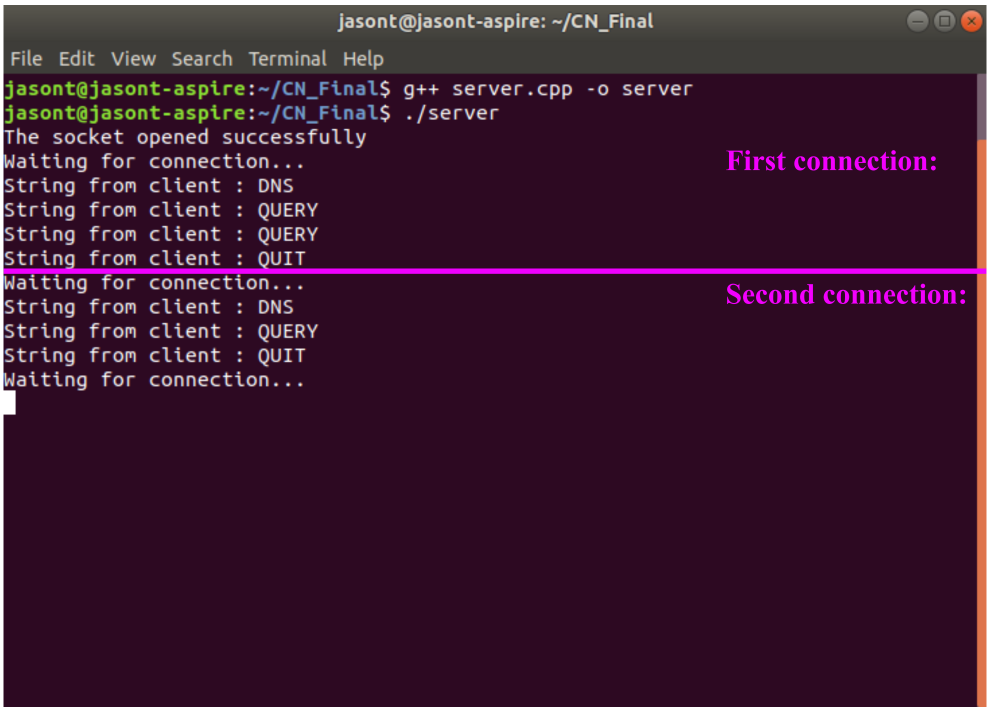
Computer Networks Final Project

107060011 電資院學士班大三 涂皓鈞

- Port : 1234
- IP address : 127.0.0.1
- Coding environment : Ubuntu 18.04

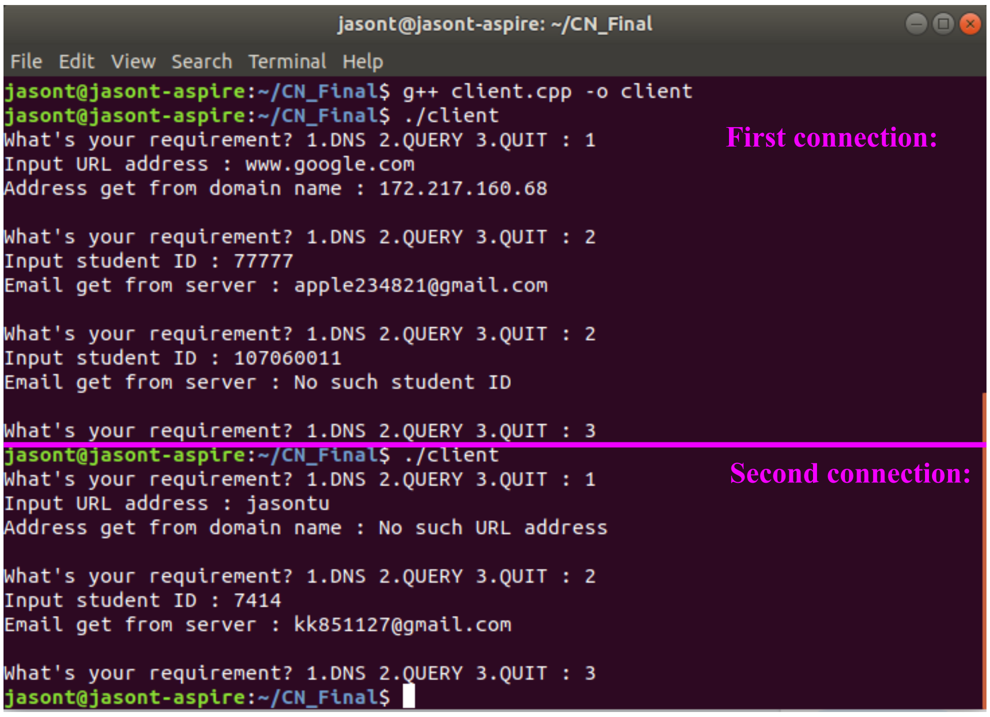
1. Results

- In server side :


jasont@jasont-aspire: ~/CN_Final
File Edit View Search Terminal Help
First connection:
Second connection:

```
jasont@jasont-aspire:~/CN_Final$ g++ server.cpp -o server
jasont@jasont-aspire:~/CN_Final$ ./server
The socket opened successfully
Waiting for connection...
String from client : DNS
String from client : QUERY
String from client : QUERY
String from client : QUIT
Waiting for connection...
String from client : DNS
String from client : QUERY
String from client : QUIT
Waiting for connection...
```

- In client side :

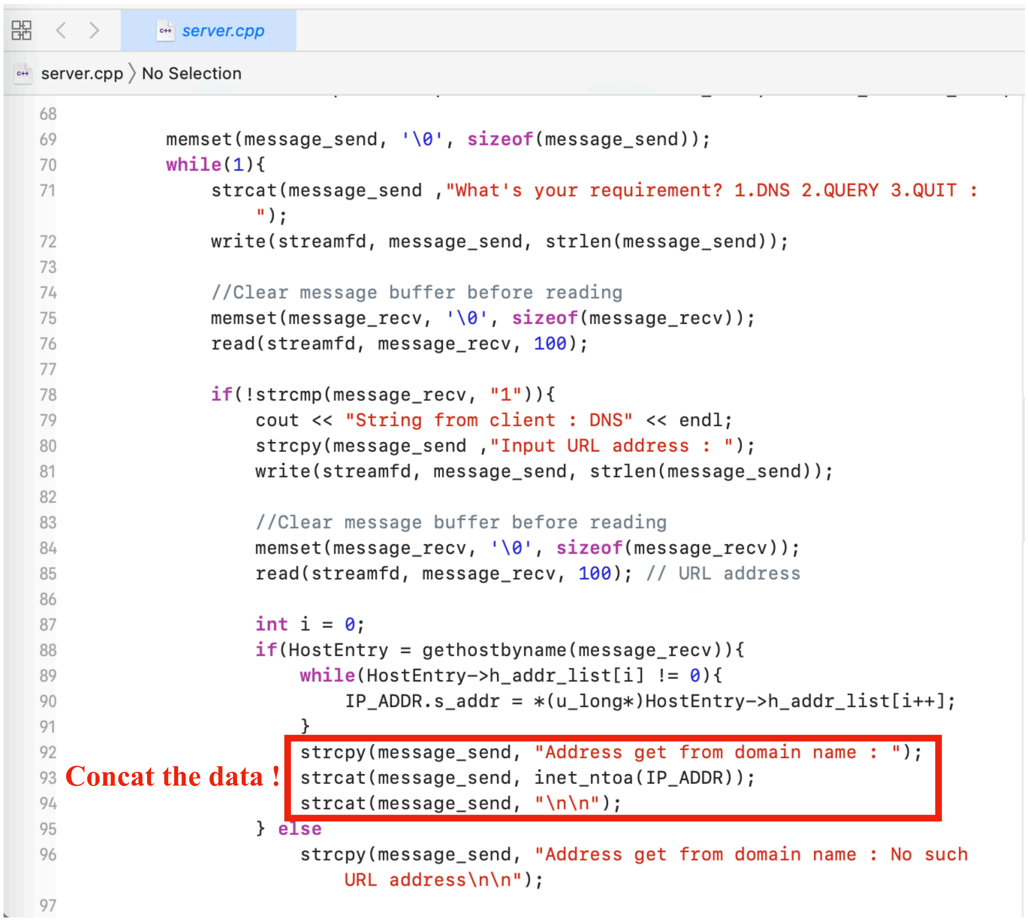

jasont@jasont-aspire: ~/CN_Final
File Edit View Search Terminal Help
First connection:
Second connection:

```
jasont@jasont-aspire:~/CN_Final$ g++ client.cpp -o client
jasont@jasont-aspire:~/CN_Final$ ./client
What's your requirement? 1.DNS 2.QUERY 3.QUIT : 1
Input URL address : www.google.com
Address get from domain name : 172.217.160.68

What's your requirement? 1.DNS 2.QUERY 3.QUIT : 2
Input student ID : 77777
Email get from server : apple234821@gmail.com

What's your requirement? 1.DNS 2.QUERY 3.QUIT : 2
Input student ID : 107060011
Email get from server : No such student ID

What's your requirement? 1.DNS 2.QUERY 3.QUIT : 3
jasont@jasont-aspire:~/CN_Final$ ./client
What's your requirement? 1.DNS 2.QUERY 3.QUIT : 1
Input URL address : jasontu
Address get from domain name : No such URL address

What's your requirement? 1.DNS 2.QUERY 3.QUIT : 2
Input student ID : 7414
Email get from server : kk851127@gmail.com

What's your requirement? 1.DNS 2.QUERY 3.QUIT : 3
jasont@jasont-aspire:~/CN_Final$
```

2. Experience

In this project, we are mainly to implement TCP/IP socket programming in C/C++. To meet the three chief requirements : 1) DNS : convert URL address the client sent into IP address, and the result is transmitted to the client. 2) QUERY : find the corresponding student' email based on the student ID in query.txt file, and the result is sent to the client. 3) QUIT : when the client quits connecting, the server remains waiting another connections. First, regarding to DNS request, in C++ library <netdb.h> supporting a **struct "hostent* gethostbyname(char* URL)"** to find its official host name, host address type, length of the address and also the IP address we want. If URL doesn't exit, gethostbyname(char* URL) will return 0, so I use it to judge this case. Second, focusing on QUERY request, I use the library of std::fstream to open the text file. Next, put the data in the file into a string array, and compare the data with the student ID the client sent. If the student ID is in the string array, the server will reply the corresponding student email to the client. If not, the server will reply "no such student ID" to the client. As for QUIT request, the server will receive the message, close the client socket, and go back to listen mode to wait for another connection.

When doing the project, I encountered a main problem : the data transferring to the server will go wrong usually. I was stuck in this problem for a long time. After searching help from Internet, I realized that the data-sending for client-server would not allow repeatedly read or repeatedly write in TCP/IP standard. Thus, we have to concat the data we want to transmit. Just as below :



```
        memset(message_send, '\0', sizeof(message_send));
        while(1){
            strcat(message_send ,"What's your requirement? 1.DNS 2.QUERY 3.QUIT :
                ");
            write(streamfd, message_send, strlen(message_send));

            //Clear message buffer before reading
            memset(message_recv, '\0', sizeof(message_recv));
            read(streamfd, message_recv, 100);

            if(!strcmp(message_recv, "1")){
                cout << "String from client : DNS" << endl;
                strcpy(message_send ,"Input URL address : ");
                write(streamfd, message_send, strlen(message_send));

                //Clear message buffer before reading
                memset(message_recv, '\0', sizeof(message_recv));
                read(streamfd, message_recv, 100); // URL address

                int i = 0;
                if(HostEntry = gethostbyname(message_recv)){
                    while(HostEntry->h_addr_list[i] != 0){
                        IP_ADDR.s_addr = *(u_long*)HostEntry->h_addr_list[i++];
                    }
                    strcpy(message_send, "Address get from domain name : ");
                    strcat(message_send, inet_ntoa(IP_ADDR));
                    strcat(message_send, "\n\n");
                } else
                    strcpy(message_send, "Address get from domain name : No such
                        URL address\n\n");
```

This final project gave us more clear images about TCP/IP socket programming and the transmission mechanism of the networks. Although the project seems quite simple, it is the first and good start to realize the computer network. I think it is interesting and practical, and maybe I can try another advanced socket programming to dig in the computer network. Finally, thank you for all TAs and professor!