Государственное бюджетное образовательное учреждение высшего образования
Московской области
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Колледж космического машиностроения и технологий

КУРСОВОЙ ПРОЕКТ

По МДК.01.02 «Прикладное программирование»

Тема: «Разработка приложения "Пункт обмена валюты"»

Выполнил студент

Овчинников П.А.

Группа П1-17

_____(Подпись)

_____ (Дата сдачи работы)

Проверил преподаватель

Гусятинер Л.Б.

_____ (Оценка)

_____(Подпись)

Королёв 2020 г.

Оглавление.

Введение.

Цель данного курсового проекта - написание программы «Пункт обмена валюты», подобные пункты имеют особую актуальность в веке постоянных политических напряжений. Моей задачей было автоматизировать процесс взаимодействия с базой данных для работающих в таких пунктах людей, путём создания специальной программы.

В трёх частях будут рассмотрены: предметная область темы, а также приведены примеры уже существующих разработок в области финансового менеджмента, их описание, характеристики, инструменты, модули, которые были разработаны, структура программной части, листинги ключевых частей программных модулей, руководство для пользователей, а также изображение графического интерфейса программы.

Глава 1. Теоретическая часть

1.1 Описание предметной области

Валютно-денежная составляющая экономики – необходимая часть экономики любой страны. Ежеминутно меняются курсы тысяч валют и совершаются множество сделок от разных людей.

Главной целью использования системы баз данных обменного пункта является автоматизированное взаимодействие с базой данных. Требования к функционалу для каждой системы могут варьироваться от создания записей до проведения транзакций под защищенным соединением.

Современные пункты обмена валют используют программы и цифровые базы данных для автоматизированной работы, а так же для быстрого и удобного доступа к и истории сделок и другой необходимой информации для работы. Встречаются такие пункты в разных видах.

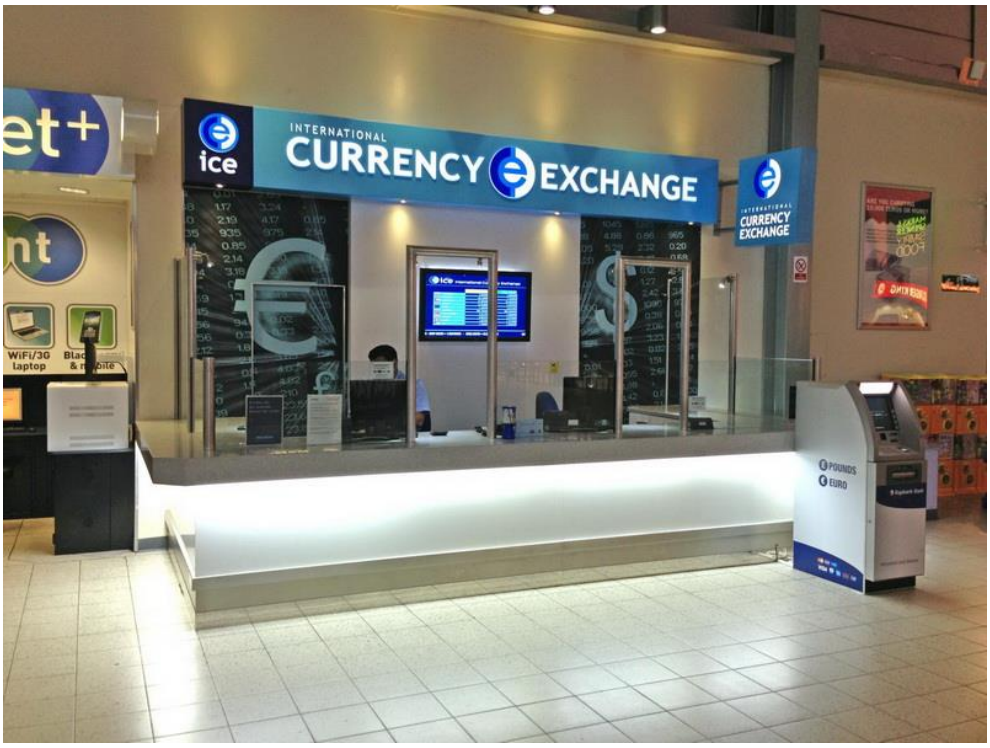

Рисунок 1 Пункт обмена валютой в торговом центре

Их можно увидеть внутри различных торговых центров и на приграничных зонах. Чаще всего они встречаются в торговых районах и рынках городов. Яркие, привлекающие глаз вывески и табло с актуальным курсом валют дадут понять, что перед вами один из таких пунктов.

Рисунок 2 Пункт обмена валюты в Санкт-Петербурге

Внутри таких зданий располагаются обычно от одной до нескольких касс. Клиент сможет обратиться в одну из свободных работающих касс, чтобы получить конкретные данные о нужной валюте. После того, как клиент определился с тем, какую валюту и в каком количестве он будет менять, он может произвести сделку. Работник пункта передает через специальное бронированное стекло или металлическую решётку для защиты персонала необходимое количество продаваемой валюты и забирает у клиента покупаемую валюту. После завершения транзакции, информация о ней, заносится в базы данных пункта.

Рисунок 3 Терминал - пункт обмена валют.

Терминалы - другой вид пунктов обмена валюты, они отличаются мобильностью, не занимают много места и требуют минимум обслуживающего персонала.

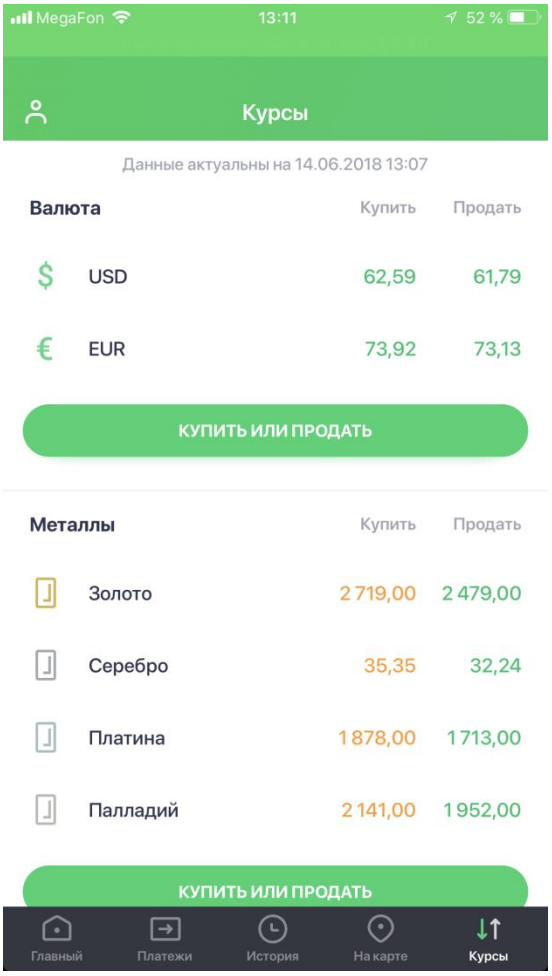

Рисунок 4 Интерфейс приложения "Сбербанк Онлайн" на телефоне платформы Андроид.

В наше время операцию по обмену валюты можно и произвести с мобильных телефонов, имеющих необходимые, поддерживаемые приложениями, характеристики.

1.2 Изучение существующих разработок

1.2.1 The World's Trusted Currency Authority

TWTCA - продукт компании Euronet Worldwide, это сайт предназначенный для отслеживания данных по стоимости валют и использования этой информации для коммерции, так же через сайт возможен перевод финансов и конвертация из одной валюты в другую.

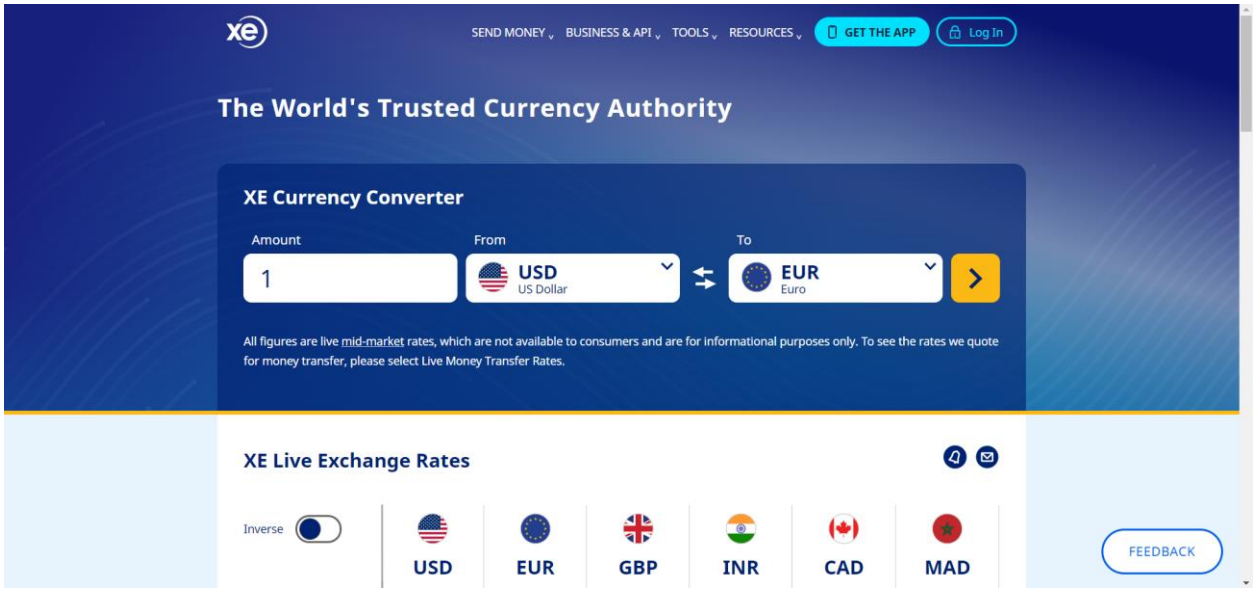

Рисунок 5. Главная страница.

Рисунок 6. Пример конвертации.

Как можно заметить из рисунка 2, конвертация на сайте происходит просто и быстро. В окно под надписью “Amount” пишется необходимая сумма переводимой валюты, окно правее при нажатии открывает меню для выбора переводимой валюты, а крайнее правое окно открывает меню для выбора валюты, в которую необходимо перевести. При нажатии на оранжевую кнопку происходит перевод и на экран выводится искомая информация.

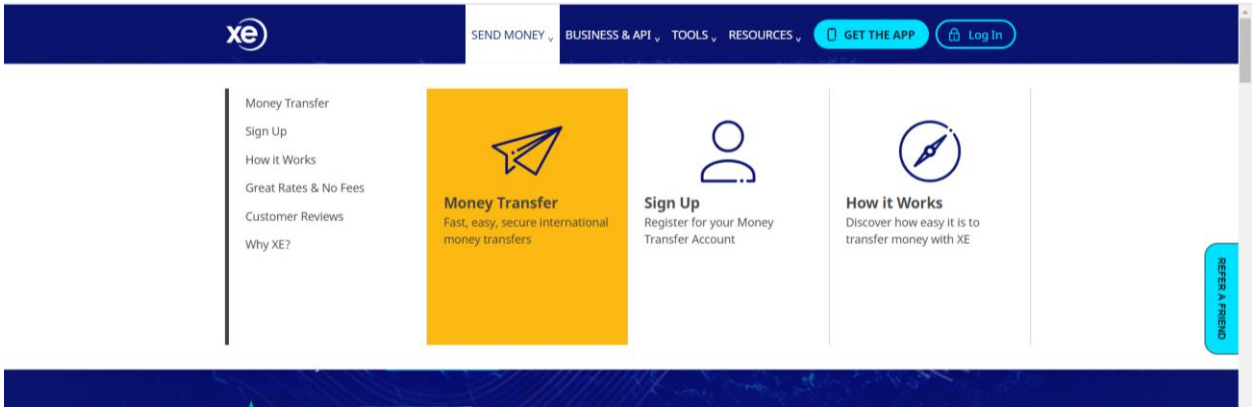

Рисунок 7. Перевод денег.

Для осуществления денежного перевода на сайте необходимо выполнить регистрацию с подтверждением электронной почты. Далее интуитивный интерфейс и подсказки позволят выполнить быстрый лёгкий и защищённый межнациональный перевод. Данный сайт предоставляет крайне удобный доступ к валютным статистическим данным и позволяет производить защищённые транзакции.

1.2.2 CurrencyXchanger.

CurrencyXchanger (CXR) – это приложение для ведения бизнеса. Оно предоставляет возможность автоматизированных переводов, ведения бухгалтерских учётов, отчетностей и других элементов работы бухгалтера. Приложение может быть использовано как на отдельных компьютерах, так и на целых компьютерных сетях, как локальных, так и сетевых, возможна функция передачи информации через облачную технологию. Главный офис разработчиков – Clear View Systems, расположен в Канаде, и благодаря их

труду, CXR набирает популярность. На данный момент программа используется более чем 350-тью компаниями ретейлеров и оптовиков. Разработчики CXR получили оценку A+ (высший балл), от ассоциации Better Business Bureau (BBB) за функционал.

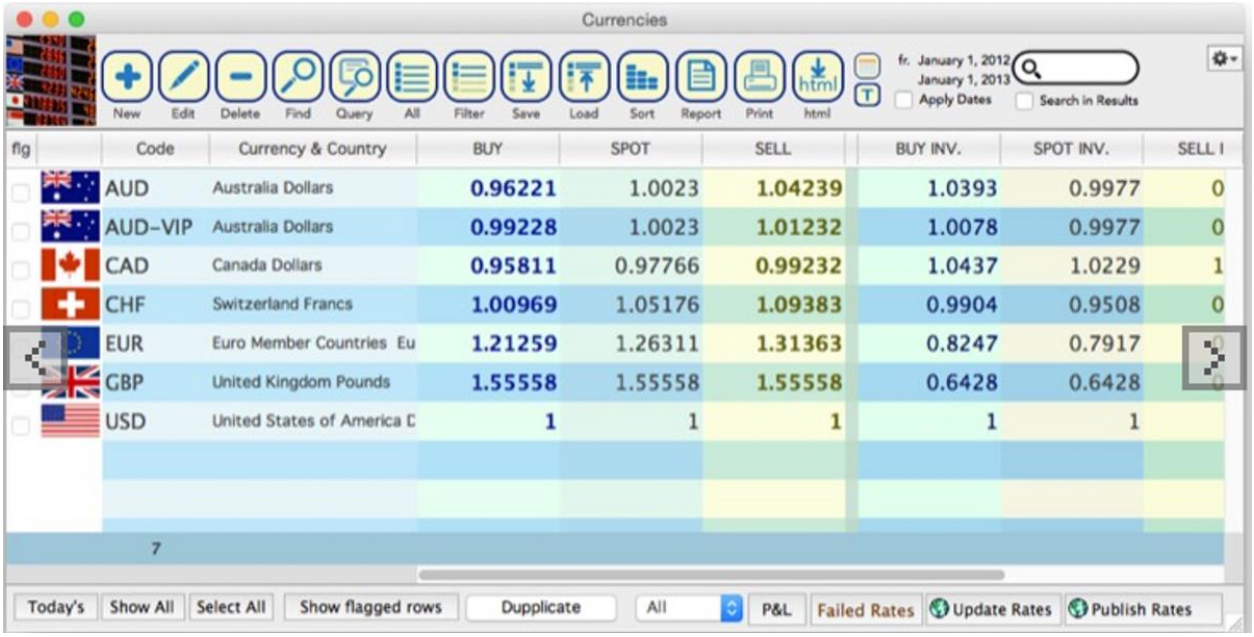

Рисунок 8 Интерфейс программы CXR на MacOS

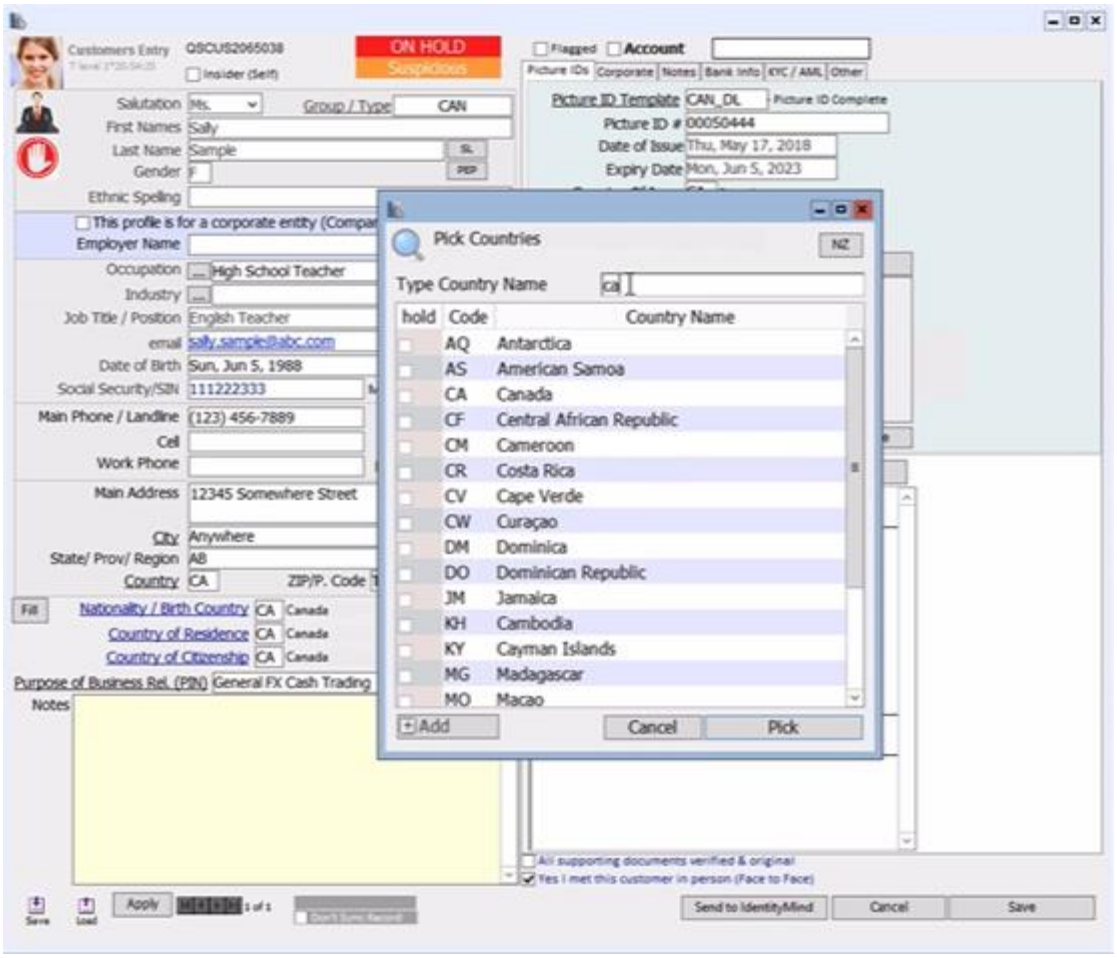

Рисунок 9 Пример регистрации клиента

Существует несколько версий CXR (Рисунок 6), находящиеся в разных ценовых категориях и имеющих соответствующий функционал.

Рисунок 10 Варианты пакетов установки CXR

Глава 2. Проектная часть

2.1 Построение диаграммы прецедентов

В данном разделе представлены две диаграммы прецедентов: на рисунке 7 показана диаграмма, объясняющая работу внутри пункта обмена валютой, рисунок 8 изображает взаимодействие с клиентом (трейдером). По этим диаграммам мы можем сопоставить функционал для каждого пользователя

Диаграмма прецедентов (рис. 7). Показывает, что рабочий имеет несколько выборов действия: просмотр данных по клиентам, сделкам, валюте, получение статистических данных, внесение данных в любую из предоставленных таблиц. Трейдер (покупатель), будучи клиентом компании, будет иметь доступ только к возможности создания новой сделки и просмотра данных о валюте.

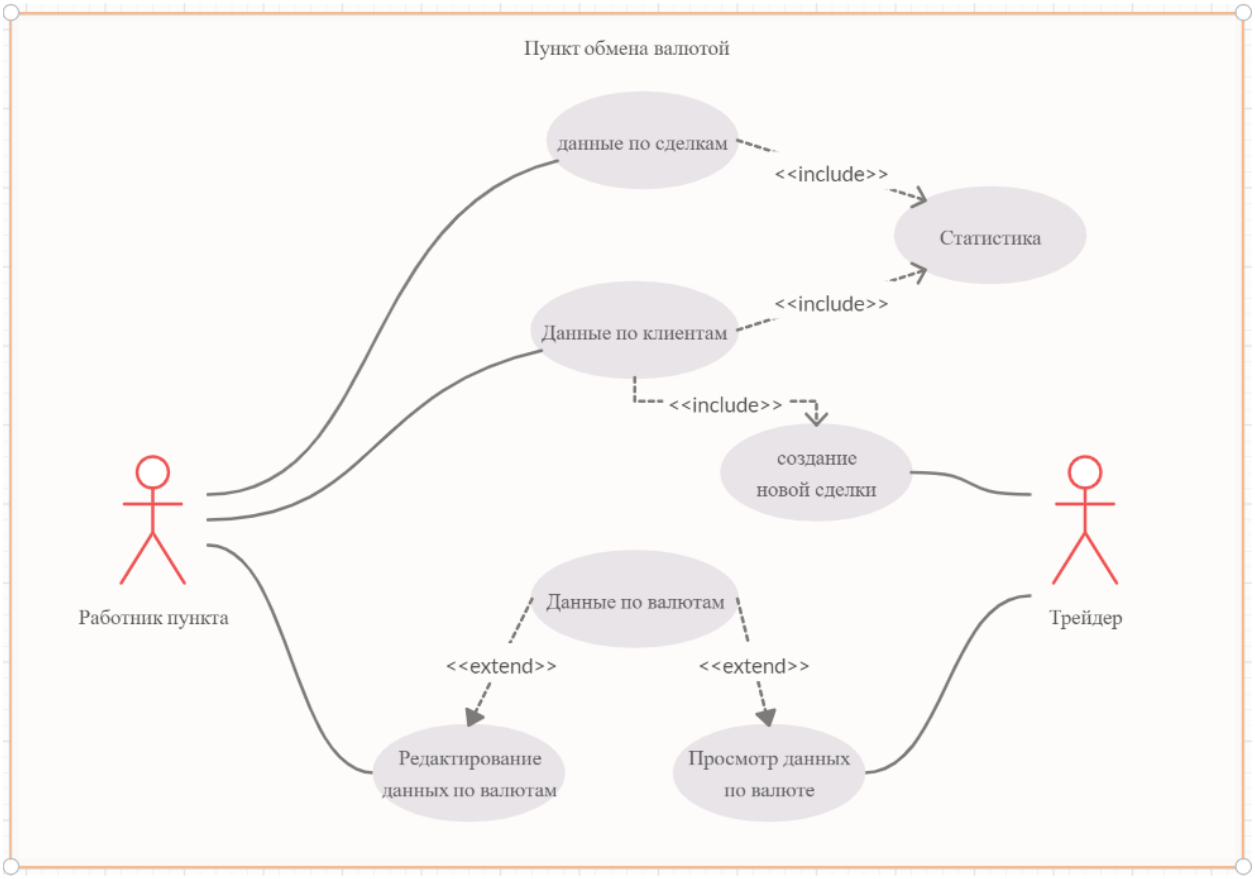

Рисунок 11 Диаграмма прецедентов в пункте обмена.

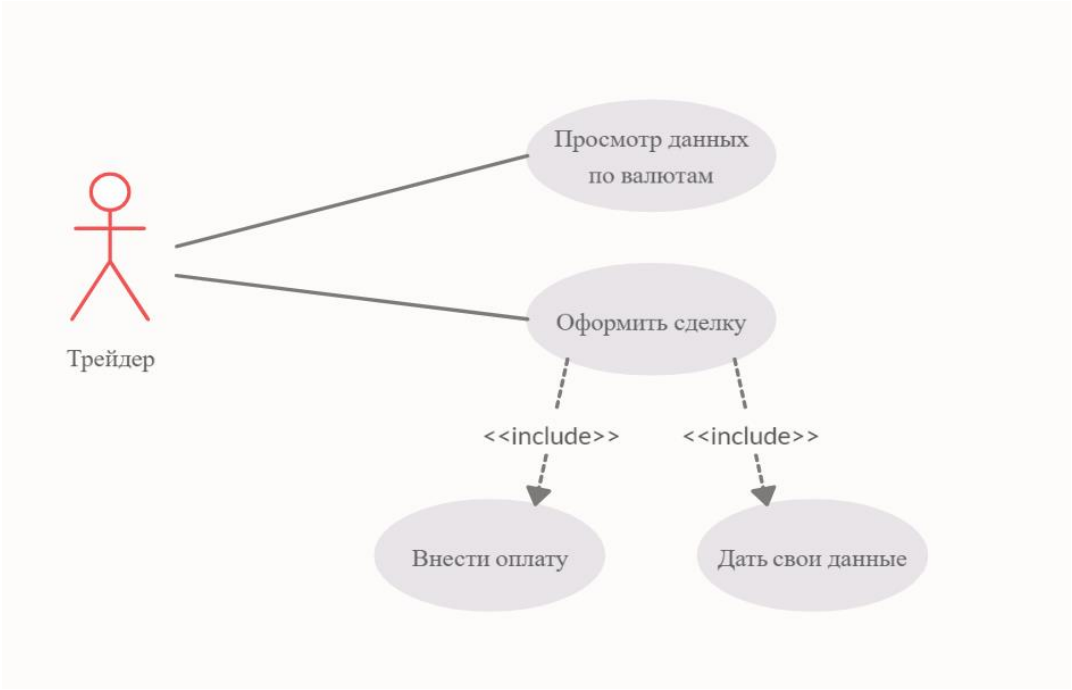

Рисунок 12 Диаграмма прецедентов для клиента (трейдера)

2.2 Выбор инструментов

При выборе инструментов я исходил из некоторых критериев, который я считал самыми главными в разработке. Важность критерия я выбирал из низкой, средней, высокой и крайне высокой.

Таблица №1 важность критерия:

Критерий	Участие в корпоративном проекте	Простота сопровождения	Наличие библиотек	Наличие документации на русском языке	Скорость разработки
Важность критерия	Средняя	Средняя	Высокая	Низкая	Крайне высокая

Исходя из этих критериев, выбор был поставлен между двумя языками программирования. Оценка критериев происходила по шкале от 0 до 10 баллов за критерий, где 0 наименьшее соответствие, 10 – наибольшее соответствие.

Таблица №2 сравнение языков программирования по критериям:

Критерий/Язык программирования	C++	Python
Участие в корпоративном проекте	10	8
Простота сопровождения	7	10
Наличие библиотек	6	10
Наличие документации на русском языке	8	6
Скорость разработки	6	10
Итого баллов	37	44

По результатам сравнения был выбран язык программирования Python.

2.3 Проектирование сценария

Ниже приведён сценарий использования программы работником пункта обмена валют.

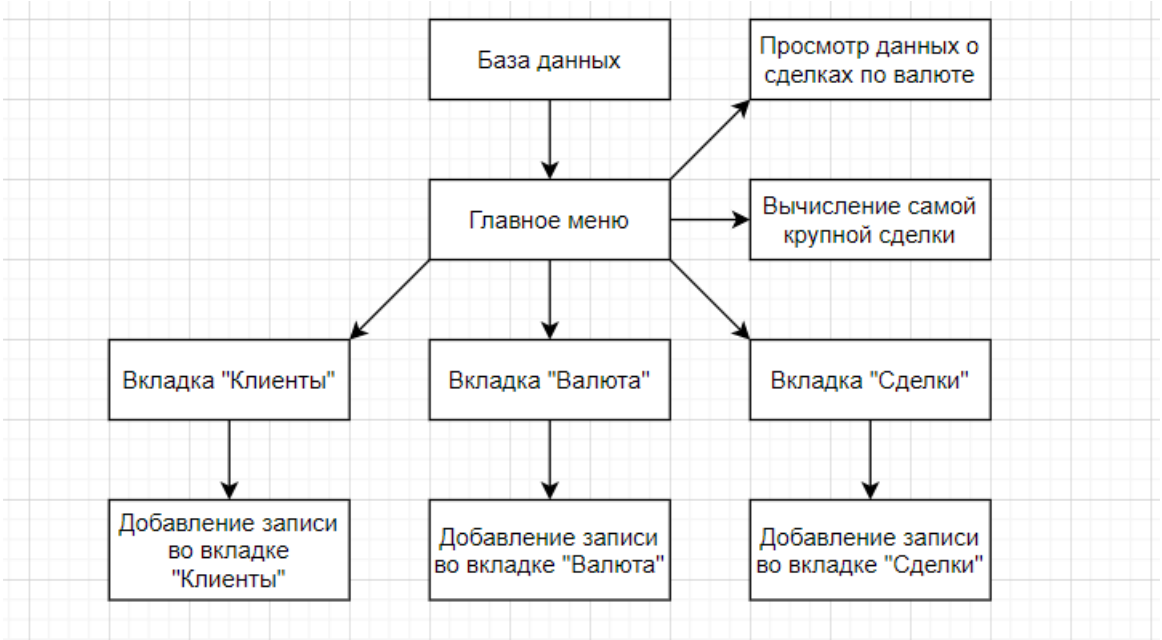

Рисунок 13. Сценарий программы

После запуска программы пользователь имеет несколько вариантов действий

1. Зайти во вкладку «Клиенты»;
2. Зайти во вкладку «Валюта»;
3. Зайти во вкладку «Сделки»;
4. Посмотреть данные о доле сделок повалютно;
5. Вычислить размер крупнейшей сделки;
6. Добавить запись в любую из перечисленных выше вкладок;

При добавлении записи, она автоматически помещается в созданный файл типа txt.

При выборе пунктов 4 и 5, информация будет выведена на консоль.

2.4 Построение диаграммы классов

В данном параграфе представлены все классы, которые используются в программе.

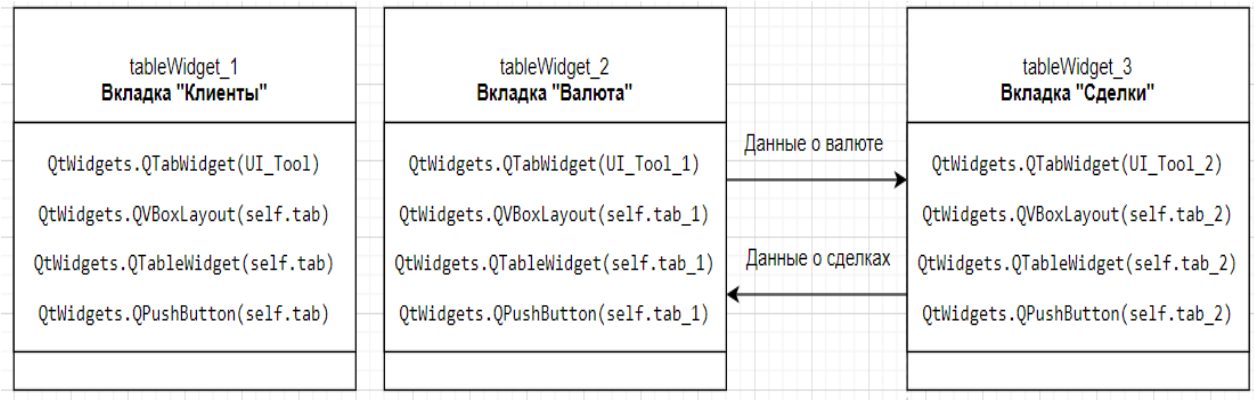

Рисунок 14 Диаграмма классов.

Класс tableWidget_1 оперирует данными из вкладки «Клиенты». Классы tableWidget_2 и tableWidget_3 позволяют взаимодействовать с вкладками программы «Валюта» и «Сделки» соответственно. Между ними существует связь, позволяющая производить вычисления, нажатием на кнопки внутри программы.

2.5 Описание главного модуля

В главный модуль разработки входит часть, отвечающая за правильную работу с всех компонентов «базы данных». Данный модуль состоит из кода, приведённого в листинге 1.

Листинг 1. Функция «retranslateUI».

```
def retranslateUi(self, TPayne_MySQL_Tool):
      #построчное прочтение файла для занесение данных в таблицу Клиенты
      for line_clients in db_file_clients.readlines():
            #метод split разделяет строчку на элементы
            splited = line_clients.split()
            #занесение данных из текстовых файлов в таблицу
            for i in range(0, 2):
                self.tableWidget_1.setItem(0, i, QTableWidgetItem(splited[i]))
            self.tableWidget_1.setItem(0, 2, QTableWidgetItem(splited[2]))
            self.tableWidget_1.insertRow(0)
      #Аналогичная работа. Данные заносятся в таблицу Валюты
      for line_curr in db_file_curr.readlines():
          splited = line_curr.split()
          for i in range(0, 2):
              self.tableWidget_2.setItem(0, i, QTableWidgetItem(splited[i]))
          self.tableWidget_2.setItem(0, 2, QTableWidgetItem(splited[2]))
          self.tableWidget_2.insertRow(0)
      #Аналогичная работа. Данные заносятся в таблицу Сделки
      for line_deals in db_file_deals.readlines():
          splited = line_deals.split()
          for i in range(0, 2):
              self.tableWidget_3.setItem(0, i, QTableWidgetItem(splited[i]))
          self.tableWidget_3.setItem(0, 2, QTableWidgetItem(splited[2]))
          self.tableWidget_3.setItem(0, 3, QTableWidgetItem(splited[3]))
          self.tableWidget_3.insertRow(0)
```

Данный метод позволяет занести информацию из текстовых файлов в окно программы. Блок-схема показана на рисунке 11.

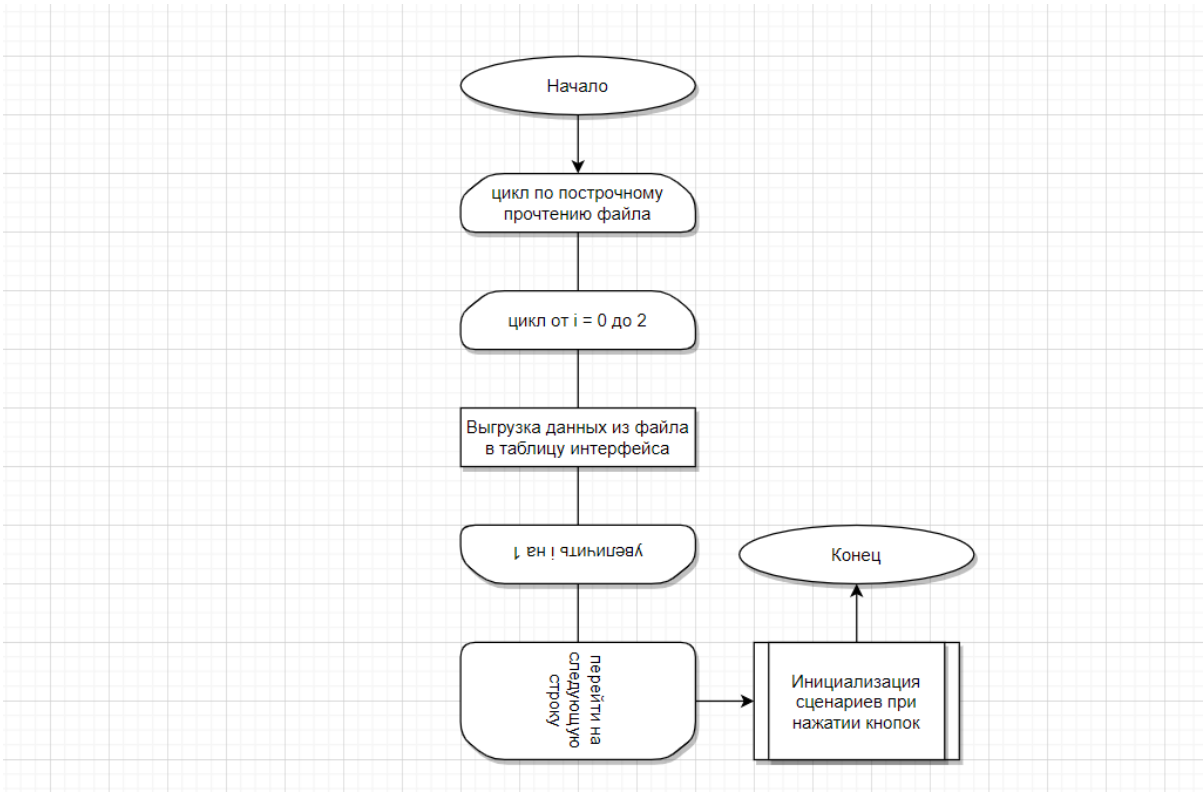

Рисунок 15 Главный модуль разработки.

2.6 Описание спецификаций к модулям

В данной главе будут описаны публичные члены модулей курсового проекта.

В коде публичным объектом является класс setupUI, который представляет собой главную форму модуля, на который закреплены объекты, предназначенные для взаимосвязи с «базой данных».

Ниже будут перечислены элементы модуля SetupUI, отвечающие за заполнение списков, и сортировку соответствующих таблиц.

1. SetupUI – объект интерфейса представленный в виде стека окон.
2. verticalLayout, verticalLayout_1, verticalLayout_2 – вертикальные менеджеры компоновки, позволяющие размещать и масштабировать объекты в зависимости от размера окна.
3. horizontalLayout, horizontalLayout_1… horizontalLayout_5 – горизонтальные менеджеры компоновки, позволяющие размещать и масштабировать объекты в зависимости от размера окна.
4. spacerItem – разделитель, который позволяет прижать виджет (элемент интерфейса) к определенному углу или выравнить его по центру. Этот спейсер прижимает кнопки в главном меню к правому краю
5. pushButton – кнопка, при нажатии на которую определяется доля сделок по каждой валюте
6. pushButton_2 – кнопка, при нажатии на которую определяется максимальный размер сделки в рублях.
7. pushButton_3, pushButton_4, pushButton_5 – кнопки добавления нового ряда в соответствующие таблицы.
8. tabWidget, tabWidget_2, tabWidget_3 – переключаемые вкладки
9. tableWidget_1, tableWidget_2, table_Widget_3 – таблицы на вкладки

2.7 Описание модулей

В данной главе мы разберем три главных модуля.

1. Центральный модуль

Данный модуль представляет собой набор обязательных элементов «базы данных» таких как таблицы, запросы и транзакции. И центральную функцию, которая представляет из себя настройку «базы данных».

2. Модуль работы с «базой данных».

Модуль представляет из себя несколько графических форм каждая из которых обращается в центральный модуль за своей таблицей из «базы данных» для дальнейшей работы с ней и подключает свою навигационную систему этой таблицы, что обеспечивает удобную работу. Так же этот модуль для удобства ввода имеет скрипт для подстановки соответствующих значений в выпадающие списки.

3. Модуль просмотра таблиц

Модуль представляет собой несколько графических форм, каждая из которых, обращается в центральный модуль за своим запросом из «базы данных» для дальнейшей его вывода, так же подключает навигацию к полученному запросу. Так же этот модуль имеет два скрипта, которые представляют из себя

1. Подстановку таблицу соответствующе значения
2. При выборе значения из списка производит изменение скрипта запроса и выводит новую отсортированную таблицу.

Листинг 2: Модуль просмотра таблицы:

```
def setupUi(self, UI_Tool):

    """setObjectName задаётся внутри программы QT Designer,
    позволяет дать имена объектам для более удобного использования"""
    UI_Tool.setObjectName("CEPDB")
    #resize отвечает за открытие окна программы в установленном разрешении
    UI_Tool.resize(610, 431)
    """ QGridLayout, QVBoxLayout - элементы модуля PyQt5,
    отвечающие за расстановку элементов внутри окна по сетке
    и их динамическое расширение при расширении окна программы """
    self.gridLayout = QtWidgets.QGridLayout(UI_Tool)
    self.gridLayout.setObjectName("gridLayout")
```

```
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        #QLabel - строка текста в окне программы
        self.label = QtWidgets.QLabel(UI_Tool)
        #QFont и setPointSize устанавливают шрифт и размер шрифта
        font = QtGui.QFont()
        font.setPointSize(11)
        """setFont, setLayoutDirection, setScaledContents,setAlignment
        отвечают за свойства и поведение текста в окне программы"""
        self.label.setFont(font)
        self.label.setLayoutDirection(QtCore.Qt.LeftToRight)
        self.label.setScaledContents(True)
        self.label.setAlignment(QtCore.Qt.AlignCenter)
        self.label.setObjectName("label")
        self.verticalLayout.addWidget(self.label)
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        #QTabWidget - виджет переключаемых вкладок
        self.tabWidget = QtWidgets.QTabWidget(UI_Tool)
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.verticalLayout_5 = QtWidgets.QVBoxLayout(self.tab)
        self.verticalLayout_5.setObjectName("verticalLayout_5")
        #QTableWidget - виджет таблицы
        self.tableWidget_1 = QtWidgets.QTableWidget(self.tab)
        self.tableWidget_1.setObjectName("tableWidget_1")
        #следующие 2 строки устанавливают колличество изначальных колонок и рядов
в таблице
        self.tableWidget_1.setColumnCount(3)
        self.tableWidget_1.setRowCount(1)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setVerticalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(1, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(2, item)
        self.tableWidget_1.horizontalHeader().setCascadingSectionResizes(True)
        self.tableWidget_1.horizontalHeader().setSortIndicatorShown(False)
        self.tableWidget_1.horizontalHeader().setStretchLastSection(False)
        self.tableWidget_1.verticalHeader().setVisible(False)
        self.verticalLayout_5.addWidget(self.tableWidget_1)
        #QPushButton – кнопка
        self.pushButton_3 = QtWidgets.QPushButton(self.tab)
        self.pushButton_3.setObjectName("pushButton_3")
        self.verticalLayout_5.addWidget(self.pushButton_3)
        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.tab_2)
        self.verticalLayout_4.setObjectName("verticalLayout_4")
        self.tableWidget_2 = QtWidgets.QTableWidget(self.tab_2)
        self.tableWidget_2.setObjectName("tableWidget_2")
```

```
self.tableWidget_2.setColumnCount(3)
self.tableWidget_2.setRowCount(1)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setVerticalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(1, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(2, item)
self.tableWidget_2.verticalHeader().setVisible(False)
self.verticalLayout_4.addWidget(self.tableWidget_2)
self.pushButton_4 = QtWidgets.QPushButton(self.tab_2)
self.pushButton_4.setObjectName("pushButton_4")
self.verticalLayout_4.addWidget(self.pushButton_4)
self.tabWidget.addTab(self.tab_2, "")
self.tab_3 = QtWidgets.QWidget()
self.tab_3.setObjectName("tab_3")
self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.tab_3)
self.verticalLayout_3.setObjectName("verticalLayout_3")
self.tableWidget_3 = QtWidgets.QTableWidget(self.tab_3)
self.tableWidget_3.setObjectName("tableWidget_3")
self.tableWidget_3.setColumnCount(4)
self.tableWidget_3.setRowCount(1)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setVerticalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(1, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(2, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(3, item)
self.tableWidget_3.verticalHeader().setVisible(False)
self.verticalLayout_3.addWidget(self.tableWidget_3)
self.pushButton_5 = QtWidgets.QPushButton(self.tab_3)
self.pushButton_5.setObjectName("pushButton_5")
self.verticalLayout_3.addWidget(self.pushButton_5)
self.tabWidget.addTab(self.tab_3, "")
self.horizontalLayout.addWidget(self.tabWidget)
self.verticalLayout.addLayout(self.horizontalLayout)
self.gridLayout.addLayout(self.verticalLayout, 0, 0, 1, 1)
self.pushButton = QtWidgets.QPushButton(UI_Tool)
self.pushButton.setAutoDefault(False)
self.pushButton.setDefault(False)
self.pushButton.setFlat(False)
self.pushButton.setObjectName("pushButton")
self.gridLayout.addWidget(self.pushButton, 1, 0, 1, 1)
self.pushButton_2 = QtWidgets.QPushButton(UI_Tool)
self.pushButton_2.setObjectName("pushButton_2")
self.gridLayout.addWidget(self.pushButton_2, 2, 0, 1, 1)

self.retranslateUi(UI_Tool)
self.tabWidget.setCurrentIndex(0)
```

```
    QtCore.QMetaObject.connectSlotsByName(UI_Tool)

#функция retranslateUi устанавливает текст на элементы окна
def retranslateUi(self, TPayne_MySQL_Tool):
    _translate = QtCore.QCoreApplication.translate
    #setWindowTitle - название окна (верхний левый угол при открытии)
    TPayne_MySQL_Tool.setWindowTitle(_translate("TPayne_MySQL_Tool", "CEPDB
Manager"))
    #setText устанавливает текст на какой - либо элемент
    self.label.setText(_translate("TPayne_MySQL_Tool", "База данных пункта
обмена валютой"))
    #HeaderItem устанавливает текст на главные колонки и ряды таблиц
    item = self.tableWidget_1.verticalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "New Row"))
    item = self.tableWidget_1.horizontalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "Код клиента"))
    item = self.tableWidget_1.horizontalHeaderItem(1)
    item.setText(_translate("TPayne_MySQL_Tool", "ФИО"))
    item = self.tableWidget_1.horizontalHeaderItem(2)
    item.setText(_translate("TPayne_MySQL_Tool", "Паспорт"))
    self.pushButton_3.setText(_translate("TPayne_MySQL_Tool", "Добавить ряд
для записи"))
    self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab),
_translate("TPayne_MySQL_Tool", "Клиенты"))
    item = self.tableWidget_2.verticalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "1"))
    item = self.tableWidget_2.horizontalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "Код валюты"))
    item = self.tableWidget_2.horizontalHeaderItem(1)
    item.setText(_translate("TPayne_MySQL_Tool", "Название"))
    item = self.tableWidget_2.horizontalHeaderItem(2)
    item.setText(_translate("TPayne_MySQL_Tool", "Курс продажи"))
    self.pushButton_4.setText(_translate("TPayne_MySQL_Tool", "Добавить ряд
для записи"))
    self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2),
_translate("TPayne_MySQL_Tool", "Валюта"))
    item = self.tableWidget_3.verticalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "New Row"))
    item = self.tableWidget_3.horizontalHeaderItem(0)
    item.setText(_translate("TPayne_MySQL_Tool", "Код сделки"))
    item = self.tableWidget_3.horizontalHeaderItem(1)
    item.setText(_translate("TPayne_MySQL_Tool", "Код клиента"))
    item = self.tableWidget_3.horizontalHeaderItem(2)
    item.setText(_translate("TPayne_MySQL_Tool", "Код валюты"))
    item = self.tableWidget_3.horizontalHeaderItem(3)
    item.setText(_translate("TPayne_MySQL_Tool", "Сумма"))
    self.pushButton_5.setText(_translate("TPayne_MySQL_Tool", "Добавить ряд
для записи"))
    self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3),
_translate("TPayne_MySQL_Tool", "Сделки"))
    self.pushButton.setText(_translate("TPayne_MySQL_Tool", "Определить долю
сделок по каждой валюте"))
    self.pushButton_2.setText(_translate("TPayne_MySQL_Tool", "Определить
максимальный размер сделки в рублях"))
```

2.8 Анализ оптимальности использования памяти и быстродействия

В данном разделе будет проведен анализ оптимальности использования памяти и быстродействия программы.

Список принятых оптимальных решений:

1. Подключение некоторых модулей внутри функций/методов.

В данном проекте некоторые модули были подключены не в весь модуль, а только в функции/методы, которые его используют. Сделано это, потому что работа с локальными объектами быстрее работы с глобальными объектами, к тому же импортироваться эти модули будут только при срабатывании этих функций, что ускорит запуск программы.

2.9 Описание тестовых наборов модулей

В этом разделе показано умение применять средства отладки.

В ходе написания курсового проекта при попытке запустить скрипт, окно программы было принуждённо закрыто и было получено данное сообщение:

```
"C:\Program Files (x86)\Python37-32\python.exe" C:/Users/Prog/Desktop/KYPCA4/ready.py

Process finished with exit code -1073740791 (0xC0000409)
```

Рисунок 16 До применения средств отладки

После получения данного сообщения были просмотрены 344 строка модуля ready.py и была обнаружена ошибка, которая впоследствии была устранена, а после попытки запуска скрипта получено данное сообщение:

Рисунок 17. После применения средств отладки

Это означает, что ошибка была устранена и скрипт запустился.

Глава 3. Эксплуатационная часть

3.1. Руководство оператора

Аннотация.

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «Currency Exchange Point», предназначенной для облегчения работы пунктами обменом валюты.

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды.

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [1)], ГОСТ 19.103-77 [2)], ГОСТ 19.104-78* [3)], ГОСТ 19.105-78* [4)], ГОСТ 19.106-78* [5)], ГОСТ 19.505-79* [6)], ГОСТ 19.604-78* [7)]).

[1)] ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов
[2)] ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов
[3)] ГОСТ 19.104-78* ЕСПД. Основные надписи
[4)] ГОСТ 19.105-78* ЕСПД. Общие требования к программным документам
[5)] ГОСТ 19.106-78* ЕСПД. Общие требования к программным документам, выполненным печатным способом
[6)] ГОСТ 19.505-79* ЕСПД. Руководство оператора. Требования к содержанию и оформлению
[7)] ГОСТ 19.604-78* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом

1.1. Функциональное назначение программы

Специальное программное обеспечение «Currency Exchange Point» используется для управления записями, информацией о клиентах, валюте и сделках.

1.2. Эксплуатационное назначение программы

Специальное программное обеспечение «Currency Exchange Point» может эксплуатироваться на объектах любого масштаба в сфере финансового бизнеса для облегчения работы персонала.

1.3. Состав функций

1. Функция смены таблицы. Эта функция позволяет менять таблицу в зависимости от необходимости;
2. Функция добавления элемента в таблицу. Эта функция позволяет добавлять нужную информацию в таблицу;
3. Функция высчитывания наибольшей сделки;
4. Функция вывода доли по сделкам;

2. Условия выполнения программы

2.1. Минимальный состав аппаратных средств

ОС: Windows 10

Процессор: Как минимум 1 ГГц или SoC.

ОЗУ: 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем).

Место на жестком диске: 20 мб.

Видеоадаптер: DirectX версии не ниже 9 с драйвером WDDM 1.0.

Дисплей: 800 x 600.

2.2. Минимальный состав программных средств

Дополнительные программные средства не требуются.

2.3. Требование к персоналу (пользователю)

Конечный пользователь программы должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы.

3. Выполнение программы

3.1. Загрузка и запуск программы

При запуске программы пользователь сразу же может наблюдать на экране окно программы. В окне таблицы уже может быть занесена информация, если заранее уже были созданы файлы database_clients.txt, database_curr.txt, database_deals.txt и в них уже была занесена какая-либо информация. В противном случае эти файлы автоматически создаются программой при первом открытии и не имеют никаких записей при открытии. В случае, представленном на рисунке 12, данные уже были занесены в текстовые файлы.

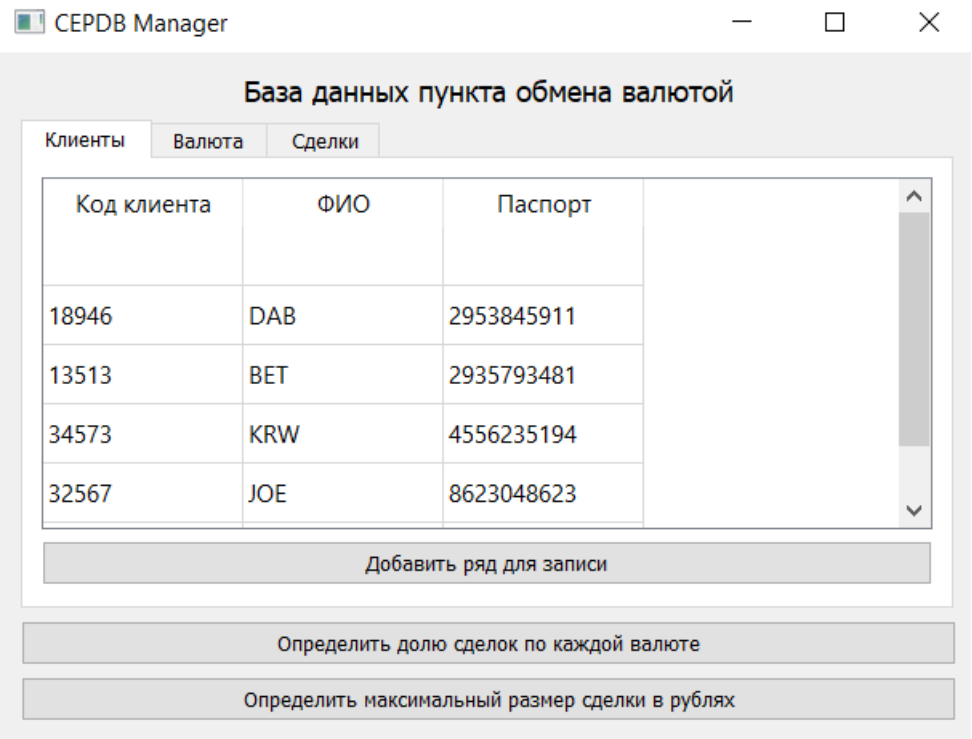

Рисунок 18 Начальный экран программы

3.2. Выполнение программы

Для перемещения по вкладкам «базы данных» необходимо навести курсор на одну из надписей «Валюта», «Сделки», или «Клиенты» и нажать левую кнопку мыши. По умолчанию программа открывается на вкладке «Клиенты», поэтому чтобы посмотреть «базу данных» по клиентам при запуске, на эту вкладку нажимать не требуется.

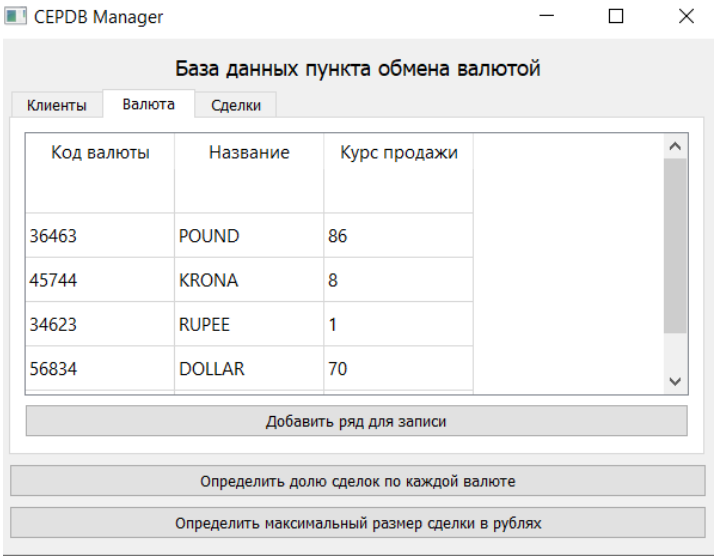

Рисунок 19 Открыта вкладка «Валюта»

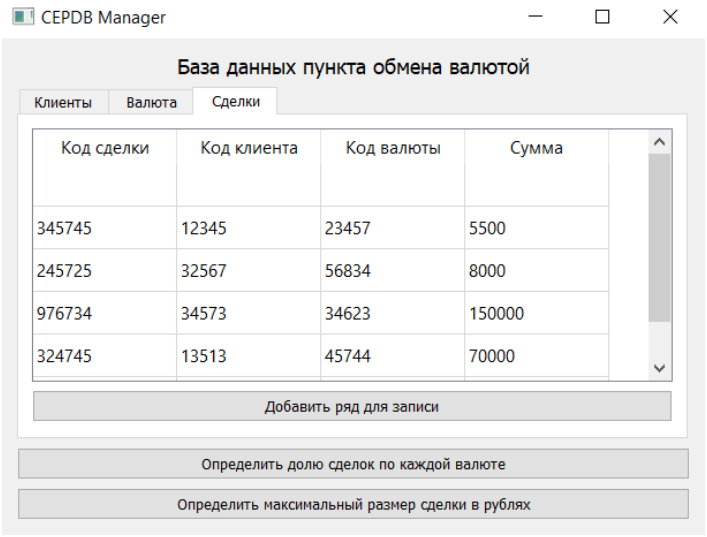

Рисунок 20 Открыта вкладка «Сделки»

3.3. Функционал интерфейса.

Интерфейс программы имеет возможность растягиваться до необходимых размеров. При наведении курсора на один из углов окна программы и появления подсказки (курсор превращается в линию со стрелками на каждом краю, направленную на 45 градусов от нижней границы экрана монитора) и зажатия левой кнопки мыши с последующим передвижения курсора в произвольном направлении, можно заметить, что

элементы окна (кроме текста) будут динамически расширяться в направлении перемещения курсора.

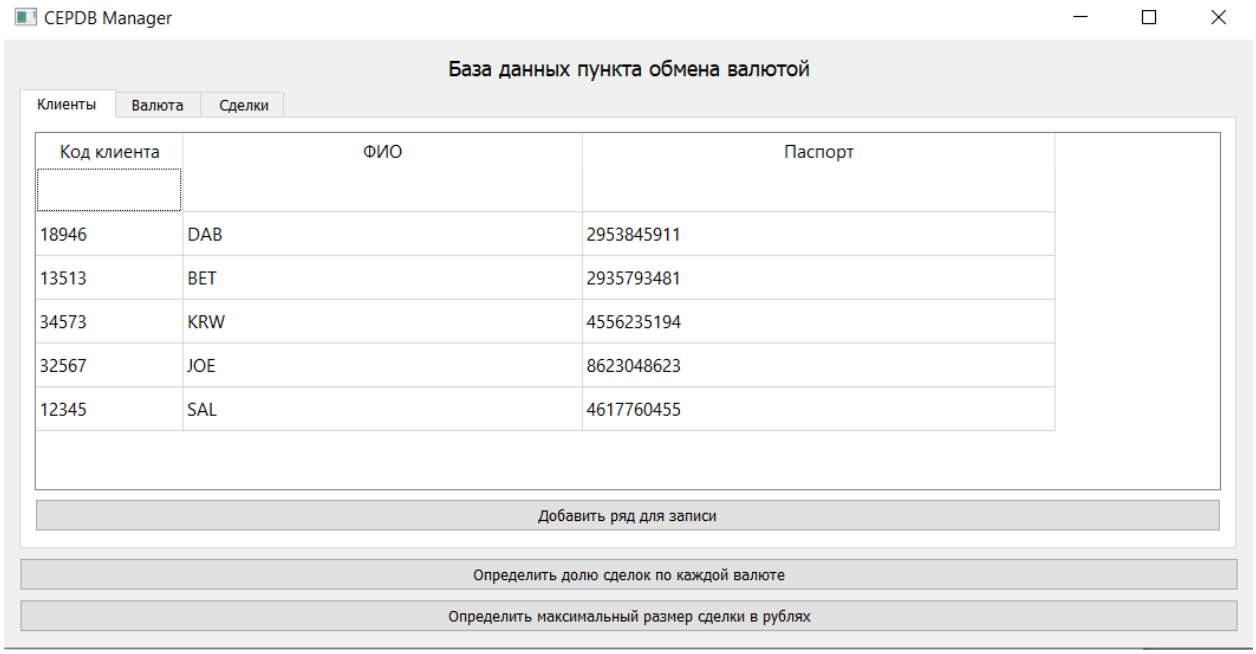

Рисунок 21 Изменённое окно программы

3.4. Функционал кнопок.

3.4.1. Кнопка «Добавить ряд для записи».

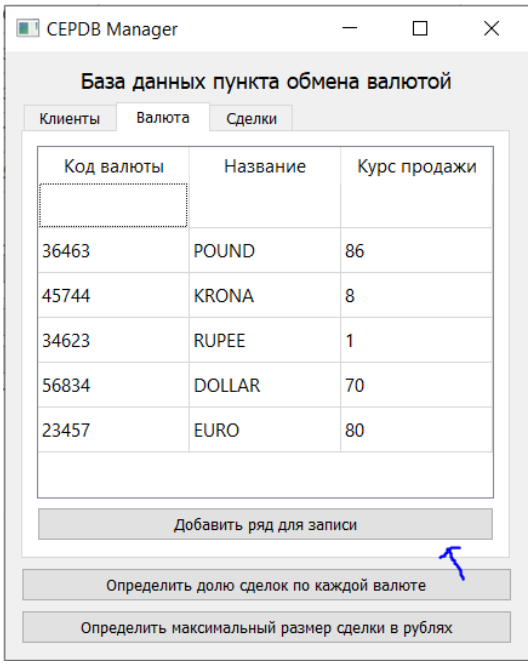

Рисунок 22 Кнопка «Добавить ряд для записи» во вкладке «Валюта»

Данная кнопка расположена в каждой из вкладок и имеет один функционал. Для правильной работы необходимо сначала вписать данные (показано на рисунке 17) в верхний ряд таблицы и только после этого произвести нажатие (показано на рисунке 18).

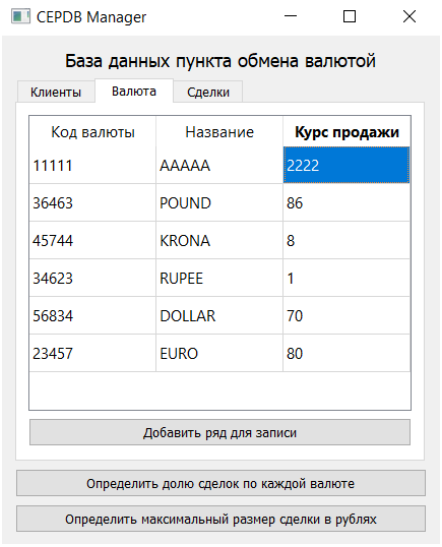

Рисунок 23 шаг 1. Ввод данных.

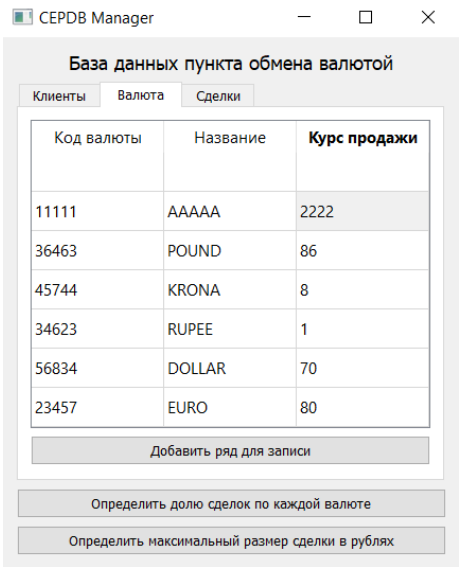

Рисунок 24 шаг 2. Нажатие на кнопку

После выполнения перечисленных действий, строка будет успешно занесена в текстовый файл (Рисунок 19)

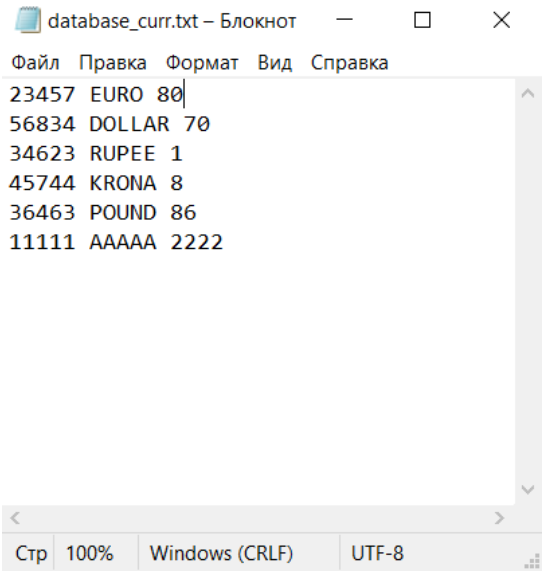

Рисунок 25 Содержимое текстового файла после выполнения операции

Приведённые выше действия имеют аналогичный результат и для других вкладок.

3.4.2. Кнопка «Определить долю сделок по каждой валюте»

При нажатии на кнопку «Определить долю сделок по каждой валюте» на консоли, возникающей при открытии программы, будет выведен список. Поочерёдно на консоли выводится название валюты из «базы данных», за ней идёт доля сделок по этой валюте, относительно общей суммы сделок.

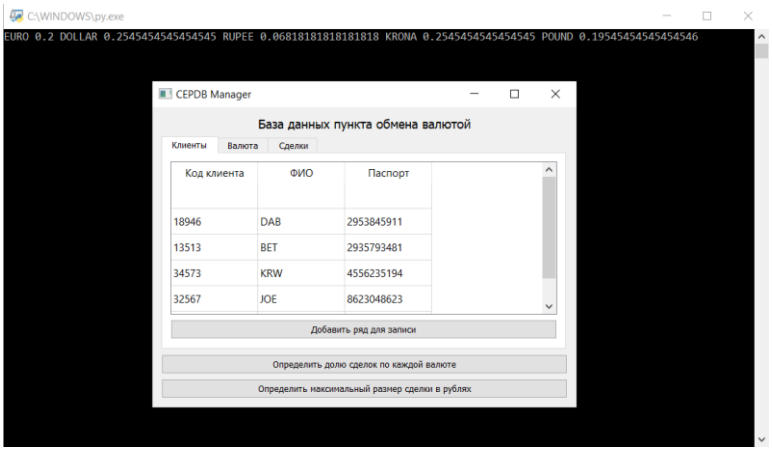

Рисунок 26 Пример работы кнопки "Определить долю сделок по каждой валюте"

3.4.3. Кнопка «Определить максимальный размер сделки в рублях»

При нажатии на кнопку «Определить максимальный размер сделки в рублях» на консоли, возникающей при открытии программы, будет выведен максимальный размер сделки в рублях.

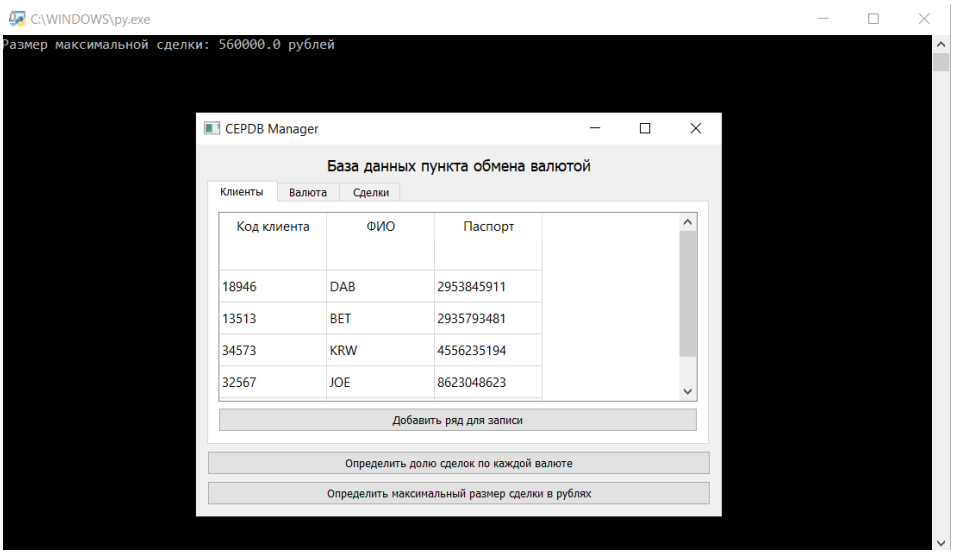

Рисунок 27 Пример работы кнопки "Определить максимальный размер сделки в рублях"

3.4.4. Кнопки внешнего интерфейса.

Рисунок 28 Кнопки внешнего интерфейса

На рисунке 22 представлены кнопки внешнего интерфейса. Их функционал слева-направо: свернуть окно программы в трей, развернуть окно программы на весь экран, закрыть окно программы (при этом завершается работа программы).

Заключение.

В результате выполнения курсового проекта была написана программа «Currency Exchange Point» для упрощения ведения бизнеса в сфере валютных бирж, а именно для автоматизации работы.

В ходе работы были проанализированы предметная область, существующие разработки, посвященные данному направлению, получены практические навыки по созданию UI с помощью библиотеки PyQt5 и приложения QT Designer.

Список литературы и интернет-источников.

1. Сайт для создания диаграмм прецедентов “Creately”.
   https://creately.com/ru/
2. Сайт для создания блок-схем “Draw.io”.
   https://app.diagrams.net/
3. Сайт, используемый как один из примеров существующих разработок:
   “The World’s Trusted Currency Authority”
   https://www.xe.com/

Приложение 1. Код программы.

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QApplication, QWidget, QTableWidgetItem
import sys

class database_ui(QtWidgets.QWidget):

    def __init__(self):
        QtWidgets.QWidget.__init__(self)
        self.setupUi(self)

    def setupUi(self, UI_Tool):
        UI_Tool.setObjectName("TPayne_MySQL_Tool")
        UI_Tool.resize(610, 431)
        self.gridLayout = QtWidgets.QGridLayout(UI_Tool)
        self.gridLayout.setObjectName("gridLayout")
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.label = QtWidgets.QLabel(UI_Tool)
        font = QtGui.QFont()
        font.setPointSize(11)
        self.label.setFont(font)
        self.label.setLayoutDirection(QtCore.Qt.LeftToRight)
        self.label.setScaledContents(True)
        self.label.setAlignment(QtCore.Qt.AlignCenter)
        self.label.setObjectName("label")
        self.verticalLayout.addWidget(self.label)
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.tabWidget = QtWidgets.QTabWidget(UI_Tool)
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.verticalLayout_5 = QtWidgets.QVBoxLayout(self.tab)
        self.verticalLayout_5.setObjectName("verticalLayout_5")
        self.tableWidget_1 = QtWidgets.QTableWidget(self.tab)
        self.tableWidget_1.setObjectName("tableWidget_1")
        self.tableWidget_1.setColumnCount(3)
        self.tableWidget_1.setRowCount(1)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setVerticalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(1, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget_1.setHorizontalHeaderItem(2, item)

self.tableWidget_1.horizontalHeader().setCascadingSectionResizes(True)
        self.tableWidget_1.horizontalHeader().setSortIndicatorShown(False)
        self.tableWidget_1.horizontalHeader().setStretchLastSection(False)
        self.tableWidget_1.verticalHeader().setVisible(False)
        self.verticalLayout_5.addWidget(self.tableWidget_1)
        self.pushButton_3 = QtWidgets.QPushButton(self.tab)
```

```
self.pushButton_3.setObjectName("pushButton_3")
self.verticalLayout_5.addWidget(self.pushButton_3)
self.tabWidget.addTab(self.tab, "")
self.tab_2 = QtWidgets.QWidget()
self.tab_2.setObjectName("tab_2")
self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.tab_2)
self.verticalLayout_4.setObjectName("verticalLayout_4")
self.tableWidget_2 = QtWidgets.QTableWidget(self.tab_2)
self.tableWidget_2.setObjectName("tableWidget_2")
self.tableWidget_2.setColumnCount(3)
self.tableWidget_2.setRowCount(1)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setVerticalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(1, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_2.setHorizontalHeaderItem(2, item)
self.tableWidget_2.verticalHeader().setVisible(False)
self.verticalLayout_4.addWidget(self.tableWidget_2)
self.pushButton_4 = QtWidgets.QPushButton(self.tab_2)
self.pushButton_4.setObjectName("pushButton_4")
self.verticalLayout_4.addWidget(self.pushButton_4)
self.tabWidget.addTab(self.tab_2, "")
self.tab_3 = QtWidgets.QWidget()
self.tab_3.setObjectName("tab_3")
self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.tab_3)
self.verticalLayout_3.setObjectName("verticalLayout_3")
self.tableWidget_3 = QtWidgets.QTableWidget(self.tab_3)
self.tableWidget_3.setObjectName("tableWidget_3")
self.tableWidget_3.setColumnCount(4)
self.tableWidget_3.setRowCount(1)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setVerticalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(0, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(1, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(2, item)
item = QtWidgets.QTableWidgetItem()
self.tableWidget_3.setHorizontalHeaderItem(3, item)
self.tableWidget_3.verticalHeader().setVisible(False)
self.verticalLayout_3.addWidget(self.tableWidget_3)
self.pushButton_5 = QtWidgets.QPushButton(self.tab_3)
self.pushButton_5.setObjectName("pushButton_5")
self.verticalLayout_3.addWidget(self.pushButton_5)
self.tabWidget.addTab(self.tab_3, "")
self.horizontalLayout.addWidget(self.tabWidget)
self.verticalLayout.addLayout(self.horizontalLayout)
self.gridLayout.addLayout(self.verticalLayout, 0, 0, 1, 1)
self.pushButton = QtWidgets.QPushButton(UI_Tool)
self.pushButton.setAutoDefault(False)
self.pushButton.setDefault(False)
```

```
        self.pushButton.setFlat(False)
        self.pushButton.setObjectName("pushButton")
        self.gridLayout.addWidget(self.pushButton, 1, 0, 1, 1)
        self.pushButton_2 = QtWidgets.QPushButton(UI_Tool)
        self.pushButton_2.setObjectName("pushButton_2")
        self.gridLayout.addWidget(self.pushButton_2, 2, 0, 1, 1)

        self.retranslateUi(UI_Tool)
        self.tabWidget.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(UI_Tool)

    def retranslateUi(self, TPayne_MySQL_Tool):
        _translate = QtCore.QCoreApplication.translate
        TPayne_MySQL_Tool.setWindowTitle(_translate("TPayne_MySQL_Tool",
"CEPDB Manager"))
        self.label.setText(_translate("TPayne_MySQL_Tool", "База данных
пункта обмена валютой"))
        item = self.tableWidget_1.verticalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "New Row"))
        item = self.tableWidget_1.horizontalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "Код клиента"))
        item = self.tableWidget_1.horizontalHeaderItem(1)
        item.setText(_translate("TPayne_MySQL_Tool", "ФИО"))
        item = self.tableWidget_1.horizontalHeaderItem(2)
        item.setText(_translate("TPayne_MySQL_Tool", "Паспорт"))
        self.pushButton_3.setText(_translate("TPayne_MySQL_Tool", "Добавить
ряд для записи"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab),
_translate("TPayne_MySQL_Tool", "Клиенты"))
        item = self.tableWidget_2.verticalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "1"))
        item = self.tableWidget_2.horizontalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "Код валюты"))
        item = self.tableWidget_2.horizontalHeaderItem(1)
        item.setText(_translate("TPayne_MySQL_Tool", "Название"))
        item = self.tableWidget_2.horizontalHeaderItem(2)
        item.setText(_translate("TPayne_MySQL_Tool", "Курс продажи"))
        self.pushButton_4.setText(_translate("TPayne_MySQL_Tool", "Добавить
ряд для записи"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2),
_translate("TPayne_MySQL_Tool", "Валюта"))
        item = self.tableWidget_3.verticalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "New Row"))
        item = self.tableWidget_3.horizontalHeaderItem(0)
        item.setText(_translate("TPayne_MySQL_Tool", "Код сделки"))
        item = self.tableWidget_3.horizontalHeaderItem(1)
        item.setText(_translate("TPayne_MySQL_Tool", "Код клиента"))
        item = self.tableWidget_3.horizontalHeaderItem(2)
        item.setText(_translate("TPayne_MySQL_Tool", "Код валюты"))
        item = self.tableWidget_3.horizontalHeaderItem(3)
        item.setText(_translate("TPayne_MySQL_Tool", "Сумма"))
        self.pushButton_5.setText(_translate("TPayne_MySQL_Tool", "Добавить
ряд для записи"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3),
_translate("TPayne_MySQL_Tool", "Сделки"))
```

```
        self.pushButton.setText(_translate("TPayne_MySQL_Tool", "Определить
долю сделок по каждой валюте"))
        self.pushButton_2.setText(_translate("TPayne_MySQL_Tool", "Определить
максимальный размер сделки в рублях"))

        db_file_clients = open('database_clients.txt', 'r')
        db_file_curr = open('database_curr.txt', 'r')
        db_file_deals = open('database_deals.txt', 'r')

        for line_clients in db_file_clients.readlines():
            splited = line_clients.split()
            for i in range(0, 2):
                self.tableWidget_1.setItem(0, i,
QTableWidgetItem(splited[i]))
            self.tableWidget_1.setItem(0, 2, QTableWidgetItem(splited[2]))
            self.tableWidget_1.insertRow(0)

        for line_curr in db_file_curr.readlines():
            splited = line_curr.split()
            for i in range(0, 2):
                self.tableWidget_2.setItem(0, i,
QTableWidgetItem(splited[i]))
            self.tableWidget_2.setItem(0, 2, QTableWidgetItem(splited[2]))
            self.tableWidget_2.insertRow(0)

        for line_deals in db_file_deals.readlines():
            splited = line_deals.split()
            for i in range(0, 2):
                self.tableWidget_3.setItem(0, i,
QTableWidgetItem(splited[i]))
            self.tableWidget_3.setItem(0, 2, QTableWidgetItem(splited[2]))
            self.tableWidget_3.setItem(0, 3, QTableWidgetItem(splited[3]))
            self.tableWidget_3.insertRow(0)

        db_file_clients.close()
        db_file_curr.close()
        db_file_deals.close()

        self.pushButton.pressed.connect(self.totaldeals)
        self.pushButton_2.pressed.connect(self.maxdeal)
        self.pushButton_3.pressed.connect(self.addrow_1)
        self.pushButton_4.pressed.connect(self.addrow_2)
        self.pushButton_5.pressed.connect(self.addrow_3)

    def addrow_1(self):
        db_file_clients = open('database_clients.txt', 'a')
        for i in range(0, 3):
            temp = self.tableWidget_1.item(0, i).text()
            db_file_clients.write(temp)
            db_file_clients.write(' ')
        db_file_clients.write('\n')
        db_file_clients.close()
        self.tableWidget_1.insertRow(0)

    def addrow_2(self):
```

```
        db_file_curr = open('database_curr.txt', 'a')
        for i in range(0, 3):
            temp = self.tableWidget_2.item(0, i).text()
            db_file_curr.write(temp)
            db_file_curr.write(' ')
        db_file_curr.write('\n')
        db_file_curr.close()
        self.tableWidget_2.insertRow(0)

    def addrow_3(self):
        db_file_deals = open('database_deals.txt', 'a')
        for i in range(0, 4):
            temp = self.tableWidget_3.item(0, i).text()
            db_file_deals.write(temp)
            db_file_deals.write(' ')
        db_file_deals.write('\n')
        db_file_deals.close()
        self.tableWidget_3.insertRow(0)

    def totaldeals(self):

        db_file_curr = open('database_curr.txt', 'r')
        db_file_deals = open('database_deals.txt', 'r')
        exit_list = []
        deals_lines = []
        curr_lines = []
        overall = 0

        for line_deals in db_file_deals.readlines():
            splited_deals = line_deals.split()
            deals_lines.append(splited_deals)

        for line_curr in db_file_curr.readlines():
            splited_curr = line_curr.split()
            curr_lines.append(splited_curr)
            curr_code = splited_curr[0]
            curr_price = splited_curr[2]
            for i in range(len(deals_lines)):
                if curr_code == splited_deals[2]:
                    val = float(curr_price) * float(splited_deals[3])
                    overall = overall + val

        for i in range(len(curr_lines)):
            for j in range(len(deals_lines)):
                if curr_lines[i][0] == deals_lines[j][2]:
                    exit_list.append(curr_lines[i][1])

exit_list.append(float(curr_lines[i][2])*float(deals_lines[j][3])/float(overa
ll))
        print(*exit_list)

        db_file_curr.close()
        db_file_deals.close()

    def maxdeal(self):
```

```
        db_file_curr = open('database_curr.txt', 'r')
        db_file_deals = open('database_deals.txt', 'r')
        exit_list = []
        deals_lines = []
        curr_lines = []

        for line_deals in db_file_deals.readlines():
            splited_deals = line_deals.split()
            deals_lines.append(splited_deals)

        for line_curr in db_file_curr.readlines():
            splited_curr = line_curr.split()
            curr_lines.append(splited_curr)

        for i in range(len(curr_lines)):
            for j in range(len(deals_lines)):
                if curr_lines[i][0] == deals_lines[j][2]:

exit_list.append(float(curr_lines[i][2])*float(deals_lines[j][3]))
        print('Размер максимальной сделки:', max(exit_list), 'рублей')


if __name__ == '__main__':

    app = QApplication(sys.argv)
    ex = database_ui()
    ex.show()

    sys.exit(app.exec_())
```