The mobile terminal of our project is an application built with android framework that can be shared by employees and users. It has a independent server background processor and integrates instant messaging module.

The core function of the application is to upload or download data to the server through the HTTP network protocol. The implementation of this function is to send POST requests to the server by creating a new OkHttpClient object (which is derived from class OkHttp3[11]). Since any time-consuming operation in android cannot be performed in the main thread, we chose AsynTask, an asynchronous thread to perform network data transmission in the sub-thread. Meanwhile, we also need to generate the header for the POST request through the FormBody before transmission, and add the data we want to send to the header through the Request object; After sending the request, the JSON format data returned from the server is received through the Response object. Finally, Google's Gson[9] class is used to parse the returned data. And Other functions will be analyzed according to the android activity page.

Common Page

The first interface MainActivity.java is an animation interface, showing the animation effects for trickling in and out, and the techniques involved in this page are:

1. Implementing delayed UI operations in sub-thread using Handler's postDelayed() method, which corresponds to the thread's sleep() method, with a delay of 1500ms (that is 1.5s). The purpose of this design is to complete the preloading of the application. Within 1.5s, it is necessary to complete the check of local memory and cache, determine whether there is a login of the existing user and check the identity of the logged-in user, whether it is employee or client, and then decide whether to jump to the user interface or the customer service interface, or go to the login interface.
2. More importantly, preloading also completes the loading and deployment of the third-party instant messaging SDK. This includes the detection of network, camera, memory card storage and access permissions and some mobile phones required by the push configuration, to ensure that we can still receive instant messages and feedback from users or employees when we close the application (not log out).

The second page LoginActivity.java contains the following functions:

1. Input field uses a listener with focus change (setOnFocusChangeListener()), When the focus of the input field changes, both the image above the input field and the icon in front of it will change to highlight the field.
2. After successful login, the user name and user type will be cached by internal storage, which will be provided to the application for uploading and judgment.

Client Page

1. Registration

The third page RegisterActivity.java completes registration, Only user accounts can be registered here, and employee accounts need to be assigned by the company. The methods and techniques included in this page are:

1. All input fields on the registration page use the TextInputLayout and addTextChangedListener to preprocess the data entered by the user. Also listen for text changes, including listening for the length of the current input string (password length must be greater than 6 characters, less than 16 characters), whether the two password inputs are consistent, and whether the captcha input matches the content on the graphic captcha. And any item typed as null cannot be submitted to the server.
2. Verification codes are generated by Verification.java. The process is:

①First declare an array of random numbers, removing confusing numbers and letters, such as: 1 and I, or 6 and b, etc, and store the remaining 45 elements {'2', '3', '4', '5', '7', '8', 'a', 'd', 'e', 'f', 'g', 'h', 'j', 'k', 'm', 'n', 'p', 'r', 's', 'u', 'v', 'w', 'x', 'y', 'z', 'A', 'B', 'D', 'E', 'F', 'H', 'J', 'K', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'} into the array, and pick any four of them.

②Draw interference lines, generate random colors, randomly generate text styles, colors, thickets and gradients, etc.

③Set four random characters(①) and elements(②) in the bitmap and return the generated bitmap to be rendered in the interface and the captcha string ready for matching. A new captcha image is switched, when the captcha picture is clicked.

1. Fill personal information

The fourth page Perfect\_Info.java is for filling in personal information, including the following functions：

1. The user's phone number and email address are still monitored by a text listener. It also uses regular expressions to verify that the phone number or E-mail address is in the correct format. The regular expressions used are as follows:

Phone:"^((17[0-9])|(14[0-9])|(13[0-9])|(15[^4,\\D])|(18[0,5-9]))\\d{8}$"

Email:"^([a-z0-9A-Z]+[-|\\.]?)+[a-z0-9A-Z]@([a-z0-9A-Z]+(-[a-z0-9A-Z]+)?\\.)+ [a-zA-Z]{2,}$"

1. IM authentication[1][2]

The fifth page is the authentication page, Only the first time into the mobile terminal is needed. This page will complete the registration of new users for instant messaging. The technique involved is calculating the value of CheckSum (CheckSumBuilder.java). In detail, CheckSum is SHA1(AppSecret + Nonce + CurTime), a string concatenated by three parameters, taking SHA1 hash and converting it to a hexadecimal character(string, lowercase), and upload it to the cloud database and kept in sync with our project database to ensure the consistency of data.

1. Home page

The sixth interface is the HomePage.java. The interface layout uses techniques that include displaying the sidebar through DrawerLayout and NavigationView, a ViewPager that can swipe through pages, and a BottomNavigationView[10]. Meanwhile we divided three tabs, the first tab is the claim order, the second tab is the item list, the last is instant messaging.

* 1. Claim order information

In the claim order interface (FragmentHome.java), we added the ability to determine the current status of each claim order and to set the color for the different state by setTextColor(). It also enlarges the currently displayed claim order by determining where the current claim item is in the list.

* 1. Item list

In the item list interface(FragmentThree.java), RecyclerView, CardView, swipeRefreshLayout (drop-down refresh) and FloatingActionButton[10] are used for layouts techniques. And this page also contains the following functions:

1. The first function is to add photos of items. There are two optional ways, one is to call the phone's camera to take a photo, and the other is to open the phone's album to select photos. According to Google's android developer documentation[6], the key to taking photos and selecting photos from albums is to dynamically check the read and write permissions for camera and storage card data[6][7], and to use content-provider instead of clear text to save the data storage path[8]. When we pass the check of the dangerous permission, open the camera or album, select the photo and finish clipping with the system tool[3], it will be stored in the album and return a parameter to the application, namely the absolute path of the picture[6][8]. In order to facilitate the insertion of image information into the database, and to better share data with the web side, we choose to turn the absolute path into a bitmap firstly. And then, the image is compressed with methods ByteArrayOutputStream(). After that, converting byte array output stream to byte array (byte[]). Finally, using Base64's default encoding (base64.encodetostring()) [4][5]to transfer the byte array into a String, and uploading to the server.
2. When we get the Base64 encoding of the image from the server side, we turn it into a byte array using the base64.decode()[4][5] method and present the byte array as an image through the Glide library.
3. When we click on the picture of each item in the list, the picture will be enlarged for picture preview, and can be enlarged double and dragged for preview.To achieve this, we mainly used photoView technology (ImageZoom.java).
   1. IM contact and chat[2]

In the IM interface(FragmentTwo.java), This page integrates with the third-party SDK so that users or administrators can communicate with each other instantly, and we use the following functions:

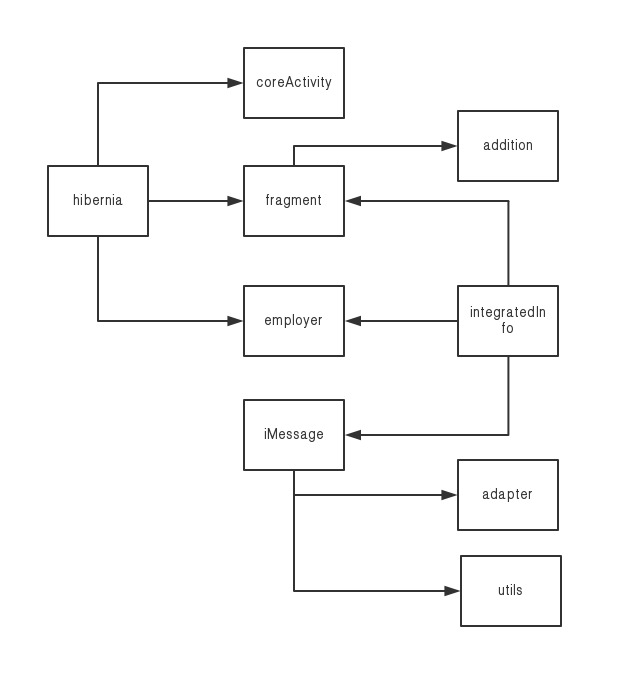
1. Send message: create the message object through the interface provided by MessageBuilder, and then call the sendMessage interface of MsgService to send it out.The messages are divided into text messages and picture messages, and the chat type is P2P single chat mode.
2. Receiving messages: by adding a message receiving observer, MsgServiceObserve#observeReceiveMessage, our project can receive notifications when new messages arrive. A typical scenario of this code is the message dialog interface, which registers a message-receiving observer in the onCreate interface and logs out the observer in onDestroy.Upon receiving the message, determine if it is the message of the current chat object, and if so, add it to the list for display.
3. Sidebar

The final user interface is the drawer layout, in which we add the user's personal information to the local cache to speed up data processing.At the same time, it also adds the function of exit, users can log out from the current interface and instant messaging system.

Employee Page

The employee interface refers to the file E\_WelcomeActivity.java &E\_ListorderActivity.java and the functions that are included are:

1. Determine which of the three status an employee has chosen (processing, approving or denying). If the employee selects the “processing”, the claim form will have the "approving" or "denying" buttons, others will not.
2. The employee interface will also greet employees according to the current system time of the device. For example, at 9:30 AM, the page will show "Good Morning". This is done by getting the Calendar.TIME.
3. The employee interface also integrates instant messaging and chat. The principle is the same as above. And the user name list of all the current users will be obtained through the POST request for the convenience of communication.



Image(1). Android Project Package Structure

1.https://wenku.baidu.com/view/d669ef93cd22bcd126fff705cc17552707225ece.html

2.https://dev.yunxin.163.com/docs/product/IM%E5%8D%B3%E6%97%B6%E9%80%9A%E8%AE%AF/SDK%E5%BC%80%E5%8F%91%E9%9B%86%E6%88%90/Android%E5%BC%80%E5%8F%91%E9%9B%86%E6%88%90/%E6%A6%82%E8%A6%81%E4%BB%8B%E7%BB%8D

3.https://blog.csdn.net/alex01550/article/details/82115074

4.https://blog.csdn.net/merbn/article/details/80410053

5.https://blog.csdn.net/qq\_35372900/article/details/69950867

6.https://developer.android.com/training/camera/photobasics

7.https://stackoverflow.com/questions/39787129/permission-denial-writing-android-support-v4-content-fileprovider-uri

8.https://www.jianshu.com/p/7be437d75768

9.https://blog.csdn.net/aa6385422/article/details/52511547

10.https://blog.csdn.net/fly\_li\_sir/article/details/79704021

11.https://square.github.io/okhttp/