ERSEL CELAL EREN

1901042673

CSE 344 FINAL PROJECT REPORT

In this project we are asked to implement a simplified version of Dropbox. To handle this task, I developed two program, Server and Client. I handled communication between server and client via “socket”. Client can connect through socket to server by localhost or specific IP address when providing it. After running server program, it creates thread pool with the name of client_pool. It is a struct that holds thread_id and socket variable.

According to my design, after connection is done server sends folder structure and contents of this folders to client. In this way, client synchronize its folder with server.

After the assignment was given, I tried to solve a problem for a long time. I couldn't connect to 'socket' using both WSL and VirtualBox. When I discussed the matter with the assistant of the course, I could not come to a conclusion. It took a long time to solve the problem, even though I tried ways such as making changes to the modem interface and turning off the firewall. In addition to these, I had to install Dual Boot Ubuntu. This also took quite a long time. Since I started my homework after a long time, I had limited time.

For this reason, I could not cultivate the following features in the assignment I submitted.

- Signal Handling

- Log File

- Two Way File Transfer

When server and client connected and client synchronized with server, Only the changes made on the Client side will be detected and the effect of the change will be seen on the Server side. The program does not support changes on the server side.

Let me explain how does the code work:

After client synchronized, client program stores its own file structure in a list and file modification times. In an infinite loop, it continuously check any changes. First, it gets file list of current and compare with older version. If there are additional files in current list, it sends a message to server with socket that notifies new file is added. If no file is added, it does the same thing for delete. And lastly, it controls last modification times of each file in client directory. If there is a difference between first version of times and current version, then program sends 'Update' message to server. In the case of there is no update, it starts over again checking.

When a file is created, "!_CREATE_!" message is sent to server. Server get this message, it starts waiting size,name, and content of new file. Then it creates that file in its own folder. Client adds this file's path into folder current paths list that it use to check any changes.

When a file is deleted, "!_DELETE_!" message is sent to server. Server get this message, it starts waiting name of the file. Then removes it from folder. Also client removes it from folder current paths list that it use to check any changes.

Same as update, when server gets "!_UPDATE_!" message, a very similar procedure to "Create" is followed. After sending new content of updated file, the list that client holds for last modification time is updated also with the new last modification time.

I tested program in same computer with multiple terminals and tested from different computers. For both situation, connection is established and program works in same way.

Tests:

- when you update content of a file in client folder, it will be updated on server

- when you delete a file in client folder, it will be deleted from server

- when you create a new file in client folder, this file will be created at server

Run the program with absolute path :

```
./BibakBOXServer /home/user/documents/serverfolder 5 8080
```

```
./BibakClientClient /home/user/documents/clientfolder 8080 127.0.0.1
```