



奶茶控 app

系统开发说明文件

课程名称： 移动智能应用开发

项目名称： 奶茶控

小组： 2021autumn-B3-bubble tea

成员： 罗嘉欣 20192005122

林舒恩 20192005048

宋轶德 20193232031

指导老师： 李慧老师 曹阳老师

目录

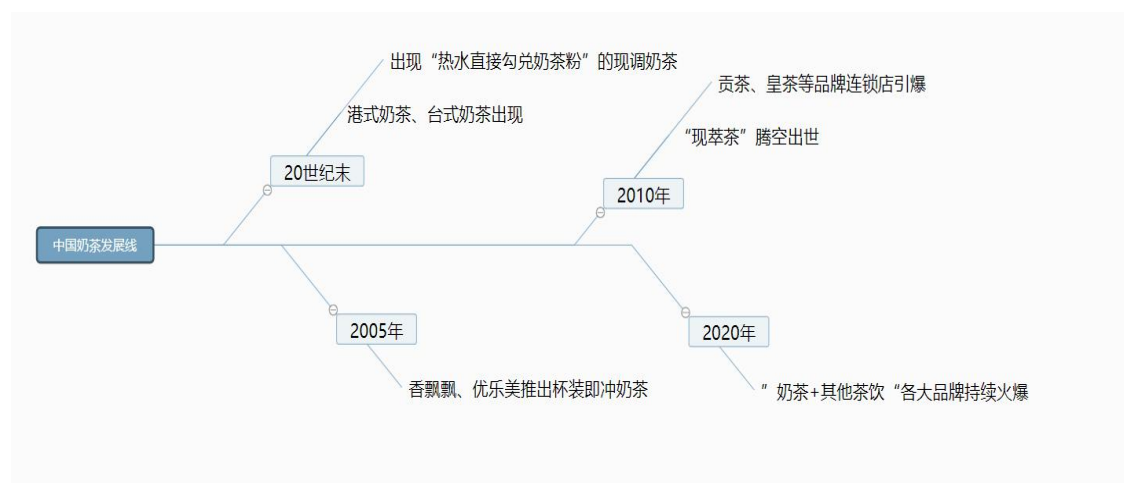
| | |
|-----------------------|----|
| 一、 产品设计方案..... | 3 |
| • 项目实施可行性报告..... | 3 |
| • 产品定位及目标..... | 7 |
| • 产品内容总策划..... | 8 |
| • 技术解决方案..... | 9 |
| 奶茶控 app 的推荐算法..... | 10 |
| • 推广方案..... | 10 |
| • 运营规划书..... | 11 |
| 二、 产品实现方案..... | 12 |
| (一) 系统的主要功能..... | 12 |
| (二) UI 界面设计..... | 12 |
| (三) 关键技术和技术难点..... | 17 |
| (四) 用户体验记录和分析..... | 25 |
| (五) 已完成的改进和存在的问题..... | 26 |
| 三、 测试大纲和测试报告..... | 29 |
| 1. 测试大纲..... | 29 |
| 2. 测试报告..... | 31 |
| 四、 产品安装和使用说明..... | 32 |
| (一) 产品安装..... | 32 |
| (二) 使用说明..... | 32 |

一、产品设计方案

▪ 项目实施可行性报告

（一）行业市场分析

奶茶行业曾经以门槛低+毛利高吸引各路创业者进入，实现爆发式增长，后随着人们对奶茶的热爱程度不断加剧，奶茶行业竞争加剧，奶茶发展到如今的新式茶饮时代，从单纯的奶精冲泡到加入珍珠、水果、牛奶、茶叶、芝士等等材料，口味和名称花样百出。所以平时我们生活中所说的“奶茶”不再是单纯的“奶+茶”，而是现在的新式茶饮，目前新式茶饮市场上奶盖茶、鲜果茶和冷萃茶占据了大部分市场份额。



奶茶具有庞大的潜在消费者，根据中国人口、城镇化率、奶茶价格等因素的综合估算，预计我国奶茶市场容量能够达到 986 亿元，接近千亿元。奈雪的茶更是以 A+轮 60 亿估值成为我国现制茶饮行业的第一个独角兽企业，喜茶单轮融资额达到 4 亿元，众多成立于 2014-2016 年的现制茶饮品牌在 2018 年也迎来了集中融资风暴，市场估值一路上升，同时吸引了很多投资机构的目光。

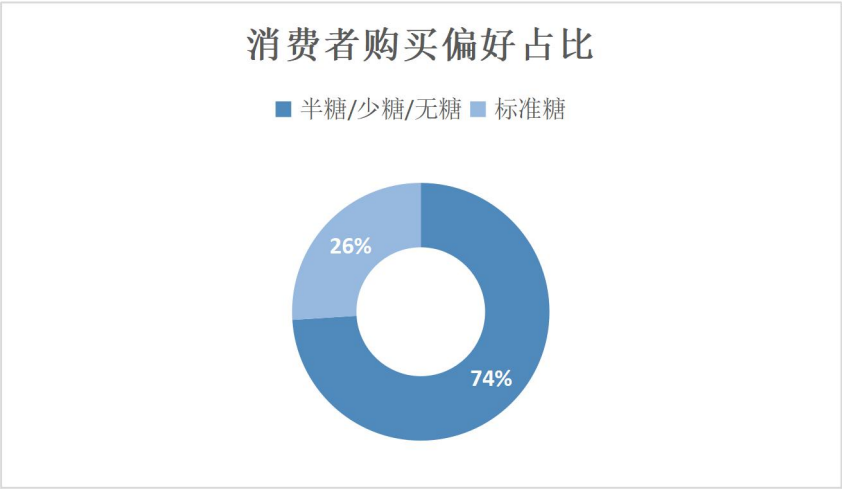
据数据显示，2014-2018 年中国奶茶市场零售额符合增长率超过 20%，2017 年销售额达 472 亿元，同比增长 14.29%，2018 年实现超过 500 亿元的销售额，据相关调查显示：约 76.1%的消费者每月会在现制奶茶上进行 1-5 次消费，14.2%的消费者会进行 6-10 次消费，还有 9.7%的消费者会进行 10 次以上的消费，消费者的月平均消费奶茶达到 4.1 杯。就我国奶茶企业总体容量和消费者对奶茶饮品的喜爱程度来看，我国奶茶行业市场仍是增量市场，还有一定供给还没有达到饱和状态。但是到了 2020 年，在城市分布方面，新式茶饮 2020 年线下与线上呈现出不同趋势。2020 年新式茶饮门店在一、二线城市的增速放缓，呈现向三、四线城市下沉的趋势。这说明一二线城市的新式茶饮行业区域饱和，三四线城市市场仍是一片蓝海。

由奈雪的茶联合 CBNDATA 发布的《2020 新式茶饮白皮书》中指出，新式茶

饮市场规模预计到 2020 年底将达到 1020 亿元。经历过 2020 年疫情洗牌的新式茶饮行业，在 2021 年迎来了新一轮的增长，顺应了国内消费需求的增长，虽然行业市场集中度有所提高，但是仍然处于良性、开发的竞争态势中。因此，我国奶茶行业发展空间前景可期。



有研究机构调研结果显示，因为“喜欢喝，没事来一杯”而购买现制奶茶的人数最多，占总人数的 54%，约有 14%的消费者购买现制奶茶是为了“和同事拼单”，而 20%的消费者是为了“网红店打卡，或者是新口味尝鲜”，还有 8%的消费者喝奶茶是为了“提神醒脑”，同时还有一部分人，点一杯奶茶仅仅只是为了“晒”朋友圈。由此可见，现制奶茶已不仅仅只是一种饮品，同时还具备功能属性和社交属性。



数据来源：町芒研究院

有关调研结果显示，仅有 26%的消费者会选择“标准糖”，而 74%的消费者会选择“半糖/少糖/无糖”。同时低卡、低糖、轻脂等健康概念也逐步加入到产品设计之中。同时，根据我们团队的问卷分析结果，在问及“是否觉得奶茶健康”的问题上，超过 70%的人在 unhealthy 这一问题上达成共识。

由此可见，越来越多的消费者和开始意识到奶茶存在的高糖高热量问题，同时对于现制奶茶低糖需求也日益突出，现制奶茶亟需向日常化、健康化的方

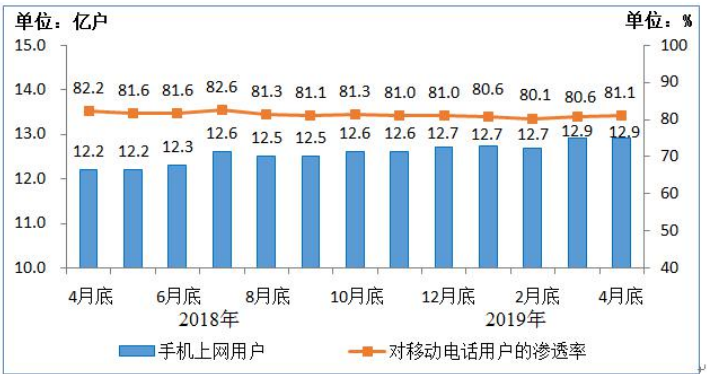
向发展。在调研过程中还发现，消费者在选购奶茶时，由于缺乏核心判断力，对奶茶的实际情况并不了解，在选择奶茶时，也只是盲目跟风打卡或者仅仅只是“喜欢某一款的口味”。

| 选项 | 小计 | 比例 |
|----------|----|-------------------------------|
| 是 | 34 | <div><div></div></div> 35.05% |
| 否 | 70 | <div><div></div></div> 72.16% |
| 本题有效填写人次 | 97 | |

表格来源：自研问卷

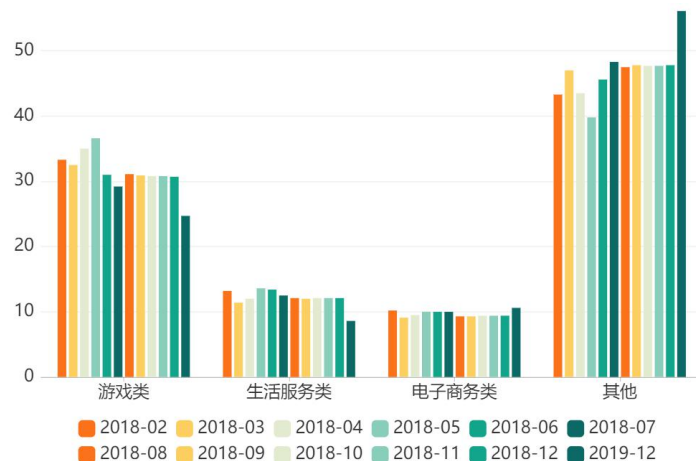
（二）同类产品（生活服务类安卓应用）分析

截止 2020 年 5 月，我国三家基础电信业务企业智能手机上网用户数达 13.17 亿户，对移动电话用户的渗透率为 82.7%，较上年末提升 0.5 个百分点。相比较于 2018、2019 年同期，可以发现我国手机用户规模始终保持稳定增长，人手一部智能手机在中国即将成为常态，这也意味着智能手机在中国完成了普及。



图片来源：工信部

同时，随着智能手机的普及，中国互联网用户数量规模也在持续增长，互联网生态的人口基数已然达到，信息时代猛然降落，也迅速改变了人们获取信息的手段。从寄信到通话，从通话到能获得大量信息的智能手机，人们获取信息的方式已经发生了质的变化。人们也不仅仅满足于手机的通话和短信功能，而是把手机当做接受讯息，了解时事的工具，加入社交网络的工具，便利生活的工具，娱乐游戏的工具。现如今，人们对各大信息平台的提供的信息具有很强的接受度和依赖度，作为以便利生活，丰富社交为出发点的各类生活服务类手机应用应运而生。



图片来源：艾媒数据中心

自互联网诞生的时候，就有了一系列生活服务类网站例如阿里巴巴、58同城、赶集网等，随着智能手机的诞生和移动应用的产生，生活服务类应用是最贴近人们生活、最能给人们提供便利的一类APP，用户一旦感受到便捷性之后便会迅速产生黏性，发展到如今，支付宝、淘宝、大众点评等移动应用就是最好的例子。

移动互联网市场的跟新迭代，各种类型的应用软件交替出现，唯有生活服务APP的留存时间是最久的，虽然生活服务应用软件不是最常用的APP软件，但就是它的实用便捷性让现代的人越来越离不开它。这也就是为什么现在越来越多的互联网大头都来抢占这一市场的原因。

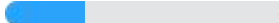

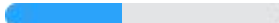


（三）主要竞争对手

根据我们团队的调研结果来看，我们主要的竞争对手是品牌自有APP和品牌自有小程序。

（1）对整个手机APP应用市场最大的垂直领域竞争对手是小程序，据数据统计，目前小程序的日活用户已经突破4.4亿，覆盖超过200个细分行业；而在去年，小程序的交易额就已突破8000亿元，2020年有望破两万亿。

小程序能迅速获得市场的重要原因是受疫情的冲击，线下实体业务收到重创，为此也加速了实体业线上转型的进程，而小程序成为布局线上的重要工具。小程序能迅速下沉市场，接触更多用户，降低成本的同时完成用户沉淀和提高单体经济价值。

同时有赖于微信生态的构建日益完善，小程序能够适配多场景，比如小程序+直播、小程序+购物、小程序+搜索引擎等可以触及到不同的用户，商家要做的就是将用户圈在一起，通过服务、支付、营销等能力，把流量变成「留量」，可以持续转化会员，沉淀用户和数字化资产。可以说小程序经济圈已经形成，以微信小程序为核心纽带，连结微信支付、企业微信、微信搜一搜、微信AI等微信生态能力，组成了一个全景生态矩阵，正在成为互联网经济中的新星。

| 选项 | 小计 | 比例 |
|----------|----|---|
| 品牌自有 APP | 28 |  28.87% |
| 品牌自有小程序 | 65 |  67.01% |
| 第三方 APP | 40 |  41.24% |
| 线下门店 | 61 |  62.89% |
| 其他途径 | 2 |  2.06% |
| 本题有效填写人次 | 97 | |

表格来源：自研问卷

(2) 品牌自有 APP 和第三方服务类 APP 是横向领域的最大竞争对手。第三方 APP 包括大众点评、口碑等生活信息服务平台，竞争的范围主要集中在推荐内容的比较上，对于品牌自有 APP，竞争区域小而直接——用户在信息获取渠道上的黏性。

(3) 细分赛道上，目前市场上没有同类产品出现，但是有类似聚合信息的细分领域的 APP 出现，并形成了一定规模。就运营经验和产品路线上，我们有后发优势，同时可以吸收经验，精益求精。

（四）自身条件分析

我方团队 APP 开发人员

- 1 熟悉 Android Studio
- 2 熟悉各种协议如 HTML/WML/HTTP
- 3 学习并熟悉三大开发语言 Java、C++和 python
- 4 学习并熟悉数据结构与各大基础算法
- 5 熟悉数据库创建和使用等初步技巧

· 产品定位及目标

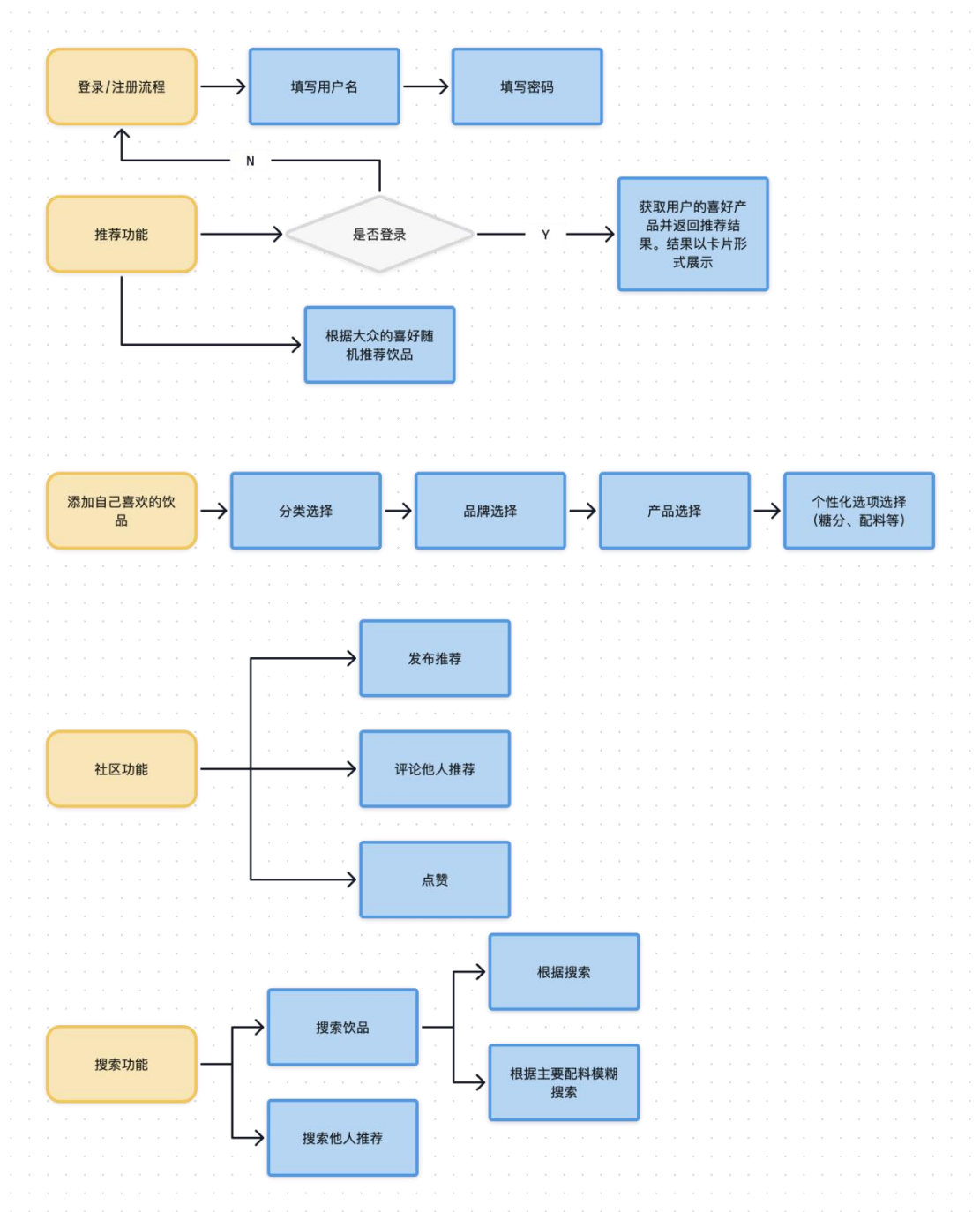
本项目主要设计推出一款主打个性化奶茶推荐及健康提醒的“奶茶控”app。项目利用数据库存储数据，实现对奶茶的糖度、卡路里、口味、适合对象等信息进行储存与分类；利用 AI 识别功能，实现对奶茶标签的识别与检索，获得产品信息；利用算法推荐等技术，实现消费群体的口味及消费水平分类，给予用户精准的推荐，同时记录和分析用户饮用奶茶的情况，进行健康提醒；利用智能排序算法，更新奶茶评分及排序；利用关键词匹配等技术，实现搜索内容精准匹配。以上技术来满足用户的选个性化推荐及健康提醒需求。最终通过公众号推广、发布平台广告、校园宣传、市场宣传等推广途径进行 app 的宣传，通过品牌形象建设、广告招租等营销方式实现盈利。

本项目的用户群体定位是喜欢喝奶茶，而且还注重奶茶的口味，糖度，卡路

里等信息的群体，主要是年轻的上班族和学生群体。由于学习工作的压力，该群体中的人较多喝奶茶。在喝奶茶时，也更注重奶茶的口味和卡路里。软件的定位是为用户个性化推荐奶茶以及喝奶茶的健康提醒，可以为该群体推荐奶茶以及通过告知奶茶的糖度，卡路里等信息来进行健康提醒。

· 产品内容总策划

(一) 应用流程规划



（二）设计与测试规范

1.设计规范

- UI 设计简约，且符合大众审美
- 进行模块化设计，将一个个功能分为模块设计
- 用户操作要求方便、简单

2.测试规范

- 对每个模块进行单元测试
- 需要遵守编码规范，以减少代码的维护成本，提高开发工作效率
- 测试时必须进行多种特殊情况的考虑

（三）开发日程表

| 时间 | 进度安排 |
|---------------|-----------------------|
| 第 6 周到第 9 周 | 完成软件的 UI 设计 |
| 第 10 周到第 15 周 | 完成软件的主体功能 |
| 第 16 周 | 完成软件测试，记录用户体验和建议，改进软件 |
| 第 17 周 | 汇报项目 |

· 技术解决方案

（一）奶茶控 app 的推荐算法

个性化推荐：

1. 基于用户推荐：

- 获取用户注册时选择的对奶茶的偏好信息
- 根据用户的注册信息对用户分类
- 给用户推荐他们所属分类中用户喜欢的奶茶

2. 基于内容推荐：

- 根据用户在社区的浏览记录，给用户推荐没接触过的社区文章
- 根据用户搜索信息，给用户推荐没接触过的奶茶品种

3. 基于关联规则推荐：

- 通过后台数据库统计用户搜索/点赞/收藏过的奶茶 A 和奶茶 B，若大比例同时浏览并对奶茶 C 有兴趣，向用户推荐比例大的奶茶 C

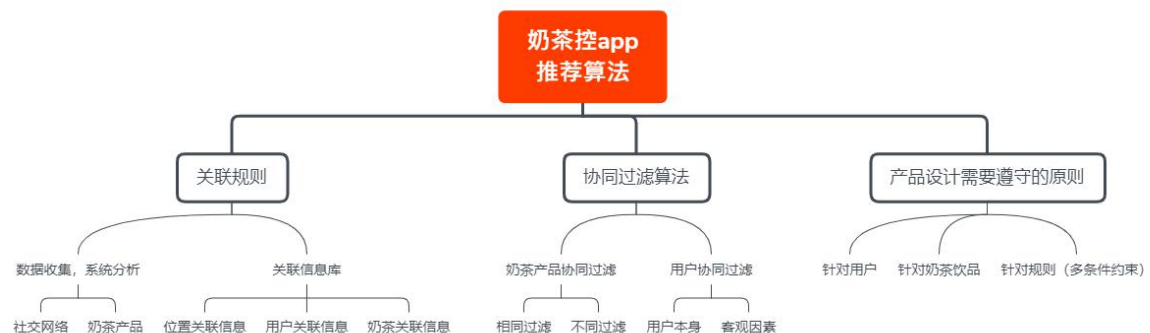
4. 基于协同过滤算法推荐：

- 相同过滤：给用户推荐和他们对奶茶偏好相似的其他用户浏览点赞喜爱的奶茶，预先将用户标签和奶茶产品标签进行关联，在对等的情况下，推荐条件相同的奶茶产品，同时基于完全对等的情况下，推荐一下稍微不同的奶茶产品给用户做额外的考虑。用户在看“喜茶”的“多肉葡萄”的时候，说不定看到季节限定品“多肉杨梅”就心动了。
- 不同过滤：差异化推荐，在突破用户选择范围外，给用户推荐另外一款口味佳的饮品。用户既然是在搜索，那么会转移注意力，“忽略”自己当前的条件去选择一下。

5. 对奶茶控 app 针对奶茶本身，用户本身和规则本身设计制定一定原则。

用途：

- ①向用户个性化推荐奶茶
- ②热门饮品推荐
- ③品牌新品推荐
- ④社区文章浏览推荐



奶茶控 app 的推荐算法

· 推广方案

1. 明确目标群体：主要面向爱喝奶茶的年轻人

2. 确定推广阶段和对应目标：

推广大概分为三个阶段：

无人知——>吸引用户——>把用户留住（不被同类型 app 替代）——>让用户为 app 掏钱（现阶段无要求）

三大核心目标：扩大用户群、寻找合适的盈利模式以增加收入、提高用户活跃度。

3. 推广渠道：线上线下结合

4. 收集数据指标：用户留存率，活跃用户，付费率（现阶段无要求），问卷调查

5. 对收集的数据研究用户心理，用户行为，用户需求，进行对应 app 更新调整。

· 运营规划书

（一）运营策略

据调查，手机用户对手机应用付费意愿方面，57.8%的用户完全不会下载付费 app，38.1%的用户偶尔会下载。若要为下载 app 花钱，用户更愿意为阅读学习、游戏娱乐和生活服务类的这三类 app 付费，而能接受付费 app 的价位区间在 5 元以下的用户居多，占 44.8%。

综合调查结果，关于个性化奶茶推荐 APP 的总体运营策略是：主要采取免费增值模式和广告招租实现盈利。

1. 前中期 APP 免费对外开放，实行为期两个月的试运营，同时进行 APP 推广以及内容营销，这两个月根据用户使用情况和反馈调整内容和模式。同时在微信和支付宝筹备建立小程序。

2. 中期加大力度推广和品牌形象建设，并逐步加入广告位实现收入增加，加强培养用户使用习惯，旨在拥有第一批高黏度的核心用户。在 APP 核心竞争力得到市场认可之后，在小程序上制作商开始发力。

3. 后期加大预算，将设立完整品牌形象和完善的 APP 运营模式和功能，正式投放市场，完成广告位的设置和引流链接通道，以及开始对服务内容实施低价会员制，实现盈利。

社群运营：通过微信公众号、微博、社交平台和问答平台，创新文案创意搭配产品设计，结合奶茶店的产品内容、服务体验以及各种与奶茶相关的讯息，针对性地进行社群运营。

积极开展特色活动策划，不定期举办线上线下相关活动，增加用户粘度。

（二）生产材料

- 1 原始数据取自各大便捷生活应用网站
- 2 APP 开发参考资料《Android Studio 开发实战从零基础到 APP 上线》、《第一行代码》、《Android 开发艺术探索》、《30 天 App 开发从 0 到 1》等
- 3 若干文献、行业报告、调查问卷等理论材料

（三）设备需求

- (1) 硬件：五台电脑
- (2) 软件：云测试服务平台、Android studio

（四）成本控制

APP 孵化总成本控制在 1800 元左右。

二、产品实现方案

（一）系统的主要功能

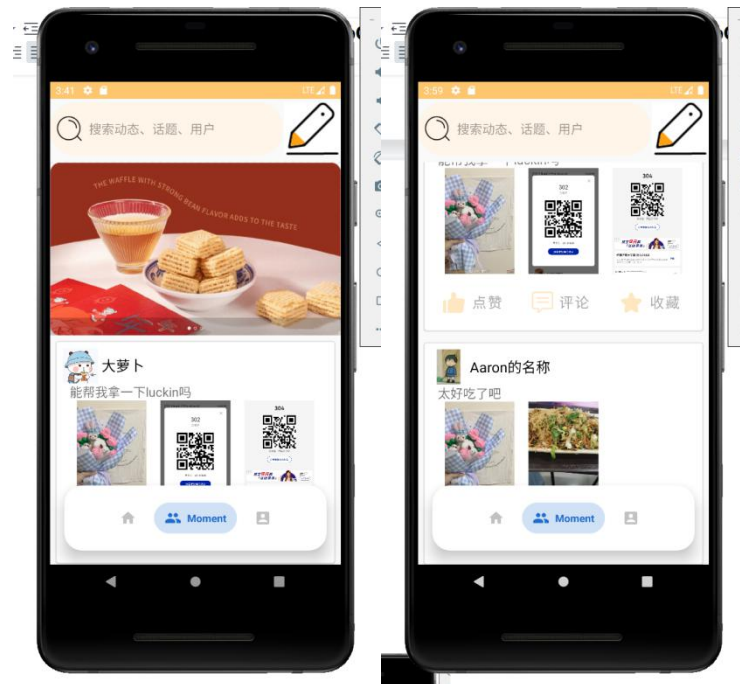
1. 首页奶茶展示
2. 社区动态展示
3. 上传动态功能
4. 搜索功能
5. 客服联系功能
6. 点赞和点赞展示功能
7. 收藏和收藏展示功能
8. 我的口味添加功能
9. 登录/注册功能

（二）UI 界面设计

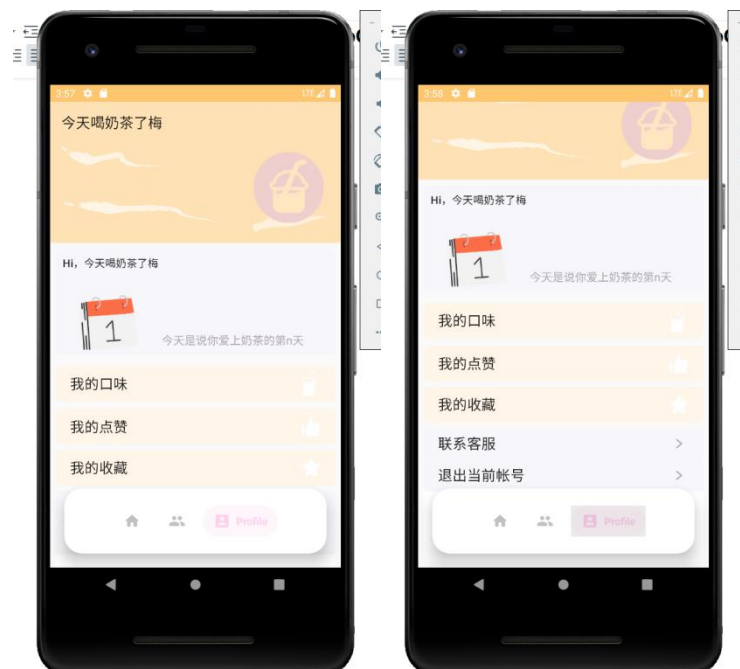
1. 首页页面



2. 社区页面



3. 我的页面



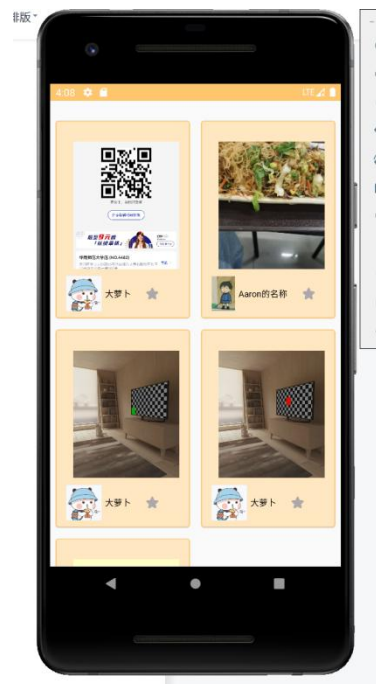
4. 饮品详情/评价



5. 我的点赞



6. 我的收藏



7. 我的口味



8. 客服联系



9. 登录



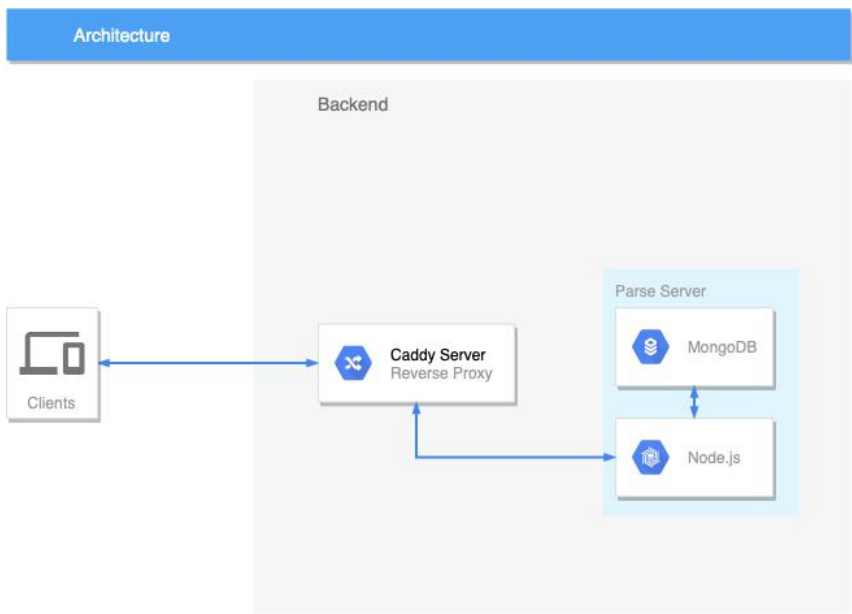
10.注册



（三）关键技术和技术难点

后端部署

后端架构：



Parse Server

Parse Server 是一个基于 node.js 的 Backend-as-a-service(BaaS)框架。最初开发 Parse 的是被 Facebook 收购了，然后倒闭了，后面开源并有社区维护直到现在，也有将近 8 年历史了。

我们用它实现：

- ① 对用户授权，鉴权
- ② 获取、上传静态的资源文件（图片等）
- ③ 对数据的增删查改操作。

为了方便统一控制 Parse Server 以及其依赖服务，我选用 Docker 作为部署工具，Docker 应该也是生产环境中常用的微服务部署形式。

Docker-compose 则是一个工具能够控制多个 docker container，可以在一个文件中定义自己的 application stack.

在当前场景下的部署要点是：

- ① 给 Parse Server 容器设置适当的环境变量，指定 masterkey,appid,serverURL 和 DatabaseURL。databaseURL 的设置通过查找 docker container 之间的通信方式的相关资料实现。
- ② MongoDB 的容器设置则是设置 DB 的鉴权方式以及 credential 的内容。
- ③ MongoExpress 则是管理 MongoDB 的 GUI 界面，方便后续调试

Docker-Compose file

```
1.  version: '3.1'
2.  services:
3.    mongo:
4.      image: mongo
5.      restart: always
6.      environment:
7.        MONGO_INITDB_ROOT_USERNAME: root
8.        MONGO_INITDB_ROOT_PASSWORD: example
9.    networks:
10.     default:
11.       ipv4_address: 172.33.33.5
12.     ports:
13.       - "27017:27017"
14.     volumes:
15.       - db_data:/data/db
16.       - configdb_data:/data/configdb
17.
18.    mongo-express:
19.      image: mongo-express
20.      restart: always
21.      networks:
```

```

22.     default:
23.     ipv4_address: 172.33.33.6
24.     ports:
25.     - "8081:8081"
26.     environment:
27.     ME_CONFIG_MONGODB_ADMINUSERNAME: root
28.     ME_CONFIG_MONGODB_ADMINPASSWORD: example
29.     ME_CONFIG_MONGODB_URL: mongodb://root:example@mongo:27017/
30.
31.     parse:
32.     image: parseplatform/parse-server
33.     ports:
34.     - "1337:1337"
35.     links:
36.     - mongo
37.     networks:
38.     default:
39.     ipv4_address: 172.33.33.7
40.     environment:
41.     - PARSE_SERVER_APPLICATION_ID=the_app
42.     - PARSE_SERVER_MASTER_KEY=xexhwx8thmvhs8o7xw8
43.     - PARSE_SERVER_DATABASE_URI='mongodb://root:example@mongo:27017/app?authSource=admin&ssl=f
      else'
44.     - PARSE_PUBLIC_SERVER_URL=https://app.moe.yt:233/parse
45.     - PARSE_SERVER_FILE_UPLOAD_ENABLE_FOR_ANONYMOUS_USER=true
46.
47.     networks:
48.     default:
49.     external: true
50.     name: parse-network
51.
52.     volumes:
53.     db_data:
54.     configdb_data:

```

Caddy

Caddy 网络服务器是一个可扩展的、跨平台的、用 Go 语言编写的开源 Web Server。它默认使用 HTTPS 具有更高的安全性和隐私性，我们主要使用它作为和后端服务通信的反向代理(Reverse Proxy)

caddy 安装使用很方便：

- ① 在官网下载系统架构对应的 binary file

- ② 给予可执行权限
- ③ 编写 Caddyfile
- ④ 启动 caddy 服务即可

Caddyfile

```
1.  {
2.      acme_ca https://acme.zerossl.com/v2/DV90
3.      https_port 233
4.      http_port 234
5.  }
6.
7.  app.moe.yt {
8.      reverse_proxy 172.33.33.7:1337
9.  }
```

Android App 主要 Feature

联系客服功能

联系客服的界面参照的是 Jetpack Compose 的官方 Example。
<https://github.com/android/compose-samples/tree/main/Jetchat>

我就重点介绍主要的技术难点：

1. 异步获取服务器中该用户先前发送的信息，并且反馈到 UI 上
2. 发送信息功能。

异步获取信息

流程：

1. 创建 UI 后开启子线程
2. 子线程中向服务器发送请求，获取用户先前发送的信息，把信息添加到 UI 中对应的数组中。

关键点在于这个数组的内容如何动态显示到 UI 上。

查询资料后发现 kotlin 中也可以实现类似 Vue 的状态管理，而且跟 vue 一样简单，就创建一个特别的数据类型 `SnapshotStateList` 就可以了。

```
1.  fun anotherFetch(messages: SnapshotStateList<Message>) {
2.      val messageFetched = fetchMessages()
3.      messages.swapList(messageFetched)
4.  }
```

```

5.
6.
7.     fun <T> SnapshotStateList<T>.swapList(newList: List<T>) {
8.         clear()
9.         addAll(newList)
10.    }

```

发送信息功能

流程：

获取用户的输入，把输入（要发送的信息）传给后端服务器，并且立刻在本地存放信息的数组中添加这条信息。

在刚才和后端通信的 Callback 函数中根据后端服务器返回的状态值对本地存放信息的数组进行二次处理。比如发送失败或者超时了就在 UI 中显示一个 Toast，并且删除数组中的信息。

登录注册

使用 Parse 的 SDK 特别方便，只要自己写好 Callback 函数就可以了。

如下：

```

1.     ParseUser.logInInBackground(username, password, new LogInCallback() {
2.         public void done(ParseUser user, ParseException e) {
3.             if (user != null) {
4.                 Toast.makeText(LoginActivity.this, "Logged in", Toast.LENGTH_SHORT).show();
5.             } else {
6.                 // Login failed. Look at the ParseException to see what happened.
7.                 Toast.makeText(LoginActivity.this, "Failed to log in", Toast.LENGTH_SHORT).show();
8.             }
9.             lpi.hide();
10.            finish();
11.        }
12.    });
13.
14.    user.signUpInBackground(new SignUpCallback() {
15.        public void done(ParseException e) {
16.            if (e == null) {
17.                Toast.makeText(v.getContext(), "Signed up", Toast.LENGTH_SHORT).show();
18.                finish();
19.            } else {
20.                Toast.makeText(v.getContext(), "Sign up failed.", Toast.LENGTH_SHORT).show();

```

```
21.         }
22.     }
23. });
```

上传图片功能

上传图片分为两步：

1. 用户选择图片，并且获取对应的 URI
2. 根据 URI 读取图片，上传到服务器中

用户选择图片

我们使用 `android-image-picker` 这个库实现选择图片功能。使用时，我们只需要实例化 `ImagePickerConfig` 对 `ImagePicker` 进行个性化的设置，就可以通过这个 `config` 启动选择图片的 `activity` 了

步骤：

- ① 开一个 `activity` 选择图片
- ② 在 `mainactivity` 中从 `intent` 获取刚才用户选取的图片 URI

Key Code

```
1.     imagePickerButton.setOnClickListener(new View.OnClickListener() {
2.         @Override
3.         public void onClick(View view){
4.
5.             if(hasImage[0]) { // 清除已经选择的照片
6.                 imagePicked.clear();
7.             } else { // 选择新照片
8.                 ImagePickerConfig conf = new ImagePickerConfig();
9.                 conf.setMode(ImagePickerMode.MULTIPLE);
10.                conf.setLimit(3);
11.                Intent intent = createImagePickerIntent(view.getContext(), conf);
12.                startActivityForResult(intent, REQ_IMG);
13.            }
14.            hasImage[0] = !hasImage[0];
15.        }
16.    });
17.
18.    @Override
19.    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data){
20.        super.onActivityResult(requestCode, resultCode, data);
```



```

21.         if (requestCode == REQ_IMG && resultCode == RESULT_OK) {
22.             List<Image> l = ImagePicker.INSTANCE.getImages(data);
23.             if(l != null) {
24.                 imagePicked.addAll(l);
25.                 Log.e("image", String.valueOf(l.size()));
26.                 for(int i=0;i< l.size();i++)
27.                 {
28.                     Picasso.get().load(l.get(i).getUri()).resize(50,50).centerCrop().into(imgs[i]);
29.                 }
30.             }
31.
32.         }
33.     }

```

根据 URI 读取图片并上传

分成三步：

- ① 从上一步的 URI 中读取图片,通过 contentResolver 转换成 bytes array
- ② 将 bytes array 上传到服务器中（服务器会返回该文件对应的 URL）
- ③ 将 URL 放到自定的数据表中

Key Code

```

1.     public byte[] getBytes(InputStream inputStream) throws IOException {
2.         ByteArrayOutputStream byteBuffer = new ByteArrayOutputStream();
3.         int bufferSize = 1024;
4.         byte[] buffer = new byte[bufferSize];
5.         int len = 0;
6.         while ((len = inputStream.read(buffer)) != -1) {
7.             byteBuffer.write(buffer, 0, len);
8.         }
9.         return byteBuffer.toByteArray();
10.    }
11.
12.
13.    for(Image i:imagePicked) {
14.        try {
15.            InputStream iStream = getContentResolver().openInputStream(i.getUri());
16.            byte[] inputData = getBytes(iStream);
17.            ParseFile file = new ParseFile("pic.jpg", inputData);
18.            file.save();
19.            attachments.add(file);

```

```

20.         } catch (IOException | ParseException e) {
21.             e.printStackTrace();
22.         }
23.     }

```

It is called the scientific method and it's what created computer that you are typing your half-witted conspiracy theories on.

发布动态

在 Parse 中，每一个数据表都对应一个 Object，所以：

- 新建对象：new ParseObject("Moment");
- 添加属性：moment.put("title","this is title");
- 上传到服务器：

```

1.     moment.saveInBackground(new SaveCallback() {
2.         @Override
3.         public void done(ParseException e) {
4.             if(e == null) {
5.                 Toast.makeText(view.getContext(), "Published.", Toast.LENGTH_SHORT).show();
6.             } else {
7.                 Log.e("Error When Saving", e.toString());
8.             }
9.         }
10.    });

```

saveInBackground 是新建一个 Task 对象，它保证动作是异步进行的，

点赞，收藏加口味

这三者的实现思路都大致相同。

数据表结构如下：

| likes Array | collections Array | flavor Array | U |
|-------------|-------------------|--|---|
| FoR0h8SIHW | N4OaW1FaYt | ["三分糖", "少冰", "中文", "波霸", "少糖", "yoo"] | 3 |

三个字段都是维护 Array，点赞和收藏是有对应的 Moment 对象的，所以存放的是指针。口味的话就直接存 String 了

1. 获取 User Object 的字段值
2. 更新 User Object(上传到服务器)
3. 调用 Adapter 的`notifyDataSetChanged` 方法

```

1.     ArrayList<String> flavors = null;
2.     try {

```

```

3.         List<String> l = currentUser.fetch().getList("flavor");
4.         if (l == null ) {
5.             flavors = new ArrayList<>();
6.         } else {
7.             flavors = new ArrayList<>(l);
8.         }
9.
10.    } catch (ParseException e) {
11.        e.printStackTrace();
12.    }
13.    flavors.add(flavor);
14.    currentUser.put("flavor", flavors);
15.    currentUser.saveInBackground(new SaveCallback() {
16.        @Override
17.        public void done(ParseException e) {
18.            saveButton.setEnabled(true);
19.            if(e == null) {
20.                Toast.makeText(LikeActivity.this, "保存成功", Toast.LENGTH_SHORT).show();
21.                flavorInput.setText("");
22.                mMyAdapter.notifyDataSetChanged();
23.            } else {
24.                Toast.makeText(LikeActivity.this, "保存失败", Toast.LENGTH_SHORT).show();
25.            }
26.        }
27.    });

```

（四）用户体验记录和分析

我们一共采访了十位用户关于奶茶控软件的经验。

用户一反映在奶茶控软件中，颜色配比感官较为舒适，但 APP 中存在的矩形框和图片占比较大。分析其原因是奶茶控 APP 是一个推荐介绍奶茶的 APP，为了使用户能更生动形象了解到奶茶的相关信息，我们必不可少的上传了一系列奶茶的图片，以便于用户可以全方位地了解奶茶。

用户二反映 APP 中没有反馈渠道，不便于用户反馈软件缺陷。分析其原因是刚开始设计界面时我们只考虑到 APP 推荐搜索奶茶的主要功能而没有去考虑软件的用户友好性。基于这一条反馈，在后续改进软件的过程中，我们增加了联系客服的界面，方便用户反馈建议。

用户三反映软件在该用户手机中的界面布局不佳，需要继续改进。分析其原因是在前期搭建软件时，我们采用的是绝对布局而非相对布局，导致不能适配不同用户的手机版本。

用户四表示界面有待改进，首页的图片分辨率较低，需要更换高清图片。分析其原因是在搭建数据库时，我们并没有上传高清图片，需要重新上传高清图片。

用户五表示界面中一些布局还没有优化，比如说没有固定用户头像的显示大小，对于不同的用户，显示得很奇怪。分析其原因是我们并没有限制用户上传的头像的尺寸，在后续改进中，我们需要提醒用户上传相应规格的图片，或是在软件中有切割图片的效果。

用户六表示社区页面的轮播图是播放完三张之后经过第二张图片再回到第一张图片的，需要改进使图片的衔接更流畅一点。分析其原因是轮播图是采用viewpager做的，设置图片的时候只是简单根据图片位置设置图片轮播，而没有设置循环播放时首尾图片改变的效果。

用户七表示界面布局还有待优化，比如说首页点击一点点的图片，但进去的页面却是奶茶。分析其原因是我们未设置详情页面根据用户点击的页面改变，也就是说，后面需要在数据库添加更多奶茶的介绍，并且使详情界面会根据用户点击改变。

用户八表示在社区界面评论信息并不能看到相应的评论信息，以及评论界面还没有搭建好。分析其原因是我们确实还未美化评论界面，以及其相应的功能还未实现。

用户九表示联系客服界面我发出去的信息在界面左边，与客服在一边，这让我很不习惯。分析其原因是我们搭建客服界面较为粗糙，未将用户发送的信息框置于界面右边。

用户十表示个人中心界面的用户名和头像没有根据我当前登录的账号修改。分析其原因是个人中心页面的用户名和头像未设置根据当前登录账户的名字和头像修改。

（五）已完成的改进和存在的问题

1. 已改进的优化点

缓存优化

在获取服务器资源时，为了防止用户等待比较长的时间，我们采用了一定的优化方法：

- ① 先获取本地的 Cache，将 Cache 中的数据先显示到 UI 中，
- ② 再从服务器获取最新的资源，获取完了再更新到 UI 中。
- ③ 同时，在每一次获取服务器中资源后，要将资源存储到本地的 Cache 中。

这一系列机制很容易用 Parse SDK 实现

Key code:

```
1. query.fromLocalDatastore().findInBackground().continueWithTask((task) -> {
2.     // Update UI with results from Local Datastore ...
3.     ParseException error = (ParseException) task.getError();
4.     if(error == null){
5.         List<ParseObject> momentList = task.getResult();
6.
7.         updateMoment(momentList);
8.     }
9.     // Now query the network:
10.    return query.fromNetwork().findInBackground();
11. }, Task.UI_THREAD_EXECUTOR).continueWithTask((task) -> {
12.    // Update UI with results from Network ...
13.    ParseException error = (ParseException) task.getError();
14.    if(error == null){
15.        List<ParseObject> momentList = task.getResult();
16.        updateMoment(momentList);
17.        ParseObject.unpinAllInBackground("MomentList", momentList, new DeleteCallback() {
18.            public void done(ParseException e) {
19.                if (e != null) {
20.                    return;
21.                }
22.                ParseObject.pinAllInBackground("MomentList", momentList);
23.            }
24.        });
25.    }
26.    return task;
27. }, Task.UI_THREAD_EXECUTOR);
```

图片预览

为了改进用户体验，增加了图片预览功能。使用的库：[StfalconImageViewer](#)

Key Code:

```

1.  ArrayList<String> images = new ArrayList<>();
2.  for(ParseFile f:model.attachments) {
3.      images.add(f.getUrl());
4.  }
5.  View.OnClickListener img_listener = new View.OnClickListener() {
6.      @Override
7.      public void onClick(View view) {
8.          new StfalconImageViewer.Builder<String>(context, images, new ImageLoader<String>() {
9.              @Override
10.             public void loadImage(ImageView imageView, String image) {
11.                 Picasso.get().load(image).into(imageView);
12.             }
13.         }).show();
14.     }
15. };

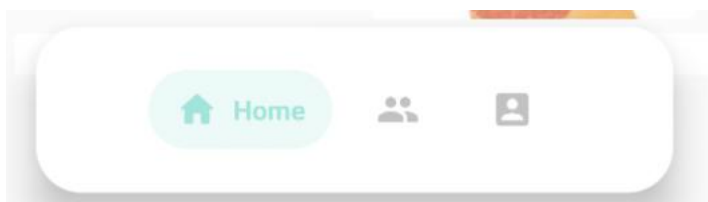
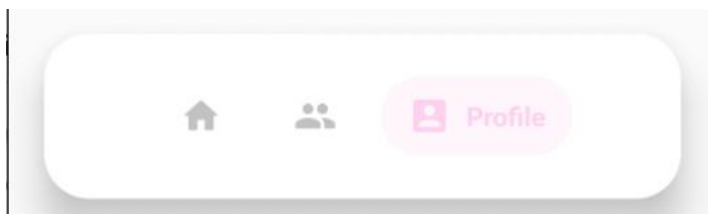
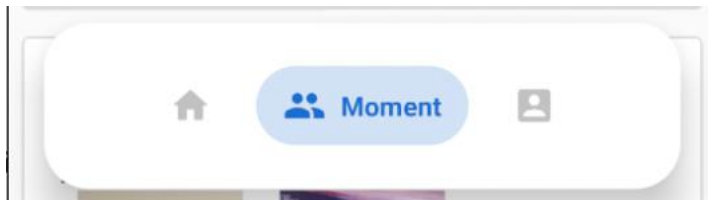
```

充满动效的底部栏：

使用的库：[ExpandableBottomBar](#)

配合 CoordinatorLayout 可以实现用户滑动时，底部栏收缩，用户可以得到更大的可视空间。

我们甄选的底部栏配色与 App 完美契合，来自：<https://colorhunt.co/>



2. 存在的问题

1. 饮品详情页缺失。因为奶茶产品的数据难以批量获取，所以暂时未能实现多种奶茶产品的详情页展示。后期可能通过人工输入产品详情数据弥补。
2. 搜索功能实现，但未能实现推荐搜索。因为我们 APP 仍处于起步阶段，真实用户量不足以支撑基于机器学习的推荐算法。后期用户量多了再考虑。
3. 用户体验的细节不够完善。如：图片未加载时 UI 显示图片区域是直接缺失的，可以考虑增加 loading 动画或显示占位图片。其他涉及到从服务器获取数据的部分都需要做这种 placeholder 替换的相关操作

三、测试大纲和测试报告

1.测试大纲

1.1 测试目的

通过多种测试手段来验证奶茶控 App 是否已经达到设计指标。

1.2 测试环境

不同用户的手机和 WeTest 平台

1.3 测试方法

采用阿尔法测试即开发人员在测试现场由用户模拟实际操作进行测试；采用 WeTest 测试不同设备的兼容性和性能。

1.4 测试范围

主要为功能测试和兼容性测试

1.4.1 功能测试

| No. | 模块 | 权重 |
|-----|------|----|
| 1 | 登录注册 | A |
| 2 | 个人中心 | A |
| 3 | 社区界面 | A |
| 4 | 我的点赞 | B |
| 5 | 我的收藏 | B |
| 6 | 联系客服 | B |
| 7 | 我的口味 | B |
| 8 | 发布动态 | A |
| 9 | 搜索奶茶 | A |

1.4.2 兼容性测试

采用 WeTest 平台使用其中的 Top50 台真机测试奶茶控 App 的兼容性。

1.5 测试用例

| 序号 | 主要模块 | 功能点 | 预期结果 | 测试结果 |
|----|------|---------------------|---------------------|------|
| 1 | 登录注册 | 注册新账号 | 注册成功 | 通过 |
| 2 | | 使用新账号登录 | 登录成功 | 通过 |
| 3 | | 使用默认账号登录 | 登录成功 | 通过 |
| 4 | 首页界面 | 展示不同奶茶图片 | 展示不同奶茶图片 | 通过 |
| 5 | | 点击奶茶进入详情界面 | 点击奶茶进入详情界面 | 通过 |
| 6 | 社区界面 | 在搜索栏输入luckin 显示瑞幸信息 | 在搜索栏输入luckin 显示瑞幸信息 | 通过 |
| 7 | | 展示图片轮播 | 展示图片轮播 | 通过 |
| 8 | | 展示多个用户的评论信息 | 展示多个用户的评论信息 | 通过 |
| 9 | | 点击评论跳转到评论界面 | 点击评论跳转到评论界面 | 通过 |

| | | | | |
|----|--------|---------------------|---------------------|----|
| 10 | | 点击图片放大图片 | 点击图片放大图片 | 通过 |
| 11 | 个人中心 | 点击我的口味进入我的口味界面 | 点击我的口味进入我的口味界面 | 通过 |
| 12 | | 点击我的收藏进入我的收藏界面 | 点击我的收藏进入我的收藏界面 | 通过 |
| 13 | | 点击我的点赞进入我的点赞界面 | 点击我的点赞进入我的点赞界面 | 通过 |
| 14 | | 点击联系客服进入联系客服界面 | 点击联系客服进入联系客服界面 | 通过 |
| 15 | | 长按头像退出登录 | 长按头像退出登录 | 通过 |
| 16 | | 点击退出登录出现消息框提示是否退出登录 | 点击退出登录出现消息框提示是否退出登录 | 通过 |
| 17 | 联系客服 | 输入消息并发送 | 输入消息并发送 | 通过 |
| 18 | 评价详情界面 | 点击上方返回按钮返回首页 | 点击上方返回按钮返回首页 | 通过 |

2.测试报告

2.1 阿尔法测试

阿尔法测试主要是由软件开发人员进行，在自测中，我们发现联系客服界面发送消息后会闪退，首页第一个卡片今日奶茶点击也会闪退，以及评价的详情界面点击上方的返回按钮未能返回。

对于联系客服界面，本项目组检查了代码后发现是因为未检查当前登录用户，导致与数据库的连接出现错误。此外，就是客服对应的数据库的权限设置出错，未设置写入的权限，再修改了这两点后，联系客服界面就可以正常使用了。

对于首页今日奶茶卡片，闪退原因是错误设置了点击效果。

对于评价的详情界面，出现缺陷的原因是未设置点击按钮的返回效果。

2.2 WeTest 平台测试

在 WeTest 平台中，我们选用了 Top50 台真机对奶茶控 App 进行测试。

在本次测试的五十台手机中，有 49 台真机通过测试，1 台真机未通过测试。未通过测试的手机出现的问题是运行内存不足。



分析该手机未通过测试原因是内存不足，该手机内存仅有 3591MB，不足以支撑奶茶控 App 打开太多页面时页面间的跳转。

根据 WeTest 平台呈现的测试报告，奶茶控 App 在 50 台真机中的平均安装时间为 2.97 秒，平均启动耗时 1.50 秒，平均占用 1.21% 的 CPU，平均占用 222.60MB 的内存，平均消耗 3132.84KB 的流量，平均每秒的帧率是 6.54。

2.3 测试结论

奶茶控的兼容性较强，能在不同的安卓手机中安装和使用。但在后续开发中仍需改进，需要进行界面的优化以及具体功能的完善。此外，由于代码设计时，创建了 Activity 后未及时销毁，导致奶茶控 App 占用的内存过大。在后续改进中，也需要检查每个 Activity 是否有及时销毁，来确保以会产生内存不足或内存溢出的问题。

四、产品安装和使用说明

（一）产品安装

- ① 将 apk 文件保存到手机中
- ② 从手机文件管理处打开 apk 文件
- ③ 打开奶茶控 app 的 apk 文件详细界面中，点击“安装”按钮。
- ④ 安装完成后，点击“打开”按钮即可运行该应用奶茶控 app。

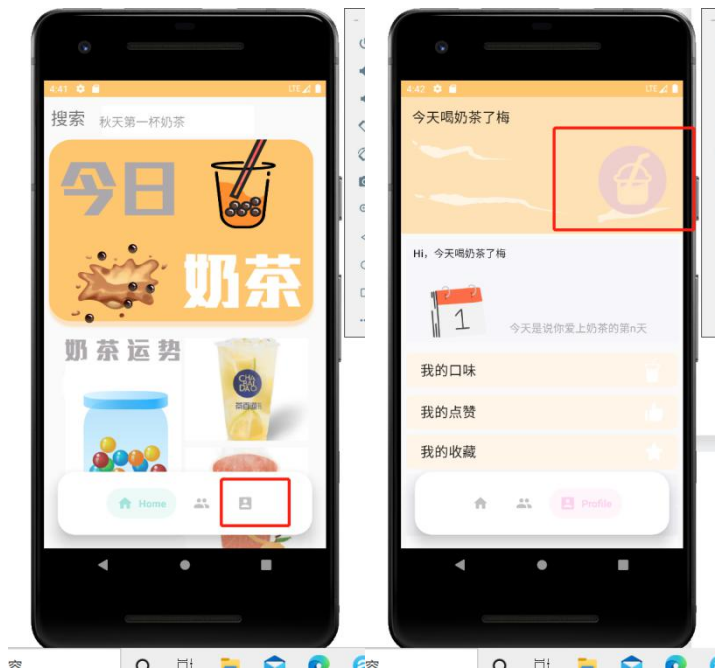
（二）使用说明

用户安装完毕后，点击桌面奶茶控 app 的图表即可进入 app 的首页，界面如下：



① 用户注册

用户第一次访问的时候，通过以下步骤进行用户注册：点击桌面图标——>导航栏最右手边图标——>点击头像——>到达登陆页面——>点击文本“还没有账号？点击注册”——>界面如图





② 账号登录

注册成功的用户可直接到达个人中心界面



也可通过登录页面，输入账号密码登录



③ 查看饮品详情页

打开首页——>点击推荐饮品——>到达饮品详情页



④ 查看饮品相关评价

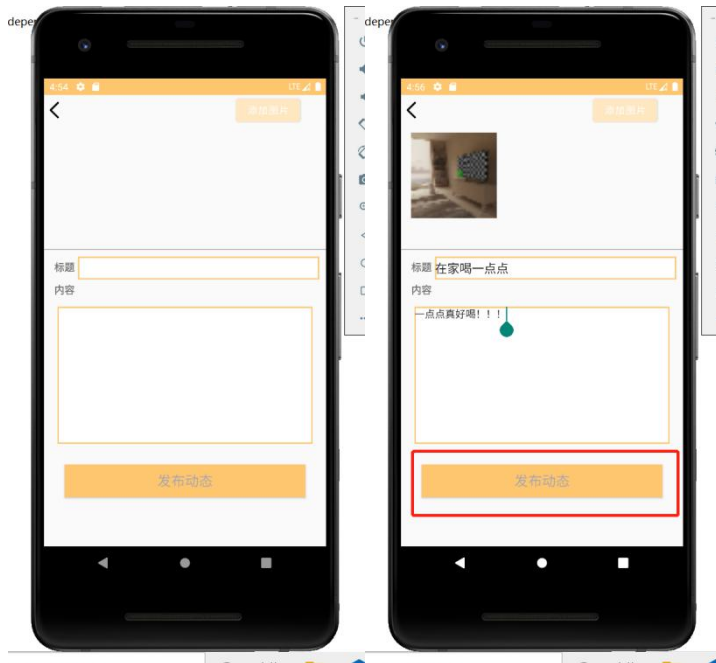
点击中间部分“评价”，即可到达



⑤ 上传动态

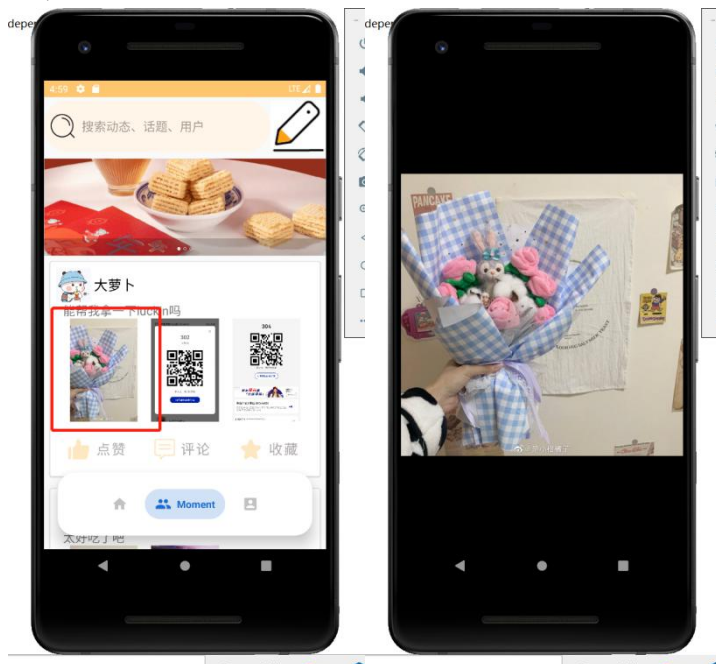
点击导航栏中间图标——>点击社区页面右上角“铅笔”图表——>进入上传动态界面，进行文案编辑以及图片上传——>点击发布





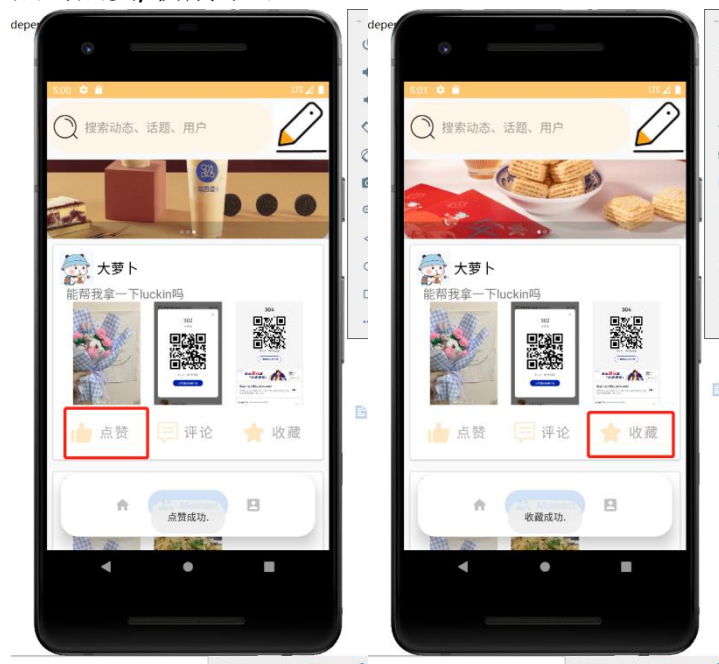
⑥ 图片预览

在社区页面点击动态图片



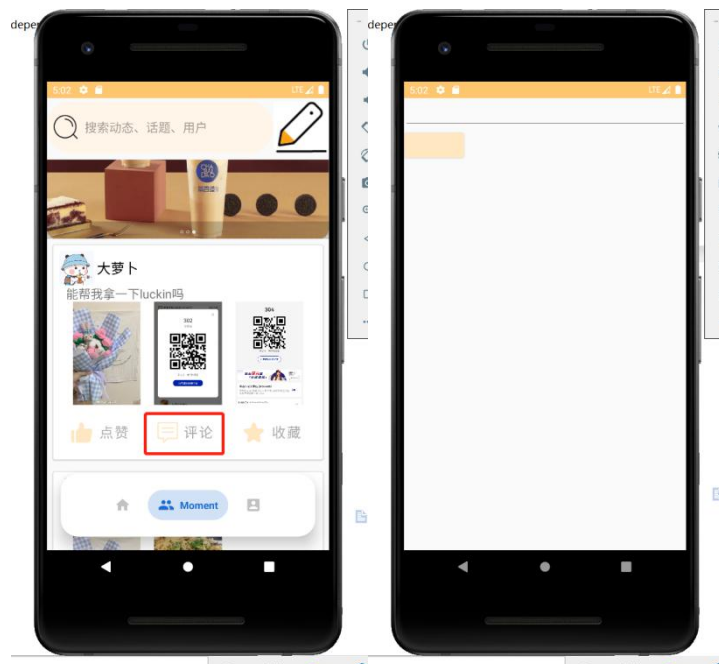
⑦ 点赞/收藏

点击点赞/收藏即可



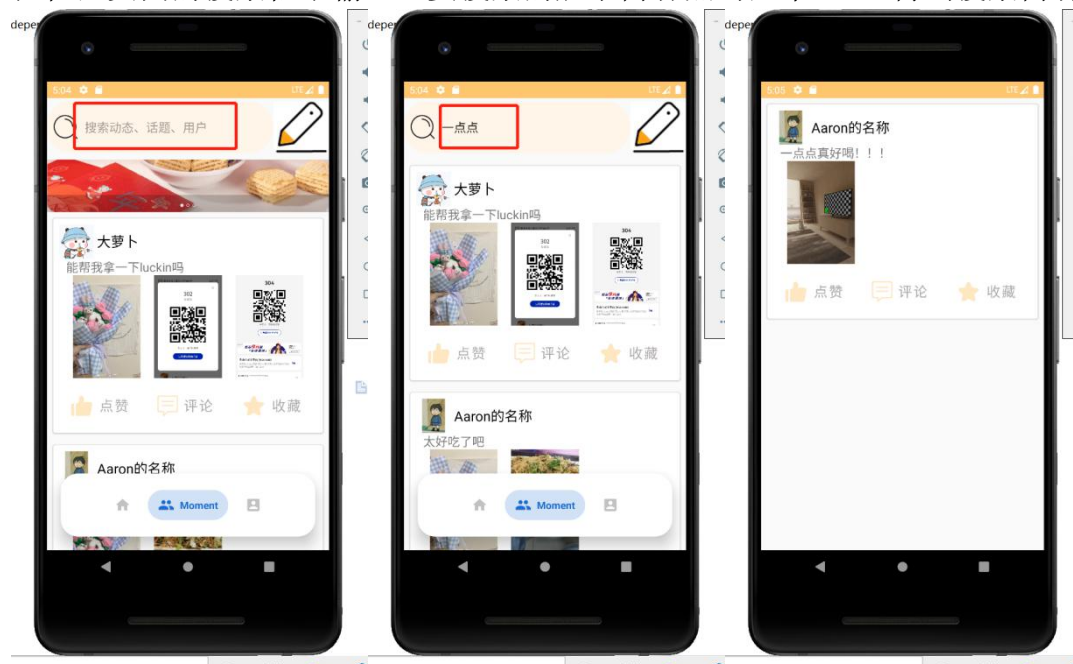
⑧ 我的评论

点击评论，输入即可



⑨ 搜索功能

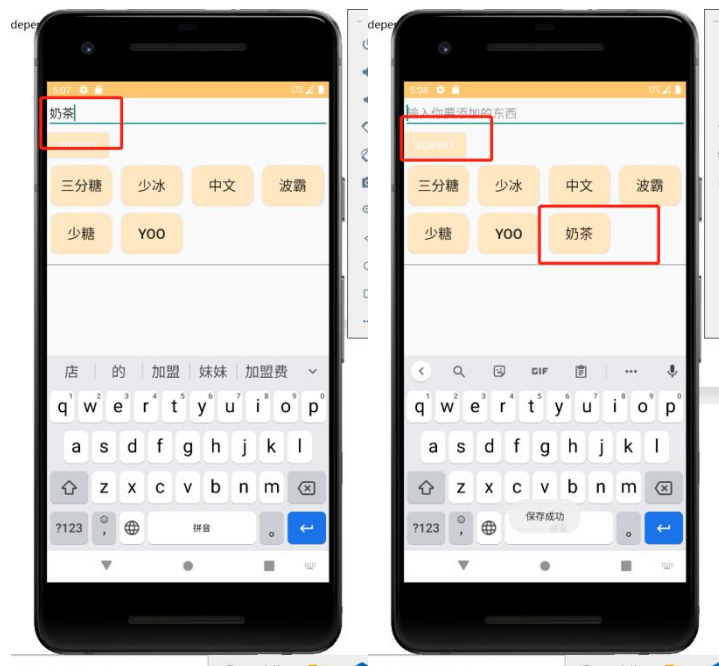
在社区页面的搜索框中输入想要搜索的奶茶内容点击回车——>得到搜索内容



⑩ 增加我的口味

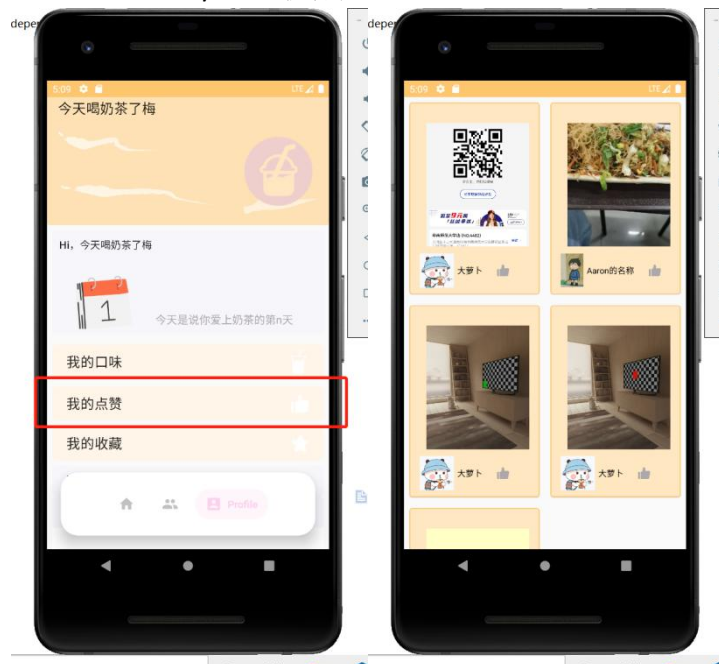
点击右下角导航栏最右边图标到达个人中心页面——>点击我的口味——>输入想要添加的口味——>点击 submit 即可

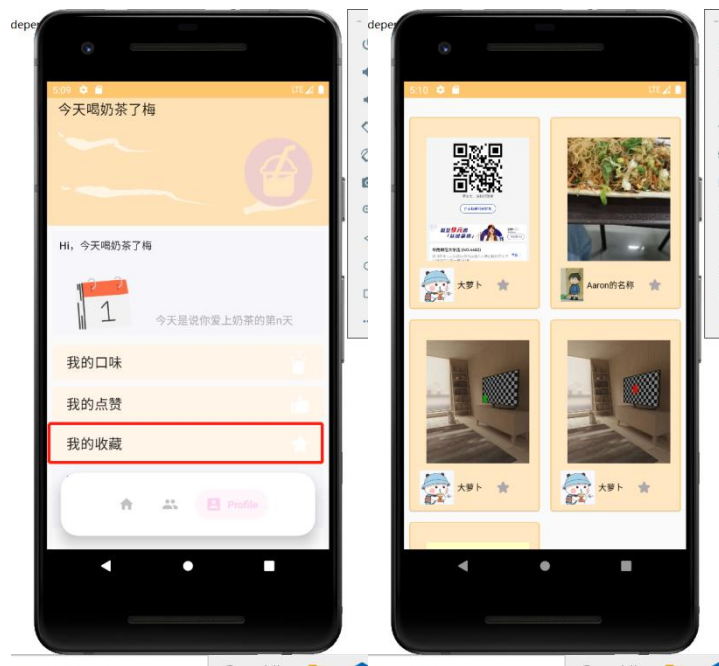




⑪ 查看我的点赞/我的收藏

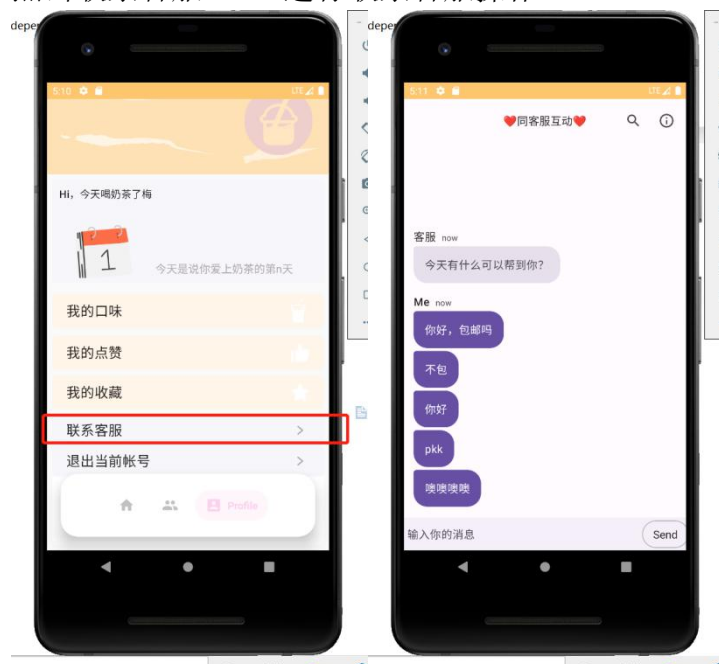
点击我的点赞/我的收藏





⑫ 联系客服功能

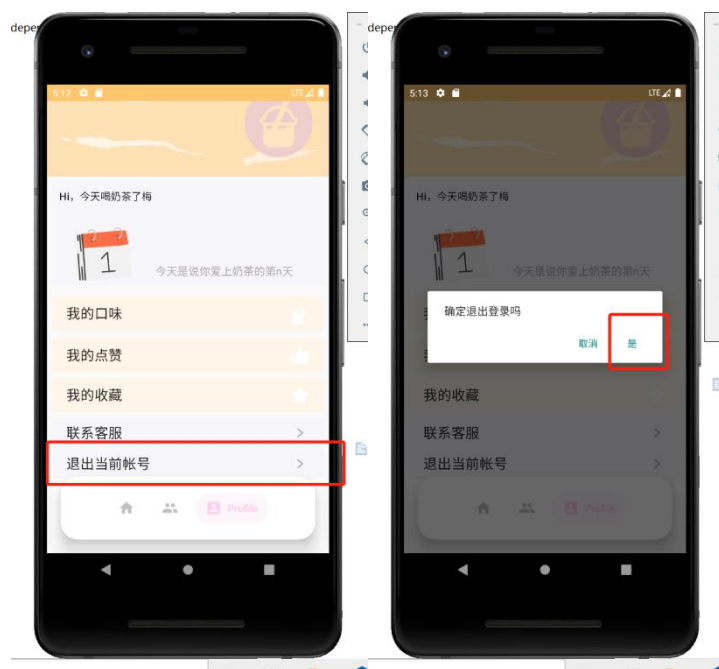
点击联系客服——>进行联系客服操作



⑬ 退出登录功能

在已登录的基础上有两个退出登录出口

1. 点击退出当前登录按钮



2. 长按头像处按钮

