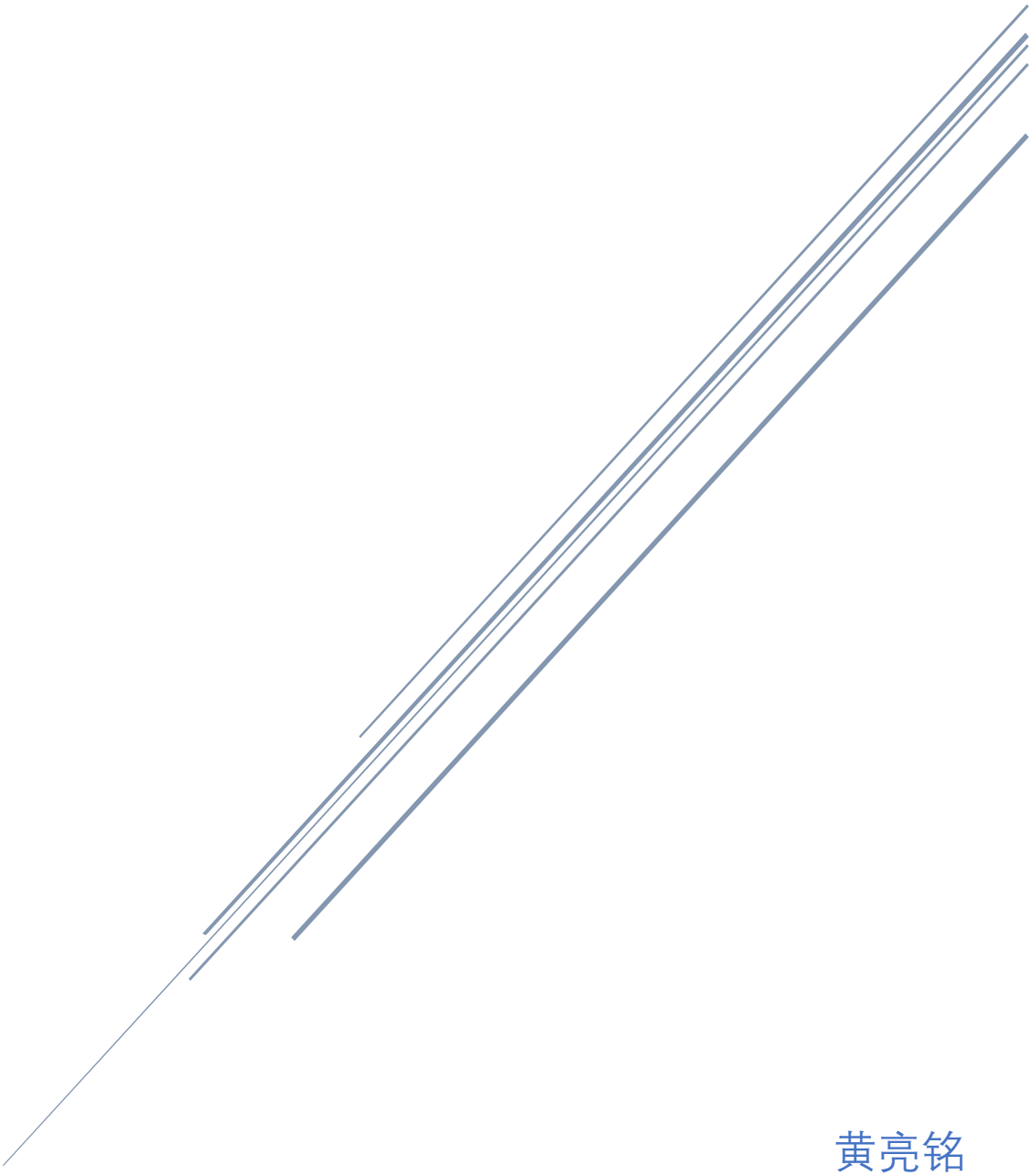
账易通

项目测试报告



黄亮铭

2022155028

目录

1 测试方法

1.1 黑盒测试

黑盒测试，也被称作黑箱测试，是一种在软件开发过程中广泛应用的测试方法。它将软件系统视为一个无法透视的“黑盒子”，专注于从用户的角度出发，依据需求规格说明书来验证软件的功能是否达到了预期的效果。黑盒测试的核心在于它与软件的内部结构和实现细节无关，使得测试过程更加直观和易于理解。

这种测试方法包括以下几种主要的测试用例设计技术：等价类划分法、边界值划分法和决策表法。

黑盒测试通常由测试人员执行，他们不需要了解软件的内部实现细节。这种测试方法适用于软件开发生命周期的早期阶段，以及任何时候想要从用户的角度来评估软件的功能性。黑盒测试有助于确保软件满足用户需求和业务需求，同时提高软件的可用性和可靠性。然而，黑盒测试也有其局限性。例如，可能无法完全揭示软件内部的逻辑错误，因为测试不涉及软件的内部结构。又或者如果测试用例设计不够全面，可能无法覆盖所有潜在的错误和边界情况。

1.2 白盒测试

白盒测试作为基于软件内部结构的测试方法，通过语句覆盖、分支覆盖、条件覆盖和路径覆盖等用例设计方法，对程序内部逻辑路径进行全面检测，其优势在于能深入挖掘代码内部的逻辑、语法及算法等错误，实现较高的代码覆盖率，但也存在依赖代码细节且因逻辑路径众多致测试成本高昂不足。

白盒测试通常由开发人员执行，因为它需要对软件的内部逻辑和结构有深入的了解。这种测试方法有助于确保软件的内部质量，并且可以在开发过程中早期发现潜在的问题。

1.3 前后端分离测试

前后端分离测试是现代软件开发实践中的一种重要测试策略，它与经典的黑盒测试和白盒测试有着本质的区别。这种测试方法与敏捷开发中的前后端分离架构紧密相关，随着软件开发向精细化发展，测试策略也相应地进行了调整和优化。

前端测试依赖于前端开发，后端测试依赖于后端开发，开发和测试仍然串行，但是在前后端联合调试时只需要执行前后端通过性案例，而无需再关注其他问题，如用户体验、客户端性能和服务端性能等，提高了测试效率，做到测试前移。

前后端分离测试不仅提高了测试的效率，而且通过专注于各自的测试领域，使得测试更加全面和深入。这种方法论在现代软件开发中越来越受到重视，因为它能够更好地适应快速变化的市场需求和技术进步。

1.4 单元测试

单元测试是软件测试中的一种基本测试方法，它主要针对软件中的最小可测试单元进行测试。在面向对象编程中，最小单元通常是类或者方法；在过程式编程中，最小单元可以是函数或者过程。单元测试的目的是验证这些单元的功能是否符合预期，检查其在各种输入情况下的输出是否正确，并且保证每个单元在独立运行时行为的正确性。

单元测试的优点在于它可以在软件开发早期发现问题，将错误遏制在较小的单元范围内，降低后续修复成本；同时方便代码维护和重构，为代码质量提供保障，还能作为代码功能的参考文档帮助开发人员理解代码。然而，单元测试也有缺点，它编写起来比较耗时耗力，增加了前期开发成本，尤其对于小型或时间紧张的项目可能不太友好，并且对测试环境要求较高，包括需要合适的测试框架和模拟工具，还可能受到测试环境搭建不完善和框架版本兼容等问题的干扰。

2 项目测试工具

项目前后端测试工具选择 postman。Postman 是一款功能强大的 API 开发和测试工具。它提供了一个直观的图形化界面，让用户可以方便地发送各种 HTTP 请求，并且能够轻松地查看和分析请求的响应结果。

Postman 的使用方法非常简单，主要步骤为：创建请求、设置请求 URL、添加请求头、设置请求体和发送请求并查看响应。

1) **创建请求**：打开 Postman 后，用户可以通过点击“+”来创建一个新的请求。在请求构建器中，首先需要选择请求方法，如 GET、POST、PUT、DELETE 等。这些请求方法对应了不同的操作，例如，GET 方法用于从服务器获取资源，POST 方法通常用于向服务器提交数据以创建新的资源。以测试一个简单的用户登录接口为例，若接口采用 POST 方法接收用户名和密码信息来验证用户身份，在 Postman 中就选择 POST 方法。
2) **设置请求 URL**：在选择好请求方法后，需要在 URL 栏中输入要请求的接口地址。这个地址应该是完整的，包括协议、域名和具体的路径。
3) **设置请求体**：如果是 POST、PUT 等需要发送数据的请求方法，就需要设置请求体。Postman 支持多种数据格式，如 form - data、x - www - form - urlencoded、raw。对于用户登录接口，若采用 JSON 格式发送用户名和密码，在请求体的 raw 选项中选择 JSON 格式，并填写类似 {"username": "testuser", "password": "testpassword"} 的数据。

3 项目测试过程

3.1 前端测试

前端测试我们主要从用户的角度去体验软件的各个功能是否能正常运行，因此我们使用黑盒测试对前端的各个页面进行测试。测试用例、测试结果以及预期结果如下图所示。为了方便展示，部分高度相似并且测试结果以及预期结果均相同的测试用例我将填写到同一个序号内。

序号	测试用例	测试值	测试结果	预期结果
1	注册功能，所有字段正常	用户名不和已存在的用户名冲突，密码字段正常	返回注册成功消息	返回注册成功消息
2	注册功能，所有字段正常	用户名和已存在的用户名冲突，密码字段正常	返回注册失败消息	返回注册失败消息
3	注册功能，用户名字段为空	用户名字段为空值，密码字段正常	返回注册失败消息	返回注册失败消息
4	注册功能，密码字段为空	密码字段为空值，用户名字段正常	返回注册成功消息	返回注册失败消息
5	注册功能，用户名字段超出限制	用户名字段为 20 位数字，密码字段正常	返回注册失败消息	返回注册失败消息
6	注册功能，密码字段超出限制	密码字段为 20 位数字，用户名字段正常	返回注册失败消息	返回注册失败消息
7	登录功能，不存在的用户	用户名字段为不存在的用户名，密码字段为存在的密码	返回登录失败消息	返回登录失败消息
8	登录功能，错误密码	用户名字段正确，密码随机错误密码	返回登录失败消息	返回登录失败消息
9	新增记录功能，默认值	所有字段保持默认	记录添加成功，金额为 0	记录添加成功，金额为 0
10	新增记录功能，随机值	所有字段随机组合	记录添加成功，所有字段被正确添加	记录添加成功，所有字段被正确添加
11	删除记录功能	无	删除成功	删除成功
12	修改记录功能，取消修改	默认值	无任何处理	无任何处理
13	修改记录功能，随机修改	所有字段随机组合（与原字段组合不相同）	修改成功	修改成功
14	日期筛选功能	随机选择任意年份和月份	记录被成功筛选	记录被成功筛选

15	统计功能	无	正确显示当月各种消费类型的金额	正确显示当月各种消费类型的金额
16	“我的”界面显示功能	无	正确显示	正确显示

图 1 前端图形化界面测试用例

3.2 后端测试

后端的 API 接口数量庞大，在有限的时间内无法对所有的 API 接口进行完备地测试。因此，我们将选择一些重要的接口进行测试。此外，我们后端与前端的交互使用了 Gin 框架，后端与数据库的交互使用了 gorm 框架，大多数沿用框架中的设计，一些特殊的用例无需再进行测试。

序号	测试用例	测试值	测试结果	预期结果
1	注册接口	输入合法的用户名、密码和确认密码	返回注册成功信息，且用户数据成功插入数据库	返回注册成功信息，且用户数据成功插入数据库
2	注册接口	已注册过的用户名，其他信息合法	返回用户名已存在的错误提示	返回用户名已存在的错误提示
3	注册接口	合法用户名，密码和确认密码，但是密码和确认密码不一致	返回密码不一致的错误提示	返回密码不一致的错误提示
4	注册接口	不符合格式要求的账号，合法密码和确认密码	返回账号格式错误的提示	返回账号格式错误的提示
5	登录接口	已注册用户的正确用户名和密码	返回登录成功信息，设置登录状态为已登录	返回登录成功信息，设置登录状态为已登录
6	登录接口	未注册的用户名和任意密码	返回用户名不存在的错误提示，登录状态保持未登录	返回用户名不存在的错误提示，登录状态保持未登录
7	登录接口	已注册用户的用户名和错误密码	返回密码错误的提示，登录状态保持未登录	返回密码错误的提示，登录状态保持未登录
8	新增记录接口	合法的收支类型、金额、分类、日期	返回新增记录成功信息	返回新增记录成功信息
9	新增记录接口	错误格式的金额，其他类型合法	返回新增记录成功信息	返回金额格式错误的提示

10	删除记录接口	已存在记录的 ID	返回删除记录成功信息	返回删除记录成功信息
11	删除记录接口	不存在的记录 ID	返回记录不存在的提示	返回记录不存在的提示
12	编辑记录接口	已存在记录的 ID，修改后的合法信息	返回删除记录成功信息和新增记录成功信息	返回编辑记录成功信息
13	查询记录接口	随机选择年份和月份	返回符合条件的数据记录	返回符合条件的数据记录
14	查询记录接口	默认筛选条件	返回当前月份的数据记录	返回当前月份的数据记录

图 2 后端重要接口测试用例

除了对上述接口测试以外，我们还对发送和接收消息的两个重要的函数进行单元测试。因为两个函数内部并没有选择分支，因此只需要一个测试用例即可覆盖函数内部的所有路径。我采取的测试方式为：向发送消息函数中传递随机长度和随机值的字节数组，由发送消息的函数发送给接收消息的函数，同时从接收消息函数中读取数据，将读取的数据与原数据进行对比。如果出现错误，则分别对两个函数进行单独的测试，否则说明两个函数可以正确地发送和接收消息。测试用例如下图所示。

序号	测试用例	测试值	测试结果	预期结果
1	消息发送	随机长度和随机值的字节数组	发送成功	发送成功
2	消息接收	无	接收成功，且接收数据反序列化后与原数据相同	接收成功，且接收数据反序列化后与原数据相同

图 3 后端重要函数测试用例

3.3 数据库测试

我们主要从两方面对数据展开测试。

- **数据完备性测试**：1）向数据库中插入各种类型的测试数据（包括正常数据和边界值数据），然后通过查询操作验证数据是否完整存储在相应的表中，且各个字段的值与插入数据一致。2）在一个事务中执行多个相关的数据操作，模拟事务过程中出现错误的情况，检查事务是否正确回滚，确保数据不会出现部分更新或插入的情况，保证数据的一致性。
- **数据准确性测试**：1）编写查询语句直接从数据库中获取数据，与前端展示的数据进行对比，验证数据库中存储的数据在经过各种业务操作后是否准确。2）对数据库中的索引进行测试，查询使用索引和不使用索引的情况下数据查询的效率，验证

索引是否能够正确加速查询操作，同时检查索引是否对数据更新、插入等操作产生负面影响，确保索引的正确性和有效性。

我们对用户表的测试用例如下。很多测试用例在软件中是不可能发生的，但是为了软件的每一个模块的健壮性，我添加了额外的测试用例。其中第 5 和第 6 个测试用例与预期并不相符，其原因是创建表字段时给用户名字段和密码字段设置了更大的空间，因此可以存放超出软件预定的长度限制的值。但是在前端我们会限制用户输入的最长长度，所以该测试结果与预期结果不符不会对软件造成影响。这两个测试用例只是为了更加完善地测试数据库。

序号	测试用例	测试值	测试结果	预期结果
1	添加用户	用户名字段为空值，密码字段正常	添加失败	添加失败
2	添加用户	用户名字段正常，密码字段为空值	添加失败	添加失败
3	添加用户	用户名字段与已有用户的用户名冲突	添加失败	添加失败
4	添加用户	用户名字段和密码字段均正常	添加成功	添加成功
5	添加用户	用户名字段超出长度限制	添加成功	添加失败
6	添加用户	密码字段超出长度限制	添加成功	添加失败
7	验证用户	用户名为空值	验证失败	验证失败
8	验证用户	密码为空值	验证失败	验证失败
9	验证用户	非已有用户的用户名字段	验证失败	验证失败
10	验证用户	已有用户	验证成功	验证成功

图 4 用户表测试用例

我们对记录表的测试用例如下。很多测试用例在软件中是不可能发生的（如添加的时候存在空值、查询时用户名或日期字段为空等），但是为了软件的每一个模块的健壮性，我添加了额外的测试用例。

序号	测试用例	测试值	测试结果	预期结果
1	添加记录	用户名字段为空，其余正常	添加失败	添加失败
2	添加记录	记录类型字段为空，其余正常	添加失败	添加失败
3	添加记录	记录类别字段为空，其余正常	添加失败	添加失败
4	添加记录	记录日期字段为空，其余正常	添加失败	添加失败
5	添加记录	记录金额字段为空，其余正常	添加失败	添加失败

6	添加记录	所有字段正常	添加成功	添加成功
7	删除记录	记录 id 字段不存在	删除失败	删除失败
8	删除记录	记录 id 字段存在	删除成功	删除成功
9	查询记录	用户名不存在，日期字段正常	查询失败	查询失败
10	查询记录	用户名为空值，日期字段正常	查询失败	查询失败
11	查询记录	日期字段为空值，用户名正常	查询失败	查询成功，返回所有用户相关的记录
12	查询记录	所有字段正常	查询成功	查询成功，返回筛选后的记录

图 5 记录表测试用例

上述测试中并未涉及修改记录的测试，因为在系统的设计中，修改记录是使用删除记录和添加记录两个功能组合实现的新功能，所以对于修改记录部分，我们会在后端部分完成后进行测试。