Core Features:

- User should be able to upload/download, update and delete the files
- File versioning (History of updates)
- File and folder sync

Traffic:

- unique users
- request per day with lots of reads and write.

Constrains:

- Bandwidth and space utilization
- Latency or Concurrency utilization

Client Components:

- Watcher is responsible for monitoring the sync folder for all the activities performed by the user such as create, update, or delete files/folders. It gives notification to the indexer and chunker if any action is performed in the files or folders.
- Chunker break the files into multiple small pieces called chunks and upload it to the cloud storage with a unique id or hash of these chunks. To recreate the files these chunks can be joined together. For any changes in the files, the chunking algorithm detects the specific chunk which is modified and only saves that specific part/chunks to the cloud storage. It reduces the bandwidth usage, synchronization time, and storage space in the cloud.

- Indexer is responsible for updating the internal database when it receives the notification from the watcher (for any action performed in folders/files). It receives the URL of the chunks from the chunker along with the hash and updates the file with modified chunks. Indexer communicates with the Synchronization Service using the Message Queuing Service once the chunks are successfully submitted to the cloud Storage.
- Internal database store all the files and chunks information, their versions, and their location in the file system.

Components:

- Metadata Database: he metadata database maintains the indexes of the various chunks. The information contains files/chunks names, their different versions along with the information of users and workspace.
- Message Queuing Service: The messaging service queue will be responsible for the asynchronous communication between the clients and the synchronization service.
    - Ability to handle lots of reads and writes requests.
    - Store lots of messages in a highly available and reliable queue.
    - High performance and high scalability.
    - Provides load balancing and elasticity for multiple instances of the Synchronization Service.

    Services:

        - Request Queue: This will be a global request queue shared among all the clients. Whenever a client receives any update or changes in the files/folder it sends the request through the request queue. This

request is received by the synchronization service to update the metadata database.

- Response Queue: here will be individual response queue corresponding to the individual clients. The synchronization service broadcast the update through this response queue and this response queue will deliver the updated messages to each client and then these clients will update their respective files accordingly. The message will never be lost even if the client will be disconnected from the internet (the benefit of using the messaging queue service).

- Synchronization Service: The client communicates with the synchronization services either to receive the latest update from the cloud storage or to send the latest request/updates to the Cloud Storage.
- Cloud Storage