**Оглавление**

[Аннотация 4](#_Toc26196310)

[Введение 5](#_Toc26196311)

[1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ 6](#_Toc26196312)

[1.1 Основное функциональное назначение программного средства 6](#_Toc26196313)

[1.2 Полное наименование программного средства 6](#_Toc26196314)

[1.3 Условное обозначение программного средства 6](#_Toc26196315)

[1.4 Разработчики программного средства 6](#_Toc26196316)

[2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 7](#_Toc26196317)

[2.1 Основание для разработки 7](#_Toc26196318)

[2.2 Назначение разработки 7](#_Toc26196319)

[2.3 Требование к программному средству 7](#_Toc26196320)

[2.4 Требования к программной документации 8](#_Toc26196321)

[2.5 Требования к эргономике и технической эстетике 8](#_Toc26196322)

[2.6 Стадии и этапы разработки 9](#_Toc26196323)

[2.7 Порядок контроля и приемки 10](#_Toc26196324)

[3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ 11](#_Toc26196325)

[3.1 Декомпозиция поставленной задачи 11](#_Toc26196326)

[3.2 Общая архитектура программного средства 12](#_Toc26196327)

[3.3 Разработка алгоритма решения задачи 15](#_Toc26196328)

[3.4 Реализация функционального назначения программного средства 16](#_Toc26196329)

[3.5 Структурная организация данных 16](#_Toc26196330)

[3.6 Разработка интерфейса ПС 25](#_Toc26196331)

[3.7 Описание структуры выходной информации 31](#_Toc26196332)

[4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 32](#_Toc26196333)

[4.1 Назначение программного средства 32](#_Toc26196334)

[4.2 Условия выполнения программного средства 32](#_Toc26196335)

[4.3 Эксплуатация программного средства 32](#_Toc26196336)

[4.4 Сообщения пользователю 38](#_Toc26196337)

[5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА 40](#_Toc26196338)

[5.1 Обоснование необходимого количества тестов 40](#_Toc26196339)

[5.2 Описание тестовых пакетов 41](#_Toc26196340)

[6. ЗАКЛЮЧЕНИЕ 48](#_Toc26196341)

[6.1 Оценка качества программного средства с помощью метрик 48](#_Toc26196342)

[7. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 51](#_Toc26196343)

[Приложение А листинг программного модуля 52](#_Toc26196344)

[Приложение B диск с исполняемым кодом программного модуля 88](#_Toc26196345)

**Аннотация**

На курсовой проект студента Матинина Н.Н. на тему: Разработка интегрированного программного модуля «Инвестиционная компания». Курсовой проект представлен пояснительной запиской на 89 листов машинописного текста. Состоит из введения, 5 глав, заключения и списка литературных источников.

В главе 1 содержится общие сведения о программном средстве Инвестиционная компания. В главе 2 приведено техническое задание на проектирование БД согласно ГОСТ 19.201. Глава 3 содержит пояснительную записку к программному продукту согласно ГОСТ 19.404-79. В главе 4 содержится руководство пользователя согласно ГОСТ 19.504-79. Глава 5 содержит обоснование количества необходимых тестов и тестовые пакеты для оценки работоспособности программного продукта. В главе 6 приведены результаты ручного и автоматического тестирования программного продукта и выполнена оценка качества с помощью метрик.

В работе содержится общее описание программного средства, предназначенного для объекта проектирования – интегрированного программного модуля «Инвестиционная компания», изложены требования к программному средству, описаны его архитектура, реализация функций, интерфейс, приведено руководство пользователя, обоснованы тестовые пакеты и выполнена оценка качества программного продукта с помощью метрик.

**Введение**

В курсовом проекте была поставлена задача разработки интегрированного программного модуля «Инвестиционная компания», который позволяет автоматизировать учет продолжительности и рисков инвестиционных проектов.

Для достижения поставленной цели был составлен основной алгоритм работы программы, а также разработаны алгоритмы вспомогательных процедур.

Для реализации алгоритмов был выбран язык программирования Java. Выбор обусловлен тем, что возможностей этого языка достаточно для достижения поставленной цели моей курсового проекта.

При разработке интерфейса программы, я учитывал наиболее нужные и важные функциональные возможности, которые должна выполнять данная программа, а также тип пользователей, которые с ней будут работать.

**1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ**

**1.1 Основное функциональное назначение программного средства**

Программа предназначена для автоматизации учета продолжительности и рисков инвестиционных проектов.. Программа не требует владения современными информационными технологиями для эффективного использования данного программного средства, т.е. подходит для любого пользователя.

**1.2 Полное наименование программного средства**

Разработка интегрированного программного модуля «Инвестиционная компания».

**1.3 Условное обозначение программного средства**

«Aurora Growth1.01»

**1.4 Разработчики программного средства**

Программное средство разработал студент Зеленодольского Механического колледжа группы 215 Матинин Н.Н.

**2. ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**2.1 Основание для разработки**

Работа ведётся на основании задания на курсовой проект по специальности 09.02.07 — «Информационные системы и программирование».

**2.2 Назначение разработки**

Программа «Aurora Growth1.01» кросс-платформенная. Программа предназначена для автоматизации деятельности администратора по учету продолжительности и рисков инвестиционных проектов. Программа позволяет вводить, изменять и удалять данные инструментов компании, просматривать рыночные данные.

**2.3 Требование к программному средству**

«Aurora Growth1.01» устанавливается на персональный компьютер, имеющий представленные минимальные системные требования:

Оперативная память: 4 Гб;

Процессор: Intel Core i / Phenom II X4 / AMD Athlon64 X2 4600 + 2.4 ГГц;

Видео карта: nVidia GeForce 8600 / ATI Radeon HD 2400;

Звуковая карта: Совместимая с DirectX;

Программный продукт «Aurora Growth1.01» является кроссплатформенным продуктом, что позволяет не ограничиваться в работе с определенной версией ОС.

«Aurora Growth1.01» написана на объектно-ориентированном языке программировании Java в интегрированной среде разработки IntelliJ IDEA. В качестве СУБД используется PostgresQL. Программа не может подлежать сторонней оптимизации и редактированию программного кода.

**2.4 Требования к программной документации**

Состав программной документации должен включать в себя:

1. техническое задание;

2. сроки выполнения работ;

3. программу и методику испытаний;

4. эксплуатационные инструкции пользователю;

**2.5 Требования к эргономике и технической эстетике**

Система обеспечивает удобный для пользователей системный интерфейс, отвечающий следующим требованиям:

- единый стиль оформления для пользовательских интерфейсов;

- удобная, интуитивно понятная навигация в интерфейсе пользователя;

- взаимодействие пользователя с системой осуществляется на двух языках по выбору (русский и английский), для наибольшего удобства

Пользовательские интерфейсы системы спроектированы и разработаны с применением единых принципов графического представления информации и организации доступа к функциональным возможностям и сервисам. Разработан графический дизайн пользовательских интерфейсов, цветовые, шрифтовые и композиционные решения для отображения текстов, изображений, таблиц, гиперссылок, управляющих и навигационных элементов (меню, кнопок, форм и т.п.).

Экранные формы спроектированы с учетом требований унификации:

- все экранные формы пользовательского интерфейса выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;

- для обозначения сходных операций использованы сходные графические значки, кнопки и другие управляющие (навигационные) элементы. Термины, используемые для обозначения типовых операций (добавление информационной сущности, редактирование поля данных), а также последовательности действий пользователя при их выполнении унифицированы;

- внешнее поведение сходных элементов интерфейса (реакция на наведение указателя «мыши», переключение фокуса, нажатие кнопки) реализованы одинаково для однотипных элементов.

**2.6 Стадии и этапы разработки**

Разработка должна быть проведена в три стадии:

1. разработка технического задания;

2. рабочее проектирование;

3. внедрение.

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;

2. разработка программной документации;

3. испытания программы.

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;

2. определение и уточнение требований к техническим средствам;

3. определение требований к программе;

4. определение стадий, этапов и сроков разработки программы и документации на нее;

5. выбор языков программирования;

6. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с пунктом Предварительный состав программной документации настоящего технического задания.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение порядка и методики испытаний;

2. проведение испытаний;

3. корректировка программы и программной документации по результатам испытаний.

**2.7 Порядок контроля и приемки**

Сдача-приёмка работ производится поэтапно, в соответствии с рабочей программой и календарным планом. Сдача-приемка осуществляется комиссией, в состав которой входят представители ГАПОУ и работодатель. По результатам приемки подписывается акт приемочной комиссии.

В указанном подразделе, согласно таблице «Сроки выполнения работ» технического задания, будет описаны перечень участвующих организаций, место и сроки проведения работ, согласно п. 2.8 ГОСТ 34.602-89.

Порядок согласования и утверждения приемочной документации должен регламентироваться организационно-распорядительной документацией организации, принимающей участие в создании программного продукта. Согласно разделу «Приемка результатов разработки» ГОСТ 15.001-88 для согласования и утверждения приемочной документации создается приемочная комиссия (приказом).

Статус приемочной комиссии определяется Заказчиком до проведения испытаний.

**3. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ**

**3.1 Декомпозиция поставленной задачи**

Для достижения цели работы была поставлена следующая задача:

Создать модуль “CodeDatabase” отвечающий за сохранение БД. Результат работы модуля - сохранение БД.

Поставленные задачи наглядно отображаются на диаграмме деятельности.

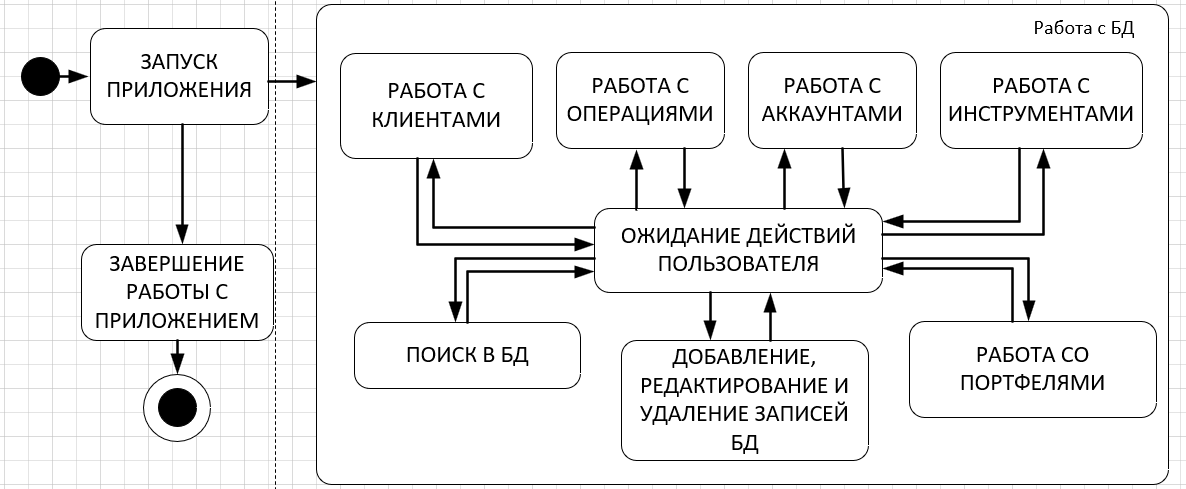


Рис.1 Декомпозиция поставленной задачи

**3.2 Общая архитектура программного средства**

Объектом моделирования является Инвестиционная компания.

Рассматриваемые процессы: Добавление и удаление информации о списке ролей, клиентов, пользователей, счетов, инструментов, портфолей, позиций портфолей, транзакций, коммссий, рыночных данных, инвестиционных стратегий, стратегий портфелей

Объекты моделирования представлены на диаграмме классов, рис.2.

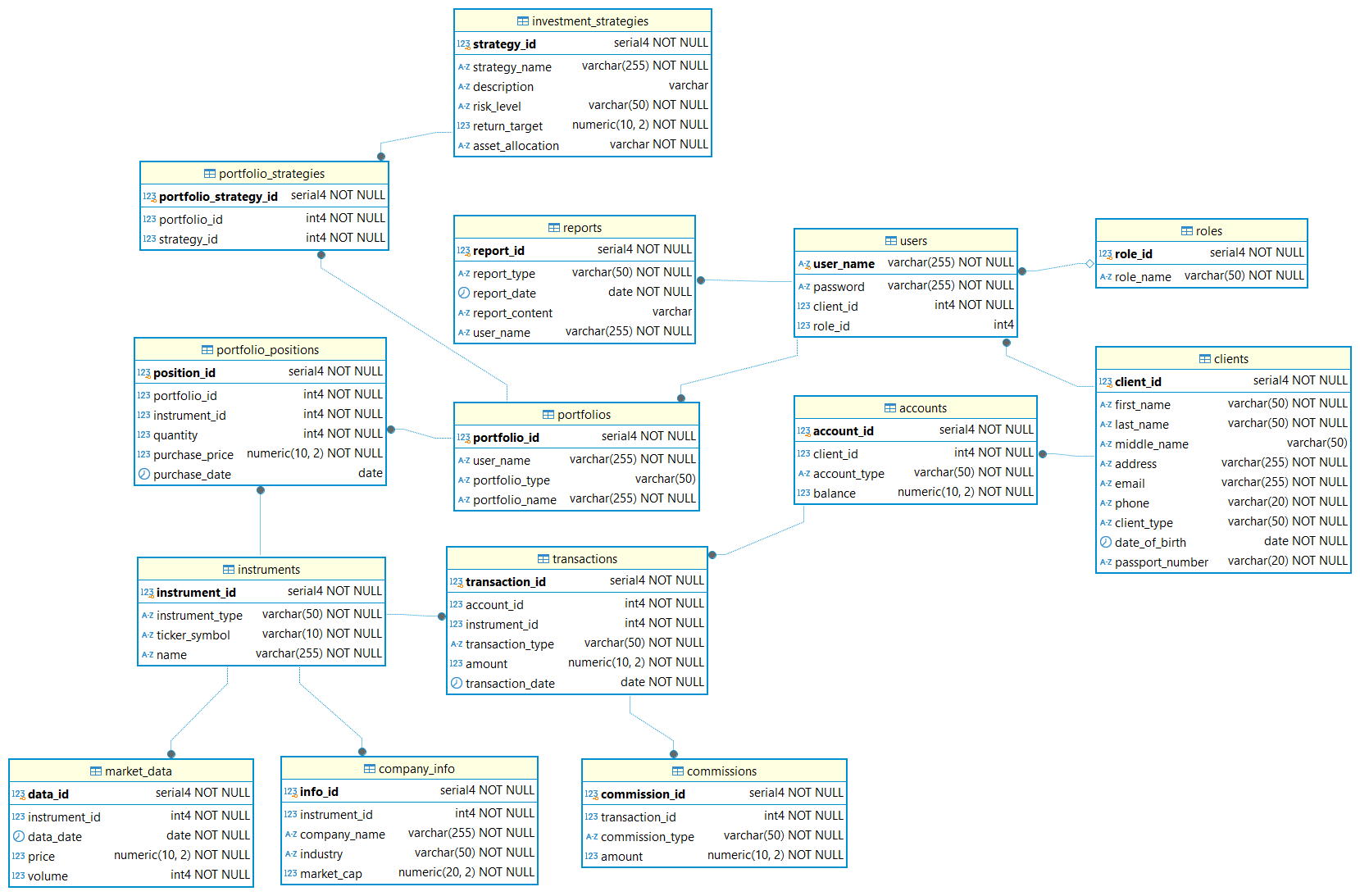


Рис.2 Диаграмма классов

По полученной декомпозиции задач была спроектирована следующая архитектура программного средства.

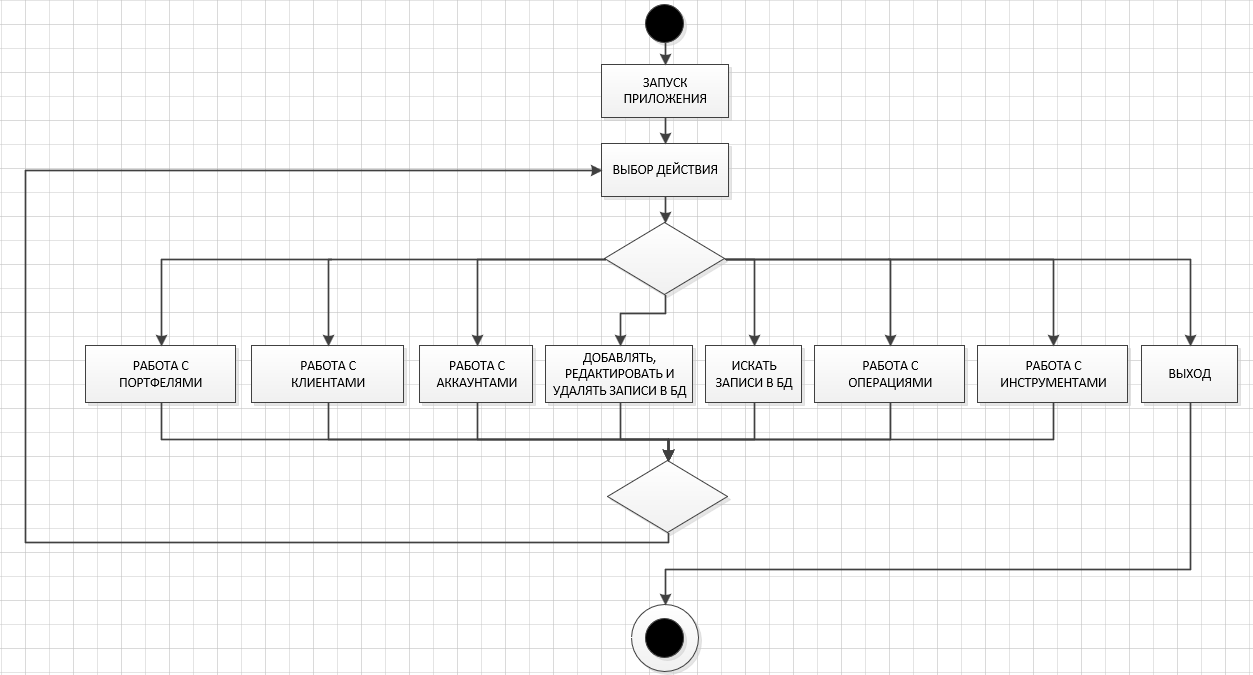


Рис.3 Схема событийно-ориентированной архитектуры

В «Aurora Growth1.01» используется событийно-ориентированная архитектура.(Рис.3)

Роль агента (источник событий) в программе выполняют: пользователи программы, за роль стоков (потребители событий) отвечают таблицы, входящие в базу данных. Например, когда пользователь выбирает определенное действие: добавить, удалить, редактировать, вывести в Pdf и т.д. система осуществит выбранные действия, и база отреагирует соответствующим образом: запись добавлена, удалена, отредактирована, либо отчет выведен в Pdf. Системная архитектура пользователя рассматривает это изменение состояния как событие, создаваемое, публикуемое, определяемое и потребляемое различными приложениями в составе архитектуры.



Рис.4 Диаграмма вариантов использования

**3.3 Разработка алгоритма решения задачи**

**Общий алгоритм**

****Рис.5 Алгоритм программы

**3.4 Реализация функционального назначения программного средства**

Программа «Aurora Growth1.01» предназначена для управления инвестиционными данными и генерации отчетов.

Данные вводятся администратором в соответствующие поля ввода, снабженные всплывающими подсказками.

Выходными данными являются: отображение списка рыночных данных по компаниям в виде строчного формата.

Данные вводятся пользователем в соответствующие поля вывода, снабженные всплывающими подсказками.

Выходные данные редактировать вручную невозможно, т.к. они служат в качестве отчетной информации для пользователя.

**3.5 Структурная организация данных**

Для создания БД необходимо определиться с данными, которые необходимы для полноценного функционирования системы. Все эти данные указаны в реляционной модели «БД Инвестиционная компания» представленная на рисунке. Любая реляционная база данных и называется реляционной, что характеризуется отношениями (relation) между таблицами. На рисунке изображены таблицы моей базы данных. Реляционная модель автоматизированной системы соответствует всем 12 правилам Кодда. (рис.6)

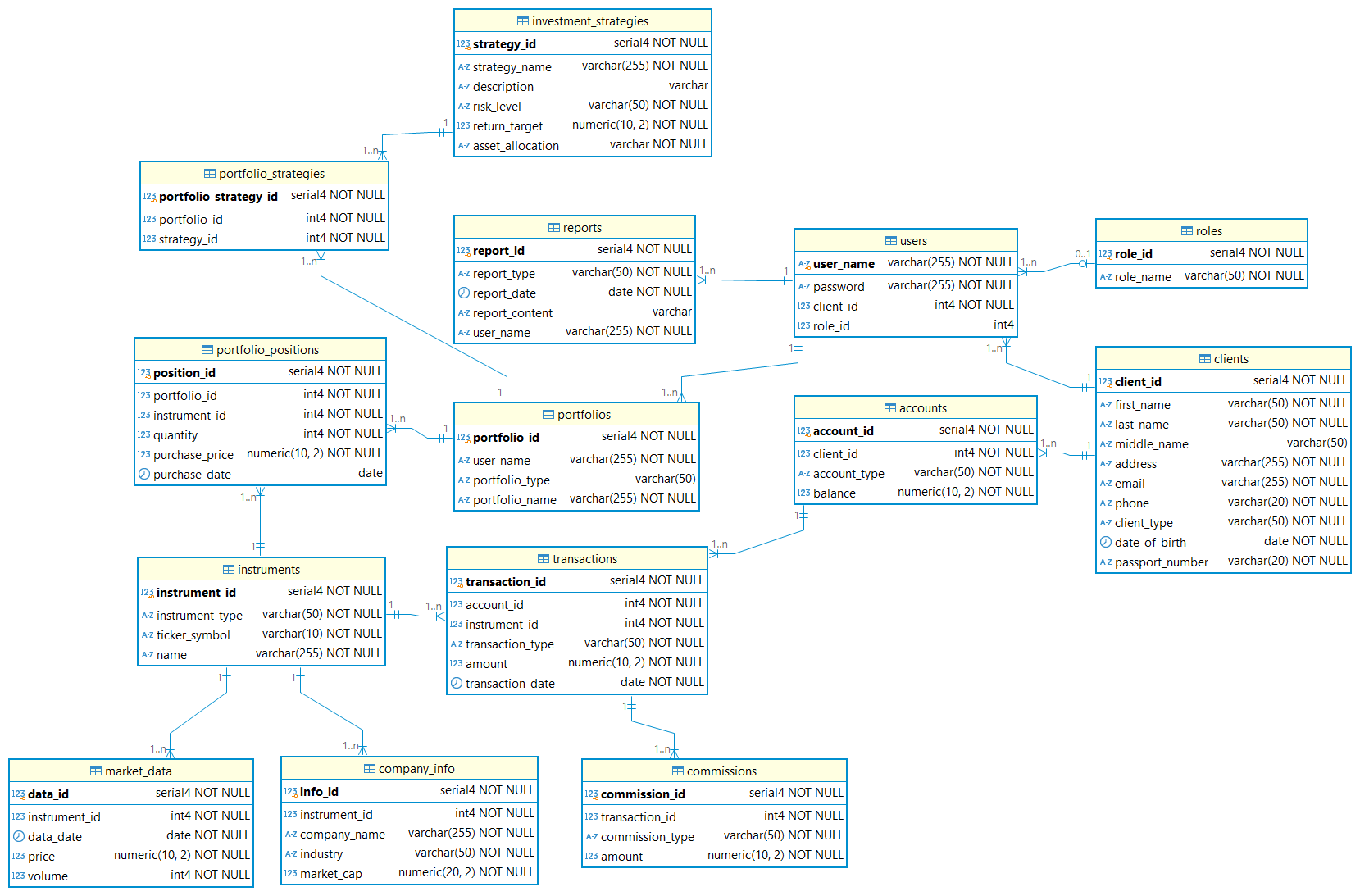


Рис.6 Реляционная модель

Первичный ключ в базе уникален, используется для организации отношений между таблицами, который не может иметь пустых и повторяющихся значений. Первичными ключами в базе являются поля: role\_Id (таблица Роли), client\_Id (таблица Клиенты), user\_Name (таблица Пользователи), account\_Id (таблица Счета), instrument\_Id (таблица Инструменты), info\_Id (таблица Информация о компании), portfolio\_Id (таблица Портфели), position\_Id (таблица Позиции портфелей), transaction\_Id (таблица Операции), commission\_Id (таблица Комиссии), data\_Id (таблица Рыночные данные), strategy\_Id (таблица Инвестиционные стратегии), portfolio\_Strategy\_Id (таблица Портфели стратегий), report\_Id (таблица Отчеты). Остальные ID-ключи являются внешними ключами.

Для организации более эффективной обработки данных применяется нормализация. Таблицы моей БД находятся в 3НФ:

* БД находится в форме -1НФ потому, что
  + Таблица не имеет повторяющихся записей;
  + Каждый атрибут отношения хранит одно-единственное значение и не является списком, ни множеством значений;
  + Таблица не имеет повторяющихся групп полей.
    - Вторая нормальная форма(2НФ):
      * Устранены атрибуты, зависящие только от части уникального (первичного) идентификатора, т.е. ID.
        + Третья нормальная форма(3НФ):

Отсутствуют атрибуты, зависящие от атрибутов, не входящих в уникальный (первичный) ключ.

На основе реляционной модели базы данных мною в DBeaver была построена следующая база данных. (Рис.7) В этой базе данных первичным ключом таблицы users является поле user\_Name. Связь между таблицами clients и users по полю client\_Id является один-ко-многим (1:М). Связь между таблицами roles и users по полю role\_Id также один-ко-многим (1:М). Связь между clients и accounts по полю client\_Id один-ко-многим (1:М). Связь между accounts и transactions по полю account\_Id один-ко-многим (1:М). Связь между instruments и transactions по полю instrument\_Id один-ко-многим (1:М). Связь между instruments и company\_Info по полю instrument\_Id один-ко-многим (1:М). Связь между users и portfolios по полю user\_Name один-ко-многим (1:М). Связь между portfolios и portfolio\_Positions по полю portfolio\_Id один-ко-многим (1:М). Связь между instruments и portfolio\_Positions по полю instrument\_Id один-ко-многим (1:М). Связь между transactions и commissions по полю transaction\_Id один-ко-многим (1:М). Связь между instruments и market\_Data по полю instrument\_Id один-ко-многим (1:М). Связь между portfolios и portfolio\_Strategies по полю portfolio\_Id один-ко-многим (1:М). Связь между investment\_Strategies и portfolio\_Strategies по полю strategy\_Id один-ко-многим (1:М). Связь между users и reports по полю user\_Name один-ко-многим (1:М). Все связи между таблицами базы данных типа один ко многим.

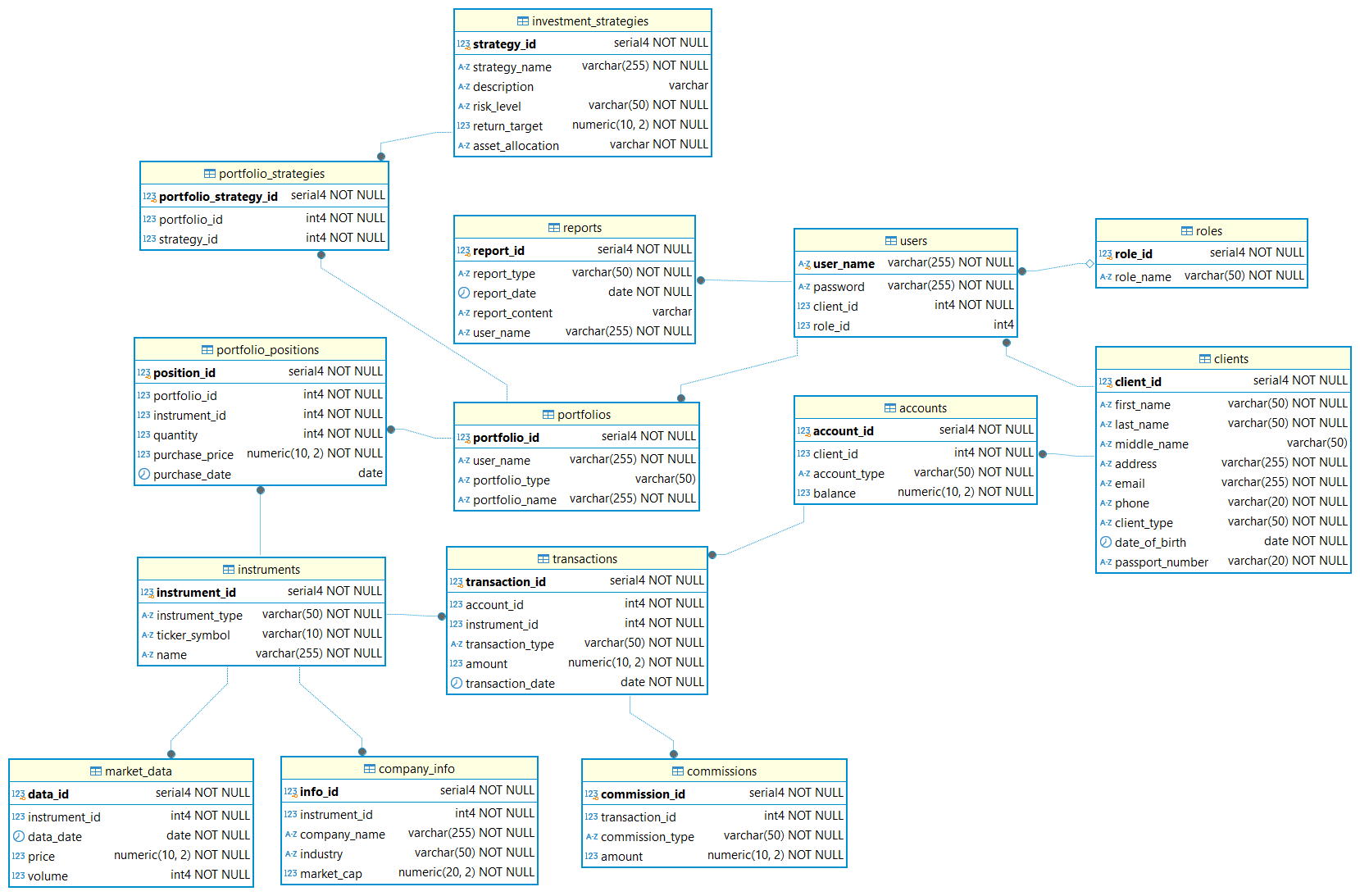


Рис.7 Схема базы данных

Таблица “roles”(Роли)

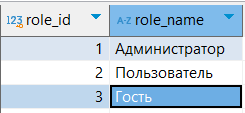


Рис.8 Таблица roles

Таблица “clients”(Клиенты)

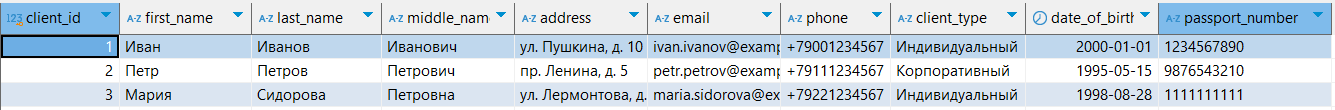


Рис.9 Таблица clients

Таблица “users”(Пользователи)

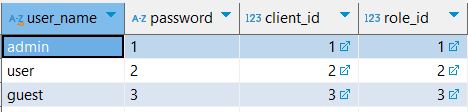


Рис.10 Таблица users

Таблица “accounts”(Аккаунты)

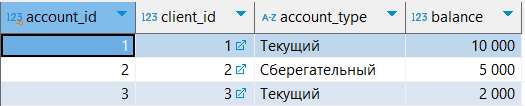


Рис.11 Таблица accounts

Таблица “instruments”(Инструменты)

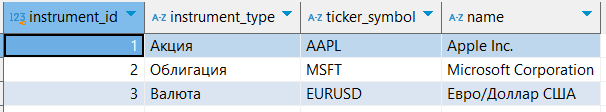


Рис.11 Таблица instruments

Таблица “company\_Info”(Информация о компаниях)

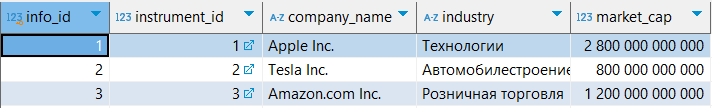


Рис.12 Таблица company\_Info

Таблица “portfolios”(Портфели)

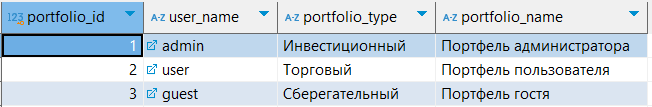


Рис.13 Таблица portfolios

Таблица “portfolio\_Positions ”(Позиции портфелей)

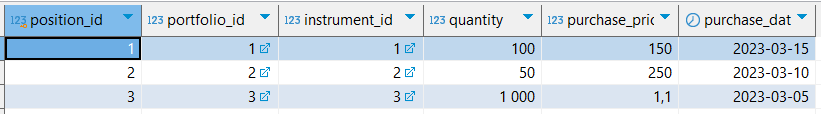


Рис.14 Таблица portfolio\_Positions

Таблица “transactions”(операции)

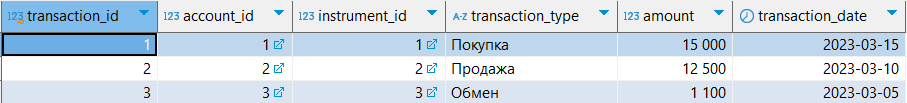


Рис.15 Таблица transactions

Таблица “commissions”(Комиссии)

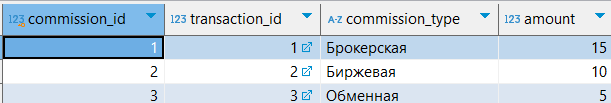


Рис.16 Таблица commissions

Таблица “market\_Data”(Рыночные данные)

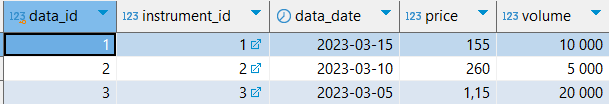


Рис.17 Таблица market\_Data

Таблица “investment\_Strategies”(Инвестиционные стратегии)

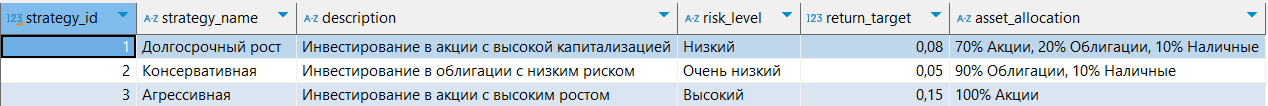


Рис.18 Таблица investment\_Strategies

Таблица “portfolio\_Strategies”(Портфели стратегий)

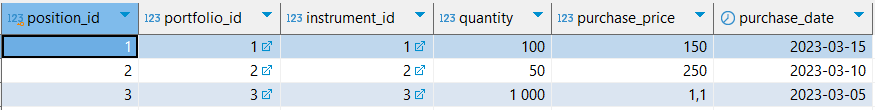


Рис.19 Таблица portfolioStrategies

Таблица “reports”(Отчеты)

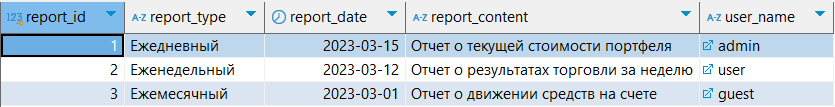


Рис.20 Таблица reports

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Словарь Данных | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| **roles** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **role\_Id** | serial4 | Y | Autoincrement |
|  | **role\_Name** | varchar(50) | Y |  |
|  |  |  |  |  |
| **clients** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **client\_Id** | serial4 | Y | Autoincrement |
|  | **first\_Name** | VARCHAR (50) | Y |  |
|  | **last\_Name** | VARCHAR (50) | Y |  |
|  | **middle\_Name** | VARCHAR (50) |  |  |
|  | **address** | VARCHAR(255) | Y |  |
|  | **email** | VARCHAR(255) | Y |  |
|  | **phone** | VARCHAR(20) | Y |  |
|  | **client\_Type** | VARCHAR (50) | Y |  |
|  | **date\_Of\_Birth** | DATE | Y |  |
|  | **passport\_Number** | VARCHAR(20) | Y |  |
|  |  |  |  |  |
| **users** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **username** | VARCHAR (255) | Y |  |
|  | **password** | VARCHAR(255) | Y |  |
| FK | **client\_Id** | int4 | Y | Внешний ключ на clients |
| FK | **role\_Id** | int4 |  | Внешний ключ на roles |
|  |  |  |  |  |
| **accounts** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **account\_Id** | serial4 | Y | Autoincrement |
| FK | **client\_Id** | int4 | Y | Внешний ключ на clients |
|  | **account\_Type** | VARCHAR(50) | Y |  |
|  | **balance** | DECIMAL(10,2) | Y |  |
|  |  |  |  |  |
| **instruments** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **instrument\_Id** | serial4 | Y | Autoincrement |
|  | **instrument\_Type** | VARCHAR(50) | Y |  |
|  | **ticker\_Symbol** | VARCHAR(10) | Y |  |
|  | **name** | VARCHAR(255) | Y |  |
|  |  |  |  |  |
| **company\_Info** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **info\_Id** | serial4 | Y | Autoincrement |
| FK | **instrument\_Id** | int4 | Y | Внешний ключ на instruments |
|  | **company\_Name** | VARCHAR(255) | Y |  |
|  | **industry** | VARCHAR(50) | Y |  |
|  | **market\_Cap** | DECIMAL(10,2) | Y |  |
|  |  |  |  |  |
| **portfolios** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **portfolio\_Id** | serial4 | Y | Autoincrement |
| FK | **user\_Name** | VARCHAR(255) | Y | Внешний ключ на users |
|  | **portfolio\_Type** | VARCHAR(50) |  |  |
|  | **portfolio\_Name** | VARCHAR(255) | Y |  |
|  |  |  |  |  |
| **portfolio\_Positions** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **position\_Id** | serial4 | Y | Autoincrement |
| FK | **portfolio\_Id** | int4 | Y | Внешний ключ на portfolios |
| FK | **instrument\_Id** | int4 | Y | Внешний ключ на instruments |
|  | **quantity** | int | Y |  |
|  | **purchase\_Price** | DECIMAL(10,2) | Y |  |
|  | **purchase\_Date** | DATE |  |  |
|  |  |  |  |  |
| **transactions** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **transaction\_Id** | serial4 | Y | Autoincrement |
| FK | **account\_Id** | int4 | Y | Внешний ключ на accounts |
| FK | **instrument\_Id** | int4 | Y | Внешний ключ на instruments |
|  | **transaction\_Type** | VARCHAR(50) | Y |  |
|  | **amount** | DECIMAL(10,2) | Y |  |
|  | **transaction\_Date** | DATE | Y |  |
|  |  |  |  |  |
| **commissions** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **commission\_Id** | serial4 | Y | Autoincrement |
| FK | **transaction\_Id** | int4 | Y | Внешний ключ на transactions |
|  | **commission\_Type** | VARCHAR(50) | Y |  |
|  | **amount** | DECIMAL(10,2) | Y |  |
|  |  |  |  |  |
| **market\_Data** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **data\_Id** | serial4 | Y | Autoincrement |
| FK | **instrument\_Id** | int4 | Y | Внешний ключ на instruments |
|  | **data\_Date** | DATE | Y |  |
|  | **price** | DECIMAL(10,2) | Y |  |
|  | **volume** | int | Y |  |
|  |  |  |  |  |
| **investment\_Strategies** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **strategy\_Id** | serial4 | Y | Autoincrement |
|  | **strategy\_Name** | VARCHAR (255) | Y |  |
|  | **description** | VARCHAR |  |  |
|  | **risk\_Level** | VARCHAR(50) | Y |  |
|  | **return\_Target** | DECIMAL(10,2) | Y |  |
|  | **asset\_Allocation** | VARCHAR | Y |  |
|  |  |  |  |  |
| **portfolio\_Strategies** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **employee\_category\_id** | serial4 | Y | Autoincrement |
| FK | **portfolio\_Id** | int4 | Y | Внешний ключ на portfolios |
| FK | **strategy\_Id** | int4 | Y | Внешний ключ на investment\_Strategies |
|  |  |  |  |  |
| **reports** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | **report\_Id** | serial4 | Y | Autoincrement |
|  | **report\_Type** | VARCHAR (50) | Y |  |
|  | **report\_Date** | DATE | Y |  |
|  | **report\_Content** | VARCHAR (50) |  |  |
| FK | **user\_Name** | VARCHAR(255) | Y | Внешний ключ на users |

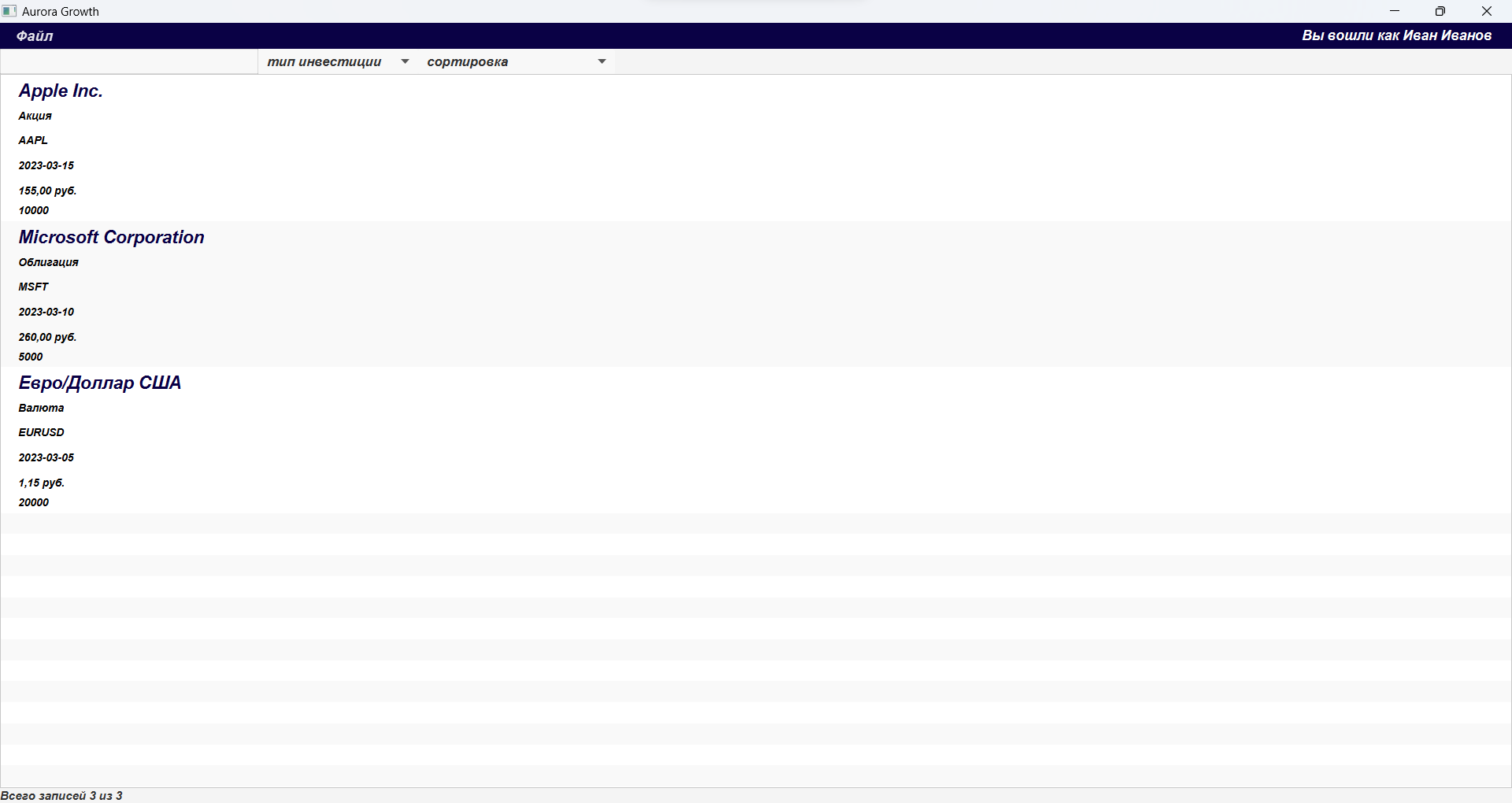
**3.6 Разработка интерфейса ПС**

В приложении предусмотрена авторизация, так как оно многопользовательское. Для запуска приложения достаточно щелкнуть дважды левой кнопкой мыши по ярлыку.

После старта приложения пользователь увидит форму авторизации. Успешная авторизация предоставит доступ к главному окну программы, где будут отображаться все события из базы данных. Неуспешная авторизация вернет пользователя на форму авторизации с соответствующим сообщением об ошибке. «Aurora Growth1.01» обладает удобным интерфейсом, так что даже неопытному в обращение с пользовательскими компьютерными программами пользователю не составит труда быстро привыкнуть к работе с данным программным средством.

На рабочем экране расположены несколько кнопок для комфортной работы с приложением.

**Главная форма**. Основная форма приложения, в которой в виде строк отображаются рыночных данных по компаниям. Компоненты формы: ListView – для отображения рыночных данных по компаниям, Combobox – выпадающие списки, TextBox, Label и MenuBar. (Рис.21)

****Рис.21 Главная

Label

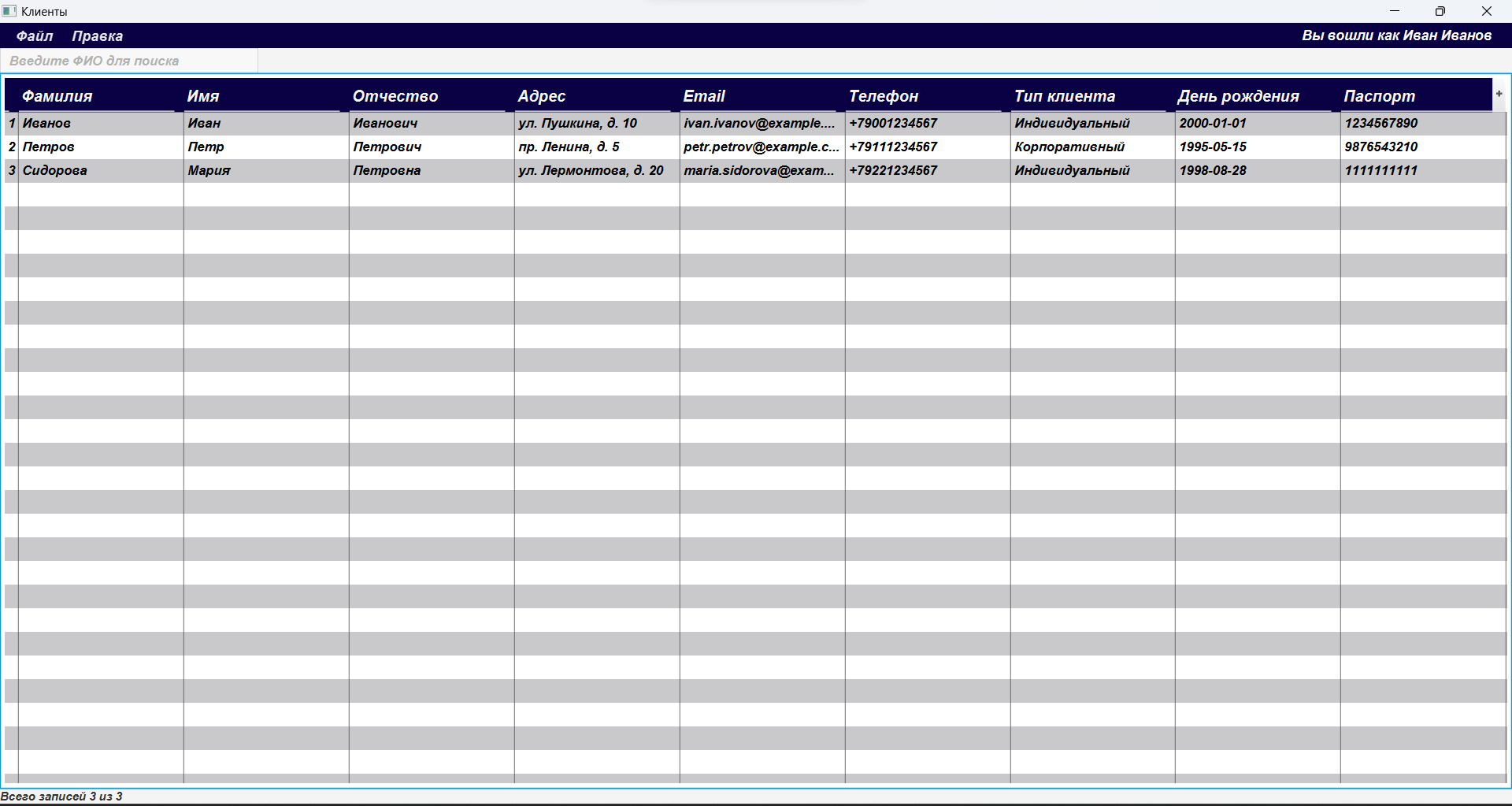
ComboBox

ListView

MenuBar

TextBox

**Страница Клиенты**. Компоненты Страницы: TableView – для отображения списка клиентов, MenuBar для перехода на дополнительные страницы, TextBox и Label. (Рис.22)



TableView

TextBox

MenuBar

Label

Рис.22 Клиенты

**Страница Добавление и изменение**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, DatePicker – выбор даты и Button. (Рис.23)



TextBox

DatePicker

Button

Рис.23 Страница Добавление и изменение

**Страница Аккаунты**. Компоненты Страницы: TableView – для отображения списка аккаунтов, MenuBar для перехода на дополнительные страницы, TextBox и Label. (Рис.24)

TableView

TextBox

MenuBar

Label

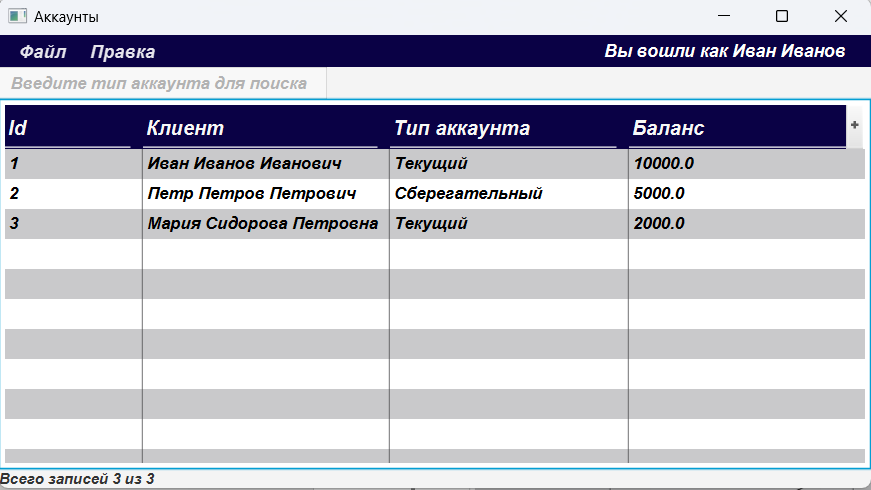
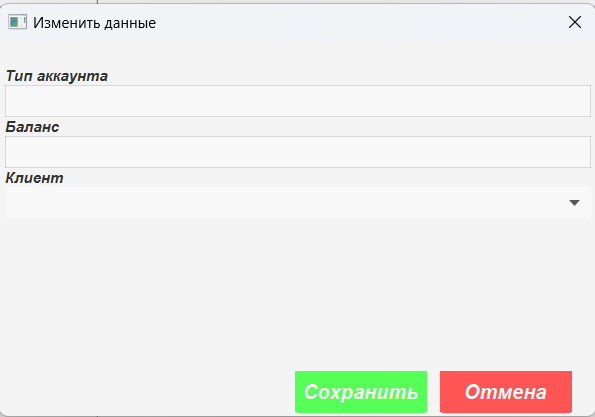


Рис.24 Аккаунты

**Страница Добавление и изменение**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, Combobox – выпадающие списки и Button. (Рис.25)



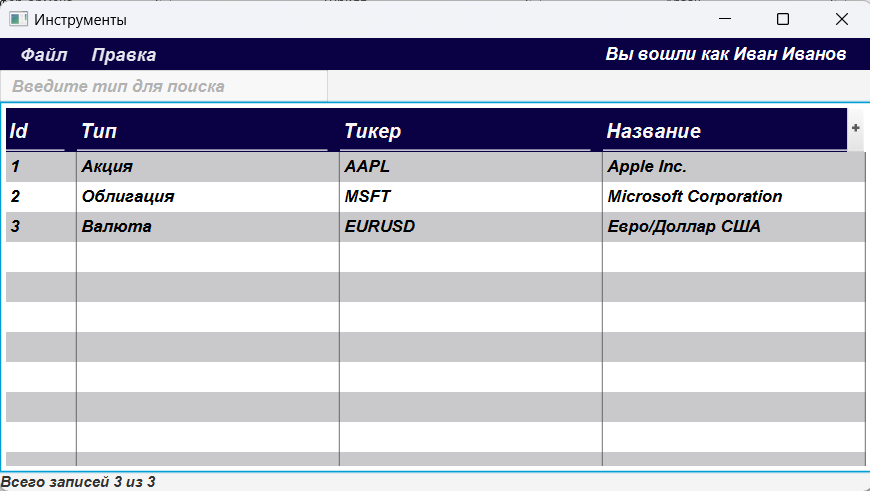
Button

ComboBox

TextBox

Рис.25 Страница Добавление и изменение

**Страница Инструменты**. Компоненты Страницы: TableView – для отображения списка инструментов, MenuBar для перехода на дополнительные страницы, TextBox и Label. (Рис.26)



TableView

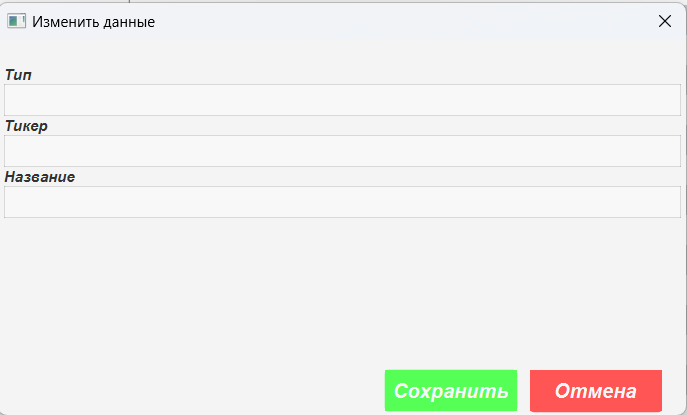
TextBox

MenuBar

Label

Рис.26 Инструменты

**Страница Добавление и изменение**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода и Button. (Рис.27)



TextBox

Button

Рис.27 Страница Добавление и изменение

**Страница Операции**. Компоненты Страницы: TableView – для отображения списка операций, MenuBar для перехода на дополнительные страницы, TextBox и Label. (Рис.28)



MenuBar

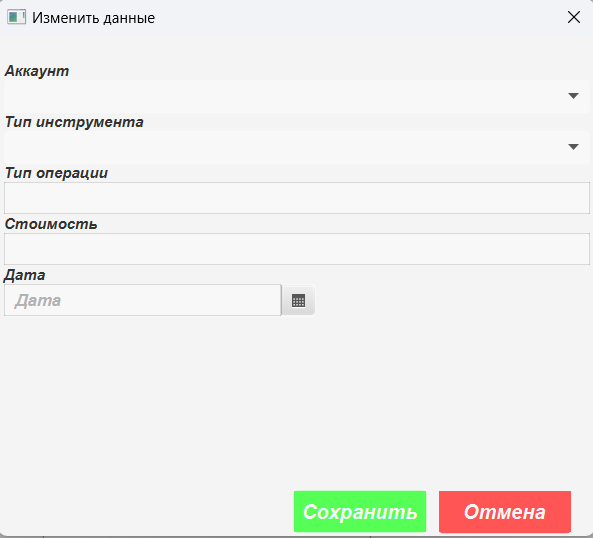
TableView

TextBox

Label

Рис.28 Операции

**Страница Добавление и изменение**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода DatePicker – выбор даты, Combobox – выпадающие списки и Button. (Рис.29)



ComboBox

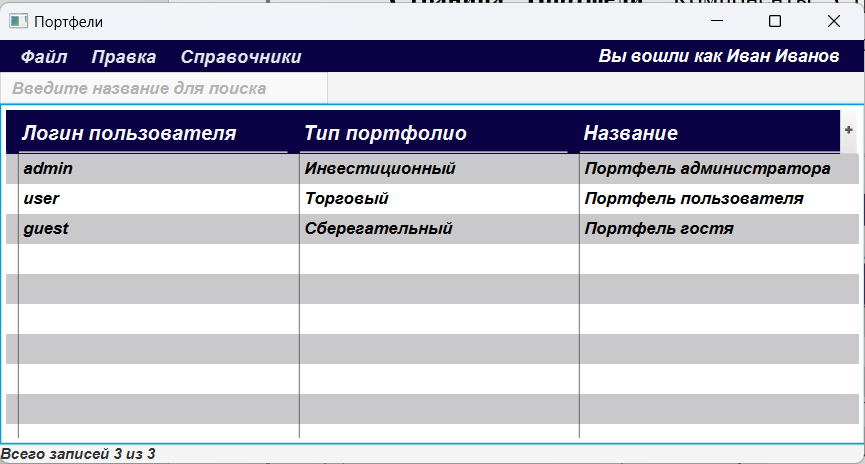
DatePicker

Button

TextBox

Рис.29 Страница Добавление и изменение

**Страница Портфели**. Компоненты Страницы: TableView – для отображения списка портфелей, MenuBar для перехода на дополнительные страницы, TextBox и Label. (Рис.30)



Label

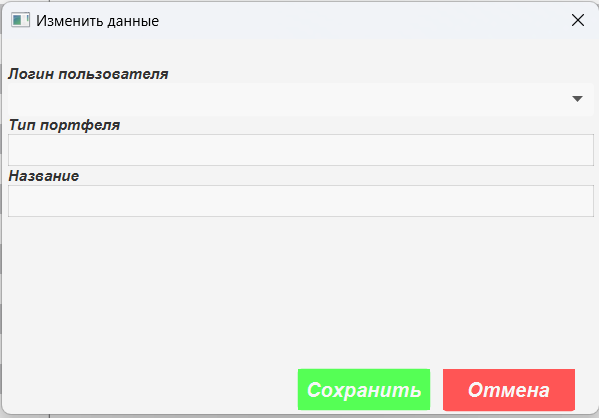
TableView

TextBox

MenuBar

Рис.30 Аккаунты

**Страница Добавление и изменение**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, Combobox – выпадающие списки и Button. (Рис.31)



Button

ComboBox

TextBox

Рис.31 Страница Добавление и изменение

**3.7 Описание структуры выходной информации**

Выходными данными являются списки рыночных данных по компаниям отфильтрованные по разным критериям.

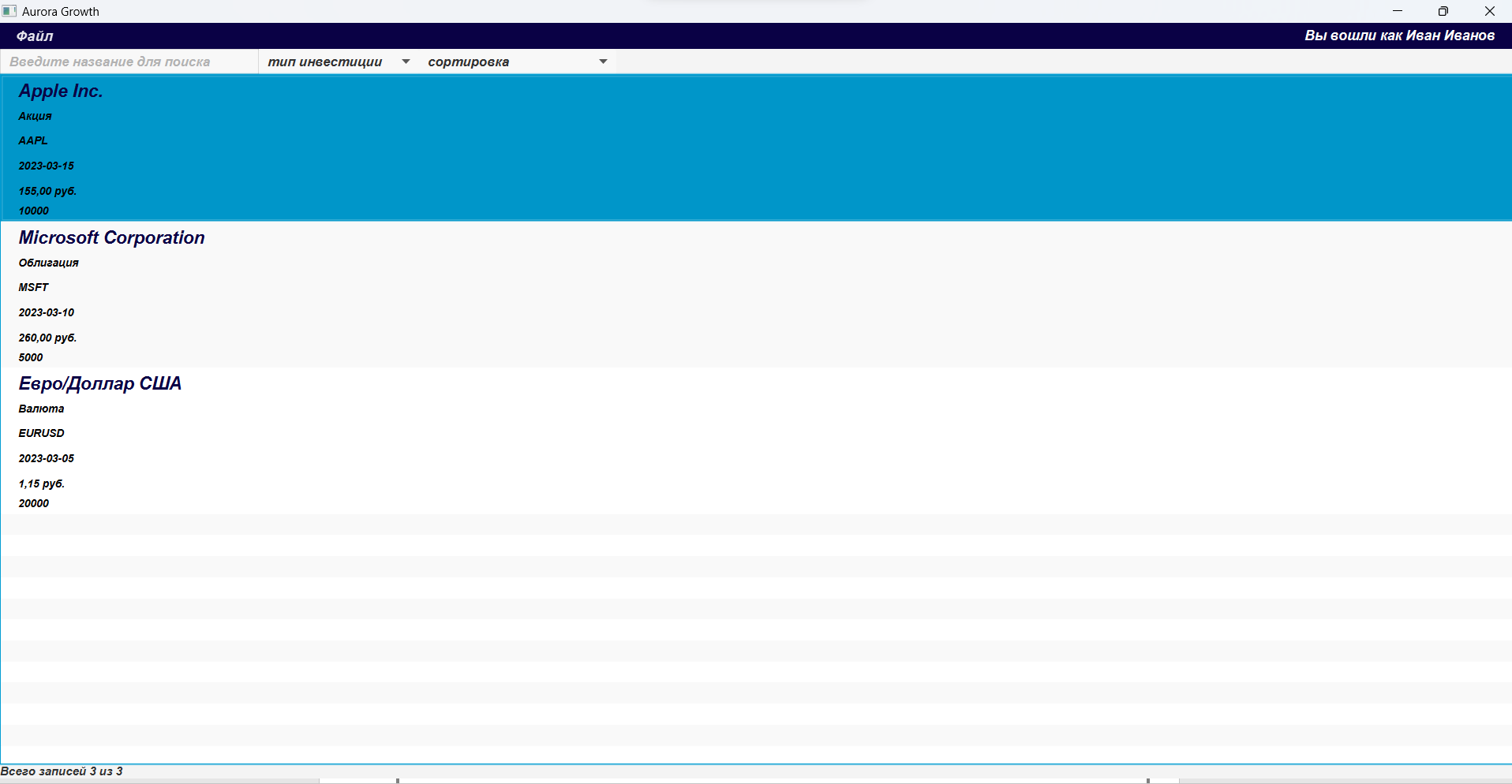


Рис.32 Списки рыночных данных по компаниям

**4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**4.1 Назначение программного средства**

Назначением программного средства является автоматизация учета мероприятий.

**4.2 Условия выполнения программного средства**

Условия, при соблюдении которых возможно выполнение работы с программным продуктом:

-Компьютер с установленной на нём «Aurora Growth1.01».

Подготовительные действия:

-На иконке «Aurora Growth1.01» рабочего стола произвести двойной щелчок левой кнопкой мыши.

**4.3 Эксплуатация программного средства**

Этот раздел содержит обзор приложений «Aurora Growth1.01» и ее основных параметров.

***Активация продукта***

Активировать копию программы Aurora Growth1.01 можно сразу после её установки, когда появится окно подтверждения подлинности Вашей версии. Ключ активации расположен внутри упаковки программного продукта. Для успешного выполнения активации ключ активации необходимо вводить именно в том виде, в котором он предоставлен.

Вводные сведения об интерфейсе пользователя

После старта приложения откроется главное окно приложения – Список рыночных данных по компаниям в строчном формате. На форме есть кнопка для перехода на другие страницы. Есть поля для поиска компаний по названию и выпадающие списки для фильтрации. (Рис.33)

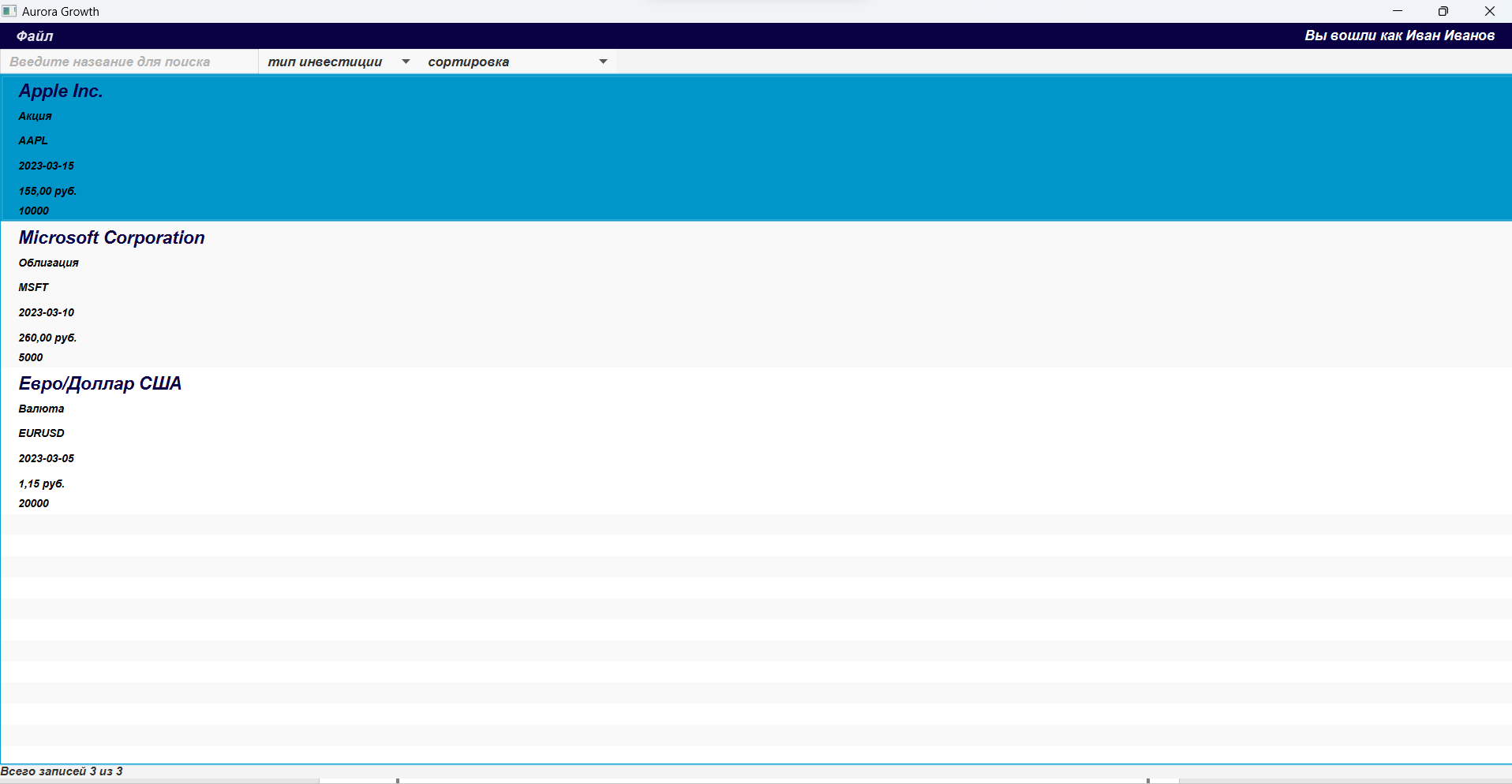


Рис.33 Окно рыночные данные по компаниям

Кликнув по кнопке «Выход» программа потребует подтверждение. (Рис.34)

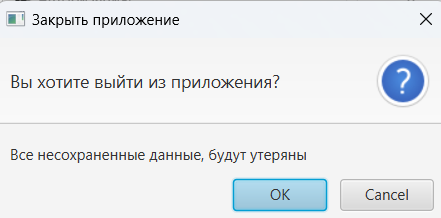


Рис.34 Сообщение программы

Ниже описаны пункты меню управления и информации.

«Страница Клиенты»: эта форма предназначена для отображения списка клиентов. (Рис. 35)

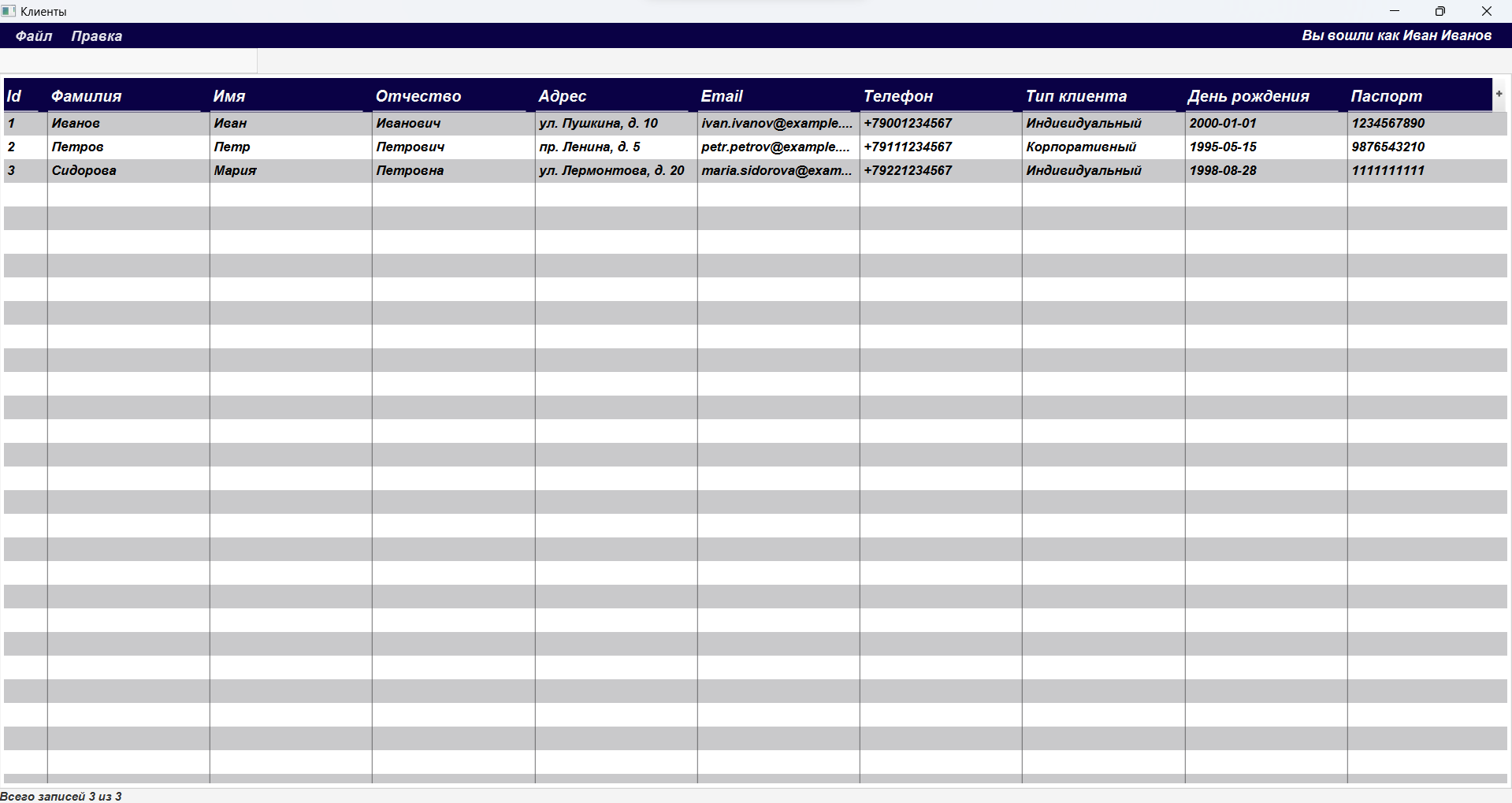


Рис.35 Страница Клиенты

«Аккаунты»: эта страница предназначена для ввода, удаления и редактирования данных в таблице Аккаунты. (Рис.36)

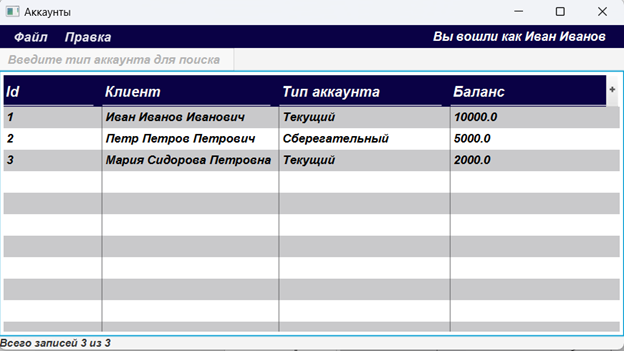


Рис.36 Страница Аккаунты

«Инструменты»: эта страница предназначена для ввода, удаления и редактирования данных в таблице Инструменты. (Рис.37)

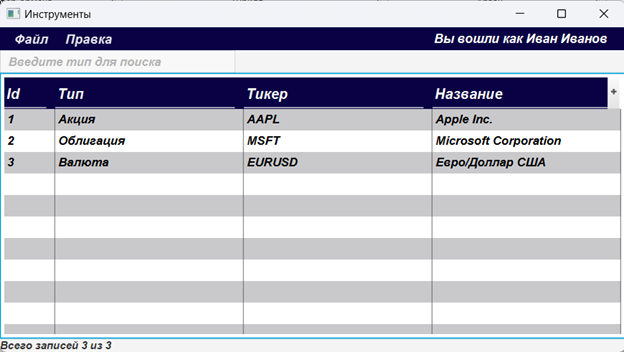


Рис.37 Страница Инструменты

«Операции»: эта страница предназначена для ввода, удаления и редактирования данных в таблице Операции. (Рис.38)



Рис.38 Страница Операции

«Портфели»: эта страница предназначена для ввода, удаления и редактирования данных в таблице Портфели. (Рис.39)

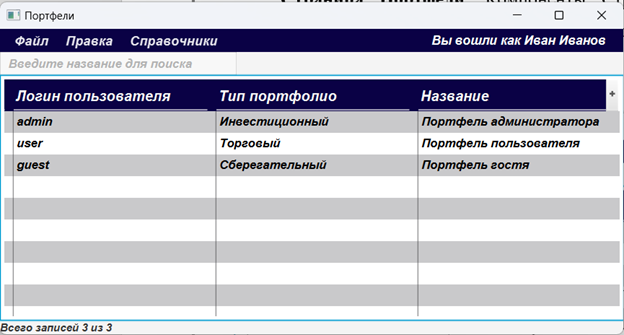


Рис.39 Страница Портфели

Работа с Aurora Growth1.01

При успешной авторизации в программу откроется главное окно. Для работы с данными используйте формы «Клиенты», «Операции», «Портфели», «Аккаунты» и «Инструменты». В каждой форме вы можете:

•Добавить новую запись: Используйте кнопку «Добавить».

•Редактировать существующую запись: Используйте кнопку «Изменить».

•Удалить запись: Используйте кнопку «Удалить». Будьте осторожны, эта операция необратима.

Кнопки управления («Добавить», «Удалить», «Изменить») расположены на каждой форме.

Чтобы закрыть программу, закройте окно приложения.

Действия этих кнопок приведены в таблице 1:

Таблица-1 Действие кнопок

|  |  |
| --- | --- |
| Добавить запись | Прежде чем добавить запись в таблицу необходимо нажать на «Правка» → «Добавить». Поля формы будут очищены, пользователю требуется ввести необходимые данные. Если пользователь нажмет кнопку «Сохранить», то введенная информация добавится в БД. Если нажать кнопку «Отменить», то введенная информация пользователем удалится. |
| Удалить запись | Кнопка «Удалить» позволяет удалить конкретную запись. |
| Изменить данные | Кнопка «Изменить» открывает форму в режиме редактирования информации |
|  | «Поиск» позволяет найти нужную информацию. |

**4.4 Сообщения пользователю**

Для уверенной и комфортной работы пользователю важно понимать поведение программы, а программе, в свою очередь, поведение пользователя. Для информирования и уточнения намерений пользователей в программах используются сообщения.

При попытке удалить запись, пользователю выводится сообщение с запросом на подтверждение или отмену действия. (Рис.40)

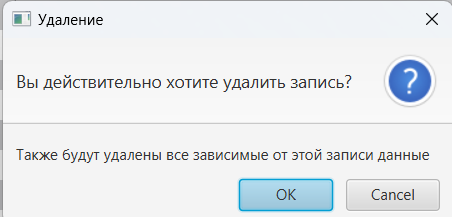


Рис.40 Сообщение Подтверждение удаления

После добавления новой записи, пользователю выводится сообщение. (Рис.41)

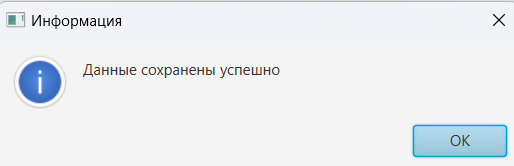


Рис.41 Сообщение Запись добавлена

После обновления записи, пользователю выводится сообщение. (Рис.42)

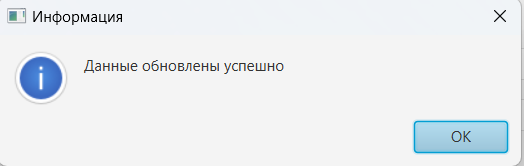


Рис.42 Сообщение Запись обновлена

Если пользователь пытается ввести текстовые данные в числовое поле, программа выведет сообщение об ошибке. (Рис.43)

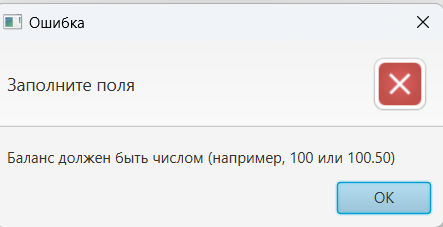


Рис.43 Сообщение Ошибка валидации

При попытке неправильно ввода email, программа выводит пользователю сообщение о невозможности действия.. (Рис.43)

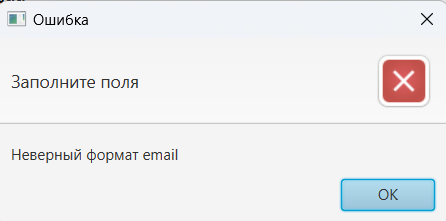


Рис.43 Сообщение Ошибка валидации email

**5. РАЗРАБОТКА ТЕСТОВОГО НАБОРА**

**5.1 Обоснование необходимого количества тестов**

Для разработанного алгоритма был построен граф.

****

Рис.27 Потоковый граф

Необходимое количество тестов определяется по формуле V(G) = E-N+2, где E – это число ребер, а N–это число вершин данного графа. Согласно представленному графу E=35, N=25. Далее по формуле рассчитаем необходимое количество тестовых сценариев (тест-кейсов).

V(G)=35-25+2=12.

Для данного программного средства необходимо разработать не менее 12 тестовых сценариев.

Чтобы определить количество тестовых прогонов для программного продукта AuroraGrowth1.01 необходимо выделить требования. Из требований к приложению выделим поддержку 2-х операционных систем с одним основным языком локализации и выполнения 6-ти основных функций. Кроме того, приложение позволяет выполнять 5 функций по взаимодействию с системой (запуск приложения, выход из приложения, сохранение результатов в файл, работа с буфером и т.п.).

Таким образом, полное покрытие требований задаёт набор из 2\*1\*(6+5) =22 тестовых прогонов.

**5.2 Описание тестовых пакетов**

Таблица 2 – Тестирование корректного добавления информации в Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Клиенты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  first\_Name = "Иван"  last\_Name = "Иванов"  middle\_Name = "Иванович"  address = "ул. Ленина, 1"  email = "ivan.ivanov@gmail.com"  phone = "+79001234567"  client\_Type = "Индивидуальный"  date\_Of\_Birth = "1980-05-10"  passport\_Number = "1234567890"  Статус = Завершилось  Цвет = Красный | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Клиенты открыта | | Отображается форма Клиенты |  |

Таблица 3 – Тестирование некорректного добавления информации в таблицу Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Клиенты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  first\_Name = "Иван"  last\_Name = "Иванов"  middle\_Name = "Иванович"  address = "ул. Ленина, 1"  email =  phone = "+79001234567"  client\_Type = "Индивидуальный"  date\_Of\_Birth = "1980-05-10"  passport\_Number = "1234567890"  Статус =  Цвет = Красный | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, так как поле email пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Клиенты открыта | | Отображается форма Клиенты |  |

Таблица 4 – Тестирование корректного редактирования информации в таблице Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Клиенты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  first\_Name = "Петр"  last\_Name = "Петров"  middle\_Name = "Иванович"  address = "ул. Ленина, 1"  email = "petr.petrov@gmail.com"  phone = "+79005434567"  client\_Type = "Индивидуальный"  date\_Of\_Birth = "1980-05-10"  passport\_Number = "1253567890"  Статус = Завершилось  Цвет = Красный | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Клиенты открыта | | Отображается форма Клиенты |  |

Таблица 5 – Тестирование некорректного редактирования информации в таблице Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Клиенты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  first\_Name = "Иван"  last\_Name = "Иванов"  middle\_Name = "Иванович"  address = "ул. Ленина, 1"  email = "ivan.ivanov@gmail.com"  phone = "+79001234567"  client\_Type =  date\_Of\_Birth = "1980-05-10"  passport\_Number = "1234567890"  Статус =  Цвет = Зеленый | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поле client\_Type пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Клиенты  открыта | | Отображается форма Клиенты |  |

Таблица 6 – Тестирование корректного добавления информации в таблицу Портфели

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Портфели | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфели | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  user\_Name = 'valid\_user'  portfolio\_Type = 'Инвестиционный'  portfolio\_Name = 'Портфель 1'  Статус = Не началось | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфели открыта | | Отображается форма Портфели |  |

Таблица 7 – Тестирование некорректного добавления информации в таблицу Портфели

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Портфели | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфели | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  user\_Name =  portfolio\_Type =  portfolio\_Name =  Статус = Не началось | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфели  открыта | | Отображается форма Портфели |  |

Таблица 8 – Тестирование корректного редактирования информации в таблице Портфели

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Портфели | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфели | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  user\_Name = 'valid\_client  portfolio\_Type = 'Инвестиционный'  portfolio\_Name = 'Портфель клиента'  Статус = Не началось | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфели открыта | | Отображается форма Портфели |  |

Таблица 9 – Тестирование некорректного редактирования информации в таблице Портфели

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Портфели | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Мероприятие | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  user\_Name =  portfolio\_Type =  portfolio\_Name =  Статус = Не началось | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфели открыта | | Отображается форма Портфели |  |

Таблица 10 – Тестирование корректного добавления информации в Инструменты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Инструменты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Инструменты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  instrument\_Type = 'Акция'  ticker\_Symbol = 'AAPL'  name = 'Apple Inc.' | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Инструменты открыта | | Отображается форма Инструменты |  |

Таблица 11 – Тестирование некорректного добавления информации в таблицу Инструменты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Инструменты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Инструменты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  instrument\_Type =  ticker\_Symbol =  name = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, так как поле пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Инструменты открыта | | Отображается форма Инструменты |  |

Таблица 12 – Тестирование корректного редактирования информации в таблице Инструменты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Инструменты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Инструменты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  instrument\_Type = 'Акция'  ticker\_Symbol = 'AAPL'  name = 'Apple Inc.' | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Инструменты открыта | | Отображается форма Инструменты |  |

Таблица 13 – Тестирование некорректного редактирования информации в таблице Инструменты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Инструменты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Инструменты | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  instrument\_Type =  ticker\_Symbol =  name = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Инструменты открыта | | Отображается форма Инструменты |  |

**6. ЗАКЛЮЧЕНИЕ**

**6.1 Оценка качества программного средства с помощью метрик**

Оценка качества программного средства проводится исходя из стандарта оценки качества ISО 9126.

На компьютере следующей конфигурации:

Установленная операционная система – Windows 7 Ultimate 32бит.

Процессор: Intel Pentium G2030 3.0 GHz.

ОЗУ: 2 Gb

Таблица 14 – Требования к количественным характеристикам качества программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Надежность** |  | | |
| *Завершенность:*  *-* наработка на отказ при отсутствии рестарта; | Часы | 10 | 10 |
| *Устойчивость:*  - относительные ресурсы на обеспечение надежности и рестарта. | % | 0,5 | 0,5 |
| *Восстанавливаемость:*  -длительность восстановления. | Минуты | 2 | 2 |
| *Доступность-готовность:*  -относительное время работоспособного функционирования. | Вероятность | 0,998 | 0,998 |
| **Эффективность** |  | | |
| *Временная эффективность:*  *-*время отклика - получение результатов на типовое задание;  - пропускная способность число типовых заданий, исполняемых в единицу времени. | Секунды  Число в минуту | 3  60 | 3  60 |
| *Используемость ресурсов:*  *-* относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства. |  | 0,001 | 0,001 |

Таблица 15 – Требования к качественным характеристикам программного средства.

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Практичность** |  | | |
| *Простота использования:*  **-**среднее время ввода заданий;  -среднее время отклика на задание. | Секунды  Секунды | 10  3 | 10  3 |
| *Изучаемость :*  - трудоемкость изучения применения ПС;  -продолжительность изучения;  -объем эксплуатационной документации; | Чел.-часы  Часы  Страницы | 5  2  15 | 5  2  15 |
| **Сопровождаемость** |  | | |
| *Изменяемость:*  -трудоемкость подготовки изменений;  -длительность подготовки изменений. | Чел.-часы  Часы | 5  5 | 5  5 |
| *Тестируемость:*  *-трудоемкость тестирования изменений;*  *- длительность тестирования изменений.* | Чел.-часы  Часы | 3  3 | 3  3 |
| **Мобильность** |  | | |
| *Адаптируемость:*  - трудоемкость адаптации;  -длительность адаптации. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Простота установки:*  -трудоемкость инсталляции;  -длительность инсталляции. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Замещаемость:*  -трудоемкость замены компонентов;  - длительность замены компонентов. | Чел.-часы  Часы | 0,2  0,2 | 0,2  0,2 |

Таблица 16 – Сложность

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип объекта | Количество | Вес | | | Итого |
|  |  | Простой | Средний | Сложный |  |
| Экран | 9 | x1 | x2 | **x3** | =27 |
| Отчет | 2 | x2 | **x5** | x8 | =10 |

Таблица 17 – Сложность экрана

|  |  |  |  |
| --- | --- | --- | --- |
| Экраны | Количество Категорииских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | **Всего <8** | Всего >8 |
|  | (<2 срв, <3 клт) | **(2-3 срв, 3-5 клт)** | (>3 срв, >5 клт) |
| <3 | Простой | Простой | Средний |
| 3-7 | Простой | Средний | Сложный |
| **>8** | Средний | **Сложный** | Сложный |

Таблица 18 – Сложность отчета

|  |  |  |  |
| --- | --- | --- | --- |
| Отчеты | Количество Категорииских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | Всего <8 | Всего >8 |
|  | (<2 срв, <3 клт) | **(2-3 срв, 3-5 клт)** | (>3 срв, >5 клт) |
| 0 или 1 | Простой | Простой | Средний |
| **2 или 3** | Простой | **Средний** | Сложный |
| >4 | Средний | Сложный | Сложный |

Производительность = 4 тысяч строк кода/2 месяца= 2 тысяч строк кода/месяц

Качество = 2 ошибки /4 тысяч строк кода=0,5 ошибок/тысяч строк кода

**7. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. К. Г. Хорстманн, Г. Корнелл - "Java. Полное руководство" (4-е издание, 2019) - 960 страниц

2. Брюс Эккель - "Философия Java" (3-е издание, 2007) - 864 страницы

3. С. Л. Мэтьюс - "Java. Руководство для начинающих" (2015) - 344 страницы

4. Кэтти Сьерра, Берти Бейтс - "Понимание Java. Как стать выдающимся программистом" (2006) - 600 страниц

5. Д. Фланаган - "Java в примерах" (2014) - 660 страниц

6. С. Б. Блейк - "Java: Библиотека программиста" (2017) - 500 страниц

7. Е. Б. Летунов - "Java. Разработка на платформе Java SE 8" (2016) - 480 страниц

8. М. Фаулер - "Java. Модернизируемый код" (2018) - 432 страницы

9. С. Оукс - "Java. Эффективная работа с памятью" (2013) - 480 страниц

10. Г. Л. Герберт - "Java. Практика программирования" (2012) - 400 страниц

11. ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению

12. ГОСТ 19.102-77.ЕСПД. Стадии разработки

13. ГОСТ 19.505-79.ЕСПД. Руководство оператора. Требования к содержанию и оформлению

14. ГОСТ 19.002-80.ЕСПД. Схемы алгоритмов и программ. Правила выполнения

15. ГОСТ 19.404-79.ЕСПД. Пояснительная записка. Требования к содержанию и оформлению

**Приложение А**

Разработка интегрированного программного модуля «Инвестиционная компания»

Текст кода программного средства

460.ЗМК.14-01 12 01

**Текст кода программного средства**

**base-styles.css**

.root {  
 -fx-font-style: italic;  
 -fx-font-family: "Arial";  
 -fx-main-bg-color: #F8F8F8;  
 -fx-additional-bg-color: #4b4b4da1;  
 -fx-akcent-bg-color: #0a0145;  
}  
.cancel-button {  
 -fx-font-style: italic;  
 -fx-font-family: "Arial";  
 -fx-border-style: solid;  
 -fx-border-color: #FF5555;  
 -fx-background-color: #FF5555;  
 -fx-padding: 5 5;  
 -fx-text-fill: -fx-main-bg-color;  
 -fx-text-align: center;  
 -fx-display: inline-block;  
 -fx-font-size: 16px;  
 -fx-font-weight: bold;  
}  
.ok-button {  
 -fx-font-style: italic;  
 -fx-font-family: "Arial";  
 -fx-border-style: solid;  
 -fx-border-color: #55FF55;  
 -fx-background-color: #55FF55;  
 -fx-padding: 5 5;  
 -fx-text-fill: -fx-main-bg-color;  
 -fx-text-align: center;  
 -fx-display: inline-block;  
 -fx-font-size: 16px;  
 -fx-font-weight: bold;  
}  
.button:hover {  
 -fx-background-color: white;  
 -fx-border-color: none;  
 -fx-text-fill: #000000;  
}  
.text-field  
{  
 -fx-border-style: solid;  
 -fx-border-width: 0.1px;  
 -fx-border-color: black;  
}  
.label  
{  
 -fx-font-weight: bold;  
}  
.percent-label  
{  
 -fx-text-fill: black;  
 -fx-font-weight: bold;  
 -fx-font-size: 28px;  
}  
.title-label  
{  
 -fx-font-weight: bold;  
 -fx-font-size: 18px;  
 -fx-text-fill: -fx-akcent-bg-color  
}  
.header-label  
{  
  
 -fx-font-size: 18px;  
 -fx-font-weight: bold;  
 -fx-background-color: -fx-akcent-bg-color;  
 -fx-text-fill: #FFF;  
}  
.user-label  
{  
  
 -fx-font-size: 14px;  
 -fx-font-weight: bold;

-fx-background-color: -fx-akcent-bg-color;  
 -fx-text-fill: #FFF;  
}  
.instrument-cell{  
 -fx-text-fill: black;  
 -fx-font-size: 11px;  
}  
.table-view {  
 -fx-base: -fx-main-bg-color;  
 -fx-table-cell-border-color: -fx-additional-bg-color;  
 -fx-table-header-border-color: -fx-akcent-bg-color;  
 -fx-padding: 5;  
}  
.table-view .column-header {  
 -fx-size: 35;  
 -fx-border-width: 0 0 1 0;  
 -fx-background-color: -fx-akcent-bg-color;  
 -fx-border-color:  
 transparent  
 transparent  
 derive(-fx-base, 80%)  
 transparent;  
 -fx-border-insets: 0 10 1 0;  
}  
.table-view .column-header .label {  
 -fx-font-size: 12pt;  
 -fx-text-fill: -fx-main-bg-color;  
 -fx-alignment: center-left;  
 -fx-opacity: 1;  
}  
.table-view .table-cell{  
 -fx-text-fill: white;  
 -fx-font-size: 10pt;  
 -fx-text-fill: black;  
 -fx-font-weight: bold;  
}  
.table-row-cell{  
 -fx-background-color: derive(-fx-additional-bg-color, 90%);  
 -fx-background-insets: 0, 0 0 1 0;  
 -fx-padding: 0.0em; /\* 0 \*/  
  
}  
.table-row-cell:odd{  
 -fx-background-color: derive(-fx-main-bg-color, 90%);  
 -fx-background-insets: 0, 0 0 1 0;  
 -fx-padding: 0.0em; /\* 0 \*/  
}  
.table-row-cell:selected {  
 -fx-background-color: #005797;  
 -fx-background-insets: 0;  
 -fx-background-radius: 1;  
}  
.menu-bar {  
 -fx-background-color:-fx-akcent-bg-color;  
}  
.context-menu {  
 -fx-background-color: derive(-fx-akcent-bg-color, 50%);  
}  
.menu-bar .label {  
 -fx-font-size: 11pt;  
 -fx-text-fill: white;  
 -fx-opacity: 0.9;  
}  
.combo-box-popup .list-view .list-cell  
{  
 -fx-font-size: 10pt;  
 -fx-padding: 4 0 4 5;  
 -fx-background-color: -fx-main-bg-color;  
 -fx-text-fill: -fx-akcent-bg-color;  
 -fx-font-weight: bold;  
}  
.combo-box-base  
{  
 -fx-font-size: 10pt;  
  
 -fx-background-color: -fx-main-bg-color;  
 -fx-text-fill: -fx-akcent-bg-color;  
 -fx-font-weight: bold;  
}  
.text-field  
{  
 -fx-font-size: 10pt;  
 -fx-background-color: -fx-main-bg-color;  
 -fx-text-fill: black;  
 -fx-font-weight: bold;  
}

**Класс Manager**

package ru.demo.investmentapp.util;  
  
import com.itextpdf.text.\*;  
import com.itextpdf.text.pdf.BaseFont;  
import com.itextpdf.text.pdf.PdfPCell;  
import com.itextpdf.text.pdf.PdfPTable;  
import com.itextpdf.text.pdf.PdfWriter;  
import javafx.application.Platform;  
import javafx.fxml.FXMLLoader;  
import javafx.geometry.Rectangle2D;  
import javafx.scene.Scene;  
import javafx.scene.control.Alert;  
import javafx.scene.control.ButtonType;  
import javafx.stage.FileChooser;  
import javafx.stage.Modality;  
import javafx.stage.Screen;  
import javafx.stage.Stage;  
import ru.demo.investmentapp.InvestmentApp;  
import ru.demo.investmentapp.controller.MainWindowController;  
import ru.demo.investmentapp.model.\*;  
import ru.demo.investmentapp.service.\*;  
  
import java.io.\*;  
import java.util.List;  
import java.util.Optional;  
  
public class Manager {  
 public static Rectangle2D screenSize = Screen.getPrimary().getVisualBounds();  
 public static UserService userService = new UserService();  
 public static ClientService clientService = new ClientService();  
 public static MarketDataService marketDataService = new MarketDataService();  
 public static ReportService reportService = new ReportService();  
 public static PortfolioService portfolioService = new PortfolioService();  
  
 public static MainWindowController mainWindowController;  
 public static User currentUser = null;  
 public static MarketData currentMarketData;  
 public static Stage mainStage;  
 public static Stage secondStage;  
 public static Stage currentStage;  
 public static Role currentRole;  
 public static User currentEditUser;  
 public static Instrument currentInstrument;  
 public static Report currentReport;  
 public static Portfolio currentPortfolio;  
 public static PortfolioStrategy currentPortfolioStrategy;  
 public static PortfolioPosition currentPortfolioPosition;  
 public static Account currentAccount;  
 public static Client currentClient;  
 public static Commission currentCommission;  
 public static CompanyInfo currentCompanyInfo;  
 public static InvestmentStrategy currentInvestmentStrategy;  
 public static Transaction currentTransaction;  
  
 public static void ShowPopup() {  
 Alert alert = new Alert(Alert.AlertType.CONFIRMATION);  
 alert.setTitle("Закрыть приложение");  
 alert.setHeaderText("Вы хотите выйти из приложения?");  
 alert.setContentText("Все несохраненные данные, будут утеряны");  
  
 Optional<ButtonType> result = alert.showAndWait();  
 if (result.get() == ButtonType.OK) {  
 Platform.exit();  
 }  
 }  
  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Ошибка");  
 alert.setHeaderText(message);  
 alert.showAndWait();  
 }  
  
 public static void MessageBox(String title, String header, String message, Alert.AlertType alertType) {  
 Alert alert = new Alert(alertType);  
 alert.setTitle(title);  
 alert.setHeaderText(header);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
  
 public static Optional<ButtonType> ShowConfirmPopup() {  
 Alert alert = new Alert(Alert.AlertType.CONFIRMATION);  
 alert.setTitle("Удаление");  
 alert.setHeaderText("Вы действительно хотите удалить запись?");  
 alert.setContentText("Также будут удалены все зависимые от этой записи данные");  
 Optional<ButtonType> result = alert.showAndWait();  
 return result;  
 }  
  
 public static void LoadSecondStageScene(String fxmlFileName, String title)  
 {  
 FXMLLoader fxmlLoader = new FXMLLoader(InvestmentApp.class.getResource(fxmlFileName));  
 Scene scene = null;  
 try {  
 scene = new Scene(fxmlLoader.load(), screenSize.getWidth(), screenSize.getHeight()- 50);  
 scene.getStylesheets().add("base-styles.css");  
 Manager.secondStage.setScene(scene);  
 Manager.secondStage.setMaximized(true);  
 Manager.secondStage.setTitle(title);  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 public static void ShowEditWindow(String fxmlFileName) {  
 Stage newWindow = new Stage();  
 FXMLLoader fxmlLoader = new FXMLLoader(InvestmentApp.class.getResource(fxmlFileName));  
  
 Scene scene = null;  
 try {  
 scene = new Scene(fxmlLoader.load());  
 scene.getStylesheets().add("base-styles.css");  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 newWindow.setTitle("Изменить данные");  
 newWindow.initOwner(Manager.secondStage);  
 newWindow.setResizable(false);  
 newWindow.initModality(Modality.WINDOW\_MODAL);  
 newWindow.setScene(scene);  
 Manager.currentStage = newWindow;  
 newWindow.showAndWait();  
 Manager.currentStage = null;  
 }  
  
 public static void PrintClientToPDF(Client client) throws FileNotFoundException, DocumentException {  
 String FONT = "src/main/resources/fonts/arial.ttf";  
 List<Client> clients = clientService.findAll();  
  
 FileChooser fileChooser = new FileChooser();  
  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("PDF files (\*.PDF)", "\*.pdf");  
 fileChooser.getExtensionFilters().add(extFilter);  
  
 File file = fileChooser.showSaveDialog(mainStage);  
  
 if (file != null) {  
 Document document = new Document();  
 PdfWriter.getInstance(document, new FileOutputStream(file));  
 Font font = FontFactory.getFont(FONT, "cp1251", BaseFont.EMBEDDED, 10);  
 document.open();  
 document.add(new Paragraph("Пользователь: " + clients.getFirst().getFullName(), font));  
  
 document.add(Chunk.NEWLINE);  
 PdfPTable table = new PdfPTable(new float[]{10, 10, 10, 10, 10, 10, 10, 10, 10, 10 });  
 PdfPCell header = new PdfPCell();  
 header.setBackgroundColor(BaseColor.LIGHT\_GRAY);  
 header.setBorderWidth(2);  
 header.setPhrase(new Phrase("№", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Имя", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Фамилия", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Отчество", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Адрес", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Email", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Телефон", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Тип клиента", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Дата рождения", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Паспорт", font));  
 table.addCell(header);  
 table.setWidthPercentage(100);  
  
 int k = 1;  
 for (Client item : clients) {  
 table.addCell(String.valueOf(k));  
 PdfPCell title = new PdfPCell();  
 title.setPhrase(new Phrase(item.getFirstName(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getLastName(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getMiddleName() , font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getAddress(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getEmail(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getPhone(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getClientType(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(String.valueOf(item.getDateOfBirth()), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getPassportNumber(), font));  
 table.addCell(title);  
 k++;  
 }  
 document.add(table);  
 document.close();  
 }  
 }  
  
 public static void PrintMarketDataToPDF(MarketData marketData) throws FileNotFoundException, DocumentException {  
 String FONT = "src/main/resources/fonts/arial.ttf";  
 List<MarketData> marketDatas = marketDataService.findAll();  
  
 FileChooser fileChooser = new FileChooser();  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("PDF files (\*.PDF)", "\*.pdf");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showSaveDialog(mainStage);  
  
 if (file != null) {  
 Document document = new Document();  
 PdfWriter.getInstance(document, new FileOutputStream(file));  
 Font font = FontFactory.getFont(FONT, "cp1251", BaseFont.EMBEDDED, 10);  
 document.open();  
  
 document.add(Chunk.NEWLINE);  
 PdfPTable table = new PdfPTable(new float[]{10, 18, 18, 18, 18});  
 PdfPCell header = new PdfPCell();  
 header.setBackgroundColor(BaseColor.LIGHT\_GRAY);  
 header.setBorderWidth(2);  
 header.setPhrase(new Phrase("№", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Инструмент", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Дата", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Цена", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Количество", font));  
 table.addCell(header);  
 table.setWidthPercentage(100);  
  
 int k = 1;  
 for (MarketData item : marketDatas) {  
 table.addCell(String.valueOf(k));  
 PdfPCell title = new PdfPCell();  
 title.setPhrase(new Phrase(item.getInstrument().getInstrumentType(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(String.valueOf(item.getDataDate()), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(String.valueOf(item.getPrice()), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(String.valueOf(item.getVolume()), font));  
 table.addCell(title);  
  
 k++;  
 }  
 document.add(table);  
 document.close();  
 }  
 }  
  
 public static void PrintReportToPDF(Report report) throws FileNotFoundException, DocumentException {  
 String FONT = "src/main/resources/fonts/arial.ttf";  
 List<Report> reports = reportService.findAll();  
  
 FileChooser fileChooser = new FileChooser();  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("PDF files (\*.PDF)", "\*.pdf");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showSaveDialog(mainStage);  
  
 if (file != null) {  
 Document document = new Document();  
 PdfWriter.getInstance(document, new FileOutputStream(file));  
 Font font = FontFactory.getFont(FONT, "cp1251", BaseFont.EMBEDDED, 10);  
 document.open();  
 document.add(new Paragraph("Тип отчета: " + reports.getFirst().getReportType(), font));  
  
 document.add(Chunk.NEWLINE);  
 PdfPTable table = new PdfPTable(new float[]{12, 22, 22, 22, 22});  
 PdfPCell header = new PdfPCell();  
 header.setBackgroundColor(BaseColor.LIGHT\_GRAY);  
 header.setBorderWidth(2);  
 header.setPhrase(new Phrase("№", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Тип отчета", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Дата", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Содержание", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Логин пользователя", font));  
 table.addCell(header);  
 table.setWidthPercentage(100);  
  
 int k = 1;  
 for (Report item : reports) {  
 table.addCell(String.valueOf(k));  
 PdfPCell title = new PdfPCell();  
 title.setPhrase(new Phrase(item.getReportType(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(String.valueOf(item.getReportDate()), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getReportContent(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getUser().getUserName(), font));  
 table.addCell(title);  
  
 k++;  
 }  
 document.add(table);  
 document.close();  
 }  
 }  
  
 public static void PrintPortfolioToPDF(Portfolio portfolio) throws FileNotFoundException, DocumentException {  
 String FONT = "src/main/resources/fonts/arial.ttf";  
 List<Portfolio> portfolios = portfolioService.findAll();  
  
 FileChooser fileChooser = new FileChooser();  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("PDF files (\*.PDF)", "\*.pdf");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showSaveDialog(mainStage);  
  
 if (file != null) {  
 Document document = new Document();  
 PdfWriter.getInstance(document, new FileOutputStream(file));  
 Font font = FontFactory.getFont(FONT, "cp1251", BaseFont.EMBEDDED, 10);  
 document.open();  
 document.add(new Paragraph("Название портфолио: " + portfolios.getFirst().getPortfolioName(), font));  
  
 document.add(Chunk.NEWLINE);  
 PdfPTable table = new PdfPTable(new float[]{10, 30, 30, 30});  
 PdfPCell header = new PdfPCell();  
 header.setBackgroundColor(BaseColor.LIGHT\_GRAY);  
 header.setBorderWidth(2);  
 header.setPhrase(new Phrase("№", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Логин пользователя", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Тип портфолио", font));  
 table.addCell(header);  
 header.setPhrase(new Phrase("Название", font));  
 table.addCell(header);  
 table.setWidthPercentage(100);  
  
 int k = 1;  
 for (Portfolio item : portfolios) {  
 table.addCell(String.valueOf(k));  
 PdfPCell title = new PdfPCell();  
 title.setPhrase(new Phrase(item.getUser().getUserName(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getPortfolioType(), font));  
 table.addCell(title);  
 title.setPhrase(new Phrase(item.getPortfolioName(), font));  
 table.addCell(title);  
  
 k++;  
 }  
 document.add(table);  
 document.close();  
 }  
 }  
}

**Окно авторизации**

**Код программы**

package ru.demo.investmentapp.controller;  
  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.FXMLLoader;  
import javafx.fxml.Initializable;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.control.PasswordField;  
import javafx.scene.control.TextField;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.RowConstraints;  
import javafx.stage.Stage;  
import ru.demo.investmentapp.InvestmentApp;  
import ru.demo.investmentapp.model.User;  
import ru.demo.investmentapp.util.MakeCaptcha;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.IOException;  
import java.net.URL;  
import java.util.\*;  
  
import static ru.demo.investmentapp.util.Manager.\*;  
  
public class LoginController implements Initializable {  
 boolean isWrongCaptha;  
 boolean isShowCaptha;  
 String captchaCode;  
 int secondsLeft;  
 @FXML  
 RowConstraints ThirdRow;  
 @FXML  
 Button BtnRenewCaptcha;  
 @FXML  
 private Button BtnCancel;  
 @FXML  
 private Button BtnOk;  
 @FXML  
 private PasswordField PasswordField;  
 @FXML  
 private TextField TextFieldUsername;  
 @FXML  
 private TextField TextFieldCaptcha;  
 @FXML  
 private ImageView ImageViewCaptcha;  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 initController();  
 }  
 @FXML  
 void BtnRenewCaptchaAction(ActionEvent event) {  
 generateCaptcha();  
 }  
 @FXML  
 void BtnCancelAction(ActionEvent event) {  
 Manager.*ShowPopup*();  
 }  
 @FXML  
 void BtnOkActon(ActionEvent event) {  
 List<User> users = *userService*.findAll();  
 Optional<User> person = users.stream().filter(user -> user.getUserName().equals(TextFieldUsername.getText()) &&  
 user.getPassword().equals(PasswordField.getText())).findFirst();  
  
 if (person.isEmpty() && isShowCaptha && !TextFieldCaptcha.getText().equals(captchaCode)) {  
 System.*out*.println("Bad error");  
 *ShowErrorMessageBox*("Не верный логин, пароль или текст капчи");  
 blockButtons();  
 return;  
 }  
 if (person.isEmpty() && (!isShowCaptha)) {  
 System.*out*.println("Bad error");  
 generateCaptcha();  
 isShowCaptha = true;  
 ThirdRow.setPrefHeight(50);  
 ImageViewCaptcha.setVisible(true);  
 TextFieldCaptcha.setVisible(true);  
 BtnRenewCaptcha.setVisible(true);  
 *ShowErrorMessageBox*("Не верный логин или пароль");  
 return;  
 }  
 if (person.isPresent() && isShowCaptha && !TextFieldCaptcha.getText().equals(captchaCode)) {  
 blockButtons();  
 *ShowErrorMessageBox*("Не верный логин, пароль или текст капчи");  
 return;  
 }  
  
 if (person.isPresent() && isShowCaptha && TextFieldCaptcha.getText().equals(captchaCode)) {  
 showMainWindow(person.get());  
 return;  
 }  
  
 if (person.isPresent() && !isShowCaptha) {  
 showMainWindow(person.get());  
 }  
 }  
  
 public void generateCaptcha() {  
 try {  
 ImageViewCaptcha.setImage(MakeCaptcha.*CreateImage*(150, 40, 4));  
 captchaCode = MakeCaptcha.*captchaCode*();  
 System.*out*.println(captchaCode);  
 } catch (IOException e) {  
 System.*out*.println(e.getMessage());  
 }  
 }  
  
 public void showMainWindow(User person) {  
 Manager.*currentUser* = person;  
 System.*out*.println(Manager.*currentUser*);  
 Manager.*mainStage*.hide();  
 Stage newWindow = new Stage();  
  
 FXMLLoader fxmlLoader = new FXMLLoader(InvestmentApp.class.getResource("main-view.fxml"));  
 Scene scene = null;  
 try {  
 scene = new Scene(fxmlLoader.load(), *screenSize*.getWidth(), *screenSize*.getHeight());  
 scene.getStylesheets().add("base-styles.css");  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 newWindow.setTitle("Aurora Growth");  
 newWindow.setMaximized(true);  
 newWindow.setScene(scene);  
 newWindow.setOnCloseRequest(e -> {  
 Manager.*mainStage*.show();  
 });  
 Manager.*secondStage* = newWindow;  
  
 newWindow.show();  
 }  
  
 public void initTimer() {  
 TimerTask task = new TimerTask() {  
 public void run() {  
 System.*out*.println("Task performed on: " + new Date() + "n" +  
 "Thread's name: " + Thread.*currentThread*().getName());  
 secondsLeft--;  
 if (secondsLeft == 0) ;  
 {  
 BtnOk.setDisable(false);  
 BtnCancel.setDisable(false);  
 this.cancel();  
 }  
 }  
 };  
 Timer timer = new Timer("Timer");  
  
 long delay = 10000L;  
 timer.schedule(task, delay);  
 }  
  
 public void blockButtons() {  
 initTimer();  
 secondsLeft = 10;  
 BtnOk.setDisable(true);  
 BtnCancel.setDisable(true);  
 }  
  
 public void initController() {  
 ThirdRow.setPrefHeight(0);  
 TextFieldUsername.setText("admin");  
 PasswordField.setText("1");  
 TextFieldCaptcha.setVisible(false);  
 BtnRenewCaptcha.setVisible(false);  
 ImageViewCaptcha.setVisible(false);  
 isWrongCaptha = false;  
 isShowCaptha = false;  
 captchaCode = "";  
 secondsLeft = 0;  
 }  
}

**Код интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.Insets?>  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.PasswordField?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.image.Image?>  
<?import javafx.scene.image.ImageView?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.ColumnConstraints?>  
<?import javafx.scene.layout.GridPane?>  
<?import javafx.scene.layout.HBox?>  
<?import javafx.scene.layout.RowConstraints?>  
  
<AnchorPane maxHeight="200.0" maxWidth="350.0" minHeight="200.0" minWidth="350.0" prefHeight="200.0" prefWidth="350.0" xmlns="http://javafx.com/javafx/23" xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.LoginController">  
 <children>  
 <GridPane layoutX="25.0" layoutY="34.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <columnConstraints>  
 <ColumnConstraints hgrow="SOMETIMES" maxWidth="100.0" minWidth="100.0" prefWidth="100.0" />  
 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />  
 </columnConstraints>  
 <rowConstraints>  
 <RowConstraints maxHeight="40.0" minHeight="40.0" prefHeight="40.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="30.0" minHeight="30.0" prefHeight="30.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="30.0" minHeight="30.0" prefHeight="30.0" vgrow="SOMETIMES" />  
 <RowConstraints fx:id="ThirdRow" maxHeight="-Infinity" minHeight="0.0" prefHeight="50.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="1.7976931348623157E308" minHeight="50.0" prefHeight="50.0" vgrow="SOMETIMES" />  
 </rowConstraints>  
 <children>  
 <HBox alignment="CENTER" fillHeight="false" nodeOrientation="LEFT\_TO\_RIGHT" prefHeight="100.0" prefWidth="200.0" spacing="10.0" GridPane.columnSpan="2" GridPane.hgrow="ALWAYS" GridPane.rowIndex="4" GridPane.vgrow="SOMETIMES">  
 <children>  
 <Button fx:id="BtnOk" alignment="CENTER" defaultButton="true" maxHeight="-Infinity"  
 maxWidth="-Infinity" mnemonicParsing="false" onAction="#BtnOkActon" prefHeight="34.0"  
 prefWidth="100.0" text="OK" textAlignment="CENTER" HBox.hgrow="ALWAYS" styleClass="ok-button"/>  
 <Button fx:id="BtnCancel" alignment="CENTER" cancelButton="true" maxHeight="-Infinity"  
 maxWidth="-Infinity" mnemonicParsing="false" onAction="#BtnCancelAction" prefHeight="34.0"  
 prefWidth="100.0" text="Cancel" textAlignment="CENTER" HBox.hgrow="ALWAYS" styleClass="cancel-button"/>  
 </children>  
 <opaqueInsets>  
 <Insets bottom="5.0" left="5.0" right="5.0" top="5.0" />  
 </opaqueInsets>  
 <padding>  
 <Insets bottom="5.0" left="5.0" right="5.0" top="5.0" />  
 </padding>  
 <GridPane.margin>  
 <Insets />  
 </GridPane.margin>  
 </HBox>  
 <Label alignment="CENTER" prefHeight="40.0" prefWidth="374.0" styleClass="header-label" text="Aurora Growth" textAlignment="CENTER" GridPane.columnSpan="2" GridPane.vgrow="SOMETIMES" />  
 <PasswordField fx:id="PasswordField" promptText="Пароль" GridPane.columnSpan="2" GridPane.rowIndex="2" GridPane.vgrow="ALWAYS">  
 <GridPane.margin>  
 <Insets left="30.0" right="30.0" />  
 </GridPane.margin></PasswordField>  
 <TextField fx:id="TextFieldUsername" promptText="Логин" GridPane.columnSpan="2" GridPane.rowIndex="1" GridPane.vgrow="ALWAYS">  
 <GridPane.margin>  
 <Insets left="30.0" right="30.0" />  
 </GridPane.margin></TextField>  
 <ImageView fitHeight="33.0" fitWidth="85.0" pickOnBounds="true" preserveRatio="true" GridPane.halignment="CENTER" GridPane.valignment="CENTER">  
 <image>  
 <Image url="@../../../../../../../../kursovaya/investment-App/src/main/resources/ru/demo/investmentapp/investment.png" />  
 </image>  
 </ImageView>  
 <ImageView fx:id="ImageViewCaptcha" fitHeight="40.0" fitWidth="150.0" pickOnBounds="true" preserveRatio="true" GridPane.columnSpan="2" GridPane.rowIndex="3">  
 <GridPane.margin>  
 <Insets bottom="5.0" left="60.0" top="5.0" />  
 </GridPane.margin></ImageView>  
 <TextField fx:id="TextFieldCaptcha" maxHeight="-Infinity" maxWidth="-Infinity" prefHeight="26.0" prefWidth="125.0" promptText="Введите капчу" GridPane.columnIndex="1" GridPane.rowIndex="3">  
 <GridPane.margin>  
 <Insets left="130.0" right="5.0" />  
 </GridPane.margin>  
 </TextField>  
 <Button fx:id="BtnRenewCaptcha" alignment="CENTER" contentDisplay="CENTER" mnemonicParsing="false" onAction="#BtnRenewCaptchaAction" prefHeight="41.0" prefWidth="21.0" text="O" textAlignment="CENTER" GridPane.rowIndex="3">  
 <GridPane.margin>  
 <Insets bottom="5.0" left="5.0" top="5.0" />  
 </GridPane.margin></Button>  
 </children>  
 </GridPane>  
 </children>  
</AnchorPane>

**Форма MainWindowController**

**Код программы**

package ru.demo.investmentapp.controller;  
  
import java.io.IOException;  
import java.net.URL;  
import java.util.\*;  
import java.util.stream.Collectors;  
  
import jakarta.persistence.criteria.CriteriaBuilder;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.\*;  
import javafx.scene.layout.BorderPane;  
  
import ru.demo.investmentapp.controller.Instrument.InstrumentCell;  
import ru.demo.investmentapp.model.Instrument;  
import ru.demo.investmentapp.model.MarketData;  
import ru.demo.investmentapp.service.InstrumentService;  
import ru.demo.investmentapp.service.MarketDataService;  
import ru.demo.investmentapp.util.Manager;  
  
import static ru.demo.investmentapp.util.Manager.\*;  
  
public class MainWindowController implements Initializable {  
  
 @FXML  
 private ResourceBundle resources;  
 @FXML  
 private URL location;  
 @FXML  
 private BorderPane BorderPaneMainFrame;  
 @FXML  
 private ComboBox<String> ComboBoxInstrument;  
 @FXML  
 private ComboBox<String> ComboBoxSort;  
 @FXML  
 private Label LabelInfo;  
 @FXML  
 private Label LabelUser;  
 @FXML  
 private ListView<Instrument> ListViewInstrument;  
 @FXML  
 private MenuBar MenuBarFile;  
 @FXML  
 private MenuItem MenuItemMarketData;  
 @FXML  
 private TextField TextFieldSearch;  
  
 private int itemsCount;  
 private MarketDataService marketDataService = new MarketDataService();  
 private InstrumentService instrumentService = new InstrumentService();  
  
 @FXML  
 void ComboBoxInstrumentAction(ActionEvent event) {  
 filterData();  
 }  
 @FXML  
 void ComboBoxSortAction(ActionEvent event) {  
 filterData();  
 }  
 @FXML  
 void MenuItemMarketDataAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("MarketData/marketData-table-view.fxml", "Рыночные данные");  
 }  
 @FXML  
 void MenuItemStrategyAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("InvestmentStrategy/investmentStrategy-table-view.fxml", "Инвестиционные стратегии");  
 }  
 @FXML  
 void MenuItemCompanyInfoAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("CompanyInfo/companyInfo-table-view.fxml", "Информация о компаниях");  
 }  
 @FXML  
 void MenuItemPortfolioAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("Portfolio/portfolio-table-view.fxml", "Портфели");  
 }  
 @FXML  
 void MenuItemReportAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("Report/report-table-view.fxml", "Отчеты");  
 }  
 @FXML  
 void MenuItemUserAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("User/user-table-view.fxml", "Учетные данные");  
 }  
 @FXML  
 void MenuItemTransactionAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("Transaction/transaction-table-view.fxml", "Операции");  
 }  
 @FXML  
 void TextFieldSearchAction(ActionEvent event) {  
 filterData();  
 }

@Override  
 public void initialize(URL location, ResourceBundle resources) {  
 *mainWindowController* = this;  
 LabelUser.setText("Вы вошли как " + *currentUser*.getClient().getFirstName() + " " +  
 Manager.*currentUser*.getClient().getLastName());  
  
 List<Instrument> instrumentList = instrumentService.findAll();  
 instrumentList.add(0, new Instrument(0L, "Все", "", ""));  
  
 Set<String> instrumentTypes = instrumentList.stream().map(Instrument::getInstrumentType)  
 .collect(Collectors.*toSet*());  
  
 ObservableList<String> instruments = FXCollections.*observableArrayList*(instrumentTypes);  
 ComboBoxInstrument.setItems(instruments);  
 ObservableList<String> instrument = FXCollections.*observableArrayList*("по возрастанию цены", "по убыванию цены");  
 ComboBoxSort.setItems(instrument);  
  
 loadRole();  
 filterData();  
 }  
  
 void filterData() {  
 List<MarketData> marketDatas = marketDataService.findAll();  
 itemsCount = marketDatas.size();  
  
 // Фильтр по типу  
 if (!ComboBoxInstrument.getSelectionModel().isEmpty()) {  
 String instrumentType = ComboBoxInstrument.getValue();  
 // Проверка на "Все"  
 if (!instrumentType.equals("Все")) {  
 marketDatas = marketDatas.stream()  
 .filter(marketData -> marketData.getInstrument().getInstrumentType().equals(instrumentType))  
 .collect(Collectors.*toList*());  
 }  
 }  
  
 // Сортировка по цене  
 if (!ComboBoxSort.getSelectionModel().isEmpty()) {  
 String price = ComboBoxSort.getValue();  
 if (price.equals("по возрастанию цены")) {  
 marketDatas = marketDatas.stream().sorted(Comparator.*comparing*(MarketData::getPrice)).collect(Collectors.*toList*());  
 } else if (price.equals("по убыванию цены")) {  
 marketDatas = marketDatas.stream().sorted(Comparator.*comparing*(MarketData::getPrice).reversed()).collect(Collectors.*toList*());  
 }  
 }  
  
 // Поиск по имени инструмента  
 String searchText = TextFieldSearch.getText();  
 if (!searchText.isEmpty()) {  
 marketDatas = marketDatas.stream()  
 .filter(marketData -> marketData.getInstrument().getName().toLowerCase().contains(searchText.toLowerCase()))  
 .collect(Collectors.*toList*());  
 }  
  
 // Обновление ListViewInstrument  
 ListViewInstrument.getItems().clear();  
 ListViewInstrument.getItems().addAll(marketDatas.stream()  
 .map(MarketData::getInstrument)  
 .distinct().collect(Collectors.*toList*()));  
 ListViewInstrument.setCellFactory(lv -> new InstrumentCell());  
  
 // Обновление LabelInfo  
 int filteredItemsCount = marketDatas.size();  
 LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);  
 }  
  
 public void loadRole() {  
 MenuBarFile.setVisible(false);  
 // вход как админ  
 if (*currentUser*.getRole().getRoleId() == 1) {  
 MenuBarFile.setVisible(true);  
 }  
 // вход как пользователь  
 if (*currentUser*.getRole().getRoleId() == 2) {  
 MenuBarFile.setVisible(true);  
 MenuItemMarketData.setVisible(false);  
 }  
 }  
}

**Код интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.Insets?>  
<?import javafx.scene.control.ComboBox?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.ListView?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.BorderPane?>  
<?import javafx.scene.layout.FlowPane?>  
<?import javafx.scene.layout.HBox?>  
  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TableColumn?>  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0"  
 prefWidth="600.0" xmlns="http://javafx.com/javafx/23" xmlns:fx="http://javafx.com/fxml/1"  
 fx:controller="ru.demo.investmentapp.controller.MainWindowController">  
 <children >  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <center>  
 <BorderPane fx:id="BorderPaneMainFrame" BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane minHeight="-Infinity" nodeOrientation="LEFT\_TO\_RIGHT" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите название для поиска" />  
 <ComboBox fx:id="ComboBoxInstrument" onAction="#ComboBoxInstrumentAction" promptText="тип инвестиции" />  
 <ComboBox fx:id="ComboBoxSort" onAction="#ComboBoxSortAction" promptText="сортировка" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <ListView fx:id="ListViewInstrument" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefHeight="201.0" prefWidth="600.0" BorderPane.alignment="CENTER"/>  
 </center>  
  
 <bottom>  
 <FlowPane BorderPane.alignment="CENTER" />  
 </bottom>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar fx:id="MenuBarFile" maxWidth="1.7976931348623157E308" prefHeight="26.0" prefWidth="231.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemMarketData" mnemonicParsing="false" onAction="#MenuItemMarketDataAction" text="Рыночные данные" />  
 <MenuItem fx:id="MenuItemCompanyInfo" mnemonicParsing="false" onAction="#MenuItemCompanyInfoAction" text="Информация о компаниях" />  
 <MenuItem fx:id="MenuItemStrategy" mnemonicParsing="false" onAction="#MenuItemStrategyAction" text="Инвестиционные стратегии" />  
 <MenuItem fx:id="MenuItemUser" mnemonicParsing="false" onAction="#MenuItemUserAction" text="Учетные данные" />  
 <MenuItem fx:id="MenuItemPortfolio" mnemonicParsing="false" onAction="#MenuItemPortfolioAction" text="Портфели" />  
 <MenuItem fx:id="MenuItemTransaction" mnemonicParsing="false" onAction="#MenuItemTransactionAction" text="Операции" />  
 <MenuItem fx:id="MenuItemReport" mnemonicParsing="false" onAction="#MenuItemReportAction" text="Отчеты" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding></Label>  
 </children>  
 </HBox>  
 </top>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Страница Клиент**

**Код интерфейса**

<Page x:Class="AfishaApp.Pages.Сatalog"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:AfishaApp.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="Каталог"

IsVisibleChanged="PageIsVisibleChanged"

Style="{StaticResource base\_page}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="30"/>

</Grid.RowDefinitions>

<WrapPanel Grid.Row="0"

HorizontalAlignment="Stretch"

Background="{StaticResource additional}">

<StackPanel Orientation="Horizontal">

<TextBlock Text="Введите название для поиска:"

Style="{StaticResource base\_textblock}"/>

<TextBox Width="275" x:Name="TBoxSearch"

TextChanged="TBoxSearchTextChanged"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Выберите категорию"

Style="{StaticResource base\_textblock}" />

<ComboBox Width="275" x:Name="ComboCategory"

SelectionChanged="ComboTypeSelectionChanged"

DisplayMemberPath="CategoryName"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Выберите статус"

Style="{StaticResource base\_textblock}" />

<ComboBox Width="275" x:Name="ComboStatus"

SelectionChanged="ComboTypeSelectionChanged"

DisplayMemberPath="StatusName"/>

</StackPanel>

</WrapPanel>

<ListView x:Name="LViewGoods" Grid.Row="1"

ScrollViewer.HorizontalScrollBarVisibility="Disabled"

ItemContainerStyle="{StaticResource good\_item}">

<ListView.ItemsPanel >

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" HorizontalAlignment="Center"/>

</ItemsPanelTemplate>

</ListView.ItemsPanel>

<ListView.ItemTemplate>

<DataTemplate>

<Grid Margin="20" Width="300">

<Grid.RowDefinitions>

<RowDefinition Height="310" />

<RowDefinition Height="auto" />

<RowDefinition Height="20" />

</Grid.RowDefinitions>

<Image Width="240" Grid.Row="0" Stretch="Uniform"

HorizontalAlignment="Center" Margin="5"

Source="{Binding Path=GetPhoto}"/>

<StackPanel Grid.Row="1" Height="100">

<TextBlock Padding="3" Height="Auto"

Style="{StaticResource item\_textblock}"

VerticalAlignment="Center"

Foreground="{StaticResource akcent}"

FontWeight="Bold" FontSize="20"

Text="{Binding EventName}"/>

<TextBlock Text="{Binding EventPlace}" Height="Auto"

Style="{StaticResource item\_textblock}"

/>

</StackPanel>

<TextBlock Text="{Binding GetStatus}" Height="Auto"

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.Insets?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.BorderPane?>  
<?import javafx.scene.layout.FlowPane?>  
<?import javafx.scene.layout.HBox?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"  
 prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/22"  
 xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.Client.ClientTableViewController">  
 <children>  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <center>  
 <BorderPane BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane nodeOrientation="LEFT\_TO\_RIGHT" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите ФИО для поиска" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <TableView fx:id="TableViewClients" prefHeight="200.0" prefWidth="200.0" tableMenuButtonVisible="true" BorderPane.alignment="CENTER">  
 <columns>  
 <TableColumn id="TableColumnId" fx:id="TableColumnId" prefWidth="82.0" text="Id" />  
 <TableColumn id="TableColumnLastName" fx:id="TableColumnLastName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Фамилия" />  
 <TableColumn id="TableColumnFirstName" fx:id="TableColumnFirstName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Имя" />  
 <TableColumn id="TableColumnMiddleName" fx:id="TableColumnMiddleName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Отчество" />  
 <TableColumn id="TableColumnAddress" fx:id="TableColumnAddress" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Адрес" />  
 <TableColumn id="TableColumnEmail" fx:id="TableColumnEmail" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Email" />  
 <TableColumn id="TableColumnPhone" fx:id="TableColumnPhone" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Телефон" />  
 <TableColumn id="TableColumnClientType" fx:id="TableColumnClientType" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Тип клиента" />  
 <TableColumn id="TableColumnBirthday" fx:id="TableColumnBirthday" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="День рождения" />  
 <TableColumn id="TableColumnPassportNumber" fx:id="TableColumnPassportNumber" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Паспорт" />  
 </columns>  
 </TableView>  
 </center>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar prefHeight="25.0" prefWidth="381.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemBack" mnemonicParsing="false" onAction="#MenuItemBackAction" text="Назад" />  
 <MenuItem fx:id="MenuItemPrintToPDF" mnemonicParsing="false" onAction="#MenuItemPrintToPDFAction" text="Распечатать в PDF" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Правка">  
 <items>  
 <MenuItem fx:id="MenuItemAdd" mnemonicParsing="false" onAction="#MenuItemAddAction" text="Добавить" />  
 <MenuItem fx:id="MenuItemUpdate" mnemonicParsing="false" onAction="#MenuItemUpdateAction" text="Изменить" />  
 <MenuItem fx:id="MenuItemDelete" mnemonicParsing="false" onAction="#MenuItemDeleteAction" text="Удалить" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefWidth="120.0" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </Label>  
 </children>  
 </HBox>  
 </top>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Код программный**

package ru.demo.investmentapp.controller.Client;  
  
import com.itextpdf.text.DocumentException;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.\*;  
import ru.demo.investmentapp.model.Client;  
import ru.demo.investmentapp.service.ClientService;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.FileNotFoundException;  
import java.net.URL;  
import java.util.List;  
import java.util.Optional;  
import java.util.ResourceBundle;  
import java.util.stream.Collectors;  
  
import static ru.demo.investmentapp.util.Manager.\*;  
  
public class ClientTableViewController implements Initializable {  
  
 @FXML  
 private MenuItem MenuItemAdd;  
 @FXML  
 private MenuItem MenuItemBack;  
 @FXML  
 private MenuItem MenuItemDelete;  
 @FXML  
 private MenuItem MenuItemUpdate;  
 @FXML  
 private Label LabelInfo;  
 @FXML  
 private Label LabelUser;  
 @FXML  
 private TableColumn<Client, String> TableColumnId;  
 @FXML  
 private TableColumn<Client, String> TableColumnFirstName;  
 @FXML  
 private TableColumn<Client, String> TableColumnLastName;  
 @FXML  
 private TableColumn<Client, String> TableColumnMiddleName;  
 @FXML  
 private TableColumn<Client, String> TableColumnAddress;  
 @FXML  
 private TableColumn<Client, String> TableColumnEmail;  
  
 @FXML  
 private TableColumn<Client, String> TableColumnPhone;  
 @FXML  
 private TableColumn<Client, String> TableColumnClientType;  
 @FXML  
 private TableColumn<Client, String> TableColumnBirthday;  
 @FXML  
 private TableColumn<Client, String> TableColumnPassportNumber;  
 @FXML  
 private TableView<Client> TableViewClients;  
 @FXML  
 private TextField TextFieldSearch;  
 private int itemsCount;  
 private ClientService clientService = new ClientService();  
  
 @FXML  
 void TextFieldSearchAction(ActionEvent event) {  
 filterData();  
 }  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 initController();  
 }  
  
 public void initController() {  
 LabelUser.setText("Вы вошли как " + *currentUser*.getClient().getFirstName() + " " +  
 Manager.*currentUser*.getClient().getLastName());  
  
 setCellValueFactories();  
  
 filterData();  
 }  
  
 void filterData() {  
 List<Client> items = clientService.findAll();  
 itemsCount = items.size();  
  
 String searchText = TextFieldSearch.getText();  
 if (!searchText.isEmpty()) {  
 items = items.stream().filter(item -> item.getFullName().toLowerCase().contains(searchText.toLowerCase())).collect(Collectors.*toList*());  
 }  
 TableViewClients.getItems().clear();  
 for (Client client : items) {  
 TableViewClients.getItems().add(client);  
 }  
  
 int filteredItemsCount = items.size();  
 LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);  
 TableViewClients.setColumnResizePolicy(TableView.*CONSTRAINED\_RESIZE\_POLICY\_FLEX\_LAST\_COLUMN*);  
 }  
  
 private void setCellValueFactories() {  
 TableColumnId.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getClientId().toString()));  
 TableColumnFirstName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getFirstName()));  
 TableColumnLastName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getLastName()));  
 TableColumnMiddleName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getMiddleName()));  
 TableColumnAddress.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getAddress()));  
 TableColumnEmail.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getEmail()));  
 TableColumnPhone.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getPhone()));  
 TableColumnClientType.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getClientType()));  
 TableColumnBirthday.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getDateOfBirth().toString()));  
 TableColumnPassportNumber.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getPassportNumber()));  
 }  
  
 @FXML  
 void MenuItemAddAction(ActionEvent event) {  
 *currentClient* = null;  
 *ShowEditWindow*("Client/client-edit-view.fxml");  
 filterData();  
 }  
  
 @FXML  
 void MenuItemBackAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("main-view.fxml", "Aurora Growth");  
 }  
  
 @FXML  
 void MenuItemPrintToPDFAction(ActionEvent event) throws DocumentException, FileNotFoundException {  
 Client client = TableViewClients.getSelectionModel().getSelectedItem();  
  
 if (client != null) {  
 Manager.*PrintClientToPDF*(client);  
 *MessageBox*("Информация", "", "Данные сохранены успешно", Alert.AlertType.*INFORMATION*);return;  
 }  
 }  
  
 @FXML  
 void MenuItemDeleteAction(ActionEvent event) {  
 Client client = TableViewClients.getSelectionModel().getSelectedItem();  
  
 Optional<ButtonType> result = *ShowConfirmPopup*();  
 if (result.get() == ButtonType.*OK*) {  
 clientService.delete(client);  
 filterData();  
 }  
 }  
  
 @FXML  
 void MenuItemUpdateAction(ActionEvent event) {  
 Client client = TableViewClients.getSelectionModel().getSelectedItem();  
 *currentClient* = client;  
 *ShowEditWindow*("Client/client-edit-view.fxml");  
 filterData();  
 }  
  
}

**Страница Портфели**

**Файл интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.Insets?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.BorderPane?>  
<?import javafx.scene.layout.FlowPane?>  
<?import javafx.scene.layout.HBox?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"  
 prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/22"  
 xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.Portfolio.PortfolioTableViewController">  
 <children>  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <center>  
 <BorderPane BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane nodeOrientation="LEFT\_TO\_RIGHT" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите название для поиска" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <TableView fx:id="TableViewPortfolios" prefHeight="200.0" prefWidth="200.0" tableMenuButtonVisible="true" BorderPane.alignment="CENTER">  
 <columns>  
 <TableColumn id="TableColumnId" fx:id="TableColumnId" prefWidth="82.0" text="Id" />  
 <TableColumn id="TableColumnUserName" fx:id="TableColumnUserName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Логин пользователя" />  
 <TableColumn id="TableColumnPortfolioType" fx:id="TableColumnPortfolioType" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Тип портфолио" />  
 <TableColumn id="TableColumnPortfolioName" fx:id="TableColumnPortfolioName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Название" />  
 </columns>  
 </TableView>  
 </center>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar prefHeight="25.0" prefWidth="381.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemBack" mnemonicParsing="false" onAction="#MenuItemBackAction" text="Назад" />  
 <MenuItem fx:id="MenuItemPrintToPDF" mnemonicParsing="false" onAction="#MenuItemPrintToPDFAction" text="Распечатать в PDF" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Правка">  
 <items>  
 <MenuItem fx:id="MenuItemAdd" mnemonicParsing="false" onAction="#MenuItemAddAction" text="Добавить" />  
 <MenuItem fx:id="MenuItemUpdate" mnemonicParsing="false" onAction="#MenuItemUpdateAction" text="Изменить" />  
 <MenuItem fx:id="MenuItemDelete" mnemonicParsing="false" onAction="#MenuItemDeleteAction" text="Удалить" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Справочники">  
 <items>  
 <MenuItem fx:id="MenuItemPortfolioStrategy" mnemonicParsing="false" onAction="#MenuItemPortfolioStrategyAction" text="Портфолио стратегий" />  
 <MenuItem fx:id="MenuItemPosition" mnemonicParsing="false" onAction="#MenuItemPositionAction" text="Позиции портфолио" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefWidth="120.0" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </Label>  
 </children>  
 </HBox>  
 </top>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Файл программы**

package ru.demo.investmentapp.controller.Portfolio;

import com.itextpdf.text.DocumentException;

import javafx.beans.property.SimpleStringProperty;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.\*;

import ru.demo.investmentapp.model.Portfolio;

import ru.demo.investmentapp.service.PortfolioService;

import ru.demo.investmentapp.util.Manager;

import java.io.FileNotFoundException;

import java.net.URL;

import java.util.List;

import java.util.Optional;

import java.util.ResourceBundle;

import java.util.stream.Collectors;

import static ru.demo.investmentapp.util.Manager.\*;

public class PortfolioTableViewController implements Initializable {

private int itemsCount;

private PortfolioService portfolioService = new PortfolioService();

@FXML

private Button BtnBack;

@FXML

private TableColumn<Portfolio, String> TableColumnId;

@FXML

private TableColumn<Portfolio, String> TableColumnPortfolioName;

@FXML

private TableColumn<Portfolio, String> TableColumnPortfolioType;

@FXML

private TableColumn<Portfolio, String> TableColumnUserName;

@FXML

private Label LabelInfo;

@FXML

private Label LabelUser;

@FXML

private TextField TextFieldSearch;

@FXML

private TableView<Portfolio> TableViewPortfolios;

@FXML

void TextFieldSearchAction(ActionEvent event) {

filterData();

}

@FXML

void MenuItemAddAction(ActionEvent event) {

currentPortfolio = null;

ShowEditWindow("Portfolio/portfolio-edit-view.fxml");

filterData();

}

@FXML

void MenuItemBackAction(ActionEvent event) {

Manager.LoadSecondStageScene("main-view.fxml", "Aurora Growth");

}

@FXML

void MenuItemPrintToPDFAction(ActionEvent event) throws DocumentException, FileNotFoundException {

Portfolio portfolio = TableViewPortfolios.getSelectionModel().getSelectedItem();

if (portfolio != null) {

Manager.PrintPortfolioToPDF(portfolio);

MessageBox("Информация", "", "Данные сохранены успешно", Alert.AlertType.INFORMATION);return;

}

}

@FXML

void MenuItemPortfolioStrategyAction(ActionEvent event) {

Manager.LoadSecondStageScene("PortfolioStrategy/portfolioStrategy-table-view.fxml", "Портфолио стратегий");

}

@FXML

void MenuItemPositionAction(ActionEvent event) {

Manager.LoadSecondStageScene("PortfolioPosition/portfolioPosition-table-view.fxml", "Позиции портфолио");

}

@FXML

void MenuItemDeleteAction(ActionEvent event) {

Portfolio portfolio = TableViewPortfolios.getSelectionModel().getSelectedItem();

Optional<ButtonType> result = ShowConfirmPopup();

if (result.get() == ButtonType.OK) {

portfolioService.delete(portfolio);

filterData();

}

}

@FXML

void MenuItemUpdateAction(ActionEvent event) {

Portfolio portfolio = TableViewPortfolios.getSelectionModel().getSelectedItem();

currentPortfolio = portfolio;

ShowEditWindow("Portfolio/portfolio-edit-view.fxml");

filterData();

}

@Override

public void initialize(URL url, ResourceBundle resourceBundle) {

initController();

}

public void initController() {

LabelUser.setText("Вы вошли как " + currentUser.getClient().getFirstName() + " " +

Manager.currentUser.getClient().getLastName());

List<Portfolio> portfolioList = portfolioService.findAll();

ObservableList<Portfolio> portfolios = FXCollections.observableArrayList(portfolioList);

TableViewPortfolios.setItems(portfolios);

setCellValueFactories();

filterData();

}

void filterData() {

List<Portfolio> portfolios = portfolioService.findAll();

itemsCount = portfolios.size();

String searchText = TextFieldSearch.getText();

if (!searchText.isEmpty()) {

portfolios = portfolios.stream().filter(portfolio -> portfolio.getPortfolioName().toLowerCase().contains(searchText.toLowerCase())

).collect(Collectors.toList());

}

TableViewPortfolios.getItems().clear();

for (Portfolio portfolio: portfolios) {

TableViewPortfolios.getItems().add(portfolio);

}

int filteredItemsCount = portfolios.size();

LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);

TableViewPortfolios.setColumnResizePolicy(TableView.CONSTRAINED\_RESIZE\_POLICY\_FLEX\_LAST\_COLUMN);

}

private void setCellValueFactories() {

TableColumnPortfolioName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getPortfolioName()));

TableColumnPortfolioType.setCellValueFactory(cellData -> new SimpleStringProperty((cellData.getValue().getPortfolioType())));

TableColumnId.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getPortfolioId().toString()));

TableColumnUserName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getUser().getUserName()));

}

}

**Страница Добавления и изменение Клиента**

**Файл интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.\*?>  
<?import javafx.scene.control.\*?>  
<?import javafx.scene.layout.\*?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="500.0"  
 prefWidth="477.0" xmlns="http://javafx.com/javafx/17.0.2-ea" xmlns:fx="http://javafx.com/fxml/1"  
 fx:controller="ru.demo.investmentapp.controller.Client.ClientEditViewController">  
 <children>  
 <BorderPane layoutX="170.0" layoutY="47.0" prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <bottom>  
 <ButtonBar prefHeight="40.0" prefWidth="200.0" BorderPane.alignment="CENTER">  
 <buttons>  
 <Button fx:id="BtnSave" defaultButton="true" mnemonicParsing="false" onAction="#BtnSaveAction" text="Сохранить" styleClass="ok-button"/>  
 <Button fx:id="BtnCancel" cancelButton="true" mnemonicParsing="false" onAction="#BtnCancelAction" text="Отмена" styleClass="cancel-button"/>  
 </buttons>  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </ButtonBar>  
 </bottom>  
 <center>  
 <VBox BorderPane.alignment="CENTER">  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Фамилия" GridPane.rowIndex="0" />  
 <TextField fx:id="TextFieldLastName" GridPane.columnIndex="1" GridPane.rowIndex="0" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Имя" GridPane.rowIndex="1" />  
 <TextField fx:id="TextFieldFirstName" GridPane.columnIndex="1" GridPane.rowIndex="1" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Отчество" GridPane.rowIndex="2" />  
 <TextField fx:id="TextFieldMiddleName" GridPane.columnIndex="1" GridPane.rowIndex="2" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Адрес" GridPane.rowIndex="3" />  
 <TextField fx:id="TextFieldAddress" GridPane.columnIndex="1" GridPane.rowIndex="3" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Email" GridPane.rowIndex="4" />  
 <TextField fx:id="TextFieldEmail" GridPane.columnIndex="1" GridPane.rowIndex="4" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Телефон" GridPane.rowIndex="5" />  
 <TextField fx:id="TextFieldPhone" GridPane.columnIndex="1" GridPane.rowIndex="5" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Тип клиента" GridPane.rowIndex="6" />  
 <TextField fx:id="TextFieldClientType" GridPane.columnIndex="1" GridPane.rowIndex="6" />  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="День рождения" GridPane.rowIndex="7" />  
 <DatePicker fx:id="DatePickerBirthday" prefHeight="25.0" prefWidth="249.0" promptText="Дата рождения" GridPane.columnIndex="1" GridPane.rowIndex="7"/>  
  
 <Label maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" text="Паспорт" GridPane.rowIndex="8" />  
 <TextField fx:id="TextFieldPassportNumber" GridPane.columnIndex="1" GridPane.rowIndex="8" />  
 <BorderPane.margin>  
 <Insets left="5.0" right="5.0" top="20.0" />  
 </BorderPane.margin>  
 </VBox>  
 </center>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Файл программного кода**

package ru.demo.investmentapp.controller.Client;  
  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Button;  
import javafx.scene.control.DatePicker;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
import ru.demo.investmentapp.model.Client;  
import ru.demo.investmentapp.service.ClientService;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.IOException;  
import java.net.URL;  
import java.sql.Date;  
import java.util.ResourceBundle;  
  
import static ru.demo.investmentapp.util.Manager.*MessageBox*;  
  
public class ClientEditViewController implements Initializable {  
 @FXML  
 private Button BtnSave;  
 @FXML  
 private Button BtnCancel;  
 private ClientService clientService = new ClientService();  
 @FXML  
 private TextField TextFieldFirstName;  
 @FXML  
 private TextField TextFieldLastName;  
 @FXML  
 private TextField TextFieldMiddleName;  
 @FXML  
 private TextField TextFieldAddress;  
 @FXML  
 private TextField TextFieldEmail;  
 @FXML  
 private TextField TextFieldPhone;  
 @FXML  
 private TextField TextFieldClientType;  
 @FXML  
 private DatePicker DatePickerBirthday;  
 @FXML  
 private TextField TextFieldPassportNumber;  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 if (Manager.*currentClient* != null) {  
 TextFieldFirstName.setText(Manager.*currentClient*.getFirstName());  
 TextFieldLastName.setText(Manager.*currentClient*.getLastName());  
 TextFieldMiddleName.setText(Manager.*currentClient*.getMiddleName());  
 TextFieldAddress.setText(Manager.*currentClient*.getAddress());  
 TextFieldEmail.setText(Manager.*currentClient*.getEmail());  
 TextFieldPhone.setText(Manager.*currentClient*.getPhone());  
 DatePickerBirthday.setValue(Manager.*currentClient*.getDateOfBirth());  
 TextFieldClientType.setText(Manager.*currentClient*.getClientType());  
 TextFieldPassportNumber.setText(Manager.*currentClient*.getPassportNumber());  
 } else {  
 Manager.*currentClient* = new Client();  
 }  
 }  
  
 @FXML  
 void BtnCancelAction(ActionEvent event) {  
 Stage stage = (Stage) BtnCancel.getScene().getWindow();  
 // do what you have to do  
 stage.close();  
 }  
  
 @FXML  
 void BtnSaveAction(ActionEvent event) throws IOException {  
 String error = checkFields().toString();  
 if (!error.isEmpty()) {  
 *MessageBox*("Ошибка", "Заполните поля", error, Alert.AlertType.*ERROR*);  
 return;  
 }  
 Manager.*currentClient*.setFirstName(TextFieldFirstName.getText());  
 Manager.*currentClient*.setLastName(TextFieldLastName.getText());  
 Manager.*currentClient*.setMiddleName(TextFieldMiddleName.getText());  
 Manager.*currentClient*.setAddress(TextFieldAddress.getText());  
 Manager.*currentClient*.setEmail(TextFieldEmail.getText());  
 Manager.*currentClient*.setPhone(TextFieldPhone.getText());  
 Manager.*currentClient*.setClientType(TextFieldClientType.getText());  
 Manager.*currentClient*.setDateOfBirth(DatePickerBirthday.getValue());  
 Manager.*currentClient*.setPassportNumber(TextFieldPassportNumber.getText());  
 if (Manager.*currentClient*.getClientId() == null) {  
 Manager.*currentClient*.setFirstName(TextFieldFirstName.getText());  
 clientService.save(Manager.*currentClient*);  
 *MessageBox*("Информация", "", "Данные сохранены успешно", Alert.AlertType.*INFORMATION*);  
 } else {  
 clientService.update(Manager.*currentClient*);  
 *MessageBox*("Информация", "", "Данные обновлены успешно", Alert.AlertType.*INFORMATION*);  
 }  
 }  
  
 StringBuilder checkFields() {  
 StringBuilder error = new StringBuilder();  
 if (TextFieldFirstName.getText().isEmpty()) {  
 error.append("Укажите фамилию\n");  
 }  
 if (TextFieldLastName.getText().isEmpty()) {  
 error.append("Укажите имя\n");  
 }  
 if (TextFieldAddress.getText().isEmpty()) {  
 error.append("Укажите адрес\n");  
 }  
 if (TextFieldEmail.getText().isEmpty()) {  
 error.append("Укажите email\n");  
 } else if (!TextFieldEmail.getText().matches("^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$")) {  
 error.append("Неверный формат email\n");  
 }  
 if (DatePickerBirthday.getValue() == null) {  
 error.append("Укажите дату рождения\n");  
 }  
 if (TextFieldEmail.getText().isEmpty()) {  
 error.append("Укажите email\n");  
 }  
 if (TextFieldPhone.getText().isEmpty()) {  
 error.append("Укажите телефон\n");  
 }  
 if (TextFieldPassportNumber.getText().isEmpty()) {  
 error.append("Укажите паспорт\n");  
 }  
 if (TextFieldClientType.getText().isEmpty()) {  
 error.append("Укажите тип клиента\n");  
 }  
 return error;  
 }  
}

**Страница Операции**

**Файл интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.\*?>  
<?import javafx.scene.control.\*?>  
<?import javafx.scene.layout.\*?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="400" minWidth="650" prefHeight="500.0" prefWidth="650.0" xmlns="http://javafx.com/javafx/17.0.2-ea" xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.Transaction.TransactionTableViewController">  
 <children>  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar prefHeight="25.0" prefWidth="381.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemBack" mnemonicParsing="false" onAction="#MenuItemBackAction" text="Назад" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Правка">  
 <items>  
 <MenuItem fx:id="MenuItemAdd" mnemonicParsing="false" onAction="#MenuItemAddAction" text="Добавить" />  
 <MenuItem fx:id="MenuItemUpdate" mnemonicParsing="false" onAction="#MenuItemUpdateAction" text="Изменить" />  
 <MenuItem fx:id="MenuItemDelete" mnemonicParsing="false" onAction="#MenuItemDeleteAction" text="Удалить" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Справочники">  
 <items>  
 <MenuItem fx:id="MenuItemCommission" mnemonicParsing="false" onAction="#MenuItemCommissionAction" text="Комиссия" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefWidth="120.0" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </Label>  
 </children>  
 </HBox>  
 </top>  
 <center>  
 <BorderPane BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane minHeight="-Infinity" nodeOrientation="LEFT\_TO\_RIGHT" prefWidth="200.0" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите тип операции для поиска" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <TableView fx:id="TableViewTransactions" prefHeight="200.0" prefWidth="200.0" tableMenuButtonVisible="true" BorderPane.alignment="CENTER">  
 <columns>  
 <TableColumn id="TableColumnTransactionId" fx:id="TableColumnTransactionId" prefWidth="70.0" resizable="false" text="Id" />  
 <TableColumn fx:id="TableColumnAccountClient" prefWidth="95.0" text="Клиент" />  
 <TableColumn fx:id="TableColumnAccount" prefWidth="95.0" text="Аккаунт" />  
 <TableColumn fx:id="TableColumnInstrument" prefWidth="100.0" text="Тип инструмента" />  
 <TableColumn id="TableColumnTransactionType" fx:id="TableColumnTransactionType" prefWidth="82.0" text="Тип операции" />  
 <TableColumn fx:id="TableColumnAmount" prefWidth="140.0" text="Стоимость" />  
 <TableColumn fx:id="TableColumnDate" prefWidth="150.0" text="Дата" />  
 </columns>  
 </TableView>  
 </center>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Файл программного кода**

package ru.demo.investmentapp.controller.Transaction;

import com.itextpdf.text.DocumentException;

import javafx.beans.property.SimpleStringProperty;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.\*;

import ru.demo.investmentapp.model.Transaction;

import ru.demo.investmentapp.service.TransactionService;

import ru.demo.investmentapp.util.Manager;

import java.io.FileNotFoundException;

import java.net.URL;

import java.util.List;

import java.util.Optional;

import java.util.ResourceBundle;

import java.util.stream.Collectors;

import static ru.demo.investmentapp.util.Manager.\*;

public class TransactionTableViewController implements Initializable {

private int itemsCount;

private TransactionService transactionService = new TransactionService();

@FXML

private Button BtnBack;

@FXML

private TableColumn<Transaction, String> TableColumnTransactionId;

@FXML

private TableColumn<Transaction, String> TableColumnTransactionType;

@FXML

private TableColumn<Transaction, String> TableColumnAmount;

@FXML

private TableColumn<Transaction, String> TableColumnDate;

@FXML

private TableColumn<Transaction, String> TableColumnInstrument;

@FXML

private TableColumn<Transaction, String> TableColumnAccount;

@FXML

private TableColumn<Transaction, String> TableColumnAccountClient;

@FXML

private Label LabelInfo;

@FXML

private Label LabelUser;

@FXML

private TextField TextFieldSearch;

@FXML

private TableView<Transaction> TableViewTransactions;

@FXML

void TextFieldSearchAction(ActionEvent event) {

filterData();

}

@FXML

void MenuItemCommissionAction(ActionEvent event) {

Manager.LoadSecondStageScene("Commission/commission-table-view.fxml", "Комиссии");

}

@FXML

void MenuItemAddAction(ActionEvent event) {

currentTransaction = null;

ShowEditWindow("Transaction/transaction-edit-view.fxml");

filterData();

}

@FXML

void MenuItemBackAction(ActionEvent event) {

Manager.LoadSecondStageScene("main-view.fxml", "Aurora Growth");

}

@FXML

void MenuItemDeleteAction(ActionEvent event) {

Transaction transaction = TableViewTransactions.getSelectionModel().getSelectedItem();

Optional<ButtonType> result = ShowConfirmPopup();

if (result.get() == ButtonType.OK) {

transactionService.delete(transaction);

filterData();

}

}

@FXML

void MenuItemUpdateAction(ActionEvent event) {

Transaction transaction = TableViewTransactions.getSelectionModel().getSelectedItem();

currentTransaction = transaction;

ShowEditWindow("Transaction/transaction-edit-view.fxml");

filterData();

}

@Override

public void initialize(URL url, ResourceBundle resourceBundle) {

initController();

}

public void initController() {

LabelUser.setText("Вы вошли как " + currentUser.getClient().getFirstName() + " " +

Manager.currentUser.getClient().getLastName());

List<Transaction> transactionList = transactionService.findAll();

ObservableList<Transaction> transactions = FXCollections.observableArrayList(transactionList);

TableViewTransactions.setItems(transactions);

setCellValueFactories();

filterData();

}

void filterData() {

List<Transaction> transactions = transactionService.findAll();

itemsCount = transactions.size();

String searchText = TextFieldSearch.getText();

if (!searchText.isEmpty()) {

transactions = transactions.stream().filter(transaction -> transaction.getTransactionType().toLowerCase().contains(searchText.toLowerCase())

).collect(Collectors.toList());

}

TableViewTransactions.getItems().clear();

for (Transaction transaction : transactions) {

TableViewTransactions.getItems().add(transaction);

}

int filteredItemsCount = transactions.size();

LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);

TableViewTransactions.setColumnResizePolicy(TableView.CONSTRAINED\_RESIZE\_POLICY\_FLEX\_LAST\_COLUMN);

}

private void setCellValueFactories() {

TableColumnAccountClient.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getAccount().getClient().getFullName()));

TableColumnAccount.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getAccount().getAccountType()));

TableColumnInstrument.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getInstrument().getInstrumentType().toString()));

TableColumnTransactionType.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getTransactionType()));

TableColumnAmount.setCellValueFactory(cellData -> new SimpleStringProperty((cellData.getValue().getAmount().toString())));

TableColumnTransactionId.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getTransactionId().toString()));

TableColumnDate.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getTransactionDate().toString()));

}

}

**Страница Аккаунты**

**Файл интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.\*?>  
<?import javafx.scene.control.\*?>  
<?import javafx.scene.layout.\*?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="400" minWidth="650" prefHeight="500.0" prefWidth="650.0" xmlns="http://javafx.com/javafx/17.0.2-ea" xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.Account.AccountTableViewController">  
 <children>  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar prefHeight="25.0" prefWidth="381.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemBack" mnemonicParsing="false" onAction="#MenuItemBackAction" text="Назад" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Правка">  
 <items>  
 <MenuItem fx:id="MenuItemAdd" mnemonicParsing="false" onAction="#MenuItemAddAction" text="Добавить" />  
 <MenuItem fx:id="MenuItemUpdate" mnemonicParsing="false" onAction="#MenuItemUpdateAction" text="Изменить" />  
 <MenuItem fx:id="MenuItemDelete" mnemonicParsing="false" onAction="#MenuItemDeleteAction" text="Удалить" />  
 </items>  
 </Menu>  
  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefWidth="120.0" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </Label>  
 </children>  
 </HBox>  
 </top>  
 <center>  
 <BorderPane BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane minHeight="-Infinity" nodeOrientation="LEFT\_TO\_RIGHT" prefWidth="200.0" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите тип аккаунта для поиска" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <TableView fx:id="TableViewAccounts" prefHeight="200.0" prefWidth="200.0" tableMenuButtonVisible="true" BorderPane.alignment="CENTER">  
 <columns>  
 <TableColumn id="TableColumnAccountId" fx:id="TableColumnAccountId" prefWidth="110.0" resizable="false" text="Id" />  
 <TableColumn id="TableColumnClient" fx:id="TableColumnClient" prefWidth="82.0" text="Клиент" />  
 <TableColumn fx:id="TableColumnType" prefWidth="75.0" text="Тип аккаунта" />  
 <TableColumn fx:id="TableColumnBalance" prefWidth="75.0" text="Баланс" />  
 </columns>  
 </TableView>  
 </center>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 </BorderPane>  
 </children>  
</AnchorPane>

</Page>

**Файл программы**

package ru.demo.investmentapp.controller.Account;  
  
import com.itextpdf.text.DocumentException;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.\*;  
import ru.demo.investmentapp.model.Account;  
import ru.demo.investmentapp.service.AccountService;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.FileNotFoundException;  
import java.net.URL;  
import java.util.List;  
import java.util.Optional;  
import java.util.ResourceBundle;  
import java.util.stream.Collectors;  
  
import static ru.demo.investmentapp.util.Manager.\*;  
  
public class AccountTableViewController implements Initializable {  
  
 private int itemsCount;  
 private AccountService accountService = new AccountService();  
 @FXML  
 private Button BtnBack;  
 @FXML  
 private TableColumn<Account, String> TableColumnAccountId;  
 @FXML  
 private TableColumn<Account, String> TableColumnClient;  
 @FXML  
 private TableColumn<Account, String> TableColumnType;  
 @FXML  
 private TableColumn<Account, String> TableColumnBalance;  
 @FXML  
 private Label LabelInfo;  
 @FXML  
 private Label LabelUser;  
 @FXML  
 private TextField TextFieldSearch;  
  
 @FXML  
 private TableView<Account> TableViewAccounts;  
  
 @FXML  
 void TextFieldSearchAction(ActionEvent event) {  
 filterData();  
 }  
  
 @FXML  
 void MenuItemAddAction(ActionEvent event) {  
 *currentAccount* = null;  
 *ShowEditWindow*("Account/account-edit-view.fxml");  
 filterData();  
 }  
  
 @FXML  
 void MenuItemBackAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("main-view.fxml", "Aurora Growth");  
 }  
  
 @FXML  
 void MenuItemDeleteAction(ActionEvent event) {  
 Account account = TableViewAccounts.getSelectionModel().getSelectedItem();  
  
 Optional<ButtonType> result = *ShowConfirmPopup*();  
 if (result.get() == ButtonType.*OK*) {  
 accountService.delete(account);  
 filterData();  
 }  
 }  
  
 @FXML  
 void MenuItemUpdateAction(ActionEvent event) {  
 Account account = TableViewAccounts.getSelectionModel().getSelectedItem();  
 *currentAccount* = account;  
 *ShowEditWindow*("Account/account-edit-view.fxml");  
 filterData();  
 }  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 initController();  
 }  
  
 public void initController() {  
 LabelUser.setText("Вы вошли как " + *currentUser*.getClient().getFirstName() + " " +  
 Manager.*currentUser*.getClient().getLastName());  
  
 List<Account> accountList = accountService.findAll();  
 ObservableList<Account> accounts = FXCollections.*observableArrayList*(accountList);  
 TableViewAccounts.setItems(accounts);  
 setCellValueFactories();  
  
 filterData();  
 }  
  
 void filterData() {  
 List<Account> accounts = accountService.findAll();  
 itemsCount = accounts.size();  
  
 String searchText = TextFieldSearch.getText();  
 if (!searchText.isEmpty()) {  
 accounts = accounts.stream().filter(account -> account.getAccountType().toLowerCase().contains(searchText.toLowerCase())  
 ).collect(Collectors.*toList*());  
 }  
 TableViewAccounts.getItems().clear();  
 for (Account account : accounts) {  
 TableViewAccounts.getItems().add(account);  
 }  
 int filteredItemsCount = accounts.size();  
 LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);  
 TableViewAccounts.setColumnResizePolicy(TableView.*CONSTRAINED\_RESIZE\_POLICY\_FLEX\_LAST\_COLUMN*);  
 }  
  
 private void setCellValueFactories() {  
 TableColumnType.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getAccountType()));  
 TableColumnBalance.setCellValueFactory(cellData -> new SimpleStringProperty((cellData.getValue().getBalance()).toString()));  
 TableColumnAccountId.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getAccountId().toString()));  
 TableColumnClient.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getClient().getFullName().toString()));  
 }  
}

**Страница Инструменты**

**Файл интерфейса**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.geometry.Insets?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.BorderPane?>  
<?import javafx.scene.layout.FlowPane?>  
<?import javafx.scene.layout.HBox?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"  
 prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/22"  
 xmlns:fx="http://javafx.com/fxml/1" fx:controller="ru.demo.investmentapp.controller.Instrument.InstrumentTableViewController">  
 <children>  
 <BorderPane prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <center>  
 <BorderPane BorderPane.alignment="CENTER">  
 <top>  
 <FlowPane nodeOrientation="LEFT\_TO\_RIGHT" rowValignment="TOP" BorderPane.alignment="CENTER">  
 <children>  
 <TextField fx:id="TextFieldSearch" onAction="#TextFieldSearchAction" prefHeight="25.0" prefWidth="262.0" promptText="Введите тип для поиска" />  
 </children>  
 </FlowPane>  
 </top>  
 <center>  
 <TableView fx:id="TableViewInstruments" prefHeight="200.0" prefWidth="200.0" tableMenuButtonVisible="true" BorderPane.alignment="CENTER">  
 <columns>  
 <TableColumn id="TableColumnId" fx:id="TableColumnId" prefWidth="82.0" text="Id" />  
 <TableColumn id="TableColumnType" fx:id="TableColumnType" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Тип" />  
 <TableColumn id="TableColumnTickerSymbol" fx:id="TableColumnTickerSymbol" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Тикер" />  
 <TableColumn id="TableColumnName" fx:id="TableColumnName" editable="false" maxWidth="1.7976931348623157E308" prefWidth="300.0" text="Название" />  
 </columns>  
 </TableView>  
 </center>  
 </BorderPane>  
 </center>  
 <bottom>  
 <Label fx:id="LabelInfo" text="Label" BorderPane.alignment="CENTER\_LEFT" />  
 </bottom>  
 <top>  
 <HBox BorderPane.alignment="CENTER">  
 <children>  
 <MenuBar prefHeight="25.0" prefWidth="381.0" HBox.hgrow="ALWAYS">  
 <menus>  
 <Menu mnemonicParsing="false" text="Файл">  
 <items>  
 <MenuItem fx:id="MenuItemBack" mnemonicParsing="false" onAction="#MenuItemBackAction" text="Назад" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Правка">  
 <items>  
 <MenuItem fx:id="MenuItemAdd" mnemonicParsing="false" onAction="#MenuItemAddAction" text="Добавить" />  
 <MenuItem fx:id="MenuItemUpdate" mnemonicParsing="false" onAction="#MenuItemUpdateAction" text="Изменить" />  
 <MenuItem fx:id="MenuItemDelete" mnemonicParsing="false" onAction="#MenuItemDeleteAction" text="Удалить" />  
 </items>  
 </Menu>  
  
 </menus>  
 </MenuBar>  
 <Label fx:id="LabelUser" alignment="CENTER\_RIGHT" maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefWidth="120.0" styleClass="user-label" text="Label" HBox.hgrow="ALWAYS">  
 <padding>  
 <Insets right="20.0" />  
 </padding>  
 </Label>  
 </children>  
 </HBox>  
 </top>  
 </BorderPane>  
 </children>  
</AnchorPane>

**Файл программы**

package ru.demo.investmentapp.controller.Instrument;  
  
import com.itextpdf.text.DocumentException;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.event.ActionEvent;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.\*;  
import ru.demo.investmentapp.model.Instrument;  
import ru.demo.investmentapp.service.InstrumentService;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.FileNotFoundException;  
import java.net.URL;  
import java.util.List;  
import java.util.Optional;  
import java.util.ResourceBundle;  
import java.util.stream.Collectors;  
  
import static ru.demo.investmentapp.util.Manager.\*;  
  
public class InstrumentTableViewController implements Initializable {  
  
 @FXML  
 private MenuItem MenuItemAdd;  
 @FXML  
 private MenuItem MenuItemBack;  
 @FXML  
 private MenuItem MenuItemDelete;  
 @FXML  
 private MenuItem MenuItemUpdate;  
 @FXML  
 private Label LabelInfo;  
 @FXML  
 private Label LabelUser;  
 @FXML  
 private TableColumn<Instrument, String> TableColumnId;  
 @FXML  
 private TableColumn<Instrument, String> TableColumnTickerSymbol;  
 @FXML  
 private TableColumn<Instrument, String> TableColumnName;  
 @FXML  
 private TableColumn<Instrument, String> TableColumnType;  
 @FXML  
 private TableView<Instrument> TableViewInstruments;  
 @FXML  
 private TextField TextFieldSearch;  
 private int itemsCount;  
 private InstrumentService instrumentService = new InstrumentService();  
  
 @FXML  
 void TextFieldSearchAction(ActionEvent event) {  
 filterData();  
 }  
  
 @Override  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
 initController();  
 }  
  
 public void initController() {  
 LabelUser.setText("Вы вошли как " + *currentUser*.getClient().getFirstName() + " " +  
 Manager.*currentUser*.getClient().getLastName());  
  
 setCellValueFactories();  
  
 filterData();  
 }  
  
 void filterData() {  
 List<Instrument> items = instrumentService.findAll();  
 itemsCount = items.size();  
  
 String searchText = TextFieldSearch.getText();  
 if (!searchText.isEmpty()) {  
 items = items.stream().filter(item -> item.getInstrumentType().toLowerCase().contains(searchText.toLowerCase())).collect(Collectors.*toList*());  
 }  
 TableViewInstruments.getItems().clear();  
 for (Instrument instrument : items) {  
 TableViewInstruments.getItems().add(instrument);  
 }  
  
 int filteredItemsCount = items.size();  
 LabelInfo.setText("Всего записей " + filteredItemsCount + " из " + itemsCount);  
 TableViewInstruments.setColumnResizePolicy(TableView.*CONSTRAINED\_RESIZE\_POLICY\_FLEX\_LAST\_COLUMN*);  
 }  
  
 private void setCellValueFactories() {  
  
 TableColumnId.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getInstrumentId().toString()));  
 TableColumnName.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getName()));  
 TableColumnTickerSymbol.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getTickerSymbol()));  
 TableColumnType.setCellValueFactory(cellData -> new SimpleStringProperty(cellData.getValue().getInstrumentType()));  
 }  
  
 @FXML  
 void MenuItemAddAction(ActionEvent event) {  
 *currentInstrument* = null;  
 *ShowEditWindow*("Instrument/instrument-edit-view.fxml");  
 filterData();  
 }  
  
 @FXML  
 void MenuItemBackAction(ActionEvent event) {  
 Manager.*LoadSecondStageScene*("main-view.fxml", "Aurora Growth");  
 }  
  
 @FXML  
 void MenuItemDeleteAction(ActionEvent event) {  
 Instrument instrument = TableViewInstruments.getSelectionModel().getSelectedItem();  
  
 Optional<ButtonType> result = *ShowConfirmPopup*();  
 if (result.get() == ButtonType.*OK*) {  
 instrumentService.delete(instrument);  
 filterData();  
 }  
 }  
  
 @FXML  
 void MenuItemUpdateAction(ActionEvent event) {  
 Instrument instrument = TableViewInstruments.getSelectionModel().getSelectedItem();  
 *currentInstrument* = instrument;  
 *ShowEditWindow*("Instrument/instrument-edit-view.fxml");  
 filterData();  
 }  
}

**Investment.app**

**Файл программы**

package ru.demo.investmentapp;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.scene.image.Image;  
import javafx.stage.Stage;  
import ru.demo.investmentapp.model.User;  
import ru.demo.investmentapp.util.Manager;  
  
import java.io.IOException;  
  
public class InvestmentApp extends Application {  
  
 public User currentUser;  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
  
 @Override  
 public void start(Stage stage) throws IOException {  
  
 stage.getIcons().add(new Image(InvestmentApp.class.getResourceAsStream("investment.png")));  
 stage.setScene(getNewScene());  
 stage.setTitle("Авторизация!");  
 stage.setResizable(false);  
 stage.setOnCloseRequest(event -> {  
 Manager.*ShowPopup*();  
 });  
 stage.setOnShown(windowEvent -> {  
 try {  
 stage.setScene(getNewScene());  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 });  
  
 Manager.*mainStage* = stage;  
 stage.show();  
 }  
  
 Scene getNewScene() throws IOException {  
 FXMLLoader fxmlLoader = new FXMLLoader(InvestmentApp.class.getResource("login-view.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 scene.getStylesheets().add("base-styles.css");  
 return scene;  
 }  
}

**Файл MakeCaptcha**

**Файл программы**

package ru.demo.investmentapp.util;  
  
import javafx.embed.swing.SwingFXUtils;  
import javafx.scene.image.Image;  
import java.awt.\*;  
import java.awt.image.BufferedImage;  
import java.io.IOException;  
import java.util.Random;  
  
public class MakeCaptcha {  
 private static String *captchaCode*;  
 public static Image CreateImage(int width, int height, int symbolCount) throws IOException {  
 Random rnd = new Random();  
 BufferedImage result = new BufferedImage(width, height, BufferedImage.*TYPE\_INT\_RGB*);  
 Graphics2D g2d = result.createGraphics();  
 g2d.setColor(Color.*lightGray*);  
 g2d.fillRect(0, 0, width, height);  
 char symbol;   
 String alphabet = "23456789QWERTYUPASDFGHJKLZXCVBNM";  
 String captcha = "";  
 int h = width / symbolCount;  
 for (int i = 0; i < symbolCount; ++i) {   
 int size = rnd.nextInt(20) + 20;  
 g2d.setFont(new Font("Arial", Font.*BOLD*, size));   
 symbol = alphabet.charAt(rnd.nextInt(alphabet.length()));  
 g2d.setPaint(*getRandomColor*());  
 captcha += symbol;   
 int Xpos = rnd.nextInt(h / 2) + h \* i;  
 int Ypos = rnd.nextInt(height / 2) + 20;  
 g2d.drawString(String.*valueOf*(symbol), Xpos, Ypos);  
 }   
 for (int i = 0; i < 10; ++i) {  
 g2d.setPaint(*getRandomColor*());  
 int x1 = rnd.nextInt(width / 10);  
 int y1 = rnd.nextInt(height);  
 int x2 = rnd.nextInt(width);  
 int y2 = rnd.nextInt(height);  
 g2d.drawLine(x1, y1, x2, y2);  
 }  
 for (int i = 0; i < 100; ++i) {  
 g2d.setPaint(*getRandomColor*());  
 int x1 = rnd.nextInt(width);  
 int y1 = rnd.nextInt(height);  
 g2d.draw(new Rectangle(x1, y1, 1, 1));  
 }  
 *captchaCode* = captcha;   
 g2d.dispose();  
 return SwingFXUtils.*toFXImage*(result, null);  
 }  
 public static String captchaCode() {  
 return *captchaCode*;  
 }  
 public static Color getRandomColor() {  
 Random rnd = new Random();  
 int red = rnd.nextInt(256);  
 int green = rnd.nextInt(256);  
 int blue = rnd.nextInt(256);  
 return new Color(red, green, blue);  
 }  
}

**Приложение B**

**Диск с исполняемым кодом программного модуля**