touch.inc

flyBird

Wi-Fi Communication platform

Final Report

Version 1.0

Table of Contents

1. Introduction

1.1. Purpose

This document is expected to provide the details about the project on a view of after its implementation iterations of Wi-Fi Communication Platform project.

Expected audience of this document is Users of the system, System evaluators, and Development team. This document provides the experiences gained from the project and it is an important thing to keep track of them for further versions of the system.

1.2. Scope

This Document provides the main details about the system. This covers from expected functionalities to the implementation strategies of the system with challenges and experiences got from the project.

1.3. Definitions, Acronyms, and Abbreviations

Term	Description
LAN	Local Area Network
OS	Operating system (Only Windows 7 or above in current system version)
User	Person who use this application
PC	Personal Computer, a computer which runs a windows operating system
UI	User Interface

2. Project overview

This application is about sharing information and do conversations wirelessly in a local area network. Main concerns here are no data charges, simplicity, sudden usage.

2.1. Motivation

Nowadays people are using devices like laptops and mobile phones that has Wi-Fi functionality. But in computers it is still been used Bluetooth stock application if needed to share some files wirelessly. It is slow and not a much user-friendly process. If this is done by Wi-Fi it will be faster than Bluetooth. Another problem in such applications that are already exist ask for another connection for that. But in this system, will be developed to use sharing functionality without having another network connection by using Wi-Fi hotspot of a using device.

2.2. Users

Main user roles:

1. Sender
2. Receiver (in a conversation a person can act as both)

Anyone in the same network can use this system without any registration.

2.3. Expected functions

It acts like a normal messenger application except it works in LAN without data charges and it limited into the LAN.

Main functions:

1. Register user profile	2. Edit user profile
3. Create a hotspot	4. Connect to the network
5. Sending text messages	6. Receiving text messages
7. Sending files	8. Receiving files
9. Send voice (Addon function compatibility)	10. Receiving voice (Addon function compatibility)

3. Use cases and System requirements

3.1. System requirements

3.1.1. Functional requirements

3.1.1.1. Add user data to the profile

A user should be able to add the data (name, profile picture) to his/her application profile. When the user starts the application at the first time it he/she needs to add his profile for other's recognition purpose. This is not compulsory. User should be able to use the application either without making the profile.

3.1.1.2. Edit user profile

A user should be able to edit his/her profile details. If the user wishes to change his profile picture or his name, there should be the functionality to change them.

3.1.1.3. Create a Wi-Fi hotspot.

A user should be able to create a Wi-Fi hotspot to make a network if they are not already in a network or they need to create a private network themselves.

So, when trying to create a hotspot system will warn about connected networks and if the user proceeds to create it will be created. This happens every time when the user needs to use the app if they are not in a network.

INPUT: hotspot name and hotspot password

OUTPUT: Confirmation with Hotspot name and password

3.1.1.4. Stop created Wi-Fi hotspot.

User should be able to stop created Wi-Fi hotspot when the communication has finished and if user wish to close the hotspot.

3.1.1.5. Connect to a Wi-Fi network

A user should be able to connect to the required Wi-Fi network using this system. When users need to communicate, they should connect to a network. It can be a created hotspot or a Wi-Fi network they have in the premises.

3.1.1.6. Disconnect a Wi-Fi network

A user should be able to disconnect a Wi-Fi connection using this system. This can be happened after the communication ends and if the user wishes to disconnect the onnected network.

3.1.1.7. See the connected devices (as contacts)

A user should be able to see the connected devices in the network as contacts names. If contacts names haven't been added their IP address and MAC address should be shown. Before starting a conversation, this happens for select the needed receiver at another end.

3.1.1.8. Start a private chat

A user should be able to start a chat with connected users to the network. This is done by selecting the required receiver from contact list.

3.1.1.9. Start a group chat

A user should be able to create a group chat with the connected users in the network. This is done by selecting the receivers from contact list.

3.1.1.10. Send a text message

A user who has joined to a chat with another user or a group should be able to send text messages. Text is typed in text box in the chat UI. It should be done by pressing the send button in the chat UI.

INPUT: Text message

OUTPUT: send to the selected receiver/group of receivers

3.1.1.11. Send files

A user who has joined to a chat with another user or a group should be able to send file messages. (without file type constraints). It can be multiple files or one file, should be at least 5 files in one row. It should be done by pressing the send button in the chat UI.

INPUT: Any kind of file type

OUTPUT: Send to the selected receiver/group of receivers

3.1.1.12. See old messages

A user should be able to see old messages that he/she have received and sent. By selecting another contact or group from the contact list.

3.2. Use case view

Use case diagram of the system is showing here.

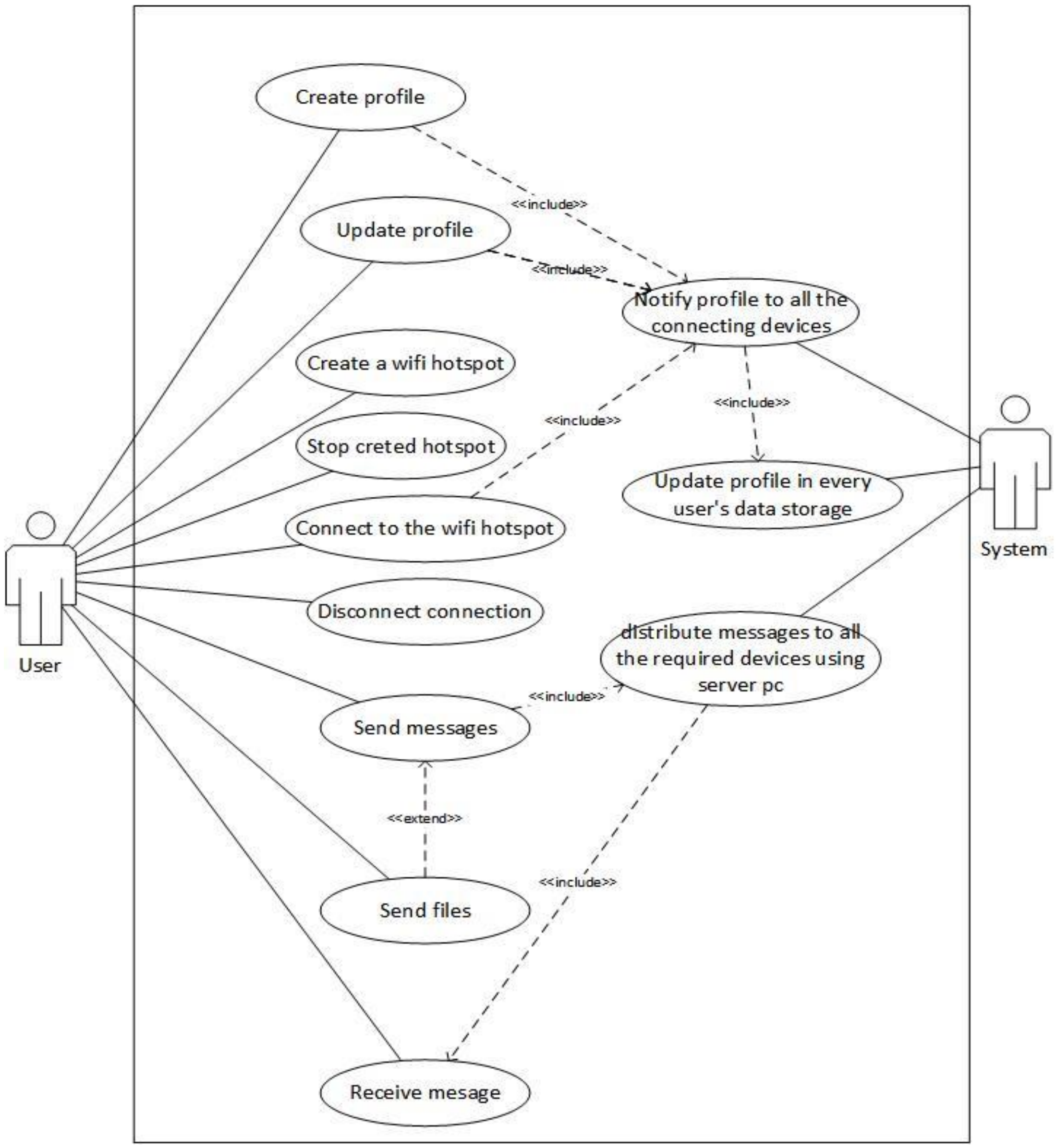


Figure 1-Use case diagram

4. System Design

4.1. Architectural design

Architecture of the system is described using class diagram, package diagram and a deployment diagram.

4.1.1. Class diagram

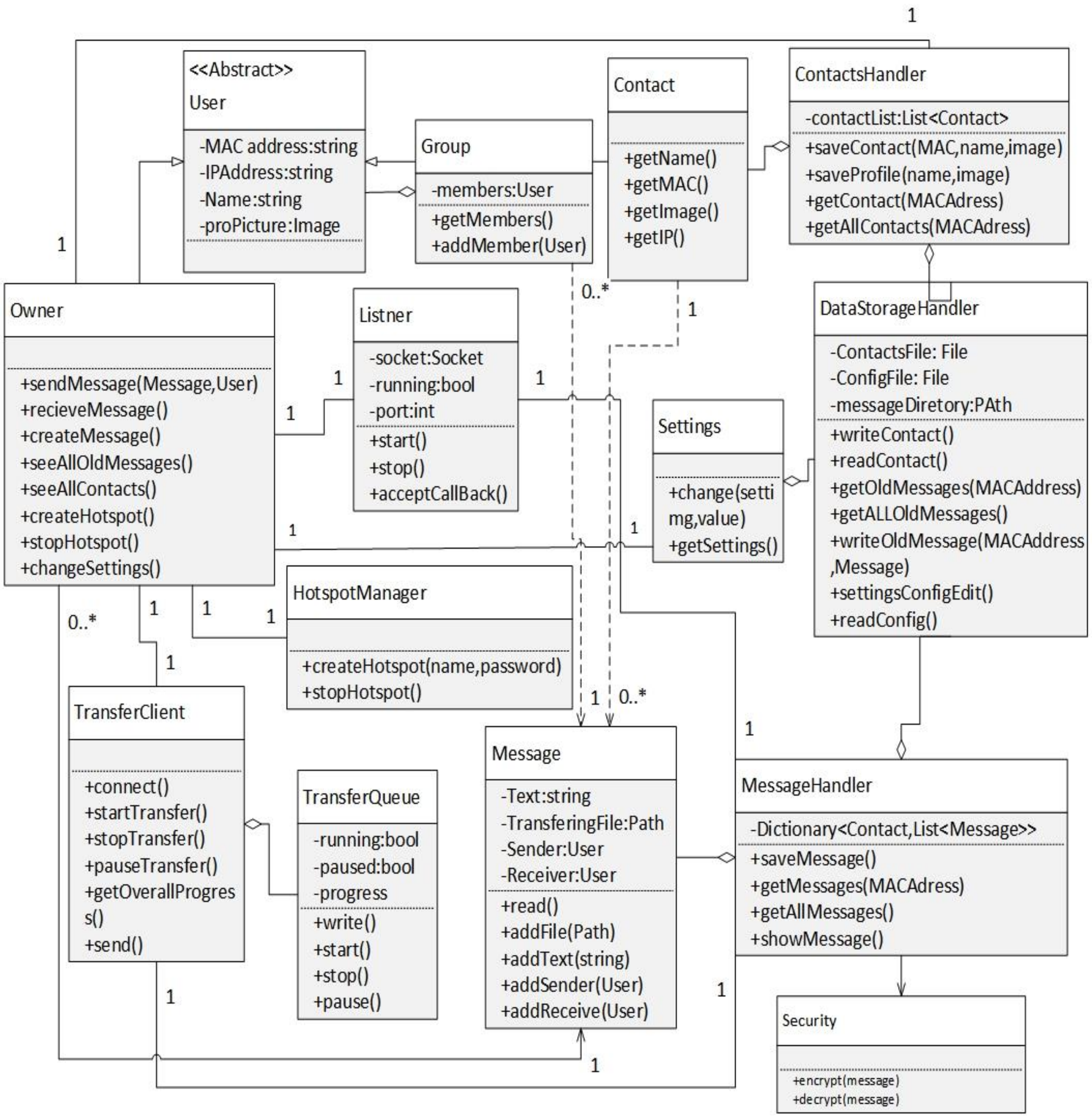


Figure 2-Class diagram

4.1.2. Package diagram

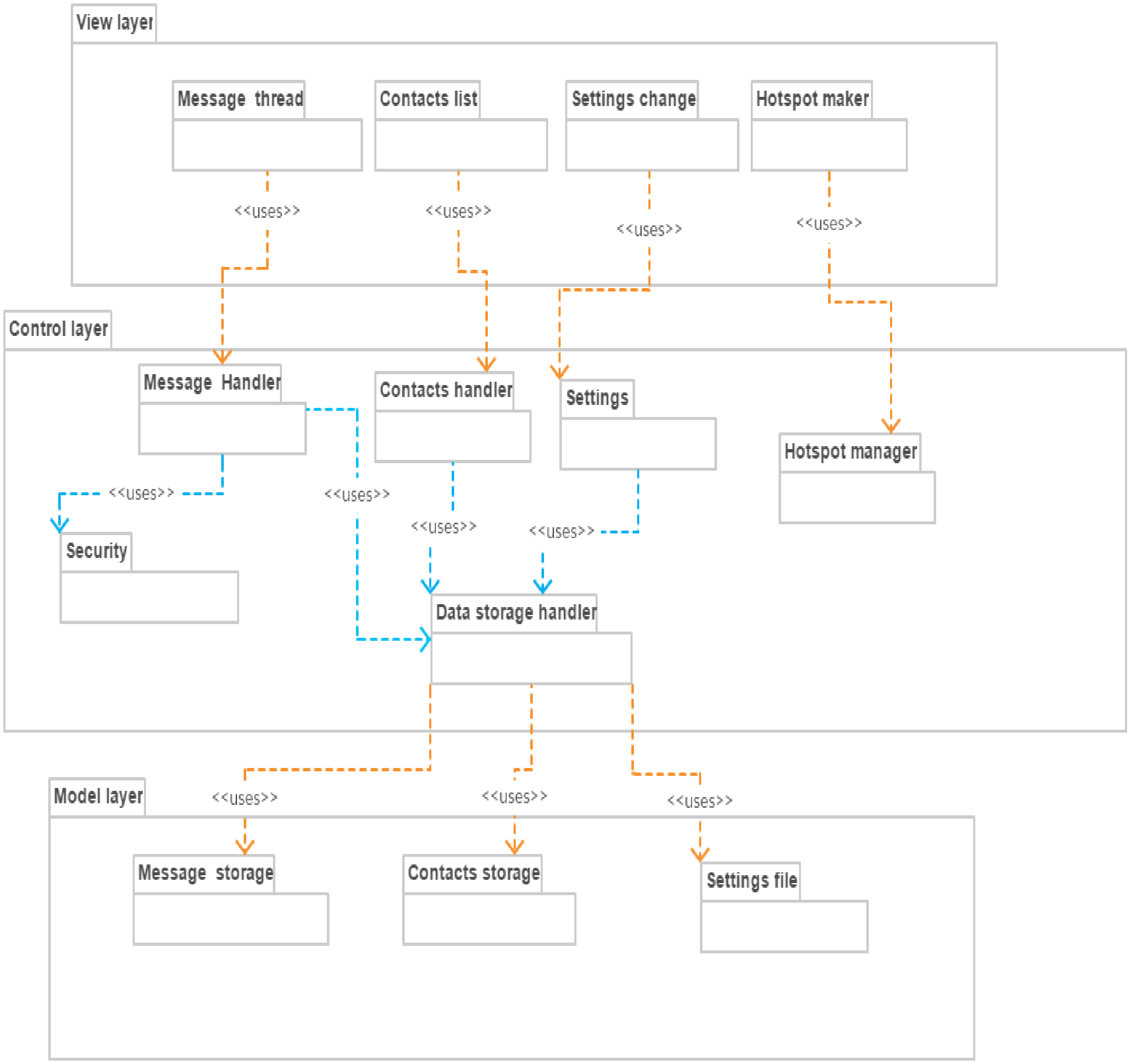


Figure 3-Package diagram

4.1.3. Deployment diagram

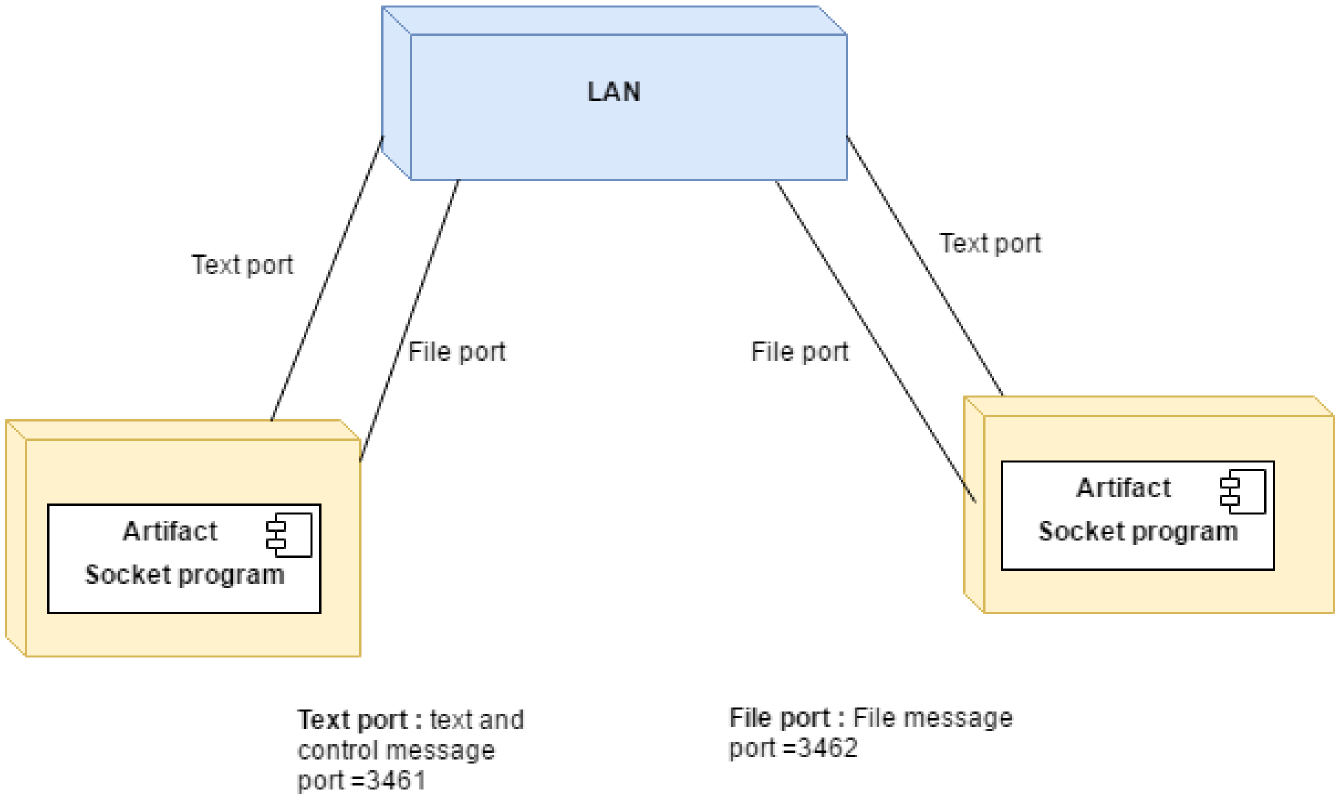


Figure 4-Deployment diagram

This application uses two ports to communicate via LAN. Those are for,

1. Text and control message sending and receiving.
2. File sending and receiving

   It is easy and reliable to use a different port for the file transfer.

5. System implementation

5.1. Tools and technologies

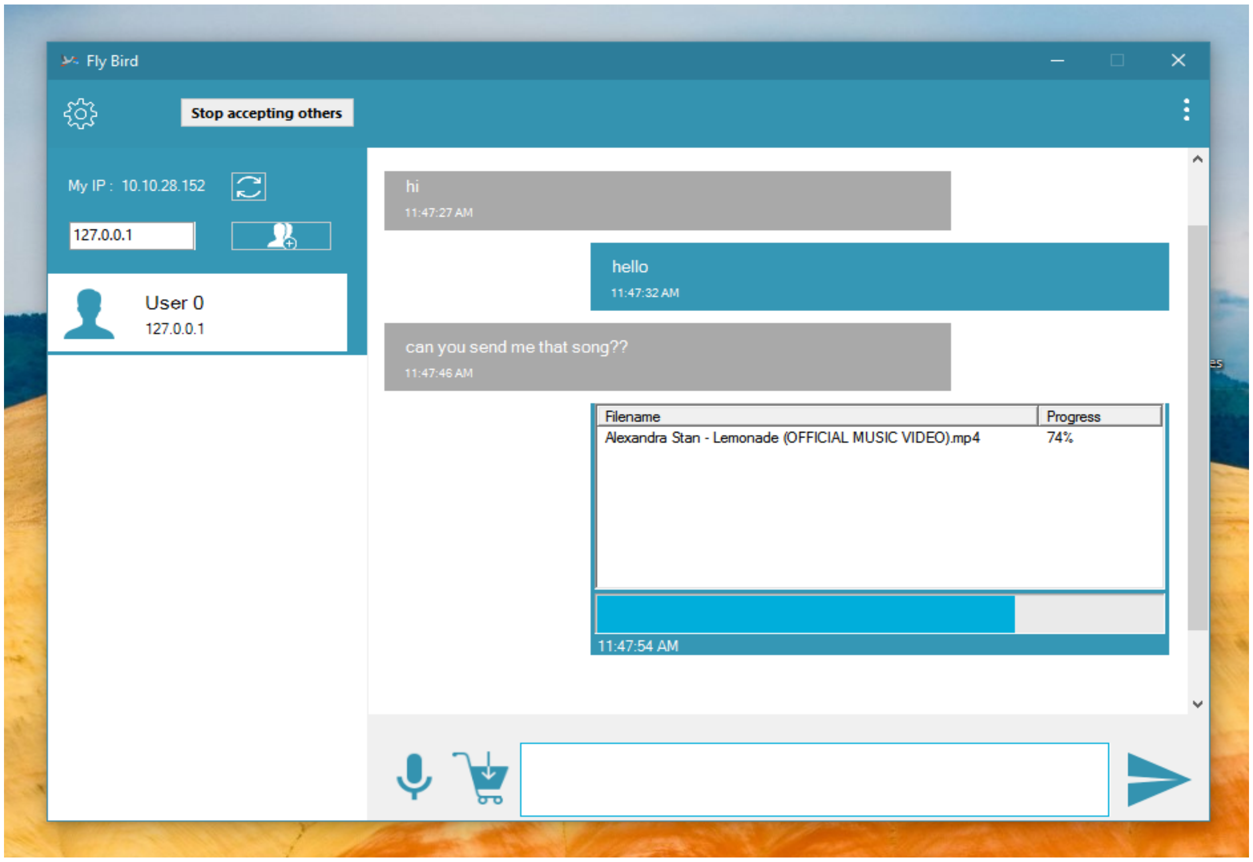
This system has developed using:

1. C# [1]
2. Visual studio 2017[2]
3. Metro UI framework [3]
4. GitHub [4].

Using C# language, it can be easily implemented the network handling part of this system and visual studio has used as the IDE.

Metro UI framework has used to create a user friendly smooth User interface. GitHub is used for version control.

5.2. User interfaces



hi
11:47:27 AM

hello
11:47:32 AM

can you send me that song??
11:47:46 AM

Filename	Progress
Alexandra Stan - Lemonade (OFFICIAL MUSIC VIDEO).mp4	74%

11:47:54 AM

Figure 5-Chat display UI

6. Project performances

System has successfully implemented the functionalities up to 85% of the planed functions.

Reliability and performance optimization has to done for better performance of the application. It is expected to improve the application in the next version.

Project has finished on the time keeping effective functions and performance in a good state.

7. Testing

This system has been testing from the beginning. It is done with every function adding. After completion of the system it is expected to test performance testing and failover testing.

Testing is done according to these topics,

- Data and Database integrity types
- Function Testing
- User Interface Testing
- Performance Profiling
- Load Testing
- Security and Access Control Testing
- Failover and Recovery Testing
- Configuration Testing

Testing tools given by Visual Studio has been using for the system testing needs.

- MSTest
- Visual studio
- Performance profiler
- Diagnostic tool

8. Lessons learned and experiences

8.1. Issues and Challenges

There were some issues with this project.

- For implementation, the application mainly the lack of knowledge on the field of socket programming was arisen. In designing part also was not easy with the un familiar socket technology.

  It has to redesign the architecture several times for better work and to get the required outcome.

- Another big issue was the lack of PC's for testing the system. Developing it only by testing on localhost was not sufficient sometimes and testing group chat cannot be done with localhost.

8.2. Overcome challenges

- It had to learn and refer documentations and tutorials on socket programming.
- The system has tested using another PC's time to time and understand the bugs and needed additional functionalities to be implemented.

8.3. What learned

- Main concepts about socket programming.
  Using IP address and socket for communication between devices could be learned with this project.

- MSTest testing
  As Visual Studio gives the built-in function of MSTest the tests have been done using it. It is a very easy and efficient time saving testing feature in Visual Studio.

- Event handling
  It's a very cool feature that I learned with this project to use events for invoking functions. In this project, it was so important to use them and it reduces the coupling also.
- Important of the architecture of a system

As I was not familiar with socket programming and event handling before my architecture was not good at first. So with I learned the concepts of these things I had to change the system architecture several times and I realized the importance of a good architecture design for a software system.

9. Current progress and Next step

9.1. Current progress

This project's target was to give a very simple easy to understand system to the users to share things using Wi-Fi and LAN. As some applications are already in Mobile phones for that purpose our first target was to develop it for PCs. With the first build of flyBird application for Windows it has achieved its desired outcomes by far.

By the way the UI designed has some limitations with Windows Form Application. So, it has identified as an area which should be improved.

9.2. Next step

9.2.1. Windows application

In the application that already developed has some areas to be improved. So the next version of the application will come up with improvements of the following areas.

1. UI.

   It is expected to develop the application UI using WPF UI method in the next build and hope to improve the UI look and feel to more user-friendly manner.

2. Architecture

   Some architectural issues are there in the system. So, it is expected to develop the architecture of the application for more reliable and less coupling manner.

3. More user-friendly design

   Creating the server also can automate so that next version will be more similar to modern social chat applications that hides all the network scenarios.

9.2.2. Androd version

This system is expected to develop for android mobiles also in the next step. Experiences gained from the first version of the application for Windows will be useful for the designing the application for mobile platforms. It should communicate between mobiles-mobiles and mobile-PC.

Later it is expected to develop this for Linux also.

10. Conclusion

"flyBird" is a data sharing and communicating application using LAN. Main target is wireless LAN. It tries to minimize the complexity of the network based scenarios and give a very user friendly interface to the users to transfer the data by using their Wi-Fi devices.

In PC's there isn't a built-in application for this purpose like Bluetooth. The problem in Bluetooth is it is very slow compared to Wi-Fi and the application doesn't provide a communicating interface except only file transferring. "flyBird" tries to be the solution for these problems. After it expand for mobile platforms it can do the job more widely.

11. References

[1] Microsoft. "C#" https://msdn.microsoft.com/en-us/library/kx37x362.aspx 2017 [Feb. 08, 2017]

[2] Microsoft. "Any Developer, Any App, Any Platform" Internet: https://www.visualstudio.com/ 2017 [Feb. 08, 2017]

[3] Dennis Magno. "Metro Modern UI - Metro Framework 1.4.0" Internet: https://www.nuget.org/packages/MetroModernUI/ July. 19,2016[Feb. 08, 2017]

[9] GitHub, Inc. "Github" Internet: https://github.com/ 2017 [Feb. 17, 2017]