Домашнее задание №6

Deadline: 2.06.19, 23:59.

Спроектировать банковскую систему с помощью UML-диаграммы классов.

Проектирование

Диаграмму спроектировать с помощью инструмента StarUml - http://staruml.io (можно использовать и более продвинутые инструменты, кот. упоминались на занятиях).

В диаграмме необходимо создать такие объекты:

Объект	Описание
Bank	Агрегирует связь с сервером и проверку пароля от пользователя на корректность. Когда пользователь посылает запрос в банк, последний посылает запрос в Checker, и, если права верные, посылает запрос от пользователя на сервер. Содержит логику проверки наличия средств и прав для совершения операции.
Bank Server	Хранит информацию об аккаунтах. Обрабатывает запрос и списывает/зачисляет соответствующие суммы на аккаунты.
User	Посылает запросы в банк. Проверить/изменить баланс, сделать перевод.
Account	Одному пользователю может соответствовать несколько счетов. У счета есть тип - Checking, Saving. В рамках предлагаемой задачи принципиально ничем не отличаются, только полем type. Пользователь может посылать запросы как на первый, так и на второй счет.
Checker	Связующее звено между банком и аккаунтом, проверяет имеет ли доступ пользователь к аккаунту. Отвечает за безопасность аккаунтов. Проверяет есть ли у запрашивающего пользователя права на управление аккаунтом.

Реализация

После построения диаграммы необходимо сгенерировать код на Java. Доделать до готового к запуску приложения (с точкой входа). Приложение должно обладать следующими свойствами:

1. Изначально на счету каждого пользователя 1000 у.е.

2. Пользователь может перевести средства со своего аккаунта на аккаунт другого пользователя. Для реализации перевода использовать Atomic-типы. Перевод должен осуществиться только если у переводящего достаточно средств на счету.
3. Пользователь может запросить перевод средств со счета любого другого пользователя.
4. При запросе средств пользователь отправляет запрос банку. Тот отправляет запрос серверу, который возвращает информацию о средствах на аккаунте и, если имеется достаточно прав и средств, операция осуществляется. Checker делает проверку в момент, когда пользователь посылает запрос банку на совершение операции.

Точкой входа должен быть класс Main, где создается несколько пользователей, и они совершают несколько транзакций между своими счетами. Должны предусматриваться сценарии, при которых совершить транзакцию невозможно (недостаточно средств).

5 пунктов выше должны проверяться Unit-тестами.

Сложный сценарий

В качестве сложного сценария реализуйте реляционную базу данных, в которой будут храниться данные о пользователях и их счетах. Заполните эту базу некоторым 20 пользователями. Методы, реализованные в предыдущих пунктах, должны при необходимости обращаться к базе.

Критерии оценивания

Задание	**Базовая часть**		**Тестирование**				**ВСЕГО**
Подзадача	UML-диаграмма	Программная реализация	1	2	3	4	
Балл (обычный сценарий)	2	4	0,5	1	1	1,5	10
Балл (сложный сценарий)	2	**9**	0,5	1	1	1,5	15