МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Экономики и управления на предприятии природопользования»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

(бакалаврская работа)
по направлению подготовки 09.03.03 Прикладная информатика
(квалификация – бакалавр)

На тему «Разработка информационной системы для управления личными финансами»

Исполнитель Нестеренко Дмитрий Валерьевич

Руководитель к.т.н., Сафонова Татьяна Владимировна

«К защите допускаю»

Руководитель кафедры _____

кандидат экономических наук

Майборода Евгений Викторович

«17» 01 2026 г.

Филиал Российского государственного гидрометеорологического университета в г. Туапсе
НОРМОКОНТРОЛЬ ПРОЙДЕН
«12» 01 2026 г.
Мартынова Е.В.
ПОДПИСЬ РАСШИФРОВКА ПОДПИСИ

Туапсе
2026

ОГЛАВЛЕНИЕ

Введение

Современные средства автоматизации позволяют на качественно новом уровне рассматривать процессы учета любой деятельности и любых ее аспектов, будь то очередь заказов, покупки, финансы и т.д. Одним из приложений средств автоматизации является учет финансовых данных, позволяющий систематизировать информацию о финансовых потоках, анализировать ее и принимать верные решения для экономии / оптимизации распределения денежных средств. Таким образом, применение автоматизированных средств учета личных финансов и расходов является актуальной задачей.

Объектом исследования в настоящей работе является процесс учета и управления личным бюджетом, предметом исследования – информационная система для автоматизации учета и управления личным бюджетом.

Целью настоящей работы является разработка программного обеспечения для автоматизации процессов ведения учета личного бюджета.

Для достижения поставленной цели в работе планируется решить следующие задачи:

– выполнить анализ предметной области;

– выполнить постановку задачи на разработку автоматизированной информационной системы;

– выбрать средства для проектирования и разработки информационной системы;

– выполнить проектирование и разработку базы данных информационной системы;

– выполнить разработку прикладного программного обеспечения информационной системы;

– оценить экономическую эффективность разработанной информационной системы.

В ходе работы применены такие методы исследования, как: моделирование, синтез, а также средства и методологии проектирования и разработки:

– методология моделирования программных систем – UML (спецификация 2.5.1);

– методология проектирования баз данных в нотациях IDEF1X и Питера-Чена;

– CASE-система Enterprise Architect 15.0;

– среда разработки IDE MS Visual Studio 2019;

– СУБД MS Access 2016;

– язык программирования высокого уровня C# .NET, технология разработки Windows Forms, технология OLE DB / .NET Framework.

Настоящая работа состоит из введения, трех разделов, списка литературы из 35 источников, заключения и 1 приложения. Работа представлена на 64 страницах и содержит 25 рисунков и 12 таблиц.

В первом разделе производится анализ предметной области, описывается экономическая сущность задачи, выполняется формализованная постановка задачи на разработку информационной системы. Второй раздел посвящен проектированию и разработки информационной системы (программное, информационное, техническое и технологическое обеспечение). В данном разделе описано руководство пользователя информационной системы. В последнем разделе приводится расчет экономической эффективности разработанной информационной системы. В приложении приведен исходный код программы.

1 Аналитическая часть

1.1 Анализ предметной области

Анализ предметной области проводится с целью определения основных процессов, происходящих в предметной области, уточнения глоссария, получения общей картины о процессе, его атрибутах, участниках. Лучшим способом описать предметную область – применить одну из нотаций моделирования бизнес-процессов [3]. При разработке информационных систем для автоматизации процессов в предметной области наиболее правильным будет описать процессы в одной из нотаций функционального моделирования [9]. Одной из таких нотаций является IDEF0. Данная нотация за долгое время использования утвердила себя как наиболее эффективная, простая в интерпретации и использовании. Данная нотация опирается на соответствующий стандарт [10] функционального моделирования и включает описание всех атрибутов процессов: I (входы), O (выходы), M (механизмы / средства / исполнители), C (управляющие регламенты и ограничения).

Для исследуемого процесса управления личными финансами можно указать следующие атрибуты. Входной информацией будут являться:

– поступаемые из разных источников денежные средства;

– планы (желания), направленные на возможное приобретение любых материальных благ за деньги;

– события и обстоятельства, которые требуют расходов – операционные, нерегулярные события;

– обстоятельства, определяемые внешними факторами (политика, новостные прогнозы, аналитика различных финансовых институтов).

Результатами процесса управления личными финансами могут быть расходы денежных средств, а также различная сводная информация, представляющая отчёты по расходам и доходам, перспективам реализации планов.

Процесс регулируется следующими факторами и обеспечением:

– договоры по кредитным обязательствам, трудовым соглашениям, расписки и другие документы, определяющие порядок поступления или выплат денежных средств;

– ценовая политика по объектам капитальных и нерегулярных расходов;

– программы и предложения банков, кредитных организаций и брокерских бирж;

– различные факторы риска, субъективно определяемые индивидуально;

– индивидуальные финансовые потребности;

– требования к подготовке отчетных форм и документов, определяемые индивидуально субъектом.

Средствами процесса управления личными финансами являются: сам субъект, используемые им средства и инструменты финансового анализа. Таким образом, на рисунке 1.1 приведена диаграмма исследуемого процесса управления личными финансами в терминах и обозначениях нотации IDEF0 (контекст).

Рисунок 1.1 – Контекстная диаграмма процесса управления личными финансами в нотации IDEF0

Стандарт IDEF0 поддерживает неограниченную декомпозицию блоков до требуемого уровня (до подпроцессов, которые в рамках рассматриваемой предметной области являются тривиальными, атомарными, не имеющими смысла в декомпозиции) [24]. Декомпозиция процессов позволяет детализировать процессы и уточнять необходимые функции с нужной степенью глубины. Так, на рисунке 1.2 приведена диаграмма декомпозиции процесса управления личными финансами в нотации IDEF0.

Рисунок 1.2 – Диаграмма декомпозиции процесса управления личными финансами в нотации IDEF0

Таким образом, процесс управления личными финансами сводится к следующим подпроцессам:

- прием поступаемых денежных потоков;
- планирование бюджета (расходов);
- осуществление расходов;
- подведение итогов в нужной форме отчетности.

Прием поступлений сводится к следующим подпроцессам:

– получение дохода из регулярных источников, к которым может относиться заработная плата, получение дивидендов по акциям, получение оплаты по договорам аренды и т. д.;

– получение дохода по разовым событиям, например: выполнение долговой расписки по отношению к субъекту, выигрыш в лотерею, продаже товара в частном порядке;

– получение доходов из инвестиционных портфелей или от реализации инвестиционных программ – такие доходы могут быть как разовыми, так и периодическими, но нерегулярными;

– финансовый анализ и прогнозирование, осуществляемое на основании выбранных финансовых стратегий, внешних условий, консолидации предложений различных банков, кредитных и брокерских фирм, но с учетом основных факторов риска.

На рисунке 1.3 приведена диаграмма декомпозиции подпроцесса приема поступлений на основании описанных выше положений.

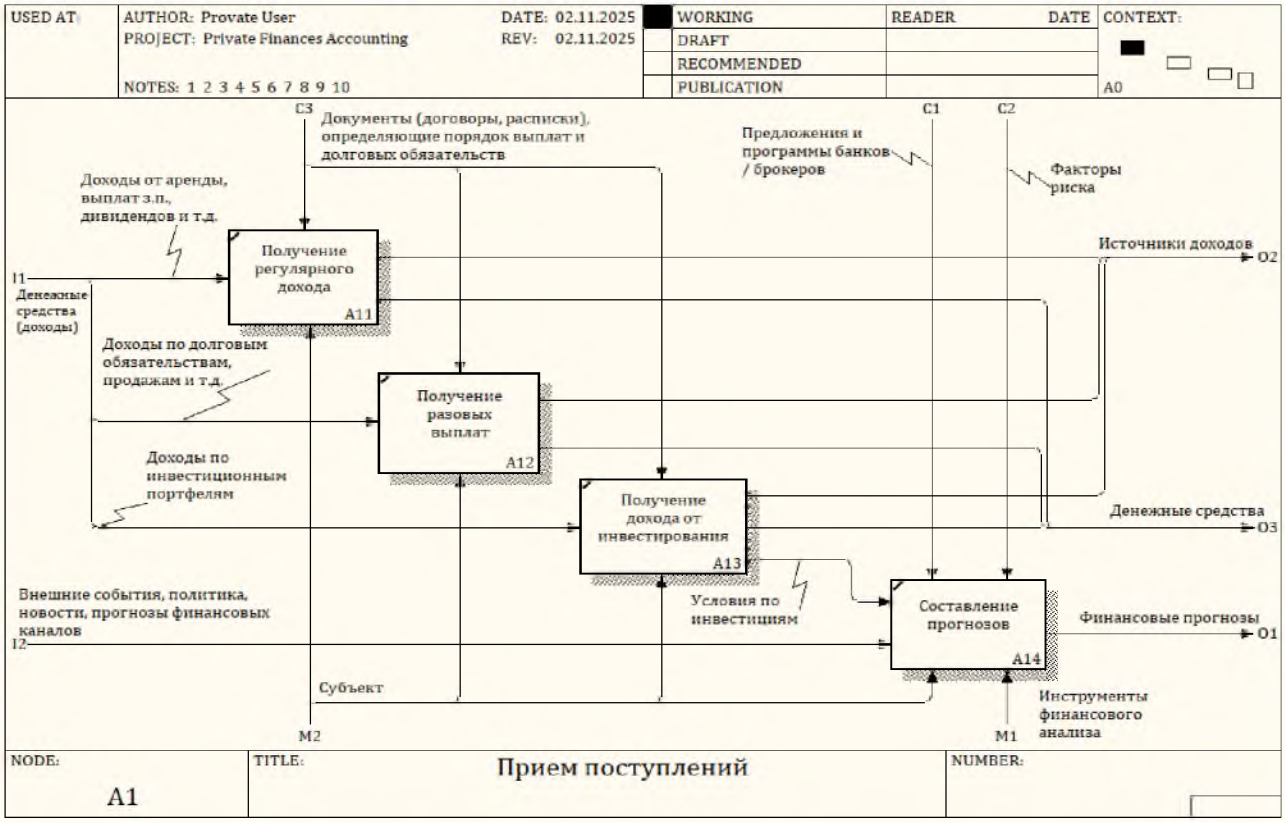

Рисунок 1.3 – Диаграмма декомпозиции подпроцесса приема поступлений

Подпроцесс планирования бюджета включает такие задачи, как:

– консолидация источников доходов и расходов;

– разработка ограничений финансов, которые могут быть потрачены на реализацию планов и инвестиции, определение размеров свободных денежных средств с учетом всех обязательств и данных по доходам и расходам, финансовых прогнозов, регулярных потребностей;

– планирование капитальных и нерегулярных расходов;

– планирование инвестирования.

Так, на рисунке 1.4 приведена диаграмма декомпозиции подпроцесса планирования бюджета.

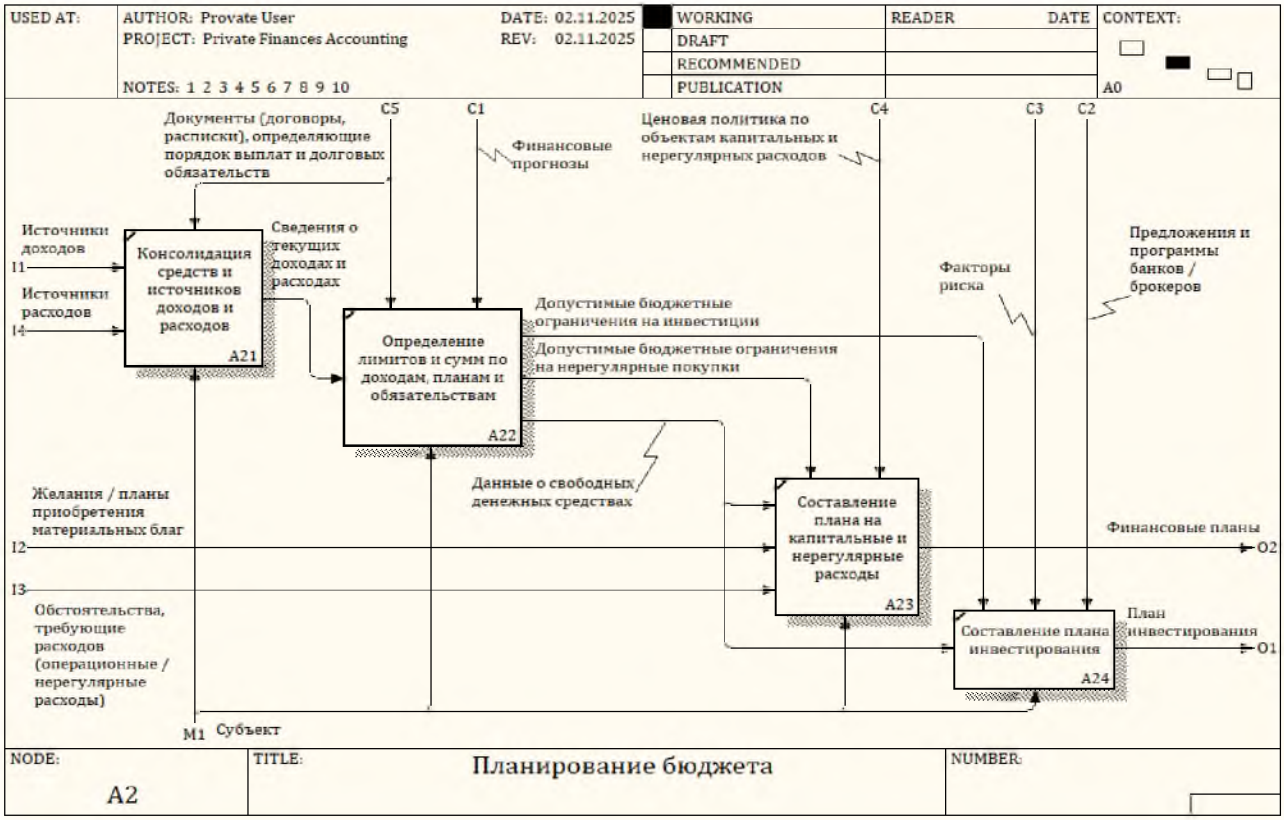

Рисунок 1.4 – Диаграмма декомпозиции подпроцессапланирования бюджета

Процесс осуществления расходов заключается в утверждении сумм на различные виды расходов:

– повседневные (регулярные) потребности (продукты питания, транспорт, топливные расходы, обслуживание автомобиля, расходы на обучение, поддержание качества жизни и быта и т. д.);

– расходы по договорным обязательствам (кредиты, договоры аренды, платежи по договорам коммунальных услуг, обеспечение расписок и других видов долговых обязательств);

– расходы на инвестиционные программы;

– расходы на нерегулярные и капитальные покупки (недвижимость, автомобили, одежда, бытовая техника, и т. д.), а также непредвиденные расходы (ремонт автомобиля, оплата штрафов, судебных и других издержек, незапланированных покупок и т. д.).

На рисунке 1.5 приведена соответствующая диаграмма декомпозиции подпроцесса осуществления расходов.

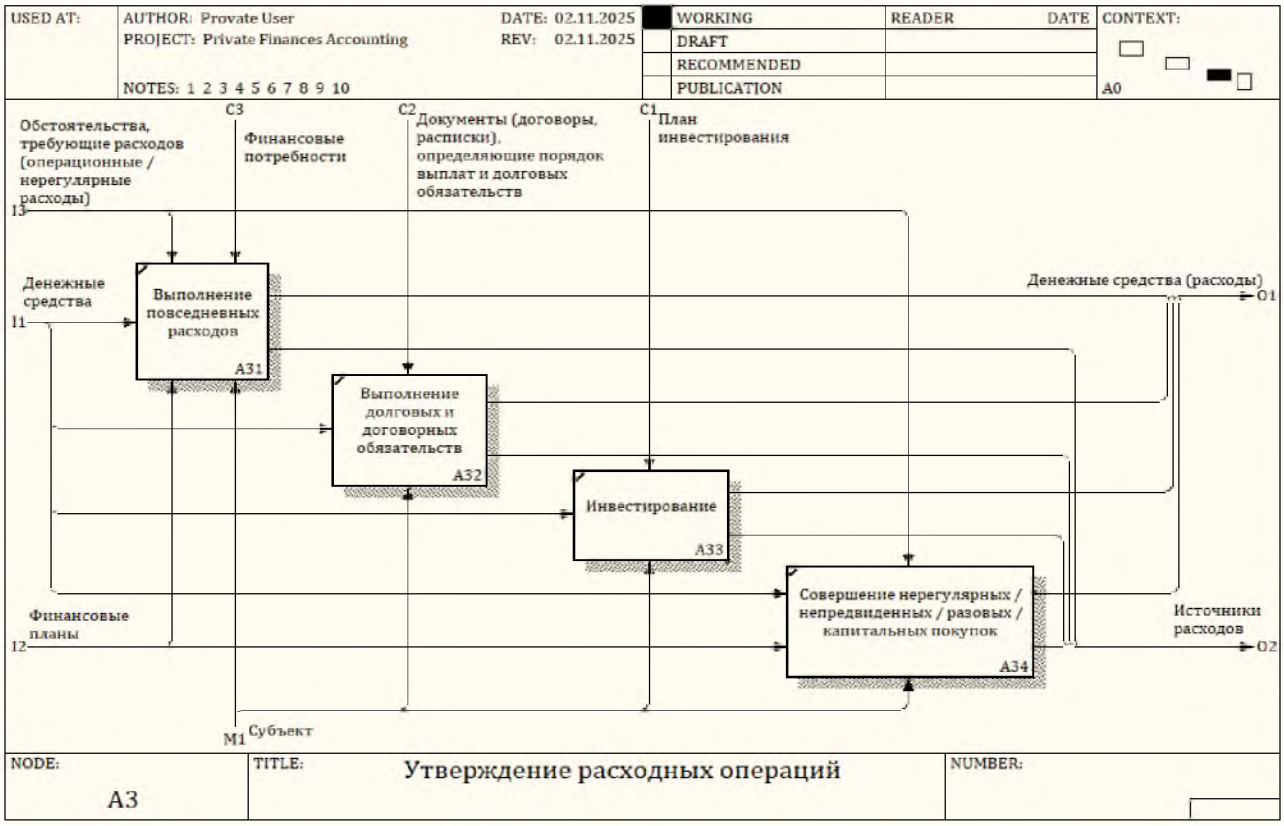

Рисунок 1.5 – Диаграмма декомпозиции подпроцесса

Таким образом, разработанная модель в нотации IDEF0 описывает предметную область с точки зрения функционального моделирования. Это позволяет выполнить анализ задач, решаемых в процессе управления личными финансами, определить основные подпроцессы, нуждающиеся в

автоматизации, и сформировать требования к информационной системе.

1.2 Обоснование выбора задачи

Учет при распределении семейного бюджета играет ключевую роль в обеспечении финансового благополучия и стабильности любой современной семьи[8]. Планирование и учет расходов позволяет определить наиболее важные статьи затрат и сконцентрироваться на их оптимизации. Учет доходов помогает контролировать поступления и определять возможности для дополнительного заработка. Ведение учета бюджета помогает выявить и устранить неэффективные траты, что позволяет сэкономить средства и направить их на долгосрочные цели или непредвиденные расходы. Кроме того, систематизация учета бюджетных средств семьи позволяет:

– предотвратить долги и кредитные обязательства, которые могут негативно повлиять на финансовое положение семьи;

– определить, где можно сократить расходы и направить сэкономленные средства на инвестиции или образование детей;

– создать финансовый резерв на случай непредвиденных обстоятельств или потери работы;

– выявить и использовать возможности для увеличения семейного дохода, например, путем инвестирования или получения дополнительных источников дохода.

Важность учета при распределении семейного бюджета очевидна. При этом наличие аналитических инструментов возможно только при использовании соответствующих средств, которыми, как правило, являются компьютерные программы, которые берут на себя всю работу по подготовке и анализу данных и построению аналитических отчетов [22].

В рамках решения задачи учета семейного бюджета важно контролировать такие параметры, как:

– доходы семьи;

– расходы семьи;

– долги и задолженности.

Имея объемы указанной информации, можно систематизировать траты, источники получения доходов и анализировать всю информацию на предмет синтеза новых решений по оптимизации бюджета.

Для эффективного учета семейного бюджета основными объектами, подлежащими систематизации, являются:

– счета;

– доходы;

– расходы;

– долги;

– планируемые доходы и покупки;

– категоризация расходов.

Кроме этого, для автоматизации процессов учета важно учитывать периодические процессы, влияющие на состояние бюджета. К периодическим процессам можно, например, отнести:

– получение заработной платы;

– получение доходов от аренды недвижимости;

– оплата аренды, коммунальных услуг;

– оплата кредиторских обязательств;

– другие.

1.3 Разработка требований к информационной системе

Помимо средств учета для управления семейным бюджетом важно также иметь средства консолидации и анализа данных, позволяющие строить различные отчеты о состоянии счетов, графики расходов и доходов и т. д. Для реализации вышеперечисленных средств применяются автоматизированные информационные системы. Консолидировав основные положения по учету и

анализу процессов в семейном бюджете, можно разработать формальную модель требований к информационной системе. Такую модель можно представить в нотации UML в виде диаграммы вариантов использования (прецедентов) [31].Диаграмма вариантов использования информационной системы «Семейный бюджет» приведена на рисунке 1.6.

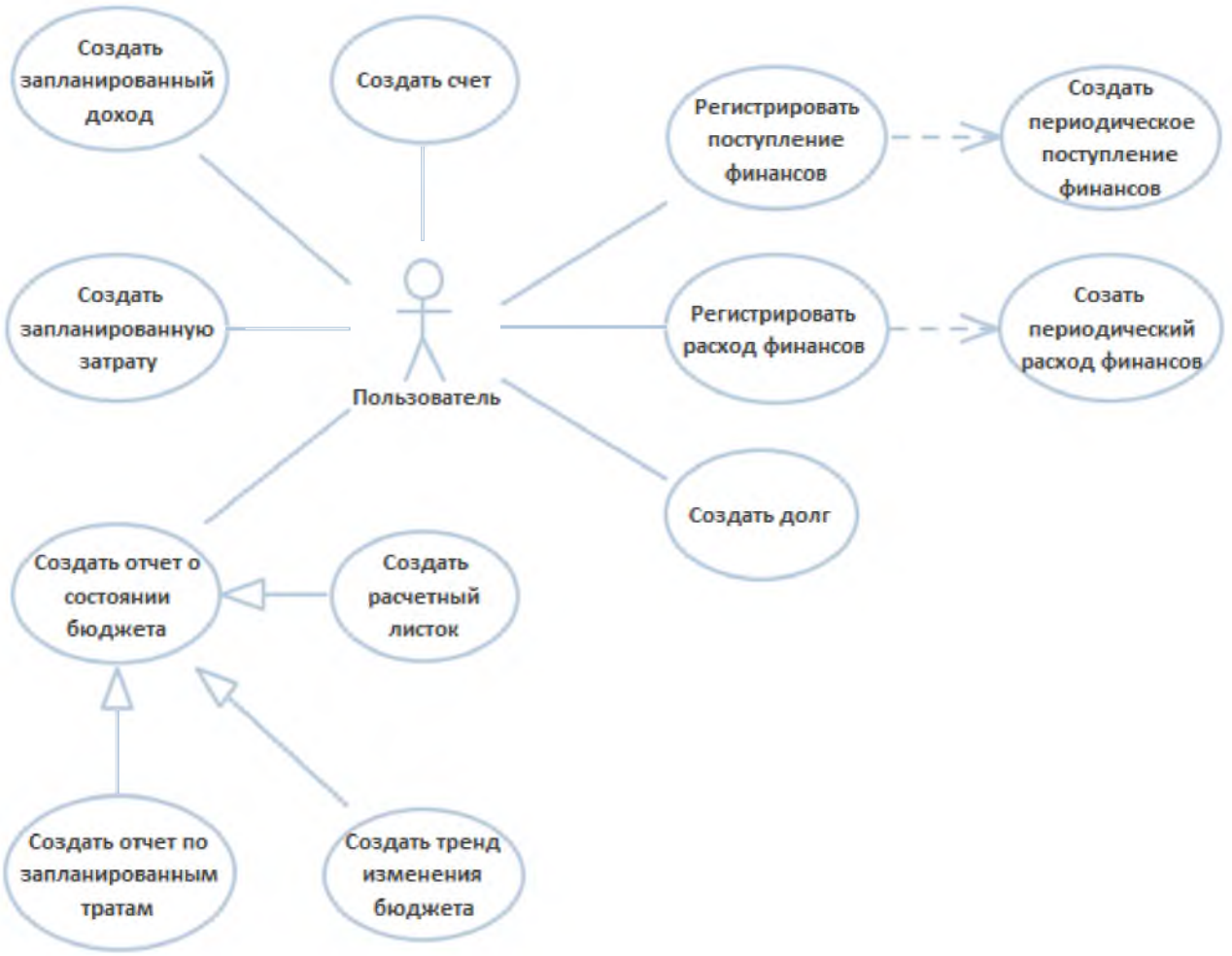

Рисунок 1.6 – Диаграмма вариантов использования информационной системы «Семейный бюджет»

Таким образом, из представленной диаграммы видно, что информационная система – однопользовательская, а каждый вариант использования представляет собой отдельную функцию.

В качестве дополнительных требований можно выделить требования к интерфейсам информационной системы. Так, для удобства работы

пользователя информационная система должна иметь следующие интерфейсы:

– форма главного меню;

– форма справочника категоризации доходов и расходов;

– форма для учета счетов;

– форма для просмотра, ввода, редактирования и удаления данных о доходах;

– форма для просмотра, ввода, редактирования и удаления данных о расходах;

– форма для просмотра, ввода, редактирования и удаления данных о долгах;

– форма для просмотра, ввода, редактирования и удаления данных о планируемых покупках и доходах;

– форма для просмотра, ввода, редактирования и удаления данных о периодических покупках и доходах;

– форма отчета сводного баланса семейного бюджета в текстовом и графическом виде.

Выводимые текстовые сообщения должны быть представлены только на русском языке. Ввод информации осуществляется с помощью стандартных средств – таких, как: клавиатура, манипулятор типа «мышь».

В информационной системе должна осуществляться проверка всех вводимых данных.

Требования к программному обеспечению:

– платформа MicrosoftWindowsVista/7/8/8.1/10;

– файл-серверная СУБД;

– портативность;

– низкая ресурсоемкость и высокая производительность.

Взаимодействие пользователей с системой должно осуществляться посредством визуального пользовательского графического интерфейса. Интерфейс должен быть понятным и удобным, не должен быть перегружен

графическими элементами и должен обеспечивать быстрое отображение экранных форм. Навигационные элементы должны быть выполнены в удобной для пользователя форме. Средства редактирования информации должны удовлетворять принятым соглашениям в части использования функциональных клавиш, режимов работы, поиска, использования оконной системы. Ввод-вывод данных системы, прием управляющих команд и отображение результатов их исполнения должны выполняться в интерактивном режиме. Интерфейс должен соответствовать современным эргономическим требованиям и обеспечивать удобный доступ к основным функциям и операциям системы. Интерфейс системы должен быть выполнен в соответствии с основными принципами проектирования пользовательских интерфейсов Нильсена-Молиха [21].

При реализации системы должны применяться следующие языки программирования / моделирования высокого уровня:

– объектно-ориентированный язык программирования C#;

– нотация моделирования программного обеспечения – UML, спецификация 2.5.1 [33];

– CASE-система для выполнения технического проекта на UML – EnterpriseArchitect 15.0 [32].

Для реализации диалога системы с пользователями должен применяться графический оконный пользовательский интерфейс.

Язык элементов пользовательского интерфейса – русский.

Язык комментариев в программном коде – русский / английский.

Язык именования переменных в программном коде, компонентов проекта ПО, таблиц и полей БД – английский.

Программный код на языке C# должен быть выполнен в соответствии со стандартом ECMA-334 [30].

Требования к техническому обеспечению:

– клавиатура, манипулятор типа «мышь»;

– монитор с частотой обновления кадров 60 Гц или выше;

– процессор от 2000 МГц и выше;

– объем ОЗУ не менее 256 МБ, не считая требований, предъявляемых операционной системой;

– видеоадаптерDirectX 9+ совместимый;

– свободное место на жестком диске – не менее 1 ГБ, не считая требований, предъявляемых операционной системой.

2 Проектная часть

2.1 Информационное обеспечение задачи

2.1.1 Описание информационных объектов, концептуальная модель данных

Входными данными проектируемой информационной системы ИС «Семейный бюджет» являются:

- информация о текущих, плановых и периодических расходах;
- информация о текущих, плановых и периодических доходах;
- информация о счетах;
- информация о категориях
- информация о долгах;

Выходной информацией являются формы отчетов, в которых суммируются все данные по расходам и доходам и выводится общий баланс семейного бюджета. Данные представляются в текстовом (табличном) и графическом видах.

В качестве сущностей концептуальной информационной модели системы можно выделить следующие основные информационные объекты:

- Доход;
- Расход;
- Долг;
- План Расхода;
- План Дохода;
- Счет;
- Категория;
- Периодическая Операция.

Так, в соответствии с идентифицированными информационными объектами на рисунке 2.1 приведена диаграмма концептуальной модели данных информационной системы «Семейный бюджет» в нотации Питера-Чена [26].

Рисунок 2.1 – Диаграмма концептуальной модели данных информационной системы «Семейный бюджет»

Таким образом, из приведенной на рисунке 2.1 модели видно, что все информационные объекты наследуются от одного общего объекта,

характеризующегося атрибутами: идентификатор, наименование, описание. Также, на схеме видны еще связи наследования у информационных объектов «Операция» и «План».

При составлении даталогической и физической моделей данных все связи наследования должны быть преобразованы, а соответствующие логические объекты и таблицы должны содержать необходимые наборы полей, в том числе и унаследованные от таблиц, расположенных по иерархии выше[6].

2.1.2 Инфологическая модель данных

В рамках логического проектирования базы данных определяются табличные сущности и связи между ними. Таблицы могут быть справочными и оперативными. Справочными являются таблицы, содержимое которых меняется редко. Такие таблицы в основном используются как ссылочные в оперативных. Оперативные таблицы содержат записи, которые могут изменяться сравнительно часто [29].

Основываясь на построенной ранее концептуальной модели, можно выделить следующий набор таблиц базы данных для информационной системы «Семейный бюджет»:

- Счет (справочная);
- Категория (справочная);
- Долг (оперативная);
- Расход (оперативная);
- Доход (оперативная);
- План Расход (оперативная);
- План Доход (оперативная);
- Периодическая Операция (оперативная).

В каждой из этих таблиц выделяются атрибуты, определяются ключевые поля, устанавливаются связи между ними. Так, в таблице 2.1 приведена спецификация атрибутов сущности (таблицы) «Счет».

Таблица 2.1 – Спецификация таблицы «Счет»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Номер Счета	Текст	–	Номер карты / счета

В таблице 2.2 приведена спецификация атрибутов сущности (таблицы) «Категория».

Таблица 2.2– Спецификация таблицы «Категория»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание

В таблице 2.3 приведена спецификация атрибутов сущности (таблицы) «Долг».

Таблица 2.3 – Спецификация таблицы «Долг»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
ДатаСоздания	Дата	–	Дата создания долга
Срок	Дата	–	Требуемая максимальная дата уплаты долга
ДатаОплаты	Дата	–	Дата фактической уплаты долга
Сумма	Денежный	–	Сумма долга
Направление	Логический	–	Направление долга (1 – расход, 0 – приход)

В таблице 2.4 приведена спецификация атрибутов сущности (таблицы) «Расход».

Таблица 2.4 – Спецификация таблицы «Расход»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Дата	Дата	–	Дата расходной операции
Сумма	Денежный	–	Сумма расхода
Аккаунт_ID	Числовой (ссылка)	ВК	Идентификатор счета, с которым связана расходная операция
Категория_ID	Числовой (ссылка)	ВК	Идентификатор категории, к которой относится указанная расходная операция

В таблице 2.5 приведена спецификация атрибутов сущности (таблицы) «Доход».

Таблица 2.5 – Спецификация таблицы «доход»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Дата	Дата	–	Дата приходной операции
Сумма	Денежный	–	Сумма поступления
Аккаунт_ID	Числовой (ссылка)	ВК	Идентификатор счета, с которым связана приходная операция
Категория_ID	Числовой (ссылка)	ВК	Идентификатор категории, к которой относится приходная операция

В таблице 2.6 приведена спецификация атрибутов сущности (таблицы) «ПланРасход».

Таблица 2.6 – Спецификация таблицы «ПланРасход»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Сумма	Денежный	–	Сумма планируемой расходной операции

В таблице 2.7 приведена спецификация атрибутов сущности (таблицы) «ПланДоход».

Таблица 2.7 – Спецификация таблицы «ПланДоход»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Дата	Дата	–	Дата планируемого пополнения
Сумма	Денежный	–	Сумма планируемого пополнения счета

В таблице 2.8 приведена спецификация атрибутов сущности (таблицы) «ПериодическаяОперация».

Таблица 2.8 – Спецификация таблицы «ПериодическаяОперация»

Атрибут	Тип	Ключ	Назначение
ID	Счетчик	ПК	Уникальный идентификатор
Наименование	Текст	–	Наименование
Описание	Текст	–	Необязательное описание
Сумма	Денежный	–	Сумма периодической денежной операции
Направление	Логический	–	Направление операции: 1 – операция расхода; 0 – операция прихода.
Период	Числовой	–	Период повторения периодической операции (в месяцах)
Аккаунт_ID	Числовой (ссылка)	ВК	Идентификатор счета, с которым связана периодическая операция (счет поступления или счет списания)
Категория_ID	Числовой (ссылка)	ВК	Идентификатор категории, к которой относится периодическая операция
ДатаНачала	Дата	–	Дата, с которой начинается выполнение периодической операции
ДатаЗавершения	Дата	–	Дата, по истечении которой периодическая операция прекращает выполняться

На рисунке 2.2 приведена информационно-логическая модель данных информационной системы «Семейный бюджет», на которой учтены все информационные объекты (сущности), их атрибуты и связи между ними, определенные спецификациями, описанными в таблицах 2.1-2.8. Приведенная информационно-логическая модель данных системы выполнена в нотации Мартина [5].

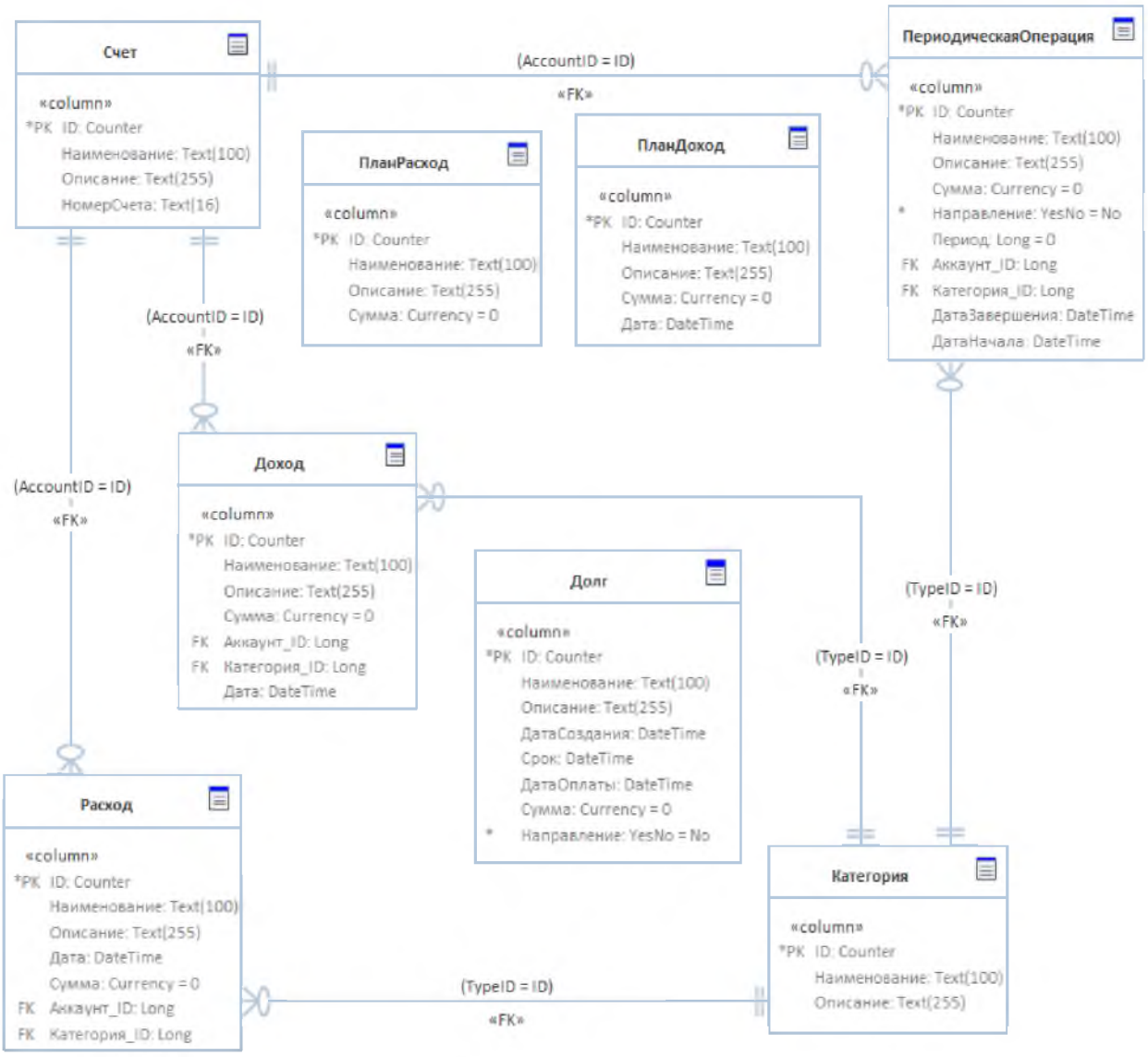

Рисунок 2.2 – Информационно-логическая модель данных информационной системы «Семейный бюджет»

Приведенная на рисунке 2.2 модель находится, как минимум, в третьей нормальной форме [13] с учетом условий и допущений, принятых для

конкретной предметной области. Третьей нормальной формы достаточно для большинства неспециальных бизнес-приложений [27].

2.1.3 Физическое проектирование базы данных

Для реализации базы данных информационной системы необходимо преобразовать информационно-логическую модель в физическую[16]. Для этого необходимо конкретизировать типы данных относительно выбранной СУБД, переименовать атрибуты в поля соответствующих таблиц (рекомендуется использовать латиницу). В качестве СУБД для информационной системы «Семейный бюджет» выбрана файл-серверная СУБД MSAccess. На рисунке 2.3 приведена схема данных в нотации IDEF1X[28].

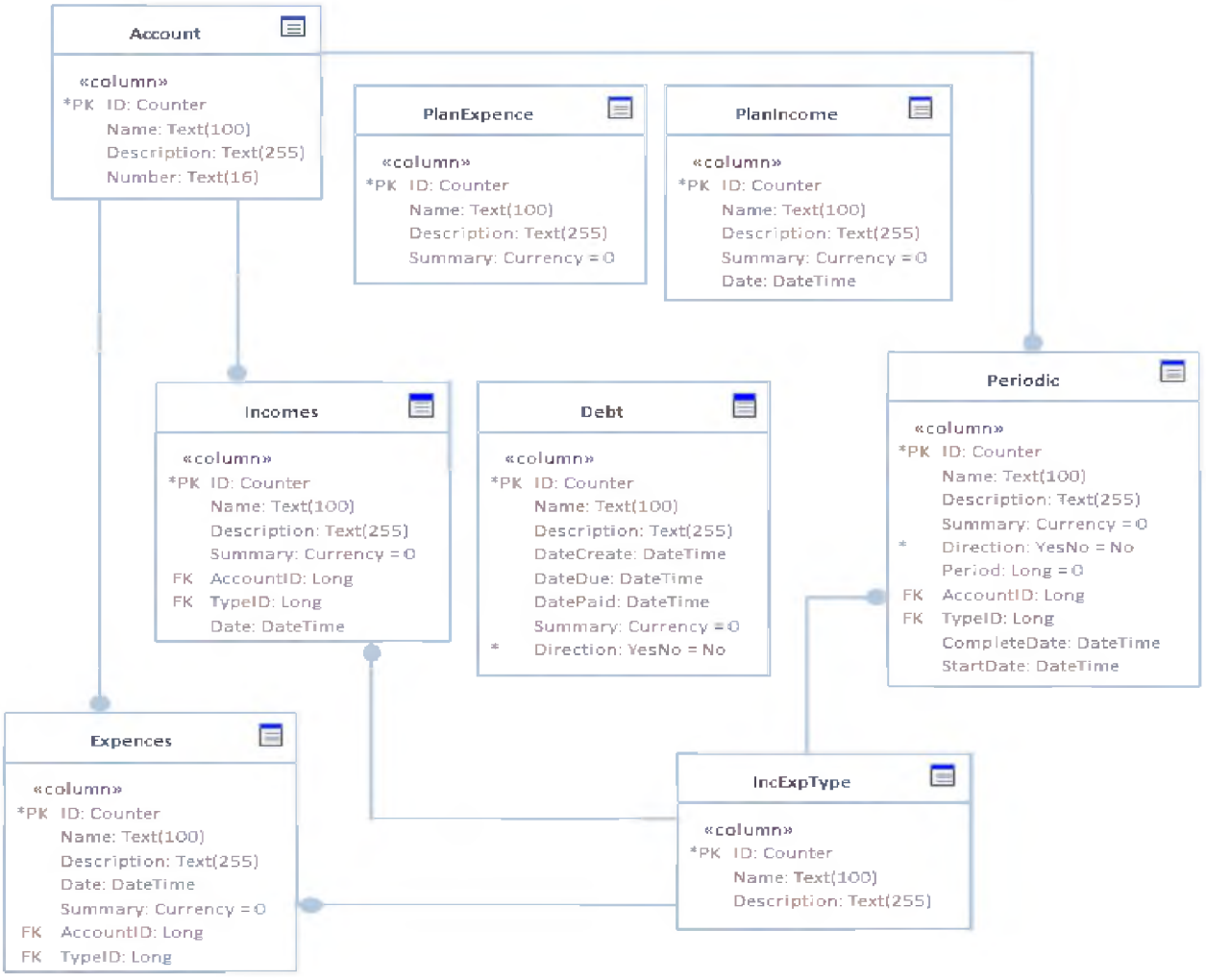

Рисунок 2.3 – Физическая модель данных информационной системы «Семейный бюджет»

Приведенный в таблицах 2.1-2.8 реквизитный состав таблиц базы данных экспортируется в СУБД для взаимодействия с прикладным программным обеспечением [19] системы, которое будет также являться интерфейсом к базе данных системы. Выбранная для проектирования CASE-система позволяет автоматизировать эту операцию, предоставляя средства ForwardDatabaseEngineering. Для этого необходимо выполнить команду «Generate» в разделе «DataModeling», выбрать таблицы, которые следует экспортировать в СУБД, и сгенерировать DDL-код [20] (рисунок 2.4).

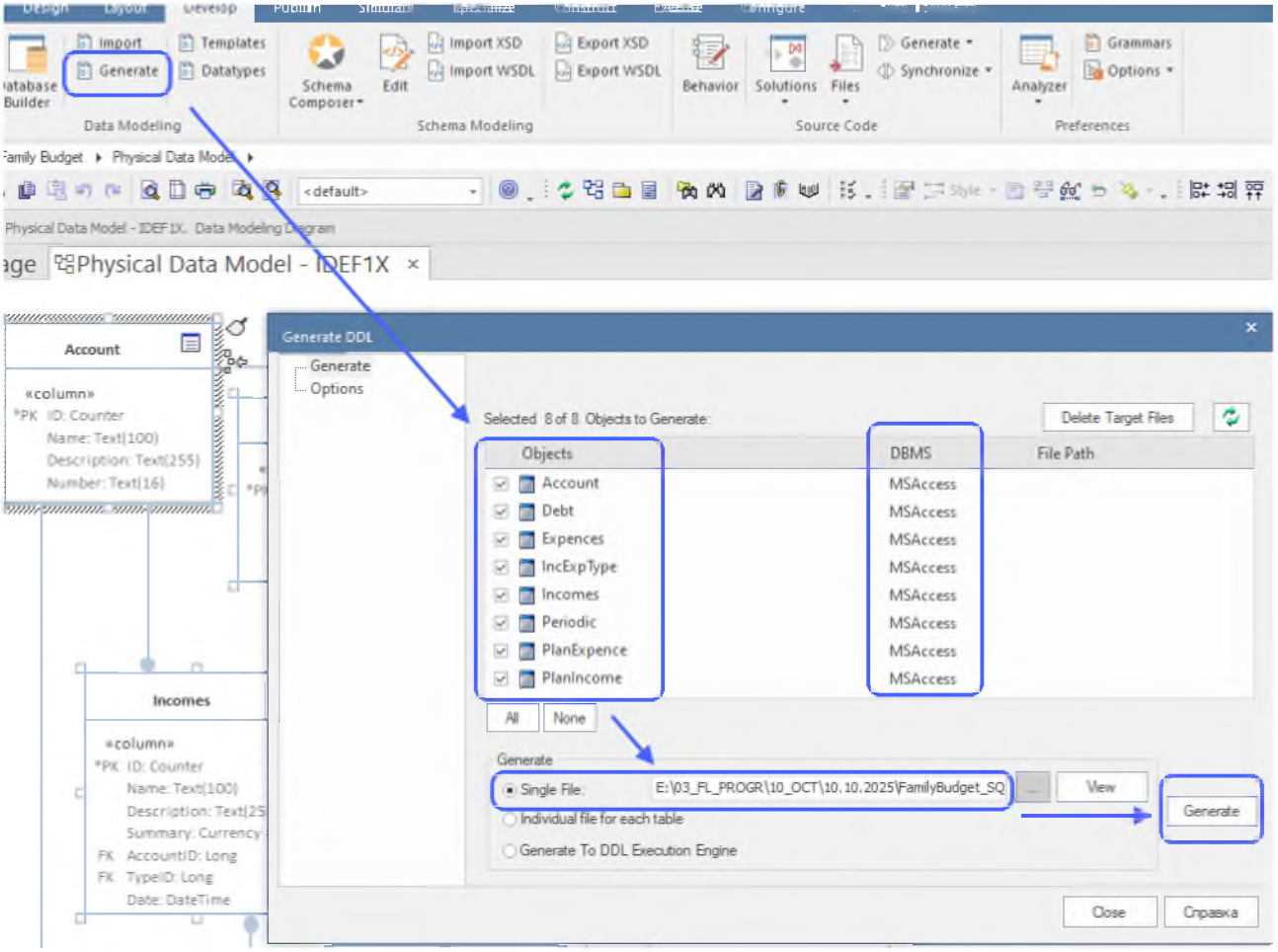

Рисунок 2.4 – Экспорт структуры БД в СУБД средствами CASE-системы EnterpriseArchitect

DDL-код (DataDescriptionLanguage) – это код на языке описания данных, который позволяет создать физические таблицы в соответствии с их спецификацией, установить между ними связи и определить ограничения ключей [17]. Так, в листинге 2.1 приведен DDL-код, автоматически

сгенерированный в CASE-системе, формирующий физическую структуру базы данных информационной системы «Семейный бюджет» в СУБД MSAccess (версия 2016 и выше).

Листинг 2.1 – DDL-код физической структуры БД информационной системы «Семейный бюджет»»

```
/* Drop Foreign Key Constraints */
ALTERTABLE[Expences]DROPCONSTRAINT[AccountExpences]
ALTERTABLE[Expences]DROPCONSTRAINT[IncExpTypeExpences]
ALTERTABLE[Incomes]DROPCONSTRAINT[AccountIncomes]
ALTERTABLE[Incomes]DROPCONSTRAINT[IncExpTypeIncomes]
ALTERTABLE[Periodic]DROPCONSTRAINT[AccountPeriodic]
ALTERTABLE[Periodic]DROPCONSTRAINT[IncExpTypePeriodic]

/* Drop Tables */
DROPTABLE[Account]
DROPTABLE[Debt]
DROPTABLE[Expences]
DROPTABLE[IncExpType]
DROPTABLE[Incomes]
DROPTABLE[Periodic]
DROPTABLE[PlanExpence]
DROPTABLE[PlanIncome]

/* Create Tables */
CREATETABLE[Account]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Number]Text(16)NULL
)
CREATETABLE[Debt]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[DateCreate]DateTimeNULL,
	[DateDue]DateTimeNULL,
	[DatePaid]DateTimeNULL,
	[Summary]CurrencyNULL,
	[Direction]YesNoNOTNULL
)
CREATETABLE[Expences]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Date]DateTimeNULL,
	[Summary]CurrencyNULL,
	[AccountID]LongNULL,
	[TypeID]LongNULL
)
CREATETABLE[IncExpType]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
```

```
	[Description]Text(255)NULL
)

CREATETABLE[Incomes]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Summary]CurrencyNULL,
	[AccountID]LongNULL,
	[TypeID]LongNULL,
	[Date]DateTimeNULL
)
CREATETABLE[Periodic]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Summary]CurrencyNULL,
	[Direction]YesNoNOTNULL,
	[Period]LongNULL,
	[AccountID]LongNULL,
	[TypeID]LongNULL,
	[CompleteDate]DateTimeNULL,
	[StartDate]DateTimeNULL
)
CREATETABLE[PlanExpence]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Summary]CurrencyNULL
)
CREATETABLE[PlanIncome]
(
	[ID]AUTOINCREMENTNOTNULL,
	[Name]Text(100)NULL,
	[Description]Text(255)NULL,
	[Summary]CurrencyNULL,
	[Date]DateTimeNULL
)

/* Create Primary Keys, Indexes, Uniques, Checks */

ALTERTABLE[Account]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[Debt]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[Expences]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[IncExpType]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[Incomes]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[Periodic]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[PlanExpence]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
ALTERTABLE[PlanIncome]ADDCONSTRAINT[PrimaryKey]
	PRIMARYKEY([ID])
```

```
/* Create Foreign Key Constraints */

ALTERTABLE[Expences]ADDCONSTRAINT[AccountExpences]
      FOREIGNKEY([AccountID])REFERENCES[Account]([ID])
ALTERTABLE[Expences]ADDCONSTRAINT[IncExpTypeExpences]
      FOREIGNKEY([TypeID])REFERENCES[IncExpType]([ID])

ALTERTABLE[Incomes]ADDCONSTRAINT[AccountIncomes]
      FOREIGNKEY([AccountID])REFERENCES[Account]([ID])
ALTERTABLE[Incomes]ADDCONSTRAINT[IncExpTypeIncomes]
      FOREIGNKEY([TypeID])REFERENCES[IncExpType]([ID])
ALTERTABLE[Periodic]ADDCONSTRAINT[AccountPeriodic]
      FOREIGNKEY([AccountID])REFERENCES[Account]([ID])
ALTERTABLE[Periodic]ADDCONSTRAINT[IncExpTypePeriodic]
            FOREIGNKEY([TypeID])REFERENCES[IncExpType]([ID])
```

Сгенерированный код выполняется экспортом, либо как серия запросов в СУБД. На рисунке 2.5 приведена полученная в СУБД схема базы данных (построена средствами СУБД).

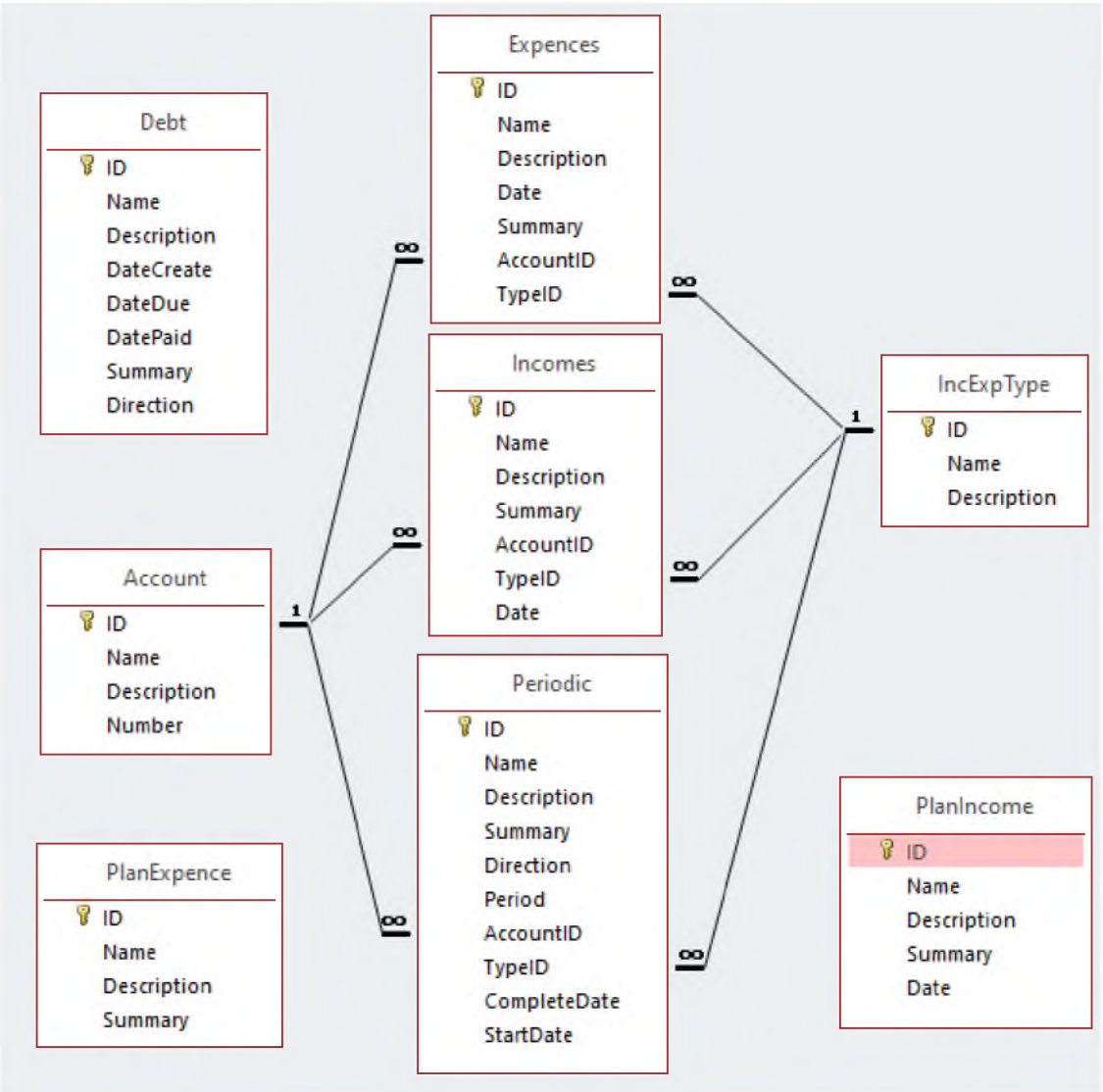

Рисунок 2.5 – Схема данных в СУБД MSAccess

Средства выбранной среды разработки – IDEVisualStudio – предлагают удобные способы интеграции источников данных на основе СУБД в проект программного обеспечения. Такие средства позволяют импортировать в проект всю структуру базы данных, спроецировать данные в специальные объектов – DataTable и оснастить их адаптерами – системными объектами, позволяющими выполнять запросы любых типов к таблицам базы данных [14]. Так, на рисунке 2.6 приведена модель данных в проекте программного обеспечения информационной системы «Семейный бюджет» в визуальном конструкторе MSVisualStudio.

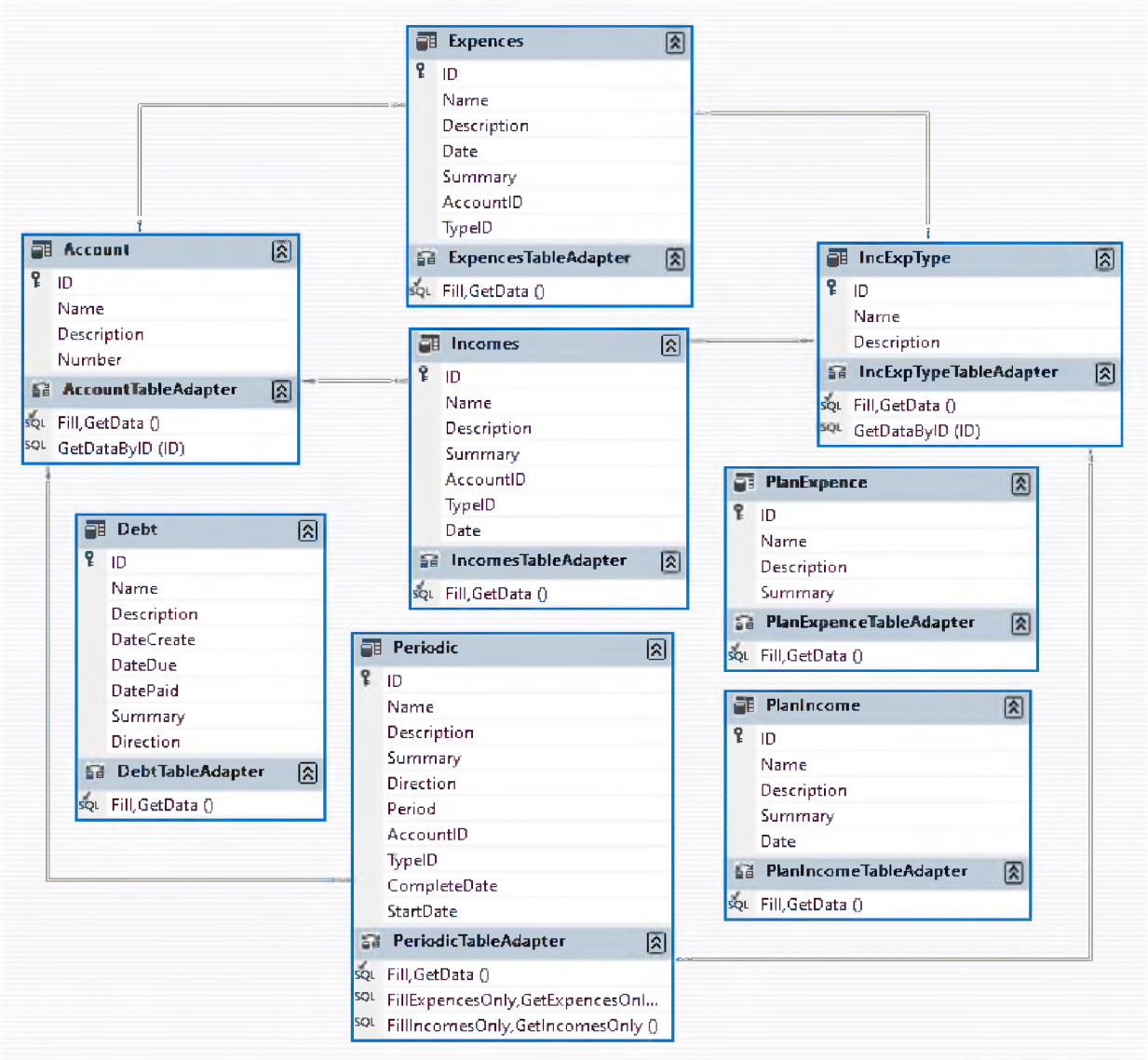

Рисунок 2.6 – Модель данных в проекте программного обеспечения информационной системы «Семейный бюджет» в визуальном конструкторе MSVisualStudio

Таким образом, очевидны преимущества консолидированного использования выбранных инструментов (CASE-системы, IDE) при создании бизнес-приложений, которые заключаются в:

– формализации и документировании на всех этапах проектирования баз данных;

– ускорении процессов инжиниринга за счет автоматизации операций экспорта / импорта;

– ускорении процессов разработки за счет средств интеграции модели данных непосредственно в проект разработки программного обеспечения информационной системы.

2.2 Технологическое обеспечение

2.2.1 Методы и средства разработки системы

Для разработки информационной системы «Семейный бюджет» использованы следующие средства и инструменты:

– CASE-система EnterpriseArchitect 15.0;

– среда разработки IDE MS VisualStudio 2019;

– СУБД MS Access 2016;

– язык программирования высокого уровня C# .NET, технология разработки WindowsForms, технологияOLEDB / .NET.

Для того, чтобы в процессе разработки системы добиться высокой эффективности, от инжинирингового инструмента требуется поддержка прямого инжиниринга – трансляции графических моделей на языки описания непосредственно программного кода или его описания [12]. EnterpriseArchitect является CASE-системой, позволяющей разрабатывать множество моделей информационных систем, включая модели на языке UML. Кроме этого, в этой CASE-системе возможно проектирование баз данных, начиная от концептуальной модели и заканчивая ее физической реализации средствами ForwardDatabaseEngineering.

MSVisualStudio является наиболее продвинутым инструментом разработчика, предоставляю самые важные функции и средства для ускоренной разработки бизнес-приложений [35]:

– редактор визуальных интерфейсов;

– средства для инжиниринга баз данных, управления подключениями к источникам данных;

– интеграция с популярными СУБД – MS Access / MySQL / SQLite / SQL Server;

– поддержка совместной командной работы;

– средства кодирования с функциями автозаполнения, подсветки синтаксиса, подсказок в режиме InteliSense;

– удобные и функциональные средства рефакторинга;

– наличие большого количества дополнительных пакетов и расширений (NuGet);

– поддержка интеграции с GitHub;

– реверс-инжиниринг и поддержка UML.

– средства тестирования программ.

Кроме того, являясь продуктом Microsoft, данная IDE отлично интегрируется с сервисами Microsoft (Reporting, DBMSSQLServer / Access, MSOfficeObjectModel и т. д.). Таким образом, VisualStudio отлично подходит для быстрой разработки приложений под Windows.

2.2.2 Технологические операции

Основными технологическими операциями в информационной системе «Семейный бюджет» являются:

– хранение / запись данных;

– обработка/ извлечение/ модификация данных;

– визуализация данных.

Технология хранения и взаимодействия с данными в проекте информационной системы выполнена посредством классов подключаемого уровня объектной модели ADO.NET.

Для этого в проекте создается новое подключение к источнику данных на основе технологии OLEDB – рисунок 2.7.

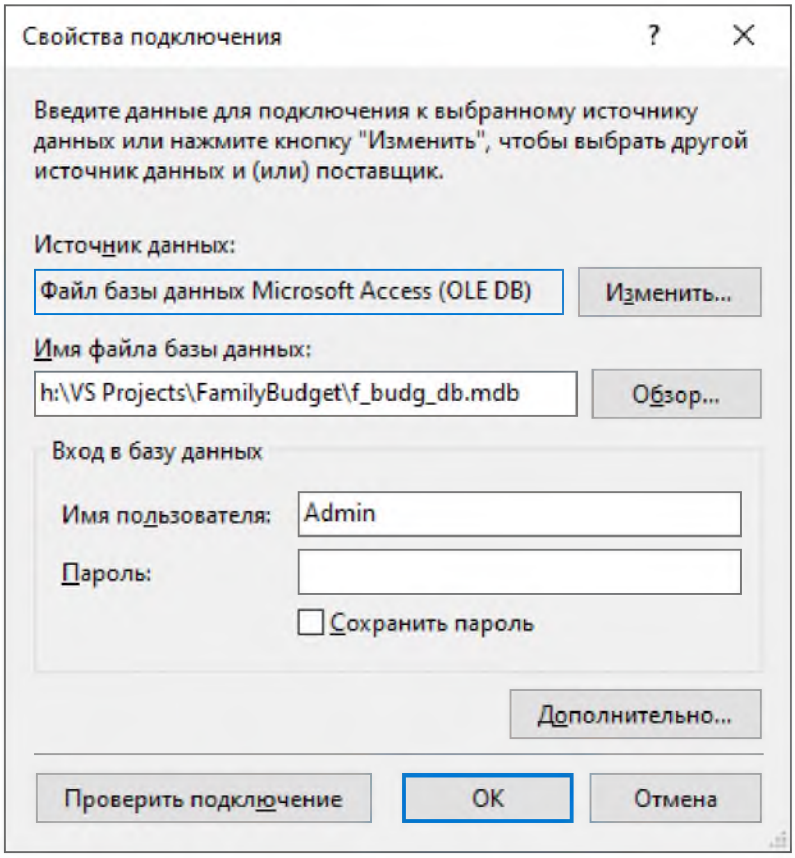

Рисунок 2.7 – Подключение БД к проекту VisualStudio

При создании датасета на основе подключенной базы данных выбираются все созданные в БД элементы (таблицы, представления и другие необходимые объекты). Система автоматически создает схему описания данных в формате XSD. В данной схеме создаются интегрированные объекты для программирования в приложении:

– DataSet на основании образа БД;

– элементы таблиц – DataTable, соответствующие таблицам, созданным в базе данных;

– специальные объекты-адаптеры, которые предназначены для

управления данными в таблицах – через них выполняются все запросы типа SELECTи запросы на манипулирование данными (CRUD).

Для визуализации данных на формах приложения создается главная форма, на которую методом Drag&Drop перетаскивается интересующая таблица из обозревателя источников данных. Система разработки сама создаст необходимые компоненты привязки, в которые добавляются кнопки обновления и изменения данных (рисунок 2.8).

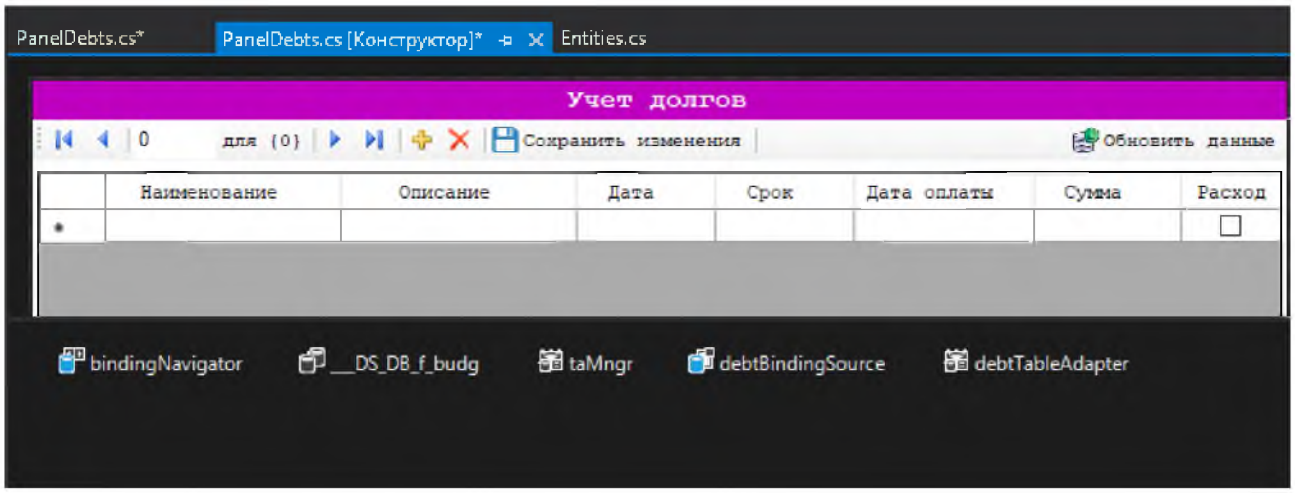

Рисунок 2.8 – Привязка элементов таблицы и управления данными к главной форме (на примере таблицы долгов)

Используя привязку объектов DataTable к графическим интерфейсам WindowsForms, можно составлять компоненты, позволяющие наполнять и вводить данные в систему с сохранением их в физической базе данных. Автоматически добавленные объекты можно стилизовать, переименовывать и менять свойства отображения и поведения.

2.3 Техническое обеспечение

Разрабатываемое приложение ИС «Семейный бюджет» характеризуется как однопользовательское с локально-расположенной базой данных. В связи с этим техническое обеспечение информационной системы составляет один персональный компьютер / ноутбук, соответствующий конфигурации, на

которой возможен запуск и нормальная работа операционной системы Windows 10. Требования к техническому обеспечению обеспечиваются следующими характеристиками комплекса:

– клавиатура;

– манипулятор типа «мышь»;

– монитор с частотой обновления кадров 60 Гц или выше;

– процессор от 2000 МГц и выше;

– объем ОЗУ не менее 256 МБ, не считая требований, предъявляемых операционной системой;

– видеоадаптерDirectX 9+ совместимый;

– свободное место на жестком диске – не менее 1 ГБ, не считая требований, предъявляемых операционной системой.

2.4 Программное обеспечение задачи

2.4.1 Архитектура программного обеспечения

Система в соответствии с [25] должна обладать такими свойствами, как целостность, делимость, коммуникативность, динамичность, иметь потенциал к развитию. Понятие программной системы определено в ГОСТ Р 51902-2002: это «система, состоящая из программного обеспечения и, возможно, компьютерного оборудования для его выполнения» [11]. Хорошая структуризация программной системы позволит правильно и эффективно распределить задачи построения и создания системы между командой разработчиков, повысив при этом уровень управляемости процессом разработки системы. При разработке архитектуры системы прежде всего должна быть обеспечена реализация функциональных требований, но также нельзя забывать и про нефункциональные, поскольку последние влияют на качественные характеристики продукта. Таким образом, основное требование к системе, при которой она не теряет смысла, это эффективность решения поставленных задач. Эффективность проектного и архитектурного решения

определяется прежде всего модульностью системы и ее логической структурой. Логическая структура системы строится из композиции логических структур ее подсистем. Иллюстрация логической структуры выполняется в нотации UML в виде диаграммы классов. Диаграмма классов информационной системы «Семейный бюджет» приведена на рисунке 2.9.

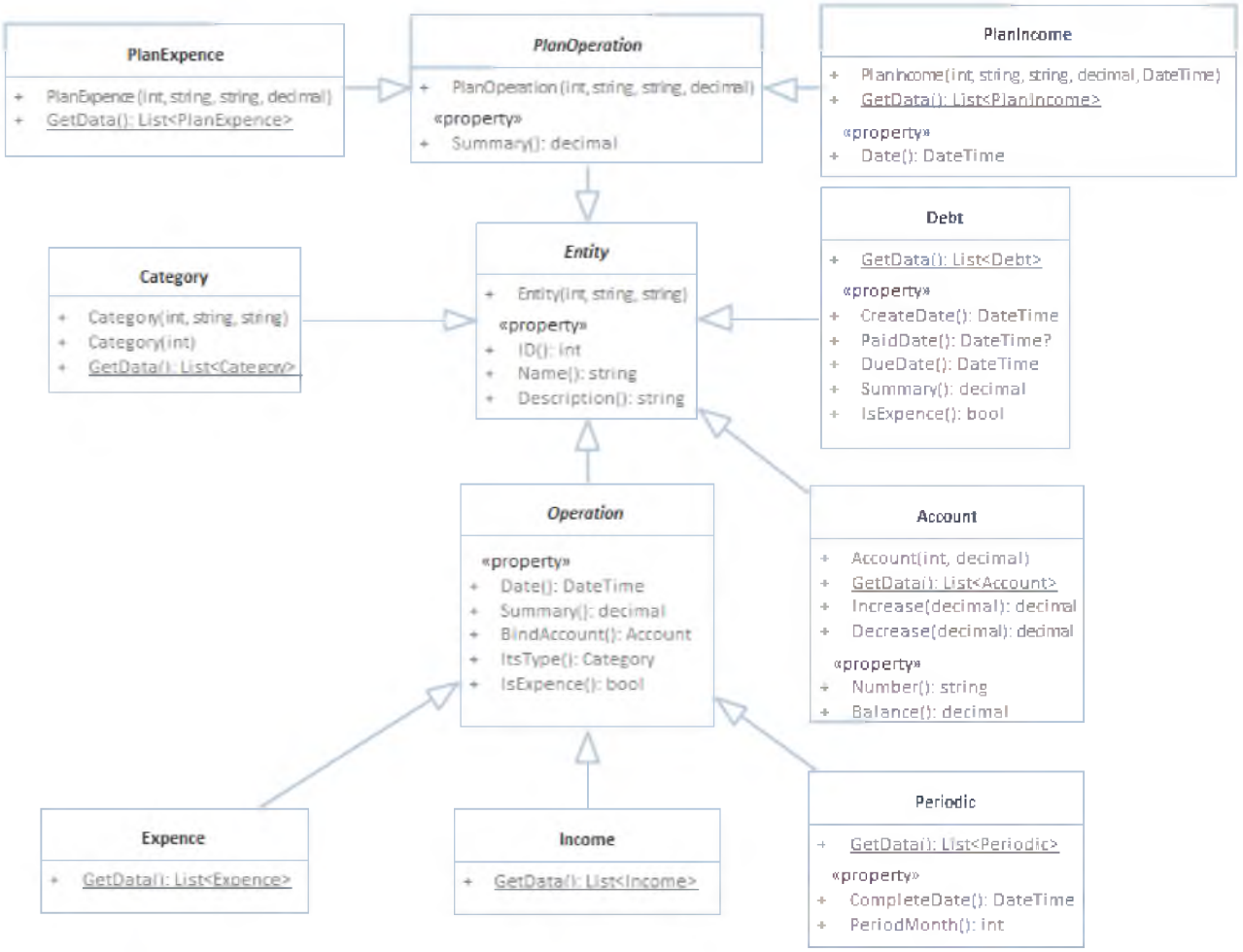

Рисунок 2.9 – Диаграмма классов ИС «Семейный бюджет»

Приведенные на рисунке 2.9 классы являются модельными прототипами соответствующих сущностей в базе данных, приведенных в разделе 2.1.

Программная система должна проектироваться так, чтобы в случае необходимости в нее можно было легко вносить изменения, легко «читать» ее структуру, сопровождать ее и моделировать новые функциональные возможности. Такое возможно только при применении модульного принципа построения архитектуры системы, при котором создаются пространства имен,

функциональные пакеты и т. д.[4]. В функциональные пакеты компонуются семантически схожие модули. При этом интенсивно используются шаблоны (паттерны) проектирования, прошедшие проверку временем и опытом применения в успешных проектах. На данный момент известно несколько десятков таких шаблонов. Одним из основополагающих принципов построения эффективной архитектуры объектно-ориентированных систем является соблюдение принципов SOLID [34]. На рисунке 2.10 приведена диаграмма пакетов информационной системы «Семейный бюджет».

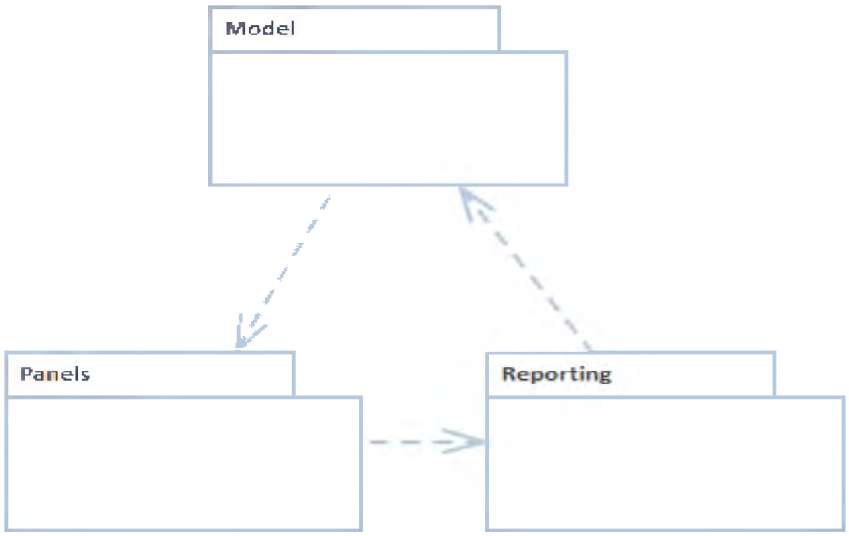

Рисунок 2.10 – Диаграмма пакетов ИС «Семейный бюджет»

Назначение пакетов, приведенных на рисунке 2.10, описано в разделе 2.4.2 – описание программных модулей.

Таким образом, построенная архитектура информационной системы «Семейный бюджет» имеет потенциал для доработки и возможного развития приложения.

2.4.2 Описание программных модулей

Приложение информационной системы «Семейный бюджет» включает в себя набор программных модулей, реализующих основной функционал системы. Модули распределены по подсистемам (пакетам):

– Model–консолидация объектов предметной области;

– Panels–пакет, содержащий основные формы подсистем;

– Reporting–пакет модулей отчетов.

Так, система состоит из следующих программных модулей:

– MainForm– основная форма приложения, из которой выполняется вызов всех панелей подсистем;

– PanelAccounts – форма, предоставляющая элементы организации и ведения справочника счетов;

– PanelDebts – форма, предоставляющая элементы организации и ведения учета долгов;

– PanelExpences – форма, предоставляющая элементы организации и ведения учета и регистрации расходов;

– PanelExpencesPlan – форма, предоставляющая элементы организации и ведения учетапланируемых расходов;

– PanelIncExpTypes – форма, предоставляющая элементы организации и ведения справочника категорий;

– PanelIncomes – форма, предоставляющая элементы организации и ведения учетаи регистрации доходов;

– PanelIncomesPlan – форма, предоставляющая элементы организации и ведения учета планируемых приходов;

– PanelPeriodicTransactions – форма, предоставляющая элементы организации и ведения учета периодических событий, определяющих приходы и расходы;

– ReportFormAccounts – форма вывода отчета типа «Расчетный листок» с данными по счетам;

– ReportFormTrend – форма графического отчета (тренда) изменения состояния счетов;

– __DS_DB_f_budg – объект доступа к базе данных (DataSet) через образ данных, включающий набор соответствующих объектов таблиц

(DataTable) и адаптеров (DataTableAdapter), реализующих методы доступа и обработки данных в СУБД;

– f_budg_db.mdb – файлбазы данных, который привязывается к проекту в качестве источника данных.

На рисунке 2.11 приведена диаграмма компонентов, показывающая структуру программных модулей информационной системы «Семейный бюджет».

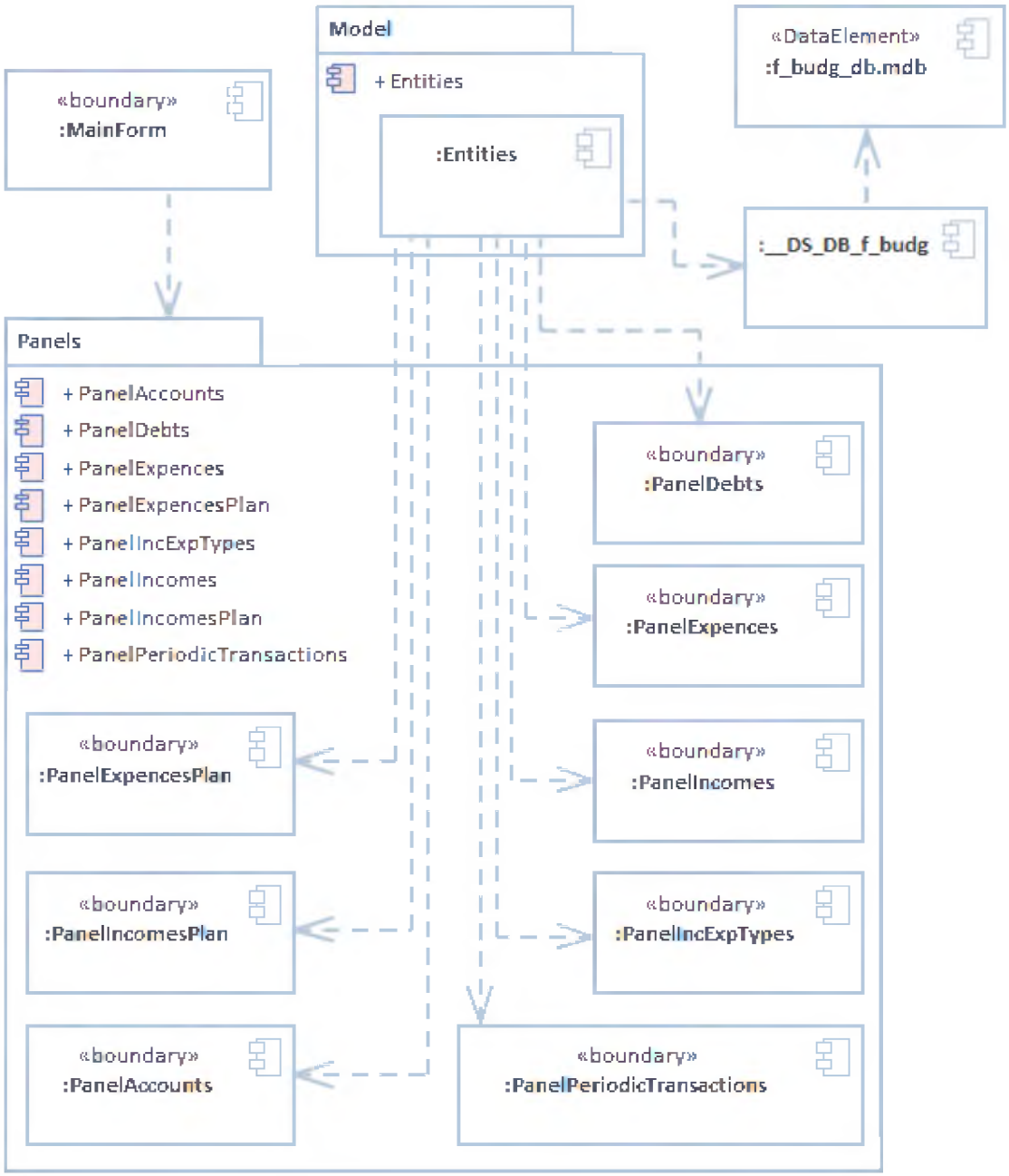

Рисунок 2.11 – Диаграмма структурная модулей информационной системы «Семейный бюджет»

Технология разработки приложения – Microsoft.Windows.Forms, т. е. для построения пользовательского интерфейса применяются стандартные компоненты, определенные пространством имен Microsoft.Windows.Forms. Само приложение состоит из основного системного окна (формы), в котором размещены необходимыекомпоненты отображения. При сборке форм использована технология контейнеров. Т. е. все элементы пользовательского интерфейса компонуются в контейнерные элементы следующих типов [7]:

- Panel;
- SplitContainer;
- FlowLayoutPanel;
- TableLayoutPanel.

Контейнер Panel представляет собой пространство формы или ее рабочую поверхность, на которой можно размещать элементы управления. Использование такого контейнера необходимо в случае, когда в одной форме или контейнере уровня выше необходимо использовать переключение видов. В этом случае в родительский контейнер можно реализовать на базе Panel, а содержимое контейнера переключать по заданной логике. Содержимое может быть выполнено отдельными компонентами на базе Microsoft.Windows.Control [15].

Контейнер SplitContainer используется для разделения области формы (контейнера уровня выше) на две составляющие горизонтально или вертикально. Основное свойство SplitContainer – наличие «плавающей» границы между разделяемыми областями. Таким образом, пользователь в процессе работы программы может динамически менять соотношение размеров разделяемых областей. Вкладывая элементы SplitContainer друг в друга, можно организовать что-то типа сетки для организации компонентов форм.

Описанные принципы контейнерной сборки компонентов пользовательского интерфейса позволяют:

– организовать функции подсистем в одном контейнере по типу переключения страниц на рабочем столе пользователя;

– выполнить компоновку различных элементов на страницах пользователя;

– организовать поддержку динамического размещения компонентов на формах;

– решить задачу автоматического упорядоченного размещения множества компонентов на формах.

Для вывода отчетов в приложении использованы:

– формат HTML и компонент типа WebBrowser

– элемент вывода графики – VisualCharting.Chart;

Форматированный текст отчета составляется в соответствующем модуле Reporting посредством консолидированных запросов к базе данных, группировке данных и использования построителя строк StringBuilder [18].

2.5 Руководство пользователя

В рамках процесса разработки приложения информационной системы «Семейный бюджет» были разработаны все формы и отчеты, описанные выше. Ниже приведены примеры демонстрации работы созданного приложения. На рисунке 2.12 приведен пример работы формы учета счетов пользователя.

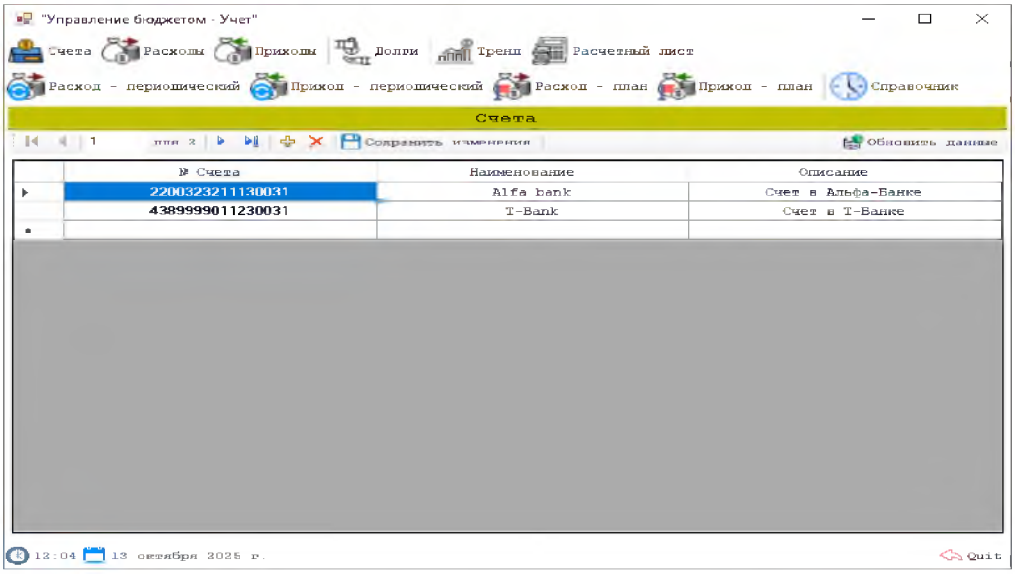

Рисунок 2.12 – Демонстрация работы приложения «Семейный бюджет»: форма учета счетов

На рисунке 2.13 приведен пример работы формы справочника категорий операций расхода / дохода.

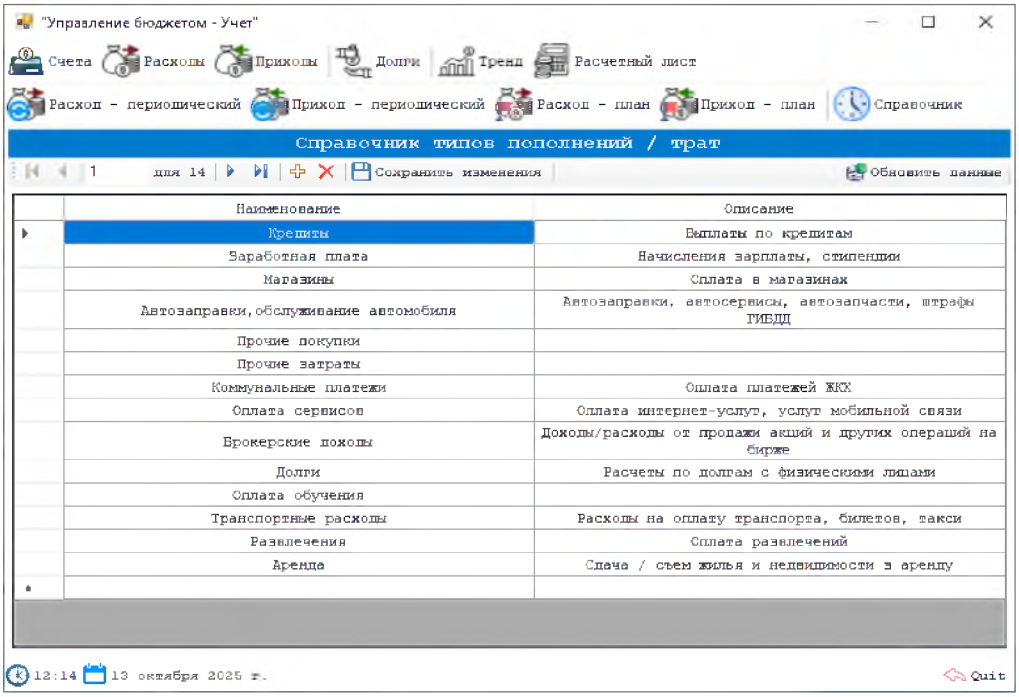

Рисунок 2.13 – Демонстрация работы приложения «Семейный бюджет»: форма справочника категорий операций расхода / дохода

На рисунке 2.14 приведен пример работы формы регистрации расходов по счетам.

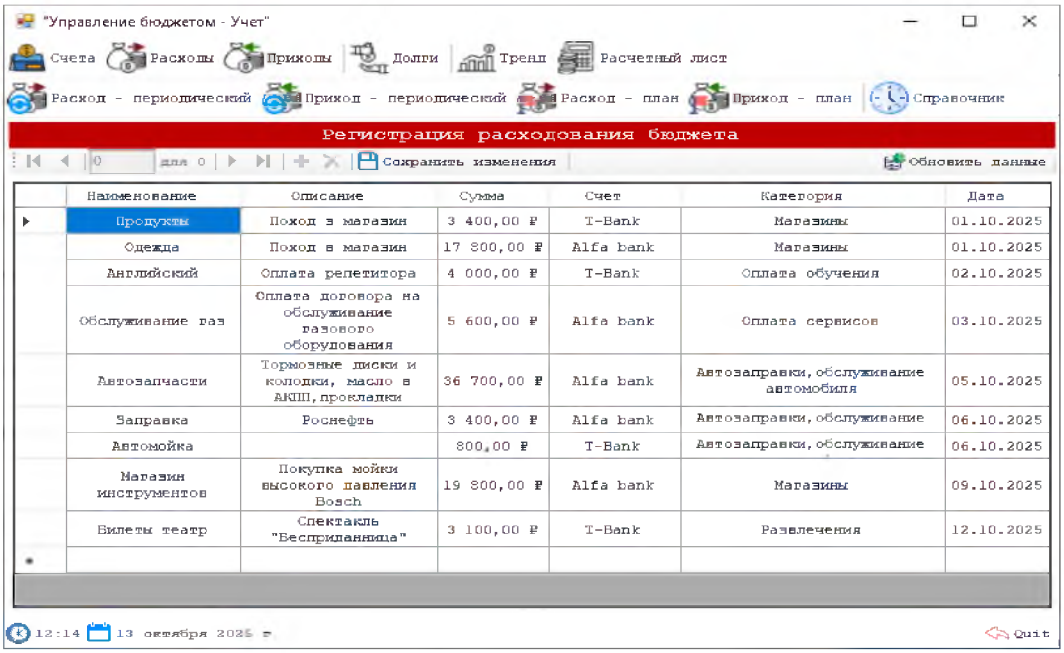

Рисунок 2.14 – Демонстрация работы приложения «Семейный бюджет»: форма регистрации расходов

На рисунке 2.15 приведен пример работы формы регистрации приходов и пополнений по счетам.

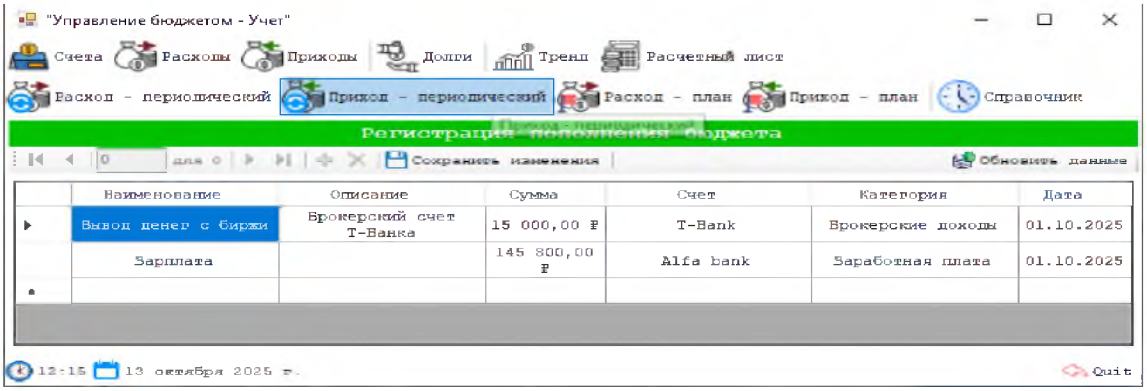

Рисунок 2.15 – Демонстрация работы приложения «Семейный бюджет»: форма регистрации доходов

На рисунке 2.16 приведен пример работы формы учета планируемых расходов.

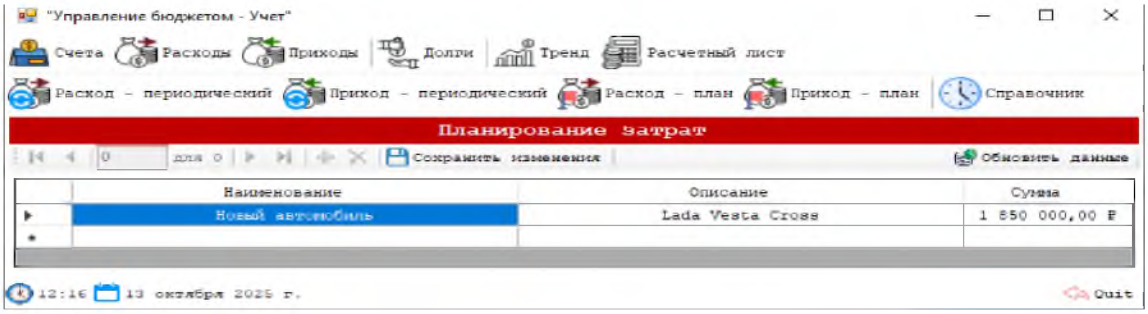

Рисунок 2.16 – Демонстрация работы приложения «Семейный бюджет»: форма учета планируемых расходов

На рисунке 2.17 приведен пример работы формы учета планируемых доходов

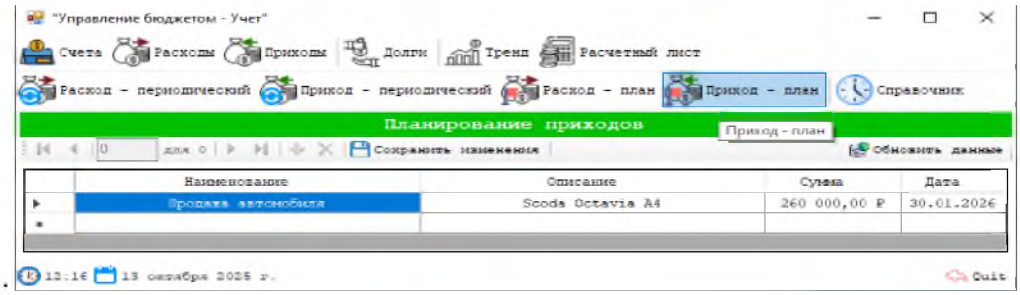

Рисунок 2.17 – Демонстрация работы приложения «Семейный бюджет»: форма учета планируемых доходов

На рисунке 2.18 приведен пример работы формы учета периодических расходов.

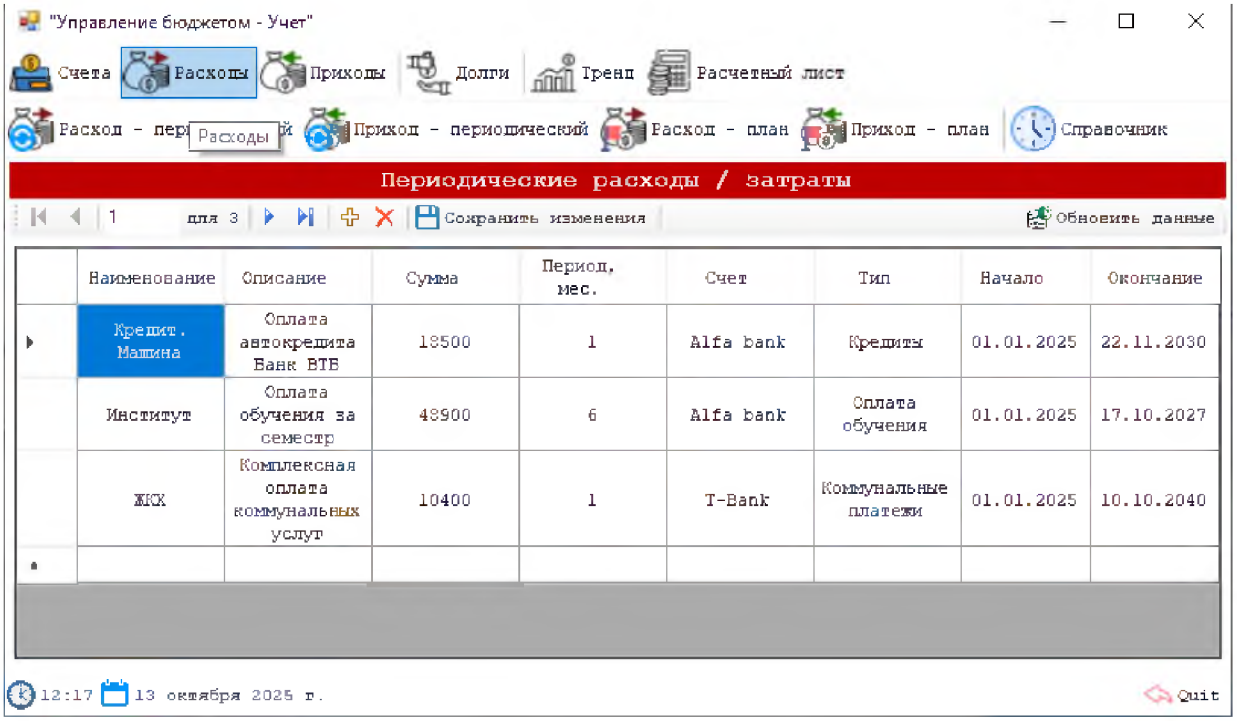

Рисунок 2.18 – Демонстрация работы приложения «Семейный бюджет»: форма учета периодических расходов

На рисунке 2.19 приведен пример работы формы учета периодических доходов.

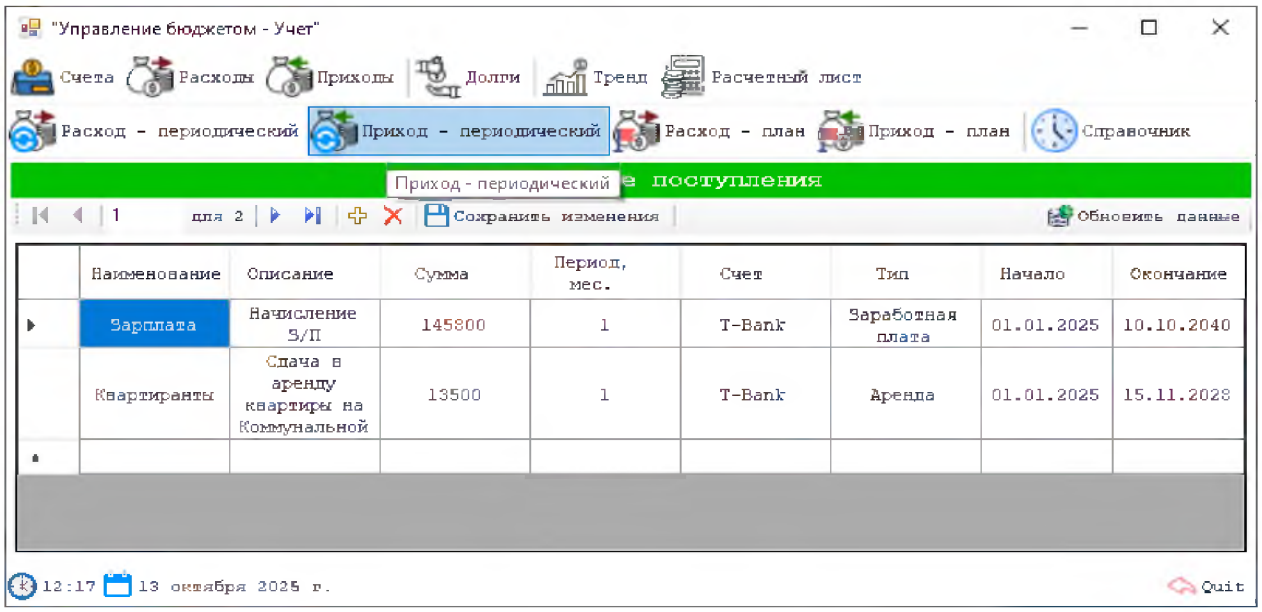

Рисунок 2.19 – Демонстрация работы приложения «Семейный бюджет»: форма учета периодических доходов

На рисунке 2.20 приведен пример работы формы учета долговых обязательств.

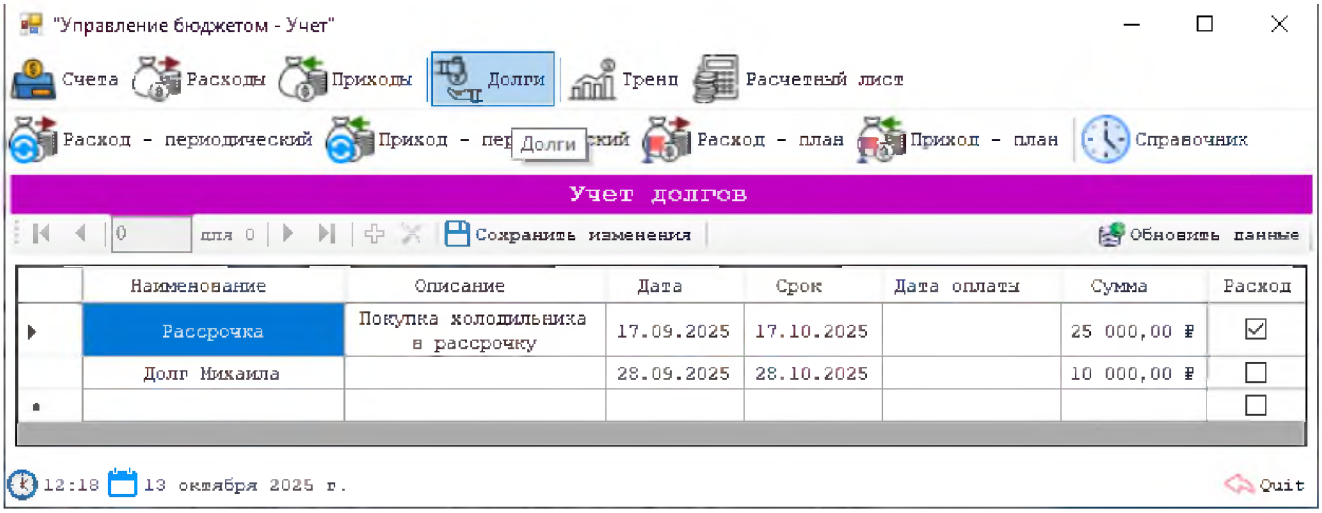

Рисунок 2.20 – Демонстрация работы приложения «Семейный бюджет»: форма учета долговых обязательств

На рисунке 2.21 приведен пример работы формы отчета по операциям по всем счетам пользователя.

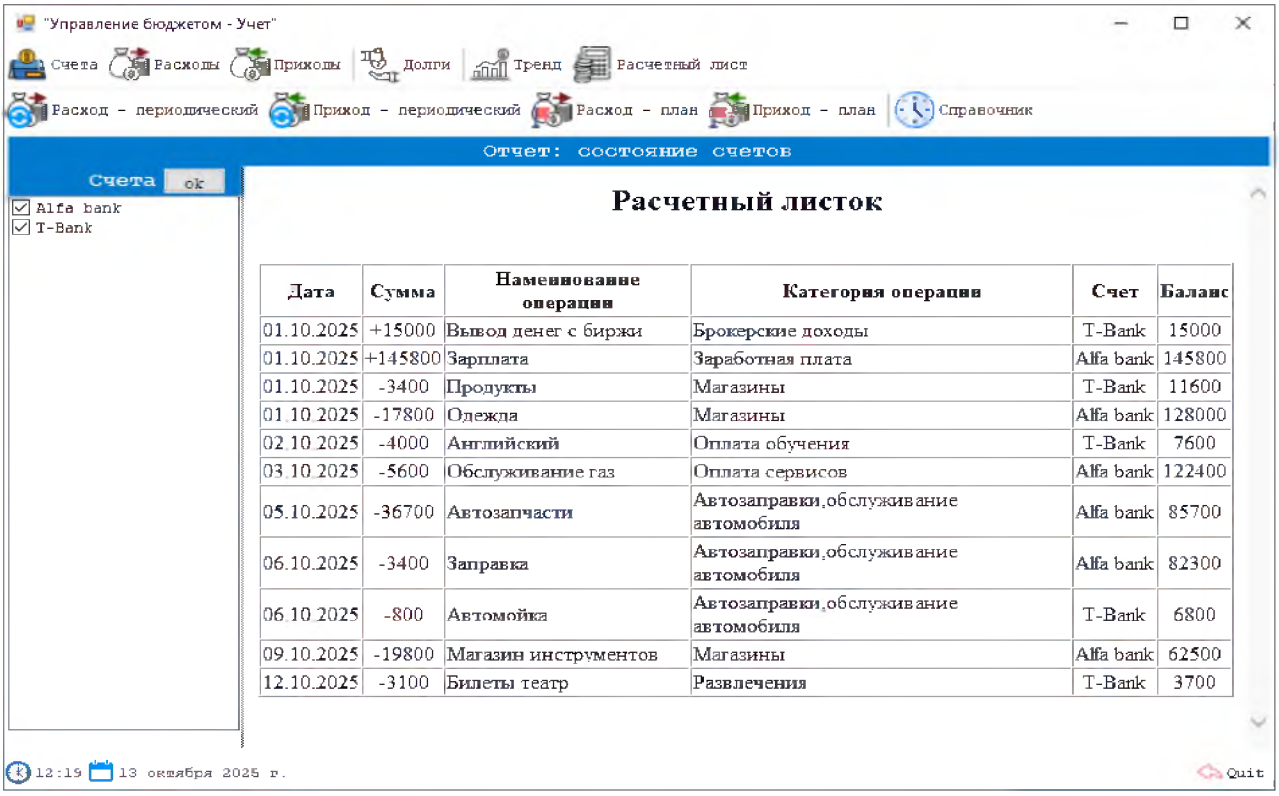

Рисунок 2.21 – Демонстрация работы приложения «Семейный бюджет»: форма отчета об операциях

На рисунке 2.22 приведен пример работы формы отчета-тренда по динамике изменения состояния счетов пользователя.

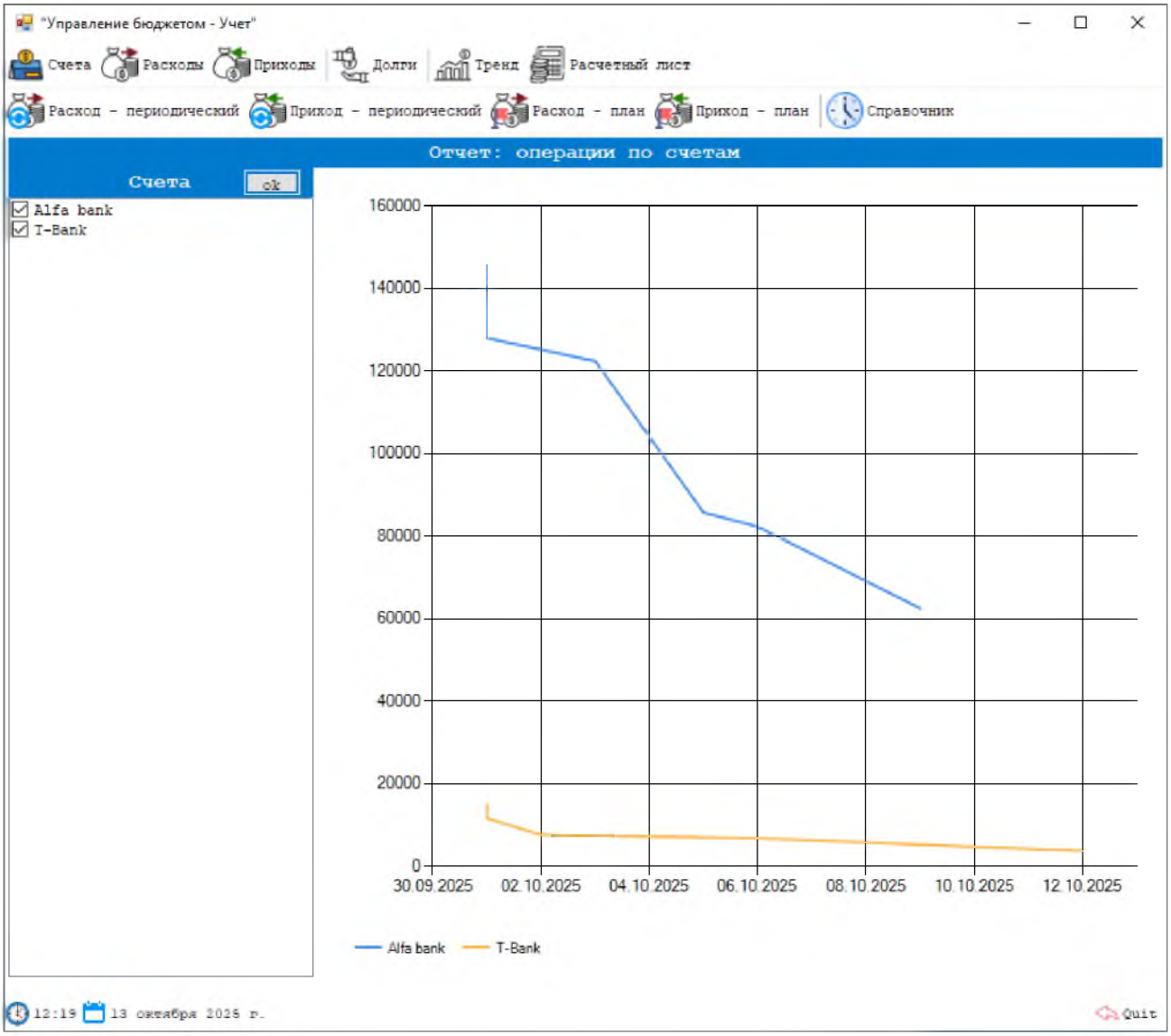

Рисунок 2.22 – Демонстрация работы приложения «Семейный бюджет»: форма отчета-тренда по динамике изменения состояния счетов пользователя

Все операции с данными на приведенных выше формах выполняются с помощью элементов, расположенных на панели меню в верхней части формы. Данные вносятся непосредственно в таблицы. Подтверждение ввода данных выполняется командой «Сохранить изменения».

Таким образом, представленные выше формы показывают, что разработанное приложение выполняет все заявленные функции в полном объеме.

3 Экономическая часть

3.1 Описание методики расчета экономической эффективности ИС

Расчет экономической эффективности проекта информационной системы «Семейный бюджет» выполняется последующей методике [2]:

– определить непосредственную стоимость разработки проекта, включая труд программиста и других участников разработки, амортизацию оборудования и необходимые капиталовложения в материалы, дополнительное оборудование, программное обеспечение и т. д.;

– провести расчёт показателей экономической эффективности при сравнении базового и проектного вариантов;

– сделать вывод по результатам разницы рассчитанных базового и проектного вариантов.

Определение себестоимости проекта выполняется в соответствии с формулой:

$$S_k = S_{hr} + S_{sw} + S_{el} + S_{am}, \quad (3.1)$$

где:

S_{hr} – оплата труда участников проектной команды, руб.,

S_{sw} – затраты на вспомогательные материалы, руб.,

S_{el} – затраты на электроэнергию и другие ресурсы, руб.

S_{am} – затраты на амортизацию оборудования, руб.

Срок окупаемости проекта рассчитывается, как:

$$T_{ок} = S_k / \Delta C_п, \quad (3.2)$$

где: $\Delta C_п$ – прогнозируемый или измеренный каким-либо способом показатель снижения затрат на выполнение бизнес-процессов после внедрения проекта, руб.

Расчетный коэффициент эффективности капитальных затрат рассчитывается по формуле:

$$E_р = 1 / T_{ок}, \quad (3.3)$$

Годовой экономический эффект равен годовому снижению стоимостных затрат:

$$Э = П - K_П \times E_н \text{, где:} \quad (3.4)$$

$Э$ – годовой экономический эффект;

$П$ – годовая экономия (годовой прирост прибыли после применения разработки, ΔС), руб.;

$K_П$ – единовременные капитальные затраты (на программирование, проектирование, отладку и внедрение), руб.;

$E_н$ – нормативный коэффициент эффективности капитальных вложений (минимальная норма эффективности капитальных вложений, ниже которой они нецелесообразны).

Значение $E_н$ принимается равным 0,15.

3.2 Расчет показателей экономической эффективности

При планировании ресурсного обеспечения проекта необходимо дифференцировать проектные роли, их вклад на разных стадиях проекта, т. к. каждая отдельная роль может иметь свою значимость, сложность и, соответственно, ставку для оплаты работы.

В таблице 3.1 приведены различные роли трудовых ресурсов, задействованные при разработке проекта информационной системы «Семейный бюджет».

Таблица 3.1 – Ресурсы проекта информационной системы «Семейный бюджет»

Название ресурса	Тип	Краткое название	Стандартная ставка	Вид начисления
Программист	Трудовой	П	500,00р./ч	Пропорциональное
Тестировщик	Трудовой	Т	370,00р./ч	Пропорциональное
Системный аналитик	Трудовой	С	290,00р./ч	Пропорциональное
Руководитель	Трудовой	Р	765,00р./ч	Пропорциональное

Данные таблицы вводятся в подсистему MSProject, в которой разрабатывается детальный график работы над проектом ИС. Конечный календарный план реализации проекта информационной системы «Семейный бюджет», разработанный в MSProject, приведен на рисунке 3.1.

Режим задачи	Название задачи	Длитель	Начало	Окончание	Трудозатрат	Затраты	Краткое ресурса
	РАЗРАБОТКА ПРОЕКТА ИС "Семейный бюджет"	**34 дней**	**Ср 01.10.25**	**Пн 17.11.25**	**175,36 ч**	**82 988,80р.**	
	Этап 1. Разработка общей концепции	**5 дней**	**Ср 01.10.25**	**Вт 07.10.25**	**22,4 ч**	**13 840,00р.**	
	Разработка общих решений	2 дней	Ср 01.10.25	Чт 02.10.25	15,2 ч	8 544,00р.	С;Р;П
	Разработка модели требований	2 дней	Пн 06.10.25	Вт 07.10.25	7,2 ч	5 296,00р.	Р;П
	Этап 2. Разработка технического проекта	**10 дней**	**Чт 09.10.25**	**Ср 22.10.25**	**53,36 ч**	**23 888,80р.**	
	Разработка моделей БД	2 дней	Чт 09.10.25	Пт 10.10.25	7,36 ч	5 308,80р.	Р;П;С
	Разработка прототипов пользовательского интерфейса	5 дней	Пн 13.10.25	Пт 17.10.25	37,6 ч	13 140,00р.	Р;П;С
	Создание объектной модели	2 дней	Ср 15.10.25	Чт 16.10.25	6,4 ч	4 304,00р.	Р;С;П
	Создание компонентной модели	1 день	Ср 22.10.25	Ср 22.10.25	2 ч	1 136,00р.	С;Р;П
	Этап 3. Разработка рабочего проекта	**8 дней**	**Пн 27.10.25**	**Ср 05.11.25**	**60,4 ч**	**25 196,00р.**	
	Разработка БД	2 дней	Пн 27.10.25	Вт 28.10.25	16 ч	11 648,00р.	П;Р;С
	Программирование обмена данными с БД	3 дней	Пт 31.10.25	Вт 04.11.25	2,4 ч	948,00р.	П;С
	Программирование функций ИС	5 дней	Чт 30.10.25	Ср 05.11.25	42 ч	12 600,00р.	С;П
	Этап 4. Внедрение	**4 дней**	**Ср 12.11.25**	**Пн 17.11.25**	**39,2 ч**	**20 064,00р.**	
	Тестирование ИС	1 день	Ср 12.11.25	Ср 12.11.25	1,6 ч	592,00р.	Т
	Доработка по итогам тестирования	2 дней	Чт 13.11.25	Пт 14.11.25	36 ч	18 880,00р.	С;Р;П
	Повторное тестирование	1 день	Пн 17.11.25	Пн 17.11.25	1,6 ч	592,00р.	Т

Рисунок 3.1 – Календарный план разработки информационной системы «Семейный бюджет»

По данным таблицы, приведенной на рисунке 3.1, в системе MSProjectв автоматизированном режиме был получен календарный план реализации проекта в виде диаграммы Ганта (рисунок 3.2).

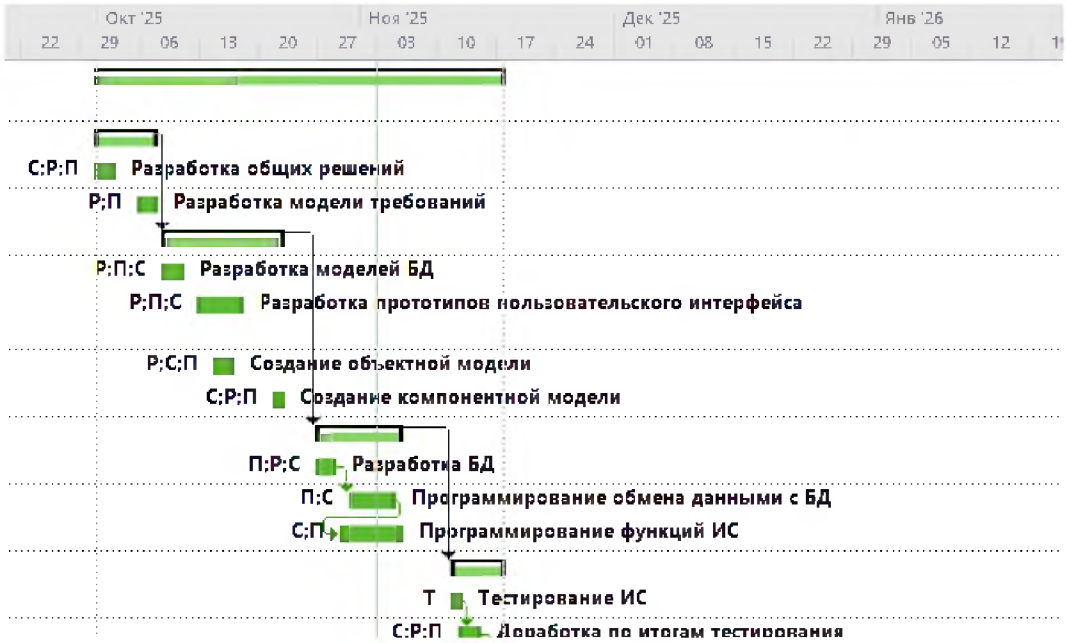

Рисунок 3.2 – Диаграмма Ганта реализации проекта информационной системы «Семейный бюджет»

Из рисунка 3.1 видно, что общие затраты на реализацию проекта информационной системы «Семейный бюджет» составляют:

– трудовые: 175,36 ч.;

– материальные:82988,80 руб.

Дополнительно к затратам на оплату труда разработчиков необходимо добавить 30,2% отчислений страховых взносов в ФОТ. Таким образом, затраты на оплату труда команды разработчиков составят:

$$S_{hr} = 82988,80\text{руб.} * 1,302 = 108051,42 \text{ руб.}$$

Амортизация основного оборудования производится следующим образом [1]. Для разработкиИС были использованы:

– персональный компьютер разработчика в конфигурации: Intel® Core™ i7-8700 CPU 3.20 GHz 16 GB RAM, 1 TB HDD, DisplayDell 27’’ FULL HD, Mouse + KeyboardLogitech;

– версия VisualStudio 2019 Community Edition.

По оценкам интернет-ресурсов, совокупная первоначальная стоимость аппаратного обеспечения составляет $П_с$ = 93000 руб.В соответствии с ОК 013–

2014 с 2017 года персональным компьютерам присвоена вторая амортизационная группа и, соответственно, срок полезного использования (СПИ) – от 25 до 36 месяцев [23]. Приняв для аппаратного комплекса СПИ = 36 месяцев и ликвидационную стоимость в $Л_с$ = 20000 руб., можно рассчитать его амортизационную стоимость в месяц:

$$S_m = (П_с - Л_с)/ 36 = 2027,78 \text{ руб.}$$

Трудозатраты на разработку проекта информационной системы «Семейный бюджет» составляют 175,36 ч. (см. рис. 3.1), что при рабочем дне в 8 часов составят 50,00 рабочих дней или два месяца.Таким образом, амортизационная стоимость эксплуатации аппаратного комплекса разработчика за три месяца составит:

$$S_{am} = S_m * 2 = 4055,56 \text{руб.}$$

Зная трудозатраты проекта, можно также рассчитать стоимость электроэнергии, которую потребит аппаратный комплекс разработчика за это время. Стоимость электроэнергии с 1 августа 2025 года по тарифу составляет 6,43 руб. за 1 кВт• ч. Мощность блока питания персонального компьютера составляет Рбп = 600 Вт. Таким образом, затраты на электроэнергию при разработке ИС составляют:

$$S_{el} = 175,36 \text{ ч.} * 0,6 \text{ КВт} * 6,43 \text{ руб.} / \text{КВт} \cdot \text{ч} = 676,54 \text{руб.}$$

Вспомогательные материалы, примененные при разработке системы, включают следующие позиции:

- носитель информации (флеш-накопитель Transcend 2 GB): 450 руб.;
- тонер для принтера HP LaserJet 1018 – 1 картридж: 1200 руб.;
- канцелярские принадлежности в ассортименте – 150 руб.

Таким образом, общая стоимость затрат на вспомогательные материалы, использованные при разработке системы, составляет:

$$S_{sw} = 450 + 1200 + 150 = 1800,00 \text{ руб.}$$

Подведя итог, можно выполнить расчёт себестоимости программной разработки проекта информационной системы «Семейный бюджет» в соответствии с (3.1):

$$S_k = S_{hr} + S_{sw} + S_{el} + S_{am} =$$

108051,42 + 1800,00 + 676,54+ 4055,56= 114583,52руб.

Согласно применяемой методологии, требуется выполнить анализ экономических выгод, полученных в результате применения средств информационной системы. В качестве экономической выгоды применительно к информационной системе «Семейный бюджет» можно отнести снижение трудоемкости выполняемых операций / расчетов /документов. В качестве таких операций можно определить:

- ведение учета операций;
- составление отчета по операциям;
- ведение планов покупок;
- составление отчета-тренда по финансовым поступлениям и тратам.

Результаты вычисления затрат на обработку указанных операций до автоматизации приведены в таблице 3.2.

Таблица 3.2 – Расчеты затрат на операции до автоматизации

№ п/п	Наименование операции	Единица измерения	Объём работы в год	Норма выработки (операций в час)	Трудоём-кость
1	2	3	4	5	6
1	Ведение учета операций	Операция	6000,00	60,00	100,00
2	Составление отчета по операциям	Отчет	30,00	0,50	60,00
3	Ведение планов покупок	Операция	300,00	10,00	30,00
4	Составление отчета-тренда по финансовым поступлениям и тратам	Отчет	30,00	0,33	90,91
					280,91

В таблице 3.3 приведены результаты вычисления затрат на обработку указанных операций после автоматизации. Данные получены на основании прогнозируемого эффекта от внедрения информационной системы для выполнения указанных задач.

Таблица 3.3 – Расчеты затрат на операции после автоматизации

№ п/п	Наименование операции	Единица измерения	Объём работы в год	Норма выработки (операций в час)	Трудоём-кость
1	2	3	4	5	6
1	Ведение учета операций	Операция	6000,00	120,00	50,00
2	Составление отчета по операциям	Отчет	30,00	30,00	1,00
3	Ведение планов покупок	Операция	300,00	60,00	20,00
4	Составление отчета-тренда по финансовым поступлениям и тратам	Отчет	30,00	30,00	1,00
					72,00

На основании рассчитанных показателей трудоёмкости операций в обоих вариантах производится расчёт изменения абсолютной и относительной динамики затрат, а также вычисляется коэффициент изменения расходов.Для расчёта стоимостных затрат рассчитывается среднечасовая норма оплаты труда сотрудника, выполняющего операцию. В данном случае можно принять:

$N_3 = 340{,}91$ руб./ч.

В таблице 3.3 представлены расчеты трудовых и экономических затрат при базовом и проектном варианте решения задачи.

Таблица 3.4 – Расчётпоказателей эффективности от внедрения проекта

Показатели	Затраты		Снижение затрат	Коэфф. изменения трудовых затрат	Индекс изменения трудовых затрат
	Базовый вариант	Проектный вариант			
Трудоёмкость	T_0 (ч)	T_1 (ч)	$\Delta T = T_0 - T_1$ (ч)	$K_T = \frac{\Delta T}{T_0}$	$I_T = \frac{T_0}{T_1}$
	280,91	72,00	208,91	0,74	1,34
Стоимость	C_0 (руб.)	C_1 (руб.)	$\Delta C = C_0 - C_1$ (руб.)	$K_C = \frac{\Delta C}{C_0}$	$I_C = \frac{C_0}{C_1}$
	95765,03	24545,52	71219,51	0,74	1,34

Анализ данных, представленных в таблице 3.3, свидетельствует о существенном уменьшении как трудовых, так и финансовых ресурсов. В абсолютных показателях снижение трудозатрат достигло 208,91чел.-ч., а экономия денежных средств составила 71219,51руб. в год.

При капитальных затратах S_k = 114583,52 руб., рассчитанных выше, срок окупаемости проекта информационной системы «Семейный бюджет» в соответствии с методикой (3.2) составит:

$$T_{ок} = 114583,52\text{руб.} / 71219,51\ \text{руб.} \approx 1,61\text{года} \approx 20\ \text{мес.}$$

Расчетный коэффициент эффективности капитальных затрат (3.3):

$$E_р = 1 / T_{ок} = 1 / 1,01 = 0,62$$

Годовой экономический эффект равен годовому снижению стоимостных затрат (3.4):

$$Э = 71219,51\ \text{руб.} – 114583,52\ \text{руб.} * 0,15 = 54031,92\text{руб.}$$

Таким образом, внедрение проекта информационной системы «Семейный бюджет» позволит достичь потенциальной экономии личного времени пользователя, которое можно выразить в денежном эквиваленте как 54031,92 руб. в год. Кроме того, пользователь получит удобный инструмент для ведения личных финансов и управления планированием бюджета.

Заключение

В рамках выполненной работы был выполнен обзор предметной области «управление личными финансами», определены основные требования для автоматизации процессов. На основании этого были формализованы функциональные требования к информационной системе в виде диаграммы вариантов использования (Use-Case). Таким образом, были определены основные функции.

На основании результатов обзора предметной области была спроектирована база данных информационной системы «Семейный бюджет», включая все основные этапы проектирования: концептуальная модель (нотация Чена), информационно-логическая модель (нотация Мартина), физическая модель (нотация IDEF1X) и реализация в СУБД (MSAccess).

В работе была разработана архитектура системы, разработанная с соблюдением принципов модульности и разделения ответственности в соответствии с шаблонами, описанными в SOLID. Разработанный технический проект информационной системы позволил выполнить рабочую версию приложения. Для разработки приложения были выбран следующий стек технологий и средств: MSVisualStudio 2019 / C# / SQL. Рабочий проект приложения был выполнен в соответствии с концептуальными решениями технического проекта.

Структурная схема модулей показала перечень и взаимосвязи программных модулей приложения. Реализация приложения позволила выполнить автоматизацию заявленных функций.

Разработанное приложение было протестировано. Тестовые запуски приложения показали корректность и полноту всех заявленных функций.

Экономическая оценка разработанного проекта показала, что сокращение трудовых затрат на выполнение операций после внедрения информационной системы в абсолютном выражении составило 208,91 чел.-ч, стоимостных затрат — 71219,51 руб. в год. Сокращение затрат в процентном выражении составляет

74%.При капитальных вложениях на разработку проекта, составляющих 114583,52 руб., годовой экономический эффект от проекта информационной системы составит свыше 54 тыс. руб.Окупаемость проекта внедрения системы, составляющая около 20 месяцев, указывает на возврат капитальных вложений за счет потенциальной экономии личного времени пользователя, которое можно выразить в денежном эквиваленте.

Таким образом, в целом цель выпускной квалификационной работы достигнута, а все ее задачи – решены.

Список литературы

1) Амортизация компьютерной техники: учёт, сроки, проводки — «Моё Дело» [Электронный ресурс] URL: https://www.moedelo.org/club/buhgalterskij-uchet/amortizaciya-kompyuternoj-tekhniki (дата обращения: 20.10.2025 г).

2) Афоничкин, А. И. Управленческие решения в экономических системах: учеб. для вузов / А.И. Афоничкин, Д.Г. Михаленко. – Санкт-Петербург: Питер, 2020. – 480 с.

3) Балдин, К.В. Информационные системы в экономике / К.В. Балдин, В.Б. Уткин. – М.: Издательский центр Академия, 2021. – 288 с.

4) Благодатских, В.А. Стандартизация разработки программных средств: учеб. пособие для вузов / В. А. Благодатских, В. А. Волнин, К. Ф. Поскакалов; под ред. О. С. Разумова. – М.: Финансы и статистика, 2022 г. – 288 с.

5) Братищенко, В.В. Проектирование информационных систем. Иркутск: Изд-во БГУЭП, 2022 г. – 84 с.

6) Будко, К. Л. Автоматизация проектирования реляционных баз данных / К. Л. Будко, З. М. Альбекова // Студенческая наука для развития информационного общества : Сборник материалов XIII Всероссийской научно-технической конференции с международным участием, Ставрополь, 12–13 декабря 2022 года. – Ставрополь: Северо-Кавказский федеральный университет, 2023. – С. 51-57.

7) ВершининМ., ИвановаЕ. C# Enterprise Edition. Технологии проектирования и разработки. – М.: BHV, 2023 г. – 1088 с.

8) Высокогляд, С. В. Семейный бюджет: основы финансового планирования / С. В. Высокогляд // Актуальные проблемы и перспективы развития потребительского рынка : Материалы XIII Международной научно-практической конференции студентов и учащихся, Пермь, 02–12 декабря 2024 года. – Пермь: Российский экономический университет им. Г.В. Плеханова, 2024. – С. 48-52.

9) Головичнер, М.Н. Проектирование информационных систем. Методические указания по подготовке к государственному экзамену / М.Н. Головичнер. – Томск, 2021 г. – 110 с.

10) ГОСТ Р 50.1.028-2001. Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования.

11) ГОСТ Р 51904-2002. Программное обеспечение встроенных систем. Общие требования к разработке и документированию.

12) Гохберг, Г.С. Информационные технологии[Текст]: учебное пособие / Г.С. Гохберг. – М.: Academia, 2021. – 474 с.

13) Дейт, К.Дж. Введение в системы баз данных. – Пер. с англ. – 6-е изд. – К. Диалектика, 2023. – 346 с.

14) Демин, А. Ю. Информатика. Программирование на C# в VisualStudio: учеб. / А. Ю. Демин, В. А. Дорофеев. — 2-е изд. — Москва: Издательство Юрайт, 2025. — 138 с. Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/565099 (дата обращения: 13.10.2025).

15) Джозеф Албахари – C#. Справочник. Полное описание языка. Пер. с англ. - М: ООО «И.Д. Вильямс», 6-е изд., 2024, 1040 с

16) Дорохова, А. М. Создание логической и физической модели базы данных / А. М. Дорохова, В. А. Шацкий, Н. В. Картечина // Наука и Образование. – 2020. – Т. 3, № 4. – С. 36.

17) Илюшечкин, В. М. Основы использования и проектирования баз данных: учеб. В. М. Илюшечкин. — Москва: Издательство Юрайт, 2025. — 213 с. Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/562514 (дата обращения: 13.10.2025).

18) Казанский, А. А. Программирование на C# : учеб. пособие. Казанский. — 3-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2025. — 181 с. Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/569863 (дата обращения: 13.10.2025)

19) Кугаевских, А. В. Проектирование информационных систем. Системная и бизнес-аналитика / А.В. Кугаевских, А.Н. Шишков. – учеб. пособие. – Новосибирск: Новосибирский государственный технический университет (НГТУ), 2024. – 206 с.

20) Маркин, А. В. SQL-программирование в системе «Ред База Данных» : учеб. пособие / А. В. Маркин. – Москва : Ай Пи Ар Медиа, 2024. – 735 с

21) Мерзлякова Е.Ю. Человеко-машинное взаимодействие. / Е.Ю. Мерзляков – Учебное пособие для дистанционного образования. – Сибирский гос. университет телекоммуникаций и информатики, Новосибирск, 2021. – 49 с.

22) Нетесова, О. Ю. Информационные системы и технологии в экономике: учеб. пособие для вузов / О. Ю. Нетесова. — 4-е изд., испр. и доп. — М/: Издательство Юрайт, 2024. — 178 с.

23) Об утверждении Федеральных стандартов бухгалтерского учета ФСБУ 6/2020 «Основные средства» и ФСБУ 26/2020 «Капитальные вложения» // Приказ от 17 сентября 2020 года № 204н.

24) Самуйлов, К.Е. Основы формальных методов описания бизнес-процессов: учеб. пособие / К.Е. Самуйлов, Н.В. Серебренникова, А.В. Чукарин, Н.В. Яркина. – М.: РУДН, 2020 г. – 130 с.

25) Силич, М.П. Общая теория систем: учеб. пособие // М.П. Силич, Л.П. Турунтаев, В.Ф. Тарасенко. – Томск: Эль Контент, 2021. – 119 с.

26) Тагиров, Т.С. СУБД [Текст]: учебно-методическое пособие. – КФУ ИГиНТ, ИММ, Казань, 2023. – 458 с.

27) Тараканов, О. В. Инженерия систем баз данных: учеб. пособие / О. В. Тараканов. – исправленное и дополненное. – Орел: Федеральное государственное казенное военное образовательное учреждение высшего образования «Академия Федеральной службы охраны Российской Федерации», 2021. – 390 с.

28) Холкин, А. В. Построение модели данных с использованием нотации Ричарда Баркера, IDEF1X, UML / А. В. Холкин // Юность и Знания - гарантия Успеха – 2022: сборник научных статей 9-й Международной молодежной

научной конференции : в 3 т., Курск, 15–16 сентября 2022 года. Том 2. – Курск: Юго-Западный государственный университет, 2022. – С. 282-286.

29) Эмбер, В. Рефакторинг баз данных. Эволюционное проектирование [Текст]/ Скотт В. Эмблер, Прамодкумар Дж. Садаладж. – Вильямс, 2020. – 445 с.

30) ECMA-334. C# Language Specification. – ECMA International, 5th edition. – CH-1204 Geneva // December 2017. – 516 p.

31) ISO/IEC/IEEE 29148:2018, International Standard – Systems and software engineering – Life cycle processes – Requirements engineering, 01 October, 2018.

32) Enterprise Architect. Enterprise architect user guide. Sparx Systems Pty Ltd. // November, 2018. – 3362 p.

33) Object Management Group Inc., Unified Modeling Language (UML) Ver. 2.5.1 Infrastructure Specification: OMG Document number: formal/2017-12-05 URL: https://www.omg.org/spec/UML. – December, 2017. – 796 p.

34) SOLID (ООП) [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/SOLID_(объектно_ориентированное_программирование) (дата обращения: 17.10.2025).

35) VisualStudioIDE, Редактор кода. // [электронная документация] // MSDN. Комплекс технической документации по продуктам Microsoft. Режим доступа: https://docs.microsoft.com/ru-ru/windows/uwp/get-started/ (дата обращения: 10.10.2025).

Приложение

Фрагменты исходного кода программы

```
usingSystem;
usingSystem.Windows.Forms;
usingSystem.Collections.Generic;


namespaceFamilyBudget.Model
{
publicabstractclassEntity
    {
publicint ID { get; protectedset; }
publicstring Name { get; protectedset; }
publicstring Description { get; protectedset; }

publicEntity(int ID, string Name, string Description)
        {
this.ID = ID;
this.Name = Name;
this.Description = Description;
}
}


publicclassAccount :Entity
    {
publicstring Number { get; privateset; }
publicdecimal Balance { get; privateset; }
publicAccount(int ID, string Name, string Description, string Number,
decimal Balance = 0)
            : base(ID, Name, Description)
        {
this.Balance = Balance;
this.Number = Number;
        }
publicAccount(int ID, decimal Balance = 0)
            : base(ID, "", "")
        {
this.Balance = Balance;

try
            {
__DS_DB_f_budg.AccountDataTable dt =
new __DS_DB_f_budgTableAdapters.AccountTableAdapter().GetDataByID(ID);
if (dt.Rows.Count> 0)
                {
base.ID = ID;
                    Name = Convert.ToString(dt.Rows[0].ItemArray[1]);
                    Description = Convert.ToString(dt.Rows[0].ItemArray[2]);
                    Number = Convert.ToString(dt.Rows[0].ItemArray[3]);
                }
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

publicstaticList<Account>GetData()
        {
try
```

Продолжение приложения

```
            {
List<Account> entities = newList<Account>();
foreach (var item innew __DS_DB_f_budgTableAdapters.AccountTableAdapter().GetData())
                {
entities.Add(newAccount(
                        item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Number));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<Account>();
            }
        }

publicdecimalIncrease(decimal Value)
        {
            Balance += Value;
return Balance;
        }
publicdecimalDecrease(decimal Value)
            =>Increase((-1) * Value);
    }


publicclassCategory :Entity
    {
publicCategory(int ID, string Name, string Description)
            : base(ID, Name, Description)
        {
        }
publicCategory(int ID) : base(ID, "", "")
        {
try
            {
__DS_DB_f_budg.IncExpTypeDataTable dt =
new __DS_DB_f_budgTableAdapters.IncExpTypeTableAdapter().GetDataByID(ID);
if (dt.Rows.Count> 0)
                {
base.ID = ID;
                    Name = Convert.ToString(dt.Rows[0].ItemArray[1]);
                    Description = Convert.ToString(dt.Rows[0].ItemArray[2]);
                }
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

publicstaticList<Category>GetData()
        {
try
            {
List<Category> entities = newList<Category>();
foreach (var item innew __DS_DB_f_budgTableAdapters.IncExpTypeTableAdapter().GetData())
```

Продолжение приложения

```
            {
entities.Add(newCategory(
                            item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description));
                    }
return entities;
                }
catch (Exception ex)
                {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<Category>();
                }
            }
        }


publicclassDebt :Entity
        {
publicDateTimeCreateDate{ get; privateset; }
publicDateTime? PaidDate{ get; privateset; }
publicDateTimeDueDate{ get; privateset; }
publicdecimal Summary { get; privateset; }
publicboolIsExpence{ get; privateset; }
publicDebt(int ID, string Name, string Description,
DateTimeCreateDate, DateTime? PaidDate, DateTimeDueDate,
decimal Summary, boolIsExpence)
                : base(ID, Name, Description)
            {
this.CreateDate = CreateDate;
this.PaidDate = PaidDate;
this.DueDate = DueDate;
this.Summary = Summary;
this.IsExpence = IsExpence;
            }

publicstaticList<Debt>GetData()
            {
try
                {

List<Debt> entities = newList<Debt>();
foreach (var item innew __DS_DB_f_budgTableAdapters.DebtTableAdapter().GetData())
                    {
entities.Add(newDebt(
                            item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.DateCreate,
Convert.IsDBNull(item.ItemArray[5]) ? null : (DateTime?)item.DatePaid,
item.DateDue,
item.Summary,
item.Direction));
                    }
return entities;
                }
catch (Exception ex)
                {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
```

Продолжение приложения

```
returnnewList<Debt>();
            }
        }
    }


publicabstractclassOperation :Entity
    {
publicDateTime Date { get; privateset; }
publicdecimal Summary { get; privateset; }
publicAccountBindAccount{ get; privateset; }
publicCategoryItsType{ get; privateset; }
publicboolIsExpence{ get; protectedset; }
publicOperation(int ID, string Name, string Description,
DateTime Date, decimal Summary, AccountBindAccount, CategoryItsType)
            : base(ID, Name, Description)
        {
this.Date = Date;
this.Summary = Summary;
this.BindAccount = BindAccount;
this.ItsType = ItsType;
        }
    }


publicclassExpence :Operation
    {
publicExpence(int ID, string Name, string Description,
DateTime Date, decimal Summary, AccountBindAccount, CategoryItsType)
            : base(ID, Name, Description, Date, Summary, BindAccount, ItsType)
        {
IsExpence = true;
        }

publicstaticList<Expence>GetData()
        {
try
            {
List<Expence> entities = newList<Expence>();
foreach (var item innew __DS_DB_f_budgTableAdapters.ExpencesTableAdapter().GetData())
                {
entities.Add(newExpence(
                        item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Date,
item.Summary,
newAccount(item.AccountID),
newCategory(item.TypeID)));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<Expence>();
            }
        }
    }
```

Продолжение приложения

```
publicclassIncome :Operation
    {
publicIncome(int ID, string Name, string Description,
DateTime Date, decimal Summary, AccountBindAccount, CategoryItsType)
            : base(ID, Name, Description, Date, Summary, BindAccount, ItsType)
        {
IsExpence = false;
        }

publicstaticList<Income>GetData()
        {
try
            {
List<Income> entities = newList<Income>();
foreach (var item innew __DS_DB_f_budgTableAdapters.IncomesTableAdapter().GetData())
                {
entities.Add(newIncome(
                        item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Date,
item.Summary,
newAccount(item.AccountID),
newCategory(item.TypeID)));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<Income>();
            }
        }
    }


publicclassPeriodic :Operation
    {
publicDateTimeCompleteDate{ get; privateset; }
publicintPeriodMonth{ get; privateset; }
publicPeriodic(int ID, string Name, string Description,
decimal Summary, AccountBindAccount, CategoryItsType,
boolIsExpence, DateTimeCompleteDate, DateTime StartDate, intPeriodMonth)
            : base(ID, Name, Description, StartDate, Summary, BindAccount, ItsType)
        {
this.IsExpence = IsExpence;
this.PeriodMonth = PeriodMonth;
this.CompleteDate = CompleteDate;
        }

publicstaticList<Periodic>GetData()
        {
try
            {
List<Periodic> entities = newList<Periodic>();
foreach (var item innew __DS_DB_f_budgTableAdapters.PeriodicTableAdapter().GetData())
                {
entities.Add(newPeriodic(
                        item.ID,
item.Name,
```

Продолжение приложения

```
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Summary,
newAccount(item.AccountID),
newCategory(item.TypeID),
item.Direction,
item.CompleteDate,
item.StartDate,
item.Period));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<Periodic>();
            }
        }
    }


publicabstractclassPlanOperation :Entity
    {
publicdecimal Summary { get; privateset; }
publicPlanOperation(int ID, string Name, string Description, decimal Summary)
            : base(ID, Name, Description)
        {
this.Summary = Summary;
        }
    }


publicclassPlanIncome :PlanOperation
    {
publicDateTime Date { get; privateset; }
publicPlanIncome(int ID, string Name, string Description, decimal Summary, DateTime Date)
            : base(ID, Name, Description, Summary)
        {
this.Date = Date;
        }

publicstaticList<PlanIncome>GetData()
        {
try
            {
List<PlanIncome> entities = newList<PlanIncome>();
foreach (var item innew __DS_DB_f_budgTableAdapters.PlanIncomeTableAdapter().GetData())
                {
entities.Add(newPlanIncome(
                        item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Summary,
item.Date));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
```

Продолжение приложения

```
returnnewList<PlanIncome>();
            }
        }
    }

publicclassPlanExpence :PlanOperation
    {
publicPlanExpence(int ID, string Name, string Description, decimal Summary)
            : base(ID, Name, Description, Summary)
        {
        }

publicstaticList<PlanExpence>GetData()
        {
try
            {
List<PlanExpence> entities = newList<PlanExpence>();
foreach (var item innew __DS_DB_f_budgTableAdapters.PlanExpenceTableAdapter().GetData())
                {
entities.Add(newPlanExpence(
                        item.ID,
item.Name,
Convert.IsDBNull(item.ItemArray[2]) ? "" :item.Description,
item.Summary));
                }
return entities;
            }
catch (Exception ex)
            {
MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

returnnewList<PlanExpence>();
}
}
    }
}
```