**基于安卓的食谱菜谱APP项目文档**

**itfxq-项目分享圈**

**version 1.0版本**

**目 录**

[一 项目概况 3](#_Toc86413058)

[1.1项目题目 3](#_Toc86413059)

[1.2.项目背景 3](#_Toc86413060)

[1.3.项目应用场景 3](#_Toc86413061)

[1.4.功能结构图 3](#_Toc86413062)

[1.6 项目特点 4](#_Toc86413063)

[1.5 功能流程图 4](#_Toc86413064)

[1.6 项目特点 5](#_Toc86413065)

[1.7 项目使用技术 5](#_Toc86413066)

[1.8小组成员及分工 6](#_Toc86413067)

[二 需求分析 6](#_Toc86413068)

[2.1 功能调研 6](#_Toc86413069)

[三项目设计 8](#_Toc86413070)

[3.1 数据库设计 8](#_Toc86413071)

[3.2 登录界面效果图 10](#_Toc86413072)

[3.3 登录实现 11](#_Toc86413073)

[3.4 注册界面效果图 13](#_Toc86413074)

[3.5 注册实现 14](#_Toc86413075)

[3.6 首页菜品推荐 15](#_Toc86413076)

[3.7 菜品列表实现 30](#_Toc86413077)

[3.8 菜品详情功能 43](#_Toc86413078)

[3.9 菜品收藏功能实现 56](#_Toc86413079)

[3.10 我的收藏 59](#_Toc86413080)

[3.12我的页面 63](#_Toc86413081)

[3.13 我基本信息 69](#_Toc86413082)

[3.14退出系统 73](#_Toc86413083)

[3.15管理员 系统登录 73](#_Toc86413084)

[3.16 个人信息 75](#_Toc86413085)

[3.17 系统管理 78](#_Toc86413086)

[3.18 后台发布菜品 79](#_Toc86413087)

[四 操作说明 88](#_Toc86413088)

[4.1 系统操作流程 88](#_Toc86413089)

[4.2 项目启动说明 88](#_Toc86413090)

[五 项目总结 89](#_Toc86413091)

[5.1 项目总结 89](#_Toc86413092)

# 一 项目概况

## 1.1项目题目

菜谱食谱app

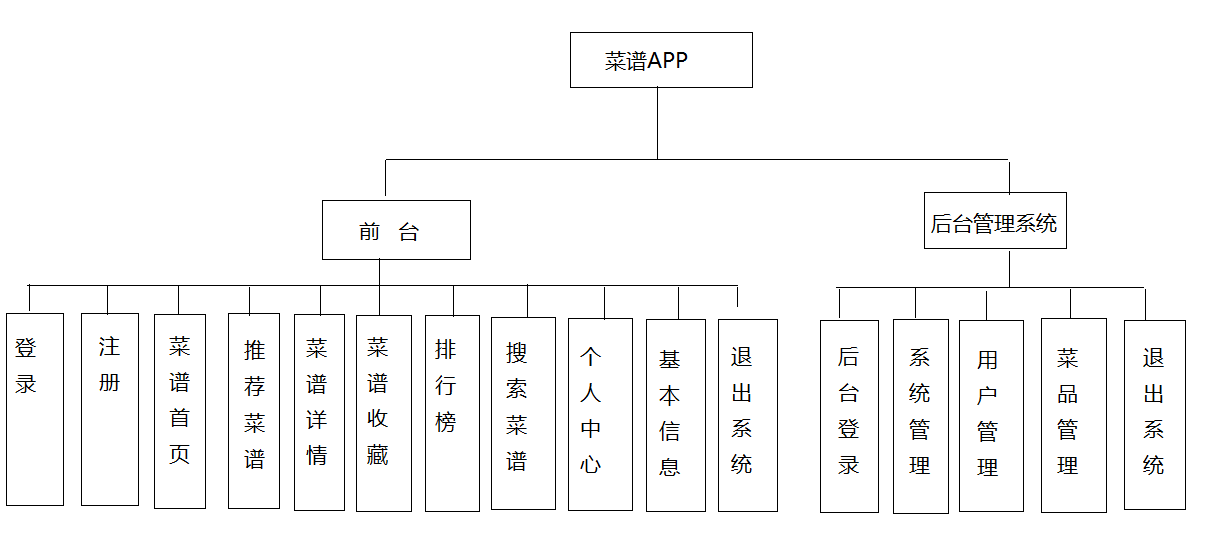
## 1.2.项目背景

唯有美食不可辜负。人们每天都要进行吃饭，吃饭是人们必须做的事情，吃饱了才有力气做事情，干活，而往往很多人在吃饭的适合，常常发愁，不知道吃什么，不知道怎么去做这套菜，而基于这样的需求，我们开发了这套食谱或者菜谱APP。解决了人们目前的事情。

## 1.3.项目应用场景

这个菜谱APP包含前台安卓端和后台管理系统，安卓端可以进行 菜品搜索，菜品推荐，收藏菜品，查看菜品排行，个人中心，注册，登录等，后台管理系统可以进行菜品的维护，用户的管理，系统管理等操作，东西非常多，如果您需要 做菜，你不知道怎么做的时候，就可以在该APP上面进行查阅 做菜的做法。非常的方便 快捷，解决人们的问题。

## 1.4.功能结构图



## 1.6 项目特点

(1) APP界面非常美观

(2) 功能比较实用

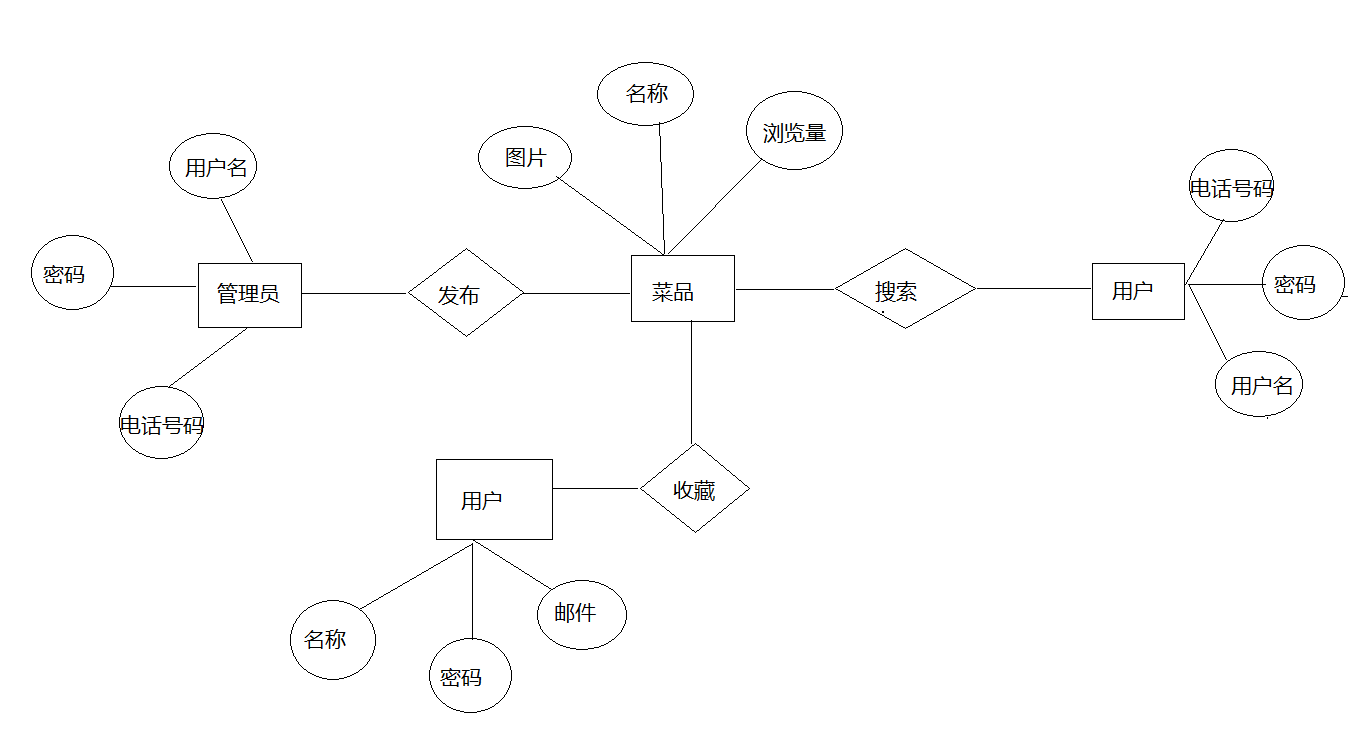
(3) 文档说明很丰富

## 1.5 功能流程图

App的菜谱APP或者食谱APP的流程图如下图：

主要有两种类型的用户，一种是管理员，一种是普通的用户，管理员可以进行发布菜品，维护菜品。普通用户可以进行在app端进行 浏览菜品 搜索菜品 查看菜品的做法,收藏菜品，菜品排行榜等功能

项目流程图:



## 1.6 项目特点

(1)界面好看

(2)技术点丰富

## 项目使用技术

该项目使用的开发工具：

android studio 或者 eclipse

数据库采用

Mysql

Jdk版本

jdk1.8以上

项目使用组件:

图片组件ImageView，

按钮组件Button，

文本框组件TextView，

编辑组件 EditText

列表组件ListView，

布局组件RelativeLayOut LinerLayOut,RecycleView,DrawLayout 等

后端使用技术：

SSM技术完成后台管理系统维护

## 1.8小组成员及分工

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 姓名 | 详细分工 | 备注（组长或组员） |
|  |  |  |  |

# 二 需求分析

## 2.1 功能调研

菜谱APP前台：(也就是安卓端 移动端)

1. 登录功能
2. 注册功能
3. 我的页面功能
4. 我的基本信息
5. 我的收藏
6. 退出功能
7. 首页菜谱推荐列表
8. 菜谱列表搜索
9. 菜谱详情功能
10. 菜谱排行榜功能

菜谱APP后台: (PC端)

1. 系统登录
2. 个人信息
3. 退出功能
4. 系统维护
5. 菜谱管理

2.2 功能说明

菜谱APP前台：(也就是安卓端 移动端)

(1)登录功能

主要完成系统的登录，未登录用户，无法使用菜谱APP

(2)注册功能

主要完成APP用户的注册功能，只有注册为该APP的用户，才能进行使用系统。在注册的时候，需要填写用户的基本信息，比如用户名称，密码，电话号码，性别，邮件等信息。

(3)我的页面功能

主要展示个人的基本信息，比如个人的姓名 邮件，电话号码等信息，退出的信息。

(4)我的个人信息

属于我的页面的里面功能，展示个人的信息。比如个人的姓名 邮件，电话号码等信息。

(5)我的收藏

展示的是我收藏的菜谱，菜谱可以取消收藏

(6)退出功能

点击退出功能。退出系统，返回登录页面，属于个人页面里面的功能。

(7)首页菜谱推荐列表

当登录进入系统之后，就可以查看到对应的系统推荐的菜谱

(8)菜谱列表

当点击首页里面的菜谱分类的时候，进可以进入该分类的菜谱列表，展示菜品的信息。同时也可以菜品列表进行 输入想要查看的菜品。

(9)菜品详情功能

点击菜单图片或者名称的时候，进入对应的菜品详情页面，同时系统会更新浏览量。

(10)收藏菜品功能

点击菜品详情的 ，收藏图标，进行菜品的收藏。会进入我的收藏列表中。多次点击只能收藏一次。

(11)菜品排行功能

展示菜品的排行信息。访问量高的 优先排到最前面。

菜谱APP后台: (PC端)

1. 系统登录

管理员可以登录到系统的后台，进行发布菜品的操作。

1. 个人信息

展示个人的信息，比如个人的姓名，联系方式，邮件，等信息。

1. 退出功能

退出后台系统

1. 系统维护

系统维护主要设计用户的维护，系统菜单的维护等功能

1. 菜品管理

移动端显示的菜品，都是经过后台管理员进行发布的菜品。同时菜品管理还可以进行修改和删除菜品的信息。并且菜品管理支出富文本编辑。

# 三项目设计

## 3.1 数据库设计

数据库库

数据库名称：itfxq\_menus\_app

数据库表

t\_user 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| username | varchar | 255 | 用户名称 |
| password | varchar | 255 | 密码 |
| email | varchar | 255 | 邮件 |
| tel | varchar | 11 | 电话号码 |
| sex | bit | 1 | 性别 |
| createTime | datetime | 0 | 创建时间 |

t\_role角色表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| name | varchar | 255 | 角色名称 |
| sn | varchar | 255 | 编号 |
| desc | varchar | 255 | 描述 |

t\_permission 权限表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键 |
| name | varchar | 255 | 权限名称 |
| title | varchar | 255 | 标题 |
| pid | bigint | 2 | 父id |
| menuid | bigint | 2 | 菜单id |

t\_role\_permission 角色权限表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| roleid | bigint | 2 | 角色id |
| permissionid | bigint | 2 | 权限id |

t\_user\_role 用户角色表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| userid | bigint | 2 | 用户id |
| roleid | bigint | 2 | 角色id |

t\_menu菜单表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| name | varchar | 255 | 菜单名称 |
| url | varchar | 255 | 菜单地址 |
| pid | bigint | 2 | 父id |
| icon | varchar | 255 | 图标 |

**t\_foods 菜品表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| foodName | varchar | 255 | 菜品名称 |
| taste | varchar | 255 | 口味 |
| price | decimal | 20 | 价格 |
| count | bigint | 2 | 数量 |
| foodPic | varchar | 255 | 菜品图片 |
| foodType | bigint | 255 | 菜品类型 |
| createTime | datetime | 0 | 创建时间 |
| ycl | varchar | 255 | 原材料 |
| zf | longtext | 0 | 做法 |
| viewnum | bigint | 2 | 浏览量 |

**t\_collection 菜品收藏表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 含义 |
| id | bigint | 2 | 主键id |
| username | varchar | 255 | 用户名 |
| foodid | bigint | 2 | 菜品id |
| createtime | datetime | 0 | 创建时间 |

## 3.2 登录界面效果图

登录效果图如下：



## 3.3 登录实现

登录功能实现：

首先设计登录界面,然后进行绑定点击事件，最后实现登录功能，登录成功之后直接进入系统，登录错误，给予用户提示。

核心代码:

|  |
| --- |
| public void loginSys(View view){  String username = loginNameEt.getText().toString().trim();  String pwd = loginPwdEt.getText().toString().trim();  ContentValues values = new ContentValues();  values.put("username",username);  values.put("password",pwd);   if(TextUtils.*isEmpty*(username) || TextUtils.*isEmpty*(pwd)){  Toast.*makeText*(this,"用户名和密码不能为空",Toast.*LENGTH\_SHORT*).show();  }else {  OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("username", username)  .add("password",pwd)  .build();   Request request = new Request.Builder()  .url(ItFxqConstants.*LOGIN\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);  *// 开启异步线程访问网络* call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res =response.body().string();Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;  msg.obj = res;  mLoginsHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {  }  });    }  } |
|  |

## 3.4 注册界面效果图



## 3.5 注册实现

注册功能的实现，首先填写用户的基本信息，输入完之后，点击注册，获取用户输入的信息，保存到数据库用户表中，最后提示保存成功的消息给用户。

核心代码：

|  |
| --- |
| public void saveUser(View view){  *//获取存入的内容-用户名* String username = usernameEt.getText().toString().trim();  *//获取存入的内容-密码* String pwd = pwdEt.getText().toString().trim();  *//获取存入的内容-电话* String tel = telEt.getText().toString().trim();  *//获取存入的内容-邮件* String email = emailEt.getText().toString().trim();  if(TextUtils.*isEmpty*(username) || TextUtils.*isEmpty*(pwd)){  Toast.*makeText*(this,"用户名和密码不能为空",Toast.*LENGTH\_SHORT*).show();  }else{  OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("username", username)  .add("password",pwd)  .add("email",email)  .add("tel",tel)  .add("sex",sex+"")  .build();   Request request = new Request.Builder()  .url(ItFxqConstants.*REG\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);  call.enqueue(new Callback() {  Message msg = new Message();  @Override  public void onResponse(Call call, Response response) throws IOException {  *//获取返回数据* String res = response.body().string();   msg.what = ItFxqConstants.*OK\_STATUS*;  msg.obj = res;  mRegHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {  msg.what = ItFxqConstants.*ERROR\_STATUS*;  mRegHandler.sendMessage(msg);  }  });  } } |

## 3.6 首页菜品推荐

首页菜品推荐效果图：



核心的代码

|  |
| --- |
| package cn.itfxq.foods.activity;  import androidx.appcompat.app.AppCompatActivity; import androidx.drawerlayout.widget.DrawerLayout;  import android.app.ActionBar; import android.content.Intent; import android.os.Bundle; import android.os.Handler; import android.os.Message; import android.view.MotionEvent; import android.view.View; import android.widget.Button; import android.widget.ImageView; import android.widget.ListView; import android.widget.PopupWindow; import android.widget.TextView; import android.widget.Toast;  import com.google.gson.Gson; import com.google.gson.reflect.TypeToken;  import java.io.IOException; import java.lang.reflect.Type; import java.util.List; import java.util.Map;  import cn.itfxq.foods.R; import cn.itfxq.foods.adapter.TjCpAdapter; import cn.itfxq.foods.entity.FoodEntity; import cn.itfxq.foods.utils.CommonUtils; import cn.itfxq.foods.utils.ItFxqConstants; import cn.itfxq.foods.utils.RemoteDatas; import okhttp3.Call; import okhttp3.Callback; import okhttp3.FormBody; import okhttp3.OkHttpClient; import okhttp3.Request; import okhttp3.RequestBody; import okhttp3.Response;  public class MainActivity extends AppCompatActivity implements View.OnClickListener {   PopupWindow popupWindow;*//定义popupWindow* Button myMenuBtn;*//我的菜单* ImageView leftMenuIv;  TextView main\_mycenter,main\_rank,main\_mysc,f1Tv,f2Tv,f3Tv,f4Tv,f5Tv;  ListView tjListView;  TjCpAdapter mTjCpAdapter;  List<FoodEntity> mtjFoodEntityList;  MainHandler mMainHandler;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  initView();  initDatas();  initEvent();  }   private void initDatas() {  OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .build();   Request request = new Request.Builder()  .url(ItFxqConstants.*TJCP\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);  *// 开启异步线程访问网络* call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;  msg.obj = res;  mMainHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });  }   public void initView() {   leftMenuIv = findViewById(R.id.*leftMenuIv*);  main\_mycenter = findViewById(R.id.*main\_mycenter*);  main\_rank = findViewById(R.id.*main\_rank*);  main\_mysc = findViewById(R.id.*main\_mysc*);  f1Tv=findViewById(R.id.*f1tv*);  f2Tv=findViewById(R.id.*f2tv*);  f3Tv=findViewById(R.id.*f3tv*);  f4Tv=findViewById(R.id.*f4tv*);  f5Tv=findViewById(R.id.*f5tv*);  tjListView = findViewById(R.id.*tjListViewId*);  mTjCpAdapter = new TjCpAdapter(this);  mMainHandler = new MainHandler();  }   public void initEvent() {   leftMenuIv.setOnClickListener(this);  main\_mycenter.setOnClickListener(this);  main\_rank.setOnClickListener(this);  main\_mysc.setOnClickListener(this);  f1Tv.setOnClickListener(this);  f2Tv.setOnClickListener(this);  f3Tv.setOnClickListener(this);  f4Tv.setOnClickListener(this);  f5Tv.setOnClickListener(this);    }   *//菜单按钮的单机事件* public void OnMenu(View v){  *//获取自定义菜单的布局文件* View popupWindow\_view=getLayoutInflater().inflate(R.layout.*menu*,null,false);  *//创建popupWindow实例，设置菜单的宽度和高度* popupWindow=new PopupWindow(popupWindow\_view, ActionBar.LayoutParams.*WRAP\_CONTENT*,ActionBar.LayoutParams.*WRAP\_CONTENT*,true);  *//设置菜单显示在按钮的下面* popupWindow.showAsDropDown(findViewById(R.id.*btn\_menu*),0,0);  myMenuBtn= popupWindow\_view.findViewById(R.id.*myMenuBtn*);   *//单机其它位置隐藏菜单* popupWindow\_view.setOnTouchListener(new View.OnTouchListener(){  @Override  public boolean onTouch(View v, MotionEvent event){  *//如果菜单存在并且处于显示状态* if (popupWindow!=null&&popupWindow.isShowing()){  popupWindow.dismiss();*//关闭菜单* popupWindow=null;  }  return false;  }  });  myMenuBtn.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {   CommonUtils.*navigateTo*(MainActivity.this,MyActivity.class);  }  });  }   @Override  public void onClick(View v) {  Intent intent=new Intent();  intent.setClass(MainActivity.this,FoodsActivity.class);  DrawerLayout drawerLayout = findViewById(R.id.*DrawerLayout*);  View leftlayout = findViewById(R.id.*left\_layout*);  drawerLayout.setScrimColor(0x00ffffff);  Bundle bundle = new Bundle();    switch (v.getId()) {   case R.id.*leftMenuIv*:  drawerLayout.openDrawer(leftlayout);  break;  case R.id.*main\_mycenter*:  *//个人中心* drawerLayout.closeDrawer(leftlayout);  CommonUtils.*navigateTo*(MainActivity.this,MyActivity.class);  break;  case R.id.*main\_mysc*:  *//我的收藏* drawerLayout.closeDrawer(leftlayout);  CommonUtils.*navigateTo*(MainActivity.this,MyScFoodActivity.class);  break;  case R.id.*main\_rank*:  *//我的排行榜* drawerLayout.closeDrawer(leftlayout);  CommonUtils.*navigateTo*(MainActivity.this,RankActivity.class);  break;  case R.id.*f1tv*:   bundle.putString("foodType", "1");  *//早餐* CommonUtils.*navigateTo*(MainActivity.this,FoodsActivity.class,bundle);   break;  case R.id.*f2tv*:  bundle.putString("foodType", "2");  *//午餐* CommonUtils.*navigateTo*(MainActivity.this,FoodsActivity.class,bundle);  break;  case R.id.*f3tv*:  bundle.putString("foodType", "3");  *//晚餐* CommonUtils.*navigateTo*(MainActivity.this,FoodsActivity.class,bundle);  break;  case R.id.*f4tv*:  bundle.putString("foodType", "4");  *//水果和甜点* CommonUtils.*navigateTo*(MainActivity.this,FoodsActivity.class,bundle);  break;  case R.id.*f5tv*:  bundle.putString("foodType", "5");  *//全部* CommonUtils.*navigateTo*(MainActivity.this,FoodsActivity.class,bundle);  break;   }  }   class MainHandler extends Handler {  @Override  public void dispatchMessage(Message msg) {  super.dispatchMessage(msg);  switch (msg.what) {  case ItFxqConstants.*OK\_STATUS*:  *//解析获取的JSON数据* Gson gson = new Gson();  *//通过反射得到type对象* Type listType = new TypeToken<Map>() {  }.getType();   String resultJson = (String) msg.obj;  *//解析获取的JSON数据* mtjFoodEntityList = RemoteDatas.*INSTANCE*.getFoodsList(resultJson);  *//创建适配器对象* mTjCpAdapter = new TjCpAdapter(MainActivity.this);  mTjCpAdapter.setData(mtjFoodEntityList);  tjListView.setAdapter(mTjCpAdapter);  CommonUtils.*setListViewHeight*(tjListView);  *//数据源变化，提示适配器更新* mTjCpAdapter.notifyDataSetChanged();    break;  case ItFxqConstants.*ERROR\_STATUS*:  Toast.*makeText*(MainActivity.this,"操作失败",Toast.*LENGTH\_SHORT*).show();  break;  }  }  } } |

## 3.7 菜品列表实现

点击首页的菜品分类的时候，就会进入菜品列表界面。从数据库查询出来，展示到列表中。

效果图:



核心代码:

|  |
| --- |
| package cn.itfxq.foods.activity;  import androidx.appcompat.app.AppCompatActivity;   import android.os.Bundle; import android.os.Handler; import android.os.Message; import android.view.View; import android.widget.ImageView; import android.widget.ListView; import android.widget.SearchView; import android.widget.TextView;  import com.bumptech.glide.Glide;  import java.io.IOException; import java.util.ArrayList; import java.util.List; import java.util.stream.Collectors;  import cn.itfxq.foods.R; import cn.itfxq.foods.adapter.FoodAdapter; import cn.itfxq.foods.entity.FoodEntity; import cn.itfxq.foods.utils.ItFxqConstants; import cn.itfxq.foods.utils.RemoteDatas; import okhttp3.Call; import okhttp3.Callback; import okhttp3.OkHttpClient; import okhttp3.Request; import okhttp3.Response;  public class FoodsActivity extends AppCompatActivity implements View.  OnClickListener {   private FoodEntity mFoodEntity;  private TextView foodNameTv;  private ImageView foodPicIv;  private FoodAdapter mFoodAdapter;  public static final int *OKSTATUS* = 1;  private FoodsHandler mFoodsHandler;   private ListView foodsListView;  *//总的食物* private List<FoodEntity> foodList = new ArrayList<>();   private ListView listLv;  SearchView menuSV;   private String foodType;      @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_foods*);   mFoodsHandler = new FoodsHandler();  mFoodAdapter=new FoodAdapter(this);  foodsListView = findViewById(R.id.*listLv*);  menuSV = findViewById(R.id.*menuSV*);  menuSV.setSubmitButtonEnabled(true);  foodType = getIntent().getStringExtra("foodType");  init();   }   private void init(){  *//初始化adapter* initAdapter();  *//初始化数据* initData();  *//初始化视图* initView();  *//设置数据* setData();  initEvent();  }   private void initEvent() {  menuSV.setOnQueryTextListener(new SearchView.OnQueryTextListener() {  *//点击搜索按钮时激发该方法* @Override  public boolean onQueryTextSubmit(String keyword) {  List tempList = new ArrayList();  *//实际应用中应该在该方法内执行实际查询，此处仅使用Toast显示用户输入的查询内容* tempList = foodList.stream().filter(foodEntity -> foodEntity.getFoodName()  .contains(keyword))  .collect(Collectors.*toList*());  mFoodAdapter.setData(tempList);  mFoodAdapter.notifyDataSetChanged();  return false;  }   *//用户输入字符时激发该方法* @Override  public boolean onQueryTextChange(String newText) {   return true;  }  });  }   private void initAdapter(){    mFoodAdapter = new FoodAdapter(this);  foodsListView.setAdapter(mFoodAdapter);   }     */\*\*  \* 初始化界面控件  \*/* private void initView() {   foodNameTv = findViewById(R.id.*foodNameTv*);  foodPicIv = findViewById(R.id.*foodPicIv*);    listLv = (ListView) findViewById(R.id.*listLv*);     }   private void initData() {  OkHttpClient okHttpClient = new OkHttpClient();  Request request = new Request.Builder().url(ItFxqConstants.*FOOD\_URL*).build();  Call call = okHttpClient.newCall(request);   call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = *OKSTATUS*;  msg.obj = res;  mFoodsHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });  }   */\*\*  \* 设置界面数据  \*/* private void setData() {  if (mFoodEntity == null) return;  foodNameTv.setText(mFoodEntity.getFoodName());  Glide.*with*(this)  .load(mFoodEntity.getFoodPic())  .error(R.mipmap.*ic\_launcher*)  .into(foodPicIv);  }   @Override  public void onClick(View v) {  switch (v.getId()){    }  }        class FoodsHandler extends Handler {  @Override  public void dispatchMessage(Message msg) {  super.dispatchMessage(msg);  switch (msg.what) {  case *OKSTATUS*:  if (msg.obj != null) {  String resultJson = (String) msg.obj;  *//解析获取的JSON数据* foodList = RemoteDatas.*INSTANCE*.getFoodsList(resultJson);  if("1".equals(foodType)){  foodList = foodList.stream().filter(foodEntity ->  foodEntity.getFoodType().equals("1")  ).collect(Collectors.*toList*());  }else if("2".equals(foodType)){  foodList = foodList.stream().filter(foodEntity ->  foodEntity.getFoodType().equals("2")  ).collect(Collectors.*toList*());  }else if("3".equals(foodType)){  foodList = foodList.stream().filter(foodEntity ->  foodEntity.getFoodType().equals("3")  ).collect(Collectors.*toList*());  }else if("4".equals(foodType)){  foodList = foodList.stream().filter(foodEntity ->  foodEntity.getFoodType().equals("5") || foodEntity.getFoodType().equals("6")  ).collect(Collectors.*toList*());  }  mFoodAdapter.setData(foodList);  mFoodAdapter.notifyDataSetChanged();   }  break;   }  }  }     } |

## 3.8 菜品详情功能

点击菜品的名称，或者菜品图片的时候，进入菜品的详情页面

效果图:



核心实现代码：

|  |
| --- |
| package cn.itfxq.foods.activity;  import androidx.appcompat.app.AppCompatActivity;  import android.content.ContentValues; import android.content.Intent; import android.os.Bundle; import android.os.Handler; import android.os.Message; import android.view.View; import android.webkit.WebBackForwardList; import android.webkit.WebView; import android.webkit.WebViewClient; import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast;  import com.bumptech.glide.Glide;  import java.io.IOException; import java.util.stream.Collectors;  import cn.itfxq.foods.R; import cn.itfxq.foods.entity.FoodEntity; import cn.itfxq.foods.utils.CommonUtils; import cn.itfxq.foods.utils.ItFxqConstants; import cn.itfxq.foods.utils.RemoteDatas; import okhttp3.Call; import okhttp3.Callback; import okhttp3.FormBody; import okhttp3.OkHttpClient; import okhttp3.Request; import okhttp3.RequestBody; import okhttp3.Response;  public class FoodDetailActivity extends AppCompatActivity {   FoodEntity mFoodEntity;  ImageView fooddetail\_foodPicIv,fooddetail\_scIv;  TextView fooddetail\_nameTv,fooddetail\_yclTv,fooddetail\_viewnumTv;  WebView fooddetail\_zfWv;  FoodsDetailHandler mFoodsDetailHandler;  FoodsDetailHandler1 mFoodsDetailHandler1;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_food\_detail*);  initView();  initData();  }   private void initData() {  mFoodEntity = (FoodEntity) getIntent().getExtras().get("foodEntity");   *//根据id去查询菜品 并且更新菜品浏览量* OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("id", mFoodEntity.getId()+"")  .build();  Request request = new Request.Builder()  .url(ItFxqConstants.*GETFOODBYID\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);   call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;  msg.obj = res;  mFoodsDetailHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });      }   class FoodsDetailHandler extends Handler {  @Override  public void dispatchMessage(Message msg) {  super.dispatchMessage(msg);  switch (msg.what) {  case ItFxqConstants.*OK\_STATUS*:  if (msg.obj != null) {  String resultJson = (String) msg.obj;  *//解析获取的JSON数据* mFoodEntity = RemoteDatas.*INSTANCE*.getFoodEntity(resultJson);  Glide.*with*(getApplicationContext())  .load(mFoodEntity.getFoodPic())  .into(fooddetail\_foodPicIv);   fooddetail\_nameTv.setText("菜品名称:"+mFoodEntity.getFoodName());  fooddetail\_yclTv.setText("材料:"+mFoodEntity.getYcl());  fooddetail\_viewnumTv.setText(mFoodEntity.getViewnum()+"");   String htmlPart1 = "<!DOCTYPE HTML html>\n" +  "<head><meta charset=\"utf-8\"/>\n" +  "<meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, minimum-scale=1.0, user-scalable=no\"/>\n" +  "</head>\n" +  "<body>\n" +  "<style> \n" +  "img{width:100%!important;height:auto!important}\n" +  " </style>";  String htmlPart2 = "</body></html>";   fooddetail\_zfWv.getSettings().setJavaScriptEnabled(true);*//设置JS可用* String html = htmlPart1 + mFoodEntity.getZf() + htmlPart2;   fooddetail\_zfWv.loadDataWithBaseURL(null, html, "text/html", "UTF-8", null);  fooddetail\_zfWv.setWebViewClient(new WebViewClient() {  @Override  public void onPageFinished(WebView view, String url) {  }  });   }  break;   }  }  }  class FoodsDetailHandler1 extends Handler {  @Override  public void dispatchMessage(Message msg) {  super.dispatchMessage(msg);  switch (msg.what) {  case ItFxqConstants.*OK\_STATUS*:  Toast.*makeText*(FoodDetailActivity.this,"收藏成功",Toast.*LENGTH\_SHORT*).show();  break;  }  }  }   private void initView() {  fooddetail\_foodPicIv = findViewById(R.id.*fooddetail\_foodPicIv*);  fooddetail\_nameTv = findViewById(R.id.*fooddetail\_nameTv*);  fooddetail\_yclTv = findViewById(R.id.*fooddetail\_yclTv*);  fooddetail\_zfWv = findViewById(R.id.*fooddetail\_zfWv*);  fooddetail\_viewnumTv = findViewById(R.id.*fooddetail\_viewnumTv*);  fooddetail\_scIv = findViewById(R.id.*fooddetail\_scIv*);  mFoodsDetailHandler = new FoodsDetailHandler();  mFoodsDetailHandler1 = new FoodsDetailHandler1();  fooddetail\_scIv.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  *//收藏 foodid username  //根据id去查询菜品 并且更新菜品浏览量* OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("foodid", mFoodEntity.getId()+"")  .add("username", CommonUtils.*getLoginUser*(getApplicationContext()).getUsername())  .build();  Request request = new Request.Builder()  .url(ItFxqConstants.*SCFOOD\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);   call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;   mFoodsDetailHandler1.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });    }  });  }   public void backPage(View view){  Intent intent=new Intent();  intent.setClass(this, MainActivity.class);  *//启动* startActivity(intent);  } } |

## 3.9 菜品收藏功能实现

在详情页面的时候，点击收藏功能，然后会进行收藏菜品，多次收藏，只能一次成功。

如下效果图:



核心代码实现:

|  |
| --- |
| fooddetail\_scIv.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  *//收藏 foodid username  //根据id去查询菜品 并且更新菜品浏览量* OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("foodid", mFoodEntity.getId()+"")  .add("username", CommonUtils.*getLoginUser*(getApplicationContext()).getUsername())  .build();  Request request = new Request.Builder()  .url(ItFxqConstants.*SCFOOD\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);   call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;   mFoodsDetailHandler1.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });  } }); |

## 3.10 我的收藏

在首页左侧菜单，或者个人中心，点击我的收藏，进入我的收藏列表



核心代码如下：

|  |
| --- |
| package cn.itfxq.foods.activity;  import androidx.appcompat.app.AppCompatActivity;  import android.os.Bundle; import android.os.Handler; import android.os.Message; import android.widget.ListView;  import java.io.IOException; import java.util.List; import java.util.stream.Collectors;  import cn.itfxq.foods.R; import cn.itfxq.foods.adapter.ScCpAdapter; import cn.itfxq.foods.entity.FoodEntity; import cn.itfxq.foods.utils.CommonUtils; import cn.itfxq.foods.utils.ItFxqConstants; import cn.itfxq.foods.utils.RemoteDatas; import okhttp3.Call; import okhttp3.Callback; import okhttp3.FormBody; import okhttp3.OkHttpClient; import okhttp3.Request; import okhttp3.RequestBody; import okhttp3.Response;  public class MyScFoodActivity extends AppCompatActivity {   ListView scLv;  ScCpAdapter mScCpAdapter;  ScFoodsHandler mScFoodsHandler;  List<FoodEntity> foodList;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_my\_sc\_food*);  initView();  initData();  }   private void initView() {  scLv = findViewById(R.id.*sclistLv*);  mScCpAdapter = new ScCpAdapter(this);  scLv.setAdapter(mScCpAdapter);  mScFoodsHandler = new ScFoodsHandler();  }    private void initData() {   OkHttpClient okHttpClient = new OkHttpClient();  RequestBody requestBody = new FormBody.Builder()  .add("username", CommonUtils.*getLoginUser*(this).getUsername())  .build();  Request request = new Request.Builder()  .url(ItFxqConstants.*MYSCFOOD\_URL*)  .post(requestBody).build();  Call call = okHttpClient.newCall(request);    call.enqueue(new Callback() {  @Override  public void onResponse(Call call, Response response) throws IOException {  String res = response.body().string();  Message msg = new Message();  msg.what = ItFxqConstants.*OK\_STATUS*;  msg.obj = res;  mScFoodsHandler.sendMessage(msg);  }  @Override  public void onFailure(Call call, IOException e) {   }  });   }   class ScFoodsHandler extends Handler {  @Override  public void dispatchMessage(Message msg) {  super.dispatchMessage(msg);  switch (msg.what) {  case ItFxqConstants.*OK\_STATUS*:  if (msg.obj != null) {  String resultJson = (String) msg.obj;  *//解析获取的JSON数据* foodList = RemoteDatas.*INSTANCE*.getFoodsList(resultJson);  mScCpAdapter.setData(foodList);  mScCpAdapter.notifyDataSetChanged();  CommonUtils.*setListViewHeight*(scLv);  }  break;   }  }  } } |

## 3.12我的页面

我的页面 展示我的个人信息，效果图如下:



核心代码如下:

|  |
| --- |
| package cn.itfxq.foods.activity;  import android.content.Intent; import android.os.Bundle; import android.view.View; import android.widget.TextView;  import androidx.appcompat.app.AppCompatActivity;  import java.util.Map;  import cn.itfxq.foods.R; import cn.itfxq.foods.entity.UserEntity; import cn.itfxq.foods.utils.CommonUtils;   public class MyActivity extends AppCompatActivity implements View.OnClickListener {   *//退出文本组件* TextView logout\_tv;  TextView my\_name\_tv,my\_scTv,my\_rankTv,my\_baseinfo,my\_exit;    @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_my*);     initView();  initData();  initEvent();  }   private void initView(){  logout\_tv = findViewById(R.id.*logout*);  my\_name\_tv = findViewById(R.id.*my\_name*);  my\_scTv = findViewById(R.id.*my\_scTv*);  my\_rankTv = findViewById(R.id.*my\_rankTv*);  my\_baseinfo = findViewById(R.id.*my\_baseinfo*);  my\_exit = findViewById(R.id.*my\_exit*);   my\_scTv.setOnClickListener(this);  my\_rankTv.setOnClickListener(this);  my\_baseinfo.setOnClickListener(this);  my\_exit.setOnClickListener(this);  logout\_tv.setOnClickListener(this);  }   private void initData(){  UserEntity loginUser = CommonUtils.*getLoginUser*(MyActivity.this);  my\_name\_tv.setText(loginUser.getUsername());   }   private void initEvent(){   }   @Override  public void onClick(View v) {  switch (v.getId()) {   case R.id.*my\_scTv*:  Intent intent=new Intent();  intent.setClass(MyActivity.this,MyScFoodActivity.class);  startActivity(intent);  break;  case R.id.*my\_rankTv*:  Intent intent1=new Intent();  intent1.setClass(MyActivity.this,RankActivity.class);  startActivity(intent1);  break;  case R.id.*my\_baseinfo*:  *//基本信息* Intent intent2=new Intent();  intent2.setClass(MyActivity.this,MyBaseInfoActivity.class);  startActivity(intent2);  break;  case R.id.*my\_exit*:  Intent intent3=new Intent();  intent3.setClass(MyActivity.this,LoginActivity.class);  startActivity(intent3);  break;  case R.id.*logout*:  Intent intent4=new Intent();  intent4.setClass(MyActivity.this,LoginActivity.class);  startActivity(intent4);  break;   }  } } |

## 3.13 我基本信息

点击我的基本信息，进入基本信息页面。



核心代码:

|  |
| --- |
| package cn.itfxq.foods.activity;  import androidx.appcompat.app.AppCompatActivity;  import android.os.Bundle; import android.widget.TextView;  import cn.itfxq.foods.R; import cn.itfxq.foods.utils.CommonUtils;  */\*\*  \* 我的基本信息  \*/* public class MyBaseInfoActivity extends AppCompatActivity {   TextView mybaseinfo\_name,mybaseinfo\_email,mybaseinfo\_tel;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_my\_base\_info*);  initView();  initData();  }   private void initView() {  mybaseinfo\_name = findViewById(R.id.*mybaseinfo\_name*);  mybaseinfo\_email = findViewById(R.id.*mybaseinfo\_email*);  mybaseinfo\_tel = findViewById(R.id.*mybaseinfo\_tel*);  }  private void initData() {  mybaseinfo\_name.setText(CommonUtils.*getLoginUser*(this).getUsername());  mybaseinfo\_email.setText(CommonUtils.*getLoginUser*(this).getEmail());  mybaseinfo\_tel.setText(CommonUtils.*getLoginUser*(this).getTel());  } } |

## 3.14退出系统

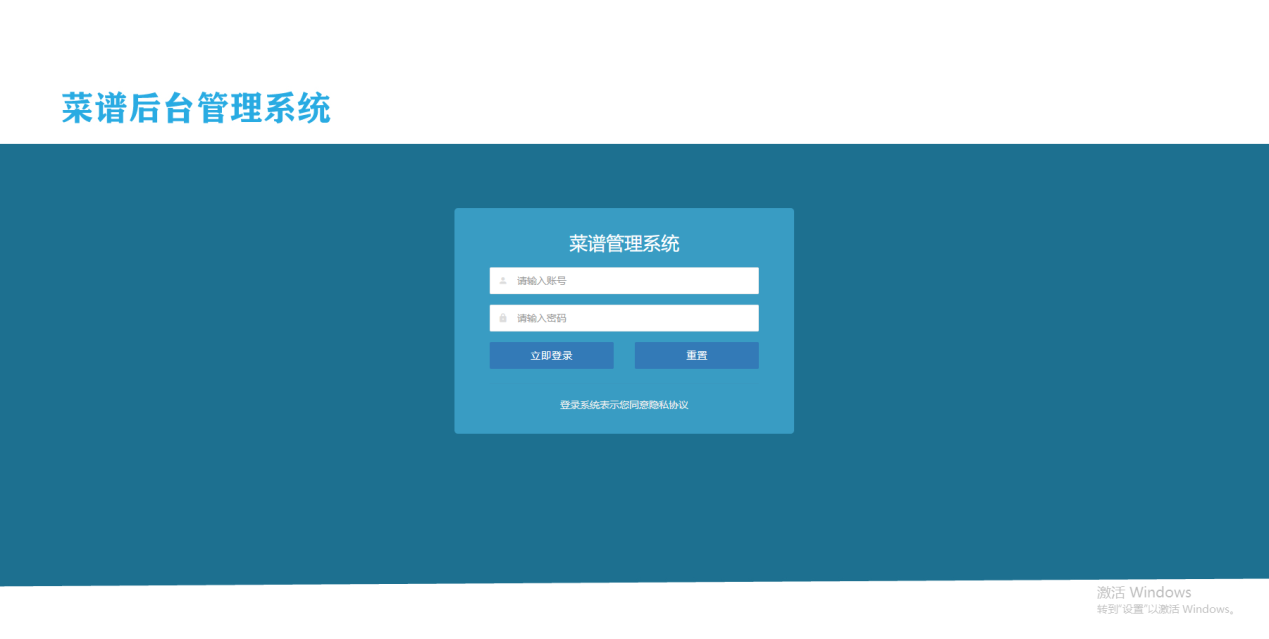
退出系统，是在我的页面中，点击退出的时候，退出APP，返回登录界面

核心代码

|  |
| --- |
| Intent intent=new Intent(); intent.setClass(MyActivity.this,LoginActivity.class); startActivity(intent); |

## 3.15管理员 系统登录

效果图：



核心代码实现:

|  |
| --- |
| <body> <div class="row coder-wrapper">  <div class="coder-login">  <div class="login-center">  <div class="login-header text-center">  <div style="color:white;font-size:26px;"> 菜谱管理系统 </div>  </div>  <form th:action="@{/login}" method="post">  <div class="form-group has-feedback feedback-left">  <input style="color:black" type="text" placeholder="请输入账号" class="form-control" name="username" id="username" />  <span class="mdi mdi-account form-control-feedback" aria-hidden="true"></span>  </div>  <div class="form-group has-feedback feedback-left">  <input style="color:black" type="password" placeholder="请输入密码" class="form-control" id="password" name="password" />  <span class="mdi mdi-lock form-control-feedback" aria-hidden="true"></span>  </div>    <div class="form-group row">  <div class="col-xs-6">  <button class="btn btn-block btn-primary" type="button" th:onclick="goLogin()">立即登录</button>  </div>  <div class="col-xs-6">  <button class="btn btn-block btn-primary" type="button" th:onclick="reset()">重置</button>  </div>  </div>  <div id="errorInfo" style="display: none"></div>  </form>  <hr>  <footer class="col-sm-12 text-center">  <p class="m-b-0"><a>登录系统表示您同意隐私协议</a></p>  </footer>  </div>  </div> </div> <script type="text/javascript" th:src="@{/static/js/jquery.min.js}"></script> <script type="text/javascript" th:src="@{/static/js/bootstrap.min.js}"></script> <!--对话框--> <script type="text/javascript" th:src="@{/static/js/jconfirm/jquery-confirm.min.js}"></script> <script type="text/javascript" th:inline="javascript">    function reset(){  $("#username").val('');  $("#password").val('');  }    function goLogin(){  //注册  var username = $("#username").val();  var password = $("#password").val();  //验证通过  $.ajax({  url: "/login",  type: "POST",  data: {"username":username,"password":password},  success: function (data) {  ***console***.log(data);  if (data.isSuccess) {  ***window***.location.href="/index"  } else{  $("#errorInfo").css('display',"block");  $("#errorInfo").html("<span style='color:red'>"+data.message+"</span>");  }  }  });  }  </script> </body> </html> |

## 3.16 个人信息

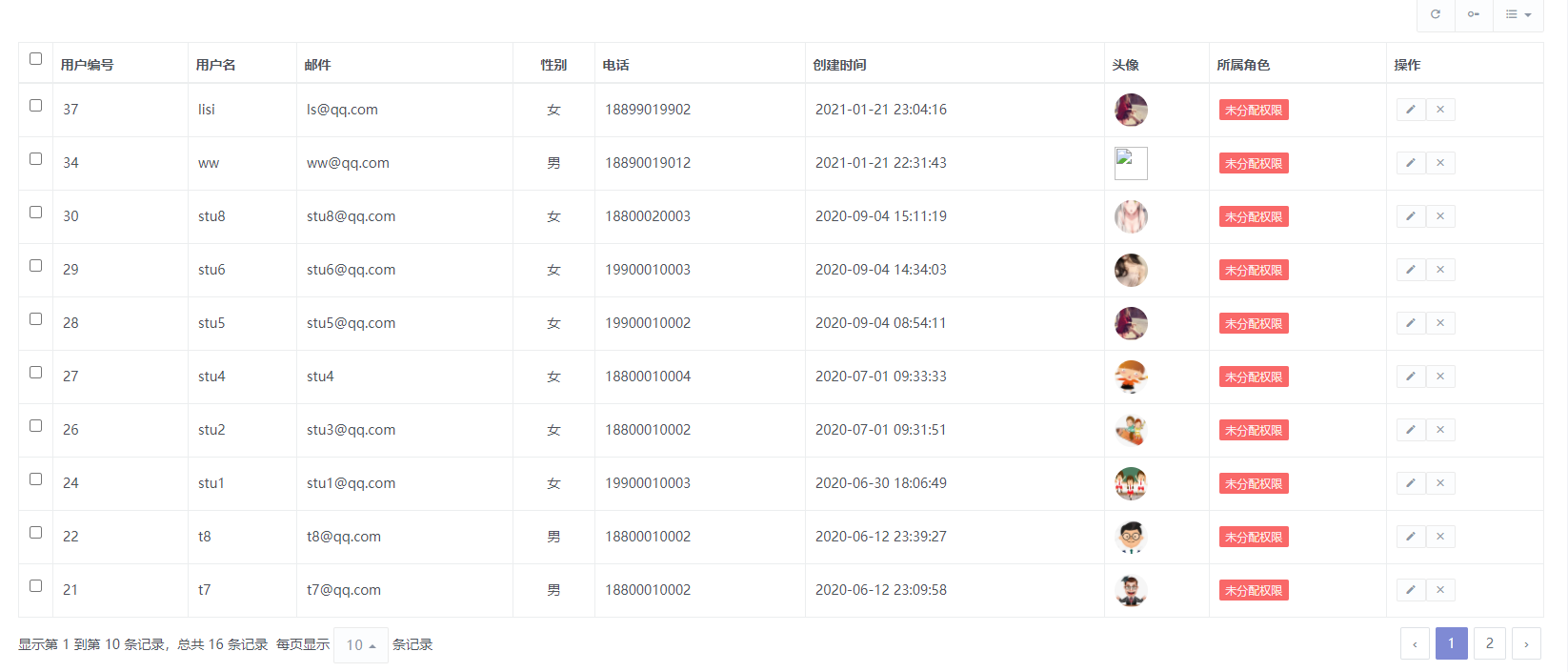
展示个人的基本信息。

核心代码实现:

|  |
| --- |
| <!--页面主要内容-->  <main class="coder-layout-content" >   <div class="container-fluid"  style="background-color:transparent;  padding-top: 0px;  padding-bottom: 0px;">  <div class="row">  <div class="col-lg-12">  <div class="card">  <!-- 新增表单start-->  <div class="modal-dialog"  style="float:left;  margin-top:0px;  width: 100%;  background-color: black;  color:#ff8f2b;" role="document">  <div class="modal-content">  <div class="modal-header">  <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>  <h4 class="modal-title" >  <span style="font-size: 18px;color:#ff8f2b;font-weight: bold">您的位置: 个人信息</span>  </h4>  </div>  <form id="stuEditForm" method="post" enctype="multipart/form-data">  <div class="modal-body">  <input type="hidden" id="e\_id" class="form-control" name="id">  <div class="form-group">  <label class="control-label">用户名：</label>  <input type="text" readonly th:value="${loginUser.username}" class="form-control" name="username">  </div>   <div class="form-group" style="width: 30%">  <label class="control-label">头像:</label>  <input id="file-pic" name="file" type="file" data-show-remove="false" multiple />  <p class="help-block">支持jpg、jpeg、png、gif,txt,"docx","zip","xlsx"格式，大小没限制</p>  <input name="fileUpdate" type="button" th:onclick="javascript:updateHeadImg()" value="更新头像" />  </div>  <div class="form-group">  <label class="control-label">邮箱：</label>  <input type="text" readonly th:value="${loginUser.email}" class="form-control" name="email" >  </div>  <div class="form-group">  <label class="control-label">电话号码：</label>  <input type="text" readonly th:value="${loginUser.tel}" class="form-control" name="tel" >  </div>  <div class="form-group" th:if="${loginUser.type==3}">  <label class="control-label" >学号：</label>  <input type="text" readonly th:value="${loginUser.stunum}" class="form-control" name="stunum" >  </div>   <div class="form-group">  <label class="control-label">性别: <span th:text="${loginUser.sex?'男':'女'}"></span> </label>   </div>  <div class="form-group">  <label class="control-label">用户类型: <span th:text="${loginUser.type }"></span> </label>   </div>   </div>   </form>  </div>  </div>  <!-- 新增表单结束-->  </div>  </div>  <div class="col-lg-12">  </div>  </div>   </div>   </div> |

## 3.17 系统管理

主要管理系统的用户，菜单的内容，效果图：

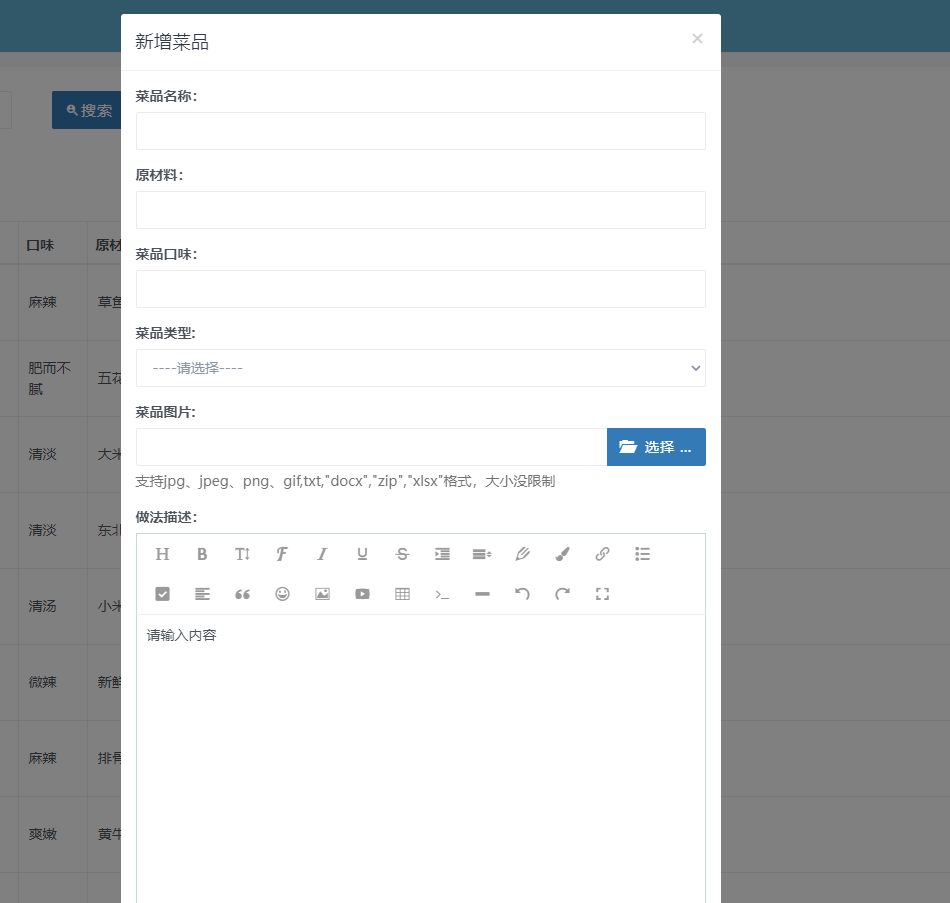


核心代码：

|  |
| --- |
| @RequestMapping("/list") @ResponseBody @PreAuthorize("hasRole('管理员')") public List<User> queryAll(){  return userService.queryAll(); }   @GetMapping("/index") public String index(String menuid, Model model){  model.addAttribute("menuid",menuid);  List<Role> roles = roleService.queryAll();  model.addAttribute("roles",roles);  //用户首页  return "views/user/user\_list"; }  @GetMapping("/listpage") @ResponseBody public PageList listpage(UserQuery userQuery){  return userService.listpage(userQuery); }  @Override public PageList listpage(BaseQuery baseQuery) {  PageList pageList = new PageList();  //查询总的条数  Long total = baseMapper.queryTotal(baseQuery);  List<T> users = baseMapper.queryData(baseQuery);  pageList.setTotal(total);  pageList.setRows(users);  //分页查询的数据  return pageList; } |

## 3.18 后台发布菜品

效果图：



|  |
| --- |
|  |

核心代码：

|  |
| --- |
| <!DOCTYPE html> <html lang="zh" xmlns:th="http://www.thymeleaf.org/thymeleaf-extras-springsecurity4"  xmlns:sec="http://www.thymeleaf.org/extras/spring-security"> <head>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, food-scalable=no" />  <title>后台管理系统模板</title>  <link rel="icon" href="favicon.ico" type="image/ico">  <meta name="keywords" content="模板">  <meta name="description" content="后台管理系统">  <meta name="author" content="SoulCoder">  <div th:include="~{views/common/include :: cssStyle }"></div> </head> <body> <div class="coder-layout-web">  <div class="coder-layout-container">  <!--左侧导航-->  <div th:include="~{views/common/include :: #asideStyle }"></div>  <!--End 左侧导航-->  <!--头部信息-->  <div th:include="~{views/common/include :: #headerStyle }"></div>  <!--End 头部信息-->  <!--页面主要内容-->  <main class="coder-layout-content">   <div class="container-fluid">  <div class="row">  <div class="col-lg-12">  <div class="card">  <div class="card-toolbar clearfix">  <!-- 查询搜索框start-->  <div id="toolbar">  <div class="my-container">  <label class="myLabel-content">名称：</label>  <div class="myText-content">  <input id="q\_foodName" name="foodName" type="text" class="form-control" placeholder="输入名称">  </div>  </div>  <div class="myBtn-content">  <button id="search" type="button" class="btn btn-primary" data-toggle="modal" data-method="search">  <i class="mdi mdi-account-search"></i>搜索</button>  </div>  </div>  <!-- 查询搜索框end-->  <!-- 按钮区start-->  <div class="toolbar-btn-action">  <button type="button" class="btn btn-primary" data-toggle="modal" data-method="add" >  <i class="mdi mdi-plus"></i>新增  </button>   </div>  <!-- 按钮区end-->  </div>  <div class="card-body">  <div class="table-responsive">  <table id="foodTable" ></table>  </div>  </div>   <!-- 新增表单start-->  <div class="modal fade" id="foodAddModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel">  <div class="modal-dialog" role="document">  <div class="modal-content">  <div class="modal-header">  <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>  <h4 class="modal-title" id="exampleModalLabel">新增菜品</h4>  </div>  <form id="foodAddForm" method="post" enctype="multipart/form-data">  <div class="modal-body">   <div class="form-group">  <label for="recipient-name" class="control-label">菜品名称：</label>  <input type="text" class="form-control" name="foodName" id="recipient-name">  </div>  <div class="form-group">  <label for="recipient-ycl" class="control-label">原材料：</label>  <input type="text" class="form-control" name="ycl" id="recipient-ycl">  </div>  <div class="form-group">  <label for="recipient-taste" class="control-label">菜品口味：</label>  <input type="text" class="form-control" name="taste" id="recipient-taste">  </div> <!-- <div class="form-group">--> <!-- <label for="recipient-price" class="control-label">价格：</label>--> <!-- <input type="text" class="form-control" name="price" id="recipient-price">--> <!-- </div>-->   <div class="form-group">  <label for="recipient-foodType" class="control-label">菜品类型:</label>  <select class="form-control" name="foodType" id="recipient-foodType">  <option value="-1">----请选择----</option>  <option value="1">早餐</option>  <option value="2">午餐</option>  <option value="3">晚餐</option>  <option value="4">小吃</option>  <option value="5">水果</option>  <option value="6">甜点</option>  <option value="7">午餐晚餐都可</option>  </select>  </div>   <!-- 头像上传-->  <div class="form-group">  <label for="recipient-foodType" class="control-label">菜品图片:</label>   <input id="file-pic" name="file" type="file" value="上传" multiple/>  <p class="help-block">支持jpg、jpeg、png、gif,txt,"docx","zip","xlsx"格式，大小没限制</p>   </div>  <div class="form-group">  <label class="control-label">做法描述：</label>  <div id="a\_zf">  <p>请输入内容</p>  </div>  <input id="zf\_Id" type="hidden"/>  </div>   </div>  <div class="modal-footer">  <button type="button" class="btn btn-default" data-dismiss="modal">关闭</button>  <button type="button" data-method='save' class="btn btn-primary">保存</button>  </div>  </form>  </div>  </div>  </div>  <!-- 新增表单结束-->    <!-- 修改表单start-->  <div class="modal fade" id="foodEditModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel">  <div class="modal-dialog" role="document">  <div class="modal-content">  <div class="modal-header">  <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>  <h4 class="modal-title" >修改菜品</h4>  </div>  <form id="foodEditForm" method="post" enctype="multipart/form-data">  <div class="modal-body">  <input type="hidden" id="e\_id" class="form-control" name="id">  <div class="form-group">  <label for="recipient-name" class="control-label">菜品名称：</label>  <input type="text" id="e\_foodName" class="form-control" name="foodName">  </div>  <div class="form-group">  <label for="recipient-taste" class="control-label">菜品口味：</label>  <input type="text" id="e\_taste" class="form-control" name="taste">  </div>  <div class="form-group">  <label for="e\_ycl" class="control-label">原材料：</label>  <input type="text" class="form-control" name="ycl" id="e\_ycl">  </div> <!-- <div class="form-group">--> <!-- <label for="recipient-price" class="control-label">价格：</label>--> <!-- <input type="text" id="e\_price" class="form-control" name="price">--> <!-- </div>-->   <div class="form-group">  <label for="recipient-foodType" class="control-label">菜品类型:</label>  <select id="e\_foodType" class="form-control" name="foodType" >  <option value="-1">----请选择----</option>  <option value="1">早餐</option>  <option value="2">午餐</option>  <option value="3">晚餐</option>  <option value="4">小吃</option>  <option value="5">水果</option>  <option value="6">甜点</option>  <option value="7">午餐晚餐都可</option>  </select>  </div>   <!-- 菜品图片-->  <div class="form-group">  <label for="recipient-foodType" class="control-label">菜品图片:</label>   <input id="e-file-pic" name="file" type="file" value="上传" multiple/>  <p class="help-block">支持jpg、jpeg、png、gif,txt,"docx","zip","xlsx"格式，大小没限制</p>   </div>   <div class="form-group">  <label class="control-label">做法描述：</label>  <div id="e\_zf">  <p>请输入内容</p>  </div>  <input id="e\_zf\_Id" type="hidden"/>  </div>   </div>  <div class="modal-footer">  <button type="button" class="btn btn-default" data-dismiss="modal">关闭</button>  <button type="button" data-method='editSave' class="btn btn-primary">保存</button>  </div>  </form>  </div>  </div>  </div>  <!-- 修改表单end-->    </div>  </div>   </div>   </div>   </main>  <!--End 页面主要内容-->  </div> </div>  <div th:include="~{views/common/include :: #jsStyle}"></div> <script type="text/javascript" th:src="@{/static/js/jquery.serialize-object.min.js}"></script> <script type="text/javascript" th:src="@{/static/plugin/wangEditor.min.js}"></script>  <script type="text/javascript" th:src="@{/static/api/food.js}"></script>    <style>  .my-container {  float: left;  display: inline-block;  margin-right:30px;  }   .myLabel-content ,.myText-content,.myBtn-content{  float: left;  display: inline-block;  line-height: 40px;  margin-left: 10px;  }  .myLabel-content,.myText-content input[type='text'],.myBtn-content .btn {  height: 38px;  font-size: 15px;  }  .myBtn-content .btn {  margin-bottom: 10px;  margin-right: 20px;  } </style>    </body> </html> |

# 四 操作说明

## 4.1 系统操作流程

首先管理员 登录PC后台管理系统，进行菜品的发布操作，菜品的维护操作

普通用户登录APP系统，进行浏览菜品，然后进行查看菜品的做法，然后可以收藏菜品，以及查看菜品的排行榜。

## 4.2 项目启动说明

(1) 启动后台管理系统：找到对应的FoodsApp.java，运行main方法 启动后台管理系统

@SpringBootApplication  
@MapperScan("cn.itfxq")  
public class FoodsApp {  
 public static void main(String[] args) {  
 SpringApplication springApplication = new SpringApplication(FoodsApp.class);  
  
 springApplication.*run*(FoodsApp.class,args);  
 }  
}

访问地址：

<http://localhost>

(2) 启动android studio 的app程序，访问前台服务

# 五 项目总结

## 5.1 项目总结

这个项目，首先让我学到很多相关android方法的知识。让我的编程能力得到了提升，同时由于时间的关系，如果项目想要更加强大，还需要添加更多的功能。感谢各位老师的辛苦指导，才能顺利的完整该项目的设计。