Functional Requirements:

1. Users are able to sync (upload/download) the changes like create/delete/modify file/folder under a local folder to a remote drive.

Non-Functional Requirements:

1. Efficient utilisation of bandwidth and space.

API Contract between Sync Client and API Server

1. User register/unregister
   PUT /ws/v1/drive/register

   userId

   DELETE /ws/v1/drive/unregister

2. Sync/unsync the directory

   PUT /ws/v1/drive/sync

   userId, path

   DELETE /ws/v1/drive/unsync

3. get/post the metadata

   GET /ws/v1/drive/metadata

   userId, path

   ADD, /path/newfile1

   DELETE, /path/newfile2

   MODIFY, /path/existFile3, chunk2

   APPEND, /path/existFile4, chunk3

4. download/upload modified/added chunks

   GET /ws/v1/drive/data

   chunkId

   Content-type: multipart/form-data

POST /ws/v1/drive/data

Data Study

RDBMS - sharded using userid hash and driveid hash

User Table: userid, name, address

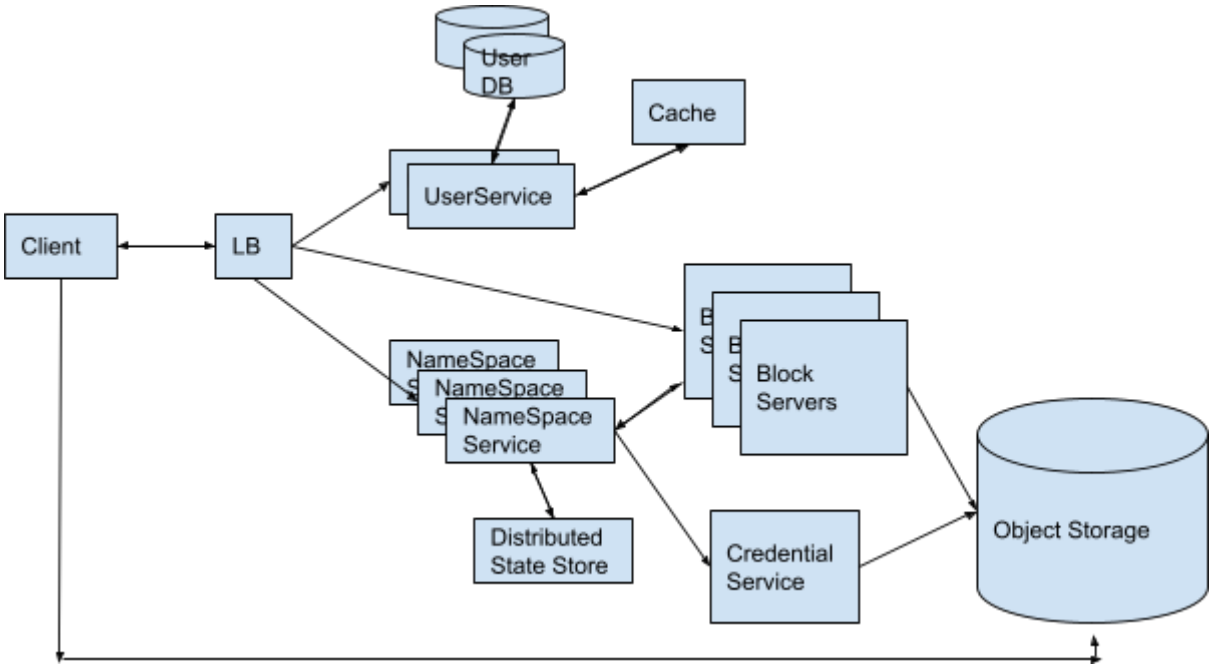
Drive Table: driveId, groupName

For Each Drive - FileSystem Hierarchy (in-Memory either through HashMap, INodeDirectory-> List<Inode>), Metadata changes with versioning

For Each File in a Drive -> Chunks -> Block Locations

Namespace Service -> Drive -> Namespace (FileSystem, Chunk Locations)

In-Memory (Leader, Follower, Follower) + WAL



Block Servers or S3 (Object Storage)

Read Path:

Write Path: