Минский колледж предпринимательства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**КАЛЬКУЛЯТОР РАСХОДОВ**

**Пояснительная записка**

**МКП.3200611–01 81 01**

*technicaldocs.ru*

*Подпись и дата*

*Взам. инв. №*

*Инв. № дубл.*

*Инв. № подл.*

*Подпись и дата*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | Учащийся |
|  |  |  |
|  |  | А.А.Сарафанов |
|  |  | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  |  |  |
|  |  | Руководитель |
|  |  |  |
|  |  | А.С.Шаповалов |
|  |  | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  |  |  |
|  |  |  |
|  |  |  |

2023

# АННОТАЦИЯ

Курсовой проект на тему «Калькулятор расходов» состоит из программного средства, пояснительной записки.

Программное средство предназначено для отслеживания своих расходов.

Программное средство разработано в среде программирования Visual Studio с использованием языка программирования C#.

Пояснительная записка выполнена на 77 листах, содержит 7 разделов и одно приложение.

# СОДЕРЖАНИЕ

[АННОТАЦИЯ 2](#_Toc152343958)

[СОДЕРЖАНИЕ 3](#_Toc152343959)

[ВВЕДЕНИЕ 5](#_Toc152343960)

[1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 7](#_Toc152343961)

[2. СТРУКТУРА БАЗЫ ДАННЫХ ПРИЛОЖЕНИЯ 8](#_Toc152343962)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 9](#_Toc152343963)

[3.1. Постановка задачи 9](#_Toc152343964)

[3.2. Описание организации данных 9](#_Toc152343965)

[3.3. Описание программных средств 9](#_Toc152343966)

[3.4. Проектирование интерфейса приложения 11](#_Toc152343967)

[3.5. Описание логической структуры программы 14](#_Toc152343968)

[4. СПЕЦИФИКАЦИЯ И АНАЛИЗ ПРОГРАММНОГО СРЕДСТВА 15](#_Toc152343969)

[4.1. Определения, акронимы, сокращения 15](#_Toc152343970)

[4.2. Краткий обзор 15](#_Toc152343971)

[4.3. Общее описание 15](#_Toc152343972)

[4.4. Перспектива изделия 15](#_Toc152343973)

[4.5. Функции изделия 15](#_Toc152343974)

[4.6. Характеристики пользователей 16](#_Toc152343975)

[4.7. Интерфейсы пользователя 16](#_Toc152343976)

[5. ДИАГРАММЫ 17](#_Toc152343977)

[5.1. IDEF0 диаграмма 17](#_Toc152343978)

[5.2. DFD диаграмма 21](#_Toc152343979)

[5.3. Модель бизнес–процессов 24](#_Toc152343980)

[5.4. Модель «сущность–связь» в разных нотациях 27](#_Toc152343981)

[5.5. Диаграмма прецедентов 29](#_Toc152343982)

[5.6. Диаграммы деятельности и состояний 30](#_Toc152343983)

[6. СХЕМА ДИАЛОГА ПОЛЬЗОВАТЕЛЯ С РАЗРАБОТАННЫМ ПРИЛОЖЕНИЕМ 34](#_Toc152343984)

[6.1. Схема диалога пользователя с разработанным приложением 34](#_Toc152343985)

[7. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 38](#_Toc152343986)

[ЗАКЛЮЧЕНИЕ 55](#_Toc152343987)

[ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 56](#_Toc152343988)

[ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ 57](#_Toc152343989)

[ПРИЛОЖЕНИЕ 1 58](#_Toc152343990)

# ВВЕДЕНИЕ

В современном обществе эффективное управление финансами играет важную роль в повседневной жизни людей. Контроль расходов и умение планировать бюджет становятся неотъемлемыми навыками для достижения финансовой стабильности. В этом контексте разработка электронного приложения, такого как калькулятор расходов, становится актуальной и полезной задачей.

Целью данного курсового проекта является разработка калькулятора расходов, который предоставит пользователям инструмент для удобного отслеживания и управления своими финансами. Благодаря такому приложению пользователи смогут вести учет доходов и расходов, анализировать свои траты, определять приоритетные области затрат и планировать свой бюджет на основе полученных данных.

В ходе разработки калькулятора расходов будет уделено особое внимание созданию удобного и интуитивно понятного пользовательского интерфейса. Пользователи смогут вносить информацию о своих доходах и расходах, категоризировать траты, просматривать статистику о своих финансовых операциях.

Предлагаемое приложение по калькулятору расходов будет полезным инструментом для всех, кто стремится улучшить свою финансовую грамотность и достичь финансовой стабильности. Оно поможет пользователям принимать осознанные финансовые решения, контролировать свои расходы и достигать поставленных финансовых целей.

Целью выполнения курсового проекта является расширение, углубление знаний в области разработки на языке C#, формирование навыков научно–исследовательской деятельности.

Задачей выполнения курсового проекта является разработка калькулятора расходов.

Цели курсового проектирования:

− закрепить, углубить и систематизировать теоретические знания и практические навыки, полученные в ходе изучения учебной дисциплины;

− разработка удобного интерфейса пользователя, системы меню (навигации), системы помощи.

Пояснительная записка содержит следующие разделы:

1) «Назначение и область применения» описывается назначение программы, краткая характеристика области применения программы. Проводится анализ предметной области и инфологическое проектирование

разрабатываемой базы данных. Приводится обзор существующих подходов к

решению задачи, их достоинства и недостатки, отличие разработанной программы от аналогов. Рассматриваются существующие аналоги.

2) «Структура базы данных» описывается структура БД её связи, таблицы и их поля.

3) «Технические характеристики» – рассматриваются следующие вопросы: формирование требований к системе в целом; требования к функциям (раздел «Постановка задачи»); требования к организации данных в проекте, описание их организации, структуры, построение концептуальной и логической модели данных, способа кодирования (при необходимости) и предварительной подготовки (раздел «Описание организации данных»); обоснование выбора программного обеспечения, используемого для разработки программного средства (раздел «Описание программных средств»); проектирование программного средства. (разделы «Описание физической структуры базы данных», «Проектирование интерфейса», «Описание логической структуры программы»).

4) «Спецификация и анализ программного средства». Описываются: определения, акронимы, сокращения, публикация, краткий обзор, общее описание, перспектива изделия, функции изделия, характеристики пользователей, требования к внешним интерфейсам и интерфейсы пользователя.

5) «Диаграммы». Описываются и предоставлены для просмотра: IDEF0 диаграмма, DFD–диаграмма, модель бизнес–процессов, модель «сущность–связь» в разных нотациях, диаграмма прецедентов, диаграммы деятельности и состояний.

6) «Схема диалога пользователя с разработанным приложением» – описывается последовательность действий пользователя, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведены описание функций, формата и возможных вариантов команд, с помощью которых пользователь управляет выполнением программы, а также ответы программы на эти команды. Так же схема диалога пользователя с разработанным приложением и схема диалога администратора с разработанным приложением.

7) «Тестирование и анализ полученных результатов». Описывается проверка работы программы на различных тестах и приводится анализ ожидаемых и полученных результатов, выводы по результатам тестирования.

# 1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

В данном курсовом проекте необходимо разработать калькулятор расходов. Данный продукт поможет пользователю отслеживать свои расходы.

Это программное средство предоставляет пользователю возможность удобно вводить и отслеживать свои расходы в разных категориях, таких как питание, жилье, транспорт, развлечения и другие. Оно также может предоставлять статистику о расходах, помогая пользователю более ясно представить свои финансовые привычки и выявить области, в которых можно сэкономить или оптимизировать расходы.

Калькулятор расходов отличается от традиционных методов учета расходов, таких как использование таблиц или записей в блокноте, тем, что он предоставляет более удобный и автоматизированный способ ввода, отслеживания и анализа данных о расходах.

Данное программное средство рассчитано на людей возраста от 8 лет.

Требования к аппаратным и операционным ресурсам для разработки:

1. Операционная система Windows 11.
2. Оперативная память 8GB.
3. SSD 256GB.

# 2. СТРУКТУРА БАЗЫ ДАННЫХ ПРИЛОЖЕНИЯ

Файл (таблица) представляет собой набор данных о том, или ином предмете или объекте. Данные в таблице (файле) хранятся в виде столбцов (полей) и строк (записей). Все данные в таблице должны относиться к объектам одного типа и только к ним.

Записью является набор сведений о человеке, предмете или событии. Каждая запись в таблице содержит один и тот же набор полей и каждое поле одного и того же рода сведения о предмете.

Таблица 2.1 – Пользователь

|  |  |  |
| --- | --- | --- |
| Поля: | Тип данных | Содержится |
| Id пользователя | Счетчик | Индекс пользователя |
| Логин | Короткий текст | Логин пользователя |
| Пароль | Короткий текст | Пароль пользователя |
| Баланс в русских рублях | Финансовый | Баланс пользователя |
| Баланс в белорусских рублях | Финансовый | Баланс пользователя |
| Баланс в долларах | Финансовый | Баланс пользователя |
| Пинкод | Числовой | Пинкод пользователя |

Таблица 2.2 – Категории

|  |  |  |
| --- | --- | --- |
| Поля: | Тип данных | Содержится |
| Id вида транспорта | Счетчик | Индекс вида транспорта |
| Id пользователя | Короткий текст | Логин пользователя |
| Название категории | Короткий текст | Название категории |

Таблица 2.3 – Записи

|  |  |  |
| --- | --- | --- |
| Поля: | Тип данных | Содержится |
| Id транспорта | Счетчик | Индекс транспорта |
| Id категории | Счетчик | Индекс категории |
| Текст записи | Короткий текст | Текст записи |
| Баланс в русских рублях | Финансовый | Баланс пользователя |
| Баланс в белорусских рублях | Финансовый | Баланс пользователя |
| Баланс в долларах | Финансовый | Баланс пользователя |

# 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Понятие «технические характеристики приложения» включает в себя совокупность технических (физических, химических, механических, органолептических и т. д.) характеристик приложения, позволяющих отличить одно приложение от другого.

## 3.1. Постановка задачи

Разработать калькулятор расходов. Продукт предназначен для отслеживания своих финансов. Суть калькулятора расходов заключается в просмотре информации о расходах.

В приложении реализовать:

* + - Создание категорий
    - Создание записей;
    - Удаление категории.
    - Пополнение баланса.
    - Выбор валюты

Необходимо реализовать просмотр статистики за определенный период времени.

При выборе категории из определенного списка, пользователю предоставляется страница, на которой есть список записей. Поле для ввода названия и цены записи, кнопка удаления категории и создание записей.

## 3.2. Описание организации данных

Входными данными для разрабатываемого ПП является:

* данные, вводимые пользователем (данные для авторизации, названия категорий, данные записей, баланс);
* данные, хранящиеся в БД системы;
* значения, используемые по умолчанию: параметры поиска, структура таблиц, переменные связи между таблицами.

Выходными данными разрабатываемого ПП является:

* информация, сохраняемая в БД;
* информация, выдаваемая пользователю по запросу.

## 3.3. Описание программных средств

C# широко используется для разработки различных типов программ, включая настольные приложения, веб-приложения, мобильные приложения и игры. Он является основным языком разработки на платформе .NET и может быть использован совместно с другими языками, такими как Visual Basic .NET и F#.

Visual Studio WinForms (Windows Forms) - это один из инструментов разработки, предоставляемых Microsoft Visual Studio, который позволяет создавать настольные приложения для операционной системы Windows с использованием технологии Windows Forms.

Windows Forms является частью .NET Framework и предоставляет программистам набор классов и компонентов для создания графического интерфейса пользователя (GUI) в приложениях Windows. Он основан на модели событий и обработчиков событий, что позволяет разработчикам реагировать на пользовательские действия, такие как щелчки мыши, нажатия клавиш и другие события взаимодействия с пользователем.

Visual Studio WinForms обеспечивает удобную среду разработки для создания приложений на основе Windows Forms. В ней доступны инструменты для визуального проектирования пользовательского интерфейса, редактирования кода, отладки и сборки приложений. Разработчики могут создавать формы, добавлять на них элементы управления (например, кнопки, текстовые поля, таблицы и другие), настраивать их свойства и определять логику взаимодействия с помощью языка программирования C# или Visual Basic .NET.

Visual Studio WinForms также предлагает широкий набор инструментов и компонентов для упрощения разработки приложений, включая возможности для работы с базами данных, поддержку многопоточности, создания пользовательских элементов управления и многое другое.

С использованием Visual Studio WinForms разработчики могут создавать разнообразные настольные приложения для Windows, включая утилиты, инструменты администрирования, клиентские приложения и другие программы, которые используются для работы на компьютере пользователя.

Хотя Visual Studio является мощным инструментом разработки и широко используется в индустрии программного обеспечения, у него также есть некоторые потенциальные минусы:

1) Ресурсоемкость: Visual Studio является мощным интегрированной средой разработки (IDE) и требует значительных ресурсов компьютера. Он может потреблять большое количество оперативной памяти и процессорного времени, особенно при работе с большими проектами. Это может привести к замедлению работы компьютера и увеличению времени компиляции и запуска проектов.

2) Сложность и избыточность: Visual Studio предлагает множество функций и возможностей, и для новичков он может показаться сложным и перегруженным. Интерфейс IDE может быть сложным для освоения, и многие функции могут оказаться избыточными для некоторых простых проектов. Это может привести к излишней сложности и затратам времени на изучение и использование определенных функций.

3) Зависимость от платформы: Visual Studio является продуктом Microsoft и прежде всего ориентирован на разработку приложений для платформы Windows. Если вам нужно разрабатывать кроссплатформенные приложения, например, для macOS или Linux, то Visual Studio может иметь ограниченные возможности и интеграцию с другими платформами.

## 3.4. Проектирование интерфейса приложения

Данное приложение будет состоять из 4 форм: «Форма регистрации и авторизации», «Форма главного меню», «Восстановления пароля».

Макет формы авторизации представлен на рисунке 3.1., а регистрации на рисунке 3.2. На данных формах пользователь может зарегистрироваться и авторизоваться.

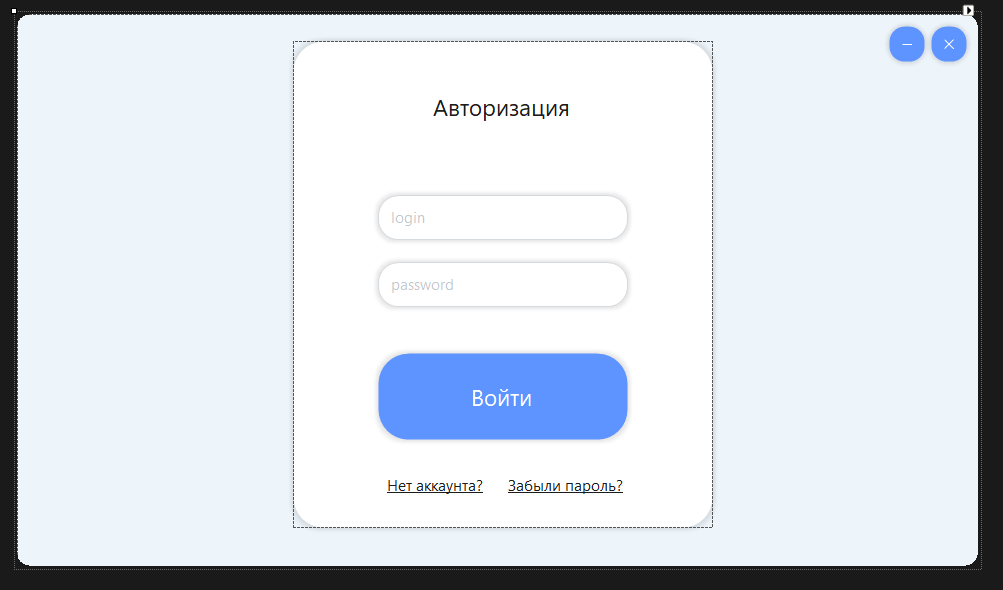


Рис. 3.1. Форма авторизации

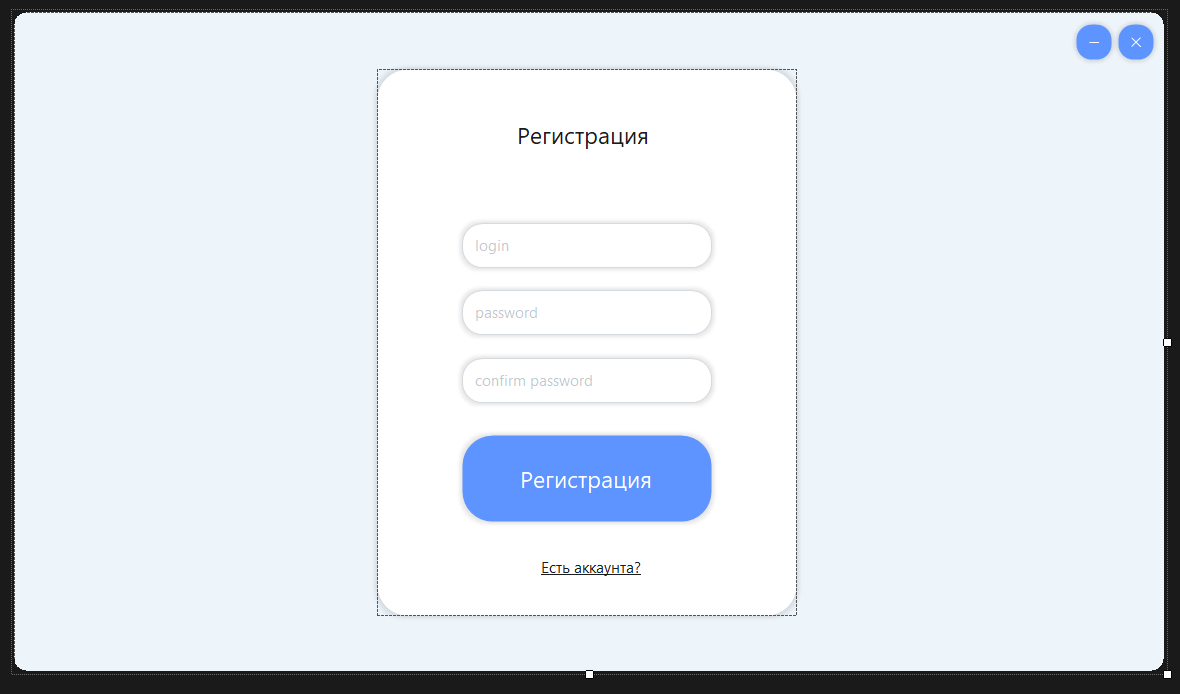


Рис. 3.2. Форма регистрации

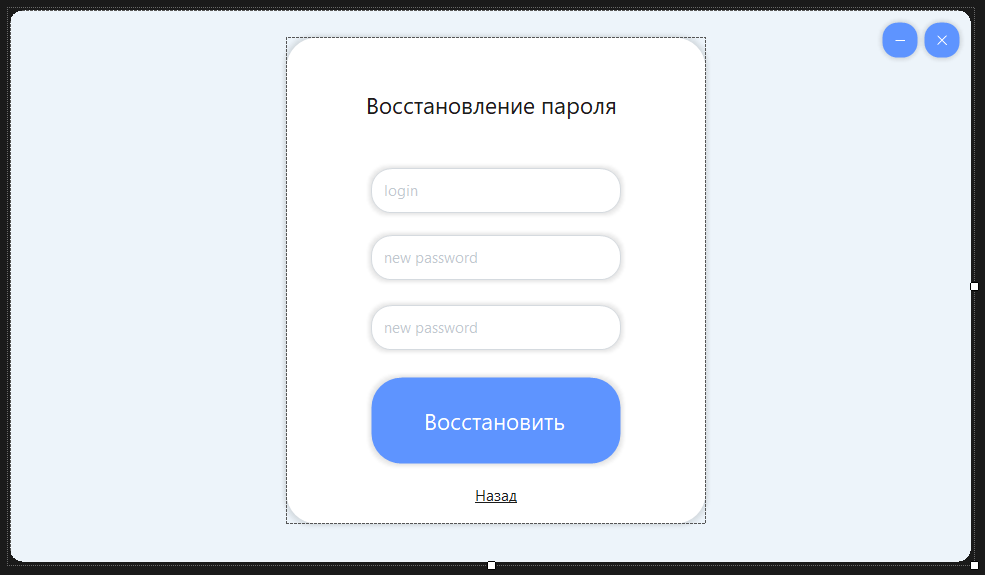


Рис. 3.3. Форма восстановления пароля

Таблица 3.1. – Элементы управления формы регистрации, авторизации и восстановления пароля

|  |  |  |
| --- | --- | --- |
| Элементы управления формы регистрации и авторизации | Свойства | Назначение |
| Label | Name = Label1 | Краткое название формы |
| Button | Name = btnLogin | Кнопка авторизации |

*Продолжение таблицы 3.1.*

|  |  |  |
| --- | --- | --- |
| Элементы управления формы регистрации и авторизации | Свойства | Назначение |
| Button | Name = btnReg | Кнопка регистрации |
| Label | Name = Label2-3 | Переход на другую форму |
| TextBox | Name = txtLogin | Поле для логина |
| TextBox | Name = txtPassword1–2 | Поля для пароля |
| Button | Name = btnExit | Кнопка выхода из приложения |
| Button | Name = btnMinimize | Кнопка свернуть приложения |

Макет формы главного меню представлен на рисунке 3.3.

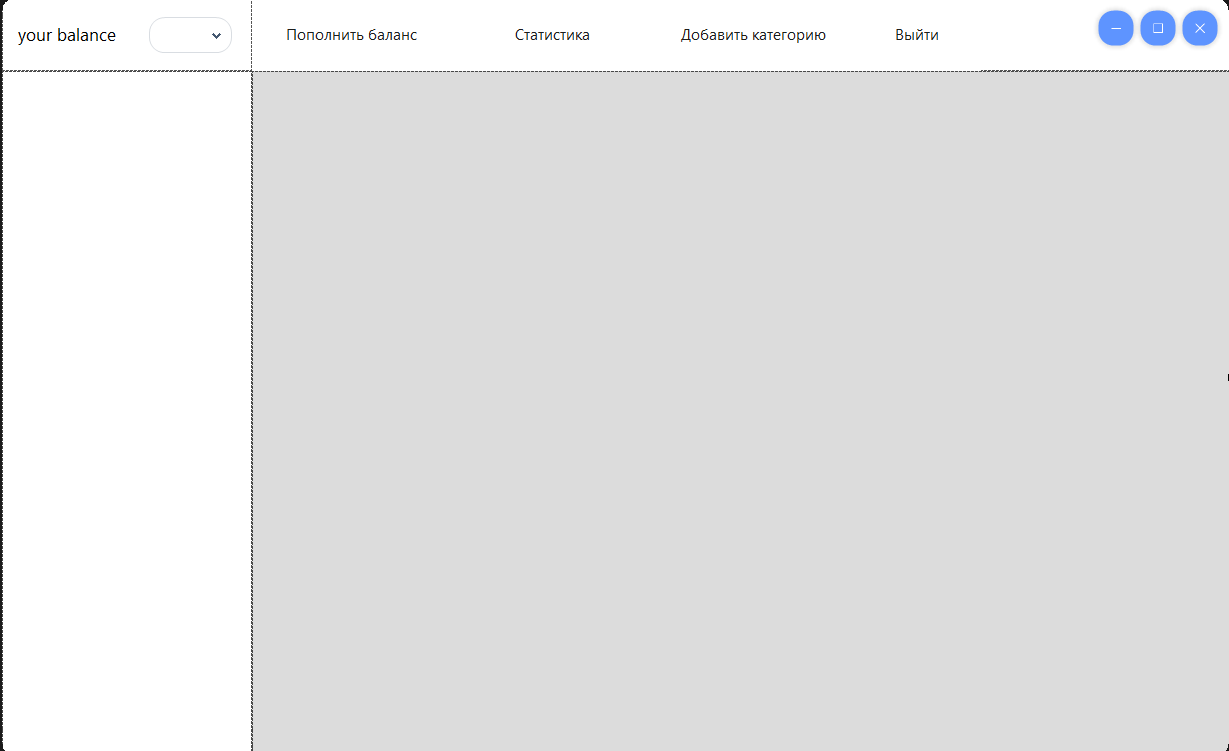


Рис. 3.3. Форма главного меню

Таблица 3.2 – Элементы управления формы главного меню

|  |  |  |
| --- | --- | --- |
| Элементы управления формы регистрации и авторизации | Свойства | Назначение |
| Button | Name = Btn1-4 | Кнопки элементов управления |
| Label | Name = lblBalance | Баланс |
| ComboBox | Name = comboBox1 | Выбор валюта |
| Panel | Name = pnl1-2 | Размещение элементов управления |

## 3.5. Описание логической структуры программы

На форме авторизации и регистрации размещены следующие функции:

* private void btnLogin\_Click (object sender, EventArgs e) — Эта функция отвечает за регистрацию пользователя в приложение­­­­­.
* private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e) — Эта функция отвечает за переход на другую форму.
* private void RegistrationForm\_FormClosing(object sender, FormClosingEventArgs e) — Эта функция отвечает за закрытие программы.
* private void btnLogin\_Click(object sender, EventArgs e) — Эта функция отвечает за вход пользователя.
* private void linkLabel1\_LinkClicked (object sender, LinkLabelLinkClickedEventArgs e) — Эта функция отвечает за переход на другую форму.

На форме главного меню размещены следующие функции:

* private void siticoneComboBox1\_SelectedIndexChanged (object sender, EventArgs e) — Эта функция отвечает за смену валюты.
* private void btnExit\_Click(object sender, EventArgs e) — Эта функция отвечает за смену формы.

На форме восстановления пароля размещены следующие функции:

* private void btnLogin\_Click(object sender, EventArgs e) ) — Эта функция отвечает за смену пароля.
* private void linkLabel1\_LinkClicked\_1(object sender, LinkLabelLinkClickedEventArgs e) — Эта функция отвечает за смену формы.

# 4. СПЕЦИФИКАЦИЯ И АНАЛИЗ ПРОГРАММНОГО СРЕДСТВА

Спецификация требований программного обеспечения (software requirements specification, SRS) – структурированный набор требований/запросов (функциональность, производительность, конструктивные ограничения и атрибуты) к программному обеспечению и его внешним интерфейсам. (Определение на основе IEEE Std 1012:2004). Предназначен для того, чтобы установить базу для соглашения между заказчиком и разработчиком (или подрядчиками) о том, как должен функционировать программный продукт.

## 4.1. Определения, акронимы, сокращения

Пользователь – лицо или организация, использующее действующую систему для выполнения конкретной функции.

## 4.2. Краткий обзор

Проект предназначен для изучения, отслеживания бюджета.

## 4.3. Общее описание

Основные задачи системы:

* сохранение информации о пользователе;
* отображение заданного запроса пользователем;
* возможность добавления информации в БД.

## 4.4. Перспектива изделия

Разрабатываемое приложение является десктопным, т. е. предназначено для загрузки или работы локально, имеющими расширенный доступ к базам данных.

Функционирование приложения предусмотрено на русском языке.

## 4.5. Функции изделия

Назначение приложения:

* сбор информации о расходах;
* сохранение информации о пользователе;
* отображения статистики;

## 4.6. Характеристики пользователей

Приложение предназначено для пользователей, желающих узнать о своих расходах.

## 4.7. Интерфейсы пользователя

Интерфейс должен быть интуитивно понятен для каждого пользователя приложения.

# 5. ДИАГРАММЫ

Диаграмма ([греч.](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B5%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Διάγραμμα (diagramma) изображение) – графическое представление данных линейными отрезками или геометрическими фигурами, позволяющее быстро оценить соотношение нескольких величин. Представляет собой геометрическое [символьное](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB) изображение [информации](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D1%8F) с применением различных приёмов техники [визуализации](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BD%D0%B4%D0%B5%D1%80%D0%B8%D0%BD%D0%B3).

Иногда для оформления диаграмм используется трёхмерная визуализация, спроецированная на [плоскость](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%BE%D1%81%D0%BA%D0%BE%D1%81%D1%82%D1%8C_(%D0%B3%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F)), что придаёт диаграмме отличительные черты или позволяет иметь общее представление об области, в которой она применяется. Например: финансовая диаграмма, связанная с денежными суммами, может представлять собой количество [купюр](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%BD%D0%BA%D0%BD%D0%BE%D1%82%D0%B0) в пачке или [монет](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%BD%D0%B5%D1%82%D0%B0) в стопке; диаграмма сравнения количества подвижного состава – различную длину изображённых [поездов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B5%D0%B7%D0%B4) и т. д. Благодаря своей наглядности и удобству использования, диаграммы часто использую тся не только в повседневной работе [бухгалтеров](https://ru.wikipedia.org/wiki/%D0%91%D1%83%D1%85%D0%B3%D0%B0%D0%BB%D1%82%D0%B5%D1%80), [логистов](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) и других служащих, но и при подготовке материалов презентаций для [клиентов](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BA%D0%B0%D0%B7%D1%87%D0%B8%D0%BA) и [менеджеров](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%BD%D0%B5%D0%B4%D0%B6%D0%B5%D1%80) различных организаций.

## 5.1. IDEF0 диаграмма

IDEF0 – методология функционального моделирования (англ. function modeling) и графическая нотация, предназначенная для формализации и описания бизнес–процессов. Отличительной особенностью IDEF0 является её акцент на соподчиненность объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность (поток работ).

Стандарт IDEF0 представляет организацию как набор модулей, здесь существует правило – наиболее важная функция находится в верхнем левом углу, кроме того, существуют правила сторон:

1. Стрелка входа всегда приходит в левую кромку активности.
2. Стрелка управления – в верхнюю кромку.
3. Стрелка механизма – нижняя кромка.
4. Стрелка выхода – правая кромка.

С точки зрения пользователя, предоставлено на рис. 5.1.1 – 5.1.7.

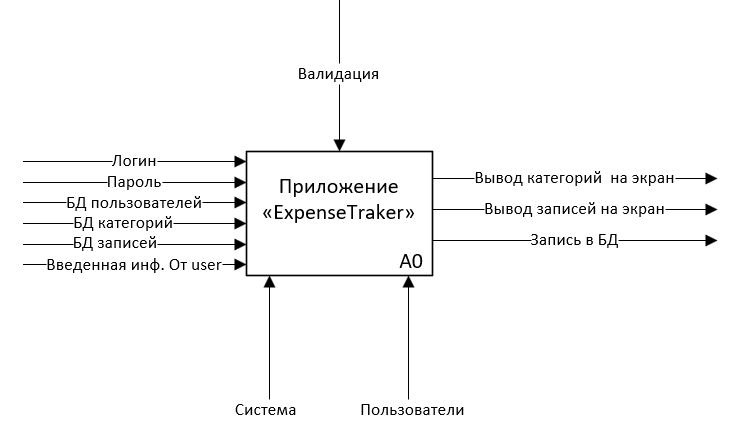


Рис. 5.1.1. IDEF0 диаграмма (часть 1)

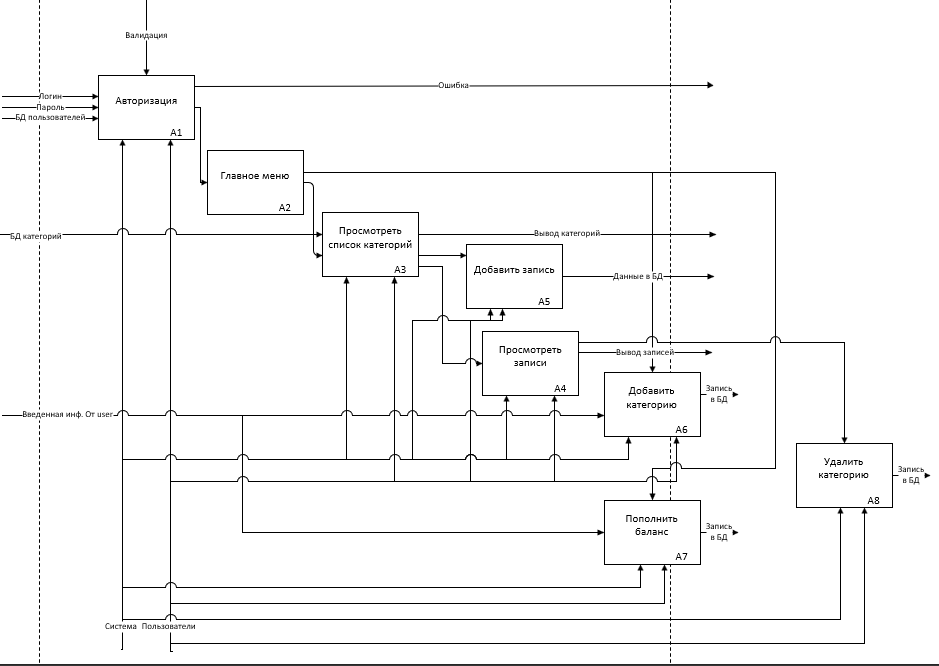


Рис. 5.1.2. IDEF0 диаграмма (часть 2)

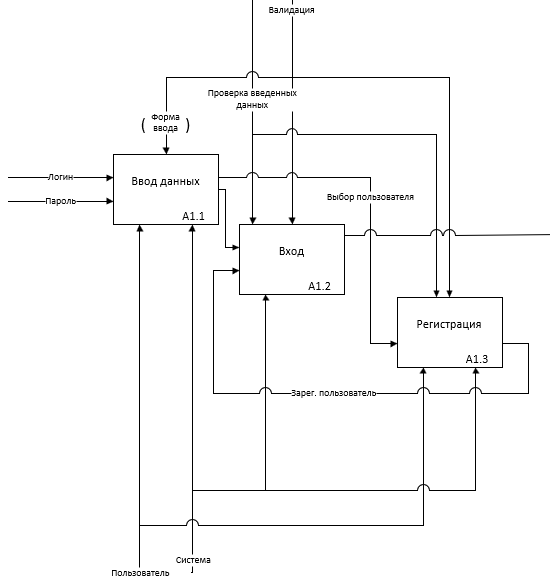


Рис. 5.1.3. IDEF0 диаграмма (часть 3)

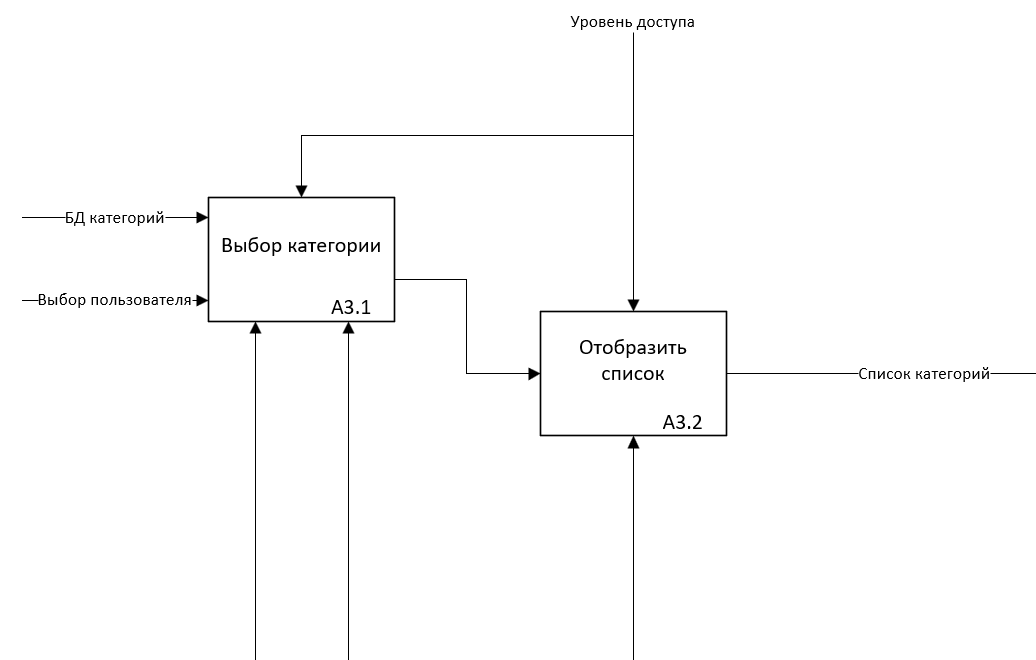


Рис. 5.1.4. IDEF0 диаграмма (часть 4)

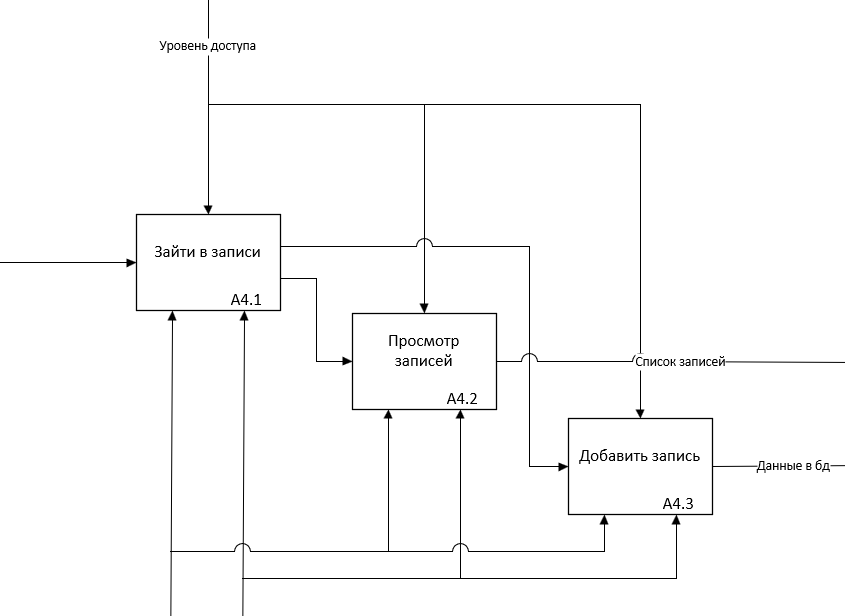


Рис. 5.1.5. IDEF0 диаграмма (часть 5)

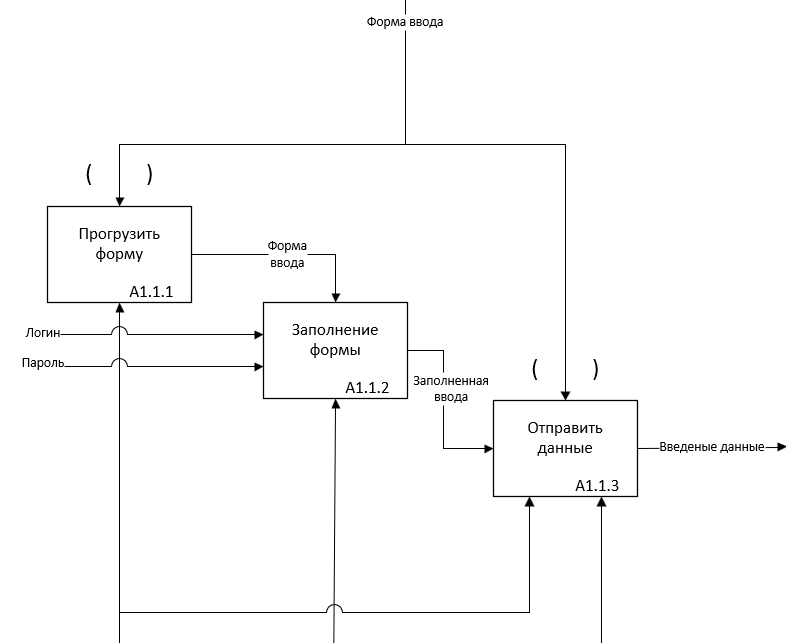


Рис. 5.1.6. IDEF0 диаграмма (часть 6)

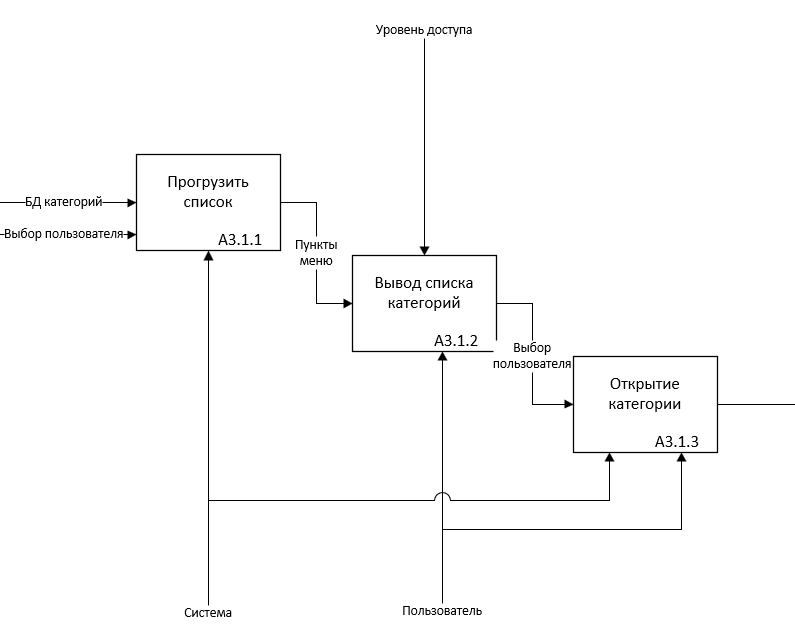


Рис. 5.1.7. IDEF0 диаграмма (часть 7)

## 5.2. DFD диаграмма

DFD – общепринятое сокращение от англ. data flow diagrams – диаграммы потоков данных. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе, источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ. Предоставлено на рисунках 5.2.1 – 5.2.6.

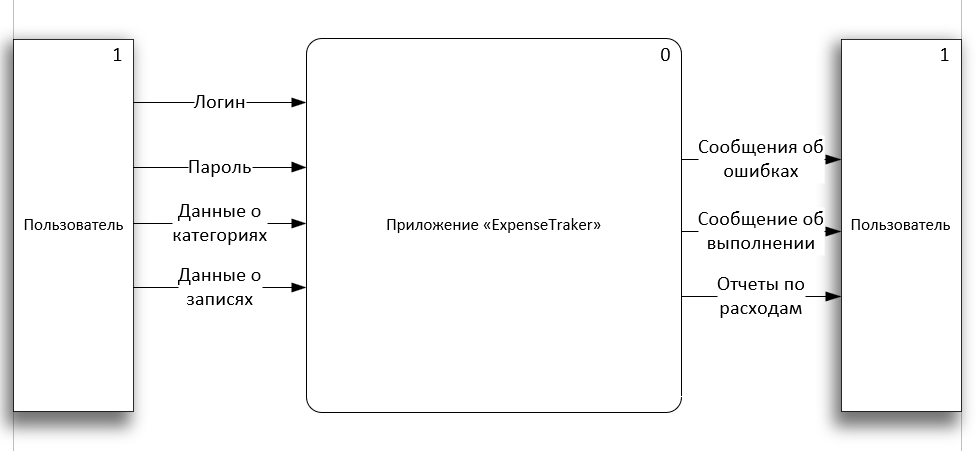


Рис. 5.2.1. DFD–диаграмма (часть 1)

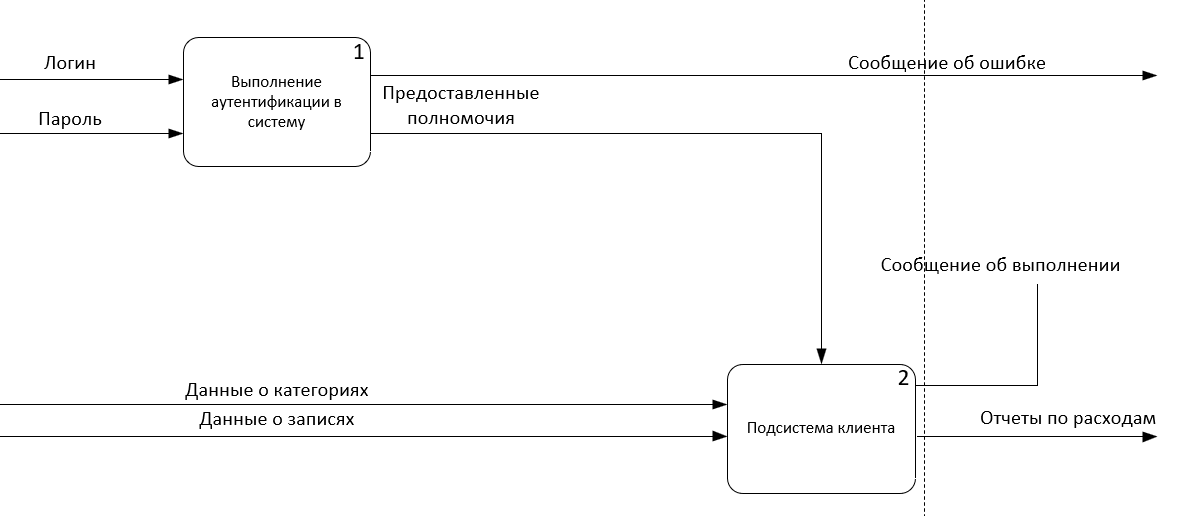


Рис. 5.2.2. DFD–диаграмма (часть 2)

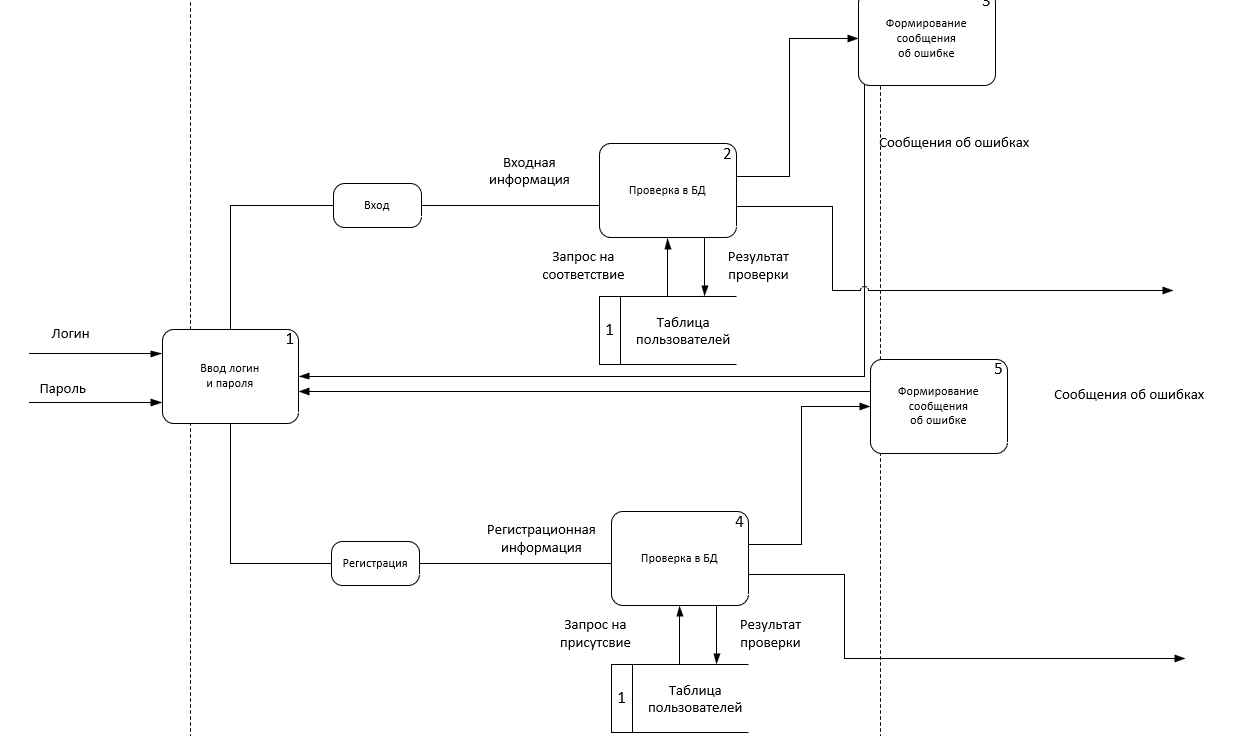


Рис. 5.2.3. DFD–диаграмма (часть 3)

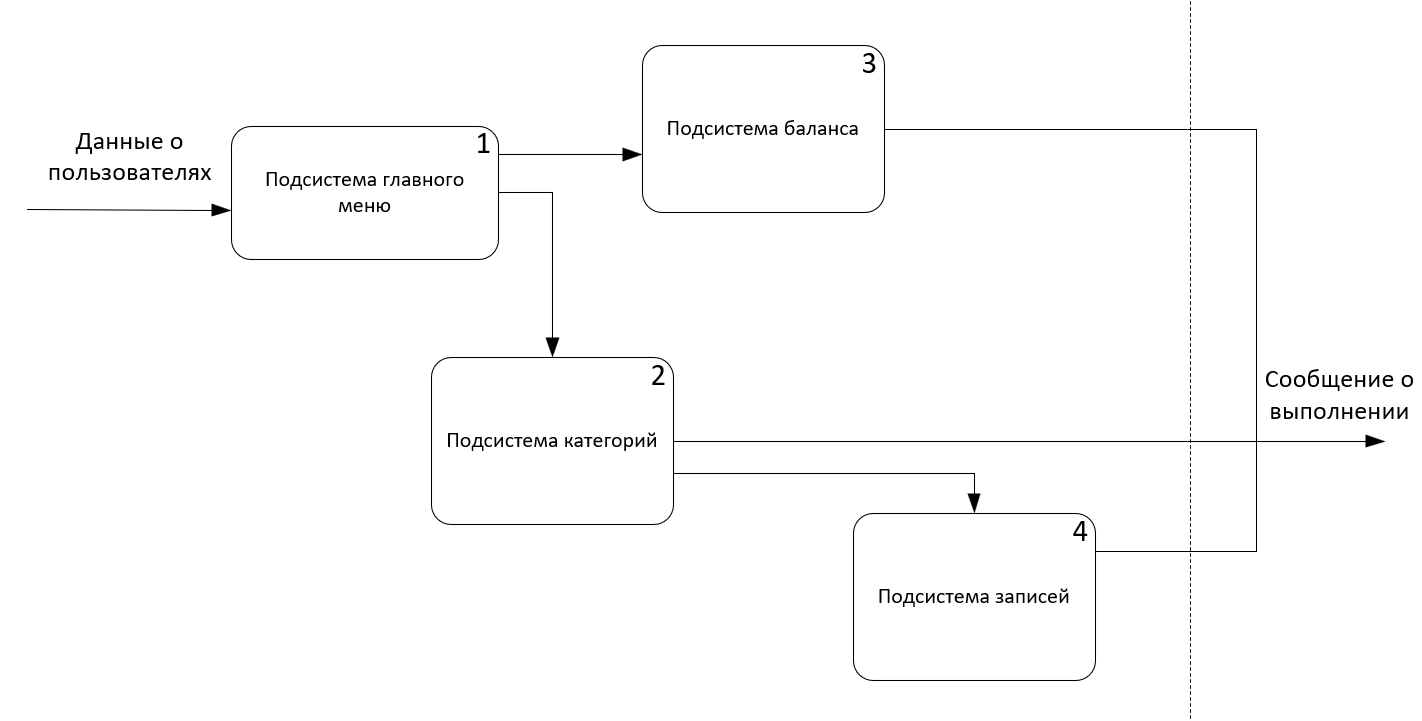


Рис. 5.2.4. DFD–диаграмма (часть 4)

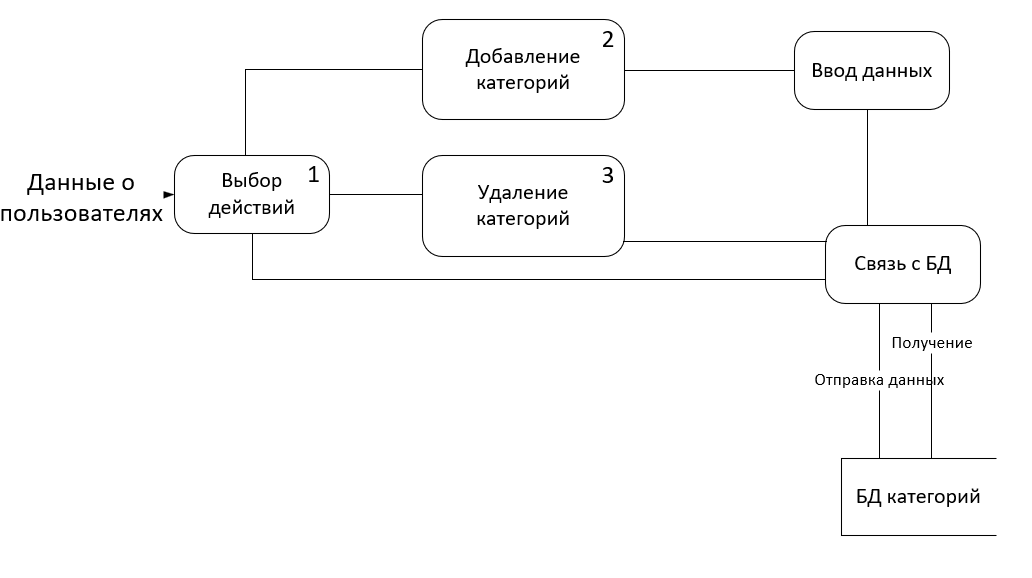


Рис. 5.2.5. DFD–диаграмма (часть 5)

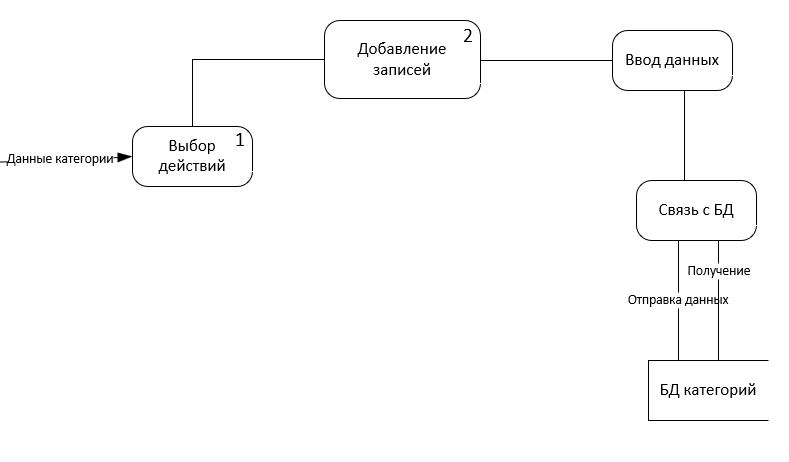


Рис. 5.2.6. DFD–диаграмма (часть 6)

## 5.3. Модель бизнес–процессов

Моделирование бизнес–процессов является одним из методов улучшения качества и эффективности работы организации. В основе этого метода лежит описание процесса через различные элементы (действия, данные, события, материалы и пр.) присущие процессу. Как правило, моделирование бизнес–процессов описывает логическую взаимосвязь всех элементов процесса от его начала до завершения в рамках организации. В более сложных ситуациях моделирование может включать в себя внешние по отношению к организации процессы или системы. Предоставлено на рисунках 5.3.1. – 5.3.3.

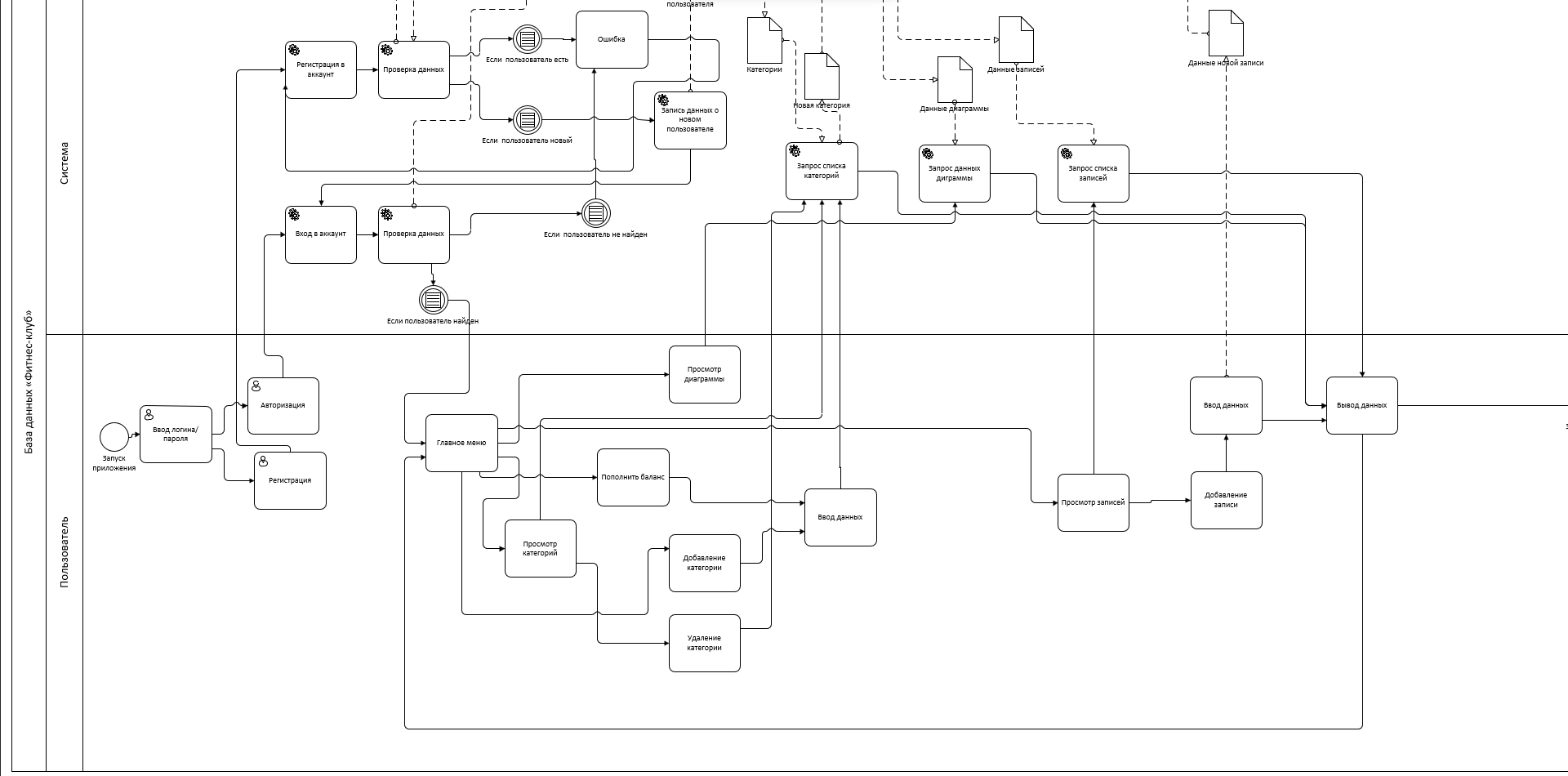


Рис. 5.3.1. Модель бизнес–процессов (часть 1)

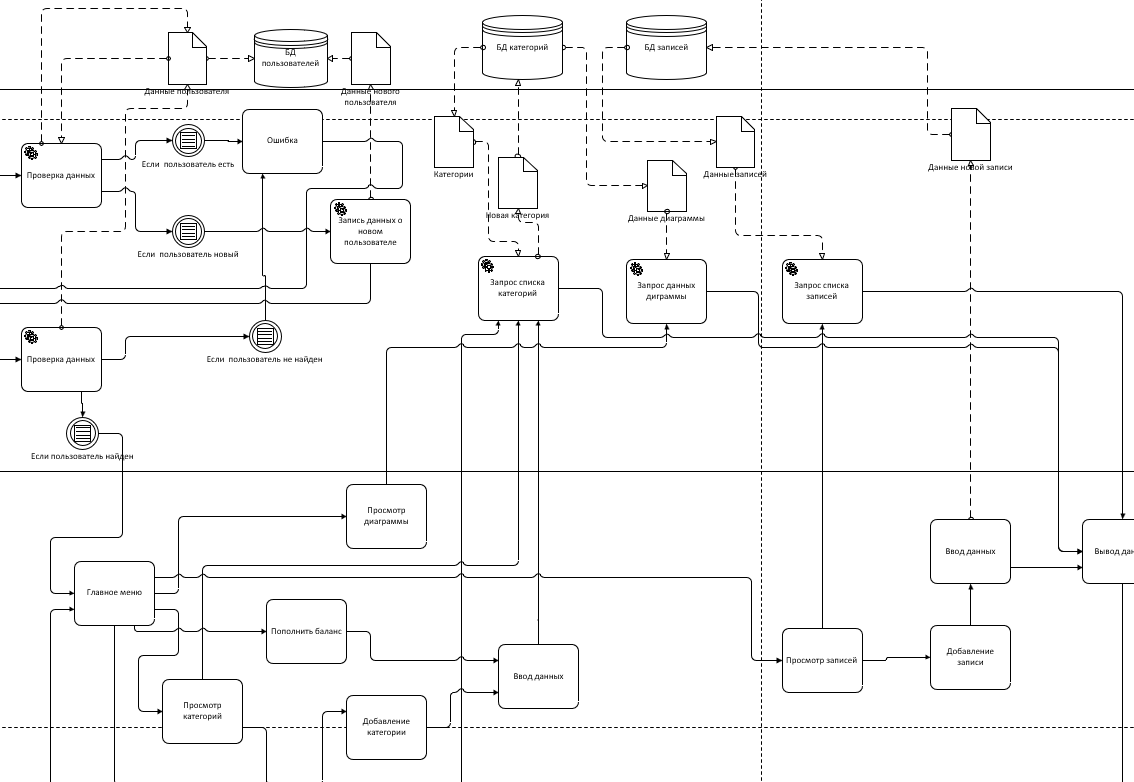


Рис. 5.3.2. Модель бизнес–процессов (часть 2)

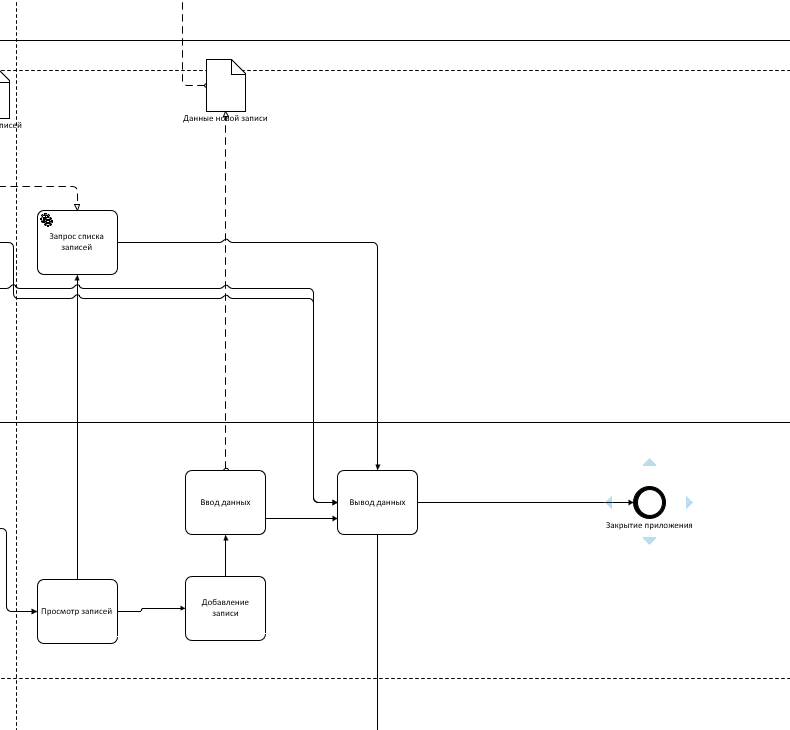


Рис. 5.3.3. Модель бизнес–процессов (часть 3)

## 5.4. Модель «сущность–связь» в разных нотациях

Модель «сущность–связь» в нотации Чена

Нотация ERD–диаграмм была предложена Ченом. Нотация Чена предоставляет богатый набор средств моделирования данных, включая, собственно, ERD–диаграммы, а также диаграммы атрибутов и диаграммы категоризации.

Эти диаграммные техники используются для моделирования и проектирования как реляционных, так и иерархических и сетевых баз данных.

Сущность представляет собой множество экземпляров реальных или абстрактных объектов (людей, событий, предметов, состояний, идей и т.п.), обладающих общими характеристиками (атрибутами).

Отношение представляет собой связь между двумя и более сущностями.

Именование отношения осуществляется с помощью грамматического оборота глагола (Имеет, определяет, может владеть и т. п.).

Диаграмма модель «сущность–связь» в нотации Чена, предоставлена на рисунке 5.4.1.

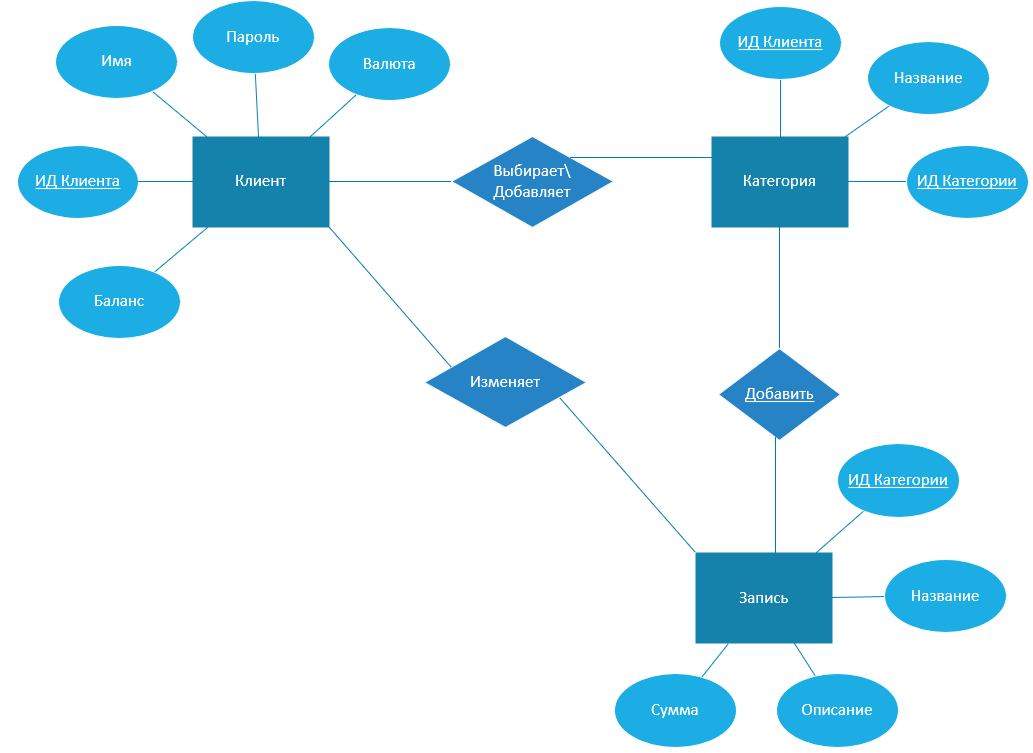


Рис. 5.4.1. Модель «сущность–связь» в нотации Чена

Модель «сущность–связь» в нотации Баркера

Дальнейшее развитие диаграммы «сущность–связь» получили в работах Баркера, предложившего оригинальную нотацию, которая позволила интегрировать средства описания моделей Чена.

В нотации Баркера используется только один тип диаграмм: собственно, ERD–диаграммы. Сущность на ERD–диаграмме представляется прямоугольником любого размера, содержащим внутри себя имя сущности, список имен атрибутов (возможно, неполный) и указатели ключевых атрибутов (знак «#» перед именем атрибута. Атрибуты сущностей могут быть обязательными и необязательными. Кроме того, атрибуты могут быть описательными (дескрипторами сущности) или входить в состав первичного ключа (уникального идентификатора сущности). Сущность, как правило, может быть полностью идентифицирована своим первичным ключом. Однако, могут быть случаи, когда в идентификации данной сущности участвуют также атрибуты другой сущности–родителя. Кроме первичных ключей могут использоваться альтернативные (возможные) ключи.

Диаграмма модель «сущность–связь» в нотации Баркера, предоставлена на рисунке 5.4.2.

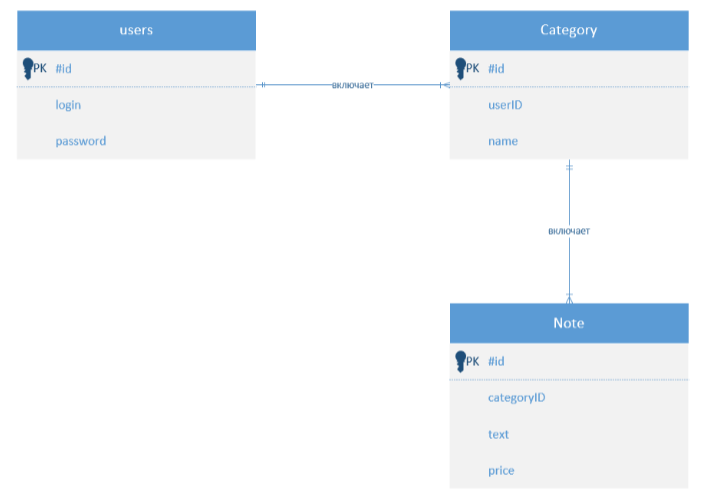


Рис. 5.4.2. Модель «сущность–связь» в нотации Баркера

Модель «сущность–связь» в нотации Мартина

Данная нотация является одной из наиболее известных в разработке баз данных, отражающей уровень логического представления базы данных с обозначением некоторых компонентов модели базы данных в графическом виде, облегчая, тем самым, отображение диаграммы в рабочем пространстве. Модели такого типа менее громоздки по сравнению с моделями в нотации Питера Чена. Тем не менее, сложность предметных областей нередко мешает представлению всей модели в рамках единого рабочего пространства, что во многих средствах моделирования баз данных исправляется возможностью формирования и представления моделей базы данных в разрезе отдельных рабочих пространств, которые могут соответствовать функциональному делению предметной области или какому–либо другому фактору, уменьшающему количество рассматриваемых элементов модели базы данных.

Диаграмма модель «сущность–связь» в нотации Мартина, предоставлена на рисунке 5.4.3.

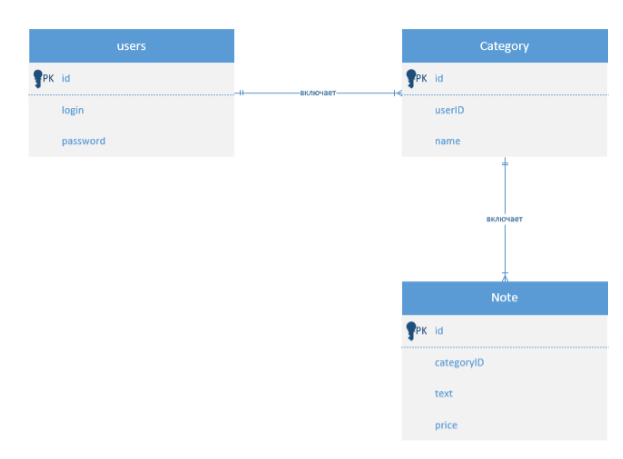


Рис. 5.4.3. Модель «сущность–связь» в нотации Мартина

## 5.5. Диаграмма прецедентов

Диаграмма вариантов использования в UML – диаграмма, отражающая отношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Диаграмма прецедентов предоставлена на рисунке 5.5.

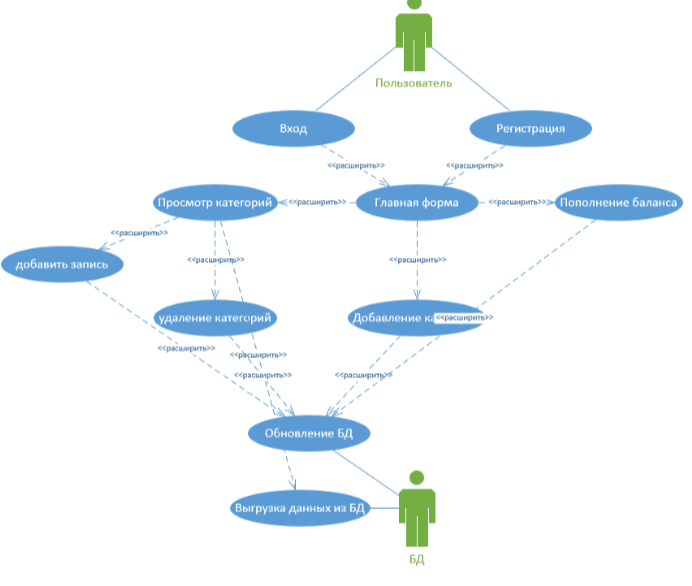


Рис. 5.5. Диаграмма прецедентов

## 5.6. Диаграммы деятельности и состояний

Диаграмма деятельности – UML–диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний.

Диаграмма состояния (фазовая диаграмма), графическое изображение всех возможных состояний термодинамической системы в пространстве основных [параметров состояния](http://www.chemport.ru/data/chemipedia/article_2684.html) . Для сложных систем, состоящих из многих фаз и компонентов, построение диаграмма состояния является единственным методом, позволяющим на практике установить, сколько фаз и какие конкретно фазы образуют систему при данных значениях параметров состояния.

Каждое реально существующее состояние системы на диаграмме состояния изображается так называемой фигуративной точкой; областям существования одной фазы отвечают участки пространства (на трехмерных диаграммах состояния) или плоскости (на двумерных диаграммах состояния), условиям сосуществования фаз соответствуют поверхности или линии; изменение фазового состояния системы рассматривается как движение фигуративной точки на диаграмме состояния. Анализ относительного расположения объемных участков, поверхностей, линий и точек, которые образуют диаграмму состояния, позволяет однозначно и наглядно определять условия фазового равновесия, появления в системе новых фаз и химических соединений, образования и распада жидких и твердых растворов и т. п.

Диаграмма деятельности предоставлена на рисунках 5.6.1 – 5.6.2.

Диаграмма состояния предоставлена на рисунках 5.6.3 – 5.6.5.

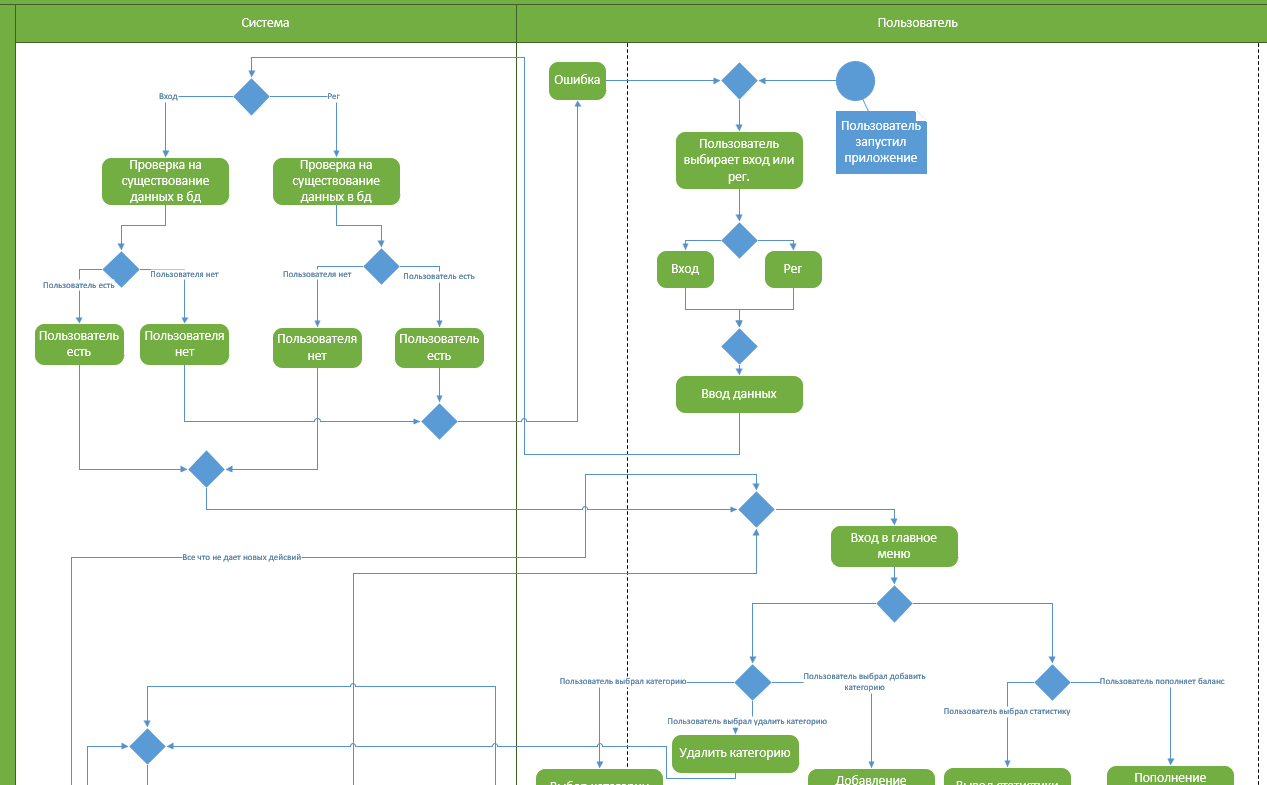


Рис. 5.6.1. Диаграмма деятельности (часть 1)

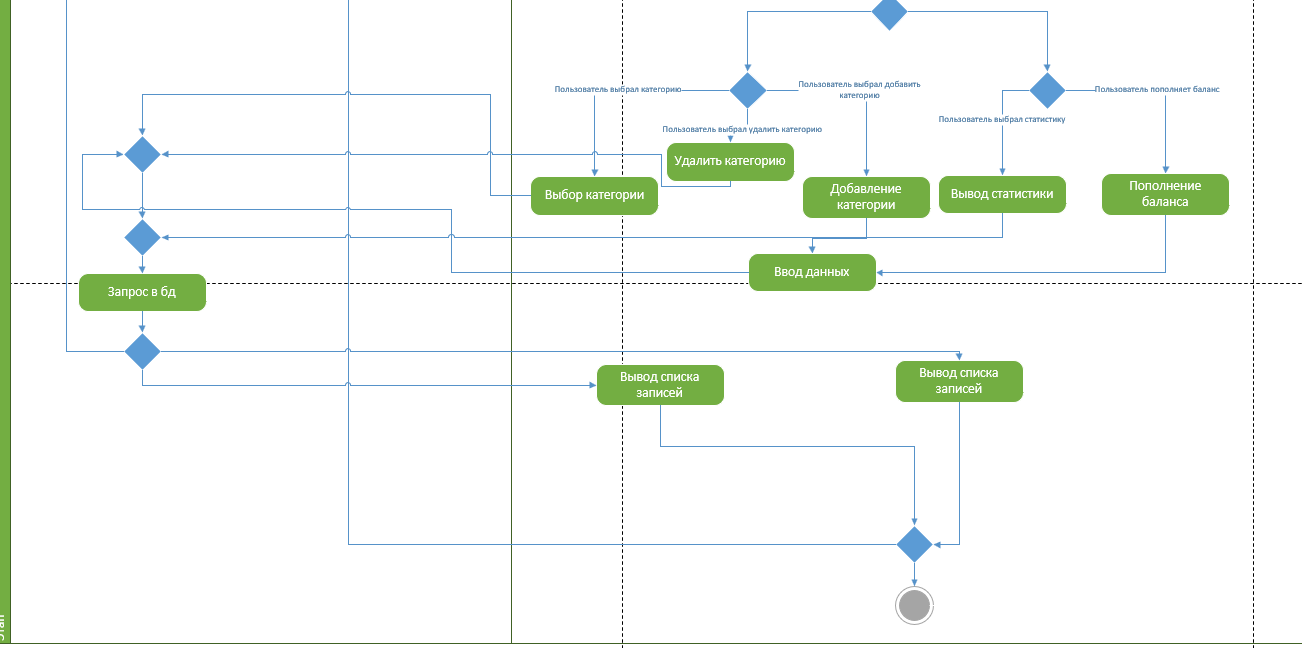


Рис. 5.6.2. Диаграмма деятельности (часть 2)

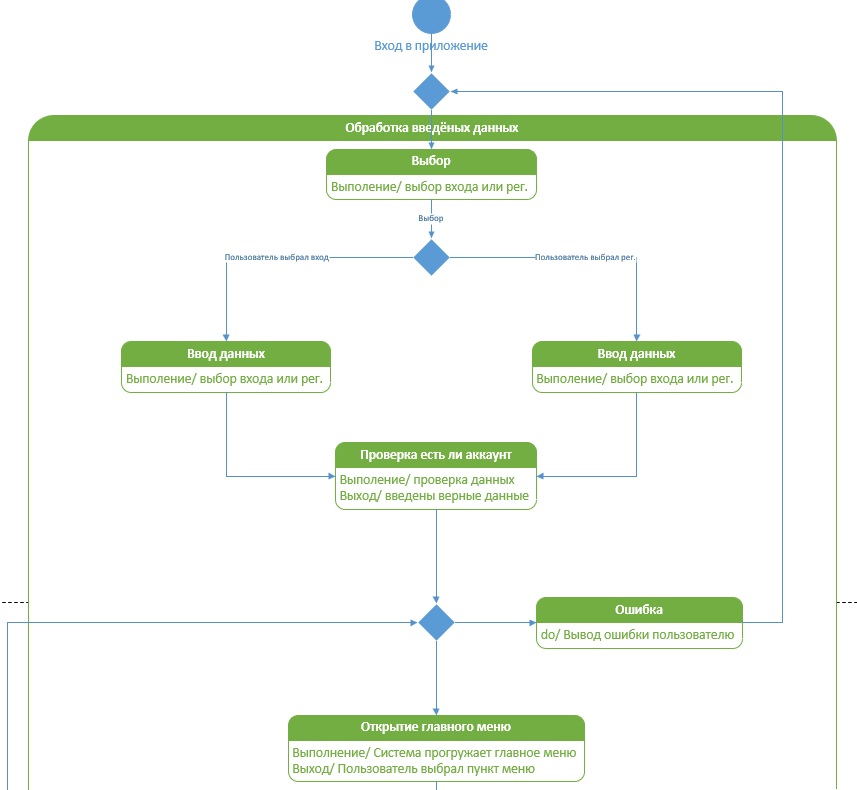


Рис. 5.6.3. Диаграмма состояния (часть 1)

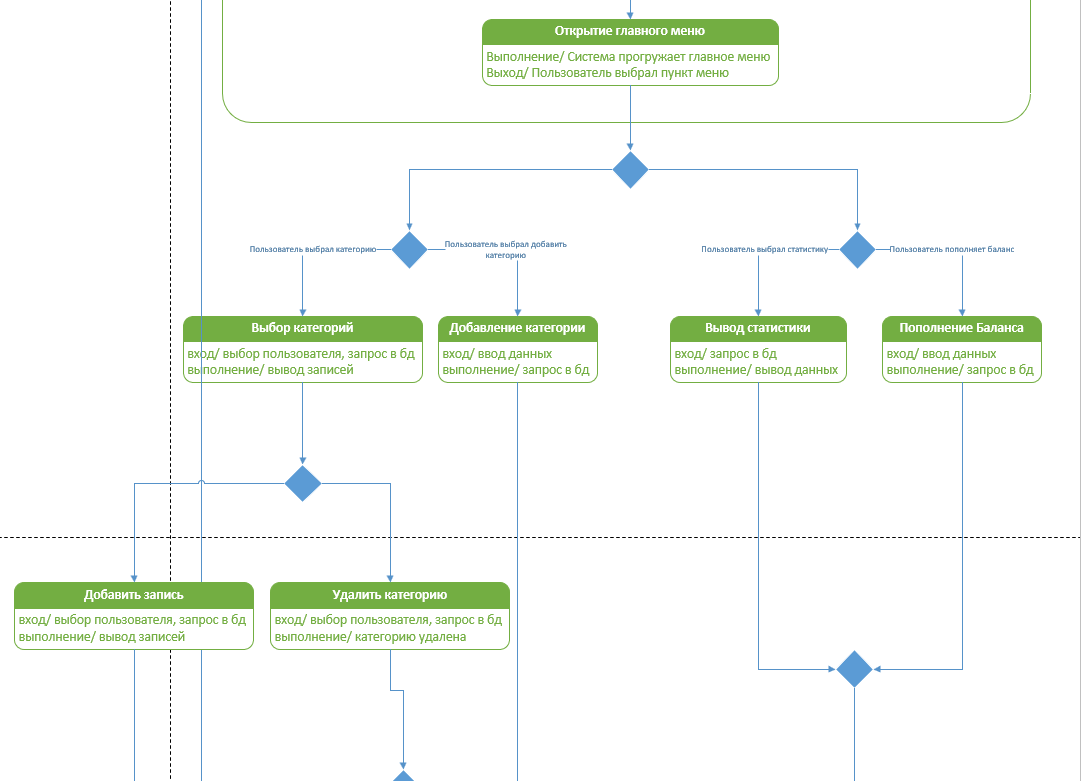


Рис. 5.6.4. Диаграмма состояния (часть 2)

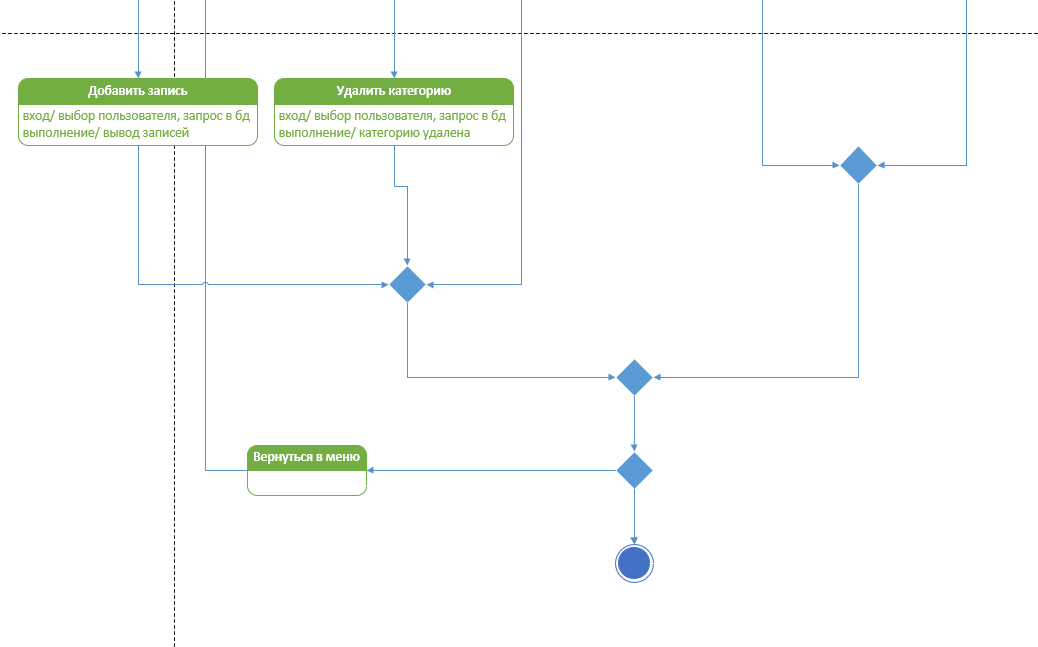


Рис. 5.6.5. Диаграмма состояния (часть 3)

# 6. СХЕМА ДИАЛОГА ПОЛЬЗОВАТЕЛЯ С РАЗРАБОТАННЫМ ПРИЛОЖЕНИЕМ

Схема диалога с пользователем предоставляет инструкцию по использованию приложения.

## 6.1. Схема диалога пользователя с разработанным приложением

Чтобы начать использование калькулятора расходов как пользователь, необходимо указать логин и пароль и нажать на кнопку «Регистрация». Пример на рисунке 6.1.

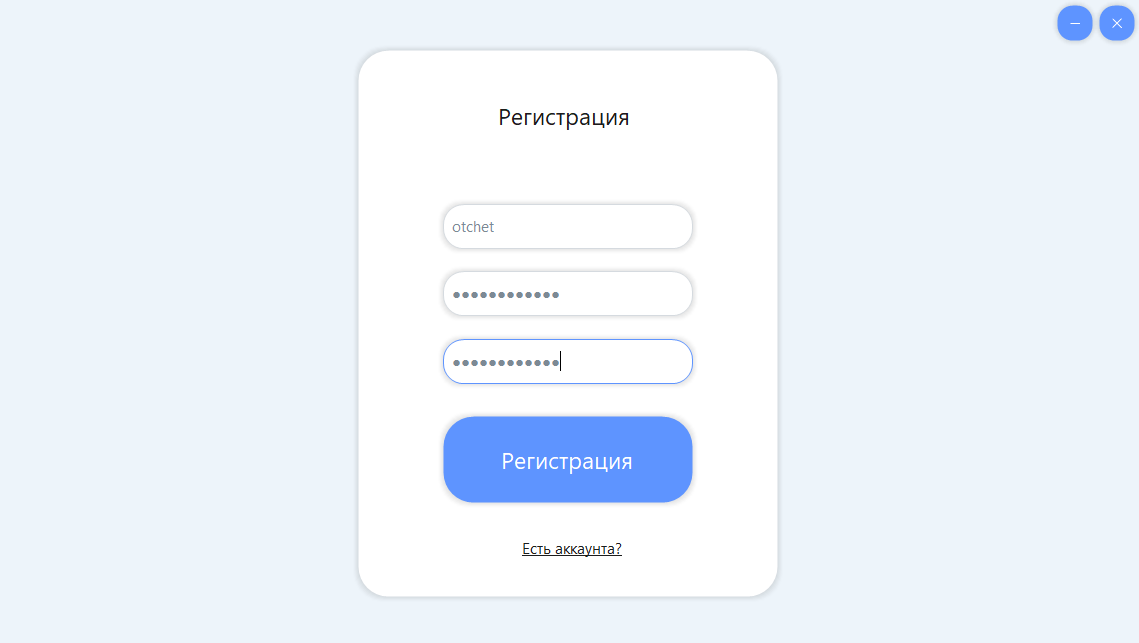


Рис. 6.1. Регистрация

Если же пользователь зарегистрирован успешно, указываем данные зарегистрированного аккаунта и нажимаем кнопку «Вход». Пример на рисунке 6.2.

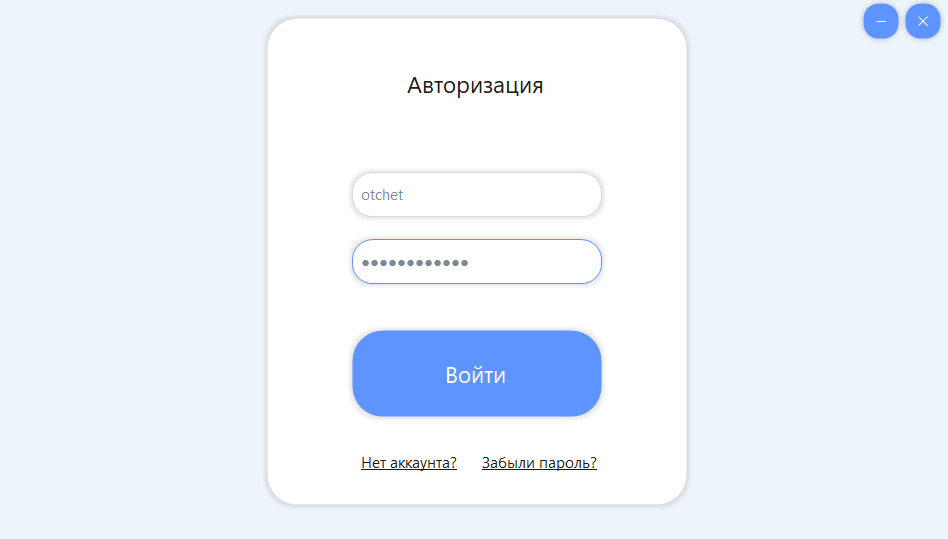


Рис. 6.2. Авторизация

Далее мы оказываемся на главной форме. Пользователь может выбрать валюту баланса. Показано на рисунке 6.3.

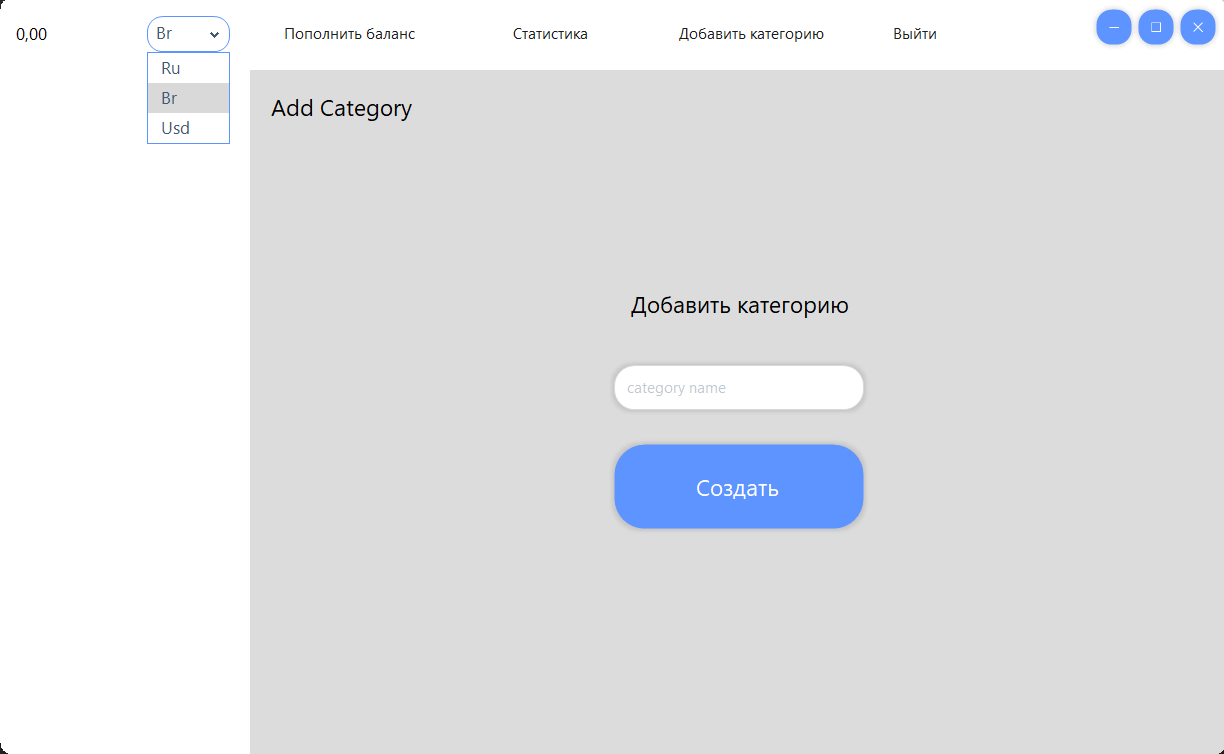


Рис. 6.3. Выбор валюты

Пользователь может создать категорию, для этого необходимо перейти во складку «Создать категорию» и указать название. Далее пользователь может увидеть категорию в списке слева. При нажатии на кнопку категории откроется страница данной категории, где можно добавить запись что представлено на рисунке 6.4.

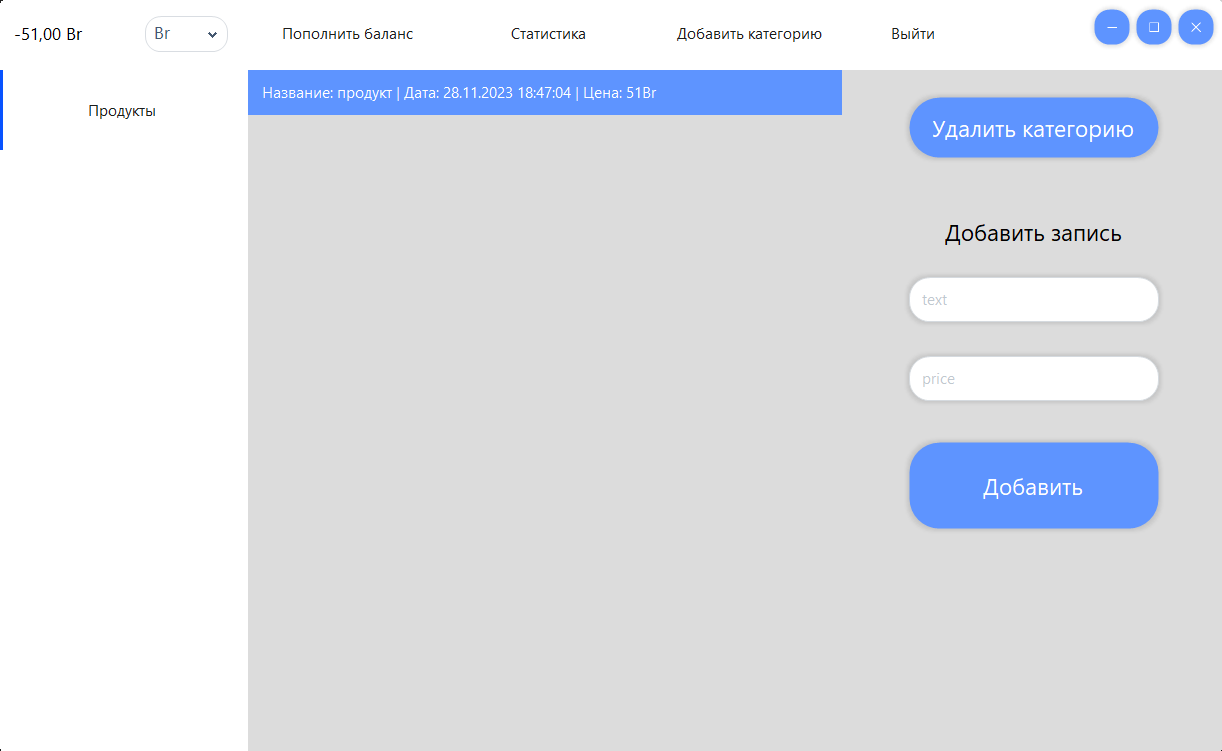


Рис. 6.4. Создание категории и записи

Пользователь может пополнить свой баланс через соответствующую вкладку. Необходимо указать сумму пополнения и нажать «Пополнить». Показано на рисунке 6.5.

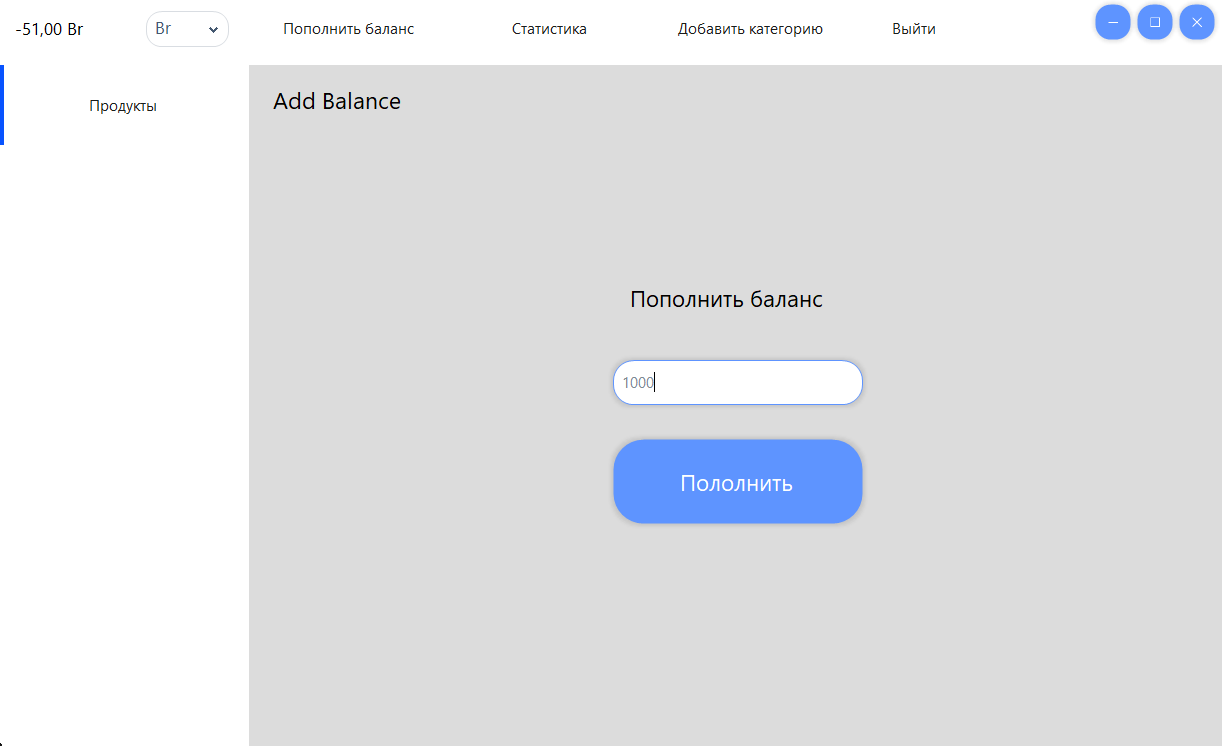


Рис. 6.5. Пополнение баланса

# 7. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Тестирование программного обеспечения – процесс исследования, испытания ПО с целью получения информации о качестве продукта.

Существующие на сегодня методы тестирования ПО не позволяют однозначно и полностью выявить все дефекты и установить корректность функционирования анализируемой программы, поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого ПО.

Такой процесс формальной проверки, или верификации, может доказать, что дефекты отсутствуют с точки зрения используемого метода. То есть, нет никакой возможности точно установить или гарантировать отсутствие дефектов в программном продукте с учетом человеческого фактора, присутствующего на всех этапах жизненного цикла ПО.

Тестирование черного ящика – специальный метод проверки работоспособности программного обеспечения, при котором вся функциональность продукта исследуется без анализа исходного кода. Тестировщики создают логически понятные тест–кейсы, опираясь исключительно на требования из спецификации на проекте.

В проекте использовался метод тестирования «черного ящика».

Программа была протестирована на следующих тестах:

Тест 1:

Тестирование регистрации

Вводим в поле логин 24

Вводим в поле пароль 24

Вводим в поле повторить пароль 24

Ожидаемый результат: Имя пользователя должно содержать от 5 до 20 символов!

Результат, представлен на рисунке 7.1.

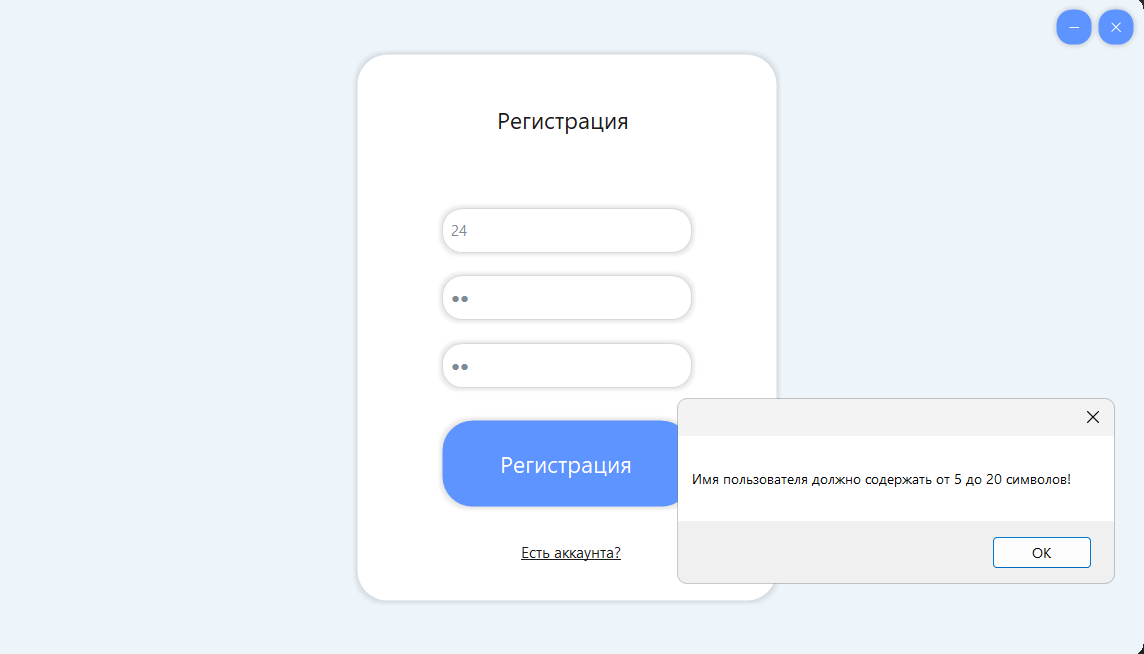


Рис. 7.1. Тест №1

Тест 2:

Тестирование регистрации

Вводим в поле логин 24

Вводим в поле пароль 2

Вводим в поле повторить пароль, 24

Ожидаемый результат: Пароли не совпадают

Результат, представлен на рисунке 7.2.

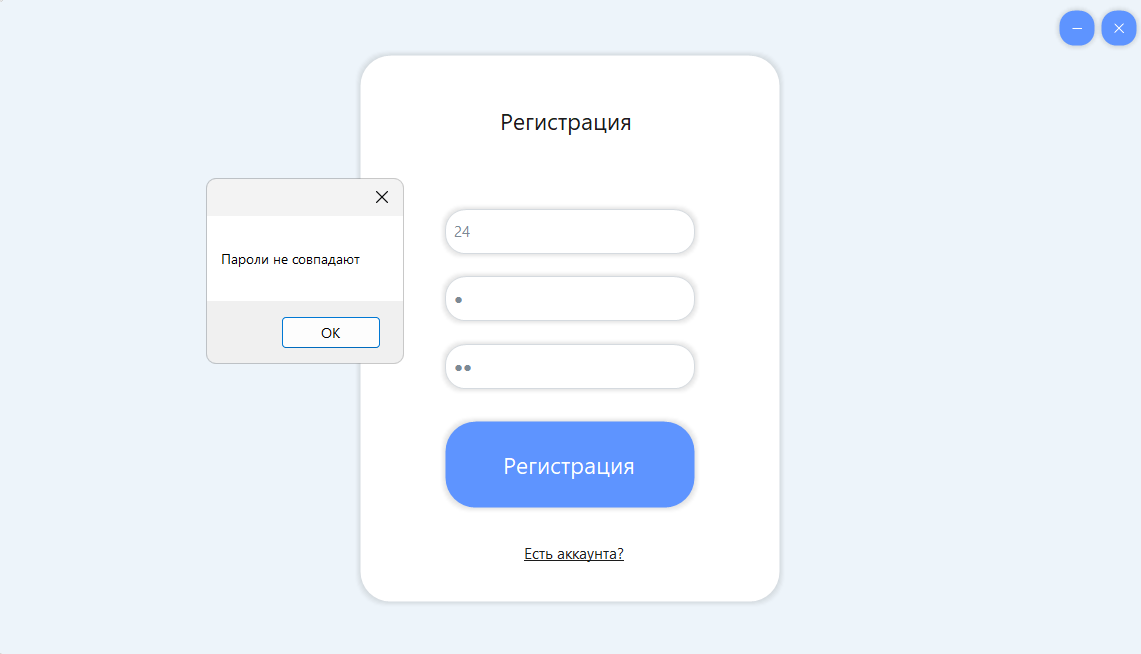


Рис. 7.2. Тест №2

Тест 3:

Тестирование авторизации

Вводим в поле логин 12345

Вводим в поле пароль 1234

Ожидаемый результат: Неправильное имя пользователя или пароль!

Результат, представлен на рисунке 7.3.

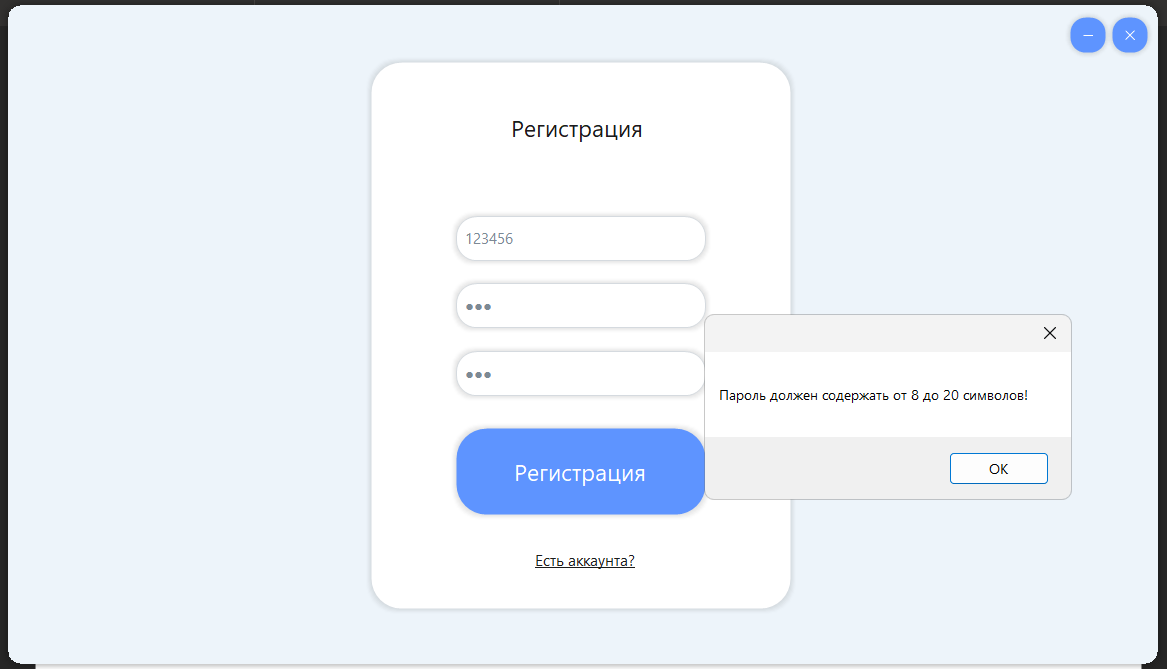


Рис. 7.3. Тест 3

Тест 4:

Тестирование регистрации

Вводим в поле логин 12345

Вводим в поле пароль 123456789

Вводим в поле повторить пароль, 123456789

Ожидаемый результат: Успешная регистрация!

Результат, представлен на рисунке 7.4.

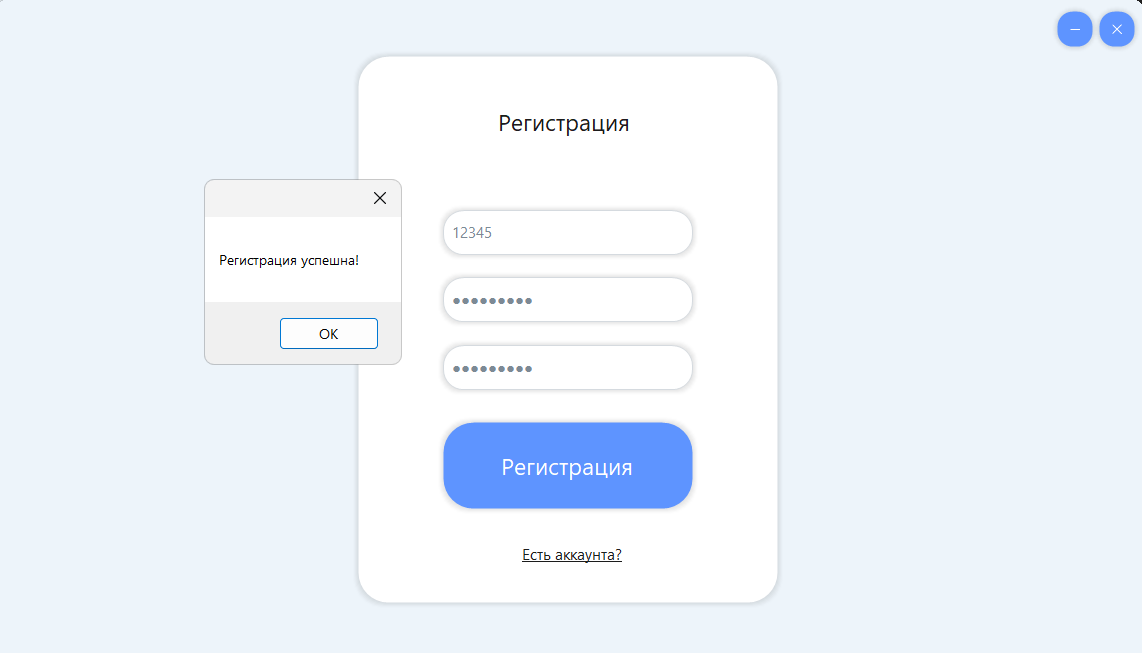


Рис. 7.4. Тест №4

Данные в БД после регистрации:

Результат, представлен на рисунке 7.5.

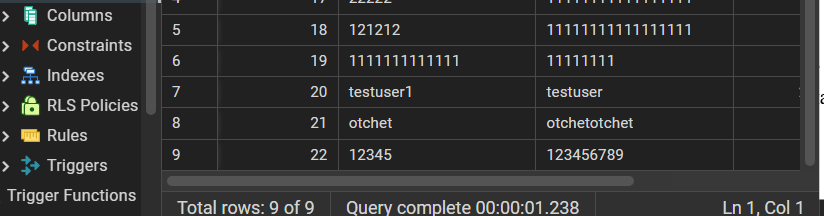


Рис. 7.5. Тест №4

Тест 5:

Тестирование регистрации

Вводим в поле логин otchet (данные уже зарегистрированного аккаунта)

Вводим в поле пароль otchetotchcet

Вводим в поле повторить пароль, otchetothcet

Ожидаемый результат: Пользователь с таким именем уже существует!

Результат, представлен на рисунке 7.6.

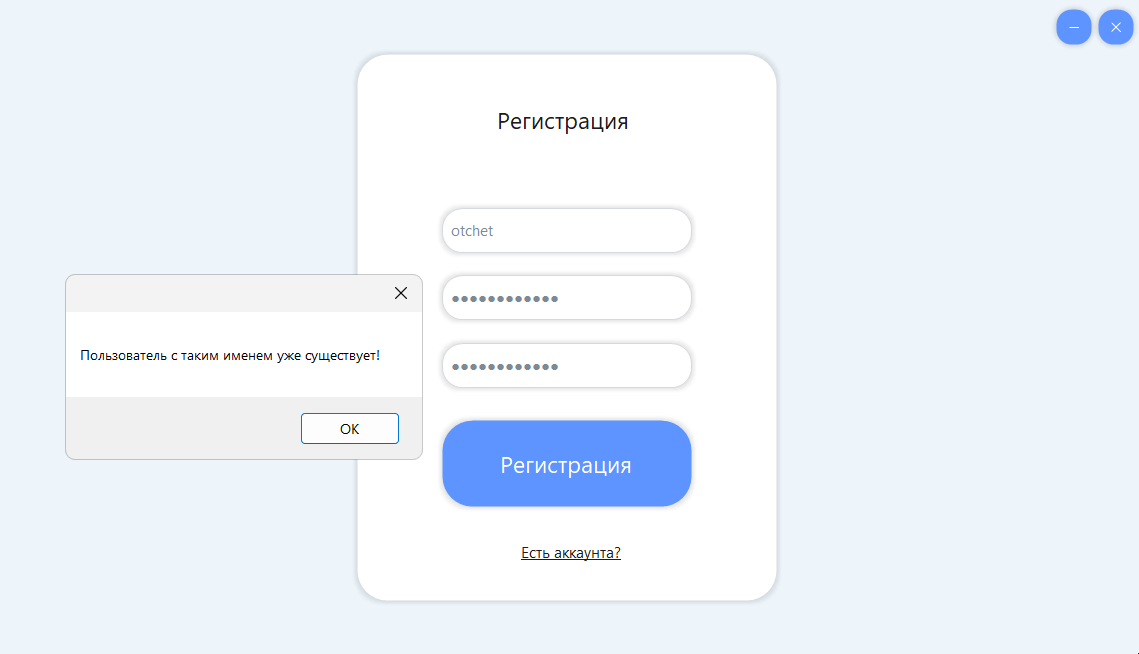


Рис. 7.6. Тест №5

Тест 6:

Тестирование авторизации

Вводим в поле логин «1234»

Вводим в поле пароль «1234»

Ожидаемый результат: Неправильное имя пользователя или пароль!

Результат, представлен на рисунке 7.7.

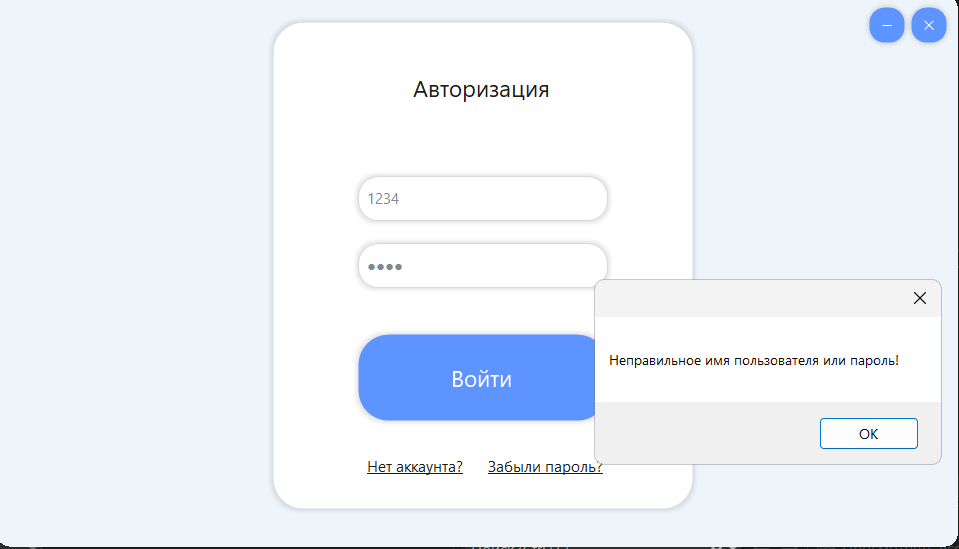


Рис. 7.7. Тест №6

Тест 7:

Тестирование авторизации

Вводим в поле логин «otchet»

Вводим в поле пароль «otchetotchet»

Ожидаемый результат: Открытие главной формы

Результат, представлен на рисунках 7.8. – 7.9.

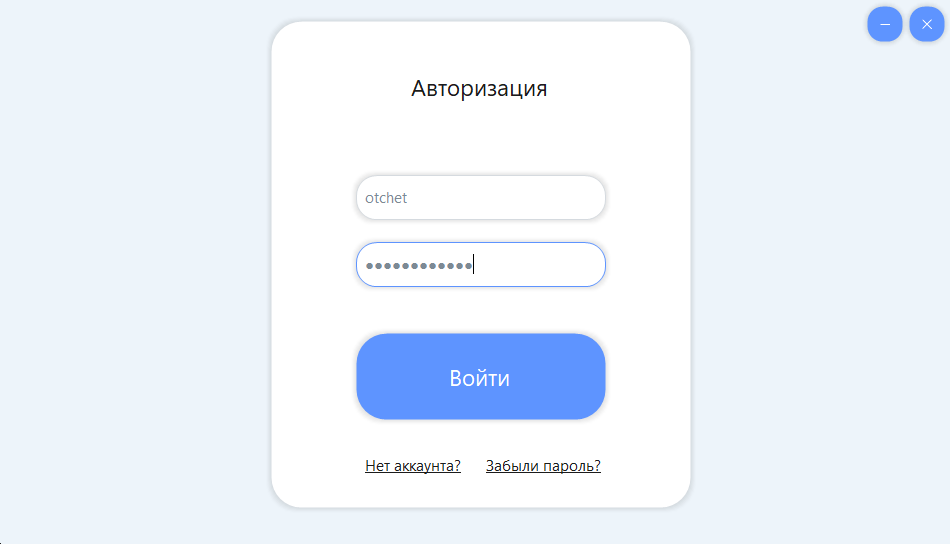


Рис. 7.8. Тест №7

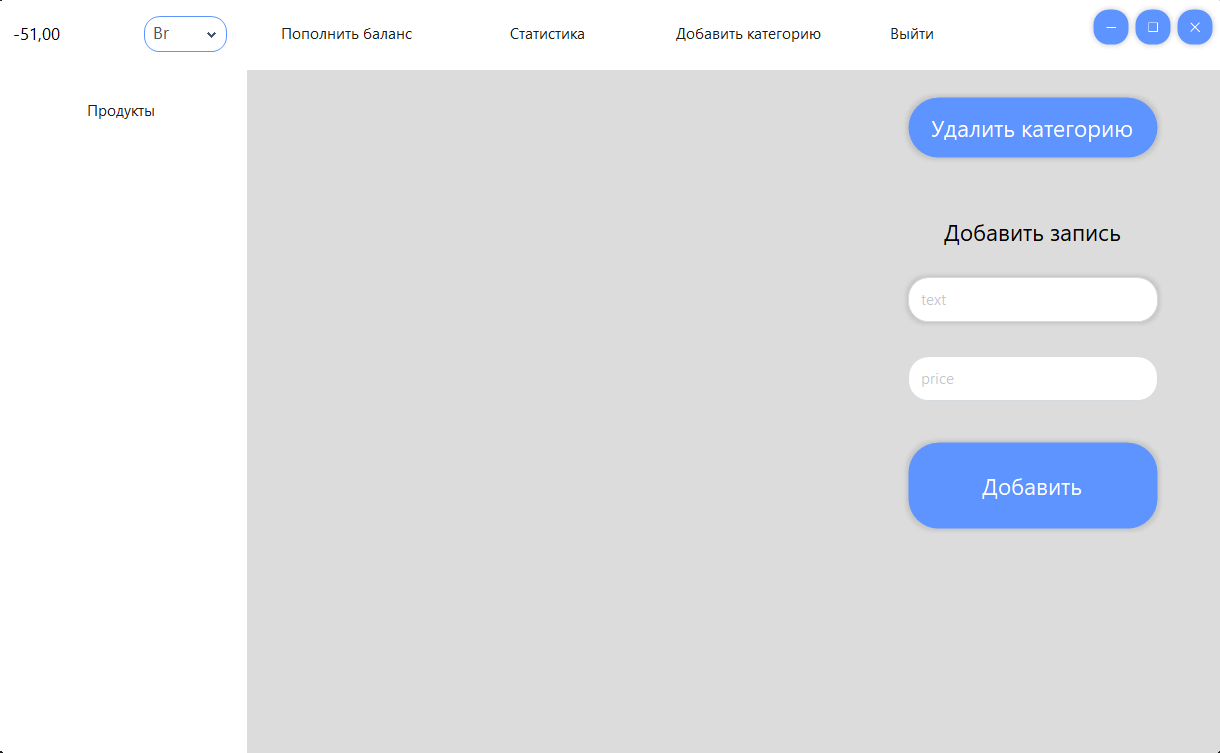


Рис. 7.9. Тест №7

Тест 8:

Тестирование главное окно

Открыть вкладку баланса, указать сумму пополнения и нажать на кнопку «Пополнить»

Ожидаемый результат: баланс увеличится на указанную сумму

Результат, представлен на рисунке 7.10. – 7.11.

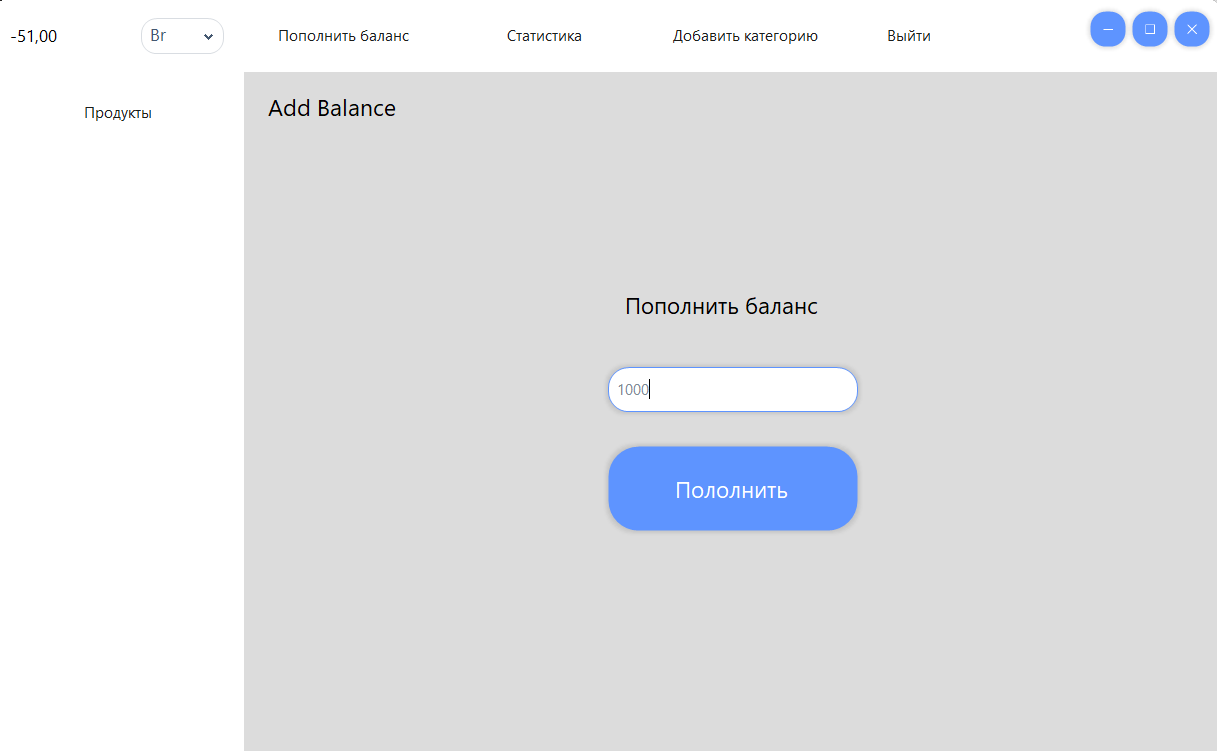


Рис. 7.10. Тест №8

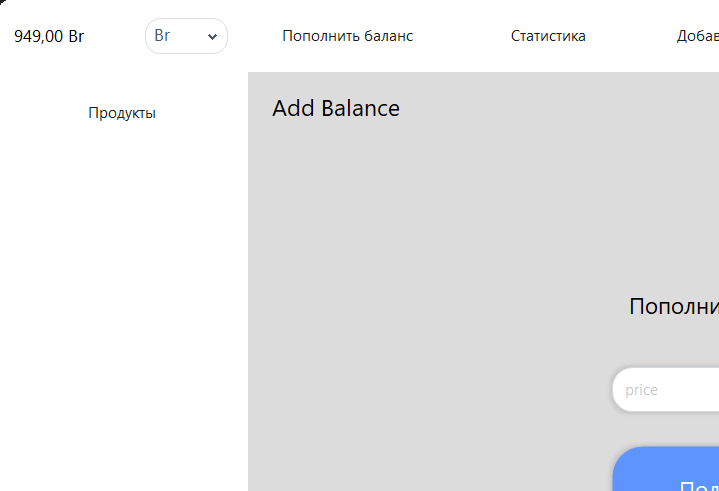


Рис. 7.11. Тест №8

Тест 9:

Тестирование главное окно

Открыть вкладку «Добавить категорию», указать название и нажать на кнопку «Создать»

Ожидаемый результат: Создание новой категории в списке категорий слева

Результат, представлен на рисунке 7.12. – 7.13.

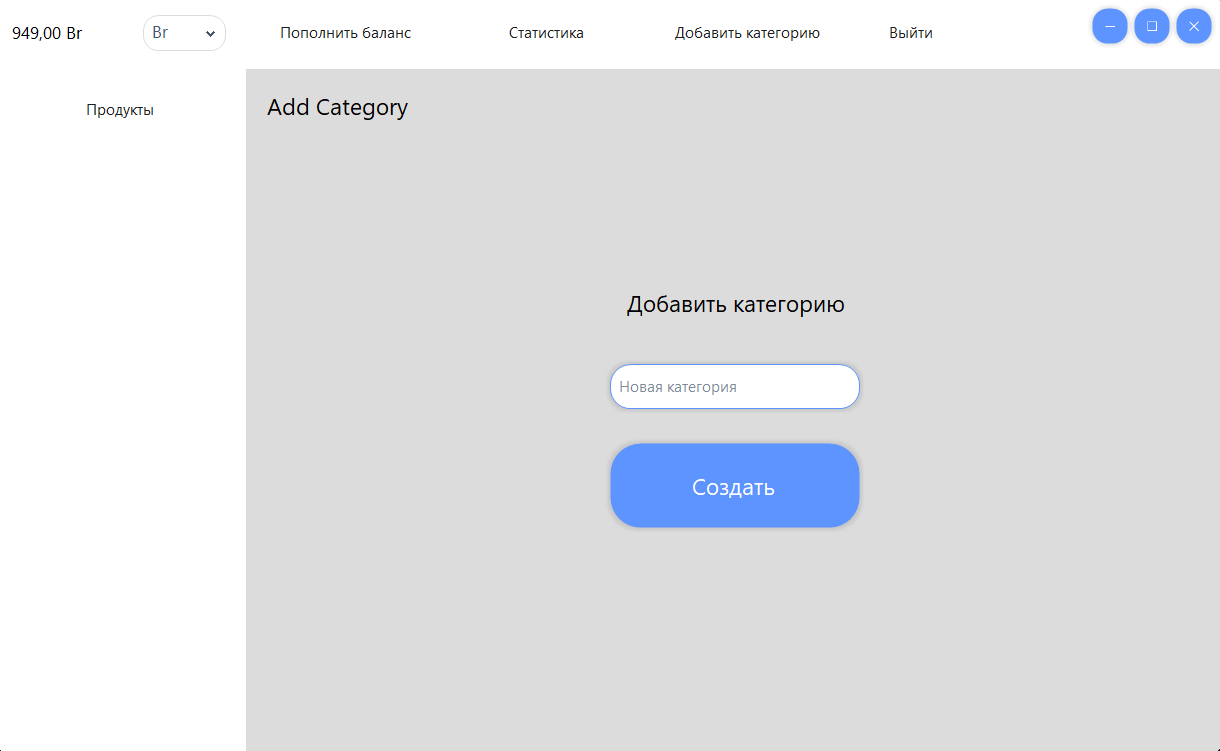


Рис. 7.12. Тест №9

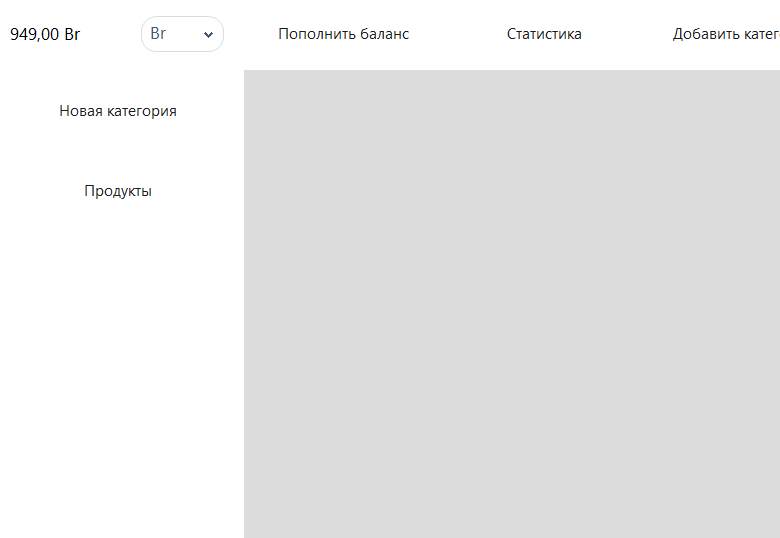


Рис. 7.13. Тест №9

Тест 10:

Тестирование создания записи в категории

Открыть категорию в левом списке, указываем данные во все поля и нажимает кнопку «Добавить»

Ожидаемый результат: создана новая запись

Результат, представлен на рисунке 7.13. – 7.14.

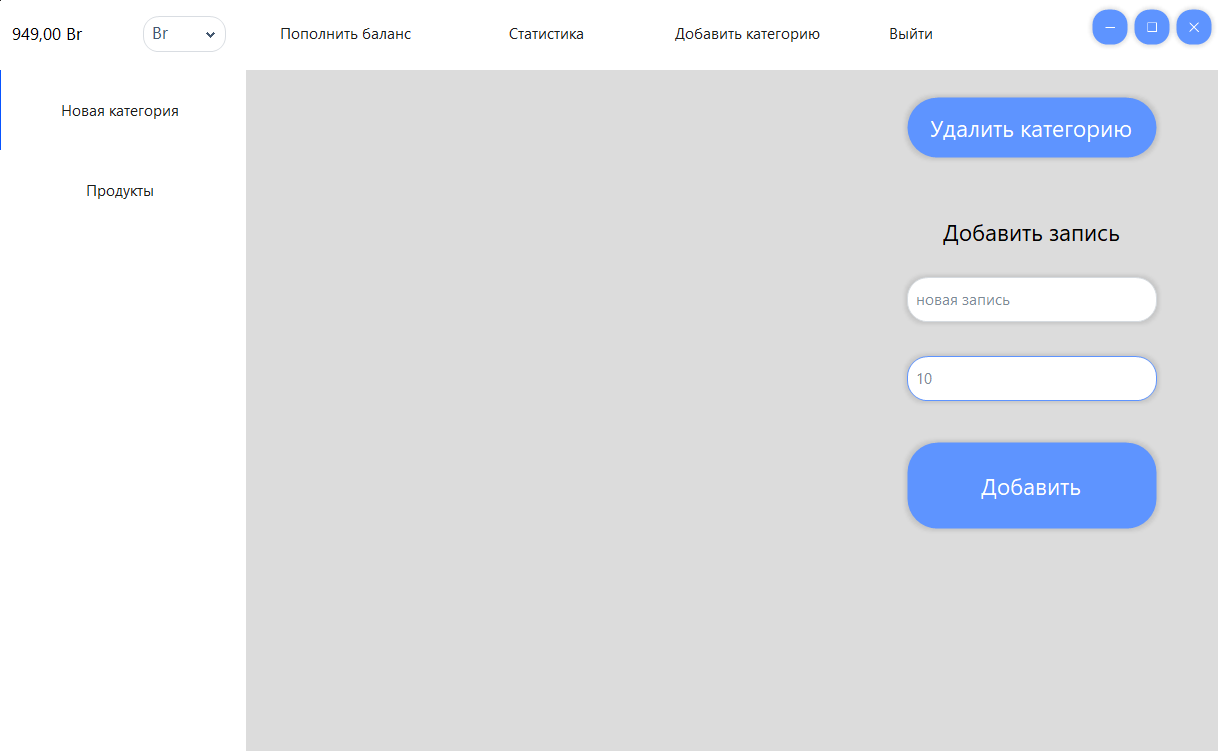


Рис. 7.13. Тест №10

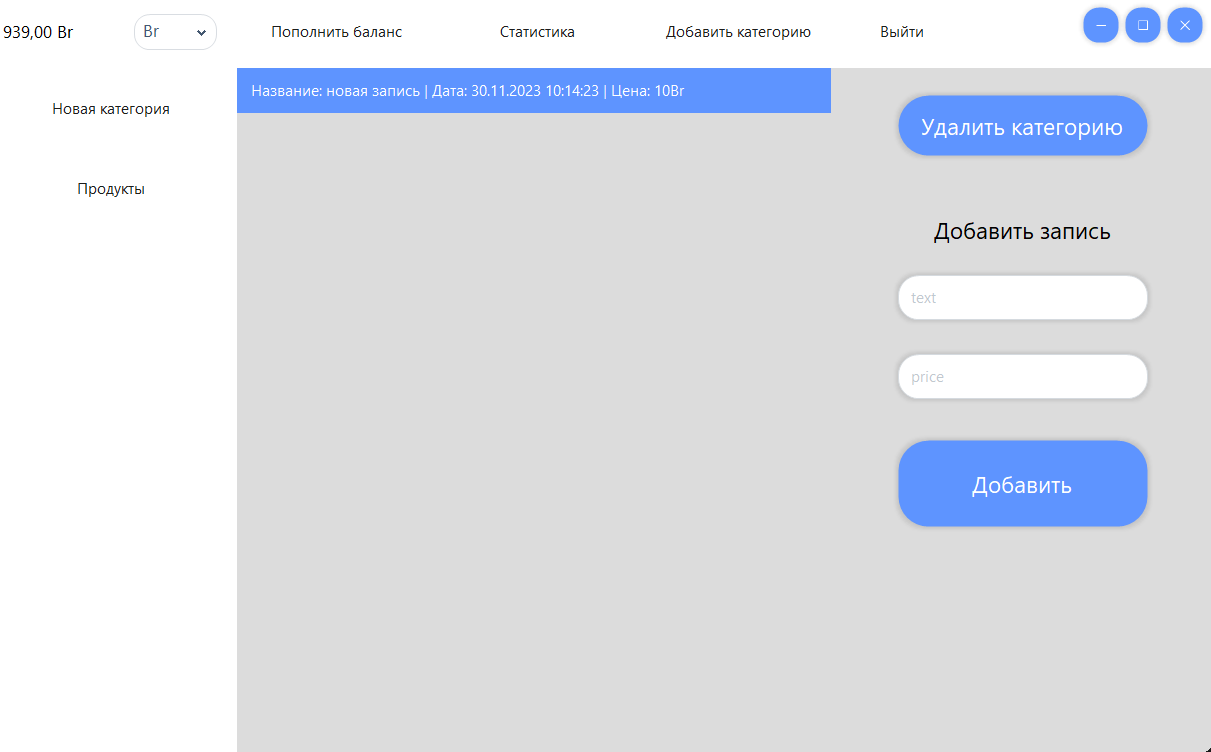


Рис. 7.14. Тест №10

Тест 11:

Тестирование главное окно

Открытие вкладки «Статистика»

Ожидаемый результат: Вывод статистики

Результат, представлен на рисунке 7.15.

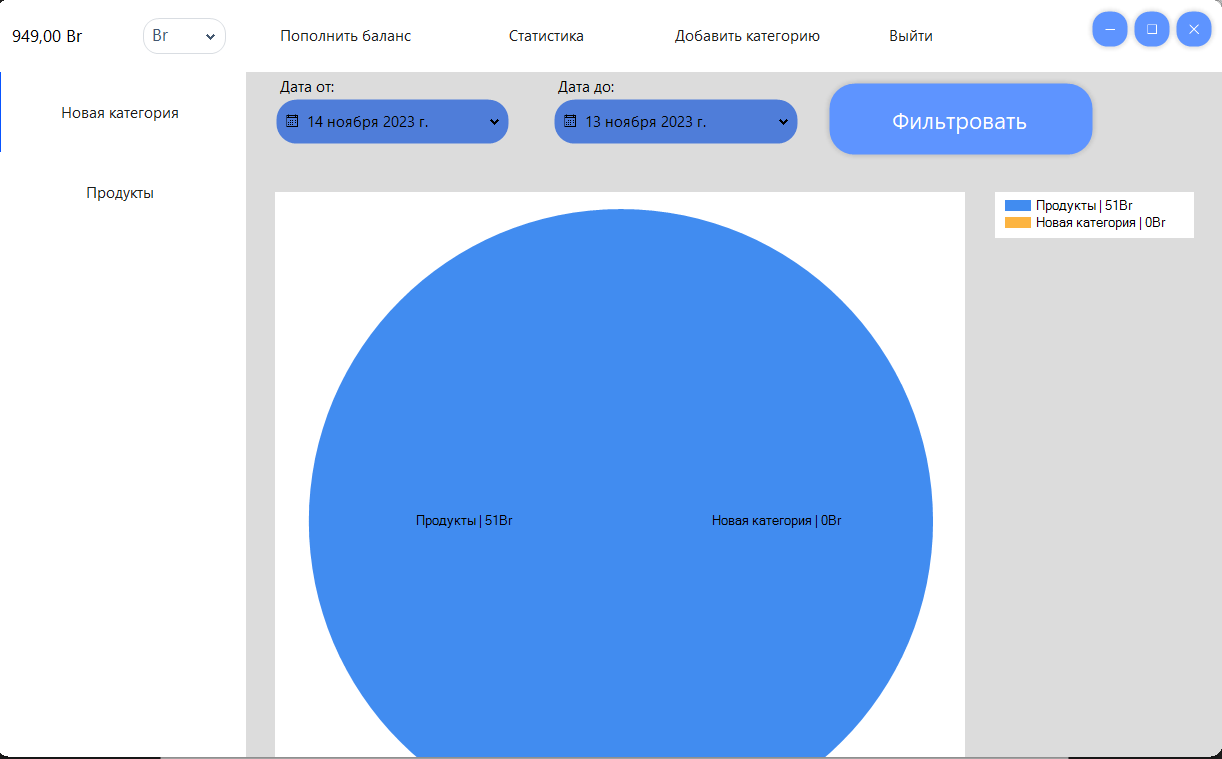


Рис. 7.15. Тест №11

Тест 12:

Тестирование главное окно

Открытие вкладки «Статистика» и выбор даты без записей

Ожидаемый результат: Статистика не выводиться

Результат, представлен на рисунке 7.16.

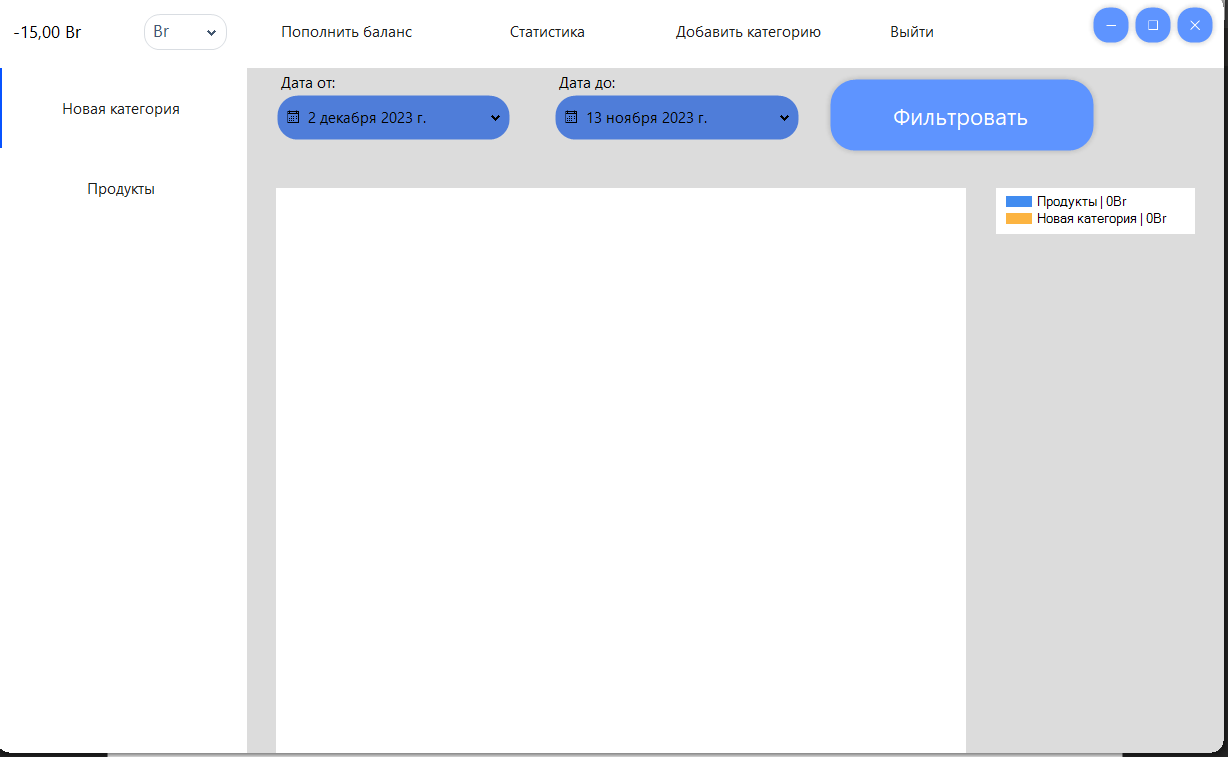


Рис. 7.16. Тест №12

Тест 13:

Тестирование главной формы

Вводим в поле баланса буквы

Ожидаемый результат Ошибка ввода!

Результат, представлен на рисунке 7.17.

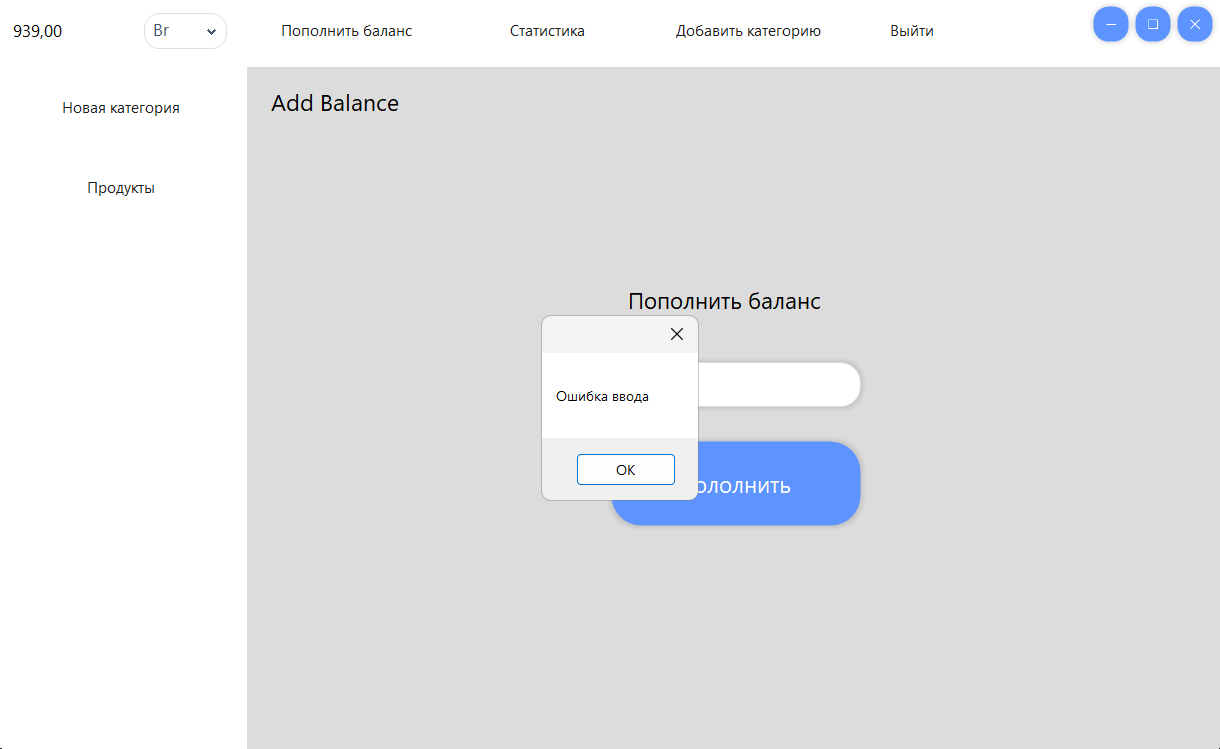


Рис. 7.17. Тест 13

Тест 14:

Тестирование главной формы

Вводим в поле стоимости буквы

Ожидаемый результат Ошибка создания!

Результат, представлен на рисунке 7.18.

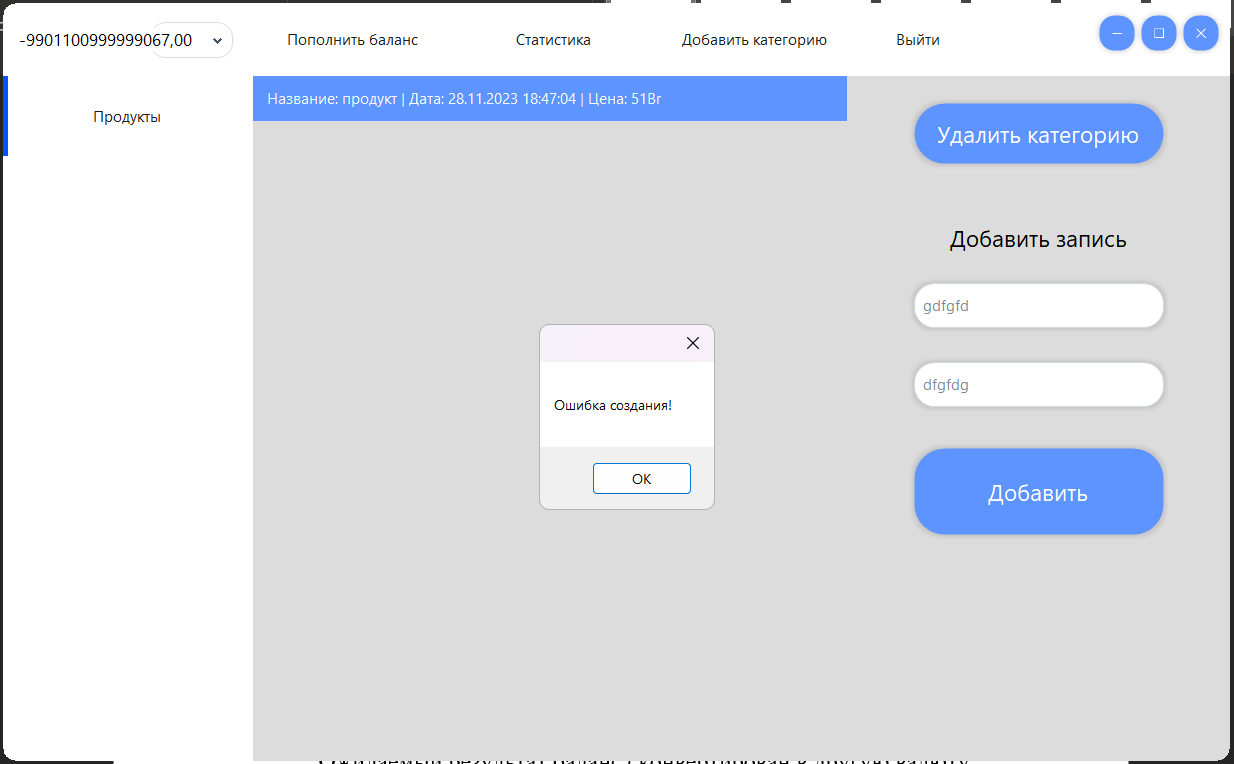


Рис. 7.18. Тест 14

Тест 15:

Тестирование главной формы

Создаем много записей

Ожидаемый результат записи можно будет листать

Результат представлен на рисунке 7.19.

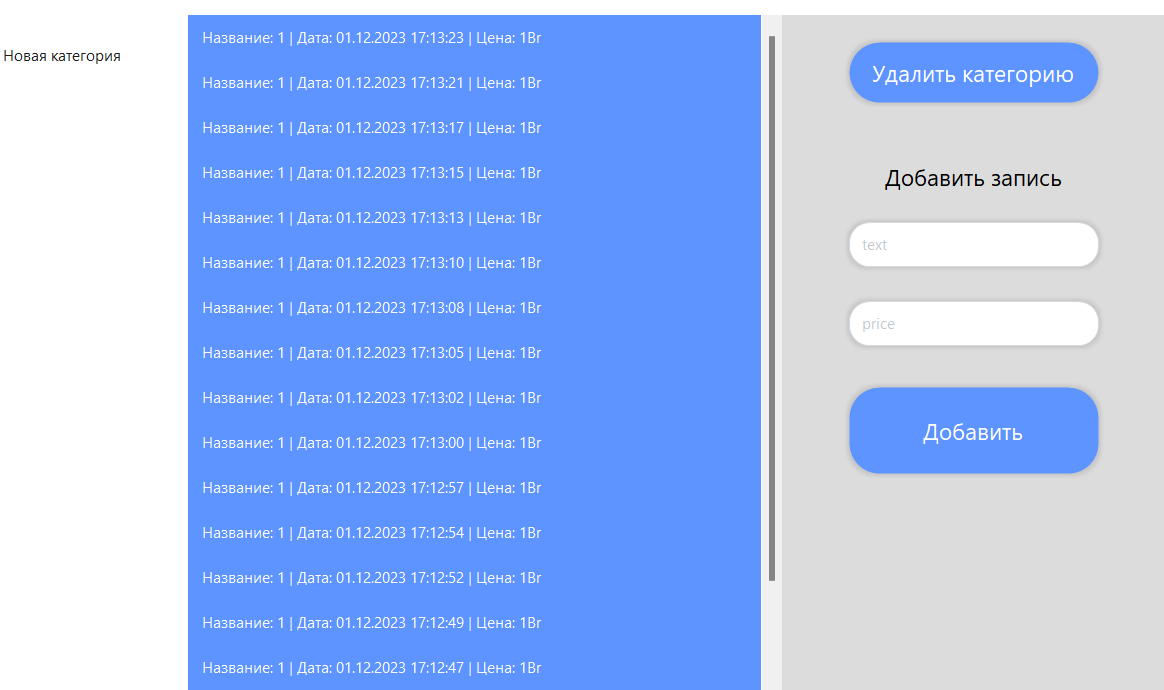


Рис. 7.19. Тест 15

Тест 16:

Тестирование главной формы

Выбираем другую валюту из выпадающего списка

Ожидаемый результат баланс сконвертирован в другую валюту

Результат, представлен на рисунке 7.20.

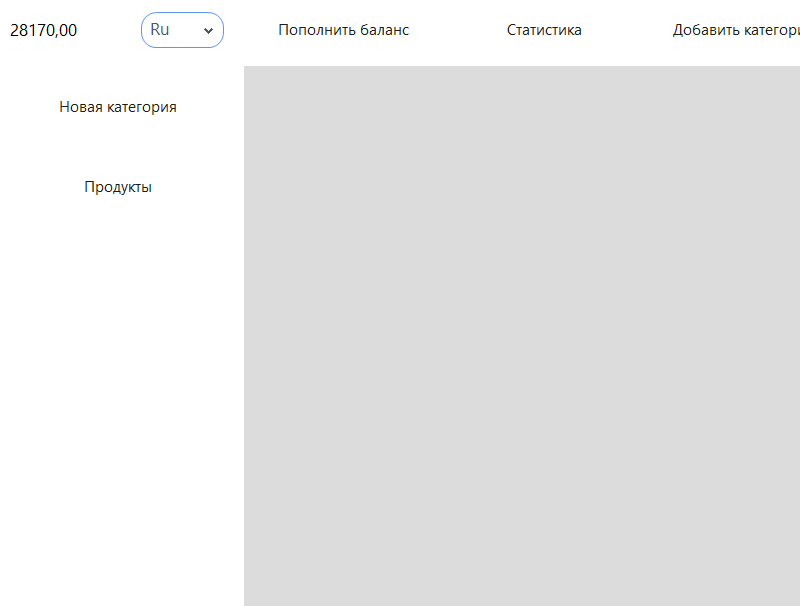


Рис. 7.20. Тест 16

Тест 17:

Тестирование главной формы

Выбираем категорию и жмем удалить

Категория будет удалена

Результат, представлен на рисунке 7.21.

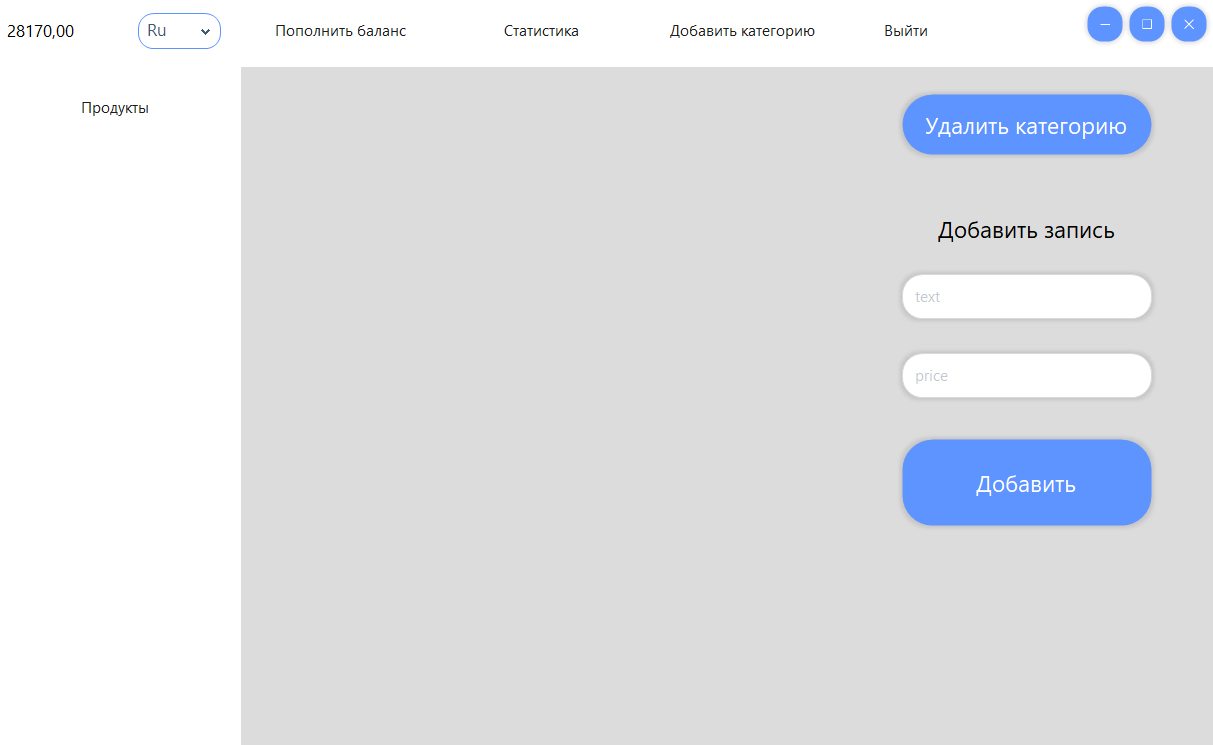


Рис. 7.21. Тест 17

Тест 18:

Тестирование главной формы

Нажать на кнопку «Раскрыть»

Элементы интерфейса будут центрированы

Результат, представлен на рисунке 7.22.

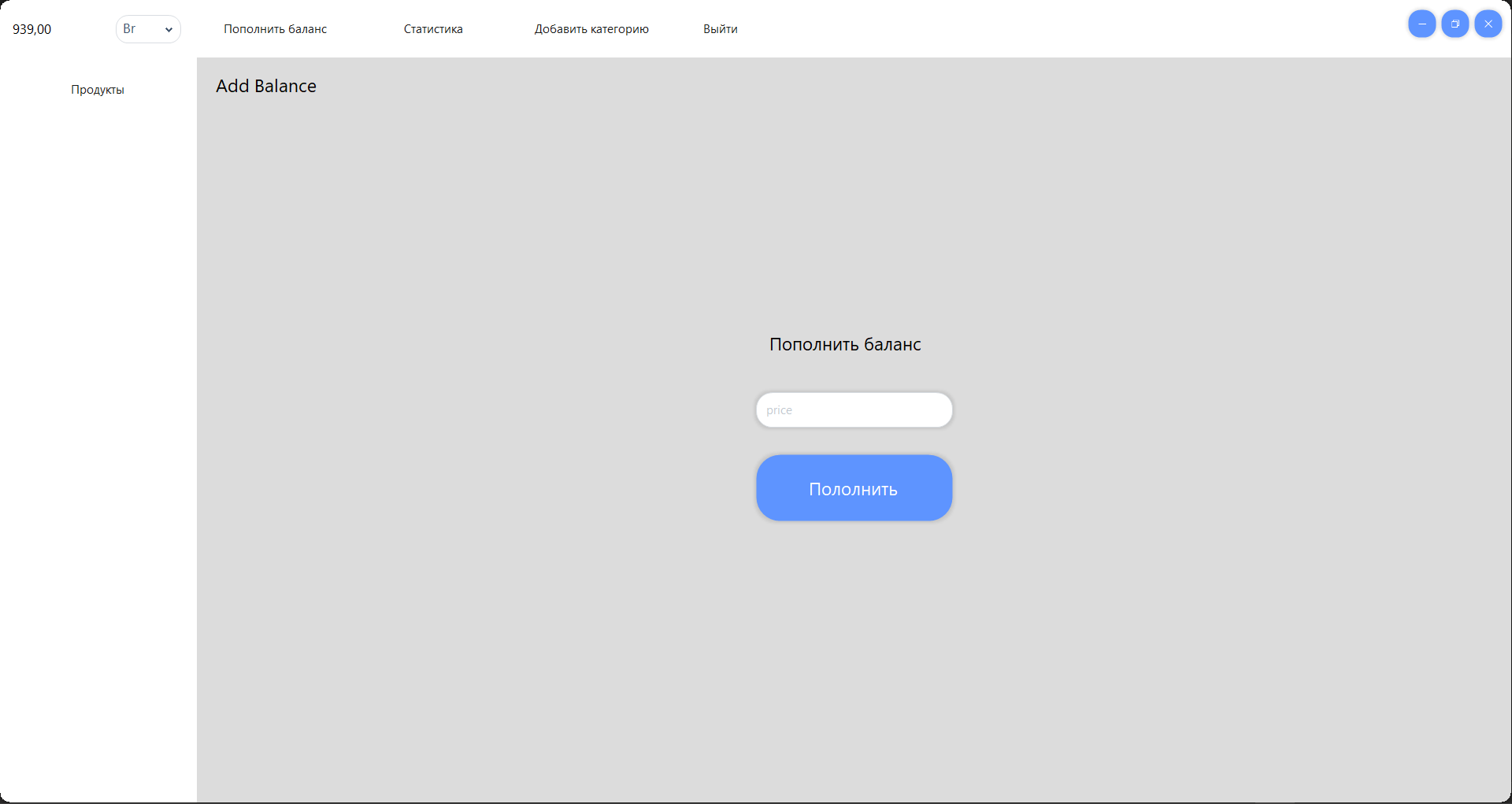


Рис. 7.22. Тест 18

Тест 19:

Тестирование перехода с формы на форму

Ожидаемый результат: при нажатии на кнопку «Выход» переход на форму авторизации

Результат, представлен на рисунке 7.23. – 7.24.

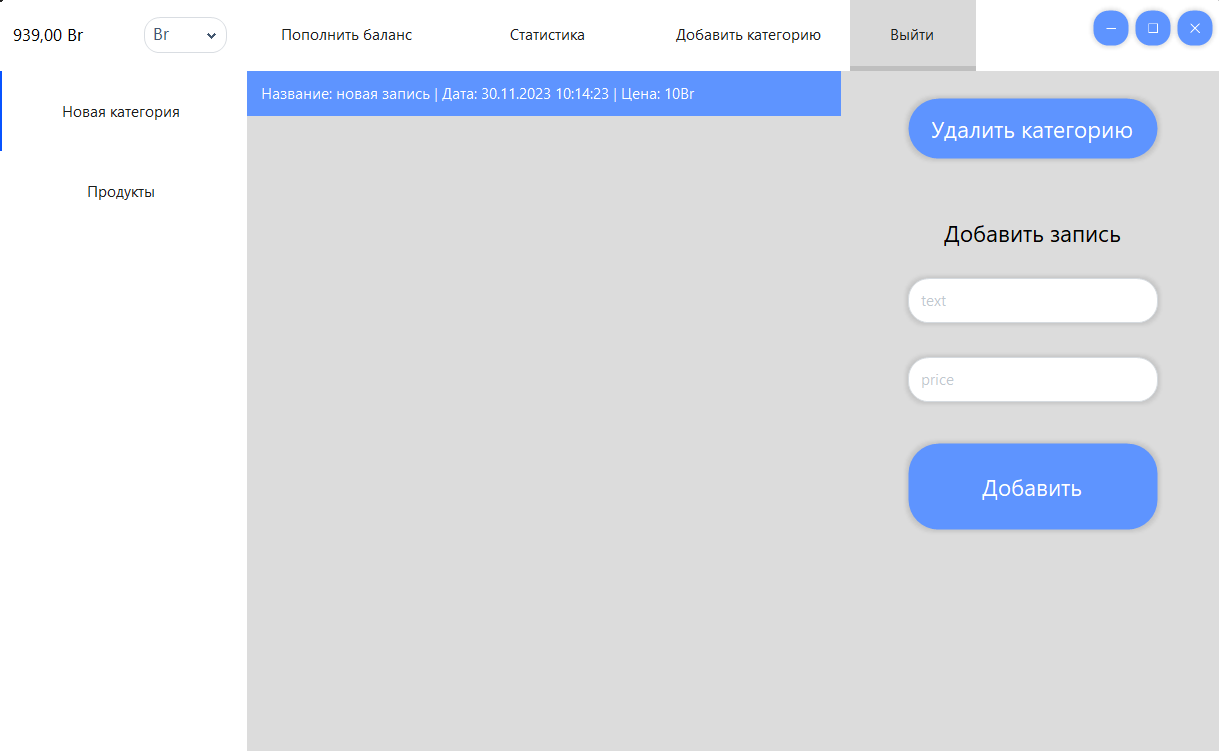


Рис. 7.23. Тест №19

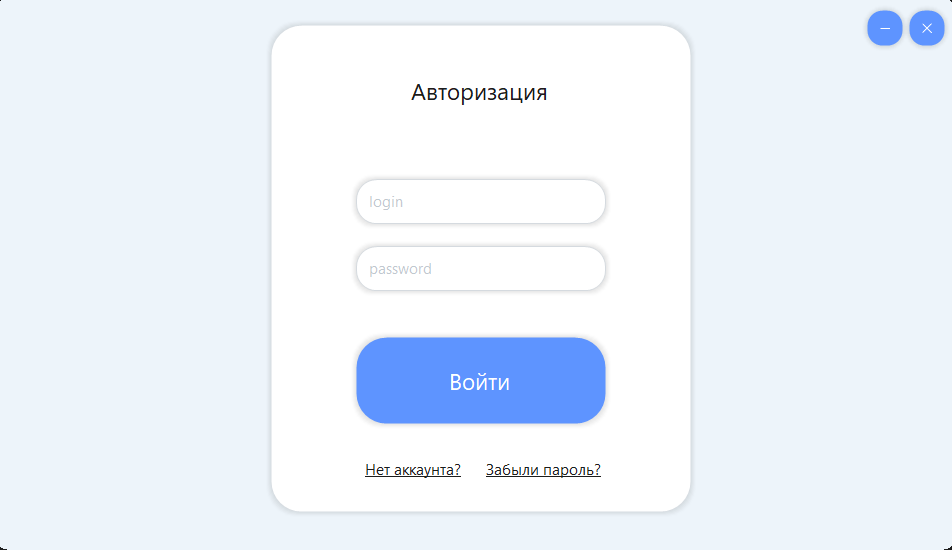


Рис. 7.24. Тест №19

Тест 20:

Тестирование восстановление пароля

Вводим в поле логин otchet

Вводим в поле пароль otchetotchet

Вводим в поле повторить пароль, otchetotchet

Ожидаемый результат: Запрос пин кода для пароля!

Результат, представлен на рисунке 7.25.

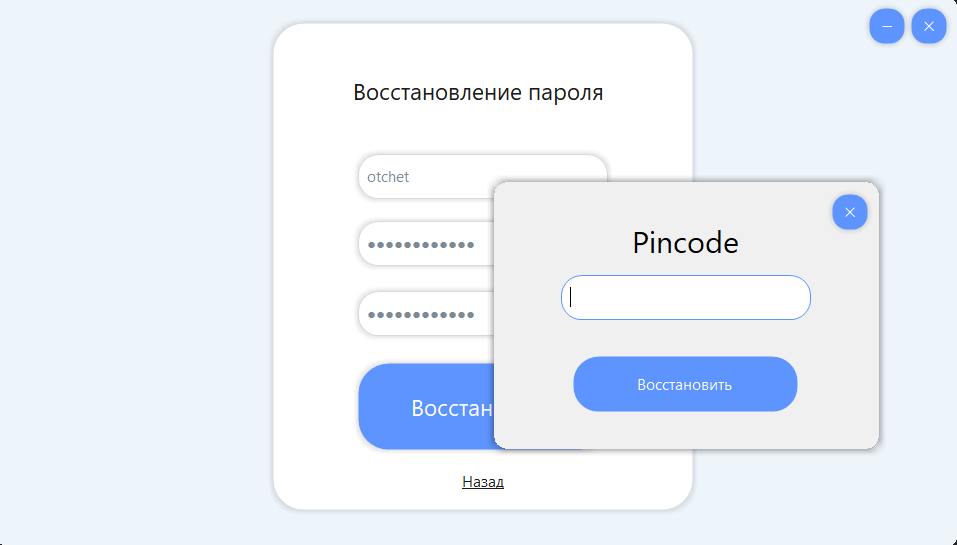


Рис. 7.25. Тест 20

Вводим в поле пинкода 2914

Ожидаемый результат Пароль восстановлен!

Результат, представлен на рисунке 7.26.

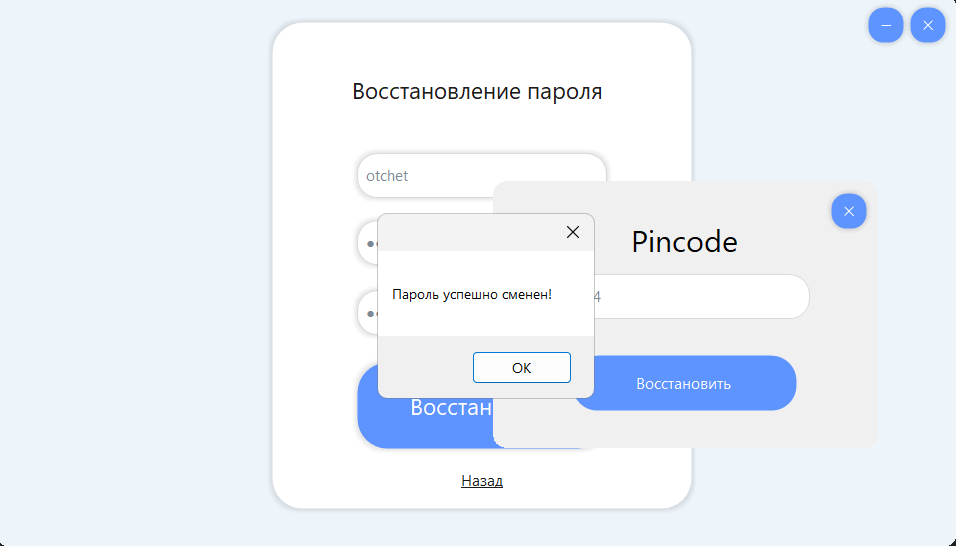


Рис. 7.26. Тест 20

# ЗАКЛЮЧЕНИЕ

В ходе курсового проекта была разработан калькулятор расходов. В приложении реализовано: пополнение баланса, создание категории и записей в ей, вывод статистики.

В качестве выбранного языка программирования используется C#.

В качестве выбранной среды разработки используется Visual Studio.

Программное средство было протестировано, которое прошло успешно.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. Stackoverflow.com [Электронный ресурс] / С#. Режим доступа: <https://stackoverflow.com/> – Дата доступа 02.08.2023.
2. Cyberforum.ru [Электронный ресурс] / С#. Режим доступа: <https://www.cyberforum.ru/> – Дата доступа 20.09.2023.
3. Metanit.com [Электронный ресурс] / C#. Режим доступа: <https://metanit.com/>– Дата доступа 20.10.2023.
4. Learn.microsoft.com [Электронный ресурс] / C#. Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/csharp/>– Дата доступа 20.10.2023.

# ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ

ПО – программное обеспечение;

БД – база данных;

# ПРИЛОЖЕНИЕ 1

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class UC\_AddBalance : UserControl

{

private MainForm mainForm;

public UC\_AddBalance(MainForm mainForm)

{

InitializeComponent();

this.mainForm = mainForm;

}

private void btnAddBalance\_Click(object sender, EventArgs e)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users.FirstOrDefault(u => u.Id == CurrentUser.userId);

if (user != null)

{

try

{

user.Usdbalance += decimal.Parse(txtPrice.Text) / 3;

user.Brbalance += decimal.Parse(txtPrice.Text);

user.Rubalance += decimal.Parse(txtPrice.Text) \* 30;

}

catch

{

MessageBox.Show("Ошибка ввода");

return;

}

dbContext.SaveChanges();

txtPrice.Clear();

UpdateInterface.UpdateBalance(mainForm);

}

}

}

}

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class UC\_AddCategory : UserControl

{

public MainForm mainForm;

public UC\_AddCategory(MainForm mainForm)

{

InitializeComponent();

this.mainForm = mainForm;

}

private void btnAddCategory\_Click(object sender, EventArgs e)

{

ExpenseTrakerDbContext \_db = new ExpenseTrakerDbContext();

var categories = \_db.Categories.Where(c => c.Userid == CurrentUser.userId).ToList();

foreach (var categ in categories)

{

if (categ.Name == txtCategoryName.Text)

{

MessageBox.Show("Категория с таким именем уже существует!");

return;

}

}

Category category = new Category()

{

Userid = CurrentUser.userId,

Name = txtCategoryName.Text

};

\_db.Categories.Add(category);

\_db.SaveChanges();

UpdateInterface.UpdateCategory(mainForm);

txtCategoryName.Clear();

}

}

}

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using Microsoft.EntityFrameworkCore;

using Microsoft.VisualBasic.ApplicationServices;

using Siticone.Desktop.UI.WinForms;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class UC\_Category : UserControl

{

public MainForm mainForm;

public UC\_Statistics uC\_Statistics;

public int CategoryId { get; set; }

public UC\_Category(MainForm mainForm)

{

InitializeComponent();

UpdateInterface.UpdateNotes(mainForm, this);

this.mainForm = mainForm;

}

private void btnDeleteCategory\_Click(object sender, EventArgs e)

{

ExpenseTrakerDbContext \_db = new ExpenseTrakerDbContext();

var notesToDelete = \_db.Notes

.Where(n => n.Categoryid == CategoryId);

\_db.Notes.RemoveRange(notesToDelete);

\_db.SaveChanges();

var deleteCategory = \_db.Categories

.FirstOrDefault(c => c.Userid == CurrentUser.userId && c.Name == this.Name);

\_db.Categories.Remove(deleteCategory);

\_db.SaveChanges();

UpdateInterface.UpdateCategory(mainForm);

}

private void btnCreate\_Click(object sender, EventArgs e)

{

if (txtPrice.Text == string.Empty) return;

ExpenseTrakerDbContext \_db = new ExpenseTrakerDbContext();

var user = \_db.Users.FirstOrDefault(u => u.Id == CurrentUser.userId);

decimal usdprice = 0;

decimal brprice = 0;

decimal ruprice = 0;

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

user.Usdbalance -= decimal.Parse(txtPrice.Text) / 100;

user.Brbalance -= decimal.Parse(txtPrice.Text) / 30;

user.Rubalance -= decimal.Parse(txtPrice.Text);

usdprice = decimal.Parse(txtPrice.Text) / 100;

brprice = decimal.Parse(txtPrice.Text) / 30;

ruprice = decimal.Parse(txtPrice.Text);

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

user.Usdbalance -= decimal.Parse(txtPrice.Text) / 3;

user.Brbalance -= decimal.Parse(txtPrice.Text);

user.Rubalance -= decimal.Parse(txtPrice.Text) \* 30;

usdprice = decimal.Parse(txtPrice.Text) / 3;

brprice = decimal.Parse(txtPrice.Text);

ruprice = decimal.Parse(txtPrice.Text) \* 30;

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

user.Usdbalance -= decimal.Parse(txtPrice.Text);

user.Brbalance -= decimal.Parse(txtPrice.Text) \* 3;

user.Rubalance -= decimal.Parse(txtPrice.Text) \* 100;

usdprice = decimal.Parse(txtPrice.Text);

brprice = decimal.Parse(txtPrice.Text) \* 3;

ruprice = decimal.Parse(txtPrice.Text) \* 100;

}

Note note = new Note()

{

Categoryid = CategoryId,

Text = txtText.Text,

Usdprice = usdprice,

Brprice = brprice,

Ruprice = ruprice,

Date = DateTime.Now.ToUniversalTime(),

};

\_db.Notes.Add(note);

\_db.SaveChanges();

UpdateInterface.UpdateNotes(mainForm, this);

UpdateInterface.CreateInterfacePages(mainForm);

UpdateInterface.UpdateBalance(mainForm);

txtPrice.Clear();

txtText.Clear();

}

}

}

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

namespace ExpenseTracker.Forms

{

public partial class UC\_Statistics : UserControl

{

MainForm mainForm;

public UC\_Statistics(MainForm mainForm)

{

InitializeComponent();

UpdateInterface.UpdateDiagram(mainForm, this);

this.mainForm = mainForm;

}

private void UC\_Statistics\_Load(object sender, EventArgs e)

{

}

private void pnlDiagram\_Paint(object sender, PaintEventArgs e)

{

}

private void btnSetFilter\_Click(object sender, EventArgs e)

{

UpdateInterface.UpdateDiagram(mainForm, this, dateTime1.Value, dateTime2.Value);

}

}

}

using ExpenseTracker.Models;

using Siticone.Desktop.UI.WinForms;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class LoginForm : Form

{

public LoginForm()

{

InitializeComponent();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

string login = txtLogin.Text;

string password = txtPassword.Text;

try

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users

.FirstOrDefault(u => u.Login == login && u.Password == password);

if (user == null)

{

MessageBox.Show("Неправильное имя пользователя или пароль!");

return;

}

CurrentUser.userId = user.Id;

CurrentUser.userLogin = user.Login;

MainForm mainForm = new MainForm();

mainForm.Show();

this.Hide();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при входе: {ex.Message}");

}

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

RegistrationForm registrationForm = new RegistrationForm();

registrationForm.Show();

this.Hide();

}

private void LoginForm\_Load(object sender, EventArgs e)

{

}

private void linkLabel2\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

NewPasswordForm newPasswordForm = new NewPasswordForm();

newPasswordForm.Show();

this.Hide();

}

}

}

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using Siticone.Desktop.UI.WinForms;

using Siticone.Desktop.UI.WinForms.Enums;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

namespace ExpenseTracker.Forms

{

public partial class MainForm : Form

{

public UC\_Statistics uC\_Statistics { get; set; }

public UC\_Category uC\_Category { get; set; }

public MainForm()

{

InitializeComponent();

mainUpdateInterface();

ComboCurrency.SelectedIndex = 1;

}

public void mainUpdateInterface()

{

UpdateInterface.UpdateBalance(this);

UpdateInterface.UpdateCategory(this);

}

private void MainForm\_FormClosing(object sender, FormClosingEventArgs e)

{

Environment.Exit(0);

}

private void siticoneComboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users.FirstOrDefault(u => u.Login == CurrentUser.userLogin);

if (user != null)

{

if (ComboCurrency.SelectedIndex == 0)

{

lblBalance.Text = (user.Rubalance).ToString();

}

else if (ComboCurrency.SelectedIndex == 1)

{

lblBalance.Text = user.Brbalance.ToString();

}

else if (ComboCurrency.SelectedIndex == 2)

{

lblBalance.Text = (user.Usdbalance).ToString();

}

}

UpdateInterface.UpdateCategory(this);

UpdateInterface.UpdateDiagram(this, uC\_Statistics);

}

private void btnExit\_Click(object sender, EventArgs e)

{

this.Hide();

LoginForm loginForm = new LoginForm();

loginForm.Show();

}

}

}

using ExpenseTracker.Models;

using Siticone.Desktop.UI.WinForms;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class NewPasswordForm : Form

{

public NewPasswordForm()

{

InitializeComponent();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

string login = txtLogin.Text;

string password1 = txtPassword1.Text;

string password2 = txtPassword2.Text;

try

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users

.FirstOrDefault(u => u.Login == login);

if (user == null)

{

MessageBox.Show("Неправильное имя пользователя!");

return;

}

if (password1 != password2)

{

MessageBox.Show("Пароли не совпадают!");

return;

}

if (password1.Length < 8 && password1.Length > 20)

{

MessageBox.Show("Пароль должен содержать от 8 до 20 символов!");

return;

}

PinForm pinForm = new PinForm(login, password1);

pinForm.ShowDialog();

LoginForm loginForm = new LoginForm();

loginForm.Show();

this.Hide();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при входе: {ex.Message}");

}

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

}

private void LoginForm\_Load(object sender, EventArgs e)

{

}

private void NewPasswordForm\_FormClosing(object sender, FormClosingEventArgs e)

{

Environment.Exit(0);

}

private void linkLabel1\_LinkClicked\_1(object sender, LinkLabelLinkClickedEventArgs e)

{

RegistrationForm registrationForm = new RegistrationForm();

registrationForm.Show();

this.Hide();

}

}

}

using ExpenseTracker.Models;

using ExpenseTracker.Service;

using Microsoft.EntityFrameworkCore;

using Microsoft.VisualBasic.ApplicationServices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class PinForm : Form

{

public string login;

public string password;

public PinForm(string login, string password)

{

InitializeComponent();

this.login = login;

this.password = password;

}

private void siticoneButton1\_Click(object sender, EventArgs e)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users

.FirstOrDefault(u => u.Login == login

&& u.Password == password

&& u.Pincode == int.Parse(txtPincode.Text));

bool flag = UpdateInterface.CheckPincode(this, int.Parse(txtPincode.Text));

if (flag)

{

MessageBox.Show("Пароль успешно сменен!");

user.Password = password;

dbContext.SaveChanges();

this.Close();

}

else

{

MessageBox.Show("Неверный пинкод!");

}

}

}

}

using ExpenseTracker.Models;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseTracker.Forms

{

public partial class RegistrationForm : Form

{

public RegistrationForm()

{

InitializeComponent();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

string login = txtLogin.Text;

string password1 = txtPassword1.Text;

string password2 = txtPassword2.Text;

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

if (password1 != password2)

{

MessageBox.Show("Пароли не совпадают");

return;

}

var existingUser = dbContext.Users.FirstOrDefault(u => u.Login == login);

if (existingUser != null)

{

MessageBox.Show("Пользователь с таким именем уже существует!");

return;

}

if (txtLogin.Text.Length < 5 || txtLogin.Text.Length > 20)

{

MessageBox.Show("Имя пользователя должно содержать от 5 до 20 символов!");

return;

}

if (txtPassword1.Text.Length < 8 || txtPassword1.Text.Length > 20)

{

MessageBox.Show("Пароль должен содержать от 8 до 20 символов!");

return;

}

Random random = new Random();

int pincode = random.Next(1000, 9999);

MessageBox.Show("Ваш пин код: " + pincode + "\nЗапомните его, для восстановления пароля");

var newUser = new User

{

Login = login,

Password = password1,

Pincode = pincode

};

dbContext.Users.Add(newUser);

dbContext.SaveChanges();

MessageBox.Show("Регистрация успешна!");

LoginForm loginForm = new LoginForm();

loginForm.Show();

this.Hide();

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

LoginForm loginForm = new LoginForm();

loginForm.Show();

this.Hide();

}

private void RegistrationForm\_FormClosing(object sender, FormClosingEventArgs e)

{

Environment.Exit(0);

}

private void txtLogin\_KeyPress(object sender, KeyPressEventArgs e)

{

if (char.IsLetter(e.KeyChar) && e.KeyChar >= 'А' && e.KeyChar <= 'я')

{

e.Handled = true;

}

}

private void txtPassword1\_KeyPress(object sender, KeyPressEventArgs e)

{

if (char.IsLetter(e.KeyChar) && e.KeyChar >= 'А' && e.KeyChar <= 'я')

{

e.Handled = true;

}

}

private void txtPassword2\_KeyPress(object sender, KeyPressEventArgs e)

{

if (char.IsLetter(e.KeyChar) && e.KeyChar >= 'А' && e.KeyChar <= 'я')

{

e.Handled = true;

}

}

}

}

using System;

using System.Collections.Generic;

namespace ExpenseTracker.Models;

public partial class Category

{

public int Id { get; set; }

public int? Userid { get; set; }

public string? Name { get; set; }

public virtual ICollection<Note> Notes { get; set; } = new List<Note>();

public virtual User? User { get; set; }

}

using System;

using System.Collections.Generic;

namespace ExpenseTracker.Models;

public partial class Note

{

public int Id { get; set; }

public int? Categoryid { get; set; }

public string? Text { get; set; }

public decimal? Ruprice { get; set; }

public decimal? Usdprice { get; set; }

public decimal? Brprice { get; set; }

public DateTime Date { get; set; }

public virtual Category? Category { get; set; }

}

using System;

using System.Collections.Generic;

namespace ExpenseTracker.Models;

public partial class User

{

public int Id { get; set; }

public string? Login { get; set; }

public string? Password { get; set; }

public decimal? Rubalance { get; set; }

public decimal? Brbalance { get; set; }

public decimal? Usdbalance { get; set; }

public int Pincode { get; set; }

public virtual ICollection<Category> Categories { get; set; } = new List<Category>();

}

using System;

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace ExpenseTracker.Models;

public partial class ExpenseTrakerDbContext : DbContext

{

public ExpenseTrakerDbContext()

{

}

public ExpenseTrakerDbContext(DbContextOptions<ExpenseTrakerDbContext> options)

: base(options)

{

}

public virtual DbSet<Category> Categories { get; set; }

public virtual DbSet<Note> Notes { get; set; }

public virtual DbSet<User> Users { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

#warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= syntax to read it from configuration - see https://go.microsoft.com/fwlink/?linkid=2131148. For more guidance on storing connection strings, see http://go.microsoft.com/fwlink/?LinkId=723263.

=> optionsBuilder.UseNpgsql("Host=localhost;Port=5432;Database=ExpenseTrakerDB;Username=postgres;Password=rootroot");

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Category>(entity =>

{

entity.HasKey(e => e.Id).HasName("category\_pkey");

entity.ToTable("category");

entity.Property(e => e.Id).HasColumnName("id");

entity.Property(e => e.Name)

.HasMaxLength(50)

.HasColumnName("name");

entity.Property(e => e.Userid).HasColumnName("userid");

entity.HasOne(d => d.User).WithMany(p => p.Categories)

.HasForeignKey(d => d.Userid)

.HasConstraintName("category\_userid\_fkey");

});

modelBuilder.Entity<Note>(entity =>

{

entity.HasKey(e => e.Id).HasName("note\_pkey");

entity.ToTable("note");

entity.Property(e => e.Id).HasColumnName("id");

entity.Property(e => e.Categoryid).HasColumnName("categoryid");

entity.Property(e => e.Ruprice).HasColumnName("ruprice");

entity.Property(e => e.Brprice).HasColumnName("brprice");

entity.Property(e => e.Usdprice).HasColumnName("usdprice");

entity.Property(e => e.Text)

.HasMaxLength(50)

.HasColumnName("text");

entity.Property(e => e.Date).HasColumnName("date");

entity.HasOne(d => d.Category).WithMany(p => p.Notes)

.HasForeignKey(d => d.Categoryid)

.HasConstraintName("note\_categoryid\_fkey");

});

modelBuilder.Entity<User>(entity =>

{

entity.HasKey(e => e.Id).HasName("users\_pkey");

entity.ToTable("users");

entity.Property(e => e.Id).HasColumnName("id");

entity.Property(e => e.Brbalance)

.HasPrecision(20, 2)

.HasDefaultValueSql("0.0")

.HasColumnName("brbalance");

entity.Property(e => e.Login)

.HasMaxLength(255)

.HasColumnName("login");

entity.Property(e => e.Password)

.HasMaxLength(255)

.HasColumnName("password");

entity.Property(e => e.Rubalance)

.HasPrecision(20, 2)

.HasDefaultValueSql("0.0")

.HasColumnName("rubalance");

entity.Property(e => e.Usdbalance)

.HasPrecision(20, 2)

.HasDefaultValueSql("0.0")

.HasColumnName("usdbalance");

entity.Property(e => e.Pincode).HasColumnName("pincode");

});

OnModelCreatingPartial(modelBuilder);

}

partial void OnModelCreatingPartial(ModelBuilder modelBuilder);

}

using ExpenseTracker.Forms;

using ExpenseTracker.Models;

using Microsoft.VisualBasic.Logging;

using Siticone.Desktop.UI.WinForms;

using Siticone.Desktop.UI.WinForms.Enums;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms.DataVisualization.Charting;

namespace ExpenseTracker.Service

{

static public class UpdateInterface

{

public static void UpdateBalance(MainForm mainForm)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users.FirstOrDefault(u => u.Login == CurrentUser.userLogin);

if (user != null)

{

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

mainForm.lblBalance.Text = user.Rubalance.ToString() + " Ru";

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

mainForm.lblBalance.Text = user.Brbalance.ToString() + " Br";

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

mainForm.lblBalance.Text = user.Usdbalance.ToString() + " $";

}

}

}

public static void CreateInterfacePages(MainForm mainForm)

{

UC\_AddCategory uC\_AddCategory = new(mainForm);

mainForm.pnlMain.Controls.Add(uC\_AddCategory);

uC\_AddCategory.Dock = DockStyle.Fill;

mainForm.btnAddCategory.Click += (sender, e) =>

{

uC\_AddCategory.BringToFront();

};

UC\_AddBalance uC\_AddBalance = new(mainForm);

mainForm.pnlMain.Controls.Add(uC\_AddBalance);

uC\_AddBalance.Dock = DockStyle.Fill;

mainForm.btnAddBalance.Click += (sender, e) =>

{

uC\_AddBalance.BringToFront();

};

var element = mainForm.pnlMain.Controls.Find("uC\_Statistics", true).ToList();

foreach (var item in element)

{

if (item != null)

mainForm.pnlMain.Controls.Remove(item);

}

UC\_Statistics uC\_Statistics = new(mainForm);

uC\_Statistics.Name = "uC\_Statistics";

mainForm.pnlMain.Controls.Add(uC\_Statistics);

uC\_Statistics.Dock = DockStyle.Fill;

mainForm.btnStatistics.Click += (sender, e) =>

{

uC\_Statistics.BringToFront();

};

mainForm.uC\_Statistics = uC\_Statistics;

}

public static void UpdateCategory(MainForm mainForm)

{

mainForm.pnlCategory.Controls.Clear();

mainForm.pnlMain.Controls.Clear();

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var categories = dbContext.Categories

.Where(c => c.Userid == CurrentUser.userId).ToList();

if (categories != null)

{

foreach (var category in categories)

{

SiticoneButton siticoneButton = new SiticoneButton();

mainForm.pnlCategory.Controls.Add(siticoneButton);

siticoneButton.Text = category.Name;

siticoneButton.Name = "btn" + category.Name;

siticoneButton.ButtonMode = ButtonMode.RadioButton;

siticoneButton.Size = new(250, 80);

siticoneButton.CustomBorderThickness = new(5, 0, 0, 0);

siticoneButton.CheckedState.CustomBorderColor = Color.FromArgb(8, 83, 255);

siticoneButton.ForeColor = Color.FromArgb(28, 28, 28);

siticoneButton.FillColor = Color.White;

siticoneButton.TextOffset = new(0, 0);

siticoneButton.CheckedState.FillColor = Color.White;

UC\_Category uC\_Category = new(mainForm);

uC\_Category.CategoryId = category.Id;

uC\_Category.Name = category.Name;

mainForm.pnlMain.Controls.Add(uC\_Category);

uC\_Category.Dock = DockStyle.Fill;

siticoneButton.CheckedChanged += (sender, e) =>

{

if (siticoneButton.Checked) uC\_Category.BringToFront();

uC\_Category.Name = category.Name;

UpdateInterface.UpdateNotes(mainForm, uC\_Category);

};

siticoneButton.Dock = DockStyle.Top;

}

}

CreateInterfacePages(mainForm);

}

public static void UpdateDiagram(MainForm mainForm, UC\_Statistics uC\_Statistics, DateTime dateTime1, DateTime dateTime2)

{

uC\_Statistics.pnlDiagram.Controls.Clear();

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var categories = dbContext.Categories

.Where(c => c.Userid == CurrentUser.userId).ToList();

Chart chart = new Chart();

Series series = new Series("Расходы по категориям");

foreach (var category in categories)

{

decimal notesSum = 0;

var notes = dbContext.Notes

.Where(n => n.Categoryid == category.Id

&& n.Date.ToUniversalTime() > dateTime1.ToUniversalTime()

&& n.Date.ToUniversalTime() < dateTime2.ToUniversalTime())

.ToList();

foreach (var note in notes)

{

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

notesSum += (decimal)note.Ruprice;

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

notesSum += (decimal)note.Brprice;

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

notesSum += (decimal)note.Usdprice;

}

}

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Ru", Math.Round(notesSum, 2));

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Br", Math.Round(notesSum, 2));

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Usd", Math.Round(notesSum, 2));

}

}

series.ChartType = SeriesChartType.Pie;

chart.Series.Add(series);

chart.BackColor = Color.Gainsboro;

chart.ChartAreas.Add("ChartArea2");

chart.Legends.Add("Legend1");

chart.Location = new System.Drawing.Point(0, 0);

chart.Size = new Size(530, 700);

uC\_Statistics.pnlDiagram.Controls.Add(chart);

chart.Dock = DockStyle.Top;

}

public static void UpdateDiagram(MainForm mainForm, UC\_Statistics uC\_Statistics)

{

uC\_Statistics.pnlDiagram.Controls.Clear();

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var categories = dbContext.Categories

.Where(c => c.Userid == CurrentUser.userId).ToList();

Chart chart = new Chart();

Series series = new Series("Расходы по категориям");

foreach (var category in categories)

{

decimal notesSum = 0;

var notes = dbContext.Notes

.Where(n => n.Categoryid == category.Id).ToList();

foreach (var note in notes)

{

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

notesSum += (decimal)note.Ruprice;

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

notesSum += (decimal)note.Brprice;

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

notesSum += (decimal)note.Usdprice;

}

}

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Ru", Math.Round(notesSum, 2));

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Br", Math.Round(notesSum, 2));

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

series.Points.AddXY(category.Name + " | " + Math.Round(notesSum, 2) + "Usd", Math.Round(notesSum, 2));

}

}

series.ChartType = SeriesChartType.Pie;

chart.Series.Add(series);

chart.BackColor = Color.Gainsboro;

chart.ChartAreas.Add("ChartArea2");

chart.Legends.Add("Legend1");

chart.Location = new System.Drawing.Point(0, 0);

chart.Size = new Size(530, 700);

uC\_Statistics.pnlDiagram.Controls.Add(chart);

chart.Dock = DockStyle.Top;

}

public static bool CheckPincode(PinForm pinForm, int pincode)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var user = dbContext.Users

.FirstOrDefault(u => u.Login == pinForm.login

&& u.Pincode == pincode);

return user != null;

}

public static void UpdateNotes(MainForm mainForm, UC\_Category uC\_Category)

{

ExpenseTrakerDbContext dbContext = new ExpenseTrakerDbContext();

var notes = dbContext.Notes

.Where(c => c.Categoryid == uC\_Category.CategoryId).ToList();

if (notes != null)

{

uC\_Category.pnlNotes.Controls.Clear();

foreach (var note in notes)

{

SiticoneButton siticoneButton = new SiticoneButton();

siticoneButton.Name = "note" + note.Text;

if (mainForm.ComboCurrency.SelectedIndex == 0)

{

siticoneButton.Text = "Название: " + note.Text + " | Дата: " + note.Date + " | Цена: " + Math.Round((decimal)note.Ruprice, 2) + "Ru";

}

else if (mainForm.ComboCurrency.SelectedIndex == 1)

{

siticoneButton.Text = "Название: " + note.Text + " | Дата: " + note.Date + " | Цена: " + Math.Round((decimal)note.Brprice, 2) + "Br";

}

else if (mainForm.ComboCurrency.SelectedIndex == 2)

{

siticoneButton.Text = "Название: " + note.Text + " | Дата: " + note.Date + " | Цена: " + Math.Round((decimal)note.Usdprice, 2) + "Usd";

}

siticoneButton.TextAlign = HorizontalAlignment.Left;

uC\_Category.pnlNotes.Controls.Add(siticoneButton);

siticoneButton.Dock = DockStyle.Top;

}

}

}

}

}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Окружение: Тестирование проводилось на ноутбуке MSI modern 14 b5m  Windows 11 | | | | | | |
| Identifier  (Идентификатор) | Description  (Описание) | Execution  (Шаги/исполнение) | Input data  (Входные данные)  Expected Result  (Ожидаемый  результат)  Actual result  (Фактический  результат)  Status  (Статус) |  |  |  |