Requirements

1. Unique File names on the same datanode (checksum)
2. Client download
3. run on diffrent machines
4. Client succes message (see arrow 6 in uploading file in the pdf) - This will make the client as a server

Enhancements

- For the datakeeper that should upload the file, we can take the one with larger available storage _ 0.7 + number of available ports _ 0.3. Now we are taking random one
- download in parall

Bugs

- file deleted crush

Atef tasks:

- images:
  - to build `docker build -t datanode`
  - to run `docker run -d -p <host_port:container_port datanode container_port>`
- checksum

tests

- large files
- one datanode down in the middle of upload or download
- multiple user upload on the same time
- same file name upload

Questions

- in replication in data keeper node i use the tcp like the client is that correct

Docker

- first you need to build the 2 images (master and data)
  - do that by running the command `docker build -t datanode -f DatanodeDockerfile .` to build the data node image
  - to build the master node image run `docker build -t masternode -f MasternodeDockerfile .`
- now you can start the containers
  - first starting the master container using `docker run --name master1 -p 9090:9090 masternode 9090` 9090 is the port the master will be using. You can add `-d` flag to detach the container terminal from host terminal
  - now starting the data containers so you can use `docker run --name data1 -p 2025:2025 datanode 2025 9090` 2025 is the port for the data node and 9090 is the port for the master

    node. Same as before you can use the `-d` flag
    - Last you can now use `go run .\client\client.go upload example.mp4 9090` to upload a file. Note that 9090 is the master port
- You can check the files uploaded in the container using `docker exec -it <container_name_or_id> /bin/sh` to log into the container with the terminal then run linux commands like `ls storage` to check the uploaded files. Use `ctrl + d` to close the container.
- You can use `docker container stop $(docker ps -a -q)` to stop all running containers
- You can use `docker container rm $(docker ps -a -q)` to remove all stopped containers

Network

- `ipconfig` to see the machine's IP
- `CTRL + R` then `wf.msc` then enable rule `File and Printer Sharing (Echo Request - ICMPv4-In) Private, Public`
- use `ping` to make sure the connection is established
- add data address & master address in datakeeper code line 62 & 63
- may need to change timeout

commands

- `protoc --go_out=. --go-grpc_out=. --go_opt=paths=source_relative --go-grpc_opt=paths=source_relative dfs.proto`
- `go run .\master\master_tracker.go`
- `go run .\datanode\data_keeper.go local 1 5000 node1`