МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Кафедра Прикладная математика и информатика
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Цифровая трансформация бизнеса
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка проекта по внедрению системы управления рисками информационной безопасности»

Обучающийся А. Г. Андреев
(Инициалы Фамилия) (личная подпись)

Руководитель доктор социологических наук, доцент, Е. В. Желнина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2025

Оглавление

Аннотация

Выпускная квалификационная работа посвящена процессу внедрения системы управления рисками информационной безопасности. Цель работы заключается в создании демонстрационного решения, которое показывает, как эффективно управлять рисками ИБ, обеспечивая защиту критически важных активов и соответствие требованиям современных стандартов безопасности.

Основные задачи работы включают: функциональное моделирование системы, разработку архитектуры системы, проектирование базы данных и реализацию алгоритмов обработки. В процессе разработки использовались современные технологии и инструменты, включая язык программирования C#, фреймворк ASP.NET Core, среду разработки Visual Studio, а также актуальные стандарты и методики.

Целью работы является практическое применение знаний и навыков, полученных в ходе изучения дисциплин программы бакалавриата по направлению «Прикладная информатика» профиля «Цифровая трансформация бизнеса». В ходе работы применены изученные методологии проектирования, разработки и управления требованиями к программному обеспечению, продемонстрировано умение интегрировать теоретические знания в практическую реализацию. Результатом работы является информационная система, которое решает актуальные задачи в области информационной безопасности, и подтверждает способность эффективно использовать изученные подходы и инструменты для создания корпоративных систем.

Работа структурно разделена на три основные главы, каждая из которых последовательно раскрывает различные аспекты.

Первая глава посвящена функциональному моделированию системы. В ней проводится анализ предметной области, определяются ключевые цели и задачи разрабатываемого решения. Методологическая база исследования включает подходы IDEF0 и UML.

Вторая глава фокусируется на логическом проектировании системы. Важным элементом является проектирование реляционной базы данных с нормализацией до нормальной формы 3. В главе также определяются алгоритмы обработки данных и разрабатываются модели взаимодействия между компонентами системы.

Третья глава посвящена физическому проектированию и практической реализации системы. В ней обосновывается выбор технологических решений, описывается реализация бизнес-алгоритмов и разрабатывается план внедрения системы.

Кроме того, в работе приводится расчет экономической эффективности проекта на начальном этапе и оптимизация затрат на этапе реализации.

Перечень сокращений и обозначений

ИТ – Информационные технологии

ИБ – Информационная безопасность

ПО – Программное обеспечение

ИС – Информационная система

ФЗ – федеральный закон

ГОСТ – государственный стандарт

СУРИБ – Система управления рисками информационной безопасности

ФСТЕК Федеральная служба по техническому и экспортному контролю

IDEF0 - Integration Definition for Function Modeling

DFD – Data flow diagram

UML – Unified Modeling Language

Введение

Внедрение систем управления рисками информационной безопасности обычно включает несколько ключевых этапов. На начальном этапе проводится анализ текущих процессов и требований бизнеса, что позволяет определить функциональные требования к системе. Далее выполняется проектирование архитектуры и разработка решения. Важным этапом является интеграция системы с существующей ИТ-инфраструктурой, включая подключение к службе каталогов, системам мониторинга и антивирусным решениям. После внедрения проводится обучение сотрудников и настройка процессов мониторинга и отчетности. Успешные кейсы внедрения подобных систем показывают, что ключевыми факторами успеха являются вовлеченность руководства, четкое планирование этапов проекта и постоянное взаимодействие между разработчиками и бизнес-пользователями. Это позволяет минимизировать риски внедрения и обеспечить быструю окупаемость инвестиций.

Актуальность темы исследования обусловлена стремительным развитием информационных технологий и увеличением числа угроз информационной безопасности в современном мире [2]. Организации всех уровней сталкиваются с необходимостью защиты своих информационных активов от кибератак, утечек данных и других рисков, которые могут привести к значительным финансовым и репутационным потерям. В условиях цифровой трансформации и роста зависимости бизнеса от информационных систем управление рисками становится критически важным элементом обеспечения устойчивости и конкурентоспособности организации.

Основной бизнес-целью проекта является повышение уровня защищенности информационных активов организации. Проект направлен на минимизацию финансовых и репутационных потерь, связанных с кибератаками, утечками данных и другими угрозами информационной безопасности. Система позволит оптимизировать процессы идентификации

рисков, их анализа и управления мерами контроля, что снизит затраты на ручное администрирование и повысит оперативность реагирования на инциденты. Кроме того, внедрение системы обеспечит соответствие требованиям отраслевых стандартов и регуляторных актов, что важно для поддержания доверия клиентов и партнеров. В долгосрочной перспективе проект способствует укреплению конкурентных преимуществ организации за счет повышения устойчивости к киберугрозам.

Объектом исследования являются системы управления рисками информационной безопасности.

Предметом исследования выступает проект по внедрению информационной системы управления рисками информационной безопасности.

Методологическую основу работы составляют:

Теоретические методы: анализ научной литературы, нормативных документов и стандартов в области информационной безопасности.

Практические методы: функциональное, логическое и физическое моделирование, анализ рисков, тестирование и экономический анализ.

Структура работы включает аннотацию, введение, три главы, заключение и приложение. Первая глава посвящена функциональному моделированию системы. Вторая глава излагает этап логического проектирования. В третьей главе описывается финальный этап физического проектирования, а также план внедрения системы. В заключении подводятся итоги работы и формулируются перспективы по дальнейшему развитию системы. Бакалаврская работа состоит из 95 страниц и включает 32 иллюстрации, 29 таблиц, 35 источников.

Глава 1 Функциональное моделирование

1.1 Цели и задачи функционального моделирования

Внедрение системы управления рисками ИБ требует тщательной проработки на всех этапах проектирования, начиная с определения функциональных требований и заканчивая физической реализацией. Первым шагом в этом процессе является функциональное моделирование, которое позволяет определить ключевые функции системы, их взаимодействие и роль в общем процессе управления рисками.

Этап функционального моделирования направлен на создание структурированного представления системы, которое отражает её основные задачи, процессы и взаимодействия. Этот этап позволяет выявить и описать функции, которые система должна выполнять для достижения поставленных целей, а также определить, как эти функции будут взаимодействовать между собой и с внешними элементами. В рамках данной главы рассматриваются цели и задачи функционального моделирования, идентификация ключевых функций системы, а также визуализация процессов с помощью диаграмм потоков данных (DFD) и UML Use-case диаграмм.

Результатом функционального моделирования становится четкое описание функциональных возможностей системы управления рисками ИБ, что является основой для последующих этапов проектирования — логического и физического. Этот этап позволяет заложить фундамент для создания эффективной и адаптивной системы, способной своевременно выявлять, оценивать и минимизировать риски, связанные с информационной безопасностью.

Цели:

- описание предметной области. Определение терминов и понятий информационной безопасности, обзор методологий;

- определение функциональности. Выявление и описание всех функций, которые система должна выполнять для удовлетворения потребностей пользователей;

- формализация требований. Преобразование бизнес-требований в структурированную модель, отражающую процессы обработки данных и взаимодействие компонентов;

- расчет экономической эффективности проекта.

- обеспечение понятности. Представление системы в виде наглядных моделей.

Задачи:

- идентификация входных и выходных данных системы;
- определение процессов обработки данных;
- построение UML диаграмм;
- учет требований к производительности, безопасности и удобству использования.

1.1.1 Определения понятий рисков информационной безопасности

В современных условиях размер организации выступает ключевым фактором, определяющим сложность управления информационными активами и формирующим специфические требования к системам защиты информации. Так, малые предприятия, как правило, характеризуются относительно простыми информационными системами и локальной структурой управления, что позволяет им использовать базовые средства защиты. Средние и крупные организации, обладающие более сложными информационными инфраструктурами, требуют применения комплексных систем безопасности, включающих многоуровневую защиту и специализированные решения для различных типов активов [11].

В таблице 1 представлены размеры предприятий, структура управления и масштаб ИС.

Таблица 1 – Размер предприятий, структура управления и масштаб ИС

Размер	Количество сотрудников	Структура управления	Информационные системы
Малые	До 50	Простая иерархия	Локальные системы
Средние	50-500	Развитая структура	Комплексные системы
Крупные	Более 500	Сложная иерархия	Масштабные системы

Особого внимания заслуживает тот факт, что современные организации часто демонстрируют гибридную структуру, сочетая в себе различные характеристики из разных классификационных групп. Это может проявляться в виде совмещения производственной и сервисной деятельности, наличия как традиционных, так и цифровых активов, а также использования различных форм организации бизнес-процессов. Подобная гибридность создает дополнительные вызовы для управления информационными активами, требуя разработки интегрированных подходов к обеспечению информационной безопасности и учета многообразия факторов при выборе защитных механизмов.

В контексте требований к информационной безопасности организации применяют комплексный подход к выбору защитных решений. Малые предприятия чаще всего ограничиваются базовыми средствами защиты, такими как антивирусное ПО, простые межсетевые экраны и системы резервного копирования. Средние организации внедряют более продвинутые решения, включая системы обнаружения вторжений (IDS), управления доступом (IAM) и защиты от утечек данных (DLP) [18].

Крупные компании, особенно работающие в чувствительных отраслях, используют полномасштабные системы безопасности, включающие SIEM-решения для мониторинга и анализа событий безопасности, комплексные системы шифрования данных, многоуровневую аутентификацию и продвинутые средства защиты периметра [28]. При этом гибридная структура современных организаций требует особого внимания к интеграции различных систем защиты, что достигается через применение унифицированных

платформ управления безопасностью и стандартизированных протоколов взаимодействия между компонентами защиты.

Риски информационной безопасности – это вероятность того, что угрозы (или комбинации угроз) могут нанести ущерб информационным активам организации или личности, нарушив их конфиденциальность, целостность или доступность. Риски включают как вероятность инцидента безопасности, так и последствия этого инцидента [12].

Ущерб от реализации атаки подразделяется на прямой и непрямой. Прямой ущерб включает в себя непосредственные и легко прогнозируемые потери, такие как утрата интеллектуальной собственности, разглашение коммерческой тайны, повреждение активов, судебные издержки и штрафные санкции. Непрямой ущерб выражается в качественных и косвенных потерях, включая снижение операционной эффективности, отток клиентов, ухудшение репутации компании, а также недополученную прибыль. В международной практике также выделяют тотальный риск, возникающий при полном отсутствии защитных мер, и остаточный риск, сохраняющийся даже после внедрения средств защиты [16].

1.1.2 Обзор технических средств информационной безопасности

Технические средства защиты информации включают комплекс программно-аппаратных решений. Межсетевые экраны осуществляют фильтрацию сетевого трафика на основе заданных правил безопасности. Системы обнаружения и предотвращения вторжений обеспечивают мониторинг сетевой активности и реагирование на подозрительные события. Антивирусное программное обеспечение защищает от вредоносных программ, а системы контроля доступа ограничивают несанкционированный доступ к информационным ресурсам [24]. Таблица 2 содержит описание технических средств информационной безопасности.

Таблица 2 – Технические средства информационной безопасности

Тип решения	Описание	Для чего используется
Antivirus (Антивирусные комплексы)	Программное обеспечение, предназначенное для защиты от вирусов, троянов, червей и другого вредоносного ПО.	Риски, связанные с вирусными угрозами, вредоносным ПО и атаками, включая троянские программы, червей и шпионские программы.
Firewall (Межсетевые экраны)	Аппаратные и программные решения для фильтрации и контроля входящего и исходящего трафика между сетями.	Риски несанкционированного доступа, атак с внешних источников (например, DDoS-атаки, манипуляции с трафиком).
IDS/IPS (Системы обнаружения и предотвращения вторжений)	Системы, которые мониторят трафик сети или систему для выявления подозрительных действий и атак.	Риски несанкционированного доступа, атак на сети, утечек данных, взломов и других вторжений в систему.
SIEM (Системы управления событиями и информацией безопасности)	Комплексные системы для сбора, анализа и корреляции событий безопасности в реальном времени.	Риски, связанные с недостаточной видимостью угроз в системе, неэффективным мониторингом и реагированием на инциденты.
DLP (Data Loss Prevention)	Системы предотвращения утечек данных, предназначенные для мониторинга и контроля перемещения конфиденциальной информации.	Риски утечек конфиденциальной информации, компрометации данных и их незаконной передачи.
VPN (Virtual Private Network)	Технология создания защищённого канала связи между пользователем и сетью.	Риски перехвата данных, атак MITM (Man-in-the-Middle), несанкционированного доступа при передаче данных по открытым сетям.
PKI (Инфраструктура открытых ключей)	Система, обеспечивающая защиту данных через использование криптографических ключей для шифрования и аутентификации.	Риски, связанные с подделкой цифровых подписей, утечкой или компрометацией ключей, несанкционированным доступом к зашифрованной информации.
HSM (Hardware Security Module)	Аппаратные модули для защиты криптографических ключей и выполнения криптографических операций.	Риски утечек ключей шифрования, компрометации криптографических операций, подделки цифровых подписей.
Tokenization Systems (Системы токенизации)	Технологии, заменяющие чувствительные данные (например, номер кредитной карты) на случайные токены.	Риски утечек и кражи чувствительных данных, таких как номера кредитных карт, идентификационные номера.

Продолжение Таблицы 2

PAM (Privileged Access Management)	Решения для управления привилегированным доступом и контроля за действиями пользователей с правами администратора.	Риски злоупотребления правами доступа, несанкционированного изменения конфигураций, инсайдерских угроз.
Endpoint Protection (Защита конечных устройств)	Программные решения, обеспечивающие защиту для рабочих станций, мобильных устройств и других конечных точек.	Риски заражения вредоносным ПО, несанкционированного доступа к данным, угрозы от мобильных устройств.
Backup Systems (Резервное копирование)	Системы для создания резервных копий данных с возможностью их восстановления в случае инцидентов.	Риски потери данных, повреждения или их уничтожения, сбоев в работе системы.
Access Control Systems (Системы контроля доступа)	Решения для контроля и ограничения доступа к физическим и информационным ресурсам.	Риски несанкционированного физического доступа, утери или кражи оборудования, злоупотребления правами доступа.
Authentication Systems (Системы аутентификации)	Технологии, такие как биометрия, двухфакторная аутентификация, для подтверждения подлинности пользователей.	Риски несанкционированного доступа, подделки учетных данных, фишинга.
WAF (Web Application Firewall)	Специализированные межсетевые экраны для защиты веб-приложений от атак, таких как SQL-инъекции, XSS и другие уязвимости.	Риски атак на веб-приложения, такие как SQL-инъекции, XSS, CSRF и другие веб-угрозы.
RAT (Remote Access Tools)	Инструменты для удалённого доступа и контроля за компьютерами, используемые для обеспечения безопасной работы с удалёнными системами.	Риски несанкционированного удалённого доступа, злоупотребления привилегиями, шпионских атак.

Все эти системы работают в комплексе, дополняя друг друга для обеспечения максимальной защиты информационных активов организации. Выбор конкретных систем зависит от специфики бизнеса, масштаба организации и особенностей её информационной инфраструктуры.

1.1.3 Обзор методологий риск-менеджмента

Кратко рассмотрим различные методологии риск-менеджмента предметной области:

- ГОСТ Р 50922-2006. Стандарт, который определяет требования к управлению рисками в области информационной безопасности. Ориентирован на оценку и управление рисками в информационных системах. Включает в себя этапы идентификации, анализа и оценки рисков. Применим для организаций различных форм собственности [17];
- ФЗ-152 "О персональных данных" Федеральный закон, регулирующий обработку персональных данных и защиту прав субъектов данных. Устанавливает требования к безопасности персональных данных и управлению рисками, связанными с их обработкой. Обязывает организации проводить оценку рисков при обработке персональных данных. Включает в себя требования к внедрению мер по защите данных [33];
- Методические рекомендации по оценке рисков информационной безопасности - рекомендации, разработанные Федеральной службой по техническому и экспортному контролю для оценки рисков в области информационной безопасности. Ориентированы на практическое применение в организациях. Включают в себя методику идентификации угроз и уязвимостей. Предоставляют рекомендации по выбору мер по снижению рисков [35];
- Система управления информационной безопасностью по требованиям ISO/IEC 27001. Адаптация международного стандарта ISO/IEC 27001 для российских организаций. Включает в себя требования к созданию и внедрению системы управления информационной безопасностью. Ориентирована на соответствие требованиям российского законодательства. Поддерживает интеграцию с другими стандартами и методологиями [5];

- Рекомендации по управлению рисками в области информационной безопасности [17] Рекомендации, разработанные Роскомнадзором для организаций, работающих с персональными данными. Ориентированы на защиту персональных данных и управление рисками, связанными с их обработкой. Включают в себя практические советы по оценке и снижению рисков. Поддерживают соблюдение законодательства в области защиты данных. [33].

Методологии управления рисками информационной безопасности и типы рисков, которые они регулируют представлены в таблице 3.

Таблица 3 – Методологии и объекты регулирования

Методология	Объекты регулирования
ГОСТ Р 50922-2006	Риски, связанные с информационными системами и процессами обработки данных.
ФЗ-152 "О персональных данных"	Риски, связанные с обработкой и защитой персональных данных.
Система управления информационной безопасностью по требованиям ISO/IEC 27001	Общие риски информационной безопасности, включая физические, технические и организационные риски.
Рекомендации Роскомнадзора	Риски, связанные с обработкой персональных данных и соблюдением законодательства.
Методология оценки рисков ФСТЭК	Риски, связанные с угрозами, уязвимостями и последствиями для информационных систем.

Помимо этого, для грамотного формирования требований к средствам управления рисками информационной безопасности, очевидно следует использовать модели и стандарты более приближенные к индустрии информационных технологий:

CMMI (Capability Maturity Model Integration) Модель зрелости, используемая для оценки процессов в сфере управления проектами и разработки программного обеспечения, но также адаптированная для информационной безопасности [3];

NIST SP 800-53 Стандарты и рекомендации от Национального института стандартов и технологий США по управлению рисками в области информационной безопасности;

ISO/IEC 27001 Международный стандарт для систем управления безопасностью информации (ISMS), который включает элементы оценки зрелости [5];

COBIT (Control Objectives for Information and Related Technologies) Фреймворк для управления и управления ИТ и информационной безопасностью [27].

Уровни зрелости информационной безопасности CMMI (Capability Maturity Model Integration) [3] (см. таблицу 4):

уровень 1 – Начальный (Initial):

- процессы не документированы и хаотичны;
- отсутствует системный подход к управлению безопасностью информации;
- в большинстве случаев защитные меры принимаются на основе интуитивных решений;

уровень 2 – Повторяемый (Repeatable):

- основные процессы задокументированы и могут быть повторены;
- есть некоторые стандарты и процедуры, но они могут быть нерегулярными;
- реакция на инциденты осуществляется, но в основном реактивно;

уровень 3 – Определенный (Defined):

- процессы четко определены и задокументированы;
- стратегии и политики информационной безопасности формализованы;
- обучение сотрудников и регулярная оценка рисков внедрены.

Таблица 4 – Сопоставление требований регулятора и зрелости процессов ИБ

Требования ФСТЕК	Уровень 1 - Начальный	Уровень 2 - Повторяемый	Уровень 3 - Определенный	Уровень 4 - Управляемый	Уровень 5 - Оптимизированный
Наличие политики информационной безопасности	Политика не задокументирована	Политика написана, но не применяется	Политика документации существует, применяется	Политика активно внедряется и пересматривается	Политика регулярно обновляется на основе анализа угроз
Оценка рисков	Отсутствует оценка рисков	Проводится спорадическая оценка рисков	Регулярная оценка рисков, методология установлена	Оценка рисков интегрирована в процессы управления	Анализ рисков на уровне организации с учетом изменений
Защита периметра сети	Минимальная защита, межсетевые экраны отсутствуют	Установлены базовые межсетевые экраны	Используются межсетевые экраны следующего поколения	Реализованы сложные механизмы защиты периметра	Внедрение адаптивной безопасности с использованием ИИ
Обучение сотрудников	Нет программ обучения	Проведение разовых информационных сессий	Регулярные тренинги по безопасности	Программа обучения интегрирована в процесс адаптации новых сотрудников	Постоянное обучение с учетом последних угроз и технологий
Мониторинг и реагирование на инциденты	Отсутствует мониторинг	Мониторинг реализуется, но без анализа	Активный мониторинг и реагирование на инциденты	Процесс управления инцидентами стандартизирован и документирован	Инциденты анализируются для выявления тенденций и улучшения процессов
Учет уязвимостей	Нет системы учета уязвимостей	Периодические сканирования на уязвимости	Регулярный аудит уязвимостей	Система управления уязвимостями оптимизирована и автоматизирована	Адаптивная система реагирования на новые уязвимости

уровень 4 – Управляемый (Managed):

- процессы управляются и контролируются;
- используются метрики для оценки эффективности процедур безопасности;
- существуют системы мониторинга и реагирования на инциденты;

уровень 5 – Оптимизированный (Optimizing):

- процессы постоянно оптимизируются на основе анализа данных и лучших практик;
- инновации в области безопасности интегрированы в управление;
- организация адаптируется к изменениям угроз и технологий, улучшая свою безопасность.

Требования ФСТЭК [35] охватывают различные аспекты, включая оценку рисков, управление инцидентами и реализацию защитных мер. Для лучшего представления поставленных перед проектом задач, регуляторные требования можно совместить с базой модели зрелости.

Сформированное понимание предметной области даёт представление о роли и месте системы в ИТ-инфраструктуре организации. Собранная информация позволяет перейти к этапу определения функциональных требований проекта.

1.2 Идентификация функций системы

Таким образом можно сформировать основные функции СУРИБ. Они должны включать процессы, которые обеспечивают выявление, оценку, минимизацию и контроль рисков, образовывая основу для построения системы защиты информационных активов [16]. Определим каждую из них конкретнее:

- идентификация активов (создание перечня активов);
- оценка угроз (использование системы для анализа угроз);
- анализ рисков (применение методик оценки рисков);

- управление мерами контроля (классификация средств защиты (брандмауэры, антивирусы, и т. п.));
- мониторинг и отчетность (использование внешних систем для сбора данных и формирования отчетов).

Бизнес-цели и требования к проекту внедрения системы управления рисками информационной безопасности станут основой для формирования плана с учетом этапов, контрольных точек, сроков и ресурсов:

- повышение уровня защищенности критически важных информационных активов организации;
- снижение финансовых и репутационных рисков, связанных с инцидентами информационной безопасности;
- обеспечение соответствия требованиям законодательства и отраслевых стандартов в области информационной безопасности;
- улучшение процессов принятия решений по управлению рисками на основе качественной и количественной оценки;
- повышение осведомленности и вовлеченности сотрудников в мероприятия по обеспечению информационной безопасности.

Требования к ИТ-проекту:

а) разработать и внедрить комплексную систему управления рисками информационной безопасности, включающую следующую функциональность:
   1) идентификация и классификация информационных активов;
   2) выявление и оценка уязвимостей;
   3) анализ и моделирование угроз;
   4) количественная оценка рисков;
   5) разработка и внедрение мер защиты;
   6) мониторинг эффективности системы управления рисками;

б) интегрировать систему управления рисками с существующими ИТ-системами и бизнес-процессами организации;

в) реализовать механизмы автоматизированного выявления, анализа и реагирования на инциденты информационной безопасности;
г) соответствовать требованиям по информационной безопасности, защите персональных данных и другим нормативным документам;
д) внедрить систему управления доступом к информации и разграничения полномочий пользователей.

1.2.1 Расчет экономической эффективности проекта

Методология оценки экономической эффективности

Ключевые показатели эффективности

Формула совокупной стоимости владения (ТСО)

$$\text{ТСО} = \text{Капитальные затраты} + \text{Операционные затраты} + \text{Затраты на поддержку}$$

Финансовый анализ проекта

Инвестиционные затраты

Структура капитальных вложений:

- Программное обеспечение
- Аппаратное обеспечение
- Консультационные услуги
- Обучение персонала

Формула расчета капитальных затрат:

$$\text{КЗ} = \sum(\text{Ci} \times \text{Qi}) \qquad (1)$$

Ci - стоимость i-й позиции

Qi - количество позиций

Операционные затраты

Статьи операционных расходов:

- Лицензирование

- Техническая поддержка
- Обновление систем
- Мониторинг и аудит

Формула годовых операционных затрат:

$$ОЗ = \sum(Oi \times Ti) \quad (2)$$

Oi - стоимость операционной статьи

Ti - период действия

Оценка экономического эффекта:

Формула чистого дисконтированного дохода (NPV)

$$NPV = \sum[ЧДП/(1 + r)^t] - IC \quad (3)$$

ЧДП - чистый денежный поток

r - ставка дисконтирования

t - период времени

IC - initial investment

Индекс рентабельности инвестиций (ROI)

$$ROI = (Прибыльотинвестиций - Стоимостьинвестиций) /Стоимостьинвестиций \times 100\%$$

Модель расчета предотвращенного ущерба:

$$ПУ = \sum[Вр \times (УбезСИБ - Убс)] \quad (4)$$

ПУ - предотвращенный ущерб

Вр - вероятность реализации риска

УбезСИБ - ущерб без системы ИБ

Убс - ущерб с внедренной системой

Экономические расчеты (см. таблицу 5):

а) исходные данные:

1) инвестиции: 5 000 000 руб.;

2) годовые операционные затраты: 1 500 000 руб.;

3) предотвращенный ущерб: 12 000 000 руб.;

б) расчет показателей:

1) чистая приведенная стоимость (NPV):

$$\text{NPV} = 12\,000\,000 - 5\,000\,000 - 1\,500\,000 = 5\,500\,000 \text{ руб.};$$

2) рентабельность инвестиций (ROI):

$$\text{ROI} = (12\,000\,000 - 6\,500\,000) / 6\,500\,000 \times 100\% = 84.6\%.$$

Таблица 5 – Экономические показатели проекта

Показатель	Значение
Инвестиции	5 000 000 руб.
Годовые операционные затраты	1 500 000 руб.
Предотвращенный ущерб	12 000 000 руб.
Чистая приведенная стоимость (NPV)	5 500 000 руб.
Рентабельность инвестиций (ROI)	84.6%

Качественные преимущества проекта:

- снижение рисков информационной безопасности;
- соответствие регуляторным требованиям;
- повышение репутации компании;
- оптимизация бизнес-процессов.

Проект демонстрирует высокую экономическую эффективность с положительной чистой приведенной стоимостью.

Рентабельность инвестиций более 80% со сроком окупаемости до 2 лет

1.3 Диаграммы потоков данных

Data Flow Diagram (DFD) – это инструмент для визуализации потоков данных между функциями системы. Они позволяют наглядно отобразить, как данные перемещаются между различными компонентами системы, какие

Контекстная диаграмма (уровень 0), изображенная на рисунке 1, показывает систему в целом и ее взаимодействие с внешними сущностями (например, пользователи, регуляторы).

Диаграмма верхнего уровня (уровень 1): детализирует основные функции системы и потоки данных между ними.

Детализированные диаграммы (уровень 2 и ниже): описывают внутренние процессы каждой функции [37].

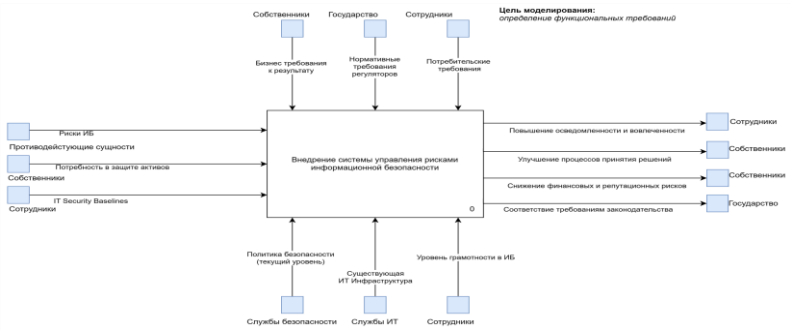

Рисунок 1 – Контекстная диаграмма (уровень 0)

Контекстная диаграмма моделируемой системы отражает взаимосвязь между заинтересованными сторонами, требованиями к системе и результатами её функционирования.

Основными заинтересованными сторонами в системе выступают собственники организации, государственные регулирующие органы и сотрудники. Каждая из указанных групп определяет специфические требования к функционированию системы: собственники формируют бизнес-требования и обеспечивают ресурсное сопровождение, государственные

органы устанавливают нормативные требования, а сотрудники формируют потребительские требования к системе информационной безопасности.

Функциональные требования к системе определяются совокупностью бизнес-требований, нормативных требований регуляторов и потребительских требований. Уделено внимание к базовым уровням безопасности (на схеме IT Security Baselines), которые непосредственно связаны с деятельностью сотрудников организации и формируют основу для разработки политики информационной безопасности.

В процессе функционирования системы ключевую роль играют службы безопасности и ИТ-службы, которые совместно с сотрудниками обеспечивают реализацию защитных механизмов. Их взаимодействие направлено на достижение результатов повышения уровня осведомленности и вовлеченности персонала, совершенствования процессов принятия решений в области информационной безопасности, минимизацию финансовых и репутационных рисков, а также обеспечение соответствия требованиям действующего законодательства.

Система управления рисками информационной безопасности функционирует как целостный механизм, где каждый компонент влияет на эффективность других элементов. Идентификация рисков и противодействующих сущностей, оценка текущего уровня безопасности, разработка и внедрение защитных механизмов, а также мониторинг эффективности системы образуют замкнутый цикл управления, обеспечивающий непрерывное совершенствование процессов информационной безопасности организации.

Диаграмма верхнего уровня (уровень 1), изображенная на рисунке 2, представляет собой концептуальную модель системы управления рисками информационной безопасности, демонстрирующую основные функциональные компоненты и потоки данных между ними. Данная модель позволяет наглядно отобразить архитектуру системы и взаимосвязи между её ключевыми элементами [37].

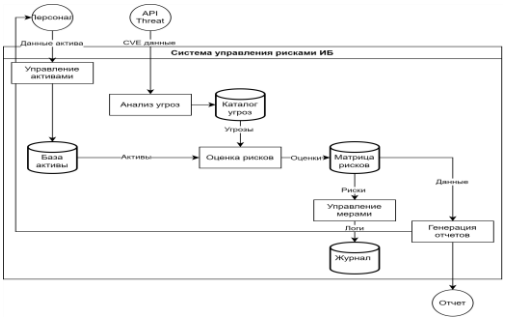

Рисунок 2 – Концептуальная диаграмма

Механизм взаимодействия функциональных компонентов реализуется через систему потоков данных. Информация об активах последовательно передается из модуля управления активами в систему оценки рисков. Данные об угрозах извлекаются из каталога угроз, проходят анализ и также направляются в модуль оценки рисков. Результаты оценки рисков используются для формирования отчетности и разработки мер по снижению рисков. При этом все операции и изменения в системе фиксируются в модуле логирования и журнале событий, что обеспечивает необходимую прозрачность и возможность последующего аудита [18].

Представленная диаграмма демонстрирует замкнутый цикл управления рисками, где каждый функциональный компонент вносит свой вклад в общую эффективность системы, а потоки данных обеспечивают необходимую интеграцию и координацию между различными элементами системы.

Детализированная диаграмма (уровень 2), изображенная на рисунке 3, представляют собой декомпозицию основных функциональных компонентов системы управления рисками информационной безопасности (СУР ИБ) и описывают внутренние процессы каждой функции.

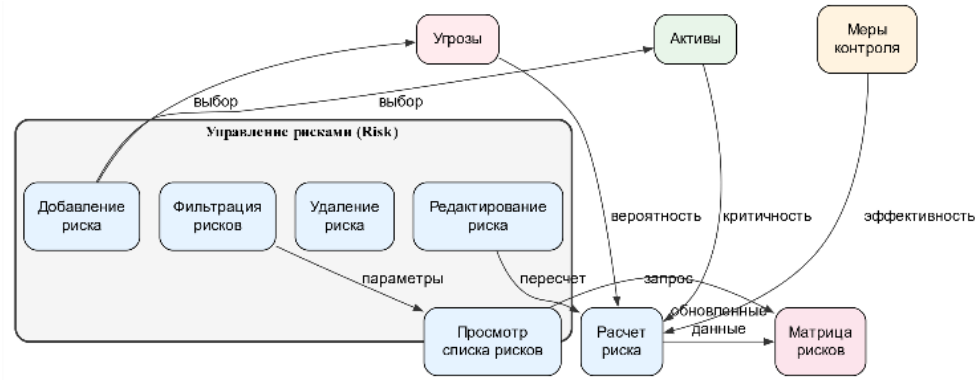

Рисунок 3 – Декомпозиция функциональных компонентов

Рассмотрим детальное описание функций модуля управления рисками (Risk), являющегося ядром системы:

Расчет риска осуществляет автоматический расчет по установленной формуле, которая будет разработана на следующем этапе.

Просмотр списка рисков предоставляет пользователю таблицу с маркировкой уровня риска. Результат работы функции зависит от корректной работы всех остальных функций системы.

Добавление риска позволяет создать связь «Актив-Угроза» с необходимыми параметрами для расчета, требуя предварительного выбора конкретного актива и угрозы из соответствующих каталогов [12].

Редактирование риска дает возможность корректировать параметры (вероятность, потенциальный ущерб и другие характеристики), при этом система автоматически пересчитывает уровень риска на основе внесенных изменений.

Удаление риска обеспечивает удаление оценки риска из системы, при этом не затрагивая данные об активах и угрозах, что обеспечивает целостность информационной базы.

Фильтрация рисков предоставляет функционал поиска по различным критериям (активы, уровень риска, статус), работая непосредственно с данными таблицы рисков.

Аналогичным образом производится детализация остальных функциональных модулей системы, включая управление активами, анализ угроз, генерацию отчетов, управление мерами и ведение логов и журнала событий. Каждый модуль имеет собственную структуру внутренних процессов, которая отражает специфику его работы и взаимосвязи с другими компонентами системы.

1.4 Диаграммы вариантов использования

Диаграммы вариантов использования (Use Case Diagrams UML) помогают описать взаимодействие пользователей (акторов) с системой и выделить основные сценарии работы.

Преимущества использования диаграмм вариантов использования:

- предоставление высокоуровневого представления о системе;
- улучшение коммуникации между заинтересованными сторонами;
- возможность имитации реальных сценариев взаимодействия;
- более полная интерпретация системы по сравнению с другими видами диаграмм;
- простота восприятия даже для нетехнических участников проекта.

Основные элементы, изображенные на рисунке 4:

1) акторы: пользователи или внешние системы, взаимодействующие с системой (например, «Администратор», «Аналитик»);
2) варианты использования: конкретные задачи или действия, которые выполняет актор (например, «Создание отчета», «Добавление данных»);
3) связи: отношения между акторами и вариантами использования (ассоциации, включения, расширения).

Use-Case сценарии:

Рисунок 4 – Диаграмма сценария администрирования системы

Управление ролями и правами предоставляет администраторам инструменты для создания, модификации и удаления пользовательских ролей, настройки разрешений доступа к модулям системы и конфигурации политик безопасности. При этом осуществляется проверка текущих прав доступа и возможна интеграция с системой уведомлений.

Мониторинг логов позволяет администраторам отслеживать действия пользователей, включая логи входа, изменения данных и попытки доступа. Функционал включает фильтрацию по различным параметрам (дата, пользователь, тип события) и может быть интегрирован с системой резервного копирования для архивации логов.

Управление уведомлениями обеспечивает настройку оповещений о критических событиях (попытки взлома, изменение важных настроек) и определение каналов доставки (email, Telegram, SMS). Активация уведомлений возможна при изменении ролей или обнаружении подозрительной активности в логах.

Резервное копирование предоставляет возможности для автоматического или ручного создания копий данных системы (настройки, логи, пользовательские данные) и их последующего восстановления. Процесс может инициироваться по расписанию или вручную, при этом все операции фиксируются в системе логов.

Взаимосвязи между функциональными возможностями реализуются следующим образом. Настройка ролей приводит к записи соответствующих событий в логах. Просмотр логов может инициировать процесс резервного копирования. Управление уведомлениями позволяет оперативно реагировать на события, зафиксированные в логах. Диаграммы сценариев изображены на рисунках 5 и 6.

Рисунок 5 – Диаграмма сценария оценка и анализ

Рисунок 6 – Диаграмма сценария формирования отчетности

СУРИБ включает несколько ключевых ролей: администратор, аналитик и руководитель. Процесс начинается с добавления активов в систему администратором. На основе этих данных аналитик проводит оценку

потенциальных угроз. Затем он использует полученные данные для анализа рисков, предлагая меры контроля для их минимизации. После внедрения мер контроля формируется отчет, который включает результаты анализа и рекомендации. Этот отчет передается руководителю для принятия решений. Схемы сценариев изображены на рисунках 7 и 8.

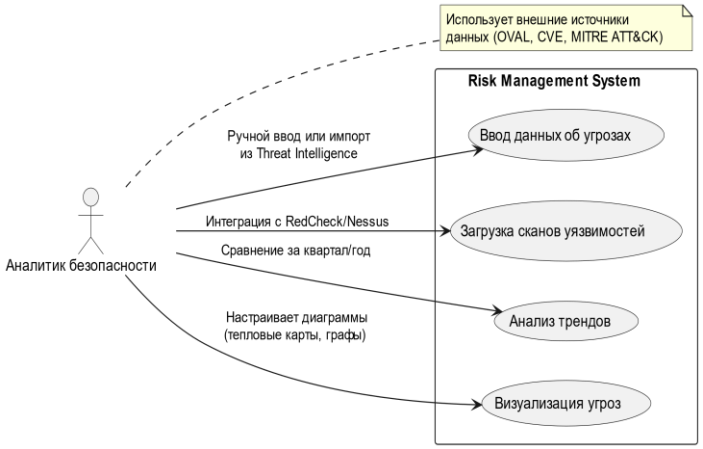

Рисунок 7 – Сценарий работы с системой аналитика безопасности

Аналитик безопасности обеспечивает актуальность данных об угрозах и уязвимостях. Основная деятельность сосредоточена на интеграции внешних источников информации, техническом анализе угроз и поддержании базы знаний системы в актуальном состоянии/

Рисунок 8 – Диаграмма взаимосвязи пользовательских сценариев

1.5 Результаты функционального моделирования

В результате работы, проделанной на этапе функционального моделирования получены следующие результаты:

Описание функций системы:

1) управление активами (Asset) (см. таблицу 6);

Таблица 6 – Декомпозиция управления активами

Функция	Назначение	Взаимосвязи
Просмотр списка активов	Отображение всех информационных активов системы	Связан с рисками и угрозами
Добавление актива	Создание записи об активе (сервер, БД, ПО и т.д.)	Используется при оценке рисков
Редактирование актива	Обновление параметров актива (критичность, описание)	Влияет на расчет уровня риска
Удаление актива	Удаление актива из системы (с проверкой на связанные риски)	Блокируется при наличии зависимостей

2) управление угрозами (Threat) (см. таблицу 7);

Таблица 7 – Декомпозиция управления угрозами

Функция	Назначение	Взаимосвязи
Просмотр списка угроз	Отображение всех возможных угроз (утечка данных, DDoS и т.д.)	Связан с рисками и активами
Добавление угрозы	Регистрация новой угрозы с параметрами (источник, уровень воздействия)	Используется при оценке рисков
Редактирование угрозы	Корректировка описания угрозы	Влияет на связанные риски
Удаление угрозы	Удаление угрозы из системы	Блокируется при наличии зависимостей

3) анализ рисков (Risk) – ядро системы (см. таблицу 8);

Таблица 8 – Декомпозиция анализа рисков

Функция	Назначение	Взаимосвязи
Расчет риска	Автоматический расчет по формуле	Использует данные активов, угроз и мер контроля
Просмотр списка рисков	Таблица с маркировкой уровня риска (низкий/средний/высокий)	Зависит от всех модулей
Добавление риска	Создание связи "Актив-Угроза" с параметрами для расчета	Требует выбор актива и угрозы
Редактирование риска	Корректировка параметров (вероятность, ущерб и т.д.)	Автоматически пересчитывает уровень
Удаление риска	Удаление оценки риска	Не затрагивает активы/угрозы
Фильтрация рисков	Поиск по активам, уровню риска или статусу	Работает с данными таблицы

4) управление мерами контроля (ControlMeasure) (см. таблицу 9);

Таблица 9 – Управление мерами контроля

Функция	Назначение	Взаимосвязи
Добавление меры	Регистрация меры защиты (фаервол, шифрование и т.д.)	Связывается с конкретным риском
Оценка эффективности	Задание эффективности меры (0-1)	Влияет на знаменатель формулы риска
Просмотр мер	Отображение всех мер с их статусом (активна/неактивна)	Фильтрация по рискам

Отчет по эффективности	Анализ снижения рисков после применения мер	Использует данные расчета рисков

5) отчетность (Report) (см. таблицу 10);

Таблица 10 – Функциональность отчетов

Функция	Назначение	Взаимосвязи
Генерация PDF-отчета	Экспорт списка рисков с визуализацией	Использует данные всех модулей
Диаграмма распределения	Круговая/столбчатая диаграмма по уровням риска	Анализирует RiskLevel
Исторический анализ	График изменения рисков за выбранный период	Сравнивает расчеты за разные даты

вспомогательные функции (см. таблицу 11);

Таблица 11 – Сопутствующие требования к системе

Функция	Назначение	Взаимосвязи
Аутентификация	Контроль доступа к системе	Обязательна для всех функций
Валидация данных	Проверка вводимых значений (например, вероятность 0-1)	Защита от некорректных расчетов
Уведомления	Оповещения о высокорисковых ситуациях	Анализирует RiskLevel
Резервное копирование	Экспорт/импорт данных системы	Работает со всеми таблицами БД

Схема взаимосвязей ключевых функций изображена на рисунке 9.

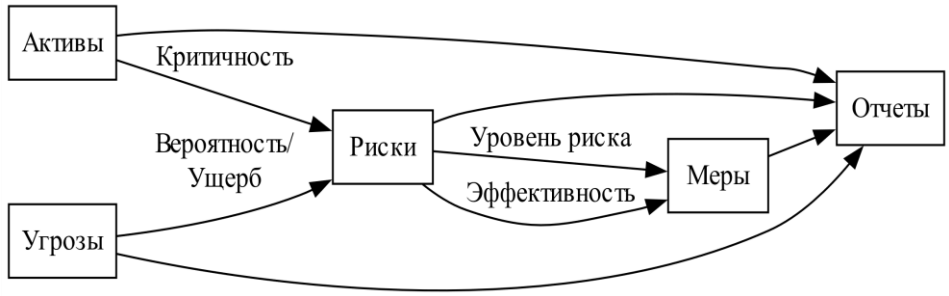

Рисунок 9 – Схема взаимосвязей ключевых функций

Критические зависимости:

Риски → Активы/Угрозы: без активов и угроз невозможно создать оценку риска.

Меры контроля → Риски: Эффективность мер влияет на пересчет уровня риска.

Отчеты → Все модули: формируются на основе комплексных данных.

Глава 2 Логическое проектирование

2.1 Цели и задачи логического проектирования

Логическое проектирование является следующим этапом после функционального моделирования и направлено на детализацию архитектуры системы, а также описание логики её работы. На этом этапе определяются основные компоненты системы, их взаимодействие и принципы функционирования. Логическое проектирование это переход от абстрактного описания функций к конкретной структуре, которая будет реализована в системе [30]. В рамках данной главы рассматривается архитектура системы и базы данных.

Важные аспекты логического проектирования:

- соответствие функциональным требованиям; логическая модель должна точно отражать функциональные требования к системе, обеспечивая выполнение всех необходимых операций и процессов;
- гибкость и масштабируемость; проектирование должно учитывать возможность расширения и модификации системы в будущем, что требует тщательного планирования архитектуры и интерфейсов;
- эффективность и оптимизация; важно стремиться к оптимальной производительности системы, минимизируя избыточные операции и улучшая алгоритмы обработки данных;
- документация; на этапе логического проектирования необходимо вести детальную документацию, описывающую структуру, компоненты и логику работы системы, что облегчит последующую реализацию и поддержку системы.

Логическое проектирование является основой для последующего физического проектирования и реализации системы, обеспечивая чёткое понимание структуры и функционирования системы управления рисками.

Цели:

1) формализация требований. Преобразование бизнес-требований в логическую модель системы, которая отражает функциональность и структуру данных;
2) организация данных и процессов. Определение способов хранения, обработки и управления данными в соответствии с потребностями пользователей;
3) обеспечение гибкости и масштабируемости. Создание модели, которая может быть адаптирована под изменения требований и расширение функционала;
4) подготовка к физическому проектированию. Разработка базовой архитектуры системы, которая послужит основой для выбора технологий и инструментов.

Задачи:

- определение бизнес-процессов и их взаимосвязей;
- разработка концептуальной модели данных (ER-диаграммы, классы объектов);
- проектирование интерфейсов взаимодействия между компонентами системы;
- учет требований к безопасности, производительности и удобству использования;
- выбор архитектурных шаблонов для последующих этапов разработки.

При разработке логической модели важно помнить, что она должна представлять собой компромисс между потребностями конкретного проекта и основными бизнес-запросами. Рекомендуется параллельно разрабатывать модели функций и данных, так как это позволяет:

- лучше понять требования к данным;
- выявить дополнительные функциональные требования;
- обеспечить целостность системы.

Результатом этого процесса должна стать четкая, понятная и недвусмысленная модель области деловой активности, которая может быть

использована для дальнейшей разработки физических моделей данных и бизнес-логики.

2.2. Архитектура системы управления рисками ИБ

Выбор математической модели для расчета рисков представляет собой основную задачу логического проектирования, поскольку именно формула расчета риска является ключевой бизнес-логикой всего приложения. От корректности выбранной математической модели зависит не только точность оценки рисков, но и эффективность принимаемых управленческих решений, что напрямую влияет на безопасность информационных активов организации. В современных условиях, когда информационные системы становятся все более сложными, а угрозы – изощренными, критически важно разработать такой алгоритм расчета, который бы учитывал все существенные факторы риска и при этом оставался достаточно гибким для адаптации к меняющимся условиям. Именно поэтому процесс выбора и обоснования математической модели требует особого внимания и тщательного анализа существующих подходов к оценке рисков.

1) базовая формула риска [1]

Риск = Вероятность × Ущерб

Вероятность — оценка от 0 до 1 (или 0-100%).

Ущерб — финансовые потери или уровень критичности (например: 1–5).

Пример кода C#:

```
public double CalculateRisk(double probability, int impactLevel)
{
   return probability * impactLevel;
}
```

Базовая формула риска изображена на рисунке 10.

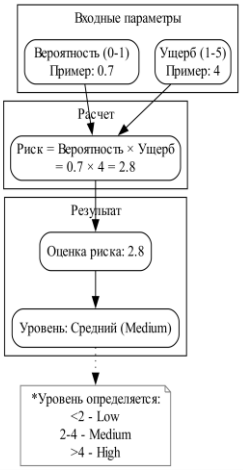

Рисунок 10 – Базовая формула риска

2) формула с учетом мер контроля [1]

Риск = (Вероятность × Ущерб) × (1 - Эффективность_контроля)

Эффективность_контроля — от 0 (нет защиты) до 1 (полная защита).

Пример кода (C#):

```
public double CalculateRiskWithControls(double probability, int impact, double controlEfficiency)
{
   return (probability * impact) * (1 - controlEfficiency);
}
```

Схема формулы с мерами контроля изображена на рисунке 11.

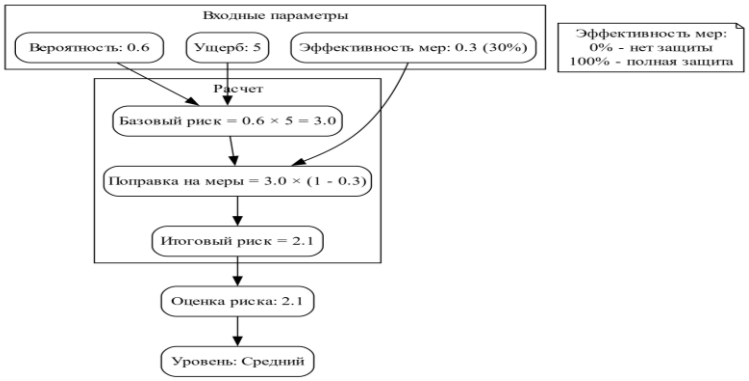

Рисунок 11 – Формула с мерами контроля

3) если угрозы импортируются из OVAL/CVE, используйте CVSS Score (Common Vulnerability Scoring System)[22]:

Risk = CVSS_Base_Score × Asset_Criticality

CVSS_Base_Score — из метаданных CVE (например, 5.4).

Asset_Criticality — важность актива (1–3).

Пример:

```
//C#
public double CalculateCvssRisk(double cvssScore, int assetCriticality)
{
   return cvssScore * assetCriticality;
}
```

Схема расчета на основе CVSS изображена на рисунке 12.

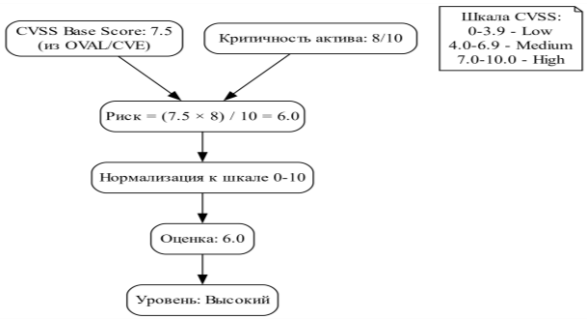

Рисунок 12 – Расчет на основе CVSS

4) комбинированная формула (для комплексных сценариев) [36]

Риск = (Вероятность × Ущерб × Важность_актива) / (Эффективность_контроля + 1)

```
//C#:
private double CalculateRiskLevel()
{
      return (Probability * Impact * AssetCriticality) / (ControlEfficiency + 1);
}
```

Схема расчета для комбинированных задач изображена на рисунке 13.

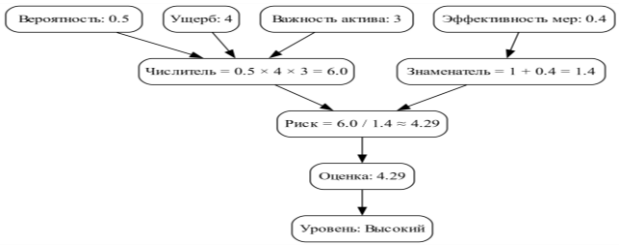

Рисунок 13 – Расчет для комбинированных задач

В таблице 12 представлено сравнение алгоритмов расчета риска.

Таблица 12 – Сравнение алгоритмов расчета риска

Формула	Описание	Преимущества	Недостатки
Базовая формула риска	Риск=Вероятность×Ущерб	Простота расчета, универсальность	Не учитывает меры контроля и важность актива
Формула с учетом мер контроля	Риск=(Вероятность×Ущерб) ×(1−Эффективность_контроля)	Учитывает эффективность мер контроля	Не включает важность актива
Расчет на основе CVSS	Риск рассчитывается на основе вектора уязвимостей	Подходит для оценки уязвимостей в информационных системах	Ограничен сферой применения
Комбинированная формула	Риск = (Вероятность × Ущерб × Важность_актива) / (Эффективность_контроля + 1)	Комплексный учет всех факторов, универсальность, точность	Более сложная для расчета

Процесс выбора и обоснования математической модели требует особого внимания и тщательного анализа существующих подходов к оценке рисков, поэтому комбинированная формула, представляется оптимальным выбором. Выбор обоснован её универсальностью, точностью и возможностью адаптации под конкретные задачи:

- учет всех ключевых факторов риска. Формула включает такие параметры, как вероятность наступления события, возможный ущерб,

важность актива и эффективность контроля. Это позволяет получить комплексную оценку риска, учитывающую как количественные, так и качественные аспекты;

- гибкость применения. Формула может быть адаптирована для различных сценариев и отраслей, что делает её универсальной и применимой в широком спектре задач;
- простота интерпретации. Результат расчета риска выражается в числовом значении, которое легко интерпретировать и использовать для принятия решений. Например, высокий уровень риска (как показано на изображении) указывает на необходимость срочных мер по снижению рисков;
- практическое применение. Формула позволяет не только оценить текущий уровень риска, но и определить, какие меры контроля наиболее эффективны для его снижения. Это особенно важно для разработки стратегий управления рисками;
- соответствие стандартам. Данная формула соответствует принципам международных стандартов управления рисками, таких как ISO 31000[36], что повышает её научную и практическую значимость.

Схема оценки рисков в контексте всей системы изображена на рисунке 14.

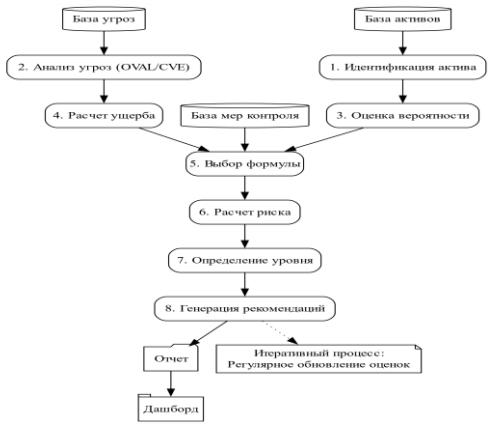

Рисунок 14 – Оценка рисков в контексте всей системы

Оценка рисков с выбором различных вариантов расчета может быть реализована в последующих версиях системы при соответствующих потребностях.

2.3. Описание процессов обработки данных системы

В процессе проектирования базы данных для системы управления рисками информационной безопасности был проведен тщательный анализ функциональных требований и определены основные сущности системы. Проектирование велось с использованием методологии нормализации, что обеспечило эффективное хранение данных и минимизацию избыточности. В результате нормализации до НФ3, были определены основные таблицы: Assets для хранения информации об активах организации, Threats для учета потенциальных угроз, Risks для оценки рисков, ControlMeasures для хранения информации о мерах контроля и Reports для формирования отчетности. В таблице 2 представлены базы данных и файлы хранилища данных.

Таблица 13 – Хранилища данных (Базы данных и файлы)

Название	Описание
Assets	База данных активов (название, тип, критичность)
Threats	Каталог угроз (описание, источник, уровень воздействия)
Risks	Расчетные риски (вероятность, воздействие, уровень)
Measures	Меры контроля (статус, эффективность)
Reports	Сгенерированные отчеты

При проектировании структуры таблиц учитывались требования к масштабируемости системы и необходимости поддержки различных типов запросов. Так в таблице Assets были определены поля для идентификации актива, его типа, критичности и описания. Таблица Threats включает характеристики угроз, их источник и уровень воздействия. Таблица Risks содержит параметры для комплексной оценки вероятности и влияния рисков.

В процессе проектирования учитывались требования к производительности системы. Были определены первичные и внешние ключи, оптимизированы индексы для часто используемых запросов. Предусмотрена возможность расширения функциональности путем добавления новых полей и таблиц без нарушения целостности данных.

Описание диаграммы:

Актив (Asset):

«Id» – уникальный идентификатор;

- «Наименование» – название актива;
- «Тип» – категория (оборудование, ПО, данные и т.д.);
- «Критичность» – важность актива для бизнеса;
- «Описание» – дополнительные сведения;

Угроза (Threat):

- «Id» – уникальный идентификатор;
- «Название» – имя угрозы (например, «Кибератака»);
- «Описание» – характеристики угрозы;
- «Источник2 – происхождение (внутреннее/внешнее);
- «УровеньВлияния2 – потенциальная тяжесть последствий;

Риск (Risk):

- «Id» – уникальный идентификатор;
- «Связи» – assetid, threatid;
- «Вероятность» – шанс возникновения (0-1);
- «Воздействие» – ущерб (шкала 1-5);
- «КритичностьАктива» – вес актива (1-3);
- «ЭффективностьКонтроля» – снижение риска (0-1);
- «УровеньРиска» (вычисляемый) – результирующая оценка;

Мера контроля (ControlMeasure):

- «Id» – уникальный идентификатор;
- «RiskId» – связанный риск;
- «МераКонтроля» – название меры;

— «Описание» – как реализуется;

— «Статус» – текущее состояние (активна/неактивна);

Отчет (Report):

— «Id» – уникальный идентификатор;

— «ДатаОтчета» – время формирования;

— «Содержание» – аналитические данные;

— «Статус» – состояние (черновик/опубликован).

Особое внимание уделялось проектированию связей между таблицами. Была реализована иерархическая структура связей, где один актив может быть связан с множеством рисков, что отражает реальную ситуацию в организации. Аналогичным образом одна угроза может порождать несколько рисков. Для каждого риска может быть определено множество мер контроля, что обеспечивает гибкость в управлении рисками. Отчеты формируются на основе информации о рисках, что позволяет получать комплексную аналитику.

ER диаграмма изображена на рисунке 15.

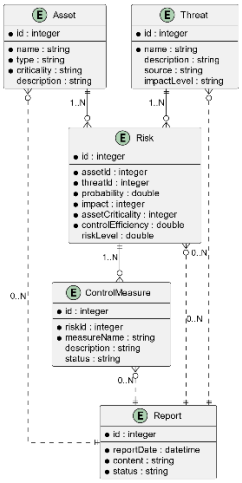

Рисунок 15 – ER диаграмма

Жёсткие связи (сплошные линии):

- 1 Актив → 0.. Рисков*"Связан с" – Каждый актив может иметь несколько рисков;

- 1 Угроза → 0.. Рисков*"Связан с" - Одна угроза может проявляться в разных рисках;
- 1 Риск → 0.. Мер контроля*"Имеет" – Для каждого риска предусмотрены меры снижения.

Связи с отчетами (пунктирные линии):

- все классы могут быть "включены в" 0..* Отчетов;
- отчеты агрегируют данные из других сущностей.

Принципы проектирования включают четкое разделение ответственности:

- активы и угрозы описывают объекты системы;
- риски оценивают их взаимодействие;
- меры контроля обеспечивают снижение рисков.

Гибкость отчетности:

- отчеты могут включать любые комбинации данных;
- связи с отчетами сделаны слабыми (агрегация).

Автоматизация расчетов:

- уровень риска вычисляется динамически на основе актуальных параметров.

Пример взаимодействия:

1) пользователь добавляет актив через форму на сайте;
2) данные сохраняются в таблицу Assets через Entity Framework;
3) модуль оценки угроз анализирует актив и предлагает возможные угрозы;
4) модуль анализа рисков рассчитывает уровень риска и сохраняет его в таблицу Risks;
5) модуль управления мерами контроля предлагает меры для снижения риска;
6) модуль отчетов формирует отчет и отображает его пользователю.

Диаграмма отражает полный цикл управления рисками: от активов и угроз через оценку рисков к мерам контроля и отчетности.

Диаграмма классов изображена на рисунке 16.

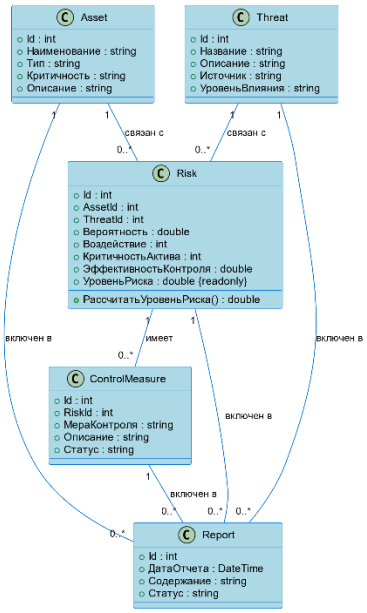

Рисунок 16 –

Разработанная структура базы данных обеспечивает надежное хранение информации о всех аспектах управления рисками информационной безопасности, поддерживает необходимые аналитические операции и позволяет масштабировать систему в соответствии с растущими потребностями организации.

2.4. Результаты логического проектирования

Основные элементы логического проектирования:

а) типы связей:

1) композиция (жёсткая): актив/угроза → риск → мера контроля;
2) агрегация (гибкая): все сущности → отчеты;

б) ключевые атрибуты:

1) для Риска: УровеньРиска (расчётный показатель);
2) для Отчёта: Статус (жизненный цикл документа);

в) бизнес-правила:

1) уровень риска пересчитывается при изменении:

- вероятности угрозы;
- критичности актива;
- эффективности мер контроля.

В таблице 14 представлены ключевые сущности данных.

Таблица 14 – Ключевые сущности данных

Сущность	Описание	Связанные компоненты
Актив (Asset)	Ресурсы организации, требующие защиты (оборудование, ПО, данные). Характеризуются важностью и типом.	Риски, Отчеты
Угроза (Threat)	Потенциальные источники опасности для активов (кибератаки, сбои, человеческий фактор).	Риски, Отчеты
Риск (Risk)	Комбинация угрозы и актива с оценкой вероятности, воздействия и уровня риска. Вычисляется автоматически.	Актив, Угроза, Меры контроля, Отчеты
Мера контроля (ControlMeasure)	Действия для снижения риска (технические, организационные). Имеют статус реализации.	Риск, Отчеты
Отчет (Report)	Документ с аналитикой по рискам, мерам и активам. Формируется на основе данных системы.	Все сущности (агрегирует данные)

Особенности реализации:

CRUD для управления данными:

CRUD (Create, Read, Update, Delete) — это набор основных операций для управления данными в информационных системах:

- «Create» (Создание) – добавление новых записей в базу данных;
- «Read» (Чтение) – извлечение и просмотр данных;
- «Update» (Обновление) – изменение существующих записей;
- «Delete» (Удаление) – удаление данных из системы.

Архитектурные шаблоны проектирования: Слоистая архитектура (Layered Architecture)

Применение:

- презентационный слой (API/UI) – обработка HTTP-запросов (GET, POST, PUT, DELETE);
- бизнес-логика (Controller/Services) – валидация, расчет, управление инменениями данных;
- доступ к данным (Repository/EF Core) – выполнение запросов к БД.

Схема основной последовательности оценки угрозы изображена на рисунке 17.

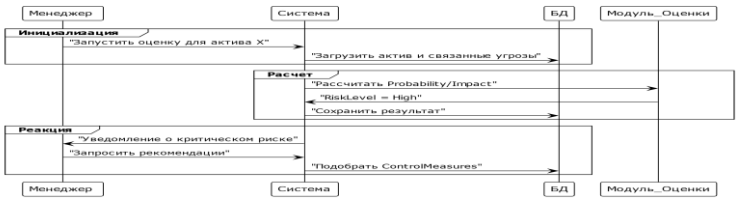

Рисунок 17 – Основная последовательность оценки угрозы

Основная последовательность оценки угрозы включает:

1. Чистая архитектура (Clean Architecture). Применение:
   - Core – бизнес-правила (расчет RiskLevel).
   - Infrastructure – реализация репозиториев (CRUD для EF Core).
2. CQRS (Command Query Responsibility Segregation). Применение:
   - Commands – операции изменения (Create, Update, Delete).
   - Queries – операции чтения (Get, List).

Поведенческие паттерны:

1) «Паттерн Стратегия»: используется для инкапсуляции различных алгоритмов расчета рисков и уровней зрелости.
2) «Паттерн Наблюдатель»: используется для реализации механизма о событиях, связанных с ИТ-активами и мерами защиты.

3) «Паттерн Команда»: используется для инкапсуляции действий, связанных с управлением ИТ-активами, мерами защиты и уровнями зрелости.

Глава 3 Физическое проектирование

3.1 Цели и задачи физического проектирования

Физическое проектирование — это этап, на котором разрабатывается конкретная реализация системы на основе логической модели. На этом этапе определяются технические детали, необходимые для развёртывания и функционирования системы в реальной среде. Физическое проектирование является завершающим этапом проектирования системы управления рисками информационной безопасности (ИБ). Основная цель физического проектирования — преобразовать логическую архитектуру и модели в работающую систему, готовую к внедрению. На этом этапе определяются конкретные технологии, инструменты и инфраструктура, которые будут использоваться для реализации системы.

Важные аспекты физического проектирования:

- соответствие техническим и бизнес-требованиям; выбранное оборудование и программное обеспечение должны соответствовать как техническим, так и бизнес-требованиям организации;
- оптимизация затрат; необходимо найти баланс между стоимостью решения и его производительностью, учитывая бюджет проекта;
- обеспечение надёжности и доступности; система должна быть спроектирована так, чтобы обеспечивать высокую надёжность и доступность услуг, минимизируя время простоев;
- гибкость и масштабируемость; физическое проектирование должно учитывать возможность будущего расширения системы и адаптации к изменяющимся условиям.

Цели:

- реализация логической модели; преобразование логической архитектуры системы в физическую структуру, которая может быть внедрена на конкретном оборудовании и программном обеспечении;

- оптимизация производительности; обеспечение эффективной работы системы за счет выбора оптимальных методов хранения данных, организации вычислений и управления ресурсами;
- обеспечение надежности и безопасности; создание механизмов защиты данных, предотвращение потери информации и обеспечение отказоустойчивости;
- снижение затрат; минимизация расходов на разработку, внедрение и эксплуатацию системы.

Задачи:

- определение физической структуры базы данных (типы файлов, индексы, методы доступа);
- выбор аппаратного обеспечения и программных средств для реализации системы;
- разработка интерфейсов взаимодействия между компонентами системы;
- проектирование механизмов резервного копирования и восстановления данных;
- учет требований к масштабируемости и поддержке системы.

3.2. Выбор технологий и инструментов

Архитектурные подходы проекта:

а) слоистая архитектура (Three-Tier Architecture). Проект следует принципам слоистой архитектуры, которая состоит из следующих уровней:
   1) презентационный уровень (UI/Web);
   2) уровень бизнес-логики (Controllers, Services);
   3) уровень доступа к данным (Repository, Database);

б) разделение ответственности (Separation of Concerns). Проект использует принципы разделения ответственности, где каждый

компонент отвечает за одну конкретную задачу. Это позволяет добиться гибкости, масштабируемости и упрощает тестирование;

в) модульность. Проект разделен на модули, отвечающие за различные функциональные возможности системы, такие как управление ИТ-активами, средствами защиты и уровнями зрелости. Это упрощает разработку, развертывание и обслуживание системы.

Диаграмма развертывания изображена на рисунке 18.

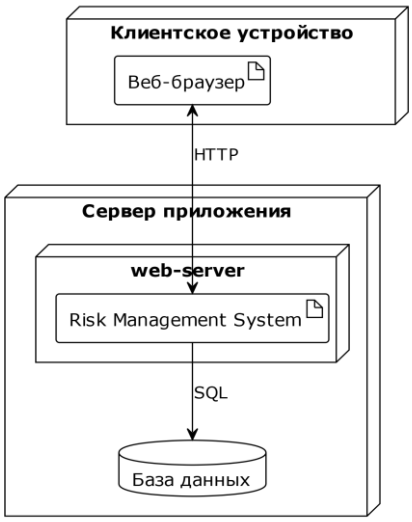

Рисунок 18 – Диаграмма развертывания

Определение технологического стека:

В качестве основной платформы разработки системы управления рисками информационной безопасности выбран язык программирования C# и фреймворк ASP.NET Core. Этот выбор обусловлен рядом преимуществ, которые делают данную технологическую связку одной из наиболее подходящих для создания современных, масштабируемых и безопасных веб-приложений.

Преимущества языка программирования C# [30]:

1) высокая производительность: C# является языком с сильной типизацией и компилируемым в промежуточный язык (IL), что обеспечивает высокую производительность приложений;
2) поддержка асинхронного программирования (async/await) позволяет эффективно работать с ресурсоемкими операциями, такими как запросы к базам данных или внешним API;
3) кроссплатформенность: благодаря .NET Core (частью которого является ASP.NET Core), приложения на C# могут работать на различных операционных системах, включая Windows, Linux и macOS;
4) безопасность: C# предоставляет встроенные механизмы для работы с криптографией, что важно для реализации функций шифрования данных и обеспечения безопасности системы. Язык поддерживает строгую типизацию и проверку на этапе компиляции, что снижает вероятность ошибок, связанных с уязвимостями;
5) широкая экосистема: C# имеет огромное количество библиотек и пакетов, доступных через NuGet, что ускоряет разработку и позволяет использовать готовые решения для типовых задач.

Преимущества фреймворка ASP.NET Core [26]:

1) модульность и масштабируемость: ASP.NET Core поддерживает модульную архитектуру, что позволяет легко добавлять или удалять функциональные блоки в зависимости от требований проекта;
2) фреймворк оптимизирован для создания распределенных систем;
3) высокая производительность: ASP.NET Core является одним из самых быстрых фреймворков для веб-разработки, что подтверждается тестами производительности, такими как TechEmpower benchmarks;
4) поддержка асинхронных операций и потоковой обработки данных позволяет эффективно работать с большими объемами информации;

5) безопасность: встроенные механизмы аутентификации и авторизации (например, Identity Framework) упрощают реализацию функций управления доступом;
6) поддержка защиты от распространенных атак, таких как CSRF (межсайтовая подделка запроса) и XSS (межсайтовый скриптинг);
7) кроссплатформенность: ASP.NET Core позволяет разрабатывать и запускать приложения на различных платформах, что обеспечивает гибкость при развертывании системы;
8) интеграция с современными технологиями: поддержка RESTful API, WebSocket, SignalR и других современных протоколов обмена данными;
9) легкая интеграция с облачными платформами, такими как Microsoft Azure, AWS и Google Cloud.

Преимущества среды разработки Visual Studio 2022[30]:

1) мощные инструменты для разработки; интегрированная среда разработки (IDE) предоставляет все необходимые инструменты для написания, отладки и тестирования кода;
2) поддержка IntelliSense, которая ускоряет написание кода за счет автодополнения и подсказок;
3) интеграция с системами контроля версий; встроенная поддержка Git и других систем контроля версий позволяет эффективно управлять исходным кодом и совместной работой над проектом;
4) инструменты для тестирования; Visual Studio 2022 предоставляет встроенные средства для модульного, интеграционного и нагрузочного тестирования, что помогает выявлять ошибки на ранних этапах разработки;
5) поддержка Docker и контейнеризации; встроенные инструменты для работы с Docker позволяют легко создавать, тестировать и развертывать контейнеры, что упрощает процесс разработки и внедрения;

6) расширяемость; Visual Studio поддерживает множество расширений, которые можно использовать для добавления дополнительных функций, таких как анализ кода, интеграция с внешними сервисами и многое другое;
7) поддержка работы с базами данных; встроенные инструменты для работы с базами данных, такие как SQL Server Data Tools (SSDT), позволяют легко проектировать, тестировать и развертывать базы данных.

Использование языка программирования C#, фреймворка ASP.NET Core и среды разработки Visual Studio 2022 обеспечивает создание высокопроизводительной, безопасной и масштабируемой системы управления рисками информационной безопасности. Эти технологии предоставляют все необходимые инструменты для реализации сложных бизнес-логик, интеграции с внешними системами и обеспечения высокого уровня защиты данных. Кроме того, кроссплатформенность и поддержка современных стандартов делают эту связку идеальным выбором для разработки корпоративных решений.

На основе логического проектирования и требований к системе выбран следующий технологический стек:

а) презентационный слой (Presentation Layer):
   1) ASP.NET MVC: Создание контроллеров и представлений для каждой функциональной области (активы, угрозы, риски, меры контроля, отчеты) пользовательского интерфейса и управления взаимодействием с пользователем;
   2) Razor Pages: генерация HTML-страниц на сервере;
   3) JavaScript (jQuery, Bootstrap): улучшение взаимодействия с пользователем и адаптивного дизайна;

б) слой бизнес-логики (Business Logic Layer):
   1) C#: реализация бизнес-логики, включая идентификацию активов, оценку рисков и управление мерами контроля;

2) LINQ: работа с данными и упрощения запросов;

в) слой доступа к данным (Data Access Layer):

1) Entity Framework Core: взаимодействие с базой данных и управление миграциями;
2) SQL Server: в качестве системы управления базами данных (СУБД).

Структурная диаграмма инструментов разработки изображена на рисунке 19.

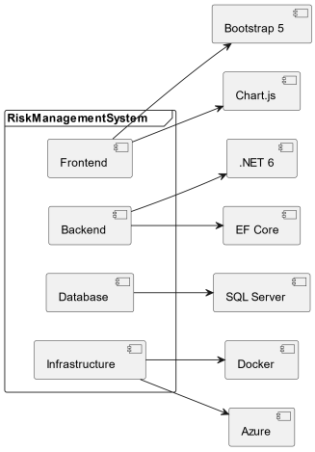

Рисунок 19 – Структурная диаграмма инструментов разработки

Для наглядности излагается архитектурная схема системы в табличном представлении – представлено в таблицах 15, 16, 17, 18.

Таблица 15 – Уровень представления (UI Layer)

Компонент	Технологии	Ответственность	Взаимодействие с
Controllers	ASP.NET Core MVC	Обработка HTTP-запросов	Services → Views
Razor Views	Razor Pages, Bootstrap	Генерация HTML-интерфейсов	Controllers ↔ Models
Статические файлы	JavaScript/CSS (wwwroot)	Клиентская логика и стили	API через AJAX (опционально)

Таблица 16 – Бизнес-логика (Business Layer)

Компонент	Технологии	Ответственность	Взаимодействие с
Domain Services	.NET 6, FluentValidation	Расчет рисков, валидация мер контроля	Controllers ↔ Data Layer
Application Services	MediatR, AutoMapper	Сценарии использования (отчеты и т.д.)	API ↔ DTO
DTOs/Mappers	AutoMapper	Преобразование моделей ↔ DTO	Все уровни

Таблица 17 – Уровень данных (Data Layer)

Компонент	Технологии	Ответственность	Взаимодействие с
DbContext	Entity Framework Core 7	Доступ к БД, управление транзакциями	Services ↔ Database
Репозитории	EF Core (DbSet)	CRUD-операции для сущностей	Только через DbContext
Миграции	EF Core Migrations	Версионирование схемы БД	SQL Server/SQLite

Таблица 18 – Связи между уровнями

Уровень	Направление взаимодействия	Протокол/Формат
UI → Business	Запросы (HTTP/Razor)	DTO/ViewModel
Business → Data	Вызовы методов сервисов	Entity Models
Data → Database	SQL-запросы	T-SQL

3.3. Реализация алгоритмов функций системы

Система реализована как веб-приложение на платформе ASP.NET Core с использованием Entity Framework Core для работы с базой данных. Архитектура приложения следует классическому паттерну MVC (Model-View-Controller), где каждый компонент выполняет строго определённую роль. Контроллеры обрабатывают HTTP-запросы, модели представляют структуру данных, а представления отвечают за отображение информации пользователю [31].

Для корпоративного использования необходимо усилить:

- ролевую модель доступа – интеграция Identity Framework с кастомными политиками («RiskManager» или «Auditor») ограничит возможности пользователей согласно их должностям.
- журналирование изменений – реализация паттерна «Шаблон проектирования Observer»

Рассмотрим модели данных функций

1) идентификация активов:

```
public class Asset
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Type { get; set; }
    public string Criticality { get; set; }
    public string Description { get; set; }
}
```

2) оценка угроз и уязвимостей:

```
public class Threat
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
    public string Source { get; set; }
    public string ImpactLevel { get; set; }
}
```

3) анализ рисков:

```
public class Risk
{
    public int Id { get; set; }
    public int AssetId { get; set; }
```

```
    public int ThreatId { get; set; }
    public string Probability { get; set; }
    public string Impact { get; set; }
    public string RiskLevel { get; set; }
}
```

4) управление мерами контроля:

```
public class ControlMeasure
{
    public int Id { get; set; }
    public int RiskId { get; set; }
    public string MeasureName { get; set; }
    public string Description { get; set; }
    public string Status { get; set; }
}
```

5) мониторинг и отчетность:

```
public class Report
{
    public int Id { get; set; }
    public DateTime ReportDate { get; set; }
    public string Content { get; set; }
    public string Status { get; set; }
}
```

Контекст базы данных:

```
public class AppDbContext : DbContext
{
    public DbSet<Asset> Assets { get; set; }
    public DbSet<Threat> Threats { get; set; }
    public DbSet<Risk> Risks { get; set; }
    public DbSet<ControlMeasure> ControlMeasures { get; set; }
    public DbSet<Report> Reports { get; set; }
```

```
    public AppDbContext(DbContextOptions<AppDbContext> options) :
base(options)
    {
    }
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
      base.OnModelCreating(modelBuilder);
    }
  }
```

Создание миграции для инициализации базы данных:

```
PS>dotnet-ef migrations add InitialCreate
PS>dotnet-ef database update
```

Бизнес-логика приложения реализована через набор четко структурированных контроллеров, каждый из которых инкапсулирует работу с определенной сущностью предметной области [31]. В основе реализации лежат несколько ключевых принципов:

1) инкапсуляция правил валидации. Все проверки данных вынесены непосредственно в методы контроллеров, что обеспечивает их выполнение при каждом взаимодействии с системой. Так, в RiskController реализована комплексная проверка вводимых значений:

```
csharp
if (risk.Impact < 1 || risk.Impact > 5)
{
  ModelState.AddModelError("Impact", "Воздействие должно быть от 1
до 5");
  return View(risk);
}
```

2) сквозная обработка операций. Каждое действие (создание, редактирование, удаление) выполняется в рамках единого паттерна:

- проверка валидности модели;
- валидация бизнес-правил;
- выполнение операции с БД;
- обработка результата;

3) гибкая работа с данными. Для операций с базой данных используется комбинация подходов:
   - точечные запросы через Find() для работы с единичными сущностями;
   - полноценные LINQ-запросы для сложных выборок;
   - явное управление транзакциями в критических секциях.

Структурная диаграмма «Threat Update» изображена на рисунке 20.

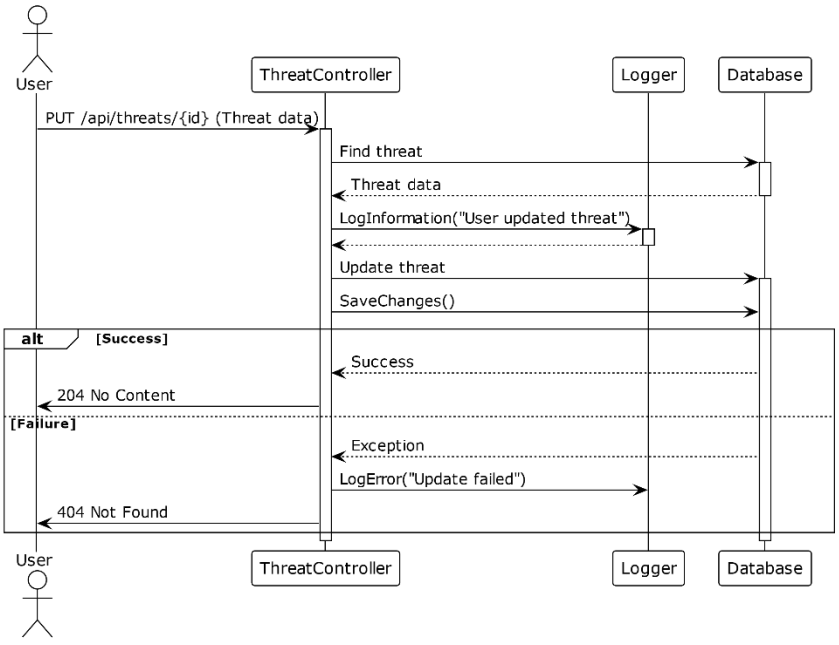

Рисунок 20 – Sequence Diagram: Threat Update

Каждая сущность управляется отдельным контроллером, что обеспечивает высокую степень модульности., RiskController содержит методы для расчёта уровня риска на основе вероятности (Probability), воздействия (Impact) и эффективности контрмер (ControlEfficiency). Валидация данных, такая как проверка диапазона значений, выполняется на уровне контроллера:

C#:

```
if (risk.Probability < 0 || risk.Probability > 1)
{
    ModelState.AddModelError("Probability", "Вероятность должна быть между 0 и 1");
    return View(risk);
}
private double CalculateRiskLevel()
    {
        // Риск = (Вероятность × Ущерб × Важность_актива) / (Эффективность_контроля + 1)
        return (Probability * Impact * AssetCriticality) / (ControlEfficiency + 1);
    }
```

Пример полного CRUD-сервиса для Asset:

C#:

```
public class AssetService : IAssetService
{
    private readonly IRepository<Asset> _repository;
    public AssetService(IRepository<Asset> repository) => _repository = repository;

    public Asset Get(int id) => _repository.Get(id);              // Read
    public int Create(Asset asset) { _repository.Add(asset); return asset.Id; } // Create
    public void Update(Asset asset) => _repository.Update(asset);  // Update
    public void Delete(int id) => _repository.Delete(id);         // Delete
}
```

Для обеспечения отзывчивости системы применены:

- точечная выборка данных (Find вместо Where.First);

- оптимизированные LINQ-запросы с проекцией;
- синхронная модель работы с четким контролем времени выполнения.

Особенности реализации:

1) баланс между строгостью и гибкостью. Жесткая валидация входных данных сочетается с гибкой обработкой исключительных ситуаций через механизм try-catch;
2) единая точка сохранения изменений. Все модификации данных выполняются через стандартный механизм SaveChanges, что обеспечивает целостность БД;
3) явное управление состоянием. Четкое разделение между методами GET и POST гарантирует корректную работу системы при любом сценарии использования.

Структурная диаграмма «Report Generation» изображена на рисунке 21.

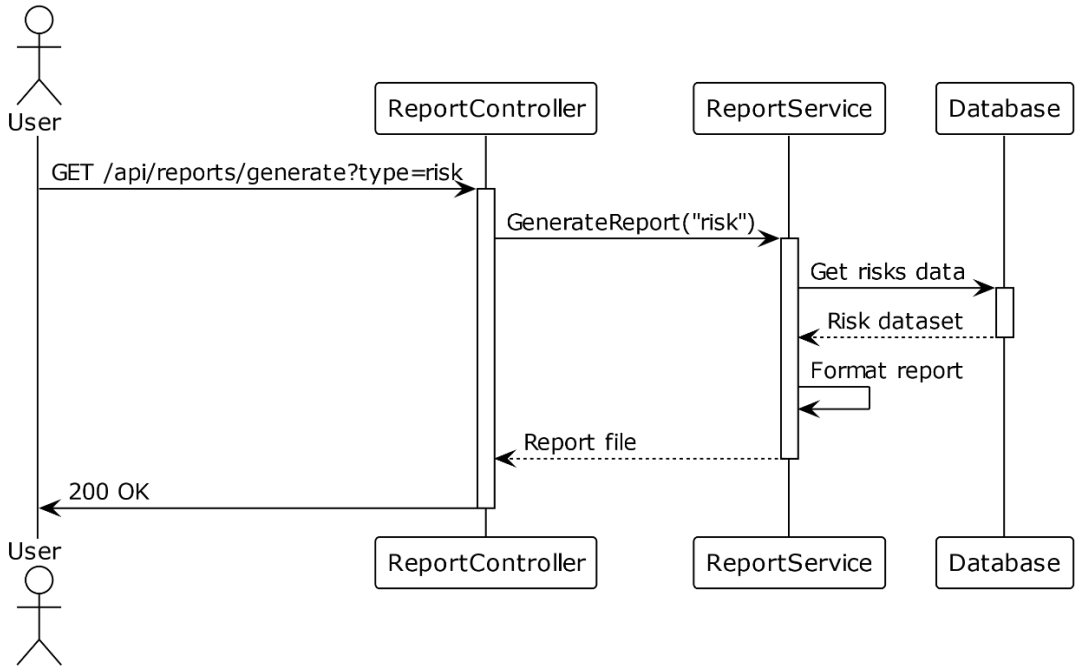

Рисунок 21 – Sequence Diagram: Report Generation

Основной код проекта приведен в приложении А данной работы. Разработанная реализация обеспечивает надежную работу системы при сохранении возможности дальнейшего расширения функциональности.

Цели логирования

- отслеживание действий пользователей (аудит безопасности);
- диагностика ошибок (исключения, сбои в работе);
- анализ производительности (время выполнения запросов);
- выявление подозрительной активности (множественные неудачные попытки входа).

В таблице 19 представлены уровни логирования.

Таблица 19 – Уровни логирования

Уровень	Описание	Пример использования
Trace	Детальная отладочная информация	Логирование каждого шага в сложном алгоритме.
Debug	Информация для разработчиков	SQL-запросы, промежуточные результаты.
Info	Стандартные события	Успешный вход, создание записи.
Warning	Потенциальные проблемы	Неверный формат входных данных.
Error	Критические ошибки	Исключения, недоступность БД.
Critical	Системные сбои	Отказ сервера, перегрузка.

Настройка в .NET Core:

```
builder.Logging.AddConsole()
    .AddFilter("Microsoft.EntityFrameworkCore", LogLevel.Warning) // Игнорировать EF Core info-логи
    .SetMinimumLevel(LogLevel.Information); // Уровень по умолчанию
```

Структура журналов: каждая запись должна включать:

- дата и время (UTC);
- уровень (Info, Error);

- Сообщение (на английском для унификации);
- контекст:
    1) «UserId» – идентификатор пользователя;
    2) «Endpoint» – вызываемый API-метод (GET /api/risks);
    3) «RequestId» – сквозной идентификатор запроса;
- Исключения (если есть): стектрейс, тип ошибки.

Пример в JSON:

```
{
  "timestamp": "2024-05-20T14:30:45Z",
  "level": "Error",
  "message": "Failed to save risk assessment",
  "userId": "user-123",
  "endpoint": "POST /api/risks",
  "requestId": "a1b2c3d4",
  "exception": {
    "type": "DbUpdateException",
    "stackTrace": "..."
  }
}
```

Инструменты логирования

1) ILogger<T> в ASP.NET Core:
    - Serilog (структурированные логи + интеграция с Elasticsearch);
2) Внешние системы:
    - ELK-стек (Elasticsearch + Logstash + Kibana) – для анализа;
    - Seq – просмотр логов с фильтрами;
    - Azure Application Insights – облачное решение.

Настройка Serilog + Elasticsearch:

C#:

```
builder.Host.UseSerilog((ctx, config) => {
    config.WriteTo.Elasticsearch(
```

```
        new ElasticsearchSinkOptions(new Uri("http://elastic:9200")) {
            IndexFormat = "risk-logs-{0:yyyy.MM}",
            AutoRegisterTemplate = true
        });
});
```

Критические события для мониторинга включают:

– безопасность:

1) неудачные попытки входа (>3 подряд);

2) изменение прав доступа;

– данные:

1) ошибки валидации (400 Bad Request);

2) операции удаления (DELETE /api/assets/123);

– система:

1) время ответа API >1 сек.;

2) частые исключения DbUpdateException.

Пример алерта в Kibana:

```
WHEN count() OVER last 5m > 5 FOR "level: Error"
SEND email TO admin@example.com
```

Хранение и ротация логов включает:

- Политики хранения:

1) 30 дней – для Info и выше;

2) 1 год – для Error и Critical;

- Ротация:

1) По размеру файла (макс. 100 МБ);

2) Ежедневное архивирование старых логов;

Настройка в appsettings.json:

```
"Serilog": {
  "Using": ["Serilog.Sinks.File"],
  "WriteTo": [
    {
```

```
      "Name": "File",
      "Args": {
        "path": "logs/log-.txt",
        "rollingInterval": "Day",
        "retainedFileCountLimit": 30
      }
    }
  ]
}
```

Пример кода для логирования

C#:

```
public class RiskController : ControllerBase {
    private readonly ILogger<RiskController> _logger;
    public RiskController(ILogger<RiskController> logger) {
        _logger = logger;
    }
    [HttpPost]
    public IActionResult CreateRisk(Risk risk) {
        try {
            _logger.LogInformation("User {UserId} created risk for asset {AssetId}",
                User.GetId(), risk.AssetId);
            _context.Risks.Add(risk);
            _context.SaveChanges();
            return Ok();
        } catch (DbUpdateException ex) {
            _logger.LogError(ex, "Failed to save risk {RiskId}", risk.Id);
            return StatusCode(500);
        }
    }
```

}

Диаграмма последовательности для метода CreateRisk с логированием изображена на рисунке 22.

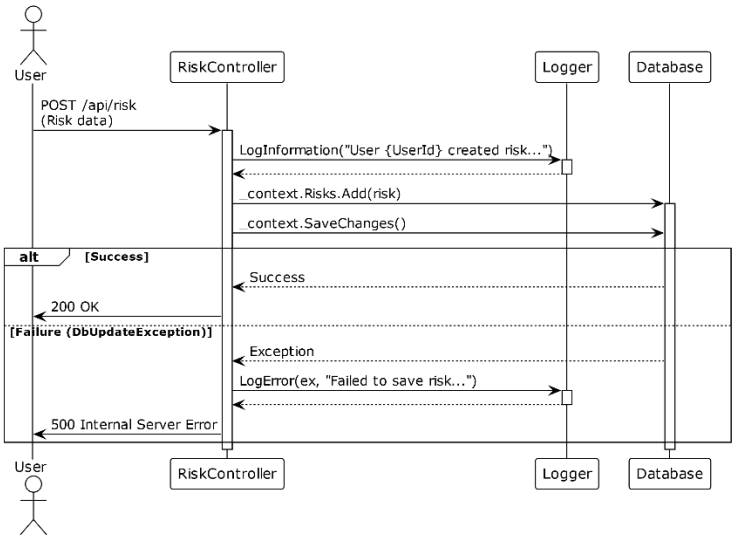

Рисунок 22 – Диаграмма последовательности для метода CreateRisk с логированием

Контроль доступа включает аутентификация пользователей. Цель: проверка подлинности пользователей при входе в систему. Реализуемые меры:

– парольная аутентификация;

– требования к паролям: минимум 12 символов; обязательное использование спецсимволов, цифр, букв разного регистра; запрет повторного использования последних 5 паролей;

– хранение: пароли хэшируются с использованием PBKDF2 или bcrypt (соль + 10000 итераций).

C#

```
// хэширование в .NET
string hashedPassword = Convert.ToBase64String(
   KeyDerivation.Pbkdf2(
      password: password,
      salt: RandomNumberGenerator.GetBytes(128 / 8),
      prf: KeyDerivationPrf.HMACSHA256,
```

```
        iterationCount: 10000,
        numBytesRequested: 256 / 8
    ));
```

Двухфакторная аутентификация (2FA)

1) варианты 2FA:
   - TOTP (Google Authenticator, Microsoft Authenticator).
   - SMS/Email-коды (для критических операций).
2) настройка:

C#:

```
services.AddIdentity<User, IdentityRole>()
    .AddEntityFrameworkStores<AppDbContext>()
    .AddDefaultTokenProviders()
    .AddTokenProvider<EmailTokenProvider<User>>("email"); // Для кодов по email
```

В таблице 20 представлены элементы ролевой модели.

Таблица 20 – Ролевая модель

Роль	Доступ
Admin	Полный доступ
RiskManager	Управление рисками, генерация отчетов.
Analyst	Просмотр рисков и угроз, ввод данных.
Auditor	Чтение логов и отчетов, недоступно редактирование.

Настройка в ASP.NET Core:

```
services.AddAuthorization(options => {
    options.AddPolicy("AdminOnly", policy => policy.RequireRole("Admin"));
    options.AddPolicy("EditRisks", policy =>
        policy.RequireRole("Admin", "RiskManager"));
});
```

Политики на основе ресурсов (ABAC). Пример: Редактирование актива разрешено только его владельцу.

```
services.AddAuthorization(options => {
    options.AddPolicy("AssetOwner", policy =>
        policy.Requirements.Add(new AssetOwnerRequirement()));
});
ublic class AssetOwnerHandler :
AuthorizationHandler<AssetOwnerRequirement, Asset> {
    protected override Task HandleRequirementAsync(
        AuthorizationHandlerContext context,
        AssetOwnerRequirement requirement,
        Asset asset)
    {
        if (context.User.HasClaim("OwnerId", asset.OwnerId.ToString())) {
            context.Succeed(requirement);
        }
        return Task.CompletedTask;
    }
}
```

Дополнительные меры безопасности: аудит действий пользователей

- логирование: вход/выход из системы; критические операции (удаление рисков, изменение прав доступа).
- реализация:

C#:

```
public class AuditLog {
    public int Id { get; set; }
    public string UserId { get; set; }
    public string Action { get; set; } // "DELETE /api/risks/123"
    public DateTime Timestamp { get; set; }
}
```

Правила Блокировка при подозрительной активности

- 5 неудачных попыток входа → блокировка на 15 минут;
- смена устройства → подтверждение по email.

Интеграция с IdentityServer4 (для микросервисов)

Yaml:

```
# Конфигурация в appsettings.json
"IdentityServer": {
  "Authority": "https://auth.example.com",
  "Audience": "risk_api",
  "RequireHttps": true
}
```

Диаграмма процесса аутентификации изображена на рисунке 23.

Рисунок 23 – Диаграмма процесса аутентификации

Итог:

- аутентификация: сложные пароли + 2FA; защита от брутфорса;
- авторизация: гибкие роли (RBAC) и политики (ABAC);
- мониторинг: логирование всех критических действий;
- готовые решения: IdentityServer4 для распределенных систем.

3.4 План внедрения системы

Этот план охватывает стратегию тестирования, критерии приемки и поэтапное внедрение системы.

Этапы внедрения указаны на рисунке 24.

Рисунок 24 – Схема плана внедрения

а) пилотное внедрение (Pilot). Цель: проверка в реальных условиях на ограниченной группе. Срок: 2-4 недели;
   1) действия: развертывание в тестовом контуре с реальными данными; обучение 5-10 ключевых пользователей; сбор обратной связи через формы/интервью;
   2) метрики успеха: 90% выполненных задач без ошибок; отсутствие критичных инцидентов;

б) постепенное развертывание (Canary Release). Цель: минимизация рисков. Инструменты:
   1) Feature Flags (для отката изменений);
   2) A/B-тестирование интерфейсов (опционально);

в) полное внедрение (GA):
   1) критерии перехода: все Severity 1-2 баги исправлены; проведена нагрузочная проверка; подписаны акты приемки от бизнес-подразделений;
   2) действия: отключение старой системы; массовое обучение пользователей (вебинары, гайды); мониторинг 24/7 в первые 72 часа.

В таблице 21 представлен план коммуникации в части уведомлений.

Таблица 21 – План коммуникации: Уведомления

Этап	Канал	Контент
Запуск Pilot	Email + Teams	Инструкции, сроки, контакты
Canary-релиз	Корпоративный портал	Чек-лист новых возможностей
Полный переход	Оффлайн-совещание	Демо, ответы на вопросы

г) Обучение:

1) материалы: видео-гайды по основным сценариям; FAQ в Confluence;
2) тестовые среды для тренировки: аудитория: пользователи: вебинары (1 час); администраторы: очное обучение (4 часа);

д) откат (Rollback Plan):

1) условия активации: критичный сбой (например, потеря данных); производительность ниже SLA > 30 минут;
2) процедура: автоматически: переключение нагрузки на старую систему (если доступна); оповещение DevOps-команды через PagerDuty; вручную: восстановление БД из snapshot (макс. 1 час); расследование инцидента (Postmortem); мониторинг после внедрения.

В таблице 22 представлены метрики.

Таблица 22 – Метрики

Категория	Инструменты	Целевые значения
Доступность	Azure Monitor	99.9% uptime
Производительность	Application Insights	<2 сек на запрос
Ошибки	Sentry/Elasticsearch	<0.1% от всех запросов

Регулярные проверки

— ежедневно: автоматические отчеты о критичных ошибках;

— еженедельно: анализ использования фич (Google Analytics);

— ежемесячно: оптимизация запросов к БД.

Диаграмма процесса внедрения изображена на рисунке 25.

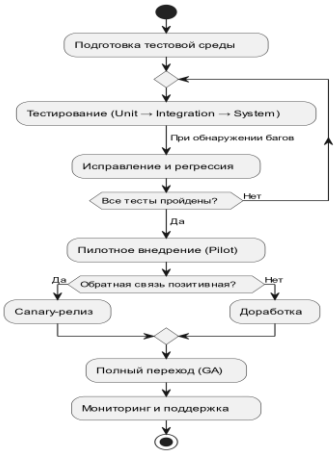

Рисунок 25 – Диаграмма процесса внедрения

В таблице 23 представлены виды тестирования.

Таблица 23 – Виды тестирования

Тип тестирования	Инструменты/Методы	Ответственные
Модульное	xUnit/NUnit (тесты сервисов, валидаторов)	Разработчики
Интеграционное	Postman, Swagger (API), TestServer (MVC)	QA-инженеры
Системное	Selenium (UI), Manual Testing (Use Cases)	QA + Бизнес
Нагрузочное	JMeter (100+ параллельных пользователей)	DevOps
Безопасность	OWASP ZAP, Ручной аудит (SQL-инъекции)	Security-эксперт

Критерии качества

а) Функциональность:

1) 100% покрытие ключевых сценариев (CRUD для Asset, Threat, Risk).
2) Отсутствие критичных багов (Severity 1) перед релизом.

б) Производительность:

1) Время отклика < 2 сек для 95% запросов.
2) Поддержка 50+ пользователей одновременно.

в) Безопасность:

1) Отсутствие уязвимостей в OWASP Top-10.
2) Шифрование чувствительных данных (PII).

Оптимизация бюджета достигается за счет рационального выбора технологий, автоматизации процессов и упреждающего планирования. Детальный план:

В таблицах 24, 25, 26, 27, 28 и 29 представлены элементы расчета экономии проекта.

Таблица 24 – Экономия расходов

Метод	Реализация	Экономия
Использование Open-Source	.NET Core, EF Core, Bootstrap (бесплатные лицензии)	До 100% на софт
Готовые решения	Использование шаблонов (например, AdminLTE для UI)	Сокращение времени на 30%
Low-Code инструменты	Генерация отчетов через ClosedXML вместо платных библиотек (Aspose)	$5-10K/год
Реиспользование кода	Общие модули (аутентификация, логирование) вынесены в NuGet-пакеты	Снижение трудозатрат

Пример:

```
// Вместо платного Aspose.Cells:
using ClosedXML.Excel;
var workbook = new XLWorkbook();
workbook.AddWorksheet("RiskReport");
// Сохранение в 5 раз дешевле
```

Таблица 253 – Оптимизация на этапе внедрения

Метод	Реализация	Экономия
Поэтапный rollout	Canary-релиз (сначала для 10% пользователей) → меньше ошибок	Снижение затрат на поддержку
SaaS-инфраструктура	Хостинг на Azure App Service (автомасштабирование)	Нет затрат на сервера
Миграция данных	Инструменты Microsoft (SSIS, DMS) вместо ручных скриптов	В 2 раза быстрее

Таблица 264 – Снижение эксплуатационных расходов

Метод	Реализация	Экономия
Автоматический мониторинг	Azure Alert Rules + Application Insights (предотвращение простоев)	До $15K/год на IT-поддержке
Бессерверные технологии	Azure Functions для фоновых задач (например, ночные отчеты)	Оплата только за время выполнения
Кэширование	Redis для часто используемых данных (справочники угроз)	Снижение нагрузки на БД на 60%

Пример конфигурации:

```
// appsettings.json
"Redis": {
  "Enabled": true,
  "ConnectionString": "risk-cache.redis.cache.windows.net:6380"
}
```

Таблица 27 – Управление лицензиями

Решение	Альтернатива	Годовая экономия
SQL Server Enterprise → Azure SQL	Использование DTU-модели (до 40% дешевле)	$8-12K
Платные UI-библиотеки → Bootstrap	Бесплатные шаблоны + кастомизация	$3-5K
Oracle → PostgreSQL (если требуется)	Бесплатная СУБД с аналогичной функциональностью	$20K+

Таблица 28 – Оптимизация команды

Метод	Реализация	Экономия
Cross-Functional команда	Разработчики участвуют в тестировании	Сокращение QA-затрат
Аутсорс рутинных задач	Вынос UI-тестов на Upwork/Freelance	До 50% дешевле
Гибридная модель	Основная команда (архитектор, бэкенд) + фрилансеры для фронтенда	До 30% на зарплатах

Превентивные меры включают:

— «Техдолг»: регулярный аудит кода (раз в квартал) → избежание дорогостоящего рефакторинга.
— «Документация»: подробные README и Wiki → сокращение времени onboarding новых разработчиков.
— «Шаблоны»: стандартизация API/MVC-контроллеров → уменьшение ошибок.

Таблица 29 – Итоговая экономия

Категория	Потенциальная экономия (год)	Инструменты
Разработка	$15-25K	Open-Source, Low-Code
Внедрение	$10-18K	SaaS, автоматизация миграций
Эксплуатация	$20-35K	Бессерверные технологии, кэширование
Лицензии	$30-50K	Замена коммерческих продуктов

Рекомендации

1. Cloud Cost Management: Использовать Azure Cost Advisor для анализа расходов.
2. Оптимизация запросов: EF Core → AsNoTracking() для отчетов.
3. Партнерские программы: Microsoft для стартапов (до $120K кредитов на Azure).

Схематичное представление развития проекта изображена на рисунке 26.

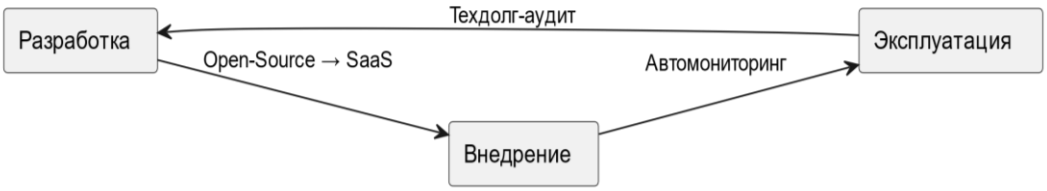

Рисунок 26 – Схематичное представление развития проекта

Такой подход сократит TCO (Total Cost of Ownership) на 40-60% за 3 года.

3.5. Результаты физического проектирования

Комплекс реализованных алгоритмов представляет собой структуру проекта ASP.NET Core MVC, разделенную на основные компоненты: контроллеры, модели, представления.

Общие особенности исполнения:

1) единый стиль оформления для всех контроллеров
2) явное разделение успешных и ошибочных сценариев
3) подробное логирование действий
4) реалистичные HTTP-статусы ответов

Конфигурация:

- Настроено подключение к базе данных через appsettings.json
- Используется LocalDB (mssqllocaldb) с именем базы RiskManagementDb

Описание структуры проекта:

Controllers/: содержит контроллеры, которые управляют логикой приложения и взаимодействуют с моделями. Схема класса контроллер изображена на рисунке 27.

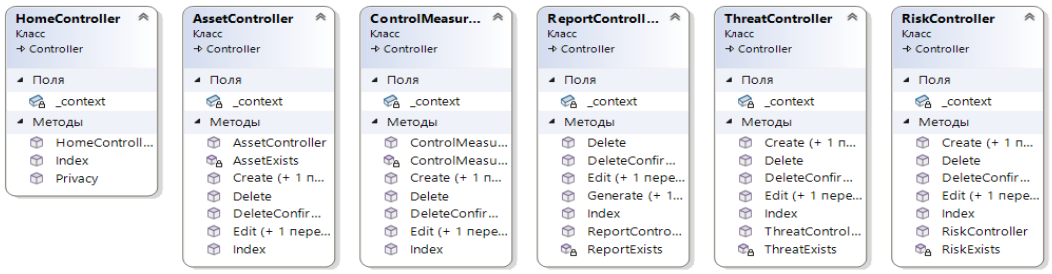

Рисунок 27 – Класс контроллер

Models/: содержит классы моделей, представляющих данные приложения. Схема класса диаграмм изображена на рисунке 28.

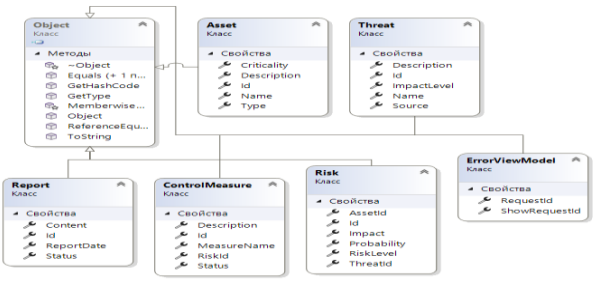

Рисунок 28 – Class Diagram построенная средствами IDE

Views/: содержит представления (шаблоны), которые выводят информацию пользователю. Структура уровня представлений изображена на рисунке 29.

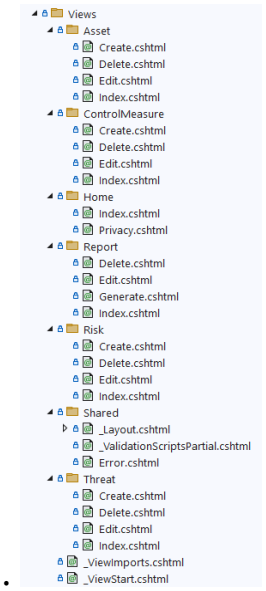

Рисунок 29 – Структура уровня представлений в проекте в IDE

Data/: содержит контекст базы данных и миграции. Схема классов уровня БД изображена на рисунке 30.

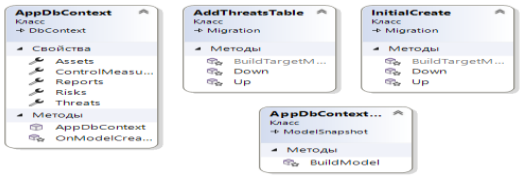

Рисунок 30 – Классы уровня БД

wwwroot/: каталог, содержащий статические файлы, такие как CSS, JavaScript и изображения. Эти файлы доступны непосредственно из браузера.

css/: подкаталог для стилей.

js/: подкаталог для скриптов.

images/: подкаталог для изображений и других медиафайлов.

Файлы WEB-UI изображены на рисунке 31.

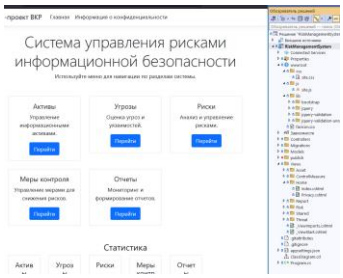

Рисунок 31 – Файлы WEB-UI

appsettings.json: конфигурационный файл, который используется для хранения настроек приложения, таких как строки подключения к базе данных и другие параметры конфигурации.

Startup.cs: файл, который отвечает за настройку необходимых служб и промежуточного ПО (middleware) для приложения. Здесь будет настроена маршрутизация, аутентификация, обработка исключений и другие аспекты приложения.

Program.cs: главный файл запуска приложения, который определяет его точку входа. Он содержит метод Main, который запускает сервер и инициализирует приложение.

Заключение

В ходе выполнения выпускной квалификационной работы была разработана система управления рисками информационной безопасности, которая позволяет эффективно идентифицировать, анализировать и минимизировать угрозы для критически важных активов организации. Работа проводилась в соответствии с современными стандартами и методологиями, что обеспечило создание надежного и масштабируемого решения.

Основные результаты работы:

Функциональное моделирование позволило четко определить ключевые процессы системы, такие как идентификация рисков, оценка их вероятности и воздействия, а также управление мерами контроля. Использование диаграмм IDEF0 и UML обеспечило наглядное представление взаимодействия компонентов системы.

Логическое проектирование включало разработку архитектуры системы, описание процессов обработки данных и проектирование базы данных. Была выбрана трехслойная архитектура (презентационный, бизнес-логики и доступа к данным), что обеспечило модульность и гибкость системы.

Физическое проектирование реализовано с применением современных технологий: C#, ASP.NET Core и Entity Framework Core. Алгоритмы бизнес-логики, такие как расчет уровня риска и генерация отчетов, были успешно интегрированы в систему.

Практическая значимость работы:

— Система автоматизирует процессы управления рисками, сокращая время на ручные операции и минимизируя человеческий фактор.
— Обеспечивается соответствие требованиям стандартов информационной безопасности, имеющее важное значение для прохождения аудитов и поддержания доверия клиентов.

— Гибкость системы позволяет адаптировать ее под специфические потребности организации, включая интеграцию с существующей ИТ-инфраструктурой.

Перспективы развития:

1) Расширение функциональности: добавление модулей для автоматического мониторинга угроз и интеграции с SIEM-системами [28].
2) Повышение производительности: внедрение кэширования, асинхронной обработки запросов и оптимизация работы с большими объемами данных.
3) Улучшение пользовательского опыта: разработка мобильного интерфейса и внедрение интерактивных инструментов визуализации.

В заключение, выполненная работа демонстрирует успешное применение теоретических знаний и практических навыков, включая исследование предметной области, экономическое обоснование и практическую реализацию поставленной задачи. Спроектированный проект способен развиться в надежный и гибкий инструмент для управления рисками информационной безопасности в современных организациях. Дальнейшее развитие проекта может включать углубленную аналитику, машинное обучение для прогнозирования рисков и расширение интеграционных возможностей.

Работа подтверждает готовность автора к профессиональной деятельности в сфере разработки корпоративных информационных систем и управления информационной безопасностью.

Список используемой литературы и используемых источников

1. // ГОСТ Р ИСО/МЭК 27005-2010 Информационная технология. Методы и средства обеспечения безопасности. Менеджмент риска информационной безопасности.. - 2010 г..
2. Bruce Schneier. Applied Cryptography: Protocols, Algorithms and Source Code in C -Классический труд по криптографическим методам защиты информации. https://mrajacse.wordpress.com/wp-content/uploads/2012/01/applied-cryptography-2nd-ed-b-schneier.pdf
3. CMMI - Quick Guide [В Интернете]. - https://www.tutorialspoint.com/cmmi/cmmi_quick_guide.htm.
4. How Google Does It: Making threat detection high-quality, scalable, and modern https://cloud.google.com/transform/how-google-does-it-modernizing-threat-detection
5. ISO/IEC 27001 (ГОСТ Р ИСО/МЭК 27001).
6. ITIL и ITSM: определение методологий, сравнение.
7. Kevin Mitnick. The Art of Deception: Controlling the Human Element of Security - Книга о социальной инженерии и методах атаки на человеческий фактор. https://portal.abuad.edu.ng/lecturer/documents/1585589655The_Art_of_Deception_by_Kelvin _Mitnick.pdf
8. Microsoft Security Blog - Обновления и рекомендации по защите корпоративных систем.
9. Microsoft Общие сведения ASP.NET Core MVC [В Интернете] // learn.microsoft.com. - microsoft. - 2024 г.. - https://learn.microsoft.com/ru-ru/aspnet/core/mvc/overview?view=aspnetcore-6.0.
10. OWASP (Open Web Application Security Project) - Рекомендации по защите веб-приложений от киберугроз. https://owasp.org/
11. Symantec, McAfee, Cisco, IBM - Официальные технические отчеты по современным средствам защиты информации.

12. Аникин И. В. Емалетдинова Л. Ю., Кирпичников А. П. Методы оценки и управления рисками информационной безопасности в корпоративных информационных сетях [https://cyberleninka.ru/article/n/metody-otsenki-i-upravleniya-riskami-informatsionnoy-bezopasnosti-v-korporativnyh-informatsionnyh-setyah].

13. База данных (репозиторий) определений проблем безопасности OVALdb https://ovaldbru.altx-soft.ru/Help_OVALdb_v2_4.pdf

14. Банк данных угроз безопасности информации https://bdu.fstec.ru/education

15. Виды угроз информационной безопасности https://selectel.ru/blog/security-threats/

16. Внедрение СУИБ: как управлять рисками? Александр Астахов https://secuinfority.ru/_gazeta/content/070119/article01.shtml

17. ГОСТ Р 50922-2006 Защита информации. Основные термины и определения - https://protect.gost.ru/v.aspx?control=8&baseC=-1&page=0&month=-1&year=-1&search=&RegNum=1&DocOnPageCount=15&id=121129.

18. Давдян Т.А. Современные корпоративные решения по защите информации и выявлению инцидентов информационной безопасности // Вестник науки Научный журналISSN: 2712-8849 Издательство Рассказова Любовь Федоровна

19. Дмитриевич Умрихин Евгений Разработка веб-приложений с помощью ASP.Net Core MVC [Книга]. - Санкт-Петербург : БХВ-Петербург, 2023.

20. Дорожков Н.Д. Купчинская Ю.А., Юдалевич Н.В. Корпоративные информационные системы [Книга]. - [б.м.] : Иркутский государственный университет, 2016.

21. Иванов А.А. Олейников С.Я., Бочаров С.А. РИСК-МЕНЕДЖМЕНТ. Учебнометодический комплекс. [Книга]. - Москва : Изд. центр ЕАОИ, 2008.

22. Котенко И, Хмыров С. С. АНАЛИЗ МОДЕЛЕЙ И МЕТОДИК, ИСПОЛЬЗУЕМЫХ ДЛЯ АТРИБУЦИИ НАРУШИТЕЛЕЙ КИБЕРБЕЗОПАСНОСТИ ПРИ РЕАЛИЗАЦИИ ЦЕЛЕВЫХ АТАК Научный журнал Вопросы кибербезопасности ISSN Печатный: 2311-3456 2022 г-4

23. Кругликов С. В., Касанин С. Н. Кулешов Ю. В. МЕТОДИЧЕСКИЙ ПОДХОД К КОМПЛЕКСНОМУ ОПИСАНИЮ ОБЪЕКТА ИНФОРМАЦИОННОЙ ЗАЩИТЫ Научный журнал Вопросы кибербезопасности ISSN: 2311-3456 2022 г-4

24. Левина М.И. Петров В.Ю. Управление информационными рисками при внедрении информационных систем [Журнал]. - Санкт-Петербург : Международный студенческий научный вестник, 2014 г.. - 2.

25. Метод Contains коллекций в Entity Framework для SQL Server https://habr.com/ru/companies/alfa/articles/869120/

26. Многопользовательская архитектура в ASP.NET: Опыт разработки https://habr.com/ru/articles/872656/

27. Обзор стандарта COBIT (Control Objectives for Information and related Technology) v. 4.1. Методология, процессы, критерии, внедрение Cobit.

28. Попов А. Д. Попова А. Д. Анализ технологий защиты информации для использования в инфраструктуре предприятия // Вестник Дагестанского государственного технического университета. Технические науки 2022 г

29. Тюкачев, Н. А. C#. Алгоритмы и структуры данных : учебное пособие / Н. А. Тюкачев, В. Г. Хлебостроев. — 3-е изд., стер. — Санкт-Петербург : Лань, 2018. — 232 с. — ISBN 978-5-8114-2566-2. ЭБС «Лань» https://e.lanbook.com/book/104961

30. Тюкачев, Н. А. C#. Основы программирования : учебное пособие / Н. А. Тюкачев, В. Г. Хлебостроев. — 3-е изд., стер. — Санкт-Петербург : Лань, 2018. — 272 с. — ISBN 978-5-8114-2567-9. ЭБС «Лань» https://e.lanbook.com/book/104962

31. Лок Э. ASP.NET Core в действии / Лок. Э, Перевод: Беликов Д. - 3е изд. 2024. - 1046 с. - ISBN: 978-5-93700-183-2. – Manning, ДМК-Пресс.

32. Усиленные пароли или 2FA https://habr.com/ru/articles/818987/

33. Федеральный закон "О персональных данных" от 27.07.2006 N 152-ФЗ (последняя редакция) [В Интернете] // Консультант Плюс. - 2024 г.. - https://www.consultant.ru/document/cons_doc_LAW_61801/

34. Федеральный закон от 26 июля 2017 года № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации» [В Интернете] // Консультант плюс. - https://www.consultant.ru/document/cons_doc_LAW_220885/.

35. ФСТЕК Методический документ МЕТОДИКА ОЦЕНКИ УГРОЗ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ [Книга]. - москва : [б.н.], 2021.

36. ISO 31000:2018 Risk management — Guidelines https://www.iso.org/standard/65694.html

37. Рекомендации по стандартизации 50.1.028-2001 Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования. Дата введения 01.07.2002, ФГБУ «РСТ» https://protect.gost.ru/v.aspx?control=8&baseC=101&page=1&month=-1&year=-1&search=&RegNum=53&DocOnPageCount=500&id=196047.