Palestine Polytechnic University

College of Information Technology and Computer Engineering

Information Technology Department

Bwera Instant Messaging

(Android Chat Client)

Prepared By

Hamza Sulimia **Zaid Sultan**

Supervised By

Mr. Mohammad Jabary

This project was submitted in partial fulfillment of the requirements of the requirements for the degree of B.Sc. in Information Technology

Jan 2013

Dedication

To the candle lighted and guided us through darkness and frustrating moments …

To our parents …

To all our brothers and sister …

To all of our friend …

To our teachers who supported us …

To all these who care about us …

To all of these people we would like to dedicate projects .

Acknowledgement

Praise be to Allah Almighty that successful in completing this project and put it in its current form. The research team thanks all those who contributed in the completion of this work, and for thank Mr. Mohammed Jabary, who did not hesitate to provide aid and assistance and providing valuable tips that we had a good help in this project.

Abstract

Bwera IM is a front end instant messaging client that connects users to multiple Instant Messaging services at the same time; the key advantage of the application is implementing several IM services, In this project we implemented Google Talk and Facebook chat, The system has been applied as application on Android powered tablets and smart phones.

Content

List Of Table

List Of Figure

Chapter One: Introduction

* Overview
* Background
* Problem Statement
* Objectives
* Project Scope
* System development schedule
* Distributing Roles
* Time Line

1.1 Overview

With the increasing growth of using technology in communication and social networks, there is a need to make the communication process easier and faster. From this point, the idea of building an Instant Messaging (IM) chat client for Android-based devices comes to mind.

This chapter addresses various points. First of all, it introduces the background and literature review about the previous related studies and projects. Secondly, the problem statement in this chapter explains the problem that our project should solve. Thirdly, the objectives of the project are introduced. Fourthly, the project scope explains what the project will cover. Finally, the schedule of the system development is explained.

Instant messaging is a term that refers to a form of communication between people over the Internet. It offers a live, text-based messaging capability between client applications. As e-mails came to light, a need for live messaging rather than off-line messaging is raised. Thus, many IM services have been introduced later during the last two decades

The technology of IM is a vital field of IT industry, nowadays, due to the exponential growth of using technology in communication and social networking; therefore, the need to make the process of communication more efficient has grown, especially that Instant Messaging became a popular service among Internet users worldwide.

Nowadays, Instant messaging has become a major field of the internet services, and users of IM are now counted in millions. According to some statistics, windows live messenger has over 300 million users, yahoo messenger has about 300 million users, Google talk, and Facebook users are more than 700 million [1][2][3][4].

1.2 Background

The services of instant messaging are offered from many companies and corporations, since the major email service providers offer an IM based on their email service. Yahoo offers an IM service for their users (yahoo messenger); Microsoft also offer an IM service for its users via windows live messenger, and Google offer an IM service using its client, Google talk, as well. Moreover, Facebook, the most widespread social network offers an IM capability through both the website and an Extensible Messaging Presence Protocol (XMPP) server for stand-alone clients.

Instant messaging clients that run multiple protocols in the same application also exist. Such client applications include pidgin, Kopete, and empathy, all of which are for personal computers. Moreover, some applications for smart-phones such as Nimbuzz and Fring also exist.

Pidgin, Kopete and empathy are open source multi-purpose instant messaging clients that run several instant messaging (IM) services such as yahoo, Hotmail, Facebook, Google talk and some other chat networks all at once. Nimbuzz and Fring, however, are proprietary, closed-source, commercial apps that run on smart-phones, which also run multiple services as well. [5][6]

"The Extensible Messaging and Presence Protocol (XMPP) is an open technology for real-time communication, which powers a wide range of applications including instant messaging, presence, multi-party chat, voice and video calls, collaboration, lightweight middle-ware, content syndication, and generalized routing of XML data. The technology pages provide more information about the various XMPP "building blocks". Several books about Jabber/XMPP technologies are available, as well" [7].

Google and Facebook chat services rely on the Extensible Messaging and Presence Protocol (XMPP) for authentication, presence, and messaging[8]. Code libraries for XMPP are available for different programming languages such as java ,C++ , python, php, c# and others; this enables developers to build a wide variety of XMPP-enabled applications[9].

1.3 Problem Statement

Instant Messaging (IM) clients are application software used to communicate with other clients that rely on the same service provider and the same protocol, like Yahoo Messenger, Windows Live Messenger and Google Talk. Those applications are commonly used by a wide variety of internet users all over the world to contact their families, colleagues and friends. This project aims to create an open source IM client that works natively on Android smartphones that supports multiple protocols and service providers. Opening the source of the application is essential, since it enables the users to use the application without any restrictions. this can also eliminate the need of third parties that are used in proprietary clients. Free and Open Source projects imply the freedom to use the software for any purpose, and the freedom to improve the program. Finally, open source applications give the community opportunity to improve and enhance the throughput to the application and to perform bug fixes.

1.4 Objectives

Our project aims to achieve the following main objectives:

1. Introduce an easy to use, functional, full featured instant messaging (IM) application, so that users can connect to the applied protocols wherever they go using their smartphone and an Internet connection either by Wi-Fi or broadband communication networks.

2. Support multiple protocols and IM service providers in the same app, which enables users to communicate with others on multiple networks and service providers without the need to use multiple apps.

3. Establish a direct connection between the application and the service providers servers without involving any third party.

4. Guarantee Privacy and security of the users data and information during transmission between the application and the service providers..

5. License this app under an open source agreement so that community of developers contribute bug-tracking, fixes, and adding value to the application.

1.5 Project Scope

The project is designed to serve the users of instant messaging services of multiple IM service providers, such as Facebook, Google, Yahoo and Hotmail. Also, the project is supposed to run on mobile devices powered by Android.

1.6 Methodology

The methodology that the team group will follow in developing the application is system development life cycle (SDLC), which states many phases in developing a software. The first step is planning the system, then setting and analysis requirements, then designing the system interfaces, then implementing and testing the system, and finally deploying the application and perform maintenance.

1.7 Distributing Roles

The team group consist of two members who work together in developing the system. The team works together in all project phases, including the analysis, programming, and all of the others tasks to introduce a high quality application.

- The first phases: gathering and analyzing data, and extract information and statistics.

- The second phase: implementation and development of the application.
- The Final phase: document the system.

1.8 Time line

1.8.1 Time Development Study

The following table shows the time needed by the teamwork to accomplish the project

Task Number	Task Name	Weeks needed
1	**Planning and gathering information**	**6**
2	**Determine system requirements**	**6**
3	**describe system requirements**	**2**
4	**Designing**	**6**
5	**Coding**	**4**
7	**Testing and debugging**	**6**
8	**Documentation**	**Documentation on each phase**

Table 1 - Time Development Study

1.8.2 Gant Chart

Task	Time / Week															
	First Semester							Second Semester								
	2	4	6	8	10	12	14	2	4	6	8	10	12	14	16
Planning and gathering information															
Determine system requirements															
describe system requirements															
Designing															
Coding															
Testing															
Documentation															

Table 2 - Gant Chart

Expected Time

Real Time

Chapter Two: System Requirements

* Current system
* Alternatives
* Proposed system
* Functional requirement
* Non-functional requirement
* System development resources
* System operating resources
* Feasibility
* Constraints and Limitations
* Work team constraints
* Suggested solutions

2.1 Current system

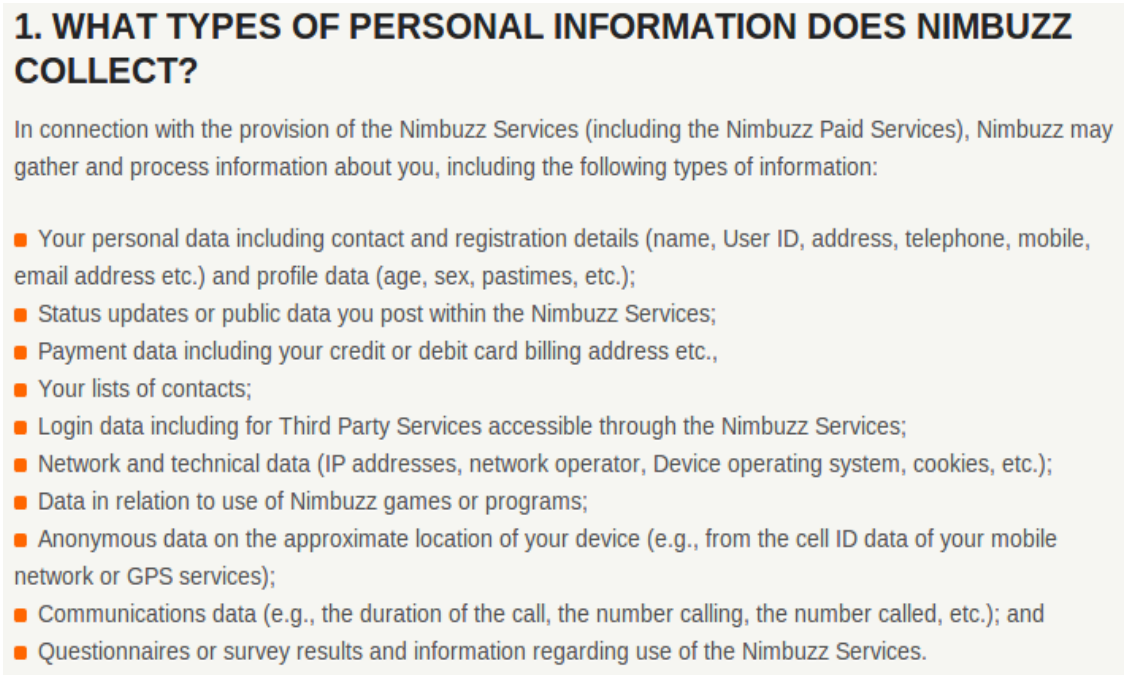
The current system of instant messaging depends on different services providers, and each service provider offers a distinct client for its service, which result-in different clients for various service providers; for example: if a user has two emails from two different service providers, s/he has to install two different clients to achieve the IM process. This makes overhead on the system. Suppose that a user has Google talk, Yahoo, Hotmail and Facebook accounts; in this case, if the user wants to chat with friends using all of those accounts, the user should install a client application for each account and each service provider.

2.2 Alternatives

Because of the overhead and the limitations of the current system, client applications and web based application that can run various IM services in one application at the same time recently appeared.

On one hand, Pidgin is a multi-purpose client chat that can run Yahoo, Hotmail, Google talk, Facebook chat and other IM services within the same program at the same time. Pidgin is an open source software which can run on multiple desktop platforms, such as windows and Linux. This application, however, doesn't run on smart phones and mobile devices.

On the other hand, we have Nimbuzz and Fring, which are commercial, multi-purpose mobile application chat clients that run multiple IM services at the same app. Those two apps run on various smartphone platforms and tablets, including Android and iOS. They are both proprietary application, and they come with a wide variety of restriction on using the software by the users, and terms of use and privacy policies that threaten and even impact both security and privacy of the user [5][6].



1. WHAT TYPES OF PERSONAL INFORMATION DOES NIMBUZZ COLLECT?

In connection with the provision of the Nimbuzz Services (including the Nimbuzz Paid Services), Nimbuzz may gather and process information about you, including the following types of information:

- Your personal data including contact and registration details (name, User ID, address, telephone, mobile, email address etc.) and profile data (age, sex, pastimes, etc.);
- Status updates or public data you post within the Nimbuzz Services;
- Payment data including your credit or debit card billing address etc.,
- Your lists of contacts;
- Login data including for Third Party Services accessible through the Nimbuzz Services;
- Network and technical data (IP addresses, network operator, Device operating system, cookies, etc.);
- Data in relation to use of Nimbuzz games or programs;
- Anonymous data on the approximate location of your device (e.g., from the cell ID data of your mobile network or GPS services);
- Communications data (e.g., the duration of the call, the number calling, the number called, etc.); and
- Questionnaires or survey results and information regarding use of the Nimbuzz Services.

Figure 1- Nimbuzz privacy statement

2.3 Proposed system

The proposed project is planned to work natively on Android-powered smart-phones and devices. It aims to cover several IM providers and services, including Google Talk, Jabber and Facebook. The final project will be published under a free and open-source application that authenticates and communicate directly with each service provider, without involving any third party in any phase of the process.

Being free and open-source, the application will be able to benefit from the open-source community, which will be able and expected to provide testing, debugging, new features, and further development after the project is done. Thus, the project will be able to continue growing relying on the community, rather than only the students involved in the implementation and the initial development of the project.

By direct authentication and communication with the original service providers, without involving any third party, the application is supposed to give better security and privacy, and even response speed to the users.

2.4 Functional Requirement

The project has a set of main required functions and services that the users take advantage of when using the final application.

The potential tasks to be done in this project:

1- Offering the login process into several IM services, so that the user can log in to different accounts from different providers at the same time.

2- Show the contact list for each IM service provider.

3- Sending and receiving messages from and to contacts.

4- The user can add new contact for each IM service.

5- The user has the choice to set the status of the service (Available, Busy, Away or Invisible).

6- Validation: there must be a client side validation to validate emails.

2.5 Non - Functional Requirement

1 - Latency: Reduce the speed of logging on the service due to the direct connection between the user and the IM service provider servers.

2. Expandability: The system must be able to be expanded to include additional protocols and services such as VOIP and camera.

3. Stability and reliability: the application should be bug-free and reliable.

4. Android Compatibility: The application should run on android operating system natively.

5. The application has to be compatible with various protocols, such as XMPP and Yahoo chat protocols.

6. The Application has to be developed on Object-Oriented based .

2.6 System Development Resources

While the system is being developed, there must be a set of pre-required resources to finalize the system development; these resources are categorized as:

- Physical Resources: consist of the required hardware and internet connection.
- Software Resources: all the software required to develop the system.
- Human Resources: consist the team group, and the community for further development.

2.6.1 Development physical resources

1. PC with the following specifications
2. Smartphone with the following Specification

Hardware	specification	number
PC	**Any PC with an enough RAM to run Android Emulator**	**1**
Smartphone	**Any android powered - device**	**2**

Table 3 - Development Physical Resources

2.6.2 Development software resources

1. Ubuntu 12.04 Operating System
2. Eclipse IDE
3. Android Software development kit.

2.6.3 Development human resources

The developers of the system consist of the project team work whom are two undergraduate students. Later on, the community will be able to participate in the development process.

2.7 System operating resources

1. Physical resources: Android-powered device(s).
2. Software resources: Android operating system, The proposed application, and Internet Connection.
3. Human resources: the users.

2.8 Feasibility

2.8.1 System Development Cost

2.8.1.1 Physical development Cost

This table shows the physical development system cost

Cost	Number	Specification	Hardware
200 [10]	1	Any PC with enough RAM to run Android Emulator	Computer
150$[11]	2	Any android powered device	Mobile
500$			Total

Table 4 - system development physical cost

2.8.1.2 Software development Cost

Cost	Number	Software
FREE	2	Ubuntu
FREE	2	Eclipse
FREE		Total

Table 5 - Software development Cost

2.8.1.3 Human Resources development Cost

Total Cost	Cost/Month	Number	Human Resource
600 $	300 $	2	Teamwork
600 $ * 7 months = 4200 $			

Table 6 - Human Resources development Cost

2.8.2 System Operating Costs

Internet connection: varies depending on the Internet Service Provider (ISP).

2.8.3 System development and operating costs

The following table shows the total development and operational costs

Cost	Operation Resources	Cost	Development Cost
ISP	Physical	500$	Physical
Free	Software	Free	Software
Free	Human	4200$	Human
4700$ + ISP costs			Total

Table 7 - total development and operational costs

2.9 Risks

1. Building the system before the deadline.
2. Extensibility: The system must be able to develop and grow.
3. The availability of Android-powered devices in the hands of the developers.
4. Internet Access availability and costs.

2.10 Work team Constraints

1. The delay of submitting on time.
2. The lack of Android programming skills.
3. There is no smart-phone with the team.

2.11 Suggested Solutions

1. Learn Android application programming to avoid the lack of skills limitation.
2. Good planning to development life cycle.
3. Distribute roles and tasks effectively on the team work.

Chapter Three: System Specification

* Introduction
* System Description
* System Requirements Description
* Use Case
* Sequence Diagram

3.1 Introduction

This chapter will contain general description of the system's main functionalities, in addition to explaining the requirement specifications in details. Moreover, this chapter will perform a general view of the user-system interaction.

3.2 System Description

The final application that will be provided at the end of the work will be able to run on Android Operating system, and to enable and facilitate the authentication and communication to several instant messaging service providers, including Google Talk, Facebook chat, Jabber and Yahoo. The application will be based completely on free and open-source technologies and applications, in order to maintain a clear future of the project, and to avoid any potential stop of the development of any software that is needed for the application to run, and to keep the development in progress, and even to sustain security and enhancements. The application will be free and open-source, so that it benefits from the community and doesn't stop when one of the core developers stop participating in the development.

3.3 System Requirements Description

1. Log In

Function	**Log In**
Description	**authenticating users**
Input	**username + password**
Sources	**android application**
Output	**logging into account**
Target	**authentication and authorization**
Requirement	**Internet Access**
Procedures	**insert user name and password then click log in button**

Table 8 - Log In Description

2. Buddy List

Function	**Buddy list**
Description	**Fetching the buddy list from the service provider's servers**
Input	**authentication**
Sources	**android application**
Output	**buddy list**
Target	**show the buddy list status**
Requirement	**authentication from service provider servers**
Procedures	**Fetching the buddy list via the server**

Table 9 - Buddy List Discription

3. Add new buddies

Function	**Add new buddies**
Description	**add new contacts to each service**
Input	**email address / user name**
Sources	**android application**
Output	**new contacts**
Target	**contact with new friends , colleagues**
Requirement	**logged in**
Procedures	**follow the instructions of adding new contact**

Table 10 - Add new buddies Discription

4. profile status

Function	**profile status**
Description	**set the status of the profile (available , busy , or away)**
Input	**The new status**
Sources	**android application**
Output	**status to be appeared to other buddies**
Target	**notify friends about my status**
Requirement	**To be logged-in**
Procedures	**pick a drop down list option**

Table 11 - profile status Discription

5. identify service provider

function	identify service provider
description	distinguish service providers during processes
input	adding contact
sources	android application
output	done processes correctly
target	effectiveness and correctness
requirement	Application Programmable Interface (API)
procedure	code work by the developers; no user interaction needed

Table 12 - identify service provider Discription

6. validation

function	validation
description	validate email addresses
input	Email address
sources	android application
output	latency
target	validate email before sending it to service servers
requirement	API, code and regular expression
procedure	code work

Table 13 - validation Discription

3.4 Use Case

The use case describes the user view of the application and what can the user do. The user can add new accounts, in addition to log into the added accounts, and then chat with contacts, and also the user can set a status and finally add new contacts.

The system consist of main operations, represented as follows :

- add account : the user at first has to add a new account service provider .

- login : after choosing an account, the user has to insert the user name and the password of the added account. After that, the user clicks the login button to send the username and the encrypted password to the service provider.

- Authentication: After sending the user-name and the password to the service provider's servers, the provider has to perform the authentication of that user's account.

- Contacts importing : if the server authenticate that account, the client imports the contacts of that account.

- If he wants to add more accounts :-

  - ✓ Add more accounts : if the user want to add more accounts of the same or different provider, s/he must redo to the first step.

  - ✓ chatting: the user starts making conversations with contacts of each provider .

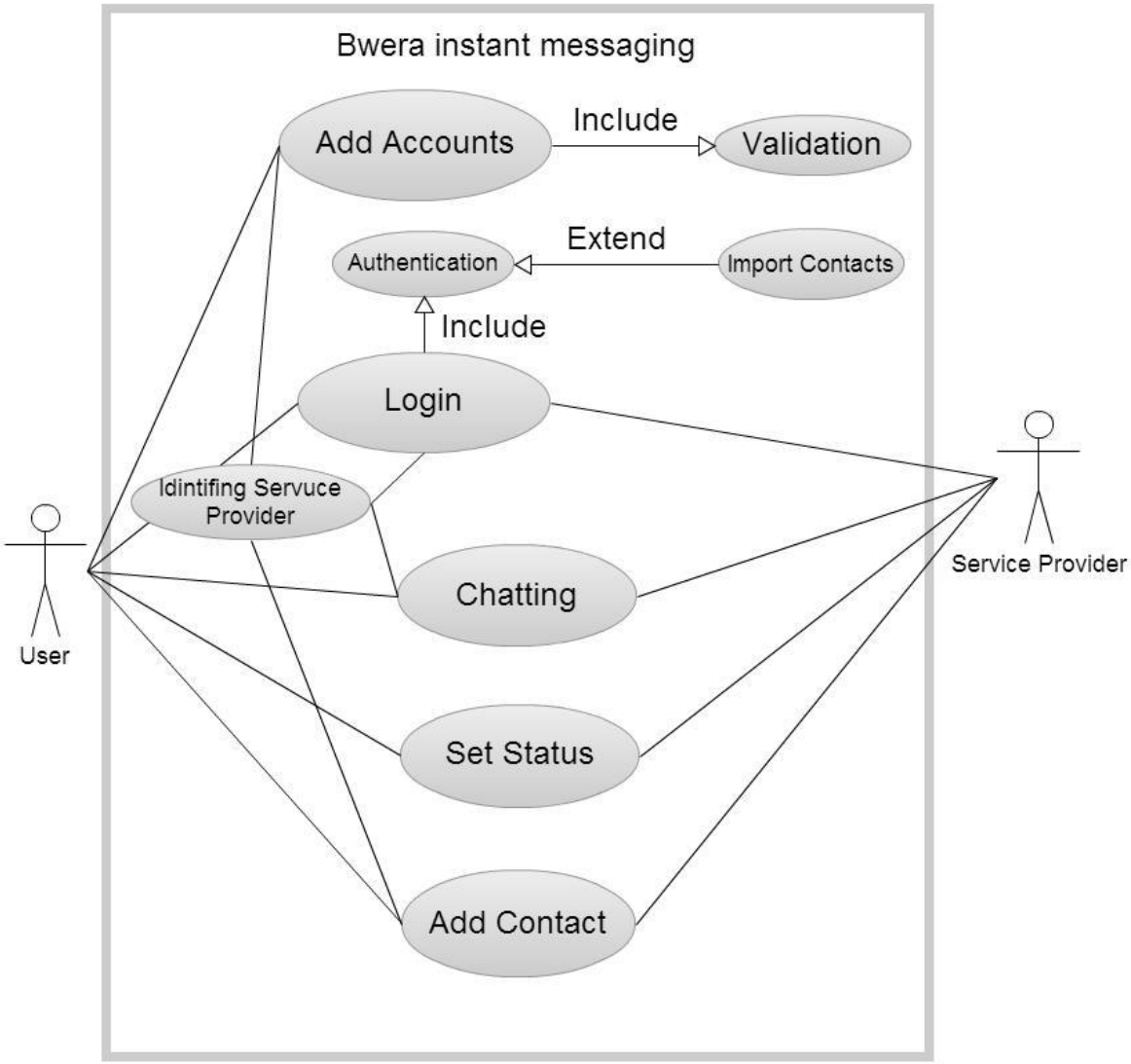


Figure 2 - System use case

3.5 Sequence Diagram

The first sequence diagram represents checking the connection with the internet.

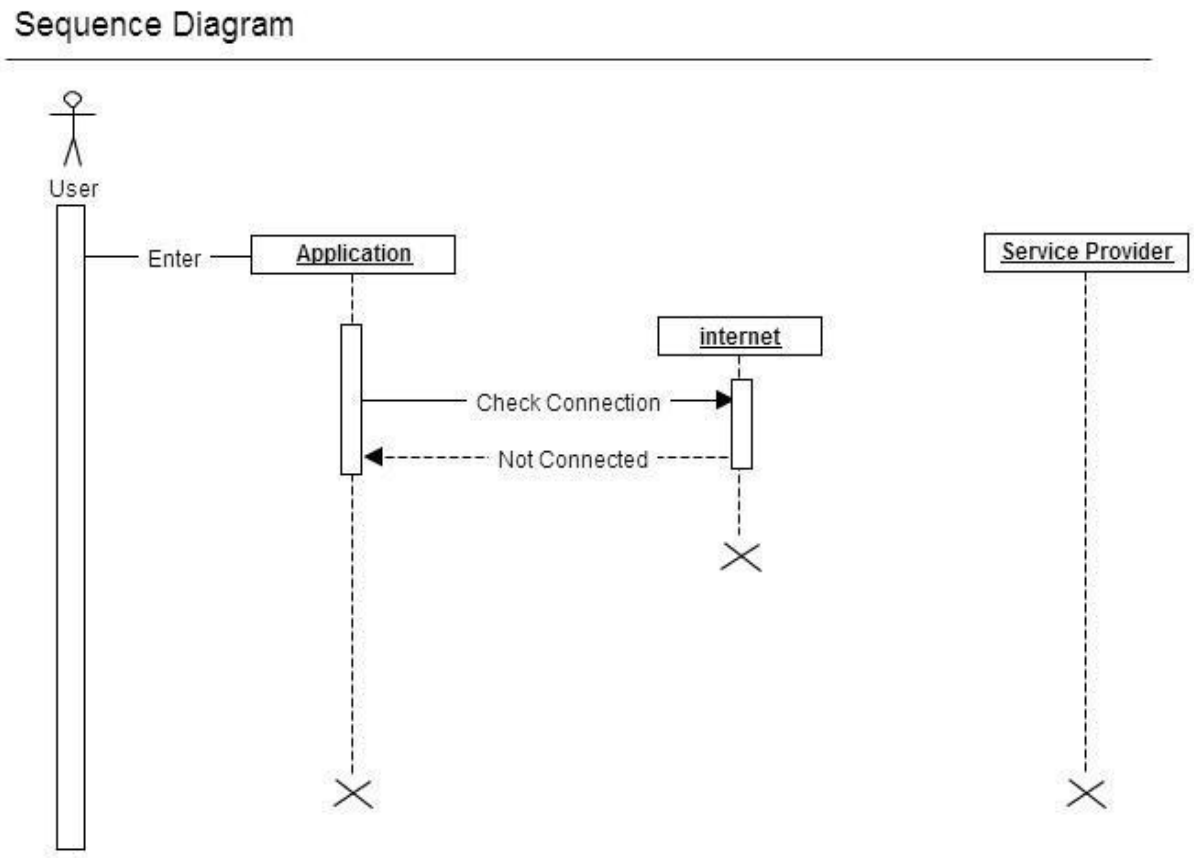


Figure 3 - Sequence Diagram(check internet)

The second sequence diagram represents the scenario of wrong username or password authentication process where the service provider reply a message of no authentication.

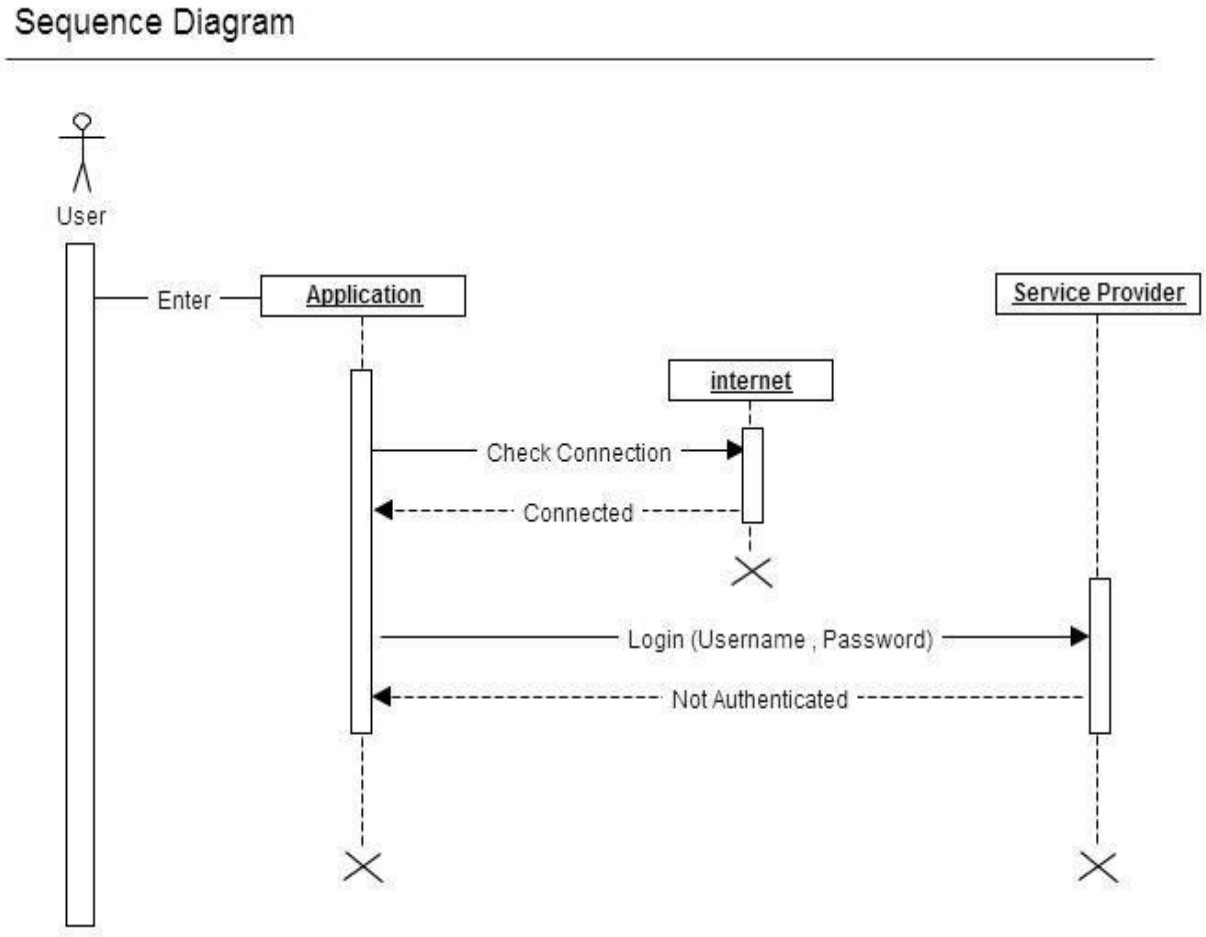


Figure 4 - Sequence Diagram(check authentication)

The last sequence diagram shows the events of sending and receiving messages after a success of authentication process.

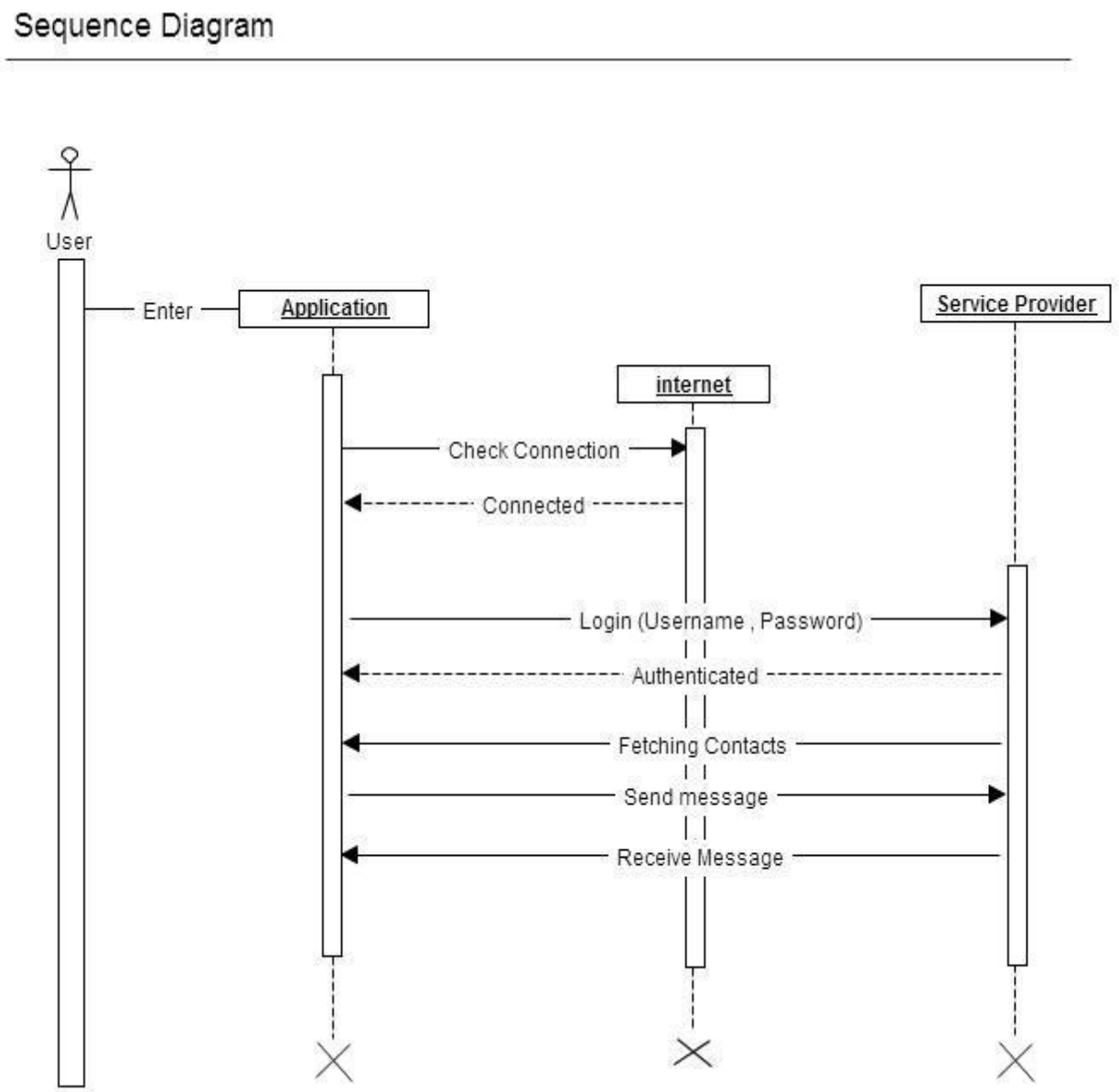


Figure 5 - Sequence Diagram

Chapter Four: System Design

* Introduction
* System Diagrams
* System Interfaces
* System Flowchart
* Chapter Summary

4.1 Introduction

The design stage is one of the important stages of any system. Which give a general overview about the total components of the project by describing it via diagrams. In this chapter we will discuss the logical and physical design of the project. including the initial screenshots of the system.

4.2 System Diagrams

This section includes all the diagrams that relevant to the system, Block diagram, sequence diagram and UML are illustrated.

4.2.1 Block Diagram

The system interact with the user and the service provider by the application programmable interface . The application act as a link between the user and the service provider.

Figure 6 - Block Diagram

Service Provider : is the back end of the chat service where the server of the provider. Each service provider provides an API to developers, where developers can build

applications based on the service. Google and Facebook rely on an open standard protocol for messaging which is Extensible Messaging and Presence Protocol (XMPP), This protocol has several programmable libraries in several programming languages (Java, C++, php, C# and others) to build a standalone frontend clients for IM service.[12]

Application : is a front end of the chat service which is connected directly with the back end provider via the internet, the application is connected via the service provider by an application programmable interface (API) supported by the service provider. Smack is an XMPP client API programmed in Java programming language. Asmack is a patch of the Smack XMPP client library that has some fixes for Android., aSmack is the library to used to connect with the IM service of Facebook chat and Google talk. [13][14]

To satisfy the functional requirements of the system we follow several steps :

1- Authentication step - This process include the connection between the user and the service provider server. asmack api used to achieve authentication by creating a connection for Google talk and Facebook chat services. this process has been done by creating a login class, this layout of this class consist of two string edittexts areas to insert username and password and consist a login button to check the connection and consist a label. this activity has three cases the first case is to check if there is no internet connection then the label will show no internet access msg, the second case is to check the username and the password then the label will show that the username and/or the password is incorrect or if they are correct the authentication process will done and the second step come.

2- Contacts fetching process - in the case of authentication will move the user to contacts class which include a layout of a listview to insert contacts into it. the contacts of the user will be fetched to the listview . clicking on a specific contact in this list view will move the user to the next messaging step.

3- Chat process - here the user can instant messaging with the intended contact, exchange messages instantly. in this class layout there are an edit text area an addition to a send button . the edit text area used to write messages and the send button used to send the message to the

intended user. these sends messages in a addition to the received messages are inserted into a listview to make a conversation.

The previous resolutions to satisfy the functional requirements are associated with the initial interfaces of the application which described later at this chapter.

asmack guarantee a secure connection so that it uses TLS encryption to encrypt and secure users data. , the whole connection including authentication and messaging is secure.

4.3 System Interfaces

In this section we will discuss an initial interfaces of the proposed system , interfaces includes :

1. Login interface : This is the main activity where the user can choose the service provider and then use his username and password to log into chat service or to add new account.

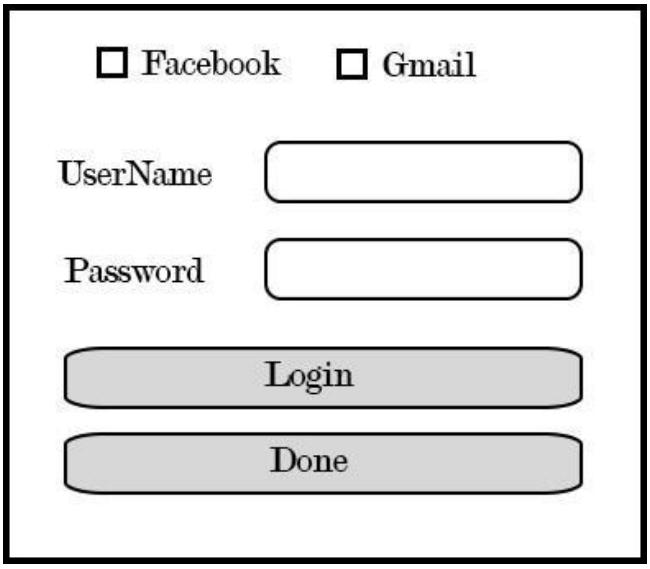


Figure 7 - Login Interface

2. Contacts Interface : This is the second activity where the login step done successfully, this activity shows the contacts of the user. a search tool included in this activity to reach contacts effectively.

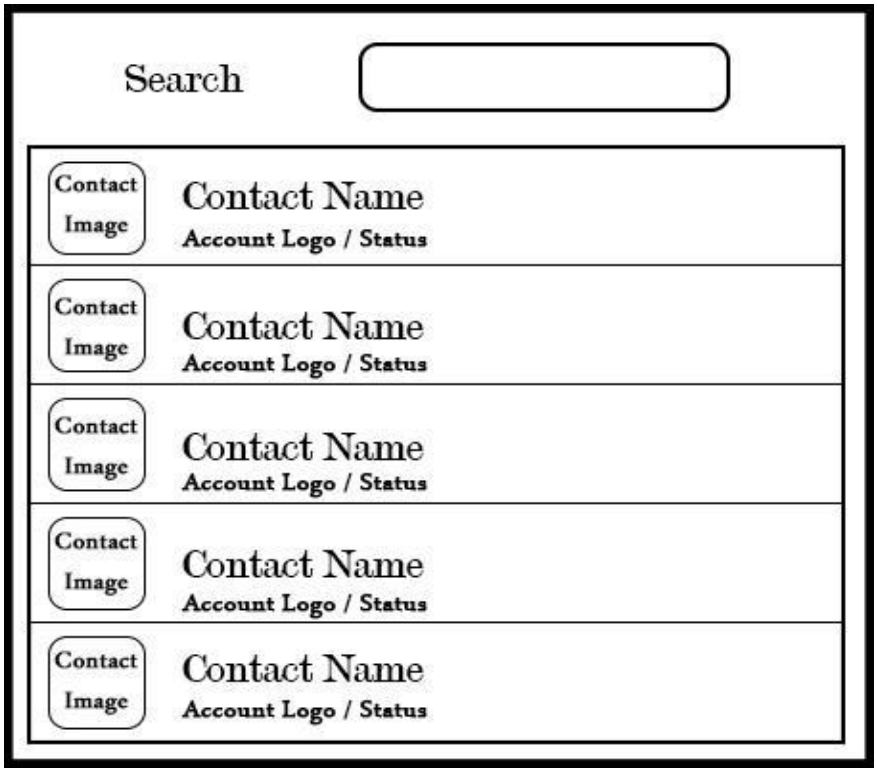


Figure 8 - Contacts Interface

3. Messaging interface : By clicking on a contact in the second activity , This activity come up to start a conversation with the intended contact.

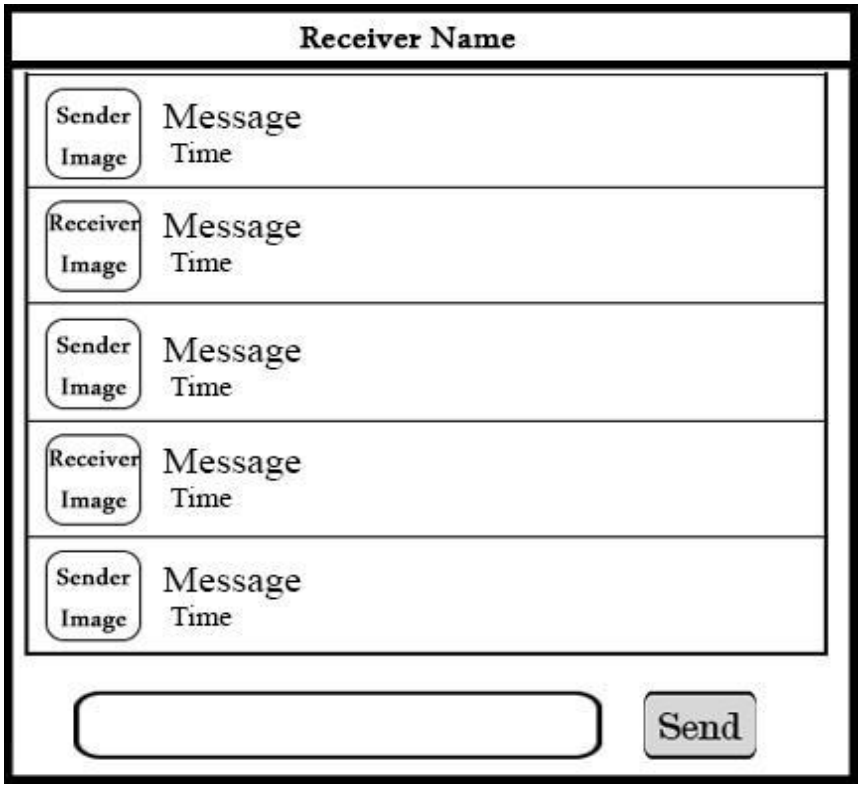


Figure 9 - Messaging Interface

4. Set Mood interface : these are the states of the user profile

Figure 10 - Set Mood Interface

5. Contact option interface : this interface should appear when the user click long on a contact. Chat and delete options are available.

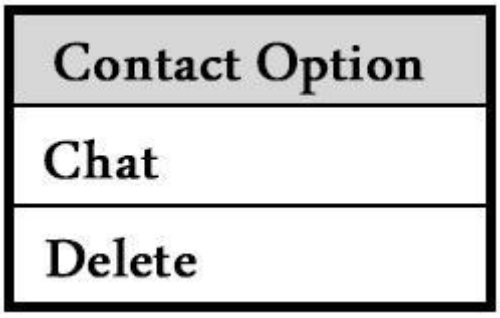


Figure 11 - Contact option interface

4.4 System Pseudo Code

```
Begin
Click on App icon on Mobile
choose service provider
enter username and password
Press “Login” Button
{
        Create Connection to service provider
        if (Connection is Connect)
        {
                Login to service provider by username and password
                if (Logged In)
                        Show Message “Done”
                else
                        Show Message “Incorrect Username or Password”
                end if
        }
        else
                Show Message “No internet Access”
```

```
        end if
}
if (need to add another service provider)
        doing previous steps
else
{
        Press “Done” Button
        Go to second Interface
        Display contacts
        choose one contact to send message
        Go to second Interface
        start chat
}
end if
```

4.5 Chapter Summary

In This Chapter we have discussed the initial user interface of the proposed system, In addition describing the system diagrams such as sequence diagram, block diagram and class diagram and finally a flow chart of the system has been illustrated

Chapter Five: System implementation, testing and Maintenance

* Introduction
* Implementation requirements
* Operations requirement
* Testing
* Maintenance description and guidelines

5.1 : Introduction

this chapter is one of the most important chapters In this chapter we will discuss the requirements of the project implementation explaining the development requirements on the other hand we will discuss the operational requirements in addition to covering testing procedures and finally describing the maintenance process and some guidelines.

5.2 Implementation requirements

5.2.1 Software needed to development :

1- Ubuntu 12.4 : is an open source linux based operating system , it's free of charge, ubuntu characterized as secure and stable. ubuntu is a good environment for programming and developing software.

2- Eclipse : is an integrated development environment (IDE) that support development in several programming languages such as java, c, c++, perl, python, and ruby. Eclipse is an independent machine IDE that can be run on Linux, Windows or Mac platform. Eclipse is free and open source and built by java.

3- Android Application Development Tools (ADT) :

- Eclipse + ADT plugin : is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications. It offers you access to many features that help you develop Android applications quickly. ADT provides GUI access to many of the command line SDK tools as well as a UI design tool for rapid prototyping, designing, and building of your application's user interface.

- Android SDK Tools : The Android SDK is composed of modular packages that you can download separately using the Android SDK Manager. For example, when the SDK Tools are updated or a new version of the Android platform is released, you can use the SDK Manager to quickly download them to your environment.

- Android Platform-tools

- Android platform : To develop an Android app, you must install at least one Android platform from the SDK Manager against which you can compile your app. Often, any given version of the Android will be revised with bug fixes or other changes, as denoted by the revision number. Below, you'll find the release notes for each version of the platform and the subsequent revisions to the platform version.

- Android system image for the emulator.[15]

4. Smack and asmack API :

Smack is an Open Source XMPP client side library for instant messaging and presence. A pure Java library, it can be embedded into your applications to create anything from a full XMPP client to simple XMPP integrations such as sending notification messages and presence-enabling devices. aSmack is a ported version of smack to support Android application development.
we are using asmack client library to build and support a facebook chat and Google talk services in our project.[16]

5.2.2 Selected code

here are some samples of the source code

1. The main classes of the asmack library we have used

```
import org.jivesoftware.smack.Roster;
import org.jivesoftware.smack.RosterEntry;
import org.jivesoftware.smack.XMPPConnection;
import org.jivesoftware.smack.XMPPException;
import org.jivesoftware.smack.filter.MessageTypeFilter;
import org.jivesoftware.smack.filter.PacketFilter;
```

```
import org.jivesoftware.smack.packet.Message;
import org.jivesoftware.smack.packet.Presence;
import org.jivesoftware.smack.provider.ProviderManager;
import org.jivesoftware.smackx.packet.VCard;
import org.jivesoftware.smackx.provider.VCardProvider;
import org.jivesoftware.smack.Chat;
import org.jivesoftware.smack.ChatManager;
import org.jivesoftware.smack.MessageListener;
import org.jivesoftware.smack.PacketListener;
import org.jivesoftware.smack.XMPPConnection;
import org.jivesoftware.smack.XMPPException;
import org.jivesoftware.smack.filter.MessageTypeFilter;
import org.jivesoftware.smack.filter.PacketFilter;
import org.jivesoftware.smack.packet.Message;
import org.jivesoftware.smack.packet.Packet;
import org.jivesoftware.smack.provider.ProviderManager;
import org.jivesoftware.smackx.packet.VCard;
import org.jivesoftware.smackx.provider.VCardProvider;
```

2.log in

This code used to make connection to Service Provider

```
ConnectionConfiguration config = new ConnectionConfiguration(host, 5222);
Connection connection = new XMPPConnection(config);
connection.connect();
connection.login(username, password);
```

3. Send Message

This code used to make connection to Send Message

```
ChatManager chatmanager = connection.getChatManager();
Chat newChat = chatmanager.createChat("jsmith@jivesoftware.com",
                                                    newMessageListener(){
        public void processMessage(Chat chat, Message message) {
                System.out.println("Received message: " + message);
                }
        });
        try {
                newChat.sendMessage("Howdy!");
                }
        catch (XMPPException e) {
                System.out.println("Error Delivering block");
        }
```

4. Receive Message

This code used to make connection to Receive Message

```
ChatManager chatmanager = connection.getChatManager().addChatListener(
new ChatManagerListener() {
        @Override
        public void chatCreated(Chat chat, boolean createdLocally){
                if(!createdLocally)
                        chat.addMessageListener(new MyNewMessageListener());
                }
  });
```

5.2.4 Screenshots

Open Eclipse IDE

Figure 12 - Eclipse IDE screenshots

Creating New Project

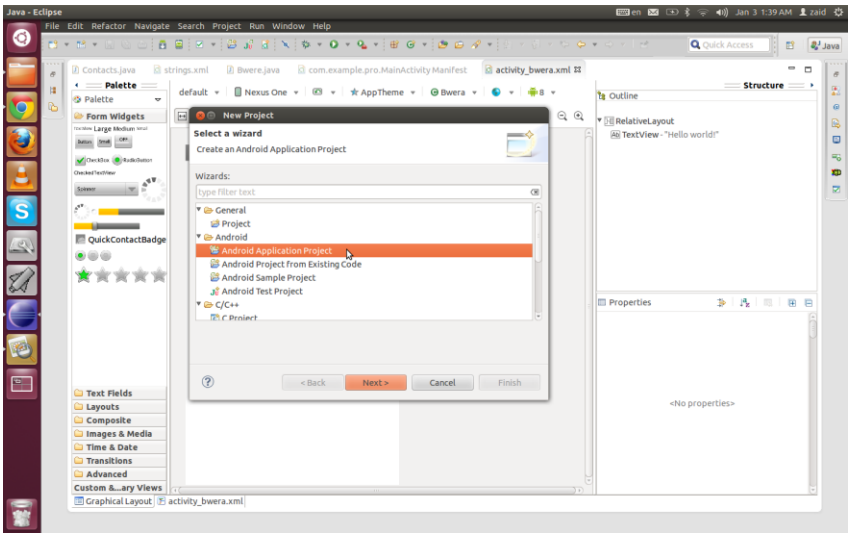


Figure 13 - Creating New Project screenshots

Naming Application, Project and Package

Selecting Android API and Version

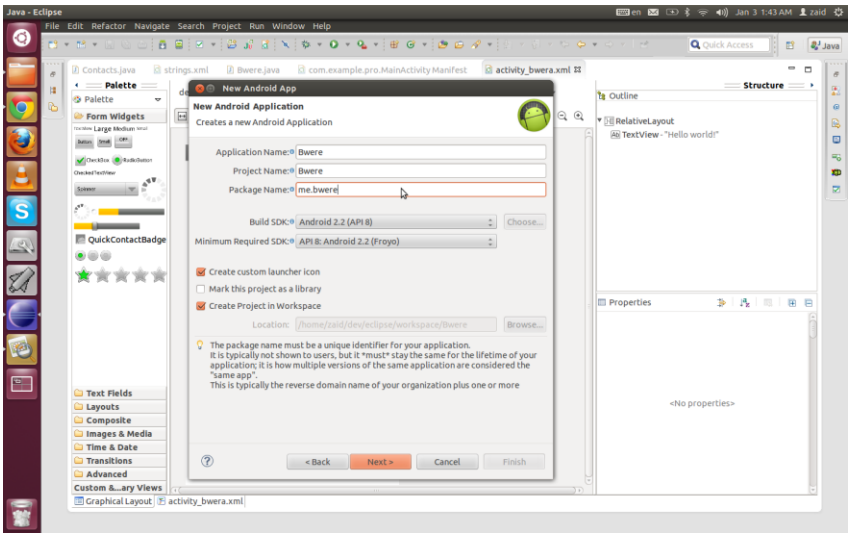


Figure 14 - Application Properties screenshots

Creating new Android Activity

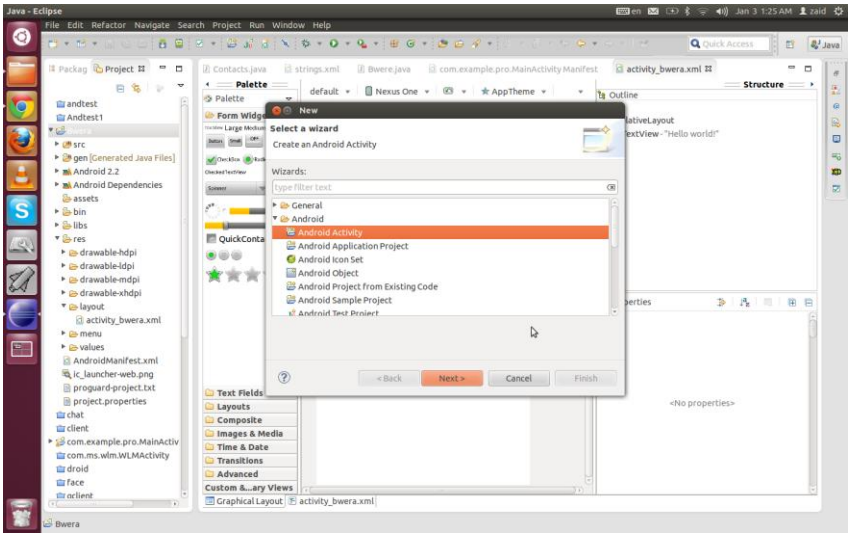


Figure 15 - Creating new Android Activity screenshots

Creating new java class

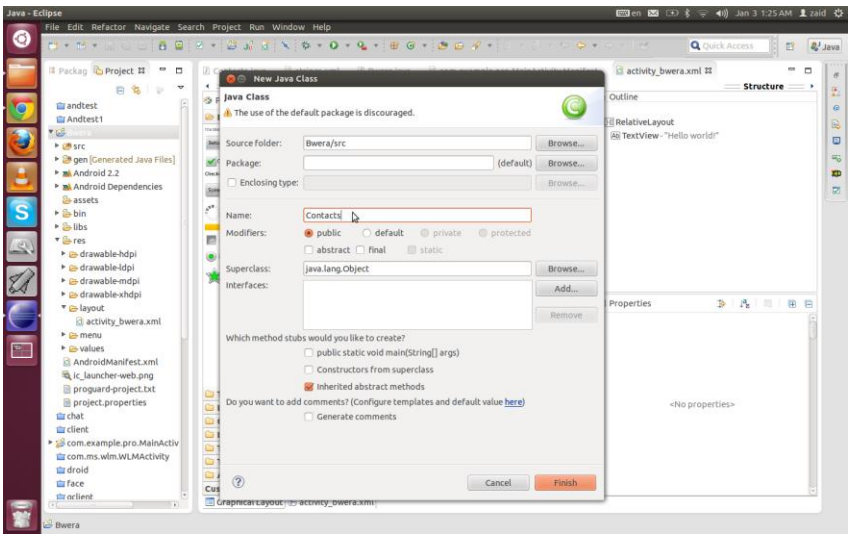


Figure 16 - Creating new java class screenshots

Importing External Jar's (asmack library)

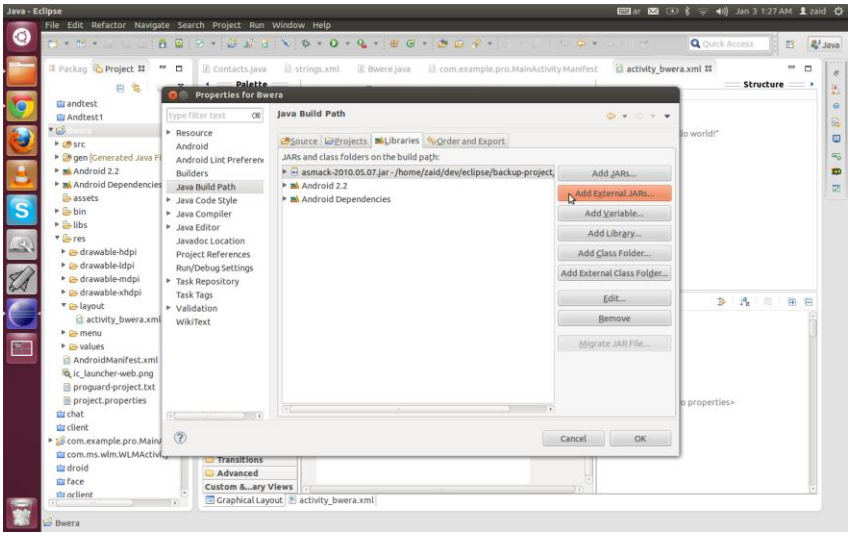


Figure 17 - Importing External Jar's screenshots

Running Android Emulator

Figure 18 - Running Android Emulator screenshots

Run the application (Main Activity)

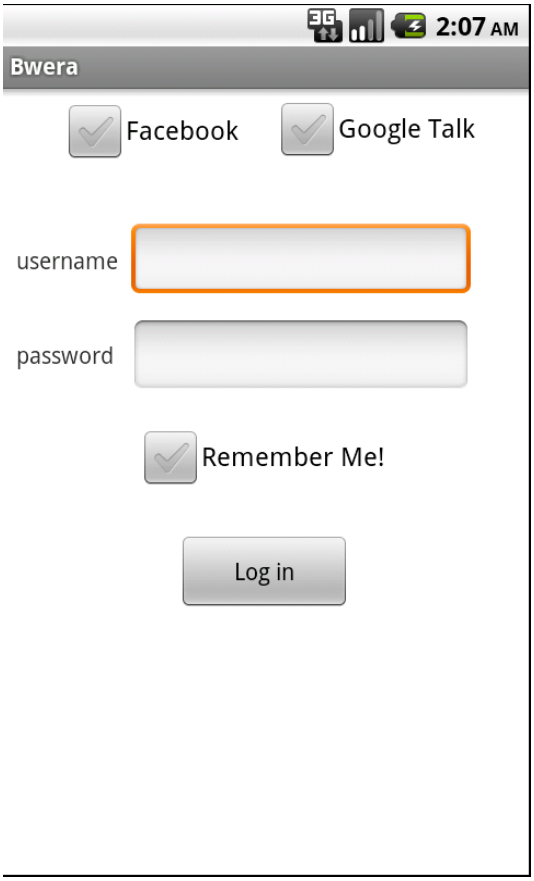


Figure 19 - Main Activity screenshots

Contact list

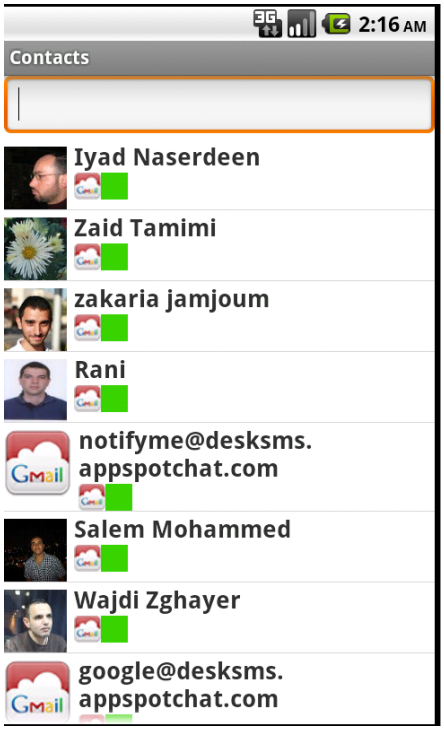


Figure 20 - Contact list screenshots

Chatting Activity

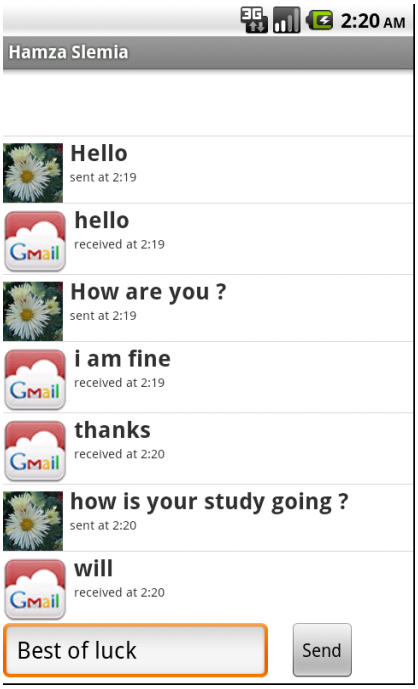


Figure 21 - Chatting Activity screenshots

5.3 Operations requirement

Android platform : Android is required to run the application.

Internet Access : Internet connection is required to connect IM services.

5.4 Testing

System testing is one of the most important phase in the system, In this phase we ensure of achieving the functional and non-functional system requirements and to ensure that the system is reliable, System testing steps as follow :

1. Alpha Testing : This type of testing depends on the developer notes and described as an elastic test.
2. Beta Testing : This type of testing depends on the real users of the system excluding the system developers. this test begin after finishing programming and done through system operating.

5.4.1 Test System Units

Where we test each unit independently of another unites and operations to ensure that it works right as expected. in addition to test the system functional requirements for each part of the system. figure 5.5 show some tested operations by the developers.

Login to Google Talk
Goal : log into Google Talk account Results : Done Successfully
Login to Facebook Chat
Goal : log into Facebook Chat account Results : Done Successfully
Sending Messages by Google Talk
Goal : sending messages from Google Talk user to contacts. Result : Done Successfully
Sending Messages by Facebook Chat
Goal : sending messages from Facebook Chat user to contacts. Result : Done Successfully
Receiving Messages by Google Talk
Goal : Recieving messages from Google Talk contact to the user. Results : Done Successfully

Receiving Messages by Facebook Chat
Goal : Recieving messages from Facebook Chat contact to the user. Results : Done Successfully
Adding Contact in Google Talk
Goal : Adding friends to chat. Results : Done Successfully
Deleting Contact in Google Talk
Goal : Deleting contact from user's list. Results : Done Successfully
Set status in Google Talk
Goal : Write custom status in user status. Restult : Done Successfully
Change Mood in Google talk
Goal : Change mood to busy, away or invisible Results : Done Successfully

5.4.2 Test System Integration

After testing each unit independently, we test these units with each other as one unit to ensure of the integration of the system as expected by sending and reciving messages at the same time.

5.4.3 System Testing

We have testing the system by installing it on various kind of devices and we have diagnose some problems regarding to this test.

5.4.4 System Acceptance Testing

We have tested the system by distribute it on random users so they give a feedback about the acceptance of the system.

5.5 Maintenance and Guidelines

5.5.1 : Deployment

In this section, The application will be deployed to the Android market (Google Play) where as an apk file, the app can be install by one download click from the market. deploying the app on the google play includes uploading the app installable file inaddition to two screenshots of the app and a brief description about the app.

5.5.2 Upgrade

upgrade step done by uploading a new version of the app to the google play where the android platform will notify the user about new versions of the app.

5.5.3 Application maintenance

Maintenance and bug tracking is a feedback from the users of the app by commenting on the app page in the google play, feedback and rates will be consider in new versions.

Chapter Six: Conclusion

* Conclusions
* Recommendations and further work
* Overall Evaluation

6.1 Conclusion

1. Log into Google Talk and Facebook Chat at the same time.
2. Direct authentication and communication with official servers without third parties.
3. Completely functional and easy to use graphical interface.

6.2 Recommendations and further work

1. Expand the application by adding more instant messaging services from different providers.
2. Make the app cross platform by building it on other mobile platforms such as iOS, Meego, and Symbian.
3. Make the app more social by implementing more social networking features such as micro blogging.
4. Add more languages rather than English to the app (multilingual).
5. Add the support of voice and video calling

6.3 Overall Evaluation

We have finished the first version of the application by implementing two instant messaging services, Facebook chat and Google Talk among several instant messaging services. This version supports only english language.

References

[1] SARAH PEREZ.(2011).TechCruch,retrieved from<http://techcrunch.com/2011/12/29/nearly-40-of-facebook-use-is-from-mobile-apps/>, retrieved on April 5 2012.

[2] UNITED STATES SECURITIES AND EXCHANGE COMMISSION, Feb 1 2012 retrieved from, <http://www.sec.gov/Archives/edgar/data/1326801/000119312512034517/d287954ds1.htm>, retrieved on April 5 2012.

[3] Inside Windows Live Messenger Archive, June 15 2009, retrieved from ,<http://messengerarchiveblog.wordpress.com/2009/06/15/share-your-favorite-personal-windo ws-live-messenger-story-with-the-world/>, retrieved on April 5 2012.

[4] Pingdom, retrieved from ,<http://royal.pingdom.com/2010/04/23/amazing-facts-and-figures-about-instant-messaging-infographic/>, retrieved on April 5 2012.

[5] Nimbuzz, retrieved from .<http://www.nimbuzz.com/en/legal/privacy-statement>, retrieved on 5 April 2012.

[6] Fring, retrieved from ,<http://info.fring.com/terms/>, retrieved on April 5 2012.

[7] XMPP, retrieved from ,<http://xmpp.org/about-xmpp/>, retrieved on April 5 2012.

[8] Johansson, Leif. (2005). "XMPPasMOM". Greater NOrdicMIddleware Symposium (GNOMIS). April, 2005. Oslo: University of Stockholm.

[9] XMPP,retrieved from ,<http://xmpp.org/xmpp-software/libraries/>, retrieved on April 5 2012.

[10] Homeplaza, retrieved from ,<www.homeplaza.ps>, retrieved on April 5 2012.

[11] Amazon, retrieved from ,<http://www.amazon.com>, retrieved on April 5 2012.

[12] http://xmpp.org/

[13] http://www.igniterealtime.org/projects/smack/

[14] https://github.com/Flowdalic/asmack

[15] http://developer.android.com

[16] http://www.igniterealtime.org/builds/smack/docs/latest/documentation/