单元测试报告

单元测试计划

1. 引言

1.1 目的

本文档为StarGames星游游戏零售平台的单元测试活动提供范围、方法、资源和进度方面的指导。

本文档的读者主要是开发（测试）经理、测试人员和开发人员。

1.2 测试策略

以类为单元，采用独立的单元测试策略，通过设计相应的驱动和桩的方法来测试类中的方法。在选择类的被测方法时，根据方法的规模和复杂度进行判定。非空非注释代码行数LOC>20，或者复杂度>3的方法进行单元测试，其他方法不进行单元测试。

对于子类的测试用例采用分层增量测试策略，对子类的变化部分设计新的测试用例，与父类相同的部分则重用父类的测试用例。

执行单元测试的次序是根据《软件详细设计说明书》中的用例实现交互图，从图中最小的依赖关系的类开始测试，再逐步扩大到依赖关系较强的类，直至所有类测试完毕。

1.3 范围

单元测试主要包含了计划阶段、设计阶段、实现阶段和执行阶段4个阶段。本单元测试计划是整个软件开发项目中的一部分，起始于详细设计阶段，直到单元测试阶段结束后终止。该计划主要处理与StarGames星游游戏零售平台系统单元测试有关的任务安排、资源需求、人力需求、风险管理、进度安排等内容。

1.4 参考文献

《软件需求规格说明书》

《软件详细设计说明书》

《用户界面规格说明书》

2. 测试项目

根据《软件详细设计说明书》中的详细设计内容，单元测试的测试项目如以下所示：

- Admin模块
- Change模块

- Friend模块
- Login模块

3. 测试方法

根据类规约和操作规约构建测试用例，合理利用传统等价类划分法、边界值分析法、判定表法等黑盒测试方法和语句覆盖、路径覆盖等白盒测试方法。

对具有特殊需求的类辅以下面两种方法设计测试用例：

1. 根据状态转换图构建测试用例。该方法根据被测试的类的对象所处的状态以及状态之间的转移来构造测试用例，对状态之间和状态内部的每一个转换及其可能发生的异常转换、转换的监护条件等进行全面测试。
2. 基于实现构建测试用例。该方法利用传统逻辑覆盖法、数据流分析法等白盒测试技术对程序的逻辑结构或数据流进行测试，以达到一定的代码覆盖率。

更详细的测试策略描述请参考《单元测试说明》。

4. 测试通过/失败标准

测试通过的标准表述如下：

1. 所有单元测试的用例都被执行并通过；
2. 所有发现的缺陷都被修正并通过回归测试；
3. 所有被测对象的前置条件和后置条件组合覆盖率达到100%，或能明确给出不需要达到的理由；
4. 单元测试报告被授权人批准。

测试失败标准表述如下：

1. 严重缺陷密度大于15个/KLOC；
2. 发现软件结构有重大设计问题，其修改会导致20%以上的接口、功能、数量的变化，进一步测试相关特性已经无意义；
3. 发现关键功能未被设计，该功能的设计会导致20%以上的接口、功能、数量的变化，进一步测试相关特性已经无意义。

测试结果审批过程：开发人员提交单元测试报告—开发或测试经理签字并提交 SQA→SQA对报告进行评审并签字（测试经理参与）→产品经理签字。

5. 测试挂起/恢复的条件

测试挂起的条件有：

- 当某个类在单元测试执行过程中发现有阻塞用例的时候，该类的单元测试被挂起。
- 当有20%以上的被测类都遇到有阻塞用例时，所有类的单元测试都被挂起。
- 当出现有新增需求的时候，与该需求相关的所有类的单元测试都被挂起。

- 当开发人员提出要进行设计变更的时候，相关类的单元测试将被挂起。

测试恢复的条件有：

- 测试被挂起的条件已经被解决。
- 需要恢复测试的对象达到单元测试入口条件，在这里要求这些被测对象已经通过代码走读（要提交走读报告）和语法检查（要提交检查结果）。

6. 单元测试交付物

- 单元测试计划
- 单元测试设计规格
- 单元测试用例规格
- 单元测试用例脚本
- 单元测试驱动和桩代码
- 单元测试执行日志
- 单元测试报告

7. 单元测试任务

单元测试任务表如下所示：

任务标识	任务描述	责任人	优先级	依赖关系
UT_TASK_001	单元测试计划制定	测试经理	高	
UT_TASK_003	单元测试计划评审	SQA	中	UT_TASK_001
UT_TASK_005	单元测试计划修改	测试经理	中	UT_TASK_001
UT_TASK_007	单元测试设计规格制定	开发或测试人员	中	UT_TASK_001
UT_TASK_009	单元测试设计规格评审	SQA	中	UT_TASK_007
UT_TASK_011	单元测试设计规格修改	开发或测试人员	中	UT_TASK_009
UT_TASK_013	单元测试用例规格设计	开发或测试人员	高	UT_TASK_011
UT_TASK_015	单元测试用例规格评审	SQA	中	UT_TASK_013
UT_TASK_017	单元测试用例规格修改	开发或测试人员	中	UT_TASK_015
UT_TASK_019	驱动、桩、用例脚本代码实现	开发或测试人员	中	UT_TASK_017

UT_TASK_021	驱动、桩、脚本代码走查	SQA	低	UT_TASK_019
UT_TASK_023	驱动、桩、脚本代码修改	开发或测试人员	低	UT_TASK_021
UT_TASK_025	单元测试执行及回归	开发或测试人员	高	UT_TASK_023
UT_TASK_027	单元测试报告	测试经理	高	UT_TASK_025
UT_TASK_029	单元测试报告审批	开发或测试经理	高	UT_TASK_027

8. 环境需求

8.1 测试工具

QTRunner、CheckStyle、PMD、JUnit等

9. 角色和职责

角色	职责
产品经理	解决资源（包括人、工具等）需求，对单元测试结果进行监督
开发经理	协助制订单元测试计划，安排单元测试任务
测试经理	制订单元测试计划，安排单元测试任务，参与单元测试结果验收
SQA	对单元测试过程（包括代码走读、正规检视活动）进行监控
开发或（和）测试人员	完成单元测试需要的输入，并完成单元测试设计规格、单元测试用例规格的制定，执行单元测试，记录发现的问题，修改问题，并负责问题的回归测试。与此同时，负责定位问题和解决问题

10. 单元测试进度

任务标识	任务描述	周期/天
UT_TASK_001	单元测试计划制定	2
UT_TASK_003	单元测试计划评审	2

UT_TASK_005	单元测试计划修改	2
UT_TASK_007	单元测试设计规格制定	3
UT_TASK_009	单元测试设计规格评审	2
UT_TASK_011	单元测试设计规格修改	2
UT_TASK_013	单元测试用例规格设计	1
UT_TASK_015	单元测试用例规格评审	1
UT_TASK_017	单元测试用例规格修改	1
UT_TASK_019	驱动、桩、用例脚本代码实现	7
UT_TASK_021	驱动、桩、脚本代码走查	2
UT_TASK_023	驱动、桩、脚本代码修改	2
UT_TASK_025	单元测试执行及回归	1
UT_TASK_027	单元测试报告	2
UT_TASK_029	单元测试报告审批	1

用例分析与设计

1. Admin模块

1.1 AdminControllerTest

1.1.1 testQueryBuyerListItems

1. 标识符定义

SG_Unit_001_001

2. 被测特性

输入BuyerID成功，查询成功

输入BuyerID错误，查询失败

3. 测试方法

BuyerID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_001_001_001	输入参数任意为空的情况	低
SG_Unit_001_001_002	输入参数非空但数据库不存在	中
SG_Unit_001_001_003	输入参数正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_001_001_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	管理员输入查询的BuyerID	
用例序号	输入	期望结果
001	BuyerID=""	返回 false，反馈 BuyerID 参 数为空的错误信息

测试项编号	SG_Unit_001_001_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	管理员输入查询的BuyerID	
用例序号	输入	期望结果
001	BuyerID="11123123"	返回 false，反馈 BuyerID 参 数不存在的错误信息

测试项编号	SG_Unit_001_001_003

优先级	高	
测试项描述	输入参数非空但数据库不存在	
前置条件	管理员输入查询的BuyerID	
用例序号	输入	期望结果
001	BuyerID="10001"	返回买家信息

1.2 AdminServiceTest

1.2.1 testBlockBuyer

1. 标识符定义

SG_Unit_001_002

2. 被测特性

输入BuyerID成功，Buyer未被封禁，封禁成功

输入BuyerID成功，Buyer已封禁，封禁失败

输入BuyerID错误，封禁失败

3. 测试方法

BuyerID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。对于数据库存在情况，又可以划分为Buyer已被封禁和未被封禁两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_001_002_001	输入参数任意为空的情况	低
SG_Unit_001_002_002	输入参数非空但数据库不存在	中
SG_Unit_001_002_003	输入正确但已被封禁	高
SG_Unit_001_002_004	成功完成封禁	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_001_002_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	管理员输入要封禁的BuyerID	
用例序号	输入	期望结果
001	BuyerID=""	返回 false，反馈 BuyerID 参 数为空的错误信息

测试项编号	SG_Unit_001_002_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	管理员输入要封禁的BuyerID	
用例序号	输入	期望结果
001	BuyerID="11123123"	返回 false，反馈 BuyerID 参 数不存在的错误信息

测试项编号	SG_Unit_001_002_003	
优先级	高	
测试项描述	输入参数非空且数据库存在，但已被封禁	
前置条件	管理员输入要封禁的BuyerID	
用例序号	输入	期望结果
001	BuyerID="10010"	返回 false，返回账号已被封禁的错误信息

测试项编号	SG_Unit_001_002_004
优先级	高
测试项描述	输入正确且Buyer未被封禁

前置条件	管理员输入要封禁的BuyerID	
用例序号	输入	期望结果
001	BuyerID="10001"	返回true，返回账号成功完成封禁

1.2.2 testGetCommodityInfo

1. 标识符定义

SG_Unit_001_003

2. 被测特性

输入CommodityID成功，查询成功

输入CommodityID错误，查询失败

3. 测试方法

CommodityID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_001_003_001	输入参数任意为空的情况	低
SG_Unit_001_003_002	输入参数非空但数据库不存在	中
SG_Unit_001_003_003	输入参数正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_001_003_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	管理员输入查询的CommodityID	

用例序号	输入	期望结果
001	CommodityID=""	返回 false，反馈 CommodityID 参数为空的错误信息

测试项编号	SG_Unit_001_003_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	管理员输入查询的CommodityID	
用例序号	输入	期望结果
001	CommodityID="4214124"	返回 false，反馈 CommodityID 参数不存在的错误信息

测试项编号	SG_Unit_001_003_003	
优先级	高	
测试项描述	输入正确	
前置条件	管理员输入查询的CommodityID	
用例序号	输入	期望结果
001	CommodityID="100001"	返回商品信息

1.2.3 testWithdrawCommodity

1. 标识符定义

SG_Unit_001_004

2. 被测特性

输入CommodityID成功，删除成功

输入CommodityID错误，删除失败

3. 测试方法

CommodityID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_001_004_001	输入参数任意为空的情况	低
SG_Unit_001_004_002	输入参数非空但数据库不存在	中
SG_Unit_001_004_003	输入参数正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_001_004_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	管理员输入要删除的CommodityID	
用例序号	输入	期望结果
001	CommodityID=""	返回 false，反馈 CommodityID 参数为空的错误信息

测试项编号	SG_Unit_001_004_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	管理员输入要删除的CommodityID	
用例序号	输入	期望结果
001	CommodityID="21412412"	返回 false，反馈 CommodityID 参数不存在的错

		误信息

测试项编号	SG_Unit_001_004_003	
优先级	高	
测试项描述	输入正确	
前置条件	管理员输入删除的CommodityID	
用例序号	输入	期望结果
001	CommodityID="100001"	返回true，成功删除对应商品

2. Change模块

2.1 ChangeControllerTest

2.2 ChangeServiceTest

2.2.1 testGetGameIDforLibrary

1. 标识符定义

SG_Unit_002_001

2. 被测特性

输入旧密码正确，输入邮箱验证码正确，输入新密码非空，更改成功

输入旧密码错误，更改失败

输入旧密码正确，输入邮箱验证码错误，更改失败

输入旧密码正确，输入邮箱验证码正确，输入新密码错误，更改失败

3. 测试方法

旧密码参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为与数据库中对应和不对应两种情况。

对于对应的情况，又可以考虑将邮箱验证码参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为与数据库中对应和不对应两种情况。对于对应的情况，又可以考虑将新密码参数的等价类划分考虑空和非空情况。

4. 测试项标识

测试项标识符	测试项描述	优先级

SG_Unit_002_001_001	输入原密码参数为空的情况	低
SG_Unit_002_001_002	输入原密码参数非空但与数据库不对应	中
SG_Unit_002_001_003	原密码参数输入正确但邮箱验证码为空的情况	低
SG_Unit_002_001_004	原密码参数输入正确但邮箱验证码与数据库不对应	中
SG_Unit_002_001_005	输入正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_002_001_001	
优先级	低	
测试项描述	输入原密码参数为空的情况	
前置条件	用户输入账号的原密码、邮箱验证码和新密码	
用例序号	输入	期望结果
001	_OLDPASSWD=""	返回 -1，反馈原密码参数为空的错误信息

测试项编号	SG_Unit_002_001_002	
优先级	中	
测试项描述	输入原密码参数非空但与数据库不对应	
前置条件	用户输入账号的原密码、邮箱验证码和新密码	
用例序号	输入	期望结果
001	_OLDPASSWD="432141"	返回 -1，反馈原密码输入错误的错误信息

测试项编号	SG_Unit_002_001_003	
优先级	低	
测试项描述	原密码参数输入正确但邮箱验证码为空的情况	
前置条件	用户输入账号的原密码、邮箱验证码和新密码	
用例序号	输入	期望结果
001	_OLDPASSWD="123456" _CODE=""	返回 -2，反馈邮箱验证码为空的错误信息

测试项编号	SG_Unit_002_001_004	
优先级	中	
测试项描述	原密码参数输入正确但邮箱验证码与数据库不对应	
前置条件	用户输入账号的原密码、邮箱验证码和新密码	
用例序号	输入	期望结果
001	_OLDPASSWD="123456" _CODE="412313"	返回 -2，反馈邮箱验证码错误的错误信息

测试项编号	SG_Unit_002_001_005	
优先级	高	
测试项描述	输入正确	
前置条件	用户输入账号的原密码、邮箱验证码和新密码	
用例序号	输入	期望结果
001	OLDPASSWD="123456" CODE="268572" NEWPASSWD=“1234567”	返回 0，反馈修改成功的提示信息

3. FriendList模块

3.1 FriendListControllerTest

3.2 FriendListServiceTest

3.2.1 testGetFriendInfo

1. 标识符定义

SG_Unit_003_001

2. 被测特性

输入Friend_ID成功，查询成功

输入Friend_ID错误，查询失败

3. 测试方法

Friend_ID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_003_001_001	输入参数任意为空的情况	低
SG_Unit_003_001_002	输入参数非空但数据库不存在	中
SG_Unit_003_001_003	输入参数正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_003_001_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	用户输入要查询的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID=""	返回 false，反馈 Friend_ID参数为空的错误信息

测试项编号	SG_Unit_003_001_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	用户输入要查询的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID="11123123"	返回 false，反馈 Friend_ID 参数不存在的错误信息

测试项编号	SG_Unit_003_001_003	
优先级	高	
测试项描述	输入正确	
前置条件	用户输入要查询的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID="10002“	返回好友信息

3.2.2 testDeleteFriend

1. 标识符定义

SG_Unit_003_002

2. 被测特性

输入Friend_ID成功，删除成功

输入Friend_ID错误，删除失败

3. 测试方法

Friend_ID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为数据库中存在和不存在两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级

SG_Unit_003_002_001	输入参数任意为空的情况	低
SG_Unit_003_002_002	输入参数非空但数据库不存在	中
SG_Unit_003_002_003	输入参数正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_003_002_001	
优先级	低	
测试项描述	输入参数任意为空的情况	
前置条件	用户输入要删除的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID=""	返回 false，反馈 Friend_ID参数为空的错误信息

测试项编号	SG_Unit_003_002_002	
优先级	中	
测试项描述	输入参数非空但数据库不存在	
前置条件	用户输入要删除的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID="11123123"	返回 false，反馈用户没有对应ID的好友

测试项编号	SG_Unit_003_002_003
优先级	高
测试项描述	输入正确

前置条件	用户输入要删除的Friend_ID	
用例序号	输入	期望结果
001	Friend_ID="10002”	返回true，返回好友已成功删除

4. Login模块

4.1 LoginControllerTest

4.2 LoginServiceTest

4.2.1 testCheckIdentity

1. 标识符定义

SG_Unit_004_001

2. 被测特性

输入UserID成功，输入密码成功，登陆成功

输入UserID成功，输入密码错误，登陆失败

输入UserID错误，登录失败

3. 测试方法

UserID参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为与数据库中存在和不存在两种情况。

对于存在的情况，又可以考虑将密码参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为与数据库中对应和不对应两种情况。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_004_001_001	输入UserID参数任意为空的情况	低
SG_Unit_004_001_002	输入参数非空但数据库不存在	中
SG_Unit_004_001_003	输入UserID正确但密码参数任意为空的情况	低
SG_Unit_004_001_004	输入UserID正确但密码参数与数据库不对应	高
SG_Unit_004_001_005	输入均正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_004_001_001	
优先级	低	
测试项描述	输入UserID参数任意为空的情况	
前置条件	用户输入用户ID和密码进行登录	
用例序号	输入	期望结果
001	User_ID=""	返回 false，反馈 User_ID参数为空的错误信息

测试项编号	SG_Unit_004_001_002	
优先级	中	
测试项描述	输入UserID非空但数据库不存在	
前置条件	用户输入用户ID和密码进行登录	
用例序号	输入	期望结果
001	User_ID="1121312"	返回 false，反馈用户没有对应用户存在

测试项编号	SG_Unit_004_001_003	
优先级	低	
测试项描述	输入UserID正确但密码参数任意为空的情况	
前置条件	用户输入用户ID和密码进行登录	
用例序号	输入	期望结果
001	User_ID="10001”	

	PassWord=""	返回 false，反馈密码为空的错误信息

测试项编号	SG_Unit_004_001_004	
优先级	高	
测试项描述	输入UserID正确但密码参数与数据库不对应	
前置条件	用户输入用户ID和密码进行登录	
用例序号	输入	期望结果
001	User_ID="10001” PassWord="123456"	返回 false，反馈密码错误的错误信息

测试项编号	SG_Unit_004_001_005	
优先级	高	
测试项描述	输入均正确	
前置条件	用户输入用户ID和密码进行登录	
用例序号	输入	期望结果
001	User_ID="10001” PassWord="1234567"	提示用户登陆成功

4.2.2 testAddNewUser

1. 标识符定义

SG_Unit_004_002

2. 被测特性

输入PhoneNumber成功，输入密码和用户类型成功，注册成功

输入PhoneNumber不合法时，注册失败

输入电话号码已存在时，注册失败

当输入参数任意为空时，注册失败

3. 测试方法

PhoneNumber参数的等价类划分考虑空和非空情况。对于非空情况，又可以划分为PhoneNumber合法与不合法情况。对于合法情况，再考虑为数据库中存在和不存在两种情况。对于存在的情况，又可以考虑将密码参数和用户类型参数的等价类划分考虑空和非空情况。对于用户类型的非空情况，要判断其是否满足对应的范围。

4. 测试项标识

测试项标识符	测试项描述	优先级
SG_Unit_004_002_001	输入PhoneNumber参数任意为空的情况	低
SG_Unit_004_002_002	输入PhoneNumber参数非空但电话号码不合法	中
SG_Unit_004_002_003	输入PhoneNumber正确但电话号码已注册的情况	高
SG_Unit_004_002_004	输入密码和用户类型任意为空的情况	低
SG_Unit_004_002_005	输入用户类型不满足对应范围的情况	低
SG_Unit_004_002_006	输入均正确	高

5. 测试通过/失败标准

所有的用例都必须被执行，且没有发现错误

6. 对应用例

测试项编号	SG_Unit_004_002_001	
优先级	低	
测试项描述	输入PhoneNumber参数任意为空的情况	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber=""	返回 false，反馈PhoneNumber参数为空的错误信息

测试项编号	SG_Unit_004_002_002	
优先级	中	
测试项描述	输入PhoneNumber参数非空但电话号码不合法	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber="1121312"	返回 false，反馈输入PhoneNumber参数不合法

测试项编号	SG_Unit_004_002_003	
优先级	高	
测试项描述	输入PhoneNumber正确但电话号码已注册的情况	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber="13158359476”	返回 false，反馈输入PhoneNumber参数已被注册的错误信息

测试项编号	SG_Unit_004_002_004	
优先级	低	
测试项描述	输入密码和用户类型任意为空的情况	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber="13691581788” PassWord="123456" Type_ID=""	返回 false，反馈Type_ID参数为空的错误信息
002	PhoneNumber="13691581788”	返回 false，反馈PassWord参数为空的错误信息

	PassWord="" Type_ID="0"	

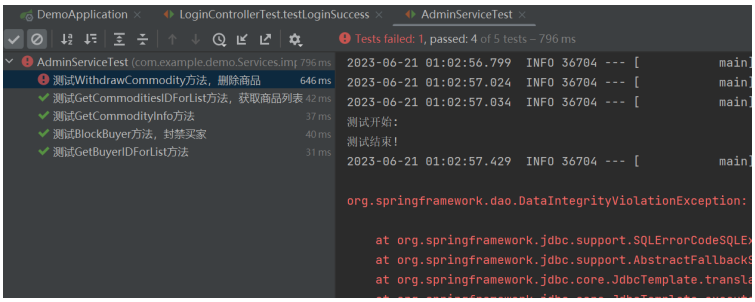
测试项编号	SG_Unit_004_002_005	
优先级	低	
测试项描述	输入用户类型不满足对应范围的情况	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber="13691581788” PassWord="123456" Type_ID="3"	返回 false，反馈Type_ID参数错误的错误信息

测试项编号	SG_Unit_004_002_006	
优先级	高	
测试项描述	输入均正确	
前置条件	用户进行注册	
用例序号	输入	期望结果
001	PhoneNumber="13691581788” PassWord="123456" Type_ID="0"	提示用户注册成功

单元测试结果

1. Admin模块

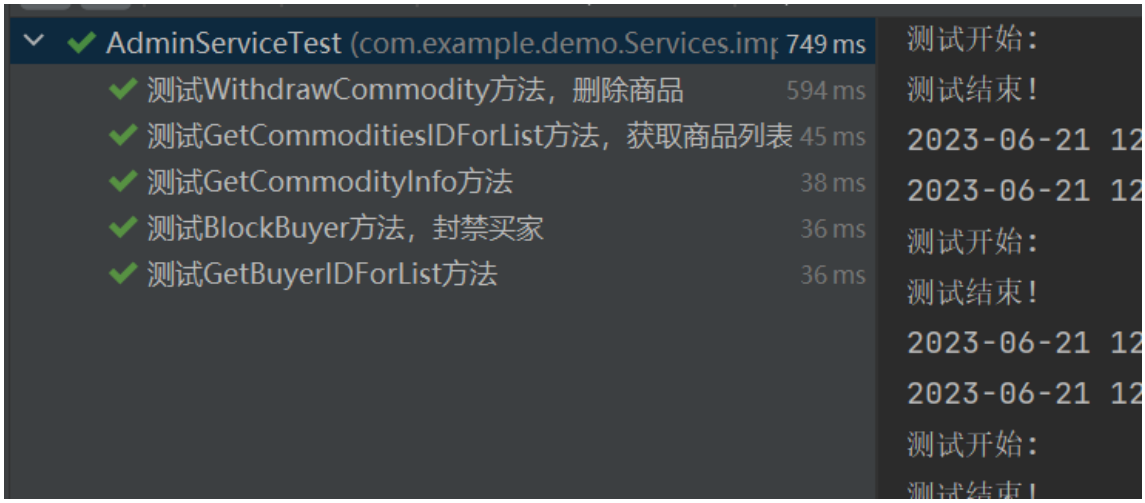
1.1 AdminServiceTest



一共5个测试用例，4个通过，一个未通过

原因：受到外键约束导致删除商品对商品评论等表的级联删除失败

修改后测试结果截图



1.2 AdminServiceStaticTest

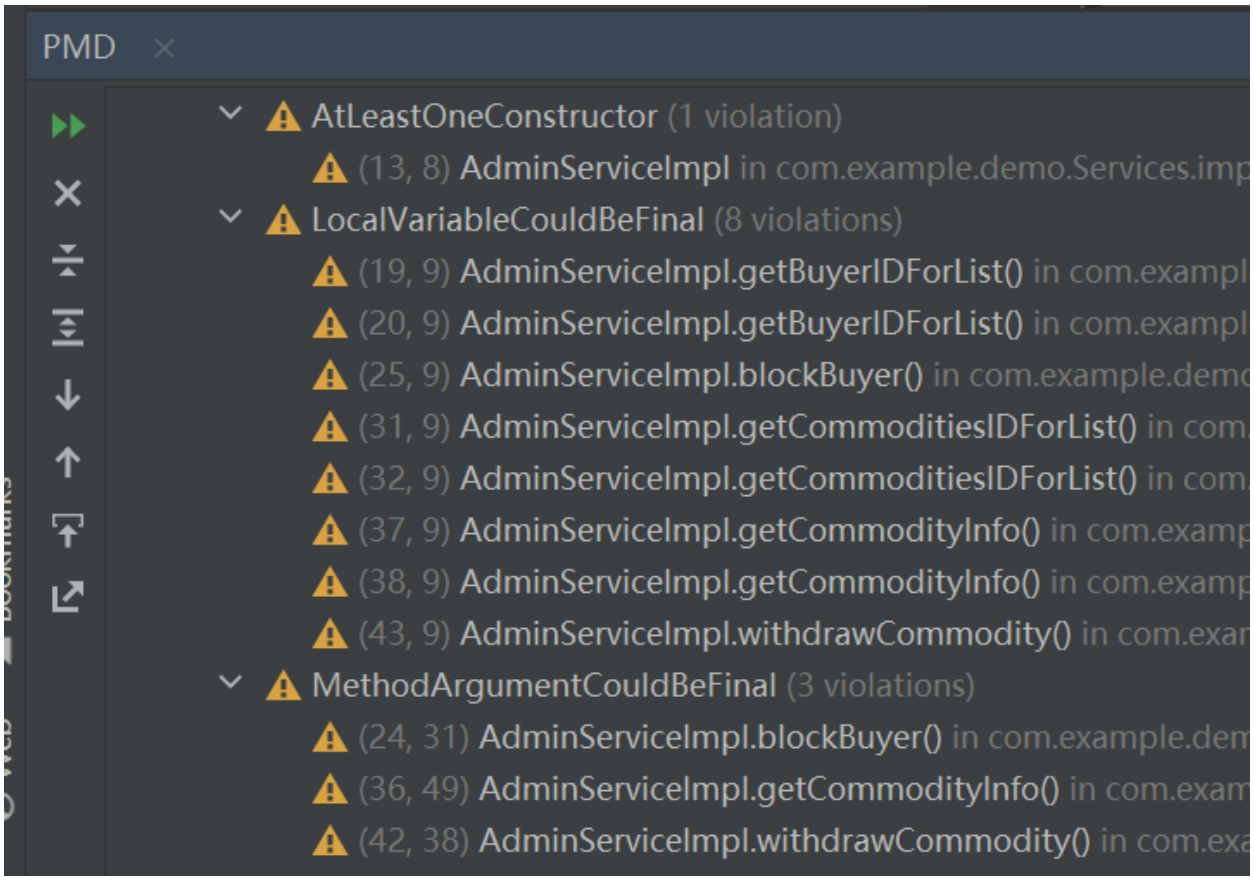
我们分别使用了Checkstyle和pmd对我们的代码进行了静态测试

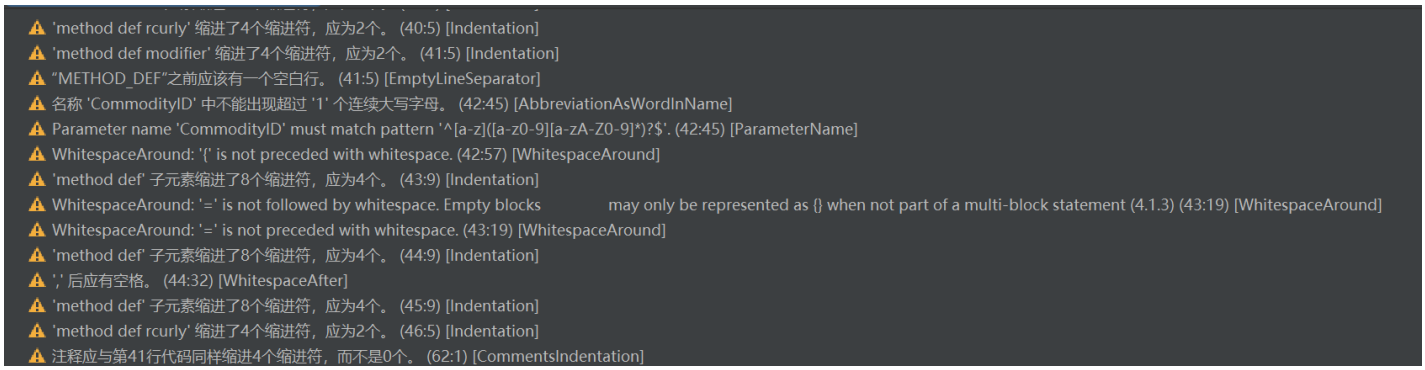
Checkstyle和PMD是两种常用的静态分析工具，它们都用于对代码进行静态测试，但在其实现和目标方面存在差异。

Checkstyle主要关注代码风格的规范化和约定，它使用特定的配置文件来定义代码风格规则并与代码库进行比较。Checkstyle可以检查代码是否符合 Java 编码惯例、命名规范、代码注释等方面的规则。在项目中使用 Checkstyle 可以帮助维护一致的代码风格，提高代码的可读性和可维护性。

PMD采用更加智能的方式分析代码，它可以检查代码中的潜在问题，如未使用的变量或方法、不必要的类型转换、不良的代码实践等，并提供相应的建议和修复措施。PMD不仅可以检测 Java 代码，还可以检测其他语言，如C++、JavaScript等。

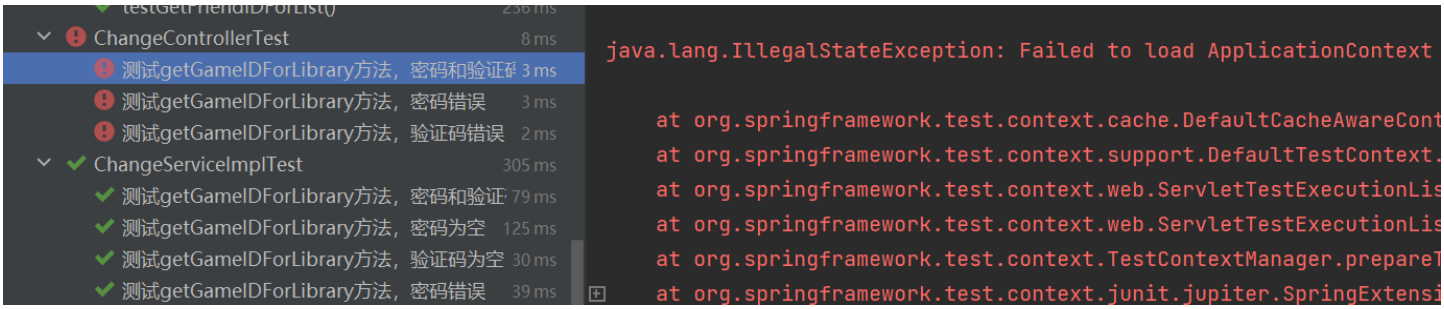
我们选择Checkstyle作为我们进行静态测试的工具，后续只展示checkstyle的部分





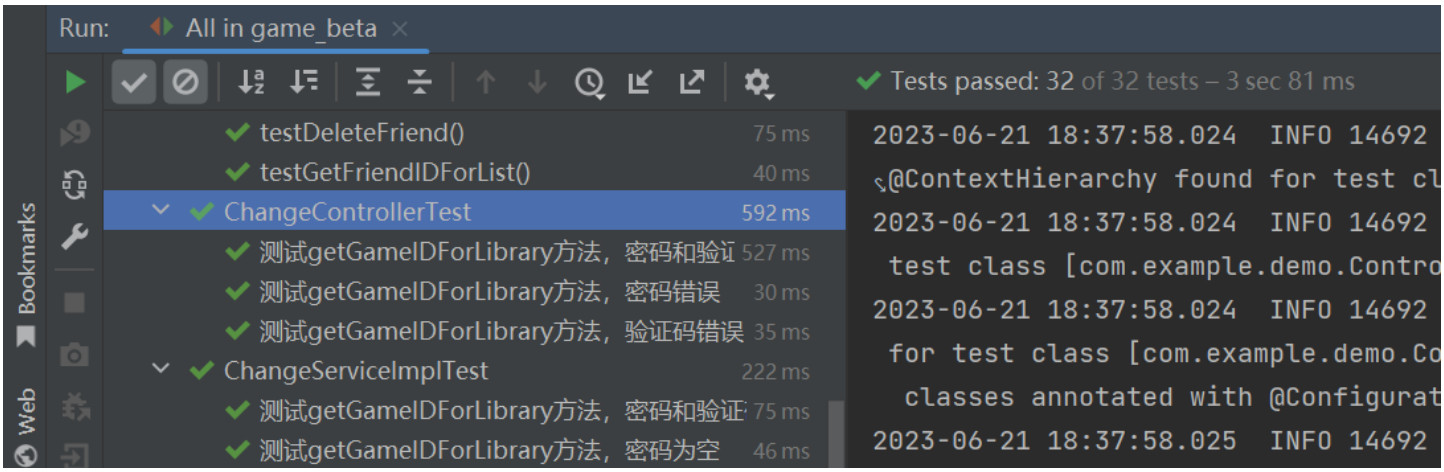
2. Change模块

2.1 ChangeControllerTest

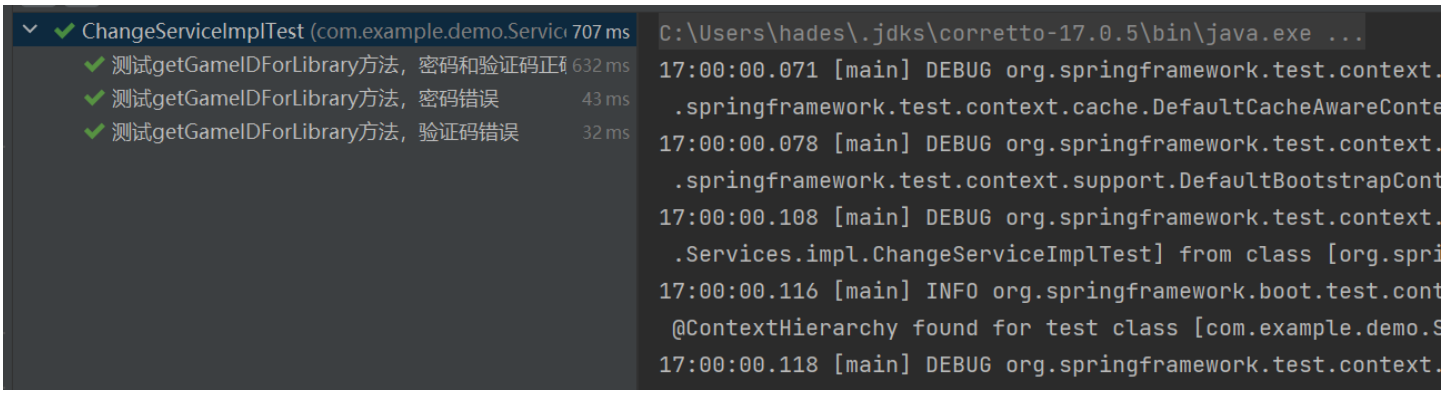


一共三个测试用例均不通过

原因：Controller类直接对ServiceImpl类进行调用，而不是使用Service提供的接口

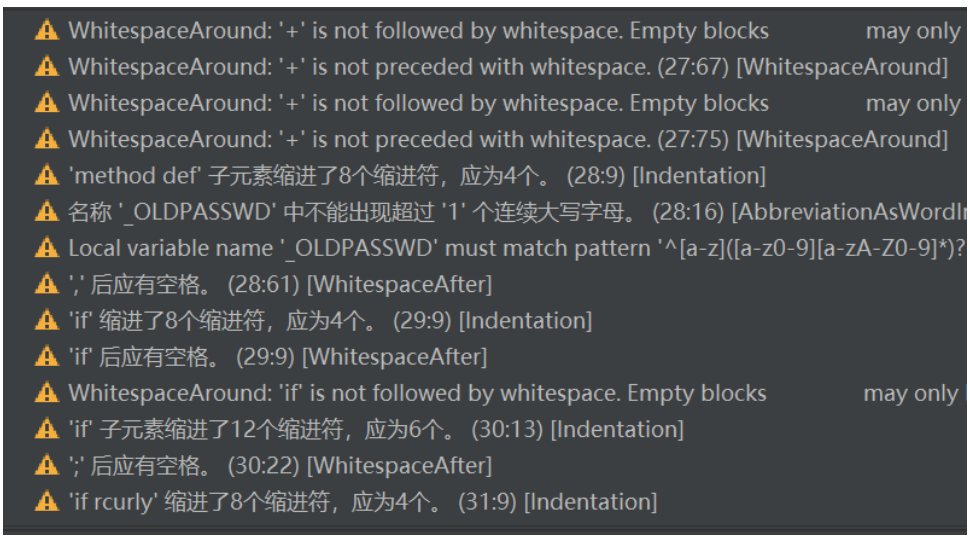


2.2 ChangeServiceTest



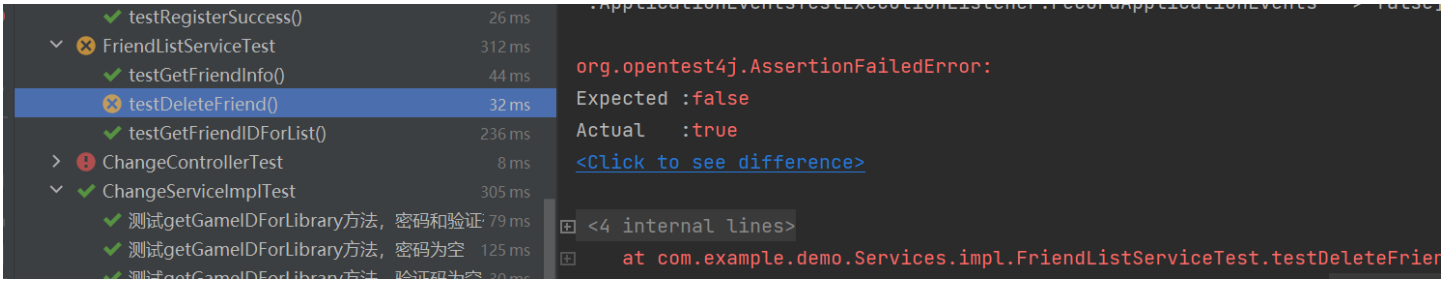
一共三个测试用例，全部通过

2.3 ChangeServiceStaticTest



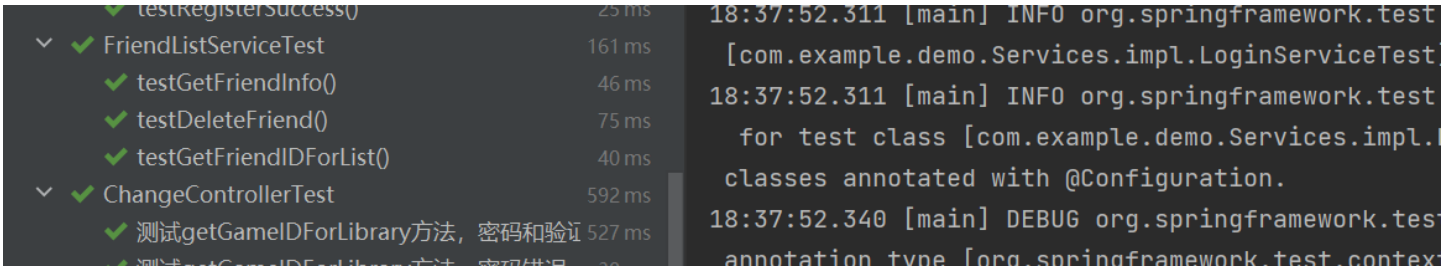
3. FriendList模块

3.1 FriendListServiceTest



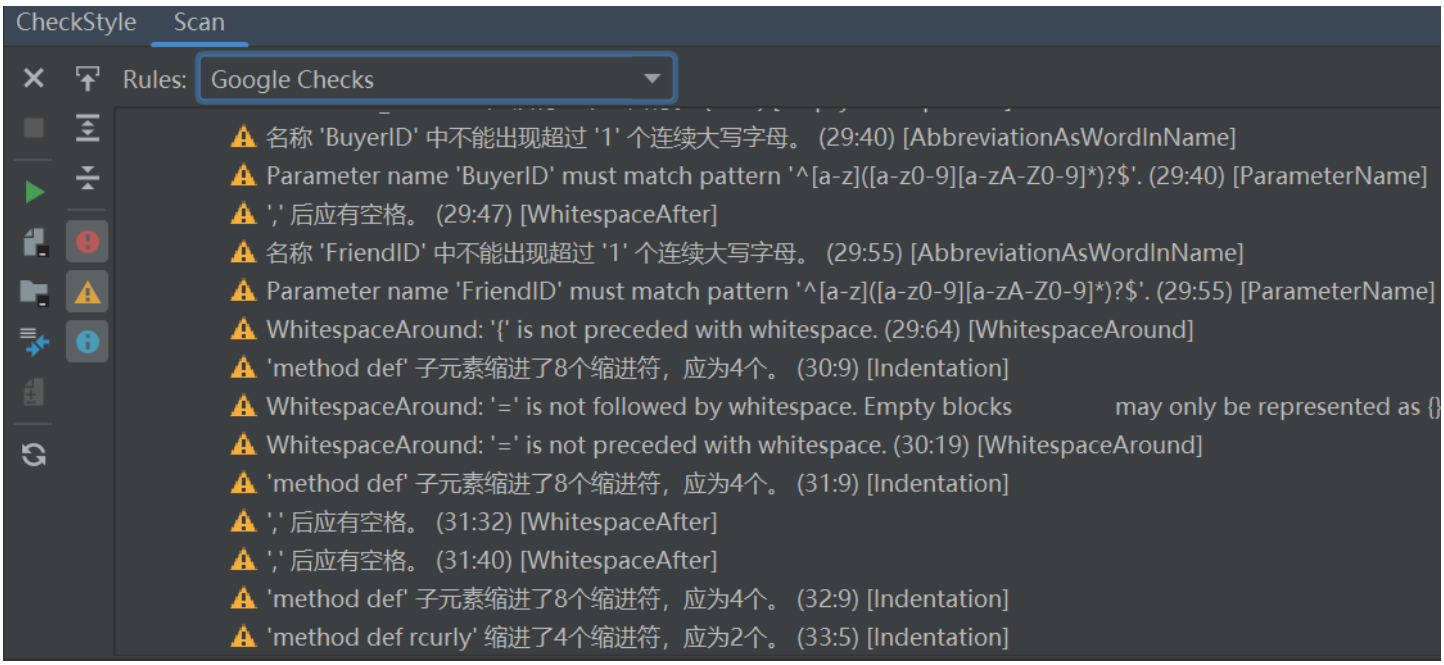
一共三个测试用例，一个未通过

原因：在删除好友时未对空值和错误值进行判断



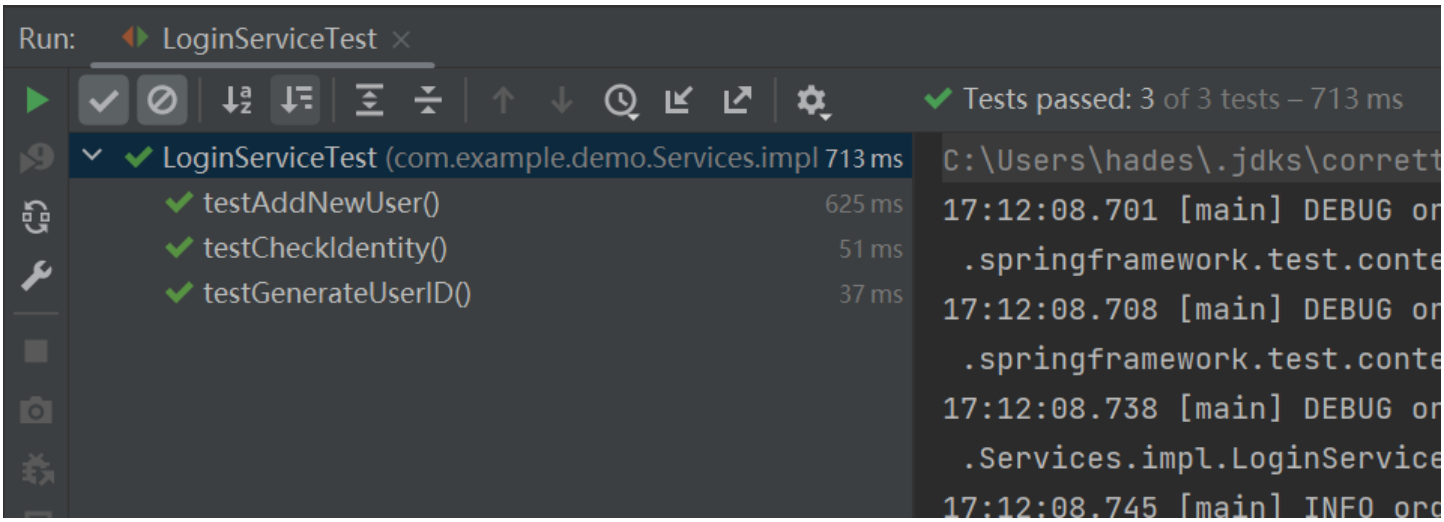
修改后三个测试用例全部通过

3.2 FriendListServiceStaticTest



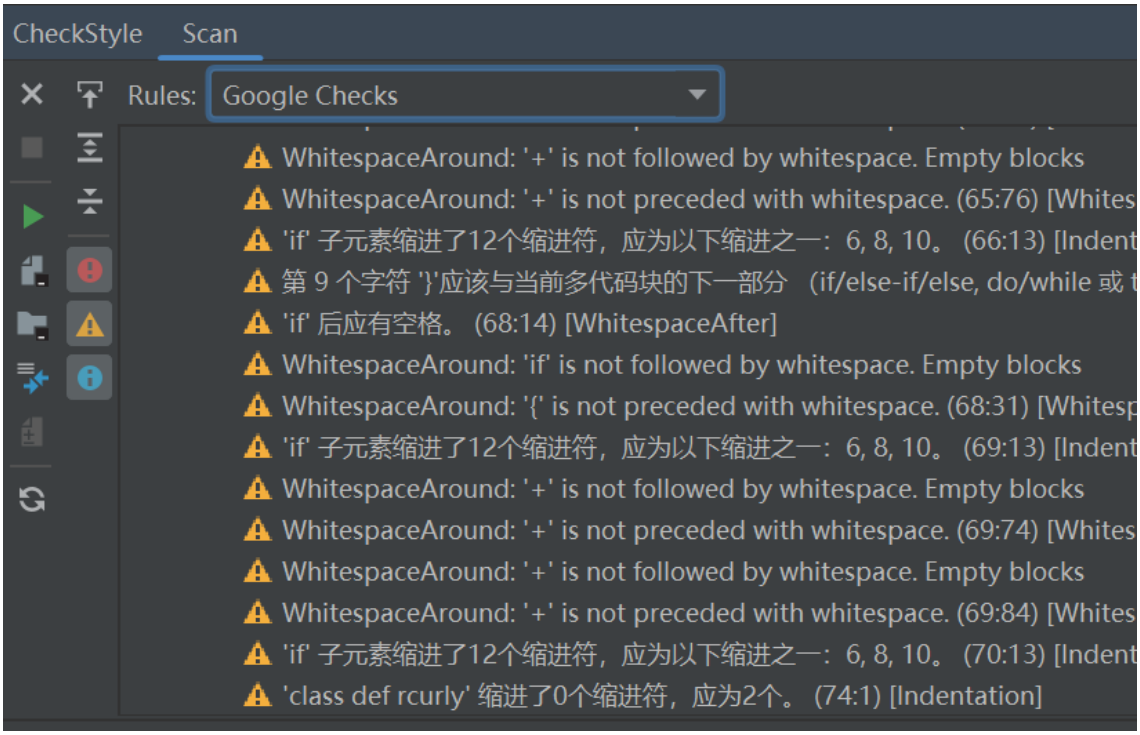
4. Login模块

4.1 LoginServiceTest

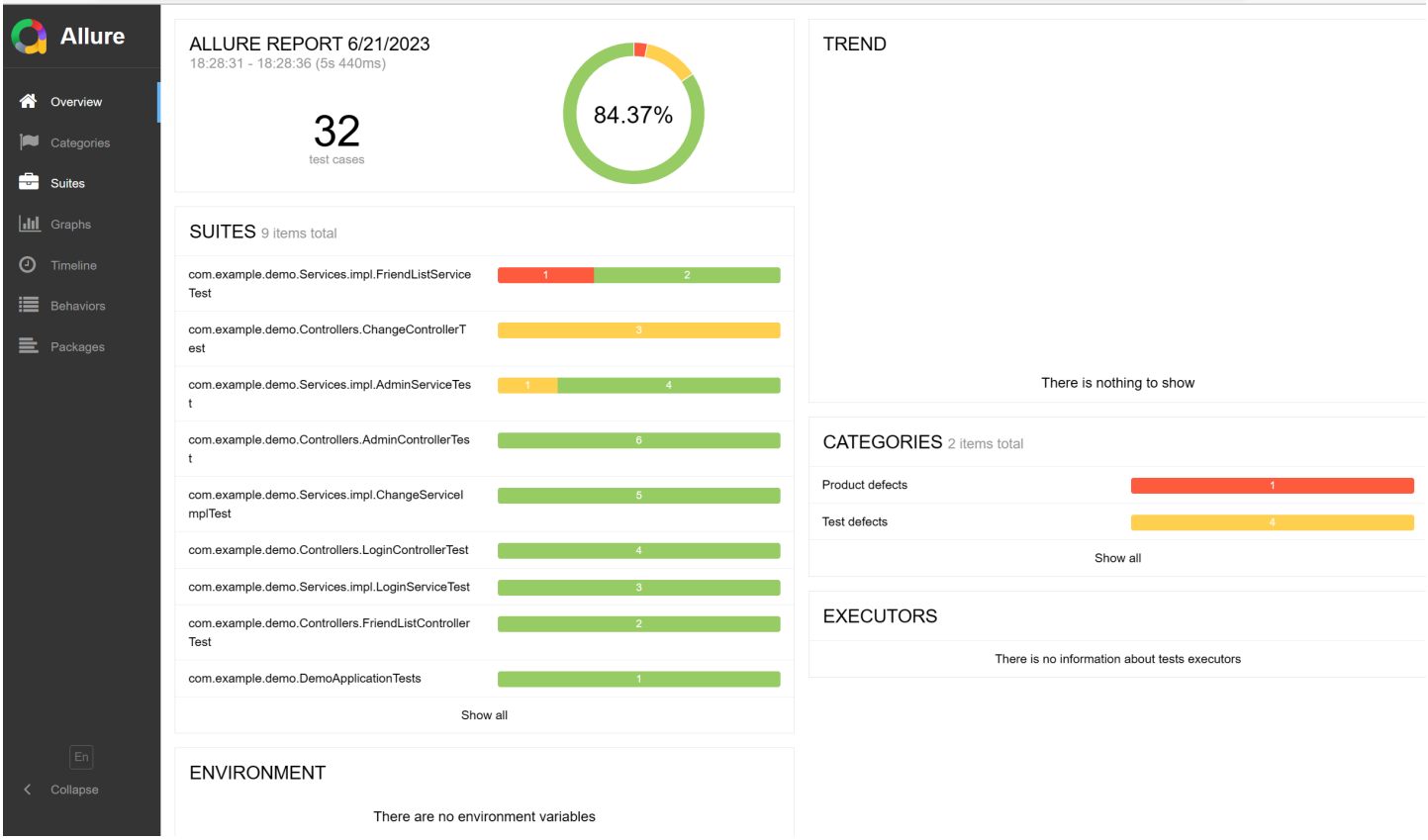


一共三个测试用例，全部通过

4.2 LoginServiceStaticTest



测试历史版本前端渲染结果



测试修复版本前端渲染结果