

Networks Assignment-8,

A socket based peer-to-peer to chat application

Atharva Naik (18CS10067)

Radhika Patwari (18CS10062)

Documentation:

This section details the file structure, functions used and design choices taken for building this application. The code is divided among two files: 1. chatapp.cpp 2. utils.h

1. **chatapp.cpp:** This is the heart of the code and contains the main function and all network related code.

The primary functions used in this file are:

```
1. vector<vector<string>> read_user_info(string fname) //  
   populate user info in a vector of vectors, from the  
   user_info.csv  
2. int get_record(vector<vector<string>> table, string key, int  
   col) /* find the row index of a record in the user_info table  
   using any unique column value as a key. In case of a  
   non-unique key value first occurrence is returned. -1 is  
   returned in case no record is matched.*/  
3. int main(int argc, char* argv[]) /* We take the port number  
   for the user as a mandatory argument, so we have an assertion  
   that argc or the count of arguments be at least 1. The second  
   optional argument is the filename from where to read  
   user_info. In case only one argument is supplied we assume the  
   existence of the user_info.csv, from which we read the  
   user_info table. We are supplying a sample user_info.csv with  
   the source code. */
```

The macros used are:

```
1. MAX_MSG_LEN 10000 // maximum characters allowed in the  
   message+send_to name  
2. MAX_USERS 5 // the maximum number of peers allowed.  
3. DEBUG false // a flag to indicate whether the code is to be  
   run in debug mode. Debug mode prints a lot of internal info,
```

```

from inside the functions. It can be used to debug the code
whenever needed.
4. const int TIMEOUT_VAL = 30 // timeout duration of 30 seconds
or 5 minutes.
5. STDIN 0 // to represent the standard input file descriptor.

```

2. **user_info.csv**: csv file containing the static port-ip-name records for each user of the chat application.
3. **utils.h**: This is a custom header file having implementations of some of the utility functions mentioned above.

The primary functions used in this file are:

```

1. vector<string> tokenize(string &in, char delim) // this
function tokenizes the input string according to the delimiter
delim. The output tokens are returned as std::vector of
std::string types.
2. void print_msg(string msg) // a function to parse the
friend/<message> format raw message and print the message.
3. void tokenize_prefix(string in, char delim, string &prefix,
string &suffix) // split the string in according to delim,
such that everything to the left of the first occurrence of
the delim is stored in the string prefix and everything after
the delim is stored in the string suffix.
4. string pad_str(string in, char delim, int n) // pads the
string in to the right with the character delim, till the
combined string has a length of n and returns it. E.g.
pad_str("hello", "*", 10) returns "hello*****".
5. string repeat(string s, int n) // repeat string s, n times.
6. void print_table(vector<vector<string>> table, vector<string>
fields) // print a vector of vectors as a table, with the
names of the columns being passed as the fields argument (a
vector of strings.)

```

System requirements:

This application assumes that the user is using a POSIX based system, having a c++11 compatible compiler. Following header files are included.

```
#include "utils.h"
#include <ctime>
#include <chrono>
#include <errno.h>
#include <fcntl.h>
#include <netdb.h>
#include <dirent.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/in.h>
#include <bits/stdc++.h>
```

How to run:

A Makefile is provided with the source code.

To compile:

```
make
```

To run:

```
./app <port_number>
```

(Note: Please replace <port_number> with an appropriate port number from the **user_info** table to assume the role of the user corresponding to that port number in the **user_info** table. e.g. “./app 3901”, will launch an instance corresponding to Rohan; 127.0.0.1; 3901)

To clean (remove the executable file):

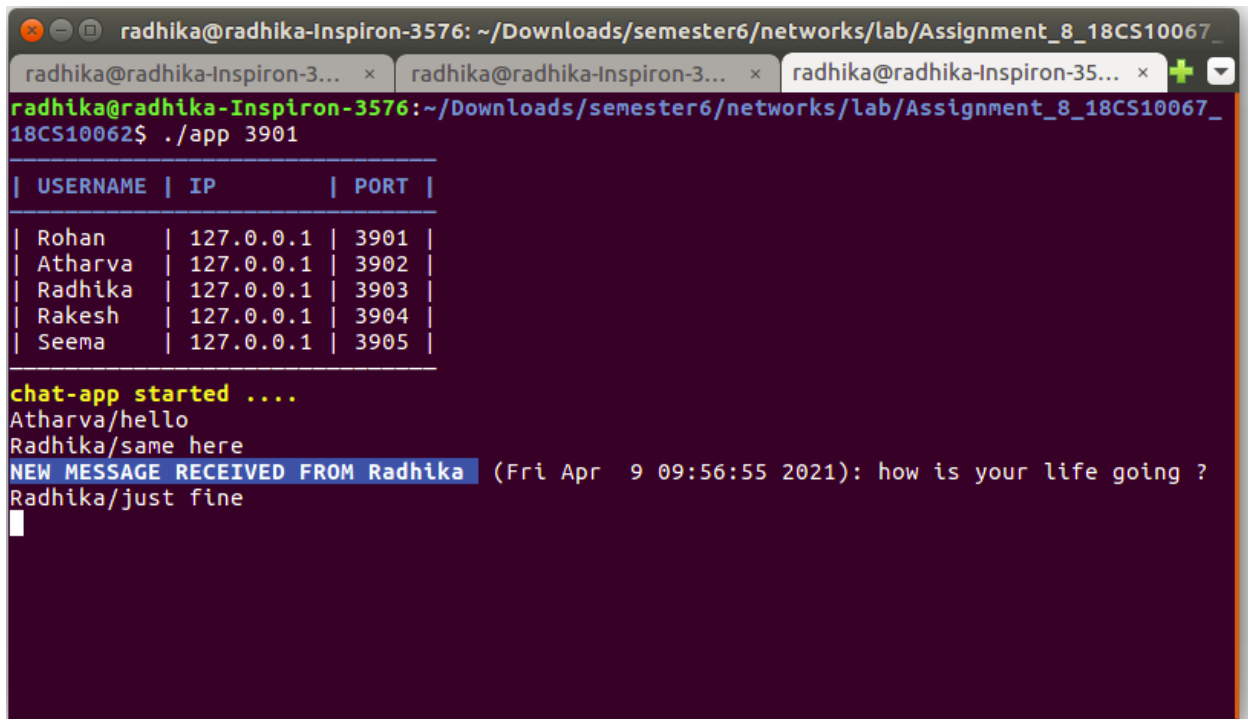
```
make clean
```

To voluntarily exit a currently running session:

Just enter `exit` on the console and hit ENTER.

Sample input/output:

This section features some sample input/output. We launched a session with three users, port 3901,3902 and 3903. Our application prints the user_info table on start. This helps the users know who they can talk to. We show the conversation taking place among these 3 uses and how the timeout and closing connection modules work in different scenarios.



```
radhika@radhika-Inspiron-3576: ~/Downloads/semester6/networks/lab/Assignment_8_18CS10067_
radhika@radhika-Inspiron-3... x radhika@radhika-Inspiron-3... x radhika@radhika-Inspiron-35... x +
radhika@radhika-Inspiron-3576:~/Downloads/semester6/networks/lab/Assignment_8_18CS10067_
18CS10062$ ./app 3901
```

USERNAME	IP	PORT
Rohan	127.0.0.1	3901
Atharva	127.0.0.1	3902
Radhika	127.0.0.1	3903
Rakesh	127.0.0.1	3904
Seema	127.0.0.1	3905

```
chat-app started ....
Atharva/hello
Radhika/same here
NEW MESSAGE RECEIVED FROM Radhika (Fri Apr 9 09:56:55 2021): how is your life going ?
Radhika/just fine
```

Screenshot of instance launched with port=3901

```
radhika@radhika-Inspiron-3576: ~/Downloads/semester6/networks/lab/Assignment_8_18CS10067_18CS10062$ ./app 3902
```

USERNAME	IP	PORT
Rohan	127.0.0.1	3901
Atharva	127.0.0.1	3902
Radhika	127.0.0.1	3903
Rakesh	127.0.0.1	3904
Seema	127.0.0.1	3905

```
chat-app started ....
Radhika/hello
NEW MESSAGE RECEIVED FROM Radhika (Fri Apr 9 09:56:00 2021): hey
Radhika/how are you doing
NEW MESSAGE RECEIVED FROM Radhika (Fri Apr 9 09:56:19 2021): i am doing well
Connection accepted from Rohan
NEW MESSAGE RECEIVED FROM Rohan (Fri Apr 9 09:56:34 2021): hello
Closing connection with: Rohan
```

Screenshot of instance launched with port=3902

```
g++ chatapp.cpp -std=c++11 -w -o app
radhika@radhika-Inspiron-3576:~/Downloads/semester6/networks/lab/Assignment_8_18CS10067_18CS10062$ ./app 3903
```

USERNAME	IP	PORT
Rohan	127.0.0.1	3901
Atharva	127.0.0.1	3902
Radhika	127.0.0.1	3903
Rakesh	127.0.0.1	3904
Seema	127.0.0.1	3905

```
chat-app started ....
Connection accepted from Atharva
NEW MESSAGE RECEIVED FROM Atharva (Fri Apr 9 09:55:44 2021): hello
Atharva/hey
NEW MESSAGE RECEIVED FROM Atharva (Fri Apr 9 09:56:09 2021): how are you doing
Atharva/i am doing well
Connection accepted from Rohan
NEW MESSAGE RECEIVED FROM Rohan (Fri Apr 9 09:56:43 2021): same here
Rohan/how is your life going ?
Closing connection with: Atharva
NEW MESSAGE RECEIVED FROM Rohan (Fri Apr 9 09:57:04 2021): just fine
```

Screenshot of instance launched with port=3903