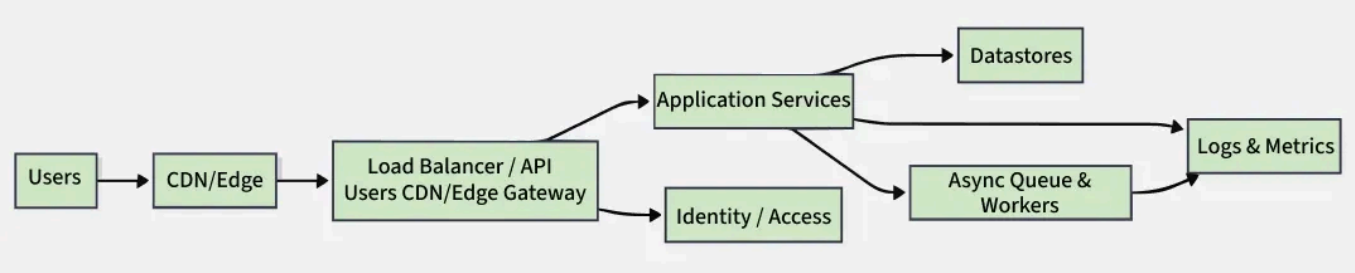
High Level Design (HLD)

High-level design (HLD) is an initial step in the development of applications where the overall structure of a system is planned.

- Focuses on how different components of the system work together **without going into internal coding or implementation details**.
- Helps everyone involved in the project understand the goals and ensures smooth communication throughout development.
- Crucial for developers, architects, and product managers as it aligns all stakeholders with the project objectives.
- Also known as **macro-level design**.

Common Request Flow in Most Applications



Users → Edge/Gateway → Services → Data → Async → User

- **Users → Edge/Gateway:**
  CDN/Edge handles caching, TLS termination, basic DDoS protection; Load Balancer/API Gateway routes requests.
- **Gateway → Services:**
  Application layer runs authentication and business logic, then fans out to internal dependencies.
- **Services → Data:**
  - Read/write operations to SQL, NoSQL, object storage, or search systems.
  - Cache is used for hot/c frequently accessed reads.
  - Writes are persisted in storage.
- **Services → Async:**
  Heavy or non-urgent tasks (emails, webhooks, re-indexing, background jobs) are offloaded to queues/workers.
- **Auth & Observability:**

    - Authentication can happen at gateway and service layers.
    - System emits logs, metrics, and traces for monitoring and alerts.
- **Response → User:**
  Final response is returned via gateway/edge. Caches may be updated for faster subsequent access.

Components of High-Level Design

Understanding the components of HLD is crucial for creating systems that meet user and technical requirements.

1. System Architecture

A high-level overview of the entire system representing the structure and relationships between components. It visually showcases how different parts interact and function.

2. Modules and Components

Breaks the system into modules or components, each with a specific role and responsibility. Each part contributes to the overall functionality, improving efficiency and maintainability.

3. Data Flow Diagrams (DFDs)

DFDs illustrate how data moves through the system—where it starts, where it flows, and how it is processed.

4. Interface Design

Includes:

- API design for system integration
- UI design for user interaction
  Ensures seamless communication and usability across system components.

5. Technology Stack

Defines the collection of tools and technologies used to build the system:

- Programming languages
- Frameworks
- Databases
- Supporting tools

6. Deployment Architecture

Describes how the system will be hosted and accessed, including:

- Server setup
- Cloud infrastructure
- Networking and security considerations

Purpose and Characteristics of High-Level Design

- The purpose of this High-Level Design (HLD) is to add the necessary detailed description to represent a suitable model.
- This is designed to help with operational requirements and will help to understand how the modules interact.
- Basically, HLD is a technical representation of functional requirements and the flow of information across components.

Characteristics of high-level design include

- A diagram representing each design aspect is included in the HLD (which is based on business requirements and anticipated results).
- Description of hardware, software interfaces, and also user interfaces.
- The workflow of the user's typical process is detailed, along with performance specifications.

How HLD is Different from LLD :

- High-level design, or HLD, is a general system design where we do tradeoffs between different frameworks, components, and different databases, and we choose the best considering what the business needs and how the system should work, both in terms of functional and non-functional aspects.
- Whereas LLD (low-level design) translates the HLD into smaller and more specific details, it includes class diagrams, methods, data structures, and algorithms, focusing on how each part will be implemented.

✅ HLD Examples

1. Designing WhatsApp – HLD

- Client → Server architecture
- Components:

    - Messaging Service
    - Notification Service
    - User Service
    - Media Storage
    - Database cluster
- Data flow:
    - Message → Queue → Chat Service → Database
- Tech choices:
    - NoSQL for chat storage
    - CDN for image delivery
    - Load Balancer for scaling

---

2. E-commerce System – HLD

Components:

- User Service
- Product Service
- Cart Service
- Payment Gateway
- Order Service
- Delivery Service
- Recommendation Engine

Includes:

- Architecture diagram
- API gateway
- Database selection (SQL for orders, NoSQL for product catalog)

---

3. Streaming Platform (Netflix) – HLD

- Video Streaming Service
- Metadata Service
- CDN for video delivery
- Encoding Service

- User Authentication
- Adaptive bitrate algorithm
- Distributed database

---

4. Hospital Appointment System – HLD (Your project example)

Modules:

- Patient Login Service
- Doctor Management
- Appointment Booking
- Notifications
- Medical Records
- Admin Portal

Data flow:

- Patient → API → Appointment Service → DB → SMS/Email Notification