Client:

1. Upload data:

* Break data into pieces (each piece has at most 512kB).
* Create a order.JSON file that saves these information: name of data, names of files, types of datas, some tables that saves the order of pieces (each table save for one file in case there are many files in data).
* Upload the .JSON file to the server.
* Receive the IP address and port from server to transit these data file for peers in networks. Choosing to upload to the peer that have the fastest speed.
* Encode the files (optional).
* Update for the server before changing to another peers or disconect.

1. Download data:

* Upload the file.JSON to the server, then get a table about positions of each file in networks.
* For each pieces, choosing to connect to the peer that has the fastest speed if there are 2 or more peers that have the same speed, choosing the one that has most pieces.
* Dowload the order.JSON from server.
* Merge these pieces and name the complete files.
* Delete order.JSON.
* After finishing downloading the necessary files. If the clients is still online, then continue downloading some pieces from other data pieces and updating for the server.

Server:

1. Upload data:

* Get the order.JSON from client which want to upload data.
* Scan all of peers that are online in network and send a list of peers with speed information. After a specific time, send again the list of peers to update the speed information.
* Create a new database to save order.JSON and list of pieces. Add this database to a hash table and receive a key for it.
* Receive the update information from clients.
* While one peer is receiving the peering, it also upload these pieces to another peer in network that doesn’t have. This proceduce will stop when all peer have at most a particular number of pieces and all pieces of data are on network. Server will control this procedure.

1. Download data:

* Get the key from data.JSON and send to client the order.JSON and pieces.JSON(includes information about where each pieces).