面向对象技术期末作业

张渊博

22214608

2023 年 5 月 22 日

目录

1 银行管理系统需求

1.1 支持所有操作都基于 web 操作

1.2 客户分类

1. 对所有的用户必须支持以下功能：开户、存款、取款、查询、转账、改密码、销户等七项工作
2. 存款、取款、查询、转账、改密码、销户操作必须要验证用户的登录名与密码
3. 对账户的操作都要产生日志。所有的数据都要永久化（存到硬盘）。系统重新启动时要能恢复数据。
4. 对于普通个人用户、个人贵宾用户 (VIP)，企业用户的操作：
   (a) 开户：调研银行管理系统需要存储的账户信息。
   开户时需要身份证号码，决定开的账户类型，存款的初始金额。银行生成用户的帐号，用户输入初始密码。系统给出开户的时间日志。
   (b) 存款：
   存款时需要给银行提供自己的账号，密码以及要存款的金额；银行系统反馈用户帐户的余额以及存款的日志。
   (c) 取款：
   取款时需要给银行提供自己的账号、密码以及要取的金额；系统判断是否有足够余额以供取款。取款后给出用户帐户的余额，产生取款日志。
   (d) 查询：
   查询时用户需要提供账号和密码；银行方面给出用户帐户的余额以及查询的日志。用户还可以查询一段时间所有的操作（流水）。
   (e) 转账：（对不同用户转账有限制，请看账户类型说明）
   用户转帐时需要判断是否能够满足转账条件。如满足，需要给银行提供转出帐户的账号、密码和用户名称及转入帐户的帐号和用户名称，用户也需要提供转账金额。系统方面将产生转账日志、转出用户帐户的余额等信息。

(f) 改密码：

用户到银行改密码时需要给银行提供身份证、帐号、以及密码，并且输入新的密码。系统方面将产生相关日志。

(g) 销户：

用户到银行销户时需要给银行提供身份证、帐号以及密码，并且把帐户的余额全部取出。系统方面将产生相关日志。

•

5. 不同类型用户的特别限制和权限：

(a) 所有的用户可开两种账户—活期和定期。用户的活期和定期账号之间可以转账

(b) 用户不能产生透支。

(c) VIP 用户的初始存款额（包括活期和定期）要求大于 100 万，余额要求不低于 10 万。如果余额低于 10 万，VIP 用户将给降级到普通个人用户。

(d) 转账只能在个人用户之间进行，或在企业用户之间进行。

(e) 普通个人用户只能接收转账转入，不能转钱到别人的账户。但普通个人用户可以在他自己的账号内转账。VIP 和企业用户容许转入、转出。

(f) 企业用户容许最多 5 个账号操作人。指定其中的一人作为账户超级用户，负责增删账户操作人。不同账户操作人必须使用不同密码

(g) 企业用户要求账面存款余额总数（包括所有的账户）不小于 1 万元.

•

•

1.3 银行雇员分类

银行雇员分类: 前台操作员，银行经理，银行业务总管、系统管理员。

1.3.1 他们各自的职责:

系统必须能存储所有的雇员信息。银行经理管理一个或多个前台操作员，同时他可以做所有的前台操作员操作。银行业务总管管理一个或多个银行经理，他可以做所有的银行前台操作员及银行经理的工作。

银行雇员操作:

1. 前台操作员：对自己登陆账号的管理，对各类客户账户除了修改密码以外的其它 6 种操作。雇员本身所作的操作的日报。前台操作员向一个银行经理报告。

2. 银行经理：作为前台操作员的所有操作，部门操作的日报。银行经理必须较容易查到他部门人员总数及具体情况。银行经理向银行业务总管报告。

3. 银行业务总管（包括个人账户和企业账户两类）：可完成作为前台操作员和部门经理的所有操作，可查整个银行分管的相关业务操作的日报。银行业务总管应较容易查到他所管人员总数及具体情况。

4. 系统管理员：雇员管理。产生整个银行的总报表和总操作日志。

5. 所有银行雇员账号的管理必须经过其系统管理员审核。

•

2 设计文档

2.1 简介

本项目为银行管理系统。整个系统采用 Spring 框架与技术，后端依据三层架构来设计。后端采用 Java 语言设计，前端页面使用 Javascript 和 Thymeleaf 设计。其中后端采用注解的方式来配置整个系统。所有操作都基于 Web 操作。

本文档的设计目的是为了更好地使使用者及开发人员维护、使用该系统，文档的内容范围仅在本项目系统中，不包含 Spring 框架的使用介绍（假设已是读者的基础知识）。

2.2 系统概述

本系统由 3-Tier 架构实现，三层分别是 Data Access Layer, Business Logic Layer, Presentation Layer，另有一个 Model 组件，以更好地使三层交互。每一层仅与其相邻的层交互。日志的记录使用面向切面编程的思想实现，使用的框架为 Spring AOP，部分日志功能在继承树中实现。

本系统功能如需求 “银行管理系统需求” 所述。

2.3 架构设计

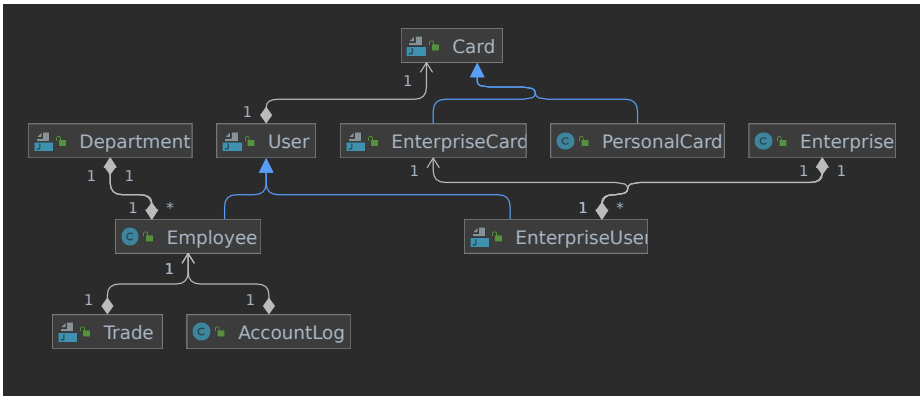
三层架构中：

- Data Access Layer 由 JPA 的 CrudRepository 接口实现；
- Business Logic Layer 的核心抽象是 Account, UserAccount 持有对应的 User 和 Card，功能的各种实现通过 UserAccount 中的实例方法借由操作 User 和 Card 实现，其中 Card 对用户不可见，抽象出账户、银行卡、用户（其中银行卡对用户不可见）。其中账户持有用户、银行卡，用户拥有银行卡。账户是对银行卡和用户的组织，而银行卡和用户是实体。通过将用户和银行卡与账户分离，将用户的密码存储在用户实体中，而不保存在银行卡中，可以实现多人使用不同密码而使用同一银行卡的功能（针对企业用户的功能）；
- Presentation Layer 由 Spring 框架的 Controller 控制，前端页面使用 Thymeleaf、html 实现。
-

UserAccount 和 EmployeeAccount 的设计都使用了面向接口设计的思想，下面图中包含 Right 后缀的均为接口，绿色的虚线为接口的实现，绿色的实现为接口的继承，蓝色的实线为继承关系。为了更好地复用代码，在 UserAccount 的基类中实现了开户、存款、取款、查询、转账、改密码、销户；EmployeeAccount 的基类中实现了查询所管雇员、查询流水，它们的子类根据需要进行重写。

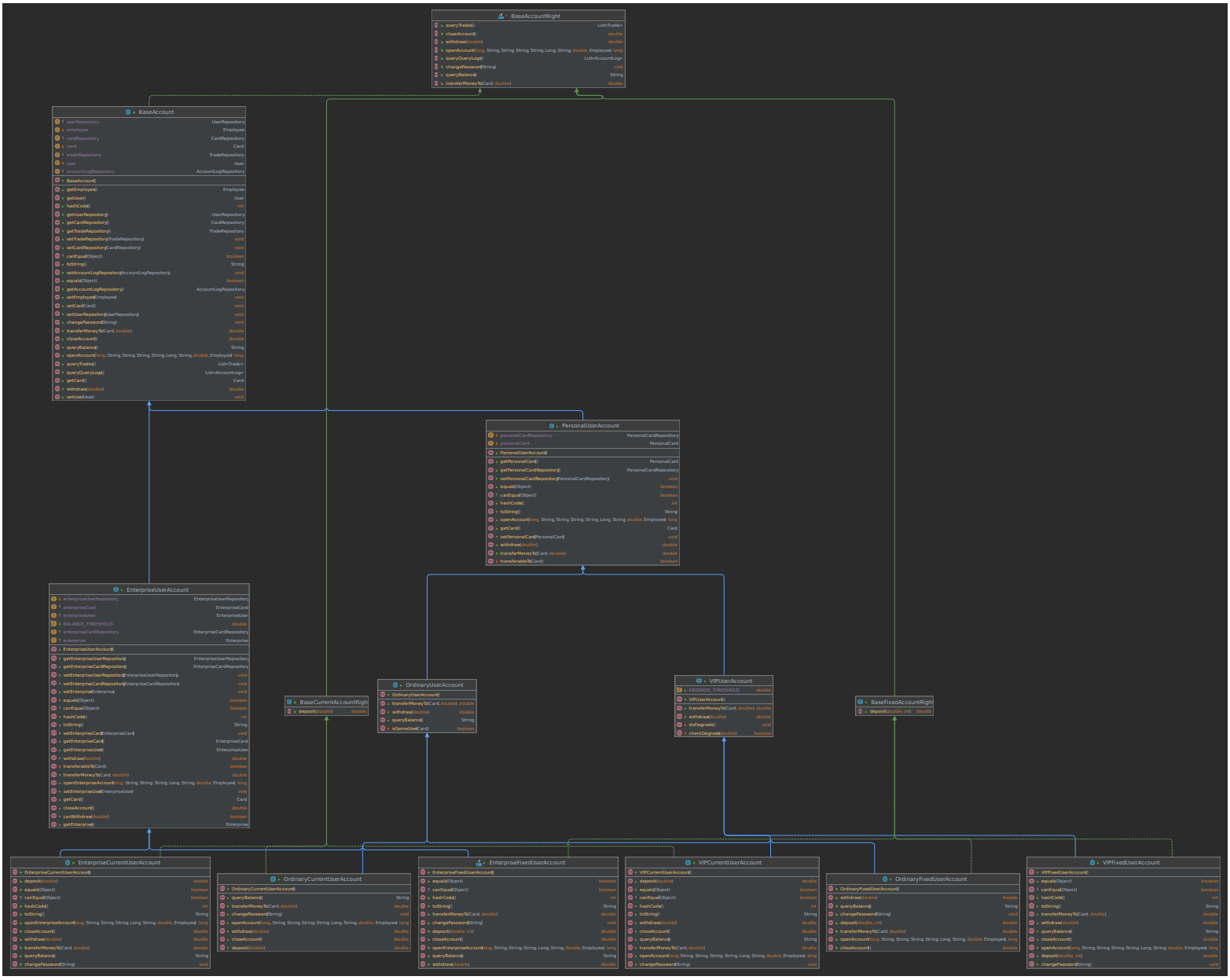
2.3.1 Model 组件的 UML 图

com.example.banksys.model 包的 UML 图。



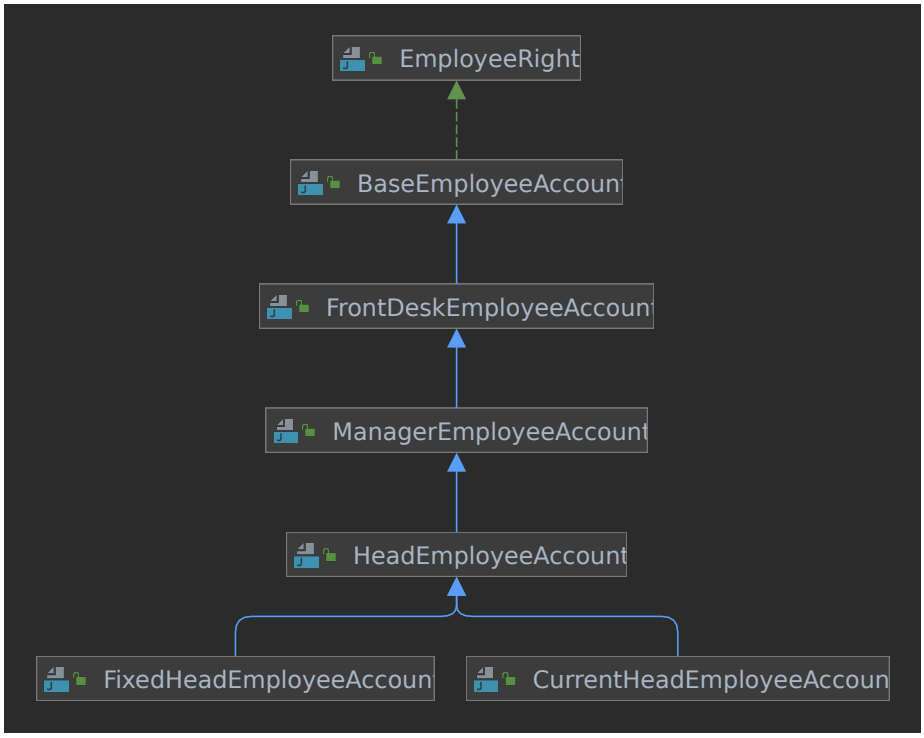
2.3.2 UserAccount 的 UML 图

com.example.banksys.businesslogiclayer.useraccount 包的 UML 图。



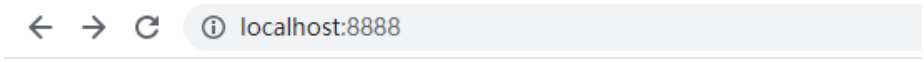
2.3.3 EmployeeAccount 的 UML 图

com.example.banksys.businesslogiclayer.employeeaccount 包的 UML 图。



2.4 UI 设计

整个 UI 界面并不是非常的精美，仅实现了最基本的功能。下图为系统首页：



银行系统分流页面

查询账户个人信息

查询账户个人信息

个人用户

前往个人用户界面

企业用户登录及开户

企业用户登录
企业用户开户

银行雇员登录及注册

雇员登录
雇员注册

系统管理员登录

管理员登录

退出当前账户

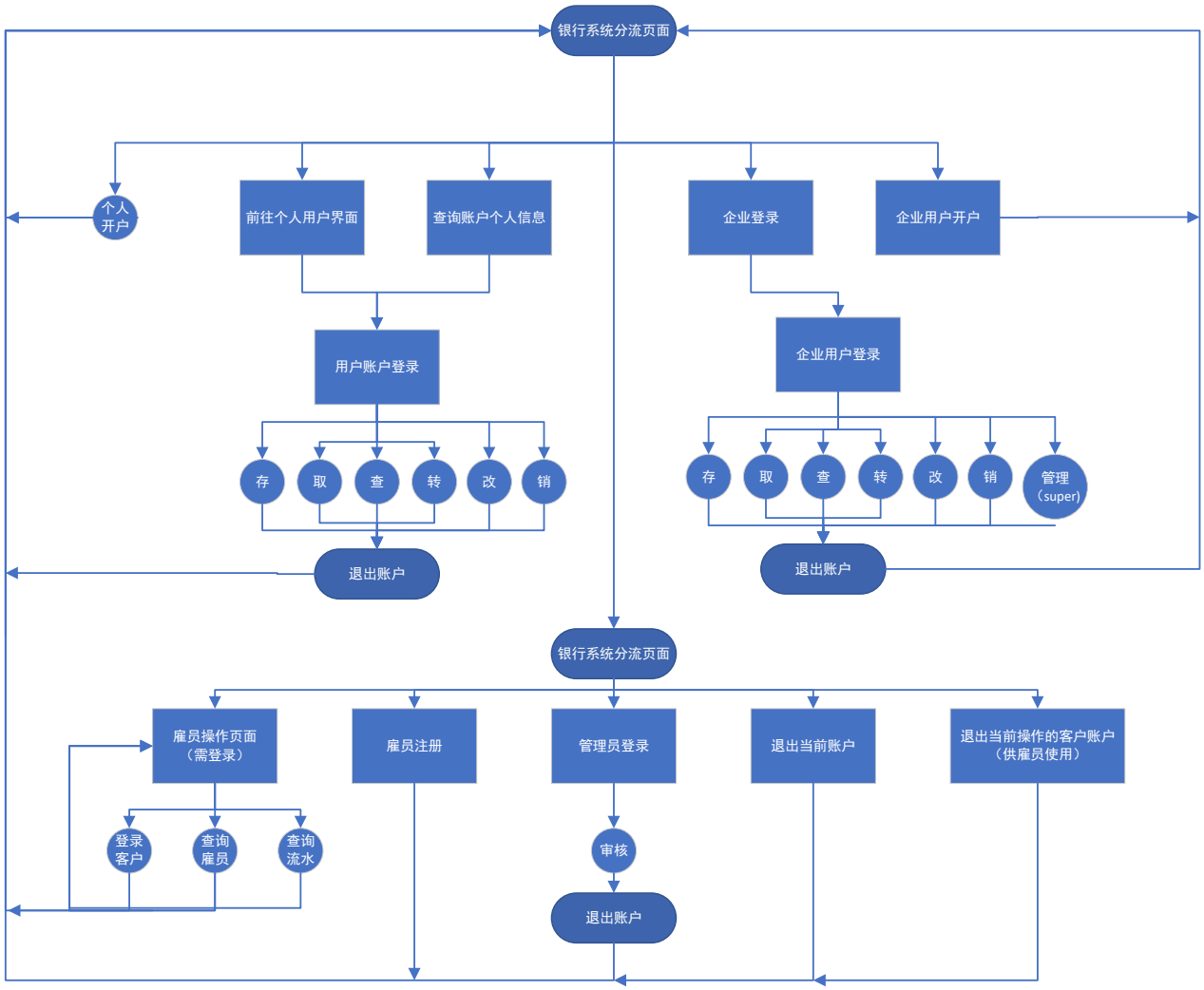
退出当前账户

退出当前操作的客户账户（供雇员使用）

退出当前操作的客户账户

2.4.1 网页业务流程设计

放大更清晰



2.5 数据设计

底层数据库使用 MySql，在 application.yml 中配置，使用的数据接口为 Hibernate，实现类使用的是 JPA。使用 MySql 实现了需求中的数据持久化、可恢复的要求。

2.6 组件设计

2.6.1 model 包

Card 由 PersonalCard 和 EnterpriseCard 继承。User 类本身为个人用户，其子类 EnterpriseUser 类为企业用户、Employee 类为雇员用户，区分的原因是企业用户有企业、权限（super 或者 user) 属性，而雇员用户有雇员类型、部门、有效（enabled，供系统管理审核功能使用）等属性。其他类顾名思义，是一些简单的实体类。

2.6.2 dataaccesslayer 包

该包是各种实体类的 Repository 的集合。

2.6.3 businesslogiclayer 包

- aop 包。是各种银行账户操作的日志记录切面类的集合。
- employeeaccount 包。EmployeeRight 为雇员权力的接口，BaseEmployeeAccount 为各种类型雇员的基类，包括查询所管雇员、查询流水的功能。类之间的继承关系如图 “EmployeeAccount 的 UML 图” 所示。雇员账户持有一个雇员，采用的是装饰者设计模式。
- exception 包。包括企业用户开户不足门限异常、企业已开户异常、企业取款后金额不足门限异常、企业账户开够五个异常、无法转账异常以及 VIP 开户不足门限异常。这些异常分别对应需求中的企业需有 10000 元开户门限金额，企业只能开一个户（银行卡，管理的是同一份钱），企业取款后金额需要大于 10000 元，企业最多只能有五个操作人，个人用户只能在个人用户之间转账、企业用户只能在企业之间进行转账且个人用户中的普通用户不能向其他人转账等。
- servcie 包。UserService 为接口，BaseUserService 为个人和企业用户 Service 的基类，负责 UserAccount 的各种操作进行进一步的封装；EmployeeService 为接口，EmployeeServiceImpl 为其实现类，负责对 EmployeeAccount 的各种操作进行进一步封装，以供 Presentation Layer 层调用。

- useraccount 包。UML 图如 “UserAccount 的 UML 图” 所示。UserAccount 持有对应的 User 和 Card，功能的各种实现通过 UserAccount 中的实例方法借由操作 User 和 Card 实现，其中 Card 对用户不可见，抽象出账户、银行卡、用户（其中银行卡对用户不可见）。其中账户持有用户、银行卡，用户拥有银行卡。账户是对银行卡和用户的组织，而银行卡和用户是实体。通过将用户和银行卡与账户分离，将用户的密码存储在用户实体中，而不保存在银行卡中，可以实现多人使用不同密码而使用同一银行卡的功能（针对企业用户的功能）。
- util 包。仅包含一个 BLLUtil 类，是一个工具类，是对一些难以使用继承来复用的方法的集合，以达到更好地复用代码，例如查询定期账户的可取余额、定期账户的存款和取款方法等。

•

2.6.4 presentationlayer 包

- controller 包。是各种 Controller 类的集合。
- form 包。相当于协议的集合，是各种 post 方法对应的表单的集合。
- helper 包。对一些难以使用继承来复用的方法的集合。
- utils 包。同上，也是对一些难以使用继承来复用的方法的集合。但其中还包括一些注解的定义，例如密码与确认密码的匹配注解，账户 ID 与账户身份证号码的匹配注解、账户 ID 与账户姓名的匹配注解等，以及 @Validated 注解中验证的顺序注解。

•

2.6.5 config 包

config 包中是一些配置类，包括 Bean 的配置以及 Spring Security 的配置，以及自定义的一个 SwitchUserFiler，以供雇员登录用户时验证用户的账户 ID 和密码。

2.6.6 更详细的 API 文档

更详细的 API 文档在 “22214608 张渊博 OOM final/doc/index.html”

2.7 安全设计

本文的系统使用 Spring Security 框架对系统的安全行进行保障，密码在持久化中也是使用的加密后的密码。数据库有对应的用户名和密码，可以保障数据安全。

3 测试报告

分别对 UserAccount、EmployeeAccount、Controller、工具类、帮助类进行测试，不仅使用了单元测试，还使用了端到端的测试。覆盖面较广，几乎每一个应该使用的方法都经过了测试，但是没能在最开始的时候使用 IDEA 的 run with Coverage 功能，所以数据显示比较低。如下图的测试报告：

52% classes, 35% lines covered in package 'com.example.banksys'

Element	Class, %	Method, %	Line, %
businesslogiclayer	54% (20/37)	26% (53/198)	34% (190/543)
config	100% (4/4)	28% (6/21)	58% (49/84)
dataaccesslayer	100% (0/0)	100% (0/0)	100% (0/0)
model	35% (6/17)	15% (17/113)	11% (25/227)
presentationlayer	53% (26/49)	41% (83/200)	43% (222/506)
BankSysApplication	100% (1/1)	0% (0/1)	50% (1/2)

4 运行报告

整体上，需求中的基本功能都可以很好地运行。但是在同一次会话中，切换不同用户登录时，可能在 UI 中设置的不方便，需要在首页手动退出，否则在使用其他用户权限的功能时会报 403 权限错误，或者有时会出现其他 BUG（在前端页面中显示未知错误）。

下面为运行时的部分截图展示：

图 1: 开户

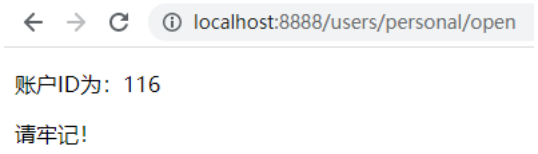


图 2: 开户成功

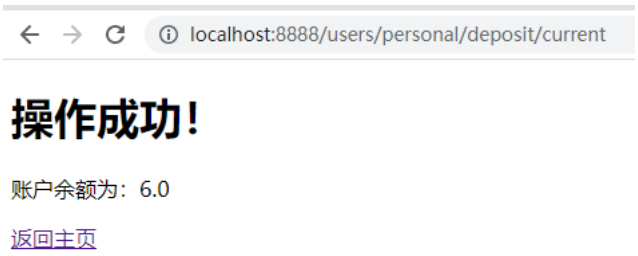


图 3: 存款成功

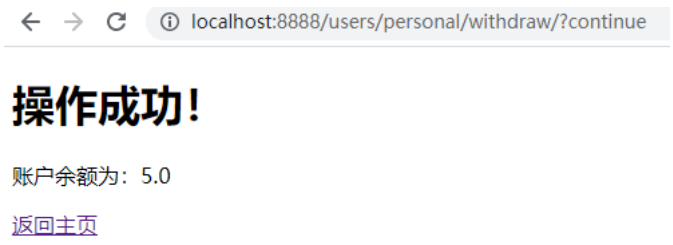


图 4: 取款成功

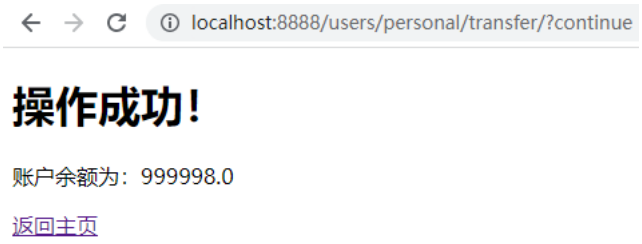


图 5: 转账成功

图 6: 转账失败


localhost:8888/users/personal/query/?continue
可取余额/总余额: 5.0/5.0
查询日志:
日期 描述 雇员ID
2023-05-18 16:21:00.552 可取余额/总余额：0.0/0.0元
2023-05-18 16:00:35.915 可取余额/总余额：0.0/0.0元
2023-05-18 15:47:29.493 可取余额/总余额：0.0/0.0元
2023-05-18 15:47:28.266 可取余额/总余额：0.0/0.0元
2023-05-18 15:43:43.969 可取余额/总余额：0.0/0.0元
2023-05-18 15:41:20.974 可取余额/总余额：0.0/0.0元
2023-05-18 15:40:44.273 可取余额/总余额：0.0/0.0元
2023-05-18 15:40:36.203 可取余额/总余额：0.0/0.0元
2023-05-18 15:40:34.698 可取余额/总余额：0.0/0.0元
2023-05-18 15:39:22.25 可取余额/总余额：0.0/0.0元
2023-05-18 15:39:21.36 可取余额/总余额：0.0/0.0元
2023-05-18 15:36:32.622 可取余额/总余额：0.0/0.0元
交易日志:
交易日期 交易类型 交易金额
2023-05-22 20:34:48.513 withdraw 1.0
2023-05-22 20:34:28.98 current 1.0

图 7: 查询成功

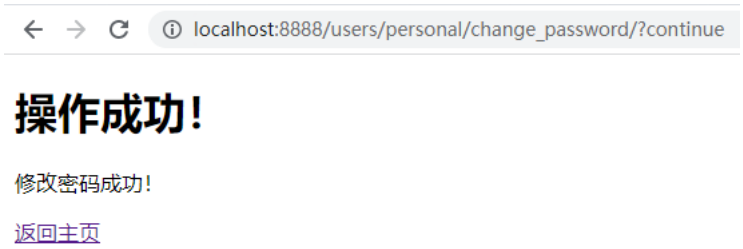


图 8: 修改密码成功

图 9: 销户成功

查询流水

所在部门：部门1

用户ID	雇员ID	操作类型	描述	时间
53	102	活期存款	存款金额：1.0元 存前余额：1.0元 存后余额：2.0元	2023-05-20 17:01:49.381
53	102	活期存款	存款金额：1.0元 存前余额：2.0元 存后余额：3.0元	2023-05-20 17:03:58.706
53	102	活期存款	存款金额：1.0元 存前余额：3.0元 存后余额：4.0元	2023-05-20 17:23:21.991

图 10: 查询流水

图 11: 查询所管雇员

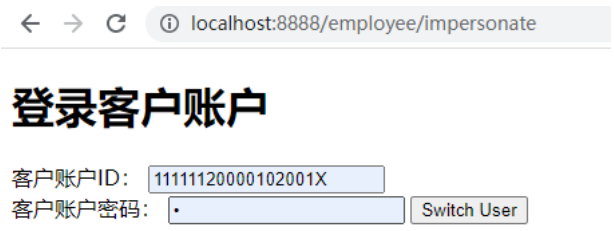


图 12: 雇员登录客户账户