软件测试分析报告

城院生态圈

——基于微信小程序的城院动植物交流论坛



项目名称： 城院生态圈

专业班级： 软件工程1903

小组成员：黄依豪，梁晓勇，李东泽

指导教师： 杨枨老师

软件测试分析说明

说明：

《软件代码规范说明》 (STR)是对计算机软件配置项CSCl,软件系统或子系统，或与软件相关项目执行合格性测试的记录。通过STR，需方能够评估所执行的合格性测试及其测试结果。该文档可能还要用《软件测试用例说明》和《软件代码走查说明》以补充。

**版本记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 修订日期 | 版本/状态 | 修订人 | 备注 |
| 01 | 2021/12/7 | 0.1 | 黄依豪 | 报告的初始版本制作 |
| 02 | 2021/12/12 | 0.2 | 黄依豪、李东泽、梁晓勇 | 白盒测试、黑盒测试、单元测试补充 |
| 03 | 2021/12/14 | 0.3 | 黄依豪 | 单元测试、集成测试补充、剩余部分补充 |
| 04 | 2021/12/26 | 0.4 | 黄依豪、李东泽 | 测试用例明确值，单元测试、集成测试继续补充 |

目录

[软件测试分析报告 1](#_Toc3619)

[软件测试分析说明 2](#_Toc13531)

[1引言 4](#_Toc29930)

[1.1标识标题：《软件测试计划》 4](#_Toc14664)

[简称：STP 4](#_Toc15098)

[版本号：0.0.1 4](#_Toc29746)

[说明：本文档适用于基于微信小程序的城院生态圈，支持Office、WPS等word文档阅读软件 4](#_Toc19853)

[1.2系统概述 5](#_Toc12966)

[1.3文档概述 5](#_Toc28495)

[1.4与其他计划的关系 5](#_Toc30502)

[1.5基线 6](#_Toc24859)

[2引用文件 6](#_Toc11775)

[3测试概要 7](#_Toc14718)

[3.1测试环境 7](#_Toc18737)

[3.1.1时间与地点 7](#_Toc10612)

[3.1.1人员与设备 7](#_Toc10435)

[4测试分析 7](#_Toc8194)

[4.1黑盒测试 7](#_Toc6769)

[4.1.1功能测试 7](#_Toc2961)

[4.1.2等价划分测试 23](#_Toc21529)

[4.1.3边界值分析测试 30](#_Toc12090)

[4.2白盒测试 30](#_Toc9816)

[4.3单元测试 34](#_Toc29544)

[4.3.1测试工具 34](#_Toc20782)

[4.3.2测试用例 34](#_Toc5229)

[3.1登录 34](#_Toc29349)

[3.2发帖 35](#_Toc26383)

[3.3论坛模块 36](#_Toc1997)

[3.4动物城友会模块 36](#_Toc27149)

[3.5动植物科普模块 37](#_Toc31379)

[3.6主页模块 38](#_Toc21797)

[4.4集成测试 39](#_Toc19249)

[4.4.1测试工具 39](#_Toc6456)

[4.4.2结构分解 40](#_Toc6736)

[4.4.3详细过程 40](#_Toc8814)

[4.5系统测试 41](#_Toc16663)

[测试方法： 41](#_Toc29828)

[测试结果： 41](#_Toc20144)

[4.7确认测试 45](#_Toc15037)

[4.7.1 Alpha测试 45](#_Toc16151)

[G005小组项目访谈记录 45](#_Toc28668)

[4.7.3 当然用户测试 45](#_Toc7459)

[G005小组项目访谈记录 45](#_Toc10519)

[5软件功能概述 46](#_Toc4659)

[5.1登录 46](#_Toc6758)

[5.2论坛功能 46](#_Toc14309)

[5.3动物城友会功能 46](#_Toc31509)

[5.4动植物科普功能 46](#_Toc25777)

[6总体分析 47](#_Toc15304)

[6.1功能分析 47](#_Toc12916)

[6.2存在的缺陷 47](#_Toc29578)

[6.3后续补救 47](#_Toc13862)

[6.4评价与结论 47](#_Toc15658)

[7注解 47](#_Toc23111)

[软件测试相关补充： 48](#_Toc20582)

[相关术语解释： 48](#_Toc6612)

[第三方测试：介于开发方和用户之间的组织测试。 49](#_Toc15772)

[附录 50](#_Toc1981)

[参考资料： 50](#_Toc10202)

# 1引言

## 1.1标识标题：《软件测试计划》

## 简称：STP

## 版本号：0.0.1

## 说明：本文档适用于基于微信小程序的城院生态圈，支持Office、WPS等word文档阅读软件

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 草稿  [ ] 正式发布  [ ] 正在修改 | 文件标识： | SE2021-G005-软件测试分析报告 |
| 当前版本： | 0.4 |
| 作者： | 黄依豪，梁晓勇，李东泽 |
| 完成日期： | 2021-12-26 |

## 1.2系统概述

本文档用于说明对于“城院生态圈”的软件测试计划。

本项目是通过小程序实现用户的交流论坛——“城院生态圈”，以爱护动植物、科普相关知识为目的，作为一个论坛平台供校内学生们交流动植物。力争做到界面清新、充实功能、无广告、无收费，操作简单易上手，吸引用户加入。本项目实现的小程序最终会在微信小程序的搜索栏下找到并运行。

开发历史：

2021/9/15-2021/9/23 课题选择

2021/9/23-2021/10/10 项目计划制定

2021/10/10-2021/10/17 可行性分析报告制定

2021/10/18-2021/10/24 需求分析报告制定

2021/11/3-2021/11/7 系统设计说明报告制定

2021/11/8-2021/11/13 软件测试计划制定

2021/11/8-2021/11/13 用户手册制定

2021/11/8-2021/11/14 详细计划制定

相关文档：

SE2021-G005-项目想法1.0.docx

SE2021-G005-项目计划表0.1.docx

SE2021-G005-可行性分析报告0.1.docx

SE2021-G005-需求分析报告1.0.docx

SE2021-G005-系统设计说明报告1.0.docx

SE2021-G005-用户手册1.0.docx

投资方、开发方：

G005小组

支持机构：

浙大城市学院

## 1.3文档概述

软件项目的测试计划是描述测试目的、范围、[方法](https://baike.baidu.com/item/%E6%96%B9%E6%B3%95/2444" \t "_blank)和[软件测试](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6%E6%B5%8B%E8%AF%95/327953" \t "_blank)的重点等的文档。对于验证[软件产品](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6%E4%BA%A7%E5%93%81/6800028" \t "_blank)的可接受程度编写测试计划文档是一种有用的方式。

本系统设计说明用于指导开发“城院生态圈”小程序项目顺利进行并最终通过评审的项目产品。本系统设计说明面向项目组全体成员。

## 1.4与其他计划的关系

是系统设计下的用于计划软件设计过程。

## 1.5基线

基线是软件生存期各个开发阶段的工作成果。

重要的检查点是里程碑，重要的需要客户确认的里程碑是基线。

主要基线：

SE2021-G005-项目想法1.0.docx

SE2021-G005-项目计划表0.1.docx

SE2021-G005-可行性分析报告1.1.docx

SE2021-G005-软件需求说明(SRS)1.1.docx

SE2021-G005-系统设计说明1.0.docx

# 2引用文件

本章应列出本文档引用的所有文档的编号、标题、修订版本和发行日期，也应标识不能通过正常的供货渠道获得的所有文档的来源。

[1]GB+T-8567-2006计算机软件文档编制规范 09 - 系统(子系统)设计(结构设计)说明(SSDD)

[2]SE2021-G005-项目想法1.0.docx

[3]SE2021-G005-项目计划表0.1.docx

[4]SE2021-G005-可行性分析报告0.1.docx

[5]SE2021-G005-需求分析报告1.0.docx

[6]SE2021-G005-系统设计说明报告1.0.docx

[7]SE2021-G005-用户手册1.0.docx

[8]张海藩，牟永敏.软件工程导论(第6版)[M]北京：清华大学出版社

[9]梁勇. JAVA语言程序设计(第10版)[M]北京：机械工业出版社

[10]王珊，萨师煊. 数据库系统概论(第5版)[M]北京：高等教育出版社

[11]覃征，李旭，王卫红. 软件体系结构(第4版)[M]北京：清华大学出版社

[12]李宁,李战怀.基于黑盒测试的软件测试策略研究与实践[J].计算机应用研究,2009,26(03):923-926.

[13]胡静.浅析黑盒测试与白盒测试[J].衡水学院学报,2008(01):30-32.

# 3测试概要

## 3.1测试环境

### 3.1.1时间与地点

**时间：**2020年12月12日

**地点：**求真1-625

### 3.1.1人员与设备

**人员：**黄依豪、李东泽、梁晓勇

**设备：**windows10 电脑，华为P40 pro，iphone 12promax

# 4测试分析

## 4.1黑盒测试

### 4.1.1功能测试

1. 主界面（帖子页面）



1. 私信界面



1. 发帖界面



1. 地图界面



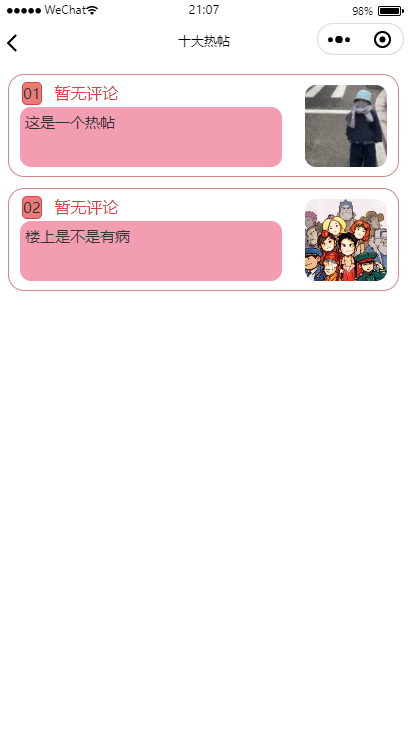
1. 我的



1. 帖子详情



1. 十大热帖



1. 动物城友会界面



1. 活动详情界面



1. 城友会发起活动



1. 城友会我加入的活动

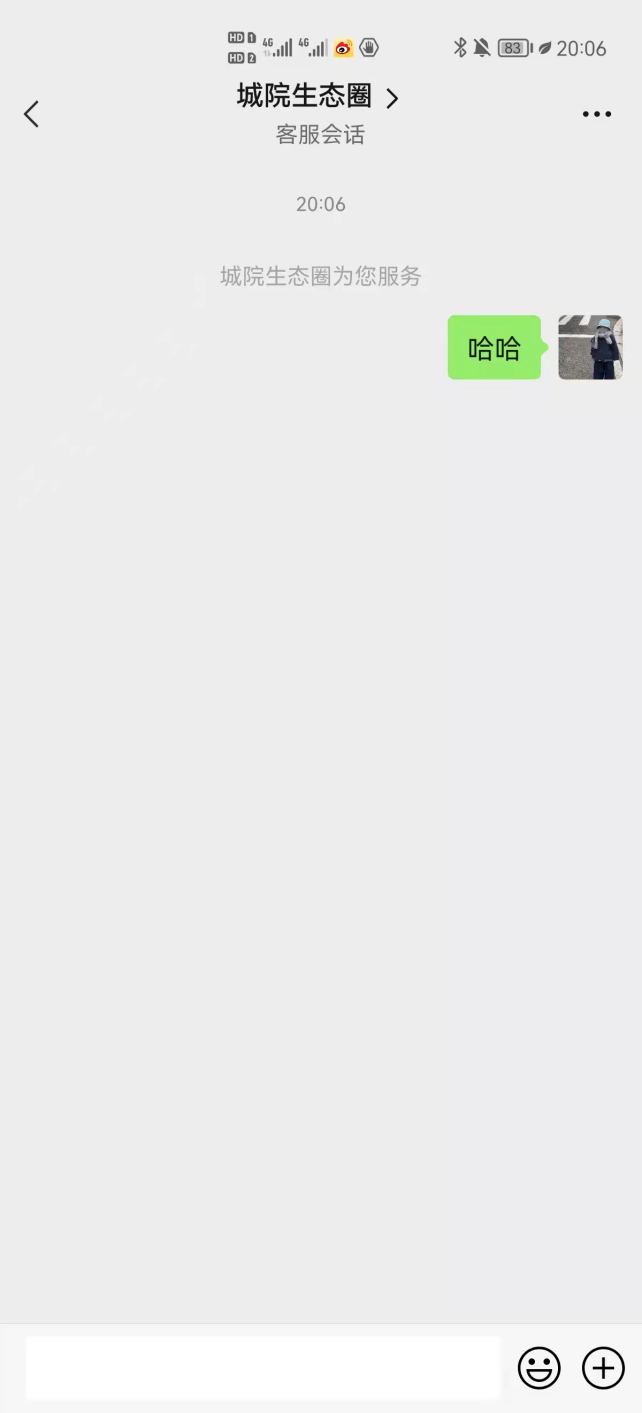


1. 动植物科普界面



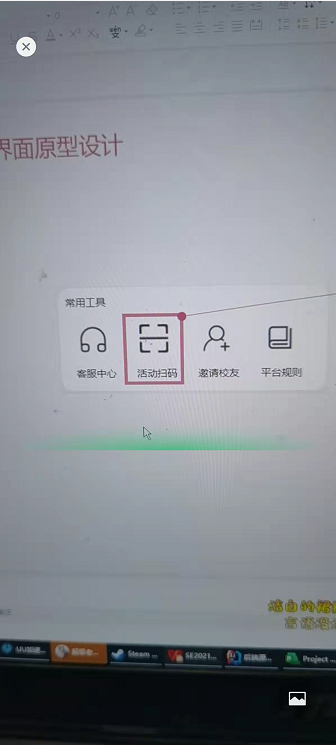
1. 客服中心



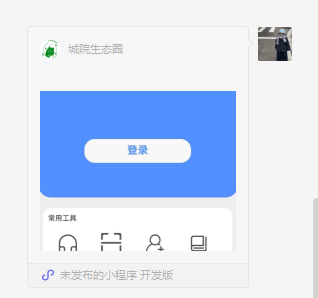


1. 活动扫码

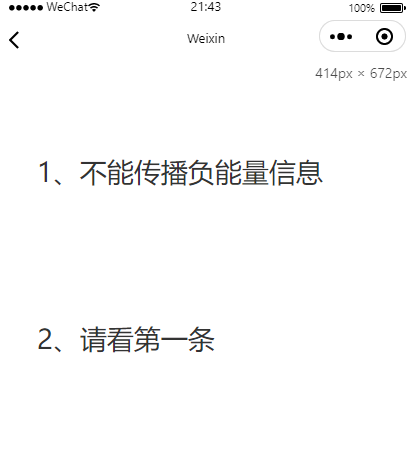




1. 邀请校友



1. 平台规则



### 4.1.2等价划分测试

1. 发帖界面测试

|  |  |
| --- | --- |
| 测试类型 | 填写主题、正文、图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 填写主题、正文，不添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 填写主题、不填写正文、不添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 不填写主题、填写正文、添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 填写主题、不填写正文、添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 不填写主题、不填写正文、添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符测试结果 | 是 |
| 测试类型 | 不填写主题、填写正文、不添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |
| 测试类型 | 不填主题、不填写正文、不添加图片 |
| 测试用例设计 |  |
| 测试结果图 |  |
| 测试结果与期望是否相符 | 是 |

### 4.1.3边界值分析测试

无，本项目没有边界值输入。

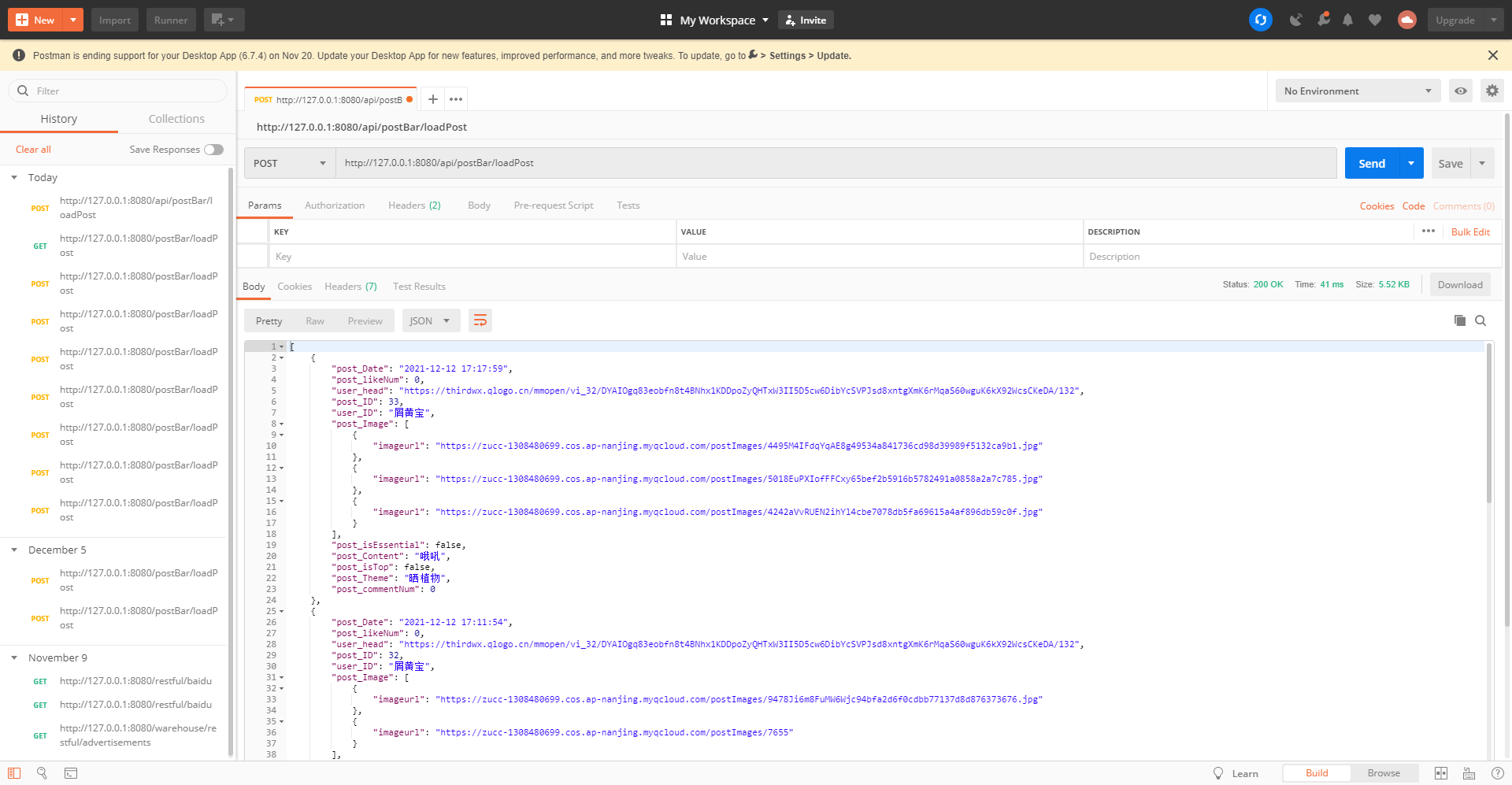
### 4.2白盒测试

**1.测试工具：**PostMan

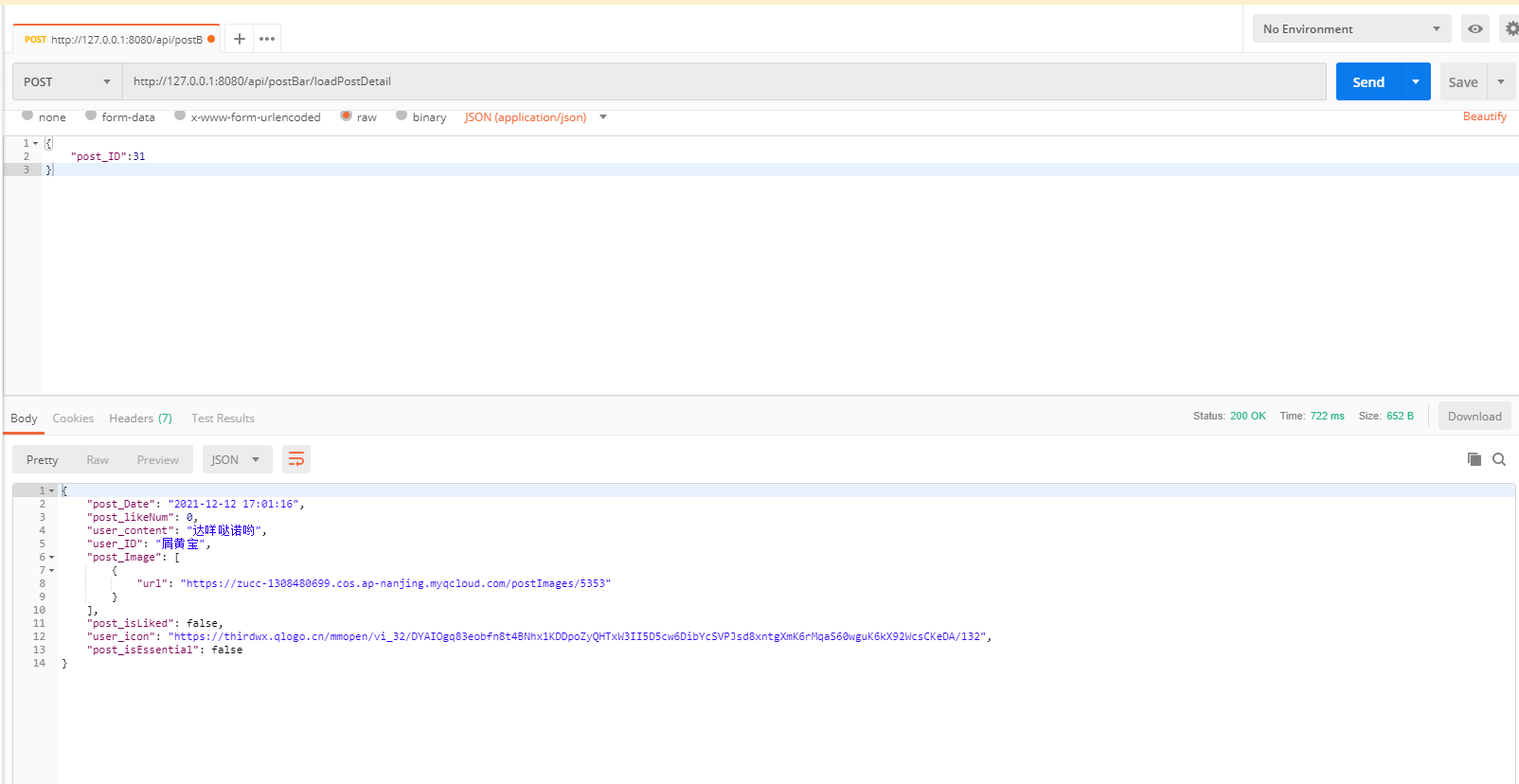
**2.测试方法：**基本路径测试

**3.测试截图：**

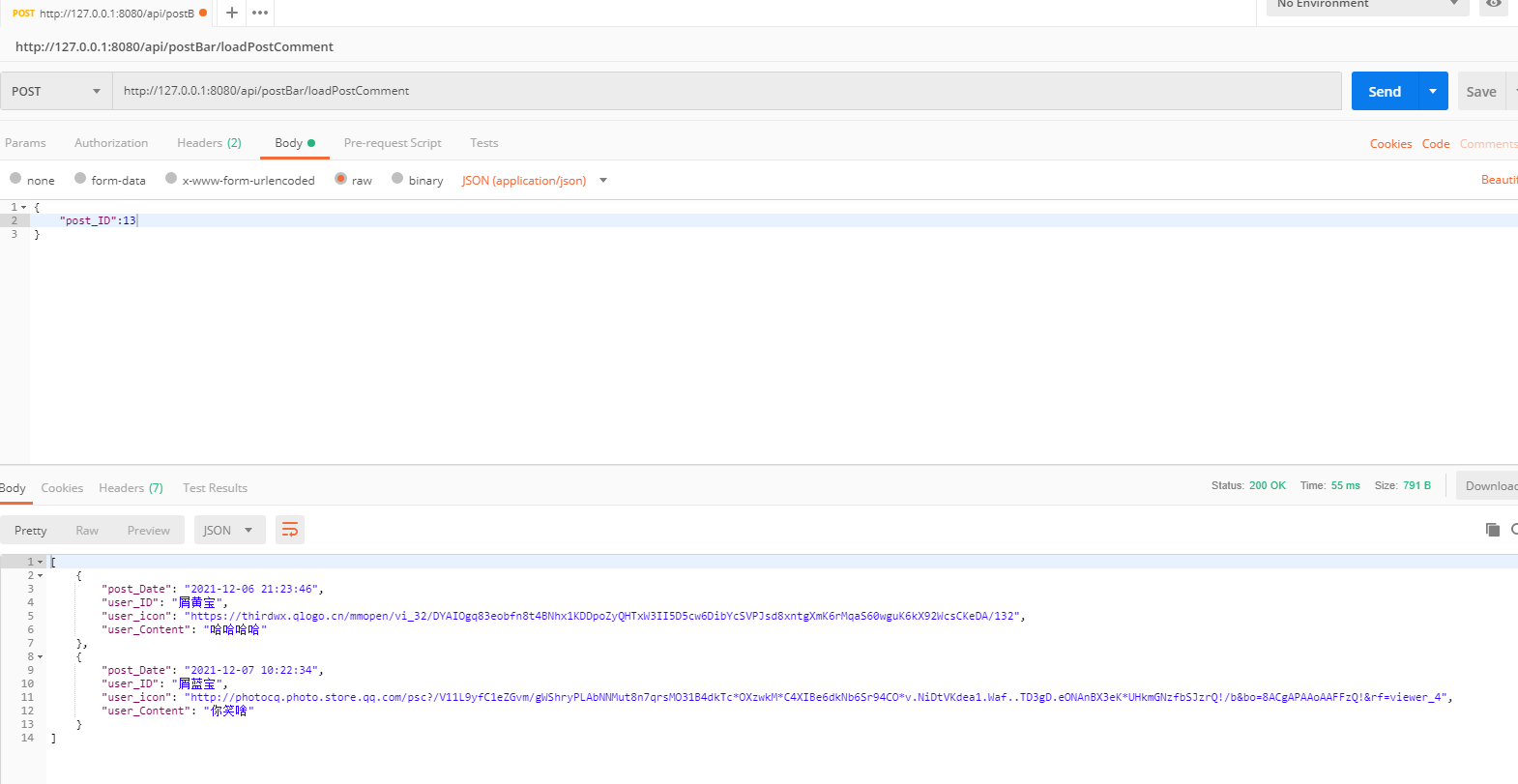
1. **论坛模块**
   1. **/api/postBar/loadPost**



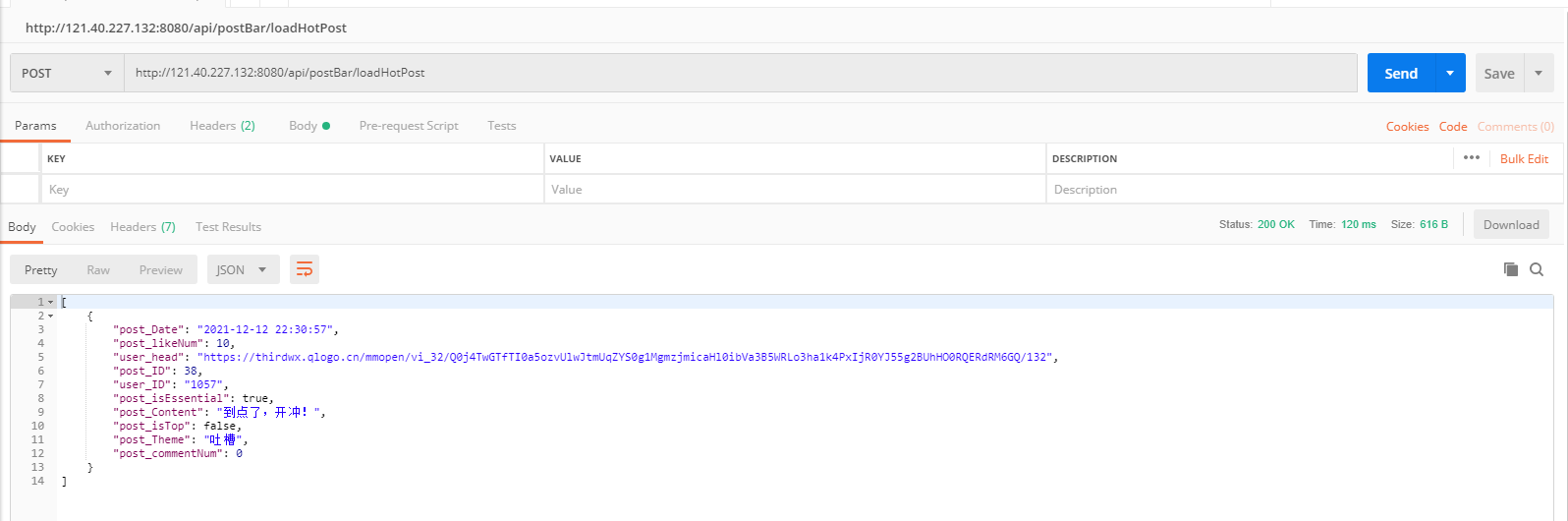
* 1. **/api/postBar/loadPostDetail**



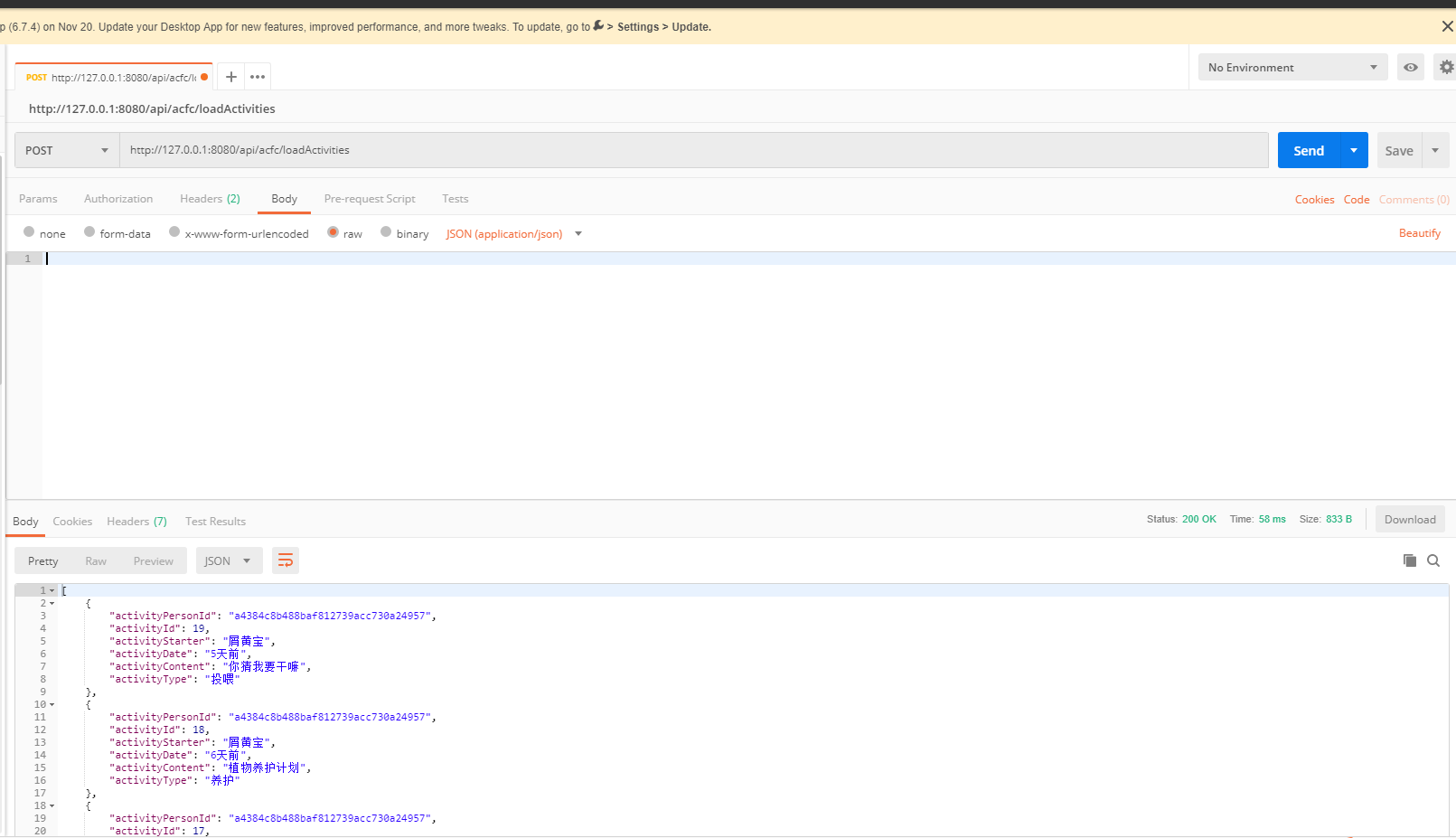
* 1. **/api/postBar/loadPostComment**



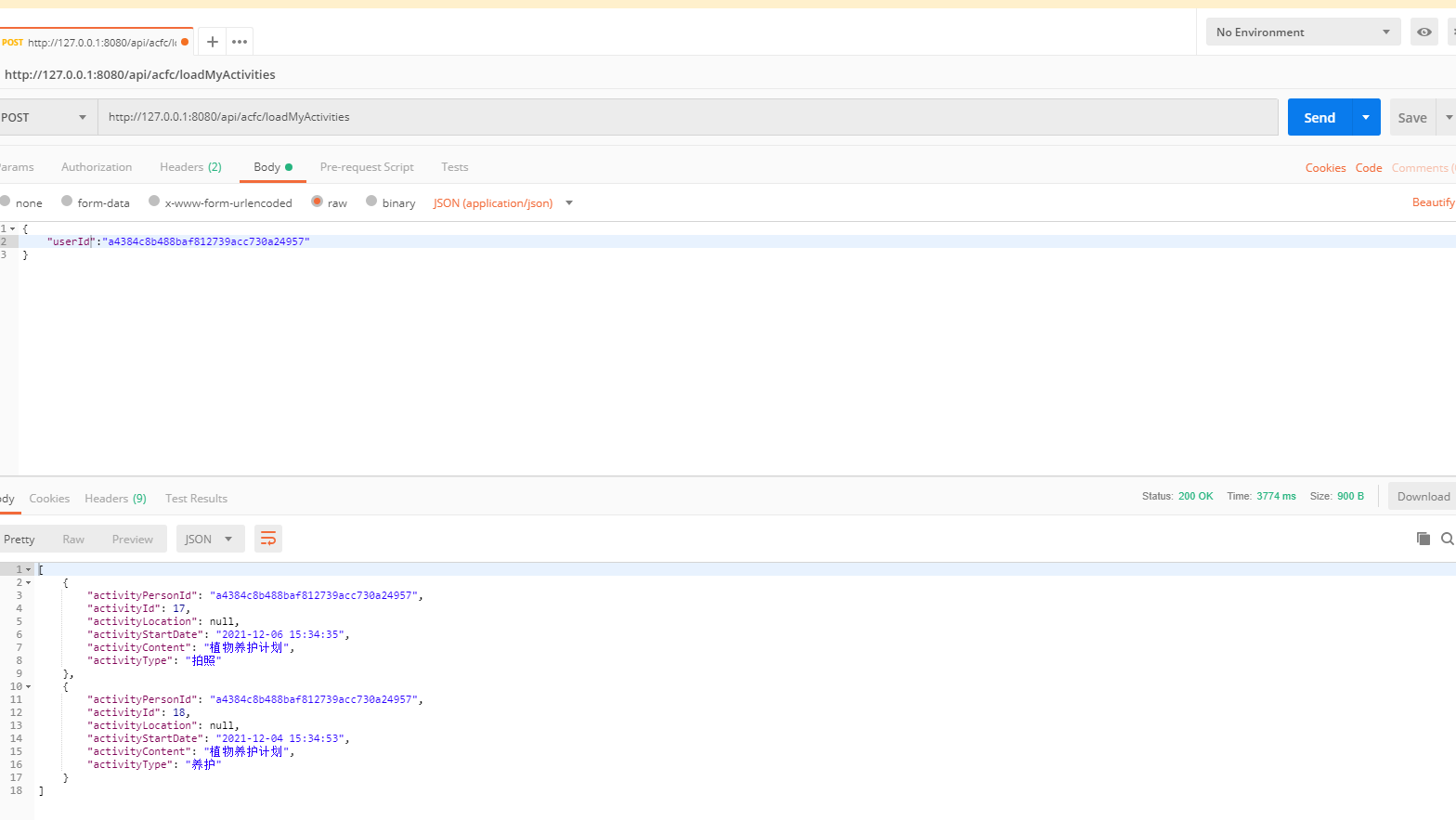
* 1. **/api/postBar/loadHotPost**



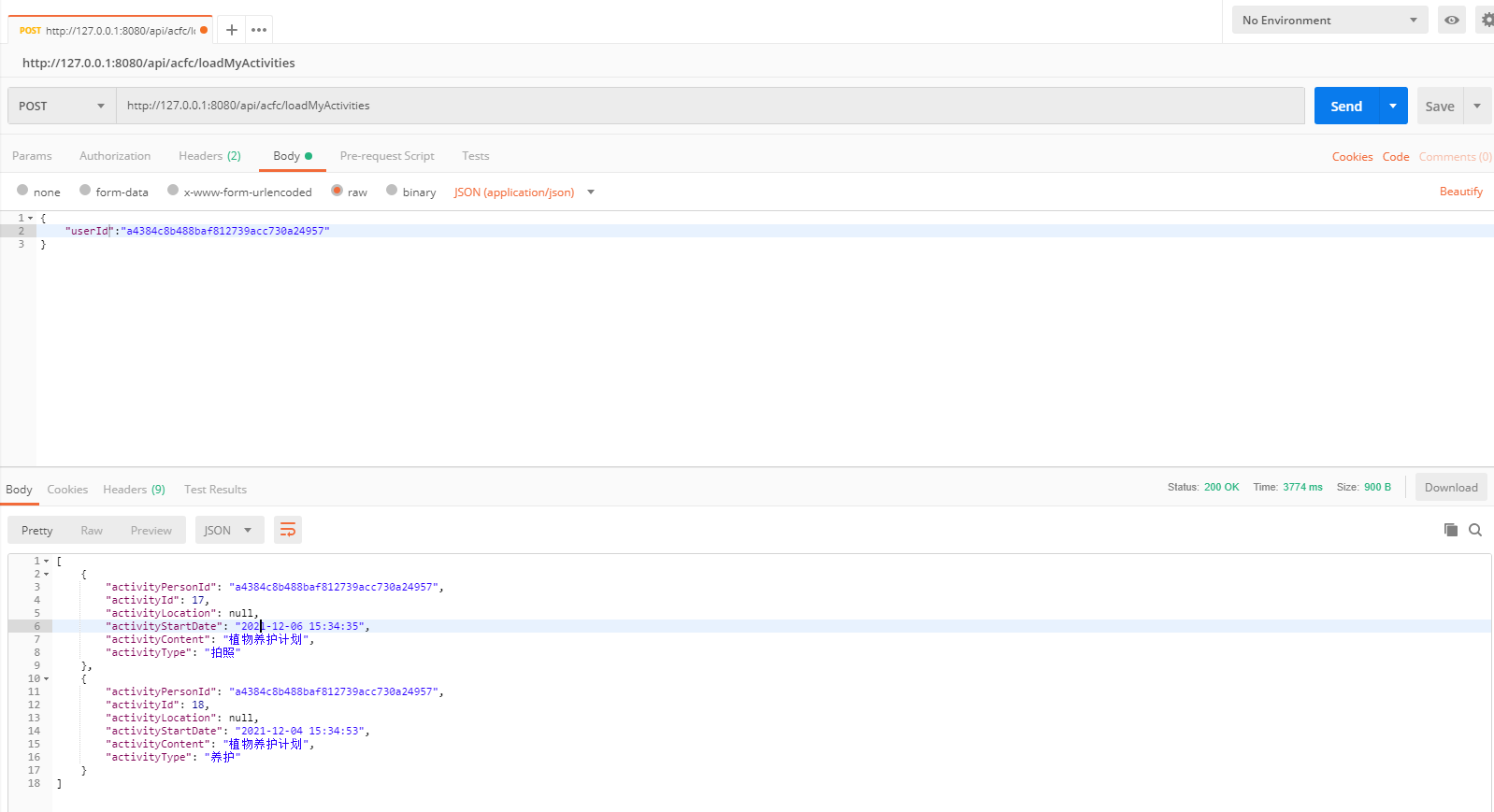
1. **动物城友会模块**
   1. **/api/acfc/loadActivities**



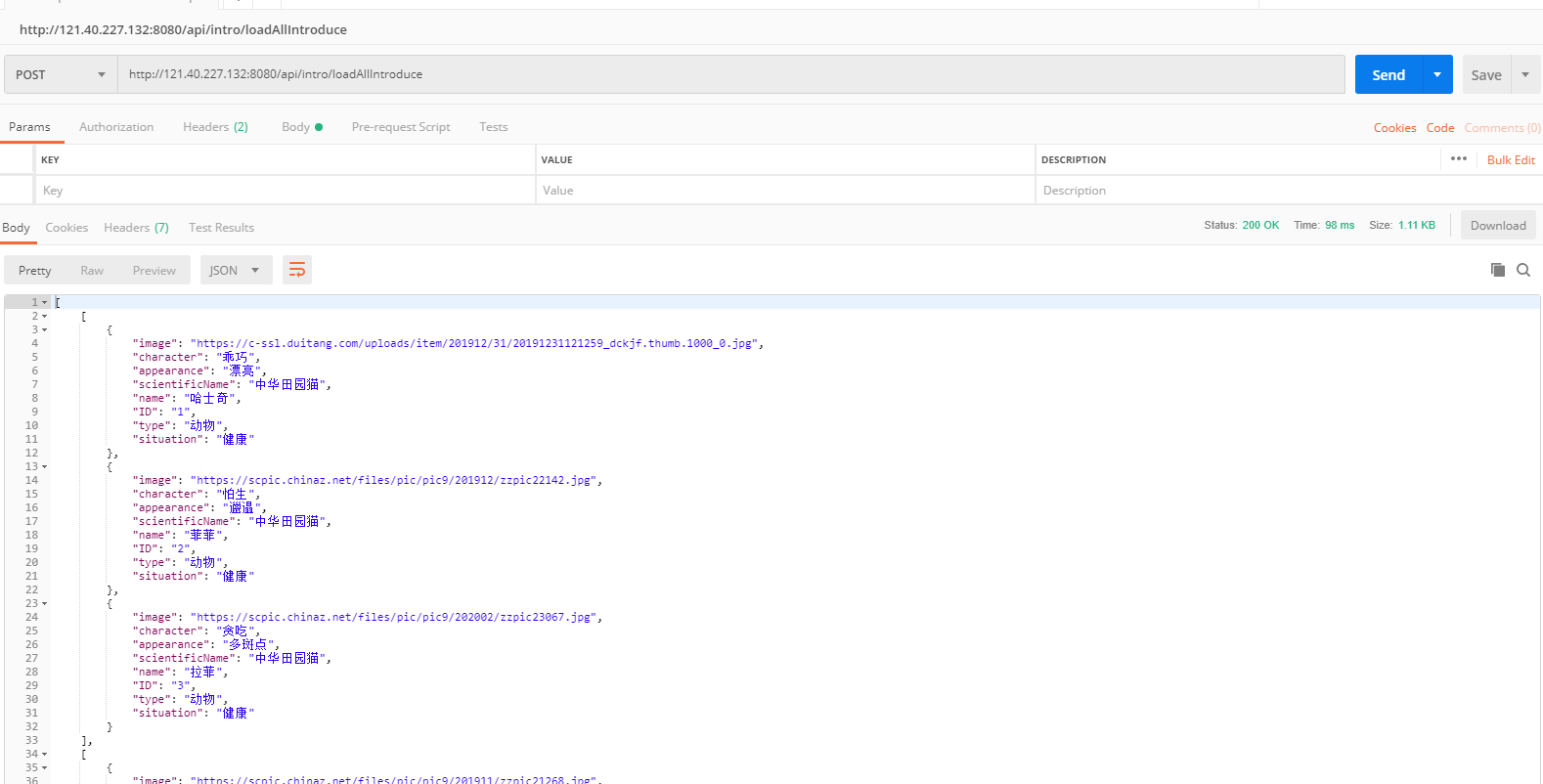
* 1. **/api/acfc/loadMyActivities**



* 1. **/api/acfc/loadActivityDetail**



1. **动植物科普模块**
   1. **/api/intro/loadAllIntroduce**



**3. 测试结果：**端口测试正常，所测试的端口都能返回相关信息。

## 4.3单元测试

### 4.3.1测试工具

1. **工具：Junit测试**
2. **测试环境：IDEA**

### 4.3.2测试用例

**设计原则：语句覆盖、路径覆盖**

## 3.1登录

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：黄依豪** | | | |
| **测试模块**：登录 | | | |
| **标识符**：1 | | | |
| **测试项**：   1. openID是否能成功获取 2. token是否能自动生成 3. 是否能存储用户信息 4. 是否能根据token拦截请求 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：   1. 点击“我的”进入个人信息界面 2. 点击登录，允许获取信息 | | | |
| **测试序号** | **输入数据** | **期望输出** | **测试是否通过** |
| 1 | 申请点击“允许” | 登录成功 | 通过 |
| 2 | 申请点击“拒绝” | 获取token但是不进行登录操作 | 通过 |
| 3 | 第一次使用小程序，申请点击“允许” | 数据库新增用户openID、头像url和昵称 | 通过 |
| 4 | 第一次使用小程序，申请点击“拒绝” | 不进行数据库信息添加操作 | 通过 |

## 3.2发帖

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：李东泽** | | | |
| **测试模块**：登录 | | | |
| **标识符**：2 | | | |
| **测试项**：   1. 输入数据的合法性 2. 点击发布是否能成功输入数据 3. 论坛能否实时更新帖子 4. 未登录能否发帖 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：  1、点击发帖进入发帖界面  2、选择主题、输入正文再点击发布  3、回到主页 | | | |
| **测试序号** | **输入数据** | **期望输出** | **测试是否通过** |
| 1 | 未登录输入数据进行发帖 | 提示“请登录” | 通过 |
| 2 | 登录后不输入数据进行发帖 | 提示“请输入完整” | 通过 |
| 3 | 登录后进行发帖 | 发帖成功，数据库更新帖子信息 | 通过 |
| 4 | 发帖后回到主页 | 看到刚发的帖子，按最近时间排序 | 通过 |
| 5 | 发纯文字帖 | 发出的帖子显示纯文字样式 | 通过 |
| 6 | 发带图片的帖子 | 发出的帖子是图片样式帖子 | 通过 |

## 3.3论坛模块

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：李东泽** | | | |
| **测试模块**：详细帖子界面展示 | | | |
| **标识符**：3 | | | |
| **测试项**：   1. 主页点进帖子界面能查看详细信息 2. 能查看评论 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：  1、点击主页，点进任意一个帖子 | | | |
| **测试序号** | **输入数据** | **期望输出** | **测试是否通过** |
| 1 | 无需输入 | 显示帖子的点赞数、评论数、评论内容、赞赏数 | 除赞赏数外，其他都通过 |
| 2 | 不登录进行点赞 | 提示登录 | 通过 |
| 3 | 登录进行点赞 | 点赞成功，图标亮起，数据库更新信息 | 通过 |
| 4 | 取消点赞 | 图标变灰，数据库更新信息 | 通过 |
| 5 | 不登录进行评论 | 提示登录 | 通过 |
| 6 | 登录进行评论 | 评论更新 | 通过 |

## 3.4动物城友会模块

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：梁晓勇** | | | |
| **测试模块**：动物城友会模块 | | | |
| **标识符**：4 | | | |
| **测试项**：   1. 点进界面是否能按时间顺序自动加载所有活动信息 2. 发起活动功能是否正常 3. 我加入的功能查看 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：   1. 点击动物城友会进入界面 2. 刷新查看最新活动 3. 点击发起活动输入内容 4. 点击发布 5. 点击我加入的活动查看内容 | | | |
| **测试序号** | **输入数据** | **期望输出** | **测试是否通过** |
| 1 | 无 | 自动加载活动并按照最近发布时间排序 | 通过 |
| 2 | 点击发起活动，不输入内容点击发布 | 提示“请输入完整” | 通过 |
| 3 | 输入内容点击发布 | 成功发布并更新数据到数据库 | 通过 |
| 4 | 点击我加入的活动 | 按照时间顺序显示加入过的活动 | 通过 |
| 5 | 在我加入的活动里进行筛选 | 依照筛选条件显示加入过的活动 | 通过，未做此功能 |
| 6 | 点活动详情显示 | 显示活动详情 |  |
| 7 | 点击立即参与 | 显示参加成功 | 不通过，未做此功能 |

## 3.5动植物科普模块

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：黄依豪** | | | |
| **测试模块**：动植物科普 | | | |
| **标识符**：5 | | | |
| **测试项**：   1. 动植物科普信息的来回切换 2. 动植物科普信息的正常显示 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：   1. 点击动植物科普按钮进入界面 2. 点击“动物”“植物”来回切换 3. 点击进入科普详细信息界面 | | | |
| **测试序号** | **输入数据** | **期望输出** | **测试是否通过** |
| 1 | 点击按钮进入界面 | 显示动物的科普条目 | 通过 |
| 2 | 点击植物 | 显示植物的科普条目 | 通过 |
| 3 | 点击动物科普信息 | 进入科普界面显示相关信息 | 通过，但页面显示有问题 |
| 4 | 点击植物科普信息 | 进入科普界面显示相关信息 | 通过，但页面显示有问题 |

## 3.6主页模块

|  |  |  |  |
| --- | --- | --- | --- |
| **测试者：黄依豪** | | | |
| **测试模块**：主页模块 | | | |
| **标识符**：6 | | | |
| **测试项**：   1. 主页查看 2. 私信查看 3. 发帖查看 4. 地图查看 5. 我的查看 6. 客服中心查看 7. 活动扫码查看 8. 邀请校友查看 9. 平台规则查看 10. 设置查看 11. 关于我们查看 12. 意见反馈查看 | | | |
| **测试环境**：小程序开发者工具、IDEA | | | |
| **前置条件**：  1、网络与服务器数据库等其他运行环境正常 | | | |
| **操作步骤**：   1. 点击主页 2. 点击私信 3. 点击发帖 4. 点击地图 5. 点击我的 6. 点击客服中心 7. 点击活动扫码 8. 点击邀请校友 9. 点击平台规则 10. 点击设置 11. 点击关于我们 12. 点击意见反馈 | | | |
| **测试序号** | **操作** | **期望输出** | **测试是否通过** |
| 1 | 点击主页 | 显示主页界面 | 通过 |
| 2 | 点击私信，切换通知和私信 | 显示私信界面，顺利切换通知和私信界面 | 通过 |
| 3 | 点击发帖 | 显示发帖界面 | 通过 |
| 4 | 点击地图 | 显示地图界面 | 通过 |
| 5 | 点击我的 | 显示我的界面 | 通过 |
| 6 | 点击客服中心 | 进入客服界面 | 不通过，开发者工具暂不支持打开客服会话，真机调试可以通过 |
| 7 | 点击活动扫码 | 打开扫码界面 | 通过 |
| 8 | 点击邀请校友 | 打开分享界面 | 通过 |
| 9 | 点击平台规则 | 打开规则界面 | 通过 |
| 10 | 点击设置 | 打开设置界面 | 通过 |
| 11 | 点击关于我们 | 显示关于我们的信息界面 | 通过 |

## 4.4集成测试

### 4.4.1测试工具

**1.概要：**

测试工具 Junit4

**2. 集成测试说明**

由于本项目结构的特殊性，我们并没有采用当今统一的服务架构，所以无法使用正常的方法对单元测试和集成测试进行区分。本项目的单元测试和集成测试将整合在一起，在一个类里执行。

1. **测试用例说明**

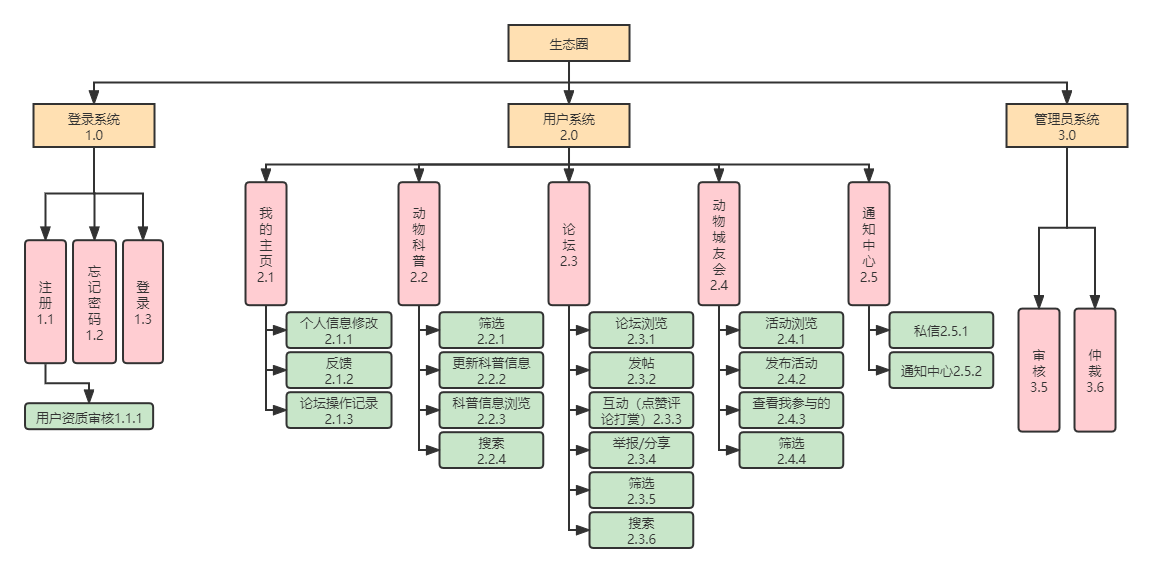
|  |  |
| --- | --- |
| 测试模块 | 动物城友会模块 |
| 测试代码 | package com.example.zuccbiosphere.controller;  import com.alibaba.fastjson.JSONObject;  import org.junit.jupiter.api.Test;  import org.junit.runner.RunWith;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;  import org.springframework.boot.test.context.SpringBootTest;  import org.springframework.http.MediaType;  import org.springframework.test.context.junit4.SpringRunner;  import org.springframework.test.web.servlet.MockMvc;  import org.springframework.test.web.servlet.MvcResult;  import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;  import org.springframework.test.web.servlet.result.MockMvcResultHandlers;  import org.springframework.test.web.servlet.result.MockMvcResultMatchers;  import static org.junit.jupiter.api.Assertions.\*;  /\*\*  \* @Author hyh  \* @Date 2021/12/27 20:31  \* @Version 1.0  \*/  @SpringBootTest()  @RunWith(SpringRunner.class)  @AutoConfigureMockMvc  class AnimalCityFriendsClubControllerTest {  @Autowired  private MockMvc mockMvc;    //模拟发起不同接口请求对整个功能进行测试  @Test  void participateActivity() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  data.put("activityId",21);  data.put("activityPersonOpenid","9311f91797025294c7c1b62142c7b37b");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/acfc/participateActivity")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void outActivity() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  data.put("activityId",21);  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/acfc/outActivity")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void loadActivities() throws Exception {  JSONObject data = new JSONObject();  data.put("openid","a4384c8b488baf812739acc730a24957");  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/acfc/loadActivities")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  @Test  void loadActivityDetail() throws Exception {  JSONObject data = new JSONObject();  data.put("userId","a4384c8b488baf812739acc730a24957");  data.put("activityId","18");  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/acfc/loadActivityDetail")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  @Test  void modifyLike() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  data.put("post\_ID",2);  data.put("post\_isLiked",true);  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/acfc/modifyLike")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  } |
| 测试结果 |  |

|  |  |
| --- | --- |
| 测试模块 | 论坛模块 |
| 测试代码 | package com.example.zuccbiosphere.controller;  import com.alibaba.fastjson.JSONObject;  import com.example.zuccbiosphere.ZuccBiosphereApplication;  import com.example.zuccbiosphere.dto.postInfoDto;  import org.junit.Before;  import org.junit.jupiter.api.Test;  import org.junit.runner.RunWith;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;  import org.springframework.boot.test.context.SpringBootTest;  import org.springframework.http.MediaType;  import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;  import org.springframework.test.context.web.WebAppConfiguration;  import org.springframework.test.web.servlet.MockMvc;  import org.springframework.test.web.servlet.MvcResult;  import org.springframework.test.web.servlet.RequestBuilder;  import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;  import org.springframework.test.web.servlet.result.MockMvcResultHandlers;  import org.springframework.test.web.servlet.result.MockMvcResultMatchers;  import org.springframework.test.web.servlet.setup.MockMvcBuilders;  import org.springframework.web.context.WebApplicationContext;  import static org.hamcrest.CoreMatchers.containsString;  import static org.hamcrest.CoreMatchers.not;  import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;  import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.post;  import static org.springframework.test.web.servlet.result.MockMvcResultHandlers.print;  import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.content;  import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.jsonPath;  import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.redirectedUrl;  import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;  import org.junit.Before;  import org.junit.runner.RunWith;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.test.context.SpringBootTest;  import org.springframework.test.context.junit4.SpringRunner;  import org.springframework.test.web.servlet.MockMvc;  import org.springframework.test.web.servlet.setup.MockMvcBuilders;  import javax.xml.ws.Action;  import static org.junit.jupiter.api.Assertions.\*;  @SpringBootTest()  @RunWith(SpringRunner.class)  @AutoConfigureMockMvc  class postBarControllerTest {    @Autowired  private MockMvc mockMvc;  //模拟发起不同接口请求对整个功能进行测试  @Test  void testloadPost(){  try{  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadPost")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void loadPostDetail() throws Exception {  JSONObject data = new JSONObject();  data.put("post\_ID","80");  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadPostDetail")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  @Test  void loadPostComment() throws Exception {  JSONObject data = new JSONObject();  data.put("post\_ID","80");  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadPostComment")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  @Test  void loadHotPost() throws Exception {  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadHotPost")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  @Test  void loadMyPost(){  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadMyPost")  .accept(MediaType.APPLICATION\_JSON)  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void modifyLike() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  data.put("post\_ID",100);  data.put("post\_isLiked",true);  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/modifyLike")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void pushComment() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  data.put("acceptID","a4384c8b488baf812739acc730a24957");  data.put("post\_ID",100);  data.put("comment","hahahhaha");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/pushComment")  .accept(MediaType.APPLICATION\_JSON)  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void loadMyRPRecord() {  JSONObject data = new JSONObject();  data.put("user\_ID","a4384c8b488baf812739acc730a24957");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/postBar/loadMyRPRecord")  .accept(MediaType.APPLICATION\_JSON)  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  /\*protected void sendRequest(String rurl,int waitSeconds) throws Exception{  DefaultHttpClient client = new DefaultHttpClient();  HttpGet httpget = new HttpGet(rurl);  HttpResponse response = client.execute(httpget);  client.getConnectionManager().shutdown();  Thread.sleep(waitSeconds\*1000);  }\*/  } |
| 测试结果 |  |

|  |  |
| --- | --- |
| 测试模块 | 动植物科普模块 |
| 测试代码 | package com.example.zuccbiosphere.controller;  import com.alibaba.fastjson.JSONObject;  import org.junit.jupiter.api.Test;  import org.junit.runner.RunWith;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;  import org.springframework.boot.test.context.SpringBootTest;  import org.springframework.http.MediaType;  import org.springframework.test.context.junit4.SpringRunner;  import org.springframework.test.web.servlet.MockMvc;  import org.springframework.test.web.servlet.MvcResult;  import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;  import org.springframework.test.web.servlet.result.MockMvcResultHandlers;  import org.springframework.test.web.servlet.result.MockMvcResultMatchers;  import static org.junit.jupiter.api.Assertions.\*;  /\*\*  \* @Author Administrator  \* @Date 2021/12/27 21:03  \* @Version 1.0  \*/  @SpringBootTest()  @RunWith(SpringRunner.class)  @AutoConfigureMockMvc  class IntroduceAnimalPlantTest {  @Autowired  private MockMvc mockMvc;  @Test  void loadAllIntroduce() throws Exception {  JSONObject data = new JSONObject();  System.out.println(data.toJSONString());  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/intro/loadAllIntroduce")  .accept(MediaType.APPLICATION\_JSON)  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }  } |
| 测试结果 |  |

|  |  |
| --- | --- |
| 测试模块 | 用户模块 |
| 测试代码 | package com.example.zuccbiosphere.controller;  import com.alibaba.fastjson.JSONObject;  import org.junit.jupiter.api.Test;  import org.junit.runner.RunWith;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;  import org.springframework.boot.test.context.SpringBootTest;  import org.springframework.http.MediaType;  import org.springframework.test.context.junit4.SpringRunner;  import org.springframework.test.web.servlet.MockMvc;  import org.springframework.test.web.servlet.MvcResult;  import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;  import org.springframework.test.web.servlet.result.MockMvcResultHandlers;  import static org.junit.jupiter.api.Assertions.\*;  /\*\*  \* @Author hyh  \* @Date 2021/12/27 21:07  \* @Version 1.0  \*/  @SpringBootTest()  @RunWith(SpringRunner.class)  @AutoConfigureMockMvc  class UserInfoControllerTest {  @Autowired  private MockMvc mockMvc;  @Test  void saveUserInfo() {  JSONObject data = new JSONObject();  data.put("openId","a4384c8b488baf812739acc730a24957");  data.put("userName","屑黄宝");  data.put("userAvatarUrl","https://thirdwx.qlogo.cn/mmopen/vi\_32/DYAIOgq83eobfn8t4BNhx1KDDpoZyQHTxW3II5D5cw6DibYcSVPJsd8xntgXmK6rMqaS60wguK6kX92WcsCKeDA/132");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/login/saveUserInfo")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void rpSend() {  JSONObject data = new JSONObject();  data.put("openId","a4384c8b488baf812739acc730a24957");  data.put("posterID","a4384c8b488baf812739acc730a24957");  data.put("point",20);  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/user/rpSend")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  @Test  void loadUserInfo() {  JSONObject data = new JSONObject();  data.put("openid","a4384c8b488baf812739acc730a24957");  try{  MvcResult mvcResult = mockMvc.perform(MockMvcRequestBuilders.post("/api/user/loadUserInfo")  .accept(MediaType.APPLICATION\_JSON)  .param("user\_ID","a4384c8b488baf812739acc730a24957")  .header("x-auth-token","eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2Mzk0Nzc4NzQsImlhdCI6MTYzOTQ3NDI3NCwidXNlcm5hbWUiOiJhNDM4NGM4YjQ4OGJhZjgxMjczOWFjYzczMGEyNDk1NyJ9.E8TpTA0L156QRzEYxNYB\_5Kf9-1tgaN5XvfEXg8THZ8")  .contentType(MediaType.APPLICATION\_JSON)  .content(data.toJSONString())  )  // .andExpect(MockMvcResultMatchers.status().isOk())  .andDo(MockMvcResultHandlers.print()).andReturn();  System.out.println("output" + mvcResult.getResponse().getContentAsString());  }catch(Exception e){  e.printStackTrace();  }  }  } |
| 测试结果 |  |

### 4.4.2结构分解



1.先对主控制模块进行测试

2.每次新加入一个模块后再次进行测试

3.在不断进行回归测试

4.重复以上步骤。

未发现各模块接口间存在的问题

### 4.4.3详细过程

**1. 顶层**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **参与测试**  **模块名称** | **操作方法** | **结果** |
| 1 | 登录 | 用户可以去个人信息查看页面点击登录进行登录，或者在主页面浏览时点击非游客功能时提示登录进行登录 | 测试通过 |

**2. 中间层**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **参与测试**  **模块名称** | **操作方法** | **结果** |
| 1 | 发帖模块 | 用户可以正常进行带图片和不带图片的发帖 | 测试通过 |
| 2 | 动态帖子加载模块 | 用户可以实时更新帖子的信息、点赞信息、评论信息 | 测试通过 |
| 3 | 私信模块 | 用户可以发送私信和查看通知 | 测试不通过 |
| 4 | 地图模块 | 用户可以查看校内地图以及动植物分布图，点击可以查看科普信息 | 测试不通过 |
| 5 | 我的模块 | 用户可以查看个人信息、发帖、评论、打赏记录，使用各种工具 | 测试通过 |
| 6 | 十大热帖模块 | 热帖可以自己滑动，点击按钮自动加载热帖 | 测试通过 |
| 7 | 动物城友会模块 | 可以查看活动详情，发布活动，查看我加入的活动 | 测试部分通过，发布活动模块有bug |
| 8 | 动植物科普模块 | 可以自动加载动植物信息 | 测试通过 |

**3. 底层**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **参与测试**  **模块名称** | **操作方法** | **结果** |
| 1 | 帖子增加模块 | 用户点击发帖更新到数据库 | 测试通过 |
| 2 | 点赞增加模块 | 用户点赞，信息更新到数据库 | 测试通过 |
| 3 | 赞赏增加模块 | 用户点击赞赏，信息更新到数据库 | 测试通过 |
| 4 | 评论增加模块 | 用户评论，信息更新到数据库 | 测试通过 |

## 4.5系统测试

**使用工具：**小程序自带调试器

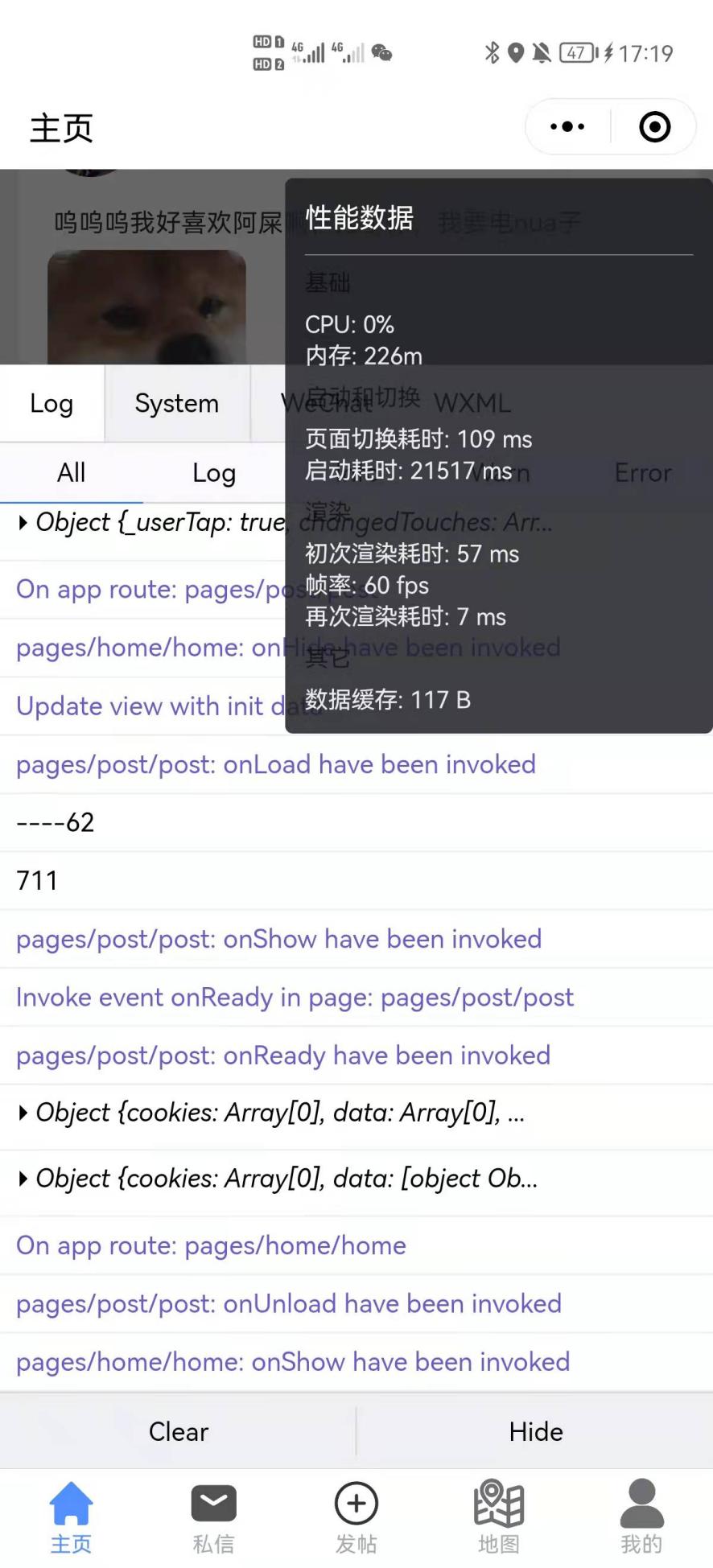
**测试方法：**

将小程序部署，用移动端打开小程序，打开调试，打开性能监控面板，使用各个功能，查看参数。

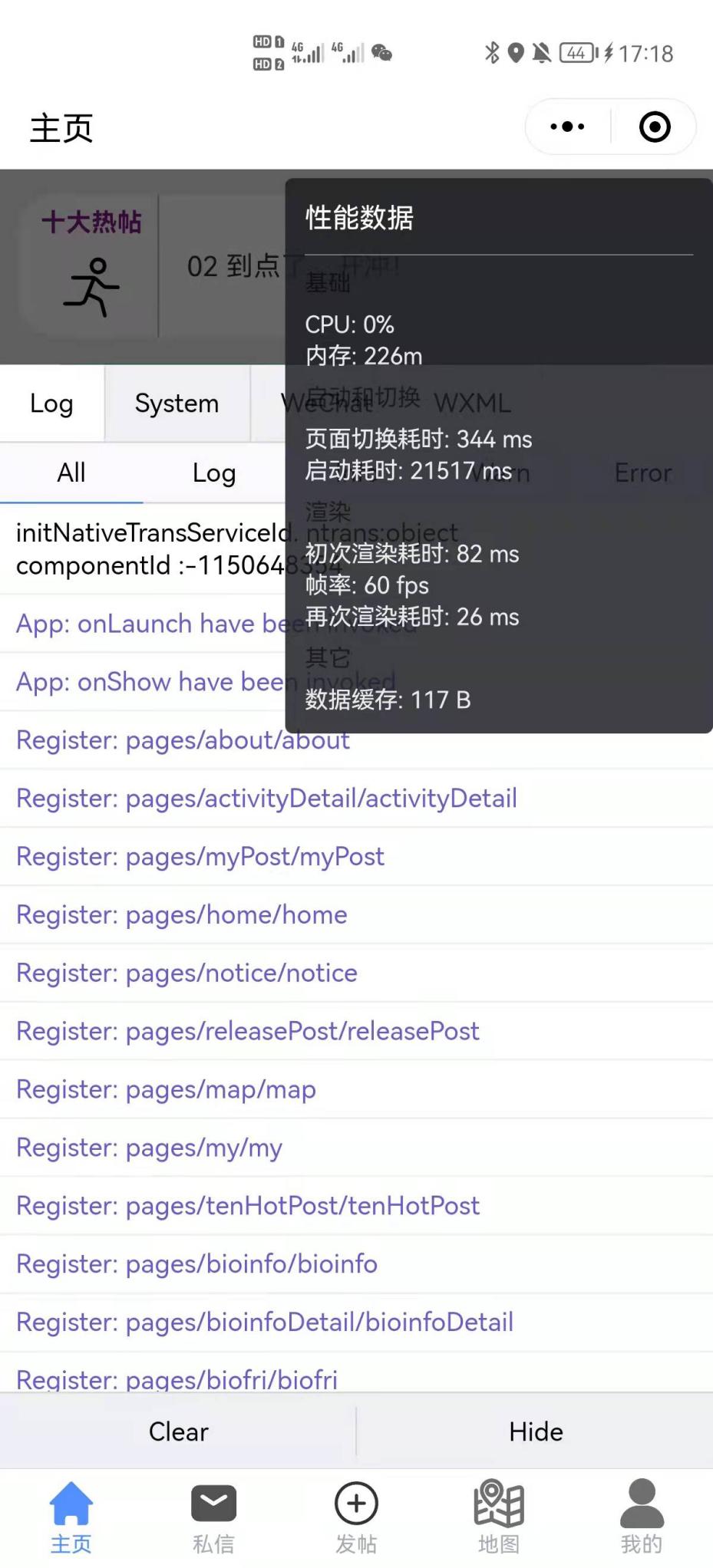
**测试结果：**功能基本能正常运行

**测试过程：**

1. **性能测试**







1. **功能测试**

**详情见本目录下[系统测试.mp4]**

## 4.7确认测试

### 4.7.1 Alpha测试

|  |  |  |  |
| --- | --- | --- | --- |
| 项目logoG005小组项目访谈记录 | | | |
| 访谈记录（2021-11-16） | | | |
| **一、基本信息** | | | |
| 访谈名称 | 典型用户功能确认访谈 | 被访谈人 | 侯永、黄浩、顾博宇 |
| 访谈日期 | 2021/12/27 | 访谈人 | 黄依豪 |
| 访谈地点 | 求真1-625 | 持续时间 | 约十分钟 |
| 记录人 | 黄依豪 | 审核人 | 全体成员 |
| **二、访谈结果汇总** | | | |
| 1. 界面UI需要美化，其他地方基本已经满足了需求（√） 2. 可以搜索帖子（√） 3. 地图功能美化一点（√） 4. 想要界面更加精美一点（√） 5. 服务器需要优化 （未完全优化） | | | |
| **三、访谈记录发放范围** | | | |
| 主送：杨枨老师  抄送：所有助教和全体小组成员 | | | |

### 4.7.2 当然用户测试

以下为当然用户（杨枨老师）测试的反馈记录表，区别于上述普通用户相关的测试。

|  |  |  |  |
| --- | --- | --- | --- |
| 项目logoG005小组项目访谈记录 | | | |
| 访谈记录（2021-11-16） | | | |
| **一、基本信息** | | | |
| 访谈名称 | 杨枨老师第二次访谈 | 被访谈人 | 杨枨老师 |
| 访谈日期 | 2021.12.28 17:00 | 访谈人 | 黄依豪 |
| 访谈地点 | 理四504 | 持续时间 | 05:17 |
| 记录人 | 黄依豪 | 审核人 | 全体成员 |
| **二、访谈结果汇总** | | | |
| 确认第一次访谈的需求完成情况  1.需要有自动登录提示，有弹窗界面（√）  2.还要设置精华帖和置顶帖（ X ）  3.回帖形式不够清晰，表明是这个回复是回复谁的（√）  4.地图上需要标明哪边有什么样的动物经常有，哪边哪种植物比较多（√）  5.发帖说明中说明只能发jpg,png等图片，说明能不能放文件和视频或者gif，给出发帖 例子（√）  6.需要有管理员界面（ X ） | | | |
| **三、访谈记录发放范围** | | | |
| 主送：杨枨  抄送：所有助教和全体小组成员 | | | |

# 5软件功能概述

## 5.1登录

1. 功能

获取用户信息存入数据库作为身份凭证，用于记录用户操作。若用户为第一次登录，则会自动生成唯一凭证openid存入数据库，无需注册

## 5.2论坛功能

1. 功能

用户可以进行自定义发帖，并且在论坛界面能够进行点赞、评论、打赏功能操作，可以查看帖子详情

## 5.3动物城友会功能

1. 功能

用户可以浏览正在进行的活动以及查看详情，可以加入活动，可以发起活动，并查看自己参加过的活动

## 5.4动植物科普功能

1. 功能

查看动植物的基本信息

# 6总体分析

## 6.1功能分析

针对论坛模块，我们提供的功能处于一个比较完善的状态，已经和正常的论坛发帖功能可以比了，但是细节方面例如交互有待优化。对于动物城友会，功能不是很完善，因为投入时间较短，有许多地方没有做完或者是有bug。对于动植物科普，这个功能相对比较简单，能实现基本的数据读取和显示，但因为我们新设计了用户也能增加科普数据，所以总体并未实现。

## 6.2存在的缺陷

1. 私信、地图功能等未完善
2. 动物城友会发起活动有bug
3. 若未登录，点击评论登录后迅速回复会显示null
4. 点赞后跳到详细页面图标不会亮
5. 不能评论图片
6. 动植物科普、动物城友会界面美化有待完善

## 6.3后续补救

1. 完善承诺过的功能
2. 优化界面
3. 尽量减少bug的存在，杜绝致命bug
4. 增加测试用户，获取完善的反馈

## 6.4评价与结论

该项目已初步达到预期目标，但功能仍等待完善。

# 7注解

**软件测试相关补充：**

在软件实现中包括编码和测试两个阶段。按照传统的软件工程方法学，编码是在对软件进行了总体设计和详细设计之后进行的，它只不过是把软件设计的结果翻译成用某种程序设计语言书写的程序，因此，程序的质量基本上取决于设计的质量。程序内部的良好文档资料，有规律的数据说明格式，简单清晰的语句构造和输人输出格式等，都对提高程序的可读性有很大作用，也在相当大的程度上改进了程序的可维护性。

目前软件测试仍然是保证软件可靠性的主要手段。测试阶段的根本任务是发现并改正软件中的错误，软件测试是软件开发过程中最艰巨最繁重的任务，大型软件的测试应该分阶段地进行，通常至少分为单元测试、集成测试和验收测试3个基本阶段。设计测试方案是测试阶段的关键技术问题，基本目标是选用最少量的高效测试数据，做到尽可能完善的测试，从而尽可能多地发现软件中的问题。

白盒测式和黑盒测试是软件测试的两类基本方法,这两类方法各有所长，相互补充。通常，在测试过程的早期阶段主要使用白盒方法，而在测试过程的后期阶段主要使用黑盒方法。为了设计出有效的测试方案。软件工程师应该深人理解并坚持运用关于软件测试的基本准则。设计白盒测试方案的技术主要有，逻辑覆盖和控制结构测试；设计黑盒测试方案的技术主要有，等价划分、边界值分析和错误推测。在测试过程中发现的软件错误必须及时改正，这就是调试的任务。为了改正错误，首先必须确定错误的准确位置，这是调试过程中最困难的工作，需要审慎周密的思考和推理。为了改正错误往往需要修正原来的设计，必须通盘考虑统筹兼顾，而不能“头疼医头、脚疼医脚”,应该尽量避免在调试过程中引进新错误。

测试和调试是软件测试阶段中的两个关系非常密切的过程，它们往往交替进行。程序中潜藏的错误的数目，直接决定了软件的可靠性。通过测试可以估算出程序中剩余的错误数。根据测试和调试过程中已经发现和改正的错误数，可以估算软件的平均无故障时间；反之，根据要求达到的软件平均无故障时间，可以估算出应该改正的错误数，从而能够判断测试阶段何时可以结束。

**相关术语解释：**

**单元测试**：对软件组成进行的测试。其目的是检验软件基本组成单位的正确性。测试对象是软件设计的最小单元：模块，又称为模块测试。

**集成测试**：也成联合测试，联调、组装测试，将程序模块采用适当的集成策略组装起来。

**系统测试**：将软件系统看成一个系统测试。包括对功能、性能以及软件所运行的硬软件环境进行测试。时间大部分在系统测试执行阶段，包括了回归测试和冒烟测试。

**验收测试**：部署软件之前的最后一个测试操作，它是技术测试室的最后一个阶段，也叫做交付测试，验收测试的目的是保证软件的准备就绪，按照项目合同、任务书、双方约定的验收依据文档，向软件的购买者展示该软件的原始的需求。

**α测试**：主要是由一个用户在开发环境进行的测试，也可以是公司内部的用户在模拟实际操作环境下进行的测试。主要的目的是：评价软件产品的FLURPS（即功能、局域化、可使用性、可靠性、性能和支持）。

**β测试**：由软件的最终的用户们在一个或者多个客户场所进行的测试。

**第三方测试**：介于开发方和用户之间的组织测试。

**手工测试**：是由人一个一个的输入测试用例，然后观察结果、和机器测试相对应，属于比较原始，大事需要一个一个步骤进行测试。

自动化测试：在预设条件下运行系统或应用程序，评估运行结果、预先条件应该包括正常的条件和异常条件。简单的说自动化测试是把人为驱动的测试行为转化为机器执行的一种过程。

**黑盒测试**（Black-box-Testing）：黑盒测试也称为功能测试，测试中把被测的软件当成一个黑盒子，不关心盒子的内部结构是什么，指关心软件的输入数据和输出数据。

**白盒测试**（White-box-Testing）：白盒测试又称结构测试，透明盒测试、逻辑驱动测试或基于代码的测试。白盒值的是打开的盒子，去研究里面的源代码和程序结果。

# 附录

**参考资料：**

[1] Flutter [EB/OL].https://flutterchina.club/ -2020/11/04

[2] Django 简介 [EB/OL]. https://www.runoob.com/django/django-intro -2020/12/05

[3]代码走查规范表[EB/OL]. https://blog.csdn.net/honsty2010/article/details/10987515 -2020/12/25

[4] 黑盒测试、白盒测试、单元测试、集成测试、系统测试、验收测试的区别与联系[EB/OL]. [https://www.cnblogs.com/xiaqiuchu/articles/10390777.html -2020/12/26](https://www.cnblogs.com/xiaqiuchu/articles/10390777.html%20-2020/12/26)

[5] 白盒测试----六种覆盖方法

[EB/OL]. https://blog.csdn.net/write6/article/details/78702977 -2020/12/26

[6] 单元测试到底是什么？应该怎么做？

[EB/OL]. https://www.zhihu.com/question/28729261 -2020/12/26

[7] 单元测试的基本测试方法[EB/OL]. http://www.51testing.com/html/49/n-222949.html -2020/12/26

[8] 黑盒测试 [EB/OL]. <https://baike.baidu.com/item/黑盒测试> -2020/12/26