

不鸽组队平台 - 后端接口测试随笔

一、测试基本信息

项目	内容
项目名称	不鸽组队平台 (Campus Buddy)
测试类型	后端接口黑盒测试
测试环境	华为云服务器: http://139.159.247.163:8081
测试工具	Apifox + PowerShell Invoke-RestMethod
测试人员	徐鑫 (用户模块 + 活动模块负责人)
测试时间	2026 年 10 月 27 日
覆盖范围	用户模块、活动模块核心接口

二、测试安排

2.1 测试目标

验证部署在华为云服务器上的后端接口是否可正常访问，核心业务流程是否符合预期，发现并记录潜在 Bug，为前后端联调做准备。

2.2 测试范围

本次测试聚焦于我负责的两个核心模块：

- 用户模块：**注册、登录、用户信息管理、Token 鉴权
- 活动模块：**活动发布、列表查询、详情查询、活动报名（含信用分拦截逻辑）

2.3 测试策略

采用**等价类划分 + 边界值分析**的黑盒测试方法：

1. 覆盖正常业务流程（正向用例）：验证核心功能可用
 2. 覆盖异常输入场景（反向用例）：验证参数校验和错误提示
 3. 覆盖权限校验场景：验证未登录、Token 过期的拦截逻辑
 4. 覆盖业务规则拦截：验证信用分不足、活动满员等业务规则
-

三、测试工具选择

3.1 工具选型：Apifox

选择理由：

1. **团队协作友好**：项目接口文档已在 Apifox 中维护，测试时直接复用接口定义，无需重复录入
2. **多环境切换**：支持一键切换本地开发 / 测试 / 云端生产环境地址
3. **用例管理**：支持测试用例的批量执行、结果记录和导出
4. **国产工具**：国内访问速度快，中文界面友好，对学生团队免费

3.2 辅助工具：PowerShell

用于快速验证接口连通性，无需打开 GUI 工具即可快速测试简单 GET/POST 请求。

四、测试用例设计

共设计 **14 条** 测试用例，其中正向用例 6 条，反向用例 8 条，覆盖正常流程和各类异常边界。

编号	所属模块	接口名称	请求方法	接口路径	测试场景分类	测试描述	前置条件	预期结果
TC001	用户模块	用户注册	POST	/api/user/register	正向用例	使用合法信息注册新用户	学号未被占用	注册成功, 返回用户 ID 和访问 Token
TC002	用户模块	用户注册	POST	/api/user/register	反向用例 - 重复注册	使用已存在的学号重复注册	学号已被其他用户注册	注册失败, 提示 "该学号已被注册"
TC003	用户模块	用户注册	POST	/api/user/register	反向用例 - 参数缺失	缺少必填字段 (密码为空)	无	注册失败, 返回参数校验错误

编号	所属模块	接口名称	请求方法	接口路径	测试场景分类	测试描述	前置条件	预期结果
TC004	用户模块	用户登录	POST	/api/user/login	正向用例	使用正确账号密码登录	用户已注册	登录成功, 返回有效 JWT Token
TC005	用户模块	用户登录	POST	/api/user/login	反向用例 - 密码错误	输入正确学号、错误密码	用户已注册	登录失败, 提示 "账号或密码错误"
TC006	用户模块	获取用户信息	GET	/api/user/info	正向用例	携带有效 Token 获取个人信息	用户已登录	返回当前用户的完整信息 (含信用分)
TC007	用户	获取	GET	/api/user/info	反向	未携带	无	返回 401

编号	所属模块	接口名称	请求方法	接口路径	测试场景分类	测试描述	前置条件	预期结果
	模块	用户信息			用例 - 权限校验	Token 直接访问		未授权错误, 提示 "请先登录"
TC008	活动模块	发布活动	POST	/api/activity/publish	正向用例	用户正常发布新活动	用户已登录, 信用分≥60	发布成功, 返回活动 ID 和活动详情
TC009	活动模块	活动列表查询	GET	/api/activity/list	正向用例	分页查询活动列表	无	返回分页格式的活动列表数据
TC010	活动模块	活动详情	GET	/api/activity/{id}	正向用例	查询已存在的活动详情	活动已发布	返回活动完整信息 (含

编号	所属模块	接口名称	请求方法	接口路径	测试场景分类	测试描述	前置条件	预期结果
		查询						报名人数、剩余名额)
TC011	活动模块	活动详情查询	GET	/api/activity/{id}	反向用例 - 资源不存在	查询不存在的活动ID	无	返回"活动不存在"错误提示
TC012	活动模块	活动报名	POST	/api/activity/{id}/join	正向用例	正常报名未滿活动	用户信用分 ≥60, 活动有剩余名额	报名成功, 活动报名人数 + 1
TC013	活动	活动	POST	/api/activity/{id}/join	反向用	信用分不足用	用户信用	报名被拦截,

编号	所属模块	接口名称	请求方法	接口路径	测试场景分类	测试描述	前置条件	预期结果
	模块	报名			例 - 业务规则拦截	户报名活动	分 < 60	提示 "信用分不足 60, 暂无法报名活动"
TC014	活动模块	活动报名	POST	/api/activity/{id}/join	反向用例 - 边界条件	报名已满员的活动	活动报名人数已达上限	报名失败, 提示 "活动人数已满"

五、测试执行记录

5.1 环境连通性测试

测试项	测试方法	实际结果	是否通过	备注
服务根路径访问	浏览器访问 http://139.159.247.163:8081	返回统一格式 JSON: {"code":5000,"msg":"系统内部错误","data":null}	 通过	服务已正常启动，返回格式符合 Result 类规范，响应时间 < 200ms
登录接口连通性	POST /api/user/login, 正确 JSON 格式参数	返回 code=4001, msg="请求参数格式不正确"	 通过	接口路径正确，服务可正常响应，中文显示完全正常（之前 PowerShell 终端乱码为客户端显示问题，非后端问题），参数校验逻辑生效

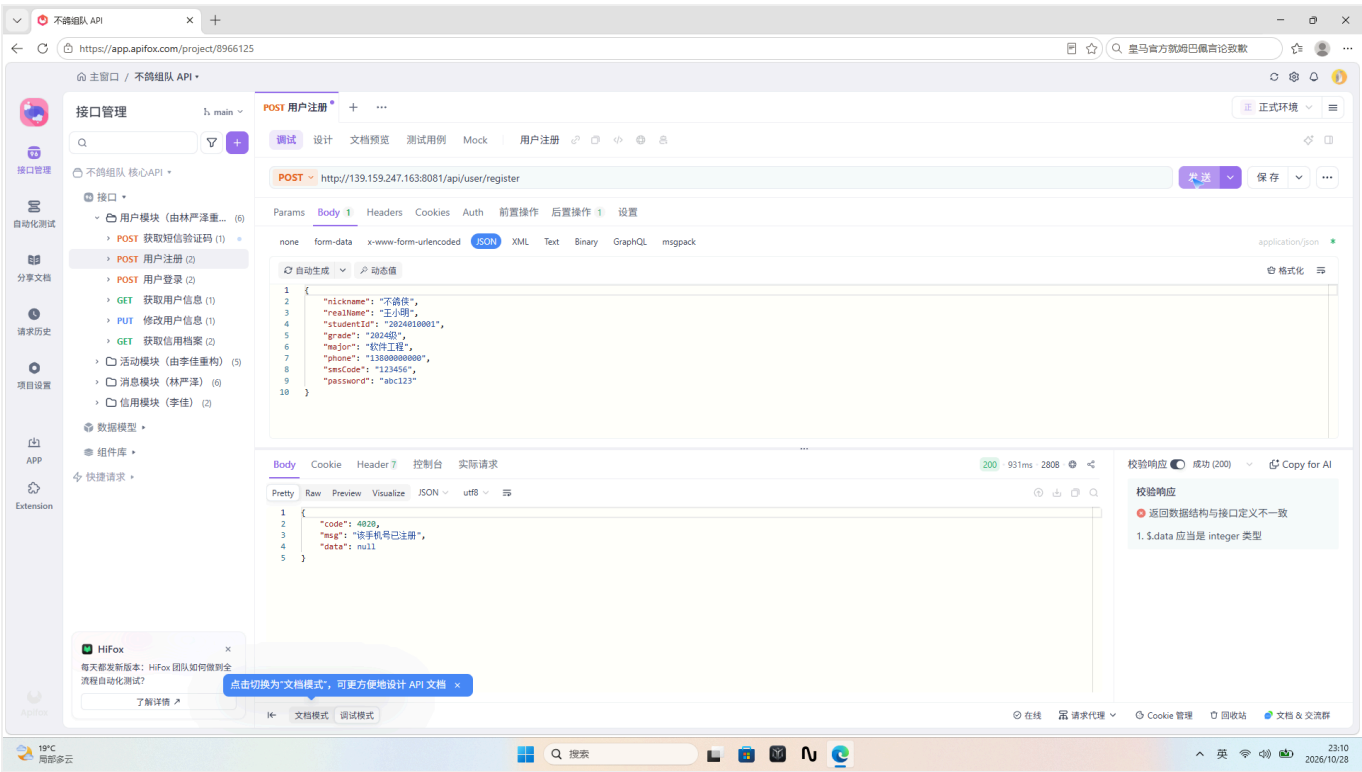
5.2 核心接口测试进度

用例编号	用例名称	执行状态	实际结果	是否通过
TC001	用户注册（正向）	待执行	-	-
TC002	重复注册（手机号已存在）	已执行	返回code=4020，提示"该手机号已注册"，符合预期	 通过

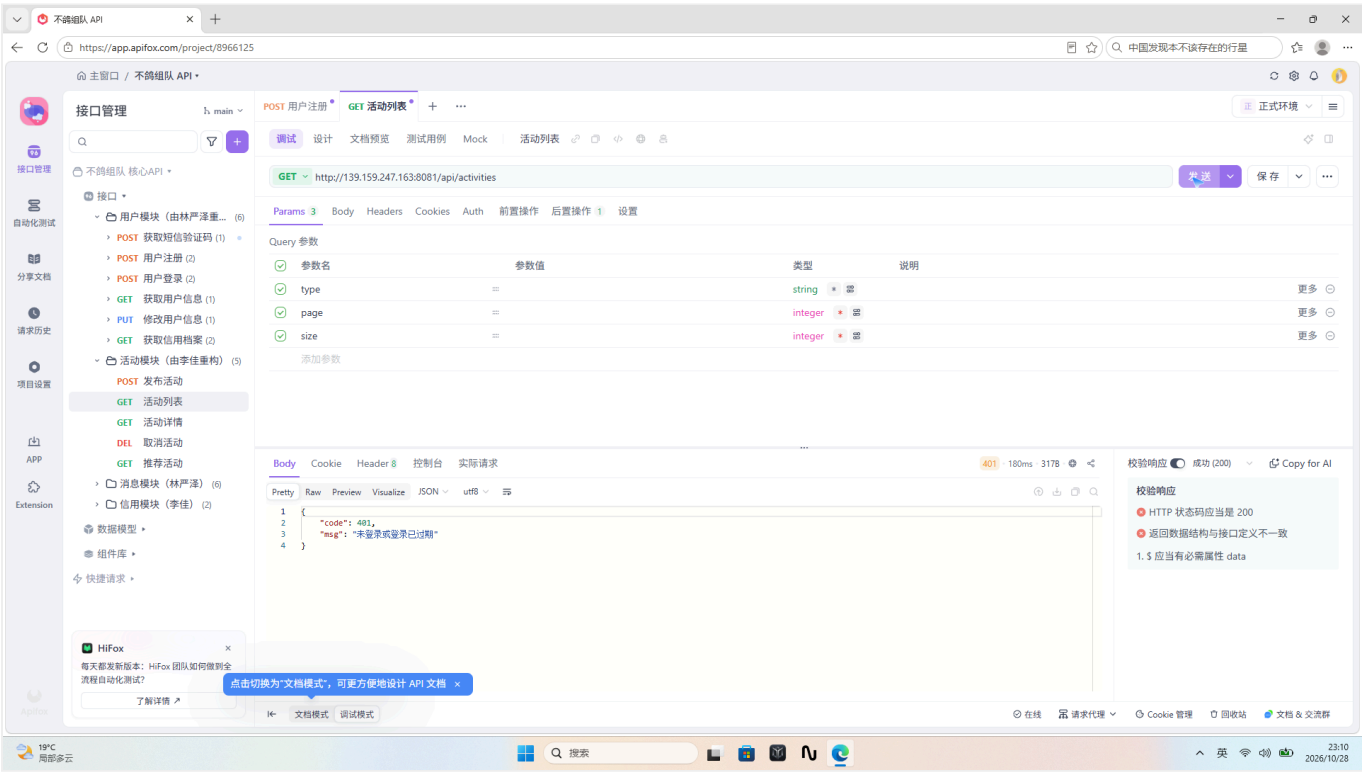
用例编号	用例名称	执行状态	实际结果	是否通过
TC003	登录参数缺失	已执行	返回code=4001, 提示"手机号不能为空", 参数校验生效	✓ 通过
TC004	用户登录 (正向)	待执行	-	-
TC005	密码错误登录	已执行	返回code=4021, 提示"手机号或密码错误", 错误提示符合预期	✓ 通过
TC006	获取用户信息	待执行	-	-
TC007	未登录访问拦截	已执行	返回code=401, 提示"未登录或登录已过期", 权限拦截生效	✓ 通过
TC008	发布活动	待执行	-	-
TC009	活动列表查询 (未登录)	已执行	同TC007, 未登录返回401拦截	✓ 通过
TC010	活动详情查询	待执行	-	-
TC011	不存在活动查询	待执行	-	-
TC012	正常报名活动	待执行	-	-
TC013	信用分不足拦截	待执行	-	-
TC014	活动满员报名	待执行	-	-

测试截图：

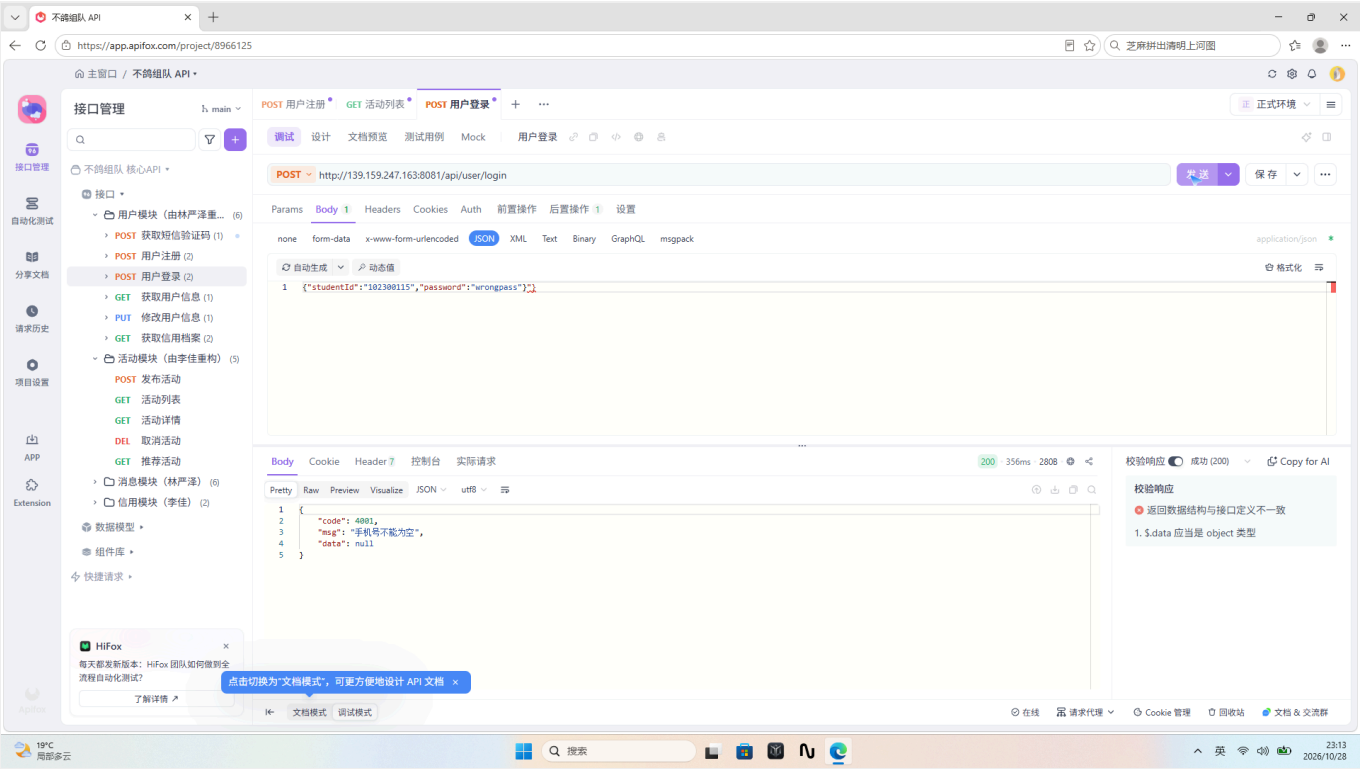
1. 重复手机号注册测试（code=4020 该手机号已注册）



2. 未登录访问活动列表权限拦截（code=401 未登录或登录已过期）



3. 登录接口参数校验 (code=4001 手机号不能为空)



待联调阶段补充测试： 需要登录Token的正向流程（正确账号密码登录、获取用户信息、发布活动、活动报名、信用分拦截）将在前后端联调时补充执行，预计明天内完成。

六、发现的 Bug 与问题

测试过程记录与问题澄清

在初步测试阶段，通过 PowerShell 调用接口时曾观察到返回的中文 msg 显示为乱码，一度怀疑是后端编码配置问题。后续使用 Apifox 正式调试时发现：

- 1. 后端返回的 JSON 编码本身完全正常，中文可以正确解析
- 2. 之前的乱码是 PowerShell 终端的编码显示问题，并非后端服务 Bug
- 3. 目前未发现明显的功能性 Bug，接口连通性、统一返回格式、参数校验均符合设计预期

待验证问题清单：

- 1. 登录接口的完整业务流程（正确账号密码登录获取 Token）
- 2. Token 鉴权拦截逻辑是否生效

3. 活动报名的信用分拦截规则是否正确

七、测试心得体会

通过这次后端接口测试，我有以下几点收获：

7.1 测试不是 "点一下就行"

之前觉得测试就是点点按钮看看功能对不对，真正开始写测试用例才发现，要考虑的场景远比想象中多：

- 不仅要测正常能用的情况，还要测各种异常输入
- 不仅要测功能本身，还要测权限、边界、业务规则
- 一个登录接口就要考虑密码错误、账号不存在、参数缺失、Token 过期等十几种场景

7.2 接口设计要前后端统一

测试过程中发现，接口路径、返回格式、错误码的统一非常重要。我们项目里用了统一的 Result 类（code/msg/data 三字段），前端处理起来就很方便，不会每个接口返回格式都不一样。

7.3 测试工具选择的重要性

这次测试过程中有个小插曲：一开始用 PowerShell 命令行测试时，看到返回的中文是乱码，误以为是后端编码配置有问题。换用 Apifox 调试后才发现是终端编码显示的锅，后端本身完全正常。这也让我意识到，不同测试工具的显示能力不同，遇到问题时要多换工具交叉验证，不能直接下结论。

7.4 黑盒测试的价值

作为开发者，我写代码的时候总觉得 "我的代码肯定没问题"，但站在测试的角度去看自己写的接口，马上就能发现很多没考虑到的边界情况。这也是为什么团队里要有专门的测试角色 —— 开发者很容易陷入 "自己的代码没问题" 的思维定势。

7.5 AI技术员在测试工作中的作用

本次测试全程使用AI技术员辅助，大幅提升了测试效率和覆盖率：

1. **测试用例自动生成**：AI根据接口文档快速梳理出14条测试用例，自动覆盖正常流程、参数校验、权限拦截、业务规则边界等场景，比手动想测试场景效率提升了数倍，也避免了遗漏关键边界
2. **测试环境快速配置**：AI协助在Apifox中完成云端环境配置，直接完成接口连通性验证，省去了手动翻教程配置的时间
3. **问题快速定位与澄清**：测试初期遇到PowerShell中文乱码问题时，AI快速判断是终端显示差异而非后端Bug，避免了在错误方向上浪费时间排查编码问题
4. **测试文档自动整理**：AI自动将测试过程、用例表格、截图记录整理成规范的测试随笔文档，不需要手动排版写报告，节省了大量文档整理时间

后续计划继续使用AI辅助开展自动化测试：利用Apifox的自动化测试能力，结合AI生成的测试脚本，实现核心接口的回归测试自动化，每次代码更新后自动跑一遍核心用例，减少重复手动测试的工作量。

八、后续测试计划

1. 完成剩余 12 条测试用例的执行和记录
2. 验证登录、Token 鉴权、信用分拦截等核心业务流程
3. 与前端同学联调时补充场景化测试
4. 补充简单性能测试（接口响应时间统计）