МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра IT-инжиниринг

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой
PhD, доцент
_____ Т.С. Картбаев
«____» _____ 2019г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка информационной системы газового предприятия АО “Intergaz”
Специальность 5В070300 – «Информационные системы»

Выполнил Аслугужиева Г.Б. Группа ИС-15-2
Научный руководитель ст. преподаватель _____А.Т.Купарова
«23» 05 2019г.

Консультанты:
по экономической части: к.э.н., доцент _____А.И.Бекишева
«05» 05 2019г.

по безопасности жизнедеятельности: ст. преп., д.т.н. _____Ш.Ш.Бекбасаров
«14» 05 2019г.

по применению программного обеспечения: ст. преп. _____М.Н. Майкотов
«14» 05 2019г.

Нормоконтролер: ст. преп._____Ж.К. Алимсеитова
«22» 05 2019г.

Рецензент: к.т.н, ассоциированный профессор, заведующий кафедрой «Информационная безопасность»КАЗНиТУ им. Сатпаева_____ Н.А.Сейлова
«____» _____ 2019г.

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Институт систем управления и информационных технологий
Кафедра IT-инжиниринг
Специальность 5В070300 – «Информационные системы»

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Аслугужиевой Гульнаре Бегмановне

Тема работы: Разработка информационной системы газового предприятия АО "Intergaz"

Утверждена приказом по университету № 124 от «26» октября 2018 г.

Срок сдачи законченного проекта «__» мая 2019 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): исследование и анализ работы газового предприятия; изучение бизнес-процессов работы компании и структур подсистем предприятия; сбор и анализ используемых данных для заказчика (номер договора, дата подписания, информация о клиентах и рабочих компании и т.д.).

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:

а) Разработка требований к программному обеспечению;

б) Проектирование ИС ;

в) Разработка программного продукта на базе Visual Studio (база данных, C#) ;

г) Реализация и аттестация ИС;

д) Приложение А. Листинг программы.

Перечень графического материала (с точным указанием обязательных чертежей):

Представлены 16 таблиц, 31 иллюстраций.

Основная рекомендуемая литература:

1) Visual Studio 2019. Обзор продукта. [Электронный документ] (https://docs.microsoft.com/ru-ru/visualstudio/ide/whats-new-visual-studio-2019?view=vs-2019).

2) Рыжко, А.Л. Информационные системы управления производственной компанией: Учебник для академического бакалавриата / А.Л. Рыжко, А.И. Рыбников, Н.А. Рыжко. - Люберцы: Юрайт, 2016. - 354 с.

3) Волкова В.Н. Теория информационных процессов и систем. — М.: Юрайт, 2016. — 504 с.

4) Грехэм, И. Объектно-ориентированные методы. Принципы и практика / И. Грехэм - М.: «Вильямс», 2004. – 1024с.

5) Мазур, И.И. Управление проектами / ИИ. Мазур, В.Д. Шапиро, Н.Г. Ольдерогге – М.: Омега-Л, 2006. - 664с.

Консультации по работе с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Экономическая часть	Бекишева А.И.	05.05.19	
Безопасность жизнедеятельности	Бекбасаров Ш.Ш.	14.05.19	
Программная часть	Майкотов М.Н.	14.05.19	
Нормоконтралер	Алимсеитова Ж.К.	01.05.19-15.05.19	

График подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Анализ и разработка требований к программному обеспечению	14.01.2019-30.01.2019	выполнено
Проектирование информационной системы	05.02.2019-12.03.2019	выполнено
Работа с графическим интерфейсом и разработка веб-приложения	15.03.2019-13.05.2019	выполнено
Экономическое обоснование разработки проекта	06.05.2019-13.05.2019	выполнено
Безопасность жизнедеятельности	07.05.2019-14.05.2019	выполнено

Дата выдачи задания «26» 10 2018 г.

Заведующий кафедрой _____ Т.С. Картбаев

Научный руководитель проекта _____ А.Т. Купарова

Задание принял к исполнению студент _____ Г.Б. Аслугужиева

Аңдатпа

Дипломдық жоба 93 беттен тұрады, оның ішінде 31 сурет, 16 кесте, 46 әдебиет көзі, 2 қосымша.

Дипломдық жобаның тақырыбы: "Intergaz ақ газ кәсіпорнының ақпараттық жүйесін әзірлеу".

Бұл жұмыста газ кәсіпорны үшін ақпараттық жүйені әзірлеу сипатталған. Дипломдық жоба бағдарламалық өнімді әзірлеудің барлық кезеңдерін сипаттаудан тұрады. Ол пәндік саланы және пәндік саланы автоматтандыру бойынша қолданыстағы шешімдерді талдауды қамтиды. Ақпараттық жүйені жобалау тарауы негізгі ақпаратты және жоба жүйесін жобалау процесі туралы деректерді қамтиды. Ақпараттық жүйені іске асыру және аттестаттау тарауында жұмыстың бағдарламалық бөлімі сипатталған. Сонымен қатар, осы дипломдық жобада өміртіршілік қауіпсіздігі және жобаның экономикалық тиімділігі сипатталған. Сонымен қатар, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету үшін, электр энергиясымен қамтамасыз ету қажет. Әлеуметтік экономикалық тиімділік есебінен осы бағдарламалық өнім 6 ай ішінде өтеледі.

Аннотация

Дипломный проект содержит 93 страницы, в том числе 31 рисунков, 16 таблиц, 46 источников литературы, 2 приложения.

Тема дипломного проекта: «Разработка Информационной системы газового предприятия АО Intergaz».

В данной работе описана разработка информационной системы для газового предприятия. Дипломный проект содержит описание всех стадий разработки программной продукта. Он включает анализ предметной области и существующих решений по автоматизации предметной области. Глава проектирования информационной системы содержит в себе основную информацию и данные о процессе проектирования системы проекта. В главе реализация и аттестация информационной системы описана программная часть работы. Также в данном дипломном проекте описаны главы безопасности жизнедеятельности и экономической эффективности проекта. Так как программа будет использована в офисном помещении с работающими компьютерами, данные в расчетах оказались в пределах нормы, а значит нет необходимости в проведении мер против шума. За счет социально экономического эффекта данный программный продукт окупится в течении 6 месяцев.

Abstract

The diploma project contains 93 pages, including 31 figures, 16 tables, 46 sources of literature, 2 applications.

Theme of the graduation project: "Development of the Information System of the gas enterprise JSC Intergaz".

This paper describes the development of an information system for a gas enterprise. The diploma project contains a description of all stages of the software product development. It includes analysis of the subject area and existing automation solutions for the subject area. The information system design chapter contains basic information and data on the project system design process. The chapter implementation and certification of the information system describes the software part of the work. Also in this thesis project describes the chapter of life safety and economic efficiency of the project. Since the program will be used in an office room with working computers, the data in the calculations were within the normal range, which means there is no need to take measures against noise. Due to the socio-economic effect of this software product will pay off within 6 months.

Содержание

Введение

Информационные технологии с большой скоростью захватывают все бо льшую часть жизнь каждого человека. На данный период времени из-за сильного натиска развития технологий, многие отрасли нуждаются в испол ьзовании информационных систем, в тех частях где они больше всего необход имы.Это поможет свести к минимуму затраты и время на обработку новых, а т акже существующих данных. В дополнении ко всему это поможет повысить п роизводительность труда.

Рассматриваемая дипломная работа написана на базе газового предприятия АО «InterGaz».

Газовая промышленность Республики Казахстан начала свое развитие не так давно, а всего лишь в 70-е годы 20-го века. ПО сравнению с другими отраслями, такими как топливно-энергетического комплекса, угольной, нефтяной, электроэнергетики, которые имеют за своими плечами вековую историю, газовая промышленность находится на ранней стадии развития.

Газотранспортная система РК управляется акционерным обществом «Intergaz Central Asia», которое было создано в июле 1997 года, также ими проводится эксплуатация и техническое обслуживание. Помимо этого, в сфере газа и газоснабжения компания входит в состав национального оператора «КазТрансГаз» и представляет его интересы в области магистральной транспортировки природного газа.

Компания эксплуатирует 3 подземных хранилища газа. Наиболее крупное из них Бозойское ПХГ (с активным объемом хранения 4 000 000 тыс. $м^3$), расположенное в Актюбинской области. Полторацкое ПХГ в Южно-Казахстанской области (с активным объемом хранения 350 000 тыс. $м^3$), а также Акыртобинское ПХГ (с активным объемом хранения 300 000 тыс. $м^3$), в Жамбылской области [1].

Для нашей страны в последнее время эта тема стала наиболее актуальна так как активно началась газификация населенных пунктов. В этих условиях возрастает потребность населения в газовом оборудовании и, следовательно, увеличивается нагрузка на персонал предприятия. Понижается эффективность обработки и поиска информации, увеличивается риск ошибок при выполнении заказов, и т.п. Во избежание этих проблем становится все более актуальной задача использования современных информационных систем, обеспечивающих сбор, обработку и накопление информации, автоматизацию технологических процессов, а также процессов управления и коммуникации.

Целью настоящего дипломного проекта является разработка и внедрение информационной системы для предприятия по установке газового оборудования.

Программа должна обеспечивать:

- работу с входными данными;
- получение выходных документов;
- формирование отчетов.

Хранение всей информации в электронных базах данных, что позволит структурировать информацию, появится возможность быстрого поиска необходимых данных.

Для достижения вышеуказанной цели необходимо решить следующие задачи:

– осуществить бизнес-моделирование процессов предприятия, для разрабатываемой информационной системы;
– провести анализ требований к системе и ее проектирование;
– разработать концептуальную и физическую модели данных;
– провести оценку эффективности технологии разработки.

1 Разработка требований к программному обеспечению

1.1 Анализ предметной области

В рамках дипломного проекта был разработан программный продукт, для его создания послужила цель автоматизации рабочего пространства газового предприятия.

Для более точного понимания газовой отрасли, в которой создавалась данная программа, будут приведены некоторые определения из Закона Республики Казахстан от 4 июля 2018 года №173-VI «О внесении изменений и дополнений в некоторые законодательные акты Республики Казахстан по вопросам газа и газоснабжения».

Система газоснабжения - комплекс технологически взаимосвязанных объектов, предназначенных для производства, транспортировки (перевозки), хранения, реализации и потребления товарного или сжиженного нефтяного газа.

Газификация - комплекс мероприятий, направленных на использование в качестве топлива товарного и (или) сжиженного нефтяного газа на объектах жилищно-коммунального хозяйства, промышленных, сельскохозяйственных и иных объектах.

Целями государственного регулирования в сфере газа и газоснабжения являются обеспечение энергетической и экологической безопасности, улучшение социально-экономического положения населения Республики Казахстан.

Государственное регулирование в сфере газа и газоснабжения основывается на следующих принципах:

1) безопасности и надежности газоснабжения;

2) рациональности использования ресурсов газа;

3) приоритетности обеспечения внутренних потребностей Республики Казахстан товарным и сжиженным нефтяным газом, производимым на территории Республики Казахстан;

4) приоритетности газификации объектов, расположенных на территории Республики Казахстан;

5) сбалансированности ценовой политики в сфере газа и газоснабжения.

Задачи государственного регулирования в сфере газа и газоснабжения:

1) создание единой системы снабжения товарным газом;

2) создание условий для бесперебойного обеспечения внутренних потребностей Республики Казахстан в товарном и сжиженном нефтяном газе;

3) увеличение доли товарного и сжиженного нефтяного газа в общем объеме потребляемых в Республике Казахстан топливно-энергетических ресурсов [2].

Одной из крупнейших газоснабжающих компаний Республики Казахстан является АО «КазТрансГаз», она представляет интересы нашей страны на мировой арене газового рынка. Финансирование, строительство, эксплуатирование трубопроводов и газохранилищ, продажа газа на

внутреннем и внешнем рынках, а также управлением транспортировкой природного газа по магистральным газопроводам является привилегией управления вышеуказанной газовой компании.

Внутренней транспортировкой и транзитом природного газа на территории Казахстана по магистральным трубопроводам протяженностью более 12 тыс. км. Занимается АО «Интергаз Центральная Азия». Помимо этого, компания занимается оказанием услуг технического обслуживания магистральных газопроводов сторонних организаций с общей протяженностью более 6 тыс. км. Транспортировка газа осуществляется 22 линейными и 4 дожимными компрессорными станциями, на которых установлено 300 газоперекачивающих агрегатов различных типов и моделей с общей мощностью 2 157,264 МВт. На договорной основе АО «Интергаз Центральная Азия» обслуживается КС-1 «Бозой» с 5 ГПА общей мощностью 68,5 МВт.

Быстрые темпы роста характеризуются использованием природного газа в качестве основного источника энергии. По данным журнала «Caspian Oil & Gas» Казахстан занимает пятнадцатое место в мире и четвертое место среди стран Содружества Независимых Государств по разведанным запасам природного газа. Свыше 95% балансовых запасов природного газа (в свободном и растворенном виде) сосредоточены в 142 месторождениях, расположенных на территории Атырауской, Актюбинской, ЗападноКазахстанской и Мангыстауской областей. Запасы свободного и попутного газа обнаружены также на территории Кызылординской (месторождения Кумколь, Акшабулак), Жамбылской (месторождение Амангельды) и Южно-Казахстанской (месторождение Придорожное) областей. Далее на рисунке 1.1 показаны газифицированные зоны РК.

Рисунок 1.1 - Газифицированные зоны РК.

Для создания единого рынка газа, а также его развития на территории нашей страны являются следующие основы:

– формирование единого большого круга потребителей газа основанного на внедрении газа в производство и быт в виде топливного и энергетического ресурса на территории субъектов Казахстана, а также развитие газификации в стране;

– создание и поддержка взаимовыгодных экономических отношений между поставщиками газа и его потребителями;

– создание необходимых условий для более надежного обеспечения газом потребителей различных категорий.

Информационная система управления предприятием нуждается в разработке прототипа, для этого необходимо создать структурно-функциональную модель планируемой системы. В связи с этим на предприятии проводится инвентаризация бизнес-процессов по состоянию моделируемой деятельности. Перечень обследованных мест является результатом проделанной работы. Затем для каждой рабочей станции составляется необходимый список функций, документов и справочных работ для выполнения задач. Впоследствии каждый филиал описывается в непосредственной близости от отделов (служб). Они показывают свой организационный состав и лидеров. Далее описывается структура филиала. На основе инвентаризации создается филиал, включающий филиалы, отделы, должности департаментов и обязанности сотрудников по каждому пункту.

По каждому процессу указывается следующее:

– входящие и исходящие документы;

– организационные единицы, который участвуют в данном процессе;

– квалификационные требования к должностям;

– допуски к информации в системе для должностей;

– цель, достигаемая в результате процесса;

– риски, возникающие в ходе подпроцесса;

– ключевые показатели результативности, характеризующие успешность достижения результата.

Для того чтобы более четко разобраться с предметной областью и понять, что требуется от проектируемой информационной системы далее приводится описание существующих бизнес-процессов, которые подлежат автоматизации.

Язык UML включает в себя специальные механизмы расширения, которые позволяют ввести в рассмотрение дополнительные графические обозначения, ориентированные для решения задач из определенной предметной области. Примеры подобных обозначений, которые используются для моделирования бизнес – систем и могут быть изображены на диаграммах вариантов использования: бизнес – актер, сотрудник и бизнес – вариант использования [3].

«Варианты бизнес-использования» - одно из широко известных расширений UML. Он устанавливает субъекты коммерциала и виды его применения (из числа других элементов).

Таблица 1.1 - Функциональные подсистемы

Уровни управле-ния	Функциональные подсистемы			
	Маркетинг	Производство	Логистика	Финансы
Стратеги-ческий	Новые продукты и услуги. Исследования и разработки	Производственные мощности. Выбор технологии	Материальные источники. Товарный прогноз	Финансовые источники. Выбор модели уплаты налогов
Тактичес-кий	Анализ и планирование объемов сбыта	Анализ и планирование производственных программ	Анализ и планирование объемов закупок	Анализ и планирование денежных потоков
Оператив-ный	Обработка требований клиентов. Выписка счетов и накладных	Обработка производствен-ных заказов	Складские операции. Заказы на закупку	Ведение бухгалтерских книг

Основой смысловой идеи модификации прецедента является моделирование взаимодействия между субъектами и системой. Вариантами применения являются единицы данной связи. Согласно умолчанию, контекст представляет собой программную системы, поэтому субъекты являются системными пользователями, а система – программной системой. Примеры использования моделируют применение программной концепции субъектами.

Это смещает роль определенной модификации согласно умолчанию. Система - это далеко не программная система, а бизнес-процесс. Актер никак не user программного обеспечения, а член бизнес-процесса (сотрудник). Бизнес-сценарий – это связь между сотрудником и бизнес-процессом, что никак не непременно обязано содержать концепцию программного представления.

Нельзя напрямую смешивать элементы бизнес-прецедентов и "нормальные" элементы прецедентов, просто потому что они находятся в разных абстракциях. Они взаимосвязаны, и субъект бизнеса может легко дать соответствующий системный субъект, но это 2 разные концепции модели. Примерами сотрудников являются менеджеры, администраторы, кассиры, и инженеры. Общее свойство сотрудников заключается в том, что она являются субъектами и входят в состав моделируемой системы. Применение этих

элементов графической нотации позволяет разработать адекватную модель бизнес-процессов организации.

Для более четкого понимания предметной области необходимо разобраться, что именно требуется от проектируемой информационной системы. Далее приведены примеры бизнес-процессов, которые необходимо автоматизировать для улучшения качества работы.

На рисунке 1.2 представлена схема бизнес-процесса по формированию заказа клиента.

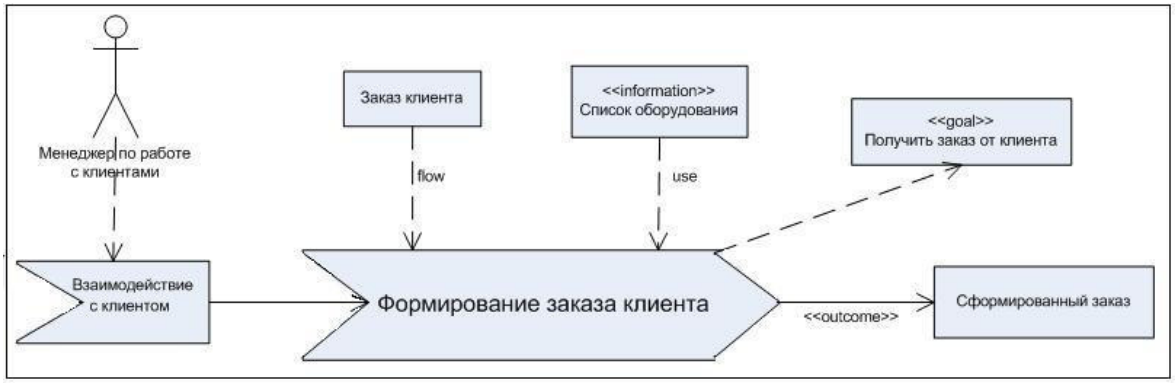


Рисунок 1.2 – Бизнес-процесс «Формирование заказа клиента».

Целью данного бизнес-процесса является получение заказа от клиента. Выходной документ, который формируется в результате этого бизнес-процесса, отражает данные о клиенте и выбранном им оборудовании.

На рисунке 1.3 представлена схема бизнес-процесса расчет с клиентом.

Целью данного бизнес процесса является выставление счета клиенту. Выходной документ, который формируется в результате этого бизнес-процесса, отражает данные об установленном клиенту оборудовании и цены на него.

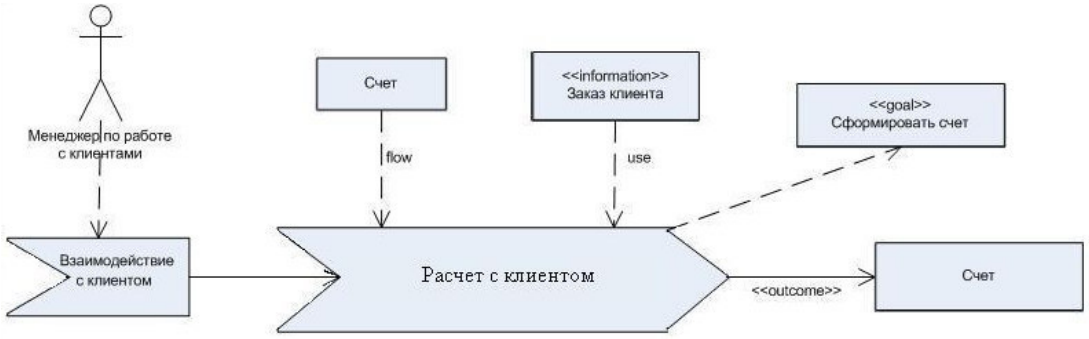


Рисунок 1.3 – Бизнес-процесс «Расчет с клиентом».

На рисунке 1.4 представлена схема бизнес-процесса контроль по срокам гарантии.

Рисунок 1.4 – Бизнес-процесс «Контроль по срокам гарантии».

Целью данного бизнес-процесса является контроль по срокам гарантии установленного оборудования. Этот бизнес-процесс является неотъемлемой частью взаимодействия с клиентом.

На рисунке 1.5 представлена схема бизнес-процесса составление акта о выполненной работе.

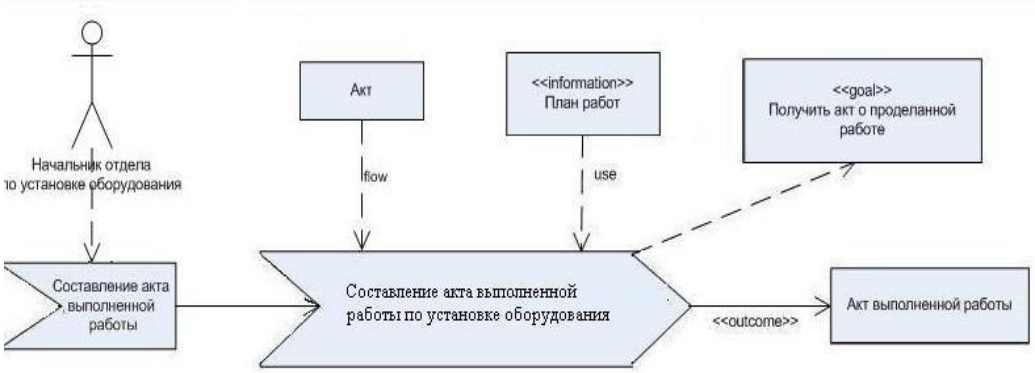


Рисунок 1.5 – Бизнес-процесс «Составление акта о выполненной работе».

Целью данного бизнес-процесса является составление акта о выполненной работе. Выходной документ, который формируется в результате этого бизнес-процесса, отражает данные об установленном оборудовании в соответствии с заказом клиента.

Структура подсистем ИС, выделенных по функционально-предметному принципу, приведена на рисунке 1.6.

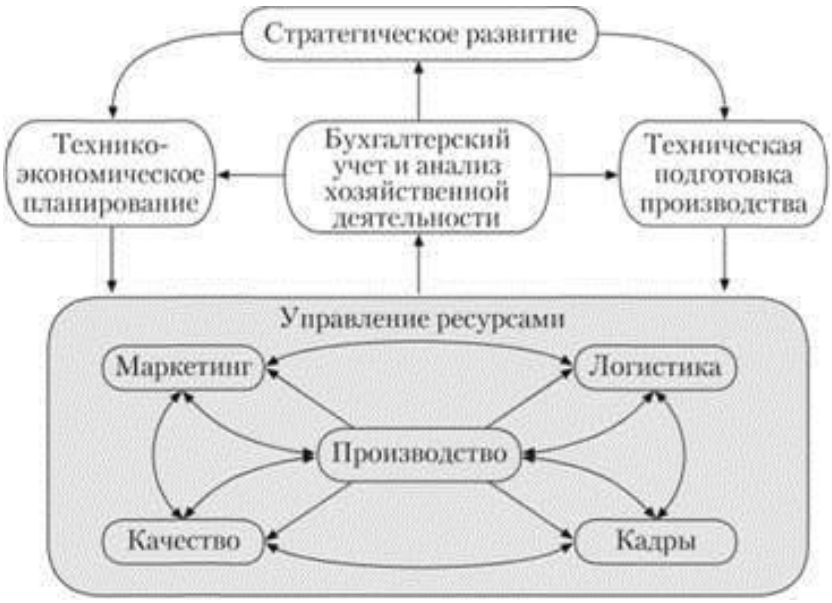


Рисунок 1.6 – Структура подсистем ИС.

Бизнес-модели - это стилизованные модели, которые описывают, как компании создают и обеспечивают ценность для своих клиентов, и как они получают вознаграждение за это. Конструкция бизнес-модели включает в себя продукт или услугу, клиента и рынок, роль компании в цепочке создания стоимости и экономический двигатель, который позволяет ей достичь своих целей прибыльности и роста. Бизнес-модели часто используются стартапами в качестве инструментов моделирования, которые помогают им проектировать, создавать прототипы и создавать новые предприятия. Они также используются признанными компаниями для планирования, разработки и поддержки своего инновационного процесса. В этой главе я использую конструкцию бизнес-модели, чтобы предсказать, как будут развиваться архитектуры компаний и процессы разработки бизнес-моделей в будущем.

Бизнес-модель - это структурированный проект, который пытается навести порядок и дисциплину в хаотическом процессе построения, развития и ведения бизнеса. Некоторые авторы широко определяют концепцию бизнес-модели 1, которая, как мне кажется, снижает ее удобство использования. Мой взгляд на концепцию бизнес-модели сосредоточен на том, как бизнес создает стоимость и извлекает доходы и прибыль, что определяется тремя основными элементами: моделью создания стоимости, моделью прибыли и логикой бизнеса. [4]. Каждая компания имеет свою собственную уникальную бизнес-модель. От степени ее совершенства зависит эффективность текущей деятельности и устойчивое будущее развитие компании. Бизнес модель компании предоставлена на рисунке 1.7. Также элементы бизнеса, отображаемые в бизнес модели предоставлены на рисунке 1.8.

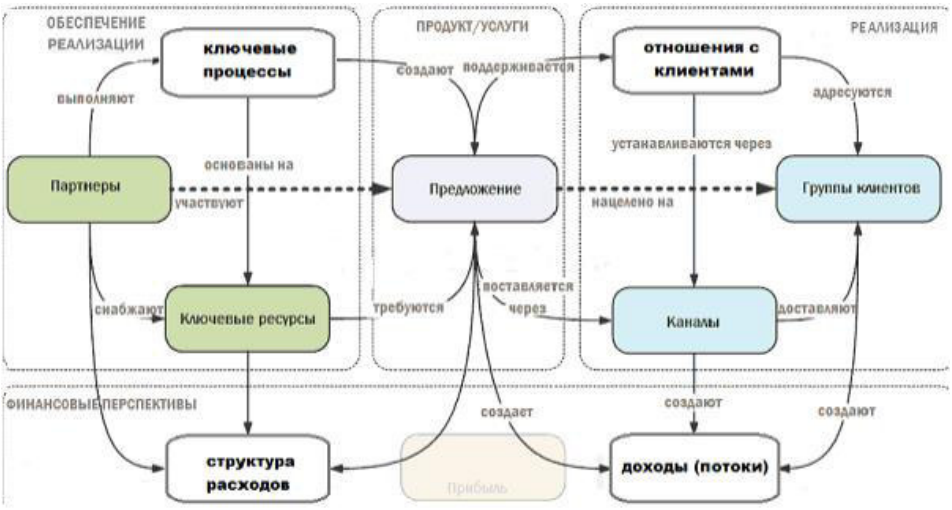


Рисунок 1.7 – Бизнес-модель компании.

Рисунок 1.8 – Элементы бизнеса, отражаемые в бизнес-моделях.

Указание модели создания стоимости. Это спецификация модели создания стоимости. Это дифференцированная ценность для клиентов. Это важно, так как это позволяет изменить ситуацию. Размеры дифференциации варьируются в зависимости от компании. Например, Walmart создает дифференцированную ценность для потребителей для недорогих потребителей. Apple создает дифференцированную ценность для

потребителей, «инновационные, инновационные продукты. Например, он принимает чек, вы сможете его использовать.

Это не проблема. Если вы ищете, что делать. Не нужно будет это делать.

Наконец, создание стоимости происходит по сквозной цепочке создания стоимости. Тем не менее, компании на самом деле будут участвовать. Например, его можно использовать либо в производстве, либо в распределении. Если вы компания, это не проблема. У компании есть собственная торговая марка. Для конечного продукта.

Указание модели прибыли. Это список деловых расходов. Так как это модель прибыли для каждого клиента.

Модели доходов. Например, 3 доллара за галлон бензина. Транзакционные доходы могут также включать ставки дисконтирования.

Это цитата другого типа шаблона. использовать в течение периода подписки (например, членство в тренажерном зале).

Если вы являетесь владельцем лицензии, вам придется заплатить за нее. Например, за использование программного обеспечения взимается лицензионный сбор.

Предприятия часто получают несколько потоков доходов. Например, если вы находитесь в интернет-магазине, вы не можете составить список. Его можно оценить. Это список доходов абонентов, список драйверов доходов и транзакции.

Концепция бизнес-моделирования хорошо соотносится с ресурсной концепцией стратегического управления. Бизнес-модель призвана отразить уникальный набор внутренних ресурсов и способностей компании, особые способы их комбинации, что порождает инновации и способствует созданию конкурентных преимуществ компании. Внутренние ресурсы и способности становятся фундаментом для выработки долгосрочной стратегии. Они помогают компании создавать ценность для потребителя и выживать в условиях конкуренции. Длительный успех в бизнесе многих успешных компаний обусловлен их стремлением использовать уникальные ресурсы, развивать организационные способности, что позволяет им быстро адаптироваться к изменяющимся условиям внешней среды.

При проектировании информационного обеспечения существенное значение имеет время доступа - интервал времени от момента поступления в систему запроса до момента представления соответствующей информации пользователю.

На первом этапе проектирования информационной системы необходимо декомпозировать комплекс проблем, т.е. определить, к какой из групп относится каждая конкретная задача. После этого можно осуществить переход ко второму этапу проектирования, на котором для каждой группы задач необходимо идентифицировать метод ее решения. Так, задачу оперативного контроля параметров технологического процесса можно решить путем разделения множества всех его состояний на состояния, характеризующие

номинальное течение процесса, и состояния, требующие вмешательства в данный технологический процесс с целью его стабилизации.

При решении такой задачи, как передача технологических данных в производственно-диспетчерскую службу предприятия, необходимо предварительно усовершенствовать систему показателей в целях совместимости с другими компонентами и модулями проектируемой единой информационной системы.

1.2 Анализ существующих решений по автоматизации предметной области

Основные возможности программы:

– единый архив для существующей документации;

– ведение справочников, которые хранят контактную и другую информацию об организации, сотрудниках, и клиентах;

– определенный набор типовой документации, отражающей предметную область, возможность расширения состава документов;

– пользовательская среда для работы с электронными документами и электронными копиями первичных документов, включающая возможности по организации персональных и общих библиотек, каталогизации и классификации документов, хранения структуры взаимосвязей и состояний документов, возможности поиска по реквизитам и содержанию;

– возможность работы с электронной почтой и хранение всей корреспонденцию;

– регистрация первичных документов и хранение их электронных копий;

– формирование реестров документов;

– формирование заданий на исполнение, контроль исполнения документов, отчеты по исполнительской дисциплине;

– учет важных документов по местам хранения;

– работа со списками документов - группировки документов в дела;

– автоматизация документооборота;

– все документы системы включены в систему документооборота;

– схемы обработки (маршруты движения) документов доступны для редактирования и настройки под особенности работы.

Другие возможности:

– напоминания;

– оперативный журнал событий, заданий;

– пообъектный учет;

– аудит действий пользователя в системе;

– средства совместной работы над документами - блокировки, удостоверение содержания.

Фрагменты конфигурации, отвечающие за использования внешних справочников через Интернет, обмен документами, поставляются без

исходных текстов и не могут быть изменены. В конфигурации предусмотрена возможность локализации и работа пользователей на нескольких языках в рамках одной информационной базы. Особенности лицензирования: Основная поставка дает право на работу с одной информационной базой (информационная база может быть распределенной) и не ограничивает пользователя рамками локальной сети.

1.3 Выбор методологии проектирования информационной системы

В любой информационной системе основой любого проекта является методология. Она реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые позволяют выполнять все процессы жизненного цикла.

Существует две основных методологии проектирования:

– методология структурного проектирования;

– методология объектно-ориентированного проектирования.

Разработка программного обеспечения состоит из нескольких важных действий, которые компилируются и объединяются для создания программного обеспечения, которое обладает безупречными качествами, функциями и функциональностью. Именно эти компоненты программного обеспечения определяют его масштабируемость, производительность, надежность, безопасность и многое другое, а также обеспечивают разработку программного обеспечения или приложения в соответствии с требованиями клиента. Однако, чтобы инициировать и осуществлять такие действия, инженеры-программисты должны подготовить надлежащий дизайн для программного обеспечения, которым они могут руководствоваться в процессе разработки программного обеспечения.

Разработка программного обеспечения - это процесс, с помощью которого агент создает спецификацию программного артефакта, предназначенного для достижения целей, с использованием набора примитивных компонентов и с учетом ограничений. Наличие процесса проектирования при разработке программного обеспечения позволяет разработчикам быть более эффективными и прозрачными. Более того, он позволяет всем членам разных команд следить за ними и позволяет им легко сотрудничать. С помощью разработки программного обеспечения разработчики могут снизить риски, поскольку они будут использовать проверенный и протестированный план, в котором не требуется ни повторение шагов, ни догадки.

Кроме того, существует несколько типов разработки программного обеспечения, которые используются для упрощения процесса проектирования, а также для снижения сложности программного обеспечения. Объектно-ориентированное проектирование (OOD) является одним из подходов проектирования программного обеспечения и определяется как процесс планирования системы взаимодействующих объектов с целью решения программной проблемы. Все компоненты информационной системы

взаимосвязаны, система разрабатывается сверху – вниз. При разработке системы снизу – вверх целостность теряется, возникают проблемы состыковки компонентов.

Наиболее распространенные модели структурного подхода:

SADT – Structured Analysis and Design Techniques – описывает модели и функциональные диаграммы;

SADT использует два типа диаграмм: модели деятельности и модели данных. Он использует стрелки для построения этих диаграмм. Представление SADT заключается в следующем:

– Главное окно, в котором указано имя процесса или действия

– В левой части этого поля входящие стрелки: входы действия.

– В верхней части входящие стрелки: данные, необходимые для действия.

– Внизу коробки, входящие стрелки: средства, используемые для действия.

– На правой стороне окна, исходящие стрелки: результаты действия.

Семантика стрелок для действий:

– Входные данные вводятся слева и представляют данные или расходные материалы, необходимые для выполнения операции.

– Выходы выходят вправо и представляют данные или продукты, произведенные действием.

– Элементы управления вводятся сверху и представляют команды или условия, которые влияют на выполнение действия, но не используются.

– Механизмы определяют средства, компоненты или инструменты, используемые для выполнения деятельности. Представляет распределение деятельности.

Семантика стрелок для данных:

– Входные данные - действия, которые производят данные.

– Выходы потребляют данные.

Элементы управления влияют на внутреннее состояние данных.

DFD – Data Flow Diagrams – диаграммы потоков данных;

Диаграмма потока данных (DFD) отображает поток информации для любого процесса или системы. Он использует определенные символы, такие как прямоугольники, круги и стрелки, а также короткие текстовые метки, чтобы показать входные данные, выходные данные, точки хранения и маршруты между каждым пунктом назначения. Блок-схемы данных могут варьироваться от простых, даже нарисованных от руки обзоров процессов, до всесторонних многоуровневых DFD, которые постепенно углубляются в то, как обрабатываются данные. Их можно использовать для анализа существующей системы или моделирования новой. Как и все лучшие диаграммы и схемы, DFD часто может визуально «сказать» то, что было бы трудно объяснить словами, и они работают как для технической, так и нетехнической аудитории, от разработчика до генерального директора. Вот

почему DFD остаются такими популярными после всех этих лет. Хотя они хорошо работают с программным обеспечением и системами для передачи данных, в настоящее время они менее применимы для визуализации интерактивного программного обеспечения или систем, ориентированных на работу в режиме реального времени или в базе данных.

ERD – Entity Relationship Diagrams – диаграммы «сущность – связь».

Диаграмма отношений объекта (ERD) показывает отношения наборов объектов, хранящихся в базе данных. Сущность в этом контексте является объектом, компонентом данных. Набор объектов - это набор похожих объектов. Эти объекты могут иметь атрибуты, которые определяют его свойства.

Определяя сущности, их атрибуты и показывая отношения между ними, диаграмма ER иллюстрирует логическую структуру баз данных.

Диаграммы ER используются, чтобы наметить проект базы данных.

Есть две причины для создания диаграммы базы данных. Вы либо разрабатываете новую схему, либо вам необходимо документировать существующую структуру.

Если у вас есть существующая база данных, которую нужно документировать, вы создаете диаграмму базы данных, используя данные непосредственно из вашей базы данных. Вы можете экспортировать структуру вашей базы данных в виде CSV-файла (здесь есть несколько сценариев, как это сделать), затем программа автоматически сгенерирует ERD.

Это будет самый точный портрет вашей базы данных и не потребует никаких рисунков с вашей стороны.

На стадии проектирования ИС модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектуру ПО, структурные схемы программ и диаграммы экранных форм.

Объект – это конкретная реализация некоторой сущности. В объекте инкапсулируется некоторая часть приложения, которая может представлять собой процесс, группу данных или какую-либо более сложную сущность.

Для реализации проекта был выбран объектно-ориентированный подход в силу следующих факторов:

- возможность повторного использования кода;
- повышение безопасности кода за счет инкапсуляции;
- гибкость при модификации и расширении системы;
- отсутствие необходимости разработки классов с нуля, за счет наследования;
- общая ориентированность объектно-ориентированной технологии на разработку информационных систем, как класса программного обеспечения и т.д.

1.4 Сбор и спецификация требований

Основной целью рабочего процесса является определение требований, они состоят в направленности процесса разработки на получение правильной системы. Исходя из этого следует что, правильная система – это та система, которая делает все необходимое и ничего более. Требование - это оценка либо критерий, который должна исполнять система.

В любой рабочей деятельности основным аспектом является сбор требований так как в том случае если разработчик будет знать, что именно от него требуется, то в итоге результат будет удовлетворителен как для первой стороны, так и для второй. Для повышения работоспособности и результативности необходим более упрощенный и удобный процесс работы, который сопровождается минимизацией временных затрат и наличие продуктивного результата.

Сбор требований – это процесс, который включает в себя те мероприятия, что необходимы для создания и утверждения документа, содержащего спецификацию системных требований.

На этапе формирования и анализа требований в соответствии с технологией разработки программного обеспечения Microsoft Solution Framework (MSF):

– осуществляется сбор требований;

– составляются профили заинтересованных лиц;

– разрабатываются варианты использования.

MSF 4.0 представляет собой комбинацию метамодели, которую можно использовать в качестве основы для предписывающих процессов разработки программного обеспечения, и двух настраиваемых и масштабируемых процессов разработки программного обеспечения. Метамодель MSF состоит из основополагающих принципов, модели команды, циклов и итераций.

MSF 4.0 предоставляет высокоуровневую структуру руководящих указаний и принципов, которые можно сопоставить с различными шаблонами предписывающих процессов. Он структурирован как в описательных, так и в предписывающих методологиях. Описательный компонент называется метамоделью MSF 4.0, которая представляет собой теоретическое описание лучших практик SDLC для создания методологий SDLC. Microsoft считает, что организации имеют разную динамику и противоположные приоритеты при разработке программного обеспечения; некоторым организациям нужна адаптивная и адаптируемая среда разработки программного обеспечения, в то время как другим нужна стандартизированная, воспроизводимая и более контролируемая среда. Чтобы удовлетворить эти потребности, Microsoft представляет метамодель MSF 4.0 в двух шаблонных методологических шаблонах, которые обеспечивают конкретное руководство процессом, для гибкой разработки программного обеспечения (MSF4ASD) и для модели зрелости возможностей (MSF4CMMI). Эти процессы разработки программного обеспечения могут быть изменены и настроены в соответствии с предпочтениями организации, клиента и проектной команды.

Философия MSF гласит, что не существует единой структуры или процесса, который бы оптимально подходил к требованиям и средам для всех видов проектов. Поэтому MSF поддерживает несколько подходов к процессам, поэтому его можно адаптировать для поддержки любого проекта, независимо от его размера или сложности. Эта гибкость означает, что она может поддерживать широкий спектр вариантов реализации процессов разработки программного обеспечения, сохраняя при этом набор основных принципов и установок.

Модель процесса MSF состоит из серии коротких циклов разработки и итераций. Эта модель охватывает быстрое итеративное развитие с постоянным обучением и совершенствованием благодаря прогрессивному пониманию бизнеса и проекта заинтересованных сторон. Идентификация требований, разработка продукта и тестирование выполняются в виде перекрывающихся итераций, что приводит к поэтапному завершению для обеспечения потока стоимости проекта. Каждая итерация имеет различную направленность и приводит к стабильной части всей системы.

Спецификация требований - процесс документирования системы в структурированном, доступном всем и управляемом формате. Спецификация требований (Software Requirements Specification, SRS) используется для текущего сопровождения проекта и представления требований, сформулированных по отношению к проекту. Спецификация требований к программному обеспечению (SRS) - это описание системы программного обеспечения, которая будет разработана. Он смоделирован по спецификации бизнес-требований (CONOPS), также известной как спецификация требований заинтересованных сторон (StRS). [Цитата нужна] В спецификации требований к программному обеспечению изложены функциональные и нефункциональные требования, и она может включать набор вариантов использования, которые описывают взаимодействие с пользователем, которое программное обеспечение должно предоставлять пользователю для идеального взаимодействия.

Спецификация требований к программному обеспечению устанавливает основу для соглашения между клиентами и подрядчиками или поставщиками о том, как должен функционировать программный продукт (в рыночном проекте эти роли могут играть отделы маркетинга и разработки). Спецификация требований к программному обеспечению - это строгая оценка требований перед более конкретными этапами проектирования системы, и ее цель состоит в том, чтобы сократить последующую модернизацию. Он также должен обеспечить реальную основу для оценки затрат на продукт, рисков и графиков. При правильном использовании спецификации требований к программному обеспечению могут помочь предотвратить сбой программного проекта.

В документе спецификации требований к программному обеспечению перечислены достаточные и необходимые требования для разработки проекта. Чтобы получить требования, разработчик должен иметь четкое и полное

понимание разрабатываемых продуктов. Это достигается за счет подробного и постоянного общения с командой проекта и заказчиком на протяжении всего процесса разработки программного обеспечения.

SRS может быть одним из описаний доставляемых по контракту элементов данных или иметь другие формы организационно-обязательного контента [5].

1.5 Анализ и моделирование требований

Для точного определения функциональных требований, которые предъявляются системе, прежде всего нужно выявить тех, кто заинтересован в данной системе после чего определить необходимый функционал, требуемый для осуществления профессиональной деятельности.

Те пользователи, что заинтересованы в предлагаемой системе, были выявлены непосредственно в процессе проведенного исследования бизнес-процессов, а также в предпроектном обследовании предприятия, изображены на рисунке 1.9.

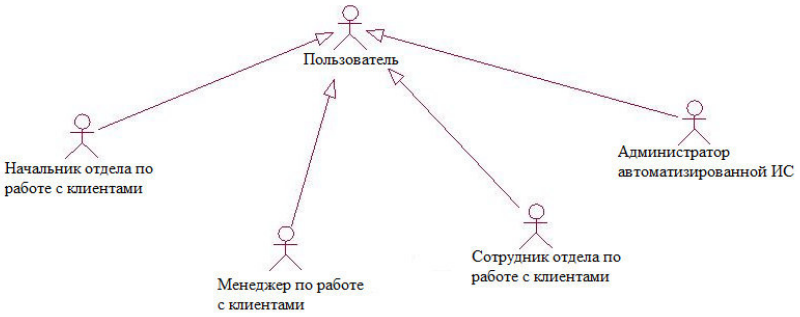


Рисунок 1.9 – Пользователи системы.

Данная схема отображает основных пользователей информационной системы предприятия.

После выявления основных пользователей системы, необходимо провести анализ вариантов того, как будет использована система. Фактическими функциональными требованиями к системе являются прецеденты.

На рисунке 1.10 представлены варианты использования системы менеджером по работе с клиентами.

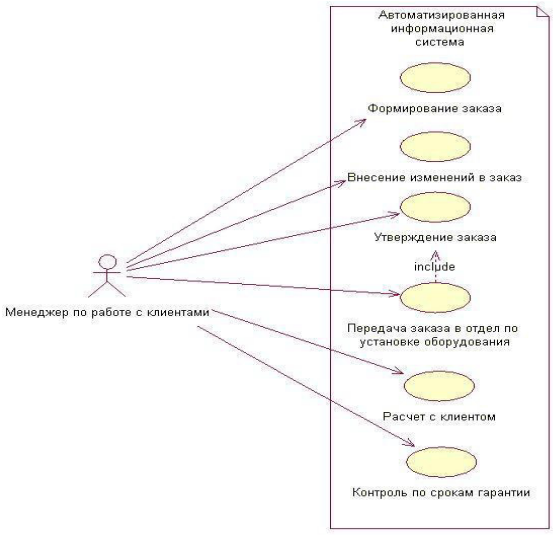


Рисунок 1.10 – Варианты использования системы менеджером по работе с клиентами

Непосредственно в процессе выполнения формирования заказа, пользователь должен выполнить пошаговое формирование необходимого заказа, при помощи последовательного ввода данных.

На рисунке 1.11 представлены варианты использования системы начальником отдела по установки оборудования.

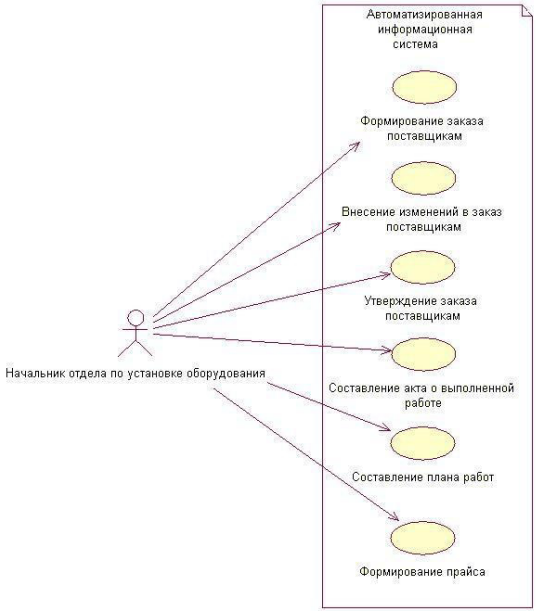


Рисунок 1.11 – Варианты использования системы менеджером по персоналу

На рисунке 1.12 представлены варианты использования системы администратора автоматизированной системы.

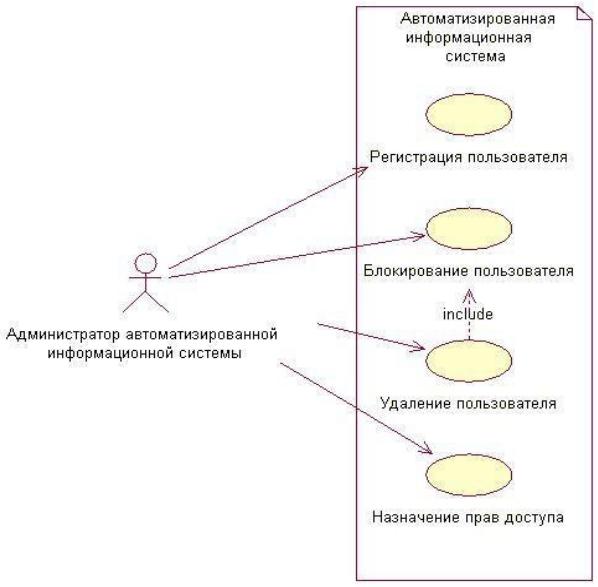


Рисунок 1.12 – Варианты использования системы специалистом администратором

1.6 Аттестация требований

Аттестация требований – процесс проверки требований на достоверность, непротиворечивость, полноту и выполнимость.

В управлении проектами программного обеспечения, тестировании программного обеспечения и разработке программного обеспечения, верификация и валидация (V & V) - это процесс проверки того, что система программного обеспечения соответствует спецификациям и соответствует ли она своему назначению. Это может также упоминаться как контроль качества программного обеспечения. Обычно это ответственность тестировщиков программного обеспечения как часть жизненного цикла разработки программного обеспечения.

Это подразумевает проверку соответствия спецификаций при запуске программного обеспечения, но это невозможно (например, как кто-нибудь может узнать, правильно ли реализована архитектура / дизайн / и т. Д. При запуске программного обеспечения?). Только просмотрев связанные с ним артефакты, кто-то может сделать вывод, что спецификации удовлетворены.

Проверка программного обеспечения проверяет, что программный продукт удовлетворяет или соответствует предполагаемому использованию (высокоуровневая проверка), то есть программное обеспечение удовлетворяет требованиям пользователя, а не артефактам спецификации или потребностям тех, кто будет использовать только программное обеспечение; но, как и потребности всех заинтересованных сторон (таких как пользователи, операторы, администраторы, менеджеры, инвесторы и т. д.).

Существует два способа проверки программного обеспечения: внутренний и внешний. Во время внутренней проверки программного

обеспечения предполагается, что цели заинтересованных сторон были правильно поняты и что они были выражены в артефактах требований точно и всесторонне. Если программное обеспечение соответствует требованиям спецификации, оно прошло внутреннюю проверку. Внешняя проверка происходит, когда она выполняется путем запроса заинтересованных сторон, соответствует ли программное обеспечение их потребностям. Различные методологии разработки программного обеспечения требуют разных уровней участия пользователей и заинтересованных сторон и обратной связи; Итак, внешняя проверка может быть дискретным или непрерывным событием. Успешная окончательная внешняя проверка происходит, когда все заинтересованные стороны принимают программный продукт и заявляют, что он удовлетворяет их потребности. Такая окончательная внешняя проверка требует использования приемочного испытания, которое является динамическим испытанием.

Тем не менее, также возможно выполнить внутренние статические тесты, чтобы выяснить, соответствует ли он спецификации требований, но это входит в область статической проверки, поскольку программное обеспечение не работает.

Согласно модели зрелости возможностей (CMMI-SW v1.1),

Проверка программного обеспечения: процесс оценки программного обеспечения во время или в конце процесса разработки, чтобы определить, удовлетворяет ли он указанным требованиям. [IEEE-STD-610]

Проверка программного обеспечения: процесс оценки программного обеспечения для определения того, удовлетворяют ли продукты данного этапа разработки условиям, установленным в начале этого этапа. [IEEE-STD-610]

Проверка в процессе разработки программного обеспечения может рассматриваться как форма проверки спецификации требований пользователя; и что в конце процесса разработки эквивалентна внутренней и / или внешней проверке программного обеспечения. С точки зрения CMMI, верификация, очевидно, относится к типу артефактов.

Другими словами, проверка программного обеспечения гарантирует, что выходные данные каждой фазы процесса разработки программного обеспечения эффективно выполняют то, что указывает соответствующий им входной артефакт (требование -> дизайн -> программный продукт), в то время как проверка программного обеспечения гарантирует, что программный продукт отвечает потребностям все заинтересованные стороны (следовательно, спецификация требований была правильно и точно выражена в первую очередь). Проверка программного обеспечения гарантирует, что «вы все правильно сделали» и подтверждает, что продукт, как предусмотрено, соответствует планам разработчиков. Проверка программного обеспечения гарантирует, что «вы создали правильную вещь» и подтверждает, что продукт, если он предусмотрен, соответствует предполагаемому использованию и целям заинтересованных сторон.

В этой статье используется строгое или узкое определение проверки.

С точки зрения тестирования:

– Ошибка - неправильная или отсутствующая функция в коде.

– Неудача - проявление ошибки во время исполнения. Программное обеспечение не было эффективным. Он не делает то, что должен делать.

– Неисправность - в соответствии со своей спецификацией система не соответствует указанным функциям. Программное обеспечение было неэффективным (потребовалось слишком много ресурсов, таких как циклы ЦП, использовалось слишком много памяти, было выполнено слишком много операций ввода-вывода и т. д.).

1.7 Выводы к разделу

После проведенного анализа в данной предметной области, было выявлено те задачи, которые необходимо автоматизировать во время разработки информационной системы предприятия. После просмотра существующих решений по информатизации было выявлено что проведение индивидуальной разработки информационной системы предприятия более целесообразно. На основе бизнес-процессов, а также вариантах их использования были разработаны и специфицированы требования к информационной системе, которые определяют последующие этапы создания информационной системы.

Таким образом объектно-ориентированный подход был использован в качестве методологии проектирования.

2 Проектирование информационной системы

2.1 Архитектурное проектирование

Вне зависимости от того на сколько сложна информационная система, при ее создании критическим аспектом является ее архитектура, в которой она представляет из себя концептуальное видение структуры функциональных процессов и технологий на системном уровне, а также во взаимосвязи. Обычно сложные информационные системы организаций проектируются в виде композиций компонентов, которые взаимодействуют на высоком уровне, и сами могут быть системами. Архитектура информационной системы организации делает понимание системы легче, определяя ее функциональность и структуру способом, который раскрывает проектировочные решения и позволяет обозревателю задавать вопросы об удовлетворении проектных требований, распределении функциональности и реализации компонентов.

При проектировании современных информационных систем организаций их архитектура должна разрабатываться с учетом многих заинтересованных сторон, она должна быть понятной не только пользователям, но и дать возможность разработчикам сделать план и графики системы, позволять определять ключевые интерфейсы, функции и технологии, а также позволять оценить график и бюджет исполнения проекта.

Архитектуры информационной системы требуется соблюдение следующих условий:

– соответствие с миссией организации;
– определенность в требованиях;
– направленность в разработке;
– возможность к адаптации;
– необходимость гибкости.

Соблюдение всех этих условий позволяет разработать архитектуру информационной системы организации, более совершенной и эффективной.

Очень большой и дорогой компьютер, способный одновременно поддерживать сотни или даже тысячи пользователей. В иерархии, которая начинается с простого микропроцессора (например, в часах) внизу и перемещается к суперкомпьютерам вверху, мэйнфреймы находятся чуть ниже суперкомпьютеров. В некотором смысле, мэйнфреймы более мощные, чем суперкомпьютеры, потому что они поддерживают больше одновременных программ. Но суперкомпьютеры могут выполнять одну программу быстрее, чем мэйнфрейм.

Клиент-серверная архитектура (клиент / сервер) - это сетевая архитектура, в которой каждый компьютер или процесс в сети является либо клиентом, либо сервером. Серверы - это мощные компьютеры или процессы, предназначенные для управления дисководами (файловые серверы), принтерами (серверы печати) или сетевым трафиком (сетевые серверы). Клиентами являются ПК или рабочие станции, на которых пользователи

запускают приложения. Клиенты полагаются на серверы для получения ресурсов, таких как файлы, устройства и даже вычислительная мощность. Другой тип сетевой архитектуры известен как одноранговая архитектура, потому что каждый узел имеет эквивалентные обязанности. Архитектура как клиент-сервер, так и одноранговая, широко используются, и каждая из них имеет свои уникальные преимущества и недостатки.

Веб-интерфейс. Здесь есть модель клиент-сервер, а не веб-браузеры приложений. Клиентский компьютер подключен к веб-серверу через LAN WAN и т. д. использовался исходный HTML. Страницы были статичными с помощью Java-скрипта и AJAX динамические страницы видны.

Клиент отправляет запрос на сервер через прикладную программу. Роль клиента заключается в обработке запроса пользователя и переводе в нужный протокол для пересылки его на сервер. Например, клиент может потребовать повышения статуса «оплаты взносов» учащегося в МОМ. Эти данные хранятся на сервере для дальнейшего использования.

Возможно, вы слышали, как люди используют такие термины, как облако, облачные вычисления или облачное хранилище. Но что такое облако? По сути, облако - это Интернет, точнее, все, к чему вы можете получить удаленный доступ через Интернет. Когда что-то находится в облаке, это означает, что оно хранится на серверах в Интернете, а не на вашем компьютере. Он позволяет вам получать доступ к своему календарю, электронной почте, файлам и многому другому с любого компьютера, подключенного к Интернету. Если вы когда-либо пользовались электронной почтой через Интернет, вы использовали облако. Все электронные письма в вашем почтовом ящике хранятся на серверах. Однако есть много других сервисов, которые используют облако по-разному.

Облачные вычисления - это использование различных служб, таких как платформы разработки программного обеспечения, серверы, хранилище и программное обеспечение, через Интернет, часто называемые «облаком». В целом, есть три характеристики облачных вычислений, которые являются общими для всех поставщиков облачных вычислений:

– Серверная часть приложения (особенно аппаратное обеспечение) полностью управляется поставщиком облачных вычислений.

– Пользователь платит только за используемые услуги (память, время обработки и пропускная способность и т. Д.).

– Услуги являются масштабируемыми.

Многие достижения облачных вычислений тесно связаны с виртуализацией. Возможность быстрой оплаты по требованию и масштабирования в значительной степени является результатом того, что поставщики облачных вычислений могут объединять ресурсы, которые могут быть разделены между несколькими клиентами. Распространено классифицировать сервисы облачных вычислений как инфраструктуру как услугу, например, хранилище (IaaS), платформа как услуга, например операционная система (PaaS) (IAAS + P) программное обеспечение как услуга

Программное обеспечение ERP / CRM (SaaS) - вместо получения лицензии на аренду программного обеспечения (IAAS + PAAS + S) окончательный расчет по Infosys Salesforce.com по TCS

Распределенные вычисления - это вычислительная концепция, которая в самом общем смысле относится к нескольким компьютерным системам, работающим над одной проблемой. В распределенных вычислениях одна проблема делится на множество частей, и каждая часть решается разными компьютерами. Пока компьютеры объединены в сеть, они могут связываться друг с другом для решения проблемы. Если все сделано правильно, компьютеры работают как единое целое. Конечная цель распределенных вычислений - максимизировать производительность, соединяя пользователей и ИТ-ресурсы экономически эффективным, прозрачным и надежным способом. Это также обеспечивает отказоустойчивость и обеспечивает доступ к ресурсам в случае сбоя одного из компонентов.

Грид-вычисления - это процессорная архитектура, которая объединяет ресурсы компьютера из разных областей для достижения основной цели. В грид-компьютинге компьютеры в сети могут совместно работать над задачей, функционируя как суперкомпьютер. Как правило, сетка работает для различных задач в сети, но она также может работать в специализированных приложениях. Он предназначен для решения проблем, которые слишком велики для суперкомпьютера, и в то же время обеспечивает гибкость при обработке множества мелких проблем. Вычислительные сети предоставляют многопользовательскую инфраструктуру, которая удовлетворяет прерывистым требованиям большого объема информации.

На рисунке 2.1 представлена схема клиент-серверной архитектуры.

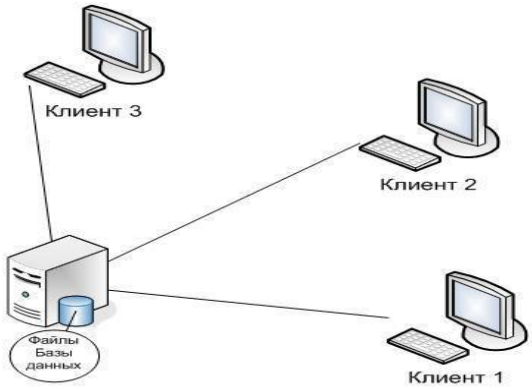


Рисунок 2.1 – Клиент-серверная архитектура.

Архитектура приложения, разделяющая пользовательские и прикладные сервисы и сервисы данных. Другое название - трехуровневая архитектура, однако термин «многоуровневая» корректнее, поскольку он предполагает, что при логическом проектировании может возникнуть более трех уровней сервисов. В разработке программного обеспечения многоуровневая архитектура (часто называемая n-уровневой архитектурой) или

многоуровневая архитектура - это архитектура клиент-сервер, в которой функции представления, обработки приложений и управления данными физически разделены. Наиболее распространенным применением многоуровневой архитектуры является трехуровневая архитектура.

N-уровневая архитектура приложений обеспечивает модель, с помощью которой разработчики могут создавать гибкие и повторно используемые приложения. Разделяя приложение по уровням, разработчики получают возможность изменить или добавить определенный слой вместо того, чтобы перерабатывать все приложение. Трехуровневая архитектура обычно состоит из уровня представления, уровня логики домена и уровня хранения данных.

Хотя понятия уровня и уровня часто используются взаимозаменяемо, одна довольно распространенная точка зрения заключается в том, что разница действительно существует. Это представление гласит, что уровень - это механизм логического структурирования для элементов, составляющих программное решение, а уровень - это механизм физического структурирования для инфраструктуры системы. Например, трехуровневое решение может быть легко развернуто на одном уровне, например, на персональной рабочей станции.

В логической многоуровневой архитектуре для информационной системы с объектно-ориентированным дизайном наиболее распространены следующие четыре:

– уровень представления (уровень пользовательского интерфейса, уровень представления, уровень представления в многоуровневой архитектуре)

– прикладной уровень (сервисный уровень или уровень контроллера GRASP)

– бизнес-уровень (a.k.a. уровень бизнес-логики (bll), уровень домена)

– уровень доступа к данным (например, уровень персистентности, журналирование, сетевые и другие услуги, необходимые для поддержки определенного бизнес-уровня)

В книге «Конструкция, управляемая доменом» описаны некоторые распространенные способы использования вышеперечисленных четырех уровней, хотя основное внимание в ней уделяется уровню домена.

Если в прикладной архитектуре нет явного различия между бизнес-уровнем и уровнем представления (т.е. уровень представления считается частью бизнес-уровня), то была реализована традиционная модель клиент-сервер (двухуровневая).

В данном проекте выбрана клиент-серверная архитектура, т.к. информационная система будет использовать одну базу данных на нескольких рабочих станциях.

Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

Диаграмма компонентов системы представлена на рисунке 2.2.

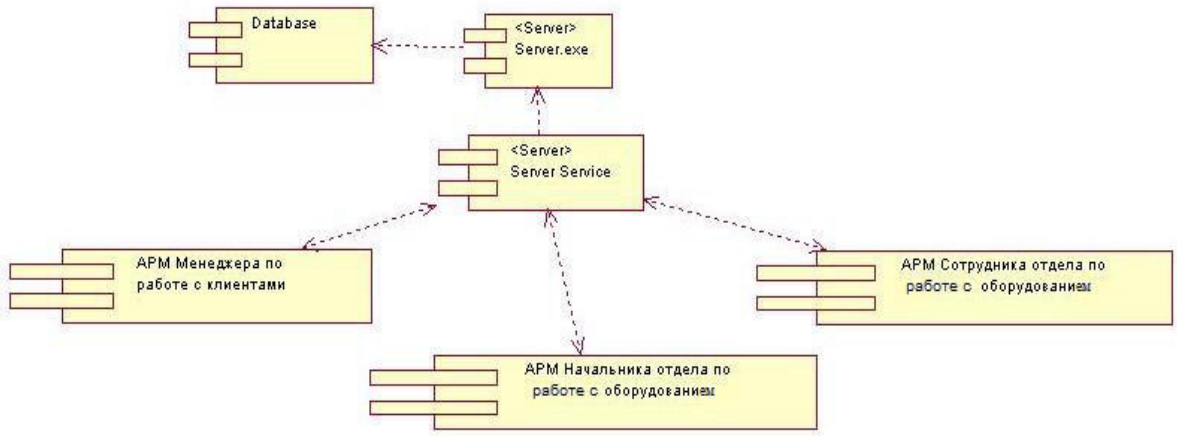


Рисунок 2.2 – Диаграмма компонентов информационной системы.

Блок-схема развертывания (иногда называемая кросс-функциональной блок-схемой) представляет собой инструмент отображения бизнес-процессов, используемый для определения этапов и заинтересованных сторон данного процесса.

«Блок-схемы развертывания состоят из последовательности действий, а также взаимодействия между людьми или группами». Каждый участник процесса отображается на карте (которая построена в виде матрицы) - задачи / действия затем последовательно формулируются в столбце, соответствующем этому заинтересованному лицу.

Поскольку потоковые диаграммы развертывания выделяют отношения между заинтересованными сторонами в дополнение к потоку процессов, они особенно полезны для выделения областей неэффективности, дублирования или ненужной обработки. Часто используемые в рамках Шесть сигма-операций, заполненные блок-схемы обычно используются для изучения интерфейсов между «участниками», которые обычно являются причинами задержек и других связанных с этим проблем. Потоковые диаграммы развертывания полезны для определения того, кому в организации требуется реализовать процесс, и иногда используются в качестве инструмента бизнес-планирования.

2.2 Проектирование пользовательского интерфейса

Пользовательский интерфейс или человеко-машинный интерфейс является частью машины, которая управляет взаимодействием человек-машина. Мембранные переключатели, резиновые клавиатуры и сенсорные экраны являются примерами физической части интерфейса человек-машина, которую мы можем увидеть и потрогать.

В сложных системах человеко-машинный интерфейс обычно компьютеризирован. Термин «человек-компьютерный интерфейс» относится к такого рода системе. В контексте вычислений этот термин обычно

распространяется и на программное обеспечение, предназначенное для управления физическими элементами, используемыми для взаимодействия человека с компьютером.

Проектирование интерфейсов человек-машина улучшается с учетом эргономики (человеческий фактор). Соответствующие дисциплины - инженерия человеческого фактора (HFE) и юзабилити (UE), которая является частью системной инженерии.

Инструменты, используемые для включения человеческого фактора в дизайн интерфейса, разработаны на основе знаний информатики, таких как компьютерная графика, операционные системы, языки программирования. В настоящее время мы используем графический пользовательский интерфейс выражения для человеко-машинного интерфейса на компьютерах, поскольку почти все они теперь используют графику.

Пользовательский интерфейс клиентской части приложения был выполнен в виде Windows-приложения.

Пользовательская среда информационной системы предприятия состоит из следующих частей:

– главное меню приложения;

– основная часть.

Главное меню приложения служит для доступа ко всем функциям системы.

Основная часть выполнена в виде панели с закладками, открываются формы, непосредственно с которыми работает пользователь.

Пользовательский интерфейс автоматически настраивается в соответствии с правами доступа текущего пользователя, то есть пользователю отображаются только доступные ему функции.

На рисунке 2.3 представлен прототип интерфейса входа в программу при помощи username и password.

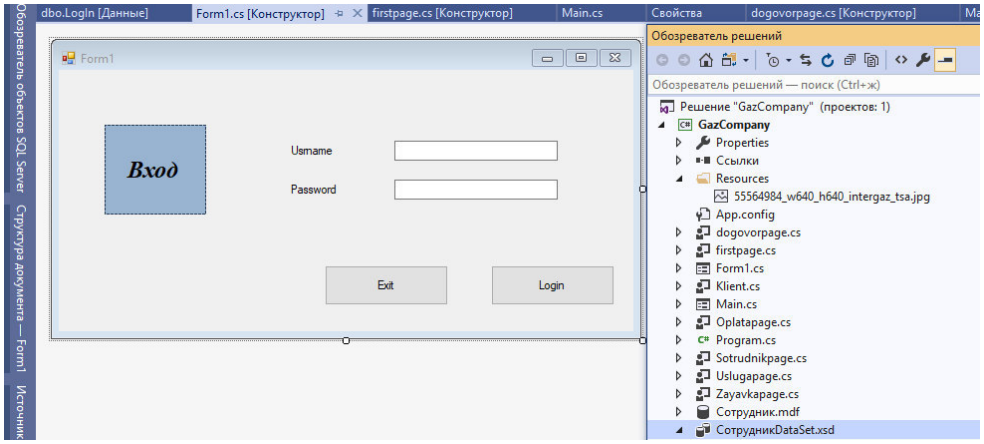


Рисунок 2.3 – прототип интерфейса входа в программу при помощи username и password.

Далее после создания страницы входа и главной страницы, выбираем функцию добавление нового элемента далее элементы Visual C#. В этом разделе выбираем пользовательский элемент управления. На рисунке 2.5 изображена форма dogovorpage.

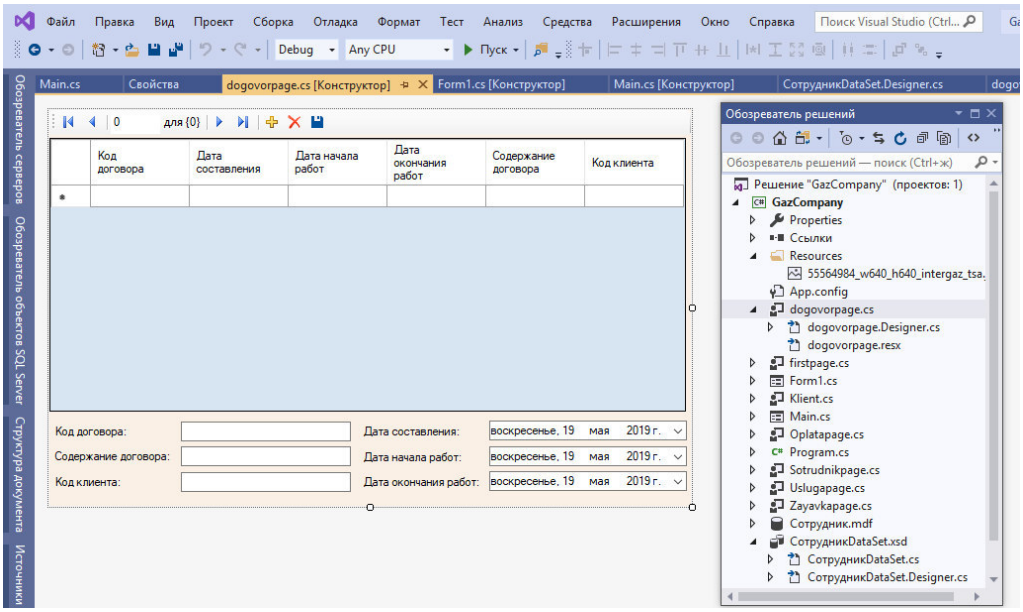


Рисунок 2.4 – Элемент управления - dogovorpage.

Данная процедура повторяется для остальных элементов управления. К ним относятся: firstpage, klient, oplatapage, sotrudnikpage, uslugapage и zayavkapage. Выше перечисленные элементы можно увидеть в обозревателе решений на рисунке 2.6.

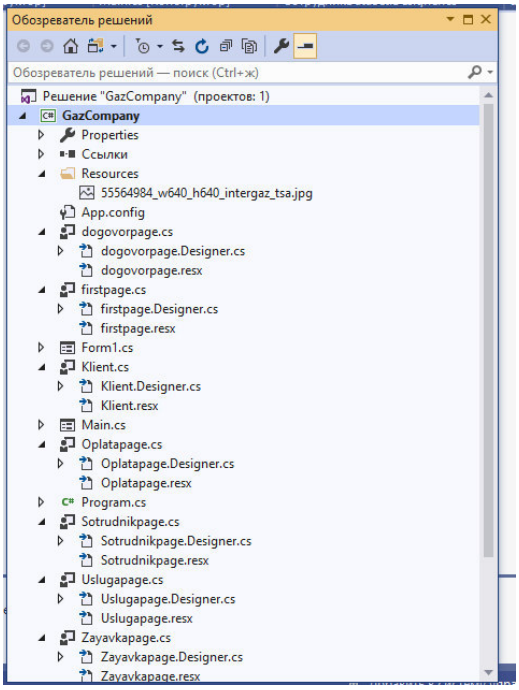


Рисунок 2.5 – Обозреватель решений.

На рисунке 2.6 отображена страница базы для регистрации пользователей в системе.

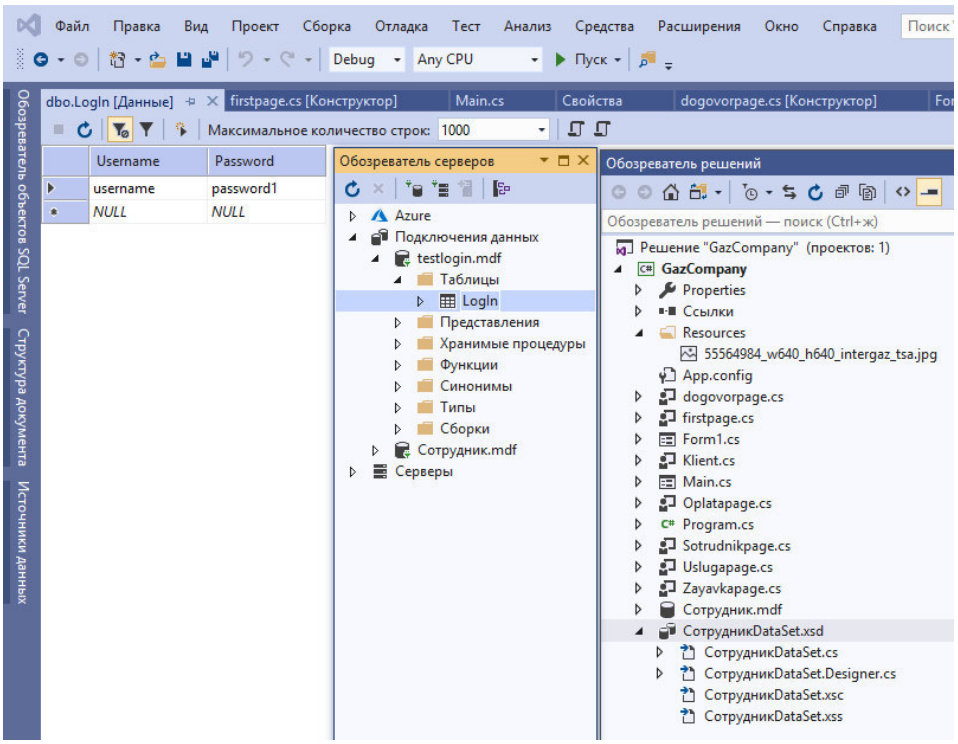


Рисунок 2.6 – База для регистрации новых пользователе в системе.

2.3 Проектирование базы данных

Основными целями проектирования базы данных являются:

– представление данных и связей между ними, которые необходимы для всех основных областей где будет применяться данное приложения и любых существующих групп его пользователей;

– создание модели данных, которая способна поддерживать выполнение любых требуемых транзакций обработки данных;

– разработка предварительного варианта проекта, структура которого позволит удовлетворить все основные требования, предъявляемые к производительности системы — например, ко времени реакции системы.

Проектирование базы данных - это организация данных в соответствии с моделью базы данных. Дизайнер определяет, какие данные должны храниться и как элементы данных взаимосвязаны. С помощью этой информации они могут начать подгонку данных к модели базы данных.

Проектирование базы данных включает в себя классификацию данных и выявление взаимосвязей. Это теоретическое представление данных называется онтологией. Онтология - теория, лежащая в основе дизайна базы данных.

Как только разработчик базы данных узнает о данных, которые должны храниться в базе данных, он должен определить, где находится зависимость в данных. Иногда при изменении данных вы можете изменять другие данные, которые не отображаются. Например, в списке имен и адресов, если предположить ситуацию, когда несколько человек могут иметь один и тот же адрес, но один человек не может иметь более одного адреса, адрес зависит от имени. Когда предоставлено имя и список, адрес может быть однозначно определен; однако обратное не выполняется - при наличии адреса и списка имя не может быть однозначно определено, поскольку по одному адресу могут находиться несколько человек. Поскольку адрес определяется именем, адрес считается зависимым от имени.

Концептуальное проектирование — это концепция чего-то, что сформулировано. Он включает в себя дизайн взаимодействий, процессов и стратегий. Он включает в себя: Распространенными артефактами концептуального дизайна являются концептуальные эскизы и модели.

Концептуальное проектирование базы данных: этот процесс требует модели данных всей информации в системе, которая должна быть произведена. Результирующая диаграмма отношений сущностей (ERDM) будет представлять реальный домен, который будет моделировать база данных, и эта модель будет проверена на избыточность и проверена, чтобы определить, соответствует ли она требованиям пользователя / бизнеса.

По окончании данного этапа получаем концептуальную модель, инвариантную к структуре базы данных. Часто она представляется в виде модели "сущность-связь".

Построение мощной и наглядной концептуальной схемы базы данных позволяет более полно оценить специфику моделируемой предметной области и избежать возможных ошибок на стадии проектирования схемы реляционной базы данных. На рисунке 2.7 пример концептуальной модели, на базе анализа сущностей.

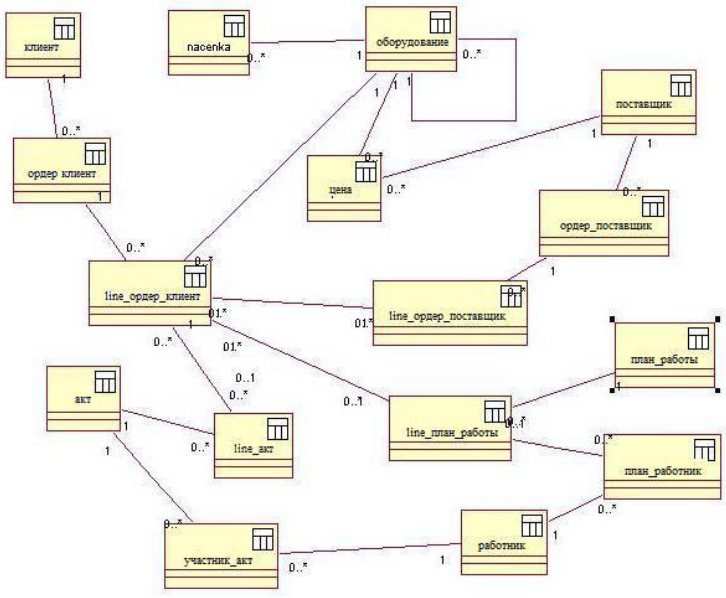


Рисунок 2.7 - Концептуальная модель данных

Логическое проектирование базы данных: этот процесс включает проектирование базы данных из ERDM. Зачастую это достигается процессом нормализации данных (обычно достаточно до 3-й нормальной формы) и моделированием сущностей для разработки таблиц и взаимосвязей, которые будут формировать структуру (или схему) базы данных. Этот этап не учитывает, какую реализацию парадигмы РСУБД, например, будут использоваться MySQL, SQL Server, DB2 и т. Д. Или на какой аппаратной платформе будет находиться база данных.

Физический дизайн базы данных. Физический дизайн базы данных определяет физическую конфигурацию базы данных в выбранной реализации СУБД. Этот шаг включает в себя описание базовых отношений, файловых организаций, индексов первичных или внешних ключей в таблицах, обеспечение определения ограничений сущности и ссылочной целостности, а также мер безопасности и т. д.

Физический уровень модели данных изображен на рисунке 2.8.

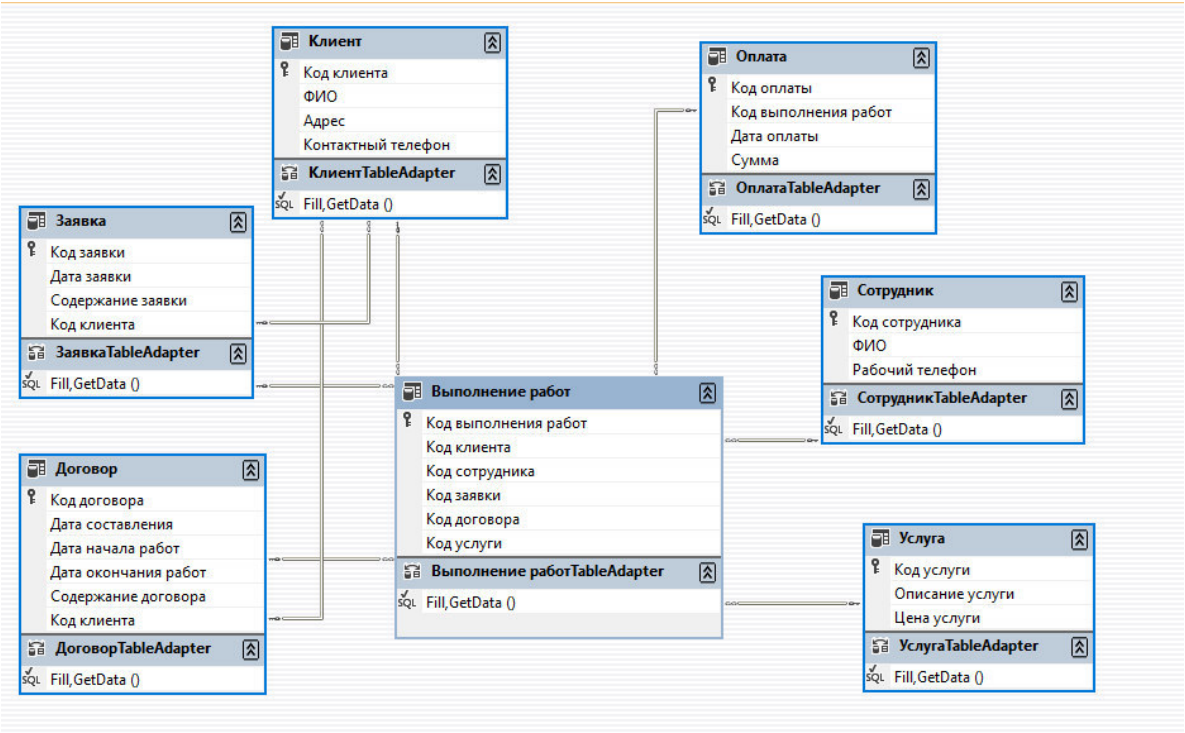


Рисунок 2.8 - Физическая модель данных

В процессе физического проектирования базы данных в среде Visual Studio была создана база данных is_enterprises, состоящая из файлов данных Сотрудник.mdf и файлов журналов транзакций Сотрудник.dataset.xsd.

Листинг скриптов, создающих информационные структуры сервера баз данных приведен в приложении В.

Перед тем как создавать физическую модель данных необходимо определиться с СУБД, в которой эту модель планируется реализовать.

Одним из требований заказчика является реализация информационного обеспечения на основе Visual Studio. Построенные на сильных сторонах программного продукта представляет собой интегрированное решение по управлению и анализу данных, которое поможет организациям различного масштаба:

– строительство, развертка и управление промышленными приложениями, которые являются более безопасными, масштабируемыми и надежными;

– увеличение продуктивности информационных технологий, за счет уменьшения сложности построения, развертывания и управления приложениями по работе с базами данных;

– разделять данные между платформами, приложениями и устройствами для облегчения соединения внутренних и внешних систем.

Преимущество Microsoft Visual Studio перед другими средствами проект ирования баз данных –

разнообразие всевозможных средств, позволяющих реализовать практичес ки любую задумку заказчика или программиста. Этому способствует большое количество встроенных в среду разработки элементов, начиная с кнопок и заканчивая средствами интеграции с файл ами баз данных, таких как Oracle, SQL Server, Access. Это означает, что можно создать базу данных в любой удобной среде моделирования.

При работе с базой данных в среде разработки Visual Studio есть в озможность не только задать все ограничения первичного ключа, но и во обще исключить пользователя из данного процесса. Для создания набора

таблиц удобно использовать встроенный элемент DataSet, поддерживающи й режим конструктора таблиц, в котором для столбца, содержащего перв ичный ключ (ПК), указаны необходимые свойства (таблица 2.1).

Таблица 2.1 - Свойства столбца с ПК

Свойство	Значение	Комментарий
AllowDBNull	False	Поле обязательное к заполнению
AutoIncrement	True	Автоматически заполнять поле
AutoIncrementSeed	1	Начальное значение для автозаполнения
AutoIncrementStep	1	Шаг автозаполнения
ReadOnly	True	Запрет на изменение пользователем
Unique	True	Все значения уникальны

Сумма всех этих свойств позволяет не указывать первичные ключи н а этапе моделирования базы данных, так как они реализуются программн о.

2.4 Обоснование выбора платформы создания информационной системы

В настоящее время обязательной возможностью считается визуальное проектирование, когда программист строит свои приложения, используя готовые модули. Примером могут служить все современные пакеты для разработчиков –Microsoft Visual Studio 2019, Python и т.д.

Чтобы средства разработки и технологии отвечали требованиям разработчиков, в корпорации Майкрософт была создана совершенно новая модель программирования для доступа к данным, основанная на .NET Framework. Построение на основе .NET Framework гарантирует единообразие доступа к данным: компоненты используют систему общих типов, общие шаблоны разработки и соглашения о пространствах имен.

.Net Framework - это платформа для разработки программного обеспечения, разработанная Microsoft. Фреймворк предназначался для создания приложений, которые будут работать на платформе Windows. Первая версия .Net Framework была выпущена в 2002 году.

Версия называлась .Net Framework 1.0. .Net Framework прошел долгий путь с тех пор, и текущая версия 4.7.1.

.Net Framework может использоваться для создания как приложений на основе форм, так и веб-приложений. Веб-сервисы также могут быть разработаны с использованием .Net Framework.

Фреймворк также поддерживает различные языки программирования, такие как Visual Basic и C #. Таким образом, разработчики могут выбирать и выбирать язык для разработки необходимого приложения. В этой главе вы познакомитесь с некоторыми основами .Net Framework.

Классы ADO.NET были разработаны для поддержки возможностей новой модели программирования: интеграции с XML, единого представления данных с возможностью комбинирования данных из различных источников, а также средств оптимизации взаимодействия с базой данных, представленных в .NET Framework.

Структура ADO.NET создана для решения задач современной модели разработки приложений. В то же время модель программирования по возможности приближена к ADO, что упрощает переход разработчиков ADO к новой среде. ADO.NET является неотъемлемой частью .NET Framework, оставаясь понятной программистам ADO.

.NET представляет собой совершенно новый способ создания распределенных настольных и встроенных приложений. Для типов .NET не нужны ни фабрики классов, ни поддержка Unknown, ни регистрация в системном реестре. Эти основные элементы COM не скрыты – их просто больше нет.

Специально для новой платформы Microsoft разработала новый язык программирования – C#, который впитал в себя многое из того лучшего, что

есть в самых разных языках программирования, и так же является составной частью Microsoft Visual Studio 2019.

Платформа .NET является полностью независимой от используемых языков программирования. Можно использовать несколько .NET-совместимых языков программирования даже в рамках одного проекта.

Основные возможности .NET следующие:

– полные возможности взаимодействия с существующим кодом;

– полное и абсолютное межъязыковое взаимодействие, межъязыковая обработка исключений и межъязыковая отладка;

– общая среда выполнения для любых приложений .NET, вне зависимости от того, на каких языках они были созданы. Один из важных моментов при этом – то, что для всех языков используется один и тот же набор встроенных типов данных;

– библиотека базовых классов, которая обеспечивает сокрытие всех сложностей, связанных с непосредственным использованием вызовов API, предлагает целостную объектную модель для всех языков программирования, поддерживающих .NET;

– отсутствует сложность COM;

– действительное упрощение процесса развертывания приложения.

В .NET нет необходимости регистрировать двойные типы в системном реестре. .NET позволяет разным версиям одного и того же модуля DLL мирно сосуществовать на одном компьютере.

Microsoft Visual Studio 2019 продолжает поддерживать технологии Microsoft .NET Framework уже в версии Microsoft .NET Framework SDK v2.0, которые предоставляют общеязыковую среду выполнения и унифицированные классы программирования. Также в Visual Studio включена библиотека MSDN, содержащая документацию по данным инструментам разработки.

Платформа Microsoft.NET для отображения данных на компьютере конечного пользователя и его интерактивного взаимодействия с системой. предоставляет класс System.Windows.Forms.Form и большое разнообразие классов элементов управления, дочерних от класса Control. Функциональность уровня представления во многом определяется составом элементов управления, входящих в коллекцию Controls для конкретной формы.

Уровень бизнес-логики отражает логику предметной области и реализует основные функции информационной системы. К таким функциям относятся вычисления на основе вводимых и хранимых данных, проверка элементов данных и обработка команд, поступающих от слоя представления, а также передача информации слою источника данных. Возможности, предоставляемые технологией Microsoft.NET, позволяют достаточно эффективно решать вопросы корректности ввода пользователем данных, и поэтому часть функций проверки элементов данных может быть решена на уровне представления.

Уровень бизнес-логики получает на вход информацию от уровня представления, проводит необходимые проверки и вычисления, сохраняет в информацию базе данных и возвращает уровню представления, определенные данные.

Бизнес-логика описывается набором методов, реализующих бизнес-транзакцию. Для платформы Microsoft.NET это типовое решение сценарий транзакций использует прямой доступ к базе данных и базируется на использовании объектов классов DataCommand и DataReader технологии ADO.NET, а также используя bindingSource, TableAdapter, DataSet. Класс, реализующий сценарий транзакций, обеспечивает прямой доступ к источнику данных и необходимую функциональность бизнес-логики. Для данного типового решения все обязанности по реализации бизнес-логики возлагаются на методы сценария транзакций.

Для разрабатываемой информационной системы выбрана платформа Microsoft Visual Studio 2019. В качестве языка реализации приложения выбран C#.

2.5 Выводы к разделу

Во втором разделе выполнено проектирование информационной системы. Разработана архитектурная модель проектирования и компонентная модель.

В качестве СУБД обосновано применение Microsoft Visual Studio 2019. Для разрабатываемой информационной системы была выбрана платформа Microsoft Visual Studio 2019. В качестве языка реализации приложения выбран C#.

3 Реализация и аттестация информационной системы

3.1 Реализация приложения

При проектировании системы используется концепция слоев – одна из общеупотребительных моделей, применяемая разработчиками программного обеспечения для разделения сложных систем на более простые части.

ADO.NET обеспечивает согласованный доступ к источникам данных, таким как SQL Server и XML, а также к источникам данных, предоставляемым через OLE DB и ODBC. Пользовательские приложения для совместного использования данных могут использовать ADO.NET для подключения к этим источникам данных, а также для получения, обработки и обновления, содержащихся в них данных.

ADO.NET разделяет доступ к данным от манипулирования данными на отдельные компоненты, которые можно использовать отдельно или в тандеме. ADO.NET включает поставщиков данных .NET Framework для подключения к базе данных, выполнения команд и получения результатов. Эти результаты либо обрабатываются напрямую, либо помещаются в объект ADO.NET DataSet, чтобы быть доступными для пользователя специальным образом, в сочетании с данными из нескольких источников, либо передаются между уровнями. Объект DataSet также может использоваться независимо от поставщика данных .NET Framework для управления данными, локальными для приложения или полученными из XML.

Классы ADO.NET находятся в System.Data.dll и интегрированы с классами XML в System.Xml.dll. Пример кода, который подключается к базе данных, получает данные из нее и затем отображает эти данные в окне консоли, см. В разделе Примеры кода ADO.NET.

ADO.NET предоставляет функциональность разработчикам, которые пишут управляемый код, аналогично функциональности, предоставленной разработчикам объектной модели компонентов (COM) собственными силами объектами данных ActiveX (ADO). Мы рекомендуем вам использовать ADO.NET, а не ADO, для доступа к данным в ваших приложениях .NET.

ADO.NET предоставляет наиболее прямой метод доступа к данным в .NET Framework. Для абстракции более высокого уровня, которая позволяет приложениям работать с концептуальной моделью вместо базовой модели хранения, см. ADO.NET Entity Framework. Реализация программного обеспечения – это процесс перевода системной спецификации в работоспособную систему. Разработка приложений подсистемы

осуществлялась на языке C# с использованием платформы .Net Framework и входящих в нее библиотек.

Итогом реализации приложения является работоспособная информационная система. Разрабатываемая информационная система будет являться приложением клиент серверного типа. Информационная система будет взаимодействовать с серверной базой данных, и состоять из двух частей. Первая часть приложения, реализующая интерфейс пользователя и находящаяся на клиентской рабочей станции. А вторая часть приложения будет отвечать за хранение, и обработка данных осуществляется на стороне сервера.

Для реализации взаимодействия клиента и сервера необходимо реализовать функции загрузки и отображения данных из базы данных, и пересылка данных в базу данных с последующим сохранением данных в базе. Для работы с данными, а так же с базами данных, используются следующие пространства имен изображенное на рисунке 3.1

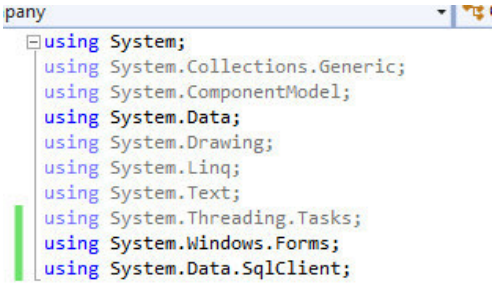


Рисунок 3.1 – Пространство имен

Входящий в состав Microsoft .NET Framework SDK v2.0, в данном проекте использовалось следующее пространственное имя для подключения к базе данных (рисунок 3.2)

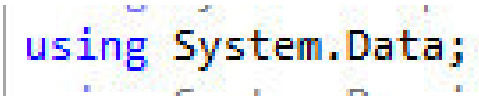
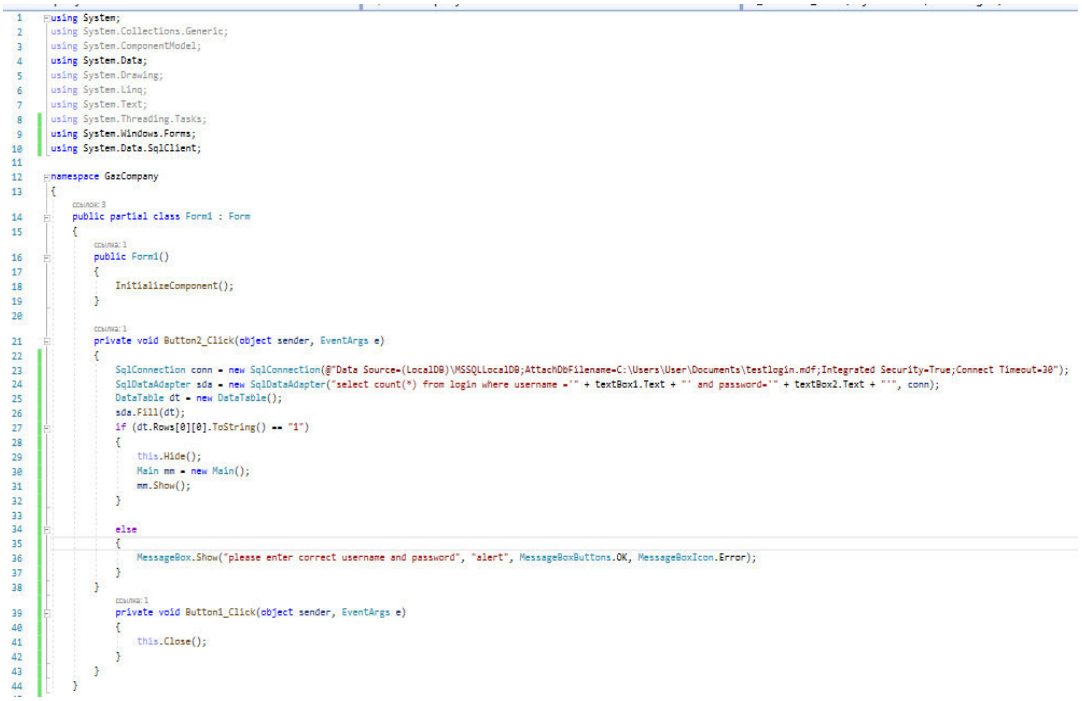


Рисунок 3.2 – Пространственное имя для подключения к базе данных

Разработка форм осуществляется с использованием специализированных мастеров Visual Studio .NET. Интегрированная среда разработки Visual Studio .NET позволяет создавать элементы в режиме визуальной разработки, где можно перетаскивать элементы прямо на форму.

В результате работы мастера проекта реализуется каркас формы являющийся экземпляром классов, унаследованного от System.Windows.Forms.Form. При отображении формы во время выполнения программы, этот класс будет использоваться как шаблон для отображения окна. Файлы C# имеют расширение «.cs». Код главной формы изображен на

рисунке 3.3. Данная форма является диспетчерской, запускающей дочерние окна данного модуля.

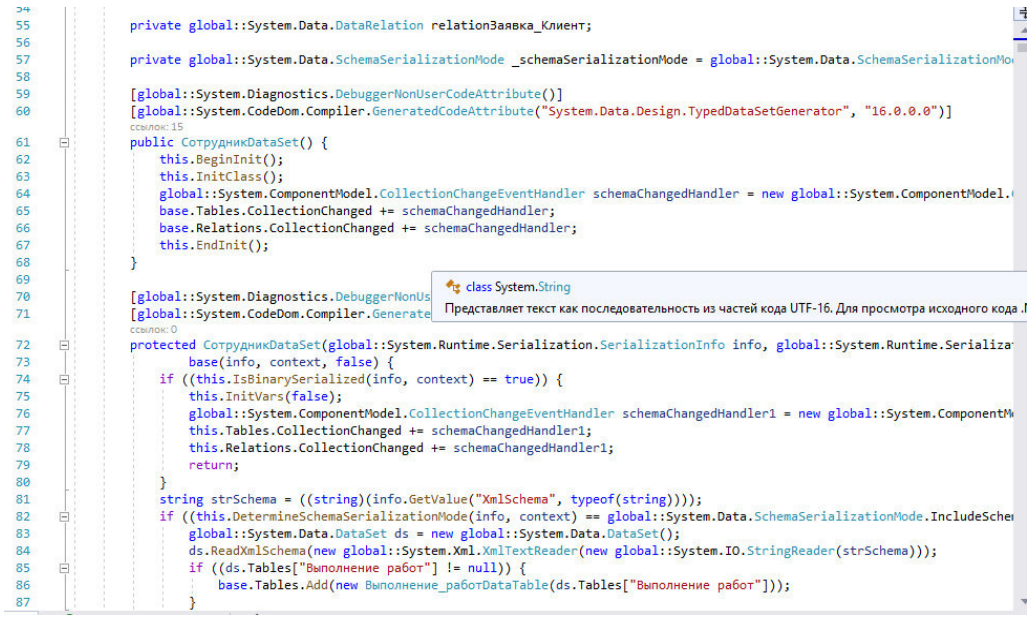


```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace GazCompany
{
    ссылок: 3
    public partial class Form1 : Form
    {
        ссылка: 1
        public Form1()
        {
            InitializeComponent();
        }

        ссылка: 1
        private void Button2_Click(object sender, EventArgs e)
        {
            SqlConnection conn = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\User\Documents\testlogin.mdf;Integrated Security=True;Connect Timeout=30");
            SqlDataAdapter sda = new SqlDataAdapter("select count(*) from login where username ='" + textBox1.Text + "' and password='" + textBox2.Text + "'", conn);
            DataTable dt = new DataTable();
            sda.Fill(dt);
            if (dt.Rows[0][0].ToString() == "1")
            {
                this.Hide();
                Main mm = new Main();
                mm.Show();
            }

            else
            {
                MessageBox.Show("please enter correct username and password", "alert", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        ссылка: 1
        private void Button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Рисунок 3.3 – Инициализация компонентов формы «Start.cs»

3.2 Взаимодействие приложения с источниками данных

Для компонентов проектируемой системы источниками данных являются с одной стороны соответствующая таблица из базы данных, а с другой данные передаваемые клиентом в компонент, позволяющие определить адрес базы данных, с которой происходит взаимодействие компонента (рисунок 3.4)



```
    private global::System.Data.DataRelation relationЗаявка_Клиент;

    private global::System.Data.SchemaSerializationMode _schemaSerializationMode = global::System.Data.SchemaSerializationMo

    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "16.0.0.0")]
    ссылок: 15
    public СотрудникDataSet() {
        this.BeginInit();
        this.InitClass();
        global::System.ComponentModel.CollectionChangeEventHandler schemaChangedHandler = new global::System.ComponentModel.
        base.Tables.CollectionChanged += schemaChangedHandler;
        base.Relations.CollectionChanged += schemaChangedHandler;
        this.EndInit();
    }

    [global::System.Diagnostics.DebuggerNonUs
    [global::System.CodeDom.Compiler.Generate
    ссылок: 0
    protected СотрудникDataSet(global::System.Runtime.Serialization.SerializationInfo info, global::System.Runtime.Serializa
            base(info, context, false) {
        if ((this.IsBinarySerialized(info, context) == true)) {
            this.InitVars(false);
            global::System.ComponentModel.CollectionChangeEventHandler schemaChangedHandler1 = new global::System.ComponentM
            this.Tables.CollectionChanged += schemaChangedHandler1;
            this.Relations.CollectionChanged += schemaChangedHandler1;
            return;
        }
        string strSchema = ((string)(info.GetValue("XmlSchema", typeof(string))));
        if ((this.DetermineSchemaSerializationMode(info, context) == global::System.Data.SchemaSerializationMode.IncludeSche
            global::System.Data.DataSet ds = new global::System.Data.DataSet();
            ds.ReadXmlSchema(new global::System.Xml.XmlTextReader(new global::System.IO.StringReader(strSchema)));
            if ((ds.Tables["Выполнение работ"] != null)) {
                base.Tables.Add(new Выполнение_работDataTable(ds.Tables["Выполнение работ"]));
            }
```

class System.String
Представляет текст как последовательность из частей кода UTF-16. Для просмотра исходного кода .

Рисунок 3.4 – Пример обращения к базе данных

SQL используется для реализации всех функциональных возможностей, которые СУБД предоставляют пользователю:

– организация данных;
– выборка данных;
– обработка данных;
– совместное использование данных;
– управление доступом;
– целостность данных.

Источники данных обеспечивают связь между приложениями и реляционными базами данных. Если вы настроили сетевое окружение.

Заявки на получение реляционной базы данных. Фабрика соединений java EE Connector Architecture (JCA), которая обеспечивает связь для EIS.

Поставщик подключения к базе данных Java (JDBC) - это связанный с ним источник данных. Можно использовать приложение. Это обеспечивает пул данных.

Вы можете создать несколько источников данных для провайдера JDBC. Например, вы можете использовать приложение базы данных. В IBM® Business Process Manager поставщики JDBC должны реализовать один или оба из следующих интерфейсов источников данных. Протокол однофазной или двухфазной транзакции. На рисунке 3.5 представлен sql-код перехода страниц

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GazCompany
{
    ссылок: 4
    public partial class Main : Form
    {
        ссылка: 1
        public Main()
        {
            InitializeComponent();
            SidePanel.Height = button7.Height;
            SidePanel.Top = button7.Top;
            firstpage1.BringToFront();
        }

        ссылка: 1
        private void Button7_Click(object sender, EventArgs e)
        {
            SidePanel.Height = button7.Height;
            SidePanel.Top = button7.Top;
            firstpage1.BringToFront();
        }

        ссылка: 1
        private void Button4_Click(object sender, EventArgs e)
        {
            SidePanel.Height = button4.Height;
```

Рисунок 3.5 - Пример sql-код

Использование хранимых процедур дает несколько очевидных преимуществ:

– сценарии выполнения хранимых процедур кэшируются на сервере, что дает ощутимый прирост в скорости при повторном вызове процедур;

– дополнительный уровень абстракции – дает возможность изменить логику работы хранимой процедуры без необходимости вносить изменения в приложение (но при этом нельзя изменять сигнатуру хранимой процедуры);

– часть бизнес-логики приложения выполняется на сервере;

– хранимые процедуры могут возвращать не только один набор результатов или, проще говоря, таблицу, но и значения выходных параметров и даже несколько наборов результатов (несколько таблиц) за один вызов.

Для построения некоторых отчетов требуется несколько таблиц. На рисунке 3.6 исходный код

```
254         [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
255         [global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "16.0.0.0")]
            ссылки
256         protected override void ReadXmlSerializable(global::System.Xml.XmlReader reader) {
257             if ((this.DetermineSchemaSerializationMode(reader) == global::System.Data.SchemaSerializationMode.IncludeSchema)) {
258                 this.Reset();
259                 global::System.Data.DataSet ds = new global::System.Data.DataSet();
260                 ds.ReadXml(reader);
261                 if ((ds.Tables["Выполнение работ"] != null)) {
262                     base.Tables.Add(new Выполнение_работDataTable(ds.Tables["Выполнение работ"]));
263                 }
264                 if ((ds.Tables["Договор"] != null)) {
265                     base.Tables.Add(new ДоговорDataTable(ds.Tables["Договор"]));
266                 }
267                 if ((ds.Tables["Заявка"] != null)) {
268                     base.Tables.Add(new ЗаявкаDataTable(ds.Tables["Заявка"]));
269                 }
270                 if ((ds.Tables["Клиент"] != null)) {
271                     base.Tables.Add(new КлиентDataTable(ds.Tables["Клиент"]));
272                 }
273                 if ((ds.Tables["Оплата"] != null)) {
274                     base.Tables.Add(new ОплатаDataTable(ds.Tables["Оплата"]));
275                 }
276                 if ((ds.Tables["Сотрудник"] != null)) {
277                     base.Tables.Add(new СотрудникDataTable(ds.Tables["Сотрудник"]));
278                 }
279                 if ((ds.Tables["Услуга"] != null)) {
280                     base.Tables.Add(new УслугаDataTable(ds.Tables["Услуга"]));
281                 }
282                 this.DataSetName = ds.DataSetName;
283                 this.Prefix = ds.Prefix;
284                 this.Namespace = ds.Namespace;
285                 this.Locale = ds.Locale;
286                 this.CaseSensitive = ds.CaseSensitive;
287                 this.EnforceConstraints = ds.EnforceConstraints;
288                 this.Merge(ds, false, global::System.Data.MissingSchemaAction.Add);
```

Проблемы не найдены

Рисунок 3.6 – Исходный код

При проектировании блока взаимодействия с базой данных был использован подход, который заключается в следующем. Формируется класс MsSqlDataAdapter. SqlDataAdapter служит мостом между DataSet и SQL Server для извлечения и сохранения данных. SqlDataAdapter обеспечивает этот мост, сопоставляя Fill, которая изменяет данные в DataSet, чтобы соответствовать данным в источнике данных, и Update, который изменяет данные в источнике данных, чтобы соответствовать данным в DataSet, используя соответствующий Transact-SQL заявления против источника данных. Обновление выполняется построчно. Для каждой вставленной, измененной и удаленной строки метод Update определяет тип изменения, которое было выполнено для нее (вставить, обновить или удалить). В зависимости от типа изменения шаблон команды «Вставить», «Обновить» или «Удалить» выполняется для распространения измененной строки в источник данных. Когда SqlDataAdapter заполняет DataSet, он создает необходимые таблицы и столбцы для возвращаемых данных, если они еще не существуют. Однако информация первичного ключа не включается в неявно созданную схему, если для свойства MissingSchemaAction не установлено значение AddWithKey. Вы также можете попросить SqlDataAdapter создать схему DataSet, включая информацию о первичном ключе, перед заполнением ее данными с помощью

FillSchema. Для получения дополнительной информации см. Добавление существующих ограничений в DataSet.

SqlDataAdapter используется вместе с SqlConnection и SqlCommand для повышения производительности при подключении к базе данных SQL Server.

Класс UserGateWay содержит специализированные функции Логины шлюза - это идентификаторы MemberGroupTrader используется для входа в TT Gateways. Вы можете назначить уникальные номера счетов и ограничения риска для каждого входа в шлюз, и вы можете назначить несколько логин шлюза для одного пользователя.

На рисунке 3.7 изображен код страницы договора.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GazCompany
{
    ссылок: 5
    public partial class dogovorpage : UserControl
    {
        ссылка: 1
        public dogovorpage()
        {
            InitializeComponent();
        }

        ссылка: 1
        private void ДоговорBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.договорBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.сотрудникDataSet);

        }
    }
}
```

Рисунок 3.7 – Код страницы договора

Функции CreateConnection() , OpenConnection(), CloseConnection() создают, открывают и закрывают соединение с сервером соответственно. Функции CreateCommand(), CreateParametrizedCommand() предназначены для создания параметризованных и обычных sql-команд. Для параметризованной команды необходимы параметры, которые создаются функциями CreateSqlParameter().

3.3 Тестирование приложения

Полностью избежать ошибок при программировании невозможно, даже профессионалы время от времени допускают ошибки, а также недоработки системы, проявляющиеся во время тестирования, а иногда и эксплуатации.

Тестирование приложений определяется как тип тестирования программного обеспечения, проводимый с помощью сценариев с целью поиска ошибок в программном обеспечении. Он имеет дело с тестами для всего приложения.

Это помогает повысить качество ваших приложений при одновременном снижении затрат, максимизации рентабельности инвестиций и экономии времени на разработку.

В программной инженерии тестирование приложений может выполняться в различных категориях, таких как GUI, функциональность, база данных (бэкэнд), нагрузочное тестирование и т. д.

Для тестирования приложений жизненные циклы тестирования включают в себя различные этапы, которые включают анализ требований, планирование тестов, анализ тестов, дизайн тестов, выполнение тестов и отчеты об ошибках и т. д.

Программные приложения и продукты имеют ряд вариаций с точки зрения функций, которые они поддерживают, а также процессов, которые они реализуют. Таким образом, Тестирование приложений гарантирует, что определенная программа или приложение функционируют должным образом.

Жизненный цикл тестирования приложений состоит из четырех этапов.

– разработка планов испытаний на основе требований приложений

– разрабатывайте тестовые сценарии вручную и автоматизированные сценарии тестирования

– выполнить функциональные тесты для проверки требований приложения

– выполнить нагрузочные тесты и настроить производительность приложения

Документ плана тестирования получен из описания продукта, спецификации требований к программному обеспечению SRS или документов прецедентов. Целью теста является то, что тестировать, как тестировать, когда тестировать и кто будет тестировать. Документ плана тестирования используется в качестве средства связи между группой тестирования и менеджерами по тестированию. Из существующих способов тестирования был выбран «черный ящик». Этот способ является одним из наиболее устоявшихся способов обеспечения качества разработки программного обеспечения и входит в набор эффективных средств современной системы обеспечения качества программного продукта. Процесс тестирования программных продуктов обеспечивает получение актуальной информации о статусе проекта разработки информационной системы или приложения в разрезе требования функциональность. Тестирование позволяет сделать процесс разработки информационной системы и программного обеспечения прозрачным и управляемым для всех участников проекта. Разработчикам тестирование дает уверенность в верном понимании задач, которые ставит перед ними заказчик. Проектным менеджерам тестирование дает понимание эволюции проекта, проблемных мест в процессе разработки, а также информацию для принятия оперативных решений о готовности продукта или его версии к продуктивной эксплуатации, продажам и т.д.

Для тестирования разрабатываемого проекта была выбрана методика тестирования «черного ящика». BLACK BOX TESTING, также известный как Поведенческое Тестирование, является методом тестирования программного обеспечения, в котором внутренняя структура / дизайн / реализация тестируемого элемента не известны тестировщику. Эти тесты могут быть

функциональными или нефункциональными, хотя обычно они функциональные.

Этот метод назван так, потому что программа, по мнению тестера, похожа на черный ящик; внутри которого никто не видит. Этот метод пытается найти ошибки в следующих категориях:

– неверные или отсутствующие функции
– ошибки интерфейса
– ошибки в структурах данных или доступ к внешней базе данных
– поведение или ошибки производительности
– ошибки инициализации и завершения
– определение по istqb.

Тестирование черного ящика: тестирование, функциональное или нефункциональное, без ссылки на внутреннюю структуру компонента или системы.

Методика разработки теста черного ящика: процедура для выведения и / или выбора тестовых случаев на основе анализа спецификации, функциональной или нефункциональной, компонента или системы без ссылки на их внутреннюю структуру.

Ниже приведены некоторые методы, которые можно использовать для разработки тестов черного ящика.

Разделение эквивалентности: это метод проектирования программного тестирования, который включает в себя разделение входных значений на допустимые и недействительные разделы и выбор репрезентативных значений из каждого раздела в качестве тестовых данных.

Анализ граничных значений: это метод проектирования программного тестирования, который включает определение границ для входных значений и выбор значений, которые находятся на границах и только внутри / снаружи границ в качестве тестовых данных.

Диаграмма причинно-следственных связей. Это методика разработки программных тестов, которая включает в себя определение случаев (входных условий) и эффектов (выходных условий), создание графика причинно-следственных связей и, соответственно, создание тестовых случаев.

Вы когда-нибудь задумывались, почему предсказатель закрывает глаза, предсказывая события? То же самое можно сказать и о тестировании черного ящика.

Таблица 3.1 наращиваемый подход к тестированию

Действие	Ожидаемый результат	Реальный результат
Ввести информацию о сотрудниках	Внесение информации о сотрудниках	Внесенная информация о сотрудниках
Сформировать заявку	Сформированная заявка	Сформированная заявка
Внесение изменений	Заявка с изменениями	Заявка с изменениями

о заявке		

Продолжение таблицы 3.1

Утверждение заявки	Утвержденная заявка	Утвержденная заявка
Составить план работ	Составленный план работ	Составленный план работ
Утвердить план работ	Утвержденный план работ	Утвержденный план работ
Составить акт о выполненной работе	Составленный акт	Составленный акт
Сформировать прайс	Сформированный прайс	Сформированный прайс

3.4 Методика развертывания приложения

Развертывание компонентов происходит на тех компьютерах системы, на которых расположены рабочие места пользователей, использующих бизнес процессы, связанные с данным компонентом.

Как было уже сказано, данная система разрабатывается для трех видов пользователей. Для данных пользователей разрабатываются соответствующие рабочие места. Требования к операционной системе для всех АРМ одинаковы, а модули для каждого АРМ будут установлены соответствующие сущностям.

Цели этапа развертывания:

– перенести решение в промышленную среду;

– признание заказчиком факта завершения проекта.

Развертывание компонентов, характерных для конкретного места установки, состоит из нескольких стадий: подготовки, установки, обучения и формального одобрения.

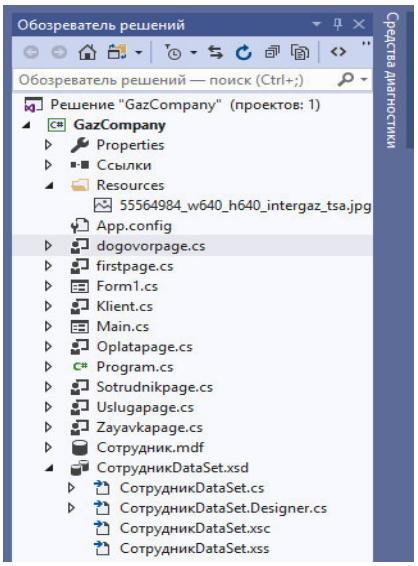


Рисунок 3.9 – Рабочее пространство программы.

Результатами этапа развертывания системы являются системы сопровождения и поддержки, хранилище документов, где размещаются все версии документов и кода, разработанных в течение проекта.

После входа в систему пользователю предоставлена главная страница, с которой он может открывать те базы, которые необходимы. На рисунке 3.8 представлена страница выполнение работ, в которой главный менеджер может просмотреть присвоенные коды клиентов, сотрудников, заявок, услуг и договоров.

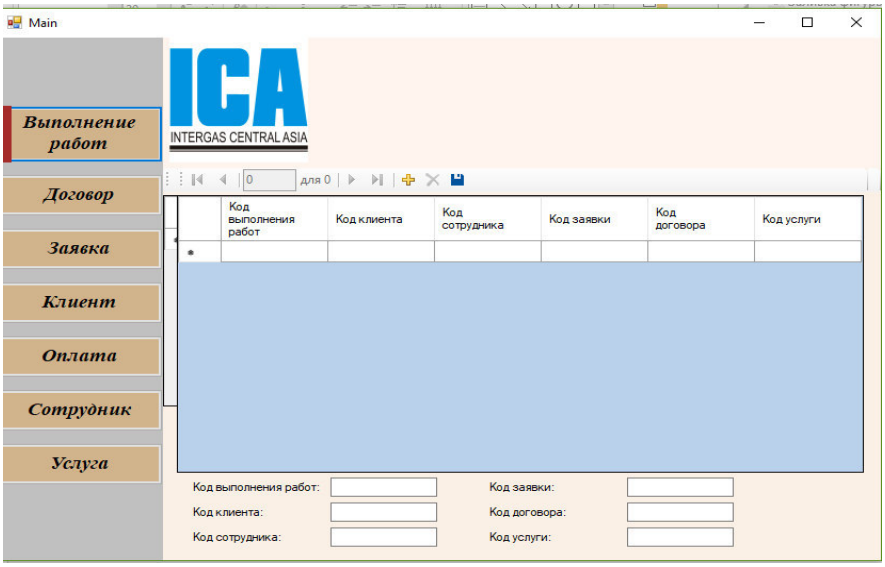


Рисунок 3.8 – Страница выполнение работ.

На рисунке 3.9 можно увидеть страницу договоров. В ней отображаются следующие данные код договора, дата составления, дата начала работ и окончания работ, содержание договора и код клиента.

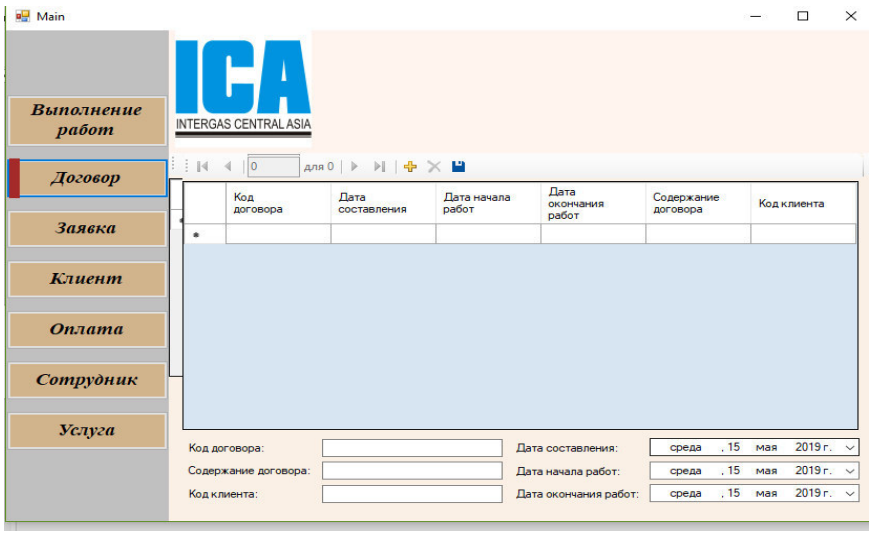


Рисунок 3.9 – Страница договор.

На рисунке 3.10 показана страница заявок, в которой отображены код заявки, дата заявки, содержание заявки, и код клиента.

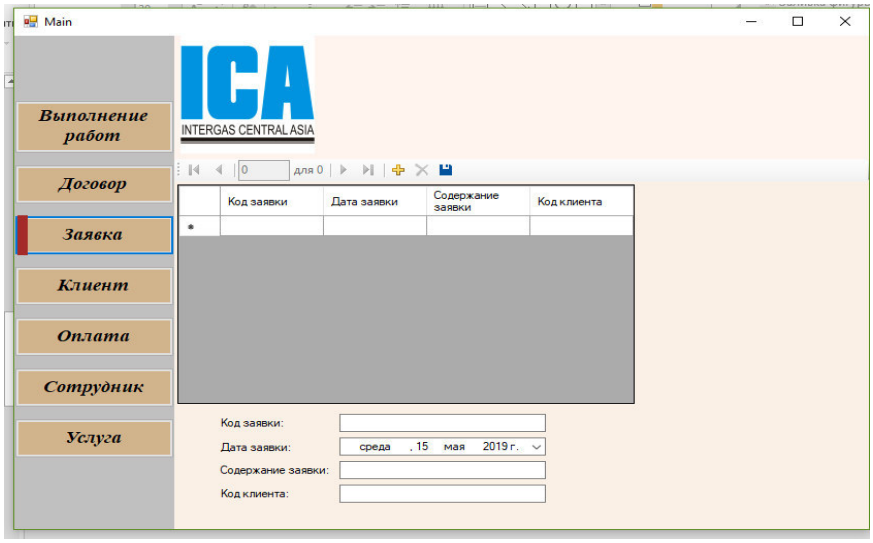


Рисунок 3.10 – Страница заявка.

На рисунке 3.11 отображены клиенты и данные используемые для данной базы. К ней относятся следующие поля: код клиента, ФИО, адрес и контактный телефон.

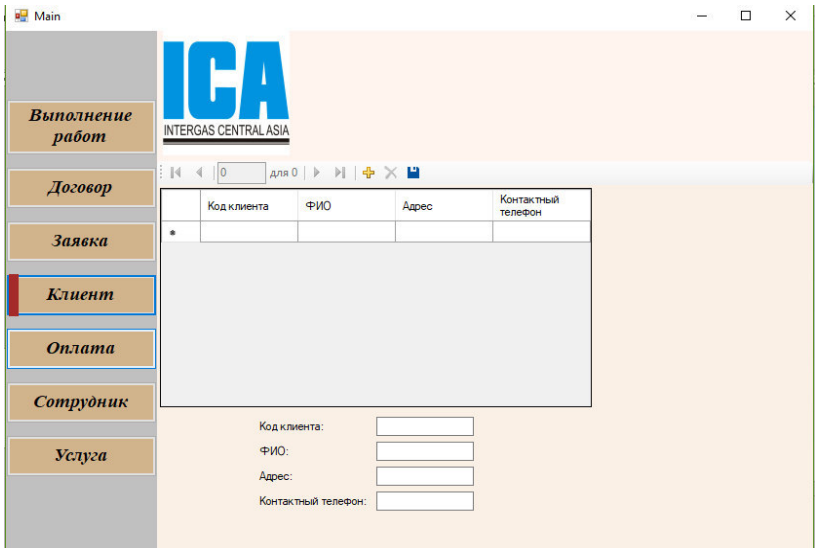


Рисунок 3.11 – Страница клиент.

Рисунок 3.12 отображает данные оплаты, такие как код оплаты, код выполнения работ, дата оплаты и сумма.

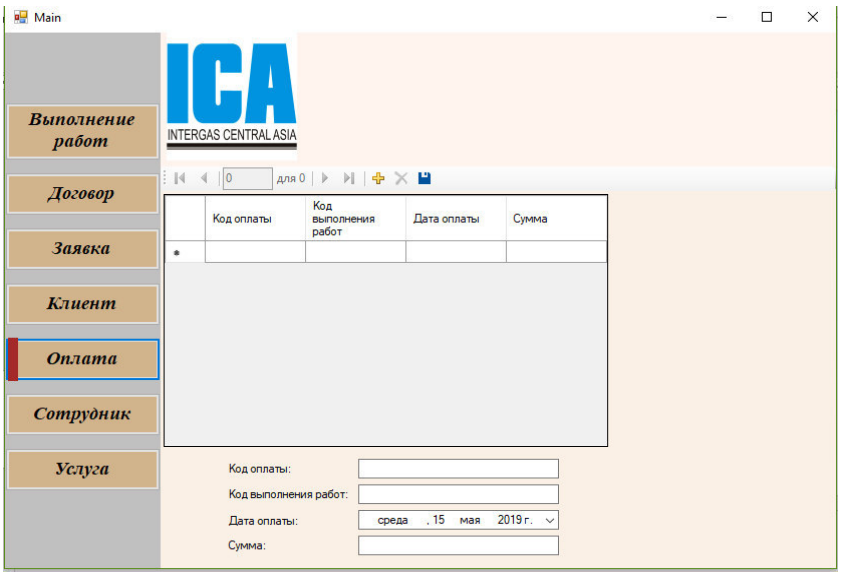


Рисунок 3.12 – Страница оплата.

Рисунок 3.13 показывает информацию о сотрудниках. В таблице указаны такие данные как код сотрудника, ФИО и рабочий телефон.

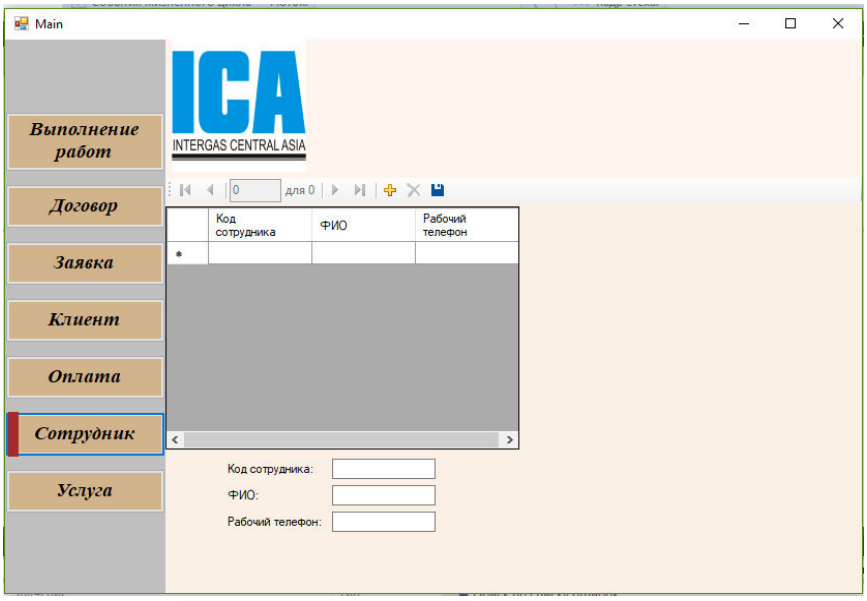


Рисунок 3.13 – Страница сотрудник.

Рисунок 3.14 отображает данные о б услугах. Таблица состоит из следующих данных: код услуги, описание услуги и цена услуги.

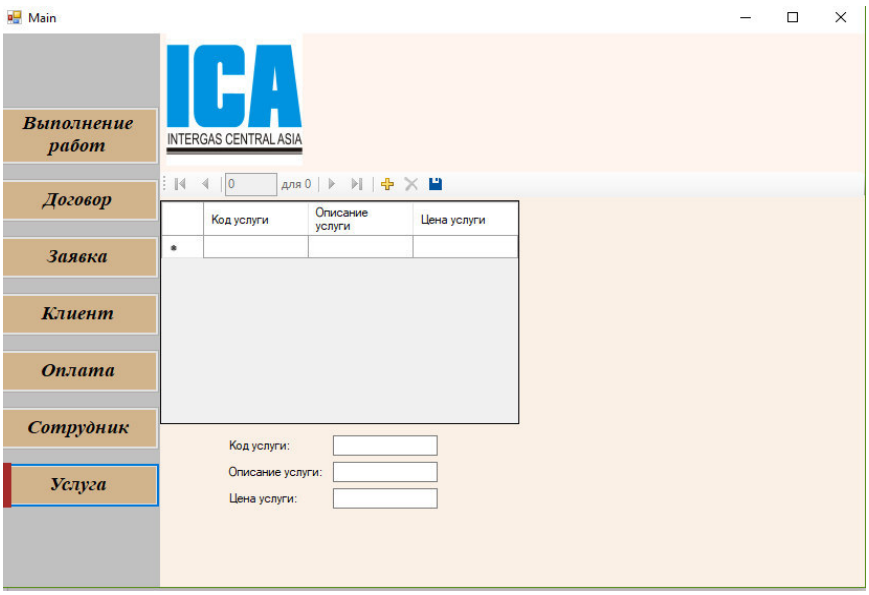


Рисунок 3.14 – Страница услуга.

4 Экономическое обоснование разработки дипломного проекта

4.1 Трудоемкость разработки программного продукта

Тема дипломного проекта – «разработка информационный системы для газового предприятия АО Intergaz».

Целью экономического обоснования дипломного проекта является расчет и анализ затрат, которые необходимы для создания и реализации информационной системы для газового предприятия АО Intergaz, изучить затраты на этапах проектирования, разработки, внедрения и функционирования ИС; определить условия и сроки окупаемости затрат. Обосновать рациональность проектирования данной ИС.

В данной главе производится экономический расчет коммерческой стоимости исследования проекта. Расчеты учитывают расходы на создание и внедрение информационной системы.

Для реализации проекта необходимы финансовые, материальные и трудовые ресурсы;

План проектирования и реализации проекта предусматривает 3 этапа в течении 2 месяцев (8 недель).

Первый этап: определение задачи; проектирование ИС; проектирование БД;

Второй этап: разработка ИС.

Третий этап: тестирование; откладка системы; внедрение.

Таблица 4.1 - распределение работ по этапам и видам и оценка их трудоемкости

Этап разработки ПП	Вид работы на данном этапе	Трудоемкость разработки ПП, чел.× ч.
Первый этап	Определение задачи; Проектирование ИС; Проектирование БД;	1×16 1×36 1×28
Второй этап	Разработка ИС	1×204
Третий этап	Тестирование; Откладка системы; Внедрение.	1×24 1×40 1×24
ИТОГО трудоемкость выполнения дипломной работы		372

4.2 Расчет затрат на разработку ПП

Определение затрат на разработку ПП производится путем составления соответствующей сметы, которая включает следующие статьи:

1) Материальные затраты.

2) Затраты на оплату труда сотрудников компании.

3) Социальный налог.

4) Амортизация основных фондов.

5) Прочие затраты.

В статью «Материальные затраты» включаются затраты на основные и вспомогательные материалы (бумага, картриджи и другие), энергию, необходимые для разработки ПП.

Таблица 4.2 - Затраты на материальные ресурсы

Наименование материального ресурса	Единица	Количество израсходованного материала	Цена за единицу, тг	Сумма, тг
Записная книга	шт.	1	320	320
Тонер для принтера	шт.	1	2000	2000
Бумага А4	пачка	1	1100	1100
Флеш карта	шт.	1	2500	2500
ИТОГО затраты на материальные ресурсы				5920

Общая сумма затрат на материальные ресурсы (ЗМ) определяем по формуле:

$$З_м = \sum_{i=1}^{n} Р_i \times Ц_i, \quad (4.1)$$

где Pi - расход i-го вида материального ресурса, натуральные единицы;

Цi - цена за единицу i-го вида материального ресурса, тг;

i - вид материального ресурса;

n - количество видов материальных ресурсов.

Затраты на электроэнергию рассчитываются по форме, приведенной в таблице 4.3.

Общая сумма затрат на электроэнергию ($З_Э$) рассчитывается по формуле 4.2:

$$З_э = \sum_{i=1}^{n} М_i * К_i * Т_i * Ц \quad (4.2)$$

где $М_i$ - паспортная мощность i-го электрооборудования, кВт;

$К_i$ - коэффициент использования мощности i-го электрооборудования (принимается $К_i$=0.7¸0.9);

$Т_i$ - время работы i-го оборудования за весь период разработки ПП ч;

Ц - цена электроэнергии, тг/кВт×ч;

i - вид электрооборудования;

n - количество электрооборудования.

$З_{э1} = 0{,}030 \cdot 0{,}7 \cdot 5 \cdot 16{,}25 = 1{,}71$ тг ≈2 тг.
$З_{э2} = 0{,}015 \cdot 0{,}7 \cdot 300 \cdot 16{,}25 = 51{,}19$ тг ≈ 51 тг.
$З_{э3} = 0{,}299 \cdot 0{,}7 \cdot 300 \cdot 16{,}25 = 1020{,}34$тг ≈ 1020тг.
$З_{э} = З_{э1+} З_{э2+} З_{э3} = 2+51 + 1020 = 1073$ тг.

Таблица 4.3 - Затраты на электроэнергию

Наименование оборудования	Паспортная мощность, кВт	Коэффициент использования мощности	Время работы оборудования для разработки ПП, ч	Цена электроэнергии, тг/кВт*ч	Сумма, тг
Лазерный принтер Samsung ProXpress	0,030	0,700	5,00	16,25	1,71
Модем TP-Link	0,015	0,700	300,00	16,25	51,1875
Ноутбук HP	0,299	0,700	300,00	16,25	1020,3375
ИТОГО затраты на электроэнергию					1073,23

Для расчёта общей суммы затрат на оплату заработной платы $З_{тр}$ воспользуемся формулой 4.3:

$$З_{тр} = \sum_{i=1}^{n} ЧС_i \cdot Т_i, \quad (4.3)$$

где n – количество разработчиков приложения;
$ЧС_i$ - часовая ставка i-го работника, тг;
$Т_i$ – трудоемкость разработки ПП, чел.×ч;
i - категория работника.

Часовая ставка инженера-разработчика составляет 650 (тг/ч), трудоемкость разработки – 372ч.

Результаты расчёта основной заработной платы представлены в таблице 4.4.

Таблица 4.4 – Результаты расчёта затрат основной заработной платы

Категория работника	Квалификация	Трудоемкость разработки ПП, чел.×ч	Часовая ставка, тг/ч	Сумма, тг
Разработчик	Junior Developer	362	650	235300
Project manager	Senior Developer	10	750	7500
ИТОГО затраты на оплату труда				242800

Также необходимо рассчитать отчисления на социальный налог, который составляет 9,5% (согласно статье 485 НК РК) от дохода работника. Социальные отчисления определим по следующей формуле:

$$З_{сзі} = (З_{тр} - З_{ПО}) \cdot 0{,}095,$$

(4.4)

где $З_{ПО}$ – пенсионный отчисления, 10% от общего фонда оплаты труда, тенге. Рассчитаем пенсионные отчисления по следующей формуле:

$$З_{ПО} = З_{тр} \cdot 0{,}1$$

(4.5)

Используя формулы 4.4 и 4.5 получаем:

$$З_{ПО} = 235300 \cdot 0{,}1 = 23530 \text{ тг}$$
$$З_{сзі} = (235300 - 23530) \cdot 0{,}095 = 20118 \text{ тг}$$

Результаты расчёта социальных отчислений представлены в таблице 4.5.

Таблица 4.5 – Результаты расчёта социальных отчислений

Категория работника	ЗП,тг	Соц. налог , %	Сумма СО , тг
Разработчик	235300	9,5	20118,15
Project manager	7500	9,5	641,25
Итоговые отчисления			20759,4

В статью «Амортизация основных фондов» включается сумма амортизационных отчислений от стоимости оборудования и приборов, используемых при разработке ПП. Амортизационные отчисления, рассчитываются по форме, приведенной в таблице 4.6.

Общая сумма амортизационных отчислений определяется по формуле:

$$З_{АМ} = \sum_{i=1}^{n} \frac{Ф_i \times Н_{Ai} \times Т_{НИРi}}{100 \times Т_{Эфi}},$$

(4.6)

где Фі - стоимость і-го оборудования, тг;

НАі - годовая норма амортизации і-го оборудования, %;

ТНИРі - время работы і-го оборудования за весь период выполнения НИР, ч;

ТЭфі - эффективный фонд времени работы і-го оборудования за год, ч/год;

і - вид оборудования;

n - количество оборудования.

Таблица 4.6 - Амортизация основных фондов

Наименование и оборудование ПО	Стоимость оборудования и ПО, тг	Годовая норма амортизации, %	Эффективный фонд времени работы оборудования и ПО, ч/год	Время работы оборудования и ПО для разработки ПП, ч	Сумма, тг
Лазерное МФУ SAMSUNG SL-M2070	55990	40	1970	5,00	56,84
Модем TP-Link	5990	40	1970	300,00	364,87
Ноутбук Lenovo	159990	40	1970	300,00	9745,58
Электронный ключ Microsoft home and student 2019, 1 устройство	49990	40	1970	300,00	3045,08
Жесткий диск	25000	40	1970	300,00	1522,84
СУБД MySQL			Бесплатно		
ПО Visual Studio			Бесплатно		
ИТОГО амортизация основных фондов			14678,38		

В статью «Прочие затраты» включаются расходы на арендную плату, включая коммунальные платежи, расходы на рекламу, канцелярские и прочие хозяйственные расходы.

Стоимость аренды помещения на месяц равна 65000 тг. (в эту сумму включены коммунальные услуги).

Арендная плата рассчитывается по формуле 4.7:

$$АП= Са*S, \quad (4.7)$$

где Са – срок аренды;

S – стоимость аренды за 1 месяц.

АП = 65000 *2 = 130000 тг

Расходы на интернет, месячная оплата которого составляет 3990 тг равны:

$Р_и$= 3990 * 2= 7 980 тг.

Размещение на хостинге и покупка домена составляют 9500 тг;

Прочие хозяйственные расходы составляют 5000 тг;

Прочие затраты = 130000 + 7980 + 9500 + 5000 = 152480 тг.

На основании полученных данных по отдельным статьям в таблице 4.7 приведена смета затрат на разработку ПП

Таблица 4.7 - Смета затрат на разработку ПП

Статьи затрат	Сумма, тг
1. Материальные затраты, в том числе:	
- материалы	55920
- электроэнергия	1073.23
2. Затраты на оплату труда.	242800
3. Отчисления на социальные нужды.	20759,4
4. Амортизация основных фондов.	1522,8
5. Прочие затраты.	157000
ИТОГО по смете	429075,47

4.3 Определение возможной (договорной) цены ПП

Величина возможной (договорной) цены ПП должна устанавливаться с учетом эффективности, качества и сроков ее выполнения на уровне, отвечающем экономическим интересам заказчика (потребителя) и исполнителя.

Договорная цена (ЦД) для прикладных ПП рассчитывается по формуле:

$$Ц_д = З_{нир} \times (1 + \frac{Р}{100}), \quad (4.8)$$

где $З_{НИР}$ - затраты на разработку ПП (из таблицы 4.7), тг;

Р - средний уровень рентабельности ПП. %

$$Ц_д = З_{нир} \times (1 + \frac{Р}{100}) = 429075,47 \times (1 + \frac{20}{100}) = 514890,57 \text{ тг.}$$

Далее определяется цена реализации с учетом налога на добавленную стоимость (НДС), ставка НДС устанавливается законодательно Налоговым Кодексом РК. На 2019 год ставка НДС установлена в размере 12%.

Цена реализации с учетом НДС рассчитывается по формуле:

$$Цр = Цд + Цд \times НДС, \quad (4.9)$$

Цр=Цд + Цд × НДС = 514890,57 + 514890,57*0,12 = 576677,44 тенге.

4.4 Оценка социально - экономических результатов функционирования ПП

К социально-экономическим показателям функционирования информационной системы является обработка таких факторов как:

– качество процесса управления;

– длительность и сроки проектирования программного продукта;

– расходы на реализацию приложения и эксплуатацию программного продукта;

– количеству разработчиков.

Разработка информационной системы для газового предприятия АО Intergaz занимает 2 месяца. Из них 10 дней занимает процесс постановки задачи и проектирование ИС, 23- разработка, 11- тестирование, отладка и внедрение. В процессе разработки участвуют 2 человека- разработчик и Project manager (руководитель). Расходы на реализацию программного продукта составляют 429075,47 тенге. Договорная цена – 514890,57 тенге. Цена реализации- 576677,44 тенге. Сумма эксплуатации программного продукта составляют 10000 тенге в год, т. к. это сумма размещения сайта на хостинге.

Данные вложения обязательно окупятся за счёт значительного социально-экономического результата. Для газового предприятия организацией процесса мониторинга занимаются главный менеджер, а также мастер. Что занимает много времени и требует много трудозатрат. Решить эту проблему можно с помощью внедрения автоматизированной информационной системы (ИС), которая упростит работу сотрудников, сократит затраты времени на сложные операции и увеличит производительность труда.

Средний оклад работника компании составляет 800тг/час, а мастера- 1000 тг/час. Контроль качества работы сотрудников компании занимает минимум 6 часов в неделю. В компании есть 2 отдела. Таким образом, ИС окупается за:

$$\frac{429075{,}47 + 10000}{(800 + 1000 * 2) * 6 * 4} = 6{,}53 \text{ месяцев}$$

Управление качеством работы сотрудников процесса составляет основной предмет управленческой деятельности на всех уровнях системы газовой промышленности.

Социальный эффект программного продукта заключается в том, что разработанная ИС мониторинга должна стать инструментом информационной поддержки, чтобы отображать достоверную информацию о деятельности сотрудников компании, сбора информации для контроля качества и эффективного управления. Пользователями могут являться сотрудники компании, население использующее газ и другие пользователи, которые хотят получить информацию об изменениях газовых тарифов, а также о датах оплаты счета и проведения технического обслуживания.

С учётом такой смешанной аудитории основными целями настоящего дипломного проекта является разработка и внедрение информационной системы для газовой компании АО Intergaz.

Программа должна обеспечивать:

– работу с входными данными;

– получение выходных документов;

– формирование отчетов.

5 Промышленная безопасность

5.1 Трудоёмкость разработки и исходные данные

Производственный шум-это совокупность звуков различной интенсивности и высоты, беспорядочно изменяющихся во времени, возникающих в условиях производства и неблагоприятно воздействующих на организм.

Звук представляет собой волнообразно распространяющийся в упругой среде колебательный процесс. Характеристикой этих волн является звуковое давление-переменное давление, возникающее при прохождении звуковых волн дополнительно к атмосферному давлению, т.е. зоны сгущения и разрежения воздуха. Параметрами звуковой волны являются также период, частота (число полных колебаний в 1 сек) и амплитуда (размах) колебаний. Единицей измерения частоты является герц (Гц)-1 колебание в секунду. Человек воспринимает лишь звуки, имеющие частоту от 20 до 20000 Гц. Ниже 20 Гц находится область инфразвука. Выше 20000 Гц- область ультразвука. Амплитуда колебаний характеризует величину звукового явления. В связи с этим звуковая волна несет определенную механическую энергию, измеряемую в ватах на 1 $см^2$.

Интенсивное шумное воздействие на организм человека неблагоприятно влияет на протекание нервных процессов, способствует развитию утомления, изменениям в сердечно-сосудистой системе и появлению шумовой патологии, среди многообразных проявлений которой ведущим клиническим признаком является медленно прогрессирующее снижение слуха по типу кохлеарного неврита.

В производственных условиях источниками шума являются работающие станки и механизмы, ручные механизированные инструменты, электрические машины, компрессоры, кузнечнопрессовое, подъемно-транспортное, вспомогательное оборудование (вентиляционные установки, кондиционеры) и т.д.

Для данного дипломного проекта необходимо произвести акустический расчет шума, а также меры защиты от воздействия шума на персонал. Источники шума находятся на расстоянии r от расчетной точки, которая расположена на высоте 1,5 м от пола. Определить октавные уровни звукового давления в расчетной точке. Исходные представлены в таблице 5.1.

Данные расчета сравнить с нормируемыми уровнями звукового давления. Определить требуемое снижение звукового давления и рассчитать

параметры кабины наблюдения в качестве меры защиты персонала от действия шума.

Таблица 5.1 - Исходные данные:

Вид оборудования	Процессоры
Количество источников	3
Расстояние от ИШ до РТ, м	r1=8,3; r2= 14 , r3= 10

Продолжение таблицы 5.1

Объем помещения, $м^3$	500
Отношение $B/S_{огр}$	0,8
L_{max}	1,2
Параметры кабины наблюдения	16*10*5
Площадь глухой стены, S_1, $м^2$	80
Площадь глухой стены, S_2, $м^2$	160
Площадь двери, S_3, $м^2$	5
Площадь окна, S_4, $м^2$	4

5.2 Расчетная часть

Для проведения расчетов используются данные приведенные в таблице 5.2. Также на рисунке 5.1 приведена схема расположения источников шума.

Таблица 5.2 - Ориентировочные уровни звукового давления L_P теплоэнергетического оборудования.

Источники шума на ТЭЦ	Среднегеометрические частоты октавных полос, Гц							
	63	125	250	500	1000	2000	4000	8000
генератор	71	61	54	49	45	42	40	38

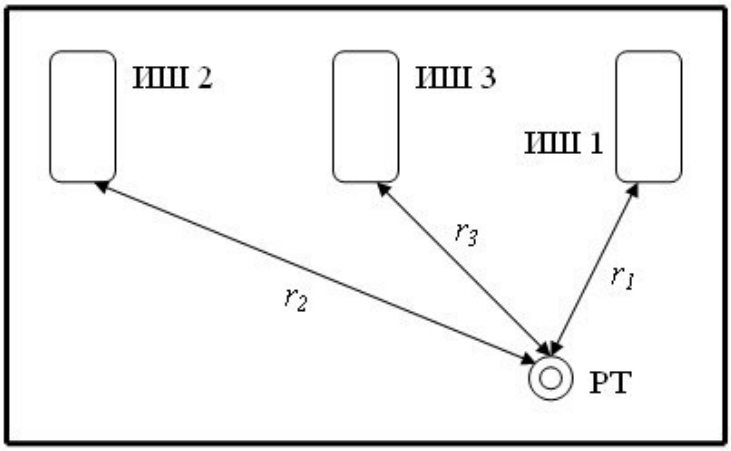


Рисунок 5.1 – Схема расположения источников шума и расчетной точки в помещении.

Октавные уровни звукового давления в расчетной точке помещения, в котором несколько источников шума определяем по формуле

$$L_{общ} = 10 \cdot \lg\left(\sum_{i=1}^{m} \frac{\Delta i \cdot \Phi_i \cdot x_i}{S_i} + \frac{4 \cdot \psi}{B} \sum_{i=1}^{n} \Delta i\right), \quad \text{где} \quad \Delta i = 10^{0.1 \cdot Lpi}$$

(5.1)

L_{pi} – октавный уровень звуковой мощности в Дб, создаваемый i-м источником шума;

$$\Delta i = 10^{0.1 \cdot Lpi} = 10^{0.1*75} = 31{,}62 \cdot 10^6$$

m= 3 – количество источников шума, ближайших к расчетной точке (т.е. источников, для которых ri < 5rmin, где rmin=8,3 – расстояние от расчетной точки, до акустического центра источника);

n = 3 – общее количество источников шума в помещении.

где χ - коэффициент, учитывающий влияние ближайшего акустического поля и принимаемый в зависимости от отношения ri / lmax. Определяется по графику (рисунок 5.2)

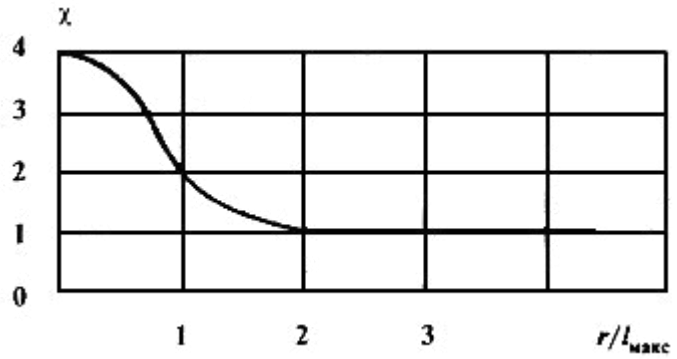


Рисунок 5.2 – График для определения коэффициента χ в зависимости от отношения r к максимальному линейному размеру источника шума l_{max}

ri / lmax : r_1= 8,3/1,2=6,9

$r_{2=}$ 14/1,2=11,7

r_3 = 10/1,2=8,3

Следовательно, χ=1.

Ф=1 – фактор направленности источника шума, безразмерный, определяемый опытным путем;

S – площадь воображаемой поверхности правильной геометрической формы, окружающей источник и проходящий через расчетную точку. Для ИШ, у которых 2· l_{max}<r (в данном случае это условие выполняется для всех ИШ): при расположении ИШ в пространстве $S=4\pi r^2$, на поверхности пола, перекрытия $S=2\pi r^2$.

$S_1=2\pi r_1^2=2\cdot 3{,}14\cdot 8{,}3^2 = 432{,}63$ м^2

$S_2=2\pi r_2^2=2\cdot 3{,}14\cdot 14^2 = 1230{,}88$ м^2

$S_3=2\pi r_3^2=2\cdot 3{,}14\cdot 10^2 = 628$ м^2

B – постоянная помещения, $B=B_{1000}\cdot\mu$,

где B_{1000} – постоянная помещения на среднегеометрической частоте 1000Гц.

Для производственного помещения $B_{1000}=V/20=500/20=25$ м^3
μ- частотный множитель, определяемый по таблице 5.3
$B= 25 \cdot 0,65=16,25$ м^3.

Таблица 5.3 – Значения частного множителя μ

Объем помещения, м^3	Частотный множитель μ на среднегеометрических частотах октавных полос, Гц							
	63	125	250	500	1000	2000	4000	8000
V<200	0,8	0,75	0,70	0,80	1	1,4	1,8	2,5
V=200…1000	0,65	0,62	0,64	0,75	1	1,5	2,4	4,2
V>1000	0,5	0,5	0,55	0,7	1	1,6	3	6

ψ = 0,575 - коэффициент, учитывающий нарушение диффузности звукового поля в помещении, зависящий от отношения $B/S_{огр}$= 0,8. Определяется по графику (рисунок 5.3)

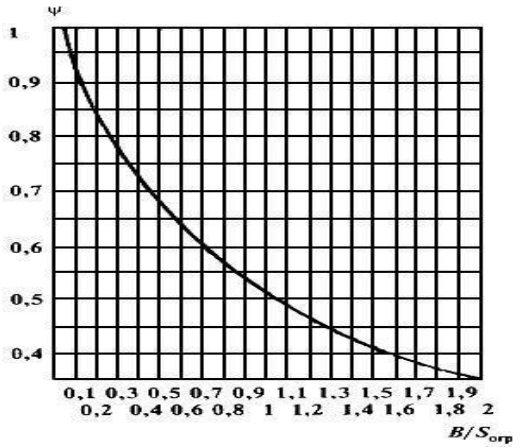


Рисунок 5.3 – График для определения коэффициента ψ в зависимости от $B/S_{огр}$

Используя известные значения $L_{доп}$ указанные в таблице 5.4, определяется требуемое снижение шума $\Delta L_{тр}=L_{общ}-L_{доп}$ значение которого должно быть отрицательным или равно нулю.

Таблица 5.4 - Допустимые уровни шума на рабочих местах

Наименование помещений и рабочих мест	Уровни звукового давления дБ, в октавных полосах среденгеометрических частот, Гц							
	63	125	250	500	1000	2000	4000	8000

Рабочие помещения офисов, банков, кабинеты и рабочие помещения в административных зданиях	71	61	54	49	45	42	40	38

Требуемую звукоизоляцию воздушного шума в дБ ограждающей конструкцией рассчитываем по формуле (5.2)

$$R_{тр}=L_{общ} - 10\cdot lgB +10\cdot lgS_i - L_{доп}+ 10\cdot lgN \quad (5.2)$$

$L_{общ}$- октавные уровни звукового давления в расчетной точке помещения, в котором несколько источников шума, данные приведены в таблице 5.5.

Величину В найдем по формуле (5.3):

$$B=B_{1000}\cdot\mu \quad (5.3)$$

Для кабины наблюдения оператора с объемом V=16*10*5=800 м^3 имеем:

$$B_{1000}=V/10=800/20=40 \text{ м}^3$$

μ - частотный множитель (таблица 2)

$$B=0{,}65\cdot 40=26 \text{ м}^3$$

S_i – площадь рассматриваемой ограждающей конструкции, через который проникает шум, м^2;

Площадь глухой стены, S_1=80 м^2

Площадь глухой стены, S_2=160 м^2

Площадь двери, S_3=5 м^2

Площадь окна, S_4=4 м^2

$L_{доп}$ – допустимый уровень шума.

N= 4 – общее количество ограждающей конструкции, через которое проникает шум.

$R_{тр1}=101 - 10\cdot lg26 + 10\cdot lg80 – 99+ 10\cdot lg4=13$дБ;

$R_{тр2}=101 - 10\cdot lg26 + 10\cdot lg160 – 99+ 10\cdot lg4=16$дБ;

$R_{тр3}=101 - 10\cdot lg26 + 10\cdot lg5 – 99+ 10\cdot lg4=1$дБ;

$R_{тр4}=101 - 10\cdot lg26 + 10\cdot lg4 – 99+ 10\cdot lg4=0$дБ.

Величина	ед. изм	Среднегеометрическая частота октановой полосы, Гц

Таблица 5.5 – Результат расчета уровней звукового давления в расчетной точке

		63	125	250	500	1000	2000	4000	8000
Lp	дБ	71	61	54	49	45	42	40	38
Δi		125892,12	12589,41	25118,64	79432,82	31622,77	15848,93	10000	6309,57
S_1	м^2	432,8486358							
S_2	м^2	1231,50432							
S_3	м^2	628,3185307							
$\frac{\Delta i_i \cdot \chi \cdot \Phi}{S_1}$	$\frac{1}{м^2}$	29084,65	2908,46	580,31	183,51	73,05	36,61	23,10	14,57
$\frac{\Delta i_i \cdot \chi \cdot \Phi}{S_2}$	$\frac{1}{м^2}$	10222,66	1022,26	203,96	64,50	25,67	12,86	8,12	5,12
$\frac{\Delta i_i \cdot \chi \cdot \Phi}{S_3}$	$\frac{1}{м^2}$	20036,42	2003,64	399,77	126,42	50,32	25,2	15,91	10,04
$\sum_{i=1}^{3} \frac{\Delta i_i \cdot \chi \cdot \Phi}{S_i}$	$\frac{1}{м^2}$	59343,74	5934,37	1184,06	374,43	149,06	74,70	47,13	29,74
B_{1000}	м^3	40							
μ		0,65	0,62	0,64	0,75	1	1,5	2,4	4,2
$B = B_{1000}$	м^3	26	24,8	25,6	30	40	60	96	168
$\frac{4\psi}{B} \sum_{i=1}^{n} \Delta i$		12599,74	15333,85	3543,62	1099,74	3667,42	1232,74	390,7	86,38
$L_{общ}$	дБ	61,20	52,02	45,63	40,55	35,81	31,16	26,41	20,64
$L_{доп}$	дБ	99	92	86	83	80	78	76	74
$L_{общ}$	дБ	-37,79	-39,97	-40,36	-42,44	-44,18	-46,83	-49,58	-53,35

Дальнейший расчет приведен в таблице 5.6.

Таблица 5.6 – Результаты акустического расчета

Величина	ед. изм	Среднегеометрическая частота октановой полосы, Гц							
		63	125	250	500	1000	2000	4000	8000

Lp	дБ	71	61	54	49	45	42	40	38
m		0,65	0,62	0,64	0,75	1	1,5	2,4	4,2
B_{1000}	$м^3$	40							
$B = B_{1000} \cdot m$	$м^3$	26	24,8	25,6	30	40	60	96	168
$L_{доп}$	дБ	99	92	86	83	80	78	76	74
$L_{общ}$	дБ	-37,80	-39,98	-40,36	-42,45	-44,18	-46,84	-49,59	-53,35
Rтр1	дБ	-125,89	-120,87	-115,39	-115,17	-115,15	-117,57	-120,36	-124,55
Rтр2	дБ	-122,88	-117,86	-112,38	-112,16	-112,14	-114,56	-117,35	-121,54
Rтр3	дБ	-137,94	-132,91	-127,44	-127,21	-127,19	-129,61	-132,40	-136,59
Rтр4	дБ	-138,91	-133,88	-128,40	-128,18	-128,16	-130,58	-133,37	-137,56

5.3 меры защиты

По сделанным расчетам, следует что, серверная в процессе работы не создает шум выше установленной границы в следствии чего нет необходимости в проведении специализированных мер по подавлению шума. Конструкции позволят осуществить полную звукоизоляцию кабины рабочего помещения.

Борьбу с шумом следует начинать еще при проектировании предприятия, рабочего места, оборудования. В данной работе был произведен акустический расчет. Эти навыки позволят в дальнейшем, столкнувшись с проблемой избыточного шума на производстве, правильно организовать мероприятия по его ликвидации, тем самым обеспечить благоприятные условия для работы персонала, а также избежать проникновения шума за пределы производственной зоны.

Для уменьшения прохождения шума в изолируемое помещение предусматриваются следующие строительно-акустические мероприятия:

а) применение необходимых материалов и конструкций при проектировании перекрытий стен, перегородок, дверей, окон, кабин наблюдений, щитов управления и т.д..

Заключение

Целью дипломного проекта являлась разработка информационной системы для газового предприятия.

Первым этапом дипломного проекта являлась определение цели и задач дипломного проекта. Был проведен анализ существующих систем по автоматизации предметной области.

В первом разделе выполнено бизнес-моделирование процессов организации. И на основе этого построена диаграмма бизнес-вариантов использования представляющая основные направления деятельности сотрудников предприятия и построена диаграмма вариантов использования информационной системы для определенного класса пользователей.

Во втором разделе проведено архитектурное проектирование информационной системы, в котором описана клиент - серверная архитектура. Также было произведено проектирование пользовательского интерфейса, который в свою очередь является важным моментом реализации системы. Данный интерфейс получился, прост для понимания пользователям. В качестве среды разработки программного обеспечения была использована Microsoft Visual Studio 2019 и язык программирования C#.

В третьем разделе дипломного проекта рассмотрена реализация программного продукта. Для тестирования разрабатываемого проекта была выбрана методика тестирования «черного ящика». Эта методика применяется в качестве средства тестирования функционала разрабатываемого программного обеспечения. Для развертывания разрабатываемой системы был составлен план действий. Результатами этапа развертывания системы являются системы сопровождения и поддержки, хранилище документов, где размещаются все версии документов и кода, разработанных в течение проекта.

В четвертом разделе дипломного проекта определялась экономическая эффективность внедрения и работы программного продукта. Были проведены расчеты основных амортизационных фондов.

В пятом разделе дипломного проекта рассмотрена промышленная безопасность внедрения и работы программы. После проведения всех расчётов, было установлено что работа не нуждается в мерах защиты, так как показатели находятся в пределах допустимо нормы.

Результатом дипломного проекта является разработанная автоматизированная информационная система, охватывающая основные бизнес процессы предприятия.

В качестве перспективы развития этой системы можно предложить дальнейшее расширение ее функциональных возможностей и постепенный охват остальных процессов.

Список литературы

1) Общая информация о компании Интергаз [сайт] (http://www.intergas.kz/ru/obwaya-informaciya/).

2) Анализ требований к автоматизированным информационным системам. [Электронный документ] (http://www.INTUIT.ru).

3) Закон Республики Казахстан от 9 января 2012 года № 532-IV «О газе и газоснабжении» (с изменениями и дополнениями по состоянию на 14.04.2019 г.)

4) Visual Studio 2019. Обзор продукта. [Электронный документ] (https://docs.microsoft.com/ru-ru/visualstudio/ide/whats-new-visual-studio-2019?view=vs-2019).

5) Колесников, С.Н. Инструментарий бизнеса: современные методологии управления предприятием. - М.: Издательско-консультационная компания «Статус-Кво 97», 2001. -336с.

6) Бизнес-правила в среде разработки и моделирования. [Электронный документ] (http://www.interface.ru/home.asp?artId=1752).

7) Куперштейн, В. Microsoft Project 2013 в управление проектами− СПб.: БХВ-Питербург, 2014. – 432 с.

8) Варфоломеева, А.О. Информационные системы предприятия: Учебное пособие / А.О. Варфоломеева, А.В. Коряковский, В.П. Романов. - М.: НИЦ ИНФРА-М, 2013. - 283 с.

9) Васильков, А.В. Информационные системы и их безопасность: Учебное пособие / А.В. Васильков, А.А. Васильков, И.А. Васильков. - М.: Форум, 2013. - 528 с.

10) Гвоздева, В.А. Информатика, автоматизированные информационные технологии и системы: Учебник / В.А. Гвоздева. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 544 с.

11) Миков, А.И. Информационные процессы и нормативные системы в IT: Математические модели. Проблемы проектирования. Новые подходы / А.И. Миков. - М.: КД Либроком, 2013. - 256 с.

12) Олейник, П.П. Корпоративные информационные системы: Учебник для вузов. Стандарт третьего поколения / П.П. Олейник. - СПб.: Питер, 2012. - 176 с.

13) Олейник, П.П. Корпоротивные информационные системы.Учебник для вузов. / П.П. Олейник, С.П. Олейник. - СПб.: Питер, 2012. - 176 с.

14) Рыжко, А.Л. Информационные системы управления производственной компанией: Учебник для академического бакалавриата / А.Л. Рыжко, А.И. Рыбников, Н.А. Рыжко. - Люберцы: Юрайт, 2016. - 354 с.

15) Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 352 с.

16) Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 352 с.

17) Пирогов, В.Ю. Информационные системы и базы данных: организация и проектирование: Учебное пособие / В.Ю. Пирогов. - СПб.: БХВ-Петербург, 2009. - 528 с.

18) Волкова В.Н. Теория информационных процессов и систем. — М.: Юрайт, 2016. — 504 с.

19) В продаже "Информационная система предприятия" – конфигурация для работы с корреспонденцией и документами. [Электронный документ] (http://www.cints.ru/news/244/).

20) Вендров, А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. / А.М. Вендров. – [Электронный документ] (http://www.infocity.kiev.ua).

21) Вигерс, К. Разработка требований к программному обеспечению. / К. Вигерс. – М.: Изд.-торг. Дом «Русская Редакция», 2014. – 576с.

22) Колдовский, В. Разработка ПО: модели жизненного цикла. [Электронный документ]. (http://ko-online.com.ua/node/21072) Проверено 25.03.2009.

23) Концептуальное проектирование реляционный баз данных с использованием языка UML. [Электронный документ]. (http://www.interface.ru/home.asp?artId=4517). Проверено 05.02.2009.

24) Полякова, Л.Н. Основы SQL БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий. / Л.Н. Полякова. - ИНТУИТ.ру, - 2007. - 462с.

25) Структурный подход к проектированию ИС. [Электронный документ](http://www.lcard.ru/~nail/database/case/glava2_1.htm).

26) Сущность структурного подхода. [Электронный документ] (http://www.citcite.ru/se/book/spodhod.htm).Проверено 12.02.2009.

27) Трофимов, С. Определение требований к программному обеспечению. [Электронный документ] (http://www.caseclub.ru/articles/treb.html). Проверено 16.03.2009.

28) Уровни требований к программному обеспечению. [Электронный документ] (http://www.atis.ru/DocItem.aspx?groupId_10=8&itemId_10=15). Проверено 24.03.2009.

29) Бекишева А.И. Методические указания к выполнению экономической части дипломной работы для бакалавров специальности 5В0703 - Информационные системы – Алматы: АУЭС; 2013. –24 с.

30) С.Е. Мананбаева, А.С. Бегимбетова Охрана труда. Методические указания к выполнению расчетно-графических работ для студентов - бакалавров всех специальностей. - Алматы: АУЭС, 2013 - 17 с.