City University of Hong Kong

Department of Electronic Engineering

EE5415

Mobile Application Design and Development

(2019-2020 Academic Year Semester B)

Final Report

[Wallet Keeper]

Group Number : [15]

Group Leader : Zhou Ziyang(55921210)

Group Members : Fang Yuxin (55941269)

Li Huimin (55997947)

Shen Yaojun (55988822)

Zhang Jianing (55831831)

Table of Contents

Abstract

As the increasing popularity of smart phones in people's lives, the need for smart phone-based personal accounting and financial analysis tool growing importance, bookkeeping mobile application shows huge potentially market.

This project aims to develop an individual bookkeeping app based on Android Studio. We will use Android Studio as development tool, and calling SQL database, use java as the develop language. Apply the plug-in RxJava to organize and calculate the bills.

Compared with other bookkeeping apps in market, Wallet keeper has some outstanding features. Firstly, It is simple to operate and the interface has no advertisement. According to our market research, many similar apps link to social network, so it is necessary for users to create a new account. Our app can be used without any registration. Secondly, ‘wish list’ function is useful for people who don’ t have regular income or salary arrangement. But if they need a sum of money to support their planning., this function will improve their sense of saving money. We all know that it is not easy to develop a habit, especially beginning, so we hope with the encourage of the wish list, users can have the motivation, and develop bookkeeping habits.

Chapter 1: Introduction

1.1 Background

With the continuous development of economy and technology, a rise in individuals' disposable income create increase purchasing power, thus simulating demand for goods and service. Various and convenient mobile payments, scanning QR code or using credit cards, have been changing people's habits of consumption dramatically that makes consumers usually spend more. 'Where did the money go? How much did I have in total? Which is my consumption level? What is my expenditure ratio?' These problems haunt us all the time. If we can not solve these problems, we will consume blindly, which will eventually make us push debt higher and living beyond our means.

Personal bookkeeping is one of the most overlooked skills in modern life. Bookkeeping is to record, arrange, and analyze consumption transactions in our lives, which can manage our consumption transactions well. What this skill can do is to record every consumption transaction, output a report reflecting our consumption behavior and analyze our consumption structure through these consumption data. Subsequently we are able to adjust pattern of consumption according to records. Personal bookkeeping make it easier to understand 'how much money we have, how much we can control, what are all the shortcomings, what are our net assets', and 'where our money has gone'. Therefore, accounting products came into being.

1.2 Market research

All of bookkeeping and wealth management application can be divided into two classifications: information service function and financial service function.

The information service-type including basic bookkeeping function that refers to the function of the user's life information and expenditure management. Such as account management, reimbursement management, credit card bill management, community strategy and other functions.

The financial service-type means that the mobile application combine basic bookkeeping function with personal finance management, providing comprehensive financial services for individuals. Functions for users including perform asset management and financial services, such as purchasing financial management, loan application, credit card application and other functions.

Because financial service-type accounting applications can be undertake part of bank business, thus they are generally developed and operated by official financial institutions such as banks. There will not be much discussion here. We mainly study information service-type accounting applications.

According to the data of "China's Bookkeeping and Wealth management application Mobile Market Quarterly Monitoring Report Q4 2015" released by Analysis think tank, China's bookkeeping and wealth management application is experiencing a significant increase in the total number of active users, the number of users has reached 16 million, and by the first quarter of 2016, the total number of active users of the bookkeeping and wealth management application totaled 21.1643 million, an increase of 30.56% from the fourth quarter of 2015[1].

While, at the same time, the users' stickiness slightly decreased. The number of per capita daily usage time decreased to a certain extent. The increase in the number of bookkeeping applications is directly related to the popularity of mobile phones. Persisting in bookkeeping makes people feel a

sense of accomplishment, but insisting on recording every day is indeed a tedious task—opening the application multiple times a day will make people bored—and require patience.

It can be concluded from the reduction of per capita daily usage time that bookkeeping application is difficult to become an application with high frequency usage. Actually, the 7-day retention of bookkeeping applications is common. Thus, the threshold for accounting needs to be lowered, and the convenience of application needs to be improved.

So the simplicity of user interactions is important, and can have a direct return in increased user engagement. Some of bookkeeping application developers link their applications with social network community or mini-games in order to attract and retain users that means it is necessary for users to create a new account before starting. However, complicated operation interface and process can demotivate, rather than motivate users. From Google store, it can be found that for utility applications, the easier the application was, the higher the score.

Chapter 2: Application Design

2.1 Functional design

We designed four functions for this app, including new addition, bills, diagrams and wish in different user interfaces.

1. In the addition function, we designed nine types for accounting, users can choose date and the type for bills, and record the price and comments for every bill.

2. In bills function, users can check and search all past records in details on monthly or annually.

3. In diagrams function, users can analyze their spending from pie chart or bar chart in certain month or year,the pie chart show the percentage of nine types of the bills,and the bar chart show the daily cost if you choose one month, and if you choose one year, it will show the cost for every month in this year, the pie chart show the amount bills and the percentage of each type,which help users know their expense structure clearly, users can see where their money is going,how much they paid for it, so that they can balance their expense better in next month.

4. We also add a special function called “wish list”. Users can set a target amount for something, and input their monthly income. This function will calculate users’ savings by minus total expense, then update progress bar.

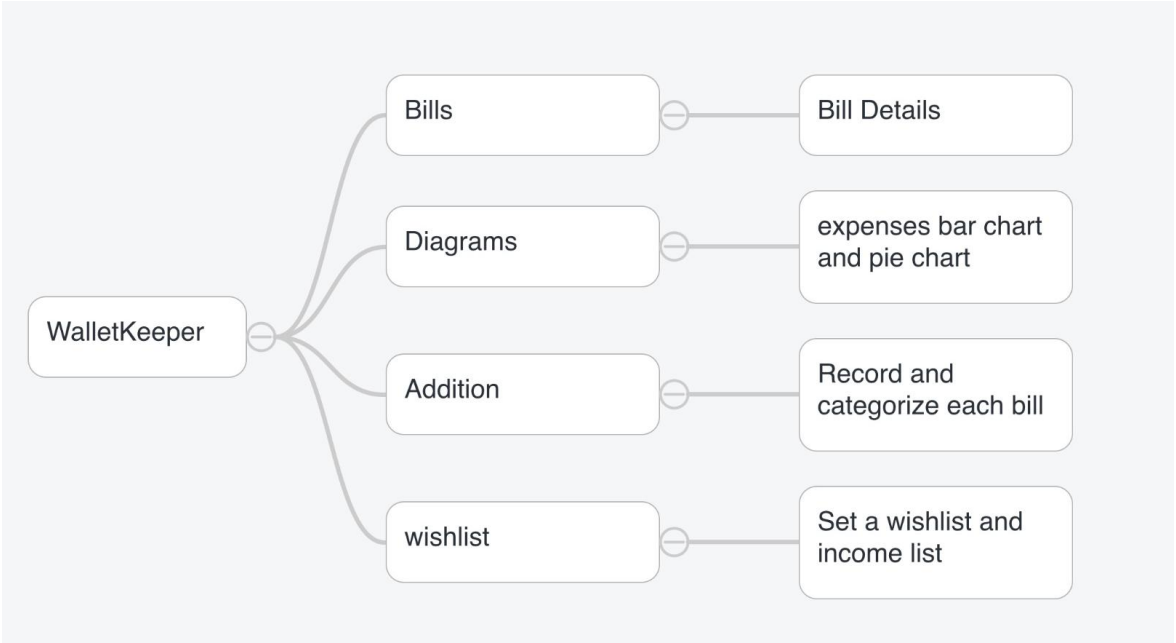


Figure1: Structural diagram

2.2 UI design

1. Log in:

Users press start to enter this app, and press clear to clear all the data, in this interface,we design a cute background picture and add our logo to make it unique.

2. Addition Interface:

Users can record the details about the expense here,choose the type and date, when you select one type, this icon will change the color from gray to yellow, input the price and comment, then click the addition button, the bill recorded can be found in bills.

3. Bill Interface:

In this function, users can see the sum of expenses for certain month or certain year ,which users can choose on the top of the interface,and see the total bills for each category.

4. Diagrams Interface:

This function show the expense daily in the form of a bar chart and pie chart, and the users can select month or year button to see the corresponding chart.

5. Wish list Interface:

Users can set wish list and add the income in this page, there is a progress bar under the wish name,which show the rate of the progress, you can input your income in the income list, then this function will calculate the progress of the wish.

Figure2: Log in

Figure3: Addition

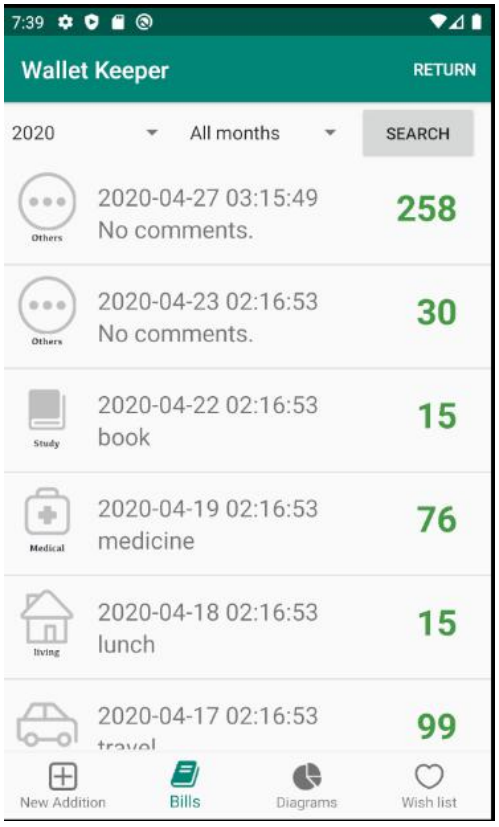


Figure4: Bills

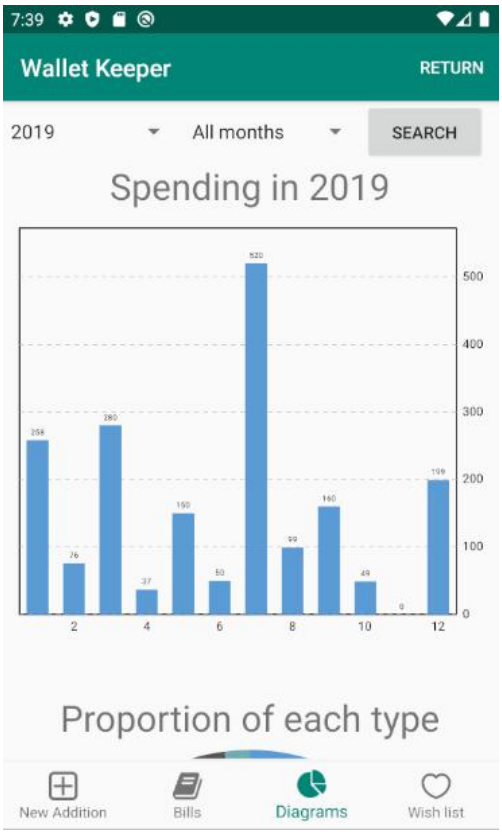
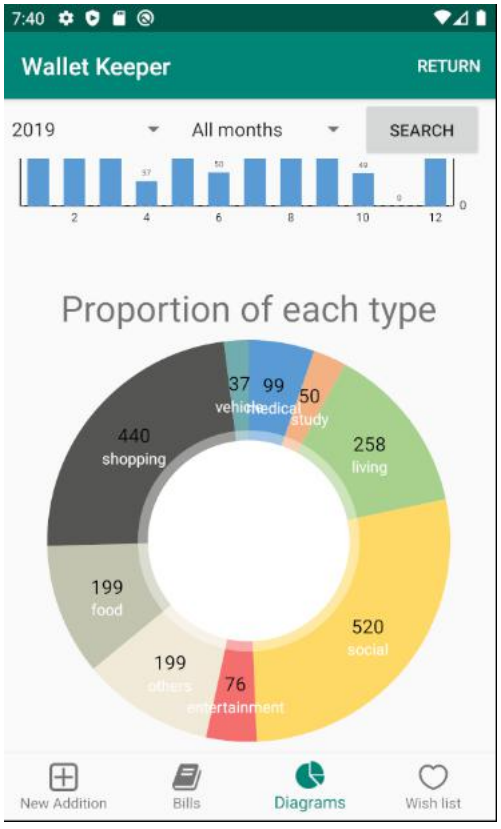


Figure5: Bar Chart



- *Figure6: Pie Chart*

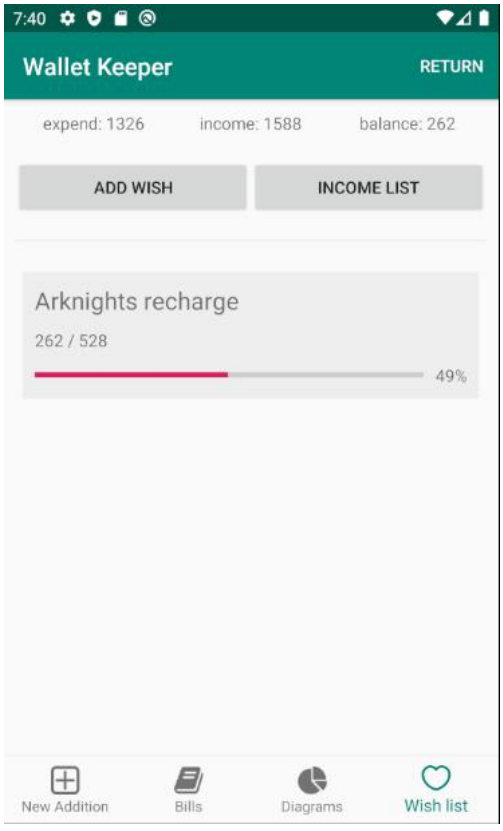


Figure7: Wish list

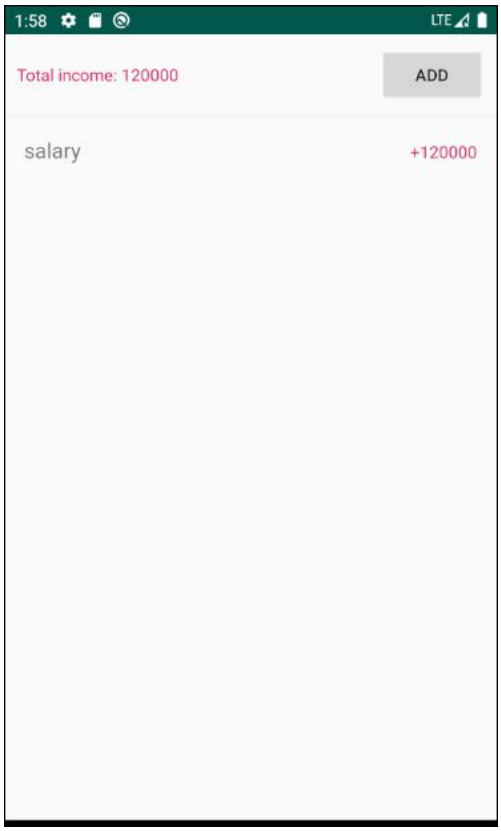


Figure8: Income List

2.3 Competitive Analysis

Take three bookkeeping apps as an example, they are not significantly different in the overall architecture, even very similar in user interface design and report output. However, in terms of function selection and operation optimization, there are still differences. The following is a selection of three bookkeeping app, Daily Cost, Shark Accounting, and Fortune City for comparison and analysis.

In terms of product functions

1. Daily Cost: It is easy and convenient for accounting, and provide multiple currencies as unit of account. This function makes app more international and facilitates people who travel overseas frequently. However, it is precisely because the product orientation is simple, there is no extra functions except basic accounting. That seems to be low price performance radio compared with high price in Google play store and Apple store.

2. Shark Accounting: The most distinguishing feature is that there is no independent user system in application. Users can synchronize their icloud or Google account in app without new enrollment, so that we can use it directly after downloaded. Besides notification setting is available—ability to remind users to add daily expense on time. Shark Accounting also link with social network, thus users can shared their accounts and communicate with each other. It can be regarded as a social application.

3. Fortune City: This is a good combination between recording daily expense and simulation games like Sim City, thus making bookkeeping much funny and less boring. That is both a strength and a weakness. Due to game-play, user interface and operation are more complicated than other bookkeeping applications.

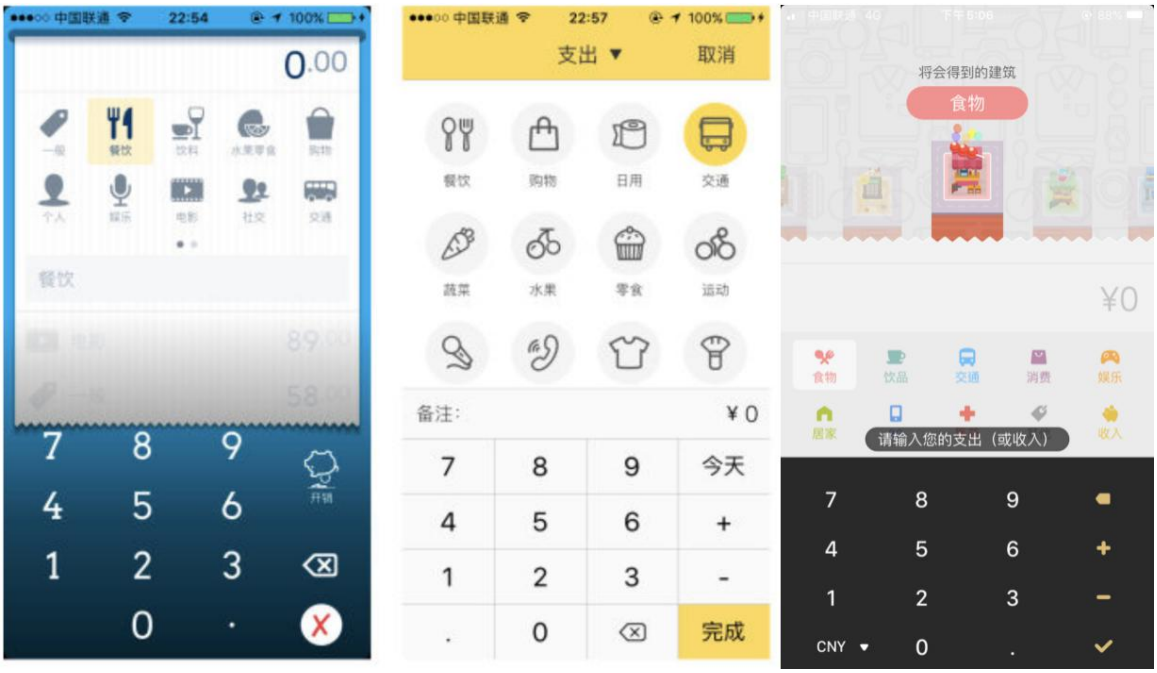


Figure9: Basic function (Daily Cost, Shark accounting, Fortune City)

Another important function is the chart of account to analyze receipts and payments. The statement of account is something that every bookkeeper will care about. From the chart, they can find their own income status and arrange property income appropriately. Thus for an bookkeeping software, its account expression can also be used as a criterion for determining whether the App is "good".

1. Daily cost is more comprehensive in the form of the account statement. It also uses line charts and pie charts to list the bill details in units of days. The line chart of the week is shown above, which is very simple and clear, allowing users to understand the expenditure promotion of each part and spending trends.

2. Shark Accounting mainly uses line charts, but you can query spending trends by week, month, or year. It displays expense and incomes details in a streamlined manner with clear and intuitive information.

3. Fortune City uses histograms and pie charts. The pie charts are used to show the cost ratio of each part. The histograms are used to visualize the balance of payments, so that users can intuitively understand their debts and balances.

Figure10: Account statement (Daily Cost, Shark accounting, Fortune City)

For structure, three products have nearly similar structures: bookkeeping, charts, personal settings. As can be concluded from the comparison of operational process, the clearest structural classification is Shark Accounting. It can helps users to find what they want according to their purpose, followed by Daily Cost, which has some functional structures, accounts and records. The set of functional sections is very comprehensive and international with different language versions. Its product positioning is simple with basic functions. The worst is Fortune City, although it has many functions even mini game, the boundaries of the structural classification are not very clear, resulting in some confusion for users.

As for interaction, Daily Cost is simplified, Shark Accounting is disciplined and style uniform, and Fortune City has more interactive forms.

1. Daily Cost pulls down the bill to add the accounting form is very vivid and interesting, but also allows users to understand at a glance, grasping the use method and purpose of use. It allows users to view account statistics on the horizontal screen which makes the user interface more flexible. While Daily Cost has some bad interactions, for example, if users want to cancel a note book, he should delete all numbers and text firstly. Besides expenses and income must also be selected before entering the amount and cannot be changed afterwards.

2. The overall interaction of shark accounting is quite satisfactory. There are not many highlights and many shortcomings. Its remarks will record the words you entered before as a label, which is convenient for use in the next remarks. This is very good.

3. The biggest feature of Fortune City is its game nature, thus user interface is lovely with plenty of icons. Given the game nature, it is not easy enough. What's more, there is no secondary labels for bills and sorts them according to the recorded time rather than the consumption time that will make users messy.

For the profit model, most bookkeeping apps are mainly make profits by inserting advertisements in user interface or upgrading VIP to enable more functions.

Chapter 3: Technology Details

3.1 Packages

1. Database : SQLite.

Use SQLiteOpenHelper to build database and save data. Use method provided by SQLiteOpenHelper and make SQL operating string like “where year=’2020’ ” to complement addition, delete, select and modify operations in database.

2. ListAdapter:

A class extends baseAdapter. Put data into listAdapter. It can link to ListView and generate ListView with auto-adapt length according to the amount of data.

3. Time package:

A class which use Calender to get instant time.

3.2 Activity

1. MainActivity:

AppBarConfiguration and NavController can generate and control bottom navigation bar. All four main function pages are fragments which are controlled by MainActivity. Therefore we can quickly switch between multiple page fragments.

2. HomeFragment:

Addition page. Use database package to link database and complement addition function. 9 imageViews can decide the type of consumption.

3. DatePickerDialog:

Time selector, which can choose the time of consumption record.

4. DashboardFragment:

Bill list page. Use database package to link database and complement deletion, selection, modification functions. List adapter can put data into ListView to generate bill records with auto-adapt length according to the amount of data.

5. Spinner:

Choose time constraints to select record in a specified period from database.

6. Click to modify:

setOnItemClickListener method is used to listen click and generate AlertDialog. In AlertDialog, if click ok to confirm modification, intent and bundle are used to deliver data to modifyActivity.

7. Long press to delete:

setOnItemLongClickListener method is used to listen record long press and generate AlertDialog. In AlertDialog, if click ok to confirm deletion, the record would be deleted from database.

8. DiagramFragment:

Using MPAndroid library to draw the diagram.

9. Spinner:

Used to select the data of years and months to be displayed and get the corresponding data to fill the graph. Each bar or part of graph is an entry, represent the sum consumption in a period or a type. We sum the data from every single recording in database to get expense in each type and in each period, save it in the HashMap and traverse it to get every single entry to fill the graph.

10. WishlistFragment:

Display the total expense and income, and calculate the balance. Add the wish to the Wishlist, fill the wishes according to the balance,

11. IncomeActivity:

Display the income list and add the single income recording, then calculate the total income. Every single income list can be deleted.

3.3 Source Code

```
//create database
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL(CREATE_ACCOUNT);
    Toast.makeText(mContext,"创建数据库功",Toast.LENGTH_LONG).show();
}

//renew database,  insert attributes

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
{


}
```

```
//generate and control bottom navigation bar
AppBarConfiguration appBarConfiguration = new
AppBarConfiguration.Builder(
        R.id.navigation_home, R.id.navigation_dashboard,
R.id.navigation_notifications)
        .build();
controller = Navigation.findNavController(this, R.id.nav_host_fragment);

NavigationUI.setupActionBarWithNavController(this, controller,
```

```
appBarConfiguration);
NavigationUI.setupWithNavController(navView, controller);
controller.setGraph(R.navigation.mobile_navigation);
controller.navigate(R.id.navigation_dashboard);

}
```

//choose the time of consumption record

```
date.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View arg0) {
        DatePickerDialog datePickerDialog = new
DatePickerDialog(ModifyActivity.this,
                new DatePickerDialog.OnDateSetListener() {
                    @Override
                    public void onDateSet(DatePicker view, int year, int
month, int dayOfMonth) {
                        time.year = year;
                        time.month = month+1;
                        time.day = dayOfMonth;
                        setTime();
                    }
                },
                time.year, time.month-1, time.day);
        datePickerDialog.show();
    }
});
```

// Inflate the menu; this adds items to the aaction bar if it is present

```
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.init_memu, menu);
    return true; }
```

//TODO Auto-generated method stub

```
super.onActivityCreated(savedInstanceState);

database = new Database(getActivity(), "account", null, 4);
db = database.getReadableDatabase();
all=0;

Years = getResources().getStringArray(R.array.years_array);
Months = getResources().getStringArray(R.array.months_array);
ySpinner = (Spinner) getActivity().findViewById(R.id.spinnerYear);
mSpinner = (Spinner) getActivity().findViewById(R.id.spinnerMonth);
```

//search data

```
button1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (v == button1) {
            show_data();
        }
    }
});
```

//clear all record

```
public void Clear(View v)
{
    new AlertDialog.Builder(this)
            .setIcon(android.R.drawable.ic_dialog_alert)
            .setTitle("Are you sure?")
            .setMessage("Do you want to clear ALL records???")
            .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    db.delete("account",null,null);
                    Toast.makeText(SplashActivity.this, "Clear All Record
Successfully!", Toast.LENGTH_LONG).show();
                }
            })
            .setNegativeButton("No", null)
            .show();
}
```

//Long press to delete record

```
    @Override
    public boolean onItemLongClick(AdapterView<?> parent, View view,
final int position, long id) {
        new AlertDialog.Builder(getActivity())
                .setIcon(android.R.drawable.ic_dialog_alert)
                .setTitle("Are you sure?")
                .setMessage("Do you want to delete this record?")
                .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which)
{
                        String t[]=Time_to_intList(Time[position]);
                        String D="year=? and "+
                                "month=? and "+
                                "day=? and "+
                                "hour=? and "+
                                "minute=? and "+
                                "second=?";
                        db.delete("account",D,t);
                        Toast.makeText(getActivity(), "Delete
successfully!", Toast.LENGTH_LONG).show();
                        show_data();
                    }
                })
                .setNegativeButton("No", null)
                .show();
        return true;
    }
```

//modify record

```
@Override
public void onItemClick(AdapterView<?> parent, View view,final int
position, long id) {
    new AlertDialog.Builder(getActivity())
            .setIcon(android.R.drawable.ic_dialog_alert)
            .setMessage("Do you want to modify this record?")
```

```
            .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Intent intent = new Intent(getActivity(),
ModifyActivity.class);
                    Bundle bundle = new Bundle();
                    bundle.putString("Time", Time[position]);
                    bundle.putInt("Money",
Integer.parseInt(Money[position]));
                    bundle.putInt("Type", typeNum[position]);
                    bundle.putString("Comments", Comments[position]);
                    intent.putExtras(bundle);

                    startActivity(intent);
                }
            })
            .setNegativeButton("No", null)
            .show();
}
```

Chapter 4: Performance Evaluation

4.1 Functional Test

Table1: Functional Test

Test contents	Check
After the installation is completed, the software can be opened normally.	Yes
Open the app, whether there is loading status progress prompt	No
Speed test	Medium
The app switches to the background, and then returns to the app to check whether it stayed on the previous operation interface	Yes
After unlocking the phone's lock screen, can the app restart normally	No
When the running process is interrupted, open the app again, can the app restart normally	Yes
After killing the app process, open the app again, can the app restart normally	Yes
Local data can be browsed when there is no network	Yes
Whether multiple apps running simultaneously affect normal functions	No
Does switching the network (3G, 4G, wifi) affect normal functions when the App is running	No

4.2 User Experience Test

Table2: User Experience Test

Test contents	check
Is there unclickable button	No
Are there too many branches	No
Whether too much data is loaded at once	No
Is there a design for landscape mode	No
Whether the content follows the label switch	Yes
Whether the content of the input box is consistent with the system function	Yes
Whether the length of the text is limited	No
Whether the information is displayed in Chinese	No
Whether there are sensitive words, keyword detection	No
Whether the icons on interface are overall order and beautiful	Yes
The content is adaptive according to the window size	Yes
Unified operation mode of each control	Yes

4.3 Compatibility Test

Table3: Compatibility Test

Test contents	Check
Compatible with mainstream apps	Yes
App data and usage are correct under various network connections	Yes (3G,4G,WIFI)
Compatibility of different operating systems	Only for Android
Compatibility of different screen resolutions	Yes (HUAWEI Pro30, Samsung A71)
Compatibility of different mobile phone brands	Yes (HUAWEI Pro30, Samsung A71)

Chapter 5: Discussion and Conclusions

5.1 Industry Trends

5.1.1 Focus on accounting requirements and improve core functions

The homogenization of products of various bookkeeping apps is very serious. To maintain competitiveness, we must continue to polish the basic functions around bookkeeping and improve professional and differentiated bookkeeping services. First, we must dig deep into user scenarios and increase financial scenarios to meet the user's accounting needs in different consumption scenarios. Become more convenient and efficient, the third is to make accounting more professional, automate and comprehensively manage the user's accounting consumption data, help users integrate personal financial data, sort out user assets and liabilities, and improve personal financial management.

5.1.2 In-depth accounting scenarios, link to life financial services

As for the accounting tools, with the continuous expansion of the user scale, users have upgraded from a single demand ecosystem to an experiential demand for compound needs. On the basis of having a large number of fan users, we must focus on how to solve user problems and provide deeper value, various bookkeeping and wealth management apps, including various bookkeeping and credit card management tools, have all transformed into comprehensive directions such as payment, financial management, lending, and bill management to meet the growing diversified financial needs of users.

5.1.3 Provide intelligent financial and financial services

Industry insiders said that spreading "one-stop" smart financial services to a wide range of individuals and families will be the future direction of the bookkeeping and wealth management app. On the one hand, the bookkeeping and wealth management APP will provide users with intelligent financial management services, including automatic processing of financial data, cash flow management, and earnings risk preference calculation; on the one hand, it will be based on user data including financial data, credit data, and behavior Data, social data, etc., according to the user's personal financial level and risk tolerance, through accurate big data and intelligent analysis and processing capabilities, intelligently match the most suitable financial services for users.

5.2 Suggest

5.2.1 Accounting function

We can divide Multiple bookkeeping books, which allows users to better distinguish different types of accounts, such as reimbursable parts and non-reimbursable parts, so that subsequent changes or deletion of accounts will be more convenient and can be operated in batches

5.2.2 Other Functions

we want to implement the function of import bills from Ali pay in bulk. Now mobile payment is very convenient, especially Ali pay, if you can directly import data from Ali pay in bulk, you can save the user to manually add it more convenient.

5.2.3 Diversity of subsequent development

With the continuous development of AI artificial intelligence technology, the bookkeeping style will also change dramatically. Previously, "writing", now "points", and later "speaking", and even

"a glance" can keep accounts. The recorded accounts will also enter the field of intelligence as a good database.

The development of bookkeeping tools will also develop in more directions, and future bookkeeping tools will be more differentiated and the profit model will be more diverse:

1. Financial asset steward direction. All assets are managed in one place, keep track of the asset status at any time, and make intelligent investment consulting based on the availability and anti-risk of your assets.

2. Service provider direction. As a basic service, bookkeeping is output to products in various fields. For example, export travel accounting to travel websites, and export decoration accounting to furnishing furniture websites.

3. Data intelligence direction. Combined with powerful data analysis capabilities, we can know when users need what they like, what kind of services are more attractive to users.

Bibliography

[1]Analysis think tank. China's Bookkeeping and Wealth management application Mobile Market Quarterly Monitoring Report Q4 2015[EB/OL].https://www.analysys.cn/article/detail/17579,2016-03-09