DeadDrop

Faran Ahmad
2013CS10220

Kartikeya Gupta
2013CS10231

Prateek Kumar Verma
2013CS10246

COP290: Design Practices

Contents

1 Objectives

We have to build an on-line file management system "Dead Drop" . A server machine maintains the files of multiple users. The user should use a simple desktop application to login into the system. The content of user's account should remain synced with the server.

2 Overall Design

1. We will begin with creating different sub components like a File Transferring System, Credential Verifier, GUI part.
2. Once the components are ready, we will Link this to the network and get basic functionality working on the local-host.
3. Once the local interface is ready, we will take this to the web portal. We will use a server to store data and users will have to send queries to it
4. Once the backend and front end is complete, we will link the two together.

3 Sub Components

3.1 User Verification

Listing 1: Class Parameters for User

```
1 class User
2 {
3     private:
4         std::string UserName;
5         std::string PassWord;
6 };
```

Listing 2: Class Parameters for UserBase

```
1 class UserBase
2 {
3 private:
4     std::unordered_map<std::string, std::string> UsersList;
5 };
```

The User Base is a hash table in which the keys are user-names and the stored values are passwords. When the credentials of the user are to be verified, the key is looked up in the table. Inserting users is also achieved easily using this model. The features which we will be provided to the user will be to verify credentials, add new users and change password. On the server, the credentials will be stored in an encrypted file which will be decrypted by the server program.

3.2 Files of User

We will use boost library to detect changes in files. For each file, the path of the file and last modified time of file is stored in a database.

Listing 3: Class Parameters for File History

```
1 class FileHistory
2 {
3     private:
4         std::string FolderLocation;
5         int TimeOfData;
6         std::vector< std::pair<std::string, int> > FileTimeBase;
7 };
```

Folder Location is the path of the synced folder. The parameter "TimeOfData" contains the system time at which the data detection was done. This will be used to determine if the server or client side file is newer and then do changes accordingly. "FileTimeBase" is a vector of a string and an integer. The string is the path of the file and the time is the time at which the file was last modified.

3.3 Network Managing Part

TODO: SOCCER

3.4 GUI interface

The interface of the application would be designed using QtCreator. It would consists of the following mainwindows.

- User Login
  Running the application would display a user login window. The users can access into their accounts by entering their user name and password in the respective fields. It also provides options for new user to signup for a new account in dead drop. In case, the user forgets his password, he can reset it using 'forgot password' option. If the user name or password entered is wrong, a message box showing this message is displayed and the user can again enter the required informations to login to his account.
- USER FILES
  Once the login procedure is complete, the user is directed to a new window. It contains the list of files of the user, both on the client and server side. The following buttons would be provided on the window for various functions
  - *sync :-* Clicking this button would sync the files on the user and the client side.

- *Share :-* This button allows the user to share files with other users. Once the user has selected the files, clicking on this button would open a new dialog box, where the user can enter the name of the ones, with whom he wants to share the selected files. It also provide users with the options to set permissions for the shared files and folders.
- *Delete from local :-* This button can be used to delete the selected files on the client side.
- *Move To Drive :-* It can be used to move the seleted files on the client side to the server.
- *Refresh :-* If there is some transfer of files between local data base on the client side, refresh button can be used to display the present ...
- *Get :-* This button can be used to transfer the selected files on the server to the client side.
- *Delete from Server :-* This button can be used to delete the selected files on the server.
- *Exit :-* This button can be used to exit the application.

- Server Side User Interface
  The window on the server side displays the list of the online users. It also provides

4 Interaction amongst Sub Components

4.1 User Authentication

- Client TODO soccer
- Server TODO soccer.
  The network part mentioned above is linked with the user base file. The instruction to be performed is decoded to be a new user or credential verification. The data base of user names and passwords are accessed for this to take place and changes if needed are made accordingly to it.

4.2 File Transfer

- Files to Transfer TODO KG
- Transferring TODO soccer

4.3 File Sharing and Syncing

- File changes TODO KG
- File Sharing and Permissions TODO KG

- File Syncing TODO KG
- File Syncing over network TODO KG TODO Soccer

4.4 Front End and Back End

- User verification TODO TODO KG
- Add User Part TODO Faran TODO KG
- File Managing Part TODO Faran TODO KG

5 Testing Of Components

5.1 General Unit Tests

Listing 4: Class Parameters for Test

```
1 class Test
2 {
3     private:
4         bool verbose;                //Variable if test is to be conducted
5         std::string description;     //String description of the test
6         bool isPass;                 //Boolean if the test has passed
7         void PrintPassFail(bool);    //Prints the status of the test
8 };
```

We will use the aforementioned class "Test" to perform unit tests on the different files created. This will ensure that all the functions work correctly against some tests.

5.2 File Discovery

To test file discovery, a folder with different files will be used. The program will be run on this to obtain the list of files with their modified time and verified to check if it is in accordance with expectations. This will involve new files being created, files being modified and removed.

5.3 File Transferring

TODO SOCCER

5.4 UI Testing

TODO FARAN write about individual components

5.5 Overall Testing

TODO KG once above 2 are done

6 Extra Features

6.1 Keeping Files On-line only.

We will give the user an option to keep a file on the cloud only and not on his PC so that lesser space will be used on the users PC. The user can get these files or remove them from the cloud if he wishes.

6.2 Intelligent UI for server

TODO faran Write about showing transfer speed and stuff

6.3 Allowing Incomplete Downloads

Allowing the user to resume downloads of files from where he left off if the connection with the server breaks in the middle.

6.4 De-duplication