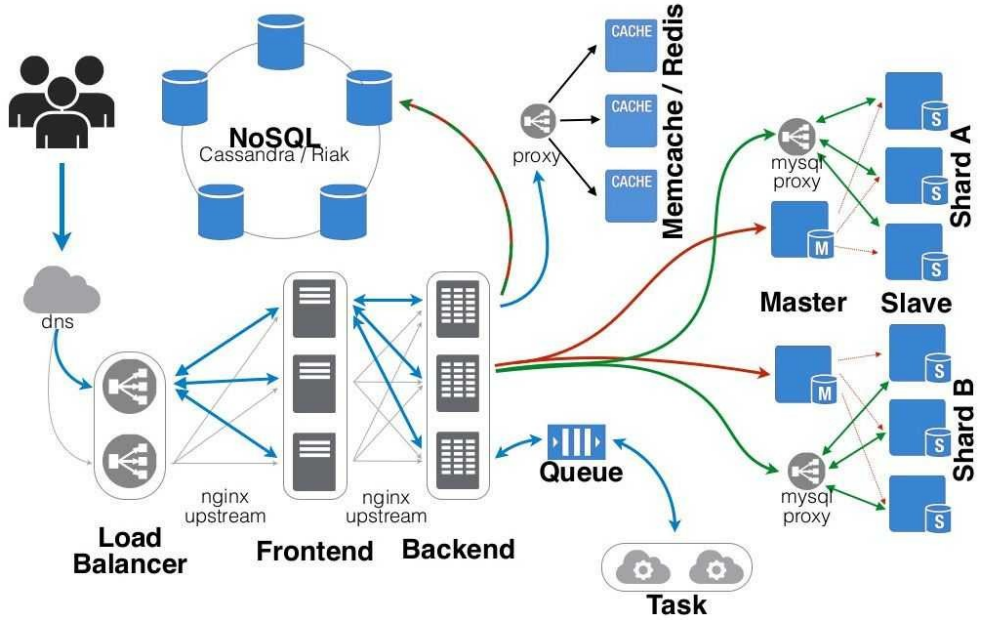
Тема 14. Архітектура високонавантажених систем

Високонавантажена система

- Високонавантажена система — це ІС, яка стабільно працює під великим або піковим навантаженням.
- Вона має забезпечувати обробку тисяч запитів у секунду (RPS) і роботу з великими обсягами даних.



Метрики високонавантаженої системи

Як зрозуміти, що система вже “високонавантажена”, і які метрики це показують? Адже термін “high-load” не означає певну цифру користувачів, а характер поведінки системи під навантаженням — наскільки вона може масштабуватись, стабільно обробляти запити й залишатись доступною.

Продуктивність (Performance) - продуктивність описує, як швидко система обробляє запити користувачів і наскільки ефективно використовує свої ресурси.

Ключовими метриками є:

- **RPS (Requests per Second)** — кількість запитів, які система може обробити за секунду. Якщо система стабільно обробляє тисячі або десятки тисяч RPS, це вже ознака високого навантаження. Наприклад, веб-API банку може мати лише 500 RPS, тоді як маркетплейс — понад 20 000.
- **Latency (затримка)** — час від запиту до отримання відповіді. Для зручності користувача важливо, щоб затримка була мінімальною: приблизно до 100 мс для API, до 1 секунди для веб-інтерфейсів. Затримка залежить не лише від швидкості коду, а й від мережі, кешу, бази даних і навіть географії серверів.
- **Throughput (пропускна здатність)** показує, скільки даних система може передати за певний час. Наприклад, відеосервіси вимірюють throughput у мегабайтах за секунду, а фінансові системи — у кількості транзакцій.
- **Concurrency (одночасність)** — скільки запитів обробляється одночасно. Для великих систем це можуть бути сотні тисяч або навіть мільйони одночасних підключень.

Високонавантажена система проявляє себе тоді, коли збільшення кількості запитів не призводить до різкого зростання затримки чи падіння RPS — тобто продуктивність зберігається стабільною.

Метрики високонавантаженої системи

Доступність — це здатність системи залишатися в робочому стані, навіть коли частина її компонентів виходить з ладу.

Це один із головних показників для будь-якої високонавантаженої архітектури.

Основні метрики:

- **Uptime** — частка часу, коли система доступна користувачам. Наприклад, 99.9% uptime означає приблизно 9 годин простою на рік,а 99.99% — лише близько 1 години. Такі цифри вимагають реплікації, автоматичного відновлення і балансування навантаження.
- **SLA, SLO, SLI** — три рівні домовленостей про доступність:
  - SLA (Service Level Agreement) — договірне зобов’язання перед клієнтами;
  - SLO (Objective) — ціль усередині компанії (наприклад, «99.95% доступності»);
  - SLI (Indicator) — реальне виміряне значення.
- **MTBF (Mean Time Between Failures)** — середній час між збоями.
- **MTTR (Mean Time To Recovery)** — середній час відновлення після збою.
- У стійких системах MTTR вимірюється хвилинами або навіть секундами.

Якщо система має високу доступність, користувач практично не помічає відмов — навіть коли відбуваються збої чи оновлення.

Метрики високонавантаженої системи

Надійність (Reliability) і **Відмовостійкість (Fault Tolerance):**

- **Error Rate** — частка невдалих запитів (% 5xx помилок).
- **Success Rate** — частка успішних відповідей.
- **Failover Time** — час автоматичного перемикання на резервний сервер.
- **Recovery Point Objective (RPO)** — скільки даних можна втратити при відмові.
- **Recovery Time Objective (RTO)** — за який час система має відновитися.

У високонавантажених системах RTO — хвилини або секунди, RPO → 0.

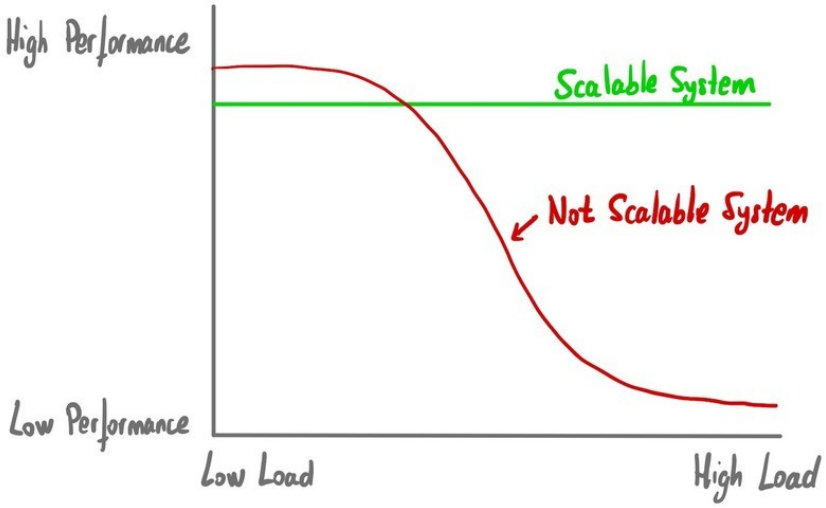
Масштабованість (Scalability)

- Elasticity — наскільки система може збільшити чи зменшити ресурси автоматично.
- Horizontal scaling efficiency — як ефективно система використовує нові ноди (ідеально, якщо подвійне збільшення ресурсів → подвійна продуктивність).
- Utilization — завантаження CPU, пам’яті, диска, мережі (%).

Якщо при горизонтальному масштабуванні система зберігає лінійну продуктивність — це класична ознака high-load.

Проблема зростання навантаження

- Класичний моноліт має обмеження: зі збільшенням користувачів зростає затримка, збої, черги запитів.



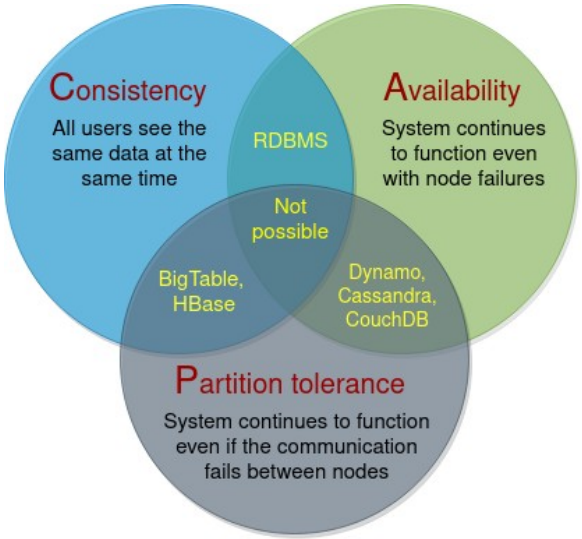
Приклади високонавантажених систем

- Facebook — понад 100 млрд запитів GraphQL на день.
- Netflix — 250 млн годин відео щодня через CDN.
- Amazon — динамічне масштабування на пікові розпродажі.

CAP теорема

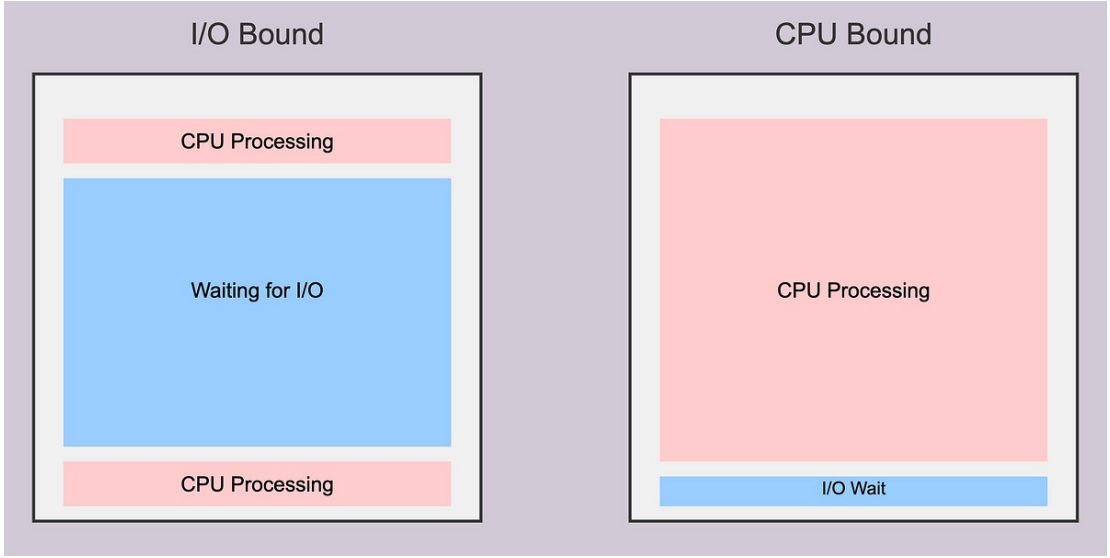
CAP теорема визначає неможливість одночасного досягнення:

- Consistency — узгодженість даних
- Availability — доступність системи
- Partition tolerance — стійкість до розділення мережі

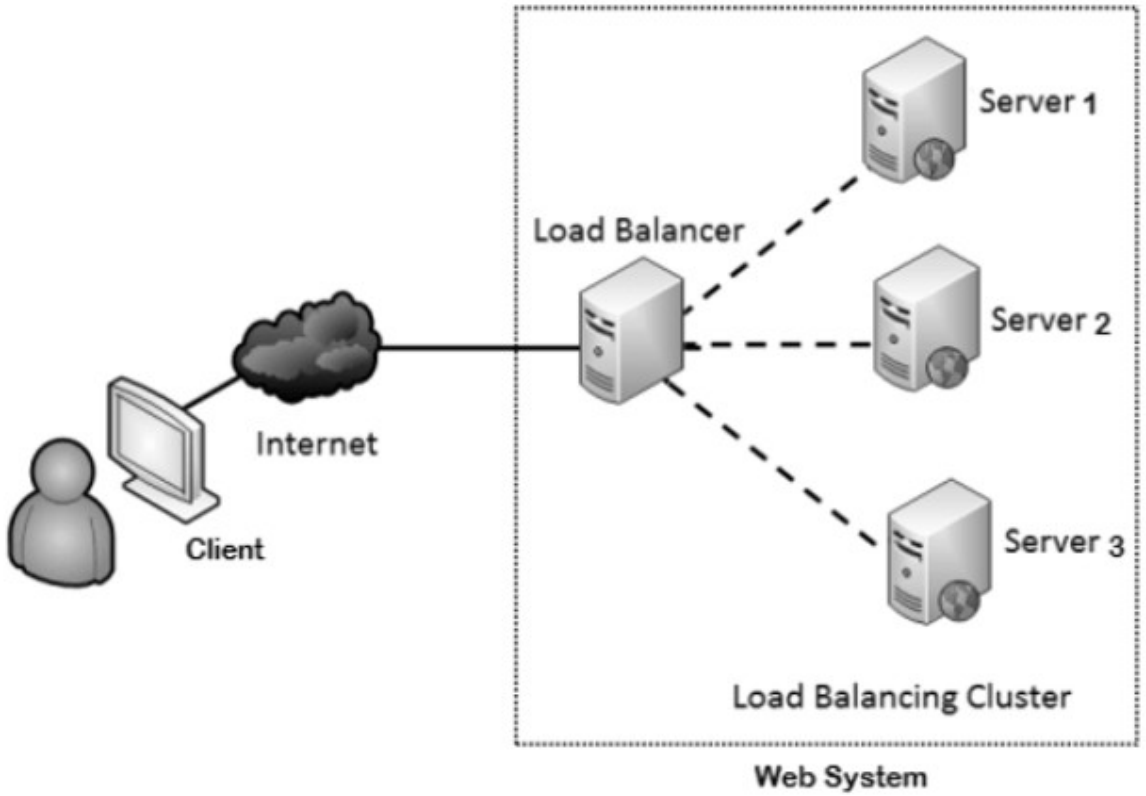


Типи навантаження

- CPU-bound — обчислення займають більшість часу.
- IO-bound — очікування відповіді від диска або мережі.
- Network-bound — обмеження пропускної здатності мережі.



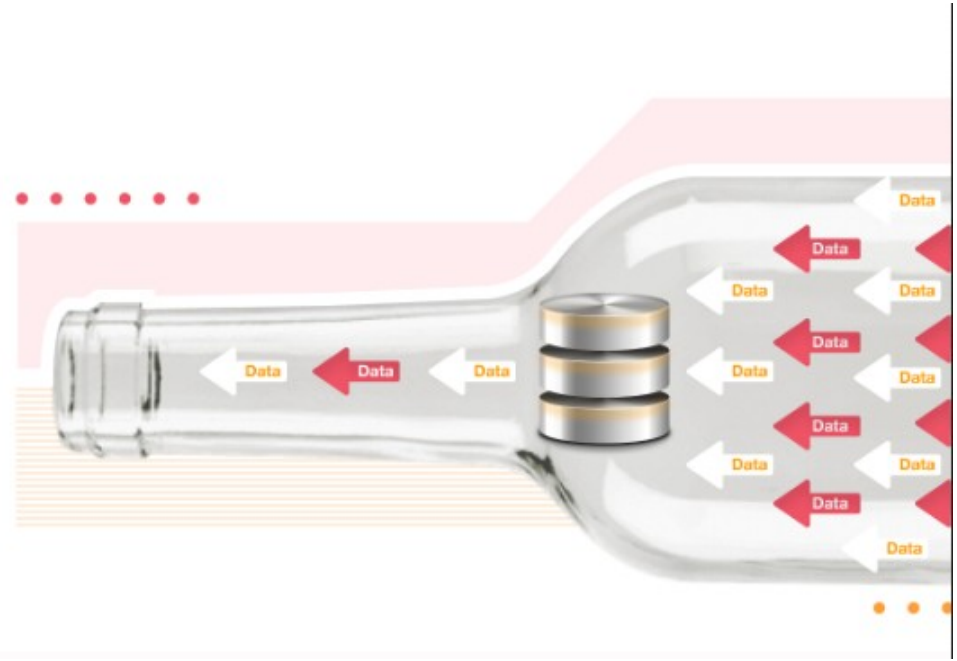
Базова архітектура веб-додатку



Вузькі місця системи

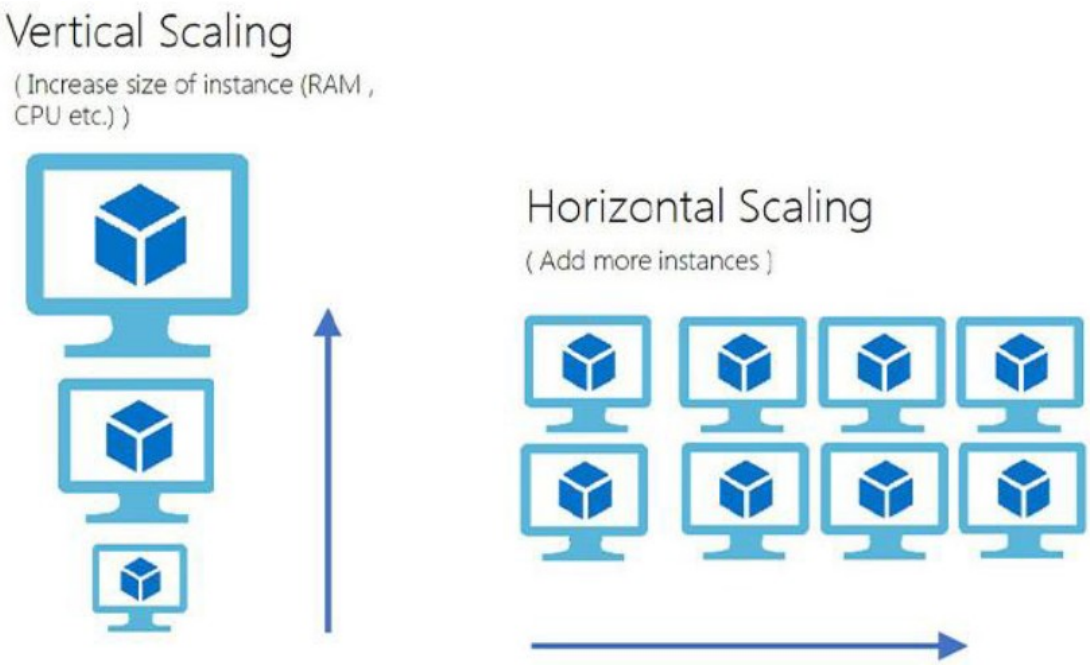
Bottlenecks — компоненти, які уповільнюють роботу всієї системи.

Приклади: повільна база даних, обмежена мережа, перевантажений процесор.



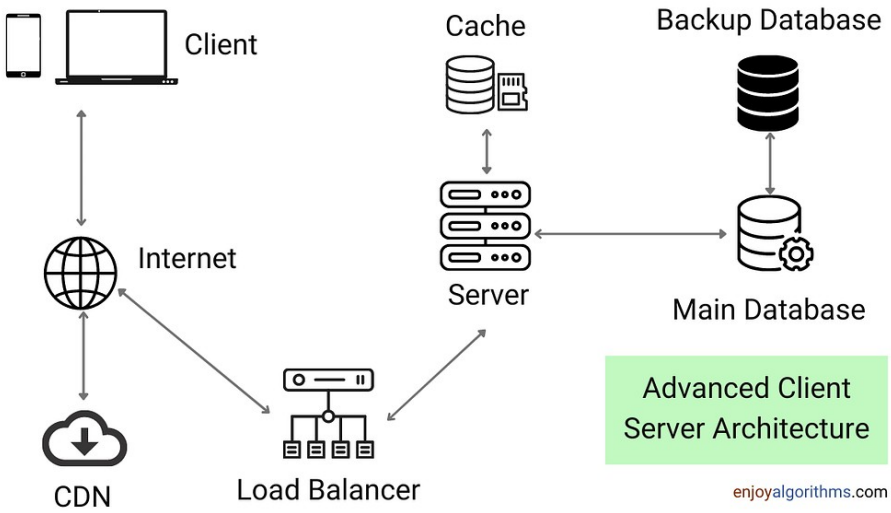
Масштабування систем

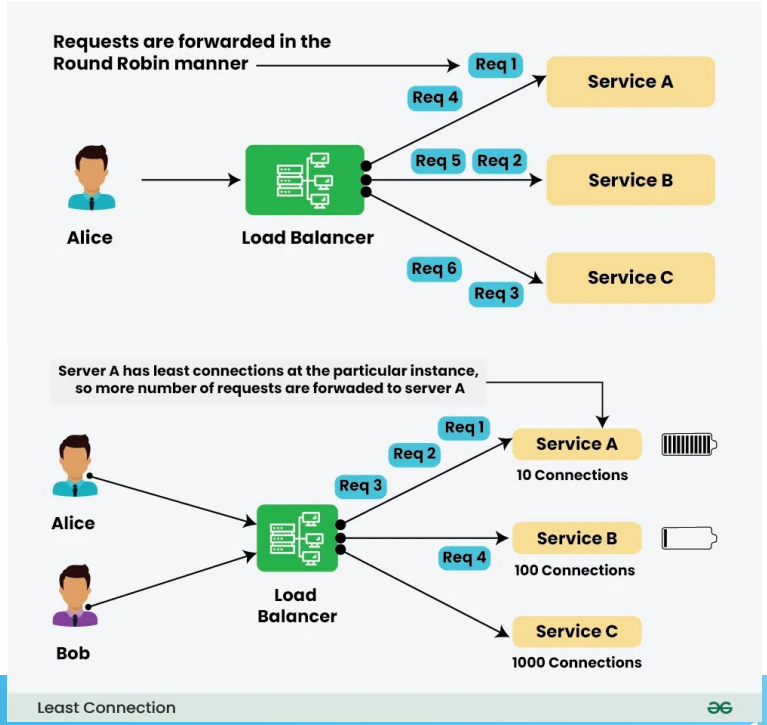
- Вертикальне — збільшення ресурсів одного сервера.
- Горизонтальне — додавання нових серверів.



Балансування навантаження

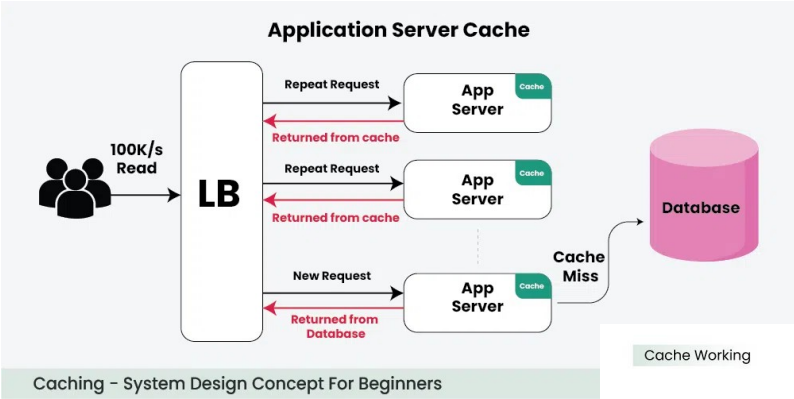
- Load Balancer — компонент, який рівномірно розподіляє трафік між серверами.
- Алгоритми: Round-robin, Least Connections, IP Hash.

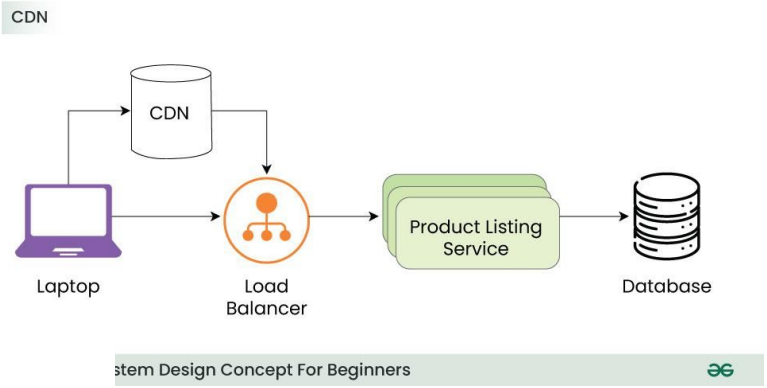


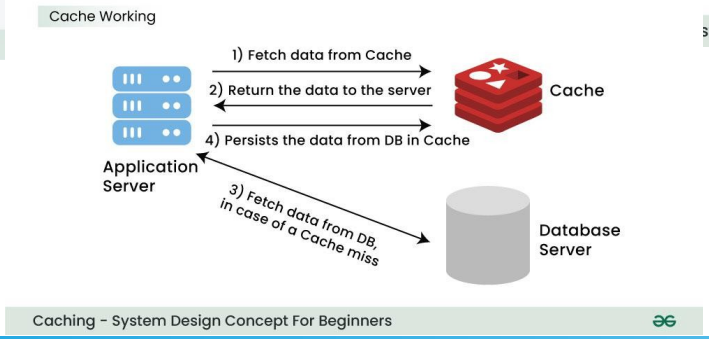


Кешування

- Кешування — збереження часто використовуваних даних для швидкого доступу.
- Рівні кешу: клієнтський, CDN, бекенд (Redis, Memcached).

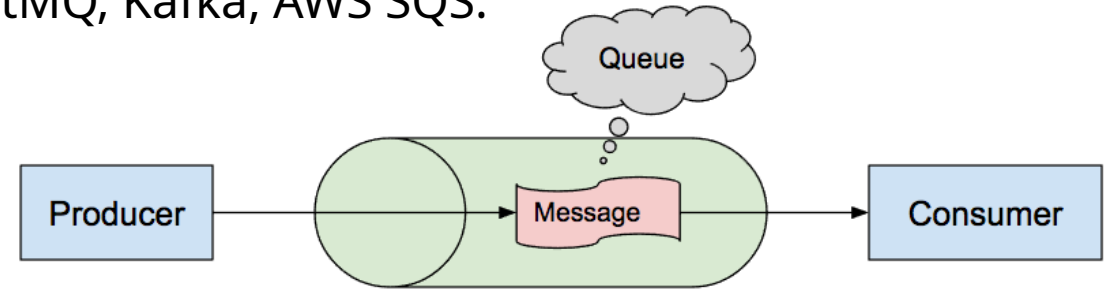


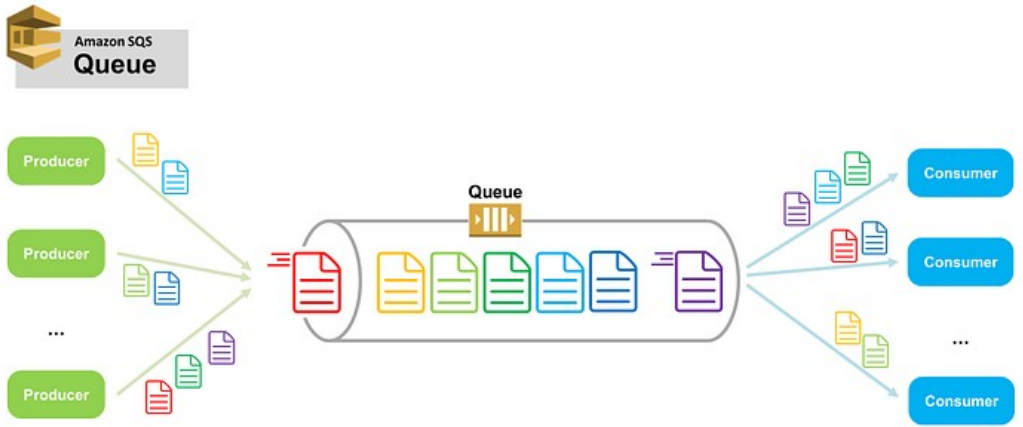




Системи черг

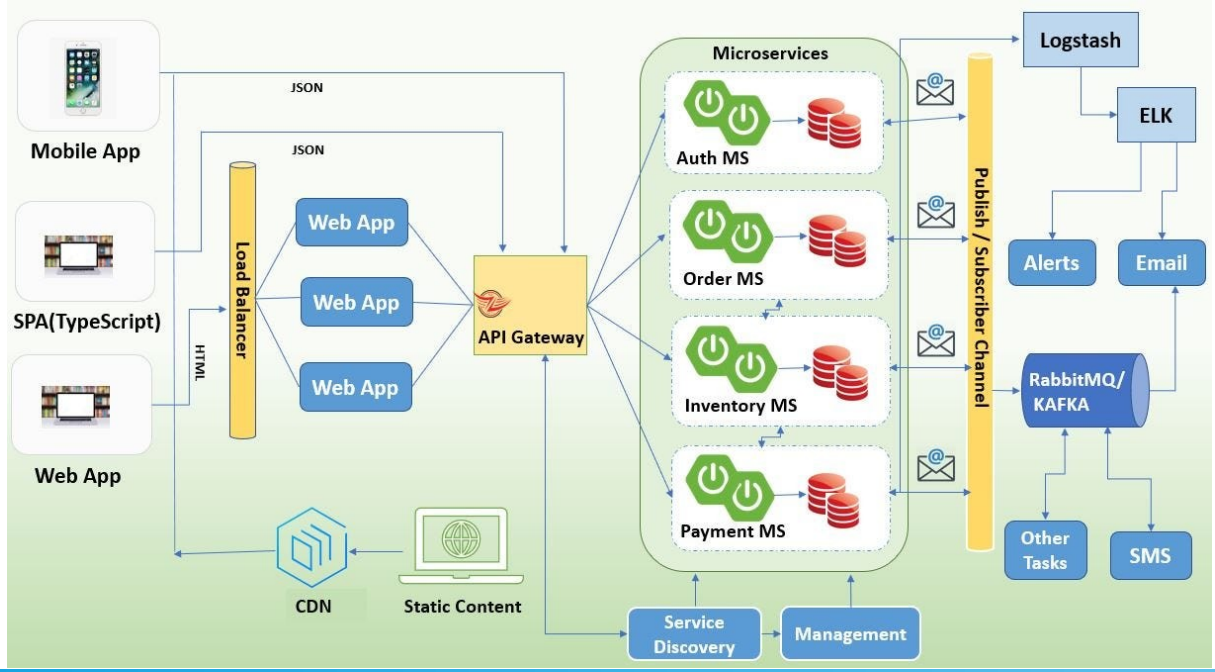
- Message Queue — буфер для асинхронної обробки запитів.
- Приклади: RabbitMQ, Kafka, AWS SQS.





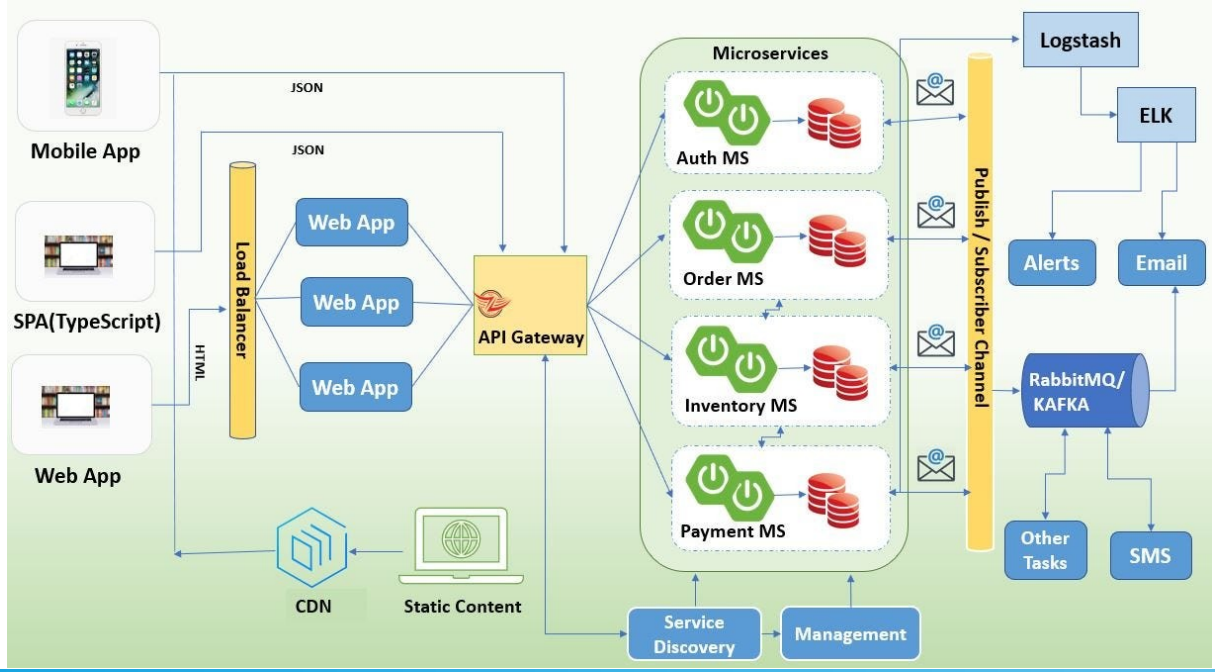
Мікросервісна архітектура

- Мікросервіси — це незалежні компоненти, які виконують окремі функції системи.
- Переваги: масштабованість, відмовостійкість, незалежні оновлення.



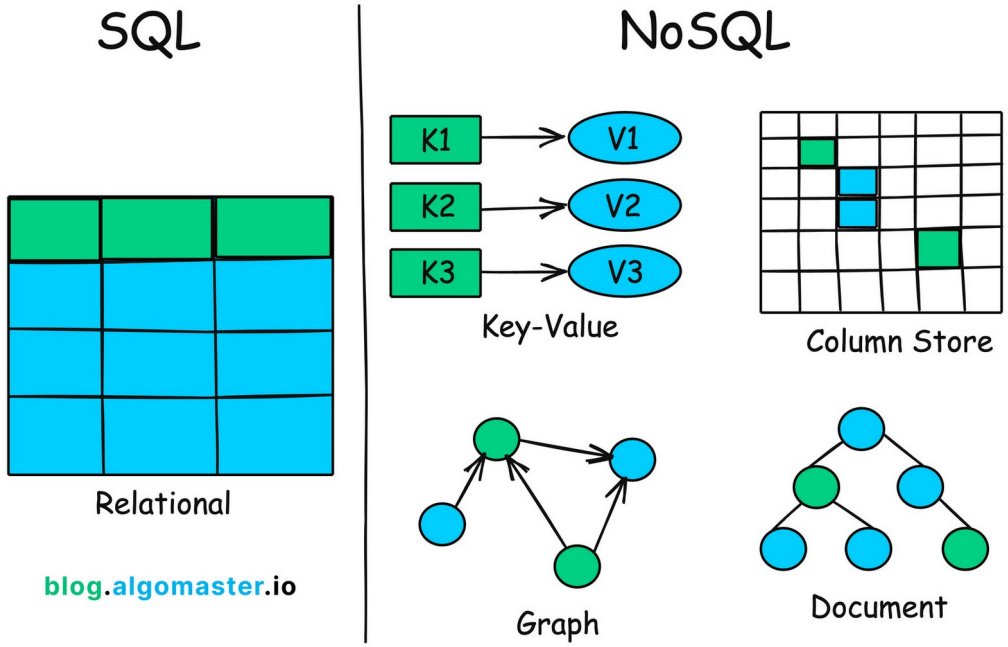
API Gateway

- API Gateway — єдина точка доступу до мікросервісів.
- Здійснює маршрутизацію, автентифікацію, обмеження трафіку.



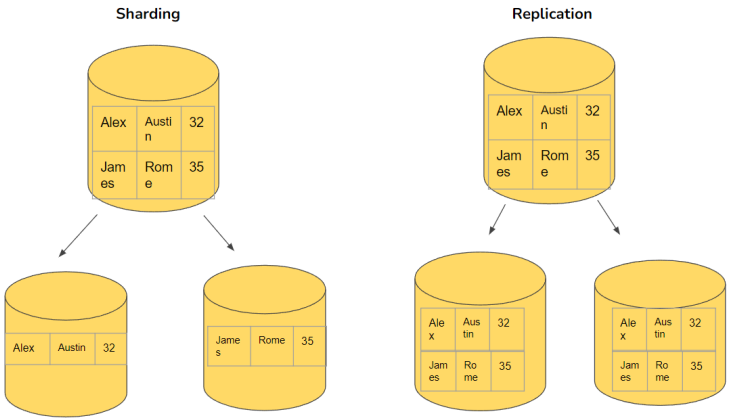
SQL vs NoSQL

- SQL — реляційна модель, ACID властивості.
  - **A** — Atomicity (атомарність). Транзакція виконується вся або нічого. Якщо сталася помилка — rollback повертає систему в попередній стан.
  - **C** — Consistency (узгодженість). Інваріанти (PRIMARY/FOREIGN KEY, UNIQUE, CHECK, тригери) завжди зберігаються до й після транзакції. Забороняє “невалідні” стани (наприклад, негативний баланс, «висяча» зовнішня ссылка).
  - **I** — Isolation (ізоляція). Паралельні транзакції не заважають одна одній логічно.
  - **D** — Durability (довговічність)
- NoSQL — гнучка модель, BASE властивості.
  - **B** — Basically **A**vailable (базова доступність)
  - **S** — Soft State (м’який стан)
  - **E** — Eventual Consistency конечна узгодженість

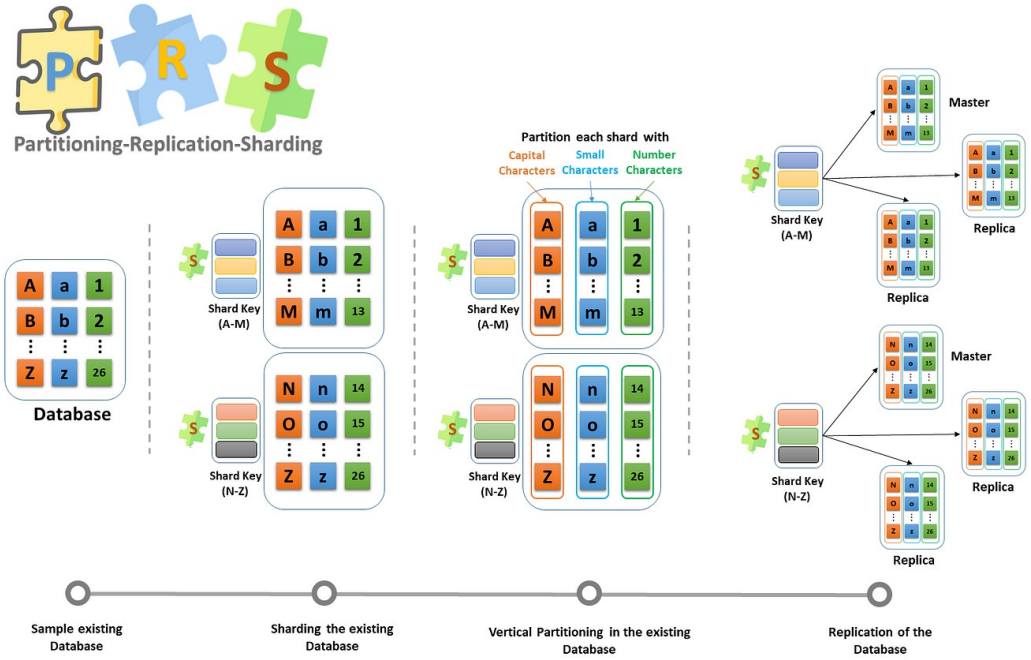


Шардінг і реплікація

- Шардінг — розподіл даних між кількома базами для зниження навантаження.

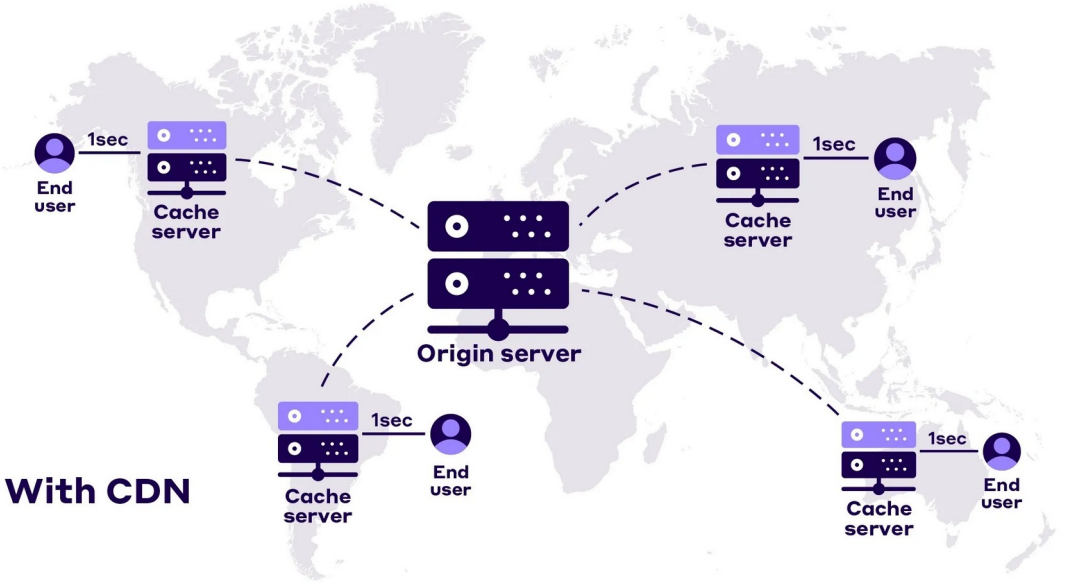


- Реплікація — створення копій баз даних для забезпечення доступності.



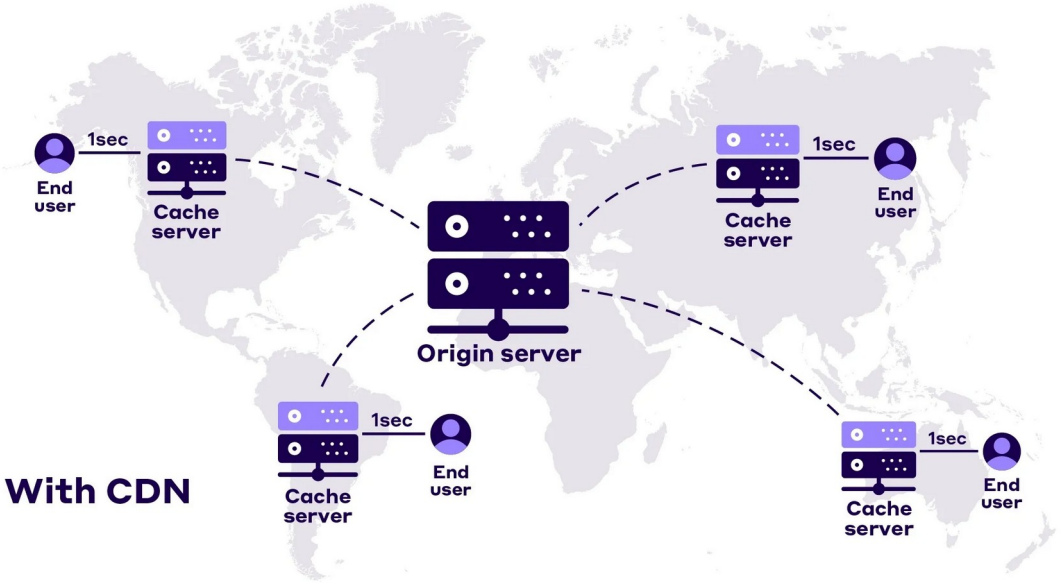
CDN

- CDN (Content Delivery Network) — глобальна мережа серверів для швидкої доставки контенту.



CDN

- CDN (Content Delivery Network) — глобальна мережа серверів для швидкої доставки контенту.



Висновки

- Архітектура високонавантажених систем — це не один шаблон, а підхід до мислення: як розподілити роботу, зменшити затримки, запобігти вузьким місцям і забезпечити зростання без втрати стабільності.
- Високонавантажені системи вимагають компромісів і балансу між швидкістю, надійністю, витратами та простотою — і саме вміння ці компроміси робити визначає рівень архітектора.

Дякую за увагу!

Ви

- CDN (Content Delivery Network) — глобальна мережа серверів для швидкої доставки контенту.