BETA SPRINT · DAY 3 · 测试岗交付

不鸽组队平台

Beta 冲刺 Day3 测试报告

并发报名压测、活动模块用例迁移与取消级联一致性校验

项目名称　不鸽组队 —— 校园搭子组队平台

团队名称　不鸽工程队

报告类型　Beta 冲刺日交付测试报告

被测版本　master 分支 e8bd103，2026-11-15 11:19 本地重新打包

编写人　王嘉萍（测试岗）

报告日期　2026 年 11 月 15 日

目 录

报告信息 1
一、执行摘要 1
二、测试环境与方法 2
2.1 版本重置与数据量 2
2.2 名额口径：为什么是 19 个坑 3
2.3 四项指标的算法 3
三、并发压测结果 4
3.1 场景一：100 并发抢 20 名额 4
3.2 场景二：500 并发抢 50 名额 4
3.3 同一账号并发重复点击 5
3.4 死锁排查与容量观察 5
四、活动模块用例迁移 TC008–TC014 6
4.1 编号映射与覆盖点 6
4.2 执行明细 7
五、取消活动数据一致性校验 7
六、全量回归结果 8
七、缺陷与问题归属 9
八、次日计划 10
九、提交记录与 AI 使用记录 10
附录：复现方式 11

报告信息

项目	内容
被测系统	不鸽组队平台后端（Spring Boot 3.2.5 + MySQL 8.0）
被测版本	master 分支 e8bd103（含取消级联、满员联动、未读计数修复）
测试环境	本机 http://127.0.0.1:8081，独立数据库 bugedui
测试数据	1595 个用户 / 660 个活动 / 6478 条报名记录
执行时间	2026-11-15 11:19 — 11:28
用例规模	22 条（环境自检 2 + 冒烟 8 + 查询与筛选 5 + 活动模块 7）
测试工具	pytest + requests 接口自动化、线程同步并发驱动、pymysql 库侧核对

说明：报告内全部数字取自 11 月 15 日上午对本机真后端（重新打包的 e8bd103）的实测记录，并发数据来自 reports/perf_summary_S1、S2 两份摘要，一致性数据来自 reports/cancel_consistency.json，未使用预估或模拟数据。

一、执行摘要

Day3 测试岗的三项任务全部完成：并发报名压测第一轮跑出两档场景、活动模块 7 条用例完成自动化迁移、取消活动的库侧一致性校验 8 项全部通过。

核心结论一　高并发下名额零越界

100 并发抢 19 个坑、500 并发抢 95 个坑，两档合计 600 次请求、114 次成功写入，**超卖 0、重复报名 0、死锁 0、失败率 0.00%**；库侧 current_people 与有效报名行数在 6 个活动上逐一对齐。

核心结论二　活动模块 7 条用例全部通过

TC008 – TC014 覆盖创建人数上下限、时间约束、详情 4040、取消越权 4030、取消级联与幂等、取消通知落库，实测 **7/7 通过**，未发现新缺陷。

核心结论三　取消级联"接口说的"与"库里发生的"一致

状态 4、名额回退 1、报名行 1→2 且保留、通知一人一条且不重发，8 项核对全 PASS，通知在接口返回后 1.0 秒内全部落库。

今日任务（05 清单）	交付物	状态
并发报名压测第一轮（100 / 500）	perf/concurrent_signup.py + 压测报告	完成，四项指标达标
活动模块用例自动化迁移	tests/test_03_activity.py（TC008 - TC014）	完成，7/7
取消活动数据一致性校验	scripts/check_cancel_consistency.py	完成，8/8
全量回归与当日报告	reports/Day3_测试报告_20261115_1125.md	完成，21 通过 + 1 xfail

二、测试环境与方法

2.1 版本重置与数据量

Day3 的被测版本必须是"当前主干"，否则测的是昨天的代码。因此先停掉占用 8081 的旧进程（jar 被运行中的进程锁定，不先停无法重新打包），再用当前分支（已 rebase 到 master e8bd103）执行 Maven 打包，启动时显式指定 --server.port=8081，日志重定向到 logs/day3-backend.log——死锁排查要 grep 这份日志，没有落盘就无法给出"死锁 0 条"的结论。

项目	数值
用户 / 活动 / 报名	1595 / 660 / 6478
活动状态分布	招募中 150、已满员 14、已结束 490、已取消 6
压测专用账号	17700000000 + i，共 500 个（注册即信用 100）
超卖核对	**0 行**

2.2 名额口径：为什么是 19 个坑

05 清单写的是"100 并发抢 20 名额、500 并发抢 50 名额"，但代码与建表约束决定了单活动装不下 50 个名额，这一条必须在报告里先说清楚，否则数字对不上会被当成测试造假。

事实	出处	对压测的影响
max_people 取值 2–20	ActivityCreateDTO 校验 + Service 判断	单活动最多 20 人
发起人占 1 个名额	createActivity 中 current_people 初始为 1	可报名坑位 = max_people − 1 = 19
满员置 status=2 后拒绝报名	tryIncreaseJoined 条件更新返回 0 行	超出的请求应返回 4010

因此执行方案为：场景一 100 个并发用户打 1 个 20 人上限活动（19 坑）；场景二把"50 名额"按 5 个 20 人上限活动拆开（95 坑），并发数保持 500，每个用户按 index % 5 落到其中一个活动。判定标准只有一条：**成功数必须恰好等于坑位数**，少一个是误拒，多一个就是超卖。

2.3 四项指标的算法

指标	算法	数据来源
超卖数	查库 current_people > max_people 的行数	pymysql 直连 bugedui 库
重复报名数	同一 user 在同一活动下状态为 1 的记录数大于 1 的分组数	pymysql 分组查询
失败率	（非预期业务码 + 连接失败）/ 总请求数；4010 满员、4011 重复、4012 信用不足属预期拒绝，不计入	每次请求的响应快照
死锁数	后端日志 grep Deadlock|Lock wait timeout 命中行数	logs/day3-backend.log

补充：延迟统计为端到端耗时（客户端发起至收到响应），含客户端排队成本，不等同于服务端处理耗时，报数时已在结论处注明。

三、并发压测结果

3.1 场景一：100 并发抢 20 名额

指标	数值
并发数 / 总请求	100 / 100
可报名坑位	19（20 人上限含发起人）
成功报名	**19**（等于坑位数）
4010 人数已满	81
4011 重复报名 / 4012 信用不足	0 / 0
失败率（异常失败）	**0.00%**
延迟 p95 / 平均 / 最大	129.4ms / 97.4ms / 136.2ms
百请求总耗时	0.18s
超卖 / 重复 / 人数不一致	0 / 0 / 0
死锁 / 锁等待超时	0

库侧现场：活动 646 的 current_people=20、max_people=20、status=2（已满员）、有效报名 19 条。20 = 19 报名 + 1 发起人，正好卡在上限，没有越界。

3.2 场景二：500 并发抢 50 名额（拆 5 个活动）

指标	数值
并发数 / 总请求	500 / 500
可报名坑位	95（5 × 19）
成功报名	**95**（等于坑位数）
4010 人数已满	405
4011 重复报名 / 4012 信用不足	0 / 0
失败率（异常失败）	**0.00%**
延迟 p95 / 平均 / 最大	578.0ms / 390.4ms / 643.1ms

指标	数值
500 请求总耗时	0.87s
超卖 / 重复 / 人数不一致	0 / 0 / 0
死锁 / 锁等待超时	0

五个活动（648 – 652）全部停在 current_people=20、status=2、有效报名 19，说明行锁 + 条件更新在跨活动竞争下同样没有漏更新：每个活动各自串行化，互不干扰。

3.3 同一账号并发重复点击

上面两档用的是不同账号，天生不会重复报名，"重复报名 = 0"并不足以证明幂等。因此每档额外做一次同一账号 10 次并发点击，模拟前端连点：

场景	成功	4011 重复	结论
S1 幂等抽查（活动 647）	1	9	通过
S2 幂等抽查（活动 653）	1	9	通过

10 个请求在同一毫秒打到同一活动、同一用户，只有 1 条落库，其余 9 条被唯一约束与防重快查拒绝，**没有出现同一人占 2 个坑**。

3.4 死锁排查与容量观察

死锁从两个方向确认：后端日志 grep 命中 **0 条**；压测后 SHOW ENGINE INNODB STATUS 输出中没有 LATEST DETECTED DEADLOCK 段落（若发生死锁，MySQL 会把两个事务的持锁与等锁现场完整打在这段里）。结构上的原因是加锁顺序统一：报名与退出都是"先锁活动行（条件更新）再写报名行"，不存在交叉等待。

场景二的 p95 为 578ms，比场景一高 4.5 倍，但错误率仍是 0、超卖仍是 0，说明瓶颈不在业务锁，而在排队：

环节	数值	影响
客户端	500 线程同时建连	本机 CPU 抢占，客户端侧排队
Tomcat	默认 maxThreads=200	约 300 个请求在 accept 队列等待
数据库	同活动行锁串行化	5 条并行队列，每队 100 个请求

结论：**报名接口的吞吐瓶颈在 Tomcat 线程池与单机资源，不在业务逻辑**。若 Day6 要把 QPS 做上去，优先调 server.tomcat.threads.max 并做连接池预热，而不是改报名流程。这也是"错误率 0 但 p95 变高"的典型特征，可用于回答答辩中"你们的压测结果怎么解读"。

四、活动模块用例迁移 TC008 – TC014

4.1 编号映射与覆盖点

α 阶段这 7 条是手工用例，本次按同一覆盖点逐条自动化，落在 tests/test_03_activity.py。全部用例自建活动、互不污染，不写死任何自增 ID，活动 ID 与报名 ID 一律取自接口返回。

编号	覆盖点	预期
TC008	创建活动-人数下限	4001
TC009	创建活动-人数上限（20 合法 / 21 拒绝）	成功 / 4001
TC010	创建活动-deadline 晚于 startTime	4001
TC011	活动详情-不存在的 ID	4040
TC012	取消活动-非发起人越权	4030
TC013	取消活动-级联退名额 + 重复取消幂等	status→4、人数→1
TC014	取消活动-每位报名者 1 条 type=4 通知	1 条 / 人，不重发

编号说明：α 手工用例表只给了 TC008 – TC014 的编号区间，未把逐条文字带进仓库；本表按"活动模块 7 个可验证功能点"落地，覆盖点与手工用例一致。若与 α 原表有出入，以文件头的映射表为准，已同步进 README 排期章节。

4.2 执行明细

编号	实测结果	耗时	结论
TC008	maxPeople=1 返回 code=4001	3ms	通过
TC009	20 返回活动 ID；21 返回 code=4001	8ms	通过
TC010	deadline=start+1h 返回 code=4001	3ms	通过
TC011	不存在 ID 返回 code=4040	3ms	通过
TC012	路人删除返回 code=4030，活动状态仍为 1	140ms	通过
TC013	取消后 status=4、人数=1、再报名 4013、重复取消仍成功	284ms	通过
TC014	报名者收到 1 条未读 type=4 通知，重复取消后仍为 1 条	—	通过

TC013 的重点是"接口说取消成功"还不够，必须同时满足三条：状态变 4、名额回退到 1（只剩发起人）、报名者再报名返回 4013 而不是 4040——后者说明活动没有被逻辑删除，报名者从历史通知点进来仍能看到"已取消"，而不是一个不存在的页面。

五、取消活动数据一致性校验

取消活动最容易出的问题是"接口返回成功，库里没退干净"。本节不只信接口返回值，而是直连数据库逐项核对。

```
=== 取消前 ===
活动 655：status=2 current_people=4 max_people=4
报名记录：已报名 3 条 / 已退出 0 条
取消通知（type=4）：0 条

=== 取消后（接口返回后 1.0s 落库完成）===
活动 655：status=4 current_people=1 max_people=4
报名记录：已报名 0 条 / 已退出 3 条（总 3 条，行未被删除）
取消通知（type=4）：3 条，按用户：{1578: 1, 1579: 1, 1580: 1}
```

#	核对项	结果	实际值
1	活动状态 → 4（已取消），不做逻辑删除	PASS	4
2	名额回退 → 1（只剩发起人），不泄漏	PASS	1
3	报名记录全部置 2（已退出，不写 4）	PASS	1:0 / 2:3
4	报名行保留，不做物理/逻辑删除	PASS	3 行
5	通知每人恰好 1 条	PASS	3 条
6	通知未误发给发起人	PASS	0
7	重复取消幂等（四项数值均不变）	PASS	通知 3 / 已报名 0
8	取消后名额无泄漏（他人再报名 → 4013）	PASS	4013

两点值得写进答辩问答：一是"名额回退"与"不允许再报名"并不矛盾——回退是数据一致性要求，而活动状态已是 4，任何新报名都必须被状态机拦住，否则会出现"已取消的活动还能报名"的严重缺陷；二是报名记录刻意写 2（已退出）而非 4，避免被信用模块误判成爽约再扣一次分。

六、全量回归结果

项目	结果
执行条数	22
通过 / 失败 / 跳过	21 / 0 / 0

项目	结果
预期失败（xfail）	1（BUG-2 分页唯一性，挂账中）
总耗时	5.2s
新增缺陷	0

xfail 一条是 Day1 挂账的 BUG-2（列表 ORDER BY created_at 无唯一兜底键，同秒并列行翻页重复/漏项），李佳合入修复后该条会自动转 PASS，作为回归验收点保留。按"通过 /（通过 + 失败）"计算通过率为 100%，报告中 95.5% 是因为把 xfail 计入了分母。

七、缺陷与问题归属

编号	描述	级别	状态	归属
BUG-1	满员后退出状态不回落	P1	已关闭	活动模块 — 李佳
BUG-2	分页排序键并列导致重复/漏项	P2	xfail 挂账	活动列表 — 李佳
BUG-3	取消接口注释与实现相反	P3	已关闭	文档 — 林严泽

Day3 新增用例未发现新缺陷。但过程中出现两次"看着像缺陷"的现象，经排查都是测试侧问题，值得记录：

现象	真实原因	处理
压测错误率 10%，10 条 4012 信用不足	造数账号 1/3 信用分只有 50（Day1 造数的 5000 条报名对应活动到期后按爽约扣 10 分，credit_log 4526 条 change_type=3）	压测改用独立号段（注册即 100 分）+ 脚本内信用复核
取消后库里仍读到 status=2、通知 0 条	核对脚本 pymysql 默认 autocommit=False，第一条 SELECT 即开启快照，之后一直读旧数据	统一 autocommit(True)，并在脚本注释中写明

第一件事的教训尤其重要：如果当时直接把"10% 错误率"写进缺陷报告，开发会花大量时间排查锁与事务，而问题在测试数据。第二件事说明——**查库核对之**

前，先确认自己看到的是最新数据。

八、次日计划

任务	具体到用例 / 接口	目标
边界矩阵专项	tests/test_04_boundary.py：maxPeople 1/2/20/21、pageSize 0/1/50/51、pageNum 0/-1、deadline==startTime、已开场取消 4013	四类边界各有用例与预期错误码
信用边界	信用分 0 / 59 / 60 / 100 的报名权限（库中已有 0 分 1 人、55 分 1 人、满档 591 人）	复用今天的天然分布，不重复造数
权限边界	未登录、伪造 token、过期 token、他人报名记录退出、非发起人查看名单	逐条对应 4030 / 授权失败
负向用例	越权评价 4030、重复评价 4011	信用模块负向闭环

九、提交记录与 AI 使用记录

commit	说明
c240936	test(perf): 并发报名压测两档落地，超卖 0 重复 0 死锁 0 - 王嘉萍
0dd2ab3	test(activity): 活动模块 TC008-TC014 迁移与取消级联库侧核对 - 王嘉萍
a628748	docs(test): Day3 测试报告与 README 排期更新 - 王嘉萍

AI 使用记录：本次用 WorkBuddy 做三件事——逐行核对 cancelActivity 的加锁顺序与通知 type 常量（避免按常识写断言）、生成并发驱动与核对脚本骨架、汇总报告表格。真正的判断仍是人工完成：4012 是不是缺陷、快照读到旧数据、名额口径怎么解释，都是先查库再下结论。

附录：复现方式

```
cd test/api
set TEST_BASE_URL=http://127.0.0.1:8081
set MYSQL_PASSWORD=******

:: 并发压测两档
python perf/concurrent_signup.py --vus 100 --activities 1 --scene S1-100
并发抢 20 名额
python perf/concurrent_signup.py --vus 500 --activities 5 --scene S2-500
并发抢 50 名额

:: 取消一致性（8 项核对）
python scripts/check_cancel_consistency.py --joiners 3

:: 全量回归 + 当日报告
python scripts/run_smoke.py --day 3
```

产物：reports/perf_summary_S1-*.json、reports/perf_summary_S2-*.json、reports/cancel_consistency.json、reports/Day3_测试报告_*.md 与同名 HTML。