# 详细设计

# 概述

## 1.1 系统简述

本平台将定位在中大型企业内，面向企业员工提供小额信贷撮合的信息中介服务，采用稳健的线上无抵押无担保的模式，主要解决员工投资和借款的需求，平台以服务员工为第一目标，不收取任何服务费，费用直接在借款人和投资人直接发生。

## 1.2 软件设计目标

1. 界面友好，易操作；
2. 稳定性好；
3. 可靠、安全。

## 1.3 参考资料

《Applying UML and Patterns》

《Software Architecture in Practice》

## 1.4 修订版本记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 说明 | 修改人 |
| 2019-06-18 | 1.0 | 初稿 | 钟宇轩 |
| 2019-06-19 | 2.0 | 更新界面设计和类图，增加补充内容 | 钟宇轩 |
| 2019-06-20 | 2.1 | 补充内容 |  |

# 术语表

（对本文档中所使用的各种术语进行说明。如果一些术语在需求规格说明书中已经说明过了，此处不用再重复，可以指引读者参考需求说明。）

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

# 设计概述

## 3.1 系统的复用计划

暂无复用计划

## 3.2 系统接口设计

#### 3.2.1界面设计

3.2.1.1 登录界面：

登录界面提供账号密码输入栏，登录形式包含普通用户，担保人，管理员三种形式；并设计对应按钮执行登陆功能或转入注册页面。



3.2.1.2 注册界面：

用户可以在注册界面输入注册账号的相关信息，例如账号、密码、确认密码，手机号以及验证码。用户需要输入系统自动发出的手机验证码验证通过后才能注册成功。系统在执行成功注册后返回登陆界面



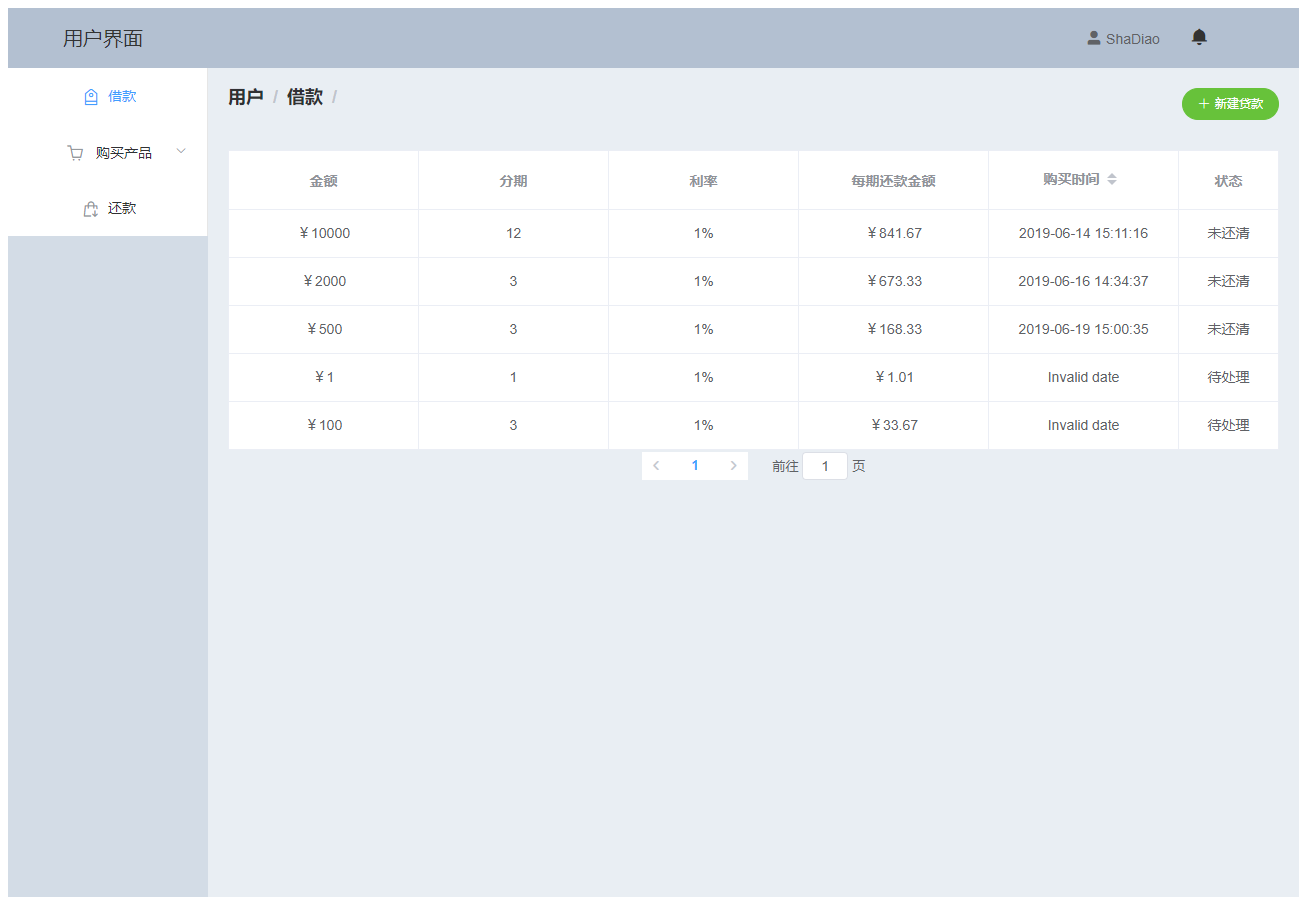
3.2.1.3 普通用户主页面：

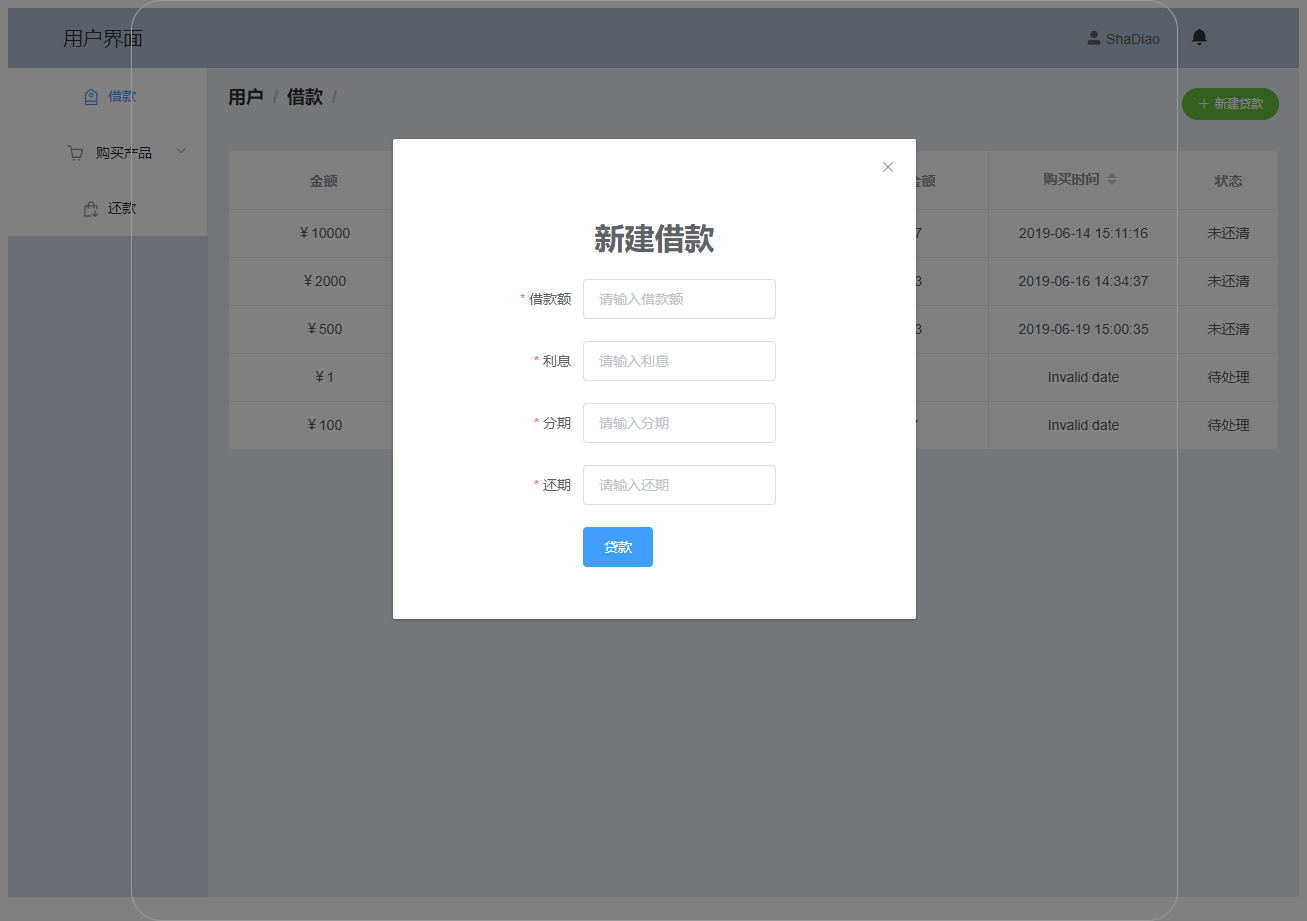
普通用户主页面由顶部功能区以及侧边的功能区组成，用户可以通过侧边的功能区的按钮在三个子页面间进行切换以使用所有主要功能。顶部功能包含用户头像图标、个人消息栏。通过鼠标移动到用户头像图标，页面会显示用户信息修改，充值，提现，登出的功能按钮，点击相应功能按钮可以实现相关跳转或弹窗。



1. 借款子页面

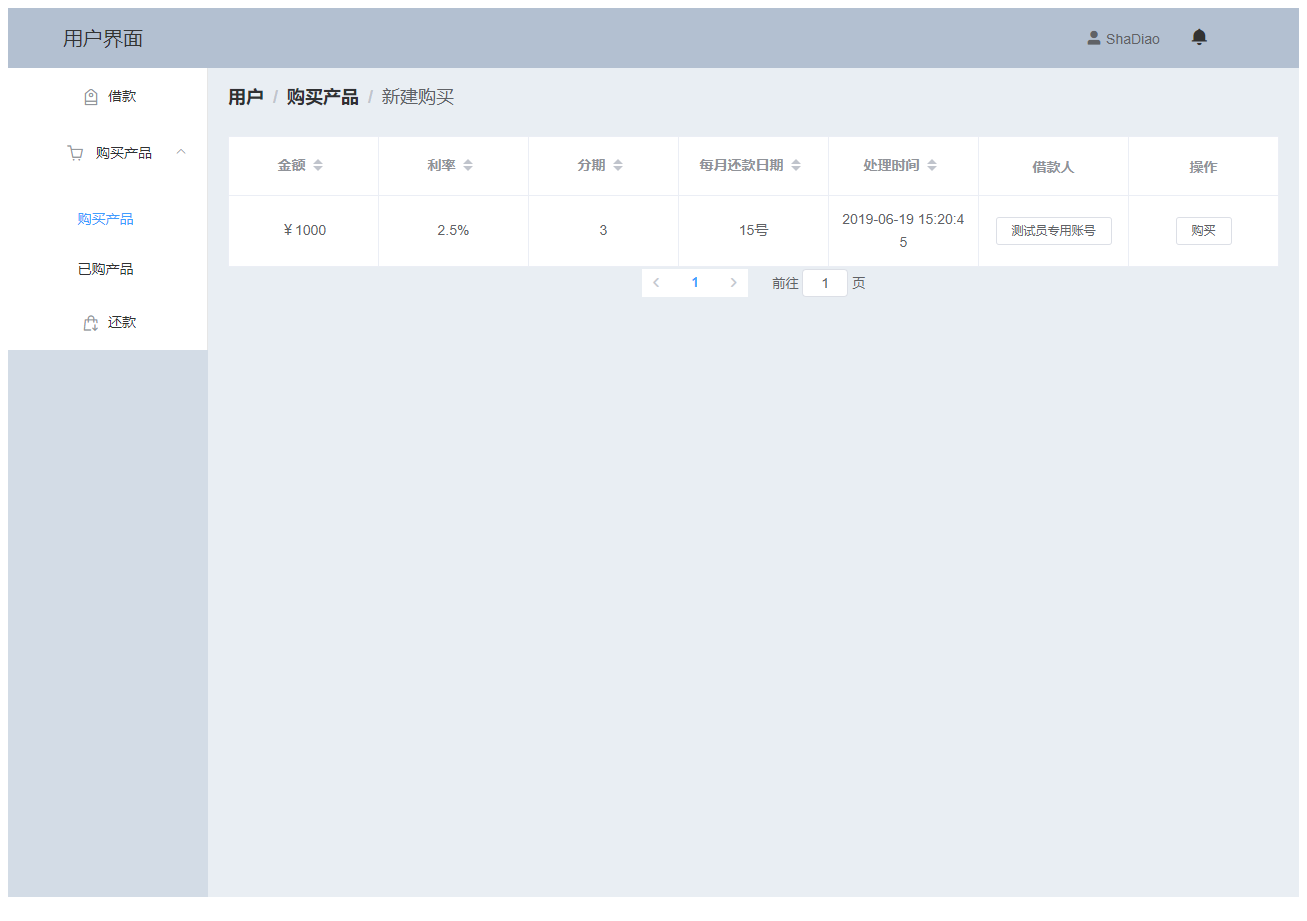
借款子页面会显示该用户当前借款信息，包括：借款金额，分期，利率，每期还款金额，提交时间，该笔借款的状态。同时还能在该页面左上角点击打新建借款按钮进行新的借款操作。

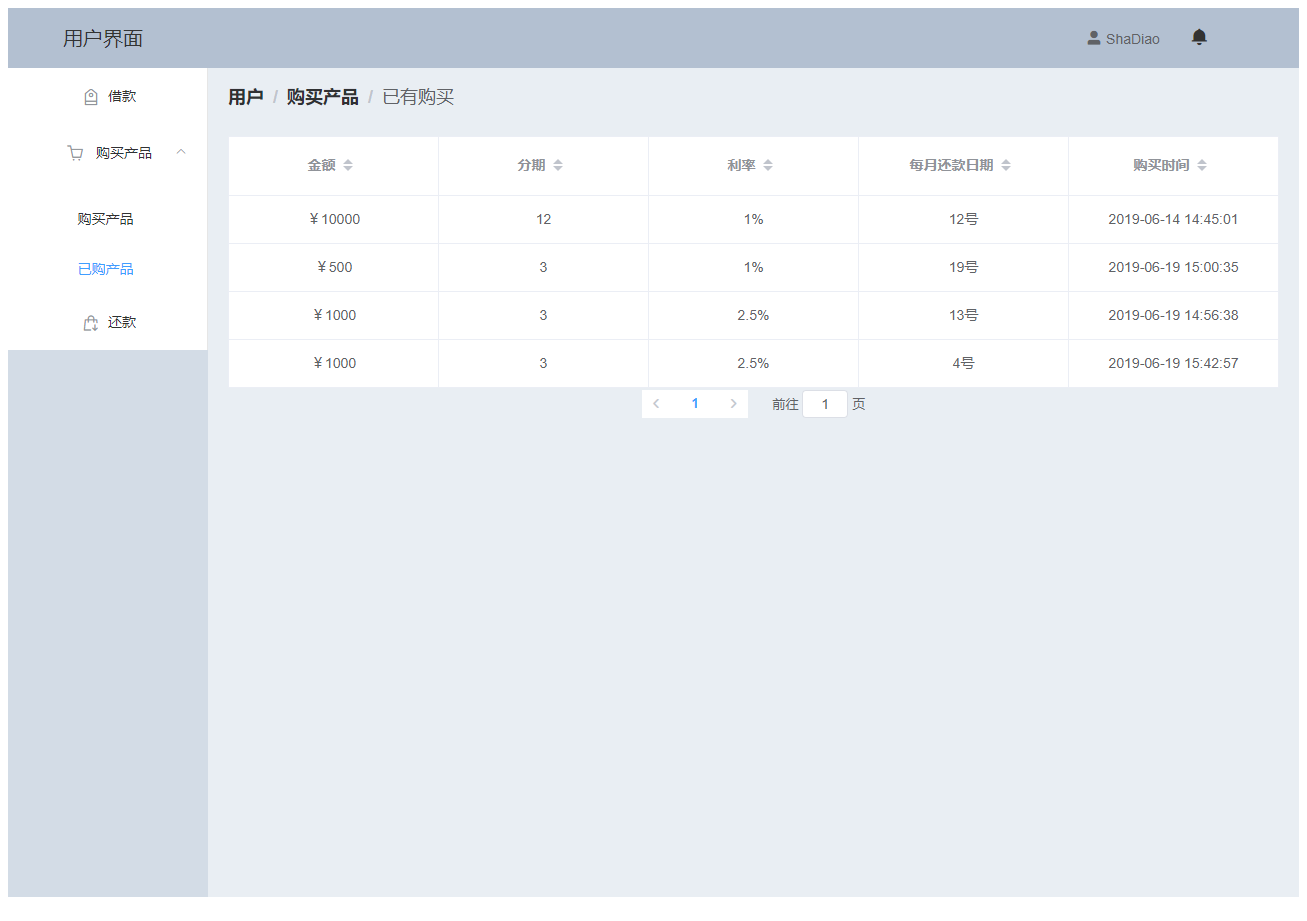




1. 购买产品子页面

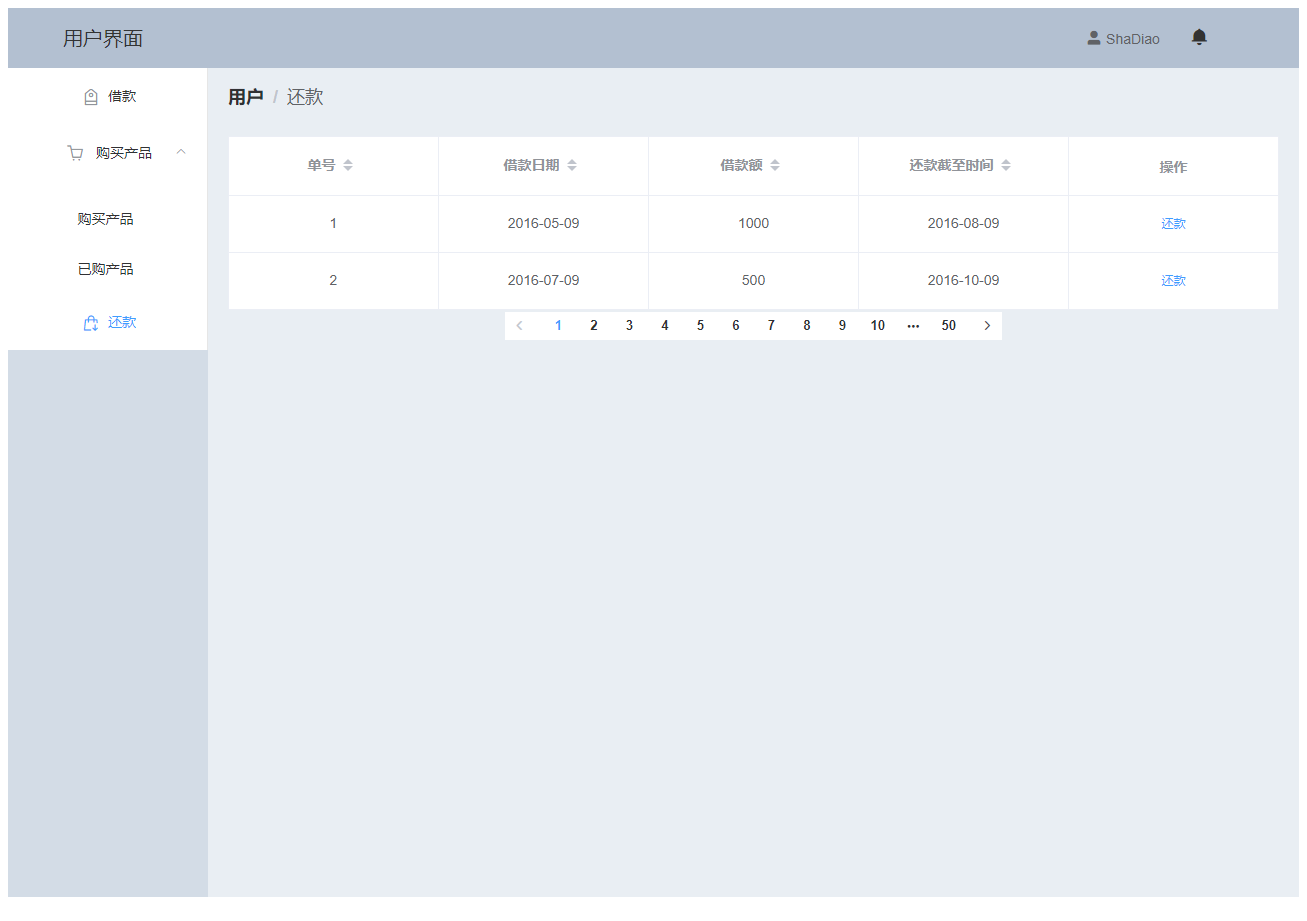
购买产品子页面显示当前平台发布的相关借款产品及其相关信息，包括金额，分期，利率，每月还款日期，处理时间，处理时间，产品发布人姓名，最后提供操作按钮，可进行选择购买。同时也可以显示用户已购买的产品信息。





1. 还款子页面

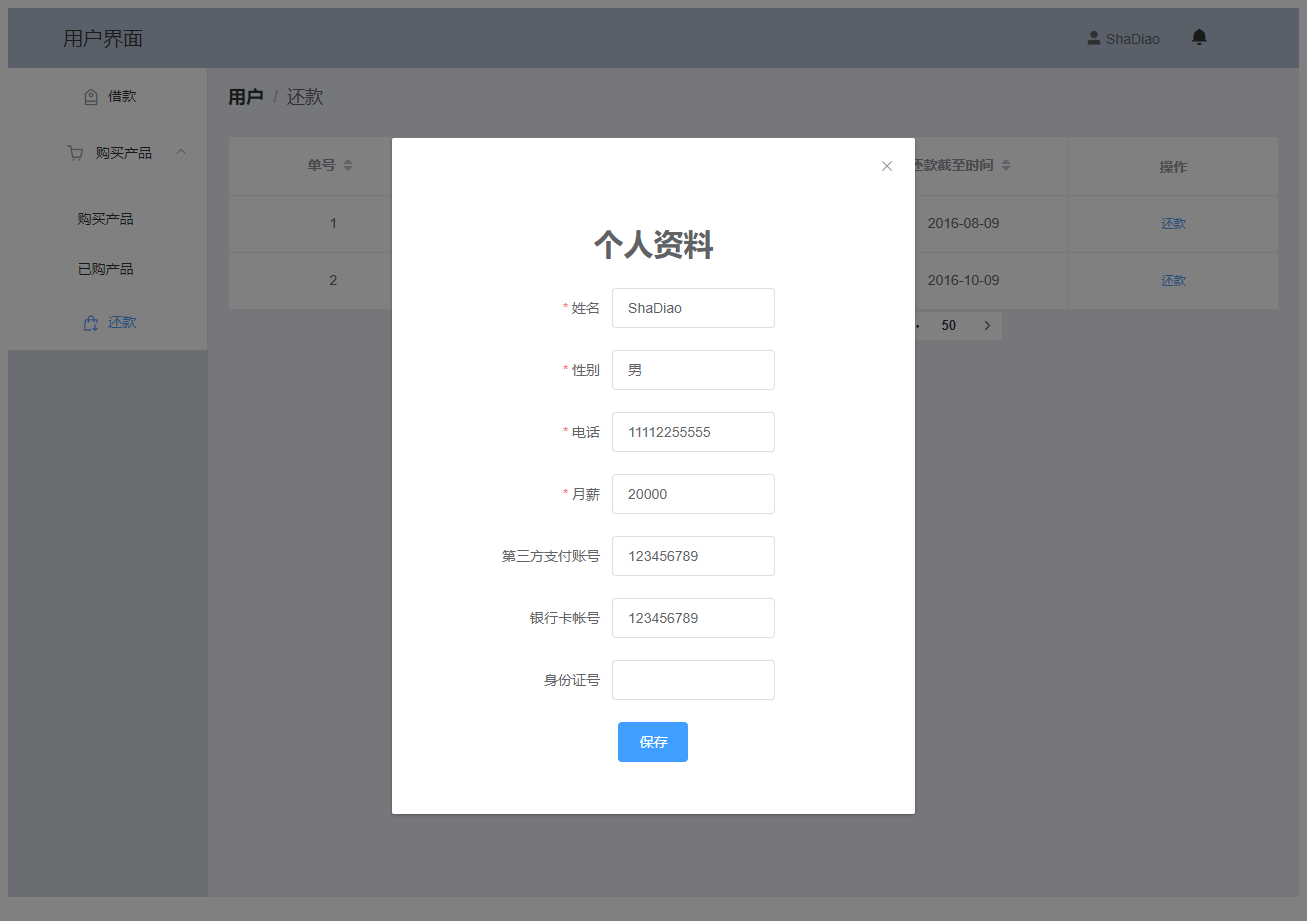
还款子页面供当前用户进行还款操作。



3.2.1.4 用户图标子页面：

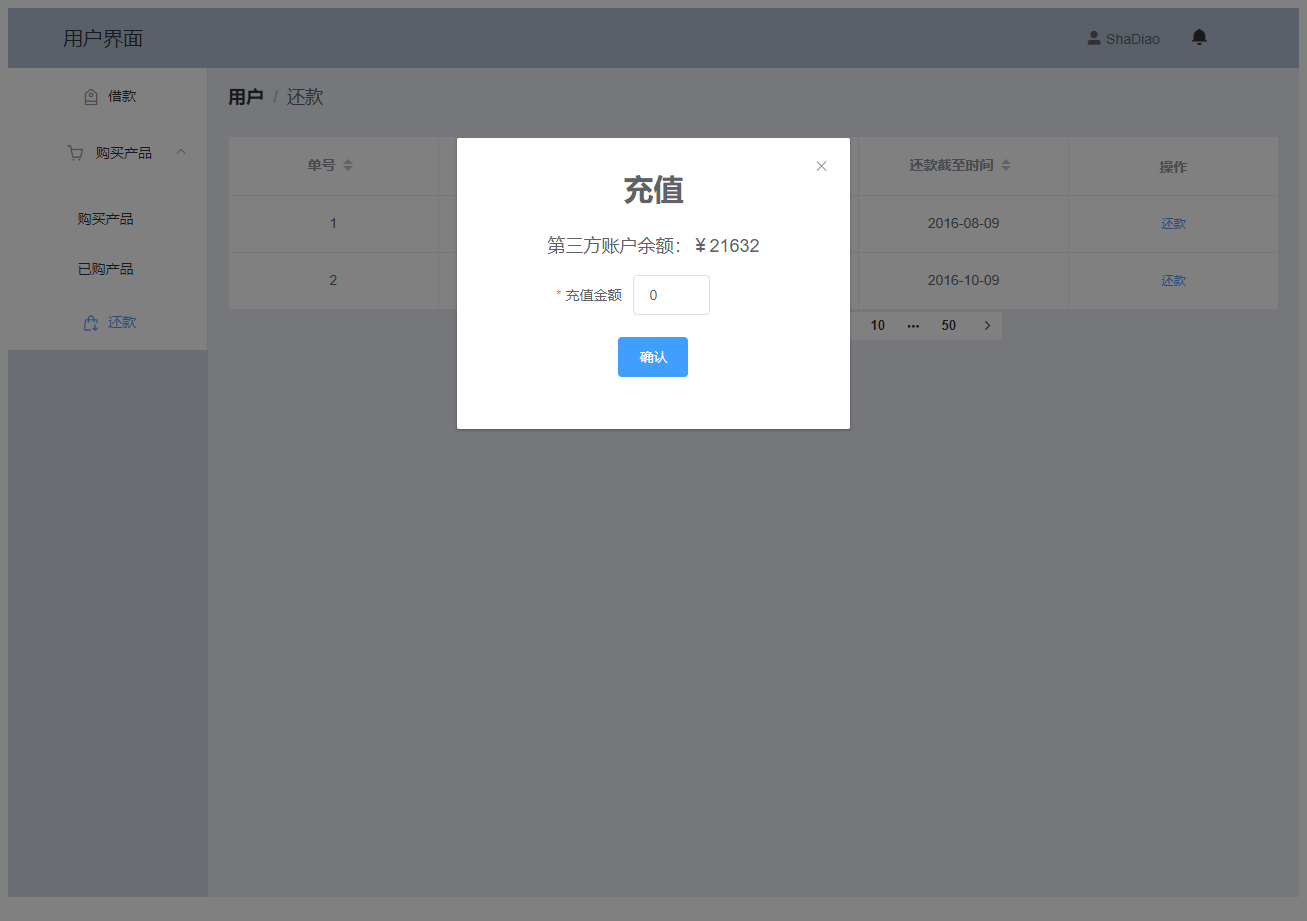
1. 个人资料子页面

用户可以在个人资料子页面中进行个人信息资料的修改。



2） 充值子页面

用户可以在该子页面中进行第三方平台账户的充值。同时该页面会显示当前用户第三方账户的余额。



3）提现子页面

用户可以在该子页面中进行从第三方账户余额到个人绑定银行卡的提现操作。



3.2.1.5 担保人主页面：

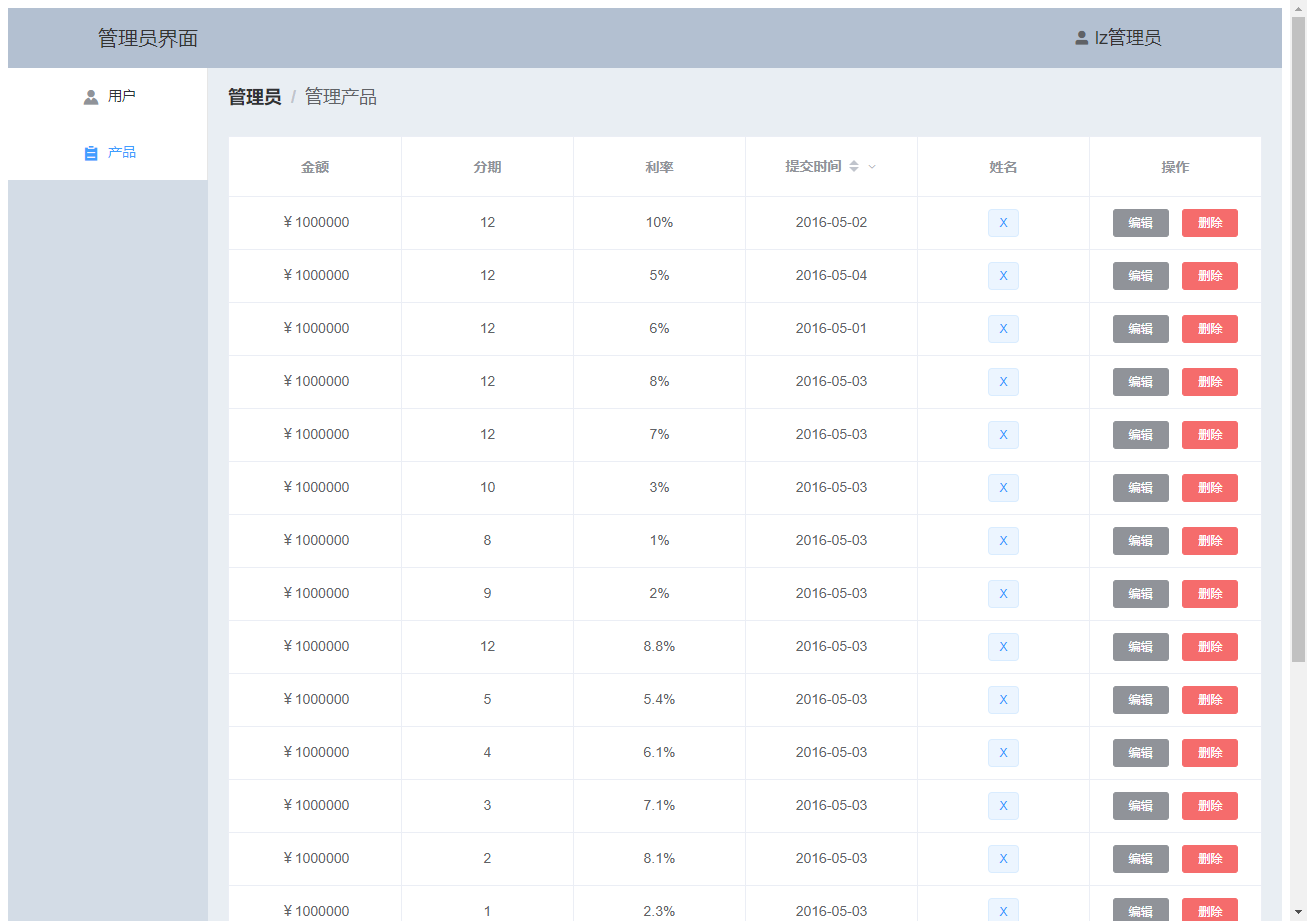
担保人登录后可在该页面对已提交借款申请的申请单进行审核。该界面显示各个申请的相关信息。在操作栏处可已进行同意或否决操作。



3.2.1.6 管理员主页面：

管理员登录后可在该页面对平台上的用户和产品进行操作。





#### 3.2.2 系统外部接口设计：与外部系统的交互设计

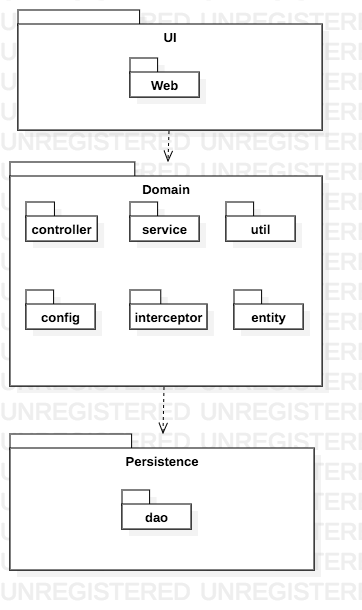
系统与外部系统暂无交互

#### 3.2.3 系统内部接口设计：各子系统，各模块间的接口设计

内部接口设计见下文对象模型详述

## 3.3 对象模型设计

#### 3.3.1 系统层级设计



系统层级设计如图，主要分为3层：UI层、领域层与持久层。

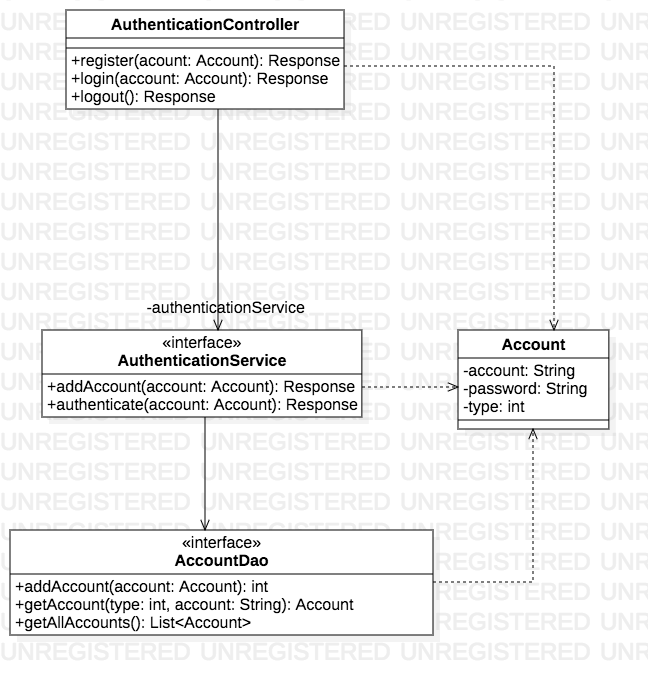
UI层主要通过以Http实现的Web通信功能完成客户端与服务器的交流。并通过基于vue实现的UI合理地将信息呈现给用户。

领域层揭示了本系统的功能核心，这一层分成六个包，controller、service、util、config、interceptor和entity并实现一定程度的错误处理功能。

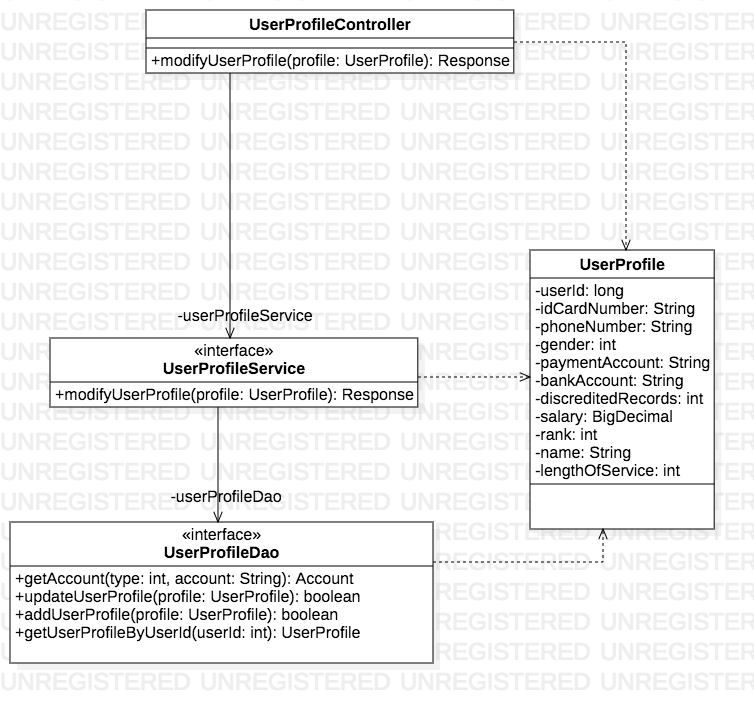
持久层主要包含信息的持久保持与为了实现entity和service与数据库的交互，这一部分的实现主要依托于所使用的数据库技术与服务器端技术。

#### 3.3.2 类图设计

3.3.2.1 认证模块



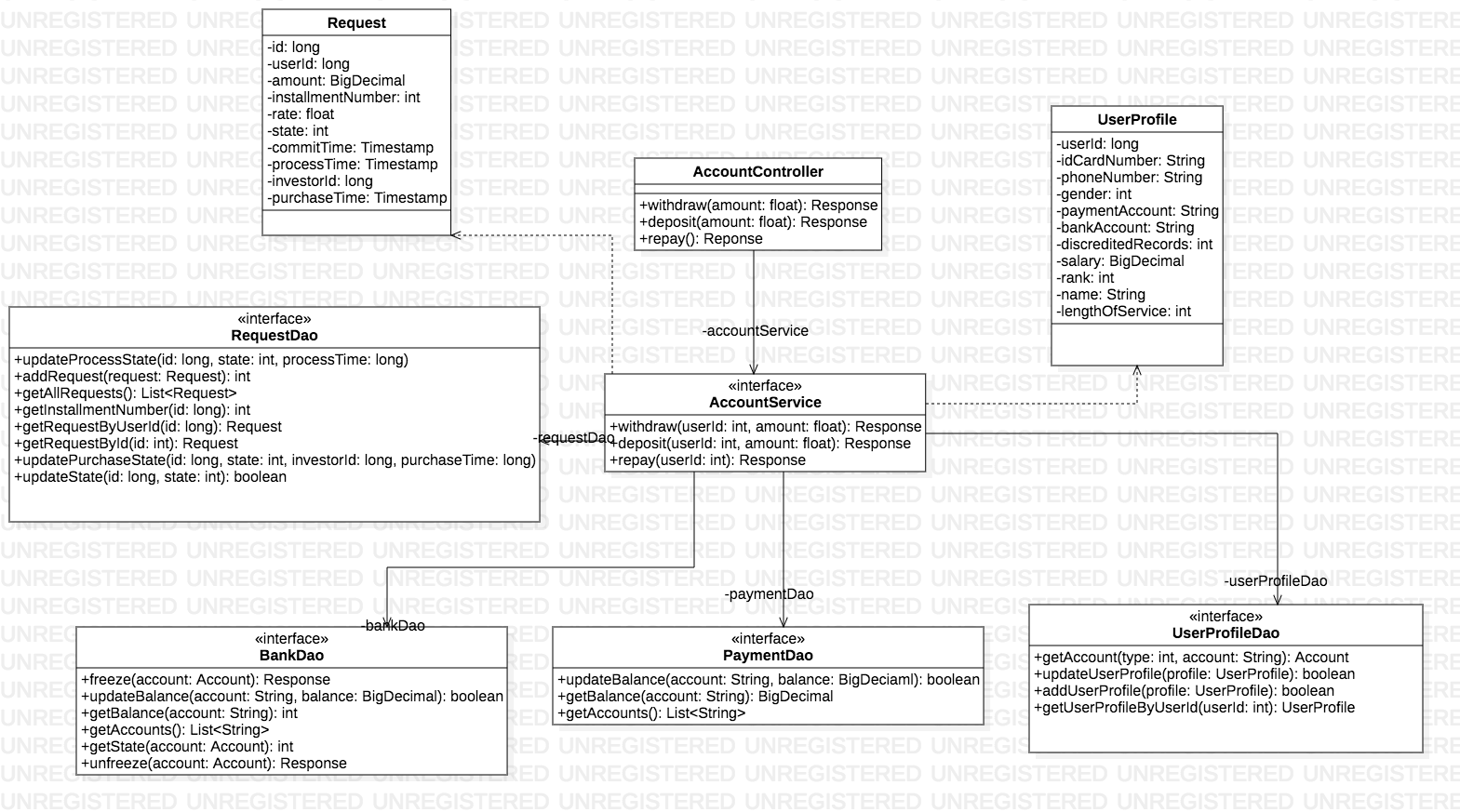
3.3.2.2 用户资料模块



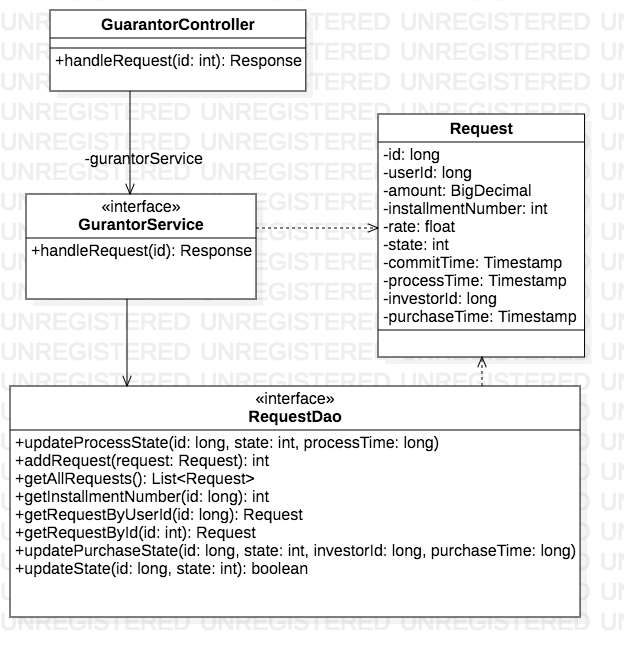
3.3.2.3 借款人模块



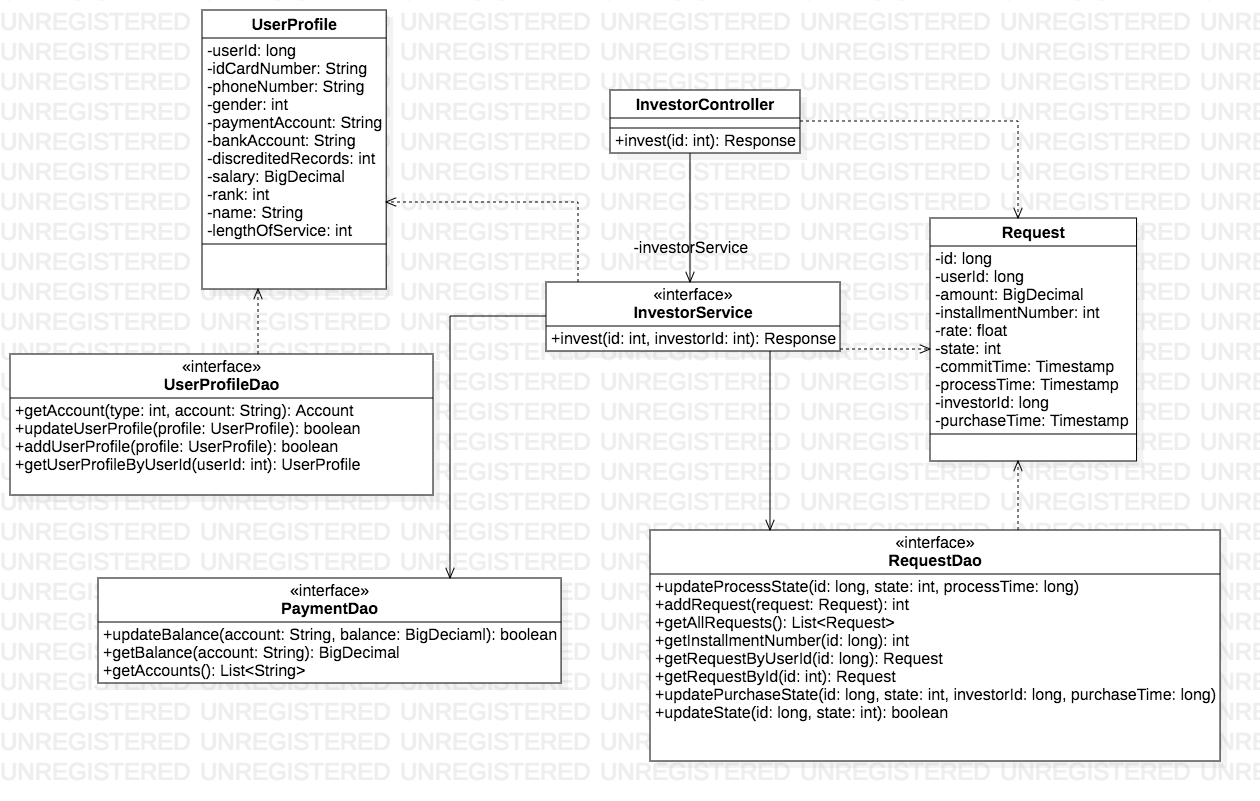
3.3.2.4账号模块



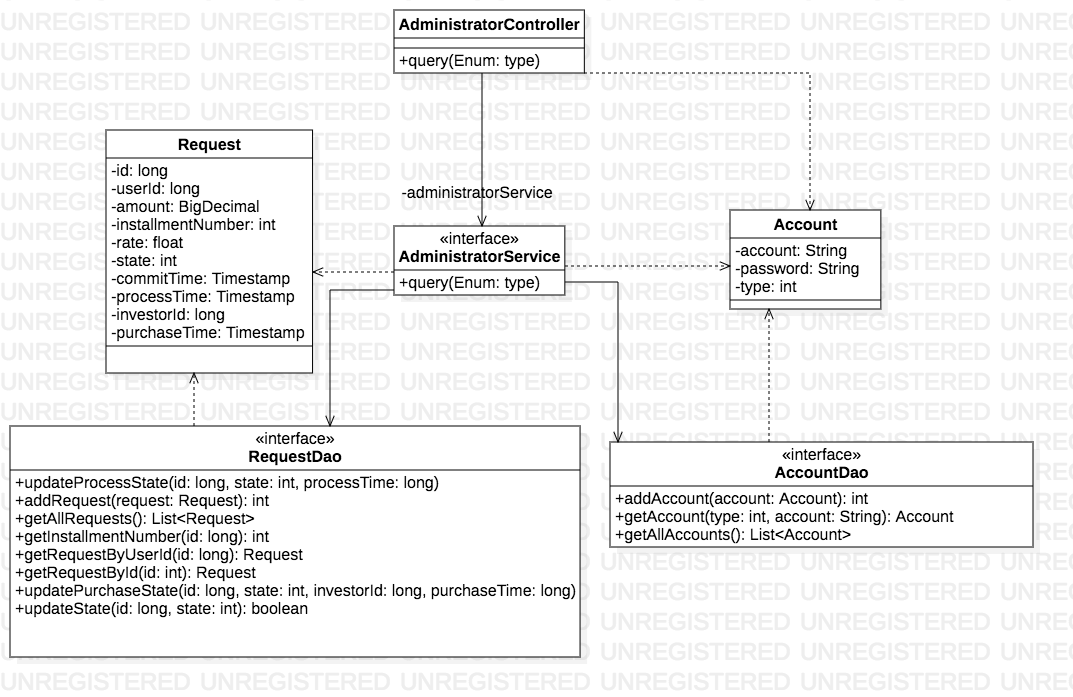
3.3.2.5担保人模块



3.3.2.6投资者模块



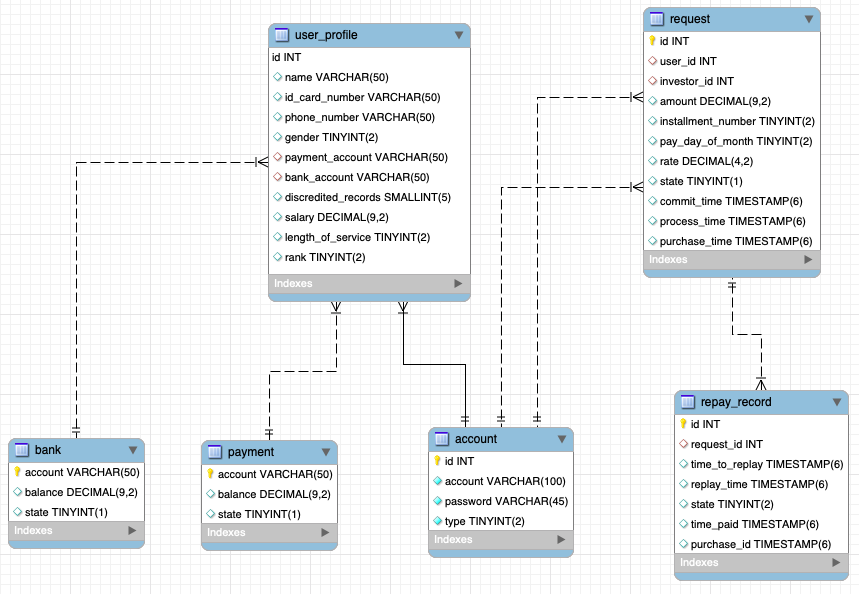
3.3.2.7管理员模块



3.3.2.8平台模块



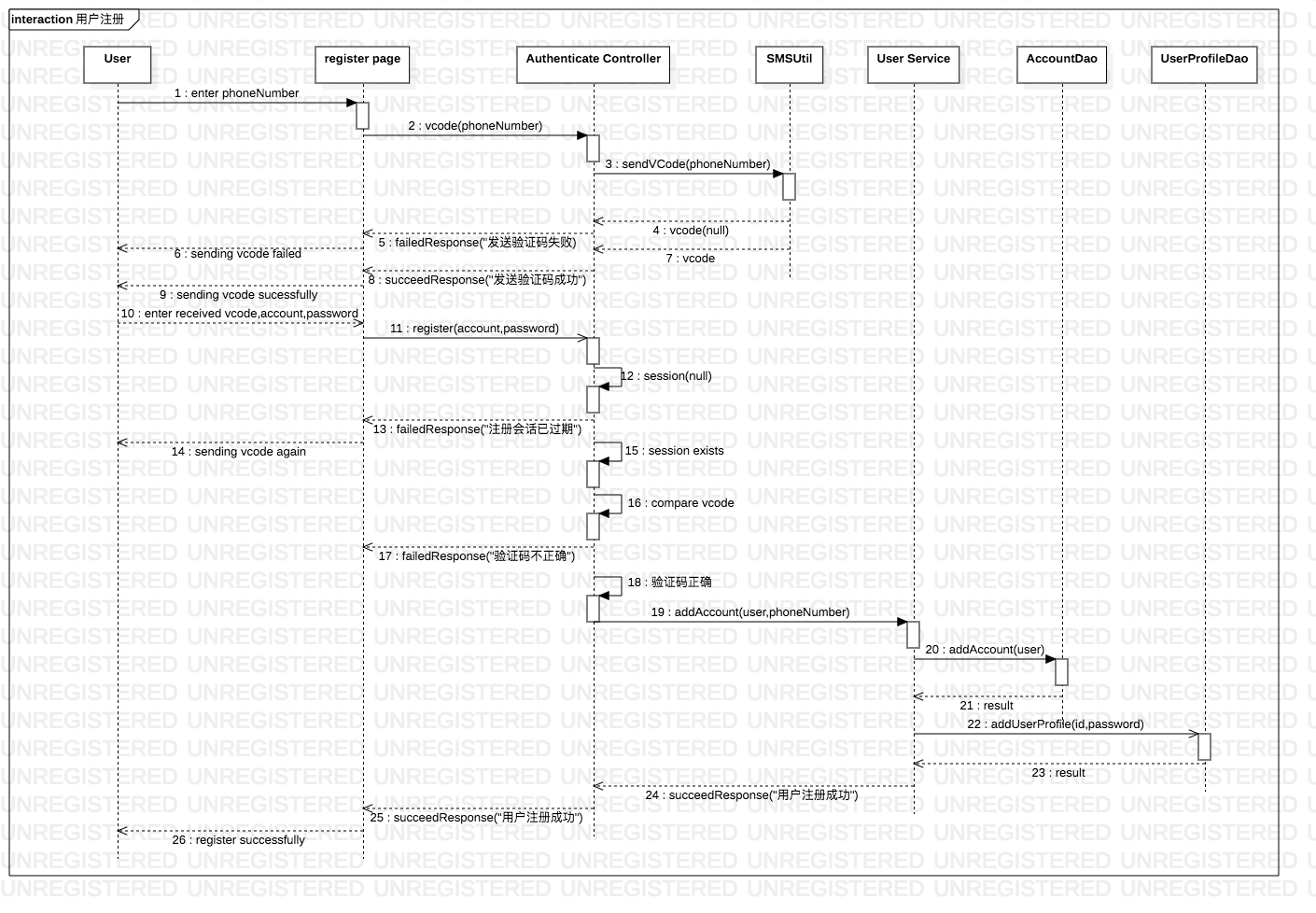
#### 3.3.3 数据库设计



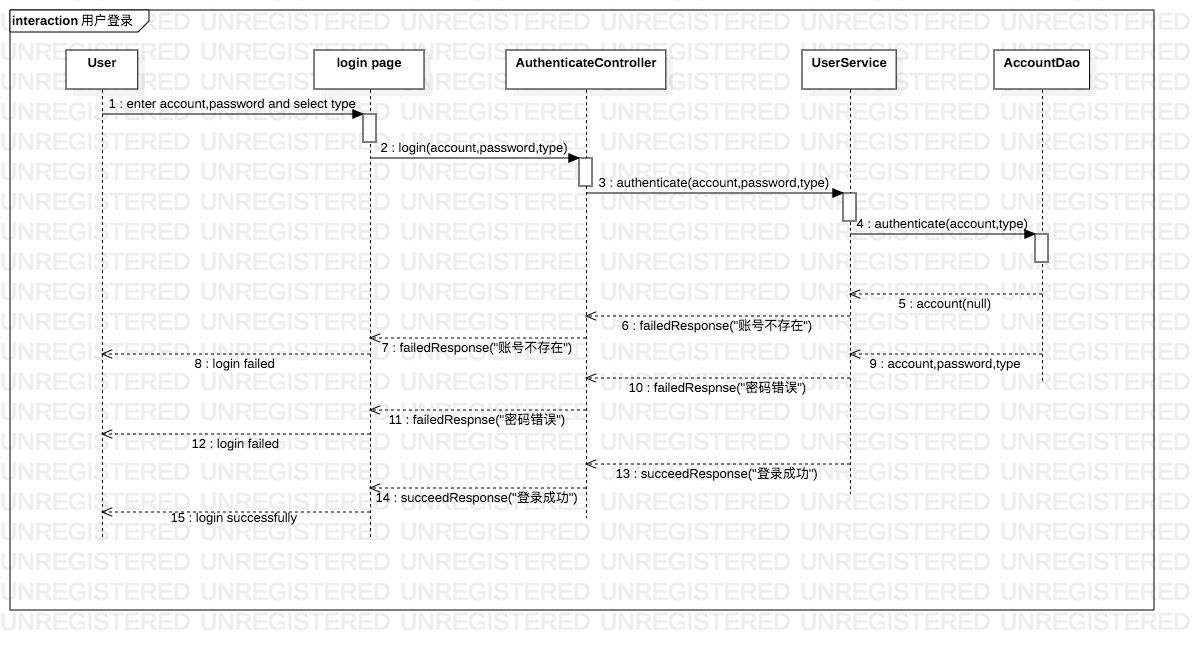
数据库设计如上图。构建满足基本的主、外键约束以及数量关系的模型，生成具体的数据库模式。黄色叹号为主键，青色菱形表示该字段不可为空。实体集中，连线的叉状末端表示该实体在两者关系中可能出现多个对应项，连线的双横线末端表示该实体在两者关系中仅有出现多个对应项

## 3.4 系统用例实现详细设计

#### 3.4.1 用户注册顺序图



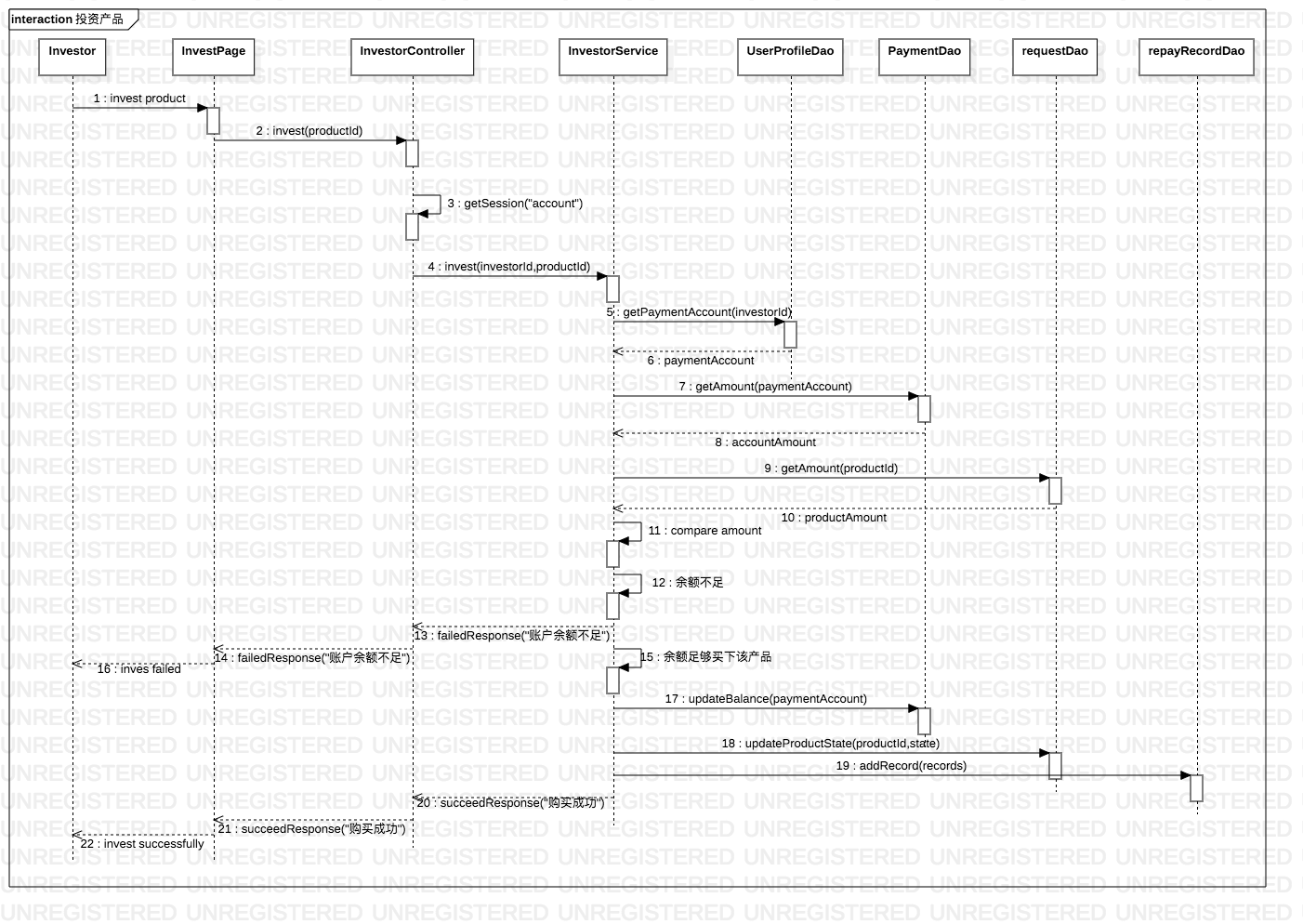
#### 3.4.2 用户登录顺序图



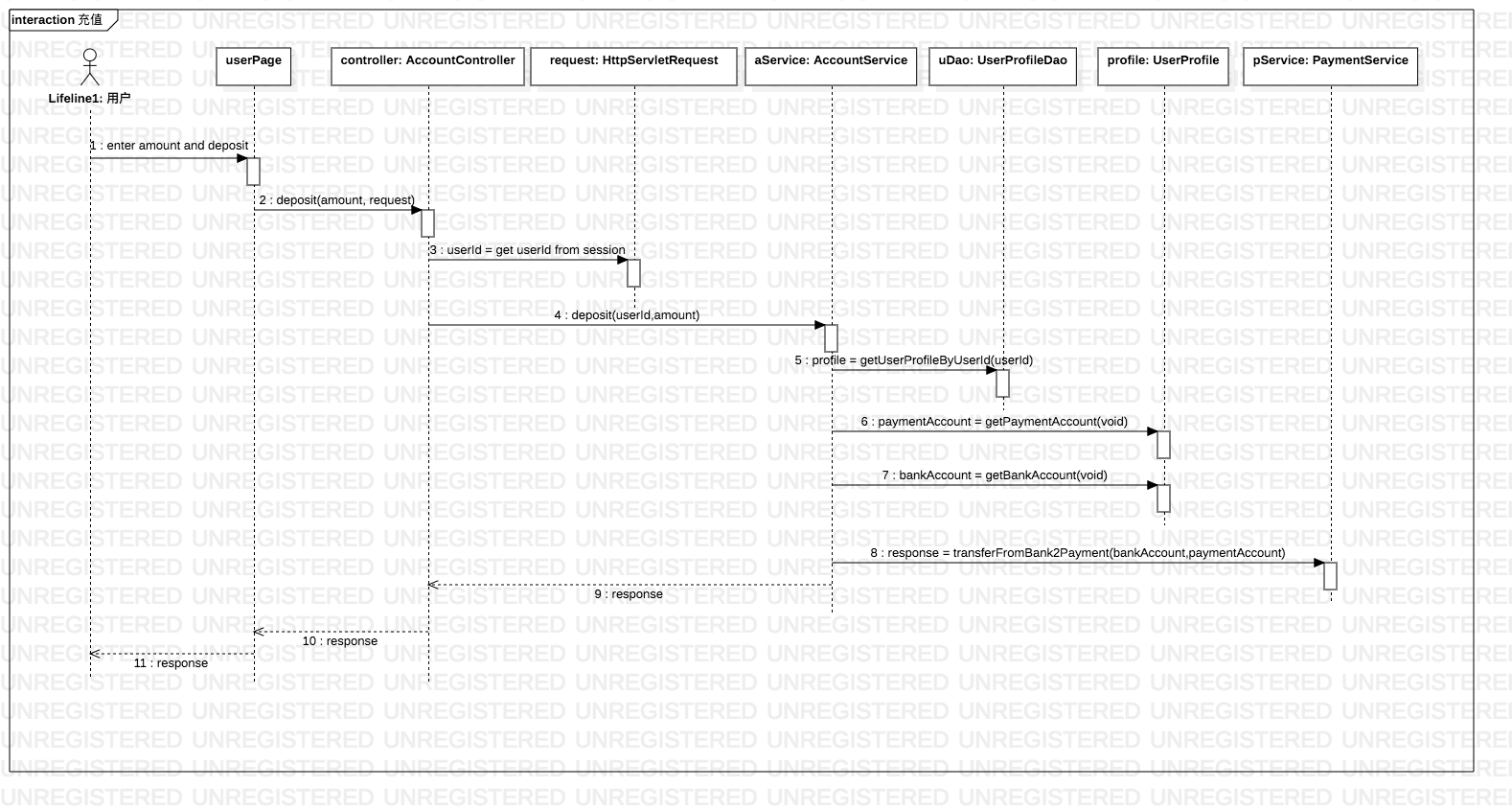
#### 3.4.3 用户登出顺序图



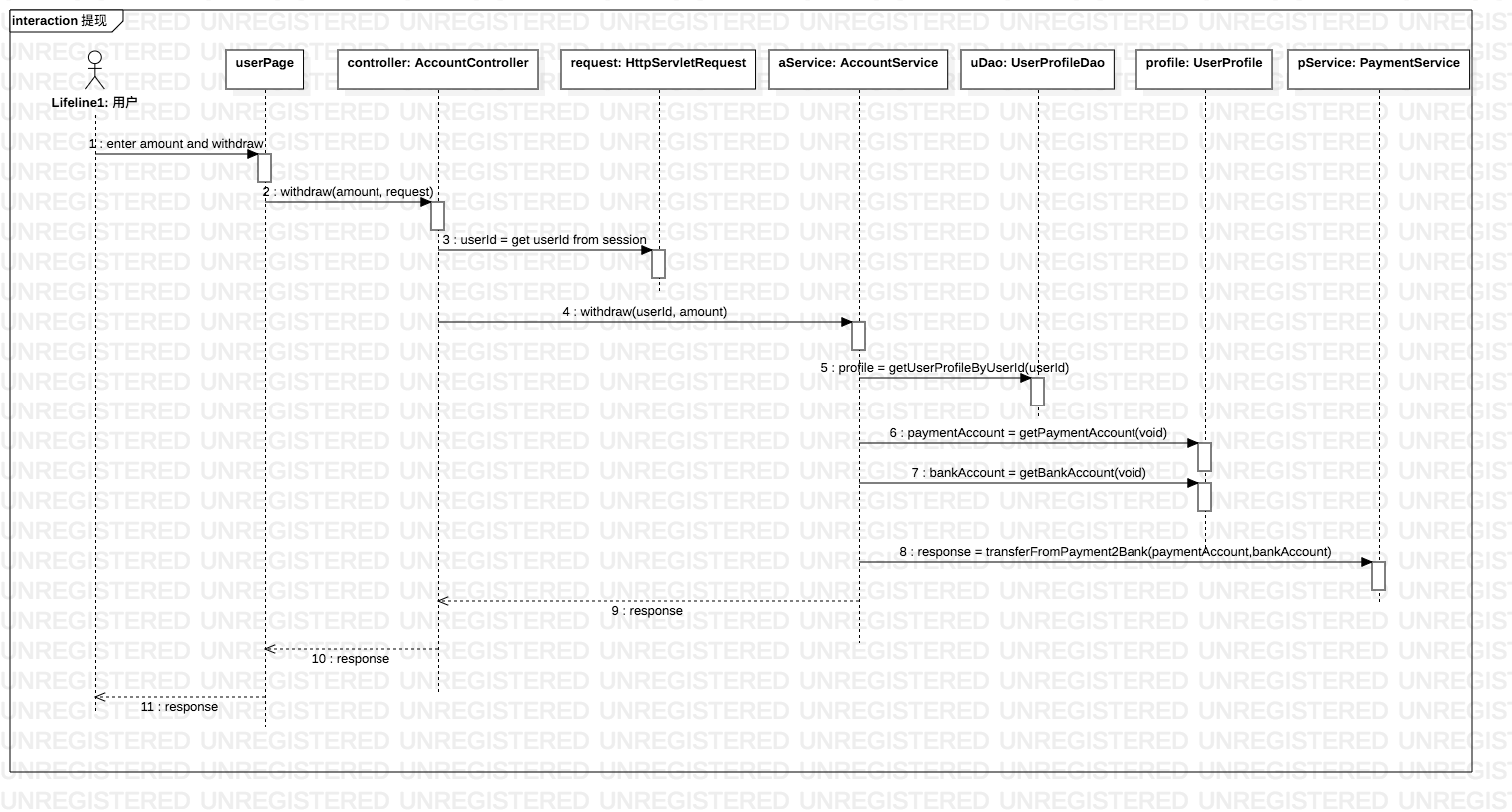
#### 3.4.4 投资产品顺序图



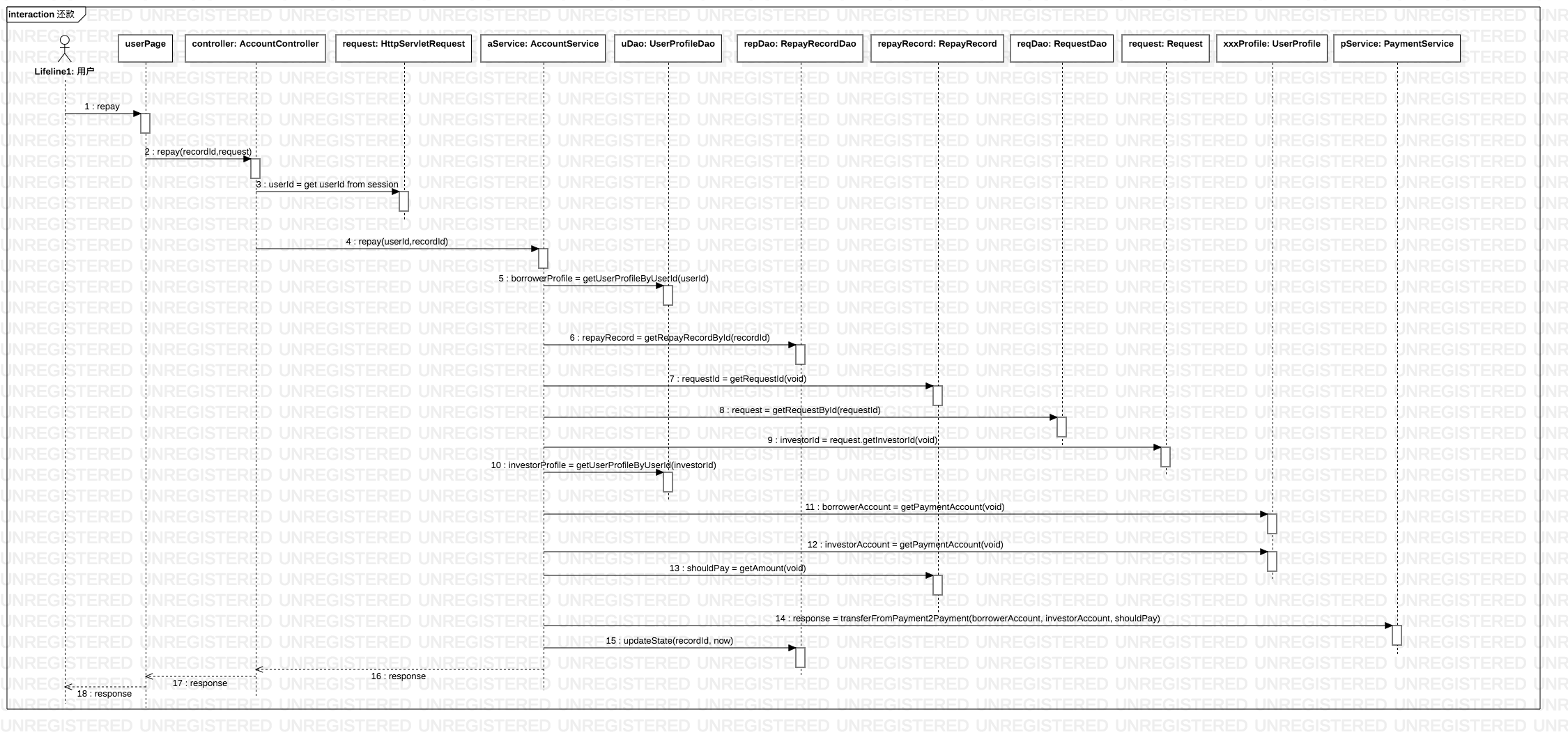
#### 3.4.5 用户充值顺序图



#### 3.4.6 用户提现顺序图



#### 3.4.7 用户还款顺序图



## 3.5 系统非功能性设计

#### 3.5.1 可用性

#### 3.5.2 可靠性

3.5.2.1 安全性

需求：系统应该保障用户账号密码在网络传输中的安全性。

设计：对密码进行加密处理。

3.5.2.2 容灾能力

需求：系统应该具备基本的容灾策略来保证服务的正常提供。

设计：如有必要，计划使用高可用性的物理部署方案

3.5.2.3 性能

1. 实时性

需求：系统需要保证时间敏感类信息的实时转发。

设计：通过较为稳定的TCP连接（Http连接）转发信息。

1. 数据传输格式

需求：系统的数据传输格式应该便于团队开发、调试，同时对系统运营有帮助。

设计：在开发前期，为了方便调试，使用简单的文本协议、JSON等数据格式。