Social Media App

This doc specifies the task which peri sir wanted all of you to do on June 3rd. **There would be no mentors or faculty to help to you during the time. You can start from level 1 task and keep building towards level 2 and level 3, if you are confident you can start from level 2.(This is what we expect all students to complete by tommorow end)**

Task :

A File Store which supports sharing of files and its associated comments. Files and its associated metadata (comments , free space etc) should be stored in a 100mb filesystem.

Objectives involved:

1. Users
2. Posts
3. Comments
4. Likes

Aspects :

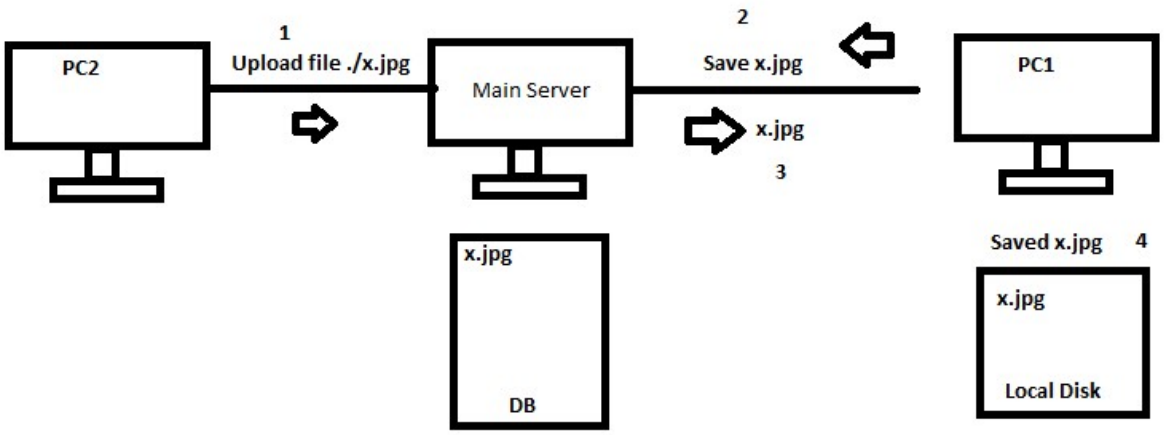
- File System should be able up support upload ,download and deletion of images.
- Upload means copying of the file data from disk into your file system (which is 100mb empty binary file). Downloading means getting that data back from file system to disk.
- Each file(image, video…) can have multiple comments max of size 100Bytes.
- All comments addition,deletion,viewing will be made only through console.
- Files will be copied into file system from disk ,when user specifies upload file to file system.

A brief view of assigment:

Create multiple USERS with required properties, User can create an user, create a post (basically a file(ppt, video file, an image, ….)) , update a post, delete a post, comment on posts, like a post, message other users, see all received messages.

Multiple users can connect though network and can sync the database to their machine, any user can do all his/her operations through network.

An example of posting an image by some user from one pc and saving it to local disk by other user in other pc.



A Deep View:

1. Memory Allocation:

   Create your own memory allocation function, where you assign some memory when every needed in your code.

2. Data persistance:

   Save the data into a some file. Bonus points If you integrate B+ trees. Use Bit vector(link) for storing the data else use your own file system.

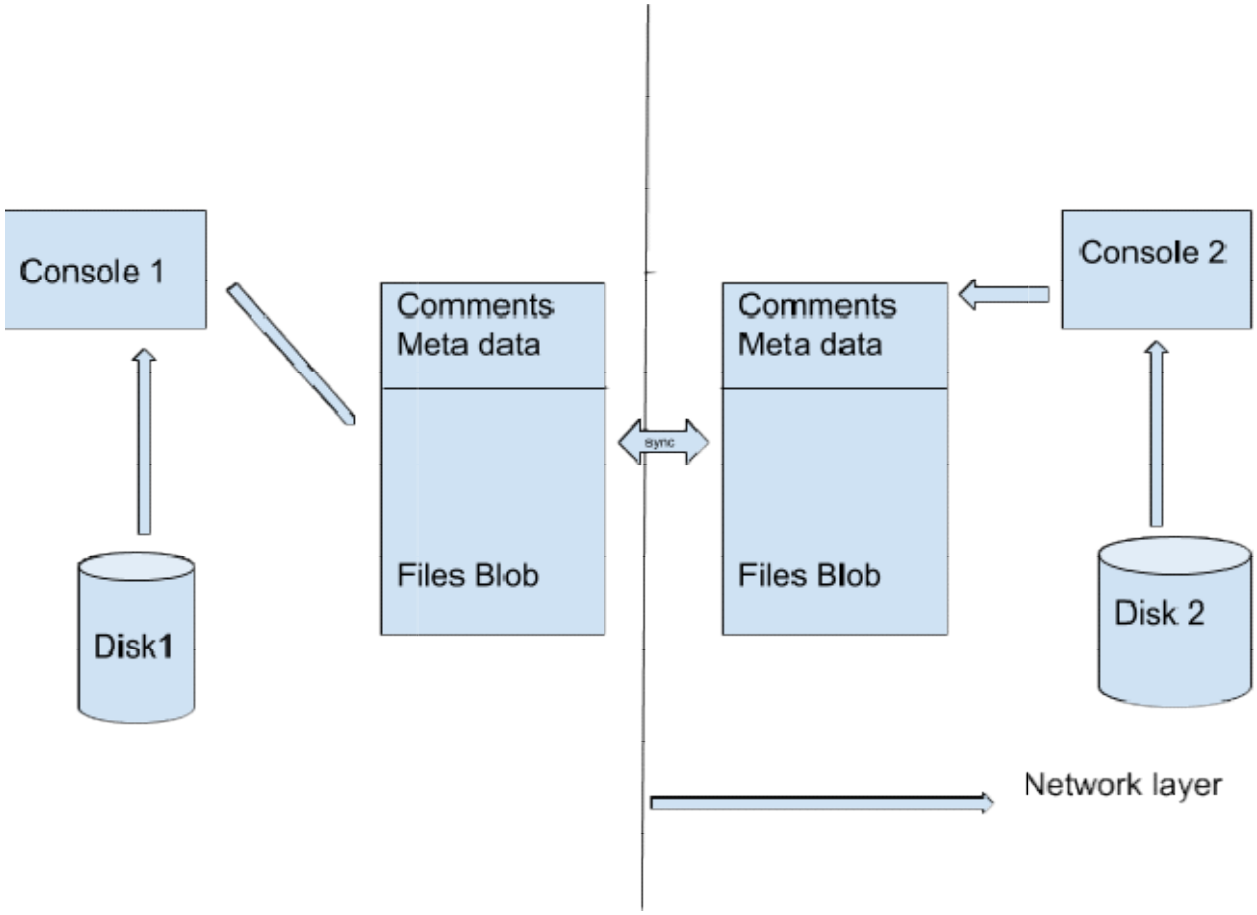
   Grade will be allocated based on your approach and code.

3. Threads for concurrency:

   Different users will use the same server at same time, allocate a thread for handling every new connection. You can also use threadPool for performing all these concurrent tasks.

4. Network communication:

   Use sockets for communicating over network, comment , post , create user in server DB.



Operations:

- ➔ If some user from pc1 sends a command "**sync**" to pc2 then the entire DataBase in pc2 needs to be stored in pc1 through network.
- ➔ Commands on user data:
  - Create User
  - Remove User
  - Update User
  - Get User
- ➔ User operations:
  - Post post_name "./Path"
  - Comment on post
  - Like post
  - Remove Post
  - Update Post
  - Save Post (Downloads that particular file into local disk)
  - Message User
  - Get all messages

Grading:

Level 1 : Local workflow : (Estimated Time : 5 Hours)

- Given a 100mb empty binary file which will be acting as a file system.
- Student will write a console app , which supports following operations.
- **Upload file**: Copies **file** user specified from disk to fs blob (Binary copy)
- **View file**: Lists all file names uploaded by user.
- **Download file** : Downloads a particular **file** from fs to disk . It is like if he uploads a image using console, and he deletes actual copy ,he can get that copy back again by downloading from fs.
- **Delete file :** Deletes the file from the filesystem
- **Add Comment :** Adds comment for an file.
- **View comments** : Views comments of a particular image (Not necessarily the same):
  Image 1 : First comment
  : Second comment etc
- **Delete comment** : Deletes a particular comment on the file.
  The local workflow should be able to work on a single system .

Level 2 : Multi-Machine work flow(Estimated Time: 6Hours) :

- There will be two machines or 2 consoles . which will communicate with each other using sockets. Here the program will be acting both as a server and client.
- Each console will have its own 100mb FS store
- The main concept here is both FS should stay in sync at all times.
- User 1 uploaded a file “A.png” in console 1 (FS1) , User 2 should be able to get that file automatically in his File store.(FS 2). If user1 console is connected to user 2 console.
- User 2 can add comments to the file “A.png” which user 1 should be able to see.
- All this “**Syncing”** should happen automatically in the background thread. (Initially each client can call **Sync manually** to make this sync happen(Single threaded) , but later this should be modelled using threads and automatic syncing should happen in background)

Sample Output of Multi machine work flow :

-> User 1 opened the program in console1. Uploaded A.png file .
-> User 2 opened the same program in console2. (It should communicate to already opened console 1 through sockets.) .
-> User 2 clicks on Sync through which he will get all data from user 1.

-> User 2 can now download A.png to his disk.
-> User 2 can add comments to the file A.png ,which on syncing by user 1, user 1 can see those comments.

-> Essentially both should be notified of changes, additions automatically .
-> Initially you can manually call sync but as an end goal **The syncing should happen automatically .**

Level 3 : Implementing Cache. Cache size is 2KB. (Estimated Time : 3Hours)
-> Store the comments into in memory cache , and user can fetch the comments from cache itself . If comments for image1 are read once , it should be stored in cache and all subsequent reads should fetch it from cache and not from file system. Read about write back and write through policies and how do you manage updating, deletion of comments both in cache and file system so it stays consistent. How do you free cache for storing new comments etc.

-> Also code for how to handle caches in multi machine scenarios. Come up with a good design and code it.

Submission :

-> By midnight on June 3rd , Write a 1Page Doc of what all you did, and send it to your team mentor and mrndtas@gmail.com .

-> Also upload the code you have written in your google drive folders .

-> Each of your code will be personally checked and graded by your team mentors and they will tell us whether you have done well in systems track or not.