**2 Проектирование задачи**

**2.1 Моделирование проекта**

Моделирование – процесс изучения моделей реально существующих объектов, процессов или явлений с целью получения объяснений и предсказания интересующих явлений.

Для разработки объективно-ориентированных систем применяется язык моделирования UML. UML (Unified Modeling Language) – язык моделирования, используемый для визуализации, проектирования и документирования систем и процессов. Является стандартом в области разработки программного обеспечения и широко применяется инженерами и аналитиками при работе над проектами.

Моделирование диаграмм в UML позволяет выразить различные аспекты системы в виде графических диаграмм. Такие диаграммы способствуют лучшему пониманию, анализу и коммуникации различных аспектов системы между заинтересованными сторонами.

**2.1.1 Описание проекта с точки зрения функционала, вариантов использования, структуры и поведения**

Разрабатываемое веб-приложение для финансового учета организации предоставляет пользователям возможность просмотра финансовой информации, изучения комментариев к операциям, поиска и фильтрации финансовых данных по различным критериям, а также добавления отчетов или операций в избранное. Для администратора реализован функционал по добавлению, редактированию и удалению финансовой информации (операций, счетов, категорий), а также управлению пользователями.

Система поддерживает регистрацию и авторизацию пользователей, восстановление пароля, просмотр ключевых финансовых показателей, а также мультиязычный интерфейс.

Диаграмма вариантов использования (Use Case diagram) отражает основные сценарии взаимодействия пользователей и администратора с системой финансового учета. Каждый сценарий описывает последовательность действий, приводящих к достижению определенного результата. Диаграмма предоставлена на рисунке 2.1.

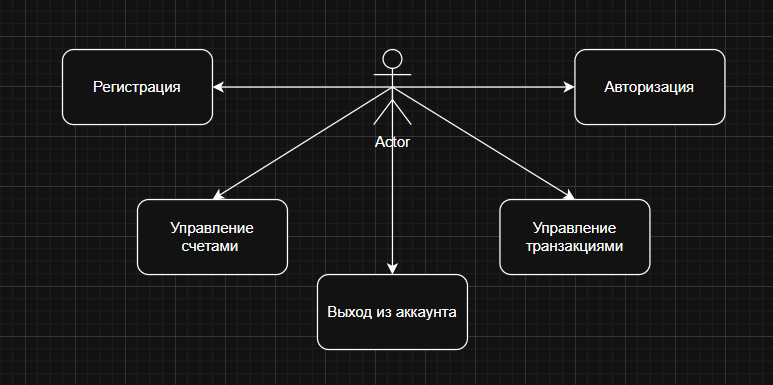


Рисунок 2.1 – Диаграмма вариантов использования

Источник: собственная разработка

Для моделирования процесса выполнения операций используется диаграмма деятельности. Диаграмма деятельности иллюстрирует поток событий при взаимодействии пользователя и администратора с системой финансового учета.

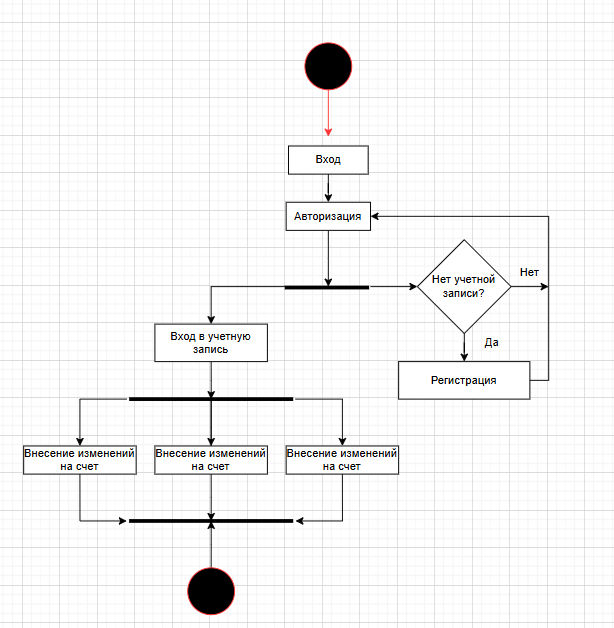


Рисунок 2.2 – Диаграмма деятельности

Источник: собственная разработка

Диаграмма деятельности представляет собой визуальное отображение основных процессов и взаимодействий в системе финансового учета. Начальная точка процесса - доступ к веб-приложению пользователем. Система проверяет статус авторизации пользователя. При отсутствии авторизации отображается окно входа с возможностью регистрации.

Процесс регистрации включает ввод персональных данных, их валидацию и сохранение в базе данных. После успешной авторизации пользователя происходит загрузка его рабочего пространства, например, списка последних финансовых операций или счетов.

Пользователи имеют возможность взаимодействовать с финансовыми данными: просматривать детальную информацию по операциям или счетам, добавлять отчеты или операции в избранное, оставлять комментарии к операциям. Авторизованные пользователи получают доступ к персональному профилю и списку избранных элементов.

Административная часть системы предоставляет расширенные возможности управления. Администраторы могут управлять пользователями, модерировать комментарии к операциям и управлять финансовой информацией (добавление, редактирование, удаление операций, счетов, категорий). Система обеспечивает контроль доступа, обработку ошибок и ведение логов всех действий.

Все процессы в системе взаимосвязаны и обеспечивают полный цикл взаимодействия между пользователями, администраторами и самой системой. Диаграмма деятельности наглядно демонстрирует последовательность действий, условия перехода между различными состояниями и параллельные процессы, происходящие в системе.

Диаграмма классов определяет основные сущности системы финансового учета, их атрибуты и связи между ними. В системе реализованы классы: Пользователь, ФинансоваяОперация, Счет, Категория, Отчет, ЖурналАудита. Каждый класс характеризуется набором атрибутов и операций. Диаграмма классов предоставлена на рисунке 2.3.

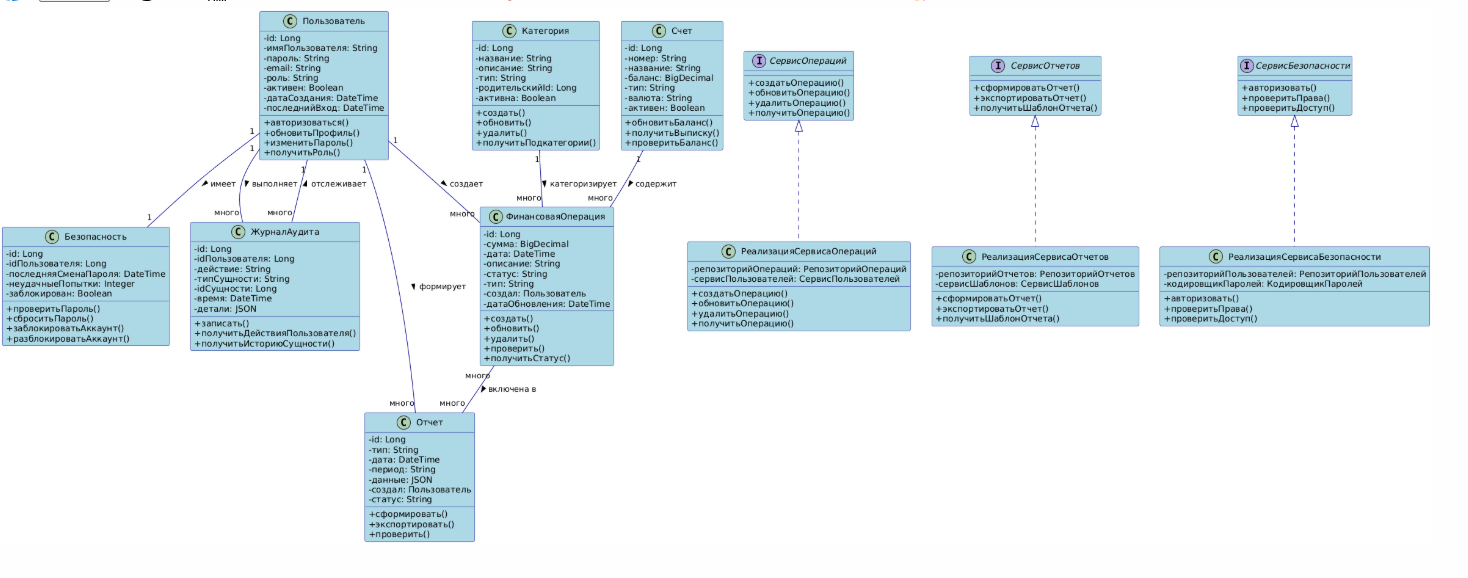


Рисунок 2.3 – Диаграмма классов

Источник: собственная разработка

Диаграмма классов показывает главные части нашей системы и как они связаны. В системе есть несколько основных частей: Пользователь, Финансовая Операция, Счет, Категория и Отчет.

Класс "Пользователь" — это те, кто пользуется системой. У него есть такие данные, как почта, пароль и логин. Пользователи могут регистрироваться, входить в систему и работать со своими финансами.

Класс "Финансовая Операция" — это запись о том, куда или откуда деньги пришли или ушли. У каждой операции есть сумма, дата, описание, тип и статус. Каждая операция связана с определенным Пользователем, Категорией и Счету.

Класс "Счет" — это место, где хранятся деньги пользователя. У него есть номер, название, текущий баланс и валюта. Каждый Счет принадлежит определенному Пользователю.

Класс "Категория" нужна, чтобы понять, на что идут деньги или откуда приходят. У нее есть название и описание. Категории помогают группировать операции.

Класс "Отчет" — это документ, который система создает, чтобы показать финансовые итоги за период. У него есть тип, дата создания и сами данные отчета. Отчеты создают Пользователи.

Атрибуты классов представлены в соответствии с таблицей 2.1

Таблица 2.1 – Атрибуты классов

|  |  |  |
| --- | --- | --- |
| Название класса | Имя атрибута | Тип данных |
| Пользователь | Email | Текст |
| Логин | Текст |
| Пароль | Текст |
| Финансовая Операция | Сумма | Число с точкой |
| Дата | Дата/Время |
| Описание | Текст |
| ID Пользователя | Число |
| Счет | Номер счета | Число |
| Название | Текст |
| Баланс | Число с точкой |
| Валюта | Текст |
| Категория | Название | Текст |
| Описание | Текст |
| ID Родителя | Число |

Источник: собственная разработка

Операции классов представлены в соответствии с таблицей 2.2

Таблица 2.2 – Операции классов

|  |  |  |
| --- | --- | --- |
| Название класса | Действие | Описание действия |
| Пользователь | Зарегистрироваться | Создание учётной записи |
| Войти | Вход в систему |
| Добавить транзакцию | Добавление транзакции |
| Счет | Добавить транзакцию | Добавление транзакции |

Источник: собственная разработка

Связи на диаграмме классов:

Пользователь – ФинансоваяОперация – ассоциация. Пользователь связан с Финансовыми Операциями, которые он создает или просматривает. Кратность 1-n: один Пользователь может создать или просмотреть множество Финансовых Операций.

Пользователь – Счет – ассоциация. Пользователь связан со Счетами, которыми он управляет. Кратность 1-n: один Пользователь может иметь несколько Счетов.

Пользователь – Отчет – ассоциация. Пользователь связан с Отчетами, которые он формирует. Кратность 1-n: один Пользователь может сформировать множество Отчетов.

Категория – Категория – ассоциация. Категории могут быть связаны друг с другом для создания иерархии (подкатегорий). Кратность 1-n: одна Категория может иметь множество подкатегорий.

**2.1.2 Организация данных**

Схема показывает, как устроены данные для веб-приложения. Система использует базу данных. В ней хранятся главные части: Пользователь, Финансовая Операция, Счет, Категория, Отчет и Журнал Аудита.

Пользователь хранит: уникальный номер, имя для входа, пароль, адрес почты, роль, дату создания записи и дату последнего входа. Эти данные нужны для входа и понимания, кто пользуется системой.

Финансовая Операция хранит: уникальный номер, сумму, дату и время, описание, статус и тип операции. Каждая операция связана с Пользователем, который ее создал, с Категорией и со Счетом.

Счет хранит: уникальный номер, номер счета, название, текущий баланс, тип счета и валюту. Каждый Счет принадлежит Пользователю.

Категория нужна для деления операций на группы. Хранит: уникальный номер, название и описание. Категории могут быть главными и подчиненными.

Отчет хранит информацию о готовых отчетах. Содержит: уникальный номер, тип отчета, дату создания, период, сами данные отчета и статус. Каждый Отчет создан Пользователем.

Журнал Аудита записывает все важные действия в системе. Хранит: уникальный номер записи, номер Пользователя, что сделано, над чем сделано, номер этой части, время и детали.

Части данных связаны между собой. Например, Пользователь делает Финансовые Операции. Операции относятся к Категориям и проходят по Счетам. Пользователь создает Отчеты. Действия Пользователя записываются в Журнал Аудита.

Как связаны части данных, показано на схеме Сущность-Связь.

Диаграмма сущность-связь предоставлена в соответствии с рисунком 2.4

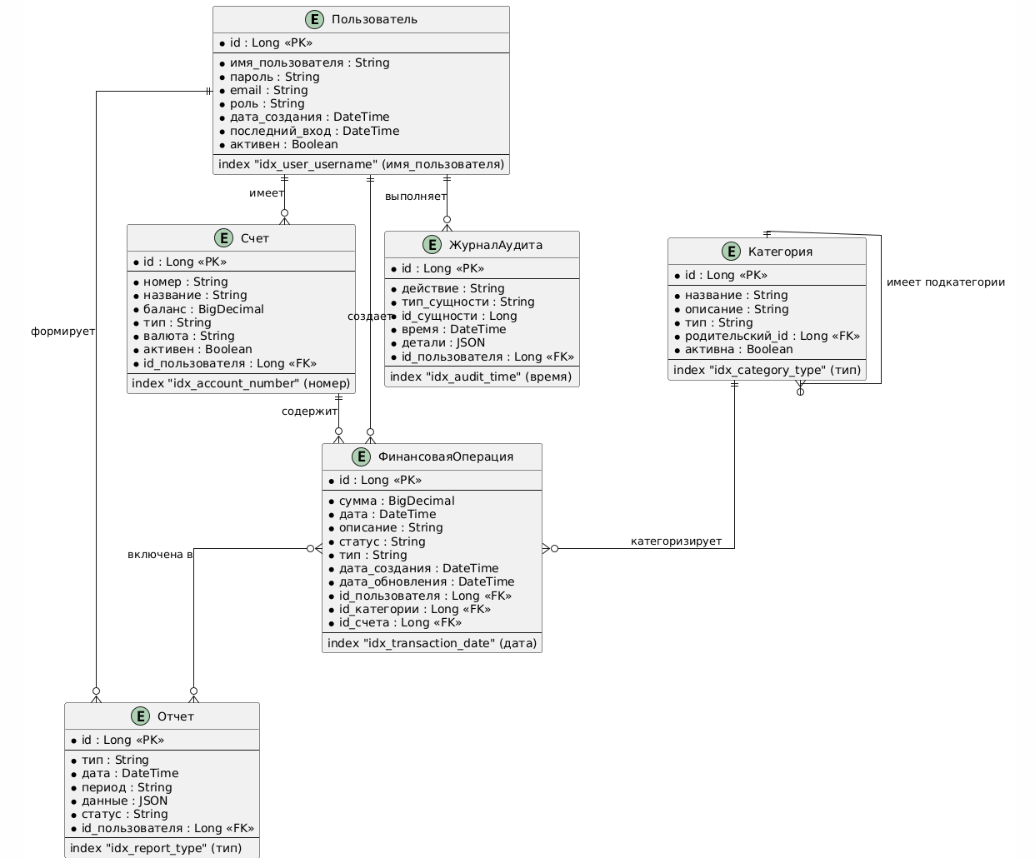


Рисунок 2.4 – Диаграмма сущность-связь

Источник: собственная разработка

Связи между частями данных показывают, что Пользователь может создавать Финансовые Операции, иметь Счета, формировать Отчеты и выполнять другие действия. Такое устройство данных помогает хранить и обрабатывать информацию о пользователях и финансах. Оно также поддерживает главные функции системы: регистрацию, вход, создание и просмотр финансовых данных.

**2.2 Описание системы меню**

В рамках дипломного проекта нужна разработка удобного интерфейса. Интерфейс обеспечит работу пользователей с приложением финансового учета. Особое внимание уделят структуре интерфейса для простоты использования финансовых данных.

Программный интерфейс (API) — набор функций. Функции одни части программы дают другим.

Термин "интерфейс" обозначает инструменты для связи двух систем. Системы могут быть людьми или программами.

Веб-интерфейс — набор веб-страниц. Страницы дают интерфейс для работы с сервисом через интернет-браузер. Веб-интерфейсы широко распространены. Это делает финансовое приложение доступным.

Меню — часть интерфейса. Оно позволяет выбрать опции или разделы приложения. Меню обычно выглядит как кнопки или ссылки. Нажатие переводит в нужный раздел. Меню — важная часть интерфейса. Оно дает структуру и удобство навигации.

Меню нужно для перехода по разделам приложения финансового учета. Меню сайта — группа ссылок. Ссылки помогают переходить между страницами с операциями, отчетами, счетами.

Есть разные виды меню:

Выпадающее меню: при наведении или клике появляются подпункты. Это удобно для группировки функций и экономии места.

Раскрывающееся меню: при клике открываются дополнительные ссылки.

Система меню для пользователя показана на схеме, рисунок 2.5. Схема показывает, как пользователь получает доступ к главным функциям финансового приложения.

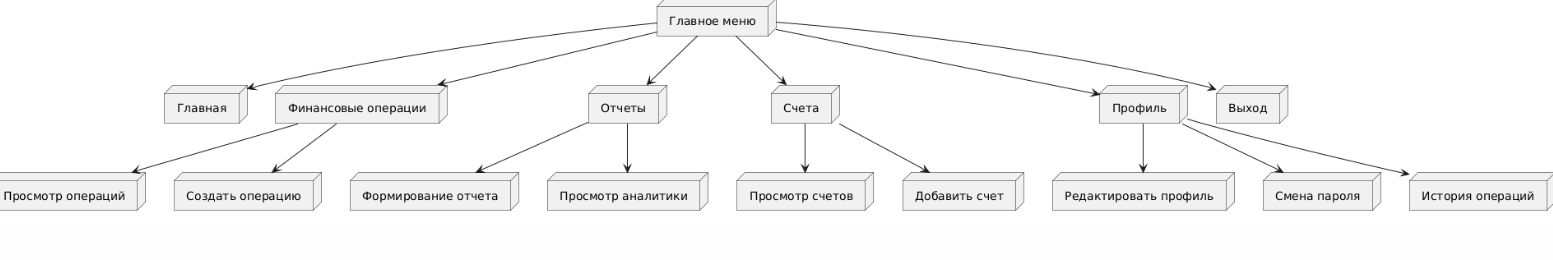


Рисунок 2.4 – Схема системы меню пользовательского модуля

Источник: собственная разработка

Схема системы меню Пользователя показывает, как устроена навигация веб-приложения. Главное меню находится сверху на каждой странице. Оно дает быстрый доступ к основным разделам.

Пункт "Главная" ведет на стартовую страницу. Там может быть общая информация или сводка по финансам. Раздел "Финансовые операции" открывает список операций. Операции можно искать и фильтровать. Пункт "Отчеты" позволяет формировать и просматривать разные финансовые отчеты. Раздел "Счета" показывает список всех счетов Пользователя и их балансы. Пункт "Избранное" дает доступ к отчетам или операциям, которые Пользователь отметил как важные. Этот раздел доступен только после входа в систему. Раздел "Профиль" позволяет смотреть и менять личные данные Пользователя. Здесь же можно управлять настройками. Доступно только вошедшим Пользователям.

Пункт "Регистрация/Вход" виден тем, кто не вошел в систему. Он ведет к формам для регистрации или входа.

Такое устройство меню помогает Пользователю легко ориентироваться. Быстро находится доступ к нужным функциям. Меню меняется в зависимости от того, вошел Пользователь в систему или нет. Это соответствует тому, как сделано приложение.

Система меню главного окна приложения администраторского модуля указана в соответствии с рисунком 2.6

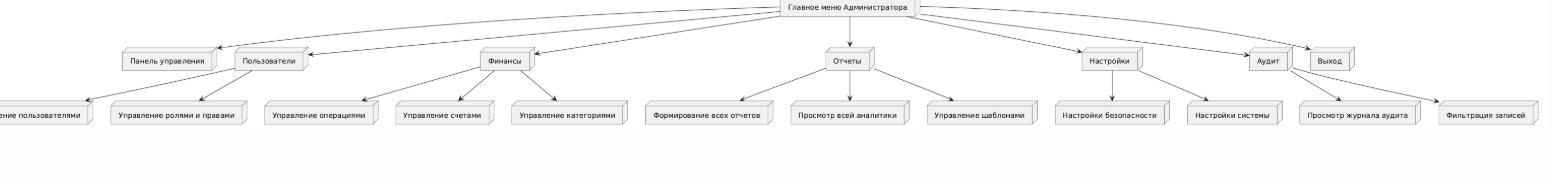


Рисунок 2.6 – Схема системы меню главного окна администратора

Источник: собственная разработка

Схема показывает структуру меню главного окна административного модуля веб-приложения. В меню администратора представлены основные разделы для управления финансовой информацией и пользователями системы. Пункт "Главная" обеспечивает переход на стартовую страницу административной панели. Там отображается общая информация и основные действия.

Раздел "Финансы" предназначен для управления финансовой информацией. Это включает добавление, редактирование и удаление операций, счетов и категорий.

Пункт "Пользователи" предназначен для управления записями пользователей. Это включает их блокировку, разблокировку и изменение прав доступа. Раздел "Комментарии" обеспечивает просмотр и модерацию комментариев к финансовым операциям. Это включает удаление при необходимости. Пункт "Импорт данных" дает возможность массово добавлять финансовые данные. Это может быть загрузка файлов или связь с другими источниками.

Структура меню нужна для быстрого доступа ко всем главным функциям административного модуля. Это повышает эффективность управления системой и делает проще выполнение рутинных задач. Каждый раздел меню — это отдельная группа функций. Это помогает логично организовать навигацию в административной панели.

**2.3 Выбор и обосновании среды разработки**

Для разработки веб-приложения, предназначенного для автоматизации процессов финансового учета, применен технологический стек, включающий язык программирования Java, фреймворк Spring Boot для реализации серверной части и шаблонизатор Thymeleaf для формирования пользовательского интерфейса.

Выбор Java в качестве основного языка разработки обусловлен высокой производительностью, надежностью и безопасностью языка Java, что является критически важным для систем, обрабатывающих финансовые данные. Обширная стандартная библиотека языка Java и зрелая экосистема фреймворков и инструментов способствуют эффективной разработке корпоративных приложений.

Spring Boot использован для построения серверной части приложения. Данный фреймворк значительно упрощает процесс конфигурирования и развертывания Spring-приложений, предоставляя функции автоматической настройки компонентов, встроенные серверы приложений и механизмы управления зависимостями. Использование Spring Boot обеспечивает быструю разработку и высокую готовность приложения к промышленной эксплуатации.

Для реализации клиентской части, ответственной за взаимодействие с пользователем, применен шаблонизатор Thymeleaf. Thymeleaf позволяет создавать динамические веб-страницы на основе HTML-шаблонов, отображая актуальную финансовую информацию и обеспечивая ввод данных. Интеграция Thymeleaf со Spring Boot упрощает процесс разработки пользовательского интерфейса. Возможное использование JavaScript направлено на повышение интерактивности элементов интерфейса и реализацию динамических обновлений данных без полной перезагрузки страниц.

В качестве системы управления базами данных выбрана MySQL. Эта реляционная СУБД характеризуется надежностью и производительностью, что делает MySQL оптимальным решением для хранения структурированных финансовых данных. MySQL поддерживает ключевые функции, необходимые для обеспечения целостности и безопасности финансовых операций, включая транзакции, внешние ключи, индексы для оптимизации запросов и подготовленные запросы для предотвращения SQL-инъекций.

Помимо основных компонентов, в проекте задействованы Spring Data JPA для упрощения реализации уровня доступа к данным и взаимодействия с базой данных посредством объектно-реляционного маппинга, Spring Security для построения надежной системы безопасности, охватывающей механизмы аутентификации и авторизации пользователей, что необходимо для защиты конфиденциальной финансовой информации, Maven как инструмент управления проектом и зависимостями проекта, стандартизирующий процесс сборки и развертывания, а также Lombok для автоматической генерации шаблонного кода, сокращая объем кода классов сущностей и DTO. Применение данного технологического стека обеспечивает достижение требуемых характеристик надежности, безопасности, производительности и масштабируемости, необходимых для успешной реализации веб-приложения по финансовому учету.