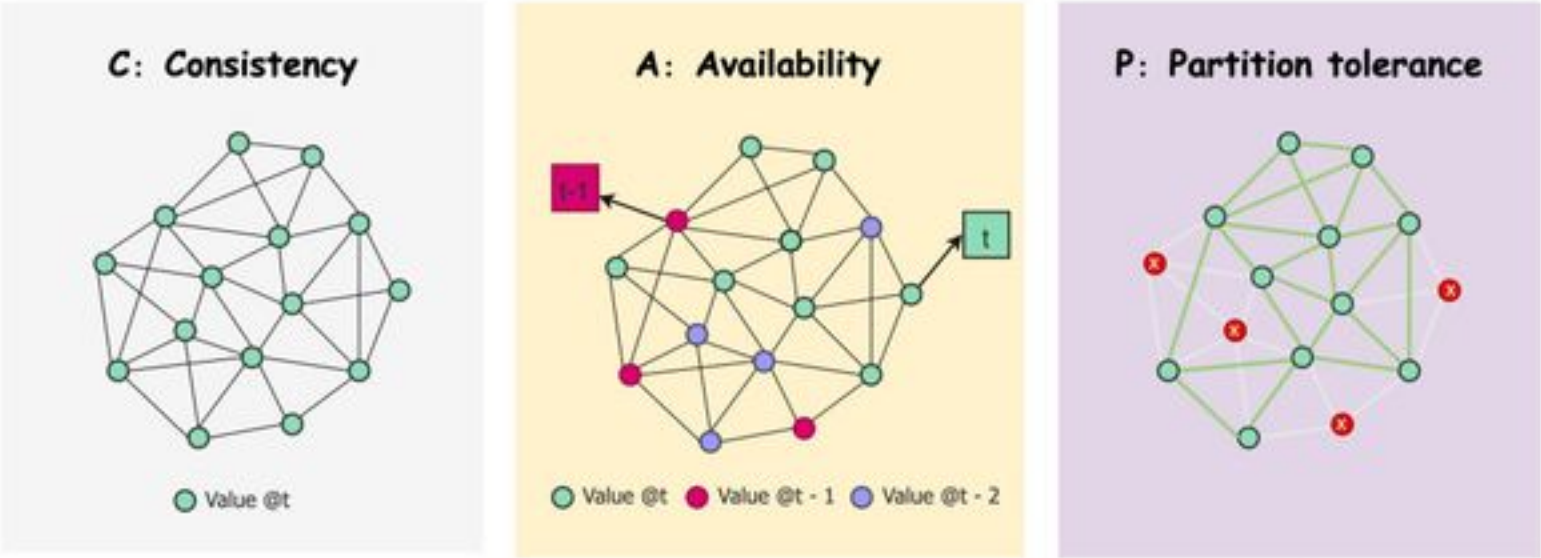
What does API Gateway do?

blog.bytebytego.com



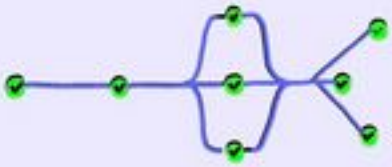
CAP Theorem

blog.bytebytego.com

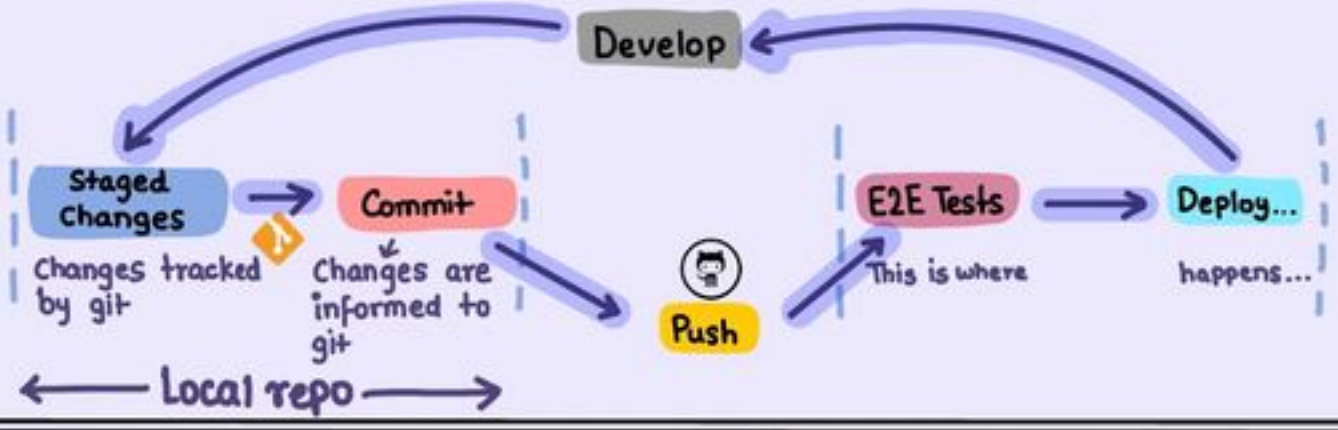


	CA systems	CP systems	AP systems
Sacrifice	No sacrifice	Availability	Consistency
Use Cases	Single-node only	Strong consitency. Banks, financial systems	Low latency. Consistency requirement is not high
Real-world examples	Single node RBMS (MySQL, Oracle)	Zookeeper, BigTable	Cassandra, CouchDB

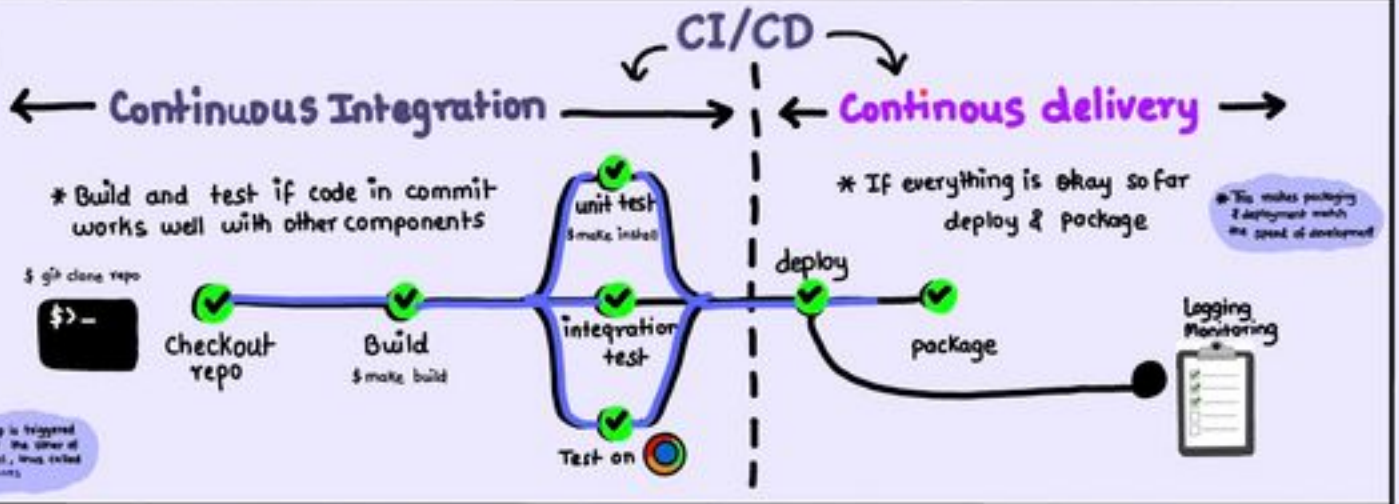
CI/CD Pipelines



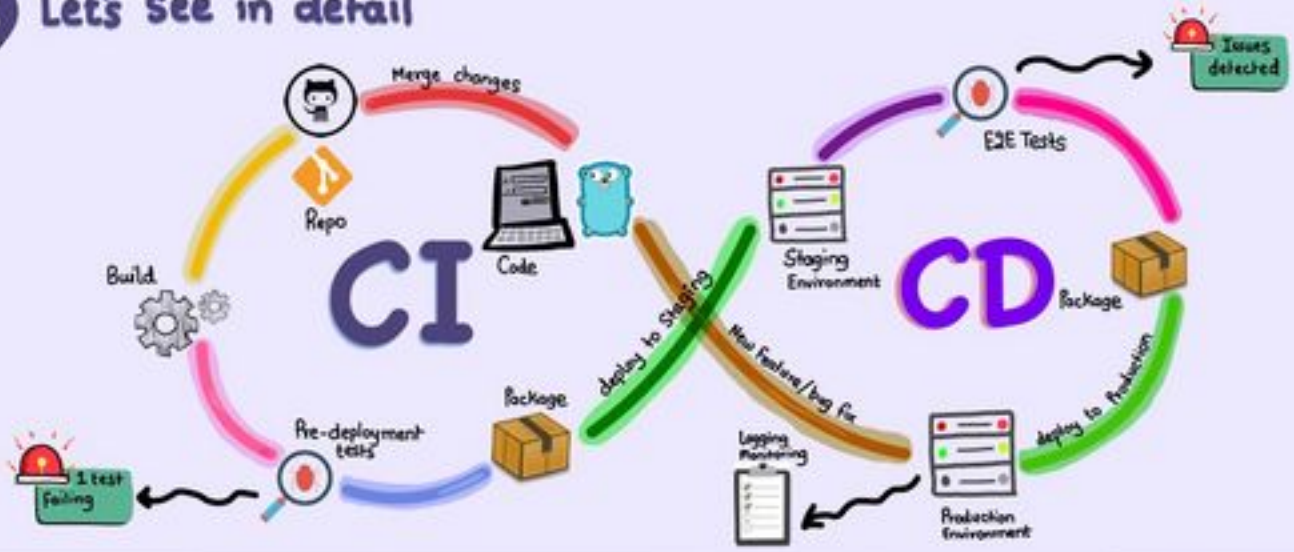
1 Software development Lifecycle



2

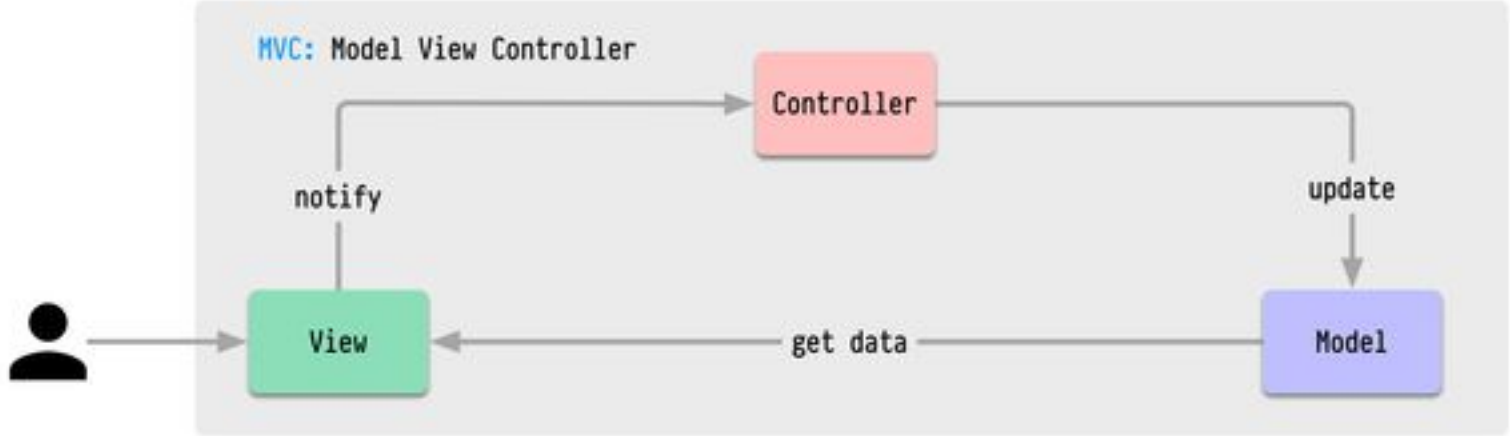


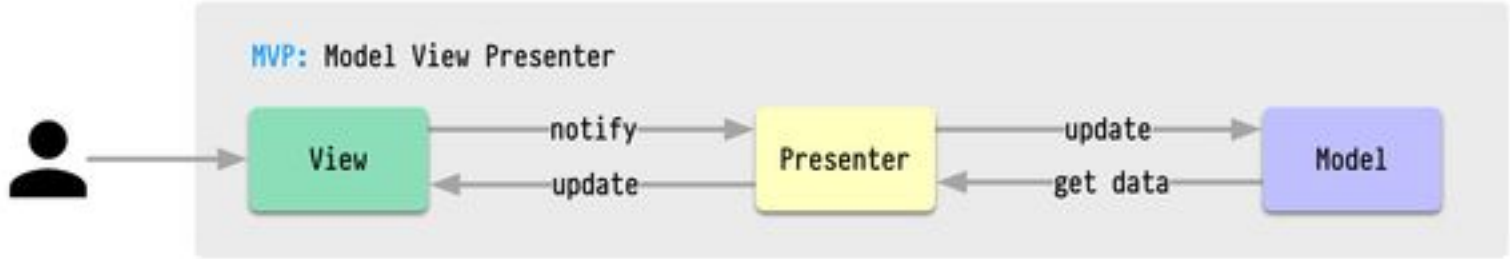
3 Let's see in detail

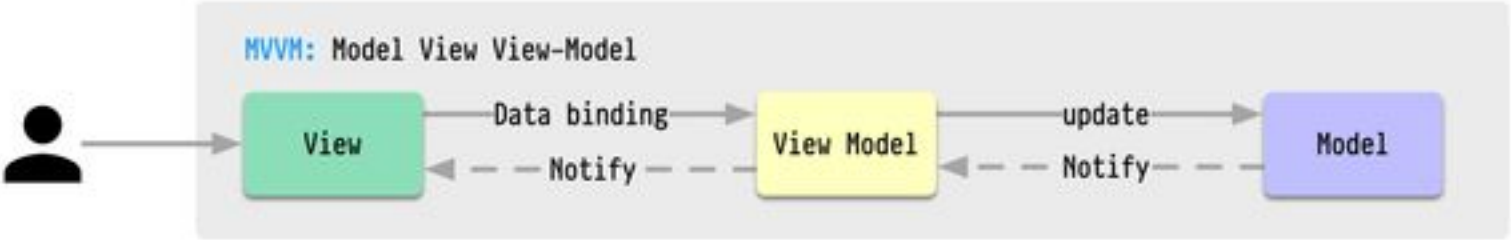


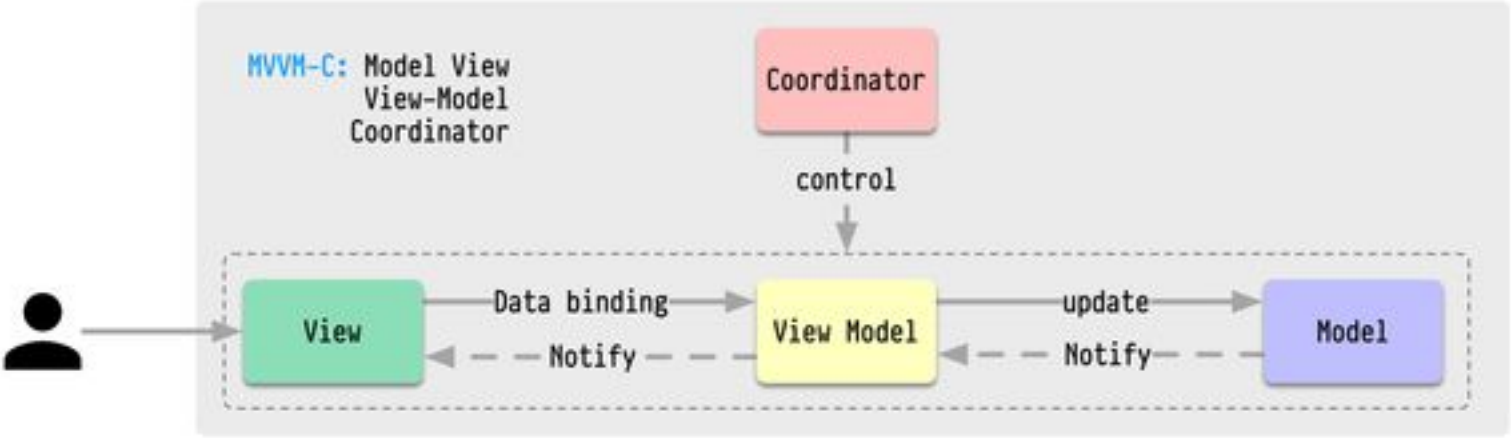
MVC, MVP, MVVM, VIPER patterns

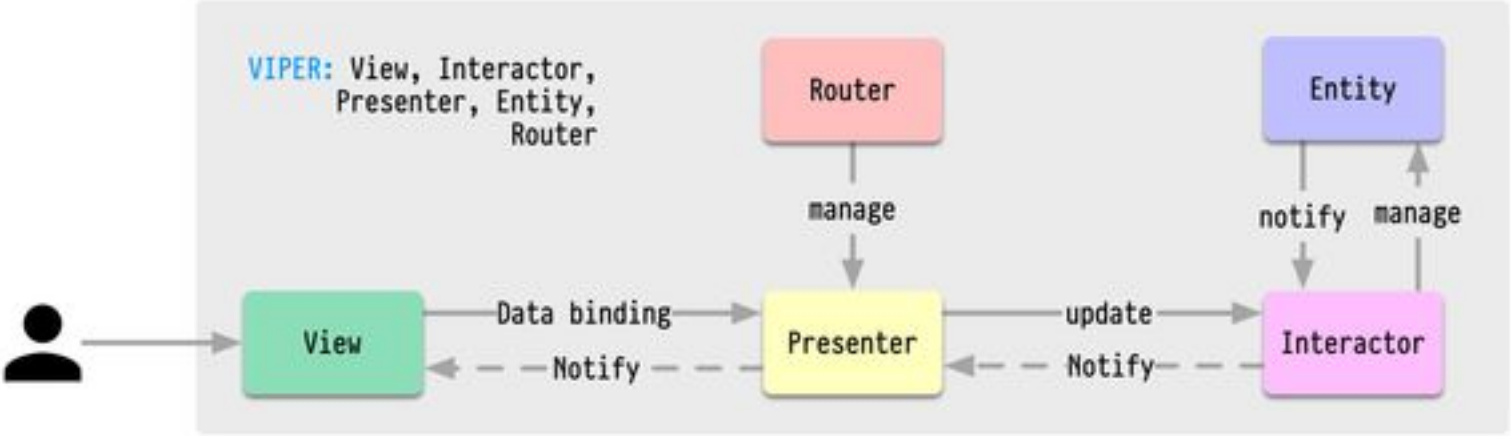
ByteByteGo.com











Cloud Comparison Cheat Sheet

blog.bytebytego.com

aws	Azure	Google Cloud	ORACLE CLOUD
Elastic Compute Cloud (EC2)	Virtual Machine	Compute Engine	Virtual Machine
Elastic Kubernetes Service (EKS)	Azure Kubernetes Service (AKS)	Google Kubernetes Engine (GKE)	Oracle Container Engine
Lambda	Azure Functions	Cloud Functions	OCI Functions
Simple Storage Service (S3)	Blob Storage	Cloud Storage	Object Storage
Elastic Block Store	Managed Disk	Persistent Disk	Persistent Volume
Elastic File System	File Storage	File Store	File Storage
Virtual Private Cloud	Virtual Network	Virtual Private Cloud	Virtual Cloud Network
Route 53	DNS	Cloud DNS	DNS
Elastic Load Balancing	Load Balancer	Cloud Load Balancing	Load Balancer
Web Application Firewall	Web Application Firewall	Cloud Armor	Web Application Firewall
RDS	SQL Database	Cloud SQL	ATP
DynamoDB	Cosmos DB	Firebase Realtime Database	NoSQL Database
Redshift	Synapse Analytics	BigQuery	Autonomous Data Warehouse
Elastic MapReduce	HDInsight	Dataproc	Big Data
Kinesis	Streaming Analytics	Dataflow	Streaming
SageMaker	Machine Learning	Vertex AI	Data Science
Glue	Data Factory	Data Fusion	Data Integration
EventBridge	Event Grid	Eventarc	Events
Simple Queuing Service	Storage Queues	Pub/Sub	Streaming
Simple Notification Service	Service Bus	Firebase Cloud Messaging	Notifications
CloudWatch	Monitor	Cloud Monitoring	Monitoring
CloudFormation	Resource Manager	Deployment Manager	Resource Manager
IAM	Active Directory	Cloud Identity	IAM
KMS	Key Vault	Cloud KMS	Vault

Cloud Database Cheat Sheet

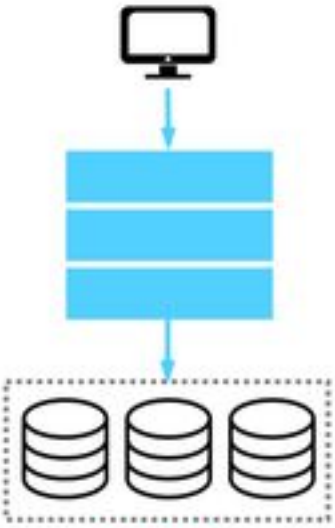
blog.bytebytego.com

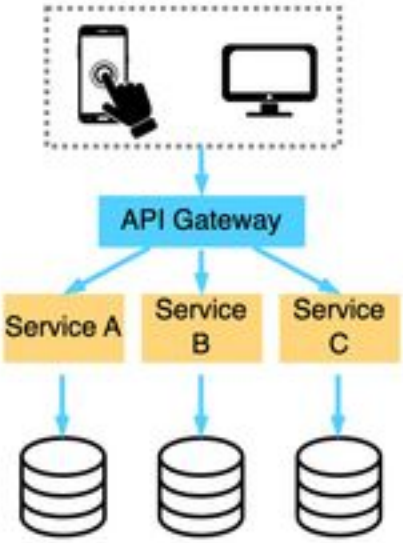
	DB Type	aws	Azure	Google Cloud	Open Source / 3rd Party
Structured	Relational	RDS	SQL Database	Cloud SQL	Oracle, PostgreSQL, MySQL, SQL Server, Snowflake, Click House
	Columnar	Redshift	Synapse Analytics	BigQuery	
Semi Structured	Key Value	DynamoDB	Cosmos DB	BigTable	Redis, Scylla
	In-Memory	ElastiCache	Azure Cache for Redis	Memory Store	Redis, Memcached
	Wide Column	Keyspaces	Cosmos DB	BigTable	Cassandra, Scylla
	Time Series	Timestream	Time Series Insights	BigTable, BigQuery	Influx, OpenTSDB
	Immutable Ledger	Quantum Ledger DB	Confidential Ledger	CloudSpanner	Hyper Ledger Fabric
	Geospatial	Keyspaces	Cosmos DB	BigTable	PostGIS, geomesa
	Graph	Neptune	Cosmos DB	CloudSpanner	OrientDB, Dgraph
	Document	Document DB	Cosmos DB	FireStore	MongoDB, Couchbase
	Text Search	OpenSearch	Cognitive Search	CloudSearch	Elastic search, Elassandra
UnStructured	Blob	S3	Blob Storage	Cloud Storage	Ceph, OpenIO

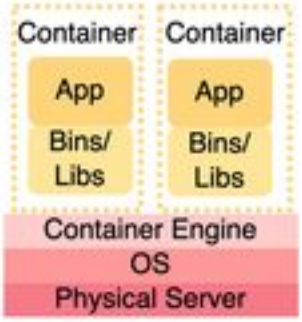
What is Cloud Native?

blog.bytebytego.com

	1980 - 1990	2000	2010 - Cloud
Development Process	Waterfall	Agile	DevOps
Application Architecture	Monolithic	N-Tier	Microservices
Deployment & Packaging	Physical server	Virtual server	Container
Application Infrastructure	Data center	Hosted	Cloud





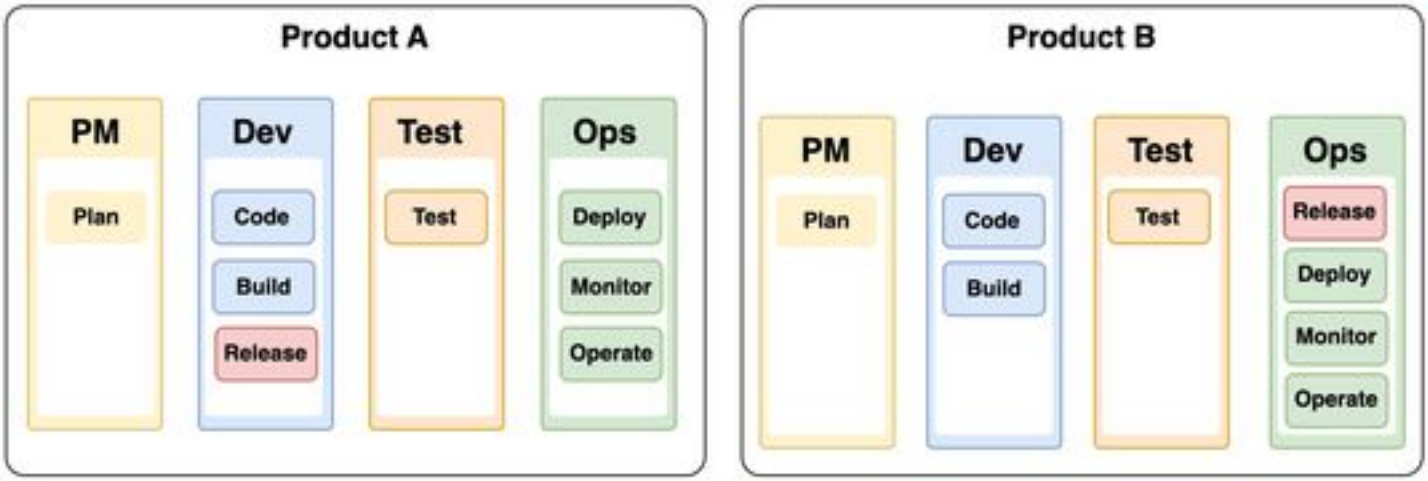


Reference: https://www.oracle.com/cloud/cloud-native/what-is-cloud-native/

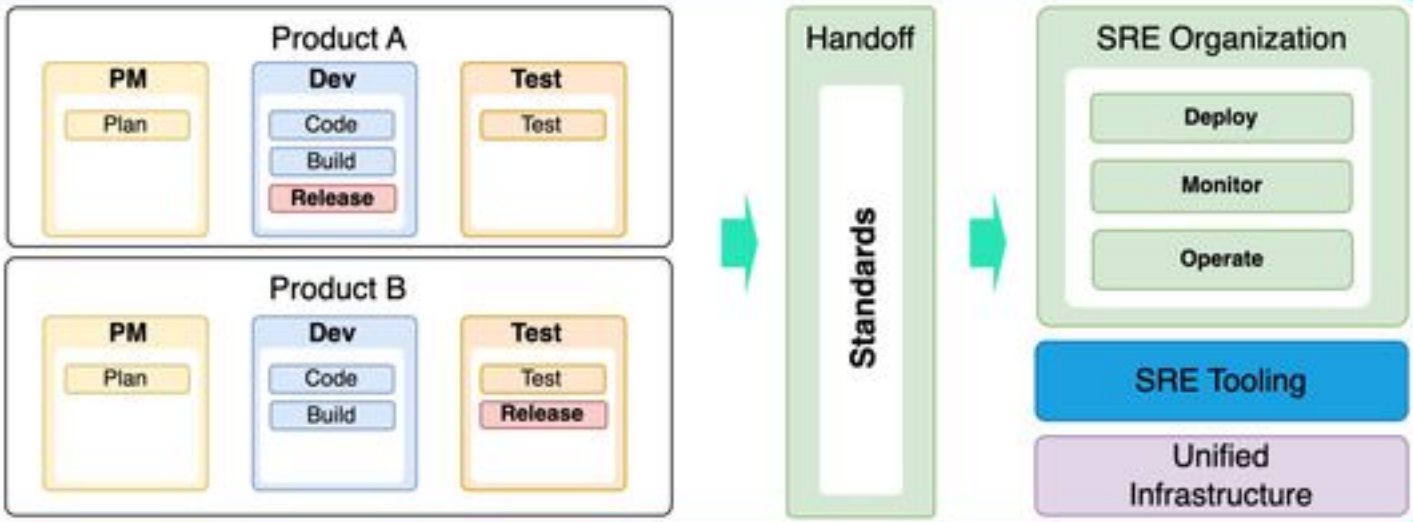
DevOps vs SRE vs Platform Engineering

blog.bytebytego.com

DevOps



SRE (Site Reliability Engineering)



Platform Engineering

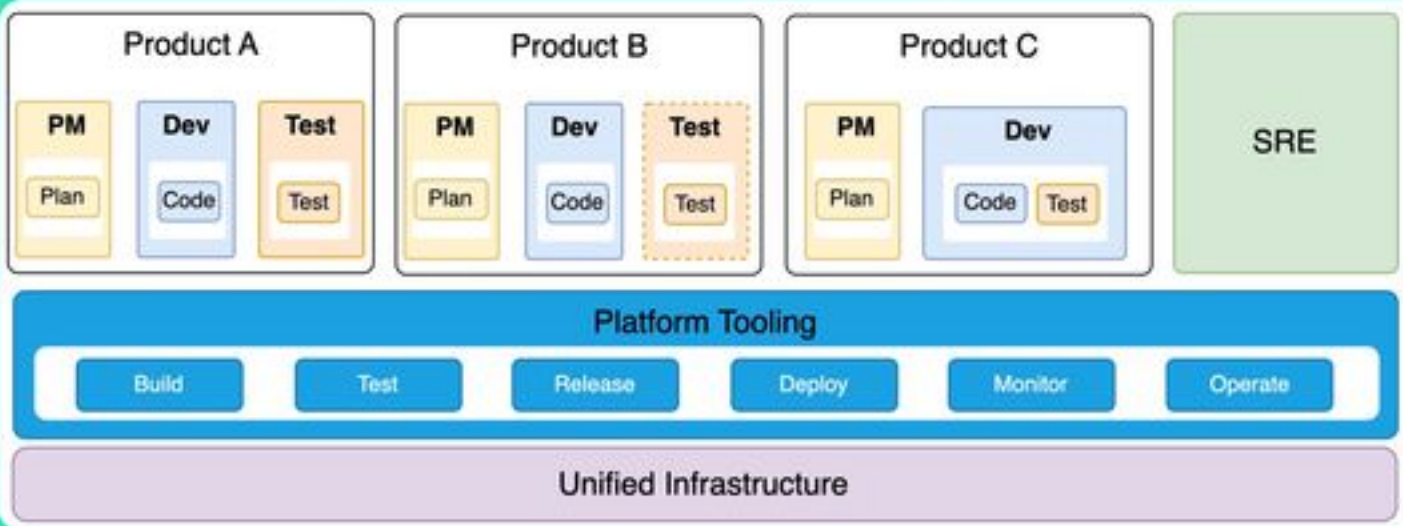
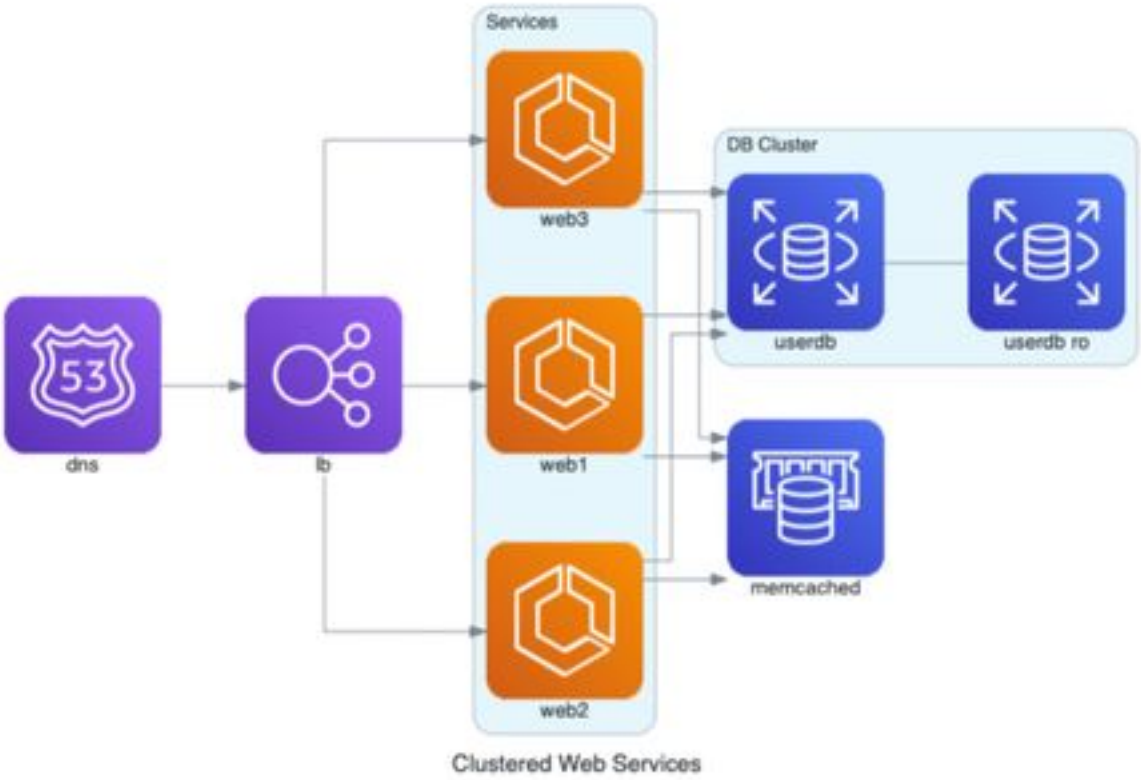


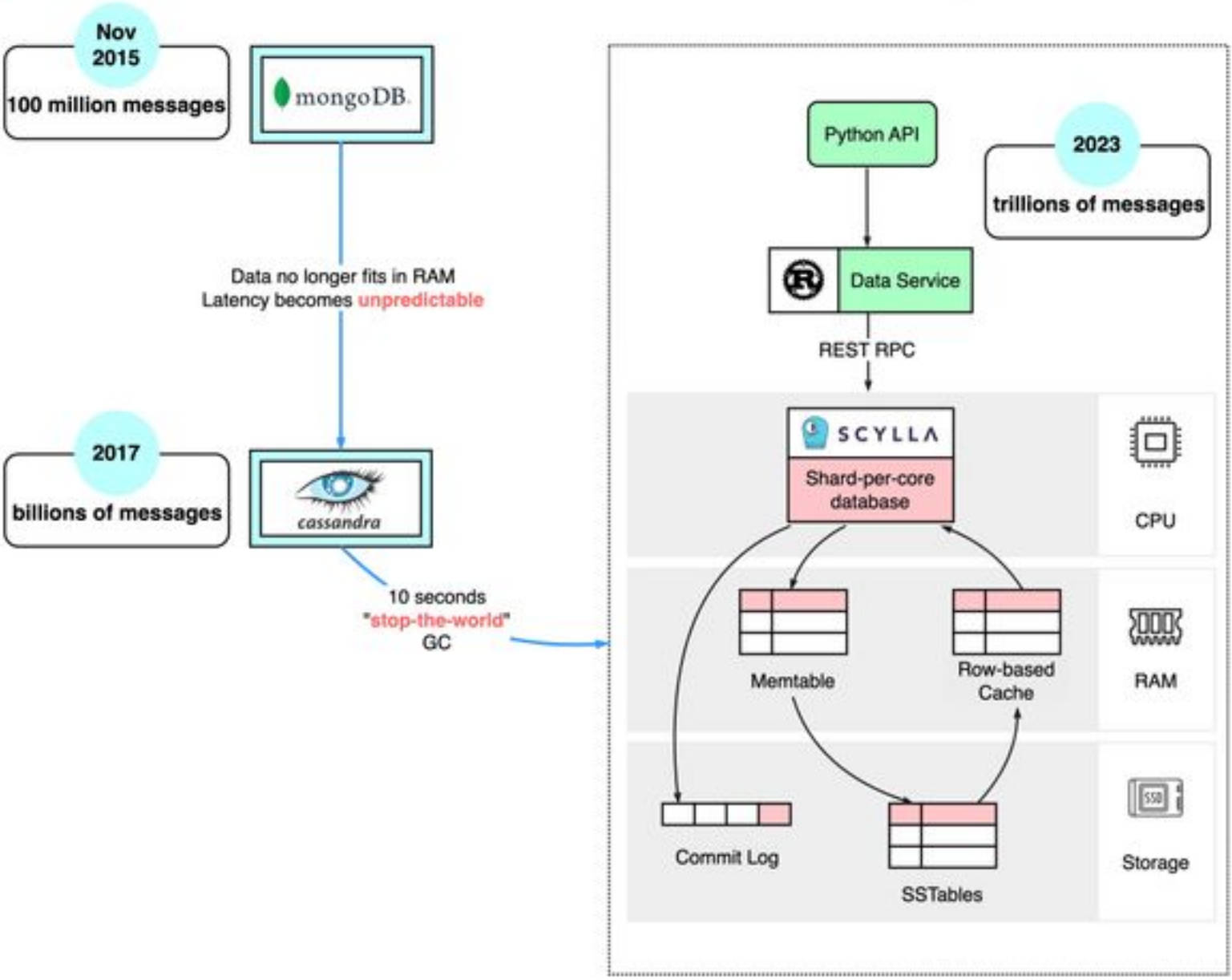
Diagram as Code

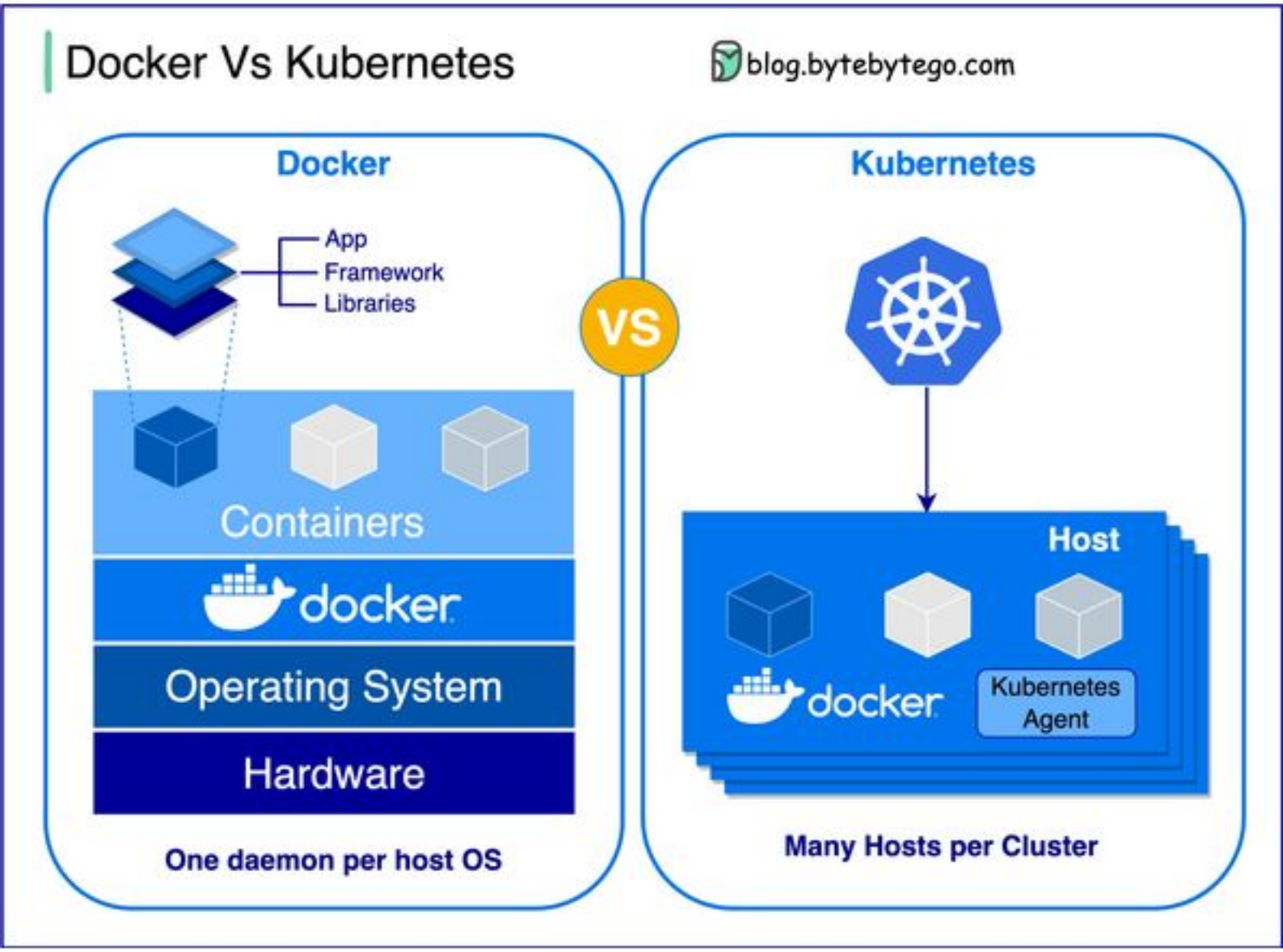
```
from diagrams import Cluster, Diagram
from diagrams.aws.compute import ECS
from diagrams.aws.database import ElastiCache, RDS
from diagrams.aws.network import ELB
from diagrams.aws.network import Route53

with Diagram("Clustered Web Services", show=False):
    dns = Route53("dns")
    lb = ELB("lb")

    with Cluster("Services"):
        svc_group = [ECS("web1"),
                     ECS("web2"),
                     ECS("web3")]

    with Cluster("DB Cluster"):
        db_primary = RDS("userdb")
        db_primary - [RDS("userdb ro")]

    memcached = ElastiCache("memcached")

    dns >> lb >> svc_group
    svc_group >> db_primary
    svc_group >> memcached
```



How Discord Stores Trillions Of Messages

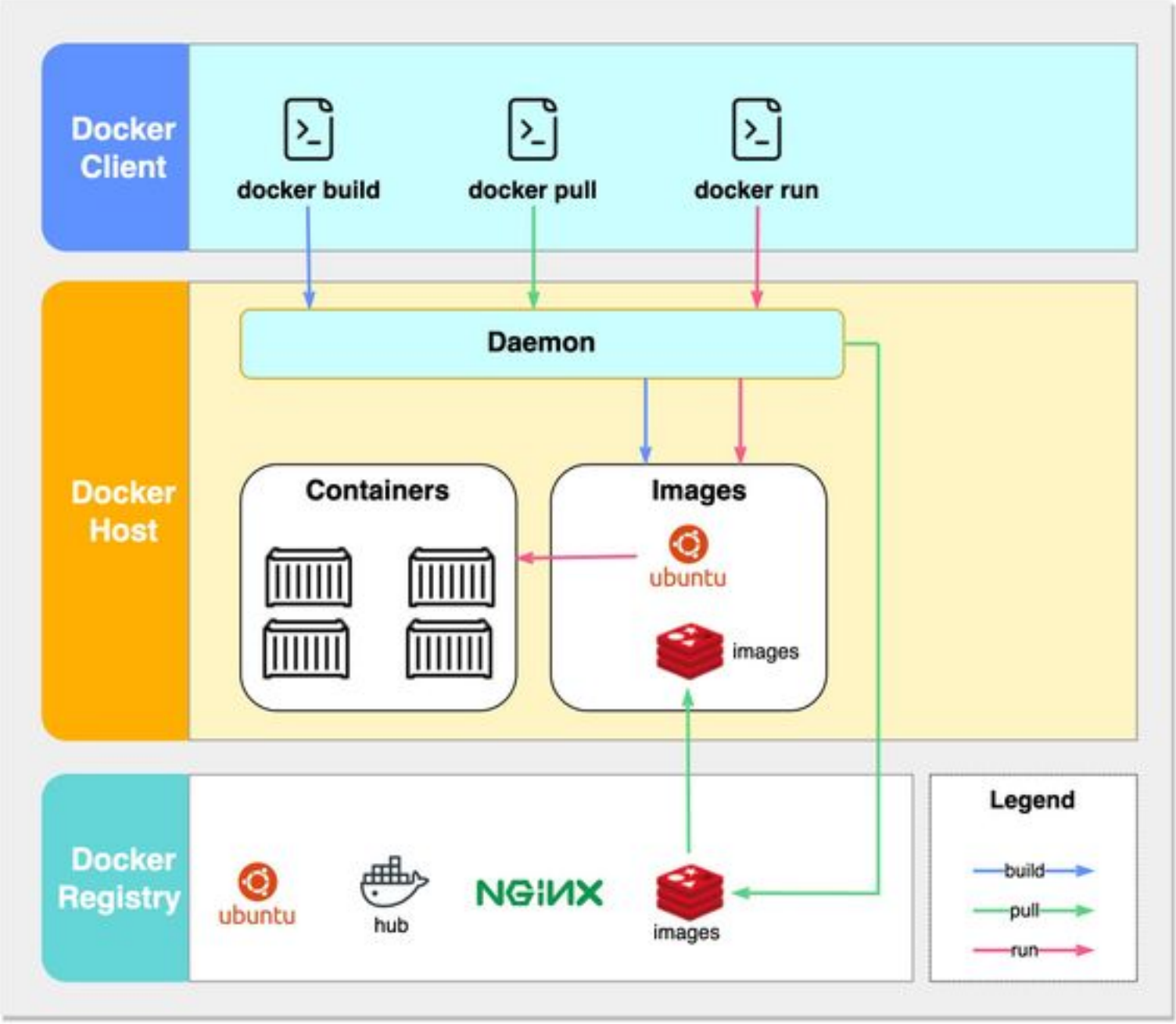
blog.bytebytego.com




Docker Vs Kubernetes
blog.bytebytego.com
Docker
App
Framework
Libraries
Containers
docker
Operating System
Hardware
One daemon per host OS
VS
Kubernetes
Host
docker
Kubernetes Agent
Many Hosts per Cluster

How does Docker Work?

blog.bytebytego.com

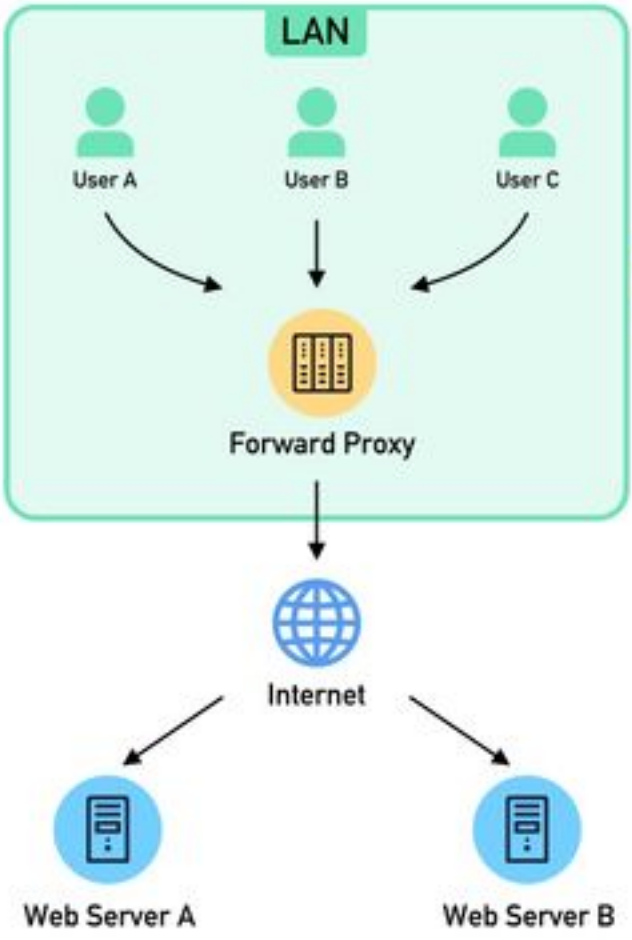


Forward Proxy v.s. Reverse Proxy

ByteByteGo.com

Forward Proxy

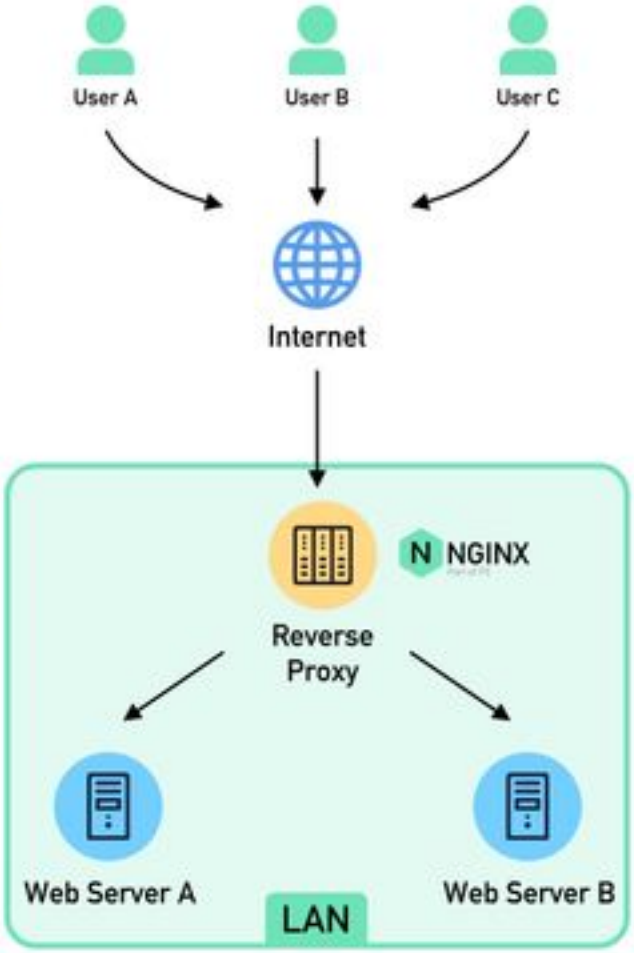
- Avoid browsing restrictions
- Block access to certain content
- Protect user identity online

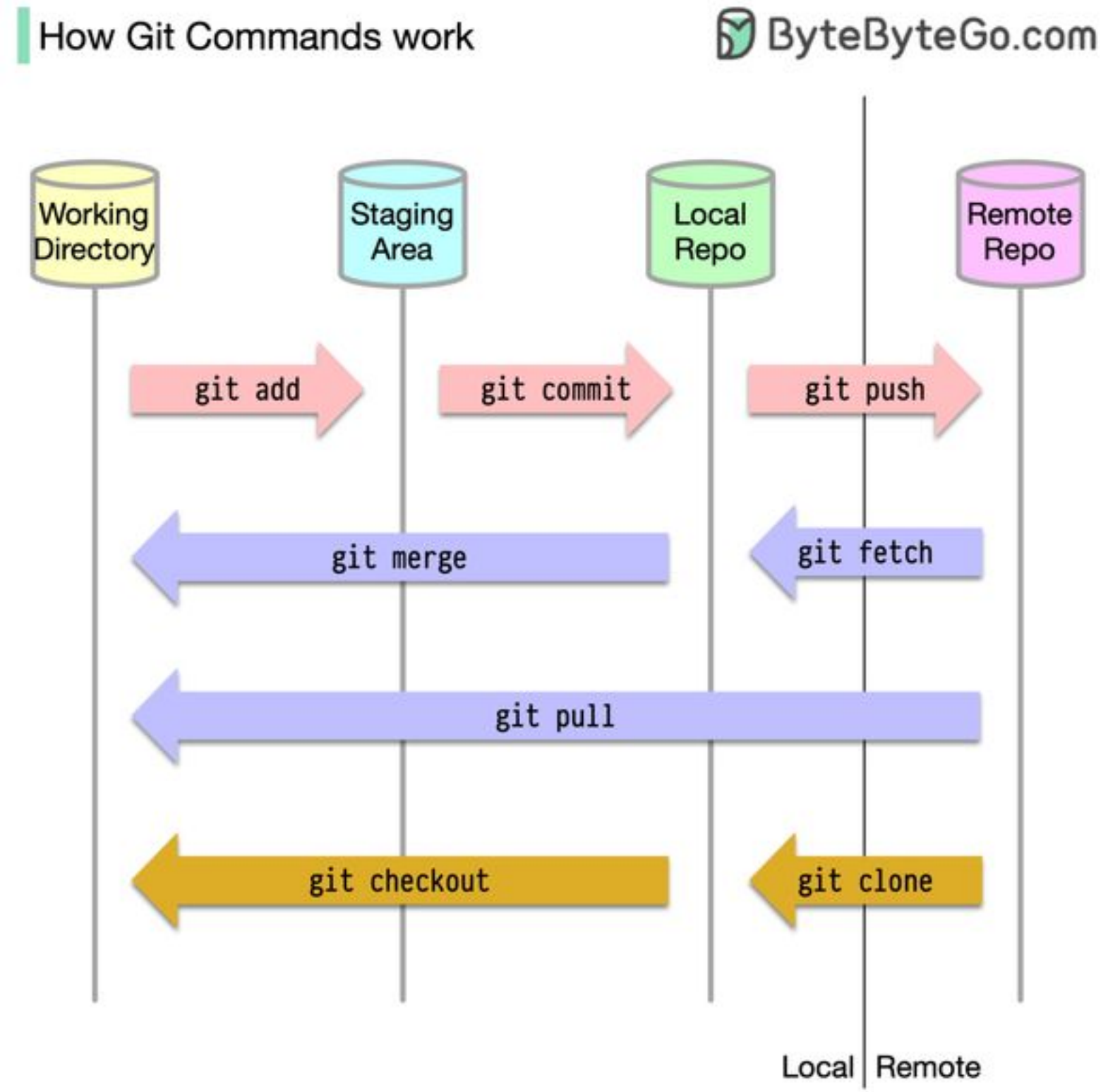


VS

Reverse Proxy

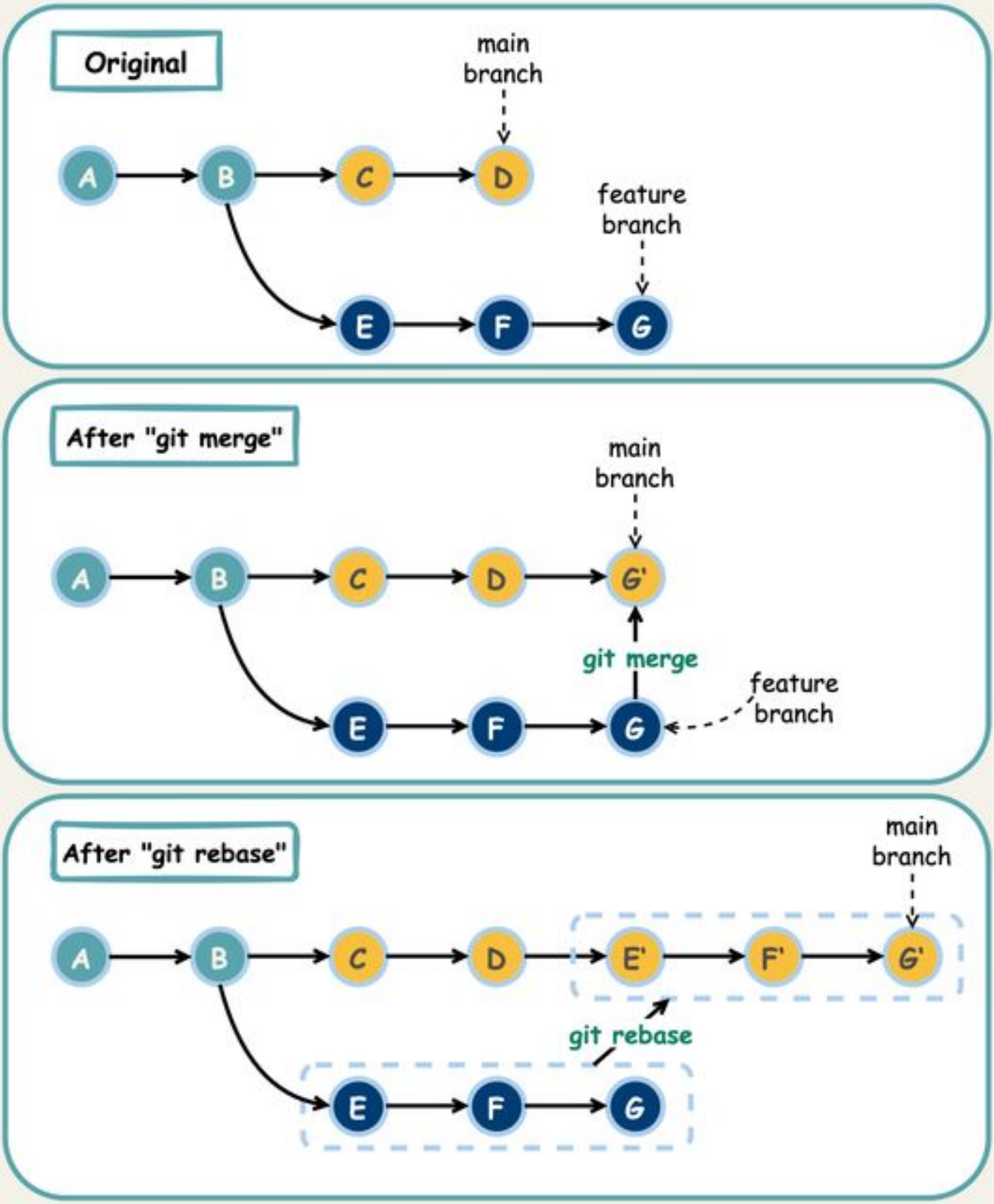
- Load balancing
- Protect from DDos attacks
- Cache static content
- Encrypt and decrypt SSL communications




How Git Commands work
ByteByteGo.com
Working Directory
Staging Area
Local Repo
Remote Repo
git add
git commit
git push
git merge
git fetch
git pull
git checkout
git clone
Local
Remote

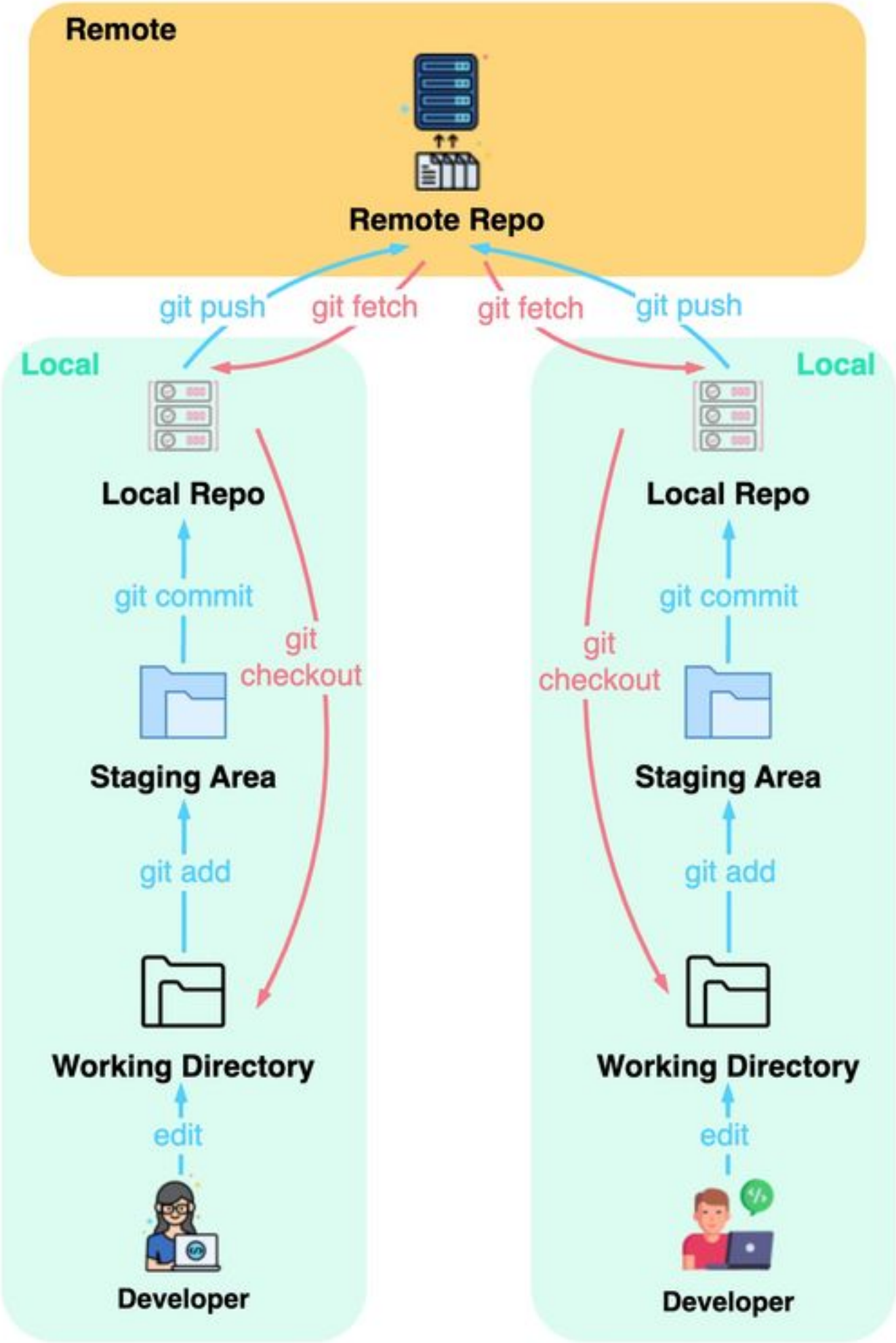
Git Merge vs. Git Rebase

blog.bytebytego.com



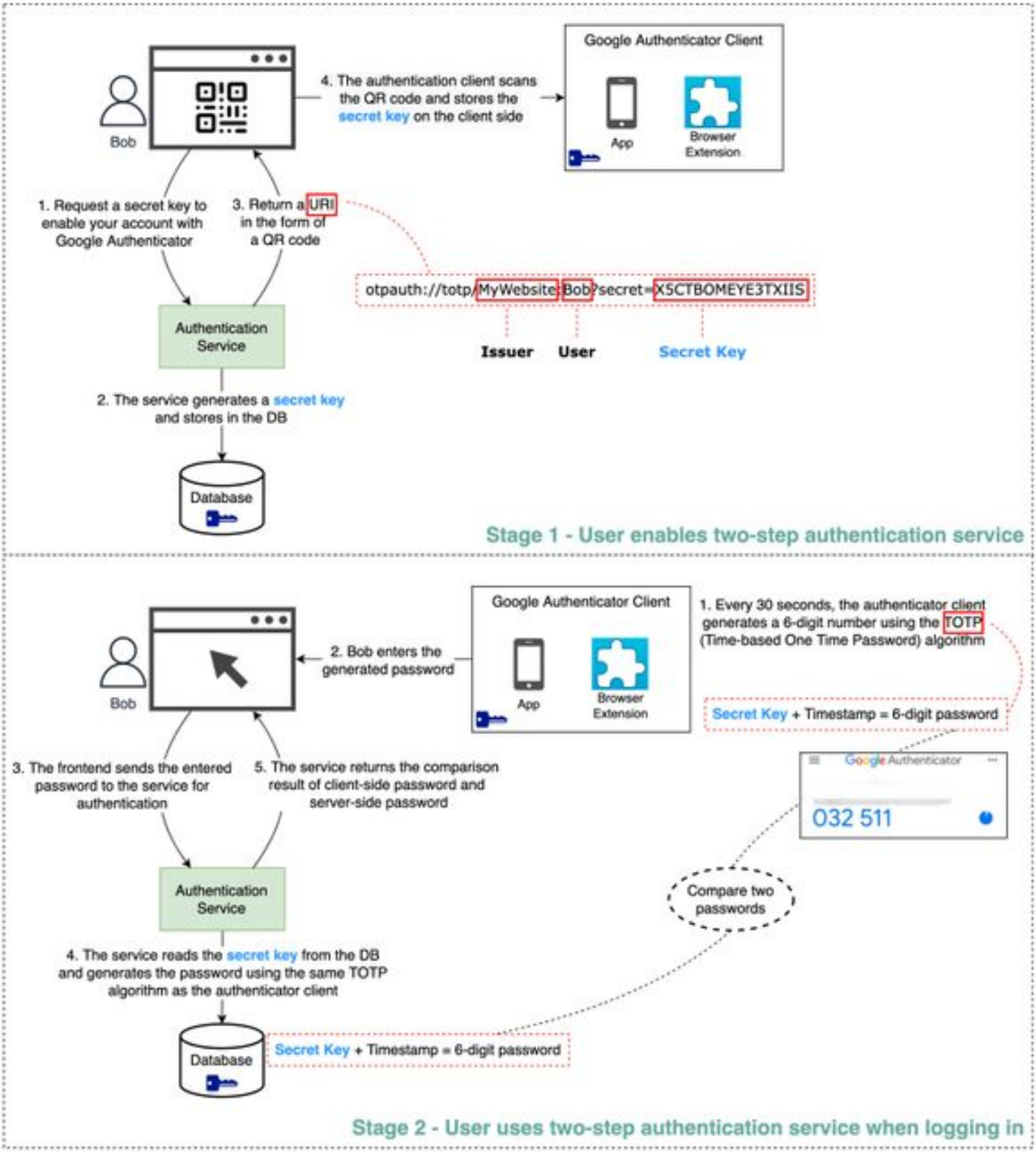
How does Git Work?

blog.bytebytego.com



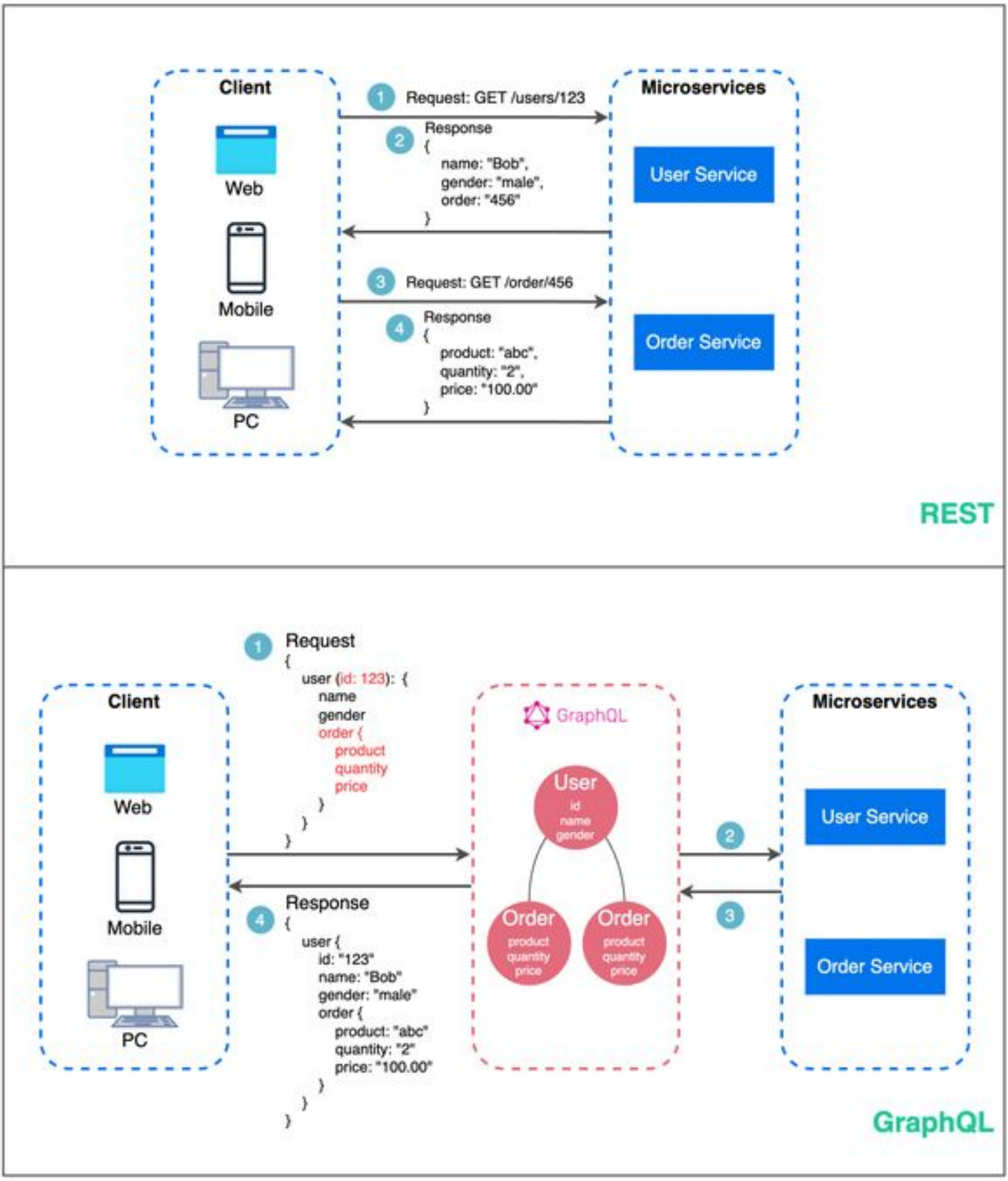
How does Google Authenticator Work?

blog.bytebytego.com



REST v.s. GraphQL

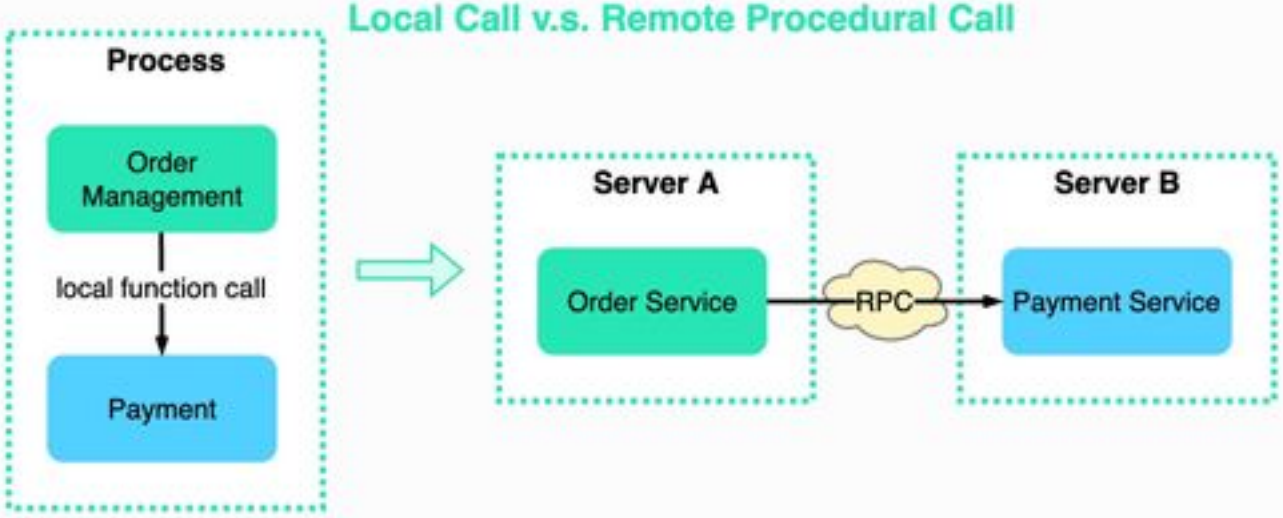
blog.bytebytego.com



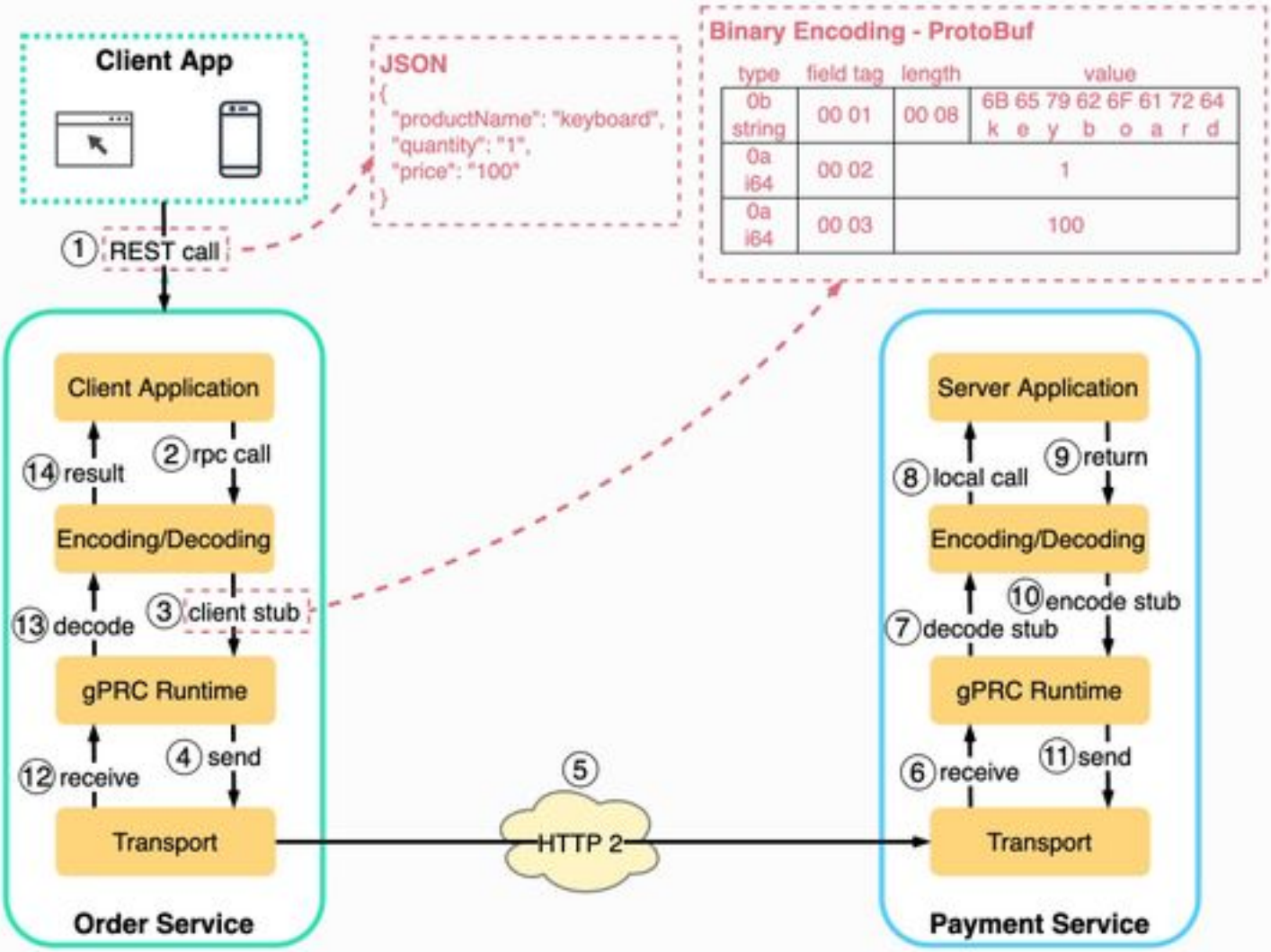
How does gRPC Work?

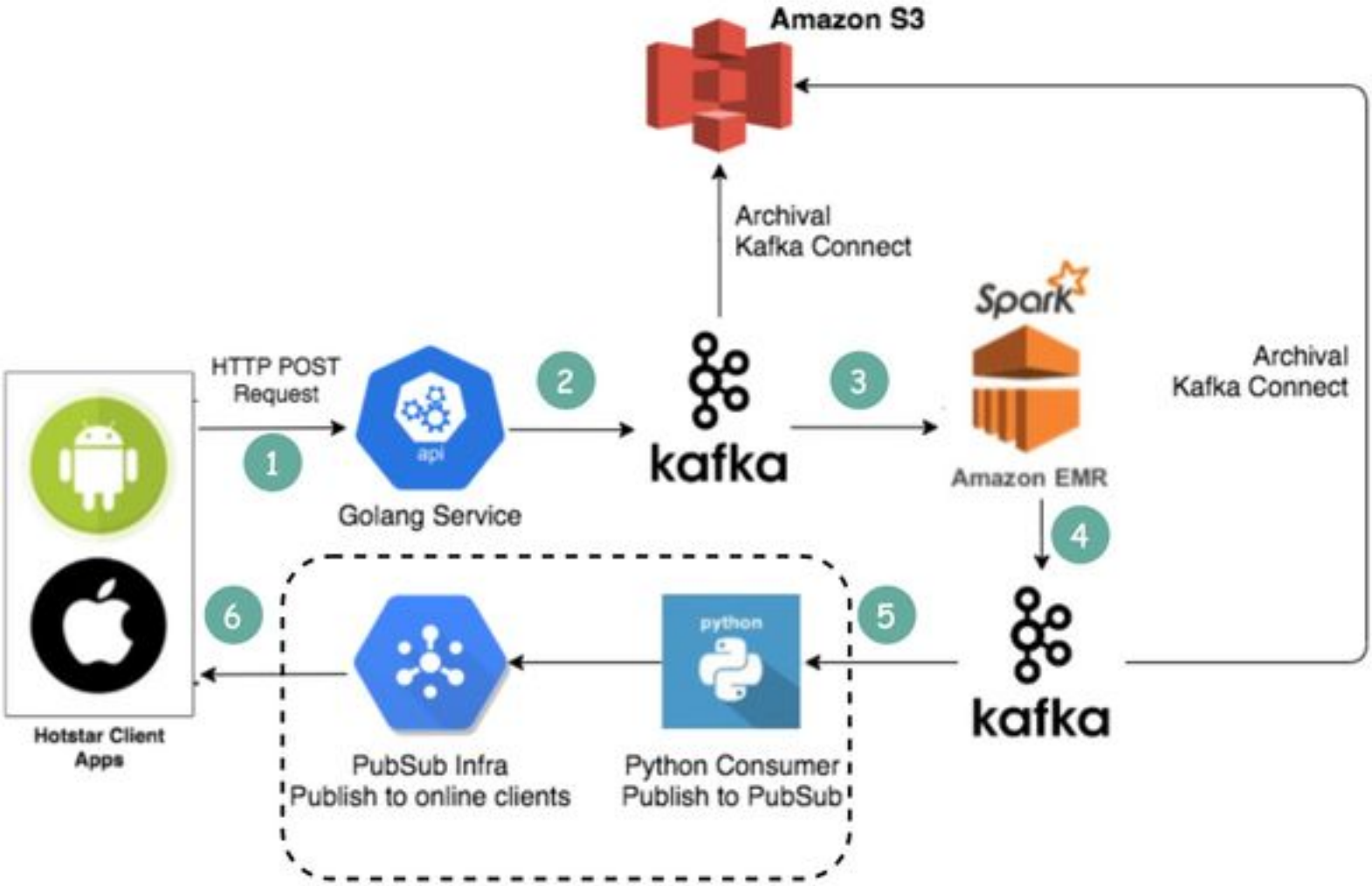
blog.bytebytego.com

Local Call v.s. Remote Procedural Call



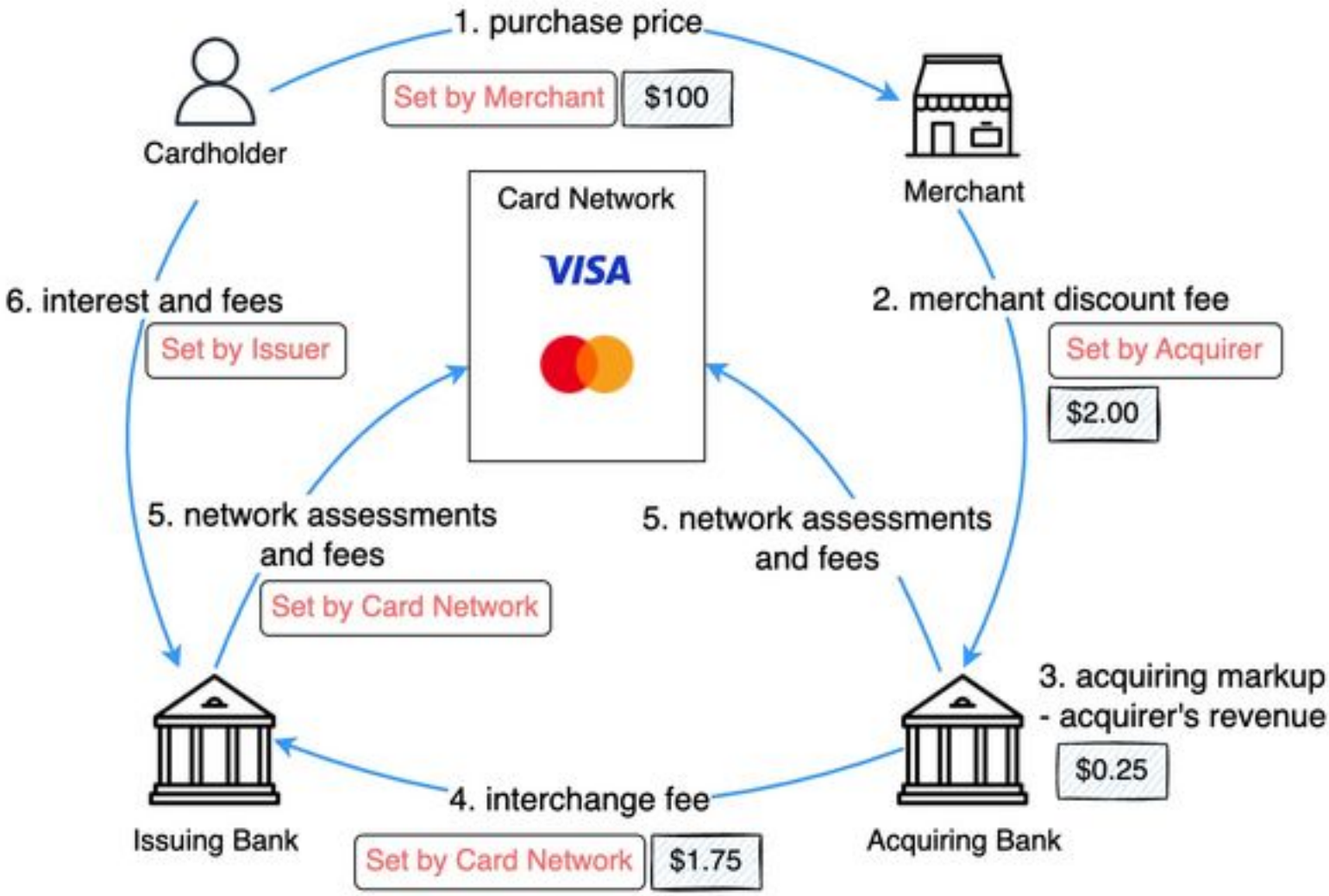
gRPC Overall Flow




Amazon S3
Archival
Kafka Connect
Spark
HTTP POST
Request
1
2
3
kafka
Amazon EMR
Archival
Kafka Connect
Golang Service
4
6
5
python
kafka
Hotstar Client
Apps
PubSub Infra
Publish to online clients
Python Consumer
Publish to PubSub

How does VISA Make Money?

blog.bytebytego.com



merchant discount fee = interchange fee + acquiring markup

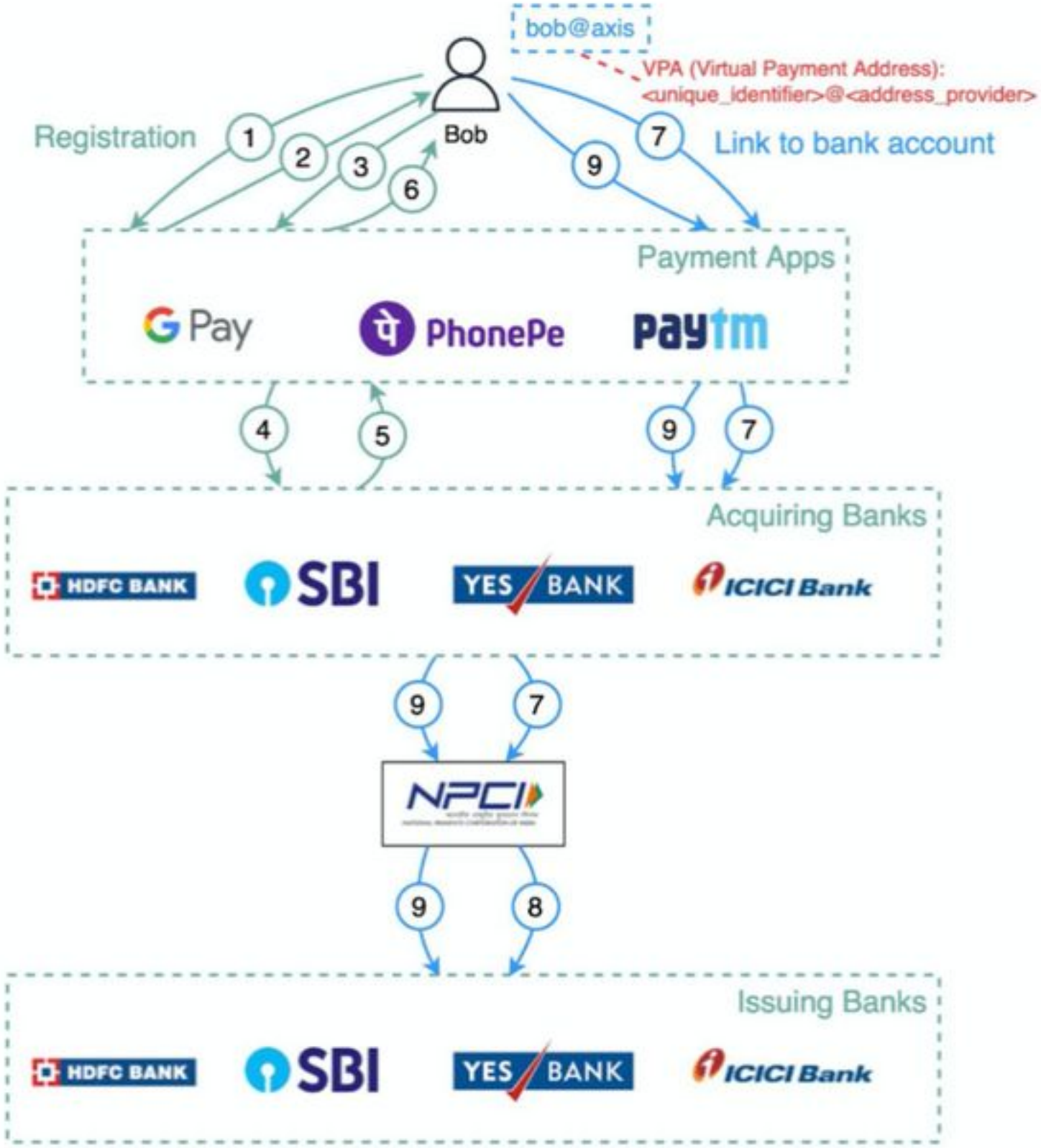
$2.00 $1.75 $0.25

The merchant needs to compensate issuer and acquirer

How does UPI Work?

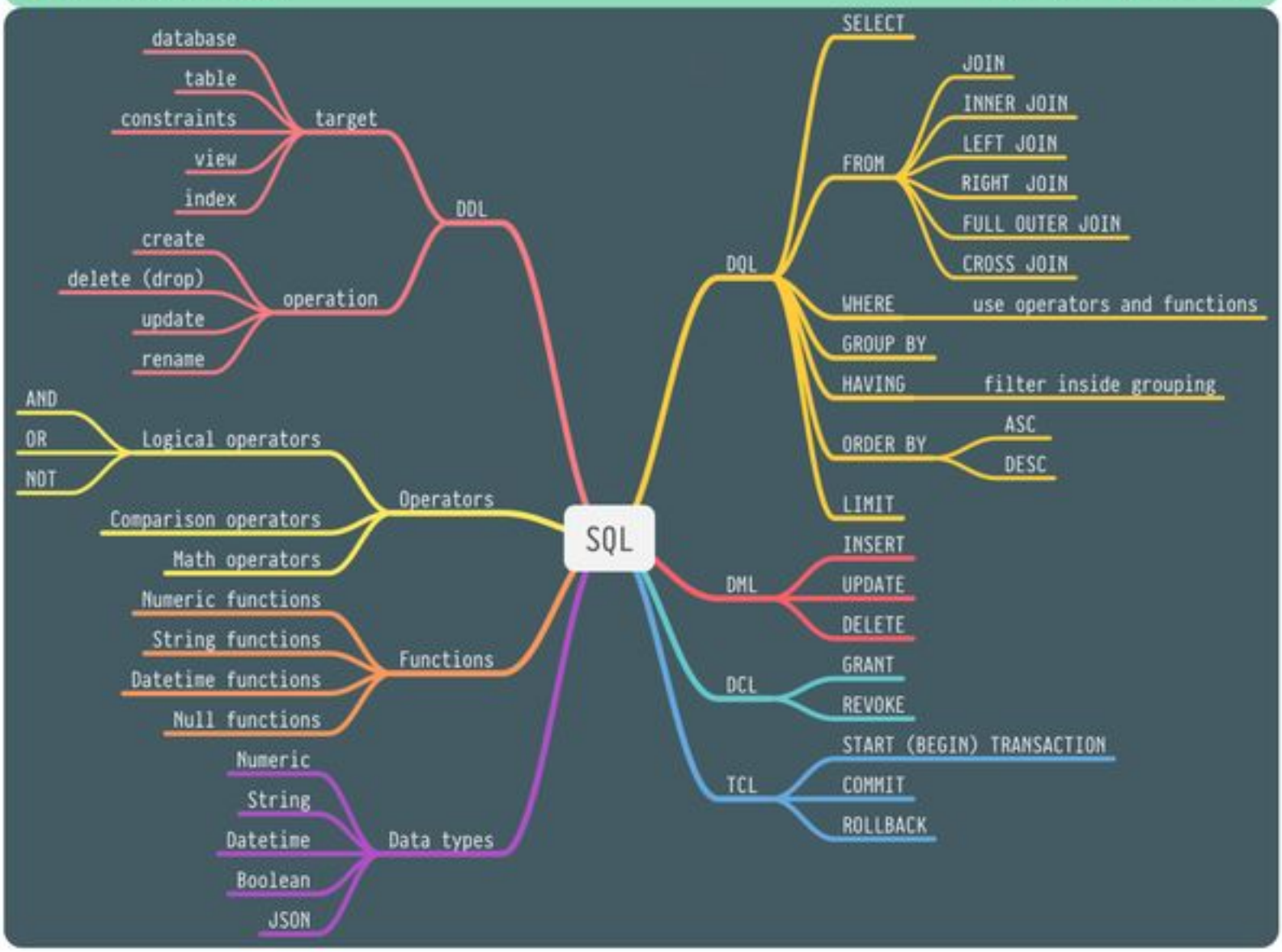
blog.bytebytego.com

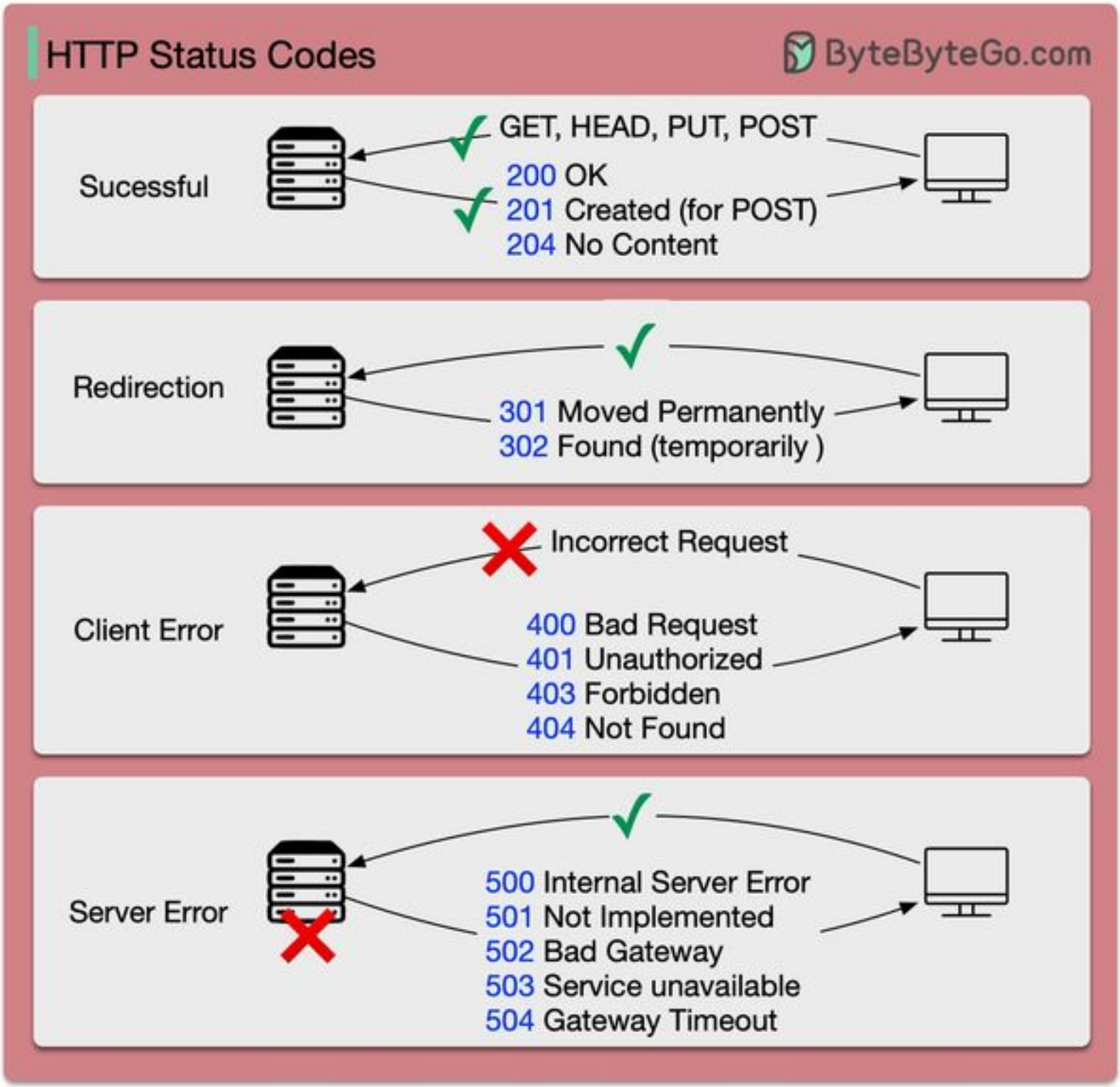
1. Registration & Link to Bank Account



How to Learn SQL?

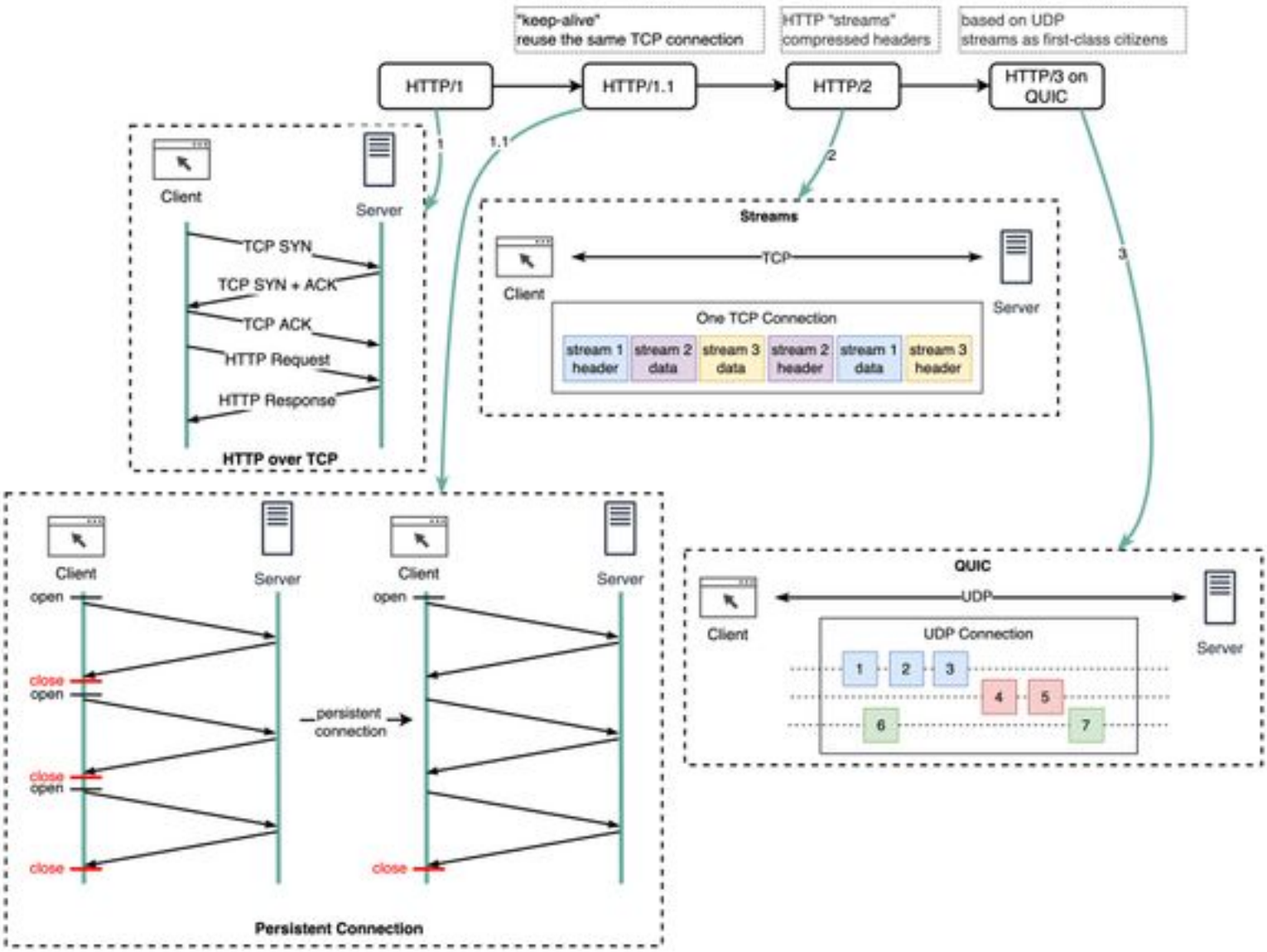
ByteByteGo.com



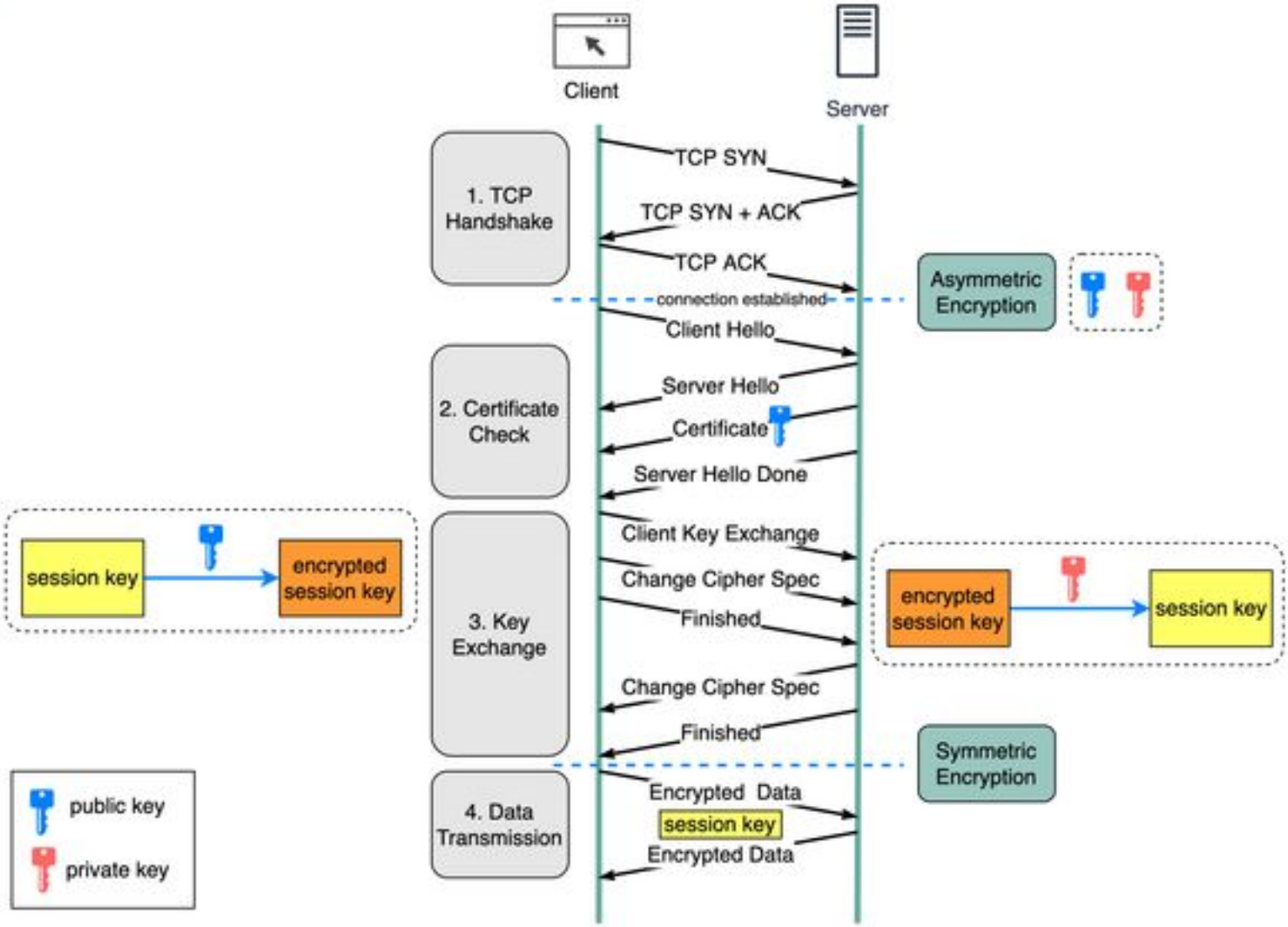

HTTP Status Codes
ByteByteGo.com
Sucessful
GET, HEAD, PUT, POST
200 OK
201 Created (for POST)
204 No Content
Redirection
301 Moved Permanently
302 Found (temporarily)
Client Error
Incorrect Request
400 Bad Request
401 Unauthorized
403 Forbidden
404 Not Found
Server Error
500 Internal Server Error
501 Not Implemented
502 Bad Gateway
503 Service unavailable
504 Gateway Timeout

How did we get to HTTP/3?

ByteByteGo



How does HTTPS Work?



```
{
  "squadName": "Super hero squad",
  "hometown": "Metro City",
  "formed": 2016,
  "secretBase": "Super tower",
  "active": true,
  "members": [
    {
      "name": "Molecule Man",
      "age": 29,
      "secretIdentity": "Dan Jukes",
      "powers": [
        "Radiation resistance",
        "Turning tiny",
        "Radiation blast"
      ]
    },
    {
      "name": "Madame Uppercut",
      "age": 39,
      "secretIdentity": "Jane Wilson",
      "powers": [
        "Million tonne punch",
        "Damage resistance",
        "Superhuman reflexes"
      ]
    },
    {
      "name": "Eternal Flame",
      "age": 1000000,
      "secretIdentity": "Unknown",
      "powers": [
        "Immortality",
        "Heat Immunity",
        "Inferno",
        "Teleportation",
        "Interdimensional travel"
      ]
    }
  ]
}
```

JWT {JSON Web Token}

With Love By @Sec_zB

1 {"what": "JSON"}

* A file format to store data in Key: value format

```
{
  "Key1" : "Value1",
  "Key2" : ["val1", "val2"],
  "Key3" : {
      Nested Json,
      [Json, Json]
  }
}
```

can be string

Key has to be string

or list of string or Json

another Json

or list of Jsons

Just a data structure :)

2 JWT Structure

3 parts

Header: `{ "alg" : "HS256", "type" : "JWT" }` → Base64 encode

data: `{ "key" : "foo" }` → Base64 encode

Signature → Base64 encode

X • Y • Z

Dots are just used for concatenation

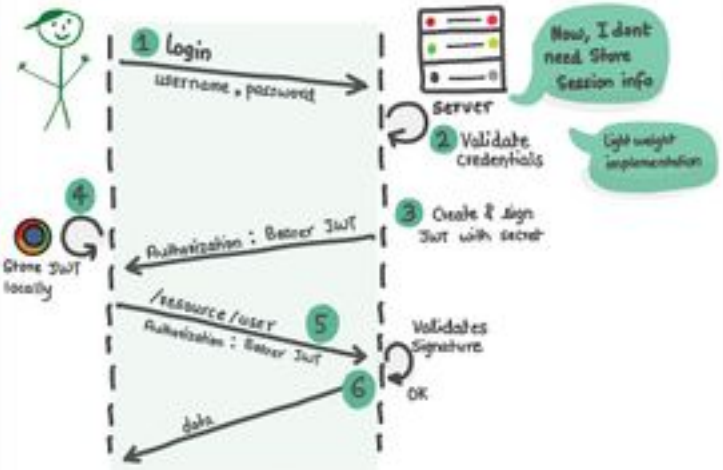
JWT = [abcdefg.F3E4GI] • [ZXJhZHUiOiA] • [IE3MDkxODMz] → X Y Z

X: I am just Encoded Header

Y: I am just Encoded Data. My keys are also called "CLAIMS". Two Reserved Keys are a few predefined claims

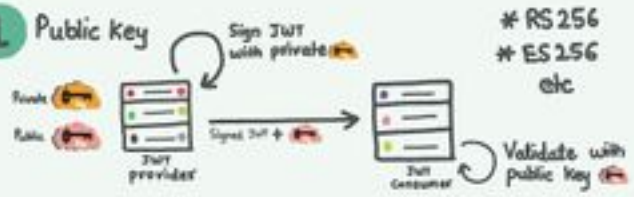
Z: I am Encoded Signature. Read about below

3 Working?



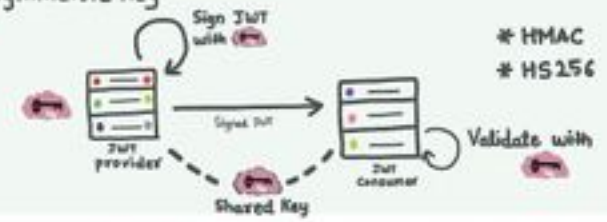
4 Signing Alg

1 Public Key



* RS256
* ES256
etc

2 Symmetric Key

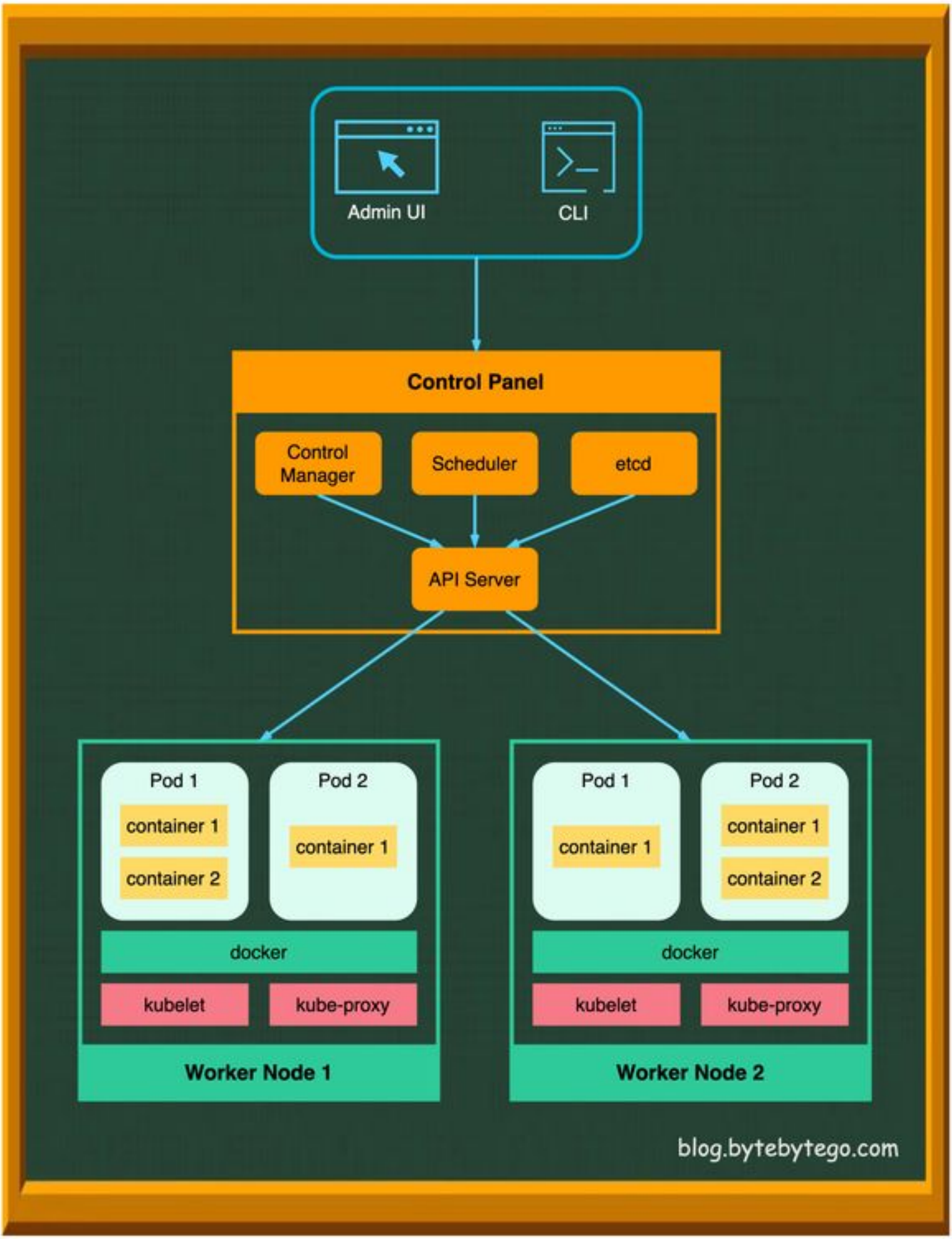


* HMAC
* HS256

SecurityZines.com In Collaboration with ByteByteGo

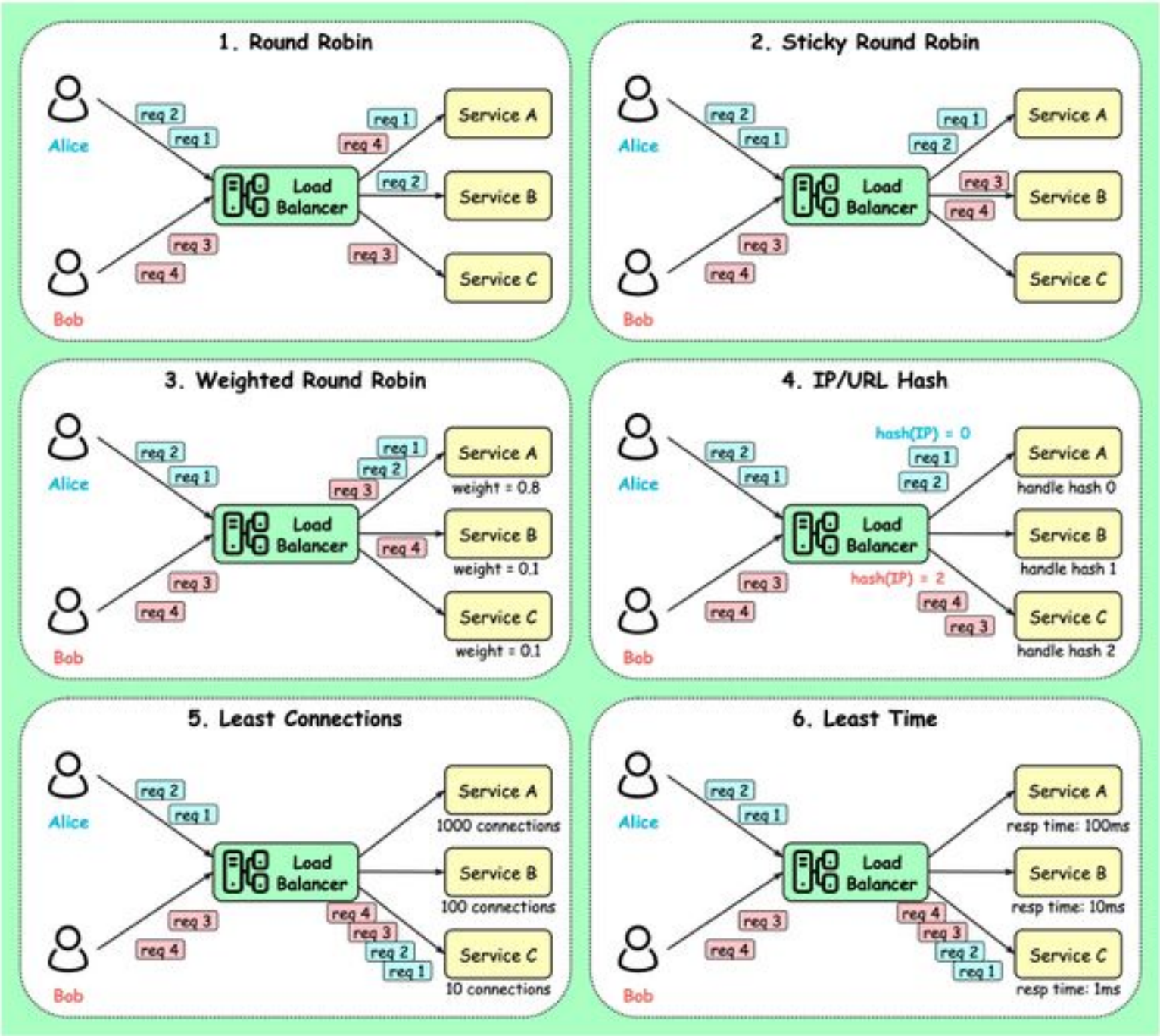
What is k8s?

blog.bytebytego.com



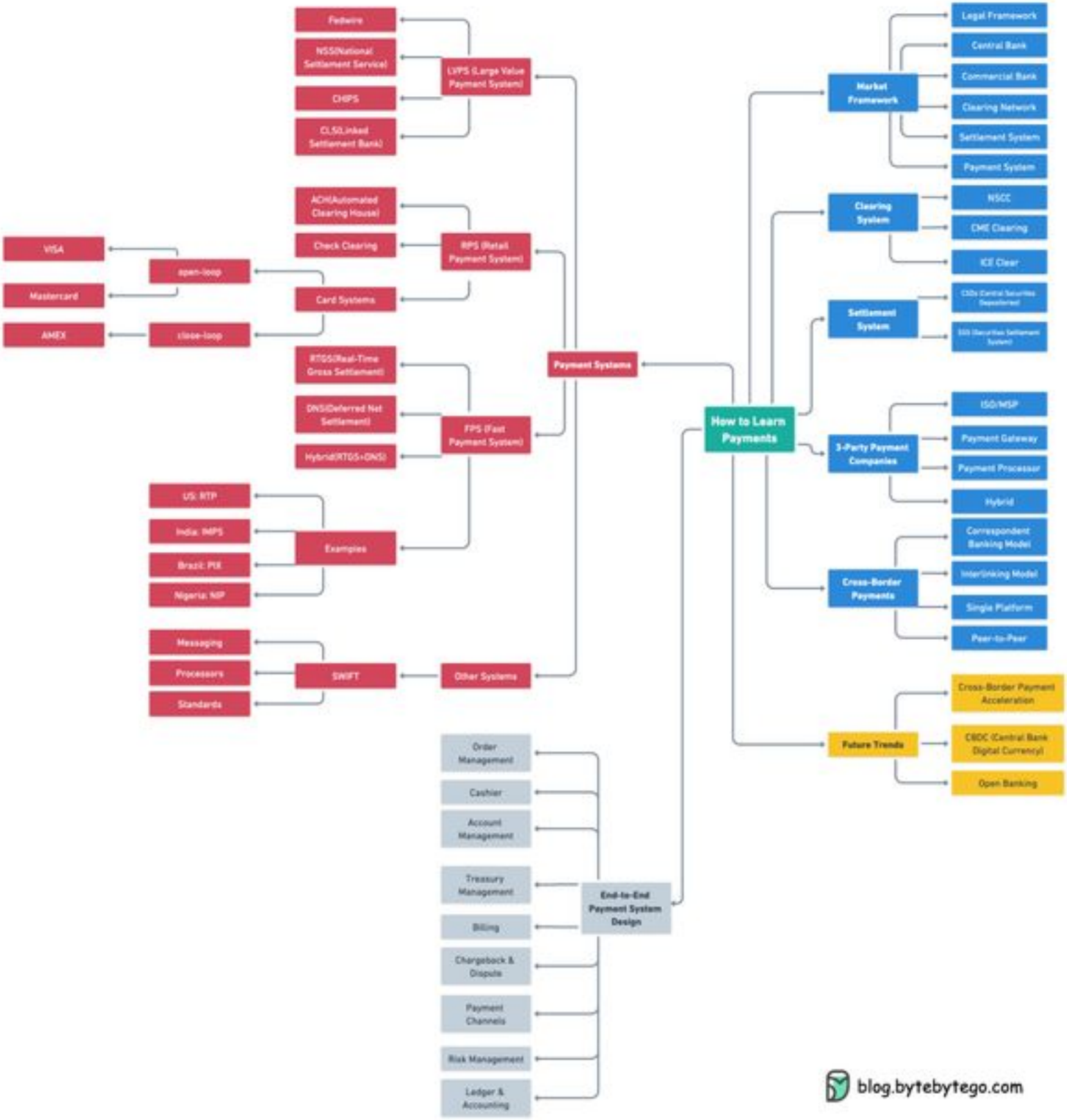
Load Balancing Algorithms

blog.bytebytego.com



How to Learn Payments?

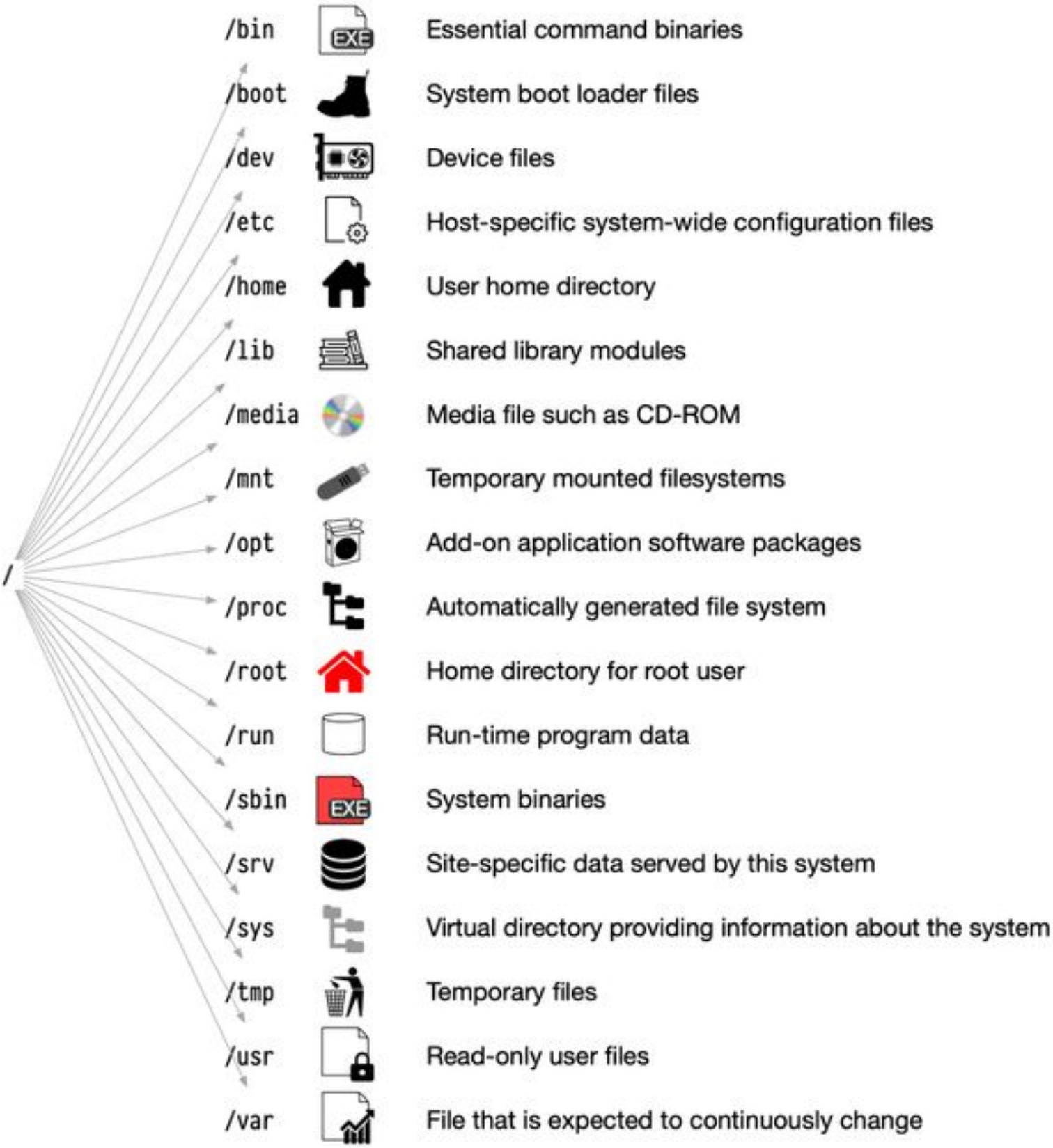
blog.bytebytego.com



blog.bytebytego.com

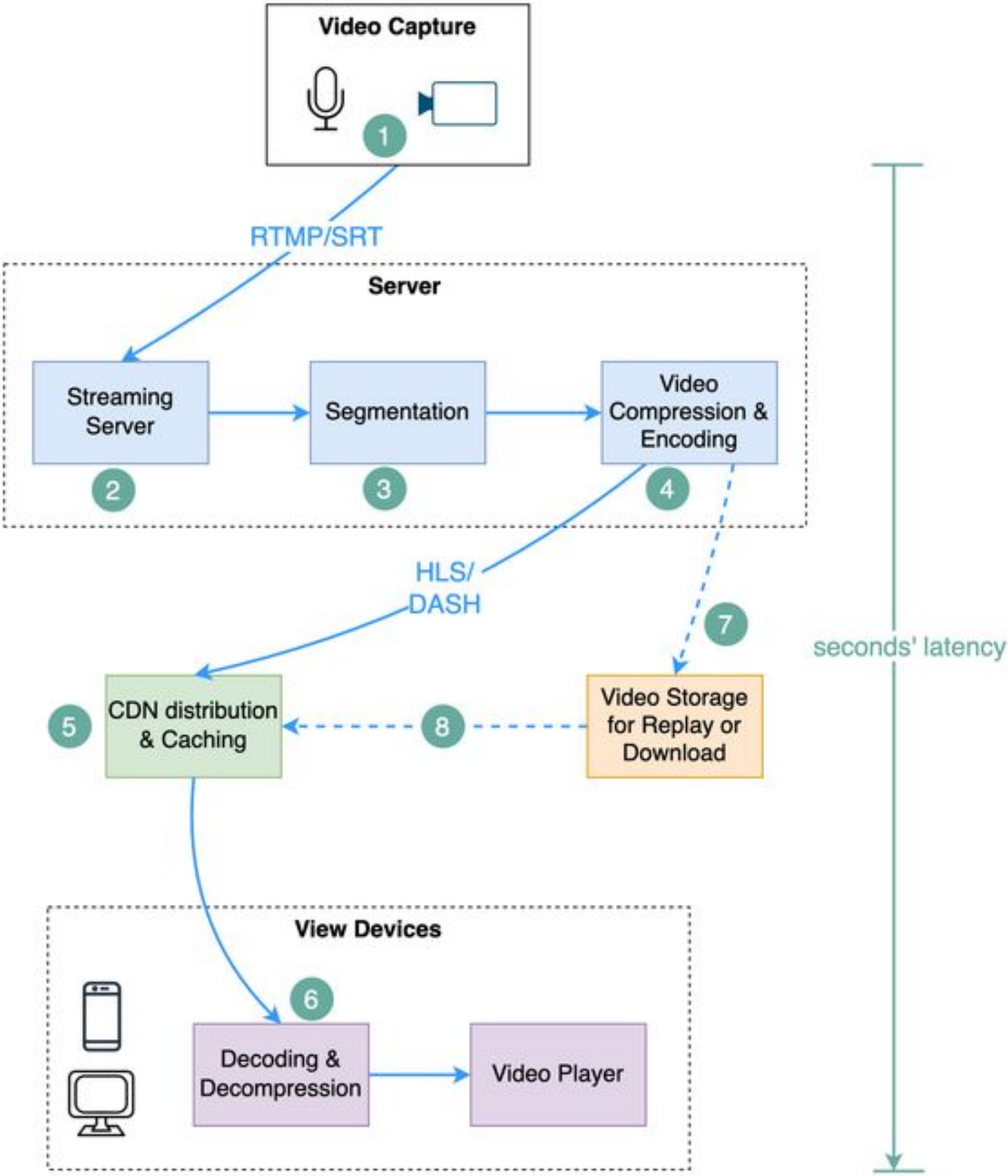
Linux File Systems

ByteByteGo.com



How does Live Streaming Work?

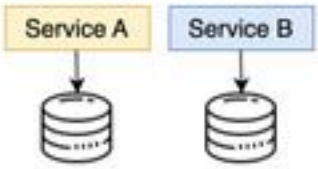
blog.bytebytego.com



Microservice Best Practices

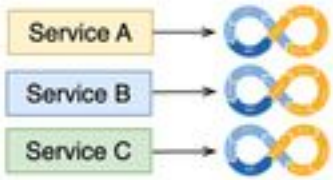
blog.bytebytego.com

1 Separate data store

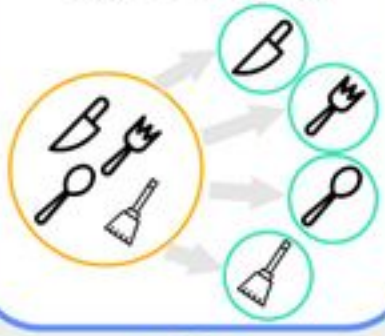


2 Keep code at a similar level of maturity

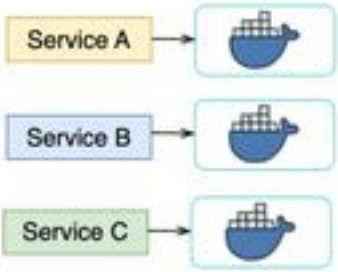
3 Separate build for each microservice



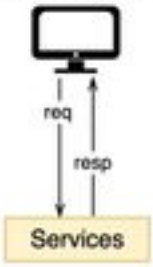
4 Single responsibility



5 Deploy into containers

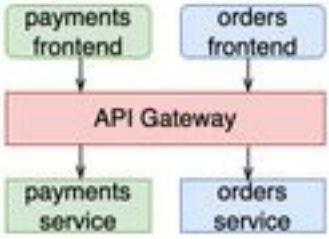


6 Treat servers as stateless



7 Domain-driven design

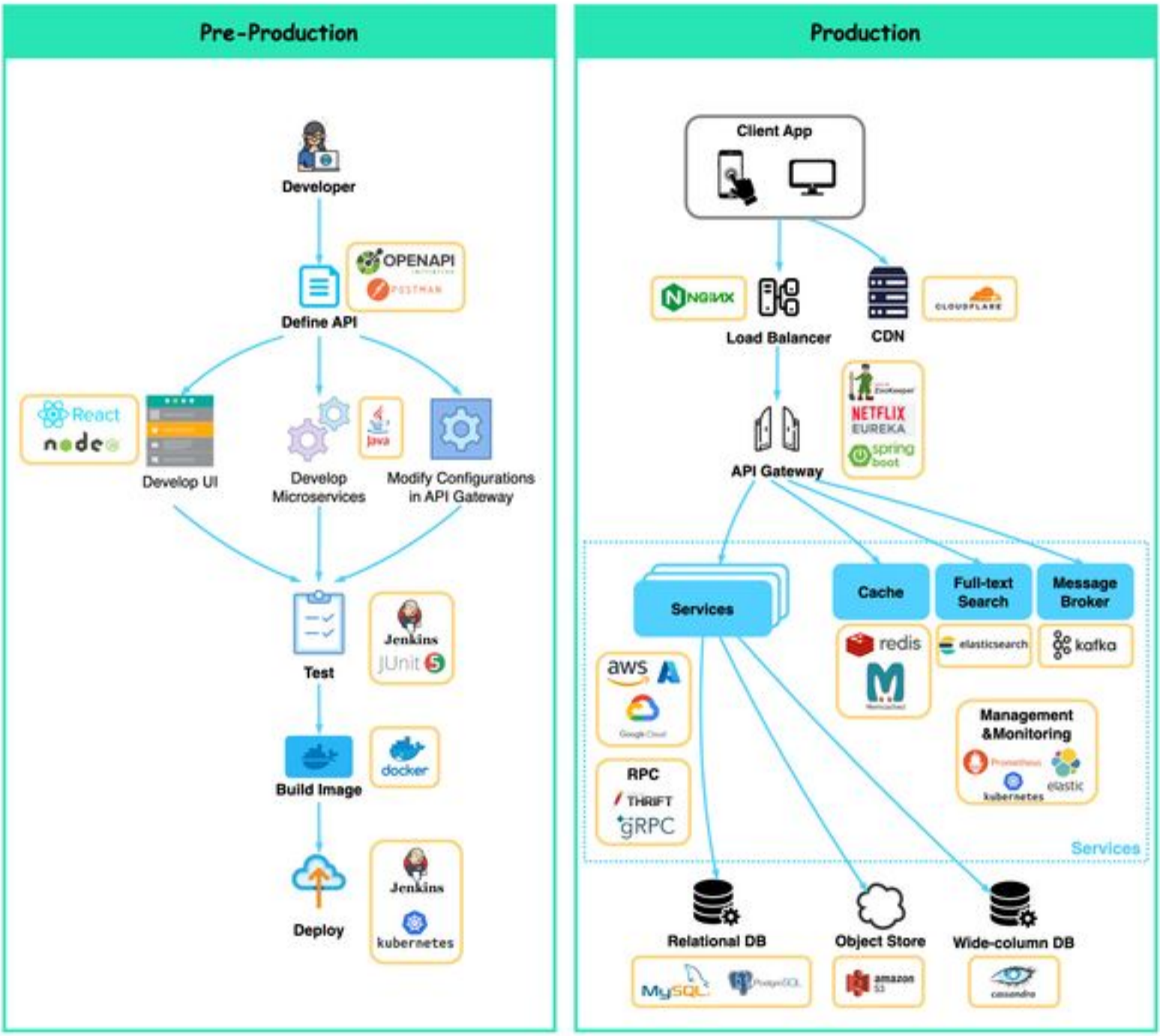
8 Micro frontend



9 Orchestrating microservices

Microservice Tech Stack

blog.bytebytego.com



Monorepo vs Microrepo: which is the best?

ByteByteGo.com

	Monorepo	Microrepo
Company	Google, Meta, Uber, airbnb, Linux, Windows	amazon, NETFLIX, LinkedIn, ORACLE, APACHE
Collaboration	Service / Owners Service A Service B Common Services work under the same repository.	Service A Service B Common Service owners work under **separate** repositories.
Dependency	Service A, Service B, Common Dependency A = 1.1; Service share the same dependency.	Service A, A = 1.0; Service B, A = 1.1; Common, A = 1.2; Service choose their own dependency.
Scalability	Stadndards Service A, Service B, Service D, Common Services share the same standard.	Stadndards Service D, Service A, Service B, Common Services set their own standard.
Tooling	Bazel, Buck, Nix, Lerna	Gradle, nebula, Maven, npm, CMake

NETFLIX Tech Stack

DevOps

CI CD

JIRA Confluence Jenkins Spinnaker N. Atlas N. Chaos Monkey Gradle nebula

Mobile

Kotlin

Swift

Frontend

React

JS

GraphQL

Backend

Services

spring boot NETFLIX ZUUL NETFLIX EUREKA

Database

MySQL cassandra CockroachDB EVCache

Messaging / Streaming

kafka A distributed streaming platform

Apache Flink

Streaming

Open Connect AWS CloudFront AWS S3

AWS Elastic Transcoder

ByteByteGo

BIG DATA

Data Storage

AWS S3 Redshift Apache Iceberg druid

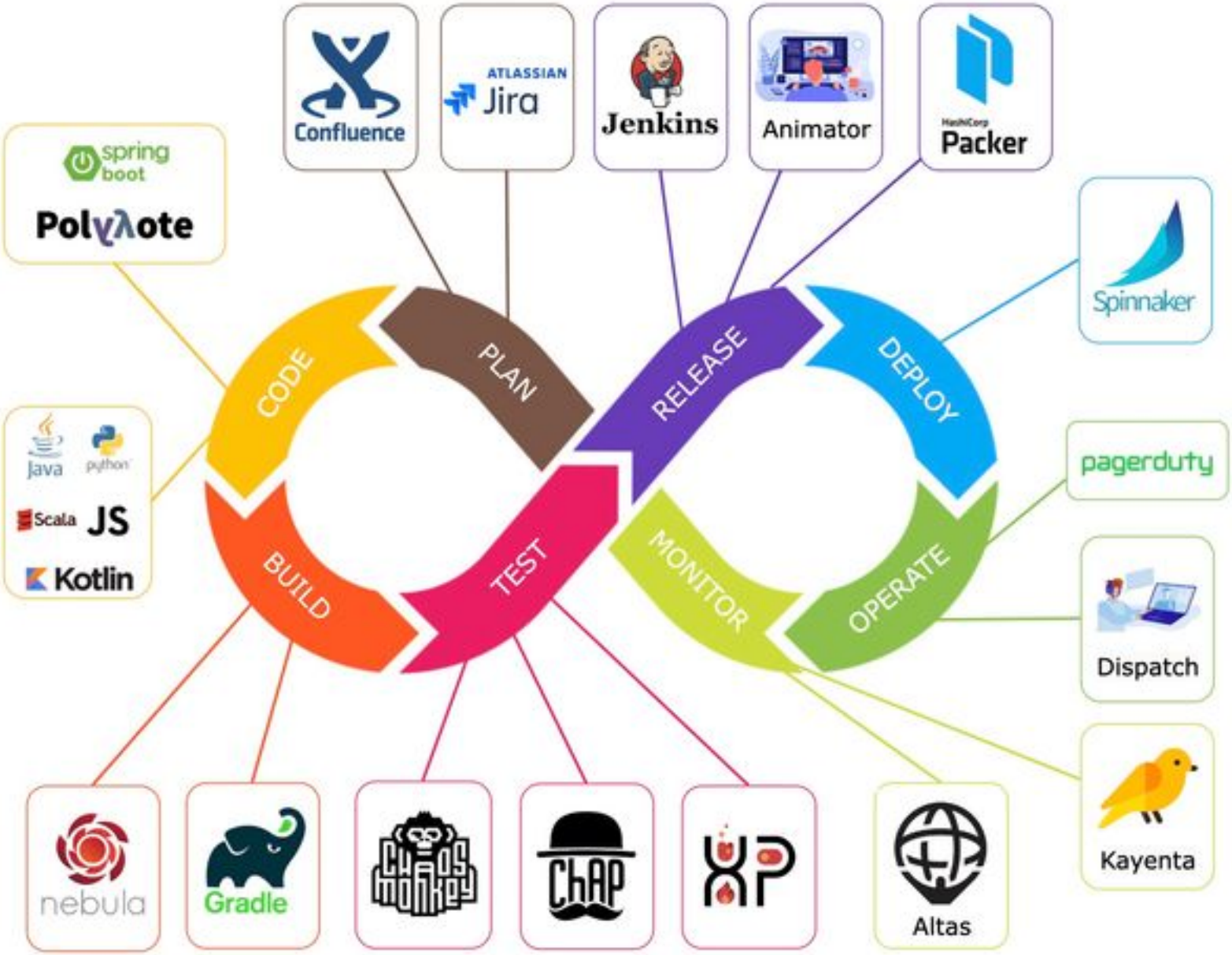
Data Processing

tableau A. Flink APACHE Spark

01101001101001101001101001101001 01

NETFLIX Tech Stack (CI/CD Pipeline)

blog.bytebytego.com



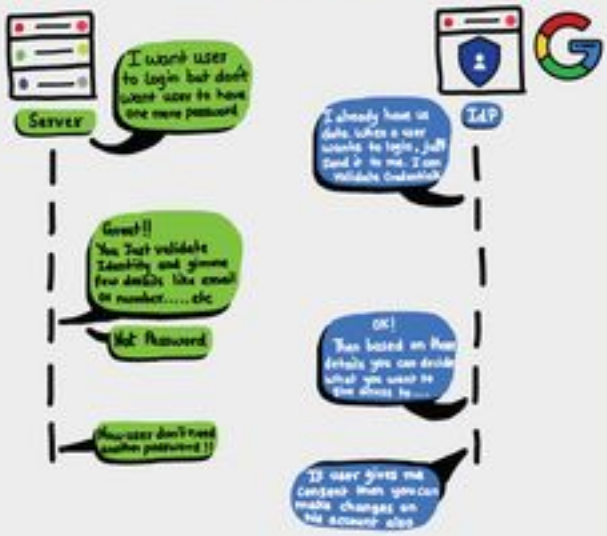
What is OAuth?

* Protocol for sharing user Authorization across systems.

Involves 3 entities

1. User — Entity who's authorization flows across the systems. Also known as resource owner
2. Server — Service, User wants to access and complete Authorization across.
3. IdP — Identity Provider which stores User's identity and validate User's credentials & Share user Authorization with other services.

OAuth



1.0 protocal designed for web browser only

2.0 protocal upgraded for
- browser App
- non browser App
- windows App
- mobile App
- APIs

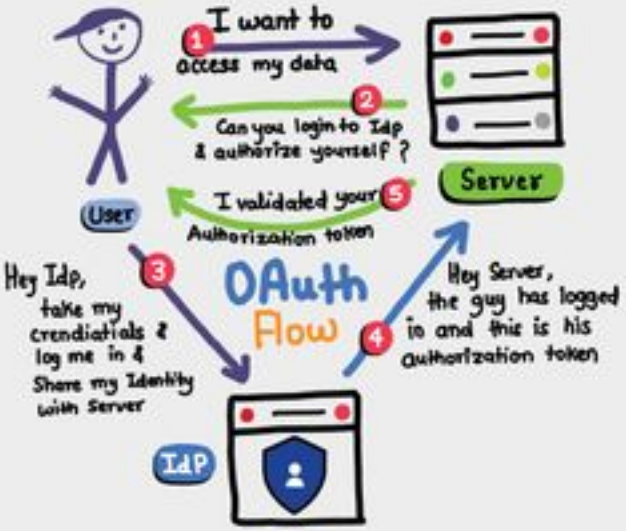
OAUTH 2.0

@Sec_x0

ByteByteGo

Open Authorization

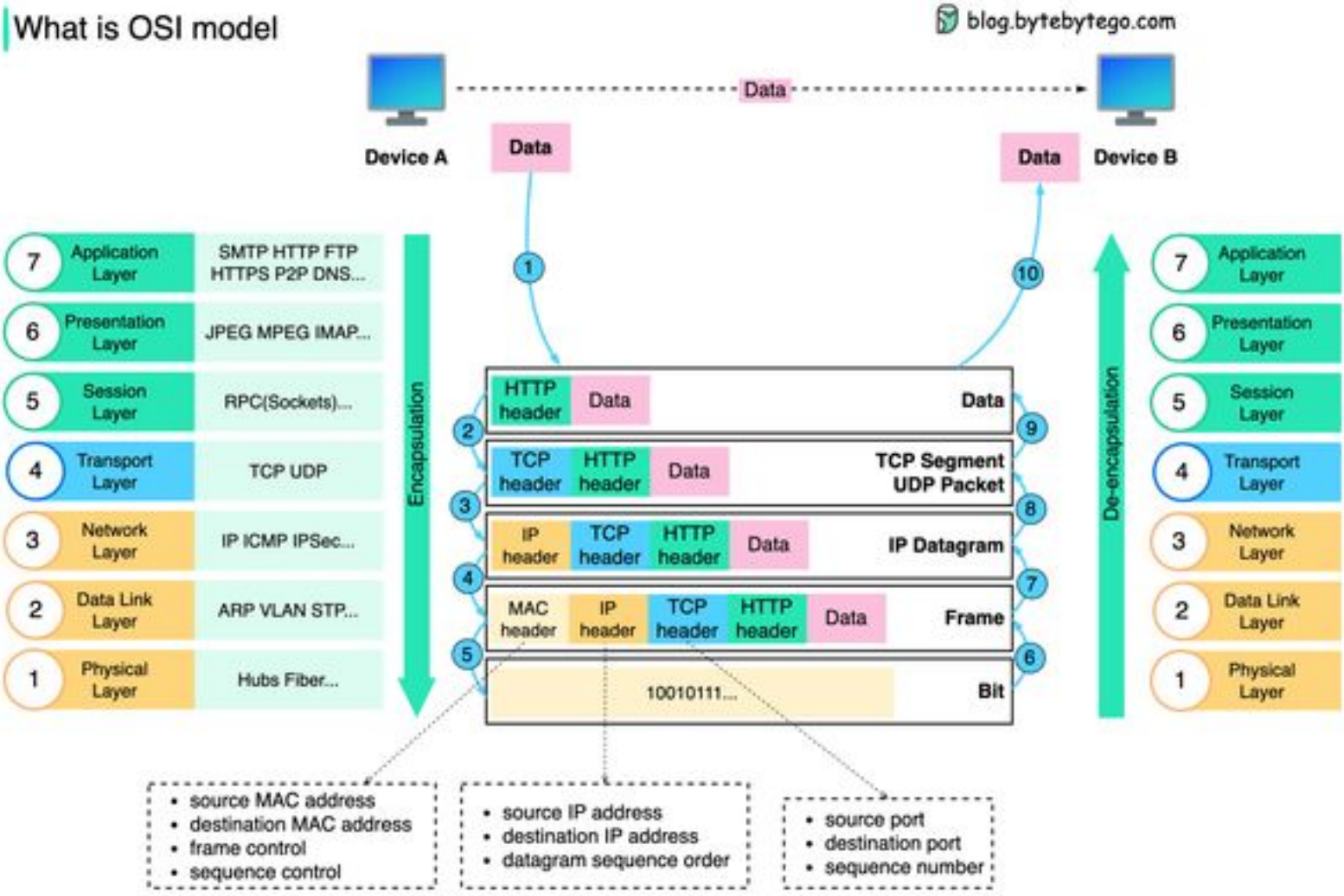
OAuth Flow



Authorization Code Flow

4 Types of OAuth Flows

1. Authorization code
2. Client Credentials
3. Implicit Code
4. Resourse owner Password


What is OSI model
blog.bytebytego.com
Data
Device A
Data
Data
Device B
7 Application Layer
SMTP HTTP FTP HTTPS P2P DNS...
6 Presentation Layer
JPEG MPEG IMAP...
5 Session Layer
RPC(Sockets)...
4 Transport Layer
TCP UDP
3 Network Layer
IP ICMP IPSec...
2 Data Link Layer
ARP VLAN STP...
1 Physical Layer
Hubs Fiber...
Encapsulation
De-encapsulation
HTTP header
Data
Data
TCP header
HTTP header
Data
TCP Segment
UDP Packet
IP header
TCP header
HTTP header
Data
IP Datagram
MAC header
IP header
TCP header
HTTP header
Data
Frame
10010111...
Bit
source MAC address
destination MAC address
frame control
sequence control
source IP address
destination IP address
datagram sequence order
source port
destination port
sequence number

Design Effective & Safe APIs

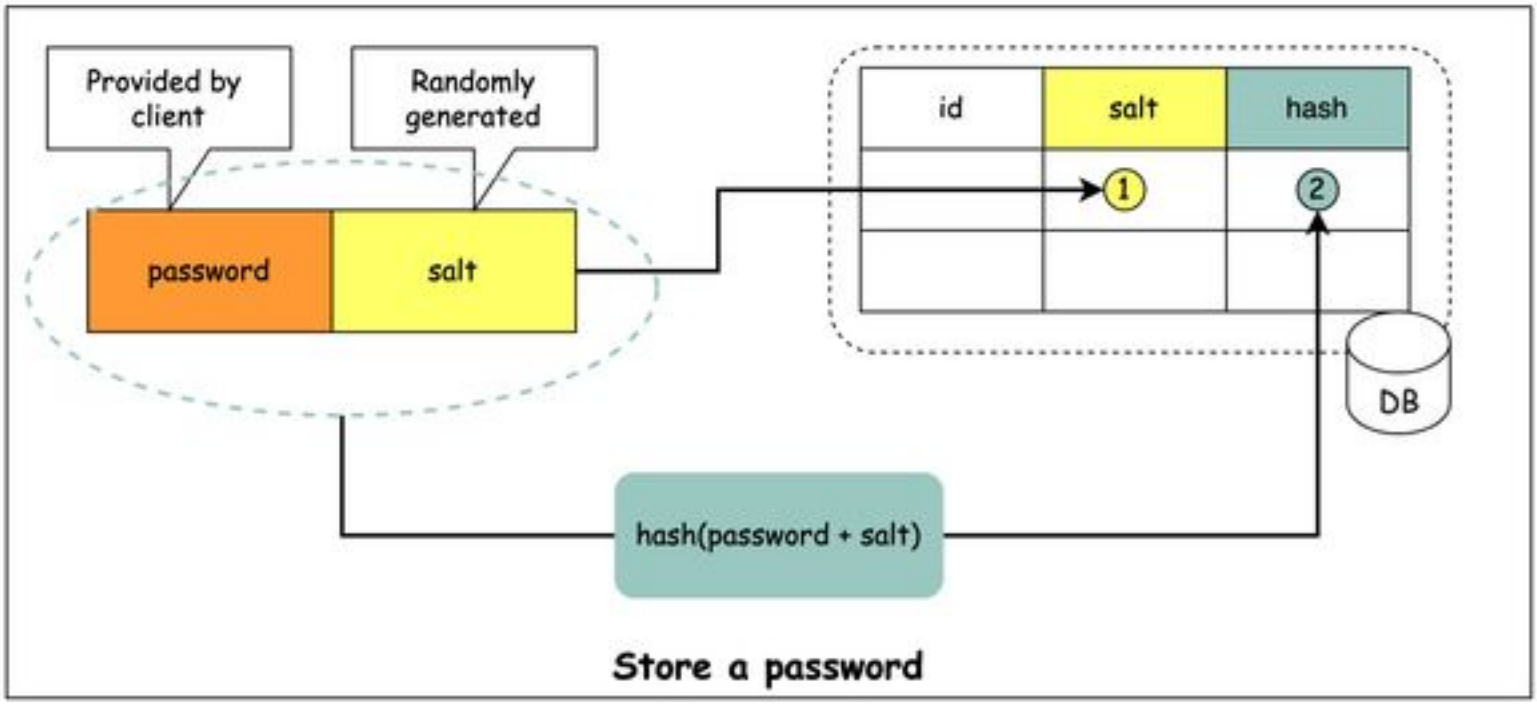
blog.bytebytego.com

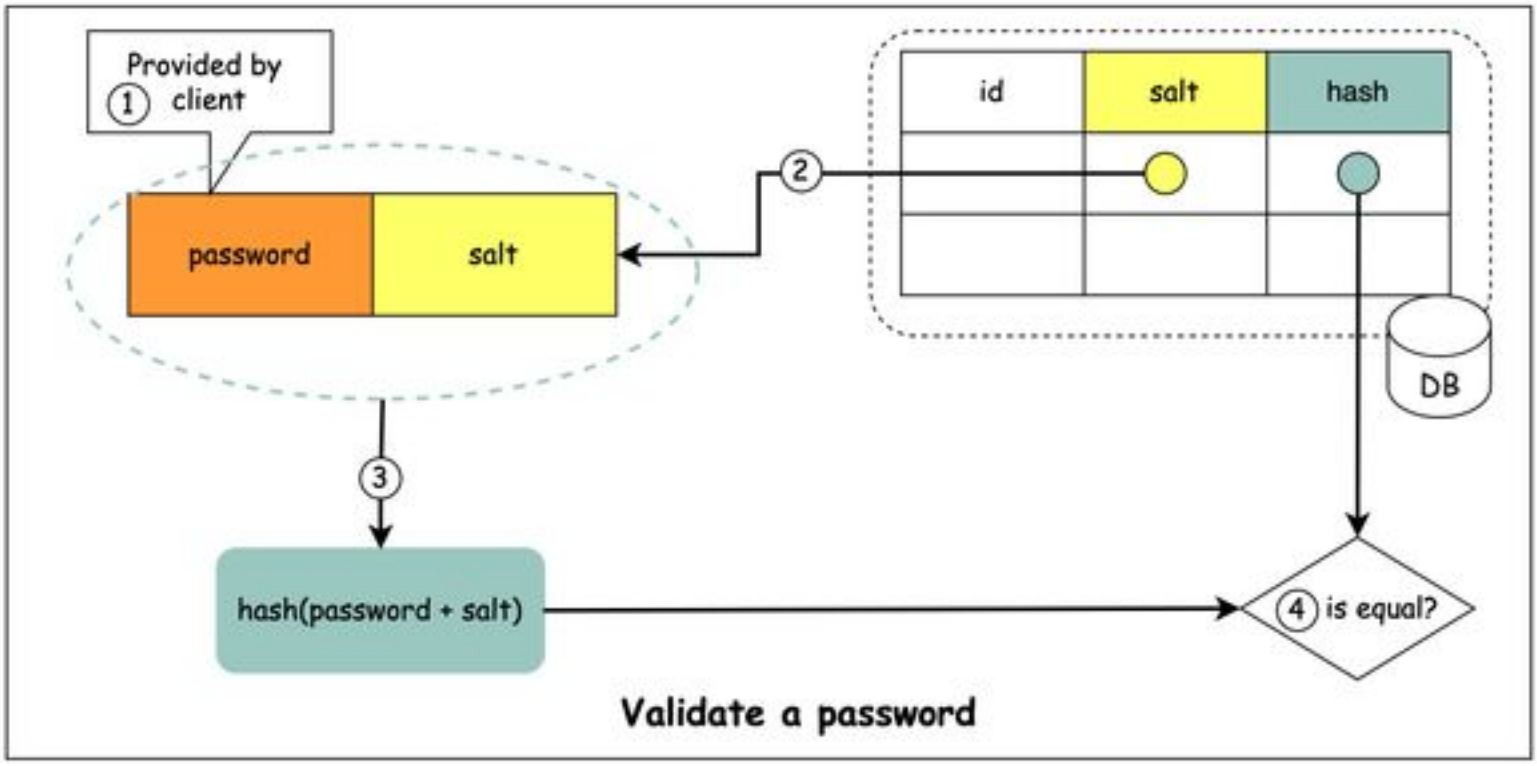
Design a Shopping Cart

Use resource names (nouns)	GET /querycarts/123	GET /carts/123
Use plurals	GET /cart/123	GET /carts/123
Idempotency	POST /carts	POST /carts {requestId: 4321}
Use versioning	GET /carts/v1/123	GET /v1/carts/123
Query after soft deletion	GET /carts	GET /carts? includeDeleted=true
Pagination	GET /carts	GET /carts? pageSize=xx&pageToken=xx
Sorting	GET /items	GET /items? sort_by=time
Filtering	GET /items	GET /items? filter=color:red
Secure Access	X-API-KEY=xxx	X-API-KEY = xxx X-EXPIRY = xxx X-REQUEST-SIGNATURE = xxx hmac(URL + QueryString + Expiry + Body)
Resource cross reference	GET /carts/123? item=321	GET /carts/123/items/321
Add an item to a cart	POST /carts/123? addItem=321	POST /carts/123/items:add { itemId: "items/321" }
Rate limit	No rate limit - DDos	Design rate limiting rules based on IP, user, action group etc

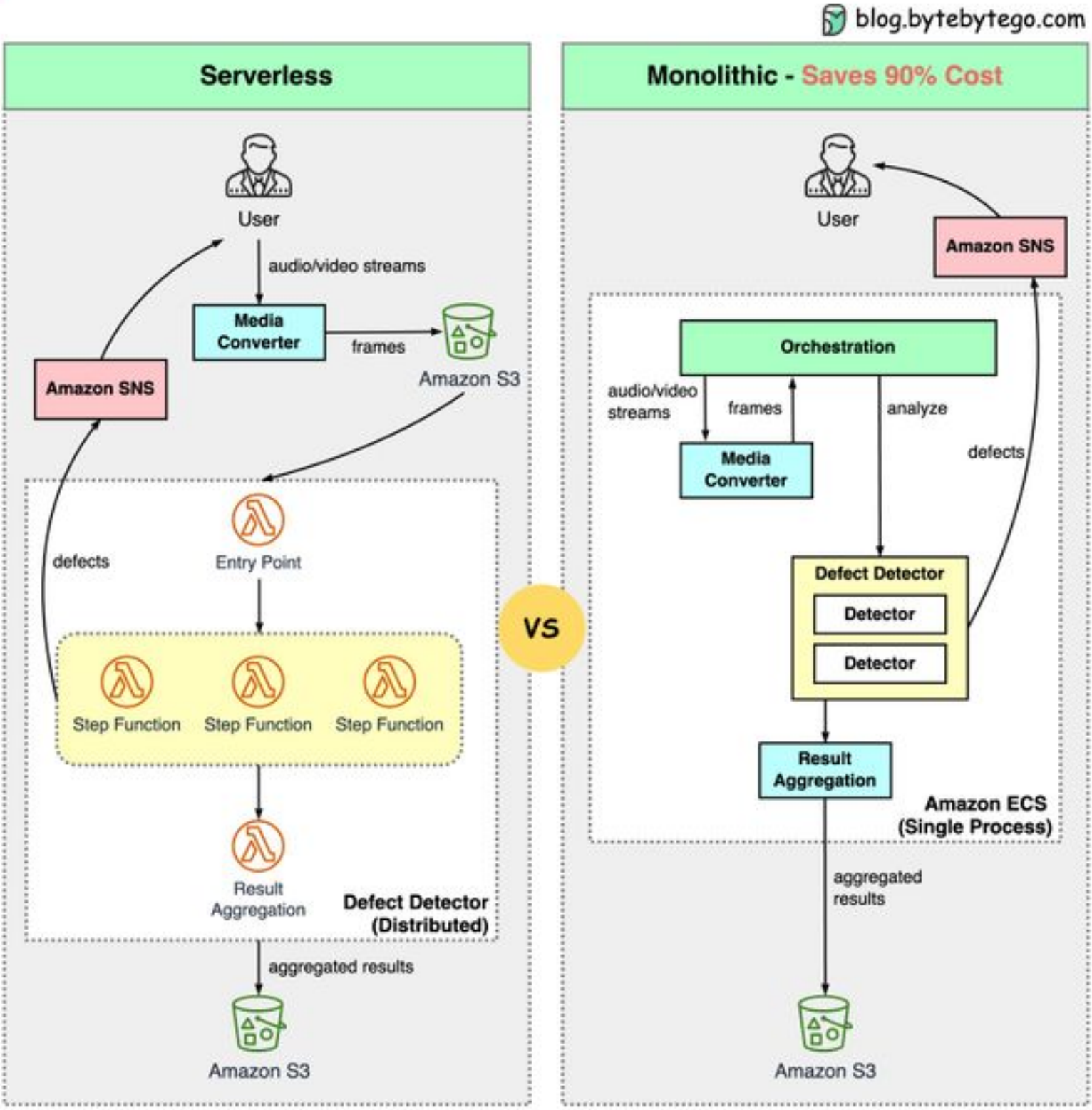
How to store passwords in DB?

blog.bytebytego.com

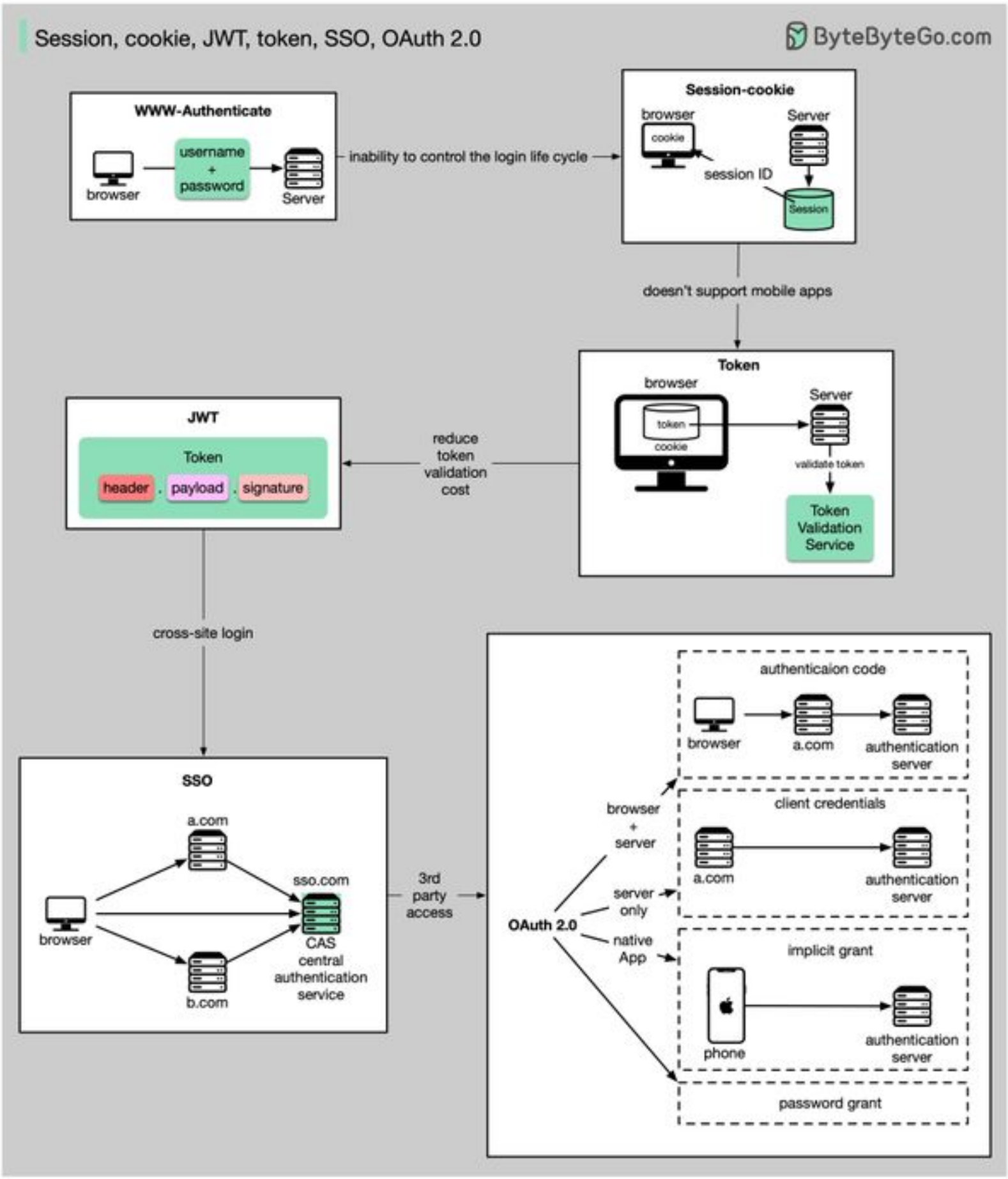




Amazon Prime Video monitoring - From Serverless to Monolithic

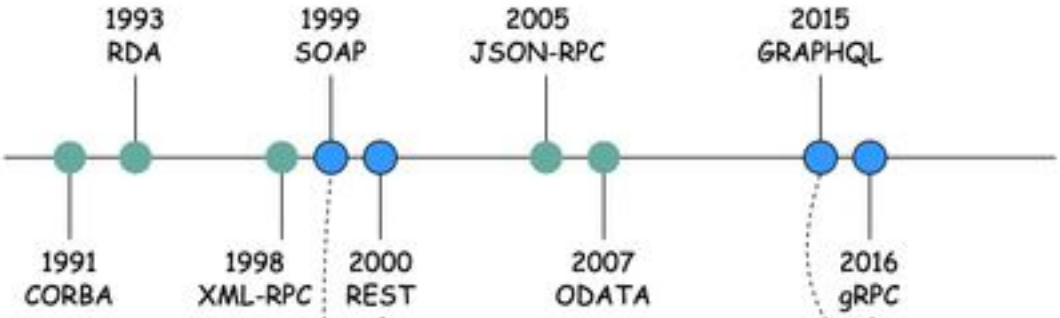


Based on: https://primevideotech.com/


Session, cookie, JWT, token, SSO, OAuth 2.0
ByteByteGo.com
WWW-Authenticate
browser
username + password
Server
inability to control the login life cycle
Session-cookie
browser
cookie
Server
session ID
Session
doesn't support mobile apps
Token
browser
token
cookie
Server
validate token
Token Validation Service
reduce token validation cost
JWT
Token
header . payload . signature
cross-site login
SSO
a.com
sso.com
browser
CAS central authentication service
b.com
3rd party access
OAuth 2.0
browser + server
server only
native App
authenticaion code
browser
a.com
authentication server
client credentials
a.com
authentication server
implicit grant
phone
authentication server
password grant

API Architectural Styles Comparison

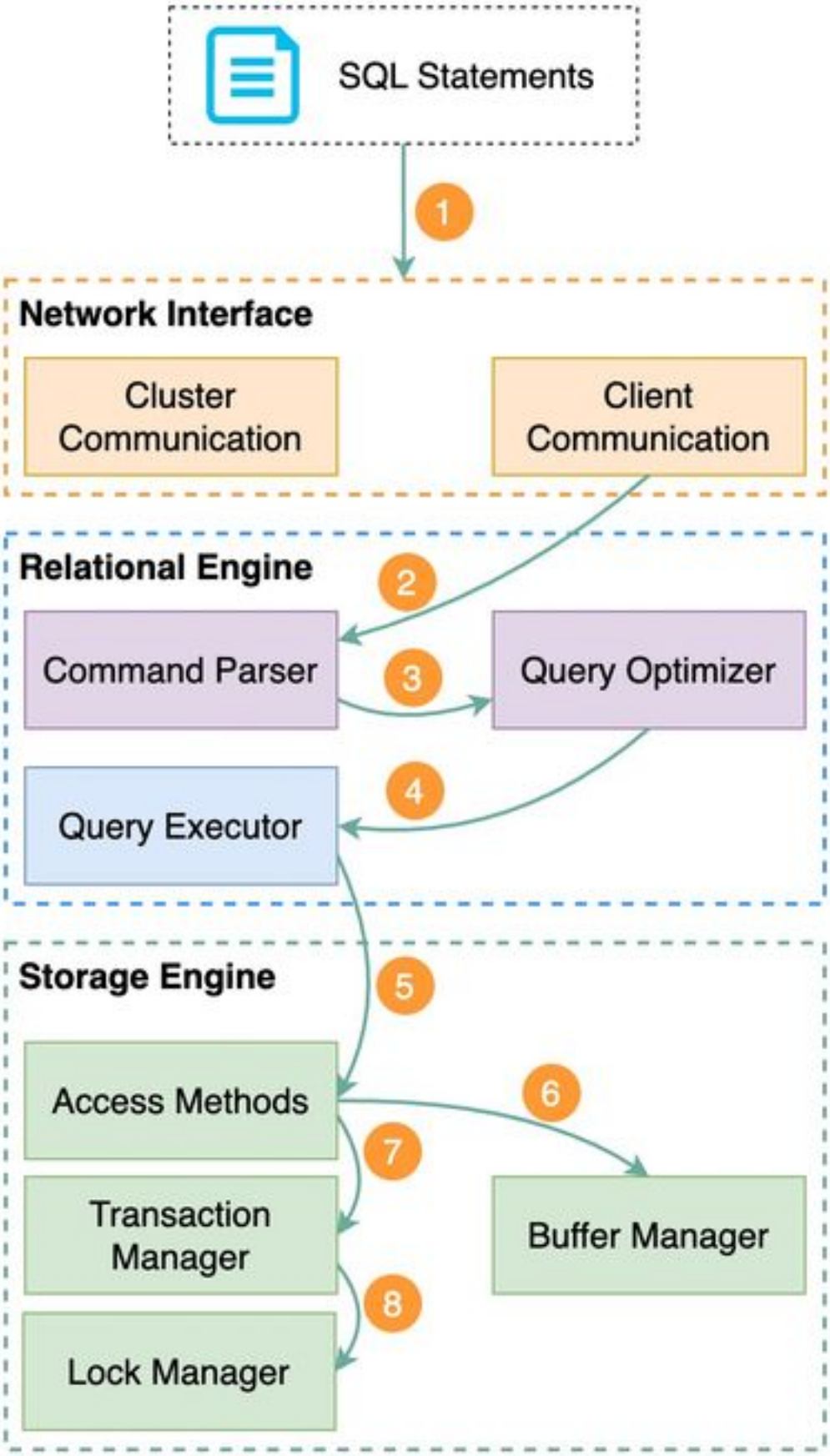
Source: altexsoft



	SOAP (Simple Object Access Protocol)	REST (REpresentational State Transfer)	GraphQL	RPC (Remote Procedure Call)
Organized in terms of	enveloped message structure	compliance with six architectural constraints	schema & type system	local procedure call
Format	XML only	XML, JSON, HTML, plain text	JSON	JSON, XML, Protobuf, Thrift, FlatBuffers
Learning curve	Difficult	Easy	Medium	Easy
Community	Small	Large	Growing	Large
Use cases	- payment gateways - identity management - CRM solutions - financial and telecommunication services - legacy system support	- public APIs - simple resource-driven apps	- mobile APIs - complex systems - micro-services	- command and action-oriented APIs - high performance communication in massive micro-services systems

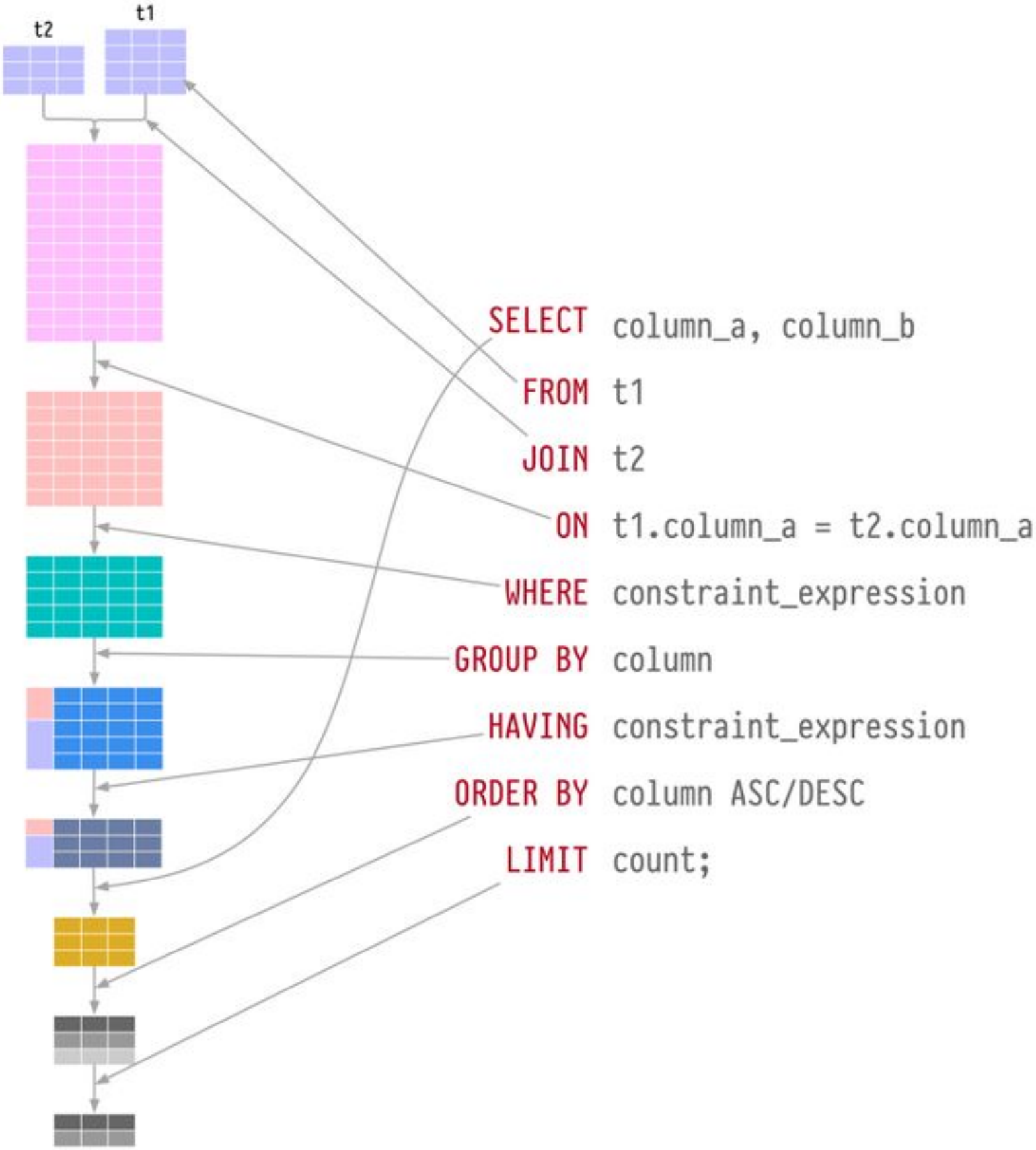
How is SQL Executed in DB?

blog.bytebytego.com



SQL Query Execution Order

ByteByteGo.com

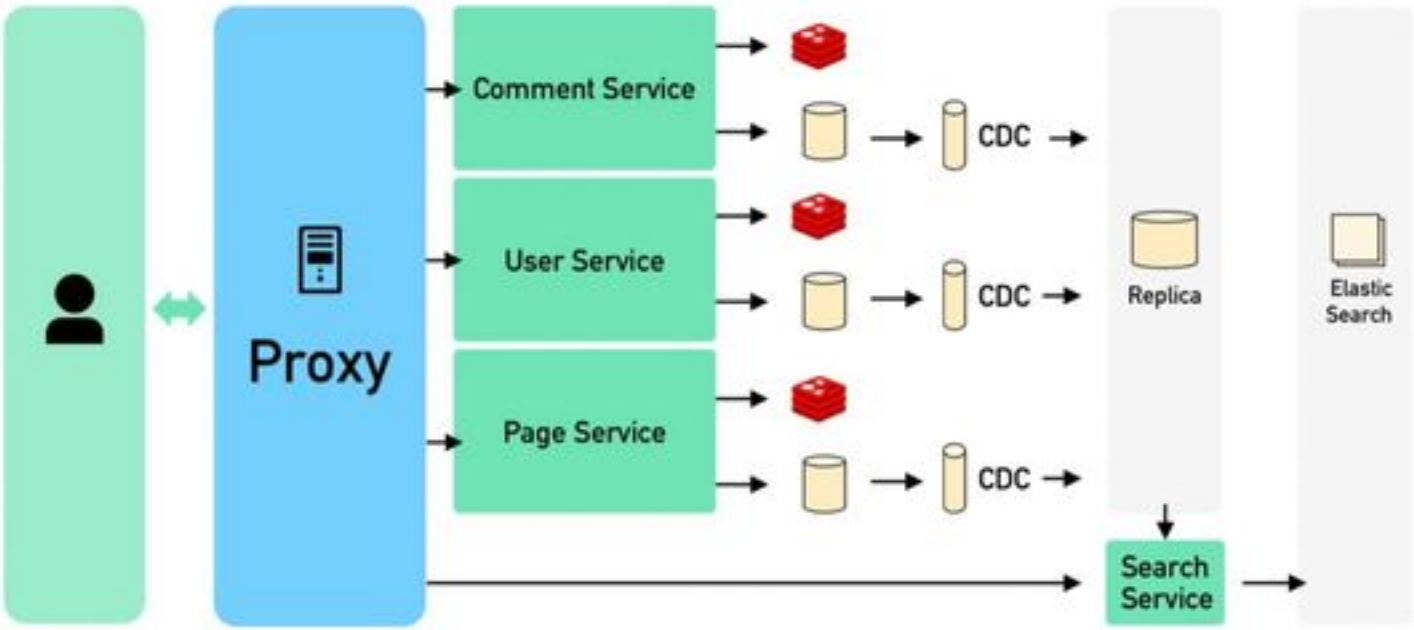


stack overflow Architecture

ByteByteGo.com

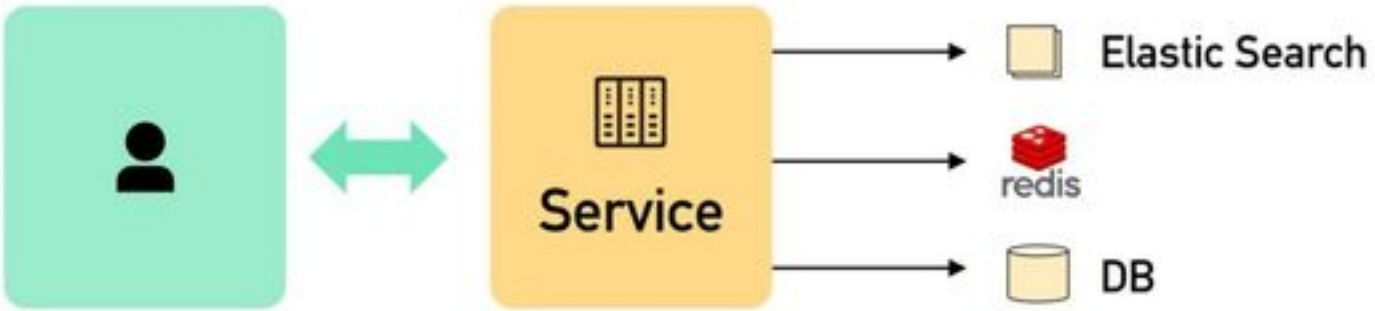
What people think it looks like

1. Microservice based
2. Event sourcing (CQRS)
3. Eventual consistency
4. Sharding
5. Heavy use cache
6. ...



What it actually is

1. Monolithic
2. Only 9 web servers



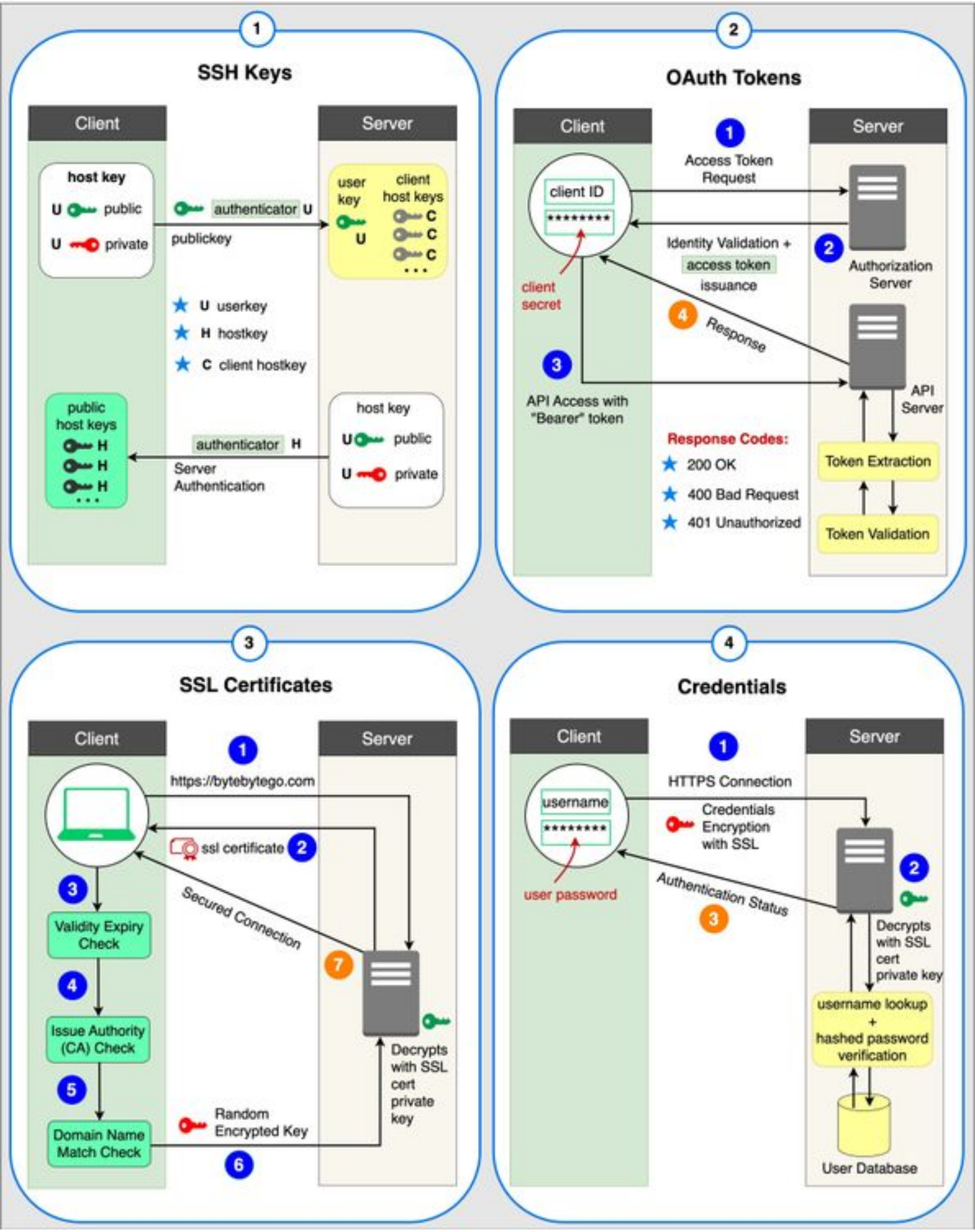
Top Redis Use Cases

blog.bytebytego.com

String	Session
	Cache
	Distributed Lock
Int	Counter
	Rate Limiter
	Global ID
Hash	Shopping Cart
Bitmap	User Retention
List	Message Queue
ZSet	Rank/Leaderboard

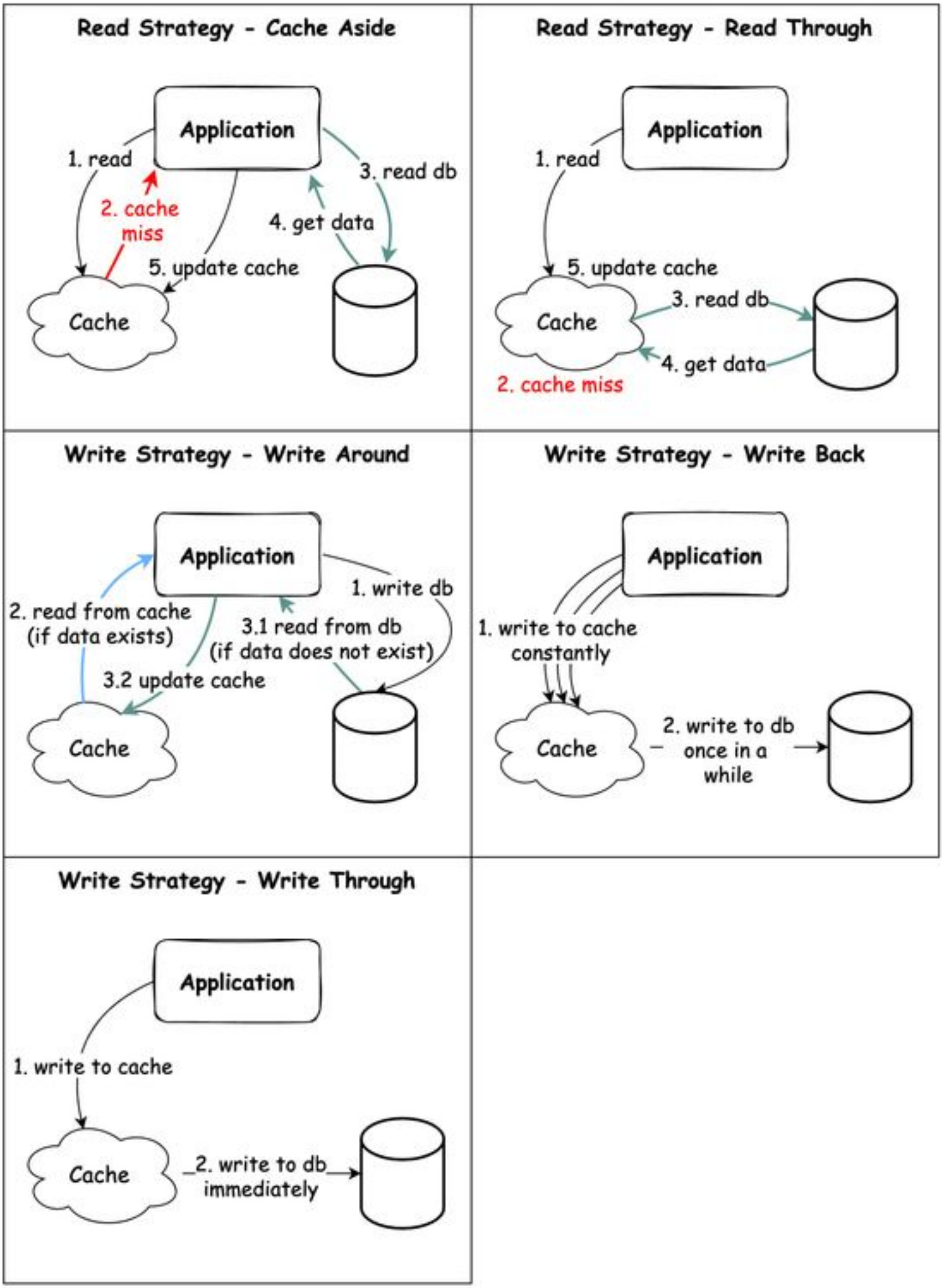
Top 4 Most Used Authentication Mechanisms

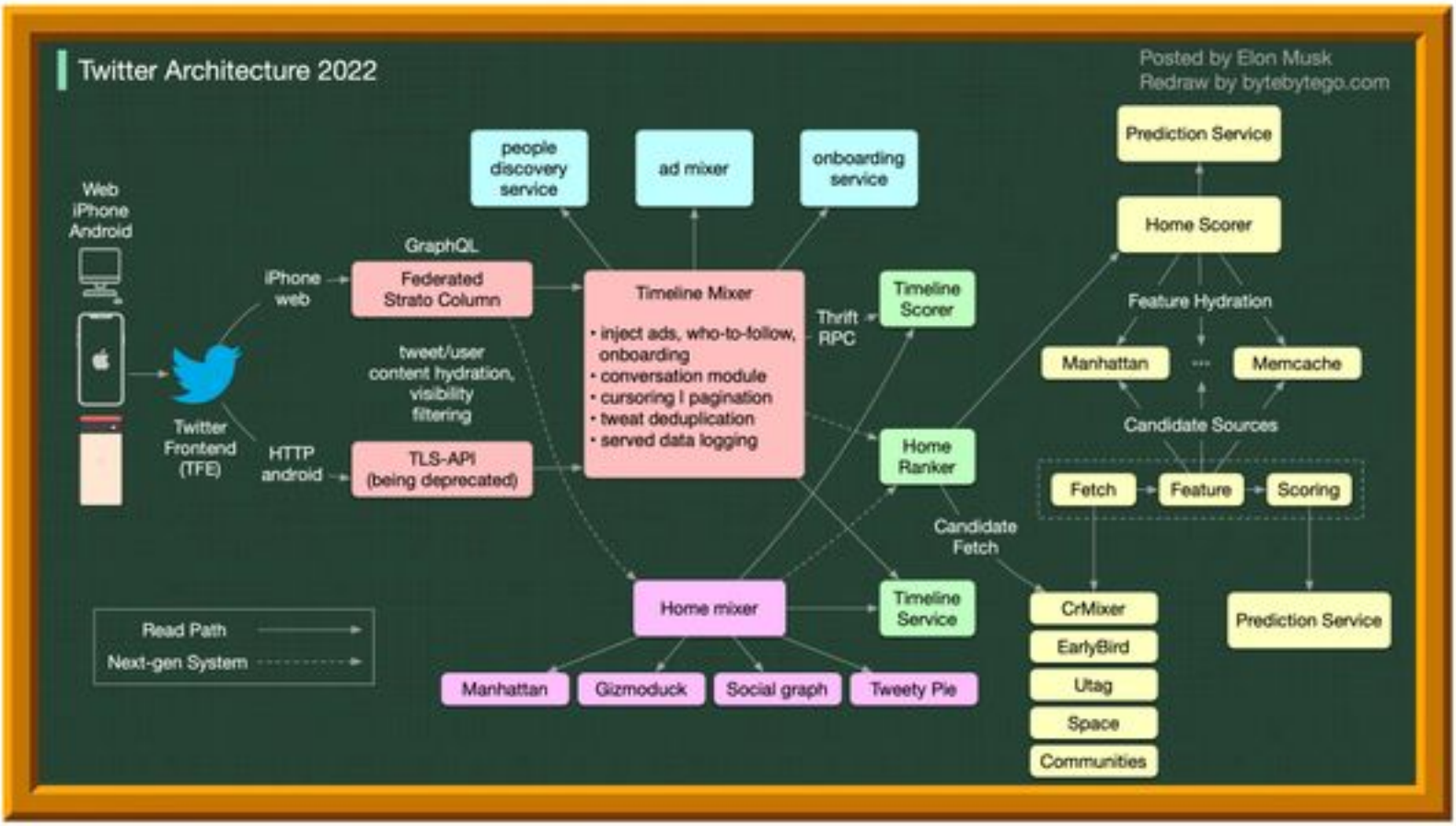
blog.bytebytego.com



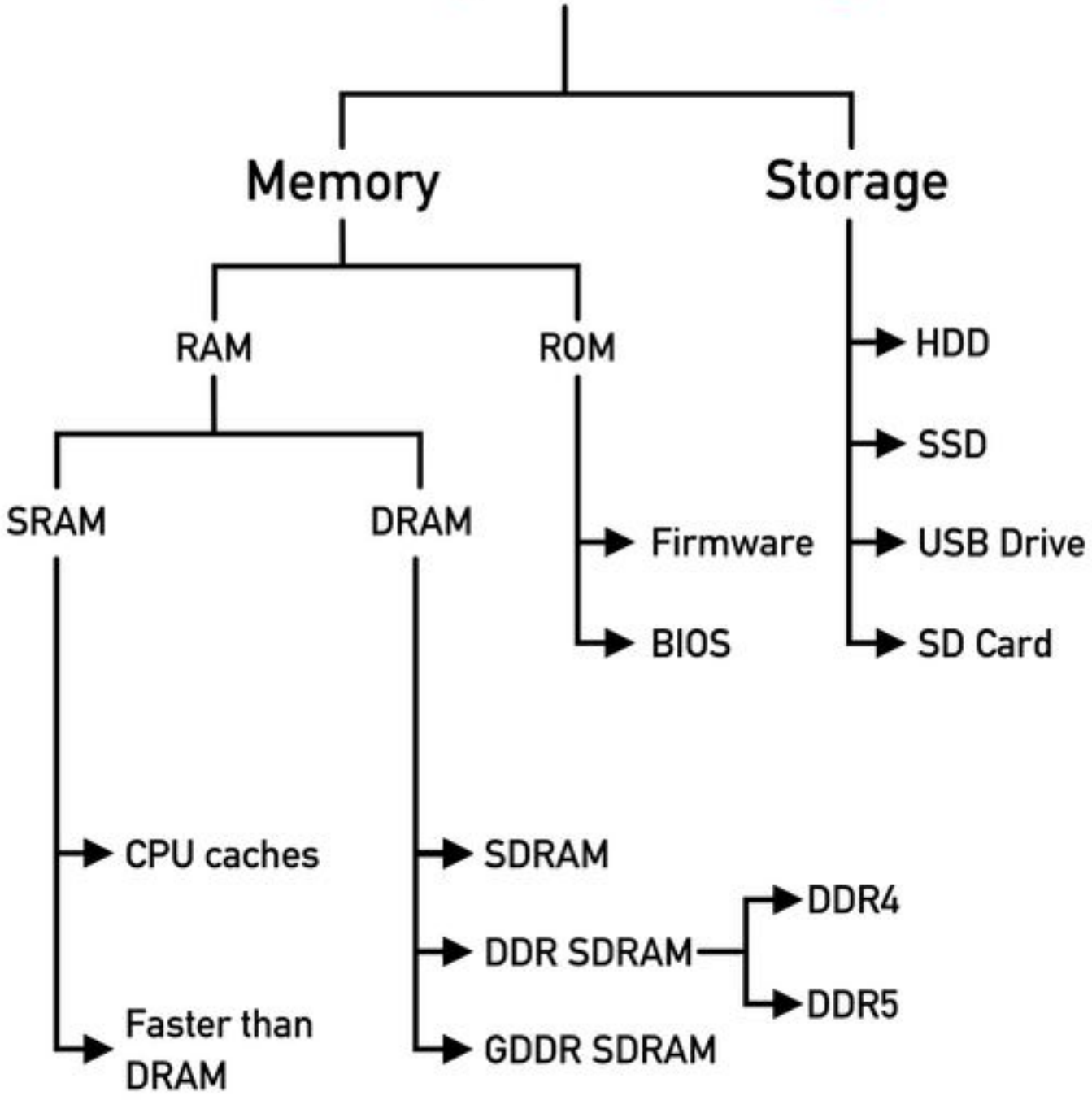
Top caching strategies

ByteByteGo




Twitter Architecture 2022
Posted by Elon Musk
Redraw by bytebytego.com
Web
iPhone
Android
Twitter
Frontend
(TFE)
iPhone
web
HTTP
android
GraphQL
Federated
Strato Column
TLS-API
(being deprecated)
tweet/user
content hydration,
visibility
filtering
people
discovery
service
ad mixer
onboarding
service
Timeline Mixer
• inject ads, who-to-follow, onboarding
• conversation module
• cursoring I pagination
• tweat deduplication
• served data logging
Thrift
RPC
Timeline
Scorer
Home
Ranker
Candidate
Fetch
Home mixer
Timeline
Service
Manhattan
Gizmoduck
Social graph
Tweety Pie
Read Path
Next-gen System
Prediction Service
Home Scorer
Feature Hydration
Manhattan
...
Memcache
Candidate Sources
Fetch
Feature
Scoring
CrMixer
EarlyBird
Utag
Space
Communities
Prediction Service

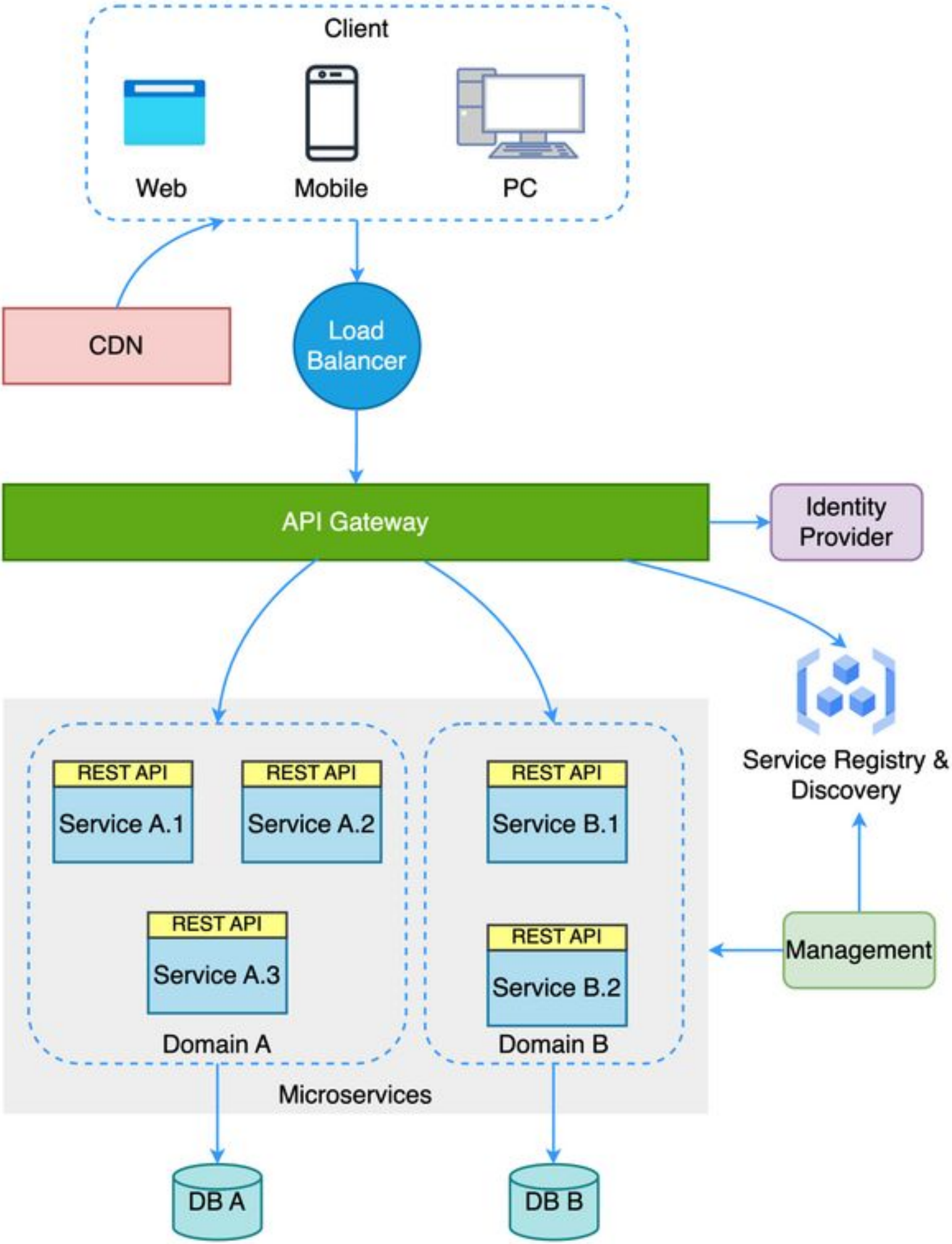
Types of Memory and Storage



ByteByteGo

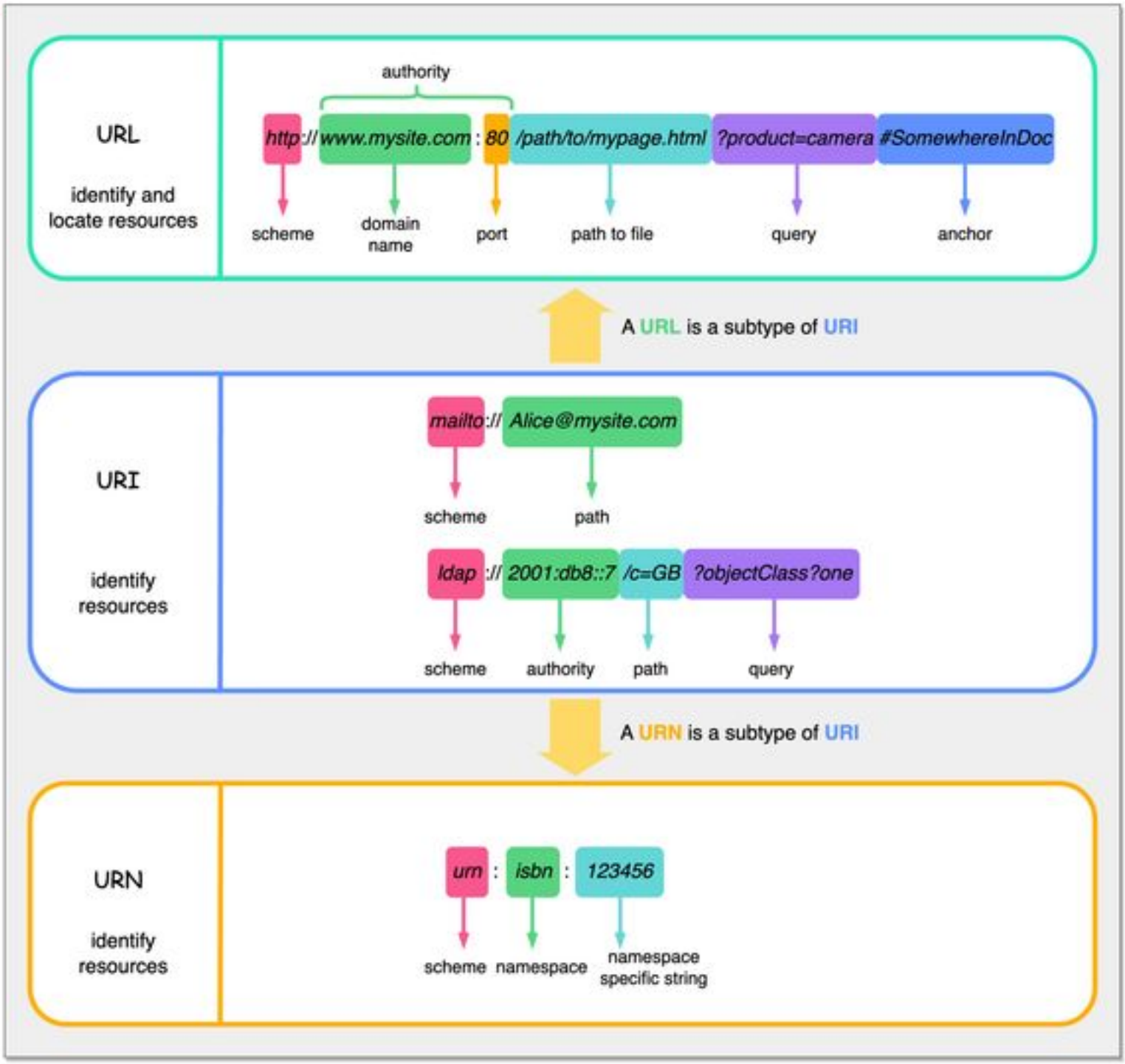
Microservice Architecture

ByteByteGo



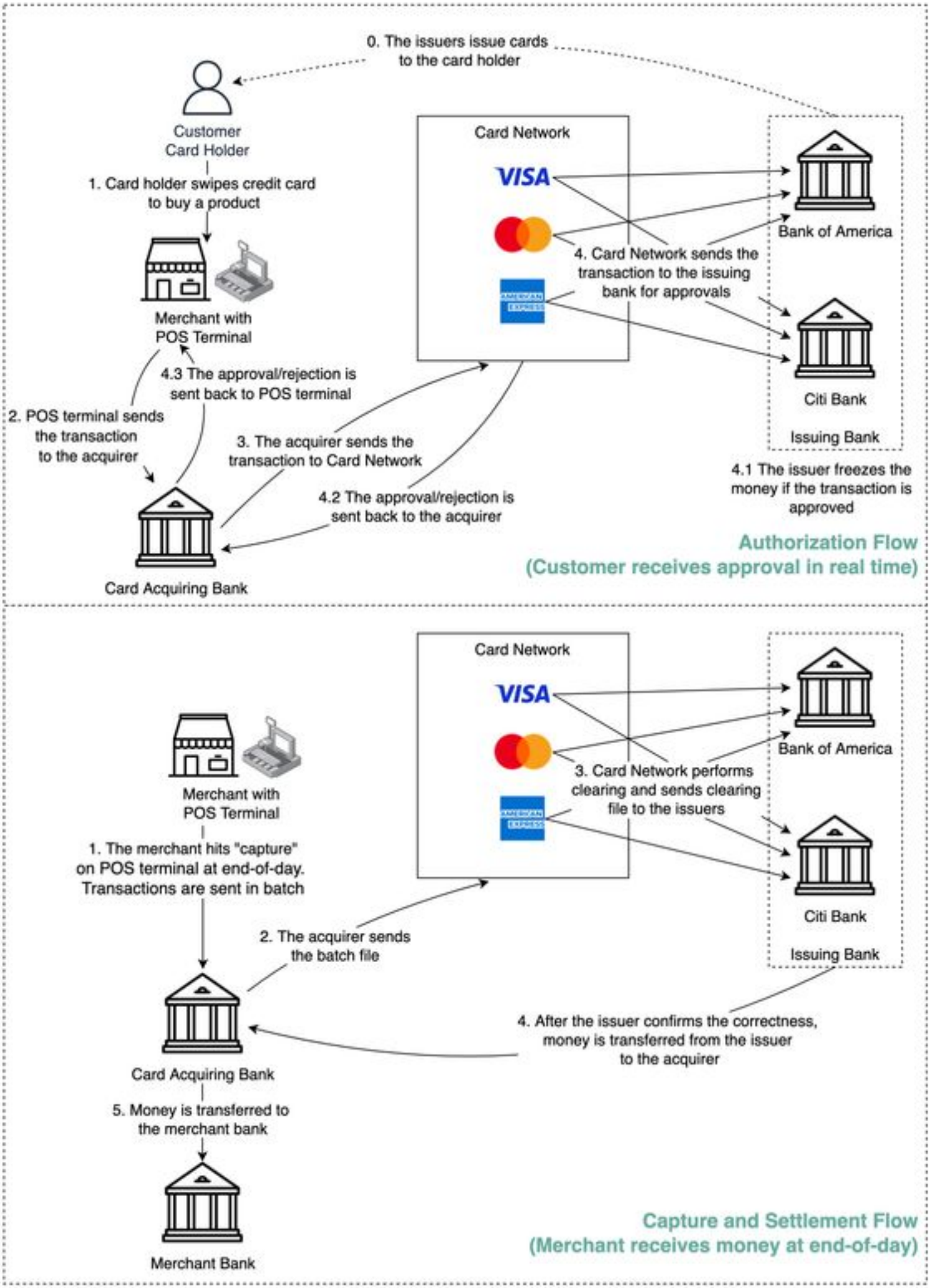
URL vs URI vs URN

blog.bytebytego.com



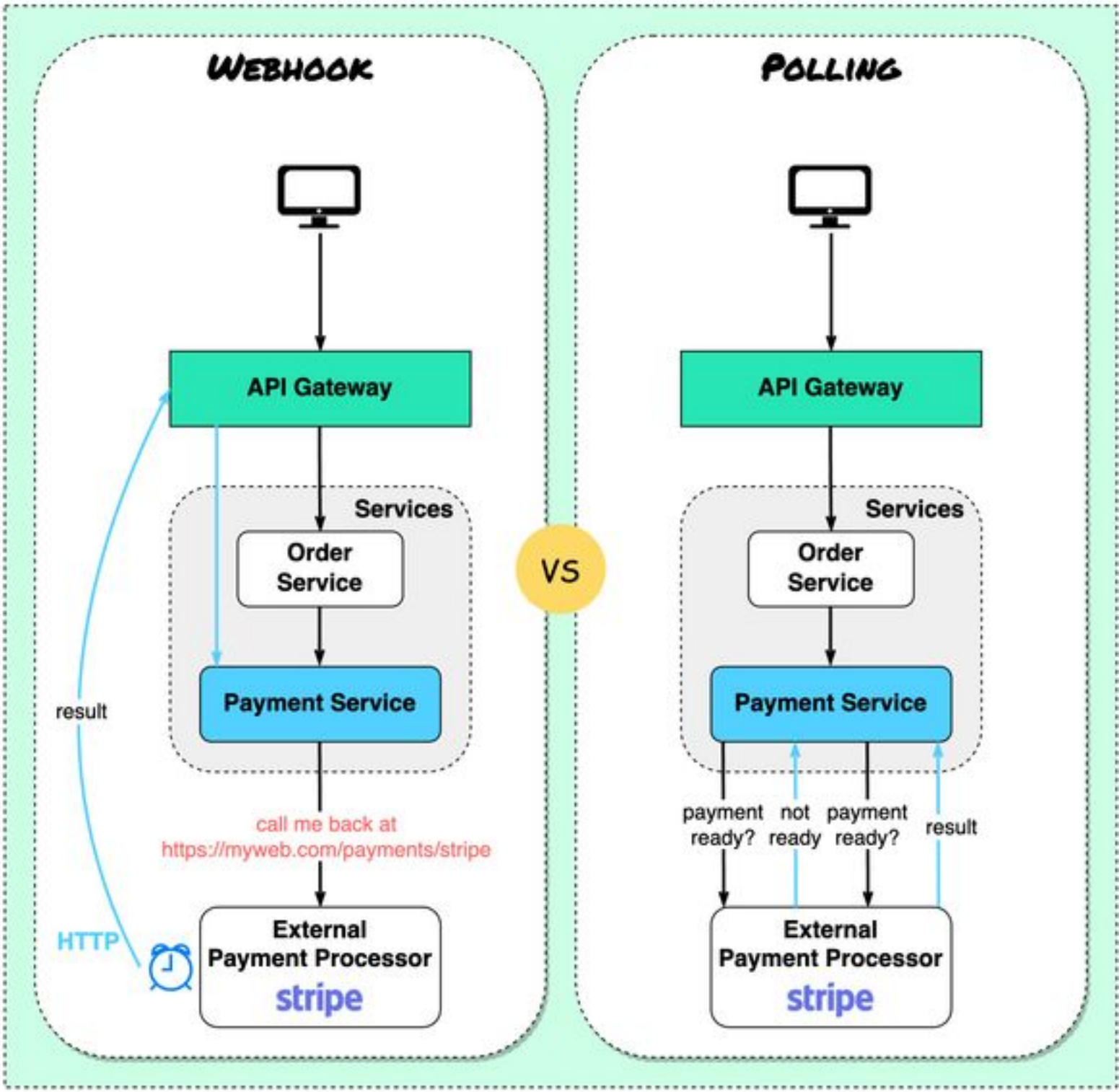
How does VISA Work?

blog.bytebytego.com

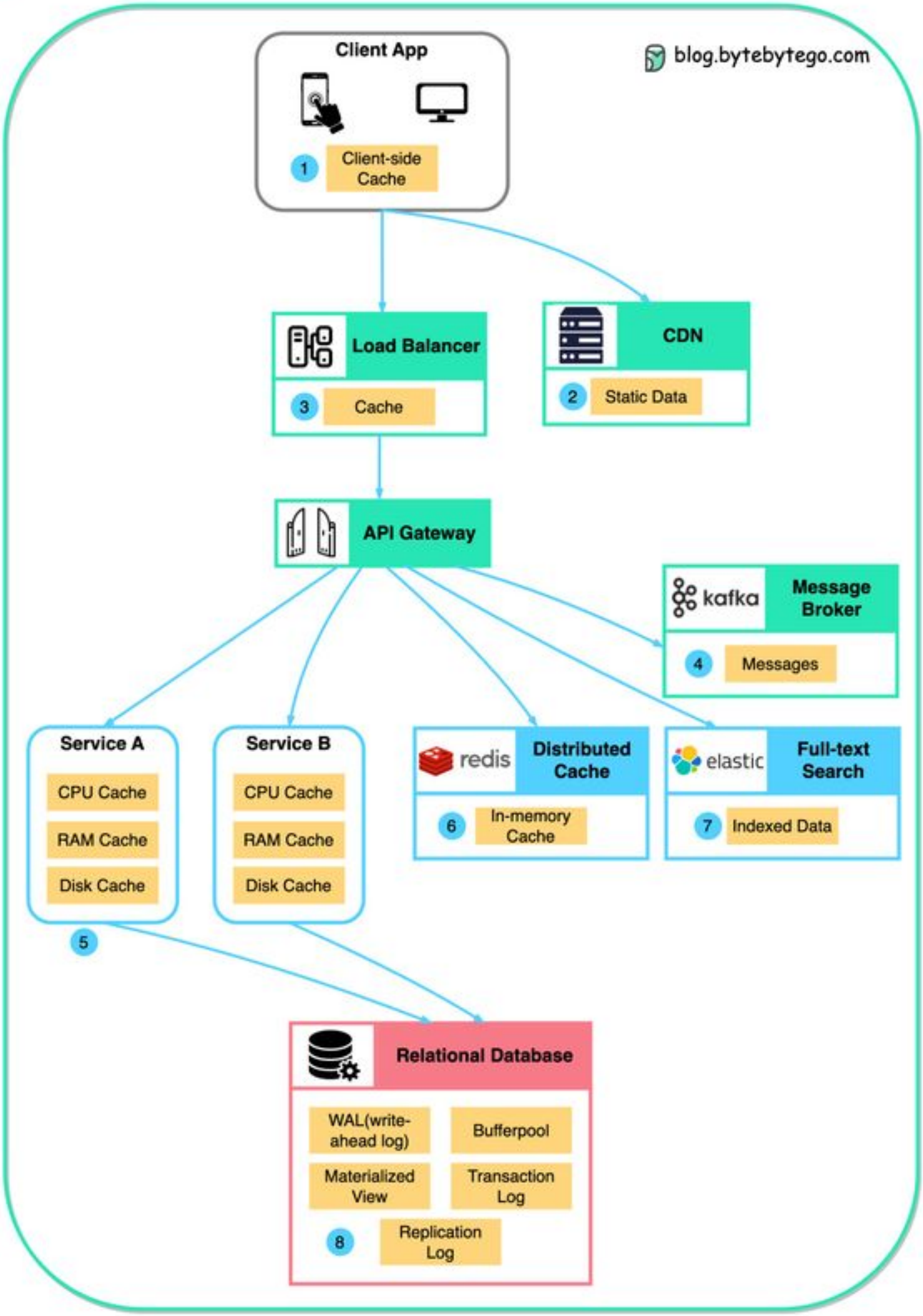


What is a Webhook?

blog.bytebytego.com

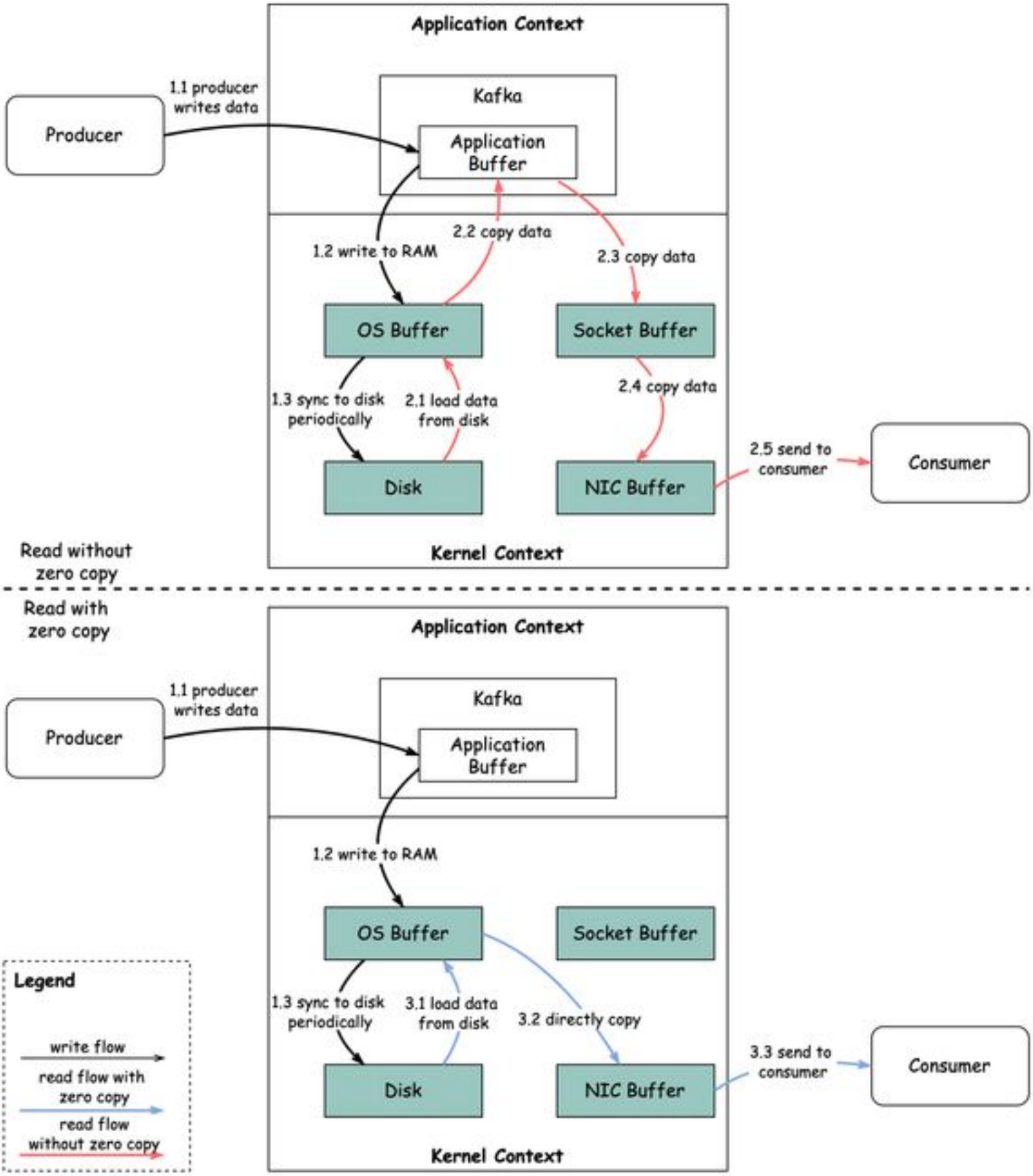


Cache Systems Every Developer Should Know



Why is Kafka Fast?

ByteByteGo



Why is Redis so fast?

100k QPS

ByteByteGo