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)

Бизнес-информатика
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка сервиса представления данных для учетов тренингов»

Обучающийся Ш.Н. Ниязов
(Инициалы Фамилия) (личная подпись)

Руководитель Н.Н. Рогова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Выпускная квалификационная работа посвящена разработке сервиса представления данных для учета тренингов.

Эффективность работы компании обеспечивается наличием высококлассных специалистов. Обучение сотрудников происходит с помощью различных обучающих курсов и тренингов.

Актуальность темы работы заключается в доступности проведения анализа достижения целей компании после проведения тренингов, на основе учета появляется возможность планирования тренингов, что поможет оптимизировать ресурсы и затраты компании.

Объект исследования – система учета тренингов.

Предмет исследования – автоматизация процесса учета тренингов.

Цель выпускной квалификационной работы – разработка сервиса предоставления данных для учета тренингов.

Выпускная квалификационная работа включает три главы описывающие все этапы выполнения разработки сервиса учета тренингов.

Характеристика компании и ее деятельность за последние годы, а также бизнес-процессы, которые осуществляются подразделениями представлено в первой главе работы. Бизнес-процессы представлены в виде диаграмм на разных уровнях декомпозиции до и после автоматизации учета тренингов.

Вторая глава описывает шаги, которые были предприняты при концептуальном проектировании системы. Рассмотрены поступающие документы, которые поступают в компанию и методы их обработки при учете тренингов.

В третьей главе описывается архитектура сервиса учета тренингов и способы его реализации. Представлен контрольный пример системы и проведен расчет экономической эффективности проекта.

Заключение демонстрирует все выполненные задачи в работе.

Оглавление

Введение

Данная выпускная квалификационная работа актуальна в современной компании, где сотрудников отправляют на тренинги для получения более качественного обучения.

Сервис представления данных позволяет автоматизировать процесс учета сотрудников, прошедших тренинги и проведение тренингов с учетом требований компании. Автоматизированный документооборот во время учета позволит минимизировать допущения ошибок при назначении сотрудников для проведения тренингов и исключить дублирование проведения тренингов одной группе сотрудников.

Цель выпускной квалификационной работы – разработка сервиса предоставления данных для учета тренингов.

Объект исследования – система учета тренингов.

Предмет исследования – автоматизация процесса учета тренингов.

Задачи выпускной квалификационной работы:

– провести исследование деятельности компании, включая анализ ее бизнес-процессов и основных операций;

– идентифицировать процессы, требующие автоматизации, и определить наиболее приоритетные с точки зрения повышения эффективности и сокращения ошибок;

– провести логическое проектирование сервиса, включающее определение функциональных требований, структуры базы данных и взаимодействия пользователей с сервисом;

– реализовать сервис представления данных для учета тренингов;

– рассчитать экономическую эффективность разработки и внедрения сервиса.

В работы были использованы программные продукты и специализированные программы позволяющие разрабатывать и моделировать

различного рода схемы и диаграммы, показывающие структуру организации и существующие бизнес-процессы компании.

Специализированная программа AllFusion ERwin Data Modeler обеспечила создание диаграмм с применением методологии IDEF0 и проектирование моделей данных, программный продукт StarUML обладает необходимыми инструментами и позволяет моделировать различные диаграммы для демонстрации архитектуры сервиса и логического проектирования.

Работа включает три главы в которых описаны все пункты по изучению, проектированию и разработке сервиса для учета тренингов.

Характеристика компании и ее деятельность за последние годы, а также бизнес-процессы, которые осуществляются подразделениями представлено в первой главе работы. Бизнес-процессы представлены в виде диаграмм на разных уровнях декомпозиции до и после автоматизации.

Вторая глава описывает шаги, которые были предприняты при концептуальном проектировании сервиса представления данных. Рассмотрены документы, которые поступают менеджерам компании при проведении тренингов и методы их обработки.

В третьей главе описывается архитектура проекта и способы реализации. Представлен контрольный пример сервиса и проведен расчет экономической эффективности проекта.

Заключение демонстрирует все выполненные задачи во время выполнения выпускной квалификационной работы.

Глава 1 Функциональное моделирование учета проведения тренингов в компании

1.1 Характеристика индивидуального предприятия «AJAYYP SOWDA»

Индивидуальное предприятие «AJAYYP SOWDA» оказывает услуги строительства по разным направлениям.

Основная цель компании – получение прибыли за счет оказания качественных услуг в сфере строительства.

Задачами компании является выполнение работ, связанных со строительством. Выполнение работ высокого качества и без задержек в сроках.

Основные направления индивидуального предприятия:

– строительство жилых и нежилых зданий;

– выполнения строительно-монтажных работ;

– выполнения строительства под заказ клиента;

– выполнение электромонтажных работ с нуля;

– монтаж отопительных систем и систем кондиционирования воздуха;

– выполнение гидроизоляционных работ.

В состав организационной структуры предприятия входят различные отделы, такие как бухгалтерия, отдел правового обеспечения и судебной защиты, производственно-технический отдел, отдел развития и информационных технологий, отдел персонала, отдел тренингов.

Главная задача отдела тренингов — это развитие компании и увеличение ее дохода за счет развития профессиональных компетенций сотрудников компании, которые занимаются непосредственно обслуживанием клиентов и продажами.

Как и все современные строительные компании руководство внедряет новейшие информационные технологии для оптимизации деятельности различных отделов и отдел тренингов не является исключением.

На рисунке 1 показана организационная структура индивидуального предприятия «AJAYYP SOWDA».

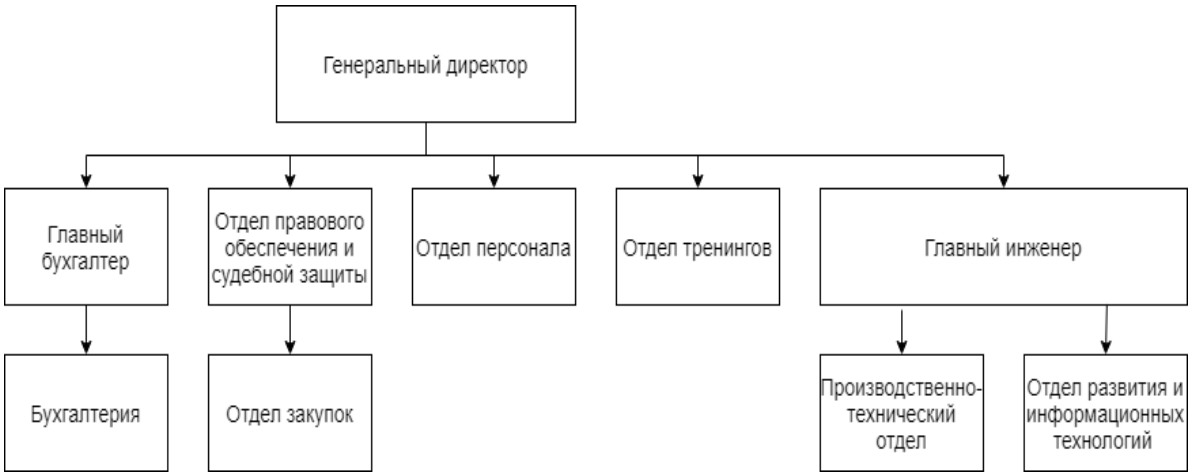

Рисунок 1 – Организационная структура индивидуального предприятия «AJAYYP SOWDA»

Отдел бухгалтерии и все ее сотрудники на протяжении рабочего времени выполняют функции, связанные с формированием документов для отчетности перед государственными органами. На основе полученных табелей рабочего времени рассчитывают заработную плату сотрудников компании.

Отдел персонала ведет набор сотрудников в компанию на основе полученных заявок от руководства или изменения штатного расписания. Также отдел занимается проведением собеседования и осуществляет помощь в сборе необходимых документов кандидатам на должность.

Производственно-технический отдел занимается разработкой проектной документации (технические чертежи, расчеты, спецификации и т.п.), планированием и организацией строительства и ведением исполнительной документации.

Отдел тренингов не имеет собственной автоматизированной системы, чтобы вести учет тренингов и ведение данных о сотрудниках, которые прошли или собираются пройти тренинги в текущем году.

Процесс ведения учета тренингов обеспечит:

– ведение данных о сотрудниках, которые прошли обучение на тренингах;

– получать информацию о проводимых тренингах в компании;

– вести планирование и список сотрудников, которым необходимо пройти тренинги;

– составлять своевременные списки сотрудников, которых необходимо отправить на тренинг.

Рассмотрим деятельность отдела тренингов, который находится в подчинении генерального директора и выполняет полученные поручения по проведению тренингов среди персонала компании.

1.2 Концептуальное моделирование учета тренингов в компании

1.2.1 Выбор технологии концептуального моделирования учета тренингов в компании

Концептуальное моделирование процесса поможет выявить данные, с которыми работают сотрудники, направлявшие персонал на тренинги. Моделирование проводится с помощью построения диаграмм в разных нотациях [9].

Принято считать самыми распространёнными нотация концептуального моделирования BPMN, UML, IDEF0, DFD, которые позволяют провести моделирование в полной мере и провести декомпозицию деятельности или отдельного процесса на несколько уровней или дополнительных диаграмм.

BPMN представлена как нотация, которая позволяет проводить моделирование процессов или целых подразделений [8].

«UML — унифицированный язык моделирования (Unified Modeling Language) — это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем» [10].

IDEF0 представлена как нотация, которая показывает деятельность предприятия с абстрактной стороны. Пользователь, использующий программу с данной нотацией, имеет спектр инструментов для моделирования, что очень удобно и легко.

DFD — нотация, которая показывает, как устроен поток информации в виде документов, файлов и т.п. За основу в нотации DFD взяты хранилища данных и внешние объекты, которые взаимодействует между собой в процессе.

Таким образом можно рассмотреть деятельность отдела тренингов с нулевого уровня до уровня, который может рассмотреть все функции и процессы отдела вплоть до документов и информации, которая передается между сотрудниками отдела [1].

1.2.2 Моделирование бизнес-процессов отдела тренингов

Отдел тренингов занимается обучением персонала разным тематическим основам, которые потом применяются в рабочей обстановке. Определим какие функции выполняются в отделе тренингов и как происходит учет прохождения и проведения тренингов в компании.

Сам процесс начинается, когда приходит в отдел поручение от генерального директора о том, что необходимо провести тренинги и с какой группой сотрудников. В отделе составляются списки сотрудников, которые объединятся в группы. Также нужно определить какой именно тренинг на данный момент наиболее необходим для той или иной группы сотрудников. Подбор тренингов происходит исходя из актуальности проведения и применимости в бизнес-процессах компании.

Сотрудники отдела тренингов формируют списки сотрудников, которые получают учебные тренинги вручную, вся документация хранится на бумажных носителях. Списки тренингов поступают к сотрудникам отдела в виде файла на электронную почту или в виде бумажного документа. При таком виде обработки и получении информации сложно вести учет проведения тренингов в компании. Сотрудники отдела тренингов не имеют возможности вести учет сотрудников, которые получили тренинги и те, которые нет.

Для более подробного изучения процесса, всех поступающих документов для обработки и изучения строится диаграмма или схема с применением разных методологий, которые обеспечивают декомпозицию процесса на подпроцессы с подробным описанием.

Разработчики сервиса получат возможность на основе диаграмм реализовать эффективный сервис представления данных для учета тренингов.

На рисунке 2 представлена диаграмма смоделированного процесса учета тренингов в отделе «как есть» в выбранной нотации.

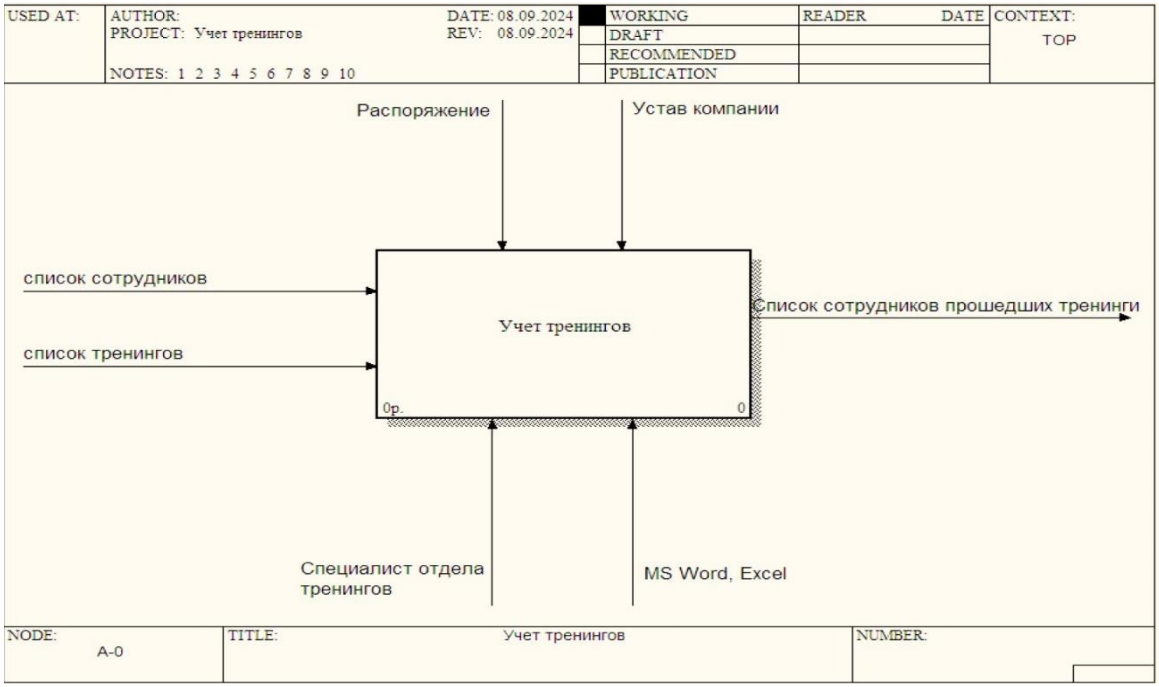

Рисунок 2 – Модель первого уровня процесса «Учет тренингов»

Подробнее провести анализ и рассмотреть, как происходит процесс учета тренингов поможет декомпозиция контекстной диаграммы. Декомпозиция процесса путем разбиения и снижения на уровень ниже покажет подпроцессы при учете тренингов [6].

Рисунок 3 показывает декомпозицию модели первого уровня.

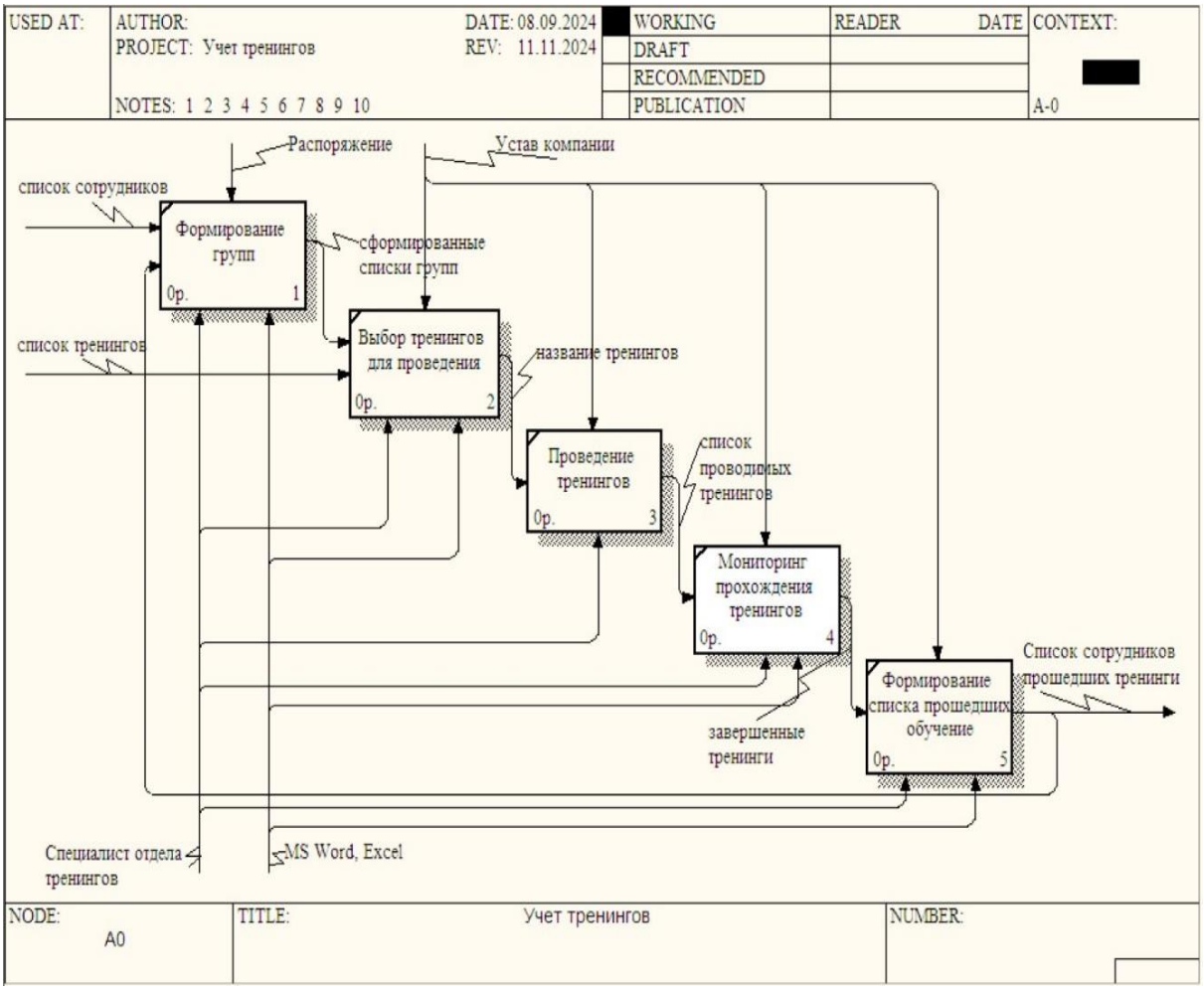

Рисунок 3 – Декомпозиция модели процесса учета тренингов

Сотрудник отдела тренингов формирует группы на основе полученного списка сотрудников, по итогу получается несколько групп которые проходят тренинги. Для дальнейшего прохождения необходимо для каждой группы

выбрать тренинг, который проводится, а в некоторых случаях и несколько тренингов для одной граппы сотрудников. Когда тренинги выбраны и определены группы проводятся, собственно, сами тренинги и формируются списки сотрудников, которые прошли тренинги с датами начала и окончания тренинга. Списки учитываются в ходе формировании групп, чтобы не было повторений среди сотрудников, которые уже прошли данные тренинги.

1.2.3 Разработка и анализ модели бизнес-процесса «КАК ЕСТЬ»

Основываясь на диаграммы процесса «как есть» можно провести анализ получения и обработки данных в процессе учета тренингов в компании. В механизмах диаграммы видно, что процессами управляет специалист отдела тренингов, который работает в офисных программах, таких как Word и Excel.

Офисные программы, которые не предназначены для ведения учета какого -либо процесса не дают возможности сотрудникам учитывать данные о проводимых тренингах в компании.

Чтобы сотрудники компании равномерно и эффективно получали знания путем занятий на тренингах, необходимо проводить их с определенной периодичностью разбив сотрудников компании на группы по специализации.

Для удобства специалистов отдела тренингов в получении информации о сотрудниках, прибывших на обучение, требуется единый автоматизированный сервис для удобства работы всех сотрудников отдела тренингов. При учете тренингов все проводимые и тренинги, которые запланированы на следующие месяцы должны храниться в системе, все списки с сотрудниками, проходящими тренинги и те, которые только должны направляться на тренинги также должны храниться в сервисе.

Создание единого автоматизированного сервиса для учета тренингов позволит сократить рутинную работу сотрудников отделов, где формируются списки сотрудников для обучения и сократит потерю данных о сотрудниках, которые уже прошли список тренингов.

Создание автоматизированного сервиса учета тренингов является главной целью в индивидуальном предприятии «AJAYYP SOWDA».

Автоматизированный сервис учета тренингов обеспечит выполнение операций:

– ввод данных о сотрудниках;

– ввод данных о тренингах;

– выбор данных из выпадающих списков;

– формирование списков;

– получение данных о сотрудниках, проходивших тренинги;

– формирование списка тренингов;

– формирование списка сотрудников, которые должны пройти тренинги.

Разработка сервиса представления данных позволит принимать и хранить данные без потери их на этапе получения. Также сервис обеспечит учет проведения тренингов и сотрудников, которые получили обучение или должны его получить в ближайшее время.

Специалисты отдела маркетинга получат возможность сидеть актуальные данные о тренингах, которые больше проводились, сотрудников, которые больше всех были на обучении и списки тех сотрудников, которые не получали обучения в виде тренингов в компании.

На рисунке 4 показан новый процесс по учету тренингов, который учитывает внедрение сервиса представления данных.

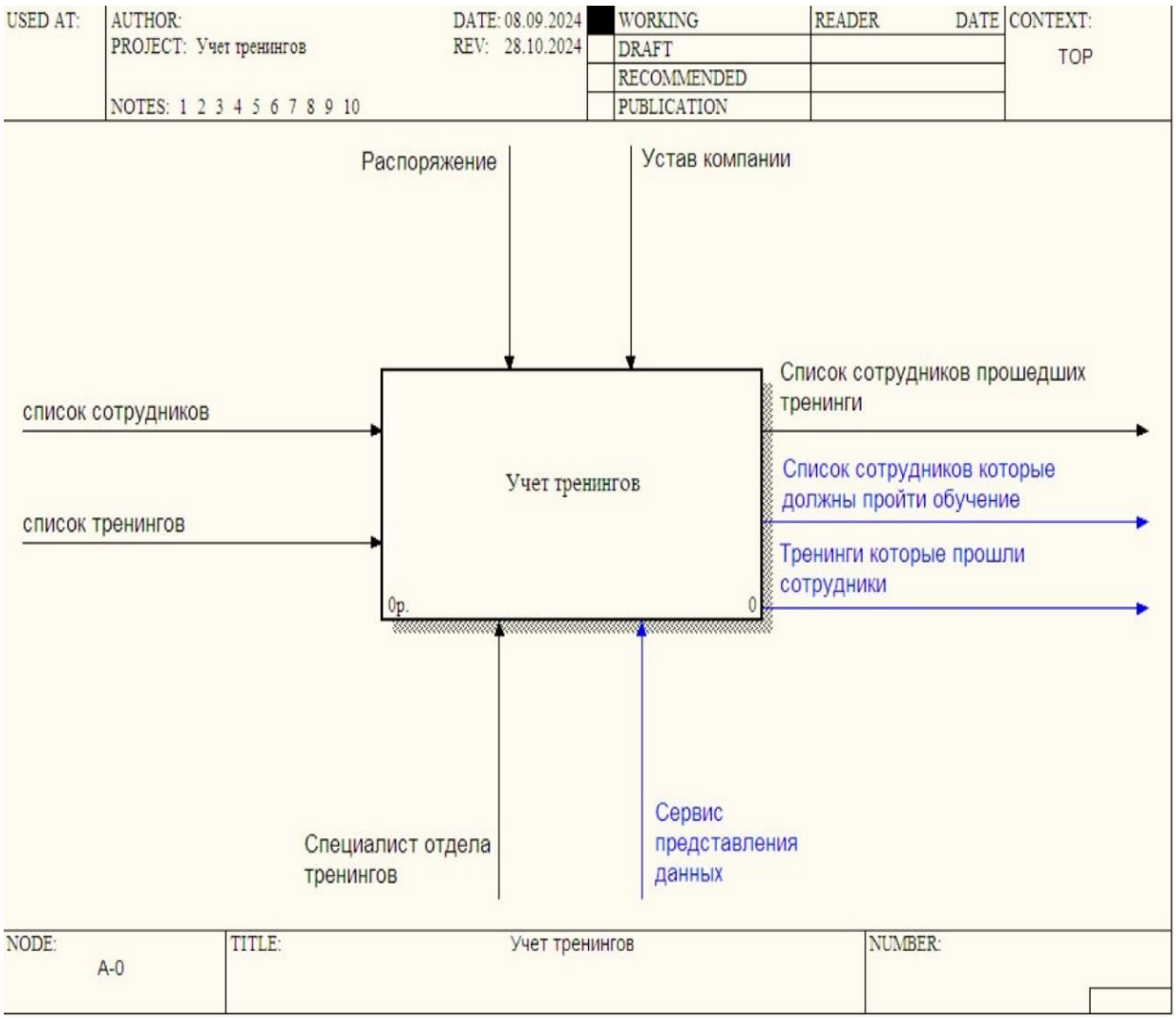

Рисунок 4 – Диаграмма модели «как должно быть» первого уровня с внедрённым сервисом

Благодаря разбиению процесса на несколько блоков можно понять как процесс выглядит после внедрения сервиса. Обновленный процесс показан на рисунке 5. Внедрение сервиса изменит выполнение нескольких блоков процесса, которые выполняются намного быстрее и эффективнее.

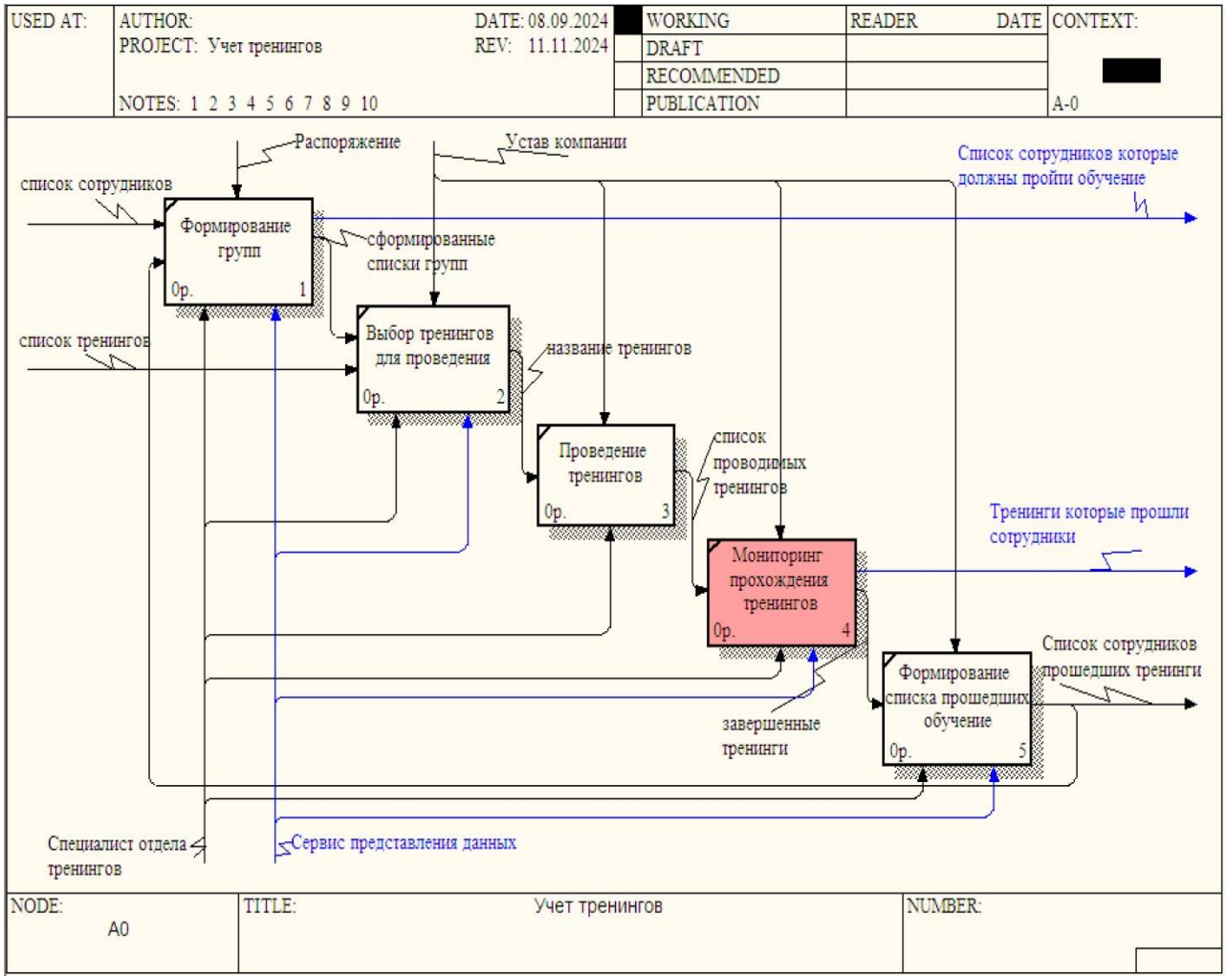

Рисунок 5 – Диаграмма декомпозиции модели «как должно быть» с внедренным сервисом

После внедрения сервиса представления данных для учета тренингов изменятся подпроцессы которые формируются на основе обновления всего процесса учета. Также в процессе изменились данные исходящие, которые получены в процессе учета тренингов, такие как тренинги, которые прошли сотрудники и данные сотрудников, которые должны пройти тренинги.

1.3 Анализ существующих разработок

Рассмотрим аналоги сервисов представления данных для учета тренингов или других обучающих программ.

«ТопФактор: Управление талантами» для оценки персонала [2].

Функциональные возможности:

– повышение вовлеченности;

– при сокращении штата;

– формирование нужных навыков;

– установка стандартов;

– определение рыночной ценности персонала;

– выявление талантов;

– оценка персонала;

– развитие и обучение персонала по результатам оценки;

Стоимость подписки за 12 месяцев – 11880 рублей.

Ведус. Расписание занятий [3].

Основные возможности:

– поддержка различных структур образовательных программ учреждения;

– планирование групповых и индивидуальных занятий, ведение обучающихся групп, работа с архивными группами;

– построение графиков групповых и индивидуальных занятий;

– контроль отсутствия пересечений графиков занятий, графиков учащихся;

– ведение истории групп, в том числе фиксация переносов дат начала занятий и замен преподавателей;

– фиксация пропусков занятий, либо присутствия учащихся на занятиях.

Стоимость ПО: 3200 руб. в месяц.

Программный продукт «1С: Управление учебным центром» [4].

«Основные функциональные возможности:

– планирование и анализ деятельности учебного центра;

– составление расписания курсов и занятий;

– проведение набора слушателей, подготовка документов, взаимодействие с контрагентами;

– управление движением контингента слушателей, подготовка, формирование приказов, анализ движения;

– формирование и учет документов об обучении, взаимодействие с ФРДО;

– оплата курсов слушателями и контроль взаиморасчетов;

– учет торговых операций;

– учет платного питания, организуемого учебным центром;

– учет учебного процесса:

– учет посещаемости;

– учет успеваемости;

– электронный журнал учета занятий с возможностью печати;

– контроль посещаемости с помощью табеля;

– настройка конфигурации и администрирование» [6].

Стоимость ПО: для одного пользователя – 30000 рублей, для 5-ти пользователей – 51600 рублей.

Сравнение программных средства показано в сравнительной таблице 1.

Таблица 1 – Сравнительная характеристика ПО

Характеристики	Программное обеспечение		
	ТопФактор	Ведус	1С: Управление учебным процессом
Настройка в соответствии с объектом автоматизации	+	+	+

Продолжение таблицы 1

Характеристики	Программное обеспечение		
	ТопФактор	Ведус	1С: Управление учебным процессом
Финансовый учет и отчетность	+	+	+
Управление расписанием и регистрацией	+	+	+
Учет посещаемости и успеваемости	-	+	-
Интеграция с другими системами	-	-	+

Все программные продукты имеют большой функционал, который применяется в больших компаниях и помогает автоматизировать сразу несколько процессов. Для учета тренингов необходимо создать узконаправленный сервис представления данных, для нужд отдела тренингов.

Таким образом целесообразно разработать собственный сервис представления данных для отдела тренингов, для выполнения функции учета. Однако необходимо учесть бюджет на разработку, включающий затраты на заработную плату разработчиков, тестировщиков, проектировщиков и группы внедрения системы в отдел тренингов.

1.4 Постановка задачи на разработку сервиса представления данных

На основе проведенного анализа программных продуктов, которые могут предоставить аналогичные функции для учета тренингов в компании определяются требования к разработке сервиса.

Сервис представления данных для учета тренингов должен позволять:

– вести данные о сотрудниках компании;

– просматривать список сотрудников, прошедших тренинг;

– формирование списка тренингов, которые проводятся;

– формирование список данных проведения будущих тренингов.

Нефункциональные требования к сервису:

– сервис должен обрабатывать не менее 50 запросов при пиковой нагрузке;

– меню на русском языке;

– доступность интерфейса;

– простота при интеграции;

– поддержка увеличения нагрузки.

Выводы по первой главе

Первая глава описывает проведённое изучение деятельности компании «AJAYYP SOWDA». Деятельность компании была представлена в виде структуры управления компанией и диаграммами процесса учета тренингов в нотации IDEF0. С помощью диаграмм были рассмотрены процессы «как есть» и «как должно быть». Процесс учета тренингов изменится после разработки и внедрения сервиса представления данных для учета тренингов. Специалисты отдела тренингов получат возможность вести данные о тренингах и сотрудниках, которые их прошли с помощью сервиса. На основе обзора программных продуктов, которые предлагают аналогичные функции, но не достаточные для учета тренингов в компании было обосновано решение собственной разработки сервиса представления данных для учета тренингов.

Глава 2 Проектирование сервиса представления данных для учета тренингов

2.1 Модели и методологии проектирования и разработки сервиса представления данных для учета тренингов

Методология проектирования и разработки поможет реализовать правильно и без лишних издержек программный продукт.

Рассмотрим некоторые из методологий, которые являются популярными при проектировании.

«Гибкая методология разработки (Agile software development) – манифест, содержащий основные ценности и принципы, на которых базируются подходы к управлению проектами, который решает проблемы традиционного проектного менеджмента. Agile подходит для инновационных проектов. Гораздо меньше он подходит для процессной деятельности. Эти подходы подразумевают интерактивную разработку, с периодическими обновлениями требований от заказчика и их реализацию посредством самоорганизующихся команд, сформированных из экспертов разного профиля. Под термином «гибкая методология разработки» следует понимать подходы на основе данного манифеста, или фреймворки» [2].

«Методология Agile создана как противоположность традиционной линейной методологии «водопад», подразумевая итеративную и пошаговую разработку ПО, что минимизирует риски. Работа с применением гибкой методологии состоит из серии коротких циклов (итераций), длительностью 2-3 недели. Каждая итерация включает в себя этапы планирования, анализа требований, проектирование, разработку, тестирование и документирование. По завершению каждой итерации команда предъявляет заказчику «осязаемые» результаты работы, например, первичную версию продукта или часть функционала, которую можно посмотреть, оценить, протестировать, а потом доработать или скорректировать» [2].

«Kanban – метод управления разработкой, при котором задачи распределяются равномерно между всеми участниками команды разработки, реализует принцип «точно в срок», и ограничивается одновременное максимальное количество выполняемых задач» [3].

«Основные принципы:

- визуализация – доска с карточками-задачами;
- имеется план разработки, отсортированный по приоритету;
- ограничение одновременно выполняемых задач;
- постоянная оптимизация процессов.

Из этих принципов можно сформулировать и ограничения метода:

- при наличии срочных задач их невозможно запустить в разработку до завершения хотя бы одной из задач в работе;
- сложно отслеживать качество выполнения задачи и эффективность отдельного сотрудника;
- команда должна работать как единый механизм;
- сложно совместить кросс-функциональные команды на одной доске;
- не предназначен для долгосрочного планирования» [3].

«Scrum — это фреймворк гибкой разработки ПО, который считается методологией «по умолчанию». Для многих является синонимом Agile. По статистике за 2016 год, предоставленной VersionOne, Scrum используют 58% «гибких» компаний. При этом её поддерживают многие SaaS-платформы. Например, решение ServiceNow, частью которого является инструмент Agile Development» [15].

«Используя методологию Scrum, представитель заказчика плотно работает с командой разработки, расставляя приоритеты в списке требований к продукту (Product Backlog). Этот список состоит из баг-фиксов, функциональных и нефункциональных требований — из всего, что нужно сделать для реализации рабочего приложения» [15].

«Функциональные элементы, добавляемые в бэклог, называют историями. Каждая задача оценивается в определенное количество очков. Очки являются абстрактной метрикой и для её оценки могут использоваться самые разные шкалы (например, ряд Фибоначчи или степени двойки). На основании списка требований заказчика команда определяет функции, которые хочет реализовать, и начинает свой спринт. Обычно он длится 30 дней. В конце подсчитывается общее количество очков, набранных командой за спринт (скорость). Это позволяет более четко планировать следующие спринты» [15].

Гибкая методология разработки Agile наиболее подходящая для реализации сервиса для учета тренингов в компании. Методология обеспечивает реализовывать программное обеспечение пошагово, что снижает возникновение рисков.

2.2 Логическая модель сервиса представления данных для учета тренингов

Предметом исследования является – автоматизация учета тренингов в отделе.

Рассмотрим один из процессов «Учет данных о сотрудниках прошедших тренингов» более наглядно на диаграмме последовательности, рисунок 6. Процесс начинается, когда сотрудник отдела открывает сервис и открывает список с данными сотрудников, выбирает определенного сотрудника, после чего открывает список тренингов и выбирает тренинг, после чего ведет поиск сотрудника, тем самым проверяется прохождение тренинга сотрудником.

Аналогичным образом можно создать остальные диаграммы последовательности [2] согласно действующим лицам и их функциям.

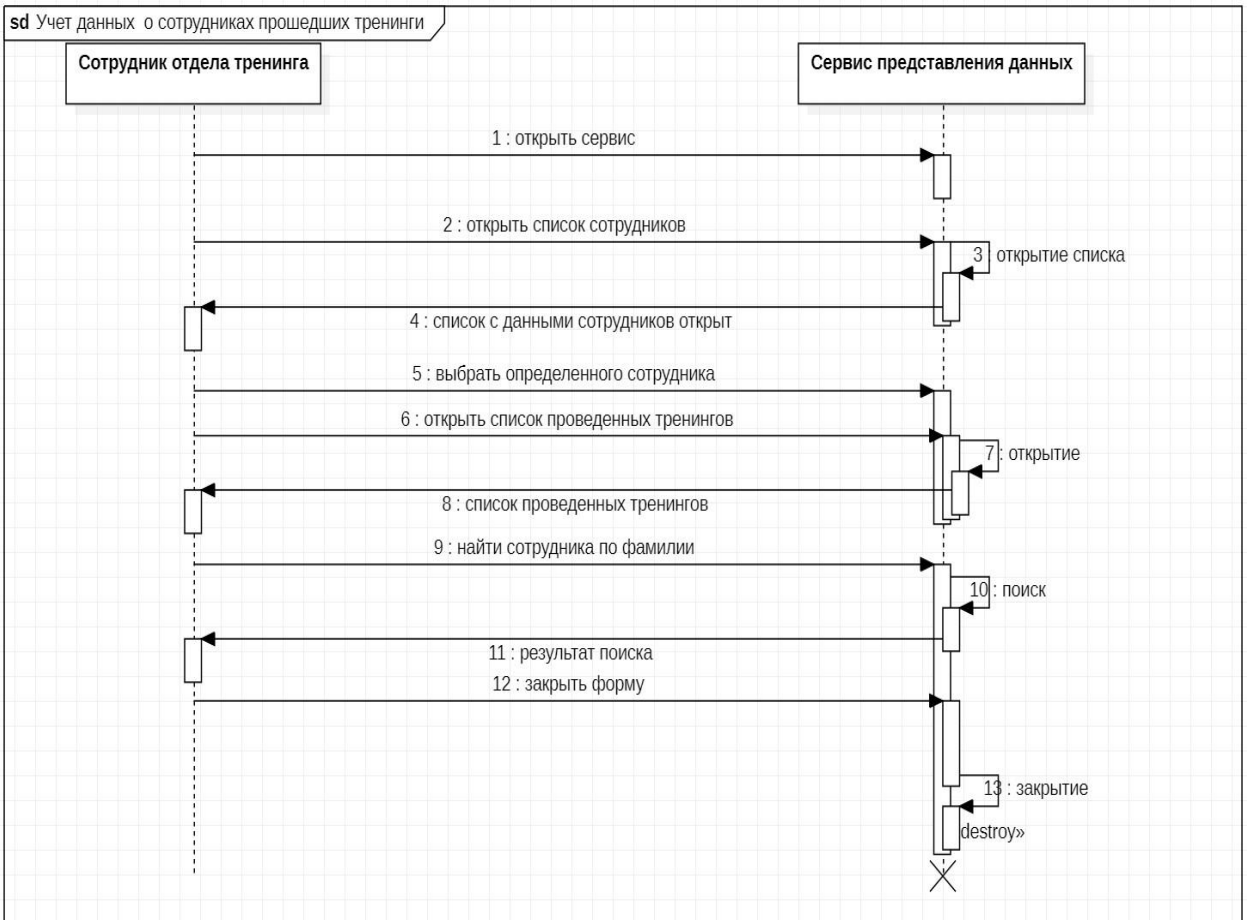

Рисунок 6 – Диаграмма последовательности процесса «Выбор тренинга»

Взаимодействие сотрудника отдела с сервисом подробно описывает шаги пользователя при выполнении определенных функций. Функции, которые пользователь сможет выполнять в сервисе показаны с помощью диаграммы вариантов использования, рисунок 7.

Сервис представления данных для учета тренингов обеспечит сотруднику отдела просмотр данных о сотрудниках в списках, обеспечит ведение данных о проводимых тренингах в компании, вести учет данных о сотрудниках, прошедших тренинги и вести учет о сотрудниках, которые только должны пройти тренинги.

Администратор сервиса сможет добавлять данные о сотрудниках и тренингах, а также удалять данных при необходимости.

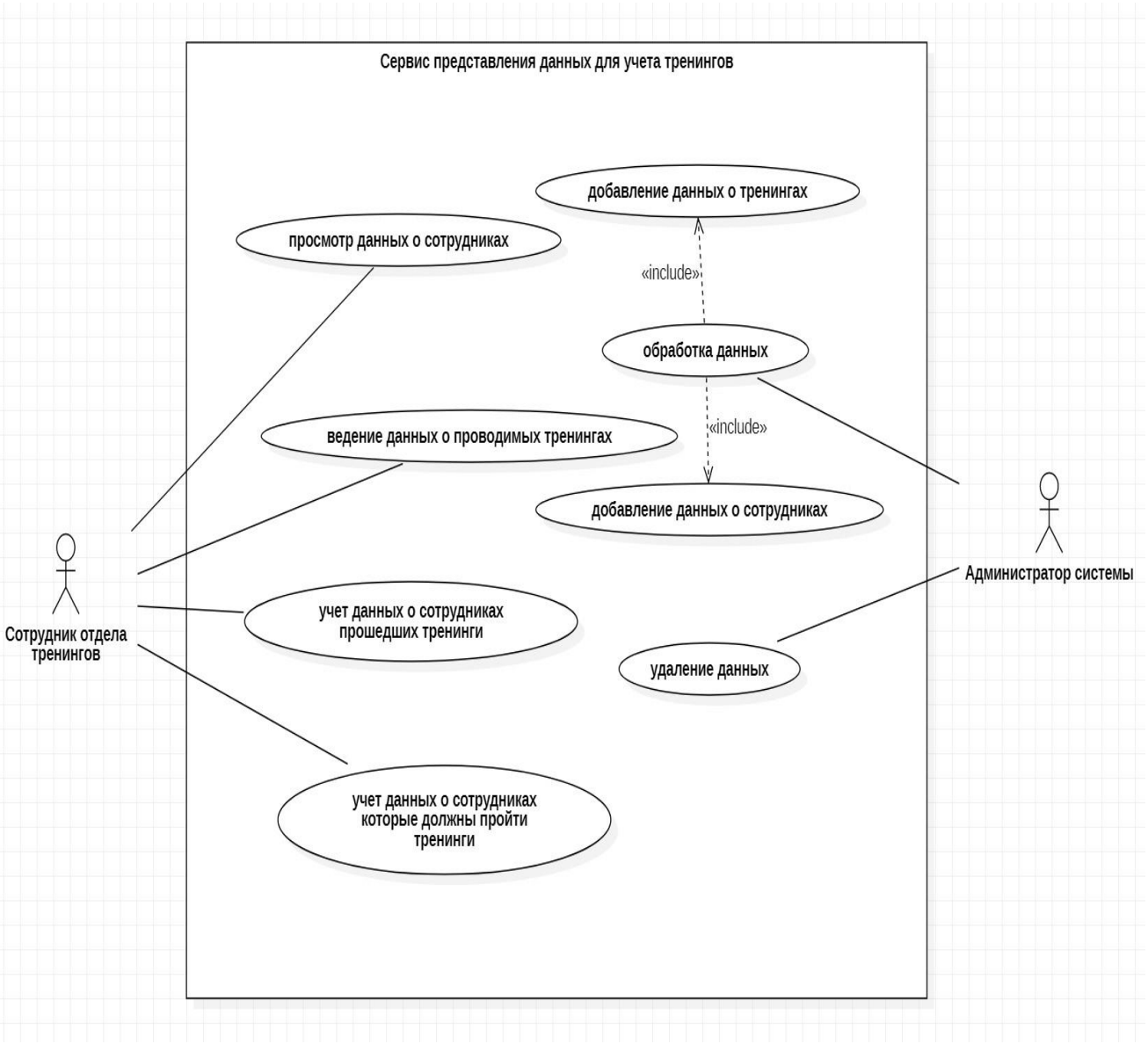

Рисунок 7 – Диаграмма вариантов использования сервиса для учета тренингов

Благодаря построенным диаграммам последовательности и вариантов использования появляется возможность построить правильно процесс учета и выделить компоненты, которые необходимы для реализации сервиса для учета тренингов. На основе чего можно смоделировать диаграмму и представить ее на рисунке 8.

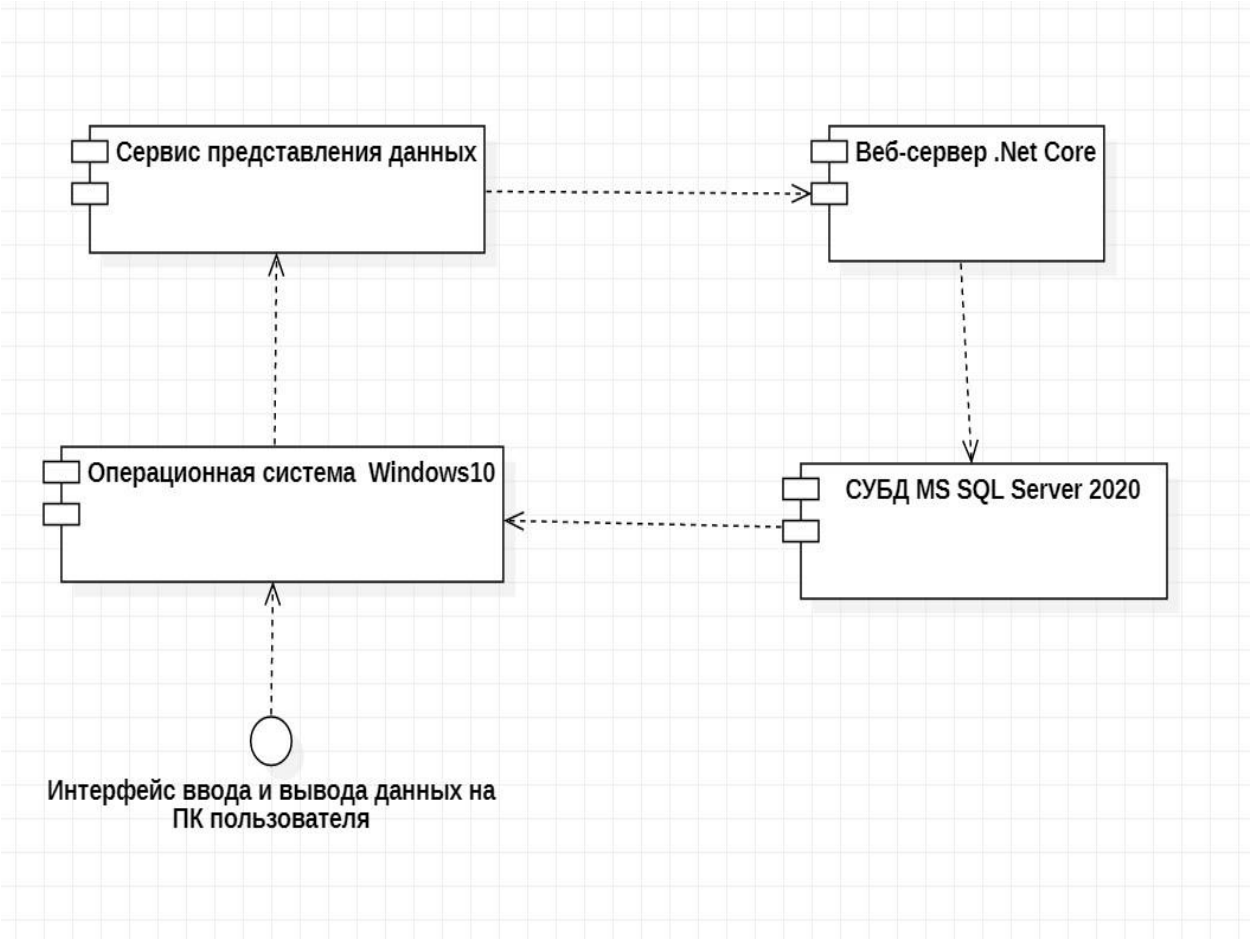

Рисунок 8 – Диаграмма компонентов системы учета тренингов

Обработка всех методов сервиса учета тренингов осуществляется с помощью классов. Классы должны содержать атрибуты и методы обработки данных. Также классы отражают взаимосвязь между объектами программного продукта и обеспечивают выводить информацию на интерфейс пользователю [10].

Объектами сервиса являются:

– тренинги, которые проводятся в компании;

– отделы, сотрудники которых проходят тренинги;

– информация о сотрудниках, которых необходимо отправить на обучение;

– данные о процессе проведения тренингов.

Диаграмма классов показана на рисунке 9.

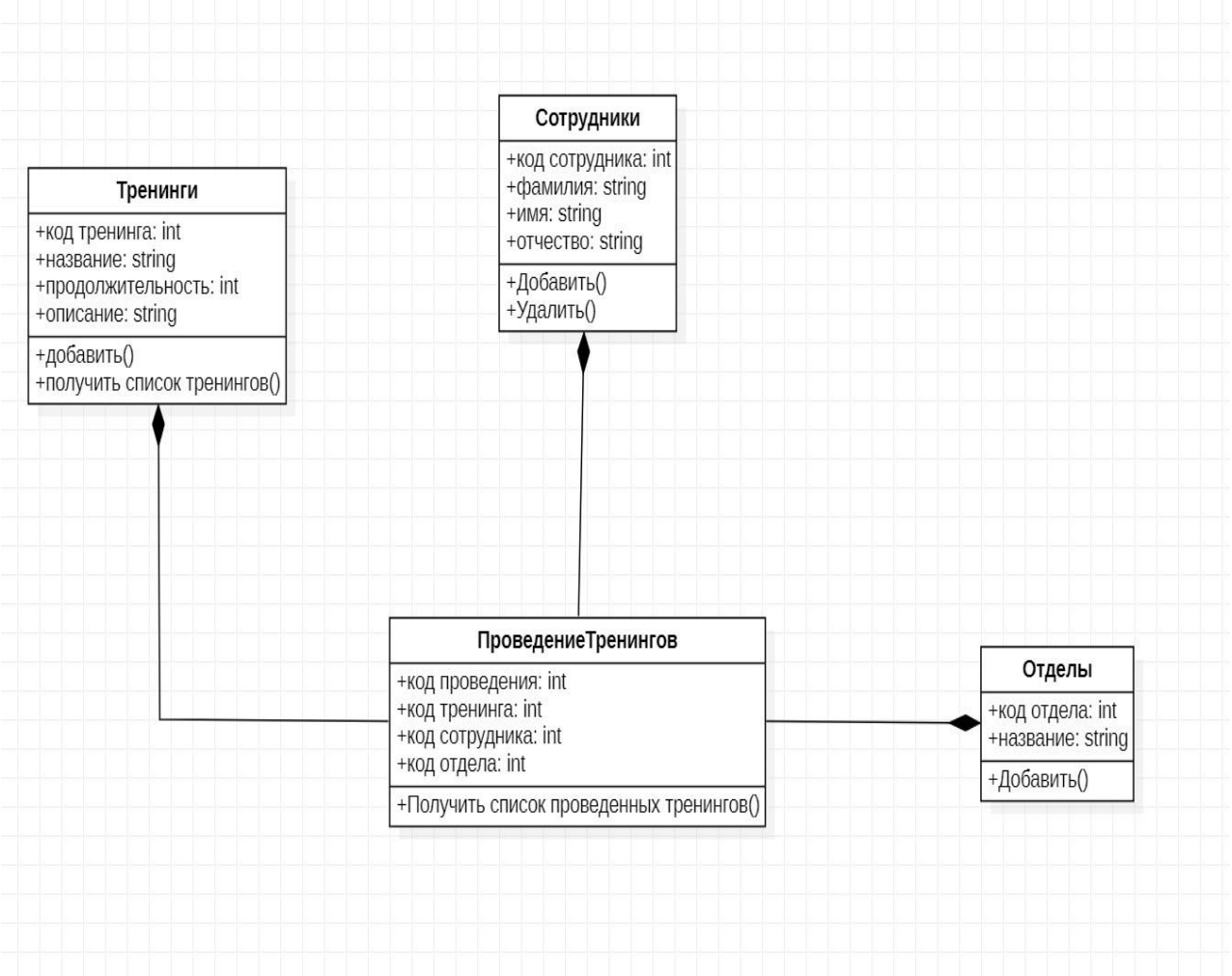

Рисунок 9 – Диаграмма классов системы учета тренингов

Благодаря построению и выделению основных объектов, которые потребуется создать при разработке сервиса, после чего можно спроектировать базу данных. База данных содержит информацию необходимую для работы сотрудников отдела тренингов.

Проектирование базы данных [11] затрагивает создание таблиц и атрибутов в базе.

2.3 Проектирование базы данных сервиса для учета тренингов

Вся информация необходимая для отдела проведения тренингов хранится в базе данных, построенной в СУБД MS SQL Server 2020.

Постоянный вход в базу данных и добавление там новых данные не потребуется благодаря использованию Entity Framework при реализации сервиса для учета тренингов.

Разработка базы данных применяется на основе Code First и Entity Framework.

«Entity Framework — это решение для работы с базами данных, которое используется в программировании на языках семейства .NET. Оно позволяет взаимодействовать с СУБД с помощью сущностей (entity), а не таблиц. Также код с использованием EF пишется гораздо быстрее» [12].

«Работая с базами данных напрямую, необходимо беспокоиться о подключении, подготовке SQL и параметров, отправке запросов и транзакций. На Entity Framework всё это делается автоматически — программист же работает непосредственно с сущностями и только говорит EF, что нужно сохранить изменения» [12].

«Entity Framework позволяет значительно сократить код работы с базами данных. При этом он предоставляет большие возможности. Можно использовать:

– foreign keys;

– связи one-to-one, one-to-many и many-to-many;

– параметризацию запросов;

– хранимые процедуры» [12].

На рисунке 10 показана диаграмма модели базы данных, созданная на уровне ER.

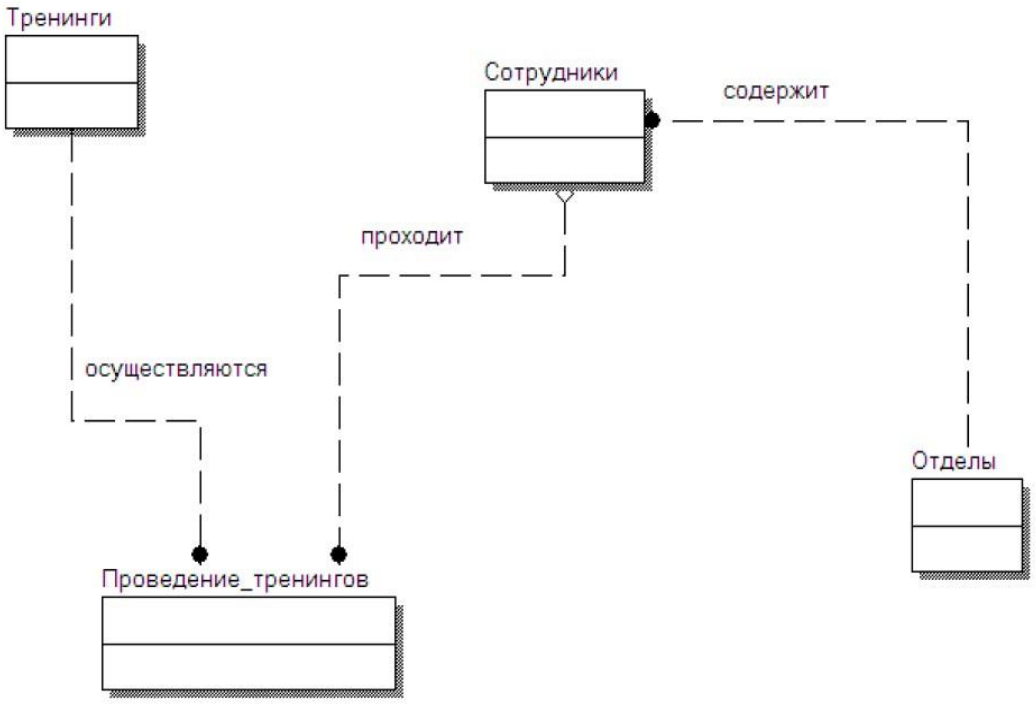

Рисунок 10 – Диаграмма ER-уровня модели данных

Дальнейшее проектирование базы данных подразумевает выделение в каждой таблице первичного ключа, который является основным ключом таблицы.

Каждая сущность должна иметь помимо первичного ключа необходимые атрибуты, определённые типом данных, сервис, который проверяет при вводе пользователем данные.

Для определения связи между таблицами базы данных выделены также и внешние ключи, которые и обеспечат связь и передачу данных между таблицами. На основе этого построим модель данных на уровне FA, рисунок 11.

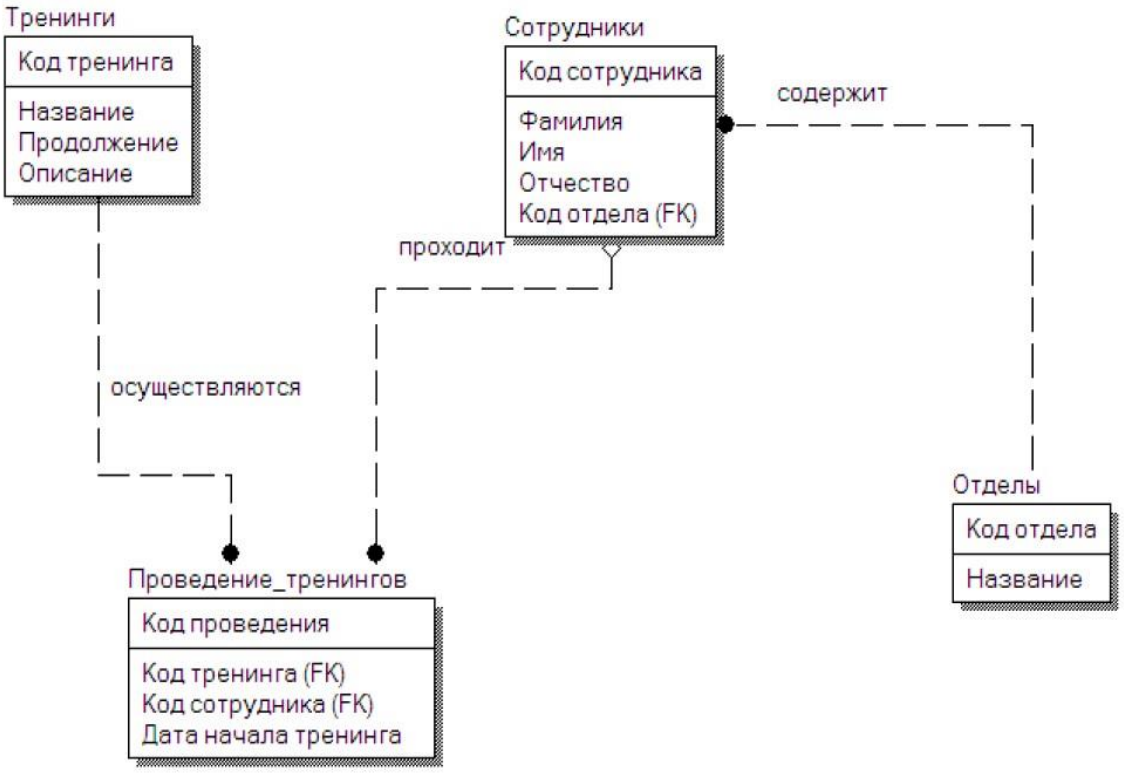

Рисунок 11 – Диаграмма FA-уровня

Все атрибуты и типы данных, которые были смоделированы показаны на рисунке 11 и в таблице 2.

Таблица 2 – Таблица, описывающая атрибуты сущностей

Имя	Смысл	Тип	Сущность	Ограничение
Код отдела	Идентификационный номер	Числовой	Отделы	Число: 1,2,3…
Название отдела	Название отдела	Текст	Отделы	255 символов
Код сотрудника	Идентификационный номер	Числовой	Сотрудники	Число: 1,2,3…

Продолжение таблицы 2

Имя	Смысл	Тип	Сущность	Ограничение
Фамилия	Фамилия сотрудника	Текст	Сотрудники	255 символов
Имя	Имя сотрудника	Текст	Сотрудники	255 символов
Отчество	Отчество сотрудника	Текст	Сотрудники	255 символов
Код отдела	Внешний ключ	Число	Сотрудники	Число: 1,2,3…
Код тренинга	Идентификационный номер	Текст	Тренинги	Число: 1,2,3…
Название	Название	Текст	Тренинги	255 символов
Продолжительность	Продолжительность	Число	Тренинги	Число: 1,2,3…
Описание	Описание	Текст	Тренинги	255 символов
Код проведения	Идентификационный номер	Число	Проведение тренингов	Число: 1,2,3…
Код тренинга	Внешний ключ	Число	Проведение тренингов	Число: 1,2,3…
Код сотрудника	Внешний ключ	Число	Проведение тренингов	Число: 1,2,3…
Дата начала тренинга	Дата начала тренинга	Дата	Проведение тренингов	Дата

На основе выделенных основных объектов базы данных, разработки атрибутов и типов данных, которые они содержат появилась возможность физически реализовать базу данных в выбранной СУБД.

2.4 Обоснование вида логической модели

Для начала работы с базой данных необходимо установить СУБД MS SQL Server [20], которая в компании работае по средствам локальной сети.

На рисунке 12 показан обозреватель объектов в СУБД, разработчик и пользователь сервиса сможет ориентироваться на основе этих объектов.

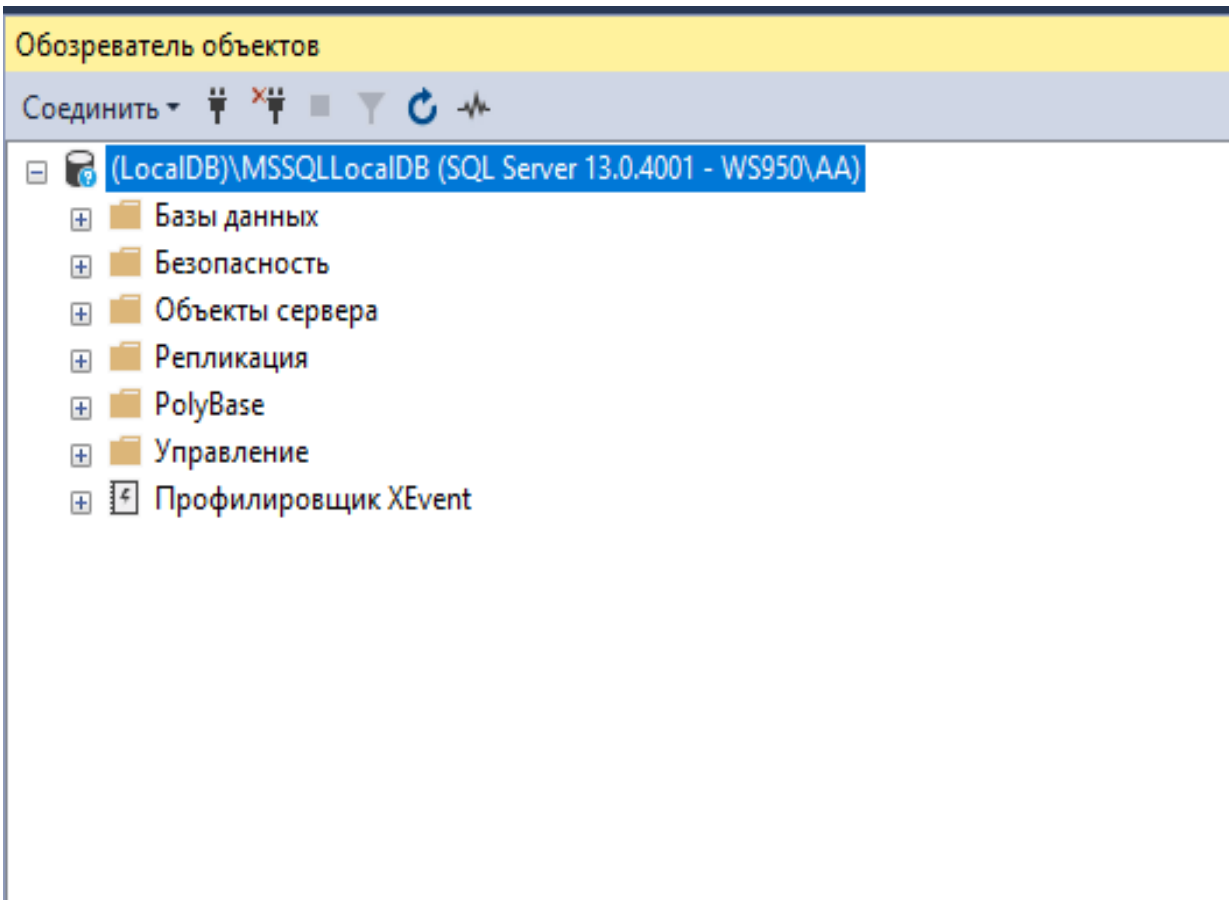

Рисунок 12 – Обозреватель объектов в СУБД

При нажатии на папку с названием «Базы данных» откроется список таблиц, которые хранятся в сервисе. При необходимости администратор вносит новые данные в сервис.

После создания базы данных для сервиса учета тренингов, потребуется также создание всех таблиц, которые были спроектированы ранее, учитывая первичные ключи и атрибуты таблиц.

Подключение базы данных к серверу осуществляет администратор сервиса

2.5 Требования к аппаратно-программному обеспечению сервиса

Разработка сервиса для учета тренингов в компании осуществлена на основе выбранной гибкой методологии и методов объектно-ориентированного программирования.

ООП позволяет создать более гибкую программу, которая удобна в использовании и легка при расширении функционала.

Объекты с одинаковыми свойствами и поведением объединяются в классы. Программа на объектно-ориентированном языке представляет собой совокупность описаний классов.

Работа пользователя в сервисе потребует наличия интерфейса пользователя, который способствует упрощению получению информации. В совокупности с интерфейсом пользователя сервис содержит и базу данных для хранения всех данных необходимых для процесса учета тренингов.

Рабочее место сотрудника отдела тренингов должно быть оборудовано для удобной работы. На каждом месте должен быть персональный компьютер, оснащенный необходимыми комплектующими и присоединен в общую локальную сеть отдела.

Выводы по главе 2

Проведенное проектирование процесса учета тренингов в компании с помощью диаграмм вариантов использования и последовательности позволит разработать качественный сервис, который позволит сотрудникам отдела тренингов выполнять учет проведения тренингов в компании.

Глава 3 Проектирование сервиса представления данных для учета тренингов

3.1 Архитектура сервиса представления данных для учета тренингов

Получение данных пользователем сервиса осуществляется путем запроса и получения ответа. Такой способ подразумевает с одной стороны сервера, а с другой клиента, которые взаимодействуют между собой посредствам запросов Http.

Для такого взаимодействия сервис должен быть разработан на основе клиент-серверной архитектуры.

Архитектура имеет пользовательский интерфейс, сотрудник сможет взаимодействовать с сервисов именно через него. Также сервис имеет контроллер и бизнес-логику, где обрабатываются данные, поступающие в сервис через пользовательский интерфейс.

СУБД осуществляет хранение данных, обработку их в системе и передачу данных пользователю через контроллеры по запросу [11].

3.2 Выбор инструментов разработки

Сервис реализован с помощью языка программирования высокого уровня, таким был выбран язык C#.

«C# — это язык с C-подобным синтаксисом. Здесь он близок в этом отношении к C++ и Java.

Являясь объектно-ориентированным языком, он много перенял у Java и C++. Как и Java, C# изначально предназначался для веб-разработки, и примерно 75% его синтаксических возможностей такие же, как у Java. Объектно-ориентированный подход позволяет строить с помощью C#

крупные, но в то же время гибкие, масштабируемые и расширяемые приложения» [13].

«C# уже давно поддерживает много полезных функций:

– инкапсуляция,

– наследование,

– полиморфизм,

– перегрузка операторов,

– статическая типизация.

При этом он всё ещё активно развивается, и с каждой новой версией появляется всё больше интересного — например лямбды, динамическое связывание, асинхронные методы и т.д.

Первая версия языка вышла вместе с релизом Microsoft Visual Studio .NET в феврале 2002 года» [13].

«Под C#, нередко имеют в виду технологии платформы .NET (Windows Forms, WPF, ASP.NET, Xamarin)» [13].

«В основе .NET — общеязыковая среда исполнения Common Language Runtime (CLR), благодаря чему платформа поддерживает несколько языков: наряду с C# это VB.NET, C++, F#, а также различные диалекты других языков, привязанные к .NET, например, Delphi.NET. Код на любом из этих языков компилируется в сборку на общем языке CIL (Common Intermediate Language) — своего рода ассемблер платформы .NET.

.NET представляет единую для всех поддерживаемых языков библиотеку классов.

Общеязыковая среда исполнения CLR и базовая библиотека классов — это основа для целого стека технологий, которые разработчики могут задействовать при создании разных приложений» [13].

«Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы,

итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML. Переняв многое от своих предшественников — языков C++, Java, Delphi, Модула и Smalltalk — C#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем: так, C# не поддерживает множественное наследование классов (в отличие от C++)» [13].

«C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем. (Однако эта закономерность была нарушена с выходом C# 3.0, представляющим собой расширения языка, не опирающиеся на расширения платформы .NET.) CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др.» [14].

Сравнение языков программирования представлено в таблице 3.

Таблица 3 – Сравнение языков программирования

Критерий сравнения	C++	C#	Java
Наличие фреймворков и библиотек	+	+	+
Строгая типизация данных	-	+	+

Продолжение таблицы 3

Критерий сравнения	C++	C#	Java
Постоянное развитие	-	+	-
Базовая библиотека классов	-	+	+
Популярность	-	+	-
Сложность изучения	+	-	-

Microsoft Visual Studio 2022 – представляет собой набор интегрированных сред разработки, он, как и многие был разработан программистами Microsoft как бесплатная и ограниченная по функциям версия несвободной Microsoft Visual Studio [18].

Visual Studio достаточно многофункциональная, позволяет использовать множество качественных плагинов, что позволяет в будущем расширять функциональность приложения и подключить другие языки [19]. C# вместе с Visual Studio обеспечит разработку эффективного сервиса для представления данных для учета тренингов.

3.4 Разработка физической модели данных

Описание фреймворка [17]. для создания таблиц в системе управления базой данных представлено выше. Код для создания таблиц базы данных показан на рисунке 13.

```
namespace EducationCenter.Model
{
    using System;
    using System.Data.Entity;
    using System.Linq;

    public class Centre : DbContext
    {
        public Centre()
            : base("name=Centre")
        {
        }

        public virtual DbSet<Course> Courses { get; set; }
        public virtual DbSet<CompanyEmployee> CompanyEmployees
{ get; set; }
        public virtual DbSet<Education> Educations { get; set; }
    }


}
```

Рисунок 13 – Код, создаваемый базы данных

Наполнение таблиц данными с помощью кода в программе показано на рисунке 14. Таблица тренинги создана с типами данных и методами.

```
namespace EducationCenter.Model
{
    public class Course
    {
        [Required]
        public int Id { get; set; }

        [Required]
        public string Name { get; set; }

        [Required]
        public string Number { get; set; }

        [Required]
        public int Duration { get; set; }

        [Required]
        public string Purpose { get; set; }

        [Required]
        public string Content { get; set; }
    }
}
```

Рисунок 14 – Код для создания таблицы Тренинги

Таблица Сотрудники созданная с помощью кода и фреймворка показана на рисунке 15.

```
namespace EducationCenter.Model
{
    public class CompanyEmployee
    {
        [Required]
        public int Id { get; set; }

        [Required]
        public string FirstName { get; set; }

        [Required]
        public string Surname { get; set; }

        [Required]
        public int PositionId { get; set; }

        [Required]
        public int DepartmentId { get; set; }

        [Required]
        public DateTime EmploymentDate { get; set; }

    }
}
```

Рисунок 15 – Таблица Сотрудники

Таблица с данными проведения тренингов показана на рисунке 16.

```
namespace EducationCenter.Model
{
    public class Education
    {
        [Required]
        public int Id { get; set; }

        public int? CompanyEmployeeId { get; set; }

        //public virtual CompanyEmployee CompanyEmployee { get;
set; }
        public CompanyEmployee CompanyEmployee { get; set; }

        public int? CourseId { get; set; }

        //public virtual Course Course { get; set; }
        public Course Course { get; set; }

        [Required]
        public DateTime Date { get; set; }

    }
}
```

Рисунок 16 – Таблица Проведение тренингов

На рисунке 17 Показана физическая модель данных

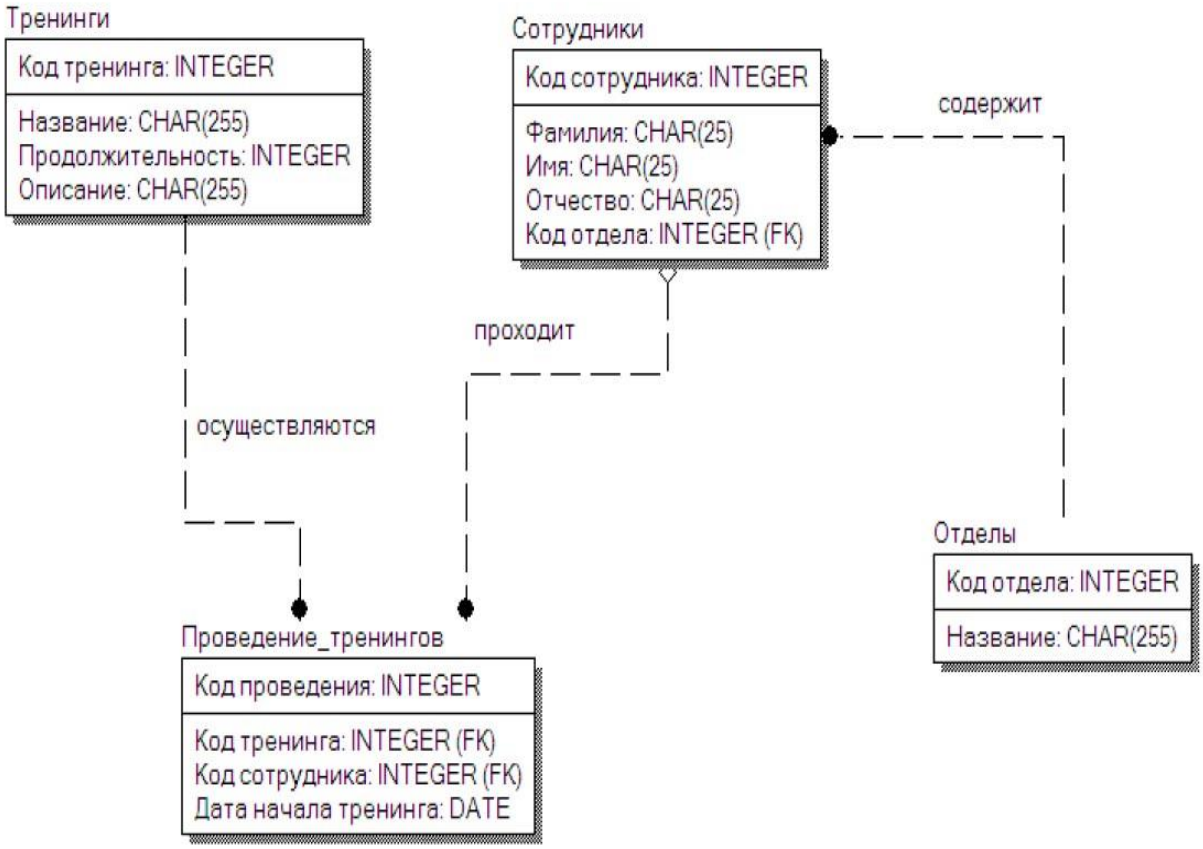

Рисунок 17 – Физическая модель данных

Установка базы данных на сервере обеспечит доступ к данным и обработку их при получении. Подключение базы данных осуществляет администратор.

3.5 Разработка сервиса представления данных для учета тренингов

Запуск приложения не требует авторизации, чтобы начать работу необходимо установить установочные файлы на свой персональный

компьютер. На рисунке 18 показан интерфейс пользователя (сотрудника отдела тренингов).

Программный код методов системы приведен в Приложении А.

Рисунок 18 – Главное окно пользователя системы

На главном меню расположены кнопки:

– список всех тренингов;

– список сотрудников, учащиеся или должны учиться в этом месяце;

– сотрудники, которые должны пройти обучение в этом году;

– тренинги, которые прошли сотрудники.

Каждая кнопка открывает списки с информацией. Под кнопкой «Список всех тренингов» получена информация, показанная на рисунке 19.

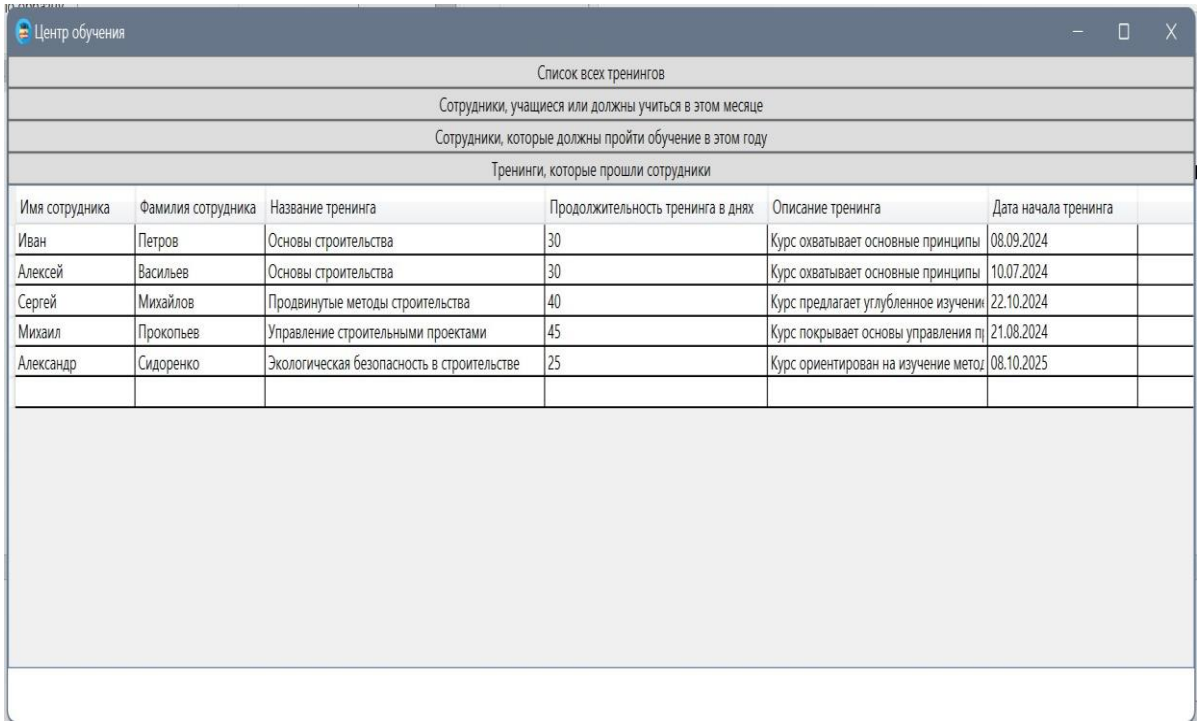

Рисунок 19 – Список всех тренингов

Открытие списка сотрудников учащихся или которые должны пройти обучение в этом месяце открывается со следующей кнопки, рисунок 20.

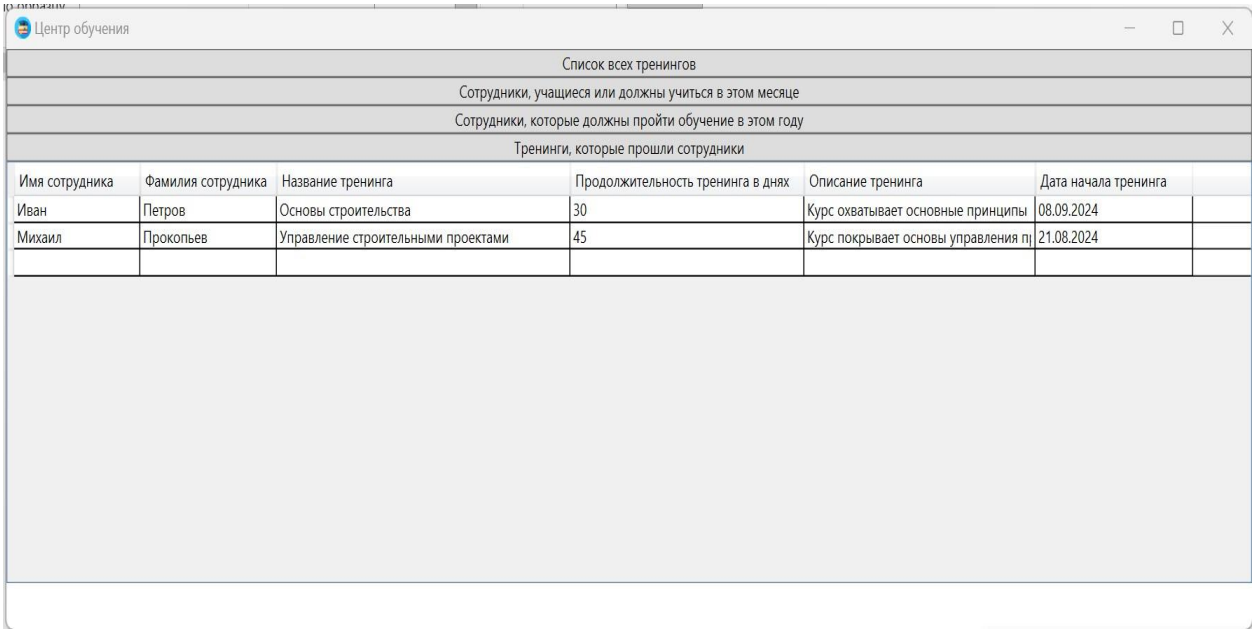

Рисунок 20 – Список сотрудников учащихся на тренингах

При нажатии на кнопку «Сотрудники, которые должны пройти обучение в этом году» откроются данные о сотрудниках, которые пройдут их в текущем году, рисунок 21.

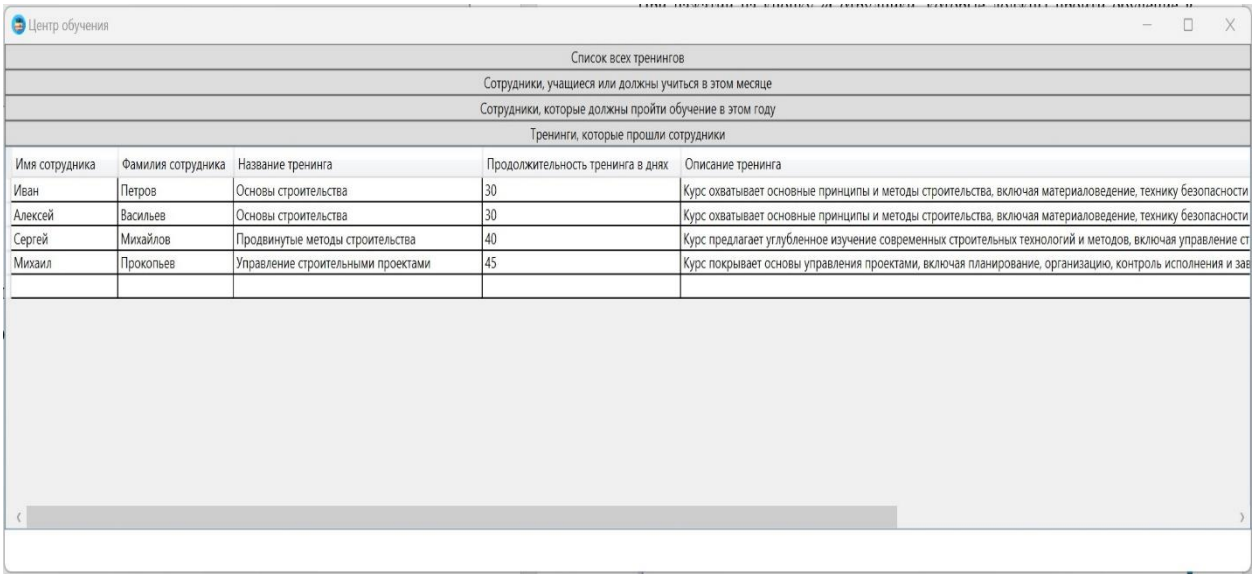

Рисунок 21 – Сотрудники, которые должны пройти обучение в этом году

При нажатии кнопки «Тренинги, которые прошли сотрудники» откроет список сотрудников, рисунок 22.

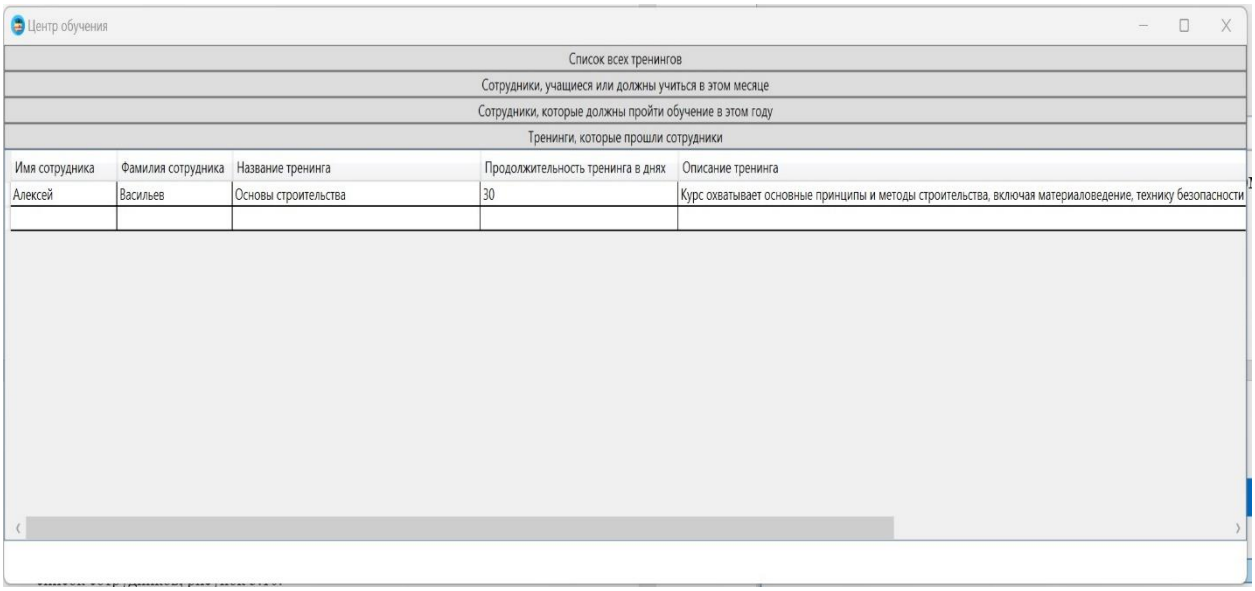

Рисунок 22 – Тренинги, которые прошли сотрудники

С помощью данных форм и получения информации происходит учет тренингов, которые были проведены в компании, а также учет сотрудников, которые прошли тренинги или те, которые должны пройти тренинги в текущем месяце или в текущем году.

3.6 Расчет экономической эффективности проекта

В экономической эффективности можно выделить два типа: абсолютную и относительную. Абсолютная экономическая эффективность оценивается на основе анализа конкретного варианта автоматизации, без учета возможных альтернатив. В случае относительной экономической эффективности проводится сравнение различных стратегий автоматизации с точки зрения их экономической эффективности.

Для разработки ПО требуются дополнительные текущие эксплуатационные расходы.

Капитальные затраты включают в себя расходы на покупку, доставку и установку технических средств автоматизации, строительство дополнительных производственных помещений для размещения программного обеспечения, а также затраты на материалы, использованные при прокладке сетей и коммуникаций.

Практика показывает, что инвестиции в проекты начинают приносить доход только спустя некоторый временной период, который требуется для обучения сотрудников и адаптации системы к определенным условиям.

Затраты на реализацию и внедрение рассчитываются путем сложения всех затрат:

$$C = C_{\text{ктс}} + \text{С}_{\text{алг}} + \text{С}_{\text{отл}} + \text{С}_{\text{вн}}, \quad (1)$$

Затраты включают такие показатели как:

– приобретенное оборудование;

– написание кода программного продукта;

– тестирование сервиса;

– внедрение сервиса.

$C_{ктс}$ = 144000 рублей

$C_{алг}$ = 81980 рублей

$C_{вн}$ = 324600 рублей

$C_{отл}$ = 712200 рублей

С = 144000+81980+324600+712200 =1 270 000 рублей

Расчет расходов на написание алгоритма:

$$С_{алг} = Зп_{разр} * В + От, \quad (2)$$

$Зп_{разр}$ – оклад разработчика;

$В$ – время, затраченное на разработку;

$От$ – отчисления на соц. страхование.

$Зп_{разр}$ = 100000 рублей,

$В$ = 15 дней,

$От$ = 21000 рублей,

$С_{алг}$ = 100000/22*15+21000=81980 рублей.

Расходы на приобретение оборудования:

$$С_{ктс} = С_{комп} + С_{прин} + С_{др}, \quad (3)$$

Затраты на внедрение.

$$С_{вн} = К_{вн} + З_{вн} \quad (4)$$

В затраты включена стоимость:

– время на внедрение сервиса;

– зарплата специалистов внедрения.

$К_{вн}$ = 211200 рублей

$З_{п.вн.}$ = 113400 рублей

$С_{вн}$ = 211200 + 113400

$С_{вн}$ = 324600 рублей

Стоимость машинного времени на время внедрения определяется по формуле:

$$К_{вн} = К * d * q, \quad (5)$$

К - время работы на ПК в день;

d – число дней работы на ПК;

q - стоимость часа машинного времени.

К = 8 часов,

d = 22 дня,

q = 1200 рублей,

$К_{вн}$ = 8*22*1200=211200 рублей.

Расчет зарплаты рассчитывается по формуле:

$$З_{вн} = Q_{вн} * \left(\frac{d}{Д}\right) + отч, \quad (6)$$

$Q_{вн.}$ - оклад разработчика;

d – число дней работы на ПК;

Д – число рабочих дней в месяц;

отч. – отчисления на соц. страхование (26% от оклада разработчика системы).

$Q_{вн.}$ = 90000 рублей,

d = 22 дня,

Д = 22 дня,

отч = 23400 рублей,

$З_{п.вн.}$ = 90000*(22/22) +23400 = 113400 рублей.

Затраты на создание кода рассчитывается по формуле:

$$С_{отл} = К_{отл} + К_{зп}, \quad (7)$$

$К_{отл}$ – время на отладку;

$К_{з.п.}$ – зарплата программиста.

$К_{отл}$ = 211200 рублей,

$К_{з.п.}$ = 501100 рублей,

$С_{отл}$ = 211200+501100=712200 рублей.

Рассчитать зарплату разработчика можно по формуле:

$$К_{зп} = К_{м} * О_{раз} + отч., \quad (8)$$

$К_{м}$ = 100 дней,

$О_{разраб.}$ = 5000 рублей,

отч. = 1050 рублей,

$К_{з.п.}$ = 100 * 5000 + 1050=501050 рублей.

Рассчитаем по формуле годовые расходы:

$$dE_{экспл} = (1 + W_o) * (1 + W_q) * B_{зпр} - E_{экспл}, \quad (9)$$

Wo = 0.35,

Wq = 0.15,

$В_{з.пр}$ = 4 * 12 * 30000 = 1440 000 рублей,

$Е_{экспл}$ = 288 000 рублей,

$dЕ_{экспл}$ = (1+0,35) * (1+0,15) *1 440 000 – 288 000 = 1 948 000 рублей.

Расчет затрат на разработку рассчитаем по формуле:

$$S = \frac{C}{N}, \tag{10}$$

C = 1270000 рублей,

N = 2,

S = 635 000 рублей.

Окупаемость внедрения сервиса для учета тренингов рассчитаем по формуле:

$$O = \frac{C}{dE_{экспл}} \ (лет), \tag{12}$$

C =1270000 рублей,

$dE_{экспл}$ = 1948000 рублей,

T = 1270000/1948000 = 0,619 лет (7,8 месяцев).

В процессе расчета подсчитана экономическая эффективность от внедрения программного обеспечения учета тренингов в компании. Кроме того, с помощью расчетов доказана эффективность применения этого комплекса программных средств.

На рисунке 23 указан график окупаемости сервиса представления данных для учета тренингов.

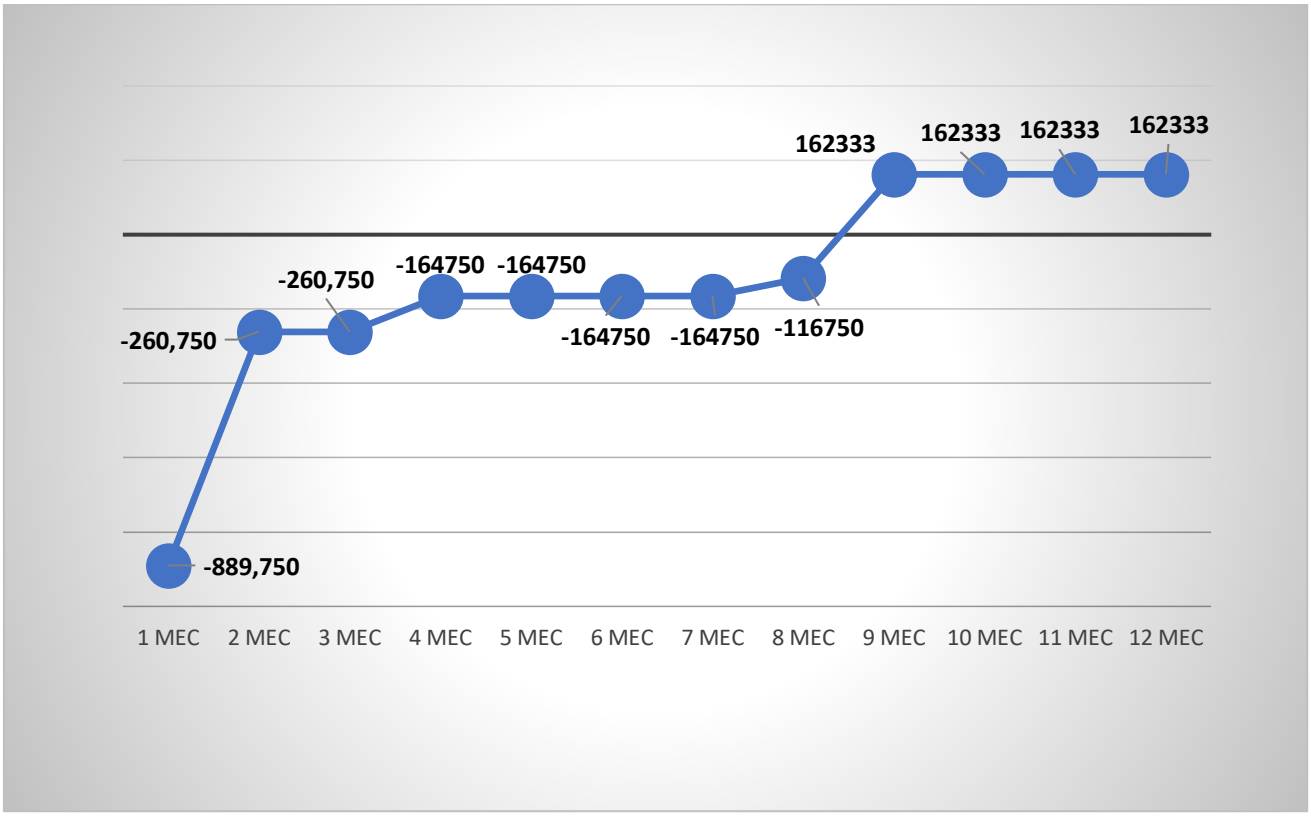

Рисунок 23 – График окупаемости программного продукта

Из расчета видно, что срок окупаемости сервиса составит 7,8 месяцев. Таким образом, внедрение сервиса представления данных для учета тренингов является выгодным.

Выводы по третьей главе

Третья глава включает выполнения проектирования сервиса представления данных для учета тренингов в компании. Проектирование системы затронуло выбор необходимых инструментов для реализации сервиса, которая помогает вести учет тренингов, проводимых в компании. Также были разработаны основные формы, которые помогают пользователю системы получать информацию о проводимых тренингах, сотрудниках, прошедших тренинги и те, которые только должны пройти эти тренинги. Завершающим шагом был проведен расчет экономической эффективности проекта.

Заключение

В результате выполнения выпускной квалификационной работы были выполнены задачи, которые помогли достичь поставленной цели.

В работе был проведен глубокий обзор предметной области и индивидуального предприятия «AJAYYP SOWDA». На основе обзора были выделены слабые места рассматриваемого процесса учета тренингов в компании и сформированы функциональные требования к системе для дальнейшей разработки.

Также был проведен обзор аналогичных систем, которые представлены на российском рынке в готовом виде и выделены плюсы и минусы при внедрении таких систем, что помогло обосновать собственную разработку данной системы.

С помощью сравнительного анализа был выбраны ЯП C#, СУБД MSSQL Server и IDE Visual Studio 2022, выбранные инструменты реализованы при разработке сервиса представления данных, также разработаны все необходимые модели, которые показывают архитектуру сервиса и взаимосвязь пользователей с системой, построены таблицы для правильной реализации системы для отдела тренингов.

Результатом работы является разработанный сервис, автоматизирующий процесс представления данных для учета тренингов в индивидуальном предприятии «AJAYYP SOWDA».

Разработанный сервис можно применять в учебном центре индивидуального предприятия «AJAYYP SOWDA», где осуществляется обучение сотрудников путем проведения тренингов.

Завершающим этапом был проведен расчет экономической эффективности проекта разработки и внедрения сервиса представления данных для учета тренингов.

Список используемой литературы и источников

1. Волк В. К. Базы данных. Проектирование, программирование, управление и администрирование: учебник /В. К. Волк. — Санкт-Петербург: Лань, 2020. — 244 с.

2. Гибкие методологии. [Электронный ресурс]. - Режим доступа: https://www.e-xecutive.ru/wiki/index.php/Agile._Гибкие_методологии (дата обращения: 04.09.2024).

3. Канбан. [Электронный ресурс]. – URL: https://vc.ru/u/752307-karina-gorbunova/218436-kanban-agile-scrum-lean-gibkie-metodologii-razrabotki (дата обращения: 04.09.2024).

4. Онлайн курс обучения программированию: методологии разработки. [Электронный ресурс]. – URL: https://javarush.ru/groups/posts/647-metodologii-razrabotki-po (дата обращения: 04.09.2024).

5. Отличия, достоинства и недостатки базы данных PostgreSQL: что такое PostgreSQL. [Электронный ресурс]. – URL: https://oracle-patches.com/common/3214-что-такое-postgresql (дата обращения: 04.09.2024).

6. 1С: Управление учебным центром. [Электронный ресурс]. – URL: https://www.connect-wit.ru/produkt-1s-upravlenie-uchebnym-tsentrom-red-2-firmy-onlajn-konsalting-poluchil-ocherednoj-sertifikat-sovmestimo-sistema-programm-1s-predpriyatie.html (дата обращения: 04.09.2024).

8. C GUI на C#: Кроссплатформенное приложение [Электронный ресурс]. - URL: https://skillbox.ru/media/code/ne_windows_(дата обращения: 04.09.2024).

9. Тамре Л. Введение в тестирование программного обеспечения /пер. с англ. М.: Вильямс, 2018. 368 с.

10. Что такое UML. [Электронный ресурс]. – URL: https://medium.com/@medvedev.rm/что-такое-uml-и-для-чего-он-нам-нужен-7d3e8a372b15 (дата обращения: 04.09.2024).

11. Access: База данных. [Электронный ресурс]. – URL: https://www.microsoft.com/ru-ru/microsoft-365/access(дата обращения: 04.09.2024).

12. Entity Framework: как быстрее написать код для работы с базой данных. [Электронный ресурс]. - URL: https://skillbox.ru/media/code/entity_framework/ (дата обращения: 04.09.2024).

13. GeekBrains – Язык программирования C#. [Электронный ресурс]. - URL: https://geekbrains.ru/posts/yazyk-programmirovaniya-c-sharp-istoriya-specifika-mesto-na-rynke (дата обращения: 04.09.2024).

13. Helpiks.org: Достоинства и недостатки языка [Электронный ресурс]. - URL: https://helpiks.org/6-21879.html (дата обращения: 04.09.2024).

14. Helpiks.org: Достоинства и недостатки языка [Электронный ресурс]. - URL: https://helpiks.org/6-21879.html (дата обращения: 04.09.2024).

15. Scrum. [Электронный ресурс]. – URL: https://habr.com/ru/companies/it-guild/articles/341924/ (дата обращения: 04.09.2024).

16. SQL Server: руководство по SQL Server. [Электронный ресурс]. – URL: https://docs.microsoft.com/ru-ru/sql/sql-server/tutorials-for-sql-server-2016?view=sql-server-ver15 (дата обращения: 04.09.2024).

17. SQL Server: программное обеспечение. [Электронный ресурс]. – URL: https://www.microsoft.com/ru-ru/sql-server/sql-server-downloads (дата обращения: 04.09.2024).

18. C#. [Электронный ресурс]. – URL: https://dotnet.microsoft.com/en-us/languages/csharp (дата обращения: 04.09.2024).

19. The C Sharp (C#) Beginner’s Guide [Электронный ресурс]. – URL: https://medium.com/c-sharp-language/the-c-beginners-guide-6a14af03ed85 (дата обращения: 04.09.2024).

20. Visual Studio 2022: программное обеспечение. [Электронный ресурс]. – URL: https://visualstudio.microsoft.com/ru/vs/ (дата обращения: 04.09.2024)

21. Visual Studio 2022. [Электронный ресурс]. – URL: https://www.techspot.com/downloads/7493-visual-studio-2022.html (дата обращения: 04.09.2024).

22. Visual Studio tutorials | C#. [Электронный ресурс]. – URL: https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/?view=vs-2022l (дата обращения: 04.09.2024).

23. Choy, K. L. Development of an intelligent customer-supplier relationship management system: the application of case-based reasoning / K. L. Choy, Kenny K H Fan, Victor Lo // Industrial Management & Data Systems. – 2003. – Vol. 103, No. 4. – P. 263-274.

24. Law, Y. F. D. An Integrated Case-Based Reasoning Approach for Intelligent Help Desk Fault Management / Y. F. D. Law, B. F. Sew, S. E. J. Kwan // Expert Systems with Applications. – 1997. – Vol. 13, No. 4. – P. 265-274.

25. Twidale, M. B. Browsing is a collaborative process / M. B. Twidale, D. M. Nichols, C. D. Paice // Information Processing & Management. – 1997. – Vol. 33, No. 6. – P. 761-783.

26. Mark, G. Conventions and Commitments in Distributed CSCW Groups / G. Mark // Computer Supported Cooperative Work. – 2002. – Vol. 11, No. 3-4. – P. 349-387.

27. Simazu, H. ExpertGuide: A Conversational Case-Based Reasoning Tool for Developing Mentors in Knowledge Spaces / H. Simazu, A. Shibata, K. Nihei // Applied Intelligence. – 2001. – Vol. 14, No. 1. – P. 33-48.

Приложение А

Программный код

```
namespace EducationCenter
{
    public partial class MainWindow : Window
    {
        private const int MaxEducationDaysInPast = -99;
        private const int MaxEducationDaysInFuture = 100;
        private static readonly Random _random = new Random();

        public MainWindow()
        {
            InitializeComponent();
            _ = InitializeDatabaseAndUpdateUI();
        }
        private static void CheckAndInitializeDatabase()
        {
            if (IsDatabaseInitialized())
                return;

            InitializeDatabase();
        }
        private static void InitializeDatabase()
        {
            var courses = GetCourses();

            var companyEmployees = GetCompanyEmployees();
```

```
            var educations = GetEducations(courses, companyEmployees,
_random);

        return new List<Education>()
                {
                    //Обучение, сотрудник, которого учится в этом месяце
                    new Education()
                    {
                        Course = courses.ElementAt(0),
                        CompanyEmployee = companyEmployees.ElementAt(0),
                        Date = DateTime.Now,
                    },

                    //Обучение, сотрудник, которого уже закончил обучение
                    new Education()
                    {
                        Course = courses.ElementAt(0),
                        CompanyEmployee = companyEmployees.ElementAt(1),
                        Date = DateTime.Now.AddDays(-
    (courses.ElementAt(0).Duration * 2)),
                    },
                    new Education()
                    {
                        Course = courses.ElementAt(2),
                        CompanyEmployee = companyEmployees.ElementAt(2),
```

```
        Date =
DateTime.Now.AddDays(random.Next(MaxEducationDaysInPast,
MaxEducationDaysInFuture)),
                    },
                    new Education()
        {
                        Course = courses.ElementAt(3),
                        CompanyEmployee = companyEmployees.ElementAt(3),
                        Date =
DateTime.Now.AddDays(random.Next(MaxEducationDaysInPast,
MaxEducationDaysInFuture)),
                    },
                     //Обучение, запланированного на следующй год
                    new Education()
                    {
                        Course = courses.ElementAt(4),
                        CompanyEmployee = companyEmployees.ElementAt(4),
                        Date =
DateTime.Now.AddDays(random.Next(MaxEducationDaysInPast,
MaxEducationDaysInFuture)).AddYears(1),
                    }
        };
        }
        private static IEnumerable<CompanyEmployee>
GetCompanyEmployees()
        {
        return new List<CompanyEmployee>()
```

```
{
new CompanyEmployee()
{
FirstName = "Иван",
Surname = "Петров",
PositionId = 1,
DepartmentId = 1,
EmploymentDate = new DateTime(2021, 08, 21)
},
            new CompanyEmployee()
            {
                FirstName = "Алексей",
                Surname = "Васильев",
                PositionId = 2,
                DepartmentId = 1,
                EmploymentDate = new DateTime(2021, 08, 22)
            },
            new CompanyEmployee()
            {
                FirstName = "Сергей",
                Surname = "Михайлов",
                PositionId = 2,
                DepartmentId = 2,
                EmploymentDate = new DateTime(2021, 07, 22)
            },
            new CompanyEmployee()
            {
```

```
                FirstName = "Михаил",
                Surname = "Прокопьев",
                PositionId = 2,
                DepartmentId = 1,

EmploymentDate = new DateTime(2021, 07, 23)
            },

            new CompanyEmployee()
            {
                FirstName = "Александр",
                Surname = "Сидоренко",
                PositionId = 3,
                DepartmentId = 2,
                EmploymentDate = new DateTime(2021, 04, 16)
            }
        };
    }
    private static IEnumerable<Course> GetCourses()
    {
        return new List<Course>()
        {
            new Course()
            {
                Name = "Основы строительства",
                Number = "101/C",
                Purpose = "Обучение основам строительства",
```

```
                Duration = 30,
                Content = "Курс охватывает основные принципы и
методы строительства, включая материаловедение, технику
безопасности и использование строительной техники."
            },
        new Course()
            {
                Name = "Техника безопасности на строительной
площадке",
                Number = "202/S",
                Purpose = "Обучение правилам безопасности",
                Duration = 15,
                Content = "Курс направлен на обучение сотрудников
правилам техники безопасности, необходимых для предотвращения
несчастных случаев на строительной площадке."
            },
            new Course()
            {
                Name = "Продвинутые методы строительства",
                Number = "303/AC",
                Purpose = "Специализированное обучение",
                Duration = 40,
                Content = "Курс предлагает углубленное изучение
современных строительных технологий и методов, включая управление
строительными проектами и инновационные строительные
материалы."
            },
```

```
                new Course()
                    {
                        Name = "Управление строительными проектами",
                        Number = "404/PM",
                Purpose = "Повышение квалификации в управлении проектами",
                        Duration = 45,
                        Content = "Курс покрывает основы управления проектами, включая планирование, организацию, контроль исполнения и завершение строительных проектов."
                    },
                    new Course()
                    {
                        Name = "Экологическая безопасность в строительстве",
                        Number = "505/ES",
                        Purpose = "Обучение экологической безопасности",
                        Duration = 25,
                        Content = "Курс ориентирован на изучение методов минимизации воздействия строительства на окружающую среду и выполнения работ в соответствии с экологическими стандартами."
                    }
                };
            }

            private void AllEducationsButton_Click(object sender, RoutedEventArgs e)
            {
```

```
        Task.Run(async () =>
                {
                    var educations = await GetAllEducationsFromDatabase();
                    Dispatcher.Invoke(() =>
                    {
                        CoursesGrid.ItemsSource = educations;
                    });
                });
            }
            private void
EducationsWhenEmployeesTrainingThisMonthButton_Click(object sender,
RoutedEventArgs e)
            {
                Task.Run(async () =>
                {
                    var educations = await
GetEducationsWhenEmployeesTrainingThisMonth(DateTime.Now);
                    Dispatcher.Invoke(() =>
                    {
                        CoursesGrid.ItemsSource = educations;
                    });
                });
            }
        private void
EducationsWhenEmployeesToBeTrainedAtCurrentYearButton_Click(object
sender, RoutedEventArgs e)
            {
```

```
            Task.Run(async () =>
            {
        var educations = await
GetEducationsWhenEmployeesToBeTrainedAtCurrentYear(DateTime.Now
);
                Dispatcher.Invoke(() =>
                {
                    CoursesGrid.ItemsSource = educations;
                });
            });
        }
        private void
CompletedEducationsByEmployeesButton_Click(object sender,
RoutedEventArgs e)
        {
            Task.Run(async () =>
            {
                var educations = await
GetCompletedEducationsByEmployees(DateTime.Now);
                Dispatcher.Invoke(() =>
                {
                    CoursesGrid.ItemsSource = educations;
                });
            });
        }
                private async Task InitializeDatabaseAndUpdateUI()
        {
```

```
            await Task.Run(() =>
              {
        Dispatcher.Invoke(() =>
                  {
                     LoadingRingGrid.Visibility = Visibility.Visible;
                     MainContainer.Visibility = Visibility.Collapsed;
                  });
                  CheckAndInitializeDatabase();
                  Dispatcher.Invoke(() =>
                  {
                     LoadingRingGrid.Visibility = Visibility.Collapsed;
                     MainContainer.Visibility = Visibility.Visible;
                  });
               });
            }
                  private async Task<IEnumerable<Education>>
GetAllEducationsFromDatabase()
            {
               using (Centre db = new Centre())
               {
                  var educations = await db.Educations.SqlQuery("SELECT *
FROM Educations").ToListAsync();

                  foreach (var education in educations)
                  {
                     db.Entry(education).Reference("Course").Load();
```

```
            db.Entry(education).Reference("CompanyEmployee").Load();
                    }
        return educations;
                }
            }
                    private async Task<IEnumerable<Education>>
GetEducationsWhenEmployeesTrainingThisMonth(DateTime
currentDateTime)
            {
                using (Centre db = new Centre())
                {
                    // Определяем начало и конец текущего месяца
                    var firstDayOfMonth = new
DateTime(currentDateTime.Year, currentDateTime.Month, 1);
                    var lastDayOfMonth =
firstDayOfMonth.AddMonths(1).AddDays(-1);

                    // Запрос на выборку образований, учитывая начало и
продолжительность обучения
                    var educationsThisMonth = await db.Educations
                                                .Where(e => (e.Date >=
firstDayOfMonth && e.Date <= lastDayOfMonth) ||  // Обучение началось
в этом месяце
                                                       (e.Date < firstDayOfMonth &&
DbFunctions.AddDays(e.Date, e.Course.Duration) >= firstDayOfMonth)) //
Обучение продолжается в этот месяц
```

```
                    .Include(e => e.CompanyEmployee) // Подгрузка данных
        о сотрудниках
        .Include(e => e.Course)        // Подгрузка данных о курсах
        .ToListAsync();
                return educationsThisMonth;
            }
        }
                private async Task<IEnumerable<Education>>
GetEducationsWhenEmployeesToBeTrainedAtCurrentYear(DateTime
currenDateTime)
        {
            using (Centre db = new Centre())
            {
                var currentYear = currenDateTime.Year;
                var educations = await db.Educations
                                .Where(e => e.Date.Year == currentYear)
                                .Include(e => e.CompanyEmployee)
                                .Include(e => e.Course)
                                .ToListAsync();
                return educations;
            }
        }
    private async Task<IEnumerable<Education>>
GetCompletedEducationsByEmployees(DateTime currentDateTime)
        {
            using (Centre db = new Centre())
            {
```

```
            // Запрос на выборку образований, где дата завершения
        обучения ранее или равна текущей дате
        var educationsWithCompletedCourses = await db.Educations
                                                .Where(e =>
DbFunctions.AddDays(e.Date, e.Course.Duration) <= currentDateTime) //
Фильтрация записей, где обучение завершено
                                                .Include(e =>
e.CompanyEmployee)
        Include(e => e.Course)
        ToListAsync();

                return educationsWithCompletedCourses;
            }
        }
    }
}
```