华南理工大学软件学院

《移动平台开发技术》课程报告（论文）

零食推荐交流 APP—食刻

年级专业：**2017** 软件工程

班级： **2017** 级 **3** 班

任课教师：程兴国

组长： 林俊涛 学号：201730682461

组员 1： 林添 学号：201730682508

组员 2： 张力丹 学号：201730685035

组员 3： 张家铭 学号：201730685011

2019—2020 学年第一学期

教师考核评分：

姓名	班级	学号	评分
林俊涛	17 级 3 班	201730682461	
林添	17 级 3 班	201730682508	
张力丹	17 级 3 班	201730685035	
张家铭	17 级 3 班	201730685011	
团队总评： 指导老师：　　日期：			

零 食 推 荐 交 流 APP— 食 刻

摘要：现在的互联网越来越发达，全球化对人们的生活的影响也越来越广泛。互联网的普及也带给了人们越来越多的方便之处。大家更愿意也更倾向于网上购物，而更轻松地买到自己满意的东西更是成了大家最高的追求。我们制作的 APP——《食刻》为所有人提供一个零食交流平台，以供大家分享自己对某个食物的想法以及它的口味。本篇课程报告，我们将从 APP 的各个方面，包括但不限于需求、功能、架构等方面去详细阐述我们的 APP。

关键词：互联网　全球化　平台　智能推荐

目录

1. 需求分析

1.1 任务概述

1.1.1 项目背景

现在的互联网越来越发达，全球化对人们的生活的影响也越来越广泛。互联网的普及也带给了人们越来越多的方便之处。大家更愿意也更倾向于网上购物，而更轻松地买到自己满意的东西更是成了大家最高的追求。我们的 APP 为所有人提供一个零食交流平台，以供大家分享自己对某个食物的想法以及它的口味。食物属于哪种口味，适合哪种人群，价格是否适中，都可以为想要购物的年轻人提供借鉴，甚至是照片、气味，都可以进行描述。能让人们更简单、轻松地了解某种食物，根据自己的需要进行选择购买。

1.1.2 实现目标

为年轻人提供一个零食推荐交流平台，主要功能包括：

1、 登陆注册自己的账号

2、 修改自己的个人信息，包括用户名、头像等

3、 发布自己对于某种零食的帖子

4、 点击帖子查看帖子的详细信息

5、 输入关键词搜索有关的帖子

6、 收藏自己感兴趣的帖子

7、 在帖子下面进行评论回复

8、 根据帖子的热度更新排行榜

1.2 功能需求

1.2.1 功能描述

系统主要有以下几个模块：登录注册，发布帖子，评论帖子，搜索帖子和收藏帖子

功能名称	功能标识	功能详细描述
注册	注册按钮	用户可以通过提供用户名和密码进行注册
登录	登录按钮	用户可以通过提供用户名和密码进行登录
修改个人信息	个人信息更改栏	在我的界面点击各个信息进行相应的更改，包括用户名、头像等
发布帖子	发布按钮	点击首页悬浮的发布按钮即可配图并编辑文字来发布帖子进行分享
评论帖子	评论按钮	在浏览帖子时可以点击帖子下方的评论按钮对该帖子发表自己的看法
搜索帖子	搜索栏	在搜索栏输入相应的关键词搜索相关的帖子
收藏帖子	收藏按钮	可以对自己感兴趣的帖子点击收藏，每被收藏一次，帖子的热度就加一
更新排行榜	排行榜界面栏	根据帖子的实时热度进行排序，并在排行榜展示

表 1

1.3 性能需求

1、时间要求：登录、注册、发布帖子、搜索帖子的延迟不可超过一秒。

2、适应性：可运行在 Android5 以上系统的任何智能手机系统环境下。

3、操作要求：操作简单，易上手，符合年轻人使用习惯

1.4 运行环境描述

1、硬件：Android 平台的移动设备

2、软件：

开发环境：Android SDK2.2

开发平台：AndroidStudio

开发语言：Java

数据库：LitePal

测试平台：Samsung，genymotion 模拟器

网络环境：局域网

1.5 市场现状

目前市场上有许多可以分享看法的平台，例如微博、小红书，但是每个 APP 内容都过杂，除分享外还有很多其他功能，使用不够简单，功能不够单纯，不能很好的找到相应的内容，而我们的 APP 是为“吃货们”提供一个交流单纯，操作简单的平台。

2．功能介绍

2.1 总用例图

如图所示：

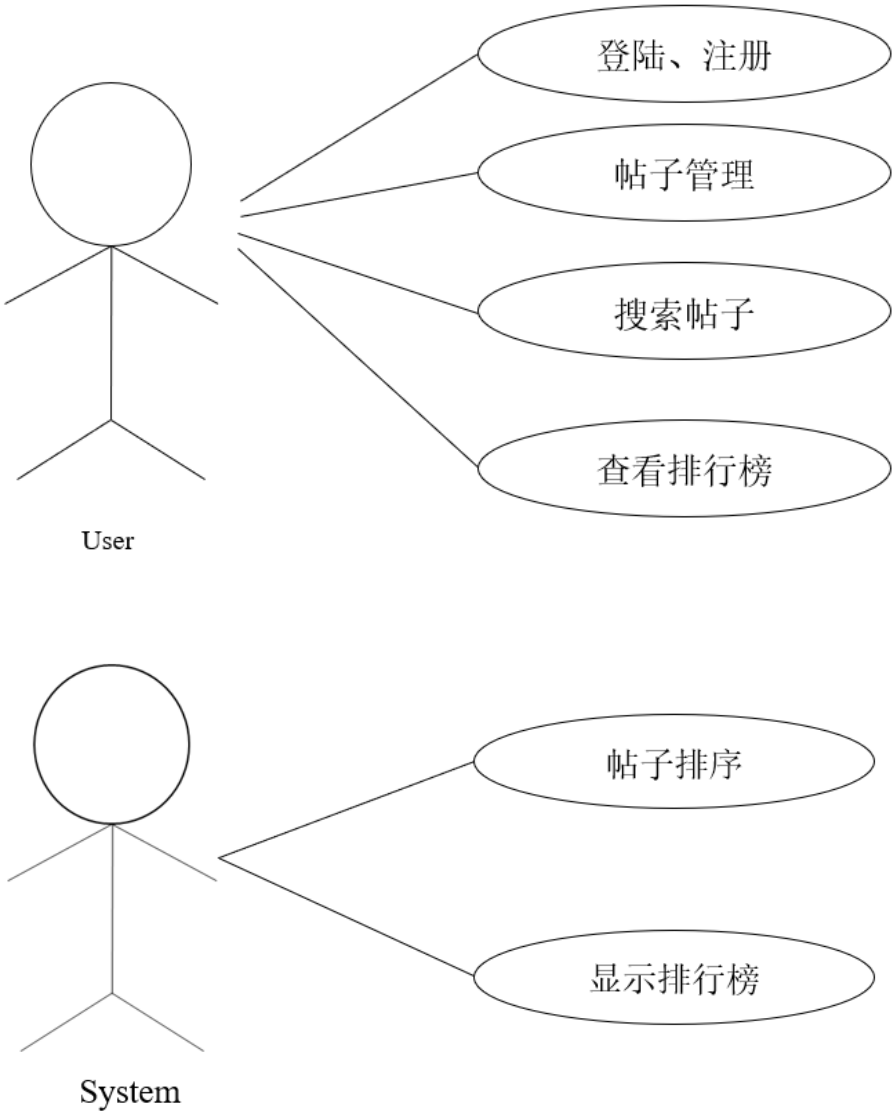

图 1

2.2 登录、注册

2.2.1 用例图

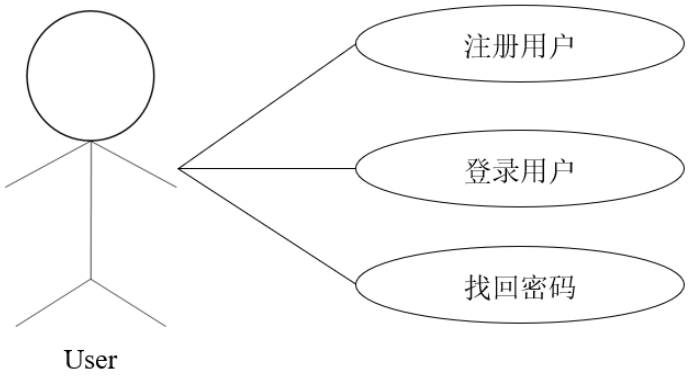

图 2

2.2.2 用例描述

1、用例简要描述:该用例主要描述用户使用软件的时候要进行的登录或注册的行为

2、用例角色:用户

3、用例事件流

基本事件流

(1) 登录已有账户；

a) 用户输入用户名与密码

b) 手机端获取用户输入内容并处理

c) 后台数据库验证用户密码是否匹配

d) 匹配成功则进入首页，失败则返回提示信息

(2) 注册新账户；

a) 用户输入注册信息

b) 手机端获取输入内容并处理

c) 后台创建新用户并更新数据库

备选事件流

(1) 填写信息未完全，弹出提醒框；

(2) 登录、注册成功，弹出提醒框并界面跳转；

(3) 登录失败，输入账号或密码错误，弹出提醒框

(4) 注册失败，输入账号已存在，弹出提醒框

(5) 忘记密码，显示初始账号及密码，弹出提醒框

2.3 帖子管理

2.3.1 用例图

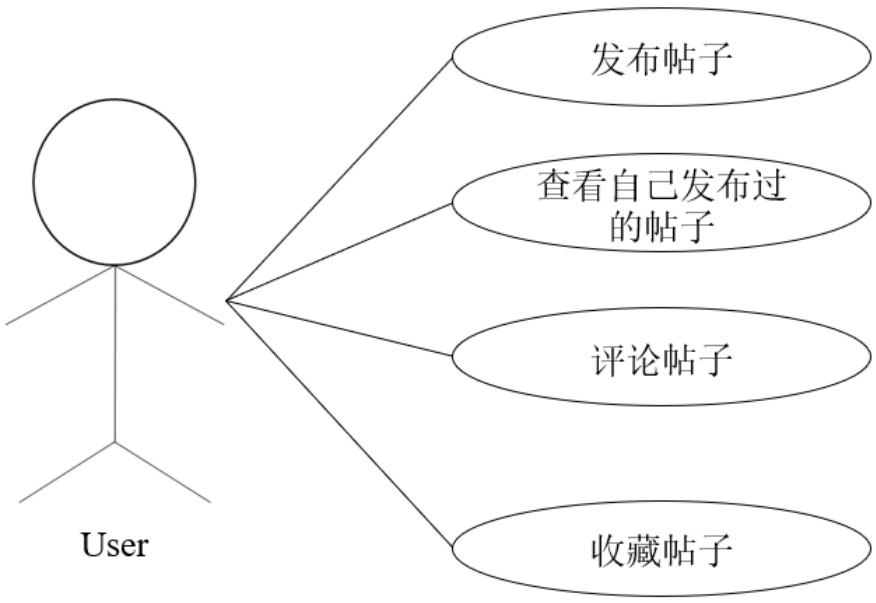

图 3

2.3.2 用例描述

1、用例简要描述:该用例描述用户在使用软件的时候发布帖子发表自己的看法，在帖子内评论自己的或是别人的帖子，在回复页面回复他人的评论，以及收藏感兴趣的帖子

2、用例角色:用户

3、用例事件流

基本事件流

(1) 用户 1 发布帖子，后台处理后将帖子更新发布在首页列表

(2) 用户 1 可以在“我的帖子”页面查看自己曾经发布的帖子，以列表方式进行展示

(3) 用户 2 可以查看用户 1 发布的帖子

(4) 用户 2 可以对用户 1 的帖子进行评论

(5) 用户 2 可以收藏用户 1 发布的帖子

(6) 用户 1 可以回复用户 2 对帖子的评论

备选事件流

(1) 用户点击心形按钮进行帖子收藏：

a) 心形按钮样式改变

b) 系统会记录下用户曾经收藏过的帖子，并在进入帖子页面时判断是否改变心形按钮的样式

c) 用户点击心形按钮收藏后，帖子外面的热点数加一

d) 帖子根据用户收藏数改变在排行榜的上下位置

(2) 用户对帖子进行评论：

a) 帖子显示该帖子被评论的数量

b) 帖子评论内容不能为空，会弹出窗口提示

c) 帖子下方动态增加用户评论

d) 用户的评论有查看回复的按钮，点击进入回复页面

(3) 用户回复评论

e) 回复页面顶部显示楼主及其评论内容

f) 回复页面记录并显示该条评论被回复的数量

g) 回复内容不能为空，会弹出窗口提示

2.4 搜索帖子

2.4.1 用例图

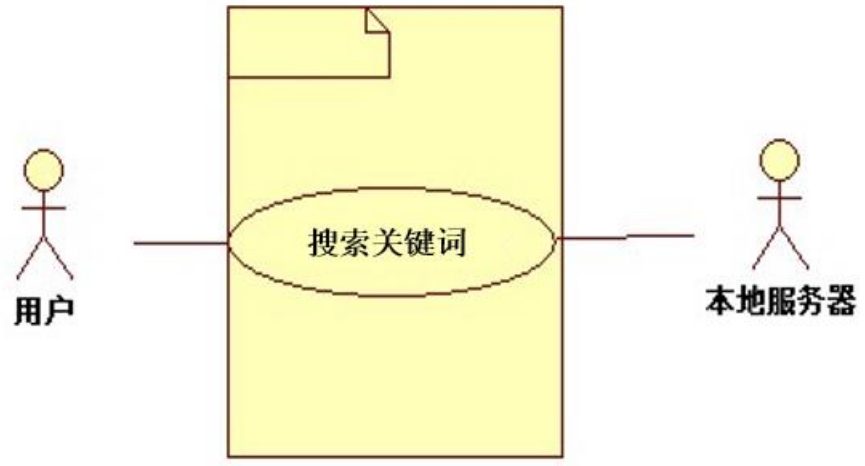

图 4

2.4.2 用例描述

1、用例简要描述:该用例主要描述该应用在搜索帖子这部分的功能，在搜索栏输入内容即可查找符合搜索条件的帖子，能简洁快速地搜索到想要的内容

2、用例角色:用户

3、用例事件流

基本事件流

(1) 在搜索栏输入关键词进行搜索；

(2) 系统返回并显示满足条件的帖子；

备选事件流

(1) 没有满足搜索条件的帖子，返回提示无相关内容。

(2) 搜索栏内容为空时，页面内容变回搜索前的首页内容

2.5 查看排行榜

2.5.1 用例图

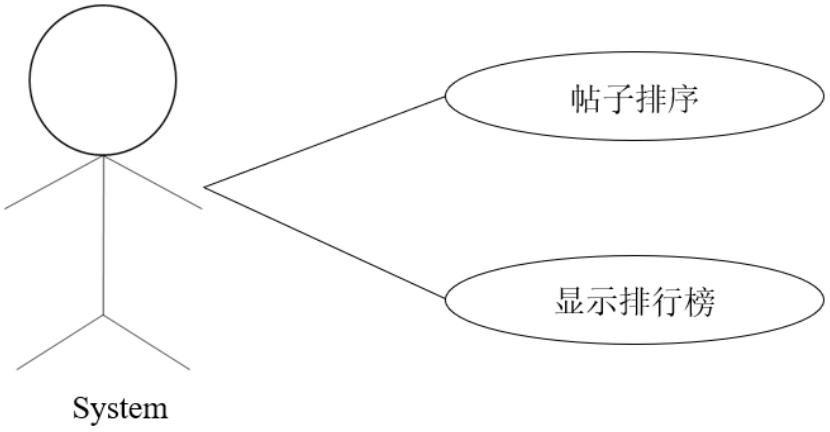

图 5

2.5.2 用例描述

1、用例简要描述:用户可以在校友圈功能中发表自己的心情感想经历等等文字信息给别的用户看到，大家共同分享。

2、用例角色:用户

3、用例事件流

基本事件流

(1) 查看实时排行榜，排行榜的帖子根据收藏数降序排列

备选事件流

(1) 排列的帖子和首页的功能一样，可点击进入查看帖子内容、评论、收藏、回复等

3．项目计划与进度

3.1 项目计划

项目计划表(部分工作并行处理)

任务名称	工期	开始时间	完成时间
《食刻》项目计划	**74 个工作日**	**2019 年 10 月 10 日**	**2019 年 12 月 24 日**
1、启动	**4 个工作日**	**2019 年 10 月 10 日**	**2019 年 10 月 14 日**
1.1 识别项目干系人	2 个工作日	2019 年 10 月 10 日	2019 年 10 月 12 日
1.2 制作项目章程	1 个工作日	2019 年 10 月 12 日	2019 年 10 月 13 日
1.3 小组召开启动会议	1 个工作日	2019 年 10 月 11 日	2019 年 10 月 12 日
1.4 项目范围规划结束	1 个工作日	2019 年 10 月 13 日	2019 年 10 月 14 日
2、计划	**5 个工作日**	**2019 年 10 月 14 日**	**2019 年 10 月 19 日**
2.1 制定项目开题报告与计划	2 个工作日	2019 年 10 月 14 日	2019 年 10 月 16 日
2.2 项目范围说明书	3 个工作日	2019 年 10 月 16 日	2019 年 10 月 19 日
2.3 项目成本估计	1 个工作日	2019 年 10 月 15 日	2019 年 10 月 16 日
2.2 项目成本计划	1 个工作日	2019 年 10 月 17 日	2019 年 10 月 18 日
2.5 编写 WBS 字典	2 个工作日	2019 年 10 月 17 日	2019 年 10 月 19 日
3、执行	**65 个工作日**	**2019 年 10 月 19 日**	**2019 年 12 月 24 日**
3.1 需求分析	**5 个工作日**	**2019 年 10 月 19 日**	**2019 年 10 月 25 日**
3.1.1 功能结构图	1 个工作日	2019 年 10 月 19 日	2019 年 10 月 20 日
3.1.2 设计用例	3 个工作日	2019 年 10 月 20 日	2019 年 10 月 23 日
3.1.3 非功能需求	1 个工作日	2019 年 10 月 23 日	2019 年 10 月 24 日
3.1.4 原型设计	4 个工作日	2019 年 10 月 25 日	2019 年 10 月 29 日
登录界面、注册界面、找回密码界面	1 个工作日	2019 年 10 月 25 日	2019 年 10 月 26 日
首页界面、排行榜界面、个人信息界面	1 个工作日	2019 年 10 月 26 日	2019 年 10 月 27 日
编辑帖子界面、帖子详情界面	1 个工作日	2019 年 10 月 27 日	2019 年 10 月 28 日
APP 图标	1 个工作日	2019 年 10 月 28 日	2019 年 10 月 29 日

3.1.5 编写需求分析文档	4 个工作日	2019 年 10 月 29 日	2019 年 11 月 1 日
编写系统功能需求	1 个工作日	2019 年 10 月 29 日	2019 年 10 月 30 日
编写系统功能详细描述	2 个工作日	2019 年 10 月 29 日	2019 年 10 月 31 日
编写系统非功能需求	1 个工作日	2019 年 10 月 31 日	2019 年 11 月 1 日
编写系统接口	1 个工作日	2019 年 10 月 31 日	2019 年 11 月 1 日
3.1.6 需求分析评审	1 个工作日	2019 年 11 月 1 日	2019 年 11 月 2 日
3.2 概要设计	**6 个工作日**	**2019 年 11 月 2 日**	**2019 年 11 月 8 日**
3.2.1 系统 E-R 图设计	1 个工作日	2019 年 11 月 2 日	2019 年 11 月 3 日
3.2.2 系统逻辑架构设计	2 个工作日	2019 年 11 月 3 日	2019 年 11 月 5 日
3.2.3 系统物理架构设计	1 个工作日	2019 年 11 月 4 日	2019 年 11 月 5 日
3.2.4 系统接口设计	1 个工作日	2019 年 11 月 5 日	2019 年 11 月 6 日
3.2.6 整合系统数据模型文档	1 个工作日	2019 年 11 月 5 日	2019 年 11 月 6 日
3.2.6 编写概要设计文档	1 个工作日	2019 年 11 月 6 日	2019 年 11 月 7 日
3.2.7 概要设计评审	0.5 个工作日	2019 年 11 月 7 日	2019 年 11 月 7 日
3.3 详细设计	**16 个工作日**	**2019 年 11 月 8 日**	**2019 年 11 月 21 日**
3.3.1 系统简述	1 个工作日	2019 年 11 月 8 日	2019 年 11 月 9 日
3.3.2 术语表	1 个工作日	2019 年 11 月 9 日	2019 年 11 月 10 日
3.3.3 设计概述	**11 个工作日**	**2019 年 11 月 10 日**	**2019 年 11 月 21 日**
3.3.3.1 系统的复用计划	1 个工作日	2019 年 11 月 10 日	2019 年 11 月 11 日
3.3.3.2 系统的接口设计	1 个工作日	2019 年 11 月 10 日	2019 年 11 月 11 日
3.3.3.3 对象模型设计（类图设计）	10 个工作日	2019 年 11 月 11 日	2019 年 11 月 21 日
客户端	4 个工作日	2019 年 11 月 11 日	2019 年 11 月 15 日
登录、注册模块	1 个工作日	2019 年 11 月 11 日	2019 年 11 月 12 日
帖子管理模块	2 个工作日	2019 年 11 月 11 日	2019 年 11 月 13 日
搜索帖子模块	1 个工作日	2019 年 11 月 12 日	2019 年 11 月 13 日
排行榜模块	2 个工作日	2019 年 11 月 13 日	2019 年 11 月 15 日
类描述	**2 个工作日**	**2019 年 11 月 19 日**	**2019 年 11 月 21 日**

3.3.3.4 系统用例实现详细设计	**4 个工作日**	**2019 年 11 月 21 日**	**2019 年 11 月 25 日**
登录	1 个工作日	2019 年 11 月 21 日	2019 年 11 月 22 日
注册	1 个工作日	2019 年 11 月 21 日	2019 年 11 月 22 日
帖子管理	2 个工作日	2019 年 11 月 22 日	2019 年 11 月 24 日
搜索帖子	1 个工作日	2019 年 11 月 23 日	2019 年 11 月 24 日
排行榜	1 个工作日	2019 年 11 月 24 日	2019 年 11 月 25 日
3.3.4 编制详细设计文档	**2 个工作日**	**2019 年 11 月 26 日**	**2019 年 11 月 28 日**
文档整合	1 个工作日	2019 年 11 月 26 日	2019 年 11 月 27 日
格式调整	1 个工作日	2019 年 11 月 26 日	2019 年 11 月 27 日
审核内容	1 个工作日	2019 年 11 月 27 日	2019 年 11 月 28 日
3.3.5 详细设计评审	1 个工作日	2019 年 11 月 27 日	2019 年 11 月 28 日
3.4 编码阶段	**20 个工作日**	**2019 年 11 月 28 日**	**2019 年 12 月 17 日**
客户端	**15 个工作日**	**2019 年 11 月 28 日**	**2019 年 12 月 7 日**
登录	**12 个工作日**	**2019 年 11 月 28 日**	**2019 年 12 月 10 日**
编写布局文件	1 个工作日	2019 年 11 月 28 日	2019 年 11 月 29 日
在 Activity 中注册布局组件	4 个工作日	2019 年 11 月 28 日	2019 年 12 月 1 日
绑定点击监听器	10 个工作日	2019 年 11 月 30 日	2019 年 12 月 10 日
取出、传递参数	8 个工作日	2019 年 12 月 1 日	2019 年 12 月 9 日
注册	**6 个工作日**	**2019 年 11 月 29 日**	**2019 年 12 月 5 日**
编写布局文件	1 个工作日	2019 年 11 月 29 日	2019 年 11 月 30 日
在 Activity 中注册布局组件	4 个工作日	2019 年 11 月 30 日	2019 年 12 月 4 日
绑定点击监听器	6 个工作日	2019 年 11 月 30 日	2019 年 12 月 5 日
取出、传递参数（消息类）	5 个工作日	2019 年 12 月 1 日	2019 年 12 月 5 日
帖子管理	**10 个工作日**	**2019 年 12 月 5 日**	**2019 年 12 月 15 日**
搜索帖子	**7 个工作日**	**2019 年 12 月 5 日**	**2019 年 12 月 12 日**
排行榜	**5 个工作日**	**2019 年 12 月 10 日**	**2019 年 12 月 15 日**
数据库	**2 个工作日**	**2019 年 12 月 11 日**	**2019 年 12 月 13 日**

user 表设计	2 个工作日	2019 年 12 月 11 日	2019 年 10 月 13 日
后台实现	**4 个工作日**	**2019 年 12 月 13 日**	**2019 年 12 月 17 日**
数据库访问接口	3 个工作日	2019 年 12 月 13 日	2019 年 12 月 16 日
3.5 测试	**7 个工作日**	**2019 年 12 月 17 日**	**2019 年 12 月 24 日**
客户端	**2 个工作日**	**2019 年 12 月 17 日**	**2019 年 12 月 19 日**
登录、注册模块	1 个工作日	2019 年 12 月 17 日	2019 年 12 月 18 日
帖子管理模块	1 个工作日	2019 年 12 月 17 日	2019 年 12 月 18 日
排行榜模块	1 个工作日	2019 年 12 月 18 日	2019 年 12 月 19 日
个人信息模块	1 个工作日	2019 年 12 月 18 日	2019 年 12 月 19 日
易用性测试	**2 个工作日**	**2019 年 12 月 20 日**	**2019 年 12 月 22 日**
按键测试	1 个工作日	2019 年 12 月 20 日	2019 年 12 月 21 日
界面切换测试	1 个工作日	2019 年 12 月 21 日	2019 年 12 月 22 日
可靠性测试	**2 个工作日**	**2019 年 12 月 22 日**	**2019 年 12 月 23 日**
服务器与客户端连接测试	1 个工作日	2019 年 12 月 22 日	2019 年 12 月 23 日
兼容性测试	1 个工作日	2019 年 12 月 22 日	2019 年 12 月 23 日
安全性测试	**1 个工作日**	**2019 年** 12 **月 23 日**	**2019 年** 12 **月 24 日**
测试用户名密码	1 个工作日	2019 年 12 月 23 日	2019 年 12 月 24 日
编制系统测试报告	1 个工作日	2019 年 12 月 23 日	2019 年 12 月 24 日

表 2

4. 软件架构

4.1 软件框架

4.1.1 Activity

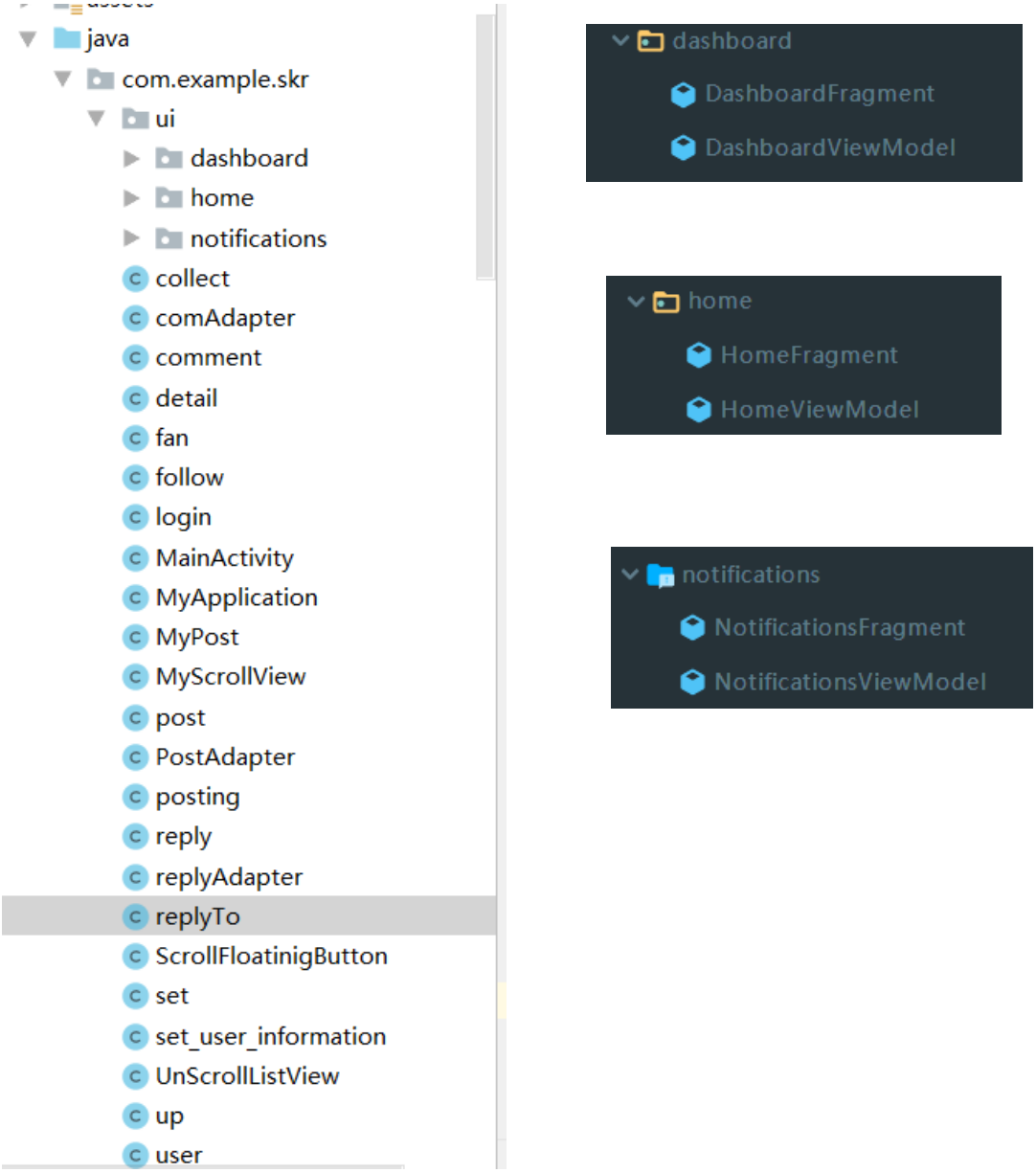

图 6

4.1.2 UI 界面

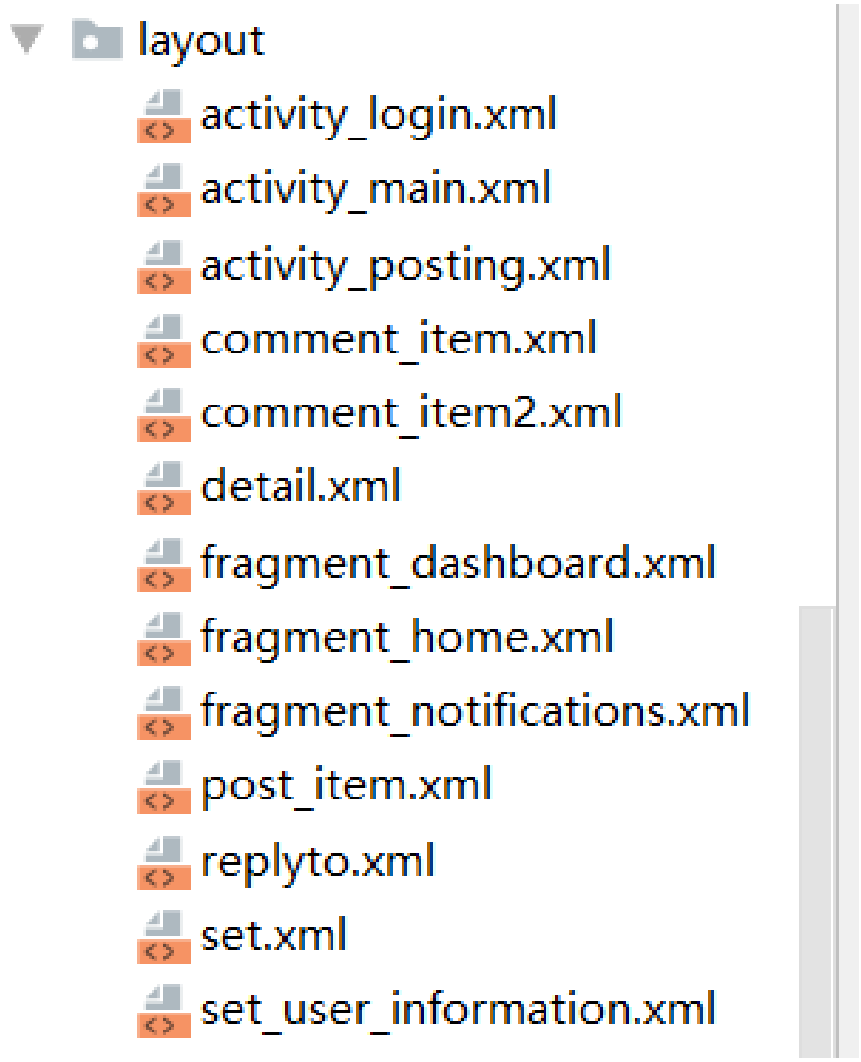

图 7

4.1.3 登录界面跳转到注册界面（同一个 loginActivity，两个 view 的切换）

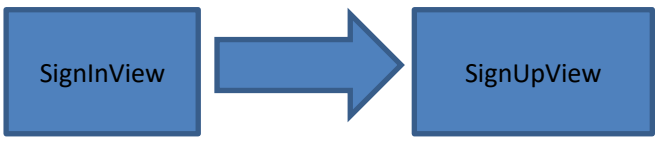

图 8

4.1.4 登录、注册界面跳转到主页面

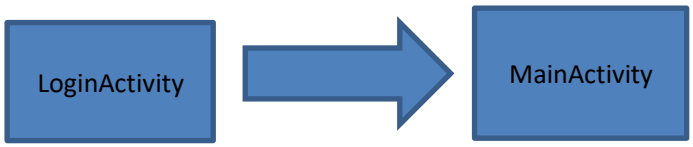

图 9

4.1.5 主页面跳转到发布页面

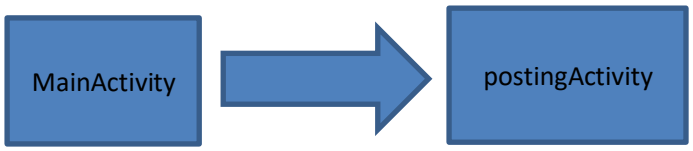

图 10

4.1.6 主页面点击帖子跳转到帖子详情页面

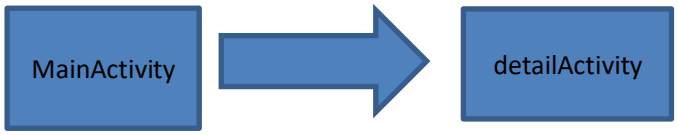

图 11

4.1.7 帖子详情页面跳转到评论回复页面

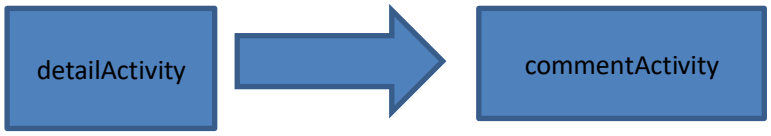

图 12

4.1.8 我的页面跳转到个人信息编辑页面

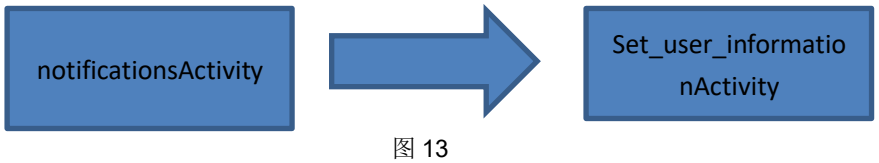

图 13

4.2 各种类说明

4.2.1 MyApplication 类

MyApplication 类

类名	MyApplication
说明	是整个应用启动时同时启动的单例实类
方法名	功能说明
onCreate	初始化 MyApplication
setWindowStatusBarColor(Activity,colorResId)	设置 Activity 对应的顶部状态栏的颜色
getNowTime	获取现在时间

表 3

4.2.2 login 类

login 类

类名	login
说明	登录注册页面
方法名	功能说明
onCreate(Bundle)	类和活动的初始化
onPostResume	其他界面返回登录界面时清除本界面过期内容
becomeClear	清除本界面内容
ClickIIB	点击登录按钮事件
ClickIUB	点击注册按钮事件，显示注册页面
ClickIFP	点击忘记密码事件
ClickUUB	点击注册按钮事件，注册用户
ClickUGB	点击返回按钮事件，显示登录界面

表 4

4.2.3 MainActivity 类

MainActivity 类

类名	MainActivity
说明	首页、排行榜、个人页面三个碎片的父活动
方法名	功能说明
onCreate(Bundle)	初始化活动，注册三个碎片
onStart	获取传来的用户账号
onKeyDown(int,KeyEvent)	监听连续点击两次返回键
exit	退出 app

表 5

4.2.4 HomeFragment 类

HomeFragment 类

类名	HomeFragment
说明	首页帖子列表
方法名	功能说明
onCreateView(LayoutInflater,ViewGroup,Budle)	创建页面，注册搜索按钮组件和浮游按钮组件
onResume	获取传来的用户账号、设置展示列表
initSnack	从数据库中获得列表所需展示的帖子的数据
onCreateOptionMenu	设置搜索按钮组件的点击事件和搜索功能实现

表 6

4.2.5 DashboardFragment 类

DashboardFragment 类

类名	DashboardFragment
说明	展示帖子排行榜

方法名	功能说明
onCreateView(LayoutInflater,ViewGroup,Budle)	初始化页面
onResume	获取传来的用户账号、设置展示列表
initSnack	从数据库中获得列表所需展示的帖子的数据，并根据帖子的收藏数按降序排序

表 7

4.2.6 NotificationsFragment 类

NotificationsFragment 类

类名	NotificationsFragment
说明	显示用户页面
方法名	功能说明
onCreateView(LayoutInflater,ViewGroup,Bundle)	初始化界面，注册页面的各种组件，设置点击事件
requestWritePermission	
onResume	获得传来的用户账号，从数据库中查找当前用户并设置界面信息

表 8

4.2.7 PostAdapter 类

PostAdapter 类

类名	PostAdapter
说明	首页帖子列表的适配器
方法名	功能说明
class ViewHolder extends RecyclerView.ViewHolder	内置类，继承 RecyclerView.ViewHolder
ViewHolder(View)	注册列表 item 的各种组件
PostAdapter	获取列表数据
onCreateViewHolder(ViewGroup,int)	初始化类
onBindViewHolder	设置每个列表项展示的内容，根据获取的列表项数据从数据库中查找补全所需信息和设置点击事件
getItemCount	获取当前列表项

表 9

4.2.8 posting 类

posting 类

类名	posting
说明	发帖页面
方法名	功能说明
onCreate(Bundle)	设置主题，注册页面各种组件并设置点击事件，获取传来的用户账号
openAlbum	打开本机相册
onRequestPermissionResult(int,String,int)	获取打开相册的授权
onActivityResult(int,int,Inter)	获取上一个界面返回的信息并判断信息类型然后进行处理

handleImageOnKitKat(Intent)	SDK_INT>19 的手机获取所选相册的照片路径的方法
handleImageBeforeKitKat(Intent)	SDK_INT<=19 的手机获取所选相册的照片路径的方法
getImagePath(Uri,String)	根据 uri 获得图片路径
displayImages(String)	根据图片路径获得图片并设置界面

表 10

4.2.9 detail 类

detail 类

类名	detail
说明	帖子详情页面
方法名	功能说明
onCreate(Bundle)	页面初始化
onStart	注册页面各种组件，获取传来的用户账号，从数据库获取并处理所需展示的帖子的各种信息，设置点击事件
initComment	获取帖子评论列表数据

表 11

4.2.10 set 类

set 类

类名	set
说明	设置页面
方法名	功能说明
onCreate(Bundle)	初始化页面，注册页面的组件，设置点击事件

表 12

4.2.11 set_user_information 类

set_user_information 类

类名	set_user_information
说明	设置个人信息页面
方法名	功能说明
onCreate(Bundle)	初始化页面，注册页面的组件，设置点击事件，获取传来的用户账号
onStart	设置点击事件
openAlbum	打开本机相册
onRequestPermissionResult(int,String,int)	获取打开相册的授权
onActivityResult(int,int,Inter)	获取上一个界面返回的信息并判断信息类型然后进行处理
handleImageOnKitKat(Intent)	SDK_INT>19 的手机获取所选相册的照片路径的方法
handleImageBeforeKitKat(Intent)	SDK_INT<=19 的手机获取所选相册的照片路径的方法
getImagePath(Uri,String)	根据 uri 获得图片路径
displayImages(String)	根据图片路径获得图片并设置界面
isGrantExternalRW(Activity)	解决 Android 6.0 或以上版本不能读取外部存储权限的问题，哪里需要读写 SD 卡的权限，就调用这个方法，必须在一个 Activity 里

表 13

4.2.12 MyPost 类

MyPost 类

类名	MyPost
说明	我发过的帖子的列表页面
方法名	功能说明
onCreate(Bundle)	初始化页面
onStart	获取传来的用户账号，从数据库获取列表所需展示的帖子的数据并展示
initSnack	从数据库获取列表数据

表 14

4.3 资源列表

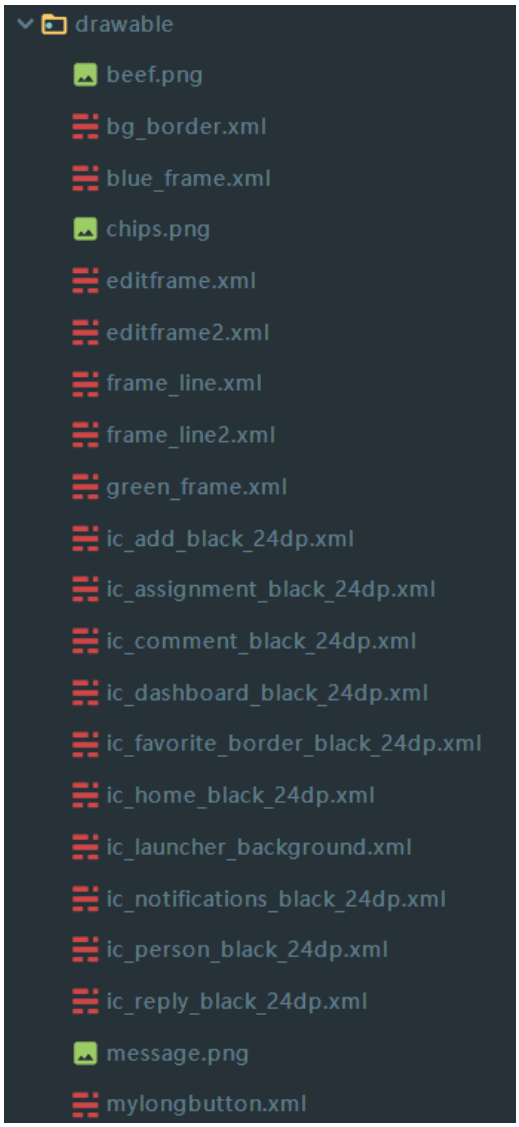

图 14

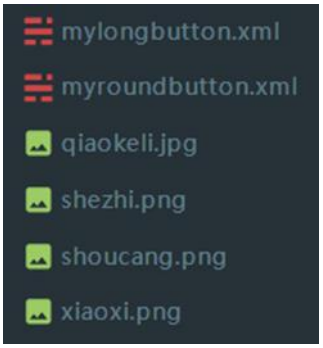

图 15

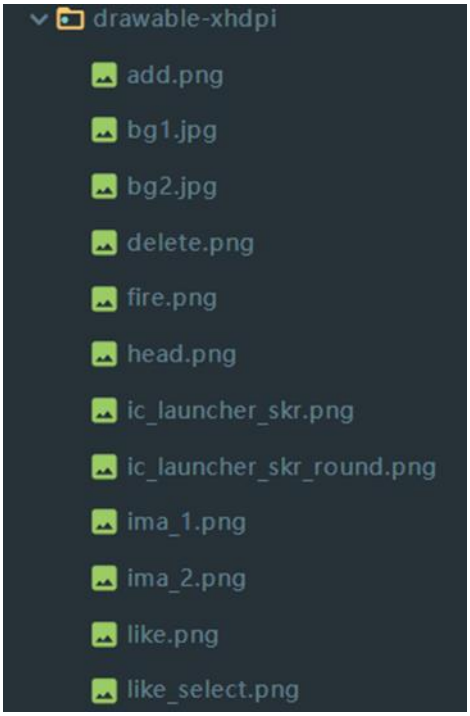
drawable-xhdpi
add.png
bg1.jpg
bg2.jpg
delete.png
fire.png
head.png
ic_launcher_skr.png
ic_launcher_skr_round.png
ima_1.png
ima_2.png
like.png
like_select.png

图 16

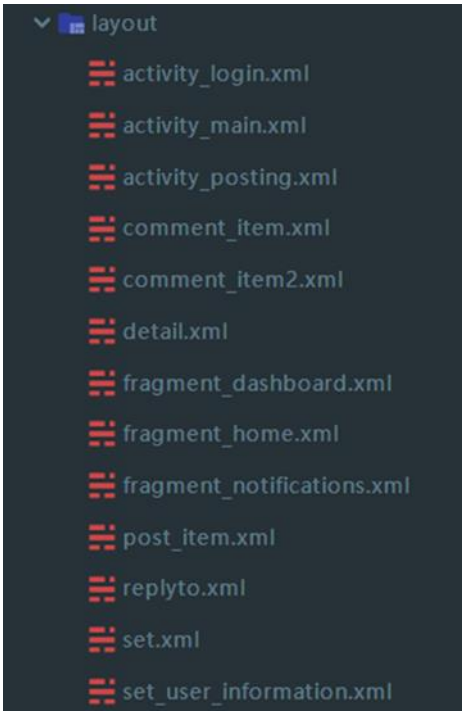
layout
activity_login.xml
activity_main.xml
activity_posting.xml
comment_item.xml
comment_item2.xml
detail.xml
fragment_dashboard.xml
fragment_home.xml
fragment_notifications.xml
post_item.xml
replyto.xml
set.xml
set_user_information.xml

图 17

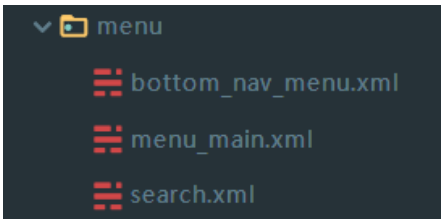
menu
bottom_nav_menu.xml
menu_main.xml
search.xml

图 18

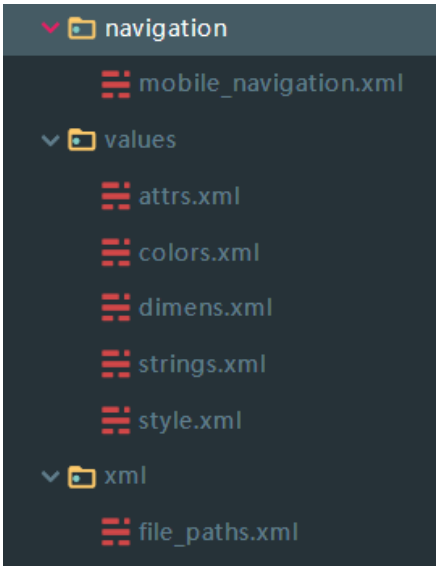
navigation
mobile_navigation.xml
values
attrs.xml
colors.xml
dimens.xml
strings.xml
style.xml
xml
file_paths.xml

图 19

5. 系统实现

5.1 各界面实现

APP 图标

图 20 图标界面

登陆界面：

① 输入正确的账号及密码，进入首页

② 账号密码输入错误，下方出现相应提示

③ 没有账号，点击注册进入注册界面

④ 忘记密码，点击忘记密码按钮找回密码

图 21 登陆界面

注册界面：

① 输入账号密码成功创建一个新用户，进入首页

② 未输入账号或密码，创建失败，下方出现提示

③ 输入的账号已被注册，下方出现提示

图 22 注册成功

图 23 注册界面

输入错误，失败

已被注册，失败

图 24 注册失败两种界面

找回密码功能：

找回初始账号及密码

图 25 找回密码界面

首页界面：

显示最新发帖信息

点击悬浮加号进入发帖界面

点击右上角放大镜进入搜索界面

图 26 首页界面

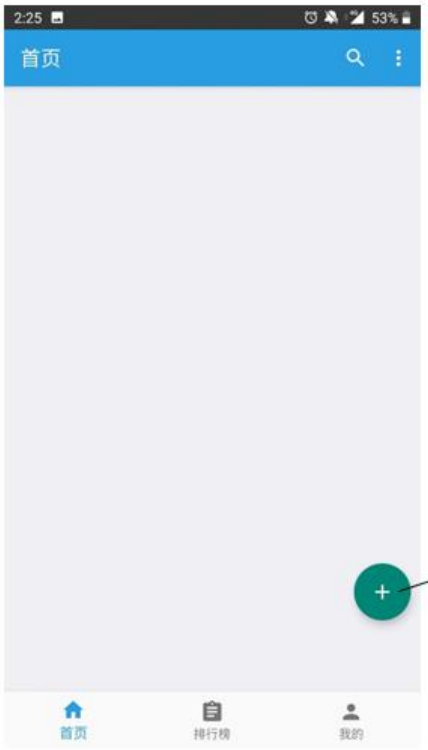

图 27 首页界面

图 28 发帖界面

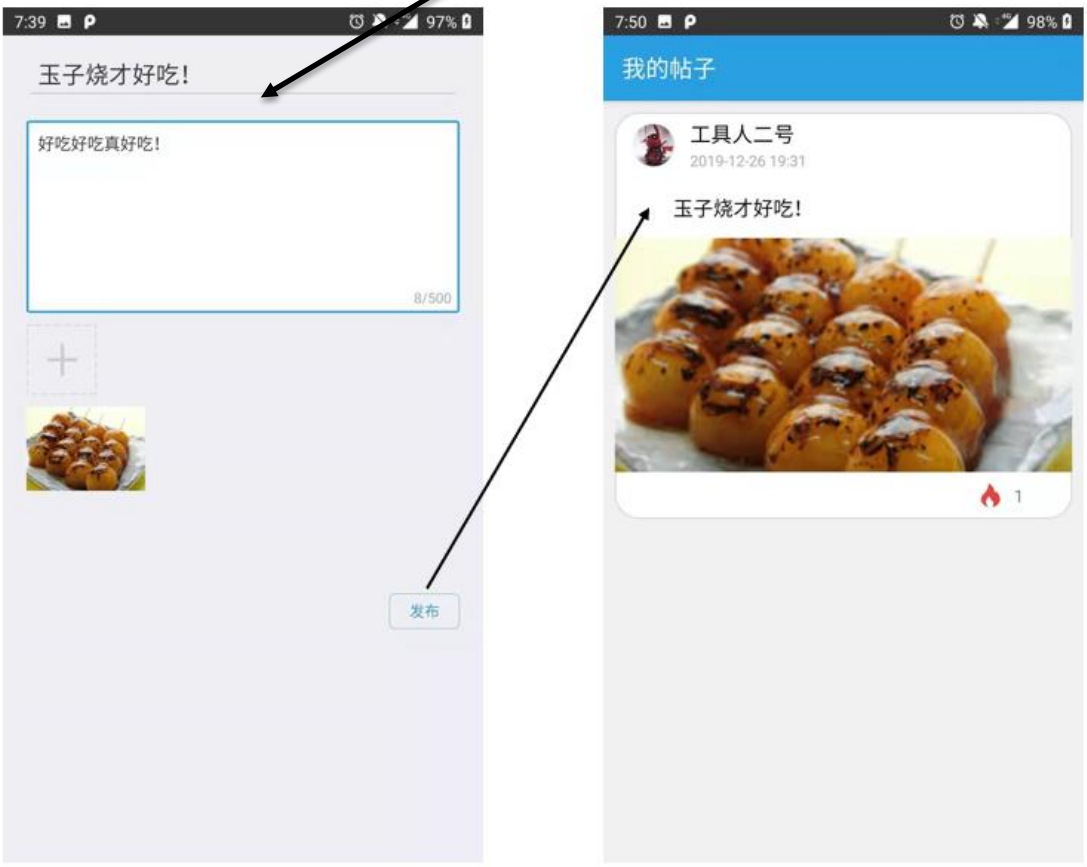

图 29 成功发布

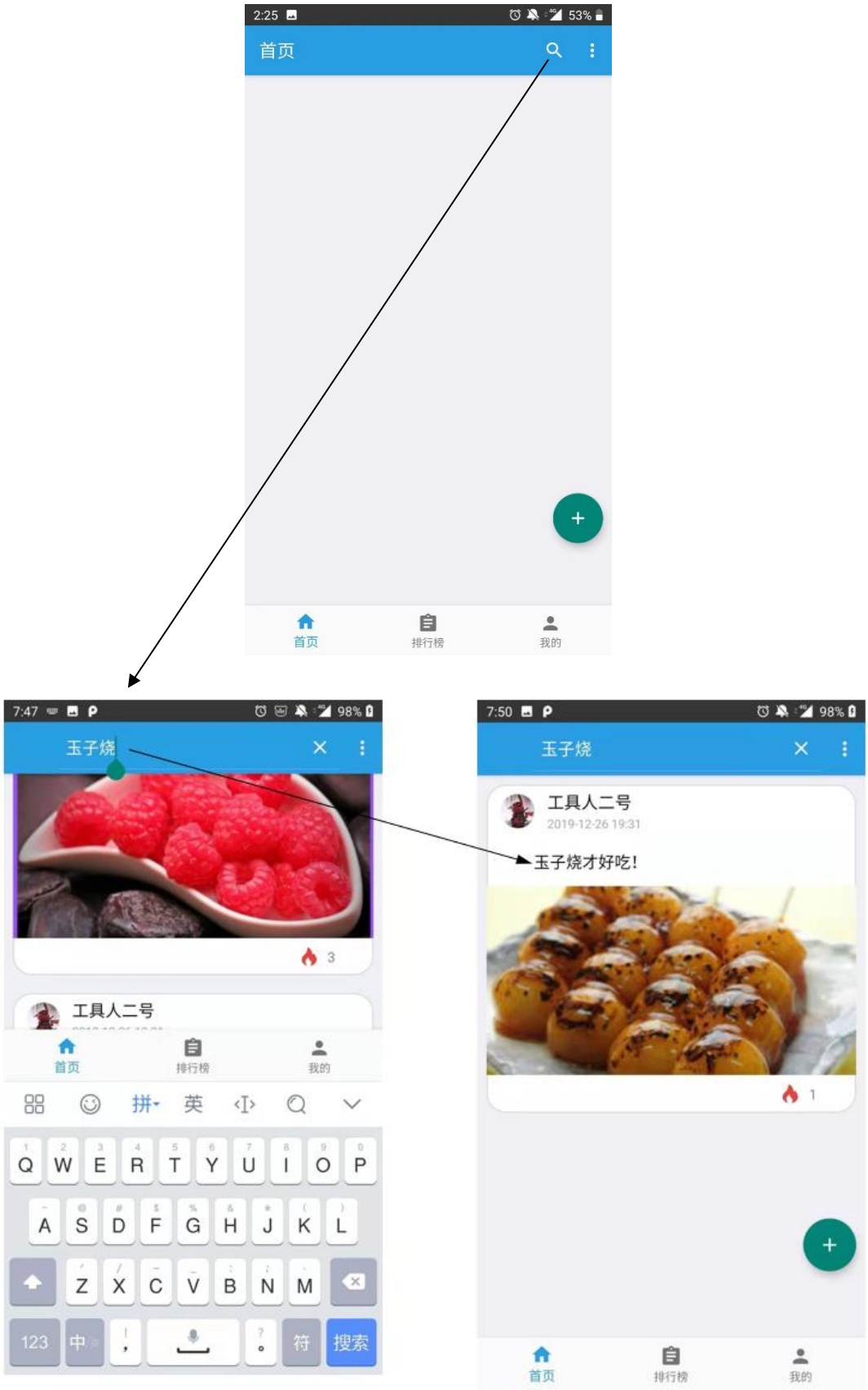

图 30 搜索界面

图 31 搜索成功界面

帖子详情界面：

① 评论原帖

② 回复其他用户的评论

③ 收藏该贴

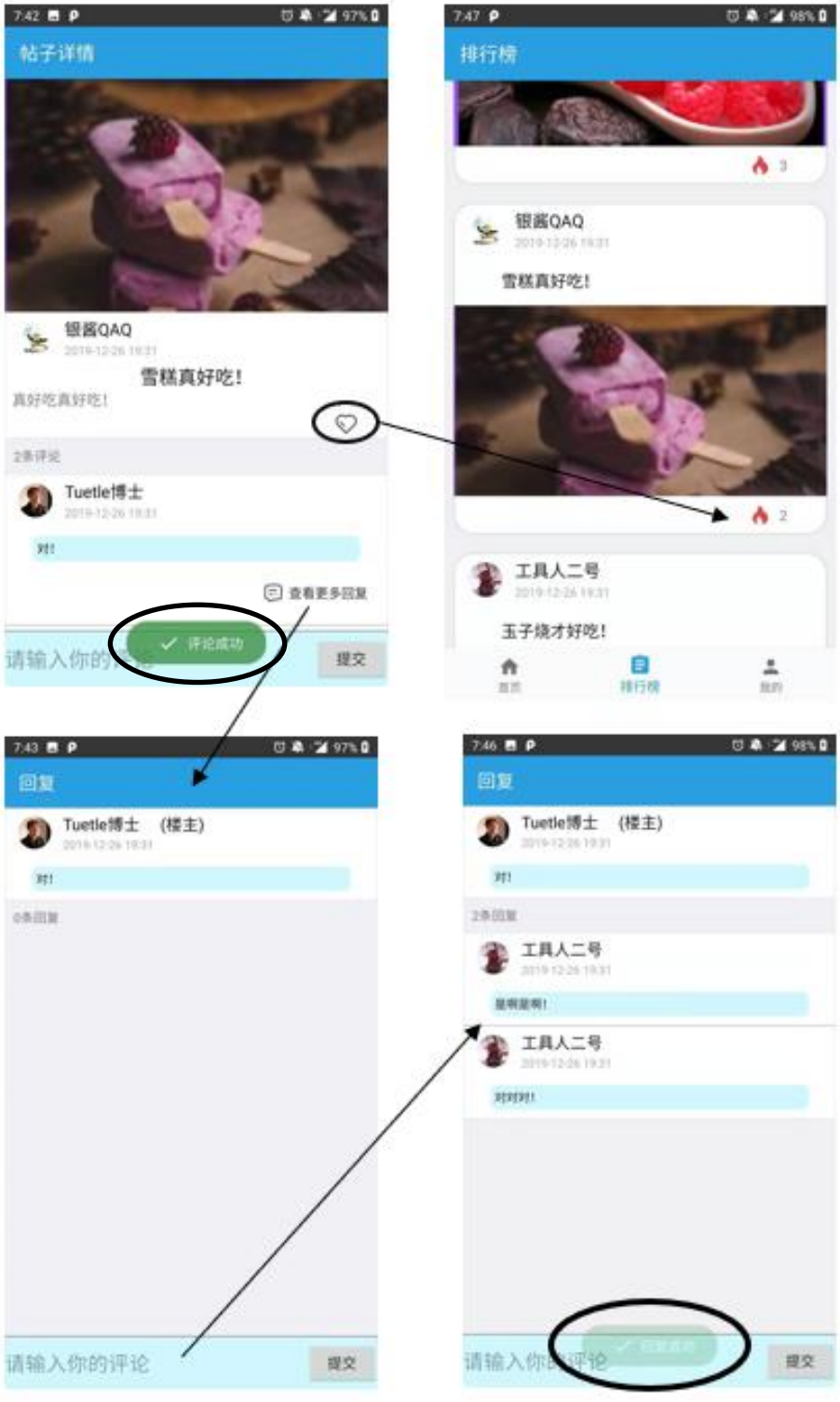

图 32 帖子详情界面及评论成功界面

排行榜界面：

根据热度排序，进行实时排行

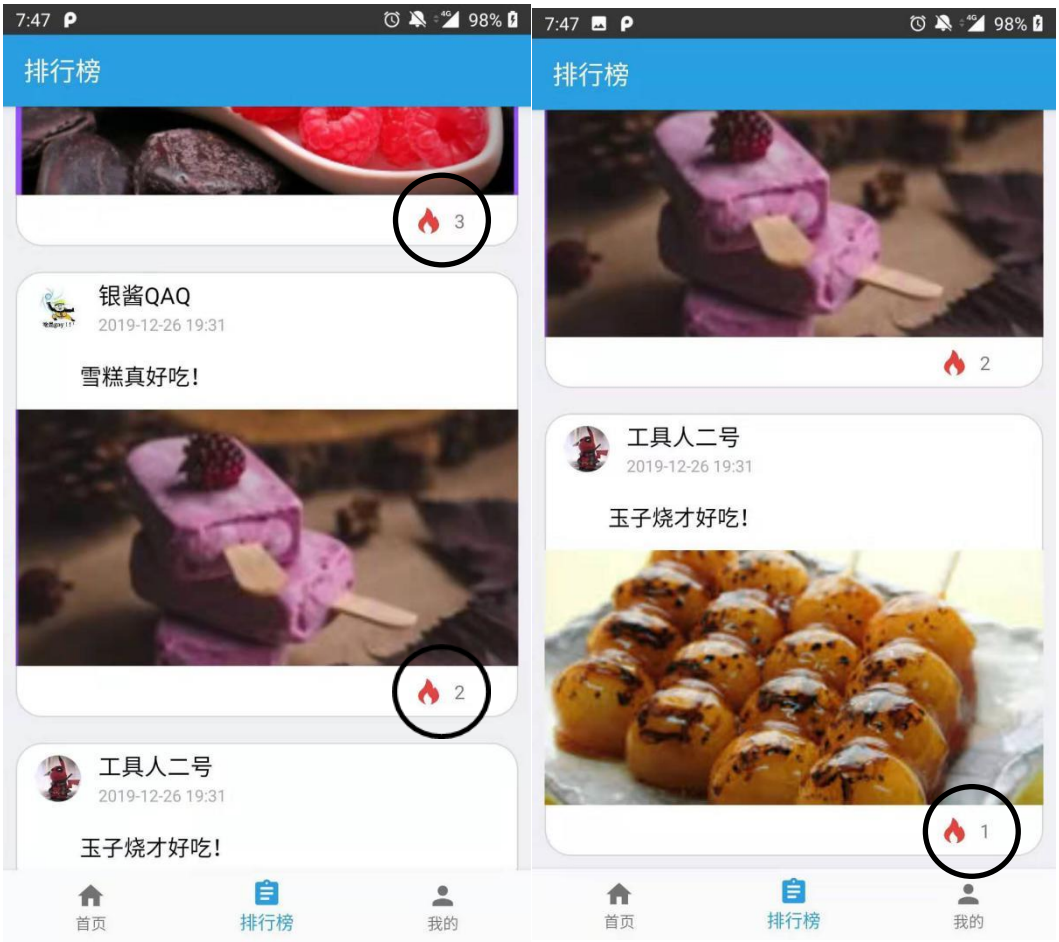

图 33　排行榜界面

我的界面：

① 更改个人信息

② 查看设置

③ 查看收藏夹

④ 查看我的历史发帖

⑤ 查看设置

图 34 我的界面

图 35　个人信息修改

图 36　设置界面

图 37 查看我的历史发帖

6．系统测试说明

6.1 测试项目及其说明

6.1.1 测试方案

用户界面测试、功能流程测试、性能测试等

6.1.2 测试项目

测试名称	内容	目的	进度
用户界面测试	运行程序，界面是否达到标准	使界面达到用户标准	15 周开始测试
功能流程测试	点击功能按钮，是否可实现其功能	使功能能够实现其效果	15 周开始测试
性能测试	通过不断运行操作，其软件的性能是否良好	使本软件更具可操作性	15 周开始测试

表 15

6.2 测试结果

6.2.1 测试结果及其相应的说明

1、用户界面测试：用户界面运行良好；功能流程顺畅；

2、功能流程测试

功能	实现情况
注册（用户正确填写信息后数据是否返回服务器、用户填写错误信息的错误提示、注册是否成功）	正确填写信息后每项数据都可以被服务器接受并写入数据库，填写信息错误的时候无法注册，但并表明错误项。
登录（系统检验用户登录密码、用户填写	利用已注册的用户名和登录密码测试，网

错误信息的错误提示、登录是否成功）	络正常情况下可以成功登录。
发布帖子（用户能否成功编辑文字，配图，以及能否成功发布）	点击发布按钮后能成功编辑内容以及发表，并能成功显示在首页。
收藏帖子（点击收藏后能否在收藏夹查看，帖子热度是否相应增加）	点击收藏以后暂不可以在收藏夹查看（暂未完成），帖子热度有进行相应增加
评论帖子（能否正常评论原帖以及别人的评论）	可以正常评论，但仅限文字
搜索帖子（在搜索栏输入关键词后能否成功查找相应内容）	可以查找到相应内容，并成功显示
修改个人信息（能否修改用户名、头像等个人信息）	可以成功修改信息
排行榜（用户查看排行榜是否按照热度进行排序）	排行榜根据实时热度进行排序

表 16

3、性能测试：性能良好，不占用大量资源

6.3 评价

优点：

1、用户体验良好，流程符合逻辑

2、UI 简洁美观

3、运行稍有卡顿现象，整体性能良好，

不足：

1、用户登录没有次数检测：可能因为暴力解锁而破坏用户的隐私。

2、注册信息检测设置不完善：可能发生恶意访问数据库等危险。

3、可拓展性较差：暂时只支持安卓平台，对 pc、ios 用户没有支持

7．用户手册

本用户手册为《食刻》项目的用户手册，目的在于指导用户如何使用本系统。预期参考人员包括用户、测试人员、开发人员、项目经理和需要阅读本报告的项目验收者。

7.1 系统说明

说明：

a. 《食刻》是一个零食推荐交流工具，主要功能是用户间进行对食物的想法分享

b. 任务提出者：由项目开发团队共同提出

c. 开发者：林俊涛、林添、张家铭、张力丹

d. 用户：主要面向年轻群体

e. 服务器：本地服务器

f. 数据库：LitePal

运行环境： Android 5.0 以上安卓操作系统

运行设备： Android 平台 5.0 版本或以上的移动设备或者模拟器

7.2 用途

7.2.1 功能

功能	解释
注册	用户可以注册个人账号，并通过完善个人信息来找到志同道合的好友
发布帖子	用户可以发布帖子表达自己的看法
收藏	收藏自己喜欢或感兴趣的帖子，方便下次查看同时为该贴增加热度
评论	对别人或是自己的帖子表达自己的看法
排行榜	根据帖子的实时热度对帖子进行排行

表 17

7.2.2 性能

1、精度

密码为数字、英文或其组合

发送的文字会精确显示在对方详情查看界面中。

2、时间特性

用户成功登陆到加载界面时间小于 1 秒

排行榜按个人需求实时刷新

3、灵活性

用户可以在所有主流安卓手机端使用本软件

7.2.3 安全保密

应用不会透露用户自己填写以外的个人信息

8. 关键代码

8.1 postAdapter 类

```
public class PostAdapter extends RecyclerView.Adapter<PostAdapter.ViewHolder> {

    private List<post> mpostList;
    String userAccount;//操作人的 userAccount

    static class ViewHolder extends  RecyclerView.ViewHolder{
        View home_post_view;
        ImageView home_post_image
                ,home_post_user_head
                ,home_post_like_notSelected
                ,home_post_like_selected;
        TextView home_post_title
                ,home_post_user_name
                ,home_post_time
                ,home_post_collect_num;

        public    ViewHolder(View view){
            super(view);
            home_post_view = (LinearLayout) view.findViewById(R.id.home_post_view);
            home_post_image = (ImageView) view.findViewById(R.id.home_post_image);
            home_post_user_head = (ImageView) view.findViewById(R.id.home_post_user_head);
            home_post_like_notSelected = (ImageView)
view.findViewById(R.id.home_post_like_notSelected);
            home_post_like_selected = (ImageView)
view.findViewById(R.id.home_post_like_selected);
            home_post_title = (TextView) view.findViewById(R.id.home_post_title);
            home_post_user_name = (TextView) view.findViewById(R.id.home_post_user_name);
            home_post_time = (TextView) view.findViewById(R.id.home_post_time);
            home_post_collect_num = (TextView)
view.findViewById(R.id.home_post_collect_num);
        }

    }

    public PostAdapter(List<post> postList , String UserAccount){
        mpostList = postList;
        userAccount = UserAccount;
        Log.d("find bug", "userAccount: "+userAccount);
    }
```

```
    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull final ViewGroup parent, int viewType) {
        View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.post_item,parent,false);
        final ViewHolder holder = new ViewHolder(view);


        return holder;

    }

    @Override
    public void onBindViewHolder(@NonNull final ViewHolder holder, int position) {
        final post mpost = mpostList.get(position);
        final String home_post_id = mpost.getPost_id();

        //获得发帖人 user 信息
        String home_post_userAccount = mpost.getUserAccount();
//        Log.d("find bug", "home_post_userAccount: "+home_post_userAccount);
        Connector.getDatabase();
        List<user> userList;
        userList = DataSupport.where("userAccount
= ?",home_post_userAccount).find(user.class);
        if (userList==null||userList.size()==0){
            Log.d("In postAdapter:", "onBindViewHolder: 得不到 userList,
home_post_userAccount="+home_post_userAccount);
        }else {
            user pUser = userList.get(0);
            Bitmap bitmapFor_home_post_user_head =
BitmapFactory.decodeFile(pUser.getPortrait());
            holder.home_post_user_head.setImageBitmap(bitmapFor_home_post_user_head);
            holder.home_post_user_name.setText(pUser.getUserName());
        }

        //获得帖子信息
        Bitmap bitmap = BitmapFactory.decodeFile(mpost.getPost_image());
        holder.home_post_image.setImageBitmap(bitmap);
        holder.home_post_title.setText(mpost.getPost_title());
        holder.home_post_time.setText(mpost.getPost_time());
        holder.home_post_collect_num.setText(mpost.getPost_collect_num()+"");//收藏数

//        //判断本人是否有收藏过(弃用)
```

```
        //点击事件
        holder.home_post_view.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent();
                intent.setClass(v.getContext(),detail.class);
                intent.putExtra("post_id",home_post_id);
                intent.putExtra("userAccount",userAccount);
                Log.d("postAdapter", "post_id : "+home_post_id);
                Log.d("postAdapter", "userAccount : "+userAccount);
                v.getContext().startActivity(intent);
            }
        });

    }

    @Override
    public int getItemCount() {
        return mpostList.size();
    }

}
```

8.2 实现登录注册功能的 login 类

```
public class login extends AppCompatActivity{

    Button signIn_signInButton,
            signIn_signUpButton,
            signIn_forgetPasswordButton,
            signUp_signUpButton,
            signUn_goBackButton;
    LinearLayout signIn_view, signUp_view;
    EditText signIn_account,
            signUp_account;
//              signIn_password,
//              signUp_password;
    PasswordEditText signIn_password, signUp_password;
//    TextView signIn_tip,signUp_tip;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        XUI.initTheme(this);
```

```
        MyApplication.setWindowStatusBarColor(this,R.color.Black);


        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        Connector.getDatabase();

        setInitialNumber();

        signIn_signInButton = (Button) findViewById(R.id.signIn_signInButton);
        signIn_signInButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ClickIIB();
            }
        });
        signIn_signUpButton = (Button) findViewById(R.id.signIn_signUpButton);
        signIn_signUpButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ClickIUB();
            }
        });
        signIn_forgetPasswordButton = (Button)
findViewById(R.id.signIn_forgetPasswordButton);
        signIn_forgetPasswordButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ClickIFP();
            }
        });
        signUp_signUpButton = (Button) findViewById(R.id.signUp_signUpButton);
        signUp_signUpButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ClickUUB();
            }
        });
        signUn_goBackButton = (Button) findViewById(R.id.signUn_goBackButton);
        signUn_goBackButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```
                ClickUGB();
            }
        });

        signIn_view = (LinearLayout) findViewById(R.id.signIn_view);
        signUp_view = (LinearLayout) findViewById(R.id.signUp_view);

        signIn_account = (EditText) findViewById(R.id.signIn_account);
//        signIn_password = (EditText) findViewById(R.id.signIn_password);
        signUp_account = (EditText) findViewById(R.id.signUp_account);
//        signUp_password = (EditText) findViewById(R.id.signUp_password);

        signIn_password = (PasswordEditText) findViewById(R.id.signIn_password);
        signUp_password = (PasswordEditText) findViewById(R.id.signUp_password);

//        signIn_tip = (TextView) findViewById(R.id.signIn_tip);
//        signUp_tip = (TextView) findViewById(R.id.signUp_tip);

    }

    @Override
    protected void onPostResume() {
        super.onPostResume();
        becomeClear();
        signUp_view.setVisibility(View.GONE);
        signIn_view.setVisibility(View.VISIBLE);
    }

    public void becomeClear(){
        signIn_account.setText("");
        signIn_password.setText("");
        signUp_account.setText("");
        signUp_password.setText("");
//        signIn_tip.setVisibility(View.GONE);
//        signUp_tip.setVisibility(View.GONE);
    }

    public void ClickIIB(){
        List<user> userList;
        List<user> userList2;
        if (signIn_account.getText().toString().trim().isEmpty() ||
signIn_password.getText().toString().trim().isEmpty()){
            XToast.error(this,"请正确输入账号和密码").show();
//            signIn_tip.setText("请正确输入账号和密码");
```

```
        }
        else {
            userList = DataSupport
                    .where("userAccount=?",signIn_account.getText().toString())
                    .find(user.class);
            userList2 = DataSupport.findAll(user.class);

            Log.d("testMessage",Integer.toString(userList.size()));
            Log.d("testMessage",Integer.toString(userList2.size()));
            for (user u : userList){
                Log.d("testMessage",
                        u.getPassword()
                                +"--"+u.getUserAccount()
                                +"--"+u.getUserName()
                                +"--"+u.getUserIntro()
                                +"--"+u.getBirthday()
                                +"--"+u.getPortrait()
                );
            }
            if (userList==null||userList.size()==0){
                XToast.error(this,"查无此账号").show();

            }
            else {
                user temp = userList.get(0);
                Log.d("testMessage",temp.getPassword() );
//                Log.d("testMessage",signIn_password.getText().toString() );
                if (signIn_password.getText().toString().equals(temp.getPassword())){
                    XToast.success(this,"登录成功").show();



                    Intent intent = new Intent(login.this,MainActivity.class);
                    intent.putExtra("userAccount",signIn_account.getText().toString());
                    startActivity(intent);




                }
                else {
                    XToast.error(this,"密码错误").show();
```

```
                }
            }
        }
    }
    public void ClickIUB(){
        becomeClear();
        signUp_view.setVisibility(View.VISIBLE);
        signIn_view.setVisibility(View.GONE);
    }
    public void ClickIFP(){
        XToast.info(this,"初始账号：20173068，密码：123456").show();

    }
    public void ClickUUB(){
        List<user> userList;
        if (signUp_account.getText().toString().trim().isEmpty() ||
signUp_password.getText().toString().trim().isEmpty()){
            XToast.error(this,"请正确输入账号和密码").show();

        }
        else {
            userList = DataSupport
                    .where("userAccount=?",signUp_account.getText().toString())
                    .find(user.class);
            if (userList==null||userList.size()==0){
                user me = new user();
                me.setUserAccount(signUp_account.getText().toString());
                me.setPassword(signUp_password.getText().toString());
                me.setBirthday("2000-01-01");
                me.setSex("男");
                me.setUserName("新用户");
                me.setUserIntro("我的个性签名");
        //                me.setPortrait("");
                me.save();

                XToast.success(this,"注册成功").show();


                Intent intent = new Intent(login.this,MainActivity.class);
                intent.putExtra("userAccount",signUp_account.getText().toString());
                startActivity(intent);
                finish();

            }
```

```
            else {
                XToast.error(this,"该账号已被注册").show();

            }
        }
    }

    public void ClickUGB(){
        becomeClear();
        signIn_view.setVisibility(View.VISIBLE);
        signUp_view.setVisibility(View.GONE);
    }


    public void setInitialNumber(){
        List<user> userList = DataSupport
                .where("userAccount=?","20173068")
                .find(user.class);
        if (userList==null||userList.size()==0){
            user me = new user();
            me.setUserAccount("20173068");
            me.setPassword("123456");
            me.setBirthday("2000-01-01");
            me.setSex("男");
            me.setUserName("初始账号");
            me.setUserIntro("我的个性签名");
            me.setPortrait("");
            me.save();
        }
    }

}
```

8.3 实现发帖功能的 posting 类

```
package com.example.skr;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.core.content.FileProvider;

import android.Manifest;
import android.annotation.TargetApi;
```

```
import android.content.ContentUris;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.content.res.Resources;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.DocumentsContract;
import android.provider.MediaStore;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;


import com.xuexiang.xui.XUI;
import com.xuexiang.xui.widget.dialog.bottomsheet.BottomSheet;
import com.xuexiang.xui.widget.edittext.ClearEditText;
import com.xuexiang.xui.widget.edittext.MultiLineEditText;
import com.xuexiang.xui.widget.toast.XToast;

import org.litepal.tablemanager.Connector;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import static org.litepal.LitePalApplication.getContext;

public class posting extends AppCompatActivity {


    public static final int CHOOSE_PHOTO=2;
    public  static  final  int TAKE_PHOTO=1;
    private Uri imageUri;
    private String posting_imagePath;
```

```
    private String userAccount;//操作人
    private ClearEditText posting_title;
    private MultiLineEditText posting_content;
    private ImageView posting_selectImageButton,posting_selectImage;
    private Button posting_postButton;
    private boolean lock=false;
//    private String[] imagesList;

    @Override//postting类的生命周期函数
    protected void onCreate(Bundle savedInstanceState) {
        XUI.initTheme(this);
        MyApplication.setWindowStatusBarColor(this,R.color.Black);
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_posting);

        Connector.getDatabase();
        Intent intent = getIntent();
        userAccount= intent.getStringExtra("userAccount");//操作本人的

        posting_title = (ClearEditText) findViewById(R.id.posting_title);
        posting_content = (MultiLineEditText) findViewById(R.id.posting_content);
        posting_selectImageButton = (ImageView)
findViewById(R.id.posting_selectImageButton);
//点击到加号按钮的点击事件
        posting_selectImageButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(lock){
                    //do nothing
                }
                else{
                    new BottomSheet.BottomListSheetBuilder(posting.this)
                            .addItem("相册")
                            .addItem("拍照")
                            .setOnSheetItemClickListener(new
BottomSheet.BottomListSheetBuilder.OnSheetItemClickListener() {
                                @Override
                                public void onClick(BottomSheet dialog, View itemView, int
position, String tag) {
                                    if(position+1==1){


                                        if(ContextCompat.checkSelfPermission(posting.this,
Manifest.permission.WRITE_EXTERNAL_STORAGE)!= PackageManager.PERMISSION_GRANTED) {
```

```
ActivityCompat.requestPermissions(posting.this,new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE},1);
                                        }
                                        else{
                                            openAlbum();
                                        }

                                    }else if(position+1==2){
                                        XToast.info(posting.this,"拍照功能未实现").show();

                                    }
                                    dialog.dismiss();
                                }
                            })
                            .build()
                            .show();
                    }
                }
            });
            posting_selectImage = (ImageView) findViewById(R.id.posting_selectImage);
            posting_postButton = (Button) findViewById(R.id.posting_postButton);
            posting_postButton.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    if(lock){
                        //do nothing
                    }
                    else{

if(posting_title.getText().toString().trim().isEmpty()||posting_content.isEmpty()||
TextUtils.isEmpty(posting_imagePath)){
                            XToast.warning(posting.this,"请完善信息").show();
                        }
                        else {
                            lock=true;
                            //存储帖子，页面跳转

                            post myNewPost = new post();
                            myNewPost.setPost_id(UUID.randomUUID().toString());
                            myNewPost.setUserAccount(userAccount);
                            myNewPost.setPost_title(posting_title.getText().toString());
                            myNewPost.setPost_content(posting_content.getContentText());
                            myNewPost.setPost_image(posting_imagePath);
```

```
                    myNewPost.setPost_time(MyApplication.getNowTime());
                    myNewPost.setPost_collect_num(0);
                    myNewPost.save();

                    XToast.success(posting.this,"发帖成功").show();

                    posting_postButton.postDelayed(new Runnable() {
                        @Override
                        public void run() {
                            finish();
                        }
                    },500);


                }
            }
        }
    });

}

@Override
protected void onStart() {
    super.onStart();
}
//打开相册所需要的函数
private void openAlbum()
{
    Intent intent=new Intent("android.intent.action.GET_CONTENT");
    intent.setType("image/*");
    startActivityForResult(intent,CHOOSE_PHOTO);
}

public void onRequestPermissionsResult(int requestCode, String[] permissions,int[]
grantResults)
{
    switch (requestCode) {
        case 1:
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                openAlbum();
            } else {
                XToast.error(posting.this,"获取授权失败").show();
            }
```

```
                break;
            default:
        }
    }

    protected void onActivityResult(int requestCode,int resultCode,Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        switch (requestCode) {
            case TAKE_PHOTO:
                if (resultCode == RESULT_OK) {
                    try {
                        Bitmap bitmap =
BitmapFactory.decodeStream(getContentResolver().openInputStream(imageUri));
                        posting_selectImage.setImageBitmap(bitmap);
                    } catch (FileNotFoundException e) {
                        e.printStackTrace();
                    }
                }
                break;
            case CHOOSE_PHOTO:
                if (resultCode == RESULT_OK) {
                    if (Build.VERSION.SDK_INT >= 19)
                        handleImageOnKitKat(data);
                    else
                        handleImageBeforeKitKat(data);
                }
                break;
            default:
                break;
        }
    }

    @TargetApi(19)//处理照片回显的问题
    private void handleImageOnKitKat(Intent data)
    {
        String imagePath=null;
        Uri uri=data.getData();
        if(DocumentsContract.isDocumentUri(this,uri)) {
            String docID = DocumentsContract.getDocumentId(uri);
            if("com.android.providers.media.documents".equals(uri.getAuthority())) {
                String id=docID.split(":")[1];
                String selection= MediaStore.Images.Media._ID+"="+id;

imagePath=getImagePath(MediaStore.Images.Media.EXTERNAL_CONTENT_URI,selection);
```

```
            } else
if("com.android.providers.downloads.documents".equals(uri.getAuthority())) {
                Uri contentUri =
ContentUris.withAppendedId(Uri.parse("content://downloads/public_downloads"),Long.valueOf(d
ocID));
                imagePath = getImagePath(contentUri,null);
            }
        } else if("content".equalsIgnoreCase(uri.getScheme())) {
            imagePath=getImagePath(uri,null);
        } else if("file".equalsIgnoreCase(uri.getScheme())) {
            imagePath=uri.getPath();
        }
        displayImage(imagePath);
    }

    private void handleImageBeforeKitKat(Intent data) {
        Uri uri=data.getData();
        String imagePath=getImagePath(uri,null);
        displayImage(imagePath);
    }

    private  String getImagePath(Uri uri,String selection) {
        String path=null;
        Cursor cursor=getContentResolver().query(uri,null,selection,null,null);
        if(cursor!=null)
        {
            if(cursor.moveToFirst())
            {
                path=cursor.getString(cursor.getColumnIndex(MediaStore.Images.Media.DATA));
            }
            cursor.close();
        }
        return path;
    }

    private void  displayImage(String imagepath) {
        if(imagepath!=null)
        {
            Bitmap bitmap = BitmapFactory.decodeFile(imagepath);
            posting_imagePath=imagepath;
            posting_selectImage.setImageBitmap(bitmap);
        }
        else {
            XToast.error(posting.this,"获取图片失败").show();
```

```
        }
    }
}
```

8.4 实现个人信息修改的 set_user_information 类

```
package com.example.skr;

import android.Manifest;
import android.annotation.TargetApi;
import android.app.Activity;
import android.content.ContentUris;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.DocumentsContract;
import android.provider.MediaStore;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import org.litepal.crud.DataSupport;
import org.litepal.tablemanager.Connector;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.localbroadcastmanager.content.LocalBroadcastManager;

import com.example.skr.ui.notifications.NotificationsFragment;

import org.litepal.util.Const;

import java.io.FileNotFoundException;
import java.util.List;
```

```
public class set_user_information extends AppCompatActivity {
    public static final int CHOOSE_PHOTO=2;
    public  static  final  int TAKE_PHOTO=1;
    private ImageView picture;
    private Uri imageUri;
    String imagePath;
    Button save;
    EditText birthday;
    EditText userName;
    EditText sex;
    TextView userAccount;
    String userOpAccount;

    @Override
//set_user_information类的生命周期函数
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.set_user_information);

        Connector.getDatabase();
        List<user> userList;
        final Intent intent=getIntent();
        userOpAccount = intent.getStringExtra("userAccount");
        Log.d("find bug", "set_user_info onstart useropaccount"+userOpAccount);
        picture=(ImageView)findViewById(R.id.picture);
        userAccount=(TextView) findViewById(R.id.userAccount);
        birthday=(EditText)findViewById(R.id.birthday);
        userName=(EditText)findViewById(R.id.userName);
        sex=(EditText) findViewById(R.id.sex);
        save=(Button)findViewById(R.id.save);

        userList=DataSupport.where("userAccount=?",userOpAccount).find(user.class);
        final user user=userList.get(0);
        if(user.getBirthday()!=null)
//        if (!TextUtils.isEmpty(user.getBirthday()))
        {
            birthday.setText(user.getBirthday());
        }
        else
        { }
        if (user.getUserAccount()!=null){
            userAccount.setText(user.getUserAccount());
        }
        if(user.getUserName()!=null)
```

```
//        if (!TextUtils.isEmpty(user.getUserName()))
        {
            userName.setText(user.getUserName());
        }
        else {}
        if(user.getSex()!=null)
//        if (!TextUtils.isEmpty(user.getSex()))
        {
            sex.setText(user.getSex());
        }else {}
        if(user.getPortrait()!=null)
//        if (!TextUtils.isEmpty(user.getPortrait()))
        {
            Bitmap bitmap = BitmapFactory.decodeFile(user.getPortrait());
            picture.setImageBitmap(bitmap);
        }
        else{}

    }

    @Override
    protected void onStart() {
        super.onStart();

//        picture.setImageBitmap();

        save.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
//点击保存之后，数据库更新原始数据的操作
                user user1=new user();
                user1.setBirthday(birthday.getText().toString());
                user1.setSex(sex.getText().toString());
                user1.setUserName(userName.getText().toString());
                user1.setPortrait(imagePath);
                user1.updateAll("userAccount=?", userAccount.getText().toString());


                finish();
                //    startActivity(intent1);
            }
        });

        picture .setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {

                if(ContextCompat.checkSelfPermission(set_user_information.this,
Manifest.permission.WRITE_EXTERNAL_STORAGE)!= PackageManager.PERMISSION_GRANTED)
                {
                    ActivityCompat.requestPermissions(set_user_information.this,new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE},1);
                }
                else{
                    openAlbum();
                }
            }
        });

    }
//打开相册的功能函数
    private void openAlbum()
    {
        Intent intent=new Intent("android.intent.action.GET_CONTENT");
        intent.setType("image/*");
        startActivityForResult(intent,CHOOSE_PHOTO);
    }
    public void onRequestPermissionsResult(int requestCode, String[] permissions,int[]
grantResults)
    {
        switch (requestCode) {
            case 1:
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    openAlbum();
                } else {
                    Toast.makeText(this, "you denied the permission", Toast.LENGTH_SHORT);
                }
                break;
            default:
        }

    }
    protected void onActivityResult(int requestCode,int resultCode,Intent data) {

        super.onActivityResult(requestCode, resultCode, data);
        switch (requestCode) {
            case TAKE_PHOTO:
```

```
                if (resultCode == RESULT_OK) {
                    try {
                        Bitmap bitmap =
BitmapFactory.decodeStream(getContentResolver().openInputStream(imageUri));
                        picture.setImageBitmap(bitmap);
                    } catch (FileNotFoundException e) {
                        e.printStackTrace();
                    }
                }
                break;
            case CHOOSE_PHOTO:
                if (resultCode == RESULT_OK) {
                    if (Build.VERSION.SDK_INT >= 19)
                        handleImageOnKitKat(data);
                    else
                        handleImageBeforeKitKat(data);
                }
                break;
            default:
                break;
        }
    }
    @TargetApi(19)
//照片回显的功能函数
    private void handleImageOnKitKat(Intent data)
    {
        imagePath=null;
        Uri uri=data.getData();
        if(DocumentsContract.isDocumentUri(this,uri))
        {
            String docID=DocumentsContract.getDocumentId(uri);
            if("com.android.providers.media.documents".equals(uri.getAuthority()))
            {
                String id=docID.split(":")[1];
                String selection= MediaStore.Images.Media._ID+"="+id;

imagePath=getImagePath(MediaStore.Images.Media.EXTERNAL_CONTENT_URI,selection);
            }
            else if("com.android.providers.downloads.documents".equals(uri.getAuthority()))
            {
                Uri contentUri=
ContentUris.withAppendedId(Uri.parse("content://downloads/public_downloads"),Long.valueOf(d
ocID));
                imagePath=getImagePath(contentUri,null);
```

```
        }
    }
    else if("content".equalsIgnoreCase(uri.getScheme()))
    {
        imagePath=getImagePath(uri,null);
    }
    else if("file".equalsIgnoreCase(uri.getScheme()))
    {
        imagePath=uri.getPath();
    }

    displayImage(imagePath);
}
private void handleImageBeforeKitKat(Intent data)
{
    Uri uri=data.getData();
    String imagePath=getImagePath(uri,null);

    displayImage(imagePath);
}
private  String getImagePath(Uri uri,String selection)
{
    String path=null;
    Cursor cursor=getContentResolver().query(uri,null,selection,null,null);
    if(cursor!=null)
    {
        if(cursor.moveToFirst())
        {
            path=cursor.getString(cursor.getColumnIndex(MediaStore.Images.Media.DATA));
        }
        cursor.close();
    }

    return path;
}
private void  displayImage(String imagepath)
{
    if(imagepath!=null)
    {
        if (Build.VERSION.SDK_INT >= 23) {
            int REQUEST_CODE_CONTACT = 101;
            String[] permissions = {Manifest.permission.WRITE_EXTERNAL_STORAGE};
            //验证是否许可权限
            for (String str : permissions) {
```

```
                    if (this.checkSelfPermission(str) != PackageManager.PERMISSION_GRANTED)
{
                        //申请权限
                        this.requestPermissions(permissions, REQUEST_CODE_CONTACT);
                        return;
                    }
                }
            }
            BitmapFactory.Options op = new BitmapFactory.Options();

            op.inSampleSize = 2;
            Bitmap bitmap = BitmapFactory.decodeFile(imagepath,op);
            picture.setImageBitmap(bitmap);
            Toast.makeText(this, imagepath, Toast.LENGTH_SHORT).show();
        }
        else {
            Toast.makeText(this,"fail to get image",Toast.LENGTH_SHORT).show();
        }
    }

    /**
     * 解决 Android 6.0 或以上版本不能读取外部存储权限的问题，哪里需要读写 SD 卡的权限，就调
用这个方法，必须在一个 Activity 里
     * @param activity
     * @return
     */
    public static boolean isGrantExternalRW(Activity activity) {
        if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.M &&
activity.checkSelfPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED){
            activity.requestPermissions(new String[]{
                    Manifest.permission.READ_EXTERNAL_STORAGE,
                    Manifest.permission.WRITE_EXTERNAL_STORAGE
            },1);
            return false;
        }
        return true;

    }
}
```

8.5 实现帖子详情的 detail 类

```
package com.example.skr;

import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.ldoublem.thumbUplib.ThumbUpView;
import com.xuexiang.xui.widget.toast.XToast;

import org.litepal.crud.DataSupport;
import org.litepal.tablemanager.Connector;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

public class detail extends AppCompatActivity {
    private List<comment> commentList = new ArrayList<>();
    private  String userAccount;//操作人的 userAccount
    private  String post_id;
    private  List <post> posts;
    private  List<user> postUser;//作者的
    ImageView post_like_notSelected
            ,post_like_selected;
    List<collect> collectList = new ArrayList<>();
    ImageView postImage;
    TextView postuserName;
    TextView postTitle;
    TextView postContent;
```

```
    TextView postTime;
    ImageView userPortrait;
    TextView commentText;
    TextView commentNum ;
//    ThumbUpView mThumbUpView;
    private boolean lock=false;
    private post post1;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.detail);


    @Override
    public void onStart() {
        Connector.getDatabase();
        Intent intent = getIntent();
        userAccount= intent.getStringExtra("userAccount");//操作本人的
        post_id = intent.getStringExtra("post_id");
        postImage = (ImageView) findViewById(R.id.post_image);
        postuserName = (TextView)findViewById(R.id.post_user_name);
        postContent =(TextView) findViewById(R.id.post_content) ;
        postTime =(TextView) findViewById(R.id.post_user_time);
        postTitle =(TextView) findViewById(R.id.post_title) ;
        userPortrait =(ImageView)findViewById(R.id.post_user_head) ;
        commentText =(TextView) findViewById(R.id.message_card_topic_replied_detail);
        commentNum =(TextView) findViewById(R.id.commentNum);

        post_like_notSelected = (ImageView) findViewById(R.id.post_like_notSelected);
        post_like_selected = (ImageView) findViewById(R.id.post_like_selected);

        //判断是否曾收藏
        collectList = DataSupport.where("post_id = ? and userAccount
= ?",post_id,userAccount).find(collect.class);
        if (collectList.size()==0||collectList==null){
            //未收藏
            post_like_notSelected.setVisibility(View.VISIBLE);
            post_like_selected.setVisibility(View.GONE);
        }
        else {
            //已收藏
            post_like_selected.setVisibility(View.VISIBLE);
            post_like_notSelected.setVisibility(View.GONE);
```

```
        }


        initComment();
        commentNum.setText(commentList.size()+"条评论");



        //没收藏变收藏
        post_like_notSelected.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                post_like_selected.setVisibility(View.VISIBLE);
                post_like_notSelected.setVisibility(View.GONE);
                if (collectList.size()==0||collectList==null){
                    collect newCollect = new collect();
                    newCollect.setCollect_id(UUID.randomUUID().toString());
                    newCollect.setPost_id(post_id);
                    newCollect.setUserAccount(userAccount);
                    newCollect.setCollect_time(MyApplication.getNowTime());
                    newCollect.save();
                    collectList.add(newCollect);
                    post1.setPost_collect_num(post1.getPost_collect_num()+1);
                    post1.updateAll("post_id = ?",post_id);
//                        XToast.success(detail.this,"收藏成功").show();
                }
                else {
//                        XToast.error(detail.this,"本帖已收藏").show();
                }

            }
        });

        //收藏变没收藏
        post_like_selected.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                post_like_notSelected.setVisibility(View.VISIBLE);
                post_like_selected.setVisibility(View.GONE);
                if (collectList.size()==0||collectList==null){

                }
                else {
                    collect newCollect = collectList.get(0);
```

```
                    DataSupport.deleteAll(collect.class ,"post_id = ? and userAccount = ?
",post_id,userAccount);

                    collectList.clear();
                    post1.setPost_collect_num(post1.getPost_collect_num()-1);
                    post1.updateAll("post_id = ?",post_id);
                }
            }
        });




        super.onStart();

        Button submmit = (Button)findViewById(R.id.sub_comment);
        submmit.setOnClickListener(new View.OnClickListener() {
            @Override

//实现评论功能的代码
            public void onClick(View view) {
                if(lock){
                    //do nothing
                }
                else{
                    if(commentText.getText().toString().trim().isEmpty()){
                        XToast.warning(detail.this,"评论不能为空").show();
                    }
                    else {
                        lock=true;
               comment comment = new comment();
                comment.setComment_content(commentText.getText().toString());
                 comment.setComment_id(UUID.randomUUID().toString());
                comment.setPost_id(post_id);
                comment.setUserAccount(userAccount);//操作人的
                comment.setComment_time(MyApplication.getNowTime());
                comment.save();
                XToast.success(detail.this,"评论成功").show();
                commentText.setText("");
                lock=false;
                onStart();


                    }
```

```
                }
            }
        });

        comAdapter adapter = new
comAdapter(detail.this,R.layout.comment_item,commentList,userAccount);//把操作人的
userAccount 传进 Adapter
        UnScrollListView  unScrollListView= (UnScrollListView) findViewById(R.id.Comment);
        unScrollListView.setAdapter(adapter);

    }
    private  void initComment(){


        //在数据库里找到对应的 post 和 user，然后把他们的信息输出来
posts = DataSupport.where("post_id=?",post_id).find(post.class);


        post1=posts.get(0);
String post_image = post1.getPost_image();
String post_title = post1.getPost_title();
String post_content = post1.getPost_content();
String post_time  = post1.getPost_time();
String post_userAccount = post1.getUserAccount();
postUser = DataSupport.where("userAccount=?",post_userAccount).find(user.class);//作者的
user postuser = postUser.get(0);
String post_userName = postuser.getUserName();
String post_userPortrait = postuser.getPortrait();
commentList = DataSupport.where("post_id=?",post_id).find(comment.class);
Bitmap bitmap = BitmapFactory.decodeFile(post_image);
postImage.setImageBitmap(bitmap);
postuserName.setText(post_userName);
postContent.setText(post_content);
postTime.setText(post_time);
postTitle.setText(post_title);
Bitmap bitmap2 = BitmapFactory.decodeFile(post_userPortrait);
userPortrait.setImageBitmap(bitmap2);

    }
    }
```

8.6 实现评论回复的 replyTo 类

```
package com.example.skr;

import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;



import com.xuexiang.xui.widget.toast.XToast;

import org.litepal.crud.DataSupport;
import org.litepal.tablemanager.Connector;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

public class replyTo extends AppCompatActivity {
    private List<comment> commentList = new ArrayList<>();
 private  String userAccount;//回复的那个评论的 id
 private  String  comment_id;//回复的那个评论的 id
    private List<user> users;
    private boolean lock=false;
  EditText replytText;


    TextView commentTime;
    ImageView userPortrait;
    TextView commentText;
    TextView commentUserName;
    TextView replyNum;
private  List<reply>replyList;
```

```
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.replyto);









    }
    @Override
    public void onStart() {
        Connector.getDatabase();//连接数据库
        Intent intent = getIntent();
        userAccount= intent.getStringExtra("userAccount");//拿到操作人的 userAccount
        comment_id = intent.getStringExtra("comment_id");
        commentText =(TextView) findViewById(R.id.comment_replied_detail) ;
        commentTime = (TextView) findViewById(R.id.comment_replied_userTime) ;
        commentUserName =(TextView)  findViewById(R.id.comment_replied_userName) ;
        userPortrait =(ImageView) findViewById(R.id.comment_replied_userPortrait) ;
        replytText = (EditText) findViewById(R.id.reply_detail) ;
        replyNum =(TextView) findViewById(R.id.replyNum);
        initComment();
        super.onStart();

       Button submmit = (Button) findViewById(R.id.sub_reply);
       submmit.setOnClickListener(new View.OnClickListener() {
           @Override
//实现评论回复功能的代码
           public void onClick(View view) {
               if(lock){
                   //do nothing
               }
               else{
//判断回复评论框内是否为空
                   if(replytText.getText().toString().trim().isEmpty()){
                       XToast.warning(replyTo.this,"回复不能为空").show();
                   }
                   else {
                       lock=true;


                       reply reply = new reply();
                       reply.setReply_content(replytText.getText().toString());
                       reply.setComment_id(comment_id);
```

```
                    reply.setUserAccount(userAccount);//操作人的
                    reply.setReply_time(MyApplication.getNowTime());
                    reply.save();
                    XToast.success(replyTo.this,"回复成功").show();
                    replytText.setText("");
                    lock=false;
                    onStart();


                }
            }
        }
    });


        replyAdapter adapter = new
replyAdapter(replyTo.this,R.layout.comment_item2,replyList,userAccount);
        UnScrollListView  unScrollListView= (UnScrollListView) findViewById(R.id.Reply);
        unScrollListView.setAdapter(adapter);
    }
    private  void initComment(){
        commentList =  DataSupport.where("comment_id=?",comment_id).find(comment.class);

        comment comment = commentList.get(0);
        commentText.setText(comment.getComment_content());
        commentTime.setText(comment.getComment_time());
        String userComment = comment.getUserAccount();
        users = DataSupport.where("userAccount=?",userComment).find(user.class);
        user user = users.get(0);
        commentUserName.setText(user.getUserName()+"      ("+"楼主"+")");
        Bitmap bitmap2 = BitmapFactory.decodeFile(user.getPortrait());
        userPortrait.setImageBitmap(bitmap2);

     replyList = DataSupport.where("comment_id=?",comment_id).find(reply.class);
     replyNum.setText(replyList.size()+"条回复");
    }
}
```

9．附录

9.1 小组分工

小组分工	
林俊涛	组长 小组分工 功能设计 编码，实现帖子详情、评论以及回复界面 测试
林添	功能设计 编码，实现我的、设置、个人信息修改界面 测试
张力丹	需求分析 编码，实现部分页面的 Layout 和排行榜 界面设计 编写文档
张家铭	功能设计、数据库设计、部分 UI 设计 编码实现用户注册登录功能、发帖功能、收藏功能、查看我发过的帖子页面、排行榜 测试、维护

表 18

9.2 版本记录

版本 1	单机版各个功能都能实现

表 19

9.3 设计心得

林俊涛： 通过这次 APP 的开发，我收益良多。我是第一次真真正正地系统地开发一个 APP，在这次开发过程中，我们遇到了挺多的困难，通过小组成员的共同努力，解决了一个又一个的 Bug，让我们的食刻 APP 得以实现。同时也巩固了老师上课所讲的内容，更加熟悉了 Android 的开发。而我身为一个组长，也要做好组长的职能，除了分配好各自的分工，之外，我还在 GitHub 上建立了自己的项目仓库，告知成员用 git 进行版本管理，建立各自的分支进行开发，让项目开发更加专业化，实际化。其中我们遇到的一个比较典型的问题就是生命周期函数的问题，我们知道 Activity 的生命周期函数如何执行，但是 Fragment 的生命周期函数跟 Activity 有所不同。例如我们建的 bottom navigation activity，来实现底部导航栏，但是他是一个 mainActivity 绑定了三个 Fragment，而 mainActivity 的生命周期是从底到上执行的，也就是先执行 Fragment 的生命周期函数再执行 mainActivity 的生命周期函数。这就导致了后面涉及到页面间通信的时候，需要注意到 intent 发送的数据能不能被捕捉到的问题。这是我们开发过程中遇到的一个比较象征性的问题。除此之外，我认为前期的需求分析和领域模型的讨论尤其重要，不然后面呈现出来 bug 会很多和进行变更的代价是很大的。 所以这次开发过程给了我很大的成长。
张家铭 这次开发我主要负责用户注册登录功能、发帖功能、收藏功能、查看我发过的帖子页面、排行榜、部分 UI 设计和参与数据交互设计等。我负责的功能主要都是写数据的功能，只要我这边一错，其他人做的功能也会出现一连串的问题，所以我在开发时也承受了很大的压力。由于我缺乏开发安卓 APP 的经验，在开发过程中也遇到了很多问题，需要边开发边学习相关内容，着手解决每一个发现的问题，虽然效率不高，但是收获甚多。 这次开发遇到了很多问题，困扰我们最久的就是如何记录和传递当前操作者的用户账户，让 app 知道现在谁在操作的问题。起初我想用继承 Application 类的 MyApplication 类来使用

全局变量，在登录跳转时设置当前操作者账号，但后来出现了之后的页面拿不到操作者账号或是退出登录重新进入却是上一个账号的错误反馈。我以为是我设的全局变量出现了问题，在修改变量之前其他页面就开始使用该变量了，于是决定弃用了全局变量，改为所有页面传递和接收 userAccount。然而问题还没能解决，尝试在退出登录回登录界面时销毁所有活动也无用，在经历几次大改，查阅和尝试各种方法之后终于找到了问题的根源，发现首页三个 fragment 的生命周期和 activity 的不一样，android 是先调用 fragment 的几个生命周期函数后，它所绑定的父 activity 才开始 onCreate，我们想在 fragment 生命周期首个调用的函数 onAttach 获取父 activity 传过来的用户账号，然而此时父 activity 都还没收到登录页面传过来的用户账号，于是发生空指针报错。还有首页页面和浮游按钮会被下部的 navigation 挡住的问题，也是由于用了大家都不熟悉的 constraintLayout 出现的问题，又是一番波折之后换回 LinearLayout 就解决了。现在看来我们被这些小问题拖了好几天，表明我们真的十分有必要多实践采坑积累经验才行，而且不能学一半就用，学而不思则罔说的就是我们这种情况。 我在这次大作业开发学到了不少东西，虽然过程很漫长，吃了没经验的亏；我认识到了团队合作交流的重要性，心态也很重要，问题很多，没有好的心态不可能很好解决问题，总的来说收获很多。
林添 在这次的安卓开发中，组长为我们明确的分配了分工，在实现”我的”界面过程中，并非很顺利，尽管用户其他信息的存储很容易实现，但是在存储用户图片的过程中难以保存。同时，组员们对我的 ui 设计也帮忙做了很多的改进，在最后的整合中，我充当了测试员，负责帮忙找出 bug 并分析为啥会出现这种情况。总的来说，这次的安卓开发完完整整的运用上了上课所学知识，是一次很好的学习经历
张力丹 这次的的团队合作很愉快，虽然一开始有些吃力，有一些还不太会，但通过组员的指导，自己的努力，编写安卓程序的能力有所提高，而且也学了一些团队合作的工具和方法，通过这次开发，对安卓，LitePal 有了更加深刻的理解，我除了做前期的需求分析外，还做了后期的测试和编档，测试的话除了要测试在不同版本安卓上的使用状况，还要对一些原有的设计进行修改。

表 20

9.4 参考文献

[1]李刚.疯狂 Android 讲义[M].电子工业出版社:北京,2013

[2]Y.Daniel Liang.Java 语言程序设计[M].机械工业出版社:北京,2012

[3]郭霖.第一行代码——Android[M].人民邮电出版社:北京,2016