



CDN (Content Delivery Network) – বিস্তারিত নোট







CDN এর পূর্ণরূপ

CDN = Content Delivery Network

CDN হলো সারা বিশ্বে ছড়িয়ে-ছিটিয়ে (geographically dispersed) থাকা অনেকগুলো সার্ভারের একটি নেটওয়ার্ক, যার মূল উদ্দেশ্য হলো **Static Content** দ্রুত ও নির্ভরযোগ্যভাবে ইউজারের কাছে পৌঁছে দেওয়া।



CDN কীভাবে কাজ করে (What is CDN?)

CDN মূল সার্ভার (Origin Server) থেকে Static Content যেমন: -  Images -  Videos -  CSS Files -  JavaScript Files - Fonts

এসব কন্টেন্ট **cache করে রাখে** এবং ইউজার যখন রিকোয়েস্ট করে, তখন ইউজারের সবচেয়ে কাছের CDN সার্ভার থেকে কন্টেন্ট serve করে।

এতে করে latency কমে এবং ওয়েবসাইট দ্রুত লোড হয়।



CDN কেন দরকার?

যদি সব ইউজার একটাই origin server থেকে ডাটা নেয়, তাহলে: - দূরের ইউজারের জন্য লোড টাইম বেশি হবে - সার্ভারের উপর অতিরিক্ত চাপ পড়বে - Bandwidth খরচ বেড়ে যাবে

CDN এই সমস্যাগুলো সমাধান করে।



CDN কীভাবে কাজ করে (Step by Step Workflow)



High-Level Flow

User A একটি image রিকোয়েস্ট করে (image.png)

ইমেজ URL সাধারণত CDN provider এর ডোমেইন ব্যবহার করে।

উদাহরণ: - <https://mysite.cloudfront.net/logo.jpg> (Amazon CloudFront) - <https://mysite.akamai.com/image-manager/img/logo.jpg> (Akamai)

CDN সার্ভার চেক করে ইমেজটি cache এ আছে কিনা

- না থাকলে → Origin Server এ রিকোয়েস্ট পাঠায়

Origin Server ইমেজটি CDN সার্ভারে পাঠায়

- সাথে একটি **TTL (Time-To-Live)** HTTP header থাকতে পারে
- TTL বলে দেয় কতক্ষণ পর্যন্ত ইমেজটি cache এ থাকবে

CDN সার্ভার: - ইমেজটি cache করে - User A কে ইমেজটি serve করে

User B একই ইমেজ রিকোয়েস্ট করে

TTL expire না হওয়া পর্যন্ত: - CDN সরাসরি cache থেকেই ইমেজ serve করে - Origin server এ নতুন করে যেতে হয় না

CDN ব্যবহারের সুবিধা (Advantages)

Fast Data Loading

→ ইউজারের কাছাকাছি সার্ভার থেকে ডাটা আসে

Reduced Latency

→ Distance কম হওয়ায় response time কমে

Bandwidth Saving

→ Origin server থেকে বারবার ডাটা পাঠাতে হয় না

High Availability & Scalability

→ Traffic spike হলেও CDN handle করতে পারে

Better User Experience

→ Website smoother & faster হয়

CDN ব্যবহার করার সময় যেসব বিষয় Consider করা উচিত

1. Cost

- CDN সাধারণত third-party provider দ্বারা চালিত
- Data transfer (in/out) এর উপর চার্জ নেয়

- খুব কম ব্যবহৃত asset CDN এ রাখলে cost বাড়ে কিন্তু benefit কম

Tip: Infrequently used assets CDN থেকে বাদ দেওয়া যেতে পারে

2. Cache Expiry (TTL) ঠিক করা

TTL খুব গুরুত্বপূর্ণ: - TTL বেশি হলে → পুরনো (stale) content serve হতে পারে - TTL কম হলে → বারবার origin server থেকে fetch করতে হবে

Balance করা জরুরি

3. CDN Fallback Strategy

CDN কখনো কখনো downtime এ যেতে পারে। তখন: - Application যেন detect করতে পারে - প্রয়োজন হলে origin server থেকে সরাসরি resource নিতে পারে

Production-grade apps এ fallback থাকা উচিত

4. CDN Cache Invalidation

TTL expire হওয়ার আগেই যদি content update করতে হয়, তখন:

Option 1: CDN Invalidation API

- CDN provider এর API ব্যবহার করে cache remove করা

Option 2: Object Versioning (Best Practice)

- URL এর সাথে version যোগ করা

উদাহরণ:

```
image.png?v=2
```

নতুন URL মানেই নতুন cache

সংক্ষেপে মনে রাখার মতো পয়েন্ট


- CDN শুধুমাত্র Static Content এর জন্য সবচেয়ে কার্যকর
- Dynamic content সাধারণত origin server থেকেই আসে

- Large-scale, global user base থাকলে CDN প্রায় অপরিহার্য
-

উপসংহার

CDN আধুনিক ওয়েব অ্যাপ্লিকেশনের একটি অত্যন্ত গুরুত্বপূর্ণ অংশ। এটি: - Performance বাড়ায় - Cost optimize করে - User experience উন্নত করে

Proper configuration ছাড়া CDN ব্যবহার করলে benefit কমে যেতে পারে, তাই **TTL, cost এবং fallback strategy** ভালোভাবে পরিকল্পনা করা উচিত।

 Interview / Exam / System Design preparation এর জন্য এই নোট খুবই গুরুত্বপূর্ণ