
High Level Design & Low Level Design

Document Control :

Project Revision History

| Date | Version | Author | Brief Description of Changes | Approver Signature |
|-------------|----------------|---------------|-------------------------------------|---------------------------|
| 19.10.2022 | 1.0 | Group 6 | | |
| | | | | |
| | | | | |

Index

| | |
|---|----|
| 1. Introduction | 4 |
| 1.1 Intended audience | 4 |
| 1.2 Project purpose | 4 |
| 1.3 Key project objective | 4 |
| 1.4 Project scope and limitation | 4 |
| 1.5 Functional overview | 4 |
| 1.5.1 Header files | 4 |
| 1.5.2 Functions | 5 |
| 2. Design overview | 6 |
| 2.1 Design objective | 7 |
| 2.2 Design alternative | 7 |
| 2.3 User interface paradigms | 7 |
| 2.4 Error detection/ Exceptional Handling | 8 |
| 2.5 Performance | 8 |
| 2.6 Maintenance | 8 |
| 3. System architecture | 8 |
| 3.1 Structure | 8 |
| 4. Detailed system design | 10 |
| 5. Environment description | 16 |
| 5.1 Time zone support | 16 |
| 5.2 Language support | 16 |
| 5.3 User desktop requirement | 16 |
| 5.4 Server-side requirement | 16 |
| 5.4.1 Deployment consideration | 16 |
| 5.4.2 Application server disk space | 16 |
| 5.4.3 Database server disk space | 16 |
| 5.4.4 Integration requirements | 16 |
| 5.4.5 Network | 17 |
| 5.5 Configuration | 17 |
| 5.5.1 Operating system | 17 |
| 6. Reference | 17 |
| 7. Tools Report | 17 |
| 8. Testing Report | 20 |

1. Introduction: -

1.1 Intended Audience: -

The target audience are the clients whose access rights are tracked and functions are specified. Also, the server who has to authenticate the clients and concurrently manage them.

1.2 Project Purpose: -

The purpose of this document is to track and record all the information and events occurring throughout the phonebook and keep the details organized so that we can track the progress and functionalities of the clients and the server properly.

1.3 Key Project Objectives: -

- Allow user to add and remove contacts.
- View details of users in phonebook directory.
- Allow admin user to add and remove groups.
- Allow authenticated users to change groups.

1.4 Project scope and limitation: -

This project scope is for creating and maintaining a remote phonebook for a client having a required set of features. It aims at smooth functioning and connections between the client and server. All clients should be able to access certain functions based on their level of authentication.

1.5 Functional Overview: -

1.5.1 Following header files are included in the program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/socket.h>
#include <string.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/ipc.h>
#include <user.h>
#include <stdbool.h>
#include <server.h>
#include <client.h>
```

Functions:

1.5.2 SERVER FUNCTIONS:

1.5.2.1: User Authentication:

The server should authenticate the user from a pool of registered user data kept with appropriate data structure.

1.5.2.2: Server side Firewall Protection:

The server should listen to port 8028 for client connection and make the client side computer access the server port.

1.5.2.3 Service starting at boot:

The server must configure to start the service automatically at boot.

1.5.2.4 Server side concurrency:

The server should support the concurrent client connection.

1.5.3 CLIENT FUNCTIONS:

1.5.3.1: Client side program starting:

The user should start the client program to connect the Server whenever he wants.

1.5.3.2: Client side connection request:

Client should request a connection to the server on port 8028.

1.5.3.3: Client side authentication:

Client program should start with an authentication

1.5.3.4: Client side user environment:

On successful authentication the user should be placed in a public group for phone book access.

1.5.3.5: Client side environment customization:

Users can use chgrp GRPNAME to change his Working group on phone book.

1.5.3.6: Client side browsing contact:

list[A] to list out all contacts in a group

1.5.3.7: Client side contact Add:

User adds contact using this function.

1.5.3.8: Client side contact deletion:

User can remove any contact from working group

Client side Group Add and Remove:

Admin authenticated user use this to add and remove groups.

Client side quit:

The user uses the subcommand bye to terminate the connection.

1.5.4 USER FUNCTIONS:

1.5.4.1: Add contact:

User can add contact to public or desired groups depending on permission for each type of user.

1.5.4.2: Remove Contact:

User can remove contact from public or desired groups depending on permission for each type of user.

1.5.4.3: Add group:

Admin can add a group.

1.5.4.4 Remove group:

Admin can remove a group.

1.5.4.5: Change group:

Authenticated user can change group to which it has access.

1.5.4.6: List:

User can list details of public or desired groups depending on permission for each type of user.

1. Design Overview: -

Remote Phonebook comprises of the following modules:

| | |
|--------------------|--|
| Name of the Module | List contact & Remove contact |
| Handled by | Ayeshkanta Adhikari |
| Description | List can view user contact details. Remove can delete contact from database |

| | |
|--------------------|--|
| Name of the Module | Server functionalities |
| Handled by | Buchireddy Gari Sai nath Reddy |
| Description | Create socket, accept connections, authenticate user, receive and process user commands, send results to clients |

| | |
|--------------------|------------------------|
| Name of the Module | Client functionalities |
| Handled by | Arpita Panda |

| | |
|-------------|---|
| Description | Create socket, connect with server get credentials and commands, send, receive and display data |
|-------------|---|

| | |
|--------------------|---|
| Name of the Module | Add & Remove groups |
| Handled by | Dileep Varma |
| Description | These will add and remove groups. Done by Admin. |

| | |
|--------------------|---|
| Name of the Module | Change group |
| Handled by | Vinaykumar J |
| Description | These will help authenticated user to change group to do its functionalities. |

| | |
|--------------------|--|
| Name of the Module | Add contacts |
| Handled by | Utkarsh Kumar Bharti |
| Description | These will add contact in database depending on groupname. |

2.1 Design Objectives: -

- A remote phonebook setup to be established.
- After server is started, it listens to port 8028 for clients.
- Connection between client and the server is established.
- Three user access categories based on authentication.
- Each user access category has specified functionalities.
- Contact deletion, contact addition and group addition and remove.
- Connection can be terminated by a specific command by user

2.2 Design Alternative: -

We have used file structures for performing operations in this project. Information is directly changed in file so less space is used during program execution.

2.3 User Interface Paradigms: -

The client would ask authentication from the server and after authentication they would remain in the public group by default. There would be three types of access categories for the user: anonymous, authenticated and admin authenticated. For an anonymous client, the server should support a virtual user with the name anonymous to add contact to a public group only without any password. All the authenticated users should be allowed to add, view and delete contact from the phone directory for the group they belong to. They should view the content of public groups. Users with authenticated admin access can access any contact in addition to add, remove group to phone directory.

2.4 Error Detection / Exceptional Handling: -

- If the user doesn't have access to group and tries to change group , then it will show error "The user doesn't belong to the group".
- If an authenticated user tries to add contact in public group, then it will show error "Authenticated user can't add the data to the public group"
- If the user tries to remove a contact which is not present in the group , then it will show error "Entered contact name is not present".

We have handled these errors by using flags at different parts of our codes and returning that to server for checking and sending the error messages to the clients.

2.5 Performance: -

Multiple clients can be connected to the remote server. The performance shall depend upon hardware components of the clients and the internet connection.

2.6 Maintenance: -

Very little maintenance should be required for this setup. An initial configuration will be the only system required interaction after system is put together. The only other user maintenance would be any changes to settings after setup, and any specified special cases where user settings or history need to be changed. Physical maintenance on the system's parts may be required, and would result in temporary loss of data or Internet. Upgrades of hardware and software should have little effect on this project but may result in downtime.

3 SYSTEM ARCHITECTURE: -

3.1 Structure Details:

The system consists of three structures Server, client and User functionalities :

1.5.2 SERVER:

1.5.2.1: User Authentication:

The server should authenticate the user from a poll of registered user data kept with appropriate data structure.

1.5.2.2: Server side Firewall Protection:

The server should listen to port 8028 for client connection and make the client side computer access the server port.

1.5.2.3 Service starting at boot:

The server must configure to start the service automatically at boot.

1.5.2.4 Server side concurrency:

The server should support the concurrent client connection.

1.5.3 CLIENT:

1.5.3.1: Client side program starting:

The user should start the client program to connect the Server whenever he wants.

1.5.3.2: Client side connection request:

Client should request a connection to the server on port 8028.

1.5.3.3: Client side authentication:

Client program should start with an authentication

1.5.3.4: Client side user environment:

On successful authentication the user should be placed in a public group for phone book access.

1.5.3.5: Client side environment customization:

Users can use chgrp GRPNAME to change his Working group on phone book.

1.5.3.6: Client side browsing contact:

list[A] to list out all contacts in a group

1.5.3.7: Client side contact Add:

User adds contact using this function.

1.5.3.8: Client side contact deletion:

User can remove any contact from working group

Client side Group Add and Remove:

Admin authenticated user use this to add and remove groups.

Client side quit:

The user uses the subcommand bye to terminate the connection.

1.5.4 USER:

1.5.4.1: Add contact:

User can add contact to public or desired groups depending on permission for each type of user.

1.5.4.2: Remove Contact:

User can remove contact from public or desired groups depending on permission for each type of user.

1.5.4.3: Add group:

Admin can add a group.

1.5.4.4 Remove group:

Admin can remove a group.

1.5.4.5: Change group:

Authenticated user can change group to which it has access.

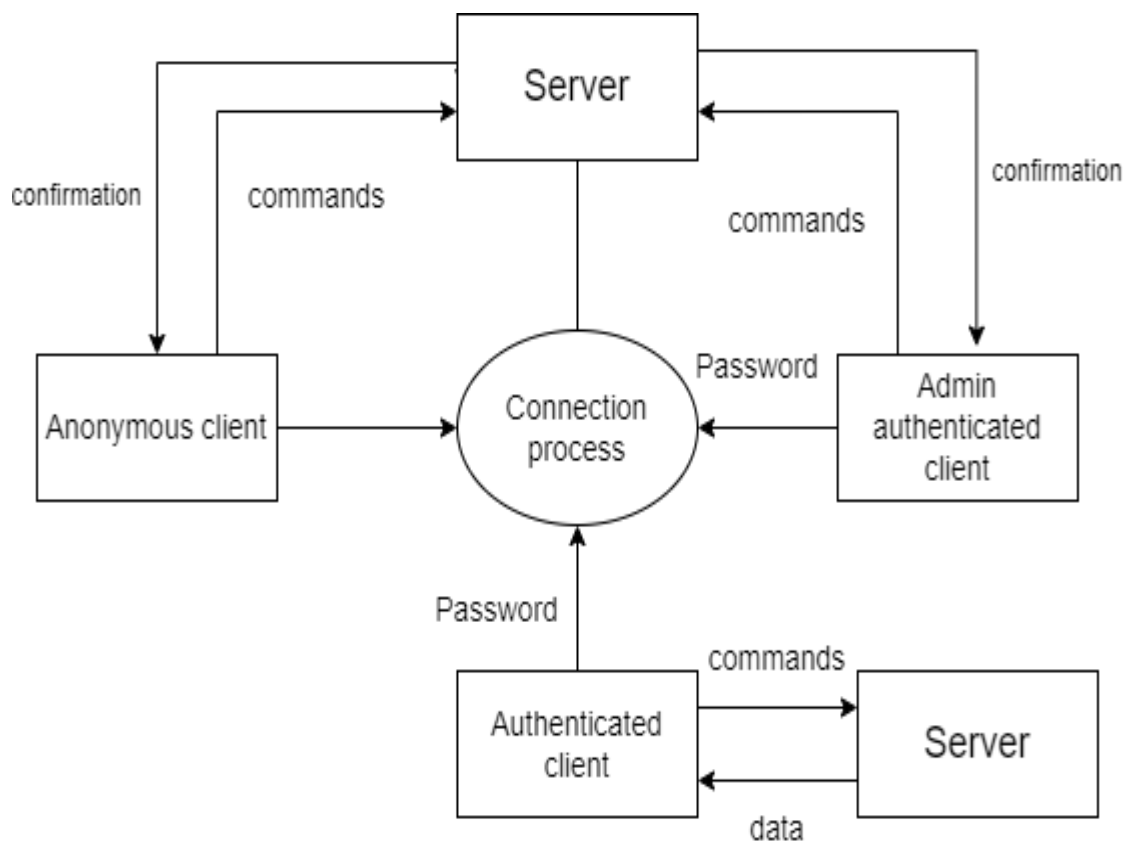
1.5.4.6: List:

User can list details of public or desired groups depending on permission for each type of user.

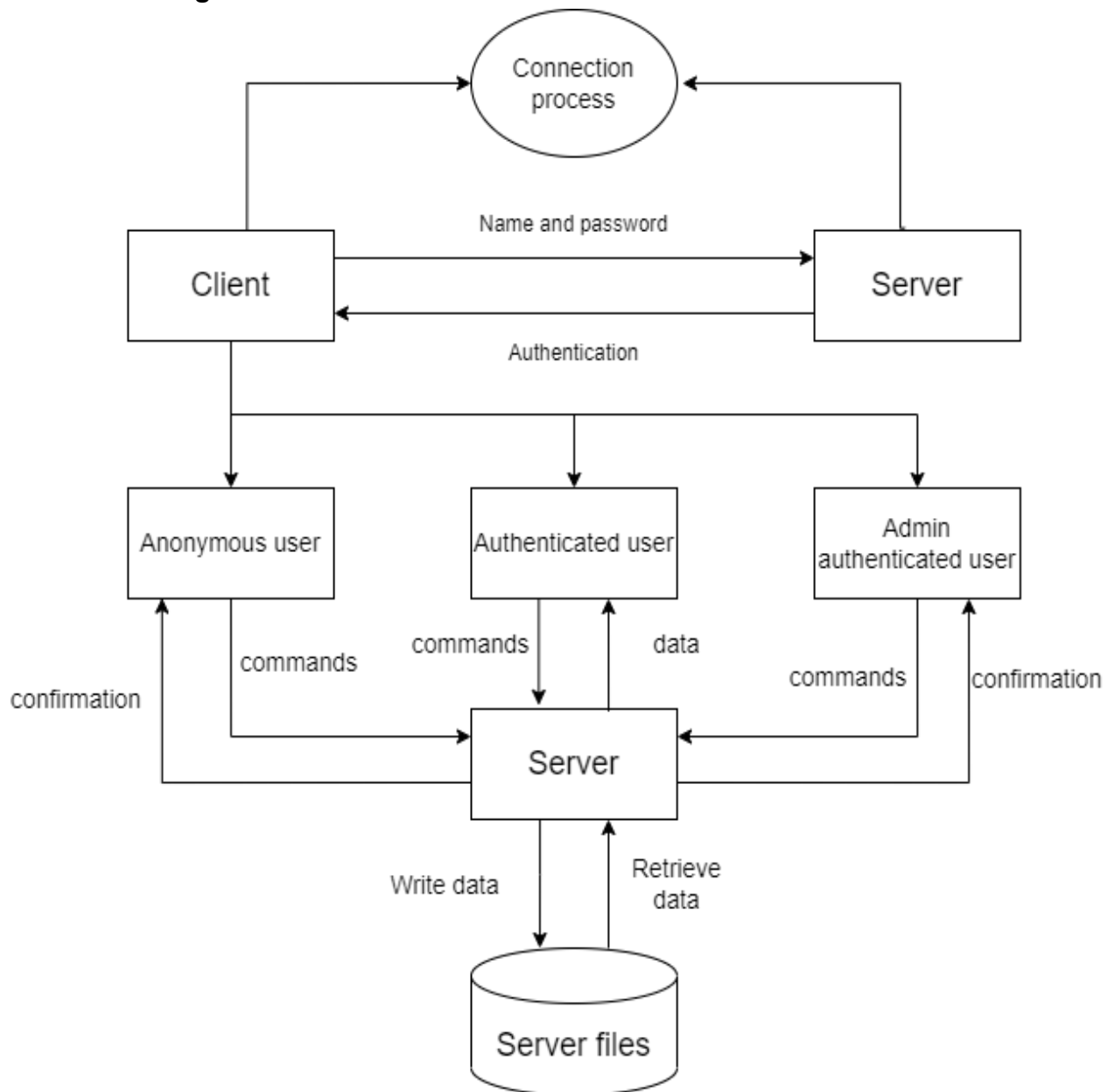
2. Detailed System Design

Design Overview

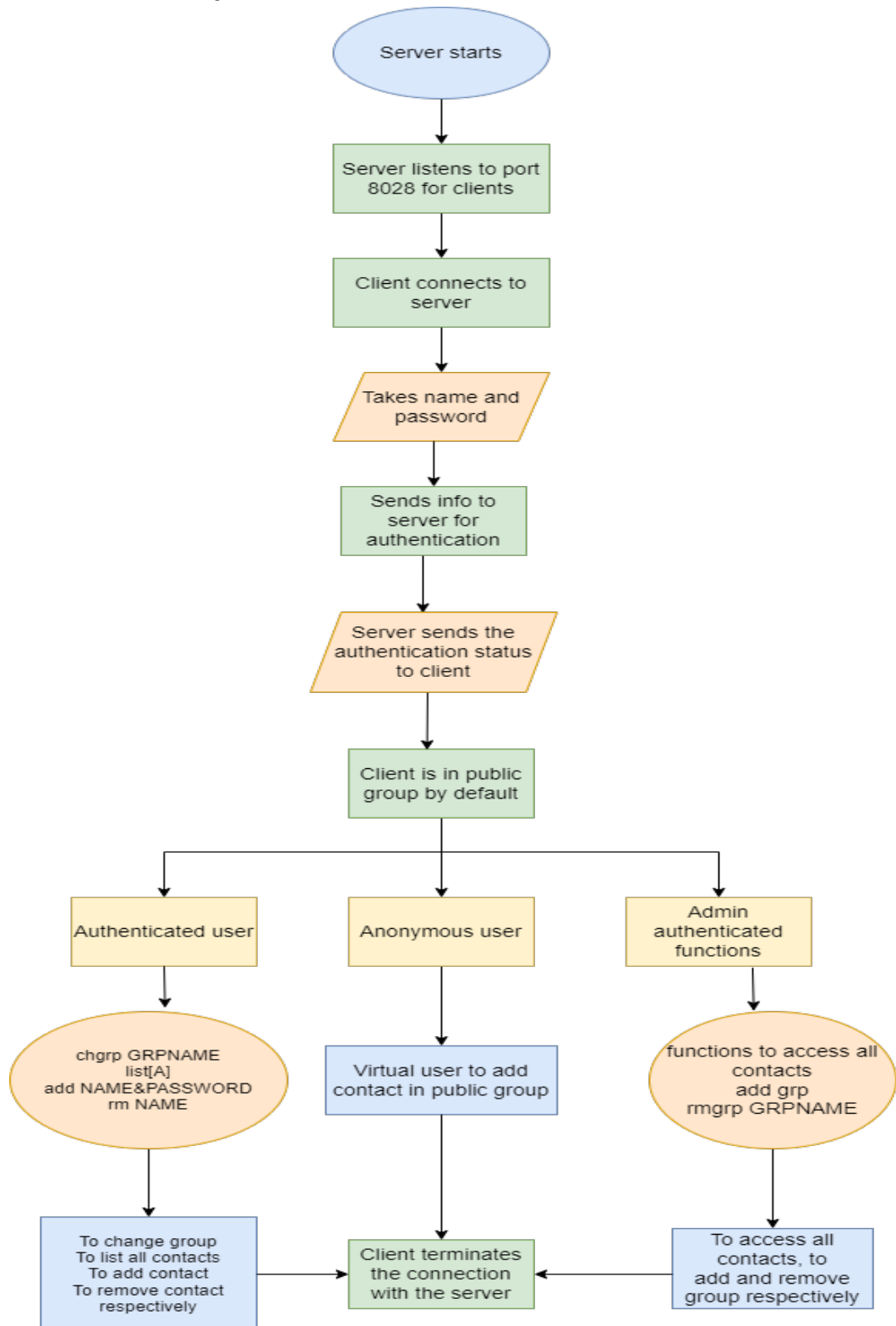
Data Flow Diagram Level 0:



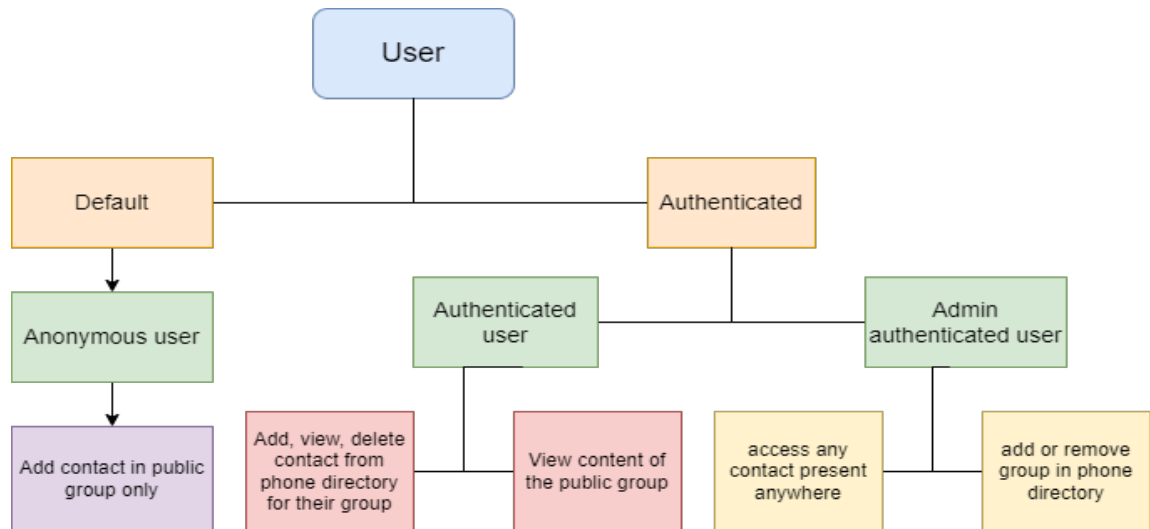
Data Flow Diagram Level 1:



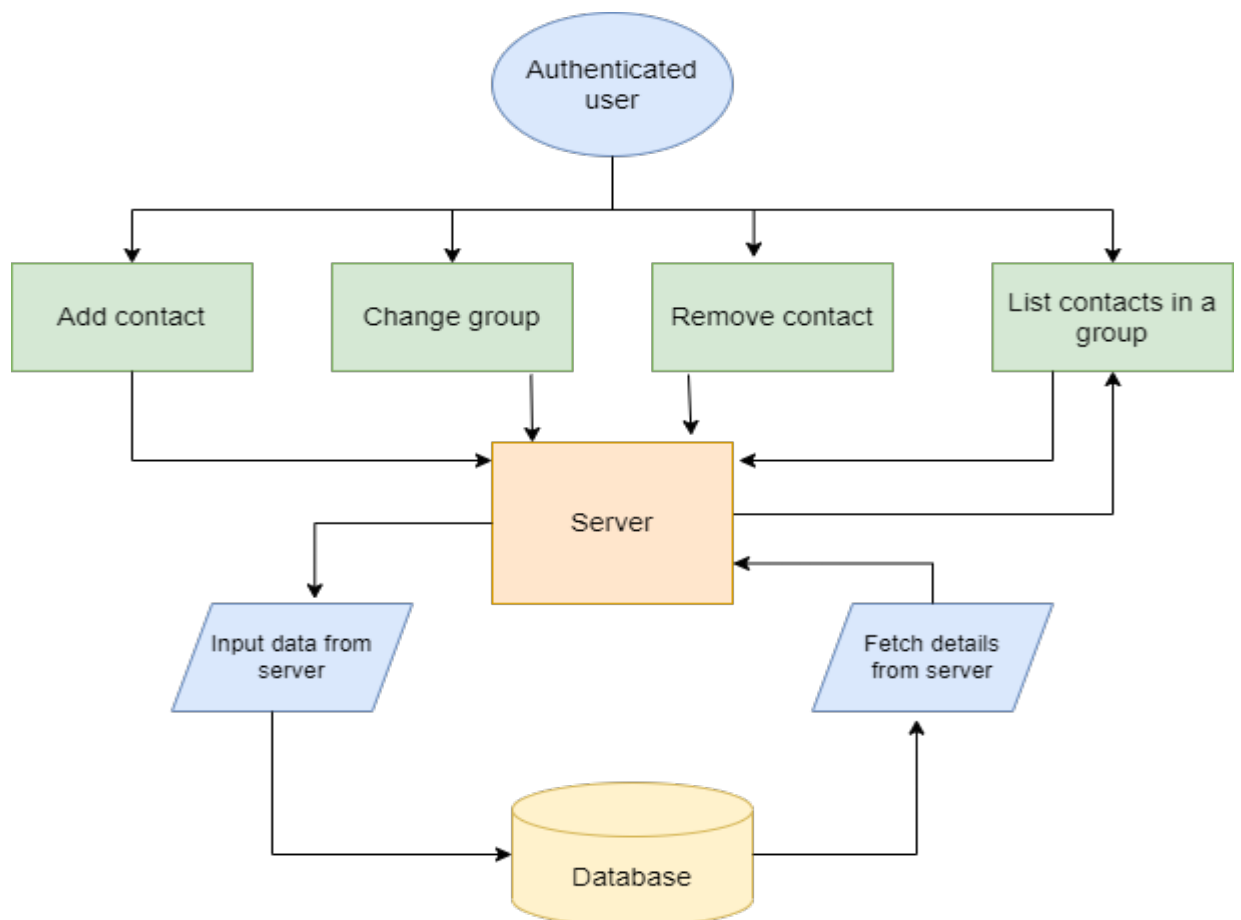
Flowchart for Remote phonebook:



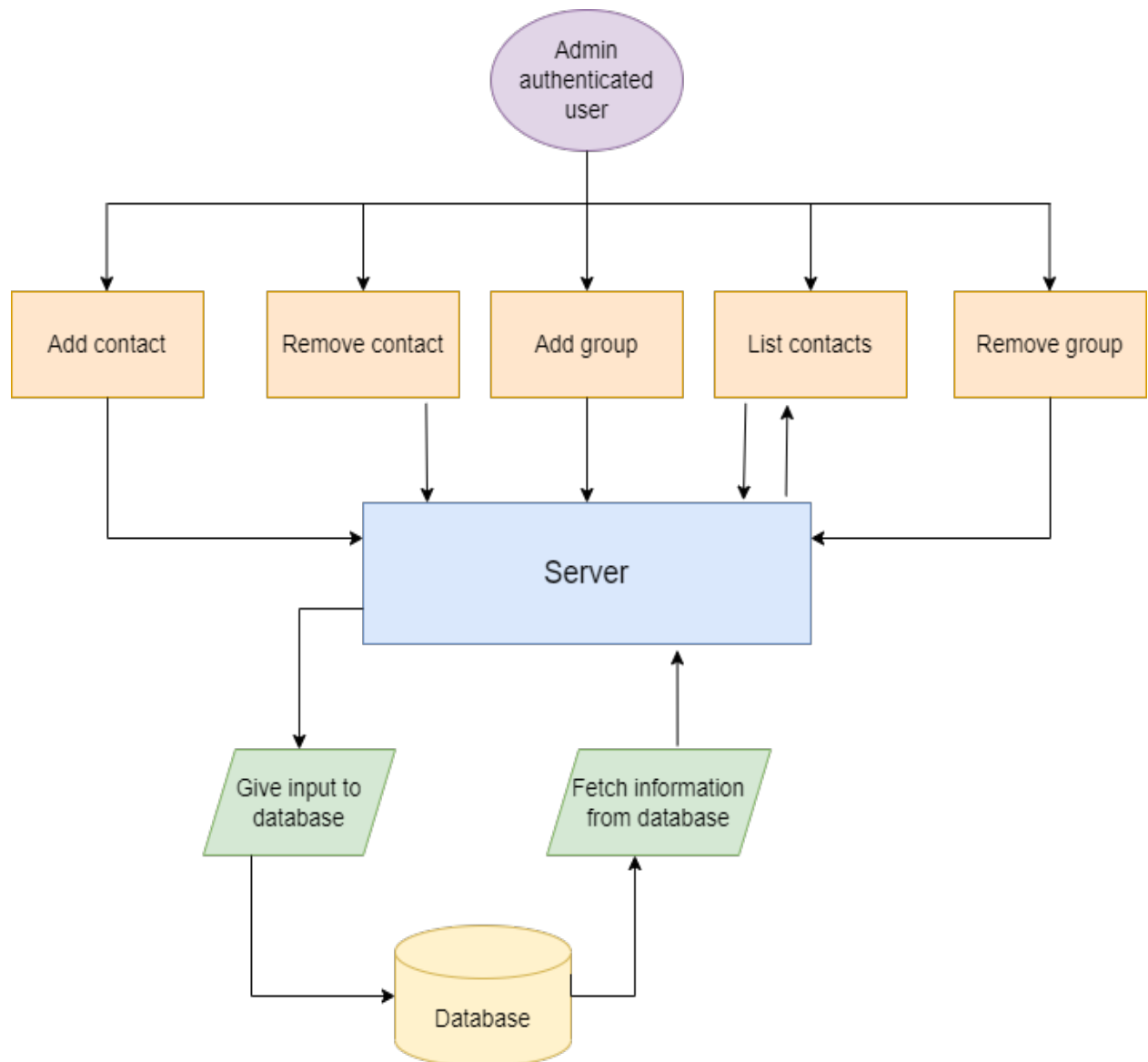
Flowchart for user:



Flowchart for functions of authenticated user:



Flowchart for functions for admin authenticated user:



3. Environment Description: -

5.1 Time Zone Support: - IST- Kolkata

5.2 Language Support: - English

5.3 User Desktop Requirements: -

- 64-bit processor, 1 GHz or faster
- At least 10 GB free hard drive space
- At least 1 GB RAM **Server**

5.4 Server-Side Requirements:

- 64-bit processor, 1 GHz or faster
- At least 2 GB free hard drive space
- At least 1GB RAM

5.4.1 Deployment Considerations: -

- Local storage is used
- No network latency to consider
- To scale buy a bigger CPU, more memory, larger hard drive, or additional hardware

5.4.2. Application Server Disk Space: -

No such disk space is required as the program is fully functional on online IDE(s) as well. Local Operating System is required and some txt files to store the records of phonebook.

5.4.3. Database Server Disk Space: -

No such disk space is required as the program is fully functional on online IDE(s) as well. Local Operating System is required and some txt files to store the records of phonebook.

5.4.4. Integration Requirements: -

- Language: - C
- Tools: - Valgrind, Makefile ,Cunit, gprof, splint, gcov
- Compiler: - gcc
- Linux Environment

5.4.5. Network: - End to End

5.5 Configuration: -

5.5.1. Operating System: - Linux environment

6. Reference: -

The references are:

- <https://www.geeksforgeeks.org/socket-programming-cc/>
- <https://www.javatpoint.com/file-handling-in-c>
- <https://www.educative.io/answers/how-to-create-a-simple-thread-in-c>

7. TOOLS AND TESTING

Gcov report:

```
$ gcov ../bin/client-clientmain.gcda
File '../src/clientmain.c'
Lines executed:72.00% of 25
Creating 'clientmain.c.gcov'

Lines executed:72.00% of 25
```

```

1  -: 0:Source: ../src/clientmain.c
2  -: 0:Graph: ../bin/client-clientmain.gcno
3  -: 0:Data: ../bin/client-clientmain.gcda
4  -: 0:Runs:4
5  -: 1://program to define the server main function
6  -: 2:#include <stdio.h>
7  -: 3:#include <signal.h>
8  -: 4:#include <string.h>
9  -: 5:#include <stdlib.h>
10 -: 6:#include "client.h"
11 -: 7:int sockid ;
12 ####: 8:void signalHandler(int signal)
13 -: 9:{
14 ####: 10:     if(signal==SIGINT)
15 -: 11:     {
16 ####: 12:         send(sockid,"bye",3,0);
17 ####: 13:         exit(0);
18 -: 14:     }
19 ####: 15:}
20 -: 16:
21 -: 17://main function to execute client
22 -: 18:
23 4: 19:int main ()
24 -: 20:{
25 4: 21:     signal(SIGINT,signalHandler); //signal to cut the client from server
26 4: 22:     char recvdData[200]="",credentials[200]="",commands[200]="";
27 4: 23:     Client(); //creating a client
28 4: 24:     sockid=ToGetSockfd(); //calling the getsockfd of client
29 4: 25:     ToGetCredentials(credentials); //calling the getcredentials functions of client
30 4: 26:     ToServerConnect(); //calling the serverconnect function to connect to the server
31 4: 27:     ToSendData(credentials); //sending the credentials to server
32 4: 28:     CToRecvData(recvdData); //receiving the message from the server using recvData function
33 4: 29:     char type[200]="";
34 4: 30:     strcpy(type,recvdData);
35 4: 31:     ToDisplayRecvData(recvdData); //calling the funtion to display the data
36 -: 32:     while(1)
37 -: 33:     {
38 33: 34:         strcpy(commands,"");
39 33: 35:         ToGetUserCommands(type,commands); //calling the function to get the usercommands
40 33: 36:         if(strcmp(commands,"")!=0)
41 -: 37:         {
42 33: 38:             ToSendData(commands); //sending the user commands to the server
43 29: 39:             CToRecvData(recvdData); //receiving the data from the server
44 29: 40:             ToDisplayRecvData(recvdData); //displaying the received data from the server
45 -: 41:         }
46 -: 42:         else
47 -: 43:         {
48 ####: 44:             printf("Subcommand can't be empty");
49 ####: 45:             continue;
50 -: 46:         }
51 -: 47:     }
52 -: 48:     return EXIT_SUCCESS;
53 -: 49:}
54 -: 50:
55

```

Gprof report:

```

$ gprof -b ../bin/client gmon.out
Flat profile:

Each sample counts as 0.01 seconds.
no time accumulated

%   cumulative   self           calls   self   total    name
time  seconds    seconds                Ts/call  Ts/call
-----
0.00    0.00    0.00              7     0.00    0.00  ToSendData
0.00    0.00    0.00              6     0.00    0.00  CToRecvData
0.00    0.00    0.00              6     0.00    0.00  ToDisplayRecvData
0.00    0.00    0.00              6     0.00    0.00  ToGetUserCommands
0.00    0.00    0.00              1     0.00    0.00  Client
0.00    0.00    0.00              1     0.00    0.00  ToCloseClientConnections
0.00    0.00    0.00              1     0.00    0.00  ToGetCredentials
0.00    0.00    0.00              1     0.00    0.00  ToGetSockfd
0.00    0.00    0.00              1     0.00    0.00  ToServerConnect

```

Places

Call graph

granularity: each sample hit covers 2 byte(s) no time propagated

| index | % time | self | children | called | name |
|-------|--------|------|----------|--------|------------------------------|
| [1] | 0.0 | 0.00 | 0.00 | 7/7 | main [15] |
| | | 0.00 | 0.00 | 7 | ToSendData [1] |
| | | 0.00 | 0.00 | 1/1 | ToCloseClientConnections [6] |
| [2] | 0.0 | 0.00 | 0.00 | 6/6 | main [15] |
| | | 0.00 | 0.00 | 6 | CToRecvData [2] |
| [3] | 0.0 | 0.00 | 0.00 | 6/6 | main [15] |
| | | 0.00 | 0.00 | 6 | ToDisplayRecvData [3] |
| [4] | 0.0 | 0.00 | 0.00 | 6/6 | main [15] |
| | | 0.00 | 0.00 | 6 | ToGetUserCommands [4] |
| [5] | 0.0 | 0.00 | 0.00 | 1/1 | main [15] |
| | | 0.00 | 0.00 | 1 | Client [5] |
| [6] | 0.0 | 0.00 | 0.00 | 1/1 | ToSendData [1] |
| | | 0.00 | 0.00 | 1 | ToCloseClientConnections [6] |
| [7] | 0.0 | 0.00 | 0.00 | 1/1 | main [15] |
| | | 0.00 | 0.00 | 1 | ToGetCredentials [7] |
| [8] | 0.0 | 0.00 | 0.00 | 1/1 | main [15] |
| | | 0.00 | 0.00 | 1 | ToGetSockfd [8] |
| [9] | 0.0 | 0.00 | 0.00 | 1/1 | main [15] |
| | | 0.00 | 0.00 | 1 | ToServerConnect [9] |

Index by function name

| | | |
|------------------------------|-----------------------|-----------------------|
| [2] CToRecvData | [3] ToDisplayRecvData | [4] ToGetUserCommands |
| [5] Client | [7] ToGetCredentials | [1] ToSendData |
| [6] ToCloseClientConnections | [8] ToGetSockfd | [9] ToServerConnect |

Splint report:

Splint 3.1.2 — 21 Feb 2021

```
< Location unknown >: Field name reused:
Code cannot be parsed. For help on parse errors, see splint -help
parseerrors. (Use -syntax to inhibit warning)
< Location unknown >: Previous use of
< Location unknown >: Previous use of
server.c: (in function ToAcceptConnections)
server.c:138:16: Unrecognized identifier: SIGCHLD
Identifier used in code has not been declared. (Use -unrecog to inhibit
warning)
server.c: (in function AuthenticatedUserFunctionalities)
server.c:151:10: Parameter 1 (comm) to function strcpy is declared unique but
is aliased externally by parameter 2 (tok) through alias comm
A unique or only parameter is aliased by some other parameter or visible
global. (Use -aliasunique to inhibit warning)
< Location unknown >: Field name reused:
< Location unknown >: Previous use of
clientmain.c:19:5: Function main defined more than once
A function or variable is redefined. One of the declarations should use
extern. (Use -redef to inhibit warning)
servermain.c:15:1: Previous definition of main
< Location unknown >: Previous use of
client.c:18:5: Variable sockfd redefined
server.c:11:5: Previous definition of sockfd
client.c:19:20: Variable servaddr redefined
server.c:12:20: Previous definition of servaddr
client.c:20:16: Variable slen redefined
server.c:14:8: Previous definition of slen
client.c:20:21: Variable mlen redefined
server.c:14:18: Previous definition of mlen
client.c:20:26: Variable connectfd redefined
server.c:15:5: Previous definition of connectfd
< Location unknown >: Previous use of
Finished checking — 10 code warnings
```

Valgrind report:

```
└─$ valgrind -s ./../bin/client
==48232== Memcheck, a memory error detector
==48232== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==48232== Using Valgrind-3.19.0 and LibVEX; rerun with -h for copyright info
==48232== Command: ./../bin/client
==48232==
Enter the Username :admin
Enter the password :1
admin
Enter the subcommand :ADD arpita,8998576897,public group
Provide three inputs name,phonenumber,filename
Enter the subcommand :rm arpita,public group
contact removed from the given group
Enter the subcommand :addgrp services
Group added
Enter the subcommand :rmgrp services
Group removed
Enter the subcommand :list a,public group
Matching contacts: ad1,de1
ad1,de1
Enter the subcommand :bye
==48232==
==48232== HEAP SUMMARY:
==48232==    in use at exit: 0 bytes in 0 blocks
==48232==   total heap usage: 2 allocs, 2 frees, 2,048 bytes allocated
==48232==
==48232== All heap blocks were freed -- no leaks are possible
==48232==
==48232== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

```
└─$ valgrind -s ./../bin/client
==53578== Memcheck, a memory error detector
==53578== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==53578== Using Valgrind-3.19.0 and LibVEX; rerun with -h for copyright info
==53578== Command: ./../bin/client
==53578==
Enter the Username :sai nath
Enter the password :group6
authenticated user
Enter the subcommand :list a
Matching contacts:
ad1,de1
Enter the subcommand :chgrp production
Group changed
Enter the subcommand :ADD jist,89989899998
Contact added
Enter the subcommand :rm jist
Contact removed
Enter the subcommand :bye
==53578==
==53578== HEAP SUMMARY:
==53578==    in use at exit: 0 bytes in 0 blocks
==53578==   total heap usage: 2 allocs, 2 frees, 2,048 bytes allocated
==53578==
==53578== All heap blocks were freed -- no leaks are possible
==53578==
==53578== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```


TESTING REPORT

Unit testing report:

```
./../bin/test

CUnit - A unit testing framework for C - Version 2.1-3
http://cunit.sourceforge.net/

Suite: suite for removing contact from a file ...
  Test: Test for removecontact() in sunny cases ... passed
  Test: Test for removecontact() in rainy cases ... passed
Suite: suite for changing group ...
  Test: Test for ToChgrp() in sunny cases ... passed
  Test: Test for ToChgrp() in rainy cases ... passed

Run Summary:   Type   Total   Ran   Passed   Failed   Inactive
               suites    2       2      n/a       0         0
               tests    4       4        4       0         0
               asserts   20      20      20       0        n/a

Elapsed time =   0.001 seconds
```

Integration testing report:

Case 1: Anonymous user

```
$ make client
../bin/client
Enter the Username :
Enter the password :
anonymous user

Enter the subcommand :ADD
ADD command should have two fields
~/Desktop/run_final/make
Enter the subcommand :ADD 1
ADD command should have two fields
~/Desktop/run_final/make
Enter the subcommand :list
Anonymous user should provide only ADD subcommand
~/Desktop/run_final/make
Enter the subcommand :ADD sai,9009989098
Contact added
~/Desktop/run_final/make
Enter the subcommand :addgrp
Anonymous user should provide only ADD subcommand
Enter the subcommand :chgrp
Anonymous user should provide only ADD subcommand
Enter the subcommand :ADD Ayeshkanta,8909089988
Contact added
Enter the subcommand :bye
(kali@kali)-[~/Desktop/run_final/make]
```

Case 2: Authenticated user

```

└─$ make client
../bin/client
Enter the Username :sai nath
Enter the password :group6
authenticated user

Enter the subcommand :addgrp production
Authenticated users can only use ADD,rm,list,chgrp and bye/Bye subcommand
s
Enter the subcommand :addgrp hr
Enter the subcommand :ADD virat,1234
Authenticated user can't add the data to the public group
Enter the subcommand :addgrp temp
Enter the subcommand :list a
Matching contacts:
ad1,de1
Enter the subcommand :rmgrp temp
Group removed

Enter the subcommand :chgrp production 98976
Group changed inputs: name,phonenumber,filename

Enter the subcommand :ADD vinay,777788889076,production
Contact added to the given group

Enter the subcommand :list v,production
Matching contacts:grp,rmgrp,list,rm or ADD - subcommand only
vinay,7777888890
Enter the subcommand :list u,production
Matching contacts:
Enter the subcommand :list vin
Matching contacts:
vinay,7777888890
Enter the subcommand :rm arpita,public group
contact removed from the given group
Enter the subcommand :list q
No contacts with given pattern

Enter the subcommand :rm vinay_final/make
Contact removed

Enter the subcommand :list q
No contacts with given pattern

Enter the subcommand :bye

```

Case 3:Admin authenticated user

```
$ make client
../bin/client
Enter the Username :admin
Enter the password :1
admin@localhost:~$

Enter the subcommand :chgrp production
Admin can give addgrp,rmgrp,list,rm or ADD subcommand only

Enter the subcommand :addgrp hr,6378998976
Group added

Enter the subcommand :addgrp temp
Group added
Enter the subcommand :rmgrp temp
Group removed

Enter the subcommand :ADD utkarsh,6378998976
Provide three inputs name,phonenumber,filename

Enter the subcommand :ADD utkarsh,6378998976,production
contact added to the given group

Enter the subcommand :kist u,production
Admin can give addgrp,rmgrp,list,rm or ADD subcommand only

Enter the subcommand :list u,production
Matching contacts:
utkarsh,6378998976

$ make server
../bin/server
Enter the subcommand :rm arpita,public group
contact removed from the given group

Got a connection
Enter the subcommand :bye
```

Case 4: Handling multiple clients

The screenshot displays a Kali Linux desktop environment with four terminal windows running the 'make' tool. The tool is designed to manage a group of contacts and handle multiple clients simultaneously.

Terminal 1 (Top Left): Shows the initial setup where a client connects to the server. The user enters the subcommand 'list', and the server responds with 'Anonymous user should provide only ADD subcommand'. The user then enters 'ADD arpita,6765998798', and the server confirms 'Contact added'.

Terminal 2 (Top Middle): Shows the user entering the subcommand 'rmgrp temp', and the server responds 'Group removed'. The user then enters 'ADD utkarsh,6378998976', and the server responds 'Provide three inputs name,phonenumber,filename'. The user then enters 'ADD utkarsh,6378998976,production', and the server responds 'contact added to the given group'. The user then enters 'list u,production', and the server responds 'Admin can give addgrp,rmgrp,list,rm or ADD subcommand only'. The user then enters 'list u,production', and the server responds 'Matching contacts: utkarsh,6378998976'.

Terminal 3 (Top Right): Shows the user entering the subcommand 'chgrp production', and the server responds 'Group changed'. The user then enters 'ADD vinay,7777888898', and the server responds 'Contact added'. The user then enters 'list v', and the server responds 'Matching contacts: vinay,7777888898'. The user then enters 'list vin', and the server responds 'Matching contacts: vinay,7777888898'.

Terminal 4 (Bottom Left): Shows the user entering the subcommand 'rm vinay', and the server responds 'Contact removed'. The user then enters 'list q', and the server responds 'No contacts with given pattern'. The user then enters 'rm vinay', and the server responds 'Contact removed'. The user then enters 'list q', and the server responds 'No contacts with given pattern'. The user then enters 'bye', and the server responds 'Contact removed'.

Terminal 5 (Bottom Middle): Shows the user entering the subcommand 'rm shani', and the server responds 'Contact removed'. The user then enters 'list s', and the server responds 'No contacts with given pattern'. The user then enters 'chgrp public group', and the server responds 'Group changed'. The user then enters 'list a', and the server responds 'Matching contacts: adi,del'. The user then enters 'bye', and the server responds 'Contact removed'.

Terminal 6 (Bottom Right): Shows the user entering the subcommand 'make server', and the server responds 'Got a connection'. The user then enters 'make server', and the server responds 'Got a connection'. The user then enters 'make server', and the server responds 'Got a connection'.