UNIVERSITY OF PÉCS

FACULTY OF SCIENCES

INSTITUTE OF MATHEMATICS AND INFORMATICS

ONLINE ORDERING SYSTEM BASED ON ANDROID

Internal Supervisor:
Dr. ZOLTÁN HORVÁTH
Instructor

Author:
YUAN CHENGE
Computer Science BSc

PÉCS, 2019

„It Is Not the Strongest of the Species that
Survives But the Most Adaptable”
(Darwin, 1859)

TABLE OF CONTENTS

Design of online ordering system based on Android

Abstract: For the catering industry, the order is an essential part. With the development of the catering industry and the popularization of computers, there has been a lot of innovation in restaurant sales. There is plenty of takeaway software on the market, but very little software available to order directly in the store. Hungary Pécs is a beautiful university town. There are international students from all over the world as well as tourists visit the restaurant to eat. This has become a very important thing however due to the language barrier and huge cultural differences. Tourists often face problem understanding a foreign language menu or communicate with the waiter for special requests. Worst case scenario ordering the wrong dish because of miscommunication.

This paper will be aimed at the restaurant industry in Pécs to provide a new way of ordering food. The reason for doing this is because the local people mostly still uses paper recipes. That is why I have decided to use the advanced technology of Android to create an online ordering system. Once the user completes the login procedure they will be able to see the detailed classification of the dishes. Consumers can search on the category they need, find their favorite food, and complete the reservation. By ordering it online, not only saves more time but also the other processes to save the restaurant labor. Allow the customers to be more precise, easy to find their own needs. This Application will achieve the menu visualization with automatic addition of shopping cart and other functions so that ordering more humanized that will make people with different language customs can also find delicious food in foreign lands that suit them.

Keywords: online ordering; Java; Android; Firebase

I. Introduction

Pecs is a town surrounded by the university. People from all over the world came here to study. Its beautiful scenery also attracts a lot of tourists. That is why by nature there will be some language cultural practices problems among the food industry. According to the survey in Pecs such as KFC and McDonald's fast-food restaurants have photo ordering guidelines. The rest of the restaurants are all text menus. While during the ordering process, something happened for instances customers aren’t able to communicate with the waiter. Sometimes it was very difficult to book a table before or during public holidays. At present, the Hungarian market does not have the proper tools for online ordering in the physical store of the application software.

Online ordering Mobile Application will focus on solving the confusion of foreign students in choosing their need of food. Besides that, it does not require direct communication between customers and waiters. Also, customers can directly place orders through the mobile phone Application therefore they can intuitively observe the dishes on the plate as well as the price. Psychologists once said that by looking at the food image will strongly increase one’s appetite and can be more conducive to sales. Stores can also reduce manpower. This Application can bring anonymous convenience for people's dining methods.

Traditional restaurant services were usually served by a waiter providing a paper food menu. Waiter waits for customers to order and writes down the content of the meal then manually input them to the computer equipment. Furthermore it delivers to the kitchen or through computer automatically transferred to the kitchen terminal. In the end, the waiter settled the total amount. However, when the human traffic in the restaurant is overpopulated appeared to be very packed and occupied, which will seriously affect the quality of service.

This ordering system stores account information in a database, which forms a menu after the account has finished ordering including the details of the dishes and the total price making the ordering process more visible.

The Application will use the Android operating system. According to the survey, Android operating system has taken part in more than 80% mobile devices on the market, iPhone, accounting for 18%, and even so, Android's market share is growing aggressively (Jkielty,2018).And all kinds of Applications are also driven by the development of the mobile phone industry were more widely used.

Smartphones are the most popular, especially for those who study or travel abroad smartphones can be called an inevitable existence, so the online ordering Application has a wide range of applicability.

1. Requirements Analysis

In terms of market's demand, we should fully understand the relationship between industry consumers and catering suppliers in the process of online ordering. By obtaining the real needs of consumers as well as understanding the main key point human faces, combine the nature and function of catering providers, and design software development ideas through reasonable analysis and summary.

In terms of technical requirements, the construction of a mobile phone application needs to have in-depth programming knowledge and design knowledge. In the process of independent development of the application, it is necessary to define the market demand, organize the development ideas, according to the Android architecture, computer into the principle and other knowledge to achieve environmental construction, design the Android system to meet the practicality, flexibility, scalability as well as maintainability.

In terms of implementing the requirements, creating an application that can really be used for people's living style, after implementing the market, design requirements, the interface and functionality need to be reviewed, tested, so that it is simple, flexible and easy to operate suitable for most consumers.

2. Research goals

The main purpose of this thesis is to expound the whole design of Android-based online ordering system in the production process, after analyzing the market and its demand degree, consumers and service providers of the main pain points, combined with modern Android technology, to design a mobile phone APP that can facilitate life.

This software mainly has the following functions:

User registration & login.

Change password.

Find the right dish for you based on the classification of the dishes.

Order online.

Displays the total price.

Log Out.

3. Existence Value:

For consumers: diagram menus to reduce the dilemma of not seeing menus. Reduce queuing time and make the whole point of meal transparent.

For restaurant: Reduce staff costs, but improve the efficiency of ordering. Convenient and quick also improve the restaurant change table rate.

This is a typical O2O(online-to-offline)e-commerce model, where stores generate their stores free of charge on the app platform, will be their own business information, commodity information transmission to consumers, consumers to screen online, choose their own needs of services, and online consumer verification and consumer experience, so not only to meet the individual needs of consumers but also through the network to spread the hotel information more widely and farther, Can instantly gather a more powerful consumer capacity to meet the needs of consumers and businesses on both sides.

II. Introduction of development environment and tools

This chapter will introduce the environment required by this application and the language of the corresponding tools required. Understanding the environment and tools is the first and most important step to start the whole application design.

1. System development Environment:

(1). Personal Computer Huawei Matebook D

(2). Development environment: Java JDK 1.8, Windows 10

(3). Development tools: Android Studio, Firebase, DB Browser For SQLite

(4). Operating Environment: Android Platform

2. Android System

2.1 Development

Android is a Linux-based free and open source operating system, Andy Rubin founded Android in 2003, Google acquired Android and its team in 2005. In 2007 Google showcased the Android operating system, Android's source code is then published as an authorization for Apache's free open source license (Burnette, E. 2015). Android system is mainly used in mobile devices, the first Android phone was released in October 2008, and then Android gradually expanded to tablets and other areas, such as television, wearable devices, digital cameras, game consoles and so on.

As the Android version continues to be updated, its functionality continues to improve. Android1.0, released since September 2008, has grown to 10.0. In the process, Android ushered in one milestone after another, with the Android system surpassing Windows (37.91%) for the first time in March 2017 with a 37.93% market share. It has become the first operating system with a global share. And the further spread of mobile devices will continue to drive the development of Android (Allen, S., Graupera, V., & Lundrigan, L. 2010).

2.2 Architecture

Android systems consist of operating systems, middleware, user interfaces, and application software (Ableson, F. 2019). The architecture of the Android system is like other operating systems uses a hierarchical architecture. From the high level to the bottom layer are the application layer, the application framework layer, the system runtime layer and the kernel layer of Linux.

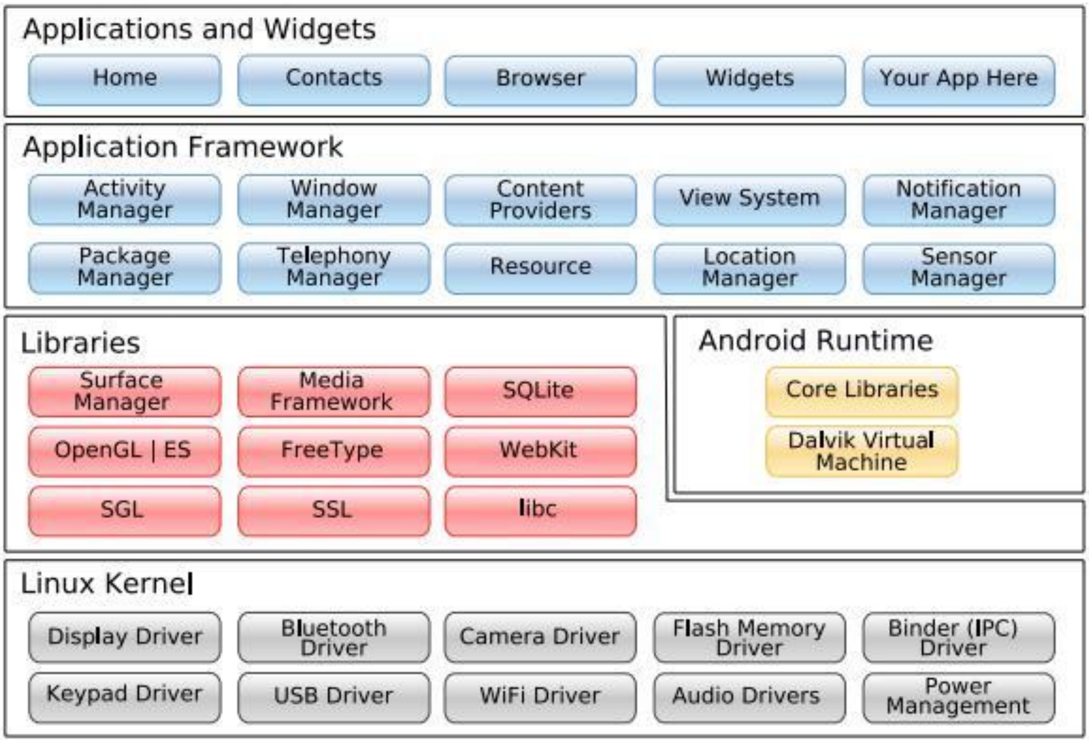


Figure 2-1 Android System architecture diagram (Ed Burnette.2019)

Application layer: All applications are written using JAVA, which includes clients, SMS short message programs, calendars, maps and browsers.

Application framework: Developers have full access to the API (application programming interface) framework used by the core application. Any application can publish its function block and any other application can use the function block it publishes. Hidden behind each app is a series of services and systems, including View, content provider, Resource Manager, Notification Manager, Activity Manager.

System Runtime: Android contains a number of C/C + + libraries that can be used by different components in the Android system.

Linux kernel layer: Android is run on the Linux kernel, but not GNU/Linux.3.2.3 Android

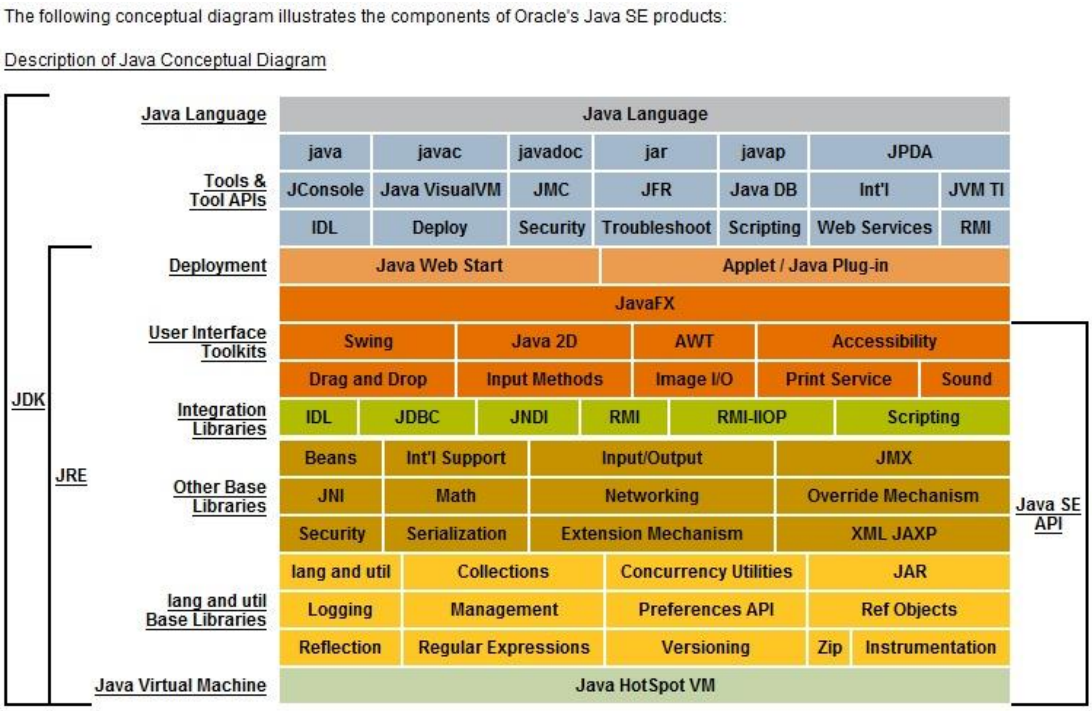
App Components

The four main components of Android development are: Activity: For performance functions. Service: Run the service in the background and do not provide interface rendering. Broadcast Receiver: Used to receive broadcasts. Content provider: supports storing and reading data in multiple apps, equivalent to a database Provider.

3. Java Technology Introduction

Java is the universal programming language of Android. It can run on a Virtual Machine (VM) so no need to recompile for different phones (Simon,c.2019), better security, many development tools are available for Java, Google provides a Java Application Programming Interface (API) to get started and compiles your files into classes (Shweta Shetye.2014). Though Google provides the Java API, Android does not use JAVA Virtual Machine (JVM) to execute class files. Rather, it uses Dalvik Virtual Machine (DVM). The class files are compiled into Dalvik Executable (DEX) format, and bundled as Android Package (APK) along with other resources (Wieruch, R. 2019).

JDK: Java Development Kit, a development Kit needed to write Java programs in Java language (Maresco, J. 2019).Since the launch of Java, JDK has become the most widely used Java SDK, and JDK contains JAVA Runtime Environment (JRE), compiling Java source compiler Javac, also contains some Java program debugging and analysis tools, such as Jconsole, Jvisualvm And so on, as well as Java the documentation required for the program to be written and demo example program (Burnette, 2010).



The following conceptual diagram illustrates the components of Oracle's Java SE products:

Description of Java Conceptual Diagram

JDK
JRE
Java SE API

Java Language: Java Language

Tools & Tool APIs:
java | javac | javadoc | jar | javap | JPDA
JConsole | Java VisualVM | JMC | JFR | Java DB | Int'l | JVM TI
IDL | Deploy | Security | Troubleshoot | Scripting | Web Services | RMI

Deployment: Java Web Start | Applet / Java Plug-in

User Interface Toolkits:
JavaFX
Swing | Java 2D | AWT | Accessibility
Drag and Drop | Input Methods | Image I/O | Print Service | Sound

Integration Libraries: IDL | JDBC | JNDI | RMI | RMI-IIOP | Scripting

Other Base Libraries:
Beans | Int'l Support | Input/Output | JMX
JNI | Math | Networking | Override Mechanism
Security | Serialization | Extension Mechanism | XML JAXP

lang and util Base Libraries:
lang and util | Collections | Concurrency Utilities | JAR
Logging | Management | Preferences API | Ref Objects
Reflection | Regular Expressions | Versioning | Zip | Instrumentation

Java Virtual Machine: Java HotSpot VM

Figure 2-2Java SE structure diagram (Programering.com)

4. XML

XML was created as a standard way to encode data from a heap of Internet-based applications, and it is a structured markup language (Harold, E. and Means, W. 2004). Because it allows information to be passed between devices in a way that can be understood consistently, in the Android World, developers use XML to create layouts as the underlying UI definition for Android applications (JD Maresco). XML can also be written to modify Java code for layout elements after the application runs, so mastering XML is an important skill in Android development (Harold, E. and Means, W. 2004).

5. JSON

JSON: JavaScript Object Notation is a data exchange format, firebase Real-time database data use JSON Format to storage (Kleppmann,2017).It is based on JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999A subset. JSON Not only easy for people to read and write, but also easy been parsed and generated by the computer, The JSON file used in this paper is ObjGen online JSON written by the authoring tool.

6. Android studio

Android Studio is an Integrated Development Environment (IDE). Google launched, is dedicated to Android development of the IDE (Smyth, N. 2018). Built on the IntelliJ IED, it provides integrated Android development tools that provide excellent out-of-the-box support for many common Android SDKs. Compared to other Android development tools such as Eclipse, Android Studio is faster, the User Interface (UI) is more beautiful, smarter, and Gradle combines the benefits of Ant and Maven for faster and easier configuration, packaging, compilation function (Darwin, I.2017). UI editor comes with multi-device real-time preview function, built-in terminal, support a variety of plug-ins, perfect integration version control functions and so on.

Android SDK: SDK tools represents the software development kit. The Android SDK is a Java code module that gives developers access to devise features such as cameras and acceleration needles (Gookin, D. 2016). A key build of the Android SDK is the library called Gradle.

Gradle basic introduction: Android studio project has a setting gradle file on the entire project, It contains references to all modules in the project, and it is automatically updated every time a new module is imported or created (Ben, S. (2015). Each Android studio module has its own build.gradle file, and if one module up dependent on another module you need to add this dependency to the build grandle file in the dependent section. When the module requires jar text or remote dependencies, it also needs to be added to the build grandle file.

7. Firebase

Firebase is a platform for providing real-time data storage and synchronization, building mobile apps, user body authentication, cloud messaging, remote configuration, and more, and is a back-end service provided by Google that we can use to Integrated into Android, IOS, web and other applications (Rouse, A. 2019) .Designed specifically for developers, the Firebase has deployed back-end in the cloud, lighter, uses cloud capabilities, builds servers for applications, and can use firebase products to monitor performance and view crash reports.

Firebase Real-time database: Use the NoSQL Cloud database to store and synchronize data, using the Android SDK, IOS SDK, JAVASCRIPT SDK when building cross-platform apps, all users share a database and data you can synchronize to all clients in real time, even if your app is offline and still available for use.

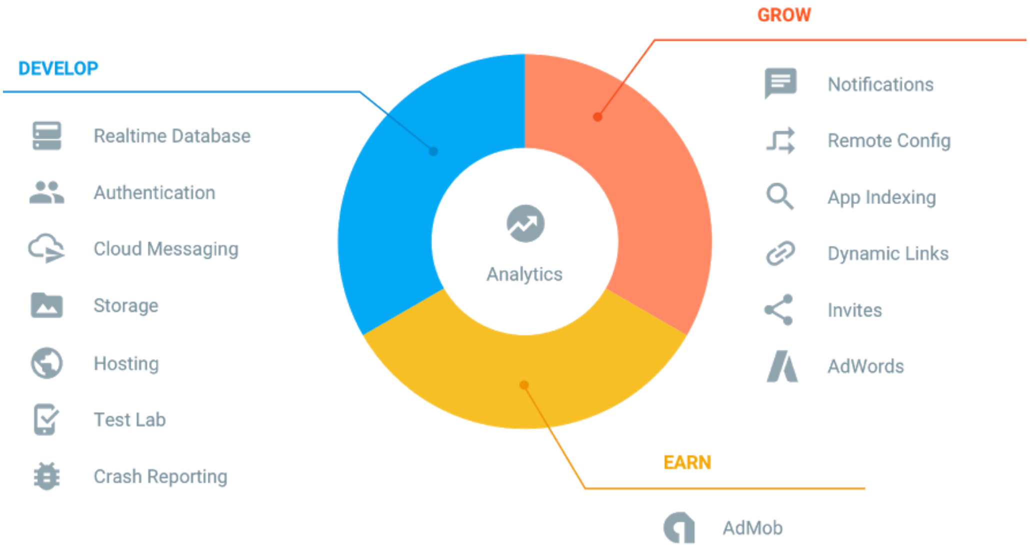


Figure 2-3 Firebase Feature Overview diagram (Alex Rouse.2016)

III. Feature introduction and design details

1.Process Analysis

Before designing the application, the required process should be taken into consideration firstly. Online ordering will include the process of ordering by users and the process of dish supply. Based on this analysis, this section will specifically describe the online ordering process and the entire process from ordering to eating.

1.1 User Registration Login

Registration: The user registers as a new user through the Mobile client main interface, which is stored in Firebase database and prompts for successful registration on the phone client.

Login: Login is divided into two situations, customer login, operator login. After entering the account number and password, the server will match the data according to the selected information, the user name and password are correct to enter the application interface.

Forget password: This function is for someone if he forgets the password. In the Sign In interface, there is a button “Forget password?”, user can click it and write the name and phone number to verify if all information matches the database information then a new page for creating a new password appears.

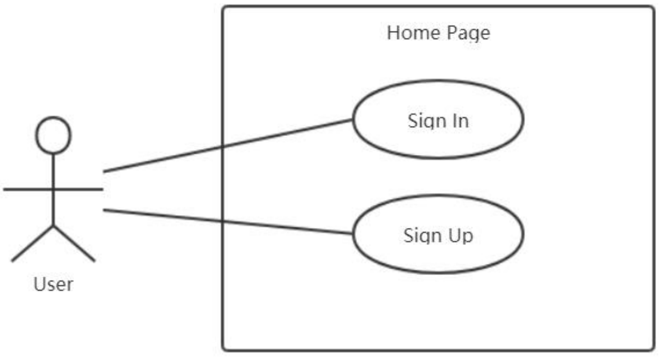


Figure 3-1 User login process (own source)

1.2 Application Interface

The application interface for client provides information about restaurant food, customers enter the application interface, by browsing food information, to choose their favorite food, at this time, customers can choose to arrive at the restaurant before ordering, or can choose to book in advance, And can give the restaurant a message about their expected arriving time, in the end, it will generate the corresponding order. Before leaving the restaurant, customers only need to show the waiter their order code for checkout.

After the customer orders, the restaurant will receive the appropriate prompts and orders, and finally according to the order to charge customers.

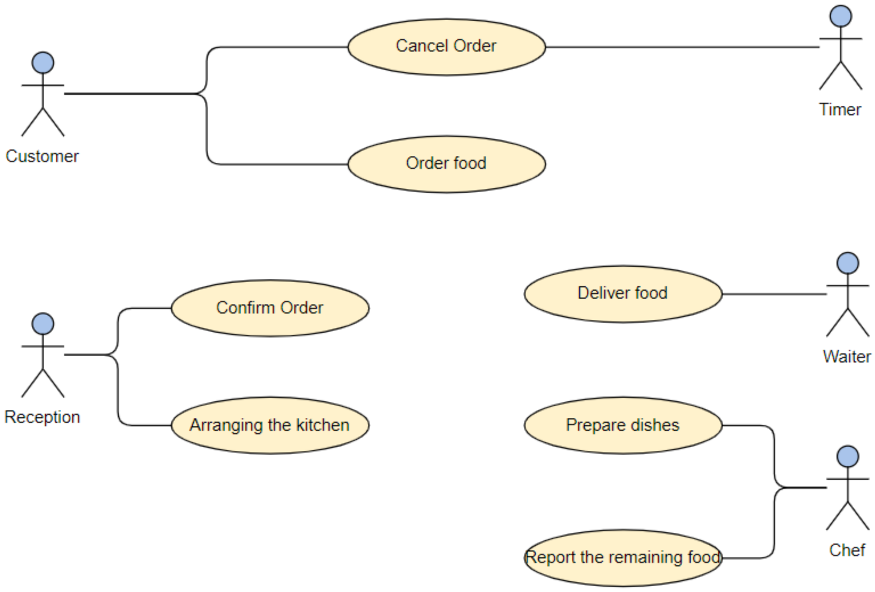


Figure 3-2 Functional Flow (own source)

1.3 Specific process

After the customer orders and has arrived at the restaurant, the waiter will need to provide the menu information to the rear chef.

The following picture shows the entire activity process including customer service staff and chefs.

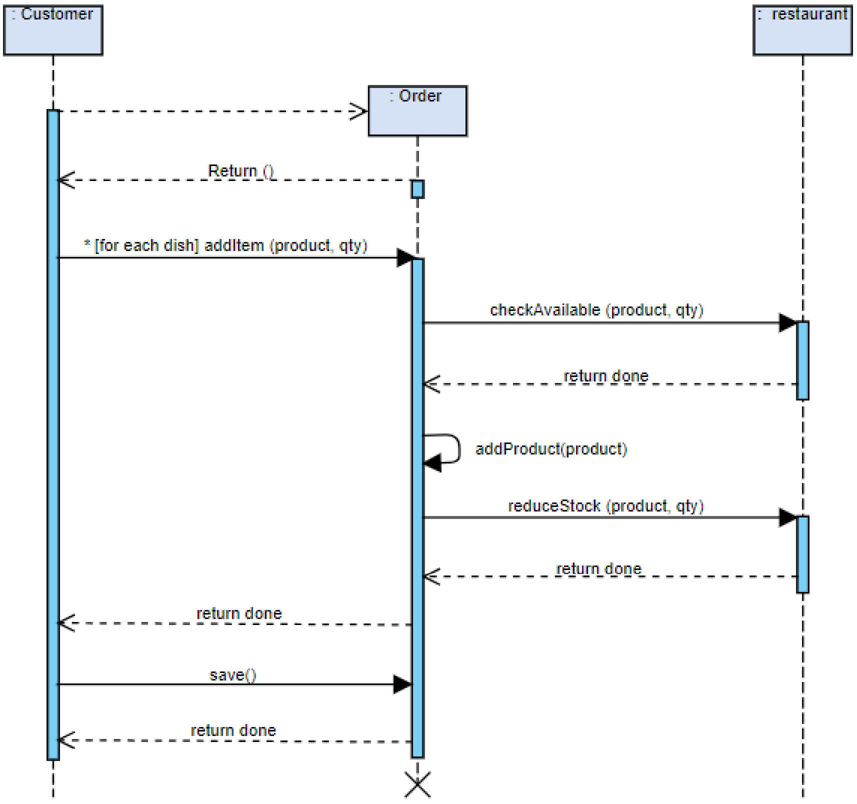


Figure 3-3 Operation Flow (own source)

1.4 Client and server physical architecture

The whole program is composed of client and server. The client is based on the Android system. The server is supported by Firebase. The client is responsible for collecting information and displaying interface. The user input message will be updated to the server and stored in the database. The server provides data downloading and data uploading interfaces to the client to exchange data, and the administrator can delete, update, and control the data.

1.5 Use case conception

User Login	
Participants	All Users
Trigger condition	User clicks the SignIn button
Prefix condition	The user opens the APP and enters the main page
Normal process	1.Open the APP 2.Click the Sign In button on the home page 3.Enter user information
Expand the process	1.When the user does not exist, it is a reminder that the user account does not exist 2.Prompt user password input incorrectly when password is incorrect

Table 3-1 User Login

User Registration	
Participants	All Users
Trigger condition	User clicks the Sign up button
Prefix condition	The user opens the APP and enters the main page
Normal process	1.Click the SignUp button 2.Enter your phone number, username, password 3. Registration is successful, return to the main page
Expand the process	The account number already exists to remind the user that the account already exists

Table 3-2 User Registration

Retrieve password	
Participants	All Users
Trigger condition	Users forget their passwords and want to get them back
Prefix condition	The user enters the login screen, finds the forgotten password and clicks
Normal process	1. Click Forgot Password 2. Enter your account number and username 3. Verify that the information is correct and enter a new password 4. Password Change succeeded
Expand the process	1. Account does not exist, remind the user to enter the correct account number 2. Reminder account number and username do not match

Table 3-3 Password Retrieval

Go to Restaurant homepage	
Participants	All user
Prefix condition	User successfully logged in
Normal process	1. User click the sign in button 2. Write the correct account and successful log in
Expand the process	1. Search the food type 2. Check the shopping cart 3. Jump to food detail page

Table 3-4 Access to the home page

Add to Shopping Cart	
Participants	All user
Prefix condition	Click on the shopping Cart logo
Normal process	1. Click on the Food details 2. Add to Shopping Cart 3. Click the Total Price 4. Enter the seat number, number, special request
Expand the process	Display the total price when the user completes the order

Table 3-5 Add to Shopping cart

Log Out	
Participants	All user
Prefix condition	Click on the log out button
Normal process	1. Find the exit button in the navigation bar 2. Exit the app

Table 3-6 User log out

2. Construction of the environment

2.1 Tools Download:

Because the system uses the JDK when using Android studio, downloading and configuring it is the first step in building a development environment.

Configuration Process: This PC property--advanced system settings--environment variables-system variables-new-variable name fill (java_home), variable value for JDK installation path-ok-path-edit text-add (% java_home \bin%)-OK. (window10).

2.2 Add Firebase to the Android app

(1). Visit the Firebase console, add items, and enter the project name.

(2). Modify the project ID and click Create Project.

(3). Add Firebase to the Android app (at which point you need to register the app, SHA-1 the fingerprint from the Android item Gradle project, the run Configurations found). Complete the Registration app.

(4). Add the Firebase Android profile (Google-services.json) to the same directory as the Build.gradle file (Eathere) at the root level.

(5). Add the initialization code.

Add the appropriate libraries to build.grandle (module) and build.grandle (Project) based on the firebase features you need.

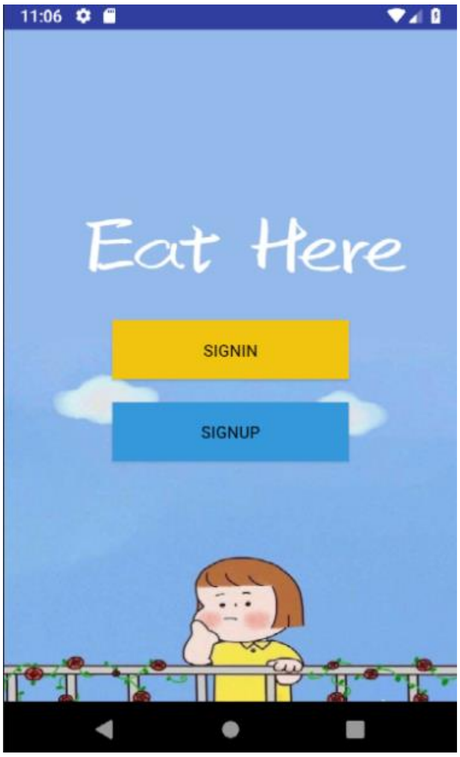
This completes the construction of the whole basic environment.

3. Realization of user login interface

User login is a total of 3 interfaces to achieve, namely: the main interface, Login interface, and registration interface.

3.1 Welcome interface

After entering the application, the first interface is the welcome screen, which contains the name of the restaurant and two buttons. New users need to register and users with existing accounts can choose to log in directly. The welcome page is shown in picture 3-1.



Picture 3-1 Welcome Interface (own source)

Key code for the Welcome interface:

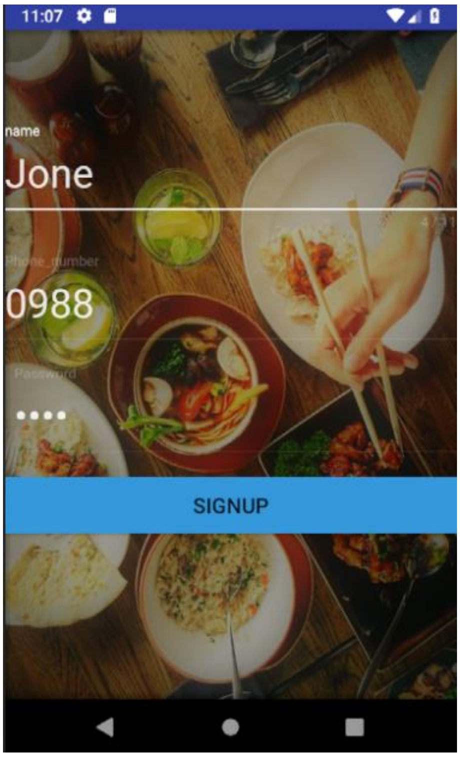
```
//Click the button to make the page jump
Protected void onCreate (Bundle savedInstanceState) {
Super.onCreate (savedInstanceState);
SetContentView (R.layout.activity_main);

BtnSignIn = (Button) findViewById (R.id.btnSignIn);
BtnSignUp = (Button) findViewById (R.id.btnSignUp);

BtnSignUp.setOnClickListener (New View.OnClickListener () {
@Override
Public void onClick {
Intent signUp = new Intent (MainActivity.this, SignUp.class);
StartActivity (signUp);
    }
```

3.2 Registration interface

The registration interface includes a total of 4 items: username, mobile phone number, password, registration button. As shown in Picture 3-2:



name

Jone

Phone_number

0988

Password

••••

Picture 3-2 Registration Interface (own source)

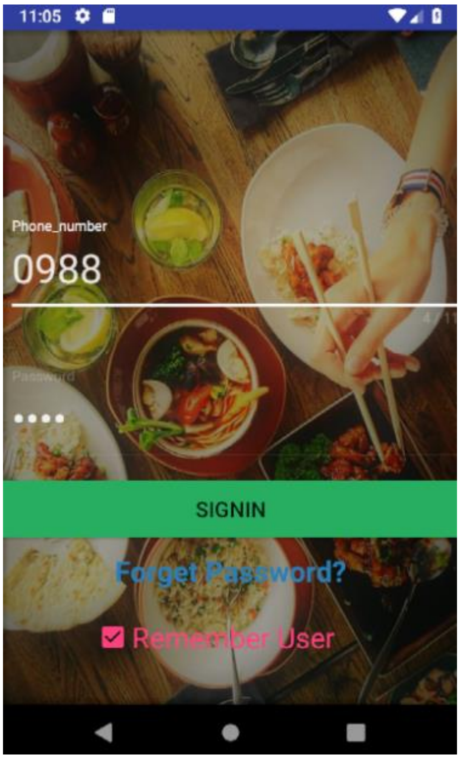
Key code for the Registration interface:

```
    table_user.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            //Check the user in database
            if (dataSnapshot.child(edtPhone.getText().toString()).exists()) {
                mDialog.dismiss();
                Toast.makeText(SignUp.this, "Phone Number already registered",
    //if it is the existing accounts then gave a infor.
                 Toast.LENGTH_SHORT).show();
    } else {
        mDialog.dismiss();
        User user = new User(edtName.getText().toString(), edtPassword.getText().toString());
        table_user.child(edtPhone.getText().toString()).setValue(user);
        Toast.makeText(SignUp.this, "Sign up successfully", Toast.LENGTH_SHORT).show();
        finish();
    }
}
```

3.3 Login interface

The login interface includes five elements. The user enters the account phone number and password and clicks the login button to log in. When the user forgets the password, he can click 'forget the password? 'enter the password remaking interface, the last element can realize the function of remembering the user. By default, it is to remember the account.

When it is opened again, the user can cancel the check and cancel the function of automatic login. As shown in Picture 3-3:



Phone_number
0988

Password
••••

☑ Remember User

Picture 3-3 Login Interface (own source)

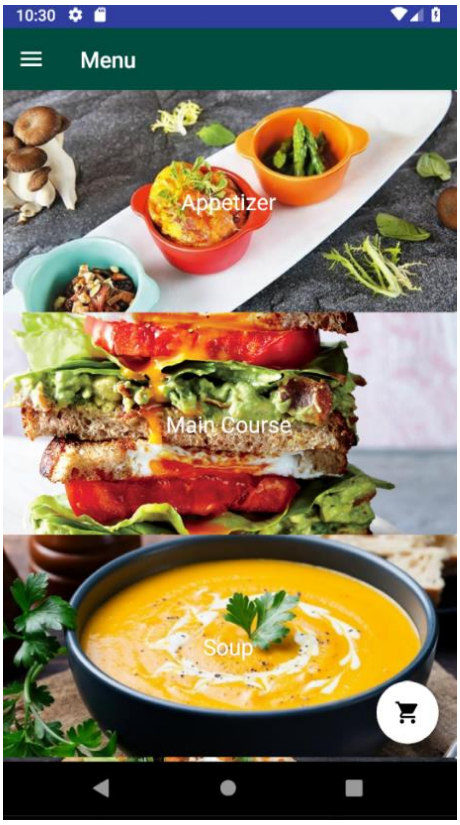
Key code for the Login interface:

```
public void onDataChange(DataSnapshot dataSnapshot) {
        if (dataSnapshot.child(edtPhone.getText().toString()).exists()) {
                mDialog.dismiss();
                User user =dataSnapshot.child(edtPhone.getText().toString()).getValue(User.class);
                user.setPhone(edtPhone.getText().toString());
            if(user.getPassword().equals(edtPassword.getText().toString()))
{Toast.makeText(SignIn.this, "Sign in successful!", Toast.LENGTH_SHORT).show()；
```

4. Implementation of the application interface

4.1 Main menu browsing

After logging in, the user will enter the application interface. The main application interface is menu navigation, which is mainly divided into five categories: appetizer, main course, soup, dessert and drink. As shown in Picture 3-4:



Picture 3-4 Main Face-Menu Navigation (own source)

After completing the UI design, the code is written through JAVA to implement the corresponding functions. As shown in the following figure:

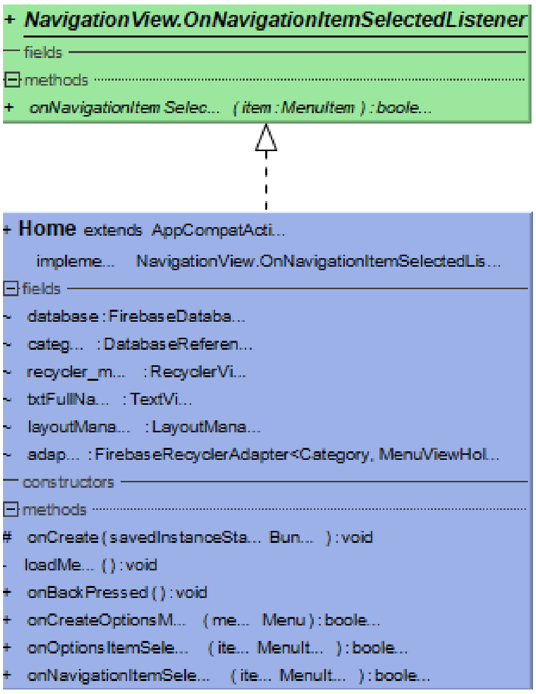

+ NavigationView.OnNavigationItemSelectedListener
fields
methods
+ onNavigationItemSelec... (item : MenuItem) : boole...
+ Home extends AppCompatActi...
impleme... NavigationView.OnNavigationItemSelectedLis...
fields
~ database : FirebaseDataba...
~ categ... : DatabaseReferen...
~ recycler_m... : RecyclerVi...
~ txtFullNa... : TextVi...
~ layoutMana... : LayoutMana...
~ adap... : FirebaseRecyclerAdapter<Category, MenuViewHol...
constructors
methods
# onCreate (savedInstanceSta... Bun...) : void
- loadMe... ( ) : void
+ onBackPressed ( ) : void
+ onCreateOptionsM... (me... Menu) : boole...
+ onOptionsItemSele... (ite... MenuIt...) : boole...
+ onNavigationItemSele... (ite... MenuIt...) : boole...

Figure 3-4 UML class Diagram (own source)

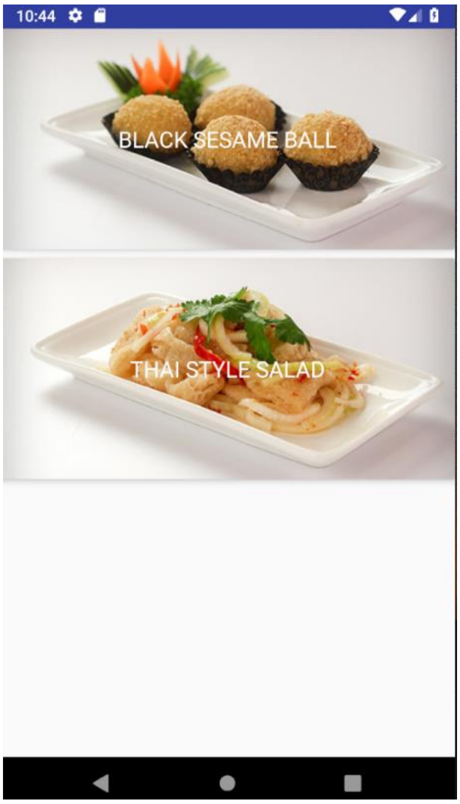
Key code for the Registration interface:

```
private void loadMenu() {
    adapter = new FirebaseRecyclerAdapter<Category,
MenuViewHolder>(Category.class,R.layout.menu_item, MenuViewHolder.class,category) {
        @Override
        protected void populateViewHolder(MenuViewHolder viewHolder, Category model,
int position) {
            viewHolder.txtMenuName.setText(model.getName());
            Picasso.with(getBaseContext()).load(model.getImage())
                    .into(viewHolder.imageView);
            final Category clickItem = model;
            viewHolder.setItemClickListener(new ItemClickListener() {
                @Override
                public void onClick(View view, int position, boolean isLongClick) {
                    //Get CategoryId and send to new Activity
                    Intent foodList = new Intent(Home.this,Foodlist.class);
                    foodList.putExtra("CategoryId",adapter.getRef(position).getKey());
                    startActivity(foodList);

                    //
Toast.makeText(Home.this,""+clickItem.getName(),Toast.LENGTH_SHORT).show();
                }
            });
        }
    };
    recycler_menu.setAdapter(adapter);
```

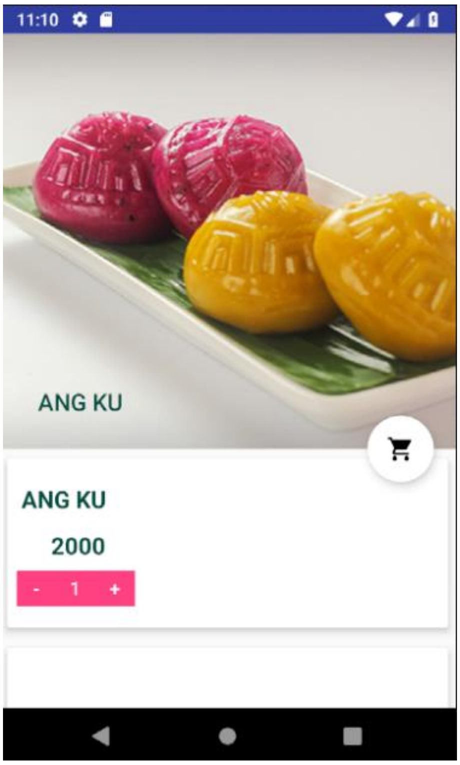
4.2 Detail menu browsing

By browsing the main menu, the user selects the desired dish type. For example, if the former dish is selected, the interface will go to the menu of all the former dishes, so that the customer can choose the one he needs, as shown in figure 4-5. At the same time, customers can click the picture to see details, including dish name, price, ingredients, etc. If they like, they can directly add them to the shopping cart, as shown in Picture 3-5:



Picture 3-5 Detail menu (own source)

This part of the design starts with the layout, RecyclerView used to bind FirebaseUI data to a menu layout. Food_item.xml was created to specify element placement.



Picture 3-6 Shopping Car related Class interface (own source)

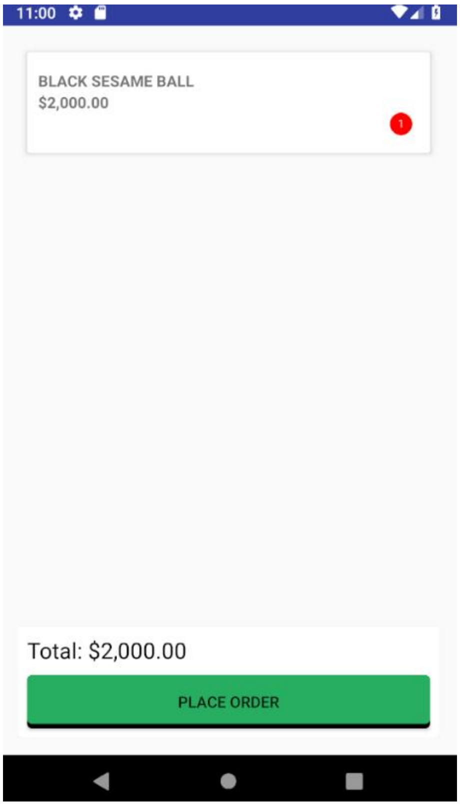
Key code for this part:

```
public MenuViewHolder(View itemView){
    super(itemView);
    txtMenuName = (TextView)itemView.findViewById(R.id.menu_name);
    imageView = (ImageView)itemView.findViewById(R.id.menu_image);
    itemView.setOnClickListener(this);
}

//code for detail menu
private void getDetailFood(String foodId) {
    foods.child(foodId).addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            currentFood = dataSnapshot.getValue(Food.class);
//set image
Picasso.with(getBaseContext()).load(currentFood.getImage()).into(food_image);
                collapsingToolbarLayout.setTitle(currentFood.getName());
                food_price.setText(currentFood.getPrice());
                food_name.setText(currentFood.getName());
                food_description.setText(currentFood.getDescription());
```

4.3 Shopping cart design

After adding food to the shopping cart, the user can return to the main interface and click the shopping cart to see the situation of the shopping cart, as shown in Picture 3-7.

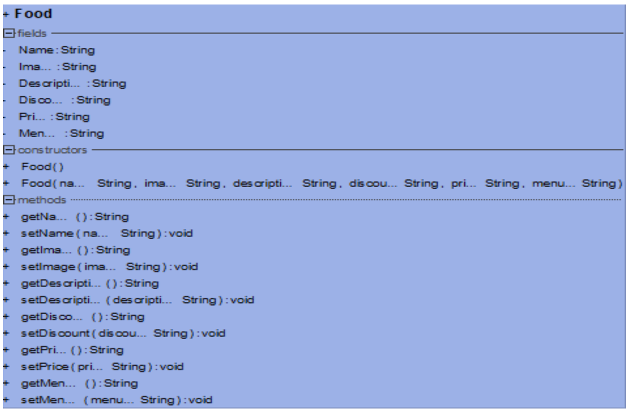


BLACK SESAME BALL
$2,000.00

Total: $2,000.00

Picture 3-7 Shopping cart interface (own source)

In the design of the shopping Cart, 5 classes were created, namely Cart class, cartAdapter class, cartViewHolder class, order class, and database class. You create an SQLite database, store the shopping cart information in it, and pass it to Firebase, which provides the data to the hotel administrator. Figure 3-5 shows the shopping cart class diagram.



```
+ Food
⊟ fields
-  Name : String
-  Ima...  : String
-  Descripti...  : String
-  Disco...  : String
-  Pri... : String
-  Men...  : String
⊟ constructors
+  Food( )
+  Food( na...  String ,  ima...  String ,  descripti...  String ,  discou...  String ,  pri...  String ,  menu...  String )
⊟ methods
+  getNa...  ( ) : String
+  setName ( na...  String ) : void
+  getIma... ( ) : String
+  setImage ( ima...  String ) : void
+  getDescripti... ( ) : String
+  setDescripti... ( descripti...  String ) : void
+  getDisco...  ( ) : String
+  setDiscount ( discou...  String ) : void
+  getPri... ( ) : String
+  setPrice ( pri...  String ) : void
+  getMen...  ( ) : String
+  setMen...  ( menu...  String ) : void
```

Figure 3-5 Menu Related Classes UML Class Diagram (own source)

Key code for this part:

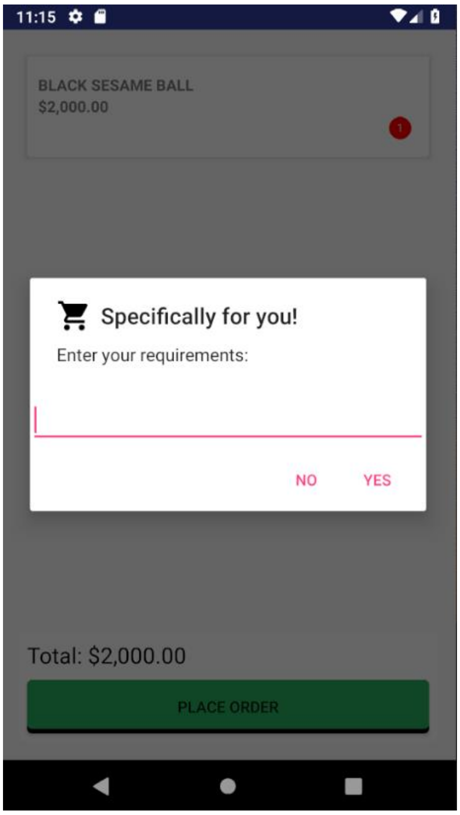
```
public class Database extends SQLiteAssetHelper {
        private static final String DB_NAME="MDB.db";
        private static final int DB_VER=1;
        public Database(Context context) {
```

```
        super(context, DB_NAME,null,DB_VER);
    }
//add the contains of cart to database
public void addToCart(Order order)
{
    SQLiteDatabase db= getReadableDatabase();
    String query = String.format("INSERT INTO
OrderDetail(ProductId,ProductName,Quantity,Price,Discount)VALUES('%s','%s','%s','%s','%s');",
            order.getProductId(),
            order.getProductName(),
            order.getQuantity(),
            order.getPrice(),
            order.getDiscount());
    db.execSQL(query);
}

     //show cart details
private void loadListFood() {
        cart = new Database(this).getCarts();
        adapter = new CartAdapter(cart,this);
        recyclerView.setAdapter(adapter);
        //count the total prices
        int total = 0;
        for(Order order:cart)
          total+=(Integer.parseInt(order.getPrice()))*(Integer.parseInt(order.getQuantity()));
        Locale locale = new Locale("en","US");
        NumberFormat fmt = NumberFormat.getCurrencyInstance(locale);
        txtTotalPrice.setText(fmt.format(total));
    }
}
```

4.4 Adds additional information

Before the user determines the order, some special requirements can be written in, such as when to eat in the store, contraindication, allergy to certain food materials, like to add more Chinese food materials, etc., so as to achieve more humanized service. See Picture 3-8.



11:15

BLACK SESAME BALL
$2,000.00

Specifically for you!

Enter your requirements:

Total: $2,000.00

Picture 3-8 Adds additional information interface (own source)

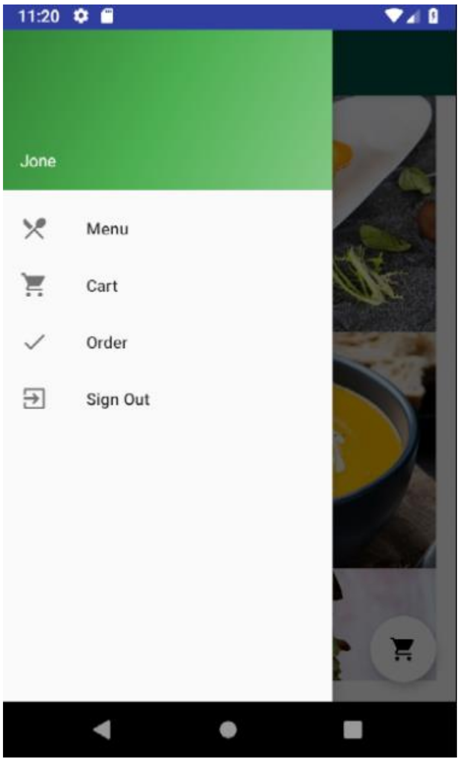
Key code for this part:

```
private void showAlertDialod(){
    AlertDialog.Builder alterDialog = new AlertDialog.Builder(Cart.this);
    alterDialog.setTitle("Specifically for you!");
    alterDialog.setMessage("Enter your requirements: ");

    final EditText edtRequirement = new EditText(Cart.this);
    LinearLayout.LayoutParams Ip = new LinearLayout.LayoutParams(
            LinearLayout.LayoutParams.MATCH_PARENT,
            LinearLayout.LayoutParams.MATCH_PARENT
    );
    edtRequirement.setLayoutParams(Ip);
    alterDialog.setView(edtRequirement);
    alterDialog.setIcon(R.drawable.ic_shopping_cart_black_24dp);
    alterDialog.setPositiveButton("Yes",new DialogInterface.OnClickListener(){
        @Override
                public void onClick(DialogInterface dialogInterface,int i){
        //Create new Request
        //Delate cart
```

4.4 User navigation bar

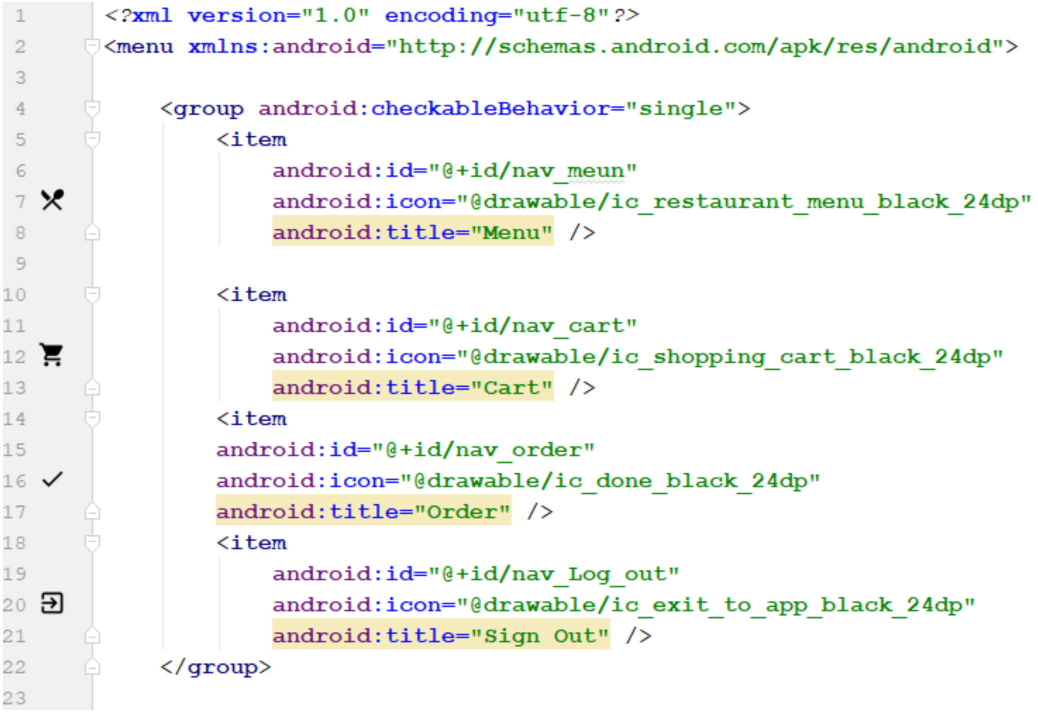
The user navigation bar includes four functions: view menu, view shopping cart, view order and exit application, as shown in Picture 3-9:



Picture 3-9 Navigation Interface (own source)

The first step is to create the menu navigation bar (Home). The purpose of this layout is to show the menu classification list. To enable FirebaseUI to bind data to the menu taxonomy layout, you need to add libraries to your project before you start designing.

Next, create a Navigation drawer activity to make the layout easier to create a list effect. To create "Activity_home. XML", "content_home. XML", "app_bar_home. XML" "nav_header_home. XML". RecyclerView to render menu listings in "content_home.xml". Add the shopping cart icon in "app_bar_home.xml", set the appropriate navigation format in "activity_home.xml", and set the ICONS needed for the navigation page in "nav_header_home.xml".



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_meun"
            android:icon="@drawable/ic_restaurant_menu_black_24dp"
            android:title="Menu" />

        <item
            android:id="@+id/nav_cart"
            android:icon="@drawable/ic_shopping_cart_black_24dp"
            android:title="Cart" />
        <item
        android:id="@+id/nav_order"
        android:icon="@drawable/ic_done_black_24dp"
        android:title="Order" />
        <item
            android:id="@+id/nav_Log_out"
            android:icon="@drawable/ic_exit_to_app_black_24dp"
            android:title="Sign Out" />
    </group>

```

Figure 3-6　XML code of navigation bar(own source)

Click the Menu button to return to the home page, click the shopping cart to view the contents of the shopping cart, click the Order to display the existing Menu, and finally the user can Sign Out the system.

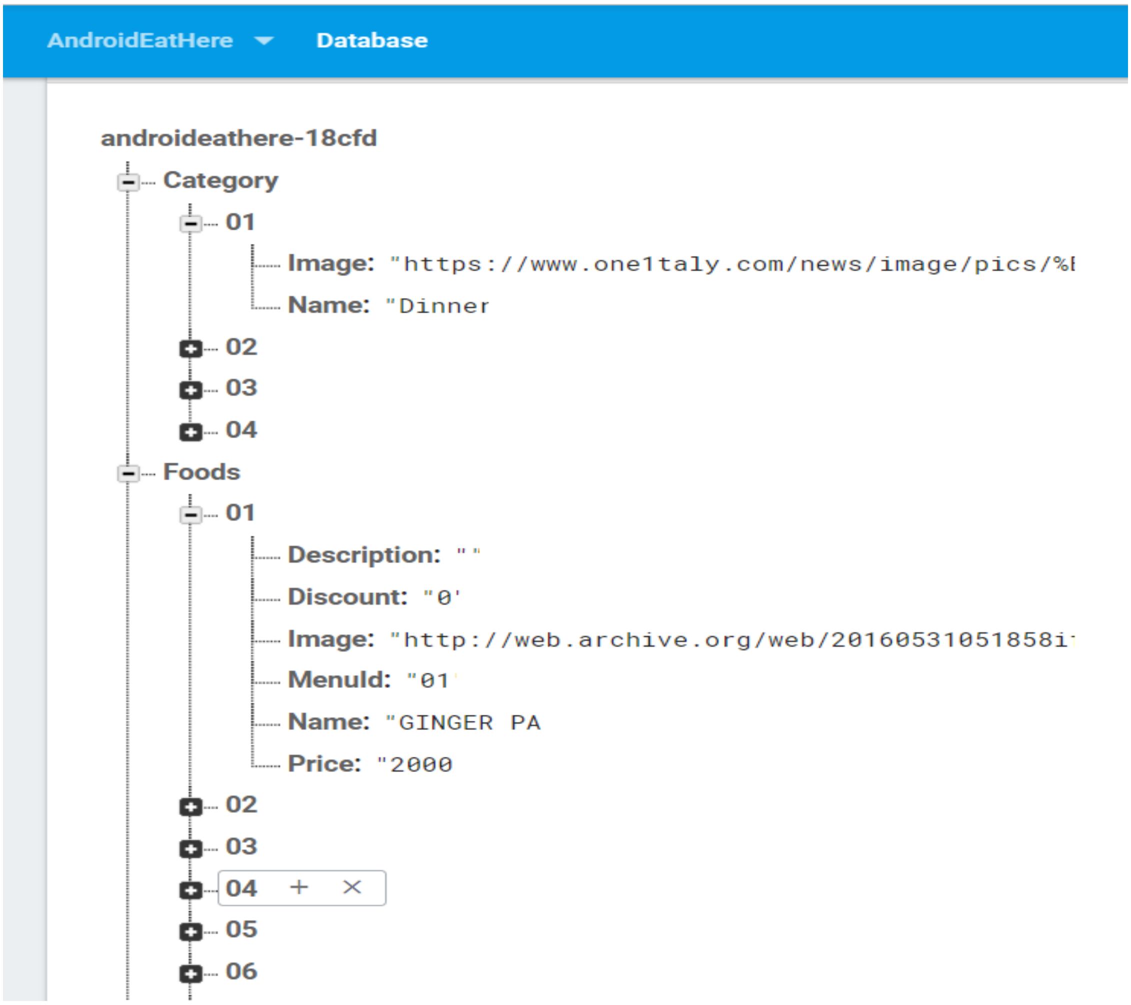
The following is part of the code to implement the navigation bar:

```
DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
        this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
drawer.setDrawerListener(toggle);
toggle.syncState();
NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
navigationView.setNavigationItemSelectedListener(this);
```

5.Database creation

Before designing the UI and Java of the interface, it is necessary to complete the creation of the database. The application interface will realize the functions of menu category navigation, click to jump to the corresponding series of menu, view the menu details and add to the shopping cart.

In the design of JSON, the Category and Foods table are added, and the corresponding dish types are included in the Category. We can use this navigation to enter Foods to see more Foods of the same type. The Category includes the name and picture of the menu Category, and the table of food includes the name, description, picture, discounts and menu ID. In this case, the menu ID is a foreign key, through which the Category can be connected with Foods in the future.



```
androideathere-18cfd
  Category
    01
      Image: "https://www.one1taly.com/news/image/pics/%I
      Name: "Dinner
    02
    03
    04
  Foods
    01
      Description: ""
      Discount: "0'
      Image: "http://web.archive.org/web/20160531051858i
      MenuId: "01
      Name: "GINGER PA
      Price: "2000
    02
    03
    04  +  ×
    05
    06
```

Figure 3-7 JSON tree (own source)

Completed this part of the design, the basic realization of the user login through the Browse menu classification navigation, find the corresponding menu, and enter the function of viewing similar cuisines.

Finally, the ability to add a shopping cart to generate menus. This section will complete the shopping Cart app by storing the menu in the SQLite database. First, add the appropriate libraries to enable Android studio to collaborate with the SQLite database.

This is followed by the creation of an SQLite database, because SQLite takes the dynamic data type, when a value is inserted into the database, SQLite will check its type, if the type does not match the associated column, SQLite will attempt to convert the value to the type of the column, if not converted, The value will be stored as its own type, SQLite calls this "weak type", because of this characteristic of SQLite, this lightweight database is more suitable for mobile applications, so here using SQLite to create a table to store information in the shopping cart, this table contains ID, ProductId, ProductName, Price, Discount, Quantity these 6 elements. Finally, the new Database.class implements the operation of the database by inheriting SQLiteAssetHelper. The application uses the DB Browser for SQLite Open source IED to create an MDB.db file. As shown in the following figure:

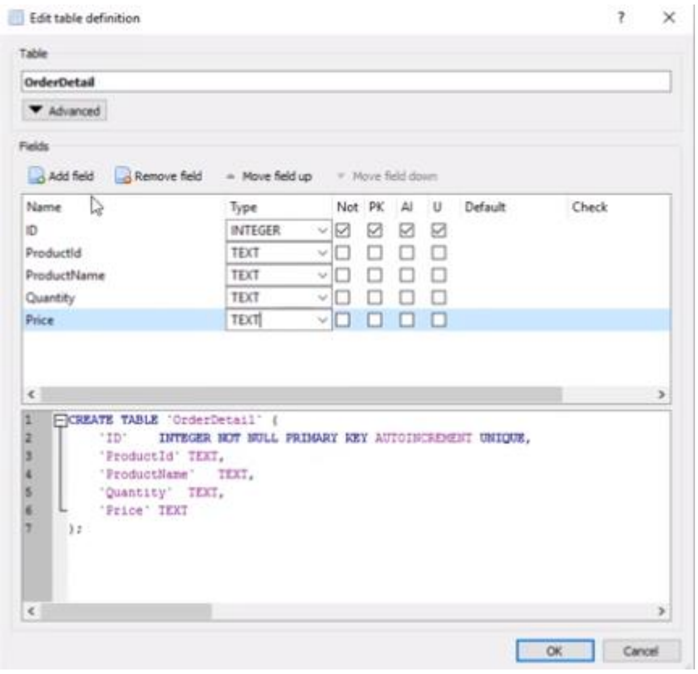


Table: OrderDetail

Fields

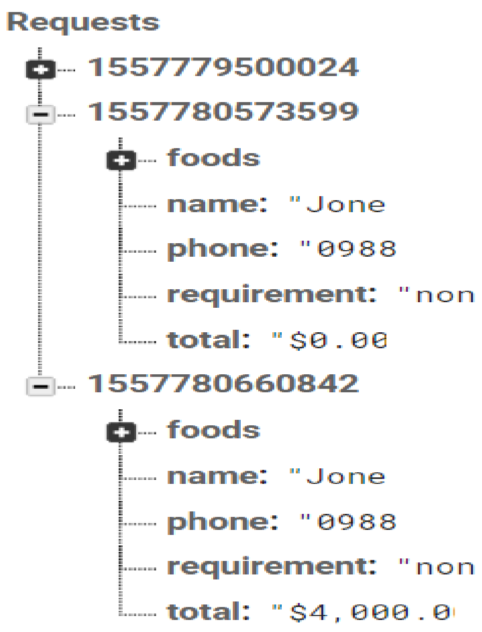
Name	Type	Not	PK	AI	U	Default	Check
ID	INTEGER	☑	☑	☑	☑		
ProductId	TEXT	□	□	□	□		
ProductName	TEXT	□	□	□	□		
Quantity	TEXT	□	□	□	□		
Price	TEXT	□	□	□	□		

```
CREATE TABLE `OrderDetail` (
	`ID`	INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
	`ProductId`	TEXT,
	`ProductName`	TEXT,
	`Quantity`	TEXT,
	`Price`	TEXT
);
```

Figure 3-8 SQlite Database Creation Diagram (own source)

6.Administrator view

Administrators can update products information in the Firebase Json file, because we are using the real-time database, so when a user to update the data content, administrators can also direct access to information, so the administrator need to view the part Requests for customer order information at any time, and in the checkout line according to the corresponding orders and customer orders, right to collect progress payment. As shown in figure 3-15.



```
Requests
+ 1557779500024
- 1557780573599
    + foods
      name: "Jone
      phone: "0988
      requirement: "non
      total: "$0.00
- 1557780660842
    + foods
      name: "Jone
      phone: "0988
      requirement: "non
      total: "$4,000.0
```

Figure 3-9 The order view visible to the administrator (own source)

7. Testing and improvement

Tests are required in all Android projects to ensure the robustness and stability of the system, followed by an introduction to the testing section.

Performance testing: performance test dimensions include smoothness, crash, memory, traffic, response time, power consumption, and CPU. As the content in the application continues to enrich and grow, the importance of performance testing will continue to emerge.

Use the LeakCanary Open source framework in this project to port it to the project to detect memory leaks.

In the detection process there is some paging white screen phenomenon, and finally used can be added to the flash screen to ease. There is also a situation where the plugin is not responding after continuously reading the logs and establishing the unit test the bug is found so that fixed the problem.

Security testing: an application that involves shopping, user security is very important. In the test link for software permissions, data permissions carried out the corresponding test checks, found that at this stage there is no leak rights or illegal access, there will be no self-startup, bundled with another software phenomenon, data encryption normal, no risk of leakage.

Crossover time test: when the application runs by other time interference, such as answering the phone, viewing other software information and so on will often lead to crashes or data loss and other anomalies, the application through the home key and back key simulation conflicts, interference events appear after the application does not appear data loss, error, terminal crash and other situations, so cross-test passes.

IV. Conclusions and possibilities

This application is for the restaurant's electronic ordering system, in the smartphone more and more popular today, the use of electronic menus is undoubtedly a cost-saving, at any time to update the information, at any time to view the information is also to the customer is convenient. Now there is a lot of taking away related app, we through this app may also be able to achieve the restaurant recipe graphical view, but these do not relieve us of the predicament of ordering when we enter the restaurant, nor can we meet the complete transparency of the order. After a personal experience, all of the relevant applications that exist in Hungary are in the Hungarian language interface, which is still not helpful for people who do not understand the Hungarian language. While this system is an all-English interface, for these foreign studies, tourism people are more usability.

In the design of this article, a series of software, such as Android Studio, completed the basic framework of the Pecs's online ordering system. But no software can exist in perfect form, and a good app needs to be constantly optimized and improved. To achieve greater humanization and rationalization is also the purpose of this application and the direction of efforts.

Outlook: in the program design due to time and energy reasons only completed the client part of the app development, Although the realization of the addition of shopping cart and other functions but the overall did not achieve perfection, during the development and testing process, it has also been found that some new features can be extended to achieve better results, and this part will be slowly perfected in subsequent program upgrades.

Specific expectations: at present, this application only applicable to specific restaurants, still lack extensive, hope to add the nearest hotel search function in the future improvement process, to achieve multi-restaurant online preview, the realization of the user evaluation function. Also in this thesis we created the login part is for the future, we can add a wallet application, a client can save their money in there count and they can get more discount, in the paying part, they can use their wallet do need a credit card or cash any more.

Resource issues: this application is currently only applicable to the administrator to add vegetable information, hope that in the future improvement process can be added to the Server part, can be directly by the clerk to complete the corresponding storefront dishes added and updated, The clerk also uploads information such as table information, whether the dishes are available or not.

Two months of software design, from the initial formation of ideas to demand analysis to Android's learning to the final completion of the design prototype, in the process I learned a lot. The writing of application not only makes full use of the content learned in the undergraduate stage but also learns a lot of new knowledge autonomously. Doing software is a process that requires concentration and patience, and even a letter error can lead to system collapse, the constant search for mistakes, continuous improvement can continue to improve.

V. Acknowledgement

Thank you to all the teachers and students who helped me in the production of this graduation project, as well as all the relevant books and the authors of the website. Special thanks to my mentor, guided in the conceptual phase of the initial design, as well as patient guidance throughout the design process.

VI. References

Smyth, N. (2018). *Android Studio 3.2 development essentials*. Payload Media, Inc.

Burnette, E. (2015). *Hello, Android: Introducing Google's Mobile Development Platform* (4th ed.). Pragmatic Bookshelf.

Allen, S., Graupera, V., & Lundrigan, L. (2010). *Pro smartphone cross-platform development: iPhone, Blackberry, Windows Mobile and Android Development and Distribution* (pp. 35-50). New York: Apress.

Ableson, F. (2019). Introduction to Android development. Retrieved from https://developer.ibm.com/articles/os-android-devel/

Simon, C. (2019). Why does Android use Java?. Retrieved from https://stackoverflow.com/questions/3560963/why-does-android-use-java

Rouse, A. (2019). Firebase: The Good, Bad, and the Ugly - RaizException - Raizlabs Developer Blog. Retrieved from https://www.raizlabs.com/dev/2016/12/firebase-case-study/

JDK 1.7 the basic concept and the directory structure. (2019). Retrieved from https://www.programering.com/a/MjM5YDMwATE.html

Burnette, E. (2010). *Hello, Android Introducing Google's Mobile Development Platform* (3rd ed.). Taibei: Qi feng zi xun gong si.

Shetye, S. (2019). Introduction to Java for Android Application Development. Retrieved from https://blog.udemy.com/java-for-android/

Maresco, J., & Maresco, J. (2019). 7 Essential Skills for Android Developers. Retrieved from https://generalassemb.ly/blog/7-essential-skills-you-need-to-be-an-android-developer/

Wieruch, R. (2019). *The road to learn React*.
Eckel, B., Wang, J., & Wang, S. (2008). *Thinking in Java*. Taibei Shi: Qi feng zi xun gu fen you xian gong si.

Bloch, J. (2018). *Effective Java*. Boston [etc.]: Addison-Wesley.

Ben, S. (2015). *Beginning JSON* (1st ed.). Apress

Martin, R. (2009). *Dai ma zheng jie zhi dao*. Beijing Shi: Ren min you dian chu ban she.

Harold, E. and Means, W. (2004). *XML in a nutshell*. Sebastopol, CA: O'Reilly.

Yahiaoui, H. (2017). *Firebase cookbook: Over 70 recipes to help you create real-time web and mobile applications with Firebase*. Birmingham, UK: Packt Publishing.

Smyth, N. (2017). *Android Studio 3.0 Development Essentials : Android 8 Edition*. 1st ed. CreateSpace Independent Publishing Platform.

Darwin, I. (2017). *Android cookbook: Problems and Solutions for Android Developers*. 2nd ed. O'Reilly Media.

Gookin, D. (2016). *Nexus tablets for dummies*. 4th ed. For Dummies.

Kleppmann, M. (2017). *Designing data-intensive applications*. 1st ed. O'Reilly Media.

DECLARATION
of the originality of the writing
(According to Code of Studies and Examinations of the University of Pécs, Annex nr. 14/1.*)*

I, the undersigned, YUAN CHENGE (ZJSZ2D), declare under penalty of perjury that every part of my writing, is the result of my own, autonomous work, I only used referred sources (special literature, tools, etc.) and I observed the pertaining rules of the University of Pécs while preparing my writing.

I am aware that the University of Pécs has the right to check the observation of copyright rules through a plagiarism tracing system.

Pécs, 29th of April, 2019

…………………………….
signature of the student