*BACHELOR OF SCIENCE DEGREE/DEGREE WITH HONOURS IN DIGITAL TECHNOLOGY
* BSc Final Year Project Report
School of Engineering and Technology
University of Hertfordshire

A VIRTUAL PET

Report by
Qian, Xingyi

Supervisor
Lambert, Alan

Date
March 2018

DECLARATION STATEMENT

I certify that the work submitted is my own and that any material derived or quoted from the published or unpublished work of other persons has been duly acknowledged (ref. UPR AS/C/6.1, Appendix I, Section 2 – Section on cheating and plagiarism)

Student Full Name: QIAN XINGYI

Student Registration Number: 16073537

Signed: ……

Date: March 2018

ABSTRACT

Some people may cannot keep a pet for some reasons unless they really want to. With the development of technology, they can have virtual pets as alternatives of the real pets. The developer made an Android virtual pet application to realize this alternative method.

This report introduces a virtual pet application with the following process: feasibility study, system analysis, the details of the system design as well as the system development. The main modules of this application mainly include advertisement module, login module, register module, feeding module, cleaning module and the online database module. The developer used Java and C# to finish this development, Android Studio and Visual Studio 2010 are used as the integrated development environment.

ACKNOWLEDGEMENTS

At the point of finishing this report, I would like to express my sincere thanks to all those who have helped me a lot in the course of my writing this paper. First of all, I would like to take this opportunity to show my sincere gratitude to my supervisor, Alan, who has given me so much useful advices on my writing and has tried his best to improve my paper. Secondly, I would like to express my gratitude to the teachers from Changzhou Institute of Technology, they have given me a lot of ideal of how to do with this project. Last but not the least, I would like to thank those leaders, teachers and working staff especially those in the School of Engineering and Technology. Without their help, it would be much harder for me to finish my study and this paper.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

Activity - An activity is a class that the developer can operate on which organized the Android application with the user interface.

Android - A mobile operating system developed by Google, based on a modified version of the Linux kernel. It can be run on the smart phones, tablet, smart TV and so on.

Android Studio - The official integrated development environment (IDE) of the Google Android operating system is based on JetBrains' IntelliJ IDEA software and is designed specifically for Android development.

API - An application programming interface (API) is a set of clearly defined communication methods between various software components.

E-R Model - An entity–relationship model (ER) describes interrelated things of interest in a specific domain of knowledge.

HTML5 - A markup language for building the World Wide Web. It is the fifth and current major version of the HTML standard.

IDE - An integrated development environment (IDE) is a software application that provides computer programmers with software development facilities.

Intent - An intent is an abstract description of the operation to perform.

Layout - The layout defines the structure of the user interface in the application.

Microsoft SQL Server Express - A version of Microsoft's SQL Server relational database management system, which can be freely downloaded, distributed and used.

SharedPreferences - A methods of the Android system to save the data into the harddisk of the device in the form of XML.

Task - A task is a series of activities that the user interacts with in the execution of a particular job.

Thread - A thread is the smallest unit of the program execution flow. The Android system allows multithreading programs.

UML - The Unified Modeling Language (UML) is a language in the field of software engineering, which aims to provide a standard method for visual system design.

View - This class is the basic building block for user interface. A View occupies a rectangular area to show it contains.

ViewGroup - A special view that can contain other views.

Web Service - A kind of service which used to let device whatever which platform they are used to communicate with each other through the Internet.

1. INTRODUCTION

1.1 BACKGROUND

Nowadays, more and more people like to keep pets, such as dogs, cats and so on. People keep pets as their friends or even members of their families so that they will not be alone and feel lonely with the company of their pets [1]. What is more, raising a pet can cultivate a person's sense of responsibility, because keeping a pet is just like raising a baby, in which much care and great patience are extremely needed. Moreover, keeping a pet is very educational and meaningful for children, who can gain more through taking care of their pets [2].

However, there are also some reasons why a family cannot keep a pet. To begin with, some people have to work, so they have no time to take care of a pet. In addition, those who live in high-rise flats can't spare any space for a pet to live in. Besides, pets may make his room in a mess [3].

What the developer wants to do is to make an application for those who want to keep pets but are unable to do so. They do not need to keep a real pet. Instead, they can adopt a virtual pet, and they can experience all the trouble and happiness that keeping a real pet may bring to them. As we all know, most people have smart phones as there are 4.93 billion mobile phone users in the world [4], and the function of a mobile phone is powerful enough to run some big games. And it is likely to put a virtual pet inside the users' mobile phone. The users just need to download my application in their spare time. Even when they are at work during a break, they can turn on their mobile phones to see whether their pets are well kept. It will be an easy and convenient way to keep a pet.

1.2 PROJECT AIM & OBJECTIVES

This project is aimed at helping those who cannot keep a pet at home but want to have an experience of keeping pets. To those who are hesitating whether to keep a pet or not, this application will help them experience a real-life feeling of keeping a pet.

What is more, the users can play the game together with their friends, helping each other take care of their pets. In this way they can have some common topics and then promote their friendships.

1.3 FEASIBILITY STUDY

1.3.1 SOFTWARE FEASIBILITY STUDY

As this application was aimed to spend out the fragmented time, to let user can use it at the moment they want to use, the developer decided to make it a mobile application.

Android was chosen to be the platform, because it is sharing 86.1% mobile market of the world [5]. The integrated development environment software can be downloaded for free from the Internet, the software is ready for this project.

1.3.2 HARDWARE FEASIBILITY STUDY

This application will not be a high CPU or a high GPU cost one as it is not a real-time online game. In theory, most of the Android device can run the application. To use some new functions and methods of Android, the developer used the Android API 26 which can support down to API 17. The final application developed by this API can run on the Android platform from 4.2 to 8.0, it means device which Android version is below 4.2 may not run this application. The data from statista.com shows that there are about 6.2% of the Android phone users still using the Android phone which version is under 4.2. It means that the final application can support more than 90% Android devices in the market, the data is shown by the Figure 1-1 [6]. This means that the hardware is suitable for this project.

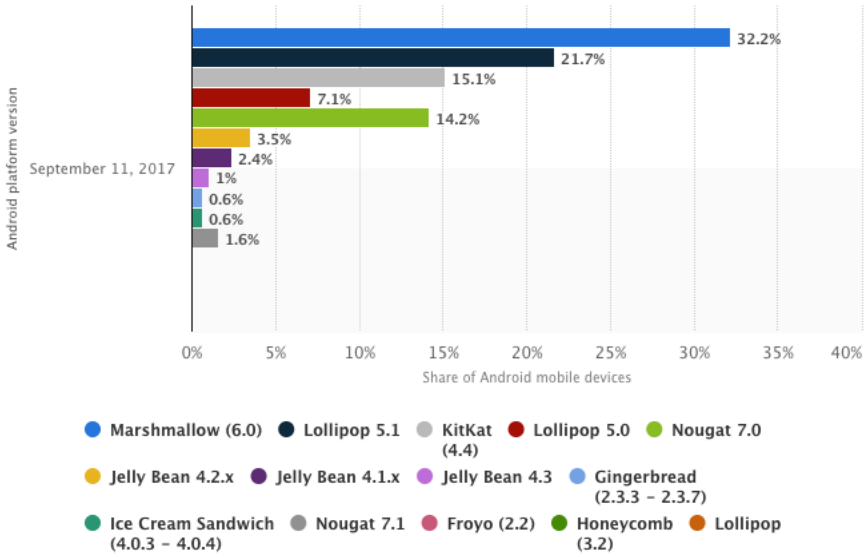

Figure 1-1 Distribution of Android operating systems used by Android phone owners in September 2017, by platform versions

2 SUBJECT REVIEW

2.1 My Talking Tom

Figure 2-1 Screenshot of My Talking Tom

My Talking Tom was a well-known application, user can click on Tom, and it will do some action like dancing, crying and laughing, users must pay money to buy some food so that the cat can do more actions. The flash point of it is that users can talk with this cat, then the cat will repeat what user said in its own voice. This application is still available from Google Play Store [7].

2.2 Travel Frog

Figure 2-2 Screenshot of Travel Frog

Travel Frog was the most famous virtual pet game in 2017, this game is a very simple one, users just need to prepare food for their frogs, and see their pets do something indoor like eating reading and sleeping. Most of the time, their frogs are travelling around the world with some photos sent to the users. This game was the No.1 free game of downloading in Apple App Store [8].

2.3 QQ Pets

Figure 2-3 Screenshot of QQ Pets

This game does not have an individual application, it is one of the module of a social application which named Tencent QQ. This module is written by HTML5. What surprised the developer is that the users can interact with each other, every user is not isolated, they can share foods, help friends do some cleaning and even steal candy which is used as money in this game from their friends. What is more, users can dress up their pets as well. As this game is based on QQ, it now has more than three hundred million users [9].

3 SYSTEM ANALYSIS

3.1 Requirements Analysis

The developer used Tencent Online Questionnaire to publish a research online, one week later, the developer collected the data, and found the requirement of a virtual pet application [10].

The requirement of this kind of application is shown as the Figure 3-1. It shows that about two thirds of the respondents want to try virtual pets. It means this kind of application is accepted by majority people.

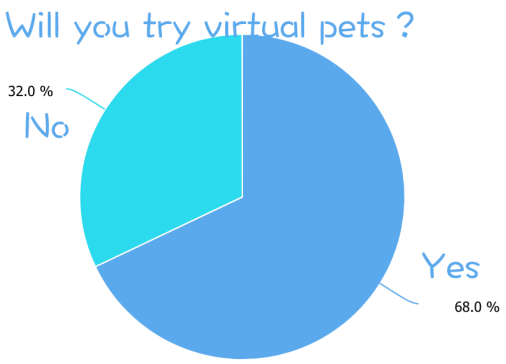

Figure 3-1 The percentage of the requirement of virtual pets

The respondents' preference of the platform is shown by the Figure 3-2. It shows that almost all respondents prefer mobile virtual pets.

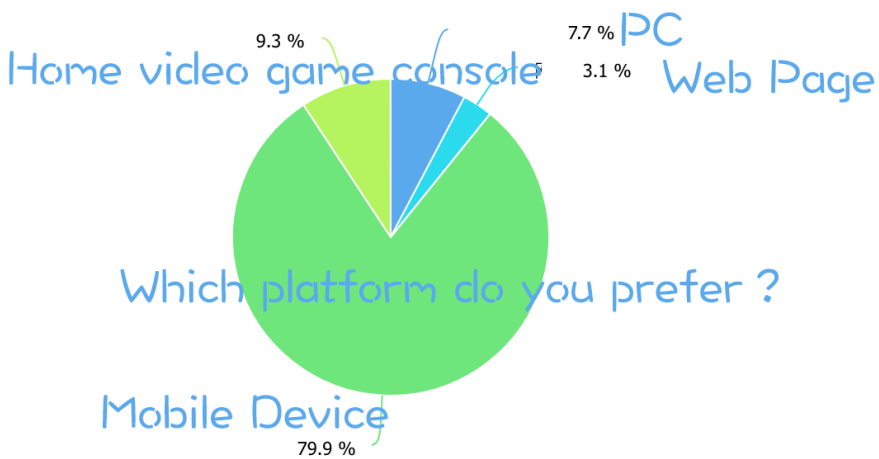

Figure 3-2 The percentage of which platform respondents prefer

Before doing this research, the developer decided to do a virtual cat. Then the topic changed into a virtual dog after doing this research, as more than 70% respondents want dogs shown by Figure 3-3.

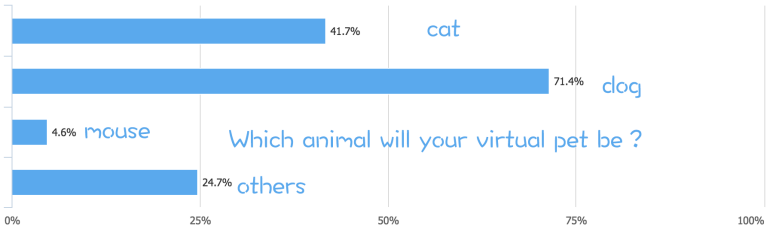

Figure 3-3 The percentage of which animal respondents prefer

The developer chosen feeding, cleaning and chatting as the main functions of the application as they accepted by most respondents shown by Figure 3-4.

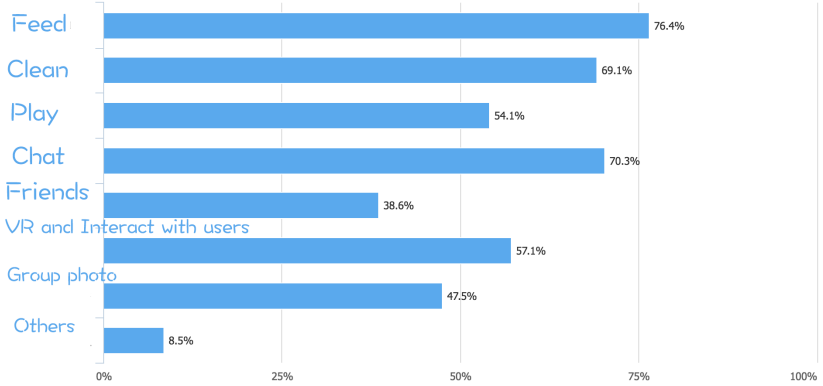

Figure 3-4 The percentage of which function respondents prefer

3.2 Data Flow Analysis

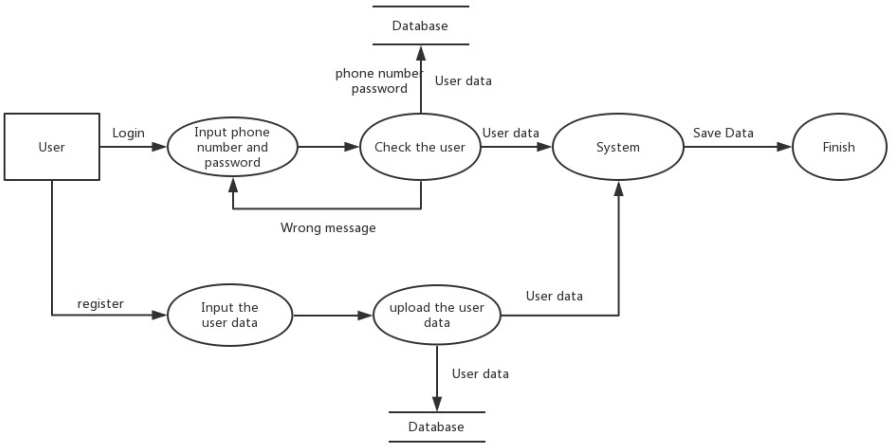

Figure 3-5 Data Flow Diagram

3.3 Scheme Design

As Android is the most popular mobile system of the world, and it is now sharing 86.1% market of the world. With a huge amount of user, it helps more and more developers choose this platform. What is more, developers can use this platform all for free. So, it will be the most suitable platform to do this project. The IDE of this project the developer used is Android Studio, the official IDE produced by Google, so it was chosen to be the software to do the development [11].

It is no doubt that Java is the programming language, as it is the official language chosen by Google who is the producer of Android [12].

The developer decided to make this game online, as people tend to buy new phone much more frequently than before [13]. With an online server store all the users' data, users do not need to restart a new game, just to log in their account, all the data will be downloaded from the cloud, so they can continue their games.

Alibaba is an outstanding online server producer, it now has a student program to help students get a server at the price of only 1.99 pounds monthly [14]. So, the developer decided to use a server produced by Alibaba to store users' data. As the operating system of the server is Windows Server 2012 R2, the online service is better to use those produced by Microsoft. SQL Server 2016 Express was chosen as the database and .NET Web Service was used to translate data between database and the users' devices [15].

4 SYSTEM DESIGN

4.1 System Flow Design

All the data need to be saved online because this system is decided to be an online one. By following the traditional way, the developer used user name and password to verify user rights. This system also contains an interface for new users to create accounts, figure 4-1 shows the logic algorithm of verifying users' login process.

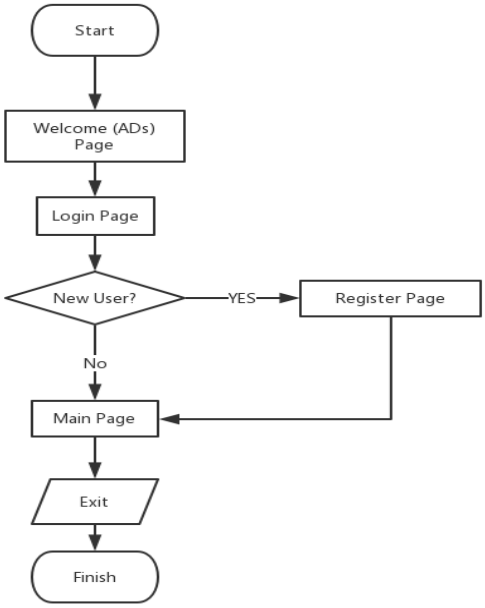

Figure 4-1 Flow Chart

4.2 System Framework Design

Feeding, cleaning and chatting those functions were chosen as the main modules of this system by the developer. To make the game online, login module and register module are also need to be added.

What the developer wants to do is to make a game which can be put into the market in the future. Nowadays, most application are free to be downloaded, and the developers earn from the advertisement [16]. Following this mind, the developer designed an advertisement module to create income. The game is free to download, this game will make money through advertisement fees from the investors.

Advertisement module, register and login modules together with the feeding, cleaning and chatting modules make the outline of this virtual pet application. The system module diagram of this application is shown by figure 4-2.

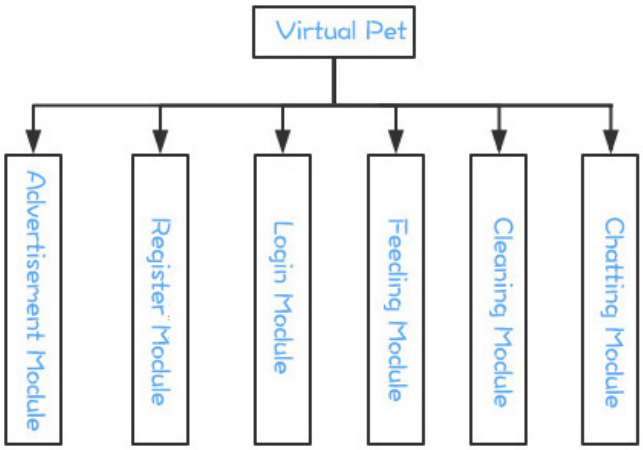

Figure 4-2 System Module Diagram

4.3 Data Structure Design

This system is mainly about the relationship among users, pets and foods. In this java program, three classes user, pet and food are created. In this system, the developer designed that each user can only keep one pet to make the relationships of this system easy to design. The link between one pet and its owner is the ID of the user, which is called uID by the developer. Foods are created to fee pets, so the developer created a property which is called hungPoint, it is the only way to measure if the pet is hungry or not, and the degree of hunger, both the pets and the foods own this property. The level of the pet decides what kind of food it can have, the pets' level is also important for the foods to keep to judge if the pet can use it. By following the above laws, the developer drawn three unified modelling language (UML) diagram to show the structures of the three classes. The structures of them are shown below.

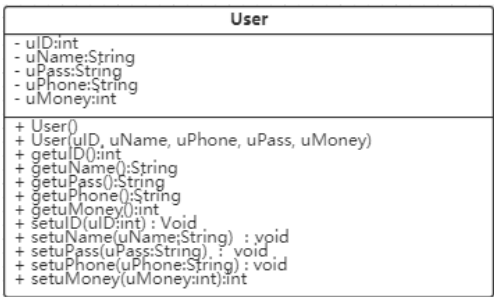

Figure 4-3 UML diagram of User.class

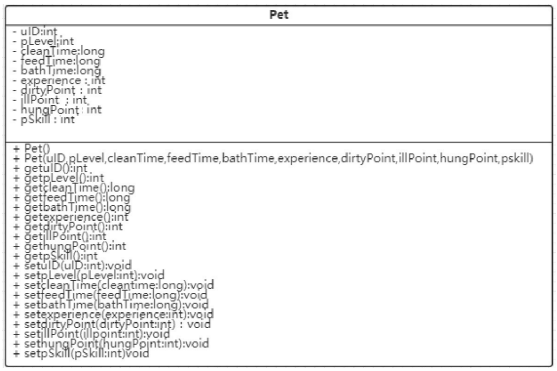

Figure 4-4 UML diagram of Pet.class

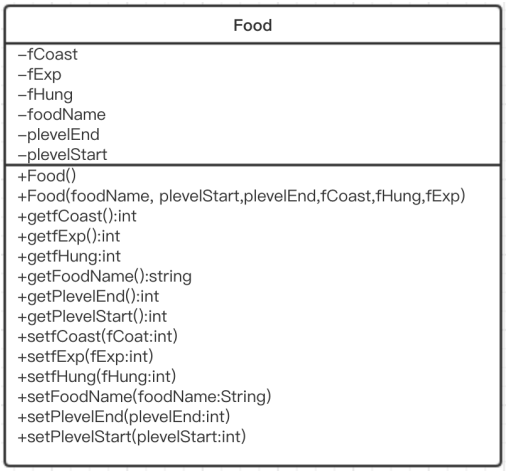

Figure 4-5 UML diagram of Food.class

4.4 Database Design

The following three table is supported to be created in the online database.
User (User's ID, User's Name, User's Password, User's Phone Number, User's Balance)

Pet (Pet's ID, User's ID, Pet's Name, Pet's Level, Pet's Fed Time, Pet's Bath Time, Pet's Cleaned Time, Pet's Huger Point, Pet's Dirty Point, Pet's Ill Point, Pet's Experience)

Food (Food's ID, Food's Name, Food's Coast, Food's Experience, Food's Hunger Point, Minimum Level to Use the Food, Maximum Level to Use the Food)

The database will follow the data structures of the three classes, but while the developer was doing the developing, he found it is not very convenient to get data from two tables, so the developer moved some properties from the users' table to the pets' table, this make the application runs far more efficiently. And the pets' ID are never used by the system but it is the primary key of the pets' table, so the developer deleted the property from the class but kept it in the database. During the developing, the developer found that the foods' data do not need to be online, as it is not a private type of data and it will be easy to manage if it is saved in the local field, so the developer deleted the whole foods' table and create a method to create those foods in the program.

The final database design is shown as the following two figures:

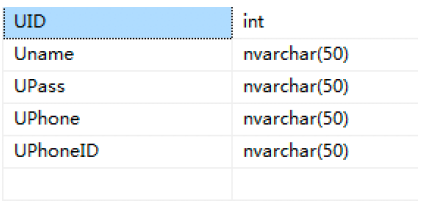

Figure 4-6 Users' Table

PID	int
UID	int
PName	nvarchar(50)
PLevel	int
PHungerPoint	int
PDirtyPoint	int
PBalance	int
PExp	int
PCleanTime	numeric(18, 0)
PFeedTime	numeric(18, 0)
PBathTime	numeric(18, 0)
PCollectTime	numeric(18, 0)

Figure 4-7 Pets' Table

4.5 Entity Relationship Design

The entity-relationship (E-R) diagram clearly shows the relationship among the three classes in this system.

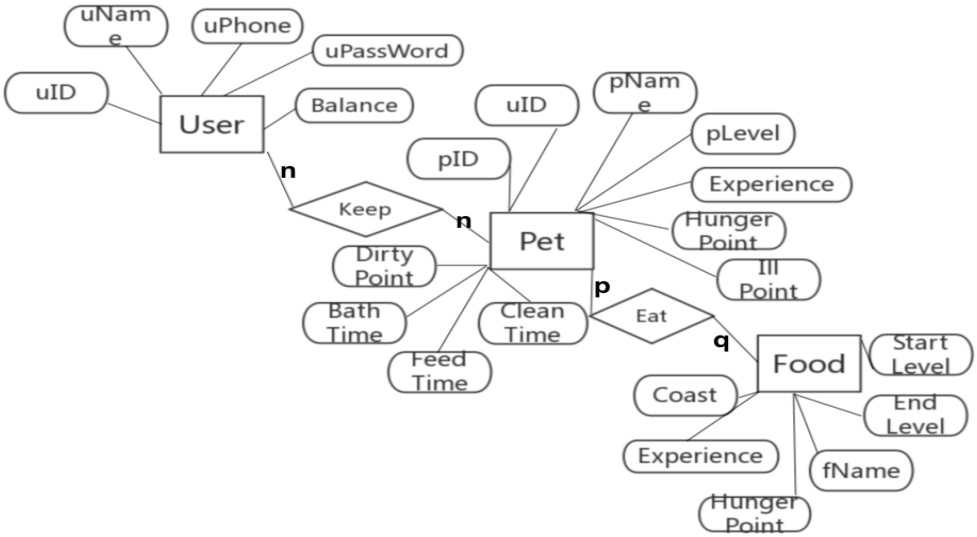

Figure 4-8 E-R Diagram

4.6 User Interface Design

4.6.1 Theme Colours Design

Different colours have different meanings, it is very important to choose some colours to parallel the topic of this application [17]. This application is aimed to help people, and the main users will be the children. So, the developer must choose some colour can convey the feeling of happy, optimistic, friendly and warm, it also can attract children. Some meanings of the colours from Wix Blog are shown by the figure 4-9.

Figure 4-9 Colour psychology

The developer chosen yellow and orange as the main theme colours. Yellow represents happiness and optimism, it can cheer the users up. Orange is very suitable to make this application looks much friendlier, it can attract not only children but also adults. The developer decided to make all the buttons into blue, because blue makes users trust in the buttons, it can help users feel comfortable and reliable.

4.6.2 Character Design

The online research shows that, most users want to keep a dog, so the developer focus on finding a dog which is suitable for this application. After finishing the subject review, the developer found that the three popular virtual pets application all used personified animal roles, this issue can help applications look friendly to users. What is more, the chosen character need to follow the theme colours to make this application organized.

Achai is a set of emoticons which shows a lot of emotions of an orange and white dog who can active like human beings [18]. This set of emoticons is made by an illustrator whose nickname is Tony on Zcool.com. Acahi is a very cute and optimistic dog, all his movements are very exaggerated and funny, that is what the application needs.

Figure 4-10 Achai

4.6.3 Icon Design

The Human Interface Guidelines of Apple Inc. noticed that “Every app needs a beautiful and memorable icon that attract in the App Store and stands out on the Home screen. Your icon is the first opportunity to communicate, at a glance, your app’s purpose.” [19] To let the user understand this application’s purpose, the developer used a picture of Achai as the main part of this icon.

Developer made the icon into a shape of square with corners round, this type of design is very popular and it was leaded by Apple Inc. [20]. This design also referenced the icons of Twitter, Facebook and Snapchat; developer used a signal colour as the background colour to make the icon looks light and simply [21]. The colour yellow which represents happiness and optimism was used to be the background colour. The final icon is shown by figure 4-11.

Figure 4-11 Icon

5 SYSTEM DEVELOPMENT

5.1 Advertisement Module

5.1.1 User Interface

The user interface of advertisement module is shown by figure 5-1. This page contains a skip button and an advertisement image. The advertisement image can be changed in the future by updating to earn money from the investors.

Figure 5-1 Advertisement module

5.1.2 Functions Introduction and Realization

The functions of this module were realized by WelcomeActivity.class. This module contains two flash points: time-lapse page transition and a button clicked event.

The time-lapse forwarding was provided by a handler thread, the developer added an intent event this handler thread to make this thread can move forward into the next page. A function named sendEmptyMessageDelayed () is used to delay the action. After user opened this page, the next page will be shown 3 seconds later automatically without any operation.

A skip button is bound with an onClicked () event which contains a same intent event in the handler thread [22]. This function was made to help the user skip the advertisement if they do not want to watch it.

```
    new Handler(new Handler.Callback() {
        @Override
        public boolean handleMessage(Message arg0) {
            if(isOpened==false)
            {startActivity(new Intent(getApplicationContext(),LoginActivity.class));
            finish();}
            return false;
        }
    }).sendEmptyMessageDelayed( what: 0, delayMillis: 3000);//延时3秒

}
public void onClicked(View v){
    switch(v.getId()){
    case R.id.editBtn:
        Intent i1=new Intent();
        ComponentName component=new ComponentName(getApplicationContext(), LoginActivity.class);
        i1.setComponent(component);
        startActivity(i1);
        isOpened=true;
        finish();
        break;
    }

}
```

Figure 5-2 Handler and onClicked () event

5.1.3 Debugging

Found in the initial debugging, after developer pressed the skip button, the handler thread was still executed, then a new login page popped up, the developer found that there are two login pages after 3 seconds.

In order to control the occurrence of this phenomenon, the developer decided to use a Boolean value to control whether to perform the intent event. The study found that sendEmptyMessageDelayed () method provides the delay effect is to delay the implementation of the thread after creating the thread. So, after the skip button is pressed, the implementation of the Boolean value assignment to store whether the page transition has occurred [23]. The intent statement is wrapped by the if statements to determine the conditions for the Boolean value is false, if false, it means the skip button has not been pressed, 3 seconds after the implementation of the page transition. If the Boolean value is true, it means the intent event has been executed, no need to be executed again.

After the modification, the bug of the page transition function can be solved.

5.2 Login Module

5.2.1 User Interface

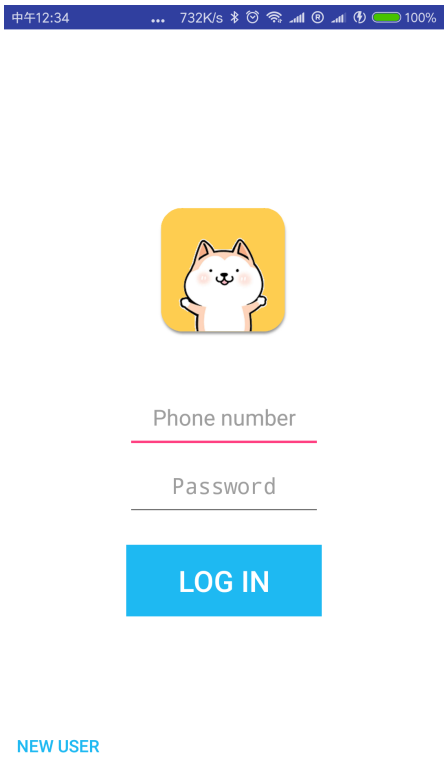

Figure 5-3 Login module

The user interface of login module is shown by figure 5-3. It contains an application icon, two input boxes, a login button and a button link to the register module.

After users enter their phone numbers and passwords, they can log into the system. However, it is highly possible that the user will make a mistake in entering the correct data. Therefore, the validation logic should immediately prompt the user with an appropriate message. While the phone number or the password is empty, the system will display a line in red to remind the users. The line will also appear if the system cannot find users' data online to remind the users to double check their inputs. The login button and the input boxes cannot be accessed while the system is comparing the users' inputs with the online database, and a dynamic progress bar will appear to give the users the message that the system is not crashed. Those feedbacks are shown by the figure 5-4.

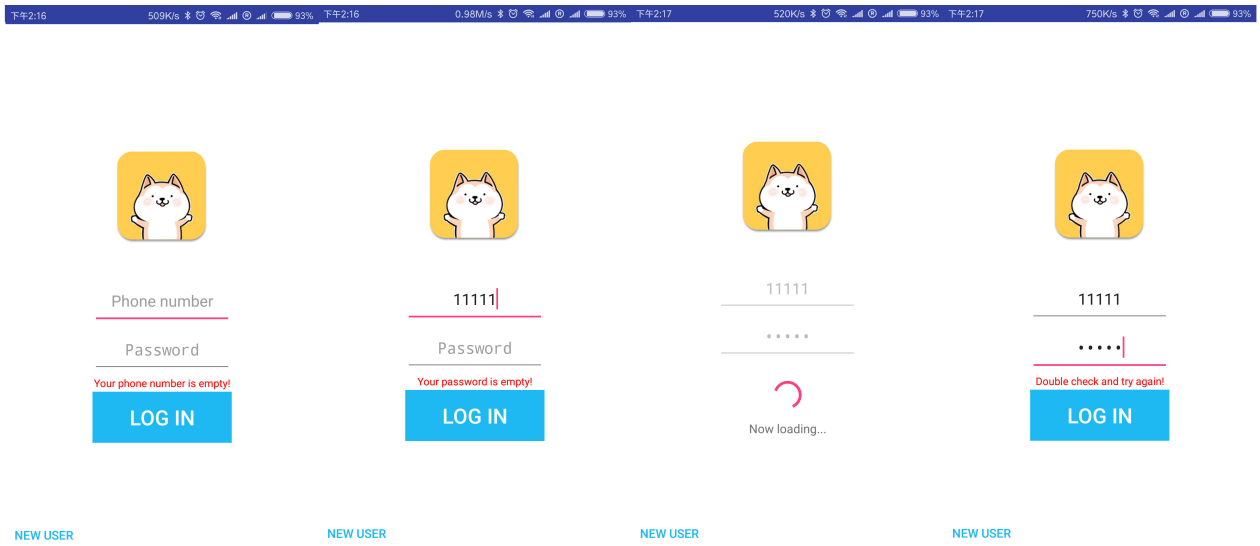

Figure 5-4 Feedbacks of operations

5.2.2 Functions Introduction

These functions of this module were realized by LoginActivity.class. The process of the module is divided into four steps: The system checked the user's entering mistake first, and then compared the input with online database. After these, system downloaded the user's data from the online database. Finally, the downloaded data was sent to the core module of this application.

5.2.3 Check User's Mistakes

To check the user's mistakes, "if" statement is need to be used, use "else if" to reduce the number of system judgements. While the system is connecting the Internet to check the user's data, this page cannot be still, if so, the user may think the system is crashed. To avoid this situation, the dynamic progress bar was added into this page to mind the user this system is still working and they have to wait.

```
case R.id.login_btn:
    phone = phoneEd.getText().toString();
    pass = passEd.getText().toString();
    str = "";
    phone = phoneEd.getText().toString();
    pass = passEd.getText().toString();
    if (phone.equals("")) {
        errorText.setText("Your phone number is empty!");
    } else if (pass.equals("")) {
        errorText.setText("Your password is empty!");
    } else {
        errorText.setText("");
        waitlayout.setVisibility(View.VISIBLE);
        logBtn.setVisibility(View.GONE);
        passEd.setEnabled(false);
        phoneEd.setEnabled(false);
        MyAsyncTask myAsyncTask = new MyAsyncTask();
        myAsyncTask.execute(5000);
    }
    break;
```

Figure 5-5 Check user's mistake

5.2.4 Get Data from Web Server

To compare the user entered data with online database, Web Service was used to connect SQL Server with the Android application, 5.6 will cover the details of Web Server, this part will focus on the Android side. To use the Web Service, the permission android.permission.INTERNET was needed to be added into the AndroidManifest.xml file [24]. In this module, checking user used the FindUserByPhone method of Web Service, downloading user's data used FindPet method. This process was in a thread, as the Internet connecting need much more time than the UI thread, so a new thread must be created to run this process so that the UI thread cannot be blocked. This method is used to make the whole system run efficiently.

As the developer used the Web Service the online server will return a XML string while the client asks for response. As the W3school described "XML is a software- and hardware-independent tool for storing and transporting data." This kind of documents "does not do anything", and users can define their own tags, it can translate data between two different platforms, this is why it can translate data from SQL Server to Android. "It simplifies data sharing, it simplifies data transport, it simplifies platform changes, it simplifies data availability." [25]

In the XML from the server, each tag represents a property, the value of it represents the value of this property. The form of the response from FindUserByPhone method is shown as the figure 5-6.

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
  <soap:Body>
    <FindUserByPhoneResponse xmlns="http://tempuri.org/">
      <FindUserByPhoneResult>
        <User>
          <uID>int</uID>
          <uname>string</uname>
          <uPass>string</uPass>
          <uPhone>string</uPhone>
          <phoneID>string</phoneID>
        </User>
        <User>
          <uID>int</uID>
          <uname>string</uname>
          <uPass>string</uPass>
          <uPhone>string</uPhone>
          <phoneID>string</phoneID>
        </User>
      </FindUserByPhoneResult>
    </FindUserByPhoneResponse>
  </soap:Body>
</soap:Envelope>
```

Figure 5-6 Response of FindUserByPhone method

The useful tags from FindUserByPhone method are shown as below:

<User>

<uID></uID>

<uName></uName>

<uPhone></uPhone>

<uPass></uPass>

</User>

The form of the response from FindPet method is shown as the figure 5-7.

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/en
  <soap:Body>
    <FindPetResponse xmlns="http://tempuri.org/">
      <FindPetResult>
        <Pet>
          <uID>int</uID>
          <pName>string</pName>
          <pLevel>int</pLevel>
          <pDirtyPoint>int</pDirtyPoint>
          <pHungerPoint>int</pHungerPoint>
          <balance>int</balance>
          <exp>int</exp>
          <cleanTime>long</cleanTime>
          <feedTime>long</feedTime>
          <bathTime>long</bathTime>
          <collectTime>long</collectTime>
        </Pet>
        <Pet>
          <uID>int</uID>
          <pName>string</pName>
          <pLevel>int</pLevel>
          <pDirtyPoint>int</pDirtyPoint>
          <pHungerPoint>int</pHungerPoint>
          <balance>int</balance>
          <exp>int</exp>
          <cleanTime>long</cleanTime>
          <feedTime>long</feedTime>
          <bathTime>long</bathTime>
          <collectTime>long</collectTime>
        </Pet>
      </FindPetResult>
    </FindPetResponse>
  </soap:Body>
</soap:Envelope>
```

Figure 5-7 Response of FindPet method

The useful tags from FindPet method are shown as below:

<Pet>

<uID></uID>

<pName></pName>

<pLevel></pLevel>

<pDirtyPoint></pDirtyPoint>

<pHungerPoint></pHungerPoint>

<balance></balance>

<exp></exp>

<cleanTime></cleanTime>

<feedTime></feedTime>

<bathTime> </bathTime>

<collectTime> </collectTime>

</Pet>

The developer used the getInputStream() method to get the response, then used the data reading method to put the stream into a string [26]. To get the password, the developer used the split () method from String.class to split the string into string array with “uPass>” as the node. After splitting, the array contains two strings, and the second string contains the

password, that is what the system need. The second string is in the form of “User’s Password + </uPass>+ some other messages”. Split the second string into two strings with “</” as the node in the same way, the first string is the password. The developer got other data from the XML data by the same way of getting password.

To make the thread can using the permission of using the Internet the following two statements are needed to be used in the thread:
policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy); [27]

As Web Server uses http protocol to get requirement from the clients, the developer sent a http URL to the server to get the response. When the system gets the response, the first thing is to check if this required is successful or not. The server will reply a response code to the client to show if it is successful, if the code is 200, it means the requirement is successful, otherwise, it means failed. Then it is time to get the response data, the data is in the form of stream, the developer used buffer to read the data, each time the buffer will get 512 bytes from the stream and put it into a string byte by byte. HttpURLConnection is a class provided by Android to establish a connection between Android and server. This class uses connect () method to establish the connection and saves response code and response data inside. The getResponseCode () method is used to get the code, and getInputStream () method is used to get the data in the form of stream. The core codes are shown by the figure 5-8.

```
public void checkTheUser() {
    Thread th = new Thread() {
        @Override
        public void run() {

            HttpURLConnection conn;
            try {
                StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
                StrictMode.setThreadPolicy(policy);
                String url1 = "http://www.moecity.cn/Service1.asmx/FindUserByPhone?searchstring=" + phone;
                URL u = new URL(url1);
                conn = (HttpURLConnection) u.openConnection();
                conn.connect();
                int recode = conn.getResponseCode();
                if (recode == 200) {
                    InputStream in = conn.getInputStream();
                    str = "";
                    int n;
                    byte[] buffer = new byte[512];
                    while ((n = in.read(buffer)) > 0) {
                        str += new String(buffer,  offset: 0, n);
                    }
                    try {
                        strs = str.split( regex: "uPass>");
                        strs = strs[1].split( regex: "</");
                        passGet = strs[0];
                        s.setuPass(pass);

                        strs = str.split( regex: "uID>");
                        strs = strs[1].split( regex: "</");
                        id = strs[0];
                        s.setuID(Integer.parseInt(id));

                        strs = str.split( regex: "uname>");
                        strs = strs[1].split( regex: "</");
                        name = strs[0];
                        s.setuName(name);

                        s.setuPhone(phone);
                        s.setPhoneID(phoneID);
```

Figure 5-8 Core codes of getting data from server

5.2.5 Local Data Storage

After got the data from the server, the user’s data is in the RAM, it will be deleted while this activity finished. A method must be used to save the data in local. The developer decided to

use SQLite at first, as it is a local database of Android. After reading the guide book of Android, the developer decided to use Shared Preferences to store data. As the guide book said, “If you have a relatively small collection of key-values that you'd like to save, you should use the Shared Preferences APIs.”[28] The data need to store in local is from a single user, so the data cannot be too much, while SQLite is used to store a whole database, it is waste of disk space to do so. In summary, Shared Preferences is the best way to save the local data.

To use the Shared Preferences, SharedPreferences objects must be declared in the field, shown as the figure 5-9. And then in the onCreate() method, the developer must name the object by the way shown as figure 5-10 to use it. To do operation on the SharedPreferences object, an Editor object should be created to do this, the developer must put the data into it and name the tag of each property, then commit it to the editor, figure 5-11 is the way developer save the user’s data into SharedPreferences.

```
private SharedPreferences userData, petData;
```

Figure 5-9 SharedPreferences objects in field

```
userData = getSharedPreferences( name: "myUser", MODE_PRIVATE);
petData = getSharedPreferences( name: "myPet", MODE_PRIVATE);
```

Figure 5-10 Naming the SharedPreferences object

```
Editor editor = userData.edit();
editor.putString("userName", name);
editor.putString("phone", phone);
editor.putBoolean("isIn", true);
editor.putInt("userID", s.getuID());
editor.putBoolean("isNew", false);
editor.putString("phoneID", phoneID);
editor.commit();
Editor editor2 = petData.edit();
editor2.putInt("uID", p.getuID());
editor2.putString("pName", p.getpName());
editor2.putInt("pLevel", p.getpLevel());
editor2.putInt("dirtyPoint", p.getDirtyPoint());
editor2.putInt("hungPoint", p.getHungPoint());
editor2.putInt("balance", balance);
editor2.putInt("experience", p.getExperience());
editor2.putLong("bathTime", p.getBathTime());
editor2.putLong("cleanTime", p.getCleanTime());
editor2.putLong("feedTime", p.getFeedTime());
editor2.putLong("collectTime", collectTime);
editor2.commit();
```

Figure 5-11 Putting data into SharedPreferences

To use the data saved in the SharedPreferences, the user just need to use the get method of Editor class, the way of get user’s name is shown as below:

```
userName = userData.getString("userName", "NoName");
```

The developer must give the data a default value, in case there is no data inside the SharedPreferences. This makes the system safe.

The SharedPreferences files is stored in the Data folder, it is a kind of xml file shown as figure 5-12.

```xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="phone">18952669536</string>
    <int name="userID" value="1" />
    <string name="phoneID">9c3cae09550d9b19</string>
    <boolean name="isNew" value="false" />
    <string name="userName">NoName</string>
    <boolean name="isIn" value="false" />
</map>
```

Figure 5-12 SharedPreferences file myUser.xml

5.2.6 Asynchronous Task

In this module, the data saving process must do after the data have been downloaded from the server. But the downloading process need to spend a lot of time that may cause the UI thread stop, this is a kind of bad user experience.

Asynchronous task is a kind of task to separate two or more tasks. Developers can use this kind of task by extending the class AsyncTask, this class can put one task in the background without any effect on the UI thread by put the task into doInBackground () method, it also can start a task after the background task finished by put the task into onPostExecute () method [29]. The developer put the thread of getting data from server in the background, so that the UI thread would not looks stopped. The process of saving data into the SharedPreferences need to do after the data is downloaded, so this part was put into the onPostExecute () method by developer. With the data saved, this activity can be finished by system and start the next activity. This asynchronous task will start after the user clicks on the login button, and all the input boxes and buttons unavailable, a progress bar appears to show the system is still working.

```
errorText.setText("");
waitlayout.setVisibility(View.VISIBLE);
logBtn.setVisibility(View.GONE);
passEd.setEnabled(false);
phoneEd.setEnabled(false);
MyAsyncTask myAsyncTask = new MyAsyncTask();
myAsyncTask.execute(5000);
```

Figure 5-13 When to start the Asynchronous Task

```
private class MyAsyncTask extends AsyncTask {

    @Override
    protected Object doInBackground(Object[] objects) {
        checkTheUser();
        return true;
    }

    @Override
    protected void onPostExecute(Object o) {
        super.onPostExecute(o);
        if (MD5.generatePassword(pass).equals(passGet)) {
            Editor editor = userData.edit();
            editor.putString("userName", name);
            editor.putString("phone", phone);
            editor.putBoolean("isIn", true);
            editor.putInt("userID", s.getuID());
            editor.putBoolean("isNew", false);
            editor.putString("phoneID", phoneID);
            editor.commit();
            Editor editor2 = petData.edit();
            editor2.putInt("uID", p.getuID());
            editor2.putString("pName", p.getpName());
            editor2.putInt("pLevel", p.getpLevel());
            editor2.putInt("dirtyPoint", p.getDirtyPoint());
            editor2.putInt("hungPoint", p.getHungPoint());
            editor2.putInt("balance", balance);
            editor2.putInt("experience", p.getExperience());
            editor2.putLong("bathTime", p.getBathTime());
            editor2.putLong("cleanTime", p.getCleanTime());
            editor2.putLong("feedTime", p.getFeedTime());
            editor2.putLong("collectTime", collectTime);
            editor2.commit();
            Toast.makeText(getBaseContext(), text: "Welcome, " + name,
                    Toast.LENGTH_LONG).show();

            Intent intent = new Intent( packageContext: LoginActivity.this, CoreActivity.class);
            intent.putExtra( name: "newLogin",  value: true);
            startActivity(intent);
            finish();
        }
    }
}
```

Figure 5-14 The Asynchronous Task

5.2.7 Debugging

After the task of creating a thread to get the data from server has been started in background, the user found the data may not be downloaded from time to time. After reading the guide book, the developer found that after the thread be started, the AsyncTask will do the onPostExecute () method, because the task to do in background is to start a thread to start the thread of getting data from the server. To get the data, the user removed the thread from the checkUser () method. The figure 5-15 shows the debugging codes.

```
public void checkTheUser() {
    HttpURLConnection conn;
    try {
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        String url1 = "http://www.moecity.cn/Service1.asmx/FindUserByPhone?searchstring=" + phone;
        URL u = new URL(url1);
        conn = (HttpURLConnection) u.openConnection();
        conn.connect();
        int recode = conn.getResponseCode();
        if (recode == 200) {
            InputStream in = conn.getInputStream();
            str = "";
            int n;
```

Figure 5-15 Remove the thread from the checkUser method

5.3 Register Module

5.3.1 User Interface

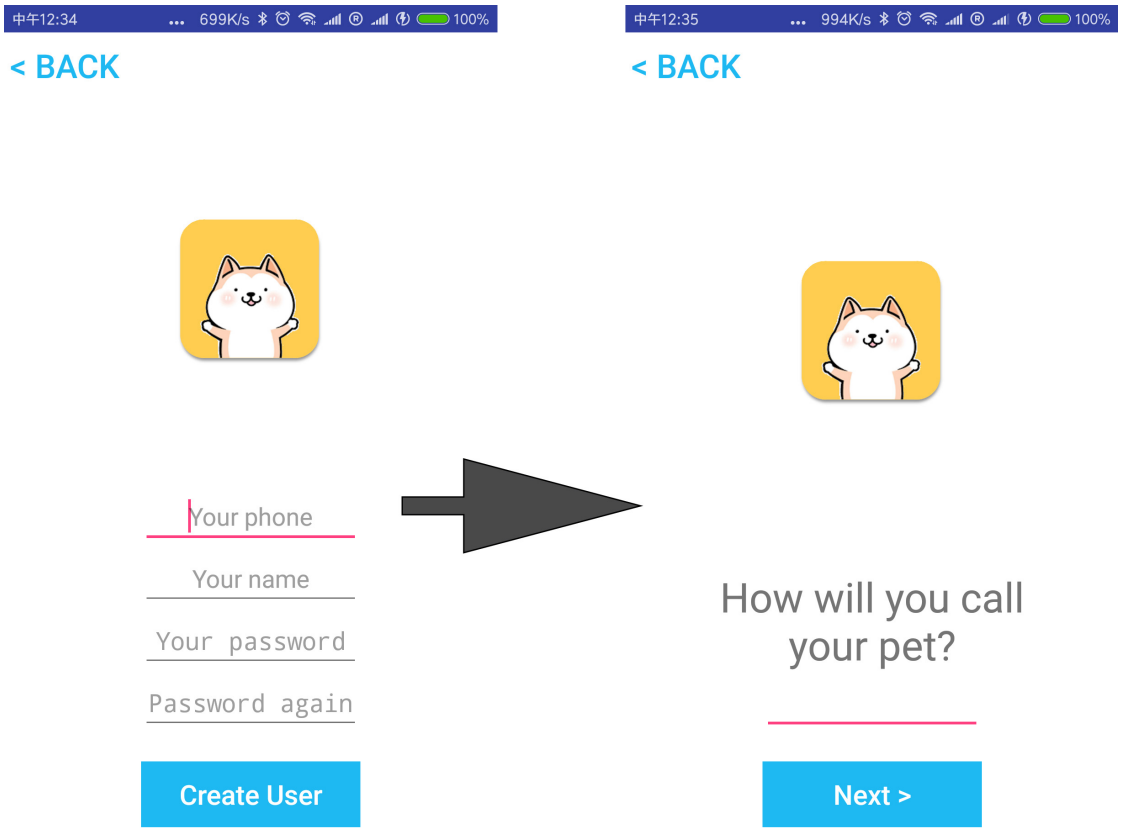

Figure 5-16 Register module

The user interface of register module is shown by figure 5-16. This module contains two pages (or activity in Android). The first page is to record the users' details and the second page is to let users name their pets.

5.3.2 Functions Introduction and Realization

This module is using the SharedPreferences and updating the online database by using the Web Server. The functions of this module were realized by NewUserActivity.class and NewPetActivity.class.

After user entered the data, the system will use the method FindUserByPhone () from Web Service to check if this phone number has been used before. If the server's response is not empty, it means the phone number has been used before, otherwise, a toast will be made to remind user to change another phone number. Checking the response if contains "<uPhone>" is a way to check the phone number. Then the system will check the password, if the password entered first is the same as the second one it means the user has confirmed the

password, otherwise, a toast will be made to remind user to check the password. After all the data has been confirmed, the system will use the method InsertUserDetails () from Web Server to insert the new user.

To make the UI looks not too crowd, a new page was created to name the pet. As the naming pet part is not in the same activity with the insert user part, the data cannot share with each other, so the SharedPreferences is used to get the user details in this page. FindUserByPhone () was used to get the user's ID, and then use InsertPetDetails() method provided by Web Service to create a pet which linked with the user by the user's ID.

5.3.3 Debugging

In this module, the password was uploaded into the database without any security method. This made a risk that information leakiness may happen. Java provide a kind of encryption algorithm which is named MD5, this kind of algorithm cannot be decrypted. It means if this application uses this algorithm, as the encrypted string saved online, even the administrator of database cannot know the user's password, only the users themselves know their passwords.

The developer used an open source project of MD5 from GitHub to encrypt the password user has entered. The decrypted password is a string with 32 characters, it is too long to put into the URL with other information. The developer came up with the idea that set the password as "0" in server at first, then use the UpdateUserPass () method to set the password into the encrypted one. The URL will in the suitable size.

As the passwords have been encrypted, the statements that confirmed the user's password with the password in server need to encrypt the user's entered password first then do the confirming work.

5.4 Core Modules

5.4.1 User Interface

Figure 5-17 Main Page

This page contains the feeding, cleaning and playing these three modules. In the top of the page are the information of the user and the user's pet, the user's name is under an icon of a user. Then it follows the pet's name, level and experience; the balance and a collect button are in the end.

The character stands in the middle of the page, the user can drag it to and place in the page they want. A menu will appear around the character when it was clicked. This menu contains three foods the pet can use at its level and a progress bar showing the hunger point of it.

In the bottom of this page, a logout button and an information button stand on two sides respectively.

5.4.2 Functions Introduction

These functions of this module were realized by CoreActivity.class. This page realized the feeding, cleaning and playing these three functions. And the developer created a thread always runs in the background to monitor the pet's and the user's data and put those data into

the UI thread. This activity also provides the algorithm of feeding, how the pet will be hungry and how the pet will ill.

5.4.3 Data Methods

The developer created a new class DataMethods () to make the rules of how to keep pets. As we know, when the users in the different levels, an operation on the pets will have different effects. So, the developer set the roles of bathing and cleaning, the roles are shown by table 5-1 and table 5-2

Table 5-1 Bathing rule

Level	Dirty Point	Experience	Level	Dirty Point	Experience
1	1	10	**16**	5	20
2	2	10	**17**	5	20
3	3	10	**18**	5	20
4	4	10	**19**	5	20
5	4	10	**20**	5	20
6	4	10	**21**	8	25
7	3	15	**22**	8	25
8	3	15	**23**	8	25
9	3	15	**24**	8	25
10	3	15	**25**	8	25
11	4	20	**26**	10	30
12	4	20	**27**	10	30
13	4	20	**28**	10	30
14	4	20	**29**	10	30
15	4	20	**30**	40	40

Table 5-2 Bathing rule

Level	Dirty Point	Experience	Level	Dirty Point	Experience
1	1	10	**16**	3	30
2	1	10	**17**	3	30
3	1	10	**18**	3	30
4	1	10	**19**	3	30
5	1	10	**20**	3	30
6	1	10	**21**	4	40
7	1	10	**22**	4	40
8	1	10	**23**	4	40

9	1	10	**24**	5	50
10	1	10	**25**	5	50
11	2	20	**26**	6	60
12	2	20	**27**	7	70
13	2	20	**28**	8	80
14	2	20	**29**	9	90
15	2	20	**30**	10	100

These rules are down by using “switch” statements, because the level of the pet must be an integer value, and each level owns different situation, so this way can make the rules run precisely [30]. In this system some levels share the same figure, in the switch statements the developer just need to remove the break statement, the codes will keep going, and those level can use the same figure. A part of the codes is shown as the figure 5-18.

```
public static Pet bathMethod(Pet myPet) {
    switch (myPet.getpLevel()) {
        case 1:
            myPet.setDirtyPoint(myPet.getDirtyPoint() - 1);
            myPet.setExperience(myPet.getExperience() + 10);
            break;

        case 2:
            myPet.setDirtyPoint(myPet.getDirtyPoint() - 2);
            myPet.setExperience(myPet.getExperience() + 10);
            break;

        case 3:
            myPet.setDirtyPoint(myPet.getDirtyPoint() - 3);
            myPet.setExperience(myPet.getExperience() + 10);
            break;
        case 4:
        case 5:
        case 6:
            myPet.setDirtyPoint(myPet.getDirtyPoint() - 4);
            myPet.setExperience(myPet.getExperience() + 10);
            break;
        case 7:
        case 8:
        case 9:
        case 10:
            myPet.setDirtyPoint(myPet.getDirtyPoint() - 3);
            myPet.setExperience(myPet.getExperience() + 15);
            break;
```

Figure 5-18 A part of the "switch" statements

The system also need to judge the level of the pet by its experience, and the relationship between the level and the experience is not a linear relationship, it is a range relationship, so the “switch” statements are no longer suitable for this situation, the developer used “if” and “else if” to divide the experience into pieces. The rules of the level and the pet’s skills are shown as the table 5-3 and table 5-4 respectively.

Table 5-3 Rule of levels

Level	Experience	Level	Experience
1	0-200	**16**	9400-10400
2	200-300	**17**	10400-11400
3	300-400	**18**	11400-12400
4	400-500	**19**	12400-13400
5	500-1000	**20**	13400-20000
6	1000-1600	**21**	20000-22000
7	1600-2200	**22**	22000-24000

8	2200-2800	**23**	24000-26000
9	2800-3400	**24**	26000-28000
10	3400-4400	**25**	28000-30000
11	4400-5400	**26**	30000-34000
12	5400-6400	**27**	34000-40000
13	6400-7400	**28**	40000-50000
14	7400-8400	**29**	50000-80000
15	8400-9400	**30**	>=80000

Table 5-4 Rule of skills

Skill ID	Skill Name	Level
1	Jump	5-8
2	Dance	8-9
3	Sing	9-12
4	Cry	12-14
5	Shy	14-16
6	Kiss	16-22
7	Angry	22-24
8	Hungry	24-30

The way of using “if” to judge the skill ID is shown as the figure 5-19, and the rule of level was done in the similar way.

```
private static Pet updateInfo(Pet myPet) {
    if (myPet.getpLevel() >= 24)
        myPet.setpSkill(8);
    else if (myPet.getpLevel() >= 22)
        myPet.setpSkill(7);
    else if (myPet.getpLevel() >= 16)
        myPet.setpSkill(6);
    else if (myPet.getpLevel() >= 14)
        myPet.setpSkill(5);
    else if (myPet.getpLevel() >= 12)
        myPet.setpSkill(4);
    else if (myPet.getpLevel() >= 9)
        myPet.setpSkill(3);
    else if (myPet.getpLevel() >= 8)
        myPet.setpSkill(2);
    else if (myPet.getpLevel() >= 5)
        myPet.setpSkill(1);
    return myPet;
}
```

Figure 5-19 UpdateInfo (): rule of judge the skill ID

As we know, after the users bathed or cleaned their pets, the experience the pets get may cause the level up. So, it is very important to check the experience of the level each time after the users bathed or cleaned their pets. Another situation may happen, it is that the user has not used the application for a long time, the hunger point or the dirty point may become a negative number followed the rules, so it is important to set it to 0 when the used uses this application again.

5.4.4 Foods Design

In this system, the developer created seven kinds of food, they are milk, bread, hotdog sandwich, hamburger, chicken and steak. To make the user experience better, the developer made the pets can have different kinds of food in different levels. With the level up, the more expensive the food will be, on the other hand the more experience and hunger points the pet will get. The details of the three kinds of food is shown below by table 5-5.

Table 5-5 Properties details of Foods

Food	Available levels	Cost	Hunger Point	Experience
Milk	1-5	2	1	10
Bread	3-10	1	2	15
Hotdog	3-10	2	3	15
Sandwich	6-20	3	4	20
Hamburger	11-30	5	7	40
Chicken	11-30	10	10	80
Steak	21-30	20	20	100

In this game, the users can only access three foods at one level in max. The developer written all the foods into the memory by declaring them in the field, then choose the food by the pet's level. Each food is put in a button, after having chosen the foods, the system need to put the image and the name of the food into the button to show the foods in the UI. The way of setting the text of button is very easy, just using setText method. The way of setting images into the buttons is given by the guide book, the developer should find the image by R.class, then put it into a Drawable object, after these the image can be put into a button, setCompoundDrawables () is the method to put the image into the button. The developer put the image of the food on the top of the text, some of the codes are shown by the figure 5-20 [31]. The foods shown in the UI are shown by the figure 5-21.

```
btn.setCompoundDrawablePadding(0);
btn.setPadding( left: 0,  top: 0,  right: 0,  bottom: 0);
if (thisFood.getFoodName().equals("Milk")) {
    Drawable drawable1 = getResources().getDrawable(R.drawable.milk);
    drawable1.setBounds( left: 0,  top: 0,  right: 100,  bottom: 100);
    btn.setCompoundDrawables( left: null, drawable1,  right: null,  bottom: null);
} else if (thisFood.getFoodName().equals("Bread")) {
    Drawable drawable1 = getResources().getDrawable(R.drawable.bread);
    drawable1.setBounds( left: 0,  top: 0,  right: 100,  bottom: 100);
    btn.setCompoundDrawables( left: null, drawable1,  right: null,  bottom: null);
```

Figure 5-20 Set images of foods

Figure 5-21 Foods shown in UI

5.4.5 Achai in Android

The Android system cannot play gif files, the developer must import the android-gif-drawable API from GitHub by using the statement “compile 'pl.droidsonroids.gif: android-gif-drawable:1.2. +'” in the app file. After this, the application can play gif files in the application [32].

Referenced from the QQ Pet, the hunger point progress bar, the food buttons and the cleaning buttons will hide before the user touch the pet. This means the UI will looks much more simple and clean when the user does not operate on it. The developer put all the food buttons and cleaning buttons in a layout which is named “foodLayout”, and put the “foodLayout”, the gifView and progress bar in a layout named “petLayout”. The developer set an onTouchListener on the “petLayout” and used a Boolean value “isHidden” to set the visibility of the “foodLayout” and progress bar. While the Boolean value is true the event will set the visibility to GONE which means invisible and take no space, and vice versa [33]. The figure 5-22 shows the two states of the “petLayout”.

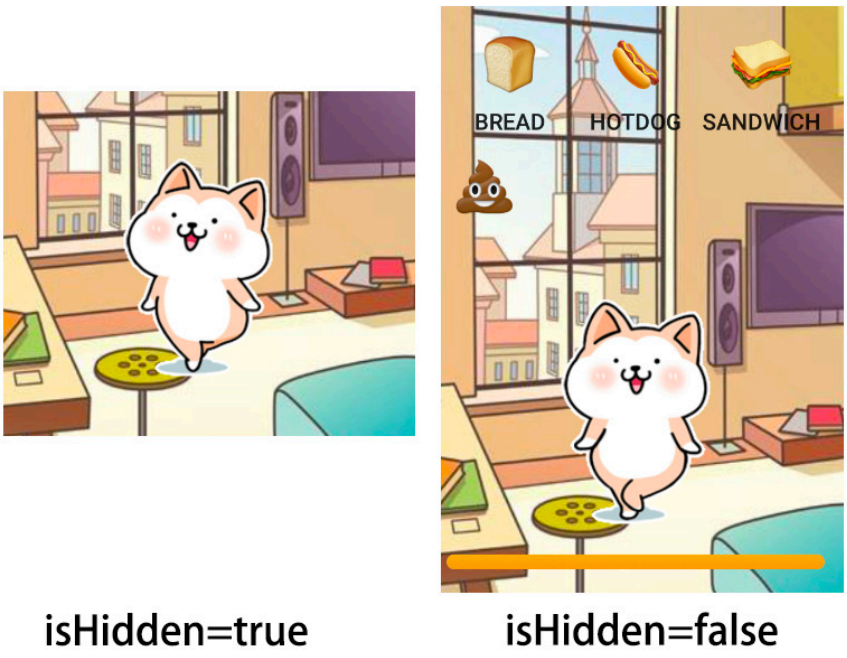

Figure 5-22 Two states of petLayout

5.4.6 Draggable Object

In this application, to make the pet can have interaction with the user and make the pet no longer a still image, the developer made the pet object can be dragged by the user. As Android system did not provide any API for the developers to make the pet can be dragged, the developer had to rewrite the basic codes of the View class. Although Android system did not provide any API to drag the controls, it provides a tool to help developers to get the data and do some operation on the control while a control is dragged. This tool is named ViewDragHelper. “It offers a number of useful operations and state tracking for allowing a

user to drag and reposition views within their parent ViewGroup" [34], explained by the Android guide book. The developer rewritten the methods of reposition to make sure the dragged object always in the page which is shown to users. The location of the object is decided by the "top" and "left". What are the "top" and "left" mean is shown by the figure 5-23.

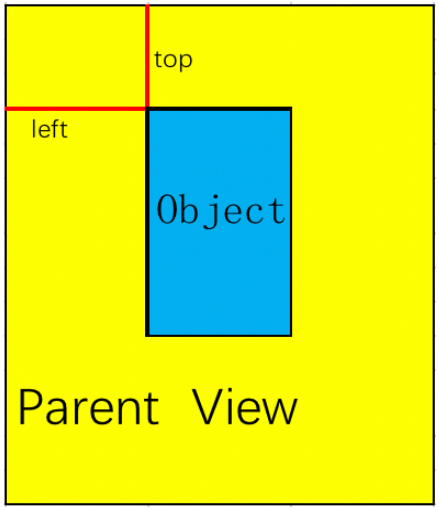

Figure 5-23 Top and left

To make the object always in the parent view, the top must larger than 0 and cannot larger than the height of the parent view minus the height of the object itself. On the other hand, the left must larger than 0 and cannot larger than the width of the parent view minus the width of the object itself. The codes of the two methods is shown as the figure 5-24.

```
@Override
public int clampViewPositionHorizontal(View child,
                                       int left, int dx) {

    if (getPaddingLeft() > left) {
        return getPaddingLeft();
    }
    if (left > (getWidth() - getPaddingLeft()
            - getPaddingRight() - child.getWidth())
            + getPaddingLeft()) {
        return (getWidth() - getPaddingLeft()
                - getPaddingRight() - child.getWidth())
                + getPaddingLeft();
    }

    if (getWidth() - child.getWidth() < left) {
        return getWidth() - child.getWidth();
    }
    return left;
}

@Override
public int clampViewPositionVertical(View child, int top,
                                     int dy) {

    if (getPaddingTop() > top) {
        return getPaddingTop();
    }

    if (getHeight() - child.getHeight() < top) {
        return getHeight() - child.getHeight();
    }

    return top;
}
```

Figure 5-24 Methods of location

The ViewDragHelper was put into a new kind of view class written by the developer which is named CustomView. After the view class was created, the developer can use the class in the xml file and do not need to write any code in the java file.

```
    android:orientation="vertical">
<include
    layout="@layout/activity_title"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    />
    <cn.moecity.virtualpet.CustomView
        android:id="@+id/moveLayOut"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:layout_weight="9"
        android:columnCount="1"
        android:orientation="vertical">
        <LinearLayout...>


    </cn.moecity.virtualpet.CustomView>
    <include
        layout="@layout/activity_buttom"
        android:layout_width="match_parent"
        android:layout_height="20dp"
        android:layout_weight="1"
        />
```

Figure 5-25 The way of using CustomView in xml file

5.4.7 Algorithms of Pet's Data

In this system, the developer designed that ill point is depends on the hunger point and dirty point. The relationship among the ill point, hunger point and dirty point is: illPoint = dirtyPoint / 2 + hungPoint / 2. While the ill point gets to 10 the pet will ill and need to take the injection to recover from ill. An injection will cost user £100.

As we know, the hunger point is depending on the gap time of feeding. The developer set that the hunger point will reduce one point every 14400 microseconds (4 hours). The maximum of the hunger points is 10.

The dirty point is much more complex, it is determined by the gap time of bathing and the gap time of cleaning. The algorithm is that: dirtyPoint=(int) (dirtyPoint + bathgap / 57600 + cleangap / 57600). This means every 18 hours; the dirty point will reduce one point without bathing or cleaning.

To make sure the systems goes efficiently, the developer made point operation only happened when then gap time is longer than 7200 microseconds (2 hours). And while the point is 10, the operation will also stop. Java provides an API to get time from the device, this class named Date, the developer used Date.getTime() to get a long integer which means the time in microsecond since 1th January 1970 [35]. To make the figure smaller, the developer subtracted 1511450000 (the day of 11st November 2017) from the value. By this way, the system can figure out the time gaps.

The codes of the algorithms are shown by the figure 5-26.

```
date = new Date();
int illPoint = myPet.getIllPoint();
int hungPoint = 0;
int dirtyPoint = 0;

current = date.getTime() / 1000 - 1511450000;
checkCollect();
feedgap = current - myPet.getFeedTime();
bathgap = current - myPet.getBathTime();
cleangap = current - myPet.getCleanTime();
if (feedgap >= 7200 && illPoint < 10) {
    hungPoint=((int) (hungPoint + (feedgap / 14400)));
    if (hungPoint >= 10) {
        hungPoint=10;
        myPet.setHungPoint(10);
        myPet.setFeedTime(current);
    }
}
if ((bathgap >= 7200 || cleangap >= 7200) && illPoint < 10) {
    dirtyPoint=(int) (dirtyPoint + bathgap / 57600 + cleangap / 57600);
    if (dirtyPoint >= 10) {
        dirtyPoint=10;
        myPet.setDirtyPoint(10);
        myPet.setBathTime(current);
        myPet.setCleanTime(current);
    }
    myPet.setHungPoint(hungPoint);
    myPet.setDirtyPoint(dirtyPoint);
    illPoint=(dirtyPoint / 2 + hungPoint / 2);
    myPet.setIllPoint(dirtyPoint / 2 + hungPoint / 2);
    if(illPoint>=10){
        illPoint=10;
        isDied=true;
    }
```

Figure 5-26 Algorithms of pet's data

5.4.8 Infinite Loop Thread

In this system requires a thread to monitor the time gaps and change the data of the pet in time in the UI thread while the system is running. However, the Android system does not allow the other threads do any operation on the UI thread. To make the developers do some operations on the UI thread, the Android system provide a framework which is named Handler to send messages into the UI thread while it is running [36].

Timer is a facility for threads to schedule tasks for future execution in a background thread. Tasks may be scheduled for one-time execution, or for repeated execution at regular intervals. So, the developer used timer to do repeat the task of checking the time gaps. In the future development, the developer found the timer may have a collision with the on click events. To minimize this kind of collision, the developer made the time schedule this task every 3000 microseconds (3 seconds) and run for 1000 microseconds (1 second) each time. This can help the data stay in a safety state and it can also help the application costs less CPU.

```
private void setTimerTask() {
    timer.schedule(new TimerTask() {

        @Override
        public void run() {
            Message message = new Message();
            message.what = 1;
            doActionHandler.sendMessage(message);
        }
    }, delay: 1000,  period: 3000);
}
```

Figure 5-27 Timer

```
    private Handler doActionHandler;

    {
        doActionHandler = new Handler() {
            @Override
            public void handleMessage(Message msg) {
e static or leaks might occur (anonymous android.os.Handler) more...
                int msgId = msg.what;
                switch (msgId) {
                    case 1:
                        updateState();
                        break;
                    default:
                        break;
                }
            }
        };
    }
```

Figure 5-28 Handler

5.4.9 User Interaction

In this system, most of the user interaction events were down by the button controls. The statement “android:onClick="handleClicked"” was added into the xml statements of each button, then a handleClicked () method was added into the Java files to define the event of each button. And the buttons were located by its id defined in the xml files by using the statement “android:id="@+id/ID OF THE VIEW"”.

This page contains a collecting button, a bathing button, a logout button, an information button, an injection button, three food buttons and five cleaning buttons.

The collecting button is used to collect money from the system which is given every two hours, and the user will get £200 every time. This button need to be monitor in the 5.4.7 infinite loop thread to update the gap time from the last collecting, and then shown in the bottom of the button. The image of the button also need to be changed while the money can be collected. A checkCollect () method was written to check if the money is read to collected, while the gap time is larger than or equals 7200 seconds (2 hours), the button will change its image into the collectable one and change the text into “collect”. Otherwise, the image will still be the one which means cannot collect, and print the time gap in the form of “hh:mm:ss”. If the money is collected, the event will save the collecting time into SharedPreferences, and add £200 into the balance.

```
private void checkCollect() {
    if (current - collectTime >= 7200) {
        collectButton.setCompoundDrawables( left: null, moneyD,  right: null,  bottom: null);
        collectButton.setText("Collect");
    } else {
        collectButton.setCompoundDrawables( left: null, moneyD1,  right: null,  bottom: null);
        String mm, ss;
        int gap = (int) (7200 - (current - collectTime));
        int shi = gap / 3600;
        int fen = (gap % 3600) / 60;
        int miao = gap % 60;
        if (fen < 10)
            mm = "0" + fen;
        else
            mm = "" + fen;
        if (miao < 10)
            ss = "0" + miao;
        else
            ss = "" + miao;
        collectButton.setText("" + shi + ":" + mm + ":" + ss);
    }
}
```

Figure 5-29 CheckCollect () method

The bathing button is used to bath the pet, the dirty point will reduce 1 point, each time will cost the user £10, if the dirty point is 0 or less, the event will do nothing but give the user an alert, “I’m very clean now!” So are the cleaning buttons, but the cleaning buttons will reduce 2 points each time. As the five clean buttons use the same event, so the developer just need to write the statements once, it is just to remove all the “break” from the “case” statements.

```
case R.id.shit1:
case R.id.shit2:
case R.id.shit3:
case R.id.shit4:
case R.id.shit5:
    updateState();
    if (balance > 0) {
        if (myPet.getDirtyPoint() > 0) {
            DataMethods.cleanMethod(myPet);
            date = new Date();
            now = date.getTime() / 1000 - 1511450000;
            balance = balance - 10;
            myPet.setCleanTime(now);
            createShit();
            updateState();
        } else {
            Toast.makeText(getApplicationContext(),
                    text: "I'm very clean now!", Toast.LENGTH_SHORT).show();
        }
    }
    break;
```

Figure 5-30 Statements of cleaning buttons

The cleaning buttons also need to be monitored by the infinite loop thread to control the numbers of the buttons shown in the UI. As one clean button represents two dirty points by an image of shit, the number of the buttons also represents the dirty point. A createShit () method was written to check the dirty point and set the visibility of the buttons. All the cleaning buttons are in an array. At the beginning, the method set all the button’s visibility to invisible and disable then by a “for” loop, then set the buttons’ visibilities to visible and enable the buttons in a “for” loop, with the time is dirtyPoint/2. The method is shown by the figure 5-31.

```
private void createShit() {
    for (int i = 0; i < 5; i++) {
        Drawable drawable1 = getResources().getDrawable(R.drawable.shit);
        drawable1.setBounds( left: 0,  top: 0,  right: 100,  bottom: 100);
        shitBtn[i].setCompoundDrawables( left: null, drawable1,  right: null,  bottom: null);
        shitBtn[i].setVisibility(View.INVISIBLE);
        shitBtn[i].setEnabled(false);
    }
    for (int i = 0; i < (myPet.getDirtyPoint() / 2); i++) {
        shitBtn[i].setVisibility(View.VISIBLE);
        shitBtn[i].setEnabled(true);
    }
}
```

Figure 5-31 CreateShit () method

The food buttons are linked with three available foods, when the buttons are clicked, the system will check if the balance is enough to pay for the food, then it will set the hunger point.

The injection button is only available when then ill point is 10, which means the pet is ill. The developer used a Boolean value isDied to mark if the pet is ill, and monitoring it in the infinite loop thread, while the Boolean value is true, the system will disable all the food buttons and hide then together with the cleaning buttons. An injection will cost £100, set the hunger point to 1, set ill point to 0, and set the Boolean value to false.

The information button is a link to a new page with shows all the data in figure. Logout event used a Boolean value which is named isOpen to judge if the user is logged out. To make the

application much more convenient to use, once the user log in the user no longer need to log in again unless he/she logged out. The value of isOpen is saved in the SharedPreferences, the first time the user login, the system will set it to “true”, each time the user opens this application the system will do the judgement, the page will go to the core page if “ture” or go to the login page if “false” or no data.

5.4.10 Debugging

After having done the main page, the developer found the pet never changed and the pet’s skills are never used. To make the page dynamic and make full of the skill property, the developer put the id of the gif files in an array by the rank of the skills. Each time when the user feeds the pet, the system will find a random number between 0 and the pet’s skill number. And then set the gif of the pet by the id from the array with the random. This made the system has more interactions with users and can encourage users to play this game more often to unlock more skills. The statements of the making random and changing gif files are shown by figure 5-32.

```
private void changeImg(Pet myPet) {
    DataMethods.levelCheck(myPet);
    imgNum = makeRandom(myPet.getpSkill());
    petView.setBackgroundResource(drawGif[imgNum]);

}

private void checkDied() {...}

private int makeRandom(int max) {
    int min = 0;
    int s = 0;
    Random random = new Random();
    if (max > 0)
        s = random.nextInt(max) % (max - min + 1) + min;
    return s;
}
```

Figure 5-32 Making random and changing gif files

5.5 Online Modules

5.5.1 Setting Online Database

The developer rented a server produced by Alibaba to set the database and linked its IP with a domain www.moecity.cn which the developer bought before, so that the developer does not need to remember the IP address but this domain while doing the development. The platform of this server id Windows Server 2012 R2, and the database of this server is SQL Server 2016 Express. Before doing the development, the developer added the port of 1433 into the rule of inbound packages and rule of outbound packages to let the SQL Server database can share data with the public network [37]. Then the developer can use the database by the Android application. The data in the SQL Server is shown by the chapter 4.4.

5.5.2 Web Service

“Web services are web application components. Web services can be published, found, and used on the Web.” W3school.com explained [38]. Web Service is the way the developer used to make a bridge between SQL Server and Android application. This Web Service is written by C# with Microsoft Visual Studio.

This web service is used to translate data between Android application and SQL Server. It need to have the methods which the Android application needs. This web service contains these methods shown by figure 5-32.

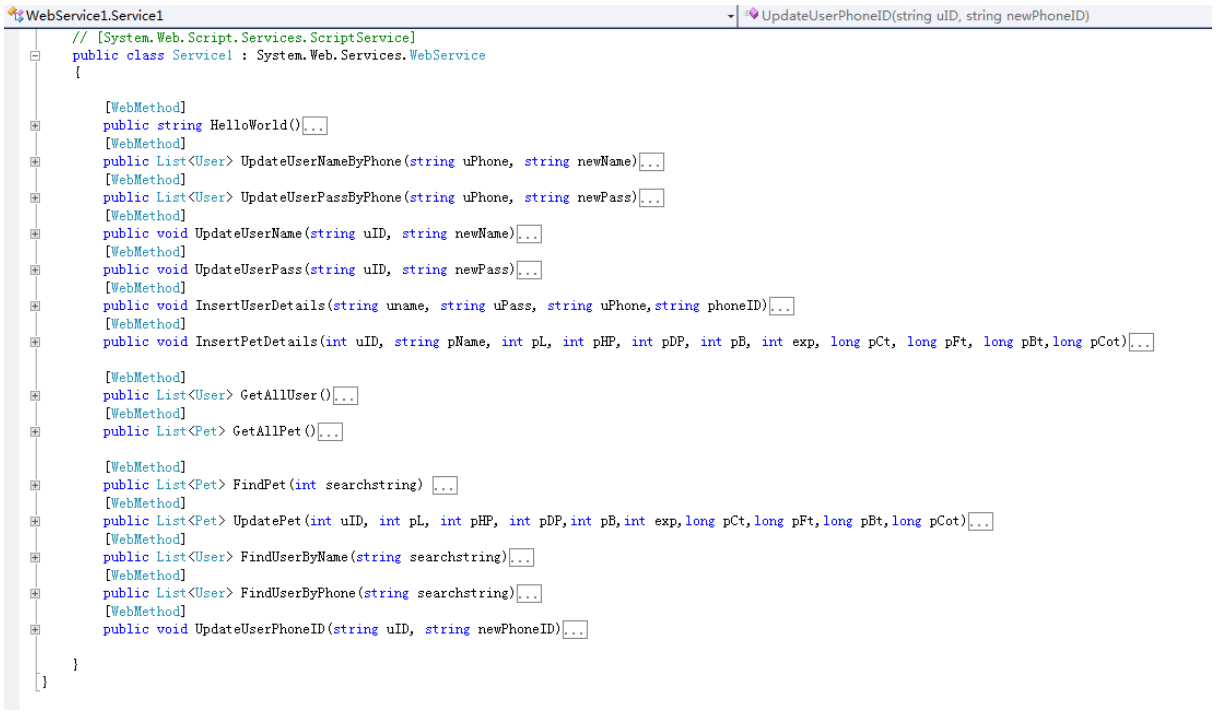

Figure 5-33 Methods of Web Service

In this system, what the web service need to do is get and send command to the database. So, the developer used SqlCommand class to send the SQL commands to the database as the database can only identify the SQL commands. Web service also provides a SqlDataReader class to read the return values of the SQL commands by the column name. Figure 5-34 shows the method of get all users’ data from the database, and other methods are written in the similar way [39].

```
[WebMethod]
public List<User> GetAllUser()
{
    List<User> UserList = new List<User>();

    // Write some SQL statement
    string SQL = "select * from UserTable";

    // Connect to DB
    SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["WebService1.Properties.Settings.Setting"].ToString());
    // Open DB
    conn.Open();

    SqlCommand cmd = new SqlCommand(SQL, conn);

    SqlDataReader reader = cmd.ExecuteReader();
    try
    {
        while (reader.Read())
        {
            User s = new User();
            s.uID = int.Parse(reader["UID"].ToString());
            s.uname = reader["Uname"].ToString();
            s.uPass = reader["UPass"].ToString();
            s.uPhone = reader["UPhone"].ToString();
            s.phoneID = reader["UPhoneID"].ToString();
            UserList.Add(s);
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
    reader.Close();
    conn.Close();

    return UserList;
}
```

Figure 5-34 Method of getting all users' data

The differences are the SQL commands among those methods. To make the system clean and easy to manage, the SQL commands are written in the way of basic CRUD rule. The commands are shown as table 5-6

Table 5-6 Rule of CRUD

Function	Command
Create	INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...)
Read	SELECT column_name FROM table_name
Update	UPDATE table_name SET column_name = new_value WHERE column_name = value1
Delete	DELETE FROM table_name WHERE column_name = value1

5.6 Account Protection

In this system as the device can save the user's data on the SharedPreferences, once the device has lost, the user's data will be in danger. What the developer need to do is to help the user to log out their account when their device is lost.

The way to log out an account is to log in it in a new device, so it comes the problem that how to judge if the user changed a new device? Is there any way that can identify an Android device? What came to the developer's mind is to use the MAC address to identity the device. However, for the security issue, Android will not provide user's hardware information include MAC address, IP address and IMEI to developers without permission since Android 6.0 [40]. ANDROID_ID is a 64 bits length string which Android provide to identity the device, and it can easily access without any permission [41].

The developer added a column which is named phoneID in the user_table to save the ANDROID_ID of the user's last used device. Every time when the user opened the CoreActivity, this system will check the lock ANDROID_ID with the one in the database. If the id is different from the online one, the system will set the Boolean value isOpen to false and save it into the SharedPreferences and change the page to LoginActivity and clean all the local data.

```
String android_id = Settings.Secure.getString(this.getApplicationContext().getContentResolver(),
        Settings.Secure.ANDROID_ID);
```

Figure 5-35 The statement of getting ANDROID_ID

To make this function comes true, a method of getting phoneID from the database was added into the web service, the statement of getting ANDROID_ID need to be added in the LoginActivity and NewPetActivity and then upload into the online database. An asynchronous task of comparing local ANDROID_ID with the online one was added into the CoreActivity to judge whether to delete the local data or not. To make the user interface more interactive, the developer used a dialog alert to give the user feedback. The codes of making dialog alert is shown as the figure 5-36. The UI of dialog alert is shown as figure 5-37.

```
@Override
protected void onPostExecute(Object o) {
    super.onPostExecute(o);
    if (!(phoneID.equals(localPhoneID))) {
        isSafe = false;
        SharedPreferences.Editor editor = userData.edit();
        editor.putBoolean("isIn", false);
        editor.putString("userName", "NoName");
        editor.commit();
        new AlertDialog.Builder( context: CoreActivity.this)

                .setTitle("Error")
                .setMessage("You have changed your phone,\nplease login again!")
                .setOnCancelListener(new DialogInterface.OnCancelListener() {
                    @Override
                    public void onCancel(DialogInterface dialog) {
                        Intent intent = new Intent( packageContext: CoreActivity.this, LoginActivity.class);
                        startActivity(intent);
                        finish();
                    }
                })
                .setPositiveButton( text: "Login",
                        new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialoginterface, int i) {
                                Intent intent = new Intent( packageContext: CoreActivity.this, LoginActivity.class);
                                startActivity(intent);
                                finish();
                            }
                        }).show();
```

Figure 5-36 Codes of making dialog alert

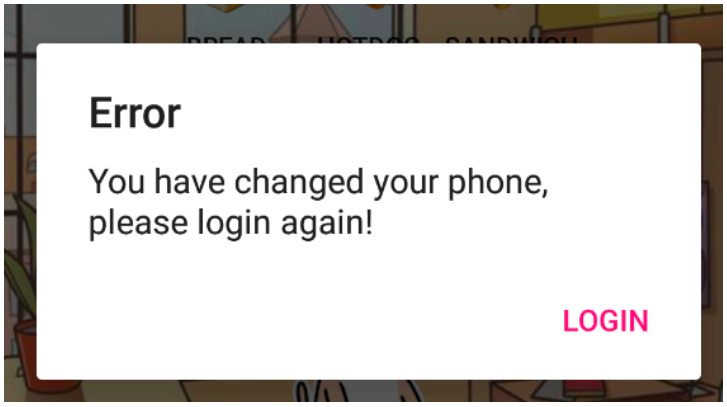

Figure 5-37 UI of dialog alert

6 RESULT AND EVALUATION

6.1 Result

In this finished application, users can see the advertisements provided by developer when they open this application. A skip button on the upper left corner can help them skip this advertisement if they do not want to watch. The advertisement will last 3 seconds, then users can log in with their accounts, or create new account if they are new users. The users who has logged in with this device before do not need to enter their account information again, the can directly go to the main page. After they enter the system, system will check the local ANDROID_ID with the online one to make sure the device is the same one. Otherwise, system will clean all the user's local data.

The name, level, balance and the dirty point were shown on the top of the screen. A logout button is on the lower right corner to let users log out their account and delete the local data. The pet stand on the centre of the screen, and the user can drag it inside the screen it will move bac to the centre once the user releases it.

A menu will appear while the user clicks on the pet. This menu shows the three available foods with the dirty point shown by shits on the top, and a progress bar shows the hunger point in the bottom. Once the user feed the pet, the pet will change its action, the actions are decided by the skills it have. Skills are defined by its level. This application contains five pages, the UI of these five pages are shown by figure 6-1.

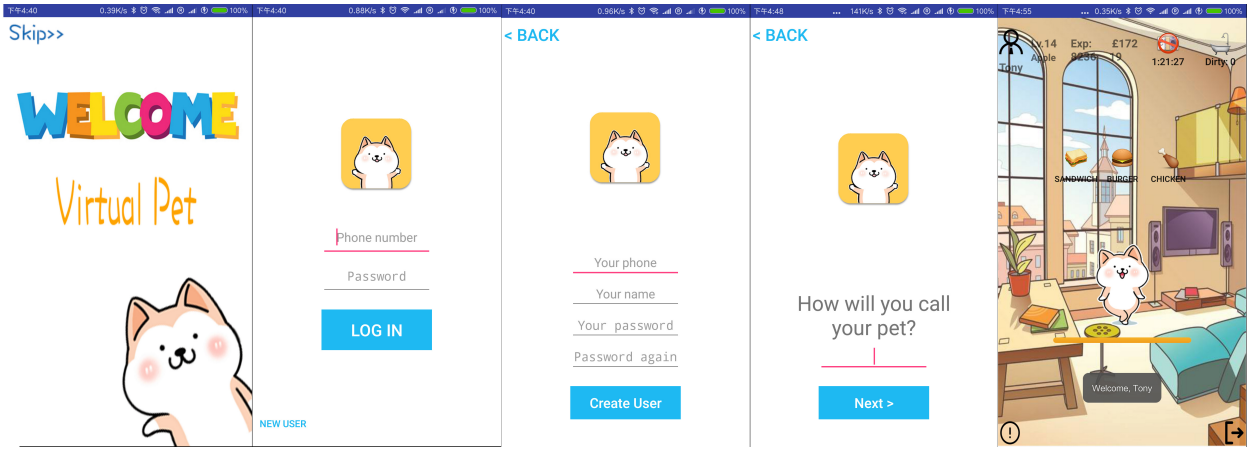

Figure 6-1 The pages of this application

6.2 Evaluation

The developer has done most of the modules which the users need in the research and added a function of protecting users' account. Because of the time limited, the developer did not finish the chatting module, although the dialogflow had been chosen as the API to realize this module.

7 FURTHER DEVELOPMENT

The developer needs to do more on the UI design in the future. And the chatting module must be done in the future development as the API had been chosen before the developer decided to cut this part. More functions can be added to this application to make this game has more interactions with users.

In the future, a personalization system can be added to make users can choose their pets' kinds and dress them up. This is the way to larger the user group, to make the users not only the dog lovers [42].

The pet which has done in this application cannot act smoothly because the actions are limited by the gif files. In the future development, the developer wants to build a 3D model, set some key points on the main joints, so that the pet can act more naturally and smoothly.

During the development, the developer found the Android do not contains any physical engines, it makes the problem that when the user dragged the pet to the high place, the pet cannot fall from that place, it seems that it is suspended in the air. Making a physical algorithm is a very hard job, there are many game engines on the market, the developer can rewrite this application based on a game engine in the future development. If so, the UI will be better, the character building will be more beautiful and the users can have more interactions with their pets [43].

8 CONCLUSION

This paper has described the process of making a virtual pet application.

The developer did a research online to find out the users' need, the developer used dog as the character because of the high public support, the main modules of chatting, feeding, cleaning and playing are all decided by the support rate of the respondents.

To please the users who are mainly children, the developer chosen yellow, orange and blue as the theme colours.

The developer set the algorithms of rules of the feeding, cleaning, bathing and level upgrading. This Android application used SharedPreferences to save the local data, SQL Server was used to save the online database and Web Service was used to link the online database with the local one. Asynchronous task was used to run the task which need to be run in the background. Timer and handler were used to realize the infinite loop thread. ViewDragHelper was used to make the object can be dragged.

What is more, the developer added an account protection method to protect the users' data, as the data security is seem to a big issue nowadays. Because of the time limited, the developer did not finish the chatting module, this made the system less intractable, that was a great regret.

9 OTHER POINTS

9.1 Initial Gantt Chart

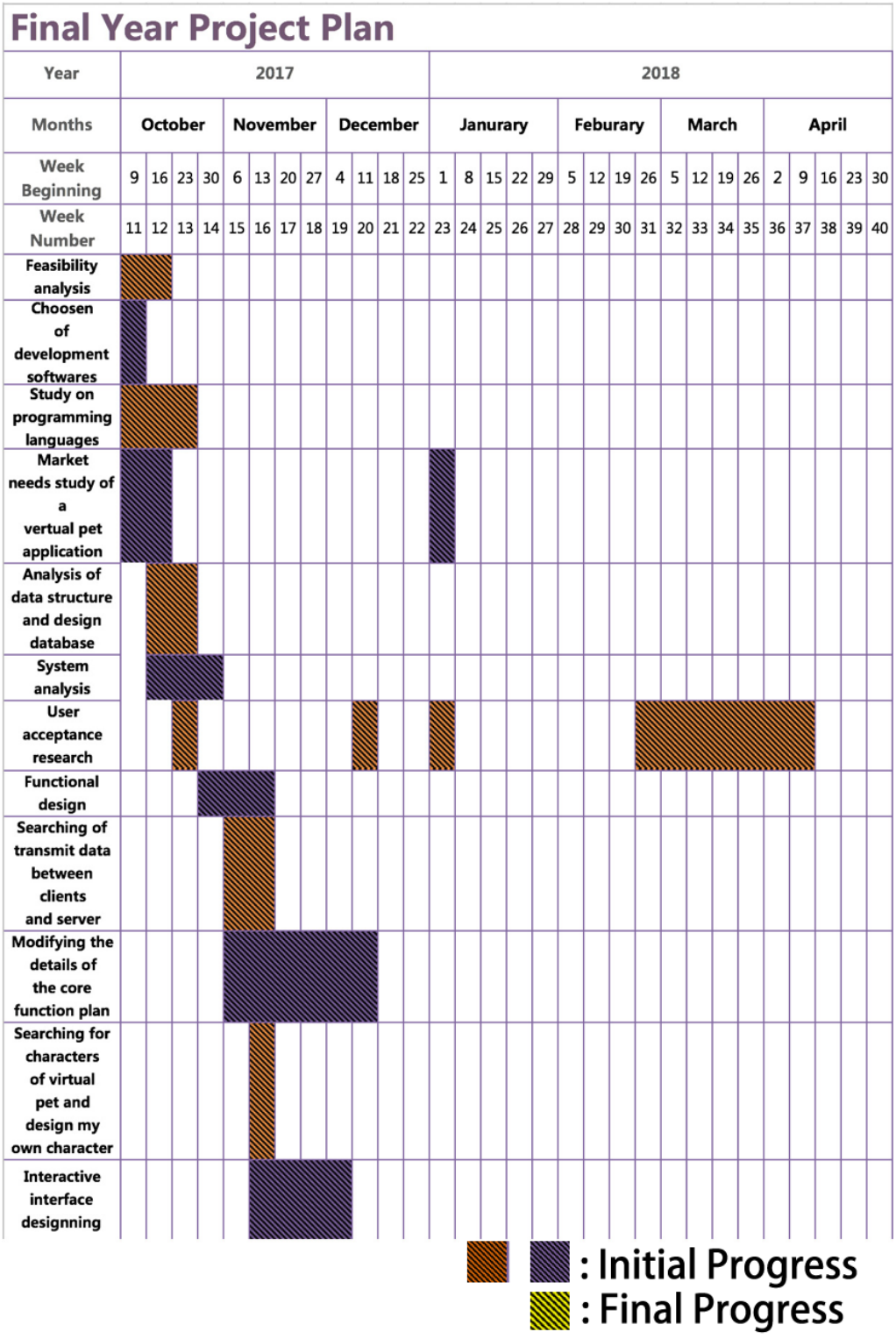

Figure 9-1 Initial Gantt chart part 1

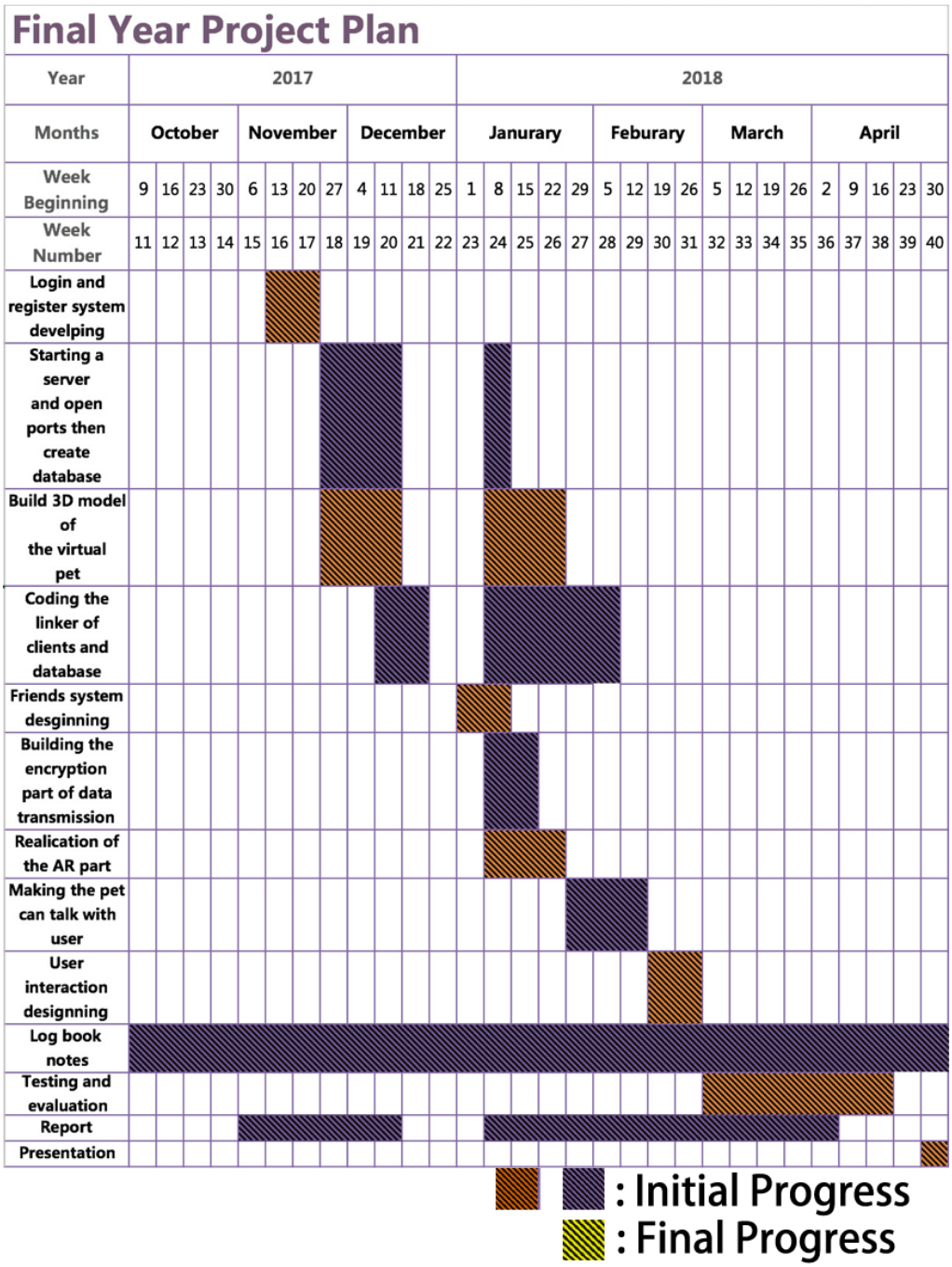

Figure 9-2 Initial Gantt chart part 2

9.2 Finial Gantt Chart

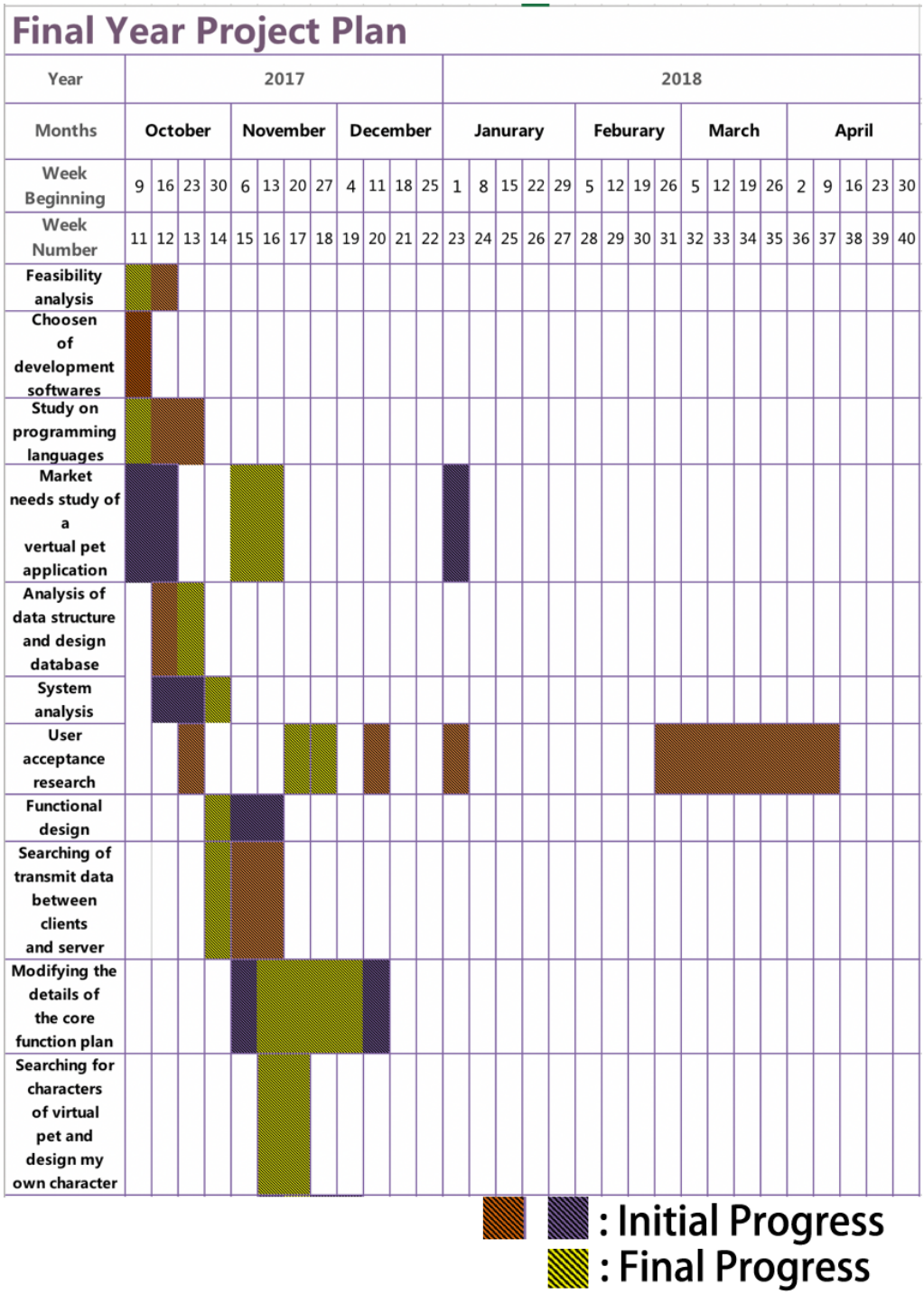

Figure 9-3 Final Gantt chart part 1

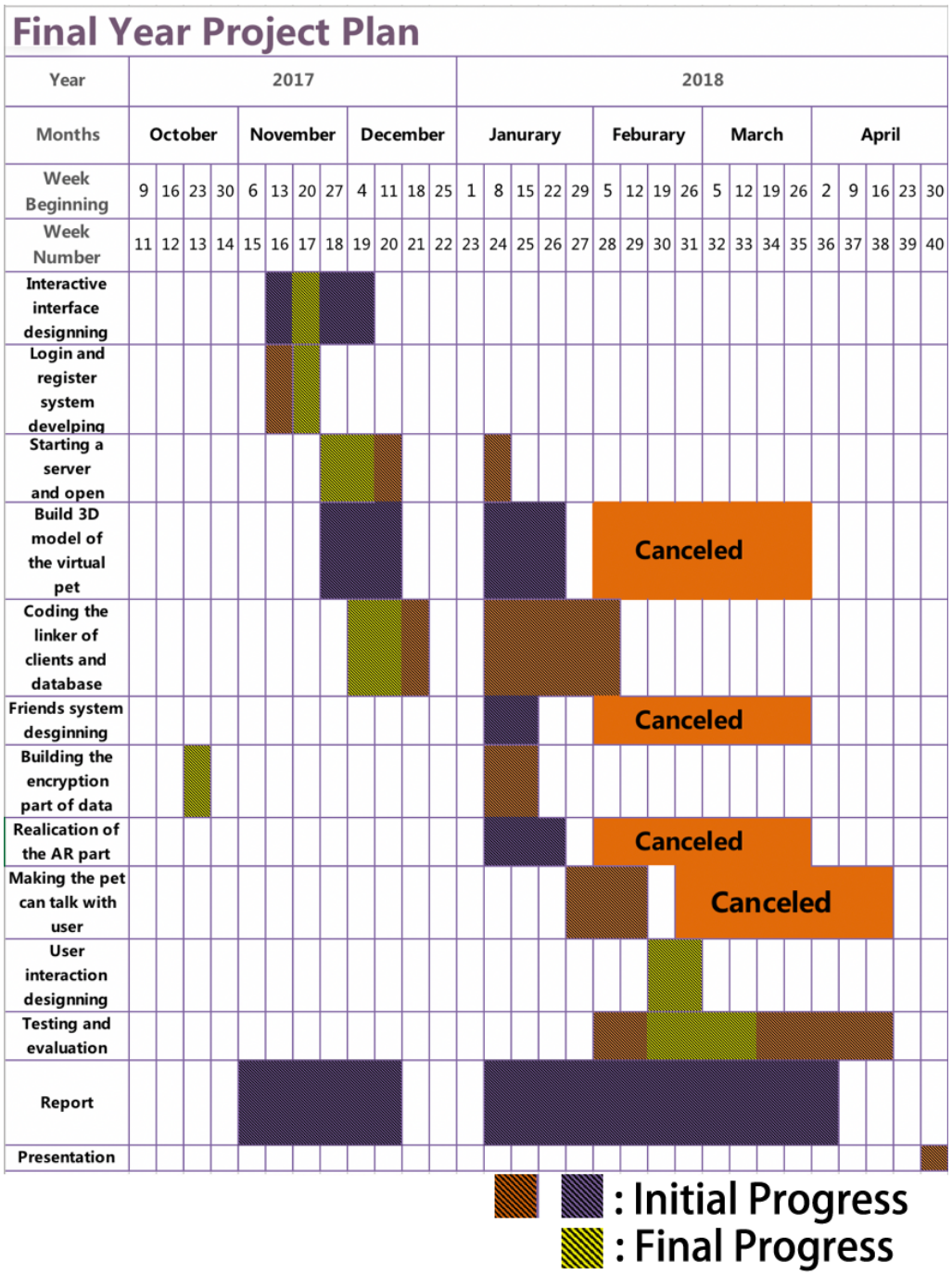

Figure 9-4 Final Gantt chart part 2

REFERENCES

[1] Bbc.com. (2018). Why do we love our pets so much? [online] Available at: http://www.bbc.com/earth/story/20150530-why-do-we-love-our-pets-so-much [Accessed 7 Mar. 2018].

[2] Schooliseasy.com. (2018)5 benefits related to pets and child development. [online] https://www.schooliseasy.com/2014/03/pets-and-child-development/ [Accessed 7 Mar. 2018].

[3] Downtowndogrescue.org (2018). The #1 reason pets are surrendered: "I'm moving and I can't take my pet." But, why?. [online] Downtown Dog Rescue. Available at: http://downtowndogrescue.org/im-moving-and-i-cant-take-my-pet-the-1-reason-pets-are-surrendered-and-why/ [Accessed 7 Mar. 2018].

[4] Statista.com. (2018). Number of mobile phone users worldwide 2013-2019 | Statista. [online] Available at: https://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/ [Accessed 22 Mar. 2018].

[5] Statista.com. (2018). Android OS smartphone shipments market share 2011-2017 | Statistic. [online] Available at: https://www.statista.com/statistics/236027/global-smartphone-os-market-share-of-android/ [Accessed 7 Mar. 2018].

[6] Statista.com. (2018). Android versions market share 2017 | Statista. [online] Statista. Available at: https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/ [Accessed 7 Mar. 2018].

[7] Outfit7.com. (2018). Outfit7 - My Talking Tom. [online] Available at: https://outfit7.com/apps/my-talking-tom/ [Accessed 7 Mar. 2018].

[8] Hit-Point Co., L. (2018). Travel Frog. [online] Hit-point.co.jp. Available at: http://www.hit-point.co.jp/games/tabikaeru/ [Accessed 7 Mar. 2018].

[9] Act.qzone.qq.com. (2018). QQ Pet. [online] Available at: https://act.qzone.qq.com/vip/2017/pets-launch-m/?_wv=1025 [Accessed 7 Mar. 2018].

[10] Wj.qq.com. (2018). Tencent Online Questionnaire. [online] Available at: https://wj.qq.com/ [Accessed 7 Mar. 2018].

[11] Developer.android.com. (2018). Download Android Studio and SDK Tools | Android Studio. [online] Available at: https://developer.android.com/studio/index.html [Accessed 7 Mar. 2018].

[12] Developer.android.com. (2018). Developer Guides | Android Developers. [online] Available at: https://developer.android.com/guide/index.html [Accessed 7 Mar. 2018].

[13] TechWorm. (2018). How long will your latest flagship smartphone really last?. [online] Available at: https://www.techworm.net/2017/07/long-will-latest-flagship-smartphone-really-last.html [Accessed 24 Mar. 2018].

[14] Alibabacloud.com. (2018). An integrated suite of cloud products, services and solutions | Alibaba Cloud. [online] Available at: https://www.alibabacloud.com/ [Accessed 7 Mar. 2018].

[15] Microsoft SQL Server. (2018). SQL Server 2016 | Microsoft. [online] Available at: https://www.microsoft.com/en-us/sql-server/sql-server-2016 [Accessed 7 Mar. 2018].

[16] TechCrunch. (2018). How Free Apps Can Make More Money Than Paid Apps. [online] Available at: https://techcrunch.com/2012/08/26/how-free-apps-can-make-more-money-than-paid-apps/ [Accessed 24 Mar. 2018].

[17] Official Wix Blog | Web Design & Small Business Tips to Promote Your Site. (2018). How to Choose The Perfect Color Palette For Your Business. [online] Available at: https://www.wix.com/blog/2017/10/how-to-choose-the-perfect-color-palette-for-your-business/?utm_source=Wix+Blog&utm_campaign=551f8e30f6-UA-2117194-5&utm_medium=email&utm_term=0_324de5e2c6-551f8e30f6-157610029 [Accessed 7 Mar. 2018].

[18] Zcool.com.cn. (2018). Achai Emoji. [online] Available at: http://www.zcool.com.cn/work/ZMjIwMTI3ODA=/2.html [Accessed 7 Mar. 2018].

[19] Developer.apple.com. (2018). App Icon - Icons and Images - iOS Human Interface Guidelines. [online] Available at: https://developer.apple.com/ios/human-interface-guidelines/icons-and-images/app-icon/ [Accessed 7 Mar. 2018].

[20] Developer.apple.com. (2018). App Icon - Icons and Images - Human Interface Guidelines for macOS Apps. [online] Available at: https://developer.apple.com/macos/human-interface-guidelines/icons-and-images/app-icon/ [Accessed 24 Mar. 2018].

[21] Applypixels.com. (2018). How to Design Better App Icons. [online] Available at: https://applypixels.com/how-to-design-better-app-icons/ [Accessed 24 Mar. 2018].

[22] Developer.android.com. (2018). View | Android Developers. [online] Available at: https://developer.android.com/reference/android/view/View.html?hl=zh-cn#attr_android:onClick [Accessed 7 Mar. 2018].

[23] Docs.oracle.com. (2018). Boolean (Java Platform SE 7). [online] Available at: https://docs.oracle.com/javase/7/docs/api/java/lang/Boolean.html [Accessed 22 Mar. 2018].

[24] Developer.android.com. (2018). Connecting to the Network | Android Developers. [online] Available at: https://developer.android.com/training/basics/network-ops/connecting.html [Accessed 7 Mar. 2018].

[25] W3schools.com. (2018). XML Introduction. [online] Available at: https://www.w3schools.com/xml/xml_whatis.asp [Accessed 7 Mar. 2018].

[26] Developer.android.com. (2018). URLConnection | Android Developers. [online] Available at: https://developer.android.com/reference/java/net/URLConnection.html [Accessed 7 Mar. 2018].

[27] Developer.android.com. (2018). StrictMode.ThreadPolicy.Builder | Android Developers. [online] Available at: https://developer.android.com/reference/android/os/StrictMode.ThreadPolicy.Builder.html [Accessed 7 Mar. 2018].

[28] Developer.android.com. (2018). Save Key-Value Data with SharedPreferences | Android Developers. [online] Available at: https://developer.android.com/training/data-storage/shared-preferences.html [Accessed 7 Mar. 2018].

[29] Developer.android.com. (2018). AsyncTask | Android Developers. [online] Available at: https://developer.android.com/reference/android/os/AsyncTask.html [Accessed 7 Mar. 2018].

[30] Liang, Y. (2015). Intro to Java Programming. 10th ed. New York: Pearson Education Limited, p.75.

[31] DiMarzio, J. (n.d.). Beginning Android programming with Android Studio. 4th ed. Indianapolis, Indiana: John Wiley & Sons, Inc, p143

[32] GitHub. (2018). koral--/android-gif-drawable. [online] Available at: https://github.com/koral--/android-gif-drawable [Accessed 7 Mar. 2018].

[33] DiMarzio, J. (n.d.). Beginning Android programming with Android Studio. 4th ed. Indianapolis, Indiana: John Wiley & Sons, Inc, p105

[34] Developer.android.com. (2018). ViewDragHelper | Android Developers. [online] Available at: https://developer.android.com/reference/android/support/v4/widget/ViewDragHelper.html [Accessed 7 Mar. 2018].

[35] DiMarzio, J. (n.d.). Beginning Android programming with Android Studio. 4th ed. Indianapolis, Indiana: John Wiley & Sons, Inc, p226

[36] Developer.android.com. (2018). Handler | Android Developers. [online] Available at: https://developer.android.com/reference/android/os/Handler.html [Accessed 7 Mar. 2018].

[37] Docs.microsoft.com. (2018). Configure the remote access Server Configuration Option. [online] Available at: https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/configure-the-remote-access-server-configuration-option [Accessed 7 Mar. 2018].

[38] W3schools.com. (2018). XML Web Services. [online] Available at: https://www.w3schools.com/xml/xml_services.asp [Accessed 7 Mar. 2018].

[39] W3schools.com. (2018). SQL Tutorial. [online] Available at: https://www.w3schools.com/sql/default.asp [Accessed 7 Mar. 2018].

[40] Developer.android.com. (2018). Best Practices for Unique Identifiers | Android Developers. [online] Available at: https://developer.android.com/training/articles/user-data-ids.html [Accessed 7 Mar. 2018].

[41] Developer.android.com. (2018). Settings.Secure | Android Developers. [online] Available at: https://developer.android.com/reference/android/provider/Settings.Secure.html [Accessed 8 Mar. 2018].

[42] SearchSalesforce. (2018). What is personalization? - Definition from WhatIs.com. [online] Available at: http://searchsalesforce.techtarget.com/definition/personalization [Accessed 22 Mar. 2018].

[43] Blog.instabug.com. (2018). [online] Available at: https://blog.instabug.com/2017/12/game-engines/ [Accessed 22 Mar. 2018].

BIBLIOGRAPHY

[1] Liang, Y. (2015). Intro to Java Programming. 10th ed. New York: Pearson Education Limited.

[2] DiMarzio, J. (n.d.). Beginning Android programming with Android Studio. 4th ed. Indianapolis, Indiana: John Wiley & Sons, Inc.

[3] Lehtimäki, J. (2013). Smashing Android UI. 1st ed. Chichester: John Wiley & Sons.

[4] Java. The complete reference. (2017). 10th ed. McGraw-Hill.

[5] Taylor, A. (1997) The JDBC developer's resource: Database programming on the Internet, Prentice Hall US, Englewood Cliffs, New Jersey, USA.

[6] Developer.android.com. (2018). Android Developers. [online] Available at: https://developer.android.com [Accessed 8 Mar. 2018].

[7] W3schools.com. (2018). W3Schools Online Web Tutorials. [online] Available at: https://www.w3schools.com [Accessed 8 Mar. 2018].

[8] Oracle.com. (2018). Java SE Documentation - APIs & Documentation. [online] Available at: http://www.oracle.com/technetwork/java/javase/documentation/api-jsp-136079.html [Accessed 22 Mar. 2018].

[9] Developer.apple.com. (2018). Apple Developer. [online] Available at: https://developer.apple.com/ [Accessed 22 Mar. 2018].

[10] Flaticon. (2018). Flaticon, the largest database of free vector icons. [online] Available at: https://www.flaticon.com/ [Accessed 22 Mar. 2018].