Programming Language: Java

Phase 1: Registration and Deregistration (UDP)

Goal: Set up communication between server and peers using UDP. Peers should be able to reg, dereg and be recorded by the server

Server Tasks: (2 ppl)

1. Create UDP socket
2. Handle incoming messages (Benny)
3. Send responses

Peer Tasks: (2 ppl)

1. Define message formats and parsing functions (Gunkeet)
2. How to store data/store data
3. Testing

Files: server.java (listening udp), peer_udp.java (reg and reg)

Phase 2: Backup & heartbeat management (UDP + TCP)

Goal: Allow a peer to backup a file to other peers and send regular heartbeats to the server

Server Tasks:

1. Receive backup req
2. Select peers with with enough storage capacity
3. Send storage reqs to the storage peers
4. Receive confirmations and update data
    a. Backup ack
    b. Storage ack
5. Storage heartbeat monitoring
    a. Marks peers inactive if timeout

Peer tasks:

1. Send backup req
2. Receive backup plan
3. Split files into chunks
4. For each chunk -> open tcp connection to designated peer
5. Send chunk and data
6. Wait for chunk_ok or chunk_error, resend if needed
7. When done-> send backup done ack

-Implement storage peer tcp listener to receiver chunks and send acks
- start a heartbeat thread that sends status to the server every few seconds

Phase 3: Recovery and Integrity Validation (TCP)

Goal: Allow a peer to recover previously backup file and verify its data

Server Side:

1. Handle Restore req's
2. Look up storage peers and reply with restore_plan ack
3. Receive restore ok or restore fail ack
4. Detect failed peers via heart beat and replicate req to reallocate chunks

Peer Side:

1. Send restore_req to server
2. Receive restore plana
3. For each chunk, open TCP connection to storage peer and request
4. Receive chunk+ binary chunk
5. Merge chunks back into original file
6. Verify data
7. Send restore_ok or restore fail

we should be able to send/restore .txt, .pdf, .jpg, .mp4 files