CPS 706 Course Project:

A Peer to Peer Shared Photo Album

DOCUMENTATION

# GROUP # 56: PORT # 20650

## **Jonathan Lam** (500792483) – Section – 03

## **Judel Villardo** (500768474) – Section – 05

## **Noah Mifsud Lattari** (500760404) – Section – 03

# User Documentation

*Please note that since the project is incomplete, I will provide a theoretical and actual procedure to run this program.*

### **Theoretical procedure**

On each machine perform the following:

*Let port = 20650*

1. Compile and run PSrv.java: *javac Psrv port*
2. Compile and run PCli.java: *javac PCli port*
3. Compile and run DirectoryServer.java: *javac DirectoryServer ID port*
   1. Where ID is a number from (1 – 4) inclusive
4. Set up Directory Server:
   1. Given the prompt: *“Enter predecessor, successor IP”*, enter the IP addresses associated with predecessor and successor Directory servers
   2. Given the prompt: “Set Primary Directory Server”, enter the IP address associated with the Directory Server with ID = 1
5. Run Commands through P2P Client Command Line:
   1. “Init” to setup initial connection and return a list of IP addresses for all the Directory Servers respectively.
   2. “query *contentName*” to retrieve the IP Address or list of IP Addresses for which hold the image in question.
   3. “get *contentName* *IP”* to retrieve image given by contentName at specified Peer *IP*.
   4. “exit” to terminate all connections.

### **Working procedure**

*Let port = 20650*

1. Compile and run PSrv.java: *javac Psrv port* on machine 1
2. Compile and run PCli.java: *javac PCli port* on machine 2
3. To retrieve image:
   1. On Client Command Line, pass in the IP of Server given by the output on the CLI of Server.
   2. When image is successfully returned, a successful response will be provided on the Client CLI and the image will be downloaded onto the “**img\_c”** folder.

# System Documentation

Within this section, we will be discussing the implementation of primarily the Directory Server and P2P Server as the P2P Client was unfortunately unable to be completed fully as per requirements.

First, the P2P Server implements as TCP connection through the use of ServerSocket, DataInputStream and DataOutputStream Java packages. A sever-socket is initialized with the provided port number and continually listens in on this port for a client-connection from a P2P Client. Once the connection is established, we can utilize the input and output stream to pass information between the P2P Client and P2P Server. For example, the P2P Client will generate a TCP Connection with the same port, connect to the P2P Server and provide an image name in the form of a string that is serialized and passed through the network. The IP address of the P2P Server is provided by the Directory Hash Table which will be discussed in further detail later. Once the P2P Server has received the file name, it will return the image, if found, in the form of a BufferedImage object, packaged and sent through the Output stream on the P2P Server side. The result is that the image, if found, will be retrieved and downloaded locally on the P2P Client end.

Second, Directory Server class handles the routing of all content that is stored within each peer. Where a peer stores an image locally and is available to be transferred to other peers on the network, that peer will update their personal Directory Server located on their local machine containing the file name as well as the IP Address associated with its location in the form of a HashMap. First, to establish the location of the Directory Servers, the P2P Client will be manually notified of the IP address associated with the Directory Server that is designated with ID = 1. An Initialize function, Init, will connect the P2P Client with the #1 Directory Server for which will initiate a chain of TCP Connections between each Directory Server. Each Directory server, since they are arranged in a circular fashion, will be aware of its predecessor and successor. As such, Directory Server #1 will contact Directory Server #2 and then to Directory Server #3 so forth until the connection returns back to Directory Server #1. At this point, the Directory Server #1 will return a list of IP Addresses for which signify all four Directory Services respectively through UDP socket to P2P Client. Other functionalities of the Directory Server include updating the HashMaps if certain peers either gain or remove content files and as such are no longer available to be retrieved. Once the P2P Client has received the list of addresses, the P2P Client can determine which P2P Client on the network has the required file through Hashing the content name (using Hasher.java) and further retrieving it through TCP.

Although we were unsuccessful in implementing the P2P Client to use the Directory Server, the requirements and functionality of the Directory has been implemented as seen in *DirectoryServer.java*. However, through the presentation, we will show that we are still able to retrieve and download a physical copy of the photo from one machine to another by providing the IP Address of the Server for which holds this photo.