



Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh

PROJECT REPORT ON

MChat

Under The Guidance Of

H. M. Abdul Fattah

Lecturer

Department of Computer Science and
Engineering

Khulna University of Engineering &
Technology

Sunanda Das

Lecturer

Department of Computer Science and
Engineering

Khulna University of Engineering &
Technology

Submitted By:

Nafiul Alam

Roll: 1807005

Course: CSE 2200

Title: Advance Programming

Platform: JAVA

MCHAT

THE CHATTING AND FILE SHARING JAVA SOFTWARE

INDEX

| | <u>Pages</u> |
|------------------------------------|---------------------|
| 1. Objectives | 3-3 |
| 2. Introduction | 4-7 |
| 3. Implementation | |
| • Detailed Description | 7-11 |
| • Flowchart | 11-13 |
| • Schema Diagram | 14-14 |
| • Sequence Diagram | 15-16 |
| • Detailed Visualization | 17-25 |
| 4. Target Vs Actual Accomplishment | 26-27 |
| 5. Risk and Issues | 28-29 |
| 6. Discussion And Conclusion | 29-29 |
| 7. References | 30-30 |

OBJECTIVES:

- To create a Java Maven project and add JavaFX library for UI.
- To create a chatting software using Java socket programming.
- To send messages and files from one user to another.
- To make a user-friendly UI like traditional chatting program.
- To create peer to peer communication. User will be connected to another user, not the server. The server will be used as a medium for communication.
- To use MongoDB database in Java Maven project.
- To store confidential in MongoDB database.
- To store previous messages in MongoDB database and restore them when the program starts.
- To check confidential every time user logs in and if a new user wants to sign up and update the data in the database in real time.
- To store sent file in user's computer's default location which is set at first time running the software.
- To have the ability to change IP, Message Port, File transfer Port and Default download location at the start of the program.
- To have the ability to see all the users who are online and offline.
- To have the ability to add friend request to another user and other user can accept the request if he/she wants. If he/she accepts, they both will be friends and it will be updated to database.
- To have the ability to communicate not only in Local Area Network but also via internet by forwarding the port of the server in the router.
- To make the program secure, every time messages are sent a special code is sent with it as a result only the specified user will get the messages also socket is closed every time a user logs out. Also, server program can watch who is connected to server.

INTRODUCTION:

MChat is a multi-client chat and file sharing software. It is built in a way that multiple clients/users can join and chat with their friends and family through server. Users will use the client program. They can login or sign up using it. The server program will have to be used by someone who has both server access and database access. Server program has to be running always so that clients can join and leave anytime. In basic settings this program runs only in Local Area Network. If someone wants to access through internet, then server needs to be hosted with real IP and Port has to be forwarded through router. The software was built using Java 16, Maven, JavaFX 16 for UI and MongoDB for database.

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning the compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The java runtime provides dynamic capabilities that are typically not available in traditional compiled languages. Java released many SDK in years. The SDK used in this program is Java 16 which is the latest one. But in MChat program, Maven module is used for automation and for dependencies.



Maven is a build automation tool used primarily for Java projects. Maven address two aspects of building software: how software is built and its dependencies. Most of Maven's functionality is in plug-ins. In Maven project plug-ins are handled by POM file. POM stands for Project Object Model which provides all the configuration for a single project. Java by default is runs on console. To make the program user-friendly GUI framework is needed. One of the popular GUI frameworks for Java is JavaFX.



JavaFX is a java library used to develop Desktop applications as well as Rich Internet Applications. The applications built in JavaFX, can run on multiple platforms including Web, Mobile and Desktops. JavaFX is intended to replace Swing in Java applications as a GUI framework. However, it provides more functionalities than Swing. Like Swing, JavaFX, also provides its own components and doesn't depend upon the operating system. It is lightweight and hardware accelerated. It supports various operating systems including Windows, Linux and Mac OS. Like Java, JavaFX has many versions. JavaFX 16 is used in this program. JavaFX uses FXML file for UI design. For visual scripting Scene Builder is used.



The program is a chatting software. For IO communication Java Socket programming is used. Java Socket programming is used for communication between the applications running on different JRE. Java Socket programming can be connection-oriented or connection-less. Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used

for connection-less socket programming. The client in socket programming must know two information:

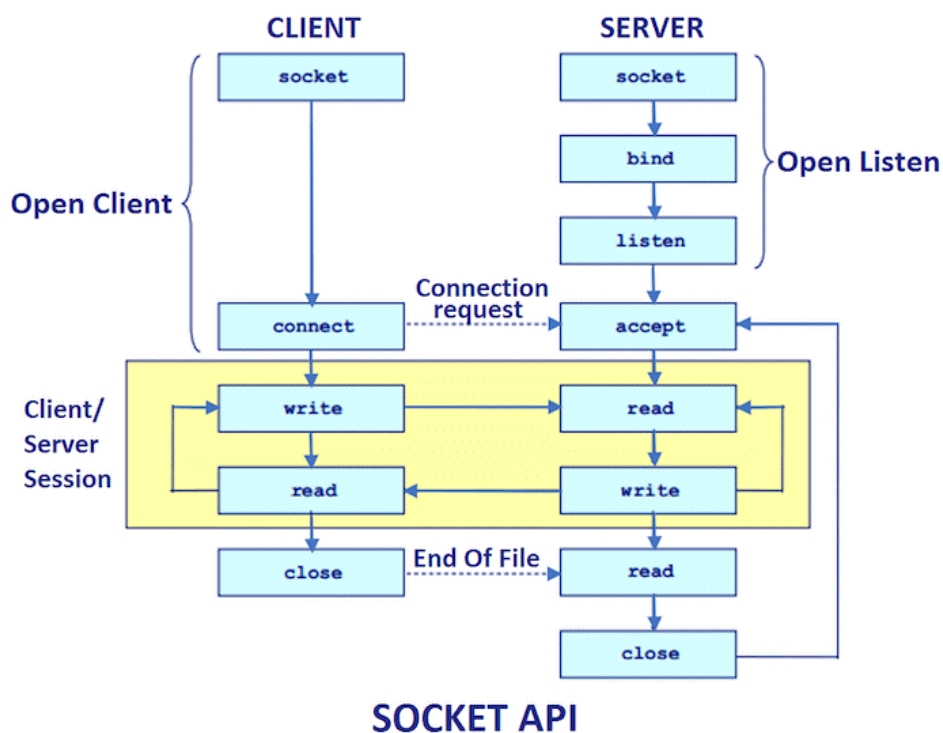
- IP address of Server
- Port Number

The diagram below shows the basic communication of sockets clients and server.

Even though MChat is a peer-to-peer chat software but in the core, there is still

Client to Server communication. In Mchat client program user's use local IP (Default) and server port to connect to the server. While server running, it is always waiting for client socket. When it finds client socket, it accepts it and information

can be exchange between them. And when a user disconnects, client's socket is closed. Also, TCP protocol is used in this kind of communication.



MongoDB database is used in the program for storing confidential data like usernames, emails and passwords of the users. Also, for storing previous messages between users. MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. In MChat MongoDB Atlas is used. It is a web-based NoSQL database and it is always running. It has many security features like IP access, User access etc. Also, database can be modified while database is online or offline. In MongoDB Atlas first a Cluster is created then databases can be added in them. Each database can contain multiple collection. And each collection can have as many as needed documents. For MChat program, MChat named Cluster is created. In this Cluster there are two databases. One is users database and other is chat database. In users database there is a collection called user which holds all the information of the users. Information is hold in BSON documents. On the other hand chat database has a collection called chitchat which holds all the messages of the users and who were they talking to.



IMPLEMENTATION:

Detailed Description:

MChat is Java based chatting and file sharing software. It is a multi-client one server socket program. It was developed using Maven module for automation tools and JavaFX for beautiful and user-friendly UI. The software is basically two parts. One is server part and other is client part.

Server part of the software has a package called com.MChat. It consists four classes.

- Server
- Client_handler
- FileShare_Server
- FileShare_client

Serve class is basically the main class. It starts all the process. It's handshakes with database, waits for clients sockets to be connected and calls Client_handler and FileShare_Server. Client_handler class checks whether is a client is trying to communicate (Sending Message) with other users through server or not. The FileShare_Server does the same work as Server class but it handles file transfer part. It also calls FileShare_client class which works same as Client_handler class. If a client will connect to MChat server and try to send message or file to another connected user, then message or file will come to server first but each message or file has a special key at the end of them which indicates whom to send it. The server deciphers it and send it to them. Server also keep a copy of file in the host computer and messages in database.

Client part is the part which will be used by the users. It has a user-friendly UI for easier use. JavaFX is used in it by FXML file. But as FXML file needs a controller class so it was created. Client part has two packages and a text file.

- java.com.mchat
 1. App class
 2. Controller class
 3. Login class
- resources
 1. CSS
 2. Images
 3. LoginUI.fxml
 4. UI.fxml
- Catalog.txt

Client part also consist of two windows.

- Login and Signup window
- Chatting and File Sharing window

Login and Signup window:

java.com.mchat has three classes. First one is App class. It is the main class. It starts the whole program using public static void main. It inherits the Application class which is need for JavaFX. It loads UI of Login page from LoginUI.fxml using FXMLLoader. As a result, login window pops. The users can either login using existing email and password which is previously loaded in database or can create new account using signup option. For first time user default download location needed to be set. Also, users can connect to any server they want by changing IP, port and file port from the settings. This program is not limited in Local Area Network. It can be accessed from anywhere in the world by using internet. Just server needs to forward its port to the router.

Chatting and File Sharing window:

After pressing login or signup new window will pop with all the friends are online. This is the chatting window. To chat with someone user just has to select and friend from the friend list and chatting panel will slide in. In the panel there is a scroll pane which consists of messages and images. If there was conversation before then those will be loaded automatically. Below that, there is a horizontal box with two buttons and a text area for writing message. Button on the left is file select and send button and right one message send button. There are some options on navigation panel in the left.

- Friend list: A list where users can see who is online and who is offline.

- Add Friend: In this option users can search for friends and send them friend request. They can also accept available friend requests.
- Settings: In this option users can change the default download location.
- Logout: Users can log out and sign in using another account.

These are all done in the front-end of the software. But in the back-hand part Controller class does all the work. In the Controller class there are all the action events for UI.fxml. When chatting and file sharing window pops Controller class is called. First it initializes database for previous messages and handshakes with the sever for communications. By default, server is on locally. So, the IP is localhost (127.0.0.1). For messaging (1234) port is used and for file sending (5555) port is used. Client connects to server using them. Also, after connecting to the server client sends it client name to the server and server updates it in database showing that client just became online. As a result, other users can see him/her online in the friend panel. After handshake is done client starts a thread for receiving message and one for receiving files with socket's Input data stream. If any data comes from the server, they will accept it and show it in scroll pane and the file will be downloaded. If the file is in PNG or JPG format it will show in scroll pane also. For sending message, message send button needs to be pressed. When it is called text from the text area will be sent to the desired user. While sending (#username) will be added with the message. This will help the server to identify whom to send the message.

File sending is different from sending message. Because file needs to be turned into byte first then it is sent through the output steam. Also, for sending file first file name is sent with who user wants to send. Then file size and lastly file itself.

This file while be first received by the server then it will be resent to the specified user.

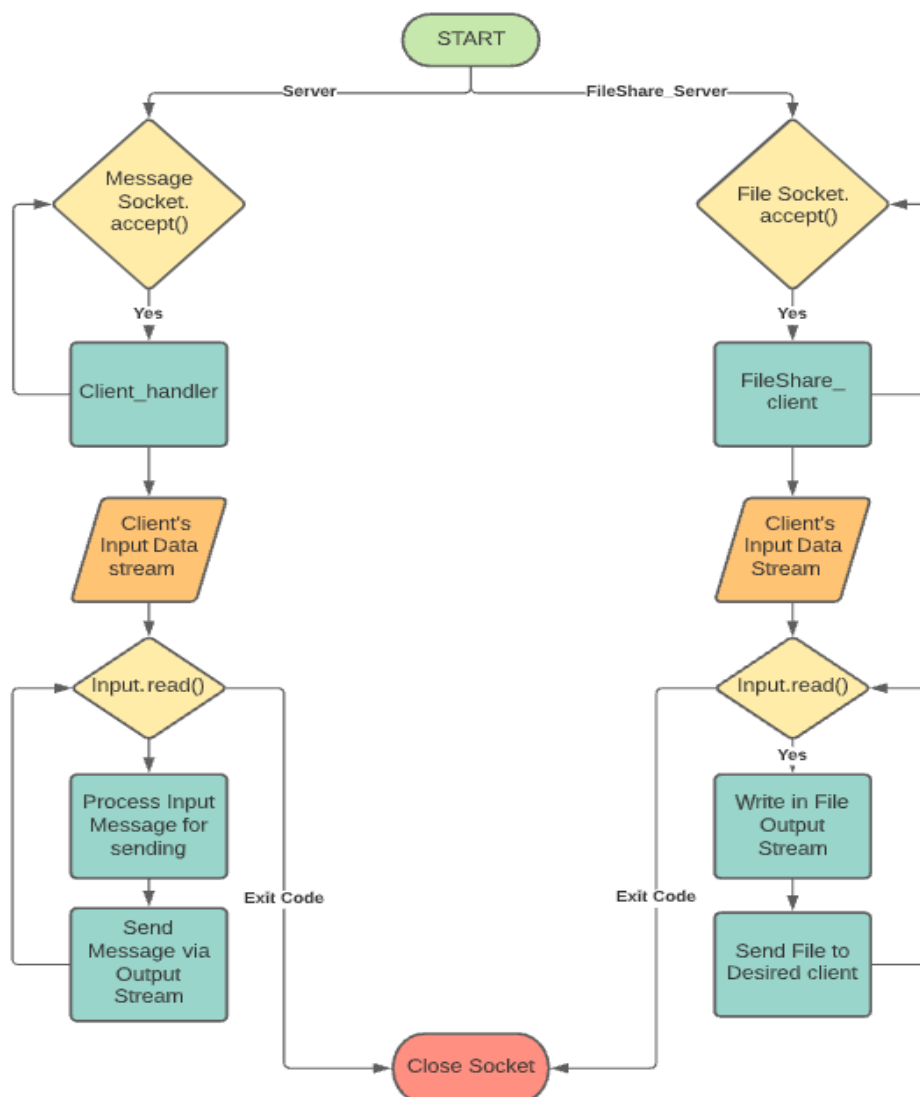
On the receiving side, messages will be received normally. But for file receive, file name will be received. After then file size. This will help the

program to allocate space and end of the file. Then it will start receiving the file itself. By using this method file transfer was done with great speed. About 30 MBPS. At the end if a user wants to log out or close the program a special code (%%\$\$&&) is sent to server notifying it to close the socket for this user and turn off all input and output data stream.

Flowchart:

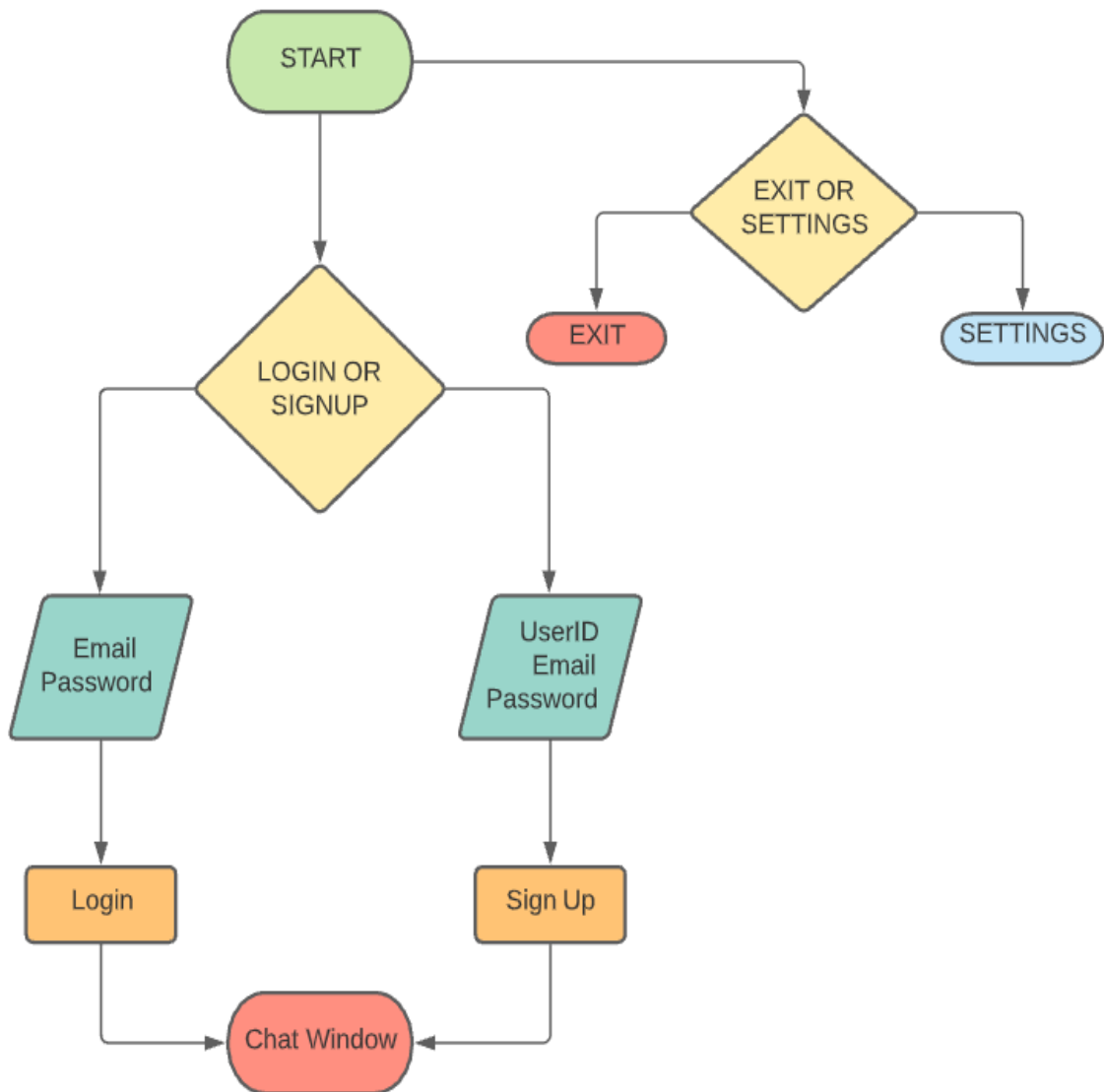
- **Server process:**

This process will start when server program is started.



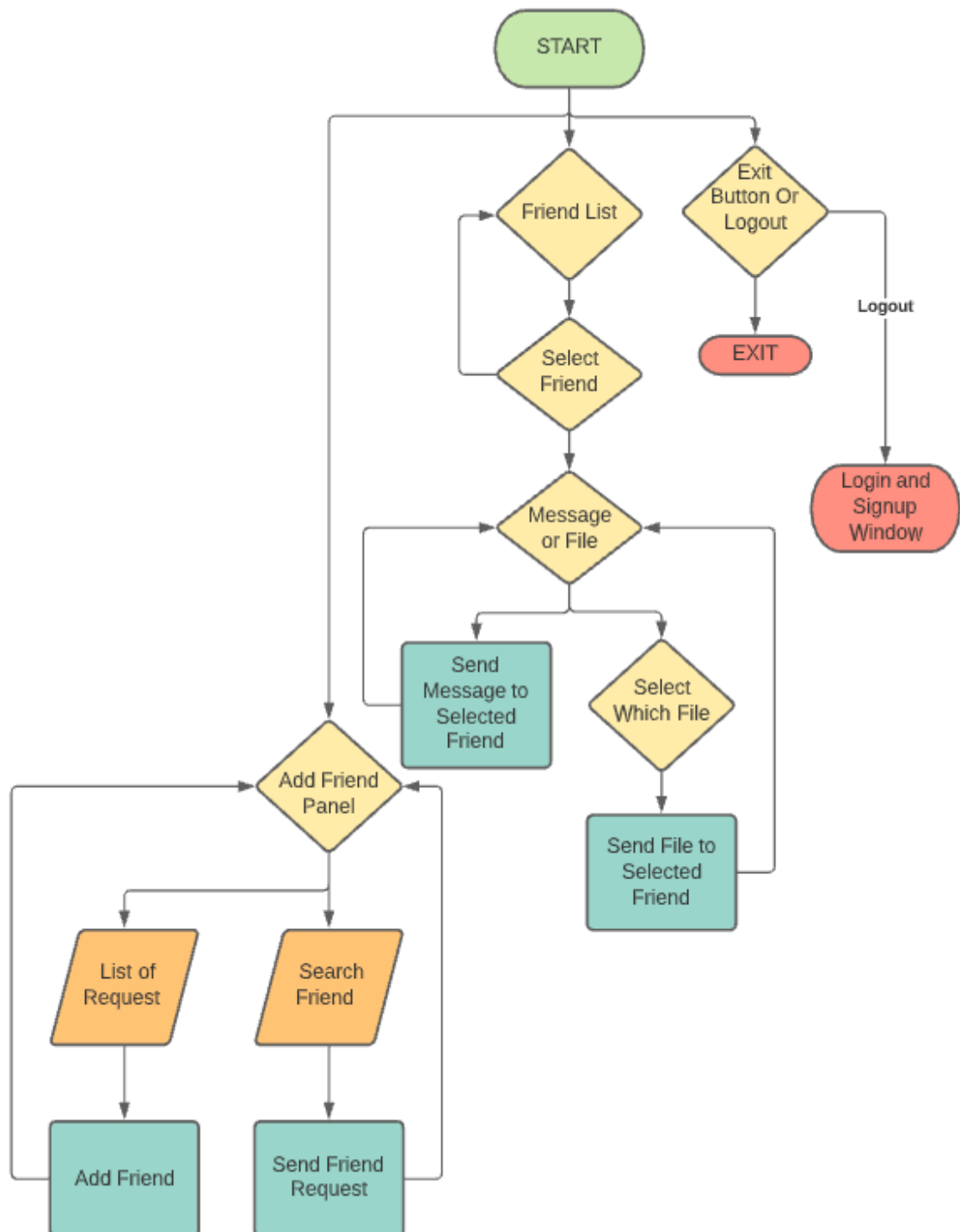
- **Login and Signup process:**

In the client program first login and signup process will start. After pressing login or signup chatting window will pop and this will be closed.



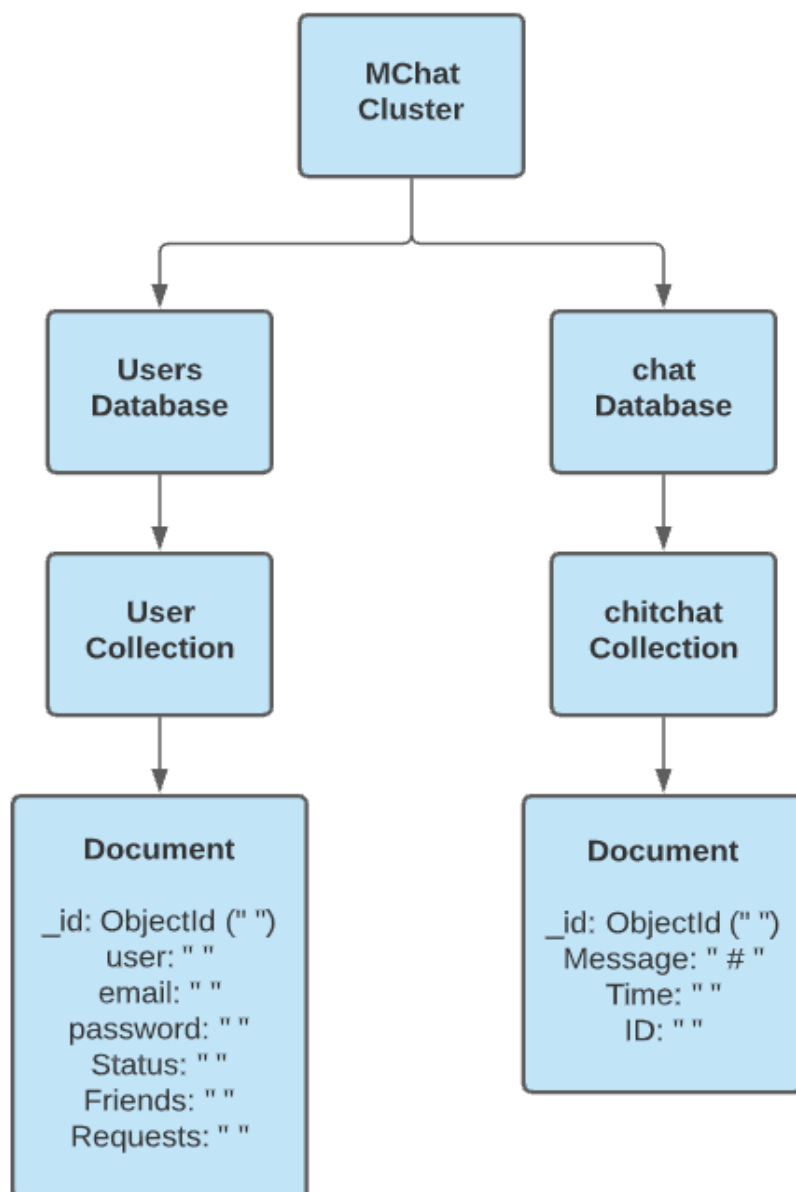
- **Chatting and File Sharing process:**

This process will start if login or signup process was success. In this process client can chat or send file to other clients.



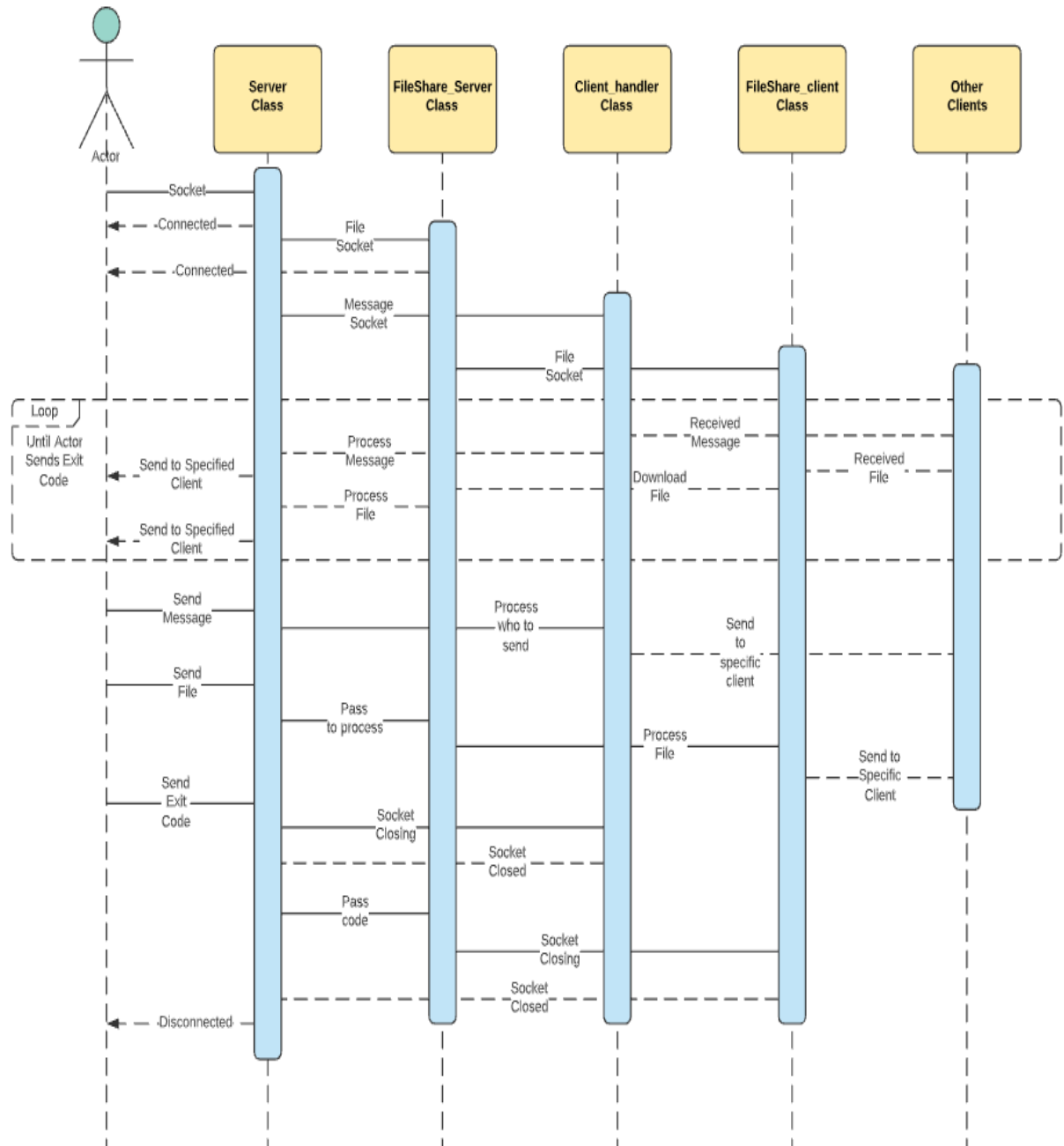
Schema Diagram:

MChat software uses MongoDB database for storing confidential data and previous messages. In MongoDB there is cluster called MChat. It has two databases. One is Users which has User collection. It holds confidential data. Other is chat which has chitchat collection. It holds previous messages.

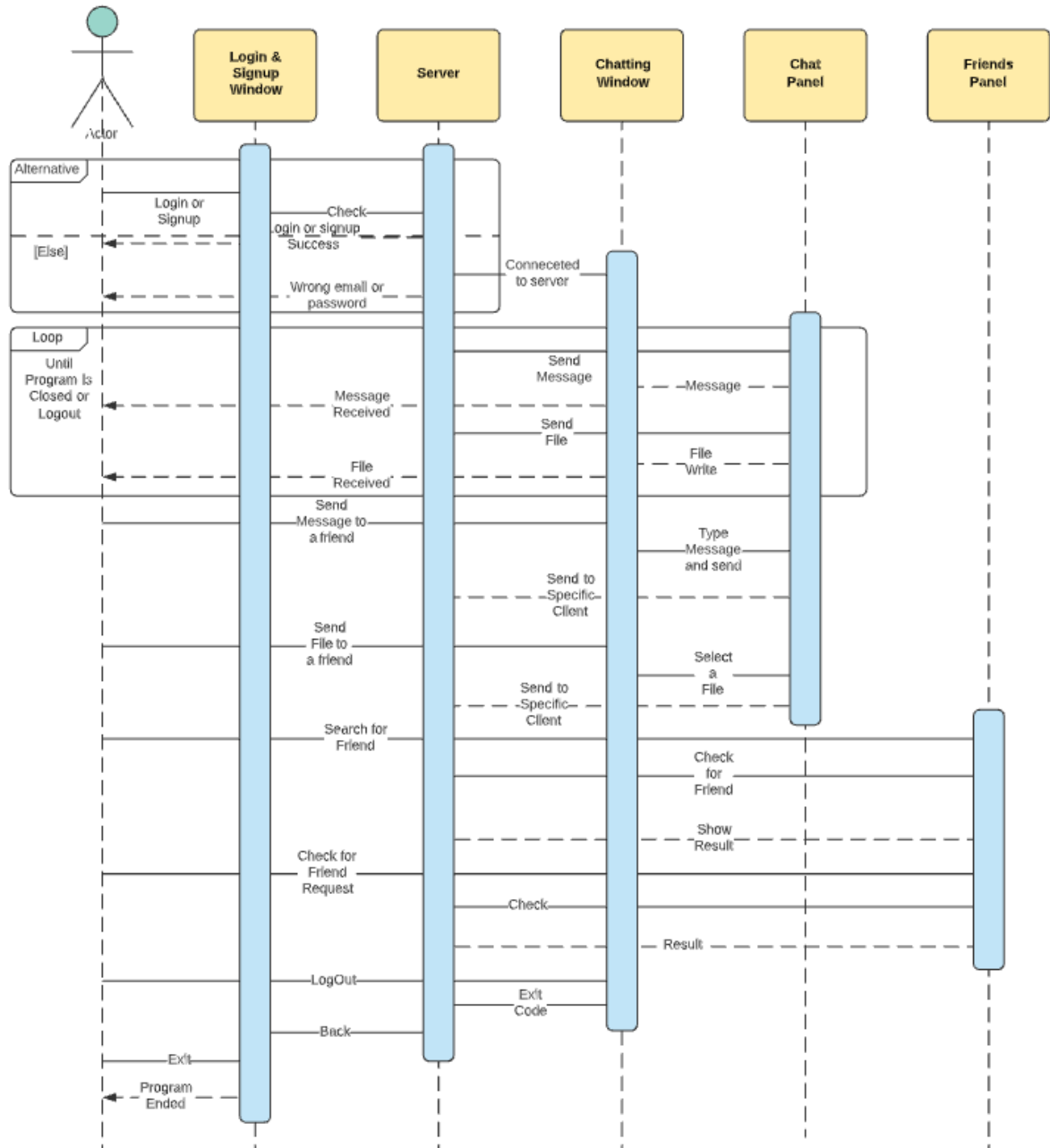


Sequence Diagram:

- Server Program:



• Client Program:



Detailed Visualization:

- **Server Program:**

The server program is basically a console program as it doesn't need any user inputs and it also doesn't need setup. Just run it and it will handle everything by itself. At start it waits for client socket.

```
"C:\Program Files\Java\jdk-16.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.1.
Jun 22, 2021 9:26:30 PM com.mongodb.diagnostics.logging.Loggers shouldUseSLF4J
WARNING: SLF4J not found on the classpath. Logging is disabled for the 'org.mongodb.driver' component
Waiting
```

When clients connect to server, server notify it by a message in the console.

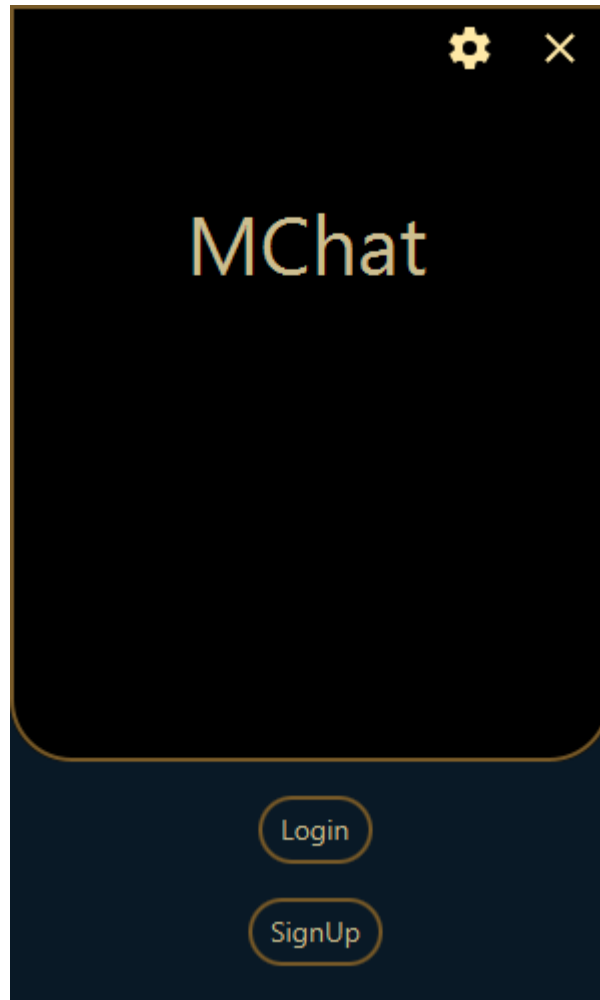
```
Connection Established
Connection established with Socket[addr=/127.0.0.1,port=60708,localport=1234]
Waiting
Waiting to receive
Client Socket[addr=/127.0.0.1,port=60708,localport=1234]is online now
```

It shows client which is connected and its Port. When clients log out (%%\$\$&&) is sent to server and again server waits for clients.

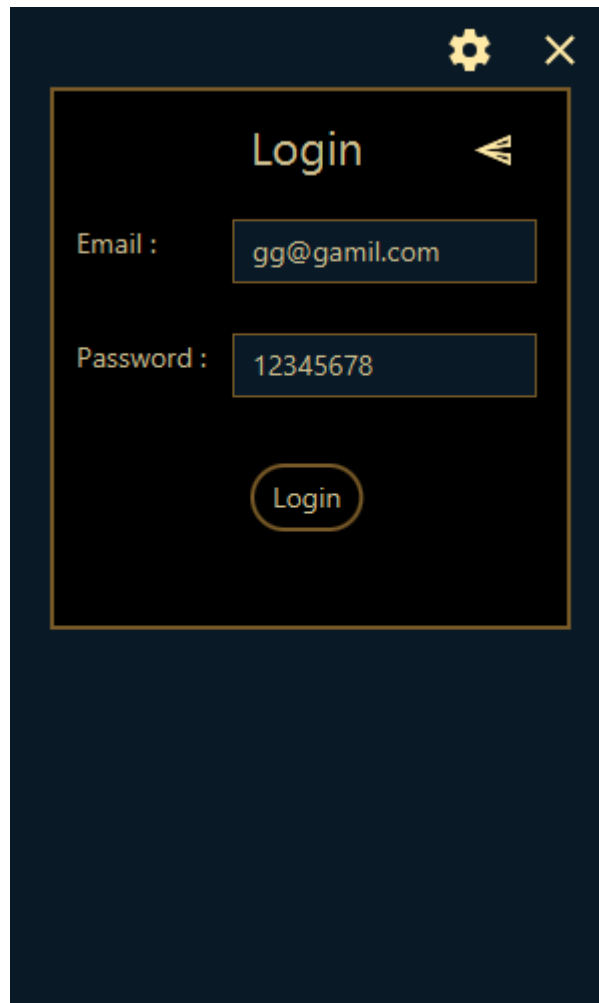
- **Client Program:**

Client program has two parts. One is Login or Signup window. Other is Chatting Window.

While starting Login or Signup window, a window pops up with two options Login and Signup.

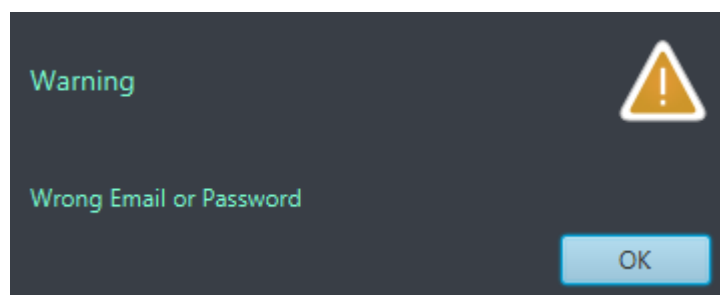


There is also Settings and Exit button. If user press Login button, a login panel will show where user can type their email and password. This email and password will be checked with database.

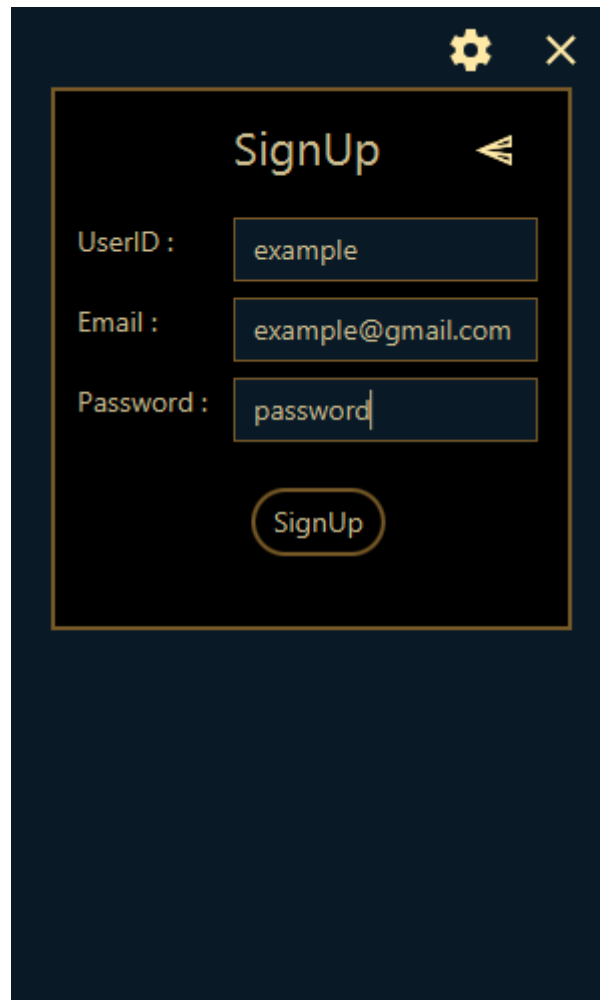


A dark-themed login dialog box with a title bar containing a gear icon and a close button. The dialog has a title "Login" and a back arrow icon. It contains two input fields: "Email :" with the text "gg@gamil.com" and "Password :" with the text "12345678". Below the fields is a rounded "Login" button.

If it matches then users can go to chatting window. If not, a message will pop say email or password is wrong.

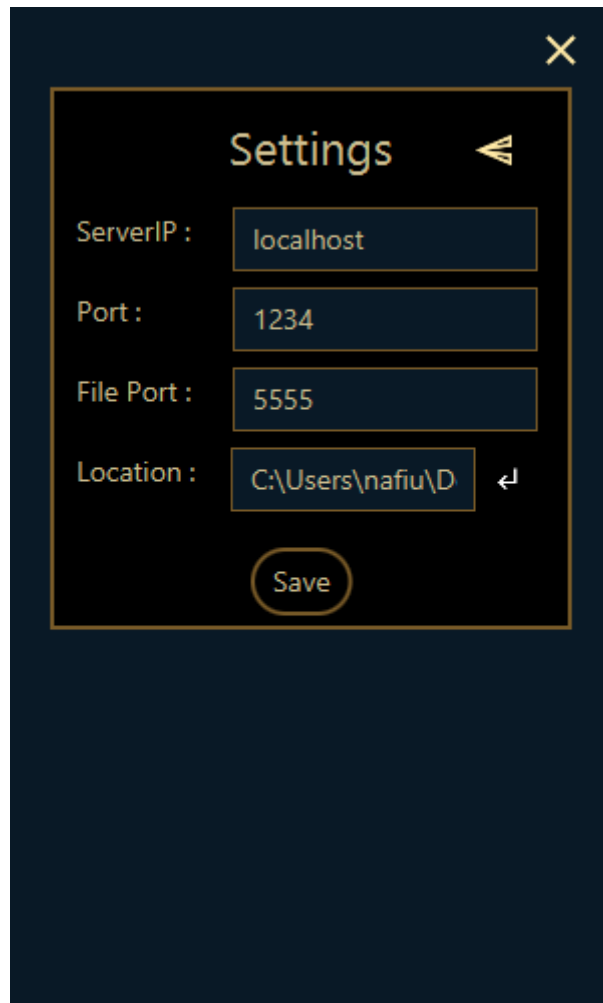


If users press signup, a signup window will pop. There users can sign up using unique user ID, email and password.



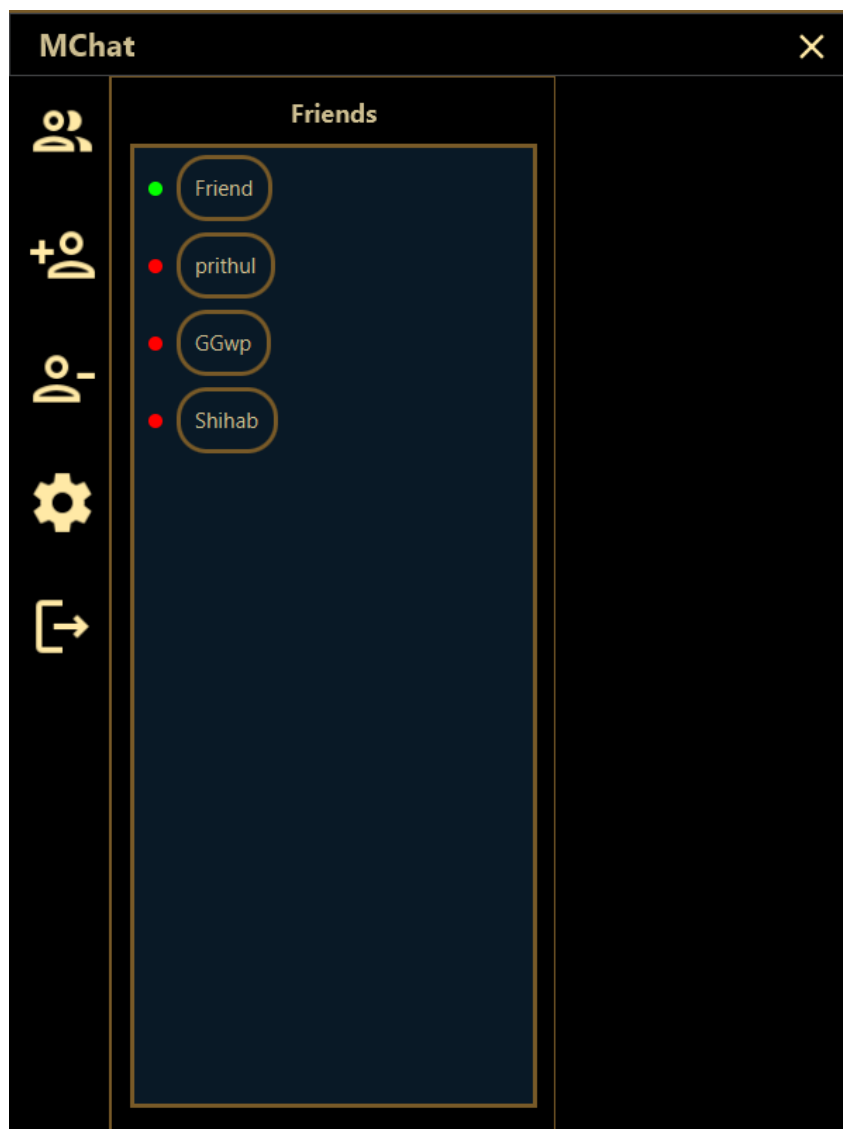
To sign up username needs to be at least four letters and unique. Email needs to be unique too. Also, password needs to be at least six letters. After creating new account, this will be saved in database. So next time user can just login through login window.

There is also settings panel where users can change the IP, Port, File Port and Default location. This will be saved in the catalog.txt in source file of the program. So that it can run on any computer without need to change the source code.

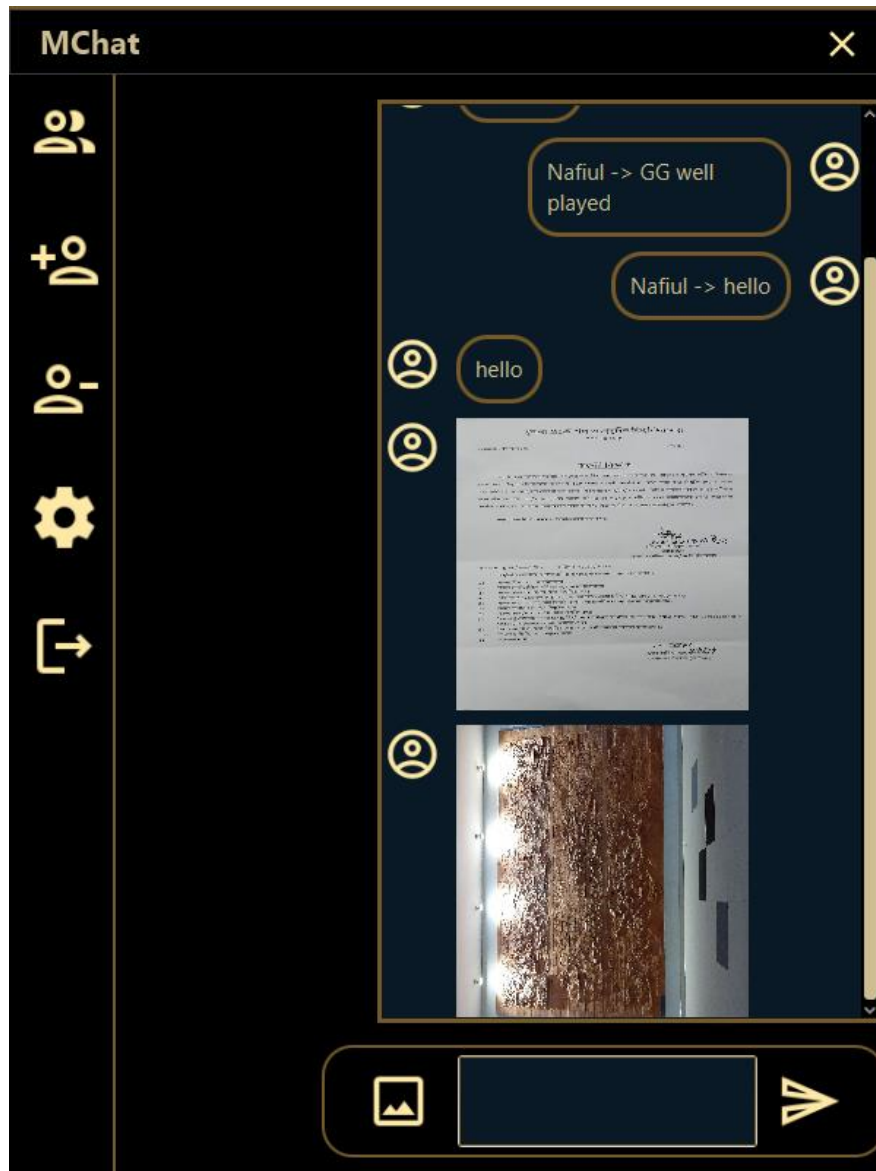


This is the full visual representation of the Login or Signup window. After Login or signup successfully, a new window will pop. This is the Chatting window. It has four parts. Chatting panel, Friends panel, Add friend panel and Settings panel.

While start up Friend panel will show. There users can see who are online and offline. This is a real time process. Program refreshes it every five second and check for any changes through database. As database keeps record if someone became online or offline.

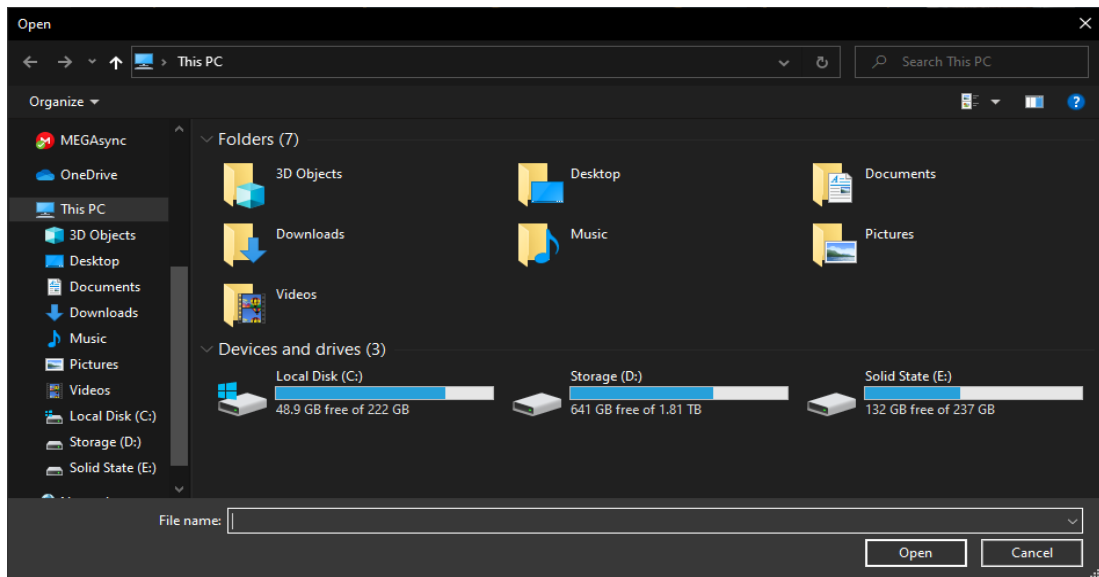


By pressing on any online friend users can start chatting with them via chatting panel.

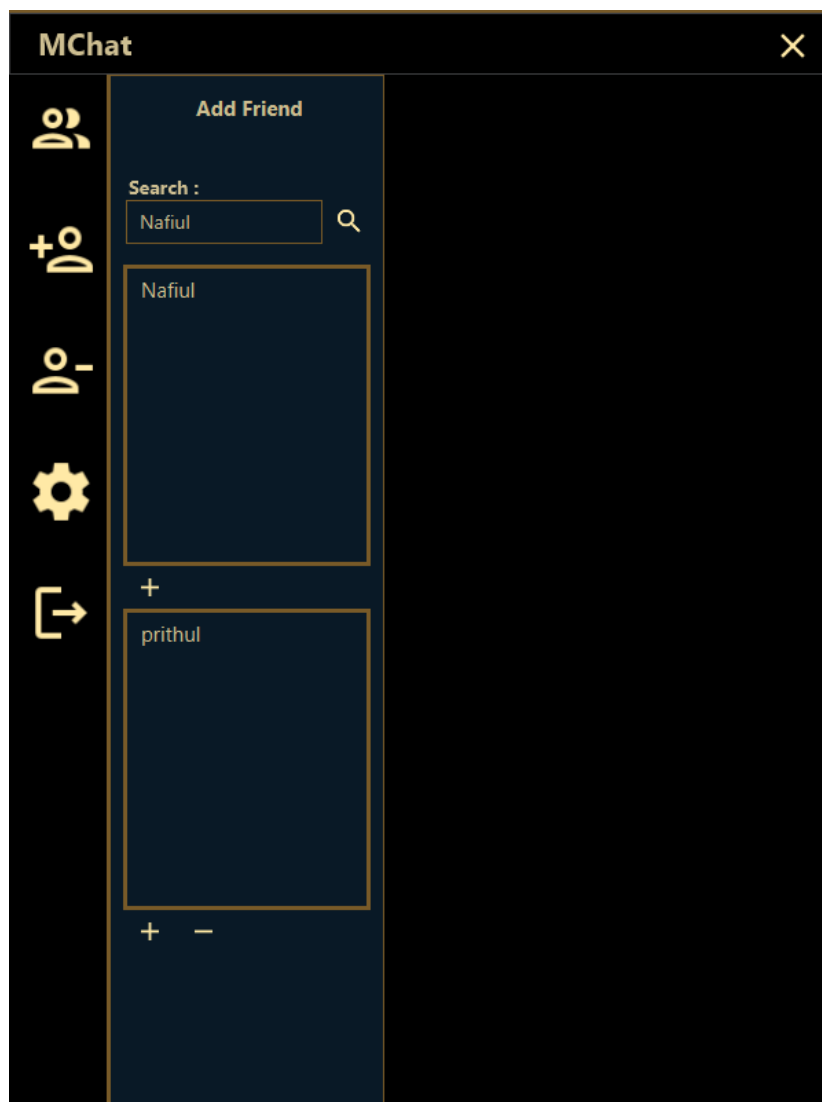


Previous messages and images between two clients will also show in chat panel.

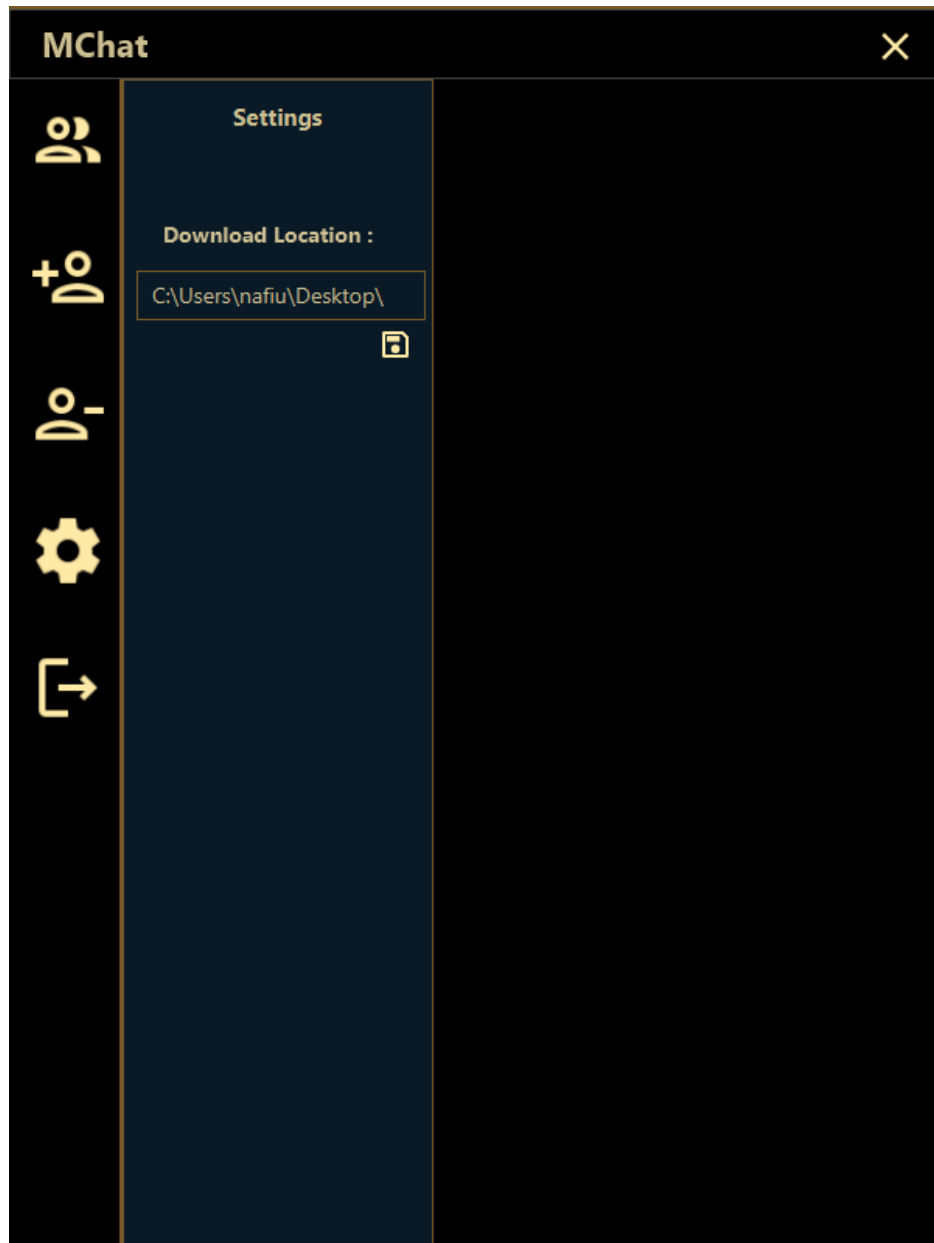
There is text area for chat and two buttons. Left one for file sharing and right one is for sending message. If user press file share button, a pop window will come where user can select files.



If user press Add friend panel button, it will slide from left.



In Add Friend Panel, there is a search bar where user can search for new friends and send them friend request. Below that there is a friend request panel where users can see if they got any friend request or not. There is also a settings panel where users can change their default download location.



This concludes all the features of MChat program with visual representation

TARGET VS ACTUAL ACCOMPLISHMENT

MChat is a chatting and file sharing software. Target was to make it as good as WhatsApp which is one of the best chatting software all over the world. But WhatsApp is mainly used in Android but MChat is a Desktop software. So many things can't be added to it. List below shows all the targets.

Target:

- Making a chatting and file sharing software using socket programming.
- In the software users can chat with another client through server.
- User's confidential data's will be saved in database.
- User can use the software from anywhere in the world using internet.
- User can send not only images but also large files to another client.
- Previous conversations between users will be saved in database. So, when the login, it will be already available in the chat window.
- Make it user-friendly like WhatsApp.
- Ability to add and block friends.
- Ability to send and get friend request.
- User can make groups and friends in it.
- Make individual chat room for each user.
- Give the users ability to either download or not the sent file.
- Add two step verifications for better security.
- Users can add and change their profile images. It will be saved in database.
- Make the software deployable with setup file. So that it can be used as an alternative for Whats App, Telegram etc.

Accomplishment:

- Making a chatting and file sharing software using socket programming.
- In the software users can chat with another client through server.
- User can't get individual panels for each friend. All chats are in one panel.
- User's confidential data's will be saved in database.
- User can use the software from anywhere in the world using internet.
- User can send not only images but also large files to another client.
- Previous conversations between users will be saved in database. So, when the login, it will be already available in the chat window.
- It has ability to add friend but not remove.
- Ability to send and get friend request.

From above we can see that even though many things have been added to actual software, there are much left. From the target only have was able to be included in the software. The main function that could not be included was individual chatting panel for each friend. The software right now can chat with individual clients but their messages or files will show in one window. Also, there is no additional security beside Login and Registration.

RISK AND ISSUES

MChat has some risks. Only security in MChat is the login and signup check. If someone can bypass this then he/she can access the server easily. Also, as server's IP and Port is available to public it can be accessed easily. But he/she will not be able to intercept any messages. Because when client connects to server, it also sends a special code that helps server identify who is trying to connect. Also, a moderator can see who is trying to connect to the server. Another risk of the software is there is not file scanning system. Also, the file auto downloads. As a result, virus affected files can be easily sent. There are issues with the software too. They are,

- Even though MChat is a peer to peer chatting software. There is no private room for each friend. Messages can be sent individuals but messages received shows in one window.
- Files can be shared easily but on receiving side file will be auto downloaded.
- There is no additional encryption beside login and sign up.
- If the client crashes, then server will crash also. Because unless exit code is sent to server, server will try to connect to the client. But as client crashed, server will get in error.
- UI needs to be more polished.
- Add Friend panel has some bugs like even though two clients are friends it will still accept friend request.
- Add friend works but it has no significance as in friend list user can chat with all the users.
- Remove Friend panel don't work.
- On a new computer user have to set the default location manually.
- If user wants to connect to other server which is out of Local Area Network, then he/she has to type the IP every time they start the software.

- There is no standalone executable file. The program has to be run on an IDE.
- If someone wants to access the software from internet then server program needs to be run on a network where port is forwarded to router.

DISCUSSION AND CONCLUSION

MChat was made to compete against existing chatting software like Telegram, WhatsApp, Discord etc. Unlike its competitors, it is still in its early stage. It is a great chatting and file sharing software in Local Area Network. Even its compatible outside of LAN, but the server program needs to run on Real IP network and Port needs to be forwarded. Also, the program is not too user-friendly. The program can crash if sequence is not maintained. The server program is a console program. Also, it can see who is connecting and disconnecting. But it has no control of itself. In future version it can be added. With this security will further increase.

But still with available features the program can be used without any issues.

In making of MChat, Java's many core parts are learned. Like, Java basic, use of Maven module, use of JavaFX for making UI, access and use MongoDB database in Java project, for communication Java Socket programming and many more. Making this software many problems are faced like in chat panel custom scroll pane is used. Because traditional Listview in JavaFX only supports one type of data but chat can be message or images or files. Also, it was difficult to control the download and upload speed. But it was solved by limiting the buffer size. Another problem aroused when program was hosted on internet for anyone to access. But by forwarding the port it was solved. Even though MChat couldn't reach its full potential but it is still usable for experimental uses.

REFERENCES

- Documentations of different Java classes from Oracle (<https://docs.oracle.com/en>)
- JavaFX documentations from Oracle/JavaFX (<https://docs.oracle.com/javase/8/javase-clienttechnologies.htm>)
- JavaFX CSS reference guide (<https://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html>)
- MongoDB Java Documentations (<https://docs.mongodb.com/drivers/java/>)
- Java Socket Programming (<https://www.javatpoint.com/socket-programming>)
- Tutorialspoint JavaFX Tutorial (<https://www.tutorialspoint.com/javafx/index.htm>)
- GeekforGeeks Java Tutorial (<https://www.geeksforgeeks.org/java>)
- Video tutorials (<https://www.youtube.com>)