AOS Assignment 3 Report

Peer-to-Peer Distributed File Sharing System

Student Details

- **Name**: Shubham Raut
- **Roll Number**: 2024201019
- **Assignment**: AOS Assignment 3

Completed Features

1. Tracker Module

The **Tracker** is responsible for managing and synchronizing client and group information, ensuring that all online trackers are synchronized.

Key Features Implemented:

- **Tracker Initialization**:
    - Command: `./tracker tracker_info.txt tracker_no`
    - The tracker is started using the tracker_info.txt file, which contains the IP and port details of all trackers in the system.
    - The tracker number (e.g., `tracker_no`) ensures that the correct instance of the tracker is started.

Files Implemented:

- `tracker_info.txt` – Contains the IP and port details of trackers.
- `tracker.cpp` – Handles all tracker-side functionalities like syncing with other trackers and managing peer information.

---

2. Client Module

The **Client** module allows users to interact with the system by creating accounts, managing groups, and downloading/uploading files.

Key Features Implemented:

- **Client Initialization**:
    - Command: `./client <IP>:<PORT> tracker_info.txt`
    - The client is run with its IP and port, and the tracker information is provided via the `tracker_info.txt` file for the client to connect to the tracker(s).
- **Account Creation**:
    - Command: `create_user <user_id> <passwd>`

    - This command allows a user to create an account with a unique `user_id` and password. The user information is stored on the tracker to allow future login attempts.
- **User Login**:
    - Command: `login <user_id> <passwd>`
    - After account creation, users can log in using their credentials. Successful login establishes a session for further group management and file sharing activities.
- **Group Management**:
    - **Create Group**: Command: `create_group <group_id>`
        - Users can create groups to share files with other members.
    - **Join Group**: Command: `join_group <group_id>`
        - Users can request to join existing groups.
    - **Leave Group**: Command: `leave_group <group_id>`
        - Users can leave groups they no longer want to participate in.
    - **List Pending Join Requests**: Command: `list_requests <group_id>`
        - Group owners can see the pending join requests for their group.
    - **Accept Group Joining Request**: Command: `accept_request <group_id> <user_id>`
        - Group owners can accept user requests to join their group.
- **File Management**:
    - **List Groups**: Command: `list_groups`
        - Users can view all the available groups on the network.
    - **List Sharable Files in Group**: Command: `list_files <group_id>`
        - Users can see a list of all the files shared within the group.
    - **Upload File**: Command: `upload_file <file_path> <group_id>`
        - Users can upload files to a specific group for sharing with other group members.

Files Implemented:

- `client.cpp`– Contains all client-side logic and interactions with the tracker.
- `tracker_info.txt` – Used to connect the fetch ip and port to the tracker.

Additional Information

Authentication and Security:

- Basic authentication is implemented for user login and account creation. Passwords are securely stored and checked by the tracker before granting access to the client.

Error Handling:

- **Input Validation**: All input commands from the user are validated to avoid crashes or invalid operations.

- **Connection Handling**: Robust mechanisms are in place to handle connection failures between clients and trackers. If a tracker goes offline, the client will attempt to connect to another tracker if available.