Система управления банковскими счетами

1. Анализ предметной области

Основной целью создания ПО "Система управления банковскими счетами" является автоматизация процедур создания, ведения и учета пользовательских банковских счетов. В соответствии с поставленной целью, система должна уметь решать следующие задачи:

- создание пользовательских счетов
- поддержка операций зачисления и списания денежных средств
- поддержка операций установки, снятия лимитов по счетам
- генерация отчетов по счетам

2. Техническое задание

1. Наименование

Полное наименование системы - "Автоматизированная система управления банковским счетами". Условное обозначение системы - программа.

2. Основание для разработки

Программный продукт создается на основе личной инициативы и задания.

3. Наименование заказчика и разработчика

Заказчик: Факультет ВМК МГУ Исполнитель: участник курса АДМО Федянин Анатолий Анатольевич

4. Назначение и цели разработки

Основной целью создания ПО "Система управления банковскими счетами" является автоматизация процедур создания, ведения и учета пользовательских банковских счетов.

5. Требования к разработке

5.1 Требования к функциональным характеристикам

В соответствии с поставленной целью, программа должна выполнять следующие функции:

5.1.1 Создание сберегательного счета

Входные данные:

- наименование пользователя
- тип счета: сберегательный
- первоначальный баланс на счете Выходные данные:
- сберегательный счет

Дополнительные требования:

- система должна уметь обрабатывать некорректный ввод пользователя
- система должна автоматически присваивать новый номер созданному счету по шаблону "S#####", где # - цифра

5.1.2 Создание текущего счета

Входные данные:

- наименование пользователя
- тип счета: текущий
- первоначальный баланс на счете Выходные данные:
- текущий счет

Дополнительные требования:

- система должна уметь обрабатывать некорректный ввод пользователя
- система должна автоматически присваивать новый номер созданному счету по шаблону "C#####", где # - цифра

5.1.3 Установка ограничений по минимальной и максимальной сумме на счете

Входные данные:

- номер счета
- минимальная сумма
- максимальная сумма Выходные данные:
- счет с установленными лимитами по минимальной максимальной сумме

Дополнительные требования:

- в случае ввода несуществующего номера счета, система должна отображать сообщение об ошибке

5.1.4 Внесение суммы на счет (депозит)

Входные данные:

- номер счета
- сумма депозита Выходные данные:
- счет с измененным балансом

Дополнительные требования:

- в случае ввода несуществующего номера счета, система должна отображать сообщение об ошибке
- в случае превышения максимального лимита, система должна отображать сообщение об ошибке

5.1.5 Снятие суммы со счета

Входные данные:

- номер счета
- сумма списания Выходные данные:
- счет с измененным балансом

Дополнительные требования:

- в случае ввода несуществующего номера счета, система должна отображать сообщение об ошибке
- в случае достижения минимального лимита, система должна отображать сообщение об ошибке

5.1.6 Выписка по счету

Входные данные:

- номер счета Выходные данные:
- детальная информация по счету

Дополнительные требования:

- в случае ввода несуществующего номера счета система должна отображать сообщение об ошибке

5.1.7 История транзакций по счету

Входные данные:

- номер счета Выходные данные:
- история транзакций: дата операции, тип транзакции депозит/списание, сумма

Дополнительные требования:

- в случае ввода несуществующего номера счета система должна отображать сообщение об ошибке

5.1.8 Реестр всех счетов

Входные данные: -

Выходные данные:

- список всех счетов в системе: тип счета, владелец, баланс, минимальный/максимальный лимиты

5.1.9 Экспорт всех транзакций в файл

Входные данные:

- имя выходного файла для экспорта Выходные данные:
- файл с реестром транзакций в текстовом формате

5.1.10 Экспорт всех счетов в файл

Входные данные:

- имя выходного файла для экспорта Выходные данные:
- файл с реестром счетов в текстовом формате

5.1.11 Импорт данных в систему из файлов

Входные данные:

- входной файл с реестром счетов
- входной файл с реестром транзакций Выходные данные:
- система, инициализированная данными из реестров

5.2 Требования к нефункциональным характеристикам

5.2.1 Аудит действий пользователя

Система должна вести аудит (логирование в файл) действий по счету:

- создание счета
- установка минимального и максимального лимита
- снятие средств
- внесение средств

5.2.2 Логирование ошибок

Система должна вести логирование в файл всех ошибок, возникающих при работе с системой

5.2.3 Набор юнит-тестов

Исходный код программы должен содержать наборы тест-кейсов с юнит-тестами для проверки работоспособности системы.

5.3 Требования к составу программных средств

5.3.1 Операционная система

Любая ОС с поддержкой интерпретатора Python

- Windows
- Linux
- MAC OS

5.3.2 Интерпретатор Python

Для работы программы требуется установленный на компьютере интерпретатор Python с версией, не ниже 3.10

5.3.4 Пакеты Python

Для корректной работы программы требуется установка внешних пакетов Python:

- click
- rich

5.4 Требования к интерфейсу программы

5.4.1 Основной интерфейс программы

Программа должна поддерживать работу интерфейса командной строки - CLI (command line interface) При вводе команд без параметров, система должна выводить сообщения о доступных командах и параметрах При вводе неполного набора параметров, система должна выводить подсказки с возможностью ввода пропущенных параметров При вводе некорректных команд и параметров, система должна выводить детальные сообщения об ошибках

5.4.2 Основные команды интерфейса

- add-account Добавить новый счет.
- all-accounts Отобразить все счета в виде списка.
- all-transactions Отобразить историю транзакций по счету.
- deposit Внести сумму на счет.
- details Отобразить выписку по счету.
- export-accounts Выгрузить реестр счетов в файл
- export-transactions Выгрузить историю транзакций в файл.
- import-data Загрузить счета и транзакции в систему.
- set-limits Установить лимиты по счету.
- withdraw Снять сумму со счета.

5.4.3 Порядок вызова команд CLI

Пример вызова команды:

```
PS C:\demo\src\bank_accounts> python main.py add-account --help
Usage: main.py add-account [OPTIONS] [[S|C]] [BALANCE]

  Добавить новый счет.

Options:
  -n, --name TEXT  Владелец счета
  --help           Show this message and exit.
```

3. Описание решения

Для реализации применена классическая слоевая архитектура:

- интерфейс пользователя
- сервисы приложения
- доменная модель

- слой доступа к данным

3.1 Интерфейс пользователя

Реализован в модуле main.py в виде интерфейса командной строки (CLI) В этом же модуле реализован сервис логирования действий пользователя и сохранение информации об ошибках

3.2 Сервисы приложения

Сервисы приложения расположены в модуле application.py Функции класса Application реализуют основные бизнес-сценарии приложения. В этом модуле реализовано сохранение/восстановление состояния системы между командами через объект bank_app.

3.3 Доменная модель

Доменная модель реализована в виде набора классов.

Модуль accounts.py - для объектов пользовательских счетов:

- Account базовый класс
- SavingAccount - наследник от Account - объект сберегательного счета
- CurrentAccount - наследник от Account - объект текущего счета
- AccountDict - реестр пользовательских счетов в виде словаря

Модуль transactions.py - для объектов транзакций по счетам:

- Transaction - объект транзакции по счету - списание, либо внесение средств
- TransactionList - реестр транзакций по счетам в виде исторического списка

3.4 Слой доступа к данным

Слой доступа к данным не выделен в отдельный модуль и реализован методами загрузки, сохранения состояния классами

- AccountDict (save, load)
- TransactionList (save, load)

4. Описание используемых структур данных

4.1 Реестр счетов (словарь)

Реализован в модуле accounts.py в виде класса AccountDict - наследника от встроенного типа dict

- ключ словаря - номер счета
- значение словаря - объект типа Account

4.2 Реестр транзакций

Реализован в модуле transactions.py в виде класса TransactionList - наследника от встроенного типа list

- значения списка - объекты типа Transaction

5. Исходный код программы

Исходный код доступен на GitHub репозитории: msu-daml-bank-accounts-demo

6. Описание тест-примеров

6.1 Добавление сберегательного счета

```
PS C:\demo\src\bank_accounts> python main.py add-account S 1500
Введите имя пользователя: Иван Табуреткин
2024-04-03 11:39:20,197 - __main__ - INFO - Создан новый счет #S00003
Номер счета: S00003
Клиент: Иван Табуреткин
Баланс: 1500.00
```

6.2 Добавление текущего счета

```
PS C:\demo\src\bank_accounts> python main.py add-account C 3500
Введите имя пользователя: Степан Капуста
2024-04-03 11:41:30,472 - __main__ - INFO - Создан новый счет #C00009
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 3500.00
```

6.3 Внесение депозита на счет

```
PS C:\demo\src\bank_accounts> python main.py deposit S00003 --amount 200
2024-04-03 11:43:40,542 - __main__ - INFO - Внесен депозит на счет #:S00003
Номер счета: S00003
Клиент: Иван Табуреткин
Баланс: 1700.00
```

6.4 Установка лимитов по счету

```
PS C:\demo\src\bank_accounts> python main.py set-limits C00009
Введите минимальный остаток на счете: 2000
Введите максимальную сумму на счете: 5000
2024-04-03 11:46:05,176 - __main__ - INFO - Установлены лимиты по счету #:C00009
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 3500.00
Минимальный лимит: 2000.00
Максимальный лимит: 5000.00
```

6.4 Списание со счета

```
PS C:\demo\src\bank_accounts> python main.py withdraw C00009 --amount 200
2024-04-03 11:48:15,395 - __main__ - INFO - Списана сумма со счета #:C00009
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 3300.00
Минимальный лимит: 2000.00
Максимальный лимит: 5000.00

PS C:\demo\src\bank_accounts> python main.py withdraw C00009
Введите сумму снятия: 600
2024-04-03 11:48:24,303 - __main__ - INFO - Списана сумма со счета #:C00009
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 2700.00
Минимальный лимит: 2000.00
Максимальный лимит: 5000.00

PS C:\demo\src\bank_accounts> python main.py withdraw C00009 --amount 1500
2024-04-03 11:48:40,939 - __main__ - ERROR - Ошибка при списании средств со счета
#C00009: Достигнуто ограничение по минимальной сумме на счете 2000.0
```

7. Результаты работы программы

7.1 Выписка по счету

```
PS C:\demo\src\bank_accounts> python main.py details C00009
Ежемесячный отчет по текущему счету
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 2700.00
Минимальный лимит: 2000.00
Максимальный лимит: 5000.00

PS C:\demo\src\bank_accounts> python main.py details S00003
Ежемесячная выписка по сберегательному счету
Номер счета: S00003
Клиент: Иван Табуреткин
Баланс: 1701.42
```

7.2 История транзакций по счету

```
PS C:\demo\src\bank_accounts> python main.py all-transactions C00009
Номер счета: C00009
Клиент: Степан Капуста
Баланс: 2700.00
Минимальный лимит: 2000.00
Максимальный лимит: 5000.00
```

Дата	Счет #	Депозит	Списание
2024-04-03	C00009		200.00
2024-04-03	C00009		600.00

7.3 Реестр всех счетов

```
PS C:\demo\src\bank_accounts> python main.py all-accounts
```

#	Тип счета	Пользователь	Баланс	Мин. лимит	Макс. лимит
C00005	Текущий	Robert Yeo	-144.62		
C00008	Текущий	Lim Ah Seng	3326.37		
S00001	Сберегательный	Lim Ah Seng	680.15		
S00002	Сберегательный	Tan Ah Lian	5809.68		
S00003	Сберегательный	Иван Табуреткин	1700.00		
C00009	Текущий	Степан Капуста	2700.00	2000.00	5000.00

7.4 Реестр всех транзакций

```
PS C:\demo\src\bank_accounts> python main.py all-transactions
```

Дата	Счет #	Депозит	Списание
2012-07-13	C00005		200.00
2012-07-13	S00002		150.79
2012-07-13	S00001	120.00	
2012-07-14	C00008	1680.45	
2012-07-14	S00001		330.00
2012-07-15	C00005		675.50
2012-07-15	S00002	760.00	
2024-04-03	S00003	200.00	
2024-04-03	C00009		200.00
2024-04-03	C00009		600.00

8. Список используемых информационных источников

Click

- Click
- Build a Command Line Interface with Python Click
- Professional CLI Applications with Click
- Create a Task Tracker App for the Terminal with Python (Rich, Typer, Sqlite3)

Logger

- Logging
- Logging-cookbook

Rich

- Beautiful Terminal Styling in Python With Rich

Libs

- 15 Python Libraries You Should Know About