System Design 3

Designing Dropbox

What to be designed?

1. Storing files in the cloud allows users to store their data on remote servers.
2. These servers are usually maintained by cloud storage providers and made available to users over a network (usually via the Internet).
3. Let's design a file hosting service such as Google Drive or Dropbox.

Feature Requirements

Primary Features:

1. Users should be able to upload their files / photos and download them from any device.
2. Our service should support automatic device synchronization, i.e. it should get synchronized on all devices after updating a file on one device.
3. The system should support storing up to a GB of large files.
4. Users must have the option to share files or folders with other users.

Secondary Requirements:

1. We need ACID-ity. It should guarantee atomicity, consistency, isolation and durability of all file operations. Our system should support the editing offline.
2. Users should be able to add / delete / modify files while offline, and all their changes should be synchronized to remote servers and other online devices as soon as they come online.

Extended Features:

1. Versioning

TO DO : Follow the below framweork to design System. Mandatory upload of hand drawn photos of design.

Capacity Estimation and Constraints

What will your estimations of scale? As a system design student you should be able to make intelligent assumptions based on real life systems.

System APIs

Database Design

Basic System Design and Algorithm

Data Partitioning and Replication

Caching

Load Balancing

Handling Edge cases in given system

Functional Requirement

Primary Features:

5. Users should be able to upload their files / photos and download them from any device.
6. Our service should support automatic device synchronization, i.e. it should get synchronized on all devices after updating a file on one device.
7. The system should support storing up to a GB of large files.
8. Users must have the option to share files or folders with other users.

Secondary Requirements:

3. We need ACID-ity. It should guarantee atomicity, consistency, isolation and durability of all file operations. Our system should support the editing offline.
4. Users should be able to add / delete / modify files while offline, and all their changes should be synchronized to remote servers and other online devices as soon as they come online.

Extended Features:

2. Versioning

Non Functional Requirement

- Consistency of File
- Availability
- Reliability
- Scalability

Capacity Estimation

Nature of the Application

- Read and Write Heavy → 5 : 1 ()
- Assume we have 500M over all Users and
- 100M daily active users.
- Retain files for 5 Years
- On average each user use 3 devices(Needed for Synchronization)

Storage

Let's assume 500M users on average 100 files each of average size 1MB

500M users x 100 files x 1 MB = 10 Billion Files * 10 MB each = 10 PB

Bandwidth

Our application is read and write heavy application. There are relatively equal number of uploads and downloads

1 M users on daily basis we need to handle the bandwidth of 1 M connections per sec.

API Design

Upload_file(userId, filePayload)
Download_file(userID, fileID)

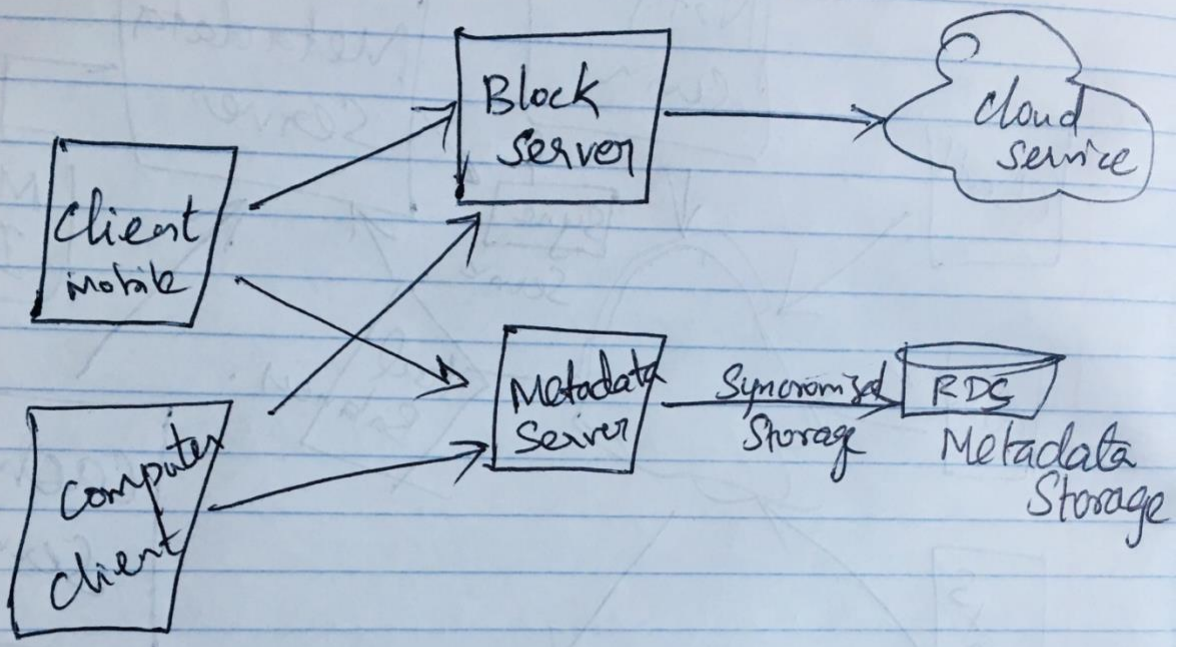
High Level Design

Client → Sends request to the Chunk Server to upload file to the cloud storage.

Meta data Server will be updated with the new file addition and deletion.

Meta Data will be synced to Cloud Relational DataBase.

Highlevel Design for DropBox



Synchronization Service

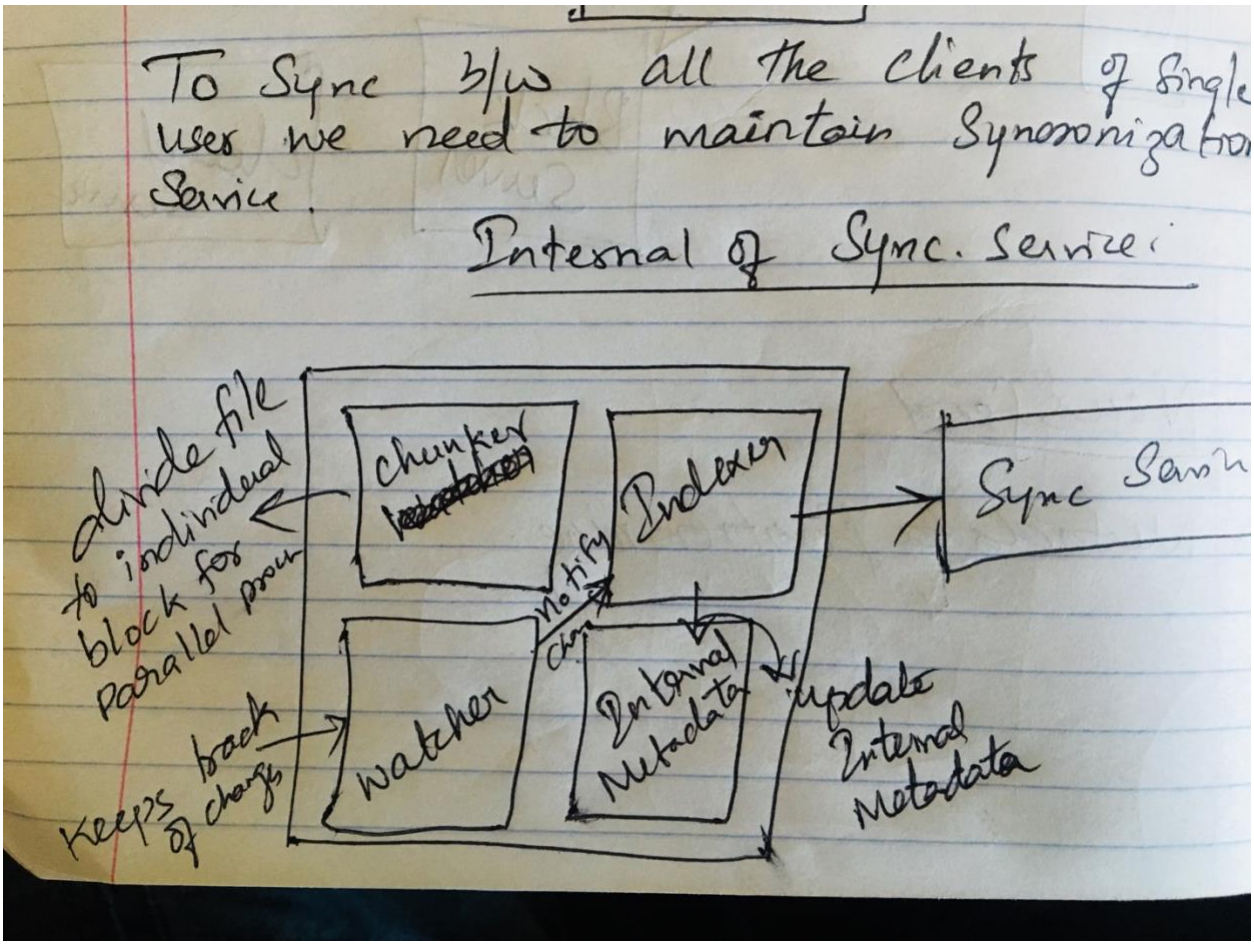
Sync service has 4 major components to sync newly added file

1 → Chunker
Chunks large file to individual blocks of chunks or equal size for parallel processing on the file.

2→ Watcher
Keeps track of the changes done in existing file. If the change is made to existing file Watcher will update only the blocks which are changed.

3 → Indexer
Indexer will update the internal metadata of the file and also sends the trigger to Synchronize the data for the individual user.

4 → Internal Metadata
Internal Metadata is to track the changes made to the file. Only most recent changes will be synced to the Sync service.



Relational Database is used to store the Metadata of the file in cloud for each client.

Over all architecture

Multiple clients upload and download files for same user. Cache server can be added to cache most frequently accessed files. Meta data will helps us to keep tracks of files changed

Synchronization service will be event driven architecture which adds new changes or upload is done by any client all the active clients will be refreshed based on PUB/SUB paradigm.

Below architecture shows the complete picture of Client uploading the file to Cloud Storage.