**Введение**

На учебную практику была поставлена задача, разработать консольное приложение на тему: разработка консольного приложения «Виртуальный банкомат».

Цель проекта: предоставление базовых функций банкомата (снятие наличных, пополнение счета, проверка личного счета) пользователю «ATM Banking».

Создаваемое консольное приложение будет рассчитано на пользователей с 16 лет. Получить информацию приложения смогут все желающие, которые имеют банковскую карту.

Далее приведем краткое описание разделов пояснительной записки.

Первый раздел “Анализ задачи”. В нем вы сможете ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе “Инструменты разработки” будет рассмотрена среда, в которой создается данный курсовой проект. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачей.

В разделе “Проектирование задачи” будут рассмотрены основные аспекты разработки программного продукта. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

“Реализация задачи” – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полная аннотация файлов, используемых в данном проекте.

В разделе “Тестирование”, будет описано полное и функциональное тестирование данной программы, т.е. будет оттестирован каждый пункт меню, каждая операция, которая выполняется приложением. Будут смоделированы все возможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе “Применение” будет описано назначение, область применения, среда функционирования курсовой программы.

“Заключение” будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

Список используемых при разработке источников будет содержать:

- В приложениях к пояснительной записке будет приведен листинг программы. В приложении будет указаны диаграмма вариантов использования, диаграмма классов, диаграмма последовательности, диаграмма деятельности, список операций.

**1 Анализ задачи**

* 1. **Постановка задачи**

**1.1.1 Организационно-экономическая сущность задачи**

**Наименование задачи:** консольное приложение «Виртуальный банкомат».

**Цель разработки:** предоставление базовых функций банкомата (снятие наличных, пополнение счета, проверка личного счета, просмотр 5 последних операций) пользователю о банкомате «ATM Banking».

**Назначение:** Создаваемое консольное приложение будет рассчитано на пользователей любого возраста. Получить информацию приложения смогут все желающие, которые имеют банковскую карту.

**Периодичность использования:** по мере необходимости.

**Источники и способы получения данных:** Беларусбанк.

**Обзор существующих аналогичных ПП:** <https://belarusbank.by>, на главной странице, мы видим список возможностей, которыми можно пользоваться в этом банке.

**1.1.2 Функциональные требования**

Описание перечня функций и задач, которые должен выполнять будущий ПП:

**Пользователь:**

1. Регистрация

2. Авторизация

3. Просмотр личного счета

4. Снятие наличных

5. Пополнение счета

6. Посмотреть последние 5 операций

**Администратор:**

1. Все действия пользователя и добавление/удаление новых функций

**1.1.3 Описание процессов с входной, выходной и условно постоянной информацией**

В таблице 1 Описаны функции программы с входной, выходной и условно-постоянной информацией

Таблица 1 – Функции программы с описанием с входной, выходной и условно-постоянной информации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Категория пользова-телей | Наименование процесса | Краткое описание алгоритма  выполнения процесса | Входная информация | Выходная информация | Условно  -постоянная  информация |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | Пользователь | Регистрация | Любому пользователю  разрешено регистрироваться в программе | Отсутствует | Отсутствует | Данные программы |
| 2 | Пользователь | Авторизация | При нажатии будет авторизация (для  зарегистрированного пользователя),  нужно заполнить  требуемые поля | Заполнение полей  авторизации  (логин, пароль) | Вход в аккаунт | Отсутствует |
| 3 | Пользователь | Просмотр личного счета | При нажатии будет показан баланс на карте | Номер личного счета | Получение информации о балансе | Отсутствует |
| 4 | Пользователь | Снятие наличных | При нажатии будет выбор суммы, которую вы хотите снять | Выбор суммы | Чек (если понадобится) | Отсутствует |
| 5 | Пользователь | Пополнение счета | При нажатии будет выбор счетов, которые вы хотите пополнить. | Сумма пополнения | Чек (если понадобится) | Отсутствует |
| 6 | Пользователь | Посмотреть последние 5 операций | При нажатии будет даваться чек, с последними 5 операциями, совершавшимися на данном аккаунте | Отсутствует | Чек с 5 последними операциями | Отсутствует |
| 7 | Администратор | Добавление/удаление новых функций | Внесение изменений в функционал программы | Отсутствует | Новая функция | Отсутствует |

**1.1.4 Эксплуатационные требования**

**Требования к применению:** Для применения программы будет использоваться консольный режим, в котором нужно будет выбрать номер команды.

**Требования к производительности:** Минимальные системные требования для программного продукта должны быть:

процессор Intel Core i5, AMD Ryzen 5;

SSD от 256 ГБ;

16 ГБ ОЗУ.

**Требования к реализации:** Для реализации и запуска программы понадобится Java.

**Требования к надежности:** Система должна быть надежная, чтобы нельзя было подобрать или угадать пароль, или логин.

**1.4 Выбор стратегии разработки и модели жизненного цикла**

Для разработки веб-ресурса следует выбрать стратегию разработки и модель жизненного цикла. Осуществляем выбор посредством составления таблиц:

Таблица 2 – Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории требований | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Да | Да | Да | Нет | Нет | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да | Да | Да | Да | Нет | Нет |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет | Нет | Нет | Нет | Да | Да |

Продолжение таблицы 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Нет | Нет | Да | Нет | Да | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет | Нет | Да | Нет | Да | Да |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Нет | Нет | Да | Да | Да | Да |

Вычисления: 3 за каскадную, 3 за V- образную, 6 за RAD, 4 за инкрементную, 4 за быстрого прототипирования и 4 за эволюционную.

Итог: На основе результатов заполнения табл. 2 подходящей является RAD модель.

Таблица 3 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |

Продолжение таблицы 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта?  Хватит ли членов команды и согласны ли они? | Да | Да | Да | Да | Нет | Нет |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |

Вычисления: 4 за каскадную, 4 за V-образную, 5 за RAD, 4 за инкрементную, 3 за быстрого прототипирования и 2 за эволюционную.

Итог: На основе результатов заполнения табл. 3 подходящей является RAD модель.

Таблица 4 – Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории коллектива пользователей | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки?  Тестирование пользователем. | Нет | Нет | Нет | Да | Да | Да |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки?  Совет. | Нет | Нет | Да | Нет | Да | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 3 за каскадную, 3 за V-образную, 1 за RAD, 2 за инкрементную, 1 за быстрого прототипирования и 3 за эволюционную.

Итог: На основе результатов заполнения табл. 4 подходящей являются каскадная модель, V-образная модель и эволюционная модель.

Таблица 5 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории типов проекта и рисков | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Разрабатывается ли в проекте продукт нового для организации (команды) направления? | Нет | Нет | Нет | Да | Да | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| 3. | Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| 8. | Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| 9. | Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 7 за каскадную, 8 за V-образную, 6 за RAD, 2 за инкрементную, 2 за быстрого прототипирования и 2 за эволюционную.

Итог: На основе результатов заполнения табл. 5 подходящими является V- образная модель.

Общий итог: в итоге заполнения табл. 2 – 5 наиболее подхочёдящей является RAD модель.

17 каскадная модель

18 V-образная модель

18 RAD модель

12 инкрементная модель

10 быстрого прототипирования модель

11 эволюционная модель.

**1.5 Инструменты разработки**

Для разработки данного проекта был выбран язык программирования «Java», а также приложение IntelliJ IDEA.

Также для разработки программы необходимы:

- IntelliJ IDEA Community Edition 2023.2.2 – нужен для запуска кода;

- Word 2013 – нужен для написания пояснительной записки

- Draw.io – нужен для создания диаграмм.

При разработке данного программного продукта был использован компьютер со следующими характеристиками:

- Процессор: 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz 2.30 GHz;

- ОЗУ: 16Gb;

- Память: SDD 512Gb;

- ОС: Windows 11 Pro.

[**2 Проектирование задачи**](#_Достоинства_и_недостатки)

[**2.1**](#_Toc406878086) **Разработка структуры сайта, системы меню, навигации**

На рисунке 1 будет показана система главного меню.

Рисунок 1 – структура главного меню для пользователей

**2.3 Разработка UML-диаграмм**

Диаграмма в языке моделирования UML — наглядное представление некоей совокупности элементов модели системы в виде графа, на котором дуги связывают вершины. В своём графическом виде различные виды диаграмм UML применяются для визуализации разных аспектов устройства или поведения моделируемой системы.

**2.3.1 Диаграмма вариантов использования**

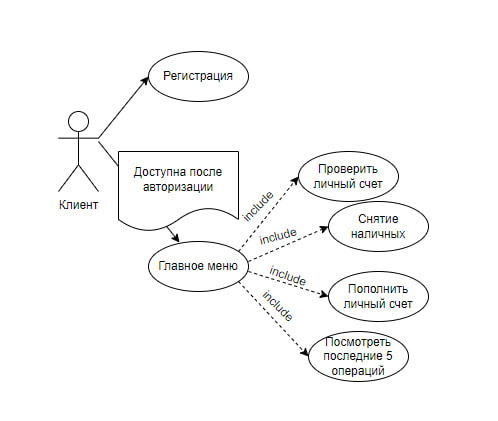
Проектирование – один из важных шагов при разработке программы, который очень часто игнорируется начинающими разработчиками. Обычно они пытаются удержать всё в голове или записать некоторые важные сведения на листе бумаги. Как результат, у них нет чёткого плана дальнейших действий, и проект может быть отложен в долгий ящик.

Обычно при проектировании разработчики изображают систему графически, поскольку человеку легко разобраться в таком представлении. Именно поэтому вместо написания огромных текстов про каждую возможность будущей программы разработчики строят различные диаграммы для описания своих систем. Это помогает им не забывать, что нужно реализовать в программе, и быстро вводить в курс дела своих коллег.

На диаграмме вариантов использования изображаются:

- Пользователь — пользователь, у которого есть доступ к банкомату.

- Действия которые может выполнить пользователь.

Рисунок 2 – Диаграмма вариантов использования

**2.3.2 Диаграмма последовательности**

Диаграмма последовательности UML — такая диаграмма, на которой показаны взаимодействия объектов, упорядоченные по времени их проявления. Основные элементы диаграммы последовательности это: обозначения объектов (прямоугольники), вертикальные линии, отображающие течение времени при деятельности объекта, и стрелки, показывающие выполнение действий объектами.

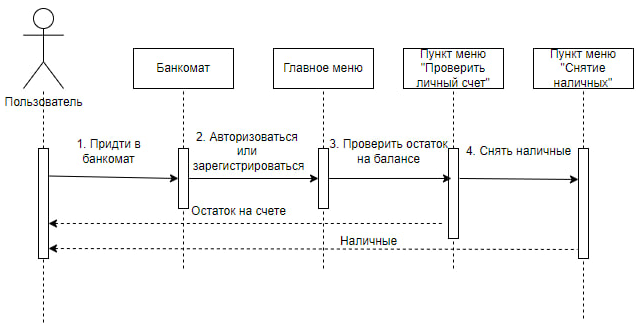


Рисунок 3 – Диаграмма последовательности

**2.4 Разработка пользовательского интерфейса**

На рисунке 4 представлен интерфейс программы «Виртуальный банкомат».

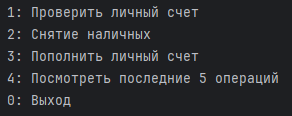


Рисунок 4 – Пользовательский интерфейс

**3 Реализация**

**3.1 Руководство программиста**

Данный программный продукт был разработан с помощью IntelliJ IDEA – IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. При входе в продукт, будет окно с входом в систему. При нажатии на кнопку «Регистрация», пользователь записывает необходимые данные, которые просит программа (имя, фамилия, логин, пароль), все эти данные сохраняются в файл. При нажатии на кнопку «Авторизация», пользователь заходит в систему под своим логином и паролем, и на это у него будет 3 попытки, по истечению всез попыток, будет завершена программа. При входе в систему будет выведено меню действий, которыми может воспользоваться пользователь. Меню: 1) Проверить личный счет. 2) Снять наличные. 3) Пополнить счет. 4) посмотреть последние 5 операций. 5) Выход. При нажатии на первый вариант меню, пользователю будет выведен баланс на его счете. При нажатии на второй вариант меню, пользователь может снять деньги со счета, но не больше, чем у него есть на счету. При нажатии на третий вариант меню, пользователь может внести депозит на счет. На четвертый вариант, ему выведет последние 5 действий, которые делались у него на аккаунте. А на пятый вариант, закроется программа.

Листинг программы содержится в Приложении А

**3.2 Спецификация проекта**

**3.2.1 Введение**

- Название программы: «Виртуальный банкомат АТМ»

- Версия: 1.0

- Цель проекта: разработка виртуального банкомата, предоставление базовых функций банкомата (снятие наличных, пополнение счета, проверка личного счета) пользователю «ATM Banking».

- Автор: Смирнов Артём Сергеевич

- Дата создания: 06.10.2023

- Дата последнего обновления: 16.11.2023

**3.2.2 Описание программы**

- Описание функциональности: в программе можно зарегестрироваться, авторизоваться, снять наличные, пополнить счет, просмотреть баланс, посмотреть последние 5 операций.

- Платформа: Windows 11.

- Язык программирования: Java.

**3.2.3 Требования**

- Требования к аппаратным ресурсам: минимальные требования

- Требования к ОС: Windows 7, XP, Vista

- Процессор: 1 ГГц и более

- Оперативная память: 2 ГБ и более

- Свободное место на диске: 5 ГБ

**3.2.4 Функциональность**

- Основные функции программы – снятие наличных, пополнение счета, проверка личного счета, просмотр 5 последних действий.

- Входные данные – выбор пунктов меню

- Алгоритм работы – окно с приветствием и выбором способа входа (регистрация или авторизация), после вхождения в систему, появляется главное меню, проделывание всех нужных операций.

**3.2.5 Используемые библиотеки и компоненты**

java.io.\*; java.util.HashMap; java.util.Map; java.util.Scanner;

**3.2.6 Среда разработки**

- Среда разработки – IntelliJ IDEA

- Версия среды разработки – 2.2

- Язык программирования: Java

[**4 Тестирование**](#_Достоинства_и_недостатки)

**4.1 Тесты на использование**

При разработке консольного приложения «Виртуальный банкомат» многие возникающие ошибки и недоработки были исправлены на этапе реализации программного продукта. После завершения испытания реализации web-ресурса было проведено тестирование. Функциональное тестирование должно гарантировать работу всех элементов программного продукта.

Таблица 6 – таблица с тест-кейсами

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Физический результат | Результат тестирования |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Проверка пункта «регистрация» | Сохранение данных в файл | Данные сохранены в файле | Выполнено |
| 2 | Проверка пункта «авторизация» | Вход в систему с данными из файла | Вход в систему | Выполнено |
| 3 | Проверка пункта меню «Проверить личный счет» | Вывод на экран баланса пользователя | Вывод на экран баланса пользователя | Выполнено |
| 4 | Проверка пункта меню «Пополнить счет» | Увеличение баланса пользователя | Увеличение баланса пользователя | Выполнено |
| 5 | Проверка пункта меню «Снятие наличных» | Уменьшение баланса на сумму, которую снял пользователь | Уменьшение баланса | Выполнено |
| 6 | Проверка пункта меню «Посмотреть последние 5 операций» | Вывод на экран 5 последних операций |  | Не выполнено |
| 7 | Проверка пункта меню «Выход» | Закрытие программы | Закрытие программы | Выполнено |

**4.2 Отчет о результатах тестирования**

В результате проведения тестирования выяснилось, что все ранее оговоренные функции и требования, были разработаны, а также протестированы. Тесты показали, что все функции работают правильно. Следовательно, разработанный консольное приложение можно передать заказчику.

В ходе тестирования программного обеспечения продукта на устройстве не было выявлено каких-либо ошибок.

**5 Руководство пользователя**

На рисунках 4 – 9 будет показана работа приложения (все действия которые работают).

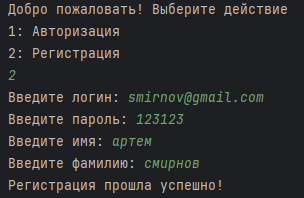


Рисунок 4 - Регистрация

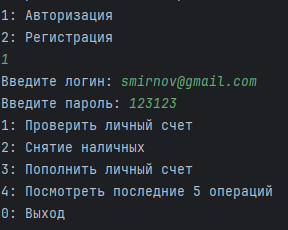


Рисунок 5 - Авторизация

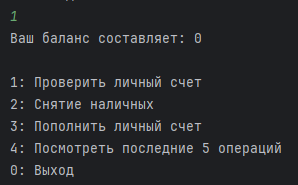


Рисунок 6 – Проверка личного счета

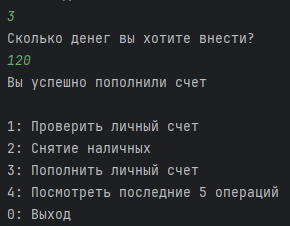


Рисунок 7 – Пополнение личного счета

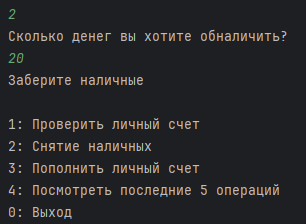


Рисунок 8 – Снятие наличных

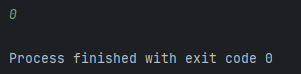


Рисунок 9 – Выход

[**Заключение**](#_Toc406878088)

Целью данного проекта Разработка программного обеспечения «Виртуальный банкомат» является разработка консольного приложения, позволяющей про проект (смог не смог реализовать, буду ли продолжать и т.д.)

В целом при реализации программного продукта, были выполнены все условия, поставленные на начальном этапе разработки. Из чего мы можем сделать вывод, что приложение можно передавать заказчику и вводить в эксплуатацию.

Так же в процессе создания программного продукта была подготовлена программная документация.

**Приложение А**

Листинг программы

import java.io.\*;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Scanner;  
  
  
  
public class Main {  
 private static Map<String, User> DATABASE;  
  
 private static final String DATABASE\_FILE\_PATH = "database.txt";  
 private static final String MENU = "1: Проверить личный счет\n2: Снятие наличных\n3: Пополнить личный счет\n" +  
 "4: Посмотреть последние 5 операций\n0: Выход";  
  
  
 public static void main(String[] args) throws FileNotFoundException {  
 DATABASE = readFromInputStream();  
 System.out.print("Добро пожаловать! Выберите действие\n1: Авторизация\n2: Регистрация\n");  
  
 Scanner sc = new Scanner(System.in);  
 int choice;  
 try {  
 choice = sc.nextInt();  
 } catch (Exception e) {  
 System.out.println("Введите нормальную тему");  
 Main.main(args);  
 choice = 0;  
 }  
 switch (choice) {  
  
 case 1:  
  
 boolean isSuccess = authorize();  
  
 if (isSuccess) {  
 printMenu();  
 } else {  
 System.out.println("Вы ввели неверные данные три раза подряд. Попробуйте позже.");  
 System.exit(1);  
  
 }  
 break;  
 case 2: {  
 Registration.apply();  
 break;  
 }  
 case 0: {  
 System.out.println("Bye bye");  
 System.exit(0);  
 }  
 default: Main.main(args);  
 }  
 Main.main(args);  
 }  
  
 private static boolean authorize() {  
 Scanner scanner = new Scanner(System.in);  
  
 int attempts = 0;  
  
 while (attempts < 3) {  
 System.out.print("Введите логин: ");  
 String login = scanner.next().toLowerCase();  
  
 System.out.print("Введите пароль: ");  
 String password = scanner.next();  
  
 if (DATABASE.containsKey(login)) {  
 User candidat = DATABASE.get(login);  
  
 if (candidat.getPassword().equals(password)) {  
 return true;  
 }  
 }  
  
 attempts++;  
  
 System.out.println("Попытка " + attempts + ", попробуйте еще раз! :/");  
  
 if (attempts >= 3) {  
 return false;  
 }  
 }  
  
 return false;  
 }  
  
 static void printMenu() {  
  
 int balance = 0, withdraw, deposit;  
  
 while (true) {  
 System.out.println(MENU);  
  
 Scanner sc = new Scanner(System.in);  
 int choice = sc.nextInt();  
  
 switch (choice) {  
  
 case 1:  
  
 System.out.println("Ваш баланс составляет: " + balance);  
 System.out.println("");  
 break;  
  
 case 2:  
  
 System.out.println("Сколько денег вы хотите обналичить?");  
 withdraw = sc.nextInt();  
  
 if (withdraw < 0) {  
 System.out.println("Так нельзя!!!");  
 }else {  
  
 if (balance >= withdraw) {  
 balance -= withdraw;  
 System.out.println("Заберите наличные");  
  
 } else {  
 System.out.println("У вас нет столько денег :(");  
 }  
 }  
 System.out.println("");  
 break;  
  
 case 3:  
  
 System.out.println("Сколько денег вы хотите внести?");  
 deposit = sc.nextInt();  
 if (deposit > 0) {  
 balance += deposit;  
 System.out.println("Вы успешно пополнили счет");  
 System.out.println("");  
 }else{  
 System.out.println("Так нельзя!!!");  
 System.out.println("");  
 }  
 break;  
  
 case 4:  
  
  
 case 0:  
 System.exit(0);  
 }  
 }  
 }  
  
 private static Map<String, User> readFromInputStream() throws FileNotFoundException {  
 Map<String, User> db = new HashMap<>();  
  
 File myObj = new File(DATABASE\_FILE\_PATH);  
 Scanner myReader = new Scanner(myObj);  
 while (myReader.hasNextLine()) {  
 String data = myReader.nextLine();  
 String[] values = data.split(" ");  
  
 User user = new User();  
 user.setLogin(values[0]);  
 user.setName(values[2]);  
 user.setPassword(values[1]);  
 user.setSecondName(values[3]);  
  
 db.put(values[0], user);  
  
 }  
 myReader.close();  
  
 return db;  
 }  
}

public class User {  
 private String login;  
 private String password;  
 private String Name;  
 private String SecondName;  
  
 public String getLogin() {  
 return login;  
 }  
  
 public void setLogin(String login) {  
 this.login = login;  
 }  
  
 public String getPassword() {  
 return password;  
 }  
  
 public void setPassword(String password) {  
 this.password = password;  
 }  
  
 public String getName() {  
 return Name;  
 }  
  
 public void setName(String Name) {  
 this.Name = Name;  
 }  
  
 public String getSecondName() {  
 return SecondName;  
 }  
  
 public void setSecondName(String SecondName) {  
 this.SecondName = SecondName;  
 }  
}

import java.io.BufferedWriter;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.Scanner;  
  
public class Registration {  
 protected static void apply() {  
 Scanner scanner = new Scanner(System.in);  
  
 System.out.print("Введите логин: ");  
 String login = scanner.next();  
  
 System.out.print("Введите пароль: ");  
 String password = scanner.next();  
  
 System.out.print("Введите имя: ");  
 String name = scanner.next();  
  
 System.out.print("Введите фамилию: ");  
 String secondname = scanner.next();  
  
 // Создаем объект для работы с файлом  
 try (FileWriter fileWriter = new FileWriter("database.txt", true);  
 BufferedWriter writer = new BufferedWriter(fileWriter)) {  
 // Записываем данные в файл  
 writer.newLine();  
 writer.write(login + " " + password + " " + name + " " + secondname);  
 } catch (IOException e) {  
 System.out.println("Ошибка при записи в файл: " + e.getMessage());  
 }  
  
 System.out.println("Регистрация прошла успешно!");  
 }  
}

**Приложение Б**

Диаграмма деятельности

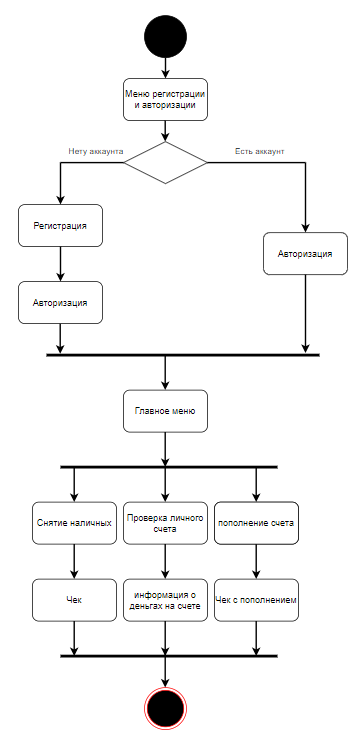


Рисунок 10 – Диаграмма классов

**Приложение В**

Диаграмма Ганта

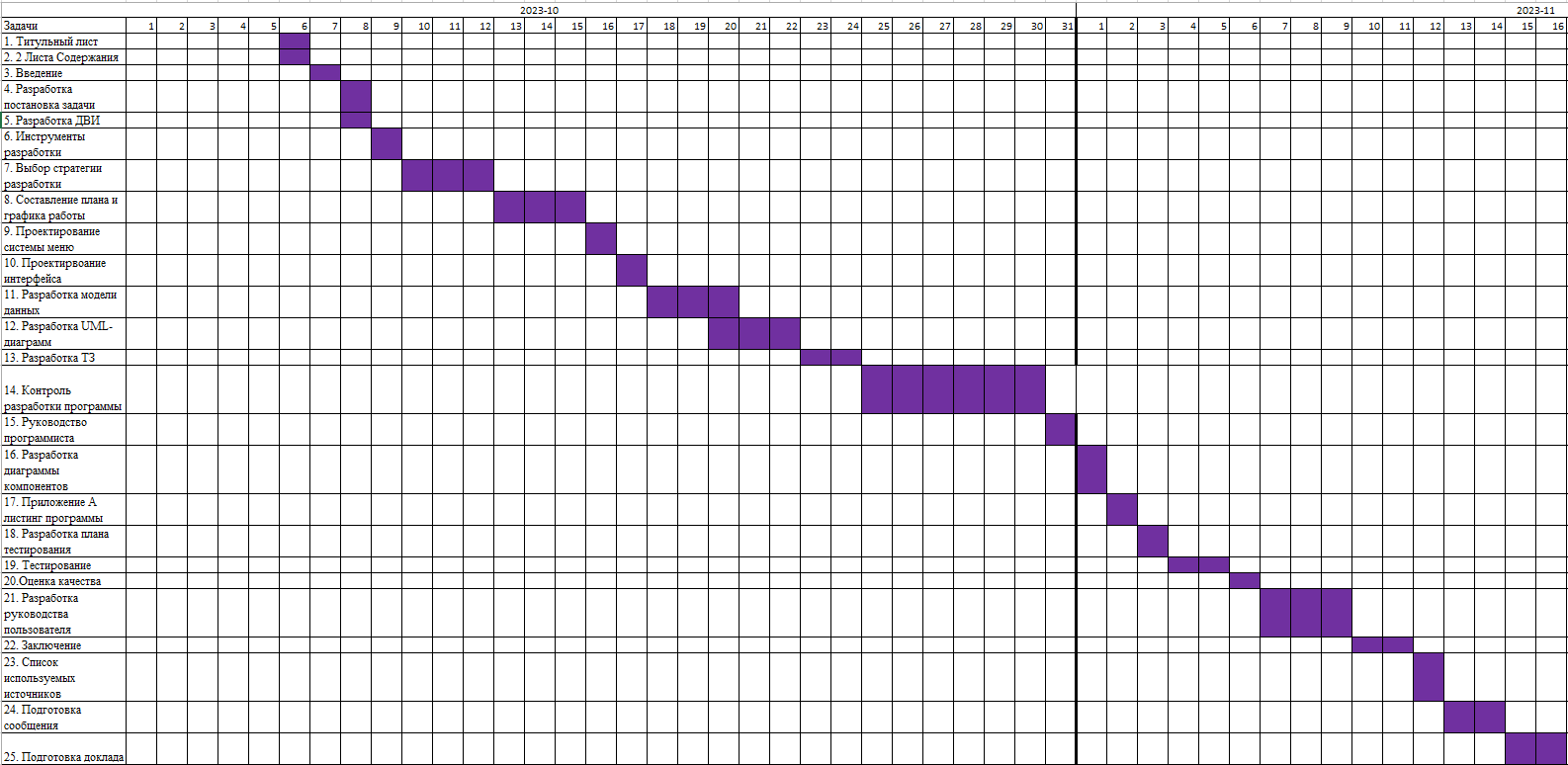


Рисунок 11 – Диаграмма Ганта