1 Beginning: Hello everyone, my name is alex and this is my partner. We design a GUI-based chat system on top of the UP3 and integrate some interesting features into it.

2 Motivation: Originally UP3 was very inconvenient to use because all the input had to be done in the console and the user input was in the form of texts, it was like you were using a computer from the last century with the most complicated operating system because you even had to remember the commands for the corresponding functions. For example, if you want to link a user, you even have to manually enter c xx. Or if you want to query some information, you have to enter the relevant command, so the threshold for a chat system without an interface is too high. And this is why we design a GUI interface. The existence of the GUI interface greatly simplifies users’ operating procedures. You query the chat history, do the poem search only with a simple click. So the importance of the GUI cannot be overstated. On top of that, designing a GUI also requires a comprehensive understanding of both the codes of client side and server side and the thorough knowledge of online(TCP) programming so that we can learn the most from this project.

Next, we want to give you a quick overview on the interface of the home page of the chatting system.

3 This is the home page designed when the user has logged in but not connected with anyone else. Here, we have some basic user-oriented functions like the electronic clock, basic user information, fileclouding function and some local widgets like music downloader, music player, and poem search, and calculator which I will go deeper into

4 And this is the home page when user is connected with other user or users, where we have additional features like fileTransferring, asychronous voice chatting features, and a clear presentation of the online information of the users, and room chatting information.

5 6 Ok, next will be some of the demo snippets of the functions of our chatting system.

7 The first feature would be login and register, So, for most chatting application like QQ, or Wechat, login and register will be the commonest while most important features to track different users’ information. We design a simplified login and register function which simulate the full set of user login and register process. Like here, we have formal validating policy as shown in the PPT. When the password doesn’t match with username, there will be error notes. And when the user is successfully registered, we save their username and password at the MySQL database in which username and their password are saved with a key-value mapping relationship. Here we use the database to mainly simulate the real-world way of saving users’ logging information.

8 The second feature is invitation function. Remember In our UP3, we can only connect to only one person. IN other words, when user A wants to connect with user B, we type in the terminal “c B” to connect, and the connected users cannot connect with other users like a chain, which is logically weird. So here in our project, we introduce the invitation function so as to enable flexible chatting environment. For example,.....

9 Next, filetransferring, which is also a common feature of chatting application. By totally rewrite the mysend and myrecv function, we can receive any kind of information from other users. As shown in the demo clip, we can target the user who wants to receive our file and the targeted users will receive the file at the bottom right and can download it right away.

10 Similar to filetransferring, we simulate the asychronous voice chatting function, which resembles wechat’s voice sending process. By using PyAudio, we collect the voice sample from microphone and save it to local application cache, and then send the cached voice file to the target user. Then the target user would receive the voice message and click and listen. When the user logged out, all the voice message and received file will be deleted from local application cache.

11 Next one is file Clouding. Due to the limit of local hardware storage, cloud service is becoming increasingly popular. In our project, users can back up their files online and the next time they went online, server will find his clouded files and provide download channel to him. Here is the demo, this is the first time user upload the file to the server. And the next time user logged in to the chatting system, the file he has uploaded will appear in this box.

1. Local music player, which is a local function that can access local music files and play them. We implement this by QMediaplayer, a powerful Widget in PyQt5. Here we can either play the music by double clicking or click the play button. We can also switch between different songs and their play mode. On top of that, we can also choose local directory where we save our music files.And the slider on the top enables us to control the playing progress.

13 The most exciting feature in our project would be this music downloader. We implement it with the idea of web crawling, which I will introduce in the tech analysis section. Here we can simply type in the name or the author of the song, then wait for a few second since there is huge amount of data being exchanged between client and the server and finally we get a whole list of searching results. Notice we can double click the song’s title and download them to the local storage.

14 Calculator is the first local function we implement, which is relatively simple. We implement this to get more familiar with the grid layout and button function of PyQt5.

15 Poem Search and Chatting history is quite simple, we just add a GUI to it

16 Offline games. We apply for the GUI design, so games will be our additional features. (补充)

17 Here is the source code of our project

In the source code, we divide the code into four parts. Initializers functions, front-and-rear interaction functions, chatting logic-implementation functions, and local utilities logic-implementation functions.

The first part is responsible for the building and initializing of the client socket and interface.

The second part mainly used for the the tracking of user’s behaviors on the interface. For example, if we type in username and password in the input box, when we click login, these front-and-rear functions will get triggered and send the message to the rear side of the project.

The third part is reponsible for the chatting utilities, which includes all the features in the demo.

The final part is for the local utilites. Like the implementation of local music player, calculator, poem search and chat history search.

18 In this project, when users are interacting with the chatting system interface, they are actually triggering the function inside the client class object with large numbers of APIs. For example, when we click the send button, the send\_my\_message function will be triggered to execute a set of commands to help us deliver the message onto the MessageRecvingBox of the target users.

Technically speaking, each button or label is bound to a set of functions to respond to users’ input. Then, OOP helps us integrate,( or encapsulate) massive amount of functions into a single class, making out codes logically clear and easier to digest.

19 Remember in the demo we have file clouding, file transferring and asynchronous voice chatting, these functions are all based on the implementation of file data transferring process. In our project, we rewrite the mysend and myrecv function to realize the transferring of all types of messages, with all magnitude of size. We know that when we transfer information online, we are actually sending streams of bytes first to the servers, then to the target users. But since there is no boundary between each file, message, and image, so how can the server differentiate between them. In other words, how does the server know when we have received this file and should move on to the next. So, we develop a general transferring protocol and it goes like this. We break the sending and receving process into three parts. Firstly, we send a sentinel information which clarify the binary length of the json dictionary to be sent to the server. Then the server know the binary length of the json dictionary to be sent. Next, we send the json dictionary and this json dictionary carries the basic information of the file to be sent like the filepath, filesize. Then the server extract the filename and filesize information out and be ready to receive the file. Finally, we send the file. To improve the running performance to avoid the sluggish performance when transferring large files, we send the file 1024 bytes by 1024 bytes. When the filesize is reached, the server know that the transfer of this file is over and will move on to recv the next file. This process clarifies the boundary between constant flow of data and make sure that the file is intact.

20 There are many ways to avoid using threading in implementing message recving process, however, we think Threading is the most powerful and interesting feature in programming since it could allow you to multitask. In our project, if we want to send a file while also receive important messages from others. Simply wait until the file is sent and then receive the message is kind of inefficient. In this sense, threading helps us keep the independent relationship between different events, allowing multitasking. For example, we use a message receiving thread to ensure the inflow of chat message while we are sending files or doing some other things on the platform.

21 With the advent of the era of big data, web crawling will play an increasingly important role in the Internet.The data in the Internet is massive, but how to automatically and efficiently obtain the information we are interested in in the Internet and use it for us is an important problem, and the crawler technology is born to solve these problems. Remember We first get contact with Web Crawling in our ICP project where instructor asked us to extract useful information from the internet HTML file using find method and recursive programming. Here, we use this idea, but with more powerful third party web crawling modules to implement the music download function. Here in the project, we have a taste of the powerful function of WebCrawling Modules and implement a web crawler for music. The url string is what I find online which is available for crawling music freely. The basic rationale of Web Crawling is that we pretend to be a normal user to access the website and do something.

24 There is still more we want to realize but due to time limit we couldn;t, for example. We’ve been wondering how to realize Real-time voice call or video chatting until one of my friends who are studying IMA told me something about opencv. If there is more time, probably this could be realized. Next would be some beautification of the buttons. We could substitute them with QLabel Widget in PyQt5 and iconize the QLabel to make the interface more stylized. Animation is heavily implemented on all kinds of website. Whether these animations are for a beautification and advertising purposes or to cover up some bugs for a smoother user experience, they are necessary in the application. The transition animation designed by Apple is a good example to better user experience. Now that our chatting system can only be runned inside pycharm and the server port and address are fixed to localhost. It is possible for us to Django framework to simulate a terminal server and create a public website address and port for any users to access.

25 Thank you, any questions?