

**School of Information Technology & Business**

**Bachelor of Information Technology**

**SD6501 – Mobile App Development**

**Final Project**

Abstract

Based on the current social actual business needs and conditions, combined with the structure, working characteristics, working methods and basic business needs of modern instant messaging, FindOne application analyzes the basic functions and performance requirements of instant messaging software system, combined with the successful experience of the current instant messaging software system. The development of a universal social friends instant messaging software. Due to the relatively limited installation of the required client, in order to facilitate safe management, this design adopts the Single model. The project adopts Android client and OpenFire integrated server side as the main design, where the data storage is realized by SQLite database and server-side embedded database, and the storage related to the information of friends realized by the software is stored by SQLite database, and the function of real-time chat communication is mainly managed by server-side embedded database information.Finally, in order to verify the operation of the software, the function and performance of FindOne software program are tested under the normal operation of the server and several basic function test samples are tested. The design software can successfully complete the basic function of the design. Through the test module, the functional integrity of the developed friend-making software is effectively guaranteed.

**Directory**

**[Abstract 2](#_Toc20734)**

**[1. Introduction 4](#_Toc12286)**

**[2.Detailed description of the components and its relevance to the requirements specified 6](#_Toc22883)**

[2.1Activity 6](#_Toc10037)

[2.2 Service 6](#_Toc18480)

[2.3 content provider 6](#_Toc19609)

[2.4 Broadcast Receivers 6](#_Toc13632)

[2.5 ListView 6](#_Toc21974)

[2.6 SharedPreferences 7](#_Toc28642)

[2.7 Handler 7](#_Toc19467)

[2.8 Adapter 7](#_Toc8088)

[2.9 asmack 7](#_Toc30727)

**[3.Conceptual Framework 8](#_Toc5099)**

[3.1 XMPP Protocol 8](#_Toc13657)

[3.2 Socket 8](#_Toc941)

[3.3 SQLite 8](#_Toc2826)

[3.4 Message 9](#_Toc16214)

[3.5 AsyncTask 9](#_Toc7928)

**[4.Advanced features and concepts added to this version 10](#_Toc14190)**

**[5.Discussion of constraints encountered, and strategies applied during the development 11](#_Toc295)**

[5.1 Null pointer exception 11](#_Toc32542)

[5.2 network connection exception 11](#_Toc30342)

[5.3 SQLException 11](#_Toc29782)

[5.4 IllegalAccessException 11](#_Toc26364)

[5.5 RejectExcuteException 11](#_Toc16950)

[5.6 IllegalStateException 11](#_Toc28856)

**[6.Debugging and Testing 13](#_Toc22418)**

[6.1 Login 13](#_Toc29461)

[6.2 Start Page 13](#_Toc31436)

[6.3 FriendsList 14](#_Toc13716)

[6.4 Home setting 15](#_Toc11618)

[6.5 Session List 15](#_Toc13244)

[6.6 FriendsChatting 16](#_Toc25807)

[6.7 AddFriends 17](#_Toc7392)

[6.8 About page 17](#_Toc3009)

[6.9 Test 18](#_Toc20242)

[6.9.1 LoginTest 18](#_Toc5551)

[6.9.2 RememberPasswordTest 19](#_Toc3802)

[6.9.3 AddFriends Test 20](#_Toc72)

[6.9.4 Home Test 21](#_Toc2399)

[6.9.5 ChattinigTest 22](#_Toc14692)

[6.9.6 emoticonSend Test 22](#_Toc23849)

[6.9.7 MessageTips Test 23](#_Toc21840)

1. **Introduction**

Communication system is the realization carrier in the current application of instant communication technology, including server, client, database and other components. C/S and P2P are two commonly used architecture methods in software design.The C/S (client/server) pattern is an asymmetric structure. The structure is divided into client and server two parts, respectively to realize different functions, through the cooperation to achieve a complete the function of the Single model needs a better performance in the computer as a central server, other are for the client, the client can request to the server resources and services, resource sharing between the client and can't, Because the server needs to provide resources to multiple clients at the same time, the requirements on the server are relatively high.

This Android job is mainly to design a convenient instant messaging platform for social users to make friends. The system uses Single structure design, the client is based on the Android platform, the specific operation database request is migrated to the Single server, the server completes the database operation request, and through the XMPP protocol in data format, sent to the client, at the same time to reduce the pressure of accessing the database, SQLite database is used to store the information related to users' friends locally on Android, and OpenFire server is used to embed chat real-time functions into the database.At the same time, the instant messaging software is designed and implemented by the client to achieve the operation of chatting and picture data processing and other functions. Through the design and implementation of the software, the most direct purpose is to provide users with a convenient information sharing, the realization of friends and other functions of the communication system.

This document is composed of six parts, first of all, the first part is to complete the summary of the overall document in terms of requirements.

The second part is the description of the Android components required by the current designed software --FindOne. On the basis of the introduction, we complete the design work of the actual software development requirements for the components.

The third part of the conceptual framework, the rapid development of the framework to greatly improve the development efficiency, this part is mainly aimed at the software development framework design, focus on explaining the definition of the client related development framework and the actual software development operation.

The fourth part of the current version of the innovation characteristics and concepts, mainly introduces the current development of the software in the process of design and development of new innovation and new technical operations.

The fifth part of the development process constraints and related strategies, mainly discusses the actual software development process encountered in the relevant constraints and the actual handling strategies.

The sixth part of the debugging program, this part includes the current code function debugging and related unit tests and user acceptance tests, including the actual execution of the program code operation in the testing process.

**2.Detailed description of the components and its relevance to the requirements specified**

**2.1Activity**

Activity as the most important component in Android development, Activity exists as an Android page, based on the Activity to achieve multiple XML components binding and related events logical processing. This component in the realization of software chat function mainly as the main implementation basis, by creating an Activity and the corresponding XML file, and realize the binding with XML file to complete the relevant function realization. Among them, the Activity component is based on the data UI display related components :ListView, Button and so on.

**2.2 Service**

Service executes only in the background, has no user interface, and has a life cycle independent of other basic components such as activities. It is used to provide services that need to run in the background for a long time, such as data downloading. In the development process of the software, the sending of chat information between users is established under a WebSocket, and the continuous polling operation of Socket is realized through Service, and the relevant Service operation is committed to realize.

**2.3 content provider**

This component enables the application to provide data to other applications. The content provider component provides data from one application to other applications through requests.In the development process of the software, the content provider is used to upload the user's picture and obtain the relevant data. In the software, the user searches for friends and the library content access all use the mechanism.

## **2.4 Broadcast Receivers**

A broadcast receiver is used to respond to broadcast messages from other applications or systems. These messages are sometimes called events or intentions.In the development process of the software, the business logic of sending message prompt operation to the friend information after the user has accepted the application of adding friends is realized mainly through this mechanism.

**2.5 ListView**

As one of the common application components in Android development, ListView component displays specific data content in the form of a list, and can be displayed on the screen adaptively according to the length of data. This component inherits from the abstract AdapterView class. The use of this component is generally based on the Adapter Adapter for rendering implementation, the adapter provides rendering data for the Listview and the Listview only needs to provide a display list framework for the data, and then complete the rendering Settings of the respective framework. The main use of this component in the project is the list of friends, the list of chat rooms.

## **2.6 SharedPreferences**

SharedPreference is a lightweight way to store data for Android Settings. It stores data in a file in the form of key-value pairs and retrieves it when you need it. In the current project, this component is used to store user name information, and the biggest benefit of this component is that it can be used and retrieved between multiple activities without any clear tasks. This mechanism is the operation mechanism of obtaining account information and querying through SQLite database in actual operation.

**2.7 Handler**

Handler is the key technology of communication between main thread and child thread. Due to the technical regulations of Android development, child threads are not allowed to modify the UI of main thread, so for the case that child threads need to change the UI of main thread, Handler message mechanism appears. This mechanism mainly consists of Message, Handler, Looper and MessageQueue, through which the message is delivered and received between the sub-thread and the main thread. In this project, the main application of this mechanism is to realize the implementation of the message passing mechanism between the main thread and the child thread of the user.

**2.8 Adapter**

Adapter as a ListView list rendering tool, together with ListView to achieve the functional requirements of the list of friends, for the component Adapter including BaseAdapter, ArrayAdapter, SimpleAdapter, SimpleCursorAdapter, of which BaseAdapter is an abstract class, inheritance it needs to achieve more methods, so it also has a high degree of flexibility; ArrayAdapter supports generic operation, the most simple, can only display a line of words. SimpleCursorAdapter can be used for simple text-only ListViews, which require the Cursor field to correspond with the UI id. If you need to implement a more complex UI, you can override other methods. It can be considered as a simple combination of SimpleAdapter for database, which can easily display the contents of the database in the form of a list.

**2.9 asmack**

asmack as a common API for instant messaging implementation, the implementation of the technology is based on the XMPP protocol implementation, with smack as the client, openFire server as the server to complete the relevant instant messaging request processing implementation. asmack as open source software under the Apace License, is a simple, powerful class library in the information package coding at the information package level. It provides more intelligent classes such as Chat class, can make instant messaging capabilities to develop a significant increase in efficiency.

**3.Conceptual Framework**

**3.1 XMPP Protocol**

XMPP protocol, the full name is Extensible Message and Space Protocol, the predecessor of the protocol is Jabber, we take XMPP protocol master to implement IM mainly because we consider XMPP protocol is based on XML, it inherits the flexibility of development in XML environment. This indicates that XMPP is extensible, so XMPP messages can be not only simple text, but also carry complex data and files in various formats, which means that the XMPP protocol can be used not only for human-to-human communication, but also for software-to-software or software-to-human communication, and XMPP contains software protocols for the server to enable it to talk to another, which makes it easier for developers to build client applications or add functionality to a configured system.

**3.2 Socket**

Socket as a new age network communication protocol, this technology is a communication protocol used for arbitrary bi-directional data transfer between a client and a server. This technology is implemented based on the TCP protocol and includes an initial handshake process and a subsequent bi-directional transfer process of multiple data frames. Its purpose is to enable the server to avoid opening multiple TCP connections during frequent bi-directional transfers in order to save resources and improve efficiency and resource utilization. If the status of the server changes, the client needs to take a "polling" approach to make repeated requests to obtain the current response status of the server. The use of the above-mentioned round-robin request is very resource-intensive to the system, so later, after improvement, the Socket protocol was invented, the most important feature of which is the realization of the function that the server can actively communicate with the client, thus improving the utilization of network resources.

**3.3 SQLite**

SQLite is a process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine.SQLite is an embedded SQL database engine, and unlike most other SQL databases, SQLite does not have a separate server process.SQLite reads and writes directly to common disk files. A complete SQL database with multiple tables, indexes, triggers and views is contained in a single disk file. The database file format is cross-platform - you can freely replicate the database between 32-bit and 64-bit systems or between big-endian and little-endian architectures.These features make SQLite a popular choice for application file formats. As a built-in database of Android Studio, SQLite only requires creating a Class class, inheriting SQLiteOpenHelper helper class, and completing OnCreate and OnUpdate as well as the constructor of the current class. Use, very convenient.

**3.4 Message**

In the whole message mechanism, message is also called task, which encapsulates the information carried by the task and the handler to handle the task, and in the project, the logical code of the sub-thread is used to set the state of the Message's obj to complete the information storage operation, and the Handler mechanism is used to extract the message information to complete the related chat The design of data listening and related operations.

**3.5 AsyncTask**

There are two ways to implement asynchronous task mechanism in Android: Handler and AsyncTask. Handler mode needs to create a new thread for each task, and send a message to UI thread through Handler instance after the task is completed to finish the interface update. AsyncTask is an abstract class which is encapsulated by Android It can execute background tasks in the thread pool, then pass the progress and final result to the main thread and update the UI in the main thread.

**4.Advanced features and concepts added to this version**

Instant messaging in the traditional context is mainly realized by server-side processing of requests and real-time client-side response. In the process of actual deployment, this approach needs to be realized with the focus on server building, and a more stable server needs to be integrated in order to achieve the most stable operation. The server to build the server side of the way is relatively cumbersome, so based on this demand currently produces a variety of scenarios to build the SDK tools and integrated build server, the project uses an integrated server to openFire-based, the server only requires a key local installation can be used, by configuring the embedded database, and open the current port can be achieved. In the development phase, the client completes the storage of relevant user registration information and friend information through the SQLite database, and in the message chat interface, remote client calls are made through the asmack client interface of the XMPP protocol, while realizing the relevant instant messaging online chat request operations.Through the four key Socket driver functions of open, message, close, error, data information sending and connection establishment, error push and connection closure, release of resources and other operations, to achieve the push of users online and real-time message sending and display storage operations.

**5.Discussion of constraints encountered, and strategies applied during the development**

**5.1 Null pointer exception**

In the process of software development, due to the fact that the initial data is not set immediately for the network query database data in the process of JSON data transmission, null pointer exception occurs when accessing the query database.Secondly, when the user is registering and uploading the profile picture, due to the change of the initial root path, the local picture cannot access the relevant data, so the null pointer exception occurs.

**5.2 network connection exception**

Network connection errors occur on the basis of the communication failure between the client and the server. The IP address and port number for the communication between the client and the server are not set properly. As a result, the client fails to obtain the communication between the server and the client.

**5.3 SQLException**

Database abnormal operation, mainly appear in the user of friends to want to add a user to send the invitation, the other party agreed to add as a friend, then the database table users and establish the relationship between the field friends, again add buddy, will result in generating a record of the same id field, which can lead to abnormal database operations.

**5.4 IllegalAccessException**

No access permission exception. With the update of Android mechanism, the previous permission to directly obtain user privacy such as image library and address book has been set as the user needs to voluntarily submit user information whether the user is authorized to access. If you do not have the permission, you need to exit. If you have the permission, you can continue the access.

**5.5 RejectExcuteException**

An asynchronous communication exception occurs when an operation of the subthread's business logic code modifies a UI resource bound to the main thread. Therefore, for the actual running application, child threads are not allowed to modify the UI resources of the main thread, which can cause the application to crash and asynchronous communication exceptions.

**5.6 IllegalStateException**

In the process of project development, for the client connection problem of openFire server, the relevant port, ip address and domain name were not set in advance, which resulted in an error when the client connected to openfire server. The error content was as follows: Not connected to the server. After learning the error code, modify the related ip address and domain, configure the related server address information, and solve the error problem.

**6.Debugging and Testing**

**6.1 Login**

When the Android application runs on the relevant Android virtual machine device or the relevant simulator, the MainActivity page will be entered first, and the login interface will be set for implementation. The login interface mainly includes the user's mobile phone number and password. In addition, for a more humanized design, the button jump permission of remembering password and stealth login other functions is added, as shown in Figure 6.1.

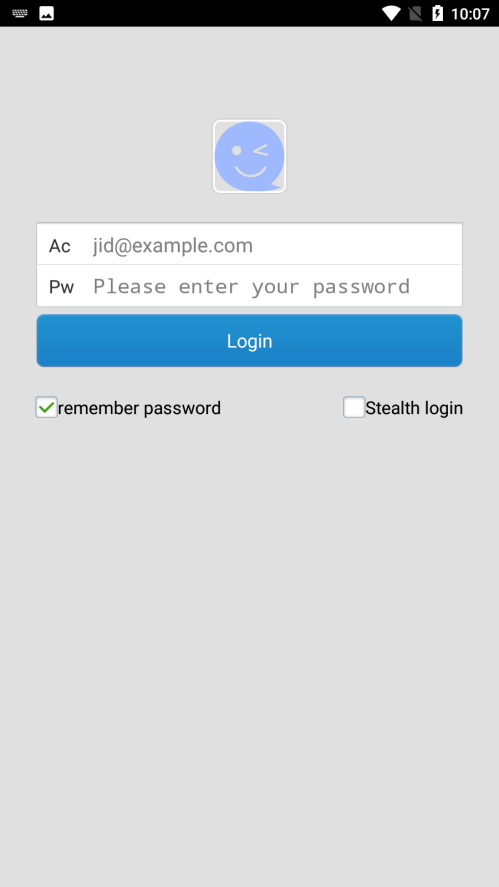


Figure 6.1 Login page

**6.2 Start Page**

The implementation of this page is mainly used to initialize some data needed for the project and pave the way for using the data later. The default data loaded on this page includes Layout static file and SQLite initial database and initialized XMPP protocol service, among which Layout static file is used to render Android application UI, SQLite database is used to initialize database table, XMPP protocol service is used to synchronize user data from openFire server.While the page has a certain cache time period to help the application to buffer time. As shown in Figure 6.2.

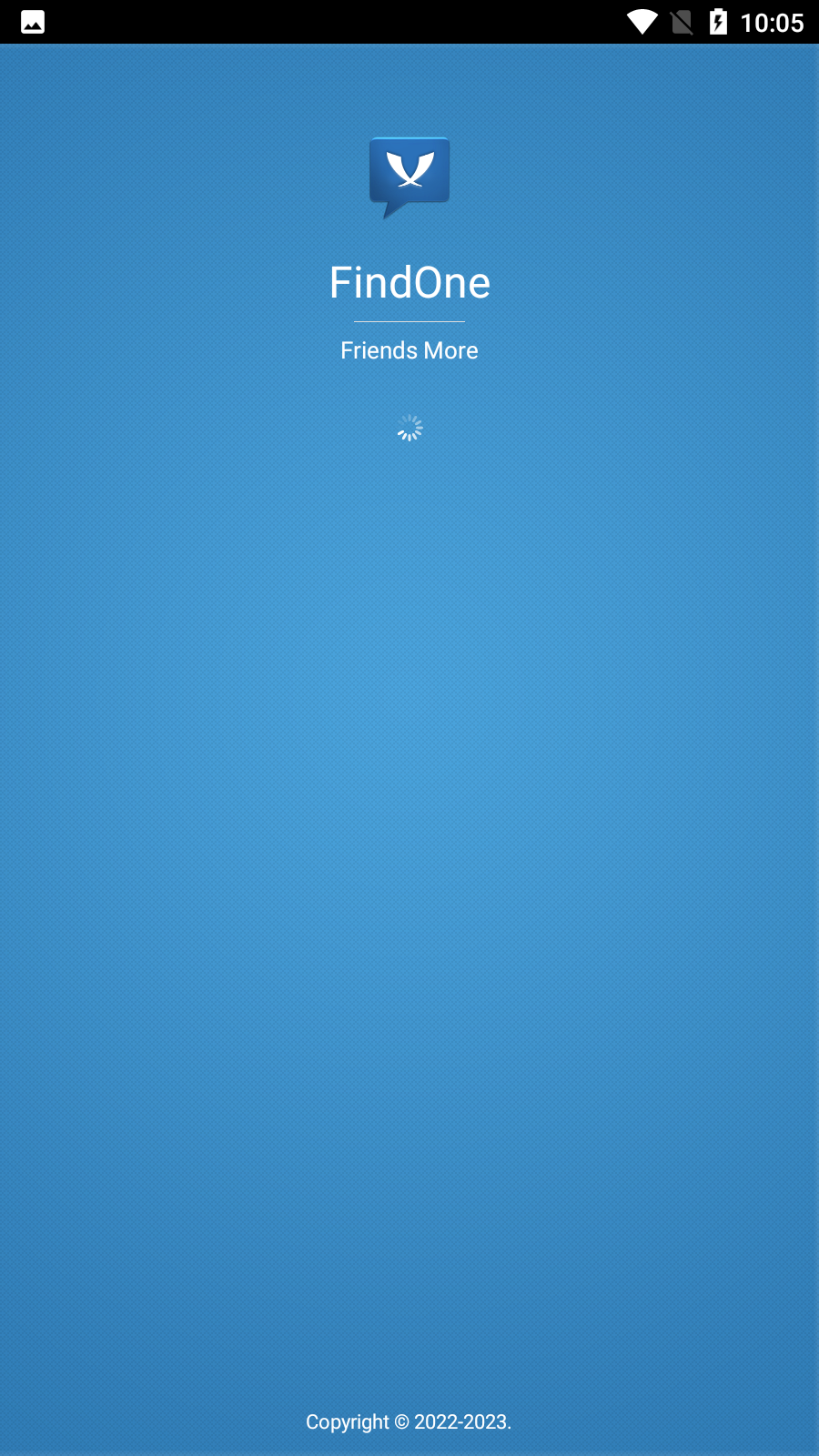


Figure 6.2 Start page

**6.3 FriendsList**

The list of friends is distinguishable from the online status of friends, while a long press on a friend will pop up an action alert box, the page is implemented mainly through the ListView and the rendering of the drop-down box to achieve. User data from the local server built by openFire server to generate friend data.When a user adds a friend, the user completes the search for the relevant friend account through the search box. After the search is completed, the user sends a friend request to the friend. At the same time, the relevant friend list information is updated in real time after the friend receives the friend request.As shown in Figure 6.3.

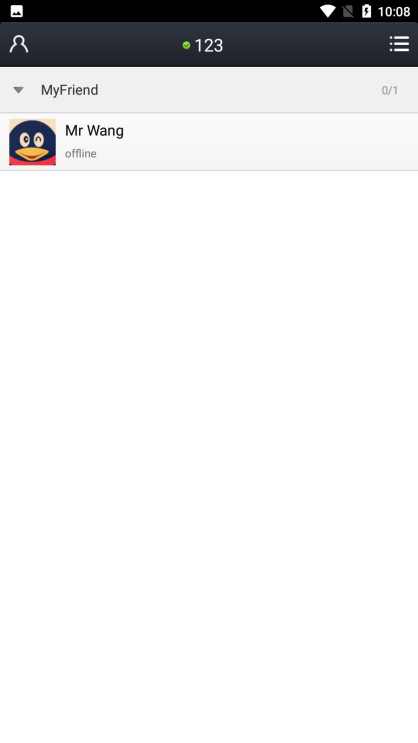


Figure 6.3 FriendsList page

**6.4 Home setting**

The main way to write the home page is to implement the menu style, which includes the message list, the list of users' friends and the details of personal information.

The page also includes a toggle for the user's personal online status, including online, offline, do not disturb, and busy. The page provides a variety of options for user usage, including silent mode, displaying offline friend information, displaying messages sent, messages received, disconnecting and reconnecting, sending error reports, as well as software usage feedback and related development information.This is shown in Figure 6.4.

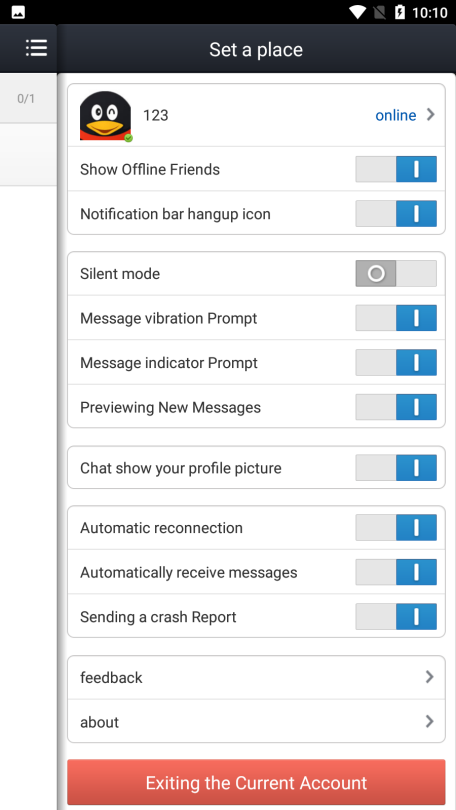


Figure 6.4 Home setting page

**6.5 Session List**

This page is the display page of the recent chat history of the current friend. It mainly includes contents such as user avatar, user name, first half of the message compressed information, message sending time and other information. If the message is received in online status, the relevant voice prompt driver will be called first, and the relevant message message box prompt will be completed and set. In terms of implementation, the recent chat log information on this page is based on SharedPreferences to save data. Using this method for data storage is mainly represented by the local storage records, and the data is stored completely in the local machine.This is shown in Figure 6.5-6.6.

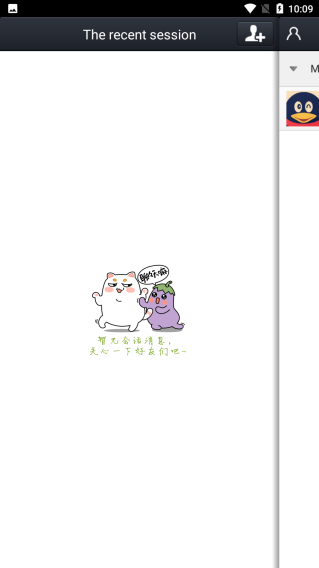


Figure 6.5 Session page

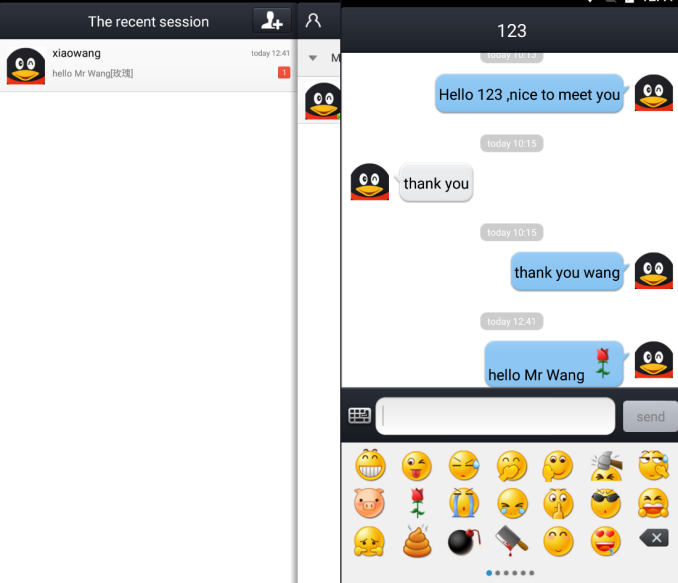


Figure 6.6 Session Test page

**6.6 FriendsChatting**

After adding friends, users can click on the list of friends to choose friends to chat or delete friends. During the chat, Socket and XMPP protocol will constantly search between the server and the client, so as to realize the operation of real-time information push.Here I use two Android emulators for the experiment, by configuring the static network parameters related to the two Android emulators to help complete the synchronization of the emulators' ip information, and by installing the APK file generated by them to the corresponding emulator, to test, by one side to the other side to add friends on the completion of the communication work, and to ensure the implementation of real-time communication technology.This is shown in Figure 6.7.



Figure 6.7 FriendsChatting page

**6.7 AddFriends**

After successfully logging in, the user can search the network database by knowing the friend account of the other party, or read the relevant contact phone number of the local address book to search, and the search results can be displayed in real time. Meanwhile, the synchronization operation of searching and adding friends can be realized.After searching for the relevant user, set the nickname remarks for the current user of your friend stored in your chat history for easy user identification, and complete the grouping operation of adding friends by creating a new group or selecting a group to achieve the operation of adding friends.This is shown in Figure 6.9.

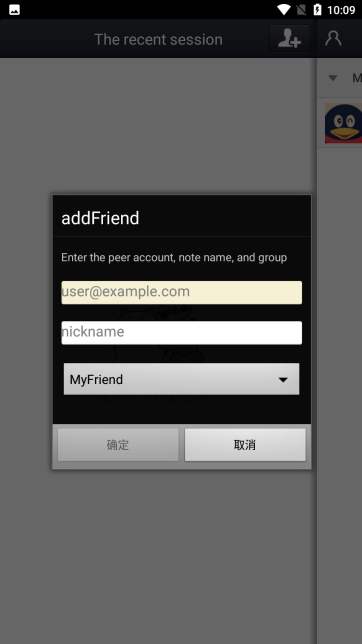


Figure 6.9 AddFriends page

**6.8 About page**

The implementation of this page is based on the user's personal information page, and the About option is called from the user's personal page to complete the display of relevant Android application-related information. The display includes information such as application icon, version number, author and email address.This is shown in Figure 6.10.

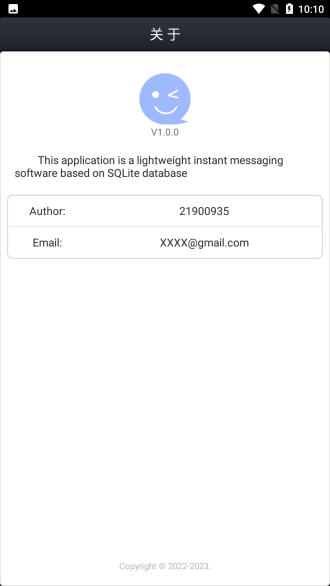


Figure 6.10 About page

**6.9 Test**

**6.9.1 LoginTest**

The user login test is first established after the openfire server has completed the relevant user creation operation, i.e. the server provides the user data for the application and completes the operation of the actual user function through the created user data. The landing page test is mainly based on the existence of the user, by following the relevant rules for input, to complete the normal login test implementation.The user login test is mainly implemented by the following three test cases, including the wrong account input, the wrong password input and the correct option input. Each test case has the corresponding test result reminder, as shown in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unit Name** | **Login Test** | | | |
| Id | Ac(account) | Pw(password) | Warning | result |
| 1 | Null | Null | Figure 1 | Failure |
| 2 | Null | 123 | Figure 1 | Failure |
| 3 | 123 | Null | Figure 2 | Failure |
| 4 | 123 | 123 | Figure 3 | Failure |
| 5 | 123@ | 123 | Figure 4 | Failure |
| 6 | 123@“my IP” | 123 | Figure 6.3 | Success |

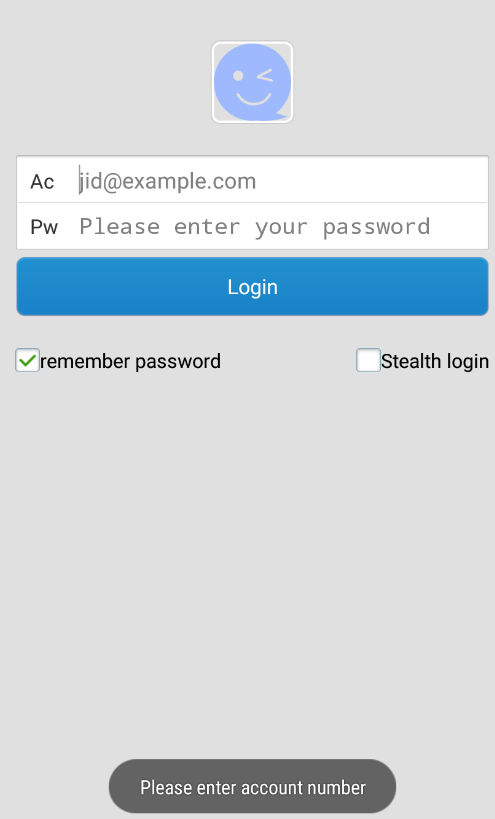
****

Figure 1 test 1

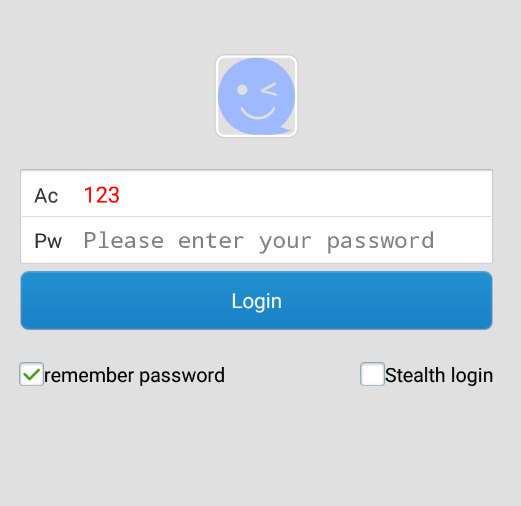
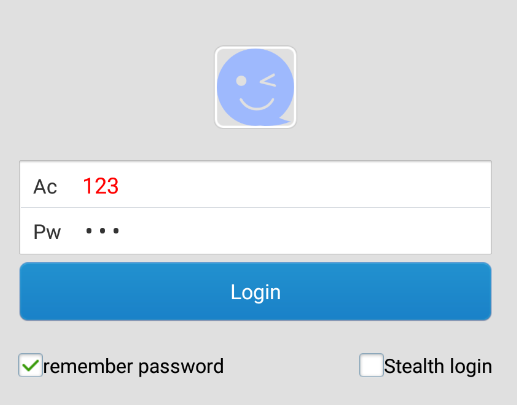
** **

Figure 2 test 2 Figure 3 test 3

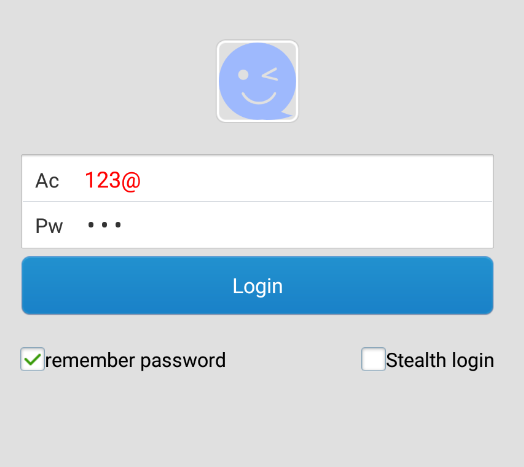
****

Figure 4 test 4

**6.9.2 RememberPasswordTest**

Remember password test need to achieve the function is, when the user after the first successful login, before the login check the remember password option, then need to achieve the effect is, after the user logged out and open the Android application again, then do not need to login again, directly into the main page of the program to carry out related operations. This is the correct function flow.As shown in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unit Name** | **RememberPassword Test** | | | |
| Id | Ac(account) | Pw(password) | Checkbox | result |
| 1 | 123@“my IP” | 123 | Yes | Success |
| 2 | 123@“my IP” | 123 | No | Failure |

**6.9.3 AddFriends Test**

The test cases of the registration function mainly include the correct format input of the mobile phone number and the regular input of the verification code, as shown in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unit Name** | **AddFriends Test** | | | |
| Id | Acoounut | Nickname | Group | result |
| 1 | Null | Null | Default | Figure 1 |
| 2 | 123 | Null | Default | Figure 2 |
| 3 | Xiaowang@192.168.111.11 | xiaowang | New | Figure 3 |
| 4 | Xiaowang@192.168.111.11 | xiaowang | Default | Figure 6.3 |

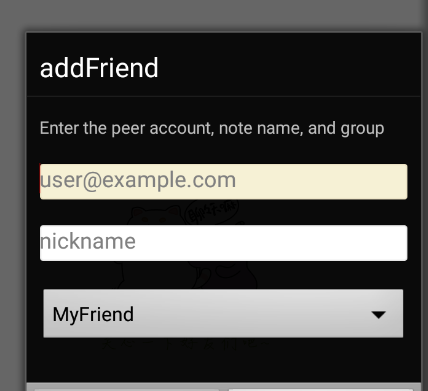
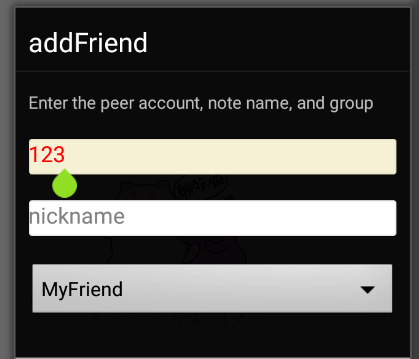
 

Figure 1 test 1 Figure 2 test 2

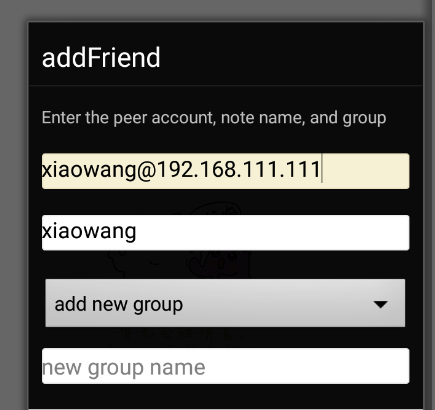


Figure 3 test 3

**6.9.4 Home Test**

The implementation of the user home page mainly focuses on the switching of user status, the opening and closing of relevant function settings, the submission of application feedback and the operation of application-related information. The user can turn on and off the relevant data controls through manual operations to achieve the relevant setting functions. As shown in the figure below.

|  |  |  |
| --- | --- | --- |
| **Unit Name** | **Home Test** | |
| Id | Operation | Result |
| 1 | Status Switching | Figure 1 |
| 2 | Control Switch | Figure 2 |
| 3 | Feedback Submission | Figure 3 |
| 4 | About | Figure 6.10 |

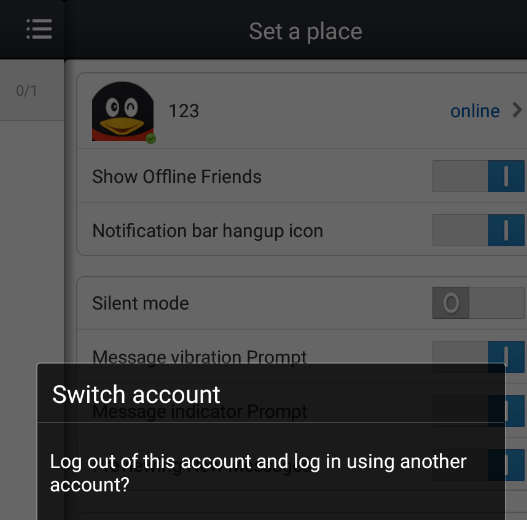
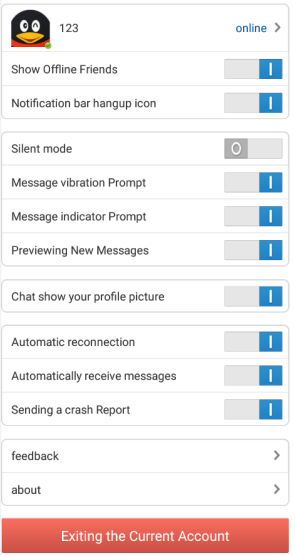
 

Figure 1 Status Switching Figure 2 Control Switch

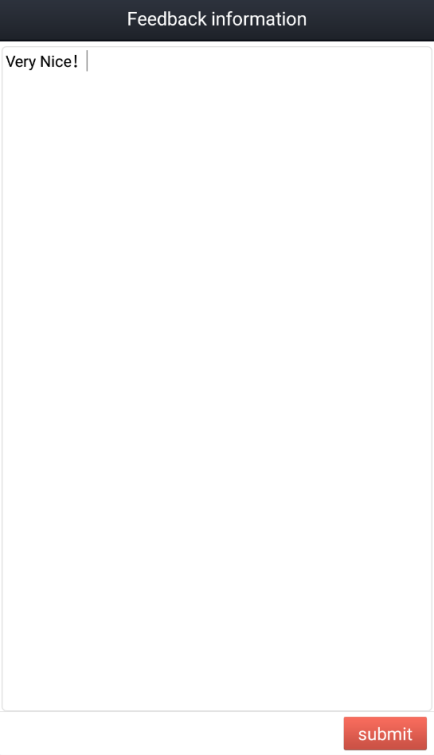


Figure 3 Feedback Submission

**6.9.5 ChattinigTest**

After the completion of adding friends, click the friend information to choose to delete friends or start chatting, and jump to the chat window to realize the function of chatting with friends.As shown in the figure below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit Name** | **Chatting Test** | | |
| Id | Sender1 | Sender2 | Result |
| 1 | “Hello Mr wang” | “Hello” | Success |
| 2 | “thank you” | “thank you” | Success |

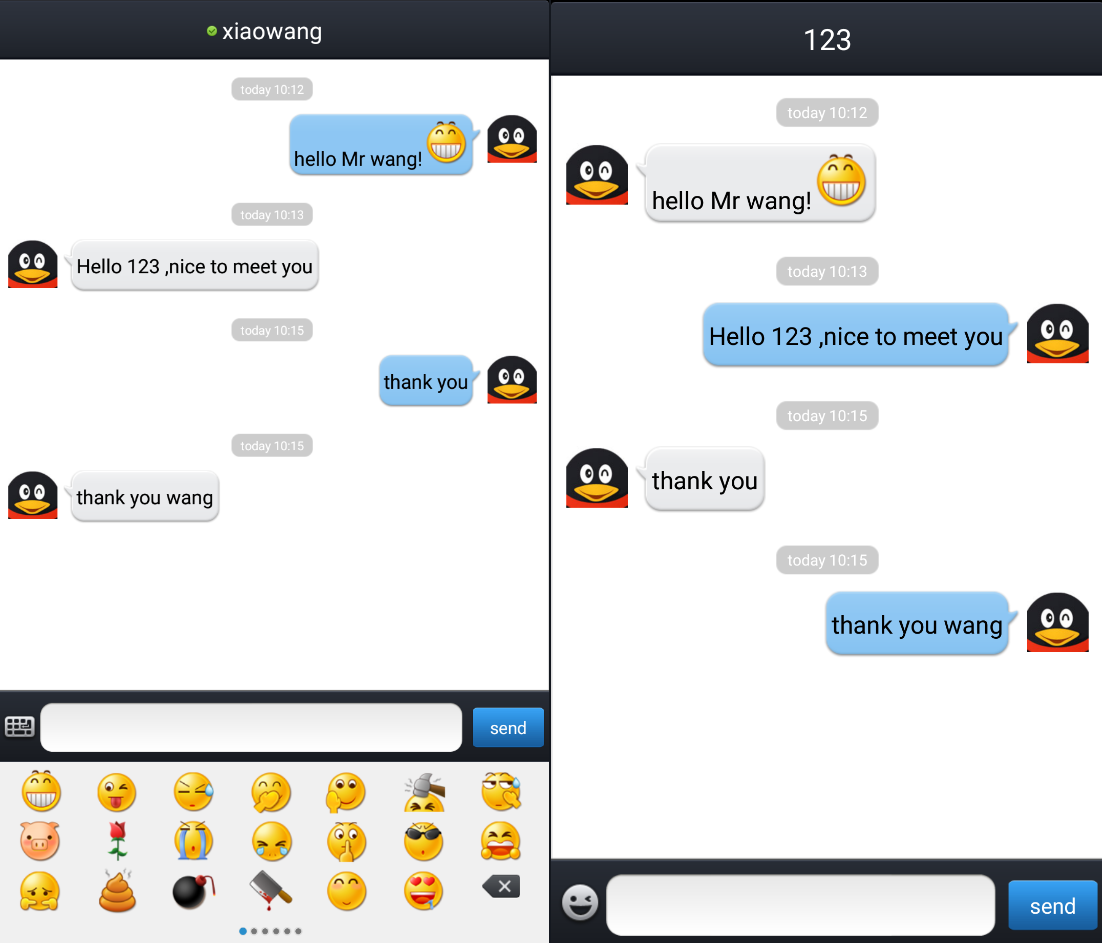


Figure 6.11 chatting

**6.9.6 emoticonSend Test**

Compared to the previous version, the application has the function of sending emoticons to chat with the user, by setting multiple emoticons in pages, while the operation of static images in each emoticon enables the dynamic sending operation of emoticons. This function is realized on the basis of implementing online chatting for users. The test diagram is shown in Figure 6.12.

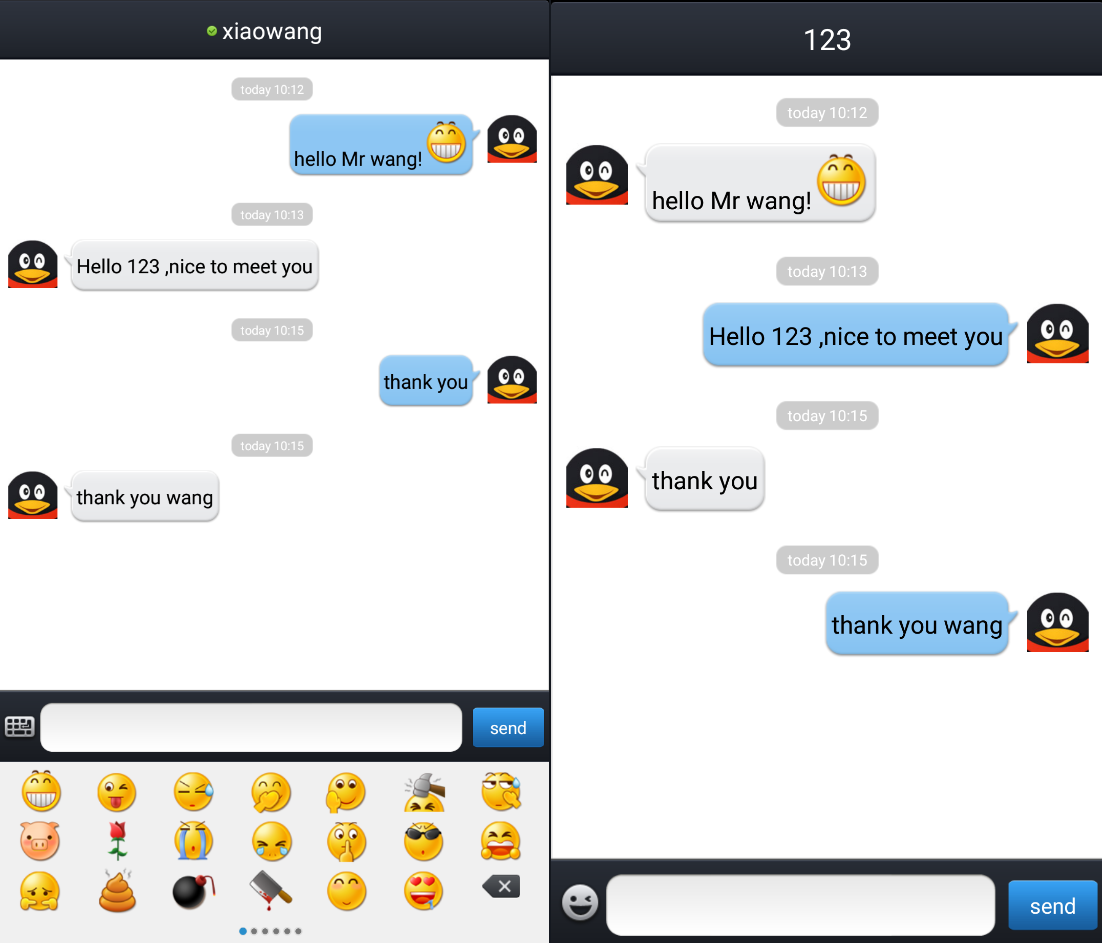


Figure 6.12 emoticonSend

**6.9.7 MessageTips Test**

When the user sends a message to another friend through the application, if the current friend does not open a chat with the current user, a message prompt is required. The message prompt implemented here mainly has a message prompt red logo and a ding-dong tone to help the user understand the voice prompt implementation in real time, as shown in Figure 6.13.

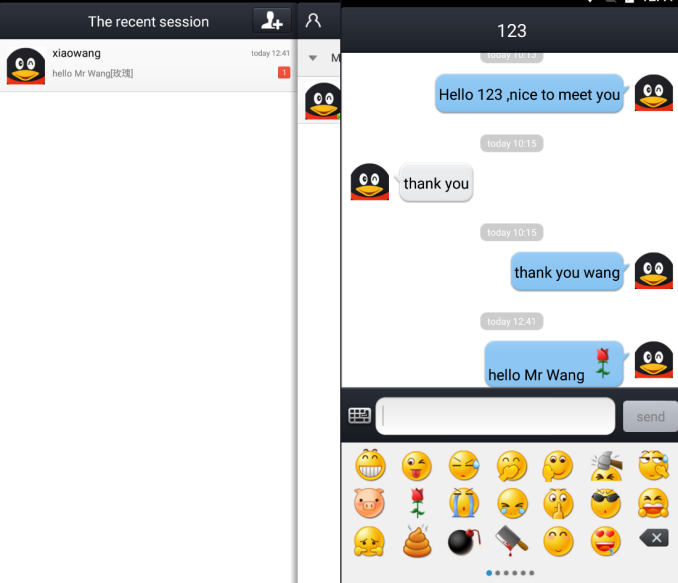


Figure 6.13 MessageTips