Types of Cache

**1. Application Server Cache** The user’s request will be stored in this cache and whenever the same request comes again, it will be returned from the cache.

**2. Distributed Cache:** Each node will have a part of the whole cache space, and then using the consistent hashing function each request can be routed to where the cache request could be found. Easily increase the cache memory by simply adding the new node to the request pool.

3. **Global Cache:** A single cache space and all the nodes use this single space. Every request will go to this single cache space

**4. CDN** (Content Distribution Network): Manages servers that are geographically distributed over different locations. Stores the web content in its servers. Attempts to direct each user to a server that is part of the CDN so as to deliver content quickly.

**Disadvantages of using Caching**

- Data inconsistency

- Cache eviction issues

- Additional complexity

**Cache Invalidation Strategies**

- time-based expiration

- invalidate inaccurate or obsolete data

**Cache eviction policies**

- Least Recently Used (LRU)

- Least Frequently Used (LFU)

- First-In-First-Out (FIFO)

**Cache Design patterns**

**1. Cache-Aside (Lazy Loading) Pattern:** Application first looks or data in the cache. If the data not found, it fetches it from the main source, stores it in the cache, and returns it to the user.

**2. Write-Through Pattern:** Ensure that whenever new data is added or updated, it is written to both the main storage and the cache, keeping them synchronized

**3. Write-Behind (Write-Back) Pattern:** First writing to new or updated data to the cache, and then updating the main storage at a later, more convenient time.

**4. Cache Invalidation Pattern:** When data in the main source changes, related information in the cache is removed to present the use of outdated or incorrect data.

**5. Refresh-Ahead Pattern:** Proactively updating data in the cache before it becomes stale or outdated.

**6. Read-Through Pattern:** Read-Through Pattern retrieves data from the cache. If the data is not present, it automatically fetches it from the main storage and adds it to the cache.

Use Cases of Caching

* Web Page Caching
* Database Query Results
* API Responses
* Session Data