CONCEPTOS CLAVE DISEÑO SISTEMAS A GRAN ESCALA

INTRODUCCIÓN A CONCEPTOS CLAVE

Daniel Blanco Calviño

LA COMPLEJIDAD EN EL DESARROLLO DE SOFTWARE

- La parte más complicada no suele estar en implementar funcionalidades.
  - **Suelen ser bastante básicas.**

- Lo difícil es conseguir que **funcionen bien para cientos de millones de usuarios.**

LA COMPLEJIDAD EN EL DESARROLLO DE SOFTWARE

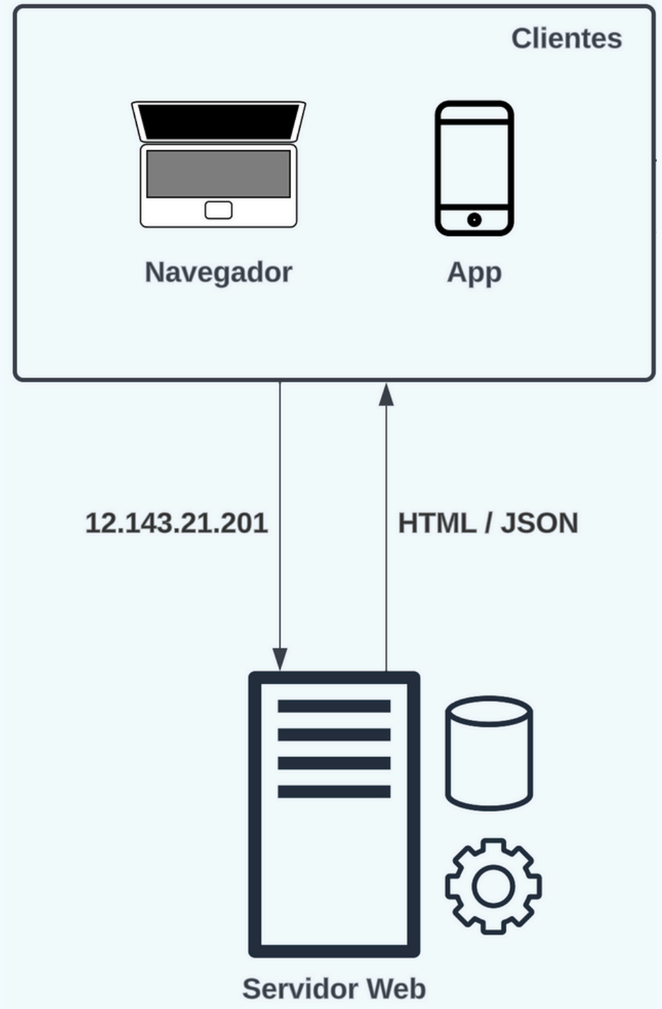
- La parte más complicada no suele estar en implementar funcionalidades.
  - **Suelen ser bastante básicas.**

- Lo difícil es conseguir que **funcionen bien para cientos de millones de usuarios.**





DISEÑO DE PARTIDA



VENTAJAS Y DESVENTAJAS PUNTO DE PARTIDA

- **Sencillo** y rápido de **implementar**.
- Fácil de **mantener**.
- **Aprendizaje**.

VENTAJAS Y DESVENTAJAS PUNTO DE PARTIDA

- **Sencillo** y rápido de **implementar**.
- Fácil de **mantener**.
- **Aprendizaje**.

- Imposible de **escalar.**
- Baja **fiabilidad**.

COMPONENTES DE UN SISTEMA A GRAN ESCALA

- **Load Balancer**
- **API Gateway**
- **Message Broker**
- **Cache**
- **CDN** (Content Delivery Network)
- **Data Centers**

DNS

https://api.example.com/user?name=Daniel

Protocolo

Subdominio

Dominio

Ruta

Query Params

DNS

Dominio	Dirección IP
whatever.com	15.23.154.21
example.com	12.143.21.201
example3.es	13.141.25.98

DNS

- **Servidores DNS gratuitos**
  - Google (8.8.8.8, 8.8.8.4), Cloudfare (1.1.1.1, 1.0.0.1)
- **Servidores DNS de pago**
  - Ofrecen características a mayores (balanceo de carga basado en DNS etc.)

Dominio	Dirección IP
whatever.com	15.23.154.21
example.com	12.143.21.201
example3.es	13.141.25.98

APIS

- **Application Programming Interface.**
  - Interfaz de comunicación con nuestro sistema.
  - Expone las operaciones que se pueden realizar en él.

APIS

- **Application Programming Interface**
  - Interfaz de comunicación con nuestro sistema.
  - Expone las operaciones que se pueden realizar en él.

- **Tipos de APIs**
  - Públicas
  - Privadas
  - Partners

APIS - BUENAS PRÁCTICAS

- **Deben encapsular por completo nuestro sistema.**
  - Si tenemos que saber cómo funciona por dentro existen problemas en el diseño.

APIS - BUENAS PRÁCTICAS

- **Deben encapsular por completo nuestro sistema.**
  - Si tenemos que saber cómo funciona por dentro existen problemas en el diseño.

- Debe estar **desacoplada** de la lógica interna.
  - De lo contrario, si hacemos cambios en el sistema seguramente afecte a la API.

APIS - BUENAS PRÁCTICAS

- **Deben encapsular por completo nuestro sistema.**
  - Si tenemos que saber cómo funciona por dentro existen problemas en el diseño.

- Debe estar **desacoplada** de la lógica interna.
  - De lo contrario, si hacemos cambios en el sistema seguramente afecte a la API.

- **Versionado.**
  - No se modifican, se versionan. Las modificaciones deben ser retrocompatibles.

APIS - BUENAS PRÁCTICAS

- **Deben encapsular por completo nuestro sistema.**
  - Si tenemos que saber cómo funciona por dentro existen problemas en el diseño.

- Debe estar **desacoplada** de la lógica interna.
  - De lo contrario, si hacemos cambios en el sistema seguramente afecte a la API.

- **Versionado.**
  - No se modifican, se versionan. Las modificaciones deben ser retrocompatibles.

- **Operaciones idempotentes.**
  - Si una API se llama varias veces con los mismos parámetros debe tener siempre el mismo resultado.

APIS - BUENAS PRÁCTICAS

- **Paginación.**
  - Evitar devolver conjuntos masivos de datos. Utilizar paginación en estos casos.

APIS - BUENAS PRÁCTICAS

- **Paginación.**
  - Evitar devolver conjuntos masivos de datos. Utilizar paginación en estos casos.

- **Utilizar operaciones asíncronas.**
  - Operaciones lentas.