目　录

第一章 需求分析

1.1 背景与目的

本小程序旨在构建一个卓越的校园服务平台，为师生提供便捷性、安全性和多样性，从而实现校园生活的完美互动与畅通交流。对标作品主要是有料同学、GCC 百宝箱等一些主要面向学生群体的优秀校园助手程序。

1.2 主要功能

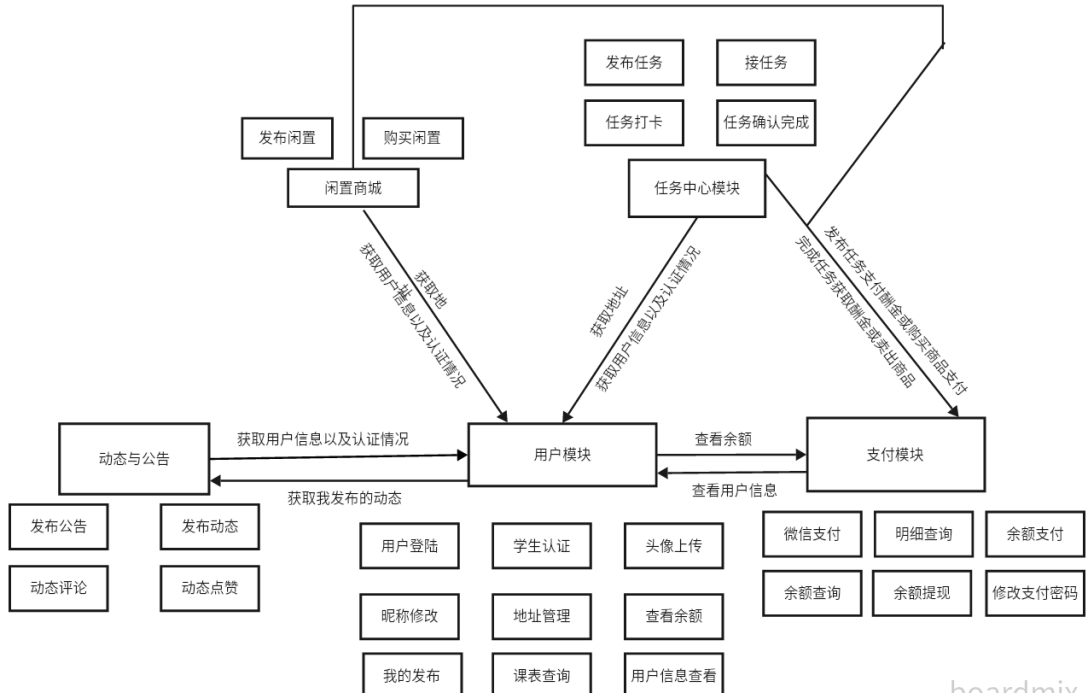

图 1-1 业务架构图

特别说明:以下功能需在个人设置上进行学生认证(可以通过上传学生证或者输入学号以及密码进行验证)，学校官方账号会分配一个专属的管理员账号,多个学校数据不共享。

如图 1-1 业务架构分为 5 个模块

官方公告：管理员发布学校重要通知和公告，师生可以点赞、关注及评论，以便学校更好地了解学生的动态。

动态专区：师生可自由发布、点赞、评论校内动态，分享校园生活。

二手闲置：学生可以发布二手物品信息，实现资源再利用（该平台采用学生证认证，以提高学生交易的安全性。详细信息请参阅架构设计)。

任务中心：用户可以自由发布校园活动、兼职等信息，促进勤工俭学，解决师生困难。

课程表：已绑定教务系统的学生，可通过小程序查看课表信息。

余额管理：小程序支持微信支付、收益提现、余额详情查询等。

地址管理：小程序可以保存常用地址，方便用户在任务中心或二手闲置中使用。

个人设置：用户可以设置昵称、头像、支付密码以及学生认证等。

1.3 竞品分析

现在市场是主流的校园小程序主要是有料同学，百度贴吧（校园吧），零点校园，叮点校园等等，但是他们都是存在着比较严重的问题，页面设计美观程度不够，广告泛滥，隐藏广告关闭按钮，功能上，比较少，如有料同学没有闲置商品，只是一个贴吧，并且贴吧内有大量的广告信息，部分不良色情信息等等，以上竞品均没有课表查询功能，没有学时学分查询功能，更没有学校官方公告信息展示，有的甚至非本校园信息，校园信息数据混乱合并等等

表 1-1 竞品比较表

	小轻（本程序）	有料同学	百度贴吧	零点校园	叮点校园
流畅度	✓	✓	✓	×	×
广告	×	✓	✓	✓	✓
私有化部署	✓	×	×	×	×
信息安全	✓	✓	✓	×	×
课表查询	✓	×	×	×	×
闲置商品	✓	×	×	×	×
官方公告	✓	×	×	×	×
动态	✓	✓	✓	✓	✓

第二章 概要设计

2.1 业务架构设计

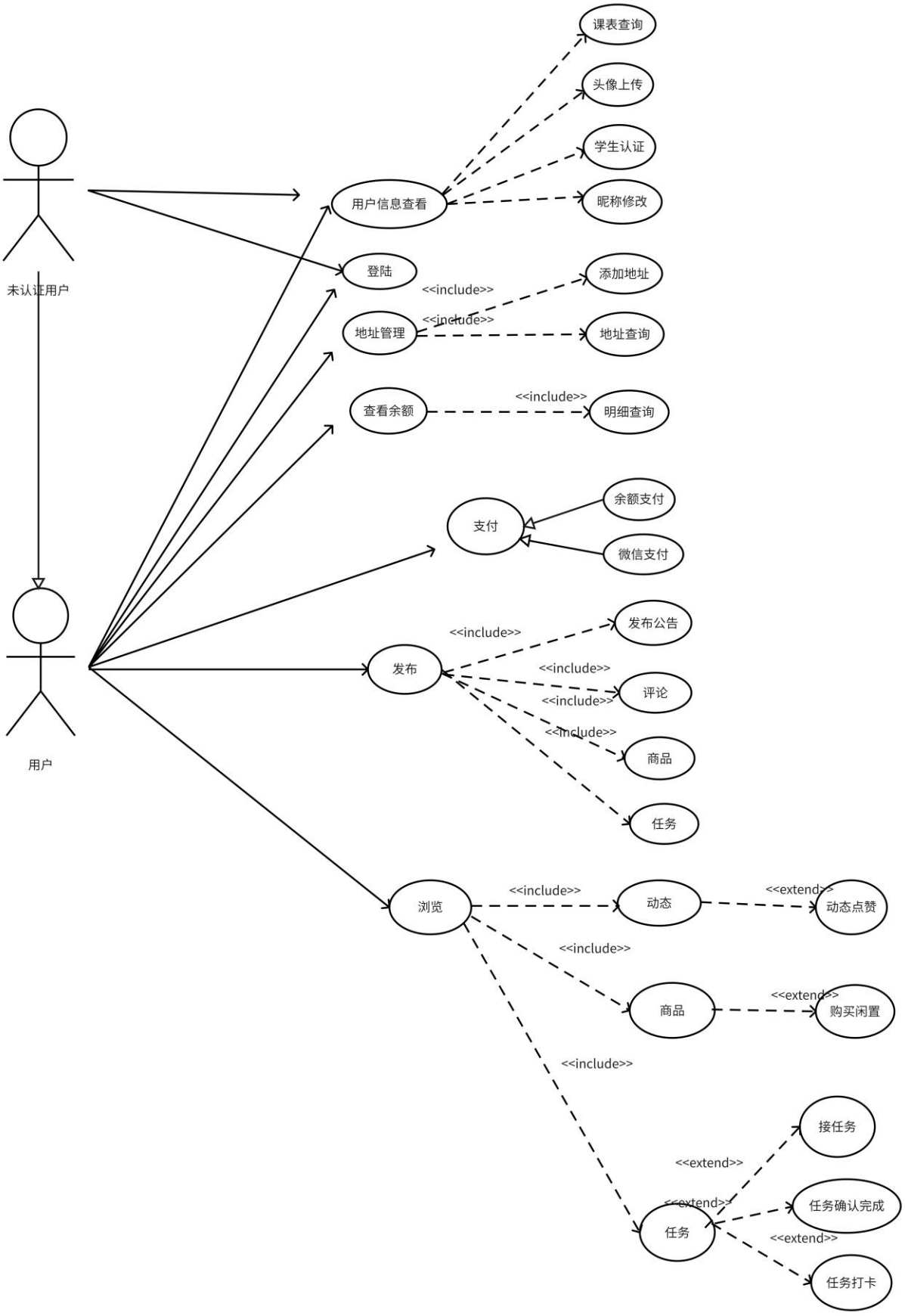
未认证用户
用户
用户信息查看
课表查询
头像上传
学生认证
昵称修改
登陆
地址管理
<<include>>
添加地址
<<include>>
地址查询
查看余额
<<include>>
明细查询
支付
余额支付
微信支付
发布
<<include>>
发布公告
<<include>>
评论
<<include>>
商品
<<include>>
任务
浏览
<<include>>
动态
<<extend>>
动态点赞
<<include>>
商品
<<extend>>
购买闲置
<<include>>
任务
<<extend>>
接任务
<<extend>>
任务确认完成
<<extend>>
任务打卡

图 2-1 程序用例图

如图 2-1 在程序用例图中，有两个参与者：未认证用户和认证用户。未认证用户拥有的用例只有登陆和用户信息查看,但是不包含课表查询;认证用户为普通用户,支持所有用例;

2.2. 应用架构设计

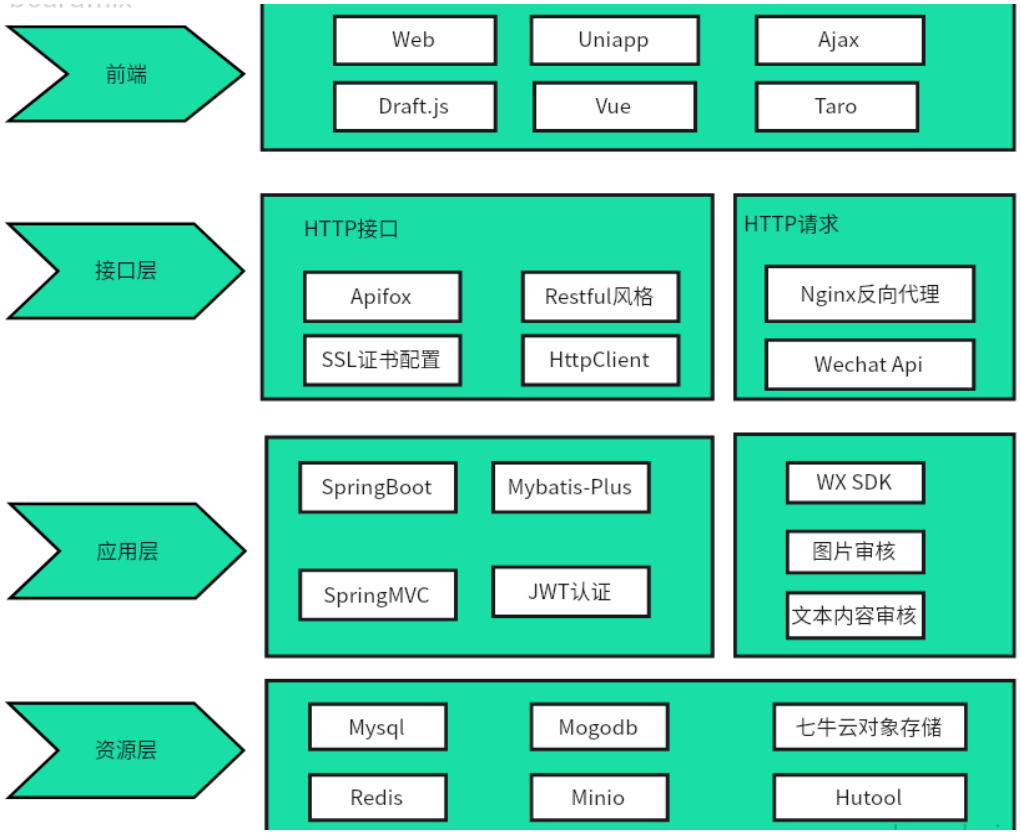

图 2-2 应用架构图

如图 2-2 应用架构图所示 开发该小程序主要用到的技术框架：

1. 数据库的重要数据均已进行加密,比如支付密码使用的是安全级别较高的 Bcrypt 加密，同时配置了 SSL 证书，支持 Https[4]，安全性得到了保障；
2. 存储数据使用了主流的数据库，Mysql 作为存储关系型较强的信息，Redis 用于保存缓存信息。由于发布时间较新的动态信息、部分用户的登陆状态信息等此类信息需要频繁读取，小程序的前端使用懒加载，提高了一定的性能；为了图片的响应速度得到保障，使用了七牛云的对象存储以及 CDN 加速，同时使用了 Minio 对象存储；
3. 由于本项目是基于小程序的开发,为此使用了 HttpClient 调用微信的 API 进行业务需求的书写，并且封装了一个 SDK 给项目调用，该 SDK 发布了在远程仓库；
4. 该小程序配置了后台管理系统(Web 端)，使用 Nginx 反向代理进行系统的部署，同时配置了 SSL 证书；
5. 该项目是前后端分离项目，前端使用了 Ajax 进行发送请求，后端接口文档使用 Apifox，类似于 Swagger 产品；
6. 后端基于 SpringBoot[2]进行项目的开发，使用基于 ORM(Object Relational Mapping)的框架 Mybatis-Plus 同时使用 JWT 作为身份认证，提高了安全性以及性能(JWT 的好处是减少了服务器的压力，同时 Token 解密需要密钥)。

第三章 详细设计

3.1 界面设计

该程序界面与主流的小程序界面不一样,如图 3-1， 图 3-3 拥有简洁优雅的界面设计,让人赏心悦目,清新明快的配色方案,保持界面元素的风格一致,包括字体,颜色， 图标等,提升了整体的美观度,为大学生营造了舒适的视觉氛围;

如图 3-2,图 3-4,图 3-6 界面内容通俗易懂,操作流程简单明了,用户能够轻松完成所需操作,易用性高,操作简单,用户可以轻松上手;

如图 3-5 课表展示,页面布局简约,元素清新,可以让用户快速知道课程信息.

本程序拥有流畅的动画,整体设计风格简洁清新,加上无广告特点,给用户留下美好的印象， 不像主流的校园类小程序,按钮错乱,广告多,页面美观度低,广告频繁,流畅度低等

图 3-1 首页(未登录)

图 3-2 官方公告

图 3-1 是用户没有登陆进去的首页,页面元素具有提醒登陆的特点

图 3-2 是官方公告区,主要展示学校官方发布的信息,如紧急通知,比赛信息等

图 3-3 我的页面

图 3-4 动态详情

图 3-3 是我的个人页面，展示了个人头像以及用户昵称信息，还有就是功能选择

图 3-4 动态详情，主要是展示动态内容，文章发布以及展示均采用富文本格式，更好的凸显有效信息

图 3- 5 课表查询

图 3- 6 我的设置

图 3-5 是课表信息展示,展示了我的课程信息

图 3-6 是我的设置页面,功能信息明确,操作简单,用户体验感更好

图 3-7 首页(已登录)

图 3-8 任务中心

图 3-7 是首页(在用户已经登陆下),展示用户发布的动态 ,记录和展示生活的美好瞬间

图 3-8 是任务中心,用户可以在这里挑选任务进行接单,勤工俭学

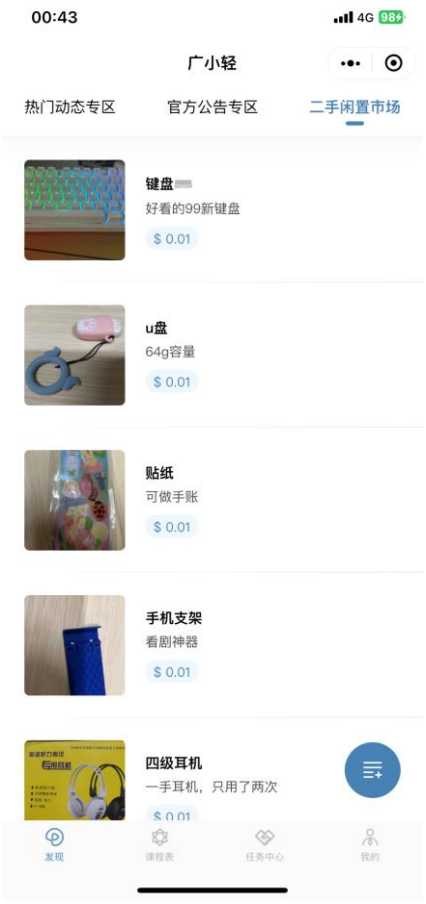

图 3-9 闲置市场

图 3-10 任务详情

如图 3-9 闲置市场,学生可以发布或者浏览购买商品,实现资源再利用

如图 3-10 用户可以浏览任务详情信息,可以选择去完成任务,获取酬金,勤工俭学

图 3-11 购买商品

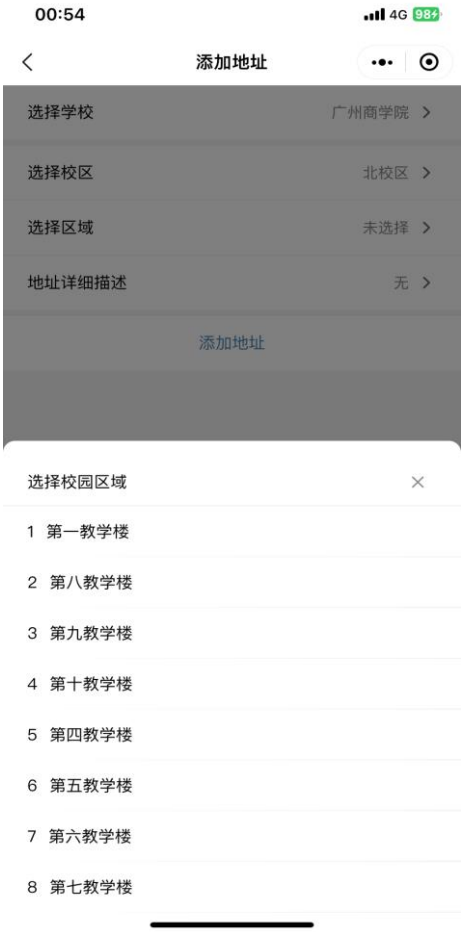

图 3-12 地址管理添加

如图 3-11 用户浏览完商品的详细信息,可以选择购买商品

如图 3-12 用户可以保存常用的地址信息,如宿舍位置,方便任务发布以及商品发布

3.2 数据库设计

程序的数据库设计如下图 3-13 数据库设计图以及表所示

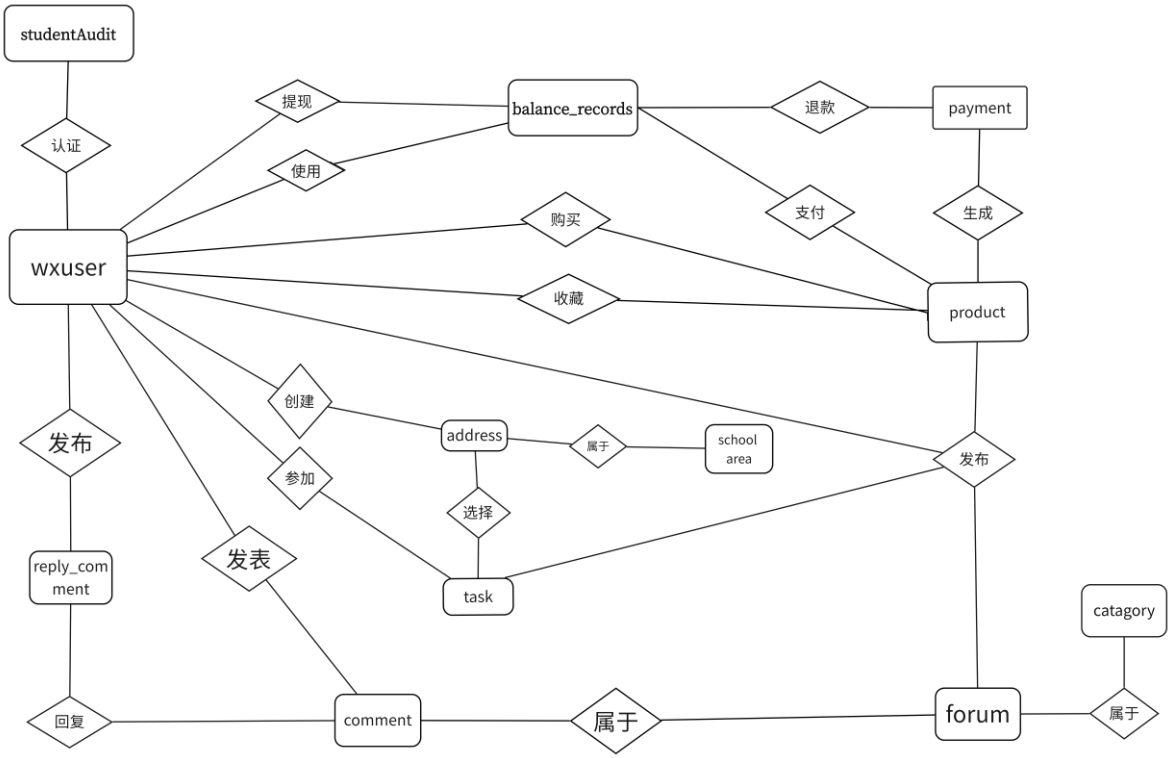

图 3-13 数据库设计图

表 3-1 wxuser

字段名称	类型	长度	是否主键	描述
id	varchar	255	Y	用户 id
openid	varchar	255		微信 openid
role	int			用户类型
user_name	varchar	255		用户名称
state	int			用户状态
avatar	varchar	255		用户头像
photo_number	varchar	255		手机号
password	varchar	255		密码
student_number	varchar	255		用户学号
password_jw	varchar	255		教务系统密码
password_pay	varchar	255		支付密码
balance	double			用户余额
create_at	timestamp			创建时间
update_at	timestamp			更新时间

表 3-2 task

字段名称	类型	长度	是否主键	描述
id	varchar	255	Y	任务 id
activity_title	varchar	255		任务标题

activity_description	text			任务描述
image_url	varchar	255		描述图片
number_of_participants	int			任务人数
location	varchar	100		地点
start_time	datetime			开始时间
end_time	datetime			结束时间
sign_out_code	varchar	50		签退码
status	tinyint	1		任务状态
price	double	10		任务价格
people	int			当前人数
initiator	varchar	100		发布者
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-3 form

字段名称	类型	长度	是否主键	描述
id	varchar	50	Y	帖子 id
title	varchar	255		标题
surface_image	varchar	255		封面
publisher	varchar	100		发布者
surface_description	varchar	255		封面描述
md_file_url	varchar	255		富文本 url
views	int			观看人数
category	varchar	50		分类 id
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-4 address

字段名称	类型	长度	是否主键	描述
id	varchar	255	Y	地址 id
region_id	varchar	255		区域 id
region_name	varchar	500		区域名称
detailed_address	varchar	1000		详细地址
creator	varchar	255		创建者
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-5 balance_records

字段名称	类型	长度	是否主键	描述
balance_id	varchar	255	Y	余额 id
user_id	varchar	255		用户 id
current_balance	double			当前余额
balance_type	int			支付/收入
balance_change	double			改变数值
withdrawal_order_id	varchar	255		提现 id

payment_order_id	varchar	255		payment_id
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-6 category

字段名称	类型	长度	是否主键	描述
id	varchar	50	Y	类型 id
category_name	varchar	255		名称
adder	varchar	255		添加者
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-7 comment

字段名称	类型	长度	是否主键	描述
comment_id	varchar	255	Y	评论 id
post_id	varchar	255		帖子 id
commenter_id	varchar	255		评论者 id
comment_content	varchar	1000		评论内容
sub_comment_count	int			内容字数
likes	int			点赞量
dislikes	int			不喜欢数量
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-8 product

字段名称	类型	长度	是否主键	描述
product_id	varchar	255	Y	商品 id
product_title	varchar	255		商品标题
product_price	double			商品价格
product_unit	varchar	50		商品单位
product_description	varchar	1000		商品描述
product_image	varchar	1000		商品图片
front_image	varchar	1000		封面
product_status	int			商品状态
product_address	varchar	255		地址
publisher_id	varchar	255		发布者 id
favorites_count	int			收藏量
views_count	int			观看量
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-9 reply_comment

字段名称	类型	长度	是否主键	描述
reply_id	varchar	255	Y	回复评论 id
user_id	varchar	255		用户 id

comment_id	varchar	255		评论 id
content	text			评论内容
likes	int			点赞量
dislikes	int			不喜欢量
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-10 school_area

字段名称	类型	长度	是否主键	描述
id	int		Y	位置 id
code	varchar	255		位置代号
name	varchar	255		名称
school	varchar	255		
campus	varchar	255		
area	varchar	255		

表 3-11 student_audits

字段名称	类型	长度	是否主键	描述
audit_id	varchar	255	Y	认证 id
user_id	varchar	255		用户 id
state	tinyint			认证状态
remarks	varchar	255		备注标记
audit_image_url	varchar	255		认证图片
created_at	timestamp			创建时间
updated_at	timestamp			更新时间
studentId	varchar	255		认证学号

表 3-12 favorits

字段名称	类型	长度	是否主键	描述
favorite_id	varchar	255	Y	收藏 id
favorite_product_id	varchar	255		商品 id
favorite_user_id	varchar	255		用户 id
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-13 link_task

字段名称	类型	长度	是否主键	描述
id	varchar	255	Y	id
task_id	varchar	255		任务 id
participant_id	varchar	255		用户 id
is_signed_out	tinyint	1		是否完成
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-14 payment

字段名称	类型	长度	是否主键	描述
out_trade_no	varchar	255	Y	支付 id
product_id	varchar	255		商品 id
status_code	varchar	255		状态
status_number	int			状态数字
recipient	varchar	255		微信官方参数
payer	varchar	255		支付者
package_value	varchar	255		微信官方参数
pay_sign	varchar	255		微信官方参数
time_stamp	varchar	255		微信官方参数
nonce_str	varchar	255		微信官方参数
sign_type	varchar	255		微信官方参数
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-15 refund

字段名称	类型	长度	是否主键	描述
out_refund_no	varchar	255	Y	退款 id
out_trade_no	varchar	255		退款订单号
total_fee	int			总费用
refund_fee	int			退款费用
status_code	varchar	10		状态码
status_number	int			状态
create_time	timestamp			创建时间
update_time	timestamp			更新时间

表 3-16 withdrawal

字段名称	类型	长度	是否主键	描述
withdrawal_id	varchar	255	Y	提现 id
user_id	varchar	255		用户 id
status	int			提现状态
withdrawal_amount	double			提现账号
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

表 3-17 followers

字段名称	类型	长度	是否主键	描述
follow_id	varchar	255	Y	关注 id
user_id	varchar	255		用户 id
followed_user_id	varchar	255		被关注的 id
created_time	timestamp			创建时间
updated_time	timestamp			更新时间

表 3-18 province

字段名称	类型	长度	是否主键	描述
id	int		Y	id
code	bigint			地区代码
name	varchar	32		地区名称
province	varchar	32		省级
city	varchar	32		市级
area	varchar	32		区级
town	varchar	32		镇级

表 3-19 like_dislike

字段名称	类型	长度	是否主键	描述
id	varchar	255	Y	点赞 id
userId	varchar	255		用户 id
comment_id	varchar	255		评论的 id
status	int			状态
created_at	timestamp			创建时间
updated_at	timestamp			更新时间

注意:

任务表的 statu:

0 开始接单（已经支付了）1 是结束（钱已经付款给接单者）2 是已经被接单了，

3 是未支付，算是暂时保存了（是已经创建了订单了）5:是没有调用微信订单，

6 是申请退款的（没有被接单的时候，也就是 0 的时候），

7 是取消订单（已经被接单了，是接单的取消）然后设置为 0，所以 7 可能不存在，因为取消订单不需要发布者同意 ）

8 是接单者确定送达（数字 1 是发布者按下确认送达后打款）（正常顺序是 530281）

user 表的 statu:* 状态，正常是 0（默认（未认证）正常（0）、封禁（2）、注销（3）、已认证（1），5 是上传了证件照认证的（在 1 的基础上））

3.3 关键技术

3.3.1 如何保障支付的安全性

支付安全是任何支付系统的核心，主要涉及数据库安全和支付身份验证两个方面。
程序采取了以下方案进行提高安全性：

1. 严格控制数据库访问权限：本程序数据库采用多用户,不同权限配置；
2. 定期进行数据库备份：服务器配置了 Shell 脚本定期备份；
3. 程序的支付密码设置为独立的，并使用 Bcrypt 进行密码的加密保存。

3.3.2 如何防止订单被恶意拍下拒绝付款以及卡订单发布导致其他用户拍不了该订单

1. 程序对接了微信支付的查询订单情况 API 以及微信支付回调 API，用户支付成功后，微信服务器访问本程序指定的接口，实现回调功能；
2. 用户下单时，程序设置定时器并加入多线程。设置 15 分钟后可通过微信支付订单查询用户是否成功支付；若用户提交订单但未付款，系统将自动取消订单。

3.3.3 任务中心如何确保接单人员以及完成任务

1. 程序采取双方确认机制：在双方确认送达后即可完成任务,完成任务后接单员即得酬金。
2. 程序采取拍照并上传，可给发布者确认完成任务。

3.4.4 图片渲染速度以及程序响应速度的慢

1. 程序采取 Redis 缓存保存部分 "高读取,低修改"的数据，如登陆的 Token 信息、动态的前 Top 10 的文章等，可减少数据库的读取。
2. 程序采用两种对象存储混合使用,采用设计模式的策略模式,减少单一对象存储的读取压力,并对某个网络延迟较高的对象存储使用 CDN 加速。
3. 小程序前端已实现懒加载,将一部分缓存保存在用户手机上,从而减少服务器请求次数。

3.4.5 课程表数据保存

问题: 通过 API 可获取到了课表信息，但格式为 JSON 格式，不适合保存在 Mysql 中。
程序采取 ORM 映射，但 JSON 格式映射仍会映射成 String，且通过一些工具类进行 JSON 读取相关信息相当于遍历字符串；不同用户的数据对应 JSON 格式的内容不同，其内容涉及到时间、日期、课程以及地点不一致，查询当天课程等需遍历一整个 JSON 字符串，耗时较久，由于 Sql 语句的书写较为繁琐，需使用子查询，给数据库带来一定的压力。
解决方案: 使用 Mogodb 而不是 Mysql
1. 由于原始数据是 JSON 格式的非关系型数据,在关系型数据库 Mysql 下保存数据,每个用户的数据字段不唯一，耦合性过高；在非关系型数据库 Mogodb 下,不存在这个问题。
2. 使用 Mogodb 保存的格式为 JSON 格式，读取相关的信息方便，可根据键值对进行遍历查询,可实现根据日期进行查询或者根据其他键值对查询。

3.4 接口设计

本程序制定了一套 API 设计规范：

1. API 风格尽量使用 RESTful[1]风格
2. 响应格式

```
interface APIResponse<T> {
  data: T;
  code: number;
  msg: string;
```

}

- ✧ data：JSON 格式的响应数据。
  - ■ 出于可扩展性考虑，data 必须是对象不能是数组。
  - ■ 如果请求失败，data 可以为 null。
- ✧ code：自定义状态码，非 HTTP 状态码。
- ✧ msg：额外消息，可用于请求失败时的错误提示。
- ✧ 字段命名风格：返回的字段统一采用驼峰命名法。

3. ID 生成
4. 统一使用 uuid。
5. 时间格式:返回的时间格式统一用 Unix 时间戳。
6. HTTP 状态码

遵循 HTTP 状态码规范，例如：

- 资源创建成功返回 201 状态码。
- 授权认证失败返回 401 状态码。
- 资源不存在则返回 404 状态码。

7. OpenAPI[3] 参见 OpenAPI[3] 规范 (中文版) https://openapi.apifox.cn/
8. 参数检验

API 规范文档(推荐):完整文档在本目录的 API 规范文档 PDF

在线链接(推荐):
https://apifox.com/apidoc/shared-7a42976b-cc7d-4e4c-979f-b9fc95139fd3

第四章 测试报告

4.1 测试方案

4.1.1 测试需求

本次测试范围为小程序的核心模块。为验证系统在大负荷情况下数据处理能力及承受能力，分别模拟系统登陆、浏览动态以及任务中心、发布动态以及任务商品等业务场景，分别从响

应时间、事务成功率、CPU 使用率、内存使用情况等维度进行结果分析。

4.1.2 测试标准环境

并发用户数	压测时长	90%用户相应时间	平均响应时间(s)	事务成功率	CPU 占用率	内存使用率
5000	15min	小于 3 秒	小于 3 秒	大于 99%	小于 75%	小于 75%
10000	15min	小于 4 秒	小于 4 秒	大于 99%	小于 75%	小于 75%
15000	15min	小于 5 秒	小于 5 秒	大于 99%	小于 75%	小于 76%

表 4-1 测试通过标准

服务器及客户端	硬件配置	软件配置
应用服务器 （3 台）	单节点配置： CPU：2 核，内存：2GB （集群总）配置： CPU：8 核，内存：8GB 运行环境：Docker 容器	操作系统: （CentOS 7.4）
数据库服务器	配置： 存储：1TB （SSD） CPU：2 核，内存：2GB	操作系统: （CentOS 7.4） 数据库：（Mysql 8.0）
测试客户端	CPU：16 核，内存：16G，存储：512GB	操作系统：（windows11）
网络要求	3M 带宽	

表 4-2 测试环境

4.1.3 测试工具

LoadRunner 性能测试工具、Nmon 服务器指标监测工具、Postman , Apifox 接口测试工具、Fiddler 抓包工具

4. 2 测试方案

首先应用服务集群基于 Docker 容器部署在云平台上，应用集群由节点数可手动扩展，本次压测设置了 3 个节点，单个 Docker 容器节点的配置为 2h CPU、2GB 内存，应用服务集群采用 Nginx 负载均衡 作为第一层负载，由 应用服务器 作为第二层负载对外提供服务。

接着关系数据库采用 Mysql 集群提供数据存储服务，应用程序通过连接池的方式与数据库建立连接。热点数据使用 Redis 缓存，集成接口及应用程序的异步处理采用了 Ajax 的方式。
压力测试客户端采用 10 个 LoadRunner 客户端组成压测集群，根据测试场景模拟用户用户数和并发数。

4. 3 测试结果

4.3.1 测试过程

图 4-1 Apifox 测试

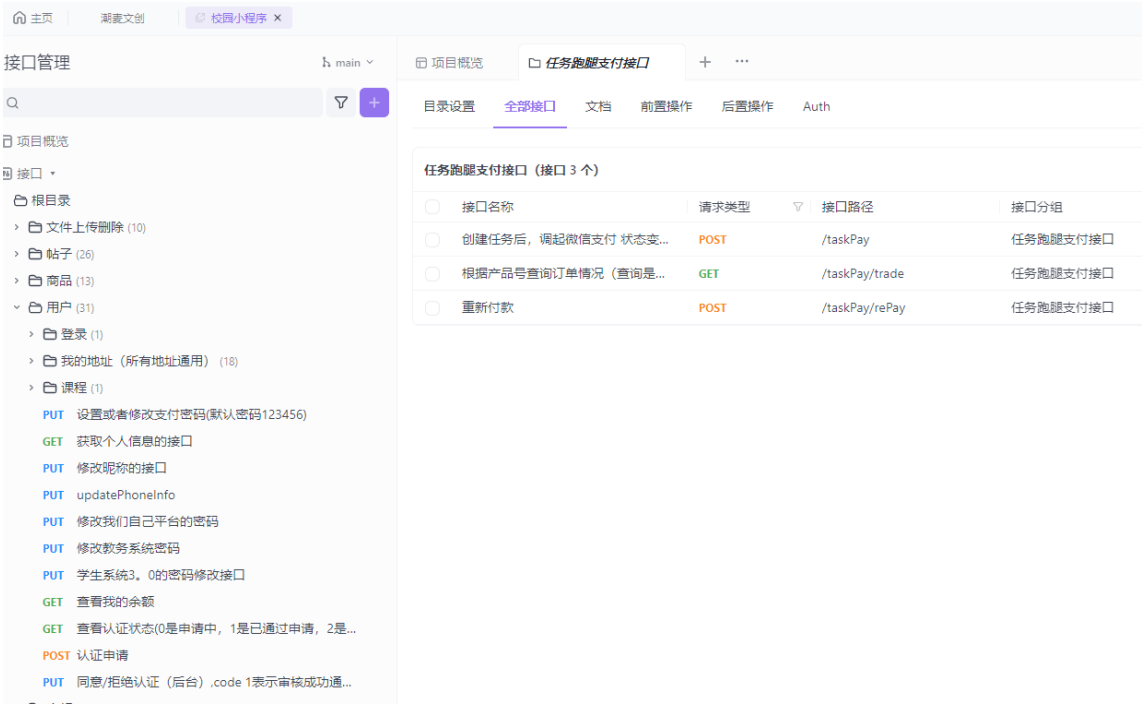

图 4-2　支付接口示例

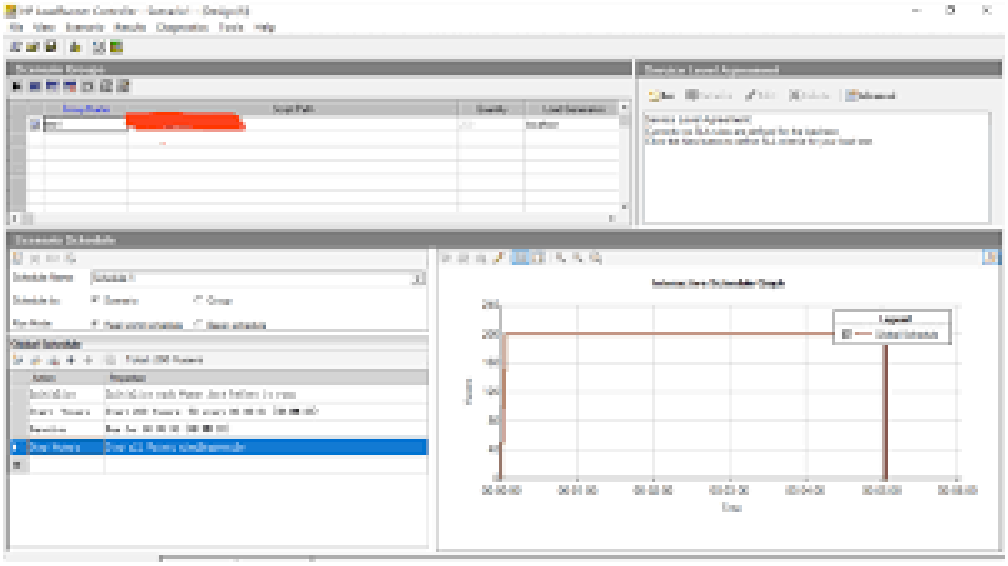

图 4-2　LoadRunner 测试

如图 4-1 是我们程序配置了 Apifox 的 openapi 脚本进行测试,可以实现跨平台和离线测试,如图 4-2 是利用 openapi 脚本导入后的支付接口的描述,同时为了程序的稳定性，在上线测试版本之前,如图 4-3 我们团队采用 LoadRunner 进行压力测试;保证了程序的稳定性,提升了用户体验

各场景数据统计分析如下：

场景	并发用户数	压测时长	90%的用户响应时间(s)	平均响应时间(s)	事务成功率	每秒处理事务	成功事物数	失败事物数	脚本运行错误数
登录	100	15min	4.625	2.391	99.28%	9.371	14358	103	205
	200	15min	6.039	3.753	98.19%	9.125	18770	345	523
	500	15 min	12.748	6.452	91.18	10.621	36134	3493	5261
浏览动态	300	15min	8.982	4.863	95.74%	11.527	26891	1202	305
	700	15min	18.208	10.603	92.23%	6.215	47582	8193	731
	500	10 min	11.748	6.452	92.32	11.338	26134	1493	3221
动态发布	100	10min	2.435	2.098	99.32%	8.371	9358	558	205
	200	10min	5.272	3.353	96.23%	8.125	12738	380	233
	500	10 min	11.748	6.452	92.32	11.338	26334	1293	3221
任务浏览	100	10min	2.435	2.098	98.73%	11.527	15811	1202	305
	200	10min	5.239	3.353	97.23%	10.125	16770	380	258
	500	10 min	8.784	7.112	94.32	11.138	26114	1423	3121

场景	并发用户数	压测时长	90%的用户响应时间(s)	平均响应时间(s)	事务成功率	每秒处理事务	成功事物数	失败事物数	脚本运行错误数
任务发布	**500**	**10 min**	**11.328**	**6.129**	**90.14**	**10.923**	**25128**	**1815**	**3991**
课表查询	**500**	**10 min**	**12.732**	**7.212**	**91.22**	**10.525**	**22162**	**1612**	**3751**

4.3.2 结论

基于目前的测试结果，对比我们制定的压测标准，测试！目前可以满足约 600 并发用户，大约为 8600 人在线，完全可以满足客户需求。

第五章 安装及使用

5.1 如何部署

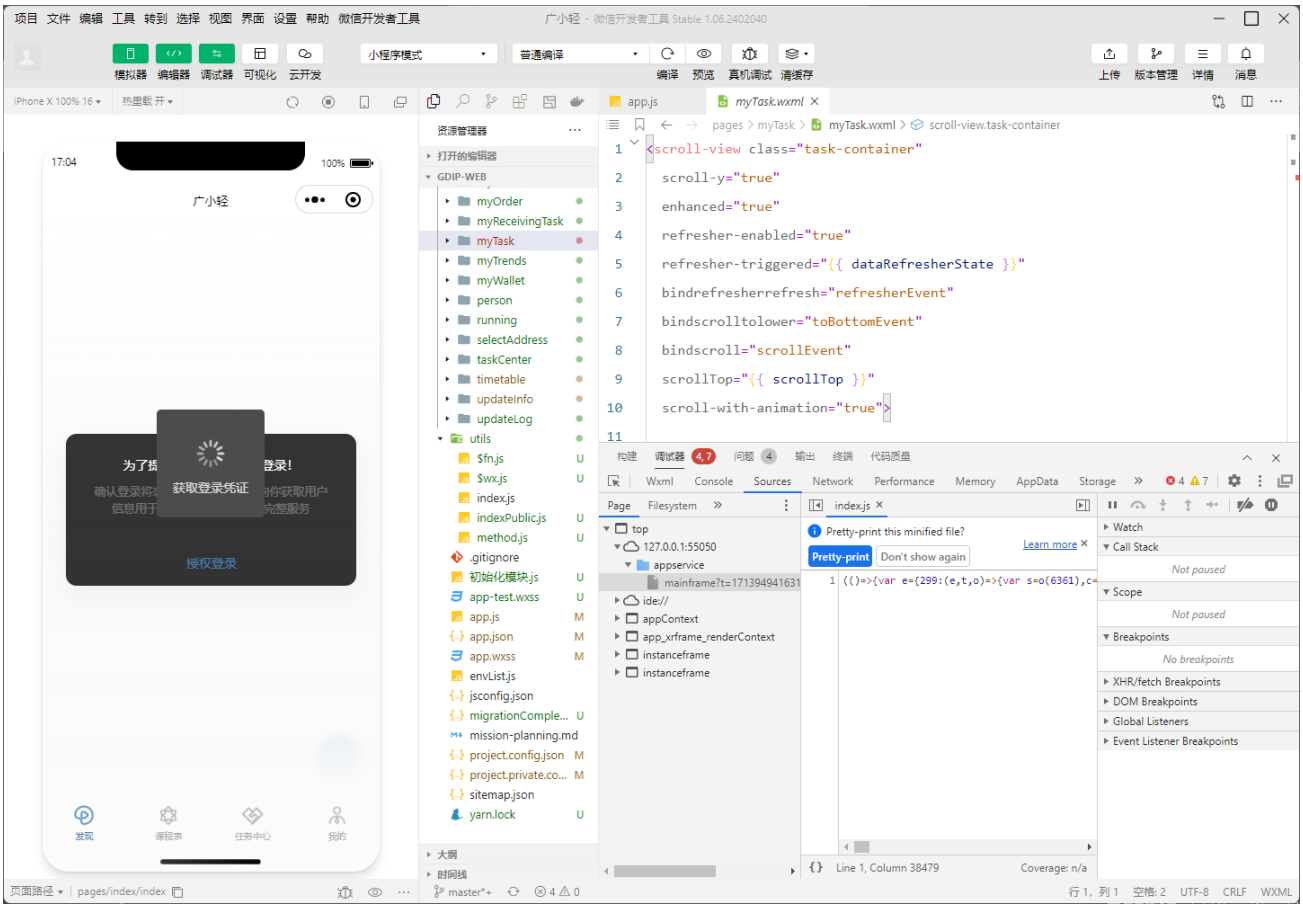

图 5-1 小程序前端开发

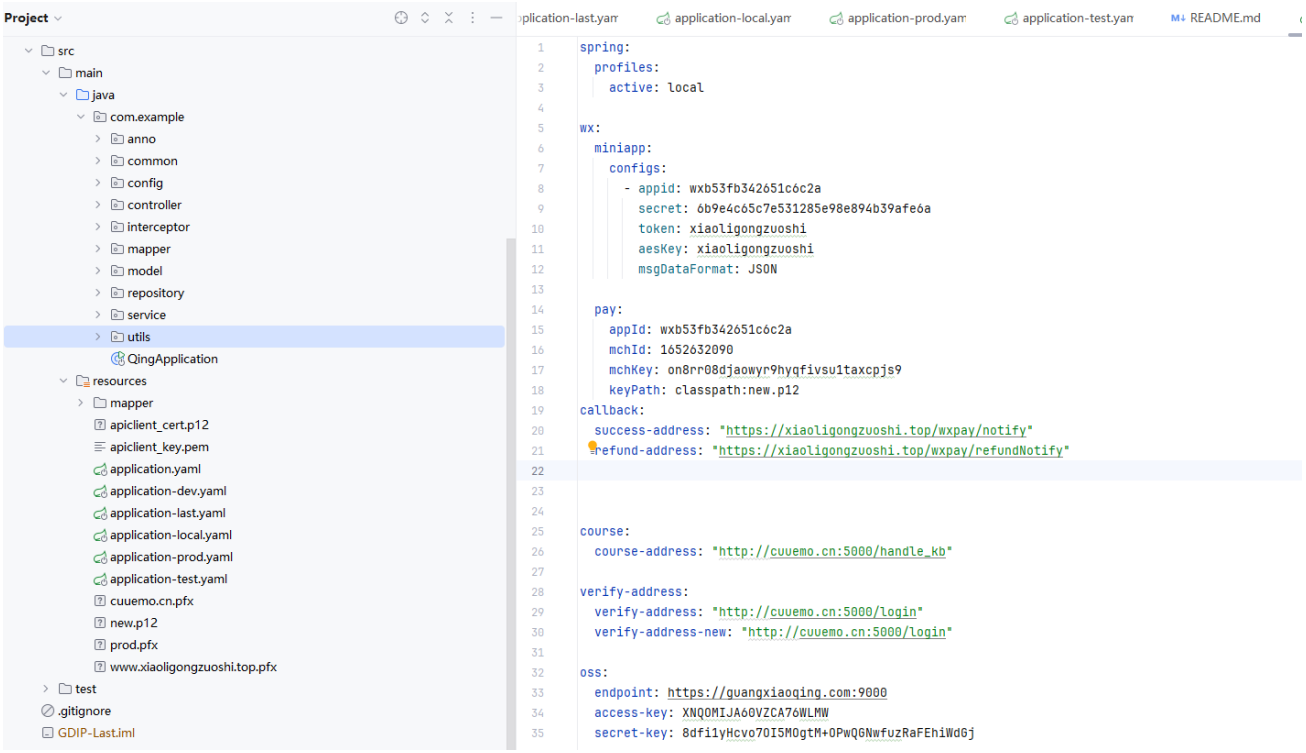

图 5-2 后端多环境配置

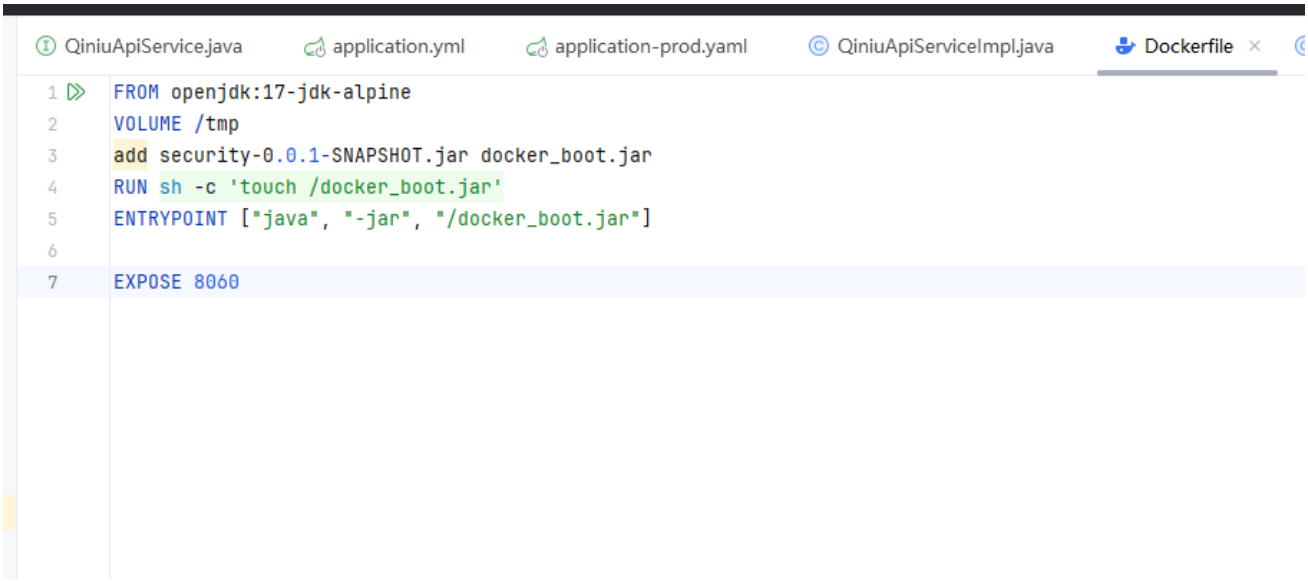

图 5-3 docker 后端构建文档

如图 5-1 小程序前端开发所示,本项目采用 uniapp 作为前端开发框架，并结合微信原生语言进行开发，具备以下部署优势：

前端部署

- **轻量便捷**：uniapp 基于 Vue 语法，无需安装大量依赖，代码体积小，部署简单快捷。
- **多端兼容**：打包编译后可生成多端应用，支持微信小程序、App 端、H5 网页等，一次开发多端运行。
- **性能优化**：uniapp 提供性能优化方案，可有效提升应用加载速度和流畅度。

后端部署

- **多环境部署**: 如图 5-2 后端多环境配置 所示,程序支持 local 环境、prod 环境以及 docker 环境，满足不同场景的部署需求。
- **Docker 镜像**: 如图 5-3 docker 后端构建文档所示,后端配置提供 Dockerfile，可轻松构建镜像，实现应用的快速部署和转移。
- **降低难度**: docker 简化部署流程，降低运维成本和操作难度，运维人员易上手。

整体部署特点

- **灵活便捷**: 前后端采用轻量化框架和多环境部署方案，部署灵活便捷，易于维护。
- **高扩展性**: 良好的代码架构和扩展性，支持后续功能平滑升级和业务扩展。
- **高性价比**: 降低部署成本和运维难度，提高项目性价比。

总结

本项目的部署方案充分考虑了项目开发和运维的易用性、灵活性和扩展性，为项目落地提供了坚实的技术基础。

5.2 部署情况调查

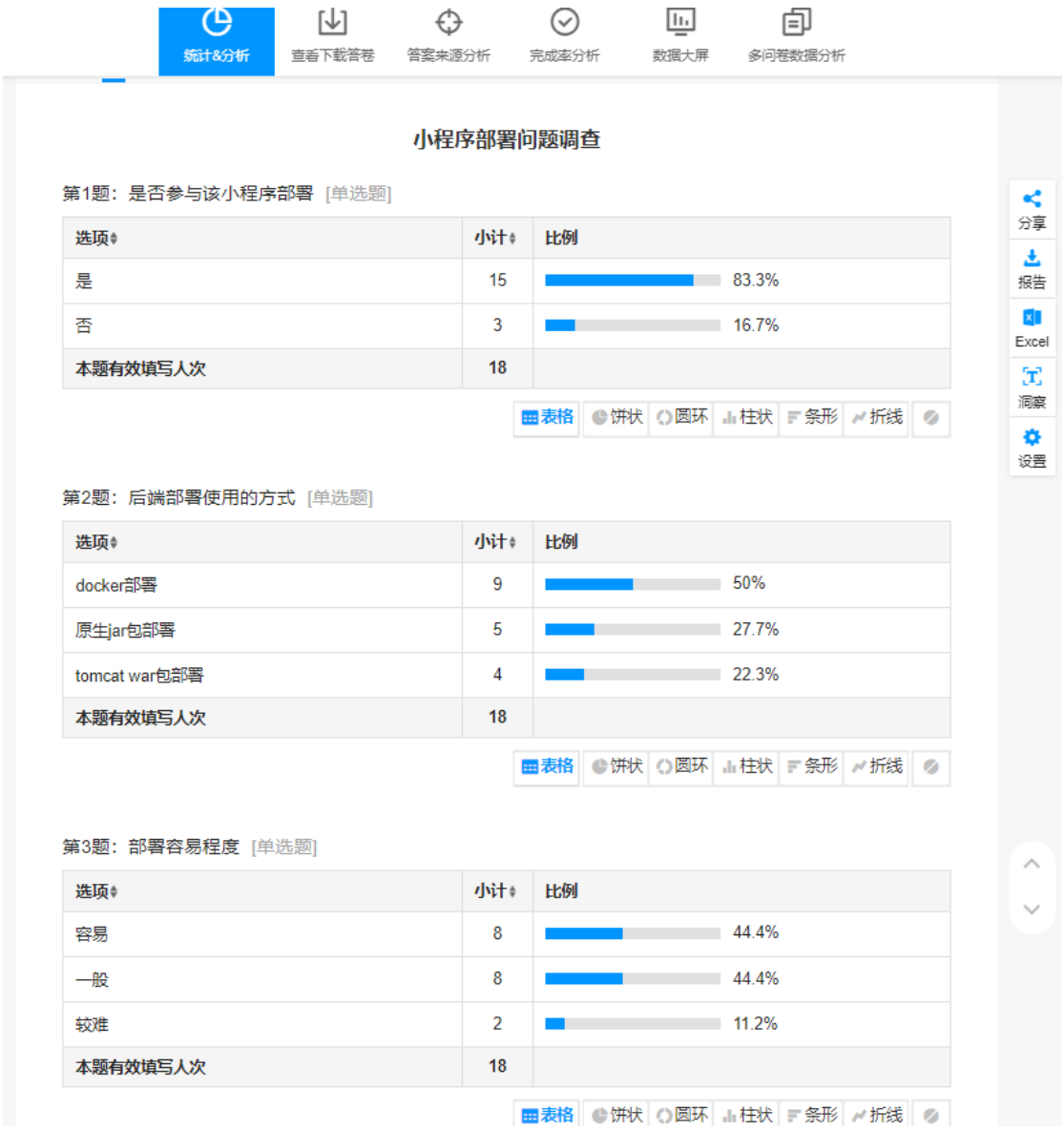

图 5-4 针对部署人员调查分析

本项目支持多校**私有化部署**，并采用**数据隔离**技术，有效保障各校数据安全。如图 5-4 所示我们对不同学校的部署人员进行问卷调查,可以看出相比同类产品，我们具备以下显著优势：

部署灵活易维护

- 简易的操作流程，**无需复杂的技术背景**，即使是普通运维人员也能轻松完成部署。
- 支持多环境部署，包括**本地环境、生产环境和 Docker 容器**，满足不同场景的部署需求。

数据安全可靠

- 采用**数据隔离**技术，各校数据**互不连通**，有效防止数据泄露和安全风险。

- 支持**私有化部署**，数据存储在学校机房内，数据安全更有保障。

超越同类产品

- 与同类型产品 **有料同学** 和 **易校园** 相比，部署更加简便快捷，运维成本更低。
- 优于大部分同类产品，**支持私有化部署**，满足对数据安全有更高要求的学校需求。

总结

本项目凭借灵活易维护的部署方案、可靠安全的数据保护措施和超越同类产品的优势，树立了多校私有化部署软件的标杆，为教育信息化建设提供优质的选择

第六章 项目总结

6.1 团队开发心得总结

本项目致力于构建一个卓越的校园服务平台，为师生提供便捷性、安全性和多样性，实现校园生活的完美互动与畅通交流。本项目的目标是对标有料同学、GCC 百宝箱等优秀校园助手程序，为学生群体提供更好的校园生活服务。

在项目开发过程中，我们主要实现了以下主要功能模块：官方公告、动态专区、学生服务（二手闲置、任务中心、课程表）、系统配套功能（余额管理、地址管理、个人设置）等。同时，我们也进行了业务架构设计、应用架构设计、界面设计、数据库设计等方面的工作。

在技术实现方面，我们采用了一系列先进的技术框架和安全措施，包括数据库加密、SSL 证书配置、微信支付、Nginx 反向代理、前后端分离、JWT 身份认证等。这些技术的运用，保障了系统的安全性、性能和用户体验。

我们认识到支付安全、订单管理、任务中心的管理等问题是需要特别关注的重点。同时，我们也应该关注系统的运行速度、安全性、扩展性、部署方便性和可用性等技术指标，以保证整个系统的稳定运行和用户满意度。

在项目其他方面，我们需要不断总结经验，提高团队协作能力，不断优化产品体验，以期实现项目的长期发展和商业价值。

本项目是一个充满挑战和机遇的创新之路，我们相信通过团队的不懈努力和持续创新，一定能够打造一个卓越的校园服务平台，为广大师生提供更好的校园生活体验。

6.2 市场调查分析

图 6-1 小程序调查问卷

图 6-2 小程序后台数据

图 6-3 小程序版本信息

如图 6-2 小程序后台信息可知.我们的小程序受到广大校友的关注与使用.累计用户接近 7000,日常使用用户接近 3000

我们团队为了完善优化该小程序.在校做过校园问卷调查,如图 6-1 小程序调查问卷,调查人数为 3700 人左右.92%的用户觉得小程序功能完善,界面美观.另外在调查意见中,96%的用户希望我们能够加入课表推送功能,82%的用户希望我们加入 AI 助手功能,从中可以看出我们团队研发的小程序,是受广大校友的喜爱的,广大校友的建议新功能开发也受到我们开发团队的关注;如图 6-3 小程序版本信息可知,我们的小程序在不断的接受用户的建议,版本升级,提升用户体验

6.3 迭代方案

如上图所示,程序已经运营了一定的时间,版本更新也去到了 V2.5.6,经过前几次的迭代更新,程序的一些 bug 已经修复,同时我们也做了大量的市场调查分析,给予我们迭代更新方向;

1.加入 AI 助手;在当今这个 AI 大语言模型的时代,大部分用户期望小程序加入 AI 助手的功能,AI 助手主要提供学校校内公告查询,如咨询 AI 助手比赛相关信息,咨询学时的学分,课程信息,咨询放假信息,或日常生活的问答,在我们团队的考虑下,下一个版本加入 AI 助手功能,AI 进校园是一种趋势, 因为问答数据知识库是管理员进行上传,所以能解决学生的大部分问题

2.加入课表推送功能,用户可以关注我们后续开发的公众号,公众号绑定用户信息,能够获取明天课程表,在规定的时间内推送给用户,减少了用户查询课表的时间以及满足用户的期望

3.采取更加智能的信息认证审核方式,如自动审核,利用 OCR 识别技术,加入训练学生证数据,可以实现自动审核功能,如果自动审核失败转为人工手动审核,此优化方案可以大大降低了人工成本以及审核时间,提升用户的体验感

参考文献

[1]. Web 端与移动端应用平台技术融合的研究 胡强
[2]. 基于 SpringBoot 的域名信息系统设计与实现 河北工业大学 雷欣
[3]. Web 应用与 OpenAPI 对接的适应性框架的研究 燕山大学 张乾
[4]. 基于国密 HTTPS 的网站安全认证 国家信息中心信息与网络安全部 王森